SKRIPSI

SCREENSAVER INFORMASI MAHASISWA WALI

Harry Senjaya Darmawan

NPM: 2017730067

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

«tahun»

UNDERGRADUATE THESIS

STUDENTS' INFORMATION SCREENSAVER

Harry Senjaya Darmawan

NPM: 2017730067

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

«tahun»

LEMBAR PENGESAHAN

SCREENSAVER INFORMASI MAHASISWA WALI

Harry Senjaya Darmawan

NPM: 2017730067

Bandung, «tanggal» «bulan» «tahun»

Menyetujui,

Pembimbing

Pascal Alfadian, Nugroho, M.Comp.

Ketua Tim Penguji	Anggota Tim Penguji
«penguji 1»	«penguji 2»

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

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

SCREENSAVER INFORMASI MAHASISWA WALI

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

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

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

Harry Senjaya Darmawan
NPM: 2017730067

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»

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

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»

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

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

KATA PENGANTAR

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

Bandung, «bulan» «tahun»

Penulis

DAFTAR ISI

Kata Pengantar **xv**

Daftar Isi **xvii**

Daftar Gambar **xix**

1 Pendahuluan **1**
- 1.1 Latar Belakang . . . 1
- 1.2 Rumusan Masalah . . . 1
- 1.3 Tujuan . . . 1
- 1.4 Batasan Masalah . . . 2
- 1.5 Metodologi . . . 2
- 1.6 Sistematika Pembahasan . . . 2

2 Landasan Teori **3**
- 2.1 *Web Scraping* . . . 3
- 2.2 Jsoup . . . 3
  - 2.2.1 Jsoup . . . 4
  - 2.2.2 Connection . . . 4
  - 2.2.3 Response . . . 4
  - 2.2.4 Elements . . . 5
  - 2.2.5 Element . . . 5
- 2.3 JavaFX dan FXML . . . 5
  - 2.3.1 JavaFX . . . 5
  - 2.3.2 FXML . . . 7
- 2.4 SIAModels . . . 7
  - 2.4.1 Mahasiswa . . . 7
  - 2.4.2 Nilai . . . 10
  - 2.4.3 ChronologicalComparator . . . 10
  - 2.4.4 MataKuliah . . . 11
  - 2.4.5 JenisKelamin . . . 11
  - 2.4.6 Status . . . 11

3 Analisis **13**
- 3.1 Analisis Pemanfaatan Jsoup . . . 13
  - 3.1.1 *Login* . . . 13
  - 3.1.2 Halaman Utama . . . 15
  - 3.1.3 Profil . . . 17
  - 3.1.4 Nilai . . . 17

Daftar Referensi **25**

A Kode Program **27**

DAFTAR GAMBAR

3.1 Halaman *Login* 1 14
3.2 Halaman *Login* 2 14
3.3 Halaman Utama Portal Akademik Mahasiswa 16
3.4 Halaman Profil 17
3.5 Halaman Nilai Per Semester 18
3.6 Halaman Daftar Perkembangan Studi (1) 18
3.7 Halaman Daftar Perkembangan Studi (2) 19
3.8 Halaman Riwayat Indeks Prestasi 21
3.9 Halaman Nilai TOEFL 22

B.1 Hasil 1 29
B.2 Hasil 2 29
B.3 Hasil 3 29
B.4 Hasil 4 29

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Setiap Penasehat Akademik memiliki data mengenai mahasiswa walinya. Namun, walaupun Penasehat Akademik memiliki data mengenai mahasiswa walinya, Penasehat Akademik juga perlu melakukan pemeriksaan data mahasiswa walinya, terutama data akademiknya secara berkala. Dengan berbagai kesibukan yang dialami oleh para Penasehat Akademik dan mahasiswa, ditambah dengan situasi Indonesia saat ini yang menyebabkan perkuliahan dilakukan secara daring, akan sangat sulit bagi Penasehat Akademik untuk menemui mahasiswa walinya. Hal ini menyebabkan Penasehat Akademik kesulitan mengamati perkembangan mahasiswa walinya.

Maka dari itu, pada skripsi ini akan dibuat sebuah perangkat lunak yang berupa *screensaver* yang dapat menampilkan data akademik mahasiswa wali secara acak. *A screensaver (or screen saver) is a computer program that blanks the screen or fills it with moving images or patterns when the computer has been idle for a long time [1].* Dengan menggunakan perangkat lunak tersebut, Penasehat Akademik dapat tetap mengamati perkembangan mahasiswa walinya, paling tidak secara akademik.

Dikarenakan terbimbing tidak memiliki akses ke SIAKAD [2] untuk mengakses data mahasiswa wali, namun terbimbing memiliki akses ke Portal Akademik Mahasiswa [3] maka, terbimbing mensimulasikan dengan Portal Akademik Mahasiswa, dan kemudian Pembimbing mengubah aksesnya ke SIAKAD. Struktur kelas yang akan digunakan dalam pembuatan perangkat lunak ini yaitu, struktur kelas SIAModels yang tersedia pada Github dan Maven Public Repository [4].

Perangkat lunak akan dibangun menggunakan bahasa pemrograman Java. Terdapat beberapa teknologi yang dapat dimanfaatkan dalam bahasa pemrograman Java. Teknologi yang pertama yaitu *library* jsoup. Jsoup dapat digunakan untuk melakukan *web scraping*, sehingga pengambilan data mahasiswa tidak memerlukan API *(Application Programming Interface)* [5]. Teknologi lainnya yang dapat dimanfaatkan yaitu JavaFX. JavaFX dapat digunakan untuk mengonversi aplikasi tersebut menjadi *screensaver*.

1.2 Rumusan Masalah

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

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

1.3 Tujuan

Tujuan yang ingin dicapai dari penulisan skripsi ini sebagai berikut:

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

1.4 Batasan Masalah

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

1.5 Metodologi

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

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

1.6 Sistematika Pembahasan

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

- Bab 1. Pendahuluan, membahas tentang latar belakang, rumusan masalah, tujuan penelitian, batasan masalah, metode penelitian dan sistematika pembahasan mengenai skripsi.
- Bab 2. Landasan Teori, membahas landasan dari teori-teori yang berhubungan serta mendukung penelitian, meliputi *web scraping*, jsoup, JavaFX, dan SIAModels.
- Bab 3. Analisis, menjelaskan tentang kebutuhan data, analisis Portal Akademik Mahasiswa versi 2020, dan analisis cara memanfaatkan jsoup.
- Bab 4. Perancangan, membahas perancangan antarmuka, diagram kelas beserta deskripsi kelas dan fungsinya.
- Bab 5. Implementasi dan pengujian, membahas hasil-hasil implementasi dan pengujian secara fungsional dan eksperimental.
- Bab 6. Kesimpulan dan saran, membahas kesimpulan yang diperoleh dari penelitian ini dan saran untuk pengembangan berikutnya.

BAB 2

LANDASAN TEORI

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

2.1 *Web Scraping*

Menurut Bo Zhao [6], *web scraping* juga dikenal sebagai ekstraksi atau pemanenan web, adalah teknik untuk mengekstrak data dari *World Wide Web* (WWW) dan menyimpannya ke sistem file atau basis data untuk diambil atau dianalisis. Secara khusus, proses *web scraping* yaitu:

1. Membuat HTTP *request* untuk memperoleh sumber daya dari situs web yang ditargetkan. Permintaan ini dapat diformat baik dalam URL yang berisi kueri GET atau bagian dari pesan HTTP yang berisi kueri POST.
2. Setelah *request* berhasil diterima dan diproses oleh situs web yang ditargetkan, sumber daya yang diminta akan diambil dari situs web dan kemudian dikirim kembali ke program *web scraping*. Sumber daya bisa dalam berbagai format, seperti halaman web yang dibuat dari HTML, *data feed* dalam format XML atau JSON, atau data multimedia seperti gambar, audio, atau video.
3. Setelah data web diunduh, proses ekstraksi dilanjutkan dengan mengurai, memformat ulang, dan mengatur data secara terstruktur.

