



**PENGEMBANGAN SISTEM MANAJEMEN DAN PEMESANAN  
JASA PEKERJA RUMAH TANGGA BERBASIS WEB  
(STUDI KASUS : CV SITIARI)**

**SKRIPSI**

Untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun oleh:

Ary Muhammad Prayoga

NIM: 155150200111118



PROGRAM STUDI TEKNIK INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA  
FAKULTAS ILMU KOMPUTER  
UNIVERSITAS BRAWIJAYA  
MALANG  
2020

## PENGESAHAN

PENGEMBANGAN SISTEM MANAJEMEN DAN PEMESANAN JASA PEKERJA RUMAH  
TANGGA BERBASIS WEB (STUDI KASUS : CV SITIARI)

SKRIPSI

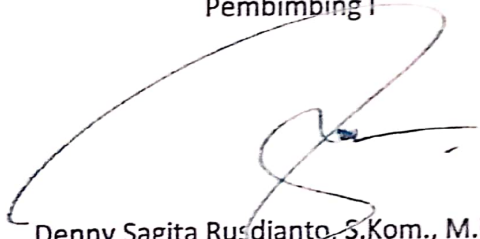
Untuk memenuhi sebagian persyaratan  
memperoleh gelar Sarjana Komputer

Disusun Oleh :  
Ary Muhammad Prayoga  
NIM: 155150200111118

Skripsi ini telah diuji dan dinyatakan lulus pada  
22 Juli 2020

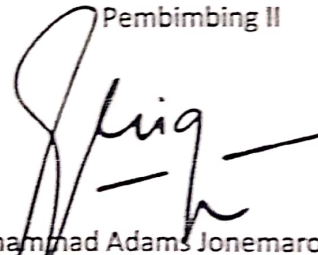
Telah diperiksa dan disetujui oleh:

Pembimbing I



Denny Sagita Rusdianto, S.Kom., M.Kom.  
NIP: 19851124 201504 1 001

Pembimbing II



Eriq Muhammad Adams Jonemaro, S.T., M.Kom.  
NIP: 19850410 201212 1 001

Mengetahui

Ketua Jurusan Teknik Informatika



Achmad Basuki, S.T., M.MG., Ph.D.  
NIP: 19741118 200312 1 002

## PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsurunsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 22 Juli 2020



Ary Muhammad Prayoga

NIM: 155150200111118



## ABSTRAK

**Ary Muhammad Prayoga, Pengembangan Sistem Manajemen Dan Pemesanan Jasa Pekerja Rumah Tangga Berbasis Web (Studi Kasus: CV. Sitiari).**

**Pembimbing: Denny Sagita Rusdianto, S.Kom., M.Kom. dan Eriq Muhammad Adams Jonemaro, S.T., M.Kom**

CV Sitiari merupakan sebuah yayasan penyalur pekerja rumah tangga yang menyalurkan pramuwisma, *babysitter*, dan juga perawat orang lanjut usia. CV Sitiari memiliki 2 cabang yang terletak di Malang. Pada pelaksanaan bisnisnya, ditemukan beberapa masalah yang dirasakan oleh CV. Sitiari. Permasalahan pertama adalah sulitnya melakukan sinkronisasi data para PRT dan majikan dari satu cabang ke cabang lain nya, seluruh data transaksi, pegawai, maupun pelanggan di sinkronisasi antar cabang dengan menggunakan *Microsoft Excel*, *WhatsApp*, maupun SMS. Hal ini membuat sinkronisasi antar cabang menjadi tidak efektif. Permasalahan kedua adalah kesalahan pencatatan dalam pemesanan dan juga pembayaran. Pencatatan pemesanan dan pembayaran yang masih dilakukan secara manual menyebabkan sinkronisasi antara pegawai yang menangani pemesanan dan pegawai yang menangani pembayaran menjadi tidak efektif dan sering terjadi kesalahan. Berdasarkan permasalahan tersebut, maka dilakukanlah pengembangan sistem manajemen dan pemesanan jasa pekerja rumah tangga yang dapat menyelesaikan permasalahan CV. Sitiari. Proses pengembangan perangkat lunak pada penelitian ini dilakukan dengan menggunakan metode *Waterfall*. Tahapan yang dilakukan pada penelitian ini adalah studi literatur, analisis kebutuhan, perancangan sistem, implementasi, pengujian, dan pengambilan kesimpulan dan saran. Sistem ini dikembangkan dengan berbasis web dan bahasa pemrograman PHP. Sistem ini menggunakan *Midtrans* sebagai *payment gateway*. Pengujian unit dilakukan dengan metode *white-box testing* dan menghasilkan 100% data valid dari 3 kasus uji. Pengujian validasi menggunakan metode *black-box testing* dan menghasilkan 100% data valid dari 39 kasus uji. Tes Kompatibilitas juga dilakukan dengan hasil 100% valid.

**Kata kunci:** manajemen, pemesanan, laravel, javascript, *payment gateway*

**ABSTRACT**

**Ary Muhammad Prayoga, Pengembangan Sistem Manajemen Dan Pemesanan Jasa Pekerja Rumah Tangga Berbasis Web (Studi Kasus: CV. Sitiari).**

**Supervisors: Denny Sagita Rusdianto, S.Kom., M.Kom. dan Eriq Muhammad Adams Jonemaro, S.T., M.Kom**

*CV Sitiari is a domestic worker supplier foundation that distributes maids, babysitters, and also carers for the elderly. In doing business, there were several problems that were felt by CV. Sitiari. The first problem is the difficulty of synchronizing data for domestic workers and employers from one branch to another, all transaction data, employees and customers are synchronized between branches using Microsoft Excel, WhatsApp, and SMS. This makes inter-branch synchronization ineffective. The second problem is recording errors in orders and payments. Recording of orders and payments that are still done manually causes synchronization between employees who handle orders and employees who handle payments to be ineffective and errors often occur. Based on these problems, a management system development and ordering services for domestic workers were carried out that could solve the problems on CV. Sitiari. The software development process in this study was carried out using the Waterfall method. The stages carried out in this research are literature study, requirement analysis, system design, implementation, testing, and drawing conclusions and suggestions. This system was developed based on web and the PHP programming language. This system uses Midtrans as a payment gateway. Unit testing was carried out by the white-box testing method and produced 100% valid data from 3 test cases. Validation testing used the black-box testing method and produced 100% valid data from 39 test cases. The compatibility test was also carried out with 100% valid results.*

**Keywords:** *management, order, laravel, javascript, payment gateway*

**DAFTAR ISI**

|   |           |
|---|-----------|
| PENGESAHAN .....  | ii        |
| PERNYATAAN ORISINALITAS .....                             | iii       |
| PRAKATA .....   | iv        |
| ABSTRAK .....   | v         |
| ABSTRACT .....  | vi        |
| DAFTAR ISI .....  | vii       |
| DAFTAR TABEL .....  | x         |
| DAFTAR GAMBAR .....                                       | xv        |
| <b>BAB 1 PENDAHULUAN</b> .....                            | <b>1</b>  |
| 1.1 Latar Belakang .....                                  | 1         |
| 1.2 Rumusan Masalah .....                                 | 2         |
| 1.3 Tujuan .....  | 2         |
| 1.4 Manfaat .....   | 3         |
| 1.5 Batasan Masalah .....                                 | 3         |
| 1.6 Sistematika Pembahasan .....                          | 3         |
| <b>BAB 2 LANDASAN KEPUSTAKAAN</b> .....                   | <b>5</b>  |
| 2.1 Kajian Pustaka .....                                  | 5         |
| 2.2 Pekerja Rumah Tangga .....                            | 7         |
| 2.3 Rekayasa Perangkat Lunak .....                        | 7         |
| 2.4 Pengembangan Perangkat Lunak .....                    | 9         |
| 2.4.1 <i>Software Development Life Cycle (SDLC)</i> ..... | 9         |
| 2.4.2 Pendekatan Berorientasi Objek .....                 | 12        |
| 2.4.3 Pengujian Perangkat Lunak .....                     | 18        |
| 2.5 Teknologi Pengembangan Sistem .....                   | 22        |
| 2.5.1 Midtrans .....                                      | 22        |
| 2.5.2 Laravel .....                                       | 23        |
| 2.5.3 MySQL .....   | 24        |
| 2.5.4 SortSite .....                                      | 24        |
| <b>BAB 3 METODOLOGI</b> .....                             | <b>26</b> |
| 3.1 Studi Literatur .....                                 | 26        |
| 3.2 Rekayasa Kebutuhan Sistem .....                       | 27        |



|   |           |
|---|-----------|
| 3.2.1 Elisitasi dan Analisis Kebutuhan.....           | 27        |
| 3.2.2 Spesifikasi Kebutuhan.....                      | 28        |
| 3.2.3 Validasi dan Verifikasi.....                    | 29        |
| 3.2.4 Manajemen Kebutuhan.....                        | 29        |
| 3.3 Perancangan Sistem.....                           | 29        |
| 3.4 Implementasi Sistem.....                          | 30        |
| 3.5 Pengujian Sistem.....                             | 30        |
| 3.6 Kesimpulan dan Saran.....                         | 31        |
| <b>BAB 4 REKAYASA KEBUTUHAN.....</b>                  | <b>32</b> |
| 4.1 Analisis Kebutuhan.....                           | 32        |
| 4.2 Gambaran Umum Sistem.....                         | 35        |
| 4.3 Identifikasi Aktor.....                           | 38        |
| 4.4 Daftar Kebutuhan Fungsional.....                  | 39        |
| 4.5 Daftar Kebutuhan Non-Fungsional.....              | 50        |
| 4.6 Pemodelan Kebutuhan.....                          | 51        |
| 4.6.1 <i>Use Case Diagram</i> .....                   | 51        |
| 4.6.2 <i>Use Case Scenario</i> .....                  | 53        |
| <b>BAB 5 PERANCANGAN DAN IMPLEMENTASI SISTEM.....</b> | <b>73</b> |
| 5.1 Arsitektur Sistem.....                            | 73        |
| 5.2 Perancangan Sistem.....                           | 73        |
| 5.2.1 Perancangan Arsitektur.....                     | 74        |
| 5.2.2 Perancangan Komponen.....                       | 78        |
| 5.2.3 Perancangan Basis Data.....                     | 81        |
| 5.2.4 Perancangan Antarmuka.....                      | 81        |
| 5.3 Implementasi Sistem.....                          | 87        |
| 5.3.1 Spesifikasi Sistem.....                         | 88        |
| 5.3.2 Implementasi Kode Program.....                  | 89        |
| 5.3.3 Implementasi Basis Data.....                    | 92        |
| 5.3.4 Implemetasi Antarmuka.....                      | 96        |
| <b>BAB 6 PENGUJIAN SISTEM.....</b>                    | <b>99</b> |
| 6.1 Pengujian Unit.....                               | 99        |
| 6.1.1 Pengujian Unit <i>Method</i> submitPesanan..... | 99        |

6.1.2 Pengujian Unit *Method delete* Pekerja..... 104

6.1.3 Pengujian Unit *Method terima* Pekerja..... 107

6.2 Pengujian Integrasi..... 110

6.3 Pengujian Validasi..... 112

6.4 Pengujian Kompatibilitas..... 141

BAB 7 PENUTUP..... 143

7.1 Kesimpulan..... 143

7.2 Saran..... 143

DAFTAR REFERENSI..... 144

LAMPIRAN A HASIL WAWANCARA..... 145

LAMPIRAN B HASIL VALIDASI KEBUTUHAN..... 148





## DAFTAR TABEL

|  |    |
|--|----|
| Tabel 2.1 Kajian Pustaka .....   | 6  |
| Tabel 2.2 Notasi <i>Use Case Diagram</i> .....                                 | 13 |
| Tabel 2.3 Notasi <i>Class Diagram</i> .....                                    | 15 |
| Tabel 2.4 Notasi <i>Sequence Diagram</i> .....                                 | 17 |
| Tabel 2.5 Complexity Number .....  | 20 |
| Tabel 4.1 Identifikasi Aktor .....   | 38 |
| Tabel 4.2 Daftar Kebutuhan Fungsional .....                                    | 39 |
| Tabel 4.3 Daftar Kebutuhan Non-Fungsional .....                                | 51 |
| Tabel 4.2 <i>Use Case Scenario Login</i> .....                                 | 53 |
| Tabel 4.3 <i>Use Case Scenario</i> Daftar Sebagai Pekerja .....                | 53 |
| Tabel 4.6 <i>Use Case Scenario</i> Daftar Sebagai Majikan .....                | 54 |
| Tabel 4.7 <i>Use Case Scenario</i> Masuk Sebagai Admin .....                   | 54 |
| Tabel 4.8 <i>Use Case Scenario</i> Melihat Daftar Pekerja Yang Tersedia .....  | 55 |
| Tabel 4.9 <i>Use Case Scenario</i> Mengajukan Permohonan <i>Resign</i> .....   | 55 |
| Tabel 4.10 <i>Use Case Scenario</i> Melihat Daftar Riwayat Pekerjaan .....     | 56 |
| Tabel 4.11 <i>Use Case Scenario</i> Melihat Daftar Riwayat Penggajian .....    | 56 |
| Tabel 4.12 <i>Use Case Scenario</i> Memesan Jasa Pekerja .....                 | 57 |
| Tabel 4.13 <i>Use Case Scenario</i> Claim Garansi .....                        | 57 |
| Tabel 4.14 <i>Use Case Scenario</i> Perpanjang Masa Kerja PRT .....            | 58 |
| Tabel 4.15 <i>Use Case Scenario Rating</i> Pekerja .....                       | 58 |
| Tabel 4.16 <i>Use Case Scenario</i> Melihat Daftar Riwayat Pemesanan .....     | 59 |
| Tabel 4.17 <i>Use Case Scenario</i> Melihat Daftar Riwayat Claim Garansi ..... | 59 |
| Tabel 4.18 <i>Use Case Scenario</i> Melihat Daftar Admin .....                 | 60 |
| Tabel 4.19 <i>Use Case Scenario</i> Edit Daftar Admin .....                    | 60 |
| Tabel 4.20 <i>Use Case Scenario</i> Hapus Daftar Admin .....                   | 61 |
| Tabel 4.21 <i>Use Case Scenario</i> Melihat Daftar Pekerja .....               | 61 |
| Tabel 4.22 <i>Use Case Scenario</i> Edit Daftar Pekerja .....                  | 61 |
| Tabel 4.23 <i>Use Case Scenario</i> Hapus Daftar Pekerja .....                 | 62 |
| Tabel 4.24 <i>Use Case Scenario</i> Melihat Daftar Majikan .....               | 62 |
| Tabel 4.25 <i>Use Case Scenario</i> Edit Daftar Majikan .....                  | 63 |
| Tabel 4.26 <i>Use Case Scenario</i> Hapus Daftar Majikan .....                 | 63 |



|   |    |
|---|----|
| Tabel 4.27 <i>Use Case Scenario</i> Tambah Admin.....                           | 64 |
| Tabel 4.28 <i>Use Case Scenario</i> Tambah Majikan.....                         | 65 |
| Tabel 4.29 <i>Use Case Scenario</i> Tambah Pekerja.....                         | 65 |
| Tabel 4.30 <i>Use Case Scenario</i> Melihat Riwayat Pekerjaan Pekerja.....      | 66 |
| Tabel 4.31 <i>Use Case Scenario</i> Melihat Riwayat Transaksi Majikan.....      | 66 |
| Tabel 4.32 <i>Use Case Scenario</i> Konfirmasi Pendaftaran Pekerja.....         | 67 |
| Tabel 4.33 <i>Use Case Scenario</i> Melihat Riwayat Pemesanan.....              | 67 |
| Tabel 4.34 <i>Use Case Scenario</i> Penggajian Pekerja.....                     | 68 |
| Tabel 4.35 <i>Use Case Scenario</i> Melihat Riwayat Penggajian Pekerja.....     | 68 |
| Tabel 4.36 <i>Use Case Scenario</i> Konfirmasi Claim Garansi.....               | 68 |
| Tabel 4.37 <i>Use Case Scenario</i> Melihat Riwayat Claim Garansi.....          | 69 |
| Tabel 4.38 <i>Use Case Scenario</i> Konfirmasi Resign Pekerja.....              | 70 |
| Tabel 4.39 <i>Use Case Scenario</i> Blacklist Pekerja.....                      | 70 |
| Tabel 4.40 <i>Use Case Scenario</i> Blacklist Majikan.....                      | 71 |
| Tabel 4.41 <i>Use Case Scenario</i> Mengubah Status Pemesanan.....              | 71 |
| Tabel 4.42 <i>Use Case Scenario</i> Logout.....                                 | 72 |
| Tabel 5.1 <i>Pseudocode</i> submitPesanan.....                                  | 79 |
| Tabel 5.2 <i>Pseudocode</i> deletePekerja.....                                  | 80 |
| Tabel 5.3 <i>Pseudocode</i> terimaPekerja.....                                  | 80 |
| Tabel 5.4 Keterangan Gambar Perancangan Antarmuka <i>Homepage</i> .....         | 82 |
| Tabel 5.5 Keterangan Gambar Perancangan Antarmuka Daftar Majikan.....           | 84 |
| Tabel 5.6 Keterangan Gambar Perancangan Antarmuka Login.....                    | 85 |
| Tabel 5.7 Keterangan Gambar Perancangan Antarmuka <i>Register</i> Majikan.....  | 86 |
| Tabel 5.8 Data Akun Sistem Pemesanan Dan Manajemen Pekerja Rumah<br>Tangga..... | 88 |
| Tabel 5.9 Spesifikasi Perangkat Keras.....                                      | 88 |
| Tabel 5.10 Spesifikasi Perangkat Lunak.....                                     | 89 |
| Tabel 5.11 Spesifikasi Sistem Operasi.....                                      | 89 |
| Tabel 5.12 Implementasi Kode Program <i>Method</i> submitPesanan.....           | 89 |
| Tabel 5.13 Implementasi Kode Program <i>Method</i> deletePekerja.....           | 91 |
| Tabel 5.14 Implementasi Kode Program <i>Method</i> terimaPekerja.....           | 91 |
| Tabel 5.15 Implementasi Data Tabel Users.....                                   | 92 |
| Tabel 5.16 Implementasi Data Tabel Pekerja.....                                 | 92 |



|  |     |
|--|-----|
| Tabel 5.17 Implementasi Data Tabel Detail Pekerja.....                                     | 93  |
| Tabel 5.18 Implementasi Data Tabel Majikan.....  | 93  |
| Tabel 5.19 Implementasi Data Tabel Detail Majikan.....                                     | 94  |
| Tabel 5.20 Implementasi Data Tabel Pemesanan.....  | 94  |
| Tabel 5.21 Implementasi Data Tabel Penggajian.....   | 95  |
| Tabel 5.22 Implementasi Data Tabel Garansi.....  | 95  |
| Tabel 6.1 <i>Pseudocode</i> Algoritme <i>Method</i> submitPesanan.....                     | 99  |
| Tabel 6.2 <i>Source Code Driver</i> pesanTest.....   | 101 |
| Tabel 6.3 Hasil pengujian unit <i>method</i> submitPesanan.....                            | 102 |
| Tabel 6.4 <i>Pseudocode</i> Algoritme <i>Method</i> deletePekerja.....                     | 104 |
| Tabel 6.5 <i>Source Code Driver</i> deleteTest.....  | 106 |
| Tabel 6.6 Hasil pengujian unit <i>method</i> deletePekerja.....                            | 106 |
| Tabel 6.7 <i>Pseudocode</i> Algoritme <i>Method</i> terimaPekerja.....                     | 108 |
| Tabel 6.8 <i>Source Code Driver</i> terimaTest.....  | 109 |
| Tabel 6.9 Hasil pengujian unit <i>method</i> deletePekerja.....                            | 109 |
| Tabel 6.7 Langkah Uji Pengujian Integrasi.....   | 111 |
| Tabel 6.8.1 <i>Source Code Method</i> tambahAdmin.....                                     | 111 |
| Tabel 6.12 <i>Source Code Method</i> stubCreate.....                                       | 112 |
| Tabel 6.13 Pengujian Validasi <i>Use Case</i> Login.....                                   | 113 |
| Tabel 6.14 Pengujian Validasi <i>Use Case</i> Login Alternatif 1.....                      | 113 |
| Tabel 6.15 Pengujian Validasi <i>Use Case</i> Login Alternatif 2.....                      | 113 |
| Tabel 6.16 Pengujian Validasi <i>Use Case</i> Daftar Sebagai Pekerja.....                  | 114 |
| Tabel 6.17 Pengujian Validasi <i>Use Case</i> Daftar Sebagai Pekerja Alternatif<br>1.....  | 115 |
| Tabel 6.18 Pengujian Validasi <i>Use Case</i> Daftar Sebagai Majikan.....                  | 116 |
| Tabel 6.19 Pengujian Validasi <i>Use Case</i> Daftar Sebagai Majikan Alternatif<br>1.....  | 116 |
| Tabel 6.20 Pengujian Validasi <i>Use Case</i> Masuk Sebagai Admin.....                     | 117 |
| Tabel 6.21 Pengujian Validasi <i>Use Case</i> Masuk Sebagai Admin Alternatif<br>1.....     | 117 |
| Tabel 6.22 Pengujian Validasi <i>Use Case</i> Masuk Sebagai Admin Alternatif<br>2.....     | 118 |
| Tabel 6.23 Pengujian Validasi <i>Use Case</i> Melihat Daftar Pekerja Yang<br>Tersedia..... | 119 |



|  |     |
|--|-----|
| Tabel 6.24 Pengujian Validasi <i>Use Case</i> Mengajukan Permohonan Resign .....         | 119 |
| Tabel 6.25 Pengujian Validasi <i>Use Case</i> Melihat Daftar Riwayat Pekerjaan .....     | 119 |
| Tabel 6.26 Pengujian Validasi <i>Use Case</i> Melihat Daftar Riwayat Penggajian .....    | 120 |
| Tabel 6.27 Pengujian Validasi <i>Use Case</i> Memesan Jasa Pekerja .....                 | 120 |
| Tabel 6.28 Pengujian Validasi <i>Use Case</i> Memesan Jasa Pekerja Alternatif 1 .....    | 121 |
| Tabel 6.29 Pengujian Validasi <i>Use Case</i> Claim Garansi .....                        | 121 |
| Tabel 6.30 Pengujian Validasi <i>Use Case</i> Claim Garansi Alternatif 1 .....           | 122 |
| Tabel 6.31 Pengujian Validasi <i>Use Case</i> Perpanjang Masa Kerja PRT .....            | 123 |
| Tabel 6.32 Pengujian Validasi <i>Use Case</i> Rating Pekerja .....                       | 123 |
| Tabel 6.33 Pengujian Validasi <i>Use Case</i> Melihat Daftar Riwayat Pemesanan .....     | 123 |
| Tabel 6.34 Pengujian Validasi <i>Use Case</i> Melihat Daftar Riwayat Claim Garansi ..... | 124 |
| Tabel 6.35 Pengujian Validasi <i>Use Case</i> Melihat Daftar Admin .....                 | 124 |
| Tabel 6.36 Pengujian Validasi <i>Use Case</i> Edit Daftar Admin .....                    | 125 |
| Tabel 6.37 Pengujian Validasi <i>Use Case</i> Hapus Daftar Admin .....                   | 125 |
| Tabel 6.38 Pengujian Validasi <i>Use Case</i> Melihat Daftar Pekerja .....               | 126 |
| Tabel 6.39 Pengujian Validasi <i>Use Case</i> Edit Daftar Pekerja .....                  | 126 |
| Tabel 6.40 Pengujian Validasi <i>Use Case</i> Hapus Daftar Pekerja .....                 | 127 |
| Tabel 6.41 Pengujian Validasi <i>Use Case</i> Hapus Daftar Pekerja Alternatif 1 .....    | 127 |
| Tabel 6.42 Pengujian Validasi <i>Use Case</i> Melihat Daftar Majikan .....               | 128 |
| Tabel 6.43 Pengujian Validasi <i>Use Case</i> Edit Daftar Majikan .....                  | 128 |
| Tabel 6.44 Pengujian Validasi <i>Use Case</i> Hapus Daftar Majikan .....                 | 129 |
| Tabel 6.45 Pengujian Validasi <i>Use Case</i> Tambah Admin .....                         | 129 |
| Tabel 6.46 Pengujian Validasi <i>Use Case</i> Tambah Admin Alternatif 1 .....            | 130 |
| Tabel 6.47 Pengujian Validasi <i>Use Case</i> Tambah Majikan .....                       | 130 |
| Tabel 6.48 Pengujian Validasi <i>Use Case</i> Tambah Majikan Alternatif 1 .....          | 131 |
| Tabel 6.49 Pengujian Validasi <i>Use Case</i> Tambah Pekerja .....                       | 132 |
| Tabel 6.50 Pengujian Validasi <i>Use Case</i> Tambah Pekerja Alternatif 1 .....          | 133 |
| Tabel 6.51 Pengujian Validasi <i>Use Case</i> Melihat Riwayat Pekerjaan Pekerja .....    | 133 |



Tabel 6.52 Pengujian Validasi *Use Case* Melihat Riwayat Transaksi Majikan..... 134

Tabel 6.53 Pengujian Validasi *Use Case* Konfirmasi Pendaftaran Pekerja..... 134

Tabel 6.54 Pengujian Validasi *Use Case* Konfirmasi Pendaftaran Pekerja Alternatif 1 ..... 135

Tabel 6.55 Pengujian Validasi *Use Case* Konfirmasi Pendaftaran Pekerja Alternatif 2 ..... 135

Tabel 6.56 Pengujian Validasi *Use Case* Melihat Riwayat Pemesanan..... 136

Tabel 6.57 Pengujian Validasi *Use Case* Penggajian Pekerja ..... 136

Tabel 6.58 Pengujian Validasi *Use Case* Melihat Riwayat Penggajian Pekerja..... 136

Tabel 6.59 Pengujian Validasi *Use Case* Konfirmasi Claim Garansi ..... 137

Tabel 6.60 Pengujian Validasi *Use Case* Konfirmasi Claim Garansi Alternatif 1 ..... 137

Tabel 6.61 Pengujian Validasi *Use Case* Melihat Riwayat Claim Garansi..... 138

Tabel 6.62 Pengujian Validasi *Use Case* Konfirmasi Resign Pekerja..... 138

Tabel 6.63 Pengujian Validasi *Use Case* Konfirmasi Resign Pekerja Alternatif 1 ..... 139

Tabel 6.64 Pengujian Validasi *Use Case* Blacklist Pekerja..... 139

Tabel 6.65 Pengujian Validasi *Use Case* Blacklist Majikan..... 140

Tabel 6.66 Pengujian Validasi *Use Case* Mengubah Status Pemesanan..... 140

Tabel 6.67 Pengujian Validasi *Use Case Logout* ..... 141

Tabel 6.68 Pengujian Kompatibilitas ..... 141



## DAFTAR GAMBAR

|  |    |
|--|----|
| Gambar 2.1 Model Pengembangan Waterfall.....   | 9  |
| Gambar 2.2 Contoh <i>Use Case Diagram</i> .....  | 15 |
| Gambar 2.3 Contoh <i>Class Diagram</i> .....   | 16 |
| Gambar 2.4 Contoh <i>Sequence Diagram</i> .....  | 18 |
| Gambar 2.5 Notasi <i>Flowgraph</i> .....   | 19 |
| Gambar 2.6 Relasi Komponen MVC.....  | 24 |
| Gambar 3.1 Diagram Alir Penelitian .....   | 26 |
| Gambar 4.1 Struktur Bab Rekayasa Kebutuhan.....  | 32 |
| Gambar 4.2 Proses Bisnis As-Is Pemesanan Pekerja Rumah Tangga .....                              | 33 |
| Gambar 4.3 Proses Bisnis As-Is Pendaftaran Calon pekerja .....                                   | 33 |
| Gambar 4.4 Proses Bisnis As-Is Penggajian Pegawai.....   | 34 |
| Gambar 4.5 Proses Bisnis As-Is Claim Garansi .....   | 35 |
| Gambar 4.6 Proses Bisnis As-Is Resign Pegawai.....   | 35 |
| Gambar 4.7 Proses Bisnis To-Be Pemesanan Pekerja Rumah Tangga.....                               | 36 |
| Gambar 4.8 Proses Bisnis To-Be Pendaftaran Calon pekerja.....                                    | 36 |
| Gambar 4.9 Proses Bisnis To-Be Pembayaran Pekerja .....  | 37 |
| Gambar 4.10 Proses Bisnis To-Be Claim Garansi.....   | 37 |
| Gambar 4.11 Proses Bisnis To-Be Resign Pegawai.....  | 38 |
| Gambar 4.12 Diagram Use Case Sistem Manajemen dan Pemesanan<br>Pekerja Rumah Tangga.....         | 52 |
| Gambar 5.1 Perancangan Arsitektur Dasar Sistem .....   | 73 |
| Gambar 5.2 Struktur Sub-bab Perancangan Sistem.....  | 74 |
| Gambar 5.3 <i>Sequence Diagram</i> Memesan Jasa Pekerja.....                                     | 75 |
| Gambar 5.4 <i>Sequence Diagram</i> Hapus Daftar Pekerja.....                                     | 76 |
| Gambar 5.5 <i>Sequence Diagram</i> Konfirmasi Pendaftaran Pekerja.....                           | 77 |
| Gambar 5.6 <i>Class Diagram</i> Sistem Manajemen dan Pemesanan Jasa<br>Pekerja Rumah Tangga..... | 78 |
| Gambar 5.7 PDM Sistem Manajemen dan Pemesanan Jasa Pekerja<br>Rumah Tangga .....                 | 81 |
| Gambar 5.8 Perancangan Antarmuka <i>Admin Dashboard</i> .....                                    | 82 |
| Gambar 5.9 Perancangan Antarmuka Daftar Majikan .....  | 84 |



Gambar 5.10 Perancangan Antarmuka *Login* ..... 85

Gambar 5.11 Perancangan Antarmuka *Register* Majikan..... 86

Gambar 5.12 Struktur Sub-bab Implementasi Sistem ..... 88

Gambar 5.13 Implementasi Antarmuka *Admin Dashboard*..... 96

Gambar 5.14 Implementasi Antarmuka Daftar Majikan ..... 97

Gambar 5.15 Implementasi Antarmuka *Login*..... 97

Gambar 5.16 Implementasi Antarmuka *Register* Majikan ..... 98

Gambar 6.1 Struktur Bab Pengujian Sistem..... 99

Gambar 6.2 *Flowgraph method* submitPesanan..... 101

Gambar 6.3 Hasil Pengujian Unit *Method* submitPesanan..... 104

Gambar 6.4 *Flowgraph method* deletePekerja ..... 105

Gambar 6.5 Hasil Pengujian Unit *Method* deletePekerja ..... 107

Gambar 6.6 *Flowgraph method* terimaPekerja..... 108

Gambar 6.7 Hasil Pengujian Unit *Method* terimaPekerja..... 110

Gambar 6.6 Diagram Hierarki Pengujian Integrasi *Method* tambahAdmin ..... 111

Gambar 6.7 Hasil Pengujian *Method* tambahAdmin Menggunakan  
stubCreate..... 112

Gambar 6.8 Hasil Pengujian Kompatibilitas..... 142



## BAB 1 PENDAHULUAN

### 1.1 Latar Belakang

Pekerja Rumah Tangga (PRT) merupakan sebuah hal yang sudah tidak asing lagi untuk masyarakat Indonesia. Berdasarkan survei dari International Labour Organization (2017), ada 859.000 PRT di Jawa Barat, 779.000 PRT di Jawa Timur, dan 630.000 PRT di Jawa Tengah. Dalam praktiknya, PRT diharuskan untuk dapat mengerjakan pekerjaan rumah seperti mencuci, membersihkan rumah, memasak, mengasuh bayi, dan juga merawat orang yang sudah lanjut usia dan mengalami keterbatasan fisik. PRT juga merupakan salah satu jenis pekerjaan yang menjadi katup pengaman pengangguran dan pengentasan kemiskinan. PRT juga membantu memberi peluang kepada pihak lain untuk menjalankan pekerjaan sesuai dengan kemampuannya. Jasa PRT bisa didapatkan melalui rekomendasi orang lain yang pernah memakai jasa PRT maupun melalui yayasan yang menyalurkan PRT. Yayasan penyalur PRT adalah sebuah lembaga yang bertugas untuk menampung para PRT dan juga menyalurkan PRT tersebut kepada orang yang membutuhkan jasa PRT. Yayasan penyalur PRT ini juga bertugas untuk menjamin keamanan kedua belah pihak, baik itu dari pihak PRT maupun dari pihak majikan. Yayasan penyalur PRT juga bertugas untuk menyalurkan gaji PRT yang diberikan oleh majikan.

CV Sitiari merupakan sebuah yayasan penyalur PRT yang menyalurkan pramuwisma, *babysitter*, dan juga perawat orang lanjut usia. CV Sitiari didirikan pada tahun 1980 dan mempunyai pegawai yang siap ditugaskan ke berbagai daerah. Sampai dengan saat ini, CV. Sitiari sudah memiliki 2 cabang. Dalam pelaksanaan usahanya, terdapat permasalahan utama yang dialami dan dirasakan oleh pihak CV Sitiari. Permasalahannya adalah sulitnya melakukan sinkronisasi data para PRT dan juga majikan dikarenakan minimnya pencatatan yang dilakukan oleh pihak CV. Sitiari. Permasalahan ini menyebabkan proses sinkronisasi data antar 2 cabang CV Sitiari menjadi kurang efektif, dikarenakan setiap ada proses pemesanan, maka pihak CV. Sitiari harus melakukan konfirmasi pada cabang lainnya melalui telfon atau *Whatsapp*. Permasalahan lain yang dihadapi oleh CV. Sitiari adalah dalam segi pemesanan dan pembayaran. Minimnya pencatatan menyebabkan kesalahan pencatatan pembayaran sering terjadi. Pihak Sitiari sudah melakukan beberapa cara untuk mengatasi permasalahan diatas seperti menggunakan aplikasi seperti *Microsoft Excel* dalam pencatatan data, tetapi solusi itu kurang efektif dikarenakan CV. Sitiari mempunyai 2 cabang dan mengharuskan data pemesanan bisa di distribusikan secara cepat. Dengan penggunaan *Microsoft Excel* data yang di catat pada 1 cabang tidak bisa langsung di distribusikan ke cabang lainnya, sehingga mengharuskan pegawai CV Sitiari harus menelepon cabang lainnya untuk menanyakan ketersediaan pekerja rumah tangga. Masalah lainnya yang muncul dengan penggunaan *Microsoft Excel* adalah admin hanya bisa mengakses data melalui komputer yang ada pada kantor cabang CV. Sitiari. Selain itu, penggunaan *Microsoft Excel* juga berdampak pada kesalahan pencatatan





pembayaran dikarenakan admin yang berbeda pada saat pemesanan dan pembayaran.

Berdasarkan permasalahan yang sudah dijabarkan diatas, maka diperlukan adanya sebuah sistem yang dapat digunakan untuk memanajemen seluruh PRT maupun majikan yang terdaftar, mencatat semua transaksi dari pihak PRT maupun majikan, dan melakukan proses pemesanan dan pembayaran. Sistem tersebut dapat dimanfaatkan oleh pihak CV Sitiari untuk mendistribusikan data antar cabang secara cepat, memonitor transaksi dari pihak PRT maupun majikan, mengurangi resiko kehilangan data, meminimalisir komplain dari pihak PRT maupun majikan, dan juga membantu proses pembayaran agar lebih teratur dan aman.

Berdasarkan permasalahan yang telah disebutkan, maka akan dibangun sebuah sistem manajemen dan pemesanan PRT yang dapat digunakan oleh pihak yayasan untuk membantu proses pemesanan, pembayaran dan manajemen PRT. Sistem ini juga dapat digunakan untuk mencatat transaksi dari pihak PRT maupun majikan. Dalam sistem ini juga akan dilengkapi fitur *rating* untuk PRT maupun majikan, pembayaran *cashless*, pendaftaran PRT secara *online*, pemesanan PRT secara *online*, serta fitur *blacklist* bagi PRT maupun majikan yang sebelumnya punya sebuah *histori track record* yang kurang baik dengan judul "Pengembangan Sistem Manajemen dan Pemesanan Pekerja Rumah Tangga Berbasis Web". Sistem ini akan dibangun menggunakan basis *website* dengan menggunakan metode *waterfall*. Sistem dibangun berbasis *website* bertujuan agar pegawai CV. Sitiari dapat mengakses data pekerja maupun majikan di berbagai perangkat. Selain itu, kompatibilitas merupakan salah satu faktor utama yang mempengaruhi pengadopsian dari *e-commerce* (Alam et al., 2007) sehingga pada bagian kebutuhan non-fungsional akan didefinisikan terkait kompatibilitas sistem dan juga akan dilakukan pengujian kompatibilitas. Dengan adanya sistem ini, diharapkan bisa menyelesaikan permasalahan yang ada di CV. Sitiari.

## 1.2 Rumusan Masalah

1. Bagaimanakah hasil analisis dan spesifikasi kebutuhan sistem untuk manajemen dan pemesanan jasa pekerja rumah tangga tersebut?
2. Bagaimanakah rancangan sistem yang sesuai dengan spesifikasi kebutuhan sistem tersebut?
3. Bagaimanakah hasil implementasi sistem yang sesuai dengan rancangan sistem tersebut?
4. Bagaimanakah hasil pengujian sistem untuk manajemen dan pemesanan jasa pekerja rumah tangga tersebut?

## 1.3 Tujuan

1. Menganalisis dan menyusun spesifikasi kebutuhan sistem untuk manajemen dan pemesanan jasa pekerja rumah tangga.



2. Merancang sistem yang mampu memenuhi hasil dari analisis kebutuhan dan juga spesifikasi kebutuhan.
3. Mengimplementasikan sistem sesuai dengan hasil dari perancangan yang telah dibuat sebelumnya.
4. Menguji sistem yang telah dikembangkan secara fungsional dan non-fungsional (sesuai kebutuhan/masalah yang difokuskan).

#### 1.4 Manfaat

1. Sebagai sarana promosi jasa pekerja rumah tangga.
2. Memudahkan pemilik dari yayasan dalam hal pembayaran, pemesanan, dan promosi.
3. Memudahkan pemilik dari yayasan dalam urusan manajemen pegawai dan majikan.

#### 1.5 Batasan Masalah

1. Sistem dibangun dengan basis *website*.
2. Studi kasus berada pada CV. Sitiari kota Malang.
3. Sistem dikembangkan menggunakan PHP : *Hypertext Processor* sebagai bahasa pemrograman serta HTML dan juga CSS sebagai bahasa untuk *styling*.
4. Sistem dikembangkan dengan menggunakan manajemen basis data *Mysql*.
5. Sistem hanya dapat dijalankan perambah safari, firefox, edge, google chrome, perangkat android, dan perangkat ios.
6. Penelitian menggunakan siklus hidup pengembangan sistem dengan model *waterfall*.

#### 1.6 Sistematika Pembahasan

Sistematika pembahasan yang akan digunakan dalam penelitian ini adalah sebagai berikut :

##### 1. Bab I PENDAHULUAN

Pada tahapan ini akan dijelaskan mengenai latar belakang dilakukannya penelitian, identifikasi masalah yang ada pada penelitian, rumusan masalah penelitian, tujuan dari penelitian itu sendiri, manfaat dari penelitian, ruang lingkup penelitian, dan juga sistematika pembahasan.

##### 2. Bab II LANDASAN KEPUSTAKAAN

Pada tahapan ini akan dijelaskan dasar - dasar teori yang akan menopang penulisan dan juga penelitian ini.

##### 3. Bab III METODE PENELITIAN



Pada tahapan ini akan dijelaskan bagaimana langkah – langkah yang akan diambil dalam mengembangkan sistem ini.

#### 4. Bab IV PERANCANGAN

Pada tahapan ini akan dijelaskan bagaimana perancangan sistem yang dipilih dalam mengembangkan sistem ini.

#### 5. Bab V IMPLEMENTASI

Pada tahapan ini akan dijelaskan deskripsi sistem dari hasil analisa kebutuhan, gambaran dari sistem itu sendiri, dan juga bagaimana perancangan itu sendiri akan diimplementasikan menjadi program komputer.

#### 6. Bab VI PENGUJIAN DAN ANALISIS

Pada tahapan ini akan dijelaskan tentang bagaimana analisis dan juga hasil pengujian dari penelitian yang sebelumnya telah dilakukan.

#### 7. Bab VII PENUTUP

Pada tahapan ini akan dijabarkan kesimpulan yang diperoleh dari peneltian dan juga saran yang bisa diberikan untuk pengembangan lebih lanjut.



## BAB 2 LANDASAN KEPUSTAKAAN

Tahapan ini merupakan tahapan yang nantinya akan menjelaskan dan juga menguraikan tentang konsep, teori, metode, dan juga model yang akan dipergunakan sebagai penopang penulisan dokumen ini. Pada tahapan landasan kepastakaan ini, terdapat landasan - landasan pemikiran dari berbagai macam sumber pustaka yang berkesinambungan dengan teori dan juga metode yang akan menjadi dasar dalam penelitian ini. Beberapa dasar teori yang akan dibahas di dalam landasan kepastakaan diantaranya adalah mengenai pekerja rumah tangga. Kemudian pembahasan lainnya yang akan dibahas adalah pembahasan tentang rekayasa perangkat lunak, lalu teori lainnya yang akan dibahas adalah pengembangan perangkat lunak dengan mengambil model pengembangan perangkat lunak *waterfall*, dan juga membahas mengenai pendekatan berorientasi objek. Hal lainnya akan dibahas adalah teknologi yang akan dipakai untuk membantu dalam hal pengembangan sistem ini yaitu Midtrans, MySQL, Laravel, dan juga SortSite.

### 2.1 Kajian Pustaka

Bab ini akan memaparkan penjelasan perihal penelitian yang telah dilakukan sebelumnya dan menjadi rujukan dalam penelitian ini. Penelitian yang dibahas adalah penelitian yang berada dalam lingkup sistem informasi, *marketplace*, dan *payment gateway*. Terdapat tiga penelitian sebelumnya dengan batas penelitian dalam lima tahun. Penelitian-penelitian tersebut telah dipaparkan pada Tabel 2.1 dengan penjelasan berupa judul penelitian, objek penelitian, dan perbedaan dari penelitian sebelumnya kemudian rencana penelitian sekarang berdasarkan objek penelitian.

Penelitian yang pertama dilakukan oleh Dharma, Satwika, dan Nirmala (2019) memiliki studi kasis di Yayasan Kasih Keluarga. Yayasan ini merupakan lembaga pengasuhan yang berdiri sejak 1995 yang melayani pengasuh bayi, asisten rumah tangga, dan pengasuh lansia. Manajemen pekerja dan juga transaksinya masih dilakukan secara manual menggunakan buku catatan, yang membuat proses bisnis pada yayasan tersebut memerlukan waktu yang lama dan rawan terjadi kerusakan dan kehilangan dokumentasi. Pembangunan system informasi penyedia asisten rumah tangga menjadi solusi atas permasalahan tersebut. Penelitian ini menggunakan SDLC *Waterfall* dan juga menggunakan *framework* Laravel dan juga Vue.

Penelitian yang kedua dilakukan oleh Suryanto (2018) Penerapan *E-Marketplace* Pada Distro Silver Squad. Penerapan aplikasi *E-marketplace* memberikan dampak yang sangat besar. Dengan model transaksi yang lebih fleksibel, ketika customer bisa melakukan pencarian dan pembelian pakaian dari banyak penjual dan melakukan proses jual beli dan terdapatnya fitur yang dapat sangat mempermudah penggunaanya serta didalam implementasi dilapangan dapat dilihat secara langsung proses jual beli dengan internet,



melakukan transaksi melalui media elektronik yang berperan sebagai penghubung dari sisi penjual dan pembeli.

