

**LAPORAN**  
**PROJECT WORK**

**APLIKASI MANAJEMEN DAN PENCETAKAN RAPOR**

Diajukan untuk memenuhi salah satu syarat kelulusan dari

SMK Negeri 1 Cimahi



NAMA : HARIZ SUFYAN MUNAWAR  
NO. INDUK SISWA : 181113834  
TINGKAT : III (TIGA)  
PAKET KEAHLIAN : REKAYASA PERANGKAT LUNAK

**SEKOLAH MENENGAH KEJURUAN NEGERI 1**

**CIMAHI**

**2020**

LEMBAR PENGESAHAN PIHAK SEKOLAH

**APLIKASI MANAJEMEN DAN PENCETAKAN RAPOR  
(REI)**

Laporan ini telah disetujui oleh:

Ketua Jurusan

Pembimbing

AGUS RAHMAWAN, S.T.

MONA MARANTIKA, S.Pd.

---

NIP. 197512022006041010

---

NUPTK. 5843765666130142

MENGETAHUI:

Kepala SMK Negeri 1 Cimahi

Drs. Daud Saleh, M.M

---

NIP. 196307181989021001

## KATA PENGANTAR

Puji syukur dipanjatkan kepada kehadiran Tuhan Yang Maha Esa karena dengan rahmat dan karunia-Nya penulis dapat menyelesaikan kegiatan Project Work yang merupakan suatu kewajiban siswa dan siswi tingkat III (tiga) jurusan Rekayasa Perangkat Lunak di SMK Negeri 1 Cimahi yang dilakukan selama kurang lebih 3 bulan.

Setelah melaksanakan kegiatan Project Work, siswa dan siswi tingkat III (tiga) jurusan Rekayasa Perangkat Lunak SMK Negeri 1 Cimahi menyajikan sebuah karya tulis sebagai sebuah laporan tentang apa yang telah dikerjakan dan dipelajari. Pada kesempatan ini, penulis telah menyelesaikan sebuah karya tulis dengan judul “APLIKASI MANAJEMEN DAN PENCETAKAN RAPOR SISWA”.

Dapat terselesaikannya karya tulis ini tidak lepas dari dukungan dan dorongan dari berbagai pihak, oleh karena itu penulis ingin memberikan rasa terima kasih yang sedalam-dalamnya kepada semua pihak yang telah memberikan dukungan dan dorongannya, diantaranya:

- 1 Allah SWT, yang karena rahmat dan izin-Nya penulis dapat menyelesaikan kegiatan Project Work ini.
- 2 Orang tua serta keluarga dari penulis yang mendukung penulis dalam bentuk material maupun non-material.
- 3 Drs. Daud Saleh, M.M selaku kepala di SMK Negeri 1 Cimahi.
- 4 Bapak Agus Rahmawan, S.T. selaku kepala di jurusan Rekayasa Perangkat Lunak.
- 5 Ibu Mona Marantika, S.Pd. selaku pembimbing penulis dalam kegiatan Project Work ini.

- 6 Bapak Yuli Pamungkas, S.P., selaku Wali Kelas tingkat III (tiga) Rekayasa Perangkat Lunak A.
- 7 Bapak dan Ibu Guru selaku pengajar dari jurusan Rekayasa Perangkat Lunak.
- 8 Teman-teman kelas serta jurusan Rekayasa Perangkat Lunak.
- 9 Dan semua pihak yang telah ikut serta dalam melancarkan kegiatan Project Work ini.

Penulis menyadari bahwa karya tulis ini masih memiliki banyak kekurangan, baik dalam aspek penulisan maupun penyampaian. Oleh karena itu penulis mengharapkan kritik dan saran dari semua pihak yang telah menyempatkan waktunya untuk membaca karya tulis ini.

Akhir kata, penulis berharap semoga karya tulis ini dapat menjadi sebuah pembelajaran berharga baik bagi penulis sendiri maupun bagi siapapun yang membacanya.

Cimahi, Januari 2021

Penulis,

Hariz Sufyan Munawar

## DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI.....	iii
DAFTAR GAMBAR.....	x
DAFTAR TABEL.....	xiv
BAB I PENDAHULUAN.....	1
1.1. Latar Belakang Masalah.....	1
1.2. Tujuan.....	2
1.3. Pembatasan Masalah.....	2
1.4. Sistematika Pembahasan.....	3
BAB II LANDASAN TEORI.....	4
2.1. Website.....	4
2.1.1. Website Statis.....	4
2.1.2. Website Dinamis.....	4
2.2. HTML.....	5
2.3. CSS.....	5
2.4. Bootstrap.....	6
2.5. JavaScript.....	6
2.5.1. JQuery.....	7
2.5.2. AJaX.....	7
2.6. Python.....	7

2.6.1. Penulisan Kode.....	8
2.6.2. Komentar.....	8
2.6.3. Variabel.....	9
2.6.4. Tipe Data.....	10
2.6.5. Operator.....	12
2.6.6. Fungsi.....	14
2.6.7. Struktur Kontrol.....	15
2.6.8. Pip.....	17
2.7. Django.....	17
2.7.1. Instalasi.....	19
2.7.2. Server Lokal.....	19
2.7.3. Konsep MTV.....	21
2.7.3.1. Model.....	21
2.7.3.2. Template.....	21
2.7.3.3. View.....	21
2.8. Database.....	21
2.8.1. SQL.....	22
2.8.2. DBMS.....	24
2.8.2.1. SQLite.....	24
2.8.2.2. MongoDB.....	24
2.9. Web Server.....	25
2.10. Flowmap.....	25

2.11. Data Flow Diagram.....	27
2.12. Entity Relation Diagram.....	28
BAB III ANALISA DAN PERANCANGAN.....	30
3.1. Analisa Aplikasi.....	30
3.1.1. Analisa Sistem Pengguna.....	30
3.1.2. Analisa Kebutuhan Perangkat Sistem.....	32
3.1.2.1. Kebutuhan Komputer Server.....	32
3.1.2.2. Kebutuhan Komputer Client.....	33
3.2. Rancangan Sistem.....	33
3.2.1. Pembuatan Dokumen Rapor Siswa.....	34
3.2.2. Pengaturan Akun Walikelas dan Staf Tata Usaha.....	34
3.2.3. Input Data Siswa.....	34
3.2.4. Input Data Nilai dan Absensi.....	34
3.2.5. Pengaturan Kelas.....	34
3.2.6. Pengaturan Mata Pelajaran.....	35
3.2.7. Pergantian Semester.....	35
3.2.8. Pengubahan Data Sekolah.....	35
3.3. Data Flow Diagram Aplikasi.....	35
3.3.1. DFD Level 0.....	35
3.3.2. DFD Level 1.....	36
3.3.3. DFD Level 2.....	37
3.3.3.1. DFD Level 2 Update Data Sekolah.....	37

3.3.3.2. DFD Level 2 Impor Data Siswa.....	38
3.3.3.3. DFD Level 2 Input Nilai Siswa.....	38
3.3.3.4. DFD Level 2 Pencetakan Rapor Digital.....	38
3.4. Flowmap.....	39
3.4.1. Flowmap Sistem Berjalan.....	40
3.4.2. Flowmap Aplikasi.....	41
3.5. Rancangan Database.....	42
3.5.1. Entity Relationship Diagram.....	42
3.6. Rancangan Layout Halaman.....	42
3.6.1. Rancangan Tata Letak Halaman Login.....	43
3.6.2. Rancangan Tata Letak Halaman Panel.....	43
BAB IV IMPLEMENTASI DAN PENGUJIAN.....	45
4.1. Implementasi.....	45
4.1.1. Desain Front-End / Tampilan.....	45
4.1.1.1. Halaman Login.....	45
4.1.1.2. Halaman Dashboard.....	46
4.1.1.3. Halaman Informasi Sekolah.....	46
4.1.1.4. Halaman Profil Pribadi.....	47
4.1.1.5. Halaman Pengaturan Semester.....	47
4.1.1.6. Halaman Daftar Guru.....	48
4.1.1.7. Halaman Buat Akun Guru.....	49
4.1.1.8. Halaman Detail Guru.....	49



4.1.1.9. Halaman Daftar Siswa.....	50
4.1.1.10. Halaman Detail Siswa.....	50
4.1.1.11. Halaman Nilai Akademik Siswa.....	51
4.1.1.12. Halaman Nilai Ekskul Siswa.....	51
4.1.1.13. Halaman Absensi Siswa.....	52
4.1.1.14. Halaman Edit Profil Siswa.....	52
4.1.1.15. Halaman Daftar Jurusan.....	53
4.1.1.16. Halaman Daftar Kelas.....	53
4.1.1.17. Halaman Detail Kelas.....	54
4.1.1.18. Halaman Pengaturan Walikelas.....	54
4.1.1.19. Halaman Pengaturan Anggota Kelas.....	55
4.1.1.20. Halaman Pengaturan Mapel Kelas.....	56
4.1.1.21. Halaman Cetak Rapor.....	56
4.1.1.22. Halaman Daftar Mata Pelajaran.....	57
4.1.1.23. Halaman Detail Mata Pelajaran.....	58
4.1.1.24. Halaman Daftar Ekskul.....	58
4.1.2. Struktur Tabel.....	59
4.1.2.1. Tabel Sekolah.....	59
4.1.2.2. Tabel Guru.....	60
4.1.2.3. Tabel Semester.....	62
4.1.2.4. Tabel Jurusan.....	62
4.1.2.5. Tabel Mata Pelajaran.....	63

4.1.2.6. Tabel KKM.....	64
4.1.2.7. Tabel Ekskul.....	64
4.1.2.8. Tabel Kelas.....	65
4.1.2.9. Tabel Siswa.....	66
4.1.2.10. Tabel Absensi.....	67
4.1.2.11. Tabel Nilai Mata Pelajaran.....	67
4.1.2.12. Tabel Nilai Ekskul.....	68
4.1.2.13. Tabel Rapor.....	69
4.1.3. Relasi Tabel.....	70
4.2. Pengujian.....	71
4.2.1. Login.....	71
4.2.2. Penggantian Password.....	74
4.2.3. Manajemen Semester.....	75
4.2.4. Penambahan Guru.....	78
4.2.5. Manajemen Jurusan.....	79
4.2.6. Manajemen Kelas.....	82
4.2.7. Manajemen Mata Pelajaran dan KKM.....	84
4.2.9. Manajemen Siswa.....	86
4.2.10. Cetak Rapor.....	89
BAB V PENUTUP.....	92
5.1. Kesimpulan.....	92
5.2. Saran.....	92

DAFTAR PUSTAKA.....	93
---------------------	----

## DAFTAR GAMBAR

Gambar 2.1 Penulisan Kode Python.....	8
Gambar 2.2 Baris Komentar Python.....	9
Gambar 2.3 Deklarasi Variabel Python.....	10
Gambar 2.4 Tipe Data Dalam Python.....	12
Gambar 2.5 Penulisan Fungsi Dalam Python.....	15
Gambar 2.6 Fungsi Menggunakan Argumen.....	15
Gambar 2.7 Struktur Kontrol If.....	16
Gambar 2.8 Struktur Kontrol While.....	17
Gambar 2. 9 Struktur Kontrol For.....	17
Gambar 2.10 Instalasi Django.....	19
Gambar 2.11 Cek Versi Django.....	19
Gambar 2.12 Memulai Projek Django.....	19
Gambar 2.13 Memulai Server Lokal.....	20
Gambar 2.14 Halaman Pertama Django.....	20
Gambar 2.15 Simbol-Simbol DFD.....	28
Gambar 3.1 DFD Level 0.....	35
Gambar 3.2 DFD Level 1.....	36
Gambar 3.3 DFD Level 2 Update Data Sekolah.....	37
Gambar 3.4 DFD Level 2 Impor Data Siswa.....	38
Gambar 3.5 DFD Level 2 Input Nilai Siswa.....	38

Gambar 3.6 DFD Level 2 Pencetakan Rapor Digital.....	39
Gambar 3.7 Flowmap Sistem Berjalan.....	40
Gambar 3.8 Flowmap Aplikasi.....	41
Gambar 3.9 Entity Relation Diagram.....	42
Gambar 3.10 Rancangan Tata Letak Halaman Login.....	43
Gambar 3.11 Rancangan Tata Letak Halaman Panel.....	44
Gambar 4.1 Tampilan Halaman Login.....	45
Gambar 4.2 Halaman Dashboard.....	46
Gambar 4.3 Halaman Informasi Sekolah.....	46
Gambar 4.4 Halaman Profil Pribadi.....	47
Gambar 4.5 Halaman Pengaturan Semester.....	48
Gambar 4.6 Halaman Daftar Guru.....	48
Gambar 4.7 Halaman Buat Akun Guru.....	49
Gambar 4.8 Halaman Detail Guru.....	49
Gambar 4.9 Halaman Daftar Siswa.....	50
Gambar 4.10 Halaman Detail Siswa.....	50
Gambar 4.11 Halaman Nilai Akademik Siswa.....	51
Gambar 4.12 Halaman Nilai Ekskul Siswa.....	51
Gambar 4.13 Halaman Absensi Siswa.....	52
Gambar 4.14 Halaman Edit Profil Siswa.....	52
Gambar 4.15 Halaman Daftar Jurusan.....	53
Gambar 4.16 Halaman Daftar Kelas.....	54

Gambar 4.17 Halaman Detail Kelas.....	54
Gambar 4.18 Halaman Pengaturan Walikelas.....	55
Gambar 4.19 Halaman Pengaturan Anggota Kelas.....	55
Gambar 4.20 Halaman Pengaturan Mapel Kelas.....	56
Gambar 4.21 Halaman Cetak Rapor.....	57
Gambar 4.22 Halaman Daftar Mata Pelajaran.....	57
Gambar 4.23 Halaman Detail Mata Pelajaran.....	58
Gambar 4.24 Halaman Daftar Ekskul.....	59
Gambar 4.25 Relasi Tabel.....	70
Gambar 4.26 Form Login Saat NIP Tidak Terdaftar.....	72
Gambar 4.27 Form Login Saat NIP Sudah Terdaftar Tetapi Password Salah.....	73
Gambar 4.28 Navigation Bar Aplikasi Saat User Berhasil Login.....	74
Gambar 4.29 User Gagal Mengubah Password.....	74
Gambar 4.30 User Gagal Mengubah Password.....	75
Gambar 4.31 User Berhasil Mengubah Password.....	75
Gambar 4.32 Pembuatan Tahun Pelajaran Gagal.....	76
Gambar 4.33 Pembuatan Tahun Pelajaran Berhasil.....	76
Gambar 4.34 Penggantian Semester Aktif.....	77
Gambar 4.35 Penghapusan Semester Gagal.....	78
Gambar 4.36 Pembuatan Akun Guru Berhasil.....	79
Gambar 4.37 Pembuatan Akun Guru Gagal.....	79
Gambar 4.38 Pembuatan Jurusan Gagal.....	80

Gambar 4.39 Pembuatan Jurusan Berhasil.....	80
Gambar 4.40 Penghapusan Jurusan Berhasil.....	81
Gambar 4.41 Penghapusan Jurusan Gagal.....	82
Gambar 4.42 Pembuatan Kelas Berhasil.....	83
Gambar 4.43 Pembuatan Kelas Gagal.....	84
Gambar 4.44 Pembuatan Mata Pelajaran Gagal.....	85
Gambar 4.45 Pengubahan Nilai KKM Gagal.....	85
Gambar 4.46 Pembuatan Ekskul Gagal.....	86
Gambar 4.47 Pembuatan Siswa Berhasil.....	86
Gambar 4.48 Pembuatan Siswa Gagal.....	87
Gambar 4.49 Proses Impor Data Siswa Berhasil.....	87
Gambar 4.50 Proses Ekspor Data Siswa Gagal.....	88
Gambar 4.51 Ekspor Data Siswa Berhasil.....	88
Gambar 4.52 Pratinjau Rapor Digital.....	89
Gambar 4.53 Unduh Rapor Digital.....	90
Gambar 4.54 Unduh Rapor Digital Per Kelas.....	91

## DAFTAR TABEL

Tabel 2.1 Operator Aritmetika.....	12
Tabel 2.2 Operator Perbandingan.....	13
Tabel 2.3 Operator Assignment.....	13
Tabel 2.4 Operator Logika.....	14
Tabel 2.5 Operator Keanggotaan.....	14
Tabel 2.6 Simbol-Simbol Flowchart.....	25
Tabel 2.7 Simbol-Simbol ERD.....	29
Tabel 3.1 Analisa Sistem Pengguna.....	30
Tabel 4.1 Tabel Sekolah.....	59
Tabel 4.2 Tabel Guru.....	60
Tabel 4.3 Tabel Semester.....	62
Tabel 4.4 Tabel Jurusan.....	62
Tabel 4.5 Tabel Mata Pelajaran.....	63
Tabel 4.6 Tabel KKM.....	64
Tabel 4.7 Tabel Ekskul.....	64
Tabel 4.8 Tabel Kelas.....	65
Tabel 4.9 Tabel Siswa.....	66
Tabel 4.10 Tabel Absensi.....	67
Tabel 4.11 Tabel Nilai Mata Pelajaran.....	67
Tabel 4.12 Tabel Nilai Ekskul.....	68



Tabel 4.13 Tabel Rapor.....	69
Tabel 4.14 Pengujian Sistem Login.....	71
Tabel 4.15 Pengujian Sistem Penggantian Password.....	74
Tabel 4.16 Pengujian Sistem Manajemen Semester.....	75
Tabel 4.17 Pengujian Penambahan Guru.....	78
Tabel 4.18 Pengujian Manajemen Jurusan.....	79
Tabel 4.19 Pengujian Manajemen Kelas.....	82
Tabel 4.20 Pengujian Manajemen Mata Pelajaran dan KKM.....	84
Tabel 4.21 Pengujian Manajemen Ekskul.....	85
Tabel 4.22 Pengujian Manajemen Siswa.....	86
Tabel 4.23 Pengujian Pencetakan Rapor.....	89

# **BAB I**

## **PENDAHULUAN**

### **1.1. Latar Belakang Masalah**

Perkembangan teknologi yang semakin canggih telah memberikan pengaruh yang sangat besar bagi dunia teknologi informasi. Berbagai macam data yang dahulu harus ditulis di media fisik sudah bisa dialihkan ke media digital agar lebih mudah diolah dan diorganisir. Pengolahan data fisik secara konvensional (ditulis di atas kertas) terkadang menyulitkan, memakan waktu lama, dan juga memberikan peluang yang besar akan terjadinya sebuah *human error*.

Misalnya dalam pencatatan nilai siswa, jika ditulis di atas kertas, maka data-data yang tertulis di atasnya akan sangat sulit untuk disalin, dan melakukan kalkulasi pun akan memakan waktu yang cukup lama. Di era komputasi ini, proses pencatatan ke media digital yang dibantu oleh sebuah aplikasi dapat jauh lebih efektif dan efisien.

Contoh lain adalah saat pembagian rapor. Jika kita melakukan pembagian rapor secara konvensional maka jumlah tenaga, sumber daya dan waktu yang diperlukan akan sangat besar. Misalnya seorang walikelas harus melakukan tanda tangan untuk setiap rapor siswa, tetapi jika kita menggunakan sebuah aplikasi untuk melakukan semua tanda tangan itu secara sekaligus maka jumlah tenaga, sumber daya, dan waktu yang diperlukan bisa kita minimalisir menjadi sangat kecil.

Dalam upaya mengkomputerisasi pengolahan dan penyimpanan data nilai siswa, maka dibuat aplikasi untuk menyimpan data tersebut dalam sebuah sistem *database* dan menggunakan aplikasi berbasis web sebagai antarmuka pengolahan datanya. Dengan demikian, pengguna (walikelas dan staf tata usaha) dapat mengakses aplikasi dan data yang disimpan di suatu *server*, serta data dapat

disimpan dan diolah secara *realtime*. Pengguna juga dapat mencetak rapor dalam bentuk *Portable Document Format* (PDF) sehingga mempermudah proses penandatanganan.

## **1.2. Tujuan**

Tujuan dibuatnya aplikasi ini adalah:

1. Memudahkan pengolahan data nilai siswa.
2. Menyediakan sistem basis data untuk data siswa, nilai, absensi, kelas, jurusan dan sebagainya.
3. Membuat penyimpanan data sekolah menjadi terpusat dan terorganisir.
4. Mencetak rapor dalam bentuk digital (PDF).

## **1.3. Pembatasan Masalah**

Untuk membuat pembahasan menjadi rinci dan tepat sasaran, maka pembahasan akan dibatasi sesuai judul dan apa yang dipelajari. Batasan masalah yang penulis buat meliputi:

1. Sistem aplikasi ini dibuat menggunakan Python web *framework* bernama “Django”.
2. Tampilan aplikasi ini dibuat menggunakan bantuan dari *cascading style sheets* (CSS) *framework* bernama “Bootstrap”.
3. Pada sistem pencetakan rapor menjadi PDF, aplikasi ini dibantu oleh sebuah *package* bernama “Weasyprint”
4. Untuk mengolah dan mengelola data yang disimpan, aplikasi ini menggunakan SQLite sebagai *database management system* (DBMS) pada tahap pengembangan dan MongoDB pada tahap produksi.
5. Aplikasi ini bisa diakses oleh tiga jenis pengguna yaitu walikelas, staf tata usaha dan juga admin.

6. Aplikasi ini dirancang untuk mengimplementasikan pencatatan dan pencetakan rapor untuk sekolah mulai dari tingkat SD sampai SMA atau SMK.
7. Rapor yang dicetak adalah rapor per-semester.

#### **1.4. Sistematika Pembahasan**

Laporan ini terdiri atas beberapa bab:

1. Bab I Pendahuluan, berisi latar belakang masalah, tujuan dari pemilihan judul, pembatasan masalah, dan sistematika pembahasan.
2. Bab II Landasan Teori, berisi teori-teori yang melandasi masalah atau judul yang dibahas.
3. Bab III Analisa dan Perancangan, berisi tentang perancangan alur sistem dalam bentuk *flowmap* dan *data flow diagram* serta struktur basis data yang digunakan.
4. Bab IV Pengujian dan Implementasi, berisi hasil implementasi dan pengujian dari rancangan yang sudah dibuat di bab sebelumnya.
5. Bab V Penutup, berisi kesimpulan dari hasil analisis / rincian pada bab III dan relevansinya dengan teori - teori pada bab II serta saran - saran yang bersifat solusi dan membangun terhadap masalah yang dibahas dalam laporan.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1. Website**

*Website* atau dalam bahasa Indonesia situs web, adalah sekumpulan halaman web yang berada pada satu situs yang sama. Sebuah *website* biasanya diidentifikasi dengan nama *domain* umum dan diterbitkan di setidaknya satu web *server* yang bisa diakses melalui jaringan seperti internet atau jaringan lokal melalui alamat yang biasa disebut dengan *Uniform Resource Locator* (URL).

Halaman web sendiri adalah berkas komputer berisi teks dan format instruksi berbasis *Hypertext Markup Language* (HTML). Halaman web ini diakses dengan protokol komunikasi jaringan bernama *Hypertext Transfer Protocol* (HTTP) menggunakan semacam aplikasi seperti peramban web, yang menerjemahkan instruksi-instruksi pada berkas teks tersebut menjadi tampilan sebuah halaman web.

##### **2.1.1. Website Statis**

*Website* statis adalah sebuah *website* yang kontennya statis / tidak berubah-ubah. Sekali dibuat dan *online* di internet, pada umumnya *website* tersebut tidak dapat diubah kecuali diubah secara manual melalui pengubahan bahasa pemrograman *website* tersebut. Oleh karena itu, terjadinya interaksi pun jarang sekali, sehingga dapat dikatakan seperti brosur *online* karena informasi yang diberikan juga terbatas. *Website* statis biasanya hanya tersusun dari HTML, CSS, dan *JavaScript* (JS).

##### **2.1.2. Website Dinamis**

Berbeda dengan *website* statis yang halaman-halamannya tidak berubah selama *source code*-nya tidak diganti oleh siapapun, isi halaman *website* dinamis

baru “ditulis” ketika diminta, membuat isinya dapat berubah-ubah pada waktu yang berbeda, atau ketika diakses oleh orang yang berbeda, dan sebagainya. *Website* seperti ini umumnya digunakan untuk berbagai kebutuhan web seperti, toko *online*, forum, sosial media, web sekolah, dan lain-lain.

## 2.2. HTML

HTML atau *Hypertext Markup Language*, adalah sebuah bahasa *markup* yang dirancang untuk ditampilkan di sebuah *browser* menjadi sebuah halaman web. HTML lebih menekankan pada penggambaran komponen-komponen, struktur dan format di dalam halaman web daripada menentukan penampilannya. HTML termasuk kedalam *client-side scripting*, artinya kode HTML dijalankan di sisi pengguna (*browser*).

