SKRIPSI

PEMODELAN KULIAH KURIKULUM 2018 DALAM FORMAT JSON

Muhammad Taufik Adianto

NPM: 2012730089

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

«tahun»

UNDERGRADUATE THESIS

«JUDUL BAHASA INGGRIS»

Muhammad Taufik Adianto

NPM: 2012730089

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

«tahun»

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»

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

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»

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

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

1.1 Latar Belakang

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][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 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. Tahapan penyusunan kurikulum 2018 meliputi kegiatan sebagai berikut:

1. Melakukan evaluasi diri dan pelacakan lulusan.
2. Merumuskan profil 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.

Teknologi baru sekarang memungkinkan untuk membangun layanan yang menjawab pertanyaan-pertanyaan secara otomatis. Sebagian besar data yang diperlukan untuk menjawab pertanyaan-pertanyaan dihasilkan oleh badan-badan publik. Namun, seringkali data yang diperlukan belum tersedia dalam bentuk yang mudah digunakan. Data terbuka berbicara tentang bagaimana membuka potensi dari informasi resmi dan lainnya untuk mengaktifkan layanan-layanan baru. Gagasan dari data terbuka itu sendiri bertujuan agar setiap orang bebas untuk mengakses dan menggunakan

[1]Panduan Penyusunan Kurikulum Pendidikan Tinggi, Kemenristekdikti, 2016

ulang untuk berbagai tujuan - sudah bergulir dalam beberapa tahun ini. Data terbuka itu sendiri memiliki arti yaitu data yang dapat secara bebas digunakan, digunakan ulang dan didistribusi ulang oleh siapapun - hanya patuh, umumnya, pada keharusan untuk menyebutkan siapa penciptanya dan berbagi dengan lisensi yang sama.[2] Defini Terbuka memberikan rincian yang tepat apa yang dimaksud data terbuka. Ringkasannya adalah:

1. **Ketersediaan dan Akses:** data harus tersedia secara keseluruhan dan tidak lebih dari pada biaya reproduksi yang masuk akal, akan lebih baik bila bisa dilakukan dengan pengunduhan melalui internet.

2. **Penggunaan-ulang dan Distribusi ulang:** data harus disediakan di bawah ketentuan yang mengizinkan untuk penggunaan-upang dan pendistribusian ulang termasuk memadukan dengan kumpulan data lainnya.

3. **Partisipasi Universal:** setiap orang harus diperbolehkan untuk menggunakan, menggunakan-ulang dan mendistribusi ulang - tidak boleh ada diskriminasi terhadap bidang kerja atau perseorangan atau kelompok.

Untuk menampung data terbuka dapat digunakan *github* sebagai salah satu penampung untuk menyimpan data. *Github* sebagai *open source* di dalamnya dapat menyimpan data dalam *format JSON*. *JSON* digunakan sebagai acuan dalam pembuatan pohon kurikulum 2018. *Format JSON* bakal diubah ke dalam *DOT Language* untuk menghasilkan graf. Penggunaan graf ditujukan agar mempermudah dalam melihat kurikulum baru. Untuk mem *visualisasi* kan graf digunakan *viz.js*, *Viz.js* ini nantinya akan membantu dalam menghasilkan graf yang akan di tampilkan.

1.2 Rumusan Masalah

Berdasarkan latar belakang masalah yang telah dijelaskan, rumusan masalah pada penelitian ini adalah:

1. Bagaimana menerjemahkan perangkat lunak dalam bentuk *word* ke bentuk *JSON*.

2. Bagaimana membuat perangkat lunak dari bentuk *JSON* ke dalam graf.

1.3 Tujuan

Berdasarkan rumusan masalah di atas, maka tujuan dari penelitian ini adalah:

1. Membuat terjemahan dari bentuk *word* ke dalam bentuk *JSON*.

2. Membuat perangkat lunak dalam bentuk graf.

1.4 Batasan Masalah

Adapun batasan masalah yang didapat dari tujuan dan rumusan masalah di atas adalah:

1. Perangkat lunak menghasilkan pohon kurikulum.

[2] "Data Terbuka", http://opendatahandbook.org/guide/id/what-is-open-data/

1.5 Metodologi Penelitian

Dalam penyusunan skripsi ini mengikuti langkah-langkah metodologi penelitian sebagai berikut:

1. Melakukan studi pustaka untuk dijadikan referensi dalam pembangunan perangkat lunak pohon kurikulum.
2. Melakukan studi tentang penggunaan vis.js untuk visualisasi pohon kurikulum.
3. Melakukan studi tentang data terbuka.
4. Melakukan studi tentang cara penggunaan DOT *Language*

1.6 Sistematika Penulisan

Keseluruhan bab yang disusun dalam penelitian ini terbagi kedalam bab-bab sebagai berikut:

1. Bab 1 Pendahuluan Bab ini membahas mengenai latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian dan sistematika penulisan.
2. Bab 2 Dasar Teori Bab ini membahas mengenai pengertian graf, data terbuka, JSON, apa itu DOT *Language*, dan visualisasi menggunakan *viz.js.*
3. Bab 3 Analisis Bab ini akan membahas mengenai JSON yang dapat dipakai sebagai sumber data terbuka.
4. Bab 4 Perancangan Bab ini akan membahas mengenai perancangan struktur pohon kurikulum untuk mahasiswa, di mana nanti di dalamnya akan berisi mata kuliah, syarat tempuh, dan syarat lulus.
5. Bab 5 Implementasi dan Pengujian Bab ini akan membahas mengenai pengujian, implementasi kode program untuk membuat pohon kurikulum.
6. Bab 6 Kesimpulan dan Saran Bab ini akan membahas mengenai kesimpulan dari penelitian yang telah dilakukan dan saran-saran untuk pengembangan lebih lanjut dari penelitian ini.

