

**PEMBANGUNAN APLIKASI PEMESANAN SAYUR KELILING DENGAN
MENERAPKAN *RATIONAL PERFORMANCE TESTER* SEBAGAI KAKAS
BANTU DALAM PENGUJIAN PERANGKAT LUNAK**

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:

Fandhi Ahmad Maulana

NIM: 115060807111009



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

PEMBANGUNAN APLIKASI PEMESANAN SAYUR KELILING DENGAN MENERAPKAN
RATIONAL PERFORMANCE TESTER SEBAGAI KAKAS BANTU DALAM PENGUJIAN
PERANGKAT LUNAK

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh:
Fandhi Ahmad Maulana
NIM: 115060807111009

Skripsi ini telah diuji dan dinyatakan lulus pada
21 Juli 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Denny Sagita R., S.Kom., M.Kom
NIP. 1985 1124 201504 1 001

Fajar Pradana, S.ST., M.Eng
NIP. 1987 1121 201504 1 004

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP.19710518 200312 1 001

PERNYATAAN ORISINALITAS SKRIPSI

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 unsur-unsur 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, Juni 2016



Fandhi Ahmad Maulana
NIM: 115060807111009

KATA PENGANTAR

Puji Syukur penulis panjatkan kepada Tuhan Yang Maha Esa karena hanya dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan skripsi dengan judul “Pembangunan Aplikasi Pemesanan Sayur Keliling dengan Menerapkan Rational Performance Tester sebagai Kakas Bantu dalam Pengujian Perangkat Lunak”. Melalui kesempatan ini, penulis ingin menyampaikan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan batuan dan dukungan selama penulisan skripsi ini, diantaranya:

1. Bapak Denny Sagita Rusdianto, S.Kom, M.Kom selaku dosen pembimbing I yang telah memberikan masukan dan ilmu serta saran dalam menyelesaikan skripsi ini.
2. Bapak Fajar Pradana, S.ST, M.Eng selaku dosen pembimbing II yang juga memberikan masukan dan ilmu serta saran dalam menyelesaikan skripsi ini.
3. Kepada kedua orang tua yaitu Bapak ir.M.Siswanto, S.H, M.M dan Ibu Tiwi Pntjowati S.H beserta keluarga besar atas segala doa, nasehat dan dukungan yang diberikan hingga terselesainya skripsi ini.
4. Staff dosen Fakultas Ilmu Komputer Universitas Brawijaya yang telah membekali penulis berbagai ilmu selama mengikuti perkuliahan sampai akhir penulisan skripsi.
5. Bapak Sabriansyah Rizqika Akbar, S.T., M.Eng selaku dosen pembimbing akademik yang telah memberikan pengarahan selama penulis menempuh pendidikan di Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya.
6. Teman-teman kelas D Informatika 2011 yang selalu memberi semangat dan membantu dalam pengerjaan skripsi ini.
7. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat secara langsung atau tidak langsung agar terselesaikannya skripsi ini.

Penulis menyadari bahwa skripsi ini masih banyak kekurangan. Oleh karena itu, kritik dan saran yang bersifat membangun sangat diharapkan untuk menyempurnakan skripsi ini. Semoga skripsi ini membawa manfaat bagi pihak lain yang menggunakannya.

Malang, Juni 2016

Fandhi Ahmad Maulana

ahmadfandhi39@gmail.com

ABSTRACT

Itinerant greengrocer “penjual sayur keliling” is one of alternative for Indonesian society who don’t have enough time to go to the groceries store to fulfill their daily need. Unfortunately, sometimes there were buyers who didn’t got their groceries though the seller has been tried to provide their groceries. But when the itinerant greengrocer overstate foodstuffs, sometimes leftovers and stale. To overcome these problems, we need a system that can bridge the gap between buyers and sellers.

The system will built web-based mobile applications using web based prototype model to facilitate user needs relate to these problem. Test or control quality is required to create a quality software product and ensure the software runs well. Tests on web-based applications is made to find errors of functionality, usability, navigation, performance, capacity and safety of the software. Performance testing is used to determine responsiveness, throughput, reliability and scalability when the system is used. Rational Performance Tester is a testing tool developed by IBM that have advantages on perform scheduling testing, display the test results in real time and eclipse based.

To require the criteria of performance testing, each called page should take time not more than 8 to 10 seconds each, if the response exceeds 10 seconds it will interfere the user’s activity and could degrade application reputation productivity, sales and hurt the company name. based on this testing, the average respond time of every page is not more than 10,000ms (1,000ms equal to 1s). Also the result of fungsional system testing has filled the functional requirements with a 100% validation value.

Keyword: Itinerant Greengrocer, Performance Test, Rational PerformanceTtester

ABSTRAK

Penjual sayur keliling merupakan salah satu alternatif pada masyarakat yang tidak memiliki waktu untuk pergi ke pasar, tetapi terkadang masih ada pembeli yang tidak mendapat bahan makanan yang diinginkan karena sudah habis terlebih dahulu meskipun para penjual sayur ini sudah berusaha menyediakan bahan makanan yang lengkap. Namun ketika penjual sayur melebihi bahan makanan, terkadang tidak habis dan jadi busuk. Untuk mengatasi permasalahan tersebut, maka dibutuhkan sebuah sistem yang dapat menjembatani antara pembeli dan penjual sayur keliling.

Sistem yang akan dibangun berbasis *web* dan menjadikan aplikasi *mobile* berbasis *web* dengan menggunakan model *prototype* untuk memudahkan memahami kebutuhan pengguna. Untuk membuat perangkat lunak yang berkualitas diperlukan pengujian untuk memastikan perangkat lunak berjalan dengan baik. Pengujian pada aplikasi berbasis *web* dilakukan untuk menemukan kesalahan pada fungsi, usabilitas, navigasi, performa, kapasitas dan keamanan. *Performance testing* digunakan untuk menentukan *responsiveness*, *throughput*, *reliability* dan *scalability* pada saat sistem digunakan. Rational Performance Tester merupakan salah satu kakas bantu pengujian yang dikembangkan oleh IBM yang memiliki kelebihan yaitu dapat melakukan penjadwalan pengujian, dapat merekam pengujian dan menampilkan hasil pengujian secara realtime serta berbasis eclipse.

Untuk memenuhi kriteria dalam pengujian performa setiap pemanggilan halaman pada aplikasi harus tidak lebih dari 8 sampai 10 detik, karena jika respon melebihi 10 detik maka dapat mengganggu aktivitas pengguna yang dapat menurunkan reputasi, produktivitas, penjualan dan merugikan perusahaan. Dari hasil pengujian yang didapatkan, setiap halaman aplikasi rata-rata menghasilkan respon waktu tidak lebih dari 7,900ms(1,000ms sama dengan 1s). Sedangkan dari hasil pengujian fungsional sistem juga telah memenuhi kebutuhan fungsional dengan nilai 100% valid.

Kata Kunci: Penjual sayur keliling, Performance Test, Rational Performance Tester.

DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS SKRIPSI	iii
KATA PENGANTAR.....	iv
ABSTRACT.....	v
ABSTRAK.....	vi
DAFTAR ISI.....	vii
DAFTAR TABEL.....	x
DAFTAR GAMBAR.....	xiii
BAB 1 PENDAHULUAN.....	1
1.1 Latar belakang.....	1
1.2 Rumusan masalah.....	2
1.3 Tujuan	2
1.4 Manfaat.....	2
1.5 Batasan masalah	2
1.6 Sistematika pembahasan.....	3
BAB 2 LANDASAN KEPUSTAKAAN	4
2.1 Kajian Pustaka	4
2.2 Rekayasa Perangkat Lunak	5
2.3 Penjual Sayur Keliling.....	5
2.4 Pemesanan.....	6
2.5 <i>Software Development Life Cycle (SDLC)</i>	6
2.5.1 <i>Prototype</i>	6
2.6 <i>Codeigniter</i>	9
2.7 Pengujian Perangkat Lunak.....	10
2.7.1 <i>Performance test</i>	10
2.7.2 <i>Black-Box Testing</i>	11
2.7.3 <i>White-Box Testing</i>	12
2.8 <i>Rational Performance tester</i>	12
BAB 3 METODOLOGI	13

3.1 Studi Literatur	14
3.2 Analisa Kebutuhan	14
3.3 Pengumpulan Data	14
3.4 Perancangan Sistem.....	15
3.5 Prototype	15
3.6 Implementasi	15
3.7 Pengujian dan Analisis	16
3.8 Evaluasi Sistem.....	16
3.9 Pengambilan Kesimpulan dan Saran	16
BAB 4 ANALISIS DAN PERANCANGAN SISTEM	17
4.1 <i>Analisa Kebutuhan</i>	17
4.1.1 Gambaran Umum.....	18
4.1.2 Identifikasi Aktor	19
4.1.3 Analisis Kebutuhan <i>Fungsional</i>	19
4.1.4 Analisis Kebutuhan <i>Non-Fungsional</i>	36
4.2 <i>Perancangan Aplikasi</i>	36
4.2.1 <i>Activity Diagram</i>	36
4.2.2 <i>Sequence Diagram</i>	45
4.2.3 <i>Class Diagram</i>	57
4.3 Perancangan Basis Data.....	59
4.4 Perancangan Antarmuka	59
4.5 Skenario Pengujian	69
BAB 5 IMPLEMENTASI	70
5.1 <i>Spesifikasi Sistem</i>	71
5.1.1 Spesifikasi Perangkat Keras.....	71
5.1.2 Spesifikasi Perangkat Lunak	71
5.2 <i>Batasan Implementasi</i>	71
5.3 <i>Implementasi Class</i>	72
5.4 <i>Implementasi Basis Data</i>	72
5.5 <i>Implementasi Antarmuka</i>	78
BAB 6 PENGUJIAN DAN ANALISA	88
6.1 <i>Pengujian Fungsional</i>	88

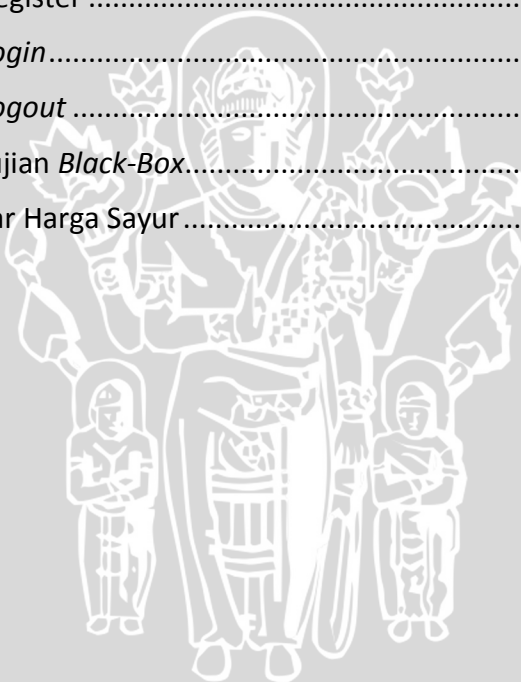
6.1.1 Pengujian <i>White-Box</i>	88
6.1.2 Pengujian <i>Black-Box</i>	88
6.2 Performance test	88
6.3 Hasil Pengujian Fungsional	96
6.3.1 Hasil Pengujian <i>White-Box</i>	96
6.3.2 Hasil Pengujian <i>Black-Box</i>	105
6.4 Performance test	116
6.4.1 Hasil Pengujian Performance Testing	116
6.5 Analisa Hasil Pengujian	167
6.5.1 Analisa pengujian <i>White-Box</i>	167
6.5.2 Analisa hasil pengujian <i>Black-Box</i>	168
6.5.3 Analisa hasil pengujian performa.....	168
6.6 EVALUASI SISTEM.....	170
BAB 7 KESIMPULAN DAN SARAN	171
7.1 Kesimpulan.....	171
9.2Saran	172
DAFTAR PUSTAKA.....	173
Lampiran dan Hasil Wawancara.....	175

DAFTAR TABEL

Tabel 4.1 Identifikasi Aktor	19
Tabel 4. 2 Kebutuhan <i>Fungsional</i> Pedagang	20
Tabel 4.3 Skenario <i>Use case</i> Kelola Pedagang.	22
Tabel 4.4 Skenario <i>Use case</i> Tambah Pedagang	23
Tabel 4.5 Skenario <i>Use case</i> Hapus Pedagang	23
Tabel 4.6 Skenario <i>Use case</i> Kelola Pesanan Pedagang	24
Tabel 4.7 Skenario <i>Use case</i> Detail Pesanan Pedagang	25
Tabel 4.8 Skenario <i>Usecase</i> Konfirmasi Pesanan Pedagang	25
Tabel 4.9 Skenario <i>Use case</i> Kelola Barang Dagangan.....	26
Tabel 4.10 <i>Usecase</i> Tambah Barang Dagangan	27
Tabel 4. 11 <i>Usecase</i> Detail Barang Dagangan	27
Tabel 4.12 <i>Usecase</i> Edit Barang Dagangan	28
Tabel 4.13 <i>Usecase</i> Profil	29
Tabel 4.14 <i>Usecase</i> Edit Profil	29
Tabel 4.15 Skenario <i>Use case</i> Kelola Pesanan Pelanggan.....	30
Tabel 4. 16 Skenario <i>Use case</i> Detail Pesanan Pelanggan	30
Tabel 4. 17 <i>Usecase</i> konfirmasi pesanan pelanggan	31
Tabel 4. 18 Skenario <i>Use case</i> Pilih Pedagang	32
Tabel 4. 19 <i>Usecase</i> Pilih Barang.....	32
Tabel 4. 20 <i>Usecase</i> Pesanan Khusus.....	33
Tabel 4. 21 <i>Usecase</i> Keranjang Belanja.....	33
Tabel 4. 22 Skenario <i>Use case</i> Register.....	34
Tabel 4. 23 Skenario <i>Usecase</i> Login	35
Tabel 4. 24 Skenario <i>Use case</i> Logout.....	35
Tabel 4. 25 Kebutuhan Non- <i>Fungsional</i>	36
Tabel 4. 26 Skenario Pengujian	69
Tabel 5. 1 Spesifikasi Perangkat Keras	71
Tabel 5. 2 Spesifikasi Perangkat Lunak	71
Tabel 5. 3 Implementasi <i>Class Controller</i>	72
Tabel 5. 4 Implementasi <i>Class Model</i>	72

Tabel 6. 1 Register	88
Tabel 6. 2 Kelola Pesanan / Beranda Pelanggan	89
Tabel 6. 3 Detail Pesanan Pelanggan	89
Tabel 6. 4 Konfirmasi Pesanan Pelanggan	90
Tabel 6. 5 Pesanan Khusus.	90
Tabel 6. 6 Beranda Pedagang / Kelola Pesanan Pedagang.	91
Tabel 6. 7 Detail Pesanan Pedagang	91
Tabel 6. 8 Konfirmasi Pesanan Pedagang.	92
Tabel 6. 9 Kelola Barang Dagangan.....	92
Tabel 6. 10 Tambah Barang Dagangan.	92
Tabel 6. 11 Detail Barang Pedagang	93
Tabel 6. 12 Edit Data Barang.....	93
Tabel 6. 13 Profil Pedagang.....	94
Tabel 6. 14 Edit Profil.	94
Tabel 6. 15 Pemesanan Barang / Order	94
Tabel 6. 16 Tambah Pedagang	95
Tabel 6. 17 Hapus Pedagang	95
Tabel 6. 18 Menampilkan Detail Pesanan pelanggan	96
Tabel 6. 19 Kasus Uji menampilkan detail pesanan pelanggan	98
Tabel 6. 20 Konfirmasi Pesanan Pelanggan.	99
Tabel 6. 21 Kasus Uji konfirmasi pesanan pelanggan	102
Tabel 6. 22 Menampilkan List Barang Pedagang	102
Tabel 6. 23 Kasus Uji Menampilkan List Barang Pedagang.....	104
Tabel 6. 24 Kasus Uji Kelola Pedagang	105
Tabel 6. 25 Kasus Uji Validasi Tambah Pedagang	105
Tabel 6. 26 Kasus Uji Validasi Hapus Pedagang	106
Tabel 6. 27 Kasus Uji Validasi Kelola Pesanan.....	106
Tabel 6. 28 Kasus Uji Detail Pesanan Pedagang.....	106
Tabel 6. 29 Kasus Uji Konfirmasi Pesanan Pedagang.....	107
Tabel 6. 30 Kasus Uji Kelola Barang	107
Tabel 6. 31 Kasus Uji Tambah Barang	108

Tabel 6. 32 Detail Barang Dagangan	108
Tabel 6. 33 Kasus Uji Edit Data Barang	108
Tabel 6. 34 Kasus Uji Profil	109
Tabel 6. 35 Kasus Uji Edit Profil	109
Tabel 6. 36 Kasus Uji Kelola Pesanan Pelanggan	109
Tabel 6. 37 Kasus Uji Detail Pesanan Pelanggan	110
Tabel 6. 38 Kasus Uji Konfirmasi Pesanan Pelanggan	110
Tabel 6. 39 Kasus Uji Pilih Pedagang	110
Tabel 6. 40 Kasus Uji Pilih Barang	111
Tabel 6. 41 Kasus Uji Pesanan Khusus	111
Tabel 6. 42 Kasus Uji Keranjang Belanja	111
Tabel 6. 43 Kasus Uji Register	112
Tabel 6. 44 Kasus Uji <i>Login</i>	112
Tabel 6. 45 Kasus Uji <i>Logout</i>	112
Tabel 6. 46 Hasil Pengujian <i>Black-Box</i>	113
Tabel Lampiran 1 Daftar Harga Sayur	175



DAFTAR GAMBAR

Gambar 2.1 Ilustrasi Model <i>Prototype</i>	7
Gambar 2.2 <i>Model-View-Controller</i>	9
Gambar 3. 1 Diagram Alir Metode Penelitian	13
Gambar 4.1 <i>Diagram</i> Pohon Perancangan.....	17
Gambar 4.2 Gambaran Umum Sistem	18
Gambar 4.3 Usecase Diagram Pemesanan Sayur Keliling.....	21
Gambar 4. 4 <i>Activity Diagram</i> Tambah Pedagang	37
Gambar 4. 5 <i>Activity Diagram</i> Hapus Pedagang	37
Gambar 4. 6 <i>Activity diagram</i> kelola pesanan pedagang	38
Gambar 4. 7 <i>Activity Diagram</i> Detail Pesanan Pedagang	38
Gambar 4. 8 <i>Activity Diagram</i> Konfirmasi Pesanan Pedagang	39
Gambar 4. 9 <i>Activity Diagram</i> Kelola Barang	39
Gambar 4. 10 <i>Activity Diagram</i> Tambah Barang	40
Gambar 4. 11 <i>Activity Diagram</i> Kelola Pesanan Pelanggan	40
Gambar 4. 12 <i>Activity diagram</i> detail pesanan.....	41
Gambar 4. 13 <i>Activity Diagram</i> Konfirmasi Pesanan Pelanggan	41
Gambar 4. 14 <i>Activity Diagram</i> Pilih Pedagang	42
Gambar 4. 15 <i>Activity Diagram</i> Pilih Barang.....	42
Gambar 4. 16 <i>Activity Diagram</i> Pesanan Khusus.....	43
Gambar 4. 17 <i>Activity Diagram</i> Keranjang Belanja.....	43
Gambar 4. 18 <i>Activity Diagram</i> Login.....	44
Gambar 4. 19 <i>Activity Diagram</i> Logout.....	44
Gambar 4. 20 <i>Activity diagram</i> resgister	45
Gambar 4. 21 <i>sequence diagram</i> login admin	45
Gambar 4. 22 <i>sequence diagram</i> login guest.....	46
Gambar 4. 23 <i>sequence diagram</i> logout.....	46
Gambar 4. 24 <i>sequence diagram</i> register.....	47
Gambar 4. 25 <i>sequence diagram</i> kelola pesanan pedagang	47
Gambar 4. 26 <i>sequence diagram</i> detail pesanan pedagang.....	48
Gambar 4. 27 <i>sequence diagram</i> konfirmasi pedagang	48

Gambar 4. 28 <i>sequence diagram</i> batal pesanan pedagang.....	49
Gambar 4. 29 <i>sequence diagram</i> kelola barang	49
Gambar 4. 30 <i>Sequence Diagram</i> Detail Barang.....	50
Gambar 4. 31 <i>Sequence Diagram</i> Edit Barang Dagangan	50
Gambar 4. 32 <i>sequence diagram</i> tambah barang dagangan.....	51
Gambar 4. 33 <i>sequence diagram</i> profil.....	51
Gambar 4. 34 <i>sequence diagram</i> Edit profil	52
Gambar 4. 35 <i>sequence diagram</i> kelola pesanan pelanggan	52
Gambar 4. 36 <i>sequence diagram</i> detail pesanan pelanggan.....	53
Gambar 4. 37 <i>sequence diagram</i> konfirmasi pesanan pelanggan.....	53
Gambar 4. 38 <i>sequence diagram</i> pilih pedagang.....	54
Gambar 4. 39 <i>sequence diagram</i> pilih barang.	54
Gambar 4. 40 <i>sequence diagram</i> pesanan khusus	55
Gambar 4. 41 <i>sequence diagram</i> keranjang belanja	55
Gambar 4. 42 <i>sequence diagram</i> tambah pedagang.....	56
Gambar 4. 43 <i>sequence diagram</i> hapus pedagang.....	56
Gambar 4. 44 <i>class diagram</i> class Controller	57
Gambar 4. 45 <i>class diagram</i> class Model.....	58
Gambar 4. 46 Entity Relationship Diagram	58
Gambar 4. 47 Antarmuka <i>Login Admin</i>	59
Gambar 4. 48 antarmuka beranda(kelola pedagang).....	60
Gambar 4. 49 antarmuka halaman tambah pedagang.....	60
Gambar 4. 50 antarmuka halaman <i>login</i>	61
Gambar 4. 51 halaman beranda(kelola pesanan).....	61
Gambar 4. 52 antarmuka halaman detail pesanan.....	62
Gambar 4. 53 antarmuka halaman kelola barang	62
Gambar 4. 54 antarmuka halaman tambah barang.....	63
Gambar 4. 55 antarmuka halaman detail barang.....	63
Gambar 4. 56 antarmuka halaman edit data barang.....	64
Gambar 4. 57 antarmuka halaman profil pedagang.....	64
Gambar 4. 58 antarmuka halaman edit data pedagang.	65
Gambar 4. 59 menunjukkan antarmuka halaman <i>login</i> pelanggan.	65

Gambar 4. 60 merupakan antarmuka <i>register</i>	66
Gambar 4. 61 menunjukkan antarmuka halaman beranda.....	66
Gambar 4. 62 antarmuka halaman detail pesanan.	67
Gambar 4. 63 antarmuka halaman pilih pedagang.....	67
Gambar 4. 64 menunjukkan antarmuka halaman <i>list</i> barang pedagang.	68
Gambar 4. 65 antarmuka halaman <i>list</i> barang pedagang.....	68
Gambar 4. 66 menunjukkan antarmuka keranjang belanja	69
Gambar 5. 1 <i>Diagram</i> Alir Implementasi	70
Gambar 5. 2 Implementasi <i>Database</i>	73
Gambar 5. 3 Impementasi tabel <i>admin</i>	74
Gambar 5. 4 implementasi tabel pesanan	74
Gambar 5. 5 Implementasi tabel Pesanan Khusus.....	75
Gambar 5. 6 Implementasi tabel Barang Pesanan.....	75
Gambar 5. 7 implementasi tabel <i>user</i>	76
Gambar 5. 8 implementasi tabel privillage.....	76
Gambar 5. 9 implementasi tabel sayuran.....	77
Gambar 5. 10 Implementasi tabel satuan harga.....	77
Gambar 5. 11 Implementasi tabel Status Pesanan.....	77
Gambar 5. 12 Implementasi tabel kategori	78
Gambar 5. 13 Implementasi Antarmuka Halaman <i>Login</i> Guest	78
Gambar 5. 14 Implementasi Antarmuka Halaman <i>Login</i> Admin	79
Gambar 5. 15 Implementasi Antarmuka Halaman <i>Register</i>	79
Gambar 5. 16 Implementasi Antarmuka Halaman <i>Register</i>	79
Gambar 5. 17 Implementasi Antarmuka Halaman Beranda <i>Admin</i>	80
Gambar 5. 18 Implementasi antarmuka Halaman Tambah Pedagang.....	80
Gambar 5. 19 Implementasi antarmuka Kelola Pesanan.....	81
Gambar 5. 20 Implementasi Antarmuka Detail Pesanan Pedagang.....	81
Gambar 5. 21 Implementasi Antarmuka Halaman Kelola Barang	82
Gambar 5. 22 Implementasi Antamuka Halaman Detail Barang.....	82
Gambar 5. 23 Implementasi Antarmuka Halaman Tambah Pedagang	83
Gambar 5. 24 Implementasi Antarmuka Halaman Edit Data Barang	83
Gambar 5. 25 Implementasi Antarmuka Halaman Profil Pedagang	84

Gambar 5. 26 Implementasi Halaman Edit Data Pedagang	84
Gambar 5. 27 Implementasi Halaman Kelola Pesanan	85
Gambar 5. 28 Implementasi Halaman Detail Pesanan	85
Gambar 5. 29 Implementasi Antarmuka Halaman Pilih Pedagang.....	86
Gambar 5. 30 Implementasi Antarmuka <i>List</i> Barang Dagangan.....	86
Gambar 5. 31 Implementasi Halaman Pesanan Khusus	87
Gambar 5. 32 Implementasi Antarmuka Halaman Keranjang Belanja	87
Gambar 6. 1 <i>Flowgraph</i> Detail Pesanan Pelanggan	97
Gambar 6. 2 <i>Flowgraph</i> Konfirmasi Pesanan Pelanggan	101
Gambar 6. 3 <i>Flowgraph</i> Menampilkan <i>List</i> Barang Pedagang.....	103
Gambar 6. 4 <i>Overall</i> Kasus Uji PT_001	116
Gambar 6. 5 <i>Page Performance</i> Kasus Uji PT_001.....	117
Gambar 6. 6 <i>Response VS Time Summary</i> Kasus Uji PT_001	117
Gambar 6. 7 <i>Server Health Summary</i> Kasus Uji PT_001	118
Gambar 6. 8 Page Size Kasus Uji PT_001	118
Gambar 6. 9 Summary Kasus Uji PT_001	119
Gambar 6. 10 Overall Kasus Uji PT_002	119
Gambar 6. 11 <i>Page performance</i> Kasus Uji PT_002.....	120
Gambar 6. 12 <i>Response VS Time Summary</i> Kasus Uji PT_002	120
Gambar 6. 13 <i>Server Health Summary</i> Kasus Uji PT_002	121
Gambar 6. 14 Page Size kasus Uji PT_002.....	121
Gambar 6. 15 Kasus Uji PT_002	122
Gambar 6. 16 <i>Overall</i> Kasus Uji PT_003.....	122
Gambar 6. 17 <i>Page Performance</i> Kasus Uji PT_003.....	123
Gambar 6. 18 <i>Response VS Time Summary</i> Kasus Uji PT_003	123
Gambar 6. 19 <i>Server Health Summary</i> Kasus Uji PT_003	124
Gambar 6. 20 Page size kasus uji PT_003	124
Gambar 6. 21 Summary Kasus Uji PT_003	125
Gambar 6. 22 <i>Overall</i> Kasus Uji PT_004.....	125
Gambar 6. 23 <i>Page Performance</i> Kasus Uji PT_004.....	126
Gambar 6. 24 <i>Response VS Time Summary</i> Kasus Uji PT_004	126
Gambar 6. 25 <i>Server Health Summary</i> Kasus Uji PT_004	127

Gambar 6. 26 Page Size kasus uji PT_004	127
Gambar 6. 27 Summary Kasus Uji PT_004	128
Gambar 6. 28 Overall Kasus Uji PT_005	128
Gambar 6. 29 Page Performance Kasus Uji PT_005	129
Gambar 6. 30 Response VS Time Summary Kasus Uji PT_005	129
Gambar 6. 31 Server Health Summary Kasus Uji PT_005	130
Gambar 6. 32 Page Size kasus uji PT_005	130
Gambar 6. 33 Summary Kasus Uji PT_005	131
Gambar 6. 34 Overall Kasus Uji PT_006	131
Gambar 6. 35 Page Performance Kasus Uji PT_006	132
Gambar 6. 36 Response VS Time Summary Kasus Uji PT_006	132
Gambar 6. 37 Server Health Summary Kasus Uji PT_006	133
Gambar 6. 38 page size kasus uji PT_006	133
Gambar 6. 39 Summary Kasus Uji PT_006	134
Gambar 6. 40 Overall Kasus Uji PT_007	134
Gambar 6. 41 Page Performance Kasus Uji PT_007	135
Gambar 6. 42 Response VS Time Summary Kasus Uji PT_007	135
Gambar 6. 43 Server Health Summary Kasus Uji PT_007	136
Gambar 6. 44 Page size kasus uji PT_007	136
Gambar 6. 45 Summary Kasus Uji PT_007	137
Gambar 6. 46 Kasus Uji PT_008.	137
Gambar 6. 47 Page Performance Kasus Uji PT_008	138
Gambar 6. 48 Response VS Time Summary Kasus Uji PT_008	138
Gambar 6. 49 Server Health Summary Kasus Uji PT_008	139
Gambar 6. 50 Page size kasus uji PT_008	139
Gambar 6. 51 Summary Kasus Uji PT_008	140
Gambar 6. 52 Overall Kasus Uji PT_009	140
Gambar 6. 53 Page Performance Kasus Uji PT_009	141
Gambar 6. 54 Response VS Time Summary Kasus Uji PT_009	141
Gambar 6. 55 Server Health Summary Kasus Uji PT_009	142
Gambar 6. 56 Page size kasus uji PT_009	142
Gambar 6. 57 Summary Kasus Uji PT_009	143

Gambar 6. 58 <i>Overall</i> Kasus Uji PT_010	143
Gambar 6. 59 <i>Page Performance</i> Kasus Uji PT_010	144
Gambar 6. 60 <i>Response VS Time Summary</i> Kasus Uji PT_010	144
Gambar 6. 61 <i>Server Health Summary</i> Kasus Uji PT_010	145
Gambar 6. 62 <i>Page Size</i> kasus uji PT_10	145
Gambar 6. 63 <i>Summary</i> Kasus Uji PT_010	146
Gambar 6. 64 <i>Overall</i> Kasus Uji PT_011	146
Gambar 6. 65 <i>Page Performance</i> Kasus uji PT_011	147
Gambar 6. 66 <i>Response VS Time Summary</i> Kasus Uji PT_011	147
Gambar 6. 67 <i>Server Health Summary</i> Kasus Uji PT_011	148
Gambar 6. 68 <i>Page Size</i> kasus uji PT_011	148
Gambar 6. 69 <i>Summary</i> Kasus Uji PT_011	149
Gambar 6. 70 <i>Overall</i> Kasus Uji PT_012	149
Gambar 6. 71 <i>Page Performance</i> Kasus Uji PT_012	150
Gambar 6. 72 <i>Response VS Time Summary</i> Kasus Uji PT_012	150
Gambar 6. 73 <i>Server Health Summary</i> Kasus Uji PT_012	151
Gambar 6. 74 <i>Page Size</i> Kasus Uji PT_012	151
Gambar 6. 75 <i>Summary</i> Kasus Uji PT_012	152
Gambar 6. 76 <i>Overall</i> Kasus Uji PT_013	152
Gambar 6. 77 <i>Page Performance</i> Kasus Uji PT_013	153
Gambar 6. 78 <i>Response VS Time Summary</i> Kasus Uji PT_013	153
Gambar 6. 79 <i>Server Health Summary</i> Kasus Uji PT_013	154
Gambar 6. 80 <i>page size</i> kasus uji PT_013	154
Gambar 6. 81 <i>Summary</i> Kasus Uji PT_013	155
Gambar 6. 82 <i>Overall</i> Kasus Uji PT_014	155
Gambar 6. 83 <i>Page Performance</i> Kasus Uji PT_014	156
Gambar 6. 84 <i>Response VS Time Summary</i> Kasus Uji PT_014	156
Gambar 6. 85 <i>Server Health Summary</i> Kasus Uji PT_014	157
Gambar 6. 86 <i>Page size</i> kasus uji PT_014	157
Gambar 6. 87 <i>Summary</i> Kasus Uji PT_014	158
Gambar 6. 88 <i>Overall</i> Kasus Uji PT_015	158
Gambar 6. 89 <i>Page Performance</i> Kasus Uji PT_015	159

Gambar 6. 90 <i>VS Time Summary</i> Kasus Uji PT_015	159
Gambar 6. 91 <i>Server Health Summary</i> Kasus Uji PT_015	160
Gambar 6. 92 page size kasus uji PT_015	160
Gambar 6. 93 Summary Kasus Uji PT_015	161
Gambar 6. 94 <i>Overall</i> Kasus Uji PT_016	161
Gambar 6. 95 <i>Page Performance</i> Kasus Uji PT_016	162
Gambar 6. 96 <i>Response VS Time Summary</i> Kasus Uji PT_016	162
Gambar 6. 97 <i>Server Health Summary</i> Kasus Uji PT_016	163
Gambar 6. 98 page size kasus uji PT_016	163
Gambar 6. 99 Summary Kasus Uji PT_016	164
Gambar 6. 100 <i>Overall</i> Kasus Uji PT_017	164
Gambar 6. 101 <i>Page Performance</i> Kasus Uji PT_017	165
Gambar 6. 102 <i>Response VS Time Summary</i> Kasus Uji PT_017	165
Gambar 6. 103 <i>Server Health Summary</i> Kasus Uji PT_017	166
Gambar 6. 104 page size kasus uji PT_017	166
Gambar 6. 105 Summary Kasus Uji PT_017	167
Gambar 6. 106 Hasil Uji <i>Page Performance</i>	168
Gambar 6. 107 Hasil Uji Page size	169
Gambar 6. 108 Hasil Uji <i>Server Health Summary</i>	169

BAB 1 PENDAHULUAN

1.1 Latar belakang

Budaya makan sayur telah ada sejak jaman dahulu, bahkan jauh sebelum ada ilmu gizi yang menyatakan dengan jelas akan manfaatnya bagi kesehatan. Sayur merupakan sumber makanan yang mengandung gizi lengkap dan sehat. Dari data yang didapatkan melalui *website* Pusdatin-Kementrian Pertanian, konsumsi sayuran perkapita nasional tahun 2009-2014 meningkat sekitar 39.5% (Pusdatin, 2015).

Semakin banyaknya wanita yang memilih berkarier saat ini membuat mereka tidak memiliki banyak waktu untuk mengurus rumah tangganya, terutama dalam memperhatikan makanan yang akan disajikan. Keterbatasan waktu dan jarak ini membuat mereka semakin kesulitan untuk mencari bahan makanan untuk memasak. Penjual sayur keliling merupakan salah satu alternatif kepada masyarakat yang tidak memiliki waktu untuk pergi ke pasar. Penjual sayur keliling atau yang sering disebut dengan (*vegetabel merchant circle*) adalah salah satu usaha yang merupakan kegiatan perdagangan eceran dan melaksanakan pemberian jasa (Malik, 2012). Dengan adanya penjual sayur keliling ini masyarakat sangat terbantu karena tidak perlu jauh-jauh pergi ke pasar untuk membeli sayur. Berdasarkan hasil wawancara pada beberapa penjual sayur, para penjual sayur keliling ini mencoba menyediakan bahan makan apa saja yang diperlukan dalam kondisi segar dan bersih sebanyak mungkin. Terkadang masih ada pembeli yang tidak mendapat bahan makanan yang diinginkan karena sudah habis terlebih dahulu meskipun para penjual sayur ini sudah berusaha menyediakan bahan makanan yang lengkap. Namun ketika penjual sayur melebihkan bahan makanan, terkadang tidak habis dan jadi busuk.

Untuk mengatasi permasalahan tersebut, maka dibutuhkan sebuah sistem yang dapat menjembatani antara pembeli dan penjual sayur keliling. Sistem akan mempermudah pembeli untuk memesan bahan makanan yang diinginkan sehingga pembeli tidak kehabisan bahan makanan dan penjual dapat mengetahui bahan makanan apa saja yang diinginkan sehingga dapat mengurangi kerugian dari bahan makanan yang sisa dan menjadi busuk. Sistem yang akan dibangun berbasis *web* dengan menggunakan model *prototype* untuk memudahkan memahami kebutuhan pengguna.

Kegagalan sistem dalam sebuah perangkat lunak tidak dapat ditoleransi, karena ketika terjadi kesalahan sistem biaya yang dikeluarkan akan sangat besar. Oleh karena itu diperlukan sebuah pencegahan untuk menanggulangi kesalahan sistem yaitu dengan *software testing* (Patil, 2012). Pengembang biasanya mengukur *quality of service* dari aplikasi *web* dengan *response time*, *throughput* dan *availability* (menasce, 2002). Pengujian pada aplikasi berbasis *web* dilakukan untuk menemukan kesalahan pada fungsi, usabilitas, navigasi, performa, kapasitas dan keamanan. *Performance testing* digunakan untuk menentukan *responsiveness*, *throughput*, *reliability* dan *scalability* pada saat sistem digunakan (Patil, 2012).

Performance test adalah sebuah *test* pada perangkat lunak yang bertujuan untuk menguji kinerja perangkat lunak dalam sistem yang terintegrasi, *performance test* melibatkan semua proses *testing* pada kinerja sistem yang diuji (Pressman, 2001). Pada survey yang telah dilakukan, para pengguna sangat sensitif pada delay atau respon waktu yang melebihi 10 detik pada penggunaan aplikasi karena dapat mengganggu aktivitas mereka (Shaw, 2000). Oleh karena itu dibutuhkan pengujian untuk menguji performa dan respon waktu dari aplikasi sebelum aplikasi tersebut diberikan kepada pelanggan.

Salah satu *software* pengujian performa yaitu *Rational Performance Tester*. *Rational Performance Tester* (RPT) adalah salah satu kakas bantu pengujian yang dikembangkan oleh IBM yang memiliki kelebihan yaitu dapat melakukan penjadwalan pengujian, dapat merekam pengujian, memiliki datapool dan menampilkan hasil pengujian secara realtime serta berbasis eclipse (IBM-Team, 2008). Dari paparan informasi di atas, penulis mengambil judul skripsi dengan judul "Pembangunan Aplikasi Pemesanan Sayur Keliling dengan Menerapkan *Rational Performance Tester* Sebagai Kakas Bantu dalam Pengujian Perangkat Lunak".

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah diuraikan di atas maka didapatkan rumusan masalah sebagai berikut:

1. Bagaimana merancang dan menerapkan model *prototype* dalam membangun aplikasi penjualan sayur menjadi sebuah aplikasi berbasis *web*?
2. Bagaimana hasil pengujian performa sistem menggunakan *Rational Performance tester*?

1.3 Tujuan

Tujuan yang ingin di capai pada pembangunan aplikasi pemesanan sayur ini adalah:

1. Mengembangkan aplikasi pemesanan sayur yang dapat membantu masyarakat untuk memesan sayur dari penjual sayur.
2. Melakukan pengujian performa pada aplikasi pemesanan sayur pada penjual sayur keliling.

1.4 Manfaat

Hasil dari penelitian ini diharapkan dapat membantu masyarakat untuk memudahkan memesan sayur pada penjual sayur keliling dengan performa yang maksimal.

1.5 Batasan masalah

Agar permasalahan yang di rumuskan dapat lebih terfokus, maka pada penelitian ini dibatasi dalam hal:

1. Ruang lingkup sistem ini hanya pada daerah siguragura malang dan sekitarnya.
2. Aplikasi ini hanya digunakan oleh pedagang sayur dan pelanggannya.
3. Performa aplikasi diuji menggunakan *Rational Performance tester*.

1.6 Sistematika pembahasan

Penyusunan skripsi ini menggunakan kerangka pembahasan yang tersusun sebagai berikut:

BAB I PENDAHULUAN

Bab ini merupakan dasar dari penyusunan skripsi ini yang terdiri dari latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, dan sistematika penulisan.

BAB II LANDASAN KEPUSTAKAAN

Bab ini berisi kajian pustaka, referensi atau sumber sumber yang berhubungan dengan permasalahan dalam skripsi yang meliputi: penjual sayur keliling, *rational performance tester*, pemesanan dan lain-lain.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan bagaimana metodologi untuk perancangan pembangunan aplikasi pemesanan sayur keliling.

BAB IV PERANCANGAN SISTEM

Bab ini menjelaskan analisa, perancangan dan kebutuhan aplikasi pemesanan sayur keliling, serta menjawab permasalahan yang diuraikan pada perumusan masalah.

BAB V IMPLEMENTASI

Bab ini membahas implementasi dari aplikasi pemesanan sayur keliling sesuai dengan perancangan yang telah dibuat.

BAB VI PENGUJIAN DAN ANALISIS

Bab ini menjelaskan hasil pengujian aplikasi berdasarkan parameter-parameter yang telah diterapkan dan kemudian dilakukan analisa terhadap hasil pengujian.

BAB VI PENUTUP

Bab ini menjelaskan kesimpulan yang dapat diambil dari pengujian dan analisis pembuatan aplikasi pemesanan sayur pada penjual sayur keliling dan disertai saran yang mendukung untuk pengembangan selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Kajian pustaka yang pertama merupakan jurnal dari *International Journal Of Emerging Technology and Advanced Engineering*, oleh Sheetal S. Patil dan Prof. S.D. Joshi dengan judul "*Identification of Performance Improving Factors for Web Application by Performance testing*". Tujuan dari penelitian untuk mengetahui pentingnya pengujian performa sebelum sistem digunakan. Kakas bantu yang digunakan pada penelitian ini yaitu *jmeter*. *Load testing* diukur dengan menggunakan simulasi *user load*, *Stress testing* diukur dengan ketahanan terhadap beban yang diberikan, *strength testing* merupakan versi lama dari *Stress testing* dan *Load testing*, *capacity testing* adalah pelengkap dari *Load testing* (Patil, 2012).

Kajian pustaka kedua merupakan jurnal dari *George Mason University*, dengan judul "*Load Testing of Web Sites*" oleh Daniel A. Menasce. Paper ini membahas tentang faktor-faktor yang mempengaruhi dalam *load testing*, termasuk pihak ketiga seperti CDNs dan iklan, distribusi geografis dan duplikasi, streaming dan akses wireless. Faktor-faktor yang telah disebutkan secara signifikan mempengaruhi kinerja *website* dan skalabilitasnya sehingga menimbulkan tantangan untuk para pengembang. Untuk menghasilkan pengujian beban yang maksimal dibutuhkan pengumpulan data yang banyak dan bervariasi (menasce, 2002).

Kajian pustaka ketiga jurnal dari fakultas teknologi informasi, universitas Kristen satya wacana dengan judul "*Aplikasi Reservasi Tiket Bus pada Handphone Android Menggunakan Web Service*" oleh Fajar Fani Hartono, Hendry Dan Ramos Somya. Paper ini membahas bagaimana membuat aplikasi reservasi tiket bus menggunakan metode *prototype* dengan platform *android*. Tujuan dari penelitian ini adalah memudahkan para konsumen untuk memesan tiket tanpa melalui agen dan dapat dengan mudah mengetahui ketersediaan tiket dengan akurat. Aplikasi ini menggunakan *web service* untuk mengintegrasikan antara server dengan *client* tanpa keterbatasan platform sistem. Pengimplementasian aplikasi ini menggunakan *ksoap2_android* sebagai *library* untuk memudahkan pembuatan aplikasi. Metode *prototype* digunakan untuk memudahkan proses analisa dan perancangan aplikasi, sehingga aplikasi dapat sesuai dengan permintaan pengguna. Hasil pembuatan aplikasi ini ada dua yaitu *web admin* sebagai server dan aplikasi *android* sebagai *client*. Penelitian ini membuktikan bahwa dengan dibuatnya aplikasi reservasi dapat membantu konsumen untuk menghemat waktu biaya dan tenaga dalam pemesanan tiket, serta dapat dengan mudah mengetahui ketersediaan tiket, karena dengan menggunakan *mobile device* transaksi bisa dilakukan kapan saja yang di mana saja (Hartono et al., 2013).

Sistem yang akan dibangun pada aplikasi pemesanan sayur ini berbasis *web responsive* dan menggunakan *android studio* sebagai alat bantu untuk membuat *web view*. Perancangan sistem ini menggunakan metode pengembangan

prototype yang akan dilakukan secara bertahap yaitu dengan mengembangkan *prototype* yang sederhana terlebih dahulu kemudian dikembangkan menjadi perangkat lunak yang siap digunakan. Tujuan utama dari sistem ini adalah untuk membantu konsumen memesan barang pada penjual sayur keliling sehingga proses pemesanan lebih efektif. Sistem dibangun dengan menggunakan bahasa pemrograman PHP dengan *framework Codeigniter* dan penyimpanan data menggunakan MySQL.

2.2 Rekayasa Perangkat Lunak

Rekayasa perangkat lunak merupakan profesi yang dilakukan oleh seorang perancang perangkat lunak yang berkaitan dengan pembuatan dan pemeliharaan aplikasi perangkat lunak dengan menerapkan teknologi dan praktek dari ilmu komputer, manajemen proyek, dan bidang-bidang lainnya. Terdapat 3 tingkatan pada perangkat lunak, yaitu tingkatan program aplikasi (*application program*, misalnya Microsoft Office), tingkatan sistem operasi (*operating system*, misalnya Microsoft Windows), dan tingkatan bahasa pemrograman (java, pascal, C, C++ dll) (Simarmata, 2010). Ada 2 jenis perangkat lunak berkaitan dengan pengembangan perangkat lunak, yaitu (Sommerville, 2011):

1. *Generic Product*

Generic Product adalah sistem yang berdiri sendiri yang diproduksi oleh pengembang dan dijual di pasar bebas kepada siapa saja yang ingin membelinya. Contoh dari jenis produk termasuk *software* untuk PC seperti *database*, pengolah kata, *tools* menggambar, *tools* proyek manajemen, dll.

2. *Customized product* (dipesan lebih dahulu)

Customized product adalah sistem yang dipesan oleh pelanggan tertentu. Pengembang perangkat lunak mengembangkan perangkat lunak khusus untuk pelanggan itu. Contoh perangkat lunak jenis ini termasuk sistem kontrol untuk perangkat elektronik, sistem dibuat untuk mendukung proses bisnis tertentu, dan sistem kontrol lalu lintas udara.

2.3 Penjual Sayur Keliling

Penjual sayur keliling atau yang sering disebut dengan (*vegetable merchant circle*) adalah salah satu usaha yang merupakan kegiatan perdagangan eceran dan melaksanakan pemberian jasa (Malik, 2012). Tidak bisa dipungkiri para penjual sayur keliling ini memiliki peran penting dalam kehidupan sehari-hari terutama di daerah yang jauh dari pasar tradisional. Dengan adanya para penjual sayur keliling ini maka masyarakat tidak perlu jauh-jauh pergi ke pasar untuk membeli sayur-mayur yang mereka butuhkan. Meskipun dalam satu daerah tidak hanya terdapat satu atau dua penjual sayur keliling namun setiap penjual sayur keliling sudah memiliki pelanggan sendiri-sendiri. Setiap penjual sayur keliling memiliki keunikan sendiri untuk menarik pelanggannya, sebagai contoh yaitu dalam pemanggilan pelanggan biasanya penjual sayur ini memiliki kode yang unik berupa suara yang bisa membuat para pelanggan mengenali mereka.

Para penjual sayur keliling mengambil barang dagangan mereka dari pasar tradisional, mereka berangkat pagi untuk segera ke pasar dan mendapatkan barang dagangan yang dicari. Dengan berangkat pagi memungkinkan mereka untuk mendapatkan barang lebih lengkap, selain berangkat lebih pagi para penjual sayur ini bersaing untuk kelengkapan barang dagangan yang dibawa untuk dijual. Faktor inilah yang membuat harga di penjual sayur keliling bisa lebih mahal sedikit dibandingkan dengan pasar tradisional. Dengan melihat manfaat hadirnya penjual sayur keliling, maka kegiatan usahanya perlu dibina dan ditata agar benar-benar bermanfaat, baik bagi pedagang sayur keliling sendiri maupun bagi masyarakat serta pemerintah.

2.4 Pemesanan

Pemesanan atau bisa juga disebut order memiliki banyak sekali pengertian, di mana pada penulisan skripsi ini pemesanan yaitu proses meminta suatu barang pada seseorang untuk disediakan. Sedangkan menurut kamus besar bahasa Indonesia pemesanan yaitu “proses, perbuatan, cara memesan atau memesan, meminta supaya disediakan barang”. Adapun pengertian pemesanan menurut para ahli di bidangnya yaitu:

1. Menurut Gouzali

Pemesanan merupakan penerimaan pesanan oleh pelanggan terhadap suatu produk yang dipesan sebelumnya. Lanjutan dari pemesanan adalah pengiriman produk sampai ke tangan pemesan dengan selamat(Gouzali, 1996).

2. Menurut Agus

Pemesanan merupakan keseluruhan proses kegiatan yang berkaitan dengan pengelolaan inventory atau persediaan tempat pendistribusian produk dan catatan keseluruhan transaksi pemesanan(Sunyoto, 1999).

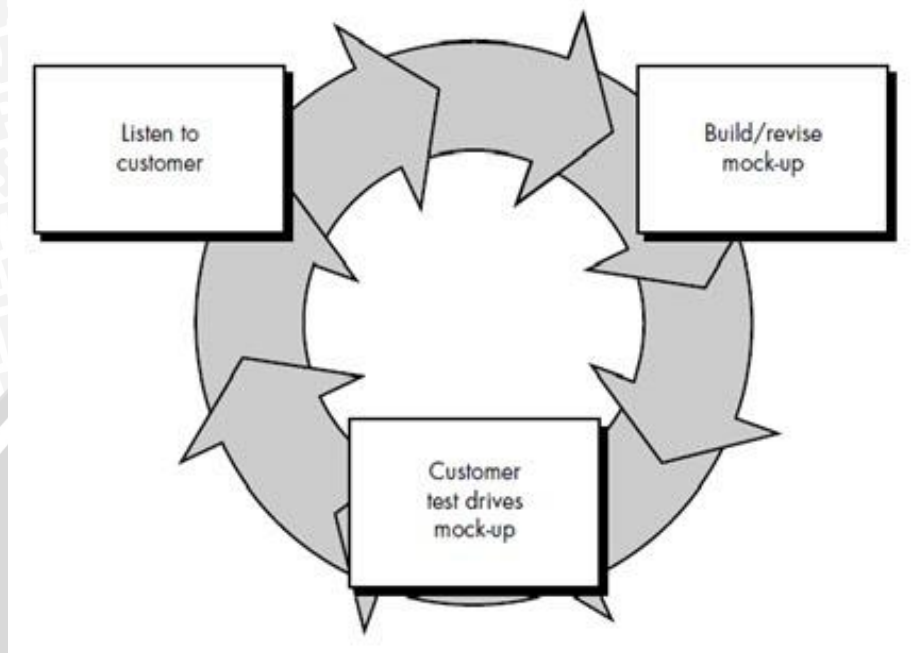
2.5 Software Development Life Cycle (SDLC)

Software development life cycle atau sering disebut juga sistem *development life cycle* merupakan proses mengembangkan atau mengubah suatu sistem perangkat lunak dengan menggunakan *model-model* dan metodologi yang digunakan orang untuk mengembangkan sistem-sistem perangkat lunak sebelumnya (Salahudin, 2011). Dalam penerapannya SDLC memiliki beberapa *Model*, semuanya memiliki kelemahan dan kelebihan pada setiap *Model*nya. Untuk memilih model SDLC pengembang perlu mengenali tipe pelanggan (*customer*) lalu memilih *Model* SDLC yang sesuai dengan karakter pelanggan (*customer*) dan sesuai dengan karakter pengembang.

2.5.1 Prototype

Seringkali pelanggan (*customer*) membayangkan kumpulan kebutuhan yang diinginkan tapi tidak secara spesifik dan detail sehingga para pengembang akan kesulitan memahami apa yang sebenarnya diinginkan oleh para pelanggan (*customer*) ini. *Model prototype* digunakan untuk menghubungkan

ketidapkahaman pelanggan mengenai hal teknis dan memperjelas spesifikasi kebutuhan yang diinginkan pelanggan kepada pengembang. Pada gambar ditunjukkan ilustrasi *Model prototype*:



Gambar 2.1 Ilustrasi Model *Prototype*

Sumber: Salahudin, 2011.

Model prototype (prototyping *Model*) dimulai dengan mengumpulkan kebutuhan pelanggan terhadap perangkat lunak yang akan dibuat yang ditunjukkan pada gambar 2.1 yaitu *listen to customer*. Selanjutnya dibuatlah *prototype* dari sistem supaya pelanggan mempunyai gambaran dari sistem yang akan dibuat dan mengetahui apa yang sebenarnya diinginkan, pada gambar 2.1 *build/revise mock-up*. Selanjutnya *customer test drives mock-up* merupakan fase di mana *customer* mencoba *prototype* untuk mengetahui apakah sesuai dengan permintaan atau tidak. Program *prototype* merupakan program yang belum jadi. Program *prototype* ini biasanya menampilkan simulasi alur perangkat lunak sehingga akan nampak seperti program yang sudah jadi. *Model prototype* sesuai digunakan untuk menjabarkan kebutuhan pelanggan secara lebih detail karena pelanggan seringkali kesulitan menyampaikan kebutuhan yang diinginkan secara detail tanpa melihat gambaran yang jelas. Adapun kelebihan dari *Model prototype* yaitu:

1. Dengan keterlibatan pengguna maka pendefinisian kebutuhan pengguna menjadi lebih baik.
2. Kepuasan pengguna meningkat dan risiko pengguna tidak menggunakan sistem lebih kecil, mengingat keterlibatan mereka yang sangat tinggi sehingga sistem memenuhi kebutuhan mereka dengan lebih baik.
3. Waktu pengembangan lebih singkat.

4. Kesalahan pembuatan program dapat diperkecil karena pada setiap versi prototipe, kesalahan segera terdeteksi oleh pemakai.
5. Pemakai memiliki kesempatan yang lebih banyak dalam meminta perubahan-perubahan.

Untuk meModelkan sebuah perangkat lunak dibutuhkan beberapa tahapan dalam proses pengembangannya. Tahapan inilah yang akan menentukan keberhasilan dari sebuah *software* itu. Tahapan-tahapan dalam *Model prototype* yaitu (Sommerville, 2001 dan Simarmata, 2010):

1. Pengumpulan kebutuhan.

Tahapan yang pertama dilakukan yaitu mengumpulkan kebutuhan atau *requirement* yang dibutuhkan dengan cara membuat kuisisioner, wawancara ataupun langsung melakukan survei dan mendefinisikan semua kebutuhan bersama dengan pelanggan. Pada tahapan ini juga membuat garis besar sistem yang akan dibuat beserta fungsi-fungsi yang akan dibuat.

2. Membangun *prototype*.

Selanjutnya pengembang membangun *prototype* dengan membuat perancangan sementara yang berpusat pada penyajian kepada pelanggan (misalnya dengan membuat *input* dan contoh *outputnya*), sehingga pelanggan mempunyai gambaran awal mengenai sistem yang akan dibuat.

3. Evaluasi prototyping.

Pelanggan melakukan evaluasi apakah *prototype* yang sudah dibangun sudah sesuai dengan keinginan pelanggan. Jika sudah sesuai maka langkah keempat akan diambil. Jika tidak, maka *prototype* diperbaiki dengan mengulang langkah 1, 2, dan 3.

4. Mengkodekan sistem.

Setelah *prototype* disetujui oleh pelanggan langkah selanjutnya yaitu pengkodean sistem. Pada tahap ini, pengembang melakukan kegiatan pengkodean yang biasa disebut coding, yaitu menterjemahkan sistem ke dalam kode program dengan bahasa tertentu.

5. Menguji sistem.

Setelah sistem sudah menjadi suatu perangkat lunak yang siap pakai, maka sistem terlebih dulu harus di uji. Proses pengujian bertujuan untuk mengetahui apakah sistem yang telah dibuat telah sesuai dengan kebutuhan pengguna atau belum. Pengujian dapat dilakukan dengan berbagai metode yaitu *White-Box*, *Black-Box*, *Basis Path*, pengujian arsitektur dan lain-lain.

6. Evaluasi sistem.

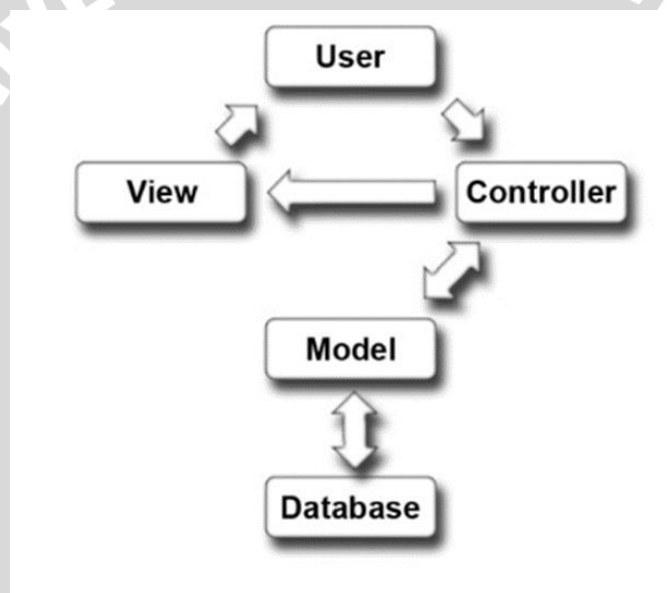
Setelah melakukan pengujian dan sistem sudah sesuai dengan kebutuhan pelanggan maka sistem siap diberikan kepada pelanggan, jika masih ada yang belum sesuai maka akan dilakukan revisi sesuai apa yang diminta oleh pelanggan dan mengulang langkah 4 dan 5.

7. Menggunakan sistem.

Langkah terakhir yaitu penyerahan sistem yang sudah siap digunakan kepada pelanggan.

2.6 Codeigniter

Codeigniter adalah aplikasi *open source* yang berupa *framework* untuk bahasa pemrograman php. *Codeigniter* dikembangkan oleh Rick Ellis, dengan versi awal yang rilis pada tanggal 28 Februari 2006. *Codeigniter* mampu berjalan pada lingkungan *shared hosting* karena memiliki ukuran yang sangat kecil, namun memiliki kinerja yang sangat luar biasa (Griffiths et al, 2010). *Codeigniter* menggunakan pola desain *Model-View-Controller* (MVC). *Model-View-Controller* memudahkan untuk meletakkan bagian dari komponen-komponen *framework Codeigniter* berada pada tempat yang benar. Pada gambar 2.2 ditunjukkan struktur kerja MVC.



Gambar 2.2 Model-View-Controller

Sumber: Griffiths et al, 2010

Adapun komponen MVC seperti berikut:

- **Model:** Merupakan entitas dari sistem, *Model* berhubungan dengan *database* untuk memanipulasi data (*insert, update, delete*). Selain itu *Model* juga berisi informasi tentang bagaimana berhubungan dengan *Model* lainnya. Beberapa contoh data yang bias digunakan adalah *RSS Feed, API calls*, dan setiap tindakan lainnya yang melibatkan pengambilan (*retrieving*), pengembalian (*returning*), memperbarui (*updating*), menghapus (*removing*) data (Griffiths et al, 2010).
- **Controller:** Merupakan pengontrol atau pengendali setiap permintaan dari pengguna yang merequest data dan menyambungkannya untuk diproses oleh aplikasi yang menjadi perantara antara *Model* dan *view*. *Controller* adalah inti

dari aplikasi karena bagian menentukan bagaimana permintaan HTTP harus ditangani (Griffiths et al, 2010).

- *View*: *View* bertanggung jawab untuk menampilkan data yang diminta dari *Model* melalui *Controller*. Tidak ada logika pemrograman, tidak ada *query insert* atau *update* yang harus dijalankan di sini, meskipun akses data bias terjadi pada *file-file* ini (Blanco et al., 2009).

2.7 Pengujian Perangkat Lunak

Pengujian bertujuan untuk menunjukkan apakah program berjalan sesuai yang diharapkan dan menemukan kesalahan dari program sebelum program siap untuk digunakan. Pengujian dilakukan dengan berbagai macam cara, setelah dilakukan pengujian dan hasilnya memuaskan barulah program siap digunakan. Proses pengujian mempunyai dua tujuan, yaitu (Sommerville, 2011):

1. Untuk mendemonstrasikan kepada pengembang dan pengguna bahwa perangkat lunak sudah memenuhi kebutuhan.
2. Untuk mencari kesalahan dari perangkat lunak yang tidak sesuai dengan kebutuhan yang telah ditetapkan sebelumnya.

2.7.1 Performance test

Menurut pressman *Performance test* adalah sebuah *test* pada perangkat lunak yang bertujuan untuk menguji kinerja perangkat lunak dalam sistem yang terintegrasi, *performance test* melibatkan semua proses *testing* pada kinerja sistem yang di uji (Pressman, 2001). Pada sistem yang berbasis bisnis masalah performa dapat memberikan banyak konsekuensi berupa kerugian penjualan, penurunan produktivitas, penurunan reputasi dan masalah keuangan (Subraya dan Subrahmanya, 2000).

Performance testing digunakan untuk menentukan *responsiveness*, *throughput*, *reliability* dan *scalability* pada saat sistem digunakan (Patil, 2012). *Responsiveness* atau waktu respon digunakan untuk mengetahui respon sistem terhadap permintaan pengguna. *Throughput* merupakan jumlah unit kerja yang dapat ditangani pada waktu tertentu (hit per detik atau permintaan per detik). *Reliability* merupakan kemampuan aplikasi untuk menjalankan fungsinya dengan baik. *Scalability* adalah kemampuan aplikasi untuk menangani pekerjaan tanpa mempengaruhi performa aplikasi (Meier et al., 2016). Lansdale dan Ormerod dalam *Understanding interfaces* yang dikutip dari *Web application performance testing* oleh J Shaw menjelaskan, bahwa pengguna aplikasi *web* sangat sensitif terhadap *delay* respon waktu yang dihasilkan pada setiap transaksi dan umumnya jika *delay* lebih dari 10 detik dapat mengganggu aktifitas pengguna aplikasi *web* (Shaw, 2000). Sedangkan menurut patil dari banyak *survey* yang dilakukan, respon aplikasi dalam tiap transaksi tidak boleh melebihi 8 detik pada waktu beban normal dan beban maksimal (patil, 2012).

Penelitian oleh *keynote system* menunjukkan bahwa setiap wilayah akan mempunyai koneksi yang berbeda. Maka dari itu disarankan untuk melakukan

pengujian pada waktu dan hari yang berbeda-beda. Setiap pengguna aplikasi *web* mempunyai waktu respon yang berbeda-beda berdasarkan faktor-faktor yang berbeda. Berikut adalah faktor-faktor yang mempengaruhi waktu respon, yaitu (Menasce, 2002):

1. Internet Service Provider (ISP) yang digunakan oleh penguji.
2. Internet Service Provider (ISP) yang digunakan oleh pengguna.
3. Bandwidth dari Internet Service Provider (ISP) yang digunakan oleh pengguna.
4. Delay yang didapatkan ketika pada website terdapat pihak ketiga (iklan).

