

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»

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»

DAFTAR ISI

DAFTAR ISI	ix
DAFTAR GAMBAR	xi
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	2
1.6 Sistematika Pembahasan	2
2 LANDASAN TEORI	5
2.1 Jsoup	5
2.1.1 Jsoup	5
2.1.2 Connection	5
2.1.3 Response	6
2.1.4 Elements	7
2.1.5 Element	7
2.2 JavaFX dan FXML	8
2.2.1 JavaFX	8
2.2.2 FXML	9
2.3 SIAModels	10
2.3.1 Mahasiswa	10
2.3.2 Nilai	13
2.3.3 ChronologicalComparator	14
2.3.4 MataKuliah	14
2.3.5 JenisKelamin	15
2.3.6 Status	15
2.3.7 TahunSemester	15
3 ANALISIS	17
3.1 Analisis Portal Akademik Mahasiswa	17
3.1.1 <i>Login</i>	17
3.1.2 Halaman Utama	19
3.1.3 Profil	19
3.1.4 Jadwal	20
3.1.5 Pembayaran	25
3.1.6 FRS/PRS	26
3.1.7 Nilai	27
3.1.8 Angket	29
3.1.9 Saran & Komentar	30
3.1.10 Kelulusan	31

3.1.11	Pengajuan	31
3.2	Analisis SIAKAD	32
3.2.1	Login	32
3.2.2	Mahasiswa	32
3.2.3	Pra Kuliah	38
3.2.4	Perkuliahinan	38
3.2.5	Ujian	38
3.2.6	Nilai	38
3.2.7	Evaluasi	38
3.2.8	Skripsi	39
3.2.9	Sidang	39
3.2.10	Kelulusan	39
3.2.11	Pengumuman	39
3.3	Data yang Dibutuhkan untuk <i>Screensaver</i>	39
3.3.1	Portal Akademik Mahasiswa	39
3.3.2	SIAKAD	42
3.4	Analisis Sistem <i>Screensaver</i>	44
3.4.1	<i>Use Case Screensaver</i>	44
3.4.2	Perancangan Kelas <i>Screensaver</i>	45
4	PERANCANGAN	47
4.1	Perancangan Kelas	47
4.2	Perancangan Antarmuka	58
5	IMPLEMENTASI DAN PENGUJIAN	61
5.1	Implementasi	61
5.1.1	Lingkungan Implementasi	61
5.1.2	Hasil Implementasi	61
5.2	Pengujian	63
5.2.1	Pengujian Fungsional	63
5.2.2	Pengujian Eksperimental	63
6	KESIMPULAN DAN SARAN	69
6.1	Kesimpulan	69
6.2	Saran	69
DAFTAR REFERENSI		71
A	KODE PROGRAM <i>Screensaver</i>	73
B	KODE PROGRAM PORTAL AKADEMIK MAHASISWA (STUDENTPORTAL)	77
C	KODE PROGRAM SIAKAD	83
D	PERBEDAAN KODE DOSEN DENGAN MAHASISWA	89
E	KODE PROGRAM FXML	93

DAFTAR GAMBAR

3.1 Halaman <i>Login 1</i>	18
3.2 Halaman <i>Login 2</i>	18
3.3 Halaman Utama Portal Akademik Mahasiswa	19
3.4 Halaman Profil	20
3.5 Halaman Kehadiran	21
3.6 Halaman Ketidakhadiran	21
3.7 Halaman Jadwal Kuliah Dalam Tabel Waktu	22
3.8 Halaman Jadwal Kuliah Tabel	22
3.9 Halaman UTS	23
3.10 Halaman UAS	23
3.11 Halaman MKU	24
3.12 Halaman Kalender Akademik	24
3.13 Halaman Daftar Kehadiran	25
3.14 Halaman Pembayaran	26
3.15 Tampilan FRS/PRS	26
3.16 Halaman Nilai Per Semester	27
3.17 Halaman Daftar Perkembangan Studi (1)	28
3.18 Halaman Daftar Perkembangan Studi (2)	28
3.19 Halaman Riwayat Indeks Prestasi	29
3.20 Halaman Nilai TOEFL	29
3.21 Halaman Angket	30
3.22 Halaman Saran & Komentar	30
3.23 Halaman Kelulusan	31
3.24 Halaman Pengajuan	32
3.25 Tangkapan Layar Halaman Daftar Mahasiswa Wali	33
3.26 Tangkapan Layar Pop-up Detail Mahasiswa	33
3.27 Tangkapan Layar Tab Identitas Mahasiswa	34
3.28 Tangkapan Layar Tab Riwayat Pendidikan	35
3.29 Tangkapan Layar Tab Orang Tua/Wali	35
3.30 Tangkapan Layar Tab Ringkasan Akademik	36
3.31 Tangkapan Layar Tab Nilai Semester Ini	36
3.32 Tangkapan Layar Tab Nilai Per Tahun Semester	37
3.33 Tangkapan Layar Tab Nilai Berdasarkan Kurikulum	37
3.34 Tangkapan Layar Tab Status Akademik	38
3.35 Tangkapan Layar PDF Daftar Perkembangan Studi	39
3.36 Diagram <i>Use Case Screensaver</i>	44
3.37 Perancangan Kelas <i>Screensaver</i>	45
4.1 Diagram Kelas Keseluruhan	47
4.2 Diagram Kelas SIAKAD	48
4.3 Diagram Kelas Studentportal	49
4.4 <i>Script</i> Data Nilai Mahasiswa Pada Halaman Nilai	55

4.5 Rancangan Antarmuka <i>Screensaver</i>	59
5.1 Tampilan <i>Screensaver</i> Mahasiswa Pertama	62
5.2 Tampilan <i>Screensaver</i> Mahasiswa Kedua	62
5.3 Tampilan <i>Screensaver</i> Mahasiswa Pertama	65
5.4 Tampilan <i>Screensaver</i> Mahasiswa Kedua	65
5.5 Tampilan <i>Screensaver</i> Mahasiswa Ketiga	66
5.6 Tampilan <i>Screensaver</i> Mahasiswa Keempat	66
5.7 Tampilan <i>Screensaver</i> Mahasiswa Kelima	67

¹

BAB 1

²

PENDAHULUAN

³ 1.1 Latar Belakang

⁴ Setiap dosen wali (dikenal juga dengan istilah penasehat akademik ¹) memiliki data mengenai
⁵ mahasiswa walinya yang dapat diakses melalui SIAKAD [1]. Namun, walaupun dosen wali memiliki
⁶ data mengenai mahasiswa walinya, dosen wali juga perlu melakukan pemeriksaan data mahasiswa
⁷ walinya, terutama data akademiknya secara berkala. Dengan berbagai kesibukan yang dialami oleh
⁸ para dosen wali dan mahasiswa, ditambah dengan situasi Indonesia saat ini yang menyebabkan
⁹ perkuliahan dilakukan secara daring, akan sangat sulit bagi dosen wali untuk menemui mahasiswa
¹⁰ walinya. Hal ini menyebabkan dosen wali kesulitan mengamati perkembangan mahasiswa walinya.
¹¹ Selain itu dalam mencari data akademik mahasiswa wali, SIAKAD juga memiliki kekurangan
¹² dimana dosen wali perlu melakukan *login*, kemudian mencari serta memilih mahasiswa wali yang
¹³ ingin dilihat. Sehingga dapat dikatakan proses tersebut tidaklah efisien.

¹⁴ Maka dari itu, pada skripsi ini akan dibuat sebuah perangkat lunak yang berupa *screensaver*
¹⁵ yang dapat menampilkan data akademik mahasiswa wali secara acak. *Screensaver* adalah program
¹⁶ komputer yang mengosongkan layar atau mengisinya dengan gambar atau pola bergerak ketika
¹⁷ komputer telah diam dalam waktu yang lama [2]. Dengan menggunakan perangkat lunak tersebut,
¹⁸ dosen wali dapat tetap mengamati perkembangan mahasiswa walinya, paling tidak secara akademik.

¹⁹ Dikarenakan terbimbing tidak memiliki akses ke SIAKAD untuk mengakses data mahasiswa
²⁰ wali, namun terbimbing memiliki akses ke Portal Akademik Mahasiswa [3] maka, terbimbing
²¹ mensimulasikan dengan Portal Akademik Mahasiswa, dan kemudian Pembimbing mengubah aksesnya
²² ke SIAKAD. Struktur kelas yang akan digunakan dalam pembuatan perangkat lunak ini yaitu,
²³ struktur kelas SIAModels yang tersedia pada Github dan Maven Public Repository [4]. Simulasi
²⁴ Portal Akademik Mahasiswa ini berdasarkan pada skripsi yang dibuat oleh Andrianto Sugiarto [5].
²⁵ Tetapi terdapat beberapa perbaikan yang perlu dilakukan, dan dapat dilihat pada sub-bab [3.1](#).

²⁶ Perangkat lunak akan dibangun menggunakan bahasa pemrograman Java. Terdapat beberapa
²⁷ teknologi yang dapat dimanfaatkan dalam bahasa pemrograman Java. Teknologi yang pertama
²⁸ yaitu *library* jsoup. Jsoup dapat digunakan untuk melakukan *scraping*, sehingga pengambilan data
²⁹ mahasiswa tidak memerlukan API (*Application Programming Interface*) [6]. Teknologi lainnya yang
³⁰ dapat dimanfaatkan yaitu JavaFX. JavaFX dapat digunakan untuk mengonversi aplikasi tersebut
³¹ 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 1.2 Rumusan Masalah

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

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

6 1.3 Tujuan

7 Tujuan yang ingin dicapai dari penulisan skripsi ini sebagai berikut:

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

10 1.4 Batasan Masalah

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

14 1.5 Metodologi

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

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

27 1.6 Sistematika Pembahasan

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

- 29 • Bab 1. Pendahuluan, membahas tentang latar belakang, rumusan masalah, tujuan penelitian,
30 batasan masalah, metode penelitian dan sistematika pembahasan mengenai skripsi.
- 31 • Bab 2. Landasan Teori, membahas landasan dari teori-teori yang berhubungan serta mendukung
32 penelitian, meliputi jsoup, JavaFX, dan SIAModels.
- 33 • Bab 3. Analisis, menjelaskan tentang analisis Portal Akademik Mahasiswa, analisis SIAKAD,
34 analisis data yang dibutuhkan untuk *screensaver*, serta analisis sistem *screensaver*.

- 1 ● Bab 4. Perancangan, membahas perancangan kelas beserta deskripsi kelas dan fungsinya,
2 serta perancangan antarmuka.
- 3 ● Bab 5. Implementasi dan pengujian, membahas hasil-hasil implementasi dan pengujian secara
4 fungsional dan eksperimental.
- 5 ● Bab 6. Kesimpulan dan saran, membahas kesimpulan yang diperoleh dari penelitian ini dan
6 saran untuk pengembangan berikutnya.

1

BAB 2

2

LANDASAN TEORI

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

5 **2.1 Jsoup**

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

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

19 Subbab-subbab berikut menjelaskan beberapa kelas dari jsoup.

20 **2.1.1 Jsoup**

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

- 23 • `public static Connection connect(String url)`

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

26 **Parameter:** URL situs web dengan protokol HTTP atau HTTPS.

27 **Kembalian:** koneksi dengan situs web.

28 **2.1.2 Connection**

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

1 sebagai berikut:

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

3 Berfungsi untuk menambahkan *cookies* ke *request*.

4 **Parameter:**

- 5 – `cookies`: Map dari *cookie*.

6 **Kembalian:** koneksi yang sama tetapi sudah diubah.

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

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

10 **Parameter:**

- 11 – `key`: kunci data.

- 12 – `value`: nilai data.

13 **Kembalian:** koneksi yang sama tetapi sudah diubah.

- 14 • `Connection method(URLConnection method)`

15 Berfungsi untuk mengatur *method request* yang akan digunakan, HTTP GET atau POST.

16 *Default*-nya adalah GET.

17 **Parameter:**

- 18 – `method`: *method request* HTTP.

19 **Kembalian:** koneksi yang sama tetapi sudah diubah.

- 20 • `Connection timeout(int millis)`

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

23 **Parameter:**

- 24 – `millis`: batas waktu dalam milidetik.

25 **Kembalian:** koneksi yang sama tetapi sudah diubah.

- 26 • `Connection.Response execute()`

27 Berfungsi untuk mengirim *request* HTTP.

28 **Kembalian:** objek Response.

29 2.1.3 Response

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

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

33 Berfungsi untuk mendapatkan seluruh *cookies*.

34 **Kembalian:** seluruh *cookies*.

- 35 • `Document parse()`

36 Berfungsi untuk mengurai *body* jawaban menjadi dokumen.

37 **Kembalian:** dokumen yang diurai.

- 38 • `String body()`

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

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

1 2.1.4 Elements

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

- 4 • `public Elements select(String query)`

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

6 **Parameter:**

7 – *query*: kueri CSS berupa CSS Selector.

8 **Kembalian:** elemen-elemen yang sudah diseleksi sesuai kueri.

- 9 • `public String val()`

10 Berfungsi untuk mendapatkan nilai dari elemen pertama.

11 **Kembalian:** nilai elemen.

- 12 • `public String text()`

13 Berfungsi untuk mendapatkan kombinasi teks dari seluruh elemen yang sesuai.

14 **Kembalian:** seluruh teks dalam *string*.

15 2.1.5 Element

16 Kelas ini merepresentasikan sebuah elemen HTML yang berisikan *tag*, atribut, dan anak elemen.

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

- 18 • `public Elements select(String cssQuery)`

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

21 **Parameter:**

22 – *cssQuery*: kueri CSS berupa CSS Selector.

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

- 24 • `public Element child(int index)`

25 Berfungsi untuk mendapatkan anak elemen berdasarkan nomor indeks.

26 **Parameter:**

27 – *index*: nomor index.

28 **Kembalian:** anak elemen.

- 29 • `public Element children()`

30 Berfungsi untuk mendapatkan seluruh anak elemen.

31 **Kembalian:** seluruh anak elemen.

- 32 • `public String className()`

33 Berfungsi untuk mendapatkan nama kelas elemen.

34 **Kembalian:** nama kelas elemen.

- 35 • `public String text()`

36 Berfungsi untuk mendapatkan teks gabungan dari elemen.

37 **Kembalian:** teks dalam *string*.

1 2.2 JavaFX dan FXML

2 2.2.1 JavaFX

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

7 Application

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

- 10 1. Membuat *instance* kelas `Application` yang ditentukan
- 11 2. Memanggil *method* `init()`
- 12 3. Memanggil *method* `start(javafx.stage.Stage)`
- 13 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
- 14 5. Memanggil *method* `stop()`

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

- 16 • `public void init()`

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

- 20 • `public abstract void start(Stage primaryStage)`

21 Titik masuk utama untuk semua aplikasi JavaFX. *Method* `start` dipanggil setelah *method*
22 `init` kembali, dan setelah sistem siap untuk aplikasi mulai berjalan.

23 Parameter:

24 – `primaryStage`: *stage* utama untuk aplikasi ini, tempat *scene* aplikasi dapat diatur.

- 25 • `public static void launch()`

26 Meluncurkan aplikasi. *Method* ini biasanya dipanggil dari `main()` *method*. Tidak boleh
27 dipanggil lebih dari sekali atau *exception* akan dilemparkan. Harus merupakan *subclass* dari
28 `Application` atau *RuntimeException* akan dilemparkan.

29

30 Stage

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

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

- 34 • `public final void setScene(Scene value)`

35 Menentukan *scene* yang akan digunakan di *stage* ini.

36 Parameter:

1 – **value**: *scene* yang akan digunakan.
2 • **public final void show()**
3 Mencoba menampilkan *window* ini dengan mengubah *visibility* menjadi true.

4 **Scene**

5 Kelas **Scene** adalah wadah untuk semua konten dalam grafik *scene*.
6 Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:
7 • **public Scene(Parent root, double width, double height)**
8 Merupakan *constructor* dari kelas **Scene**.
9 **Parameter:**
10 – **root**: Node root dari grafik *scene*.
11 – **width**: Lebar *scene*.
12 – **height**: Tinggi *scene*.

13 **FXMLLoader**

14 Memuat hierarki objek dari dokumen XML.
15 Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:
16 • **public FXMLLoader(URL location)**
17 Merupakan *constructor* dari kelas **FXMLLoader**.
18 **Parameter:**
19 – **location**: lokasi dokumen fxml.
20 • **public <T> T load**
21 Memuat hierarki objek dari dokumen FXML.

22 **2.2.2 FXML**

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

28 **Controller Method Event Handlers**

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

34 **Variable Resolution**

35 Dokumen FXML mendefinisikan *namespace* variabel dimana elemen bernama dan variabel dapat
36 diidentifikasi secara unik. Menetapkan nilai fx:id ke elemen akan membuat variabel dokumen yang

- 1 nantinya dapat dirujuk oleh atribut dereferensi variabel. Selain itu, jika tipe objek mendefinisikan
- 2 properti `id`, nilai ini juga akan diteruskan ke `method setId()` milik objek.

3 2.3 SIAModels

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

7 2.3.1 Mahasiswa

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

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

18 Beberapa `method` yang dimiliki kelas ini adalah sebagai berikut:

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

21 **Parameter:**

22 – `npm`: nomor pokok mahasiswa.

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

25 **Kembalian:** nama mahasiswa.

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

28 **Parameter:**

29 – `nama`: nama mahasiswa.

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

32 **Kembalian:** nomor pokok mahasiswa.

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

35 **Kembalian:** URL dari foto mahasiswa.

- 36 • `public void setPhotoPath(String photoPath)`
Berfungsi untuk mengubah URL dari foto mahasiswa.

38 **Parameter:**

39 – `photoPath`: URL dari foto mahasiswa.

- ```
1 • public List<JadwalKuliah> getJadwalKuliahList
2 Berfungsi untuk mendapatkan jadwal kuliah mahasiswa.
3 Kembalian: jadwal kuliah mahasiswa dalam list.
4 • public void setJadwalKuliahList(List<JadwalKuliah> jadwalKuliahList)
5 Berfungsi untuk mengubah jadwal kuliah mahasiswa.
6 Parameter:
7 – jadwalKuliahList: jadwal kuliah mahasiswa dalam list.
8 • public String getEmailAddress()
9 Berfungsi untuk mendapatkan email mahasiswa.
10 Kembalian: email mahasiswa.
11 • public List<Nilai> getRiwayatNilai()
12 Berfungsi untuk mendapatkan riwayat nilai mahasiswa.
13 Kembalian: riwayat nilai mahasiswa dalam List.
14 • public SortedMap<LocalDate, Integer> getNilaiTOEFL()
15 Berfungsi untuk mendapatkan nilai TOEFL mahasiswa.
16 Kembalian: nilai TOEFL mahasiswa dalam SortedMap.
17 • public void setNilaiTOEFL(SortedMap<LocalDate, Integer> nilaiTOEFL)
18 Berfungsi untuk mengubah nilai TOEFL mahasiswa.
19 Parameter:
20 – nilaiTOEFL: nilai TOEFL mahasiswa dalam SortedMap.
21 • public Status getStatus()
22 Berfungsi untuk mendapatkan status mahasiswa.
23 Kembalian: status mahasiswa.
24 • public void setStatus(Status status)
25 Berfungsi untuk mengubah status mahasiswa.
26 Parameter:
27 – status: status mahasiswa.
28 • public LocalDate getTanggalLahir()
29 Berfungsi untuk mendapatkan tanggal lahir mahasiswa.
30 Kembalian: tanggal lahir mahasiswa.
31 • public void setTanggalLahir(LocalDate tanggalLahir)
32 Berfungsi untuk mengubah tanggal lahir mahasiswa.
33 Parameter:
34 – tanggalLahir: tanggal lahir mahasiswa.
35 • public JenisKelamin getJenisKelamin()
36 Berfungsi untuk mendapatkan jenis kelamin mahasiswa.
37 Kembalian: jenis kelamin mahasiswa.
38 • public void setJenisKelamin(JenisKelamin jenisKelamin)
39 Berfungsi untuk mengubah jenis kelamin mahasiswa.
40 Parameter:
41 – jenisKelamin: jenis kelamin mahasiswa.
42 • public byte[] getPhotoImage()
```

1 Berfungsi untuk mendapatkan foto mahasiswa yang dibungkus dalam kelas `java.awt.Image`.  
2 Berbeda dengan *method* `getPhotoPath()`, *method* ini akan menghasilkan image, apapun  
3 bentuk photo path nya (bisa berupa URL ataupun base64 string).

