BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perancangan

Perancangan merupakan tahap lanjutan dari analisis kebutuhan. Hasil dari analisis kebutuhan yang telah didefinisikan sebelumnya akan digunakan di dalam tahap perancangan ini. Dalam tahap perancangan, hasil dari analisis kebutuhan akan dilakukan identifikasi berdasarkan spesifikasi kebutuhan dan use case *scenario*. Hasil dari analisis kebutuhan akan dijelaskan lebih detail serta bagaimana hubungannya. Proses perancangan pada Sistem Ujian Harian Siswa ini dibagi menjadi 5 tahap, yaitu pemodelan *sequence diagram*, pemodelan *class diagram*, pemodelan data, perancangan komponen, serta perancangan *interface*.

5.1.1 Pemodelan *Sequence Diagram*

Sequence diagram adalah suatu diagram yang menggambarkan suatu kolaborasi antar objek. *Sequence diagram* didapatkan dari *use case scenario* pada tahapan rekayasa kebutuhan. Jumlah *sequence diagram* adalah sama dengan jumlah *use case scenario*. Dilihat dari jumlah *use case scenario* maka jumlah *sequence diagram* adalah tiga puluh enam buah diagram. Namun yang akan ditampilkan sebagai sampel hanya tiga buah diagram saja yaitu *login*, mengerjakan ujian dan melihat hasil ujian dari sisi siswa.

1. Sequence Diagram Login

Sequence diagram login menggambarkan alur yang terjadi saat melakukan *login*. *Sequence diagram login* dapat dilihat pada Gambar 5.1.

2. Sequence Diagram Mengerjakan Ujian

Sequence diagram mengerjakan ujian menggambarkan alur yang terjadi saat mengerjakan ujian. *Sequence diagram* mengerjakan ujian dapat dilihat pada Gambar 5.2.

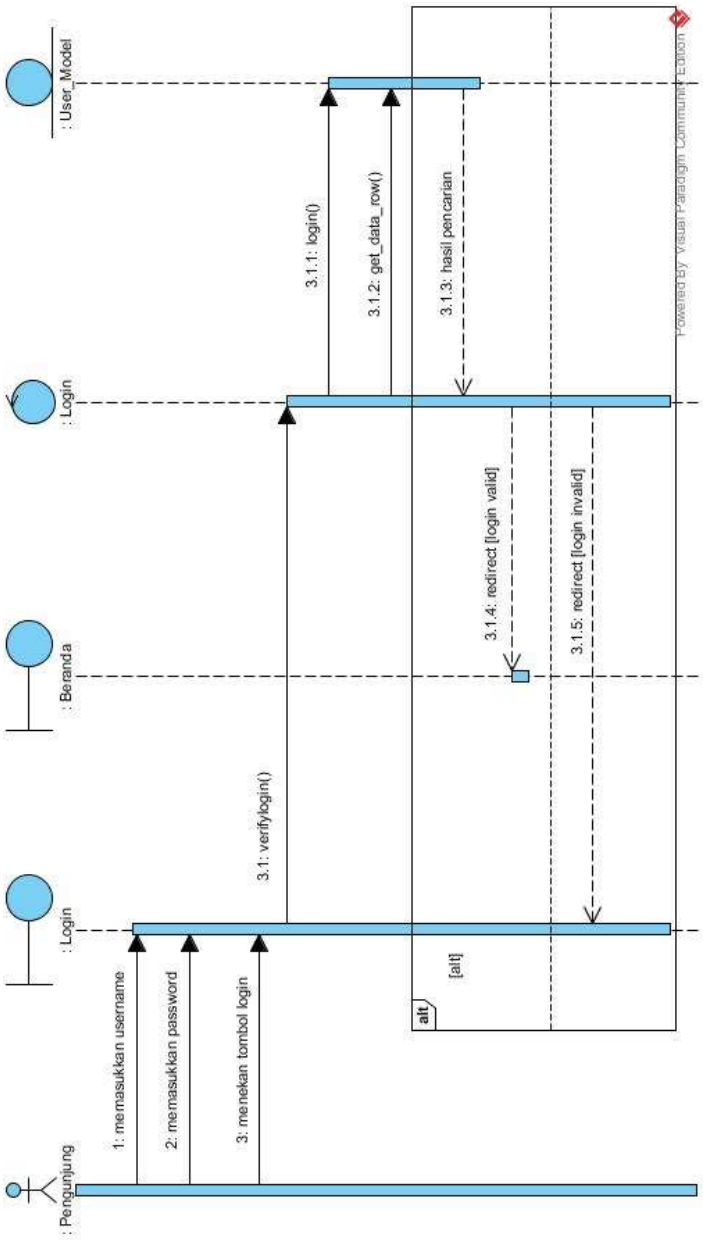
3. Sequence Diagram Hasil Ujian

Sequence diagram hasil ujian meggambarkan alur yang terjadi saat melihat hasil ujian. *Sequence diagram* hasil ujian dapat dilihat pada Gambar 5.3.