Ada banyak sekali macam-macam pengujian performa. Microsoft menyebutkan 4 pengujian performa untuk aplikasi berbasis *website*, yaitu (Meier et al., Farre, Bansode, Barber, Rea. 2016):

1. *Performance test*

Dilakukan untuk menentukan atau mengevaluasi kecepatan (kecepatan sistem merespons permintaan *user*), *scalability* (kemampuan sistem untuk menangani penambahan data) dan *stability* (kestabilan sistem pada saat digunakan).

2. *Load testing*

Untuk mengetahui kinerja sistem pada saat normal dan pada saat digunakan. *Load testing* memungkinkan untuk mengukur waktu respons, keluaran yang dihasilkan, dan tingkat pemanfaatan sumber daya, dan untuk mengidentifikasi titik maksimal pada sistem, dengan asumsi bahwa titik maksimal terjadi di bawah kondisi beban maksimal yang ditentukan.

3. *Stress testing*

Kemampuan sistem untuk dapat diakses banyak *user* sekaligus dalam 1 waktu. Tujuan dari *Stress testing* adalah untuk menemukan *bug* aplikasi yang muncul pada saat sistem dalam kondisi beban tinggi. Bug dapat mencakup hal-hal seperti masalah sinkronisasi, kecepatan, dan kebocoran memori. *Stress testing* memungkinkan untuk mengidentifikasi kelemahan sistem, dan menunjukkan bagaimana aplikasi berperilaku di bawah kondisi beban *ekstrem*.

4. *Capacity testing*

untuk menentukan seberapa besar data yang dapat disimpan. Pengujian kapasitas dilakukan dalam hubungannya dengan perencanaan kapasitas yang digunakan untuk merencanakan peningkatan ukuran volume data atau *user* yang memakainya.

2.7.2 Black-Box Testing

Pengujian *Black-Box* difokuskan pada kebutuhan fungsional perangkat lunak, pengujian *Black-Box* memungkinkan pengembang untuk mengatur kondisi *input* untuk melaksanakan semua fungsional program. *Blackbox* dirancang untuk mengungkap kesalahan, pengujian *black-box* digunakan untuk memperlihatkan bahwa fungsi-fungsi perangkat lunak dapat beroperasi, bahwa input diterima

dengan baik dan output dihasilkan dengan tepat, dan integritas informasi eksternal (seperti file data)(Nidhra and Dondeti, 2012). Pengujian *Black-Box* bertujuan untuk menemukan eror seperti:

1. Fungsi yang tidak bekerja atau fungsi yang hilang.
2. Kesalahan *interface*.
3. Kesalahan pada struktur data (pengaksesan basis data).
4. Kesalahan kinerja program.
5. Kesalahan inisiasi dan *terminasi*.

2.7.3 White-Box Testing

Pengujian *White-Box* adalah sebuah metode pengujian yang didasarkan pada pengecekan terhadap detail perancangan, menggunakan struktur *control* dari desain program secara *procedural* untuk membagi pengujian ke dalam beberapa kasus pengujian(Sommerville, 2011). Setiap representasi desain *procedural* yang berupa *flowchart* dapat diterjemahkan ke dalam *flowgraph*. *flowgraph* bertujuan untuk mengetahui kompleksitas dari program yang sudah dibuat.

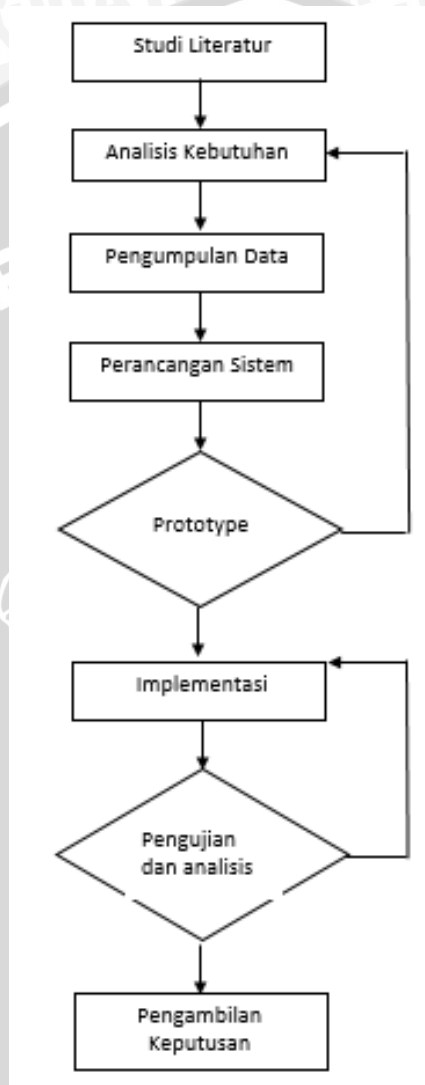
2.8 Rational Performance tester

Rational Performance tester (RPT) adalah salah satu kakas bantu pengujian yang dikembangkan oleh IBM yang memiliki kelebihan yaitu dapat melakukan penjadwalan pengujian, dapat merekam pengujian dan menampilkan hasil pengujian secara *realtime* serta berbasis *eclipse* (IBM-Team, 2008). *Rational Performance tester* mengukur waktu respons dengan mensimulasikan aksi pada sistem dengan beberapa pengguna sekaligus secara bersamaan. Pada RPT ada 6 tipe pengujian yaitu *Citrix Test*, *HTTP Test*, *SAP Test*, *Service Test*, *Socket Test* dan *TN3270 Test*. *Rational Performance tester* ini bisa di unduh secara gratis melalui situs resmi IBM namun hanya menyediakan pengujian *HTTP Test* dan *Socket Test*, dan jika ingin menggunakan aplikasi ini secara *full* versi maka diharuskan untuk membayar.

Untuk melakukan pengujian menggunakan RPT pengguna harus melakukan *record* menjalankan fitur sistem yang ingin diuji. Setelah melakukan proses *record* dilanjutkan dengan melakukan *performance schedule* untuk mengetahui respon tiap fitur yang diuji dengan parameter yang ada. RPT akan langsung mengeluarkan hasil ketika selesai menjalankan *performance schedule*. Hasil yang dikeluarkan berupa respons waktu dari setiap fitur yang ada pada sistem dengan banyak perbandingan. Data dari hasil pengujian ditampilkan dalam bentuk tabel, grafik dan juga diagram sehingga memudahkan pengguna untuk memahami hasil keluaran dari pengujian menggunakan RPT ini. Dengan kemudahan dalam menampilkan semua hasil respon maka kita tidak perlu lagi menghitung waktu respon rata-rata tiap fitur.

BAB 3 METODOLOGI

Dalam bab ini akan dibahas metodologi atau langkah-langkah yang digunakan dalam penelitian untuk membuat aplikasi pemesanan sayur keliling. Langkah-langkah tersebut terdiri dari: Metode Pengembangan, studi literatur, pengumpulan data, analisa kebutuhan, perancangan sistem, implementasi, pengujian dan analisis, serta pengambilan kesimpulan.



Gambar 3. 1 Diagram Alir Metode Penelitian

Pada gambar 3.1 diagram alir metode penelitian menjelaskan langkah pengerjaan dalam penelitian ini. Langkah mengerjakan penelitian melalui beberapa fase yaitu studi literatur, analis kebutuhan, pengumpulan data, perancangan sistem, prototype(pada fase ini, jika pengguna merasa belum puas maka akan mengulang fase analisis kebutuhan dan jika sudah cukup akan melanjutkan fase berikutnya), implementasi, pengujian dan analisis(jika pengguna merasa belum puas maka akan mengulang fase implementasi jika sudah cukup

puas dapat melanjutkan fase selanjutnya) dan yang terakhir fase pengambilan keputusan.

3.1 Studi Literatur

Studi literatur dilakukan bertujuan untuk mempelajari serta memahami konsep-konsep sistem agar ketika dilakukan perancangan tidak terlalu mengalami kendala. Pada tahap studi literatur ini mempelajari mengenai teori-teori yang digunakan dalam pengerjaan skripsi.

Dalam hal ini, studi literatur yang digunakan adalah:

1. Rekayasa Perangkat Lunak.
2. Penjual Sayur Keliling.
3. Pemesanan.
4. *Prototyping Model*
5. Pengujian Perangkat Lunak.
6. *Rational Performance tester*
7. *Codeigniter*.
8. DBMS *MySQL*.

3.2 Analisa Kebutuhan

Analisis kebutuhan bertujuan untuk mendapatkan semua kebutuhan yang diperlukan dari sistem yang akan dibangun. Analisis kebutuhan dilakukan dengan cara mengidentifikasi siapa saja aktor yang akan terlibat kedalam sistem dan semua kebutuhan (*requirements*) sistem. Analisis dilakukan untuk mengetahui kondisi lapangan yang ada sehingga dapat diketahui implementasi perangkat lunak yang akan digunakan.

Tahap analisis kebutuhan terdiri dari 4 langkah, yaitu: melakukan penjabaran tentang gambaran umum aplikasi atau sistem, melakukan proses identifikasi aktor yang terlibat di dalam aplikasi atau sistem, membuat daftar kebutuhan pengguna menggunakan pemodelan *diagram use case* dan menganalisa kebutuhan non-fungsional. *Use case* merupakan representasi relasi antara pengguna dengan sistem. Analisis kebutuhan digunakan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh aplikasi atau sistem agar dapat memenuhi kebutuhan pengguna.

3.3 Pengumpulan Data

Pengumpulan data dalam penelitian ini bertujuan untuk mendata harga-harga sayur di pasaran yang dibutuhkan sebagai bahan dalam proses pengerjaan penelitian. Teknik pengumpulan data yang digunakan adalah dengan melakukan observasi dan wawancara untuk mendapatkan daftar harga jual dan harga beli dari pedagang di pasar dan penjual sayur keliling. Observasi dan wawancara dilakukan

di beberapa penjual sayur keliling dan beberapa penjual sayur di pasar yang berada di sekitar siguragura Kota Malang. Tahap observasi dilakukan pada bulan September 2015 di daerah Siguragura Kota Malang.

3.4 Perancangan Sistem

Perancangan sistem dilakukan setelah semua kebutuhan yang dibutuhkan telah diperoleh. Perancangan sistem dimulai dari hasil kebutuhan sistem dalam bentuk *diagram use case*, kemudian diidentifikasi dan di *model* kan dengan *diagram sequence*. Setelah itu, perancangan *class* dan relasinya yang akan di *model* kan dalam *diagram class*.

Perancangan basis data dilakukan untuk mengetahui representasi data yang akan digunakan oleh sistem. Tahap ini banyak membahas tentang bagaimana mengolah data dan bagaimana sistem dapat bekerja. Data-data yang digunakan akan disimpan di dalam *database MySQL*. Aplikasi yang dibuat berbasis *web* dan *webview* sehingga dapat dengan mudah dibuka pada *smartphone* tanpa membuka *browser*. Menerapkan *prototype Model*, sesuai *use case* yang ada maka dibuatlah *prototype* aplikasi untuk memudahkan *user* melihat gambaran aplikasi yang akan dibuat. Dengan metode *prototype* langkah kerja yang dilakukan akan lebih mudah untuk mengetahui apa yang diinginkan *user*.

3.5 Prototype

Setelah proses analisis kebutuhan dan pengumpulan data selesai selanjutnya proses pembuatan prototype dengan membuat perancangan sementara yang berpusat pada penyajian kepada pengguna. Tidak semua fitur dibuat pada proses pembuatan prototype ini, hanya beberapa fitur untuk mewakili gambaran umum sistem saja. Ketika prototype selesai dibuat maka akan diberikan kepada pengguna, apakah para pengguna merasa cocok atau masih kurang. Jika dirasa belum cukup memenuhi apa yang diinginkan pengguna maka akan diulang kembali proses analisis kebutuhan dan pengumpulan data, namun jika telah sesuai maka dilanjutkan pada proses implementasi sistem.

3.6 Implementasi

Implementasi sistem dilakukan berdasarkan pada hasil perancangan sistem sebelumnya. Aplikasi dibuat berbasis *web* dengan menggunakan bahasa pemrograman PHP dengan *framework Codeigniter*, pembuatan *webview* untuk *smartphone* menggunakan *Android Studio*. Untuk penyimpanan data menggunakan *database MySQL*. Implementasi dilakukan dengan membuat *interface* yang sesuai untuk 3 aktor yaitu halaman *admin*, halaman pedagang dan juga halaman pelanggan atau pembeli. Pembuatan *interface* menggunakan *bootstrap* sebagai pelengkap *interface*. Proses pengkodean PHP dibantu dengan aplikasi *sublime*, sedangkan untuk akses *localhost* menggunakan *xampp*.

3.7 Pengujian dan Analisis

Setelah sistem telah benar-benar terintegrasi, diperlukan untuk menguji sistem untuk mengetahui apakah sistem telah berjalan sesuai yang diharapkan. Dengan adanya proses pengujian maka akan diketahui kekurangan dari sistem, selanjutnya dengan melakukan analisa dapat diketahui kekurangan dari sistem sehingga akan lebih mudah untuk memperbaiki sistem yang memiliki kekurangan, sehingga harapan dari pengguna untuk mendapat aplikasi yang diinginkan tercapai.

Metode Pengujian dilakukan dengan *whitebox*, *blackbox*, dan uji performa menggunakan *Rational Performance tester*. Pengujian *White-Box* dilakukan dengan menggunakan teknik *Basis Path testing*. Teknik pengujian ini dilakukan dengan menggunakan rancangan atau kode program sebagai dasar untuk membuat *flow graph*. Setelah itu, berdasarkan *flowgraph* yang telah dibuat akan ditentukan kompleksitas siklomatik serta himpunan basis dari jalur-jalur independen secara linier.

Skenario pada pengujian *Black-Box* dilakukan dengan menjalankan sistem sesuai dengan fungsi-fungsi internal yang akan dijabarkan pada bab analisis dan perancangan sub bab *diagram use case*. Setelah itu dilakukan pencatatan dalam bentuk tabel apakah hasil kebutuhan *fungsional* berhasil terpenuhi oleh sistem (*valid*) atau tidak (*tidak valid*).

Rational Performance tester merupakan tool dari IBM. Dengan menggunakan tool ini akan mempermudah mengetahui performa atau kinerja dari sistem yang dibangun. Pengujian dilakukan dengan merekam jalannya sistem melalui aplikasi *Rational Performance tester*, setelah proses merekam selesai proses pengujian dilanjutkan dengan menjalankan rekaman. Keluaran dari rekaman yang diputar yaitu respons waktu dari tiap fitur yang dijalankan. Fitur yang akan diuji yaitu semua fitur yang ada dan dapat berjalan pada sistem.

3.8 Evaluasi Sistem

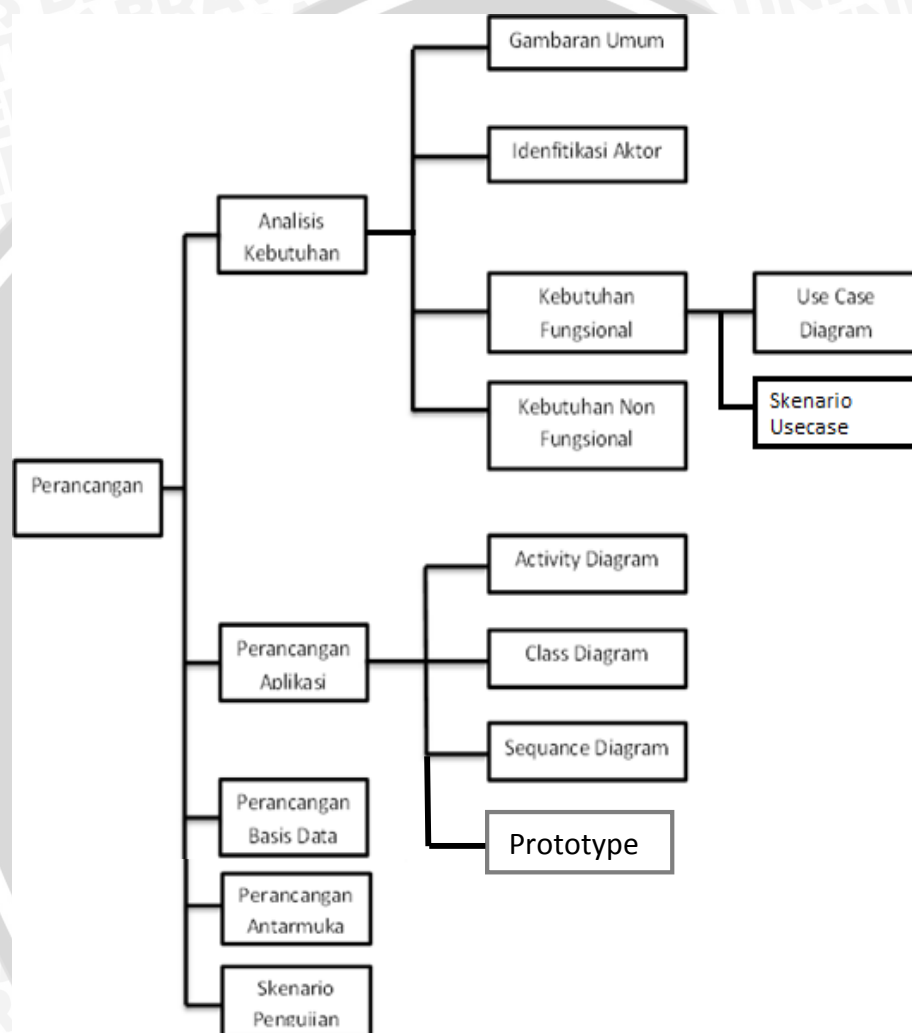
Evaluasi sistem dilakukan setelah semua tahapan perancangan perangkat lunak, implementasi perangkat lunak, dan pengujian perangkat lunak telah selesai dilakukan. Jika sistem sudah sesuai dengan kebutuhan maka proses pembuatan aplikasi telah selesai. Namun Jika aplikasi masih belum sesuai maka akan dilakukan kembali proses implementasi sistem.

3.9 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan perangkat lunak, implementasi perangkat lunak, dan pengujian perangkat lunak telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari pengerjaan dokumentasi adalah saran yang bertujuan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan perangkat lunak lebih lanjut.

BAB 4 ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini akan menjelaskan mengenai perancangan sistem aplikasi pemesanan sayur keliling. Perancangan sistem ini terdiri dari analisis kebutuhan perangkat lunak dan perancangan perangkat lunak. Pada gambar 4.1 berikut terdapat diagram pohon dalam perancangan aplikasi pemesanan sayur keliling dengan menggunakan metode pengembangan *prototype*.



Gambar 4.1 Diagram Pohon Perancangan

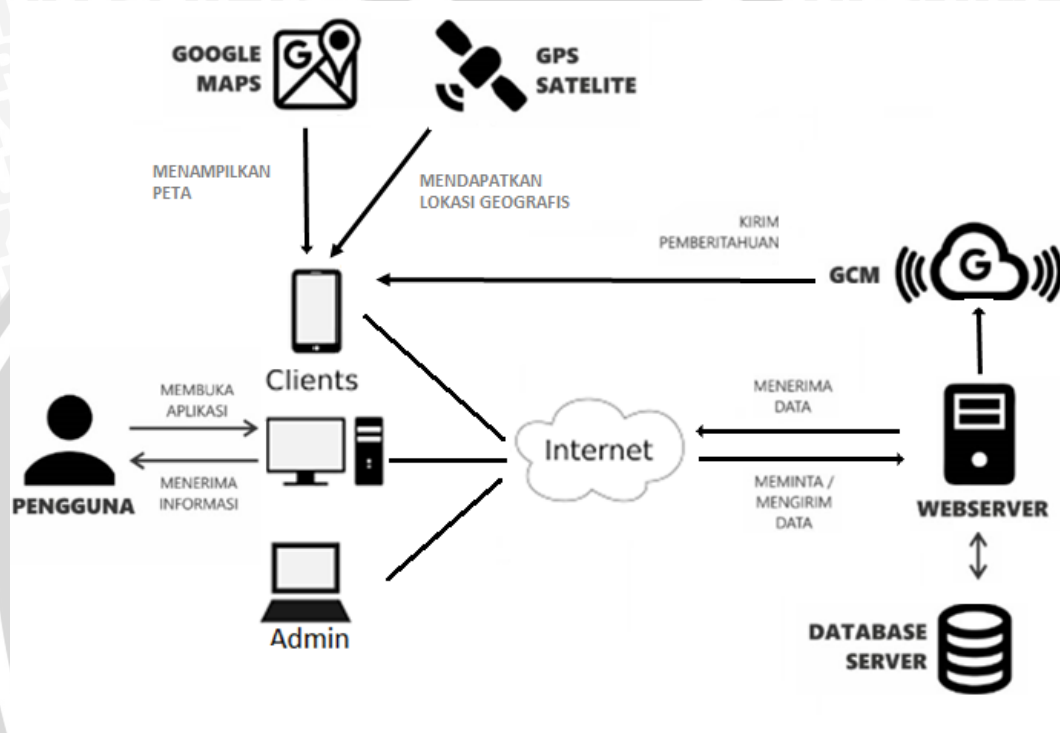
4.1 Analisa Kebutuhan

Tahap analisis kebutuhan ini akan melakukan identifikasi kebutuhan aplikasi pemesanan sayur. Terdapat beberapa tahap dalam analisis kebutuhan ini, antara lain: gambaran aplikasi, identifikasi aktor, analisis kebutuhan *fungsional*, dan analisis kebutuhan *non-fungsional* dengan menggunakan *use case diagram*. Analisis kebutuhan ini bertujuan untuk memberikan gambaran secara jelas

mengenai hal-hal yang harus disediakan oleh sistem demi memenuhi kebutuhan pengguna.

4.1.1 Gambaran Umum

Gambaran aplikasi merupakan bentuk representasi desain arsitektur sistem yang dibuat secara umum. Tahap ini merupakan tahap awal perancangan sistem yang akan dibangun. Gambaran aplikasi secara umum akan ditunjukkan pada gambar 4.2 berikut.



Gambar 4.2 Gambaran Umum Sistem

Gambar 4.2 menjelaskan bahwa ada 2 tipe pengguna dalam sistem yaitu *admin* dan *client*, di sini *client* sebagai pedagang atau pembeli. *Admin* hanya bisa mengakses melalui pc atau *desktop* sedangkan *client* dapat mengakses melalui PC ataupun *smartphone*. *Database* sistem menggunakan *MySQL* yang berada pada server, sehingga untuk mengaksesnya diperlukan koneksi *internet*.

Pada aplikasi *client* terdapat 2 pengguna yaitu pedagang dan pembeli. Untuk menjalankan aplikasi diperlukan untuk *login* terlebih dahulu, jika belum memiliki akun disarankan untuk mendaftar terlebih dahulu. Pedagang dan pembeli memiliki halaman dan menu yang berbeda. Pada pedagang setelah *login* akan menampilkan beranda pedagang dan menampilkan menu kelola barang untuk menambah dan melihat barang serta menu lihat pesanan. Pada menu lihat pesanan pedagang dapat mengonfirmasi barang yang bisa dipesan.

Sedangkan pada halaman pelanggan akan muncul beranda pelanggan yang menampilkan daftar pesanan pelanggan tersebut. Lalu terdapat menu pilih pedagang untuk mulai memesan barang dagangan. Setelah memilih pedagang

pembeli dapat memilih barang dagangan yang dijual oleh pedagang yang dipilih dan memasukkannya ke keranjang belanja. Jika barang yang diinginkan tidak ada, pelanggan dapat memilih menu pesanan khusus. Halaman *admin* berfungsi untuk mengelola pedagang, yaitu menambah, menghapus dan melihat *list* pedagang. Pelanggan yang belum mendaftar kedalam sistem dapat memilih menu register terlebih dahulu.

Pada gambar 4.2 terdapat google maps dan gps satellite yang berfungsi untuk menampilkan peta dan mendapatkan lokasi geografis dimana pengguna berada. Sedangkan GCM berfungsi untuk proses pengiriman notifikasi atau pemberitahuan pada perangkat mobile. Sedangkan web server berfungsi untuk menjembatani para pengguna dengan sistem untuk menerima data dan mengirim data kedalam database serta untuk mengirim perintah pemberitahuan pada gcm.

Aplikasi ini bersifat *responsive* dapat dibuka di *browser* PC dan *smartphone*. Aplikasi ini juga dibuat menjadi *webview* sehingga pengguna dapat menginstal aplikasi dan membuka aplikasi tanpa harus membuka *browser* terlebih dahulu.

4.1.2 Identifikasi Aktor

Pada tahap ini bertujuan untuk melakukan identifikasi terhadap aktor yang berhubungan dengan aplikasi yang akan dibangun. Terdapat tiga aktor yang berhubungan dengan aplikasi. Penjelasan masing-masing aktor dapat dilihat pada tabel 4.1 berikut.

Tabel 4.1 Identifikasi Aktor

Aktor	Deskripsi
<i>Admin</i>	<i>Admin</i> dapat melihat <i>list</i> pedagang, menghapus dan menambah pedagang.
Pedagang	Pedagang dapat melihat <i>list</i> pesanan pelanggan, dapat juga menambah barang dagangan dan mengonfirmasi barang dagangan yang dapat dipesan pelanggan.
Pelanggan	Pelanggan atau pembeli dapat melihat <i>list</i> pesannya sendiri dan dapat mengonfirmasi apakah barang dagangan sudah dikirim atau belum. Pelanggan juga dapat memilih pedagang dan memilih barang yang ingin dipesan.

4.1.3 Analisis Kebutuhan *Fungsional*

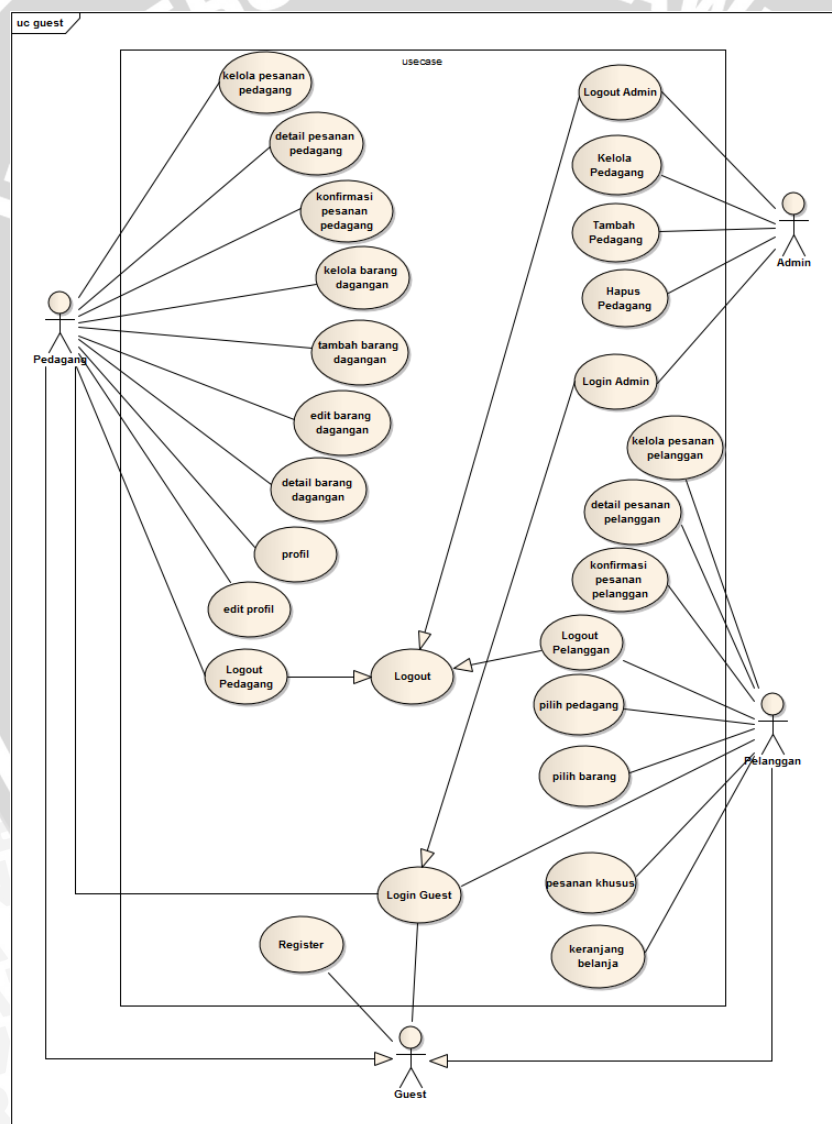
Pada analisis kebutuhan *fungsional* ini terdiri dari spesifikasi kebutuhan *fungsional* perangkat lunak berupa kebutuhan *fungsional*, aktor yang menggunakan sistem, dan nama *use case* yang dapat dilihat pada tabel 4.2 dengan penomoran menggunakan SRS (*Software Requirement Specification*).

Tabel 4. 2 Kebutuhan *Fungsional* Pedagang

Nomor SRS	Kebutuhan <i>Fungsional</i>	Nama <i>Use case</i>
SRS_01_001	Aplikasi dapat menampilkan data pedagang	Kelola Pedagang
SRS_01_002	Aplikasi harus menyediakan fitur untuk menambah data pedagang	Tambah Pedagang
SRS_01_003	Aplikasi harus menyediakan fitur untuk menghapus data pedagang	Hapus Pedagang
SRS_01_004	Aplikasi dapat menampilkan pesanan-pesanan dari pelanggan	Kelola Pesanan Pedagang
SRS_01_005	Aplikasi dapat menampilkan detail pesanan pelanggan	Detail Pesanan Pedagang
SRS_01_006	Aplikasi mampu mengkonfirmasi dan mengirimkan notifikasi	Konfirmasi Pesanan Pedagang
SRS_01_007	Aplikasi dapat menampilkan barang dagangan pelanggan	Kelola Barang Dagangan
SRS_01_008	Aplikasi dapat menambah barang dagangan	Tambah Barang Dagangan
SRS_01_009	Aplikasi mampu menampilkan detail barang dagangan	Detail Barang Dagangan
SRS_01_010	Aplikasi mampu merubah data barang dagangan	Edit Barang Dagangan
SRS_01_011	Aplikasi dapat menampilkan data	Profil
SRS_01_012	Aplikasi mampu merubah data pedagang	Edit Profil
SRS_01_013	Aplikasi dapat menampilkan pesanan pelanggan sebelumnya	Kelola Pesanan Pelanggan
SRS_01_014	Aplikasi dapat menampilkan detail pesanan pelanggan	Detail Pesanan Pelanggan
SRS_01_015	Aplikasi mampu mengkonfirmasi dan mengirimkan notifikasi	Konfirmasi Pesanan Pelanggan
SRS_01_016	Aplikasi menyediakan fitur memilih pedagang	Pilih Pedagang
SRS_01_017	Aplikasi mampu menampilkan daftar barang dagangan	Pilih Barang

SRS_01_018	Aplikasi mampu menampilkan form pesanan	Pesanan Khusus
SRS_01_019	Aplikasi mampu menampilkan barang belanjaan	Keranjang Belanja
SRS_01_020	Aplikasi harus menyediakan fitur untuk daftar pada aplikasi	Register
SRS_01_021	Aplikasi harus menyediakan fitur untuk login	Login
SRS_01_022	Aplikasi harus menyediakan fitur untuk logout	Logout

4.1.3.1 Use case Diagram



Gambar 4.3 Usecase Diagram Pemesanan Sayur Keliling

Use case diagram merupakan *diagram* yang menggambarkan dengan jelas kebutuhan dan implementasi dari sistem berupa beberapa *use case* yang dilakukan oleh aktor dalam sistem yang akan dibangun. Setiap *use case* akan dijelaskan lebih detail pada skenario *use case*. *Use case* menggambarkan hubungan antara aktor dan sistem. Sedangkan *Use case* skenario untuk mendeskripsikan *use case*, terdiri dari kondisi awal, reaksi aktor, reaksi sistem, dan kondisi akhir dari sistem.

Use case diagram yang ditunjukkan pada gambar 4.3 merupakan *use case diagram* pada sistem pemesanan sayur keliling ini. Pada *usecase* ditunjukkan ada 3 orang aktor yaitu admin, pedagang dan pelanggan. *Use case diagram* dibuat sesuai dengan kebutuhan *fungsi*ional agar semua kebutuhan aktor dengan sistem yang dibangun dapat terpenuhi.

4.1.3.2 Skenario Use case

Dalam skenario *use case* berisi penjelasan detail dari masing-masing *use case* yang ada pada tabel kebutuhan fungsional 4.2 atau gambar 4.3. Pada skenario *use case*, diberikan nama *use case*, aktor yang terlibat, tujuan, deskripsi, langkah-langkah *use case*, kondisi awal yang harus dipenuhi (*pre-condition*), kondisi akhir yang diharapkan (*post-condition*), aliran utama *use case*, dan aliran alternatif *use case*. Skenario *use case* dapat dilihat pada tabel 4.3 sampai tabel.

1. Skenario Use case Admin

Pada skenario ini menjelaskan skenario kelola pesanan pedagang, skenario tambah pedagang dan juga hapus pedagang. Untuk skenario kelola pesanan pedagang dapat dilihat pada tabel 4.5, skenario tambah pedagang dapat dilihat pada tabel 4.6 dan skenario hapus pedagang dapat dilihat pada tabel 4.7.

Tabel 4.3 Skenario Use case Kelola Pedagang.

Nomor SRS	SRS_01_001
Nama Use case	Kelola Pedagang
Aktor	Admin
Tujuan	Untuk menampilkan pedagang yang telah terdaftar di dalam sistem
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana <i>admin</i> dapat melihat <i>list</i> atau daftar pedagang yang terdaftar
Pre-Condition (kondisi awal)	<i>Admin</i> harus <i>login</i> terlebih dahulu untuk bisa melihat menu kelola pedagang.
Post-Condition (kondisi akhir)	Sistem akan menampilkan data pedgag yang terdaftar dalam sistem.
Skenario Utama	

Aksi dari Aktor	Respons dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman kelola pedagang.

Tabel 4.4 Skenario Use case Tambah Pedagang

Nomor SRS	SRS_01_002
Nama Use case	Tambah Pedagang
Aktor	Admin
Tujuan	Untuk menambah pedagang yang ingin bergabung dalam sistem
Deskripsi	Use case ini menjelaskan bagaimana admin menambahkan data pedagang ke dalam sistem.
Pre-Condition (kondisi awal)	Admin harus <i>login</i> terlebih dahulu untuk bisa menambahkan data pedagang.
Post-Condition (kondisi akhir)	Setelah menambahkan data pedagang sistem akan menampilkan data semua pedagang.
Skenario Utama	
Aksi dari Aktor	Respons dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	4. Memeriksa status <i>login</i> .
	5. Menampilkan halaman kelola pedagang.
6. Memilih fitur tambah pedagang.	7. Menampilkan <i>form</i> tambah pedagang.
8. Menginputkan data pedagang.	9. Menampilkan menu kelola pedagang.
Skenario Alternatif	
Tidak mengisi data lengkap pada <i>form</i> .	Muncul peringatan untuk mengisi semua data.

Tabel 4.5 Skenario Use case Hapus Pedagang

Nomor SRS	SRS_01_003
Nama Use case	Hapus Pedagang
Aktor	Admin
Tujuan	Untuk menghapus data dari pedagang.

Deskripsi	<i>Use case</i> ini menjelaskan bagaimana <i>admin</i> menghapus data dari pedagang dari <i>database</i> melalui sistem.
Pre-Condition (kondisi awal)	<i>Admin</i> harus <i>login</i> dulu sebelum memilih fitur hapus yang berada pada menu kelola pedagang.
Post-Condition (kondisi akhir)	Menampilkan menu kelola sayur yang menampilkan data pedagang.
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman kelola pedagang
4. Memilih data pedagang yang akan dihapus dan pilih fitur hapus.	5. Menampilkan pesan konfirmasi apakah benar data ingin dihapus.
6. Tekan setuju	7. Data dihapus
	8. Menampilkan menu kelola pedagang.
Skenario Alternatif	
	Menampilkan pesan konfirmasi apakah benar data ingin dihapus.
Memilih tidak setuju.	Kembali ke menu kelola pedagang.

2. Skenario *Use case* Pedagang

Pada skenario ini menjelaskan skenario kegiatan pedagang pada sistem yaitu skenario kelola pesanan pedagang yang dapat dilihat pada tabel 4.8, skenario detail pesanan pedagang pada tabel 4.9, skenario konfirmasi pesanan pedagang pada tabel 4.10, skenario kelola barang dagangan pada tabel 4.11, skenario tambah barang dagangan pada tabel 4.12, skenario detail barang dagangan pada tabel 4.13, skenario edi barang dagangan pada tabel 4.14, skenario profil pada tabel 4.15, skenario edit profil pada tabel 4.16

Tabel 4.6 Skenario *Use case* Kelola Pesanan Pedagang

Nomor SRS	SRS_01_004
Nama <i>Use case</i>	Kelola Pesanan Pedagang
Aktor	Pedagang
Tujuan	Agar pedagang bisa melihat pesanan dari para pelanggannya.

Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses melihat pesanan dari pelanggan.
Pre-Condition (kondisi awal)	Pedagang harus melakukan <i>login</i> ke dalam aplikasi <i>client</i> .
Post-Condition (kondisi akhir)	Menampilkan halaman lihat pesanan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pedagang.

Tabel 4.7 Skenario *Use case* Detail Pesanan Pedagang

Nomor SRS	SRS_01_005
Nama <i>Use case</i>	Detail Pesanan Pedagang
Aktor	Pedagang
Tujuan	Untuk menampilkan detail pesanan pelanggan
Deskripsi	<i>Use case</i> ini menunjukkan bagaimana proses pedagang menampilkan halaman detail pesanan pelanggan.
Pre-Condition (kondisi awal)	Pedagang harus <i>login</i> terlebih dahulu untuk menampilkan halaman detail pesanan pelanggan.
Post-Condition (kondisi akhir)	Menampilkan halaman detail pesanan pelanggan
Skenario Utama	
Aksi Dari Aktor	Aksi Dari Aktor
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pedagang.
4. Memilih menu detail pesanan pelanggan.	5. Menampilkan detail pesanan pelanggan.

Tabel 4.8 Skenario *Usecase* Konfirmasi Pesanan Pedagang

Nomor SRS	SRS_01_006
Nama <i>Use case</i>	Konfirmasi Pesanan Pedagang
Aktor	Pedagang

Tujuan	Untuk mengkonfrmasi pesnan pelanggan
Deskripsi	<i>Usecase</i> ini menjelaskan proses menampilkan konfirmasi pesanan oleh pedagang.
Pre-Condition (kondisi awal)	Pedagang harus melakukan <i>login</i> untuk bisa mengkonfirmasi pesanan.
Post-Condition (kondisi akhir)	Menampilkan halaman kelola pesanan.
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih pesanan	
5. Mengkonfirmasi pesanan	6. Menampilkan kelola pesanan

Tabel 4.9 Skenario *Use case* Kelola Barang Dagangan

Nomor SRS	SRS_01_007
Nama <i>Use case</i>	Kelola Barang
Aktor	Pedagang
Tujuan	Untuk menampilkan <i>list</i> barang dagangan
Deskripsi	<i>Use case</i> ini menjelaskan proses menampilkan kelola barang
Pre-Condition (kondisi awal)	Pedagang harus melakukan <i>login</i> untuk bisa membuka halaman kelola barang.
Post-Condition (kondisi akhir)	Menampilkan halaman kelola barang.
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih menu kelola barang.	5. Menampilkan halaman kelola barang.
Skenario Utama	

Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih menu kelola barang.	5. Menampilkan menu kelola barang.
6. Memilih fitur tambah barang.	7. Menampilkan <i>form</i> tambah barang.
8. Mengisi data barang dagangan	9. Data ditambahkan
	10. Menampilkan halaman kelola barang
Skenario Alternatif	
Tidak mengisi <i>form</i> dengan benar	Menampilkan konfirmasi isi data dengan benar.

Tabel 4.10 Usecase Tambah Barang Dagangan

Nomor SRS	SRS_01_008
Nama <i>Use case</i>	Tambah barang dagangan
Aktor	Pedagang
Tujuan	Untuk menambah barang dagangan
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses menambah barang dagangan.
Pre-Condition (kondisi awal)	Pedagang harus melakukan <i>login</i> terlebih dahulu untuk dapat menambah barang.
Post-Condition (kondisi akhir)	Akan menampilkan <i>list</i> barang dagangan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih menu kelola barang dagangan	
5. Memilih button tambah barang	6. Menampilkan form tambah barang
7. Mengisi form dan menekan tombol submit	8. Menampilkan halaman kelola barang

Tabel 4. 11 Usecase Detail Barang Dagangan

Nomor SRS	SRS_01_009
-----------	------------

Nama Use case	Detail barang dagangan
Aktor	Pedagang
Tujuan	Untuk menampilkan detail barang yang dipilih
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses menampilkan halaman detail barang
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat melihat detail barang.
Post-Condition (kondisi akhir)	Akan menampilkan detail barang
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih menu kelola barang	
5. Memilih barang	6. Menampilkan detail barang

Tabel 4.12 Usecase Edit Barang Dagangan

Nomor SRS	SRS_01_0010
Nama Use case	Edit barang dagangan
Aktor	Pedagang
Tujuan	Untuk merubah data barang dagangan
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses merubah data barang dagangan
Pre-Condition (kondisi awal)	Pedagang harus melakukan <i>login</i> terlebih dahulu untuk dapat merubah data barang.
Post-Condition (kondisi akhir)	Akan menampilkan <i>list</i> barang dagangan dari halaman kelola barang dagangan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih kelola barang	

5. Memilih barang	
6. Memilih button edit barang	7. Menampilkan form edit

Tabel 4.13 Usecase Profil

Nomor SRS	SRS_01_011
Nama Use case	Profil
Aktor	Pedagang
Tujuan	Untuk menampilkan data pedagang
Deskripsi	Use case ini menjelaskan bagaimana proses menampilkan halaman profil
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat melihat data pedagang.
Post-Condition (kondisi akhir)	Akan menampilkan data pedagang
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pelanggan.
4. Memilih menu profil	5. Menampilkan data profil pedagang

Tabel 4.14 Usecase Edit Profil

Nomor SRS	SRS_01_012
Nama Use case	Edit profil
Aktor	Pedagang
Tujuan	Untuk merubah profil data pedagang
Deskripsi	Use case ini menjelaskan bagaimana proses merubah data pedagang
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat merubah data.
Post-Condition (kondisi akhir)	Akan menampilkan halaman kelola pesanan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem

1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pedagang.
4. Memilih menu profil	5. Menampilkan data pedagang
6. Memilih edit data	7. Menampilkan form edit data

3. Skenario *Use case* Pelanggan

Pada skenario ini menjelaskan skenario kegiatan pelanggan pada sistem yaitu skenario kelola pesanan pelanggan yang dapat dilihat pada tabel 4.17, skenario detail pesanan pelanggan pada tabel 4.18, skenario konfirmasi pesanan pelanggan pada tabel 4.19, skenario pilih pedagang pada tabel 4.20, skenario pilih barang pada tabel 4.21, skenario pesanan khusus pada tabel 4.22, skenario keranjang belanja pada tabel 4.23.

Tabel 4.15 Skenario *Use case* Kelola Pesanan Pelanggan

Nomor SRS	SRS_01_013
Nama <i>Use case</i>	Kelola Pesanan Pelanggan
Aktor	Pelanggan
Tujuan	Untuk menampilkan <i>list</i> pesanan pelanggan
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses menampilkan halaman kelola pesanan pelanggan
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat melihat <i>list</i> barang.
Post-Condition (kondisi akhir)	Akan menampilkan <i>list</i> barang dagangan dari halaman kelola pesanan pelanggan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
7. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	8. Memeriksa status <i>login</i> .
	9. Menampilkan halaman beranda pelanggan.

Tabel 4. 16 Skenario *Use case* Detail Pesanan Pelanggan

Nomor SRS	SRS_01_014
Nama <i>Use case</i>	Detail Pesanan Pelanggan
Aktor	Pelanggan
Tujuan	Untuk menampilkan detail pesanan pelanggan

Deskripsi	<i>Use case</i> ini menunjukkan bagaimana proses pelanggan menampilkan halaman detail pesanan pelanggan.
Pre-Condition (kondisi awal)	Pelanggan harus <i>login</i> terlebih dahulu untuk menampilkan halaman detail pesanan pelanggan.
Post-Condition (kondisi akhir)	Menampilkan halaman detail pesanan pelanggan
Skenario Utama	
Aksi Dari Aktor	Aksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pelanggan.
4. Memilih menu detail pesanan pelanggan.	5. Menampilkan detail pesanan pelanggan.

Tabel 4. 17 Usecase konfirmasi pesanan pelanggan

Nomor SRS	SRS_01_015
Nama Use case	Konfirmasi pesanan pelanggan
Aktor	Pelanggan
Tujuan	Untuk mengkonfirmasi bahwa barang telah dikirim
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses mengkonfirmasi barang kiriman
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat mengkonfirmasi
Post-Condition (kondisi akhir)	Akan menampilkan halaman kelola pesanan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda pelanggan.
4. Memilih pesanan	5. Menampilkan detail pesanan
6. mengkonfirmasi	7. mengirim notifikasi konfirmasi

Tabel 4. 18 Skenario *Use case* Pilih Pedagang

Nomor SRS	SRS_01_016
Nama <i>Use case</i>	Pilih Pedagang
Aktor	Pelanggan
Tujuan	Memilih pedagang
Deskripsi	Menunjukkan proses pelanggan untuk memilih pedagang
Pre-Condition (kondisi awal)	Harus melakukan <i>login</i> untuk bisa memilih pedagang
Post-Condition (kondisi akhir)	Menampilkan barang dagangan pedagang
Skenario Utama	
Aksi Dari Aktor	Aksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pelanggan.
4. Memilih fitur tambah pedagang.	5. Menampilkan <i>list</i> pedagang.

Tabel 4. 19 *Usecase* Pilih Barang

Nomor SRS	SRS_01_017
Nama <i>Use case</i>	Pilih barang
Aktor	Pelanggan
Tujuan	Untuk menampilkan <i>list</i> barang dagangan
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses menampilkan halaman pilih barang
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat melihat <i>list</i> barang.
Post-Condition (kondisi akhir)	Akan menampilkan <i>list</i> barang dagangan dari halaman kelola pesanan pelanggan
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pelanggan.

4. Memilih fitur tambah pedagang.	5. Menampilkan <i>list</i> pedagang.
6. Memilih pedagang	7. Menampilkan barang dagangan pedagang.

Tabel 4. 20 Usecase Pesanan Khusus

Nomor SRS	SRS_01_018
Nama Use case	Pesanan khusus
Aktor	Pelanggan
Tujuan	Untuk menampilkan <i>form</i> pesanan khusus
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses memesan barang khusus
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat memesan khusus.
Post-Condition (kondisi akhir)	Akan menampilkan <i>form</i> pesanan khusus
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pelanggan.
4. Memilih fitur tambah pedagang.	5. Menampilkan <i>list</i> pedagang.
6. Memilih pedagang	7. Menampilkan barang dagangan pedagang.
8. Memilih menu pesanan khusus	9. Menampilkan halaman pesanan khusus

Tabel 4. 21 Usecase Keranjang Belanja

Nomor SRS	SRS_01_019
Nama Use case	Keranjang belanja
Aktor	Pelanggan
Tujuan	Untuk menampilkan keranjang belanja
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana proses menampilkan halaman keranjang belanja
Pre-Condition (kondisi awal)	Pelanggan harus melakukan <i>login</i> terlebih dahulu untuk dapat melihat keranjang belanja

Post-Condition (kondisi akhir)	Akan menampilkan keranjang belanja
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka halaman <i>login</i> dan melakukan <i>login</i> .	2. Memeriksa status <i>login</i> .
	3. Menampilkan halaman beranda untuk pelanggan.
4. Memilih fitur tambah pedagang.	5. Menampilkan <i>list</i> pedagang.
6. Memilih pedagang	7. Menampilkan barang dagangan pedagang.
8. Memilih keranjang belanja	9. Menampilkan keranjanga belanja

4. Skenario *Use case* Guest

Skenario ini menjelaskan skenario kegiatan guest untuk *login* dan register untuk mendaftar ke dalam sistem. Skenario register dapat dilihat pada 4.24, skenario *login* pada tabel 4.25 dan skenario *logout* pada tabel 4.26.

Tabel 4. 22 Skenario *Use case* Register.

Nomor SRS	SRS_01_020
Nama <i>Use case</i>	Register
Aktor	Pelanggan
Tujuan	Untuk mendaftar supaya bisa menggunakan aplikasi
Deskripsi	<i>Use case</i> ini menjelaskan bagaimana pelanggan melakukan register.
Pre-Condition (kondisi awal)	Membuka aplikasi lalu memilih fitur <i>register</i> .
Post-Condition (kondisi akhir)	Akan menampilkan halaman <i>login</i> setelah berhasil <i>register</i> .
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka aplikasi.	2. Menampilkan halaman <i>login</i> .
3. Memilih fitur <i>register</i> .	4. Menampilkan <i>form</i> untuk <i>register</i> .
5. Mengisi data yang diminta oleh sistem.	6. Menampilkan halaman <i>login</i> .
Skenario Alternatif	

Tidak mengisi data lengkap pada form.	Muncul peringatan untuk mengisi semua data.
---------------------------------------	---

Tabel 4. 23 Skenario Usecase Login

Nomor SRS	SRS_01_021
Nama Use case	Login
Aktor	Pelanggan
Tujuan	Untuk dapat menggunakan aplikasi diperlukan login.
Deskripsi	Use case ini menjelaskan bagaimana proses untuk login.
Pre-Condition (kondisi awal)	Pelanggan masuk aplikasi dan mengisi username dan password.
Post-Condition (kondisi akhir)	Menampilkan halaman beranda pelanggan.
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Membuka aplikasi.	2. Menampilkan halaman beranda login.
3. Mengisi form username dan password untuk login.	4. Cek status login.
	5. Menampilkan halaman beranda pelanggan.
Skenario Alternatif	
Data yang dimasukkan salah.	Muncul notifikasi data salah dan diminta untuk memasukkan data yang benar.

Tabel 4. 24 Skenario Use case Logout

Nomor SRS	SRS_01_022
Nama Use case	Logout
Aktor	Pelanggan
Tujuan	Untuk keluar dari sistem.
Deskripsi	Use case ini menjelaskan bagaimana proses untuk keluar dari sistem.
Pre-Condition (kondisi awal)	Menekan tombol keluar pada pojok kanan atas sistem.

Post-Condition (kondisi akhir)	Menampilkan halaman <i>login</i> .
Skenario Utama	
Aksi Dari Aktor	Reaksi Dari Sistem
1. Memilih menu <i>logout</i>	2. Melakukan <i>logout</i> .
	3. Menampilkan halaman <i>login</i> .

4.1.4 Analisis Kebutuhan Non-Fungsional

Kebutuhan *non-fungsional* adalah kebutuhan yang berisi property perilaku yang dimiliki oleh sistem. Berikut ini adalah kebutuhan *non-fungsional* dari sistem penentuan dosen pembimbing dan penguji skripsi ini yaitu dijelaskan pada tabel 4.27 berikut ini.

Tabel 4. 25 Kebutuhan Non-Fungsional

Nomer SRS	Kebutuhan Non-Fungsional	Deskripsi
SRS_02_001	<i>Rational Performance tester</i>	Sistem dapat memenuhi performa yang diinginkan

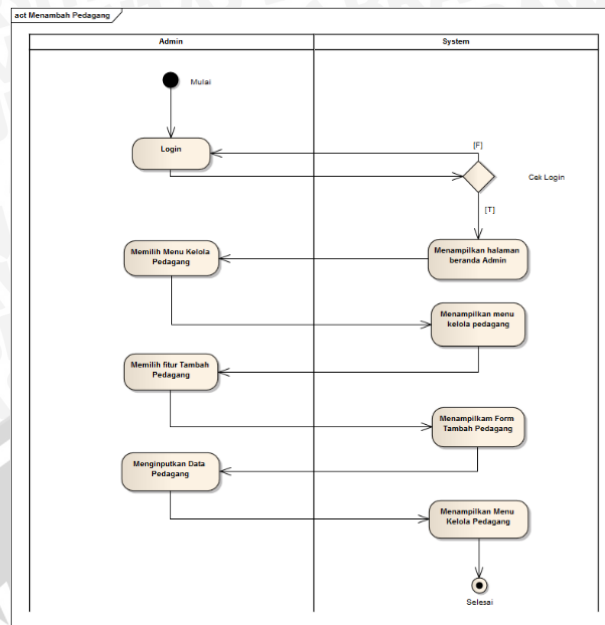
4.2 Perancangan Aplikasi

Perancangan aplikasi perangkat lunak sendiri terdiri dari beberapa tahapan, yaitu: perancangan *activity diagram*, perancangan *sequence diagram*, perancangan *class diagram*, perancangan basis data, perancangan arsitektur sistem, perancangan navigasi dan antarmuka dan perancangan skenario pengujian. Perancangan arsitektur sistem ada 2, yaitu: perancangan arsitektur *client* dan *sever*. Perancangan navigasi dan antarmuka terdiri dari 2 tahap, yaitu: perancangan *screen flow* dan perancangan antarmuka.

4.2.1 Activity Diagram

a. Menambah Pedagang

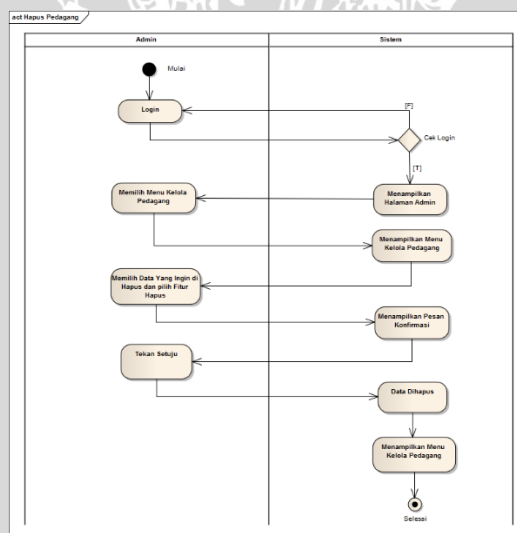
Activity diagram Tambah Pedagang merupakan proses di mana *admin* memasukkan data pedagang yang mendaftar ke dalam sistem. Untuk melakukan tambah pedagang ini, *admin* diharuskan untuk *login* terlebih dahulu, lalu membuka menu kelola pedagang dan memilih fitur tambah pedagang dan mengisi *form* tambah pedagang. *Diagram activity* tambah pedagang dapat dilihat pada gambar 4.4.



Gambar 4. 4 Activity Diagram Tambah Pedagang.

b. Activity Diagram Hapus Pedagang

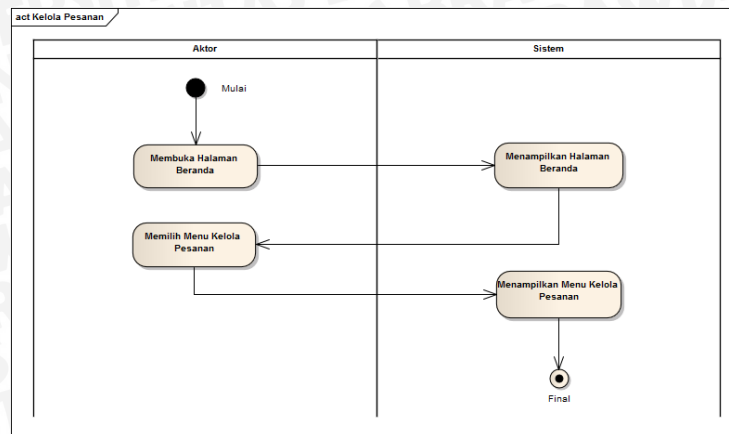
Activity diagram Hapus pedagang merupakan proses di mana seorang *admin* menghapus data pedagang dari sistem. Untuk menghapus data pedagang, *admin* diharuskan untuk *login*, lalu membuka menu kelola pedagang dan memilih fitur hapus pedagang. Activity diagram hapus pedagang dapat dilihat pada gambar 4.5.



Gambar 4. 5 Activity Diagram Hapus Pedagang

c. Activity Diagram Kelola Pesanan Pedagang

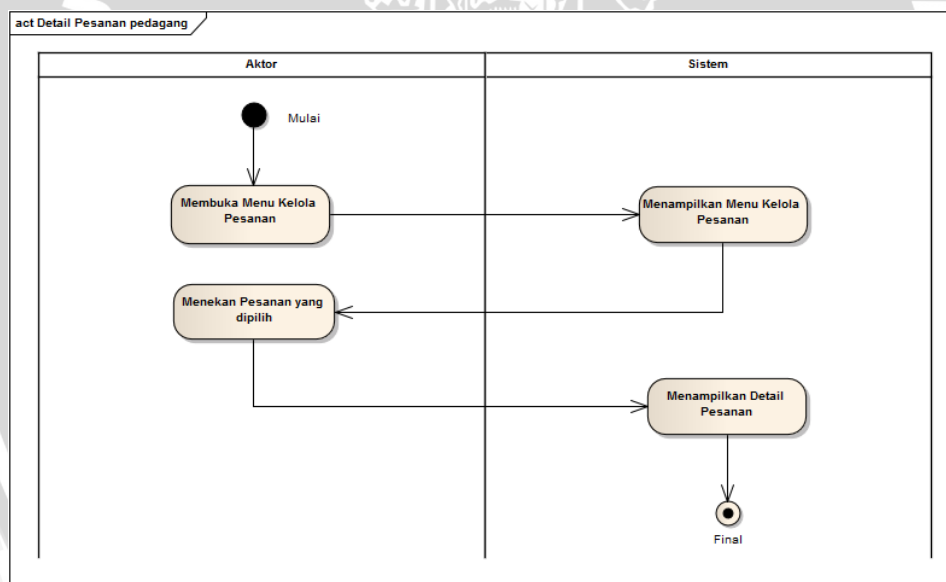
Activity diagram Kelola Pesanan Pedagang merupakan proses di mana pedagang melihat pesanan yang sudah dikirimkan oleh para pelanggan. Untuk melihat pesanan, pedagang harus melakukan *login* terlebih dahulu, lalu memilih lihat pesanan. Activity diagram lihat pesanan dapat dilihat pada gambar 4.6.



Gambar 4. 6 Activity diagram kelola pesanan pedagang

d. Activity Diagram Detail Pesanan Pedagang

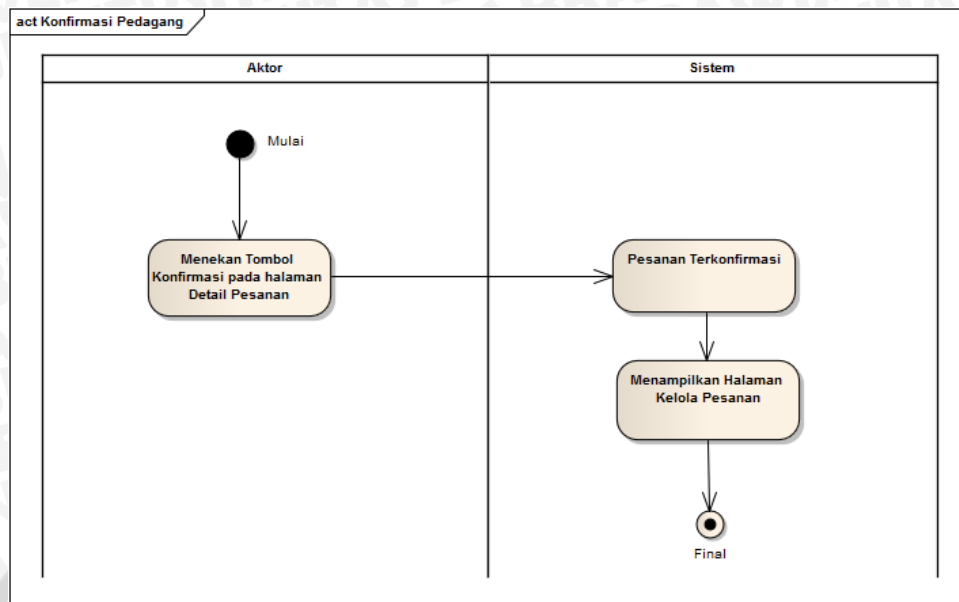
Activity diagram detail pesanan pedagang merupakan proses untuk melihat detail pesanan pelanggan. Untuk dapat melihat detail pesanan pelanggan, pedagang diharuskan untuk *login* terlebih dahulu. Activity diagram detail pesanan pedagang dapat dilihat pada gambar 4.7.



Gambar 4. 7 Activity Diagram Detail Pesanan Pedagang

e. Activity Diagram Konfirmasi Pesanan Pedagang

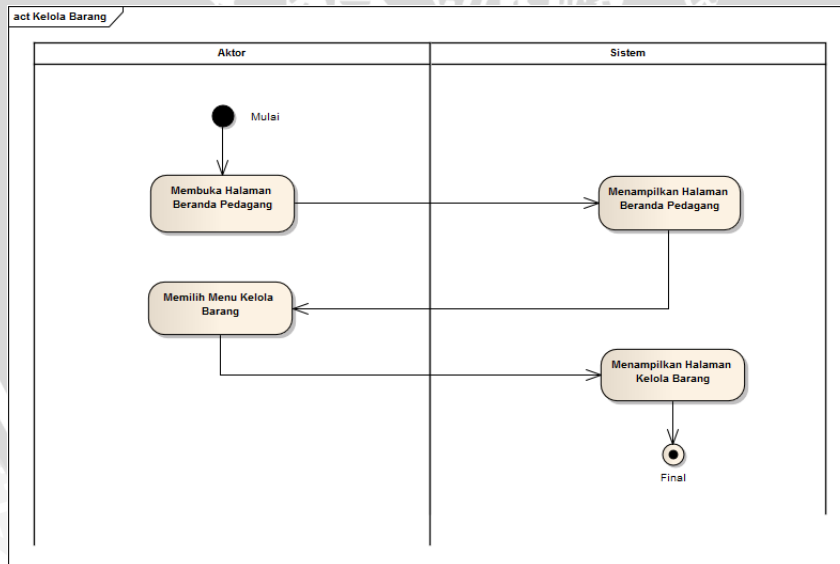
Activity diagram konfirmasi pesanan pedagang yaitu proses di mana pedagang menerima pesanan pelanggan. Untuk mengonfirmasi pesanan pelanggan maka pedagang diharuskan untuk *login* terlebih dahulu. Activity diagram konfirmasi pesanan pedagang dapat dilihat pada gambar 4.8.