BAB 2

DASAR TEORI

Pada bab ini akan diuraikan teori-teori yang berhubungan dengan pembangunan pohon kurikulum. Teori-teori tersebut adalah teori tentang pengertian graf,data terbuka, *JSON*, *DOT language*, dan visualisasi pohon menggunakan *viz.js*.

2.1 Graf

2.1.1 Definisi Graf

Suatu graf didefinisikan oleh himpunan verteks dan himpunan sisi *(edge)*. [2] [1] Verteks menyatakan entitas-entitas data dan sisi menyatakan keterhubungan antara verteks. Biasanya untuk suatu graf G digunakan notasi matematis.

G=(V,E)
G = Graf
V = Simpul atau verteks
E = Sisi atau garis

V adalah himpunan verteks dan E himpunan sisi yang terdefinisi antara pasangan-pasangan verteks. Sebuah sisi antara verteks x dan y ditulis x, y. Suatu graf H = (V1,E1) disebut subgraf dari graf G jika V1 adalah himpunan bagian dari V dan E1 himpunan bagian dari E.

2.1.2 Istilah dalam Graf

1. *Incident*
   Jika e merupakan busur dengan simpul-simpulnya adalah v dan w yang ditulis e=(v,w), maka v dan w disebut "terletak" pada e, dan e disebut incident dengan v dan w.

2. *Degree*
   Di dalam Graf ada yang disebut dengan *Degree*, *Degree* mempuyai 3 jenis antara lain :

   - Degree dari suatu verteks x dalam *undigraph* adalah jumlah busur yang incident dengan simpul tersebut.
   - Indegree dari suatu verteks x dalam digraf adalah jumlah busur yang kepalanya incident dengan simpul tersebut, atau jumlah busur yang "masuk" atau menuju simpul tersebut.

[1] "Graf", Matematika Diskrit, Juli 2015

- Outdegree dari suatu verteks x dalam digraf adalah jumlah busur yang ekornya incident dengan simpul tersebut, atau jumlah busur yang "keluar" atau berasal dari simpul tersebut.

3. *Adjacent*
   Pada graf tidah berarah, 2 buah simpul disebut adjacent bila ada busur yang menghubungkan kedua simpul tersebut. Simpul v dan w disebut adjacent. Pada graf berarah, simpul v disebut adjacent dengan simpul w bila ada busur dari w ke v.

4. *Successor dan Predecessor*
   Pada graf berarah, bila simpul v *adjacent* dengan simpul w, maka simpul v adalah *successor* simpul w, dan simpul w adalah *predecessor* dari simpul v.

2.2 Data Terbuka

Teknologi sekarang memungkinkan untuk membangun layanan yang menjawab pertanyaan-pertanyaan secara otomatis. [3] [2] Sebagian besar data yang diperlukan untuk menjawab pertanyaan-pertanyaan dihasilkan oleh badan-badan publik. Namun, seringkali data yang diperlukan belum tersedia dalam bentuk yang mudah digunakan. Gagasan dari data terbuka mengarah kepada informasi di mana setiap orang bebas untuk mengakses dan menggunakan ulang untuk berbagai tujuan - sudah bergulir dalam beberapa tahun ini.

2.2.1 Apa itu Data Terbuka

Data terbuka adalah data yang dapat digunakan secara bebas, dimanfaatkan, dan didistribusikan kembali oleh siapapun tanpa syarat, kecuali dengan mengutip sumber dan pemilik data. Selain itu, seluruh data yang dipublikasikan harus mengikuti peraturan perundang-undangan yang berlaku. Kriteria penting dari data terbuka adalah:

1. Ketersediaan dan Akses Data harus tersedia utuh dan bebas biaya. Akan lebih baik jika data dapat diunduh melalui internet. Data juga harus tersedia dalam bentuk yang mudah digunakan dan dapat diolah kembali.

2. Penggunaan dan Pendistribusian Data yang digunakan dan didistribusikan kembali harus memenuhi syarat-syarat yang telah ditentukan.

3. Partisipasi Universal Setiap orang bebas menggunakan dan mendistribusikan kembali *dataset*. Tidak diperkenankan adanya diskriminasi atas bidang usaha, orang, atau kelompok.