Kemudian penelitian yang terakhir dilakukan oleh Febriyanto, Rahardja, dan Alnabawi (2018) yang menerapkan integrasi pihak ketiga berupa *payment gateway* yaitu midtrans. iPanda adalah tempat menjual hosting dan domain yang bersifat online dan dapat diakses dengan mudah melalui website dengan berbaai perambah. Pada umumnya digunakan untuk mahasiswa tingkat akhir yang sedang dalam proses pengerjaan dan membutuhkan hosting dan domain untuk tugas mereka yang membutuhkan koneksi daring. Berhubung dengan pesatnya transaksi yang terjadi maka sistem harus dapat membantu proses transaksi secara efektif dan efisien. Sehingga diterapkan metode pembayaran yang menggunakan *paymeny gateway* yang bernama Midtrans untuk membantu porses bisnis tersebut. Berbagai metode pembayaran telah difasilitasi oleh midtrans sehingga iPanda memiliki proses bisnis transaksi yang lebih efektif dan efisien.

Tabel 2.1 Kajian Pustaka

| No. | Judul Penelitian  | Objek Penelitian  | Perbedaan   |   |
|-----|---|---|---|---|
|     |   |   | Penelitian Terdahulu  | Rencana Penelitian  |
| 1   | Rancang Bangun Sistem Informasi Penyedia Asisten Rumah Tangga di Yayasan Kasih Keluarga (Dharma, Satwika, dan Nirmala, 2019). | Menerapkan manajemen pekerja dan transaksi secara <i>online</i> pada yayasan kasih keluarga dalam sebuah sistem informasi berbasis web.           | Menerapkan manajemen pekerja dan transaksi secara <i>online</i> pada yayasan kasih keluarga dalam sebuah sistem informasi berbasis web.           | Menerapkan manajemen pekerja dan transaksi secara <i>online</i> pada CV. Sitiari dalam sebuah sistem berbasis web.  |
| 2   | Penerapan E-Marketplace pada Distro Silver Squad (Suryanto, 2018).  | Menerapkan konsep <i>marketplace</i> pada Distro Silver Squad sehingga customer dapat melakukan pencarian dan pembelian barang dari sekian banyak | Menerapkan konsep <i>marketplace</i> pada Distro Silver Squad sehingga customer dapat melakukan pencarian dan pembelian barang dari sekian banyak | Menerapkan konsep <i>marketplace</i> pada CV. Sitiari sehingga customer dapat melakukan pencarian dan memesan pekerja rumah tangga secara <i>online</i> . |



|   | penjual.   | penjual.   | penjual.   |   |
|---|--|--|--|---|
| 3 | Penerapan Midtrans sebagai Sistem Verifikasi Pembayaran pada Website iPanda (Febriyanto, Rahardja and Alnabawi, 2018). | Penggunaan midtrans dalam mekanisme transaksi pada website iPanda. | Penggunaan midtrans dalam mekanisme transaksi pada website iPanda. | Penggunaan midtrans dalam mekanisme transaksi pada aplikasi pemesanan jasa PRT. |

## 2.2 Pekerja Rumah Tangga

Pekerja rumah tangga (PRT) merupakan seorang tenaga kerja yang melakukan pekerjaan - pekerjaan pada rumah tangga dan mendapatkan upah dalam jumlah tertentu (Perda DKI No. 6, 2004). Seorang PRT biasanya bekerja pada suatu rumah tangga dengan tujuan untuk membantu menyelesaikan pekerjaan rumah tangga yang tidak dapat diselesaikan karena alasan tertentu. Seperti, suami dan istri terlalu sibuk dengan pekerjaannya sehingga mereka tidak mempunyai waktu untuk mengurus pekerjaan rumah tangga dan juga alasan lainnya.

Seorang PRT biasanya dipekerjakan oleh sebuah yayasan yang nantinya akan menawarkan jasa PRT tersebut kepada orang yang membutuhkan jasa PRT. Tetapi, tidak sedikit PRT yang menawarkan jasanya secara langsung kepada suatu keluarga tanpa melalui yayasan. Jika seseorang membutuhkan jasa PRT, maka cara paling mudah untuk menemukan jasa PRT adalah dengan mendatangi yayasan PRT terdekat. Setelah itu, pihak yayasan akan segera mencari jasa PRT yang sesuai dengan kriteria yang diinginkan contoh (bisa memasak, bisa menginap di rumah, hanya bekerja di siang hari, dan lainnya).

Tugas PRT, selain diharuskan untuk mengerjakan pekerjaan rumah tangga seperti mencuci, menyiram tanaman, membersihkan halaman rumah, dan memasak, tidak jarang diberikan tugas tambahan untuk mengasuh anak-anak kecil (*babysitter*), karena majikan yang sibuk bekerja untuk mencari nafkah untuk memenuhi kebutuhan keluarganya (Wahy, 2012).

## 2.3 Rekeyasa Perangkat Lunak

Pengertian dari rekeyasa perangkat lunak yaitu sebuah penggunaan prinsip – prinsip penting dari rekeyasa agar pengguna mendapatkan sebuah perangkat lunak yang dapat diandalkan dan bekerja secara efisien pada mesin yang sesungguhnya secara ekonomis (Pressman, 2010).

Menurut Pressman (2010), tahapan yang harus dilakukan dalam merekeyasa perangkat lunak adalah:

- a. Memahami Masalah



Memahami sebuah masalah adalah bagian vital dalam rekayasa perangkat lunak. Ada beberapa hal yang harus dimengerti agar proses memahami masalah menjadi lebih mudah, yaitu:

- Tentukan siapa saja pemangku kebutuhan yang akan menggunakan perangkat lunak.
- Tentukan data, fungsi, dan juga fitur yang diperlukan untuk memecahkan masalah secara baik dan benar.
- Representasikan masalah yang lebih kecil, agar lebih mudah dipahami.
- Membuat model analisis.

#### b. Merencanakan Solusi

Setelah masalah yang akan diselesaikan sudah dipahami, langkah selanjutnya adalah membuat desain.

- Mencari perangkat lunak yang menyelesaikan masalah yang serupa.
- Mencari elemen dari solusi sebelumnya yang dapat digunakan kembali.
- Membuat model desain.

#### c. Menjalankan Rencana yang Telah Dibuat

Desain yang telah dibuat sebelumnya berfungsi sebagai pedoman untuk membangun perangkat lunak. Agar dapat membantu ketika terjadi suatu masalah dalam proses pengembangan perangkat lunak.

- Memastikan solusi yang dibangun sesuai dengan rencana yang telah dibuat.
- Memastikan *source code* sesuai dengan desain yang telah dibuat sebelumnya.
- Memastikan seluruh fungsi telah berjalan dengan baik dan benar.

#### d. Memeriksa Hasil

Tahapan ini bertujuan untuk memastikan solusi yang dibuat sudah di uji coba dan juga untuk menemukan *error* sebanyak mungkin.

- Menentukan metode uji coba yang sesuai.
- Memastikan perangkat lunak telah memenuhi kebutuhan dari seluruh *stakeholder*.



## 2.4 Pengembangan Perangkat Lunak

Pembahasan yang akan dibahas mengenai pengembangan perangkat lunak adalah model pengembangan perangkat lunak yang akan digunakan dan juga pendekatan dengan orientasi objek.

### 2.4.1 Software Development Life Cycle (SDLC)

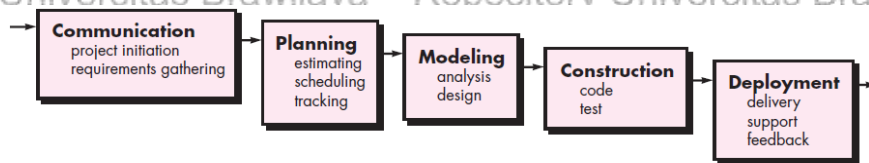
Software Development Life Cycle (SDLC) adalah koleksi dari berbagai langkah yang diikuti untuk sistematis pengembangan, desain, dan pemeliharaan proyek perangkat lunak dan memastikan bahwa semua persyaratan pengguna dipenuhi dengan konsumsi sumber daya paling sedikit (Barjtya, Sharma, & Rani, 2017). Metodologi ini membantu untuk memberikan produk berkualitas tepat waktu dan sesuai kebutuhan klien. Model SDLC ini cocok untuk jenis proyek tertentu, satu model tidak dapat diterapkan untuk semua proyek perangkat lunak karena setiap proyek memiliki persyaratan yang berbeda.

#### 2.4.1.1 Model Pengembangan Waterfall

Pemilihan model pengembangan perangkat lunak akan disesuaikan dengan hasil analisis dari masalah yang dihadapi. Model *waterfall* dipilih karena spesifikasi-spesifikasi kebutuhan telah terdefinisi dengan baik dan telah stabil.

Model *waterfall* kerap kali disebut dengan siklus hidup klasik (*classic life cycle*) dimana model ini mempunyai pendekatan yang sistematis dan berurutan pada pengembangan perangkat lunak. Siklus ini dimulai dengan spesifikasi kebutuhan pengguna (*communication*), lalu berlanjut melalui tahapan-tahapan perencanaan (*planning*), diikuti dengan tahap pemodelan (*modelling*), konstruksi (*construction*), dan tahap terakhir yaitu penyerahan sistem ke para pengguna (*deployment*), yang diakhiri dengan dukungan berkelanjutan untuk perangkat lunak yang dihasilkan (Pressman, 2010).

Kerangka alur pengembangan ini bertujuan untuk memahami alur pengembangan yang akan dilakukan pada penelitian ini. Diagram alir runtutan pengembangan dalam penelitian ini akan dijabarkan pada Gambar 2.1.



Gambar 2.1 Model Pengembangan Waterfall

Sumber: Pressman (2010)

Menurut Pressman (2010), pengembangan dengan model *waterfall* mempunyai tahapan sebagai berikut:





## 1. Komunikasi

Pengembang beserta pengguna bekerja sama untuk mendefinisikan kebutuhan sistem dan mengidentifikasi seluruh garis besar sistem. Tahapan dalam proses komunikasi diantaranya adalah rekayasa kebutuhan. Rekayasa kebutuhan sistem merupakan sebuah proses yang membangun jembatan antara tahapan desain dan tahapan pembangunan (Pressman, 2010). Rekayasa kebutuhan menyediakan sebuah mekanisme yang sesuai agar kita sebagai pihak *developer* mengerti kebutuhan apa saja yang dibutuhkan oleh *stakeholder*, analisis kebutuhan, menilai kelayakan dari sebuah sistem, negosiasi solusi yang bisa diterima akal, menspesifikasikan solusi dari suatu permasalahan yang tidak ambigu, memvalidasi seluruh spesifikasi kebutuhan, dan manajemen kebutuhan hingga berubah menjadi sebuah fungsi. Rekayasa kebutuhan dilakukan dengan menerapkan OOA (Object Oriented Analysis). Tahapan yang ada pada rekayasa kebutuhan dimulai dari analisis kebutuhan, lalu dilanjutkan dengan spesifikasi kebutuhan yang telah didapat, lalu melakukan validasi dan verifikasi dari kebutuhan, dan melakukan manajemen kebutuhan. Penjelasan lebih lanjut mengenai tahapan rekayasa kebutuhan adalah sebagai berikut :

### - Elisitasi dan Analisis Kebutuhan

Tahap elisitasi merupakan tahapan yang dijalankan untuk mendefinisikan kebutuhan apa saja yang dibutuhkan oleh user. Analisis dilakukan dengan melihat pada domain masalah dengan tujuan untuk memproduksi sebuah model konseptual informasi yang ada di daerah yang sedang dianalisis. Elisitasi dilakukan dengan menggunakan teknik wawancara dengan *stakeholder* yang berkepentingan.

Hasil yang didapatkan setelah melakukan elisitasi dan analisis kebutuhan adalah daftar aktor, daftar kebutuhan, dan skenario dari sistem yang akan dibuat.

### - Spesifikasi Kebutuhan

Di dalam konteks sistem dengan basis komputer (dan perangkat lunak), spesifikasi mempunyai arti yang berbeda - beda. Spesifikasi dapat dituangkan kedalam sebuah dokumen, spesifikasi juga bisa dituangkan kedalam model grafik, kumpulan skenario penggunaan, model matematika formal, *prototype*, atau kombinasi dari semuanya. Spesifikasi kebutuhan berguna sebagai pondasi untuk rekayasa perangkat keras, rekayasa perangkat lunak, rekayasa basis data, dan juga rekayasa manusia. Spesifikasi kebutuhan menjelaskan fungsi dan performa sebuah sistem dan juga batasan dari sistem itu sendiri. Hasil yang didapatkan dari tahapan spesifikasi adalah spesifikasi kebutuhan yang menjelaskan sebuah kebutuhan secara spesifik.

### - Validasi dan Verifikasi

Tahapan ini merupakan sebuah tahapan untuk memeriksa spesifikasi dan memastikan seluruh kebutuhan sudah dituliskan tanpa



ada ambigu. Tahapan ini juga bertujuan untuk memastikan ketidakkonsistenan, kelalaian, dan error sudah terdeteksi dan terbenahi. Pihak yang melakukan validasi dan verifikasi adalah seluruh *stakeholder* yang berkepentingan. Teknik yang digunakan untuk validasi dan verifikasi adalah menggunakan kuisioner *Requirement Validation Checklist*.

#### - Manajemen Kebutuhan

Kebutuhan pada sistem dengan basis komputer selalu berubah, dan keinginan untuk mengubah kebutuhan tetap ada sepanjang umur sistem. Manajemen kebutuhan merupakan serangkaian aktivitas yang membantu tim proyek mengidentifikasi, mengendalikan, dan juga melacak kebutuhan dan perubahan terhadap kebutuhan kapan saja saat proyek berlangsung. Diagram *use case* merupakan hasil yang didapatkan pada tahapan manajemen kebutuhan.

### 2. Perencanaan

Pengembang membuat penjadwalan pembuatan sistem yang akan digunakan sebagai pedoman dalam pengembangan perangkat lunak.

### 3. Pemodelan

Pengembang melakukan pemodelan dari kebutuhan yang sudah didapatkan pada proses sebelumnya dengan tujuan untuk memudahkan proses selanjutnya yaitu proses konstruksi. Tahapan pemodelan juga dimaksudkan untuk menentukan solusi teknis atau rancangan/*computer-based* dimana solusi ini berdasarkan kebutuhan sistem yang telah diidentifikasi pada proses analisis. Pemodelan dilakukan dengan menggunakan OOD (*Object Oriented Design*).

### 4. Konstruksi

Dalam tahap ini, pengembang melakukan penulisan kode program (*coding*) lalu melakukan proses pengujian terhadap perangkat lunak yang sudah dikodekan. Tahapan ini bertujuan untuk merealisasikan desain yang sudah dibuat sebelumnya kedalam kode program, dan melakukan pengujian untuk menentukan apakah program yang dibuat sudah sesuai dengan kebutuhan user.

### 5. Penyerahan perangkat lunak kepada pengguna

Setelah perangkat lunak telah diuji dan sesuai dengan kebutuhan, maka perangkat lunak akan dikirim kepada pengguna (*delivery*), lalu pengembang juga memberikan dukungan terhadap pengguna ketika terdapat sebuah kesalahan pada perangkat lunak (*maintenance*), yang diakhiri dengan umpan balik (*feedback*) dari pengguna untuk pengembang.



## 2.4.2 Pendekatan Berorientasi Objek

Pendekatan ini merupakan sebuah metode pendekatan yang nantinya akan mendefinisikan objek apa saja yang ada pada suatu masalah (Sommerville, 2011). Keuntungan penggunaan metode ini adalah jika nantinya terdapat suatu perubahan yang terjadi pada program, maka akan lebih mudah untuk melakukan perubahan tersebut. Keuntungan lainnya dalam penggunaan pendekatan dengan orientasi objek ini adalah lebih konsisten dalam hal analisis dan juga perancangan. Dan keuntungan yang terakhir adalah dapat memangkas waktu pengembangan. Pendekatan ini mempunyai beberapa tahapan, tahapan yang pertama merupakan tahapan analisis kebutuhan, tahapan berikutnya adalah perancangan sistem yang dilakukan setelah menyelesaikan tahapan analisis, tahapan ke 3 adalah implementasi sistem yang dilakukan setelah perancangan selesai, dan tahapan terakhir adalah pengujian (Pressman, 2010).

### 2.4.2.1 Konsep Berorientasi Objek

Konsep yang ada pada pendekatan ini merupakan memecah suatu domain masalah ke dalam bentuk objek yang nyata. Seluruh objek yang ada pada domain masalah nantinya akan diberikan atribut yang bertujuan untuk mendefinisikan objek tersebut. Dari pemberian atribut tersebut nantinya objek – objek tersebut akan dibagi kedalam klas dan subklas. Setiap klas dan juga sub klas akan diberikan operasi maupun atribut yang nantinya akan menjelaskan perilaku dari klas maupun sub klas itu sendiri. Pendekatan ini mempunyai beberapa konsep diantaranya adalah *encapsulation*, *polymorphism*, dan juga *inheritence* (Pressman, 2010).

Proses berorientasi objek yang dilakukan pada saat pengembangan perangkat lunak adalah analisis berorientasi objek, lalu dilanjutkan dengan perancangan berorientasi objek, setelah itu akan dilakukan pemrograman berorientasi objek, dan diikuti oleh pengujian berorientasi objek (Pressman, 2010). Analisis berorientasi objek merupakan sebuah tahapan yang memvisualisasikan suatu objek dari sistem dari berbagai sudut seperti, struktur, fungsi, dan juga interaksi dengan objek. Setelah kebutuhan didapatkan pada proses analisis berorientasi objek, maka kebutuhan tersebut akan dimodelkan menggunakan *use case diagram*. Perancangan berorientasi objek adalah sebuah proses untuk mengubah model analisis yang sudah didapatkan ke dalam model perancangan. Pemrograman berorientasi objek adalah perubahan model perancangan ke dalam bahasa pemrograman. Pengujian berorientasi objek adalah proses pengujian, diantaranya adalah, pengujian unit, pengujian integrasi, dan pengujian validasi.

### 2.4.2.2 Pemodelan Berorientasi Objek

Tahapan analisis dan juga perancangan pada pemodelan dengan orientasi objek dapat divisualisasikan melalui diagram UML (*Unified Modelling Language*) (Pressman, 2010). Diagram tersebut merupakan diagram dan juga bahasa yang sudah menjadi acuan dalam industri untuk sarana visualisasi, perancangan, dan

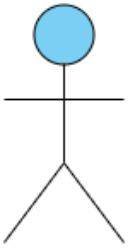



juga mendokumentasi perangkat lunak. UML disajikan dalam berbagai jenis dan memiliki peranan masing-masing.

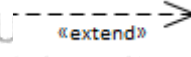
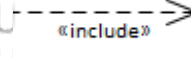
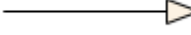
**A. Use Case Diagram**

Diagram ini merupakan sebuah diagram yang diperuntukkan untuk memvisualisasikan interaksi seorang aktor pada sebuah sistem (Whitten and Lonnie D. Bentley, 2007). Diagram ini memvisualisasikan interaksi antara sistem dan juga pengguna. Oleh karena itu, secara grafis memvisualisasikan siapa yang akan menggunakan sistem dan juga dengan cara apa pengguna berinteraksi dengan sistem. Notasi yang ada di dalam *use case diagram* dijelaskan pada Tabel 2.1.

**Tabel 2.2 Notasi Use Case Diagram**

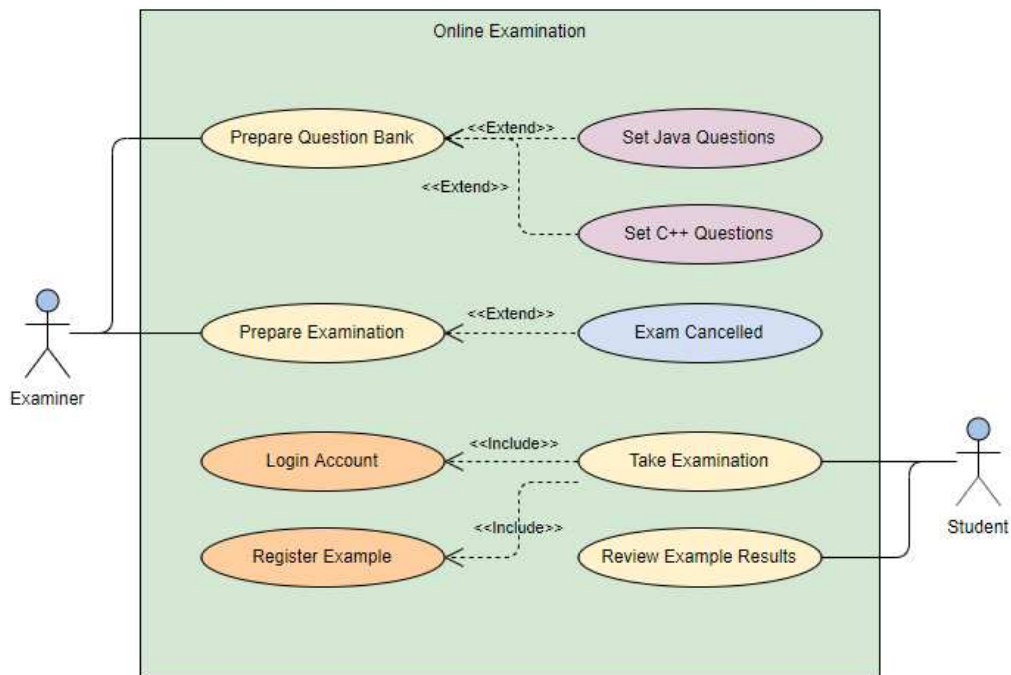
| Notasi  | Deskripsi   |
|---|---|
| <p data-bbox="296 797 375 826">Aktor</p>  <p data-bbox="448 1115 501 1137">Actor</p>                | <p data-bbox="675 797 1348 1373"><i>Use case diagram</i> memiliki pengguna eksternal yang diberi nama aktor/pelaku. Aktor memulai kegiatan sistem, yaitu sebuah <i>use case</i>, dengan harapan mendapatkan beberapa tugas bisnis yang akan menghasilkan sesuatu yang bisa diukur. Aktor mempunyai peran layaknya seorang pengguna yang akan berinteraksi dengan sistem dan tidak menggambarkan individu maupun nama pekerjaan tunggal. Dalam prakteknya, seorang aktor tidak harus seorang manusia, aktor mungkin saja berupa perusahaan, sistem lainnya, alat eksternal seperti sensor suhu, atau bisa juga konsep waktu. Seorang aktor divisualisasikan secara grafis sebagai <i>stickman</i> dengan nama peran yang diperankan oleh aktor tersebut.</p> |
| <p data-bbox="296 1397 421 1426"><i>Use Case</i></p>  <p data-bbox="421 1525 523 1547">UseCase</p> | <p data-bbox="675 1397 1348 1861">Notasi ini memvisualisasikan seluruh fungsi sistem melalui sudut pandang pengguna eksternal dan dalam sebuah terminologi dan juga cara yang mereka pahami. <i>Use case</i> adalah sebuah hasil dari penyusunan kembali lingkup fungsionalitas sistem menjadi banyak pernyataan fungsionalitas sistem yang lebih kecil. <i>Use case</i> divisualisasikan secara grafis dengan simbol elips horizontal dengan nama <i>use case</i> di dalam elips tersebut. Sebuah <i>use case</i> menggambarkan satu runtutan kegiatan, tujuan tunggal dari sistem, dan interaksi aktor untuk mencapai tujuan.</p>   |
| <p data-bbox="296 1883 580 1912">Asosiasi / <i>association</i></p>  | <p data-bbox="675 1883 1348 1989">Notasi asosiasi ini menggambarkan hubungan antara seorang aktor dan satu <i>use case</i>. Asosiasi digambarkan dengan sebagai satu garis lurus yang</p>   |



|   |  |
|---|--|
|   | <p>menggabungkan aktor dengan <i>use case</i>. Asosiasi mengindikasikan interaksi antara <i>use case</i> dan aktor. Ketika satu aktor dikaitkan dengan satu <i>use case</i>, maka bisa dikatakan aktor tersebut berkomunikasi dengan <i>use case</i>.</p>  |
| <p>Ekstensi / <i>extend</i></p>                | <p>Pada sebuah sistem, beberapa fungsi sering kali mempunyai tahapan yang rumit dan kompleks. Karena tahapan yang rumit itu menjadikan logika dari fungsi itu menjadi sulit dipahami. Untuk mempermudah memahami fungsi tersebut, dibuatlah fungsi baru yang akan menjabarkan proses rumit dari suatu fungsi ke dalam fungsi baru. Untuk menggambarkan fungsi yang rumit beserta kepanjangan fungsinya maka digunakanlah notasi ekstensi</p> |
| <p><i>Includes</i></p>                         | <p>Pada sebuah sistem, beberapa fungsi seringkali membutuhkan fungsi lainnya untuk dapat menyelesaikan suatu pekerjaan. Maka dari itu digunakanlah notasi <i>include</i> yang menjelaskan fungsi mana yang membutuhkan fungsi lainnya untuk menyelesaikan suatu pekerjaan.</p>   |
| <p>Generalisasi / <i>Generalization</i></p>  | <p>Pada sebuah sistem, para aktor seringkali mempunyai hubungan umum-khusus, yang dapat diartikan aktor satu lebih umum dari aktor lainnya. Untuk memvisualisasikan hubungan tersebut, digunakanlah notasi generalisasi.</p>   |

Sumber: Whitten and Lonnie D. Bentley (2007)

Pada Gambar 2.2 dapat dilihat contoh dari diagram *use case* sesuai dengan penjelasan yang sudah dijabarkan sebelumnya.



**Gambar 2.2 Contoh Use Case Diagram**

Sumber: Visual Paradigm (2019)


### 2.4.2.3 Class Diagram

Diagram ini merupakan sebuah visualisasi grafis yang akan menampilkan interaksi dari objek pada kelas – kelas di dalam sistem yang berinteraksi satu sama lain. Diagram ini juga berfungsi untuk memvisualisasikan struktur objek dari suatu sistem (Whitten and Lonnie D. Bentley, 2007). Tabel 2.2 akan menjelaskan notasi yang ada di dalam diagram kelas.

**Tabel 2.3 Notasi Class Diagram**

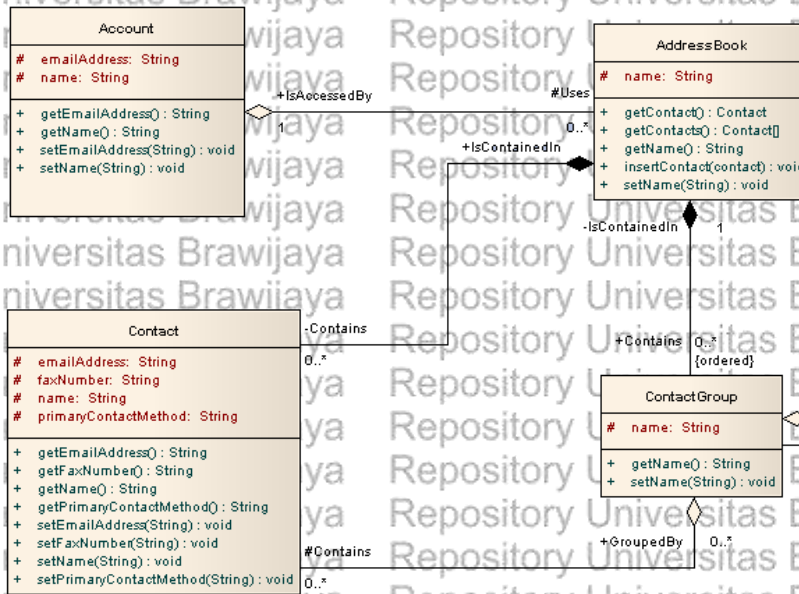
| Notasi  | Deskripsi   |
|---|---|
| <p><i>Class / Kelas</i></p> <div style="border: 1px solid black; padding: 5px; margin: 10px 0;"> <p style="text-align: center;"><b>Nama Kelas</b></p> <p>- Attribute_1: int<br/>+ Attribute_2: double<br/><br/>+ methods_1(): void<br/>+ methods_2(): void</p> </div> | <p>Kelas merupakan sebuah notasi yang akan menjabarkan sebuah kelas yang ada di dalam sistem. Notasi ini dibagi ke dalam 3 bagian. Bagian atas yang merupakan nama dari kelas tersebut. Bagian tengah yang merupakan penjabaran dari atribut yang dimiliki oleh kelas tersebut seperti tipe data maupun jenis <i>modifier</i>. Dan bagian bawah yang merupakan penjabaran dari metode, <i>modifier</i>, dan <i>return type</i> yang ada pada kelas terkait.</p> |
| <p><i>Association / Asosiasi</i></p> <p>_____ }<br/>1..*            1</p>   | <p>Asosiasi merupakan sebuah notasi yang memvisualisasikan hubungan antara kelas yang umum. Notasi ini juga menggambarkan tipe hubungan antar kelas yaitu <i>one-to-one</i>, <i>one-to-</i></p>   |



|  |  |
|--|--|
|  <p><b>Composition / Komposisi</b></p>        | <p><i>many</i>, dan <i>many-to-many</i>.</p> <p>Komposisi merupakan sebuah notasi yang memvisualisasikan hubungan antar kelas yang membutuhkan kelas lainnya dan tidak bisa berdiri sendiri.</p> |
|  <p><b>Aggregations / Agregasi</b></p>        | <p>Agregasi merupakan kebalikan dari komposisi yang menjelaskan hubungan antar kelas yang membutuhkan kelas lainnya tetapi bisa berdiri sendiri.</p>   |
|  <p><b>Generalizations / Generalisasi</b></p> | <p>Generalisasi merupakan sebuah notasi yang memvisualisasikan bentuk umum-khusus dari kelas lainnya.</p>  |

Sumber: Whitten and Lonnie D. Bentley (2007)

Berikut adalah contoh *class diagram* dari penjelasan komponen *class diagram* diatas, yang dapat dilihat pada Gambar 2.3.



**Gambar 2.3 Contoh Class Diagram**

Sumber: Sparx System (2017)



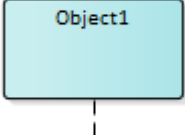

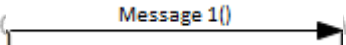
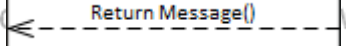
**2.4.2.4 Sequence Diagram**

Diagram ini merupakan sebuah diagram yang memvisualisasikan interaksi antar objek dengan menggunakan pesan. Interaksi yang di visualisasikan merupakan sebuah eksekusi dari sebuah fungsi pada sistem (Whitten and



Lonnie D. Bentley, 2007). Pada Tabel 2.3, akan dijelaskan notasi yang ada pada diagram *sequence*

**Tabel 2.4 Notasi Sequence Diagram**

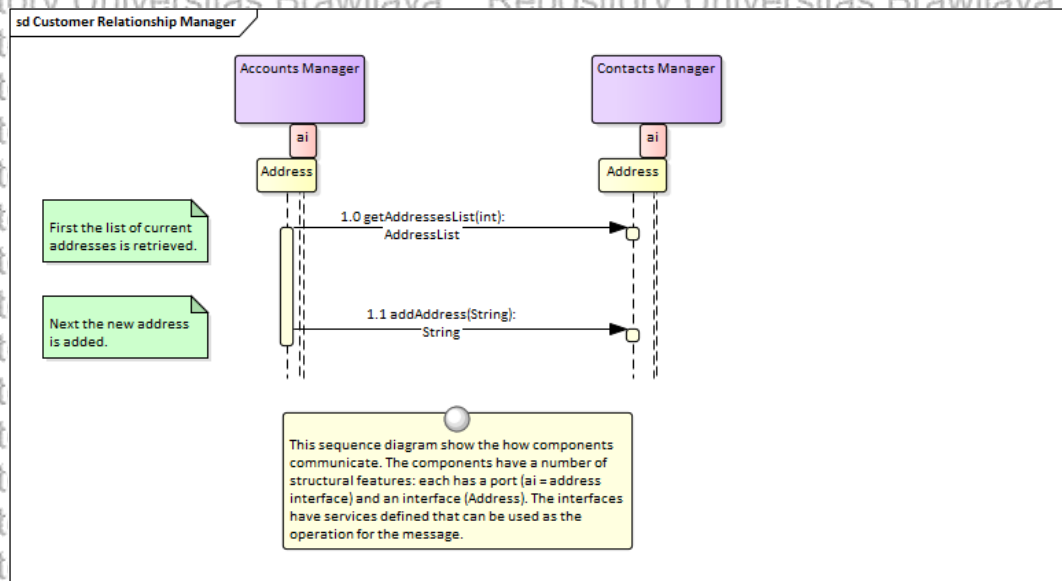
| Notasi   | Deskripsi   |
|--|---|
| <p>Aktor</p>                      | <p>Aktor merupakan sebuah visualisasi dari seorang pengguna yang akan menggunakan / berinteraksi dengan sistem secara langsung. Pengguna juga dapat diartikan sebagai sistem lain diluar sistem yang akan dibangun dan juga proses lain. Aktor dinamai dengan kata benda.</p> |
| <p>Lifeline</p>                   | <p>Lifeline merupakan sebuah visualisasi <i>timeline</i> aktifitas dari sebuah objek.</p>   |
| <p>Object</p>                   | <p>Object merupakan notasi yang memvisualisasikan sebuah objek yang berinteraksi yang mengirim pesan untuk menjalankan sebuah fungsi.</p>   |
| <p>Activation bar</p>           | <p>Activation Bar merupakan notasi yang memvisualisasikan waktu interaksi suatu objek.</p>  |
| <p>Input message</p>            | <p>Input Message merupakan sebuah notasi yang menggambarkan pemanggilan fungsi yang dilakukan objek terhadap objek lainnya. <i>Input message</i> dapat mempunyai parameter didalamnya.</p>  |
| <p>Output Message (return)</p>  | <p>Output Message merupakan sebuah notasi yang memvisualisasikan bahwa suatu fungsi dari objek telah berhasil dijalankan, dan mengembalikan sebuah pesan / data kepada objek yang memanggil fungsi tersebut.</p>  |
| <p>Frame</p>   | <p>Frame merupakan sebuah notasi yang memvisualisaikan perulangan, pilihan, maupun flow alternatif.</p>   |





Sumber: Whitten and Lonnie D. Bentley. (2007)

Berikut adalah contoh *sequence diagram* dari penjelasan komponen *sequence diagram* diatas, yang dapat dilihat pada Gambar 2.4.



**Gambar 2.4 Contoh Sequence Diagram**

Sumber: Sparx System (2017)

### 2.4.3 Pengujian Perangkat Lunak

Tahapan ini merupakan serangkaian kegiatan yang dilakukan secara sistematis dan telah direncanakan sebelumnya. Oleh sebab itu, maka *template* untuk pengujian perangkat lunak yang merupakan serangkaian langkah di mana pengembang dapat menempatkan teknik desain kasus pengujian khusus dan metode pengujian seharusnya didefinisikan untuk proses perangkat lunak (Pressman, 2010). Tahapan ini bertujuan untuk menemukan hal yang tidak seharusnya ada pada sistem maupun menemukan *error* pada sistem. Kesalahan/ *error* yang sering ditemukan di dalam pengembangan sebuah sistem adalah kesalahan pada konten, fungsi, *navigability*, *usability*, performa, kapasitas, dan juga keamanan. Metode pengujian yang tersedia diantaranya adalah *white-box testing*, *integration testing*, dan juga *black-box testing*. Lalu untuk pengujian dari sisi kemudahan pengaksesan pengguna adalah *usability testing*.

Teknik yang biasa digunakan dalam pengujian perangkat lunak adalah *black-box testing* dan *white-box testing* (Pressman, 2010):

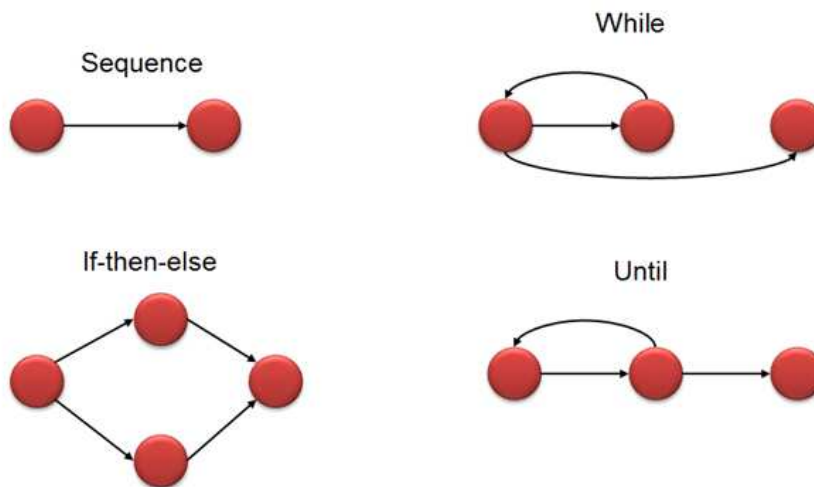


### 2.4.3.1 White-box testing

Pengujian ini dilakukan agar seluruh jalur logika yang ada telah dieksekusi minimal satu kali dan juga didefinisikan dalam sebuah jalur yang independen. Kemudian dijalankan seluruh kondisi logis yang tersedia dengan status benar atau salah. Selanjutnya menjalankan semua perulangan yang tersedia dalam batas dan diluar batas untuk memastikan bahwa syarat terpenuhi. Dalam penerapan *white-box testing*, terdapat teknik *basis path testing*. Teknik ini berguna untuk mendefinisikan test case melalui jalur yang tersedia (Pressman, 2010). *Basis path testing* nantinya akan menciptakan nilai kompleksitas logika (Cyclomatic Complexity). Dalam menerapkan metode ini, ada tiga tahapan yang harus dilakukan, yaitu dengan membuat *independent path*, berdasarkan *flowgraph* yang telah divisualisasikan. Dalam pembuatan *flowgraph*, ada tiga komponen yang dihasilkan, diantaranya:

1. **Node**  
Adalah sebuah notasi prosedur baik itu satu prosedur maupun lebih dari satu prosedur yang dinotasikan berbentuk lingkaran dengan penjelasan angka di dalamnya.
2. **Edge**  
Merupakan sebuah representasi alur dari *flowgraph* yang di visualisasikan dengan bentuk anak panah yang menuju ke node berikutnya.
3. **Predicate Node**  
Merupakan sebuah visualisasi suatu kondisi tertentu yang memiliki output dua buah.

Pada Gambar 2.5 akan ditampilkan pendefinisian notasi dari *flowgraph* dimana pada penggambarannya akan membentuk dua atau lebih node yang pada ujung-ujungnya akan dihubungkan oleh edge, notasi-notasi tersebut diantaranya adalah model *flow sequence*, *while*, *if-then-else*, dan *until* (Pressman, 2010).



Gambar 2.5 Notasi *Flowgraph*

Sumber: (Pressman, 2010)



Dari hasil visualisasi *flowgraph* dan juga perhitungan *independent path* akan menghasilkan sebuah metrik kompleksitas, metrik tersebut bernama Cyclomatic Complexity  $V(G)$ . Perhitungan untuk mendapatkan nilai Cyclomatic Complexity dapat dilakukan dengan menggunakan rumus seperti seperti yang dicontohkan pada keterangan dihalaman berikutnya (Pressman, 2010):

$$V(G) = E - N + 2$$

Keterangan:

E = merupakan jumlah dari ketersediaan edge

N = merupakan jumlah dari ketersediaan node

$$V(G) = P + 1$$

Keterangan:

P = merupakan jumlah ketersediaan predicate node

Kemudian jumlah *region* yang terdapat pada *flowgraph* juga dapat menentukan nilai  $V(G)$ . Nilai kompleksitas dari  $V(G)$  memiliki beberapa tingkatan berdasarkan *range* angka yang diklasifikasikan menjadi empat level. Pada Tabel 2.4 dapat dilihat pendefinisian klasifikasi nilai cyclomatic complexity.

**Tabel 2.5 Complexity Number**

| Complexity Number | Meaning  |
|-------------------|--|
| 1 – 10            | <i>Well written and structured code<br/>Testability is high<br/>Less effort and cost</i> |
| 10 – 20           | <i>The Code is Complex<br/>Testability is medium<br/>Medium effort and cost</i>          |
| 20 – 40           | <i>The code is very complex<br/>Testability is low<br/>High cost and effort</i>          |
| > 40              | <i>Not testable at all<br/>Cost and Effort is Very High</i>                              |

Sumber: Pressman (2010)

### 2.4.3.2 Black-box testing

Tahapan ini dilakukan untuk melakukan pengujian tetapi dengan tidak melihat kode program. Pengujian ini juga dilakukan berdasarkan spesifikasi sistem yang telah dijabarkan sebelumnya. (Sommerville, 2011). Pengujian ini juga kerap kali disebut dengan *behavioral testing system* yang disebabkan oleh penggunaannya yang lebih dikhususkan untuk kebutuhan fungsional sistem.



Pengujian ini juga akan menentukan apakah hasil implementasi sistem sudah sesuai dengan pendefinisian kebutuhan di awal. Pengujian ini juga mempunyai tujuan untuk mencari kesalahan yang terdapat pada sebuah sistem. Ada beberapa kategori kesalahan yang tersedia dalam pengujian *Black-box*, kategorinya adalah:

1. Kesalahan pada fungsi dan juga fungsi yang tidak dapat ditemukan.
2. Kategori kesalahan pada antarmuka.
3. Kategori kesalahan dalam pengaksesan database *non-internal* dan juga kesalahan dalam struktur data.
4. Kategori kesalahan fungsi yang terjadi pada sebuah proses.
5. Kategori kesalahan terminasi maupun kesalahan inisialisasi.

Pengujian terhadap kebutuhan non-fungsional juga dapat dilakukan menggunakan pengujian ini. Aspek yang akan dinilai adalah aspek sama sekali tidak mempunyai hubungan dengan aksi seorang aktor kepada sistem dan juga dengan fungsi tertentu. Fungsi dari pengujian non-fungsional adalah menguji apakah sistem yang akan dibuat bisa dijalankan di berbagai lingkungan yang berbeda dan mempunyai kebutuhan yang tidak sama. Pemilihan konfigurasi lingkungan juga dilakukan di dalam pengujian kompatibilitas (Yoon et al., 2008). Kompatibilitas sendiri merupakan salah satu faktor yang mempengaruhi pengadopsian dari *e-commerce* (Alam et al., 2007).

#### 2.4.3.3 Compatibility Testing

Komputer, tampilan layar, jenis browser, sistem operasi, dan kecepatan koneksi jaringan yang tidak sama memiliki pengaruh yang cukup signifikan pada operasi WebApp. Setiap konfigurasi komputasi dapat menghasilkan perbedaan dalam kecepatan pemrosesan sisi klien, resolusi tampilan, dan kecepatan koneksi. Antara satu browser dengan browser yang lain terkadang menghasilkan hasil yang tidak sama, terlepas dari tingkat standar HTML dalam WebApp. Kebutuhan yang diperlukan mungkin tidak tersedia untuk konfigurasi tertentu (Pressman, 2010).

