

SKRIPSI

SCREENSAVER INFORMASI MAHASISWA WALI



Harry Senjaya Darmawan

NPM: 2017730067

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
«tahun»

UNDERGRADUATE THESIS

STUDENTS' INFORMATION SCREENSAVER



Harry Senjaya Darmawan

NPM: 2017730067

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY**

«tahun»

LEMBAR PENGESAHAN

SCREENSAVER INFORMASI MAHASISWA WALI

Harry Senjaya Darmawan

NPM: 2017730067

Bandung, «**tanggal**» «**bulan**» «**tahun**»

Menyetujui,

Pembimbing

Pascal Alfadian, Nugroho, M.Comp.

Ketua Tim Penguji

«**penguji 1**»

Anggota Tim Penguji

«**penguji 2**»

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

SCREENSAVER INFORMASI MAHASISWA WALI

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuahkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal «tanggal» «bulan» «tahun»



Harry Senjaya Darmawan
NPM: 2017730067

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»

Kata-kata kunci: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Indonesia»

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»

Keywords: «Tuliskan di sini kata-kata kunci yang anda gunakan, dalam bahasa Inggris»

«kepada siapa anda mempersembahkan skripsi ini. . . ?»

KATA PENGANTAR

«Tuliskan kata pengantar dari anda di sini . . . »

Bandung, «bulan» «tahun»

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xix
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	1
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	2
1.6 Sistematika Pembahasan	2
2 LANDASAN TEORI	3
2.1 Jsoup	3
2.1.1 Jsoup	3
2.1.2 Connection	3
2.1.3 Response	4
2.1.4 Elements	4
2.1.5 Element	5
2.2 JavaFX dan FXML	5
2.2.1 JavaFX	5
2.2.2 FXML	6
2.3 SIAModels	7
2.3.1 Mahasiswa	7
2.3.2 Nilai	10
2.3.3 ChronologicalComparator	10
2.3.4 MataKuliah	11
2.3.5 JenisKelamin	11
2.3.6 Status	11
2.3.7 TahunSemester	12
3 ANALISIS	13
3.1 Analisis Portal Akademik Mahasiswa	13
3.1.1 <i>Login</i>	13
3.1.2 Halaman Utama	15
3.1.3 Profil	15
3.1.4 Jadwal	16
3.1.5 Pembayaran	21
3.1.6 FRS/PRS	21
3.1.7 Nilai	22
3.1.8 Angket	24

3.1.9	Saran & Komentar	25
3.1.10	Kelulusan	25
3.1.11	Pengajuan	26
3.2	Analisis SIAKAD	26
3.2.1	Login	26
3.2.2	Mahasiswa	26
3.2.3	Pra Kuliah	32
3.2.4	Perkuliahinan	32
3.2.5	Ujian	32
3.2.6	Nilai	32
3.2.7	Evaluasi	32
3.2.8	Skripsi	32
3.2.9	Sidang	32
3.2.10	Kelulusan	33
3.2.11	Pengumuman	33
3.3	Data yang Dibutuhkan untuk <i>Screensaver</i>	33
3.3.1	Portal Akademik Mahasiswa	33
3.3.2	SIAKAD	35
3.4	Analisis Sistem <i>Screensaver</i>	36
3.4.1	<i>Use Case Screensaver</i>	36
3.4.2	Perancangan Kelas <i>Screensaver</i>	38
4	PERANCANGAN	39
4.1	Perancangan Kelas	39
4.2	Perancangan Antarmuka	48
5	IMPLEMENTASI DAN PENGUJIAN	51
5.1	Implementasi	51
5.1.1	Lingkungan Implementasi	51
5.1.2	Hasil Implementasi	51
5.2	Pengujian	52
5.2.1	Pengujian Fungsional	52
5.2.2	Pengujian Eksperimental	53
6	KESIMPULAN DAN SARAN	57
6.1	Kesimpulan	57
6.2	Saran	57
DAFTAR REFERENSI		59
A	KODE PROGRAM <i>Screensaver</i>	61
B	KODE PROGRAM PORTAL AKADEMIK MAHASISWA (STUDENTPORTAL)	65
C	KODE PROGRAM SIAKAD	71
D	PERBEDAAN KODE DOSEN DENGAN MAHASISWA	77
E	KODE PROGRAM FXML	81

DAFTAR GAMBAR

3.1 Halaman <i>Login 1</i>	14
3.2 Halaman <i>Login 2</i>	14
3.3 Halaman Utama Portal Akademik Mahasiswa	15
3.4 Halaman Profil	16
3.5 Halaman Kehadiran	16
3.6 Halaman Ketidakhadiran	17
3.7 Halaman Jadwal Kuliah Dalam Tabel Waktu	17
3.8 Halaman Jadwal Kuliah Tabel	18
3.9 Halaman UTS	18
3.10 Halaman UAS	19
3.11 Halaman MKU	19
3.12 Halaman Kalender Akademik	20
3.13 Halaman Daftar Kehadiran	20
3.14 Halaman Pembayaran	21
3.15 Tampilan FRS/PRS	21
3.16 Halaman Nilai Per Semester	22
3.17 Halaman Daftar Perkembangan Studi (1)	22
3.18 Halaman Daftar Perkembangan Studi (2)	23
3.19 Halaman Riwayat Indeks Prestasi	23
3.20 Halaman Nilai TOEFL	24
3.21 Halaman Angket	24
3.22 Halaman Saran & Komentar	25
3.23 Halaman Kelulusan	25
3.24 Halaman Pengajuan	26
3.25 Tangkapan Layar Halaman Daftar Mahasiswa Wali	27
3.26 Tangkapan Layar Pop-up Detail Mahasiswa	27
3.27 Tangkapan Layar Tab Identitas Mahasiswa	28
3.28 Tangkapan Layar Tab Riwayat Pendidikan	28
3.29 Tangkapan Layar Tab Orang Tua/Wali	29
3.30 Tangkapan Layar Tab Ringkasan Akademik	29
3.31 Tangkapan Layar Tab Nilai Semester Ini	30
3.32 Tangkapan Layar Tab Nilai Per Tahun Semester	30
3.33 Tangkapan Layar Tab Nilai Berdasarkan Kurikulum	31
3.34 Tangkapan Layar Tab Status Akademik	31
3.35 Tangkapan Layar PDF Daftar Perkembangan Studi	32
3.36 Diagram <i>Use Case Screensaver</i>	36
3.37 Perancangan Kelas <i>Screensaver</i>	38
4.1 Diagram Kelas Keseluruhan	39
4.2 Diagram Kelas SIAKAD	40
4.3 Diagram Kelas Studentportal	41
4.4 <i>Script</i> Data Nilai Mahasiswa Pada Halaman Nilai	46

4.5 Rancangan Antarmuka <i>Screensaver</i>	49
5.1 Tampilan <i>Screensaver</i> Mahasiswa Pertama	52
5.2 Tampilan <i>Screensaver</i> Mahasiswa Kedua	52
5.3 Tampilan <i>Screensaver</i> Mahasiswa Pertama	54
5.4 Tampilan <i>Screensaver</i> Mahasiswa Kedua	55
5.5 Tampilan <i>Screensaver</i> Mahasiswa Ketiga	55
5.6 Tampilan <i>Screensaver</i> Mahasiswa Keempat	56
5.7 Tampilan <i>Screensaver</i> Mahasiswa Kelima	56

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Setiap dosen wali (dikenal juga dengan istilah penasehat akademik ¹) memiliki data mengenai mahasiswa walinya yang dapat diakses melalui SIAKAD [1]. Namun, walaupun dosen wali memiliki data mengenai mahasiswa walinya, dosen wali juga perlu melakukan pemeriksaan data mahasiswa walinya, terutama data akademiknya secara berkala. Dengan berbagai kesibukan yang dialami oleh para dosen wali dan mahasiswa, ditambah dengan situasi Indonesia saat ini yang menyebabkan perkuliahan dilakukan secara daring, akan sangat sulit bagi dosen wali untuk menemui mahasiswa walinya. Hal ini menyebabkan dosen wali kesulitan mengamati perkembangan mahasiswa walinya. Selain itu dalam mencari data akademik mahasiswa wali, SIAKAD juga memiliki kekurangan dimana dosen wali perlu melakukan *login*, kemudian mencari serta memilih mahasiswa wali yang ingin dilihat. Sehingga dapat dikatakan proses tersebut tidaklah efisien.

Maka dari itu, pada skripsi ini akan dibuat sebuah perangkat lunak yang berupa *screensaver* yang dapat menampilkan data akademik mahasiswa wali secara acak. *Screensaver* adalah program komputer yang mengosongkan layar atau mengisinya dengan gambar atau pola bergerak ketika komputer telah diam dalam waktu yang lama [2]. Dengan menggunakan perangkat lunak tersebut, dosen wali dapat tetap mengamati perkembangan mahasiswa walinya, paling tidak secara akademik.

Dikarenakan terbimbing tidak memiliki akses ke SIAKAD untuk mengakses data mahasiswa wali, namun terbimbing memiliki akses ke Portal Akademik Mahasiswa [3] maka, terbimbing mensimulasikan dengan Portal Akademik Mahasiswa, dan kemudian Pembimbing mengubah aksesnya ke SIAKAD. Struktur kelas yang akan digunakan dalam pembuatan perangkat lunak ini yaitu, struktur kelas SIAModels yang tersedia pada Github dan Maven Public Repository [4]. Simulasi Portal Akademik Mahasiswa ini berdasarkan pada skripsi yang dibuat oleh Andrianto Sugiarto [5]. Tetapi terdapat beberapa perbaikan yang perlu dilakukan, dan dapat dilihat pada sub-bab 3.1.

Perangkat lunak akan dibangun menggunakan bahasa pemrograman Java. Terdapat beberapa teknologi yang dapat dimanfaatkan dalam bahasa pemrograman Java. Teknologi yang pertama yaitu *library* jsoup. Jsoup dapat digunakan untuk melakukan *scraping*, sehingga pengambilan data mahasiswa tidak memerlukan API (*Application Programming Interface*) [6]. Teknologi lainnya yang dapat dimanfaatkan yaitu JavaFX. JavaFX dapat digunakan untuk mengonversi aplikasi tersebut menjadi *screensaver*.

1.2 Rumusan Masalah

Rumusan masalah yang akan dibahas pada skripsi ini adalah sebagai berikut:

- Bagaimana cara memanfaatkan jsoup untuk mengambil data mahasiswa?
- Bagaimana cara memanfaatkan JavaFX untuk mengonversi aplikasi tersebut menjadi *screensaver*?

¹Sebagaimana dituliskan pada <https://unpar.ac.id/akademik/>. Pada skripsi ini istilah yang digunakan adalah dosen wali karena istilah tersebut yang muncul pada aplikasi SIAKAD [1]

1.3 Tujuan

Tujuan yang ingin dicapai dari penulisan skripsi ini sebagai berikut:

- Memanfaatkan jsoup untuk mengambil data mahasiswa.
- Memanfaatkan JavaFX untuk mengonversi aplikasi tersebut menjadi *screensaver*.

1.4 Batasan Masalah

Dikarenakan terbimbing tidak memiliki akses ke SIAKAD untuk mengakses data mahasiswa wali, namun terbimbing memiliki akses ke Portal Akademik Mahasiswa maka, terbimbing mensimulasikan dengan Portal Akademik Mahasiswa, dan kemudian Pembimbing mengubah aksesnya ke SIAKAD.

1.5 Metodologi

Langkah-langkah yang akan dilakukan dalam melakukan penelitian ini yaitu:

1. Melakukan studi mengenai jsoup.
2. Melakukan studi mengenai cara mengonversi aplikasi menjadi *screensaver*.
3. Mempelajari struktur kelas SIAModels.
4. Menganalisis IF Portal Akademik Mahasiswa dan Portal Akademik Mahasiswa.
5. Merancang struktur kelas aplikasi.
6. Melakukan studi mengenai cara mendesain antarmuka aplikasi
7. Mendesain antarmuka aplikasi.
8. Mengimplementasikan jsoup untuk mengambil data mahasiswa.
9. Mengonversi aplikasi menjadi *screensaver* dengan menggunakan JavaFX.
10. Melakukan pengujian dan eksperimen.
11. Menulis dokumen skripsi.

1.6 Sistematika Pembahasan

Dokumen dibagi ke dalam beberapa bab dengan sistematika pembahasan sebagai berikut:

- Bab 1. Pendahuluan, membahas tentang latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, metode penelitian dan sistematika pembahasan mengenai skripsi.
- Bab 2. Landasan Teori, membahas landasan dari teori-teori yang berhubungan serta mendukung penelitian, meliputi jsoup, JavaFX, dan SIAModels.
- Bab 3. Analisis, menjelaskan tentang analisis Portal Akademik Mahasiswa, analisis SIAKAD, analisis data yang dibutuhkan untuk *screensaver*, serta analisis sistem *screensaver*.
- Bab 4. Perancangan, membahas perancangan kelas beserta deskripsi kelas dan fungsinya, serta perancangan antarmuka.
- Bab 5. Implementasi dan pengujian, membahas hasil-hasil implementasi dan pengujian secara fungsional dan eksperimental.
- Bab 6. Kesimpulan dan saran, membahas kesimpulan yang diperoleh dari penelitian ini dan saran untuk pengembangan berikutnya.

BAB 2

LANDASAN TEORI

Bab Landasan Teori ini berisi teori-teori yang menjadi dasar penelitian ini, meliputi jsoup, JavaFX, dan SIAModels.

2.1 Jsoup

Salah satu teknologi yang dapat dimanfaatkan untuk melakukan *scraping* yaitu *library* Java jsoup. Jsoup adalah *library* Java untuk mengerjakan dokumen HTML yang menyediakan API yang baik untuk mengekstraksi, memanipulasi data, dan menyelesaikan pembersihan data awal menggunakan metode terbaik dari *Document Object Model* (DOM), *Cascading Style Sheets* (CSS), dan metode lain yang mirip dengan jQuery. Jsoup mengimplementasikan spesifikasi WHATWG HTML5, dan mem-parsing HTML ke DOM yang sama seperti yang dilakukan *browser* modern. Pada skripsi ini akan digunakan jsoup versi 1.13.1. Berikut adalah layanan utama yang tersedia di jsoup [6]:

1. *Scrape* dan *parse* HTML dari URL, *file*, atau string.
2. Mencari dan ekstrak data menggunakan traversal DOM dan CSS *selector*.
3. Memanipulasi elemen HTML, atribut HTML, dan teks.
4. Membersihkan konten yang dikirim oleh pengguna yang menggunakan *safe white-lists* untuk mencegah serangan XSS.
5. Menghasilkan HTML yang rapi.

Subbab-subbab berikut menjelaskan beberapa kelas dari jsoup.

2.1.1 Jsoup

Kelas ini merupakan inti untuk mengakses fungsionalitas jsoup. Salah satu *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public static Connection connect(String url)`

Berfungsi untuk membuat koneksi baru ke URL. Digunakan untuk mengambil dan mengurai halaman HTML.

Parameter: URL situs web dengan protokol HTTP atau HTTPS.

Kembalian: koneksi dengan situs web.

2.1.2 Connection

Kelas ini merupakan *interface* yang menyediakan antarmuka yang nyaman untuk mengambil konten dari web, dan menguraikannya menjadi dokumen. Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `Connection cookies(Map<String, String> cookies)`

Berfungsi untuk menambahkan *cookies* ke *request*.

Parameter:

– *cookies*: Map dari *cookie*.

Kembalian: koneksi yang sama tetapi sudah diubah.

- `Connection data(String key, String value)`

Berfungsi untuk menambahkan parameter data *request* yang bisa dikirim melalui *method* HTTP GET atau POST.

Parameter:

- `key`: kunci data.
- `value`: nilai data.

Kembalian: koneksi yang sama tetapi sudah diubah.

- `Connection method(URLConnection.Method method)`

Berfungsi untuk mengatur *method request* yang akan digunakan, HTTP GET atau POST. *Default*-nya adalah GET.

Parameter:

- `method`: *method request* HTTP.

Kembalian: koneksi yang sama tetapi sudah diubah.

- `Connection timeout(int millis)`

Berfungsi untuk mengatur batas waktu *request*. Batas waktu *default* adalah 30 detik. Batas waktu nol akan dianggap sebagai batas waktu yang tak terhingga.

Parameter:

- `millis`: batas waktu dalam milidetik.

Kembalian: koneksi yang sama tetapi sudah diubah.

- `Connection.Response execute()`

Berfungsi untuk mengirim *request* HTTP.

Kembalian: objek Response.

2.1.3 Response

Kelas ini merepresentasikan *response* HTTP. Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `Map<String, String> cookies()`

Berfungsi untuk mendapatkan seluruh *cookies*.

Kembalian: seluruh *cookies*.

- `Document parse()`

Berfungsi untuk mengurai *body* jawaban menjadi dokumen.

Kembalian: dokumen yang diurai.

- `String body()`

Berfungsi untuk mendapatkan *body* jawaban dalam bentuk *string*.

Kembalian: *body* jawaban dalam bentuk *string*.

2.1.4 Elements

Kelas ini merepresentasikan kumpulan elemen HTML. Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public Elements select(String query)`

Berfungsi untuk menemukan elemen-elemen yang sesuai dalam *list* elemen.

Parameter:

- `query`: kueri CSS berupa CSS Selector.

Kembalian: elemen-elemen yang sudah diseleksi sesuai kueri.

- `public String val()`

Berfungsi untuk mendapatkan nilai dari elemen pertama.

Kembalian: nilai elemen.

- `public String text()`

Berfungsi untuk mendapatkan kombinasi teks dari seluruh elemen yang sesuai.

Kembalian: seluruh teks dalam *string*.

2.1.5 Element

Kelas ini merepresentasikan sebuah elemen HTML yang berisikan *tag*, atribut, dan anak elemen. Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public Elements select(String cssQuery)`

Berfungsi untuk menemukan elemen yang cocok dengan kueri CSS Selector, dengan elemen ini sebagai konteks awal.

Parameter:

- `cssQuery`: kueri CSS berupa CSS Selector.

Kembalian: elemen-elemen HTML yang sesuai dengan kueri CSS.

- `public Element child(int index)`

Berfungsi untuk mendapatkan anak elemen berdasarkan nomor indeks.

Parameter:

- `index`: nomor index.

Kembalian: anak elemen.

- `public Element children()`

Berfungsi untuk mendapatkan seluruh anak elemen.

Kembalian: seluruh anak elemen.

- `public String className()`

Berfungsi untuk mendapatkan nama kelas elemen.

Kembalian: nama kelas elemen.

- `public String text()`

Berfungsi untuk mendapatkan teks gabungan dari elemen.

Kembalian: teks dalam *string*.

2.2 JavaFX dan FXML

2.2.1 JavaFX

JavaFX adalah platform aplikasi klien *open source* generasi berikutnya untuk *desktop*, *mobile*, dan *embedded systems* yang dibangun dengan Java. JavaFX memungkinkan untuk membuat aplikasi Java dengan antarmuka pengguna (UI) modern dengan akselerasi perangkat keras yang sangat portabel. [7]. Subbab-subbab berikut menjelaskan beberapa kelas dari JavaFX. [7]

Application

Titik masuk untuk aplikasi JavaFX adalah kelas `Application`. JavaFX melakukan hal berikut, secara berurutan, setiap kali aplikasi diluncurkan:

1. Membuat *instance* kelas `Application` yang ditentukan
2. Memanggil *method* `init()`
3. Memanggil *method* `start(javafx.stage.Stage)`
4. Menunggu aplikasi selesai, yang terjadi jika salah satu dari hal berikut terjadi:
 - aplikasi memanggil `Platform.exit()`
 - *window* terakhir telah ditutup dan atribut `implicitExit` di `Platform` adalah true
5. Memanggil *method* `stop()`

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public void init()`

Method inisialisasi aplikasi. *Method* ini dipanggil segera setelah kelas `Application` dimuat dan dibangun. *Method* ini dapat ditimpak untuk melakukan inisialisasi sebelum aplikasi sebenarnya dimulai.

- `public abstract void start(Stage primaryStage)`

Titik masuk utama untuk semua aplikasi JavaFX. *Method* `start` dipanggil setelah *method*

`init` kembali, dan setelah sistem siap untuk aplikasi mulai berjalan.

Parameter:

- `primaryStage`: *stage* utama untuk aplikasi ini, tempat *scene* aplikasi dapat diatur.
- `public static void launch()`
Meluncurkan aplikasi. *Method* ini biasanya dipanggil dari `main()` *method*. Tidak boleh dipanggil lebih dari sekali atau *exception* akan dilemparkan. Harus merupakan *subclass* dari `Application` atau `RuntimeException` akan dilemparkan.

Stage

Kelas `Stage` adalah *container* JavaFX tingkat atas. `Stage` utama dibangun oleh platform. Objek `Stage` harus dibuat dan dimodifikasi pada JavaFX Application Thread.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public final void setScene(Scene value)`
Menentukan *scene* yang akan digunakan di *stage* ini.
- Parameter:**
 - `value`: *scene* yang akan digunakan.
- `public final void show()`
Mencoba menampilkan *window* ini dengan mengubah *visibility* menjadi true.

Scene

Kelas `Scene` adalah wadah untuk semua konten dalam grafik *scene*.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public Scene(Parent root, double width, double height)`
Merupakan *constructor* dari kelas `Scene`.
- Parameter:**
 - `root`: Node root dari grafik *scene*.
 - `width`: Lebar *scene*.
 - `height`: Tinggi *scene*.

FXMLLoader

Memuat hierarki objek dari dokumen XML.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public FXMLLoader(URL location)`
Merupakan *constructor* dari kelas `FXMLLoader`.
- Parameter:**
 - `location`: lokasi dokumen fxml.
- `public <T> T load`
Memuat hierarki objek dari dokumen FXML.

2.2.2 FXML

FXML adalah bahasa *markup* berbasis XML yang dapat dituliskan untuk membangun grafik objek Java. FXML memberikan alternatif yang nyaman untuk membuat grafik dalam kode prosedural, dan cocok untuk mendefinisikan antarmuka pengguna aplikasi JavaFX, karena struktur hierarki dari dokumen XML sangat mirip dengan struktur grafik *scene* JavaFX. [7] Subbab-subbab berikut menjelaskan beberapa bagian dari FXML. [8]

Controller Method Event Handlers

Controller Method Event Handlers adalah *method* yang ditentukan oleh "pengontrol" dokumen. *Controller* adalah objek yang dikaitkan dengan konten *deserialized* dari dokumen FXML dan bertanggung jawab untuk mengoordinasikan perilaku objek (seringkali elemen antarmuka pengguna) yang ditentukan oleh dokumen. Sebuah *event handler metode controller* ditentukan oleh simbol *hash* terkemuka diikuti dengan nama *method*.

Variable Resolution

Dokumen FXML mendefinisikan *namespace* variabel dimana elemen bernama dan variabel dapat diidentifikasi secara unik. Menetapkan nilai fx:id ke elemen akan membuat variabel dokumen yang nantinya dapat dirujuk oleh atribut dereferensi variabel. Selain itu, jika tipe objek mendefinisikan properti *id*, nilai ini juga akan diteruskan ke *method setId()* milik objek.

2.3 SIAModels

SIAModels merupakan kelas-kelas dalam bahasa Java yang merepresentasikan objek-objek yang tersedia di Sistem Informasi Akademik UNPAR [4]. Pada skripsi ini akan digunakan SIAModels versi 5.1.1.

2.3.1 Mahasiswa

Kelas ini merepresentasikan mahasiswa. Atribut yang dimiliki kelas ini antara lain:

- `String npm` : Nomor Pokok Mahasiswa (NPM).
- `String nama` : nama mahasiswa.
- `List<Nilai> riwayatNilai` : riwayat nilai yang dimiliki mahasiswa.
- `String photoPath` : URL dari foto mahasiswa.
- `List<JadwalKuliah> jadwalKuliahList` : daftar jadwal kuliah mahasiswa
- `SortedMap<LocalDate, Integer> nilaiTOEFL` : nilai TOEFL mahasiswa.
- `Status status` : status mahasiswa.
- `LocalDate tanggalLahir` : tanggal lahir mahasiswa.
- `JenisKelamin jenisKelamin` : jenis kelamin mahasiswa.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public Mahasiswa(String npm)`
Merupakan *constructor* dari kelas `Mahasiswa`.

Parameter:

– `npm`: nomor pokok mahasiswa.

- `public String getNama()`
Berfungsi untuk mendapatkan nama mahasiswa.

Kembalian: nama mahasiswa.

- `public void setNama(String nama)`
Berfungsi untuk mengubah nama mahasiswa.

Parameter:

– `nama`: nama mahasiswa.

- `public String getNpm()`
Berfungsi untuk mendapatkan nomor pokok mahasiswa.

Kembalian: nomor pokok mahasiswa.

- `public String getPhotoPath()`
Berfungsi untuk mendapatkan URL dari foto mahasiswa.

Kembalian: URL dari foto mahasiswa.

- `public void setPhotoPath(String photoPath)`
Berfungsi untuk mengubah URL dari foto mahasiswa.
Parameter:
 - `photoPath`: URL dari foto mahasiswa.
- `public List<JadwalKuliah> getJadwalKuliahList`
Berfungsi untuk mendapatkan jadwal kuliah mahasiswa.
Kembalian: jadwal kuliah mahasiswa dalam *list*.
- `public void setJadwalKuliahList(List<JadwalKuliah> jadwalKuliahList)`
Berfungsi untuk mengubah jadwal kuliah mahasiswa.
Parameter:
 - `jadwalKuliahList`: jadwal kuliah mahasiswa dalam *list*.
- `public String getEmailAddress()`
Berfungsi untuk mendapatkan *email* mahasiswa.
Kembalian: *email* mahasiswa.
- `public List<Nilai> getRiwayatNilai()`
Berfungsi untuk mendapatkan riwayat nilai mahasiswa.
Kembalian: riwayat nilai mahasiswa dalam List.
- `public SortedMap<LocalDate, Integer> getNilaiTOEFL()`
Berfungsi untuk mendapatkan nilai TOEFL mahasiswa.
Kembalian: nilai TOEFL mahasiswa dalam SortedMap.
- `public void setNilaiTOEFL(SortedMap<LocalDate, Integer> nilaiTOEFL)`
Berfungsi untuk mengubah nilai TOEFL mahasiswa.
Parameter:
 - `nilaiTOEFL`: nilai TOEFL mahasiswa dalam SortedMap.
- `public Status getStatus()`
Berfungsi untuk mendapatkan status mahasiswa.
Kembalian: status mahasiswa.
- `public void setStatus(Status status)`
Berfungsi untuk mengubah status mahasiswa.
Parameter:
 - `status`: status mahasiswa.
- `public LocalDate getTanggalLahir()`
Berfungsi untuk mendapatkan tanggal lahir mahasiswa.
Kembalian: tanggal lahir mahasiswa.
- `public void setTanggalLahir(LocalDate tanggalLahir)`
Berfungsi untuk mengubah tanggal lahir mahasiswa.
Parameter:
 - `tanggalLahir`: tanggal lahir mahasiswa.
- `public JenisKelamin getJenisKelamin()`
Berfungsi untuk mendapatkan jenis kelamin mahasiswa.
Kembalian: jenis kelamin mahasiswa.
- `public void setJenisKelamin(JenisKelamin jenisKelamin)`
Berfungsi untuk mengubah jenis kelamin mahasiswa.
Parameter:
 - `jenisKelamin`: jenis kelamin mahasiswa.
- `public byte[] getPhotoImage()`
Berfungsi untuk mendapatkan foto mahasiswa yang dibungkus dalam kelas `java.awt.Image`.
Berbeda dengan *method* `getPhotoPath()`, *method* ini akan menghasilkan image, apapun bentuk photo path nya (bisa berupa URL ataupun base64 string).
Kembalian: foto mahasiswa.
- `public double calculateIPKLulus()`

Berfungsi untuk menghitung IPK mahasiswa sampai saat ini, dengan aturan kuliah yang tidak lulus tidak dihitung dan jika pengambilan beberapa kali, diambil nilai terbaik. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Kembalian: IPK lulus.

- `public double calculateIPTempuh(boolean lulusSaja)`

Berfungsi untuk menghitung IP mahasiswa sampai saat ini, dengan aturan perhitungan kuliah yang tidak lulus ditentukan parameter, dan jika pengambilan beberapa kali, diambil nilai terbaik.

Parameter:

- `lulusSaja`: `lulusSaja` set `true` jika ingin membuang mata kuliah tidak lulus, `false` jika ingin semua (sama dengan "IP N. Terbaik" di DPS). Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Kembalian: IPK lulus.

- `public double calculateIPKumulatif()`

Berfungsi untuk menghitung IP Kumulatif mahasiswa sampai saat ini, dengan aturan jika pengambilan beberapa kali, diambil semua. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Kembalian: IPK lulus.

- `public double calculateIPS()`

Berfungsi untuk menghitung IPS semester terakhir sampai saat ini, dengan aturan kuliah yang tidak lulus dihitung. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Kembalian: nilai IPS sampai saat ini.

- `public int calculateSKSLulus()`

Berfungsi untuk menghitung jumlah SKS lulus mahasiswa saat ini. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Kembalian: SKS lulus.

- `public int calculateSKSTempuh(boolean lulusSaja)`

Berfungsi untuk menghitung jumlah SKS tempuh mahasiswa saat ini. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Parameter:

- `lulusSaja`: `lulusSaja` set `true` jika ingin membuang SKS tidak lulus

Kembalian: SKS tempuh

- `public Set<TahunSemester> calculateTahunSemesterAktif()`

Berfungsi untuk mendapatkan seluruh tahun semester di mana mahasiswa ini tercatat sebagai mahasiswa aktif, dengan strategi memeriksa riwayat nilainya. Jika ada satu nilai saja pada sebuah tahun semester, maka dianggap aktif pada semester tersebut.