Semua kriteria yang ada di dalam data terbuka sangat penting karena menunjukkan kejelasan tentang apa yang dimaksud dengan terbuka itu sendiri. Istilah yang digunakan untuk menjelaskan ketiga kriteria data terbuka adalah *interoperabilitas*. Interoperabilitas sangat penting karena memungkinkan komponen-komponen yang berbeda untuk bisa bekerja sama. Kemampuan untuk mengkomponenisasi komponen-komponen sangatlah esensial untuk membangun sistem yang besar

[2] http://opendatahandbook.org/guide/id/introduction/

dan kompleks. Tanpa *interoperabilitas* hal ini menjadi tidak mungkin di mana kemampuan untuk berkomunikasi (lintas operasi) sangat berpengaruh terhadap keberhasilan suatu rencana.

Inti dari sebuah "keumuman" data merupakan salah satu bagian dari materi "terbuka". *Interoperabilitas* ini merupakan komponen penting untuk merealisasikan praktik utama manfaat dari "keterbukaan": Peningkatan dramatis kemampuan untuk mengkombinasikan sekumpulan data berbeda secara bersama-sama sehingga merangsang pengembangan produk dan layanan yang lebih baik. keterbukaan dapat memastikan bahwa ketika ada dua kumpulan data dari dua sumber berbeda, maka kita dapat menggabungkan data tersebut secara bersama-sama, dan memastikan bahwa data yang kita dapat informasinya benar.

2.2.2 Mengapa Data Terbuka

Data terbuka adalah sumber daya luar biasa yang belum dimanfaatkan sepenuhnya. Banyak individu dan organisasi mengumpulkan berbagai jenis data berbeda dalam rangka untuk melakukan tugas mereka. Pemerintah sangat signifikan dalam hal ini, tidak hanya karena kuantitas dan sentralitas dari data yang dikumpulkan, tetapi juga karena sebagian besar dari data pemerintah adalah bersifat publik secara hukum, dan oleh karena itu bisa dibuat terbuka dan tersedia untuk orang lain untuk dipergunakan. hal itu menjadi menarik karena banyak individu atau kelompok yang ingin mengetahui data yang ada.

Ada banyak area di mana kita bisa mengharapkan data terbuka untuk menjadi sebuah nilai, dan menjadi contoh bagaimana data terbuka telah digunakan. Ada juga kelompok dengan banyak orang berbeda dan organisasi yang dapat meraih keuntungan dari ketersediaan data yang terbuka, termasuk pemerintah itu sendiri. Pada saat yang sama adalah mustahil untuk memprediksi secara tepat bagaimana dan di mana nilai akan dibuat di masa depan. Sifat alami dari inovasi adalah bahwa pengembangan seringkali datang dari tempat yang tidak mungkin. Hal ini sudah dimungkinkan dengan merujuk pada sejumlah data terbuka yang telah menciptakan nilai. Beberapa nilai ini meliputi:

- Transparansi dan kendali
- Partisipasi
- Penguatan mandiri
- Inovasi
- Efisiensi dan Efektivitas lebih baik dari layanan yang sudah ada
- Pengukuran pengaruh dari kebijakan-kebijakan
- Pengetahuan baru dari kombinasi sumber data dan pola-pola dalam volume data yang besar

2.2.3 Cara Membuka Data

Data Terbuka dapat dibuka para pemegang data. Para pemegang data dapat melakukannya secara mendasar, tetapi juga mencakup masalah-masalah yang tersembunyi dan menjebak. Terdapat tiga aturan kunci yang kami rekomendasikan saat membuka data:

1. Jadikan lebih praktis.
2. Terlibat dari awal dan melibatkan diri sesering mungkin.
3. Mengatasi kekhawatiran umum dan kesalahpahaman.

Ada empat langkah utama dalam membuat data terbuka, yang masing-masing akan dibahas secara rinci di bawah ini. Langkah-langkah tersebut adalah yang paling memungkinkan - banyak dari langkah-langkah tersebut dapat dilakukan secara bersamaan.

1. **Pilih Kumpulan Data**
   Pemilihan kumpulan-kumpulan data yang direncanakan untuk menjadikannya terbuka merupakan langkah pertama meskipun perlu diingat bahwa seluruh proses pembukaan data akan berulang dan dapat kembali ke langkah ini bila mengalami masalah di kemudian hari. Jika sudah mengetahui persis kumpulan-kumpulan data maka dapat merencanakan untuk dibuka dan dapat langsung ke bagian berikutnya. Bagaimanapun juga, dalam banyak kasus, terutama untuk lembaga-lembaga yang besar, untuk berfokus pada memilih kumpulan data menjadi sebuah tantangan.

2. **Menerapkan sebuah Lisensi Terbuka (Keterbukaan Resmi)**
   Di kebanyakan yurisdiksi terdapat hak kekayaan intelektual di dalam data yang mencegah pihak ketiga dari penggunaannya, penggunaan ulang dan pendistribusian data tanpa izin eksplisit. Bahkan di tempat di mana keberadaan hak hukum serba tidak pasti, penting untuk menerapkan lisensi demi sebuah kejelasan. Dengan demikian, jika kita berencana untuk membuat data tersedia, maka harus menaruh lisensi di atasnya dan jika data menjadi terbuka ini bahkan lebih penting lagi.

3. **Menjadikan Data Terseddia**
   Data terbuka membutuhkan keterbukaan secara teknis sebagaimana keterbukaan yang resmi secara hukum. Khususnya, data harus bisa tersedia secara masal dalam format (yang dapat dibaca mesin).