Dalam beberapa kasus, masalah kompatibilitas kecil tidak menimbulkan masalah yang serius. Tetapi dalam kasus lain, kesalahan yang serius dapat ditemukan. Misalnya, kecepatan *download* tidak dapat diterima, kurangnya *plugin* yang diperlukan membuat konten menjadi tidak tersedia pada lingkungan tertentu, perbedaan browser dapat mempengaruhi tata letak halaman secara signifikan, gaya font dapat menjadi tidak terbaca, ataupun formulir tidak terorganisir dengan baik. Pengujian kompatibilitas diharapkan untuk menyingkap masalah ini sebelum WebApp online (Pressman, 2010).

Pengujian kompatibilitas bertujuan untuk mengecek kompatibilitas suatu sistem. Apakah suatu sistem dapat dijalankan di lingkungan yang bermacam-macam atau tidak. Dikarenakan sistem operasi, hardware, dan aplikasi yang bermacam-macam. Tingkat kompatibilitas suatu sistem ditentukan dari seberapa mampunya sistem berjalan pada lingkungan yang berbeda. Semakin



banyak sistem bisa dijalankan di sebuah lingkungan, maka semakin tinggi tingkat kompatibilitasnya.

Pengujian kompatibilitas akan menguji kebutuhan non-fungsional, yaitu menguji apakah sistem dapat dijalankan pada lingkungan yang berbeda-beda. Pengujian kompatibilitas ini nantinya akan menentukan konfigurasi dari lingkungan tertentu. Pengujian kompatibilitas nantinya akan membantu mengetahui permasalahan yang ada pada suatu lingkungan tertentu. Tiga kategori masalah pada pengujian kompatibilitas yaitu masalah minor (*minor issues*), masalah major (*major issues*), dan masalah kritis (*critical issues*). *Critical issues* menandakan bahwa ada beberapa fitur yang tidak didukung pada suatu lingkungan. *Major issues* menandakan bahwa ada masalah pada performa maupun tampilan dari sebuah sistem. *Minor issues* menandakan bahwa ada permasalahan performa maupun tampilan yang minor dari sebuah sistem.

## 2.5 Teknologi Pengembangan Sistem

Midtrans, Laravel, MySQL, dan SortSite merupakan teknologi yang akan dibahas pada bab ini. Teknologi tersebut juga akan digunakan untuk membantu dalam pengembangan sistem.

### 2.5.1 Midtrans

Midtrans adalah sebuah instansi yang menawarkan layanan *payment gateway*. Midtrans juga mempunyai bermacam fitur yang dapat membantu proses transaksi para penggunanya. Midtrans menyediakan berbagai metode pembayaran, diantaranya adalah *card payment*, *bank transfer cardless debit*, *e-wallet*, *direct debit*, dan yang lainnya. Fasilitas lain yang ditawarkan oleh pihak midtrans adalah sistem satu *dashboard* yang bisa melihat semua transaksi yang terjadi dan juga sistem *anti-fraud* yang mendukung banyak transaksi, mengurangi *chargeback*, dan meningkatkan omset bisnis dengan sistem deteksi kecurangan yang dikembangkan khusus untuk pasar Indonesia. Midtrans telah dipakai oleh banyak perusahaan besar maupun startup seperti pegipegi, traveloka, matahari mall, gojek, dan sebagainya.

Pada penelitian ini, midtrans digunakan sebagai sarana pembayaran pada pemesanan dan juga perpanjangan pekerja rumah tangga. Midtrans diterapkan pada bagian *back-end* sistem. Calon majikan yang akan melakukan pemesanan pada sistem akan memilih calon pekerja yang ingin dipesan, lalu sistem akan mengirimkan beberapa data ke API dari midtrans yang selatnya akan diproses oleh midtrans dan menghasilkan sebuah pemesanan lengkap dengan kode akun virtual pembayaran. Kode ini dikirimkan kembali oleh midtrans ke sistem, lalu pihak majikan melakukan pembayaran sesuai dengan jumlah dan metode pembayaran yang ada. Setelah majikan melakukan pembayaran, pembayaran tersebut akan diperiksa secara otomatis oleh midtrans, lalu diproses pembayarannya dan midtrans mengirimkan notifikasi kepada sistem yang menyatakan bahwa pembayaran telah selesai dilakukan.



### 2.5.2 Laravel

Laravel merupakan sebuah *framework* yang digunakan dalam pengembangan aplikasi *web*. *Model-view-controller* (MVC) merupakan model yang diadopsi oleh laravel, dan menggunakan PHP sebagai bahasa pemrograman. Laravel sendiri mempunyai tujuan untuk meningkatkan kualitas dari sistem yang akan dikembangkan dan mengurangi biaya pada proses pengembangan sistem, serta perawatan perangkat lunak. Selain itu, laravel juga memiliki beberapa fitur yang dapat mempercepat proses implementasi suatu perangkat lunak (McCool, 2012).

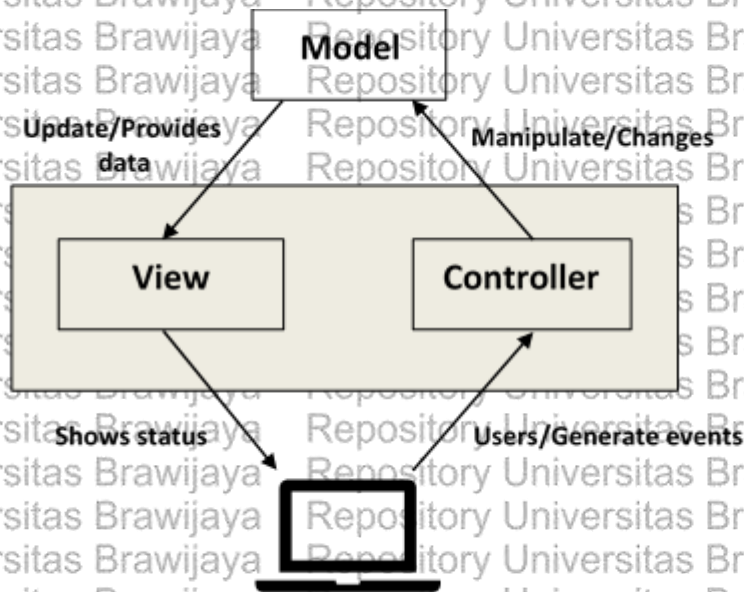
Pembangunan arsitektur *framework* Laravel menggunakan pola MVC, maka dari itu struktur dalam *framework* Laravel memisahkan *model*, *view*, dan *controller* ke satu bagian masing – masing. Menurunkan ikatan dari setiap komponen merupakan tujuan dipisahkannya *model*, *view*, dan *controller*. Maka jika terdapat sebuah perubahan pada satu komponen maka tidak akan menimbulkan dampak besar bagi komponen lainnya. Di dalam penelitiannya, (Caytiles and Lee, 2014) mengungkapkan jika sistem yang dibuat menggunakan pola MVC akan menghasilkan tiga objek, diantaranya adalah :

*Model*, yang berperan sebagai perekam aksi dari domain permasalahan maupun antarmuka. *Model* juga digunakan untuk mengelola data beserta logika data dan aturan – aturan aplikasi dan juga melakukan manipulasi data.

*View*, yang berperan sebagai penampil keluaran apapun dan juga dalam bentuk apapun kedalam antarmuka. Keluaran yang ditampilkan diantaranya adalah teks, grafik, *checkbox*, gambar, diagram, maupun bentuk lainnya yang merepresentasikan tampilan. Sehingga beberapa elemen dalam satu *view* bisa digabungkan.

*Controller*, berperan sebagai penghubung antara model dengan *view*. Dimana *controller* menyimpan dan mengatur semua logika melalui inputan user dan mengatur logika dalam memanipulasi data.

Pada Gambar 2.5 dapat dilihat penjelasan dari model MVC.



Gambar 2.6 Relasi Komponen MVC

Sumber: (Caytiles dan Lee, 2014)

### 2.5.3 MySQL

Pengertian dari MySQL adalah sebuah aplikasi yang digunakan untuk manajemen basis data SQL atau biasa disebut dengan *database management system* atau disingkat menjadi DBMS. Aplikasi ini punya karakteristik yaitu *multi user* dan juga *multi thread*. MySQL telah di *install* lebih dari enam juta di seluruh dunia. MySQL dalam penggunaannya menggunakan Structured Query Language atau biasa disingkat dengan sebutan (SQL). Aplikasi ini juga mempunyai berbagai macam fitur yang sangat bisa menopang proses manajemen dari sebuah absis data. MySQL AB adalah seorang pemberi sponsor yang juga bertindak sebagai pemilik dari MySQL, dimana MySQL AB memegang hak cipta penuh dari MySQL (Solichin, 2010).

### 2.5.4 SortSite

SortSite adalah sebuah aplikasi yang menyediakan layanan untuk melakukan pengujian terhadap sistem dengan basis web. Pengujian dilakukan dengan cara menguji seluruh halaman web yang tersedia pada sebuah sistem. Diantara pengujian yang bisa dilakukan oleh SortSite, salah satu pengujiannya adalah pengujian dari kualitas sebuah sistem (PowerMapper, 2019). SortSite juga mempunyai sebuah layanan yang mampu untuk melakukan pengujian terhadap kompatibilitas suatu sistem. Proses pengujian yang akan dilakukan adalah pengujian terhadap beberapa perambah yang biasanya sering digunakan. Jenis-jenis perambah yang diuji oleh sortsite diantaranya adalah firefox, edge, google chrome, safari, perangkat android, dan perangkat ios. SortSite memaparkan tiga



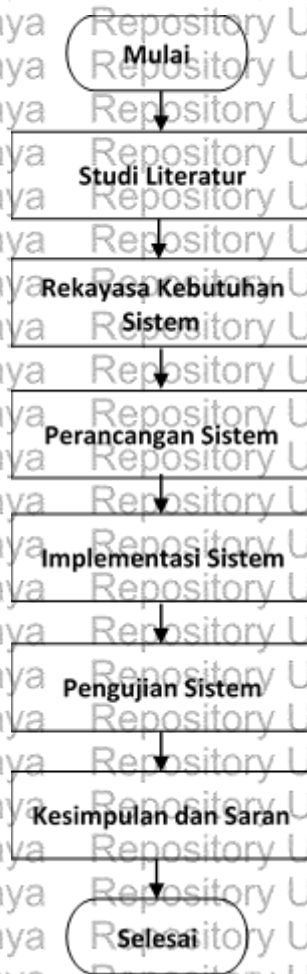




## BAB 3 METODOLOGI

Pada tahapan ini akan dijabarkan mengenai metode yang akan digunakan pada penelitian kali ini. Beberapa hal dalam metodologi penelitian yaitu tahapan penelitian, studi literatur, rekayasa kebutuhan sistem, lalu melakukan perancangan sistem dilanjutkan dengan implementasi sistem, lalu melakukan pengujian sistem, dan yang terakhir adalah pengambilan kesimpulan dan saran.

Kerangka alur metodologi ini bertujuan untuk memahami alur dari penelitian yang dilakukan. Gambar 3.1 menjelaskan urutan dari diagram alir pengerjaan dalam penelitian yang akan dilakukan.



Gambar 3.1 Diagram Alir Penelitian

### 3.1 Studi Literatur

Sub bab ini akan menjabarkan dasar dari teori yang akan dijadikan acuan didapat dari penelitian sebelumnya, jurnal, *ebook*, buku cetak, dan juga literatur



yang diambil dari internet untuk menunjang proses penulisan tugas akhir ini. Teori - teori pendukung yang digunakan tersebut meliputi:

1. Pekerja Rumah Tangga
2. Rekayasa Perangkat Lunak
3. Pendekatan Berorientasi Objek
4. Pemodelan Berorientasi Objek
5. Pengembangan Perangkat Lunak
6. Pengujian Perangkat Lunak
7. *Payment Gateway*
8. Midtrans
9. Laravel
10. MySQL
11. SortSite

### 3.2 Rekayasa Kebutuhan Sistem

Sub bab ini akan menjelaskan tentang rekayasa kebutuhan sistem yang merupakan sebuah proses yang membangun jembatan antara tahapan desain dan tahapan pembangunan (Pressman, 2010). Rekayasa kebutuhan menyediakan sebuah mekanisme yang sesuai agar kita sebagai pihak *developer* mengerti kebutuhan apa saja yang dibutuhkan oleh *stakeholder*, analisis kebutuhan, menilai kelayakan dari sebuah sistem, negosiasi solusi yang bisa diterima akal, menspesifikasikan solusi dari suatu permasalahan yang tidak ambigu, memvalidasi seluruh spesifikasi kebutuhan, dan manajemen kebutuhan hingga berubah menjadi sebuah fungsi. Tahapan yang ada pada rekayasa kebutuhan dimulai dari analisis kebutuhan, lalu dilanjutkan dengan spesifikasi kebutuhan yang telah didapat, lalu melakukan validasi dan verifikasi dari kebutuhan, dan melakukan manajemen kebutuhan.

#### 3.2.1 Elisitasi dan Analisis Kebutuhan

Sub bab ini menjelaskan tentang tahap yang dijalankan untuk mendefinisikan kebutuhan apa saja yang dibutuhkan oleh user. Elisitasi dan analisis kebutuhan dilakukan dengan cara wawancara kepada pegawai CV. Sitiari, pekerja CV. Sitiari, dan pelanggan CV. Sitiari. Pertanyaan yang ditanyakan adalah tujuan dari perangkat lunak yang akan dibuat, apa yang ingin dicapai, bagaimana perangkat lunak yang dibuat nantinya akan memenuhi kebutuhan bisnis dari CV. Sitiari sendiri, dan yang terakhir adalah bagaimana nantinya perangkat lunak yang akan dibuat dipakai sehari-hari. Tahapan yang akan digunakan dalam elisitasi dan analisis kebutuhan adalah :

1. Menilai kelayakan teknis dan juga bisnis untuk sistem yang diusulkan.
2. Mengidentifikasi orang-orang yang nantinya akan membantu menemukan kebutuhan dan memahami bias dari organisasi mereka.
3. Menetapkan lingkungan pengembangan.
4. Mengidentifikasi batasan domain.



5. Mendefinisikan satu ataupun lebih dari satu metode elisitasi kebutuhan.
6. Meminta kesediaan dari banyak orang untuk berpartisipasi agar kebutuhan bisa didapat dari berbagai sudut pandang.
7. Mengidentifikasi kebutuhan yang ambigu.

Ada 3 aspek yang membuat elisitasi kebutuhan menjadi sulit. Diantaranya adalah :

1. Permasalahan ruang lingkup  
Batasan sistem tidak jelas atau pelanggan / pengguna menentukan sebuah detail teknis yang sebenarnya tidak diperlukan yang nantinya dapat membingungkan tujuan sistem secara keseluruhan.
2. Permasalahan pemahaman  
Pelanggan / pengguna tidak yakin sepenuhnya dengan apa yang mereka butuhkan, memiliki pemahaman yang kurang baik mengenai kemampuan dan keterbatasan dari lingkungan komputasi yang ada, tidak memiliki pemahaman penuh tentang domain masalah, kesulitan berkomunikasi kebutuhan dengan insinyur sistem, menghilangkan informasi yang diyakini "jelas," kebutuhan pelanggan / pengguna yang bertentangan dengan kebutuhan pelanggan / pengguna lain, dan juga menentukan persyaratan yang ambigu atau tidak dapat diuji.
3. Permasalahan volatilitas  
Kebutuhan berubah seiring dengan waktu berjalan. Untuk membantu mengatasi masalah ini, Anda harus menjalankan pendekatan dalam mencari kebutuhan secara terorganisir.

Hasil yang didapatkan setelah melakukan elisitasi dan analisis kebutuhan adalah daftar aktor, daftar kebutuhan, dan skenario dari sistem yang akan dibuat.

### 3.2.2 Spesifikasi Kebutuhan

Di dalam konteks sistem dengan basis komputer (dan perangkat lunak), spesifikasi mempunyai arti yang berbeda-beda. Spesifikasi dapat dituangkan kedalam sebuah dokumen, spesifikasi juga bisa dituangkan kedalam model grafik, kumpulan skenario penggunaan, model matematika formal, *prototype*, atau kombinasi dari semuanya. Spesifikasi kebutuhan berguna sebagai pondasi untuk rekayasa perangkat keras, rekayasa perangkat lunak, rekayasa basis data, dan juga rekayasa manusia. Spesifikasi kebutuhan menjelaskan fungsi dan performa sebuah sistem dan juga batasan dari sistem itu sendiri. Hasil yang didapatkan dari tahapan spesifikasi adalah spesifikasi kebutuhan yang menjelaskan sebuah kebutuhan secara spesifik.



### 3.2.3 Validasi dan Verifikasi

Tahapan ini merupakan sebuah tahapan untuk memeriksa spesifikasi dan memastikan seluruh kebutuhan sudah dituliskan tanpa ada ambiguitas. Tahapan ini juga bertujuan untuk memastikan ketidakkonsistenan, kelalaian, dan error sudah terdeteksi dan terbenahi. Pihak yang melakukan validasi dan verifikasi adalah *developer*, pegawai CV. Sitiari, pekerja CV. Sitiari, dan pelanggan CV. Sitiari. Teknik yang digunakan untuk validasi dan verifikasi adalah menggunakan kuisisioner *Requirement Validation Checklist*.

### 3.2.4 Manajemen Kebutuhan

Kebutuhan pada sistem dengan basis komputer selalu berubah, dan keinginan untuk mengubah kebutuhan tetap ada sepanjang umur sistem. Manajemen kebutuhan merupakan serangkaian aktivitas yang membantu tim proyek mengidentifikasi, mengendalikan, dan juga melacak kebutuhan dan perubahan terhadap kebutuhan kapan saja saat proyek berlangsung. Diagram *use case* merupakan hasil yang didapatkan pada tahapan manajemen kebutuhan.

## 3.3 Perancangan Sistem

Tahapan ini merupakan tahapan penerjemahan kebutuhan dari sistem menjadi model struktural sistem. Tahap perancangan dilakukan untuk merubah model kebutuhan menjadi model perancangan sistem yang beruparancangan detail arsitektur perangkat lunak, rancangan struktur data, dan rancangan komponen yang diperlukan untuk mewujudkan sistem. Proses perancangan sistem yang dilakukan pada penelitian ini akan dilakukan dengan menggunakan metode OOD dan dimodelkan dengan menggunakan UML. Hasil yang didapatkan dari tahapan ini menjadi jawaban dari sebuah sistem manajemen dan pemesanan pekerja rumah tangga. Mekanisme proses perancangan dilakukan dengan alur sebagaimana tertulis pada list dibawah ini.

1. Pemodelan *sequence diagram*. Dilakukan dengan cara mendefinisikan hubungan antar objek yang telah ada.
2. Pemodelan *class diagram*. Pemodelan ini dilakukan dengan mendefinisikan daftar kelas beserta hubungan antar kelas berdasarkan objek yang ada pada *sequence diagram*.
3. Perancangan komponen. Perancangan ini akan dilakukan dengan merancang algoritme dari method yang ada pada kelas. Hasil dari perancangan komponen yaitu sebuah *pseudocode* untuk setiap algoritme.
4. Perancangan data. Dilakukan dengan pemodelan *physical data model* atau biasa disingkat (PDM). PDM mendefinisikan daftar tabel, relasi antar tabel, daftar kolom di setiap tabel, *primary key*, *foreign key*, dan tipe data dari setiap kolom.



5. Perancangan antarmuka. Perancangan ini dibuat dengan memvisualisasikan rancangan antarmuka pengguna dari sistem yang akan dikembangkan.

### 3.4 Implementasi Sistem

Tahapan ini merupakan sebuah fase pembangunan sistem yang berkaca pada tahapan perancangan sistem yang telah dilakukan sebelumnya. Proses implementasi dilakukan dengan memanfaatkan metode OOP. Dalam implementasi sistem ini dibangun menggunakan HTML dan CSS sebagai bahasa penyusunan *layout* dan PHP sebagai bahasa pemrograman dalam mengelola informasi. Adapun teknologi-teknologi lain yang akan dilibatkan pada implementasi adalah: Laravel dan Midtrans. Proses implementasi akan dijabarkan pada list berikut:

1. Implementasi logika program yang memuat algoritme pada perancangan sistem. Nantinya, logika tersebut akan di implementasikan menggunakan bahasa PHP dan menggunakan Laravel sebagai framweork. Logika program diterapkan pada *layer controller* dan bisa juga diterapkan pada *layer model*. Selain itu, pada logika program bagian pembayaran akan diterapkan Midtrans sebagai framework pembayaran secara *online*.
2. Implementasi basis data yang dilakukan dengan cara mendefinisikan struktur tabel berdasarkan PDM (*physical data model*) yang sudah dibuat pada tahapan sebelumnya yaitu tahapan perancangan sistem.
3. Implementasi antarmuka dengan cara membuat tampilan berdasarkan perancangan sebelumnya yaitu perancangan antarmuka yang sebelumnya dibuat pada tahapan perancangan sistem.

### 3.5 Pengujian Sistem

Tahapan ini merupakan sebuah proses yang dilakukan untuk mengetahui hasil uji dari implementasi. Tujuan dari tahapan ini adalah memastikan apakah sistem yang dibangun telah berjalan dengan benar dan dengan harapan untuk mengurangi atau bahkan menghilangkan *error* / masalah yang ada pada sistem. Langkah – langkah yang ada pada pengujian sistem adalah :

1. Pengujian Unit  
Tahapan ini merupakan tahapan yang bertujuan untuk menguji komponen yang paling kecil dari sistem yang akan dikembangkan. Komponen yang dimaksud bisa jadi adalah sebuah klas, fungsi, maupun sebuah objek. Nantinya, akan dilihat bagaimana cara kerja dari suatu komponen di dalam sebuah klas. Teknik pengujian yang akan digunakan pada tahapan ini adalah teknis pengujian kotak putih atau biasa disebut (*white-box*). Metode yang digunakan dalam mengumpulkan *test case* adalah metode *basis path testing*.
2. Pengujian Integrasi



Tahapan ini merupakan tahapan yang bertujuan untuk melakukan uji coba pada modul dengan modulnya apakah modul tersebut sudah berjalan dengan baik dan terintegrasi dengan baik. Metode yang akan digunakan adalah metode top-down integration. Tahapan pengujian ini akan menggunakan metode pengujian kotak hitam atau yang biasa disebut dengan (*black-box*). Dalam pelaksanaan pengujian ini, akan dibuat kelas stub yang tujuan dibuatnya adalah untuk berkomunikasi dengan class pada level di atasnya.

### 3. Pengujian Validasi

Tahapan ini merupakan tahapan yang bertujuan untuk melakukan uji coba terhadap fitur atau fungsi yang ada pada sistem. Pengujian ini akan membandingkan apakah hasil yang didapatkan pada implementasi sistem sudah sesuai dengan hasil yang ada pada rekayasa kebutuhan. Pengujian ini tidak memperhatikan bagian dalam sistem, oleh karena itu pengujian ini akan menggunakan teknik kotak hitam. Dalam pelaksanaannya, pengujian ini akan menguji apakah jika suatu fungsi dimasukkan inputan tertentu, hasil yang didapatkan akan sesuai dengan *expected result*.

### 4. Pengujian Kompatibilitas

Tahapan ini merupakan tahapan yang bertujuan untuk melakukan uji coba kepada kebutuhan yang *non-fungsional*. Pada pelaksanaan pengujian ini, akan digunakan perangkat lunak SortSite untuk membantu menguji kompatibilitas sistem ini. Nantinya, SortSite akan menguji dan menjalankan sistem pada 8 jenis perambah.

## 3.6 Kesimpulan dan Saran

Tahapan ini merupakan tahapan yang dilakukan setelah semua tahapan di atas selesai. Tahapan ini bertujuan untuk menarik kesimpulan dari hasil pengujian dan analisis. Tahap selanjutnya adalah penarikan saran yang berfungsi untuk memperbaiki ketika ada kesalahan dalam penulisan dan juga untuk memberikan pertimbangan dalam pengembangan sistem yang lebih lanjut.

## BAB 4 REKAYASA KEBUTUHAN

Tahapan ini merupakan sebuah tahapan yang akan dilakukan pertama kali dalam pengembangan sistem ini. Tujuan dari tahapan ini adalah menggali kebutuhan pengguna yang harus dan dibutuhkan oleh pengguna, dalam hal ini adalah kebutuhan yang harus ada pada Sistem Manajemen dan Pemesanan Pekerja Rumah Tangga. Nantinya, tahapan ini akan menghasilkan 2 komponen yang sangat krusial untuk proses pengembangan selanjutnya yaitu *use case diagram* dan *use case scenario*. Komponen *use case diagram* berguna untuk mengidentifikasi aktor yang akan berinteraksi dengan sistem. Sedangkan *use case scenario* digunakan untuk menjabarkan interaksi dari suatu kebutuhan terhadap pengguna atau aktor yang menjalankan fungsi tersebut. Struktur dari bab rekayasa kebutuhan akan ditunjukkan pada Gambar 4.1.



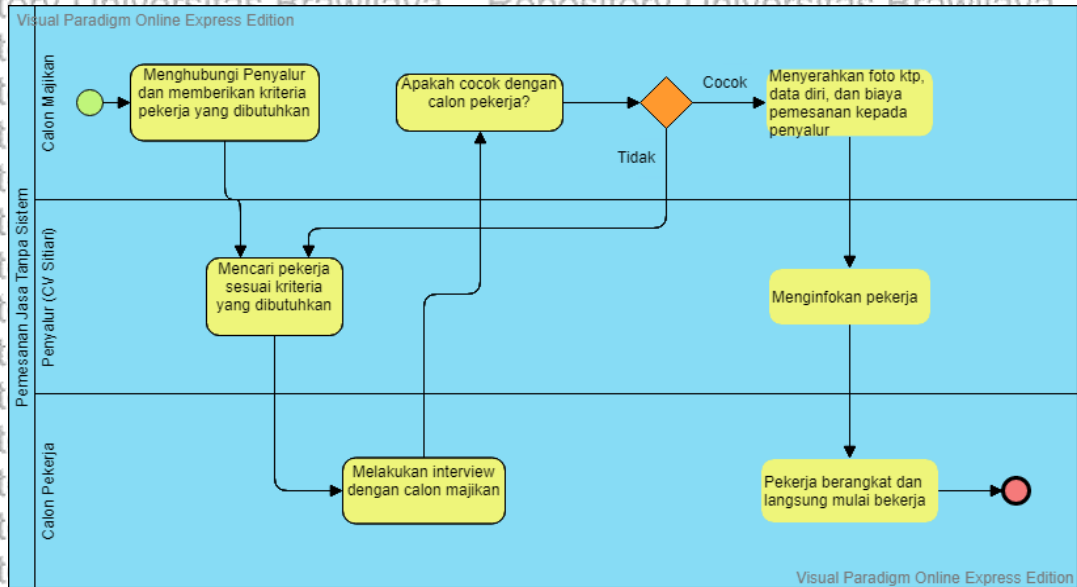
Gambar 4.1 Struktur Bab Rekayasa Kebutuhan

### 4.1 Analisis Kebutuhan

Tahapan analisis kebutuhan adalah sebuah tahapan yang memiliki tujuan untuk mendefinisikan seluruh domain masalah yang ada pada lingkup yayasan penalar pekerja rumah tangga aitu CV.Sitiari, dalam pelaksanaan analisis kebutuhan, akan digunakan metode wawancara. Metode ini digunakan dikarenakan metode wawancara akan lebih mudah dan efektif dalam melakukan elisitasi data kualitatif. Menggunakan teknik wawancara akan memudahkan transformasi data menjadi sebuah kebutuhan. Wawancara akan dilakukan kepada pihak CV. Sitiari yang ada di kota Malang.

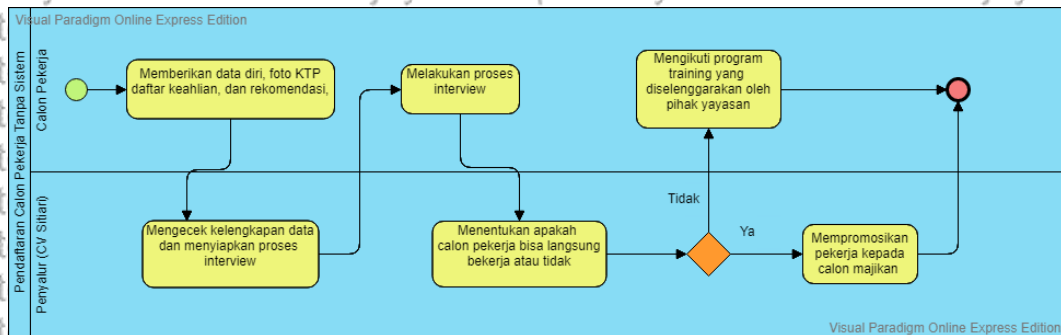
Proses bisnis untuk memesan jasa pekerja rumah tangga di CV. Sitiari akan dijabarkan pada Gambar 4.2. Proses bisnis dimulai dari calon majikan yang menghubungi pihak CV. Sitiari untuk memesan jasa PRT melalui *whatsapp* atau mendatangi cabang CV. Sitiari. Calon majikan memberikan kriteria pekerja sesuai yang diinginkan, lalu pihak Sitiari melakukan pengecekan ketersediaan pekerja sesuai dengan kriteria yang diinginkan calon majikan. Jika pekerja dengan kriteria

yang diinginkan calon majikan tersedia maka selanjutnya calon majikan akan melakukan interview kepada calon PRT melalui whatsapp, telfon, atau tatap muka secara langsung dengan mendatangi cabang CV. Sitiari. Jika calon majikan cocok dengan pekerja yang tersedia, maka calon majikan harus menyerahkan berkas yang dibutuhkan seperti foto KTP, data diri, dan juga biaya pemesanan kepada pihak CV. Sitiari. Setelah itu maka pekerja akan langsung dikirimkan ke tempat majikan.



**Gambar 4.2 Proses Bisnis As-Is Pemesanan Pekerja Rumah Tangga**

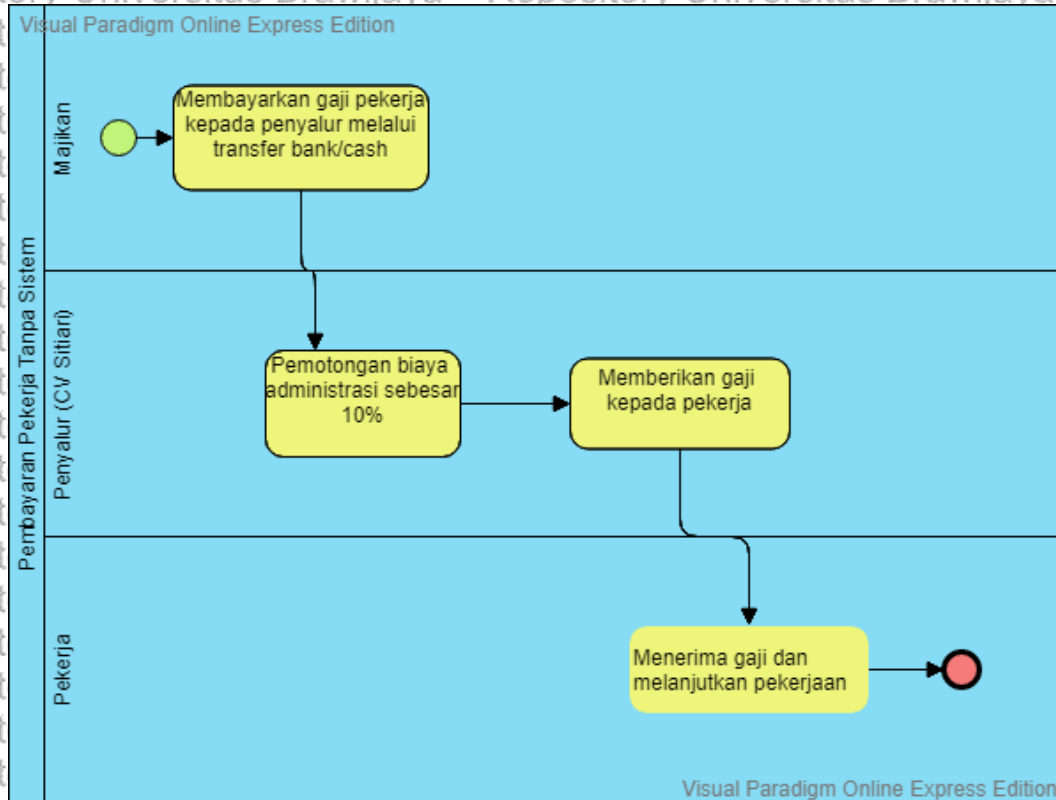
Gambar 4.3 merupakan ilustrasi proses bisnis dari pendaftaran calon pekerja ke CV. Sitiari. Proses bisnis ini dimulai dari calon pekerja yang mendatangi cabang CV. Sitiari dan memberikan data diri dan foto KTP sebagai syarat pendaftaran. Setelah itu pihak Sitiari akan melakukan pengecekan terhadap kelengkapan data dan melakukan proses interview kepada calon pekerja. Jika pihak CV. Sitiari merasa calon pekerja sudah siap bekerja maka calon pekerja tersebut akan langsung dimasukkan kedalam list PRT siap bekerja. Jika Sitiari menganggap calon pekerja belum siap untuk bekerja maka pihak Sitiari akan melakukan pelatihan internal untuk calon pekerja tersebut.



**Gambar 4.3 Proses Bisnis As-Is Pendaftaran Calon pekerja**

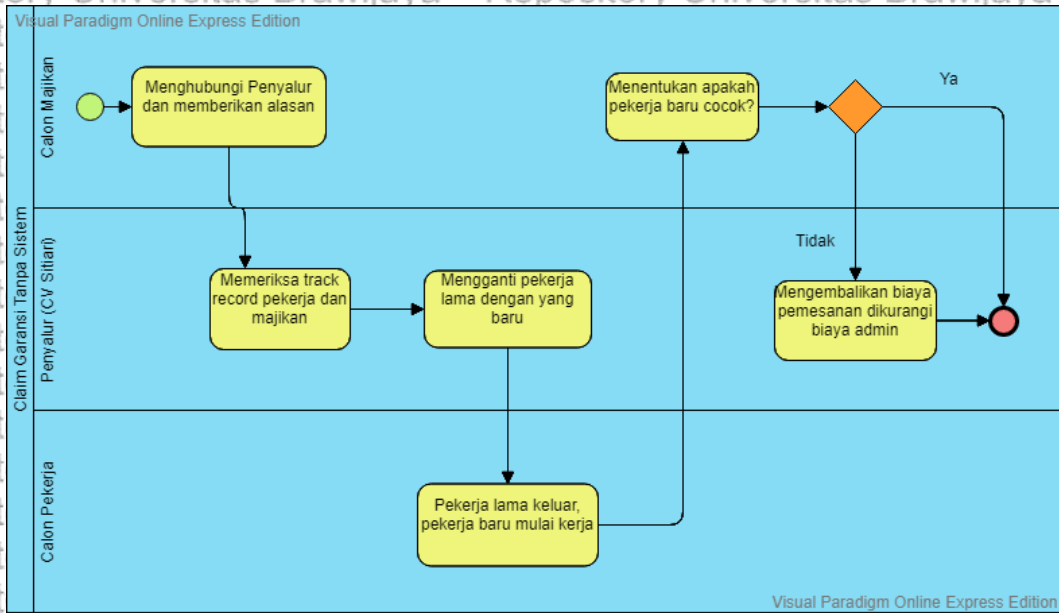


Gambar 4.4 adalah ilustrasi proses bisnis untuk penggajian pegawai. Proses bisnis dimulai dengan majikan membayar biaya bulanan kepada pihak CV. Sitiari lalu pihak Sitiari akan memberikan gaji tersebut kepada PRT dengan memotong sebesar 10%.



**Gambar 4.4 Proses Bisnis As-Is Penggajian Pegawai**

Gambar 4.5 merupakan sebuah ilustrasi dari proses bisnis dari claim garansi. Proses bisnis dimulai dari pihak majikan yang menghubungi pihak Sitiari untuk mengajukan claim garansi beserta alasan claim garansi. Setelah itu pihak Sitiari akan melakukan pengecekan track record pekerja serta majikan dan mengganti pekerja dengan yang baru. Jika majikan merasa cocok dengan pekerja yang baru maka pekerja akan langsung mulai bekerja. Jika tidak cocok maka uang akan dikembalikan tetapi dikurangi biaya admin.



Gambar 4.5 Proses Bisnis As-Is Claim Garansi

Gambar 4.6 merupakan sebuah ilustrasi dari proses bisnis dari resign pegawai. Proses bisnis dimulai dengan pekerja mengajukan permohonan resign kepada Silitari. Jika pekerja tersebut sudah tidak mempunyai tanggungan pekerjaan maka pegawai tersebut bisa melakukan resign.



Gambar 4.6 Proses Bisnis As-Is Resign Pegawai

## 4.2 Gambaran Umum Sistem

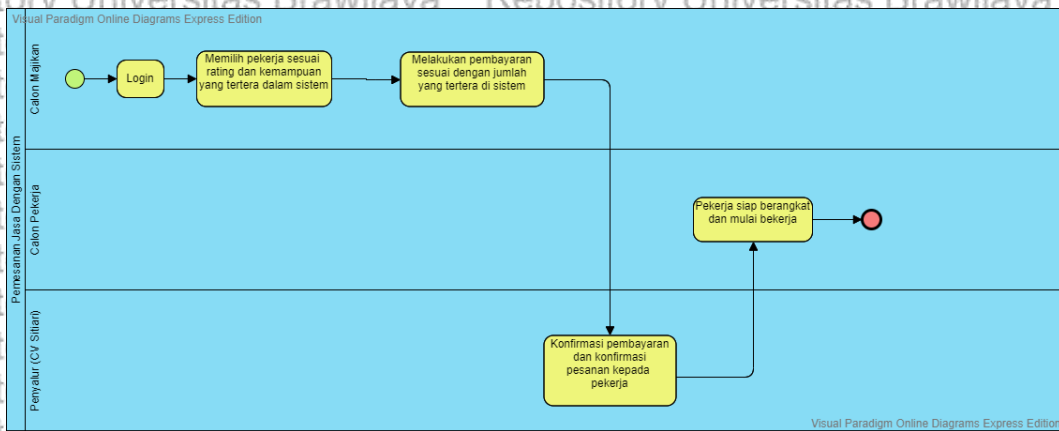
Sistem manajemen dan pemesanan jasa pekerja rumah tangga ini merupakan perangkat lunak yang mempunyai tujuan untuk melakukan manajemen data majikan dan juga pekerja. Sistem ini juga bertujuan untuk melakukan proses pembayaran secara online serta pencatatan pembayaran. Sistem ini merupakan sistem yang dikembangkan dengan basis web. Basis web digunakan agar sistem dapat digunakan oleh *device customer* yang beragam. Sistem ini dibuat dengan menggunakan framework Laravel. Laravel digunakan untuk memastikan sistem dapat berjalan dengan aman. Pengaksesannya dilakukan dengan menggunakan



web browser. Sedangkan untuk pembayaran digunakan framework Midtrans agar pembayaran lebih mudah dan juga aman.

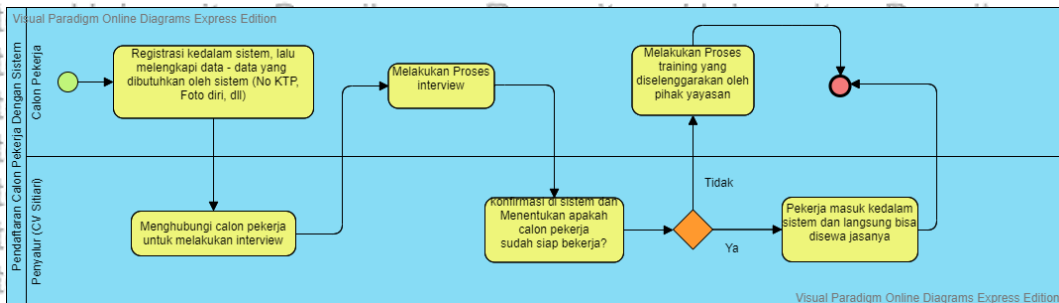
Sistem ini menyediakan beberapa fungsi yang nantinya akan digunakan oleh para pekerja maupun majikan CV. Sitiari. Selain itu, admin dari Sitiari sendiri juga akan menggunakan sistem ini untuk melakukan manajemen pekerja dan majikan dan juga manajemen transaksi pembayaran.

Pada Gambar 4.7 dijelaskan proses bisnis to-be pemesanan pekerja rumah tangga setelah menggunakan sistem. Dapat diperhatikan bahwa proses bisnis yang terjadi setelah adanya sistem dapat memotong alur proses bisnis yang kurang efektif. Selain itu, para calon majikan jadi lebih leluasa memilih calon pekerja rumah tangga tanpa perlu bertanya kepada pihak CV. Sitiari terlebih dahulu.



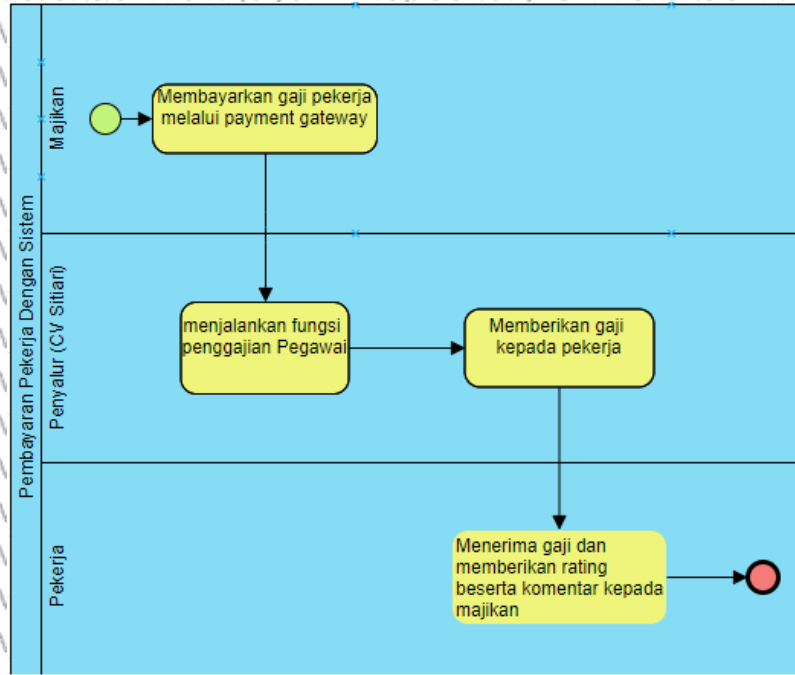
**Gambar 4.7 Proses Bisnis To-Be Pemesanan Pekerja Rumah Tangga**

Pada Gambar 4.8 dipaparkan proses bisnis to-be pendaftaran calon pekerja setelah menggunakan sistem. Dapat diperhatikan bahwa proses bisnis yang terjadi setelah adanya sistem memungkinkan pendaftar untuk mendaftar secara online dan melakukan interview secara online, karena tidak harus datang untuk menyerahkan data diri.



