BAB III

METODE PENELITIAN DAN PERANCANGAN SISTEM

3.1 Metode Penelitian

Metode penelitian yang digunakan dalam pembuatan sistem informasi ini yaitu :

3.1.1 Pembuatan *Model*

Pembuatan sistem aplikasi *web service* "*Pengembangan Web api Pada Sistem Assesmen Berbasis Tag Sebagai Pembantu Penyusunan Strategi Pembelajaran"* ini menggunakan struktur *model* waterfall dengan beberapa tahapan aktifitas yang terstruktur dimana dari tiap-tiap tahapan akan dicapai hasil yang maksimal guna menunjang pembuatan aplikasi sistem yang baik (Rumbaugh, J. dkk, 1991). Adapun tahapan-tahapan yang dilakukan antara lain analisa dan definisi kebutuhan sistem, desain sistem, implementasi sistem yang akan dijelaskan pada bagian Prosedur Penelitian, sedang untuk tahapan uji sistem akan dijelaskan pada bagian Evaluasi.

3.1.2 Prosedur Penelitian

Tahapan-tahapan pembuatan sistem aplikasi *web service* ini dapat dijabarkan sebagai berikut :

a. Menganalisa dan mendefinisikan kebutuhan sistem

   Tahapan ini dimaksudkan agar kita mengetahui tentang apa yang perlu dipelajari, serta data-data pendukung apa saja yang diperlukan aplikasi *web*

service "*Pengembangan Web api Pada Sistem Assesmen Berbasis Tag Sebagai Pembantu Penyusunan Strategi Pembelajaran"*.

b. Desain Sistem

Pembuatan desain sistem dari aplikasi *web service* "*Sistem Assesmen Dan Berbasis Tag Sebagai Pembantu Penyusunan Strategi Pembelajaran"* pada tahapan ini meliputi beberapa langkah, diantaranya pembuatan :

1. *Use Case Diagram*
2. *Activity Diagram*
3. *Class Diagram*
4. Desain Skema Sistem *Web Service*
5. Desain Database dan *ER Diagram*

3.1.3 Evaluasi

Pengujian terhadap sistem yang telah dibuat untuk menentukan validasinya. tahapan-tahapan pengujian dapat dijabarkan sebagai berikut :

a. Desain Uji Coba

Pengujian sistem dilakukan secara uji perseorangan dengan harapan masukan-masukan terhadap sistem, pencarian kesalahan terhadap aplikasi sistem yang telah dibuat penting guna proses perbaikan aplikasi pada akhirnya.

b. Tahap Pengumpulan Data

Pelaksanaan wawancara langsung diperlukan untuk memperoleh pemahaman tentang penggunaan *tools*, dan *web api* kepada user.

c. Analisis Hasil Uji Coba

Proses analisis hasil dilakukan untuk menentukan kelayakan sistem.

3.1.4 Alat dan Bahan

Alat yang digunakan dalam penelitian ini mencakup perangkat keras dan perangkat lunak.

3.1.5 Perangkat Keras

Adapun perangkat lunak yang dibutuhkan dalam merancang dan membuat *web service* dari *website* perencanaan program *"Pengembangan Web api Pada Sistem Assesmen Dan Berbasis Tag Sebagai Pembantu Penyusunan Strategi Pembelajaran*" :

1. Sistem operasi windows 8.1 pro
2. Bahasa Pemrograman : C#, *MVC, WEB API*
3. Database Server : Ms SQL Server
4. *Web* Browser.

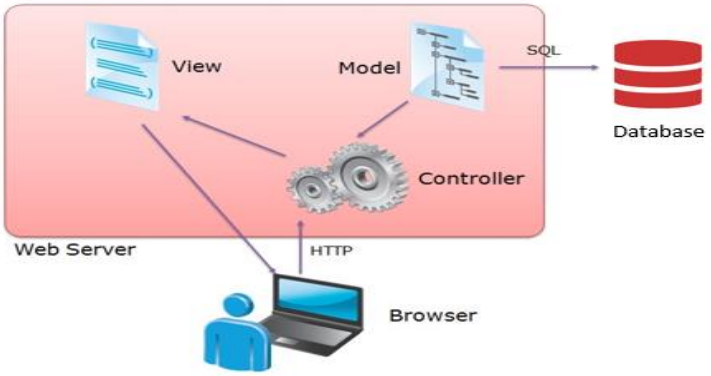
3.1.6 Perangkat Lunak Untuk Pengembang

Daftar perangkat lunak yang digunakan pengembang dapat dilihat pada tabel berikut:

Tabel 3 1 Perangkat Lunak

No	Perangkat Lunak	Fungsi
1.	Microsoft Visio 2013	Digunakan untuk pembuatan perancangan sistem.
2.	Microsoft Visual Studio 2015	Digunakan untuk pembuatan sistem *web service* pada *website.*
3.	IIS Server	Digunakan sebagai *web* server pada aplikasi *website*.
4.	SQL Server Management Studio 2014	Digunakan untuk pengolahan *database* yang akan digunakan dalam aplikasi *website*.

- memanggil *model* untuk melakukan operasi, kemudian mengirimkan data pada *view*. Akhirnya *view* akan menampilkan data tersebut dalam bentuk sebuah halaman html.Arsitektur proses kontrol dapat dilihat pada gambar berikut :



Gambar 3. 1 Arsitektur Proses Kontrol *Model MVC*

3.2 Analisis Kebutuhan

Analisis kebutuhan yang didapat berikut merupakan analisis kebutuhan pada aplikasi :

a. Data Guru untuk melihat *Web api* data guru, standar nilai, ujian, dan mata pelajaran.
b. Data Siswa untuk melihat *Web api* data siswa, nilai.
c. Soal untuk melihat *Web api* isi soal yang di tambahkan guru.
d. Mata Pelajaran untuk melihat *Web api* isi mata pelajaran yang di inputkan guru, dan *Tag* berdasar mata pelajaran.
e. Nilai untuk melihat *Web api* hasil nilai dari ujian siswa.
f. Event Ujian untuk melihat *Web api* dari ujian, nilai siswa, dan soal.

g. Nilai *Tag* By Siswa untuk melihat *Web api* nilai per *Tag* siswa berdasar *Tag*_id dan nama *Tag*.
h. *Tag* untuk melihat *Web api* keseluruhan isi *Tag* dari soal dan mata pelajaran.
i. *Tag* By Siswa untuk melihat *Web api* isi *Tag* berdasar siswanya.