   - **Available**
     Data seharusnya dihargai tidak lebih dari biaya reproduksi yang wajar, sebaiknya dijadikan sebagai unduhan gratis dari internet. Model penghargaan ini dapat dicapai karena lembaga anda tidak perlu menangani biaya apapun saat menyediakan data untuk digunakan.
   - **In Bulk**
     Data harus tersedia dalam kumpulan yang lengkap. Jika anda memiliki daftar yang dikoleksi di bawah aturan undang-undang, seluruh daftar tersebut harus tersedia untuk diunduh.
   - **In an open, machine-readable format**
     Penggunaan-ulang data yang disediakan oleh sektor publik tidak seharusnya tunduk pada pembatasan paten.

- **Jadikan hingga Mudah untuk ditemukan**
  terbitkan di web dan mungkin kelola sebuah pusat katalog untuk membuat daftar dari kumpulan data terbuka.

2.3 *Creative Commons*

Creative commons bisa diartikan sebagai suatu organisasi tidak menguntungkan yang memiliki tujuan untuk memperluas cakupan karya kreatif sehingga karya tersebut legal untuk digunakan orang lain secara gratis tanpa mengurangi esensi hak cipta bagi sang pencipta karya tersebut. Dalam hal ini lisensi *creative commons* akan menyediakan standar bagi pemegang hak cipta untuk memberikan izin pada orang lain yang ingin menggunakan hasil karyanya. Umunya file-file berlisensi CC berada dalam domain publik seperti di *youtube*, forum dan *website* yang sudah memiliki izin pendistribusian dari pemilikinya.

Creative commons itu sendiri terbagi dalam empat macam jenis lisensi karya hak cipta orang lain. Berikut adalah tiga macam jenis lisensi *Creative Commons*

1. **CC-NC (Creative Commons non-commercial)**
   Creative Commons non-commercial adalah Mengizinkan orang lain menyalin, mendistribusikan, menampilkan, serta membuat karya turunan berdasarkan suatu karya hanya untuk tujuan nonkomersial. Artinya jika anda mengunduh sebuah file yang lisensinya CC-NC lalu ingin mereupload ke *platform* yang diinginkan maka tidak bisa menghasilkan uang dari video tersebut.

2. **CC-SA (Creative Commons share-alike)**
   Creative Commons share-alike adalah Mengizinkan orang lain untuk mendistribusikan suatu karya turunan hanya di bawah suatu lisensi yang identik dengan lisensi yang diberikan pada karya aslinya. Sebagai contoh anda ingin mengabungkan beberapa file, maka di dalam file tersebut anda hanya boleh menggabung file yang memberi izin tertulis kepada anda. Misalnya yang memberi lisensi media x maka seluruh file bersumber dari media x.

3. **CC-BY (Creative Commons Attribution)**
   Creative Commons attribution adalah suatu lisensi yang mengizinkan orang lain untuk menyalin, mendistribusikan, menampilkan, serta membuat karya turunan berdasarkan suatu karya hanya jika orang tersebut memberikan penghargaan pada pencipta atau pemberi lisensi dengan cara yang disebutkan dalam lisensi. Sebagai contoh anda menemukan file yang berada dalam domain publik tapi dalam keterangannya file tersebut lisensinya adalah CC-BY, anda diharuskan memberi kredit kepada sang pemilik file, biasanya kredit dalam bentuk *Title*, Kredit *scroll* baik di akhir file maupun di awal file. Jenis CC-BY yang satu ini bebas dipakai.

2.4 *JSON*

JSON *(JavaScript Object Notation)* adalah format pertukaran data. *JSON* merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun.[3] Kenapa *JSON*? Karena ukuran

[3] "JSON", https://www.json.org/json-id.html

datanya lebih kecil dibanding dengan *XML*, sifatnya *"self-describing"* dan mudah dimengerti. Formatnya berbasis teks dan terbaca manusia serta digunakan untuk mempresentasikan struktur data sederhana. Format teks dari JSON itu sendiri identik dengan kode untuk membuat objek *JavaScript* memiliki kesamaan dengan *Javascript*, hanya saja *JSON* lebih mudah dimengerti.

2.4.1 Struktur JSON

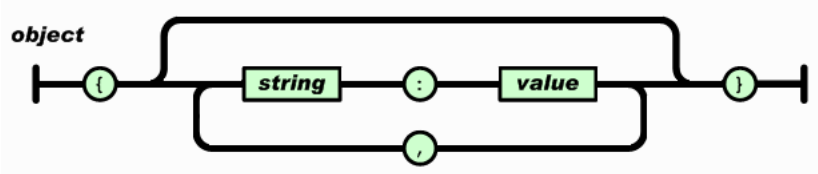
JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek *(object)*, rekaman *(record)*, struktur *(struct)*, kamus *(dictionary)*, tabel hash*(hash table)*, daftar berkunci *(keyed list)*, atau *associative array.*

2. Daftar nilai terurutkan *(an ordered list of values).* Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik *(array)*, vektor *(vector)*, daftar *(list)*, atau urutan *(sequence).*

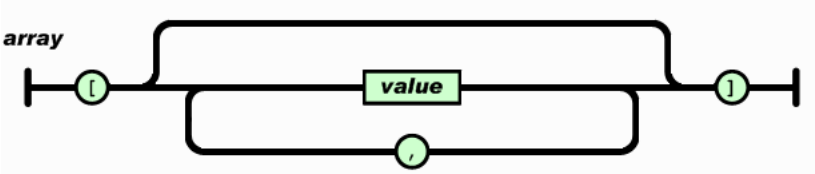
JSON menggunakan bentuk sebagai berikut:

1. **Objek** adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan (kurung kurawal buka) dan diakhiri dengan (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma).