Kembalian: kumpulan tahun semester di mana mahasiswa ini aktif.

- `public boolean hasLulusKuliah(String kodeMataKuliah)`

Berfungsi untuk memeriksa apakah mahasiswa ini sudah lulus mata kuliah tertentu. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Parameter:

- `kodeMataKuliah`: kode mata kuliah yang ingin diperiksa kelulusannya.

Kembalian: `true` jika sudah pernah mengambil dan lulus, `false` jika belum.

- `public boolean hasTempuhKuliah(String kodeMataKuliah)`

Memeriksa apakah mahasiswa ini sudah pernah menempuh mata kuliah tertentu. Sebelum memanggil *method* ini, *getRiwayatNilai()* harus sudah mengandung nilai per mata kuliah.

Parameter:

- `kodeMataKuliah`: kode mata kuliah yang ingin diperiksa kelulusannya.

Kembalian: `true` jika sudah pernah mengambil, `false` jika belum.

- `public int getTahunAngkatan()`

Mendapatkan tahun angkatan mahasiswa ini berdasarkan NPM-nya.

Kembalian: tahun angkatan.

2.3.2 Nilai

Kelas ini merepresentasikan nilai yang ada pada riwayat nilai mahasiswa. Atribut yang dimiliki kelas ini antara lain:

- **TahunSemester** `tahunSemester`: tahun dan semester kuliah ini diambil
- **MataKuliah** `mataKuliah`: mata kuliah yang diambil.
- **Character kelas**: kelas kuliah.
- **Map<String, Double> nilaiTugas**: nilai Angka Rata-rata Tugas (ART).
- **Double nilaiUTS**: nilai Ujian Tengah Semester (UTS).
- **Double nilaiUAS**: nilai Ujian Akhir Semester (UAS).
- **String nilaiAkhir**: nilai akhir.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public Nilai(TahunSemester tahunSemester, MataKuliah mataKuliah, Character kelas, Map<String, Double> nilaiTugas, Double nilaiUTS, Double nilaiUAS, String nilaiAkhir)`

Merupakan *constructor* dari kelas `Nilai`.

Parameter:

- `tahunSemester`: tahun dan semester kuliah ini diambil.
- `mataKuliah`: mata kuliah yang diambil.
- `kelas`: kelas kuliah.
- `nilaiTugas`: nilai ART.
- `nilaiUTS`: nilai UTS.
- `nilaiUAS`: nilai UAS.
- `nilaiAkhir`: nilai akhir.

- `public MataKuliah getMataKuliah()`

Mendapatkan mata kuliah yang diambil.

Kembalian: mata kuliah.

- `public String getNilaiAkhir()`

Mengembalikan nilai akhir dalam bentuk huruf (A, B, C, D, ..., atau K).

Kembalian: nilai akhir dalam huruf, atau `null` jika tidak ada.

- `public Double getAngkaAkhir()`

Mendapatkan nilai akhir dalam bentuk angka.

Kembalian: nilai akhir dalam angka, atau `null` jika `getNilaiAkhir()` mengembalikan `null`.

- `public int getTahunAjaran()`

Mendapatkan tahun ajaran saat pengambilan mata kuliah.

Kembalian: tahun ajaran saat pengambilan mata kuliah.

- `public TahunSemester getTahunSemester()`

Mendapatkan tahun dan semester pengambilan mata kuliah.

Kembalian: tahun dan semester pengambilan mata kuliah.

- `public Semester getSemester()`

Mendapatkan semester pengambilan mata kuliah.

Kembalian: semester pengambilan mata kuliah

2.3.3 ChronologicalComparator

Pembandingan antara satu nilai dengan nilai lainnya, secara kronologis waktu pengambilan. *Method* yang dimiliki kelas ini adalah sebagai berikut:

- `public int compare(Nilai o1, Nilai o2)`

Berfungsi untuk membandingkan nilai.

Parameter:

- `o1`: nilai pertama yang akan dibandingkan.
- `o2`: nilai kedua yang akan dibandingkan.

Kembalian: hasil perbandingan.

2.3.4 MataKuliah

Kelas ini merepresentasikan sebuah mata kuliah. Atribut yang dimiliki kelas ini antara lain:

- `String kode`: kode mata kuliah
- `String nama`: nama mata kuliah
- `Integer sks`: sks mata kuliah.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- `public MataKuliah(String kode, String nama, int sks)`
Merupakan *constructor* dari kelas MataKuliah.

Parameter:

- `kode`: kode mata kuliah.
- `nama`: nama mata kuliah.
- `sks`: sks mata kuliah.

- `public String getKode()`
Mendapatkan kode mata kuliah.

Kembalian: kode mata kuliah.

- `public int getSks()`
Mendapatkan sks mata kuliah.

Kembalian: sks mata kuliah.

- `public String getNama()`
Mendapatkan nama mata kuliah.

Kembalian: nama mata kuliah.

2.3.5 JenisKelamin

Kelas ini berupa enum yang merepresentasikan jenis kelamin mahasiswa. Nilai dari enum ini antara lain:

- `LAKI_LAKI("Laki-laki")`
- `PEREMPUAN("Perempuan")`

2.3.6 Status

Kelas ini berupa enum yang merepresentasikan status mahasiswa. Nilai dari enum ini antara lain:

- `SEMUA("00")`
- `AKTIF("01")`
- `GENCAT("02")`
- `CUTI_SEBELUM_FRS("03")`
- `CUTI_SETELAH_FRS("04")`
- `KELUAR("05")`
- `LULUS("06")`
- `DROP_OUT("07")`
- `SISIPAN("08")`

2.3.7 TahunSemester

Kelas ini menyimpan konstanta untuk semester beserta tahunnya di UNPAR. Atribut yang dimiliki kelas ini antara lain:

- **String kodeTahunSemester:** kode semester 3 digit, 2 digit pertama berupa tahun, digit terakhir menandakan semester dengan definisi 1 untuk ganjil, 2 untuk genap, 4 untuk pendek, dan 6 untuk transfer.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:

- **public TahunSemester(String kodeTahunSemester)**
Method ini merupakan constructor dari kelas TahunSemester.

Parameter:

- **kodeTahunSemester:** semester dalam bentuk teks (GANJIL, GENAP, PENDEK, TRANSFER, dan UNKNOWN5).

- **public TahunSemester(int tahun, Semester semester)**
Method ini merupakan constructor dari kelas TahunSemester.

Parameter:

- **tahun:** tahun ajaran.
 - **semester:** semester dari tahun ajaran.

- **public Semester getSemester()**

Method ini berfungsi untuk mendapatkan semester.

Kembalian: semester dalam teks.

- **public int getTahun()**

Method ini berfungsi untuk mendapatkan tahun.

Kembalian: tahun ajaran.

- **private static void validateKodeSemester(String kodeTahunSemester)**

Method ini berfungsi untuk melakukan validasi terhadap kode tahun semester.

Parameter:

- **kodeTahunSemester:** kode tahun semester.

BAB 3

ANALISIS

Pada bab ini akan dijelaskan mengenai analisis Portal Akademik Mahasiswa, analisis SIAKAD, analisis data yang dibutuhkan untuk *screensaver*, serta analisis sistem *screensaver*.

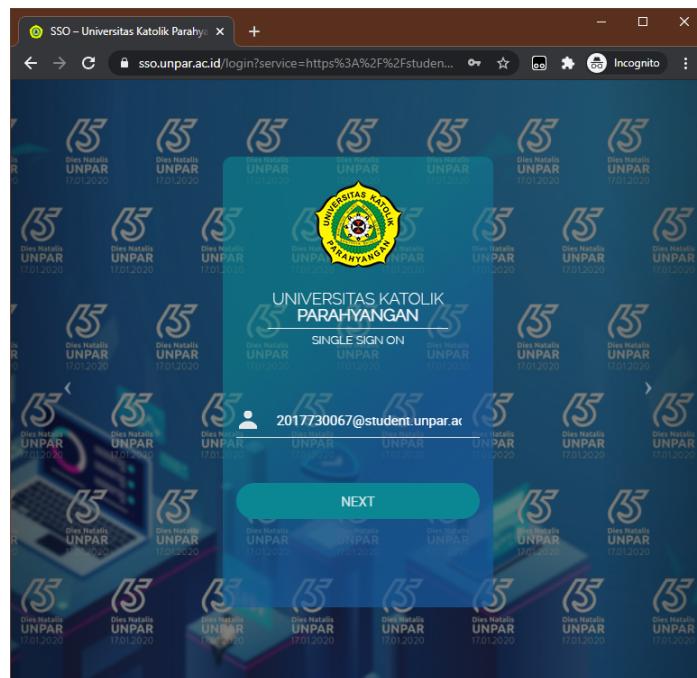
3.1 Analisis Portal Akademik Mahasiswa

Untuk mengambil data mahasiswa, diperlukan sumber data mahasiswa tersebut. Sumber data mahasiswa tersebut dapat diperoleh melalui Portal Akademik Mahasiswa. Portal Akademik Mahasiswa merupakan sebuah situs yang diperuntukkan bagi mahasiswa untuk mendapatkan informasi mengenai profil dan kegiatan akademik mahasiswa tersebut. Mahasiswa dapat mengakses Portal Akademik Mahasiswa melalui *URL* <https://studentportal.unpar.ac.id/>. Untuk mengakses Portal Akademik Mahasiswa, mahasiswa harus melakukan *login* menggunakan *email* dan *password* mahasiswa tersebut.

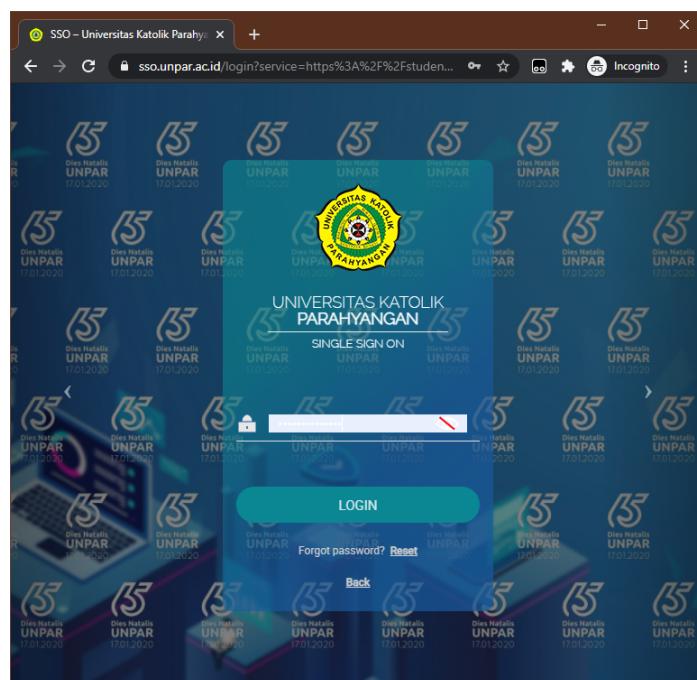
Aplikasi *screensaver* akan melakukan *http request* ke Portal Akademik Mahasiswa untuk mendapatkan data untuk setiap kebutuhan dari masing-masing fitur yang ada, dimana fitur-fitur yang terdapat dalam aplikasi Mahasiswa Wali *Screensaver* adalah informasi umum mengenai mahasiswa, serta prestasi akademik mahasiswa. Pengambilan data secara langsung dari Portal Akademik Mahasiswa dilakukan menggunakan *library jsoup*. Beberapa implementasi pemanfaatan jsoup untuk mengambil data-data tersebut sudah diimplementasikan pada skripsi Andrianto Sugiarto [5] sebelumnya. Data yang telah didapat dari Portal Akademik Mahasiswa kemudian diolah ke dalam SIAModels, dan ditampilkan sesuai dengan fitur-fitur yang ada pada aplikasi Mahasiswa Wali *Screensaver*.

3.1.1 *Login*

Halaman *Login* (Gambar 3.1 dan 3.2) merupakan halaman dimana mahasiswa memasukkan *email* dan *password* untuk mengakses menu-menu Portal Akademik Mahasiswa.



Gambar 3.1: Halaman Login 1



Gambar 3.2: Halaman Login 2

Login dilakukan dengan mengirim *request method* POST, dan kemudian mengambil session yang akan digunakan sebagai *cookies* apabila *login* berhasil. Terdapat beberapa perubahan yang terjadi pada situs Portal Akademik Mahasiswa semenjak skripsi Andrianto Sugiarto [5], yang mengakibatkan perlunya perubahan (Kode 3.1) terhadap implementasi jsoup:

1. Menghapus pemanggilan fungsi validateTLCertificates() dikarenakan sudah *deprecated* [6].
2. Menghapus pengambilan data dengan kueri css "*input [name=lt]*" dikarenakan kueri tersebut sudah dihapus oleh Portal Akademik Mahasiswa.
3. Menghapus *form data* dengan *key "submit"* yang memiliki *value ""*.

Kode 3.1: Perubahan Implementasi Jsoup Login

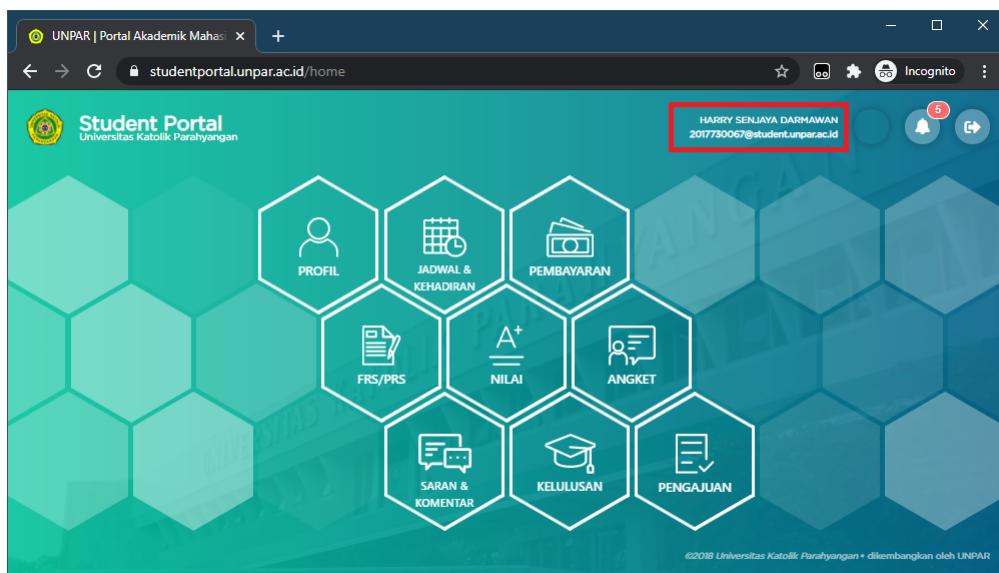
```

1 @@ -78,11 +77,9 @@ public class Scraper {
2     Connection conn = Jsoup.connect(LOGIN_URL);
3     conn.data("Submit", "Login");
4     conn.timeout(0);
5     - conn.validateTLCertificates(false);
6     conn.method(Connection.Method.POST);
7     Response resp = conn.execute();
8     Document doc = resp.parse();
9     - String lt = doc.select("input[name=lt]").val();
10    String execution = doc.select("input[name=execution]").val();
11    String jsessionid = resp.cookie("JSESSIONID");
12    /* SSO LOGIN */
13 @@ -90,12 +87,9 @@ public class Scraper {
14     loginConn.cookies(resp.cookies());
15     loginConn.data("username", user);
16     loginConn.data("password", pass);
17     - loginConn.data("lt", lt);
18     loginConn.data("execution", execution);
19     loginConn.data("_eventId", "submit");
20     - loginConn.data("submit", "");
21     loginConn.timeout(0);
22     - loginConn.validateTLCertificates(false);
23     loginConn.method(Connection.Method.POST);
24     resp = loginConn.execute();
25     if (resp.body().contains(user)) {

```

3.1.2 Halaman Utama

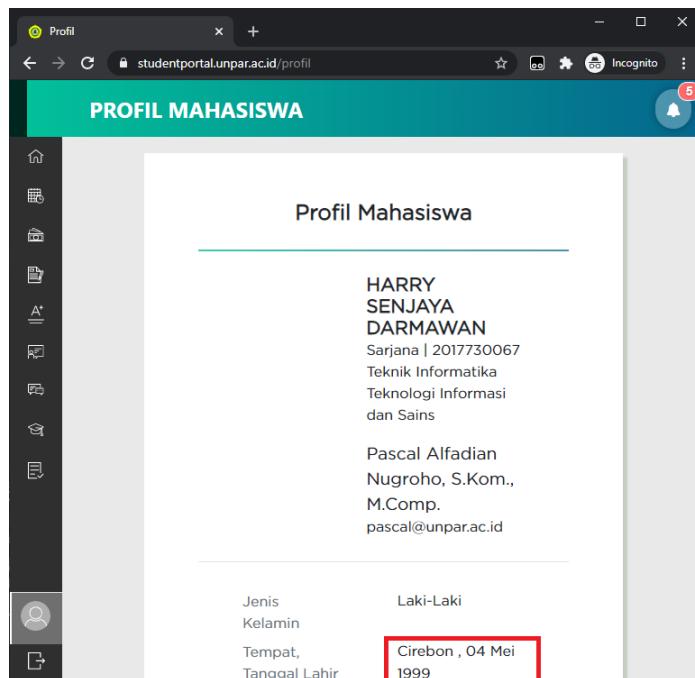
Pada Halaman Utama Portal Akademik Mahasiswa (Gambar 3.3) terdapat beberapa menu yang dapat digunakan sebagai sumber data.



Gambar 3.3: Halaman Utama Portal Akademik Mahasiswa

3.1.3 Profil

Menu Profil merupakan halaman yang menampilkan data diri mahasiswa (Gambar 3.4).



Gambar 3.4: Halaman Profil

3.1.4 Jadwal

Menu Jadwal terdiri dari beberapa submenu:

- Kehadiran

Submenu ini berfungsi untuk menandakan kehadiran mahasiswa di suatu mata kuliah pada hari dimana mahasiswa tersebut mengakses halaman tersebut (Gambar 3.5).

No	Kode Mata Kuliah	Nama Mata Kuliah	Pertemuan-ke	Kelas	Waktu Perkuliahan	Status Kehadiran	Waktu Hadir	Link Pembelajaran	Presensi	Validasi Perkuliahan
1.	AIF184002	Skripsi 2	3	A	14:00 - 15:00 batas kehadiran : 15 menit batas keterlambatan : 30 menit	Hadir	14:00	https://zoom.us/j/93.....		

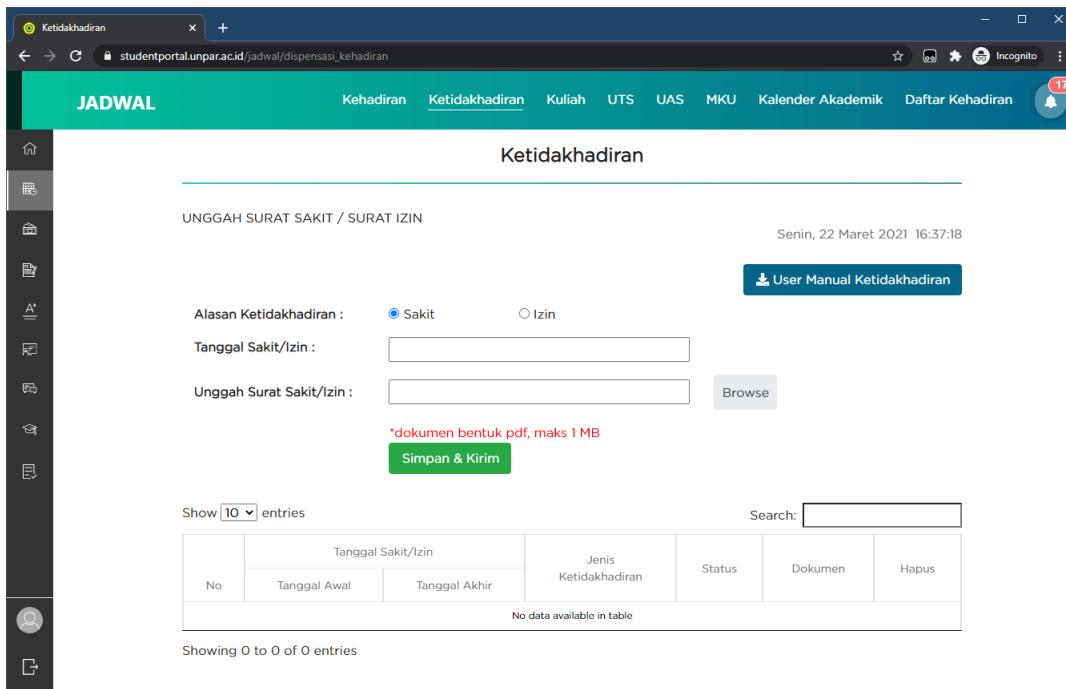
Keterangan baris merah :

- Kegiatan perkuliahan ditunda pada hari libur nasional yang ditetapkan oleh pemerintah dan UNPAR.
- Bertepatan dengan jadwal Ujian Tengah Semester Ganjil 2020/2021 tanggal 9 - 20 November 2020.

Gambar 3.5: Halaman Kehadiran

- Ketidakhadiran

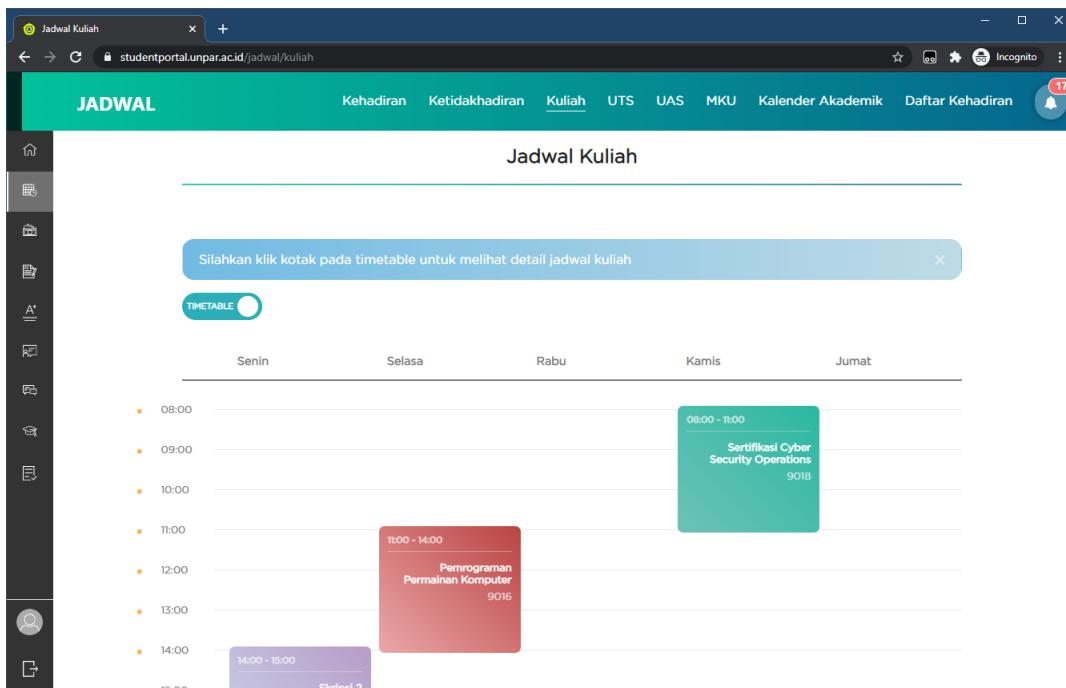
Submenu ini berfungsi untuk mengunggah surat sakit atau surat izin mahasiswa. (Gambar 3.6).



Gambar 3.6: Halaman Ketidakhadiran

- Kuliah

Submenu ini berisi tentang jadwal kuliah yang dapat disusun per semester dan terdapat 2 tampilan, yaitu tabel waktu (Gambar 3.7) dan tabel biasa (Gambar 3.8).



Gambar 3.7: Halaman Jadwal Kuliah Dalam Tabel Waktu

Hari	Waktu	Kode	Ruang	Nama	SKS	Kelas	Nama Dosen	Temu
Senin	14:00-15:00	AIF184002	Ruang Kuliah 9120	Skripsi 2	5	A	• Mariska Tri Adithia, S.Si., M.Sc., PDEng.	1
Selasa	11:00-14:00	AIF183120	Ruang 9016 Lab. Komputer 3	Pemrograman Permainan Komputer	3	A	• Jefvin Viriya, S.T.	1
Kamis	08:00-11:00	AIF183240	Ruang 9018 Lab Komputer 1	Sertifikasi Cyber Security Operations	3	A	• Chandra Wijaya, S.T., M.T.	1

Gambar 3.8: Halaman Jadwal Kuliah Tabel

- UTS

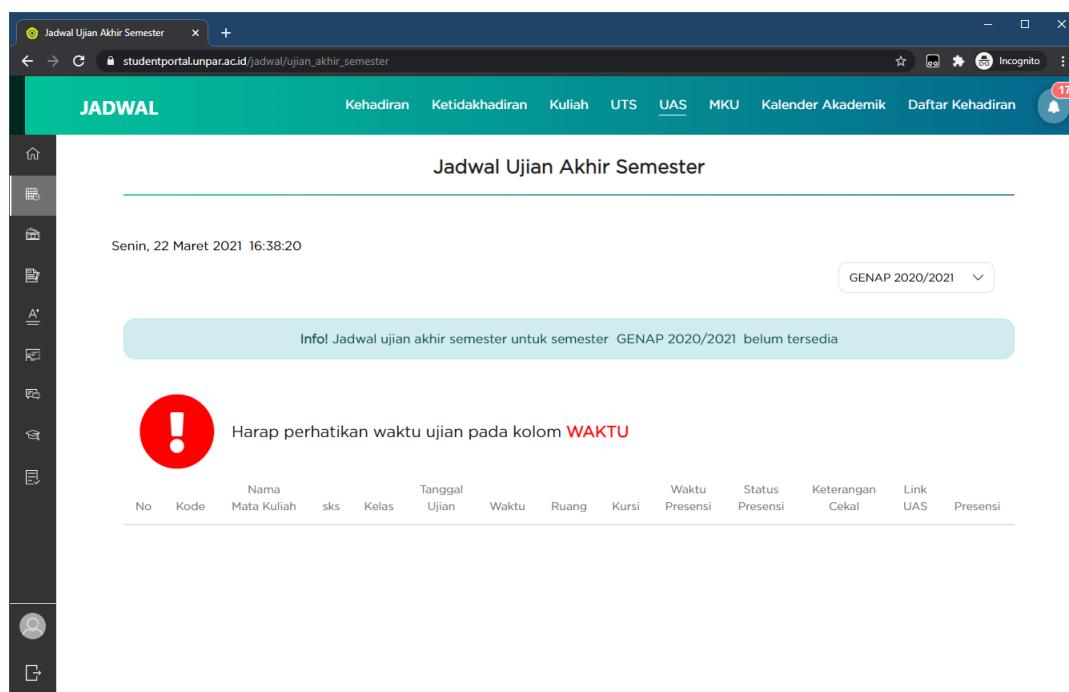
Submenu ini berisi tentang UTS yang dapat disusun per semester (Gambar 3.9).

No	Kode	Nama Mata Kuliah	sks	Kelas	Tanggal Ujian	Waktu	Ruang	Kursi	Waktu Presensi	Status Presensi	Link UTS	Presensi
----	------	------------------	-----	-------	---------------	-------	-------	-------	----------------	-----------------	----------	----------

Gambar 3.9: Halaman UTS

- UAS

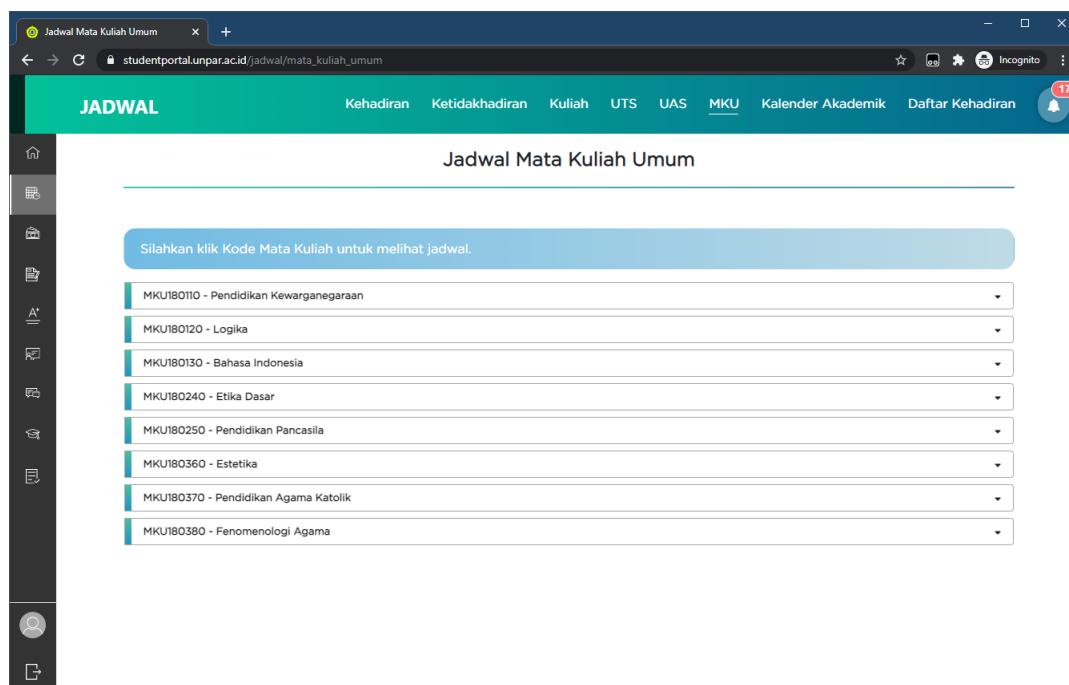
Submenu ini berisi tentang UAS yang dapat disusun per semester (Gambar 3.10).