2.2 Jsoup

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

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

Subbab-subbab berikut menjelaskan beberapa kelas dari jsoup [5].

2.2.1 Jsoup

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

- `public static Connection connect(String url)`
  Berfungsi untuk membuat koneksi baru ke URL. Digunakan untuk mengambil dan mengurai halaman HTML.
  Parameter: URL situs web dengan protokol HTTP atau HTTPS.
  Kembalian: koneksi dengan situs web.

2.2.2 Connection

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

- `Connection cookies(Map<String,String> cookies)`
  Berfungsi untuk menambahkan *cookies* ke *request*.
  Parameter:
  - `cookies` `Map` dari *cookie*.

  Kembalian: koneksi yang sama tetapi sudah diubah.
- `Connection data(String key, String value)`
  Berfungsi untuk menambahkan parameter data *request* yang bisa dikirim melalui *method* HTTP GET atau POST.
  Parameter:
  - `key` kunci data.
  - `value` nilai data.

  Kembalian: koneksi yang sama tetapi sudah diubah.
- `Connection method(Connection.Method method)`
  Berfungsi untuk mengatur *method request* yang akan digunakan, HTTP GET atau POST. *Default*-nya adalah GET.
  Parameter:
  - `method` *method request* HTTP.

  Kembalian: koneksi yang sama tetapi sudah diubah.
- `Connection timeout(int millis)`
  Berfungsi untuk mengatur batas waktu *request*. Batas waktu *default* adalah 30 detik. Batas waktu nol akan dianggap sebagai batas waktu yang tak terhingga.
  Parameter:
  - `millis` batas waktu dalam milidetik.

  Kembalian: koneksi yang sama tetapi sudah diubah.
- `Connection.Response execute()`
  Berfungsi untuk mengirim *request* HTTP.
  Kembalian: objek `Response`.

2.2.3 Response

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

- `Document parse()`
  Berfungsi untuk mengurai *body* jawaban menjadi dokumen.
  Kembalian: dokumen yang diurai.
- `String body()`
  Berfungsi untuk mendapatkan *body* jawaban dalam bentuk *string*.
  Kembalian: *body* jawaban dalam bentuk *string*.

2.2.4 Elements

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

- `public Elements select(String query)`
  Berfungsi untuk menemukan elemen-elemen yang sesuai dalam *list* elemen.
  Parameter:
  - `query` kueri CSS berupa CSS Selector.

  Kembalian: elemen-elemen yang sudah diseleksi sesuai kueri.
- `public String val()`
  Berfungsi untuk mendapatkan nilai dari elemen pertama.
  Kembalian: nilai elemen.
- `public String text()`
  Berfungsi untuk mendapatkan kombinasi teks dari seluruh elemen yang sesuai.
  Kembalian: seluruh teks dalam *string*.

2.2.5 Element

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

- `public Elements select(String cssQuery)`
  Berfungsi untuk menemukan elemen yang cocok dengan kueri CSS Selector, dengan elemen ini sebagai konteks awal.
  Parameter:
  - `cssQuery` kueri CSS berupa CSS Selector.

  Kembalian: elemen-elemen HTML yang sesuai dengan kueri CSS.
- `public Element child(int index)`
  Berfungsi untuk mendapatkan anak elemen berdasarkan nomor indeks.
  Parameter:
  - `index` nomor index.

  Kembalian: anak elemen.
- `public Element children()`
  Berfungsi untuk mendapatkan seluruh anak elemen.
  Kembalian: seluruh anak elemen.
- `public String className()`
  Berfungsi untuk mendapatkan nama kelas elemen.
  Kembalian: nama kelas elemen.
- `public String text()`
  Berfungsi untuk mendapatkan teks gabungan dari elemen.
  Kembalian: teks dalam *string*.

2.3 JavaFX dan FXML

2.3.1 JavaFX

JavaFX adalah platform aplikasi klien *open source* generasi berikutnya untuk *desktop*, *mobile*, dan *embedded systems* yang dibangun dengan Java. JavaFX memungkinkan untuk membuat aplikasi Java dengan antarmuka pengguna (UI) modern dengan akselerasi perangkat keras yang sangat portabel. [8]. Subbab-subbab berikut menjelaskan beberapa kelas dari JavaFX. [8]

Application

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

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

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

- `public void init()`
  Method inisialisasi aplikasi. *Method* ini dipanggil segera setelah kelas `Application` dimuat dan dibangun. *Method* ini dapat ditimpa untuk melakukan inisialisasi sebelum aplikasi sebenarnya dimulai.
- `public abstract void start(Stage primaryStage)`
  Titik masuk utama untuk semua aplikasi JavaFX. *Method* `start` dipanggil setelah *method* `init` kembali, dan setelah sistem siap untuk aplikasi mulai berjalan.
  Parameter:
  - `primaryStage` *stage* utama untuk aplikasi ini, tempat *scene* aplikasi dapat diatur.
- `public static void launch()`
  Meluncurkan aplikasi. *Method* ini biasanya dipanggil dari `main()` *method*. Tidak boleh dipanggil lebih dari sekali atau *exception* akan dilemparkan. Harus merupakan *subclass* dari `Application` atau *RuntimeException* akan dilemparkan.

Stage

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

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

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

Scene

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

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

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

FXMLLoader

Memuat hierarki objek dari dokumen XML.

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

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

2.3.2 FXML

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

2.4 SIAModels

SIAModels merupakan kelas-kelas dalam bahasa Java yang merepresentasikan objek-objek yang tersedia di Sistem Informasi Akademik UNPAR [4]. Pada skripsi ini akan digunakan SIAModels versi 4.0.0.

2.4.1 Mahasiswa

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

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

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

- `public Mahasiswa(String npm)`
  Merupakan *constructor* dari kelas `Mahasiswa`.
  Parameter:
  - `npm` nomor pokok mahasiswa.
- `public String getNama()`
  Berfungsi untuk mendapatkan nama mahasiswa.
  Kembalian: nama mahasiswa.
- `public void setNama(String nama)`
  Berfungsi untuk mengubah nama mahasiswa.
  Parameter:
  - `nama` nama mahasiswa.
- `public String getNpm()`
  Berfungsi untuk mendapatkan nomor pokok mahasiswa.
  Kembalian: nomor pokok mahasiswa.
- `public String getPhotoPath()`
  Berfungsi untuk mendapatkan URL dari foto mahasiswa.
  Kembalian: URL dari foto mahasiswa.

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

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

- `public double calculateIPTempuh(boolean lulusSaja)`
  Berfungsi untuk menghitung IP mahasiswa sampai saat ini, dengan aturan perhitungan kuliah yang tidak lulus ditentukan parameter, dan jika pengambilan beberapa kali, diambil nilai terbaik.
  Parameter:
  - `lulusSaja` lulusSaja set true jika ingin membuang mata kuliah tidak lulus, false jika ingin semua (sama dengan "IP N. Terbaik" di DPS). Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.

  Kembalian: IPK lulus.
- `public double calculateIPKumulatif()`
  Berfungsi untuk menghitung IP Kumulatif mahasiswa sampai saat ini, dengan aturan jika pengambilan beberapa kali, diambil semua.Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.
  Kembalian: IPK lulus.
- `public double calculateIPS()`
  Berfungsi untuk menghitung IPS semester terakhir sampai saat ini, dengan aturan kuliah yang tidak lulus dihitung. Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.
  Kembalian: nilai IPS sampai saat ini.
- `public int calculateSKSLulus()`
  Berfungsi untuk menghitung jumlah SKS lulus mahasiswa saat ini. Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.
  Kembalian: SKS lulus.
- `public int calculateSKSTempuh(boolean lulusSaja)`
  Berfungsi untuk menghitung jumlah SKS tempuh mahasiswa saat ini. Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.
  Parameter:
  - `lulusSaja` lulusSaja set true jika ingin membuang SKS tidak lulus

  Kembalian: SKS tempuh
- `public Set<TahunSemester> calculateTahunSemesterAktif()`
  Berfungsi untuk mendapatkan seluruh tahun semester di mana mahasiswa ini tercatat sebagai mahasiswa aktif, dengan strategi memeriksa riwayat nilainya.Jika ada satu nilai saja pada sebuah tahun semester, maka dianggap aktif pada semester tersebut.
  Kembalian: kumpulan tahun semester di mana mahasiswa ini aktif.
- `public boolean hasLulusKuliah(String kodeMataKuliah)`
  Berfungsi untuk memeriksa apakah mahasiswa ini sudah lulus mata kuliah tertentu. Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.
  Parameter:
  - `kodeMataKuliah` kode mata kuliah yang ingin diperiksa kelulusannya.