Elemen HTML digambarkan oleh *tag*, ditulis menggunakan tanda kurung sudut (< >). Kebanyakan *tag* HTML memiliki sebuah *tag* penutup yang memiliki nama yang sama hanya diawali oleh karakter slash (/) sebelum nama *tag*. HTML bukanlah bahasa pemrograman, karena di dalam HTML tidak akan ditemukan struktur yang biasa ditemukan dalam bahasa pemrograman seperti *if*, *loop*, *variabel*, dan lain-lain. HTML hanya sebuah bahasa struktur yang fungsinya untuk menandai bagian-bagian dari sebuah halaman.

## 2.3. CSS

*Cascading Style Sheet* (CSS) merupakan aturan untuk mengatur beberapa komponen dalam sebuah web sehingga akan lebih terstruktur dan seragam. CSS bukan merupakan bahasa pemrograman. Secara singkat, CSS dapat diartikan sebagai kumpulan kode program yang digunakan untuk mendefinisikan desain dari bahasa *markup*, salah satunya adalah HTML. Kegunaan utama CSS adalah untuk mendesain atau mempercantik tampilan halaman *website*. Dengan CSS, seseorang dapat mengubah desain sebuah elemen dalam HTML, misalnya

mengubah warna sebuah teks, mengganti ukuran suatu elemen, mengubah warna latar, dan lain lain.

CSS biasanya disimpan dalam dokumen terpisah berekstensi “.css” dan dihubungkan melalui sepenggal kode dalam dokumen bahasa *markup*. Akan tetapi, kita tetap bisa menggunakan CSS tanpa membuat dokumen terpisah. Caranya dengan menyematkan sintaks CSS ke dalam *tag* <style> di dokumen HTML tersebut. Kita juga dapat menyematkan sintaks CSS langsung di dalam atribut style pada suatu tag HTML untuk menerapkan style yang hanya akan digunakan dalam *tag* itu saja.

## **2.4. Bootstrap**

Bootstrap adalah sebuah *framework* CSS yang berfokus untuk membuat halaman web kita menjadi responsif. Dengan menggunakan Bootstrap kita hanya tinggal mengetik nama *class* yang dibutuhkan kedalam elemen *tag* HTML. Bootstrap juga menyediakan beberapa komponen yang siap digunakan seperti *navbar*, *form-control*, *dropdown*, dan lain-lain.

## **2.5. JavaScript**

JavaScript dibuat dan didesain selama sepuluh hari oleh Brandan Eich, seorang karyawan Netscape, pada bulan September 1995. Awalnya bahasa pemrograman ini disebut Mocha, kemudian diganti ke Mona, lalu LiveScript sebelum akhirnya resmi menyandang nama JavaScript. Versi pertama dari bahasa ini hanya terbatas di kalangan Netscape saja. Fungsionalitas yang ditawarkan pun terbatas. Namun, JavaScript terus dikembangkan oleh komunitas developer yang tak henti-hentinya mengerjakan bahasa pemrograman ini.

Biasanya JavaScript di-*embed* secara langsung ke halaman *website* atau diarahkan melalui berkas “.js” yang terpisah. JavaScript merupakan bahasa *client-side*.

### 2.5.1. JQuery

JQuery adalah *library* JavaScript populer. Bahasa pemrograman ini dibuat oleh John Resig, tepatnya pada tahun 2006, untuk memudahkan para pengembang dalam menggunakan dan menerapkan JavaScript di website. JQuery bukanlah bahasa pemrograman yang berdiri sendiri, melainkan bekerja sama dengan JavaScript.

Fungsi utama JQuery adalah untuk menyederhanakan sintaks Javascript yang terkenal rumit dan sulit dipahami. Adanya jQuery juga memudahkan penggunaan Ajax.

### 2.5.2. AJaX

AJaX adalah singkatan dari *Asynchronous JavaScript and XML*. AjaX memungkinkan kita untuk mengambil konten dari *server back-end* secara *asynchronous*, tanpa perlu *me-refresh* halaman. Dengan menggunakan AjaX, maka pengguna dapat memperbarui konten halaman web tanpa memuat ulang atau *reload*.

## 2.6. Python

Python diciptakan oleh Guido van Rossum pada awal tahun 1990-an. Nama “Python” bukanlah terinspirasi dari jenis ular melainkan terinspirasi dari sebuah acara sketsa komedi yang ditayangkan di BBC Channel yakni “Monty Python’s Flying Circus”. Tidak seperti bahasa pemrograman lainnya yang banyak dikerjakan dan dirilis oleh perusahaan besar dengan melibatkan para profesional, Python adalah bahasa pemrograman “*Open Source*” yang artinya, Python dikembangkan secara berkesinambungan oleh ribuan programmer, tester dan user yang kebanyakan bukan ahli IT dari seluruh dunia hingga akhirnya menjadi seperti sekarang.



Versi pertama Python saat pertama kali dirilis adalah versi 0.9.0. Kemudian pada tahun 2000 Python 2.0 dirilis, dan Python terbaru yang dirilis saat laporan ini ditulis adalah versi 3.9 yang dirilis pada Oktober 2020.

### 2.6.1. Penulisan Kode

Kode Python harus ditulis dalam berkas berekstensi “.py”. Tidak seperti kebanyakan bahasa pemrograman lainnya, Python tidak memerlukan karakter *semicolon* (;) untuk mengakhiri sebuah sintaks. Walaupun *semicolon* dapat digunakan untuk menulis beberapa sintaks dalam satu baris, tetapi menurut *style guide* dalam penulisan kode Python, cara penulisan seperti itu tidak dianjurkan karena akan membuat kode yang ditulis sulit untuk dibaca dan dipahami.

Python juga tidak menggunakan *curly bracket* ({} ) untuk membuat blok program. Python menggunakan karakter *colon* (:) dan indentasi (tab atau spasi 2x atau spasi 4x) untuk membuat sebuah blok program. Untuk mencetak suatu output kita hanya tinggal menulis “print”.



```
1 a = 'foo'
2 b = 'bar';c = 'boo'
3
4 if a == 'foo':
5     print(a)
6     print('hello world');print(b)
```

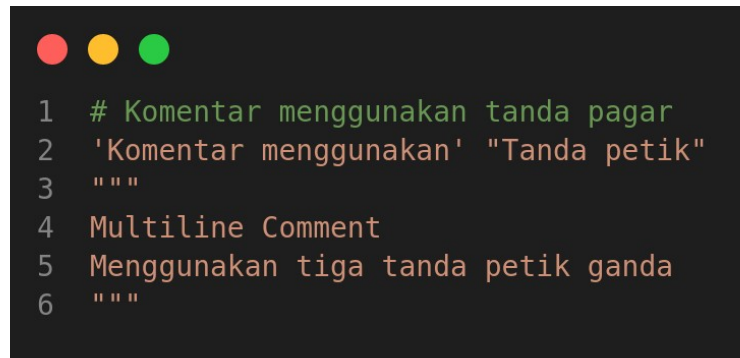
Gambar 2.1 Penulisan Kode Python

### 2.6.2. Komentar

Komentar adalah baris kode yang tidak akan dieksekusi. Ada 3 cara untuk menulis komentar pada bahasa pemrograman Python:

1. Menggunakan tanda pagar (#)

2. Menggunakan tanda petik tunggal (') atau tanda petik ganda (")
3. Multiline Comment, menggunakan tiga tanda petik ganda (""")



```
1  # Komentar menggunakan tanda pagar
2  'Komentar menggunakan' "Tanda petik"
3  """
4  Multiline Comment
5  Menggunakan tiga tanda petik ganda
6  """
```

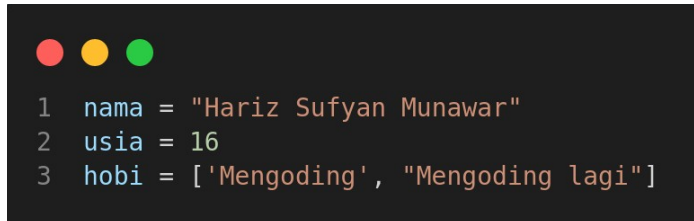
Gambar 2.2 Baris Komentar Python

### 2.6.3. Variabel

Variabel adalah lokasi memori yang dicadangkan untuk menyimpan nilai-nilai. Variabel menyimpan data yang dilakukan selama program dieksekusi, yang nantinya isi dari variabel tersebut dapat diubah oleh operasi-operasi tertentu pada program yang menggunakan variabel. Di dalam bahasa pemrograman Python, variabel mempunyai sifat dinamis, artinya variabel Python tidak perlu dideklarasikan tipe data tertentu dan variabel Python dapat diubah saat program dijalankan.

Penulisan pengidentifikasi variabel Python juga memiliki aturan tertentu, yaitu:

1. Karakter pertama harus berupa huruf atau garis bawah
2. Karakter selanjutnya dapat berupa huruf, garis bawah, atau angka
3. Karakter pada nama variabel bersifat case-sensitive. Artinya huruf kecil dan huruf besar adalah karakter yang berbeda. Sebagai contoh, variabel “fooBar” dan “foobar” adalah dua variabel yang berbeda.



```
1 nama = "Hariz Sufyan Munawar"
2 usia = 16
3 hobi = ['Mengoding', "Mengoding lagi"]
```

Gambar 2.3 Deklarasi Variabel Python

#### 2.6.4. Tipe Data

Dalam bahasa pemrograman, tipe data adalah klasifikasi data yang mengenalkan kompilator atau penerjemah bagaimana kita sebagai programmer bermaksud untuk menggunakan suatu data. Berikut adalah macam-macam tipe data yang ada di dalam bahasa pemrograman Python:

1. Boolean

*Boolean* hanya menyatakan True atau False. Tipe data ini biasanya digunakan untuk struktur kontrol. Cara membuat variabel *boolean* adalah dengan menulis True atau False saat pendeklarasian variabel, perhatikan cara penulisannya karena Python hanya menerima True atau False.

2. String

*String* adalah tipe data yang menyatakan karakter atau kalimat bisa berupa huruf, angka, atau simbol-simbol lain. Cara membuat variabel *string* adalah dengan menulis teks yang diapit oleh tanda petik tunggal (') maupun tanda petik ganda (").

3. Integer

*Integer* menyatakan bilangan bulat. Dibuat dengan menulis angka pada saat pendeklarasian variabel.

4. Float

*Float* adalah tipe bilangan desimal. Dibuat layaknya *integer* tetapi dapat menerima tanda titik dan angka dibelakang tanda titik.

#### 5. List

*List* atau lebih dikenal sebagai *array* di berbagai bahasa pemrograman, adalah sebuah tipe data yang dapat menampung berbagai macam *value* dari tipe data apa saja dan isinya bisa lebih dari satu. Dideklarasikan menggunakan tanda kurung siku “[ ]”.

#### 6. Tuple

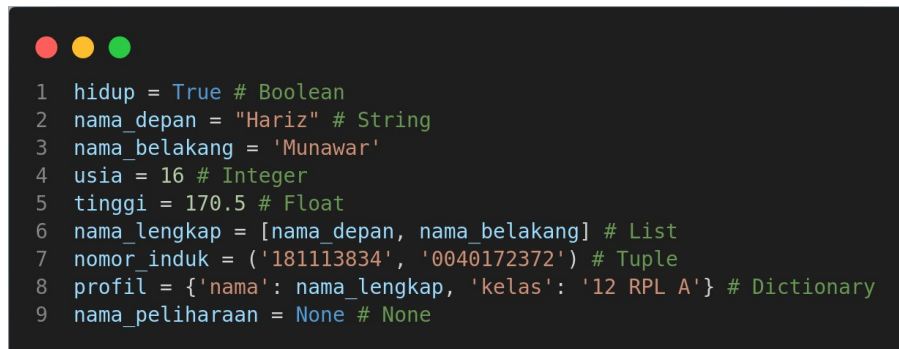
*Tuple* adalah tipe data seperti *list* yang isinya tidak dapat diubah setelah pendeklarasian. Kelebihan dari tipe data ini dibanding *list* adalah penggunaan memorinya yang jauh lebih kecil. Dideklarasikan menggunakan tanda kurung “()”.

#### 7. Dictionary

*Dictionary* adalah sebuah tipe data yang menyimpan data secara berpasangan dalam bentuk berupa *key* dan *value*.

#### 8. None

*None* adalah tipe data yang digunakan untuk menyatakan bahwa sebuah variabel tidak memiliki nilai atau NULL. Tipe data ini akan mengembalikan False jika digunakan untuk sebuah ekspresi.



```

1 hidup = True # Boolean
2 nama_depan = "Hariz" # String
3 nama_belakang = 'Munawar'
4 usia = 16 # Integer
5 tinggi = 170.5 # Float
6 nama_lengkap = [nama_depan, nama_belakang] # List
7 nomor_induk = ('181113834', '0040172372') # Tuple
8 profil = {'nama': nama_lengkap, 'kelas': '12 RPL A'} # Dictionary
9 nama_peliharaan = None # None

```

Gambar 2.4 Tipe Data Dalam Python

### 2.6.5. Operator

*Operator* digunakan untuk melakukan operasi seperti memanipulasi, melakukan perbandingan, memberikan nilai dan lain lain kepada suatu value. Berikut adalah *operator* yang terdapat dalam bahasa pemrograman Python.

Tabel 2.1 Operator Aritmetika

Nama	Contoh	Hasil
<i>Addition</i>	$X + Y$	Jumlah dari X ditambah Y
<i>Subtraction</i>	$X - Y$	Jumlah dari X dikurang Y
<i>Multiplication</i>	$X * Y$	Jumlah dari X dikali Y
<i>Division</i>	$X / Y$	Jumlah dari X dibagi Y
<i>Modulus</i>	$X \% Y$	Sisa pembagian dari X dibagi Y
<i>Exponentiation</i>	$X ** Y$	Jumlah dari X pangkat Y
<i>Floor Division</i>	$X // Y$	Jumlah dari X dibagi Y, dibulatkan kebawah

Tabel 2.2 Operator Perbandingan

Nama	Contoh	Hasil
<i>Equal</i>	$X == Y$	True, jika nilai X sama dengan Y
<i>Not Equal</i>	$X != Y$	True, jika nilai X tidak sama dengan Y
<i>Greater Than</i>	$X > Y$	True, jika nilai X lebih besar dari Y
<i>Less Than</i>	$X < Y$	True, jika nilai X lebih kecil dari Y
<i>Greater Than or Equal To</i>	$X >= Y$	True, jika nilai X lebih besar atau sama dengan Y
<i>Less Than or Equal To</i>	$X <= Y$	True, jika nilai X lebih kecil atau sama dengan Y

Tabel 2.3 Operator Assignment

Operator	Contoh	Sama Dengan
$=$	$X = 5$	$X = 5$
$+=$	$X += 5$	$X = X + 5$
$-=$	$X -= 5$	$X = X - 5$
$*=$	$X *= 5$	$X = X * 5$
$/=$	$X /= 5$	$X = X / 5$
$\% =$	$X \% = 5$	$X = X \% 5$
$// =$	$X // = 5$	$X = X // 5$
$** =$	$X ** = 5$	$X = X ** 5$

Tabel 2.4 Operator Logika

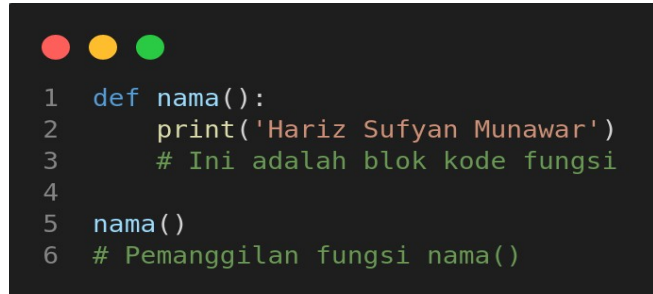
Operator	Contoh	Hasil
and	<code>X == 1 and Y == 2</code>	True, jika nilai X adalah 1 dan nilai Y adalah 2
or	<code>X == 1 or Y == 2</code>	True, jika nilai X adalah 1 atau nilai Y adalah 2
not	<code>not X == 1</code>	True, jika nilai X bukanlah 1

Tabel 2.5 Operator Keanggotaan

Operator	Contoh	Hasil
in	<code>X in [1, 'ok', True]</code>	True, jika nilai X adalah Integer 1 atau String 'ok' atau Boolean True
not in	<code>X not in [2, 'rei']</code>	True, jika nilai X bukanlah Integer 2 atau String 'rei'

#### 2.6.6. Fungsi

Fungsi adalah serangkaian kode yang setelah ditulis sekali dapat dijalankan berkali-kali dengan memanggil nama fungsinya. Aturan penulisan nama fungsi sama dengan penulisan nama variabel, namun diawali dengan keyword “def” lalu diikuti kurung buka dan kurung tutup. Kode yang akan menjadi bagian fungsi haruslah sebuah blok kode.



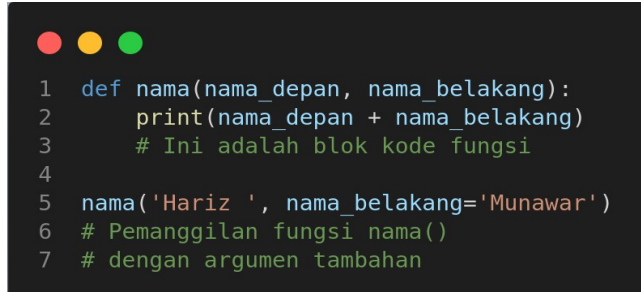
```

1  def nama():
2      print('Hariz Sufyan Munawar')
3      # Ini adalah blok kode fungsi
4
5  nama()
6  # Pemanggilan fungsi nama()

```

Gambar 2.5 Penulisan Fungsi Dalam Python

Fungsi dapat memiliki argumen, yaitu nilai yang diberikan ketika kita memanggil fungsi tersebut. Argumen ditulis di dalam tanda kurung setelah nama fungsi, dan dipisah dengan koma jika jumlah argumen lebih dari satu. Pada saat kita memanggil fungsi dan memberikan argumen, kita harus memberikan nilai argumen sesuai urutan kecuali kita menulis secara langsung argumen apa yang akan diberi nilai.



```

1  def nama(nama_depan, nama_belakang):
2      print(nama_depan + nama_belakang)
3      # Ini adalah blok kode fungsi
4
5  nama('Hariz ', nama_belakang='Munawar')
6  # Pemanggilan fungsi nama()
7  # dengan argumen tambahan

```

Gambar 2.6 Fungsi Menggunakan Argumen

### 2.6.7. Struktur Kontrol

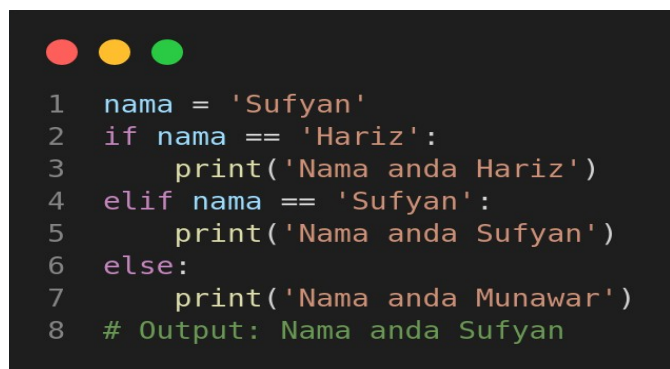
Perintah kode Python dijalankan secara berurutan, dimulai pada baris pertama kemudian ke baris berikutnya sampai dengan baris akhir. Namun, bagian-bagian kode dapat dijalankan hanya pada kondisi tertentu atau diulang hingga kondisi tertentu dengan menggunakan *control statement*. Seperti fungsi, bagian dari *control statement* juga wajib merupakan blok kode.

1. If



Struktur kontrol *if* dapat mengatur kode apa yang dijalankan berdasarkan nilai dari variabel yang diberikan. Kode dalam blok *if* akan dijalankan jika value dari *expression*-nya bernilai True.

Struktur kontrol *if* dapat diikuti dengan *elif* dan atau *else*. Jika blok *if* tidak dijalankan karena *expression*-nya tidak bernilai True, *expression* pada *elif* akan dicek dan kode pada bloknya akan dijalankan jika *expression*-nya bernilai True. Jika semua *expression* tidak ada yang bernilai True, maka kode pada blok *else* yang akan dijalankan. Dalam satu struktur “if”, hanya diperbolehkan ada satu statement *if* dan *else* sedangkan untuk statement *elif* tidak terbatas.

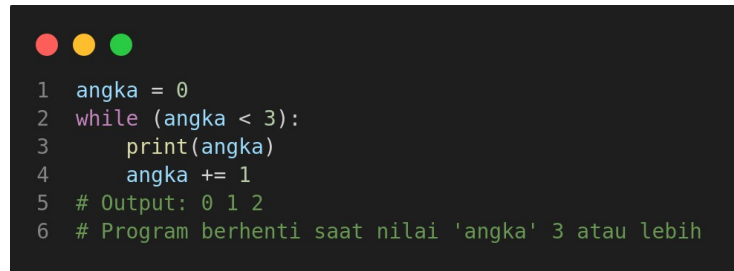


```
1  nama = 'Sufyan'
2  if nama == 'Hariz':
3      print('Nama anda Hariz')
4  elif nama == 'Sufyan':
5      print('Nama anda Sufyan')
6  else:
7      print('Nama anda Munawar')
8  # Output: Nama anda Sufyan
```

Gambar 2.7 Struktur Kontrol If

## 2. While

Struktur kontrol *while* adalah bentuk perulangan. Kode dalam bloknya akan terus diulang sampai *expression*-nya bernilai False atau diinstruksikan untuk berhenti menggunakan *keyword break*.



```

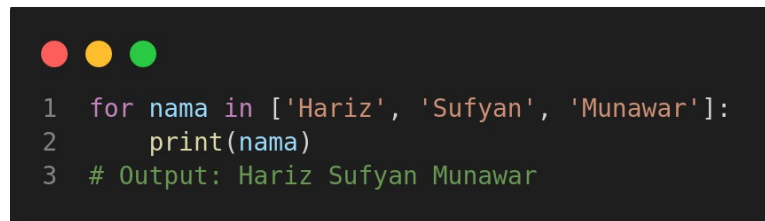
1  angka = 0
2  while (angka < 3):
3      print(angka)
4      angka += 1
5  # Output: 0 1 2
6  # Program berhenti saat nilai 'angka' 3 atau lebih

```

Gambar 2.8 Struktur Kontrol While

### 3. For

Struktur kontrol *for* juga merupakan struktur perulangan. *For* dalam Python lebih terfokus untuk melakukan iterasi terhadap rentetan data (list, tuple, dictionary, dll).



```

1  for nama in ['Hariz', 'Sufyan', 'Munawar']:
2      print(nama)
3  # Output: Hariz Sufyan Munawar

```

Gambar 2. 9 Struktur Kontrol For

#### 2.6.8. Pip

Pip adalah sebuah *package manager* untuk bahasa pemrograman Python. Fungsi dari *package manager* sendiri adalah untuk memasang, memperbarui dan menghapus *package*. Beberapa contoh dari *package* Python yang penulis gunakan dalam aplikasi ini adalah django, weasyprint, whitenoise, dan lain-lain.

### 2.7. Django

Django adalah sebuah *framework full-stack* untuk membuat aplikasi web dengan bahasa pemrograman Python. *Full-stack* artinya, django meliputi sisi *front-end* dan juga *back-end*. Django dibuat pada tahun 2003 oleh Simon Wilison

dan Adrian Holovaty. Nama django sendiri diambil dari nama seorang gitaris kebangsaan Belgia dan Perancis yaitu, Django Reinhardt.

Django merupakan *framework* yang populer karena memiliki banyak keunggulan dibanding *framework-framework* lain, diantaranya:

1. Django ditulis menggunakan bahasa Python

Django ditulis menggunakan bahasa Python yang notabene merupakan bahasa pemrograman yang mudah dipahami. Python juga memiliki kegunaan yang sangat luas dalam dunia pemrograman seperti *data science* atau *machine learning*, dengan menggunakan django kita bisa dengan mudah mengintegrasikan semua proyek berbasis Python kedalam aplikasi web kita.

2. Prinsip KISS dan DRY sangat dijunjung tinggi

Django sangat menekankan prinsip KISS (Keep It Short and Simple) dan DRY (Dont Repeat Yourself). KISS berarti kode django yang ditulis harus singkat dan mudah dimengerti. Sedangkan DRY berarti kita tidak boleh melakukan hal yang sama berulang kali, gunakan fungsi agar kita tidak mengulang kode dan agar kode lebih singkat.

3. Mempunyai *built-in* ORM yang *powerful*

ORM (Object Relational Mapper) adalah sebuah fungsi yang dapat kita gunakan sebagai pengganti SQL pada saat kita mengakses *database*. Kelebihan ORM dibandingkan dengan kita menggunakan SQL secara langsung adalah, ORM lebih sederhana dan karena ORM merupakan sebuah fungsi Python, kita bisa dengan mudah memanipulasi data baik yang didapat atau akan dikirimkan menuju *database*.

