BAB 1

PENDAHULUAN

1.1 Latar Belakang

Electron adalah salah satu kerangka kerja terbaru yang memungkinkan pengembang membuat aplikasi *desktop* asli dengan teknologi web populer : JavaScript, HTML5, dan CSS. Dengan *Electron*, pengembang web dapat menggunakan keterampilan yang mereka miliki untuk membangun aplikasi yang memiliki banyak kemampuan seperti aplikasi *desktop* asli. *Elektron* telah menjadi sangat populer sejak dirilis dan digunakan oleh perusahaan, seperti : *Microsoft*, *Facebook*, *Slack*, dan *Docker*. Aplikasi ini dapat dikemas untuk dapat berjalan langsung di *macOS*, *Windows*, dan *Linux*. Bisa juga didistribusikan melalui *Mac App Store* atau *Microsoft Store*.

Vis.js adalah sebuah *library* visualisasi berbasis *browser* yang bersifat dinamis. *Library* ini dirancang agar mudah digunakan untuk menangani data dinamis dalam jumlah yang besar dan memungkinkan untuk memanipulasi serta berinteraksi dengan data tersebut. *Library* ini terdiri dari komponen - komponen, seperti *DataSet*, *Timeline*, *Network* (*tree*), *Graph2d*, dan *Graph3d*. *DataSet* berfungsi untuk mengelola data yang tidak terstruktur. Terdapat fitur *add*, *update*, dan *remove* data di dalamnya. *Timeline* berfungsi untuk menampilkan data dalam bentuk *timeline* yang dapat disesuaikan dengan *item* dan rentangnya. *Network* (*tree*) berfungsi untuk menampilkan data dalam bentuk jaringan yang dinamis, dapat diatur secara otomatis, dan dapat disesuaikan. *Graph2d* berfungsi untuk menampilkan data dalam bentuk grafik dan diagram batang pada *timeline* yang interaktif sesuai yang diinginkan. *Graph3d* berfungsi untuk menampilkan data dalam bentuk grafik 3d dengan animasi yang interaktif.

GitHub adalah sebuah aplikasi berbasis *website* dengan *Version Control System* (VCS) yang menyediakan layanan untuk menyimpan *repository* dengan gratis. VCS adalah sebuah infrastruktur yang dapat mendukung pengembangan *software* secara kolaboratif. Setiap anggota yang berada di dalam sebuah tim pengembangan *software* dapat menulis kode programnya masing - masing kemudian digabungkan ke server yang sudah memiliki VCS yang digunakan. *Respository* merupakan tempat yang dapat digunakan untuk menyimpan berbagai file berupa *source code*. Aplikasi ini termasuk sangat populer dan banyak digunakan termasuk oleh perusahaan - perusahaan besar, seperti : *Facebook*, *Google*, dan *Twitter*.

Saat mahasiswa akan melakukan *FRS*, seringkali mereka kesulitan untuk melihat kurikulum tahun ajaran yang berlaku. Karena setiap kurikulum memiliki aturan yang berbeda dalam pengambilan matakuliah ataupun matakuliah yang disediakan, maka mahasiswa kadang bingung untuk memilih matakuliah apa yang akan diambil di semester berikutnya.

Maka dari itu, pada skripsi ini akan dibuat sebuah aplikasi visualisasi kurikulum 2018 berbasis *Electron* dengan menggunakan *library Vis.js*. Dengan aplikasi ini diharapkan mahasiswa dapat lebih mudah untuk melihat kurikulum yang ada sehingga mempermudah mereka untuk meemilih matakuliah apa yang akan diambil di semester berikutnya, kemudian dengan penggunaan *framework cross platform Electron* ini diharapkan semua mahasiswa pengguna *macOs*, *Windows*, dan *Linux* dapat mengaksesnya dengan mudah.

1.2 Rumusan Masalah

Rumusan masalah yang akan dibahas pada skripsi ini adalah :

1. Bagaimana cara memvisualisasikan kurikulum 2018 dalam bentuk *tree*?
2. Bagaimana cara memvisualisasikan kurikulum 2018 dalam bentuk *timeline*?
3. Bagaimana cara membaca kurikulum 2018 FTIS UNPAR dari *github*?

1.3 Tujuan

Tujuan yang akan dicapai dari penulisan skripsi ini adalah :

1. Memahami cara memvisualisasikan kurikulum 2018 dalam bentuk *tree*.
2. Memahami cara memvisualisasikan kurikulum 2018 dalam bentuk *timeline*.
3. Memahami cara membaca data kurikulum 2018 FTIS UNPAR dari *github*.

1.4 Batasan Masalah

Batasan-batasan masalah yang ditetapkan adalah sebagai berikut :

1. Kurikulum 2018 yang digunakan hanya milik program studi teknik informatika.
2. Perangkat lunak ini hanya memvisusalisasikan dalam bentuk *tree* dan *timeline*.
3. Perangkat lunak ini hanya akan menggunakan bahasa *HTML*, *PHP* dan *JavaScript*

1.5 Metodologi

Bagian - bagian pengerjaan skripsi ini adalah :

1. Melakukan studi tentang *framework Electron* dan *Library Vis.js*.
2. Mempelajari cara membuat aplikasi berbasis *Electron*.
3. Mempelajari cara memvisualisasikan data dalam bentuk *tree* dan *timeline* dengan *Vis.js*.
4. Mempelajari data kurikulum 2018 di github beserta cara pengambilan datanya.
5. Merancang aplikasi berbasis *Electron*.
6. Merancang visualisasi kurikulum 2018 dalam bentuk *tree*.
7. Merancang visualisasi kurikulum 2018 dalam bentuk *timeline*.
8. Mendesain antarmuka aplikasi.
9. Melakukan pengujian dan ekperimen.
10. Membuat dokumen skripsi.

1.6 Sistematika Pembahasan

Skripsi ini terdiri dari enam bab, yaitu pendahuluan, landasan teori, analisis, perancangan, implementasi, dan kesimpulan dan saran.