4 **Kembalian:** foto mahasiswa.

- 5 • `public double calculateIPKLulus()`

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

9 **Kembalian:** IPK lulus.

- 10 • `public double calculateIPTempuh(boolean lulusSaja)`

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

14 **Parameter:**

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

18 **Kembalian:** IPK lulus.

- 19 • `public double calculateIPKumulatif()`

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

23 **Kembalian:** IPK lulus.

- 24 • `public double calculateIPS()`

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

28 **Kembalian:** nilai IPS sampai saat ini.

- 29 • `public int calculateSKSLulus()`

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

32 **Kembalian:** SKS lulus.

- 33 • `public int calculateSKSTempuh(boolean lulusSaja)`

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

36 **Parameter:**

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

38 **Kembalian:** SKS tempuh

- 39 • `public Set<TahunSemester> calculateTahunSemesterAktif()`

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

1       **Kembalian:** kumpulan tahun semester di mana mahasiswa ini aktif.

- 2     • `public boolean hasLulusKuliah(String kodeMataKuliah)`

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

5       **Parameter:**

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

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

- 8     • `public boolean hasTempuhKuliah(String kodeMataKuliah)`

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

11       **Parameter:**

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

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

- 14     • `public int getTahunAngkatan()`

15       Mendapatkan tahun angkatan mahasiswa ini berdasarkan NPM-nya.

16       **Kembalian:** tahun angkatan.

17     

### 2.3.2 Nilai

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

- 20     • `TahunSemester tahunSemester`: tahun dan semester kuliah ini diambil

- 21     • `MataKuliah mataKuliah`: mata kuliah yang diambil.

- 22     • `Character kelas`: kelas kuliah.

- 23     • `Map<String, Double> nilaiTugas`: nilai Angka Rata-rata Tugas (ART).

- 24     • `Double nilaiUTS`: nilai Ujian Tengah Semester (UTS).

- 25     • `Double nilaiUAS`: nilai Ujian Akhir Semester (UAS).

- 26     • `String nilaiAkhir`: nilai akhir.

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

- 28     • `public Nilai(TahunSemester tahunSemester, MataKuliah mataKuliah,`

29           `Character kelas, Map<String, Double> nilaiTugas, Double nilaiUTS,`

30           `Double nilaiUAS, String nilaiAkhir)`

31       Merupakan *constructor* dari kelas `Nilai`.

32       **Parameter:**

- 33           – `tahunSemester`: tahun dan semester kuliah ini diambil.

- 34           – `mataKuliah`: mata kuliah yang diambil.

- 35           – `kelas`: kelas kuliah.

- 36           – `nilaiTugas`: nilai ART.

- 37           – `nilaiUTS`: nilai UTS.

- 38           – `nilaiUAS`: nilai UAS.

- 39           – `nilaiAkhir`: nilai akhir.

- 40     • `public MataKuliah getMataKuliah()`

41       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.

1       **Kembalian:** kode mata kuliah.

- 2       • `public int getSks()`

3       Mendapatkan sks mata kuliah.

4       **Kembalian:** sks mata kuliah.

- 5       • `public String getNama()`

6       Mendapatkan nama mata kuliah.

7       **Kembalian:** nama mata kuliah.

8     

### 2.3.5 JenisKelamin

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

- 11      • `LAKI_LAKI("Laki-laki")`

- 12      • `PEREMPUAN("Perempuan")`

13    

### 2.3.6 Status

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

- 15      • `SEMUA("00")`

- 16      • `AKTIF("01")`

- 17      • `GENCAT("02")`

- 18      • `CUTI_SEBELUM_FRS("03")`

- 19      • `CUTI_SETELAH_FRS("04")`

- 20      • `KELUAR("05")`

- 21      • `LULUS("06")`

- 22      • `DROP_OUT("07")`

- 23      • `SISIPAN("08")`

24    

### 2.3.7 TahunSemester

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

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

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

- 31      • `public TahunSemester(String kodeTahunSemester)`

32       *Method* ini merupakan constructor dari kelas TahunSemester.

33       **Parameter:**

34       – `kodeTahunSemester:` semester dalam bentuk teks (GANJIL, GENAP, PENDEK, TRAN-  
35       SFER, dan UNKNOWN5).

- 36      • `public TahunSemester(int tahun, Semester semester)`

37       *Method* ini merupakan constructor dari kelas TahunSemester.

38       **Parameter:**

1       – **tahun**: tahun ajaran.  
2       – **semester**: semester dari tahun ajaran.  
3     • **public Semester getSemester()**  
4       *Method* ini berfungsi untuk mendapatkan semester.  
5       **Kembalian:** semester dalam teks.  
6     • **public int getTahun()**  
7       *Method* ini berfungsi untuk mendapatkan tahun.  
8       **Kembalian:** tahun ajaran.  
9     • **private static void validateKodeSemester(String kodeTahunSemester)**  
10      *Method* ini berfungsi untuk melakukan validasi terhadap kode tahun semester.  
11      **Parameter:**  
12       – **kodeTahunSemester**: kode tahun semester.

<sup>1</sup>

## BAB 3

<sup>2</sup>

## ANALISIS

- <sup>3</sup> Pada bab ini akan dijelaskan mengenai analisis Portal Akademik Mahasiswa, analisis SIAKAD,  
<sup>4</sup> analisis data yang dibutuhkan untuk *screensaver*, serta analisis sistem *screensaver*.

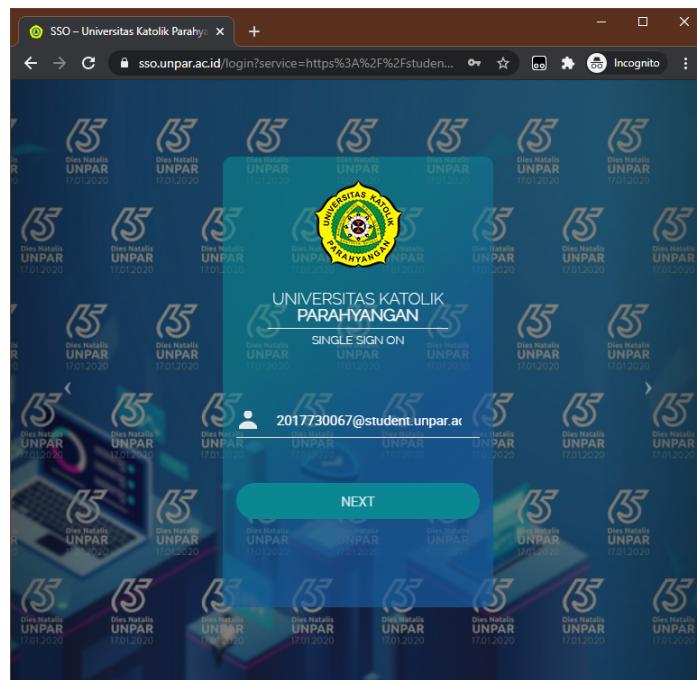
### <sup>5</sup> 3.1 Analisis Portal Akademik Mahasiswa

<sup>6</sup> Untuk mengambil data mahasiswa, diperlukan sumber data mahasiswa tersebut. Sumber data  
<sup>7</sup> mahasiswa tersebut dapat diperoleh melalui Portal Akademik Mahasiswa. Portal Akademik Ma-  
<sup>8</sup> siswa merupakan sebuah situs yang diperuntukkan bagi mahasiswa untuk mendapatkan informasi  
<sup>9</sup> mengenai profil dan kegiatan akademik mahasiswa tersebut. Mahasiswa dapat mengakses Portal  
<sup>10</sup> Akademik Mahasiswa melalui URL <https://studentportal.unpar.ac.id/>. Untuk mengakses  
<sup>11</sup> Portal Akademik Mahasiswa, mahasiswa harus melakukan *login* menggunakan *email* dan *password*  
<sup>12</sup> mahasiswa tersebut.

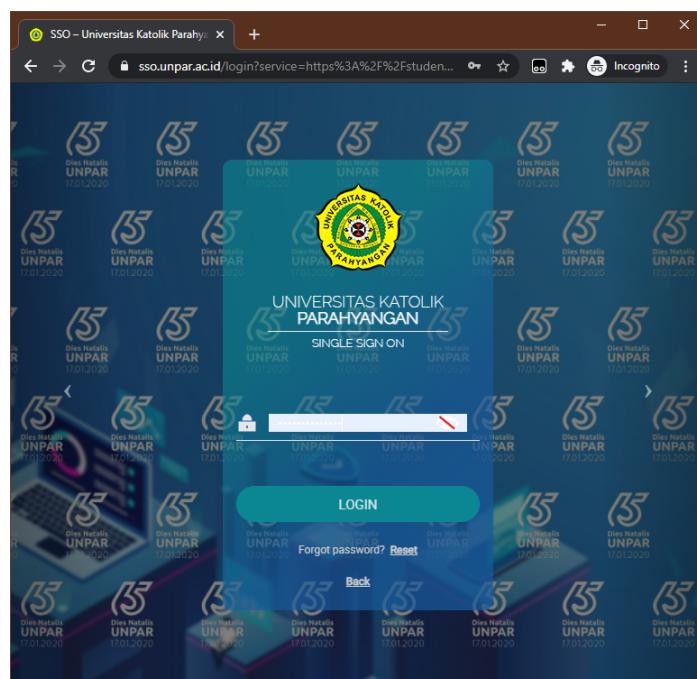
<sup>13</sup> Aplikasi *screensaver* akan melakukan *http request* ke Portal Akademik Mahasiswa untuk menda-  
<sup>14</sup> patkan data untuk setiap kebutuhan dari masing-masing fitur yang ada, dimana fitur-fitur yang  
<sup>15</sup> terdapat dalam aplikasi Mahasiswa Wali *Screensaver* adalah informasi umum mengenai mahasiswa,  
<sup>16</sup> serta prestasi akademik mahasiswa. Pengambilan data secara langsung dari Portal Akademik  
<sup>17</sup> Mahasiswa dilakukan menggunakan *library jsoup*. Beberapa implementasi pemanfaatan jsoup  
<sup>18</sup> untuk mengambil data-data tersebut sudah diimplementasikan pada skripsi Andrianto Sugiarto  
<sup>19</sup> [5] sebelumnya. Data yang telah didapat dari Portal Akademik Mahasiswa kemudian diolah ke  
<sup>20</sup> dalam SIAModels, dan ditampilkan sesuai dengan fitur-fitur yang ada pada aplikasi Mahasiswa  
<sup>21</sup> Wali *Screensaver*.

#### <sup>22</sup> 3.1.1 *Login*

<sup>23</sup> Halaman *Login* (Gambar 3.1 dan 3.2) merupakan halaman dimana mahasiswa memasukkan *email*  
<sup>24</sup> dan *password* untuk mengakses menu-menu Portal Akademik Mahasiswa.



Gambar 3.1: Halaman Login 1



Gambar 3.2: Halaman Login 2

- 1 *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:
2. 1. Menghapus pemanggilan fungsi validateTLCertificates() dikarenakan sudah *deprecated* [6].
2. Menghapus pengambilan data dengan kueri css "`input [name=lt]`" dikarenakan kueri tersebut

1       sudah dihapus oleh Portal Akademik Mahasiswa.

2       3. Menghapus *form data* dengan *key* "submit" yang memiliki *value* "".

Kode 3.1: Perubahan Implementasi Jsoup Login

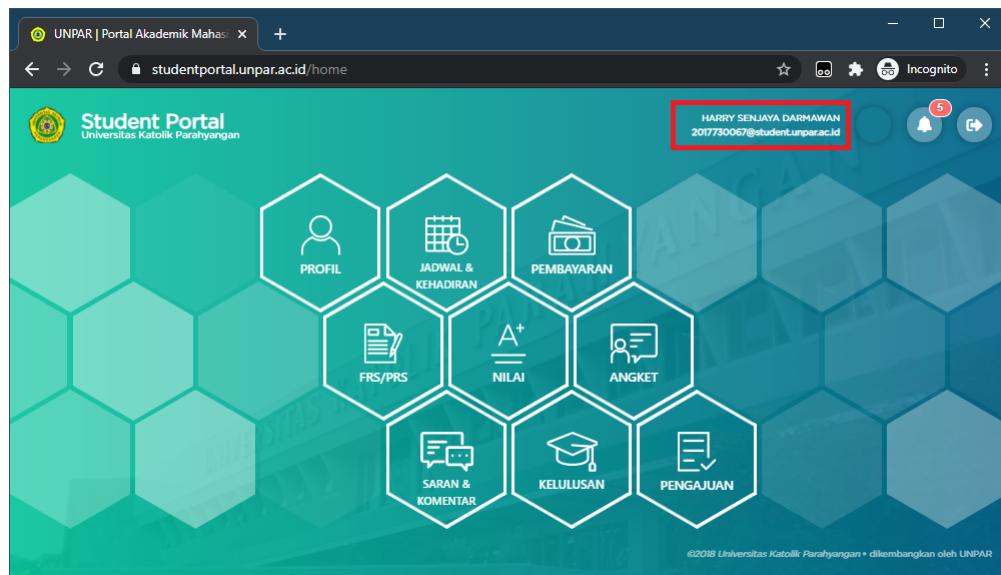
```

3 41 @@ -78,11 +77,9 @@ public class Scraper {
5 52 Connection conn = Jsoup.connect(LOGIN_URL);
6 63 conn.data("Submit", "Login");
7 74 conn.timeout(0);
8 85 - conn.validateTLCertificates(false);
9 96 conn.method(Connection.Method.POST);
10 107 Response resp = conn.execute();
11 118 Document doc = resp.parse();
12 129 String lt = doc.select("input[name=lt]").val();
13 130 String execution = doc.select("input[name=execution]").val();
14 141 String jsessionid = resp.cookie("JSESSIONID");
15 152 /* SSO LOGIN */
16 163 @@ -90,12 +87,9 @@ public class Scraper {
17 174 loginConn.cookies(resp.cookies());
18 185 loginConn.data("username", user);
19 196 loginConn.data("password", pass);
20 207 - loginConn.data("lt", lt);
21 218 loginConn.data("execution", execution);
22 229 loginConn.data("eventId", "submit");
23 230 - loginConn.data("submit", "");
24 241 loginConn.timeout(0);
25 252 - loginConn.validateTLCertificates(false);
26 263 loginConn.method(Connection.Method.POST);
27 274 resp = loginConn.execute();
28 285 if (resp.body().contains(user)) {
29

```

### 30     3.1.2 Halaman Utama

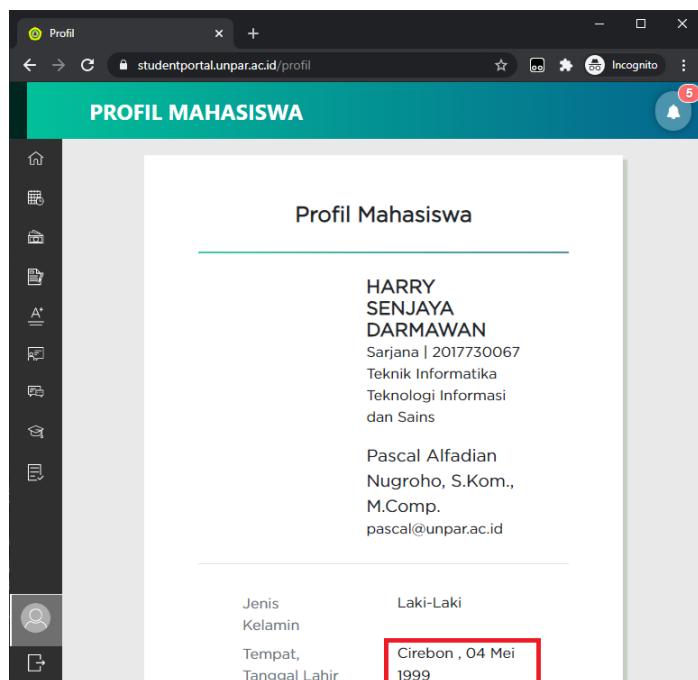
31   Pada Halaman Utama Portal Akademik Mahasiswa (Gambar 3.3) terdapat beberapa menu yang  
 32   dapat digunakan sebagai sumber data.



Gambar 3.3: Halaman Utama Portal Akademik Mahasiswa

### 33     3.1.3 Profil

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



Gambar 3.4: Halaman Profil

#### 1 3.1.4 Jadwal

- 2 Menu Jadwal terdiri dari beberapa submenu:
- 3 • Kehadiran
- 4 Submenu ini berfungsi untuk menandakan kehadiran mahasiswa di suatu mata kuliah pada
- 5 hari dimana mahasiswa tersebut mengakses halaman tersebut (Gambar 3.5).

The screenshot shows the 'Kehadiran' (Attendance) section of the student portal. At the top, there is a red circular icon with a white exclamation mark containing the text: 'Harap perhatikan waktu kehadiran pada kolom WAKTU PERKULIAHAN. Validasi perkuliahan digunakan untuk menyatakan perkuliahan ada atau tidak. Tombol validasi perkuliahan (VALIDASI) aktif apabila ditunjuk sebagai validator.' Below this, the date 'Senin, 22 Maret 2021 16:36:44' is displayed. There are two buttons at the top right: 'Video User Manual Kehadiran' and 'User Manual Kehadiran'. The main content is a table with columns: No, Kode Mata Kuliah, Nama Mata Kuliah, Pertemuan-ke, Kelas, Waktu Perkuliahan, Status Kehadiran, Waktu Hadir, Link Pembelajaran, Presensi, and Validasi Perkuliahan. One row is shown for 'AIFI84002 Skripsi 2' in 'Kelas A' at '14:00 - 15:00'. The 'Status Kehadiran' is 'Hadir', 'Waktu Hadir' is '14:00', 'Link Pembelajaran' is 'https://zoom.us/j/93.....', and there are checkboxes for 'Presensi' and 'Validasi Perkuliahan'.