Gambar 4. 8 Activity Diagram Konfirmasi Pesanan Pedagang

f. Activity Diagram Kelola Barang Dagangan

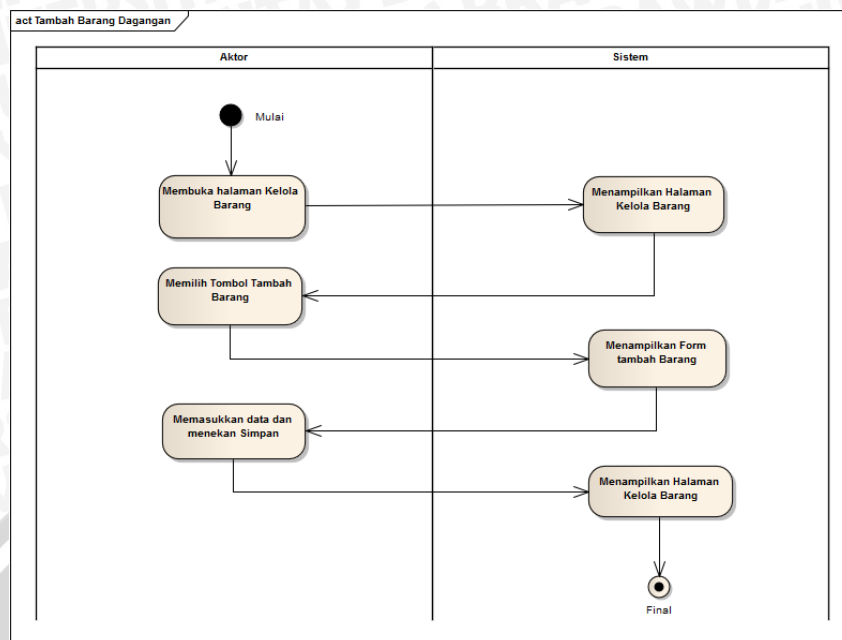
Activity diagram kelola barang dagangan merupakan proses membuka halaman kelola barang dan menampilkan *list* barang dagangan. Untuk melihat *list* barang dagangan ini pedagang diharuskan untuk *login* terlebih dahulu. Acitivty diagram kelola barang dagangan dapat dilihat pada gambar 4.9.



Gambar 4. 9 Activity Diagram Kelola Barang

g. Activity Diagram Tambah Barang Dagangan

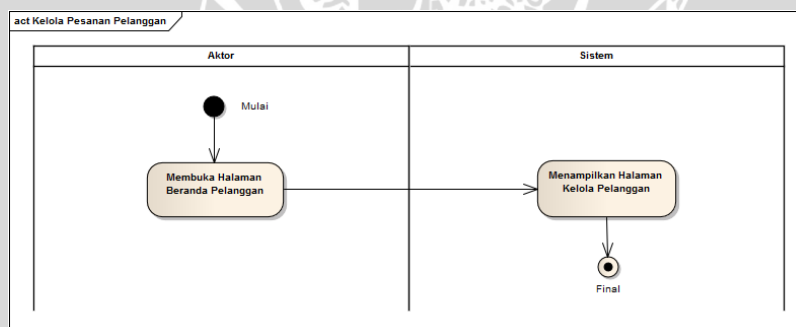
Activity diagram tambah barang dagangan merupakan proses di mana pedagang menambah barang dagangan. Untuk melakukan aksi ini pedagang diharuskan untuk *login* terlebih dahulu. Activity diagram tambah barang dagangan dapat dilihat pada gambar 4.10



Gambar 4. 10 Activity Diagram Tambah Barang

h. Activity Diagram Kelola Pesanan Pelanggan

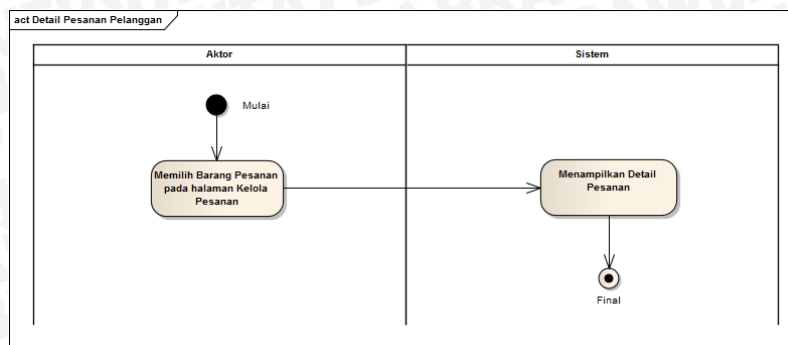
Activity diagram kelola pesanan pelanggan merupakan proses untuk memesan barang pada pedagang. Untuk memesan barang, pelanggan diharuskan untuk *login* terlebih dahulu, lalu memilih menu kelola pesanan dan memilih fitur tambah pesanan. *Activity diagram* kelola pesanan dapat dilihat pada gambar 4.11.



Gambar 4. 11 Activity Diagram Kelola Pesanan Pelanggan

i. Activity Diagram Detail Pesanan Pelanggan

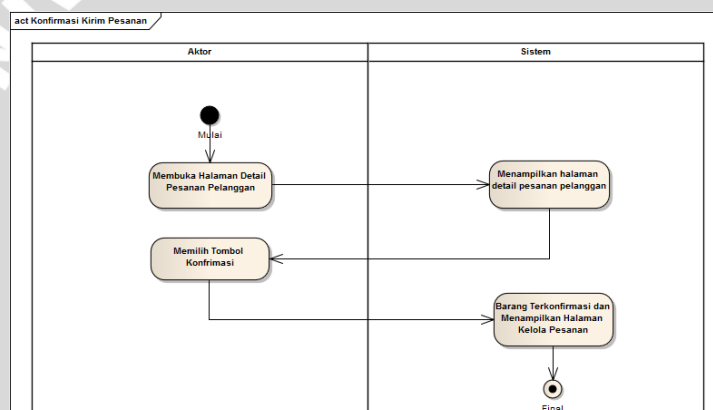
Activity diagram detail pesanan pelanggan merupakan proses untuk melihat detail pesanan pelanggan. Untuk dapat melihat detail pesanan pelanggan, pelanggan diharuskan untuk *login* terlebih dahulu. *Activity diagram* detail pesanan pelanggan dapat dilihat pada gambar 4.12



Gambar 4. 12 Activity diagram detail pesanan

j. Activity Diagram Konfirmasi Pesanan Pelanggan

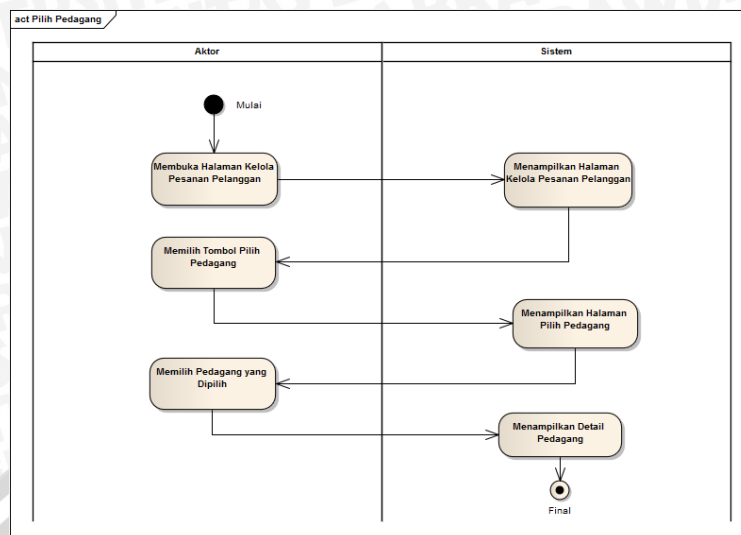
Activity diagram konfirmasi pesanan pelanggan yaitu di mana pelanggan menerima kiriman pesanan dari pedagang. Untuk mengonfirmasi pesanan pelanggan maka pelanggan diharuskan untuk *login* terlebih dahulu. *Activity diagram* konfirmasi pesanan pelanggan dapat dilihat pada gambar 4.13.



Gambar 4. 13 Activity Diagram Konfirmasi Pesanan Pelanggan

k. Activity Diagram Pilih Pedagang

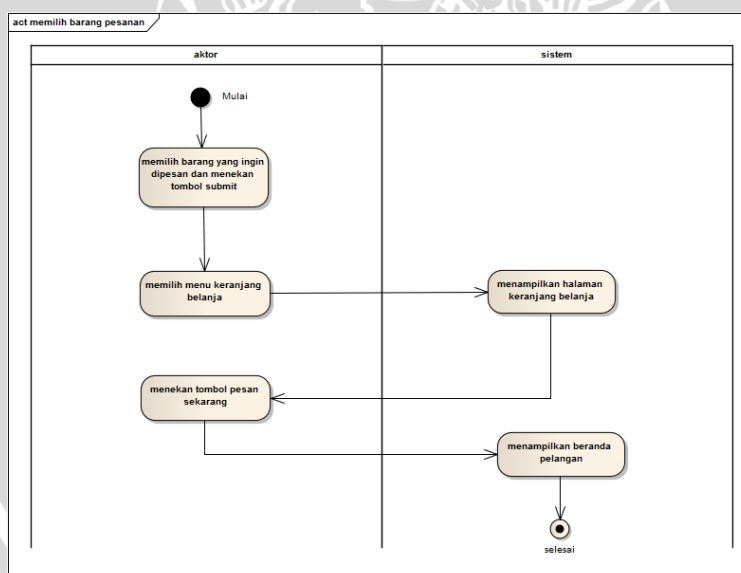
Activity diagram pilih pedagang adalah proses di mana pelanggan memilih pedagang untuk memesan barang. Untuk melakukan aksi ini pelanggan diharuskan untuk *login* terlebih dahulu. *Activity diagram* pilih pedagang dapat dilihat pada gambar 4.14.



Gambar 4. 14 Activity Diagram Pilih Pedagang

I. Activity Diagram Pilih Barang

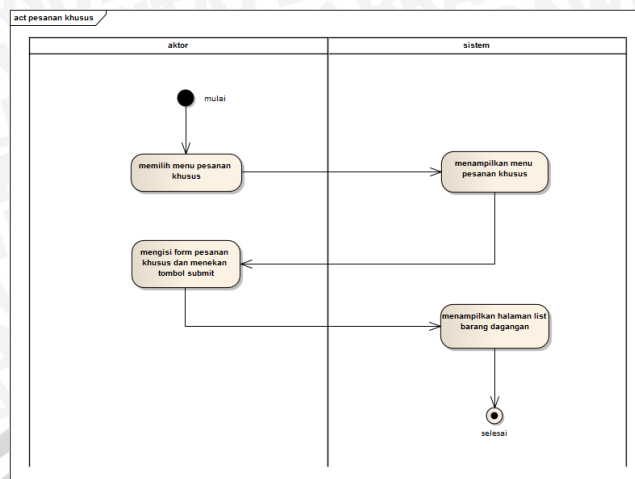
Activity diagram memesan barang adalah proses di mana pelanggan memilih barang yang akan dipesan dengan memilih barang dari list barang dagangan pedagang yang dipilih. Untuk melakukan aksi ini pelanggan diharuskan login terlebih dahulu. Activity diagram memesan barang dapat dilihat pada gambar 4.15.



Gambar 4. 15 Activity Diagram Pilih Barang

m. Activity Diagram Pesanan Khusus

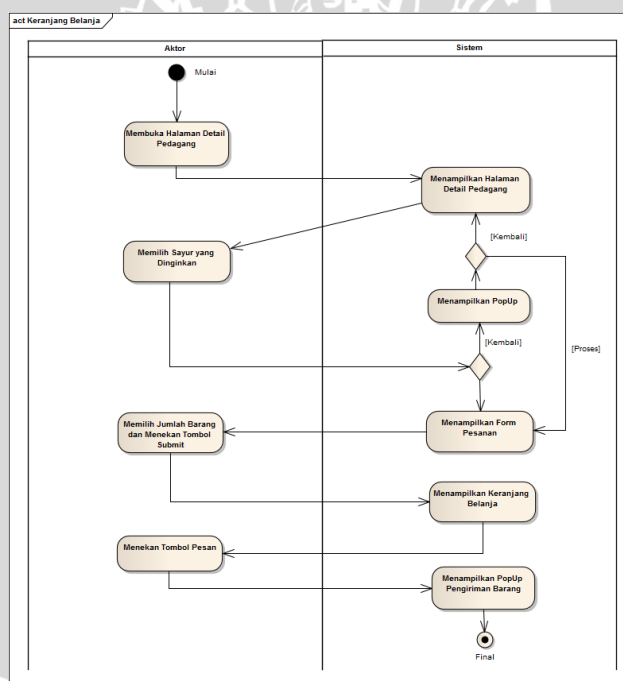
Activity diagram pesanan khusus merupakan proses di mana pelanggan memesan barang yang tidak ada pada list barang dagangan dengan mengisi form yang telah disediakan. Activity diagram pesanan khusus dapat dilihat pada gambar 4.16.



Gambar 4. 16 Activity Diagram Pesanan Khusus

n. Activity Diagram Keranjang Belanja

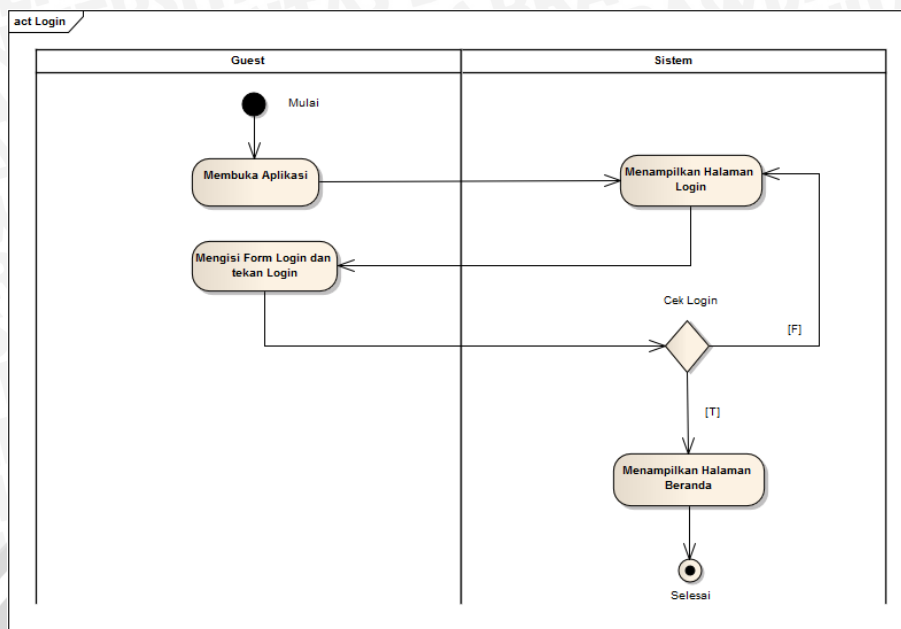
Activity diagram keranjang belanja yaitu proses di mana pelanggan memilih dan memasukkan barang pesanan ke dalam daftar keranjang belanja. Untuk melakukan aksi ini diharuskan untuk *login* terlebih dahulu. Activity diagram keranjang belanja dapat dilihat pada gambar 4.17



Gambar 4. 17 Activity Diagram Keranjang Belanja

o. Login

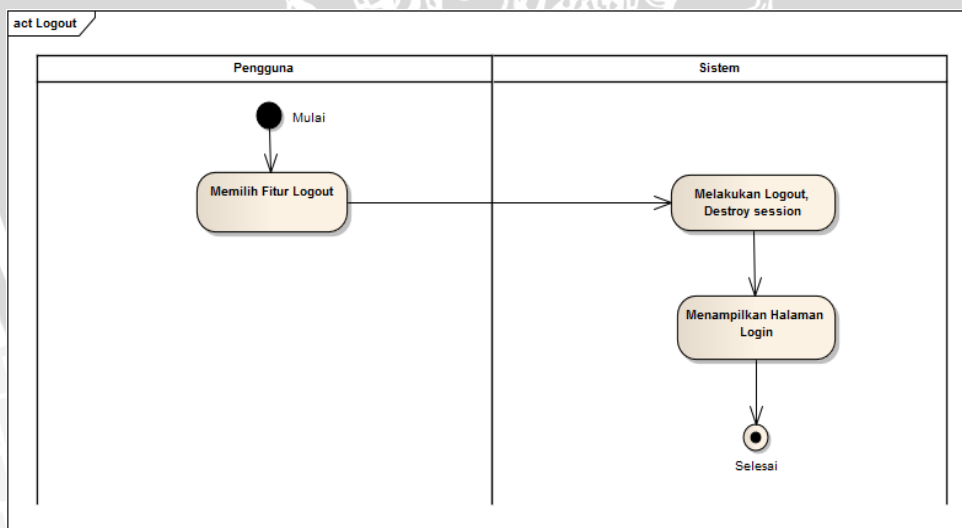
Activity diagram Login merupakan proses di mana guest memasukkan data berupa *password* dan *username*, yang mana nantinya setelah proses *login* selesai, guest ini akan diketahui sebagai *admin*, pedagang atau pelanggan. Activity diagram Login dapat dilihat pada gambar 4.18.



Gambar 4. 18 Activity Diagram Login.

p. Logout

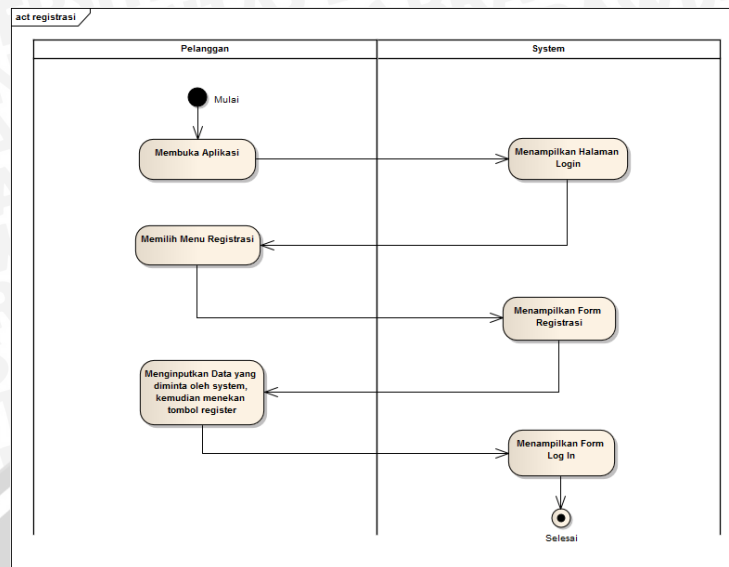
Activity diagram logout merupakan proses di mana pengguna memilih tombol keluar yang ada pada pojok kanan atas sistem. Sistem akan menghapus *session* dari pengguna lalu akan keluar dari dalam menu dan ditampilkan halaman *login*. *Activity diagram Logout* dapat dilihat pada gambar 4.19.



Gambar 4. 19 Activity Diagram Logout.

q. Activity diagram Register

Activity diagram register adalah proses di mana pelanggan mendaftar ke dalam sistem, setelah memilih menu *register* diharapkan pelanggan mengisi data dengan benar. *Activity diagram register* dapat dilihat pada gambar 4.20.



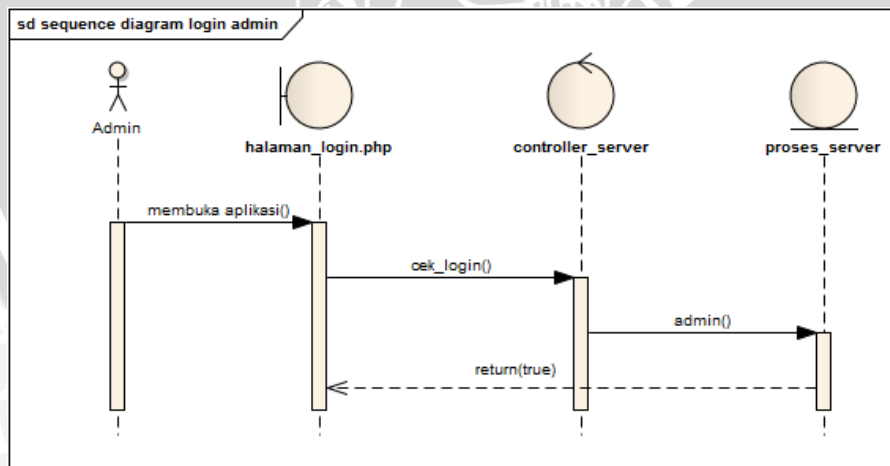
Gambar 4. 20 Activity diagram resgister

4.2.2 Sequence Diagram

Diagram sequence merepresentasikan Model alir jalannya proses interaksi antar objek. Dengan kata lain, diagram ini menggambarkan urutan langkah-langkah interaksi antar komponen dalam sistem.

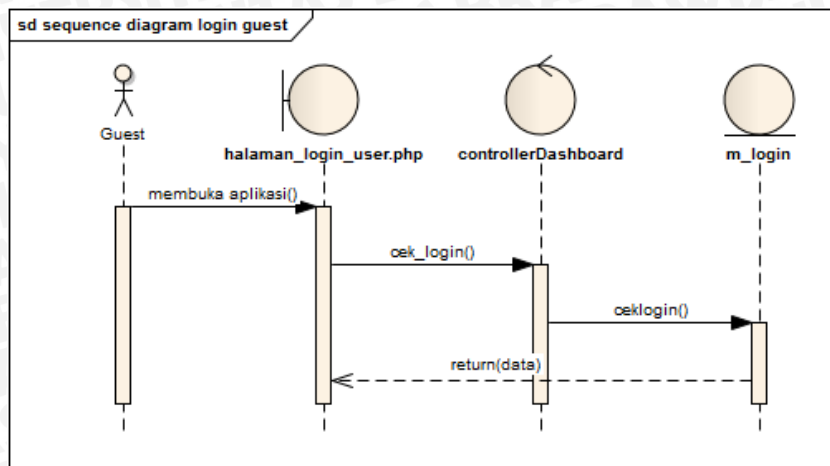
1. Sequence Diagram Login

Sequence diagram login menjelaskan urutan kegiatan login ke dalam sistem. Pada gambar 4.21 ditunjukkan sequence diagram login admin dan pada gambar 4.22 ditunjukkan sequence diagram login guest.



Gambar 4. 21 sequence diagram login admin

Gambar 4.21 menjelaskan urutan kegiatan admin untuk melakukan login yaitu admin membuka halaman login lalu ketika menekan tombol masuk akan menjalankan fungsi cek_login pada Controller yang memanggil fungsi admin pada Model.

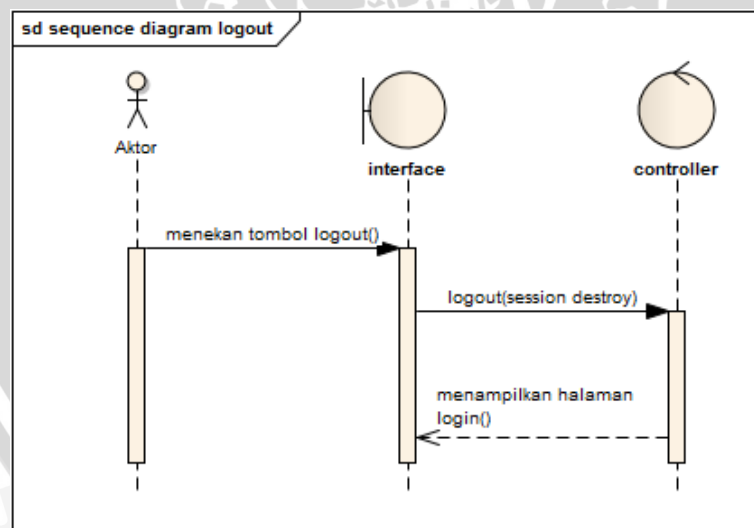


Gambar 4. 22 sequence diagram login guest

Gambar 4.22 menjelaskan urutan kegiatan guest untuk melakukan *login* yaitu guest membuka halaman *login* lalu ketika menekan tombol masuk akan menjalankan fungsi *cek_login* pada *Controller* yang memanggil fungsi *cek_login* pada *Model*.

2. Sequence Diagram Logout

Pada gambar 4.23 menjelaskan urutan kegiatan untuk keluar dari sistem atau disebut *logout*. Untuk melakukan *logout* aktor diharuskan untuk menekan tombol *logout* yang ada pada setiap halaman. Setelah menekan tombol *logout* sistem akan menjalankan fungsi *logout* yang akan menghapus *session* yang disimpan sistem.

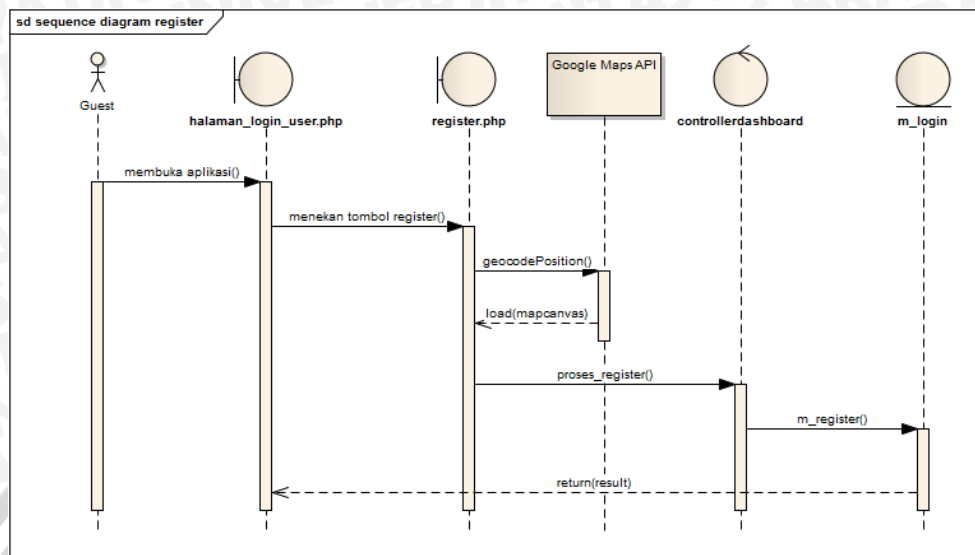


Gambar 4. 23 sequence diagram logout

3. Sequence Diagram Register

Pada gambar 4.24 menunjukkan urutan kegiatan mendaftar ke dalam sistem atau biasa disebut *register*. Untuk mendaftar ke dalam sistem aktor harus memilih tombol *register* pada halaman *login*. Setelah mengisi data dan menekan tombol

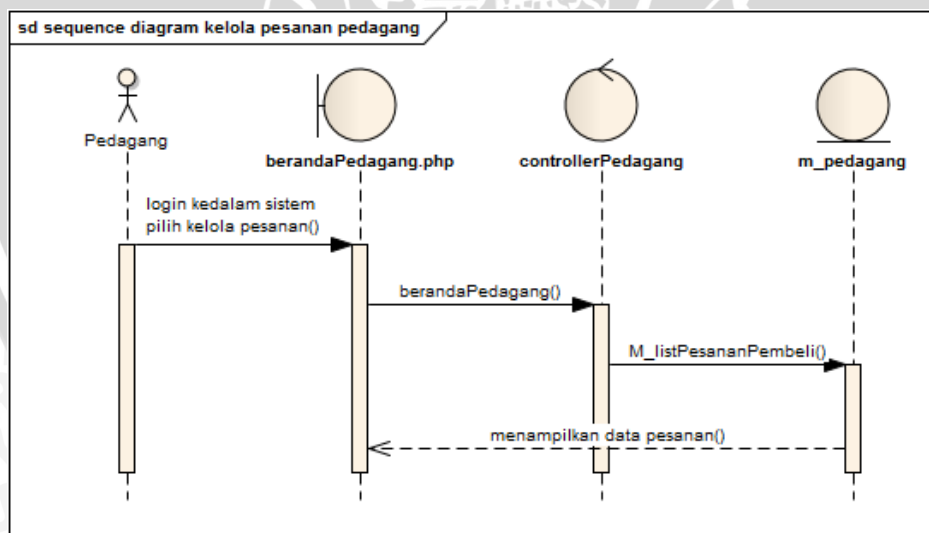
daftar sistem akan menjalankan fungsi *proses_register* pada *Controller* dan *m_register* pada *Model* untuk memasukkan ke dalam *database*.



Gambar 4. 24 *sequence diagram register*.

4. Sequence Diagram Kelola Pesanan Pedagang

Pada gambar 4.25 menunjukkan urutan kegiatan *aktor* setelah *login* sebagai pedagang yang akan menampilkan halaman beranda atau kelola pesanan. Untuk membuka menu kelola pedagang ini diperlukan pemanggilan fungsi *berandaPedagang* pada *Controller* dan *M_listPesananPembeli* pada *Model*.

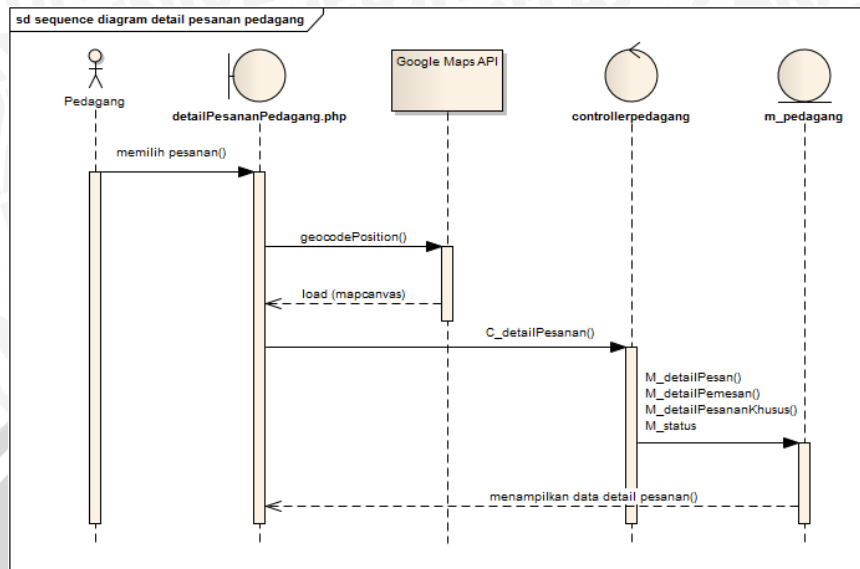


Gambar 4. 25 *sequence diagram kelola pesanan pedagang*

5. Sequence Diagram Detail Pesanan Pedagang

Pada gambar 4.26 menunjukkan urutan kegiatan melihat detail pesanan pelanggan pada sisi pedagang. Untuk membuka menu detail pesanan sistem perlu menjalankan fungsi *c_detailPesanan* pada *Controller* dan pada *Model* menjalankan

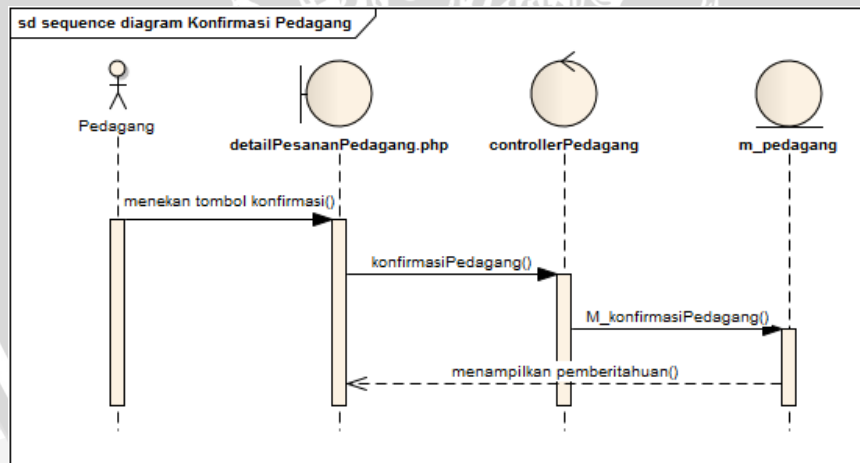
4 fungsi yaitu M_detailPesanan, M_detailPemesan, M_detailPesananKhusus dan M_status.



Gambar 4. 26 sequence diagram detail pesanan pedagang

6. Sequence Diagram Konfirmasi Pesanan Pedagang

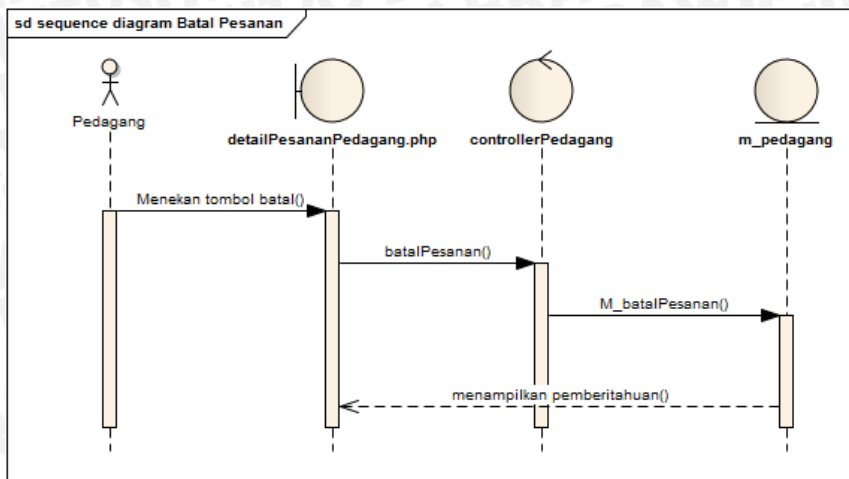
Pada gambar 4.27 ditunjukkan urutan kegiatan mengonfirmasi pesanan pelanggan. Untuk mengonfirmasi pelanggan sistem perlu memanggil fungsi konfirmasi pada *Controller* dan fungsi M_konfirmasiPedagang pada *Model*.



Gambar 4. 27 sequence diagram konfirmasi pedagang

7. Sequence Diagram Batal Pesanan Pedagang

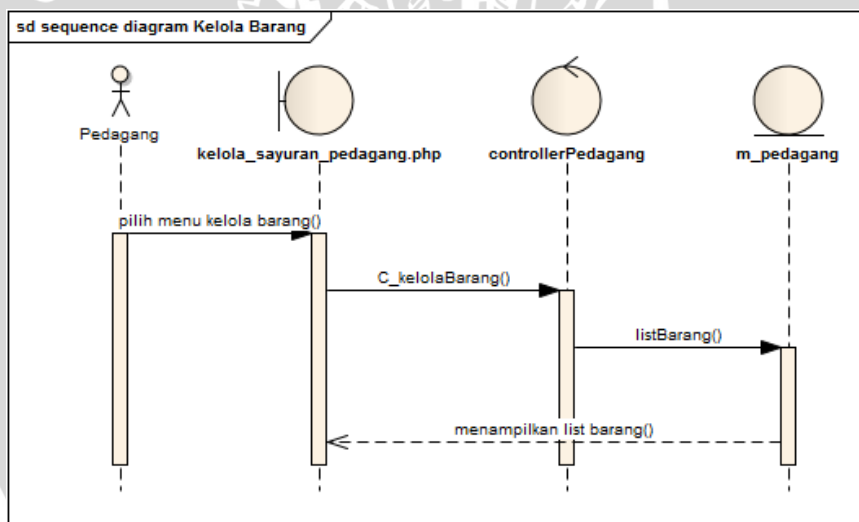
Gambar 4.28 menunjukkan urutan kegiatan pedagang membatalkan pesanan dikarenakan beberapa alasan. Untuk membatalkan pesanan pelanggan sistem menjalankan fungsi batalPesanan pada *Controller* dan M_batalPesanan pada *Model*.



Gambar 4. 28 *sequence diagram* batal pesanan pedagang

8. Sequence Diagram Kelola Barang

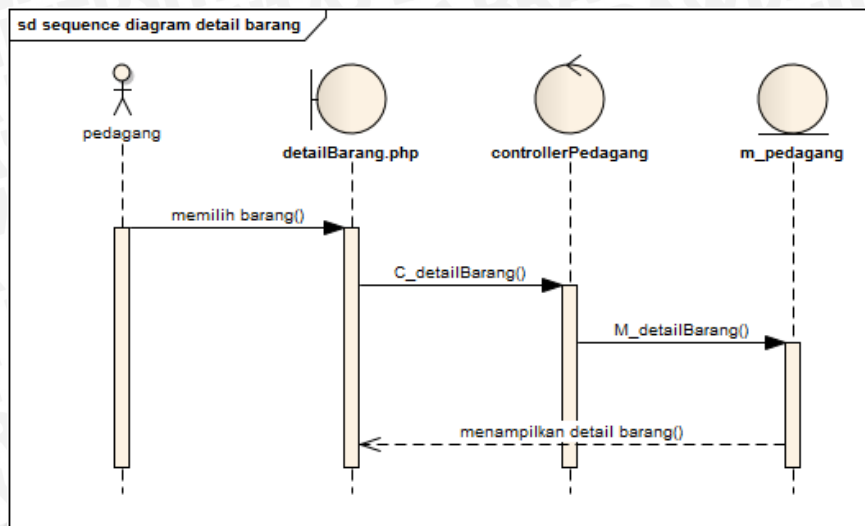
Pada gambar 4.29 *sequence diagram* kelola barang menunjukkan urutan pedagang untuk membuka halaman kelola barang. Untuk masuk menu kelola barang sistem menjalankan fungsi C_kelolaBarang pada *Controller* dan *listBarang* serta *listKategori* pada *Model*.



Gambar 4. 29 *sequence diagram* kelola barang

9. Sequence Diagram Detail Barang

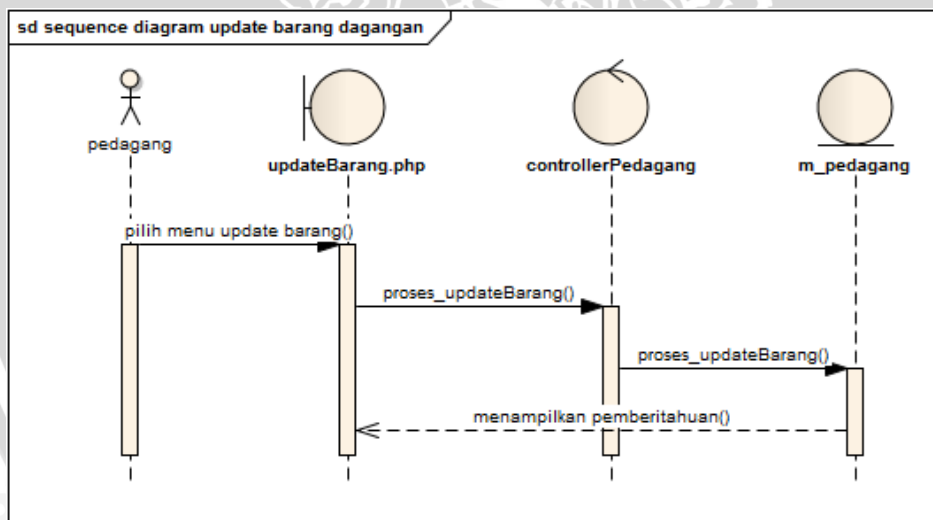
Pada gambar 4.30 *sequence diagram* detail barang menunjukkan urutan kegiatan pedagang untuk masuk ke halaman detail barang dengan menekan barang yang diinginkan. Untuk menampilkan detail barang yang sudah dipilih sistem menjalankan fungsi C_detailBarang dan M_detailBarang.



Gambar 4. 30 Sequence Diagram Detail Barang

10. Sequence Diagram Edit Barang Dagangan

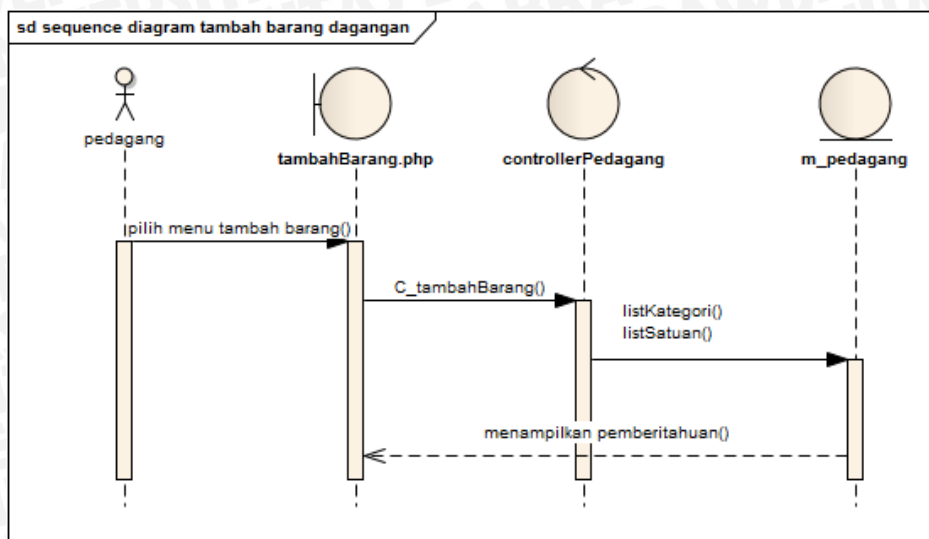
Gambar 4.31 Sequence Diagram Update Barang Dagangan menunjukkan urutan kegiatan untuk melakukan update data barang. Untuk melakukan update sistem perlu menjalankan fungsi proses_updateBarang pada Controller dan proses_updateBarang pada Model.



Gambar 4. 31 Sequence Diagram Edit Barang Dagangan

11. Sequence Diagram Tambah Barang Dagangan

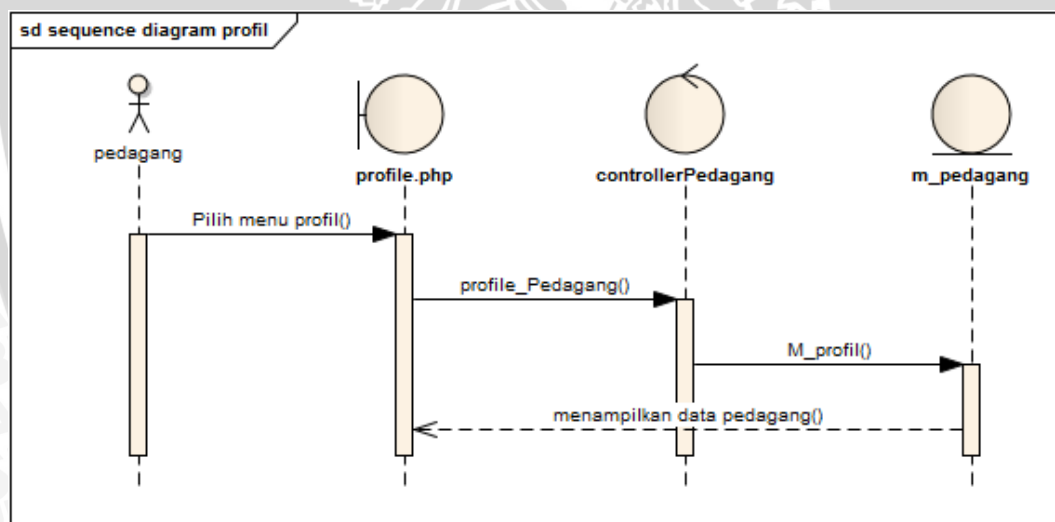
Gambar 4.32 menunjukkan urutan kegiatan menambah barang dagangan pedagang. Untuk melakukan aksi ini diperlukan menjalankan fungsi C_tambahBaranf pada Controller dan listKategori serta listSatuan pada Model.



Gambar 4. 32 *sequence diagram* tambah barang dagangan

12. Sequence Diagram Profil

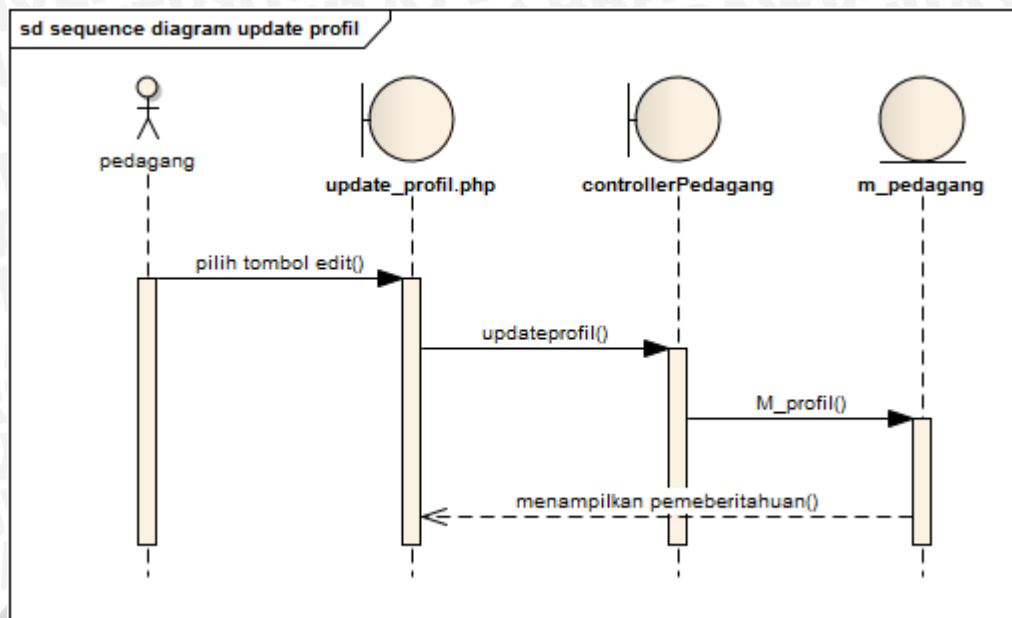
Gambar 4.33 *sequence diagram* profil menunjukkan urutan kegiatan untuk membuka halaman pprofil yang ada pada pojok atas kanan halaman. Untuk membuka halaman ini sistem perlu memanggil fungsi profile_Pedagang pada *Controller* dan fungsi M_profil pada *Model*.



Gambar 4. 33 *sequence diagram* profil

13. Sequence Diagram Edit Profil

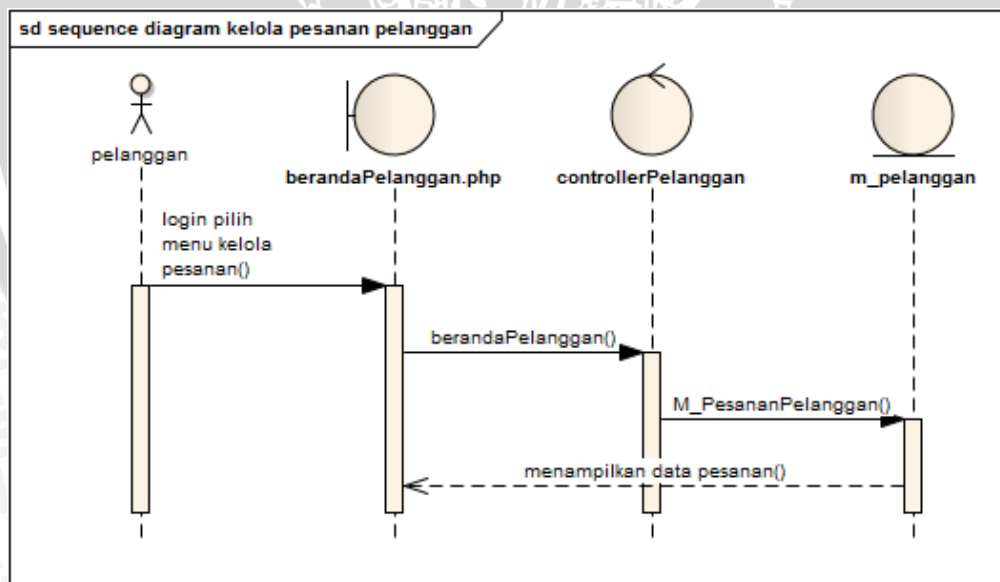
Gambar 4.34 *sequence diagram* update profil menunjukkan urutan kegiatan untuk pedagang mengubah data diri pedagang. Untuk menjalankan update profil diperlukan menjalankan fungsi updateProfil pada *Controller* dan M_profil pada *Model*.



Gambar 4. 34 sequence diagram Edit profil

14. Sequence Diagram Kelola Pesanan Pelanggan

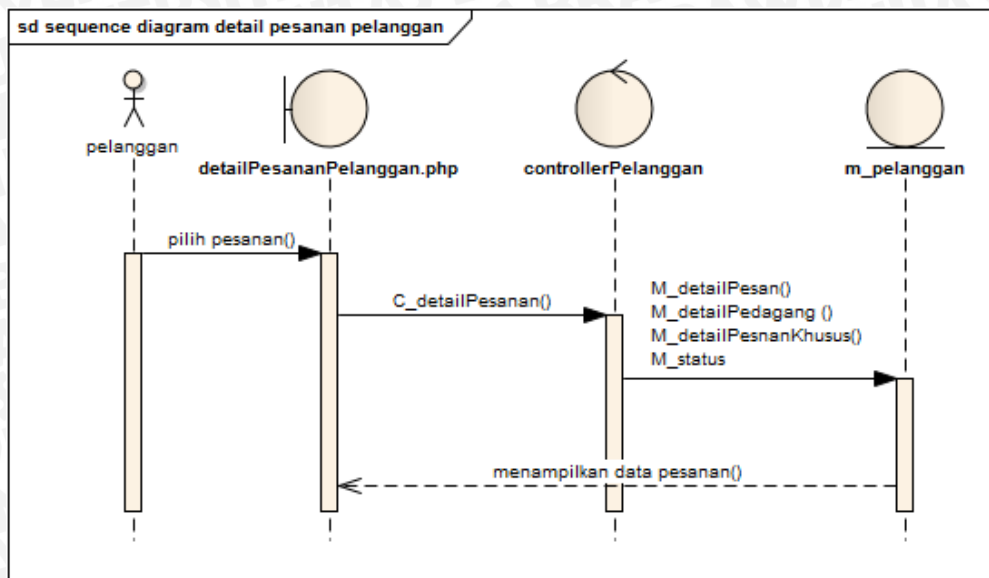
Pada gambar 4.35 ditunjukkan urutan kegiatan untuk melihat pesanan yang pernah dipesan oleh pengguna tersebut. Untuk menjalankan aksi ini sistem perlu menjalankan `berandaPelanggan` pada *Controller* dan `M_PesananPelanggan` pada *Model*.



Gambar 4. 35 sequence diagram kelola pesanan pelanggan

15. Sequence Diagram Detail Pesanan Pelanggan

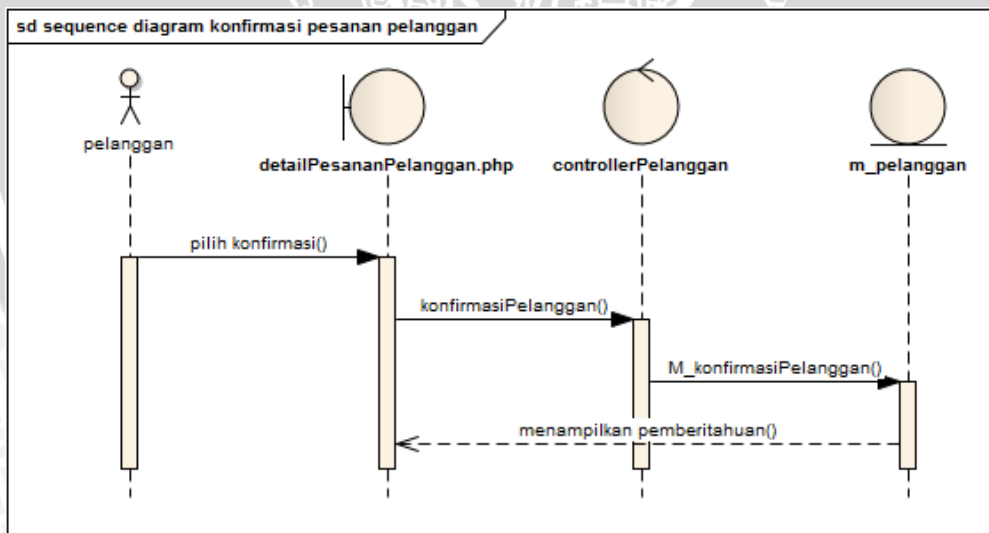
Gambar 4.36 menunjukkan urutan kegiatan pelanggan untuk melihat pesanan yang sebelumnya sudah dipesan. Untuk menjalankan aksi ini sistem perlu menjalankan fungsi `C_detailPesanan` pada *Controller* dan `M_detailPesan`, `M_detailPedagang`, `M_detailPesanKhusus` serta `M_status` pada *Model*.



Gambar 4. 36 *sequence diagram* detail pesanan pelanggan

16. Sequence Diagram Konfirmasi Pesanan Pelanggan

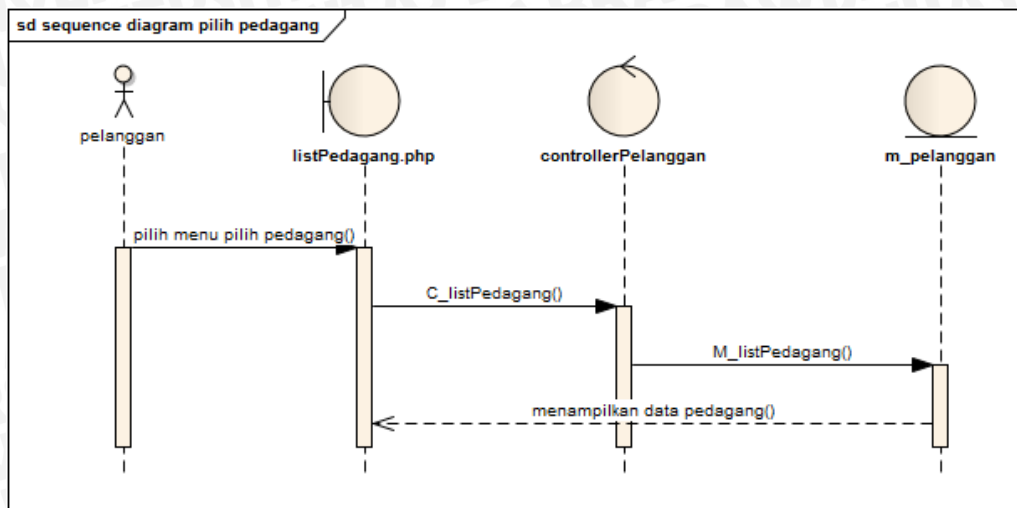
Gambar 4.37 menunjukkan urutan pelanggan untuk mengonfirmasi bahwa barang telah dikirim atau untuk membatalkan pesanan. Pada aksi konfirmasi ini ada 2 aksi yaitu batal dan konfirmasi dalam 1 fungsi, konfirmasiPelanggan. Untuk menjalankan konfirmasi ini sistem memanggil fungsi konfirmasiPelanggan pada *Controller* dan M_konfirmasiPelanggan pada *Model*.



Gambar 4. 37 *sequence diagram* konfirmasi pesanan pelanggan

17. Sequence Diagram Pilih Pedagang

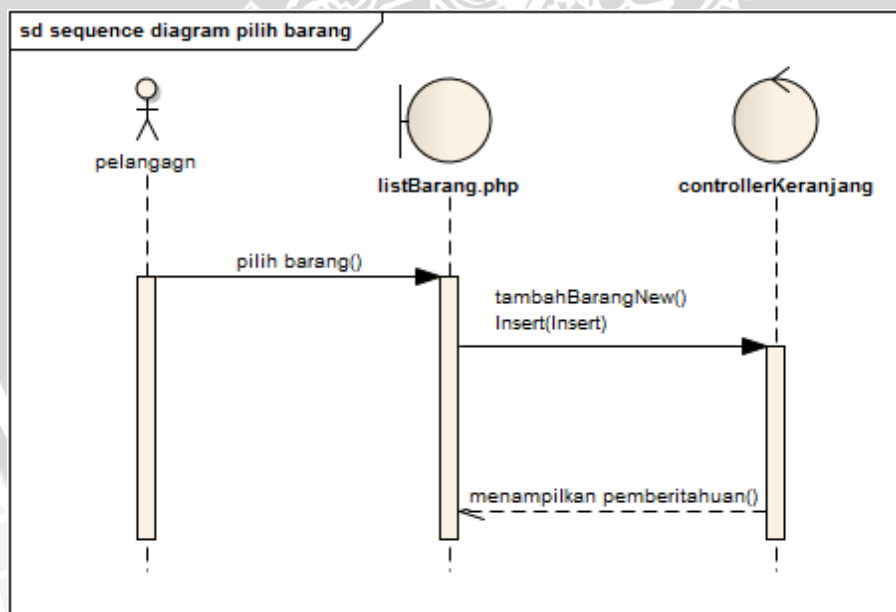
Gambar 4.38 menunjukkan urutan kegiatan pelanggan untuk memilih pedagang sebelum memesan barang ke pada pedagang. Untuk membuka halaman ini sistem perlu memanggil fungsi C_listPedagadang pada *Controller* dan M_listPedagang pada *Model*.



Gambar 4. 38 sequence diagram pilih pedagang

18. Sequence Diagram Pilih Barang

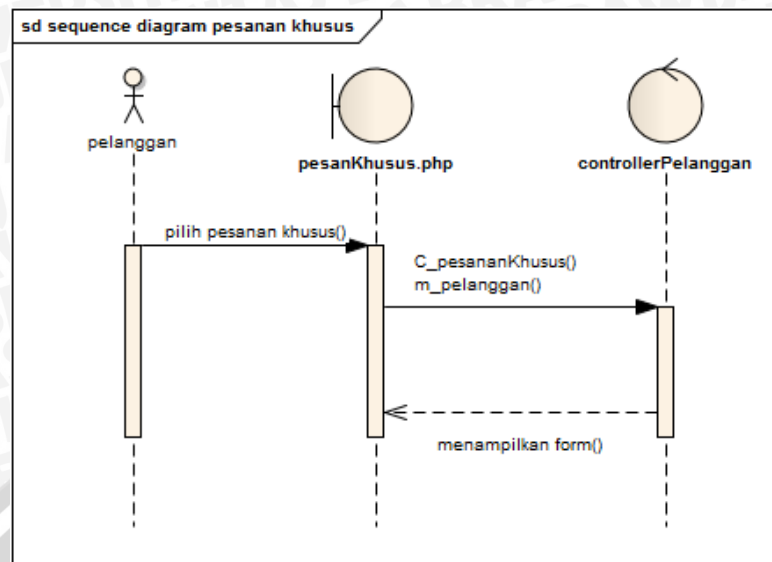
Gambar 4.39 menunjukkan urutan kegiatan pelanggan untuk memilih barang yang akan dipesan pada menu *list* barang dagangan. Ketika memilih barang, barang akan dimasukkan ke dalam keranjang, untuk memasukkan keranjang sistem memanggil fungsi *tambahBarangNew* pada *Controller* dan *insert* ke dalam *library cart* yang sudah disediakan oleh *Codeigniter*.



Gambar 4. 39 sequence diagram pilih barang.

19. Sequence Diagram Pesanan Khusus

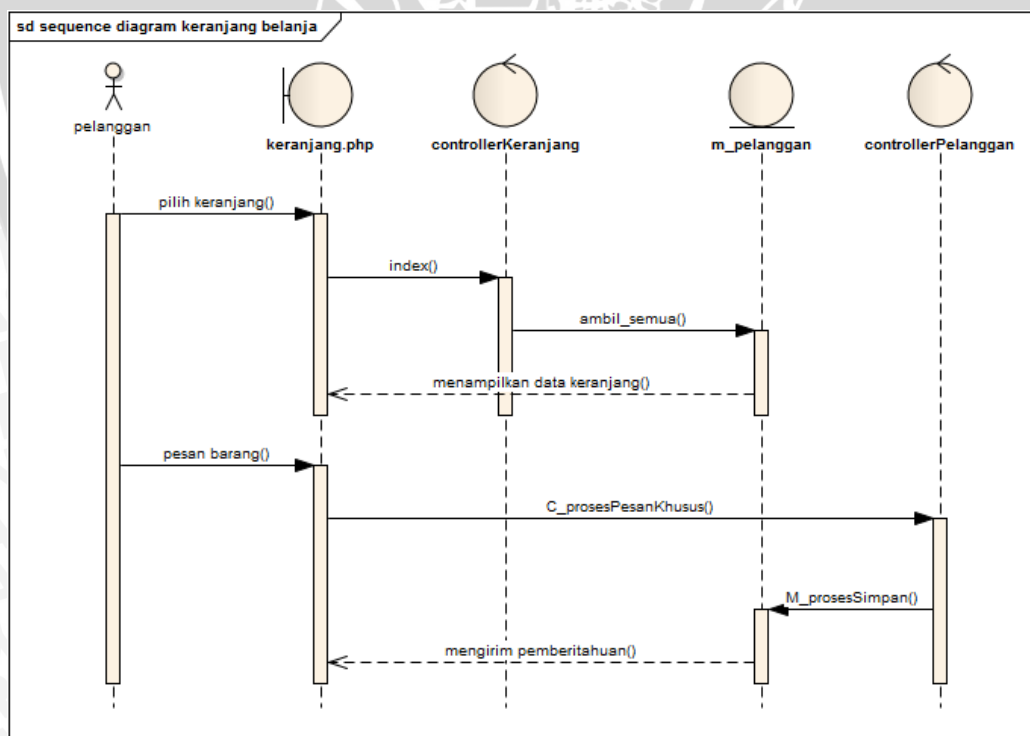
Gambar 4.40 menunjukkan urutan kegiatan pelanggan untuk memesan pesanan khusus yang bertujuan untuk memesan barang yang tidak ada dalam *list* barang jualan pedagang. Sama dengan aksi memilih barang pesan khusus juga dimasukkan ke dalam keranjang dengan memanggil fungsi *C_prosesPesananKhusus* pada *Controller* dan *insert* untuk masuk ke dalam *cart*.