Gambar 3.10: Halaman UAS

- MKU

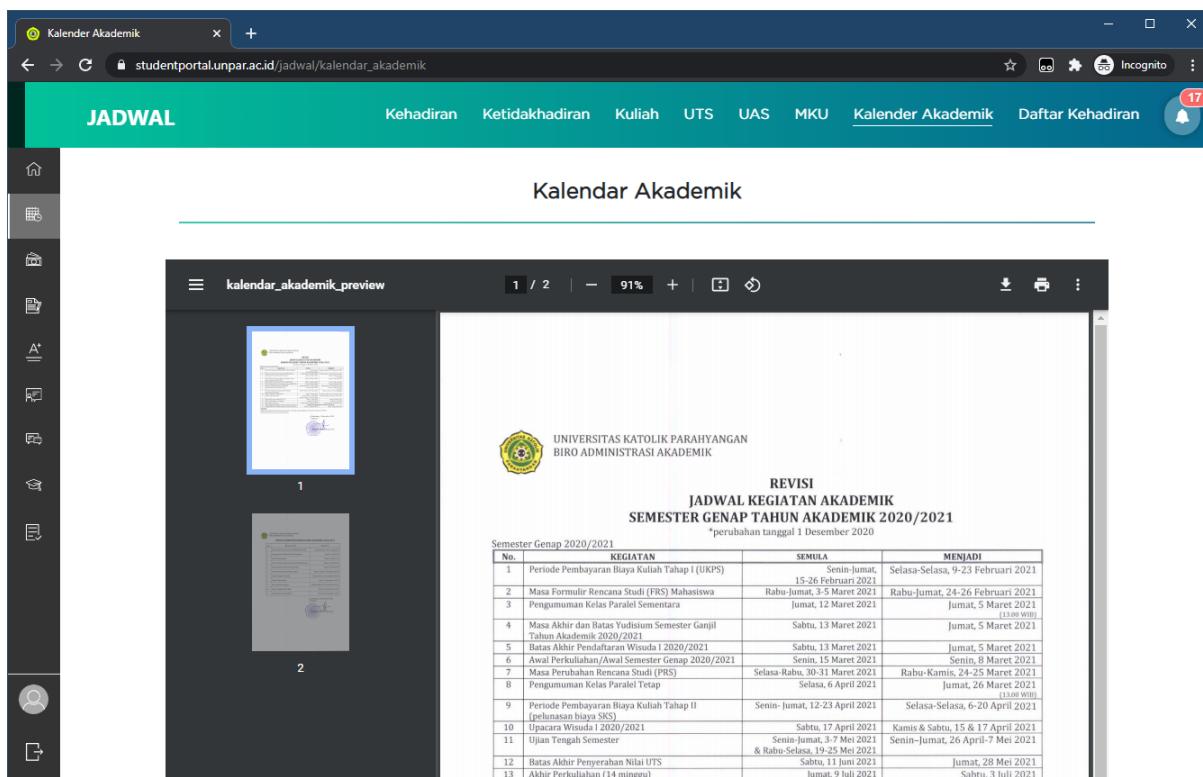
Submenu ini menampilkan seluruh jadwal Mata Kuliah Umum (MKU) yang memberikan informasi tentang kelas-kelas yang dibuka oleh Pusat Kajian Humaniora (PKH) (Gambar 3.11).



Gambar 3.11: Halaman MKU

- Kalender Akademik

Submenu ini menampilkan informasi mengenai kalender akademik UNPAR (Gambar 3.12).



Gambar 3.12: Halaman Kalender Akademik

- Daftar Kehadiran

Submenu ini menampilkan informasi mengenai daftar kehadiran mahasiswa pada setiap mata kuliah yang dapat disusun per semester (Gambar 3.13).

Daftar Kehadiran Genap - 2020/2021

GENAP 2020/2021											
No	Mata Kuliah	Dosen Pengajar	Sesi	Proses Pembelajaran	Hadir	Sakit	Izin	Alpa	Total Pertemuan Terverifikasi	Persentase Pembelajaran	Persentase Keseluruhan
1	AIF183120-03 Pemrograman Permainan Komputer	Jefvin Viriya, S.T.	<ul style="list-style-type: none"> Sesi 1 : Selasa , 11:00 -14:00, Ruang 9016 Lab. Komputer 3 	Kuliah	2	0	0	0	2	100%	100%
2	AIF183240-03 Sertifikasi Cyber Security Operations	Chandra Wijaya, S.T., M.I.	<ul style="list-style-type: none"> Sesi 1 : Kamis , 08:00 -11:00, Ruang 9018 Lab Komputer 1 	Kuliah	1	0	0	0	1	100%	100%
3	AIF184002-05 Skripsi 2	Mariskha Tri Adithia, S.Si, M.Sc., PDEng.	<ul style="list-style-type: none"> Sesi 1 : Senin , 14:00 -15:00, Ruang Kuliah 9120 	Kuliah	1	0	0	0	1	100%	100%

Gambar 3.13: Halaman Daftar Kehadiran

3.1.5 Pembayaran

Menu ini berfungsi untuk melihat data tagihan pembayaran uang kuliah, riwayat pembayaran, dan keterangan cara-cara pembayaran uang kuliah yang dapat disusun per semester (Gambar 3.14).

Jenis Tagihan	Jumlah Tagihan
HUTANG SEBELUMNYA	Rp. 0.-
Tahap 01	Rp. 6.000.000.-
Denda Tahap 01	Rp. 0.-
Tahap 02	Rp. 0.-
Denda Tahap 02	Rp. 0.-
PENAMBAHAN	Rp. 0.-
PENGEMBALIAN	Rp. 0.-
TOTAL	Rp. 6.000.000.-

Riwayat Pembayaran

Gambar 3.14: Halaman Pembayaran

3.1.6 FRS/PRS

Menu ini berfungsi sebagai formulir pengisian rencana studi awal (FRS), perubahan rencana studi (PRS) dan menampilkan informasi mata kuliah yang telah diambil saat FRS atau PRS (Gambar 3.15).

No	Kode MK	Nama Mata Kuliah	SKS
1	AIFI83010	Kerja Praktek 2	3
2	AIFI83120	Pemrograman Permainan Komputer	3
3	AIFI84002	Skripsi 2	5

Jumlah SKS : 11

Gambar 3.15: Tampilan FRS/PRS

3.1.7 Nilai

Menu Nilai terdiri dari beberapa submenu:

- Nilai per Semester

Submenu ini menampilkan informasi nilai per semester. Mahasiswa dapat melihat nilai sesuai dengan semester yang dipilih (Gambar 3.16).

No	Kode Mata Kuliah	Nama Mata Kuliah	SKS	Kelas	Nilai	AA	NA	Statistik Hasil Nilai
1	AIF183119	Keamanan Informasi	2	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai
2	AIF183341	Pola Komputasi Big Data	3	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai
3	AIF183348	Sistem Kecerdasan Bisnis	3	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai
4	AIF184001	Skripsi 1	3	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai
5	AIF184005	Komputer dan Masyarakat	2	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai
6	AIF184235	Layanan Berbasis Web	3	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai
7	AIF184303	Projek Sistem Informasi 2	3	A	Tampilkan Detail Nilai	-	-	Tampilkan Grafik Nilai

Keterangan:
\$: Nilai tidak dapat dilihat karena status pembayaran belum lunas
#: Nilai belum tersedia
%: Nilai sedang dalam proses

Gambar 3.16: Halaman Nilai Per Semester

- Daftar Perkembangan Studi

Submenu ini menampilkan seluruh riwayat mata kuliah dan nilai yang pernah ditempuh mahasiswa (Gambar 3.17). Submenu ini juga menampilkan statistik sks, nilai, dan indeks prestasi mahasiswa (Gambar 3.18).

Semester 1			
Kode MK	Nama MK	Nilai	Tahun Sem
AIF131105	Pengantar Informatika	A	20171
AIF181101	Pemodelan untuk Komputasi		
AIF181103	Matematika Dasar	A	20172
AIF181105	Pengantar Informatika		
AIF181107	Matematika Diskret	A	20171
MKU180110	Pendidikan Kewarganegaraan	A-	20181
MKU180120	Logika	A	20181
MKU180130	Bahasa Indonesia		

Semester 2			
Kode MK	Nama MK	Nilai	Tahun Sem
AIF131101	Pemrograman Berorientasi Objek	A	20171
AIF132205	Arsitektur Komputer	B	20172
AIF181100	Dasar-dasar Pemrograman		
AIF181104	Logika Informatika	A	20172

Gambar 3.17: Halaman Daftar Perkembangan Studi (1)

Nilai

Kode Semester:	Ket. Mk. Bid.	Ket. Mt Kuliah:
1 = Ganjil, 2 = Genap, 4 = Padat, 6 = Transfer	Peminatan: 01 = Teknologi Informasi Bisnis 02 = Ilmu Komputer 03 = Telematika	<ul style="list-style-type: none"> [M] = Mk. Kendali Mutu [X] = Mk. Disisihkan ["] = Mk. yang diambil semester (2017-2)

Nilai Akhir

Nilai	Akhir	A	A-	B+	B	B-	C+	C	D	E
Mata Kuliah	20	4	8	5	1	2	1	0	0	0
SKS	59	10	21	12	4	6	3	0	0	0

Nilai TOEFL

No	Tanggal	Skor
1	05-10-2020	540

Keterangan IP Mahasiswa

IPK (> 115 sks) : 3.56
IPS (> 17 sks) : 3.96

Jumlah sks

Ditempuh : 115 sks
Lulus Wajib : 83 sks
Lulus Pilihan : 32 sks
Lulus Wajib Peminatan : 0 sks
Lulus Pilihan Peminatan : 0 sks

Total Lulus : 115 sks

Syarat Kelulusan

Lulus min. 144 sks terdiri dari : Mk. Wajib + Mk. Pilihan
I.P. Lulus minimum : 2.00

Gambar 3.18: Halaman Daftar Perkembangan Studi (2)

- Riwayat Indeks Prestasi

Submenu ini menampilkan seluruh riwayat Indeks Prestasi Semester (IPS) dan Indeks Prestasi Kumulatif (IPK) setiap semester mahasiswa (Gambar 3.19).

Grafik Indeks Prestasi

Indeks Prestasi

Tahun-Semester

Legend: IPK (Blue Line), IPS (Red Line)

No.	Tahun	Sem	Status Akademik	Jumlah MK	SKS	IPS	IPK	SKS lulus
1	2017	1	Aktif	-	18	3.89	3.89	18
2	2017	2	Aktif	-	20	3.65	3.76	20
3	2018	1	Aktif	-	21	3.27	3.59	21
4	2018	2	Aktif	-	19	3.40	3.54	19
5	2019	1	Aktif	-	20	3.28	3.49	20
6	2019	2	Aktif	-	17	3.96	3.56	17
7	2020	1	Aktif	-	0	0	0	0

Keterangan:

IPS: Indeks Prestasi Semester (Indeks prestasi pada semester terkait).
IPK: Indeks Prestasi Kumulatif (berdasarkan nilai terbaik, termasuk nilai E jika nilai tersebut hanya satu-satunya yang terbaik).

Gambar 3.19: Halaman Riwayat Indeks Prestasi

- Nilai TOEFL

Submenu ini menampilkan seluruh riwayat skor dan detail skor *Test of English as Foreign Language* (TOEFL) yang pernah ditempuh mahasiswa (Gambar 3.20).

No	Tanggal	Listening	Structure	Reading	Total
1	05-10-2020	167	181	192	540

Gambar 3.20: Halaman Nilai TOEFL

3.1.8 Angket

Menu Angket merupakan halaman dimana mahasiswa diminta untuk mengisi angket dari para dosen yang mengajarnya di semester yang sedang ditempuh mahasiswa tersebut (Gambar 3.21).

Senin, 22 Maret 2021 | 16:40:59 WIB

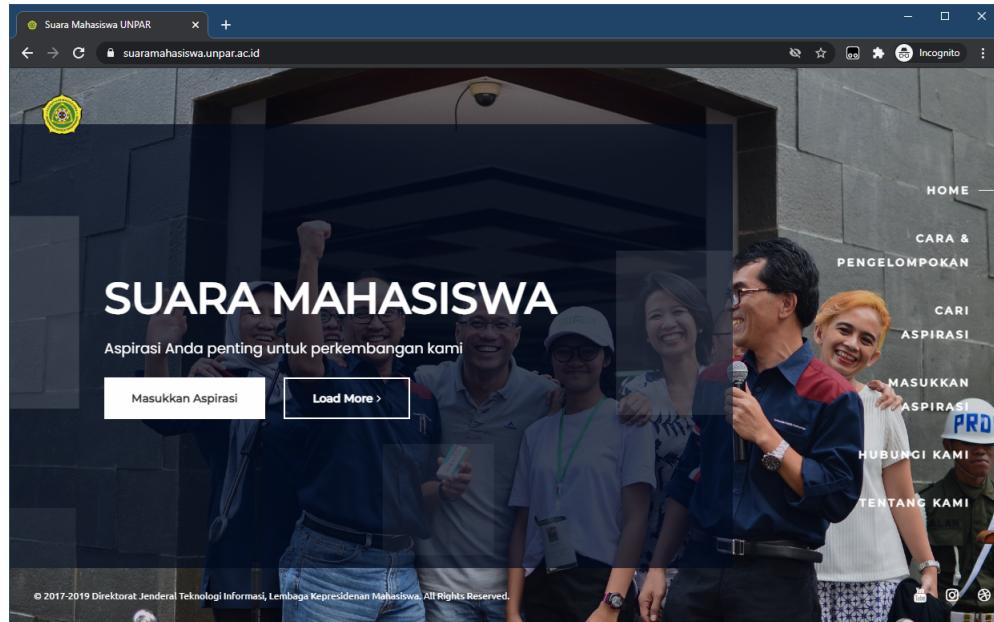
Daftar Angket Mata Kuliah

Tidak ada periode angket yang aktif

Gambar 3.21: Halaman Angket

3.1.9 Saran & Komentar

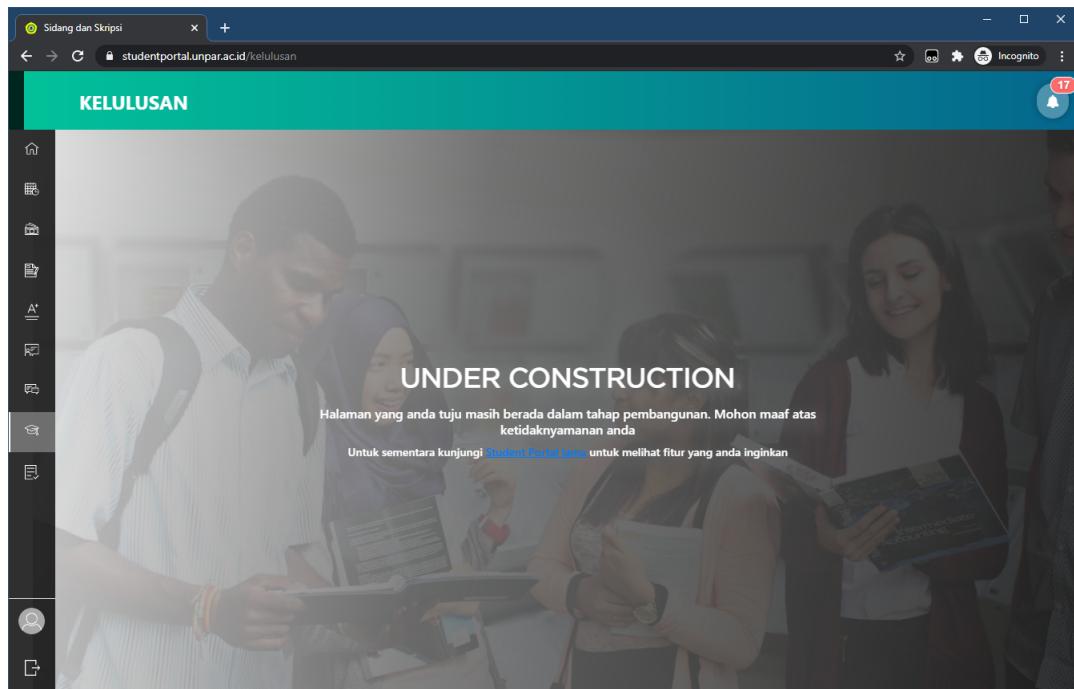
Menu Saran & Komentar akan membuka halaman <https://suaramahasiswa.unpar.ac.id/> (Gambar 3.22).



Gambar 3.22: Halaman Saran & Komentar

3.1.10 Kelulusan

Menu Kelulusan sedang dalam tahap pembangunan (Gambar 3.23).



Gambar 3.23: Halaman Kelulusan

3.1.11 Pengajuan

Menu Pengajuan merupakan halaman dimana mahasiswa dapat mengajukan topik skripsi atau tugas akhir (Gambar 3.24).

No	Semester	Mata Kuliah	Tanggal Pengajuan	Usulan Topik	Dosen Pembimbing		Status Pengajuan	Cetak Kartu Pembimbing	Edit
					Usulan	Ditunjuk			
1	2020/2021 Genap	AIF184002 - Skripsi 2						<button>Cetak</button>	<button>Edit</button>

Gambar 3.24: Halaman Pengajuan

3.2 Analisis SIAKAD

Subbab ini ditulis oleh dosen pembimbing.

SIAKAD adalah sistem informasi yang disediakan oleh Biro Teknologi Informasi kepada staf UNPAR, untuk menangani hal-hal yang berkaitan dengan akademik. SIAKAD dapat diakses pada alamat <https://siakad.unpar.ac.id> dari lingkungan jaringan UNPAR atau melalui VPN (*Virtual Private Network*). Modul-modul yang ditampilkan pada SIAKAD dapat berbeda, bergantung pada peran pengguna yang login. Pada subbab ini, akan dijabarkan modul-modul yang ditampilkan kepada dosen.

3.2.1 Login

Tata cara login untuk mengakses SIAKAD tidak jauh berbeda dengan pada Portal Akademik Mahasiswa. Dosen atau staf diarahkan ke situs web SSO (Gambar 3.1 dan 3.2). Perbedaannya hanyalah bahwa SIAKAD membatasi hanya akun dosen atau staf yang diperbolehkan masuk, berdasarkan alamat e-mailnya.

3.2.2 Mahasiswa

Untuk dosen, hanya ada satu submenu dari menu Mahasiswa, yaitu “Cari Mahasiswa”. Halaman ini memungkinkan dosen untuk mendapatkan daftar mahasiswa waliinya. Untuk setiap mahasiswa, ada tiga tautan yang dapat diklik, yaitu: NPM (untuk membuka pop-up Detail Mahasiswa), Data Diri, dan Data Akademik. Gambar 3.25 menunjukkan tampilan daftar mahasiswa wali dosen.

No	NPM	Nama Mahasiswa	Dosen Wali	Program Studi	Bidang Peminatan	Status Akademik	Aksi
1	2016730014	RICHARD WIJAYA		Teknik Informatika		Gencat	Data Diri Data Akademik
2	2016730068	GABRIEL PANJI LAZUARDI		Teknik Informatika		Aktif	Data Diri Data Akademik
3	2016730081	JONATHAN LAKSAMANA PURNOMO		Teknik Informatika		Aktif	Data Diri Data Akademik
4	2017730005	DIONISIUS SALVAVICTORI WANGGUR		Teknik Informatika		Aktif	Data Diri Data Akademik
5	2017730015	DAVID CHRISTOPHER SENTOSA		Teknik Informatika		Aktif	Data Diri Data Akademik
6	2017730016	STEPHEN HADI		Teknik Informatika		Aktif	Data Diri Data Akademik
7	2017730018	NICHOLAS ADITYA HALIM		Teknik Informatika		Aktif	Data Diri Data Akademik
8	2017730028	JOSHUA DELAVO SETIADI		Teknik Informatika		Aktif	Data Diri

Gambar 3.25: Tangkapan Layar Halaman Daftar Mahasiswa Wali

Detail Mahasiswa Detail ringkas mahasiswa ditampilkan dalam bentuk pop-up (Gambar 3.26) sehingga dosen wali dapat melihat secara sekilas data mahasiswa satu per satu tanpa perlu berpindah ke halaman lain.

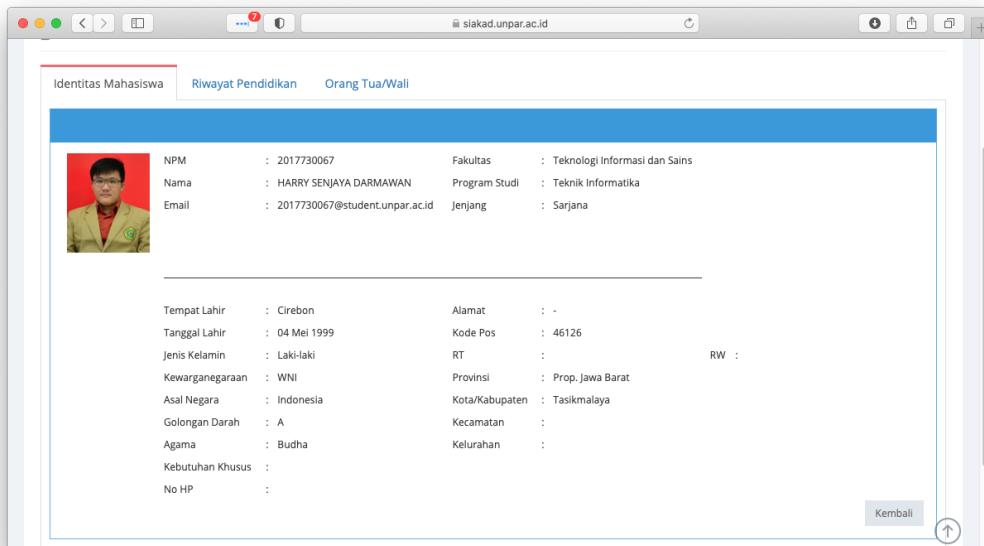
Mahasiswa		Detail Mahasiswa			
		NPM : 2017730067 Nama : HARRY SENJAYA DARMAWAN Email : 2017730067@student.unpar.ac.id Telp/Hp :	Fakultas : Teknologi Informatika dan Sains Program Studi : Teknik Informatika Jenjang : Sarjana Dosen Wali : -	Email Dosen : :	
Info Status Akademik & Pembayaran					
No	Tahun Akademik	Semester Akademik	Status Akademik	Status Keuangan	
				Tahap 1	Tahap 2
1	2020	Genap	Aktif	Sudah Bayar	Belum Bayar
2	2020	Ganjil	Aktif	Sudah Bayar	Sudah Bayar
3	2019	Pendek	Gencat		Sudah Bayar
4	2019	Genap	Aktif	Sudah Bayar	Sudah Bayar
5	2019	Ganjil	Aktif	Sudah Bayar	Sudah Bayar
6	2018	Genap	Aktif	Sudah Bayar	Sudah Bayar
7	2018	Ganjil	Aktif	Sudah Bayar	Sudah Bayar
8	2017	Pendek	Gencat		Sudah Bayar
9	2017	Genap	Aktif		
10	2017	Ganjil	Aktif		

Gambar 3.26: Tangkapan Layar Pop-up Detail Mahasiswa

Data Diri

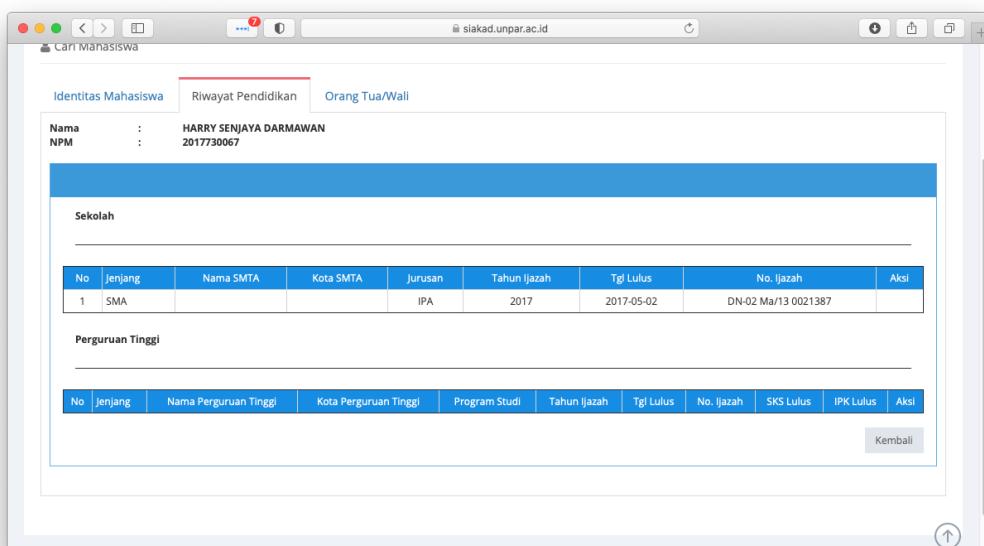
Data diri mahasiswa ditampilkan pada halaman baru yang terdiri dari beberapa tab. Halaman ini berisi detail data diri / pribadi mahasiswa yang tidak terkait data akademik di UNPAR. Halaman ini terdiri dari tiga tab, yaitu Identitas Mahasiswa, Riwayat Pendidikan, dan Orang Tua/Wali.

Identitas Mahasiswa Tab ini berisi data identitas mahasiswa, seperti foto, tempat/tanggal lahir, jenis kelamin, kewarganegaraan, serta data pribadi lainnya (Gambar 3.27).



Gambar 3.27: Tangkapan Layar Tab Identitas Mahasiswa

Riwayat Pendidikan Tab ini berisi riwayat pendidikan mahasiswa sebelum berkuliahan di UNPAR, baik SMA maupun di perguruan tinggi lain (Gambar 3.28).



Gambar 3.28: Tangkapan Layar Tab Riwayat Pendidikan

Orang Tua/Wali Tab ini berisi data identitas orang tua atau wali mahasiswa, seperti nama, tempat/tanggal lahir, dan sebagainya (Gambar 3.29).

Ayah	
Nama	: HARRY SENJAYA DARMAWAN
NPM	: 2017730067
Tempat Lahir : Kota Tasikmalaya	
Tanggal Lahir	: 17 Maret 1954
Kewarganegaraan	: WNI
Negara	: Indonesia
Agama	: Budha
Pekerjaan	: Virausaha
Pendidikan Terakhir	: SMTA
Kebutuhan Khusus	:
Status Kehidupan	: Masih hidup
No. Telp/No. HP	:

Ibu	
Nama	: PHAN MAN YIN
Tempat Lahir	: Kota Cirebon

Gambar 3.29: Tangkapan Layar Tab Orang Tua/Wali

Data Akademik

Data akademik mahasiswa ditampilkan pada halaman baru yang terdiri dari beberapa tab. Halaman ini berisi detail data akademik di UNPAR. Halaman ini terdiri dari lima tab, yaitu Ringkasan, Nilai Semester Ini, Nilai Per Tahun Semester, Nilai Berdasarkan Kurikulum, dan Status Akademik.

Ringkasan Tab ini berisi ringkasan data akademik mahasiswa, seperti statistik nilai, IPS, IPK, serta grafik perkembangannya (Gambar 3.30).

Ringkasan		Nilai Semester Ini - 2021	Nilai Per Tahun Semester	Nilai Berdasarkan Kurikulum	Status Akademik				
Indeks Prestasi									
Keterangan	Jumlah sks	Nilai							
IPK	134	3.58							
IPS (2020-21)	19	3.74							
Statistik Nilai									
Nilai	A	A-	B+	B	B-	C+	C	D	E
Jumlah	25	4	9	6	1	2	1	0	0
Jumlah Sks	72	10	24	15	4	6	3	0	0

Grafik Perkembangan IPS / IPK	
IPS / IPK	4.0
	3.5
	3.0
	2.5
	2.0
	1.5

Grafik Perkembangan Skor Akademik	
Skor Akademik	4.0
	3.5
	3.0
	2.5
	2.0
	1.5

Gambar 3.30: Tangkapan Layar Tab Ringkasan Akademik

Nilai Semester Ini Tab ini berisi data nilai dari kuliah-kuliah yang sedang diambil pada semester ini (Gambar 3.31).

The screenshot shows the 'Nilai Semester Ini' tab in the Siakad application. At the top, there is a student profile picture and basic information: NPM 2017730067, Name HARRY SENJAYA DARMAWAN, Email 2017730067@student.unpar.ac.id, Faculty Teknologi Informatika dan Sains, Program Studi Teknik Informatika, Jenjang Sarjana, Bidang Peminatan -, Dosen Wali -, Status Akademik Aktif, and Email Dosen Wali -. Below this, there are tabs: Ringkasan, Nilai Semester Ini - 202 (which is selected), Nilai Per Tahun Semester, Nilai Berdasarkan Kurikulum, and Status Akademik. The main content is a table titled 'Kelompok Tugas' showing grades for TP1 through TP20 for three subjects: Kerja Praktek 2, Pemrograman Permainan Komputer, and Skripsi 2. The table also includes a 'Total sks' row. A note at the bottom says 'Keterangan: jika nilai tidak muncul mungkin dikarenakan Anda belum melunasi seluruh pembayaran.' and a 'Kembali' button.