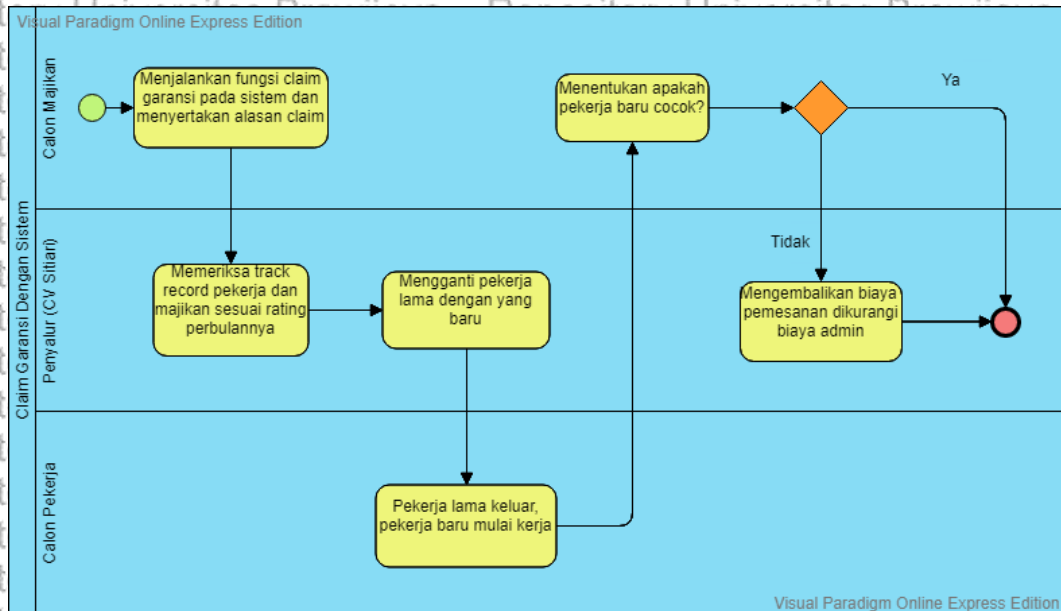
**Gambar 4.8 Proses Bisnis To-Be Pendaftaran Calon pekerja**

Pada Gambar 4.9 dipaparkan proses bisnis to-be pembayaran pekerja setelah menggunakan sistem.



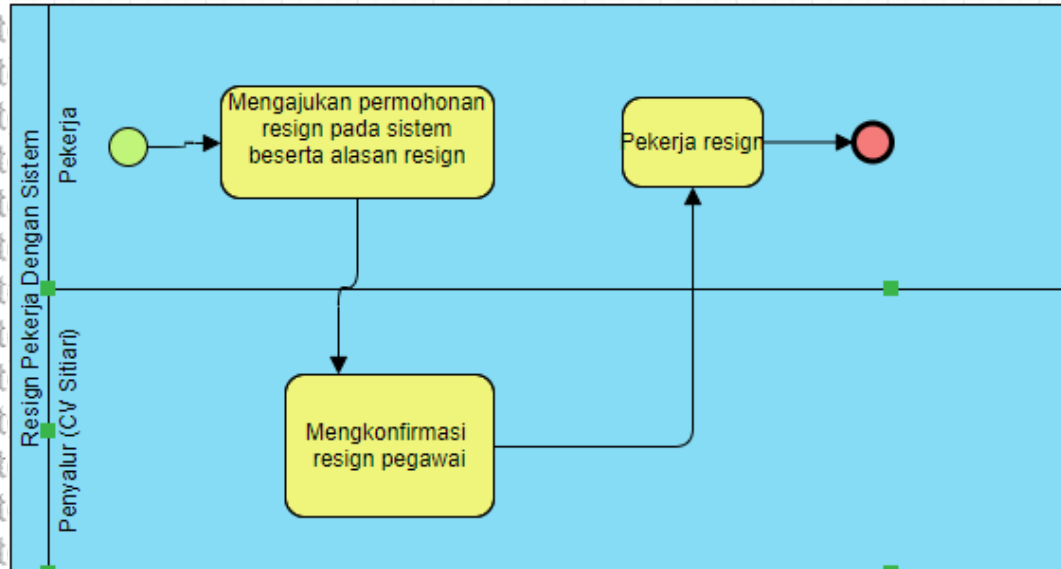
**Gambar 4.9** Proses Bisnis To-Be Pembayaran Pekerja

Pada Gambar 4.10 dipaparkan proses bisnis to-be claim garansi setelah menggunakan sistem. Dapat diperhatikan proses claim garansi yang dilakukan dengan menggunakan sistem, proses bisnis yang terjadi akan lebih efektif karena proses bisnis dilakukan menggunakan sistem.



**Gambar 4.10** Proses Bisnis To-Be Claim Garansi

Pada Gambar 4.11 dipaparkan proses bisnis to-be resign pegawai setelah menggunakan sistem. Dapat diperhatikan proses bisnis setelah menggunakan sistem memungkinkan pekerja dan pihak sitiari melakukan proses resign secara online.



Gambar 4.11 Proses Bisnis To-Be Resign Pegawai

### 4.3 Identifikasi Aktor

Sub bab ini akan menjabarkan aktor - aktor yang akan berhubungan dengan sistem beserta perannya. Aktor-aktor yang tersedia diantaranya adalah *Guest*, *Pekerja*, *Majikan*, *Midtrans*, dan *Admin*. Hasil identifikasi para aktor dan juga deskripsinya akan dipaparkan pada Tabel 4.1.

Tabel 4.1 Identifikasi Aktor

| No. | Aktor   | Deskripsi   |
|-----|---------|---|
| 1   | Majikan | Aktor ini adalah seorang aktor / pengguna yang telah melakukan proses <i>login</i> dan diberikan akses kepada fitur – fitur yang diperuntukkan untuk majikan. |
| 2   | Pekerja | Merupakan seorang aktor yang telah melakukan proses <i>login</i> dan diberikan akses kepada fitur – fitur yang diperuntukkan untuk pekerja.                   |
| 3   | Admin   | Merupakan seorang aktor yang telah melakukan proses <i>login</i> dan mempunyai akses penuh pada sistem dan bisa melakukan perubahan pada sistem.              |



|   |          |   |
|---|----------|---|
| 4 | Midtrans | Merupakan seorang aktor yang memiliki hak akses melakukan perubahan status pemesanan. |
| 5 | Guest    | Adlah seorang aktor yang belum melakukan proses <i>login</i> ke dalam sistem.         |

#### 4.4 Daftar Kebutuhan Fungsional

Setelah melakukan analisis kebutuhan, maka disusunlah sebuah daftar kebutuhan fungsional yang bertujuan untuk menjelaskan kebutuhan apa saja yang berhasil di elisitasi. Nantinya, kebutuhan tersebut akan diberi kode PRT\_F\_XXX dengan arti PRT adalah sebagai kode nama sistem, F merupakan kode yang menandakan bahwa kebutuhan ini adalah kebutuhan dengan jenis fungsional, sedangkan XXX adalah nomor dari fungsi tersebut. Setelah melakukan analisis kebutuhan, ada tiga puluh sembilan kebutuhan fungsional yang didapatkan beserta spesifikasi kebutuhannya. Pada Tabel 4.2 akan dipaparkan seluruh kebutuhan fungsional yang ada beserta spesifikasinya.

**Tabel 4.2 Daftar Kebutuhan Fungsional**

| No. | Kode      | Nama Fungsi            | Deskripsi  | Aktor |
|-----|-----------|------------------------|--|-------|
| 1   | PRT_F_001 | Login                  | <p>Sistem yang akan dibuat harus mempunyai layanan untuk aktor dapat masuk kedalam sistem dan memberikan akses kepada fitur sesuai dengan jenis aktor.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Sistem harus bisa memperlihatkan halaman masuk yang berisi form email, dan password.</li> </ol> | Guest |
| 2   | PRT_F_002 | Daftar Sebagai Pekerja | <p>Sistem harus bisa menyediakan layanan daftar untuk membuat akun.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Sistem harus menyediakan sebuah form pendaftaran yang berisi field email, field password, field</li> </ol>   | Guest |



|   |           |                                      |  |         |
|---|-----------|--------------------------------------|--|---------|
|   |           |                                      | nama, nomor KTP, tempat tanggal lahir, nomor telepon, alamat, keahlian, agama, dan foto KTP.   |         |
| 3 | PRT_F_003 | Daftar Sebagai Majikan               | Sistem harus bisa menyediakan layanan daftar untuk membuat akun<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa menyediakan sebuah form pendaftaran yang berisi email, password, nama, nomor KTP, nomor telepon, alamat, dan foto KTP.                 | Guest   |
| 4 | PRT_F_004 | Masuk Sebagai Admin                  | Sistem harus bisa menyediakan layanan untuk bisa masuk ke dalam sistem dan mendapatkan fitur yang sesuai dengan ketentuan sistem.<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa memperlihatkan halaman masuk yang berisi form username dan password. | Guest   |
| 5 | PRT_F_005 | Melihat daftar pekerja yang tersedia | Sistem harus bisa menyediakan layanan untuk menampilkan daftar pekerja yang tersedia.<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa memperlihatkan halaman yang berisi data – data pekerja yang tersedia.  | Guest   |
| 6 | PRT_F_006 | Mengajukan Permohonan                | Sistem harus bisa menyediakan fungsi untuk   | Pekerja |



|   |           |                                   |   |         |
|---|-----------|-----------------------------------|---|---------|
|   |           | Resign                            | pekerja mengajukan permohonan resign  |         |
|   |           |                                   | <b>Spesifikasi Kebutuhan :</b><br>1. Pekerja hanya bisa mengajukan resign setelah seluruh pekerjaannya selesai.   |         |
| 7 | PRT_F_007 | Melihat daftar riwayat pekerjaan  | Sistem harus bisa menyediakan fungsi menampilkan riwayat pekerjaan dari pekerja tersebut.<br><b>Spesifikasi Kebutuhan :</b><br>1. Riwayat pekerjaan yang ditampilkan adalah tanggal masuk, tanggal keluar, majikan, dan rating. | Pekerja |
| 8 | PRT_F_008 | Melihat daftar riwayat penggajian | Sistem harus bisa menyediakan fungsi menampilkan riwayat penggajian dari pekerja tersebut.<br><b>Spesifikasi Kebutuhan :</b><br>1. Riwayat penggajian yang ditampilkan adalah tanggal penggajian, dan jumlah gaji.              | Pekerja |
| 9 | PRT_F_009 | Memesan jasa pekerja              | Sistem harus bisa menyediakan fungsi pemesanan jasa PRT<br><b>Spesifikasi Kebutuhan :</b><br>1. Memesan jasa PRT dengan memilih PRT yang diinginkan dan menekan tombol pesan<br>2. Sistem hanya menampilkan pekerja             | Majikan |





|    |           |                                  |  |         |
|----|-----------|----------------------------------|--|---------|
|    |           |                                  | dengan status aktif.   |         |
| 10 | PRT_F_010 | Claim garansi                    | Sistem harus bisa menyediakan fungsi untuk claim garansi.<br><b>Spesifikasi Kebutuhan :</b><br>1. Garansi bisa di claim jika masih dalam masa percobaan, masa percobaan adalah 1 bulan.<br>2. Claim garansi dilakukan dengan cara mengisi form yang berisi alasan claim. | Majikan |
| 11 | PRT_F_011 | Perpanjang masa kerja PRT        | Sistem harus bisa menyediakan fungsi untuk memperpanjang masa kerja pegawai rumah tangga.<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa menampilkan tombol perpanjangan PRT setiap bulannya.   | Majikan |
| 12 | PRT_F_012 | Rating Pekerja                   | Sistem harus bisa menyediakan fungsi untuk majikan memberikan rating kepada pekerja.<br><b>Spesifikasi Kebutuhan :</b><br>1. Majikan dapat memberikan rating kepada pekerja dari 1 sampai 5 bintang dengan menekan tombol rating.  | Majikan |
| 13 | PRT_F_013 | Melihat daftar riwayat pemesanan | Sistem harus bisa menyediakan fungsi menampilkan riwayat pemesanan dari majikan tersebut.<br><b>Spesifikasi Kebutuhan :</b>  | Majikan |



|    |           |                                      |  |   |         |
|----|-----------|--------------------------------------|--|---|---------|
|    |           |                                      |  | 1. Riwayat pemesanan yang ditampilkan adalah nama pekerja, tanggal pesan, jumlah pembayaran, status pembayaran, dan rating.   |         |
| 14 | PRT_F_014 | Melihat daftar riwayat garansi claim |  | Sistem harus bisa menyediakan fungsi untuk menampilkan riwayat claim garansi dari majikan tersebut.<br><b>Spesifikasi Kebutuhan :</b><br>1. Riwayat claim garansi yang ditampilkan adalah nama pekerja, tanggal pesan, status pemesanan, rating, waktu pengajuan garansi, status refund, dan alasan refund. | Majikan |
| 15 | PRT_F_015 | Melihat daftar admin                 |  | Sistem harus bisa menyediakan fitur yang ditujukan untuk menampilkan daftar admin yang sudah terdaftar di dalam sistem.<br><b>Spesifikasi Kebutuhan :</b><br>1. Data yang bisa dilihat adalah id, nama, dan email.  | Admin   |
| 16 | PRT_F_016 | Edit daftar admin                    |  | Sistem harus bisa mempersiapkan fitur untuk mengubah data admin yang tersimpan dalam database.<br><b>Spesifikasi Kebutuhan :</b><br>1. Data yang bisa diubah adalah data password, nama, dan email.   | Admin   |
| 17 | PRT_F_017 | Hapus daftar admin                   |  | Sistem harus bisa mempersiapkan fitur yang  | Admin   |



|    |           |                        |  |       |
|----|-----------|------------------------|--|-------|
|    |           |                        | mampu menghilangkan data admin dari dalam sistem.  |       |
|    |           |                        | <b>Spesifikasi Kebutuhan :</b><br>1. Admin dapat menghapus majikan dari daftar pekerja dengan menginputkan id majikan dan keterangan.  |       |
| 18 | PRT_F_018 | Melihat daftar pekerja | Sistem harus bisa menyediakan fitur yang ditujukan untuk menampilkan daftar pekerja yang sudah terdaftar di dalam sistem.  | Admin |
|    |           |                        | <b>Spesifikasi Kebutuhan :</b><br>1. Data yang bisa dilihat adalah email, nama, nomor KTP, tempat tanggal lahir, nomor telepon, alamat, keahlian, agama, dan foto KTP.                                       |       |
| 19 | PRT_F_019 | Edit daftar pekerja    | Sistem harus bisa menyediakan fungsi untuk mengedit data pekerja.  | Admin |
|    |           |                        | <b>Spesifikasi Kebutuhan :</b><br>1. Data yang bisa diubah adalah data gaji perbulan, rating pekerja, password, nama, nomor KTP, tempat tanggal lahir, nomor telepon, alamat, keahlian, agama, dan foto KTP. |       |
| 20 | PRT_F_020 | Hapus daftar pekerja   | Sistem harus bisa mempersiapkan fitur yang mampu menghilangkan data pekerja dari dalam sistem.   | Admin |



|    |           |                        |   |
|----|-----------|------------------------|---|
|    |           |                        | <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>Admin dapat menghapus pekerja dari daftar pekerja dengan menginputkan id pekerja dan keterangan.</li> </ol>  |
| 21 | PRT_E_021 | Melihat daftar majikan | <p>Sistem harus bisa menyediakan fitur yang ditujukan untuk menampilkan daftar majikan yang sudah terdaftar di dalam sistem..</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>Data yang bisa dilihat adalah id, nama, email, nomor KTP, no telpon, dan alamat</li> </ol> |
| 22 | PRT_E_022 | Edit daftar majikan    | <p>Sistem harus bisa menyediakan fungsi untuk mengedit data majikan.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>Data yang bisa diubah adalah data password, rating majikan, nama, nomor KTP, nomor telepon, alamat, dan foto KTP.</li> </ol>                        |
| 23 | PRT_E_023 | Hapus daftar majikan   | <p>Sistem harus bisa mempersiapkan fitur yang mampu menghilangkan data majikan dari dalam sistem.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>Admin dapat menghapus majikan dari daftar pekerja dengan menginputkan id majikan dan</li> </ol>                        |



|    |           |                                   | keterangan.  |       |
|----|-----------|-----------------------------------|--|-------|
| 24 | PRT_F_024 | Tambah admin                      | Sistem harus bisa menyiapkan fungsi untuk menambah admin baru.<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa mempersiapkan form yang berisi field nama, email, dan password.   | Admin |
| 25 | PRT_F_025 | Tambah majikan                    | Sistem harus bisa menyiapkan fungsi untuk menambah majikan baru.<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa menyiapkan form yang berisi email, password, nama, nomor KTP, nomor telepon, alamat, dan foto KTP.  | Admin |
| 26 | PRT_F_026 | Tambah pekerja                    | Sistem harus bisa menyiapkan fungsi untuk menambah pekerja baru.<br><b>Spesifikasi Kebutuhan :</b><br>1. Sistem harus bisa mempersiapkan form dengan field email, password, nama, nomor KTP, tempat tanggal lahir, nomor telepon, alamat, keahlian, kondisi khusus, agama, dan foto KTP. | Admin |
| 27 | PRT_F_027 | Melihat riwayat pekerjaan pekerja | Sistem harus bisa menyiapkan fungsi untuk menampilkan riwayat pekerjaan dari pekerja.<br><b>Spesifikasi Kebutuhan :</b>  | Admin |



|    |           |                                   |  |       |
|----|-----------|-----------------------------------|--|-------|
|    |           |                                   | 1. Riwayat yang bisa dilihat adalah waktu pesan, waktu bayar, jumlah pembayaran, status pembayaran, status pemesanan, dan rating.  |       |
| 28 | PRT_F_028 | Melihat riwayat transaksi majikan | Sistem harus bisa menyediakan fungsi untuk melihat <i>track record</i> dari majikan.<br><b>Spesifikasi Kebutuhan :</b><br>1. Riwayat yang bisa dilihat adalah waktu pesan, waktu bayar, jumlah pembayaran, status pembayaran, status pemesanan, dan rating.        | Admin |
| 29 | PRT_F_029 | Konfirmasi pendaftaran pekerja    | Sistem harus bisa menyiapkan fungsi untuk mengkonfirmasi pendaftaran pekerja.<br><b>Spesifikasi Kebutuhan :</b><br>1. Jika pendaftaran pekerja dikonfirmasi, maka pekerja tersebut akan langsung masuk ke dalam daftar pekerja.                                    | Admin |
| 30 | PRT_F_030 | Melihat riwayat pemesanan         | Sistem harus bisa menyiapkan fungsi untuk menampilkan riwayat pemesanan yang dilakukan oleh majikan.<br><b>Spesifikasi Kebutuhan :</b><br>1. Riwayat pemesanan yang bisa dilihat adalah nama majikan, nama pekerja, waktu pesan, waktu bayar, jumlah bayar, status | Admin |



|    |           |                                    |  |       |
|----|-----------|------------------------------------|--|-------|
|    |           |                                    | pemesanan, status pembayaran, dan rating.  | Admin |
| 31 | PRT_F_031 | Penggajian Pekerja                 | <p>Sistem harus bisa menyediakan fungsi untuk menggaji pekerja setiap bulannya</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Penggajian pekerja dilakukan dengan menekan tombol gaji jika gaji pekerja sudah dibayarkan sesuai jumlah yang tertera didalam sistem.</li> </ol>  | Admin |
| 32 | PRT_F_032 | Melihat riwayat penggajian pekerja | <p>Sistem harus bisa menyediakan fungsi untuk melihat riwayat penggajian pekerja.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Riwayat penggajian yang bisa dilihat adalah nama pekerja, waktu pemesanan, jumlah pembayaran, status pembayaran, dan waktu pembayaran.</li> </ol>  | Admin |
| 33 | PRT_E_033 | Konfirmasi claim garansi.          | <p>Sistem harus bisa menyediakan fungsi untuk mengkonfirmasi claim garansi yang masuk kedalam sistem.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Jika claim garansi diterima, maka akan dicarikan PRT baru dengan kriteria yang sesuai dan akan tercatat dalam sistem.</li> <li>2. Jika claim garansi ditolak, maka uang</li> </ol> | Admin |



|    |           |                               |   |       |
|----|-----------|-------------------------------|---|-------|
|    |           |                               | administrasi akan dikembalikan dan status pemesanan menjadi selesai dan status pekerja menjadi tersedia kembali.  |       |
| 34 | PRT_F_034 | Melihat riwayat claim garansi | <p>Sistem harus bisa menyediakan fungsi untuk melihat riwayat claim garansi.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Riwayat claim garansi yang bisa dilihat adalah nama pekerja, waktu pemesanan, status pemesanan, rating, waktu pengajuan garansi, status refund, alasan refund, waktu konfirmasi, dan id pesanan baru.</li> </ol> | Admin |
| 35 | PRT_F_035 | Konfirmasi resign pekerja     | <p>Sistem harus bisa menyediakan fungsi untuk mengkonfirmasi resign pekerja yang masuk kedalam sistem.</p> <p><b>Spesifikasi Kebutuhan :</b></p> <ol style="list-style-type: none"> <li>1. Jika alasan resign sudah jelas, maka admin dapat menekan tombol konfirmasi lalu pekerja yang mengajukan resign akan dihapus dari daftar pekerja aktif.</li> </ol>                      | Admin |
| 36 | PRT_F_036 | Blacklist Pekerja             | Sistem harus bisa menyiapkan fungsi untuk memblacklist pekerja yang sebelumnya terdaftar dalam  | Admin |





|    |           |                           |  |                         |
|----|-----------|---------------------------|--|-------------------------|
|    |           |                           | sistem.  |                         |
|    |           |                           | <b>Spesifikasi Kebutuhan :</b><br>1. Jika pekerja di blacklist maka tidak akan terdaftar di pemesanan pekerja.   |                         |
| 37 | PRT_F_037 | Blacklist Majikan         | Sistem harus bisa menyiapkan fungsi untuk memblacklist pekerja yang terdaftar di dalam sistem.<br><b>Spesifikasi Kebutuhan :</b><br>1. Jika majikan di blacklist maka tidak akan bisa memesan pekerja. | Admin                   |
| 38 | PRT_F_038 | Mengubah status pemesanan | Sistem harus mampu merubah status pemesanan<br><b>Spesifikasi Kebutuhan :</b><br>1. Status-status perubahan yang diubah diantaranya <i>capture</i> (sukses), <i>pending</i> , dan <i>expire</i> .      | Midtrans                |
| 39 | PRT_F_039 | Logout                    | Sistem harus mampu mempersiapkan sebuah fitur yang berguna untuk mengeluarkan dan mencabut hak aktor.  | Admin, Pekerja, Majikan |

#### 4.5 Daftar Kebutuhan Non-Fungsional

Setelah melakukan analisis kebutuhan, maka disusunlah sebuah daftar kebutuhan non-fungsional yang bertujuan untuk menjelaskan kebutuhan apa saja yang berhasil di elisitasi. Nantinya, kebutuhan tersebut akan diberi kode PRT\_NF\_XXX dengan arti PRT adalah sebagai kode nama sistem yang dikembangkan, NF merupakan kode yang menandakan bahwa kebutuhan ini adalah kebutuhan dengan jenis non fungsional, sedangkan XXX adalah nomor dari fungsi tersebut. Setelah melakukan analisis kebutuhan, ada satu kebutuhan non-fungsional yang didapatkan, kebutuhan tersebut adalah *compatibility*. *Compatibility* dibutuhkan karena pihak CV. Sitiari, majikan, dan pekerja akan menggunakan sistem ini pada berbagai perangkat seperti handphone, tablet,



maupun komputer. Untuk itu dibutuhkan kebutuhan yang bias menjamin system dapat dijalankan pada berbagai macam perangkat. Pada Tabel 4.3 akan dipaparkan seluruh kebutuhan non-fungsional yang ada beserta spesifikasinya.

**Tabel 4.3 Daftar Kebutuhan Non-Fungsional**

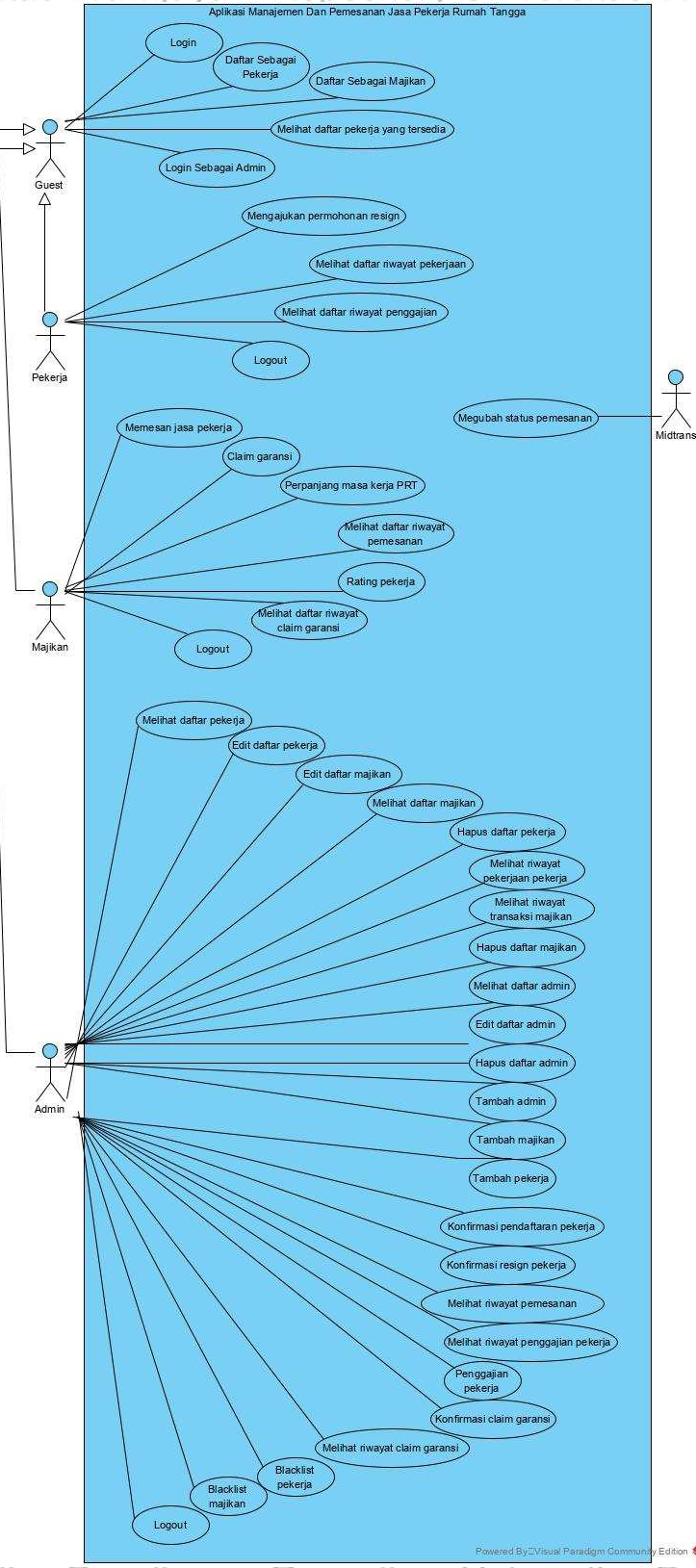
| No. | Kode       | Parameter            | Deskripsi  |
|-----|------------|----------------------|--|
| 1   | PRT_NF_001 | <i>Compatibility</i> | Sistem yang dibuat harus memastikan dapat dijalankan di berbagai perambah yang berbeda seperti perambah Safari, perambah firefox, Google Chrome, perambah edge, perambah pada perangkat android, dan perambah pada perangkat iOS |

#### 4.6 Pemodelan Kebutuhan

Pada tahapan ini akan dilakukan pemodelan dari kebutuhan yang sudah didapatkan. Dalam tahapan ini akan ada dua sub bab yang nantinya akan menjabarkan *use case diagram* dan *use case scenario*.

##### 4.6.1 Use Case Diagram

Diagram ini merupakan sebuah diagram yang berfungsi untuk memvisualisasikan interaksi yang terjadi diantara aktor dengan fungsional sistem. Fungsi yang nantinya akan divisualisasikan di dalam use case diagram akan sepenuhnya mengikuti kebutuhan fungsional yang sebelumnya telah di jabarkan. Tujuan utama dibuatnya use case diagram adalah untuk memudahkan dalam memahami, membaca, dan melihat fungsionalitas sistem yang akan dikembangkan. Gambar 4.12 akan menjabarkan diagram use case dari sistem yang nantinya akan dikembangkan.



Gambar 4.12 Diagram Use Case Sistem Manajemen dan Pemesanan Pekerja Rumah Tangga



#### 4.6.2 Use Case Scenario

Tahapan ini berfungsi untuk mendefinisikan dan juga menjelaskan tahapan yang akan dilalui saat aktor akan menjalankan suatu fungsi. Tabel 4.4 sampai dengan Tabel 4.42 akan menjabarkan skenario dari tiap fungsionalitas.

Tabel 4.4 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi *Login* terjadi.

**Tabel 4.2 Use Case Scenario Login**

|                         |   |
|-------------------------|---|
| <i>Code</i>             | PRT_F_001   |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk memasukkan <i>user</i> kedalam sistem agar bisa menggunakan fungsi – fungsi sistem.   |
| <i>Actor</i>            | <i>Guest</i>  |
| <i>Pre-condition</i>    | Aktor telah mengakses halaman <i>Login</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor memasukkan <i>email</i> dan <i>password</i> yang sebelumnya sudah didaftarkan.</li> <li>2. Sistem melakukan pengecekan apakah akun sudah terdaftar atau belum.</li> <li>3. Sistem mengarahkan aktor ke halaman <i>home</i></li> </ol> |
| <i>Alternative Flow</i> | <ol style="list-style-type: none"> <li>1.1 Apabila email dan pasword tidak diinputkan maka akan muncul pesan “Silahkan isi field”.</li> <li>2.1 Apabila aktor belum terdaftar, maka sistem akan menampilkan pesan “Data Tidak Ditemukan”.</li> </ol>                                  |
| <i>Post-condition</i>   | Aktor masuk ke dalam sistem.  |

Tabel 4.5 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi daftar sebagai pekerja terjadi.

**Tabel 4.3 Use Case Scenario Daftar Sebagai Pekerja**

|                       |  |
|-----------------------|--|
| <i>Kode Kebutuhan</i> | PRT_F_002  |
| <i>Objective</i>      | <i>Use case</i> ini bertujuan mendaftarkan aktor kedalam sistem sebagai pekerja.   |
| <i>Actor</i>          | <i>Guest</i>   |
| <i>Pre-condition</i>  | Aktor telah berada pada halaman <i>register</i> pekerja.   |
| <i>Main Flow</i>      | <ol style="list-style-type: none"> <li>1. Aktor mengisikan data yang berisi email, password, nama, nomor KTP, tempat tanggal lahir, nomor telepon, alamat, keahlian, kondisi khusus, agama, dan foto KTP.</li> </ol> |



|                         |   |
|-------------------------|---|
|                         | <ol style="list-style-type: none"> <li>2. Aktor menekan tombol daftar.</li> <li>3. Sistem melakukan pengecekan apakah seluruh data sudah diisi dengan benar.</li> <li>4. Sistem mendaftarkan aktor kedalam sistem.</li> <li>5. Sistem mengarahkan aktor ke halaman <i>Login</i>.</li> </ol> |
| <i>Alternative Flow</i> | 3.1 Jika data belum diisi semua maka sistem akan memberikan peringatan bahwa data harus diisi.  |
| <i>Post-condition</i>   | Aktor terdaftar kedalam sistem dan diarahkan ke halaman <i>Login</i> .  |

Tabel 4.6 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi daftar sebagai majikan terjadi.

**Tabel 4.6 Use Case Scenario Daftar Sebagai Majikan**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_003  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan mendaftarkan aktor kedalam sistem sebagai majikan.   |
| <i>Actor</i>            | <i>Guest</i>   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>register</i> majikan.   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor mengisikan data yang berisi email, password, nama, nomor KTP, nomor telepon, alamat, dan foto KTP.</li> <li>2. Aktor menekan tombol daftar.</li> <li>3. Sistem melakukan pengecekan apakah seluruh data sudah diisi dengan benar.</li> <li>4. Sistem mendaftarkan aktor kedalam sistem.</li> <li>5. Sistem mengarahkan aktor ke halaman <i>Login</i>.</li> </ol> |
| <i>Alternative Flow</i> | 3.2 Jika data belum diisi semua maka sistem akan memberikan peringatan bahwa ada data yang belum terisi.   |
| <i>Post-condition</i>   | Aktor terdaftar kedalam sistem dan diarahkan ke halaman <i>Login</i> .   |

Tabel 4.7 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi masuk sebagai admin

**Tabel 4.7 Use Case Scenario Masuk Sebagai Admin**



|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_004   |
| <i>Objective</i>        | Use case ini bertujuan untuk memasukkan aktor kedalam sistem sebagai admin.   |
| <i>Actor</i>            | Guest   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>Login admin</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor mengisikan <i>form Login</i> berupa <i>email</i> dan <i>password</i>.</li> <li>2. Aktor menekan tombol <i>Login</i>.</li> <li>3. Sistem mengarahkan aktor ke halaman <i>dashboard admin</i>.</li> </ol> |
| <i>Alternative Flow</i> | <ol style="list-style-type: none"> <li>1.1 Apabila email dan pasword tidak diinputkan maka akan muncul pesan "Silahkan isi field".</li> <li>3.1 Apabila aktor belum terdaftar, maka sistem akan menampilkan pesan "Data Tidak Ditemukan".</li> </ol>    |
| <i>Post-condition</i>   | Sistem memasukkan aktor ke dalam sistem sebagai <i>admin</i> .  |

Tabel 4.8 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar pekerja yang tersedia terjadi.

**Tabel 4.8 Use Case Scenario Melihat Daftar Pekerja Yang Tersedia**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_005   |
| <i>Objective</i>        | Use case ini bertujuan untuk menampilkan daftar pekerja yang tersedia.  |
| <i>Actor</i>            | Guest   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman utama.  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol home.</li> <li>2. Sistem menampilkan daftar pekerja yang tersedia kepada aktor.</li> </ol> |
| <i>Alternative Flow</i> |   |
| <i>Post-condition</i>   | Sistem menampilkan daftar pekerja yang tersedia.  |

Tabel 4.9 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi mengajukan permohonan resign terjadi.

**Tabel 4.9 Use Case Scenario Mengajukan Permohonan Resign**

|                |           |
|----------------|-----------|
| Kode Kebutuhan | PRT_F_006 |
|----------------|-----------|



|                         |  |
|-------------------------|--|
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk pekerja mengajukan permohonan <i>resign</i> .                |
| <i>Actor</i>            | Pekerja  |
| <i>Pre-condition</i>    | Aktor berada pada halaman <i>resign</i>  |
| <i>Main Flow</i>        | 1. Aktor menekan tombol <i>resign</i> .<br>2. Sistem melakukan pencatatan <i>resign</i> pekerja. |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem mencatat permohonan <i>resign</i> pekerja.  |

Tabel 4.10 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar riwayat pekerjaan terjadi.

**Tabel 4.10 Use Case Scenario Melihat Daftar Riwayat Pekerjaan**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_007  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat riwayat pekerjaan                          |
| <i>Actor</i>            | Pekerja  |
| <i>Pre-condition</i>    | Aktor berada pada halaman pekerja.   |
| <i>Main Flow</i>        | 1. Aktor menekan tombol riwayat pekerjaan.<br>2. Sistem menampilkan riwayat pekerjaan. |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem menampilkan riwayat pekerjaan.  |

Tabel 4.11 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar riwayat penggajian terjadi.

**Tabel 4.11 Use Case Scenario Melihat Daftar Riwayat Penggajian**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_008  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat riwayat penggajian.                          |
| <i>Actor</i>            | Pekerja  |
| <i>Pre-condition</i>    | Aktor berada pada halaman pekerja.   |
| <i>Main Flow</i>        | 1. Aktor menekan tombol riwayat penggajian.<br>2. Sistem menampilkan riwayat penggajian. |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem menampilkan riwayat penggajian.   |



Tabel 4.12 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi memesan jasa pekerja terjadi.

**Tabel 4.12 Use Case Scenario Memesan Jasa Pekerja**

|                  |  |
|------------------|--|
| Kode Kebutuhan   | PRT_F_009  |
| Objective        | Use case ini bertujuan untuk melakukan pemesanan pekerja rumah tangga melalui sistem.  |
| Actor            | Majikan  |
| Pre-condition    | Aktor telah berada pada halaman dashboard  |
| Main Flow        | <ol style="list-style-type: none"> <li>1. Aktor memilih jenis PRT yang akan dipesan.</li> <li>2. Sistem menampilkan daftar PRT dengan jenis yang sudah dipilih.</li> <li>3. Aktor memilih PRT yang ingin dipesan.</li> <li>4. Sistem menyimpan data pemesanan kedalam database.</li> <li>5. Sistem mengarahkan aktor ke halaman pembayaran.</li> </ol> |
| Alternative Flow | 4.1 Jika pemesanan gagal, maka akan muncul pesan error.  |
| Post-condition   | Aktor berhasil memesan pekerja.  |

Tabel 4.13 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi claim garansi terjadi.

**Tabel 4.13 Use Case Scenario Claim Garansi**

|                |  |
|----------------|--|
| Kode Kebutuhan | PRT_F_010  |
| Objective      | Use case ini bertujuan untuk melakukan claim garansi pada pemesanan pekerja.   |
| Actor          | Majikan  |
| Pre-condition  | Aktor telah berada pada halaman claim garansi.   |
| Main Flow      | <ol style="list-style-type: none"> <li>1. Aktor memilih transaksi yang ingin di claim garansi.</li> <li>2. Aktor mengisikan alasan untuk claim garansi.</li> <li>3. Sistem akan mencatat claim garansi tersebut ke dalam database untuk nantinya di verifikasi oleh</li> </ol> |





|                         |  |
|-------------------------|--|
|                         | admin.   |
| <i>Alternative Flow</i> | 2.1 Jika aktor tidak mengisikan alasan claim, maka sistem akan menampilkan pesan “Silahkan isi field”. |
| <i>Post-condition</i>   | Aktor diarahkan ke halaman claim garansi.  |

Tabel 4.14 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi perpanjangan masa kerja PRT terjadi.

**Tabel 4.14 Use Case Scenario Perpanjangan Masa Kerja PRT**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_011  |
| <i>Objective</i>        | Use case ini bertujuan untuk melakukan perpanjangan masa kerja PRT.  |
| <i>Actor</i>            | Majikan  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman perpanjangan masa kerja.   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor memilih pekerja yang ingin diperpanjang masa kerjanya.</li> <li>2. Sistem menyimpan data perpanjangan pekerja.</li> <li>3. Sistem menampilkan halaman pembayaran.</li> </ol> |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Aktor berhasil melakukan perpanjangan masa kerja.  |

Tabel 4.15 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi rating pekerja terjadi.

**Tabel 4.15 Use Case Scenario Rating Pekerja**

|                      |   |
|----------------------|---|
| Kode Kebutuhan       | PRT_F_012   |
| <i>Objective</i>     | Use case ini bertujuan untuk memberikan rating kepada pekerja yang telah selesai atau sedang bekerja.   |
| <i>Actor</i>         | Majikan   |
| <i>Pre-condition</i> | Aktor telah berada pada halaman rating pekerja.   |
| <i>Main Flow</i>     | <ol style="list-style-type: none"> <li>1. Aktor memilih pekerja yang ingin diberikan rating.</li> <li>2. Aktor memilih rating untuk pekerja tersebut dan menekan tombol submit.</li> <li>3. Sistem akan menyimpan rating dari aktor ke dalam</li> </ol> |



|                         |   |
|-------------------------|---|
|                         | <i>database.</i>                                  |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Aktor diarahkan ke halaman <i>rating</i> pekerja. |

Tabel 4.16 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar riwayat pemesanan terjadi.

**Tabel 4.16 Use Case Scenario Melihat Daftar Riwayat Pemesanan**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_013  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat daftar riwayat pemesanan majikan yang sedang Login.  |
| <i>Actor</i>            | Majikan  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol riwayat pemesanan.</li> <li>2. Sistem menampilkan data riwayat pemesanan aktor ke dalam halaman riwayat pemesanan.</li> </ol> |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem menampilkan daftar riwayat pemesanan majikan yang sedang Login.   |

Tabel 4.17 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar riwayat claim garansi terjadi.

**Tabel 4.17 Use Case Scenario Melihat Daftar Riwayat Claim Garansi**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_014  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat daftar riwayat claim garansi majikan yang sedang Login.  |
| <i>Actor</i>            | Majikan  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol riwayat claim garansi.</li> <li>2. Sistem menampilkan data riwayat claim garansi aktor ke dalam halaman riwayat claim garansi.</li> </ol> |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem menampilkan daftar riwayat claim garansi majikan yang sedang Login.   |



Tabel 4.18 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar admin terjadi.

**Tabel 4.18 Use Case Scenario Melihat Daftar Admin**

|                  |  |
|------------------|--|
| Kode Kebutuhan   | PRT_F_015  |
| Objective        | Use case ini bertujuan untuk menampilkan daftar admin kepada aktor.  |
| Actor            | Admin  |
| Pre-condition    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| Main Flow        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar admin.</li> <li>2. Sistem menampilkan daftar admin ke halaman daftar admin.</li> </ol> |
| Alternative Flow | _  |
| Post-condition   | Sistem menampilkan daftar admin.   |

Tabel 4.19 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi edit daftar admin terjadi.

**Tabel 4.19 Use Case Scenario Edit Daftar Admin**

|                  |   |
|------------------|---|
| Kode Kebutuhan   | PRT_F_016   |
| Objective        | Use case ini bertujuan untuk mengubah data diri admin.  |
| Actor            | Admin   |
| Pre-condition    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| Main Flow        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar admin.</li> <li>2. Aktor menekan ikon edit.</li> <li>3. Aktor menginputkan data baru.</li> <li>4. Aktor menekan tombol submit.</li> <li>5. Sistem menyimpan data baru ke dalam database.</li> </ol> |
| Alternative Flow | _   |
| Post-condition   | Sistem menampilkan daftar admin dengan data yang telah diedit sebelumnya.   |

Tabel 4.20 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi hapus daftar admin terjadi.



Tabel 4.20 Use Case Scenario Hapus Daftar Admin

|                  |   |
|------------------|---|
| Kode Kebutuhan   | PRT_F_017   |
| Objective        | Use case ini bertujuan untuk menghapus data admin.  |
| Actor            | Admin   |
| Pre-condition    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| Main Flow        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar admin.</li> <li>2. Aktor menekan ikon hapus.</li> <li>3. Aktor menekan tombol submit.</li> <li>4. Sistem menghapus admin terpilih dari dalam database.</li> </ol> |
| Alternative Flow |   |
| Post-condition   | Sistem menampilkan daftar admin setelah terjadi penghapusan.  |

Tabel 4.21 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar pekerja terjadi.

Tabel 4.21 Use Case Scenario Melihat Daftar Pekerja

|  |  |
|--|--|
| <i>Flow of Events for</i> melihat daftar pekerja |  |
| Kode Kebutuhan                                   | PRT_F_018  |
| Objective  | Use case ini bertujuan untuk menampilkan daftar pekerja kepada aktor.  |
| Actor  | Admin  |
| Pre-condition                                    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| Main Flow  | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar pekerja.</li> <li>2. Sistem menampilkan daftar pekerja ke halaman daftar pekerja.</li> </ol> |
| Alternative Flow                                 |  |
| Post-condition                                   | Sistem menampilkan daftar pekerja.   |

Tabel 4.22 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi edit daftar pekerja terjadi.