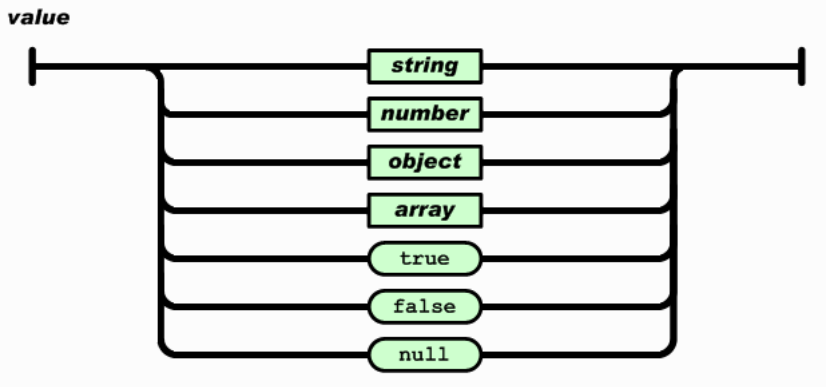
Gambar 2.1: Objek

2. **Larik** adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma).



Gambar 2.2: Larik

3. **Nilai**, dapat berupa sebuah *string* dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.



Gambar 2.3: Nilai

2.4.2 Contoh Sintaks

Contoh berikut menunjukkan representasi JSON untuk suatu objek yang mendeskripsikan seseorang.

```
{
"namaDepan": "Budi",
"namaBelakang": "Subudi",
"alamat": {
        "namaJalan": "Jl. Sudirman 15A",
        "kota": "Jakarta Selatan",
        "provinsi": "DKI Jakarta",
        "kodePos": 11111 },
        "nomerTelepon": [
        "021 555-1234",
        "021 555-4567"
        ]
}
```

2.5 DOT Language

DOT adalah bahasa yang dapat digunakan untuk menampilkan grafik secara teks, sehingga dapat diproses melalui titik untuk membuat grafik sebagai representasi grafis dalam format yang berbeda seperti .ps, .pdf, dll. [4] [4] DOT telah dikembangkan sebagai bagian dari proyek *Graphviz*, yang merupakan kumpulan alat untuk visualisasi grafik.

2.5.1 Dasar Menggambar Graf

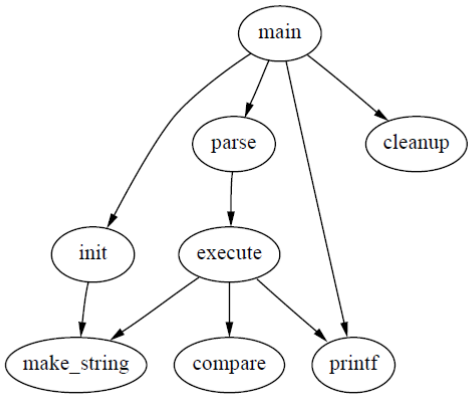
Dot mengambil empat langkah utama dalam menggambar grafik. Langkah pertama menetapkan diskrit peringkat ke node dalam gambar atas ke bawah, menentukan peringkat di koordinat Y. Tepi yang membentang lebih banyak dari satu peringkat dipecah menjadi rantai simpul dan tepi unit. Langkah kedua *node* dalam barisan untuk menghindari penyeberangan. Langkah ketiga menetapkan

[4] "Drawing Graphs with DOT", E Koutsofios, S North - 1991

koordinat *node* X untuk disimpan dibaris terpendek. Langkah terakhir rute tepi splines. Grafik menggunakan dot memiliki tiga jenis *item*: grafik, simpul, dan tepi. Grafik sendiri memiliki dua bentuk yaitu grafik (tidak diarahkan) atau digraf (diarahkan). Karena dot membuat *layout* grafik yang diarahkan maka contoh dalam kasus ini menggunakan digraf.