No	Kode MK	Nama Matakuliah	sks	Kelas	Kelompok Tugas																				Kelompok Ujian	AA	NA
					TP1	TP2	TP3	TP4	TP5	TP6	TP7	TP8	TP9	TP10	TP11	TP12	TP13	TP14	TP15	TP16	TP17	TP18	TP19	TP20			
1	AIF183010	Kerja Praktek 2	3																		0	0					
2	AIF183120	Pemrograman Permainan Komputer	3	A																	0	0					
3	AIF184002	Skripsi 2	5	A																	0	0					
Total sks					11																						

Keterangan: jika nilai tidak muncul mungkin dikarenakan Anda belum melunasi seluruh pembayaran.

Kembali

Gambar 3.31: Tangkapan Layar Tab Nilai Semester Ini

Nilai Per Tahun Semester Tab ini berisi data nilai mahasiswa, dibagi per tahun semester (Gambar 3.32).

The screenshot shows the 'Nilai Per Tahun Semester' tab in the Siakad application. At the top, there are tabs: Ringkasan, Nilai Semester Ini - 202 (selected), Nilai Per Tahun Semester, Nilai Berdasarkan Kurikulum, and Status Akademik. The main content is divided into two sections: 'Tahun Akademik 2020/2021' and 'Tahun Akademik 2021/2022'. Each section contains a table for 'Semester Genap' and 'Semester Ganjil'. The tables show grades for various subjects. A note at the bottom says 'Keterangan: jika nilai tidak muncul mungkin dikarenakan Anda belum melunasi seluruh pembayaran.' and a 'Kembali' button.

No.	Kode MK	Nama Matakuliah	sks	Kelas	Kelompok Tugas																				Kelompok Ujian	AA	NA
					TP1	TP2	TP3	TP4	TP5	TP6	TP7	TP8	TP9	TP10	TP11	TP12	TP13	TP14	TP15	TP16	TP17	TP18	TP19	TP20			
1	AIF183010	Kerja Praktek 2	3																		0	0					
2	AIF183120	Pemrograman Permainan Komputer	3	A																	0	0					
3	AIF184002	Skripsi 2	5	A																	0	0					
Total sks					11																						

Tahun Akademik 2020/2021

No.	Kode MK	Nama Matakuliah	sks	Kelas	Kelompok Tugas																				Kelompok Ujian	AA	NA
					TP1	TP2	TP3	TP4	TP5	TP6	TP7	TP8	TP9	TP10	TP11	TP12	TP13	TP14	TP15	TP16	TP17	TP18	TP19	TP20			
1	AIF183119	Keamanan Informatika	2	A		65															95	81	81	A			
2	AIF183341	Pola Komputasi Big Data Sistem	3	A	44	90	70	70	70	66											73	74	71	B			

Tahun Akademik 2021/2022

No.	Kode MK	Nama Matakuliah	sks	Kelas	Kelompok Tugas																				Kelompok Ujian	AA	NA
					TP1	TP2	TP3	TP4	TP5	TP6	TP7	TP8	TP9	TP10	TP11	TP12	TP13	TP14	TP15	TP16	TP17	TP18	TP19	TP20			
1	AIF183119	Keamanan Informatika	2	A		65															95	81	81	A			
2	AIF183341	Pola Komputasi Big Data Sistem	3	A	44	90	70	70	70	66											73	74	71	B			

Keterangan: jika nilai tidak muncul mungkin dikarenakan Anda belum melunasi seluruh pembayaran.

Kembali

Gambar 3.32: Tangkapan Layar Tab Nilai Per Tahun Semester

Nilai Berdasarkan Kurikulum Tab ini berisi data nilai mahasiswa, dibagi berdasarkan acuan linimasa kurikulum (Gambar 3.33).

The screenshot shows the 'Nilai Berdasarkan Kurikulum' tab in the SIAKAD application. At the top, there are tabs: Ringkasan, Nilai Semester Ini - 202, Nilai Per Tahun Semester, Nilai Berdasarkan Kurikulum (which is selected), and Status Akademik. Below the tabs, there are two main tables.

The first table, titled 'Komposisi sks:', shows the distribution of credits (Wajib, Peminatan) for each semester. The second table, titled 'Nilai Akhir', shows the final grades for each course across the same semesters.

Status tempuh	Rincian	Jumlah sks	Nilai Akhir																																						
			2017-2018					2018-2019					2019-2020					2020-2021					2021-2022					2022-2023					2023-2024					2024-2025			
Sks Tempuh		1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6
Lulus	Wajib	88	18	20			21	19			20	17			19																										
	Wajib Peminatan	0																																							
	Pilihan	46																																							
	Pilihan Peminatan	0																																							
Total	134																																								

Kode	Matakuliah	sks	Nilai Akhir																																									
			2017-2018					2018-2019					2019-2020					2020-2021					2021-2022					2022-2023					2023-2024					2024-2025					2025-2026	
1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6	1	2	4	5	6					
	Wajib																																											
AlF131105	Pengantar Informatika	3	A																																									
AlF181101	Pemodelan untuk Komputasi	3																																										
AlF181103	Matematika Dasar	4	A																																									
AlF181105	Pengantar Informatika	2																																										

Gambar 3.33: Tangkapan Layar Tab Nilai Berdasarkan Kurikulum

Status Akademik Tab ini berisi riwayat status akademik mahasiswa, termasuk perubahan dosen wali (Gambar 3.34).

The screenshot shows the 'Status Akademik' tab in the SIAKAD application. At the top, there are tabs: Ringkasan, Nilai Semester Ini - 202, Nilai Per Tahun Semester, Nilai Berdasarkan Kurikulum, and Status Akademik (selected). Below the tabs, there is a table.

Status Akademik							
No.	Tahun	Semester	Status	Dosen Wali	Catatan Status Akademik	Tgl. Status Akademik	Aksi
1	2020	Genap	Aktif				
2	2020	Ganjil	Aktif	Pascal Alfadian Nugroho, S.Kom., M.Comp.			
3	2019	Pendek	Gencat				
4	2019	Genap	Aktif	Pascal Alfadian Nugroho, S.Kom., M.Comp.			
5	2019	Ganjil	Aktif	Dr. Veronica Sri Moertini, Ir., M.T.			
6	2018	Genap	Aktif	Dr. Veronica Sri Moertini, Ir., M.T.			
7	2018	Ganjil	Aktif	Dr. Veronica Sri Moertini, Ir., M.T.			
8	2017	Pendek	Gencat				
9	2017	Genap	Aktif				
10	2017	Ganjil	Aktif	Dr. Veronica Sri Moertini, Ir., M.T.			

Gambar 3.34: Tangkapan Layar Tab Status Akademik

3.2.3 Pra Kuliah

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.4 Perkuliahan

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.5 Ujian

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.6 Nilai

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.7 Evaluasi

Pada menu Evaluasi, hanya ada satu submenu untuk dosen, yaitu Laporan Evaluasi, yang memiliki satu subsubmenu juga, yaitu Daftar Perkembangan Studi. Di halaman ini, dosen dapat mencari data mahasiswa walinya, berupa Daftar Perkembangan Studi dalam bentuk PDF, seperti ditunjukkan pada Gambar 3.35.

The screenshot shows a PDF document generated from the university's system. At the top, it displays the university's logo and name: UNIVERSITAS KATOLIK PARAHYANGAN, Jalan Ciumbulut 94 Bandung 40141, telp : +62 22 2030918-20 ext. 109401, 109422, fax: (022)203 1110, Telp.(022) 203 2655,(022) 204 2044. Below this, it lists student details: Nama: HARRY SENJAYA DARMAWAN, NPM: 2017730067, Status: Aktif. It also shows faculty information: Dosen Wali: [empty], Fakultas: Teknologi Informasi dan Sains, Program Studi: Teknik Informatika, Email: [empty], Bidang Peminatan: [empty]. A table follows, showing academic performance from 2017/2018 to 2025/2026. The table includes columns for Status tempuh (Wajib, Pilihan), Rincian (Jumlah sks), and various years. Below this is another table for subjects taken, showing Matakuliah, semester, grades, and semester 2 results. The tables include many rows of subject codes and their respective details.

Gambar 3.35: Tangkapan Layar PDF Daftar Perkembangan Studi

3.2.8 Skripsi

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.9 Sidang

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.10 Kelulusan

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.2.11 Pengumuman

Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

3.3 Data yang Dibutuhkan untuk *Screensaver*

3.3.1 Portal Akademik Mahasiswa

Halaman Utama

Pada Halaman Utama Portal Akademik Mahasiswa terdapat nama lengkap dan foto dari mahasiswa tersebut yang dapat diambil dan digunakan (Gambar 3.3 dengan kotak merah). Nama mahasiswa dapat diambil dengan mencari elemen "div" yang memiliki kelas "namaUser d-none d-lg-block mr-3", sehingga kueri css yang dihasilkan adalah "div[class=namaUser d-none d-lg-block mr-3]". Foto mahasiswa dapat diambil dengan mencari elemen "img" yang memiliki kelas "img-fluid fotoProfil", sehingga kueri css yang dihasilkan adalah "img[class=img-fluid fotoProfil]". Terdapat beberapa perubahan (Kode 3.2) yang perlu dilakukan terhadap skripsi Andrianto Sugiarto [5]:

1. Menghapus pemanggilan fungsi validateTLCertificates() dikarenakan sudah *deprecated* [6].
2. Sebelumnya semester yang sedang dijalani mahasiswa diambil dari menu FRS/PRS, namun karena terdapat perubahan dimana menu FRS/PRS tidak menuju ke Portal Akademik Mahasiswa melainkan menuju ke https://restu.unpar.ac.id/frs_prs, sehingga perlu melakukan *login* kembali apabila mengakses URL tersebut, maka semester yang sedang dijalani mahasiswa diambil dari menu Nilai. Dengan begitu, kueri css untuk pengambilan semester yang sedang dijalani juga perlu diubah, yaitu dengan mencari elemen "select" yang memiliki kelas "custom-select mr-3", sehingga kueri css yang dihasilkan adalah "select[class=custom-select mr-3]". Kemudian mengolah data yang didapat sehingga dapat dijadikan objek bertipe TahunSemester sesuai dengan SIAModels.

Kode 3.2: Perubahan Implementasi Jsoup Halaman Utama

```

1 @@ -96,24 +96,22 @@ public class Scraper {
2     Connection connection = Jsoup.connect(HOME_URL);
3     connection.cookie("ci_session", phpsessid);
4     connection.timeout(0);
5     - connection.validateTLCertificates(false);
6     connection.method(Connection.Method.GET);
7     Response resp = connection.execute();
8     Document doc = resp.parse();
9     String nama = doc.select("div[class=namaUser d-none d-lg-block mr-3]").text();
10    mhs.setNama(nama.substring(0, nama.indexOf(mhs.getEmailAddress())));
11    Element photo = doc.select("img[class=img-fluid fotoProfil]").first();
12    String photoPath = photo.attr("src");
13    mhs.setPhotoPath(photoPath);
14    - connection = Jsoup.connect(FRSPRS_URL);
15    + connection = Jsoup.connect(NILAI_URL);
16    connection.cookie("ci_session", phpsessid);
17    connection.timeout(0);
18    - connection.validateTLCertificates(false);
19    connection.method(Connection.Method.GET);
20    resp = connection.execute();
21    doc = resp.parse();
22    - String curr_sem = doc.select(".custom-selectContent span").text();
23    + String[] sem_set = parseSemester(curr_sem);
24    + Elements curr_sem = doc.select("select[class=custom-select mr-3]");
25    + String[] sem_set = parseSemester(curr_sem.first().child(curr_sem.first().childrenSize() - 1).text());
26    TahunSemester currTahunSemester = new TahunSemester(Integer.parseInt(sem_set[0]),
27                Semester.fromString(sem_set[1]));
28    return currTahunSemester;
29 @@ -214,7 +212,7 @@ public class Scraper {
30 }
31
32     public String[] parseSemester(String sem_raw) {
33     - String[] sem_set = sem_raw.split("//")[0].split("-");
34     + String[] sem_set = sem_raw.split("//")[0].split(" ");
35     return new String[]{sem_set[1].trim(), sem_set[0].trim()};
36 }
```

Halaman Profil

Pada Halaman Profil (Gambar 3.4 dengan kotak merah), tanggal lahir mahasiswa akan diambil untuk ditampilkan pada *screensaver*. Implementasi jsoup tersebut belum diimplementasikan pada skripsi Andrianto Sugiarto [5], sehingga perlu dilakukan penambahan fitur untuk mengambil data tersebut. Tanggal lahir mahasiswa dapat diambil dengan mencari elemen "div" yang memiliki kelas "offset-md-1 col-md-10 col-12 headerWrapper my-0 border-bottom", sehingga kueri css yang dihasilkan adalah "div[class=offset-md-1 col-md-10 col-12 headerWrapper my-0 border-bottom]".

Halaman Daftar Perkembangan Studi

Pada Halaman Daftar Perkembangan Studi (Gambar 3.17 dan 3.18 dengan kotak merah), data yang dapat dimanfaatkan dari halaman ini adalah IPK, IPS, jumlah sks yang lulus, dan jumlah sks yang ditempuh. Namun, pada SIAModels sudah terdapat *method* yang melakukan kalkulasi untuk mendapatkan data-data tersebut, sehingga tidak perlu dilakukan pengambilan data menggunakan jsoup. Untuk dapat memanfaatkan *method* tersebut diperlukan seluruh riwayat mata kuliah dan nilai yang pernah ditempuh mahasiswa, sehingga perlu dilakukan pengambilan data menggunakan jsoup. Implementasi pengambilan data tersebut sudah diimplementasikan sebelumnya pada skripsi Andrianto Sugiarto [5]. Proses dari pengambilan data tersebut yaitu:

- Mengambil data nilai berdasarkan tahun dan semester dengan mencari elemen "select" yang memiliki id "dropdownSemester", dan memiliki kelas "custom-select mr-3" sehingga kueri css yang dihasilkan adalah "select#dropdownSemester.custom-select.mr-3".
- Dikarenakan perlunya melakukan koneksi berkali-kali sebanyak semester yang telah ditempuh mahasiswa, sehingga dibutuhkan waktu yang tidak sebentar. Karena pada halaman nilai tidak dapat menampilkan seluruh semester seperti Portal Akademik Mahasiswa yang lama, sehingga untuk mengatasi masalah ini dibuat menjadi paralel. Untuk itu dibuat kelas yang mengimplementasikan kelas *interface Runnable*, yaitu kelas *MultipleRequest*. Kelas inilah yang melakukan koneksi ke setiap semester yang telah ditempuh mahasiswa, dan mengambil data-data tersebut.

Namun, terdapat beberapa perubahan (Kode 3.3) yang perlu dilakukan:

1. Menghapus pemanggilan fungsi validateTLCertificates() dikarenakan sudah *deprecated* [6].
2. Indeks yang digunakan untuk melakukan kueri css berubah, sehingga untuk mengantisipasi tersebut, indeks yang digunakan menjadi *size* dari kueri css dikurangi 1.

Kode 3.3: Perubahan Implementasi Jsoup Halaman Daftar Perkembangan Studi

```

1 @@ -50,7 +50,7 @@ public class MultipleRequest implements Runnable {
2     Connection.Response resp = connection.execute();
3     Document doc = resp.parse();
4
5     - Element script = doc.select("script").get(10);
6     + Element script = doc.select("script").get(doc.select("script").size()-1);
7     String scriptDataMataKuliah = script.html().substring(script.html().indexOf("var data_mata_kuliah = []"), script.
8         html().indexOf("var data_angket = []"));
9     engine.eval(scriptDataMataKuliah);
10    ScriptObjectMirror dataMataKuliah = (ScriptObjectMirror) engine.get("data_mata_kuliah");
11
12 @@ -131,7 +131,6 @@ public class Scraper {
13     Connection connection = Jsoup.connect(NILAI_URL);
14     connection.cookie("ci_session", phpsessid);
15     connection.timeout(0);
16     - connection.validateTLCertificates(false);
17     connection.method(Connection.Method.POST);
18     Response resp = connection.execute();
      Document doc = resp.parse();

```

Halaman TOEFL

Pada Halaman TOEFL (Gambar 3.20 dengan kotak merah), riwayat skor TOEFL dapat diambil dengan mencari elemen "table" yang memiliki elemen "tbody" didalamnya, serta memiliki elemen "tr" didalam elemen "tbody". Terdapat beberapa perubahan yang terjadi pada halaman TOEFL

semenjak skripsi Andrianto Sugiarto [5], yang mengakibatkan perlunya perubahan (Kode 3.4) terhadap implementasi jsoup:

1. Menghapus pemanggilan fungsi validateTLCertificates() dikarenakan sudah *deprecated* [6].
2. Perubahan format tanggal TOEFL pada Portal Akademik Mahasiswa menjadi dd-mm-yyyy.

Kode 3.4: Perubahan Implementasi Jsoup TOEFL

```

1 @@ -174,7 +174,6 @@ public class Scraper {
2     Connection connection = Jsoup.connect(TOEFL_URL);
3     connection.cookie("ci_session", phpsessid);
4     connection.timeout(0);
5     -    connection.validateTLCertificates(false);
6     connection.method(Connection.Method.POST);
7     Response resp = connection.execute();
8     Document doc = resp.parse();
9 @@ -183,45 +182,7 @@ public class Scraper {
10        for (int i = 0; i < nilaiTOEFL.size(); i++) {
11            Element nilai = nilaiTOEFL.get(i).select("td").get(5);
12            Element tgl_toefl = nilaiTOEFL.get(i).select("td").get(1);
13            -    String[] tanggal = tgl_toefl.text().split(" ");
14            switch (tanggal[1].toLowerCase()) {
15            -        case "januari":
16                -            tanggal[1] = "1";
17                break;
18            -        case "februari":
19                -            tanggal[1] = "2";
20                break;
21            -        case "maret":
22                -            tanggal[1] = "3";
23                break;
24            -        case "april":
25                -            tanggal[1] = "4";
26                break;
27            -        case "mei":
28                -            tanggal[1] = "5";
29                break;
30            -        case "juni":
31                -            tanggal[1] = "6";
32                break;
33            -        case "juli":
34                -            tanggal[1] = "7";
35                break;
36            -        case "agustus":
37                -            tanggal[1] = "8";
38                break;
39            -        case "september":
40                -            tanggal[1] = "9";
41                break;
42            -        case "oktober":
43                -            tanggal[1] = "10";
44                break;
45            -        case "november":
46                -            tanggal[1] = "11";
47                break;
48            -        case "desember":
49                -            tanggal[1] = "12";
50                break;
51        }
52 +    String[] tanggal = tgl_toefl.text().split("-");
53
54     LocalDate localDate = LocalDate.of(Integer.parseInt(tanggal[2]), Integer.parseInt(tanggal[1]),
55                                         Integer.parseInt(tanggal[0]));

```

3.3.2 SIAKAD

Subbab ini ditulis oleh dosen pembimbing.

Mendapatkan daftar mahasiswa

Untuk mendapatkan daftar mahasiswa, bisa memanfaatkan halaman “Cari Mahasiswa”. Caranya dengan mensimulasikan penekanan tombol “Cari” dengan parameter *default*, yang akan menampilkan daftar mahasiswa dari dosen wali yang bersangkutan. Hasilnya dalam bentuk tabel dan di-parsing, dengan ketentuan sebagai berikut:

1. NPM tersedia pada kolom kedua.
2. Nama Mahasiswa tersedia pada kolom ketiga.
3. Kolom-kolom lainnya diabaikan.

Mendapatkan detail mahasiswa

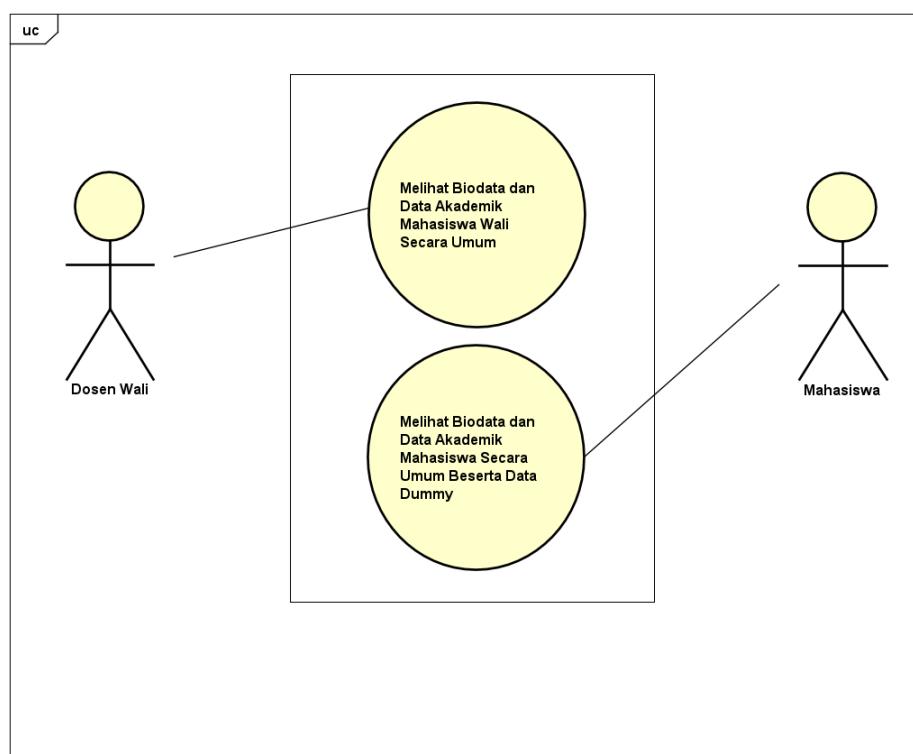
Riwayat Nilai Pada SIAKAD, riwayat nilai mahasiswa tersedia pada beberapa halaman:

1. **Nilai Per Tahun Semester** berisikan tabel nilai yang dibagi per tahun semester. Data dari halaman inilah yang dipakai untuk mendapatkan nilai mahasiswa, mencakup: kode dan nama mata kuliah, SKS, kelas, TP1 s.d. TP20, UTS, UAS, AA, dan NA.
 2. **Nilai Berdasarkan Kurikulum** berisikan tabel nilai yang dibagi berdasarkan linimasa kurikulum. Data ini kurang cocok dipakai karena tidak mengandung nilai tugas dan ujian, hanya nilai akhir saja.
 3. **Evaluasi** berisikan tabel nilai lengkap dalam bentuk Daftar Perkembangan Studi dengan format PDF. Data ini sulit diekstraksi karena tidak dalam bentuk HTML.
- Kesimpulannya, riwayat nilai diambil dari halaman “Nilai Per Tahun Semester”.

Data Diri Pada SIAKAD, data diri dapat diambil dari halaman “Data Diri”, tab identitas mahasiswa. Foto tersedia dalam bentuk Data URL¹ yang di-encode dalam format *base64*. Data-data diri lainnya dapat diambil dengan cukup sederhana dari berbagai elemen yang ada di tab tersebut.

3.4 Analisis Sistem *Screensaver*

3.4.1 *Use Case Screensaver*



Gambar 3.36: Diagram *Use Case Screensaver*

Pada diagram *use case screensaver* (Gambar 3.36) terdapat 2 fitur utama dari *screensaver* yaitu, melihat biodata dan data akademik mahasiswa wali secara umum bagi dosen, serta melihat biodata dan data akademik mahasiswa secara umum beserta data dummy bagi mahasiswa. Dimana biodata yang dimaksud adalah nama, angkatan, tanggal lahir. Serta data akademik yang dimaksud adalah status, email, nilai TOEFL, IPS, IPK, SKS lulus, SKS tempuh mahasiswa.

¹https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Data_URLs

Skenario *Use Case*

1. Melihat Biodata dan Data Akademik Mahasiswa Wali Secara Umum

- Nama: Melihat Biodata dan Data Akademik Mahasiswa Wali Secara Umum
- Aktor: Dosen
- Deskripsi: Menjalankan *screensaver* yang menampilkan biodata dan data akademik mahasiswa wali secara umum.
- Kondisi awal: Komputer tidak digunakan selama beberapa saat.
- Kondisi Akhir: *Screensaver* berjalan dengan menampilkan biodata dan data akademik mahasiswa wali secara umum.
- Skenario utama:

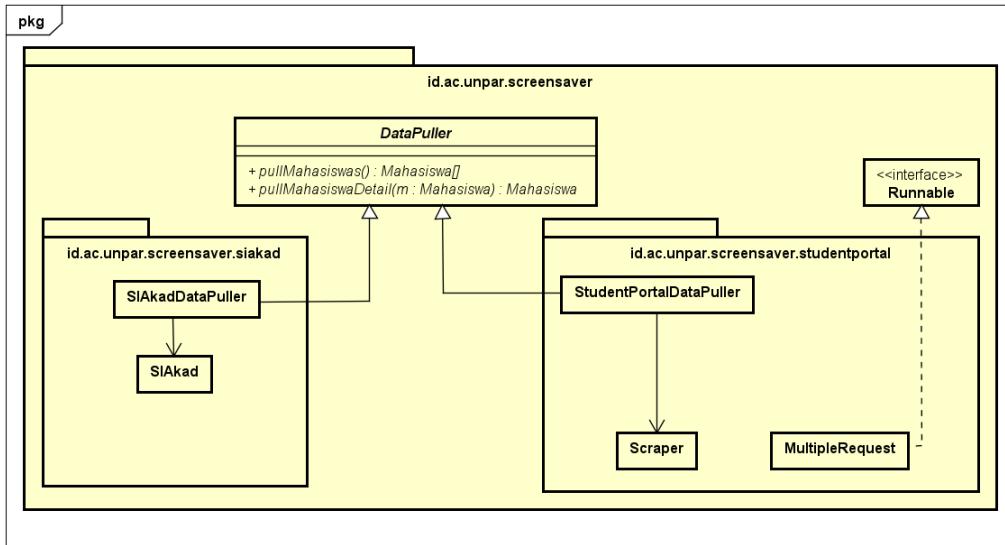
No	Aksi Aktor	Reaksi Sistem
1	Dosen tidak menggunakan komputer selama beberapa saat.	<i>Screensaver</i> berjalan dengan menampilkan biodata dan data akademik mahasiswa wali secara umum.

2. Melihat Biodata dan Data Akademik Mahasiswa Secara Umum Beserta Data Dummy

- Nama: Melihat Biodata dan Data Akademik Mahasiswa Secara Umum Beserta Data Dummy
- Aktor: Mahasiswa
- Deskripsi: Menjalankan *screensaver* yang menampilkan biodata dan data akademik mahasiswa secara umum beserta dengan data dummy.
- Kondisi awal: Komputer tidak digunakan selama beberapa saat.
- Kondisi Akhir: *Screensaver* berjalan dengan menampilkan biodata dan data akademik mahasiswa secara umum beserta dengan data dummy.
- Skenario utama:

No	Aksi Aktor	Reaksi Sistem
1	Mahasiswa tidak menggunakan komputer selama beberapa saat.	<i>Screensaver</i> berjalan dengan menampilkan biodata dan data akademik mahasiswa secara umum beserta dengan data dummy.

3.4.2 Perancangan Kelas Screensaver



Gambar 3.37: Perancangan Kelas Screensaver

Gambar 3.37 merupakan perancangan kelas *screensaver*. Terdapat sebuah *abstract class* yang bernama *DataPuller* yang memiliki 2 buah *method* yaitu:

- **public abstract Mahasiswa[] pullMahasiswa()**
Berfungsi untuk mengambil seluruh daftar mahasiswa.
- **public abstract Mahasiswa pullMahasiswaDetail(Mahasiswa m)**
Berfungsi untuk mengambil detail lebih lanjut mengenai data mahasiswa tersebut.

Kelas *DataPuller* diturunkan ke 2 buah kelas yaitu *SIAkadDataPuller*, dan *StudentPortalDataPuller*, dimana kedua kelas tersebut berfungsi sebagai pintu masuk dalam mengambil data mahasiswa. Data mahasiswa pada halaman *SIAKAD* diambil oleh kelas yang bernama *SIAkad*, sedangkan data mahasiswa pada halaman *Portal Akademik Mahasiswa* diambil oleh kelas yang bernama *Scraper*. Penjelasan kelas *MultipleRequest* terdapat pada bagian 3.3.1.