4. Django Admin

Django memiliki sebuah halaman admin yang otomatis dan mudah untuk dikostumisasi.

### 2.7.1. Instalasi

Sama seperti *package* Python lain, django di-*install* menggunakan bantuan pip. Caranya adalah dengan mengetik “pip install django” atau “pip3 install django” pada command line atau terminal.

```
(test-env) munawar@Munawar-134:~/Desktop$ pip3 install django
Collecting django
  Using cached Django-3.1.5-py3-none-any.whl (7.8 MB)
Collecting asgiref<4,>=3.2.10
  Using cached asgiref-3.3.1-py3-none-any.whl (19 kB)
Collecting pytz
  Using cached pytz-2020.5-py2.py3-none-any.whl (510 kB)
Collecting sqlparse>=0.2.2
  Using cached sqlparse-0.4.1-py3-none-any.whl (42 kB)
Installing collected packages: asgiref, pytz, sqlparse, django
Successfully installed asgiref-3.3.1 django-3.1.5 pytz-2020.5 sqlparse-0.4.1
```

Gambar 2.10 Instalasi Django

Jika django sudah terpasang, versi django bisa dicek dengan mengetik “django-admin --version”. Akan muncul nomor versi jika django memang sudah terpasang dengan benar.

```
(test-env) munawar@Munawar-134:~/Desktop$ django-admin --version
3.1.5
```

Gambar 2.11 Cek Versi Django

### 2.7.2. Server Lokal

Sebelum menjalankan *server* lokal kita perlu membuat projek terlebih dahulu. Cara membuat projek django adalah dengan mengetik “django-admin startproject <namaprojek>” pada command line atau terminal.

```
(test-env) munawar@Munawar-134:~/Desktop$ django-admin startproject REI
```

Gambar 2.12 Memulai Projek Django

Jika proyek berhasil dibuat, maka akan muncul folder dengan nama proyek. Lalu untuk menjalankan *server* lokal kita harus masuk kedalam folder tadi dan mengetik “python manage.py runserver” atau “python3 manage.py runserver” pada command line atau terminal.