Gambar graf1 adalah contoh grafik dalam bahasa dot. Baris 1 memberi nama dan jenis grafik. Baris berikut membuat node, tepi, atau subgraf, dan atur atribut. Nama merupakan *identifier* C, nomor, atau kutipan C. Sebuah simpul diciptakan pertama kali namanya muncul di *file*. Tepian dibuat saat node berada bergabung dengan operator tepi ->. Pada contoh, baris 2 membuat tepi lalu mengurai dari *parse* ke *execute*. Untuk menjalankan dot pada file ini (dimisalkan graf1.dot) dapat mengetikan $dot - Tpsgraf1.dot - ograf1.ps$ dan akan menghasilkan gambar graf1.

```
1: digraph G {
2: main -> parse -> execute;
3: main -> init;
4: main -> cleanup;
5: execute -> make_string;
6: execute -> printf
7: init -> make_string;
8: main -> printf;
9: execute -> compare;
10: }
```



Gambar 2.4: Gambar Graf 1

2.5.2 Subgraf dan Pengelompokan

Subgraf memiliki tiga peran di *Graphviz*. Pertama, subgraf dapat digunakan untuk mewakili struktur grafik, yang menunjukkan bahwa simpul dan tepi tertentu harus dikelompokkan bersama. Informasi pada subgraf ditentukan secara semantik tentang komponen grafik. Tepi dibuat dari setiap simpul di sebelah kiri ke setiap simpul di sebelah kanan. Contohnya sebagai berikut

```
A -> {B C}
```

```
sama dengan
A -> B
A -> C
```

Kedua, subgraf dapat memberikan konteks untuk mengatur atribut. Sebagai contoh, sebuah subgraf dapat menentukan bahwa warna biru adalah warna *default* untuk semua node yang didefinisikan di dalamnya. Dalam konteks gambar grafik, contohnya sebagai berikut

```
subgraf {
peringkat = sama; A; B; C;
}
```

Subgraf ini menentukan bahwa simpul A, B dan C semuanya harus ditempatkan pada rangking yang sama jika ditarik menggunakan titik.

Ketiga untuk subgraf secara langsung melibatkan bagaimana grafik akan ditata oleh mesin. Jika nama subgraf dimulai dengan *cluster*, *Graphviz* mencatat subgraf sebagai subgraf *cluster* khusus. Jika didukung, mesin akan melakukan tata letak sehingga simpul milik cluster digambar bersama, dengan keseluruhan gambar cluster yang ada di dalam persegi panjang yang melintang. Subgraf *cluster* bukan bagian dari bahasa DOT, namun hanya konvensi sintaks yang dipatuhi oleh mesin.

2.5.3 Atribut Menggambar

Dalam membuat graf dibutuhkan beberapa atribut untuk menyempurnakan gambar. Atribut tersebut berisi

1. Bentuk dan Label.
   Pada bentuk dan label nantinya akan ditentukan *node* akan berbentuk apa dan label pada node akan berisi apa. Secara *default* bentuk dari node sendiri adalah elips. Tetapi ada bentuk lain yang diberikan untuk *node* yaitu kotak, lingkaran, polygon, dll.

2. Tampilan Graf
   Simpul dan tepi memiliki atribut warna dan gaya. Penggunaan warna dalam membuat graf memiliki beberapa syarat. Pertama hindari menggunakan terlalu banyak warna cerah. Kedua, ketika node dipenuhi warna gelap label nampaknya lebih mudah dibaca dengan *fontcolor* = putih dan *fontname* = *Helvetica*. Ketiga, menentukan ruang warna dengan mendefinisikan *nodecolor*, *edgecolor*, atau *graphcolor* dalam file library. Misalnya, untuk menggunakan warna RGB, letakkan baris berikut di file lib.ps. / nodecolor setrgbcolor bind def. Gunakan opsi baris perintah -l untuk memuat file ini. $dot - Tps - llib.psfile.dot - ofile.ps$

3. Ukuran Gambar dan Jarak
   Seringkali gambar yang dibuat dengan ukuran dan pemisahan *nodes default* terlalu besar untuk target atau untuk ruang yang diizinkan untuk gambar dalam dokumen. Ada beberapa cara untuk mencoba mengatasi masalah ini. Pertama, melihat bagaimana titik pada ukuran tata letak akhir. Tata letak awalnya dibuat secara internal dengan ukuran awal, dengan menggunakan pengaturan *default*. Secara default, *nodes* paling sedikit 0,75 inci dengan lebar 0,5; *font* adalah 14, *nodes* dipisahkan paling sedikit 0,25 dan diberi peringkat oleh 0,5 Tidak ada batasan ukuran atau aspek rasio gambar, jadi jika grafiknya besar, tata letaknya juga

besar. Jika tidak menentukan ukuran atau rasio, maka ukuran awal akan dicetak. Cara termudah untuk mengontrol ukuran output gambar adalah dengan mengatur ukuran = x; y pada file grafik (atau pada baris perintah menggunakan -G). Ini menentukan kotak pembatas tata letak akhir.

Tabel untuk atribut menggambar sebagai berikut:

(a) *Node Attributes*, Pada Tabel di bawah ini menunjukkan apa saja isi dari *Node Attributes*

Tabel 2.1: *Node Attributes*

Nama	Default	Value
color	black	warna bentuk node
fontcolor	black	warna huruf
fontname	times-roman	jenis font
fontsize	14	ukuran dari font
height, width	.5,.75	tinggi dan panjang dalam bentuk inchi
label	node name	kalimat
layer	overlay range	semua id
shape	ellipse	ellipse, box, circle, doublecircle, plaintext, polygon
shapefile		external EPSF file if epsf shape
style		graphics options (bold, dotteed, filled)

(b) *Edge Attributes*, Pada Tabel di bawah ini menunjukkan apa saja isi dari *Edge Attributes*