Bab I membahas latar belakang dibuatnya skripsi, rumusan masalah yang terdapat pada skripsi, tujuan skripsi ini dibuat, batasan masalah agar skripsi yang dibuat tidak terlalu luas, dan metodologi yang berisi langkah - langkah pengerjaan skripsi agar berjalan sistematis.

Bab II berisi teori - teori yang berfungsi sebagai referensi dalam pembuatan skripsi dan membantu dalam menyelesaikan masalah pada skripsi.

Bab III berisi analisis terhadap perangkat lunak yang telah dibuat.

Bab IV berisi perancangan perangkat lunak menggunakan aplikasi *Electrorn* dan *library Vis.js*.

Bab V berisi implementasi perangkat lunak yang berlandaskan teori - teori yang telah dipelajari.

Bab VI berisi kesimpulan skripsi yang telah dibuat dan juga saran yang ditujukan untuk skripsi berikutnya.

BAB 2

LANDASAN TEORI

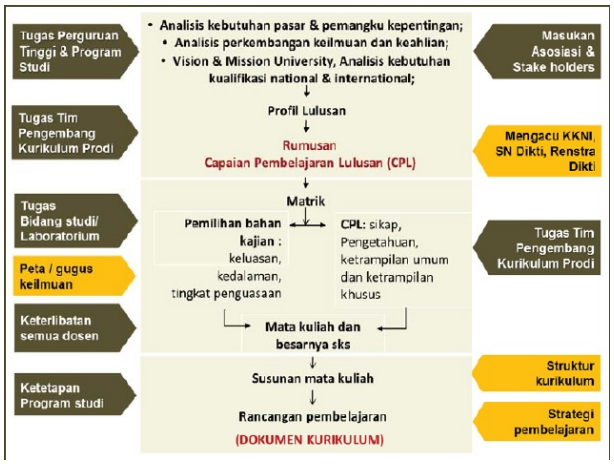
2.1 Kurikulum 2018

Kurikulum didefinisikan sebagai seperangkat rencana dan pengaturan mengenai capaian pembelajaran lulusan, bahan kajian, proses, dan penilaian yang digunakan sebagai pedoman penyelenggaraan program studi menjadi sarana utama untuk mencapai tujuan tersebut. [1]

Penyusunan Kurikulum 2018 berpegang pada prinsip bahwa kurikulum yang baik adalah kurikulum yang tidak hanya kokoh, secara teoretis konseptual dapat dipertanggungjawabkan, namun juga secara praktis dapat dilaksanakan. Selain itu kurikulum juga harus cukup fleksibel agar dapat mengakomodasi perubahan-perubahan, namun tanpa kehilangan ciri atau kekhasan dari program studi.

Dalam penyusunan Kurikulum 2018 Program Studi Teknik Informatika secara khusus juga memperhatikan Kerangka Kualifikasi Nasional Indonesia (KKNI) yang tertuang dalam Peraturan Presiden no 8 tahun 2012. KKNI merupakan pernyataan kualitas SDM Indonesia, di mana tolok ukur kualifikasinya ditetapkan berdasarkan capaian pembelajaran (learning outcomes) yang dimilikinya. Penyusunan kurikulum mengikuti tahapan perancangan kurikulum yang disarankan oleh Kemenristekdikti yang diberikan pada Gambar 2.1. Tahapan penyusunan kurikulum 2018 meliputi kegiatan sebagai berikut:

1. Melakukan evaluasi diri
2. Merumuskan profil lulusan dengan pelacakan lulusan
3. Menentukan capaian pembelajaran
4. Menentukan bahan kajian
5. Menyusun matriks pembelajaran dan bahan kajian
6. Membentuk mata kuliah
7. Menyusun struktur kurikulum dan menentukan metode pembelajaran



Gambar 2.1: Tahapan penyusunan kurikulum

2.1.1 Kodifikasi

Kodifikasi tiap mata kuliah dibuat berdasarkan Peraturan Rektor UNPAR No. III/PRT/2017-03/46 tentang Standar Penyusunan Kurikulum Program Studi di Lingkungan UNPAR. Kode ini terdiri atas 11 dijit, dengan rincian berikut:

- 3 digit - kode khas Program Studi: AIF
- 2 digit - tahun diberlakukannya kurikulum (2 digit terakhir): 18
- 1 digit - urutan tahun pengajaran
- 1 digit - nomor urut KBI pengampu mata kuliah
- 2 digit - nomor urut mata kuliah per semester, dengan angka pada dijit terakhir sebagai penentu semester; ganjil atau genap
- 2 digit - jumlah sks mata kuliah

Informasi lengkap terkait kodifikasi ini diberikan di Gambar 2.2.

Penyelenggara	Universitas	Prodi
Kode khas prodi	MKU	AIF
Tahun berlaku kurikulum	18	18
Urutan tahun pengajaran	0	1: tahun pertama 2: tahun kedua 3: tahun ketiga 4: tahun keempat
Nomor urut KBI pengampu	**	0: Prodi 1: Teori Komputasi 2: Sistem Terdistribusi 3: Sistem Informasi
Nomor urut mata kuliah	**	Urutan mata kuliah per semester, dengan angka pada dijit terakhir sebagai penentu semester; ganjil atau genap
Jumlah sks	**	Jumlah sks

**Kode mata kuliah MKU ditentukan oleh universitas

Gambar 2.2: Kodifikasi mata kuliah

2.1.2 Bobot Pemrograman

Berdasarkan hasil evaluasi Kurikulum 2013, salah satu masalah yang ditemukan adalah bahwa mahasiswa masih sulit menguasai materi kuliah di jalur pemrograman, yang merupakan kuliah inti dari Prodi Teknik Informatika UNPAR. Selain karena memang logika pemrograman tidak mudah untuk dipahami, kurangnya pengalaman mahasiswa dalam membangun program komputer juga menjadi penyebab munculnya permasalahan ini. [1]

Selain memperbaiki struktur kuliah jalur pemrograman, dan perbaikan materi perkuliahan, cara lain yang digunakan untuk mendukung kemampuan pemrograman mahasiswa adalah dengan menempatkan bobot pemrograman di kuliah-kuliah yang cocok. Bobot pemrograman ini menentukan di kuliah mana saja mahasiswa harus membangun program komputer, dan seberapa besar skala program komputer yang dibuat. Bagian pembangunan program komputer misalnya dapat diletakkan pada saat praktikum, atau dijadikan bagian dari tugas kuliah.