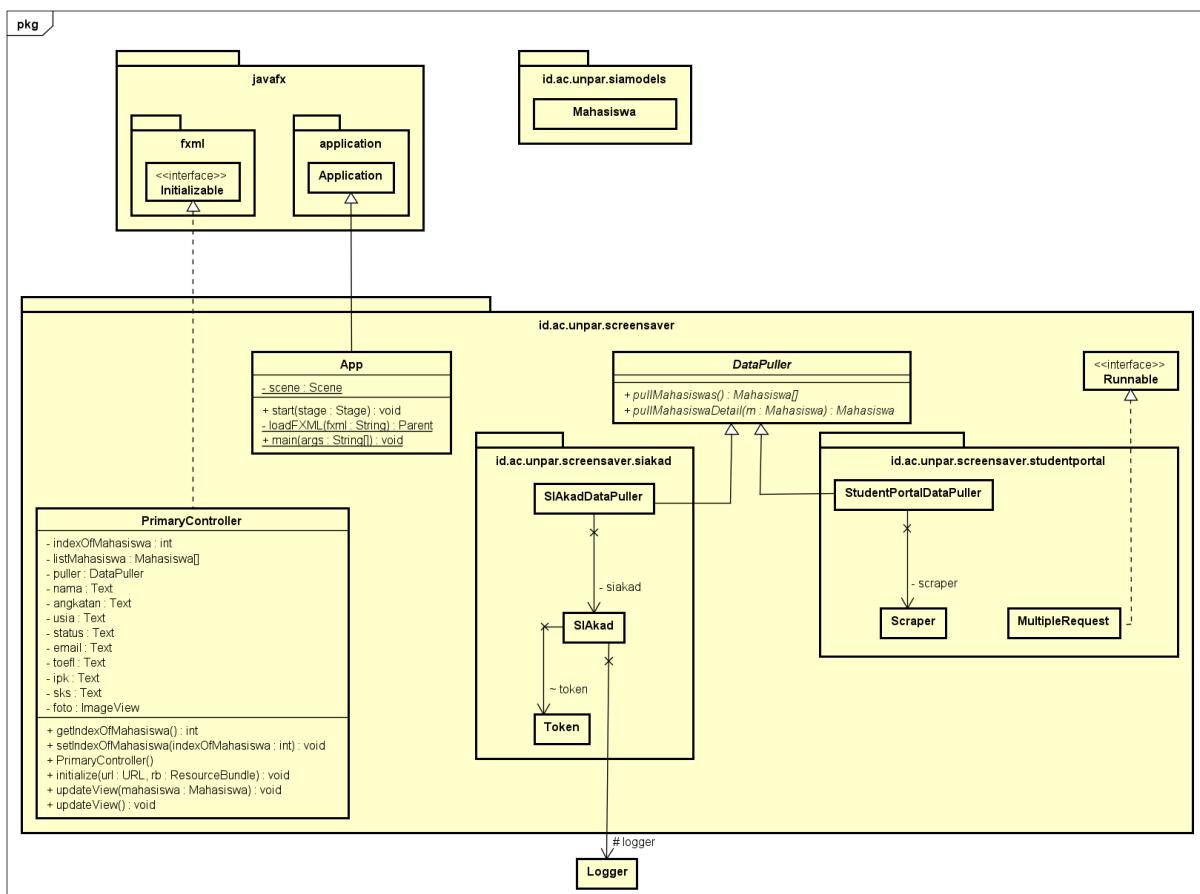
BAB 4

PERANCANGAN

Pada bab ini akan dijelaskan mengenai perancangan kelas beserta deskripsi kelas dan fungsinya, serta perancangan antarmukanya.

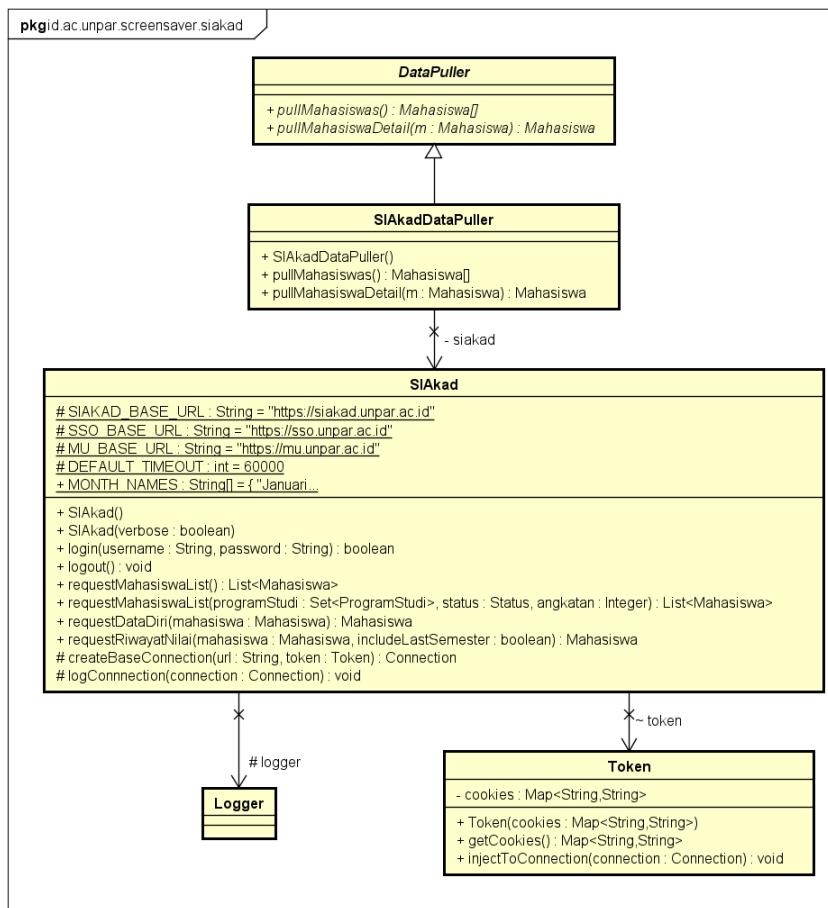
4.1 Perancangan Kelas

Diagram kelas secara keseluruhan dapat dilihat pada gambar 4.1, dimana terdapat *package screensaver* yang didalamnya terdapat *package siakad* (Gambar 4.2), dan *package studentportal* (Gambar 4.3). Penjelasan mengenai kelas, *method*, dan atribut jsoup terdapat pada bagian 2.1. Penjelasan mengenai kelas, *method*, dan atribut JavaFX serta FXML terdapat pada bagian 2.2. Penjelasan mengenai kelas, *method*, dan atribut SIAModels terdapat pada bagian 2.3.



Gambar 4.1: Diagram Kelas Keseluruhan

Penjelasan mengenai kelas, *method*, dan atribut pada *package screensaver* adalah sebagai berikut:



Gambar 4.2: Diagram Kelas SIAKAD

1. App

Kelas ini merupakan turunan dari kelas [Application 2.2.1](#). Atribut yang dimiliki kelas ini adalah sebagai berikut:

- **private static Scene scene:** menyimpan objek Scene.

Method yang dimiliki kelas ini adalah sebagai berikut:

- **public void start(Stage stage)**

Berfungsi untuk menampilkan Stage dengan Scene yang digunakan.

Parameter:

– stage: stage utama untuk aplikasi ini.

- **private static Parent loadFXML(String fxml)**

Berfungsi untuk memuat hierarki objek dari dokumen FXML.

Parameter:

– fxml: dokumen fxml.

Kembalian: hierarki objek dari dokumen FXML.

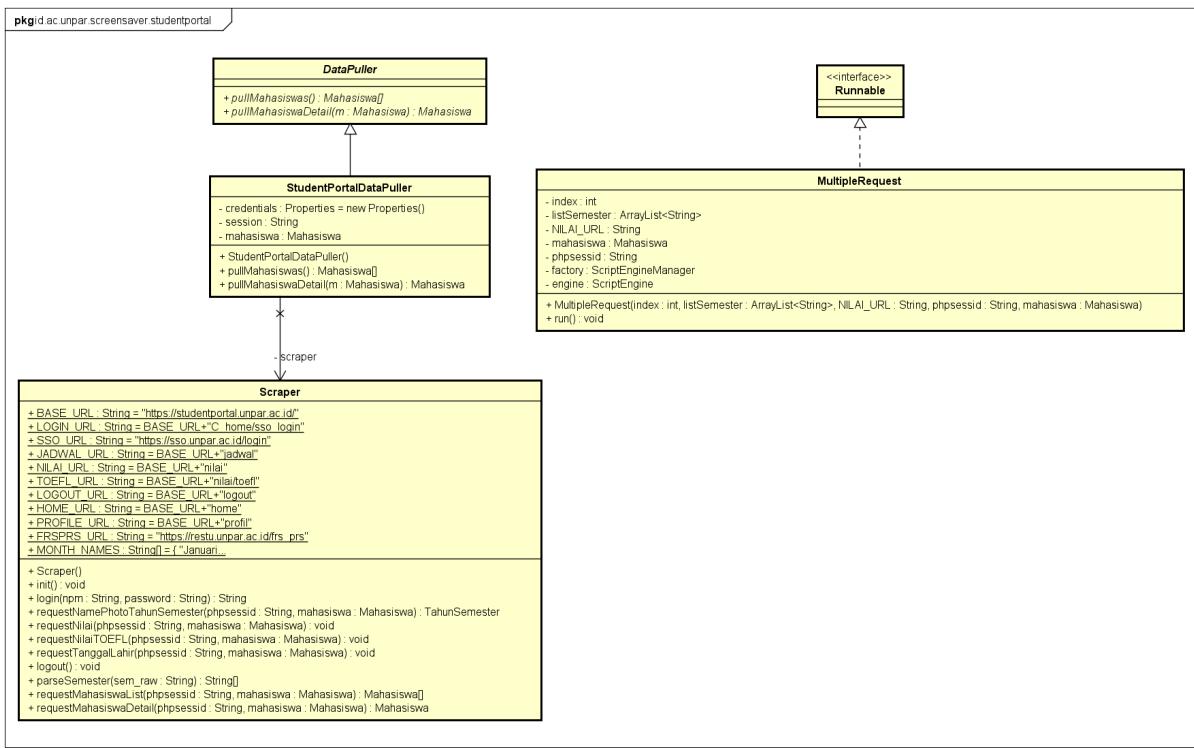
- **public static void main(String[] args)**

Berfungsi untuk memanggil method `launch()` pada kelas [Application](#).

2. PrimaryController

Kelas ini mengimplementasikan kelas [interface Initializable 2.2.2](#). Atribut yang dimiliki kelas ini adalah sebagai berikut:

- **private int indexOfMahasiswa:** menyimpan indeks mahasiswa yang akan ditampilkan pada screensaver.
- **private Mahasiswa[] listMahasiswa:** menyimpan daftar mahasiswa yang akan di-



Gambar 4.3: Diagram Kelas Studentportal

tampilkan pada *screensaver*.

- **private DataPuller puller:** menyimpan objek **DataPuller**.
- **private Text nama:** menyimpan objek **Text** yang digunakan untuk keterangan nama mahasiswa.
- **private Text angkatan:** menyimpan objek **Text** yang digunakan untuk keterangan angkatan mahasiswa.
- **private Text usia:** menyimpan objek **Text** yang digunakan untuk keterangan usia mahasiswa.
- **private Text status:** menyimpan objek **Text** yang digunakan untuk keterangan status mahasiswa.
- **private Text email:** menyimpan objek **Text** yang digunakan untuk keterangan *email* mahasiswa.
- **private Text toefl:** menyimpan objek **Text** yang digunakan untuk keterangan nilai TOEFL mahasiswa.
- **private Text ipk:** menyimpan objek **Text** yang digunakan untuk keterangan IPK mahasiswa.
- **private Text sks:** menyimpan objek **Text** yang digunakan untuk keterangan SKS mahasiswa.
- **private ImageView foto:** menyimpan objek **ImageView** yang digunakan untuk foto mahasiswa.

Method yang dimiliki kelas ini adalah sebagai berikut:

- **public int getIndexOfMahasiswa()**
Berfungsi untuk mengambil indeks mahasiswa.
Kembalian: indeks mahasiswa.
 - **public void setIndexOfMahasiswa(int indexOfMahasiswa)**
Berfungsi untuk menyimpan indeks mahasiswa.
- Parameter:**

- `indexOfMahasiswa`: indeks mahasiswa.
- `public PrimaryController()`
Berfungsi sebagai *constructor* kelas PrimaryController.
- `public void initialize(URL url, ResourceBundle rb)`
Berfungsi untuk menginisialisasi pengontrol setelah elemen akarnya diproses sepenuhnya. *Method* ini juga berfungsi untuk mengambil *list* mahasiswa beserta dengan keterangan detil mahasiswa. **Parameter:**
 - `url`: Lokasi yang digunakan untuk menyelesaikan jalur relatif untuk objek `root`, atau `null` jika lokasi tidak diketahui. item `rb`: *resource* yang digunakan untuk melokalisasi objek `root`, atau `null` jika objek `root` tidak dilokalkan.
- `public void updateView(Mahasiswa mahasiswa)`
Berfungsi untuk memperbarui tampilan *screensaver* dengan data mahasiswa yang menjadi parameter.
Parameter:
 - `mahasiswa`: objek mahasiswa.
- `public void updateView()`
Berfungsi untuk memperbarui tampilan *screensaver* ketika koneksi internet tidak berjalan dengan normal.

3. DataPuller

Kelas ini merupakan *abstract class*. *Method* yang dimiliki kelas ini adalah sebagai berikut:

- `public abstract Mahasiswa[] pullMahasiswas()`
Merupakan *abstract method* yang akan diimplementasikan oleh turunannya.
- `public abstract Mahasiswa pullMahasiswaDetail(Mahasiswa m)`
Merupakan *abstract method* yang akan diimplementasikan oleh turunannya.

Penjelasan mengenai kelas, *method*, dan atribut pada *package studentportal* adalah sebagai berikut:

1. StudentPortalDataPuller

Kelas ini merupakan turunan dari kelas `DataPuller` [3](#). Kelas ini memiliki fungsi untuk mengambil *npm* dan *password* mahasiswa yang disimpan dalam sebuah *file*, dan kemudian memanggil *method* pada kelas `Scaper`. Atribut yang dimiliki kelas ini adalah sebagai berikut:

- `private Scaper scraper`: menyimpan objek `Scaper`.
- `private final Properties credentials`: mengakses *file* yang berisi *npm* dan *password* mahasiswa.
- `private Mahasiswa mahasiswa`: menyimpan objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- `private String session`: menyimpan *session* yang dapat digunakan untuk mengakses menu di Portal Akademik Mahasiswa.

Method yang dimiliki kelas ini adalah sebagai berikut:

- `public StudentPortalDataPuller()`
Berfungsi sebagai *constructor* kelas `StudentPortalDataPuller`.
- `public Mahasiswa[] pullMahasiswas()`
Berfungsi untuk memanggil *method* `requestMahasiswaList` pada kelas `Scaper` [\(2\)](#).
Kembalian: *array* yang berisi seluruh mahasiswa yang memiliki dosen wali yang sama dengan yang melakukan *login* (dalam implementasi menggunakan Portal Akademik Mahasiswa, mahasiswa yang diambil adalah hanya mahasiswa yang melakukan *login*).
- `public Mahasiswa pullMahasiswaDetail()`
Berfungsi untuk memanggil *method* `requestMahasiswaDetail` pada kelas `Scaper` [\(2\)](#).
Kembalian: objek mahasiswa yang telah ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.

2. Scaper

Kelas ini memiliki fungsi untuk melakukan pengambilan data dari Portal Akademik Mahasiswa untuk kemudian diolah dan ditampilkan. Atribut yang dimiliki kelas ini adalah sebagai berikut:

- `public static final String BASE_URL`: menyimpan *url* utama Portal Akademik Mahasiswa.
- `public static final String LOGIN_URL`: menyimpan *url* halaman *login* Portal Akademik Mahasiswa.
- `public static final String SSO_URL`: menyimpan *url* halaman *login Single Sign On*(SSO) UNPAR.
- `public static final String JADWAL_URL`: menyimpan *url* halaman jadwal Portal Akademik Mahasiswa.
- `public static final String NILAI_URL`: menyimpan *url* halaman nilai Portal Akademik Mahasiswa.
- `public static final String TOEFL_URL`: menyimpan *url* halaman nilai TOEFL Portal Akademik Mahasiswa.
- `public static final String LOGOUT_URL`: menyimpan *url* untuk melakukan *logout*.
- `public static final String HOME_URL`: menyimpan *url* halaman utama Portal Akademik Mahasiswa setelah melakukan *login*.
- `public static final String PROFILE_URL`: menyimpan *url* halaman profil mahasiswa Portal Akademik Mahasiswa.
- `public static final String FRSPRS_URL`: menyimpan *url* halaman FRS/PRS Portal Akademik Mahasiswa.
- `public static final String[] MONTH_NAMES`: menyimpan nama-nama bulan dalam bahasa Indonesia.

Method yang dimiliki kelas ini adalah sebagai berikut:

- `public Scrapper()`
Berfungsi sebagai *constructor* kelas `Scrapper`.
- `public void init()`
Berfungsi untuk melakukan koneksi ke halaman utama Portal Akademik Mahasiswa.
- `public String login(String npm, String password)`
Berfungsi untuk melakukan *login* ke Portal Akademik Mahasiswa.

Parameter:

- `npm`: *npm* mahasiswa yang dipakai *login*.
- `password`: *password* mahasiswa yang dipakai *login*.

Kembalian: *session* yang dapat digunakan untuk mengakses menu di Portal Akademik Mahasiswa.

- `public TahunSemester requestNamePhotoTahunSemester(String phpsessid, Mahasiswa mahasiswa)`

Berfungsi untuk melakukan pengambilan nama, foto, serta semester yang sedang dijalani mahasiswa dari Portal Akademik Mahasiswa.

Parameter:

- `phpsessid`: *session* yang didapatkan dari proses *login* ke Portal Akademik Mahasiswa.
- `mahasiswa`: objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.

Kembalian: semester yang sedang dijalani mahasiswa.

- `public void requestNilai(String phpsessid, Mahasiswa mahasiswa)`

Berfungsi untuk melakukan pengambilan seluruh nilai mata kuliah mahasiswa dari Portal Akademik Mahasiswa.

Parameter:

- `phpsessid`: *session* yang didapatkan dari proses *login* ke Portal Akademik Mahasiswa.

- **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- **public void requestNilaiTOEFL(String phpsessid, Mahasiswa mahasiswa)**
Berfungsi untuk melakukan pengambilan seluruh riwayat nilai TOEFL mahasiswa dari Portal Akademik Mahasiswa.

Parameter:

- **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Mahasiswa.
- **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- **public void requestTanggalLahir(String phpsessid, Mahasiswa mahasiswa)**
Berfungsi untuk melakukan pengambilan data tanggal lahir mahasiswa dari Portal Akademik Mahasiswa.

Parameter:

- **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Mahasiswa.
- **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- **public void logout()**
Berfungsi untuk melakukan *logout* dari Portal Akademik Mahasiswa.
- **public String[] parseSemester(String sem_raw)**
Berfungsi untuk melakukan *parsing* data semester mahasiswa dari Portal Akademik Mahasiswa agar dapat diolah lebih lanjut.

Parameter:

- **sem_raw:** data semester yang didapat dari Portal Akademik Mahasiswa.

Kembalian: *array* yang berisi semester yang sudah dilakukan *parsing*.

- **public Mahasiswa[] requestMahasiswaList(String phpsessid, Mahasiswa mahasiswa)**
Berfungsi untuk mengambil seluruh mahasiswa dengan data yang diambil berupa nama, foto, semester yang sedang dijalani, dimana seluruh mahasiswa tersebut memiliki dosen wali yang sama dengan yang melakukan *login* (dalam implementasi menggunakan Portal Akademik Mahasiswa, mahasiswa yang diambil adalah hanya mahasiswa yang melakukan *login*).

Parameter:

- **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Mahasiswa.
- **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- **public Mahasiswa requestMahasiswaDetail(String phpsessid, Mahasiswa mahasiswa)**
Berfungsi untuk mengambil detil lebih lanjut mengenai data mahasiswa berupa nilai TOEFL, seluruh nilai mata kuliah, tanggal lahir mahasiswa.

Parameter:

- **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Mahasiswa.
- **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.

Kembalian: objek mahasiswa yang telah ditambahkan datanya dari pengambilan data

di Portal Akademik Mahasiswa.

3. MultipleRequest

Kelas ini mengimplementasikan kelas `interface Runnable` milik Java. Kelas ini memiliki fungsi untuk melakukan koneksi ke setiap semester yang telah ditempuh mahasiswa pada halaman nilai di Portal Akademik Mahasiswa (penjelasan lebih lanjut di [3.3.1](#)). Atribut yang dimiliki kelas ini adalah sebagai berikut:

- `private int index`: menyimpan indeks semester yang akan digunakan untuk mengakses `listSemester`.
- `private ArrayList<String> listSemester`: menyimpan daftar semester yang telah ditempuh mahasiswa.
- `private String NILAI_URL`: menyimpan *url* halaman nilai Portal Akademik Mahasiswa.
- `private String phpsessid`: menyimpan *session* yang dapat digunakan untuk mengakses menu di Portal Akademik Mahasiswa.
- `private Mahasiswa mahasiswa`: menyimpan objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- `private ScriptEngineManager factory`: untuk menjalankan *javascript*.
- `private ScriptEngine engine`: untuk menjalankan *javascript*.

Method yang dimiliki kelas ini adalah sebagai berikut:

- `public MultipleRequest(int index, ArrayList<String> listSemester, String NILAI_URL, String phpsessid, Mahasiswa mahasiswa)`
Berfungsi sebagai *constructor* kelas `MultipleRequest`.

Parameter:

- `index`: indeks semester.
- `listSemester`: *list* semester yang ditempuh mahasiswa.
- `NILAI_URL`: *url* halaman nilai pada Portal Akademik Mahasiswa.
- `phpsessid`: *session* yang didapatkan dari proses login ke Portal Akademik Mahasiswa.
- `mahasiswa`: objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.

- `public void run()`

Method ini merupakan *method* turunan dari kelas `interface Runnable`. Untuk mendapatkan data nilai dilakukan dengan cara:

- (a) Mendapatkan tahun dan semester yang ditempuh mahasiswa dari atribut `ArrayList<String> listSemester` diambil dari value atribut `int index`. Kemudian String dibagi menjadi tahun dan semester yang dibutuhkan.
- (b) Setelah mendapatkan tahun dan semester. Kemudian melakukan koneksi ke *url* nilai berdasarkan tahun dan semester (sebagai contoh dalam gambar [4.4](#), *url* yang digunakan yaitu <https://studentportal.unpar.ac.id/nilai/2017/1>).
- (c) Setelah berhasil, kemudian melakukan kueri css berdasarkan script yang mengandung nilai mahasiswa (Gambar [4.4](#)).
- (d) Selanjutnya adalah mendapatkan script yang mengandung script “`var data_mata_kuliah = [];` sampai indeks dari “`var data_angket = [];`”.
- (e) Setelah mendapatkan script yang dibutuhkan, selanjutnya menjalankan script menggunakan *method* milik kelas `ScriptEngine` yaitu `Object eval(String script)`.
- (f) Setelah berhasil, data yang didapatkan bertipe `ScriptObjectMirror` yang membungkus hasil eksekusi. Data nilai didapatkan dengan menggunakan *method* `Object get(Object key)`.
- (g) Setelah berhasil, kemudian memasukkan data nilai ke daftar riwayat nilai mahasiswa pada atribut kelas `Mahasiswa` yaitu `List<Nilai> riwayatNilai` menggunakan *method* `List<Nilai> getRiwayatNilai()`. Proses ini dilakukan berulang kali se-

banyak jumlah mata kuliah per semesternya.

No	Kode Mata Kuliah	Nama Mata Kuliah	SKS	Kelas	Nilai	AA	NA	Hasil Nilai
1	AIF131101	Pemrograman Berorientasi Objek	6	-	Tampilkan Detail Nilai	-	A	Tampilkan Grafik Nilai
2	AIF181107	Matematika Diskret	3	-	Tampilkan Detail Nilai	-	A	Tampilkan Grafik Nilai
3	AIF131105	Pengantar Informatika	3	-	Tampilkan Detail Nilai	-	A	Tampilkan Grafik Nilai
4	MKUI30001	Pendidikan Pancasila	2	-	Tampilkan Detail Nilai	-	A	Tampilkan Grafik Nilai
5	MKUI30008	Etika	2	-	Tampilkan Detail Nilai	-	B	Tampilkan Grafik Nilai
6	MKUI30010	Bahasa Inggris	2	-	Tampilkan Detail Nilai	-	A	Tampilkan Grafik Nilai

```

var data_mata_kuliah = [];data_mata_kuliah[0] = {};"data_mata_kuliah['nama_mata_kuliah'] = 'Pemrograman Berorientasi Objek';"data_mata_kuliah[0]['kode_mata_kuliah'] = 'AIF131101';"data_mata_kuliah[0]['jumlah_sks'] = '6';"data_mata_kuliah[0]['kelas'] = '-';"data_mata_kuliah[0]['nilai'] = [];"data_mata_kuliah[0]['aa'] = '';"data_mata_kuliah[0]['na'] = 'A';"data_mata_kuliah[1] = {};"data_mata_kuliah[1]['nama_mata_kuliah'] = 'Matematika Diskret';"data_mata_kuliah[1]['kode_mata_kuliah'] = 'AIF181107';"data_mata_kuliah[1]['jumlah_sks'] = '3';"data_mata_kuliah[1]['kelas'] = '-';"data_mata_kuliah[1]['nilai'] = [];"data_mata_kuliah[1]['aa'] = '';"data_mata_kuliah[1]['na'] = 'A';"data_mata_kuliah[2] = {};"data_mata_kuliah[2]['nama_mata_kuliah'] = 'Pengantar Informatika';"data_mata_kuliah[2]['kode_mata_kuliah'] = 'AIF131105';"data_mata_kuliah[2]['jumlah_sks'] = '3';"data_mata_kuliah[2]['kelas'] = '-';"data_mata_kuliah[2]['nilai'] = [];"data_mata_kuliah[2]['aa'] = '';"data_mata_kuliah[2]['na'] = 'A';"data_mata_kuliah[3] = {};"data_mata_kuliah[3]['nama_mata_kuliah'] = 'Pendidikan Pancasila';"data_mata_kuliah[3]['kode_mata_kuliah'] = 'MKUI30001';"data_mata_kuliah[3]['jumlah_sks'] = '2';"data_mata_kuliah[3]['kelas'] = '-';"data_mata_kuliah[3]['nilai'] = [];"data_mata_kuliah[3]['aa'] = '';"data_mata_kuliah[3]['na'] = 'A';"data_mata_kuliah[4] = {};"data_mata_kuliah[4]['nama_mata_kuliah'] = 'Etika';"data_mata_kuliah[4]['kode_mata_kuliah'] = 'MKUI30008';"data_mata_kuliah[4]['jumlah_sks'] = '2';"data_mata_kuliah[4]['kelas'] = '-';"data_mata_kuliah[4]['nilai'] = [];"data_mata_kuliah[4]['aa'] = '';"data_mata_kuliah[4]['na'] = 'B';"data_mata_kuliah[5] = {};"data_mata_kuliah[5]['nama_mata_kuliah'] = 'Bahasa Inggris';"data_mata_kuliah[5]['kode_mata_kuliah'] = 'MKUI30010';"data_mata_kuliah[5]['jumlah_sks'] =

```

Gambar 4.4: Script Data Nilai Mahasiswa Pada Halaman Nilai

Penjelasan mengenai kelas, *method*, dan atribut pada package *SIAKAD* adalah sebagai berikut:

1. SIAkadDataPuller

Kelas ini merupakan turunan dari kelas *DataPuller* 3. Kelas ini memiliki fungsi untuk mengambil username dan *password* dosen yang disimpan dalam sebuah *file*, dan kemudian memanggil *method* pada kelas *SIAkad*. Atribut yang dimiliki kelas ini adalah sebagai berikut:

- **private final SIAkad siakad:** menyimpan objek *SIAkad*.

Method yang dimiliki kelas ini adalah sebagai berikut:

- **public SIAkadDataPuller()**

Berfungsi sebagai *constructor* kelas *SIAkadDataPuller*.

- **public Mahasiswa[] pullMahasiswas()**

Berfungsi untuk memanggil *method* *requestMahasiswaList* pada kelas *SIAkad*.

Kembalian: *array* yang berisi daftar mahasiswa yang diwaliakan oleh dosen terlogin.

- **public Mahasiswa pullMahasiswaDetail()**

Berfungsi untuk memanggil *method* *requestRiwayatNilai*, dan *requestDataDiri* pada kelas *SIAkad*.

Kembalian: objek mahasiswa yang telah ditambahkan datanya dari pengambilan data di *SIAKAD*.

2. SIAkad

Kelas ini memiliki fungsi untuk melakukan pengambilan data dari SIAKAD untuk kemudian diolah dan ditampilkan. Atribut yang dimiliki kelas ini adalah sebagai berikut:

- `protected static final String SIAKAD_BASE_URL`: menyimpan *url* utama SIAKAD.
- `protected static final String SSO_BASE_URL`: menyimpan *url* halaman *login Single Sign On*(SSO) UNPAR.
- `protected static final String MU_BASE_URL`: menyimpan *url* MU UNPAR yang digunakan untuk memeriksa apakah ada pengguna dengan id tertentu.
- `protected static final int DEFAULT_TIMEOUT`: waktu *timeout*.
- `protected final Logger logger`: untuk menulis log.
- `public static final String[] MONTH_NAMES`: menyimpan nama-nama bulan dalam bahasa Indonesia.
- `Token token`: menyimpan *cookies* yang dapat digunakan untuk mengakses menu di SIAKAD.

Method yang dimiliki kelas ini adalah sebagai berikut:

- `public SIAkad()`
Berfungsi sebagai *constructor* kelas SIAkad.
- `public SIAkad(boolean verbose)`
Berfungsi sebagai *constructor* kelas SIAkad.

Parameter:

- `verbose`: apabila bernilai `true`, akan menampilkan informasi-informasi yang detil.
Apabila bernilai `false`, tidak akan menampilkan informasi-informasi yang detil.

- `public boolean login(String username, String password)`

Berfungsi untuk melakukan login ke SI Akademik. Semua transaksi lain harus diawali dengan login.

Parameter:

- `username`: username dosen.
- `password`: password dosen.

Kembalian: *true* jika *login* berhasil, *false* jika *username/password* salah.

- `public void logout()`

Berfungsi untuk melakukan *logout* dari SIAKAD.

- `public List<Mahasiswa> requestMahasiswaList()`

Berfungsi untuk mendapatkan daftar mahasiswa yang diwalikan oleh dosen terlogin.

Kembalian: daftar mahasiswa yang diwalikan oleh dosen terlogin.

- `public List<Mahasiswa> requestMahasiswaList(Set<ProgramStudi>`

`programStudi, Mahasiswa.Status status, Integer angkatan)`

Berfungsi untuk mendapatkan daftar mahasiswa yang diwalikan oleh dosen terlogin.

Parameter:

- `programStudi`: program studi yang dipilih, atau null jika ingin mencari dari seluruh program studi yang terdaftar di SIAModels.
- `status`: status mahasiswa yang ingin ditampilkan, atau null jika default (aktif).
- `angkatan`: tahun angkatan, atau null jika semua.