  Kembalian: `true` jika sudah pernah mengambil dan lulus, `false` jika belum.
- `public boolean hasTempuhKuliah(String kodeMataKuliah)`
  Memeriksa apakah mahasiswa ini sudah pernah menempuh mata kuliah tertentu. Sebelum memanggil *method* ini, `getRiwayatNilai()` harus sudah mengandung nilai per mata kuliah.
  Parameter:
  - `kodeMataKuliah` kode mata kuliah yang ingin diperiksa kelulusannya.

  Kembalian: `true` jika sudah pernah mengambil, `false` jika belum.
- `public int getTahunAngkatan()`

Mendapatkan tahun angkatan mahasiswa ini berdasarkan NPM-nya.
Kembalian: tahun angkatan.

2.4.2 Nilai

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

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

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

- `public Nilai(TahunSemester tahunSemester, MataKuliah mataKuliah, Character kelas, Map<String, Double> nilaiTugas, Double nilaiUTS, Double nilaiUAS, String nilaiAkhir)`
  Merupakan *constructor* dari kelas `Nilai`.
  Parameter:
  - `tahunSemester` tahun dan semester kuliah ini diambil.
  - `mataKuliah` mata kuliah yang diambil.
  - `kelas` kelas kuliah.
  - `nilaiTugas` nilai ART.
  - `nilaiUTS` nilai UTS.
  - `nilaiUAS` nilai UAS.
  - `nilaiAkhir` nilai akhir.
- `public MataKuliah getMataKuliah()`
  Mendapatkan mata kuliah yang diambil.
  Kembalian: mata kuliah.
- `public String getNilaiAkhir()`
  Mengembalikan nilai akhir dalam bentuk huruf (A, B, C, D, ..., atau K).
  Kembalian: nilai akhir dalam huruf, atau `null` jika tidak ada.
- `public Double getAngkaAkhir()`
  Mendapatkan nilai akhir dalam bentuk angka.
  Kembalian: nilai akhir dalam angka, atau `null` jika `getNilaiAkhir()` mengembalikan `null`.
- `public int getTahunAjaran()`
  Mendapatkan tahun ajaran saat pengambilan mata kuliah.
  Kembalian: tahun ajaran saat pengambilan mata kuliah.
- `public TahunSemester getTahunSemester()`
  Mendapatkan tahun dan semester pengambilan mata kuliah.
  Kembalian: tahun dan semester pengambilan mata kuliah.
- `public Semester getSemester()`
  Mendapatkan semester pengambilan mata kuliah.
  Kembalian: semester pengambilan mata kuliah

2.4.3 ChronologicalComparator

Pembanding 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.

2.4.4 MataKuliah

Kelas ini merepresentasikan sebuah mata kuliah. Atribut yang dimiliki kelas ini antara lain:
- `String kode:` kode mata kuliah
- `String nama:` nama mata kuliah
- `Integer sks:` sks mata kuliah.

Beberapa *method* yang dimiliki kelas ini adalah sebagai berikut:
- `public MataKuliah(String kode, String nama, int sks)`
  Merupakan *constructor* dari kelas `MataKuliah`.
  Parameter:
  - `kode` kode mata kuliah.
  - `nama` nama mata kuliah.
  - `sks` sks mata kuliah.
- `public String getKode()`
  Mendapatkan kode mata kuliah.
  Kembalian: kode mata kuliah.
- `public int getSks()`
  Mendapatkan sks mata kuliah.
  Kembalian: sks mata kuliah.
- `public String getNama()`
  Mendapatkan nama mata kuliah.
  Kembalian: nama mata kuliah.

2.4.5 JenisKelamin

Kelas ini berupa enum yang merepresentasikan jenis kelamin mahasiswa. Nilai dari enum ini antara lain:
- `LAKI_LAKI("Laki-laki")`
- `PEREMPUAN("Perempuan")`

2.4.6 Status

Kelas ini berupa enum yang merepresentasikan status mahasiswa. Nilai dari enum ini antara lain:
- `SEMUA("00")`
- `AKTIF("01")`
- `GENCAT("02")`
- `CUTI_SEBELUM_FRS("03")`
- `CUTI_SETELAH_FRS("04")`
- `KELUAR("05")`
- `LULUS("06")`
- `DROP_OUT("07")`
- `SISIPAN("08")`

BAB 3

ANALISIS

Pada bab ini akan dijelaskan mengenai analisis pemanfaatan jsoup untuk mengambil data mahasiswa.

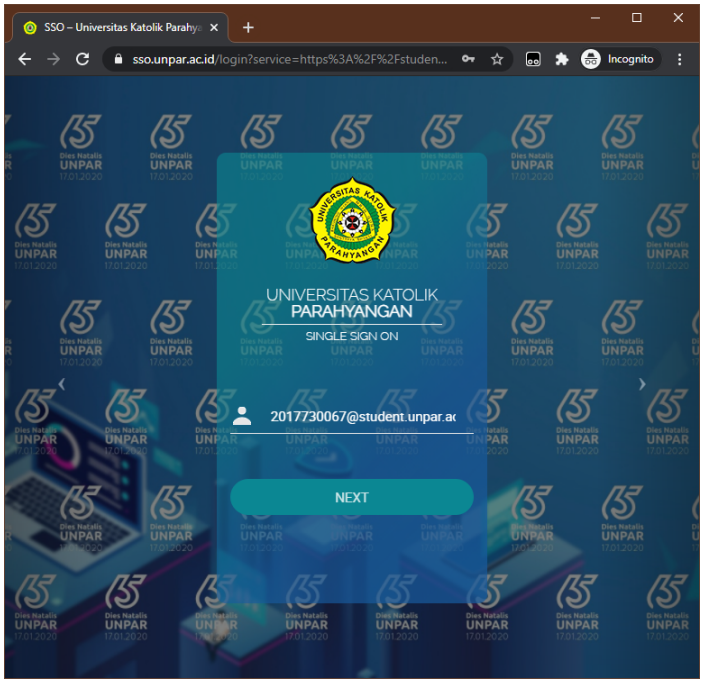
3.1 Analisis Pemanfaatan Jsoup

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

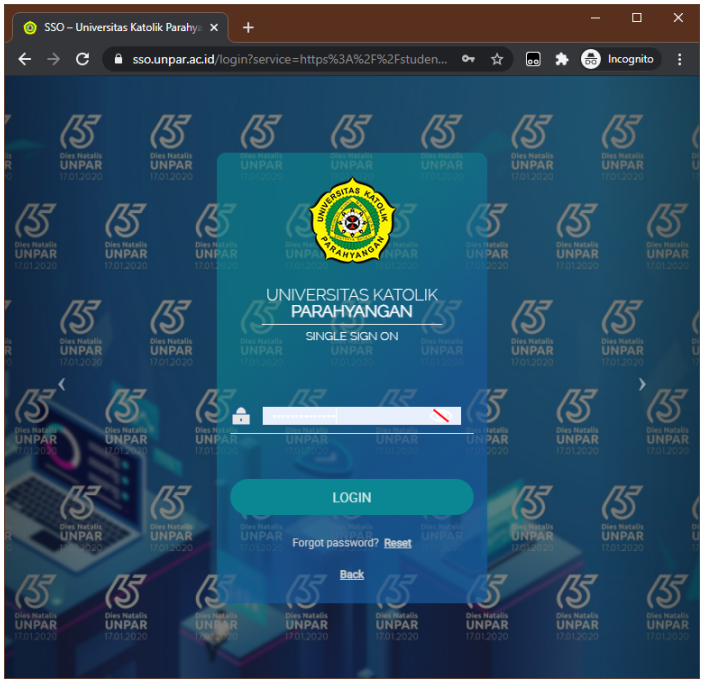
Aplikasi *screensaver* akan melakukan *http request* ke Portal Akademik Mahasiswa untuk mendapatkan data untuk setiap kebutuhan dari masing-masing fitur yang ada. Pengambilan data secara langsung dari Portal Akademik Mahasiswa dilakukan menggunakan *library* jsoup. Beberapa implementasi pemanfaatan jsoup untuk mengambil data-data tersebut sudah diimplementasikan pada skripsi Andrianto Sugiarto [9] sebelumnya. Data yang telah didapat dari Portal Akademik Mahasiswa kemudian diolah ke dalam SIAModels, dan ditampilkan sesuai dengan fitur-fitur yang ada pada aplikasi Mahasiswa Wali *Screensaver*.