3.2.1 Definisi Penelitian

Pada tahapan ini perlu dilakukan analisis terhadap permasalahan yang dihad*api* dalam pembuatan tugas akhir ini, yaitu bagaimana memahami konsep teknologi *web service* dan menerapkan teknologi *web service* tersebut kesebuah contoh kasus, dimana pada tugas akhir ini contoh kasus yang diambil adalah sistem assesmen dan berbasis *Tag* sebagai pembantu penyusunan strategi pembelajaran. Adapun kebutuhan-kebutuhan dalam pembuatan aplikasi sistem *web service* ini adalah dokumen/data-data penunjang aplikasi " *Pengembangan Web api Pada Sistem Assesmen Dan Berbasis Tag Sebagai Pembantu Penyusunan Strategi Pembelajaran"*, seperti data guru, data siswa, *Tag*, *Tag* siswa, nilai *Tag* setiap siswa, mata pelajaran, soal dan nilai keseluruhan siswa. Untuk dapat menyelesaikan tahap ini hal-hal yang perlu dilakukan adalah :

a. Studi Literatur

Dilakukan studi literatur ini adalah untuk dapat memahami bagaimana membuat sebuah aplikasi *web api* beserta bahasa pendukung pembuatan aplikasi, yaitu *Visual Studio* dan *ASP.NET*. Studi literatur dapat dilakukan dengan cara membaca buku-buku referensi ataupun dengan browsing di internet. Hasil dari pelaksanaan studi literatur ini adalah dengan adanya tutorial tentang pembuatan aplikasi *web api* maka dapat dimanfaatkan untuk mencoba membuat aplikasi-aplikasi *web service* sederhana seperti *web service* operasi matematika (penjumlahan dan lain-lain).

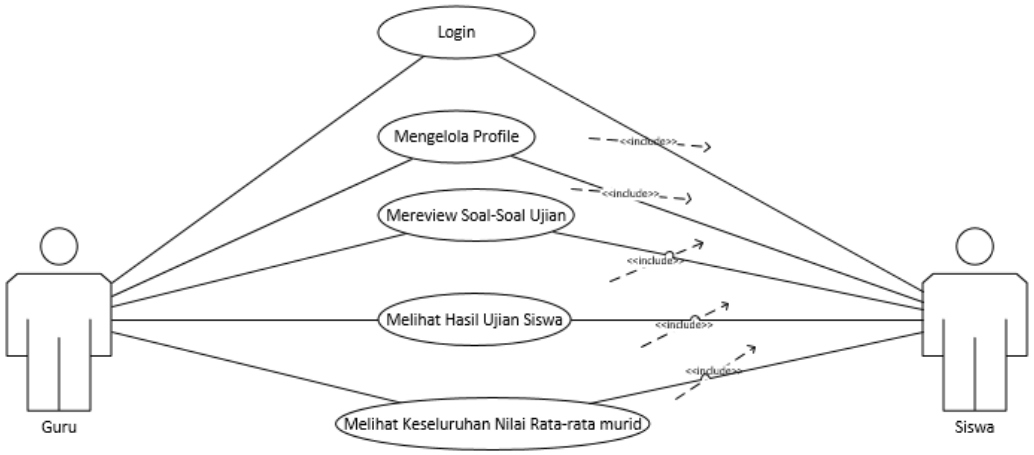
b. Observasi

Dengan mencari aplikasi-aplikasi yang memiliki kemiripan fungsi dan kemudian dilakukan analisa terhadap keunggulan dan kelemahan aplikasi tersebut agar dapat dijadikan sebuah referensi.

3.2.2 Rancangan Sistem

Desain sistem *web* ini meliputi pembuatan seperti di bawah ini :

a. Use Case Diagram*Use Case Diagram* erupakan gambaran graphical dari beberapan peran aktor dan interaksi diantaranya yang memperkenalkan suatu sistem, yang dapat di tunjukkan pada Gambar 3.2.



Gambar 3. 2 *Use Case Diagram*

Penjelasan pada gambar 3.2 *use case diagram* diatas adalah :

1. Gambaran Umum

   Use case ini digunakan oleh aktor untuk memperoleh akses *web api* ke sistem. Login didasarkan pada sebuah id unik yaitu email dari username dan password yang berupa rangkaian karakter.