Gambar 4. 40 sequence diagram pesanan khusus

20. Sequence Diagram Keranjang Belanja

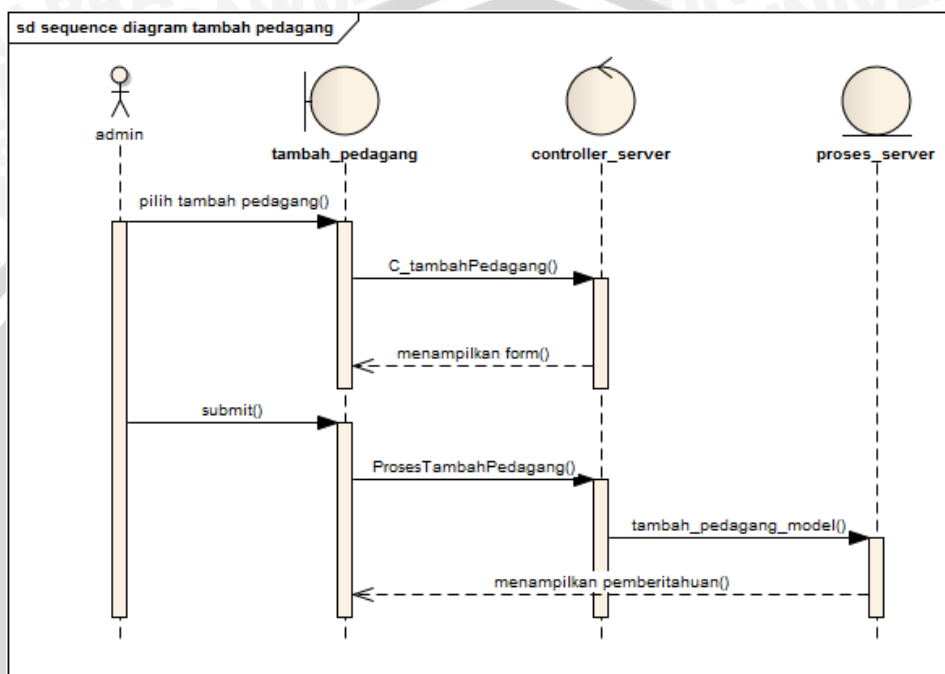
Gambar 4.41 menunjukkan urutan kegiatan pelanggan untuk masuk ke halaman keranjang belanja dan memesan barang. Untuk masuk ke halaman keranjang sistem perlu memanggil fungsi index pada *Controller* keranjang dan fungsi ambil_semua pada *Model* m_pelanggan. Ketika memesan barang sistem menjalankan fungsi prosesSimpan pada *Controller* dan M_prosesSimpan pada *Model*.



Gambar 4. 41 sequence diagram keranjang belanja

21. Sequence Diagram Tambah Pedagang

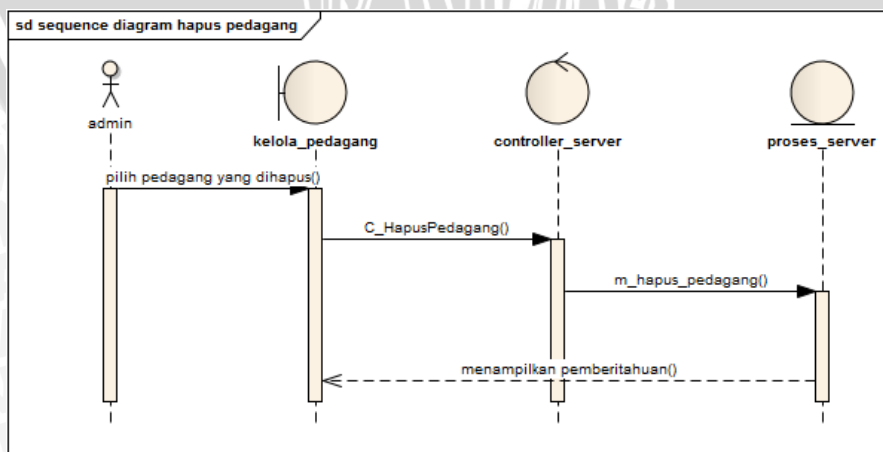
Gambar 4.42 menunjukkan urutan kegiatan *admin* untuk menambah pedagang. Untuk menambah pedagang *admin* harus membuka halaman tambah pedagang dengan memanggil fungsi *C_tambahPedagang* dan sistem akan menampilkan *form* untuk mengisi data. Sedangkan untuk menyimpan data sistem menjalankan fungsi *prosesTambahPedagang* pada *Controller* dan *tambah_pedagang_Model*.



Gambar 4. 42 sequence diagram tambah pedagang.

22. Sequence Diagram Hapus Pedagang

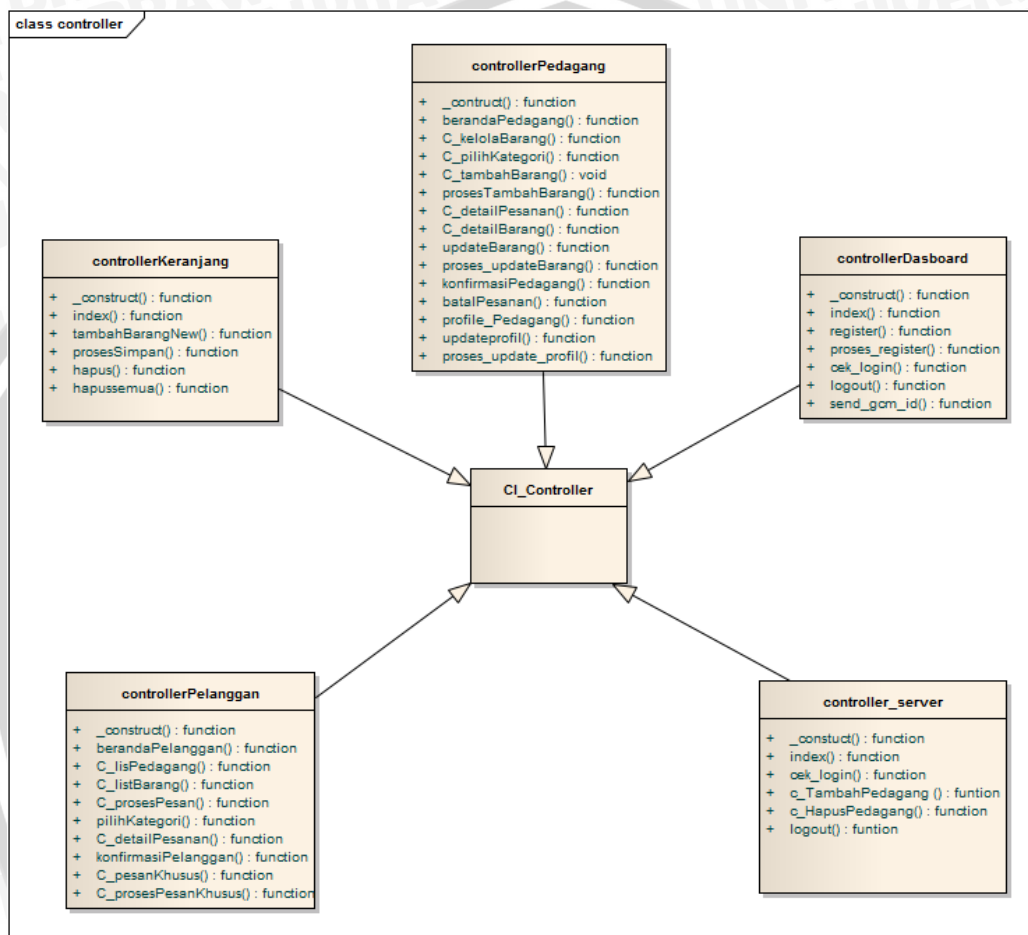
Gambar 4.43 menunjukkan urutan kegiatan *admin* untuk menghapus pedagang. Ketika menghapus barang sistem menjalankan fungsi *C_hapusPedagang* dan *m_hapus_pedagang*.



Gambar 4. 43 sequence diagram hapus pedagang

4.2.3 Class Diagram

Diagram kelas pada penelitian ini dibagi menjadi tiga *package* yaitu kelas *Controller*, *mode*, *view*. Hal tersebut dilakukan karena sistem ini menggunakan pola arsitektur MVC (*Model*, *View* *Controller*). Kelas pada *Controller* merupakan kelas proses yang didapatkan dari *diagram use case*.



Gambar 4. 44 class diagram class Controller

Sedangkan kelas pada *Model* merupakan kelas data yang berinteraksi dengan basis data sistem. Gambar 4.44 merepresentasikan *class Controller*, Gambar 4.45 merepresentasikan *class Model*.



4.3 Perancangan Basis Data

Perancangan basis data digunakan untuk merepresentasikan penyimpanan data-data dari *user* serta barang dagangan yang diperoleh dari tahap pengumpulan data pada bab II. Perancangan basis data digambarkan dalam bentuk *diagram* hubungan entitas yang menyediakan tampilan grafis dari struktur sistem melalui entitas dan relasi antar entitas di dalamnya. Gambar dari ERD pada sistem ini ditunjukkan pada gambar 4.46.

4.4 Perancangan Antarmuka

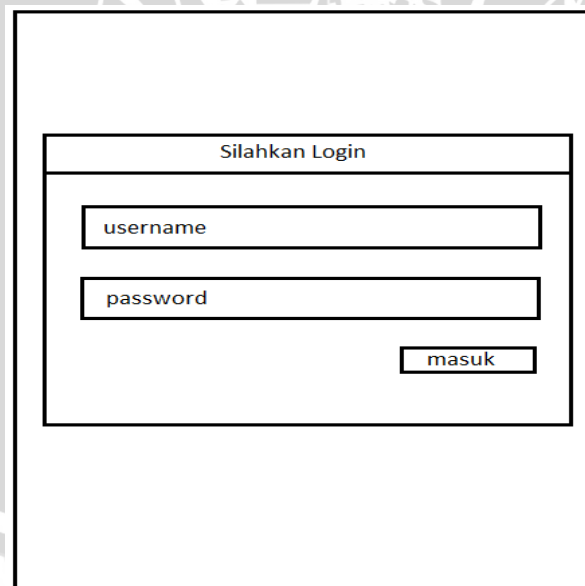
Berikut ini adalah perancangan antarmuka dari sistem pemesanan sayur. Antarmuka sistem ini dibagi menjadi 3, yaitu antarmuka *admin*, antarmuka pedagang dan antarmuka pelanggan atau pembeli.

1. Perancangan Antarmuka *Admin*

Antarmuka pengguna untuk sistem *user* berupa halaman *web* yang menggunakan *framework* PHP Codeigniter.

a. Halaman *Login Admin*

Halaman *login* merupakan halaman awal *admin* untuk masuk ke dalam sistem *admin*. Pada halaman ini terdapat *form* untuk mengisi *username* dan *password*. Untuk melanjutkan masuk ke dalam sistem *admin* harus memasukkan *username* dan *password* lalu menekan tombol masuk. Antarmuka halaman *login admin* dapat dilihat pada gambar 4.47.



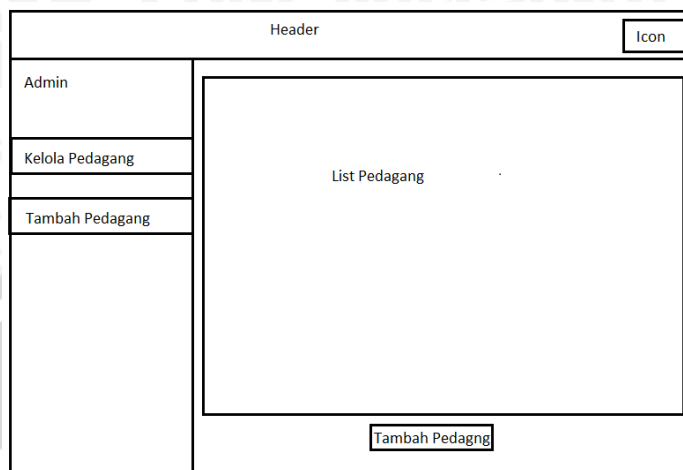
Silahkan Login	
username	
password	
masuk	

Gambar 4. 47 Antarmuka *Login Admin*

b. Halaman Beranda (Kelola Pedagang)

Halaman beranda atau kelola pedagang merupakan antarmuka *admin* setelah melakukan *login*. Pada halaman ini terdapat tabel yang berisi daftar pedagang yang sudah terdaftar ke dalam sistem. Pada bagian bawah tabel terdapat tombol

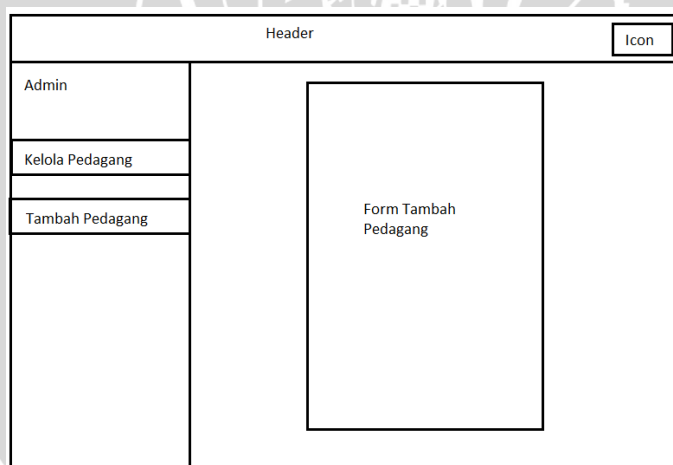
untuk menambah pedagang, selain tombol tambah pedagang pada bagian bawah tabel juga terdapat menu di samping kiri yang berisi kelola pedagang dan juga tambah pedagang. *Icon* pada header berfungsi untuk melakukan *logout*. Antarmuka halaman ini dapat dilihat pada gambar 4.48.



Gambar 4. 48 antarmuka beranda(kelola pedagang)

c. Halaman Tambah Pedagang

Halaman tambah pedagang merupakan halaman yang dapat diakses *admin* setelah memilih menu tambah pedagang. Pada halaman ini berisi *form* untuk mengisi data dari pedagang yang akan didaftarkan ke dalam sistem pemesanan sayur. Antarmuka halaman ini dapat dilihat pada gambar 4.49.



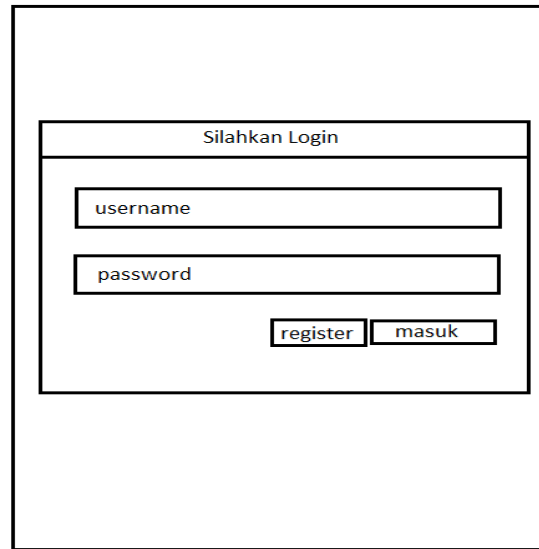
Gambar 4. 49 antarmuka halaman tambah pedagang

2. Perancangan Antarmuka Pedagang

Antarmuka pedagang pada sistem ini merupakan *web responsive* yang dapat dibuka pada *browser* pc dan handphone. Dengan *web responsive* memudahkan dan memberikan kenyamanan pengguna pada saat membuka *web*, karena *web* dapat menyesuaikan ukuran dari layar.

a. Halaman *login*

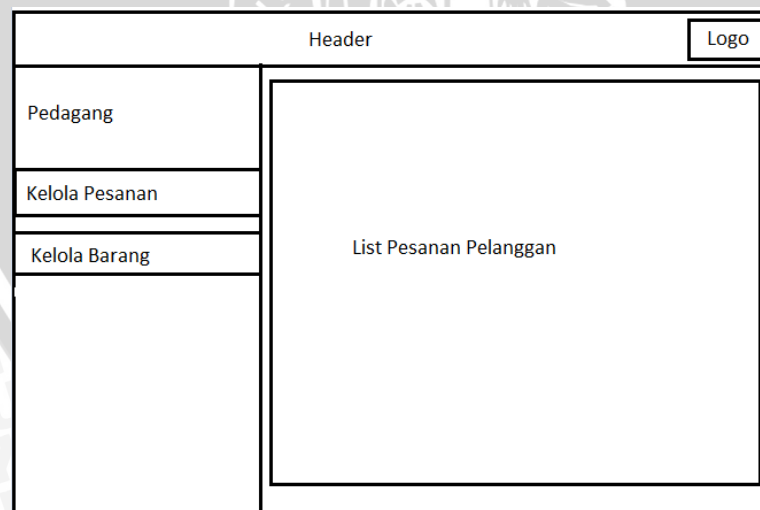
Halaman *login* merupakan halaman yang pertama kali muncul saat membuka sistem ini. Halaman *login* pedagang dan pembeli merupakan halaman yang sama. Pada halaman *login* berisi *form* untuk memasukkan *username* dan *password* serta tombol masuk. Gambar 4.50 akan menunjukkan antarmuka dari halaman *login*.



Gambar 4. 50 antarmuka halaman *login*

b. Halaman Beranda(Kelola Pesanan)

Halaman ini merupakan antarmuka dari beranda pedagang setelah melakukan *login*. Pada halaman ini menampilkan daftar pesanan dari pelanggan. Menu di sebelah kiri merupakan *leftside* untuk memudahkan pedagang memilih menu. Antarmuka halaman ini dapat dilihat pada gambar 4.51.



Gambar 4. 51 halaman beranda(kelola pesanan)

c. Halaman Detail Pesanan

Halaman antarmuka ini menunjukkan detail dari pesanan yang dipilih. Halaman ini menunjukkan data dari pemesan berupa nama, email, no handphone serta alamat. Selain data pemesan juga menampilkan barang yang dipesan. Pada bagian bawah juga ditampilkan maps untuk memudahkan pedagang melihat alamat pemesan. Antarmuka halaman ini dapat dilihat pada gambar 4.52.

Header	
Pedagang	Data Pemesan
Kelola Pesanan	List Barang Pesanan
Kelola Barang	
	maps

Gambar 4. 52 antarmuka halaman detail pesanan

d. Halaman Kelola Barang

Halaman kelola barang merupakan antarmuka untuk menampilkan barang dagangan yang dijual oleh pedagang tersebut. Halaman ini berisi *list* barang dagangan serta tombol submit dan edit barang dagangan untuk mengubah data barang. Selain itu ada juga menu kategori untuk memudahkan pedagang mengelompokkan barang dagangan yang ingin dilihat. Antarmuka halaman ini dapat dilihat pada gambar 4.53.

Header	
Pedagang	Kategori
Kelola Pesanan	list barang
Kelola Barang	
	submit

Gambar 4. 53 antarmuka halaman kelola barang

e. Halaman Tambah Barang

Halaman tambah barang merupakan antarmuka pedagang untuk menambah barang dagangan. Halaman ini berisi *form* untuk mengisi data barang dagangan, juga berisi tombol submit untuk proses memasukkan data ke dalam *database*. Gambar 4.54 menunjukkan antarmuka halaman tambah barang.

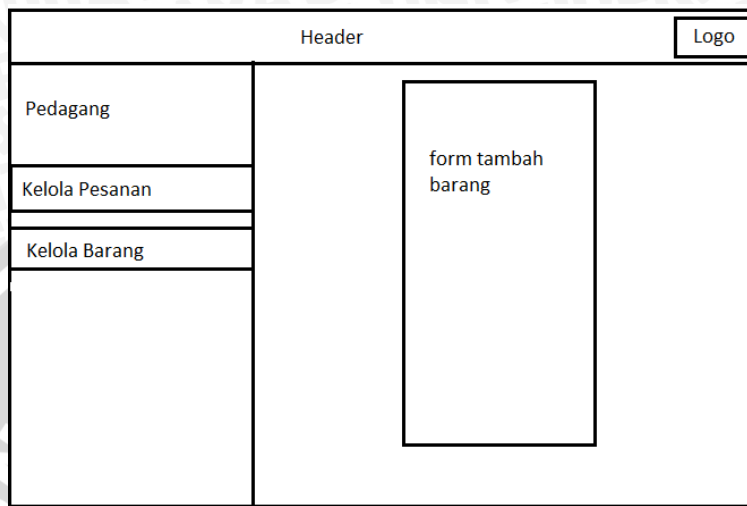


Diagram of the 'Tambah Barang' (Add Item) interface. It features a header bar with a 'Logo' button on the right. Below the header is a sidebar menu with three items: 'Pedagang', 'Kelola Pesanan', and 'Kelola Barang'. The main content area is divided into two sections: a large central area labeled 'form tambah barang' and a smaller area on the right.

Gambar 4. 54 antarmuka halaman tambah barang

f. Halaman Detail Barang

Halaman detail barang merupakan antarmuka untuk menampilkan detail data barang. Untuk masuk halaman ini dibutuhkan aksi menekan baris nama barang. Pada halaman ini berisi data barang dan juga tombol edit untuk mengubah data barang. Gambar 4.55 menunjukkan antarmuka halaman detail barang.

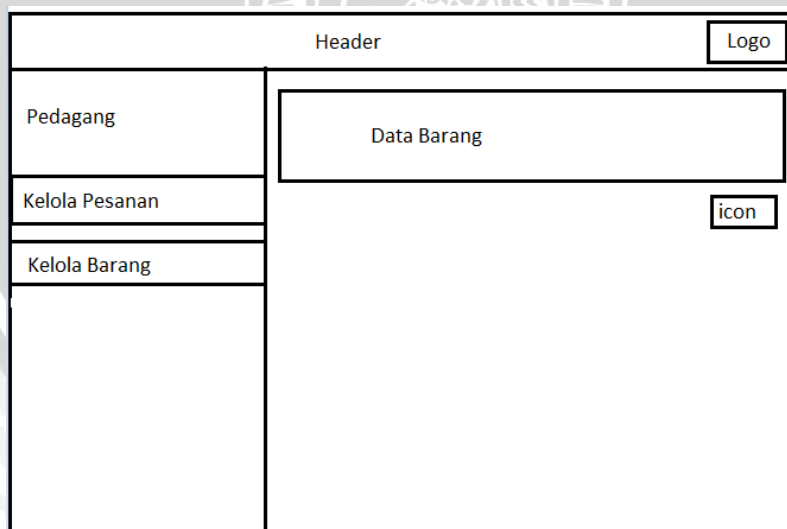


Diagram of the 'Detail Barang' (Item Detail) interface. It features a header bar with a 'Logo' button on the right. Below the header is a sidebar menu with three items: 'Pedagang', 'Kelola Pesanan', and 'Kelola Barang'. The main content area is divided into two sections: a large central area labeled 'Data Barang' and a smaller area on the right containing an 'icon' button.

Gambar 4. 55 antarmuka halaman detail barang.

g. Halaman Edit Data Barang

Halaman edit data barang ini merupakan antarmuka untuk mengubah atau mengganti data barang yang ingin diganti. Pada halaman ini berisi *form* data barang dan tombol

submit yang berada dibawah *form*. Gambar 4.56 menunjukkan antarmuka halaman edit data barang.

Header		Logo
Pedagang		edit data pedagang
Kelola Pesanan		
Kelola Barang		

Gambar 4. 56 antarmuka halaman edit data barang

h. Halaman Profil Pedagang

Halaman profil merupakan antarmuka untuk menampilkan data diri pedagang. Pada halaman ini terdapat informasi data pedagang dan juga tombol edit untuk mengganti data pedagang. Gambar 4.57 menunjukkan antarmuka halaman profil pedagang.

Header		Logo
Pedagang		data pedagang
Kelola Pesanan		
Kelola Barang		
	icon	

Gambar 4. 57 antarmuka halaman profil pedagang

i. Halaman Edit Data Pedagang

Halaman edit data pedagang merupakan antarmuka untuk mengubah data pedagang. Pada halaman ini berisi *form* untuk mengganti data diri pedagang dan tombol submit. Gambar 4.58 menunjukkan antamuka halaman edit data pedagang.

Header		Logo
Pedagang	edit data pedagang	
Kelola Pesanan		
Kelola Barang		

Gambar 4. 58 antarmuka halaman edit data pedagang.

3. Perancangan Antarmuka Pembeli (Pelanggan)

Antarmuka pembeli(pelanggan) pada sistem ini merupakan *web responsive* yang dapat dibuka pada *browser* pc dan handphone. Dengan *web responsive* memudahkan dan memberikan kenyamanan pengguna pada saat membuka *web*, karena *web* dapat menyesuaikan ukuran dari layar.

a. Halaman Login

Halaman *login* merupakan halaman yang pertama kali muncul saat membuka sistem ini. Halaman *login* pembeli atau pelanggan juga sama dengan halaman *login* pedagang. Pada halaman *login* berisi *form* untuk memasukkan *username* dan *password* serta tombol masuk. Gambar 4.59 akan menunjukkan antarmuka dari halaman *login*.

Silahkan Login	
username	
password	
register	masuk

Gambar 4. 59 menunjukkan antarmuka halaman *login* pelanggan.

b. Halaman *Register*

Halaman *register* merupakan antarmuka guest yang diasumsikan sebagai pelanggan. Pada halaman ini berisi *form* untuk mengisi data diri yang berfungsi untuk melengkapi persyaratan mendaftar ke dalam sistem. *Form* pada halaman ini berupa *username*, *realname*, *password*, *email*, nomor handphone, alamat dan maps untuk memilih alamat pelanggan. Gambar 4.60 merupakan antarmuka *register*.

Gambar 4. 60 merupakan antarmuka *register*.

c. Halaman Beranda(Kelola Pesanan)

Halaman beranda merupakan antarmuka pelanggan yang muncul setelah melakukan *login*. Pada halaman ini berisi daftar barang pesanan yang pernah dipesan. Disebelah kiri halaman terdapat *leftside* yang berfungsi untuk memudahkan pelanggan memilih menu kelola pesanan, pilih pedagang dan melakukan *logout* atau keluar dari sistem. Pada pojok kanan atas juga terdapat *icon* untuk memilih pedagang yang fungsinya sama dengan *leftside*. Gambar 4.61 menunjukkan antarmuka halaman beranda.

Gambar 4. 61 menunjukkan antarmuka halaman beranda.

d. Halaman Detail Pesanan

Halaman detail pesanan merupakan antarmuka ketika pelanggan memilih pesanan untuk melihat detail pesanan. Halaman ini berisi data pedagang, daftar barang pesanan dan map yang berfungsi untuk menunjukkan alamat pedagang. Gambar 4.62 menunjukkan antarmuka halaman detail pesanan.

+ Pilih Pedagang	
Pembeli	Data Pedagang
Kelola Pesanan	List Barang Pesanan
Pilih Pedagang	
Logout	
	maps

Gambar 4. 62 antarmuka halaman detail pesanan.

e. Halaman Pilih Pedagang

Halaman pilih pedagang merupakan antarmuka untuk menunjukkan daftar pedagang yang ada. Pada halaman ini hanya menampilkan daftar pedagang saja, untuk memilih pedagang cukup menekan baris pedagang yang diinginkan. Gambar 4.63 menunjukkan antarmuka halaman pilih pedagang.

Pembeli	List Pedagang
Kelola Pesanan	
Pilih Pedagang	
Logout	

Gambar 4. 63 antarmuka halaman pilih pedagang.

f. Halaman *List* Barang Pedagang

Halaman *list* barang pedagang merupakan antarmuka untuk melihat dan memilih barang dagangan pedagang yang dipilih. Halaman ini berisi daftar barang

dagangan, kategori pada bagian atas untuk memudahkan mengelompokkan barang dan pesanan khusus untuk memesan barang yang tidak ada dalam daftar. Sedangkan pada pojok kiri atas merupakan *icon* keranjang belanja. Gambar 4.64 menunjukkan antarmuka halaman *list* barang pedagang.

Header	
Pembeli	Kategori
Kelola Pesanan	pesanan khusus
Pilih Pedagang	list barang
Logout	

Gambar 4. 64 menunjukkan antarmuka halaman *list* barang pedagang.

g. Halaman Pesanan Khusus

Halaman pesanan khusus merupakan antarmuka untuk menampilkan *form* pesanan ketika barang yang diinginkan tidak ada pada daftar barang jualan pedagang. Pada halaman ini berisi *form* pesanan dan tombol submit, pelanggan dapat sekaligus memesan banyak barang pada halaman ini. Gambar 4.65 menunjukkan antarmuka halaman *list* barang pedagang.

Header	
Pembeli	Kategori
Kelola Pesanan	pesanan khusus
Pilih Pedagang	list barang
Logout	

Gambar 4. 65 antarmuka halaman *list* barang pedagang.

h. Halaman Keranjang Belanja

Halaman keranjang belanja merupakan antarmuka untuk menunjukkan keranjang belanja. Halaman keranjang belanja ini menampilkan daftar data barang yang dipesan dan dimasukkan keranjang oleh pelanggan. Data ditampilkan dalam

bentuk tabel dan terdapat tombol hapus barang, hapus keranjang dan pesan. Gambar 4.67 menunjukkan antarmuka keranjang belanja.

Header												
Pembeli	<table border="1"> <thead> <tr> <th colspan="4">Keranjang Belanja</th> </tr> </thead> <tbody> <tr> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>				Keranjang Belanja							
Keranjang Belanja												
Kelola Pesanan												
Pilih Pedagang												
Logout												

Gambar 4. 66 menunjukkan antarmuka keranjang belanja

4.5 Skenario Pengujian

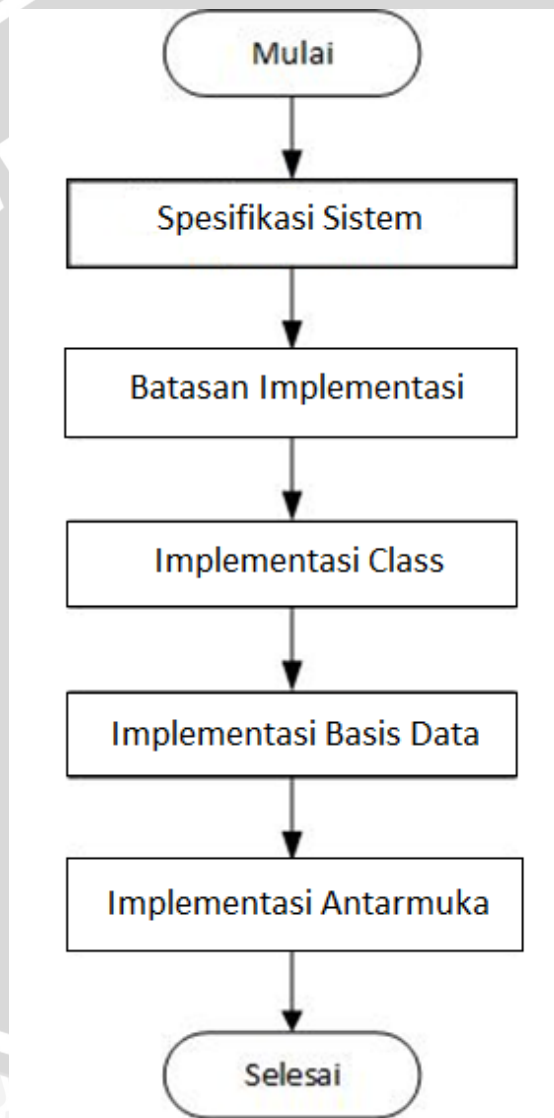
Dalam subbab ini akan dijelaskan skenario pengujian yang akan dilakukan untuk pengujian performa. Skenario pengujian akan ditunjukkan pada tabel 4.28.

Tabel 4. 26 Skenario Pengujian

Aktivitas	Deskripsi
Identify Test Environment	Menggunakan Rational Performance tester sebagai kakas bantu pengujian,
Identify Performance Acceptance Criteria	1. Setiap halaman pada aplikasi harus mampu menampilkan halaman tidak lebih dari 10 detik. 2. Aplikasi mampu menjalankan fungsinya dengan baik.
Plan and Design Tests	Membuat mekanisme pengujian dan menjelaskan fitur atau fungsi yang akan diuji.
Configure the Test Environment	Pengujian dilakukan pada pagi hari antara jam 5 pagi sampai dengan jam 7 pagi dengan menggunakan 1 user.
Implement the Test Design	Melakukan test record pada aplikasi menggunakan rational performance testing.
Execute the Test	Melakukan test schedule dengan menggunakan hasil dari test record sebelumnya
Analyze Results, Report, and Retest	Menganalisa hasil dari pengujian

BAB 5 IMPLEMENTASI

Pada bab ini akan membahas implementasi aplikasi pemesanan sayur berdasarkan analisis kebutuhan dan perancangan yang telah dilakukan sebelumnya di bab 4. Pembahasan pada bab ini terdiri dari penjelasan spesifikasi perangkat keras dan sistem yang dibutuhkan dalam tahap implementasi. Selain itu, terdapat juga penjelasan mengenai batasan-batasan implementasi, implementasi *class*, implementasi *database*, dan implementasi kode program.



Gambar 5. 1 Diagram Alir Implementasi

5.1 Spesifikasi Sistem

Spesifikasi sistem dibagi menjadi 2 jenis, yaitu spesifikasi perangkat keras dan spesifikasi perangkat lunak

5.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan dalam implementasi aplikasi pemesanan sayur dapat dilihat pada tabel 5.1 berikut.

Tabel 5. 1 Spesifikasi Perangkat Keras

Komponen	Spesifikasi
<i>Sistem Model</i>	Asus A46 C
<i>Processor</i>	Intel(R) Core(TM) i5-3337 CPU 1.80 Ghz
<i>Memory</i>	4096 MB DDR3
<i>Display</i>	NVIDIA GeForce 740M

5.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam proses implementasi aplikasi pemesanan sayur dijelaskan pada Tabel 5.2.

Tabel 5. 2 Spesifikasi Perangkat Lunak

Komponen	spesifikasi
Sistem operasi	<i>Windows 8.1 Pro</i>
Bahasa pemrograman	<i>PHP+HTML</i>
<i>Tools Developmnet</i>	<i>Sublime text, Enterprise Architect, Word 2010, Android Studio, Visio 2013, Rational Performance tester</i>
<i>Tools Server</i>	<i>XAMPP</i>
DBMS	<i>MySQL</i>
<i>Web Browser</i>	<i>Mozilla Firefox, Google Chrome</i>

5.2 Batasan Implementasi

Pada tahap implementasi aplikasi pemesanan sayur memiliki beberapa batasan proses pengerjaan, antara lain:

1. Aplikasi pemesanan sayur di implementasikan dengan bahasa Pemrograman *PHP* dan *HTML*.
2. Penyimpanan data menggunakan *database MySQL*.
3. Aplikasi dapat berjalan dengan baik apabila ada koneksi *internet*.
4. Menggunakan MVC dengan *framework Codeigniter*.
5. Terdiri dari 3 aktor dan 3 halaman yang berbeda.

5.3 Implementasi Class

Class diagram dari sistem telah dirancang pada tahap perancangan sistem dan penjelasannya, akan dijelaskan dalam Tabel 5.3 tentang *class Controller*. Sedangkan implementasi *class Model* dijelaskan dalam Tabel 5.4.

Tabel 5. 3 Implementasi Class Controller

Nama Package	Nama Class	Nama File
<i>Controller</i>	<i>ControllerDashboard</i>	<i>ControllerDasboard.php</i>
<i>Controller</i>	<i>ControllerPedagang</i>	<i>ControllerPedagang.php</i>
<i>Controller</i>	<i>ControllerPelanggan</i>	<i>ControllerPelanggan.php</i>
<i>Controller</i>	<i>ControllerKeranjang</i>	<i>ControllerKeranjang.php</i>
<i>Controller</i>	<i>Controller_Server</i>	<i>Controller_server.php</i>

Tabel 5.3 berisi daftar *Controller* pada sistem pemesanan sayur. *ControllerDashboard* merupakan *Controller* yang dipanggil pertama kali pada saat sistem dijalankan untuk memanggil halaman *login*, halaman *register*. *ControllerPedagang* untuk memanggil fungsi yang berhubungan dengan halaman pedagang. *ControllerPelanggan* untuk memanggil fungsi yang berhubungan dengan halaman pelanggan. *ControllerKeranjang* untuk memanggil fungsi yang berhubungan dengan halaman keranjang. *ControllerServer* untuk memanggil fungsi yang berhubungan dengan halaman *admin*.

Tabel 5. 4 Implementasi Class Model

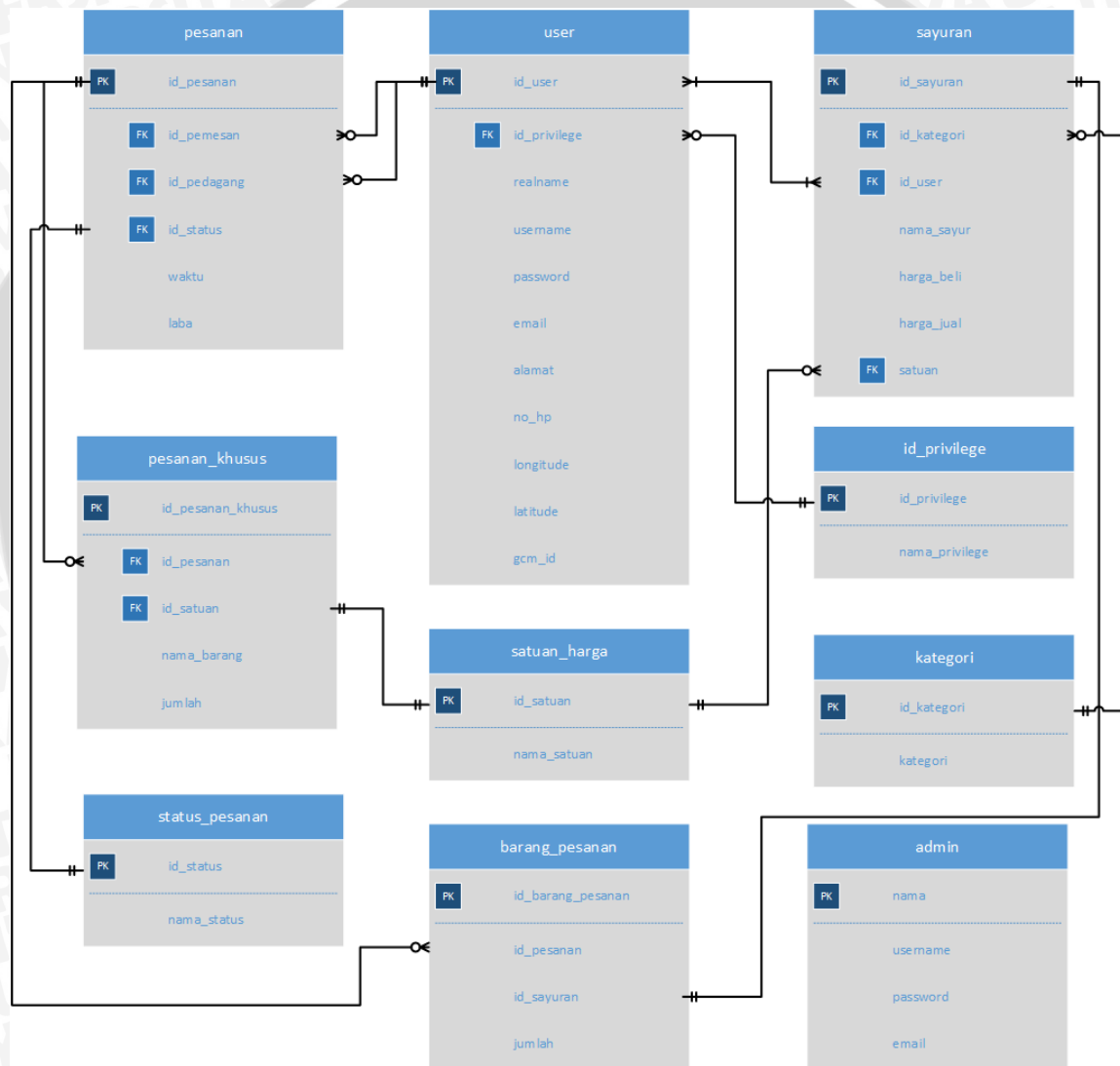
Nama Package	Nama Class	Nama File
<i>Model</i>	<i>M_login</i>	<i>m_login.php</i>
<i>Model</i>	<i>M_Pedagang</i>	<i>m_Pedagang.php</i>
<i>Model</i>	<i>M_pelanggan</i>	<i>m_Pelanggan.php</i>
<i>Model</i>	<i>Proses_server</i>	<i>Proses_server.php</i>

Tabel 5.4 berisi daftar *Controller* pada sistem pemesanan sayur yang berhubungan dengan *database*. *M_login* berfungsi untuk cek data ke *database* pada saat login, *ModelPedagang* berfungsi untuk pertukaran data dari *database* yang berhubungan dengan pedagang, *ModelPelanggan* berfungsi untuk pertukaran data dari *database* yang berhubungan dengan pelanggan. *Proses_server* berfungsi untuk pertukaran data dari *database* yang berhubungan dengan *admin*.

5.4 Implementasi Basis Data

Implementasi basis data dilakukan berdasarkan perancangan sistem dengan menggunakan *MySQL*. Basis data yang telah diimplementasikan dapat dilihat pada Gambar 5.2. Dari gambar 5.2 dapat diketahui terdapat beberapa relasi antar tabel yaitu tabel pesanan yang berelasi dengan tabel pesanan_khusus, status_pesanan

tabel user dan tabel barang_pesanan. Tabel user berelasi dengan tabel sayuran, tabel id_privilege dan tabel pesanan. Tabel sayuran berelasi dengan tabel user, tabel satuan_harga, tabel kategori dan tabel barang_pesanan. Tabel id_privilege berelasi dengan tabel user. Tabel kategori berelasi dengan tabel sayuran. Tabel admin tidak memiliki relasi dengan tabel yang lain. Tabel satuan harga berelasi dengan tabel sayuran dan tabel pesanan_khusus. Tabel barang_pesanan berelasi dengan tabel sayuran dan tabel pesanan. Tabel status_pesanan berelasi dengan tabel pesanan dan yang terakhir tabel pesanan khusus berelasi dengan tabel satuan_harga dan tabel pesanan. Selanjutnya setiap tabel akan dijelaskan pada subbab implementasi basisdata.



Gambar 5. 2 Implementasi Database

1. Tabel Admin

admin
nama VARCHAR(20)
username VARCHAR(20)
password VARCHAR(20)
email VARCHAR(20)

Gambar 5. 3 Impementasi tabel *admin*

Pada gambar 5.3 ditunjukkan type data dan atribut dari tabel *admin*. Pada tabel ini terdapat beberapa atribut, di antaranya nama, *username*, *password* dan email, atribut nama berfungsi untuk menyimpan nama asli dari *admin*, *username* berfungsi untuk menyimpan id untuk masuk ke dalam sistem, *password* untuk menyimpan *password* dari *admin* untuk masuk ke dalam sistem dan atribut email untuk menyimpan alamat email dari *admin*.

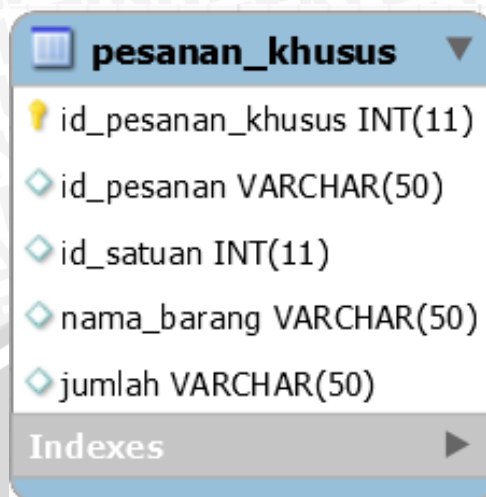
2. Tabel pesanan

pesanan
id_pesanan VARCHA...
id_pemesan INT(11)
id_pedagang INT(11)
id_status INT(11)
waktu DATETIME(100)
laba INT(11)
Indexes

Gambar 5. 4 implementasi tabel *pesanan*

Pada gambar 5.4 ditunjukkan type data dan atribut dari tabel *pesanan*. Atribut dari tabel ini yaitu, *id_pesanan* sebagai *primary key* untuk menyimpan id pesanan pembeli, *id_pemesan* sebagai *foreign key* yang mereference ke tabel *user* untuk menyimpan id pemesan, *id_pedagang* sebagai *foreign key* yang mereferences ke tabel *user* untuk menyimpan id pedagang yang dipilih, *id_status* sebagai *foreign key* untuk menyimpan status pesanan, waktu untuk menyimpan data waktu pada saat si pemesan melakukan pemesanan dan laba untuk menyimpan laba hasil penjualan.

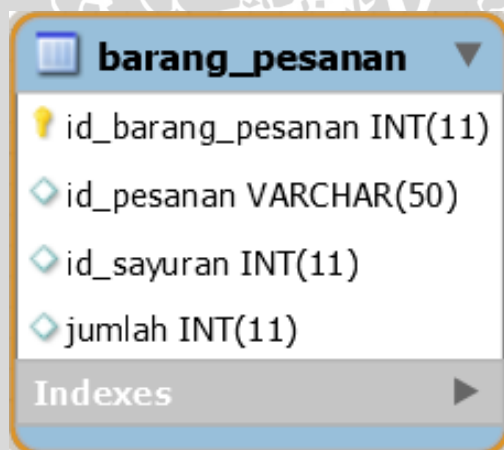
3. Tabel Pesanan Khusus



Gambar 5. 5 Implementasi tabel Pesanan Khusus

Pada gambar 5.5 di tunjukkan type data dan atribut dari tabel pesanan khusus yang berfungsi untuk menyimpan data pesanan khusus. Atribut dari tabel ini yaitu `id_pesanan_khusus` sebagai *primary key* untuk menyimpan id pesnaan khusus, `id_pesanan` sebagai *foreign key* untuk menyimpan id pesanan yang diperoleh dari tabel pesanan, `id_satuan` sebagai *foreign key* untuk menyimpan id satuan, `nama_barang` untuk menyimpan barang yang dipesan dan `jumlah` untuk menyimpan jumlah pesanan.

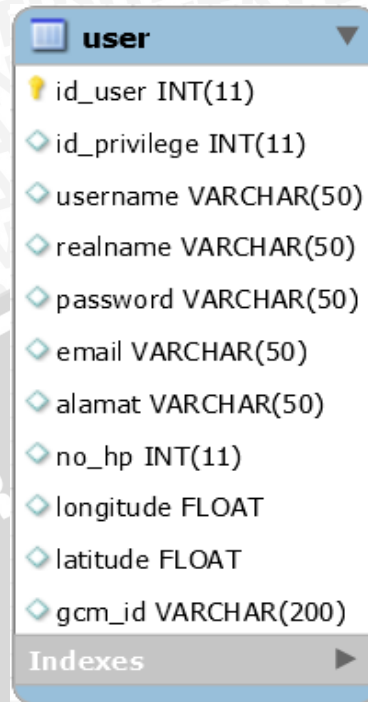
4. Tabel Barang Pesanan



Gambar 5. 6 Implementasi tabel Barang Pesanan

Gambar 5.6 menunjukkan tabel barang pesanan yang berisi atribut dan tipe data tabel barang pesanan. `Id_barang_pesanan` sebagai *primary key* untuk menyimpan id barang, `id_pesanan` sebagai *foreign key* untuk menyimpan id pesanan yang diperoleh dari tabel pesanan, `id_sayuran` sebagai *foreign key* untuk menyimpan id sayuran dan `jumlah` untuk menyimpan jumlah barang pesanan.

5. Tabel *User*

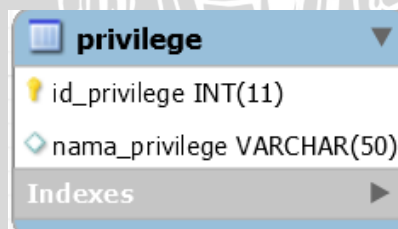


user	
id_user	INT(11)
id_privilege	INT(11)
username	VARCHAR(50)
realname	VARCHAR(50)
password	VARCHAR(50)
email	VARCHAR(50)
alamat	VARCHAR(50)
no_hp	INT(11)
longitude	FLOAT
latitude	FLOAT
gcm_id	VARCHAR(200)
Indexes	

Gambar 5. 7 implementasi tabel *user*

Pada gambar 5.7 dijelaskan bahwa tabel *user* berfungsi untuk menyimpan data dari *user* atau pembeli yang telah mendaftar ke dalam sistem. Atribut dari tabel *user* antara lain, *id_user* sebagai *primary key* untuk menyimpan id user, *id_privillage* sebagai *foreign key* yang mereferences ke tabel *privillage* untuk menyimpan id privilege, *realname* untuk menyimpan nama pembeli, *username* untuk menyimpan id untuk masuk ke dalam sistem, *password* untuk kata sandi masuk sistem, *email* untuk menyimpan alamat email, *alamat* untuk menyimpan alamat tempat tinggal pembeli, dan *no_hp* untuk menyimpan nomor hp pembeli.

6. Tabel *Privilege*



privilege	
id_privilege	INT(11)
nama_privilege	VARCHAR(50)
Indexes	

Gambar 5. 8 implementasi tabel *privillage*

Pada gambar 5.8 ditunjukkan bahwa tabel *privillage* memiliki 2 atribut yaitu *id_privillage* sebagai *primary key* untuk menyimpan id privilege dan *nama_privillage* sebagai indikator nama *privillage*. Tabel ini berfungsi untuk menyimpan nilai atau batasan tiap *user*.

7. Tabel Sayuran

sayuran	
id_sayuran	INT(11)
id_kategori	INT(11)
id_user	INT(11)
nama_sayur	VARCHAR(50)
sayurancol	VARCHAR(50)
harga_jual	VARCHAR(50)
satuan	VARCHAR(50)
Indexes	

Gambar 5. 9 implementasi tabel sayuran

Pada gambar 5.9 ditunjukkan bahwa tabel sayuran berfungsi untuk menyimpan data sayuran, tabel ini memiliki beberapa atribut yaitu, id_sayur sebagai *primary key*, id_user sebagai *foreign key*, nama sayur untuk menyimpan nama sayur, id_kategori sebagai *foreign key* untuk menyimpan id kategori, harga jual dan harga beli untuk menyimpan harga dan satuan sebagai *foreign key* untuk menyimpan id satuan.

8. Tabel Satuan Harga

satuan_harga	
id_satuan_harga	INT(11)
nama_satuan	VARCHAR(50)
Indexes	

Gambar 5. 10 Implementasi tabel satuan harga

Pada gambar 5.10 ditunjukkan atribut dan tipe data dari tabel satuan harga. Atribut dari tabel ini terdiri dari 2, yaitu id_satuan sebagai *primary key* untuk menyimpan id satuan dan nama_satuan untuk menyimpan nama satuan yang ditentukan.

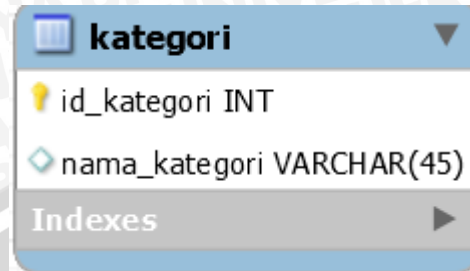
9. Tabel Status Pesanan

status_pesanan	
id_status_pesanan	INT(1)
nama_status	VARCHAR(50)
Indexes	

Gambar 5. 11 Implementasi tabel Status Pesanan

Gambar 5.11 menunjukkan isi dari tabel status pesanan. Atribut dari tabel ini yaitu id_status sebagai *primary key* untuk menyimpan id status dan nama status untuk menyimpan nama status.

10. Tabel Kategori



Gambar 5. 12 Implementasi tabel kategori

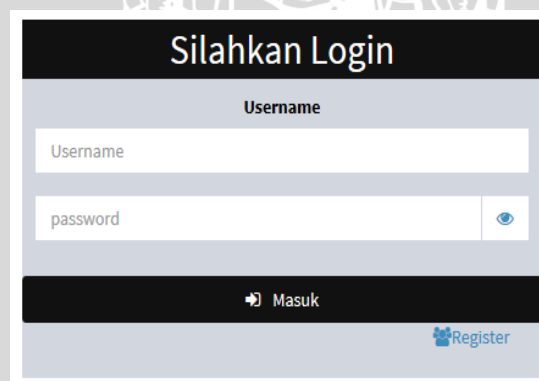
Gambar 5.12 menunjukkan bahwa tabel ini menyimpan data kategori. Atribut tabel ini yaitu id_kategori sebagai *primary key* untuk menyimpan id kategori dan nama kategori untuk menyimpan nama kategori.

5.5 Implementasi Antarmuka

Pada sub bab ini akan menjelaskan dan menunjukkan implementasi antarmuka sistem pemesanan sayur.

a. Halaman login

Halaman *login* menampilkan antarmuka yang digunakan *user* untuk masuk ke dalam sistem. Halaman *login admin* berbeda dengan halaman *login guest*. Antarmuka halaman *login Guest* ditunjukkan pada gambar 5.13 dan gambar 5.14 menunjukkan halaman *login admin*.



Gambar 5. 13 Implementasi Antarmuka Halaman Login Guest

Gambar 5. 14 Implementasi Antarmuka Halaman *Login Admin*

b. Halaman *Register*

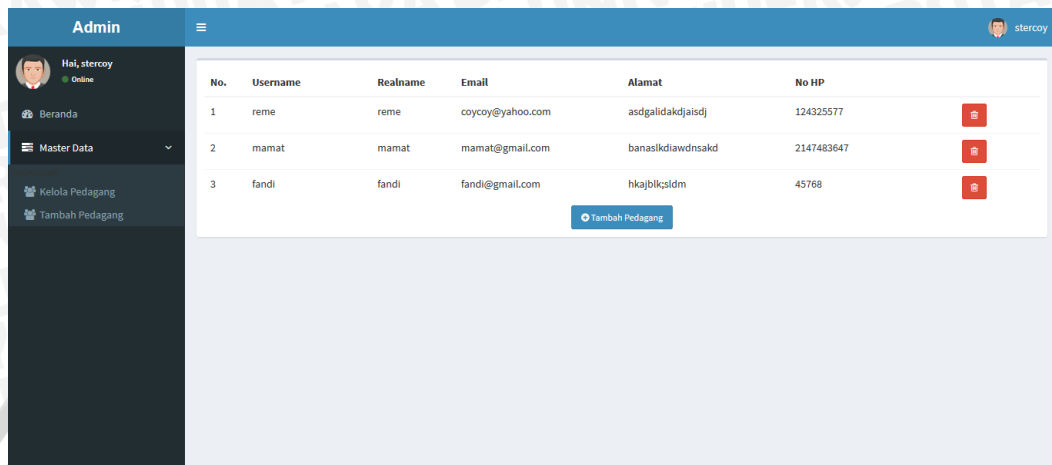
Halaman *register* berisi *form* untuk memasukkan data ke dalam sistem, karena sebelum bisa menggunakan aplikasi pemesanan sayur ini diharuskan untuk mendaftar ke dalam sistem terlebih dahulu. Antarmuka halaman *Register* akan ditunjukkan pada gambar 5.15 dan gambar 5.16.

Gambar 5. 15 Implementasi Antarmuka Halaman *Register*

Gambar 5. 16 Implementasi Antarmuka Halaman *Register*

c. Halaman Beranda *Admin*

Halaman beranda *admin* merupakan antarmuka setelah *admin* masuk ke dalam sistem. Setiap *user* memiliki halaman beranda yang berbeda-beda. Halaman beranda *admin* dan kelola pedagang merupakan halaman yang sama. Antarmuka halaman beranda *admin* atau kelola pedagang dapat dilihat pada gambar 5.17.



Gambar 5. 17 Implementasi Antarmuka Halaman Beranda *Admin*

d. Halaman Tambah Pedagang

Halaman tambah pedagang merupakan halaman yang menampilkan antarmuka berupa *form* untuk memasukkan data pedagang yang akan dimasukkan ke dalam sistem supaya dapat masuk ke dalam sistem sebagai pedagang. Antarmuka halaman tambah pedagang akan ditunjukkan pada gambar 5.18.

Tambah Pedagang

Username Realname

Password Email

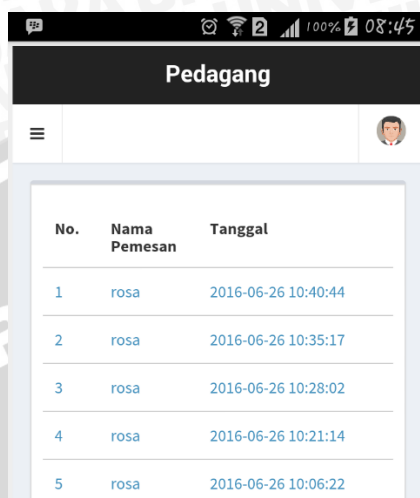
Address No HP

Submit

Gambar 5. 18 Implementasi antarmuka Halaman Tambah Pedagang

e. Halaman Antarmuka Beranda Pedagang (Kelola Pesanan)

Halaman beranda pedagang merupakan halaman pertama setelah melakukan *login*. Halaman ini juga sebagai halaman kelola pesanan yang menampilkan *list* pemesanan dan tanggal memesan para pelanggan. Antarmuka Kelola Pesanan ditunjukkan pada gambar 5.19.

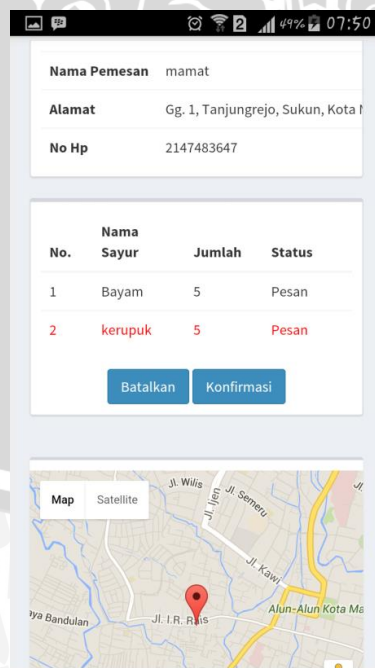


No.	Nama Pemesan	Tanggal
1	rosa	2016-06-26 10:40:44
2	rosa	2016-06-26 10:35:17
3	rosa	2016-06-26 10:28:02
4	rosa	2016-06-26 10:21:14
5	rosa	2016-06-26 10:06:22

Gambar 5. 19 Implementasi antarmuka Kelola Pesanan

f. Halaman Antarmuka Detail Pesanan Pedagang

Halaman ini merupakan antarmuka detail pesanan, diakses setelah memilih pemesanan yang ingin dilihat. Halaman ini berisi detail pesanan dan detail data pemesan serta tombol konfirmasi. Baris berwarna merah merupakan pesanan khusus. Antarmuka halaman ini dapat dilihat pada gambar 5.20.



Nama Pemesan : mamat

Alamat : Gg. 1, Tanjungrejo, Sukun, Kota M

No Hp : 2147483647

No.	Nama Sayur	Jumlah	Status
1	Bayam	5	Pesan
2	kerupuk	5	Pesan

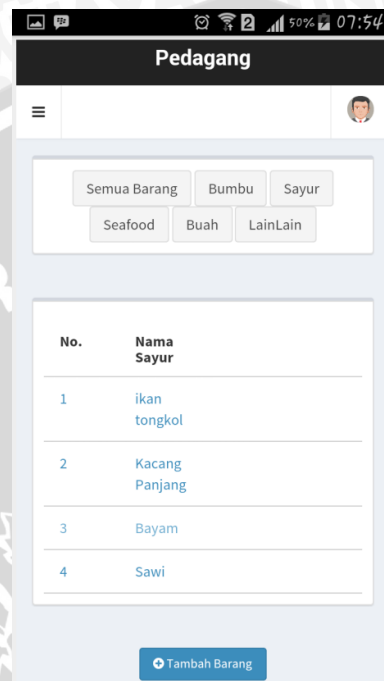
Map Satellite

Jl. Wilis, Jl. Jember, Jl. Kawi, Jl. I.R. Rais, Alun-Alun Kota Me

Gambar 5. 20 Implementasi Antarmuka Detail Pesanan Pedagang.

g. Halaman Antarmuka Kelola Barang

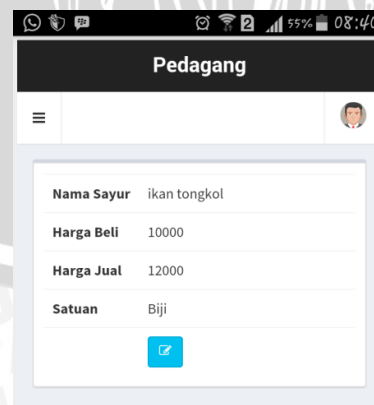
Halaman kelola pedagang merupakan antarmuka yang menampilkan *list* barang dagangan pedagang dan menampilkan kategori barang. Setiap pedagang memiliki barang dan harga yang berbeda-beda. Ketika kategori dipilih maka sistem akan mengelompokkan barang sesuai kategori yang dipilih. Antarmuka halaman kelola barang akan ditampilkan pada gambar 5.21.



Gambar 5. 21 Implementasi Antarmuka Halaman Kelola Barang

h. Halaman Antarmuka Detail Barang

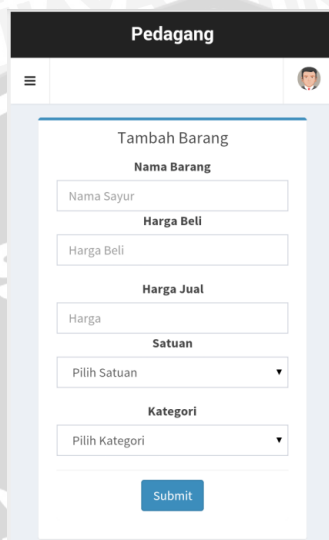
Halaman detail barang merupakan antarmuka ketika barang dipilih yang akan menampilkan halaman yang berisi data lengkap tentang barang dagangan berupa harga, jual harga beli satuan dan tombol edit data barang. Antarmuka halaman ini dapat dilihat pada gambar 5.22.



Gambar 5. 22 Implementasi Antarmuka Halaman Detail Barang

i. Halaman Antarmuka Tambah Barang Dagangan

Halaman antarmuka tambah barang dagangan merupakan halaman antarmuka untuk menampilkan *form* yang berfungsi untuk memasukkan data barang dagangan para pedagang. *Form* yang muncul antara lain nama barang, harga beli, harga jual, satuan dan kategori. Gambar antarmuka tambah barang dagangan akan ditampilkan pada gambar 5.23.

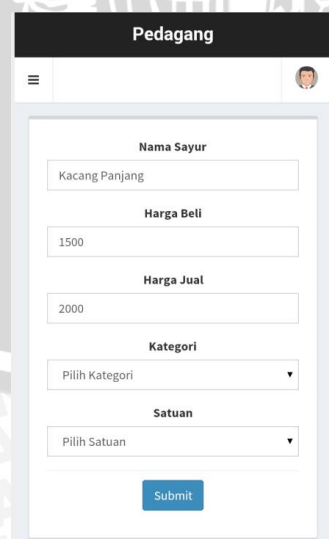


The screenshot shows a mobile application interface for a merchant. At the top, there's a header bar labeled 'Pedagang'. Below it, a menu icon and a user profile icon are visible. The main content area is titled 'Tambah Barang' (Add Item). It contains several input fields: 'Nama Barang' (Item Name) with the placeholder 'Nama Sayur', 'Harga Beli' (Buy Price), 'Harga Jual' (Sell Price) with the placeholder 'Harga', 'Satuan' (Unit) with a dropdown menu showing 'Pilih Satuan', and 'Kategori' (Category) with a dropdown menu showing 'Pilih Kategori'. A blue 'Submit' button is located at the bottom of the form.

Gambar 5. 23 Implementasi Antarmuka Halaman Tambah Pedagang

j. Halaman Edit Data Barang Dagangan

Halaman ini berfungsi untuk mengganti data barang dagangan, misal mengganti harga barang yang bias turun dan naik. Halaman ini akan menampilkan *form* seperti tambah barang, data yang muncul merupakan data barang yang dipilih. Ketika menekan tombol submit maka sistem akan mengupdate data barang. Antarmuka halaman ini dapat dilihat pada gambar 5.24.