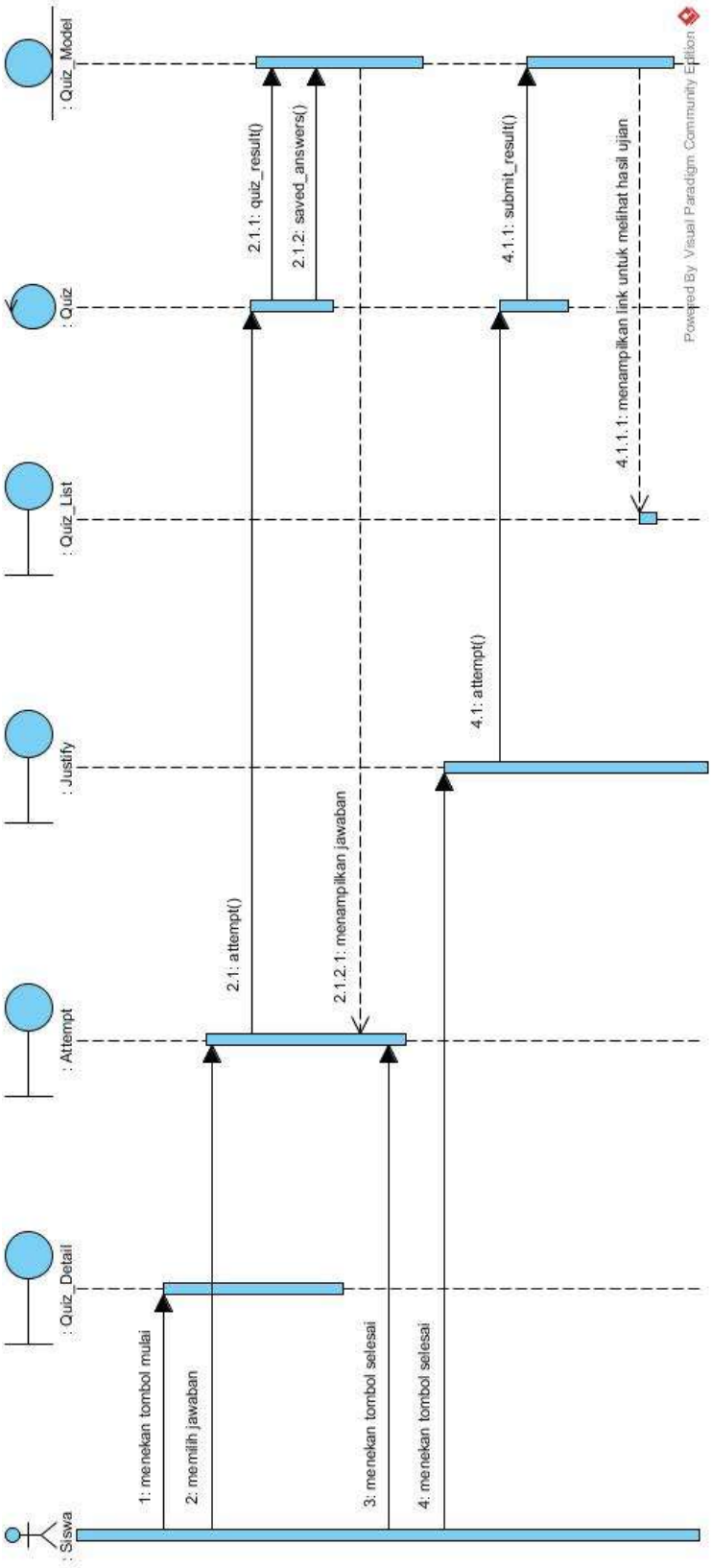
5.1.2 Pemodelan *Class Diagram*

Pemodelan *class diagram* merupakan pemodelan dan pengembangan *class diagram* level analisis yang siap untuk diimplementasikan. Pemodelan *class diagram* pada Sistem Ujian Harian Siswa ini merupakan pemodelan *class diagram* yang berdasarkan pada *framework* yang digunakan yaitu *Code Igniter* yang menggunakan model MVC (*Model, View, Controller*) sehingga *class diagram* yang dimodelkan dibagi menjadi class diagram *controller*, *class diagram model* dan *class diagram view*. Class dari *class diagram* analisis yang meliputi *class* Siswa, Guru dan Admin dijadikan *class controller* dengan nama *class User* dan *class model* dengan nama *User_Model. Class Category* pada *class diagram* analisis dijadikan *model*

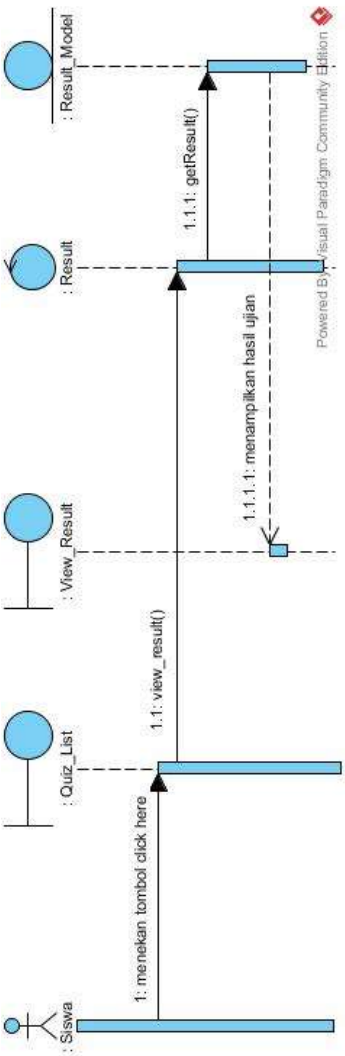
dengan nama class *Mapel_Model*. *Class Result* dijadikan *model* dengan nama *class Resut_Model* dan *class controller* dengan nama *Result*. *Class Quiz* dijadikan *model* dengan nama *class Quiz_Model* dan *class controller* dengan nama *Quiz*. *Class Qbank* dijadikan *model* dengan nama *class Qbank_Model* dan *class controller* dengan nama *Qbank*. Gambaran umum pemodelan *class diagram* dapat dilihat pada Gambar 5.4. Spesifik dari pemodelan *class diagram controller* dapat dilihat pada Gambar 5.5, dan spesifik dari pemodelan *class diagram model* dapat dilihat pada Gambar 5.6.