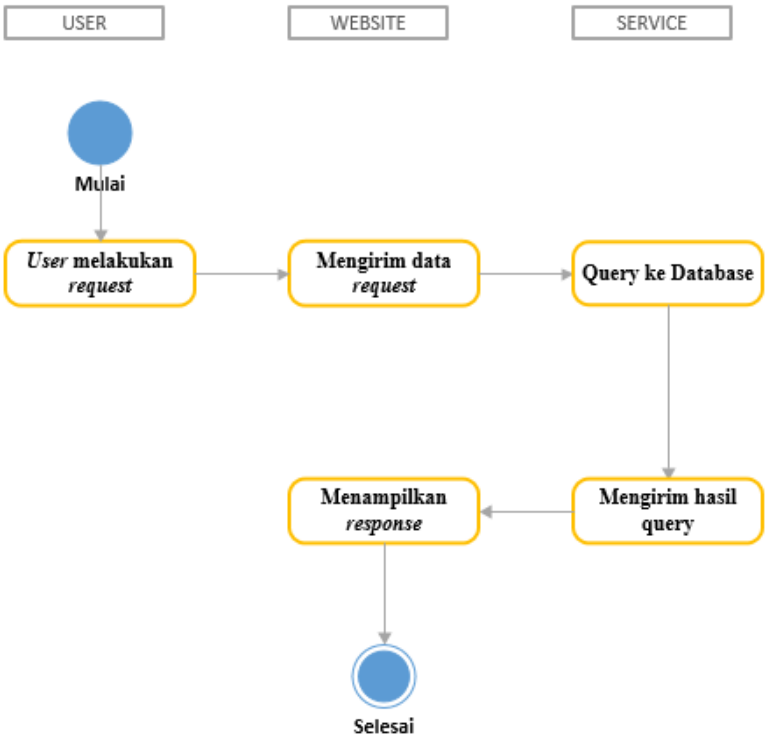
2. Aktor Utama
   - Pengguna
3. Aktor Pendukung
   - Tidak ada
4. Alur Dasar
   a. Use case ini dimulai ketika aktor memilih untuk melakukan login
   b. Sistem menampilkan *request* antarmuka untuk login.
   c. Aktor memasukkan username dan password.
   d. Sistem memeriksa id dan password yang di inputkan actor.
   o E-1 Password atau id user tidak sesuai.
   e. Sistem memberikan akses ke actor.
   f. Use Case ini selesai
5. Alur Alternatif
   - Tidak ada
6. Alur Kesalahan
   - Password atau nama user tidak sesuai

   a. Sistem menampilkan peringatan bahwa id user atau password tidak sesuai

   b. Kembali ke alur dasar langkah ke-c
7. Kondisi Awal
   - Tidak ada
8. Kondisi Akhir

- Aktor memasuki sistem dan dapat menggunakan fungsi-fungsi pada sistem.

Activity Diagram merupakan diagram yang menggambarkan worlflow (aliran kerja) atau aktivitas dari sebuah sistem atau proses. Yang perlu diperhatikan adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang di lakukan aktor.



Gambar 3. 3 *Activity Diagram*

Penjelasan *Activity Diagram* sebagai berikut :

Activity Diagram seperti diatas yang ditunjukkan pada gambar 3.3, dapat terlihat bahwa proses dari setiap aktor yang berada di dalam sistem tersebut adalah Guru dan Siswa yang memiliki hak akses seperti dibawah ini :

Tabel 3 2 *API* Data Guru

***Request* Guru**	***Response Web Service* (*API*)**
1. Guru *request* data guru	=> Menampilkan *response*
2. Guru *request* data grups	=> Menampilkan *response*
3. Guru *request* data Soal	=> Menampilkan *response*
4. Guru *request* data Standard Nilai	=> Menampilkan *response*

Penjelasan pada tabel 3.2 *API* Data Guru adalah :

Pada penjelasan diatas dimana guru dapat melakukan sebuah *request web api* seperti melihat data guru, *grups*, soal, dan standar nilai ke halaman *website.* Kemudian hasil *request* tersebut akan di publikasikan ke *website* tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON.*

Tabel 3 3 *API* Data Siswa

***Request* Siswa**	***Response Web Service* (*API*)**
1. Siswa *request* data siswa	=> Menampilkan *response*
2. Siswa *request* nilai per tag	=> Menampilkan *response*
3. Siswa *request* siswa_id	=> Menampilkan *response*
4. Siswa *request* nama_siswa	=> Menampilkan *response*

Penjelasan pada tabel 3.3 *API* Data Siswa adalah :

Pada penjelasan diatas dimana siswa dapat melakukan sebuah *request web api* seperti melihat data siswa, nilai per *tag*, siswa_id, dan nama_siswa ke halaman *website.* Kemudian hasil *request* tersebut akan di publikasikan ke *website* tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON.*

Tabel 3 4 *API* Mata Pelajaran

***Request* Mata Pelajaran**	***Response Web Service* (*API*)**
1. Mata Pelajaran *request* data mata pelajaran	=> Menampilkan *response*
2. Mata Pelajaran *request* mata pelajaran per ujian	=> Menampilkan *response*
3. Siswa *request* mata pelajaran per data guru	=> Menampilkan *response*
4. Siswa *request* mata pelajaran per *tag*	=> Menampilkan *response*

Penjelasan pada tabel 3.4 *API* Mata Pelajaran adalah :

Pada penjelasan tabel diatas dimana mata pelajaran dapat melakukan sebuah *request* seperti melihat data mata pelajaran, mata pelajaran per ujian, mata pelajaran per data guru , dan mata pelajaran per tag ke halaman *website.* Kemudian hasil *request* tersebut akan di publikasikan ke *website* tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON.*

Tabel 3 5 *API* Nilai

***Request* Nilai**	***Response Web Service* (*API*)**
1. Nilai *request* data keseluruhan nilai	=> Menampilkan *response*
2. Nilai *request* nilai dengan data siswa	=> Menampilkan *response*
3. Nilai *request* nilai per event ujian	=> Menampilkan *response*

Penjelasan pada tabel 3.5 *API* Nilai adalah :

Pada penjelasan tabel diatas dimana nilai dapat melakukan sebuah *request* seperti melihat data keseluruhan nilai, nilai per ujian, dan nilai dengan data siswa ke halaman *website.* Kemudian hasil *request* tersebut akan di publikasikan ke

website tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON*.

Tabel 3 6 *API* Soal

***Request* Soal**	***Response Web Service* (*API*)**
1. Soal *request* data keseluruhan soal	=> Menampilkan *response*
2. Soal *request* soal dengan data guru	=> Menampilkan *response*
3. Soal *request* soal per tag	=> Menampilkan *response*
4. Soal *request* per *event* ujian	=> Menampilkan *response*

Penjelasan pada tabel 3.6 *API* Soal adalah :

Pada penjelasan tabel diatas dimana soal dapat melakukan sebuah *request* seperti melihat data soal, soal dengan data guru, dan soal per *tag*, dan soal per *event* ujian ke halaman *website*. Kemudian hasil *request* tersebut akan di publikasikan ke *website* tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON*.