Besar bobot pemrograman dalam kurikulum ini adalah 0.25, 0.5, 0.75, dan 1. Penjelasan terkait masing - masing bobot ini diberikan pada Gambar 2.3.

Bobot	Deskripsi
0.25	• Minimal 1 tugas berbentuk pembangunan program komputer • Kuliah tidak berpraktikum
0.5	• Minimal setengah dari tugas yang diberikan berbentuk pembangunan program komputer • Kuliah tidak berpraktikum atau yang berfokus pada analisis
0.75	• Di luar tugas praktikum, ada tugas kuliah berupa pembangunan program komputer • Kuliah berpraktikum atau merupakan kuliah skripsi
1	Kuliah berpraktikum dengan capaian pembelajaran adalah keahlian pemrograman atau merupakan kuliah proyek

Gambar 2.3: Rincian Bobot Pemrograman

2.1.3 Prasyarat Mata Kuliah

Di Prodi Teknik Informatika terdapat 2 jenis prasyarat, yaitu prasyarat lulus dan prasyarat tempuh. Prasyarat lulus artinya seorang mahasiswa harus lulus mata kuliah prasyarat (nilai minimum D), baru dapat mengambil suatu mata kuliah, sedangkan prasyarat tempuh artinya seorang mahasiswa harus pernah menempuh mata kuliah prasyarat, sebelum dapat mengambil suatu mata kuliah. Rincian prasyarat mata kuliah wajib diberikan pada Gambar 2.4, Gambar 2.5, Gambar 2.6, Gambar 2.7, Gambar 2.8, Gambar 2.9, Gambar 2.10, dan Gambar 2.11, sedangkan rincian prasayarat mata kuliah pilihan diberikan pada Gambar 2.12, Gambar 2.13, Gambar 2.14, dan Gambar 2.15.

No	Kode	Mata Kuliah	Mata Kuliah Prasyarat	
			Tempuh	Lulus
Semester 1				
1	AIF181101-03	Pemodelan untuk Komputasi		
2	AIF181103-04	Matematika Dasar		
3	AIF181105-02	Pengantar Informatika		
4	AIF181107-03	Matematika Diskret		
5	MKU180130-02	Bahasa Indonesia		
6	MKU180110-02	Pendidikan Kewarganegaraan		
7	MKU180120-02	Logika		

Gambar 2.4: Daftar mata kuliah wajib semester satu beserta prasyaratnya

Semester 2				
1	AIF181100-04	Dasar Pemrograman		Mulai angkatan 2018: AIF181101-03
2	AIF181202-04	Arsitektur dan Organisasi Komputer		
3	AIF181104-03	Logika Informatika		
4	AIF181106-03	Matriks dan Ruang Vektor		
5	MKU180240-02	Etika		
6	MKU180360-02	Estetika		

Gambar 2.5: Daftar mata kuliah wajib semester dua beserta prasyaratnya

Semester 3				
1	AIF182101-03	Algoritma dan Struktur Data		AIF181100-04
2	AIF182103-04	Struktur Diskret	AIF181107-03	
3	AIF182105-02	Pemrograman Berorientasi Objek		AIF181100-04
4	AIF182007-02	Teknik Presentasi		
5	AIF182109-03	Statistika untuk Komputasi		
6	MKU180370-02 / MKU180380-02	Agama Katolik/Fenomenologi Agama		
7	MKU180250-02	Pancasila		

Gambar 2.6: Daftar mata kuliah wajib semester tiga beserta prasyaratnya

Semester 4				
1	AIF182100-04	Analisis dan Desain Perangkat Lunak		AIF182105-02
2	AIF182302-04	Manajemen Informasi dan Basis Data	AIF182101-03	
3	AIF182204-03	Pemrograman Berbasis Web	AIF182302-04 (bersamaan atau sudah tempuh)	
4	AIF182106-03	Desain dan Analisis Algoritma	AIF182103-04	AIF182101-03
5	AIF182308-03	Pengantar Sistem Informasi	AIF182302-04 (bersamaan atau sudah tempuh)	AIF181105-02
6	AIF182210-02	Pengantar Jaringan Komputer		

Gambar 2.7: Daftar mata kuliah wajib semester empat beserta prasyaratnya

Semester 5				
1	AIF183201-03	Sistem Operasi	AIF182101-03	
2	AIF183303-03	Rekayasa Perangkat Lunak	AIF182100-04	
3	AIF183305-02	Manajemen Proyek	AIF183303-03 (bersamaan atau sudah tempuh)	
4	AIF183107-03	Pengantar Sistem Cerdas	AIF182106-03 AIF181104-03	
5	AIF183209-03	Pemrograman pada Perangkat Bergerak	AIF182210-02 AIF182100-04	
6	AIF183111-03	Interaksi Manusia Komputer		

Gambar 2.8: Daftar mata kuliah wajib semester lima beserta prasyaratnya

Semester 6				
1	AIF183300-02	Teknologi Basis Data		AIF182302-04
2	AIF183002-02	Penulisan Ilmiah		
3	AIF183204-02	Jaringan Komputer	AIF182210-02	
4	AIF183106-06	Proyek Informatika	AIF183305-02	
	AIF183308-03	Proyek Sistem Informasi 1	AIF183305-02	AIF182308-03
	AIF183310-03	Proyek Data Science 1	AIF183305-02	AIF182109-03

Gambar 2.9: Daftar mata kuliah wajib semester enam beserta prasyaratnya

Semester 7				
1	AIF184001-03	Skripsi 1		AIF183002-02 Sudah lulus 108 sks Mulai angkatan 2017: AIF183002-02 AIF182007-02
				Sudah lulus 108 sks
2	AIF184303-03	Proyek Sistem Informasi 2		AIF183308-03
	AIF184305-03	Proyek Data Science 2		AIF183310-03
3	AIF184005-02	Komputer dan Masyarakat		