Tabel 4.22 Use Case Scenario Edit Daftar Pekerja

|                |           |
|----------------|-----------|
| Kode Kebutuhan | PRT_F_019 |
|----------------|-----------|



|                         |   |
|-------------------------|---|
| <i>Objective</i>        | Use case ini bertujuan untuk mengubah data diri pekerja.  |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar pekerja.</li> <li>2. Sistem menampilkan daftar pekerja.</li> <li>3. Aktor menekan ikon edit.</li> <li>4. Sistem menampilkan form edit pekerja.</li> <li>5. Aktor menginputkan data baru lalu menekan tombol submit.</li> <li>6. Sistem menampilkan pesan "data berhasil diubah".</li> </ol> |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Sistem menampilkan daftar pekerja dengan data yang telah diedit sebelumnya.   |

Tabel 4.23 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi hapus daftar pekerja terjadi.

**Tabel 4.23 Use Case Scenario Hapus Daftar Pekerja**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_020   |
| <i>Objective</i>        | Use case ini bertujuan untuk menghapus data pekerja.  |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar pekerja.</li> <li>2. Aktor menekan ikon hapus.</li> <li>3. Aktor menekan tombol submit.</li> <li>4. Sistem menghapus pekerja terpilih dari dalam database.</li> </ol> |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Sistem menampilkan daftar pekerja setelah terjadi penghapusan.  |

Tabel 4.24 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat daftar majikan terjadi.

**Tabel 4.24 Use Case Scenario Melihat Daftar Majikan**



|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_021  |
| <i>Objective</i>        | Use case ini bertujuan untuk menampilkan daftar majikan kepada aktor.  |
| <i>Actor</i>            | Admin  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar majikan.</li> <li>2. Sistem menampilkan daftar majikan ke halaman daftar pekerja.</li> </ol> |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem menampilkan daftar majikan.   |

Tabel 4.25 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi edit daftar majikan terjadi.

**Tabel 4.25 Use Case Scenario Edit Daftar Majikan**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_022   |
| <i>Objective</i>        | Use case ini bertujuan untuk mengubah data diri majikan.  |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar majikan.</li> <li>2. Aktor menekan ikon edit.</li> <li>3. Aktor menginputkan data baru.</li> <li>4. Aktor menekan tombol submit.</li> <li>5. Sistem menyimpan data baru ke dalam database.</li> </ol> |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Sistem menampilkan daftar majikan dengan data yang telah diedit sebelumnya.   |

Tabel 4.26 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi hapus daftar majikan terjadi.

**Tabel 4.26 Use Case Scenario Hapus Daftar Majikan**

|                  |  |
|------------------|--|
| Kode Kebutuhan   | PRT_F_023  |
| <i>Objective</i> | Use case ini bertujuan untuk menghapus data majikan. |
| <i>Actor</i>     | Admin  |



|                         |   |
|-------------------------|---|
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar majikan.</li> <li>2. Aktor menekan ikon hapus.</li> <li>3. Aktor menekan tombol submit.</li> <li>4. Sistem menghapus majikan terpilih dari dalam database.</li> </ol> |
| <i>Alternative Flow</i> |   |
| <i>Post-condition</i>   | Sistem menampilkan daftar majikan setelah terjadi penghapusan.  |

Tabel 4.27 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi tambah admin terjadi.

**Tabel 4.27 Use Case Scenario Tambah Admin**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_024  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk menambahkan admin.   |
| <i>Actor</i>            | Admin  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar admin.</li> <li>2. Aktor menekan ikon tambah admin.</li> <li>3. Sistem menampilkan form tambah admin.</li> <li>4. Aktor mengisi data sesuai dengan form yang tersedia.</li> <li>5. Aktor menekan tombol submit.</li> <li>6. Sistem menyimpan data admin ke dalam <i>database</i>.</li> </ol> |
| <i>Alternative Flow</i> | 4.1 Jika aktor tidak mengisi data pada form, maka sistem akan menampilkan pesan "Silahkan isi field".  |
| <i>Post-condition</i>   | Sistem menampilkan daftar admin setelah terjadi penambahan admin.  |

Tabel 4.28 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi tambah majikan terjadi.



Tabel 4.28 Use Case Scenario Tambah Majikan

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_025   |
| <i>Objective</i>        | Use case ini bertujuan untuk menambahkan majikan.   |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman admin <i>dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar majikan.</li> <li>2. Aktor menekan ikon tambah majikan.</li> <li>3. Sistem menampilkan form tambah majikan.</li> <li>4. Aktor mengisikan data sesuai dengan form yang tersedia.</li> <li>5. Aktor menekan tombol submit.</li> <li>6. Sistem menyimpan data majikan ke dalam <i>database</i>.</li> </ol> |
| <i>Alternative Flow</i> | 4.2 Jika aktor tidak mengisikan data pada form, maka sistem akan menampilkan pesan "Silahkan isi field".  |
| <i>Post-condition</i>   | Sistem menampilkan daftar majikan setelah terjadi penambahan majikan.   |

Tabel 4.29 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi tambah pekerja terjadi.

Tabel 4.29 Use Case Scenario Tambah Pekerja

|                      |   |
|----------------------|---|
| Kode Kebutuhan       | PRT_F_026   |
| <i>Objective</i>     | Use case ini bertujuan untuk menambahkan pekerja.   |
| <i>Actor</i>         | Admin   |
| <i>Pre-condition</i> | Aktor telah berada pada halaman admin <i>dashboard</i> .  |
| <i>Main Flow</i>     | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar pekerja.</li> <li>2. Aktor menekan ikon tambah pekerja.</li> <li>3. Sistem menampilkan form tambah pekerja.</li> <li>4. Aktor mengisikan data sesuai dengan form yang tersedia.</li> <li>5. Aktor menekan tombol submit.</li> <li>6. Sistem menyimpan data pekerja ke dalam <i>database</i>.</li> </ol> |





|                         |  |
|-------------------------|--|
| <i>Alternative Flow</i> | 4.3 Jika aktor tidak mengisikan data pada form, maka sistem akan menampilkan pesan “Silahkan isi field”. |
| <i>Post-condition</i>   | Sistem menampilkan daftar pekerja setelah terjadi penambahan pekerja.                                    |

Tabel 4.30 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat riwayat pekerjaan pekerja terjadi.

**Tabel 4.30 Use Case Scenario Melihat Riwayat Pekerjaan Pekerja**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_027   |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat riwayat pekerjaan yang pernah dilakukan oleh pekerja.   |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar pekerja.</li> <li>2. Aktor menekan tombol transaksi pada pekerja terpilih.</li> <li>3. Sistem menampilkan riwayat pekerja.</li> </ol> |
| <i>Alternative Flow</i> |   |
| <i>Post-condition</i>   | Sistem menampilkan riwayat pekerjaan pekerja.   |

Tabel 4.31 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat riwayat transaksi majikan terjadi.

**Tabel 4.31 Use Case Scenario Melihat Riwayat Transaksi Majikan**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_028   |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat riwayat transaksi yang dilakukan oleh majikan.  |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol daftar majikan.</li> <li>2. Aktor menekan tombol transaksi pada majikan terpilih.</li> <li>3. Sistem menampilkan riwayat transaksi majikan.</li> </ol> |
| <i>Alternative Flow</i> |   |



|                       |  |
|-----------------------|--|
| <i>Post-condition</i> | Sistem menampilkan riwayat transaksi majikan majikan |
|-----------------------|--|

Tabel 4.32 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi konfirmasi pendaftaran pekerja terjadi.

**Tabel 4.32 Use Case Scenario Konfirmasi Pendaftaran Pekerja**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_029   |
| <i>Objective</i>        | Use case ini bertujuan untuk mengkonfirmasi pekerja yang mendaftar.   |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol pendaftaran pekerja.</li> <li>2. Sistem menampilkan daftar pekerja yang mendaftar dan butuh konfirmasi.</li> <li>3. Aktor menekan tombol konfirmasi pada pendaftar yang terpilih.</li> <li>4. Sistem menyimpan data pekerja dengan status tersedia.</li> </ol> |
| <i>Alternative Flow</i> | 4.1 Jika data pekerja tidak ditemukan maka akan muncul pesan 'Terjadi Kesalahan'.   |
| <i>Post-condition</i>   | Sistem mengkonfirmasi pendaftaran pekerja yang masuk.   |

Tabel 4.33 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat riwayat pemesanan terjadi.

**Tabel 4.33 Use Case Scenario Melihat Riwayat Pemesanan**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_030   |
| <i>Objective</i>        | Use case ini bertujuan untuk melihat riwayat pemesanan majikan.   |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol riwayat pemesanan</li> <li>2. Sistem menampilkan riwayat pemesanan majikan.</li> </ol> |
| <i>Alternative Flow</i> |   |
| <i>Post-condition</i>   | Sistem menampilkan riwayat pemesanan majikan.   |



Tabel 4.34 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi penggajian pekerja terjadi.

**Tabel 4.34 Use Case Scenario Penggajian Pekerja**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_031  |
| <i>Objective</i>        | Use case ini bertujuan untuk mencatat pembayaran gaji kepada pekerja.  |
| <i>Actor</i>            | Admin  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol penggajian pekerja.</li> <li>2. Aktor menekan tombol gaji pada pekerja yang dipilih.</li> <li>3. Sistem menyimpan data penggajian ke dalam database.</li> </ol> |
| <i>Alternative Flow</i> |  |
| <i>Post-condition</i>   | Sistem menyimpan data penggajian ke dalam database.  |

Tabel 4.35 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat riwayat penggajian pekerja terjadi.

**Tabel 4.35 Use Case Scenario Melihat Riwayat Penggajian Pekerja**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_032  |
| <i>Objective</i>        | Use case ini bertujuan untuk melihat riwayat penggajian pekerja.   |
| <i>Actor</i>            | Admin  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol rekap gaji.</li> <li>2. Sistem menampilkan riwayat penggajian pekerja.</li> </ol> |
| <i>Alternative Flow</i> |  |
| <i>Post-condition</i>   | Sistem menampilkan riwayat penggajian pekerja.   |

Tabel 4.36 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi konfirmasi claim garansi terjadi.

**Tabel 4.36 Use Case Scenario Konfirmasi Claim Garansi**

|                |           |
|----------------|-----------|
| Kode Kebutuhan | PRT_F_033 |
|----------------|-----------|



|                         |  |
|-------------------------|--|
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk mengkonfirmasi claim garansi.  |
| <i>Actor</i>            | Admin  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol claim garansi.</li> <li>2. Sistem menampilkan daftar claim garansi.</li> <li>3. Aktor menekan tombol konfirmasi pada salah satu daftar.</li> <li>4. Aktor mengisikan pekerja baru untuk menggantikan lalu menekan tombol ok.</li> <li>5. Sistem menyimpan data claim garansi sebagai "diterima".</li> </ol> |
| <i>Alternative Flow</i> | <ol style="list-style-type: none"> <li>3.1.1 Aktor menekan tombol tolak</li> <li>3.1.2 Sistem menyimpan data claim garansi sebagai "ditolak" dan mengubah data status pekerja menjadi "tersedia".</li> </ol>   |
| <i>Post-condition</i>   | Sistem menyimpan data claim garansi ke dalam database.   |

Tabel 4.37 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi melihat riwayat claim garansi terjadi.

**Tabel 4.37 Use Case Scenario Melihat Riwayat Claim Garansi**

|                         |  |
|-------------------------|--|
| Kode Kebutuhan          | PRT_F_034  |
| <i>Objective</i>        | <i>Use case</i> ini bertujuan untuk melihat riwayat claim garansi.   |
| <i>Actor</i>            | Admin  |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .   |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol riwayat claim garansi.</li> <li>2. Sistem menampilkan riwayat claim garansi majikan.</li> </ol> |
| <i>Alternative Flow</i> | -  |
| <i>Post-condition</i>   | Sistem menampilkan riwayat claim garansi majikan.  |

Tabel 4.38 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi konfirmasi resign pekerja terjadi.



Tabel 4.38 Use Case Scenario Konfirmasi Resign Pekerja

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_035   |
| <i>Objective</i>        | Use case ini bertujuan untuk mengkonfirmasi resign pekerja.   |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol konfirmasi resign pekerja.</li> <li>2. Sistem menampilkan daftar resign pekerja.</li> <li>3. Aktor menekan tombol konfirmasi pada salah satu daftar.</li> <li>4. Sistem menyimpan data resign pekerja dan mengubah status pekerja menjadi 'resign'.</li> </ol> |
| <i>Alternative Flow</i> | <ol style="list-style-type: none"> <li>3.1.1 Aktor menekan tombol tolak</li> <li>3.1.2 Sistem mengubah status pekerja menjadi "tersedia".</li> </ol>  |
| <i>Post-condition</i>   | Sistem menyimpan data resign pekerja ke dalam database.   |

Tabel 4.39 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi blacklist pekerja terjadi.

Tabel 4.39 Use Case Scenario Blacklist Pekerja

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_036   |
| <i>Objective</i>        | Use case ini bertujuan untuk memblacklist pekerja dari sistem.  |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol blacklist pekerja.</li> <li>2. Sistem menampilkan daftar pekerja.</li> <li>3. Aktor menekan tombol blacklist pada salah satu daftar.</li> <li>4. Sistem menyimpan data blacklist pekerja dan mengubah status pekerja menjadi 'blacklist'.</li> </ol> |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Sistem menyimpan data blacklist pekerja ke dalam database.  |



Tabel 4.40 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi blacklist majikan terjadi.

**Tabel 4.40 Use Case Scenario Blacklist Majikan**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_037   |
| <i>Objective</i>        | Use case ini bertujuan untuk memblacklist majikan dari sistem.  |
| <i>Actor</i>            | Admin   |
| <i>Pre-condition</i>    | Aktor telah berada pada halaman <i>admin dashboard</i> .  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol blacklist majikan.</li> <li>2. Sistem menampilkan daftar majikan.</li> <li>3. Aktor menekan tombol blacklist pada salah satu daftar.</li> <li>4. Sistem menyimpan data blacklist majikan dan mengubah status majikan menjadi 'blacklist'.</li> </ol> |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Sistem menyimpan data blacklist majikan ke dalam database.  |

Tabel 4.41 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi mengubah status pemesanan terjadi.

**Tabel 4.41 Use Case Scenario Mengubah Status Pemesanan**

|                         |   |
|-------------------------|---|
| Kode Kebutuhan          | PRT_F_038   |
| <i>Objective</i>        | Use case ini bertujuan untuk mengubah status pemesanan yang telah dilakukan.  |
| <i>Actor</i>            | Midtrans  |
| <i>Pre-condition</i>    | Aktor telah mengetahui <i>end point</i> API ubah status pemesanan.  |
| <i>Main Flow</i>        | <ol style="list-style-type: none"> <li>1. Aktor mengakses <i>end point</i> API ubah status pemesanan dengan mengirim data status pemesanan.</li> <li>2. Sistem mengubah status pemesanan sesuai dengan data yang diterima.</li> </ol> |
| <i>Alternative Flow</i> | -   |
| <i>Post-condition</i>   | Aktor berhasil melakukan perubahan status pemesanan pada sistem.  |



Tabel 4.42 akan menjelaskan bagaimana interaksi antara aktor dengan fungsi logout terjadi.

**Tabel 4.42 Use Case Scenario Logout**

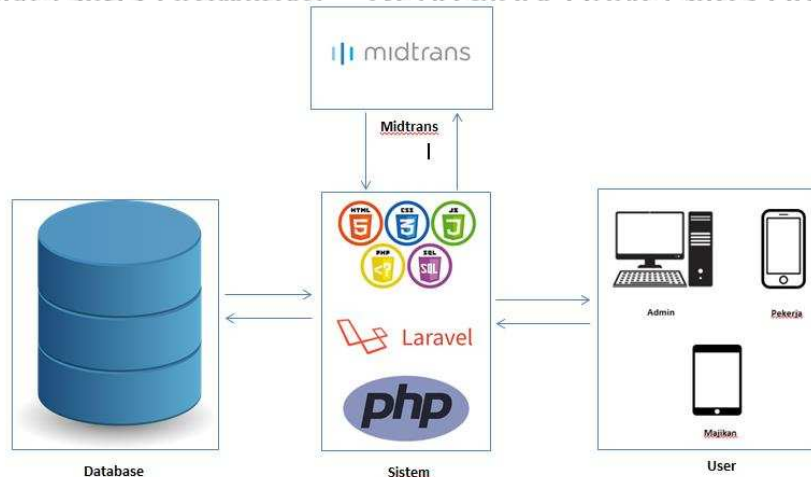
|                  |  |
|------------------|--|
| Kode Kebutuhan   | PRT_F_039  |
| Objective        | Use case ini bertujuan untuk mengeluarkan aktor dari sistem.   |
| Actor            | Admin, Pekerja, dan Majikan  |
| Pre-condition    | Aktor sudah Login  |
| Main Flow        | <ol style="list-style-type: none"> <li>1. Aktor menekan tombol Logout.</li> <li>2. Sistem mengeluarkan akun yang sudah Login.</li> <li>3. Sistem menampilkan halaman utama web.</li> </ol> |
| Alternative Flow |  |
| Post-condition   | Aktor berhasil keluar dari sistem.   |



## BAB 5 PERANCANGAN DAN IMPLEMENTASI SISTEM

### 5.1 Arsitektur Sistem

Tahapan ini merupakan tahapan perancangan arsitektur sistem. Rancangan ini akan memvisualisasikan sistem yang akan dibuat. Berikut adalah rancangan umum sistem yang akan divisualisasikan pada Gambar 5.1.



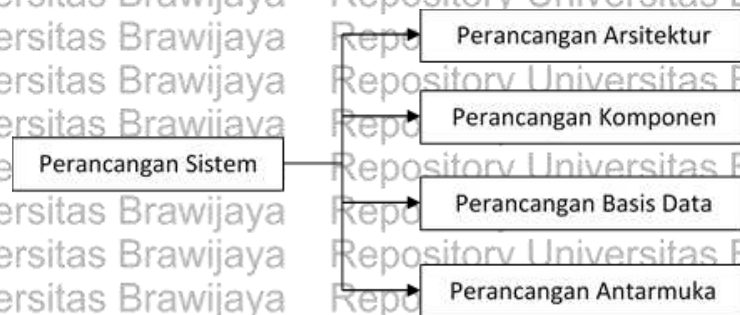
Gambar 5.1 Perancangan Arsitektur Dasar Sistem

Pada Gambar 5.1 dijelaskan secara umum dasar dari pembangunan sistem. Pembangunan sistem ini menggunakan beberapa komponen web dasar yaitu, HTML, CSS, JS, dan PHP. Sistem ini juga menggunakan *framework* Laravel dan juga Midtrans.

### 5.2 Perancangan Sistem

Tahapan ini merupakan tahapan yang akan dilakukan ketika analisis kebutuhan sistem telah dilakukan dengan baik. Tahapan ini sepenuhnya mengacu pada tahapan analisis. Tahap perancangan Sistem Manajemen dan Pemesanan Pekerja Rumah Tangga akan dipecah kedalam empat bagian yaitu, perancangan arsitektur, lalu dilanjutkan dengan tahapan perancangan komponen, setelah itu tahapan yang dilakukan adalah perancangan data, dan tahapan yang terakhir dilakukan adalah perancangan antarmuka. Gambar 5.2 akan menjabarkan struktur dari sub-bab perancangan sistem.





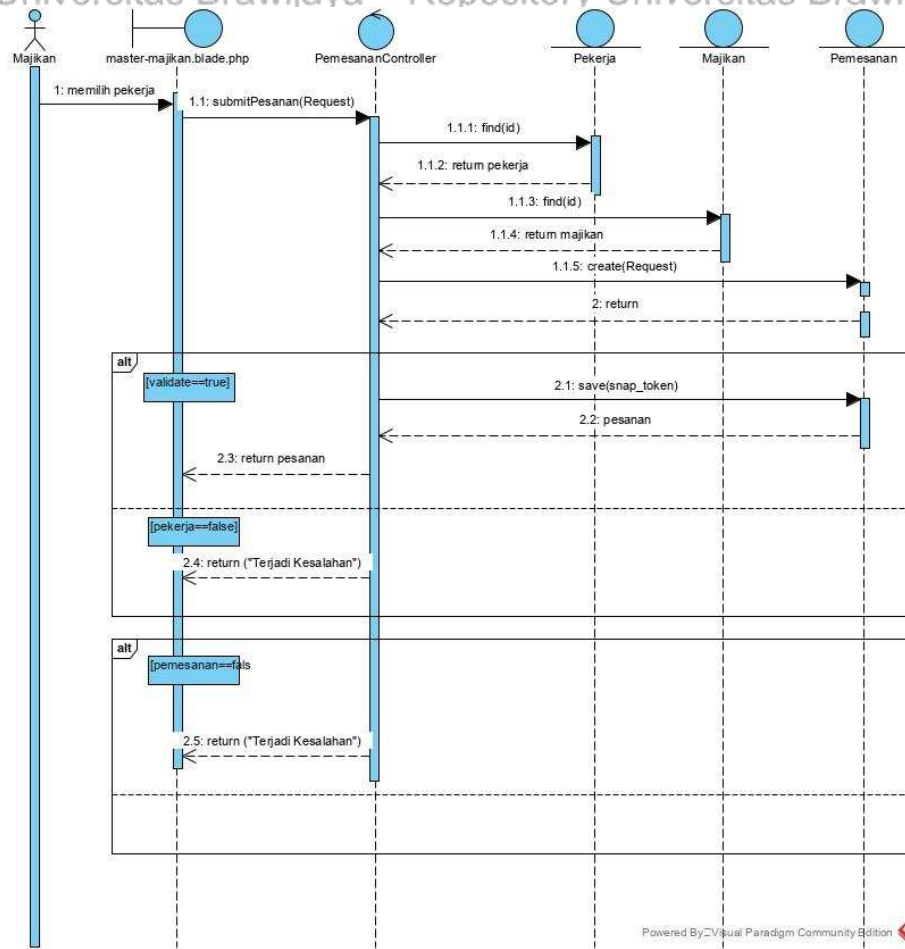
**Gambar 5.2 Struktur Sub-bab Perancangan Sistem**

### 5.2.1 Perancangan Arsitektur

Tahapan ini akan menjelaskan bagaimana interaksi antara satu kelas dengan kelas yang lainnya. Tahapan ini juga akan memaparkan pertukaran data yang terjadi pada suatu kelas. Tahapan ini akan menghasilkan diagram yang bernama *sequence diagram* dan *class diagram*. Ada tiga fungsionalitas yang akan digunakan dalam pembuatan *sequence diagram*, yaitu melakukan memesan jasa pekerja, hapus daftar pekerja, dan konfirmasi pendaftaran pekerja.

#### 5.2.1.1 Sequence Diagram Memesan Jasa Pekerja

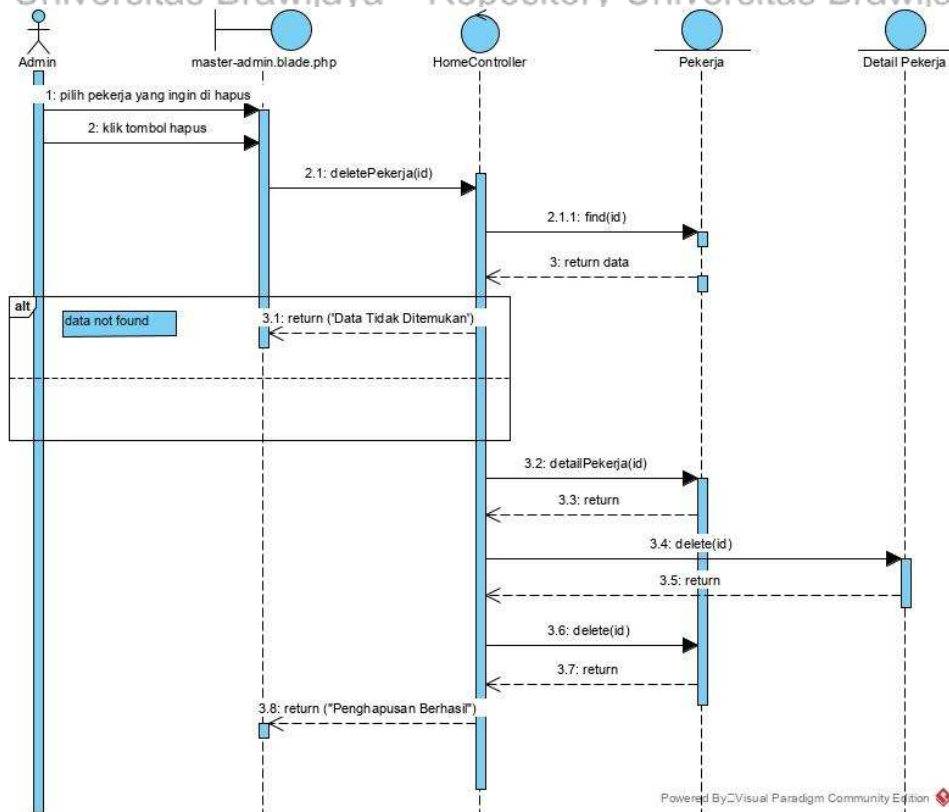
*Sequence diagram* memesan jasa pekerja akan ditampilkan pada Gambar 5.3. *Sequence diagram* ini mempunyai enam notasi, diantaranya adalah notasi aktor yaitu majikan, notasi *boundary* yaitu master-majikan.blade.php, notasi *controller* yaitu PemesananController, dan tiga buah notasi *entity* yaitu Pekerja, Majikan, dan juga Pemesanan. Dalam melakukan *use case* memesan jasa pekerja, aktor diharuskan memilih pekerja mana yang ingin dipesan. Setelah itu akan dijalankan method `submitPesanan()`, nantinya sistem akan memasukkan data pemesanan ke dalam database pemesanan, jika terjadi kegagalan saat menyimpan data maka akan muncul pesan error.



Gambar 5.3 Sequence Diagram Memesan Jasa Pekerja

5.2.1.2 Sequence Diagram Hapus Daftar Pekerja

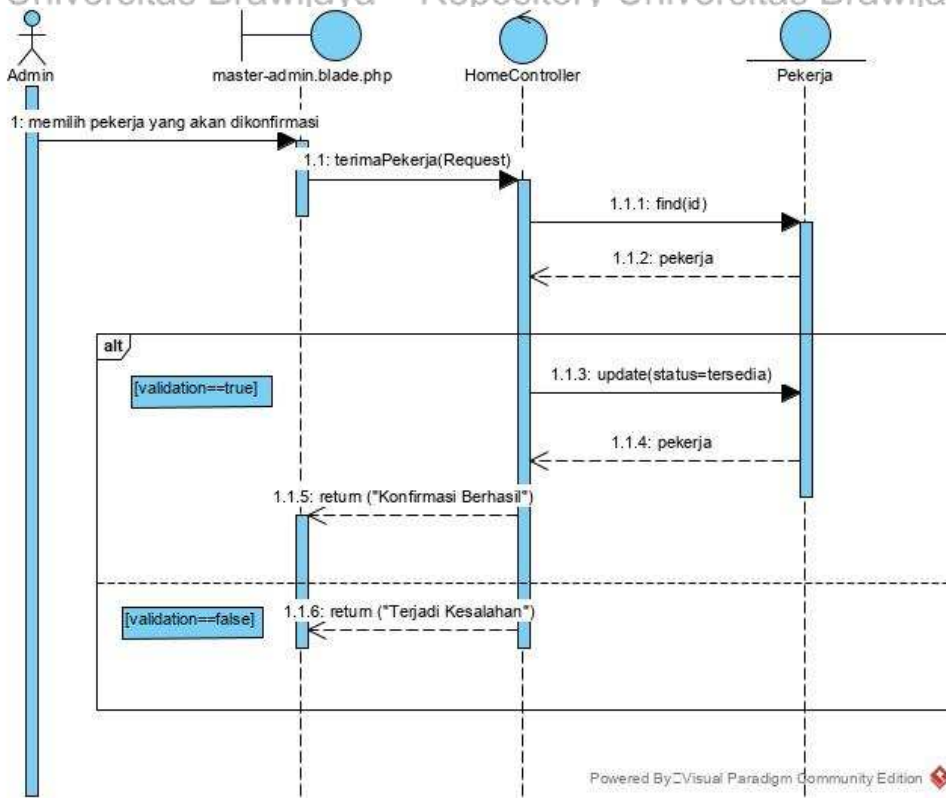
Sequence diagram melakukan penggajian pekerja akan ditampilkan pada Gambar 5.4. Sequence diagram ini mempunyai lima notasi, diantaranya adalah notasi aktor yaitu admin, notasi boundary yaitu master-admin.blade.php, notasi controller yaitu HomeController, dan dua buah notasi entity yaitu Pekerja dan Detail Pekerja. Dalam melakukan use case hapus daftar pekerja, aktor akan memilih pekerja yang nantinya datanya akan dihapus lalu menekan tombol hapus. Setelah itu sistem akan menjalankan method deletePekerja(). Lalu sistem melakukan pengecekan terhadap entitas Pekerja apakah data dengan id tersebut ada atau tidak, jika ada maka data dalam entitas Detail Pekerja dengan id yang diinputkan akan dihapus. lalu selanjutnya akan dilakukan penghapusan data dari entitas Pekerja sesuai dengan id yang diinputkan sebelumnya.



Gambar 5.4 Sequence Diagram Hapus Daftar Pekerja

### 5.2.1.3 Sequence Diagram Konfirmasi Pendaftaran Pekerja

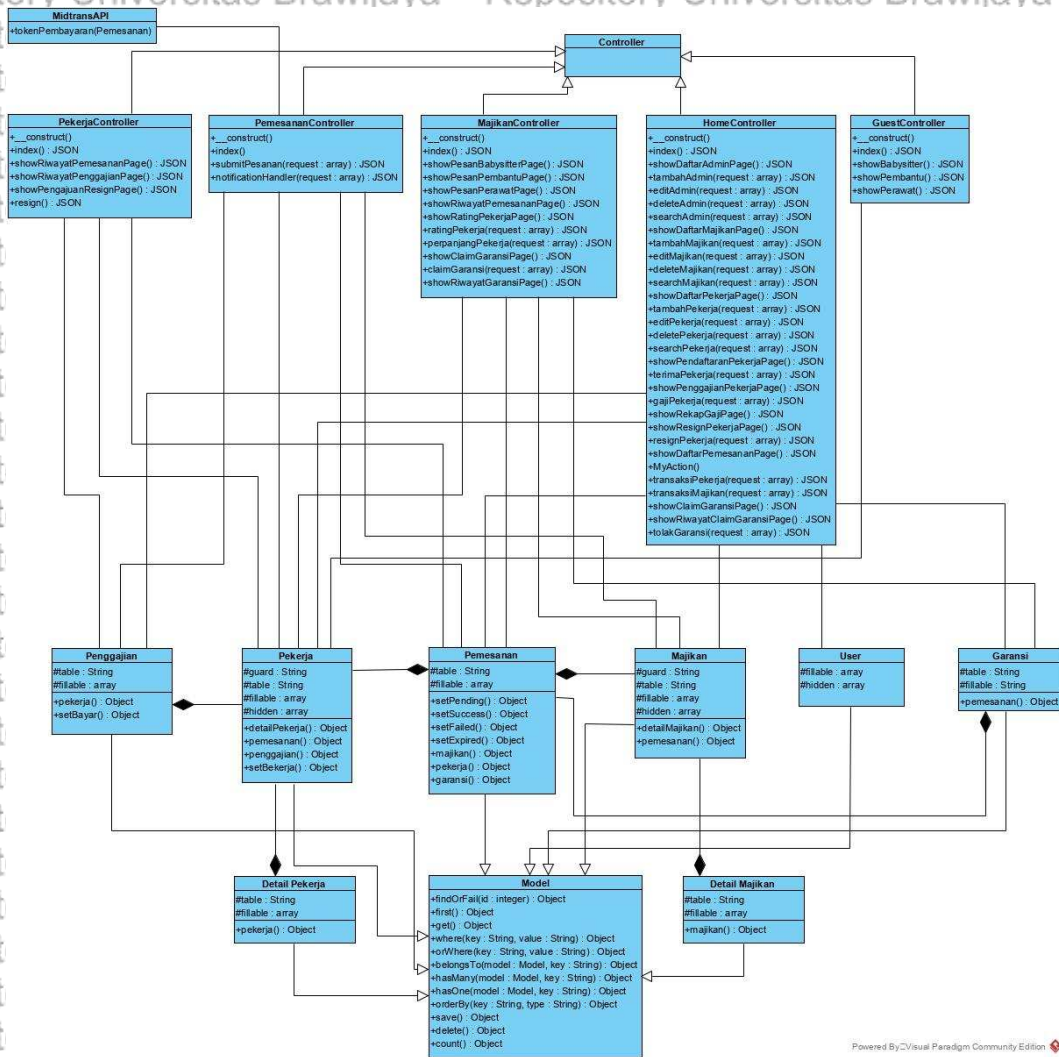
Sequence diagram melakukan penggajian pekerja akan ditampilkan pada Gambar 5.5. Sequence diagram ini mempunyai empat notasi, diantaranya adalah notasi aktor yaitu admin, notasi boundary yaitu pendaftaran-pekerja.blade.php, notasi controller yaitu HomeController, dan notasi entity yaitu Pekerja. Dalam melakukan use case Konfirmasi Pendaftaran pekerja, aktor harus memilih opsi konfirmasi pendaftaran pekerja dengan melakukan klik pada tombol konfirmasi pendaftaran pekerja sehingga nantinya method terimaPekerja() pada controller HomeController dapat dijalankan dan melakukan pengecekan apakah id pekerja yang dipilih terdaftar di database. Kemudian, data status pada entity pekerja dapat diubah dari pending menjadi tersedia.



Gambar 5.5 Sequence Diagram Konfirmasi Pendaftaran Pekerja

5.2.1.4 Class Diagram

Class diagram dari sistem yang akan dikembangkan diperlihatkan pada Gambar 5.6. Diagram ini berfungsi untuk menjabarkan kelas – kelas apa saja yang terdapat didalam sistem ini. Diagram ini juga berfungsi untuk menjelaskan relasi antar kelas dalam berinteraksi satu sama lainnya. Terdapat dua jenis kelas yang tersedia pada gambar class diagram. Kelas yang tersedia diantara lainnya adalah kelas controller dan kelas model.



Gambar 5.6 Class Diagram Sistem Manajemen dan Pemesanan Jasa Pekerja Rumah Tangga

### 5.2.2 Perancangan Komponen

Tahapan perancangan komponen memuat rancangan algoritme pada tiga fungsi yang terdapat pada *class diagram*. Penulis menggunakan *pseudocode* untuk menjelaskan alur kerja dari suatu algoritme atau komponen. Algoritme yang akan dirancangan yaitu algoritme `submitPesanan`, `deletePekerja`, dan `terimaPekerja`.

#### 5.2.2.1 Algoritme `submitPesanan`

Algoritme `submitPesanan` berguna untuk melakukan pemesanan dan menyimpan data pemesanan ke dalam *database*. Tabel 5.1 akan menjabarkan algoritme `submitPesanan`.

Nama kelas: `PemesananController`

Nama metode: `submitPesanan`



**Tabel 5.1 Pseudocode submitPesanan**

| Pseudocode metode submitPesanan |  |
|---------------------------------|--|
| 1                               | BEGIN  |
| 2                               |  |
| 3                               | FIND majikan with(id = id majikan of Request)        |
| 4                               | FIND pekerja with(id = id pekerja of Request)        |
| 5                               | IF(majikan and pekerja is true) {                    |
| 6                               | Now = Get value of date                              |
| 7                               | TRY {  |
| 8                               | CREATE pemesanan(majikan_id, pekerja_id,             |
| 9                               | jumlah_bayar, waktu_pesan, pertama) VALUE(majikan_id |
| 10                              | of Request, pekerja_id of Request, jumlah_bayar of   |
| 11                              | Request, waktu_pesan of Request, pertama of Request) |
| 12                              | RETURNING pemesanan                                  |
| 13                              | }  |
| 14                              | CATCH {  |
| 15                              | RETURN response to JSON ("Terjadi Kesalahan")        |
| 16                              | }  |
| 17                              | IF (pemesanan is null){                              |
| 18                              | RETURN response to JSON ("Terjadi Kesalahan")        |
| 19                              | }  |
| 20                              | Payload = array of (transaction_details,             |
| 21                              | customer_details, item_details)                      |
| 22                              | }  |
| 23                              | Snap_token = Get value of snap token                 |
| 24                              | UPDATE pemesanan(snap_token) VALUE(snap_token)       |
| 25                              | RETURNING pemesanan                                  |
| 26                              | }  |
| 27                              | RETURN response to JSON (order and store success     |
| 28                              | status 201)  |
| 29                              | }  |
| 30                              | ELSE {   |
| 31                              | RETURN response to JSON ("Terjadi Kesalahan")        |
| 32                              | }  |
| 33                              | }  |
| 34                              | END  |

### 5.2.2.2 Algoritme deletePekerja

Algoritme deletePekerja merupakan algoritme yang berfungsi untuk menghapus data pekerja dari database. Tabel 5.2 akan memperlihatkan algoritme dari metode deletePekerja.

Nama kelas: HomeController

Nama metode: deletePekerja

**Tabel 5.2 Pseudocode deletePekerja**

| Pseudocode metode deletePekerja |   |
|---------------------------------|---|
| 1                               | BEGIN   |
| 2                               |   |
| 3                               | FIND pekerja with(id = id of Request)             |
| 4                               | IF (pekerja.is TRUE){                             |
| 5                               | FIND detailPekerja with (id = id of Request)      |
| 6                               | DELETE detailPekerja WITH(id = id of Request)     |
| 7                               | RETURNING pekerja                                 |
| 8                               | DELETE pekerja WITH(id = id of Request) RETURNING |
| 9                               | pekerja   |
| 10                              |   |
| 11                              | RETURN response to JSON ("Penghapusan Berhasil")  |
| 12                              | }   |
| 13                              | ELSE {  |
| 14                              | RETURN response to JSON ("Data Tidak Ditemukan")  |
| 15                              | }   |
| 16                              | END   |

**5.2.2.3 Algoritme terimaPekerja**

Algoritme terimaPekerja berguna untuk mengkonfirmasi pendaftaran yang dilakukan oleh calon pekerja. Admin dapat menentukan apakah calon pekerja layak untuk mendaftar atau tidak. Algoritme konfirmasi nantinya akan mengubah data status sesuai dengan inputan admin, apakah pekerja bisa langsung bekerja atau tidak. Tabel 5.3 akan memperlihatkan algoritme terimaPekerja.

Nama kelas: HomeController

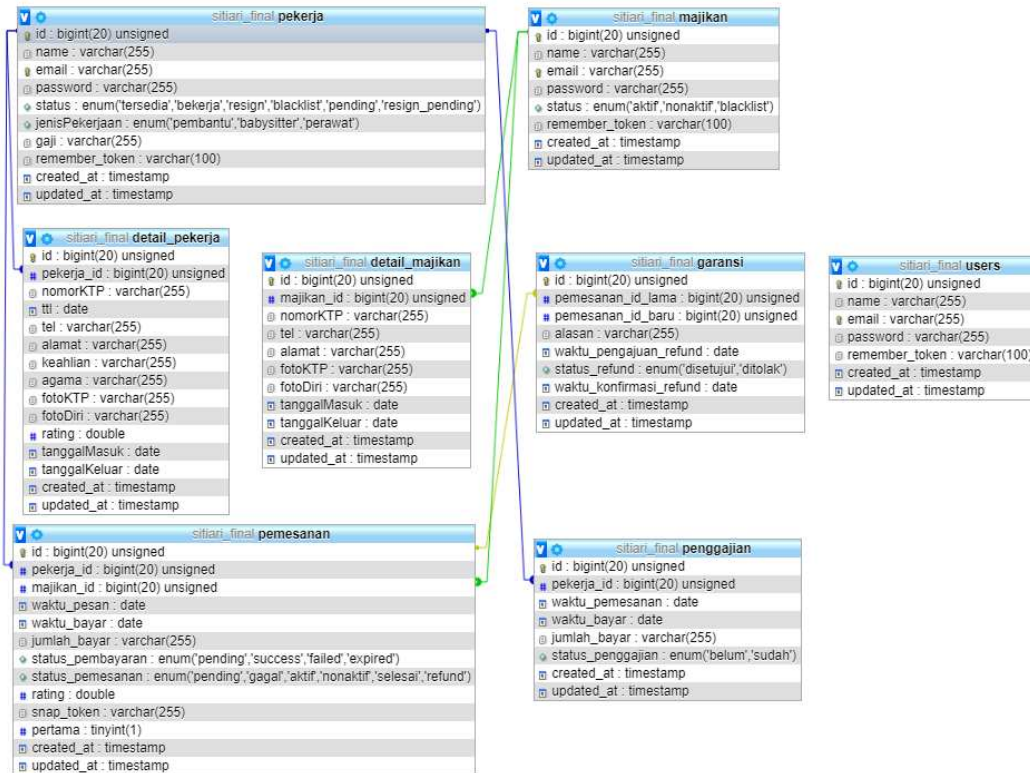
Nama metode: terimaPekerja

**Tabel 5.3 Pseudocode terimaPekerja**

| Pseudocode metode terimaPekerja |  |
|---------------------------------|--|
| 1                               | BEGIN  |
| 2                               |  |
| 3                               | FIND pekerja with(id = id of Request)              |
| 4                               | IF (pekerja.is TRUE) {                             |
| 5                               | UPDATE pekerja(status) VALUE("tersedia") RETURNING |
| 6                               | pekerja  |
| 7                               | RETURN response to JSON ("Konfirmasi Berhasil")    |
| 8                               | }  |
| 9                               | ELSE {   |
| 10                              | RETURN response to JSON ("Terjadi Kesalahan")      |
| 11                              | }  |
| 12                              | END  |
| 13                              |  |

### 5.2.3 Perancangan Basis Data

Tahapan ini merupakan tahapan perancangan basis data yang digambarkan dengan Gambar 5.7. Struktur data entitas yang telah didefinisikan pada *class diagram*, kemudian digambarkan dengan *physical data model* untuk merepresentasikan struktur fisik secara detail dari *database* yang akan dibangun. Pada diagram ini atribut entitas yang merupakan komposisi dari entitas lain digambarkan dengan *foreign key* dan atribut indeks digambarkan dengan *primary key*. Gambar 5.7 juga menjelaskan hubungan antar entitas.