Tabel 3 7 *API* Standar Nilai

***Request* Standar Nilai**	***Response Web Service* (*API*)**
1. Standar Nilai *request* data keseluruhan standar nilai	=> Menampilkan *response*
2. Standar Nilai *request* standar nilai dengan data guru	=> Menampilkan *response*

Penjelasan pada tabel 3.7 *API* Standar Nilai adalah :

Pada penjelasan tabel diatas dimana standar nilai dapat melakukan sebuah *request* seperti melihat data keseluruhan standar nilai, dan standar nilai dengan data

guru ke halaman *website*. Kemudian hasil *request* tersebut akan di publikasikan ke *website* tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON*.

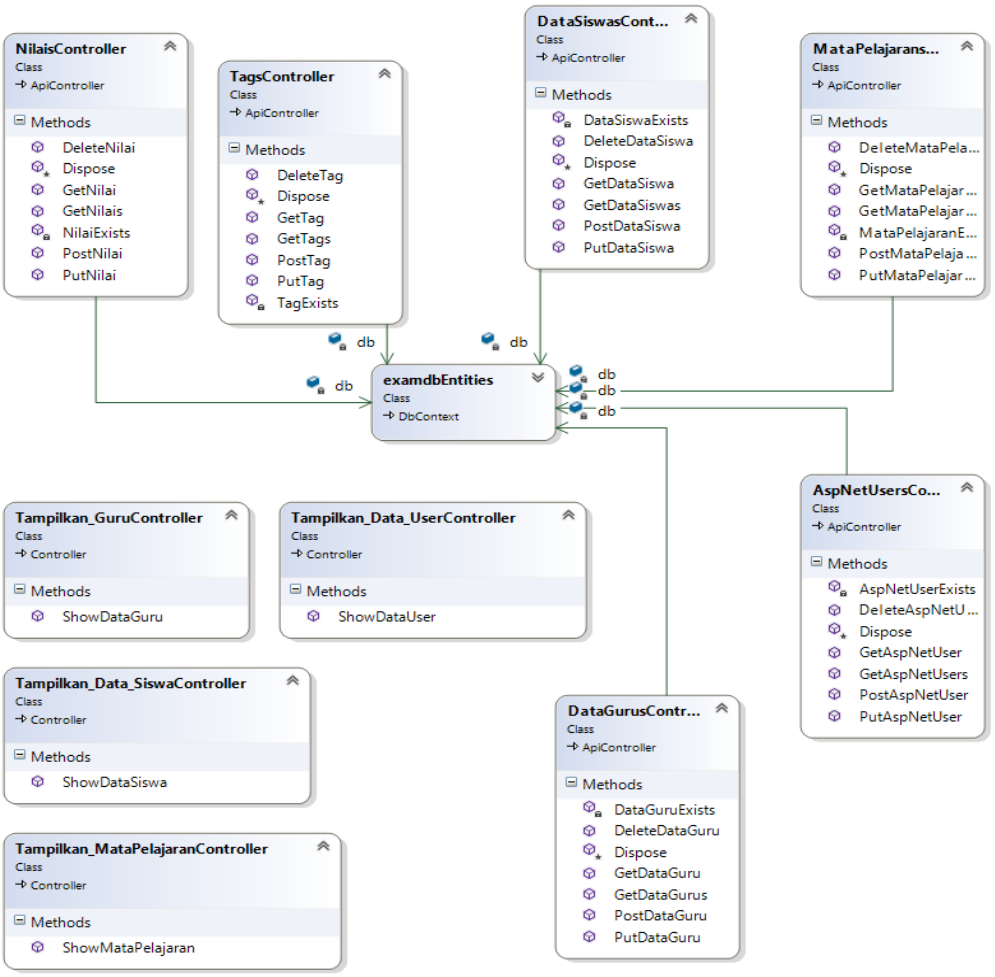
Tabel 3 8 *API Tags*

Request Tags	***Response Web Service* (*API*)**
1. *Tags request* data keseluruhan *tags*	=> Menampilkan *response*
2. *Tags request tags* per mata pelajaran	=> Menampilkan *response*
3. *Tags request tags* per soal	=> Menampilkan *response*

Penjelasan pada tabel 3.8 *API Tags* adalah :

Pada penjelasan tabel diatas dimana *tags* dapat melakukan sebuah *request* seperti melihat data keseluruhan *tags*, *tags* per mata pelajaran, dan *tags* per soal ke halaman *website*. Kemudian hasil *request* tersebut akan di publikasikan ke *website* tadi dan akan menampilkan sebuah *response* dari hasil *request* guru tersebut. Sebagai pengirim pesannya adalah berupa *JSON*.

Class Diagram adalah *model* statis yang menggambarkan struktur dan deskripsi class serta hubungannya antara class. *Class Diagram* mirip *ER-Diagram* pada perancangan database, bedanya pada *ER-Diagram* tdk terdapat operasi/methode t*api* hanya atribut, yang dapat di tunjukkan pada gambar 3.4.