The screenshot shows the same mobile application interface as before, but now it's for editing an existing item. The header 'Pedagang' and navigation icons remain. The form is titled 'Edit Data Barang'. The input fields are now populated with data: 'Nama Sayur' contains 'Kacang Panjang', 'Harga Beli' contains '1500', 'Harga Jual' contains '2000', 'Kategori' has a dropdown with 'Pilih Kategori', and 'Satuan' has a dropdown with 'Pilih Satuan'. A blue 'Submit' button is at the bottom.

Gambar 5. 24 Implementasi Antarmuka Halaman Edit Data Barang

k. Halaman Profil Pedagang

Halaman ini berfungsi untuk melihat data semua informasi pedagang, pedagang juga dapat mengubah data diri dengan menekan tombol edit. Data pedagang yang muncul antara lain *username*, *realname*, *password*, email alamat dan no *handphone*. Antarmuka halaman ini dapat dilihat pada gambar 5.25.

Pedagog	
Username	mamat
Realname	mamat
Password	mamat
Email	mamat@gmail.com
Alamat	siguragura5
No Hp	2147483647
Edit Data	

Gambar 5. 25 Implementasi Antarmuka Halaman Profil Pedagang

l. Halaman Update Profil

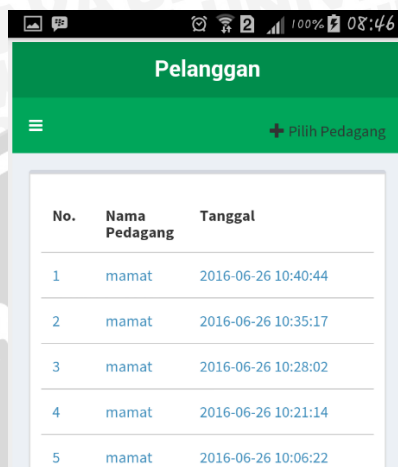
Halaman edit data pedagang merupakan halaman yang berfungsi untuk mengganti data pedagang. Halaman ini berisi *form* data pedagang yang bisa dirubah atau diganti. Halaman ini menampilkan *form* yang sama dengan tambah pedagang, pada halaman ini pedagang dapat mengganti data diri yang ingin diganti. Ketika menekan tombol submit sistem akan meng*update* data. Antarmuka halaman ini dapat dilihat pada gambar 5.26.

Pedagog	
Username	mamat
Realname	mamat
Password	mamat
Email	mamat@gmail.com
Alamat	siguragura5
No HP	2147483647

Gambar 5. 26 Implementasi Halaman Edit Data Pedagang.

m. Halaman Beranda Pelanggan (Kelola Pesanan)

Halaman beranda pedagang merupakan halaman pertama setelah melakukan *login*. Halaman ini juga sebagai halaman kelola pesanan yang menampilkan *list* pedagang dan tanggal memesan para pelanggan. Antarmuka Kelola Pesanan ditunjukkan pada gambar 5.27

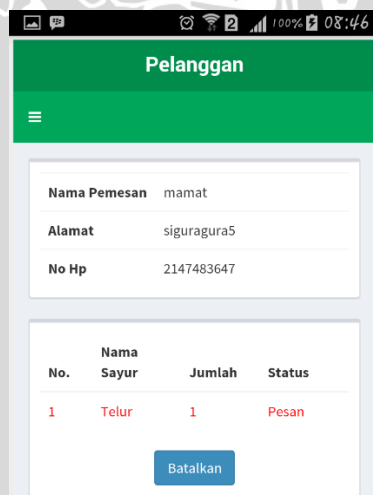


No.	Nama Pedagang	Tanggal
1	mamat	2016-06-26 10:40:44
2	mamat	2016-06-26 10:35:17
3	mamat	2016-06-26 10:28:02
4	mamat	2016-06-26 10:21:14
5	mamat	2016-06-26 10:06:22

Gambar 5. 27 Implementasi Halaman Kelola Pesanan

n. Halaman Detail Pesanan Pelanggan

Halaman detail pesanan merupakan antarmuka ketika pelanggan memilih salah satu pesanan untuk dilihat detailnya. Halaman ini berisi *informasi* detail pesanan biasa, dan pesanan khusus serta *informasi* pedagang yang telah dipilih. Antarmuka halaman ini dapat dilihat pada gambar 5.28.



Nama Pemesan	mamat
Alamat	siguragura5
No Hp	2147483647

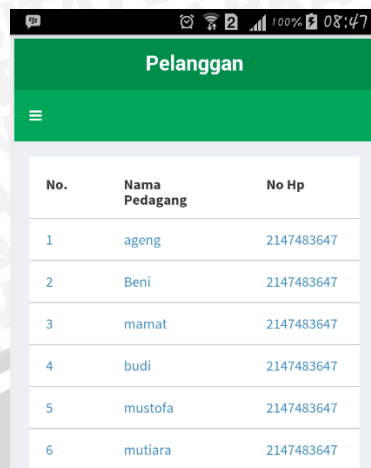
No.	Nama Sayur	Jumlah	Status
1	Telur	1	Pesan

Batalan

Gambar 5. 28 Implementasi Halaman Detail Pesanan

o. Halaman Pilih Pedagang

Halaman pilih pedagang merupakan antarmuka yang menampilkan *list* pedagang yang telah mendaftar ke dalam sistem. Data yang ditampilkan berupa nama pedagang dan juga no *handphone* pedagang. Antarmuka halaman pilih pedagang akan ditunjukkan pada gambar 5.29.

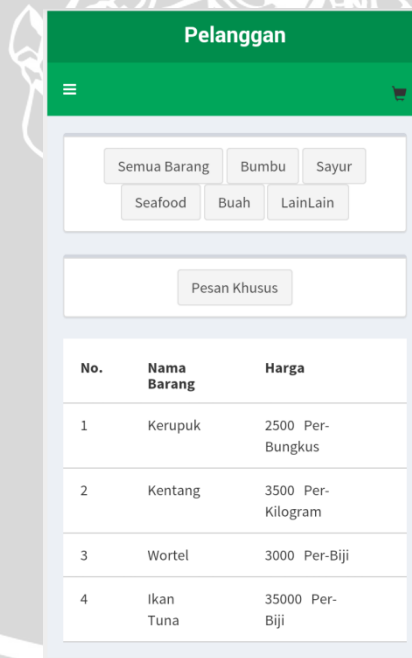


No.	Nama Pedagang	No Hp
1	ageng	2147483647
2	Beni	2147483647
3	mamat	2147483647
4	budi	2147483647
5	mustofa	2147483647
6	mutiara	2147483647

Gambar 5. 29 Implementasi Antarmuka Halaman Pilih Pedagang

p. Halaman *List* Barang Dagangan

Halaman *list* barang dagangan merupakan antarmuka setelah pilih pedagang yang menampilkan list barang dagangan yang dijual oleh tiap pedagang. Selain *list* barang halaman ini menampilkan menu kategori untuk memudahkan pelanggan mengelompokkan barang yang ingin dicari. Menu pesanan khusus di bawah menu kategori berfungsi untuk memesan barang ya tidak ada pada *list* barang. Antarmuka halaman ini akan ditampilkan pada gambar 5.30.



No.	Nama Barang	Harga
1	Kerupuk	2500 Per-Bungkus
2	Kentang	3500 Per-Kilogram
3	Wortel	3000 Per-Biji
4	Ikan Tuna	35000 Per-Biji

Gambar 5. 30 Implementasi Antarmuka *List* Barang Dagangan

q. Halaman Pesanan Khusus

Halaman pesanan khusus merupakan antarmuka untuk menampilkan *form* pesanan saat barang yang ingin dipesan tidak ada pada *list* barang. Antarmuka halaman ini dapat dilihat pada gambar 5.31.

Pelanggan

Menu

Nama Barang

Jumlah Barang

Pilih Satuan

Submit

Gambar 5. 31 Implementasi Halaman Pesanan Khusus

r. Halaman Keranjang Belanja

Halaman ini merupakan antarmuka untuk menampilkan keranjang belanja. Pada halaman ini menampilkan apa saja barang yang telah dimasukkan ke keranjang dan siap dipesan, berupa nama barang, jumlah barang, satuan dan total harga. Untuk melanjutkan memesan barang dapat menekan tombol pesan sekarang. Antarmuka halaman ini dapat dilihat pada gambar 5.32.

Pelanggan

Menu

No.	Nama Barang	Harga
1	ikan tongkol	
	5 Biji	Rp. 60000
2	(pesanan khusus) lempur	
	30 Buah	Rp. -
Total		Rp. 60000

Hapus Keranjang

Pesan

Gambar 5. 32 Implementasi Antarmuka Halaman Keranjang Belanja

BAB 6 PENGUJIAN DAN ANALISA

6.1 Pengujian Fungsional

Pengujian *fungsional* digunakan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. Acuan untuk melakukan pengujian fungsional didapatkan dari daftar kebutuhan fungsional yang telah dirumuskan sebelumnya pada bab 4. Metode yang digunakan pada pengujian fungsional yaitu dengan menggunakan metode pengujian *Black-Box* dan *White-Box*.

6.1.1 Pengujian *White-Box*

Tujuan dilakukannya pengujian *White-Box* yaitu untuk menguji algoritma yang telah diimplementasikan sebelumnya. Algoritma yang diuji merupakan algoritma yang memiliki prioritas tinggi telah diimplementasikan sesuai dengan yang diharapkan. Pengujian *White-Box* melibatkan kode-kode program yang dilakukan dengan cara membuat *flowgraph* kemudian dilanjutkan dengan menghitung *cyclomatic complexity*, lalu menentukan *independent path*.

6.1.2 Pengujian *Black-Box*

Pengujian *Black-Box* ini dilakukan dengan mengamati hasil eksekusi melalui data uji dan memeriksa secara *fungsional* dari perangkat lunak yang telah dibuat perancangan sistem sebelumnya. Tujuan dilakukannya pengujian *Black-Box* adalah untuk memastikan setiap spesifikasi kebutuhan perangkat lunak yang didefinisikan telah sesuai dengan yang diharapkan.

6.2 Performance test

Pengujian performa dilakukan dengan menggunakan *Rational Performance tester* dengan cara menjalankan aplikasi *Rational Performance tester* dan melakukan record pada tiap fitur yang diuji.

6.2.1.1 Mekanisme Pengujian

Pengujian performa dilakukan dengan menguji fitur dari sistem yang tersedia. Perlu dilakukan record pada setiap fitur untuk bisa diuji. Berikut kasus uji untuk pengujian performa.

Tabel 6.1 menjelaskan kasus uji performa sistem pada saat menjalankan fitur register untuk mendaftar kedalam sistem.

Tabel 6. 1 Register

Nomor Kasus Uji	PT_001
Nama Kasus Uji	Register
Deskripsi	Melakukan register untuk mendaftar kedalam sistem

Jumlah User	1
Prosedur Uji	1. Melakukan record fitur register 2. Menjalankan <i>test record</i> 3. Melakukan performance schedule 4. Analisa hasil <i>test</i>
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.2 menjelaskan kasus uji performa aplikasi untuk membuka halaman beranda atau kelola pesanan.

Tabel 6. 2 Kelola Pesanan / Beranda Pelanggan

Nomor Kasus Uji	PT_002
Nama Kasus Uji	Kelola Pesanan / Beranda
Deskripsi	Membuka halaman kelola pesanan / beranda
Jumlah User	1
Prosedur Uji	1. Melakukan record fitur kelola pesanan / beranda dengan membuka menu atau login kedalam sistem. 2. Menjalankan <i>test record</i> 3. Melakukan performance schedule 4. Menganalisa hasil <i>test</i>
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.3 menjelaskan kasus uji aplikasi untuk membuka detail pesanan pelanggan untuk melihat detail pesanan yang telah dipesan.

Tabel 6. 3 Detail Pesanan Pelanggan

Nomor Kasus Uji	PT_003
Nama Kasus Uji	Detail Pesanan Pelanggan.
Deskripsi	Membuka halaman detail pesanan pelanggan.
Jumlah User	1
Prosedur Uji	1. Melakukan record fitur detail pesanan pelanggan dengan membuka halaman detail pesanan pelanggan. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule 4. Menganalisa hasil <i>test</i>
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.4 menjelaskan kasus uji untuk mengkonfirmasi barang yang sudah dikirimkan oleh pedagang. Untuk mengkonfirmasi barang pesanan dapat melalui halaman detail pesanan.

Tabel 6. 4 Konfirmasi Pesanan Pelanggan

Nomor Kasus Uji	PT_004
Nama Kasus Uji	Konfirmasi PesananPelanggan
Deskripsi	Membuka halaman detail pesanan dan melakukan konfirmasi
Jumlah User	1
Prosedur Uji	1. Melakukan record saat melakukan konfirmasi pesanan. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.5 menjelaskan kasus uji pesanan khusus. Pesanan khusus digunakan untuk memesan barang saat barang yang diinginkan pengguna tidak ada dalam daftar barang dagangan penjual atau pedagang.

Tabel 6. 5 Pesanan Khusus.

Nomor Kasus Uji	PT_005
Nama Kasus Uji	Pesanan Khusus
Deskripsi	Membuka halaman pesanan khusus dan memesan.
Jumlah User	1
Prosedur Uji	1. Melakukan record fitur pesanan khusus dengan membuka halaman pesanan khusus dan memesan barang dilanjutkan dengan membuka keranjang belanja dan melakukan proses pesan. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.6 menjelaskan kasus uji untuk membuka halaman beranda pedagang atau kelola pesanan pedagang. Untuk membuka halaman ini diharuskan masuk kedlaam sistem atau membuka halaman kelola pesanan.

Tabel 6. 6 Beranda Pedagang / Kelola Pesanan Pedagang.

Nomor Kasus Uji	PT_006
Nama Kasus Uji	Beranda Pedagang / Kelola Pesanan Pedagang
Deskripsi	Membuka halaman beranda pedagang
Jumlah User	1
Prosedur Uji	1. Melakukan record fitur pesanan khusus dengan membuka halaman pesanan khusus. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.7 menjelaskan kasus uji untuk membuka menu detail pesanan. Detail pesanan menampilkan detail pesanan barang yang dipesan oleh pelanggan.

Tabel 6. 7 Detail Pesanan Pedagang

Nomor Kasus Uji	PT_007
Nama Kasus Uji	Detail Pesanan Pedagang
Deskripsi	Melihat Detail Pesanan
Jumlah User	1
Prosedur Uji	1. Melakukan record membuka halaman detail pesanan. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.8 menjelaskan kasus uji untuk melakukan konfirmasi pesanan oleh pedagang. Setelah barang dikonfirmasi akan muncul notifikasi pada *handpone*.

Tabel 6. 8 Konfirmasi Pesanan Pedagang.

Nomor Kasus Uji	PT_008
Nama Kasus Uji	Konfirmasi pedanan pedagang
Deskripsi	Melakukan konfirmasi pesanan oleh pedagang
Jumlah User	1
Prosedur Uji	1. Melakukan record mengkonfirmasi pesanan pelanggan. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.9 menjelaskan kasus uji untuk membuka halaman kelola barang dagangan. Untuk membuka halaman ini dapat memilih menu kelola barang.

Tabel 6. 9 Kelola Barang Dagangan

Nomor Kasus Uji	PT_009
Nama Kasus Uji	Kelola Barang Dagangan
Deskripsi	Membuka halaman kelola barang
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk membuka halaman kelola barang. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.10 menunjukkan kasus uji untuk melakukan tambah barang dagangan. Menu ini dapat dibuka dengan memilih *button link* tambah barang pada halaman kelola barang.

Tabel 6. 10 Tambah Barang Dagangan.

Nomor Kasus Uji	PT_010
Nama Kasus Uji	Tambah Barang Dagangan
Deskripsi	Melakukan Tambah Barang Dagangan
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk melakukan tambah barang dagangan.

	2. Menjalankan <i>test record</i> . 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i> .
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.11 menunjukkan kasus uji untuk melihat detail barang dan melakukan edit data barang dagangan.

Tabel 6. 11 Detail Barang Pedagang

Nomor Kasus Uji	PT_011
Nama Kasus Uji	Detail Barang Pedagang
Deskripsi	Membuka halaman detail barang
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk membuka halaman detail 2. Menjalankan <i>test record</i> . 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i> .
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.12 menunjukkan kasus uji untuk melakukan Edit Data Barang.

Tabel 6. 12 Edit Data Barang.

Nomor Kasus Uji	PT_012
Nama Kasus Uji	Edit Data Barang
Deskripsi	Membuka halaman detail barang dan memilih edit data.
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk membuka halaman detail dan melakukan edit data. 2. Menjalankan <i>test record</i> . 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i> .
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.13 menunjukkan kasus uji untuk membuka halaman profil pedagang.

Tabel 6. 13 Profil Pedagang

Nomor Kasus Uji	PT_013
Nama Kasus Uji	Profil Pedagang
Deskripsi	Membuka halaman profil pedagang
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk membuka halaman profil 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.14 menunjukkan asus uji untuk melakukan Edit Data Pedagang.

Tabel 6. 14 Edit Profil.

Nomor Kasus Uji	PT_014
Nama Kasus Uji	Edit profil
Deskripsi	Membuka halaman profil pedagang dan melakukan edit data pedagang.
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk membuka halaman profil dan melakukan edit data. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.15 menjelaskan proses pemesanan barang dengan cara memilih pedagang, memilih barang lalu melanjutkan ke keranjang belanja

Tabel 6. 15 Pemesanan Barang / Order

Nomor Kasus Uji	PT_015
Nama Kasus Uji	Pemesanan Barang / Order
Deskripsi	Proses memesan barang
Jumlah User	1

Prosedur Uji	2. Melakukan record untuk memesan barang dengan cara memilih pedagang, dilanjutkan memilih barang dan membuka keranjang barang untuk proses pemesanan. 3. Menjalankan <i>test record</i>. 4. Melakukan performance schedule. 5. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.16 menjelaskan proses menambah barang pada halaman admin. Pada kasus uji ini akan membuka halaman kelola pedagang dan tambah pedagang.

Tabel 6. 16 Tambah Pedagang

Nomor Kasus Uji	PT_016
Nama Kasus Uji	Tambah Pedagang
Deskripsi	Proses menambah pedagang
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk menambah pedagang. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

Tabel 6.17 menjelaskan proses menghapus pedagang dari database. Pada kasus uji akan membuka halaman kelola pedagang.

Tabel 6. 17 Hapus Pedagang

Nomor Kasus Uji	PT_017
Nama Kasus Uji	Hapus Pedagang
Deskripsi	Proses menghapus pedagang
Jumlah User	1
Prosedur Uji	1. Melakukan record untuk menghapus pedagang. 2. Menjalankan <i>test record</i>. 3. Melakukan performance schedule. 4. Menganalisa hasil <i>test</i>.
Hasil yang diharapkan	Aplikasi mampu menampilkan respon waktu tiap <i>test</i> yang dilakukan.

6.3 Hasil Pengujian Fungsional

Pada bagian ini akan dijabarkan hasil dari pengujian *fungsional* yang sudah dilakukan. Hasil pengujian akan dibagi berdasarkan subbab dan ditampilkan dalam tabel.

6.3.1 Hasil Pengujian White-Box

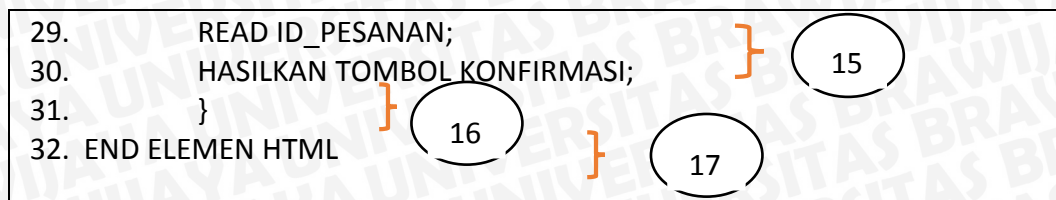
Pada pengujian *White-Box* akan dijabarkan *code program* yang diuji beserta *flowgraph* dan dihitung *cyclomatic complexity* serta menentukan *independent path*.

6.3.1.1 Menampilkan Detail Pesanan

Pada tabel 6.18 menunjukkan *method* detail pesanan pelanggan yang akan diuji menggunakan metode *White-Box* dengan menggunakan *pseudocode*. Untuk memudahkan pembacaan kode program dan *flowgraph* maka akan dibagi berdasarkan nomor yang ditunjukkan pada tabel 6.18.

Tabel 6. 18 Menampilkan Detail Pesanan pelanggan

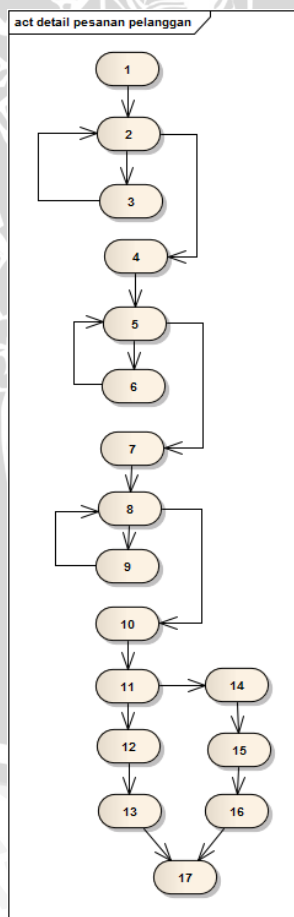
1.	ELEMEN HTML	1
2.	INT A = 1;	
3.	FOREACH (DETAIL AS ROW1){	2
4.	PRINT A;	
5.	PRINT ROW1 --> NAMA_SAYUR;	
6.	PRINT ROW1 --> JUMLAH;	
7.	PRINT ROW1 --> NAMA_STATUS;	3
8.	READ ID_USER;	
9.	READ LAST_UPDATE;	
10.	A++;	4
11.	}	
12.	FOREACH (DETAILKHUSUS AS ROW1){	5
13.	PRINT A;	
14.	PRINT ROW1 -->NAMA_BARANG;	
15.	PRINT ROW1 -->JUMLAH;	6
16.	PRINT ROW1 -->NAMA_STATUS;	
17.	A++;	7
18.	}	
19.	FOREACH (CEKSTATUS AS ROW2){	8
20.	STRING ID_STATUS = ROW2 -->ID_STATUS;	
21.	}	9
22.	IF (STRING ID_STATUS == 1) THEN {	10
23.	READ ID_PESANAN;	
24.	HASILKAN TOMBOL BATAL;	11
25.	}	12
26.	ELSEIF (STRING ID_STATUS == 2)THEN {	13
27.		
28.		14



Pada gambar 6.1 digambarkan *flowgraph* dari aktivitas menampilkan detail pesanan pelanggan yang didapatkan dari kode program yang telah dijabarkan. *Flowgraph* terdiri dari 17 node (N), 20 edge (E) dan 4 percabangan (P). Selanjutnya, dari *flowgraph* tersebut akan ditentukan *independent path* dan nilai *cyclomatic clompexity*.

Dari *flowgraph* yang ada pada Gambar 6.1 didapatkan *independent path* sebanyak 5 path, diantaranya:

1. Path 1 : 1-2-4-5-7-8-10-11-12-13-17
2. Path 2 : 1-2-4-5-7-8-10-11-14-15-16-17
3. Path 3 : 1-2-3-2-4-5-7-8-10-11-12-13-17
4. Path 4 : 1-2-4-5-6-5-7-8-10-11-12-13-17
5. Path 5 : 1-2-4-5-7-8-9-8-10-11-12-13-17



Gambar 6. 1 *Flowgraph* Detail Pesanan Pelanggan

Dari *flowgraph* yang sudah dijelaskan pada gambar 6.1 juga dapat dihitung nilai *cyclomatic complexity* sebagai berikut:

$$\begin{aligned}
 V(G) &= 5 \text{ Regions} \\
 V(G) &= E - N + 2 \\
 &= 20 - 17 + 2 \\
 &= 3 + 2 \\
 &= 5 \\
 V(G) &= P + 1 \\
 &= 4 + 1 \\
 &= 5
 \end{aligned}$$

Berdasarkan 5 jalur independen yang telah didefinisikan tersebut maka dapat dibentuk kasus uji. Tabel 6.19 menjelaskan kasus uji dari *method* menampilkan detail pesanan pelanggan.

Tabel 6. 19 Kasus Uji menampilkan detail pesanan pelanggan

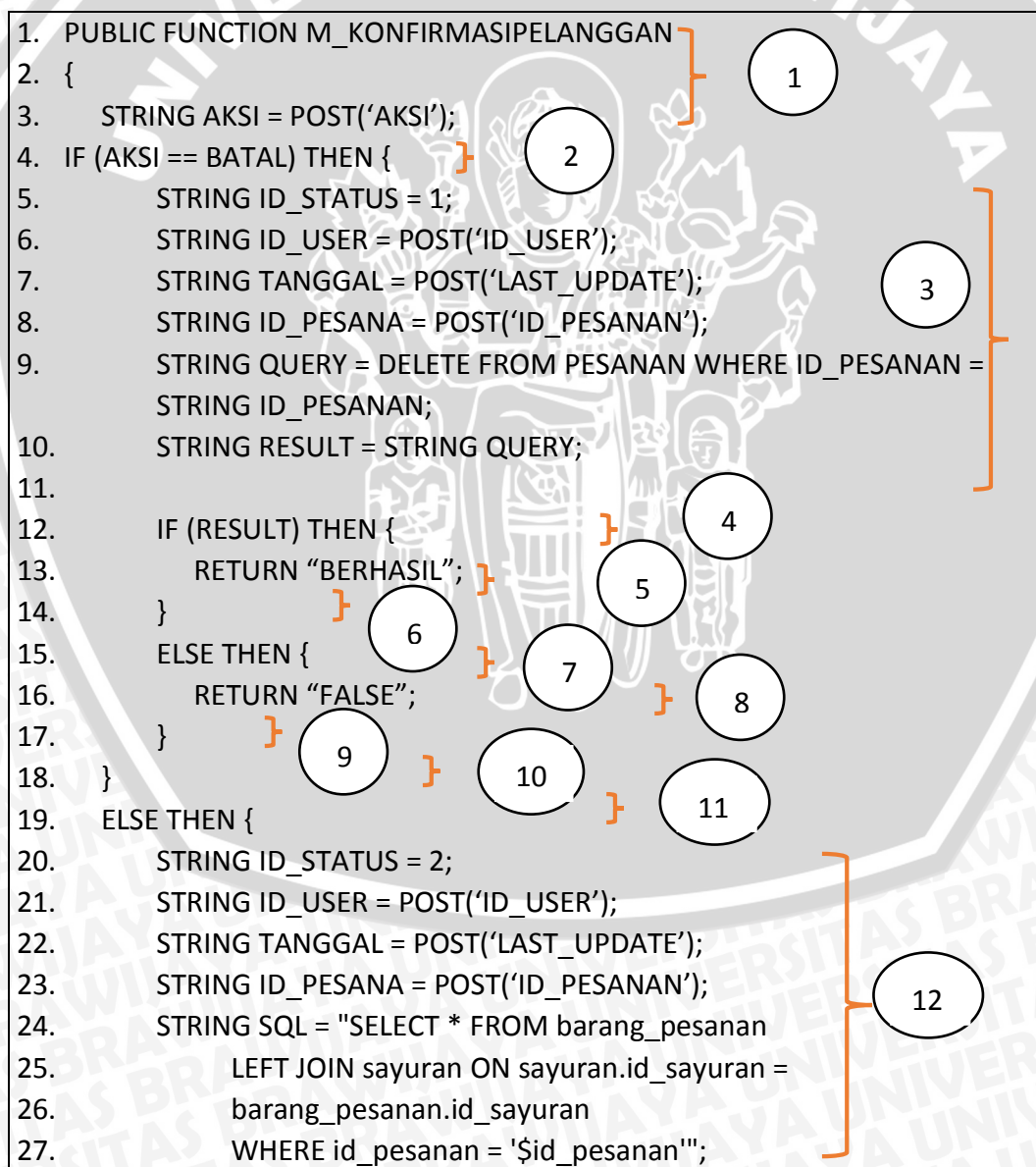
No	User	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	Pelanggan	If id_status == 1	Kode program mampu menampilkan button batal pesanan	Kode program berhasil menampilkan button batal pesanan	Valid
2	Pelanggan	Elseif id_status == 2	Kode program mampu menampilkan button konfirmasi	Kode program berhasil menampilkan button konfirmasi	Valid
3	Pelanggan	Foreach(detail as row1)	Kode program mampu menampilkan data detail pesanan	Kode program berhasil menampilkan data pesanan	Valid
4	Pelanggan	Foreach(detailKhusus as row1)	Kode program mampu menampilkan detail pesanan khusus	Kode program berhasil menampilkan data pesanan khusus	Valid

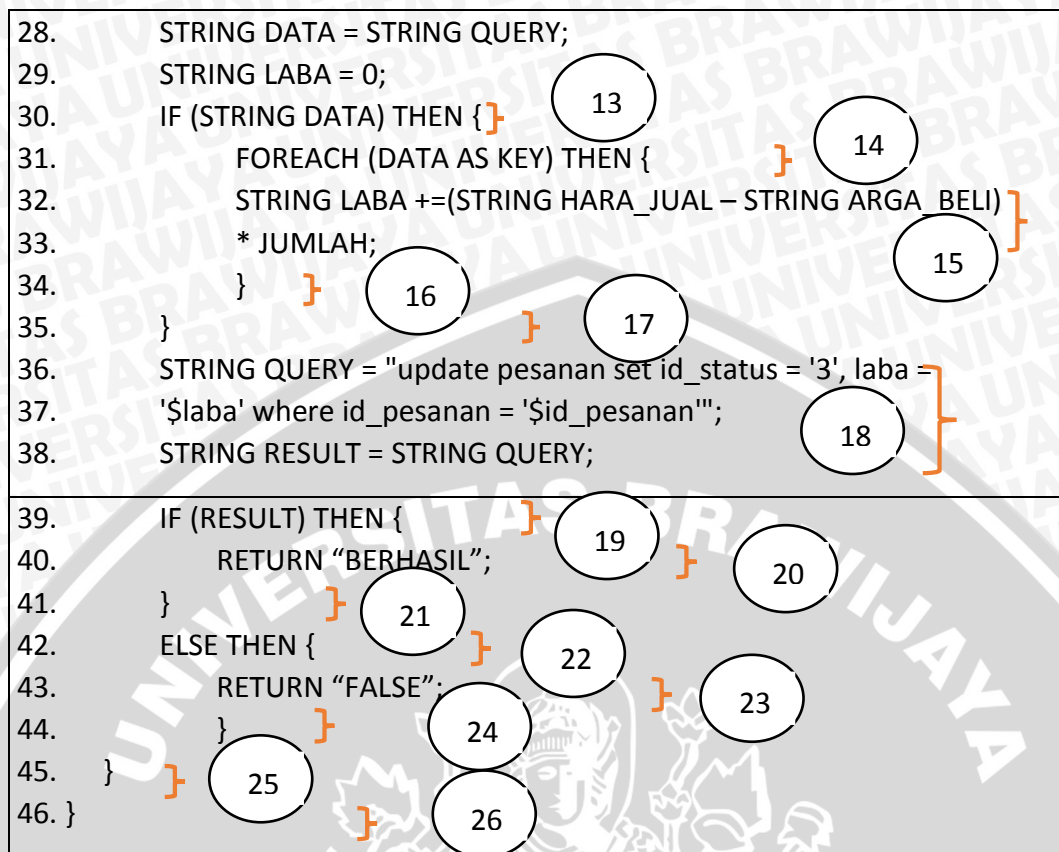
5	Pelanggan	Foreach(cekstatus as row2)	Kode program mampu menampilkan status pesanan	Kode program berhasil menampilkan status pesanan	Valid
---	-----------	-----------------------------------	--	---	-------

6.3.1.2 Konfirmasi Pesanan Pelanggan

Pada tabel 6.4 menunjukkan *method* konfirmasi pesanan pelanggan. Konfirmasi pelanggan digunakan untuk mengkonfirmasi pesanan yang sudah dikirimkan oleh pedagang. Ada dua konfirmasi pada pelanggan yaitu batal pesanan dan konfirmasi barang. Konfirmasi pelanggan akan diuji menggunakan metode *White-Box* dengan menggunakan *pseudocode*. Untuk memudahkan pembacaan kode program dan *flowgraph* maka akan dibagi berdasarkan nomor yang ditunjukkan pada tabel 6.20.

Tabel 6. 20 Konfirmasi Pesanan Pelanggan.





Pada gambar 6.2 ditunjukkan *flowgraph* dari aktivitas menampilkan detail pesanan pelanggan yang didapatkan dari kode program yang telah dijabarkan. *Flowgraph* terdiri dari 26 node (N), 29 edge (E) dan 4 percabangan (P). Selanjutnya, dari *flowgraph* tersebut akan ditentukan *independent path* dan nilai *cyclomatic clomplexity*.

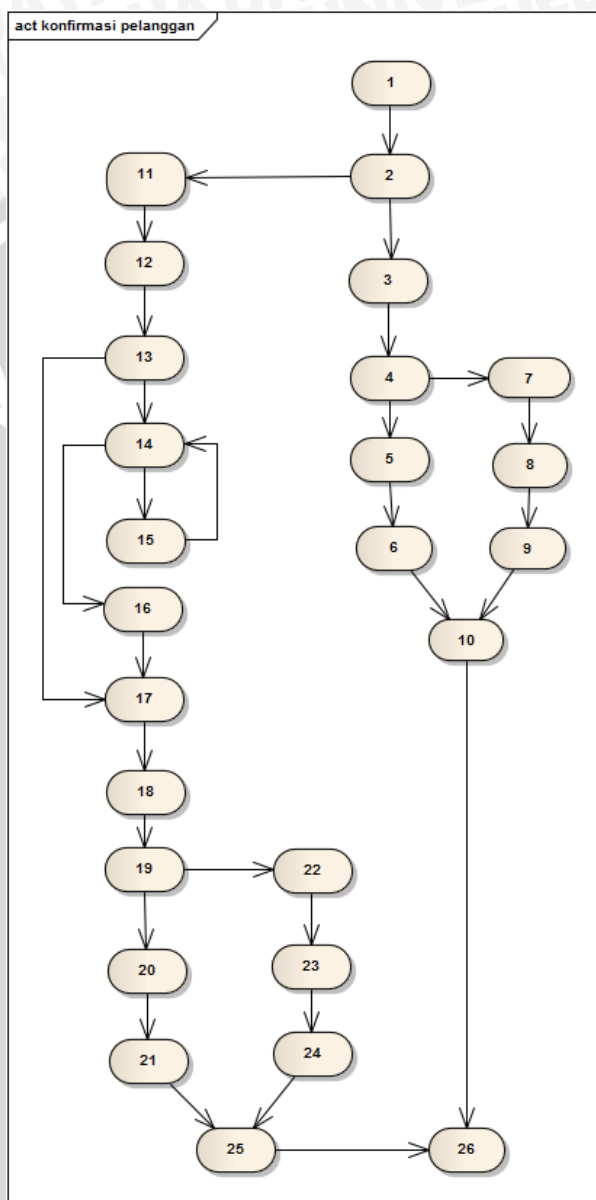
Dari *flowgraph* yang ada pada Gambar 6.2 didapatkan *independent path* sebanyak 5 path, diantaranya:

1. Path 1 : 1 – 2 – 3 – 4 – 5 – 6 – 10 – 26
2. Path 2 : 1 – 2 – 3 – 7 – 8 – 9 – 10 – 26
3. Path 3 : 1 – 2 – 11 – 12 – 13 – 17 – 18 – 19 – 20 – 21 – 25 – 26
4. Path 4 : 1 – 2 – 11 – 12 – 13 – 14 – 15 – 14 – 16 – 17 – 18 – 19 – 20 – 21 – 25 – 26
5. Path 5 : 1 – 2 – 11 – 12 – 13 – 14 – 15 – 14 – 16 – 17 – 18 – 22 – 23 – 24 – 25 – 26

Dari *flowgraph* pada Gambar 6.2 juga dapat dihitung nilai *cyclomatic clomplexity* sebagai berikut:

$$\begin{aligned}
 V(G) &= 4 \text{ regions} \\
 V(G) &= E - N + 2 \\
 &= 29 - 26 + 2 \\
 &= 3 + 2 \\
 &= 5
 \end{aligned}$$

$$\begin{aligned} V(G) &= P + 1 \\ &= 4 + 1 \\ &= 5 \end{aligned}$$



Gambar 6. 2 Flowgraph Konfirmasi Pesanan Pelanggan

Berdasarkan 5 jalur independen yang telah didefinisikan tersebut maka dapat dibentuk kasus uji. Tabel 6.21 menjelaskan kasus uji dari *method* konfirmasi pesanan pelanggan.

Tabel 6. 21 Kasus Uji konfirmasi pesanan pelanggan

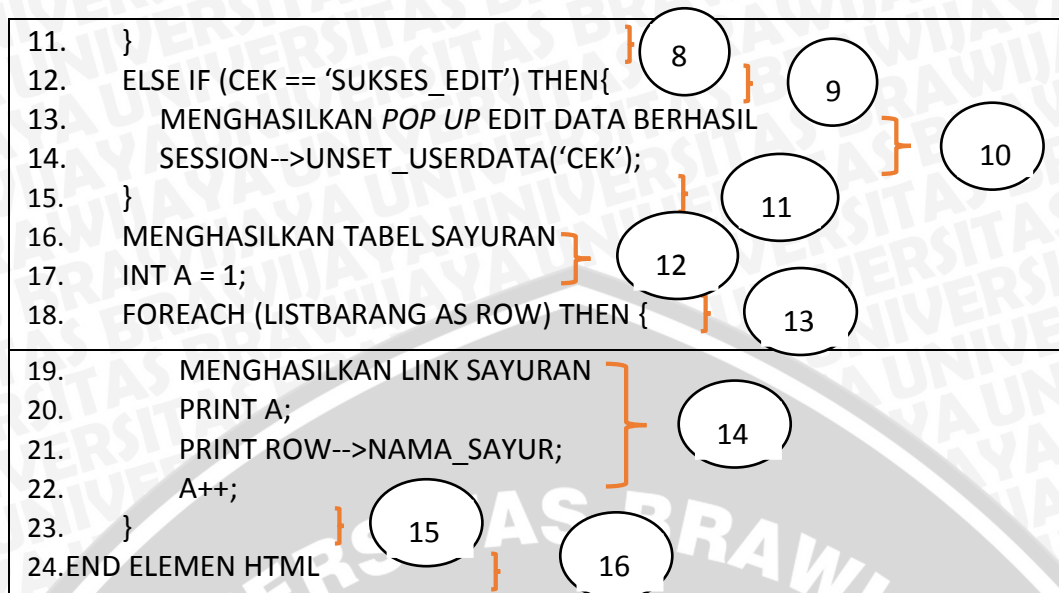
No	User	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	Pelanggan	If (result){ Return "berhasil"; }	Kode program mampu menghapus data pesanan	Kode program berhasil menghapus data pesanan	Valid
2	Pelanggan	Else(result){ return "false" }	Kode program tidak menghapus data pesanan	Data pesanan tidak terhapus	Valid
3	Pelanggan	If (result){ return "berhasil"} }	Kode program dapat mengkonfirmasi pesanan	Pesanan dapat terkonfirmasi	Valid
4	Pelanggan	Foreach (data as key)	Kode program dapat menghasilkan laba dan masuk ke database	Kode program berhasil menghasilkan laba dan masuk ke database	Valid
5	Pelanggan	Else (result) return "false" }	Kode program tidak dapat mengkonfirmasi pesanan	Pesanan tidak terkonfirmasi	Valid

6.3.1.3 Menampilkan List Barang Pedagang

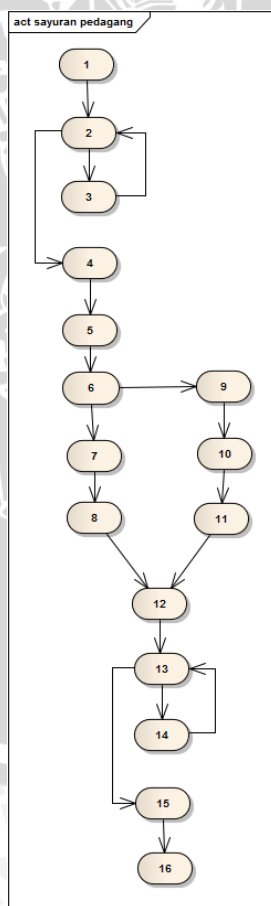
Pada tabel 6.6 menunjukkan *method* menampilkan *list* baranng pedagang yang akan diuji menggunakan metode *White-Box* dengan menggunakan *pseudocode*. Untuk memudahkan pembacaan kode program dan *flowgraph* maka akan dibagi berdasarkan nomor yang ditunjukkan pada tabel 6.22

Tabel 6. 22 Menampilkan List Barang Pedagang

1. ELEMEN HTML	
2. MENGHASILKAN TOMBOL SEMUA BARANG;	
3. FOREACH (PILIHKATEGORI AS ROW) THEN {	
4. MENGHASILKAN TOMBOL KATEGORI	
5. }	
6. STRING CEK = SESSION('CEK');	
7. IF (CEK == 'SUKSES_TAMBAH') THEN {	
8. MENGHASILKAN POP UP BARANG DITAMBAHKAN	
9. SESSION-->UNSET_USERDATA('CEK');	
10.	



Pada Gambar 6.3 digambarkan *flowgraph* menampilkan *list* barang pedagang yang didapatkan dari kode program yang telah dijabarkan. *Flowgraph* terdiri dari 16 node (N), 18 edge (E), dan 4 percabangan (P). Selanjutnya, dari *flowgraph* tersebut akan ditentukan *independent path* dan nilai *cyclomatic clompexity*.



Gambar 6. 3 Flowgraph Menampilkan List Barang Pedagang

Dari *flowgraph* yang ada pada Gambar 6.3 didapatkan *independent path* sebanyak 4 path, diantaranya :

1. Path 1 : 1 – 2 – 4 – 5 – 6 – 7 – 8 – 12 – 13 – 15 – 16
2. Path 2 : 1 – 2 – 4 – 5 – 6 – 9 – 10 – 11 – 12 – 13 – 15 – 16
3. Path 3 : 1 – 2 – 3 – 2 – 4 – 5 – 6 – 7 – 8 – 12 – 13 – 15 – 16
4. Path 4 : 1 – 2 – 4 – 5 – 6 – 7 – 8 – 12 – 13 – 14 – 13 – 15 – 16

Setelah mendapatkan *independent path*, selanjutnya akan dihitung nilai *cyclomatic complexity* sebagai berikut:

$$\begin{aligned}
 V(G) &= E - N + 2 \\
 &= 18 - 16 + 2 \\
 &= 4 \\
 V(G) &= P + 1 \\
 &= 3 + 1 \\
 &= 4
 \end{aligned}$$

Berdasarkan 4 jalur independen yang telah didefinisikan tersebut maka dapat dibentuk kasus uji. Tabel 6.23 menjelaskan kasus uji dari *method* menampilkan barang pedagang.

Tabel 6. 23 Kasus Uji Menampilkan *List* Barang Pedagang

No	User	Data Input	Hasil yang diharapkan	Hasil yang diperoleh	Status
1	Pedagang	If (Cek == 'sukses_tambah')	Kode program dapat menampilkan <i>pop up</i> Barang ditambahkan	Pop up tambah barang berhasil tampil	Valid
2	Pedagang	If (cek == 'sukses_edit')	Kode program dapat menampilkan <i>pop up</i> edit data berhasil	Pop up edit data berhasil ditampilkan	valid
3	Pedagang	Foreach (pilihKategori)	Kode program dapat menampilkan menu kategori	Menu Kategori berhasil ditampilkan	valid
4	Pedagang	Foreach (listBarang as row)	Kode program dapat menampilkan list sayur	List sayur berhasil ditampilkan	Valid

6.3.2 Hasil Pengujian *Black-Box*

Pada bagian ini akan dijabarkan hasil pengujian *Black-Box* dan mekanisme kasus uji yang diujikan untuk pengujian. Kasus uji dan hasil pengujian akan ditampilkan dalam bentuk tabel.

6.3.2.1 Mekanisme Pengujian

Pengujian *Black-Box* dilakukan dengan uji validasi pada semua kebutuhan *functional* yang ada pada tabel 6.8. Setiap spesifikasi kebutuhan perangkat lunak yang didefinisikan pada tahap analisis kebutuhan akan diujikan pada pengujian ini dengan cara mendefinisikan kasus uji terhadap setiap kebutuhan tersebut, lalu membandingkannya dengan hasil yang diperoleh. Berikut Kasus Uji pada sistem pemesanan sayur.

Tabel 6.24 kasus uji dari pengujian validasi kelola pedagang pada halaman admin untuk menampilkan daftar pedagang.

Tabel 6. 24 Kasus Uji Kelola Pedagang

Nomor Kasus uji	VAL_001
Nama Kasus uji	Kelola Pedagang
Nomor <i>Use case</i>	SRS_01_001
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan daftar pedagang.
Prosedur Uji	1. Penguji melakukan login dan masuk kedalam halaman kelola pedagang
Hasil yang Diharapkan	Sistem dapat menampilkan daftar pedagang

Tabel 6.25 menjelaskan kasus uji pada pengujian validasi tambah pedagang dari fungsi *c_TambahPedagang* untuk menambahkan pedagang.

Tabel 6. 25 Kasus Uji Validasi Tambah Pedagang

Nomor Kasus uji	VAL_002
Nama Kasus uji	Tambah Pedagang
Nomor <i>Use case</i>	SRS_01_002
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan penambahan data pedagang.
Prosedur Uji	1. Penguji mengisi <i>form</i> tambah pedagang 2. Penguji menekan tombol submit
Hasil yang Diharapkan	Sistem menampilkan <i>form</i> tambah data dan dapat menambah data pedagang

Tabel 6.26 menjelaskan kasus uji dari pengujian validasi hapus pedagang dari fungsi *c_HapusPedagang* untuk menghapus pedagang.

Tabel 6. 26 Kasus Uji Validasi Hapus Pedagang

Nomor Kasus uji	VAL_003
Nama Kasus uji	Hapus Pedagang
Nomor <i>Use case</i>	SRS_01_003
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menghapus data pedagang dari <i>database</i> .
Prosedur Uji	1. Penguji memilih pedagang yang akan dihapus 2. Penguji menekan <i>icon</i> atau tombol hapus
Hasil yang Diharapkan	Sistem dapat menghapus dan menampilkan <i>pop up</i> hapus data berhasil.

Tabel 6.27 menjelaskan kasus uji dari pengujian validasi kelola pesanan pedagang pada fungsi berandaPedagang untuk menampilkan pesanan pelanggan.

Tabel 6. 27 Kasus Uji Validasi Kelola Pesanan

Nomor Kasus uji	VAL_004
Nama Kasus uji	Kelola Pesanan Pedagang
Nomor <i>Use case</i>	SRS_01_004
Tujuan Pengujian	Pegujian dilakukan untuk memastikan bahwa sistem dapat menampilkan data pesanan pelanggan yang sudah memesan
Prosedur Uji	Penguji melakukan <i>login</i> atau masuk sistem
Hasil yang Diharapkan	Sistem dapat menampilkan data pesanan pelanggan yang sudah memesan.

Tabel 6.28 menjelaskan kasus uji dari pengujian validasi detail pesanan pedagang dari fungsi C_detailPesanan untuk menampilkan detail pesanan pelanggan.

Tabel 6. 28 Kasus Uji Detail Pesanan Pedagang

Nomor Kasus uji	VAL_005
Nama Kasus uji	Detail Pesanan Pedagang
Nomor <i>Use case</i>	SRS_01_005
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan detail pesanan pelangga yang dipilih.
Prosedur Uji	1. Penguji memilih salah 1 pesanan pelanggan pada halaman kelola pedagang 2. Sistem menampilkan halaman detail pesanan

Hasil yang Diharapkan	Sistem dapat menampilkan data pesanan pelanggan yang dipilih
-----------------------	--

Tabel 6.29 menjelaskan kasus uji dari pengujian validasi Konfirmasi Pesanan Pedagang pada fungsi konfirmasiPedagang untuk mengonfirmasi pesanan pelanggan.

Tabel 6. 29 Kasus Uji Konfirmasi Pesanan Pedagang

Nomor Kasus uji	VAL_006
Nama Kasus uji	Konfirmasi Pesanan Pedagang
Nomor <i>Use case</i>	SRS_01_006
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan konfirmasi pesanan.
Prosedur Uji	1. Penguji menekan tombol konfirmasi 2. Sistem menampilkan <i>popup</i>
Hasil yang Diharapkan	Sistem menampilkan <i>popup</i> konfirmasi berhasil dan mengirim notifikasi kepada pelanggan

Tabel 6.30 menjelaskan kasus uji dari pengujian validasi kelola barang pada fungsi C_kelolaBarang untuk menampilkan *list* barang pedagang. Selain menampilkan barang pada kelola barang juga dapat memilih detail barang dan dapat melakukan mengubah data barang.

Tabel 6. 30 Kasus Uji Kelola Barang

Nomor Kasus uji	VAL_007
Nama Kasus uji	Kelola Barang
Nomor <i>Use case</i>	SRS_01_007
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan <i>list</i> barang pedagang
Prosedur Uji	1. Penguji memilih menu kelola barang 2. Sistem menampilkan <i>list</i> barang 3. Sistem menampilkan detail barang 4. Sistem dapat mengubah data barang
Hasil yang Diharapkan	Sistem dapat menampilkan <i>list</i> barang milik pedagang, dapat menampilkan detail barang dan mengubah data barang.

Tabel 6.31 menjelaskan kasus uji dari pengujian validasi tambah barang pada fungsi prosesTambahBarang untuk menambahkan barang.

Tabel 6. 31 Kasus Uji Tambah Barang

Nomor Kasus uji	VAL_008
Nama Kasus uji	Tambah Barang
Nomor <i>Use case</i>	SRS_01_008
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan halaman tambah barang dan dapat menambah barang.
Prosedur Uji	1. Penguji memilih menu tambah barang 2. Penguji mengisi <i>form</i> tambah barang 3. Penguji menekan tombol submit
Hasil yang Diharapkan	Sistem dapat menambah barang dan menampilkan <i>popup</i> bahwa tambah data berhasil

Tabel 6.32 menjelaskan kasus uji dari pengujian validasi detail barang dagangan untuk menampilkan detail barang dagangan yang dipilih.

Tabel 6. 32 Detail Barang Dagangan

Nomor Kasus uji	VAL_009
Nama Kasus uji	Detail Barang Dagangan
Nomor <i>Use case</i>	SRS_01_008
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan detail barang
Prosedur Uji	1. Penguji memilih menu tambah barang 2. Penguji memilih barang
Hasil yang Diharapkan	Sistem dapat menampilkan detail barang

Tabel 6.33 menjelaskan kasus uji edit dbarang dagangan untuk merubah data barang dagangan.

Tabel 6. 33 Kasus Uji Edit Data Barang

Nomor Kasus uji	VAL_010
Nama Kasus uji	Edit Barang Dagangan
Nomor <i>Use case</i>	SRS_01_010
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat merubah data barang
Prosedur Uji	1. Penguji memilih menu tambah barang 2. Penguji memilih barang 3. Pennguji memilih fitur edit dan mengisi data
Hasil yang Diharapkan	Sistem dapat merubah data barang

Tabel 6.34 menjelaskan kasus uji dari pengujian profil dari fitur profil untuk menampilkan data pedagang.

Tabel 6. 34 Kasus Uji Profil

Nomor Kasus uji	VAL_011
Nama Kasus uji	Profil
Nomor <i>Use case</i>	SRS_01_011
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan data pedagang
Prosedur Uji	Penguji memilih menu profil
Hasil yang Diharapkan	Sistem dapat menampilkan data pedagang

Tabel 6.35 menjelaskan kasus uji dari pengujian validasi edit profil untuk dapat merubah data pedagang.

Tabel 6. 35 Kasus Uji Edit Profil

Nomor Kasus uji	VAL_012
Nama Kasus uji	Edit Profil
Nomor <i>Use case</i>	SRS_01_012
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat merubah data pedagang
Prosedur Uji	1. Penguji memilih edit data 2. Penguji mengisi form
Hasil yang Diharapkan	Sistem dapat merubah data pedagang

Tabel 6.36 menjelaskan kasus uji dari pengujian validasi kelola pesanan pelanggan pada fungsi berandaPelanggan untuk menampilkan *list* pesanan pembeli pada pedagang.

Tabel 6. 36 Kasus Uji Kelola Pesanan Pelanggan

Nomor Kasus uji	VAL_013
Nama Kasus uji	Kelola Pesanan Pelanggan
Nomor <i>Use case</i>	SRS_01_013
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan <i>list</i> pesanan pada pedagang.
Prosedur Uji	1. Penguji melakukan <i>login</i> ke dalam sistem 2. Sistem menampilkan <i>list</i> pesanan
Hasil yang Diharapkan	Sistem dapat menampilkan <i>list</i> pesanan pada pedagang yang pernah dipesan barangnya.

Tabel 6.35 menjelaskan kasus uji dari pengujian validasi detail pesanan pelanggan pada fungsi C_DetailPesanan untuk menampilkan detail pesanan pelanggan.

Tabel 6. 37 Kasus Uji Detail Pesanan Pelanggan

Nomor Kasus uji	VAL_014
Nama Kasus uji	Detail pesanan Pelanggan
Nomor <i>Use case</i>	SRS_01_014
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan detail pesanan pelanggan
Prosedur Uji	1. Penguji memilih pesanan yang diinginkan
Hasil yang Diharapkan	Sistem menampilkan data pesanan pelanggan

Tabel 6.38 menjelaskan kasus uji dari pengujian validasi konfirmasi pesanan pelanggan pada fungsi konfirmasiPelanggan.

Tabel 6. 38 Kasus Uji Konfirmasi Pesanan Pelanggan

Nomor Kasus uji	VAL_015
Nama Kasus uji	Konfirmasi Pesanan Pelanggan
Nomor <i>Use case</i>	SRS_01_015
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat mengonfirmasi pesanan
Prosedur Uji	Penguji menekan tombol konfirmasi
Hasil yang Diharapkan	Sistem dapat mengonfirmasi dan menampilkan <i>popup</i> berhasil mengonfirmasi

Tabel 6.34 menjelaskan kasus uji dari pengujian validasi pilih pedagang pada fungsi C_listPedagang untuk menampilkan *list* pedagang dan memilih pedagang.

Tabel 6. 39 Kasus Uji Pilih Pedagang

Nomor Kasus uji	VAL_016
Nama Kasus uji	Pilih Pedagang
Nomor <i>Use case</i>	SRS_01_016
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan <i>list</i> pedagang
Prosedur Uji	1. Penguji memilih menu pilih pedagang 2. Penguji memilih pedagang
Hasil yang Diharapkan	Sistem dapat menampilkan <i>list</i> pedagang

Tabel 6.40 menjelaskan penujian validasi kasus uji pilih barang untuk memilih barang yang akan dipesan.

Tabel 6. 40 Kasus Uji Pilih Barang

Nomor Kasus uji	VAL_017
Nama Kasus uji	Pilih Barang
Nomor <i>Use case</i>	SRS_01_017
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan data barang dagangan
Prosedur Uji	1. Penguji memilih salah satu pedagang
Hasil yang Diharapkan	Sistem dapat menampilkan data barang

Tabel 6.41 menjelaskan kasus uji dari pengujian validasi pesanan khusus untuk memesan barang secara khusus.

Tabel 6. 41 Kasus Uji Pesanan Khusus

Nomor Kasus uji	VAL_018
Nama Kasus uji	Pesanan Khusus
Nomor <i>Use case</i>	SRS_01_018
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat menampilkan form pesanan khusus dan dapat mengirim pesanan
Prosedur Uji	1. Penguji memilih menu pesanan khusus 2. Penguji mengisi form dan menekan submit
Hasil yang Diharapkan	Sistem dapat menyimpan pesanan dan menampilkan form

Tabel 6.42 menjelaskan kasus uji dari pengujian validasi keranjang belanja pada fitur keranjang belanja untuk memilih barang dan menampilkan keranjang belanja serta pesanan khusus.

Tabel 6. 42 Kasus Uji Keranjang Belanja

Nomor Kasus uji	VAL_019
Nama Kasus uji	Keranjang Belanja
Nomor <i>Use case</i>	SRS_01_019
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat memilih barang dan menampilkan keranjang belanja
Prosedur Uji	1. Penguji memilih barang pesanan 2. Penguji memilih pesanan khusus

	3. Penguji menekan tombol submit
Hasil yang Diharapkan	Sistem menampilkan menu <i>popup</i> barang dimasukkan ke dalam keranjang dan menampilkan keranjang belanja

Tabel 6.43 menjelaskan kasus uji dari pengujian validasi registrasi pada fitur register untuk melakukan pendaftaran ke dalam sistem.

Tabel 6. 43 Kasus Uji Register

Nomor Kasus uji	VAL_20
Nama Kasus uji	Registrasi
Nomor <i>Use case</i>	SRS_01_020
Tujuan Pengujian	Pengujian dilakukan untuk memastikan sistem dapat melakukan proses penambahan data dan menampilkan <i>form</i> register.
Prosedur Uji	1. Penguji memilih menu <i>register</i> 2. Penguji mengisi <i>form</i> dan menekan tombol submit.
Hasil yang Diharapkan	Sistem mampu menampilkan <i>form register</i> , sistem dapat menambahkan data dan menampilkan <i>popup</i> tambah data berhasil

Tabel 6.44 menjelaskan kasus uji dari pengujian *login* pada fitur *login* untuk masuk ke dalam sistem. Pada tabel 6.21 akan dijelaskan prosedur uji untuk menguji fitur login ke dalam sistem.

Tabel 6. 44 Kasus Uji Login

Nomor Kasus uji	VAL_21
Nama Kasus uji	<i>Login</i>
Nomor <i>Use case</i>	SRS_01_021
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan proses <i>login</i> masuk ke dalam sistem.
Prosedur Uji	Penguji mengisi <i>username</i> dan <i>password</i> pada halaman <i>login</i>
Hasil yang Diharapkan	Sistem mampu melakukan cek data dari <i>database</i> dan <i>login</i> ke dalam sistem

Tabel 6.45 kasus uji dari pengujian validasi *logout* pada fungsi *logout* untuk keluar dari sistem.

Tabel 6. 45 Kasus Uji Logout

Nomor Kasus uji	VAL_22
Nama Kasus uji	<i>Logout</i>

Nomor <i>Use case</i>	SRS_01_022
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa sistem dapat melakukan <i>logout</i> dengan menghapus <i>session</i> .
Prosedur Uji	Penguji menekan tombol <i>logout</i>
Hasil yang Diharapkan	Sistem dapat menghapus <i>session</i> dan keluar dari sistem lalu menampilkan halaman login.

6.3.2.2 Hasil Pengujian

Pada tabel 6.46 menjelaskan hasil pengujian terhadap sistem yang dilakukan dengan metode *Black-Box*.

Tabel 6. 46 Hasil Pengujian *Black-Box*

No Kasus Uji	Nama Kasus Uji	Hasil Yang Diharapkan	Hasil Yang Diperoleh	Status
Val_001	Kelola Pedagang	Sistem menampilkan daftar pedagang	Sistem berhasil menampilkan daftar pedagang	<i>Valid</i>
VAL_002	Kasus uji tambah pedagang	Sistem menampilkan <i>form</i> tambah data dan dapat menambah data pedagang	Sistem berhasil menampilkan <i>form</i> dan dapat menambah pedagang	<i>Valid</i>
VAL_003	Kasus uji hapus pedagang	Sistem dapat menghapus dan menampilkan <i>pop up</i> hapus data berhasil.	Sistem berhasil menghapus data dan menampilkan <i>popup</i>	<i>Valid</i>
VAL_004	Kasus uji kelola pesanan pedagang	Sistem dapat menampilkan data pesanan pelanggan yang sudah memesan.	Sistem berhasil menampilkan data pesanan pelanggan yang sudah memesan.	<i>Valid</i>
VAL_005	Kasus uji detail pesanan pedagang	Sistem dapat menampilkan data pesanan pelanggan yang dipilih	Sistem berhasil menampilkan data pesanan pelanggan yang dipilih	<i>Valid</i>
VAL_006	Kasus uji konfirmasi	Sistem menampilkan	Sistem berhasil menampilkan	<i>Valid</i>

	pesanan pedagang	<i>popup</i> konfirmasi berhasil dan mengirim notifikasi kepada pelanggan	<i>popup</i> konfirmasi berhasil dan mengirim notifikasi kepada pelanggan	
VAL_007	Kasus uji kelola barang	Sistem dapat menampilkan <i>list</i> barang milik pedagang, dapat menampilkan detail barang dan mengubah data barang.	Sistem berhasil menampilkan <i>list</i> barang milik pedagang, dapat menampilkan detail barang dan mengubah data barang.	<i>Valid</i>
VAL_008	Kasus uji tambah barang	Sistem dapat menambah barang dan menampilkan <i>popup</i> bahwa tambah data berhasil	Sistem berhasil menambah barang dan menampilkan <i>popup</i> bahwa tambah data berhasil	<i>Valid</i>
VAL_009	Kasus uji detil barang dagangan	Sistem dapat menampilkan detail barang	Sistem berhasil menampilkan detail barang	<i>Valid</i>
VAL_010	Kasus uji edit data barang	Sistem dapat merubah data barang	Sistem berhasil merubah data barang	<i>Valid</i>
VAL_011	Kasus uji profil	Sistem dapat menampilkan data pedagang	Sistem berhasil menampilkan data pedagang	<i>Valid</i>
Val_012	Kasus uji edit profil	Sistem dapat merubah data pedagang	Sistem berhasil merubah data pedagang	<i>Valid</i>
VAL_013	Kasus uji kelola pesanan pelanggan	Sistem dapat menampilkan <i>list</i> pesanan pada pedagang yang pernah dipesan barangnya.	Sistem berhasil menampilkan <i>list</i> pesanan pada pedagang yang pernah dipesan barangnya.	<i>Valid</i>
VAL_014	Kasus uji detail	Sistem menampilkan	Sistem berhasil menampilkan	<i>Valid</i>

	pesanan pelanggan	data pesanan pelanggan	data pesanan pelanggan	
VAL_015	Kasus uji konfirmasi pesanan pelanggan	Sistem dapat mengonfirmasi dan menampilkan <i>popup</i> berhasil mengonfirmasi	Sistem berhasil mengonfirmasi dan menampilkan <i>popup</i> berhasil mengonfirmasi	<i>Valid</i>
VAL_016	Kasus uji pilih pedagang	Sistem dapat menampilkan <i>list</i> pedagang	Sistem berhasil menampilkan <i>list</i> pedagang	<i>Valid</i>
VAL_017	Kasus uji pilih barang	Sistem dapat menampilkan daftar barang dagangan	Sistem berhasil menampilkan daftar barang dagangan	<i>Valid</i>
VAL_018	Kasus uji pesanan khusus	Sistem dapat menampilkan form dan menyimpan pesanan khusus	Sistem berhasil menyimpan pesanan khusus dan menampilkan form	<i>Valid</i>
VAL_019	Kasus uji keranjang belanja	Sistem menampilkan menu <i>popup</i> barang dimasukkan ke dalam keranjang dan menampilkan keranjang belanja	Sistem berhasil menampilkan menu <i>popup</i> barang dimasukkan ke dalam keranjang dan menampilkan keranjang belanja	<i>Valid</i>
VAL_020	Kasus uji register	Sistem mampu menampilkan <i>form register</i> , sistem dapat menambahkan data dan menampilkan <i>popup</i> tambah data berhasil	Sistem berhasil menampilkan <i>form register</i> , sistem dapat menambahkan data dan menampilkan <i>popup</i> tambah data berhasil	<i>Valid</i>
VAL_021	Kasus uji login	Sistem mampu melakukan cek data dari <i>database</i> dan <i>login</i> ke dalam sistem	Sistem berhasil melakukan cek data dari <i>database</i> dan <i>login</i> ke dalam sistem	<i>Valid</i>

VAL_022	Kasus uji logout	Sistem dapat menghapus <i>session</i> dan keluar dari sistem lalu menampilkan halaman login.	Sistem berhasil menghapus <i>session</i> dan keluar dari sistem lalu menampilkan halaman login.	<i>Valid</i>
---------	------------------	--	---	--------------

6.4 Performance test

Pengujian performa bertujuan untuk mengetahui performa dari aplikasi yang telah dibuat berdasarkan kebutuhan yang ada. Pengujian performa dilakukan dengan aplikasi *Ratioanal Performance tester*. Mekanisme pengujian performa dan hasil uji akan ditunjukkan dalam bentuk tabel.