Kembalian: daftar mahasiswa yang diwalikan oleh dosen terlogin.

- `public Mahasiswa requestDataDiri(Mahasiswa mahasiswa)`

Berfungsi untuk mendapatkan data diri mahasiswa. Saat ini hanya mendukung *URL* foto, tanggal lahir, dan jenis kelamin tanpa data diri yang lain.

Parameter:

- `mahasiswa`: mahasiswa yang ingin diperiksa.

Kembalian: objek mahasiswa yang sama, dengan data diri yang sudah dilengkapi.

- `public Mahasiswa requestRiwayatNilai(Mahasiswa mahasiswa, boolean includeLastSemester)`

Berfungsi untuk mendapatkan daftar lengkap riwayat nilai mahasiswa, berdasarkan

halaman "Data Akademik". Metode ini dapat mengisi lengkap kelas, nilai-nilai Tugas, UTS, UAS, serta NA.

Parameter:

- `mahasiswa`: mahasiswa yang ingin diperiksa.
- `includeLastSemester`: apakah ingin mengikutsertakan nilai semester terakhir yang tercatat. Kelemahan metode "Data Akademik" adalah tidak bisa membedakan antara nilai yang belum rilis dengan yang sudah. Jika mahasiswa sedang menjalani mata kuliah tersebut, nilai sudah muncul tetapi kemungkinan NA nya berisi E karena nilai tugas/UTS/UAS belum lengkap. Jika mahasiswa sedang tidak menjalani kuliah tersebut (misal, saat FRS atau saat libur, maka nilai semester terakhir perlu diikutsertakan, karena mengacu ke nilai yang sudah rilis). Tentu saja ke depannya perlu mekanisme yang lebih baik yang dapat mendeteksi nilai yang sudah/belum rilis ini.

Kembalian: objek mahasiswa yang sama, dengan nilai yang sudah didapatkan (terurut secara kronologis dari yang paling lama ke baru).

- `protected Connection createBaseConnection(String url, Token token)`
Berfungsi untuk membuat koneksi, dengan kondisi sudah melakukan *login*.

Parameter:

- `url`: *URL* yang dituju.
- `token`: token autentikasi.

Kembalian: koneksi tersebut.

- `protected void logConnection(Connection connection)`
Berfungsi untuk memeriksa *verbose flag*, dan apabila *true*, akan menampilkan *debug output*.

Parameter:

- `connection`: *connection request* dan *response*.

3. Token

Kelas ini memiliki fungsi untuk menyimpan *cookies*. Atribut yang dimiliki kelas ini adalah sebagai berikut:

- `private final Map<String, String> cookies`: menyimpan *cookies*.

Method yang dimiliki kelas ini adalah sebagai berikut:

- `public Token(Map<String, String> cookies)`
Berfungsi sebagai *constructor* kelas *Token*.
 - `public Map<String, String> getCookies()`
Berfungsi untuk mendapatkan *cookies*.
- Kembalian:** *cookies*.
- `public void injectToConnection(Connection connection)` Berfungsi untuk memasukkan *session* ke dalam *connection*.

Parameter:

- `connection`: koneksi yang ingin dimasukkan *session*.

4.2 Perancangan Antarmuka

Gambar 4.5 menunjukkan rancangan antarmuka dari aplikasi *screensaver*. Antarmuka aplikasi akan menampilkan foto dari mahasiswa apabila foto tersebut tersedia. Baris pertama merupakan nama dari mahasiswa, kemudian baris kedua akan diisi oleh angkatan dari mahasiswa tersebut. Baris ketiga menampilkan usia dan tanggal lahir mahasiswa. Baris keempat merupakan status dari mahasiswa tersebut apabila tersedia. Baris kelima merupakan *email* mahasiswa dimana format *email* mahasiswa UNPAR adalah `NomorPokokMahasiswa@student.unpar.ac.id`. Baris keenam berisi nilai TOEFL terakhir mahasiswa apabila mahasiswa tersebut sudah mengambil ujian TOEFL, atau diisi "Tidak Tersedia" apabila mahasiswa tersebut belum mengambil ujian TOEFL.

Baris ketujuh menampilkan IPS dan IPK dari mahasiswa tersebut. Baris terakhir diisi oleh jumlah sks yang telah lulus dan jumlah sks yang telah ditempuh oleh mahasiswa tersebut. Setiap 5 detik tampilan aplikasi akan berubah ke mahasiswa berikutnya. Namun pada aplikasi yang dibuat oleh pembimbing, data yang ditampilkan hanya mahasiswa itu sendiri, sedangkan pada aplikasi yang dibuat oleh pembimbing, data yang ditampilkan akan berubah.



Gambar 4.5: Rancangan Antarmuka *Screensaver*

BAB 5

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dijelaskan mengenai implementasi perangkat lunak, serta pengujian perangkat lunak. Implementasi perangkat lunak berisi penjelasan lingkungan pengembangan perangkat lunak dan hasil implementasi. Sedangkan pengujian perangkat lunak berisi hasil pengujian fungsional dan eksperimental terhadap perangkat lunak yang telah dibangun.

5.1 Implementasi

5.1.1 Lingkungan Implementasi

Implementasi perangkat lunak ini dilakukan di komputer penulis dengan spesifikasi berikut:

1. *Processor*: Intel Core i5-10300H
2. *Random Access Memory(RAM)*: 16 GB DDR4
3. Sistem Operasi: Windows 10
4. Versi Java: 1.8.0_291
5. Versi JavaFX: 8.0.291
6. Versi Netbeans: 12.1
7. Versi Scenebuilder: 11.0.0

5.1.2 Hasil Implementasi

Hasil implementasi berupa sebuah *file* yang berekstensi **jar**. Agar dapat digunakan sebagai *screensaver*, *file* tersebut perlu diubah ekstensinya menjadi **scr**. Untuk mengubah ekstensi menjadi **scr**, perlu dilakukan perubahan menjadi ekstensi **exe** terlebih dahulu. Perubahan ekstensi menjadi **exe** tersebut dilakukan dengan bantuan aplikasi bernama Launch4j¹. Setelah didapatkan *file* berekstensi **exe**, maka untuk mengubahnya menjadi berekstensi **scr** cukup dengan melakukan rename pada file tersebut menjadi berekstensi **scr**. File berekstensi **scr** tersebut dapat disimpan pada direktori “System32” didalam folder “Windows” agar dapat terdeteksi sebagai *screensaver* oleh Windows. Setelah mengatur aplikasi menjadi *screensaver*, aplikasi tersebut akan dijalankan secara otomatis setelah komputer tidak digunakan selama beberapa saat. Gambar 5.1 dan 5.2 merupakan tampilan dari aplikasi *screensaver*. Dikarenakan pada Portal Akademik Mahasiswa tidak terdapat foto mahasiswa, maka foto digantikan dengan *base64 image* yang dimasukkan secara *hardcode*. *Layout* dari aplikasi disimpan dalam *file* bertipe FXML (lampiran E). *File* FXML tersebut tidak dibuat secara manual, melainkan dengan menggunakan aplikasi Scene Builder².

¹<http://launch4j.sourceforge.net>

²<https://gluonhq.com/products/scene-builder/>

HARRY SENJAYA DARMAWAN



Mahasiswa angkatan 2017

Usia: 22 tahun 0 bulan (lahir 1999-05-04)

Status: Tidak Tersedia

Email: 2017730067@student.unpar.ac.id

Nilai TOEFL: 540

IPS/IPK: 3.74/3.58

SKS Lulus/Tempuh: 134/134

Gambar 5.1: Tampilan *Screensaver* Mahasiswa Pertama

DUMMY DATA

Mahasiswa angkatan 2017

Usia: 22 tahun 5 bulan (lahir 1999-01-01)

Status: Tidak Tersedia

Email: 2017730001@student.unpar.ac.id

Nilai TOEFL: 540

IPS/IPK: 3.74/3.58

SKS Lulus/Tempuh: 134/134

Gambar 5.2: Tampilan *Screensaver* Mahasiswa Kedua

5.2 Pengujian

5.2.1 Pengujian Fungsional

Pengujian fungsional dilakukan untuk mengetahui kesesuaian reaksi perangkat lunak dengan reaksi yang diharapkan berdasarkan aksi pengguna terhadap perangkat lunak. Tabel 5.1 merupakan hasil pengujian perangkat lunak yang dilakukan di komputer penulis dengan spesifikasi berikut:

1. Processor: Intel Core i5-10300H
2. Random Access Memory(RAM): 16 GB DDR4

3. Sistem Operasi: Windows 10
4. Resolusi Layar: 1920 x 1080
5. Versi Java: 1.8.0_291

Tabel 5.1: Tabel Pengujian Fungsional

No.	Aksi Pengguna	Reaksi yang diharapkan	Reaksi Perangkat Lunak
1.	Dosen tidak menggunakan komputer selama beberapa saat.	<i>Screensaver</i> berjalan dengan menampilkan biodata dan data akademik mahasiswa wali secara umum.	sesuai
2.	Mahasiswa tidak menggunakan komputer selama beberapa saat.	<i>Screensaver</i> berjalan dengan menampilkan biodata dan data akademik mahasiswa secara umum beserta dengan data dummy.	sesuai

5.2.2 Pengujian Eksperimental

Subbab ini ditulis oleh dosen pembimbing.

Pengujian eksperimental dilakukan oleh dosen pembimbing, karena mahasiswa tidak memiliki akses ke SIAKAD apalagi ke data mahasiswa di luar penulis sendiri. Pengujian eksperimental dilakukan dengan melakukan *git branching* dari hasil (hampir) final dari penulis di commit d60be65. Pengujian eksperimental dilakukan pada komputer dosen dengan spesifikasi seperti berikut³:

1. Komputer: iMac (21.5-inch, Late 2013)
2. Sistem Operasi: macOS Catalina 10.15.7
3. Processor: 2,7 GHz Quad-Core Intel Core i5
4. Memory: 8 GB 1600 MHz DDR3
5. Graphics: Intel Iris Pro 1536 MB
6. Resolusi Layar: 1920 x 1080
7. Versi Java: 1.8.0_282

Ada beberapa penyesuaian yang dilakukan oleh dosen pembimbing supaya *screensaver* bisa dijalankan dengan cukup baik pada lingkungan dosen. Karena keterbatasan waktu, penyesuaian ini dilakukan tanpa berkoordinasi secara intensif dengan mahasiswa. Penyesuaian-penesuaian tersebut antara lain:

1. **Data diambil dari SIAKAD bukan dari Portal Akademik Mahasiswa.** Daftar mahasiswa yang diambil diacak urutannya sehingga saat ditampilkan tidak selalu dimulai dari angkatan tertua.
2. **Upgrade versi SIAModels dari 4.0.0 menjadi 5.0.1.** Pada versi terbaru SIAModels, ada beberapa perbaikan, di mana salah satunya adalah *bug* saat menentukan angkatan mahasiswa berdasarkan NPM nya. Perbaikan lainnya adalah pada perhitungan IPK dan IPS. SIAModels merupakan pustaka eksternal dan bukan merupakan bagian inti dari *screensaver* yang dibuat.
3. **Load data di *thread* terpisah.** Pada program yang dibuat oleh mahasiswa, program hanya mengunduh satu data dari Portal Akademik Mahasiswa, yaitu data yang bersangkutan sendiri. Pada SIAKAD, diunduh banyak data dan ternyata menimbulkan masalah saat mengunduh data mahasiswa untuk ditampilkan pada slide berikutnya. Masalah muncul karena saat load data mahasiswa berikutnya, program menjadi tidak responsif. Oleh karena itu, saat menampilkan sebuah slide, program diubah supaya mengunduh data mahasiswa untuk slide

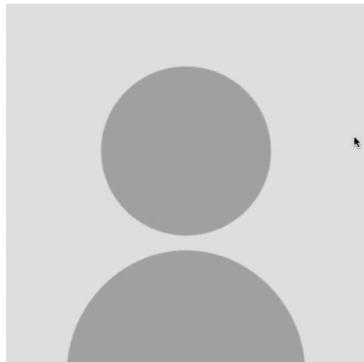
³Idealnya pengujian dilakukan di komputer UNPAR dengan sistem operasi Microsoft Windows sehingga bisa benar-benar digunakan sebagai *screensaver*. Namun, karena keadaan pandemi sehingga dosen tidak bisa mengunjungi UNPAR dan hanya bisa melakukan pengujian di rumah dengan komputer macOS

berikutnya di latar belakang. Saat slide berikutnya ditampilkan, sudah tidak perlu melakukan pengunduhan kembali karena datanya sudah siap.

4. **Saat mengulang ke mahasiswa pertama, tidak dilakukan load data kembali.**
5. **Jika data mahasiswa belum siap ditampilkan, tidak maju ke slide berikutnya.**
6. **Bugfix untuk pembacaan bulan Februari.** Pada SIAKAD, ternyata bulan kedua dituliskan sebagai “Pebruari” dengan huruf “P”, karena itu perlu penyesuaian sehingga mahasiswa dengan bulan lahir Februari dapat dibaca dengan baik.

Perubahan lengkap kode dibandingkan dengan kode milik mahasiswa dapat dilihat pada lampiran D.1. Setelah perbaikan-perbaikan tersebut dilakukan, program dapat dijalankan dengan baik, dan untuk menampilkan 30 mahasiswa “wali” dari dosen pembimbing, dibutuhkan waktu sekitar 8 menit, di mana setelahnya mengulang kembali dari mahasiswa pertama. Beberapa hasil tangkapan layar dapat dilihat pada gambar 5.3, 5.4, 5.5, 5.6, dan 5.7. Dosen sudah meminta izin kepada kelima mahasiswa tersebut untuk datanya dapat ditampilkan pada laporan ini, melalui e-mail.

GHARLAN WINARNO



Mahasiswa angkatan 2019

Usia: 20 tahun 2 bulan (lahir 2001-04-03)

Status: Tidak Tersedia

Email: 6181901044@student.unpar.ac.id

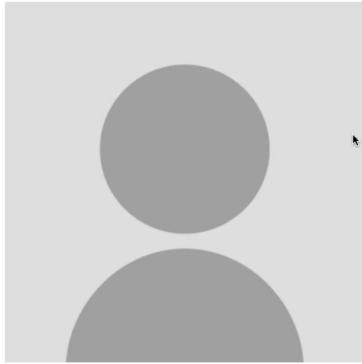
Nilai TOEFL: Tidak Tersedia

IPS/IPK: 2.9/2.48

SKS Lulus/Tempuh: 42/49

Gambar 5.3: Tampilan *Screensaver* Mahasiswa Pertama

YEREMY FERELL HIDAYAT



Mahasiswa angkatan 2019

Usia: 20 tahun 5 bulan (lahir 2001-01-14)

Status: Tidak Tersedia

Email: 6181901011@student.unpar.ac.id

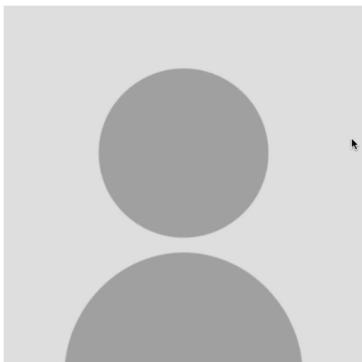
Nilai TOEFL: Tidak Tersedia

IPS/IPK: 2.2/2.36

SKS Lulus/Tempuh: 51/54

Gambar 5.4: Tampilan *Screensaver* Mahasiswa Kedua

WILSON NATHANAEEL



Mahasiswa angkatan 2019

Usia: 20 tahun 5 bulan (lahir 2000-12-22)

Status: Tidak Tersedia

Email: 6181901053@student.unpar.ac.id

Nilai TOEFL: Tidak Tersedia

IPS/IPK: 3.64/3.48

SKS Lulus/Tempuh: 60/64

Gambar 5.5: Tampilan *Screensaver* Mahasiswa Ketiga



JONATHAN LAKSAMANA PURNOMO

Mahasiswa angkatan 2016

Usia: 24 tahun 6 bulan (lahir 1996-12-11)

Status: Tidak Tersedia

Email: 7316081@student.unpar.ac.id

Nilai TOEFL: Tidak Tersedia

IPS/IPK: 2.28/2.21

SKS Lulus/Tempuh: 128/146

Gambar 5.6: Tampilan *Screensaver* Mahasiswa Keempat



NICHOLAS ADITYA HALIM

Mahasiswa angkatan 2017

Usia: 21 tahun 10 bulan (lahir 1999-07-19)

Status: Tidak Tersedia

Email: 2017730018@student.unpar.ac.id

Nilai TOEFL: Tidak Tersedia

IPS/IPK: 3.63/3.46

SKS Lulus/Tempuh: 134/134

Gambar 5.7: Tampilan *Screensaver* Mahasiswa Kelima

BAB 6

KESIMPULAN DAN SARAN

6.1 Kesimpulan

Dari hasil pembangunan aplikasi *screensaver*, didapatkan kesimpulan-kesimpulan sebagai berikut:

1. Aplikasi *screensaver* telah dapat mengambil data-data yang dibutuhkan untuk ditampilkan dengan baik. Namun *screensaver* dengan metode *scraping* ini memiliki kelemahan yaitu jika terdapat perubahan struktur dan penyedia layanan data berhenti atau menghilangkan data yang dibutuhkan, maka data tersebut tidak dapat ditampilkan.
2. Aplikasi *screensaver* telah dapat dijalankan pada seluruh perangkat dengan sistem operasi windows.

6.2 Saran

Dari hasil penelitian termasuk kesimpulan yang didapat, berikut adalah beberapa saran untuk pengembangan lebih lanjut:

1. Aplikasi *screensaver* sebaiknya mengganti metode pengambilan data yang sebelumnya menggunakan metode *scraping* diganti dengan metode lain, sehingga jika terjadi perubahan struktur, atau penyedia layanan data berhenti, atau menghilangkan data yang dibutuhkan, maka masih dapat menampilkan data yang sesuai dengan kebutuhan aplikasi.
2. Tampilan aplikasi *screensaver* sebaiknya dibuat secara *responsive*, sehingga aplikasi *screensaver* dapat dijalankan dengan baik di berbagai resolusi layar.
3. Pengambilan data mahasiswa pada aplikasi *screensaver* sebaiknya dijalankan secara paralel, sehingga ketika tampilan berubah ke mahasiswa selanjutnya tidak perlu menunggu pengambilan data mahasiswa tersebut terlebih dahulu.

DAFTAR REFERENSI

- [1] Alfadian, P. (2016) SI Akademik. <http://bt1.unpar.ac.id/akademik/>. 19 Oktober 2020.
- [2] Wikipedia (2021) Screensaver. <https://en.wikipedia.org/wiki/Screensaver>. 07 Maret 2021.
- [3] 2018, T. P. P. A. M. P. (2018) BUKU PANDUAN PENGGUNAAN PORTAL AKADEMIK MAHASISWA (PAM) 2018. https://studentportal.unpar.ac.id/assets/BUKU_PANDUAN_PENGGUNAAN_FRS_GABUNGAN.pdf. 28 Februari 2021.
- [4] Alfadian, P. (2020) SIAModels. <https://github.com/pascalalfadian/SIAModels>. 19 Oktober 2020.
- [5] Sugiarto, A. (2018) PENYESUAIAN SIAMODELS DAN IFSTUDENTPORTAL KE KURIKULUM 2018. Skripsi. Universitas Katolik Parahyangan, Indonesia.
- [6] Version 1.13.1 (2009-2020) *jsoup: Java HTML Parser*. Jonathan Hedley. Seattle, Washington.
- [7] Version 15 (2008-2020) *JavaFX 15*. Gluon HQ. Industrieweg 3, Leuven, BE 3001, BE.
- [8] Version 8.0 (2008-2015) *Introduction to FXML*. Oracle. 2300 Oracle Way Austin, TX 78741.

LAMPIRAN A

KODE PROGRAM *SCREENSAVER*

Kode A.1: App.java

```
1 package id.ac.unpar.screensaver;
2
3 import javafx.application.Application;
4 import javafx.fxml.FXMLLoader;
5 import javafx.scene.Parent;
6 import javafx.scene.Scene;
7 import javafx.stage.Stage;
8
9 import java.io.IOException;
10
11 public class App extends Application {
12
13     private static Scene scene;
14
15     @Override
16     public void start(Stage stage) throws IOException {
17         scene = new Scene(loadFXML("primary"));
18         stage.setScene(scene);
19         stage.setFullScreen(true);
20         stage.show();
21     }
22
23     private static Parent loadFXML(String fxml) throws IOException {
24         FXMLLoader fxmlLoader = new FXMLLoader(App.class.getResource(fxml + ".fxml"));
25         return fxmlLoader.load();
26     }
27
28     public static void main(String[] args) {
29         launch();
30     }
31 }
```

Kode A.2: DataPuller.java

```
1 package id.ac.unpar.screensaver;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4
5 public abstract class DataPuller {
6
7     public abstract Mahasiswa[] pullMahasiswas();
8     public abstract Mahasiswa pullMahasiswaDetail(Mahasiswa m);
9 }
```

Kode A.3: PrimaryController.java

```
1 package id.ac.unpar.screensaver;
2
3 //import id.ac.unpar.screensaver.siaakad.SIAkadDataPuller;
4 import id.ac.unpar.screensaver.studentportal.StudentPortalDataPuller;
5 import id.ac.unpar.siamodels.Mahasiswa;
6 import id.ac.unpar.siamodels.TahunSemester;
7 import java.io.ByteArrayInputStream;
8 import javafx.scene.image.Image;
9 import java.io.IOException;
10 import java.net.URL;
11 import java.time.LocalDate;
12 import java.time.Period;
13 import java.util.ResourceBundle;
14 import java.util.logging.Level;
15 import java.util.logging.Logger;
16 import javafx.animation.Animation;
17 import javafx.animation.KeyFrame;
18 import javafx.animation.Timeline;
19 import javafx.fxml.FXML;
20 import javafx.fxml.Initializable;
21 import javafx.scene.image.ImageView;
22 import javafx.scene.text.Text;
23 import javafx.util.Duration;
24
25 public class PrimaryController implements Initializable {
```

```

26
27     @FXML
28     private Text nama, angkatan, usia, status, email, toefl, ipk, sks;
29
30     @FXML
31     private ImageView foto;
32
33     private int indexOfMahasiswa = 0;
34     private Mahasiswa[] listMahasiswa;
35     private DataPuller puller;
36
37     public int getIndexOfMahasiswa() {
38         return indexOfMahasiswa;
39     }
40
41     public void setIndexOfMahasiswa(int indexOfMahasiswa) {
42         this.indexOfMahasiswa = indexOfMahasiswa;
43     }
44
45     public PrimaryController() throws IOException {
46
47     }
48
49     @Override
50     public void initialize(URL url, ResourceBundle rb) {
51
52         try {
53             puller = new StudentPortalDataPuller();
54             listMahasiswa = puller.pullMahasiswa();
55         } catch (IllegalStateException ex) {
56             Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
57         }
58         if (listMahasiswa != null) {
59             try {
60                 listMahasiswa[this.getIndexOfMahasiswa()] = puller.pullMahasiswaDetail(listMahasiswa[this.getIndexOfMahasiswa()]);
61                 this.updateView(listMahasiswa[this.getIndexOfMahasiswa()]);
62                 this.setIndexOfMahasiswa(this.getIndexOfMahasiswa() + 1);
63             } catch (IOException ex) {
64                 Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
65             }
66         } else {
67             updateView();
68         }
69         Timeline timeline = new Timeline(
70             new KeyFrame(
71                 Duration.seconds(5),
72                 event -> {
73                     if (listMahasiswa == null) {
74                         updateView();
75                     try {
76                         puller = new StudentPortalDataPuller();
77                         listMahasiswa = puller.pullMahasiswa();
78                     } catch (IllegalStateException ex) {
79                         Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
80                     }
81                 } else {
82                     if (this.getIndexOfMahasiswa() == listMahasiswa.length) {
83                         this.setIndexOfMahasiswa(0);
84                     } else {
85                         if (listMahasiswa[this.getIndexOfMahasiswa()].getTanggalLahir() == null) {
86                             listMahasiswa[this.getIndexOfMahasiswa()] = puller.pullMahasiswaDetail(listMahasiswa[this.getIndexOfMahasiswa()]);
87                         }
88                         if (listMahasiswa[this.getIndexOfMahasiswa()].getTanggalLahir() == null) {
89                             updateView();
90                         } else {
91                             try {
92                                 this.updateView(listMahasiswa[this.getIndexOfMahasiswa()]);
93                                 this.setIndexOfMahasiswa(this.getIndexOfMahasiswa() + 1);
94                             } catch (IOException ex) {
95                                 Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
96                             }
97                         }
98                     }
99                 }
100            )
101        );
102        timeline.setCycleCount(Animation.INDEFINITE);
103        timeline.play();
104    }
105
106 }
107
108     public void updateView() {
109         this.foto.setVisible(false);
110         this.nama.setText("Pastikan_koneksi_internet_berfungsi_dengan_normal!");
111         this.angkatan.setText("—");
112         this.usia.setText("—");
113         this.status.setText("—");
114         this.email.setText("—");
115         this.toefl.setText("—");
116         this.ipk.setText("—");
117         this.sks.setText("—");
118     }
119
120     public void updateView(Mahasiswa mahasiswa) throws IOException {
121         System.out.println("Updating_view_with_" + mahasiswa.getNama());
122         if (mahasiswa.getPhotoPath() != null) {
123             ByteArrayInputStream bais = new ByteArrayInputStream(mahasiswa.getPhotoImage());

```

```
124     Image image = new Image(bais);
125     this.foto.setImage(image);
126
127     this.foto.setVisible(true);
128 } else {
129     this.foto.setVisible(false);
130 }
131 this.nama.setText(mahasiswa.getNama());
132 this.angkatan.setText(mahasiswa.getTahunAngkatan() + "");
133 this.usia.setText(Period.between(mahasiswa.getTanggalLahir(), LocalDate.now()).getYears() + " tahun" + Period.between(
134     mahasiswa.getTanggalLahir(), LocalDate.now()).getMonths() + " bulan" + "(lahir" + mahasiswa.getTanggalLahir().toString() + ")");
135 if (mahasiswa.getStatus()!=null) {
136     this.status.setText(mahasiswa.getStatus().toString());
137 } else{
138     this.status.setText("Tidak Tersedia");
139 }
140 this.email.setText(mahasiswa.getEmailAddress());
141 if (mahasiswa.getNilaiTOEFL() != null && mahasiswa.getNilaiTOEFL().size() > 0) {
142     this.toefl.setText(mahasiswa.getNilaiTOEFL().get(mahasiswa.getNilaiTOEFL().firstKey()).toString());
143 } else {
144     this.toefl.setText("Tidak Tersedia");
145 }
146 TahunSemester tahunSemesterTerakhir = null;
147 for (Mahasiswa.Nilai nilai : mahasiswa.getRiwayatNilai()) {
148     if (tahunSemesterTerakhir == null || nilai.getTahunSemester().compareTo(tahunSemesterTerakhir) > 0) {
149         tahunSemesterTerakhir = nilai.getTahunSemester();
150     }
151 }
152 this.ipk.setText(Math.round(mahasiswa.calculateIPS(tahunSemesterTerakhir) * 100.0) / 100.0 + "/" + Math.round(mahasiswa.
153     calculateIPK() * 100.0) / 100.0);
154 this.sks.setText(+mahasiswa.calculateSKSLulus() + "/" + mahasiswa.calculateSKSTempuh(false));
155 }
```


LAMPIRAN B

KODE PROGRAM PORTAL AKADEMIK MAHASISWA (STUDENTPORTAL)

Kode B.1: StudentPortalDataPuller.java

```
1 package id.ac.unpar.screensaver.studentportal;
2
3 import id.ac.unpar.screensaver.DataPuller;
4 import id.ac.unpar.screensaver.PrimaryController;
5 import id.ac.unpar.siamodels.Mahasiswa;
6 import id.ac.unpar.siamodels.TahunSemester;
7 import java.io.FileReader;
8 import java.io.IOException;
9 import java.util.Properties;
10 import java.util.logging.Level;
11 import java.util.logging.Logger;
12
13 public class StudentPortalDataPuller extends DataPuller {
14
15     private Scraper scraper;
16
17     private final Properties credentials = new Properties();
18
19     private Mahasiswa mahasiswa;
20     private String session;
21
22     public StudentPortalDataPuller() {
23         try {
24             this.credentials.load(new FileReader("login.properties"));
25             String npm = credentials.getProperty("user.npm");
26             String password = credentials.getProperty("user.password");
27             this.mahasiswa = new Mahasiswa(npm);
28             this.scraper = new Scraper();
29             this.session = this.scraper.login(npm, password);
30         } catch (IOException ex) {
31             Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
32         }
33     }
34
35     @Override
36     public Mahasiswa[] pullMahasiswas() {
37         Mahasiswa[] mahasiswas = null;
38         try {
39             mahasiswas = this.scraper.requestMahasiswaList(this.session, this.mahasiswa);
40         } catch (IllegalStateException ex) {
41             Logger.getLogger(StudentPortalDataPuller.class.getName()).log(Level.SEVERE, null, ex);
42         } catch (IOException ex) {
43             Logger.getLogger(StudentPortalDataPuller.class.getName()).log(Level.SEVERE, null, ex);
44         } catch (InterruptedException ex) {
45             Logger.getLogger(StudentPortalDataPuller.class.getName()).log(Level.SEVERE, null, ex);
46         }
47         return mahasiswas;
48     }
49
50     @Override
51     public Mahasiswa pullMahasiswaDetail(Mahasiswa m) {
52         try {
53             this.scraper.requestMahasiswaDetail(this.session, m);
54         } catch (IOException ex) {
55             Logger.getLogger(StudentPortalDataPuller.class.getName()).log(Level.SEVERE, null, ex);
56         }
57         return m;
58     }
59 }
60 }
```