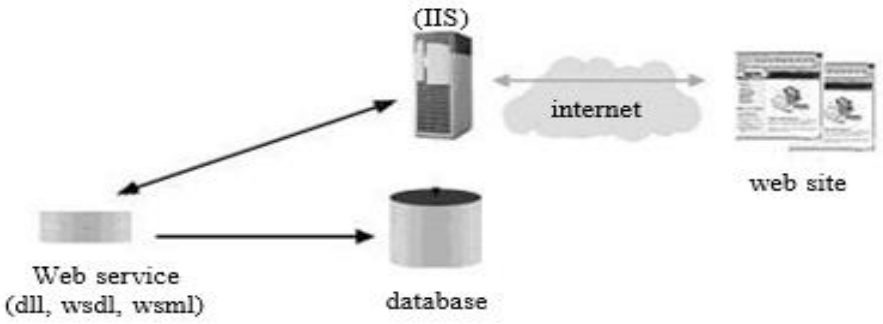
Gambar 3. 4 Deskripsi *Class Diagram*

Penjelasan Gambar 3.4 *Class Diagram* diatas adalah :

Class	Jenis Class	Deskripsi
Nilais	*Controller*	Class yang berisi method-method yang berfungsi melihat nilai siswa rdasar Siswa_Id
Tags	*Controller*	Class yang berisi method-method yang berfungsi melihat MataPelajaran rdasarkan *Tag*_Id

DataSiswas	*Controller*	Class yang berisi method-method yang berfungsi melihat profile, serta dapat melihat nilai berdasar Siswa_Id
MataPelajarans	*Controller*	Class yang berisi method-method yang berfungsi melihat nama mata pelajaran berdasar *Tag* yang di inputkan oleh guru
AspNetUsers	*Controller*	Class yang berisi method-method yang berfungsi melihat data admin, serta data pengguna sesuai role yang tentukan
Da*Tag*urus	*Controller*	Class yang berisi method-method yang berfungsi melihat data guru, mata pelajaran, dan soal-soal yang di inputkan

c. Skema *Web Service*



Gambar 3. 5 Skema *web service*

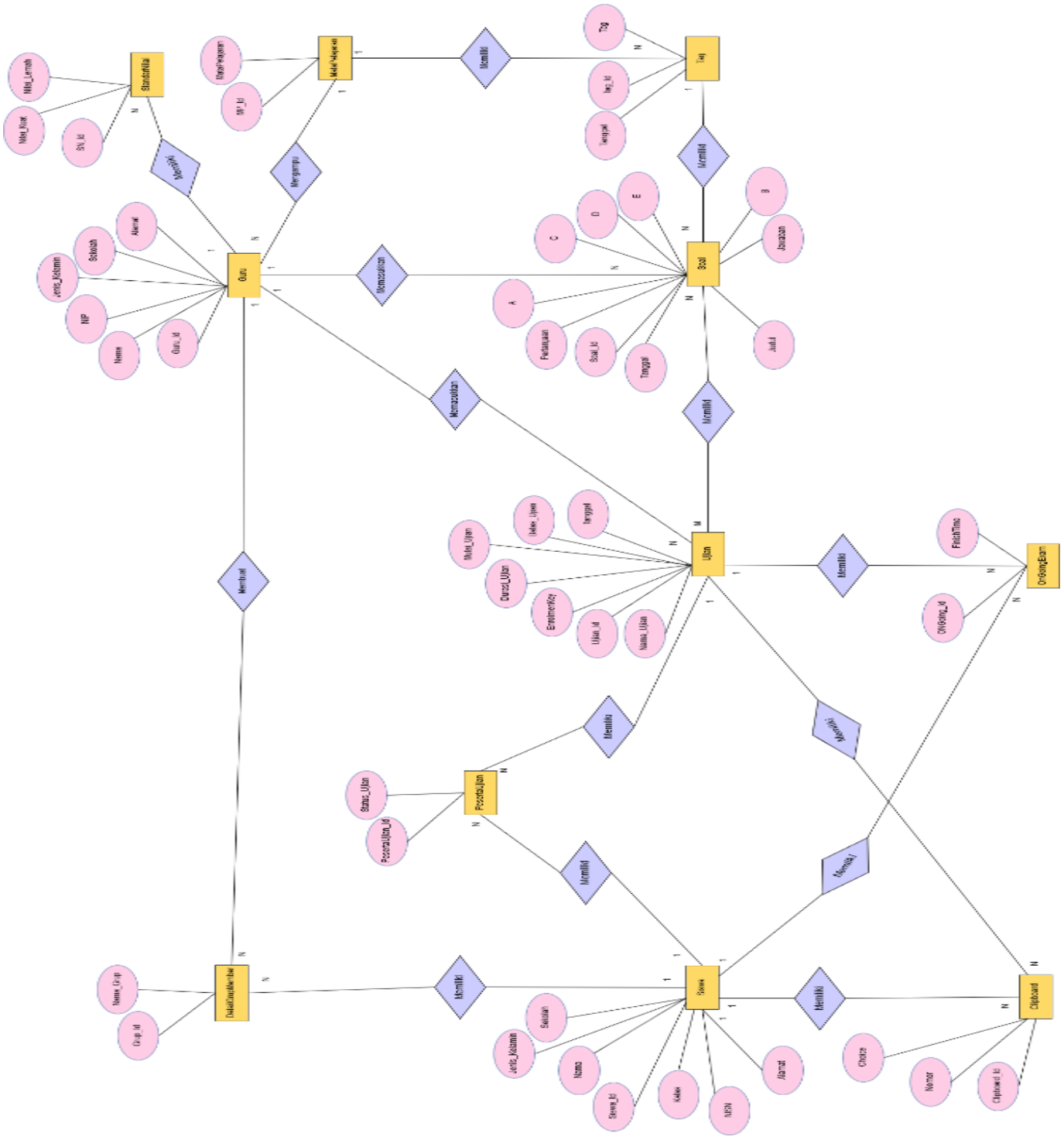
Penjelasan untuk *web service* yang di tunjukkan gambar 3.5 terdiri dari tiga file:

a. DLL
b. WSDL
c. WSML

Dari skema diatas dapat dijelaskan bahwa pertama-tama program client melakukan request yang kemudian Internet Information System menjalankan .NET *Framework*. Lalu .NET *Framework* akan memeriksa apakah format pesan pada request sama dengan format pesan pada file WSDL setelah pengecekan selesai maka diteruskan pada file WSML untuk memetakan operasi yang diminta oleh program client pada object. Setelah operasi selesai maka .NET *Framework* akan memberikan response ke program client. Pembuatan aplikasi *web service* pada tugas akhir ini menggunakan Visual Studio untuk komponen *web service*.

d. Desain Database *ER-Diagram*

Setelah menentukan *entity* dan *attributes* untuk sistem asesmen dan pemetaan hasil asesmen, maka dari semua *entity* tersebut dapat membetuk sebuah ERD diagram. ERD diagram dapat dilihat pada gambar 3.5.