Gambar 3.5: Halaman Kehadiran

1     ● Ketidakhadiran

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

The screenshot shows the 'Ketidakhadiran' (Absence) section of the student portal. At the top, there is a red circular icon with a white exclamation mark containing the text: 'UNGGAH SURAT SAKIT / SURAT IZIN'. Below this, the date 'Senin, 22 Maret 2021 16:37:18' is displayed. There is a button 'User Manual Ketidakhadiran'. The form fields include 'Alasan Ketidakhadiran' (radio buttons for 'Sakit' and 'Izin'), 'Tanggal Sakit/Izin' (text input), 'Unggah Surat Sakit/Izin' (file input with 'Browse' button), and a note '\*dokumen bentuk pdf, maks 1 MB'. A green button 'Simpan & Kirim' is present. Below the form is a table with columns: No, Tanggal Sakit/Izin (sub-columns Tanggal Awal and Tanggal Akhir), Jenis Ketidakhadiran, Status, Dokumen, and Hapus. The table currently shows 'No data available in table'. At the bottom, it says 'Showing 0 to 0 of 0 entries'.

Gambar 3.6: Halaman Ketidakhadiran

4     ● Kuliah

- 1 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).

| Senin | Selasa | Rabu | Kamis | Jumat |
|-------|--------|------|-------|-------|
| 08:00 |        |      |       |       |
| 09:00 |        |      |       |       |
| 10:00 |        |      |       |       |
| 11:00 |        |      |       |       |
| 12:00 |        |      |       |       |
| 13:00 |        |      |       |       |
| 14:00 |        |      |       |       |
| 15:00 |        |      |       |       |

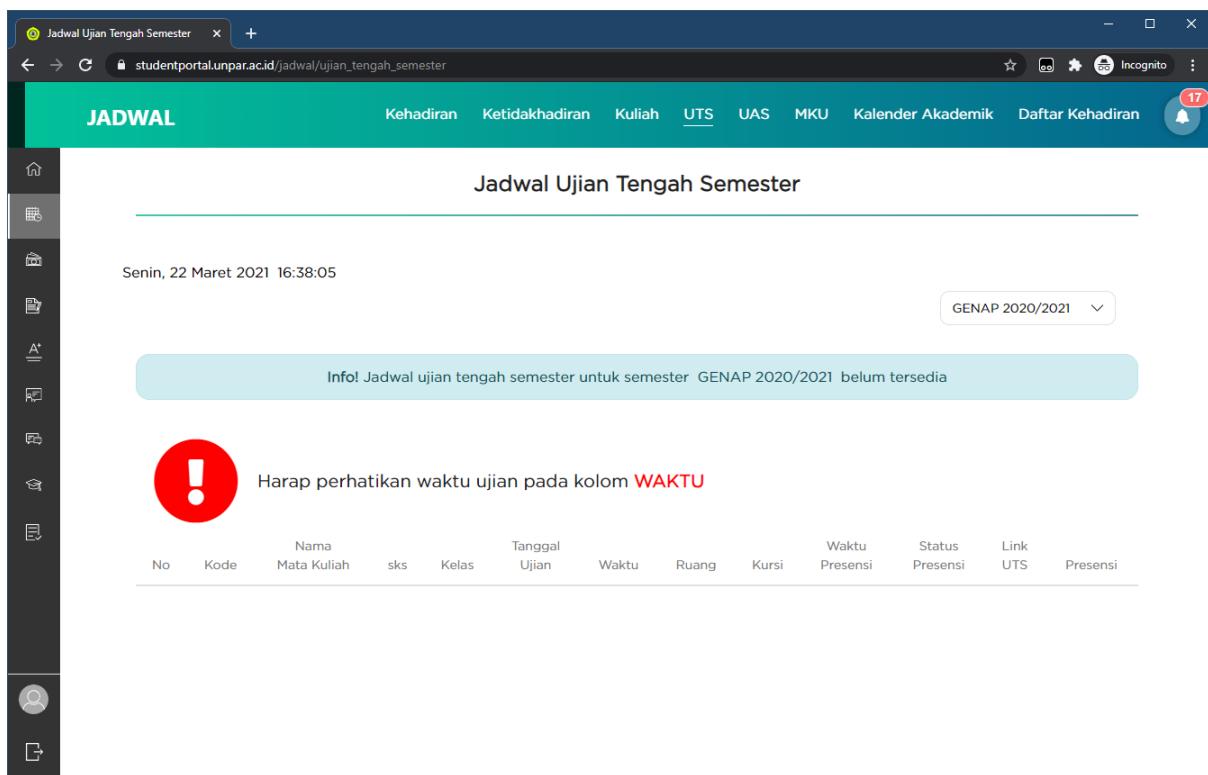
Gambar 3.7: Halaman Jadwal Kuliah Dalam Tabel Waktu

| 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

3 • UTS

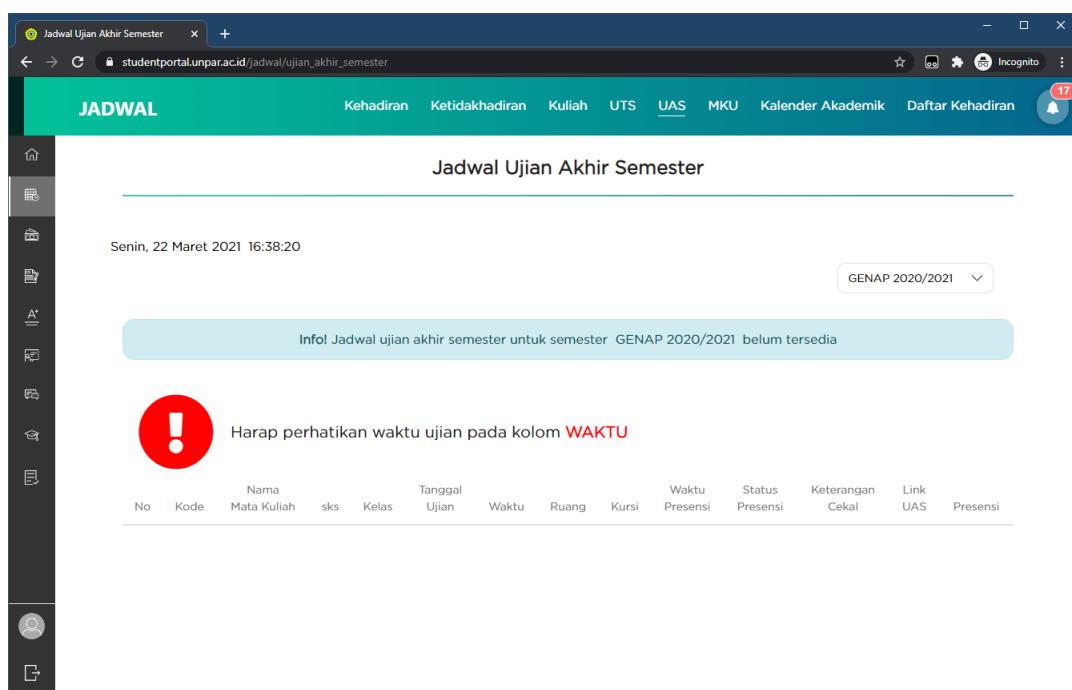
- 4 Submenu ini berisi tentang UTS yang dapat disusun per semester (Gambar 3.9).



Gambar 3.9: Halaman UTS

## 1 • UAS

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



Gambar 3.10: Halaman UAS

## 3 • 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).

| No. | KEGIATAN                                                               | SEMULA                                                     | MENJADI                             |
|-----|------------------------------------------------------------------------|------------------------------------------------------------|-------------------------------------|
| 1   | Periode Pembayaran Biaya Kuliah Tahap I (UKPS)                         | Senin-Jumat, 15-30 Februari 2021                           | Selasa-Selasa, 9-23 Februari 2021   |
| 2   | Masa Formulir Rencana Studi (PRS) Mahasiswa                            | Rabu-Jumat, 3-5 Maret 2021                                 | Rabu-Jumat, 24-26 Februari 2021     |
| 3   | Pengumuman Kelas Paralel Sementara                                     | Jumat, 12 Maret 2021                                       | Jumat, 5 Maret 2021<br>(13.00 WIB)  |
| 4   | Masa Akhir dan Batas Yudisium Semester Ganjil Tahun Akademik 2020/2021 | Sabtu, 13 Maret 2021                                       | Jumat, 5 Maret 2021                 |
| 5   | Batas Akhir Pendaftaran Wisuda I 2020/2021                             | Sabtu, 13 Maret 2021                                       | Jumat, 5 Maret 2021                 |
| 6   | Awal Perkuliahan/Awal Semester Genap 2020/2021                         | Senin, 15 Maret 2021                                       | Senin, 8 Maret 2021                 |
| 7   | Masa Perubahan Rencana Studi (PRS)                                     | Selasa-Rabu, 30-31 Maret 2021                              | Rabu-Kamus, 24-25 Maret 2021        |
| 8   | Pengumuman Kelas Paralel Tetap                                         | Selasa, 6 April 2021                                       | Jumat, 26 Maret 2021<br>(13.00 WIB) |
| 9   | Periode Pembayaran Biaya Kuliah Tahap II (pembayaran biaya SKS)        | Senin-Jumat, 12-23 April 2021                              | Selasa-Selasa, 6-20 April 2021      |
| 10  | Upacara Wisuda I 2020/2021                                             | Sabtu, 17 April 2021                                       | Kamus & Sabtu, 15 & 17 April 2021   |
| 11  | Ujian Tengah Semester                                                  | Senin-Jumat, 3-7 Mei 2021<br>& Rabu-Selasa, 19-20 Mei 2021 | Senin-Jumat, 26 April-7 Mei 2021    |
| 12  | Batas Akhir Penyerahan Nilai UTS                                       | Sabtu, 11 Juni 2021                                        | Jumat, 28 Mei 2021                  |
| 13  | Akhir Perkuliahan (14 minggu)                                          | Jumat, 9 Juli 2021                                         | Sabtu, 3 Juli 2021                  |

Gambar 3.12: Halaman Kalender Akademik

1     ● Daftar Kehadiran

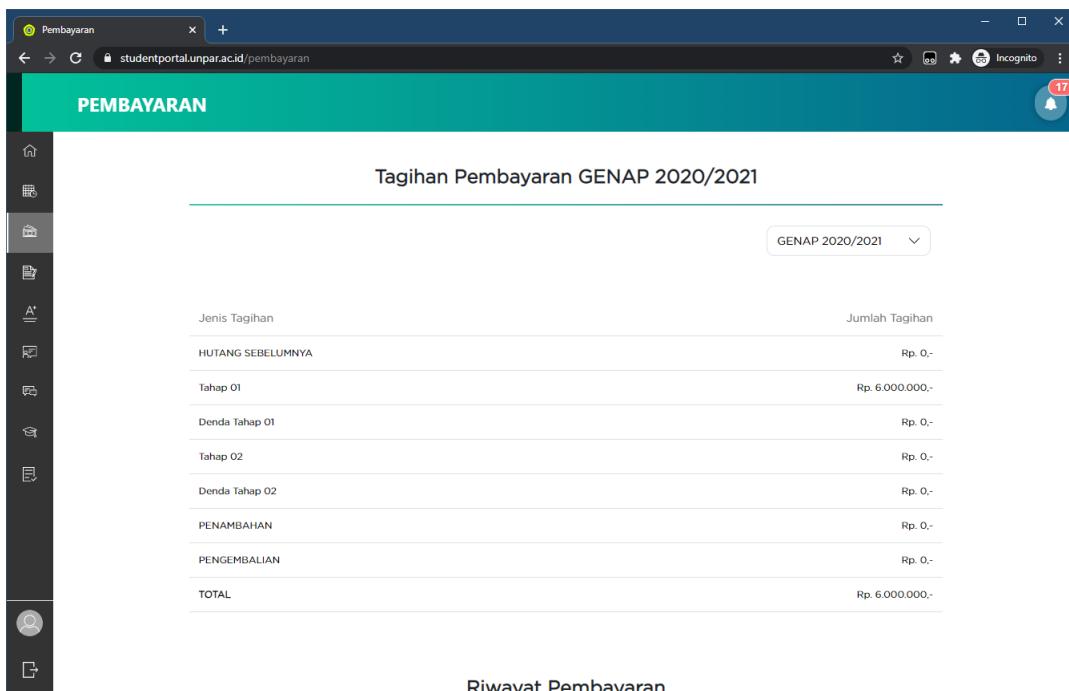
2       Submenu ini menampilkan informasi mengenai daftar kehadiran mahasiswa pada setiap mata  
3       kuliah yang dapat disusun per semester (Gambar 3.13).

| 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.                      | • 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.T.               | • 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                             | • Mariska Tri Adithia, S.Si, M.Sc., PDEng. | • Sesi 1 : Senin , 14:00 -15:00, Ruang Kuliah 9120           | Kuliah              | 1     | 0     | 0    | 0    | 1                             | 100%                    | 100%                   |

Gambar 3.13: Halaman Daftar Kehadiran

4     4.3.1.5 Pembayaran

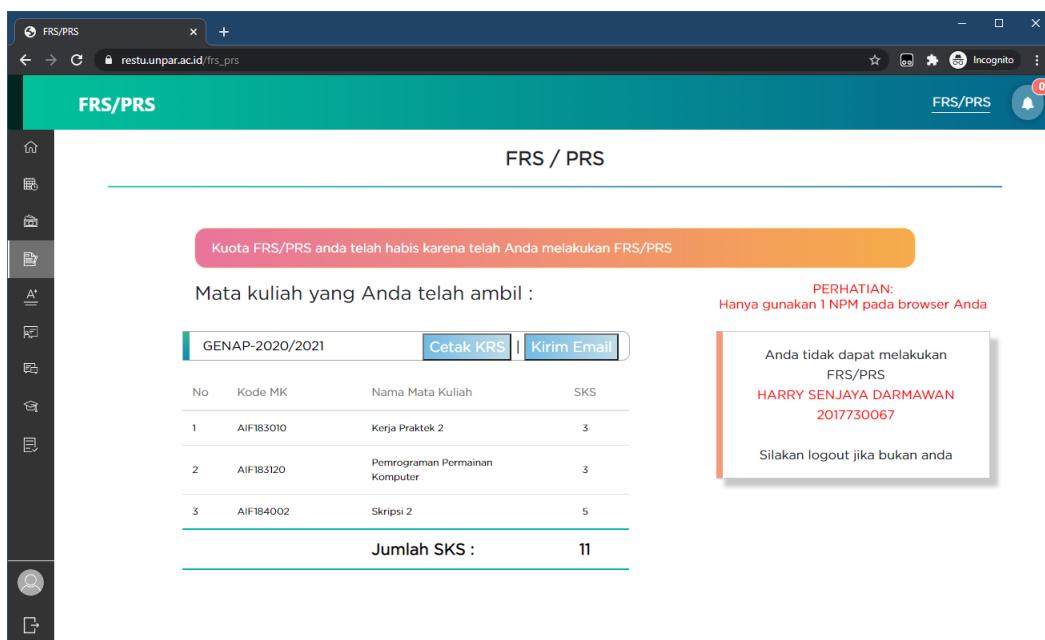
5       Menu ini berfungsi untuk melihat data tagihan pembayaran uang kuliah, riwayat pembayaran, dan  
6       keterangan cara-cara pembayaran uang kuliah yang dapat disusun per semester (Gambar 3.14).



Gambar 3.14: Halaman Pembayaran

### 3.1.6 FRS/PRS

- 2 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).



Gambar 3.15: Tampilan FRS/PRS

### 1 3.1.7 Nilai

2 Menu Nilai terdiri dari beberapa submenu:

- 3 • Nilai per Semester

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

| No | Kode Mata Kuliah | Nama Mata Kuliah         | SKS | Kelas | Nilai                                  | AA | NA | Statistik Hasil Nilai                  |
|----|------------------|--------------------------|-----|-------|----------------------------------------|----|----|----------------------------------------|
| 1  | AIFB3119         | Keamanan Informasi       | 2   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |
| 2  | AIFB3341         | Pola Komputasi Big Data  | 3   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |
| 3  | AIFB3348         | Sistem Kecerdasan Bisnis | 3   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |
| 4  | AIFB4001         | Skripsi 1                | 3   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |
| 5  | AIFB4005         | Komputer dan Masyarakat  | 2   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |
| 6  | AIFB4235         | Layanan Berbasis Web     | 3   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |
| 7  | AIFB4303         | Proyek Sistem Informasi  | 2   | A     | <a href="#">Tampilkan Detail Nilai</a> | -  | -  | <a href="#">Tampilkan Grafik Nilai</a> |

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

Gambar 3.16: Halaman Nilai Per Semester

- 6 • Daftar Perkembangan Studi

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

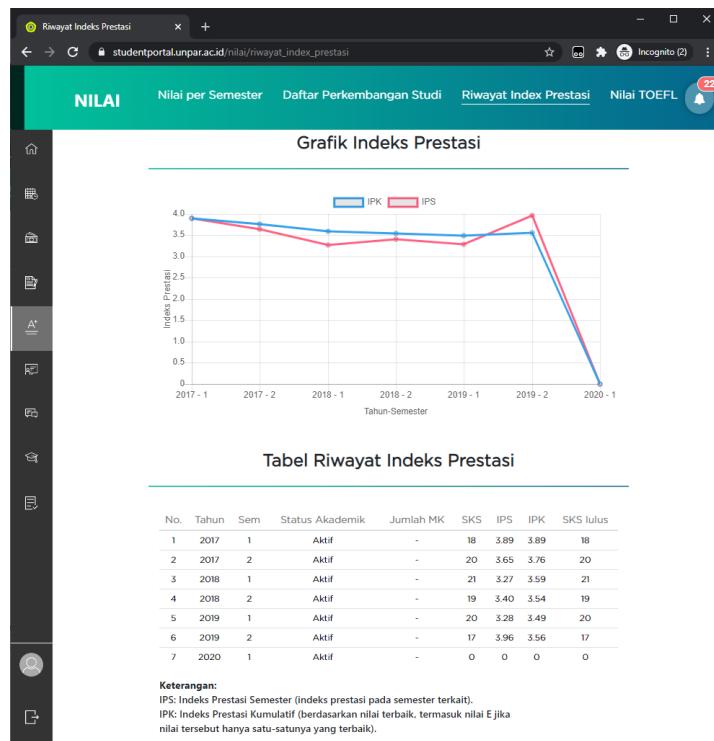
| Kode MK    | Nama MK                        | Nilai | Tahun Sem |
|------------|--------------------------------|-------|-----------|
| Semester 1 |                                |       |           |
| AIF181105  | Pengantar Informatika          | A     | 20171     |
| AIF181101  | Pemodelan untuk Komputasi      | A     | 20172     |
| AIF181103  | Matematika Dasar               | A     | 20172     |
| AIF181105  | Pengantar Informatika          | A     | 20171     |
| AIF181107  | Matematika Diskret             | A     | 20171     |
| MKU180110  | Pendidikan Kewarganegaraan     | A-    | 20181     |
| MKU180120  | Logika                         | A     | 20181     |
| MKU180130  | Bahasa Indonesia               |       |           |
| Semester 2 |                                |       |           |
| 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)

|                                                |                                                                                     |                                                                                                                                                                |                                                                                                            |
|------------------------------------------------|-------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| MKU130011                                      | Estetika                                                                            | B                                                                                                                                                              | 20172                                                                                                      |
| Kode Semester:                                 | Ket. Mk. Bid.                                                                       | Ket. Mt Kuliah:                                                                                                                                                |                                                                                                            |
| 1 = Ganjil, 2 = Genap, 4 = Padat, 6 = Transfer | Peminatan:<br>01 = Teknologi Informasi Bisnis<br>02 = Ilmu Komputer 03 = Telematika | <ul style="list-style-type: none"> <li>• [M] = Mk. Kendali Mutu</li> <li>• [X] = Mk. Disisihkan</li> <li>• ["] = Mk. yang diambil semester (2017-2)</li> </ul> |                                                                                                            |
| Nilai Akhir                                    |                                                                                     | Keterangan IP Mahasiswa                                                                                                                                        |                                                                                                            |
| Nilai                                          |                                                                                     | IPK (-) (15 sks) : 3.56<br>IPS (-) (17 sks) : 3.96                                                                                                             |                                                                                                            |
| Akhir A A- B+ B B- C+ C D E                    |                                                                                     | Jumlah sks                                                                                                                                                     |                                                                                                            |
| Mata Kuliah                                    | 20 4 8 5 1 2 1 0 0                                                                  | Ditempuh : 115 sks<br>Lulus Wajib : 83 sks<br>Lulus Pilihan : 32 sks<br>Lulus Wajib Peminatan : 0 sks<br>Lulus Pilihan Peminatan : 0 sks                       |                                                                                                            |
| SKS                                            | 59 10 21 12 4 6 3 0 0                                                               | Total Lulus : 115 sks<br>Ditempuh Semester ini : 17 sks<br>Diizinkan untuk semester yad. : 24 sks                                                              |                                                                                                            |
| Nilai TOEFL                                    |                                                                                     | Cuti Studi : 0 semester<br>Akhir masa : SEMESTER GENAP Studi : 2023/2024                                                                                       |                                                                                                            |
| No                                             | Tanggal                                                                             | Skor                                                                                                                                                           |                                                                                                            |
| 1                                              | 05-10-2020                                                                          | 540                                                                                                                                                            | Syarat Kelulusan<br>Lulus min. 144 sks terdiri dari : Mk. Wajib + Mk. Pilihan<br>I.P. Lulus minimum : 2.00 |