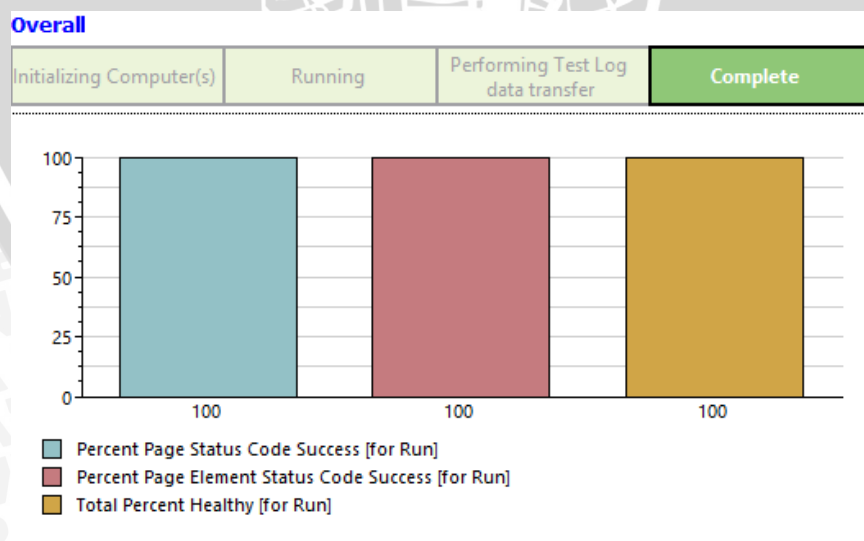
6.4.1 Hasil Pengujian Performance Testing

Pada bagian ini akan dijabarkan hasil pengujian performa berdasarkan mekanisme pengujian yang telah dijabarkan pada sub bab sebelumnya. Pengujian dilakukan dengan menggunakan virtual user.

1. Kasus Uji PT_001

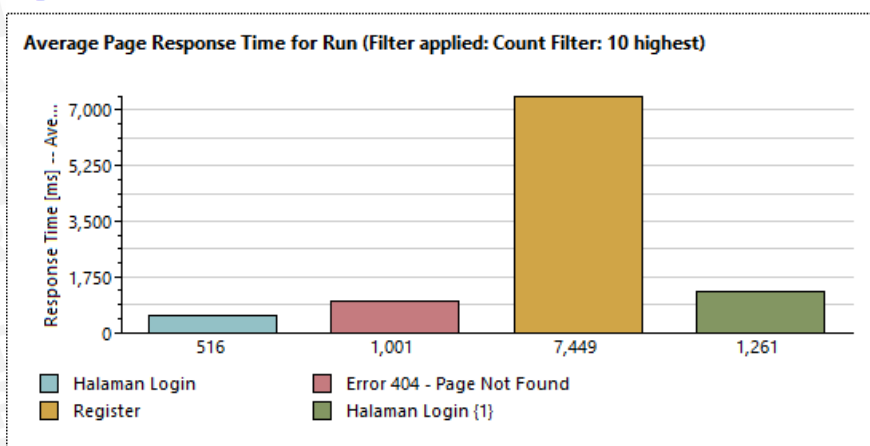
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_001 dari fitur *register* yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.

Gambar 6.4 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_001. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.



Gambar 6. 4 Overall Kasus Uji PT_001

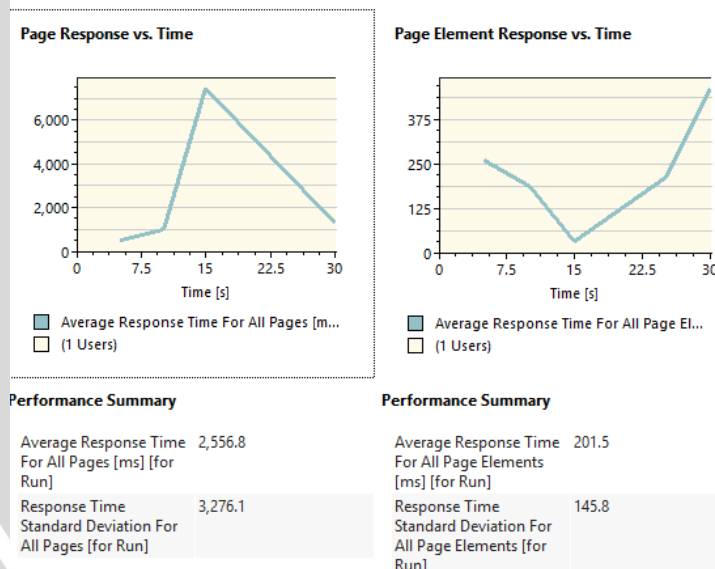
Page Performance



Gambar 6. 5 Page Performance Kasus Uji PT_001

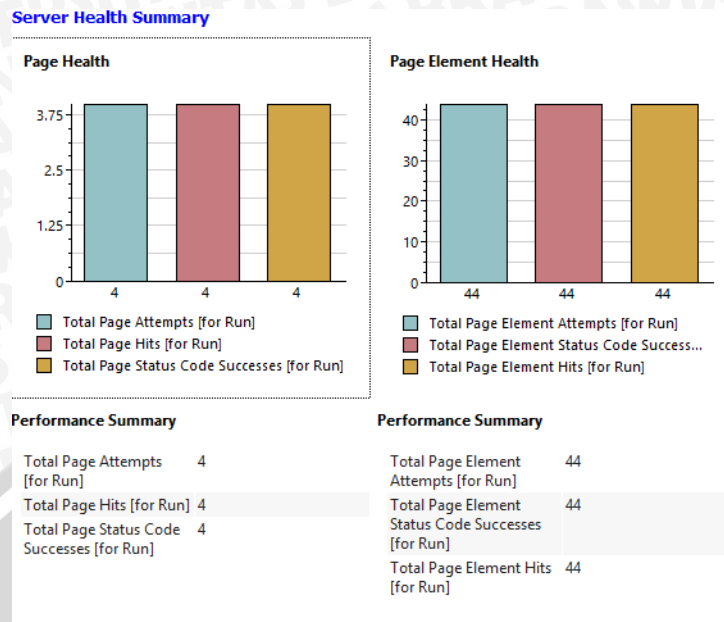
Pada gambar 6.5 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur register. Dari gambar 6.5 dapat diketahui bahwa pada *response* waktu pada saat membuka halaman login 516ms, halaman register 7,499ms dan halaman login 1,261ms. Sedangkan *page not found* selama 1,001ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

Response vs. Time Summary



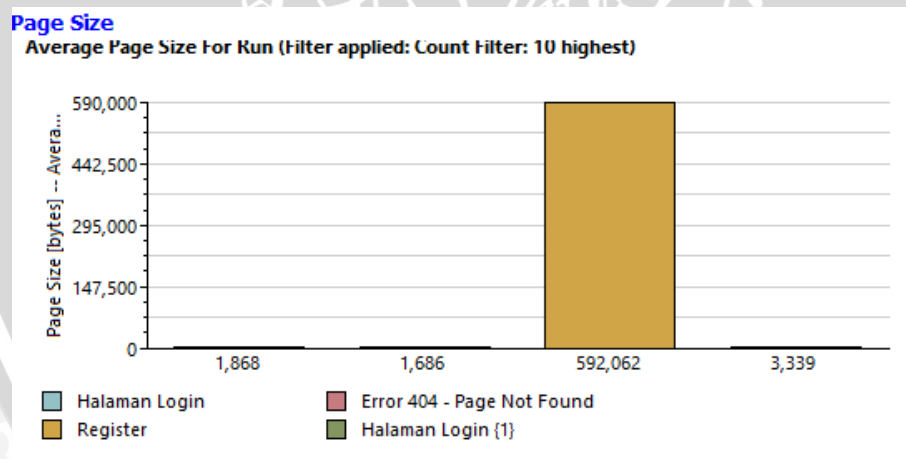
Gambar 6. 6 Response VS Time Summary Kasus Uji PT_001

Pada gambar 6.6 ditunjukkan *average response time for all page* 2,556.8ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 3,276.1ms. Sedangkan *Average response time for all page elements* 201.5ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 145.8ms



Gambar 6. 7 Server Health Summary Kasus Uji PT_001

Gambar 6.7 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.6 bahwa total page yang dijalankan sebanyak 4 halaman dan total elemen yang dijalankan ada 44 elemen.



Gambar 6. 8 Page Size Kasus Uji PT_001

Gambar 6.8 menunjukkan ukuran halaman atau page size dari kasus uji PT_001. Dapat dilihat pada gambar bahwa halaman register memiliki ukuran paling besar yaitu 592, 062bytes. Halaman register memiliki ukuran paling besar karena memiliki elemen yang lebih banyak dibandingkan dengan halaman lain. Halaman login memiliki ukuran 1,863, halaman login setelah sukses register sebesar 3,339.

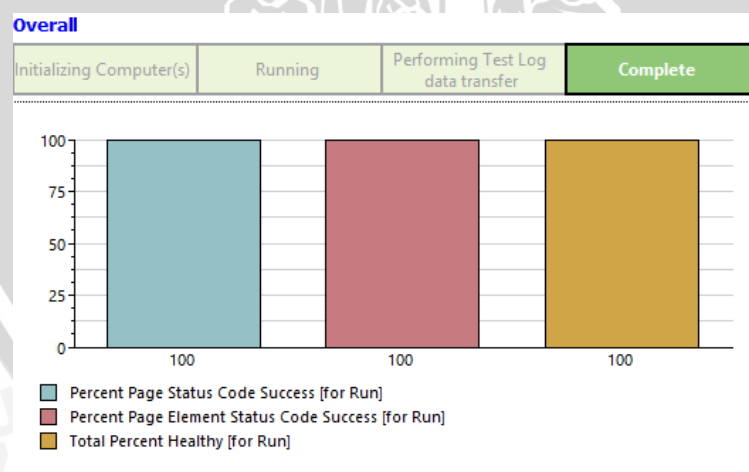
Summary			
Run Summary		Page Summary	
Executed Test	register1	Total Page Attempts [for Run]	4
Active Users	0	Total Page Hits [for Run]	4
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Average Response Time For All Pages [ms] [for Run]	2,556.8
Elapsed Time [H:M:S]	0:00:32	Response Time Standard Deviation For All Pages [for Run]	3,276.1
Run Status	Complete	Maximum Response Time For All Pages [ms] [for Run]	7,449
Displaying Results for Computer:	All Hosts	Minimum Response Time For All Pages [ms] [for Run]	516
Page Element Summary			
Total Page Element Attempts [for Run]	44		
Total Page Element Hits [for Run]	44		
Average Response Time For All Page Elements [ms] [for Run]	201.5		
Response Time Standard Deviation For All Page Elements [for Run]	145.8		

Gambar 6. 9 Summary Kasus Uji PT_001

Gambar 6.9 merupakan rangkuman dari pengujian pada kasus uji PT_001. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halama.

2. Kasus Uji PT_002

Pada bagian ini menjelaskan hasil pengujian kasus uji PT_002 dari fitur kelola pesanan pelanggan yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.

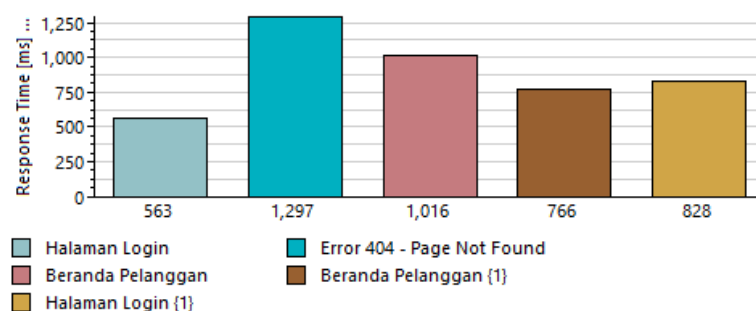


Gambar 6. 10 Overall Kasus Uji PT_002

Gambar 6.10 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_002. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

Page Performance

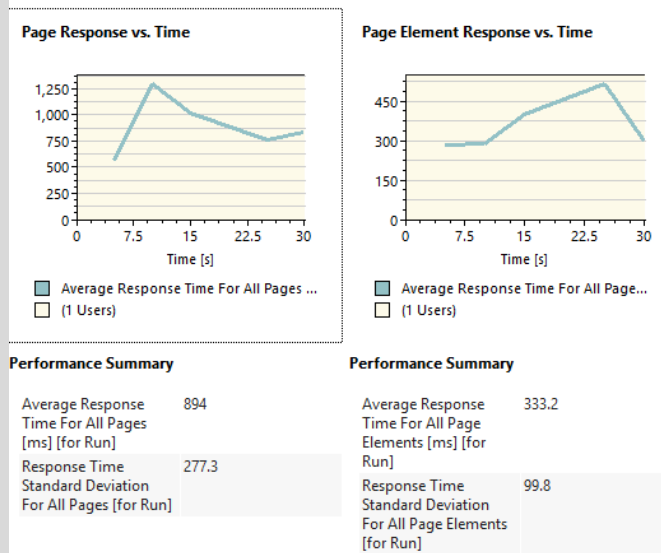
Average Page Response Time for Run (Filter applied: Count Filter: 10 highest)



Gambar 6. 11 Page performance Kasus Uji PT_002

Pada gambar 6.11 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola pesanan. Dari gambar 6.10 dapat diketahui bahwa pada untuk memanggil halaman login membutuhkan waktu 563ms, halaman beranda pelanggan setelah login 1,016ms, setelah memilih beranda pelanggan 766ms, setelah logout memanggil halaman login 828ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

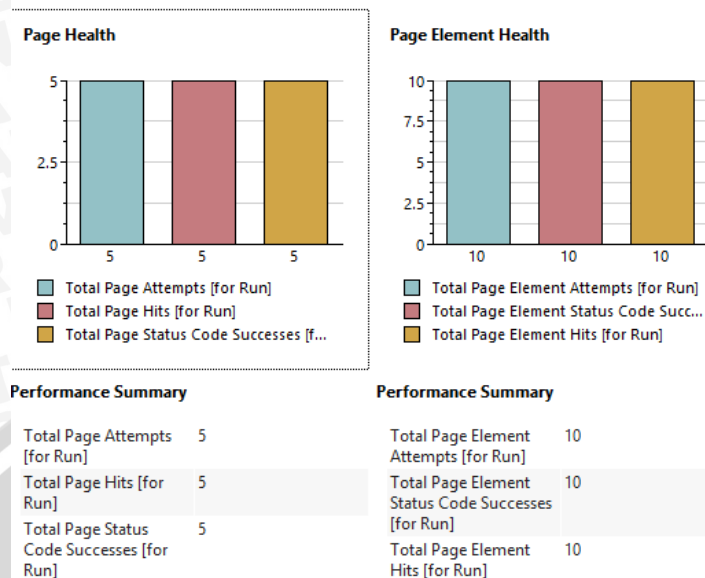
Response vs. Time Summary



Gambar 6. 12 Response VS Time Summary Kasus Uji PT_002

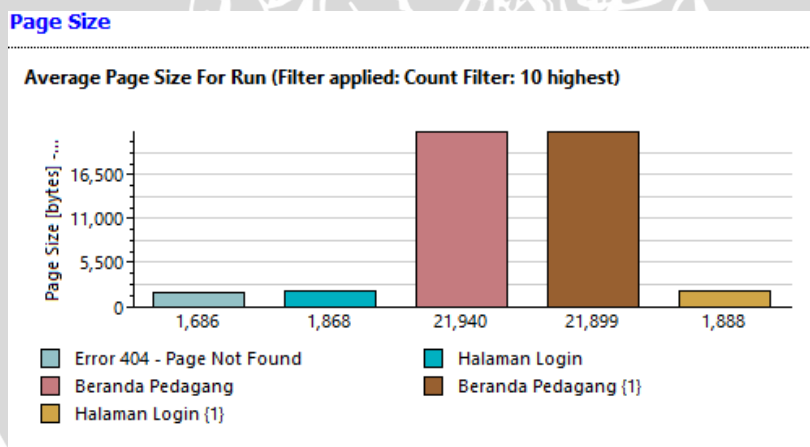
Pada gambar 6.12 ditunjukkan *average response time for all page* 894ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 277.3ms. Sedangkan *Average response time for all page elements* 333.2ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 99.8ms.

Server Health Summary



Gambar 6. 13 Server Health Summary Kasus Uji PT_002

Gambar 6.13 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.12 bahwa total page yang dijalankan sebanyak 5 halaman dan total elemen yang dijalankan ada 10 elemen.



Gambar 6. 14 Page Size kasus Uji PT_002

Gambar 6.14 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Halaman beranda memiliki ukuran paling besar karena pada saat membuka halaman beranda akan memanggil atau menampilkan daftar pesanan pelanggan.

Summary

Run Summary

Executed Test	kelolaPesananPelanggan
Active Users	0
Completed Users	1
Total Users	1
Elapsed Time [H:M:S]	0:00:32
Run Status	Complete
Displaying Results for Computer:	All Hosts

Page Summary

Total Page Attempts [for Run]	5
Total Page Hits [for Run]	5
Total Percent Healthy [for Run]	100
Total Healthy [for Run]	5
Average Response Time For All Pages [ms] [for Run]	894
Response Time Standard Deviation For All Pages [for Run]	277.3
Maximum Response Time For All Pages [ms] [for Run]	1,297
Minimum Response Time For All Pages [ms] [for Run]	563

Page Element Summary

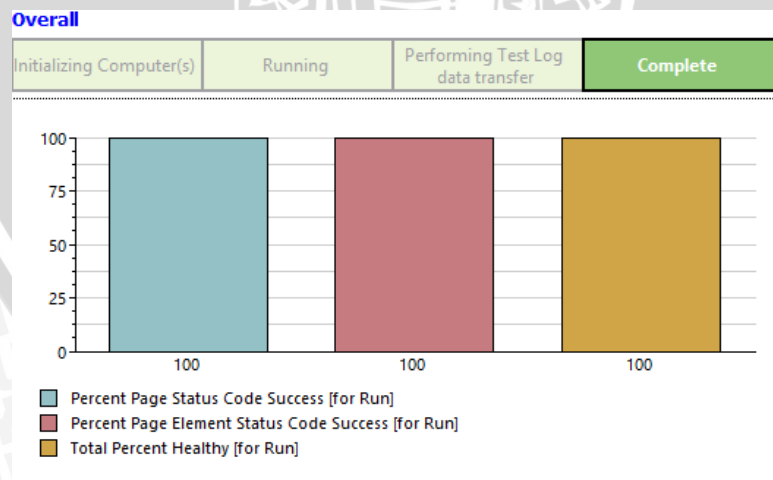
Total Page Element Attempts [for Run]	10
Total Page Element Hits [for Run]	10
Average Response Time For All Page Elements [ms] [for Run]	333.2
Response Time Standard Deviation For All Page Elements [for Run]	99.8

Gambar 6. 15 Kasus Uji PT_002

Gambar 6.15 merupakan rangkuman dari pengujian pada kasus uji PT_002. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

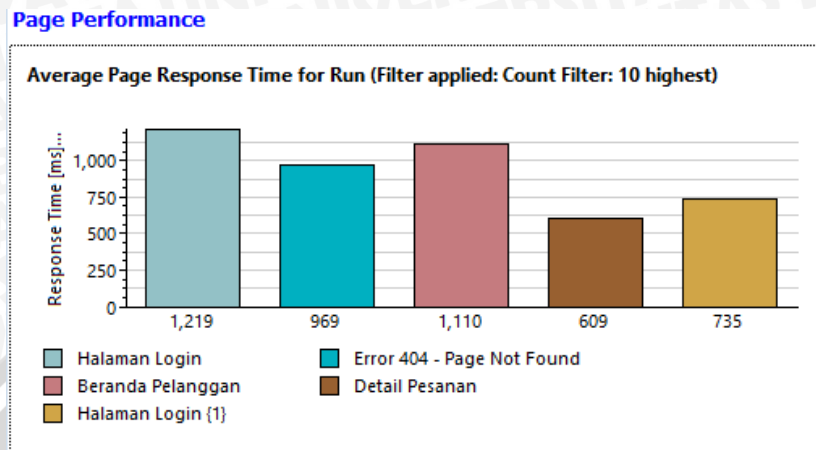
3. Kasus Uji PT_003

Pada bagian ini menjelaskan hasil pengujian kasus uji PT_003 dari fitur detail pesanan pelanggan yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



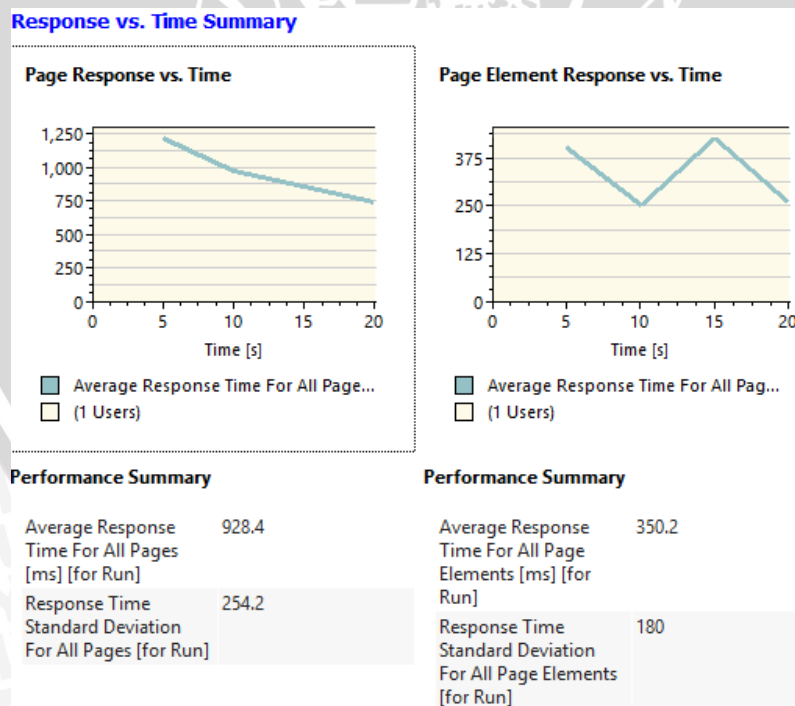
Gambar 6. 16 Overall Kasus Uji PT_003

Gambar 6.16 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_003. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.



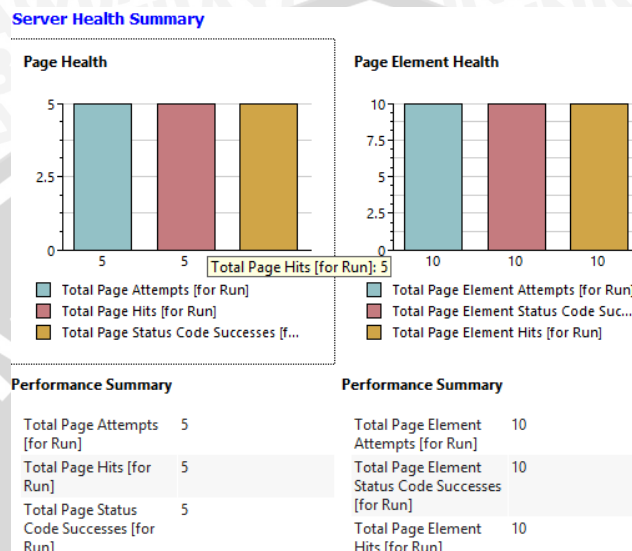
Gambar 6. 17 Page Performance Kasus Uji PT_003

Pada gambar 6.17 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur detail pesanan pelanggan. Dari gambar 6.15 dapat diketahui bahwa untuk membuka halaman login dibutuhkan waktu 1,219ms, beranda pelanggan 1,110ms, detail pesanan 609ms, dan ketika logout dan membuka halaman login 735ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.



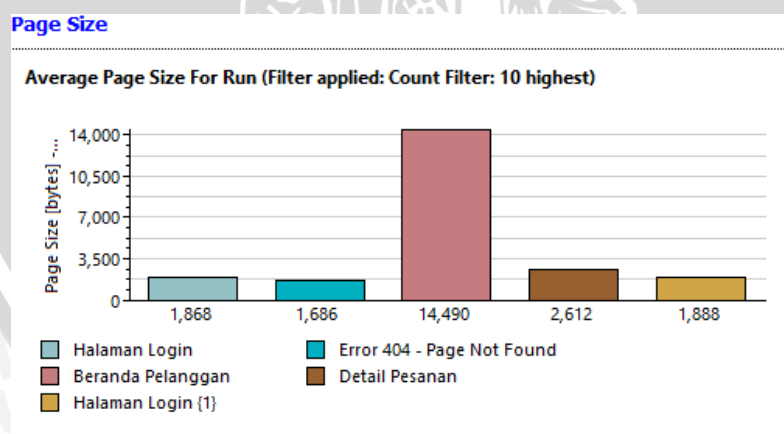
Gambar 6. 18 Response VS Time Summary Kasus Uji PT_003

Pada gambar 6.18 ditunjukkan *average response time for all page* 928.4ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 254.2ms. Sedangkan *Average response time for all page elements* 350.2ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 180ms.



Gambar 6. 19 Server Health Summary Kasus Uji PT_003

Gambar 6.19 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.17 bahwa total page yang dijalankan sebanyak 5 halaman dan total elemen yang dijalankan ada 10 elemen.



Gambar 6. 20 Page size kasus uji PT_003

Gambar 6.20 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Halaman beranda memiliki ukuran paling besar karena pada saat membuka halaman beranda akan memanggil atau menampilkan daftar pesanan pelanggan.

Summary

Run Summary		Page Summary	
Executed Test	detailPesananPelanggan	Total Page Attempts [for Run]	5
Active Users	0	Total Page Hits [for Run]	5
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	5
Elapsed Time [H:M:S]	0:00:22	Average Response Time For All Pages [ms] [for Run]	928.4
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	254.2
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	1,219
		Minimum Response Time For All Pages [ms] [for Run]	609

Page Element Summary

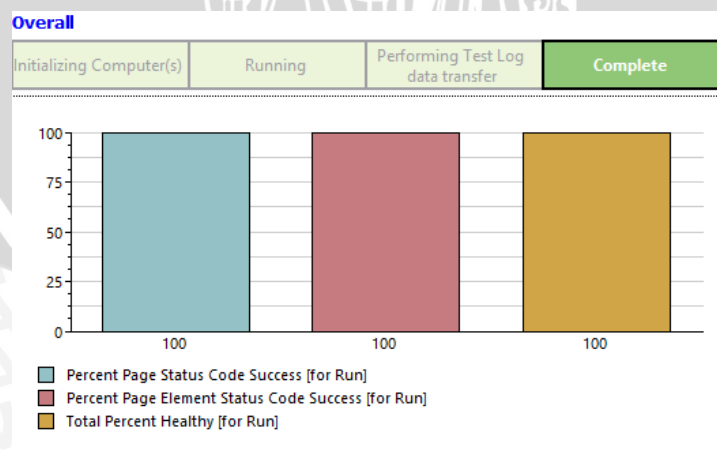
Total Page Element Attempts [for Run]	10
Total Page Element Hits [for Run]	10
Average Response Time For All Page Elements [ms] [for Run]	350.2
Response Time Standard Deviation For All Page Elements [for Run]	180

Gambar 6. 21 Summary Kasus Uji PT_003

Gambar 6.21 merupakan rangkuman dari pengujian pada kasus uji PT_003. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

4. Kasus Uji PT_004

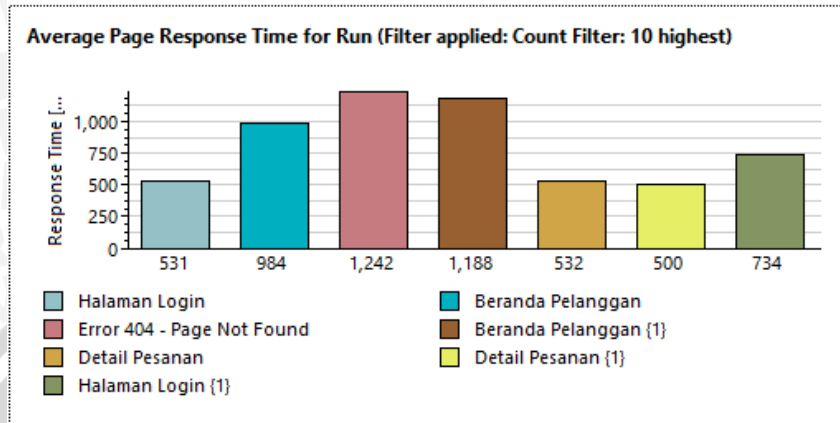
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_004 dari fitur konfirmasi yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 22 Overall Kasus Uji PT_004

Gambar 6.22 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_004. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

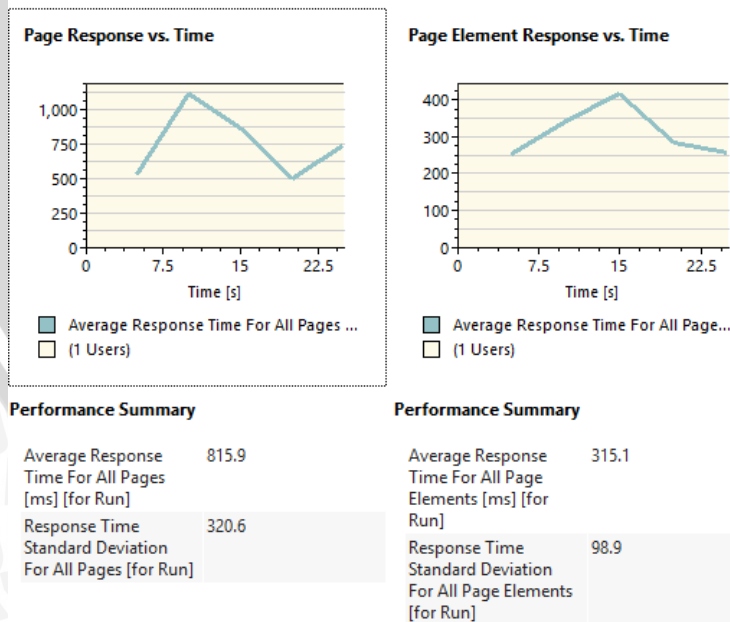
Page Performance



Gambar 6. 23 Page Performance Kasus Uji PT_004

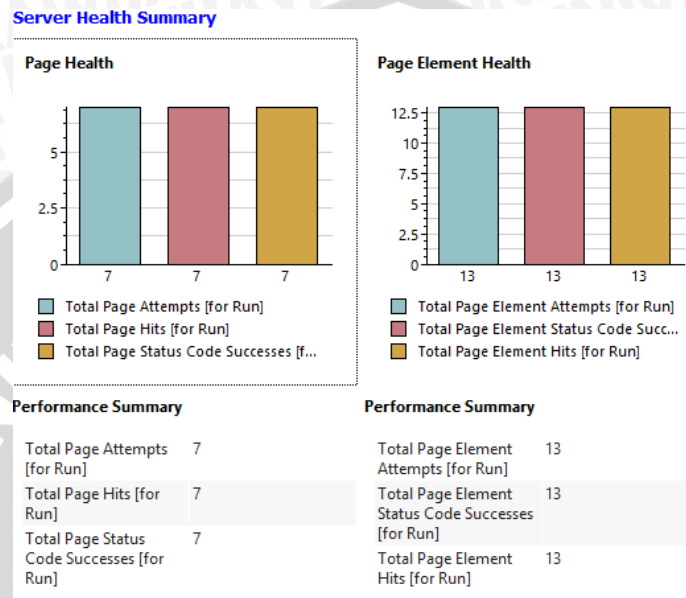
Pada gambar 6.20 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur konfirmasi pesanan pelanggan. Dari gambar 6.20 dapat diketahui bahwa untuk memanggil halaman login membutuhkan waktu 531ms, halaman beranda 1,242, halaman detail pesanan 532ms, halaman beranda(1) setelah megkonfirmasi 1,188, halaman detail pesansan(1) untuk cek apakah sudah terkonfirmasi 500ms dan halaman login setelah logout 734ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

Response vs. Time Summary



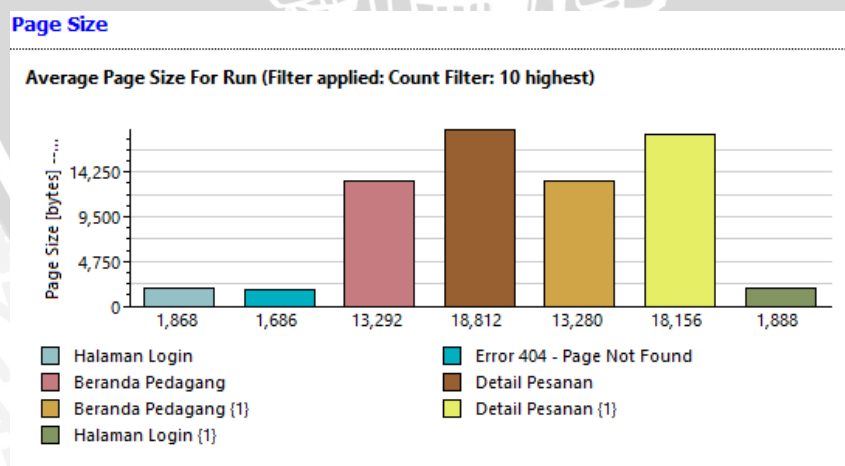
Gambar 6. 24 Response VS Time Summary Kasus Uji PT_004

Pada gambar 6.24 ditunjukkan *average response time for all page* 815ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 320.6ms. Sedangkan *Average response time for all page elements* 315.1ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 98.9ms.



Gambar 6. 25 Server Health Summary Kasus Uji PT_004

Gambar 6.25 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.22 bahwa total page yang dijalankan sebanyak 7 halaman dan total elemen yang dijalankan ada 13 elemen.



Gambar 6. 26 Page Size kasus uji PT_004

Gambar 6.26 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat

pada setiap halaman. Halaman detail pesanan memiliki ukuran paling besar karena pada saat membuka halaman detail pesanan akan memanggil atau menampilkan daftar pesanan pelanggan.

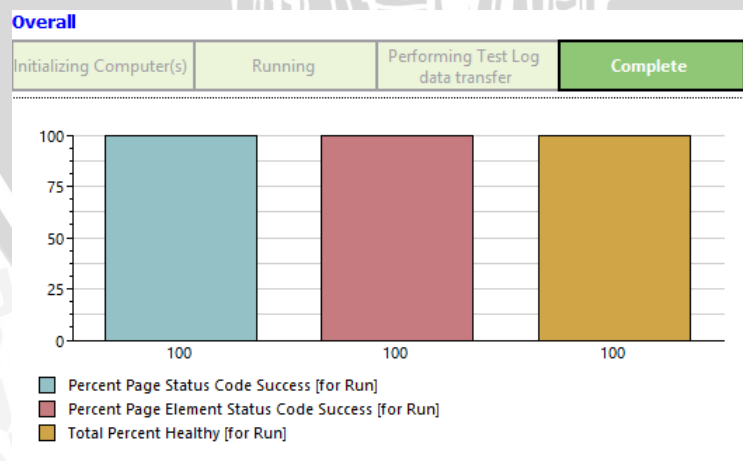
Summary			
Run Summary		Page Summary	
Executed Test	konfirmasiPelanggan	Total Page Attempts [for Run]	7
Active Users	0	Total Page Hits [for Run]	7
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	7
Elapsed Time [H:M:S]	0:00:32	Average Response Time For All Pages [ms] [for Run]	815.9
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	320.6
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	1,242
		Minimum Response Time For All Pages [ms] [for Run]	500
Page Element Summary			
Total Page Element Attempts [for Run]	13		
Total Page Element Hits [for Run]	13		
Average Response Time For All Page Elements [ms] [for Run]	315.1		
Response Time Standard Deviation For All Page Elements [for Run]	98.9		

Gambar 6. 27 Summary Kasus Uji PT_004

Gambar 6.27 merupakan rangkuman dari pengujian pada kasus uji PT_004. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

5. Kasus Uji PT_005

Pada bagian ini menjelaskan hasil pengujian kasus uji PT_005 dari fitur pesanan khusus yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.

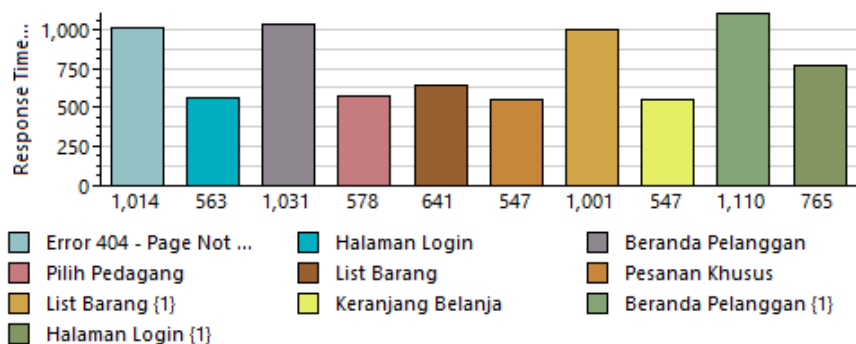


Gambar 6. 28 Overall Kasus Uji PT_005

Gambar 6.28 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_005. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

Page Performance

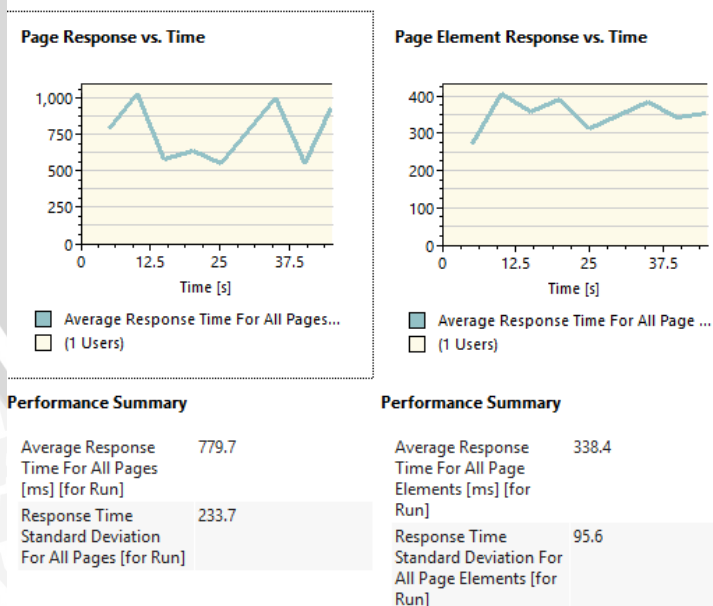
Average Page Response Time for Run (Filter applied: Count Filter: 10 highest)



Gambar 6. 29 Page Performance Kasus Uji PT_005

Pada gambar 6.29 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur pesanan khusus. Dari gambar 6.5 dapat diketahui bahwa untuk memanggil halaman login butuh waktu 563ms, beranda 1,031, pilih pedagang 578ms, list barang 641ms, pesanan khusus 574, list barang setelah submit pesan khusus 1,001, keranjang belanja 547ms, beranda pelanggan setelah submit pesanan 1,110ms dan logout untuk menampilkan halaman login 765ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

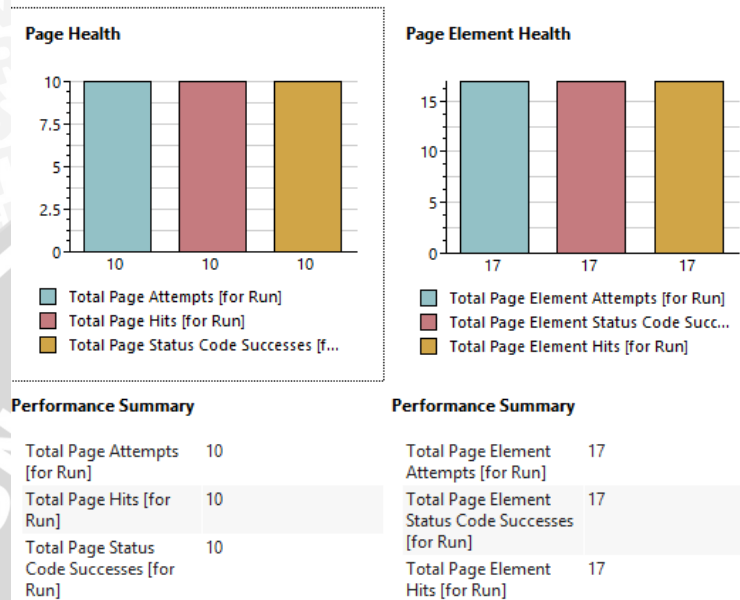
Response vs. Time Summary



Gambar 6. 30 Response VS Time Summary Kasus Uji PT_005

Pada gambar 6.30 ditunjukkan *average response time for all page* 779.7ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 233.7ms. Sedangkan *Average response time for all page elements* 338ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 95.6ms.

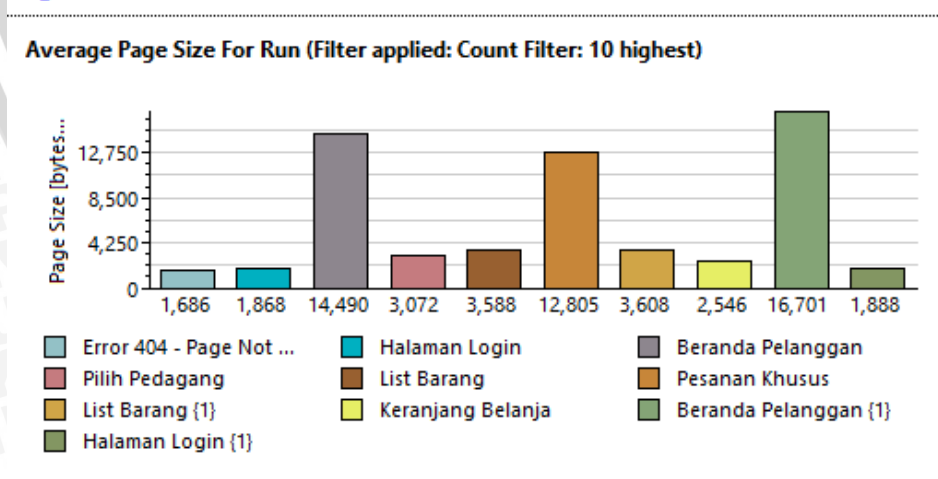
Server Health Summary



Gambar 6. 31 Server Health Summary Kasus Uji PT_005

Gambar 6.31 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.6 bahwa total page yang dijalankan sebanyak 10 halaman dan total elemen yang dijalankan ada 17 elemen.

Page Size



Gambar 6. 32 Page Size kasus uji PT_005

Gambar 6.32 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat

pada setiap halaman. Halaman beranda memiliki ukuran besar karena pada saat membuka halaman beranda akan memanggil atau menampilkan daftar pesanan pelanggan. Sedangkan halaman pesanan khusus juga memiliki ukuran yang besar karena menampilkan form untuk mengisi pesanan.

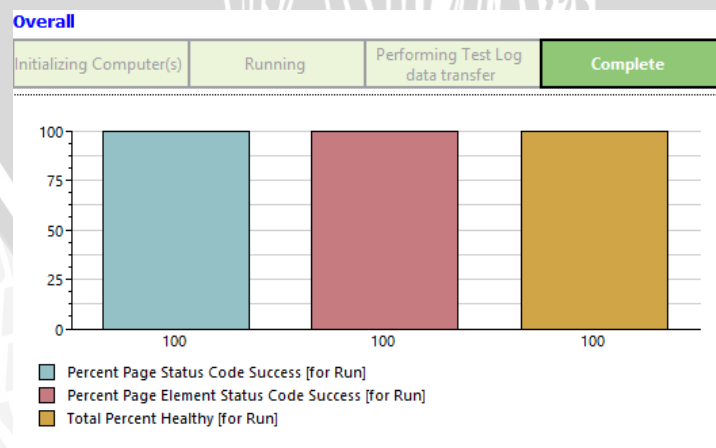
Summary			
Run Summary		Page Summary	
Executed Test	pesananKhusus	Total Page Attempts [for Run]	10
Active Users	0	Total Page Hits [for Run]	10
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	10
Elapsed Time [H:M:S]	0:00:52	Average Response Time For All Pages [ms] [for Run]	779.7
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	233.7
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	1,110
		Minimum Response Time For All Pages [ms] [for Run]	547
Page Element Summary			
Total Page Element Attempts [for Run]	17		
Total Page Element Hits [for Run]	17		
Average Response Time For All Page Elements [ms] [for Run]	338.4		
Response Time Standard Deviation For All Page Elements [for Run]	95.6		

Gambar 6. 33 Summary Kasus Uji PT_005

Gambar 6.33 merupakan rangkuman dari pengujian pada kasus uji PT_005. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

6. Kasus Uji PT_006

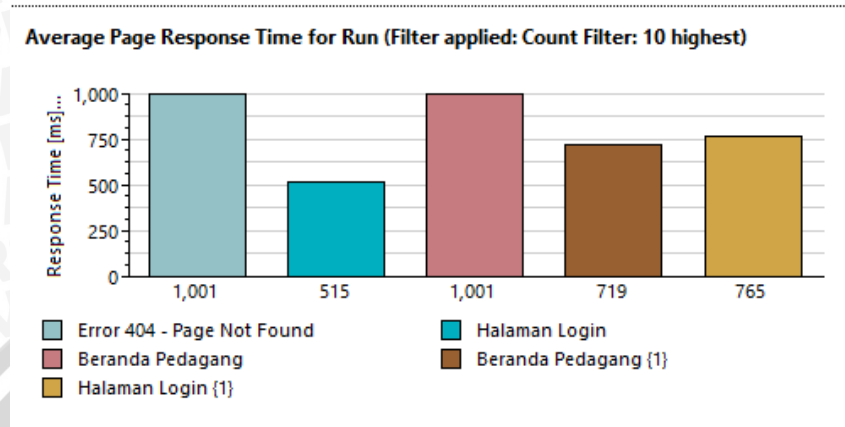
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_006 dari fitur kelola pesanan pedagang yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 34 Overall Kasus Uji PT_006

Gambar 6.34 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_006. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

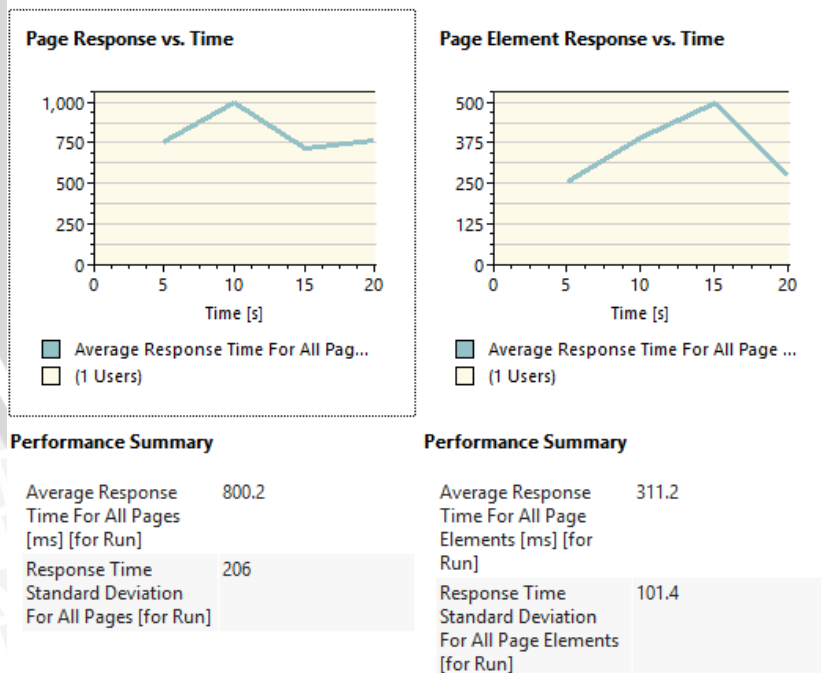
Page Performance



Gambar 6. 35 Page Performance Kasus Uji PT_006

Pada gambar 6.35 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola pesanan pedagang. Dari gambar 6.5 dapat diketahui bahwa untuk membuka halaman login membutuhkan waktu 515ms, beranda pedagang butuh waktu 1,001ms, memilih kelola pesanan 719ms dan setelah logout untuk membuka halaman login 765ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

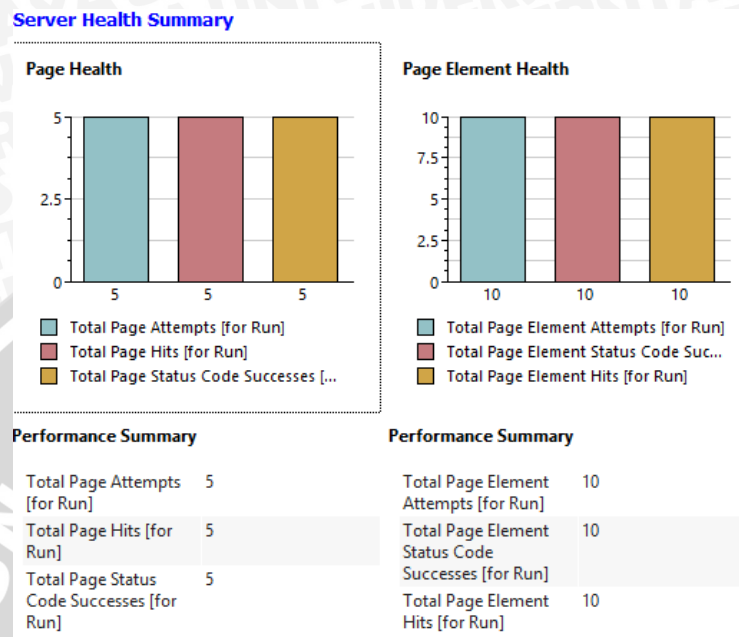
Response vs. Time Summary



Gambar 6. 36 Response VS Time Summary Kasus Uji PT_006

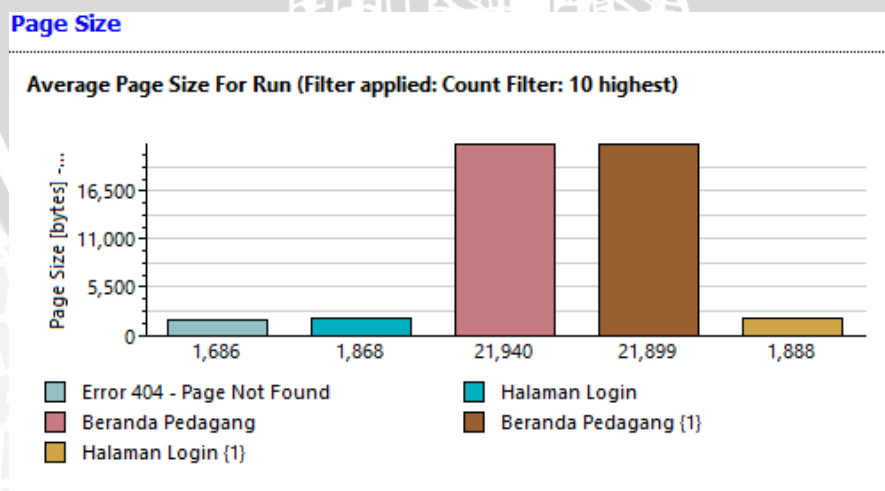
Pada gambar 6.36 ditunjukkan *average response time for all page* 800.2ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga

menunjukkan *response time standard deviation* 206ms. Sedangkan *Average response time for all page elements* 311.2ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan *standard deviation* 101.4ms.



Gambar 6. 37 Server Health Summary Kasus Uji PT_006

Gambar 6.37 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.6 bahwa total page yang dijalankan sebanyak 5 halaman dan total elemen yang dijalankan ada 10 elemen.



Gambar 6. 38 page size kasus uji PT_006

Gambar 6.38 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Halaman beranda memiliki ukuran paling besar yaitu 21,940

dan 21,899 hal ini dikarenakan pada saat membuka halaman beranda akan memanggil atau menampilkan daftar pesanan pelanggan.

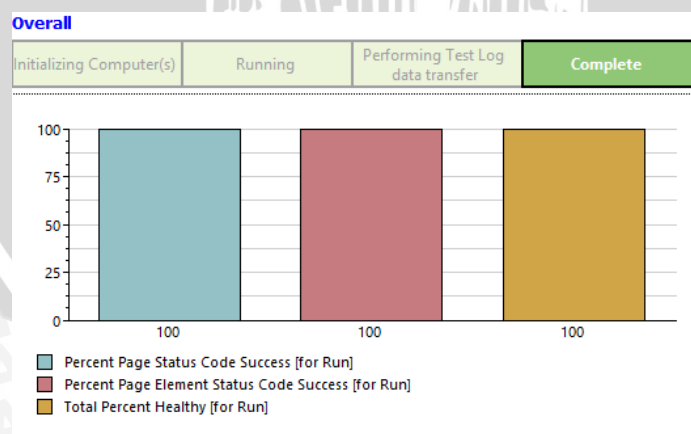
Summary			
Run Summary		Page Summary	
Executed Test	kelolaPesananPedagang	Total Page Attempts [for Run]	5
Active Users	0	Total Page Hits [for Run]	5
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	5
Elapsed Time [H:M:S]	0:00:22	Average Response Time For All Pages [ms] [for Run]	800.2
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	206
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	1,001
		Minimum Response Time For All Pages [ms] [for Run]	515
Page Element Summary			
Total Page Element Attempts [for Run]	10		
Total Page Element Hits [for Run]	10		
Average Response Time For All Page Elements [ms] [for Run]	311.2		
Response Time Standard Deviation For All Page Elements [for Run]	101.4		

Gambar 6. 39 Summary Kasus Uji PT_006

Gambar 6.39 merupakan rangkuman dari pengujian pada kasus uji PT_006. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

7. Kasus Uji PT_007

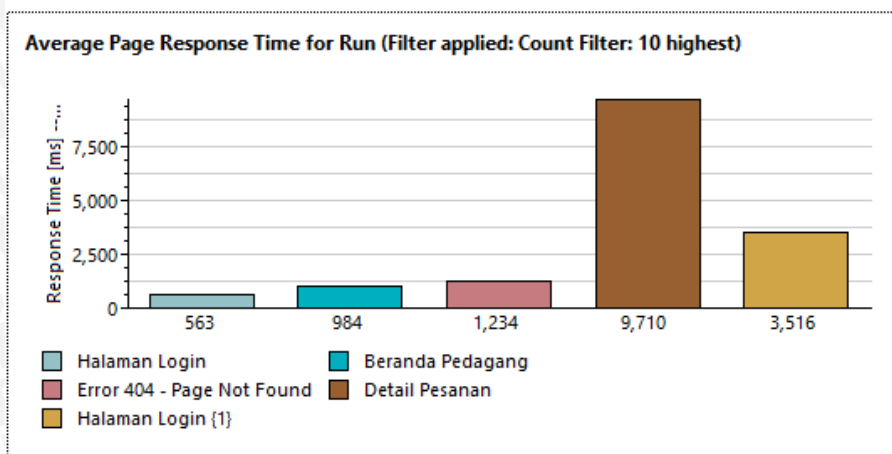
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_007 dari fitur detail pesanan pedagang yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 40 Overall Kasus Uji PT_007

Gambar 6.40 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_007. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

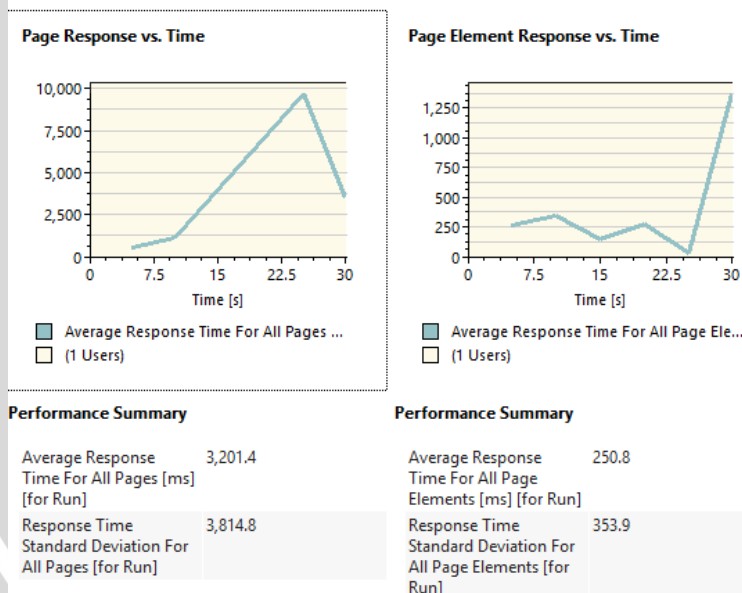
Page Performance



Gambar 6. 41 Page Performance Kasus Uji PT_007

Pada gambar 6.41 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur detail pesanan pedagang. Dari gambar 6.35 dapat diketahui bahwa untuk membuka halaman login butuh waktu 563ms, halaman beranda 984ms, halaman detail pesanan 9,710 dan halaman login setelah logout 3,516ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

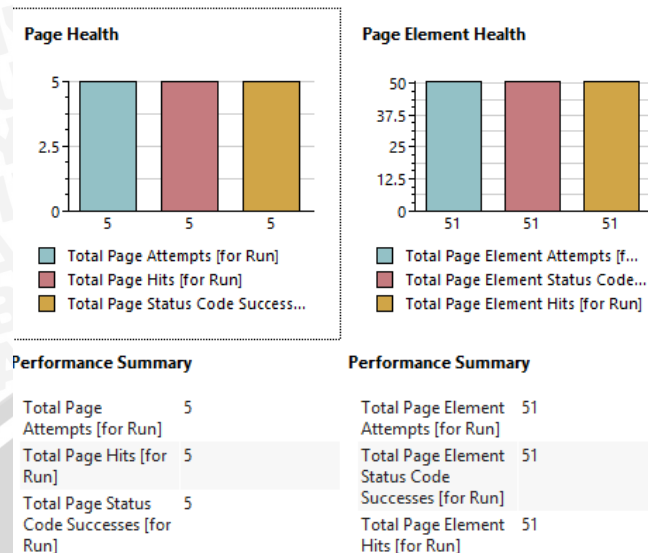
Response vs. Time Summary



Gambar 6. 42 Response VS Time Summary Kasus Uji PT_007

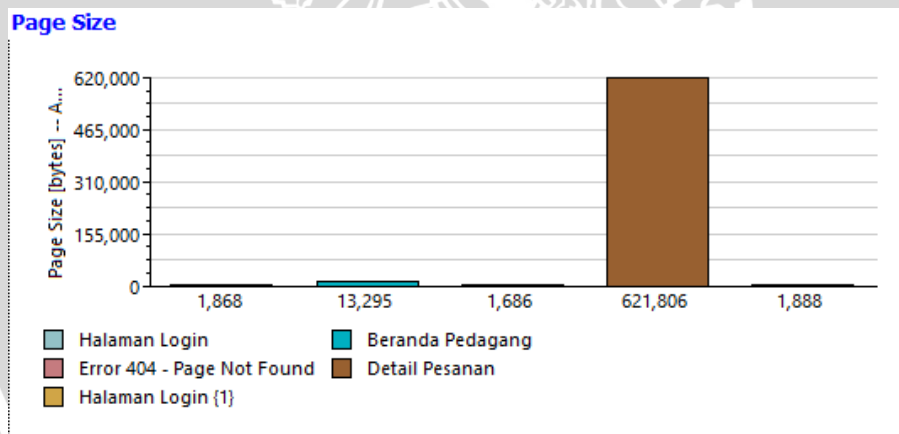
Pada gambar 6.42 ditunjukkan *average response time for all page* 3,438.8ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 3,785.1ms. Sedangkan *Average response time for all page elements* 500.4ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 644.4ms.

Server Health Summary



Gambar 6. 43 Server Health Summary Kasus Uji PT_007

Gambar 6.43 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.37 bahwa total page yang dijalankan sebanyak 6 halaman dan total elemen yang dijalankan ada 89 elemen.

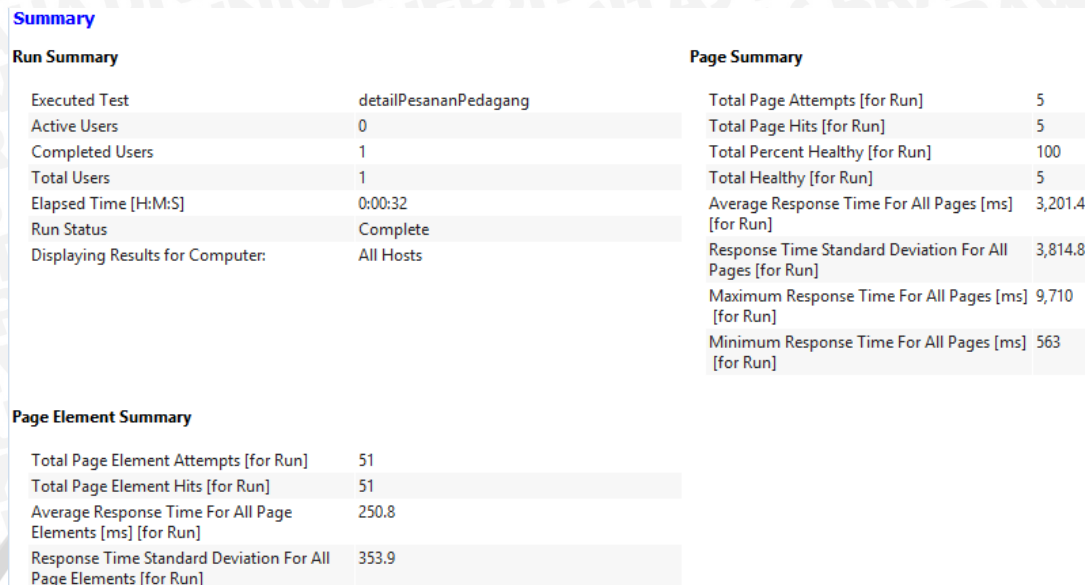


Gambar 6. 44 Page size kasus uji PT_007

Gambar 6.44 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Pada halaman detail pesanan memiliki ukuran yang besar yaitu 621,806 hal ini karena pada halaman tersebut memanggil banyak data yaitu data pesanan pelanggan, data pesanan khusus, data diri pelanggan dan memanggil maps untuk melihat alamat pelanggan.