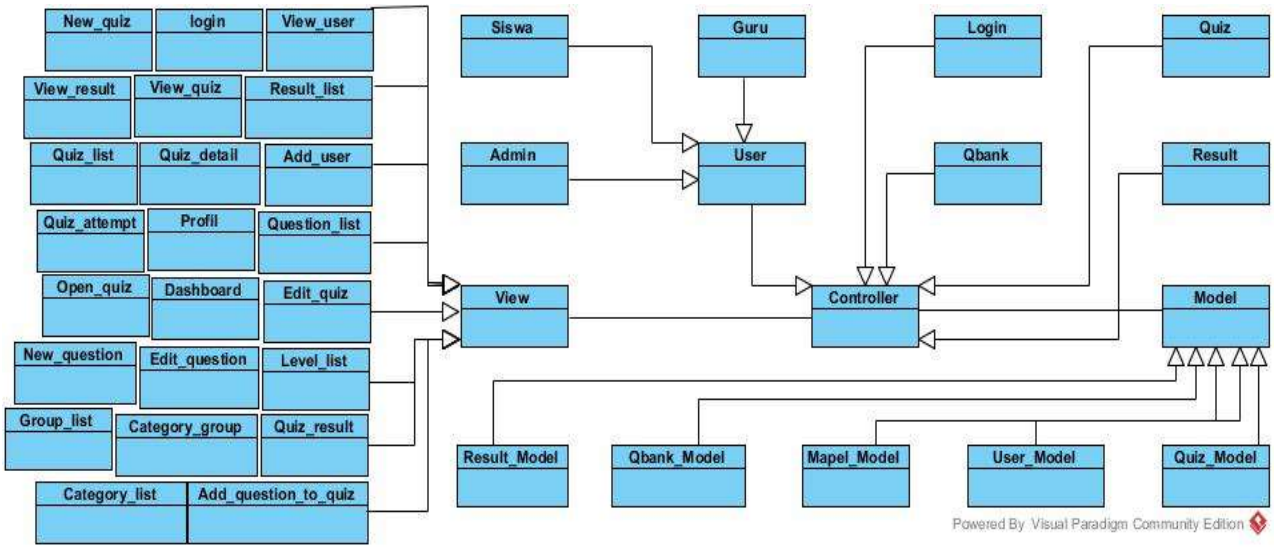
Gambar 5.1 *Sequence diagram login*



Gambar 5.2 *Sequence diagram* mengerjakan ujian



Gambar 5.3 *Sequence diagram* hasil ujian



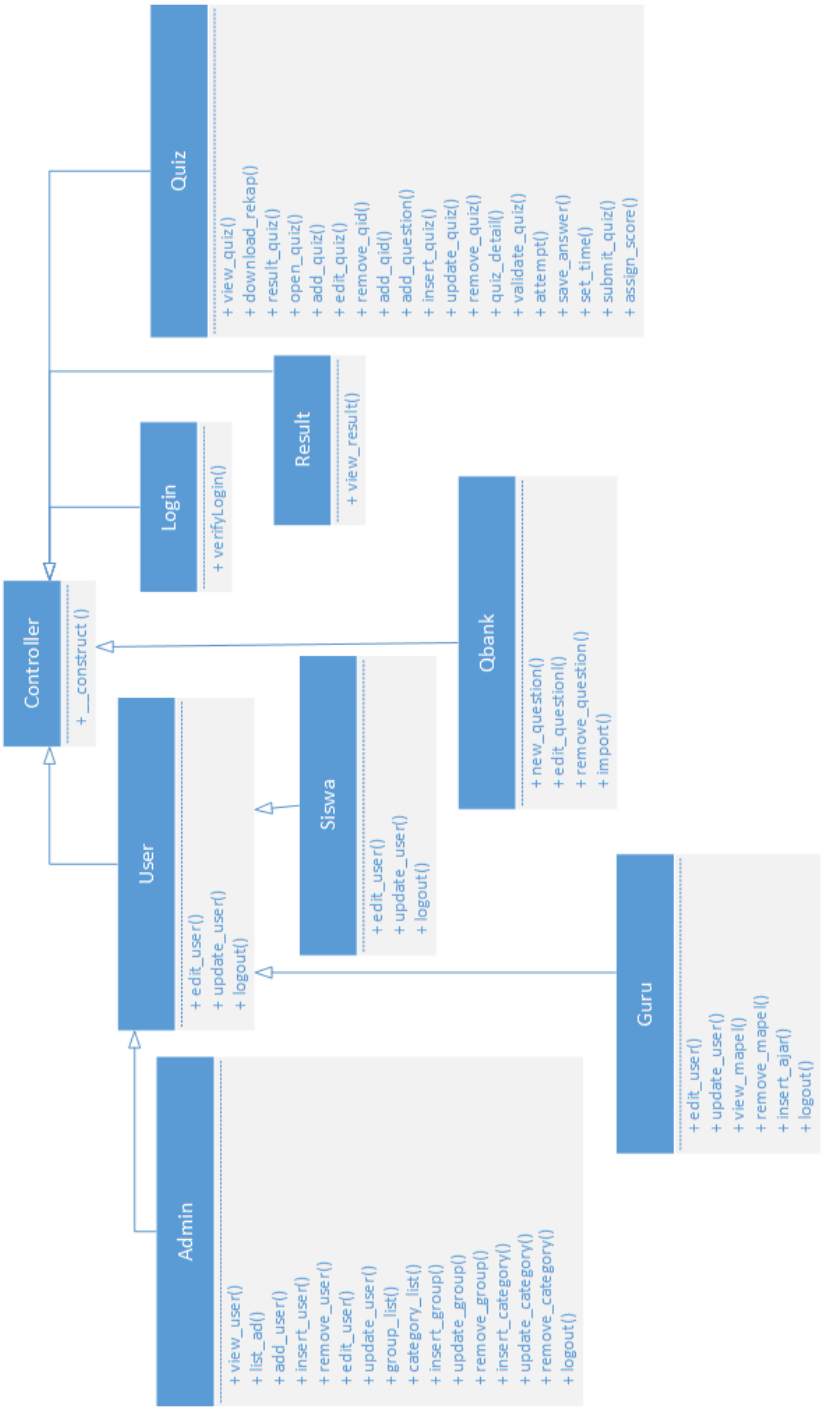
Gambar 5.4 Gambaran umum pemodelan *class diagram*

Pada Gambar 5.4 *class Controller, Model, dan View* merupakan *class library* yang terdapat pada *framework*. Terdapat 3 *class* yang memiliki hubungan *inheritance* dengan *class User* dan *class User* memiliki hubungan *inheritance* dengan *class Controller*, selain itu terdapat 5 *class* yang memiliki hubungan *inheritance* dengan *class Model*. *Class* yang memiliki hubungan *inheritance* dengan *class User* yaitu *class* Admin, *class* Guru dan *class* Siswa. Sedangkan *class Login*, *class Qbank*, *class Quiz* dan *class Result* memiliki hubungan *inheritance* langsung dengan *class Controller*. *Class* yang memiliki hubungan *inheritance* dengan *class Model* yaitu *class Result_Model, class Qbank_Model, class Mapel_Model, class User_Model,* dan *class Quiz_Model*. Selain itu terdapat 23 *class* yang memiliki hubungan *inheritance* dengan *class View*.