Tabel 2.2: *Edge Attributes*

Nama	Default	Value
color	black	warna garis
decorate		gambar yang menghubungkan label
dir	forward	forward, back, both, or none
fontcolor	black	warna forn
fontname	times-roman	jenis font
fontsize	14	ukuran fonr
id		optional value
label		label, if not empty
layer	overlay range	all id
minlen	1	minimum rank distance between head and tail
style		graphics options (bold, dotteed, filled)
weight	1	integer reflecting inmportance of edge

(c) *Graph Attributes*, Pada Tabel di bawah ini menunjukkan apa saja isi dari *Graph Attributes*

Tabel 2.3: *Graph Attributes*

Nama	Default	Value
center		when true, centers drawing on page
cluster rank	local	may be global or none
color	black	node shape color
fontcolor	black	type face color
fontname	times-roman	PostScript font family
fontsize	14	point size of label
label		any string
layerseq		id:id:id
margin	.5,.5	margin include in pages
mclimit	1.0	if set to f adjusts mincross iterations by (f)
nodesep	.25	separation between nodes in inches
ordering		out (for ordered edges)
page		unit of pagination
rank		same, min, max
rankdir	TB	LR(left to right) or TB(top to bottom)
ranksep	.75	separation between ranks in inches
ratio		aprroximate aspect ratio desired
size		drawing bounding box in inches

2.6 Visualisasi Graf dengan Viz.js

JSON sebagai salah satu format terbuka digunakan untuk membuat graf. Graf ini dihasilkan dengan menggunakan *viz.js* yang merupakan mesin pembaca *DOT*. *DOT* sendiri dibuat dengan melihat struktur JSON. Agar graf dapat ditampilkan pada suatu *web browser*, salah satu caranya adalah dengan menggunakan *viz.js*. Visualisasi ini dapat dilakukan dengan menggunakan javascript dan HTML5 untuk membuat sebuah graf pada halaman *web*. Hal pertama yang perlu dilakukan adalah dengan melakukan *install viz.js* di https://github.com/mdaines/viz.js/releases.[5] Lalu data tersebut diletakan pada *file* yang akan digunakan. Berikut adalah contoh dalam menggunakan viz.js.

```
<html>
    <body>
        <div id="graph"></div>
        <script src="assets/js/jquery.js"></script>
        <script src="assets/js/viz.js"></script>
        <script>
            $.get('kurikulum.dot', function (res) {
                var graph = Viz(res, { format: "svg", engine: "dot" });
                $("#graph").append(graph);
            })
        </script>
    </body>
```

[5] "viz", https://github.com/mdaines/viz.js/releases

</html>

Terdapat beberapa opsi parameter yang dapat digunakan untuk merubah tampilan dari graf yang akan ditampilkan, yaitu:

- *format* menetapkan *format* keluaran, dan hasilnya salah satu dari "svg", "xdot", "plain", "ps", "json", atau "png-image-element".
- *engine*, mengatur mesin *Graphviz* untuk digunakan, salah satunya "circo", "dot", "fdp", "neato", "osage", or "twopi".

BAB 3

ANALISIS

Berdasarkan hasil studi pustaka yang telah dilakukan, pada bab ini akan dijelaskan hasil analisis yang berupa spesifikasi dari perangkat lunak, diagram use-case, skenario.

3.1 Analisis JSON yang akan dibuat

Struktur JSON yang akan digunakan adalah dalam bentuk array yang di dalamnya memiliki objek. Objek di dalam JSON ini menjadi acuan dalam membuat pohon kurikulum. Contoh penulisan JSON sebagai berikut:

```
[
  {
    "kode": "AIF181101",
    "nama": "Computational Thinking",
    "prasyarat": {
      "tempuh": [],
      "lulus": [],
      "bersamaan": [],
      "angkatan" : []
    },
    "sks": 3,
    "wajib": "true",
    "semester": 1
  }
]
```

Pada bentuk di atas JSON memiliki beberapa objek sebagai acuan.

1. **kode**, berisikan kode matakuliah yang akan di ambil di dalam pembuatan pohon kurikulum.
2. **nama**, berisikan nama mata kuliah yang ada di semester 1 sampai semester 8.
3. **prasyarat**, prasyarat memiliki 4 isi yaitu:
    - **tempuh**, berisikan kode mata kuliah yang menunjukan mahasiswa sudah mengambil mata kuliah yang menjadi syarat atau belum.

- **lulus**, berisikan kode mata kuliah yang menunjukan mahasiswa sudah mengambil mata kuliah tersebut dan lulus mata kuliah tersebut.
- **bersamaan**, berisikan kode mata kuliah yang menunjukan mahasiswa dapat mengambil mata kuliah yang memiliki syarat bersamaan dengan mata kuliah yang sudah tempuh.
- **angkatan**, berisikan tahun angkatan yang merepresentasikan berlaku syarat mata kuliah.

4. **sks**, menunjukkan berapa banyak tanggungan belajar mahasiswa.
5. **wajib**, menunjukkan mata kuliah tersebut jenisnya wajib atau pilihan.
6. **semester**, menunjukkan mata kuliah tersebut ada di semester berapa.