Gambar 2.10: Daftar mata kuliah wajib semester tujuh beserta prasyaratnya

Semester 8				
1	AIF184000-02	Etika Profesi		
2	AIF184002-05	Skripsi 2		AIF184001-03 Jika diambil bersamaan dengan AIF184001-03 Prasyarat: lulus AIF183002-02 AIF182007-02 dan lulus 124 sks
3	AIF184004-08	Tugas Akhir		AIF183002-02 Sudah lulus 124 sks Mulai angkatan 2017: AIF183002-02 AIF182007-02 Sudah lulus 124 sks

Gambar 2.11: Daftar mata kuliah wajib semester delapan beserta prasyaratnya

No	Kode	Mata Kuliah	Prasyarat	
			Tempuh	Lulus
1	AIF182111-03	Pemrograman Kompetitif 1		AIF182101-03 (minimum C)
2	AIF182001-03	Penelitian 1		AIF181100-04
3	AIF182301-03	Pengantar Data Science		
4	AIF182112-03	Pemrograman Kompetitif 2		AIF182111-03 (minimum B)
5	AIF182102-03	Statistika dengan R		AIF182109-03
6	AIF182002-03	Penelitian 2		AIF181100-04
7	AIF183013-02	Kerja Praktek 1		
8	AIF183015-03	Pendidikan Pengabdian kepada Masyarakat		

Gambar 2.12: Daftar mata kuliah pilihan beserta prasyaratnya (1)

No	Kode	Mata Kuliah	Prasyarat	
			Tempuh	Lulus
9	AIF183117-02	Grafika Komputer	AIF181103-04	AIF182105-02
10	AIF183119-02	Keamanan Informasi		AIF181107-03
11	AIF183121-03	Pemrograman Kompetitif 3		AIF182112-03 (minimum B)
12	AIF183123-02	Topik Khusus Informatika 1		
13	AIF183113-03	Statistika Multivariat dengan R	AIF182102-03	
14	AIF183003-03	Penelitian 3		AIF181100-04
15	AIF183225-03	Sertifikasi Administrasi Jaringan Komputer 1		
16	AIF183229-02	Topik Khusus Sistem Terdistribusi 1		
17	AIF183331-03	Sistem e-Commerce		AIF182308-03
18	AIF183329-03	Basis Data dan Pemrograman SQL untuk Big Data		AIF182302-04 AIF182210-02
19	AIF183335-03	Pengantar Penambangan Data dengan Python	AIF182302-04 (atau tempuh bersama) AIF182109-03 (atau tempuh bersama)	
20	AIF183337-02	Topik Khusus Sistem Informasi 1		
21	AIF183339-02	Sertifikasi Perancangan dan Pemrograman Basis Data dengan Oracle	AIF182302-04	
22	AIF183341-03	Pola Komputasi Big Data	AIF182101-03 AIF182210-02	
23	AIF183143-03	Pemodelan Formal		AIF181104-03
24	AIF183147-02	Sertifikasi Dasar-dasar Java	AIF182105-02	
25	AIF183155-03	Metode Numerik		AIF181103-04 AIF181100-04
26	AIF183203-02	Internet of Things		AIF182210-02
27	AIF183010-03	Kerja Praktek 2		
28	AIF183112-02	Pengujian Perangkat Lunak		AIF183303-03
29	AIF183114-03	Algoritma Kriptografi	AIF183119-02	
30	AIF183116-02	Komputasi Paralel		AIF182101-03

Gambar 2.13: Daftar mata kuliah pilihan beserta prasyaratnya (2)

No	Kode	Mata Kuliah	Prasyarat	
			Tempuh	Lulus
31	AIF183120-03	Pemrograman Permainan Komputer		AIF182101-03 (minimum B)
32	AIF183122-03	Pemodelan Simulasi	AIF182101-03	
33	AIF183128-03	Topik Khusus Informatika 2		
34	AIF183232-03	Pemrograman Berbasis Web Lanjut		AIF182204-03 AIF182302-04
35	AIF183236-03	Sertifikasi Administrasi Jaringan Komputer 2		AIF183225-03
36	AIF183238-03	Topik Khusus Sistem Terdistribusi 2		
37	AIF183240-03	Sertifikasi Cyber Ops		AIF182210-02
38	AIF183342-03	Kewirausahaan Berbasis Teknologi		Sudah lulus 90 sks
39	AIF183346-03	Topik Khusus Sistem Informasi 2		
40	AIF183348-03	Sistem Kecerdasan Bisnis	AIF182302-04	
41	AIF183350-02	IBM Professional Data Science Certificate 1		AIF183335-03
42	AIF184007-04	Kerja Praktek 3		
43	AIF184109-03	Pembelajaran Mesin		AIF183107-03
44	AIF184115-02	Pencarian dan Temu Kembali Informasi		AIF181103-04
45	AIF184119-02	Perancangan Permainan Komputer		AIF182100-04 (minimum B) AIF183120-03 (minumum B)
46	AIF184123-03	Teknologi Mesin Pencari	AIF181106-03	
47	AIF184125-03	Pengolahan Bahasa Alami		AIF183107-03
48	AIF184129-03	Sertifikasi Administrasi Jaringan Komputer 3		AIF183236-03
49	AIF184235-03	Layanan Berbasis Web		AIF182204-03 AIF182302-04 AIF183204-02
50	AIF184339-03	Pengendalian dan Audit Teknologi Informasi	AIF182308-03	
51	AIF184341-03	Penambangan Data		AIF182101-03

Gambar 2.14: Daftar mata kuliah pilihan beserta prasyaratnya (3)