Gambar 3. 6 *ER-Diagram*

Penjelasan *ERD* pada Gambar 3.6 adalah sebagai berikut :

a. Entity Guru mempunyai relasi many to 1 terhadap entity MataPelajaran.
b. Entity Guru mempunyai relasi 1 to many terhadap entity StandarNilai.
c. Entity Guru mempunyai relasi 1 to many terhadap entity Soal.
d. Entity Ujian mempunyai relasi many to many dengan entity Soal.
e. Entity Soal mempunyai relasi many to 1 dengan entity *Tag*.
f. Entity *Tag* mempunyai relasi many to 1 terhadap entity MataPelajaran.
g. Entity Ujian mempunyai relasi many to 1 dengan entity Guru.
h. Entity DetailGrupMember mempunyai relasi many to 1 dengan entity Guru.
i. Entity Clipboard relasi many to 1 dengan entity Ujian.
j. Entity Clipboard relasi many to 1 dengan entity Siswa.
k. Entity OnGoingExam mempunyai relasi many to 1 dengan entity Siswa
l. Entity OnGoingExam mempunyai relasi many to 1 dengan entity Ujian.
m. Entity PesertaUjian mempunyai relasi many to 1 dengan entity Siswa.
n. Entity PesertaUjian mempunyai relasi many to 1 dengan entity Ujian.

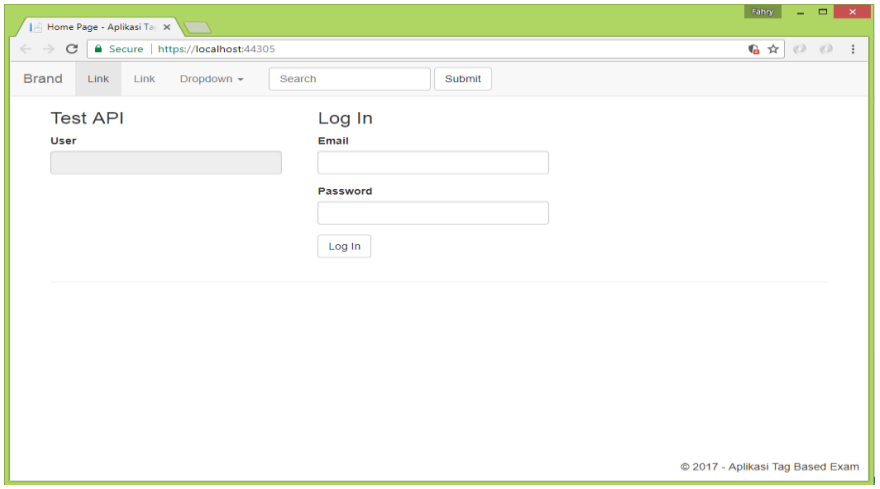
o. Entity Siswa mempunyai relasi 1 to many dengan entity DetailGrupMember..

3.3 Perancangan Antar Muka

Untuk memudahkan proses pembuatan *web api* ini, maka terlebih dahulu membuat rancangan antar muka pada sebuah *website* diantaranya Tampilan Login, Pengujian Test *API*, Tampilan *response* Data Guru, dan Tampilan *response* Data Siswa.

3.3.1 Perancangan Antar Muka Menu Login dan Test *API*

Perancangan Antar Muka Login, Register dan Test *API* adalah Desain Interface dari konten Menu *Web api* yang dapat user gunakan untuk masuk ke halaman selanjutnya sesuai test *API* nya yang sudah tersedia.



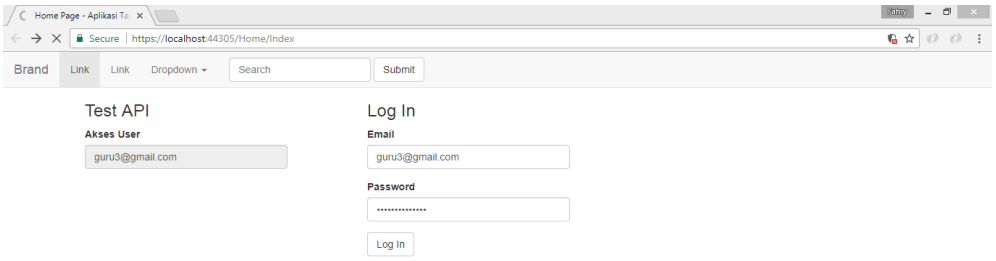
Gambar 3. 7 Tampilan Login dan Test *API*

Pada gambar 3.7 memperlihatkan *login test API* yang digunakan untuk mendapatkan *access_token user*.

```
<div class="col-sm-4">
    <form data-bind="submit: callApi">
        <h3>Test API</h3>
        <div class="form-group">
            <label>Akses User</label>
            <input class="form-control" type="text" readonly data-bind="value: user" />
        </div>
        <div class="form-group error-messages" data-bind="foreach: errors">
            <p data-bind="text: $data" />
        </div>
    </form>
</div>
<div class="col-sm-4">
    <h3>Log In</h3>
    <form data-bind="submit: login">
        <div class="form-group">
            <label>Email</label>
            <input class="form-control" type="text" data-bind="value: loginEmail" placeholder="Masukkan
        </div>
        <div class="form-group">
            <label>Password</label>
            <input class="form-control" type="password" data-bind="value: loginPassword" placeholder="Ma
        </div>
        <div class="form-group">
            <button type="submit" class="btn btn-default">Log In</button>
        </div>
    </form>
</div>
```