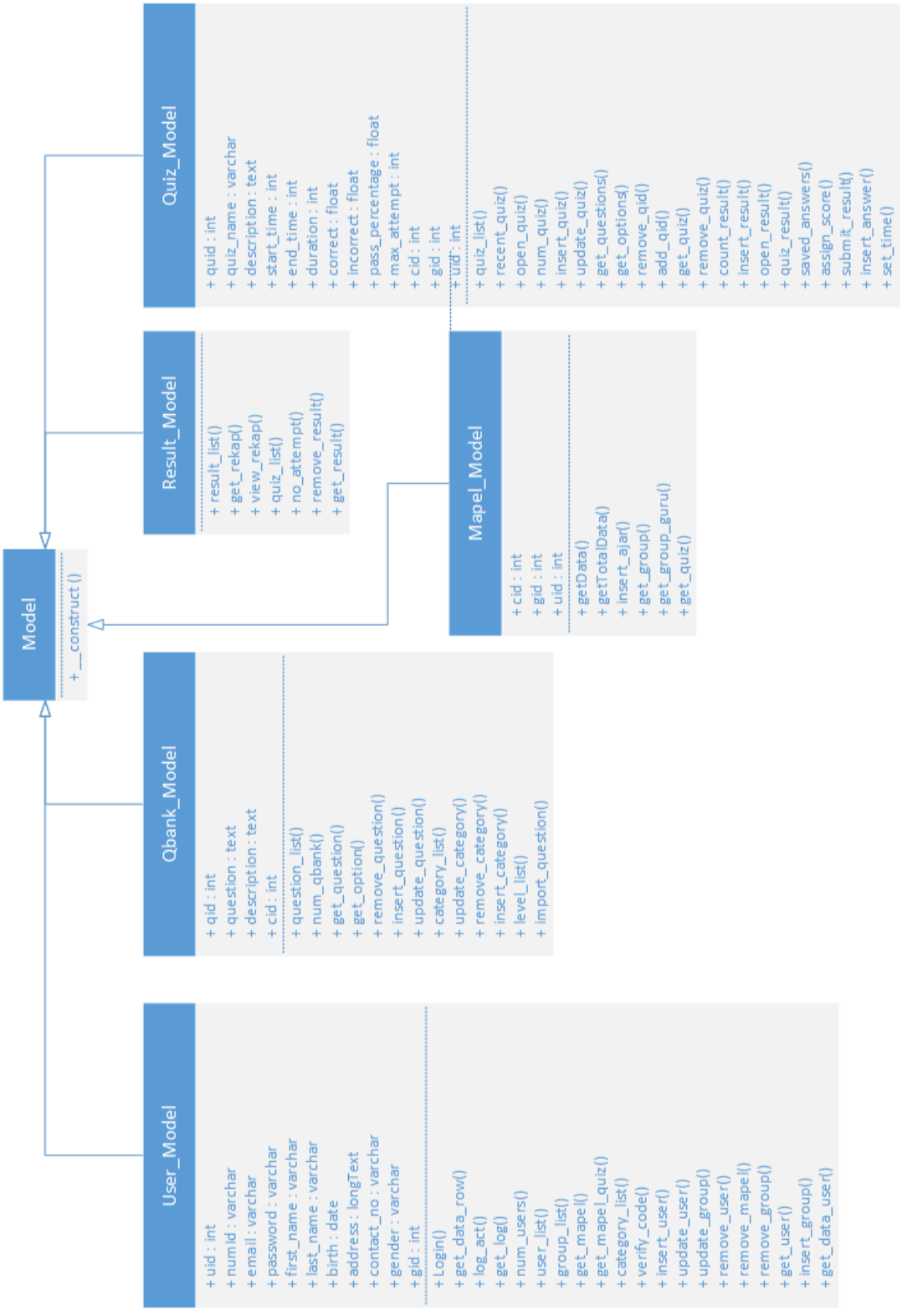
5.1.3 Pemodelan Data

Pemodelan basis data dibuat untuk merancang manajemen data yang akan digunakan di dalam pembangunan sistem. Manajemen data terdiri dari data relevan yang akan digunakan di dalam sistem serta didukung oleh *Database Manajemen System* (DBMS). Dalam pembangunan sistem ini, pemodelan data digambarkan dengan *Entity Relationship Diagram* (ERD). ERD Sistem Ujian Harian Siswa Berbasis *Web* dapat dilihat pada Gambar 5.7.

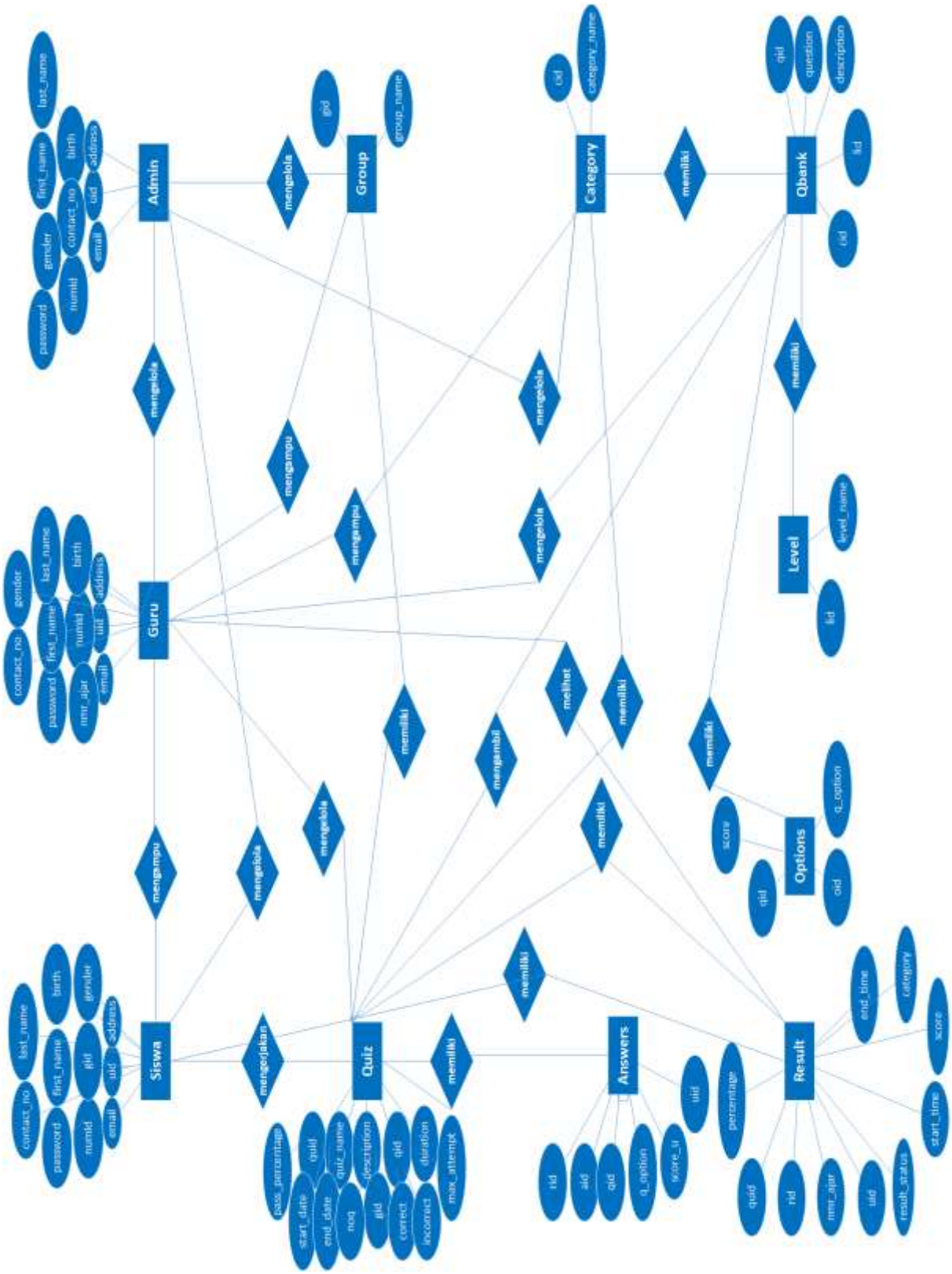
Dalam *Entity Relationship Diagram* Sistem Ujian Harian Siswa ini terdapat sebelas *entity* diantaranya Siswa, Guru, Admin, *Group*, *Category*, *Qbank*, *Level*, *Option*s, *Quiz*, *Answers* dan *Result*. Guru dengan Siswa, *Category* serta *Group* memiliki relasi yaitu Guru mengampu Siswa, *Group* dan *Category*. Terdapat pula relasi antara Admin dengan Siswa, Guru, *Group* dan *Category* yaitu Admin mengelola data Siswa, Guru, *Group* dan *Category.* Relasi Guru dengan *Qbank* dan *Quiz* adalah Guru mengelola semua soal yang ada pada *Qbank* dan Guru mengelola *Quiz* untuk nantinya dikerjakan oleh Siswa. Relasi Guru dengan *Result* adalah Guru dapat melihat hasil ujian yang telah dikerjakan oleh Siswa dalam sistem. Relasi Siswa dengan *Quiz* adalah Siswa dapat mengerjakan ujian dalam sistem. Relasi Siswa dengan *Result* adalah Siswa memiliki hasil ujian. Relasi *Quiz* dengan *Answers* dan *Result* adalah setiap ujian memiliki jawaban yang dikerjakan Siswa dan memiliki hasil ujian Siswa. Relasi *Quiz* dengan *Qbank* adalah setiap ujian yang ada selalu mengambil soal dari bank soal. Relasi *Quiz* dengan *Category* dan *Group* adalah setiap ujian memiliki kelas dan mata pelajaran. Serta relasi antara *Qbank* dengan *Level* dan *Options* adalah bank soal memiliki pilihan *level* dan pilihan opsi pada setiap soalnya.