3.1.1 *Login*

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



Gambar 3.1: Halaman *Login* 1



Gambar 3.2: Halaman *Login* 2

Login dilakukan dengan mengirim *request method* POST, dan kemudian mengambil session yang akan digunakan sebagai *cookies* apabila *login* berhasil. Terdapat beberapa perubahan yang terjadi pada situs Portal Akademik Mahasiswa semenjak skripsi Andrianto Sugiarto [9], yang mengakibatkan perlunya perubahan (Kode 3.1) terhadap implementasi jsoup:

1. Menyimpan atribut mahasiswa pada kelas tersebut agar tidak perlu melakukan *request login* berulang kali ketika mengakses menu-menu Portal Akademik Mahasiswa.
2. Menghapus pemanggilan fungsi validateTLSCertificates() dikarenakan sudah *deprecated* [5].

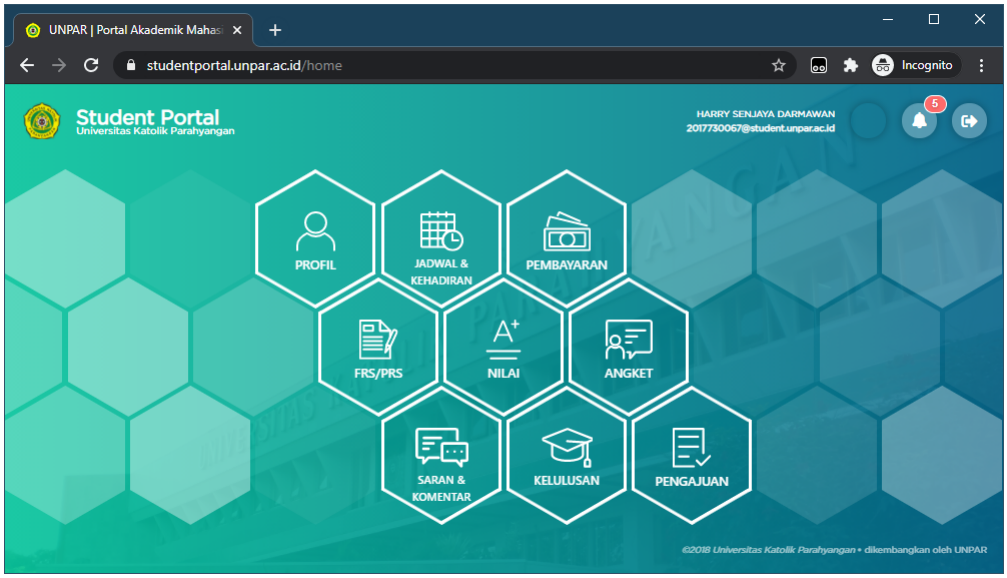
3. Menghapus pengambilan data dengan kueri css "input[name=lt]" dikarenakan kueri tersebut sudah dihapus oleh Portal Akademik Mahasiswa.
4. Menghapus atribut jsessionid dikarenakan tidak dibutuhkan.

Kode 3.1: Perubahan Implementasi Jsoup Login

```
@@ -38,40 +38,32 @@ public class Scraper {
        baseConn.execute();
}

-       public String login(String npm, String pass) throws IOException {
-           init();
-           Mahasiswa logged_mhs = new Mahasiswa(npm);
-           String user = logged_mhs.getEmailAddress();
-           Connection conn = Jsoup.connect(LOGIN_URL);
-           conn.data("Submit", "Login");
-           conn.timeout(0);
-           conn.validateTLSCertificates(false);
-           conn.method(Connection.Method.POST);
-           Response resp = conn.execute();
-           Document doc = resp.parse();
-           String lt = doc.select("input[name=lt]").val();
-           String execution = doc.select("input[name=execution]").val();
-           String jsessionid = resp.cookie("JSESSIONID");
-           /* SSO LOGIN */
-           Connection loginConn = Jsoup.connect(SSO_URL + ";jsessionid=" + jsessionid + "?service=" + LOGIN_URL);
-           loginConn.cookies(resp.cookies());
-           loginConn.data("username", user);
-           loginConn.data("password", pass);
-           loginConn.data("lt", lt);
-           loginConn.data("execution", execution);
-           loginConn.data("_eventId", "submit");
-           loginConn.data("submit", "");
-           loginConn.timeout(0);
-           loginConn.validateTLSCertificates(false);
-           loginConn.method(Connection.Method.POST);
-           resp = loginConn.execute();
-           if (resp.body().contains(user)) {
-                   Map<String, String> phpsessid = resp.cookies();
-                   return phpsessid.get("ci_session");
-           } else {
-                   return null;
-           }
-       }
+       public void login() throws IOException {
+           init();
+           this.mahasiswa = new Mahasiswa(this.npm);
+           String user = this.mahasiswa.getEmailAddress();
+           Connection conn = Jsoup.connect(LOGIN_URL);
+           conn.data("Submit", "Login");
+           conn.timeout(0);
+           conn.method(Connection.Method.POST);
+           Response resp = conn.execute();
+           Document doc = resp.parse();
+           String execution = doc.select("input[name=execution]").val();
+
+           /* SSO LOGIN */
+           Connection loginConn = Jsoup.connect(SSO_URL + "?service=" + LOGIN_URL);
+           loginConn.data("username", user);
+           loginConn.data("password", this.password);
+           loginConn.data("execution", execution);
+           loginConn.data("_eventId", "submit");
+           loginConn.timeout(0);
+           loginConn.method(Connection.Method.POST);
+           resp = loginConn.execute();
+           if (resp.body().contains(user)) {
+               Map<String, String> sessionId = resp.cookies();
+               this.session = sessionId.get("ci_session");
+           }
+       }
```

3.1.2 Halaman Utama

Pada Halaman Utama Portal Akademik Mahasiswa (Gambar 3.3), terdapat nama lengkap dan foto dari mahasiswa tersebut yang dapat diambil dan digunakan. Nama mahasiswa dapat diambil dengan mencari elemen "div" yang memiliki kelas "namaUser d-none d-lg-block mr-3", sehingga kueri css yang dihasilkan adalah "div[class=namaUser d-none d-lg-block mr-3]". Foto mahasiswa dapat diambil dengan mencari elemen "img" yang memiliki kelas "img-fluid fotoProfil", sehingga kueri css yang dihasilkan adalah "img[class=img-fluid fotoProfil]". Terdapat beberapa perubahan (Kode 3.2) yang perlu dilakukan terhadap skripsi Andrianto Sugiarto [9]:



Gambar 3.3: Halaman Utama Portal Akademik Mahasiswa

1. Menghapus pemanggilan fungsi validateTLSCertificates() dikarenakan sudah *deprecated* [5].
2. Menyimpan atribut mahasiswa pada kelas tersebut agar tidak perlu melakukan *request login* berulang kali ketika mengakses menu-menu Portal Akademik Mahasiswa.
3. Menghapus pengambilan semester yang sedang dijalani karena tidak akan ditampilkan pada *screensaver*.