Pada gambar 6.45 ditunjukkan screenshot rangkuman dari pengujian pada kasus uji PT_007. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon

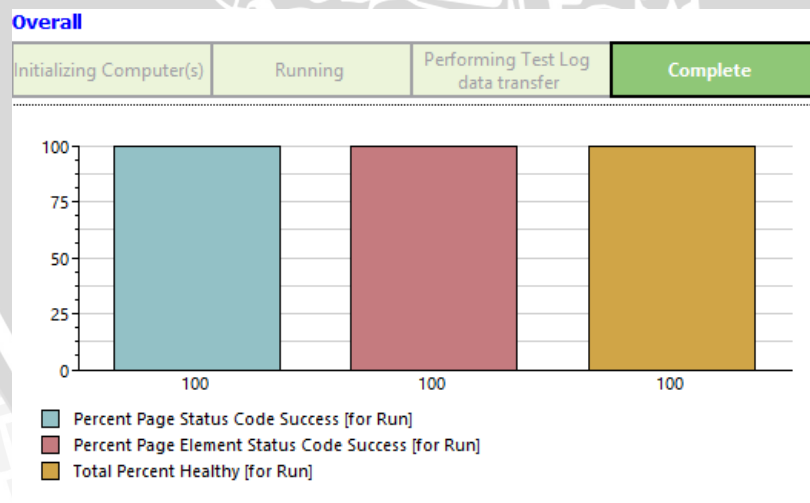
waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.



Gambar 6. 45 Summary Kasus Uji PT_007

8. Kasus Uji PT_008

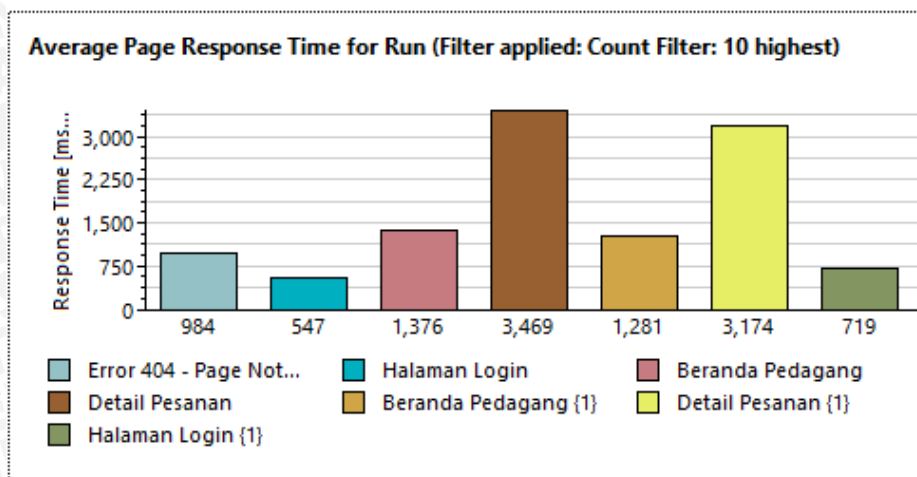
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_008 dari fitur kelola barang dagangan yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 46 Kasus Uji PT_008.

Gambar 6.46 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_008. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

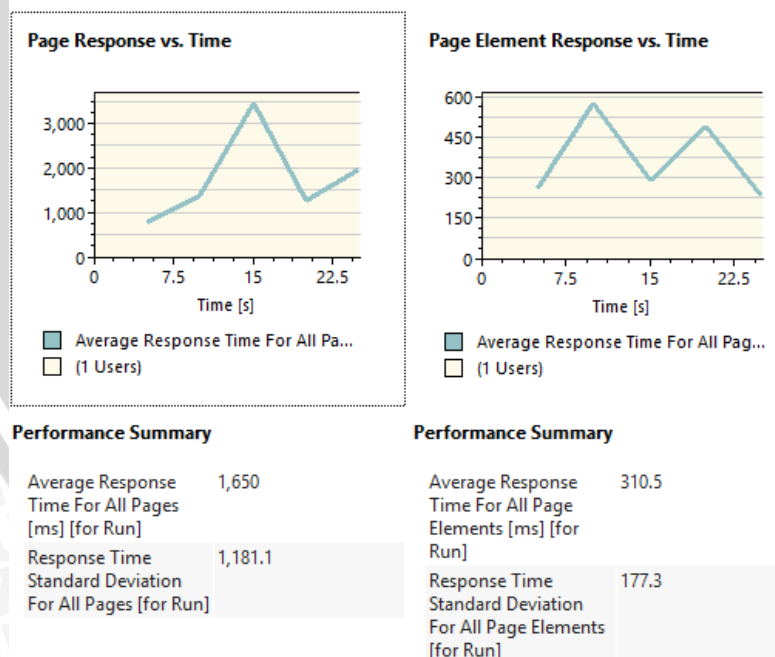
Page Performance



Gambar 6. 47 Page Performance Kasus Uji PT_008

Pada gambar 6.47 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur konfirmasi pesanan pedagang. Dari gambar 6.40 dapat diketahui bahwa untuk membuka halaman login membutuhkan waktu 547ms, beranda peadgang 1,376, detail pesanan 3,469, beranda pedagang setelah melakukan konfrimasi 1,281ms, detail pesanan untuk cek apakah sudah terkonfirmasi dan halaman login setelah logout 719ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

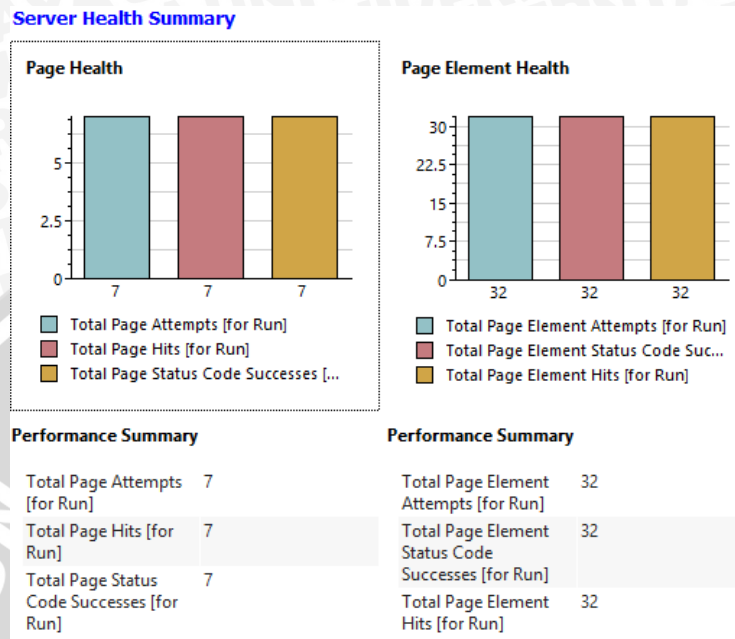
Response vs. Time Summary



Gambar 6. 48 Response VS Time Summary Kasus Uji PT_008

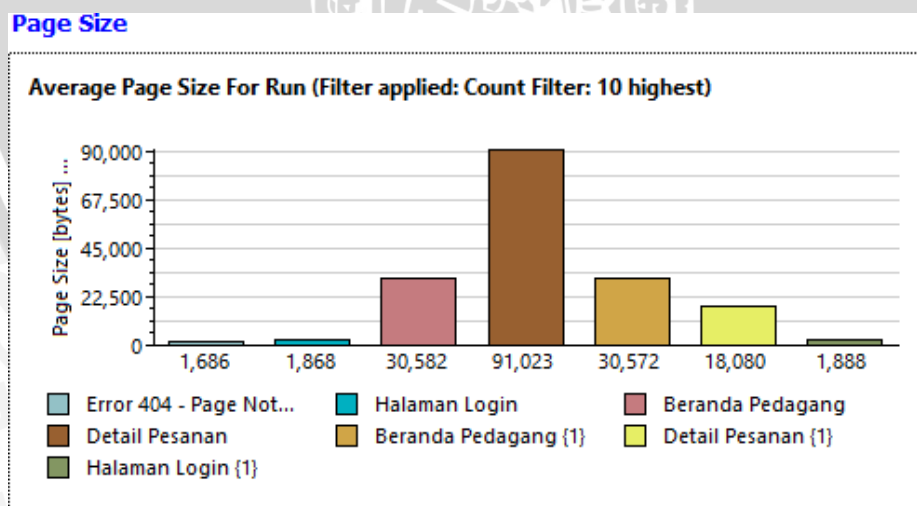
Pada gambar 6.48 ditunjukkan *average response time for all page* 1,650ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga

menunjukkan *response time standard deviation* 1,181.1ms. Sedangkan *Average response time for all page elements* 310.5ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan *standard deviation* 177.3ms.



Gambar 6. 49 Server Health Summary Kasus Uji PT_008

Gambar 6.49 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.42 bahwa total page yang dijalankan sebanyak 7 halaman dan total elemen yang dijalankan ada 32 elemen.



Gambar 6. 50 Page size kasus uji PT_008

Gambar 6.50 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Pada halaman detail pesanan memiliki ukuran yang besar yaitu 91,023 hal ini karena pada halaman tersebut memanggil banyak data yaitu

data pesanan pelanggan, data pesanan khusus, data diri pelanggan dan memanggil maps untuk melihat alamat pelanggan. Halaman beranda pedagan juga memiliki ukuran besar karena menampilkan daftar pesanan pelanggan.

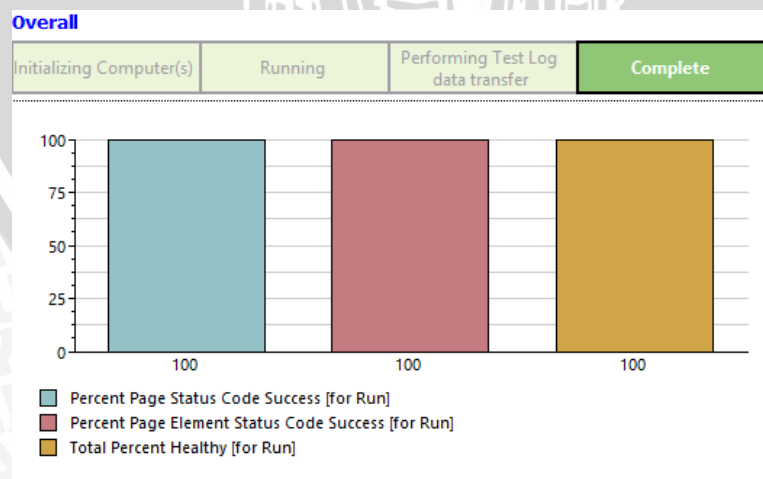
Summary			
Run Summary		Page Summary	
Executed Test	konfirmasiPedagang	Total Page Attempts [for Run]	7
Active Users	0	Total Page Hits [for Run]	7
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	7
Elapsed Time [H:M:S]	0:00:32	Average Response Time For All Pages [ms] [for Run]	1,650
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	1,181.1
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	3,469
		Minimum Response Time For All Pages [ms] [for Run]	547
Page Element Summary			
Total Page Element Attempts [for Run]	32		
Total Page Element Hits [for Run]	32		
Average Response Time For All Page Elements [ms] [for Run]	310.5		
Response Time Standard Deviation For All Page Elements [for Run]	177.3		

Gambar 6. 51 Summary Kasus Uji PT_008

Gambar 6.51 merupakan rangkuman dari pengujian pada kasus uji PT_008. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

9. Kasus Uji PT_009

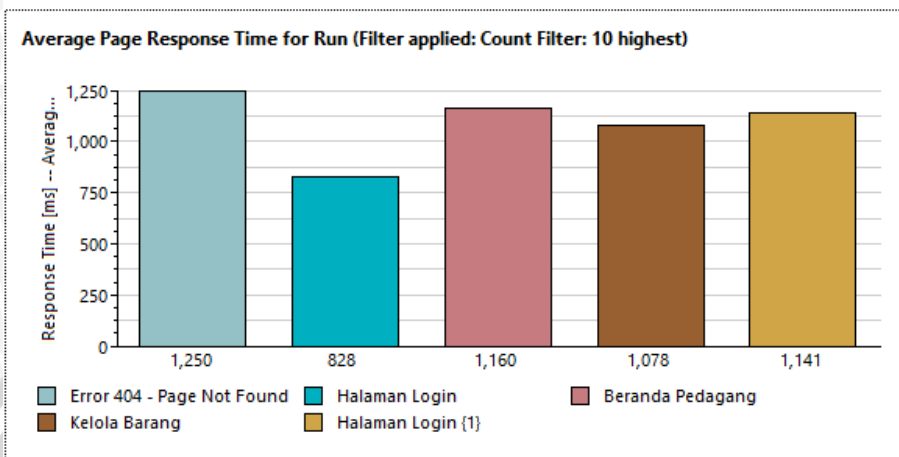
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_009 dari fitur kelola barang dagangan yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 52 Overall Kasus Uji PT_009

Gambar 6.52 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_009. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

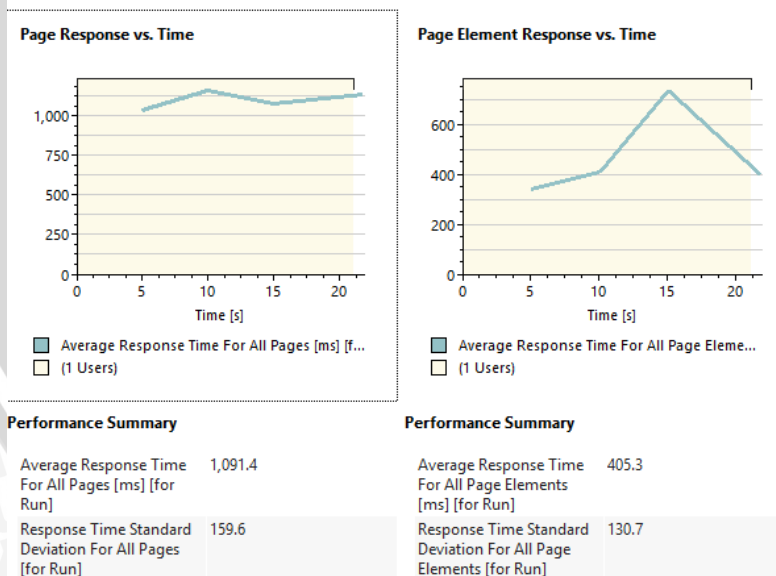
Page Performance



Gambar 6. 53 Page Performance Kasus Uji PT_009

Pada gambar 6.53 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola barang dagangan. Dari gambar 6.53 dapat diketahui bahwa untuk membuka halaman login membutuhkan waktu 828ms, beranda pedagang 1,160ms, kelola barang 1,078 dan halaman login setelah logout 1,141ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

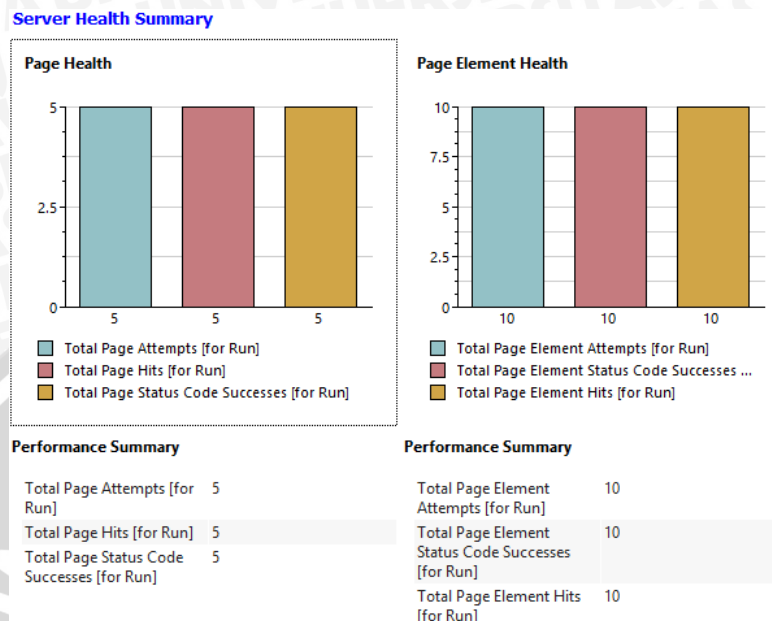
Response vs. Time Summary



Gambar 6. 54 Response VS Time Summary Kasus Uji PT_009

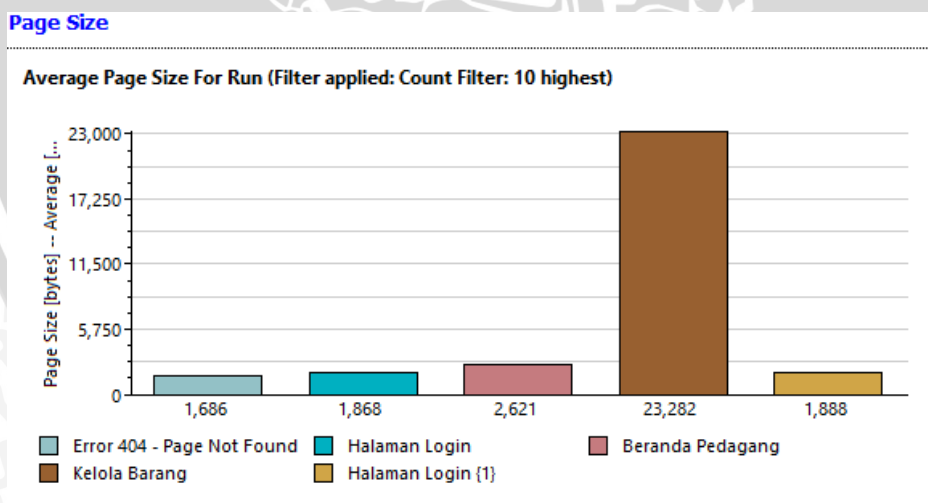
Pada gambar 6.54 ditunjukkan *average response time for all page* 1,091.4ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 159.6ms. Sedangkan *Average*

response time for all page elements 405.3ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 130.7ms.



Gambar 6. 55 Server Health Summary Kasus Uji PT_009

Gambar 6.47 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.47 bahwa total page yang dijalankan sebanyak 5 halaman dan total elemen yang dijalankan ada 10 elemen.



Gambar 6. 56 Page size kasus uji PT_009

Pada gambar 6.65 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Halaman kelola barang memiliki ukuran paling besar yaitu 23,282byte hal ini dikarenakan halaman ini memanggil daftar barang dagangan peangang sehingga membutuhkan banyak data.

Summary

Run Summary

Executed Test	kelolaBarangDagangan
Active Users	0
Completed Users	1
Total Users	1
Elapsed Time [H:M:S]	0:00:22
Run Status	Complete
Displaying Results for Computer:	All Hosts

Page Summary

Total Page Attempts [for Run]	5
Total Page Hits [for Run]	5
Total Percent Healthy [for Run]	100
Total Healthy [for Run]	5
Average Response Time For All Pages [ms] [for Run]	1,091.4
Response Time Standard Deviation For All Pages [for Run]	159.6
Maximum Response Time For All Pages [ms] [for Run]	1,250
Minimum Response Time For All Pages [ms] [for Run]	828

Page Element Summary

Total Page Element Attempts [for Run]	10
Total Page Element Hits [for Run]	10
Average Response Time For All Page Elements [ms] [for Run]	405.3
Response Time Standard Deviation For All Page Elements [for Run]	130.7

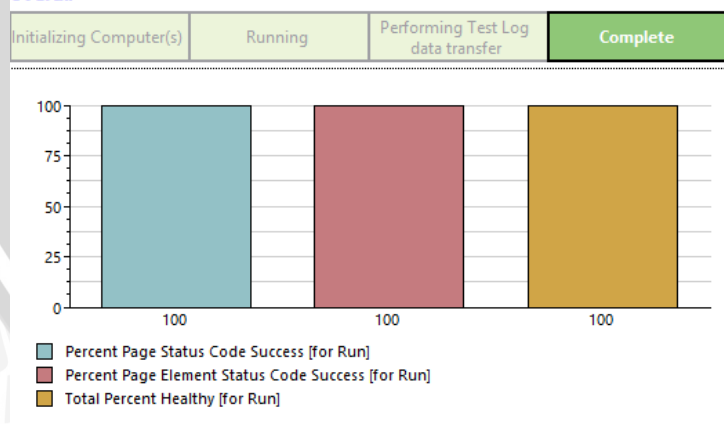
Gambar 6. 57 Summary Kasus Uji PT_009

Gambar 6.57 merupakan rangkuman dari pengujian pada kasus uji PT_009. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

10.Kasus Uji PT_010

Pada bagian ini menjelaskan hasil pengujian kasus uji PT_010 dari fitur tambah barang dagangan yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.

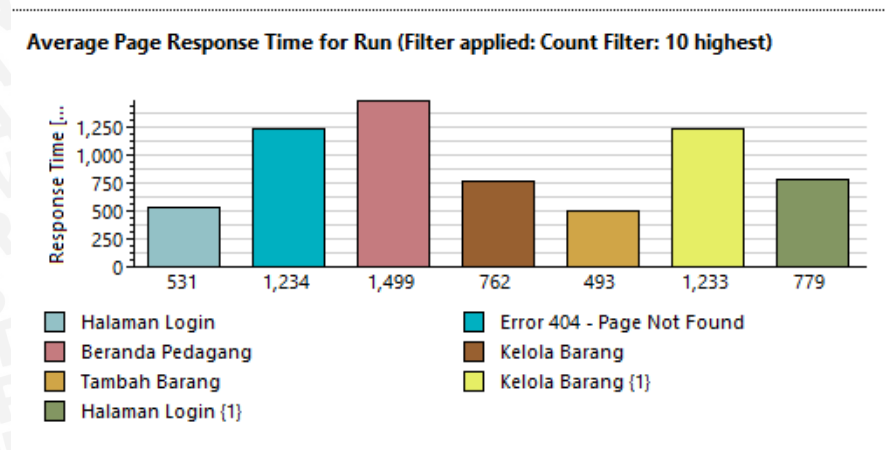
Overall



Gambar 6. 58 Overall Kasus Uji PT_010

Gambar 6.58 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_010. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

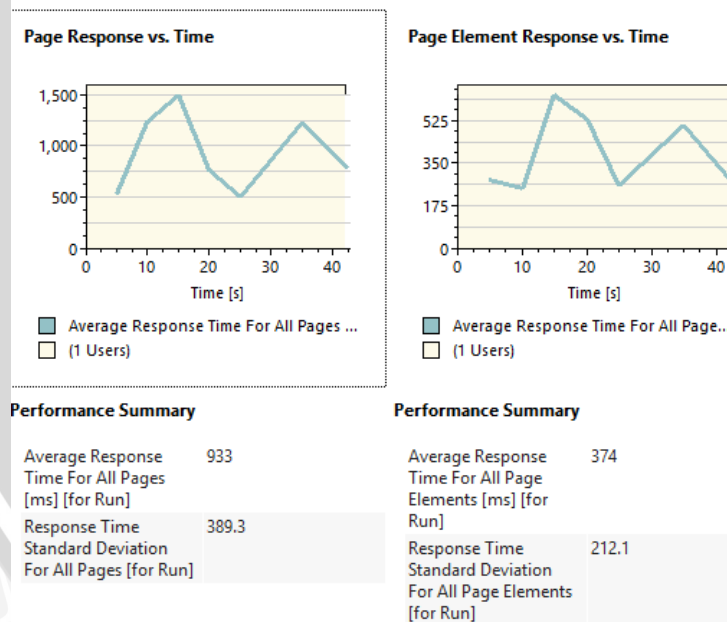
Page Performance



Gambar 6. 59 Page Performance Kasus Uji PT_010

Pada gambar 6.59 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola barang dagangan. Dari gambar 6.50 dapat diketahui bahwa untuk membuka halaman login membutuhkan waktu 531ms, beranda pedagang 1,499ms, kelola barang 762ms, tambah barang 493ms, kelola barang setelah tambah barang 1,233ms dan halaman login setelah melakukan logout 779ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

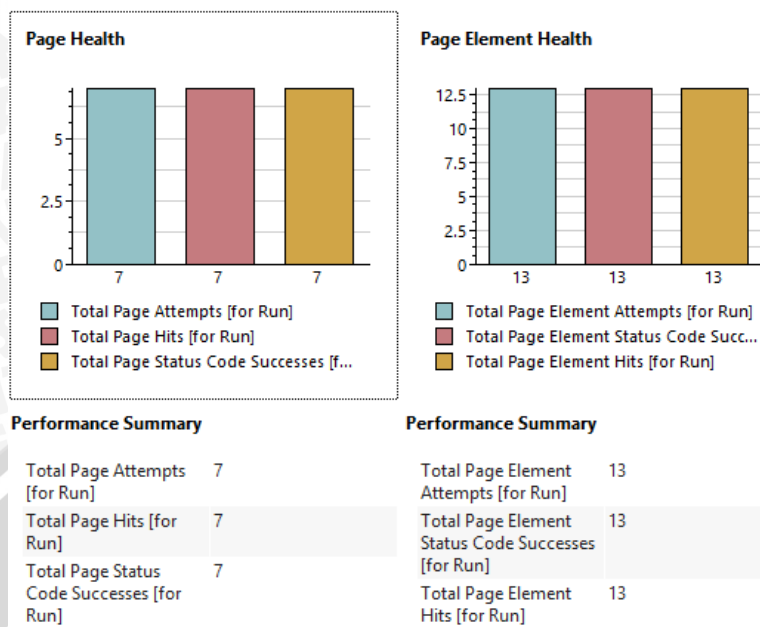
Response vs. Time Summary



Gambar 6. 60 Response VS Time Summary Kasus Uji PT_010

Pada gambar 6.60 ditunjukkan *average response time for all page* 933ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 389,3ms. Sedangkan *Average response time for all page elements* 374ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 212,1ms.

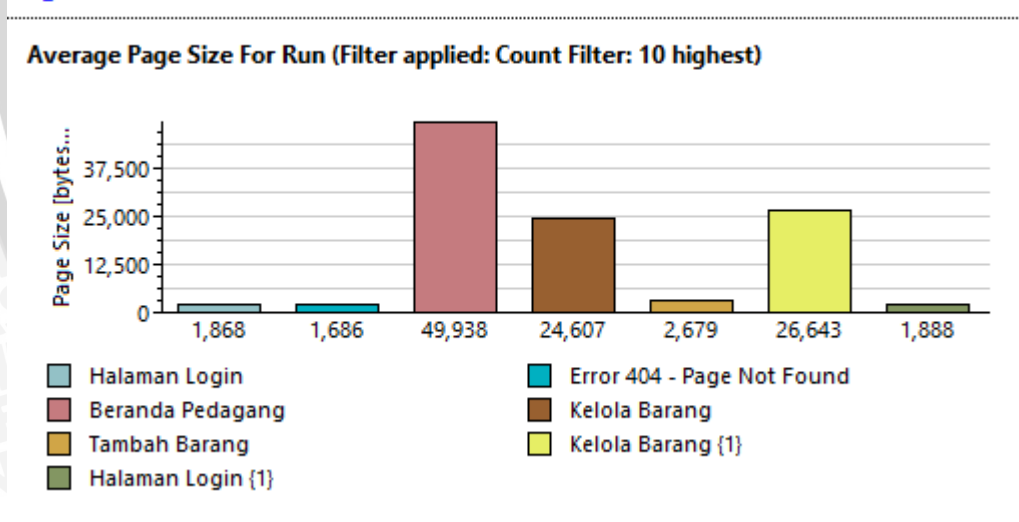
Server Health Summary



Gambar 6. 61 Server Health Summary Kasus Uji PT_010

Gambar 6.61 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.61 bahwa total page yang dijalankan sebanyak 7 halaman dan total elemen yang dijalankan ada 13 elemen.

Page Size



Gambar 6. 62 Page Size kasus uji PT_10

Pada gambar 6.62 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Halaman beranda memiliki ukuran paling besar yaitu 49,938byte karena pada saat membuka halaman beranda akan memanggil atau menampilkan daftar pesanan pelanggan. Sedangkan kelola barang yang

memanggil halaman kelola barang dagangan yang memanggil daftar barang dagangan memiliki ukuran 24,607byte.

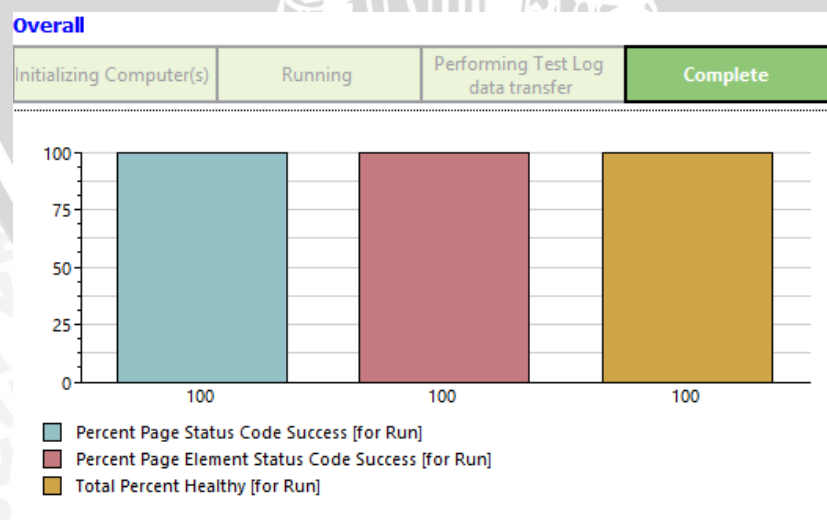
Summary			
Run Summary		Page Summary	
Executed Test	detailPesananPedagang	Total Page Attempts [for Run]	6
Active Users	0	Total Page Hits [for Run]	6
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	6
Elapsed Time [H:M:S]	0:00:42	Average Response Time For All Pages [ms] [for Run]	3,438.8
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	3,785.1
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	10,198
		Minimum Response Time For All Pages [ms] [for Run]	841
Page Element Summary			
Total Page Element Attempts [for Run]	89		
Total Page Element Hits [for Run]	89		
Average Response Time For All Page Elements [ms] [for Run]	500.4		
Response Time Standard Deviation For All Page Elements [for Run]	644.4		

Gambar 6. 63 Summary Kasus Uji PT_010

Gambar 6.63 merupakan rangkuman dari pengujian pada kasus uji PT_010. Pada bagian *run summary* menjelaskan banyak user yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

11.Kasus Uji PT_011

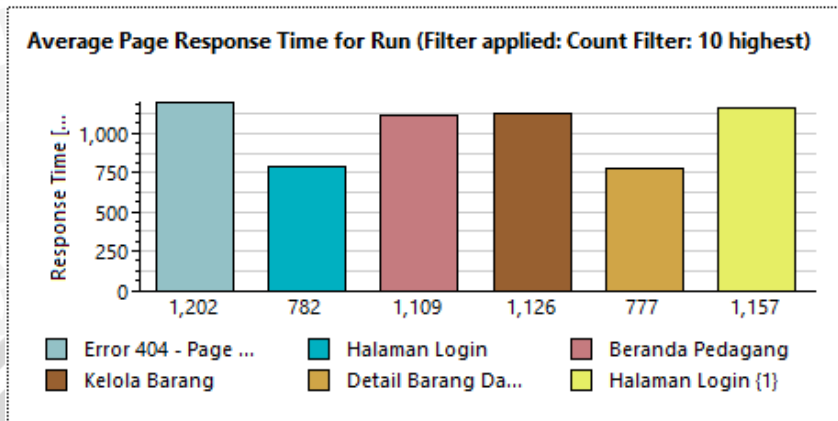
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_011 dari fitur detail barang pedagang yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 64 Overall Kasus Uji PT_011

Gambar 6.64 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_011. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

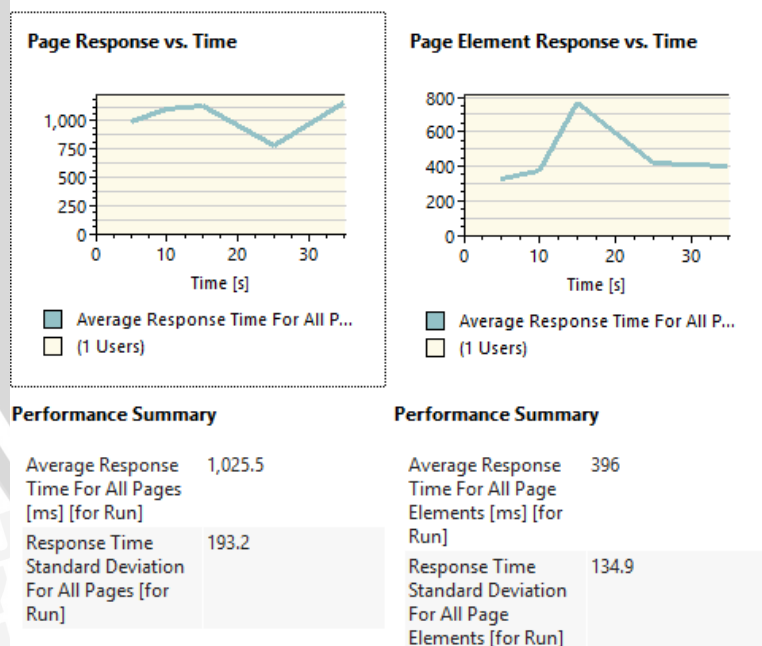
Page Performance



Gambar 6. 65 Page Performance Kasus uji PT_011

Pada gambar 6.65 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur detail barang pedagang. Dari gambar 6.65 dapat diketahui bahwa untuk memanggil halaman login membutuhkan waktu 782ms, beranda 1,109ms, kelola barang 1,126ms, detail barang dagangan 777 dan halaman login setelah logout 1,157ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

Response vs. Time Summary

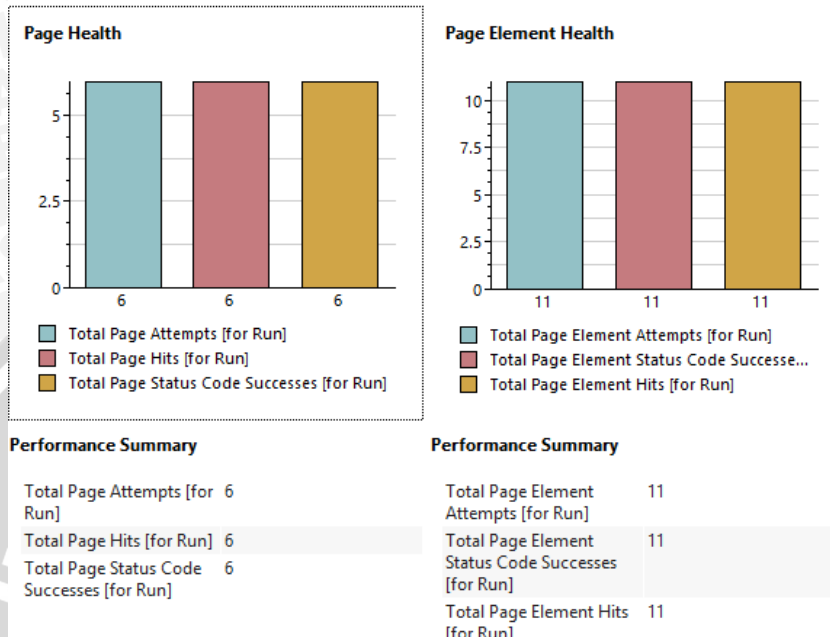


Gambar 6. 66 Response VS Time Summary Kasus Uji PT_011

Pada gambar 6.66 ditunjukkan *average response time for all page* 1,025.5ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga

menunjukkan *response time standard deviation* 193.2ms. Sedangkan *Average response time for all page elements* 396ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 134.9ms.

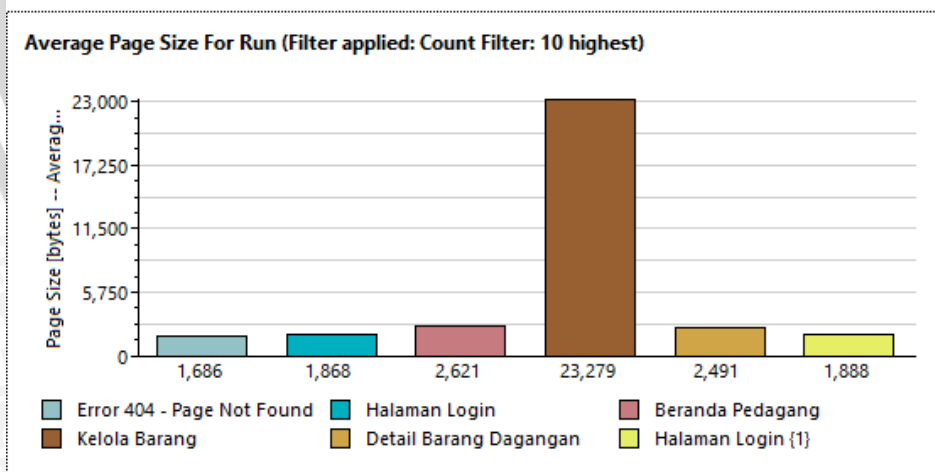
Server Health Summary



Gambar 6. 67 Server Health Summary Kasus Uji PT_011

Gambar 6.67 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.67 bahwa total page yang dijalankan sebanyak 6 halaman dan total elemen yang dijalankan ada 11 elemen.

Page Size



Gambar 6. 68 Page Size kasus uji PT_011

Pada gambar 6.68 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Pada halaman kelola barang memiliki ukuran paling besar yaitu 23, 279byte, karena kelola barang menampilkan daftar dagangan yang

diambil dari database, sehingga membutuhkan banyak data yang diambil dan ditampilkan.

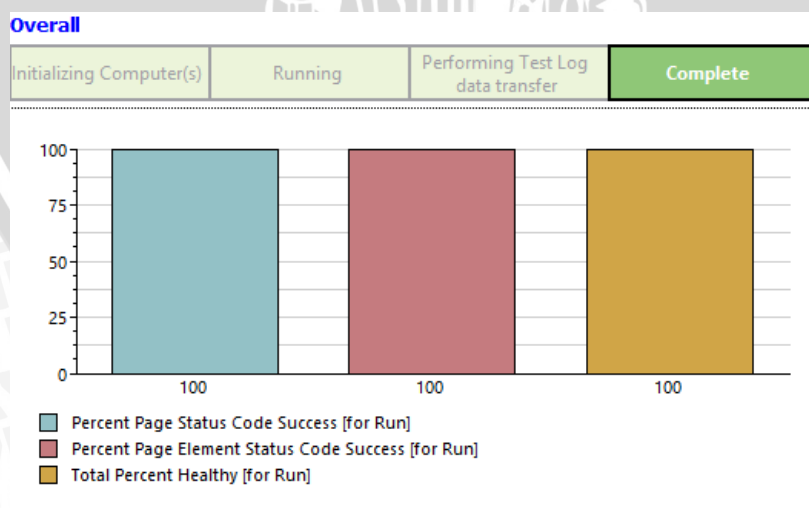
Summary	
Run Summary	
Executed Test	detailBarang
Active Users	0
Completed Users	1
Total Users	1
Elapsed Time [H:M:S]	0:00:52
Run Status	Complete
Displaying Results for Computer:	All Hosts
Page Summary	
Total Page Attempts [for Run]	6
Total Page Hits [for Run]	6
Total Percent Healthy [for Run]	100
Total Healthy [for Run]	6
Average Response Time For All Pages [ms] [for Run]	1,025.5
Response Time Standard Deviation For All Pages [for Run]	193.2
Maximum Response Time For All Pages [ms] [for Run]	1,202
Minimum Response Time For All Pages [ms] [for Run]	777
Page Element Summary	
Total Page Element Attempts [for Run]	11
Total Page Element Hits [for Run]	11
Average Response Time For All Page Elements [ms] [for Run]	396
Response Time Standard Deviation For All Page Elements [for Run]	134.9

Gambar 6. 69 Summary Kasus Uji PT_011

Gambar 6.69 merupakan rangkuman dari pengujian pada kasus uji PT_011. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

12. Kasus Uji PT_012

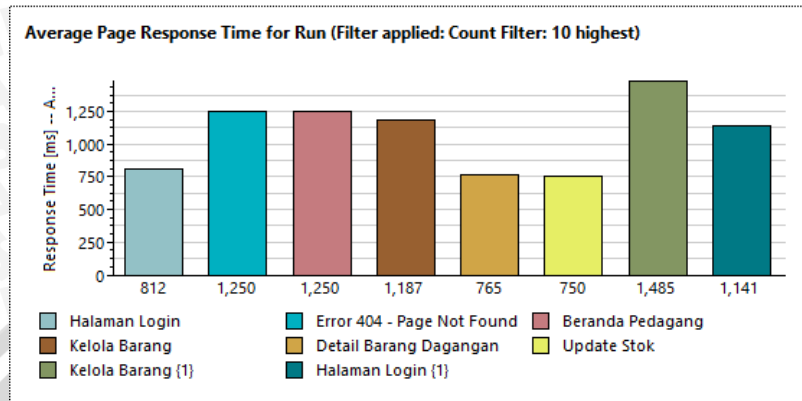
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_012 dari fitur edit data barang yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 70 Overall Kasus Uji PT_012

Gambar 6.70 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_012. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

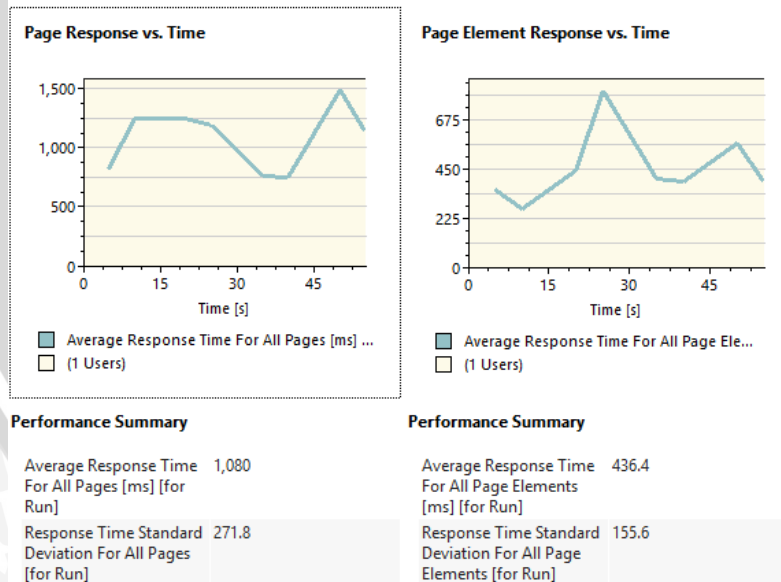
Page Performance



Gambar 6. 71 Page Performance Kasus Uji PT_012

Pada gambar 6.71 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur edit data barang. Dari gambar 6.60 dapat diketahui bahwa untuk membuka halamanlogin membutuhkan waktu 812ms, beranda pedagang 1,250, kelola barang 1,187, detail barang dagangan 756ms, update stok 750ms, kelola barang setelah edit data barang 1,485 dan halaman login setelah logout 1,141ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

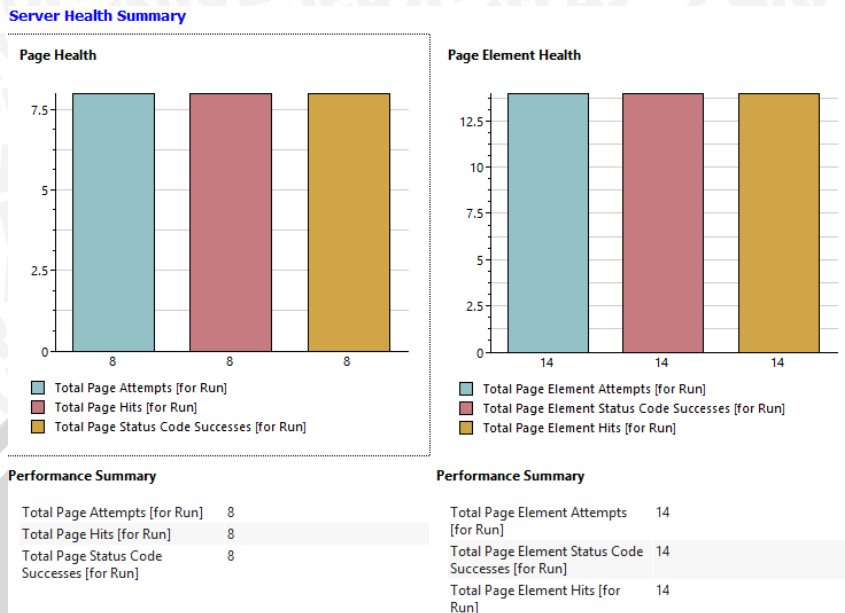
Response vs. Time Summary



Gambar 6. 72 Response VS Time Summary Kasus Uji PT_012

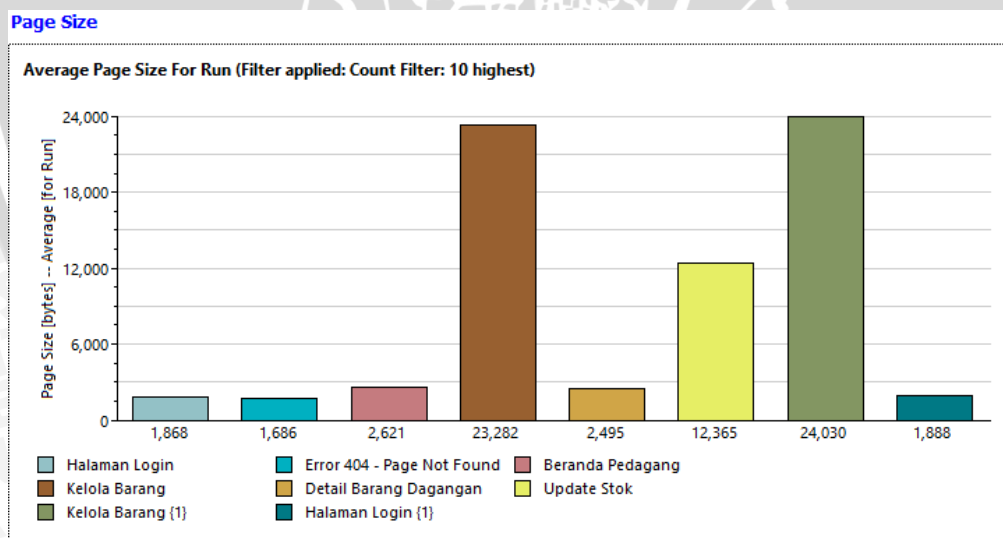
Pada gambar 6.72 ditunjukkan *average response time for all page* 1,080ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 271.8ms. Sedangkan *Average*

response time for all page elements 436ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 155.6ms.



Gambar 6. 73 Server Health Summary Kasus Uji PT_012

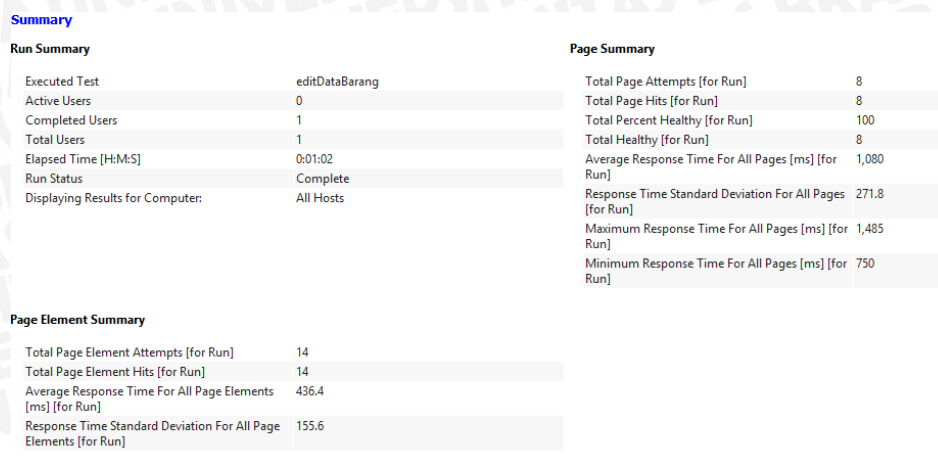
Gambar 6.73 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.62 bahwa total page yang dijalankan sebanyak 8 halaman dan total elemen yang dijalankan ada 14 elemen.



Gambar 6. 74 Page Size Kasus Uji PT_012

Pada gambar 6.74 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Halaman kelola barang memiliki ukuran yang cukup besar yaitu 23,621bytes dan 24,030byte hal ini dikarenakan pada halaman kelola barang memanggil banyak data dari database base sehingga membuat ukuran

halaman lebih besar. Sedangkan untuk halaman lain memiliki ukuran yang bervariasi antara 1,868byte sampai 2,621bytes.

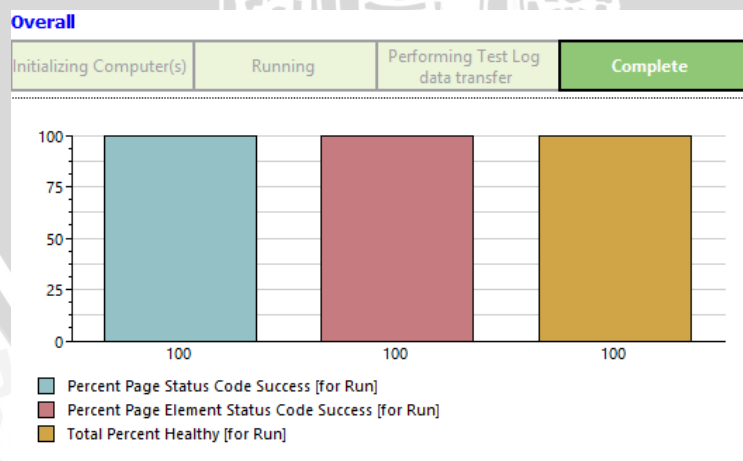


Gambar 6. 75 Summary Kasus Uji PT_012

Gambar 6.75 merupakan rangkuman dari pengujian pada kasus uji PT_012. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

13. Kasus Uji PT_013

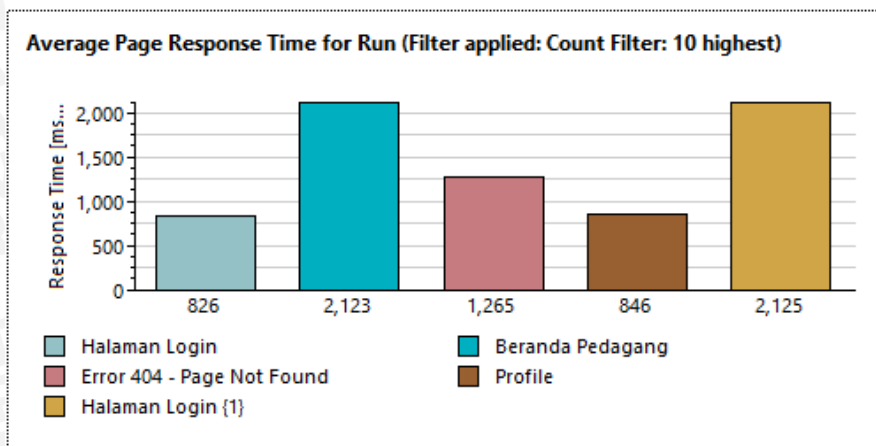
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_013 dari fitur profil pedagang yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 76 Overall Kasus Uji PT_013

Gambar 6.76 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_013. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

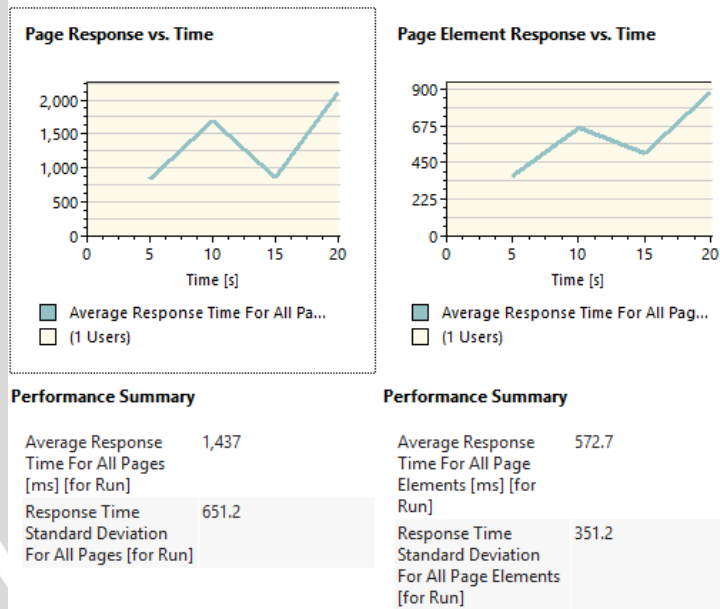
Page Performance



Gambar 6. 77 Page Performance Kasus Uji PT_013

Pada gambar 6.77 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola barang dagangan. Dari gambar 6.77 dapat diketahui bahwa untuk memanggil halaman login membutuhkan waktu 826ms, beranda pedagang 2,123ms, profile 846ms dan halaman login setelah logout 2,125ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

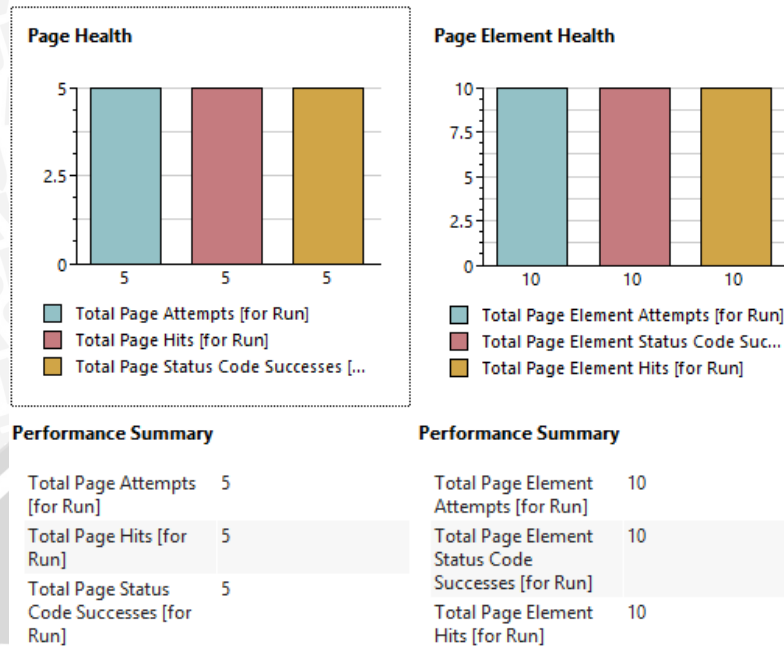
Response vs. Time Summary



Gambar 6. 78 Response VS Time Summary Kasus Uji PT_013

Pada gambar 6.78 ditunjukkan *average response time for all page* 1,437ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 651.2ms. Sedangkan *Average response time for all page elements* 572.7ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 351.2ms.

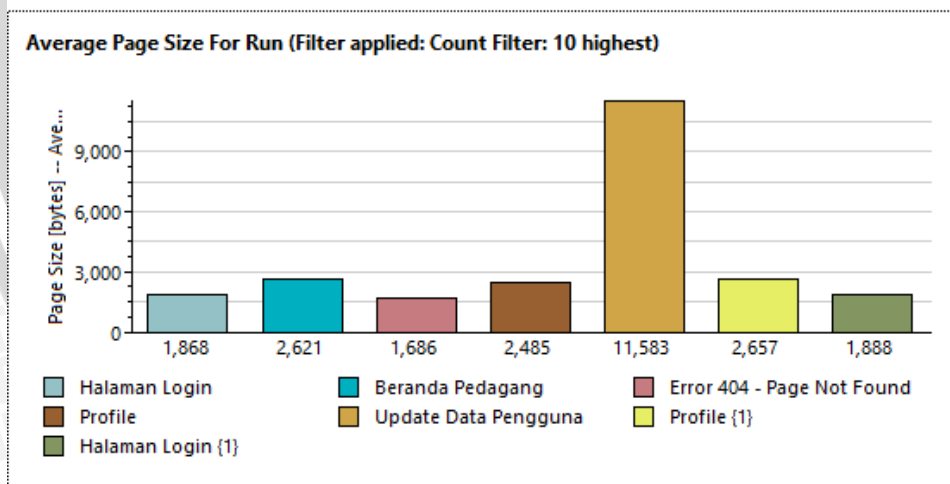
Server Health Summary



Gambar 6. 79 Server Health Summary Kasus Uji PT_013

Gambar 6.79 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.79 bahwa total page yang dijalankan sebanyak 5 halaman dan total elemen yang dijalankan ada 10 elemen.

Page Size



Gambar 6. 80 page size kasus uji PT_013

Pada gambar 6.80 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Pada halaman update data pengguna memiliki ukuran yang besar yaitu 11,583byte dikarenakan pada halaman ini selain menampilkan data juga menampilkan form untuk merubah data pedagang. Selain itu halaman lain memiliki ukuran yang bervariasi antara 1,868byte sampai 2,657byte.

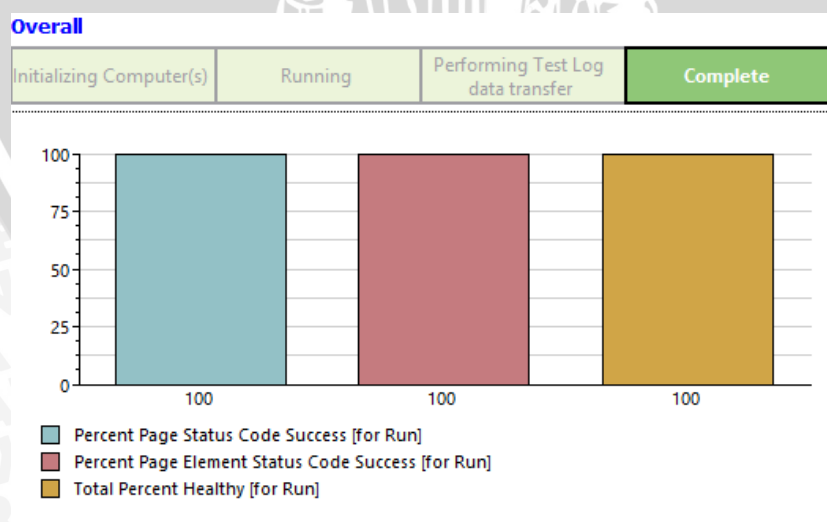
Summary				
Run Summary			Page Summary	
Executed Test	profilPedagang		Total Page Attempts [for Run]	5
Active Users	0		Total Page Hits [for Run]	5
Completed Users	1		Total Percent Healthy [for Run]	100
Total Users	1		Total Healthy [for Run]	5
Elapsed Time [H:M:S]	0:00:22		Average Response Time For All Pages [ms] [for Run]	1,437
Run Status	Complete		Response Time Standard Deviation For All Pages [for Run]	651.2
Displaying Results for Computer:	All Hosts		Maximum Response Time For All Pages [ms] [for Run]	2,125
			Minimum Response Time For All Pages [ms] [for Run]	826
Page Element Summary				
Total Page Element Attempts [for Run]	10			
Total Page Element Hits [for Run]	10			
Average Response Time For All Page Elements [ms] [for Run]	572.7			
Response Time Standard Deviation For All Page Elements [for Run]	351.2			

Gambar 6. 81 Summary Kasus Uji PT_013

Gambar 6.81 merupakan rangkuman dari pengujian pada kasus uji PT_013. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

14. Kasus Uji PT_014

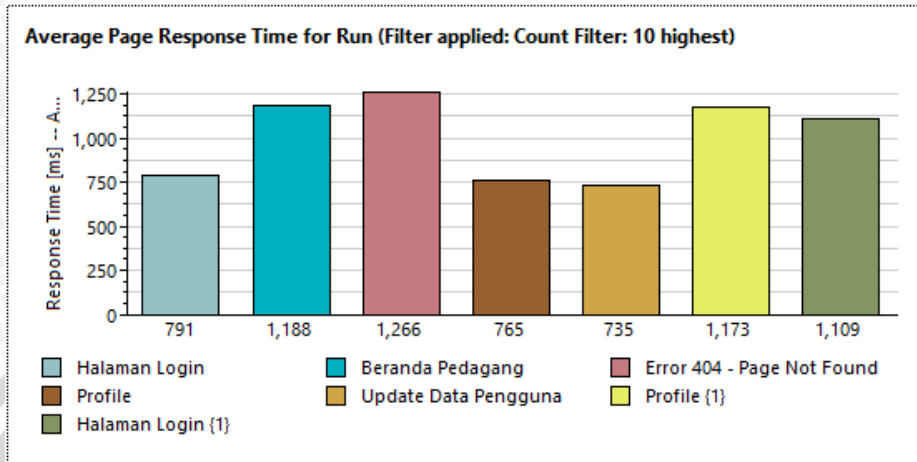
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_014 dari fitur kelola barang dagangan yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 82 Overall Kasus Uji PT_014

Gambar 6.82 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_014. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

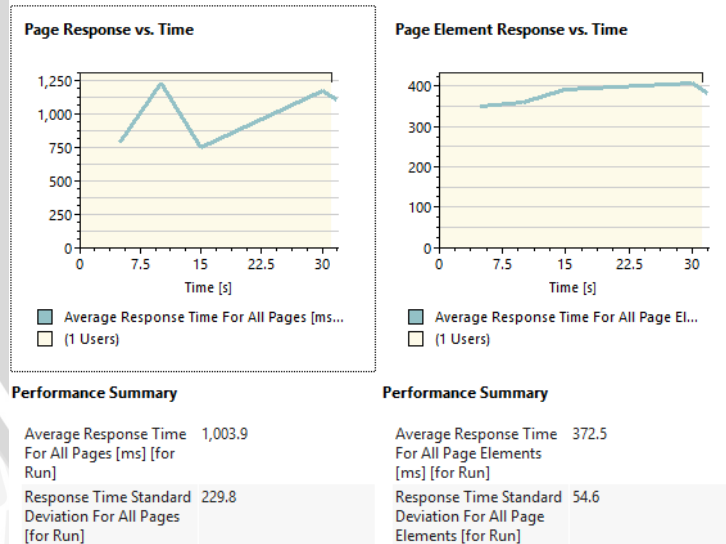
Page Performance



Gambar 6. 83 Page Performance Kasus Uji PT_014

Pada gambar 6.83 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur edit data pedagang. Dari gambar 6.71 dapat diketahui bahwa untuk membuka halaman login membutuhkan waktu 791ms, beranda pedagang 1,266, profile 765ms, update data 735, profile setelah rubah data 1,173 dan halaman login setelah logout 1,109. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

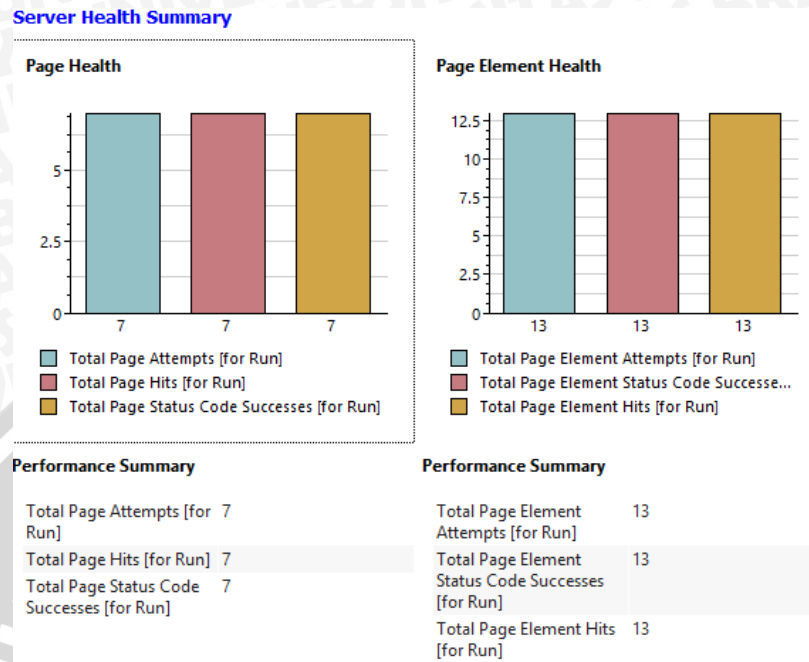
Response vs. Time Summary



Gambar 6. 84 Response VS Time Summary Kasus Uji PT_014

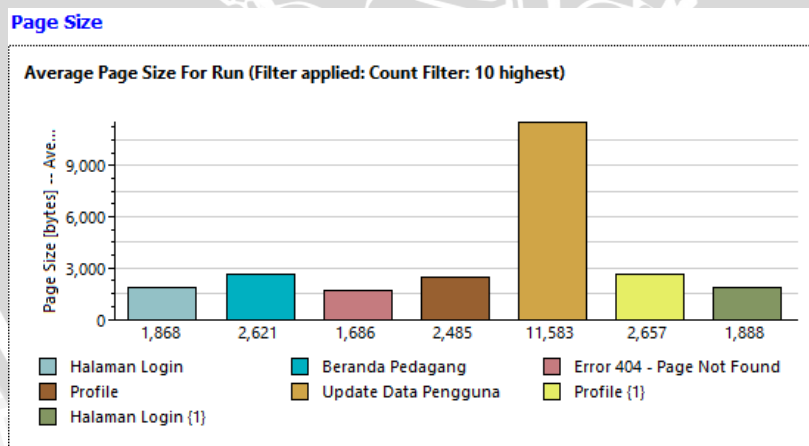
Pada gambar 6.84 ditunjukkan *average response time for all page* 1,003.9ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 229.8ms. Sedangkan *Average response time for all page elements* 372.5ms yang dihasilkan dari menghitung

rata-rata respon elemen halaman dan juga menunjukkan standard deviation 54.6ms.