Gambar 3.18: Halaman Daftar Perkembangan Studi (2)

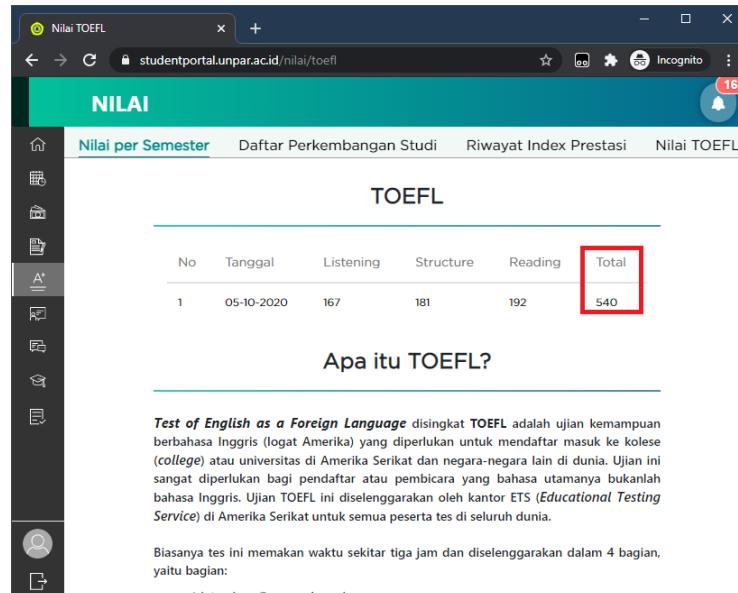
- Riwayat Indeks Prestasi
- Submenu ini menampilkan seluruh riwayat Indeks Prestasi Semester (IPS) dan Indeks Prestasi Kumulatif (IPK) setiap semester mahasiswa (Gambar 3.19).



Gambar 3.19: Halaman Riwayat Indeks Prestasi

#### • Nilai TOEFL

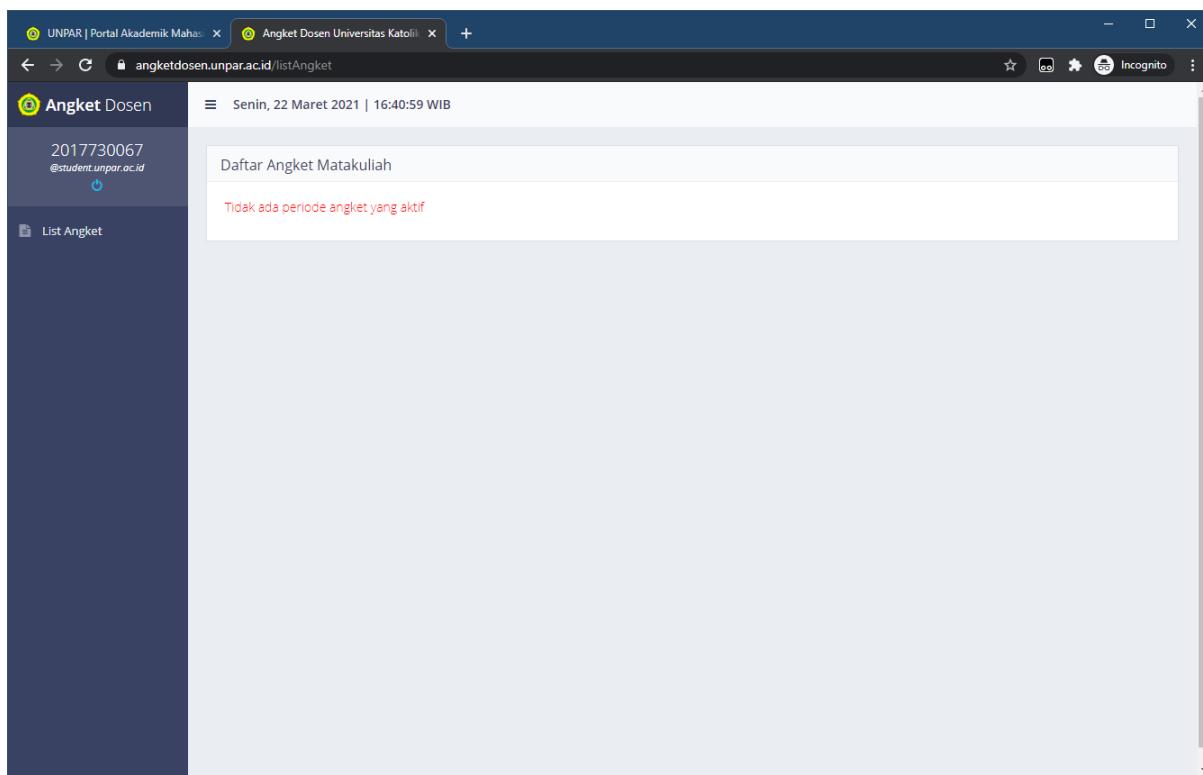
Submenu ini menampilkan seluruh riwayat skor dan detail skor *Test of English as Foreign Language* (TOEFL) yang pernah ditempuh mahasiswa (Gambar 3.20).



Gambar 3.20: Halaman Nilai TOEFL

#### 3.1.8 Angket

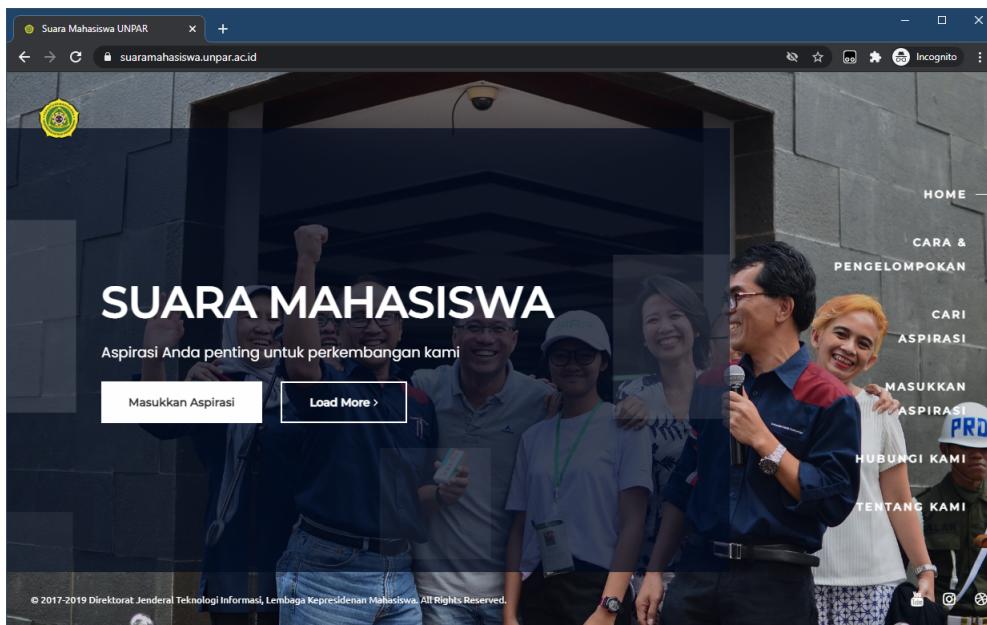
Menu Angket merupakan halaman dimana mahasiswa diminta untuk mengisi angket dari para dosen yang mengajarnya di semester yang sedang ditempuh mahasiswa tersebut (Gambar 3.21).



Gambar 3.21: Halaman Angket

### 3.1.9 Saran & Komentar

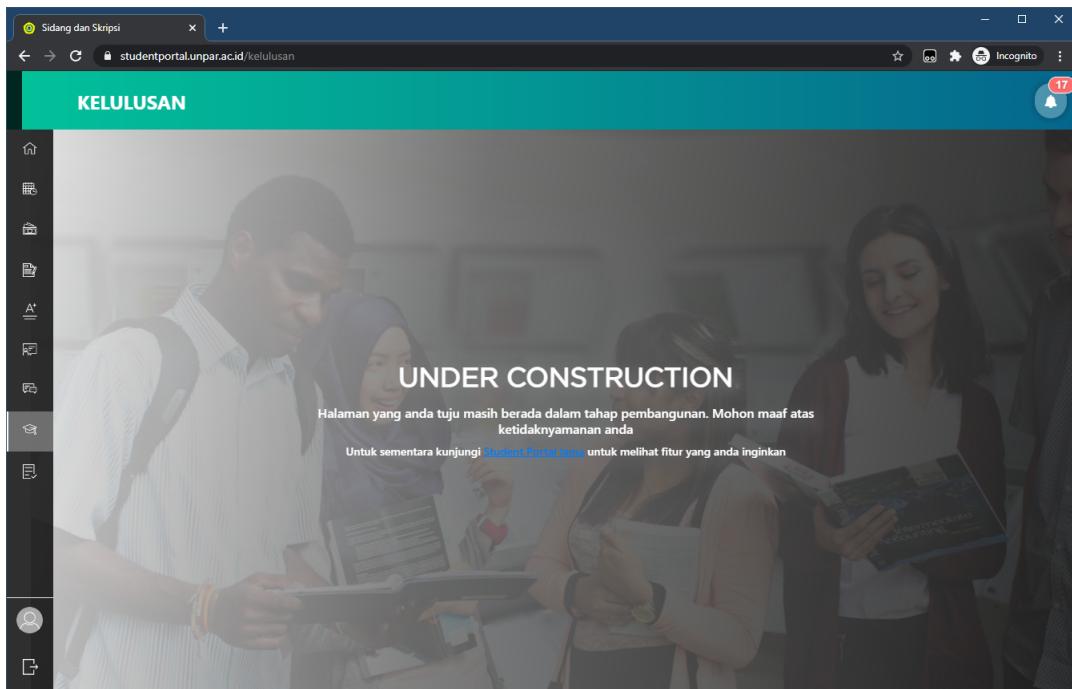
- 1 Menu Saran & Komentar akan membuka halaman <https://suaramahasiswa.unpar.ac.id/> (Gambar 3.22).



Gambar 3.22: Halaman Saran & Komentar

### **1 3.1.10 Kelulusan**

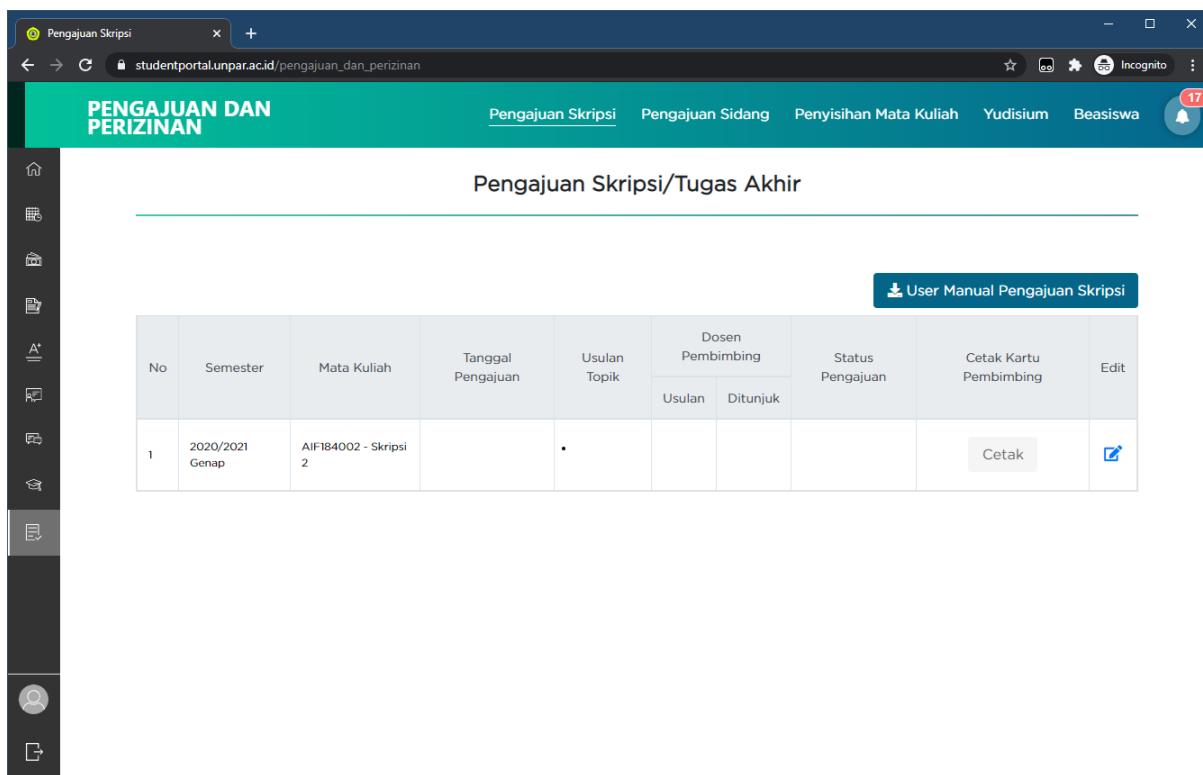
- 2 Menu Kelulusan sedang dalam tahap pembangunan (Gambar 3.23).



Gambar 3.23: Halaman Kelulusan

### **3 3.1.11 Pengajuan**

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



Gambar 3.24: Halaman Pengajuan

## 1 3.2 Analisis SIAKAD

2 *Subbab ini ditulis oleh dosen pembimbing.*

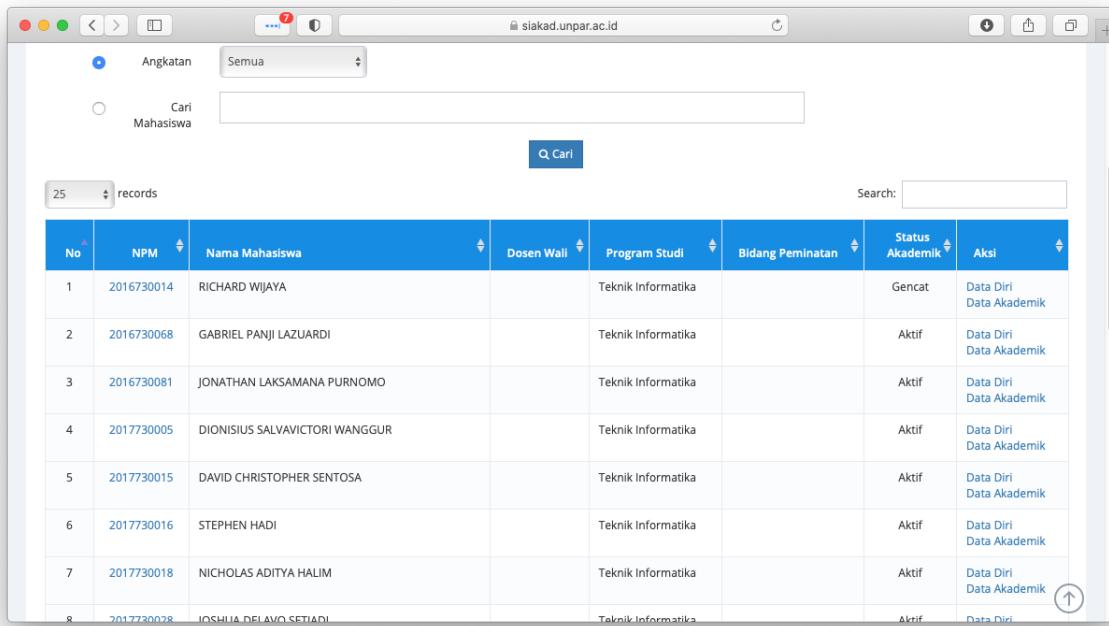
3 SIAKAD adalah sistem informasi yang disediakan oleh Biro Teknologi Informasi kepada staf  
 4 UNPAR, untuk menangani hal-hal yang berkaitan dengan akademik. SIAKAD dapat diakses pada  
 5 alamat <https://siakad.unpar.ac.id> dari lingkungan jaringan UNPAR atau melalui VPN (*Virtual*  
 6 *Private Network*).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

### 9 3.2.1 Login

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

### 14 3.2.2 Mahasiswa

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

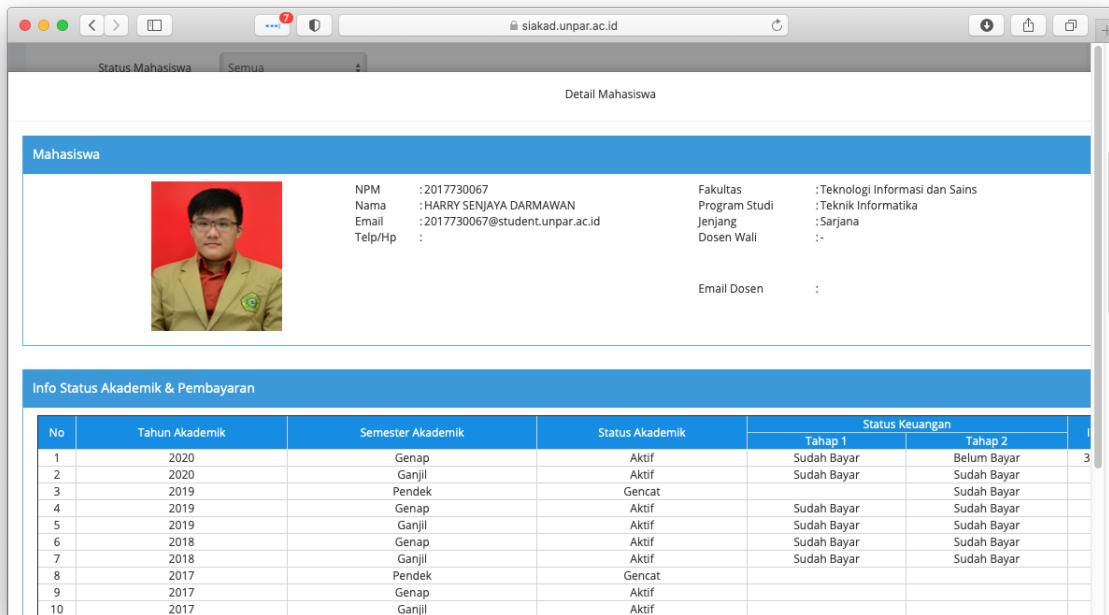


The screenshot shows a table with columns: No, NPM, Nama Mahasiswa, Dosen Wali, Program Studi, Bidang Peminatan, Status Akademik, and Aksi. The data includes:

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

Gambar 3.25: Tangkapan Layar Halaman Daftar Mahasiswa Wali

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



The modal window displays the following details:

**Mahasiswa**

|  |                                                                                                          |                                                                                                                       |
|--|----------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|
|  | NPM : 2017730067<br>Nama : HARRY SENJAYA DARMAWAN<br>Email : 2017730067@student.unpar.ac.id<br>Telp/Hp : | Fakultas : Teknologi Informasi dan Sains<br>Program Studi : Teknik Informatika<br>Jenjang : Sarjana<br>Dosen Wali : - |
|--|----------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|

**Email Dosen** :

**Info Status Akademik & Pembayaran**

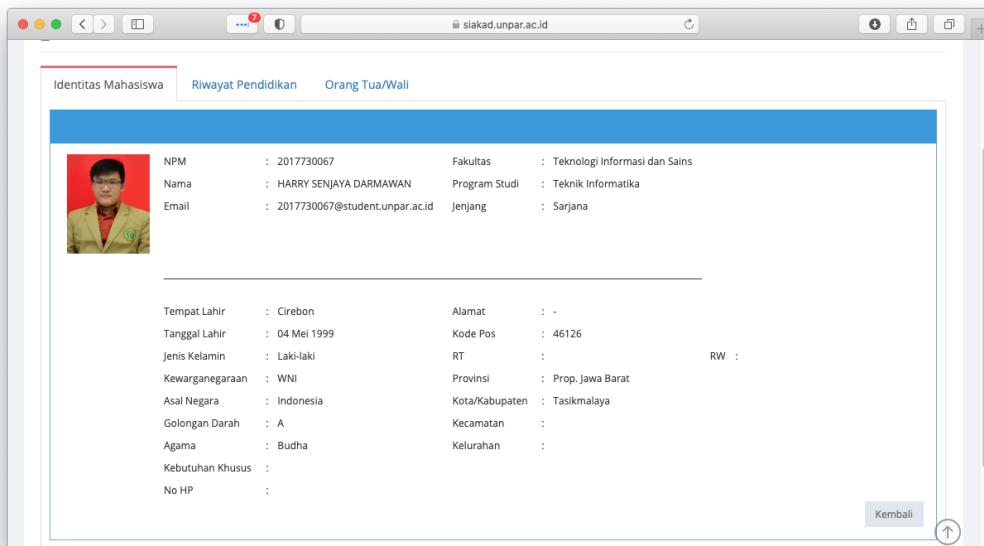
| No | Tahun Akademik | Semester Akademik | Status Akademik | Status Keuangan |             |
|----|----------------|-------------------|-----------------|-----------------|-------------|
|    |                |                   |                 | Tahap 1         | Tahap 2     |
| 1  | 2020           | 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

## 1 Data Diri

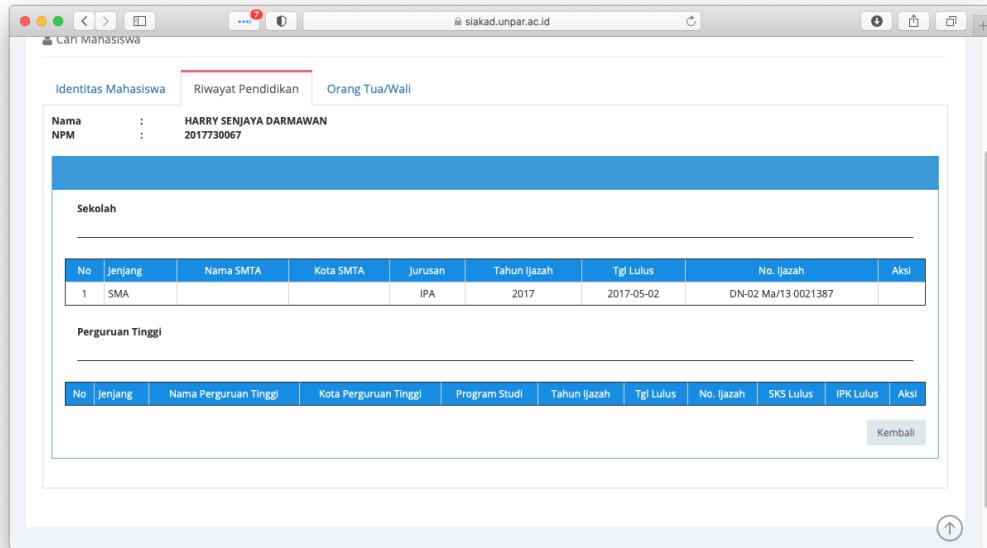
2 Data diri mahasiswa ditampilkan pada halaman baru yang terdiri dari beberapa tab. Halaman ini berisi detail data diri / pribadi mahasiswa yang tidak terkait data akademik di UNPAR. Halaman ini terdiri dari tiga tab, yaitu Identitas Mahasiswa, Riwayat Pendidikan, dan Orang Tua/Wali.

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



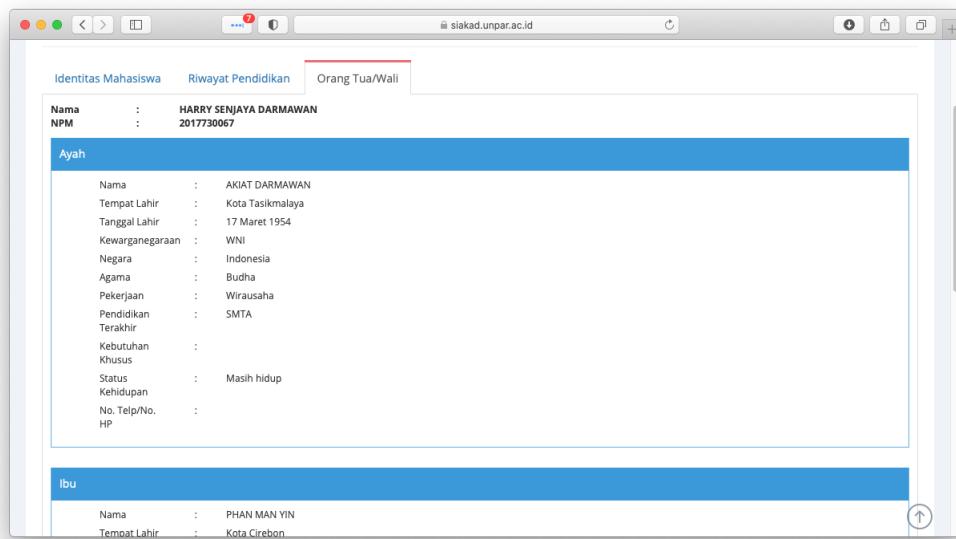
Gambar 3.27: Tangkapan Layar Tab Identitas Mahasiswa

7 **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

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

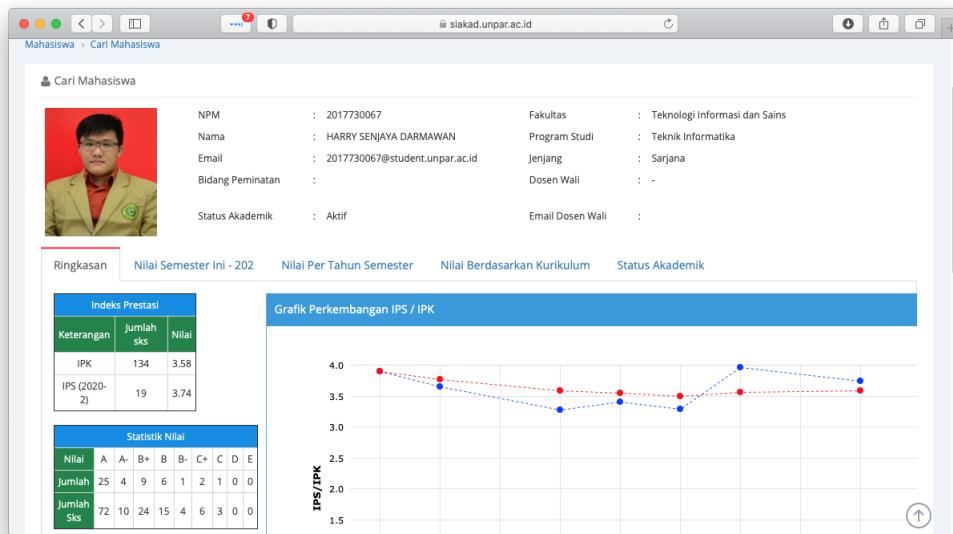


Gambar 3.29: Tangkapan Layar Tab Orang Tua/Wali

### 3 Data Akademik

- 4 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.

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



Gambar 3.30: Tangkapan Layar Tab Ringkasan Akademik

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

| No | Kode MK   | Nama Matakuliah                | sks | Kelas     | TP1 | TP2 | TP3 | TP4 | TP5 | TP6 | TP7 | TP8 | TP9 | TP10 | TP11 | TP12 | TP13 | TP14 | TP15 | TP16 | TP17 | TP18 | TP19 | TP20 | UTS | UAS | Kelompok Ujian | AA | NA |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|----|-----------|--------------------------------|-----|-----------|-----|-----|-----|-----|-----|-----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|-----|-----|----------------|----|----|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|--|
| 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.

Gambar 3.31: Tangkapan Layar Tab Nilai Semester Ini

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

**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 |
| <b>Semester Genap</b> |           |                                |     |       |                |     |     |     |     |     |     |     |     |      |      |      |      |      |      |                |    |    |      |      |
| 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    |
|                       |           | <b>Total sks</b>               | 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 |
| <b>Semester Ganjil</b> |           |                                |     |       |                |     |     |     |     |     |     |     |     |      |      |      |      |      |      |                |    |    |      |      |      |
| 1                      | AIF183119 | Keamanan Informasi             | 2   | A     |                | 65  |     |     |     |     |     |     |     |      |      |      |      |      |      |                |    |    | 95   | 81   | 81   |
| 2                      | AIF183341 | Pola Komputasi Big Data Sistem | 3   | A     | 44             | 90  | 70  | 70  | 70  | 66  |     |     |     |      |      |      |      |      |      |                |    | 73 | 74   | 71   |      |

Gambar 3.32: Tangkapan Layar Tab Nilai Per Tahun Semester

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

**Komposisi sks :**

| Status tempuh | Rincian           | Jumlah sks | Nilai Akhir |           |           |           |           |           |           |           |   |   |    |    |   |   |   |    |   |   |   |   |
|---------------|-------------------|------------|-------------|-----------|-----------|-----------|-----------|-----------|-----------|-----------|---|---|----|----|---|---|---|----|---|---|---|---|
|               |                   |            | 2017-2018   | 2018-2019 | 2019-2020 | 2020-2021 | 2021-2022 | 2022-2023 | 2023-2024 | 2024-2025 |   |   |    |    |   |   |   |    |   |   |   |   |
|               |                   |            | 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        |             |           |           |           |           |           |           |           |   |   |    |    |   |   |   |    |   |   |   |   |

**Nilai Akhir**

| 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 |
| <b>Semester 1</b> |                       |                           |                  |                       |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |
| <b>Wajib</b>      |                       |                           |                  |                       |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |
| AIF131105         | Pengantar Informatika | 3                         | A                |                       |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |
|                   | AIF181101             | Pemodelan untuk Komputasi | 3                |                       |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |
|                   |                       | AIF181103                 | Matematika Dasar | 4                     | A |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |
|                   |                       |                           | AIF181105        | Pengantar Informatika | 2 |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |           |   |   |

Gambar 3.33: Tangkapan Layar Tab Nilai Berdasarkan Kurikulum

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

| 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

- 4 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 3.2.4 Perkuliahinan

- 6 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 3.2.5 Ujian

- 8 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 3.2.6 Nilai

- 10 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 3.2.7 Evaluasi

- 12 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 with the following details:

- Universitas Katolik Parahyangan** logo and address: Jalan Ciwideyut 94 Bandung 40141, telp. : +62.22.2030918-28 ext. 100401, 100422, fax: (022)203 1110, Telp.(022) 203 2655;(022) 204 2004
- Student Information:**
  - Nama : HARRY SENJAYA DARMAWAN
  - NPM : 2017730067
  - Status : Aktif
  - Dosen Wali : [ ]
  - Fakultas : Teknologi Informasi dan Sains
  - Program Studi : Teknik Informatika
  - Email : [ ]
  - Bidang Peminatan : [ ]
- Academic Record Grids:**
  - Top Grid (Status Tempuh):** Shows the number of students per grade (1, 2, 4, 5, 6) for each year from 2017/2018 to 2025/2026. For example, in 2017/2018, there were 88 students in grade 1.
  - Bottom Grid (Nilai Akhir):** Shows the final grade (A, B, C, D, E) for each course across the years. Courses listed include Pengantar Informatika, Matematika Dasar, Logika, and Bahasa Indonesia.

Gambar 3.35: Tangkapan Layar PDF Daftar Perkembangan Studi

### 1 3.2.8 Skripsi

2 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 3 3.2.9 Sidang

4 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 5 3.2.10 Kelulusan

6 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

### 7 3.2.11 Pengumuman

8 Bagian ini tidak dijelaskan karena tidak berkaitan dengan informasi mahasiswa.

## 9 3.3 Data yang Dibutuhkan untuk *Screensaver*

### 10 3.3.1 Portal Akademik Mahasiswa

#### 11 Halaman Utama

12 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

1 kelas "img-fluid fotoProfil", sehingga kueri css yang dihasilkan adalah "img[class=img-fluid  
2 fotoProfil]". Terdapat beberapa perubahan (Kode 3.2) yang perlu dilakukan terhadap skripsi  
3 Andrianto Sugiarto [5]:

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

Kode 3.2: Perubahan Implementasi Jsoup Halaman Utama

```

14
15 1 @@ -96,24 +96,22 @@ public class Scraper {
16 2 Connection connection = Jsoup.connect(HOME_URL);
17 3 connection.cookie("ci_session", phpsessid);
18 4 connection.timeout(0);
19 5 - connection.validateTLCertificates(false);
20 6 connection.method(Connection.Method.GET);
21 7 Response resp = connection.execute();
22 8 Document doc = resp.parse();
23 9 String nama = doc.select("div[class=namaUser d-none d-lg-block mr-3]").text();
24 0 mhs.setNama(nama.substring(0, nama.indexOf(mhs.getEmailAddress())));
25 1 Element photo = doc.select("img[class=img-fluid fotoProfil]").first();
26 2 String photoPath = photo.attr("src");
27 3 mhs.setPhotoPath(photoPath);
28 4 - connection = Jsoup.connect(FRSPRS_URL);
29 5 + connection = Jsoup.connect(NILAI_URL);
30 6 connection.cookie("ci_session", phpsessid);
31 7 connection.timeout(0);
32 8 - connection.validateTLCertificates(false);
33 9 connection.method(Connection.Method.GET);
34 0 resp = connection.execute();
35 1 doc = resp.parse();
36 2 - String curr_sem = doc.select(".custom-selectContent span").text();
37 3 - String[] sem_set = parseSemester(curr_sem);
38 4 + Elements curr_sem = doc.select("select[class=custom-select mr-3]");
39 5 + String[] sem_set = parseSemester(curr_sem.first().child(curr_sem.first().childrenSize() - 1).text());
40 6 TahunSemester currTahunSemester = new TahunSemester(Integer.parseInt(sem_set[0]),
41 7 Semester.fromString(sem_set[1]));
42 8 return currTahunSemester;
43 9 @@ -214,7 +212,7 @@ public class Scraper {
44 0 }
45 1
46 2 public String[] parseSemester(String sem_raw) {
47 3 - String[] sem_set = sem_raw.split("/")[0].split("-");
48 4 + String[] sem_set = sem_raw.split("/")[0].split(" ");
49 5 return new String[]{sem_set[1].trim(), sem_set[0].trim()};
50 6 }

```

## 52 Halaman Profil

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

## 1 Halaman Daftar Perkembangan Studi

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

- 10     • Mengambil data nilai berdasarkan tahun dan semester dengan mencari elemen "select" yang  
  11       memiliki id "dropdownSemester", dan memiliki kelas "custom-select mr-3" sehingga kueri  
  12       css yang dihasilkan adalah "select#dropdownSemester.custom-select.mr-3".  
  13     • Dikarenakan perlunya melakukan koneksi berkali-kali sebanyak semester yang telah ditempuh  
  14       mahasiswa, sehingga dibutuhkan waktu yang tidak sebentar. Karena pada halaman nilai  
  15       tidak dapat menampilkan seluruh semester seperti Portal Akademik Mahasiswa yang lama,  
  16       sehingga untuk mengatasi masalah ini dibuat menjadi paralel. Untuk itu dibuat kelas yang  
  17       mengimplementasikan kelas *interface Runnable*, yaitu kelas *MultipleRequest*. Kelas inilah  
  18       yang melakukan koneksi ke setiap semester yang telah ditempuh mahasiswa, dan mengambil  
  19       data-data tersebut.  
 20   Namun, terdapat beberapa perubahan (Kode 3.3) yang perlu dilakukan:  
 21   1. Menghapus pemanggilan fungsi validateTLSCertificates() dikarenakan sudah *deprecated* [6].  
 22   2. Indeks yang digunakan untuk melakukan kueri css berubah, sehingga untuk mengantisipasi  
  23       tersebut, indeks yang digunakan menjadi *size* dari kueri css dikurangi 1.