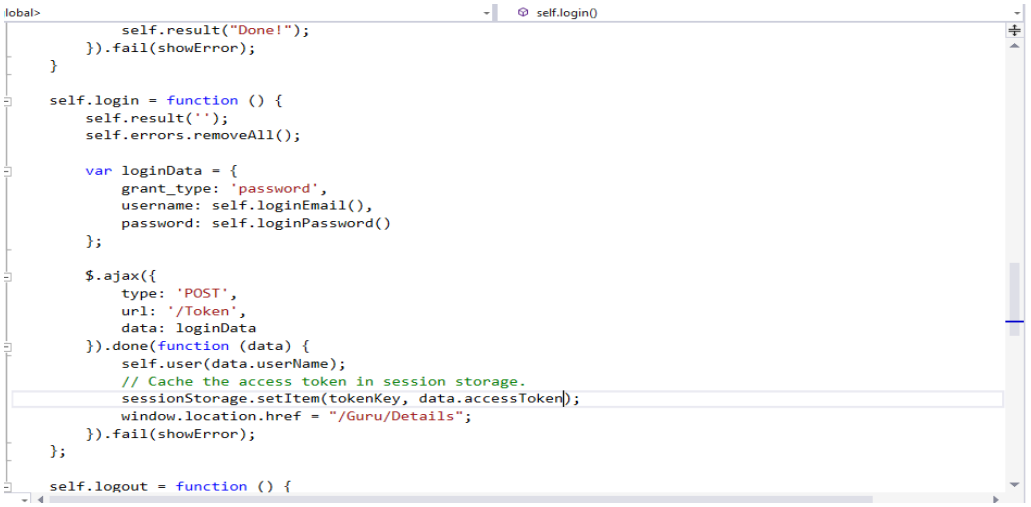
Gambar 3. 8 *Coding interface* pada halaman *Login*

Pada gambar 3.8 memperlihatkan *coding* yang digunakan untuk menampilkan *form login*. Pada awal *coding* terdapat *input value user* dan data ajax untuk menampilkan pesan bahwa *login* dengan *email* dan *password* sesuai role data *API* pengguna.



Gambar 3. 9 *Response* data *API* Pengguna

Pada gambar 3.9 memperlihatkan *coding* yang digunakan untuk menampilkan *response API input value user* bahwa *login* dengan *email* dan *password* sesuai role data *API* pengguna.



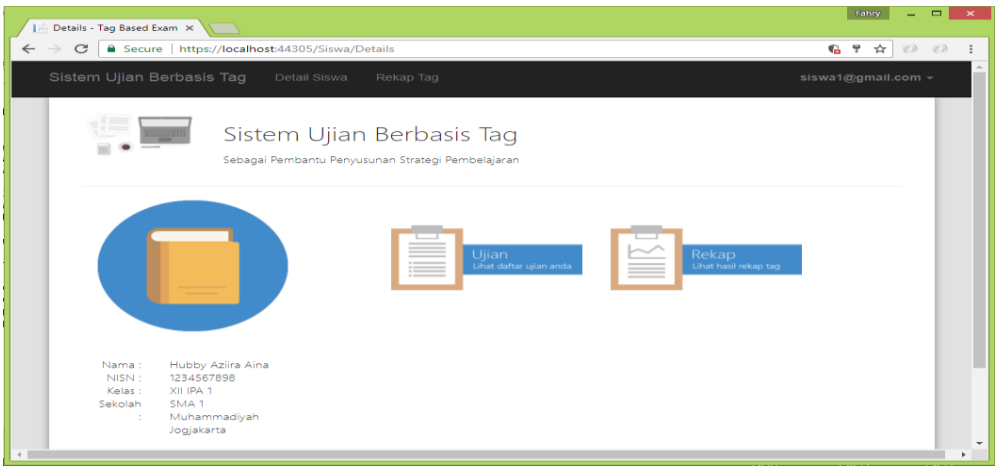
```
lobal>                                                    self.login()
            self.result("Done!");
        }).fail(showError);
    }

    self.login = function () {
        self.result('');
        self.errors.removeAll();

        var loginData = {
            grant_type: 'password',
            username: self.loginEmail(),
            password: self.loginPassword()
        };

        $.ajax({
            type: 'POST',
            url: '/Token',
            data: loginData
        }).done(function (data) {
            self.user(data.userName);
            // Cache the access token in session storage.
            sessionStorage.setItem(tokenKey, data.accessToken);
            window.location.href = "/Guru/Details";
        }).fail(showError);
    };

    self.logout = function () {
```

Gambar 3. 10 *Coding Access Token Users*

Pada gambar 3.10 memperlihatkan *coding* yang digunakan untuk menampilkan *access_token* para *users* yang dapat melihat isi data dengan melalui *access_token* tersebut.