Gambar 5.5 *Class diagram controller*



Gambar 5.6 *Class diagram model*



Gambar 5.7 ERD Sistem Ujian Harian Siswa

5.1.4 Perancangan Komponen

Perancangan komponen ini berupa algoritme yang akan menjelaskan setiap operasi yang ada pada masing-masing *class*. Algoritme yang dibuat berupa *pseudocode* yang menjembatani antara bahasa manusia dengan sistem. Algoritme pada setiap operasi akan menjelaskan langkah-langkah suatu sistem untuk dapat menghasilkan keluaran. Dalam perancangan komponen ini hanya akan menjelaskan tiga operasi yaitu operasi *verifylogin()* pada *class Login*, operasi *attempt()* pada *class controller Quiz*, dan operasi *view_result()* pada *class controller Result*.

1. Algoritme operasi *verifylogin*() pada *class controller login*

Algoritme operasi *verifylogin()* ini menggambarkan proses verifikasi pengguna oleh sistem sehingga yang tidak memiliki kepentingan tidak dapat mengakses sistem dan pengguna dapat dibedakan berdasarkan otoritasnya. Algoritme ini dimulai dengan proses inisialisasi variabel yaitu *username, password* dan *user* dimana variabel *user* didapatkan dari model *User_Model*. Apabila data *username* dan *password* benar maka sistem akan mengijinkan pengguna tersebut masuk ke dalam sistem sesuai dengan otoritasnya dan menampilkan halaman *dashboard* pengguna tersebut, namun apabila *username* dan/atau *password* salah maka sistem akan menampilkan pesan data yang dimasukkan salah dan tetap menampilkan halaman *login.*

Berikut Tabel 5.1 yang berisi algoritme operasi *verifylogin().*

Tabel 5.1 Algoritme operasi *verifylogin()* pada *class controller login*

```
START
        username = POST[email]
        password = POST[password]
        user = model(User_Model)
        IF(user){
                table = user
                where = array (email = username)
                data = model(User_Model)
                idUser = data
                SESSION[setdatauser] = logged_in sebagai user
                SESSION[setdatauser] = LoginPengguna sebagai username
                SESSION[setdatauser] = idUser sebagai idUser
                redirect dashboard
        ELSE
                SESSION[setflashdata] = message sebagai invalidLogin
```

Tabel 5.1 Algoritme operasi *verifylogin*() pada *class controller login* (lanjutan)

```
                redirect Login

END
```

2. Algoritme operasi *attempt*() pada *class controller quiz*

Algoritme operasi *attempt()* ini menggambarkan proses mengerjakan ujian sampai selesai. Algoritme operasi *verifylogin()* dimulai dengan melakukan pengecekan pada variabel *idResult2,* apabila nilai *idResult* tidak sama dengan *idResult2* atau dapat diartikan bahwa *idResult2* telah terisi maka sistem akan menampilkan pesan bahwa ujian telah berakhir dan akan menampilkan halaman *Quiz*. Apabila tidak memenuhi kondisi sebelumnya maka sistem akan melakukan pengecekan pada kondisi apabila tanggal selesai ujian telah terlewati maka sistem akan menampilkan pesan bahwa ujian telah berakhir dan menampilkan halaman *Quiz_detail*. Namun apabila tidak memenuhi kedua kondisi di atas maka sistem akan melakukan pengecekan lagi pada kondisi waktu mulai ujian dan waktu pelaksanaan ujian masih ada atau terpenuhi maka sistem akan menampilkan halaman ujian dan siswa dapat mengerjakan ujian, apabila waktu ujian telah berakhir maka sistem akan menampilkan pesan waktu berakhir dan menampilkan halaman *Quiz* beserta *link* halaman hasil ujian.