Kode 3.3: Perubahan Implementasi Jsoup Halaman Daftar Perkembangan Studi

```

24
25 1 @@ -50,7 +50,7 @@ public class MultipleRequest implements Runnable {
26 2 Connection.Response resp = connection.execute();
27 3 Document doc = resp.parse();
28 4
29 5 - Element script = doc.select("script").get(10);
30 6 + Element script = doc.select("script").get(doc.select("script").size()-1);
31 7 String scriptDataMataKuliah = script.html().substring(script.html().indexOf("var data_mata_kuliah = []"), script.
32 html().indexOf("var data_angket = []"));
33 8 engine.eval(scriptDataMataKuliah);
34 9 ScriptObjectMirror dataMataKuliah = (ScriptObjectMirror) engine.get("data_mata_kuliah");
35 0
36 1 @@ -131,7 +131,6 @@ public class Scraper {
37 2 Connection connection = Jsoup.connect(NILAI_URL);
38 3 connection.cookie("ci_session", phpsessid);
39 4 connection.timeout(0);
40 5 - connection.validateTLSCertificates(false);
41 6 connection.method(Connection.Method.POST);
42 7 Response resp = connection.execute();
43 8 Document doc = resp.parse();

```

## 45 Halaman TOEFL

- 46 Pada Halaman TOEFL (Gambar 3.20 dengan kotak merah), riwayat skor TOEFL dapat diambil  
 47 dengan mencari elemen "table" yang memiliki elemen "tbody" didalamnya, serta memiliki elemen  
 48 "tr" didalam elemen "tbody". Terdapat beberapa perubahan yang terjadi pada halaman TOEFL  
 49 semenjak skripsi Andrianto Sugiarto [5], yang mengakibatkan perlunya perubahan (Kode 3.4)  
 50 terhadap implementasi jsoup:

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

Kode 3.4: Perubahan Implementasi Jsoup TOEFL

```

3 1 @@ -174,7 +174,6 @@ public class Scraper {
5 2 Connection connection = Jsoup.connect(TOEFL_URL);
6 3 connection.cookie("ci_session", phpsessid);
7 4 connection.timeout(0);
8 5 - connection.validateTLCertificates(false);
9 6 connection.method(Connection.Method.POST);
10 7 Response resp = connection.execute();
11 8 Document doc = resp.parse();
12 9 @@ -183,45 +182,7 @@ public class Scraper {
13 0 for (int i = 0; i < nilaiTOEFL.size(); i++) {
14 1 Element nilai = nilaiTOEFL.get(i).select("td").get(5);
15 2 Element tgl_toefl = nilaiTOEFL.get(i).select("td").get(1);
16 3 - String[] tanggal = tgl_toefl.text().split(" ");
17 4 - switch (tanggal[1].toLowerCase()) {
18 5 - case "januari":
19 6 - tanggal[1] = "1";
20 7 - break;
21 8 - case "februari":
22 9 - tanggal[1] = "2";
23 0 - break;
24 1 - case "maret":
25 2 - tanggal[1] = "3";
26 3 - break;
27 4 - case "april":
28 5 - tanggal[1] = "4";
29 6 - break;
30 7 - case "mei":
31 8 - tanggal[1] = "5";
32 9 - break;
33 0 - case "juni":
34 1 - tanggal[1] = "6";
35 2 - break;
36 3 - case "juli":
37 4 - tanggal[1] = "7";
38 5 - break;
39 6 - case "agustus":
40 7 - tanggal[1] = "8";
41 8 - break;
42 9 - case "september":
43 0 - tanggal[1] = "9";
44 1 - break;
45 2 - case "oktober":
46 3 - tanggal[1] = "10";
47 4 - break;
48 5 - case "november":
49 6 - tanggal[1] = "11";
50 7 - break;
51 8 - case "desember":
52 9 - tanggal[1] = "12";
53 0 - break;
54 1 - }
55 2 + String[] tanggal = tgl_toefl.text().split("-");
56 3
57 4 LocalDate localDate = LocalDate.of(Integer.parseInt(tanggal[2]), Integer.parseInt(tanggal[1]),
58 5 Integer.parseInt(tanggal[0]));
59 6

```

### 60    3.3.2 SIAKAD

61    *Subbab ini ditulis oleh dosen pembimbing.*

### 62    Mendapatkan daftar mahasiswa

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

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

#### 4 Mendapatkan detail mahasiswa

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

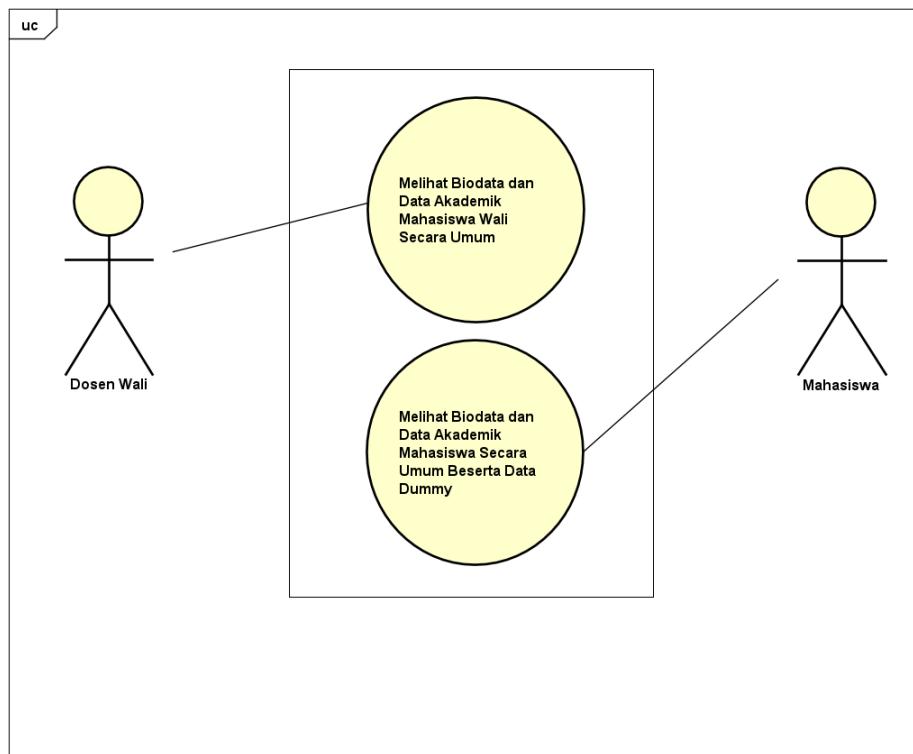
- 6 1. **Nilai Per Tahun Semester** berisikan tabel nilai yang dibagi per tahun semester. Data dari  
7 halaman inilah yang dipakai untuk mendapatkan nilai mahasiswa, mencakup: kode dan nama  
8 mata kuliah, SKS, kelas, TP1 s.d. TP20, UTS, UAS, AA, dan NA.
  - 9 2. **Nilai Berdasarkan Kurikulum** berisikan tabel nilai yang dibagi berdasarkan linimasa  
10 kurikulum. Data ini kurang cocok dipakai karena tidak mengandung nilai tugas dan ujian,  
11 hanya nilai akhir saja.
  - 12 3. **Evaluasi** berisikan tabel nilai lengkap dalam bentuk Daftar Perkembangan Studi dengan  
13 format PDF. Data ini sulit diekstraksi karena tidak dalam bentuk HTML.  
14 Kesimpulannya, riwayat nilai diambil dari halaman “Nilai Per Tahun Semester”.
- 15 **Data Diri** Pada SIAKAD, data diri dapat diambil dari halaman “Data Diri”, tab identitas  
16 mahasiswa. Foto tersedia dalam bentuk Data URL<sup>1</sup> yang di-*encode* dalam format *base64*. Data-data  
17 diri lainnya dapat diambil dengan cukup sederhana dari berbagai elemen yang ada di tab tersebut.

---

<sup>1</sup>[https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics\\_of\\_HTTP/Data\\_URIs](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Data_URIs)

## <sup>1</sup> 3.4 Analisis Sistem *Screensaver*

### <sup>2</sup> 3.4.1 *Use Case Screensaver*



Gambar 3.36: Diagram *Use Case Screensaver*

- <sup>3</sup> Pada diagram *use case screensaver* (Gambar 3.36) terdapat 2 fitur utama dari *screensaver* yaitu,
- <sup>4</sup> melihat biodata dan data akademik mahasiswa wali secara umum bagi dosen, serta melihat biodata
- <sup>5</sup> dan data akademik mahasiswa secara umum beserta data dummy bagi mahasiswa. Dimana biodata
- <sup>6</sup> yang dimaksud adalah nama, angkatan, tanggal lahir. Serta data akademik yang dimaksud adalah
- <sup>7</sup> status, email, nilai TOEFL, IPS, IPK, SKS lulus, SKS tempuh mahasiswa.

### <sup>8</sup> Skenario *Use Case*

#### <sup>9</sup> 1. Melihat Biodata dan Data Akademik Mahasiswa Wali Secara Umum

- <sup>10</sup> • Nama: Melihat Biodata dan Data Akademik Mahasiswa Wali Secara Umum
- <sup>11</sup> • Aktor: Dosen
- <sup>12</sup> • Deskripsi: Menjalankan *screensaver* yang menampilkan biodata dan data akademik
- <sup>13</sup> mahasiswa wali secara umum.
- <sup>14</sup> • Kondisi awal: Komputer tidak digunakan selama beberapa saat.
- <sup>15</sup> • Kondisi Akhir: *Screensaver* berjalan dengan menampilkan biodata dan data akademik
- <sup>16</sup> mahasiswa wali secara umum.
- <sup>17</sup> • Skenario utama:

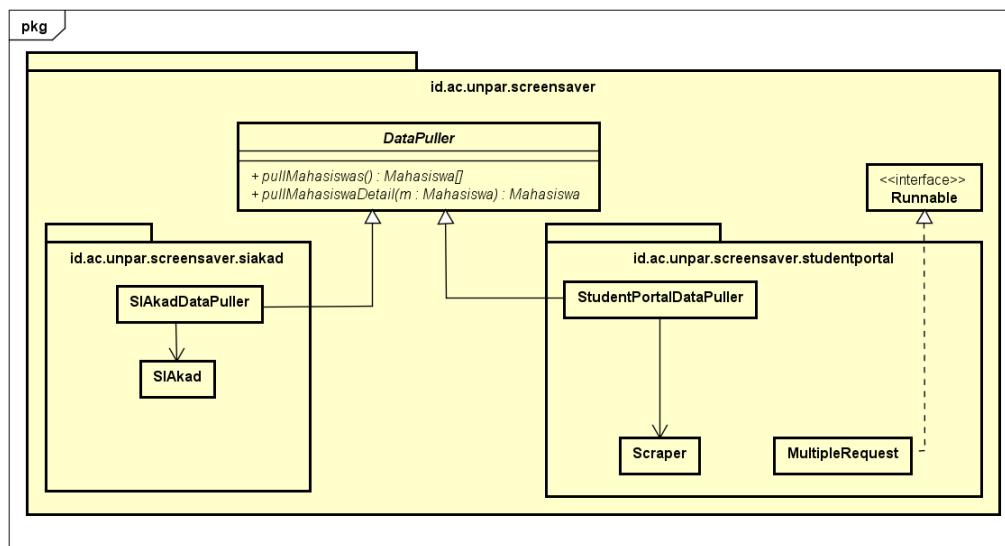
| No | Aksi Aktor                                             | Reaksi Sistem                                                                                        |
|----|--------------------------------------------------------|------------------------------------------------------------------------------------------------------|
| 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*

- 16 Gambar 3.37 merupakan perancangan kelas *screensaver*. Terdapat sebuah *abstract class* yang  
17 bernama *DataPuller* yang memiliki 2 buah *method* yaitu:  
18 • public abstract Mahasiswa[] pullMahasiswa()

- 1 Berfungsi untuk mengambil seluruh daftar mahasiswa.
- 2 • **public abstract Mahasiswa pullMahasiswaDetail(Mahasiswa m)**
- 3 Berfungsi untuk mengambil detail lebih lanjut mengenai data mahasiswa tersebut.
- 4 Kelas **DataPuller** diturunkan ke 2 buah kelas yaitu **SIAkadDataPuller**, dan **StudentPortalDataPuller**,
- 5 dimana kedua kelas tersebut berfungsi sebagai pintu masuk dalam mengambil data mahasiswa.
- 6 Data mahasiswa pada halaman SIAKAD diambil oleh kelas yang bernama **SIAkad**, sedangkan data
- 7 mahasiswa pada halaman Portal Akademik Mahasiswa diambil oleh kelas yang bernama **Scraper**.
- 8 Penjelasan kelas **MultipleRequest** terdapat pada bagian [3.3.1](#).

1

## BAB 4

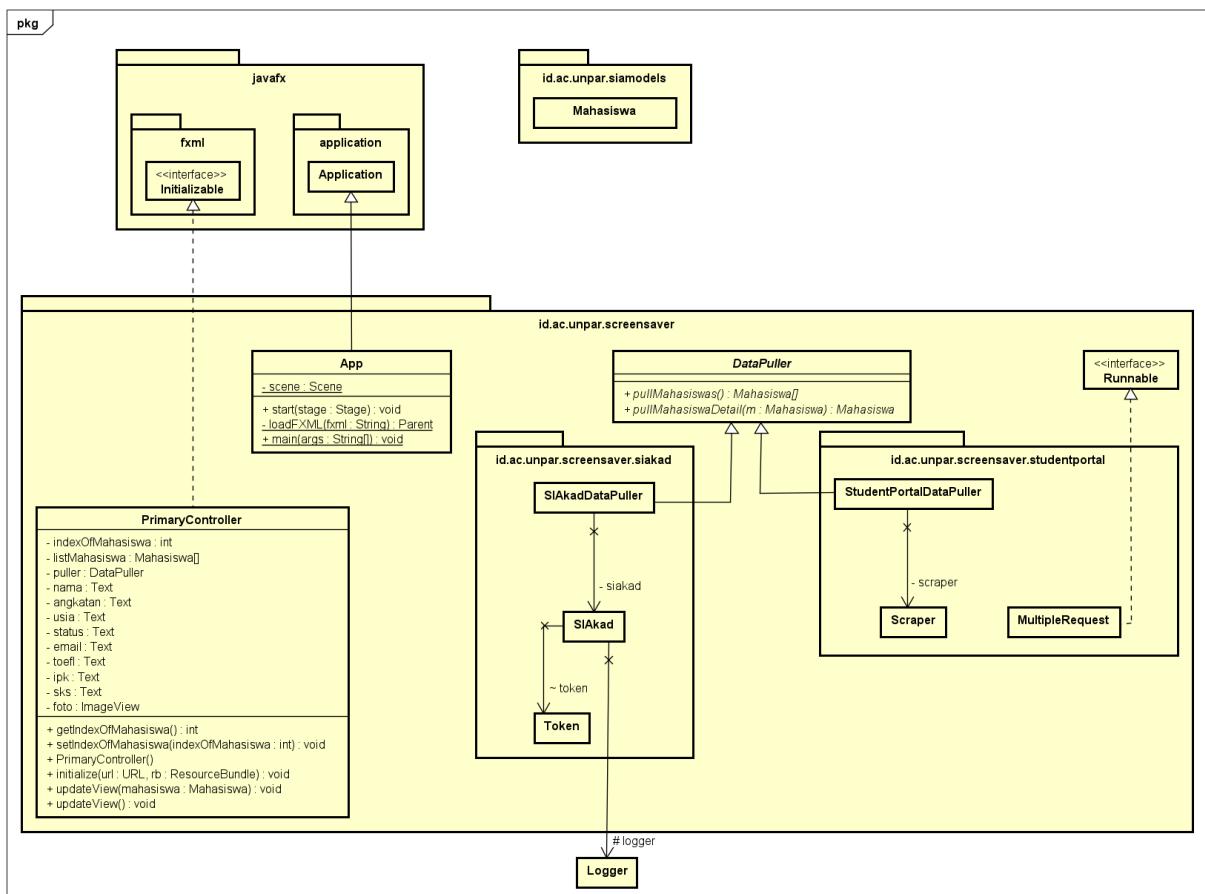
2

## PERANCANGAN

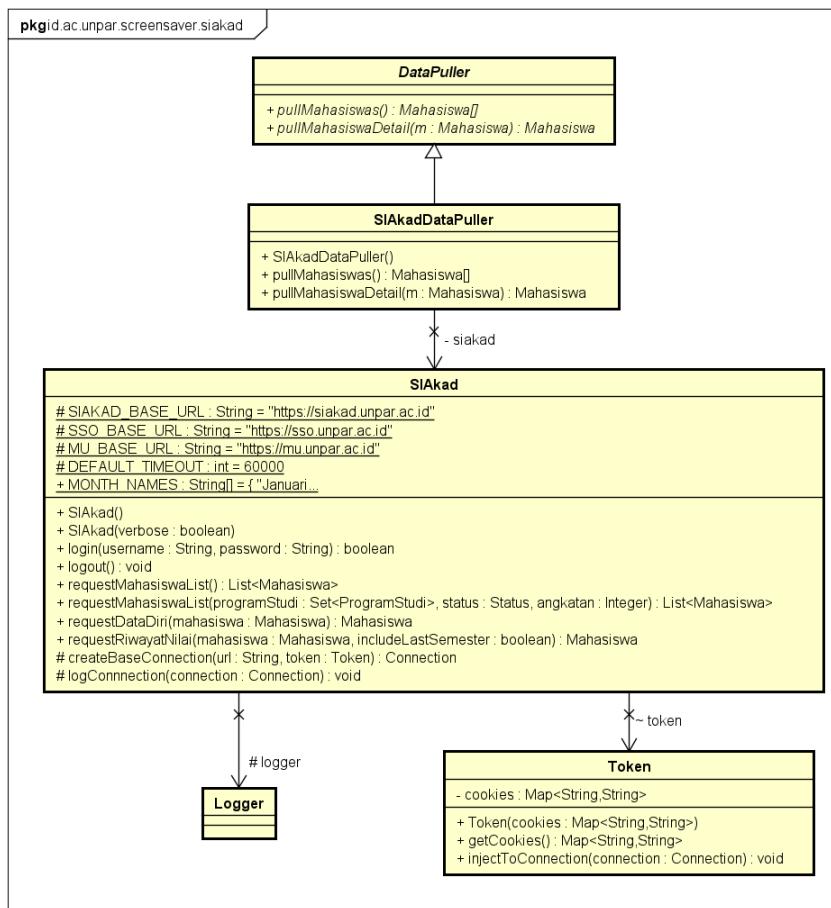
- 3 Pada bab ini akan dijelaskan mengenai perancangan kelas beserta deskripsi kelas dan fungsinya,  
 4 serta perancangan antarmuka.

### 5 4.1 Perancangan Kelas

- 6 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.  
 9 Penjelasan mengenai kelas, *method*, dan atribut JavaFX serta FXML terdapat pada bagian 2.2.  
 10 Penjelasan mengenai kelas, *method*, dan atribut SIAModels terdapat pada bagian 2.3.



Gambar 4.1: Diagram Kelas Keseluruhan



Gambar 4.2: Diagram Kelas SIAKAD

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

### 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.

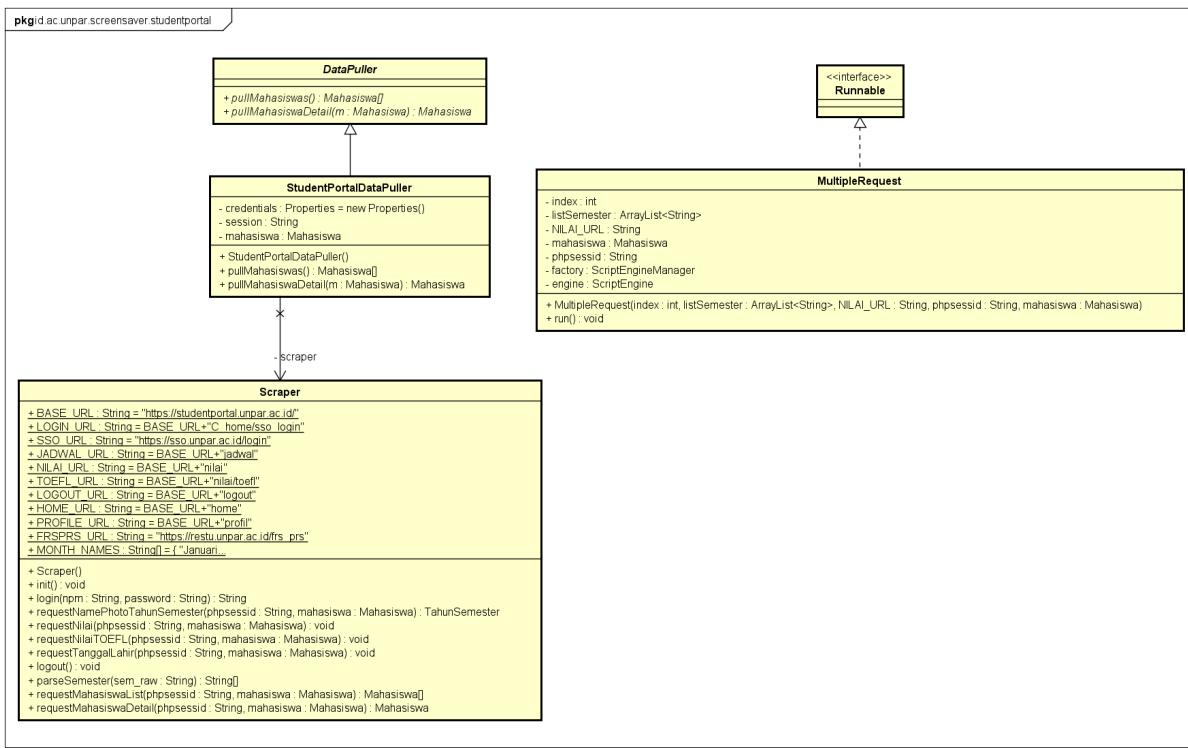
**Kembalian:** hierarki objek dari dokumen FXML.