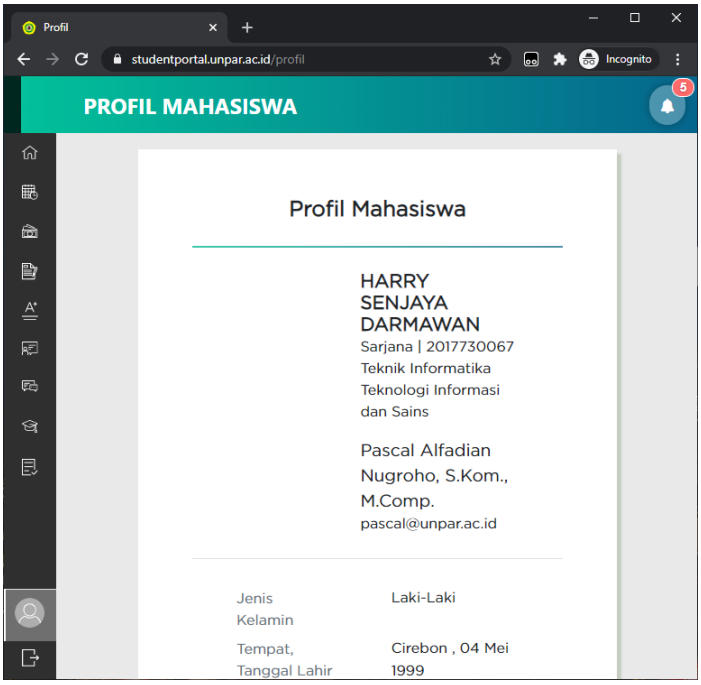
Kode 3.2: Perubahan Implementasi Jsoup Halaman Utama

```
@@ -73,31 +73,20 @@ public class Scraper {
        }
}

-       public TahunSemester requestNamePhotoTahunSemester(String phpsessid, Mahasiswa mhs) throws IOException {
+       public void requestNamePhotoTahunSemester(String phpsessid) throws IOException {
            Connection connection = Jsoup.connect(HOME_URL);
-           connection.cookie("ci_session", phpsessid);
-           connection.timeout(0);
-           connection.validateTLSCertificates(false);
-           connection.method(Connection.Method.GET);
-           Response resp = connection.execute();
-           Document doc = resp.parse();
-           String nama = doc.select("div[class=namaUser d-none d-lg-block mr-3]").text();
-           mhs.setNama(nama.substring(0, nama.indexOf(mhs.getEmailAddress())));
-           Element photo = doc.select("img[class=img-fluid fotoProfil]").first();
-           String photoPath = photo.attr("src");
-           mhs.setPhotoPath(photoPath);
-           connection = Jsoup.connect(NILAI_URL);
-           connection.cookie("ci_session", phpsessid);
-           connection.timeout(0);
-           connection.validateTLSCertificates(false);
-           connection.method(Connection.Method.GET);
-           resp = connection.execute();
-           doc = resp.parse();
-           Elements options = doc.getElementsByAttributeValue("name", "dropdownSemester").first().children();
-           String curr_sem = options.last().val();
-           curr_sem = curr_sem.substring(2,4).concat(curr_sem.substring(5));
-           TahunSemester currTahunSemester = new TahunSemester(curr_sem);
-           return currTahunSemester;
+           connection.cookie("ci_session", phpsessid);
+           connection.timeout(0);
+           connection.method(Connection.Method.GET);
+           Response resp = connection.execute();
+
+           Document doc = resp.parse();
+           String nama = doc.select("div[class=namaUser d-none d-lg-block mr-3]").text();
+           this.mahasiswa.setNama(nama.substring(0, nama.indexOf(this.mahasiswa.getEmailAddress())));
+
+           Element photo = doc.select("img[class=img-fluid fotoProfil]").first();
+           String photoPath = photo.attr("src");
+           this.mahasiswa.setPhotoPath(photoPath);
        }
```

Pada halaman utama Portal Akademik Mahasiswa juga terdapat beberapa menu yang dapat digunakan sebagai sumber data yaitu:

3.1.3 Profil

Menu Profil merupakan halaman yang menampilkan data diri mahasiswa (Gambar 3.4). Pada halaman ini tempat, dan tanggal lahir mahasiswa akan diambil untuk ditampilkan pada *screensaver*. Implementasi jsoup tersebut belum diimplementasikan pada skripsi Andrianto Sugiarto [9], sehingga perlu dilakukan penambahan fitur untuk mengambil data tersebut. Tempat, dan tanggal lahir mahasiswa dapat diambil dengan mencari elemen "div" yang memiliki kelas "offset-md-1 col-md-10 col-12 headerWrapper my-0 border-bottom", sehingga kueri css yang dihasilkan adalah "div[class=offset-md-1 col-md-10 col-12 headerWrapper my-0 border-bottom]".

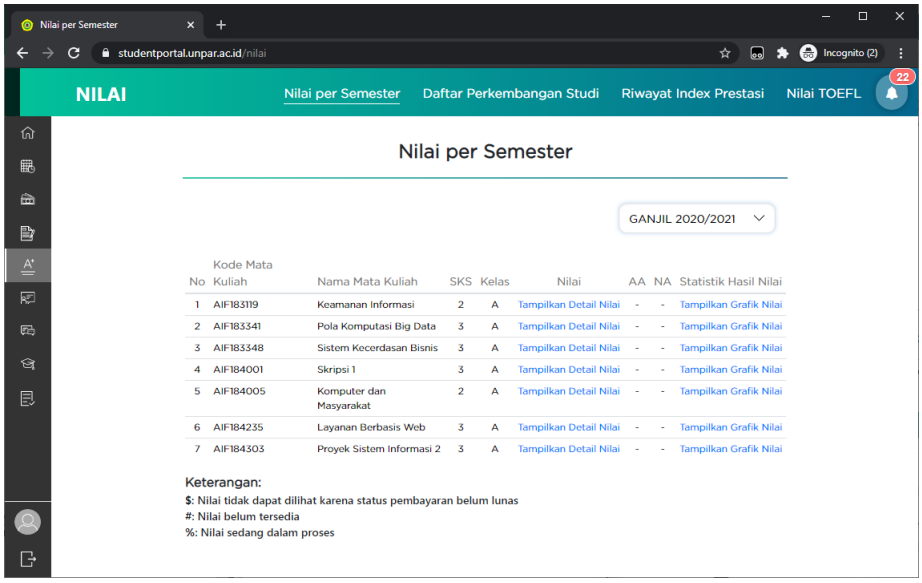


Gambar 3.4: Halaman Profil

3.1.4 Nilai

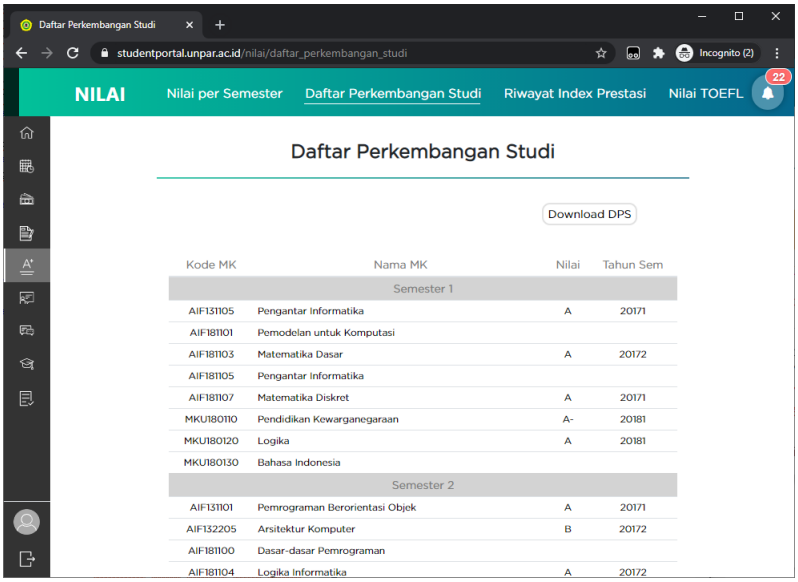
Menu Nilai terdiri dari beberapa submenu:

- Nilai per Semester
  Submenu ini menampilkan informasi nilai per semester. Mahasiswa dapat melihat nilai sesuai dengan semester yang dipilih (Gambar 3.5).