Berikut Tabel 5.2 yang berisi algoritme operasi *attempt().*

Tabel 5.2 Algoritme operasi *attempt()* pada *class controller quiz*

```
START
        idResult2 = SESSION[idResult]
        IF (idResult!= idResult2)
                SESSION[setflashdata] = alert UJIAN BERAKHIR
                redirect Quiz
        ELSEIF (data Quiz, tanggalberakhir < time)
                model(Quiz) menjalankan submit_result
                SESSION[unsetDataUser] = idResult
                SESSION[setflashdata] = alert UJIAN BERAKHIR
                redirect Quiz/Quiz_detail , data(Quiz, idQuiz)
        ELSEIF (data Quiz, waktuMulai + data Quiz, waktuPelaksanaan < time)
                model(Quiz) menjalankan submit_result
                SESSION[unsetDataUser] = idResult
                SESSION[setflashdata] = alert WAKTU BERAKHIR
                redirect Quiz/Quiz_detail , data(Quiz, idQuiz)
        ENDIF
```

Tabel 5.2 Algoritme operasi *attempt*() pada *class controller quiz* (lanjutan)

*data(seconds) = (data Quiz, waktuPelaksanaan*60) – (time – data Quiz, waktuMulai)*
data(question) = model(Quiz) menjalankan getQuestion
data(option) = model(Quiz) memanggil getOption
data(title) = data nama Quiz
template (nama view, data)
END

3. Algoritme operasi *view_result*() pada *class controller result*

Algoritme operasi *view_result()* ini menggambarkan proses melihat hasil ujian. Algoritme operasi *view_result()* dimulai dengan pengecekan *user* apakah sebagai siswa atau guru. Apabila siswa maka akan menjalankan *method getGroup()* dan diijinkan melihat hasil ujian. Kemudian dilakukan pengecekan lagi, apabila memenuhi kondisi siswa diijinkan melihat jawaban yang benar maka sistem akan memanggil data *submit_answers*, *question* dan *option* pada *Quiz_Model*. Namun jika tidak memenuhi kondisi diijinkan melihat jawaban yang benar maka sistem hanya akan menampilkan hasil ujian saja. Apabila guru maka sistem hanya akan menampilkan hasil ujian siswa saja.

Berikut Tabel 5.3 yang berisi algoritme operasi *view_result().*

Tabel 5.3 Algoritme operasi *view_result()* pada *class controller result*

START
IF (logged_in sebagai siswa)
data(getGroup) = model(Mapel) menjalankan getGroup
ELSEIF (logged_in sebagai guru)
data(getGroupGuru) = model(Mapel) menjalankan getGroupGuru
ENDIF
data(result) = model(Result) menjalankan getResult
data(attempt) = model(Result) menjalankan attempt (data result, idQuiz , data Result, idUser)
IF (data result, view_answers = 1)
model(Quiz)
data(submit_answers) = model(Quiz) menjalankan submit_answers
data(question) = model(Quiz) menjalankan getQuestion

Tabel 5.3 Algoritme operasi *view_result()* pada *class controller result* (lanjutan)

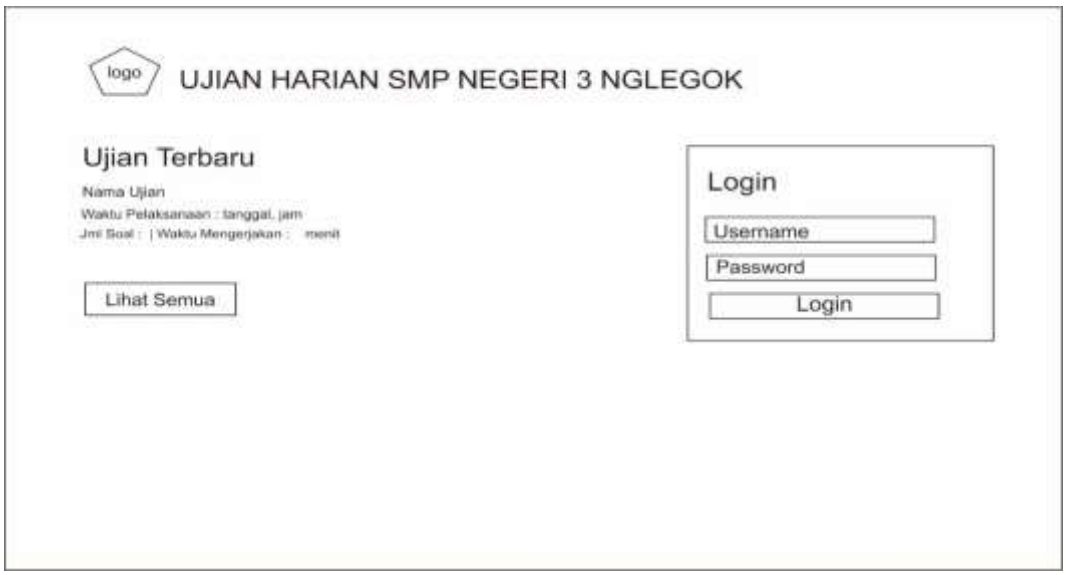
data(answers) = model(Quiz) menjalankan getAnswers *ELSEIF (SESSION[userdata] = logged_in)* *template (nama view, data)* *ELSE* *template (nama view, data)* *END*

5.1.5 Perancangan *Interface*

Perancangan *interface* dirancang untuk menghasilkan *interface* sistem agar output yang dihasilkan oleh sistem dapat diterima oleh pengguna. Output yang dihasilkan oleh sistem diantaranya *Login*, mengerjakan ujian, dan hasil ujian.

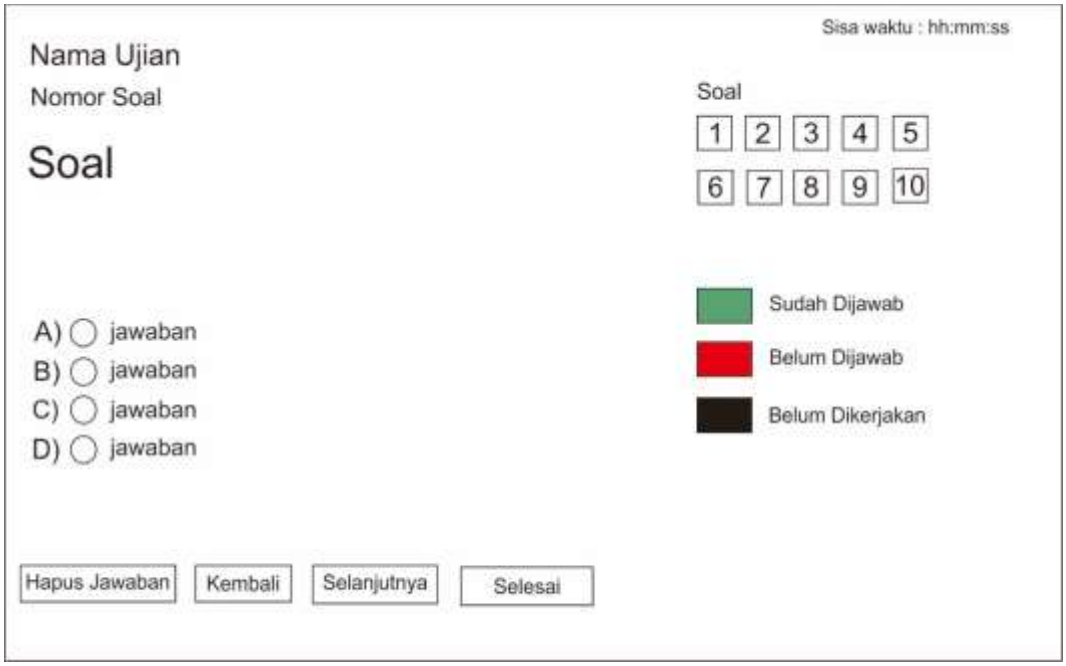
1. Interface* Halaman *Login

Halaman ini adalah halaman untuk melakukan *Login* kedalam sistem. Dalam halaman *Login* terdapat form *Login* yang terdiri dari *username*, *password* dan tombol *Login*. Selain itu dalam halaman *Login* juga terdapat nama-nama ujian terbaru yang terdapat dalam sistem serta tombol lihat semua untuk menampilkan seluruh ujian terbaru yang terdapat dalam sistem. Interface halaman *login* dapat dilihat pada Gambar 5.8.