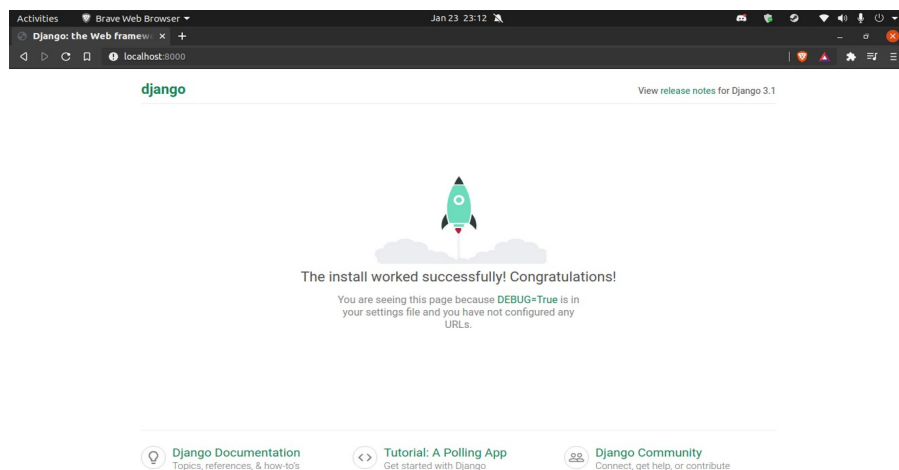
```
(test-env) munawar@Munawar-134:~/Desktop$ cd REI/
(test-env) munawar@Munawar-134:~/Desktop/REI$ python3 manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).

You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth, contenttypes, session
5.
Run 'python manage.py migrate' to apply them.
January 23, 2021 - 16:10:12
Django version 3.1.5, using settings 'REI.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

Gambar 2.13 Memulai Server Lokal

Secara *default server* django akan berada pada port 8000, jadi untuk mengakses proyek tadi kunjungi “localhost:8000” atau “127.0.0.1:8000”. Jika semua berjalan sesuai dengan yang diharapkan maka halaman ini akan terlihat.



Gambar 2.14 Halaman Pertama Django

### 2.7.3. Konsep MTV

Setiap *framework* pasti memiliki sebuah arsitektur agar alur kerja dari *framework* dapat berjalan dengan maksimal. Kebanyakan web *framework* lain mengadopsi arsitektur *Model View Controller* (MVC), sedangkan django mengadopsi arsitektur yang diberi nama *Model Template View* (MTV). Konsep MTV adalah konsep yang membagi komponen-komponen utama dalam membangun sebuah website menjadi 3 bagian.

#### 2.7.3.1. Model

Model adalah bagian yang berhubungan erat dengan basis data. Bagian ini berisi tentang bagaimana skema basis data yang nantinya akan dibuat, apa saja yang harus dilakukan ketika sebuah data akan dimasukkan kedalam atau dihapus dari *record* basis data, bagaimana cara memvalidasi data yang akan dimasukkan kedalam *record* basis data, dan lain-lain.

#### 2.7.3.2. Template

Template adalah bagian yang menentukan bagaimana hasil atau tampilan yang akan diterima oleh *user*. Bagian ini biasanya berisi *script client-side* seperti HTML, CSS dan JavaScript.

#### 2.7.3.3. View

View adalah bagian yang dapat mengakses Model dan Template. Bisa dibilang View adalah bagian yang menghubungkan kedua komponen tadi. View biasanya berisi dengan logika yang bermaksud untuk mengolah data yang diterima dari Model agar bisa dikirimkan menuju Template.

## 2.8. Database

*Database* (basis data) didefinisikan sebagai sekumpulan data yang saling berhubungan, disimpan dengan minimum redundansi untuk melayani banyak aplikasi secara optimal. Basis data merupakan sistem yang terdiri atas kumpulan

file atau tabel yang saling berhubungan dan *Database Management System* (DBMS) yang memungkinkan beberapa pemakai untuk mengakses dan manipulasi file-file tersebut.

### 2.8.1. SQL

*Structured Query Language* (SQL) adalah bahasa *query* yang dirancang untuk pengambilan informasi tertentu dari database. Bahasa *query* merupakan bahasa yang digunakan untuk memanipulasi data di DBMS. Hal ini berlaku baik untuk memasukkan, merubah, atau menghapusnya. Perintah dasar SQL dibagi kedalam tiga jenis.

#### 1. Data Definition Language (DDL)

Jenis perintah dasar yang pertama adalah *Data Definition Language* atau biasa disingkat dengan DDL. Perintah dasar ini sebenarnya merupakan perintah paling mendasar dari bahasa SQL. Tujuannya untuk membuat struktur sebuah database. Perintah-perintah yang termasuk kedalam DDL adalah:

1. Create, sebuah perintah yang digunakan ketika membuat sebuah basis data baru, baik itu berupa tabel baru atau sebuah kolom baru.
2. Alter, digunakan untuk mengubah struktur tabel yang sebelumnya sudah ada, seperti nama tabel, penambahan kolom, mengubah, maupun menghapus kolom serta menambahkan atribut lainnya.
3. Rename, digunakan untuk mengubah sebuah nama di sebuah tabel ataupun kolom yang ada.
4. Drop, digunakan untuk menghapus baik itu berupa *database*, tabel maupun kolom hingga index.
5. Show, digunakan untuk menampilkan tabel atau *databases* yang ada.

#### 2. Data Manipulation Language (DML)

Yang kedua adalah *Data Manipulation Language* (DML). Perintah SQL ini bertujuan untuk memanipulasi data yang ada dalam sebuah database. Ada 4 perintah yang termasuk kedalam DML.

1. Insert, digunakan untuk memasukkan sebuah *record* baru kedalam sebuah tabel.
2. Select, digunakan untuk menampilkan atau mengambil data dari satu atau lebih tabel dalam sebuah *database*.
3. Update, digunakan untuk memperbarui data. Contohnya saat ada kesalahan pada proses Insert data, daripada melakukan proses Delete lalu melakukan insert ulang, menggunakan Update jauh lebih sederhana.
4. Delete, digunakan untuk menghapus *record* yang ada di dalam sebuah tabel.

### 3. Data Control Language (DCL)

Perintah dasar berikutnya adalah *Data Control Language* atau DCL. Perintah SQL ini digunakan untuk mengatur hak apa saja yang dimiliki oleh pengguna. Baik itu hak terhadap sebuah *database* ataupun pada tabel maupun *field* yang ada. Ada 2 perintah dasar yang termasuk kedalam DCL.

1. Grant, digunakan ketika admin database ingin memberikan hak akses ke *user* lainnya. Tentu pemberian hak akses ini dapat dibatasi atau diatur. Dalam hal ini admin pun dapat memberikan akses mengenai perintah dalam DML di atas.
2. Revoke, digunakan untuk mencabut maupun menghapus hak akses seorang pengguna yang awalnya diberikan akses oleh admin database melalui perintah Grant sebelumnya.



### 2.8.2. DBMS

*Database Management System* (DBMS) adalah suatu sistem atau *software* yang dirancang khusus untuk mengelola suatu basis data dan menjalankan operasi terhadap data yang diminta oleh banyak pengguna. DBMS merupakan perantara untuk *user* dengan basis data.

#### 2.8.2.1. SQLite

SQLite adalah *software library* yang menerapkan *engine database* SQL secara mandiri, tanpa memerlukan *server*, tanpa perlu melakukan konfigurasi, dan bersifat transaksional. SQLite adalah *engine database* SQL yang paling banyak digunakan di dunia. SQLite adalah sebuah *engine database* SQL yang langsung tertanam atau pada aplikasi. Tidak seperti kebanyakan *database* SQL lainnya, SQLite tidak memiliki *server* yang terpisah dari aplikasi. SQLite membaca dan menulis langsung ke *file disk* biasa.

#### 2.8.2.2. MongoDB

MongoDB merupakan salah satu jenis *database* NoSQL yang berbasis dokumen dengan format JSON. MongoDB didirikan oleh Kevin Ryan, Dwight Meriman, Eliot Horowitz yang kini tergabung dalam MongoDB Inc. Dan masing-masing memiliki peran dan jabatan yang sangat penting yakni Kevin Ryan yang tengah menjabat sebagai board member. Kemudian Dwight berkedudukan sebagai chairman serta Eliot Horowitz dengan jabatan CTO.

Keunggulan dari MongoDB ini adalah dalam sistem penyimpanan data tidak lagi menggunakan tabel. Akan tetapi, menggunakan dokumen terstruktur layaknya JSON karena telah menggunakan JavaScript. Sehingga performa yang dihasilkan oleh MongoDB akan lebih cepat. MongoDB juga menggunakan *cloud based server*, sehingga kita tidak memerlukan *server* lokal untuk mengakses *database*.

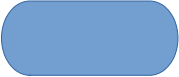




## 2.9. Web Server








*Server* atau *web server* adalah sebuah *software* yang memberikan layanan berbasis data dan berfungsi menerima permintaan dari HTTP atau HTTPS pada klien yang dikenal dan biasanya kita kenal dengan nama *web browser* (Mozilla Firefox, Google Chrome) dan untuk mengirimkan kembali hasil dalam bentuk beberapa halaman web dan pada umumnya akan berbentuk dokumen HTML.

## 2.10. Flowmap

*Flowmap* (peta alir) adalah bagan yang menggambarkan aliran kerja dari sebuah prosedur. *Flowmap* merupakan pengembangan dari *flowchart*, namun pada *flowmap* lebih dijelaskan tokoh-tokoh yang menjalankan suatu pekerjaan. Ada beberapa simbol yang digunakan dalam *flowmap*, berikut adalah sedikit penjelasan dari beberapa simbol pada *flowmap*.

Tabel 2.6 Simbol-Simbol Flowchart

Simbol	Nama	Arti
	Terminator	Simbol awal atau akhir dari program.
	Process	Simbol proses yang dijalankan oleh komputer atau sistem.
	On Page Connector	Simbol penyambungan proses dalam halaman yang sama.
	Off Page Connector	Simbol penyambungan proses yang berbeda halaman.
	Manual Operation	Simbol proses yang bukan dijalankan oleh komputer atau sistem.









Simbol	Nama	Arti
	Decision	Simbol yang berfungsi untuk memilih proses berdasarkan kondisi yang ada.
	Input-Output Data	Simbol yang berfungsi untuk meminta input dan menampilkan output.
	Manual Input	Simbol yang berfungsi untuk pemasukan data secara manual on-line keyboard
	Predefined Process	Simbol yang berfungsi untuk pelaksanaan suatu bagian (sub-program) / prosedur.
	Display	Simbol yang berfungsi untuk menyatakan peralatan output yang digunakan yaitu layar, plotter, printer dan sebagainya.
	Document	Simbol yang berfungsi untuk menyatakan input berasal dari dokumen dalam bentuk kertas atau output dicetak ke kertas.
	Multi Document	Simbol yang berfungsi untuk menyatakan input berasal dari banyak dokumen dalam bentuk kertas atau output dicetak ke kertas.

## 2.11. Data Flow Diagram

*Data Flow Diagram* (Diagram Alur Data) atau disingkat DFD adalah diagram yang menggambarkan alur data dalam sebuah sistem informasi. Sebuah DFD menggambarkan data macam apa yang menjadi *input* dan *output* dari sebuah sistem, dari mana data tersebut datang dan kemana data tersebut akan disalurkan, serta dimana data tersebut akan disimpan.

DFD dipecah menjadi beberapa level, dimulai dari diagram konteks yang bentuknya sederhana, hanya berisi satu notasi proses yang mencakup seluruh sistem dan hubungannya dengan entitas luar. Kemudian DFD dipecah menjadi level 0, dimana notasi proses sistem diuraikan menjadi proses-proses utama. Diagram masih bisa terus dipecah, menjelaskan lagi proses-proses yang ada. Umumnya penggambaran DFD tidak lebih dari level 2.

Dalam menggambar DFD ada beberapa gaya notasi. Salah satunya, yang disebut notasi Yourdon/DeMarco, menggambarkan simbol-simbol DFD dengan sebagai berikut: lingkaran untuk proses, garis atas dan garis bawah untuk tempat penyimpanan data, persegi panjang untuk entitas luar, dan anak panah untuk alur data.

Notation	Yourdon and Coad	Gane and Sarson
External Entity		
Process		
Data Store		
Data Flow		



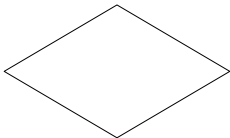
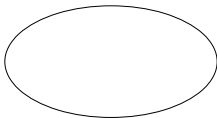
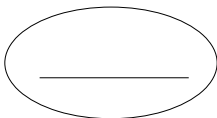

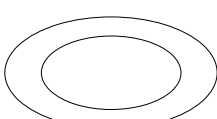
Gambar 2.15 Simbol-Simbol DFD

## 2.12. Entity Relation Diagram

*Entity Relation Diagram* (ERD) adalah suatu model yang digambar untuk menjelaskan struktur dan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi. Pada dasarnya ada 3 komponen yang digunakan, yaitu:

1. Entitas, merupakan objek yang mewakili sesuatu yang nyata dan dapat dibedakan dari sesuatu yang lain. Simbol dari entitas ini biasanya digambarkan dengan persegi panjang.
2. Atribut, setiap entitas pasti mempunyai elemen yang disebut atribut yang berfungsi untuk mendeskripsikan karakteristik dari entitas tersebut. Isi dari atribut mempunyai sesuatu yang dapat mengidentifikasi isi elemen satu dengan yang lain. Gambar atribut diwakili oleh simbol elips.
3. Relasi, adalah hubungan antara sejumlah entitas yang berasal dari himpunan entitas yang berbeda.