No	Kode	Mata Kuliah	Prasyarat	
			Tempuh	Lulus
52	AIF184337-03	IBM Professional Data Science Certificate 2		AIF183350-02
53	AIF184351-03	Analisis Big Data	AIF183335-03 AIF183341-03 (atau tempuh bersama)	
54	AIF184247-03	Jaringan Komputer Lanjut		AIF183204-02
55	AIF184006-05	Kerja Praktek 4		
56	AIF184106-02	Analisis Data Permainan Komputer		AIF184119-02 (minimum B)
57	AIF184116-02	Sistem Multi Agen	AIF183201-03 AIF183107-03	
58	AIF184222-03	Sertifikasi Administrasi Jaringan Komputer 4		AIF184129-03
59	AIF184334-03	Sistem Informasi Skala Besar		AIF182308-03
60	AIF184338-03	Manajemen Proses Bisnis	AIF182105-02 AIF182204-03	
61	AIF184332-03	Teknologi Big Data dan Cloud Computing	AIF184351-03	
62	AIF184352-03	IBM Professional Data Science Certificate 3		AIF184337-03
63	AIF184330-02	Big Data dan Machine Learning dengan Google Cloud Platform	AIF184351-03 (atau tempuh bersama)	
64	AIF184328-03	Data Science pada Domain Spesifik		AIF183113-03 atau AIF183335-03 atau AIF184351-03

Gambar 2.15: Daftar mata kuliah pilihan beserta prasyaratnya (4)

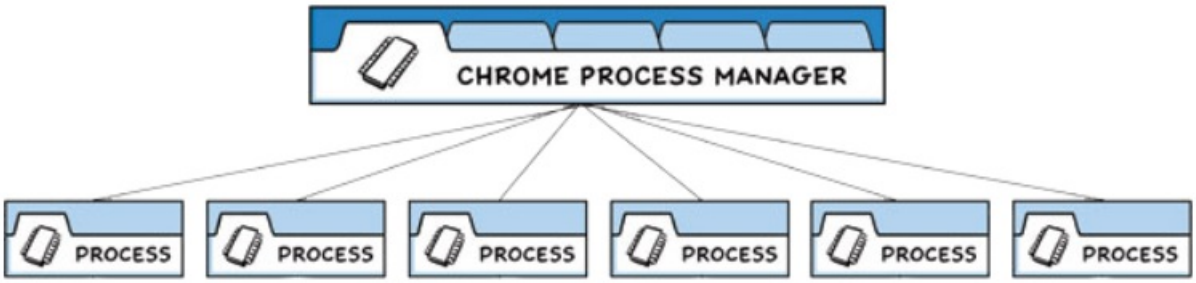
2.2 Electron

Electron adalah sebuah *framework* untuk membangun aplikasi *desktop* menggunakan *JavaScript*, *HTML*, dan *CSS*. Dengan menyematkan *Chromium* dan *Node.js* ke dalam binernya, *Electron* memungkinkan kita untuk mempertahankan satu basis kode *JavaScript* dan membuat aplikasi lintas *platform* yang berfungsi di *Windows*, *macOS*, dan *Linux*, dimana tidak diperlukan pengalaman *native development*. [2]

Electron mewarisi arsitektur *multi-process* dari *Chromium*, yang membuat kerangka kerja secara arsitektur sangat mirip dengan *browser web* moderen. Selanjutnya, akan dijelaskan konsep pengetahuan tentang *Electron*.

Browser web merupakan sebuah aplikasi yang sangat rumit. Selain kemampuan utama mereka untuk menampilkan konten web, mereka memiliki banyak tanggung jawab sekunder, seperti mengelola beberapa jendela atau tab dan memuat ekstensi pihak ketiga. Pada hari - hari sebelumnya, browser biasanya menggunakan satu proses untuk semua fungsi ini. Meskipun pola ini lebih sedikit *overhead* untuk setiap tab yang dibuka, itu juga berarti bahwa satu situs web yang mogok atau macet akan memengaruhi seluruh browser.

Untuk mengatasi masalah tersebut, tim *Chrome* memutuskan bahwa setiap tab akan melakukan *render* untuk setiap prosesnya masing - masing, membatasi bahaya yang dapat ditimbulkan oleh kode berbahaya pada halaman web pada aplikasi secara keseluruhan. Satu proses browser kemudian mengontrol proses ini, serta siklus hidup aplikasi secara keseluruhan. Gambar 2.16 dari *Komik Chrome* memvisualisasikan model ini:



Gambar 2.16: Proses *render Chrome*

Aplikasi *Electron* terstruktur sangat mirip. Sebagai pengembang aplikasi, kita mengontrol dua jenis proses : *main* dan *renderer*. Ini adalah analog dengan browser Chrome dan proses *renderer* yang telah diuraikan di atas. Setiap aplikasi *Electron* memiliki satu proses utama, yang bertindak sebagai titik masuk aplikasi. Proses utama berjalan di lingkungan Node.js, artinya ia memiliki kemampuan untuk membutuhkan modul dan menggunakan semua *Node.js APIs*.

Tujuan dari proses utama adalah untuk membuat dan mengelola jendela aplikasi dengan modul *BrowserWindow*. Setiap contoh dari kelas *BrowserWindow* membuat jendela aplikasi yang memuat halaman web dengan proses *renderer* yang terpisah. Kita dapat berinteraksi dengan konten web ini dari proses utama menggunakan jendela objek webContents. Contohnya dapat dilihat pada Kode 2.1.

Kode 2.1: Contoh kode penggunaan BrowserWindow

```
const { BrowserWindow } = require('electron')

const win = new BrowserWindow({ width:800, height: 1500})
win.loadURL('https://github.com')

const contents = win.webContents
console.log(contents)
```

Karena modul *BrowserWindow* adalah *EventEmitter*, kita juga dapat menambahkan *handlers* untuk berbagai aktivitas pengguna (misalnya, meminimalkan atau memaksimalkan jendela). Saat instance *BrowserWindow* dimusnahkan, proses *renderer* yang terkait akan dihentikan.

Proses utama juga mengontrol siklus hidup aplikasi melalui modul aplikasi *Electron*. Modul ini menyediakan serangkaian besar kejadian dan metode yang dapat digunakan untuk menambahkan perilaku aplikasi khusus (misalnya, menutup aplikasi secara terprogram, atau memodifikasi dokumentasi aplikasi). Contohnya, pada Kode 2.2 menggunakan API aplikasi untuk membuat pengalaman aplikasi *window* menjadi lebih nyata.