Gambar 5.8 *Interface* halaman *login*

2. *Interface* Halaman Mengerjakan Ujian

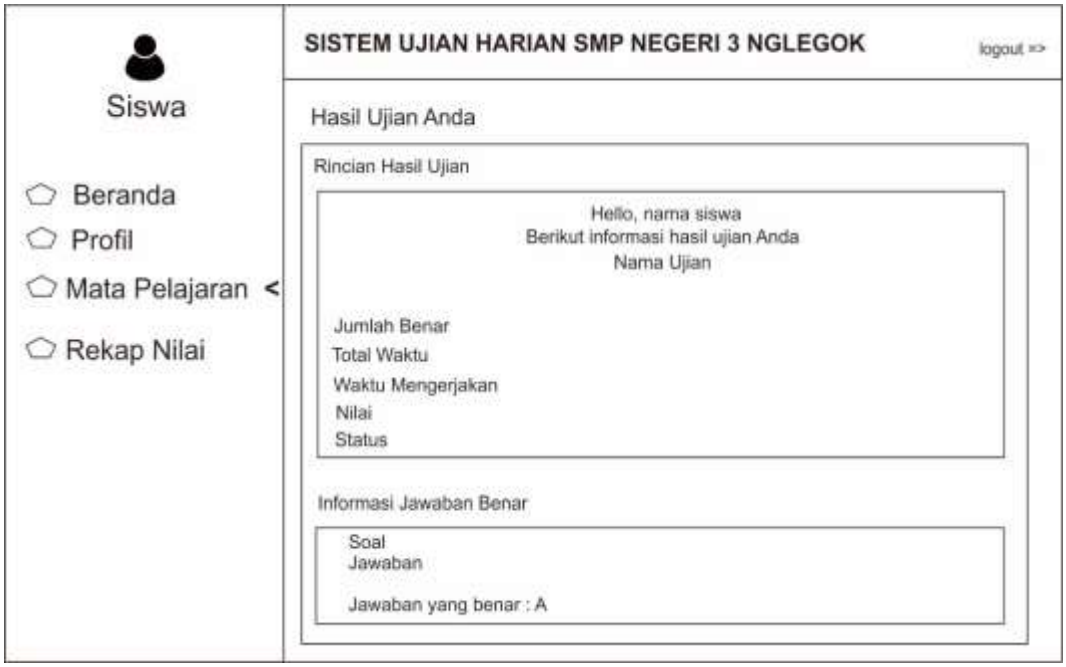


Gambar 5.9 *Interface* halaman mengerjakan ujian

Gambar 5.9 menggambarkan halaman mengerjakan ujian yang dilakukan oleh siswa. Di halaman ini terdapat nama ujian, nomor soal, soal, pilihan jawaban, *timer*, penanda nomor soal, keterangan tentang soal jika sudah dijawab berwarna hijau, belum dijawab berwarna merah, belum dikerjakan berwarna hitam, serta tombol hapus jawaban, tombol kembali untuk kembali ke soal sebelumnya, tombol selanjutnya untuk ke soal selanjutnya, tombol selesai untuk menyudahi ujian dan keluar dari ujian. Apabila siswa telah selesai mengerjakan ujian dan menekan tombol selesai maka akan menampilkan pesan peringatan jika siswa ingin menyudahi ujian dapat menekan tombol selesai dan jika ingin membatalkan dapat menekan tombol belum.

3. *Interface* Halaman Hasil Ujian

Halaman ini adalah halaman untuk melihat hasil ujian. Halaman ini menampilkan hasil ujian siswa yang terdiri dari rincian hasil ujian yang meliputi nama siswa, nama ujian, jumlah jawaban benar, total waktu yang digunakan siswa untuk mengerjakan ujian sampai selesai, waktu mengerjakan siswa yang berisi tanggal dan jam mulai mengerjakan, nilai yang didapat siswa, dan status ujian siswa apakah lulus atau tidak. Selain itu dalam halaman hasil ujian juga terdapat informasi jawaban terkait ujian tersebut yang meliputi soal, jawaban, dan jawaban yang benar. Dalam halaman ini terdapat juga sidebar dan topbar. Sidebar siswa meliputi menu beranda, profil, mata pelajaran, dan rekap nilai. Sedangkan topbar siswa meliputi nama sistem yaitu Sistem Ujian Harian Siswa SMP Negeri 3 Nglegok dan tombol *logout*. Perancangan *interface* halaman hasil ujian dapat dilihat pada Gambar 5.10.



Gambar 5.10 *Interface* halaman hasil ujian

5.2 Implementasi

Dalam proses implementasi Sistem Ujian Harian Siswa Berbasis *Web* ini meliputi spesifikasi lingkungan implementasi, batasan implementasi, implementasi basis data, implementasi *class*, implementasi algoritme, serta implementasi *interface*.

5.2.1 Spesifikasi Lingkungan Implementasi

Dalam pembangunan Sistem Ujian Harian Siswa Berbasis *Web* ini dikembangkan dalam lingkungan yang terdiri dari perangkat keras dan perangkat lunak. Berikut spesifikasi lingkungan perangkat keras dan perangkat lunak yang digunakan.

1. Spesifikasi Perangkat Keras

Dalam pembangunan Sistem Ujian Harian Siswa menggunakan spesifikasi perangkat keras yang dapat dilihat pada Tabel 5.4 berikut.

Tabel 5.4 Spesifikasi perangkat keras

Nama Komponen	Spesifikasi
Prosesor	Intel Core i3
Memori (RAM)	6 GB
Harddisk	

2. Spesifikasi Perangkat Lunak

Dalam pembangunan Sistem Ujian Harian Siswa menggunakan spesifikasi perangkat lunak yang dapat dilihat pada Tabel 5.5 berikut.