Tabel 2.7 Simbol-Simbol ERD

Notasi	Arti
	Entity
	Weak Entity
	Relationship
	Atribut
	Atribut Primary Key
	Atribut Derivatif
	Atribut Multi Value

## BAB III

### ANALISA DAN PERANCANGAN

#### 3.1. Analisa Aplikasi

Aplikasi ini merupakan web aplikasi yang memiliki 2 fungsi utama, yaitu sebagai media pencatatan nilai serta alat untuk mencetak rapor digital. Web aplikasi ini dibuat dengan menggunakan *framework* Python bernama Django agar memudahkan *developer* untuk mengembangkan aplikasi serta *framework* CSS Bootstrap untuk sisi *frontend*.

##### 3.1.1. Analisa Sistem Pengguna

Berikut adalah tabel analisa kebutuhan pengguna, meliputi daftar pihak berupa *level* pengguna yang terlibat dalam aplikasi. Adapun pihak dan hak aksesnya adalah sebagai berikut.

Tabel 3.1 Analisa Sistem Pengguna

Level	Hak Akses
Staf TU	<ol style="list-style-type: none"><li>1. Melakukan <i>Create, Read, Update, Delete</i> (CRUD) akun guru (Walikelas atau Staf TU).</li><li>2. Melakukan CRUD pada data siswa.</li><li>3. Melakukan CRUD pada informasi sekolah.</li><li>4. Melakukan CRUD pada data jurusan.</li><li>5. Melakukan CRUD pada data mata pelajaran serta nilai Kriteria Ketuntasan Minimal (KKM)-nya.</li><li>6. Melakukan CRUD pada data ekstrakurikuler.</li><li>7. Melakukan impor atau ekspor data siswa dari dan</li></ol>

Level	Hak Akses
	<p>ke berkas spreadsheet.</p> <p>8. Mengatur relasi kelas (Siapa saja anggota kelas, apa saja mata pelajaran yang akan dipelajari pada kelas itu, siapa walikelas yang akan mewalikan kelas itu)</p>
Walikelas	<p>1. Mengisi nilai siswa yang diwalikan.</p> <p>2. Mengisi nilai ekstrakurikuler siswa yang diwalikan.</p> <p>3. Mengisi absensi siswa yang diwalikan.</p> <p>4. Mencetak rapor siswa yang diwalikan secara individual.</p> <p>5. Mencetak seluruh rapor siswa yang diwalikan.</p>
Admin	<p>1. Melakukan <i>Create, Read, Update, Delete</i> (CRUD) akun guru (Walikelas, Staf TU dan Admin).</p> <p>2. Melakukan CRUD pada data siswa.</p> <p>3. Melakukan CRUD pada informasi sekolah.</p> <p>4. Melakukan CRUD pada data jurusan.</p> <p>5. Melakukan CRUD pada data mata pelajaran serta nilai Kriteria Ketuntasan Minimal (KKM)-nya.</p> <p>6. Melakukan CRUD pada data ekstrakurikuler.</p> <p>7. Melakukan impor atau ekspor data siswa dari dan ke berkas spreadsheet.</p> <p>8. Mengatur relasi kelas (Siapa saja anggota kelas, apa saja mata pelajaran yang akan dipelajari pada kelas itu, siapa walikelas yang akan mewalikan</p>



Level	Hak Akses
	<p>kelas itu)</p> <p>9. Mengisi nilai siswa.</p> <p>10. Mengisi nilai ekstrakurikuler siswa.</p> <p>11. Mengisi absensi siswa.</p> <p>12. Mencetak rapor siswa secara individual.</p> <p>13. Mencetak rapor siswa per-kelas.</p>

### 3.1.2. Analisa Kebutuhan Perangkat Sistem

Dalam pembuatan aplikasi, dibutuhkan sistem yang digunakan *developer* untuk memudahkan dalam pengembangan. Berikut spesifikasi perangkat keras dan perangkat lunak yang direkomendasikan untuk perangkat *server* maupun *client*.

#### 3.1.2.1. Kebutuhan Komputer Server

##### 1. Perangkat Keras

- a) *Processor* : Intel® Core™ i3-2330M CPU @ 2.20GHz
- b) *Memori* : 4,00 *GigaByte*(GB)
- c) *Harddisk* : 500 GB

Spesifikasi di atas bukanlah sebuah keharusan, namun sangat direkomendasikan.

##### 2. Perangkat Lunak

- a) *Sistem Operasi* : Windows 7, 8, 10 atau sistem operasi berbasis GNU/Linux
- b) *Python* versi 3 atau lebih.
- c) *PIP* atau *PIP3*.
- d) *Peramban Web*, seperti Google Chrome, Microsoft Edge, dll.
- e) *Text Editor*, seperti Visual Studio Code, Atom, dll.

f) *Software* pendukung: *GTK3 runtime*.

#### **3.1.2.2. Kebutuhan Komputer Client**

Agar pengguna dapat mengoperasikan aplikasi dengan baik, disarankan komputer *client* memiliki *software* dan spesifikasi *hardware* sebagai berikut:

1. Peramban web versi terbaru agar bisa menjalankan fungsi Ajax dengan baik.
2. Monitor dengan lebar setidaknya 800 *pixel*.
3. Memori lebih dari 2,00 GB.
4. Koneksi internet yang setidaknya stabil.

Spesifikasi di atas selain poin nomor 4 bukanlah sebuah keharusan, namun sangat direkomendasikan.

### **3.2. Rancangan Sistem**

Aplikasi ini direncanakan untuk bisa menangani pemasukkan nilai serta pembuatan rapor digital. Aplikasi ini dapat digunakan oleh 3 jenis pengguna yaitu walikelas, staf tata usaha, dan admin. Walikelas mempunyai hak untuk melakukan *Create, Read, Update, Delete* (CRUD) pada nilai dan kehadiran siswa yang diwalikan dan dapat mencetak rapor siswa yang diwalikan. Staf tata usaha mempunyai hak untuk melakukan CRUD pada data siswa, data kelas, data mata pelajaran, data jurusan, data ekskul, dan data sekolah. Staf tata usaha juga dapat membuat akun pengguna (walikelas dan staf tata usaha). Sedangkan admin mempunyai hak untuk melakukan CRUD pada semua data dan dapat mencetak rapor setiap siswa.

Fitur yang terdapat pada aplikasi ini adalah pembuatan dokumen rapor siswa, pengaturan akun walikelas dan staf tata usaha, *input* data siswa, *input* nilai dan absensi siswa, pengaturan kelas dan mata pelajaran, pergantian semester, dan pengubahan data sekolah.

### **3.2.1. Pembuatan Dokumen Rapor Siswa**

Aplikasi ini dapat membuat sebuah dokumen dalam bentuk *Portable Document Format* (PDF) yang berisi halaman sampul, informasi sekolah, halaman biodata siswa, dan nilai (mata pelajaran, ekstrakurikuler) serta kehadiran siswa. Rapor yang dibuat adalah rapor per-semester. Pembuatan rapor juga dapat dilakukan per-kelas, hasil yang akan didapat berupa sebuah berkas *Zone Improvement Plan* (ZIP) yang didalamnya berisi berkas PDF setiap siswa dari kelas tersebut.

### **3.2.2. Pengaturan Akun Walikelas dan Staf Tata Usaha**

Aplikasi ini dapat menambah, mengurangi, dan mengubah akun guru dan staf yang digunakan untuk mengakses kedua aplikasi.

### **3.2.3. Input Data Siswa**

Aplikasi ini menyediakan antarmuka untuk memasukkan data siswa dan kelas ke dalam *database*. Terdapat dua antarmuka yang disediakan untuk memasukkan data siswa, pertama adalah *form* HTML yang hanya bisa melakukan *input* satu per satu dan yang kedua adalah melalui sebuah berkas *spreadsheet* yang dapat melakukan *input* siswa baik satu siswa maupun lebih. Aplikasi ini juga menyediakan validasi data siswa sebelum dimasukkan ke dalam *database*.

### **3.2.4. Input Data Nilai dan Absensi**

Aplikasi ini menyediakan antarmuka untuk memasukkan data nilai (mata pelajaran dan ekstrakurikuler) dan absensi siswa ke dalam *database*. Nilai dan absensi siswa hanya bisa di-*input* oleh walikelas masing-masing.

### **3.2.5. Pengaturan Kelas**

Aplikasi ini menyediakan antarmuka untuk melakukan manajemen kelas. Seperti mengatur walikelas untuk setiap kelas, mengatur jurusan, dan mengatur siswa yang menjadi anggota suatu kelas.

### 3.2.6. Pengaturan Mata Pelajaran

Aplikasi ini dapat mengatur apa saja mata pelajaran yang ada di dalam suatu kelas dan berapa nilai ketuntasan minimalnya. Hanya akun Staf TU yang dapat melakukan aksi ini.

### 3.2.7. Pergantian Semester

Ketika semester berubah, aplikasi harus bisa menandai data-data yang berlaku pada semester yang lalu (seperti data nilai) sebagai data lama, kemudian aplikasi dapat digunakan kembali seperti biasa, siap untuk diisi dengan data-data baru. Hanya akun Staf TU yang dapat melakukan CRUD data semester dan mengganti semester yang aktif.

### 3.2.8. Perubahan Data Sekolah

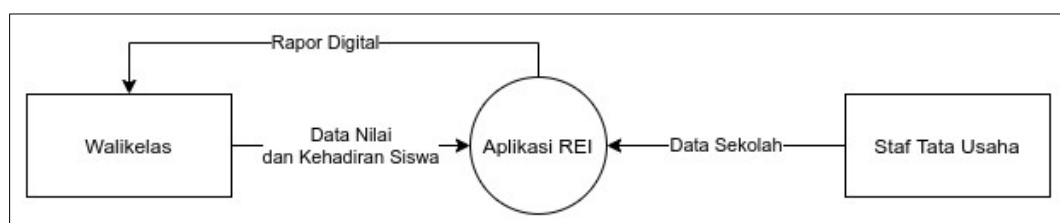
Data sekolah juga dapat diubah agar aplikasi lebih dinamis dan bisa digunakan oleh banyak instansi sekolah. Beberapa fitur juga akan menyesuaikan berdasarkan tingkat sekolah. Misalnya jika tingkat sekolah adalah SMK maka fitur CRUD “Jurusan” akan tersedia.

## 3.3. Data Flow Diagram Aplikasi

Berikut adalah DFD dari aplikasi, mulai dari level 0 sampai dengan level 2.

### 3.3.1. DFD Level 0

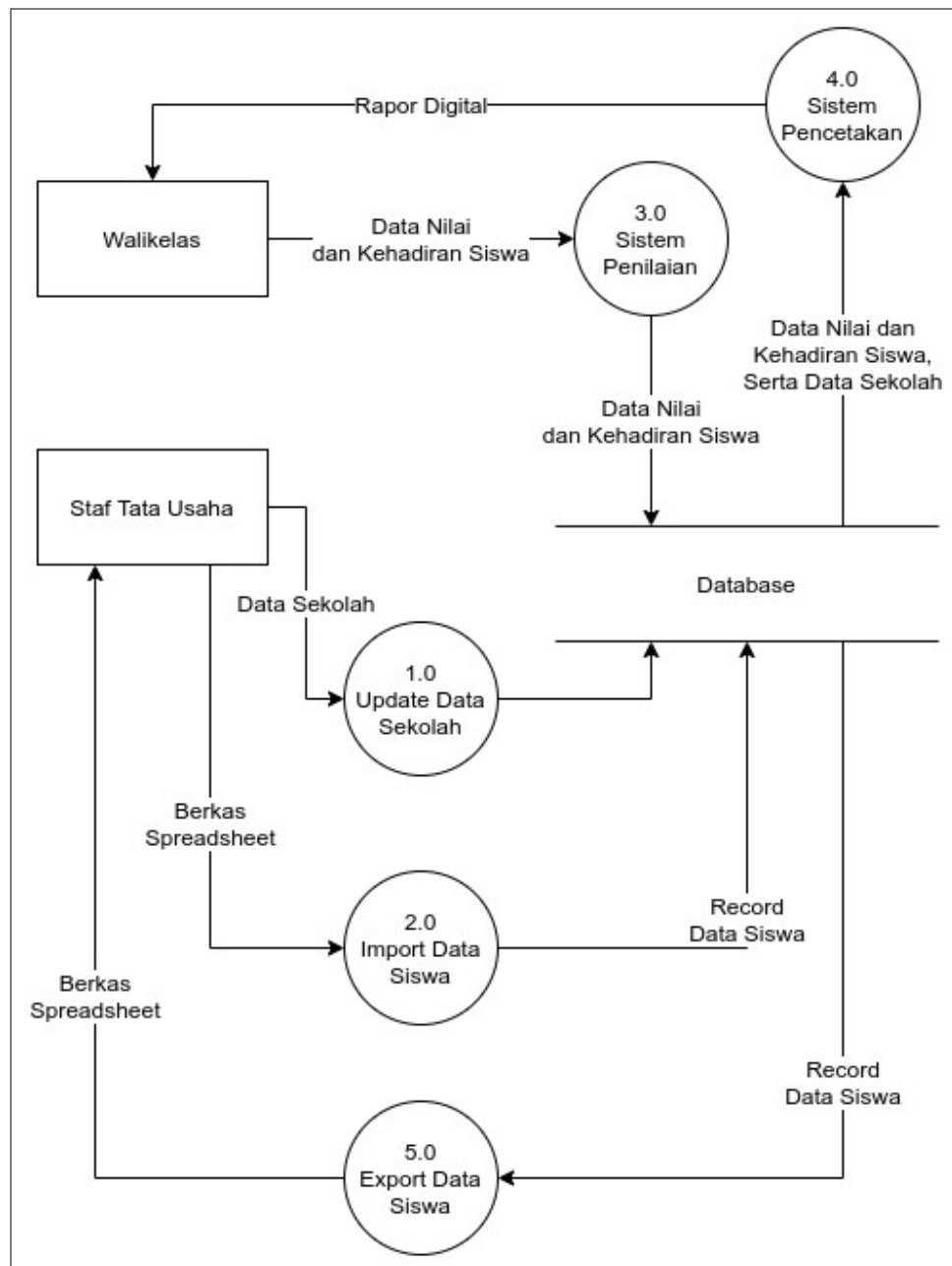
Berikut adalah DFD level 0 dari aplikasi ini.



Gambar 3.1 DFD Level 0

### 3.3.2.DFD Level 1

Pada Diagram Level 1 di bawah terdapat lima buah notasi proses, yaitu *update* data sekolah, sistem penilaian, sistem pencetakan rapor, dan impor ekspor data siswa yang akan diperjelas pada DFD level 2.

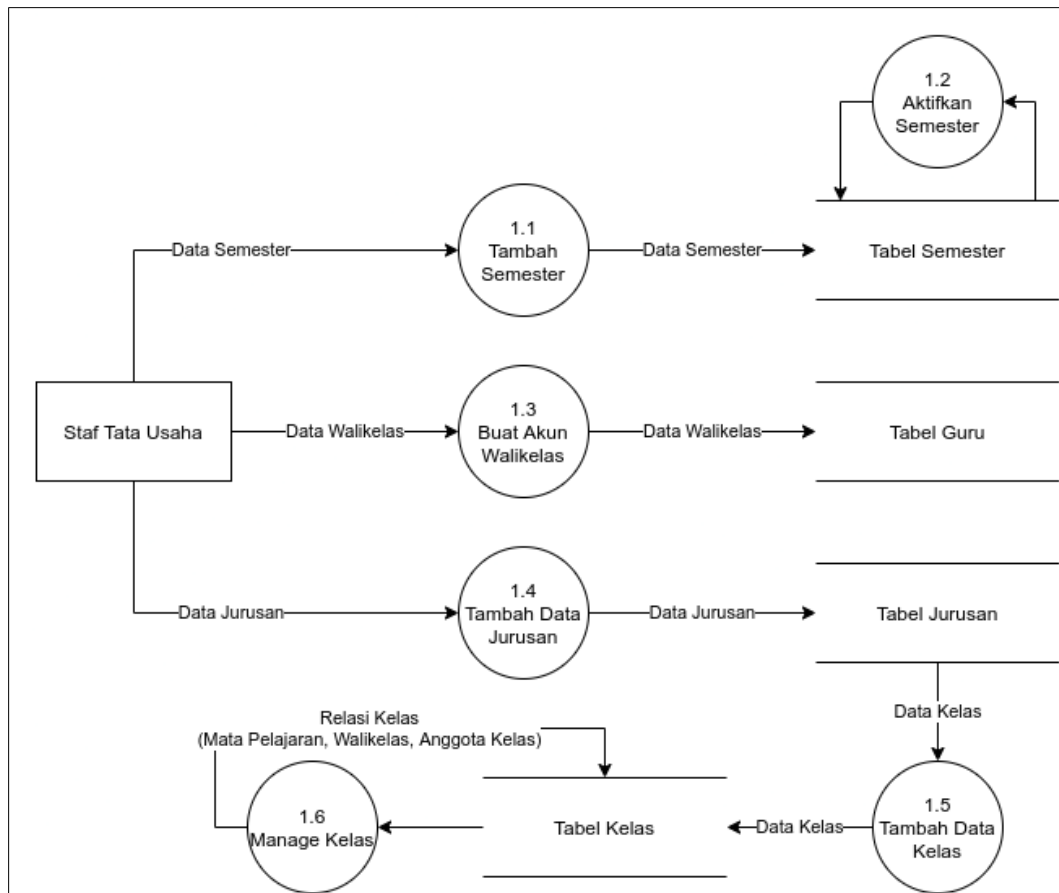


Gambar 3.2 DFD Level 1

### 3.3.3.DFD Level 2

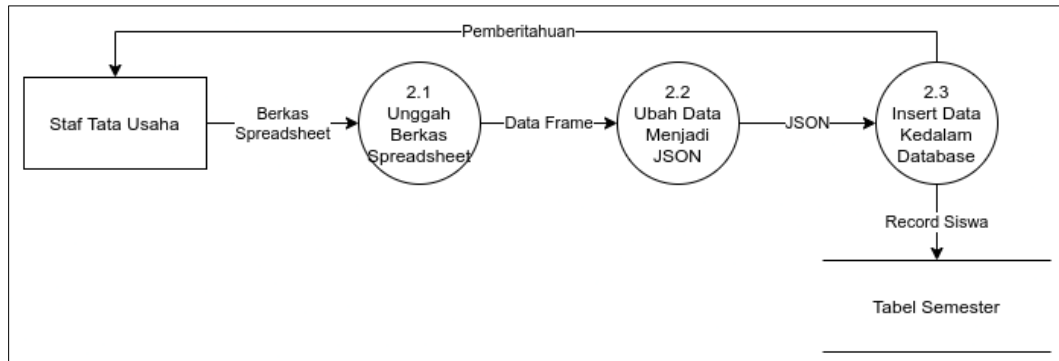
Berikut adalah DFD level 2 dari aplikasi ini.

#### 3.3.3.1. DFD Level 2 Update Data Sekolah



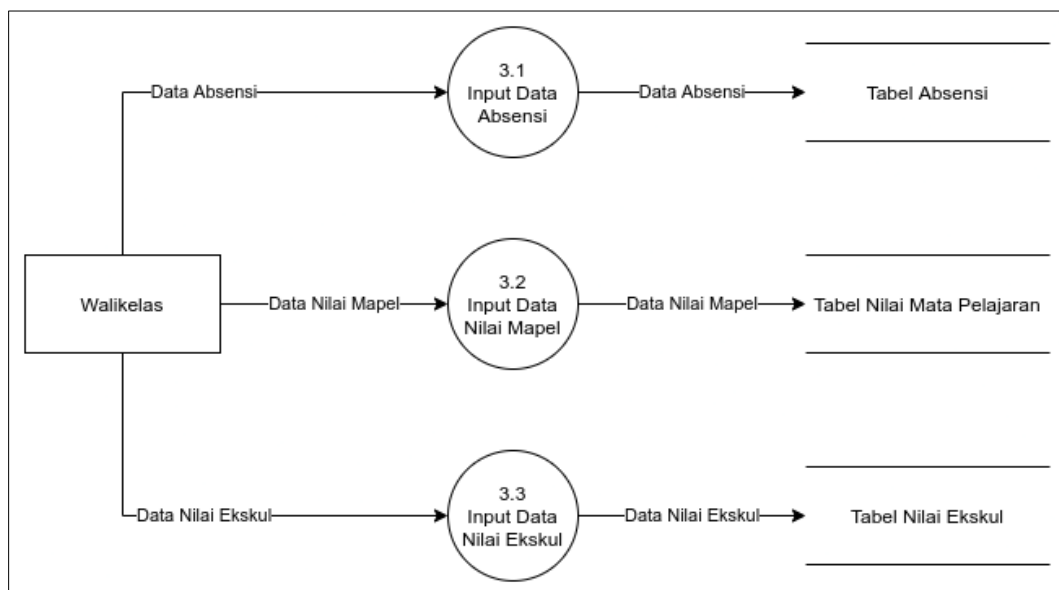
Gambar 3.3 DFD Level 2 Update Data Sekolah

### 3.3.3.2. DFD Level 2 Impor Data Siswa



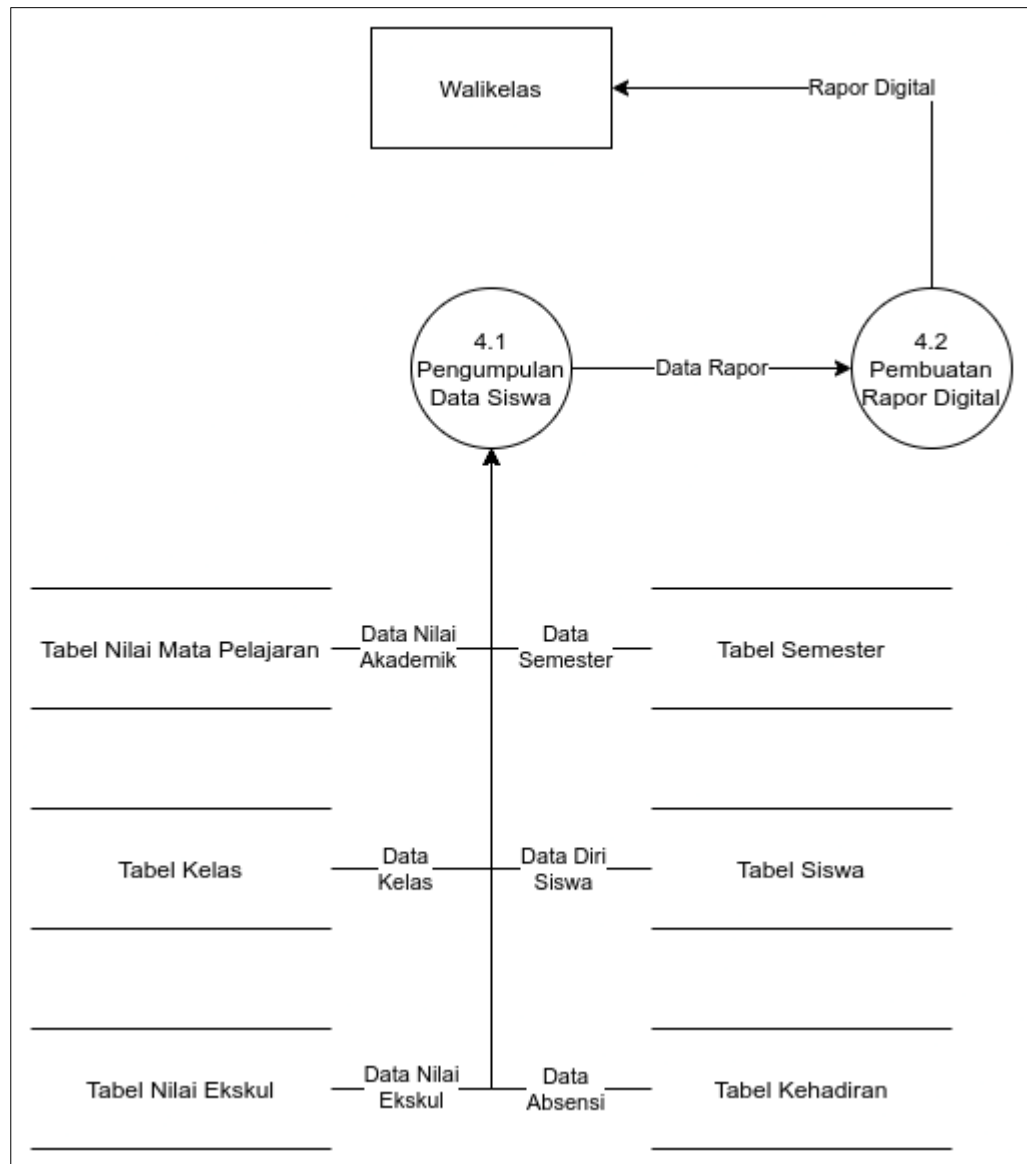
Gambar 3.4 DFD Level 2 Impor Data Siswa

### 3.3.3.3. DFD Level 2 Input Nilai Siswa



Gambar 3.5 DFD Level 2 Input Nilai Siswa

### 3.3.3.4. DFD Level 2 Pencetakan Rapor Digital



Gambar 3.6 DFD Level 2 Pencetakan Rapor Digital

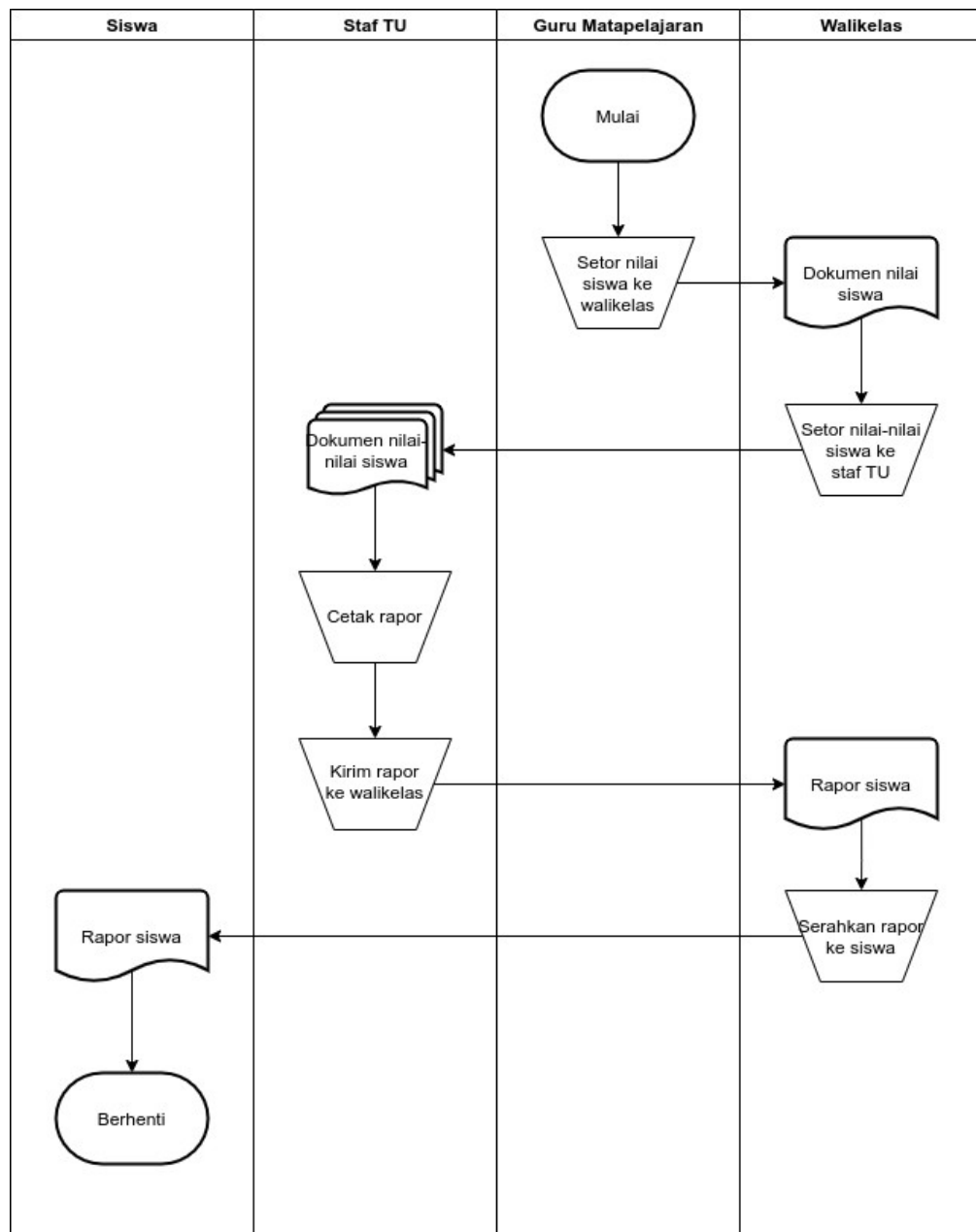
### 3.4. Flowmap

Berikut adalah *flowmap* dari sistem berjalan dan sistem aplikasi ini.



### 3.4.1. Flowmap Sistem Berjalan

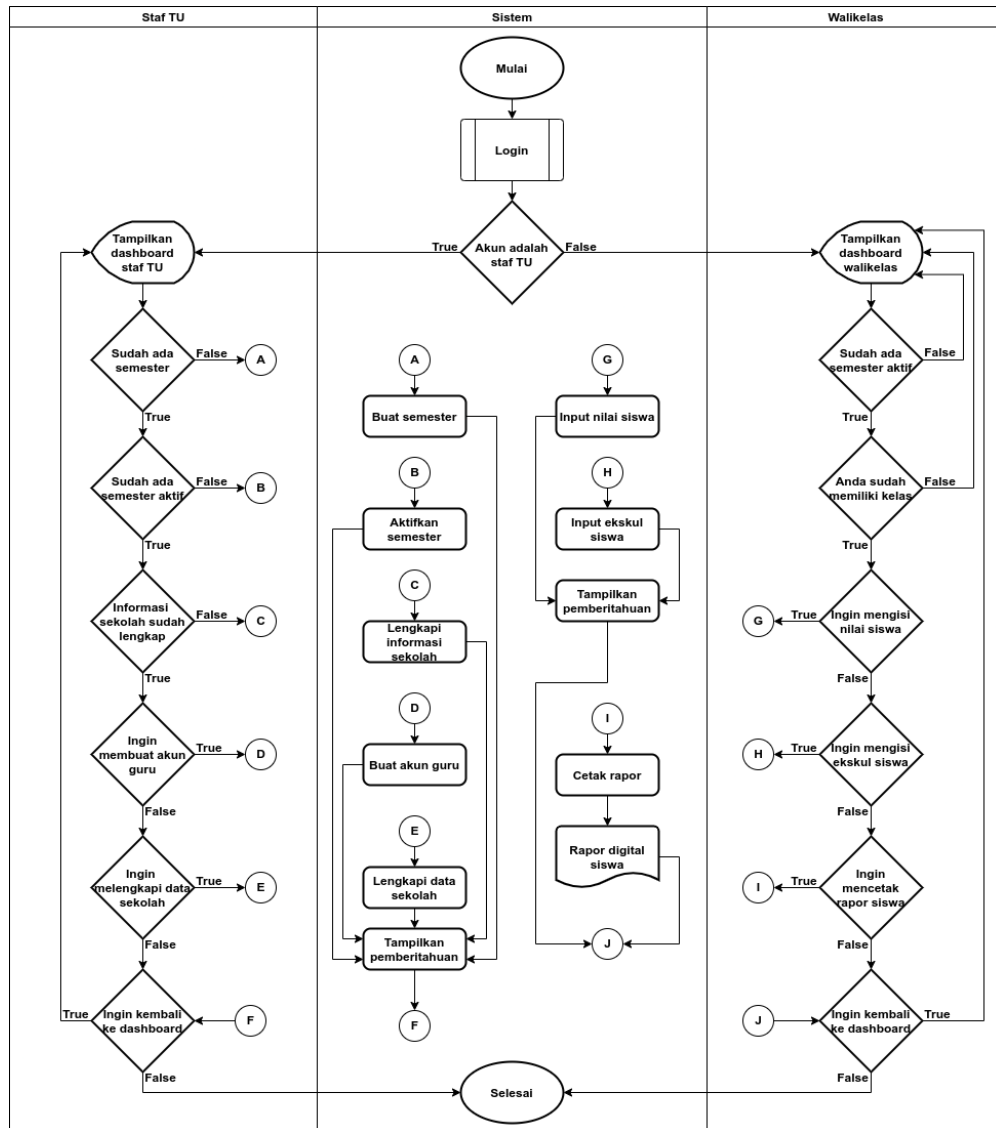
Sistem berjalan adalah sistem yang sudah digunakan dalam mencetak sebuah rapor sebelum aplikasi ini dibuat.



Gambar 3.7 Flowmap Sistem Berjalan

### 3.4.2. Flowmap Aplikasi

Berikut adalah bagaimana aplikasi ini akan berjalan secara garis besar.

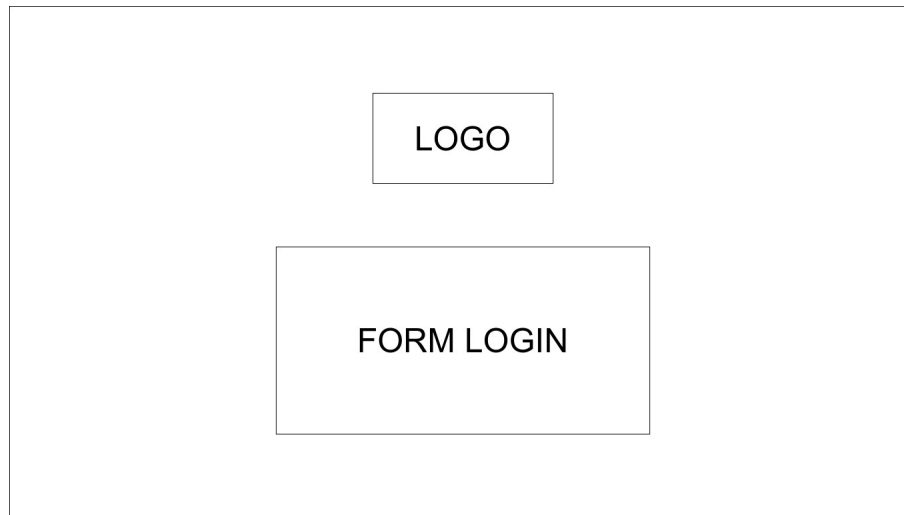


Gambar 3.8 Flowmap Aplikasi



### 3.6.1. Rancangan Tata Letak Halaman Login

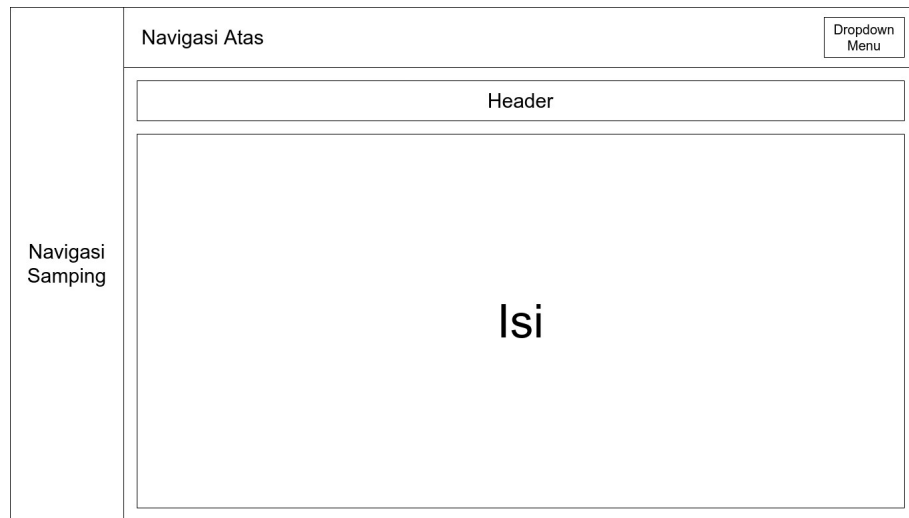
Pada halaman *login* ditampilkan logo dan form yang berisi isian untuk melakukan proses *login* seperti *username* dan *password*. Logo diletakkan di bagian atas halaman dan *form login* diletakkan dibawahnya.



Gambar 3.10 Rancangan Tata Letak Halaman Login

### 3.6.2. Rancangan Tata Letak Halaman Panel

Semua halaman selain halaman *login* akan mengikuti tata letak ini. Bagian isi akan berubah-ubah sesuai dengan halaman yang sedang dikunjungi. Navigasi samping akan berisi tulisan informasi atau *link* (tautan) yang akan membawa pengguna ke halaman lain. Navigasi atas akan menunjukkan informasi tentang halaman apa yang sedang dikunjungi dan akun yang digunakan untuk melakukan *login*. Bagian kepala (*header*) diisi dengan tulisan informasi seperti *error warning* atau *success info*. Lalu bagian *dropdown* akan diisi tautan menuju halaman informasi sekolah, *logout*, *about dev*, dan profil pribadi.



Gambar 3.11 Rancangan Tata Letak Halaman Panel

## BAB IV

### IMPLEMENTASI DAN PENGUJIAN

#### 4.1. Implementasi

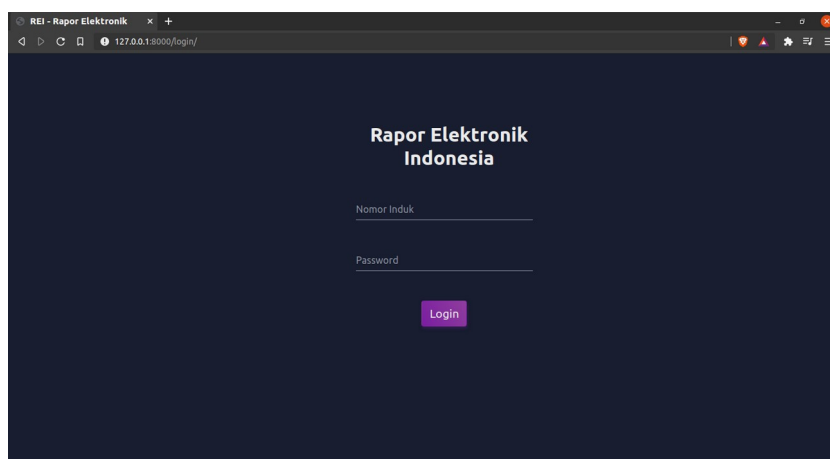
Setelah proses pembuatan aplikasi selesai, aplikasi ini dengan sistem *database*-nya dapat diimplementasikan, dan diharapkan akan memenuhi tujuan pembuatan aplikasi.

##### 4.1.1. Desain Front-End / Tampilan

Tampilan dari *user interface* aplikasi disusun menurut rancangan tata letak pada bab sebelumnya, dengan beberapa perubahan yang menyesuaikan dengan kebutuhan.