Gambar 6. 85 Server Health Summary Kasus Uji PT_014

Gambar 6.85 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.72 bahwa total page yang dijalankan sebanyak 7 halaman dan total elemen yang dijalankan ada 13 elemen.



Gambar 6. 86 Page size kasus uji PT_014

Pada gambar 6.86 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Pada halaman update data pengguna memiliki ukuran yang besar yaitu 11,583byte dikarenakan pada halaman ini selain menampilkan data juga menampilkan form untuk merubah data pedagang. Selain itu halaman lain memiliki ukuran yang bervariasi antara 1,868byte sampai 2,657byte.

Summary

Run Summary

Executed Test	editDataPedagang
Active Users	0
Completed Users	1
Total Users	1
Elapsed Time [H:M:S]	0:00:32
Run Status	Complete
Displaying Results for Computer:	All Hosts

Page Summary

Total Page Attempts [for Run]	7
Total Page Hits [for Run]	7
Total Percent Healthy [for Run]	100
Total Healthy [for Run]	7
Average Response Time For All Pages [ms] [for Run]	1,003.9
Response Time Standard Deviation For All Pages [for Run]	229.8
Maximum Response Time For All Pages [ms] [for Run]	1,266
Minimum Response Time For All Pages [ms] [for Run]	735

Page Element Summary

Total Page Element Attempts [for Run]	13
Total Page Element Hits [for Run]	13
Average Response Time For All Page Elements [ms] [for Run]	372.5
Response Time Standard Deviation For All Page Elements [for Run]	54.6

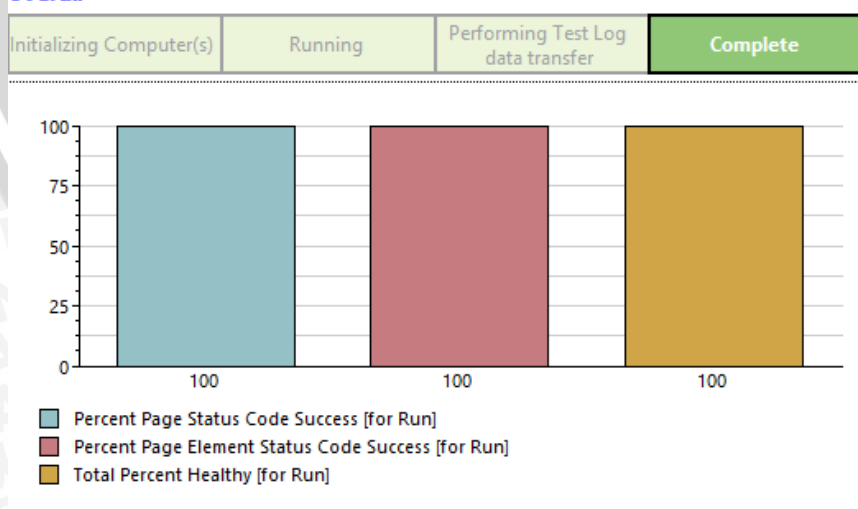
Gambar 6. 87 Summary Kasus Uji PT_014

Gambar 6.87 merupakan rangkuman dari pengujian pada kasus uji PT_014. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

15. Kasus Uji PT_015

Pada bagian ini menjelaskan hasil pengujian kasus uji PT_015 dari fitur pemesanan barang. Pemesanan barang merupakan gabungan dari fitur pilih pedagang, pilih barang keranjang belanja dan memesan pesanan. Selanjutnya yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.

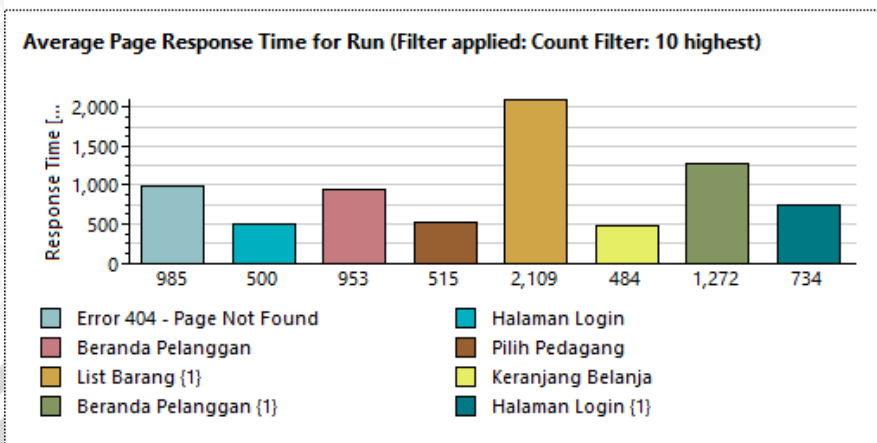
Overall



Gambar 6. 88 Overall Kasus Uji PT_015

Gambar 6.88 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_015. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

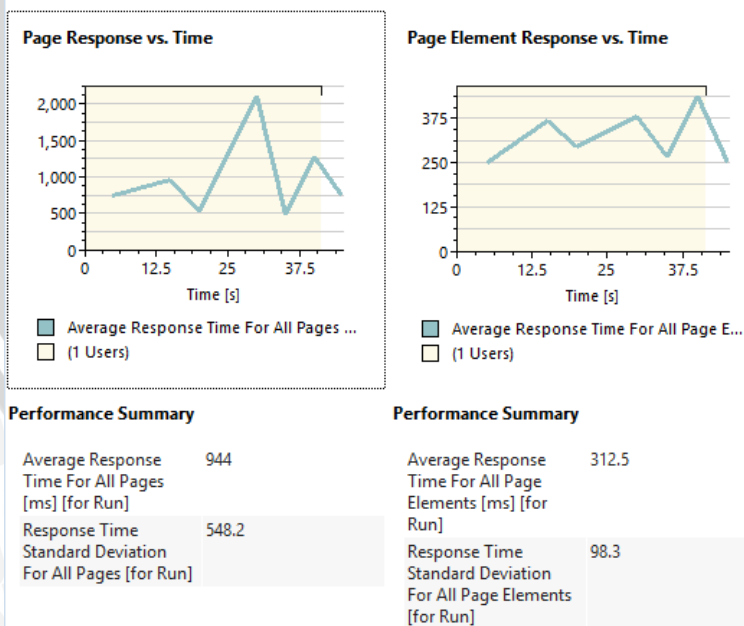
Page Performance



Gambar 6. 89 Page Performance Kasus Uji PT_015

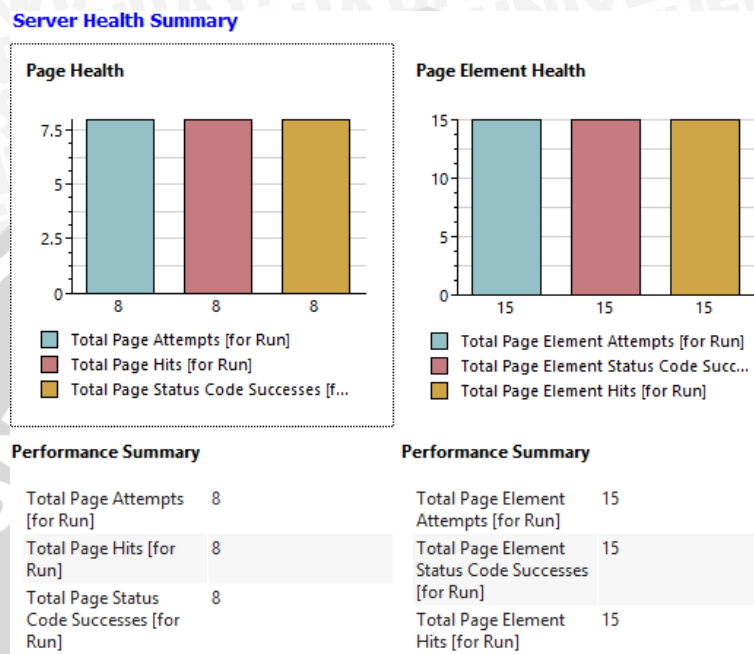
Pada gambar 6.89 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola barang dagangan. Dari gambar 6.89 dapat diketahui bahwa untuk membuka halaman login 500ms, beranda pedagang 953ms, pilih pedagang 515, list barang 2,109ms, keranjang belanja 484ms, beranda pelanggan setelah proses pesan 1,272ms dan halaman login setelah logout 734ms. halaman list barang memiliki waktu yang lama dikarenakan perlu memanggil banyak data dari database untuk ditampilkan. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

Response vs. Time Summary



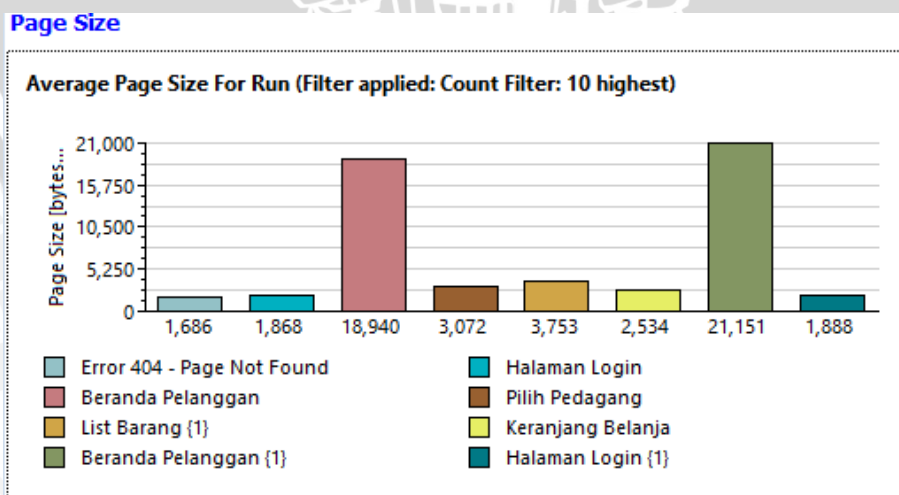
Gambar 6. 90 VS Time Summary Kasus Uji PT_015

Pada gambar 6.90 ditunjukkan *average response time for all page* 944ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 548.2ms. Sedangkan *Average response time for all page elements* 312.5ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 98.3ms.



Gambar 6. 91 Server Health Summary Kasus Uji PT_015

Gambar 6.91 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.91 bahwa total page yang dijalankan sebanyak 8 halaman dan total elemen yang dijalankan ada 15 elemen.



Gambar 6. 92 page size kasus uji PT_015

Pada gambar 6.92 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Pada halaman beranda memiliki ukuran yang besar dikarenakan pada beranda menampilkan data pesanan dari waktu ke waktu sehingga perlu menampilkan banyak data. Sedangkan halaman lain mempunyai besar yang bervariasi dan tidak begitu besar.

Summary

Run Summary

Executed Test	order
Active Users	0
Completed Users	1
Total Users	1
Elapsed Time [H:M:S]	0:00:52
Run Status	Complete
Displaying Results for Computer:	All Hosts

Page Summary

Total Page Attempts [for Run]	8
Total Page Hits [for Run]	8
Total Percent Healthy [for Run]	100
Total Healthy [for Run]	8
Average Response Time For All Pages [ms] [for Run]	944
Response Time Standard Deviation For All Pages [for Run]	548.2
Maximum Response Time For All Pages [ms] [for Run]	2,109
Minimum Response Time For All Pages [ms] [for Run]	484

Page Element Summary

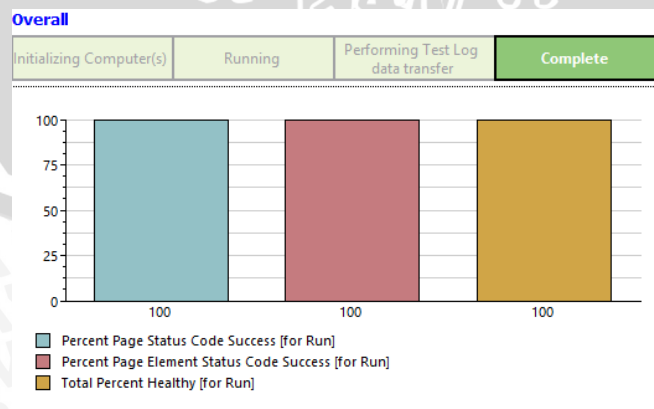
Total Page Element Attempts [for Run]	15
Total Page Element Hits [for Run]	15
Average Response Time For All Page Elements [ms] [for Run]	312.5
Response Time Standard Deviation For All Page Elements [for Run]	98.3

Gambar 6. 93 Summary Kasus Uji PT_015

Gambar 6.93 merupakan rangkuman dari pengujian pada kasus uji PT_015. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

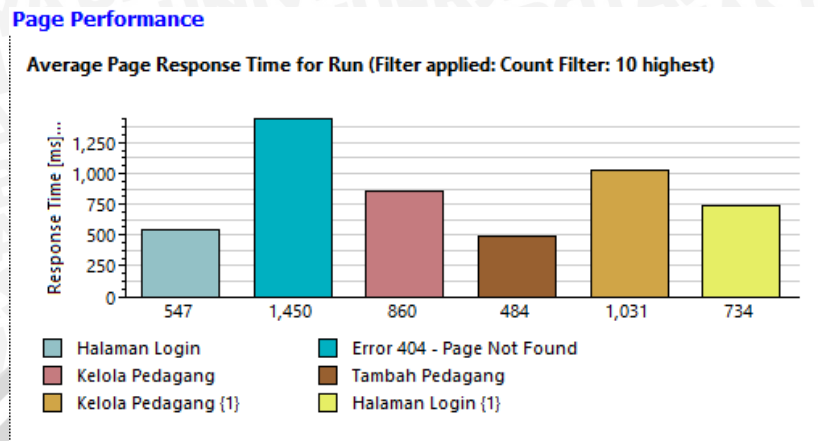
16. Kasus Uji PT_016

Pada bagian ini menjelaskan hasil pengujian kasus uji PT_016 dari fitur tambah pedagang. Tambah Pedagang merupakan fungsi pada admin untuk dapat menambah pedagang. Selanjutnya akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



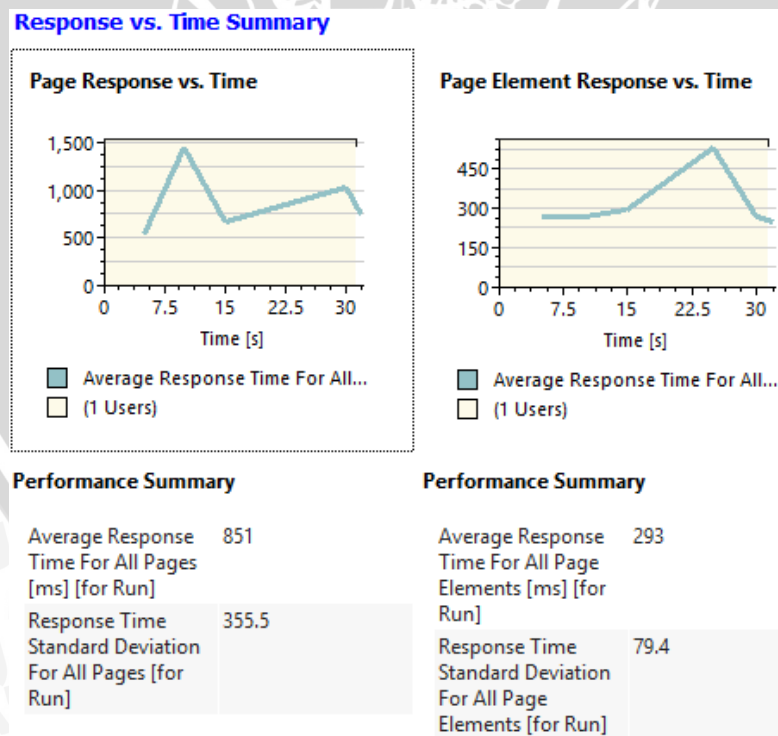
Gambar 6. 94 Overall Kasus Uji PT_016

Gambar 6.94 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_015. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.



Gambar 6. 95 Page Performance Kasus Uji PT_016

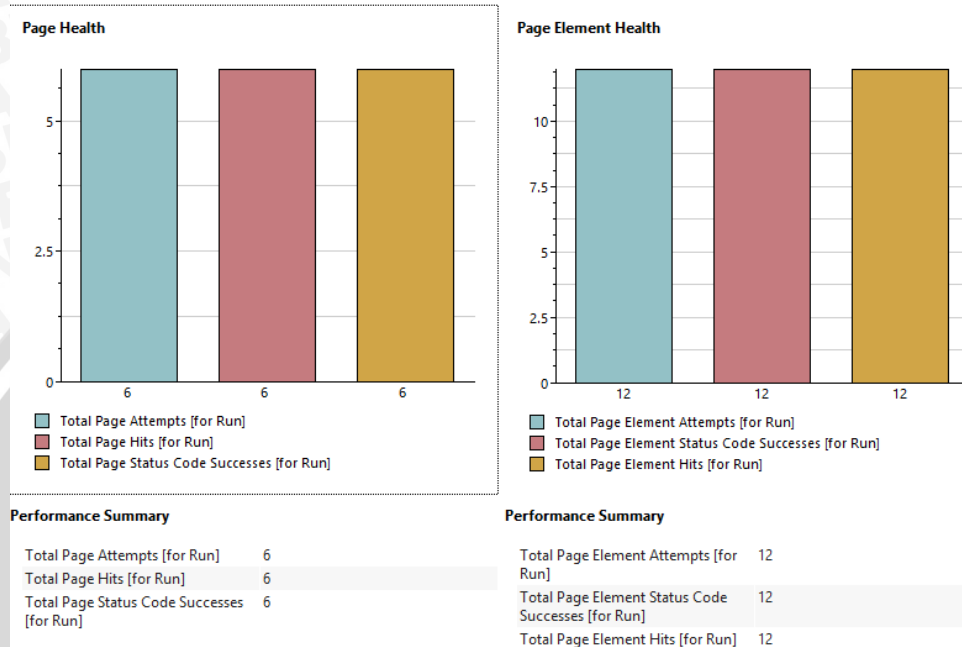
Pada gambar 6.95 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola barang dagangan. Dari gambar 6.95 dapat diketahui bahwa untuk membuka halaman login 547ms, halaman kelola pedagang 860ms, tambah pedagang 484ms, halaman kelola pedagang setelah proses tambah pedagang 1,031 dan halaman login setelah logout 734ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.



Gambar 6. 96 Response VS Time Summary Kasus Uji PT_016

Pada gambar 6.96 ditunjukkan *average response time for all page* 851ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga menunjukkan *response time standard deviation* 255.5ms. Sedangkan *Average response time for all page elements* 293ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan standard deviation 79.4ms.

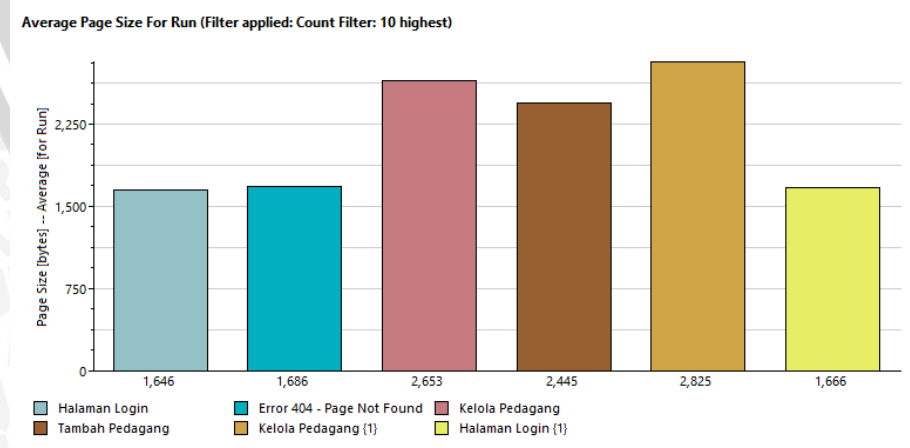
Server Health Summary



Gambar 6. 97 Server Health Summary Kasus Uji PT_016

Gambar 6.97 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.97 bahwa total page yang dijalankan sebanyak 6 halaman dan total elemen yang dijalankan ada 12 elemen.

Page Size



Gambar 6. 98 page size kasus uji PT_016

Pada gambar 6.98 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang terdapat pada setiap halaman. Kelola pedagang memiliki ukuran yang paling besar dikarenakan harus memanggil dan menampilkan banyak data.

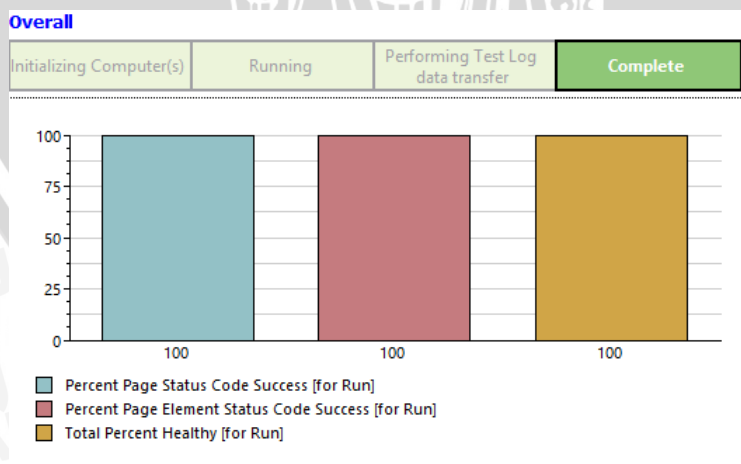
Summary			
Run Summary		Page Summary	
Executed Test	tambahPedagang7	Total Page Attempts [for Run]	6
Active Users	0	Total Page Hits [for Run]	6
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	6
Elapsed Time [H:M:S]	0:00:32	Average Response Time For All Pages [ms] [for Run]	851
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	355.5
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	1,450
		Minimum Response Time For All Pages [ms] [for Run]	484
Page Element Summary			
Total Page Element Attempts [for Run]	12		
Total Page Element Hits [for Run]	12		
Average Response Time For All Page Elements [ms] [for Run]	293		
Response Time Standard Deviation For All Page Elements [for Run]	79.4		

Gambar 6. 99 Summary Kasus Uji PT_016

Gambar 6.99 merupakan rangkuman dari pengujian pada kasus uji PT_015. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

17. Kasus Uji PT_017

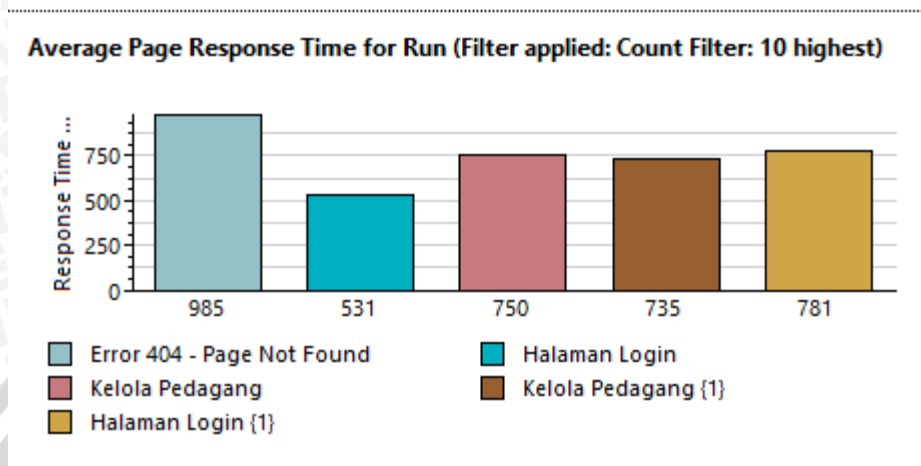
Pada bagian ini menjelaskan hasil pengujian kasus uji PT_017 dari fitur hapus pedagang. Hapus pedagang merupakan fitur dari halman admin untuk menghapus pedagang. Selanjutnya yang akan ditampilkan dalam bentuk tabel dan *screenshot* hasil pengujian. Hasil yang didapatkan pada pengujian berupa waktu respon dari tiap halaman.



Gambar 6. 100 Overall Kasus Uji PT_017

Gambar 6.100 menunjukkan persentase keberhasilan pada saat pengujian kasus uji PT_017. Dapat diketahui dari gambar bahwa pengujian yang dilakukan telah berhasil.

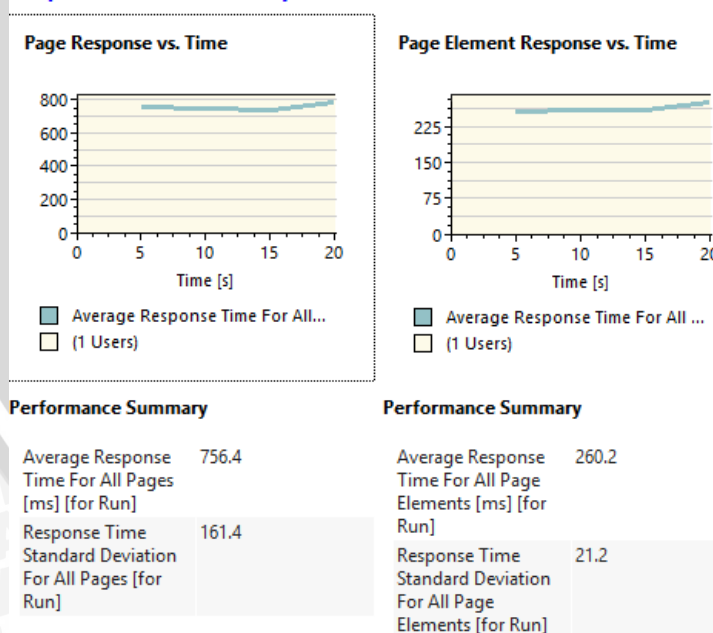
Page Performance



Gambar 6. 101 Page Performance Kasus Uji PT_017

Pada gambar 6.101 ditunjukkan waktu respon dari tiap halaman pada saat menjalankan fitur kelola barang dagangan. Dari gambar 6.101 dapat diketahui bahwa untuk membuka halaman login membutuhkan waktu 531ms, halaman kelola pedagang 750ms, halaman kelola pedagang setelah proses hapus pedagang 735ms dan halaman login setelah logout 781ms. *Page performance* merupakan bagian paling penting karena menjelaskan respon waktu setiap halaman.

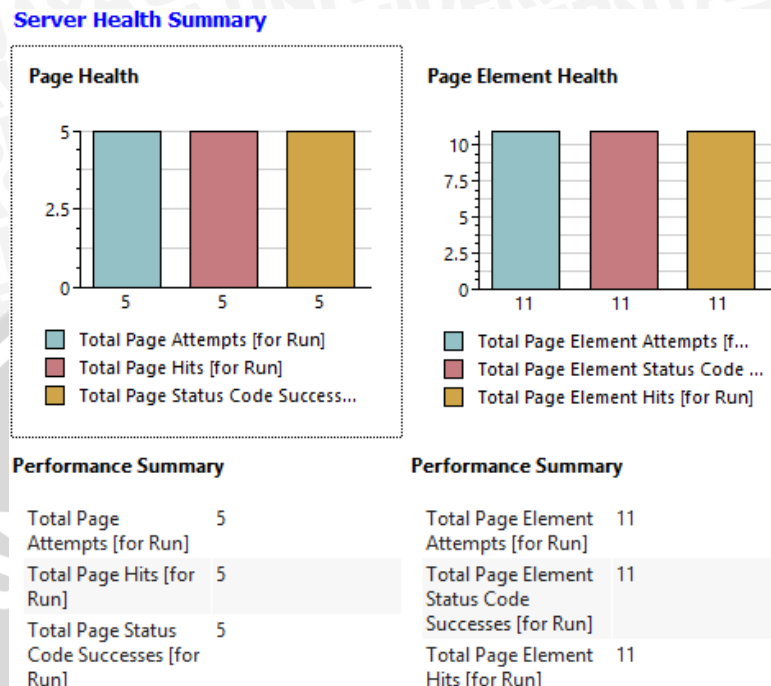
Response vs. Time Summary



Gambar 6. 102 Response VS Time Summary Kasus Uji PT_017

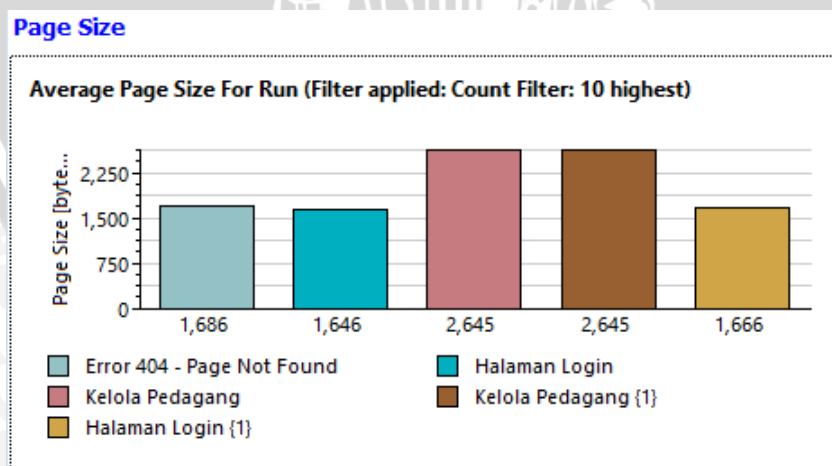
Pada gambar 6.102 ditunjukkan *average response time for all page* 756.4ms yang didapatkan dari menghitung respon rata-rata semua halaman dan juga

menunjukkan *response time standard deviation* 161ms. Sedangkan *Average response time for all page elements* 260.2ms yang dihasilkan dari menghitung rata-rata respon elemen halaman dan juga menunjukkan *standard deviation* 21.2ms.



Gambar 6. 103 Server Health Summary Kasus Uji PT_017

Gambar 6.103 menjelaskan tentang banyaknya *page element* dan *page* yang dijalankan pada saat pengujian. Dapat diketahui dari gambar 6.103 bahwa total page yang dijalankan sebanyak 5 halaman dan total elemen yang dijalankan ada 11 elemen.



Gambar 6. 104 page size kasus uji PT_017

Pada gambar 6.104 menjelaskan ukuran data yang diminta pada server dari setiap halaman berdasarkan banyaknya data yang dipanggil dan elemen yang

terdapat pada setiap halaman. kelola pedagang memiliki ukuran yang paling besar dikarenakan harus memanggil dan menampilkan banyak data.

Summary			
Run Summary		Page Summary	
Executed Test	order	Total Page Attempts [for Run]	8
Active Users	0	Total Page Hits [for Run]	8
Completed Users	1	Total Percent Healthy [for Run]	100
Total Users	1	Total Healthy [for Run]	8
Elapsed Time [H:M:S]	0:00:52	Average Response Time For All Pages [ms] [for Run]	944
Run Status	Complete	Response Time Standard Deviation For All Pages [for Run]	548.2
Displaying Results for Computer:	All Hosts	Maximum Response Time For All Pages [ms] [for Run]	2,109
		Minimum Response Time For All Pages [ms] [for Run]	484
Page Element Summary			
Total Page Element Attempts [for Run]	15		
Total Page Element Hits [for Run]	15		
Average Response Time For All Page Elements [ms] [for Run]	312.5		
Response Time Standard Deviation For All Page Elements [for Run]	98.3		

Gambar 6. 105 Summary Kasus Uji PT_017

Gambar 6.105 merupakan rangkuman dari pengujian pada kasus uji PT_017. Pada bagian *run summary* menjelaskan banyak *user* yang menjalankan pengujian dan waktu yang ditempuh pada saat pengujian. *Page summary* menampilkan jumlah halaman, persentase kesuksesan dan juga respon waktu semua halaman. Sedangkan *page element summary* menampilkan total elemen halaman dan rata-rata respon elemen semua halaman.

6.5 Analisa Hasil Pengujian

6.5.1 Analisa pengujian *White-Box*

Berdasarkan hasil pengujian *White-Box* yang telah dilakukan pada kasus uji menampilkan detail pesanan pelanggan, konfirmasi pesanan pelanggan dan menampilkan list barang dagangan pedagang dapat disimpulkan bahwa kasus uji menampilkan detail pesanan pelanggan dan konfirmasi pesanan pelanggan memiliki nilai *cyclomatic complexity* sama, yaitu 5 sedangkan kasus uji menampilkan list barang dagangan pedagang memiliki nilai *cyclomatic complexity* 4. Hasil pengujian akan ditampilkan dalam tabel 6.40.

Tabel 6.40 analisa hasil pengujian *white-box*

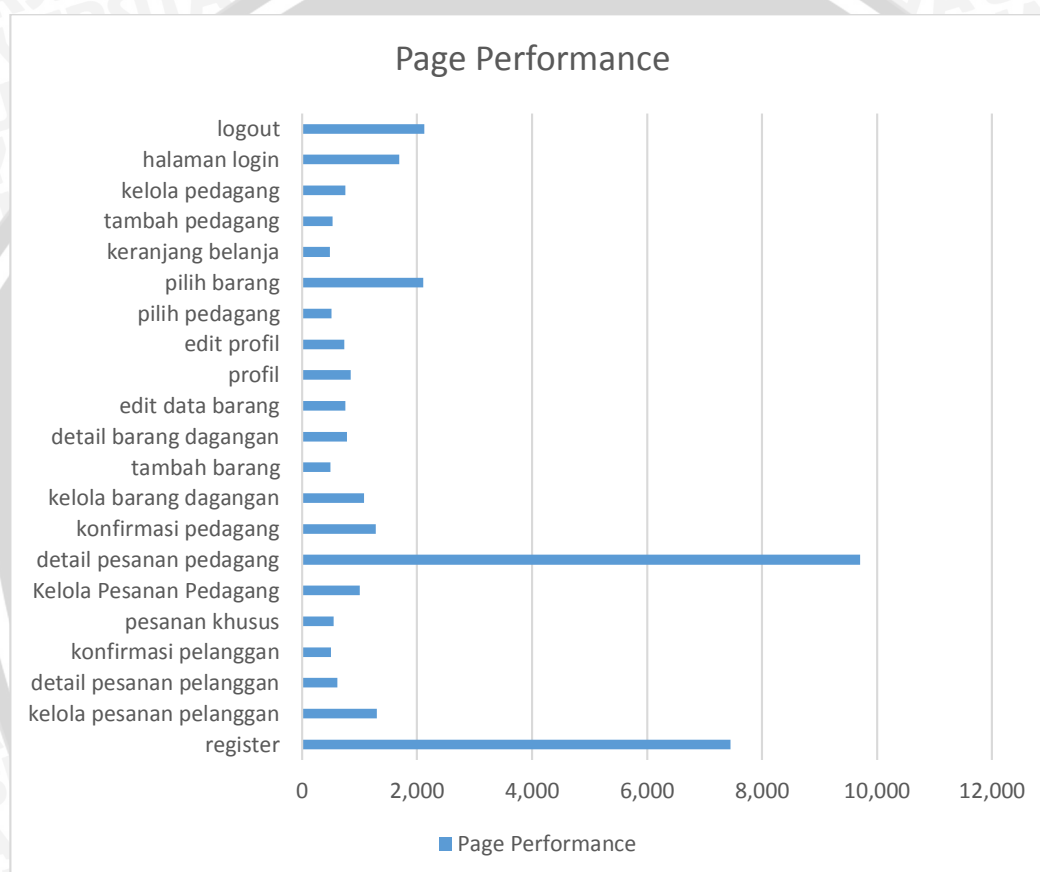
Kasus uji	Total Regions	Total Independent Path	Node	Edge	Percabangan
Menampilkan detail pesanan pelanggan	5	5	17	20	4
Konfirmasi pesanan pelanggan	5	5	26	29	4

Menampilkan list barang dagangan	4	4	16	18	4
----------------------------------	---	---	----	----	---

6.5.2 Analisa hasil pengujian *Black-Box*

Berdasarkan kasus uji untuk pengujian *blackbox* yang telah didefinisikan sebelumnya dapat disimpulkan bahwa seluruh spesifikasi kebutuhan perangkat lunak telah memenuhi kebutuhan yang didefinisikan sebelumnya karena seluruh hasil pengujian *blackbox* menghasilkan hasil yang *valid*.

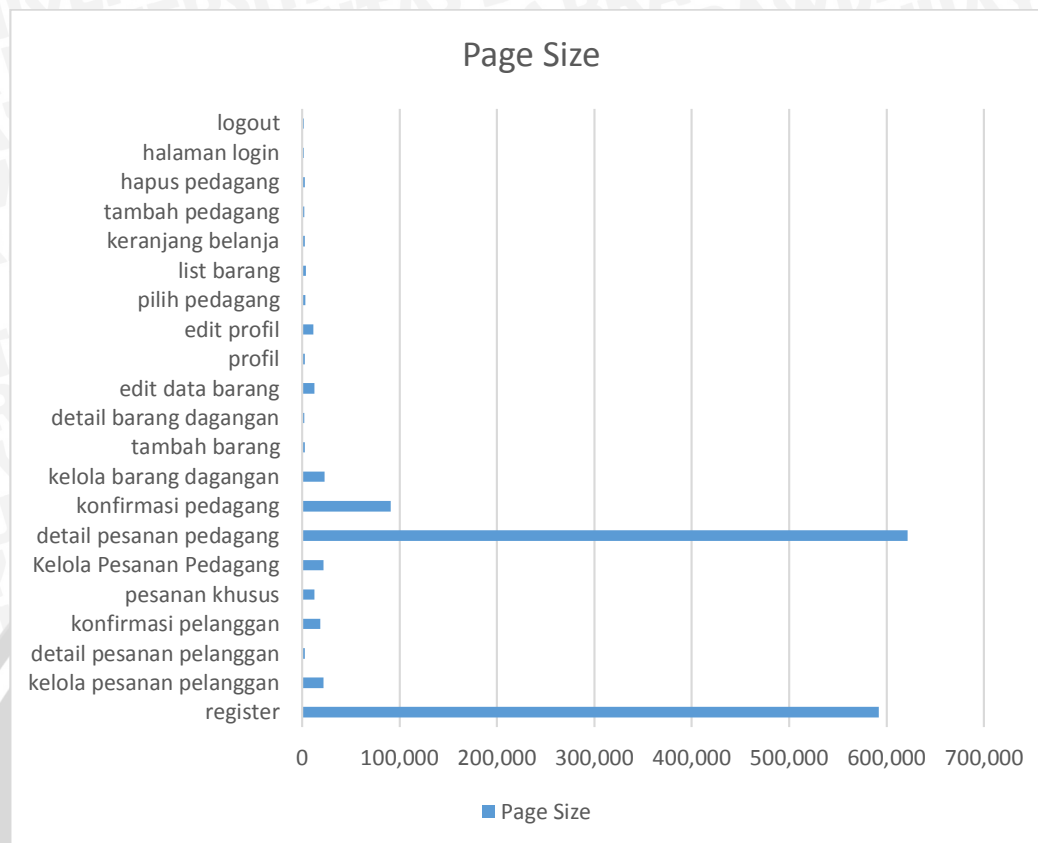
6.5.3 Analisa hasil pengujian performa



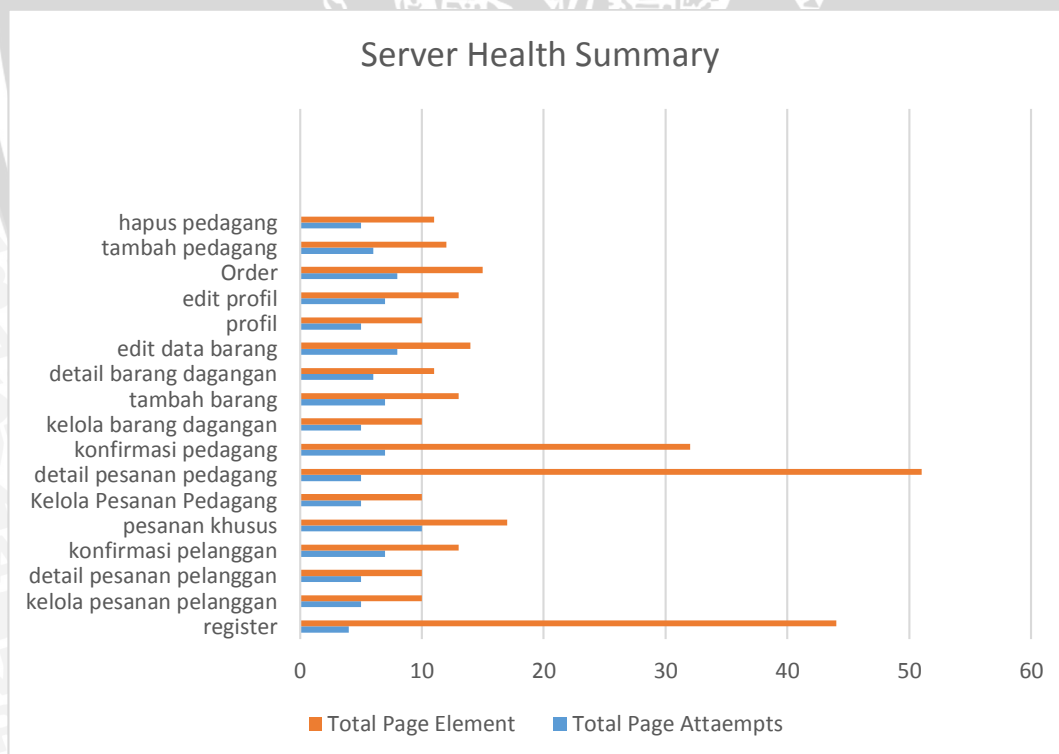
Gambar 6. 106 Hasil Uji *Page Performance*

Gambar 6.106 ditunjukkan respon waktu dari semua halaman yang diambil dari semua pengujian. Pada gambar 6.106 waktu respon paling lama ada pada detail pesanan dan register, hal ini dikarenakan pada 2 halaman ini mempunyai ukuran halaman (*page size*) dan *element* yang banyak sehingga membuat halaman ini membutuhkan waktu untuk menampilkan isi halaman.

Gambar 6.107 menunjukkan ukuran setiap halaman dalam byte. Pada gambar 6.107 halaman halaman register dan detail pesanan memiliki ukuran yang besar dikarenakan pada 2 halaman tersebut memiliki jumlah element dan jumlah pemanggaan data yang besar, sehingga membutuhkan waktu yang cukup lama untuk menampilkan halaman tersebut.



Gambar 6. 107 Hasil Uji Page size



Gambar 6. 108 Hasil Uji Server Health Summary

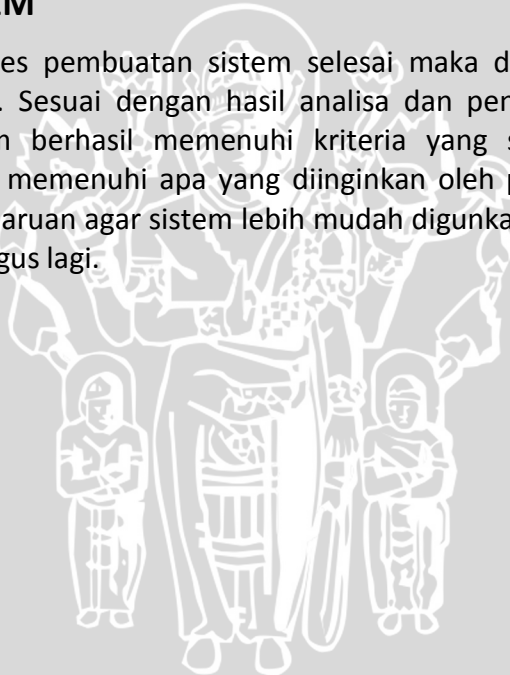
Gambar 6.108 menunjukkan rangkuman jumlah halaman yang dipanggil dan jumlah elemen yang dipanggil pada setiap halaman. Pada gambar 6.108 halaman detail pesanan pesanan pedagang dan halaman register yang memiliki jumlah elment paling banyak

Berdasarkan hasil pengujian yang didapatkan sistem telah berhasil dan siap digunakan. Dari sumber yang diadapat pada jurnal dijelaskan bahwa respon waktu setiap halaman setidaknya kurang dari 8 sampai 10 detik, sedangkan dari hasil pengujian menggunakan *Rational Performance Tester* didapatkan waktu respon dari setiap halaman tidak lebih dari 10,00ms (1000ms sama dengan 1 detik).

Dari hasil pengujian performa yang telah dilakukan dapat disimpulkan bahwa pengujian telah selesai dan berhasil. Pengujian dilakukan dengan menjalankan setiap fitur yang telah didefinisikan pada kasus uji pengujian performa dengan 1 user. Hasil respon waktu tiap halaman bervariasi bergantung pada elemen yang ada pada setiap halaman dan koneksi internet yang tersedia.

6.6 EVALUASI SISTEM

Setelah semua proses pembuatan sistem selesai maka dilanjutkan dengan proses evaluasi sistem. Sesuai dengan hasil analisa dan pengujian yang telah dilakukan maka sistem berhasil memenuhi kriteria yang sebelumnya telah didefinisikan dan telah memenuhi apa yang diinginkan oleh pengguna. Namun perlu diadakan pembaharuan agar sistem lebih mudah digunkan dan mempunyai performa yang lebih bagus lagi.



BAB 7 KESIMPULAN DAN SARAN

7.1 Kesimpulan

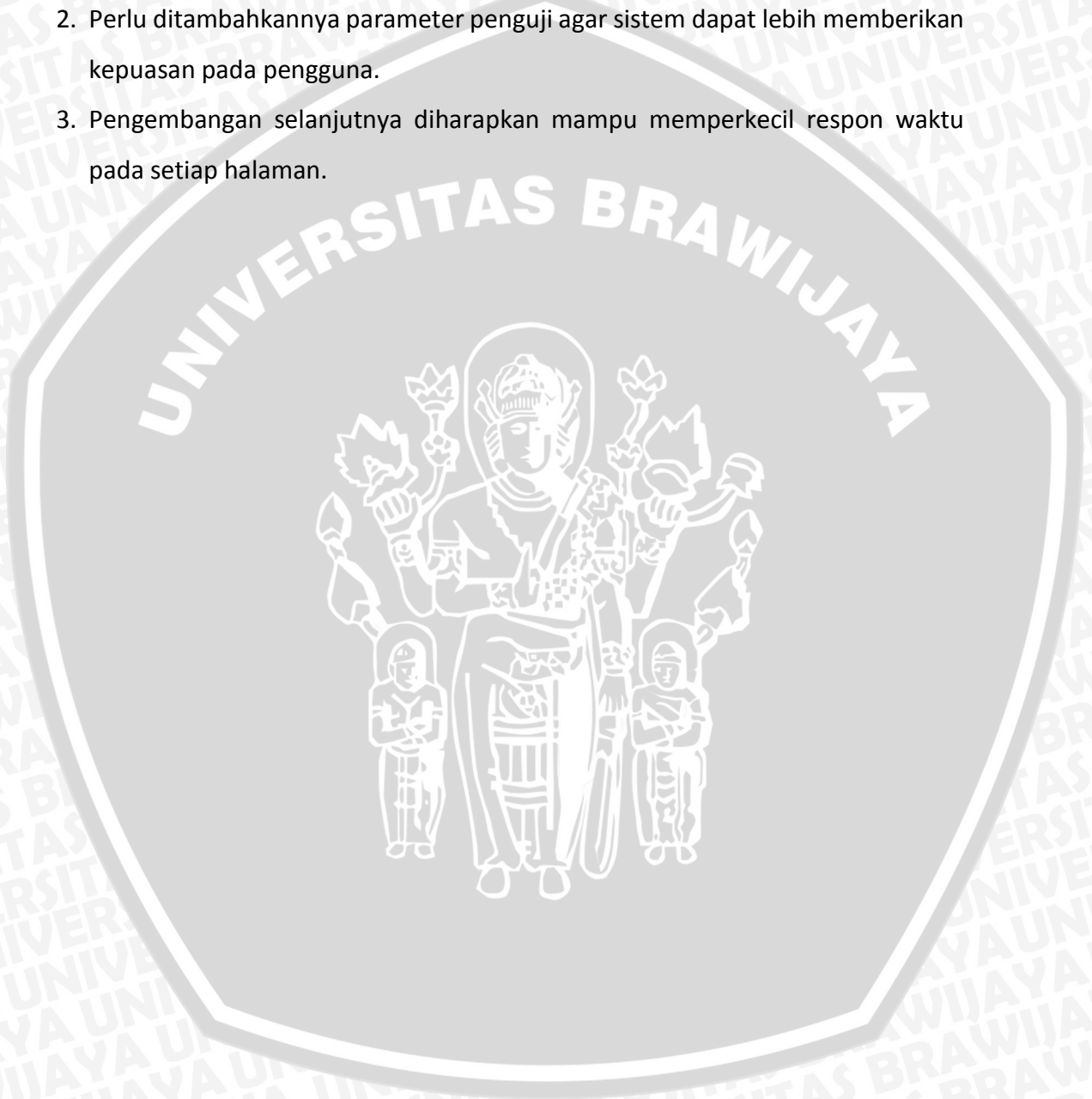
Berdasarkan hasil analisa perancangan, implementasi dan pengujian yang telah dilakukan, maka dapat diambil kesimpulan sebagai berikut:

1. Perancangan dan pengimplementasian aplikasi pemesanan sayur keliling menjadi aplikasi berbasis web telah sesuai dengan spesifikasi kebutuhan dan perancangan.
2. Aplikasi pada mobile dapat mengirimkan notifikasi pemesanan ketika ada pesanan baru dan konfirmasi pemesanan.
3. Aplikasi mampu menghitung laba yang dihasilkan dari penjualan yang dihasilkan setelah konfirmasi dari pelanggan dilakukan.
4. Berdasarkan jurnal dengan judul "*web application performance testing-a case study of an on-line learning application*" dijelaskan bahwa pengguna tidak dapat mentoleransi delay waktu yang melebihi 10 detik karena dapat mengganggu aktivitas pengguna. Sedangkan dari hasil pengujian performa dapat diketahui bahwa respon waktu semua halaman tidak lebih dari 10,000*milisecond* (1,000ms sama dengan 1s).
5. *Page not found* yang terdapat pada setiap pengujian terjadi karena aplikasi penguji (Rational Performance Tester) tidak dapat membaca *favicon*(icon kecil yang dibuat secara khusus yang muncul ketika suatu website telah selesai loading) yang ada pada aplikasi pemesanan sayur keliling ini. Meskipun terjadi kesalahan namun hal ini tidak mempengaruhi jalannya aplikasi dan aplikasi tetap dapat berjalan sebagaimana mestinya.
6. Waktu respon pada setiap halaman bervariasi bergantung pada besar halaman dalam byte yang didapatkan dari jumlah elemen yang dipanggil serta jaringan atau koneksi pada saat mengakses halaman tersebut.

9.2 Saran

Saran yang dapat diberikan untuk pengembangan aplikasi pemesanan sayur keliling selanjutnya, antara lain:

1. Perlu pengujian performa yang lebih mendalam dengan menggunakan metode lain (load Testing, Stress Testing, Capacity testing dll.)
2. Perlu ditambahkannya parameter pengujian agar sistem dapat lebih memberikan kepuasan pada pengguna.
3. Pengembangan selanjutnya diharapkan mampu memperkecil respon waktu pada setiap halaman.



DAFTAR PUSTAKA

- Automation Consultant Team, 2015. Rational Performance Tester. <<http://www.automation-consultants.com/index.php/products/ibm-products/rational-performance-tester>>. Diakses pada 18 Oktober 2015.
- Blanco, Jose A. Upton, David., 2009. Codeigniter 1.7. Packt Publishing. <<https://books.google.co.id/>> Birmingham.
- DailySocial, 2015. GfK: Pertumbuhan Penjualan Smartphone di Indonesia Tertinggi di Kawasan Asia Tenggara. <<https://dailysocial.id/post/gfk-pertumbuhan-penjualan-smartphone-di-indonesia-tertinggi-di-kawasan-asia-tenggara>>. [Diakses 09 September 2015].
- Habib, Adien Faishol. 2013. *Rancang Bangun Aplikasi Perangkat Bergerak Penanda Dini Bahaya Banjir Menggunakan Web Service Berbasis SOAP*. Malang: Teknik Informatika PTIHK, Universitas Brawijaya.
- Hartono, fajar fani. Hendry. Ramos somya., 2013. Aplikasi Reservasi Tiket Bus pada Handphone Android Menggunakan Web Service (Studi Kasus: PO. Rosalia Indah). Fakultas teknologi informasi. Universitas Kristen stya wacana. Salatiga.
- IBM-Team, 2008. Using Rational Performance Tester Version 7. Redbooks IBM.
- Ibrahim, Ali., 2011. Perancangan System Pemesanan Tiket Pesawat Berbasis Web. Fakultas Ilmu Komputer, Uversitas Sriwijaya. Palembang.
- InovasiPintar, 2015. Perkembangan Internet di Indonesia 2006-2015. <<http://inovasipintar.com/perkembangan-internet-di-indonesia-2006-2015/>>. [Diakses 20 November 2015].
- Gouzali, Saydam., 1996. Manajemen Sumber Daya Manusia (Human Resources Management) Suatu Pendekatan Mikro (Dalam Tanya Jawab). Djambatan. Jakarta.
- Griffiths, Adam., 2010. Codeigniter 1.7 Professional Development. Pakct Publishing. <<https://books.google.co.id/>> Birmingham.
- Malik, Harno., 2012. Pengertian Pedagang Sayur Keliling, <<http://harno-net.blogspot.com/2012/05/pengertian-pedagang-sayur-keliling.html>>. [Diakses pada Januari 2016].
- Meier, J.D. Farre, Carlos. Bansode, Prashant. Barber, Scot. Rea, Dennis. 2016. Performance Testing Guidance for Web Applications. <https://msdn.microsoft.com/en-us/library/bb924357.aspx>. Microsoft. Diakses Pada 23 April 2016.
- Menasce, Daniel A. 2002. *Load Testing of Web Sites*. George Mason University.
- Nidhra, Srinivas and Jagruthi Dondeti. 2012. *Black Box And White Box Testing Techniques –A Literature Review*. Sweden: School of Computing, Blekinge Institute of Technology, Karlskrona.

Patil , Sheetal S. 2012. Indentification of Performance Improving Factors for Web Application by Performance Testing. India.

Pusdatin-Kementrian Pertanian, 2015. Perkembangan Trend Pemasaran Sayur di Indonesia. <<http://pusdatin.setjen.pertanian.go.id/>>. [Diakses 20 November 2015].

Pressman, Roger., 2001. Software Engineering. McGraw-Hill series in computer science.

RoyalCyber Inc, 2009. Rational Performance Tester. IBM Global Service. United States of America.

Salahudin, Rosa., 2011. Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek. Informatika. Bandung.

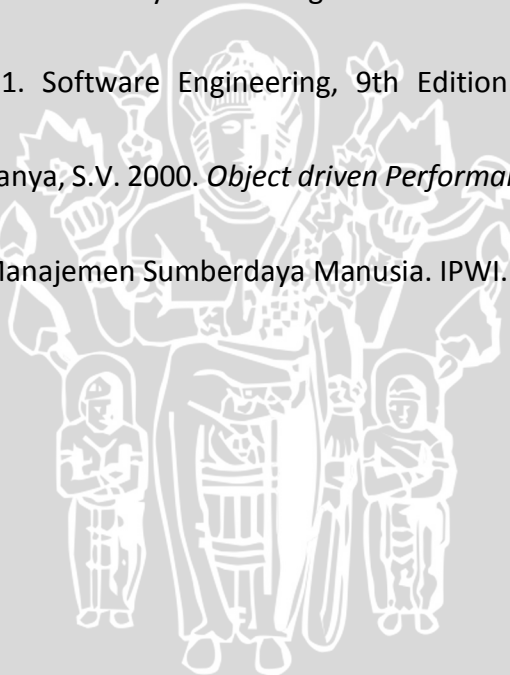
Shaw, J. 2000. *Web Application Performance Testing*. BT Technol J Volume 18 no 2.

Simarmata, Janner., 2010. Rekayasa Perangkat Lunak. CV ANDI OFFSET. Yogyakarta.

Sommerville, Ian., 2011. Software Engineering, 9th Edition. Addison-Wesley. Boston.

Subraya, B.M, Subrahmanya, S.V. 2000. *Object driven Performance Testing of Web Application*. India.

sunyoto, Agus., 1999. Manajemen Sumberdaya Manusia. IPWI. Jakarta.



Lampiran dan Hasil Wawancara

Tabel Lampiran 1 Daftar Harga Sayur

Nama Sayur	Harga Beli	Harga Jual	Satuan
Bawang Putih	38.600	39.500	Kg
Bawang Merah	44.700	46.000	Kg
Tahu	4.400	5.000	Bungkus
Tempe	4.500	5.000	Bungkus
Cabe Rawit	23.800	25.000	Kg
Cabe Merah	28.500	30.000	Kg
Sawi Hijau	1.000	2.000	Ikut
Asem	500	1.000	Bungkus
Kubis	1.500	2.000	Biji
Kacang Panjang	1.500	2.000	Ikut
Tomat	10.500	12.000	Kg
Kerupuk	8.200	2.000	Bungkus
Kentang	1.500	10.000	Kg
Hueci	3.500	5.000	Bungkus
Gurami	35.000	40.000	Kg
Tongkol	18.000	20.000	Kg
Semangka Kuning	5.600	6000	kg
Jagung	6.800	7.500	Kg
Telur Ayam	19.000	22.000	Kg
Daging Sapi	105.000	120.000	kg

Hasil Wawancara

Wawancara dilakukan oleh penulis dengan beberapa penjual sayur keliling disekitar siguragura barat, Malang pada tanggal 20 September 2015, dengan pertanyaan sebagai berikut:

1. Bagaimana proses penjualan pada penjual sayur keliling?

Dalam menjual barang dagangannya para penjual sayur keliling ini sering membawa pulang kembali barang dagangannya yang sudah layu karena para pembeli tidak mau membeli barang yang sudah jelek atau layu. Sehingga para penjual sayur ini tidak mendapatkan untung yang maksimal. Hal ini disebabkan pada penjual sayur keliling hanya mengisi gerobak dagangannya sebanyak dan selengkap mungkin tanpa mengetahui apa yang diinginkan oleh pembeli.

2. Bagaimana proses jual beli antara penjual sayur keliling dengan pembeli?

Para penjual sayur kebanyakan menggunakan sepeda motor untuk menjajakan barang dagangannya. Untuk memanggil para pembeli ini penjual sayur keliling menggunakan klakson supaya para pembeli tahu kalau penjuar sayur keliling lewat.

3. Dimana para penjual sayur keliling mengambil barang dagangannya?

Para penjual sayur keliling biasanya mengambil barang dagangan langsung dari pengepul atau dari pasar subuh.

4. Jam berapa biasanya para penjual sayur mulai mengambil barang dagangan?

Biasanya para penjual sayur keliling sudah berangkat dari jam 3 pagi dari rumah untuk mengaambil barang dagangan.

5. Berapa untung yang diambil dan berapa harga beli serta harga jual sayuran?

Harga jual yang diberikan penjual sayur keliling rata-rata hampir sama berkisar antara 500-3000 rupiah bergantung dari barang yang dijual.