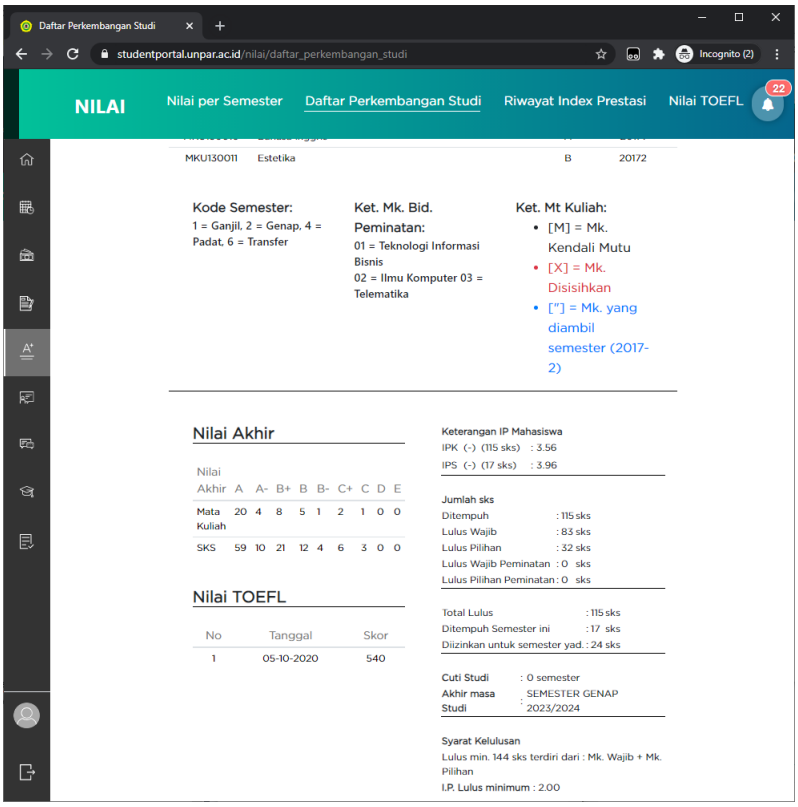


Gambar 3.5: Halaman Nilai Per Semester

- Daftar Perkembangan Studi
  Submenu ini menampilkan seluruh riwayat mata kuliah dan nilai yang pernah ditempuh mahasiswa (Gambar 3.6). Submenu ini juga menampilkan statistik sks, nilai, dan indeks prestasi mahasiswa (Gambar 3.7).



Gambar 3.6: Halaman Daftar Perkembangan Studi (1)



Gambar 3.7: Halaman Daftar Perkembangan Studi (2)

Data yang dapat dimanfaatkan dari halaman ini adalah IPK, IPS, jumlah sks yang lulus, dan jumlah sks yang ditempuh. Namun, pada SIAModels sudah terdapat *method* yang melakukan kalkulasi untuk mendapatkan data-data tersebut, sehingga tidak perlu dilakukan pengambilan data menggunakan jsoup. Untuk dapat memanfaatkan *method* tersebut diperlukan seluruh riwayat mata kuliah dan nilai yang pernah ditempuh mahasiswa, sehingga perlu dilakukan pengambilan data menggunakan jsoup. Implementasi pengambilan data tersebut sudah diimplementasikan sebelumnya pada skripsi Andrianto Sugiarto [9]. Proses dari pengambilan data tersebut yaitu:

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

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

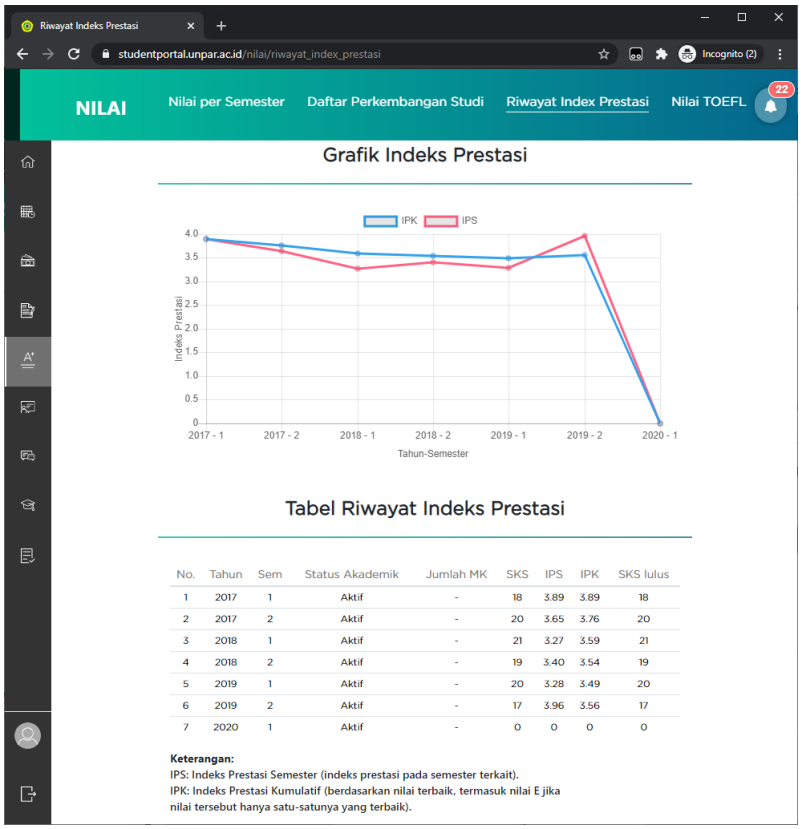
1. Menghapus pemanggilan fungsi validateTLSCertificates() dikarenakan sudah *deprecated* [5].
2. Indeks yang digunakan untuk melakukan keri css berubah, sehingga untuk mengantisipasi tersebut, indeks yang digunakan menjadi *size* dari kueri css dikurangi 1.

Kode 3.3: Perubahan Implementasi Jsoup Halaman Daftar Perkembangan Studi

```
@@ -175,7 +175,7 @@ public class Scraper {
        return jadwalList;
}
```