##### 4.1.1.1. Halaman Login

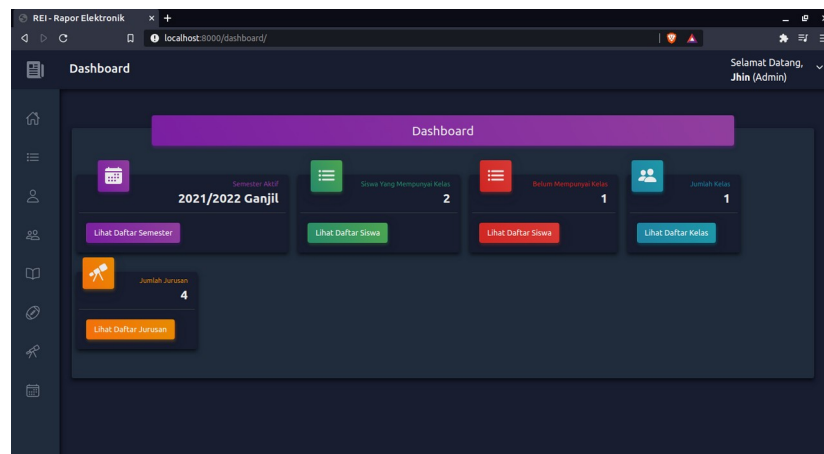
Halaman *login* terdiri atas judul aplikasi dan *form* untuk melakukan proses *login*. *Form* ini memiliki tiga komponen, yaitu isian *username* yang harus diisi dengan NIP seorang guru atau staf TU, isian *password* dan tombol *submit*.



Gambar 4.1 Tampilan Halaman Login

#### 4.1.1.2. Halaman Dashboard

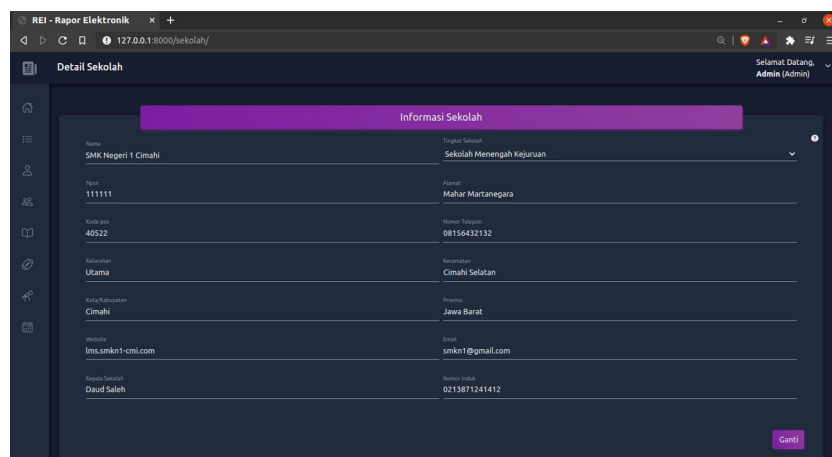
Halaman *dashboard* adalah halaman utama yang akan memberikan pintasan-pintasan singkat ke semua halaman lain.



Gambar 4.2 Halaman Dashboard

#### 4.1.1.3. Halaman Informasi Sekolah

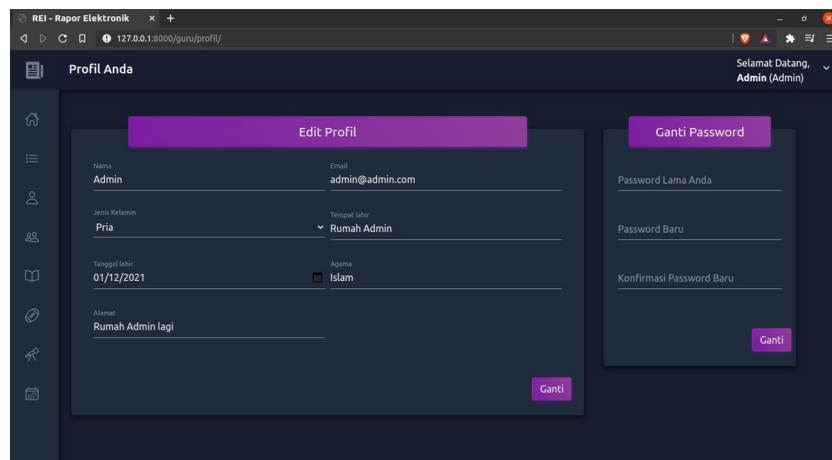
Halaman informasi sekolah ini berisikan sebuah *form*, dimana *form* tadi digunakan untuk menampilkan dan atau mengubah informasi sekolah. *Form* ini akan bersifat “*read-only*” bila diakses oleh akun guru yang tidak memiliki akses staf atau admin.



Gambar 4.3 Halaman Informasi Sekolah

#### 4.1.1.4. Halaman Profil Pribadi

Halaman ini berisi dua buah *form*. *Form* pertama adalah *form* yang digunakan untuk menampilkan dan atau mengubah data seorang guru (kecuali password). *Form* ini akan bersifat “*read-only*” bila diakses oleh akun guru yang tidak memiliki akses staf atau admin. Dan yang kedua adalah *form* yang digunakan untuk mengganti *password* akun yang sedang aktif.

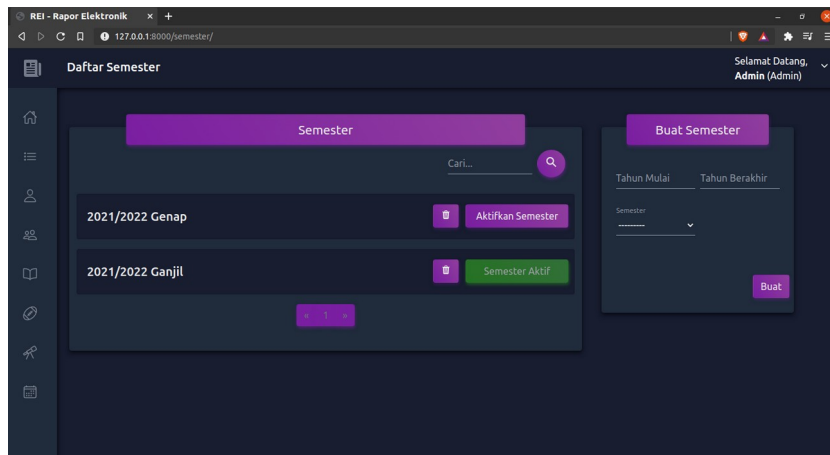


Gambar 4.4 Halaman Profil Pribadi

#### 4.1.1.5. Halaman Pengaturan Semester

Halaman ini hanya bisa diakses oleh akun yang memiliki akses staf atau admin. Berisi daftar semester yang sudah dibuat dan sebuah *form* yang digunakan untuk membuat sebuah semester. Ada juga tombol yang digunakan untuk mengaktifkan sebuah semester.

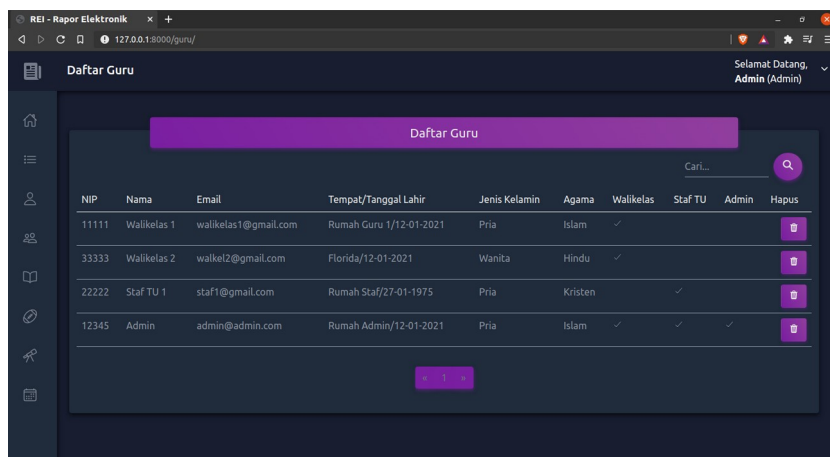




Gambar 4.5 Halaman Pengaturan Semester

#### 4.1.1.6. Halaman Daftar Guru

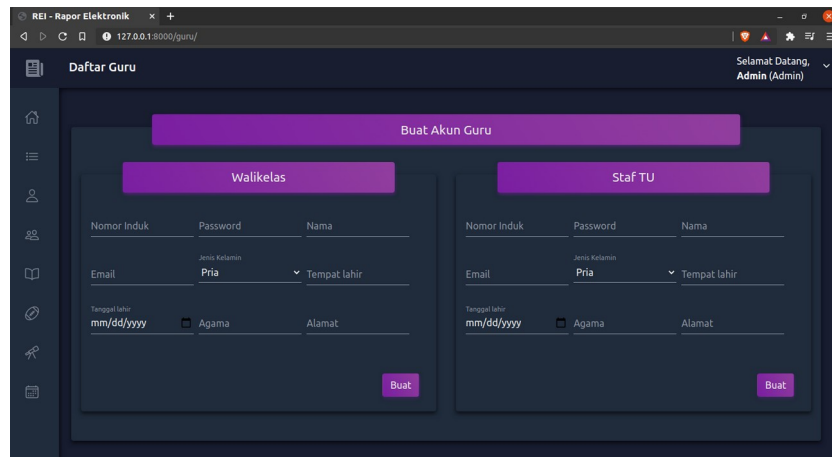
Halaman ini berisikan sebuah tabel yang menampilkan data dari daftar akun guru yang sudah dibuat. Terdapat juga sebuah *search bar* dan *pagination*. Ada juga sebuah tombol untuk menghapus sebuah akun yang hanya bisa dijalankan oleh guru yang mempunyai akses staf atau admin.



Gambar 4.6 Halaman Daftar Guru

#### 4.1.1.7. Halaman Buat Akun Guru

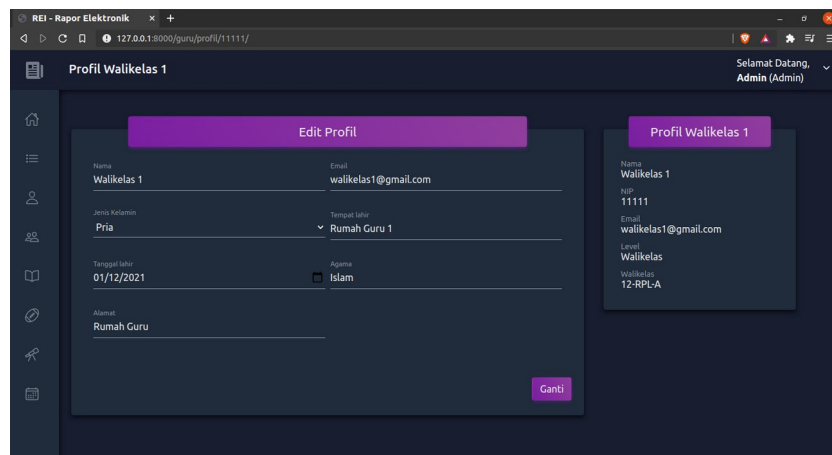
Hanya bisa diakses oleh guru yang mempunyai akses staf atau admin. Berisi 2 *form* yang berfungsi untuk membuat akun.



Gambar 4.7 Halaman Buat Akun Guru

#### 4.1.1.8. Halaman Detail Guru

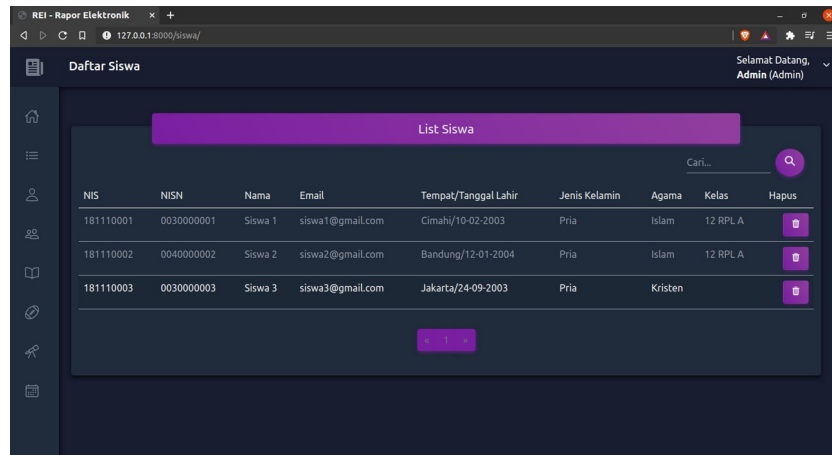
Halaman ini memiliki isi yang hampir mirip dengan halaman profil pribadi. Bedanya disini *form* akan menampilkan detail guru yang dipilih. Dan tidak ada *form* penggantian *password*.



Gambar 4.8 Halaman Detail Guru

#### 4.1.1.9. Halaman Daftar Siswa

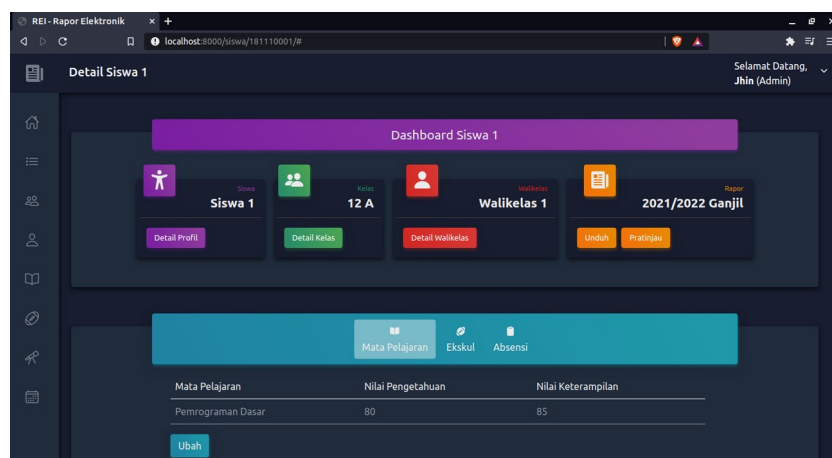
Menampilkan tabel yang berisi daftar siswa. Terdapat juga *search bar* dan *pagination*. Kolom kelas akan ditampilkan berdasarkan semester yang sedang aktif.



Gambar 4.9 Halaman Daftar Siswa

#### 4.1.1.10. Halaman Detail Siswa

Tujuan utama dibuatnya halaman ini adalah untuk memudahkan guru dalam mengatur data seorang siswa. Halaman ini memberikan pintasan menuju data-data siswa yang bisa diubah baik oleh staf atau walikelas.



Gambar 4.10 Halaman Detail Siswa

#### 4.1.1.11. Halaman Nilai Akademik Siswa

Hanya bisa diakses oleh guru yang menjadi walikelas siswa yang bersangkutan. Halaman ini berfungsi untuk mengubah data nilai akademik siswa pada semester yang sedang aktif.

Nilai Siswa	
Bahasa Indonesia	
Pengetahuan	Keterampilan
80	70
Bahasa Inggris	
Pengetahuan	Keterampilan
0	0
Pemrograman Dasar	
Pengetahuan	Keterampilan
0	0
<button>Update</button>	

**Profil**  
Nama Siswa 1  
NIS/NSN 181110001/0030000001  
Usia 17 Tahun  
Kelas 12 RPL A  
Walikelas Walikelas 1  
Kembali ke Detail Siswa

Gambar 4.11 Halaman Nilai Akademik Siswa

#### 4.1.1.12. Halaman Nilai Ekskul Siswa

Hanya bisa diakses oleh guru yang menjadi walikelas siswa yang bersangkutan. Halaman ini berfungsi untuk mengubah data nilai ekskul siswa pada semester yang sedang aktif.

**Ekskul Siswa**

Formasi Diambil Isian: A Update

**Tambah Ekskul**

Ekskul	Nilai
--------	-------

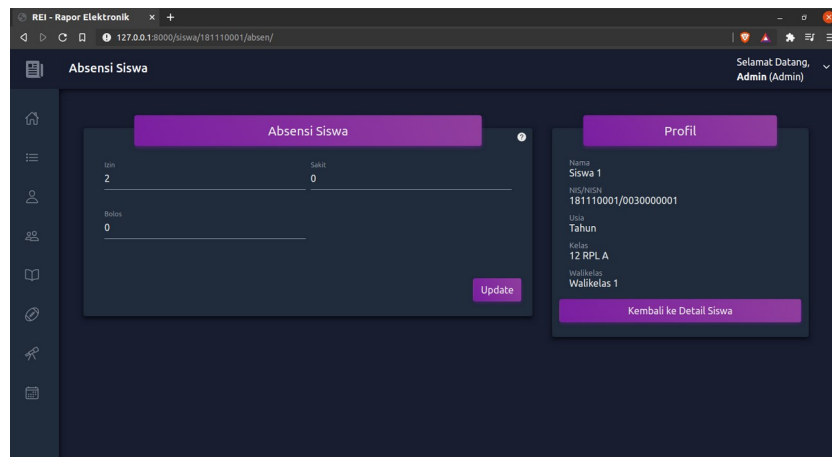
Tambah

**Profil**  
Nama Siswa 1  
NIS/NSN 181110001/0030000001  
Usia Tahun  
Kelas 12 RPL A  
Walikelas Walikelas 1  
Kembali ke Detail Siswa

Gambar 4.12 Halaman Nilai Ekskul Siswa

#### 4.1.1.13. Halaman Absensi Siswa

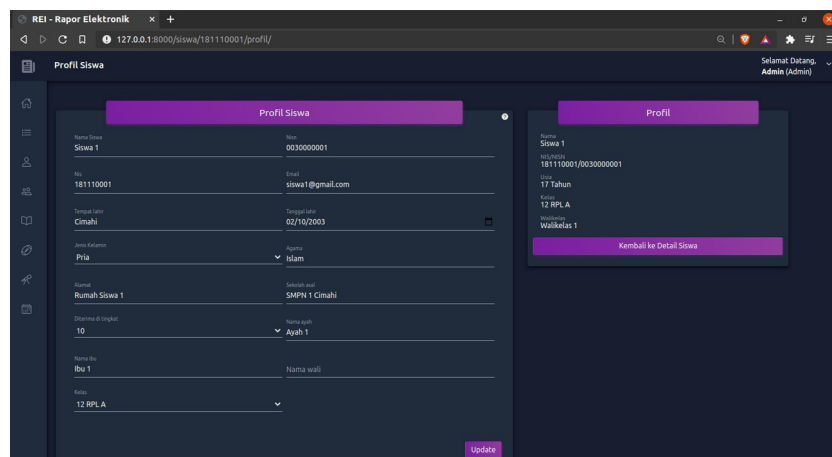
Hanya bisa diakses oleh guru yang menjadi walikelas siswa yang bersangkutan. Halaman ini berfungsi untuk mengubah data absensi siswa pada semester yang sedang aktif.



Gambar 4.13 Halaman Absensi Siswa

#### 4.1.1.14. Halaman Edit Profil Siswa

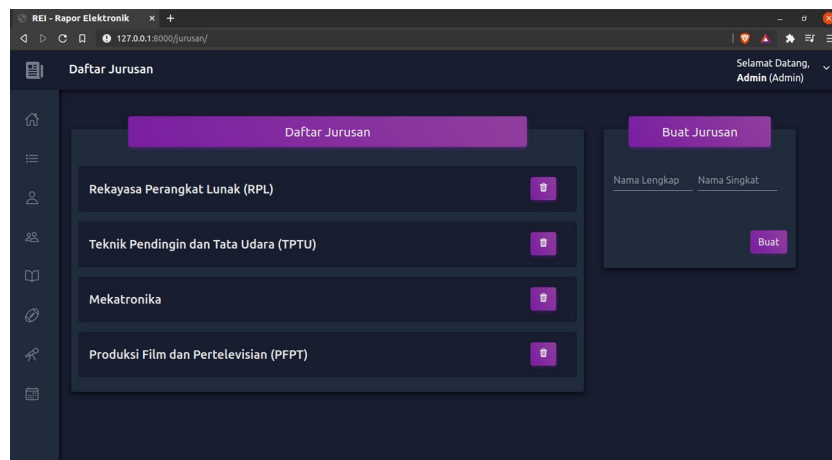
Hanya bisa diakses oleh guru yang mempunyai akses staf atau admin.



Gambar 4.14 Halaman Edit Profil Siswa

#### 4.1.1.15. Halaman Daftar Jurusan

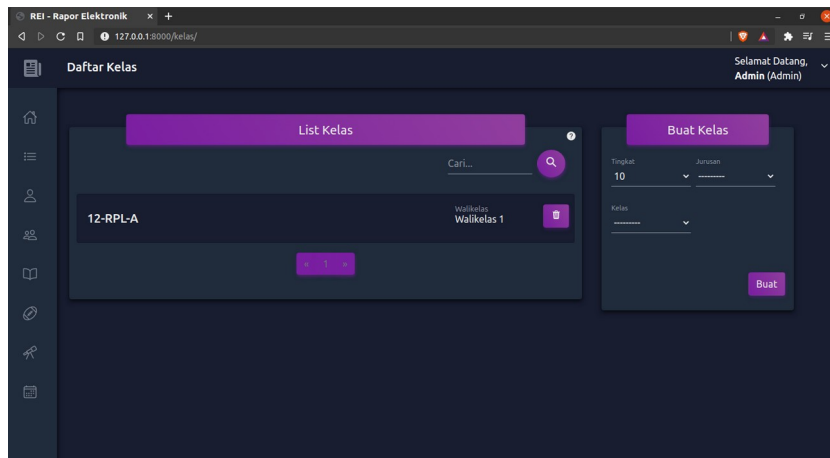
Halaman ini terdiri atas dua bagian. Pertama adalah daftar jurusan beserta tombol untuk menghapus jurusan. Jurusan yang memiliki satu atau lebih kelas tidak akan bisa dihapus. Yang kedua adalah *form* pembuatan jurusan. Aksi membuat dan menghapus jurusan hanya bisa dilakukan oleh guru yang mempunyai akses staf dan admin.



Gambar 4.15 Halaman Daftar Jurusan

#### 4.1.1.16. Halaman Daftar Kelas

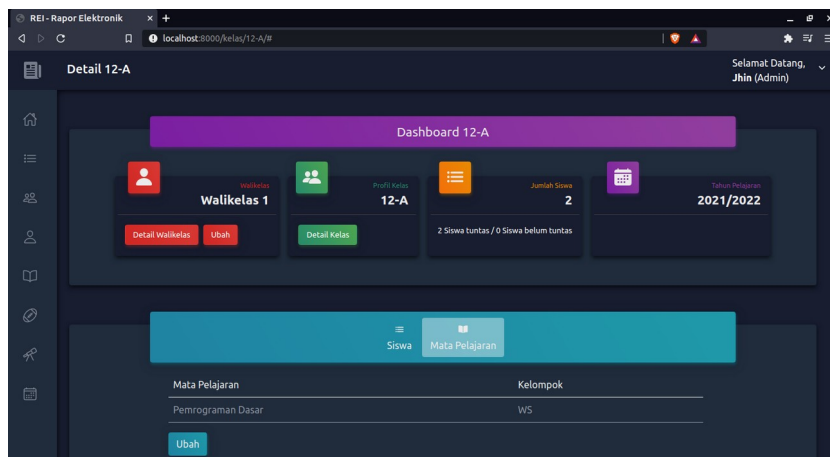
Halaman ini hampir mirip dengan halaman daftar jurusan. Bagian pertama menampilkan daftar kelas beserta walikelasnya jika tersedia, berdasarkan semester yang aktif. Bagian kedua adalah *form* pembuatan kelas. Hanya saja halaman ini memiliki *search bar* dan *pagination*.



Gambar 4.16 Halaman Daftar Kelas

#### 4.1.1.17. Halaman Detail Kelas

Halaman ini berisi pintasan yang memudahkan pengguna untuk mengubah data yang berhubungan dengan sebuah kelas.

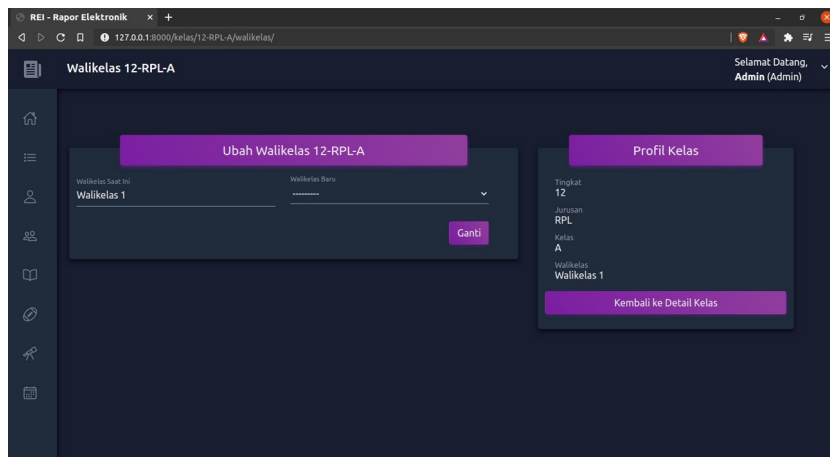


Gambar 4.17 Halaman Detail Kelas

#### 4.1.1.18. Halaman Pengaturan Walikelas

Halaman ini digunakan untuk mengubah walikelas dari suatu kelas dan hanya bisa diakses oleh guru yang memiliki akses staf atau admin. Akan ditampilkan sebuah *form* dimana *form* tadi memiliki sebuah isian “walikelas saat

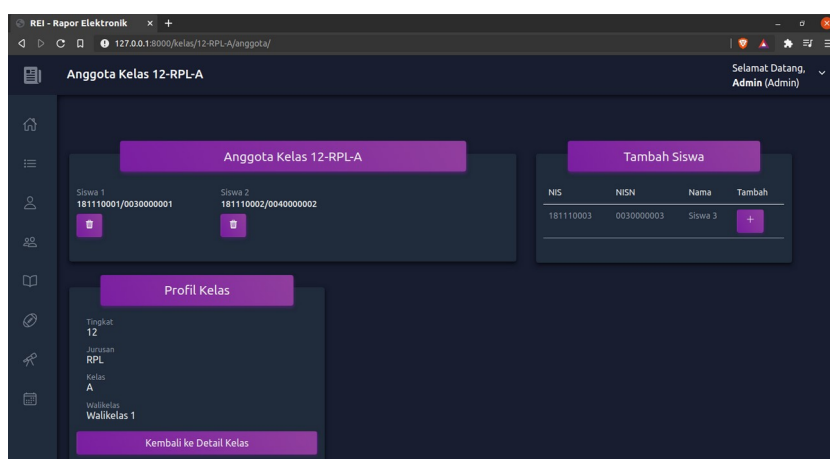
ini” yang bersifat *read-only* dan sebuah isian pilihan yang berisi akun walikelas yang belum memiliki kelas pada semester yang sedang aktif.



Gambar 4.18 Halaman Pengaturan Walikelas

#### 4.1.1.19. Halaman Pengaturan Anggota Kelas

Halaman ini hanya bisa diakses oleh guru yang memiliki akses staf atau admin. Terdiri dari dua bagian, pertama adalah daftar siswa yang sudah menjadi anggota kelas tersebut. Dan kedua adalah sebuah tabel berisi siswa yang belum memiliki kelas pada semester yang sedang aktif.

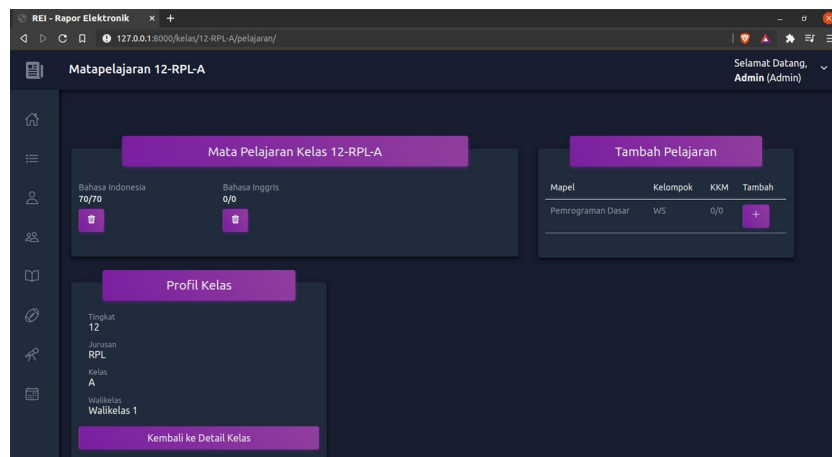


Gambar 4.19 Halaman Pengaturan Anggota Kelas



#### 4.1.1.20. Halaman Pengaturan Mapel Kelas

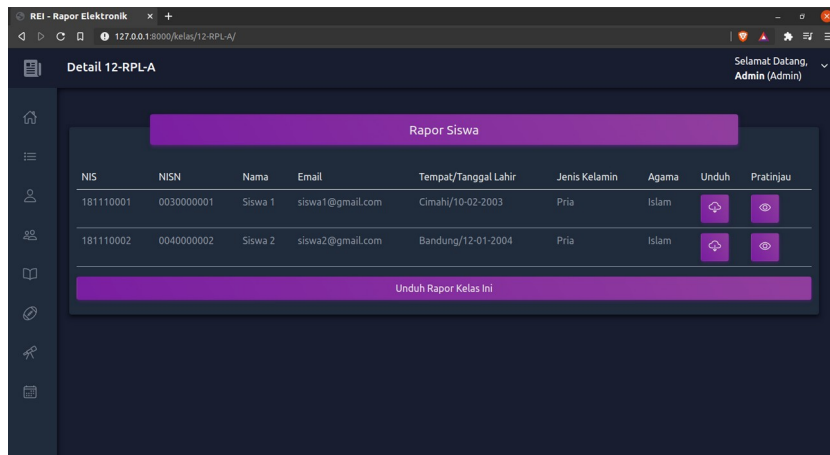
Halaman ini hanya bisa diakses oleh guru yang memiliki akses staf atau admin. Pada dasarnya halaman ini sama dengan halaman pengaturan anggota kelas. Hanya saja disini kita mengatur mata pelajaran apa yang akan diajarkan di suatu kelas.



Gambar 4.20 Halaman Pengaturan Mapel Kelas

#### 4.1.1.21. Halaman Cetak Rapor

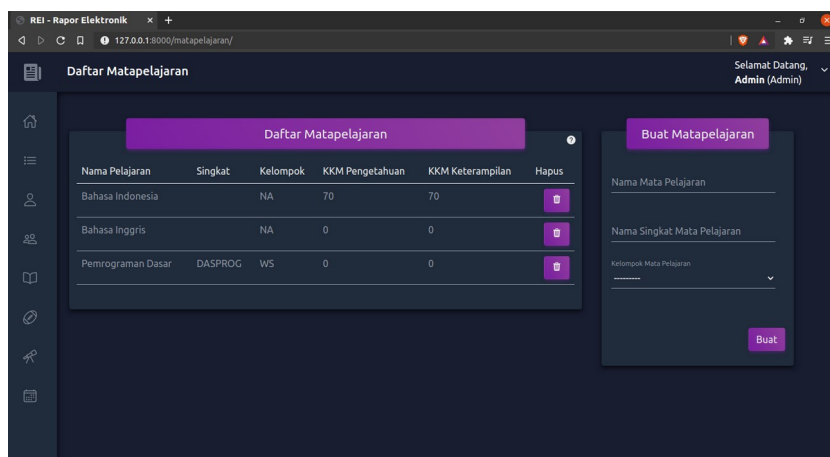
Halaman ini hanya bisa diakses oleh guru yang memiliki akses walikelas atau admin. Walikelas hanya bisa melakukan pencetakan rapor pada siswa yang diwalikan.



Gambar 4.21 Halaman Cetak Rapor

#### 4.1.1.22. Halaman Daftar Mata Pelajaran

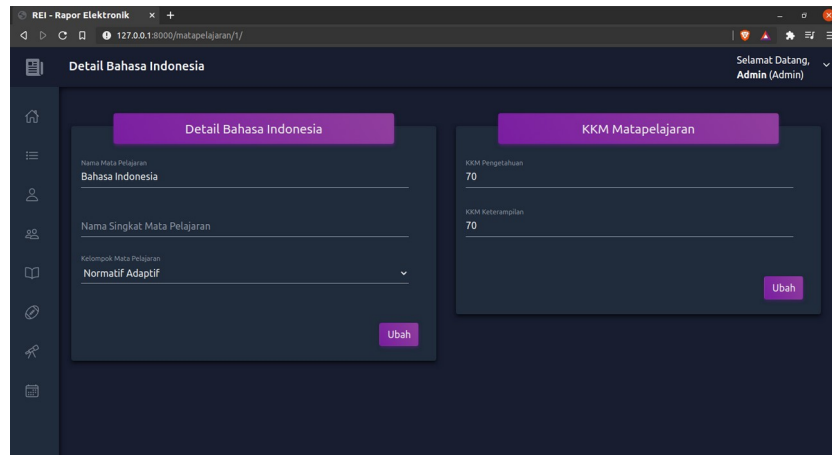
Halaman ini terdiri dari dua bagian. Pertama adalah tabel yang berisi daftar mata pelajaran dan KKM-nya berdasarkan semester yang sedang aktif. Bagian kedua adalah *form* pembuatan mata pelajaran. Aksi membuat dan menghapus mata pelajaran hanya bisa dilakukan oleh guru yang memiliki akses staf atau admin.



Gambar 4.22 Halaman Daftar Mata Pelajaran

#### 4.1.1.23. Halaman Detail Mata Pelajaran

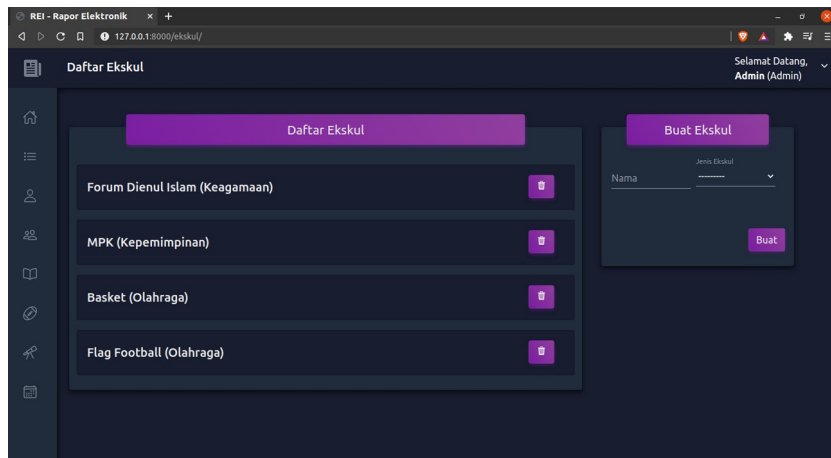
Halaman ini hanya bisa diakses oleh guru yang memiliki akses staf atau admin. Halaman ini terdiri dari *form* untuk mengubah mata pelajaran itu sendiri, dan *form* untuk mengubah KKM dari mata pelajaran tersebut.



Gambar 4.23 Halaman Detail Mata Pelajaran

#### 4.1.1.24. Halaman Daftar Ekskul

Halaman ini terdiri dari dua bagian. Bagian pertama adalah daftar ekskul yang sudah dibuat beserta tombol untuk menghapus ekskul tersebut. Bagian kedua adalah *form* untuk membuat sebuah ekskul. Aksi membuat dan menghapus ekskul hanya bisa dilakukan oleh guru yang mempunyai akses staf atau admin.



Gambar 4.24 Halaman Daftar Ekskul

#### 4.1.2. Struktur Tabel

Struktur tabel dari aplikasi ini disusun menurut rancangan yang telah dibuat pada bab sebelumnya.

##### 4.1.2.1. Tabel Sekolah

Tabel 4.1 Tabel Sekolah

Kolom	Tipe Data	Keterangan
nama	char(255)	
tingkat	enum('SD', 'SMP', 'SMA', 'SMK')	
npsn	char(8)	
alamat	char(255)	
keluarahan	char(50)	
kecamatan	char(50)	

Kolom	Tipe Data	Keterangan
kota	char(50)	
provinsi	char(50)	
email	char(50)	
no_telepon	char(20)	
kode_pos	char(5)	
kepsek	char(255)	
nip_kepsek	char(18)	

Tabel sekolah merupakan *singleton model*, artinya jumlah *record* yang ada di dalam tabel ini tidak boleh lebih dari satu. Pembuatan dan penghapusan *record* tidak bisa dilakukan dan kita hanya bisa melakukan perubahan. Tabel ini digunakan untuk menampung informasi sekolah yang akan menggunakan aplikasi ini.

#### 4.1.2.2. Tabel Guru

Tabel 4.2 Tabel Guru

Kolom	Tipe Data	Keterangan
id	integer(11)	Primary key
nip	char(18)	Unique
password	char(10)	
nama	char(255)	

Kolom	Tipe Data	Keterangan
gender	enum('Pria', 'Wanita')	
email	char(50)	
tempat_lahir	char(50)	
tanggal_lahir	date	
agama	char(50)	
alamat	char(255)	
is_walikelas	boolean	
is_stafu	boolean	
is_admin	boolean	
is_active	boolean	

Tabel Guru memiliki kolom *id* karena merupakan *master table*. Kolom *nip* dibuat *unique* karena seharusnya tidak ada guru yang memiliki NIP yang sama. Kolom *is\_walikelas*, *is\_stafu*, dan *is\_admin* adalah kolom yang akan menentukan hak apa yang akan diberikan kepada akun guru tersebut. Kolom *is\_active* adalah kolom yang digunakan agar kita dapat melakukan *soft delete* bila kita akan menghapus sebuah akun guru, jadi daripada kita benar-benar menghapus *record* dari *database*, kita bisa membuat akun tadi nonaktif agar nanti bisa diaktifkan kembali jika memang perlu.

#### 4.1.2.3. Tabel Semester

Tabel 4.3 Tabel Semester

Kolom	Tipe Data	Keterangan
id	integer(11)	Primary key
tahun_mulai	char(4)	
tahun_akhir	char(4)	
semester	enum('Ganjil', 'Genap')	
is_active	boolean	
Unique together: (tahun_mulai, tahun_akhir, semester)		