Kode B.2: Scraper.java

```
1 package id.ac.unpar.screensaver.studentportal;
2
3 import id.ac.unpar.siamodels.JenisKelamin;
4 import id.ac.unpar.siamodels.Mahasiswa;
5 import id.ac.unpar.siamodels.Mahasiswa.Nilai;
6 import id.ac.unpar.siamodels.Semester;
7 import id.ac.unpar.siamodels.TahunSemester;
```

```

8| import java.io.BufferedReader;
9| import java.io.FileInputStream;
10| import java.io.FileNotFoundException;
11| import java.io.FileReader;
12| import java.io.IOException;
13| import java.net.ProtocolException;
14| import java.time.LocalDate;
15| import java.time.Month;
16| import java.util.ArrayList;
17| import java.util.Arrays;
18| import java.util.Collections;
19| import java.util.Comparator;
20| import java.util.List;
21| import java.util.Map;
22| import java.util.Properties;
23| import java.util.SortedMap;
24| import java.util.StringTokenizer;
25| import java.util.TreeMap;
26| import java.util.logging.Level;
27| import java.util.logging.Logger;
28| import org.jsoup.Connection;
29| import org.jsoup.Connection.Response;
30| import org.jsoup.Jsoup;
31| import org.jsoup.nodes.Document;
32| import org.jsoup.nodes.Element;
33| import org.jsoup.select.Elements;
34|
35| public class Scraper {
36|
37|     public static final String BASE_URL = "https://studentportal.unpar.ac.id/";
38|     public static final String LOGIN_URL = BASE_URL + "C_home/sso_login";
39|     public static final String SSO_URL = "https://sso.unpar.ac.id/login";
40|     public static final String JADWAL_URL = BASE_URL + "jadwal";
41|     public static final String NILAI_URL = BASE_URL + "nilai";
42|     public static final String TOEFL_URL = BASE_URL + "nilai/toefl";
43|     public static final String LOGOUT_URL = BASE_URL + "logout";
44|     public static final String HOME_URL = BASE_URL + "home";
45|     public static final String PROFILE_URL = BASE_URL + "profil";
46|     public static final String FRSPRS_URL = "https://restu.unpar.ac.id/frsprs";
47|
48|     public static final String[] MONTH_NAMES = {
49|         "Januari", "Februari", "Maret", "April", "Mei", "Juni", "Juli", "Agustus", "September", "Oktober", "November", "Desember"
50|     };
51|
52|     public Scraper() throws FileNotFoundException, IOException {
53|     }
54|
55|     public void init() throws IOException {
56|         Connection baseConn = Jsoup.connect(BASE_URL);
57|         baseConn.timeout(0);
58|         baseConn.method(Connection.Method.GET);
59|         baseConn.execute();
60|     }
61|
62|
63|     public String login(String npm, String password) throws IOException {
64|         init();
65|         Mahasiswa logged_mhs = new Mahasiswa(npm);
66|         String user = logged_mhs.getEmailAddress();
67|         Connection conn = Jsoup.connect(LOGIN_URL);
68|         conn.data("Submit", "Login");
69|         conn.timeout(0);
70|         conn.method(Connection.Method.POST);
71|         Response resp = conn.execute();
72|         Document doc = resp.parse();
73|         String execution = doc.select("input[name=execution]").val();
74|         String jsessionid = resp.cookie("JSESSIONID");
75|         /* SSO LOGIN */
76|         Connection loginConn = Jsoup.connect(SSO_URL + ";jsessionid=" + jsessionid + "?service=" + LOGIN_URL);
77|         loginConn.cookies(resp.cookies());
78|         loginConn.data("username", user);
79|         loginConn.data("password", password);
80|         loginConn.data("execution", execution);
81|         loginConn.data("_eventId", "submit");
82|         loginConn.timeout(0);
83|         loginConn.method(Connection.Method.POST);
84|         resp = loginConn.execute();
85|         if (resp.body().contains(user)) {
86|             Map<String, String> phpsessid = resp.cookies();
87|             return phpsessid.get("ci_session");
88|         } else {
89|             return null;
90|         }
91|     }
92|
93|     public TahunSemester requestNamePhotoTahunSemester(String phpsessid, Mahasiswa mahasiswa) throws IOException {
94|         Connection connection = Jsoup.connect(HOME_URL);
95|         connection.cookie("ci_session", phpsessid);
96|         connection.timeout(0);
97|         connection.method(Connection.Method.GET);
98|         Response resp = connection.execute();
99|         Document doc = resp.parse();
100|        String nama = doc.select("div[class=namaUser_d-none_d-lg-block_mr-3]").text();
101|        mahasiswa.setNama(nama.substring(0, nama.indexOf(mahasiswa.getEmailAddress())));
102|        Element photo = doc.select("img[class=img-fluid_fotoProfil]").first();
103|        String photoPath = photo.attr("src");
104|        mahasiswa.setPhotoPath(photoPath);
105|        connection = Jsoup.connect(NILAI_URL);
106|        connection.cookie("ci_session", phpsessid);

```

```

107|     connection.timeout(0);
108|     connection.method(Connection.Method.GET);
109|     resp = connection.execute();
110|     doc = resp.parse();
111|     Elements curr_sem = doc.select("select[class=custom-select_mr-3]");
112|     String[] sem_set = parseSemester(curr_sem.first().child(curr_sem.first().childrenSize() - 1).text());
113|     TahunSemester currTahunSemester = new TahunSemester(Integer.parseInt(sem_set[0]),
114|             Semester.fromString(sem_set[1]));
115|     return currTahunSemester;
116}
117
118 public void requestNilai(String phpsessid, Mahasiswa mahasiswa) throws IOException, InterruptedException {
119     Connection connection = Jsoup.connect(NILAI_URL);
120     connection.cookie("ci_session", phpsessid);
121     connection.timeout(0);
122     connection.method(Connection.Method.POST);
123     Response resp = connection.execute();
124     Document doc = resp.parse();
125
126     Elements dropdownSemester = doc.select("#dropdownSemester_option");
127     ArrayList<String> listSemester = new ArrayList<String>();
128     for (Element semester : dropdownSemester) {
129         listSemester.add(semester.attr("value"));
130     }
131
132     Thread[] threadUrl = new Thread[listSemester.size() - 1];
133     for (int i = 0; i < listSemester.size() - 1; i++) {
134         threadUrl[i] = new Thread(new MultipleRequest(i, listSemester, NILAI_URL, phpsessid, mahasiswa));
135         threadUrl[i].start();
136     }
137     for (int i = 0; i < listSemester.size() - 1; i++) {
138         threadUrl[i].join();
139     }
140     Collections.sort(mahasiswa.getRiwayatNilai(), new Comparator<Nilai>() {
141         @Override
142         public int compare(Nilai o1, Nilai o2) {
143             if (o1.getTahunAjaran() < o2.getTahunAjaran()) {
144                 return -1;
145             }
146             if (o1.getTahunAjaran() > o2.getTahunAjaran()) {
147                 return +1;
148             }
149             if (o1.getSemester().getOrder() < o2.getSemester().getOrder()) {
150                 return -1;
151             }
152             if (o1.getSemester().getOrder() > o2.getSemester().getOrder()) {
153                 return +1;
154             }
155             return 0;
156         }
157     });
158 }
159
160 public void requestNilaiTOEFL(String phpsessid, Mahasiswa mahasiswa) throws IOException {
161     SortedMap<LocalDate, Integer> nilaiTerakhirTOEFL = new TreeMap<>();
162     Connection connection = Jsoup.connect(TOEFL_URL);
163     connection.cookie("ci_session", phpsessid);
164     connection.timeout(0);
165     connection.method(Connection.Method.POST);
166     Response resp = connection.execute();
167     Document doc = resp.parse();
168     Elements nilaiTOEFL = doc.select("table").select("tbody").select("tr");
169     if (!nilaiTOEFL.isEmpty()) {
170         for (int i = 0; i < nilaiTOEFL.size(); i++) {
171             Element nilai = nilaiTOEFL.get(i).select("td").get(5);
172             Element tgl_toefl = nilaiTOEFL.get(i).select("td").get(1);
173             String[] tanggal = tgl_toefl.text().split("-");
174
175             LocalDate localDate = LocalDate.of(Integer.parseInt(tanggal[2]), Integer.parseInt(tanggal[1]),
176                     Integer.parseInt(tanggal[0]));
177
178             nilaiTerakhirTOEFL.put(localDate, Integer.parseInt(nilai.text()));
179         }
180     }
181     mahasiswa.setNilaiTOEFL(nilaiTerakhirTOEFL);
182 }
183
184 public void requestTanggalLahir(String phpsessid, Mahasiswa mahasiswa) throws IOException {
185     Connection connection = Jsoup.connect(PROFILE_URL);
186     connection.cookie("ci_session", phpsessid);
187     connection.timeout(0);
188     connection.method(Connection.Method.POST);
189     Response resp = connection.execute();
190     Document doc = resp.parse();
191
192     Element elementTempatTanggalLahir = doc.select("div[class=offset-md-1_col-md-10_col-12_headerWrapper_my-0_border-bottom]")
193         .first().children().get(1).children().get(1).select("h8").get(1);
194     String tempatTanggalLahir = elementTempatTanggalLahir.text().substring(2);
195
196     StringTokenizer tokenizer = new StringTokenizer(tempatTanggalLahir);
197     int day = Integer.parseInt(tokenizer.nextToken());
198     int month = Arrays.asList(MONTH_NAMES).indexOf(tokenizer.nextToken()) + 1;
199     if (month < 0) {
200         throw new ProtocolException("Month_name_not_recognized_in_this_date:_ " + tempatTanggalLahir);
201     }
202     int year = Integer.parseInt(tokenizer.nextToken());
203     mahasiswa.setTanggalLahir(LocalDate.of(year, month, day));
204 }

```

```

205     public void logout() throws IOException {
206         Connection logoutConn = Jsoup.connect(LOGOUT_URL);
207         logoutConn.timeout(0);
208         logoutConn.method(Connection.Method.GET);
209         logoutConn.execute();
210     }
211
212     public String[] parseSemester(String sem_raw) {
213         String[] sem_set = sem_raw.split("\\/")[0].split("\\_");
214         return new String[]{sem_set[1].trim(), sem_set[0].trim()};
215     }
216
217     public Mahasiswa[] requestMahasiswaList(String phpsessid, Mahasiswa mahasiswa) throws IllegalStateException, IOException,
218         InterruptedException {
219         if (phpsessid == null) {
220             throw new IllegalStateException("Mohon_login_terlebih_dahulu");
221         }
222         this.requestNamePhotoTahunSemester(phpsessid, mahasiswa);
223         String file = "foto_harry.txt";
224
225         BufferedReader reader = new BufferedReader(new FileReader(file));
226         String currentLine = reader.readLine();
227         reader.close();
228
229         mahasiswa.setPhotoPath(currentLine);
230
231         List<Mahasiswa> mahasiswaList;
232         mahasiswaList = new ArrayList<>();
233         mahasiswaList.add(mahasiswa);
234
235         Mahasiswa dummy = new Mahasiswa("2017730001");
236         dummy.setNama("DUMMY_DATA");
237         this.requestMahasiswaDetail(phpsessid, dummy);
238         dummy.setTanggalLahir(LocalDate.of(1999, Month.JANUARY, 1));
239         mahasiswaList.add(dummy);
240         Mahasiswa[] mahasiswaArray = new Mahasiswa[mahasiswaList.size()];
241         for (int i = 0; i < mahasiswaArray.length; i++) {
242             mahasiswaArray[i] = mahasiswaList.get(i);
243         }
244         return mahasiswaArray;
245     }
246
247     public Mahasiswa requestMahasiswaDetail(String phpsessid, Mahasiswa mahasiswa) throws IOException {
248         try {
249             this.requestNilaiTOEFL(phpsessid, mahasiswa);
250             this.requestNilai(phpsessid, mahasiswa);
251             this.requestTanggalLahir(phpsessid, mahasiswa);
252         } catch (InterruptedException ex) {
253             Logger.getLogger(Scraper.class.getName()).log(Level.SEVERE, null, ex);
254         }
255         return mahasiswa;
256     }
257 }

```

Kode B.3: MultipleRequest.java

```

1 package id.ac.unpar.screensaver.studentportal;
2
3 import id.ac.unpar.siamodels.Mahasiswa;
4 import id.ac.unpar.siamodels.MataKuliah;
5 import id.ac.unpar.siamodels.MataKuliahFactory;
6 import id.ac.unpar.siamodels.TahunSemester;
7 import jdk.nashorn.api.scripting.ScriptObjectMirror;
8 import org.jsoup.Connection;
9 import org.jsoup.Jsoup;
10 import org.jsoup.nodes.Document;
11 import org.jsoup.nodes.Element;
12
13 import javax.script.ScriptEngine;
14 import javax.script.ScriptEngineManager;
15 import javax.script.ScriptException;
16 import java.io.IOException;
17 import java.util.ArrayList;
18 import java.util.Map;
19
20 public class MultipleRequest implements Runnable {
21
22     private int index;
23     private ArrayList<String> listSemester;
24     private String NILAI_URL;
25     private String phpsessid;
26     private Mahasiswa mahasiswa;
27     private ScriptEngineManager factory;
28     private ScriptEngine engine;
29
30     public MultipleRequest(int index, ArrayList<String> listSemester, String NILAI_URL, String phpsessid, Mahasiswa mahasiswa) {
31         this.index = index;
32         this.listSemester = listSemester;
33         this.NILAI_URL = NILAI_URL;
34         this.phpsessid = phpsessid;
35         this.mahasiswa = mahasiswa;
36         factory = new ScriptEngineManager();
37         engine = factory.getEngineByName("JavaScript");
38     }
39
40     @Override

```

```
41| public void run() {
42|     try {
43|         String[] thn_sem = listSemester.get(index).split("-");
44|         String thn = thn_sem[0];
45|         String sem = thn_sem[1];
46|         Connection connection = Jsoup.connect(NILAI_URL + "/" + thn + "/" + sem);
47|         connection.cookie("ci_session", phpsessid);
48|         connection.timeout(0);
49|         connection.method(Connection.Method.POST);
50|         Connection.Response resp = connection.execute();
51|         Document doc = resp.parse();
52|
53|         Element script = doc.select("script").get(doc.select("script").size() - 1);
54|         String scriptDataMataKuliah = script.html().substring(script.html().indexOf("var_data_mata_kuliah_=;"), script.html()
55|             .indexOf("var_data_angket_=;"));
56|         engine.eval(scriptDataMataKuliah);
57|         ScriptObjectMirror dataMataKuliah = (ScriptObjectMirror) engine.get("data_mata_kuliah");
58|         TahunSemester tahunSemesterNilai = new TahunSemester(Integer.parseInt(thn), sem.charAt(0));
59|         for (Map.Entry<String, Object> mataKuliahEntry : dataMataKuliah.entrySet()) {
60|             ScriptObjectMirror mataKuliah = (ScriptObjectMirror) mataKuliahEntry.getValue();
61|             MataKuliah curr_mk = MataKuliahFactory.getInstance().createMataKuliah((String) mataKuliah.get("kode_mata_kuliah"),
62|                 Integer.parseInt((String) mataKuliah.get("jumlah_sks")), (String) mataKuliah.get("nama_mata_kuliah"));
63|             mahasiswa.getRiwayatNilai()
64|                 .add(new Mahasiswa.Nilai(tahunSemesterNilai, curr_mk, (String) mataKuliah.get("na")));
65|         }
66|     } catch (IOException e) {
67|         e.printStackTrace();
68|     } catch (ScriptException se) {
69|         se.printStackTrace();
70|     }
71| }
```


LAMPIRAN C

KODE PROGRAM SIAKAD

Kode C.1: SIAkadDataPuller.java

```
1 package id.ac.unpar.screensaver.siaakad;
2
3 import id.ac.unpar.screensaver.DataPuller;
4 import id.ac.unpar.siamodels.Mahasiswa;
5 import java.io.FileNotFoundException;
6 import java.io.FileReader;
7 import java.io.IOException;
8 import java.util.List;
9 import java.util.Properties;
10 import java.util.logging.Level;
11 import java.util.logging.Logger;
12
13 /**
14 * 
15 * @author pascal
16 */
17 public class SIAkadDataPuller extends DataPuller {
18
19     private final SIAkad siaakad;
20
21     public SIAkadDataPuller() throws FileNotFoundException, IOException {
22         Properties auth = new Properties();
23         auth.load(new FileReader("login-dosen.properties"));
24         String username = auth.getProperty("username");
25         String password = auth.getProperty("user.password");
26
27         this.siaakad = new SIAkad();
28         this.siaakad.login(username, password);
29     }
30
31     @Override
32     public Mahasiswa[] pullMahasiswas() {
33         List<Mahasiswa> mahasiswas = null;
34         try {
35             mahasiswas = siaakad.requestMahasiswaList();
36         } catch (IllegalStateException ex) {
37             Logger.getLogger(SIAkadDataPuller.class.getName()).log(Level.SEVERE, null, ex);
38         } catch (IOException ex) {
39             Logger.getLogger(SIAkadDataPuller.class.getName()).log(Level.SEVERE, null, ex);
40         }
41         return (Mahasiswa[]) mahasiswas.toArray(new Mahasiswa[mahasiswas.size()]);
42     }
43
44     @Override
45     public Mahasiswa pullMahasiswaDetail(Mahasiswa m) {
46         try {
47             siaakad.requestRiwayatNilai(m, false);
48             siaakad.requestDataDiri(m);
49         } catch (IllegalStateException ex) {
50             Logger.getLogger(SIAkadDataPuller.class.getName()).log(Level.SEVERE, null, ex);
51         } catch (IOException ex) {
52             Logger.getLogger(SIAkadDataPuller.class.getName()).log(Level.SEVERE, null, ex);
53         }
54         return m;
55     }
56 }
57 }
```

Kode C.2: SIAkad.java

```
1 package id.ac.unpar.screensaver.siaakad;
2
3 import id.ac.unpar.siamodels.JenisKelamin;
4 import id.ac.unpar.siamodels.Mahasiswa;
5 import id.ac.unpar.siamodels.MataKuliahFactory;
6 import id.ac.unpar.siamodels.ProgramStudi;
7 import id.ac.unpar.siamodels.Semester;
8 import id.ac.unpar.siamodels.TahunSemester;
9 import java.io.IOException;
10 import java.net.ProtocolException;
11 import java.time.LocalDate;
12 import java.util.ArrayList;
13 import java.util.Arrays;
```

```

14| import java.util.List;
15| import java.util.Map;
16| import java.util.Set;
17| import java.util.StringTokenizer;
18| import java.util.TreeMap;
19| import java.util.TreeSet;
20| import java.util.logging.Handler;
21| import java.util.logging.Level;
22| import java.util.logging.Logger;
23| import java.util.regex.Matcher;
24| import java.util.regex.Pattern;
25| import org.jsoup.Connection;
26| import org.jsoup.Connection.Method;
27| import org.jsoup.Connection.Response;
28| import org.jsoup.HttpStatusException;
29| import org.jsoup.Jsoup;
30| import org.jsoup.nodes.Document;
31| import org.jsoup.nodes.Element;
32| import org.jsoup.select.Elements;
33|
34| /**
35|  * @author pascal
36|  */
37|
38| public class SIAkad {
39|
40|     protected static final String SIAKAD_BASE_URL = "https://siakad.unpar.ac.id";
41|     protected static final String SSO_BASE_URL = "https://sso.unpar.ac.id";
42|     protected static final String MU_BASE_URL = "https://mu.unpar.ac.id";
43|     protected static final int DEFAULT_TIMEOUT = 60000;
44|
45|     protected final Logger logger = Logger.getLogger("id.ac.unpar.siaakadgateway");
46|
47|     // Implementation choice: I choose to hardcode the names here, because
48|     // it will be more flexible in case SIAkad uses different names compared
49|     // to Java/OS's standard.
50|     public static final String[] MONTH_NAMES = {
51|         "Januari", "Februari", "Maret", "April", "Mei", "Juni", "Juli", "Agustus", "September", "Oktober", "November", "Desember"
52|     };
53|
54|     Token token = null;
55|
56|     public SIAkad() {
57|         this(false);
58|     }
59|
60|     public SIAkad(boolean verbose) {
61|         if (verbose) {
62|             for (Handler handler : logger.getHandlers()) {
63|                 handler.setLevel(Level.WARNING);
64|             }
65|         }
66|     }
67|
68| /**
69|  * Melakukan login ke SI Akademik. Semua transaksi lain harus diawali
70|  * dengan login.
71|  *
72|  * @param username username dosen (...@unpar.ac.id)
73|  * @param password password CAS
74|  * @return true jika login berhasil, false jika username/password salah.
75|  * @throws IOException jika terjadi kesalahan komunikasi
76|  */
77|     public boolean login(String username, String password) throws IOException {
78|         try {
79|             Map<String, String> cookies;
80|             // 0. Trigger SIAkad website
81|             Connection connection = createBaseConnection(SIAKAD_BASE_URL, null);
82|             connection.timeout(DEFAULT_TIMEOUT);
83|             Response response = connection.execute();
84|             cookies = response.cookies();
85|             logConnection(connection);
86|             // Should get a CAS response
87|
88|             // 1. Simulate user lookup
89|             Connection ajaxConnection = Jsoup.connect(MU_BASE_URL + "/api/users/lookup");
90|             ajaxConnection.data("username", username);
91|             ajaxConnection.cookies(cookies);
92|             ajaxConnection.method(Connection.Method.POST);
93|             ajaxConnection.ignoreContentType(true); // JSON is expected
94|             Response ajaxResponse = ajaxConnection.execute();
95|             cookies = ajaxResponse.cookies();
96|             logConnection(ajaxConnection);
97|             // We skip JSON parsing for now
98|
99|             // 2. Perform login at SSO
100|             Document document = response.parse();
101|             String execution = document.select("input[name=execution]").val();
102|             String jsessionid = response.cookie("JSESSIONID");
103|             connection = Jsoup.connect(SSO_BASE_URL + "/login");
104|             connection.cookies(cookies);
105|             connection.data("username", username);
106|             connection.data("password", password);
107|             connection.data("execution", execution);
108|             connection.data("_eventId", "submit");
109|             connection.data("geolocation", "");
110|             connection.data("submit", "Login3");
111|             connection.method(Connection.Method.POST);
112|             response = connection.execute();

```

```

113        logConnection(connection);
114        token = new Token(response.cookies());
115        return true;
116    } catch (HttpStatusException hse) {
117        if (hse.getStatusCode() / 100 == 5) {
118            throw new IOException("Protocol_error:" + hse.getStatusCode() + " " + hse.getUrl());
119        } else {
120            return false;
121        }
122    }
123 }
124 /**
125 * Perform logout
126 *
127 * @throws IOException when there is communication error.
128 */
129 public void logout() throws IOException {
130     Connection connection = createBaseConnection(SIAKAD_BASE_URL + "/logout", null);
131     connection.method(Method.GET);
132     connection.execute();
133     logConnection(connection);
134     token = null;
135 }
136 /**
137 * Mendapatkan daftar mahasiswa yang diwalikan oleh dosen terlogin
138 * (backward compatible tanpa parameter: seluruh program studi (yang
139 * terdaftar), status aktif, dan seluruh angkatan
140 *
141 * @return mahasiswa yang diwalikan
142 * @throws IllegalStateException jika belum login
143 * @throws IOException kesalahan komunikasi
144 */
145 public List<Mahasiswa> requestMahasiswaList() throws IllegalStateException, IOException {
146     return requestMahasiswaList(null, null, null);
147 }
148 /**
149 * Mendapatkan daftar mahasiswa yang diwalikan oleh dosen terlogin
150 *
151 * @param programStudi program studi yang dipilih, atau null jika ingin
152 * mencari dari seluruh program studi yang terdaftar di SIAModels
153 * @param status status mahasiswa yang ingin ditampilkan, atau null jika
154 * default (aktif).
155 * @param angkatan tahun angkatan, atau null jika semua.
156 * @return mahasiswa yang diwalikan
157 * @throws IllegalStateException jika belum login
158 * @throws IOException kesalahan komunikasi
159 */
160 public List<Mahasiswa> requestMahasiswaList(Set<ProgramStudi> programStudi, Mahasiswa.Status status, Integer angkatan) throws
161     IllegalStateException, IOException {
162     if (token == null) {
163         throw new IllegalStateException("Mohon_login_terlebih_dahulu");
164     }
165     if (programStudi == null) {
166         programStudi = new TreeSet<>();
167         programStudi.addAll(Arrays.asList(ProgramStudi.values()));
168     }
169     String url = new StringBuilder(SIAKAD_BASE_URL + "/load_table_cari_mahasiswa/");
170     boolean first = true;
171     for (ProgramStudi ps : programStudi) {
172         if (first) {
173             first = false;
174         } else {
175             url.append(" ");
176         }
177         url.append(ps.getSIAKADCode());
178     }
179     if (status == null) {
180         status = Mahasiswa.Status.AKTIF;
181     }
182     url.append("/").append(status.getSIAKADCode());
183     url.append("/").append(angkatan == null ? "00" : angkatan);
184     url.append("/0"); // This is for NPM filter, but not used
185     Connection connection = createBaseConnection(url.toString(), this.token);
186     connection.method(Method.GET);
187     Connection.Response response = connection.execute();
188     logConnection(connection);
189     Document document = Jsoup.parse(response.body(), response.url().toString());
190     List<Mahasiswa> mahasiswaList;
191     Elements rows = document.select("#data_table_tr");
192     // Note: start from 1 as header is skipped.
193     mahasiswaList = new ArrayList<>(rows.size() - 1);
194     for (int i = 1; i < rows.size(); i++) {
195         Elements columns = (rows.get(i)).select("td");
196         String npm = columns.get(1).select("a").text();
197         String nama = columns.get(2).text();
198         Mahasiswa newMahasiswa = new Mahasiswa(npm);
199         newMahasiswa.setNama(nama);
200         mahasiswaList.add(newMahasiswa);
201     }
202     return mahasiswaList;
203 }
204 /**
205 * Mendapatkan data diri mahasiswa. Saat ini hanya mendukung URL foto,
206 * tanggal lahir, dan jenis kelamin tanpa data diri yang lain.
207 */
208 
```

```

211 * @param mahasiswa mahasiswa yang ingin diperiksa,
212 * @link Mahasiswa#getNpm()} tidak boleh null.
213 * @return objek mahasiswa yang sama, dengan data diri yang sudah
214 * dilengkapi
215 * @throws IllegalStateException jika belum login
216 * @throws IOException kesalahan komunikasi
217 * @throws ProtocolException kesalahan format yang diterima dari SIAkad
218 */
219 public Mahasiswa requestDataDiri(Mahasiswa mahasiswa) throws IllegalStateException, IOException, ProtocolException {
220     if (token == null) {
221         throw new IllegalStateException("Mohon_login_terlebih_dahulu");
222     }
223     Connection connection = createBaseConnection(SIAKAD_BASE_URL + "/data_diri_mahasiswa/" + mahasiswa.getNpm() + "/0", this.
224         token);
225     connection.method(Method.GET);
226     Connection.Response response = connection.execute();
227     logConnection(connection);
228     Document document = Jsoup.parse(response.body(), response.url().toString());
229     Element table = document.selectFirst(".portlet.box.blue_table.table-condensed");
230
231     // Photo
232     Element firstRow = table.selectFirst("tr");
233     String photoPath = firstRow.select("img").attr("src");
234     mahasiswa.setPhotoPath(photoPath);
235
236     Elements tableCells = table.select("td");
237     for (int i = 0; i < tableCells.size(); i++) {
238         if (i < tableCells.size() - 2 && tableCells.get(i + 1).text().equals(":")) {
239             String fieldName = tableCells.get(i).text();
240             String fieldValue = tableCells.get(i + 2).text();
241             switch (fieldName) {
242                 case "Tanggal_Lahir":
243                     StringTokenizer tokenizer = new StringTokenizer(fieldValue);
244                     int day = Integer.parseInt(tokenizer.nextToken());
245                     int month = Arrays.asList(MONTH_NAMES).indexOf(tokenizer.nextToken()) + 1;
246                     if (month < 0) {
247                         throw new ProtocolException("Month_name_not_recognized_in_this_date:_ " + fieldValue);
248                     }
249                     int year = Integer.parseInt(tokenizer.nextToken());
250                     mahasiswa.setTanggalLahir(LocalDate.of(year, month, day));
251                     break;
252                 case "Jenis_Kelamin":
253                     mahasiswa.setJenisKelamin(JenisKelamin.fromSIAKADCode(fieldValue));
254                     break;
255             }
256         }
257     }
258     return mahasiswa;
259 }
260 /**
261 * Mendapatkan daftar lengkap riwayat nilai mahasiswa, berdasarkan
262 * halaman "Data Akademik". Metode ini dapat mengisi lengkap kelas,
263 * nilai-nilai Tugas, UTS, UAS, serta NA.
264 *
265 * {@link Mahasiswa#getNpm()} tidak boleh null.
266 *
267 * @param mahasiswa mahasiswa yang ingin diperiksa,
268 * @param includeLastSemester <strong>(rekomendasi: false)</strong> apakah ingin mengikutsertakan nilai semester terakhir yang
269 * tercatat.
270 * Kelemahan metode "Data Akademik" adalah tidak bisa membedakan antara nilai yang belum rilis dengan yang sudah.
271 * Jika mahasiswa sedang menjalani mata kuliah tersebut, nilai sudah muncul tetapi kemungkinan NA nya berisi E karena
272 * nilai tugas/UTS/UAS belum lengkap. Jika mahasiswa sedang tidak menjalani kuliah tersebut (misal, saat FRS atau saat libur,
273 * maka nilai semester terakhir perlu diikutsertakan, karena mengacu ke nilai yang sudah rilis). Tentu saja ke depannya perlu
274 * mekanisme yang lebih baik yang dapat mendeteksi nilai yang sudah/belum rilis ini.
275 *
276 * @return objek mahasiswa yang sama, dengan nilai yang sudah didapatkan
277 * (terurut secara kronologis dari yang paling lama ke baru).
278 * @throws IllegalStateException jika belum login
279 * @throws IOException kesalahan komunikasi
280 * @see {@link #requestRiwayatNilai(Mahasiswa)}
281 */
282 public Mahasiswa requestRiwayatNilai(Mahasiswa mahasiswa, boolean includeLastSemester) throws IllegalStateException,
283     IOException {
284     if (token == null) {
285         throw new IllegalStateException("Mohon_login_terlebih_dahulu");
286     }
287     // Step 1: Dapatkan semester saat ini
288     Connection connection = createBaseConnection(SIAKAD_BASE_URL + "/data_akademik_mahasiswa/" + mahasiswa.getNpm() + "/0",
289         this.token);
290     connection.method(Method.GET);
291     Connection.Response response = connection.execute();
292     logConnection(connection);
293     Document document = Jsoup.parse(response.body(), response.url().toString());
294     Element nilaiSemesterIni = document.select("#nilai_semester_ini_tab").first();
295     String nilaiSemesterIniText = nilaiSemesterIni.text().trim();
296     TahunSemester tahunSemesterIni = new TahunSemester(nilaiSemesterIniText.substring(nilaiSemesterIniText.length() - 3));
297
298     // Step 2: Dapatkan seluruh nilai
299     List<Mahasiswa.Nilai> riwayatNilai = mahasiswa.getRiwayatNilai();
300     riwayatNilai.clear();
301     connection = createBaseConnection(SIAKAD_BASE_URL + "/load_nilai_per_tahun_semester/" + mahasiswa.getNpm() + "/0", this.
302         token);
303     connection.method(Method.GET);
304     response = connection.execute();
305     logConnection(connection);
306     document = Jsoup.parse(response.body(), response.url().toString());
307     Elements tables = document.select("table.table-condensed");