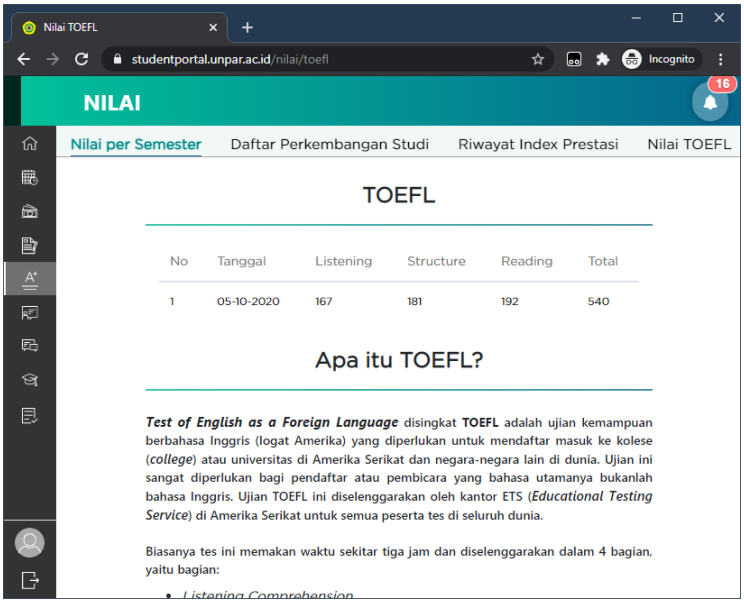
```
-   public void requestNilai(String phpsessid, Mahasiswa logged_mhs) throws IOException, InterruptedException {
+       public void requestNilai(String phpsessid) throws IOException, InterruptedException {
            Connection connection = Jsoup.connect(NILAI_URL);
-           connection.cookie("ci_session", phpsessid);
-           connection.timeout(0);
-           connection.validateTLSCertificates(false);
-           connection.method(Connection.Method.POST);
-           Response resp = connection.execute();
-           Document doc = resp.parse();
+           connection.cookie("ci_session", phpsessid);
+           connection.timeout(0);
+           connection.method(Connection.Method.POST);
+           Response resp = connection.execute();
+           Document doc = resp.parse();

-           Elements dropdownSemester = doc.select("#dropdownSemester option");
-           ArrayList<String> listSemester = new ArrayList<String>();
-           for (Element semester : dropdownSemester){
-                   listSemester.add(semester.attr("value"));
-           }
+           Elements dropdownSemester = doc.select("#dropdownSemester option");
+           ArrayList<String> listSemester = new ArrayList<String>();
+           for (Element semester : dropdownSemester){
+               listSemester.add(semester.attr("value"));
+           }

-           Thread[] threadUrl = new Thread[listSemester.size()-1];
-           for(int i = 0; i < listSemester.size()-1; i++){
-               threadUrl[i] = new Thread(new MultipleRequest(i, listSemester, NILAI_URL, phpsessid, logged_mhs));
-               threadUrl[i].start();
-           }
-           for(int i = 0; i < listSemester.size()-1; i++){
-               threadUrl[i].join();
-           }
-           Collections.sort(logged_mhs.getRiwayatNilai(), new Comparator<Nilai>() {
-                   @Override
-                   public int compare(Nilai o1, Nilai o2) {
-                           if (o1.getTahunAjaran() < o2.getTahunAjaran()) {
-                                   return -1;
-                           }
-                           if (o1.getTahunAjaran() > o2.getTahunAjaran()) {
-                                   return + 1;
-                           }
-                           if (o1.getSemester().getOrder() < o2.getSemester().getOrder()) {
-                                   return -1;
-                           }
-                           if (o1.getSemester().getOrder() > o2.getSemester().getOrder()) {
-                                   return +1;
-                           }
-                           return 0;
-                   }
-           });
+           Thread[] threadUrl = new Thread[listSemester.size()-1];
+           for(int i = 0; i < listSemester.size()-1; i++){
+                threadUrl[i] = new Thread(new MultipleRequest(i, listSemester, NILAI_URL, phpsessid, this.mahasiswa));
+                threadUrl[i].start();
+           }
+           for(int i = 0; i < listSemester.size()-1; i++){
+                threadUrl[i].join();
+           }
+           Collections.sort(this.mahasiswa.getRiwayatNilai(), new Comparator<Nilai>() {
+                @Override
+                public int compare(Nilai o1, Nilai o2) {
+                        if (o1.getTahunAjaran() < o2.getTahunAjaran()) {
+                                return -1;
+                        }
+                        if (o1.getTahunAjaran() > o2.getTahunAjaran()) {
+                                return + 1;
+                        }
+                        if (o1.getSemester().getOrder() < o2.getSemester().getOrder()) {
+                                return -1;
+                        }
+                        if (o1.getSemester().getOrder() > o2.getSemester().getOrder()) {
+                                return +1;
+                        }
+                        return 0;
+                }
+           });
        }
```

- Riwayat Indeks Prestasi
  Submenu ini menampilkan seluruh riwayat Indeks Prestasi Semester (IPS) dan Indeks Prestasi Kumulatif (IPK) setiap semester mahasiswa (Gambar 3.8).



Gambar 3.8: 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 3.9). Riwayat skor tersebut dapat diambil dengan mencari elemen "table" yang memiliki elemen "tbody" didalamnya, serta memiliki elemen "tr" didalam elemen "tbody". Terdapat beberapa perubahan yang terjadi pada halaman TOEFL semenjak skripsi Andrianto Sugiarto [9], yang mengakibatkan perlunya perubahan (Kode 3.4) terhadap implementasi jsoup:
  1. Menyimpan atribut mahasiswa pada kelas tersebut agar tidak perlu melakukan *request login* berulang kali ketika mengakses menu-menu Portal Akademik Mahasiswa.
  2. Menghapus pemanggilan fungsi validateTLSCertificates() dikarenakan sudah *deprecated* [5].
  3. Perubahan format tanggal TOEFL pada Portal Akademik Mahasiswa menjadi dd-mm-yyyy.



Gambar 3.9: Halaman Nilai TOEFL

Kode 3.4: Perubahan Implementasi Jsoup TOEFL

```
@@ -218,68 +218,29 @@ public class Scraper {
        });
}

-       public void requestNilaiTOEFL(String phpsessid, Mahasiswa mahasiswa) throws IOException {
-           SortedMap<LocalDate, Integer> nilaiTerakhirTOEFL = new TreeMap<>();
-           Connection connection = Jsoup.connect(TOEFL_URL);
-           connection.cookie("ci_session", phpsessid);
-           connection.timeout(0);
-           connection.validateTLSCertificates(false);
-           connection.method(Connection.Method.POST);
-           Response resp = connection.execute();
-           Document doc = resp.parse();
-           Elements nilaiTOEFL = doc.select("table").select("tbody").select("tr");
-           if (!nilaiTOEFL.isEmpty()) {
-               for (int i = 0; i < nilaiTOEFL.size(); i++) {
-                   Element nilai = nilaiTOEFL.get(i).select("td").get(5);
-                   Element tgl_toefl = nilaiTOEFL.get(i).select("td").get(1);
-                   String[] tanggal = tgl_toefl.text().split(" ");
-                   switch (tanggal[1].toLowerCase()) {
-                   case "januari":
-                       tanggal[1] = "1";
-                       break;
-                   case "februari":
-                       tanggal[1] = "2";
-                       break;
-                   case "maret":
-                       tanggal[1] = "3";
-                       break;
-                   case "april":
-                       tanggal[1] = "4";
-                       break;
-                   case "mei":
-                       tanggal[1] = "5";
-                       break;
-                   case "juni":
-                       tanggal[1] = "6";
-                       break;
-                   case "juli":
-                       tanggal[1] = "7";
-                       break;
-                   case "agustus":
-                       tanggal[1] = "8";
-                       break;
-                   case "september":
-                       tanggal[1] = "9";
-                       break;
-                   case "oktober":
-                       tanggal[1] = "10";
-                       break;
-                   case "november":
-                       tanggal[1] = "11";
-                       break;
-                   case "desember":
-                       tanggal[1] = "12";
-                       break;
-                   }
+       public void requestNilaiTOEFL(String phpsessid) throws IOException {
+           SortedMap<LocalDate, Integer> nilaiTerakhirTOEFL = new TreeMap<>();
```

```
+           Connection connection = Jsoup.connect(TOEFL_URL);
+           connection.cookie("ci_session", phpsessid);
+           connection.timeout(0);
+           connection.method(Connection.Method.POST);
+           Response resp = connection.execute();
+           Document doc = resp.parse();
+           Elements nilaiTOEFL = doc.select("table").select("tbody").select("tr");
+           if (!nilaiTOEFL.isEmpty()) {
+               for (int i = 0; i < nilaiTOEFL.size(); i++) {
+                   Element nilai = nilaiTOEFL.get(i).select("td").get(5);
+                   Element tgl_toefl = nilaiTOEFL.get(i).select("td").get(1);
+                   String[] tanggal = tgl_toefl.text().split("-");

-                   LocalDate localDate = LocalDate.of(Integer.parseInt(tanggal[2]), Integer.parseInt(tanggal[1]),
-                                         Integer.parseInt(tanggal[0]));
+                   LocalDate localDate = LocalDate.of(Integer.parseInt(tanggal[2]), Integer.parseInt(tanggal[1]),
+                                         Integer.parseInt(tanggal[0]));

-                   nilaiTerakhirTOEFL.put(localDate, Integer.parseInt(nilai.text()));
-               }
-           }
-           mahasiswa.setNilaiTOEFL(nilaiTerakhirTOEFL);
-       }
+                   nilaiTerakhirTOEFL.put(localDate, Integer.parseInt(nilai.text()));
+               }
+           }
+           this.mahasiswa.setNilaiTOEFL(nilaiTerakhirTOEFL);
+       }
```

DAFTAR REFERENSI

[1] Wikipedia (2021) Screensaver. https://en.wikipedia.org/wiki/Screensaver. 07 Maret 2021.