Kode 2.2: Contoh kode untuk keluar dari aplikasi

```
app.on('window-all-closed', function(){
    if(process.platform !== 'darwin') app.quit()
})
```

Setiap aplikasi *Electron* memunculkan proses penyaji terpisah untuk setiap *BrowserWindow* yang terbuka. Seperti namanya, perender bertanggung jawab untuk merender konten web. Untuk semua maksud dan tujuan, kode yang dijalankan dalam proses perender harus berperilaku sesuai dengan standar web (setidaknya sejauh yang dilakukan Chromium). Oleh karena itu, semua pengguna *interface* dan fungsionalitas aplikasi dalam satu jendela browser harus ditulis dengan alat dan paradigma yang sama dengan yang digunakan di web. Terdapat beberapa spesifikasi web yang harus dipahami, seperti :

- File HTML untuk proses rendering.
- Menambahkan styling UI melalui Cascading Style Sheets (CSS).
- Kode JavaScript yang dapat dieksekusi dengan menambahkannya pada elemen <script>.

Maka dari itu, perender tidak memiliki akses langsung ke *require* atau Node.js APIs yang lain. Untuk menyertakan modul NPM secara langsung di perender, kita harus menggunakan *bundler toolchains* yang sama (misalnya, webpack atau parcel) yang digunakan di web.

Untuk cara membuat aplikasi dengan *electron* adalah sebagai berikut :

1. Install terlebih dahulu Node.js dan npm (pakai versi terakhir LTS yang tersedia).
2. Cek versinya dengan mengetik node -v dan npm -v pada command prompt.
3. Buat folder dan inisialisasi paket npm dengan mengetik mkdir namaFolder lalu ketik cd namaFolder setelah itu ketik npm init pada *command prompt*. Maka isi *package.json* akan seperti pada Kode 2.3.

Kode 2.3: Package.json

```
{
    "name": "my-electron-app",
    "version": "1.0.0",
    "description": "Hello World!",
    "main": "main.js",
    "author": "Jane Doe",
    "license": "MIT",
}
```

4. Install aplikasi *electron* ke dalam folder yang telah dibuat dengan cara mengetik npm install –save-dev electron pada command prompt.
5. Pada *script file* package.json tambahkan *command start* seperti pada Kode 2.4.

Kode 2.4: *Start command*

```
{
    "scripts": {
        "start": "electron ."
    }
}
```

6. Jalankan aplikasi *electron* dengan mengetik npm start pada command prompt.

2.3 Vis.js

Vis.js adalah sebuah visualisasi *library* berbasis *browser* yang dinamis. *Library* dirancang agar mudah digunakan, untuk menangani sejumlah besar data dinamis, dan memungkinkan untuk manipulasi dan berinteraksi dengan data. *Library* tersebut terdiri dari komponen *DataSet*, *Timeline*, *Network*, *Graph2d* dan *Graph3d*. [3]

2.3.1 Timeline

Timeline adalah grafik visualisasi interaktif untuk memvisualisasikan data dalam bentuk waktu. Item data dapat berlangsung pada satu tanggal, atau memiliki tanggal mulai dan berakhir. Kita dapat dengan bebas memindahkan dan memperbesar *timeline* dengan menyeret dan menggulir di *timeline*. Item dapat dibuat, diedit, dan dihapus di *timeline*. Skala waktu pada sumbu disesuaikan secara otomatis, dan mendukung skala mulai dari milidetik hingga tahun. *Timeline* menggunakan HTML DOM biasa untuk merender *timeline* dan item yang diletakkan di *timeline*. Hal ini memungkinkan penyesuaian yang fleksibel menggunakan *css style*. Contoh kode berikut pembuatan *timeline* dapat dilihat pada Kode 2.5 dan untuk hasilnya dapat dilihat pada gambar 2.17.

Kode 2.5: Contoh kode untuk membuat *timeline* menggunakan *vis.js*

```
<!DOCTYPE HTML>
<html>
<head>
  <title>Timeline | Basic demo</title>

  <style type="text/css">
    body, html {
      font-family: sans-serif;
    }
  </style>

  <script src="../../dist/vis.js"></script>
  <link href="../../dist/vis-timeline-graph2d.min.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="visualization"></div>

<script type="text/javascript">
  // DOM element where the Timeline will be attached
  var container = document.getElementById('visualization');

  // Create a DataSet (allows two way data-binding)
  var items = new vis.DataSet([
    {id: 1, content: 'item 1', start: '2013-04-20'},
    {id: 2, content: 'item 2', start: '2013-04-14'},
    {id: 3, content: 'item 3', start: '2013-04-18'},
    {id: 4, content: 'item 4', start: '2013-04-16', end: '2013-04-19'},
    {id: 5, content: 'item 5', start: '2013-04-25'},
    {id: 6, content: 'item 6', start: '2013-04-27'}
  ]);

  // Configuration for the Timeline
  var options = {};

  // Create a Timeline
  var timeline = new vis.Timeline(container, items, options);
</script>
</body>
</html>
```

Gambar 2.17: Hasil contoh untuk membuat *timeline* menggunakan *vis.js*

2.3.2 Network

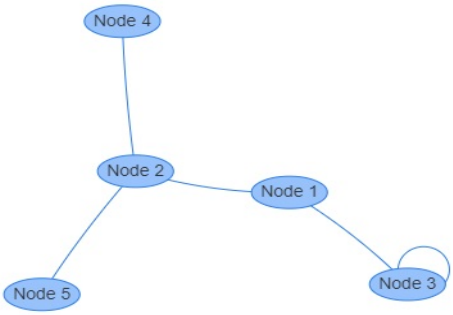
Jaringan (*Network*) adalah visualisasi untuk menampilkan jaringan - jaringan yang terdiri dari *node* dan *edge*. Visualisasinya mudah digunakan dan mendukung bentuk kustom, gaya, warna, ukuran, gambar, dan lain - lain. Visualisasi jaringan bekerja dengan lancar di *browser* moderen apapun hingga beberapa ribu *node* dan *edge*. Untuk menangani jumlah node yang lebih besar, jaringan memiliki dukungan pengelompokan. Jaringan menggunakan *HTML canvas* untuk *rendering*.