3.3.2 Perancangan Antar Muka Tampilan Data Siswa

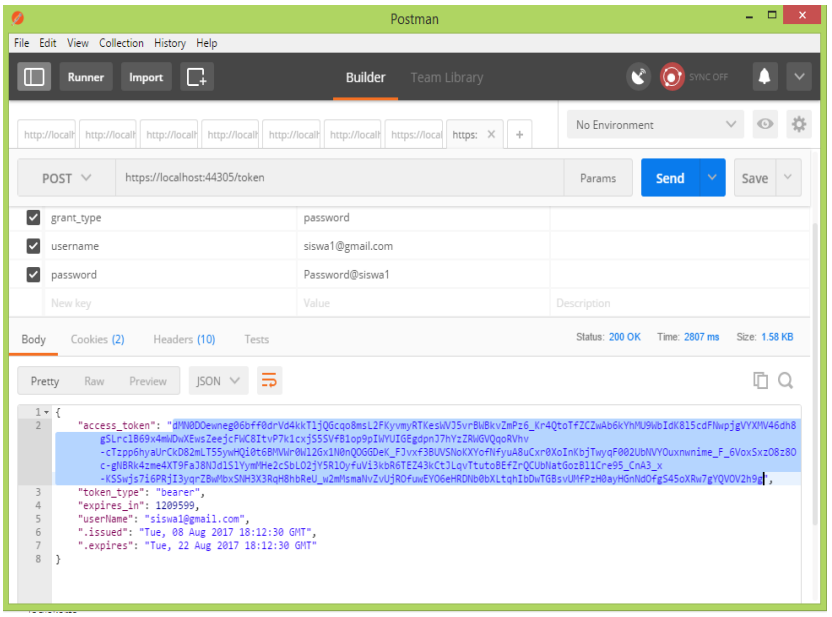
Perancangan Antar Muka Tampilan Data Siswa adalah Desain Interface dari konten Tampilan Data Siswa *Web api* yang dapat user gunakan untuk melihat ujian siswa yang telah di kerjakan.



Gambar 3. 11 Tampilan Siswa

Penjelasan yang ditunjukkan pada gambar 3.11 adalah :

Pada gambar diatas ialah sebuah halaman atau tampilan dari data siswa ketika melakukan proses *login.*



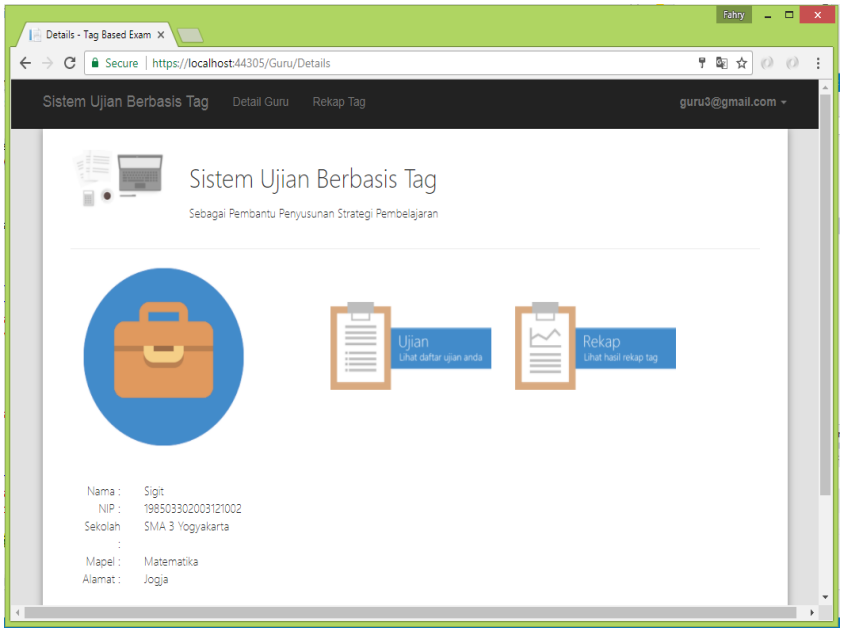
Gambar 3. 12 *Request access_token Siswa*

Penjelasan yang ditunjukkan pada gambar 3.12 adalah :

Dimana seorang siswa melakukan login untuk mendapatkan sebuah *access_token* untuk melihat data *api* data siswa.

3.3.3 Perancangan Antar Muka Tampilan Data Guru

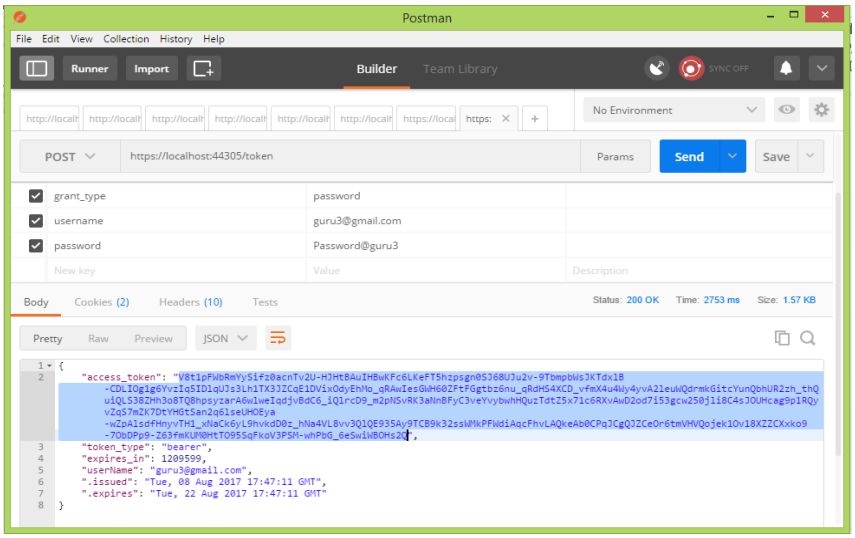
Perancangan Antar Muka Tampilan Data Guru adalah Desain Interface dari konten Tampilan Guru *Web api* yang dapat user gunakan untuk melihat data ujian yang telah dibuat , dan rekapan para siswa.



Gambar 3. 13 Tampilan Guru

Penjelasan yang ditunjukkan pada gambar 3.13 adalah :

Pada gambar diatas ialah sebuah halaman atau tampilan dari data guru ketika melakukan proses *login.*



Gambar 3. 14 *Request access_token Guru*

Penjelasan yang ditunjukkan pada gambar 3.14 adalah :

Dimana seorang guru melakukan login untuk mendapatkan sebuah *access_token* untuk melihat data *api* data guru.