Tabel 5.5 Spesifikasi perangkat lunak

Nama Komponen	Spesifikasi
Sistem Operasi	Windows 8.1
Browser	Mozilla Firefox 53.0.3
Code Editor	Sublime Text 2.0.2
Basis Data	MySQL 5.6
Pemodelan, UML	Astah Community 6.6.4, Visual Paradigm 13.2, 14.0
Framework	Code Igniter 2.1.4, Bootstrap
Dokumentasi	MS Word 2013

5.2.2 Batasan Implementasi

Berikut ini batasan implementasi Sistem Ujian Harian Siswa :

1. Sistem Ujian Harian Siswa dibangun dengan menggunakan bahasa pemrograman PHP, *framework* Code Igniter dan Bootsrap untuk pembuatan *web*, serta menggunakan MySQL sebagai pendukung basis data.
2. Sistem Ujian Harian Siswa memiliki fitur utama berupa menambah soal dan jawaban kedalam bank soal, membuat ujian harian, mengerjakan ujian harian, melihat hasil ujian.
3. Data yang digunakan dalam pembangunan Sistem Ujian Harian Siswa diambil dari SMPN 3 Nglegok.

5.2.3 Implementasi Program

Proses implementasi Sistem Ujian Harian Siswa ini menggunakan bahasa pemrograman PHP dengan *framework* Code Igniter yang menggunakan konsep MVC. Dalam implementasi sistem ini terdapat fungsi-fungsi seperti *Login*, mengerjakan ujian dan hasil ujian. Berikut adalah implementasi program dari beberapa fungsi yang ada di dalam Sistem Ujian Harian Siswa.

1. Implementasi Program *Login*

Berikut Tabel 5.6 yang berisi kode program *Login* pada *method verifylogin*() pada *class* controller *Login*.

Tabel 5.6 Kode program *login*

```
$username=$this->input->post('email');
$password=$this->input->post('password');
$user=$this->user_model->Login($username,$password);
if($user){
      $table = "users";
      $where=array(
            'email' => $username);
      $data['data']=$this->user_model->get_data_row
      ($table,$where);
      $uid = $data['data']->uid;
      $user['base_url']=base_url();
      $this->session->set_userdata('logged_in', $user);
      $this->session->set_userdata('user_Login', $username);
      $this->session->set_userdata('user_id', $uid);
      redirect('dashboard');
}else{
      $this->session->set_flashdata('message',$this->lang-
      >line('invalid_Login'));
      redirect('Login');
}
```

2. Implementasi Mengerjakan Ujian

Berikut Tabel 5.7 yang berisi kode program mengerjakan ujian pada *method attempt*() pada *class* controller *Quiz*.

Tabel 5.7 Kode program mengerjakan ujian

```
$srid=$this->session->userdata('rid');
if($rid != $srid){
      $this->session->set_flashdata('message',"<div class='alert
      alert-danger'>".$this->lang->line('quiz_ended')."</div>");
      redirect('quiz/');
}
$data['quiz']=$this->quiz_model->quiz_result($rid);
$data['saved_answers']=$this->quiz_model->saved_answers($rid);
if($data['quiz']['end_date'] < time()){
      $this->quiz_model->submit_result($rid);
```

Tabel 5.7 Kode program mengerjakan ujian (lanjutan)

```
$this->session->unset_userdata('rid');
      $this->session->set_flashdata('message',"<div class='alert
      alert-danger'>".$this->lang->line('quiz_ended')."</div>");
      redirect('quiz/quiz_detail/'.$data['quiz']['quid']);
}
if(($data['quiz']['start_time']+($data['quiz']['duration']*60))
< time()){
      $this->quiz_model->submit_result($rid);
      $this->session->unset_userdata('rid');
      $this->session->set_flashdata('message',"<div class='alert
      alert-danger'>".$this->lang->line('time_over')." </div>");
      redirect('quiz/quiz_detail/'.$data['quiz']['quid']);
}
$data['seconds']=($data['quiz']['duration']*60)-(time()-$data
['quiz']['start_time']);
$data['questions']=$this->quiz_model->get_questions($data
['quiz']['r_qids']);
$data['options']=$this->quiz_model->get_options($data['quiz']
['r_qids']);
$data['title']=$data['quiz']['quiz_name'];
$this->load->view('header',$data);
$this->load->view('quiz_attempt',$data);
$this->load->view('footer',$data);
```

3. Implementasi Hasil Ujian

Berikut Tabel 5.8 yang berisi kode program hasil ujian pada *method view_result*() pada *class* controller *Result*.

Tabel 5.8 Kode program hasil ujian

```
if($logged_in['su']=='0'){
      $data['get_kelas']=$this->Mapel_model->get_kelas
      ($logged_in['uid']);
}
if($logged_in['su']=='2'){
      $data['get_kelas_guru']=$this->Mapel_model->get_kelas_guru
      ($logged_in['uid']);
}
$data['result']=$this->result_model->get_result($rid);
$data['attempt']=$this->result_model->no_attempt($data['result']
['quid'],$data['result']['uid']);
```

Tabel 5.8 Kode program hasil ujian (lanjutan)

```
$data['title']=$this->lang->line('result_id').''.$data['result']
['rid'];
if($data['result']['view_answer']=='1'||$logged_in ['su']=='1'){
      $this->load->model("quiz_model");
      $data['saved_answers']=$this->quiz_model->saved_answers
      ($rid);
      $data['questions']=$this->quiz_model->get_questions
      ($data['result']['r_qids']);
      $data['options']=$this->quiz_model->get_options($data
      ['result']['r_qids']);
}
if($this->session->userdata('logged_in')){
      $this->load->view('view_result',$data);
}else{
      $this->load->view('view_result_without_Login',$data);
}
```

5.2.4 Implementasi *Interface*

Interface sistem merupakan sarana yang digunakan pengguna untuk berinteraksi dengan sistem. Berikut beberapa penjelasan mengenai implementasi *interface* yang telah dibuat pada Sistem Ujian Harian Siswa.

1. ***Implementasi* Interface Login**

   Implementasi *interface Login* dapat dilihat pada Gambar 5.11.

2. **Implementasi *Interface* Mengerjakan Ujian**

   Implementasi *interface* mengerjakan ujian dapat dilihat pada Gambar 5.12.

3. **Implementasi *Interface* Melihat Hasil Ujian**

   Implementasi *interface* melihat hasil ujian dapat dilihat pada Gambar 5.13, 5.14, 5.15.

Gambar 5.11 *Interface* halaman *login*

Gambar 5.12 *Interface* halaman mengerjakan ujian

Gambar 5.13 *Interface* melihat hasil ujian (1)

Gambar 5.14 *Interface* melihat hasil ujian (2)

Gambar 5.15 *Interface* melihat hasil ujian (3)