Untuk menggunakan *vis-network*, kita harus menyertakan file *vis-network.js* dan *vis-network.css* yang dapat diunduh dari visjs.org, atau dengan tautkan dari unpkg.com atau bisa juga dengan *import* dari paket npm. Jika kita menambahkan ini ke aplikasi kita, kita perlu menentukan *node* dan *edgenya*. Kita juga dapat menggunakan *vis.DataSets* untuk pengikatan data dinamis, misalnya, mengubah warna, label, atau pilihan apapun setelah kita menginisialisasi jaringan.

Setelah kita memiliki data, yang kita butuhkan hanyalah *container div* untuk memberi tahu *vis* di mana harus meletakkan jaringan kita. Selain itu, kita dapat menggunakan pilihan objek untuk menyesuaikan banyak aspek jaringan. Contoh kode pembuatan *network* dimana file *vis-network.js* dimasukkan dengan cara menautkan dari unpkg.com dapat dilihat pada Kode 2.6 dan untuk hasilnya dapat dilihat pada gambar 2.18.

Kode 2.6: Contoh kode untuk membuat *network* menggunakan *vis.js*

```
<html>
<head>
    <script type="text/javascript" src="https://unpkg.com/vis-network/standalone/umd/vis-network.min.js"></script>

    <style type="text/css">
        #mynetwork {
            width: 600px;
            height: 400px;
            border: 1px solid lightgray;
        }
    </style>
</head>
<body>
<div id="mynetwork"></div>

<script type="text/javascript">
    // create an array with nodes
    var nodes = new vis.DataSet([
        {id: 1, label: 'Node 1'},
        {id: 2, label: 'Node 2'},
        {id: 3, label: 'Node 3'},
        {id: 4, label: 'Node 4'},
        {id: 5, label: 'Node 5'}
    ]);

    // create an array with edges
    var edges = new vis.DataSet([
        {from: 1, to: 3},
        {from: 1, to: 2},
        {from: 2, to: 4},
        {from: 2, to: 5}
    ]);

    // create a network
    var container = document.getElementById('mynetwork');

    // provide the data in the vis format
    var data = {
        nodes: nodes,
        edges: edges
    };
    var options = {};

    // initialize your network!
    var network = new vis.Network(container, data, options);
</script>
</body>
</html>
```



Gambar 2.18: Hasil contoh untuk membuat *network* menggunakan *vis.js*

2.4 FTIS Open Data

FTIS (Fakultas Teknologi Informasi dan Sains) adalah sebuah fakultas milik UNPAR (Universitas Katolik Parahyangan) yang memiliki tiga program studi, yaitu : Matematika, Fisika, dan Teknik Informatika. *Open data* adalah data yang dapat bebas digunakan oleh semua orang untuk digunakan dan diterbitkan kembali sesuai keinginan, tanpa adanya batasan hak cipta, paten, atau mekanisme kontrol lainnya. Maka dari itu FTIS *Open Data* dapat diartikan sebagai data milik FTIS khususnya program studi teknik informatika yang secara bebas dapat digunakan secara bebas oleh orang lain. FTIS *open data* tersebut dapat diakses pada `https://ftisunpar.github.io/data/prasyarat.json`. [4]

Endpoint didefinisikan sebagai *url* yang mengikuti setelah basis *url* `https://ftisunpar.github.io/data/`. Sebagai contoh *endpoint* prasyarat.json memiliki alamat *url* lengkap `https://ftisunpar.github.io/data/prasyarat.json`. Prasyarat.json memiliki informasi tentang seluruh matakuliah, jumlah SKS, posisi semester, serta prasyarat yang ada. Dengan bentuk datanya adalah array dari matakuliah. Setiap struktur matakuliah memiliki properti sebagai berikut :

- kode (String)
- nama (String)
- prasyarat
  - tempuh (String[])
  - lulus (String[])
  - bersamaan (String[])
  - berlakuAngkatan (Number|null)
- sks (Number)
- wajib (Boolean)
- semester (Number)

Untuk lebih jelasnya dapat dilihat pada Kode 2.7.

Kode 2.7: prasyarat.json

```
{
    "kode": "AIF181100",
    "nama": "Dasar Pemrograman",
    "prasyarat": {
      "tempuh": [],
      "lulus": [
        "AIF181101"
      ],
      "bersamaan": [],
      "berlakuAngkatan" : 2018
    },
    "sks": 4,
    "wajib": true,
    "semester": 2
}
```

Pada prasyarat lulus, diberikan sebuah string berupa kode mata kuliah yang menjadi prasyarat lulus untuk mata kuliah tersebut. Definisi prasyarat berdasarkan macamnya :

- Tempuh
  Mahasiswa diharuskan sudah pernah menempuh mata kuliah yang disebutkan.
- Lulus
  Mahasiswa diharuskan untuk lulus mata kuliah yang disebutkan.
- Bersamaan
  Mata kuliah tersebut harus di ambil secara bersamaan dengan mata kuliah yang disebutkan (butuh informasi lagi).
- Berlaku Angkatan
  Property ini mulai berlaku karena pergantian kurikulum. Prasyarat ini memiliki maksud bahwa matakuliah ini mulai berlaku semenjak angkatan x.