- **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



Gambar 4.3: Diagram Kelas Studentportal

ini adalah sebagai berikut:

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

- 1     • `private ImageView foto:` menyimpan objek `ImageView` yang digunakan untuk foto  
2       mahasiswa.

3     Method yang dimiliki kelas ini adalah sebagai berikut:

- 4     • `public int getIndexOfMahasiswa()`  
5       Berfungsi untuk mengambil indeks mahasiswa.  
6       **Kembalian:** indeks mahasiswa.
- 7     • `public void setIndexOfMahasiswa(int indexOfMahasiswa)`  
8       Berfungsi untuk menyimpan indeks mahasiswa.

#### 9     **Parameter:**

10    – `indexOfMahasiswa:` indeks mahasiswa.

- 11    • `public PrimaryController()`  
12       Berfungsi sebagai *constructor* kelas `PrimaryController`.
- 13    • `public void initialize(URL url, ResourceBundle rb)`  
14       Berfungsi untuk menginisialisasi pengontrol setelah elemen akarnya diproses sepenuhnya.  
15       Method ini juga berfungsi untuk mengambil *list* mahasiswa beserta dengan keterangan  
16       detil mahasiswa. **Parameter:**  
17       – `url:` Lokasi yang digunakan untuk menyelesaikan jalur relatif untuk objek `root`,  
18       atau `null` jika lokasi tidak diketahui. item `rb:` *resource* yang digunakan untuk  
19       melokalisasi objek `root`, atau `null` jika objek `root` tidak dilokalkan.

- 20    • `public void updateView(Mahasiswa mahasiswa)`  
21       Berfungsi untuk memperbarui tampilan *screensaver* dengan data mahasiswa yang menjadi  
22       parameter.

#### 23     **Parameter:**

24    – `mahasiswa:` objek mahasiswa.

- 25    • `public void updateView()`  
26       Berfungsi untuk memperbarui tampilan *screensaver* ketika koneksi internet tidak berjalan  
27       dengan normal.

### 29     3. DataPuller

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

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

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

#### 37     1. StudentPortalDataPuller

38     Kelas ini merupakan turunan dari kelas `DataPuller` 3. Kelas ini memiliki fungsi untuk  
39       mengambil *npm* dan *password* mahasiswa yang disimpan dalam sebuah *file*, dan kemudian  
40       memanggil *method* pada kelas `Scrapers`. Atribut yang dimiliki kelas ini adalah sebagai berikut:

- 41     • `private Scrapers scraper:` menyimpan objek `Scrapers`.
- 42     • `private final Properties credentials:` mengakses *file* yang berisi *npm* dan *pass-*

- 1                   *word* mahasiswa.
- 2     • **private Mahasiswa mahasiswa:** menyimpan objek mahasiswa yang akan ditambahkan  
3        datanya dari pengambilan data di Portal Akademik Mahasiswa.
- 4     • **private String session:** menyimpan *session* yang dapat digunakan untuk mengakses  
5        menu di Portal Akademik Mahasiswa.

6     *Method* yang dimiliki kelas ini adalah sebagai berikut:

- 7     • **public StudentPortalDataPuller()**  
8        Berfungsi sebagai *constructor* kelas StudentPortalDataPuller.
- 9     • **public Mahasiswa[] pullMahasiswas()**  
10      Berfungsi untuk memanggil *method* requestMahasiswaList pada kelas Scraper (2).  
11      **Kembalian:** *array* yang berisi seluruh mahasiswa yang memiliki dosen wali yang sama  
12        dengan yang melakukan *login* (dalam implementasi menggunakan Portal Akademik  
13        Mahasiswa, mahasiswa yang diambil adalah hanya mahasiswa yang melakukan *login*).  
14     • **public Mahasiswa pullMahasiswaDetail()**  
15      Berfungsi untuk memanggil *method* requestMahasiswaDetail pada kelas Scraper (2).  
16      **Kembalian:** objek mahasiswa yang telah ditambahkan datanya dari pengambilan data  
17        di Portal Akademik Mahasiswa.

18     2. Scraper

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

- 21     • **public static final String BASE\_URL:** menyimpan *url* utama Portal Akademik Ma-  
22        hasiswa.
- 23     • **public static final String LOGIN\_URL:** menyimpan *url* halaman *login* Portal Aka-  
24        demik Mahasiswa.
- 25     • **public static final String SSO\_URL:** menyimpan *url* halaman *login Single Sign*  
26        *On(SSO)* UNPAR.
- 27     • **public static final String JADWAL\_URL:** menyimpan *url* halaman jadwal Portal  
28        Akademik Mahasiswa.
- 29     • **public static final String NILAI\_URL:** menyimpan *url* halaman nilai Portal Aka-  
30        demik Mahasiswa.
- 31     • **public static final String TOEFL\_URL:** menyimpan *url* halaman nilai TOEFL Por-  
32        tal Akademik Mahasiswa.
- 33     • **public static final String LOGOUT\_URL:** menyimpan *url* untuk melakukan *logout*.
- 34     • **public static final String HOME\_URL:** menyimpan *url* halaman utama Portal Aka-  
35        demik Mahasiswa setelah melakukan *login*.
- 36     • **public static final String PROFILE\_URL:** menyimpan *url* halaman profil mahasis-  
37        wa Portal Akademik Mahasiswa.
- 38     • **public static final String FRSPRS\_URL:** menyimpan *url* halaman FRS/PRS Portal  
39        Akademik Mahasiswa.
- 40     • **public static final String[] MONTH\_NAMES:** menyimpan nama-nama bulan dalam  
41        bahasa Indonesia.

42     *Method* yang dimiliki kelas ini adalah sebagai berikut:

- 1     • **public Scrapper()**  
2         Berfungsi sebagai *constructor* kelas **Scrapper**.
- 3     • **public void init()**  
4         Berfungsi untuk melakukan koneksi ke halaman utama Portal Akademik Mahasiswa.
- 5     • **public String login(String npm, String password)**  
6         Berfungsi untuk melakukan *login* ke Portal Akademik Mahasiswa.

7     **Parameter:**

- 8         – **npm:** *npm* mahasiswa yang dipakai *login*.
- 9         – **password:** *password* mahasiswa yang dipakai *login*.

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

- 12   • **public TahunSemester requestNamePhotoTahunSemester(String phpsessid,**  
13         **Mahasiswa mahasiswa)**  
14         Berfungsi untuk melakukan pengambilan nama, foto, serta semester yang sedang dijalani  
15         mahasiswa dari Portal Akademik Mahasiswa.

16     **Parameter:**

- 17         – **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Maha-  
18         siswa.
- 19         – **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan  
20         data di Portal Akademik Mahasiswa.

21   **Kembalian:** semester yang sedang dijalani mahasiswa.

- 22   • **public void requestNilai(String phpsessid, Mahasiswa mahasiswa)**  
23         Berfungsi untuk melakukan pengambilan seluruh nilai mata kuliah mahasiswa dari Portal  
24         Akademik Mahasiswa.

25     **Parameter:**

- 26         – **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Maha-  
27         siswa.
- 28         – **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan  
29         data di Portal Akademik Mahasiswa.

- 30   • **public void requestNilaiTOEFL(String phpsessid, Mahasiswa mahasiswa)**  
31         Berfungsi untuk melakukan pengambilan seluruh riwayat nilai TOEFL mahasiswa dari  
32         Portal Akademik Mahasiswa.

33     **Parameter:**

- 34         – **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Maha-  
35         siswa.
- 36         – **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan  
37         data di Portal Akademik Mahasiswa.

- 38   • **public void requestTanggalLahir(String phpsessid, Mahasiswa mahasiswa)**  
39         Berfungsi untuk melakukan pengambilan data tanggal lahir mahasiswa dari Portal  
40         Akademik Mahasiswa.

41     **Parameter:**

- 42         – **phpsessid:** *session* yang didapatkan dari proses *login* ke Portal Akademik Maha-

1 siswa.

2 – **mahasiswa**: objek mahasiswa yang akan ditambahkan datanya dari pengambilan  
3 data di Portal Akademik Mahasiswa.

4 • **public void logout()**

5 Berfungsi untuk melakukan *logout* dari Portal Akademik Mahasiswa.

6 • **public String[] parseSemester(String sem\_raw)**

7 Berfungsi untuk melakukan *parsing* data semester mahasiswa dari Portal Akademik  
8 Mahasiswa agar dapat diolah lebih lanjut.

9 **Parameter:**

10 – **sem\_raw**: data semester yang didapat dari Portal Akademik Mahasiswa.

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

12 • **public Mahasiswa[] requestMahasiswaList(String phpsessid, Mahasiswa  
13 mahasiswa)**

14 Berfungsi untuk mengambil seluruh mahasiswa dengan data yang diambil berupa nama,  
15 foto, semester yang sedang dijalani, dimana seluruh mahasiswa tersebut memiliki dosen  
16 wali yang sama dengan yang melakukan *login* (dalam implementasi menggunakan Portal  
17 Akademik Mahasiswa, mahasiswa yang diambil adalah hanya mahasiswa yang melakukan  
18 *login*).

19 **Parameter:**

20 – **phpsessid**: *session* yang didapatkan dari proses *login* ke Portal Akademik Maha-  
21 siswa.

22 – **mahasiswa**: objek mahasiswa yang akan ditambahkan datanya dari pengambilan  
23 data di Portal Akademik Mahasiswa.

24 **Kembalian:** *array* yang berisi seluruh mahasiswa yang memiliki dosen wali yang sama  
25 dengan yang melakukan *login* (dalam implementasi menggunakan Portal Akademik  
26 Mahasiswa, mahasiswa yang diambil adalah hanya mahasiswa yang melakukan *login*).

27 • **public Mahasiswa requestMahasiswaDetail(String phpsessid, Mahasiswa  
28 mahasiswa)**

29 Berfungsi untuk mengambil detail lebih lanjut mengenai data mahasiswa berupa nilai  
30 TOEFL, seluruh nilai mata kuliah, tanggal lahir mahasiswa.

31 **Parameter:**

32 – **phpsessid**: *session* yang didapatkan dari proses *login* ke Portal Akademik Maha-  
33 siswa.

34 – **mahasiswa**: objek mahasiswa yang akan ditambahkan datanya dari pengambilan  
35 data di Portal Akademik Mahasiswa.

36 **Kembalian:** objek mahasiswa yang telah ditambahkan datanya dari pengambilan data  
37 di Portal Akademik Mahasiswa.

38 3. MultipleRequest

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

- 1     • **private int index:** menyimpan indeks semester yang akan digunakan untuk mengakses **listSemester**.
- 2
- 3     • **private ArrayList<String> listSemester:** menyimpan daftar semester yang telah ditempuh mahasiswa.
- 4
- 5     • **private String NILAI\_URL:** menyimpan *url* halaman nilai Portal Akademik Mahasiswa.
- 6
- 7     • **private String phpsessid:** menyimpan *session* yang dapat digunakan untuk mengakses menu di Portal Akademik Mahasiswa.
- 8
- 9     • **private Mahasiswa mahasiswa:** menyimpan objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.
- 10
- 11    • **private ScriptEngineManager factory:** untuk menjalankan *javascript*.
- 12    • **private ScriptEngine engine:** untuk menjalankan *javascript*.

Method yang dimiliki kelas ini adalah sebagai berikut:

- 14    • **public MultipleRequest(int index, ArrayList<String> listSemester,**
- 15      **String NILAI\_URL, String phpsessid, Mahasiswa mahasiswa)**
- 16      Berfungsi sebagai *constructor* kelas **MultipleRequest**.

#### Parameter:

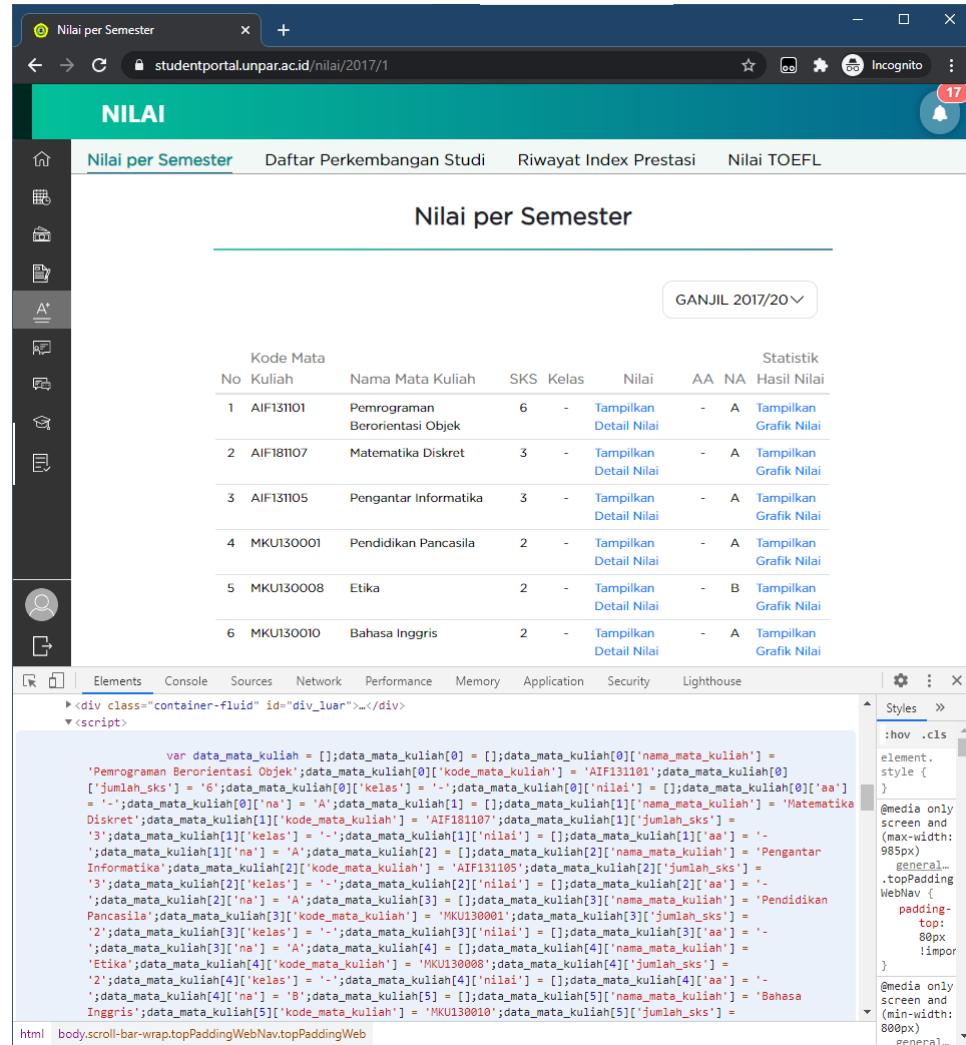
- 18     – **index:** indeks semester.
- 19     – **listSemester:** *list* semester yang ditempuh mahasiswa.
- 20     – **NILAI\_URL:** *url* halaman nilai pada Portal Akademik Mahasiswa.
- 21     – **phpsessid:** *session* yang didapatkan dari proses login ke Portal Akademik Mahasiswa.
- 22
- 23     – **mahasiswa:** objek mahasiswa yang akan ditambahkan datanya dari pengambilan data di Portal Akademik Mahasiswa.

- 25    • **public void run()**

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

- 28    (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.
- 30
- 31    (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>).
- 33
- 34    (c) Setelah berhasil, kemudian melakukan kueri css berdasarkan script yang mengandung nilai mahasiswa (Gambar 4.4).
- 35
- 36    (d) Selanjutnya adalah mendapatkan script yang mengandung script “var data\_mata\_kuliah = [];;” sampai indeks dari “var data\_angket = [];;”.
- 37
- 38    (e) Setelah mendapatkan script yang dibutuhkan, selanjutnya menjalankan script menggunakan *method* milik kelas **ScriptEngine** yaitu **Object eval(String script)**.
- 39
- 40    (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 sebanyak jumlah mata kuliah per semesternya.



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 STAkad siakad;`: menyimpan objek `STAkad`.

*Method* yang dimiliki kelas ini adalah sebagai berikut:

- public SIAkadDataPuller()

Berfungsi sebagai *constructor* kelas SIAkadDataPuller.

- public Mahasiswa[] pullMahasiswa()

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

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

- 1     • `public Mahasiswa pullMahasiswaDetail()`

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

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

6     2. SIAkad

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

- 9     • `protected static final String SIAKAD_BASE_URL`: menyimpan *url* utama SIAKAD.  
10    • `protected static final String SSO_BASE_URL`: menyimpan *url* halaman *Single Sign On*(SSO) UNPAR.  
11    • `protected static final String MU_BASE_URL`: menyimpan *url* MU UNPAR yang  
12      digunakan untuk memeriksa apakah ada pengguna dengan id tertentu.  
13    • `protected static final int DEFAULT_TIMEOUT`: waktu *timeout*.  
14    • `protected final Logger logger`: untuk menulis log.  
15    • `public static final String[] MONTH_NAMES`: menyimpan nama-nama bulan dalam  
16      bahasa Indonesia.  
17    • `Token token`: menyimpan *cookies* yang dapat digunakan untuk mengakses menu di  
18      SIAKAD.  
19

20         *Method* yang dimiliki kelas ini adalah sebagai berikut:

- 21     • `public SIAkad()`

22         Berfungsi sebagai *constructor* kelas SIAkad.

- 23     • `public SIAkad(boolean verbose)`

24         Berfungsi sebagai *constructor* kelas SIAkad.

25         **Parameter:**

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

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

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

31         **Parameter:**

32             – `username`: username dosen.

33             – `password`: password dosen.

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

- 35     • `public void logout()`

36         Berfungsi untuk melakukan *logout* dari SIAKAD.

- 37     • `public List<Mahasiswa> requestMahasiswaList()`

38         Berfungsi untuk mendapatkan daftar mahasiswa yang diwalikan oleh dosen terlogin.

39         **Kembalian:** daftar mahasiswa yang diwalikan oleh dosen terlogin.