Gambar 5.7 PDM Sistem Manajemen dan Pemesanan Jasa Pekerja Rumah Tangga

### 5.2.4 Perancangan Antarmuka

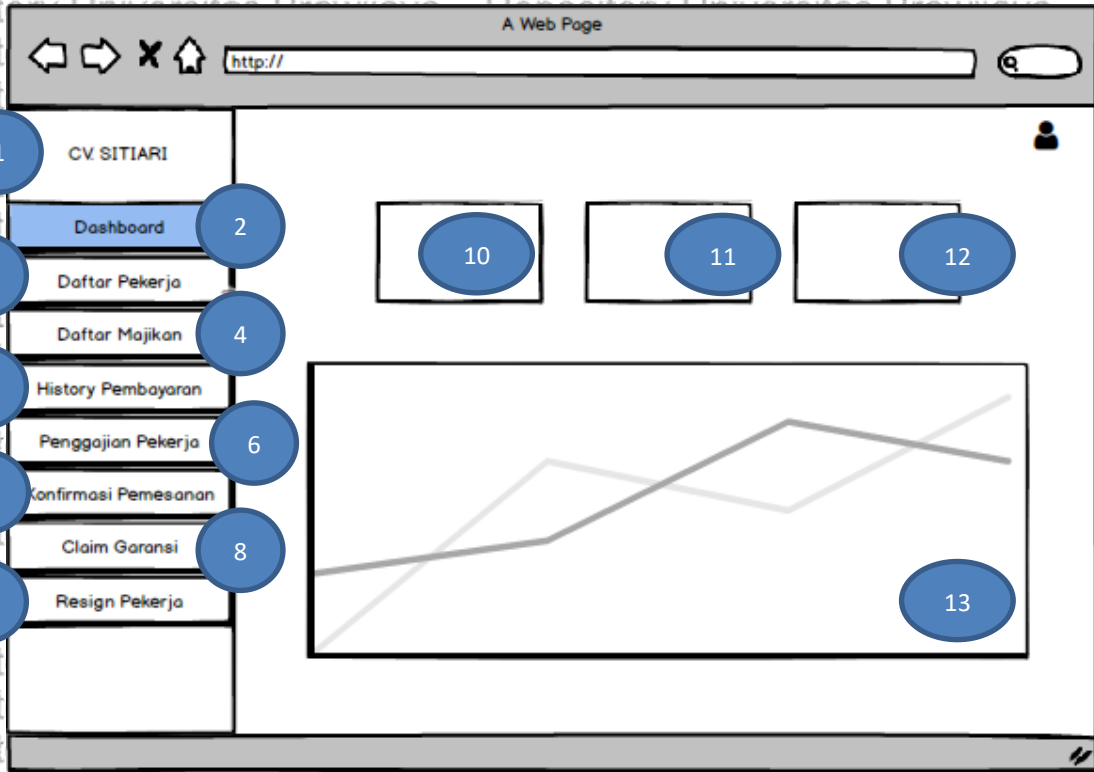
Tahapan ini bertujuan untuk menjelaskan dan memvisualisasikan rancangan dari tampilan antarmuka sistem. Rancangan tampilan yang dimaksud adalah rancangan komponen dari suatu halaman web. Nantinya, rancangan ini akan menjadi jembatan antara interaksi pengguna dan juga sistem. Selanjutnya, tahapan ini akan dijadikan acuan dalam tahapan berikutnya yaitu tahapan implementasi antarmuka. Pada penelitian ini perancangan antarmuka yang akan dijabarkan adalah perancangan antarmuka *Admin Dashboard*, perancangan antarmuka *Daftar Majikan*, perancangan antarmuka *Login*, dan perancangan antarmuka *Register Majikan*.





Penjelasan mengenai perancangan antarmuka akan dijelaskan pada sub bab 5.2.4.1, 5.2.4.2, 5.2.4.3, dan 5.2.4.4.

**5.2.4.1 Perancangan Antarmuka Admin Dashboard**



**Gambar 5.8 Perancangan Antarmuka Admin Dashboard**

Visualisasi perancangan antarmuka halaman Admin Dashboard akan dijabarkan pada Gambar 5.8. Sedangkan Tabel 5.4 akan menjabarkan penjelasan dari gambar tersebut.

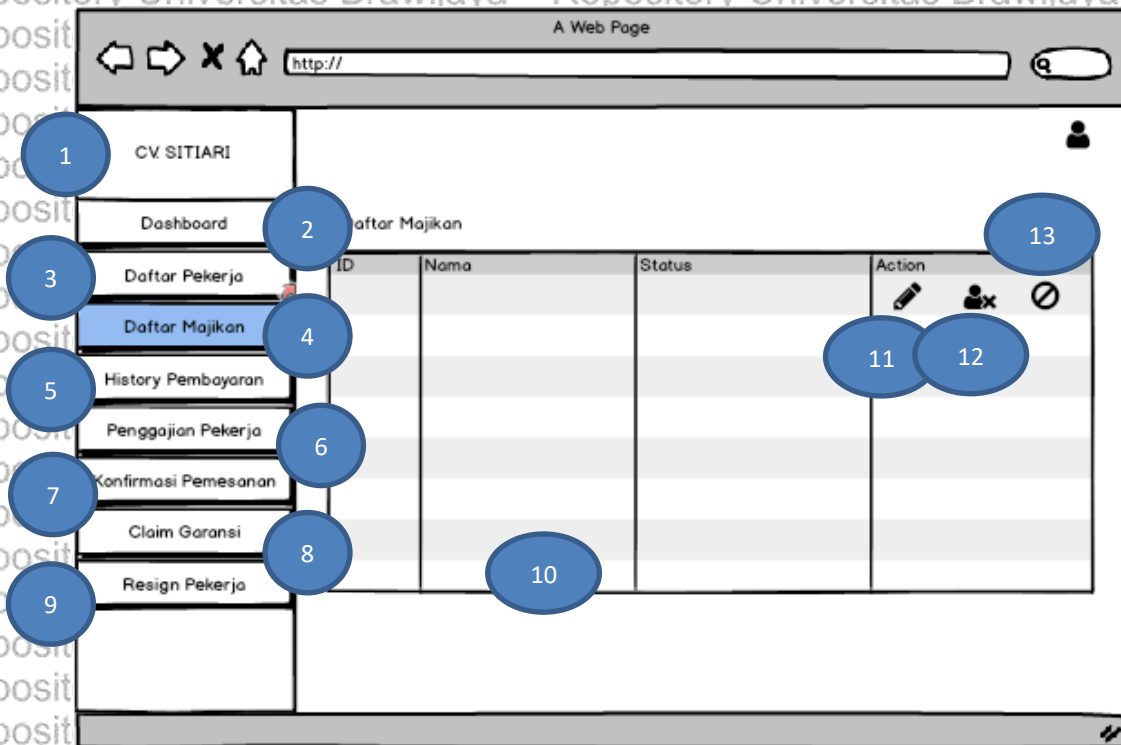
**Tabel 5.4 Keterangan Gambar Perancangan Antarmuka Homepage**

| No | Obyek              | Tip Obyek | Penjelasan  |
|----|--------------------|-----------|---|
| 1  | Logo               | Gambar    | Menampilkan Logo sistem dalam bentuk gambar       |
| 2  | Dashboard          | Tombol    | Tombol untuk mengarahkan admin ke dashboard admin |
| 3  | Daftar pekerja     | Tombol    | Tombol untuk menampilkan daftar pekerja           |
| 4  | Daftar Majikan     | Tombol    | Tombol untuk menampilkan daftar majikan           |
| 5  | History Pembayaran | Tombol    | Tombol untuk menampilkan history pembayaran       |



|    |                      |        |   |
|----|----------------------|--------|---|
| 6  | Penggajian Pekerja   | Tombol | Tombol untuk mengarahkan admin ke menu penggajian pekerja   |
| 7  | Konfirmasi Pemesanan | Tombol | Tombol untuk mengarahkan admin ke menu konfirmasi pemesanan |
| 8  | Klaim Garansi        | Tombol | Tombol untuk mengarahkan admin ke menu klaim garansi        |
| 9  | Resign Pekerja       | Tombol | Tombol untuk mengarahkan admin ke menu resign pekerja       |
| 10 | Daftar Pekerja       | Tombol | Tombol untuk menampilkan daftar pekerja                     |
| 11 | Daftar Majikan       | Tombol | Tombol untuk menampilkan daftar majikan                     |
| 12 | History Pembayaran   | Tombol | Tombol untuk menampilkan history pembayaran                 |
| 13 | Grafik pemesanan     | Grafik | Grafik yang menampilkan pembesaran yang terjadi.            |

**5.2.4.2 Perancangan Antarmuka Daftar Majikan**





### Gambar 5.9 Perancangan Antarmuka Daftar Majikan

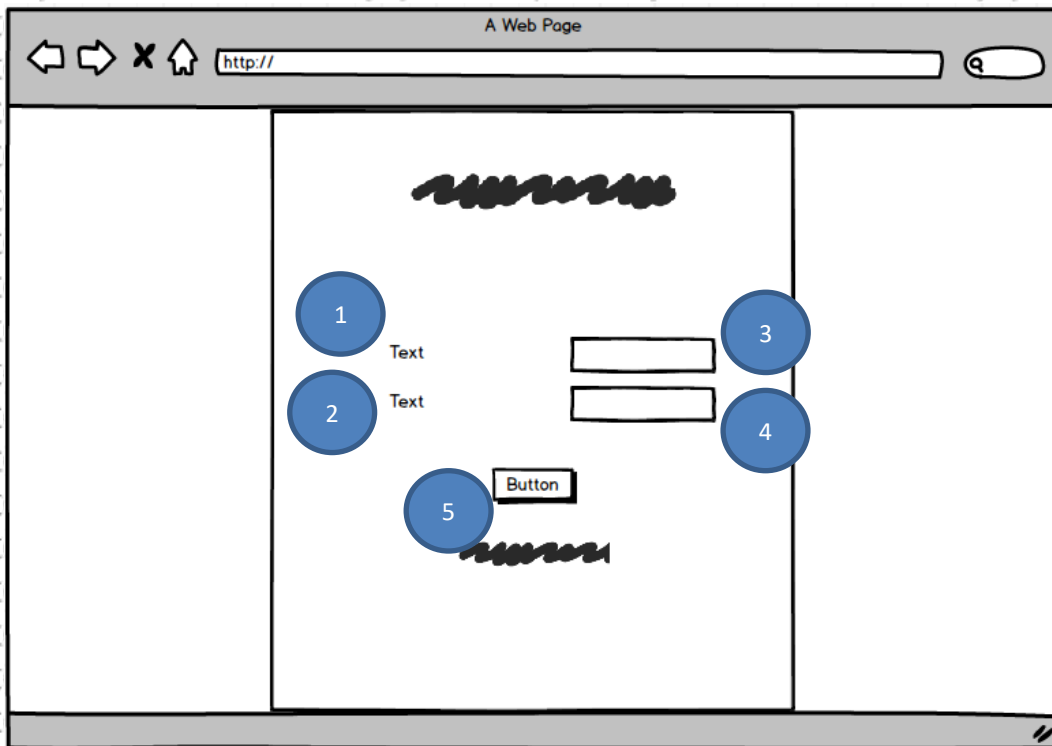
Visualisasi perancangan antarmuka halaman Daftar Majikan akan dijabarkan pada Gambar 5.9. Sedangkan Tabel 5.5 akan menjabarkan penjelasan dari gambar tersebut.

**Tabel 5.5 Keterangan Gambar Perancangan Antarmuka Daftar Majikan**

| No | Obyek                | Tip Obyek | Penjelasan   |
|----|----------------------|-----------|--|
| 1  | Logo                 | Gambar    | Memvisualisasikan logo sistem                                    |
| 2  | Dashboard            | Tombol    | Tombol yang berfungsi untuk mengarahkan admin ke dashboard admin |
| 3  | Daftar pekerja       | Tombol    | Tombol untuk menampilkan daftar pekerja                          |
| 4  | Daftar Majikan       | Tombol    | Tombol untuk menampilkan daftar majikan                          |
| 5  | History Pembayaran   | Tombol    | Tombol untuk menampilkan history pembayaran                      |
| 6  | Penggajian Pekerja   | Tombol    | Tombol untuk mengarahkan admin ke menu penggajian pekerja        |
| 7  | Konfirmasi Pemesanan | Tombol    | Tombol untuk mengarahkan admin ke menu konfirmasi pemesanan      |
| 8  | Klaim Garansi        | Tombol    | Tombol untuk mengarahkan admin ke menu klaim garansi             |
| 9  | Resign Pekerja       | Tombol    | Tombol untuk mengarahkan admin ke menu resign pekerja            |
| 10 | Tabel Daftar Majikan | Tabel     | Tabel untuk menampilkan daftar majikan                           |
| 11 | Edit Majikan         | Tombol    | Tombol yang berfungsi untuk menjalankan fungsi edit data majikan |
| 12 | Hapus Majikan        | Tombol    | Tombol yang berfungsi untuk menjalankan fungsi hapus majikan.    |
| 13 | Blacklist Majikan    | Tombol    | Tombol untuk menjalankan fungsi blacklist majikan.               |



5.2.4.3 Perancangan Antarmuka Login



Gambar 5.10 Perancangan Antarmuka Login

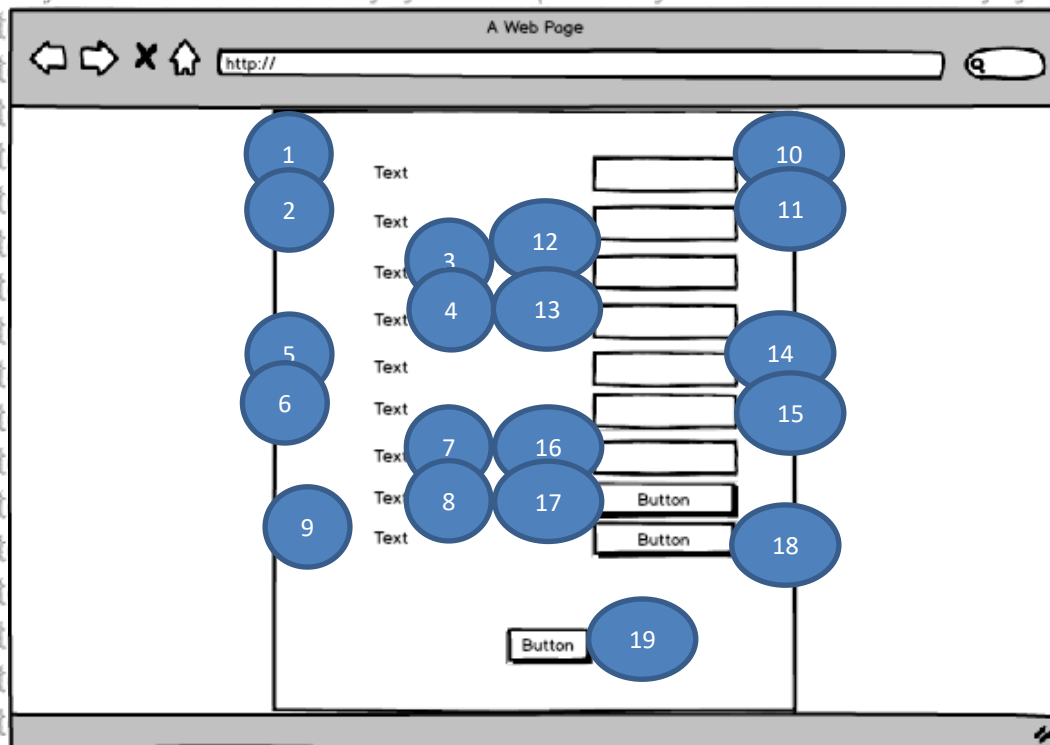
Visualisasi perancangan antarmuka halaman Login akan dijabarkan pada Gambar 5.10. Sedangkan Tabel 5.6 akan menjabarkan penjelasan dari gambar tersebut.

Tabel 5.6 Keterangan Gambar Perancangan Antarmuka Login

| No | Obyek              | Tip Obyek | Penjelasan                                   |
|----|--------------------|-----------|--|
| 1  | Text Email         | Label     | Memvisualisasikan text "email".              |
| 2  | Text Password      | Label     | Menampilkan text "Password".                 |
| 3  | Email Textfield    | Textfield | Mengambil inputan dari user berupa email.    |
| 4  | Password Textfield | Textfield | Mengambil inputan dari user berupa password. |
| 5  | Login              | Tombol    | Tombol untuk menjalankan fungsi Login.       |



### 5.2.4.4 Perancangan Antarmuka Register Majikan



**Gambar 5.11 Perancangan Antarmuka Register Majikan**

Visualisasi perancangan antarmuka halaman Register Majikan akan dijabarkan pada Gambar 5.11. Sedangkan Tabel 5.7 akan menjabarkan penjelasan dari gambar tersebut.

**Tabel 5.7 Keterangan Gambar Perancangan Antarmuka Register Majikan**

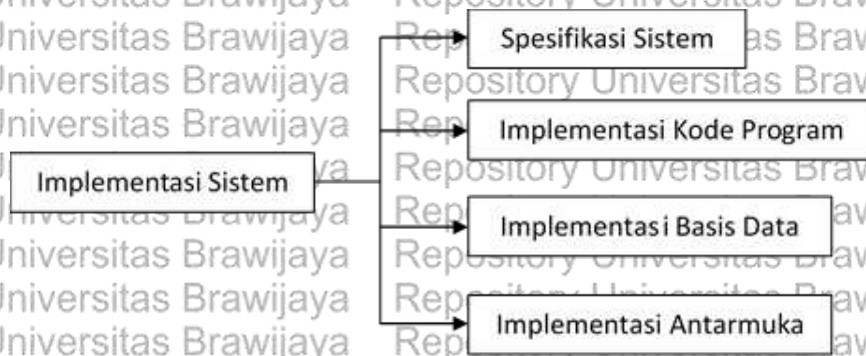
| No | Obyek                    | Tip Obyek | Penjelasan                           |
|----|--------------------------|-----------|--------------------------------------|
| 1  | Text Nama                | Label     | Memunculkan text "Nama".             |
| 2  | Text Email               | Label     | Menampilkan text "email".            |
| 3  | Text Password            | Label     | Menampilkan text "Password".         |
| 4  | Text Konfirmasi Password | Label     | Menampilkan text "Confirm Password". |
| 5  | Text Nomor KTP           | Label     | Menampilkan text "Nomor KTP".        |
| 6  | Text Nomor Telepon       | Label     | Menampilkan text "Nomor Telepon".    |
| 7  | Text Alamat              | Label     | Menampilkan text "Alamat".           |
| 8  | Text Foto KTP            | Label     | Menampilkan text "Foto KTP".         |



|    |                     |           |   |
|----|---------------------|-----------|---|
| 9  | Text Foto Diri      | Label     | Menampilkan text "Foto Diri"                            |
| 10 | Nama                | Textfield | Mengambil inputan dari user berupa nama.                |
| 11 | Email               | Textfield | Mengambil inputan dari berupa email.                    |
| 12 | Password            | Textfield | Mengambil inputan dari berupa password.                 |
| 13 | Konfirmasi Password | Textfield | Mengambil inputan dari berupa konfirmasi password.      |
| 14 | Nomor KTP           | Textfield | Mengambil inputan dari berupa nomor KTP.                |
| 15 | Nomor Telepon       | Textfield | Mengambil inputan dari berupa nomor telepon.            |
| 16 | Alamat              | Textfield | Mengambil inputan dari berupa alamat.                   |
| 17 | Foto KTP            | Tombol    | Tombol yang berfungsi untuk mengupload foto KTP.        |
| 18 | Foto Diri           | Tombol    | Tombol yang berfungsi untuk mengupload foto diri.       |
| 19 | Register            | Tombol    | Tombol yang berfungsi untuk menjalankan fungsi register |

### 5.3 Implementasi Sistem

Tahapan ini akan dilaksanakan ketika seluruh tahapan perancangan telah selesai dilakukan. Tahapan ini adalah sebuah penggabungan antara tahapan analisis kebutuhan dengan tahapan perancangan sistem. Seluruh kebutuhan yang sebelumnya telah didefinisikan harus diimplementasikan kedalam sistem. Implementasi antarmuka pun juga harus sesuai dengan perancangan pada tahap sebelumnya. Acuan dalam implementasi sistem adalah *pseudocode* algoritme dan struktur kelas pada *class diagram*. Tahapan ini akan menjelaskan dan juga mendefinisikan spesifikasi dari sistem, implementasi data, implementasi kode program, dan implementasi antarmuka. Gambar 5.12 akan menjelaskan struktur dari sub-bab implementasi sistem.



**Gambar 5.12 Struktur Sub-bab Implementasi Sistem**

Tabel 5.8 akan menjelaskan informasi akun yang dipakai di dalam sistem yang akan dikembangkan.

**Tabel 5.8 Data Akun Sistem Pemesanan Dan Manajemen Pekerja Rumah Tangga**

| Email          | Password   | Keterangan  |
|----------------|------------|---|
| admin@me.com   | qwertyuiop | Merupakan informasi data autentikasi <i>admin</i> . |
| pekerja@me.com | qwertyuiop | Merupakan informasi data autentikasi pekerja.       |
| majikan@me.com | qwertyuiop | Merupakan informasi data autentikasi majikan.       |

### 5.3.1 Spesifikasi Sistem

Pada sub bab ini akan menjabarkan tentang spesifikasi dari sistem yang akan digunakan. Baik itu perangkat keras, perangkat lunak, maupun sistem operasi.

#### 5.3.1.1 Spesifikasi Perangkat Keras

Tabel 5.9 akan menjelaskan tentang perangkat keras yang akan digunakan dalam pengembangan sistem beserta spesifikasinya.

**Tabel 5.9 Spesifikasi Perangkat Keras**

| Nama Komponen                    | Spesifikasi  |
|----------------------------------|--------------|
| Model Dari Sistem Yang Digunakan | ASUS A455LN  |
| Jenis Dari Prosesor              | I5-5200U CPU |
| Jumlah Memori                    | 12288MB RAM  |



|                       |         |
|-----------------------|---------|
| Kapasitas Penyimpanan | 1TB HDD |
|-----------------------|---------|

### 5.3.1.2 Spesifikasi Perangkat Lunak

Tabel 5.10 akan menjelaskan tentang perangkat lunak yang akan digunakan dalam pengembangan sistem beserta spesifikasinya.

**Tabel 5.10 Spesifikasi Perangkat Lunak**

| Nama Komponen              | Spesifikasi               |
|----------------------------|---------------------------|
| Editor Untuk Dokumentasi   | Microsoft Word 2010       |
| Editor Untuk Perancangan   | Visual Paradigm 15.2      |
| Editor Untuk Pemrograman   | Visual Studio Code 1.31.1 |
| Bahasa Pemrograman Utama   | PHP 7.3.2, Javascript     |
| Framework Yang Digunakan   | Laravel 6.0               |
| Database Management System | Mysql 4.8.5               |

### 5.3.1.3 Spesifikasi Sistem Operasi

Tabel 5.11 akan menjabarkan spesifikasi dari sistem operasi yang dipergunakan dalam pengembangan sistem.

**Tabel 5.11 Spesifikasi Sistem Operasi**

| Nama Komponen  | Spesifikasi                    |
|----------------|--------------------------------|
| Sistem Operasi | Windows 10 (Build 1909) 64-bit |

### 5.3.2 Implementasi Kode Program

Implementasi dari kode program akan berkaca pada perancangan komponen yang telah dibuat pada bab sebelumnya. *Pseudocode* yang sebelumnya telah dibuat akan diubah kedalam bahasa pemrograman yang sesuai dengan spesifikasi perangkat Lunak.

#### 5.3.2.1 Implementasi Kode Program *Method* submitPesanan

Implementasi kode program *method* submitPesanan berguna untuk melakukan pemesanan dan melakukan penyimpanan data pemesanan ke *database*. Kode program submitPesanan akan dijelaskan pada Tabel 5.12.

Nama kelas: PemesananController

Nama metode: submitPesanan

**Tabel 5.12 Implementasi Kode Program *Method* submitPesanan**





```
Kode metode submitPesanan
```

```

1 $majikan = Majikan::find($request->id_majikan);
2 $pekerja = Pekerja::find($request-
3 >id_pekerja);
4 if($majikan && $pekerja){
5     $now = date('Y-m-d');
6
7     try{
8         $pesanan = Pemesanan::create([
9             'majikan_id' => $request-
10 >id_majikan,
11             'pekerja_id' => $request-
12 >id_pekerja,
13             'jumlah_bayar' => $request-
14 >jumlahBayar,
15             'waktu_pesan' => $now,
16             'pertama' => $request->kategori,
17         ]);
18     } catch(QueryException $e){
19         return "Terjadi Kesalahan";
20     }
21     $payload = [
22         'transaction details' => [
23             'order_id' => $pesanan->id,
24             'gross_amount' => $pesanan-
25 >jumlah_bayar,
26         ],
27         'customer details' => [
28             'first_name' => $majikan->name,
29             'email' => $majikan-
30 >email,
31             'phone' => $majikan-
32 >detailMajikan->tel,
33             'address' => $majikan-
34 >detailMajikan->alamat,
35         ],
36         'item_details' => [
37             'id' => $pesanan->id,
38             'price' => $pesanan-
39 >jumlah_bayar,
40             'quantity' => 1,
41             'name' => 'Jasa PRT 1
42 Bulan',
43         ],
44     ];
45
46     $snapToken =
47     Veritrans_Snap::getSnapToken($payload);
48     $pesanan->snap_token = $snapToken;
49     $pesanan->save();
50     return $pesanan;
51 }
52

```



```

53         else
54         {
55             return "Terjadi Kesalahan";
56         }

```

### 5.3.2.2 Implementasi Kode Program *Method deletePekerja*

Implementasi kode program method *deletePekerja* berguna untuk menghapus data pekerja dari dalam *database*. Kode program *deletePekerja* akan dijelaskan pada Tabel 5.13.

Nama klas: *HomeController*

Nama metod: *deletePekerja*

**Tabel 5.13 Implementasi Kode Program *Method deletePekerja***

| Source code method <i>deletePekerja</i> |  |
|---|--|
| 1                                       | <code>\$pekerja = Pekerja::find(\$request-&gt;id);</code>      |
| 2                                       | <code>if(\$pekerja){</code>                                    |
| 3                                       | <code>    \$pekerja-&gt;detailPekerja()-&gt;delete();</code>   |
| 4                                       | <code>    \$pekerja-&gt;delete();</code>                       |
| 5                                       | <code>    return back()-&gt;with('sukses', 'Penghapusan</code> |
| 6                                       | <code>    Berhasil');</code>                                   |
| 7                                       | <code>}</code>   |
| 8                                       | <code>else{</code>   |
| 9                                       | <code>    return back()-&gt;with('alert', 'Data Tidak</code>   |
| 10                                      | <code>    Ditemukan');</code>                                  |
| 11                                      | <code>}</code>   |

### 5.3.2.3 Implementasi Kode Program *Method terimaPekerja*

Kode program method *terimaPekerja* yang telah diimplementasi berguna untuk mengkonfirmasi data pekerja yang mendaftar ke dalam *database*. Kode program *terimaPekerja* dapat dilihat pada Tabel 5.14.

Nama klas: *HomeController*

Nama metode: *terimaPekerja*

**Tabel 5.14 Implementasi Kode Program *Method terimaPekerja***

| Source code method <i>konfirmasiPendaftar</i> |   |
|---|---|
| 1   | <code>\$pekerja = Pekerja::find(\$request-&gt;id);</code>     |
| 2   | <code>if(\$pekerja){</code>                                   |
| 3   | <code>    \$pekerja-&gt;status = 'tersedia';</code>           |
| 4   | <code>    \$pekerja-&gt;save();</code>                        |
| 5   | <code>    return back()-&gt;with('sukses', 'Konfirmasi</code> |
| 6   | <code>    Berhasil');</code>                                  |
| 7   | <code>} else{</code>  |
| 8   | <code>    return back()-&gt;with('alert', 'Terjadi</code>     |
| 9   | <code>    Kesalahan');</code>                                 |
| 10  | <code>}</code>  |



### 5.3.3 Implementasi Basis Data

Tahapan ini dilakukan setelah tahapan perancangan basis data selesai dilakukan. Basis data akan diimplementasi dengan menyesuaikan pada perancangan yang telah dibuat sebelumnya. Pada sistem ini, implementasi data menghasilkan tujuh tabel. Pada sub bab ini akan dijabarkan Query *Data Definition Language* (DDL) dari implementasi ke tujuh tabel tersebut. Tabel 5.15 sampai 5.21 adalah penjabaran dari Query DDL basis data.

#### 5.3.3.1 Implementasi Data Tabel Users

Tabel 5.15 akan menjabarkan implementasi basis data tabel users yang akan divisualisasikan dalam bentuk *query Data Definition Language* (DDL).

**Tabel 5.15 Implementasi Data Tabel Users**

| Implementasi Data Tabel Users |   |
|-------------------------------|---|
| 1                             | CREATE TABLE `users` (                              |
| 2                             | `id` bigint(20) UNSIGNED NOT NULL,                  |
| 3                             | `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT  |
| 4                             | NULL,   |
| 5                             | `email` varchar(255) COLLATE utf8mb4_unicode_ci NOT |
| 6                             | NULL,   |
| 7                             | `password` varchar(255) COLLATE utf8mb4_unicode_ci  |
| 8                             | NOT NULL,   |
| 9                             | `remember_token` varchar(100) COLLATE               |
| 10                            | utf8mb4_unicode_ci DEFAULT NULL,                    |
| 11                            | `created_at` timestamp NULL DEFAULT NULL,           |
| 12                            | `updated_at` timestamp NULL DEFAULT NULL,           |
| 13                            | )   |

#### 5.3.3.2 Implementasi Data Tabel Pekerja

Tabel 5.16 akan menjabarkan implementasi basis data tabel pekerja yang akan divisualisasikan dalam bentuk *query Data Definition Language* (DDL).

**Tabel 5.16 Implementasi Data Tabel Pekerja**

| Implementasi Data Tabel Pekerja |  |
|---------------------------------|--|
| 1                               | CREATE TABLE `pekerja` (                                 |
| 2                               | `id` bigint(20) UNSIGNED NOT NULL,                       |
| 3                               | `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT       |
| 4                               | NULL,  |
| 5                               | `email` varchar(255) COLLATE utf8mb4_unicode_ci NOT      |
| 6                               | NULL,  |
| 7                               | `password` varchar(255) COLLATE utf8mb4_unicode_ci NOT   |
| 8                               | NULL,  |
| 9                               | `status`   |
| 10                              | enum('tersedia','bekerja','resign','blacklist','pending' |
| 11                              | ,'resign pending') COLLATE utf8mb4_unicode_ci NOT NULL   |
| 12                              | DEFAULT 'pending',                                       |



```

13 `jenisPekerjaan`
14 enum('pembantu','babysitter','perawat') COLLATE
15 utf8mb4_unicode_ci NOT NULL,
16 `gaji` varchar(255) COLLATE utf8mb4_unicode_ci NOT
17 NULL DEFAULT '1500000',
18 `remember_token` varchar(100) COLLATE
19 utf8mb4_unicode_ci DEFAULT NULL,
20 `created_at` timestamp NULL DEFAULT NULL,
21 `updated_at` timestamp NULL DEFAULT NULL
22 )

```

### 5.3.3.3 Implementasi Data Tabel Detail Pekerja

Tabel 5.17 akan menjabarkan implementasi basis data tabel detail pekerja yang akan divisualisasikan dalam bentuk *query Data Definition Language (DDL)*.

**Tabel 5.17 Implementasi Data Tabel Detail Pekerja**

```

Implementasi Data Tabel Detail Pekerja
1 CREATE TABLE `detail_pekerja` (
2 `id` bigint(20) UNSIGNED NOT NULL,
3 `pekerja_id` bigint(20) UNSIGNED NOT NULL,
4 `nomorKTP` varchar(255) COLLATE utf8mb4_unicode_ci NOT
5 NULL,
6 `ttl` date NOT NULL,
7 `tels` varchar(255) COLLATE utf8mb4_unicode_ci NOT
8 NULL,
9 `alamat` varchar(255) COLLATE utf8mb4_unicode_ci NOT
10 NULL,
11 `keahlian` varchar(255) COLLATE utf8mb4_unicode_ci NOT
12 NULL,
13 `agama` varchar(255) COLLATE utf8mb4_unicode_ci NOT
14 NULL,
15 `fotoKTP` varchar(255) COLLATE utf8mb4_unicode_ci NOT
16 NULL,
17 `fotoDiri` varchar(255) COLLATE utf8mb4_unicode_ci NOT
18 NULL,
19 `rating` double NOT NULL DEFAULT 5,
20 `tanggalMasuk` date NOT NULL,
21 `tanggalKeluar` date DEFAULT NULL,
22 `created_at` timestamp NULL DEFAULT NULL,
23 `updated_at` timestamp NULL DEFAULT NULL
24 )

```

### 5.3.3.4 Implementasi Data Tabel Majikan

Tabel 5.18 akan menjabarkan implementasi basis data tabel majikan yang akan divisualisasikan dalam bentuk *query Data Definition Language (DDL)*.

**Tabel 5.18 Implementasi Data Tabel Majikan**



#### Implementasi Data Tabel Majikan

```

1 CREATE TABLE `majikan` (
2   `id` bigint(20) UNSIGNED NOT NULL,
3   `name` varchar(255) COLLATE utf8mb4_unicode_ci NOT
4   NULL,
5   `email` varchar(255) COLLATE utf8mb4_unicode_ci NOT
6   NULL,
7   `password` varchar(255) COLLATE utf8mb4_unicode_ci NOT
8   NULL,
9   `status` enum('aktif','nonaktif','blacklist') COLLATE
10  utf8mb4_unicode_ci NOT NULL DEFAULT 'aktif',
11  `remember_token` varchar(100) COLLATE
12  utf8mb4_unicode_ci DEFAULT NULL,
13  `created_at` timestamp NULL DEFAULT NULL,
14  `updated_at` timestamp NULL DEFAULT NULL
15 )

```

#### 5.3.3.5 Implementasi Data Tabel Detail Majikan

Tabel 5.19 akan menjabarkan implementasi basis data tabel detail majikan yang akan divisualisasikan dalam bentuk *query Data Definition Language (DDL)*.

**Tabel 5.19 Implementasi Data Tabel Detail Majikan**

#### Implementasi Data Tabel Detail Majikan

```

1 CREATE TABLE `detail_majikan` (
2   `id` bigint(20) UNSIGNED NOT NULL,
3   `majikan_id` bigint(20) UNSIGNED NOT NULL,
4   `nomorKTP` varchar(255) COLLATE utf8mb4_unicode_ci NOT
5   NULL,
6   `tel` varchar(255) COLLATE utf8mb4_unicode_ci NOT
7   NULL,
8   `alamat` varchar(255) COLLATE utf8mb4_unicode_ci NOT
9   NULL,
10  `fotoKTP` varchar(255) COLLATE utf8mb4_unicode_ci NOT
11  NULL,
12  `fotoDiri` varchar(255) COLLATE utf8mb4_unicode_ci NOT
13  NULL,
14  `tanggalMasuk` date NOT NULL,
15  `tanggalKeluar` date DEFAULT NULL,
16  `created_at` timestamp NULL DEFAULT NULL,
17  `updated_at` timestamp NULL DEFAULT NULL
18 )

```

#### 5.3.3.6 Implementasi Data Tabel Pemesanan

Tabel 5.20 akan menjabarkan implementasi basis data tabel pemesanan yang akan divisualisasikan dalam bentuk *query Data Definition Language (DDL)*.

**Tabel 5.20 Implementasi Data Tabel Pemesanan**



#### Implementasi Data Tabel Pemesanan

```

1 CREATE TABLE `pemesanan` (
2   `id` bigint(20) UNSIGNED NOT NULL,
3   `pekerja_id` bigint(20) UNSIGNED NOT NULL,
4   `majikan_id` bigint(20) UNSIGNED NOT NULL,
5   `waktu_pesan` date NOT NULL,
6   `waktu_bayar` date DEFAULT NULL,
7   `jumlah_bayar` varchar(255) COLLATE utf8mb4_unicode_ci
8   NOT NULL,
9   `status_pembayaran`
10  enum('pending','success','failed','expired') COLLATE
11  utf8mb4_unicode_ci NOT NULL DEFAULT 'pending',
12  `status_pemesanan`
13  enum('pending','gagal','aktif','nonaktif','selesai','ref
14  und') COLLATE utf8mb4_unicode_ci NOT NULL DEFAULT
15  'pending',
16  `rating` double DEFAULT NULL,
17  `snap_token` varchar(255) COLLATE utf8mb4_unicode_ci
18  DEFAULT NULL,
19  `pertama` tinyint(1) NOT NULL,
20  `created_at` timestamp NULL DEFAULT NULL,
21  `updated_at` timestamp NULL DEFAULT NULL
22  )

```

#### 5.3.3.7 Implementasi Data Tabel Penggajian

Tabel 5.21 akan menjabarkan implementasi basis data tabel penggajian yang akan divisualisasikan dalam bentuk *query Data Definition Language (DDL)*.

**Tabel 5.21 Implementasi Data Tabel Penggajian**

#### Implementasi Data Tabel Penggajian

```

1 CREATE TABLE `penggajian` (
2   `id` bigint(20) UNSIGNED NOT NULL,
3   `pekerja_id` bigint(20) UNSIGNED NOT NULL,
4   `waktu_pemesanan` date NOT NULL,
5   `waktu_bayar` date DEFAULT NULL,
6   `jumlah_bayar` varchar(255) COLLATE utf8mb4_unicode_ci
7   NOT NULL,
8   `status_penggajian` enum('belum','sudah') COLLATE
9   utf8mb4_unicode_ci NOT NULL DEFAULT 'belum',
10  `created_at` timestamp NULL DEFAULT NULL,
11  `updated_at` timestamp NULL DEFAULT NULL
12  )

```

#### 5.3.3.8 Implementasi Data Tabel Garansi

Tabel 5.22 akan menjabarkan implementasi basis data tabel garansi yang akan divisualisasikan dalam bentuk *query Data Definition Language (DDL)*.

**Tabel 5.22 Implementasi Data Tabel Garansi**

#### Implementasi Data Tabel Garansi



```

1 CREATE TABLE `garansi` (
2   `id` bigint(20) UNSIGNED NOT NULL,
3   `pemesanan_id_lama` bigint(20) UNSIGNED NOT NULL,
4   `pemesanan_id_baru` bigint(20) UNSIGNED DEFAULT NULL,
5   `alasan` varchar(255) COLLATE utf8mb4_unicode_ci NOT
6   NULL,
7   `waktu_pengajuan_refund` date DEFAULT NULL,
8   `status_refund` enum('disetujui','ditolak') COLLATE
9   utf8mb4_unicode_ci DEFAULT NULL,
10  `waktu_konfirmasi_refund` date DEFAULT NULL,
11  `created_at` timestamp NULL DEFAULT NULL,
12  `updated_at` timestamp NULL DEFAULT NULL
13 )

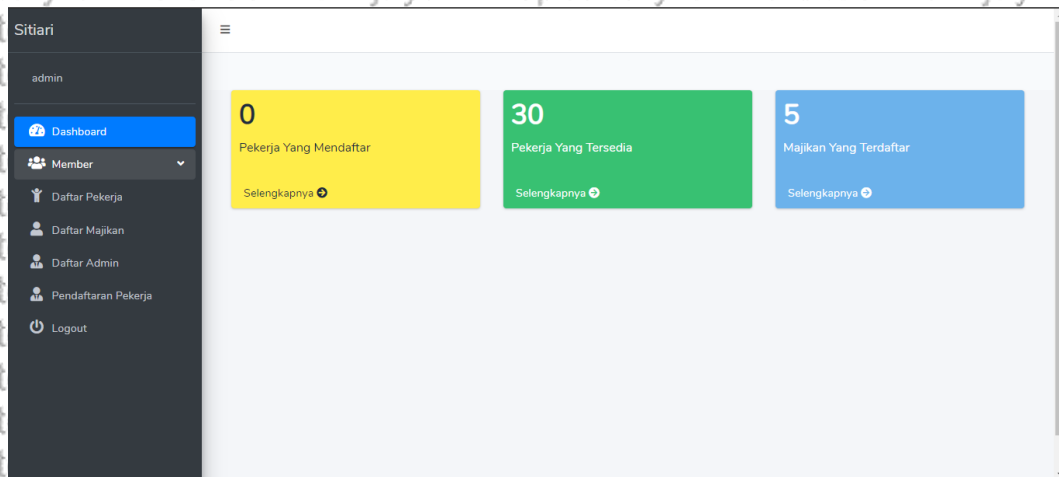
```

### 5.3.4 Implementasi Antarmuka

Tahapan ini akan memvisualisasikan implementasi dari antarmuka. Visualisasi yang ditampilkan akan berkaca pada perancangan antarmuka yang telah dilakukan sebelumnya. Pada sub bab ini, akan dijelaskan implementasi antarmuka admin dashboard, daftar majikan, Login, dan *register* majikan. Sub bab 5.2.4.1 sampai dengan sub bab 5.2.4.4 akan menjelaskan implementasi antarmuka secara lebih rinci.

#### 5.3.4.1 Implementasi Antarmuka Admin Dashboard

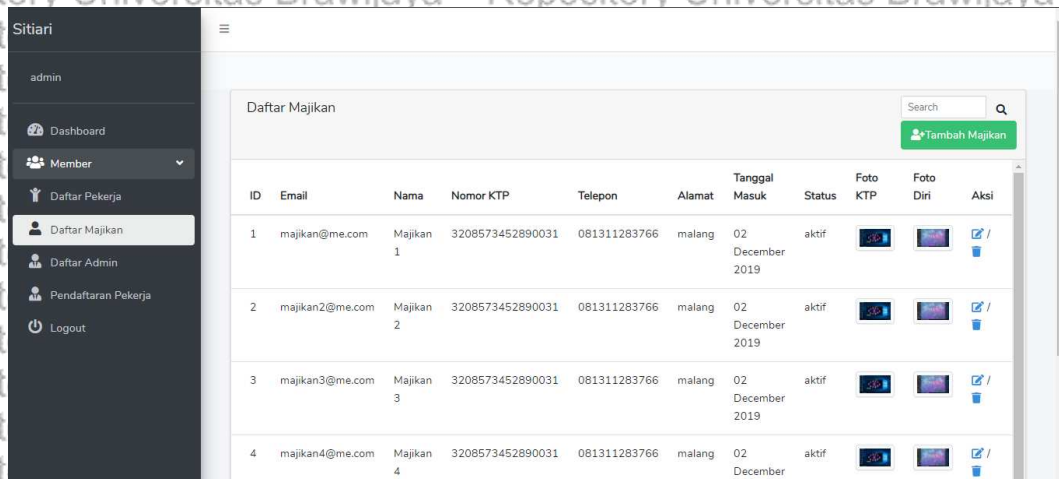
Gambar 5.13 akan memvisualisasikan hasil implementasi dari perancangan antarmuka *admin dashboard*.



**Gambar 5.13 Implementasi Antarmuka Admin Dashboard**

#### 5.3.4.2 Implementasi Antarmuka Daftar Majikan

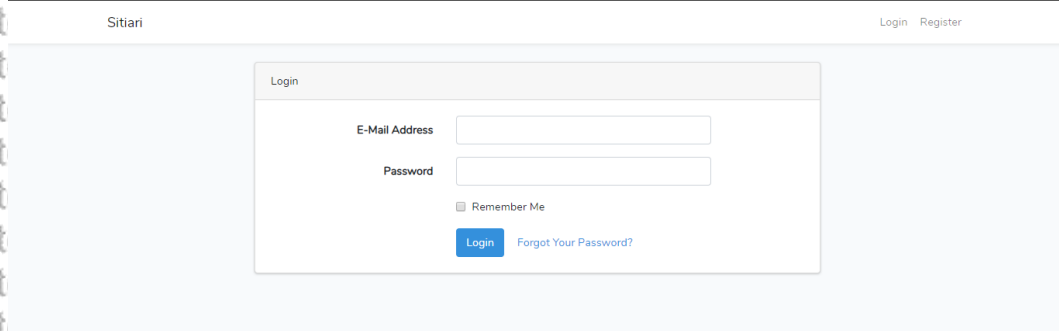
Gambar 5.14 akan memvisualisasikan hasil implementasi dari perancangan antarmuka daftar majikan.



Gambar 5.14 Implementasi Antarmuka Daftar Majikan

### 5.3.4.3 Implementasi Antarmuka Login

Gambar 5.15 akan memvisualisasikan hasil implementasi dari perancangan antarmuka *Login*.