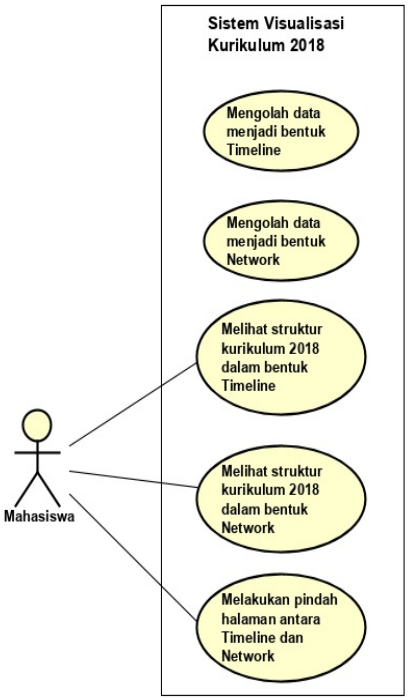
BAB 3

ANALISIS

3.1 Analisis Kebutuhan Sistem

3.2 Analisis Rancangan Aplikasi

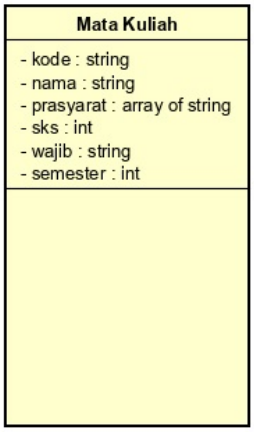
3.2.1 Use Case Diagram



Gambar 3.1: Use Case Diagram

- Sistem akan mengambil data dari https://raw.githubusercontent.com/ftisunpar/data/master/prasyarat.json.
- Sistem mengolah data menjadi bentuk *Timeline*.
- Sistem mengolah data menjadi bentuk *Network*.
- Mahasiswa melihat struktur kurikulum 2018 dalam bentuk *Timeline*.
- Mahasiswa melihat struktur kurikulum 2018 dalam bentuk *Network*.
- Mahasiswa dapat melakukan pindah halaman antara halaman *Timeline* dengan halaman *Network*.

3.2.2 Class Diagram



Gambar 3.2: Class Diagram

3.3 Analisis Hasil Survei

DAFTAR REFERENSI

[1] Mariskha Tri Adithia, H. H. V. S. M. C. W., Cecilia E. Nugraheni (2018) *kurikulum-2018*. Universitas Katolik Parahyangan, Bandung.

[2] Electron. https://www.electronjs.org/. Accessed: 2021-04-27.

[3] Vis.js. https://visjs.org/. Accessed: 2021-04-27.

[4] Ftis open data. https://github.com/ftisunpar/data. Accessed: 2021-04-27.

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
```

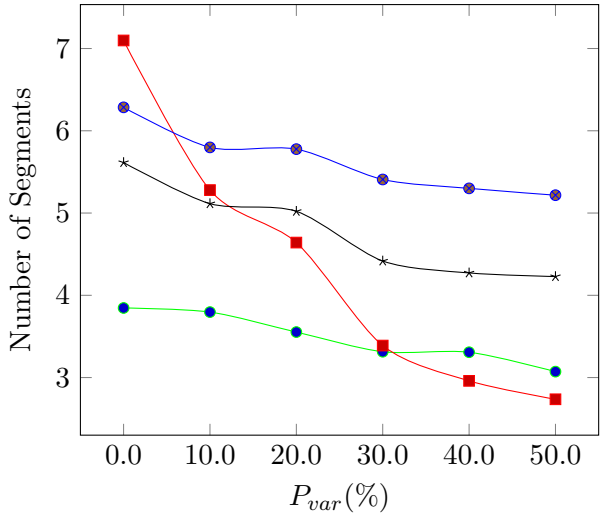
Kode A.2: MyCode.java

```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);
        }
    }

}
```

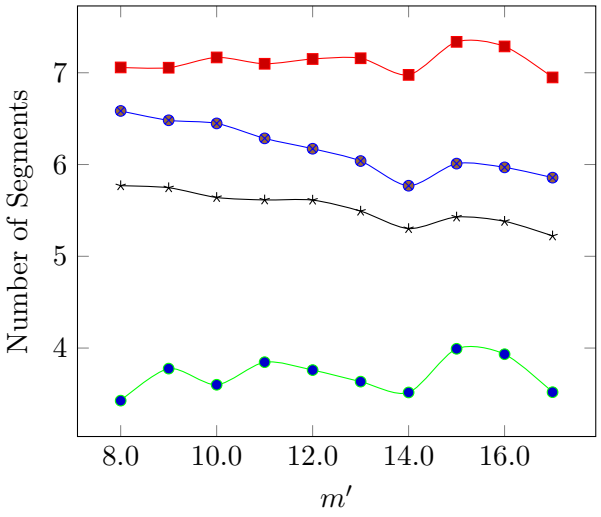
LAMPIRAN B

HASIL EKSPERIMEN

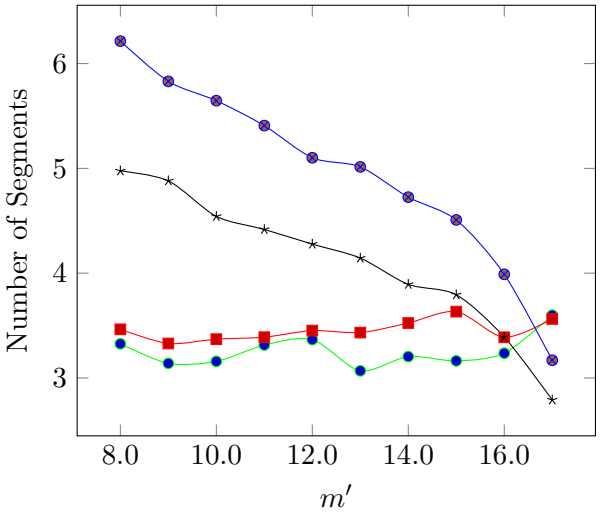
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



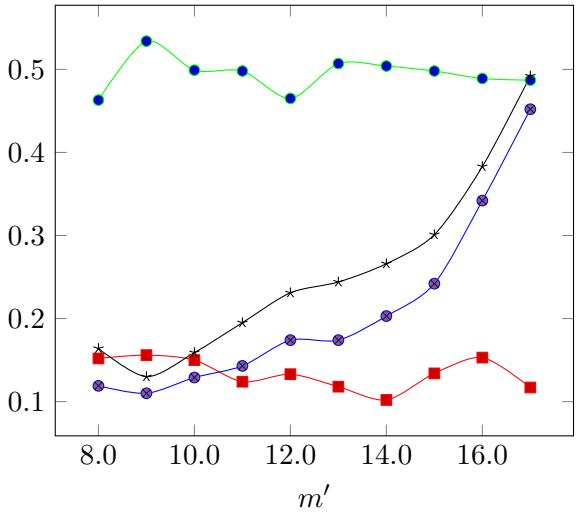
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4