[2] Alfadian, P. (2016) SI Akademik. http://bti.unpar.ac.id/akademik/. 19 Oktober 2020.

[3] 2018, T. P. P. A. M. P. (2018) BUKU PANDUAN PENGGUNAAN PORTAL AKADEMIK MAHASISWA (PAM) 2018. https://studentportal.unpar.ac.id/assets/BUKU_PANDUAN_PENGGUNAAN_FRS_GABUNGAN.pdf. 28 Februari 2021.

[4] Alfadian, P. (2020) SIAModels. https://github.com/pascalalfadian/SIAModels. 19 Oktober 2020.

[5] Version 1.13.1 (2009-2020) *jsoup: Java HTML Parser*. Jonathan Hedley. Seattle, Washington.

[6] Zhao, B. (2017) Web Scraping. Bagian dari Schintler, L. A. dan McNeely, C. L. (ed.), *Encyclopedia of Big Data*. Springer International Publishing.

[7] Cokrowibowo, S., Nur, N., dan Irmayanti (2020) Web page extraction and classification using jsoup and naïve bayes. *IOP Conference Series: Materials Science and Engineering*, **875**, 012089.

[8] Version 15 (2008-2020) *JavaFX 15*. Gluon HQ. Industrieweg 3, Leuven, BE 3001, BE.

[9] Sugiarto, A. (2018) PENYESUAIAN SIAMODELS DAN IFSTUDENTPORTAL KE KURIKULUM 2018. Skripsi. Universitas Katolik Parahyangan, Indonesia.

LAMPIRAN A

KODE PROGRAM

Kode A.1: MyCode.c

```c

// This does not make algorithmic sense,
// but it shows off significant programming characters.

#include<stdio.h>

void myFunction( int input, float* output ) {
    switch ( array[i] ) {
        case 1: // This is silly code
            if ( a >= 0 || b <= 3 && c != x )
                *output += 0.005 + 20050;
            char = 'g';
            b = 2^n + ~right_size - leftSize * MAX_SIZE;
            c = (--aaa + &daa) / (bbb++ - ccc % 2 );
            strcpy(a,"hello $@?");
    }
    count = ~mask | 0x00FF00AA;
}

// Fonts for Displaying Program Code in LATEX
// Adrian P. Robson, nepsweb.co.uk
// 8 October 2012
// http://nepsweb.co.uk/docs/progfonts.pdf
```

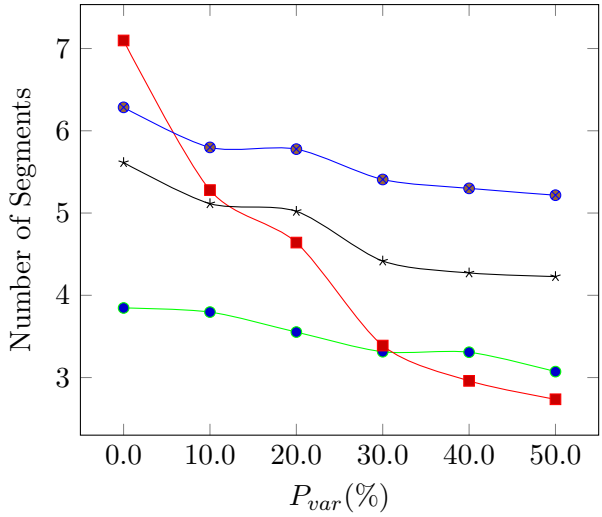
Kode A.2: MyCode.java

```java
import java.util.ArrayList;
import java.util.Collections;
import java.util.HashSet;

//class for set of vertices close to furthest edge
public class MyFurSet {
    protected int id;                                   //id of the set
    protected MyEdge FurthestEdge;                      //the furthest edge
    protected HashSet<MyVertex> set;                    //set of vertices close to furthest edge
    protected ArrayList<ArrayList<Integer>> ordered;    //list of all vertices in the set for each trajectory
    protected ArrayList<Integer> closeID;               //store the ID of all vertices
    protected ArrayList<Double> closeDist;              //store the distance of all vertices
    protected int totaltrj;                             //total trajectories in the set

    /*
     * Constructor
     * @param id : id of the set
     * @param totaltrj : total number of trajectories in the set
     * @param FurthestEdge : the furthest edge
     */
    public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
        this.id = id;
        this.totaltrj = totaltrj;
        this.FurthestEdge = FurthestEdge;
        set = new HashSet<MyVertex>();
        ordered = new ArrayList<ArrayList<Integer>>();
        for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
        closeID = new ArrayList<Integer>(totaltrj);
        closeDist = new ArrayList<Double>(totaltrj);
        for (int i = 0;i <totaltrj;i++) {
            closeID.add(-1);
            closeDist.add(Double.MAX_VALUE);
        }
    }

}
```

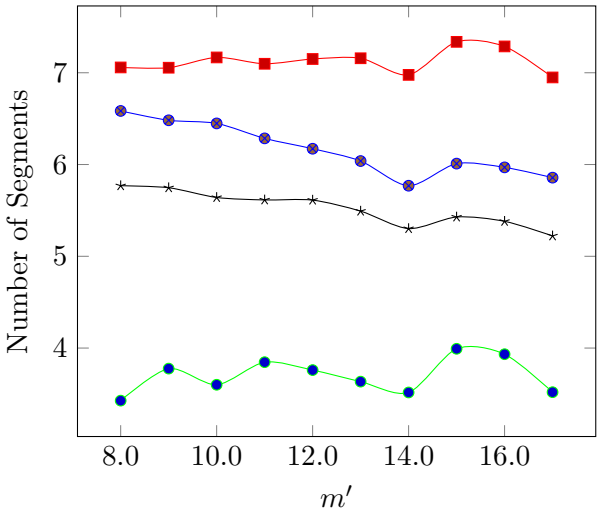
LAMPIRAN B

HASIL EKSPERIMEN

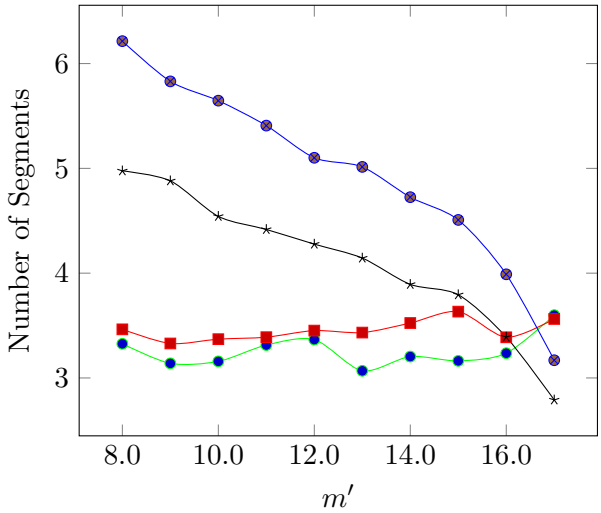
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



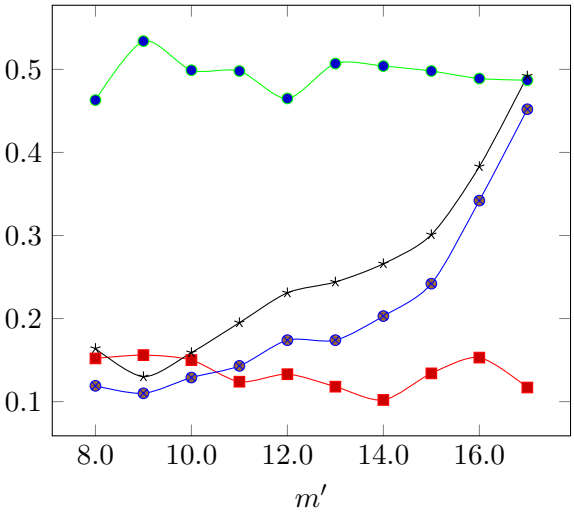
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4