Gambar 5.15 Implementasi Antarmuka Login

### 5.3.4.4 Implementasi Antarmuka Register Majikan

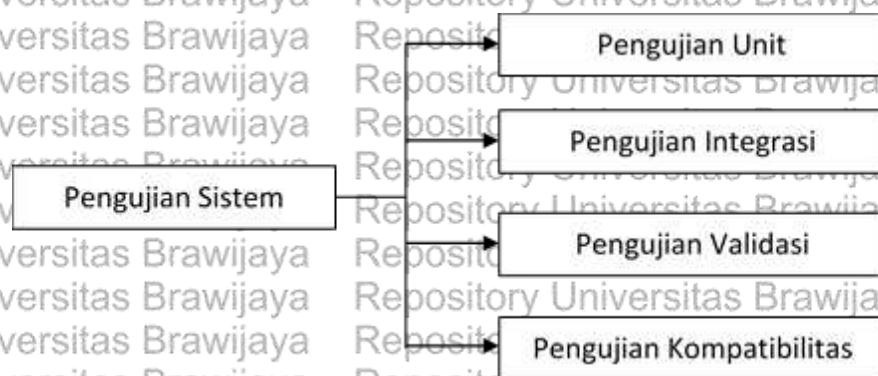
Gambar 5.16 akan memvisualisasikan hasil implementasi dari perancangan antarmuka *register* majikan.





## BAB 6 PENGUJIAN SISTEM

Tahapan ini akan melakukan uji coba terhadap sistem yang sudah diimplementasikan. Tujuan pengujian adalah agar seluruh kebutuhan, perancangan, dan implementasi yang sudah dilakukan sesuai dengan kebutuhan pengguna. Pada penelitian ini akan dilakukan pengujian dengan 4 jenis metode pengujian. Metode pengujian yang nantinya akan dilaksanakan adalah pengujian unit, lalu dilanjutkan dengan pengujian integrasi, setelah ini akan dilakukan pengujian validasi, dan yang terakhir akan dilakukan pengujian kompatibilitas. Gambar 6.1 akan menjabarkan struktur dari bab pengujian sistem.



Gambar 6.1 Struktur Bab Pengujian Sistem

### 6.1 Pengujian Unit

Tahapan ini merupakan sebuah tahapan yang memfokuskan pengujian kepada unit terkecil dari sebuah sistem. Unit terkecil yang dimaksud adalah method. Sebelum pengujian dilakukan, perancangan dan implementasi harus sudah dilakukan sebelumnya. Pada prakteknya, pengujian ini akan menguji tiga *method* yaitu *method submitPesanan*, *method deletePekerja*, dan *method terimaPekerja* dengan menggunakan metode *whitebox testing*.

#### 6.1.1 Pengujian Unit *Method submitPesanan*

##### 1. *Pseudocode*

Nama kelas: *PemesananController*

Nama metode: *submitPesanan*

Pada pengujian unit *method submitPesanan* akan menggunakan *pseudocode* sebagai dasar dalam pembuatan *flowgraph*. Tabel 6.1 akan memaparkan *pseudocode* dari metode *submitPesanan*.

Tabel 6.1 *Pseudocode* Algoritme *Method submitPesanan*

| Pseudocode algoritme method submitPesanan |       |
|---|-------|
| 1   | BEGIN |
| 2   | 1     |



```

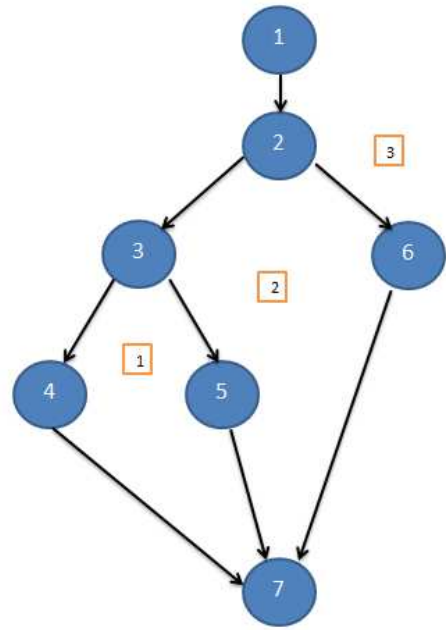
3  FIND majikan with(id = id_majikan of Request)
4  FIND pekerja with(id = id_pekerja of Request)
5  IF(majikan and pekerja is true) {
6    Now = Get value of date
7    TRY {
8      CREATE pemesanan(majikan_id, pekerja_id,
9      jumlah_bayar, waktu_pesan, pertama) VALUE(majikan_id
10     of Request, pekerja_id of Request, jumlah_bayar of
11     Request, waktu_pesan of Request, pertama of Request)
12     RETURNING pemesanan
13   }
14   CATCH {
15     RETURN response to JSON ("Terjadi Kesalahan")
16   }
17   Payload = array of (transaction_details,
18   customer_details, item_details)
19   Snap_token = Get value of snap_token
20   UPDATE pemesanan(snap_token) VALUE(snap_token)
21   RETURNING pemesanan
22   RETURN response to JSON (order and store success
23   status 201)
24 }
25 ELSE {
26   RETURN response to JSON ("Terjadi Kesalahan")
27 }
28 }
29 }
30 }
31 END

```

## 2. Basis Path Testing

### 2.1 Flow Graph

Gambar 6.2 akan memvisualisasikan penggambaran dari *flowgraph* yang didasari *pseudocode* pada pengujian metode *submitPesanan*.



Gambar 6.2 Flowgraph method submitPesanan

2.2 Cyclomatic Complexity

- $V(G) = \text{region total} = 3$
- $V(G) = \text{edge total} - \text{node total} + 2 = 8 - 7 + 2 = 3$
- $V(G) = \text{predicate node total} + 1 = 2 + 1 = 3$

2.3 Independent Path

- Jalur Independen 1: 1-2-6-7
- Jalur Independen 2: 1-2-3-5-7
- Jalur Independen 3: 1-2-3-4-7

Tabel 6.2 akan menampilkan kode program untuk driver yang digunakan dalam pengujian *method* submitPesanan.

Tabel 6.2 Source Code Driver pesanTest

| Source code | method | pesanTest                                 |
|-------------|--------|---|
| 1           | class  | pesanTest extends TestCase                |
| 2           | {      |   |
| 3           | use    | RefreshDatabase;                          |
| 4           | public | function testPath1()                      |
| 5           | {      |   |
| 6           |        | \$this->withoutExceptionHandling();       |
| 7           |        | \$this->actingAs(factory(Majikan::class)- |
| 8           |        | >create());                               |
| 9           |        | \$response = \$this->from('/home')-       |
| 10          |        | >post('/majikan/pemesanan/store');        |



```

11     'id_majikan' => 100,
12     'id_pekerja' => 1,
13     'jumlah_bayar' => 1500000,
14     'kategori' => 1,
15     );
16     $response->assertSeeText("Terjadi Kesalahan",
17     $escaped = true);
18     $this->assertTrue(true);
19     }
20
21     public function testPath2()
22     {
23         $this->withoutExceptionHandling();
24         $this->actingAs(factory(Majikan::class)-
25         >create());
26         $response = $this->from('/home')-
27         >post('/majikan/pemesanan/store', [
28             'id_majikan' => 2,
29             'id_pekerja' => 2,
30             'jumlahBayar' => 1234000,
31             'kategori' => 1,
32             ]);
33         $response->assertStatus(201);
34         $this->assertTrue(true);
35     }
36
37     public function testPath3()
38     {
39         $this->withoutExceptionHandling();
40         $this->actingAs(factory(Majikan::class)-
41         >create());
42         $response = $this->from('/home')-
43         >post('/majikan/pemesanan/store', [
44             'id_majikan' => 2,
45             'id_pekerja' => 2,
46             'kategori' => 1,
47             ]);
48         $response->assertSeeText("Terjadi
49     Kesalahan", $escaped = true);
50         $this->assertTrue(true);
51     }
52 }

```

Tabel 6.3 akan menjelaskan hasil dari pengujian metode submitPesanan secara lebih jelas.

**Tabel 6.3 Hasil pengujian unit *method* submitPesanan**

| No. | Jalur   | Tahapan Uji             | Hasil (Diharapkan)       | Hasil (Didapatkan) | Status |
|-----|---------|-------------------------|--------------------------|--------------------|--------|
| 1   | 1-2-6-7 | Memanggil <i>method</i> | Tidak terjadi penambahan | Sistem menampilkan | Valid  |



|   |           |   |  |   |       |
|---|-----------|---|--|---|-------|
|   |           | submitPesanan pada <i>class</i> Pemesanan Controller dengan menginputkan data yang tidak benar<br><br>id_majikan = 100, id_pekerja = 1, jumlah_bayar = 1500000, kategori = 1.                       | data pada saat method dijalankan, lalu system menampilkan pesan "Terjadi Kesalahan"      | pesan "Terjadi Kesalahan" dan tidak terjadi penambahan data.                                  |       |
| 2 | 1-2-3-5-7 | Memanggil <i>method</i> submitPesanan pada <i>class</i> Pemesanan Controller dengan menginputkan data yang tidak benar<br><br>id_majikan = 2, id_pekerja = 2, jumlah_bayar = 1234000, kategori = 1. | Terjadi penambahan data pada database pemesanan dan menghasilkan status 201.             | Mengembalikan status 201 dan terjadi penambahan data pada database pemesanan.                 | Valid |
| 3 | 1-2-3-4-7 | Memanggil <i>method</i> submitPesanan pada <i>class</i> Pemesanan Controller dengan menginputkan data yang tidak benar (tidak diinputkan seluruh data)<br><br>id_majikan = 2, id_pekerja = 2,       | Tidak terjadi penambahan pada adatabase dan system menampilkan pesan "Terjadi Kesalahan" | System menampilkan pesan "Terjadi Kesalahan" dan tidak terjadi penambahan data pada database. | Valid |



|  |                       |               |                       |                       |
|--|-----------------------|---------------|-----------------------|-----------------------|
|  | Universitas Brawijaya | kategori = 1. | Universitas Brawijaya | Universitas Brawijaya |
|--|-----------------------|---------------|-----------------------|-----------------------|

Hasil eksekusi dari driver terhadap *method* submitPesanan akan ditunjukkan pada Gambar 6.5. Pada gambar tersebut akan menampilkan status dari pengujian dan juga jumlah dari *test* yang dilakukan sesuai dengan *test case* yang telah didefinisikan sebelumnya.

```

C:\WINDOWS\system32\cmd.exe

C:\project\Sitiari-Final>php artisan test --filter pesanTest
Warning: TTY mode is not supported on Windows platform.

 RUNS  Tests\Feature\pesanTest
  • path1

Tests: 3 pending

 RUNS  Tests\Feature\pesanTest
  • path2

Tests: 1 passed, 2 pending

 RUNS  Tests\Feature\pesanTest
  • path3

Tests: 2 passed, 1 pending

 PASS  Tests\Feature\pesanTest
  [x] path1
  [x] path2
  [x] path3

Tests: 3 passed
Time: 0.79s

C:\project\Sitiari-Final>
    
```

Gambar 6.3 Hasil Pengujian Unit *Method* submitPesanan

### 6.1.2 Pengujian Unit *Method* deletePekerja

#### 1. Pseudocode

Nama klas: HomeController

Nama metode: deletePekerja

Pada pengujian unit *method* deletePekerja akan menggunakan *pseudocode* sebagai dasar dalam pembuatan *flowgraph*. Tabel 6.1 akan memaparkan *pseudocode* dari metode deletePekerja.

Tabel 6.4 Pseudocode Algoritme *Method* deletePekerja

| Pseudocode algoritme method deletePekerja |   |
|---|---|
| 1   | BEGIN   |
| 2   | 1   |
| 3   | FIND pekerja with(id = id of Request)         |
| 4   | IF (pekerja is TRUE) {                        |
| 5   | FIND detailPekerja with (id = id of Request)  |
| 6   | DELETE detailPekerja WITH(id = id of Request) |
|   | 2   |



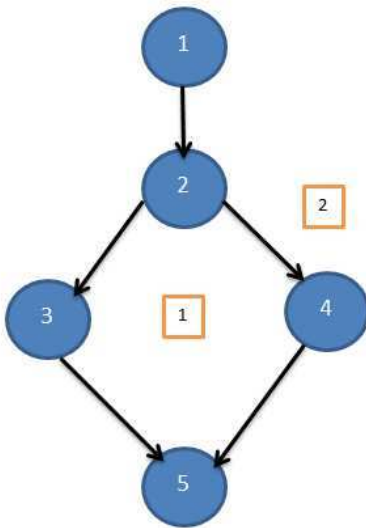
```

7 RETURNING pekerja
8 DELETE pekerja WITH(id = id of Request) RETURNING
9 pekerja
10
11 RETURN response to JSON ("Penghapusan Berhasil")
12 }
13 ELSE {
14 RETURN response to JSON ("Data Tidak Ditemukan")
15 }
16 END
    
```

2. Basis Path Testing

2.1 Flow Graph

Gambar 6.5 akan memvisualisasikan penggambaran dari *flowgraph* yang didasari *pseudocode* pada pengujian metode deletePekerja.



Gambar 6.4 Flowgraph method deletePekerja

2.2 Cyclomatic Complexity

- $V(G) = \text{region total} = 2$
- $V(G) = \text{edge total} - \text{node total} + 2 = 5 - 5 + 2 = 2$
- $V(G) = \text{predicate node total} + 1 = 1 + 1 = 2$

2.3 Independent Path

- Jalur independen 1: 1-2-3-5
- Jalur independen 2: 1-2-4-5

Tabel 6.5 akan menampilkan kode program untuk driver yang digunakan dalam pengujian *method* deletePekerja.



**Tabel 6.5 Source Code Driver deleteTest**

```

Source code method deleteTest
1 class deleteTest extends TestCase
2 {
3     use RefreshDatabase;
4     public function testPath1()
5     {
6         $this->withoutExceptionHandler();
7         $this->assertCount(1, Pekerja::all());
8         $this->assertCount(1, DetailPekerja::all());
9         $response = $this->from('/home')->
10 >post('/delete-pekerja', ['id' => 1]);
11 $this->assertCount(0, Pekerja::all());
12 $this->assertCount(0, DetailPekerja::all());
13 $response->assertRedirect('/home');
14 $response->assertSessionHas('sukses', $value =
15 'Penghapusan Berhasil');
16 $this->assertTrue(true);
17 }
18
19     public function testPath2()
20     {
21         $this->withoutExceptionHandler();
22         $this->assertCount(1, Pekerja::all());
23         $this->assertCount(1, DetailPekerja::all());
24         $response = $this->from('/home')->
25 >post('/delete-pekerja', ['id' => 2]);
26 $this->assertCount(1, Pekerja::all());
27 $this->assertCount(1, DetailPekerja::all());
28 $response->assertRedirect('/home');
29 $response->assertSessionHas('alert', $value =
30 'Data Tidak Ditemukan');
31 $this->assertTrue(true);
32 }
33 }
    
```

Tabel 6.6 akan menjelaskan hasil dari pengujian metode deletePekerja secara lebih jelas.

**Tabel 6.6 Hasil pengujian unit *method* deletePekerja**

| No. | Jalur   | Tahapan Uji   | Hasil (Diharapkan)   | Hasil (Didapatkan)   | Status |
|-----|---------|---|--|--|--------|
| 1   | 1-2-3-5 | Memanggil <i>method</i> deletePekerja pada <i>class</i> HomeController dengan data input yang benar | Data dengan id "1" dihapus dari database dan sistem menampilkan pesan "Penghapusan | Sistem menampilkan pesan "Penghapusan Berhasil" dan data pekerja dengan id "1" | Valid  |



|   |         |  |  |  |       |
|---|---------|--|--|--|-------|
|   |         | id = 1.  | Berhasil"  | telah dihapus dari database.                     |       |
| 2 | 1-2-4-5 | Memanggil <i>method</i> deletePekerja pada <i>class</i> HomeController dengan data input yang tidak benar<br>id = 2. | Sistem menampilkan pesan "Data Tidak Ditemukan". | Sistem menampilkan pesan "Data Tidak Ditemukan". | Valid |

Hasil eksekusi dari driver terhadap *method* deletePekerja akan ditunjukkan pada Gambar 6.5. Pada gambar tersebut akan menampilkan status dari pengujian dan juga jumlah dari *test* yang dilakukan sesuai dengan *test case* yang telah didefinisikan sebelumnya.

```

C:\WINDOWS\system32\cmd.exe

C:\project\Sitiari-Final>php artisan test --filter deleteTest
Warning: TTY mode is not supported on Windows platform.

 RUNS  Tests\Feature\deleteTest
  • path1

Tests:  2 pending

 RUNS  Tests\Feature\deleteTest
  • path2

Tests:  1 passed, 1 pending

 PASS  Tests\Feature\deleteTest
  [x] path1
  [x] path2

Tests:  2 passed
Time:   0.49s

C:\project\Sitiari-Final>
    
```

Gambar 6.5 Hasil Pengujian Unit *Method* deletePekerja

### 6.1.3 Pengujian Unit *Method* terimaPekerja

#### 1. Pseudocode

Nama kelas: HomeController

Nama metode: terimaPekerja

Pada pengujian unit *method* terimaPekerja akan menggunakan *pseudocode* sebagai dasar dalam pembuatan *flowgraph*. Tabel 6.4 akan memaparkan *pseudocode* dari metode terimaPekerja.



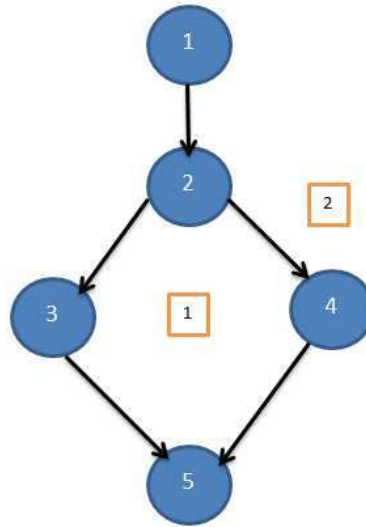
**Tabel 6.7 Pseudocode Algoritme Method terimaPekerja**

| Pseudocode algoritme method terimaPekerja |  |
|---|--|
| 1   | BEGIN <b>1</b>   |
| 2   |  |
| 3   | FIND pekerja with(id = id of Request) <b>2</b>           |
| 4   | IF (pekerja is TRUE) {                                   |
| 5   | UPDATE pekerja(status) VALUE("tersedia") RETURNING       |
| 6   | pekerja  |
| 7   | RETURN response to JSON ("Konfirmasi Berhasil") <b>3</b> |
| 8   | }  |
| 9   | ELSE {   |
| 10  | RETURN response to JSON ("Terjadi Kesalahan") <b>4</b>   |
| 11  | }  |
| 12  | END <b>5</b>   |

**2. Basis Path Testing**

**2.1 Flow Graph**

Gambar 6.6 akan memvisualisasikan penggambaran dari *flowgraph* yang didasari *pseudocode* pada pengujian metode terimaPekerja.



**Gambar 6.6 Flowgraph method terimaPekerja**

**2.2 Cyclomatic Complexity**

- $V(G) = \text{region total} = 2$
- $V(G) = \text{edge total} - \text{node total} + 2 = 5 - 5 + 2 = 2$
- $V(G) = \text{predicate node total} + 1 = 1 + 1 = 2$

**2.3 Independent Path**

- Jalur independen 1: 1-2-3-5



- Jalur independen 2: 1-2-4-5

Tabel 6.5 akan menampilkan kode program untuk driver yang digunakan dalam pengujian *method* terimaPekerja.

**Tabel 6.8 Source Code Driver terimaTest**

```
Source code method deleteTest
1 class terimaTest extends TestCase
2 {
3     use RefreshDatabase;
4     public function testPath1()
5     {
6         $this->withoutExceptionHandler();
7         $this->actingAs(factory(User::class)
8 >create());
9         $response = $this->from('/home')-
10 >post('/terima-pekerja', ['id' => 1]);
11         $response->assertRedirect('/home');
12         $response->assertSessionHas('sukses', $value =
13 'Konfirmasi Berhasil');
14         $this->assertTrue(true);
15     }
16
17     public function testPath2()
18     {
19         $this->withoutExceptionHandler();
20         $this->actingAs(factory(User::class)
21 >create());
22         $response = $this->from('/home')-
23 >post('/terima-pekerja', ['id' => 20]);
24         $response->assertRedirect('/home');
25         $response->assertSessionHas('alert', $value =
26 'Terjadi Kesalahan');
27         $this->assertTrue(true);
28     }
29 }
```

Tabel 6.9 akan memaparkan hasil dari pengujian unit metode terimaPekerja dengan lebih jelas.

**Tabel 6.9 Hasil pengujian unit *method* deletePekerja**

| No. | Jalur   | Tahapan Uji   | Hasil (Diharapkan)                              | Hasil (Didapatkan)                              | Status |
|-----|---------|---|---|---|--------|
| 1   | 1-2-3-5 | Memanggil <i>method</i> terimaPekerja pada <i>class</i> HomeController dengan id yang | Sistem menampilkan pesan "Konfirmasi Berhasil". | Sistem menampilkan pesan "Konfirmasi Berhasil". | Valid  |



|   |         |   |  |  |       |
|---|---------|---|--|--|-------|
|   |         | sesuai<br>id = 1.   |  |  |       |
| 2 | 1-2-4-5 | Memanggil<br><i>method</i><br>terimaPekerja<br>pada <i>class</i><br>HomeController<br>dengan id yang<br>tidak sesuai.<br>id = 20. | Sistem<br>menampilkan<br>pesan "Terjadi<br>Kesalahan". | Sistem<br>menampilkan<br>pesan "Terjadi<br>Kesalahan". | Valid |

Hasil eksekusi dari driver terhadap *method* terimaPekerja akan ditunjukkan pada Gambar 6.7. Pada gambar tersebut akan menampilkan status dari pengujian dan juga jumlah dari *test* yang dilakukan sesuai dengan *test case* yang telah didefinisikan sebelumnya.

```

C:\WINDOWS\system32\cmd.exe
C:\project\Sitiari-Final>php artisan test --filter terimaTest
Warning: TTY mode is not supported on Windows platform.

 RUNS  Tests\Feature\terimaTest
  • path1

Tests:  2 pending

 RUNS  Tests\Feature\terimaTest
  • path2

Tests:  1 passed, 1 pending

 PASS  Tests\Feature\terimaTest
  ☐ path1
  ☐ path2

Tests:  2 passed
Time:   0.52s

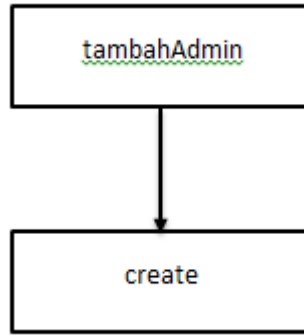
C:\project\Sitiari-Final>

```

Gambar 6.7 Hasil Pengujian Unit *Method* terimaPekerja

## 6.2 Pengujian Integrasi

Tahapan ini bertujuan untuk mengujidesain dari sebuah sistem. Pada prakteknya, pengujian ini akan dilakukan dengan menjalankan suatu fungsi yang didalamnya terdapat integrasi dari *method* lainnya. Pendekatan top down akan diglakukan dalam pengujian integrasi di penelitian ini. Relasi antara *method* tambahAdmin dan *method* create akan dipaparkan dalam Gambar 6.8.



**Gambar 6.6 Diagram Hierarki Pengujian Integrasi Method tambahAdmin**

Tabel 6.10 akan menjabarkan secara spesifik langkah yang akan dilakukan dalam pengujian integrasi ini.

**Tabel 6.7 Langkah Uji Pengujian Integrasi**

| No. | Langkah Uji               | Keterangan   |
|-----|---------------------------|--|
| 1   | tambahAdmin () + create() | Metode tambahAdmin berfungsi untuk melakukan penyimpanan data yang diinputkan oleh aktor. Metode tambahAdmin akan dijalankan dengan bantuan metode stubCreate. Untuk mengetahui apakah metode tambahAdmin berhasil dijalankan maka akan diisikan nilai string "Add Success". |

Stub digunakan dalam tahap awal pengujian integrasi sebagai pembantu pengujian metode tambahAdmin dan metode create. Metode yang akan digunakan untuk membantu adalah metode stubCreate. Metode stubCreate akan diberikan kondisi tambahAdmin dengan masukan bertipe data string "Add Success". Tabel 6.11 akan menjabarkan kode program untuk method tambahAdmin yang diintegrasikan dengan stubCreate.

**Tabel 6.81 Source Code Method tambahAdmin**

```

Source code
1  $this->validate($request,[
2  'name' => 'required|string|max:191',
3  'email' => 'required|string|email|max:191|unique:users',
4  'password' => 'required|string|min:4'
5  ]);
6
7  $hashedPassword = Hash::make($request['password']);
8  $request->merge(['password' => $hashedPassword]);
9
10 $admin = $this->stubCreate('Add Success');
11 return response()->json(compact('admin'),
12 201);
13
  
```

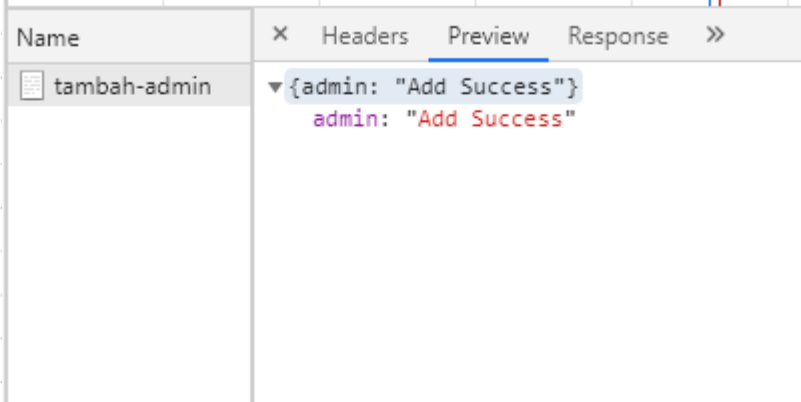


Tabel 6.12 akan menjabarkan kode program metode stubCreate yang digunakan pada pengujian ini.

**Tabel 6.12 Source Code Method stubCreate**

| Source code                                   |
|---|
| 1 public function stubCreate(\$tambahAdmin) { |
| 2 return \$tambahAdmin;                       |
| 3   |
| 4 }   |

Dengan menjalankan method tambahAdmin, maka akan didapatkan hasil pengujian integrasi. Pada method tambahAdmin akan diuji integrasi dari method tersebut dengan stub yang telah dibuat yaitu stubCreate. Ketika stubCreate dipanggil maka akan akan megirimkan pesan "Add Success". Hal tersebut menandakan bahwa method tambahAdmin berhasil diintegrasikan dengan stubCreate. Hasil pengujian dibuktikan dengan Gambar 6.9.



**Gambar 6.7 Hasil Pengujian Method tambahAdmin Menggunakan stubCreate**

### 6.3 Pengujian Validasi

Tahapan pengujian ini akan menguji sistem dengan cara menjalankan skenario fungsional yang sbeelumnya telah didefinisikan. Tujuan dari tahapan ini adalah memastikan kebutuhan berjalan dengan benar. Dan output yang didapatkan akan sesuai dengan ekspektasi dan juga ssuai dengan yang sudah didefinisikan pada bab sebelumnya. Pada Tabel 6.12 sampai dengan Tabel 6.67 akan dipaparkan hasil dari pengujian validasi sistem.

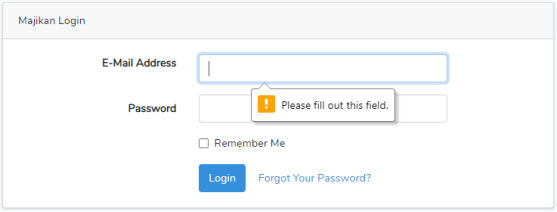
Hasil dari pengujian validasi yang dilakukan pada *use case Login* membuahkan nilai valid. Tabel 6.13 akan menjelaskan secara lebih jelas terkait dengan hasil validasi yang menyangkut *use case* ini.

**Tabel 6.13 Pengujian Validasi Use Case Login**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_001  |
| Nama Fungsi           | Login  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Memasuki halaman Login.</li> <li>2. Mengisi <i>form</i> Login dengan data yang benar, email = "majikan@me.com" dan password "qwertyuiop".</li> <li>3. Menekan tombol Login.</li> </ol> |
| Hasil yang Diharapkan | Sistem mengarahkan aktor ke halaman utama.   |
| Hasil yang Didapatkan | Sistem mengarahkan aktor ke halaman utama.   |
| Status                | Valid  |

Hasil dari pengujian validasi yang dilakukan pada *use case Login* alternatif 1 membuahkan nilai valid. Tabel 6.14 akan menjelaskan secara lebih jelas terkait dengan hasil validasi yang menyangkut *use case* ini.

**Tabel 6.14 Pengujian Validasi Use Case Login Alternatif 1**

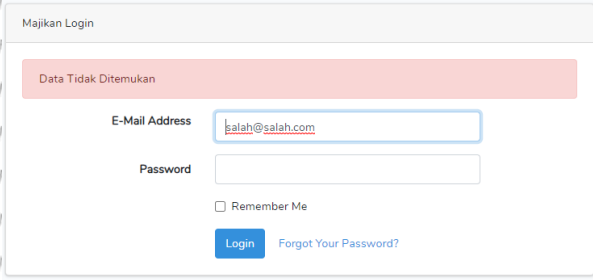
|  |  |
|--|--|
| Kode Kebutuhan   | PRT_F_001  |
| Nama Fungsi  | Login  |
| Prosedur Pengujian   | <ol style="list-style-type: none"> <li>1. Memasuki halaman Login.</li> <li>2. Tidak mengisikan field pada form.</li> <li>3. Menekan tombol Login.</li> </ol> |
| Hasil yang Diharapkan  | Sistem menampilkan pesan "Silahkan isi Field".   |
| Hasil yang Didapatkan  | Sistem menampilkan pesan "Silahkan isi Field".   |
|  |  |
| Status   | Valid  |

Hasil dari pengujian validasi yang dilakukan pada *use case Login* alternatif 2 membuahkan nilai valid. Tabel 6.15 akan menjelaskan secara lebih jelas terkait dengan hasil validasi yang menyangkut *use case* ini.

**Tabel 6.15 Pengujian Validasi Use Case Login Alternatif 2**





|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_001  |
| Nama Fungsi           | Login  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>Memasuki halaman Login.</li> <li>Mengisikan field pada form yang tersedia dengan data yang salah, email = "salah@salah.com" dan password = "salah".</li> <li>Menekan tombol Login.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Data Tidak Ditemukan"  |
| Hasil yang Didapatkan | <p>Sistem menampilkan pesan "Data Tidak Ditemukan"</p>    |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* daftar sebagai pekerja membuahkn nilai valid. Tabel 6.13 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.16 Pengujian Validasi Use Case Daftar Sebagai Pekerja**

|                    |  |
|--------------------|--|
| Kode Kebutuhan     | PRT_F_002  |
| Nama Fungsi        | Daftar Sebagai Pekerja   |
| Prosedur Pengujian | <ol style="list-style-type: none"> <li>Memasuki halaman pendaftaran pekerja.</li> <li>Mengisikan field pada form yang tersedia dengan data yang benar, nama = "Pekerja", email = "pekerja@me.com", password = "qwertyuiop", konfirmasi password = "qwertyuiop", nomor ktp = "3201012879687782", tanggal lahir = "16/06/1987", nomor telepon =</li> </ol> |

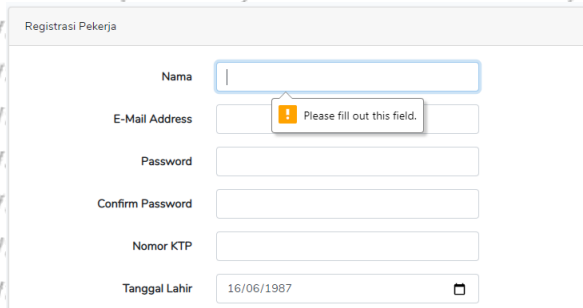


|                       |   |
|-----------------------|---|
|                       | <p>“081345728377”, alamat = “Malang”, Keahlian = “Memasak”, Agama = “Islam”, jenis pekerjaan = “Pembantu”, foto ktp = “file foto ktp”, foto diri “file foto diri”.</p> <p>3. Menekan tombol daftar.</p> |
| Hasil yang Diharapkan | Sistem melakukan penyimpanan data berdasarkan nilai yang telah diisi pada field-field yang ada di dalam form dan mengarahkan aktor ke halaman Login.  |
| Hasil yang Didapatkan | Sistem melakukan penyimpanan data berdasarkan nilai yang telah diisi pada field-field yang ada di dalam form dan mengarahkan aktor ke halaman Login.  |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* daftar sebagai pekerja alternatif 1 membuah nilai valid. Tabel 6.17 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.17 Pengujian Validasi Use Case Daftar Sebagai Pekerja Alternatif 1**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_002   |
| Nama Fungsi           | Daftar Sebagai Pekerja  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Memasuki halaman pendaftaran pekerja.</li> <li>2. Mengosongkan seluruh field pada form.</li> <li>3. Menekan tombol daftar.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan “silahkan isi field”.  |
| Hasil yang Didapatkan | Sistem menampilkan pesan “silahkan isi field”.  |
| Status                | Valid   |





Hasil dari pengujian validasi *use case* daftar sebagai majikan membuahkan nilai valid. Tabel 6.18 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.18 Pengujian Validasi Use Case Daftar Sebagai Majikan**

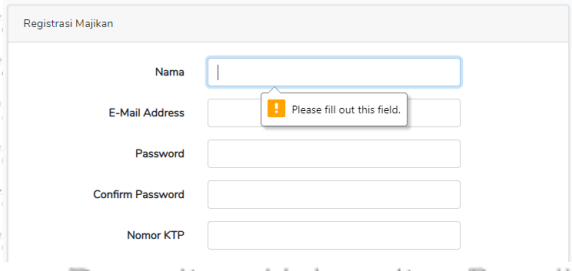
|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_003   |
| Nama Fungsi           | Daftar Sebagai Majikan  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>Memasuki halaman pendaftaran majikan.</li> <li>Mengisikan field pada form yang tersedia dengan data yang benar, nama = "Majikan", email = "majikan@me.com", password = "qwertyuiop", konfirmasi password = "qwertyuiop", nomor ktp = "3201012879687782", nomor telepon = "081345728377", alamat = "Malang", foto ktp = "file foto ktp", foto diri "file foto diri".</li> <li>Menekan tombol daftar.</li> </ol> |
| Hasil yang Diharapkan | Sistem melakukan penyimpanan data berdasarkan nilai yang telah diisi pada field-field yang ada di dalam form dan mengarahkan aktor ke halaman Login.  |
| Hasil yang Didapatkan | Sistem melakukan penyimpanan data berdasarkan nilai yang telah diisi pada field-field yang ada di dalam form dan mengarahkan aktor ke halaman Login.  |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* daftar sebagai majikan alternatif 1 membuahkan nilai valid. Tabel 6.19 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.19 Pengujian Validasi Use Case Daftar Sebagai Majikan Alternatif 1**

|                    |   |
|--------------------|---|
| Kode Kebutuhan     | PRT_F_003   |
| Nama Fungsi        | Daftar Sebagai Majikan  |
| Prosedur Pengujian | <ol style="list-style-type: none"> <li>Memasuki halaman pendaftaran majikan.</li> </ol> |



|                       |   |
|-----------------------|---|
|                       | <ol style="list-style-type: none"> <li>2. Mengosongkan seluruh field pada form.</li> <li>3. Menekan tombol daftar.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "silahkan isi field".  |
| Hasil yang Didapatkan | Sistem menampilkan pesan "silahkan isi field".  |
|                       |   |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* masuk sebagai admin membuahkan nilai valid. Tabel 6.20 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.20 Pengujian Validasi Use Case Masuk Sebagai Admin**

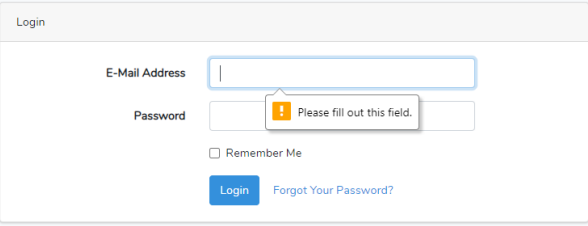
|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_004  |
| Nama Fungsi           | Masuk Sebagai Admin  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Memasuki halaman Login admin.</li> <li>2. Mengisi <i>form</i> Login dengan data yang benar, email = "admin@me.com" dan password "qwertyuiop".</li> <li>3. Menekan tombol Login.</li> </ol> |
| Hasil yang Diharapkan | Sistem mengarahkan aktor ke halaman <i>dashboard admin</i> .   |
| Hasil yang Didapatkan | Sistem mengarahkan aktor ke halaman <i>dashboard admin</i> .   |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* masuk sebagai admin alternatif 1 membuahkan nilai valid. Tabel 6.21 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.21 Pengujian Validasi Use Case Masuk Sebagai Admin Alternatif 1**

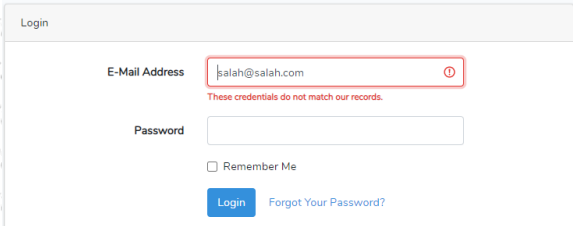
|                |           |
|----------------|-----------|
| Kode Kebutuhan | PRT_F_004 |
|----------------|-----------|



|                       |  |
|-----------------------|--|
| Nama Fungsi           | Masuk Sebagai Admin  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Memasuki halaman Login admin.</li> <li>2. Tidak mengisikan field pada form.</li> <li>3. Menekan tombol Login.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Silahkan isi Field".   |
| Hasil yang Didapatkan | <p>Sistem menampilkan pesan "Silahkan isi Field".</p>                            |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* masuk sebagai admin alternatif 2 membuahkan nilai valid. Tabel 6.22 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.22 Pengujian Validasi Use Case Masuk Sebagai Admin Alternatif 2**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_E_004   |
| Nama Fungsi           | Masuk Sebagai Admin   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Memasuki halaman Login admin.</li> <li>2. Mengisikan field pada form yang tersedia dengan data yang salah, email = "salah@salah.com" dan password = "salah".</li> <li>3. Menekan tombol Login.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Data Tidak Ditemukan".  |
| Hasil yang Didapatkan | <p>Sistem menampilkan pesan "Data Tidak Ditemukan".</p>   |



|        |       |
|--------|-------|
| Status | Valid |
|--------|-------|

Hasil dari pengujian validasi *use case* melihat daftar pekerja yang tersedia membuahkan nilai valid. Tabel 6.23 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.23 Pengujian Validasi Use Case Melihat Daftar Pekerja Yang Tersedia**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_005  |
| Nama Fungsi           | Melihat Daftar Pekerja Yang Tersedia             |
| Prosedur Pengujian    | 1. Membuka halaman utama.                        |
| Hasil yang Diharapkan | Sistem menampilkan daftar pekerja yang tersedia. |
| Hasil yang Didapatkan | Sistem menampilkan daftar pekerja yang tersedia. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* mengajukan permohonan resign membuahkan nilai valid. Tabel 6.24 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.24 Pengujian Validasi Use Case Mengajukan Permohonan Resign**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_006  |
| Nama Fungsi           | Mengajukan Permohonan Resign   |
| Prosedur Pengujian    | 1. Membuka halaman resign pekerja.<br>2. Menekan tombol resign.                  |
| Hasil yang Diharapkan | Sistem menyimpan data resign dan mengarahkan aktor ke halaman dashboard pekerja. |
| Hasil yang Didapatkan | Sistem menyimpan data resign dan mengarahkan aktor ke halaman dashboard pekerja. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* melihat daftar riwayat pekerjaan membuahkan nilai valid. Tabel 6.25 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.25 Pengujian Validasi Use Case Melihat Daftar Riwayat Pekerjaan**



|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_007                                    |
| Nama Fungsi           | Melihat Daftar Riwayat Pekerjaan             |
| Prosedur Pengujian    | 1. Membuka halaman riwayat pekerjaan.        |
| Hasil yang Diharapkan | Sistem menampilkan daftar riwayat pekerjaan. |
| Hasil yang Didapatkan | Sistem menampilkan daftar riwayat pekerjaan. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* melihat daftar riwayat penggajian membuah nilai valid. Tabel 6.26 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.26 Pengujian Validasi Use Case Melihat Daftar Riwayat Penggajian**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_E_008                                     |
| Nama Fungsi           | Melihat Daftar Riwayat Penggajian             |
| Prosedur Pengujian    | 1. Membuka halaman riwayat penggajian.        |
| Hasil yang Diharapkan | Sistem menampilkan daftar riwayat penggajian. |
| Hasil yang Didapatkan | Sistem menampilkan daftar riwayat penggajian. |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* memesan jasa pekerja membuah nilai valid. Tabel 6.27 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.27 Pengujian Validasi Use Case Memesan Jasa Pekerja**

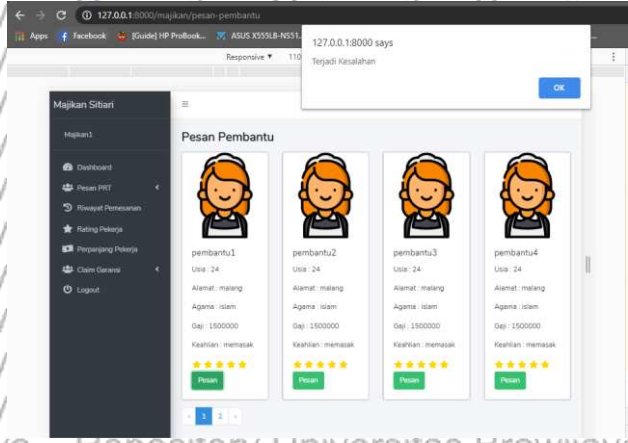
|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_009  |
| Nama Fungsi           | Memesan Jasa Pekerja   |
| Prosedur Pengujian    | 1. Membuka halaman pesan pekerja.<br>2. Memilih pekerja.<br>3. Menekan tombol pesan.           |
| Hasil yang Diharapkan | Sistem menyimpan data pemesanan ke dalam database dan mengarahkan aktor ke halaman pembayaran. |
| Hasil yang Didapatkan | Sistem menyimpan data pemesanan ke dalam database dan mengarahkan aktor ke halaman             |



|        |             |
|--------|-------------|
|        | pembayaran. |
| Status | Valid       |

Hasil dari pengujian validasi *use case* memesan jasa pekerja alternatif membuahkn nilai valid. Tabel 6.28 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.28 Pengujian Validasi Use Case Memesan Jasa Pekerja Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_009  |
| Nama Fungsi           | Memesan Jasa Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman pesan pekerja.</li> <li>2. Memilih pekerja.</li> <li>3. Menekan tombol pesan.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Terjadi Kesalahan"   |
| Hasil yang Didapatkan | <p>Sistem menampilkan pesan "Terjadi Kesalahan"</p>             |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* claim garansi membuahkn nilai valid. Tabel 6.29 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.29 Pengujian Validasi Use Case Claim Garansi**

|                    |  |
|--------------------|--|
| Kode Kebutuhan     | PRT_F_010  |
| Nama Fungsi        | Claim Garansi  |
| Prosedur Pengujian | <ol style="list-style-type: none"> <li>1. Membuka halaman claim garansi.</li> <li>2. Memilih transaksi yang ingin di claim.</li> </ol> |





|                       |  |
|-----------------------|--|
|                       | 3. Menekan tombol claim.                               |
|                       | 4. Memasukkan alasan claim.                            |
|                       | 5. Menekan tombol submit.                              |
| Hasil yang Diharapkan | Sistem menyimpan data claim garansi ke dalam database. |
| Hasil yang Didapatkan | Sistem menyimpan data claim garansi ke dalam database. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* claim garansi alternatif 1 membuahkan nilai valid. Tabel 6.30 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.30 Pengujian Validasi Use Case Claim Garansi Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_010  |
| Nama Fungsi           | Claim Garansi  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman claim garansi.</li> <li>2. Memilih transaksi yang ingin di claim.</li> <li>3. Menekan tombol claim.</li> <li>4. Tidak memasukkan alasan claim.</li> <li>5. Menekan tombol submit.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Silahkan isi field"  |
| Hasil yang Didapatkan | Sistem menampilkan pesan "Silahkan isi field"  |
|                       |  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* perpanjangan masa kerja PRT membuahkan nilai valid. Tabel 6.31 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.