Tabel Semester memiliki kolom *id* karena merupakan *master table*. Kolom *tahun\_mulai*, *tahun\_akhir*, dan *semester* diberikan *constraint unique together* artinya kombinasi ke-tiga kolom tadi tidak boleh sama atau harus *unique*. Kolom *is\_active* merupakan kolom yang akan menjadi acuan semester apa yang sedang aktif, jika kolom *is\_active* berisi *True* pada suatu *record* maka *record* tadi akan dijadikan semester aktif dan aplikasi akan menggunakan data berdasarkan semester itu. Hanya boleh ada 1 *record is\_active* yang bernilai *True*. Jika semester aktif sudah ada dan *user* mengaktifkan lagi semester lain, maka semester aktif yang lama akan dinonaktifkan digantikan oleh semester aktif yang baru.

#### 4.1.2.4. Tabel Jurusan

Tabel 4.4 Tabel Jurusan

Kolom	Tipe Data	Keterangan
id	integer(11)	Primary key

Kolom	Tipe Data	Keterangan
nama	char(255)	
nama_singkat	char(10)	Null

Tabel Jurusan memiliki kolom *id* karena merupakan *master table*. Nama jurusan lengkap seperti “Rekayasa Perangkat Lunak” disimpan di kolom *nama* dan singkatannya seperti “RPL” disimpan di kolom *nama\_singkat*. Kolom *nama\_singkat* memiliki *constraint null* berarti kolom ini boleh tidak memiliki isi.

#### 4.1.2.5. Tabel Mata Pelajaran

Tabel 4.5 Tabel Mata Pelajaran

Kolom	Tipe Data	Keterangan
id	integer(11)	Primary key
nama	char(255)	
nama_singkat	char(10)	Null
kelompok	enum('NA', 'WS', 'MULOK')	

Tabel Mata Pelajaran memiliki kolom *id* karena merupakan *master table*. Nama mata pelajaran lengkap seperti “Pemrograman Dasar” disimpan di kolom *nama* dan singkatannya seperti “Dasprog” disimpan di kolom *nama\_singkat*. Kolom *nama\_singkat* memiliki *constraint null* berarti kolom ini boleh tidak memiliki isi.



#### 4.1.2.6. Tabel KKM

Tabel 4.6 Tabel KKM

Kolom	Tipe Data	Keterangan
id_matapelajaran	integer(11)	Foreign key
id_semester	integer(11)	Foreign key
pengetahuan	smallint	
keterampilan	smallint	
Unique together: (id_matapelajaran, id_semester)		

Tabel KKM merupakan sebuah *transaction table* jadi kolom *primary key* tidak terlalu diperlukan. Kolom *id\_matapelajaran* merujuk pada kolom *id* pada tabel Mata Pelajaran dan kolom *id\_semester* merujuk pada kolom *id* pada tabel Semester. Kolom *id\_matapelajaran*, dan *id\_semester* diberikan *constraint unique together* artinya kombinasi ke-tiga kolom tadi tidak boleh sama atau harus *unique*.

#### 4.1.2.7. Tabel Ekskul

Tabel 4.7 Tabel Ekskul

Kolom	Tipe Data	Keterangan
id	integer(11)	Primary key
nama	char(255)	
jenis	enum('Kepemimpinan', 'Keagamaan', 'Kesenian', 'Olahraga', 'Lain-lain')	

Tabel Ekskul memiliki kolom *id* karena merupakan *master table*.

#### 4.1.2.8. Tabel Kelas

Tabel 4.8 Tabel Kelas

Kolom	Tipe Data	Keterangan
id	integer(11)	Primary key
tingkat	char(3)	
id_jurusan	integer(11)	Foreign key, Null
kelas	enum('A', 'B', 'C', 'D')	
id_matapelajaran	many-to-many	Foreign key
id_guru	integer(11)	Foreign key
id_semester	integer(11)	Foreign key
Unique together: (tingkat, id_jurusan, kelas, id_semester)		

Tabel Kelas memiliki kolom *id* karena merupakan *master table*. Kolom *id\_jurusan* merujuk pada kolom *id* pada tabel Jurusan dan diberikan *constraint null* agar jika kelas yang akan dibuat bukanlah kelas SMK maka kelas dapat tetap dibuat. Kolom *id\_guru* merujuk pada kolom *id* pada tabel Guru, kolom ini akan menjadi acuan tentang guru mana yang menjadi walikelas suatu kelas. Kolom *id\_matapelajaran* merupakan kolom *many-to-many*, fungsinya untuk mengetahui mata pelajaran apa saja yang diajarkan di suatu kelas.

#### 4.1.2.9. Tabel Siswa

Tabel 4.9 Tabel Siswa

Kolom	Tipe Data	Keterangan
id	Integer(11)	Primary key
nisl	char(10)	Unique
nisl	char(9)	Unique
nama	char(255)	
gender	enum('Pria', 'Wanita')	
email	char(50)	
tempat_lahir	char(50)	
tanggal_lahir	date	
agama		
alamat	char(255)	
sekolah_asal	char(50)	
nama_ayah	char(255)	
nama_ibu	char(255)	
id_kelas	integer(11)	Foreign key

Tabel Siswa memiliki kolom *id* karena merupakan *master table*. Kolom *id\_kelas* merujuk pada kolom *id* pada tabel Kelas. Kolom *nisl* dan *nisl* diberi

*constraint unique* karena seharusnya tidak ada siswa yang memiliki NIS atau NISN yang sama. Kolom lainnya adalah kolom yang nanti nilainya akan digunakan pada halaman biodata siswa dalam rapor digital.

#### 4.1.2.10. Tabel Absensi

Tabel 4.10 Tabel Absensi

Kolom	Tipe Data	Keterangan
id_siswa	integer(11)	Foreign key
id_semester	integer(11)	Foreign key
izin	smallint	
sakit	smallint	
bolos	smallint	
Unique together: (id_siswa, id_semester)		

Tabel Absensi merupakan sebuah *transaction table* jadi kolom *primary key* tidak terlalu diperlukan. Kolom *id\_siswa* merujuk pada kolom *id* pada tabel Siswa dan kolom *id\_semester* merujuk pada kolom *id* pada tabel Semester. Kedua kolom tadi diberikan *constraint unique together*.

#### 4.1.2.11. Tabel Nilai Mata Pelajaran

Tabel 4.11 Tabel Nilai Mata Pelajaran

Kolom	Tipe Data	Keterangan
id_siswa	integer(11)	Foreign key
id_matapelajaran	integer(11)	Foreign key

Kolom	Tipe Data	Keterangan
id_semester	integer(11)	Foreign key
nilai_pengetahuan	smallint	
nilai_keterampilan	smallint	
Unique together: (id_siswa, id_matapelajaran, id_semester)		

Tabel Nilai Mata Pelajaran merupakan sebuah *transaction table* jadi kolom *primary key* tidak terlalu diperlukan. Kolom *id\_siswa* merujuk pada kolom *id* pada tabel Siswa, kolom *id\_semester* merujuk pada kolom *id* pada tabel Semester, dan kolom *id\_matapelajaran* merujuk pada kolom *id* pada tabel Mata Pelajaran. Ketiga kolom tadi diberikan *constraint unique together*.

#### 4.1.2.12. Tabel Nilai Ekskul

Tabel 4.12 Tabel Nilai Ekskul

Kolom	Tipe Data	Keterangan
id_siswa	integer(11)	Foreign key
id_ekskul	integer(11)	Foreign key
id_semester	integer(11)	Foreign key
nilai	enum('A', 'B', 'C', 'D')	
Unique together: (id_siswa, id_ekskul, id_semester)		

Tabel Nilai Ekskul merupakan sebuah *transaction table* jadi kolom *primary key* tidak terlalu diperlukan. Kolom *id\_siswa* merujuk pada kolom *id* pada tabel Siswa, kolom *id\_semester* merujuk pada kolom *id* pada tabel Semester, dan kolom *id\_ekskul* merujuk pada kolom *id* pada tabel Ekskul. Ketiga kolom tadi diberikan *constraint unique together*.

#### 4.1.2.13. Tabel Rapor

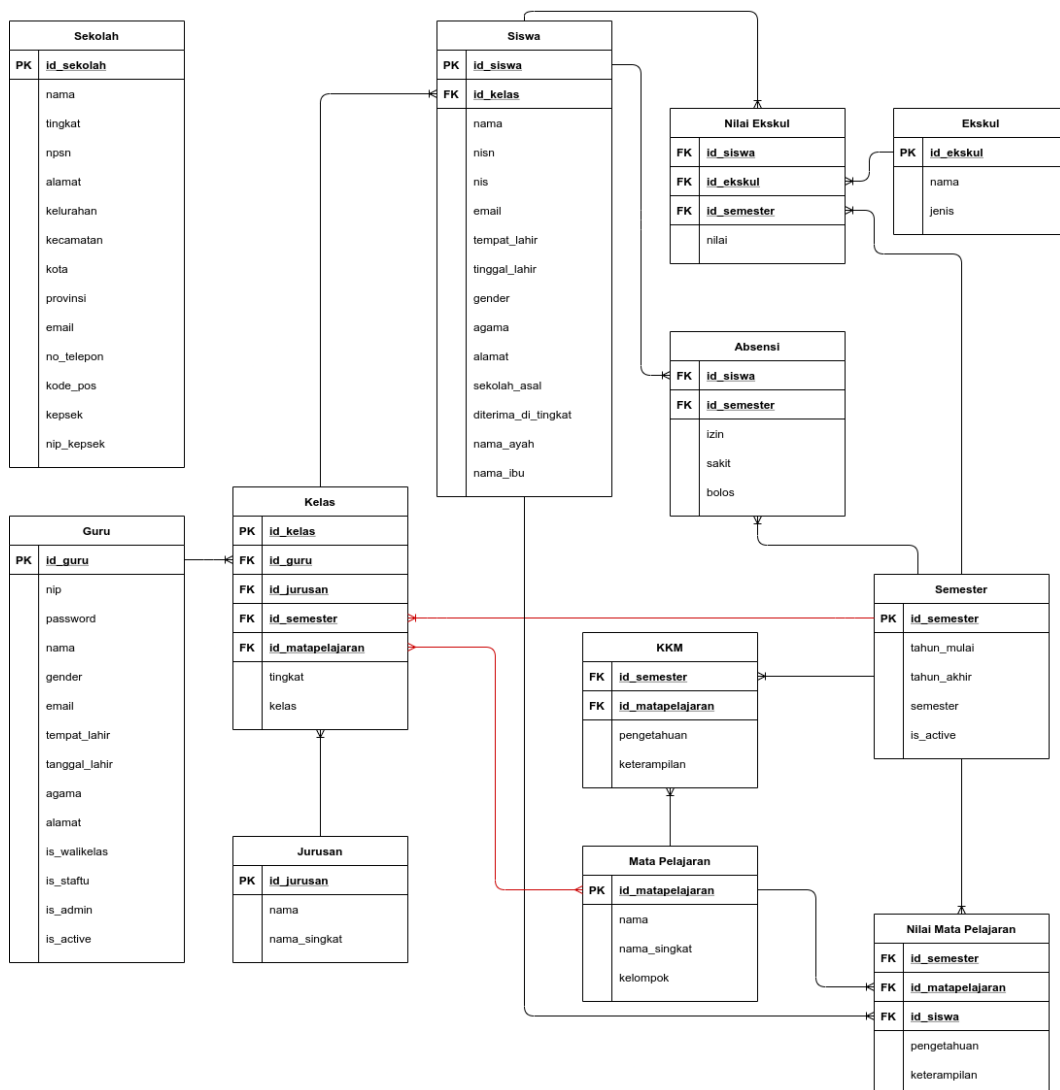
Tabel 4.13 Tabel Rapor

Kolom	Tipe Data	Keterangan
id_siswa	integer(11)	Foreign key
id_semester	integer(11)	Foreign key
rapor	varchar	
Unique together: (id_siswa, id_semester)		

Tabel Nilai Rapor merupakan sebuah *transaction table* jadi kolom *primary key* tidak terlalu diperlukan. Kolom *id\_siswa* merujuk pada kolom *id* pada tabel Siswa dan kolom *id\_semester* merujuk pada kolom *id* pada tabel Semester. Kedua kolom tadi diberikan *constraint unique together*. Kolom *rapor* digunakan untuk menyimpan lokasi berkas PDF rapor.

#### 4.1.3. Relasi Tabel

Berikut adalah relasi tabel aplikasi ini setelah mengetahui seluruh struktur *database* yang sudah diimplementasikan.



Gambar 4.25 Relasi Tabel

## 4.2. Pengujian

Untuk mengecek apakah aplikasi berjalan dengan baik, serangkaian pengujian dilakukan pada fitur-fitur aplikasi. Perlu diingat, jika ada tahap pengisian data pada fitur yang diuji, data yang digunakan dalam pengujian adalah data tes, bukan data sungguhan atau *production data*.

### 4.2.1. Login

Berikut adalah tabel pengujian sistem *login*.

Tabel 4.14 Pengujian Sistem Login

No	Kasus	Ekspektasi	Hasil
1	<i>Login</i> dengan NIP yang tidak terdaftar.	Proses <i>login</i> gagal dan pengguna diarahkan kembali ke halaman <i>login</i> .	Sesuai ekspektasi



# Rapor Elektronik Indonesia

Nomor Induk

0555

Password

\* Nomor Induk atau Password salah silahkan  
coba lagi

Login

Gambar 4.26 Form Login Saat NIP Tidak Terdaftar

2	<i>Login</i> dengan NIP yang sudah terdaftar tetapi dengan <i>password</i> yang tidak sesuai.	Proses <i>login</i> gagal dan pengguna diarahkan kembali ke halaman <i>login</i> .	Sesuai ekspektasi
---	---	--	-------------------

# Rapor Elektronik Indonesia

Nomor Induk

11111

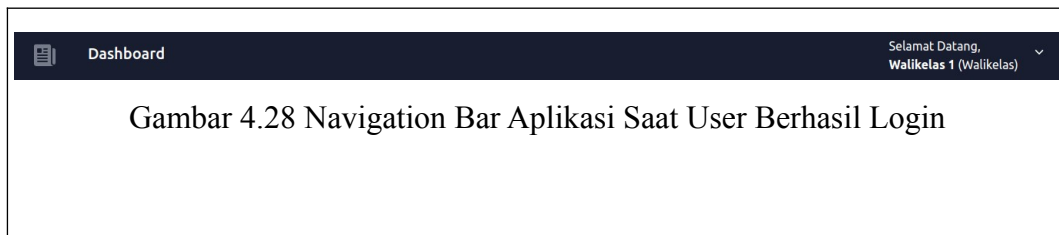
Password

\* Nomor Induk atau Password salah silahkan  
coba lagi

Login

Gambar 4.27 Form Login Saat NIP Sudah Terdaftar Tetapi Password Salah

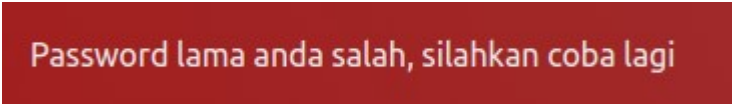
3	<i>Login</i> dengan NIP yang sudah terdaftar dan <i>password</i> yang sesuai.	Proses <i>login</i> berhasil dan pengguna diarahkan ke halaman <i>dashboard</i> dengan nama pengguna yang tepat tertulis di kanan atas halaman.	Sesuai ekspektasi
---	---	---	-------------------



#### 4.2.2. Penggantian Password

Berikut adalah tabel pengujian sistem penggantian *password* pengguna.

Tabel 4.15 Pengujian Sistem Penggantian Password

No	Kasus	Ekspektasi	Hasil
1	Mengisi <i>form</i> ganti <i>password</i> dengan <i>password</i> lama yang tidak sesuai.	Proses penggantian <i>password</i> akan gagal dan akan muncul pemberitahuan bahwa <i>password</i> lama yang pengguna isi salah.	Sesuai ekspektasi
 <p>Gambar 4.29 User Gagal Mengubah Password</p>			
2	Mengisi <i>form</i> ganti <i>password</i> dengan <i>password</i> lama yang sudah sesuai tetapi isian <i>password</i> baru dan konfirmasi <i>password</i> baru tidak sama.	Proses penggantian <i>password</i> akan gagal dan akan muncul pemberitahuan bahwa isian konfirmasi <i>password</i> tidak sama.	Sesuai ekspektasi

Konfirmasi password baru anda salah

Gambar 4.30 User Gagal Mengubah Password

3	Mengisi <i>form</i> ganti <i>password</i> dengan <i>password</i> lama yang sudah sesuai juga <i>password baru</i> dan konfirmasi <i>password</i> baru yang sama.	Proses penggantian <i>password</i> berhasil, lalu pengguna akan di- <i>logout</i> dan diarahkan ke halaman <i>login</i> .	Sesuai ekspektasi
---	--	---	-------------------

Password anda berhasil diganti

**Rapor Elektronik Indonesia**

Nomor Induk  
|

Password  
|

Login

Gambar 4.31 User Berhasil Mengubah Password

#### 4.2.3. Manajemen Semester

Berikut adalah tabel pengujian sistem manajemen semester.

Tabel 4.16 Pengujian Sistem Manajemen Semester

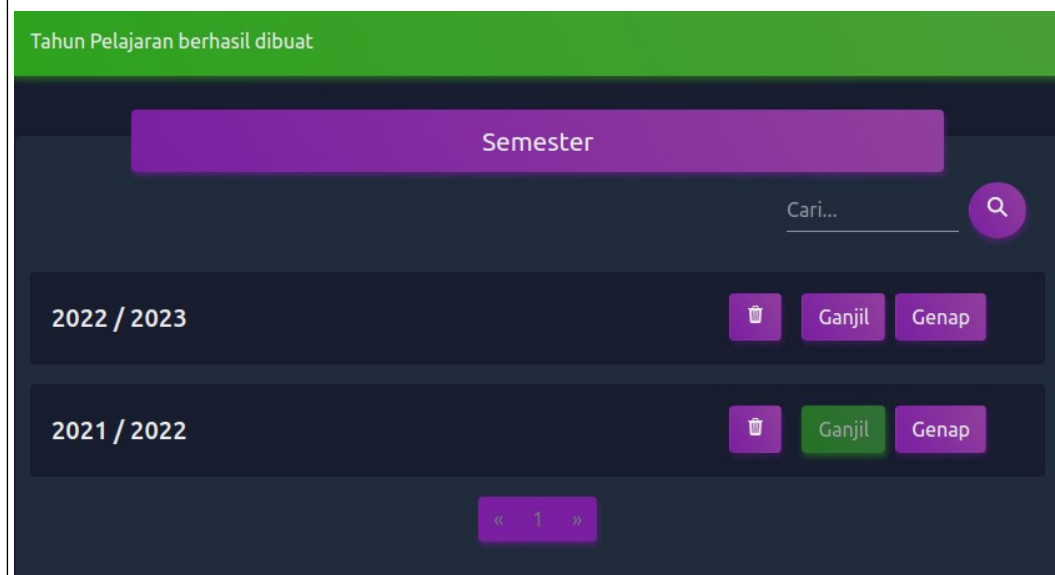
No	Kasus	Ekspektasi	Hasil
----	-------	------------	-------

1	Membuat semester dengan tahun mulai dan tahun akhir yang sudah pernah ada.	Proses pembuatan semester gagal dan muncul pemberitahuan bahwa semester tersebut sudah tersedia.	Sesuai ekspektasi
---	--	--	-------------------

Tahun Pelajaran dengan data persis seperti itu sudah ada

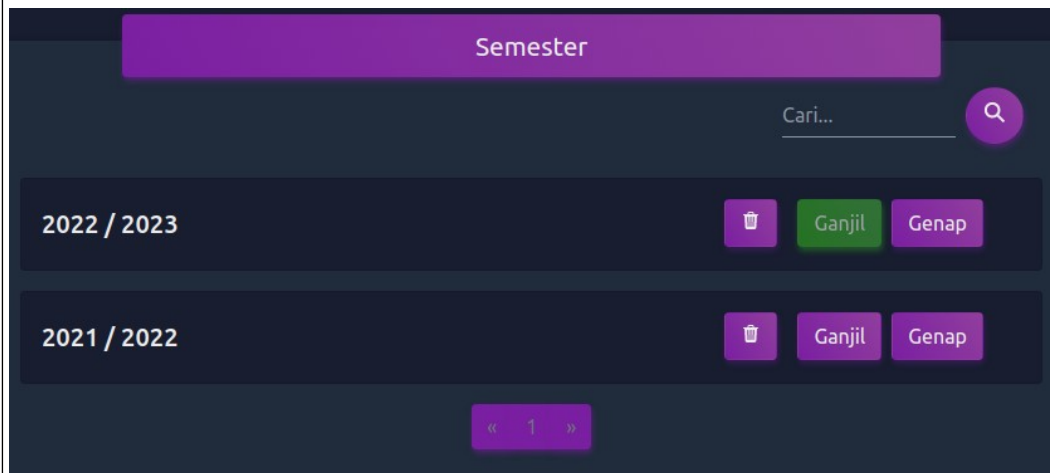
Gambar 4.32 Pembuatan Tahun Pelajaran Gagal

2	Membuat semester dengan tahun mulai dan tahun akhir baru.	Proses pembuatan semester berhasil dan daftar semester akan diperbarui, juga muncul pemberitahuan bahwa pembuatan semester telah berhasil dijalankan.	Sesuai ekspektasi
---	---	---	-------------------



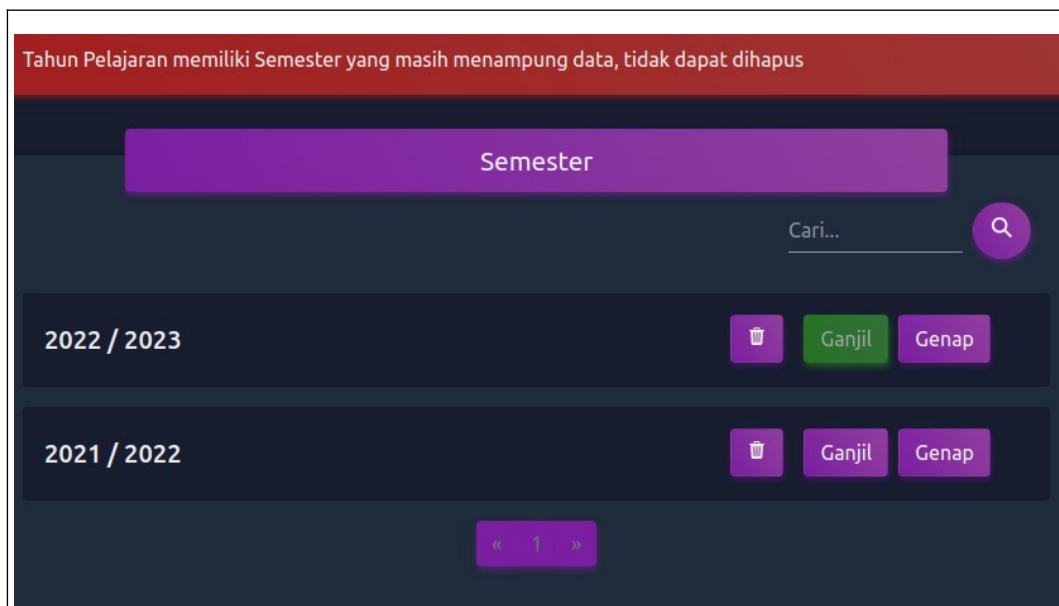
Gambar 4.33 Pembuatan Tahun Pelajaran Berhasil

3	Mengaktifkan semester saat sudah ada semester yang sedang aktif.	Semester aktif lama akan menjadi nonaktif dan semester baru yang diaktifkan akan menjadi semester aktif.	Sesuai ekspektasi