```

```

304    Pattern tahunAkademikPattern = Pattern.compile("Tahun_Akademik_(\\d{4})/\\d{4}");
305    Pattern semesterPattern = Pattern.compile("Semester_(Ganjil|Genap|Pendek)");
306    for (Element table : tables) {
307        List<String> rowLabels;
308
309        Elements rows = table.select("tr");
310        // Row 0, 3: Tahun Akademik, Semester
311        String tahunAkademikString = rows.get(0).selectFirst("td").text().trim();
312        Matcher tahunAkademikMatcher = tahunAkademikPattern.matcher(tahunAkademikString);
313        int tahun;
314        if (tahunAkademikMatcher.matches()) {
315            tahun = Integer.parseInt(tahunAkademikMatcher.group(1));
316        } else {
317            throw new IOException("Can't find tahun_akademik_in_SIakad_Output: " + tahunAkademikString);
318        }
319        String semesterString = rows.get(3).selectFirst("td").text().trim();
320        Matcher semesterMatcher = semesterPattern.matcher(semesterString);
321        Semester semester;
322        if (semesterMatcher.matches()) {
323            semester = Semester.fromString(semesterMatcher.group(1));
324        } else {
325            throw new IOException("Can't find semester_in_SIakad_Output: " + semesterString);
326        }
327        TahunSemester tahunSemester = new TahunSemester(tahun, semester);
328
329        // Row 2: Labels
330        Elements cols = rows.get(2).select("td");
331        rowLabels = new ArrayList<>(cols.size());
332        for (Element col : cols) {
333            rowLabels.add(col.text());
334        }
335
336        // Row 4 to N-1: The grades
337        for (int i = 4; i < rows.size() - 1; i++) {
338            cols = rows.get(i).select("td");
339            String kode = cols.get(1).text();
340            String nama = cols.get(2).text();
341            int sks = Integer.parseInt(cols.get(3).text());
342            Character kelas;
343            kelas = cols.get(4).text().length() > 0 ? cols.get(4).text().charAt(0) : null;
344            Double uts = null, uas = null;
345            Map<String, Double> nilaiTugas = new TreeMap<>();
346            for (int j = 0; j < rowLabels.size(); j++) {
347                String cellText = cols.get(5 + j).text();
348                if (cellText.length() > 0) {
349                    double cellValue = Double.parseDouble(cellText);
350                    switch (rowLabels.get(i)) {
351                        case "UTS":
352                            uts = cellValue;
353                            break;
354                        case "UAS":
355                            uas = cellValue;
356                            break;
357                        default:
358                            nilaiTugas.put(rowLabels.get(j), cellValue);
359                            break;
360                    }
361                }
362            }
363            String nilaiAkhir = cols.get(cols.size() - 1).text();
364            if (nilaiAkhir.length() == 0) {
365                nilaiAkhir = null;
366            }
367            if (includeLastSemester || !tahunSemester.equals(tahunSemesterIni)) {
368                // Exclude nilai from current tahun/semester be cause most likely it's yet to be released
369                Mahasiswa.Nilai nilai = new Mahasiswa.Nilai(tahunSemester,
370                    MataKuliahFactory.getInstance().createMataKuliah(kode, sks, nama), kelas, nilaiTugas, uts, uas, nilaiAkhir
371                );
372                riwayatNilai.add(nilai);
373            }
374        }
375        return mahasiswa;
376    }
377
378    /**
379     * Create the base connection, with logged in state..
380     *
381     * @param url URL to connect
382     * @param token the auth token, if any
383     * @return the connection
384     */
385    protected Connection createBaseConnection(String url, Token token) {
386        Connection connection = Jsoup.connect(url);
387        if (token != null) {
388            token.injectToConnection(connection);
389        }
390        connection.timeout(DEFAULT_TIMEOUT);
391        return connection;
392    }
393
394    /**
395     * This method checks the verbose flag, and when true will print debug
396     * output.
397     *
398     * @param connection the connection request and response
399     */
400    protected void logConnection(Connection connection) {
401        logger.log(Level.FINE, "Request:{0}{1}{2}{3}{4}\nResponse:{5}{6}\nBody:{7}\n====\n",

```

```
402     new Object[]{connection.request().method(),
403     connection.request().url(),
404     connection.request().headers(),
405     connection.request().data(),
406     connection.request().cookies(),
407     connection.response().statusCode(),
408     connection.response().headers(),
409     connection.response().body()});
410 }
411 }
```

Kode C.3: Token.java

```
1 package id.ac.unpar.screensaver.siakad;
2
3 import java.util.Map;
4 import org.jsoup.Connection;
5
6 public class Token {
7
8     private final Map<String, String> cookies;
9
10    public Token(Map<String, String> cookies) {
11        this.cookies = cookies;
12    }
13
14    public Map<String, String> getCookies() {
15        return cookies;
16    }
17
18    /**
19     * Inject this session into a Jsoup connection
20     *
21     * @param connection the connection to inject
22     */
23    public void injectToConnection(Connection connection) {
24        cookies.entrySet().forEach((cookie) -> {
25            connection.cookie(cookie.getKey(), cookie.getValue());
26        });
27    }
28 }
```

LAMPIRAN D

PERBEDAAN KODE DOSEN DENGAN MAHASISWA

Kode D.1: Perbedaan kode dosen dengan mahasiswa

```
1 diff --git a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/PrimaryController.java b/ScreenSaver/src/main/java/id/ac/unpar/
2 index 8ca4319..d8f3f26 100644
3 --- a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/PrimaryController.java
4 +++ b/ScreenSaver/src/main/java/id/ac/unpar/screensaver/PrimaryController.java
5 @@ -1,7 +1,6 @@
6 package id.ac.unpar.screensaver;
7
8 -//import id.ac.unpar.screensaver.siakad.SIAkadDataPuller;
9 -import id.ac.unpar.screensaver.studentportal.StudentPortalDataPuller;
10 +import id.ac.unpar.screensaver.siakad.SIAkadDataPuller;
11 import id.ac.unpar.siamodels.Mahasiswa;
12 import id.ac.unpar.siamodels.TahunSemester;
13 import java.io.ByteArrayInputStream;
14 @@ -32,16 +31,22 @@ public class PrimaryController implements Initializable {
15
16     private int indexOfMahasiswa = 0;
17     private Mahasiswa[] listMahasiswa;
18 +    private boolean[] mahasiswaLoaded;
19     private DataPuller puller;
20
21     public int getIndexOfMahasiswa() {
22         return indexOfMahasiswa;
23     }
24
25 -    public void setIndexOfMahasiswa(int indexOfMahasiswa) {
26 +    public void setIndexOfMahasiswaAndPreload(int indexOfMahasiswa) {
27         this.indexOfMahasiswa = indexOfMahasiswa;
28 +        if (!mahasiswaLoaded[indexOfMahasiswa]) {
29 +            new MahasiswaDetailPuller(listMahasiswa[indexOfMahasiswa]).start();
30 +        } else {
31 +            System.out.println("No longer pulling mahasiswa detail for " + listMahasiswa[indexOfMahasiswa].getNama() + " because
already pulled before");
32 +        }
33     }
34 -
35 +
36     public PrimaryController() throws IOException {
37
38 }
39 @@ -49,71 +54,36 @@ public class PrimaryController implements Initializable {
40     @Override
41     public void initialize(URL url, ResourceBundle rb) {
42         try {
43             puller = new StudentPortalDataPuller();
44 +            puller = new SIAkadDataPuller();
45             listMahasiswa = puller.pullMahasiswas();
46 -            } catch (IllegalStateException ex) {
47 -                Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
48 -            }
49 -            if (listMahasiswa != null) {
50 -                try {
51 -                    listMahasiswa[this.getIndexOfMahasiswa()] = puller.pullMahasiswaDetail(listMahasiswa[this.getIndexOfMahasiswa()]);
52 -                this.updateView(listMahasiswa[this.getIndexOfMahasiswa()]);
53 -                this.setIndexOfMahasiswa(this.getIndexOfMahasiswa() + 1);
54 -                } catch (IOException ex) {
55 -                    Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
56 -                }
57 -            } else {
58 -                updateView();
59 -            }
60 -            Timeline timeline = new Timeline(
61 -                new KeyFrame(
62 -                    Duration.seconds(5),
63 -                    event -> {
64 -                        if (listMahasiswa == null) {
65 -                            updateView();
66 +                        listMahasiswa[this.getIndexOfMahasiswa()] = puller.pullMahasiswaDetail(listMahasiswa[this.getIndexOfMahasiswa()]);
67 +                        mahasiswaLoaded = new boolean[listMahasiswa.length];
68 +                        this.updateView(listMahasiswa[this.getIndexOfMahasiswa()]);
69 +                        this.setIndexOfMahasiswaAndPreload(this.getIndexOfMahasiswa() + 1);
70 +                        Timeline timeline = new Timeline(
71 +                            new KeyFrame(
72 +                                Duration.seconds(15), // May need to adjust longer if internet is slow
```

```

73+           event -> {
74-             try {
75-               puller = new StudentPortalDataPuller();
76-               listMahasiswa = puller.pullMahasiswa();
77-             } catch (IllegalStateException ex) {
78-               Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
79-             }
80-           } else {
81-             if (this.getIndexOfMahasiswa() == listMahasiswa.length) {
82-               this.setIndexOfMahasiswa(0);
83-             } else {
84-               if (listMahasiswa[this.getIndexOfMahasiswa()].getTanggalLahir() == null) {
85-                 listMahasiswa[this.getIndexOfMahasiswa()] = puller.pullMahasiswaDetail(listMahasiswa[this
86-                   .getIndexOfMahasiswa()]);
87-               }
88-             }
89-             if (listMahasiswa[this.getIndexOfMahasiswa()].getTanggalLahir() == null) {
90-               updateView();
91-             } else {
92+               try {
93+                 if (mahasiswaLoaded[this.getIndexOfMahasiswa()]) {
94+                   // Update view only if mahasiswa is loaded. Otherwise, wait for next turn
95+                   this.updateView(listMahasiswa[this.getIndexOfMahasiswa()]);
96+                   this.setIndexOfMahasiswa(this.getIndexOfMahasiswa() + 1);
97+                 } catch (IOException ex) {
98+                   Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
99+                   this.setIndexOfMahasiswaAndPreload((this.getIndexOfMahasiswa() + 1) % listMahasiswa.
100+                     length);
101+                 } else {
102+                   System.out.println("Mahasiswa " + listMahasiswa[this.getIndexOfMahasiswa()].getNama() + " "
103+                     .isNotReady. Waiting for next turn...");
104+                 }
105+               }
106-             }
107-           );
108-         timeline.setCycleCount(Animation.INDEFINITE);
109-         timeline.play();
110-       }
111+     );
112+   );
113+   timeline.setCycleCount(Animation.INDEFINITE);
114+   timeline.play();
115 }

116 }

117

118 public void updateView() {
119   this.foto.setVisible(false);
120   this.nama.setText("Pastikan koneksi internet berfungsi dengan normal!");
121   this.angkatan.setText("-");
122   this.usia.setText("-");
123   this.status.setText("-");
124   this.email.setText("-");
125   this.toefl.setText("-");
126   this.ipk.setText("-");
127   this.sks.setText("-");
128+ } catch (IllegalStateException | IOException ex) {
129+   Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
130+ }
131 }

132 public void updateView(Mahasiswa mahasiswa) throws IOException {
@@ -129,13 +99,12 @@ public class PrimaryController implements Initializable {
133   }
134   this.nama.setText(mahasiswa.getNama());
135   this.angkatan.setText(mahasiswa.getTahunAngkatan() + "");
136   this.usia.setText(Period.between(mahasiswa.getTanggalLahir(), LocalDate.now()).getYears() + " tahun " + Period.between(
137     mahasiswa.getTanggalLahir(), LocalDate.now()).getMonths() + " bulan " + "(lahir " + mahasiswa.getTanggalLahir().toString()
138     + ")");
139   if (mahasiswa.getStatus() != null) {
140     this.status.setText(mahasiswa.getStatus().toString());
141   } else{
142     this.status.setText("Tidak Tersedia");
143   }
144   this.usia.setText(
145+   mahasiswa.getTanggalLahir() != null ?
146+     (Period.between(mahasiswa.getTanggalLahir(), LocalDate.now()).getYears() + " tahun " + Period.between(mahasiswa.
147     getTanggalLahir(), LocalDate.now()).getMonths() + " bulan (lahir " + mahasiswa.getTanggalLahir().toString() + ")") :
148+     "Tidak tersedia"
149+   );
150+   this.status.setText(mahasiswa.getStatus() != null ? mahasiswa.getStatus().toString() : "Tidak Tersedia");
151   this.email.setText(mahasiswa.getEmailAddress());
152   if (mahasiswa.getNilaiTOEFL() != null && mahasiswa.getNilaiTOEFL().size() > 0) {
153     this.toefl.setText(mahasiswa.getNilaiTOEFL().get(mahasiswa.getNilaiTOEFL().firstKey()).toString());
@@ -143,12 +112,34 @@ public class PrimaryController implements Initializable {
154   }
155   this.toefl.setText("Tidak Tersedia");
156 }
157 TahunSemester tahunSemesterTerakhir = null;
158 for (Mahasiswa.Nilai nilai : mahasiswa.getRiwayatNilai()) {
159+ for (Mahasiswa.Nilai nilai : mahasiswa.getRiwayatNilai()) {
160   if (tahunSemesterTerakhir == null || nilai.getTahunSemester().compareTo(tahunSemesterTerakhir) > 0) {
161     tahunSemesterTerakhir = nilai.getTahunSemester();
162   }
163 }
164 this.ipk.setText(Math.round(mahasiswa.calculateIPS(tahunSemesterTerakhir) * 100.0) / 100.0 + "/" + Math.round(mahasiswa.
calculateIPK() * 100.0) / 100.0);

```

```

165+     this.ipk.setText(mahasiswa.getRiwayatNilai().isEmpty() ?
166+         "Tidak tersedia" :
167+         Math.round(mahasiswa.calculateIPSKetikaTerakhir) * 100.0) / 100.0 + "/" + Math.round(mahasiswa.
168+             calculateIPK() * 100.0) / 100.0;
169+     this.sks.setText(+mahasiswa.calculateSKSLulus() + "/" + mahasiswa.calculateSKSTempuh(false));
170+
171+
172+     private class MahasiswaDetailPuller extends Thread {
173+         private Mahasiswa mahasiswa;
174+         public MahasiswaDetailPuller(Mahasiswa mahasiswa) {
175+             this.mahasiswa = mahasiswa;
176+         }
177+         public void run() {
178+             System.out.println("Pulling mahasiswa detail for " + mahasiswa.getNama());
179+             puller.pullMahasiswaDetail(mahasiswa);
180+             for (int i = 0; i < listMahasiswa.length; i++) {
181+                 if (listMahasiswa[i] == this.mahasiswa) {
182+                     mahasiswaLoaded[i] = true;
183+                     System.out.println("Pulled mahasiswa detail for " + mahasiswa.getNama() + " (index " + i + ")");
184+                     break;
185+                 }
186+             }
187+         }
188+     }
189+ }
190}
191diff —git a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/siakad/SIAkad.java b/ScreenSaver/src/main/java/id/ac/unpar/
screensaver/siakad/SIAkad.java
192index a1fc3e..66a4512 100644
193--- a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/siakad/SIAkad.java
194+++ b/ScreenSaver/src/main/java/id/ac/unpar/screensaver/siakad/SIAkad.java
195@@ -53,7 +53,7 @@ public class SIAkad {
196    // it will be more flexible in case SIAkad uses different names compared
197    // to Java/OS's standard.
198    public static final String[] MONTH_NAMES = {
199-        "Januari", "Februari", "Maret", "April", "Mei", "Juni", "Juli", "Agustus", "September", "Oktober", "November", "Desember"
200+        "Januari", "Pebruari", "Maret", "April", "Mei", "Juni", "Juli", "Agustus", "September", "Oktober", "November", "Desember"
201    };
202
203    Token token = null;
204@@ -247,7 +247,7 @@ public class SIAkad {
205        StringTokenizer tokenizer = new StringTokenizer(fieldValue);
206        int day = Integer.parseInt(tokenizer.nextToken());
207        int month = Arrays.asList(MONTH_NAMES).indexOf(tokenizer.nextToken()) + 1;
208-        if (month < 0) {
209+        if (month <= 0) {
210            throw new ProtocolException("Month name not recognized in this date: " + fieldValue);
211        }
212        int year = Integer.parseInt(tokenizer.nextToken());
213diff —git a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/siakad/SIAkadDataPuller.java b/ScreenSaver/src/main/java/id/ac/
unpar/screensaver/siakad/SIAkadDataPuller.java
214index 82c35bc..27c0eed 100644
215--- a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/siakad/SIAkadDataPuller.java
216+++ b/ScreenSaver/src/main/java/id/ac/unpar/screensaver/siakad/SIAkadDataPuller.java
217@@ -10,10 +10,12 @@ import id.ac.unpar.siamodels.Mahasiswa;
218import java.io.FileNotFoundException;
219import java.io.FileReader;
220import java.io.IOException;
221+import java.util.Collections;
222import java.util.List;
223import java.util.Properties;
224import java.util.logging.Level;
225import java.util.logging.Logger;
226+import java.util.Random;
227
228 /**
229 *
230@@ -36,8 +38,10 @@ public class SIAkadDataPuller extends DataPuller {
231    @Override
232    public Mahasiswa[] pullMahasiswas() {
233        List<Mahasiswa> mahasiswas = null;
234+        Random random = new Random(13); // "stable" random
235        try {
236            mahasiswas = siakad.requestMahasiswaList();
237+            Collections.shuffle(mahasiswas, random);
238        } catch (IllegalStateException ex) {
239            Logger.getLogger(SIAkadDataPuller.class.getName()).log(Level.SEVERE, null, ex);
240        } catch (IOException ex) {
241diff —git a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/MultipleRequest.java b/ScreenSaver/src/main/java/id/
ac/unpar/screensaver/studentportal/MultipleRequest.java
242index 30b3207..d166180 100644
243--- a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/MultipleRequest.java
244+++ b/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/MultipleRequest.java
245@@ -4,7 +4,7 @@ import id.ac.unpar.siamodels.Mahasiswa;
246import id.ac.unpar.siamodels.MataKuliah;
247import id.ac.unpar.siamodels.MataKuliahFactory;
248import id.ac.unpar.siamodels.TahunSemester;
249-import jdk.nashorn.api.scripting.ScriptObjectMirror;
250+// import jdk.nashorn.api.scripting.ScriptObjectMirror;
251import org.jsoup.Connection;
252import org.jsoup.Jsoup;
253import org.jsoup.nodes.Document;
254@@ -50,17 +50,17 @@ public class MultipleRequest implements Runnable {
255        Connection.Response resp = connection.execute();
256        Document doc = resp.parse();
257
258-        Element script = doc.select("script").get(doc.select("script").size() - 1);
259+        Element script = doc.select("script").get(doc.select("script").size()-1);

```

```

260     String scriptDataMataKuliah = script.html().substring(script.html().indexOf("var data_mata_kuliah = []"), script.
261     html().indexOf("var data_angket = []"));
262 -     engine.eval(scriptDataMataKuliah);
262 +     ScriptObjectMirror dataMataKuliah = (ScriptObjectMirror) engine.get("data_mata_kuliah");
263 + // ScriptObjectMirror dataMataKuliah = (ScriptObjectMirror) engine.get("data_mata_kuliah");
264     TahunSemester tahunSemesterNilai = new TahunSemester(Integer.parseInt(thn), sem.charAt(0));
265 -     for (Map.Entry<String, Object> mataKuliahEntry : dataMataKuliah.entrySet()) {
266 -         ScriptObjectMirror mataKuliah = (ScriptObjectMirror) mataKuliahEntry.getValue();
267 -         MataKuliah curr_mk = MataKuliahFactory.getInstance().createMataKuliah((String) mataKuliah.get("kode_mata_kuliah"))
268 -         , Integer.parseInt((String) mataKuliah.get("jumlah_sks")), (String) mataKuliah.get("nama_mata_kuliah"));
269 -         mahasiswa.getRiwayatNilai()
270 -             .add(new Mahasiswa.Nilai(tahunSemesterNilai, curr_mk, (String) mataKuliah.get("na")));
270 +
271 +         // for (Map.Entry<String, Object> mataKuliahEntry : dataMataKuliah.entrySet()) {
272 +             // ScriptObjectMirror mataKuliah = (ScriptObjectMirror) mataKuliahEntry.getValue();
273 +             // MataKuliah curr_mk = MataKuliahFactory.getInstance().createMataKuliah((String) mataKuliah.get("kode_mata_kuliah"), Integer.parseInt((String) mataKuliah.get("jumlah_sks")), (String) mataKuliah.get("nama_mata_kuliah"));
274 +             // mahasiswa.getRiwayatNilai()
275 +             // .add(new Mahasiswa.Nilai(tahunSemesterNilai, curr_mk, (String) mataKuliah.get("na")));
276 +
277     } catch (IOException e) {
278         e.printStackTrace();
279     } catch (ScriptException se) {
280 diff —git a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/Scraper.java b/ScreenSaver/src/main/java/id/ac/unpar/
281 screensaver/studentportal/Scraper.java
281 index 2786909..1f7a8ed 100644
282 —— a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/Scraper.java
283 +++ b/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/Scraper.java
284 @@ -237,8 +237,9 @@ public class Scraper {
285
286     Mahasiswa dummy = new Mahasiswa("2017730001");
287     dummy.setNama("DUMMY DATA");
288 +     dummy.setJenisKelamin(JenisKelamin.PEREMPUAN);
289 +     dummy.setTanggalLahir(LocalDate.of(1, 1, 1));
290     this.requestMahasiswaDetail(phpsessid, dummy);
291 -     dummy.setTanggalLahir(LocalDate.of(1999, Month.JANUARY, 1));
292     mahasiswaList.add(dummy);
293     Mahasiswa[] mahasiswaArray = new Mahasiswa[mahasiswaList.size()];
294     for (int i = 0; i < mahasiswaArray.length; i++) {
295 diff —git a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/StudentPortalDataPuller.java b/ScreenSaver/src/main/
296 java/id/ac/unpar/screensaver/studentportal/StudentPortalDataPuller.java
296 index e0e9e02..a165218 100644
297 —— a/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/StudentPortalDataPuller.java
298 +++ b/ScreenSaver/src/main/java/id/ac/unpar/screensaver/studentportal/StudentPortalDataPuller.java
299 @@ -36,6 +36,7 @@ public class StudentPortalDataPuller extends DataPuller {
300     this.mahasiswa = new Mahasiswa(np);
301     this.scrapers = new Scrapers();
302     this.session = this.scrapers.login(np, password);
303 +
304     } catch (IOException ex) {
305         Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
306     }

```

LAMPIRAN E

KODE PROGRAM FXML

Kode E.1: primary.fxml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <?import javafx.geometry.Insets?>
4 <?import javafx.scene.image.ImageView?>
5 <?import javafx.scene.layout.HBox?>
6 <?import javafx.scene.layout.StackPane?>
7 <?import javafx.scene.layout.VBox?>
8 <?import javafx.scene.text.Font?>
9 <?import javafx.scene.text.Text?>
10
11 <StackPane xmlns="http://javafx.com/javafx/11.0.1" xmlns:fx="http://javafx.com/fxml/1" fx:controller="id.ac.unpar.screensaver.PrimaryController">
12     <children>
13         <ImageView fx:id="foto" fitWidth="500.0" pickOnBounds="true" preserveRatio="true" StackPane.alignment="CENTER_LEFT">
14             <StackPane.margin>
15                 <Insets left="50.0" right="50.0" />
16             </StackPane.margin>
17         </ImageView>
18         <VBox alignment="CENTER_LEFT" nodeOrientation="LEFT_TO_RIGHT" StackPane.alignment="CENTER">
19             <children>
20                 <Text fx:id="nama" strokeType="OUTSIDE" strokeWidth="0.0" VBox.vgrow="ALWAYS">
21                     <font>
22                         <Font size="55.0" />
23                     </font>
24                     <VBox.margin>
25                         <Insets bottom="50.0" top="50.0" />
26                     </VBox.margin>
27                 </Text>
28             <HBox>
29                 <children>
30                     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Mahasiswa_angkatan:<u>">
31                         <font>
32                             <Font size="40.0" />
33                         </font>
34                     </Text>
35                     <Text fx:id="angkatan" strokeType="OUTSIDE" strokeWidth="0.0">
36                         <font>
37                             <Font size="40.0" />
38                         </font>
39                     </Text>
40                 </children>
41                 <VBox.margin>
42                     <Insets bottom="40.0" />
43                 </VBox.margin>
44             <HBox>
45                 <children>
46                     <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Usia:<u>">
47                         <font>
48                             <Font size="40.0" />
49                         </font>
50                     </Text>
51                     <Text fx:id="usia" strokeType="OUTSIDE" strokeWidth="0.0">
52                         <font>
53                             <Font size="40.0" />
54                         </font>
55                     <HBox.margin>
56                         <Insets />
57                     </HBox.margin>
58                 </Text>
59             </children>
60             <VBox.margin>
61                 <Insets bottom="40.0" />
62             </VBox.margin>
63         <HBox>
64             <children>
65                 <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Status:<u>">
66                     <font>
67                         <Font size="40.0" />
68                     </font>
69                 </Text>
70                 <Text fx:id="status" strokeType="OUTSIDE" strokeWidth="0.0">
71                     <font>
72                         <Font size="40.0" />
73                     </font>
74                 </Text>
75             </children>
76         </HBox>
77     </children>
78 
```

```
75          </font>
76      </Text>
77  </children>
78  <VBox.margin>
79      <Insets bottom="40.0" />
80  </VBox.margin>
81 </HBox>
82 <HBox>
83     <children>
84         <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Email:_">
85             <font>
86                 <Font size="40.0" />
87             </font>
88         </Text>
89         <Text fx:id="email" strokeType="OUTSIDE" strokeWidth="0.0">
90             <font>
91                 <Font size="40.0" />
92             </font>
93         </Text>
94     </children>
95     <VBox.margin>
96         <Insets bottom="40.0" />
97     </VBox.margin>
98 </HBox>
99 <HBox>
100    <children>
101        <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Nilai_TOEFL:_">
102            <font>
103                <Font size="40.0" />
104            </font>
105        </Text>
106        <Text fx:id="toefl" strokeType="OUTSIDE" strokeWidth="0.0">
107            <font>
108                <Font size="40.0" />
109            </font>
110        </Text>
111    </children>
112    <VBox.margin>
113        <Insets bottom="40.0" />
114    </VBox.margin>
115 </HBox>
116 <HBox>
117     <children>
118         <Text strokeType="OUTSIDE" strokeWidth="0.0" text="IPS/IPK:_">
119             <font>
120                 <Font size="40.0" />
121             </font>
122         </Text>
123         <Text fx:id="ipk" strokeType="OUTSIDE" strokeWidth="0.0">
124             <font>
125                 <Font size="40.0" />
126             </font>
127         </Text>
128     </children>
129     <VBox.margin>
130         <Insets bottom="40.0" />
131     </VBox.margin>
132 </HBox>
133 <HBox>
134     <children>
135         <Text strokeType="OUTSIDE" strokeWidth="0.0" text="SKS_Lulus/Tempuh:_">
136             <font>
137                 <Font size="40.0" />
138             </font>
139         </Text>
140         <Text fx:id="sks" strokeType="OUTSIDE" strokeWidth="0.0">
141             <font>
142                 <Font size="40.0" />
143             </font>
144         </Text>
145     </children>
146     <VBox.margin>
147         <Insets bottom="40.0" />
148     </VBox.margin>
149 </HBox>
150 </children>
151 <StackPane.margin>
152     <Insets left="600.0" right="50.0" />
153 </StackPane.margin>
154 </VBox>
155 </children>
156 </StackPane>
```