**Tabel 6.31 Pengujian Validasi Use Case Perpanjang Masa Kerja PRT**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_011   |
| Nama Fungsi           | Perpanjang Masa Kerja PRT   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman perpanjang pekerja.</li> <li>2. Memilih pekerja yang ingin di perpanjang.</li> <li>3. Menekan tombol perpanjang.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data perpanjangan dan mengarahkan aktor ke halaman pembayaran.   |
| Hasil yang Didapatkan | Sistem menyimpan data perpanjangan dan mengarahkan aktor ke halaman pembayaran.   |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* rating pekerja membuahkan nilai valid. Tabel 6.32 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.32 Pengujian Validasi Use Case Rating Pekerja**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_012  |
| Nama Fungsi           | Rating Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman rating pekerja.</li> <li>2. Memilih pekerja yang ingin di rating.</li> <li>3. Menekan tombol rating.</li> <li>4. Mengisikan jumlah rating.</li> <li>5. Menekan tombol submit.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data rating sesuai dengan inputan aktor.  |
| Hasil yang Didapatkan | Sistem menyimpan data perpanjangan dan mengarahkan aktor ke halaman pembayaran.  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* melihat daftar riwayat pemesanan membuahkan nilai valid. Tabel 6.33 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.33 Pengujian Validasi Use Case Melihat Daftar Riwayat Pemesanan**



|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_013                                  |
| Nama Fungsi           | Melihat Daftar Riwayat Pemesanan           |
| Prosedur Pengujian    | 1. Membuka halaman riwayat pemesanan.      |
| Hasil yang Diharapkan | Sistem menampilkan data riwayat pemesanan. |
| Hasil yang Didapatkan | Sistem menampilkan data riwayat pemesanan. |
| Status                | Valid                                      |

Hasil dari pengujian validasi *use case* melihat daftar riwayat claim garansi membuahkan nilai valid. Tabel 6.34 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.34 Pengujian Validasi Use Case Melihat Daftar Riwayat Claim Garansi**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_014                                      |
| Nama Fungsi           | Melihat Daftar Riwayat Claim Garansi           |
| Prosedur Pengujian    | 1. Membuka halaman riwayat claim garansi.      |
| Hasil yang Diharapkan | Sistem menampilkan data riwayat claim garansi. |
| Hasil yang Didapatkan | Sistem menampilkan data riwayat claim garansi. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* melihat daftar admin membuahkan nilai valid. Tabel 6.35 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.35 Pengujian Validasi Use Case Melihat Daftar Admin**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_015  |
| Nama Fungsi           | Melihat Daftar Admin                                       |
| Prosedur Pengujian    | 1. Membuka halaman daftar admin.                           |
| Hasil yang Diharapkan | Sistem menampilkan data admin yang terdaftar dalam sistem. |
| Hasil yang Didapatkan | Sistem menampilkan data admin yang                         |



|        |                         |
|--------|-------------------------|
|        | terdaftar dalam sistem. |
| Status | Valid                   |

Hasil dari pengujian validasi *use case* edit daftar admin membuahkan nilai valid. Tabel 6.36 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.36 Pengujian Validasi Use Case Edit Daftar Admin**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_016  |
| Nama Fungsi           | Edit Daftar Admin  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar admin.</li> <li>2. Memilih admin yang akan diubah data nya.</li> <li>3. Menekan tombol edit.</li> <li>4. Mengisikan data yang ingin diubah.</li> <li>5. Menekan tombol edit.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data sesuai dengan inputan yang ada di dalam form.  |
| Hasil yang Didapatkan | Sistem menyimpan data sesuai dengan inputan yang ada di dalam form.  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* hapus daftar admin membuahkan nilai valid. Tabel 6.37 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.37 Pengujian Validasi Use Case Hapus Daftar Admin**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_017  |
| Nama Fungsi           | Hapus Daftar Admin   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar admin.</li> <li>2. Memilih admin yang akan dihapus data nya.</li> <li>3. Menekan tombol hapus.</li> <li>4. Menekan tombol hapus pada popup konfirmasi.</li> </ol> |
| Hasil yang Diharapkan | Sistem menghapus data admin terpilih.  |



|                       |                                       |
|-----------------------|---------------------------------------|
| Hasil yang Didapatkan | Sistem menghapus data admin terpilih. |
| Status                | Valid                                 |

Hasil dari pengujian validasi *use case* melihat daftar pekerja membuahakan nilai valid. Tabel 6.38 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.38 Pengujian Validasi Use Case Melihat Daftar Pekerja**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_018  |
| Nama Fungsi           | Melihat Daftar Pekerja                                       |
| Prosedur Pengujian    | 1. Membuka halaman daftar pekerja.                           |
| Hasil yang Diharapkan | Sistem menampilkan data pekerja yang terdaftar dalam sistem. |
| Hasil yang Didapatkan | Sistem menampilkan data pekerja yang terdaftar dalam sistem. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* edit daftar pekerja membuahakan nilai valid. Tabel 6.39 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.39 Pengujian Validasi Use Case Edit Daftar Pekerja**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_019  |
| Nama Fungsi           | Edit Daftar Pekerja  |
| Prosedur Pengujian    | 1. Membuka halaman daftar pekerja.<br>2. Memilih pekerja yang akan diubah data nya.<br>3. Menekan tombol edit.<br>4. Mengisikan data yang ingin diubah.<br>5. Menekan tombol edit. |
| Hasil yang Diharapkan | Sistem menyimpan data sesuai dengan inputan yang ada di dalam form.  |
| Hasil yang Didapatkan | Sistem menyimpan data sesuai dengan inputan yang ada di dalam form.  |
| Status                | Valid  |



Hasil dari pengujian validasi *use case* hapus daftar pekerja membuahakan nilai valid. Tabel 6.40 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.40 Pengujian Validasi Use Case Hapus Daftar Pekerja**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_020  |
| Nama Fungsi           | Hapus Daftar Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar pekerja.</li> <li>2. Memilih pekerja yang akan dihapus data nya.</li> <li>3. Menekan tombol hapus.</li> <li>4. Menekan tombol hapus pada popup konfirmasi.</li> </ol> |
| Hasil yang Diharapkan | Sistem menghapus data pekerja terpilih.  |
| Hasil yang Didapatkan | Sistem menghapus data pekerja terpilih.  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* hapus daftar pekerja alternatif 1 membuahakan nilai valid. Tabel 6.41 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.41 Pengujian Validasi Use Case Hapus Daftar Pekerja Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_020  |
| Nama Fungsi           | Hapus Daftar Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar pekerja.</li> <li>2. Memilih pekerja yang akan dihapus data nya.</li> <li>3. Menekan tombol hapus.</li> <li>4. Menekan tombol hapus pada popup konfirmasi.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Data Tidak Ditemukan"  |
| Hasil yang Didapatkan | Sistem menampilkan pesan "Data Tidak Ditemukan"  |



|        |       |
|--------|-------|
|        |       |
| Status | Valid |

Hasil dari pengujian validasi *use case* melihat daftar majikan membuah nilai valid. Tabel 6.42 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.42 Pengujian Validasi Use Case Melihat Daftar Majikan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_021  |
| Nama Fungsi           | Melihat Daftar Majikan                                       |
| Prosedur Pengujian    | 1. Membuka halaman daftar majikan.                           |
| Hasil yang Diharapkan | Sistem menampilkan data majikan yang terdaftar dalam sistem. |
| Hasil yang Didapatkan | Sistem menampilkan data majikan yang terdaftar dalam sistem. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* edit daftar majikan membuah nilai valid. Tabel 6.43 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.43 Pengujian Validasi Use Case Edit Daftar Majikan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_022  |
| Nama Fungsi           | Edit Daftar Majikan  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar majikan.</li> <li>2. Memilih majikan yang akan diubah data nya.</li> <li>3. Menekan tombol edit.</li> <li>4. Mengisikan data yang ingin diubah.</li> <li>5. Menekan tombol edit.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data sesuai dengan inputan yang ada di dalam form.  |
| Hasil yang Didapatkan | Sistem menyimpan data sesuai dengan  |



|        |                                 |
|--------|---------------------------------|
|        | inputan yang ada di dalam form. |
| Status | Valid                           |

Hasil dari pengujian validasi *use case* hapus daftar majikan membuahkan nilai valid. Tabel 6.44 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.44 Pengujian Validasi Use Case Hapus Daftar Majikan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_E_023  |
| Nama Fungsi           | Hapus Daftar Majikan   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar majikan.</li> <li>2. Memilih majikan yang akan dihapus data nya.</li> <li>3. Menekan tombol hapus.</li> <li>4. Menekan tombol hapus pada popup konfirmasi.</li> </ol> |
| Hasil yang Diharapkan | Sistem menghapus data majikan terpilih.  |
| Hasil yang Didapatkan | Sistem menghapus data majikan terpilih.  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* tambah admin membuahkan nilai valid. Tabel 6.45 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.45 Pengujian Validasi Use Case Tambah Admin**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_E_024   |
| Nama Fungsi           | Tambah Admin  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar admin.</li> <li>2. Menekan tombol tambah admin.</li> <li>3. Mengisikan data secara benar, nama = "admin", email = "admin@me.com", password = "qwertyuiop"</li> <li>4. Menekan tombol tambah pada popup.</li> </ol> |
| Hasil yang Diharapkan | Sistem mendaftarkan akun admin baru sesuai dengan data pada field form.   |
| Hasil yang Didapatkan | Sistem mendaftarkan akun admin baru sesuai  |



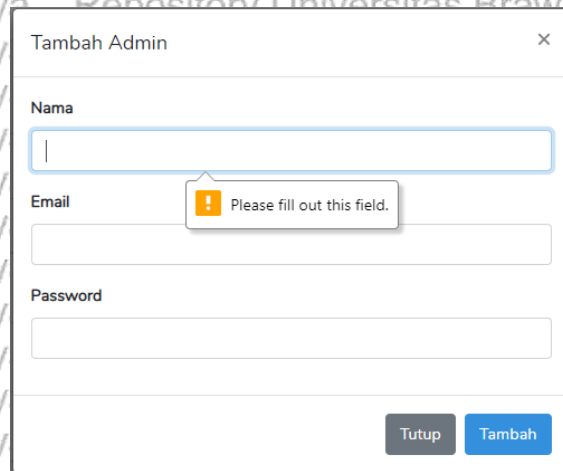


|        |                              |
|--------|------------------------------|
|        | dengan data pada field form. |
| Status | Valid                        |

Hasil dari pengujian validasi *use case* tambah admin alternatif 1 membuahkn nilai valid. Tabel 6.46 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.46 Pengujian Validasi Use Case Tambah Admin Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_024  |
| Nama Fungsi           | Tambah Admin   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar admin.</li> <li>2. Menekan tombol tambah admin.</li> <li>3. Tidak mengisikan data pada form.</li> <li>4. Menekan tombol tambah pada popup.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Silahkan isi field".   |
| Hasil yang Didapatkan | Sistem menampilkan pesan "Silahkan isi field".   |



|        |       |
|--------|-------|
| Status | Valid |
|--------|-------|

Hasil dari pengujian validasi *use case* tambah majikan membuahkn nilai valid. Tabel 6.47 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.47 Pengujian Validasi Use Case Tambah Majikan**

|                    |  |
|--------------------|--|
| Kode Kebutuhan     | PRT_F_025  |
| Nama Fungsi        | Tambah Majikan   |
| Prosedur Pengujian | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar majikan.</li> </ol> |



|                       |   |
|-----------------------|---|
|                       | <ol style="list-style-type: none"> <li>Menekan tombol tambah majikan.</li> <li>Mengisikan field pada form yang tersedia dengan data yang benar, nama = "Majikan", email = "majikan@me.com", password = "qwertyuiop", konfirmasi password = "qwertyuiop", nomor ktp = "3201012879687782", nomor telepon = "081345728377", alamat = "Malang", foto ktp = "file foto ktp", foto diri "file foto diri".</li> <li>Menekan tombol tambah pada popup.</li> </ol> |
| Hasil yang Diharapkan | Sistem mendaftarkan akun majikan baru sesuai dengan data pada field form.   |
| Hasil yang Didapatkan | Sistem mendaftarkan akun majikan baru sesuai dengan data pada field form.   |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* tambah majikan alternatif 1 membuahkan nilai valid. Tabel 6.48 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.48 Pengujian Validasi Use Case Tambah Majikan Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_025  |
| Nama Fungsi           | Tambah Majikan   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>Membuka halaman daftar majikan.</li> <li>Menekan tombol tambah majikan.</li> <li>Tidak mengisikan data pada form.</li> <li>Menekan tombol tambah pada popup.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Silahkan isi field".   |
| Hasil yang Didapatkan | Sistem menampilkan pesan "Silahkan isi field".   |



|        |       |
|--------|-------|
|        | Valid |
| Status | Valid |

Hasil dari pengujian validasi *use case* tambah pekerja membuahkn nilai valid. Tabel 6.49 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.49 Pengujian Validasi Use Case Tambah Pekerja**

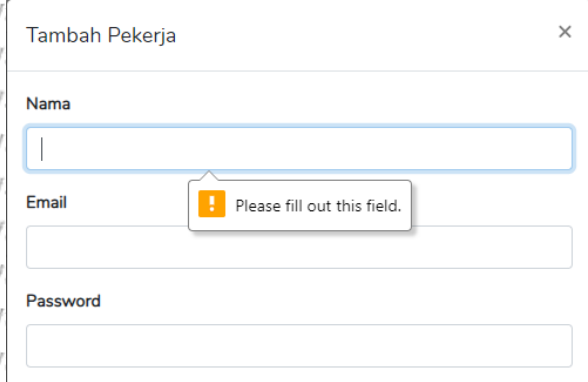
|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_026   |
| Nama Fungsi           | Tambah Pekerja  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar pekerja.</li> <li>2. Menekan tombol tambah pekerja.</li> <li>3. Mengisikan field pada form yang tersedia dengan data yang benar, nama = "Pekerja", email = "pekerja@me.com", password = "qwertyuiop", konfirmasi password = "qwertyuiop", nomor ktp = "3201012879687782", tanggal lahir = "16/06/1987", nomor telepon = "081345728377", alamat = "Malang", Keahlian = "Memasak", Agama = "Islam", jenis pekerjaan = "Pembantu", foto ktp = "file foto ktp" foto diri "file foto diri"</li> <li>4. Menekan tombol tambah pada popup.</li> </ol> |
| Hasil yang Diharapkan | Sistem mendaftarkan akun pekerja baru sesuai dengan data pada field form.   |
| Hasil yang Didapatkan | Sistem mendaftarkan akun pekerja baru sesuai dengan data pada field form.   |



|        |       |
|--------|-------|
| Status | Valid |
|--------|-------|

Hasil dari pengujian validasi *use case* tambah pekerja alternatif 1 membuahkan nilai valid. Tabel 6.50 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.50 Pengujian Validasi Use Case Tambah Pekerja Alternatif 1**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_026   |
| Nama Fungsi           | Tambah Pekerja  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar pekerja.</li> <li>2. Menekan tombol tambah pekerja.</li> <li>3. Tidak mengisi data pada form.</li> <li>4. Menekan tombol tambah pada popup.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Silahkan isi field".  |
| Hasil yang Didapatkan | <p>Sistem menampilkan pesan "Silahkan isi field".</p>    |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* melihat riwayat pekerjaan pekerja membuahkan nilai valid. Tabel 6.51 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.51 Pengujian Validasi Use Case Melihat Riwayat Pekerjaan Pekerja**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_027  |
| Nama Fungsi           | Melihat Riwayat Pekerjaan Pekerja  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar pekerja.</li> <li>2. Menekan tombol transaksi pada pekerja terpilih.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan data transaksi pekerja  |



|                       |   |
|-----------------------|---|
|                       | terpilih.   |
| Hasil yang Didapatkan | Sistem menampilkan data transaksi pekerja terpilih. |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* melihat riwayat transaksi majikan membuahkan nilai valid. Tabel 6.52 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.52 Pengujian Validasi Use Case Melihat Riwayat Transaksi Majikan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_E_028  |
| Nama Fungsi           | Melihat Riwayat Transaksi Majikan  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman daftar majikan.</li> <li>2. Menekan tombol transaksi pada majikan terpilih.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan data transaksi majikan terpilih.  |
| Hasil yang Didapatkan | Sistem menampilkan data transaksi majikan terpilih.  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* konfirmasi pendaftaran pekerja membuahkan nilai valid. Tabel 6.53 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.


**Tabel 6.53 Pengujian Validasi Use Case Konfirmasi Pendaftaran Pekerja**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_029  |
| Nama Fungsi           | Konfirmasi Pendaftaran Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman pendaftaran pekerja.</li> <li>2. Memilih pekerja yang ingin di konfirmasi.</li> <li>3. Menekan tombol terima.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data penerimaan pekerja.  |
| Hasil yang Didapatkan | Sistem menyimpan data penerimaan pekerja.  |
| Status                | Valid  |



Hasil dari pengujian validasi *use case* konfirmasi pendaftaran pekerja alternatif 1 membuahkan nilai valid. Tabel 6.54 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.54 Pengujian Validasi Use Case Konfirmasi Pendaftaran Pekerja Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_029  |
| Nama Fungsi           | Konfirmasi Pendaftaran Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman pendaftaran pekerja.</li> <li>2. Memilih pekerja yang ingin di konfirmasi.</li> <li>3. Menekan tombol terima.</li> </ol> |
| Hasil yang Diharapkan | Sistem menampilkan pesan "Terjadi Kesalahan"   |
| Hasil yang Didapatkan | <p>Sistem menampilkan pesan "Terjadi Kesalahan".</p>    |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* konfirmasi pendaftaran pekerja alternatif 2 membuahkan nilai valid. Tabel 6.55 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.55 Pengujian Validasi Use Case Konfirmasi Pendaftaran Pekerja Alternatif 2**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_029   |
| Nama Fungsi           | Konfirmasi Pendaftaran Pekerja  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman pendaftaran pekerja.</li> <li>2. Memilih pekerja yang ingin di konfirmasi.</li> <li>3. Menekan tombol tolak.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data penolakan pekerja dan menghapus data pekerja yang mendaftar.  |
| Hasil yang Didapatkan | Sistem menyimpan data penolakan pekerja dan menghapus data pekerja yang mendaftar.  |



|        |       |
|--------|-------|
| Status | Valid |
|--------|-------|

Hasil dari pengujian validasi *use case* melihat riwayat pemesanan membuahkan nilai valid. Tabel 6.56 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.56 Pengujian Validasi Use Case Melihat Riwayat Pemesanan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_030                                  |
| Nama Fungsi           | Melihat Riwayat Pemesanan                  |
| Prosedur Pengujian    | 1. Membuka halaman riwayat pemesanan.      |
| Hasil yang Diharapkan | Sistem menampilkan data riwayat pemesanan. |
| Hasil yang Didapatkan | Sistem menampilkan data riwayat pemesanan. |
| Status                | Valid                                      |

Hasil dari pengujian validasi *use case* penggajian pekerja membuahkan nilai valid. Tabel 6.57 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.57 Pengujian Validasi Use Case Penggajian Pekerja**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_031   |
| Nama Fungsi           | Penggajian Pekerja  |
| Prosedur Pengujian    | 1. Membuka halaman penggajian pekerja.<br>2. Memilih pekerja yang akan digaji.<br>3. Menekan tombol gaji. |
| Hasil yang Diharapkan | Sistem menyimpan data penggajian.   |
| Hasil yang Didapatkan | Sistem menyimpan data penggajian.   |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* melihat riwayat penggajian pekerja membuahkan nilai valid. Tabel 6.58 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.58 Pengujian Validasi Use Case Melihat Riwayat Penggajian Pekerja**

|                |                                    |
|----------------|------------------------------------|
| Kode Kebutuhan | PRT_F_032                          |
| Nama Fungsi    | Melihat Riwayat Penggajian Pekerja |



|                       |   |
|-----------------------|---|
| Prosedur Pengujian    | 1. Membuka halaman riwayat penggajian.              |
| Hasil yang Diharapkan | Sistem menampilkan data riwayat penggajian pekerja. |
| Hasil yang Didapatkan | Sistem menampilkan data riwayat penggajian pekerja. |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* konfirmasi claim garansi membuah nilai valid. Tabel 6.59 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.59 Pengujian Validasi Use Case Konfirmasi Claim Garansi**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_033   |
| Nama Fungsi           | Konfirmasi Claim Garansi  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman claim garansi.</li> <li>2. Memilih claim yang akan di konfirmasi.</li> <li>3. Menekan tombol terima.</li> <li>4. Memilih pekerja yang akan menggantikan.</li> <li>5. Menekan tombol pilih.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data pemesanan baru dan menonaktifkan pesanan lama.  |
| Hasil yang Didapatkan | Sistem menyimpan data pemesanan baru dan menonaktifkan pesanan lama.  |
| Status                | Valid   |

Hasil dari pengujian validasi *use case* konfirmasi claim garansi alternatif 1 membuah nilai valid. Tabel 6.60 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.60 Pengujian Validasi Use Case Konfirmasi Claim Garansi Alternatif 1**

|                    |  |
|--------------------|--|
| Kode Kebutuhan     | PRT_F_033  |
| Nama Fungsi        | Konfirmasi Claim Garansi   |
| Prosedur Pengujian | <ol style="list-style-type: none"> <li>1. Membuka halaman claim garansi.</li> <li>2. Memilih claim yang akan di konfirmasi.</li> <li>3. Menekan tombol tolak.</li> </ol> |





|                       |  |
|-----------------------|--|
| Hasil yang Diharapkan | Sistem menyimpan data claim garansi dengan status ditolak. |
| Hasil yang Didapatkan | Sistem menyimpan data claim garansi dengan status ditolak. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* melihat riwayat claim garansi membuah nilai valid. Tabel 6.61 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.61 Pengujian Validasi Use Case Melihat Riwayat Claim Garansi**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_034                                      |
| Nama Fungsi           | Melihat Riwayat Claim Garansi                  |
| Prosedur Pengujian    | 1. Membuka halaman riwayat claim garansi.      |
| Hasil yang Diharapkan | Sistem menampilkan data riwayat claim garansi. |
| Hasil yang Didapatkan | Sistem menampilkan data riwayat claim garansi. |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* konfirmasi resign pekerja membuah nilai valid. Tabel 6.62 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.62 Pengujian Validasi Use Case Konfirmasi Resign Pekerja**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_035   |
| Nama Fungsi           | Konfirmasi Resign Pekerja   |
| Prosedur Pengujian    | 1. Aktor membuka halaman resign pekerja<br>2. Memilih data resign pembantu1 yang akan di konfirmasi.<br>3. Menekan tombol konfirmasi. |
| Hasil yang Diharapkan | Sistem menyimpan data resign pembantu1 dan mengubah status pembantu1 menjadi "resign"   |
| Hasil yang Didapatkan | Sistem menyimpan data resign pembantu1 dan mengubah status pembantu1 menjadi  |



|        |          |
|--------|----------|
|        | “resign” |
| Status | Valid    |

Hasil dari pengujian validasi *use case* resign pekerja alternatif 1 membuahkan nilai valid. Tabel 6.63 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.63 Pengujian Validasi Use Case Konfirmasi Resign Pekerja Alternatif 1**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_035  |
| Nama Fungsi           | Konfirmasi Resign Pekerja  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman resign pekerja</li> <li>2. Memilih data resign pembantu1 yang akan di konfirmasi.</li> <li>3. Menekan tombol tolak.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data resign pembantu1 dan mengubah status pembantu1 menjadi “tersedia”  |
| Hasil yang Didapatkan | Sistem menyimpan data resign pembantu1 dan mengubah status pembantu1 menjadi “tersedia”  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* blacklist pekerja membuahkan nilai valid. Tabel 6.64 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.64 Pengujian Validasi Use Case Blacklist Pekerja**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_036   |
| Nama Fungsi           | Blacklist Pekerja   |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman blacklist pekerja</li> <li>2. Memilih data pembantu2 yang akan di blacklist.</li> <li>3. Menekan tombol blacklist.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data blacklist pembantu2 dan mengubah status pembantu2 menjadi “blacklist”   |
| Hasil yang Didapatkan | Sistem menyimpan data blacklist pembantu2   |



|        |   |
|--------|---|
|        | dan mengubah status pembantu2 menjadi "blacklist" |
| Status | Valid   |

Hasil dari pengujian validasi *use case* blacklist majikan membuahkn nilai valid. Tabel 6.65 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.65 Pengujian Validasi Use Case Blacklist Majikan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_037  |
| Nama Fungsi           | Blacklist Majikan  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Membuka halaman blacklist majikan</li> <li>2. Memilih data majikan1 yang akan di blacklist.</li> <li>3. Menekan tombol blacklist.</li> </ol> |
| Hasil yang Diharapkan | Sistem menyimpan data blacklist majikan1 dan mengubah status majikan1 menjadi "blacklist"  |
| Hasil yang Didapatkan | Sistem menyimpan data blacklist majikan1 dan mengubah status majikan1 menjadi "blacklist"  |
| Status                | Valid  |

Hasil dari pengujian validasi *use case* mengubah status pemesanan membuahkn nilai valid. Tabel 6.66 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.66 Pengujian Validasi Use Case Mengubah Status Pemesanan**

|                       |  |
|-----------------------|--|
| Kode Kebutuhan        | PRT_F_038  |
| Nama Fungsi           | Mengubah Status Pemesanan  |
| Prosedur Pengujian    | <ol style="list-style-type: none"> <li>1. Mengakses <i>end point</i> API ubah status pemesanan.</li> </ol> |
| Hasil yang Diharapkan | Sistem melakukan perubahan terhadap status pemesanan berdasarkan data yang diberikan oleh aktor Midtrans.  |
| Hasil yang Didapatkan | Sistem melakukan perubahan terhadap status pemesanan berdasarkan data yang diberikan oleh aktor Midtrans.  |



|        |       |
|--------|-------|
| Status | Valid |
|--------|-------|

Hasil dari pengujian validasi *use case Logout* membuahkan nilai valid. Tabel 6.67 akan menjelaskan secara lebih rinci terkait hasil dari pengujian validasi *use case* ini.

**Tabel 6.67 Pengujian Validasi Use Case Logout**

|                       |   |
|-----------------------|---|
| Kode Kebutuhan        | PRT_F_039   |
| Nama Fungsi           | <i>Logout</i>   |
| Prosedur Pengujian    | 1. Menekan tombol <i>Logout</i> .   |
| Hasil yang Diharapkan | Sistem mengeluarkan aktor yang telah <i>Login</i> kedalam sistem dan menjadi <i>guest</i> . |
| Hasil yang Didapatkan | Sistem mengeluarkan aktor yang telah <i>Login</i> kedalam sistem dan menjadi <i>guest</i> . |
| Status                | Valid   |

#### 6.4 Pengujian Kompatibilitas

Tahapan ini adalah tahapan yang menguji kompatibilitas dari sistem yang dibuat. Tahapan ini akan menguji kebutuhan non fungsional yang telah dijabarkan sebelumnya. Kebutuhan yang dimaksud adalah kebutuhan dengan kode PRT\_NF\_001. Tabel 6.68 akan menjelaskan pengujian kompatibilitas secara rinci.

**Tabel 6.68 Pengujian Kompatibilitas**

|                    |  |
|--------------------|--|
| Kode Kebutuhan     | PRT_NF_001   |
| Nama Kasus Uji     | Sistem yang dibuat harus memastikan dapat dijalankan di berbagai perambah yang berbeda seperti perambah Safari, perambah firefox, Google Chrome, perambah edge, perambah pada perangkat android, dan perambah pada perangkat iOS |
| Prosedur           | 1. Menjalankan sistem di beberapa <i>browser</i> yang berbeda seperti Firefox, Google Chrome, Safari, Edge, perangkat Android, dan perangkat iOS.  |
| Hasil (Diharapkan) | Menampilkan sistem dengan baik di <i>browser</i> seperti Firefox, Google Chrome, Safari, Edge, perangkat Android, dan perangkat iOS.   |



|                    |  |
|--------------------|--|
| Hasil (Didapatkan) | Menampilkan sistem dengan baik di <i>browser</i> seperti Firefox, Google Chrome, Safari, Edge, perangkat Android, dan perangkat iOS. |
| Status             | Valid  |

Pengujian kompatibilitas dilakukan dengan bantuan aplikasi SortSite. Aplikasi ini akan menjalankan sistem di berbagai perambah. Perambah-perambah tersebut adalah Firefox, Google Chrome, Safari, Edge, perangkat Android, dan perangkat iOS. Hasil dari pengujian tersebut mendapatkan hasil nilai valid.

Pada *sortsite*, terdapat 3 kategori permasalahan pada pengujian kompatibilitas, yaitu :

- *Critical issues* yang menyebabkan terjadinya kehilangan konten dan fungsi tidak berjalan sebagaimana mestinya.
- *Major issues* yang merupakan kesalahan pada tata letak maupun kesalahan yang mempengaruhi kinerja browser.
- *Minor issues* yang merupakan kesalahan minor atau kesalahan yang mempengaruhi kinerja browser

Gambar 6.8 akan menampilkan tangkapan layar dari hasil pengujian yang dilakukan menggunakan SortSite.

The screenshot shows the SortSite interface with the following data:

| Browser         | IE | Edge | Firefox | Safari | Opera | Chrome | iOS  |    |    | Android |    |
|-----------------|----|------|---------|--------|-------|--------|------|----|----|---------|----|
| Version         | 11 | 81   | 76      | 13     | 68    | 81     | ≤ 11 | 12 | 13 | ≤ 3     | 4* |
| Critical Issues | ✓  | ✓    | ✓       | ✓      | ✓     | ✓      | ✓    | ✓  | ✓  | ✓       | ✓  |
| Major Issues    | ⚠  | ✓    | ⚠       | ⚠      | ⚠     | ✓      | ⚠    | ⚠  | ⚠  | ⚠       | ⚠  |
| Minor Issues    | ⚠  | ⚠    | ⚠       | ⚠      | ⚠     | ⚠      |      |    |    | ⚠       | ⚠  |

Key:  
 ● Missing content or functionality  
 ● Major layout or performance problems  
 ● Minor layout or performance problems

\* Most Android devices from 4.4 onwards use Chrome as the default browser, older versions use the original Android stock browser

**Gambar 6.8 Hasil Pengujian Kompatibilitas**

Gambar 6.8 menunjukkan hasil pengujian kompatibilitas yang telah dilakukan. Pengujian ini menyatakan bahwa sistem terbebas dari *critical issues*. Terdapat beberapa *major* dan *minor issue* pada sistem, tetapi tidak mempengaruhi jalannya fungsi dan fungsi masih bisa berjalan sebagaimana mestinya.



## BAB 7 PENUTUP

### 7.1 Kesimpulan

Berlandaskan hasil yang didapat mulai dari tahap awal hingga tahap akhir penelitian ini, diperoleh lah beberapa kesimpulan diantaranya adalah:

1. Pada tahapan analisis kebutuhan yang sebelumnya telah dilakukan, sistem ini memiliki tiga puluh sembilan kebutuhan fungsional dan juga satu kebutuhan non-fungsional yang membantu CV. Sitiari dalam mengelola daftar pekerja, daftar majikan, pemesanan, pembayaran, dan juga claim garansi yang terjadi di lingkungan internal CV. Sitiari. Proses wawancara dan juga studi literatur telah dilakukan sehingga mendapatkan permasalahan utama diantaranya, permasalahan dari segi pencatatan pekerja dan majikan, permasalahan dari segi manajemen transaksi, dan permasalahan dari segi efisiensi pelanggan dalam memesan pekerja. Selanjutnya pada sisi pembagian pengguna didapatkan 5 aktor yang dapat berinteraksi dengan sistem diantaranya Pekerja, Majikan, Midtrans, Guest, dan Admin.
2. Pada tahapan perancangan, sistem ini menghasilkan beberapa poin perancangan, diantaranya adalah perancangan arsitektur yang nantinya akan menghasilkan *sequence diagram* yang digunakan untuk melihat interaksi antar method dalam klas dan juga *class diagram*. Poin berikutnya yang dihasilkan adalah perncangan kmponen, yang isinya adalah algoritme yang digunakan di dalam sistem. Selanjutnya adalah perncangan basis data dengan hasil berupa *Physical Data Model* (PDM). Dan yang terakhir adalah perancangan antarmuka ang berisi layout tampilan.
3. Pada tahapan implementasi, Sistem ini menghasilkan implementasi kode program yang mengacu pada rancangan algoritme, implementasi dari basis data, dan juga implementasi antarmuka.
4. Pada tahapan pengujian sistem, diperoleh hasil dari satu pengujian integrasi, tiga buah pengujian unit, dan tiga puluh sembilan pengujian validasi dengan nilai 100% valid. Untuk pengujian kompatibilitas, sistem menghasilkan phasil pengujian dan ditemukan bahwa sistem ini dapat berjalan di delapan jenis *browser* diantaranya Firefox, Google Chrome, Safari, Edge, perangkat Android, dan perangkat iOS.

### 7.2 Saran

Pengembangan lebih lanjut sistem ini menghasilkan saran yang perlu diberikan, saran tersebut antara lain adalah

1. Pada sistem ini, proses blacklist pekerja masih dilakukan secara manual yang menyebabkan pihak admin harus selalu melakukan pengecekan track record pada pekerja. Oleh karena itu, pada penelitian selanjutnya bisa dibuat sistem blacklist otomatis dengan memanfaatkan kecerdasan buatan yang nantinya bisa membantu admin dalam melakukan blacklist.

**DAFTAR REFERENSI**

- Alam, S. S., Khatibi, A., Ahmad, M. I., & Ismail, H. B. (2007). *Factors Affecting E-Commerce Adoption in the Electronic Manufacturing Companies in Malaysia*, 17(1), 125-139.
- Caytiles, R. D., & Sunguk, L. (2014). *A Review of an MVC Framework based Software Development*, VIII(10), 213-220.
- Clark, D. (2011). *Beginning C# Object-Oriented Programming*. Canada: Apress.
- International Labour Organization, 2017. *Pekerja Rumah Tangga Di Indonesia*.
- McCool, S. (2012). *Laravel Starter*. Birmingham: Packt Publishing Ltd.
- Midtrans. (2015). *Yuk Mengenal Payment Gateway dan Berbagai Keuntungannya*. Dipetik March 17, 2020, dari <https://blog.midtrans.com/yuk-mengenal-payment-gateway-dan-keuntungan-bila-menggunakannya/>
- Nugroho, A. (2009). *Rekayasa Perangkat Lunak Menggunakan UML dan Java*. Yogyakarta: Penerbit ANDI.
- Perda DKI No. 6 tahun 2004, pasal 1 ayat (17)
- PowerMapper. (2019). *One Click Website Testing*. Dipetik March 20, 2020, dari <https://www.powermapper.com/products/sortsite/>
- Pressman, R. S. (2010). *Software Quality Engineering: A Practitioner's Approach* (7th ed.). New York: McGraw-Hill.
- Singarimbun, M., & Effendi, S. (1991). *Metode Penelitian Survei*. Jakarta: LP3S.
- Solichin, A. (2010). *MySQL Dari Pemula Hingga Mahir*. Jakarta: Universitas Budi Luhur.
- Sommerville, I. (2011). *Software Engineering* (9th ed.). London: Addison-Wesley.
- Wahy, H. (2012). *Keluarga Sebagai Basis Pendidikan Pertama dan Utama. Keluarga Sebagai Basis Pendidikan Pertama dan Utama*, XII(2), 245-258.
- Whitten, J. L., & Bentley, L. D. (2007). *System Analysis & Design Methods* (7th ed.). New York: McGraw-Hill.
- Yoon, I., Sussman, A., Memon, A., & Porter, A. (2008). *Effective and Scalable Software Compatibility Testing*, 63-74.

**LAMPIRAN A HASIL WAWANCARA**

Narasumber : Bu Priskila

Jabatan : Pengelola CV. Sitiari

| No | Pertanyaan   | Jawaban  |
|----|--|--|
| 1  | Jasa apa sajakah yang disediakan oleh CV. Sitiari?   | Pekerja rumah tangga, babysitter, dan perawat lansia.  |
| 2  | Bagaimana alur pemesanan pekerja?  | Majikan datang atau menghubungi pihak yayasan, lalu yayasan akan memberikan list pekerja yang tersedia, lalu majikan menyerahkan persyaratan yang diperlukan dan membayar biaya pekerja, lalu pekerja akan datang kerumah. |
| 3  | Apakah ada fasilitas yang ditawarkan oleh CV. Sitiari?   | Ada, yaitu fasilitas garansi dengan waktu 1 bulan.   |
| 4  | Bagaimana pencatatan transaksi yang dilakukan sebelumnya?  | Pencatatan dilakukan dengan menggunakan buku dan juga microsoft excel.   |
| 5  | Apakah ada kendala dalam pencatatan?   | Ya, karena kami memiliki 2 cabang maka kami kadang kesulitan untuk mensinkronisasi data pada 2 cabang tersebut.  |
| 6  | Apakah ada kendala lain?   | Ada, kendala pembayaran yang terkadang keliru dan tidak tercatat.  |
| 7  | Menurut ibu, apakah perlu dibuat sistem untuk manajemen pekerja dan majikan juga mempermudah pembayaran dan pencatatan pembayaran? | Ya, Perlu  |
| 8  | Jika nantinya dibuatkan sistem untuk CV. Sitiari, apakah ada saran dalam pembuatan sistem?   | Ya, sistem tidak hanya bisa diakses di 1 komputer cabang, melainkan dapat diakses dari seluruh cabang. Sistem juga harus bisa diakses lewat hp, jadi kami bisa akses dirumah.  |
| 9  | Bagaimana prosedur untuk pekerja resign?   | Pekerja hanya harus menyelesaikan pekerjaan dan menghubungi pihak yayasan.   |
| 10 | Untuk penggajian pekerja, apakah ada potongan?   | Gaji pekerja dikenakan potongan administrasi.  |
| 11 | Apakah ada kendala saat penggajian pekerja?  | Ya, terkadang tidak tercatat.  |

Malang, 26 September 2019

( PRISKILA MANIKBY )





Narasumber : Bu Tyas

Jabatan : Pelanggan CV. Sitiari

| No | Pertanyaan   | Jawaban  |
|----|--|--|
| 1  | Sudah berapa lama menjadi pelanggan CV. Sitiari?         | 3 Tahun.   |
| 2  | Bagaimana alur pemesanan pekerja?                        | Di awal saya datang dan melakukan pemesanan, lalu membayar secara cash, lalu nanti pekerja akan datang kerumah.                          |
| 3  | Apakah ada fasilitas yang ditawarkan oleh CV. Sitiari?   | Ada, masa percobaan (garansi) selama 1 bulan   |
| 4  | Apakah ibu pernah melakukan pemesanan secara online?     | Ya, Pernah   |
| 5  | Apakah ada kendala pada saat pemesanan?                  | Untuk kendala yang besar tidak ada, hanya saja terkadang jika menghubungi lewat WA untuk menanyakan ketersediaan pekerja responnya lama. |
| 6  | Apakah diperlukan adanya sistem pemesanan secara online? | Ya, tetapi harus ada metode pembayaran dengan transfer karena selama ini pembayaran dilakukan dengan transfer dan cash                   |

Malang, 26 September 2019

B. TYAS CV Sitiari



Narasumber : Bu Arni

Jabatan : Pekerja CV. Sitiari

| No | Pertanyaan   | Jawaban   |
|----|--|---|
| 1  | Sudah berapa lama menjadi pekerja di CV. Sitiari?        | 2 Tahun   |
| 2  | Bagaimana alur pendaftaran pekerja?                      | Melakukan pendaftaran dengan syarat foto ktp dan juga data diri, jika ada menyerahkan surat rekomendasi. Selanjutnya akan mengikuti proses pelatihan.   |
| 3  | Apakah ada fasilitas yang ditawarkan oleh CV. Sitiari?   | Ada, masa pelatihan.  |
| 4  | Bagaimana proses pemesanan pekerja?                      | Melakukan pemesanan seperti biasa, lalu setelah pembayaran dilakukan saya akan langsung datang kerumah.   |
| 5  | Bagaimana proses penggajian?                             | Perbulannya CV. Sitiari mengirimkan gaji langsung ke rekening, atau jika saya ingin cash bisa mengajukan secara cash dengan mengambil ke kantor cabang. |
| 6  | Apakah ada kendala pada saat pemesanan?                  | Terkadang ada kesalahan dari admin di pencatatan yang mengakibatkan saya salah alamat.  |
| 7  | Apakah diperlukan adanya sistem pemesanan secara online? | Ya, tetapi harus mencatat semua hal yang menyangkut pemesanan, sehingga meminimalisir kesalahan alamat dan lain - lain.                                 |

Malang, 26 September 2019

  
 ( ARNI )



## LAMPIRAN B HASIL VALIDASI KEBUTUHAN

| No | Kebutuhan   | Checklist |
|----|---|-----------|
| 1  | Masuk kedalam sistem menggunakan email dan password.  | √         |
| 2  | Mendaftar sebagai pekerja dengan menginputkan : email, password, nama, nomor KTP, tanggal lahir, nomor telepon, alamat, foto diri dan foto ktp. | √         |
| 3  | Mendaftar sebagai majikan dengan menginputkan : email, password, nama, nomor KTP, nomor telepon, alamat, dan foto ktp.                          | √         |
| 4  | Melihat daftar admin yang terdaftar.  | √         |
| 5  | Melihat daftar pekerja dengan status tersedia, pending, blacklist, maupun bekerja.  | √         |
| 6  | Mengajukan permohonan resign bagi pekerja.  | √         |
| 7  | Melihat daftar riwayat pekerjaan dan penggajian pekerja.  | √         |
| 8  | Melihat daftar riwayat pemesanan, pembayaran dan claim garansi majikan.   | √         |
| 9  | Melakukan pemesanan secara online.  | √         |
| 10 | Melakukan claim garansi bagi majikan.   | √         |
| 11 | Memperpanjang masa kerja PRT bagi majikan.  | √         |
| 12 | Memberikan rating kepada pekerja.   | √         |
| 13 | Memberikan rating kepada majikan  | x         |
| 14 | Melakukan edit pada daftar admin, majikan, dan pekerja.   | √         |



|    |  |   |
|----|--|---|
| 15 | Melakukan penghapusan pada daftar admin, majikan, dan pekerja. | √ |
| 16 | Melakukan penambahan pada daftar admin, majikan, dan pekerja.  | √ |
| 17 | Melakukan konfirmasi pendaftaran pekerja.                      | √ |
| 18 | Melakukan pencatatan penggajian pekerja.                       | √ |
| 19 | Melakukan konfirmasi pada claim garansi.                       | √ |
| 20 | Melakukan konfirmasi pada pekerja yang ingin resign.           | √ |
| 21 | Melakukan blacklist pada pekerja maupun majikan.               | √ |
| 22 | Keluar dari sistem (logout).                                   | √ |