---	--	--	-------------------



Gambar 4.34 Penggantian Semester Aktif

4	Menghapus semester yang sudah memiliki data	Semester tidak akan bisa terhapus dan akan muncul pemberitahuan bahwa semester tidak dapat dihapus.	Sesuai ekspektasi
---	---	---	-------------------



Gambar 4.35 Penghapusan Semester Gagal

#### 4.2.4. Penambahan Guru

Berikut adalah tabel pengujian penambahan akun guru.

Tabel 4.17 Pengujian Penambahan Guru

No	Kasus	Ekspektasi	Hasil
1	Menambah guru dengan NIP yang belum ada di dalam <i>database</i> .	Proses pembuatan akun guru berhasil dan daftar data guru diperbarui, juga muncul pemberitahuan bahwa pembuatan walikelas telah berhasil dijalankan.	Sesuai ekspektasi

Akun berhasil dibuat

Gambar 4.36 Pembuatan Akun Guru Berhasil

2	Menambah guru dengan NIP yang sudah ada di dalam <i>database</i> .	Proses pembuatan akun guru gagal dan muncul pemberitahuan bahwa pembuatan akun guru telah gagal dijalankan.	Sesuai ekspektasi
---	--	---	-------------------

Akun dengan NIP itu sudah ada

Gambar 4.37 Pembuatan Akun Guru Gagal

#### 4.2.5. Manajemen Jurusan

Berikut adalah tabel pengujian untuk sistem manajemen jurusan.

Tabel 4.18 Pengujian Manajemen Jurusan

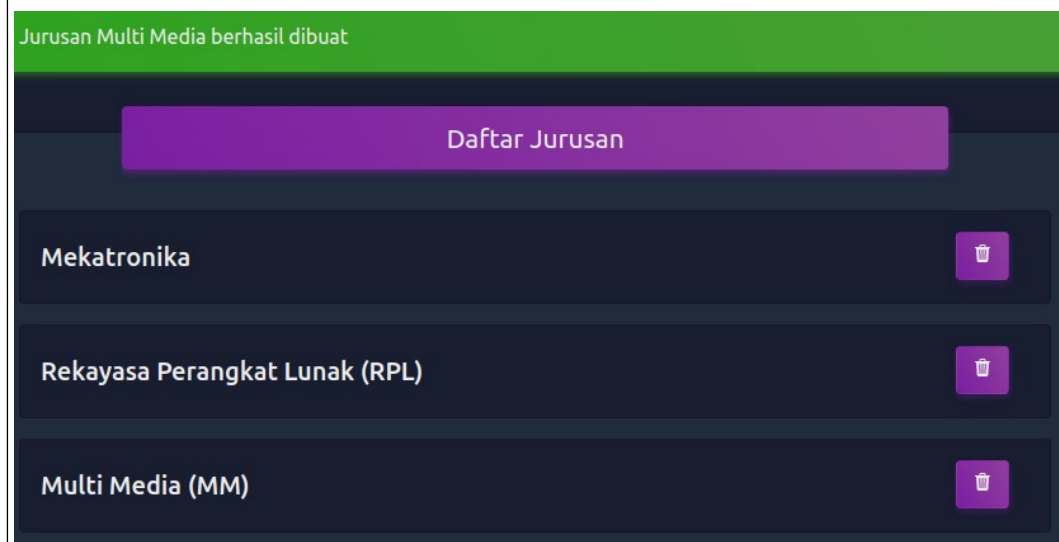
No	Kasus	Ekspektasi	Hasil
1	Menambah jurusan dengan nama dan singkatan jurusan yang sama dengan jurusan yang sudah ada.	Proses pembuatan jurusan gagal dan muncul pemberitahuan bahwa jurusan tadi sudah tersedia. Karena tabel jurusan memiliki <i>constraint unique together</i> antara nama dan singkatan jurusan.	Sesuai ekspektasi






Gambar 4.38 Pembuatan Jurusan Gagal

2	Menambah jurusan dengan nama dan singkatan jurusan yang baru.	Jurusan baru tercatat dan daftar jurusan diperbarui. Pemberitahuan juga akan muncul yang mengkonfirmasi keberhasilan pembuatan jurusan.	Sesuai ekspektasi
---	---	--	-------------------



Gambar 4.39 Pembuatan Jurusan Berhasil

3	Menghapus jurusan yang	Jurusan terhapus dan daftar	Sesuai
---	------------------------	-----------------------------	--------

	belum memiliki kelas.	jurusan diperbarui. Pemberitahuan juga akan muncul yang mengkonfirmasi penghapusan jurusan.	ekspektasi
 <p>Multi Media berhasil dihapus dari daftar jurusan sekolah</p> <p>Daftar Jurusan</p> <p>Mekatronika</p> <p>Rekayasa Perangkat Lunak (RPL)</p>			
Gambar 4.40 Penghapusan Jurusan Berhasil			
4	Menghapus jurusan yang sudah memiliki kelas.	Jurusan tidak terhapus dan muncul pemberitahuan bahwa jurusan yang masih memiliki kelas tidak dapat dihapus.	Sesuai ekspektasi



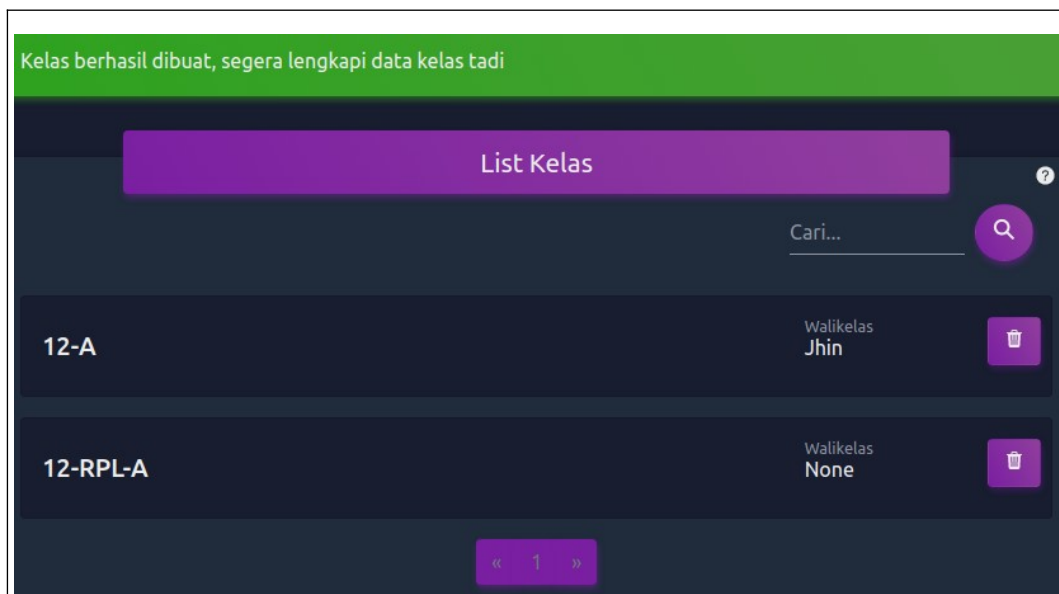
Gambar 4.41 Penghapusan Jurusan Gagal

#### 4.2.6. Manajemen Kelas

Berikut adalah tabel pengujian untuk sistem manajemen kelas.

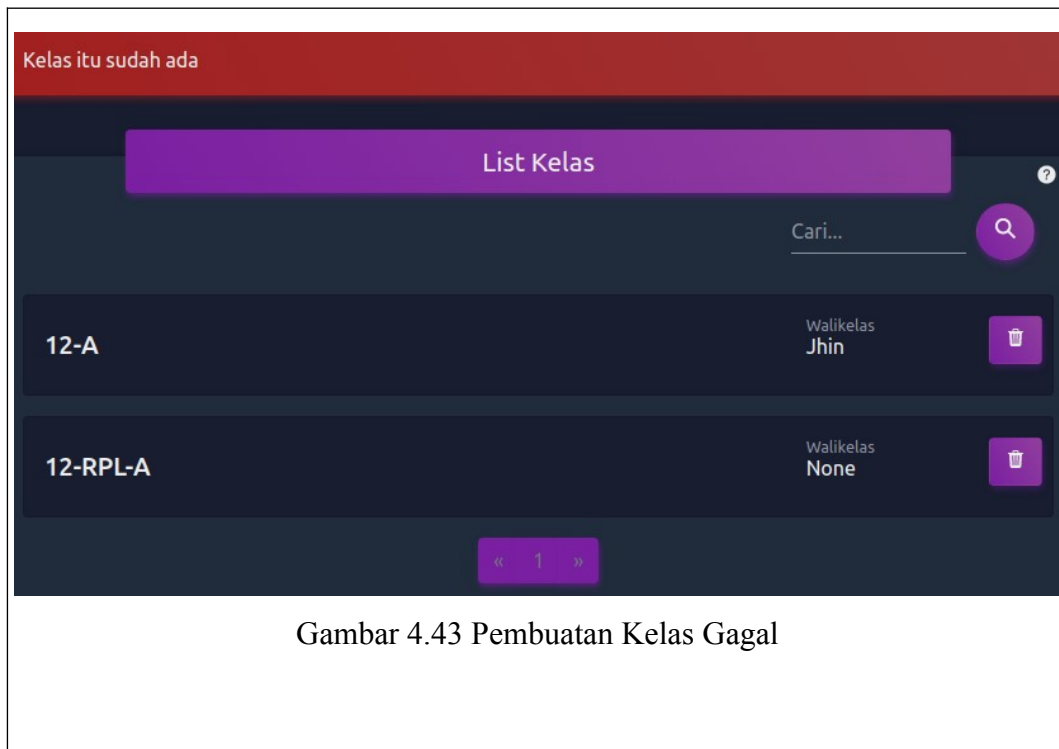
Tabel 4.19 Pengujian Manajemen Kelas

No	Kasus	Ekspektasi	Hasil
1	Membuat kelas baru dengan tingkat, jurusan dan kelas yang tidak bentrok.	Kelas baru tercatat dan daftar kelas diperbarui. Pemberitahuan juga akan muncul yang mengkonfirmasi keberhasilan pembuatan kelas.	Sesuai ekspektasi



Gambar 4.42 Pembuatan Kelas Berhasil

2	Membuat kelas baru dengan tingkat, jurusan, dan kelas yang bentrok dengan kelas lain yang berada pada semester yang sama.	Kelas tidak tercatat karena kelas tadi sudah tersedia. Pemberitahuan yang mengkonfirmasi kegagalan pembuatan kelas.	Sesuai ekspektasi
---	---	---	-------------------

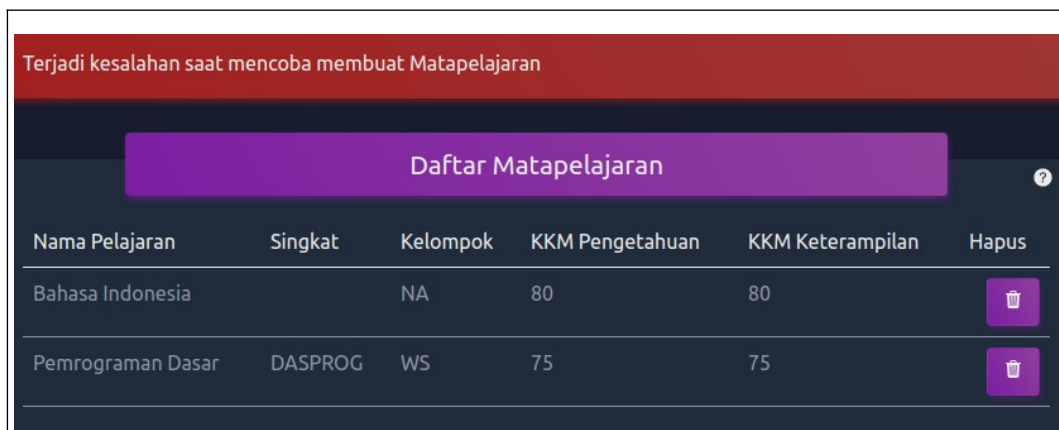


#### 4.2.7. Manajemen Mata Pelajaran dan KKM

Berikut adalah tabel pengujian untuk sistem manajemen mata pelajaran dan KKM-nya.

Tabel 4.20 Pengujian Manajemen Mata Pelajaran dan KKM

No	Kasus	Ekspektasi	Hasil
1	Membuat mata pelajaran dengan nama, singkatan, dan kelompok pelajaran yang bentrok dengan mata pelajaran yang sudah tersedia.	Mata pelajaran tidak akan tercatat karena bentrok dengan mata pelajaran yang sudah ada. Pemberitahuan juga akan muncul yang mengkonfirmasi kegagalan penambahan mata pelajaran.	Sesuai ekspektasi



Gambar 4.44 Pembuatan Mata Pelajaran Gagal

2	Membuat KKM dengan nilai lebih dari 100 atau kurang dari 0.	KKM tidak akan ditambahkan.	Tidak sesuai ekspektasi. Aplikasi akan otomatis menetapkan KKM menjadi 0.
---	---	-----------------------------	---

Nilai yang di-input tidak diantara 0-100. Aplikasi otomatis menyimpan 0 ke dalam database

Gambar 4.45 Pengubahan Nilai KKM Gagal

#### 4.2.8. Manajemen Ekskul

Berikut adalah tabel pengujian sistem manajemen ekstrakurikuler.

Tabel 4.21 Pengujian Manajemen Ekskul

No	Kasus	Ekspektasi	Hasil
1	Membuat ekskul dengan	Ekskul tidak akan tercatat	Sesuai

	nama dan jenis ekstrakurikuler yang bentrok dengan ekstrakurikuler yang sudah tersedia.	karena bentrok dengan ekstrakurikuler yang sudah ada. Pemberitahuan juga akan muncul yang mengkonfirmasi kegagalan penambahan ekstrakurikuler.	ekspektasi
--	---	--	------------

Ekskul dengan data persis seperti itu sudah tersedia

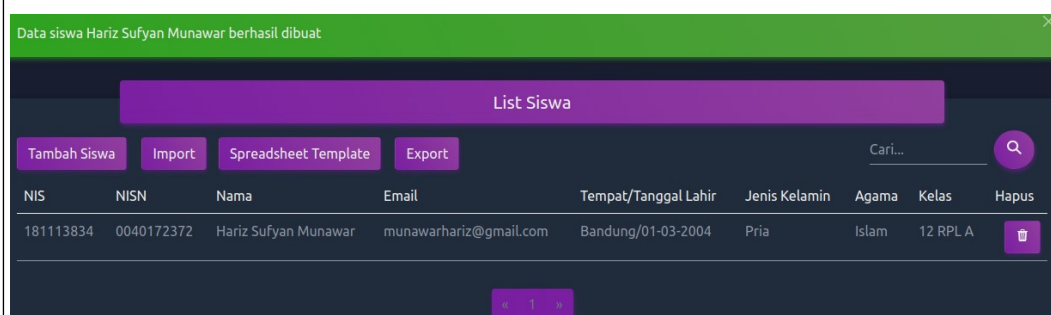
Gambar 4.46 Pembuatan Ekskul Gagal

#### 4.2.9. Manajemen Siswa

Berikut adalah tabel pengujian sistem manajemen siswa.

Tabel 4.22 Pengujian Manajemen Siswa

No	Kasus	Ekspektasi	Hasil
1	Membuat siswa dengan NIS atau NISN baru.	Data siswa tercatat ke <i>database</i> . Pemberitahuan juga akan muncul yang mengkonfirmasi keberhasilan pembuatan siswa.	Sesuai ekspektasi



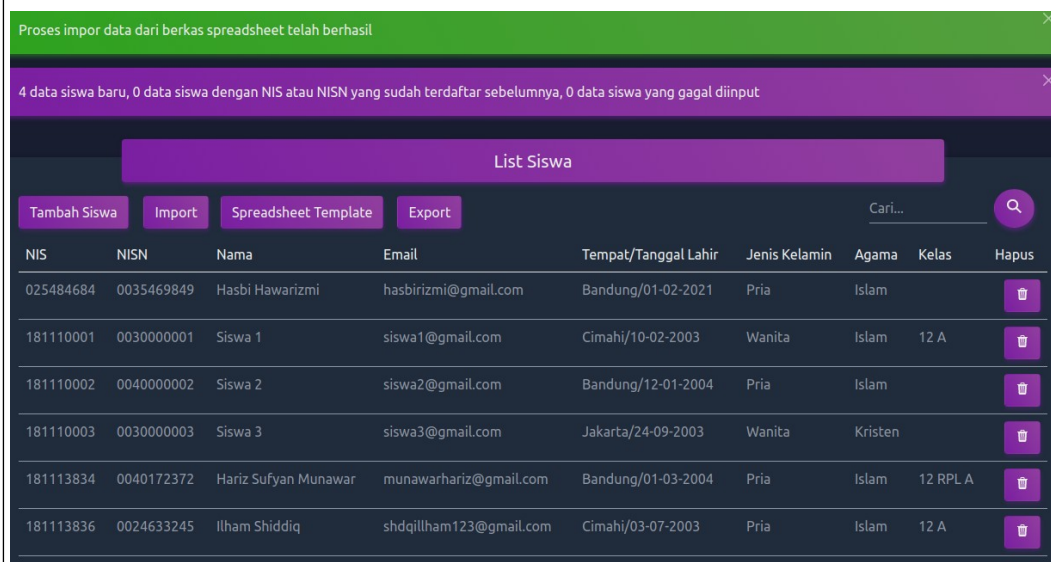
Gambar 4.47 Pembuatan Siswa Berhasil

2	Membuat siswa dengan NIS atau NISN yang sama dengan siswa lain yang sudah terdaftar.	Data siswa gagal tercatat karena data NIS atau NISN harus <i>unique</i> .	Sesuai ekspektasi
---	--	---	-------------------

Data gagal dibuat, periksa apakah NIS atau NISN sudah tersedia didata sebelumnya

Gambar 4.48 Pembuatan Siswa Gagal

3	Melakukan proses impor data siswa	Data siswa yang tercatat di dalam berkas excel akan masuk ke dalam <i>database</i> selama data yang terdapat di berkas excel valid.	Sesuai ekspektasi
---	-----------------------------------	---	-------------------



Proses impor data dari berkas spreadsheet telah berhasil

4 data siswa baru, 0 data siswa dengan NIS atau NISN yang sudah terdaftar sebelumnya, 0 data siswa yang gagal diinput

List Siswa

Tambah Siswa Import Spreadsheet Template Export

Cari...

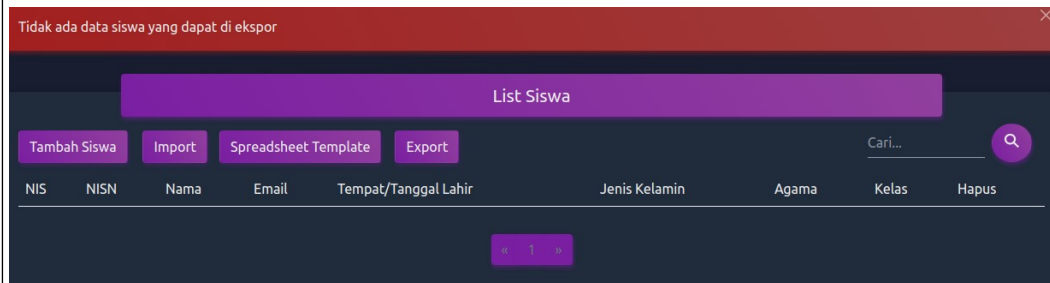
NIS	NISN	Nama	Email	Tempat/Tanggal Lahir	Jenis Kelamin	Agama	Kelas	Hapus
025484684	0035469849	Hasbi Hawarizmi	hasbirizmi@gmail.com	Bandung/01-02-2021	Pria	Islam		
181110001	0030000001	Siswa 1	siswa1@gmail.com	Cimahi/10-02-2003	Wanita	Islam	12 A	
181110002	0040000002	Siswa 2	siswa2@gmail.com	Bandung/12-01-2004	Pria	Islam		
181110003	0030000003	Siswa 3	siswa3@gmail.com	Jakarta/24-09-2003	Wanita	Kristen		
181113834	0040172372	Hariz Sufyan Munawar	munawarhariz@gmail.com	Bandung/01-03-2004	Pria	Islam	12 RPL A	
181113836	0024633245	Ilham Shiddiq	shdqillham123@gmail.com	Cimahi/03-07-2003	Pria	Islam	12 A	

Gambar 4.49 Proses Impor Data Siswa Berhasil

4	Melakukan proses ekspor data siswa saat belum ada	Proses ekspor akan gagal dan akan muncul sebuah	Sesuai ekspektasi
---	---	---	-------------------

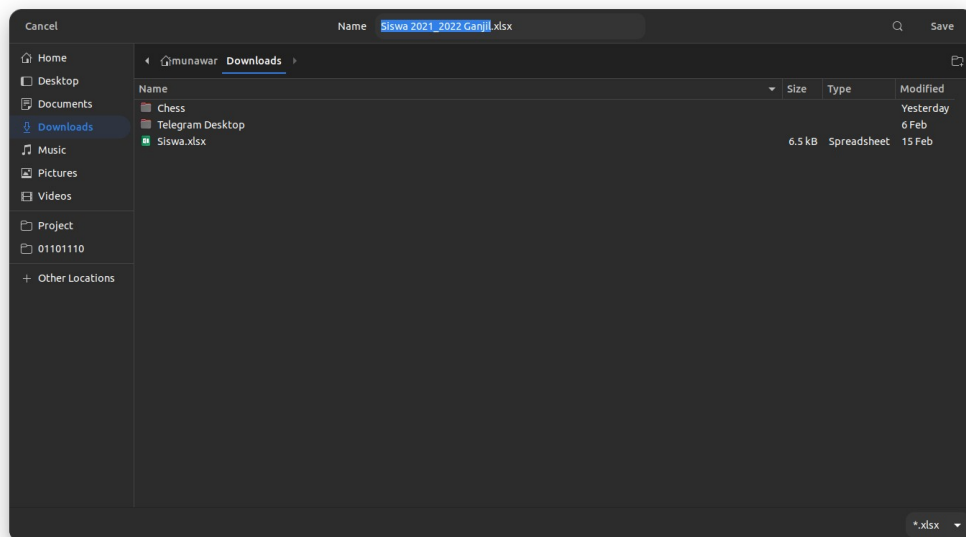


	data siswa yang terdaftar	pemberitahuan.	
--	---------------------------	----------------	--



Gambar 4.50 Proses Ekspor Data Siswa Gagal

5	Melakukan proses ekspor data siswa setelah sudah ada data siswa yang terdaftar	Semua data siswa akan diubah menjadi sebuah berkas <i>spreadsheet</i> dan akan muncul <i>window download</i> agar berkas tadi bisa diunduh	Sesuai ekspektasi
---	--	--	-------------------

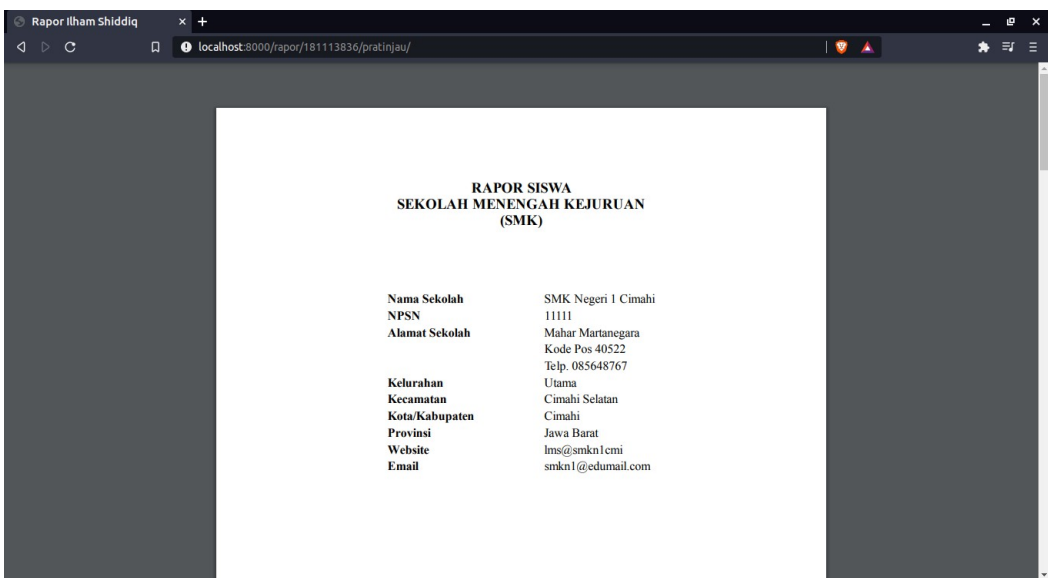


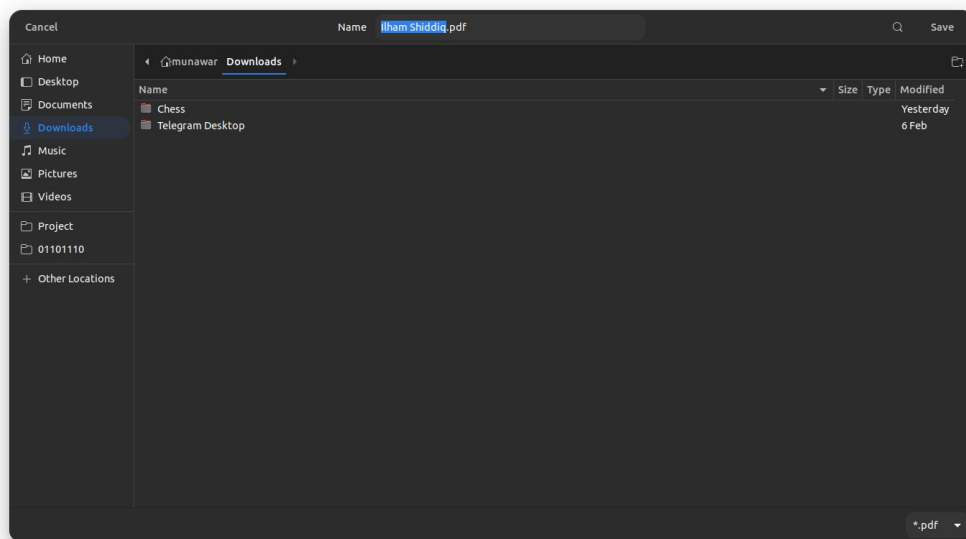
Gambar 4.51 Ekspor Data Siswa Berhasil

#### 4.2.10. Cetak Rapor

Berikut adalah tabel pengujian sistem pencetakan rapor.

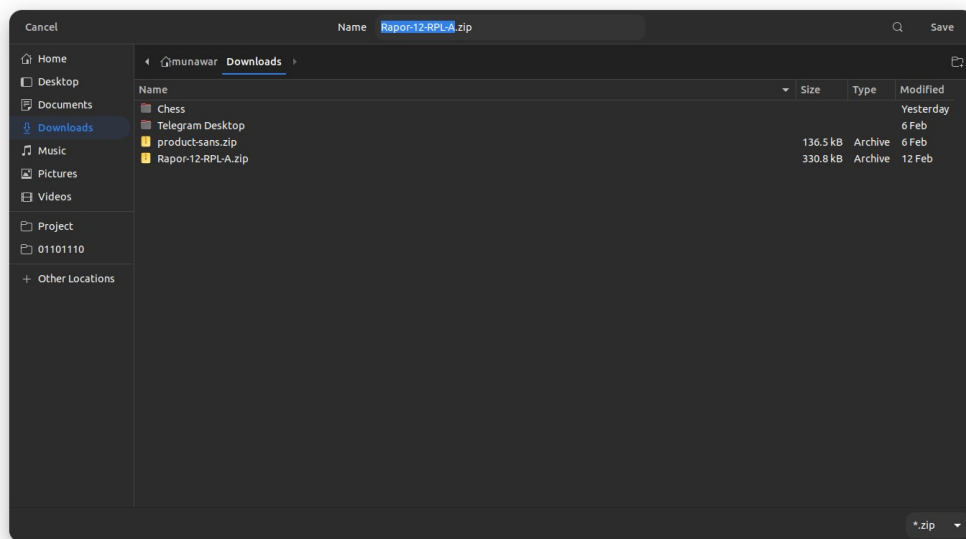
Tabel 4.23 Pengujian Pencetakan Rapor

No	Kasus	Ekspektasi	Hasil
1	Melihat pratinjau rapor seorang siswa.	Pengguna akan diarahkan ke halaman yang isinya merupakan sebuah berkas PDF yang di- <i>embed</i> .	Sesuai ekspektasi
			
Gambar 4.52 Pratinjau Rapor Digital			
2	Mengunduh rapor seorang siswa.	Muncul sebuah <i>window</i> pada <i>browser</i> untuk mengunduh berkas yang berisi rapor siswa tersebut.	Sesuai ekspektasi



Gambar 4.53 Unduh Rapor Digital

3	Mengunduh rapor suatu kelas.	Muncul sebuah <i>window</i> pada <i>browser</i> untuk mengunduh berkas ZIP yang berisi rapor dari siswa-siswa pada kelas tersebut.	Sesuai ekspektasi
---	------------------------------	--	-------------------



Gambar 4.54 Unduh Rapor Digital Per Kelas

## **BAB V**

### **PENUTUP**

#### **5.1. Kesimpulan**

Beberapa kesimpulan yang dapat diambil dari keseluruhan proses pembuatan aplikasi ini adalah.

1. Pengolahan data siswa ke dalam bentuk rapor menjadi lebih mudah daripada dengan cara manual.
2. Aplikasi ini telah menyediakan sistem *database* berupa struktur tabel yang dapat diimplementasikan pada komputer *database server*.
3. Dengan menempatkan *database* sistem aplikasi ini pada komputer *server* milik sekolah, penyimpanan data sekolah menjadi terpusat dan terorganisir.

#### **5.2. Saran**

Aplikasi ini masih dapat dikembangkan lagi ke depannya sehingga memiliki banyak fitur yang bermanfaat. Beberapa poin yang dapat dikembangkan dari aplikasi ini adalah:

1. Membuat sistem *failsafe* untuk mengatasi kasus di mana semua staf kehilangan akses pada akun mereka.
2. Memperbaiki tampilan aplikasi saat dibuka menggunakan perangkat yang memiliki resolusi layar kecil seperti ponsel genggam.
3. Membuat fitur “*send rapor as email*” agar mempermudah pembagian rapor.

## DAFTAR PUSTAKA

Nugraha, Dani. 2016. Aplikasi Rapor Kurikulum 2013. Cimahi: Laporan Praktik Kerja Industri Rekayasa Perangkat Lunak SMKN 1 Cimahi.

Esaputra, Fiska. 2020. *Ngoding Web Dinamis atau Statis, Apa Perbedaannya?*. Diakses pada 16 Januari 2021, dari <https://www.dicoding.com/blog/ngoding-web-dinamis-atau-statis/>

Pembahasan Mendalam Mengenai Python. Diakses pada 16 Januari 2021, dari <https://belajarpython.com/tutorial/apa-itu-python>

Hanafiah, Ramadhan Lutfi. 2017. *Belajar Django*. Diakses pada 23 Januari 2021, dari <https://medium.com/@ramadhan/belajar-django-1-c561326be15f>

Dokumentasi Weasyprint. Diakses pada 23 Januari 2021, dari <https://weasyprint.readthedocs.io/en/stable/>

*What Is SQL?*. Diakses pada 24 Januari 2021, dari <http://www.sqlcourse.com/intro.html>

*What Is DBMS*. 2019. Diakses pada 24 Januari 2021, dari <https://www.edureka.co/blog/what-is-dbms>

*Flowmap atau Flowchart*. Diakses pada 25 Januari 2021, dari <https://en.wikipedia.org/wiki/Flowchart>

Patni, Shubham. 2020. *What Is Data Flow Diagram*. Diakses pada 25 Januari 2021, dari <https://www.geeksforgeeks.org/what-is-dfddata-flow-diagram/>

*What Is Entity Relation Diagram (ERD)*. Diakses pada 25 Januari 2021, dari <https://www.visual-paradigm.com/guide/data-modeling/what-is-entity-relationship-diagram/>