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

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

42         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

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

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

4 *Method* yang dimiliki kelas ini adalah sebagai berikut:

- 5 • `public Token(Map<String, String> cookies)`

6 Berfungsi sebagai *constructor* kelas *Token*.

- 7 • `public Map<String, String> getCookies()`

8 Berfungsi untuk mendapatkan *cookies*.

9 **Kembalian:** *cookies*.

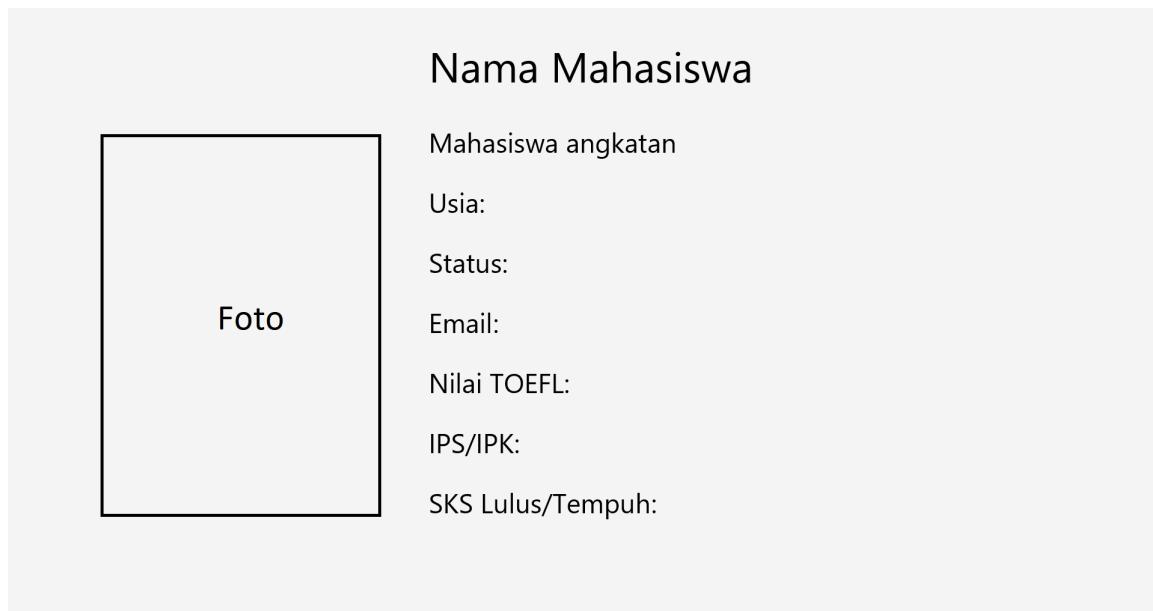
- 10 • `public void injectToConnection(Connection connection)` Berfungsi untuk mema-  
11 sukkan *session* ke dalam *connection*.

12 **Parameter:**

- 13 – `connection: koneksi yang ingin dimasukkan session.`

## 14 4.2 Perancangan Antarmuka

15 Gambar 4.5 menunjukkan rancangan antarmuka dari aplikasi *screensaver*. Antarmuka aplikasi  
16 akan menampilkan foto dari mahasiswa apabila foto tersebut tersedia. Baris pertama merupakan  
17 nama dari mahasiswa, kemudian baris kedua akan diisi oleh angkatan dari mahasiswa tersebut.  
18 Baris ketiga menampilkan usia dan tanggal lahir mahasiswa. Baris keempat merupakan status  
19 dari mahasiswa tersebut apabila tersedia. Baris kelima merupakan *email* mahasiswa dimana  
20 format *email* mahasiswa UNPAR adalah **NomorPokokMahasiswa@student.unpar.ac.id**. Baris  
21 keenam berisi nilai TOEFL terakhir mahasiswa apabila mahasiswa tersebut sudah mengambil ujian  
22 TOEFL, atau diisi "Tidak Tersedia" apabila mahasiswa tersebut belum mengambil ujian TOEFL.  
23 Baris ketujuh menampilkan IPS dan IPK dari mahasiswa tersebut. Baris terakhir diisi oleh jumlah  
24 sks yang telah lulus dan jumlah sks yang telah ditempuh oleh mahasiswa tersebut. Setiap 5 detik  
25 tampilan aplikasi akan berubah ke mahasiswa berikutnya. Namun pada aplikasi yang dibuat oleh  
26 terbimbing, data yang ditampilkan hanya mahasiswa itu sendiri, sedangkan pada aplikasi yang  
27 dibuat oleh pembimbing, data yang ditampilkan akan berubah.



Gambar 4.5: Rancangan Antarmuka *Screensaver*



1

## BAB 5

2

### IMPLEMENTASI DAN PENGUJIAN

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

7 **5.1 Implementasi**

8 **5.1.1 Lingkungan Implementasi**

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

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

17 **5.1.2 Hasil Implementasi**

18 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**,  
19 perlu dilakukan perubahan menjadi ekstensi **exe** terlebih dahulu. Perubahan ekstensi menjadi **exe**  
20 tersebut dilakukan dengan bantuan aplikasi bernama Launch4j<sup>1</sup>. Setelah didapatkan *file* berekstensi  
21 **exe**, maka untuk mengubahnya menjadi berekstensi **scr** cukup dengan melakukan rename pada  
22 *file* tersebut menjadi berekstensi **scr**. File berekstensi **scr** tersebut dapat disimpan pada direktori  
23 “System32” didalam folder “Windows” agar dapat terdeteksi sebagai *screensaver* oleh Windows.  
24 Setelah mengatur aplikasi menjadi *screensaver*, aplikasi tersebut akan dijalankan secara otomatis  
25 setelah komputer tidak digunakan selama beberapa saat. Gambar 5.1 dan 5.2 merupakan tampilan  
26 dari aplikasi *screensaver*. Dikarenakan pada Portal Akademik Mahasiswa tidak terdapat foto  
27 mahasiswa, maka foto digantikan dengan *base64 image* yang dimasukkan secara *hardcode*. *Layout*  
28 dari aplikasi disimpan dalam *file* bertipe FXML (lampiran E). *File* FXML tersebut tidak dibuat  
29

---

<sup>1</sup><http://launch4j.sourceforge.net>

- 1 secara manual, melainkan dengan menggunakan aplikasi Scene Builder<sup>2</sup>.

## 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

---

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

## <sup>1</sup> 5.2 Pengujian

### <sup>2</sup> 5.2.1 Pengujian Fungsional

<sup>3</sup> Pengujian fungsional dilakukan untuk mengetahui kesesuaian reaksi perangkat lunak dengan reaksi  
<sup>4</sup> yang diharapkan berdasarkan aksi pengguna terhadap perangkat lunak. Tabel 5.1 merupakan hasil  
<sup>5</sup> pengujian perangkat lunak yang dilakukan di komputer penulis dengan spesifikasi berikut:

- <sup>6</sup> 1. *Processor*: Intel Core i5-10300H
- <sup>7</sup> 2. *Random Access Memory(RAM)*: 16 GB DDR4
- <sup>8</sup> 3. Sistem Operasi: Windows 10
- <sup>9</sup> 4. Resolusi Layar: 1920 x 1080
- <sup>10</sup> 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                 |

### <sup>11</sup> 5.2.2 Pengujian Eksperimental

<sup>12</sup> *Subbab ini ditulis oleh dosen pembimbing.*

<sup>13</sup> Pengujian eksperimental dilakukan oleh dosen pembimbing, karena mahasiswa tidak memiliki  
<sup>14</sup> akses ke SIAKAD apalagi ke data mahasiswa di luar penulis sendiri. Pengujian eksperimental  
<sup>15</sup> dilakukan dengan melakukan *git branching* dari hasil (hampir) final dari penulis di commit d60be65.  
<sup>16</sup> Pengujian eksperimental dilakukan pada komputer dosen dengan spesifikasi seperti berikut<sup>3</sup>:

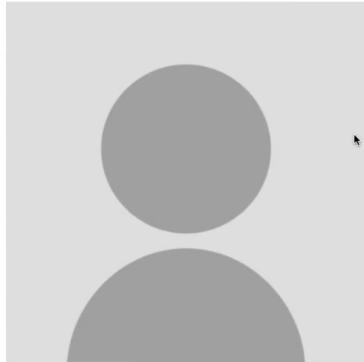
- <sup>17</sup> 1. Komputer: iMac (21.5-inch, Late 2013)
- <sup>18</sup> 2. Sistem Operasi: macOS Catalina 10.15.7
- <sup>19</sup> 3. *Processor*: 2,7 GHz Quad-Core Intel Core i5
- <sup>20</sup> 4. *Memory*: 8 GB 1600 MHz DDR3
- <sup>21</sup> 5. *Graphics*: Intel Iris Pro 1536 MB
- <sup>22</sup> 6. Resolusi Layar: 1920 x 1080
- <sup>23</sup> 7. Versi Java: 1.8.0\_282

<sup>24</sup> Ada beberapa penyesuaian yang dilakukan oleh dosen pembimbing supaya *screensaver* bisa  
<sup>25</sup> dijalankan dengan cukup baik pada lingkungan dosen. Karena keterbatasan waktu, penyesuaian ini  
<sup>26</sup> dilakukan tanpa berkoordinasi secara intensif dengan mahasiswa. Penyesuaian-penyesuaian tersebut  
<sup>27</sup> antara lain:

<sup>3</sup>Idealnya 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

1. **Data diambil dari SIAKAD bukan dari Portal Akademik Mahasiswa.** Daftar ma-  
2. hasiswa yang diambil diacak urutannya sehingga saat ditampilkan tidak selalu dimulai dari  
3. angkatan tertua.  
4. **Upgrade versi SIAModels dari 4.0.0 menjadi 5.0.1.** Pada versi terbaru SIAModels, ada  
5. beberapa perbaikan, di mana salah satunya adalah *bug* saat menentukan angkatan mahasiswa  
6. berdasarkan NPM nya. Perbaikan lainnya adalah pada perhitungan IPK dan IPS. SIAModels  
7. merupakan pustaka eksternal dan bukan merupakan bagian inti dari *screensaver* yang dibuat.  
8. **Load data di *thread* terpisah.** Pada program yang dibuat oleh mahasiswa, program hanya  
9. mengunduh satu data dari Portal Akademik Mahasiswa, yaitu data yang bersangkutan sendiri.  
10. Pada SIAKAD, diunduh banyak data dan ternyata menimbulkan masalah saat mengunduh  
11. data mahasiswa untuk ditampilkan pada slide berikutnya. Masalah muncul karena saat  
12. load data mahasiswa berikutnya, program menjadi tidak responsif. Oleh karena itu, saat  
13. menampilkan sebuah slide, program diubah supaya mengunduh data mahasiswa untuk slide  
14. berikutnya di latar belakang. Saat slide berikutnya ditampilkan, sudah tidak perlu melakukan  
15. pengunduhan kembali karena datanya sudah siap.  
16. **Saat mengulang ke mahasiswa pertama, tidak dilakukan load data kembali.**  
17. **Jika data mahasiswa belum siap ditampilkan, tidak maju ke slide berikutnya.**  
18. **Bugfix untuk pembacaan bulan Februari.** Pada SIAKAD, ternyata bulan kedua ditu-  
19. liskan sebagai “Pebruari” dengan huruf “P”, karena itu perlu penyesuaian sehingga mahasiswa  
20. dengan bulan lahir Februari dapat dibaca dengan baik.  
21. Perubahan lengkap kode dibandingkan dengan kode milik mahasiswa dapat dilihat pada lampiran  
22. D.1. Setelah perbaikan-perbaikan tersebut dilakukan, program dapat dijalankan dengan baik, dan  
23. untuk menampilkan 30 mahasiswa “wali” dari dosen pembimbing, dibutuhkan waktu sekitar 8  
24. menit, di mana setelahnya mengulang kembali dari mahasiswa pertama. Beberapa hasil tangkapan  
25. layar dapat dilihat pada gambar 5.3, 5.4, 5.5, 5.6, dan 5.7. Dosen sudah meminta izin kepada kelima  
26. 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

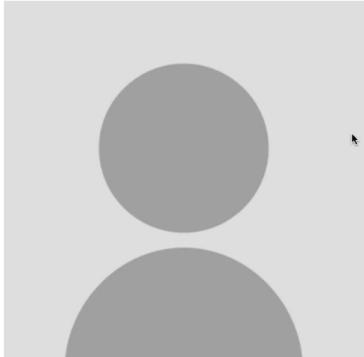
Nilai TOEFL: Tidak Tersedia

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

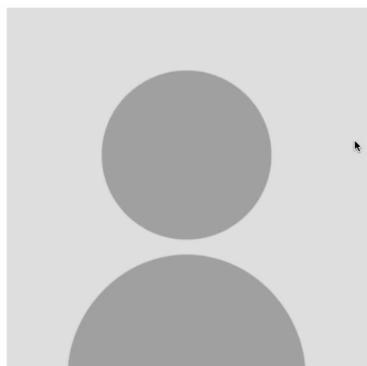
Nilai TOEFL: Tidak Tersedia

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



<sup>1</sup>

## BAB 6

<sup>2</sup>

### KESIMPULAN DAN SARAN

#### <sup>3</sup> 6.1 Kesimpulan

- <sup>4</sup> Dari hasil pembangunan aplikasi *screensaver*, didapatkan kesimpulan-kesimpulan sebagai berikut:
- <sup>5</sup> 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.
- <sup>6</sup> 2. Aplikasi *screensaver* telah dapat dijalankan pada seluruh perangkat dengan sistem operasi windows.
- <sup>7</sup>
- <sup>8</sup>
- <sup>9</sup>
- <sup>10</sup>

#### <sup>11</sup> 6.2 Saran

- <sup>12</sup> Dari hasil penelitian termasuk kesimpulan yang didapat, berikut adalah beberapa saran untuk pengembangan lebih lanjut:
- <sup>13</sup> 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.
- <sup>14</sup> 2. Tampilan aplikasi *screensaver* sebaiknya dibuat secara *responsive*, sehingga aplikasi *screensaver* dapat dijalankan dengan baik di berbagai resolusi layar.
- <sup>15</sup> 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.
- <sup>16</sup>
- <sup>17</sup>
- <sup>18</sup>
- <sup>19</sup>
- <sup>20</sup>
- <sup>21</sup>
- <sup>22</sup>



## 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](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 8 (2008-2015) *JavaFX 8*. Oracle. 2300 Oracle Way Austin, TX 78741.
- [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
209
210

```

```

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 (rekomendasi: false) 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.siaakad.SIAkadDataPuller;
9 -import id.ac.unpar.screensaver.studentportal.StudentPortalDataPuller;
10 +import id.ac.unpar.screensaver.siaakad.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
 .getIndexOfMahasiswa()]);
86- }
87- }
88- if (listMahasiswa[this.getIndexOfMahasiswa()].getTanggalLahir() == null) {
89- updateView();
90- } else {
91- try {
92+ if (mahasiswaLoaded[this.getIndexOfMahasiswa()]) {
93- // Update view only if mahasiswa is loaded. Otherwise, wait for next turn
94- this.updateView(listMahasiswa[this.getIndexOfMahasiswa()]);
95- this.setIndexOfMahasiswa(this.getIndexOfMahasiswa() + 1);
96- } catch (IOException ex) {
97- Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
98+ this.setIndexOfMahasiswaAndPreload((this.getIndexOfMahasiswa() + 1) % listMahasiswa.
 length);
99+ } else {
100+ System.out.println("Mahasiswa " + listMahasiswa[this.getIndexOfMahasiswa()].getNama() + "
 is not ready. Waiting for next turn...");
101+ }
102+ } catch (IOException ex) {
103+ Logger.getLogger(PrimaryController.class.getName()).log(Level.SEVERE, null, ex);
104+ }
105- }
106- }
107- }
108-);
109- timeline.setCycleCount(Animation.INDEFINITE);
110- timeline.play();
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
133 public void updateView(Mahasiswa mahasiswa) throws IOException {
134- @@ -129,13 +99,12 @@ public class PrimaryController implements Initializable {
135- }
136- this.nama.setText(mahasiswa.getNama());
137- this.angkatan.setText(mahasiswa.getTahunAngkatan() + "");
138- this.usia.setText(Period.between(mahasiswa.getTanggalLahir(), LocalDate.now()).getYears() + " tahun " + Period.between(
 mahasiswa.getTanggalLahir(), LocalDate.now()).getMonths() + " bulan " + "(lahir " + mahasiswa.getTanggalLahir().toString()
 + ")");
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.
 getTanggalLahir(), LocalDate.now()).getMonths() + " bulan (lahir " + mahasiswa.getTanggalLahir().toString() + ")") :
147+ "Tidak tersedia"
148+);
149+ this.status.setText(mahasiswa.getStatus() != null ? mahasiswa.getStatus().toString() : "Tidak Tersedia");
150+ this.email.setText(mahasiswa.getEmailAddress());
151- if (mahasiswa.getNilaiTOEFL() != null && mahasiswa.getNilaiTOEFL().size() > 0) {
152- this.toefl.setText(mahasiswa.getNilaiTOEFL().get(mahasiswa.getNilaiTOEFL().firstKey()).toString());
153- }
154- @@ -143,12 +112,34 @@ public class PrimaryController implements Initializable {
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.calculateIPSK(tahunSemesterTerakhir) * 100.0) / 100.0 + "/" + Math.round(mahasiswa.
168+ calculateIPK() * 100.0) / 100.0
169+);
170+ this.sks.setText(+mahasiswa.calculateSKSLulus() + "/" + mahasiswa.calculateSKSTempuh(false));
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"))
274 + // Integer.parseInt((String) mataKuliah.get("jumlah_sks")), (String) mataKuliah.get("nama_mata_kuliah"));
275 + // mahasiswa.getRiwayatNilai()
276 + // .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
22
23
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
32
33
34 </Text>
35 <Text fx:id="angkatan" strokeType="OUTSIDE" strokeWidth="0.0">
36
37
38
39 </Text>
40 </children>
41 <VBox.margin>
42 <Insets bottom="40.0" />
43 </VBox.margin>
44 </HBox>
45 <HBox>
46 <children>
47 <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Usia:<u>">
48
49
50
51 </Text>
52 <Text fx:id="usia" strokeType="OUTSIDE" strokeWidth="0.0">
53
54
55
56 <HBox.margin>
57 <Insets />
58 </HBox.margin>
59 </Text>
60 </children>
61 <VBox.margin>
62 <Insets bottom="40.0" />
63 </VBox.margin>
64 </HBox>
65 <HBox>
66 <children>
67 <Text strokeType="OUTSIDE" strokeWidth="0.0" text="Status:<u>">
68
69
70
71 </Text>
72 <Text fx:id="status" strokeType="OUTSIDE" strokeWidth="0.0">
73
74
```

```
75
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
86
87
88 </Text>
89 <Text fx:id="email" strokeType="OUTSIDE" strokeWidth="0.0">
90
91
92
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
103
104
105 </Text>
106 <Text fx:id="toefl" strokeType="OUTSIDE" strokeWidth="0.0">
107
108
109
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
120
121
122 </Text>
123 <Text fx:id="ipk" strokeType="OUTSIDE" strokeWidth="0.0">
124
125
126
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
137
138
139 </Text>
140 <Text fx:id="sks" strokeType="OUTSIDE" strokeWidth="0.0">
141
142
143
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>
```