3.2 Analisis Perangkat Lunak yang Dibangun

Dari pengetahuan yang diperoleh melalui studi pustaka yang dilakukan. Telah ditentukan beberapa analisis untuk membangun Perangkat Lunak Pohon Kurikulum 2018 menggunkan JSON. Berikut beberapa analisis yang telah diambil dari bab 2:

- **Menggunakan JSON yang konversikan ke dalam *DOT Language***
  Untuk membuat graf harus dilakukan konversi dari JSON ke dalam *DOT Language*. *DOT Language* sendiri digunakan untuk menghasilkan graf yang akan menampilkan pohon kurikulum.
- **JSON akan disimpan di URL berikut** `github.com`
  JSON akan di simpan di `github.com`. Tujuannya agar JSON menjadi format data terbuka. Setelah disimpan di dalam *github* JSON dapat dilihat oleh siapa saja. Karena *github* sendiri merupakan *open source*.
- **Perangkat Lunak Menghasilkan Graf Berbentuk Pohon Kurikulum**
  Perangkat lunak yang akan dibangun akan menghasilkan graf yang berbentuk pohon kurikulum. Pohon kurikulum ini diperlukan agar mahasiswa mengetahui mata kuliah yang akan di ambil di semester baru.
- **Perangkat Lunak akan Menampilkan Mata Kuliah**
  Perangkat Lunak yang dibangun setelah menghasilkan pohon kurikulum akan menampilkan mata kuliah yang ada di kurikulum baru. Mata Kuliah akan berisi mata kuliah wajib, mata kuliah pilihan, dan mata kuliah pilihan wajib.

3.3 Kebutuhan Data Terbuka

Tujuan utama dari Data Terbuka adalah memaksimalkan penggunaan data seluas-luasnya untuk menciptakan suatu nilai tambah. Untuk mencapai hal tersebut, Data Terbuka memberikan dua komponen dari keterbukaan, yaitu: terbuka secara teknis dan terbuka secara legal.

Secara sederhana, yang dimaksud dengan data yang terbuka secara legal adalah tidak ada halangan dari sisi aturan atau hukum untuk menggunakan data tersebut. Data boleh dan dapat digunakan oleh siapa saja, untuk tujuan apa saja, kapan saja, tanpa prasyarat apapun, kecuali dengan memberikan atribusi kepada pemilik data. Oleh karena itu, dibutuhkan suatu mekanisme

yang jelas untuk mencapai hal-hal tersebut. Salah satu pilihan mekanisme yang bisa digunakan adalah pemberian lisensi kepada data yang hendak dijadikan terbuka secara legal.

Pembubuhan lisensi atas data diartikan sebagai pemberian hak menggunakan data untuk kepentingan tertentu sesuai dengan syarat dan ketentuan yang tertera di dalam lisensi tersebut tanpa meniadakan hak cipta atas data tersebut. Dengan pemberian lisensi kepada data, data dapat dipergunakan secara mudah (dengan syarat dan ketentuan tertentu) tanpa harus meminta izin melalui mekanisme hak cipta.

Untuk kebutuhan data terbuka yang bebas biaya dapat menggunakan *github*. Di dalamnya harus menambahkan lisensi untuk pemakaian *github*. Lisensi *Creative Commons* adalah salah satu lisensi publik yang memungkinkan pemilik karya untuk mendistribusikan dan memperbolehkan penggunaan karyanya secara bebas. [5] Ada beberapa tipe lisensi *Creative Commons* yang dapat dipergunakan ketika seorang pemilik karya hak cipta hendak mendistribusikan dan memperbolehkan penggunaan atas karyanya.

Tipe-tipe lisensi Creative Commons antara lain:

1. Attribution (BY)
   Mengizinkan orang lain untuk menyalin, mendistribusikan, menampilkan, serta membuat karya turunan berdasarkan suatu karya hanya jika orang tersebut memberikan penghargaan pada pencipta atau pemberi lisensi dengan cara yang disebutkan dalam lisensi.

2. ShareAlike (SA)
   Mengizinkan orang lain untuk mendistribusikan suatu karya turunan hanya di bawah suatu lisensi yang identik dengan lisensi yang diberikan pada karya aslinya.

3. Non-Commercial (NC)
   Mengizinkan orang lain menyalin, mendistribusikan, menampilkan, serta membuat karya turunan berdasarkan suatu karya hanya untuk tujuan non-komersial.

Dari tipe-tipe lisensi di atas, maka lisensi *Creative Commons* yang disarankan untuk digunakan untuk Data Terbuka pada proyek ini adalah lisensi *Creative Commons by Attribution* (CC-BY) dikarenakan lisensi CC-BY inilah yang paling mendekati persyaratan keterbukaan dari Data Terbuka, yaitu: pengguna dapat mendistribusikan data dan menggunakan data secara bebas baik untuk kepentingan komersil, maupun non-komersil tanpa syarat kecuali dengan memberikan atribusi kepada pemilik data. Selain itu, lisensi CC-BY ini bisa dikatakan sebagai lisensi publik paling mudah dipakai untuk memaksimalkan penggunaan atas data tanpa membuat pengguna data khawatir atas status legalitas dari penggunaan data dan akibat dari penggunaan data tersebut.

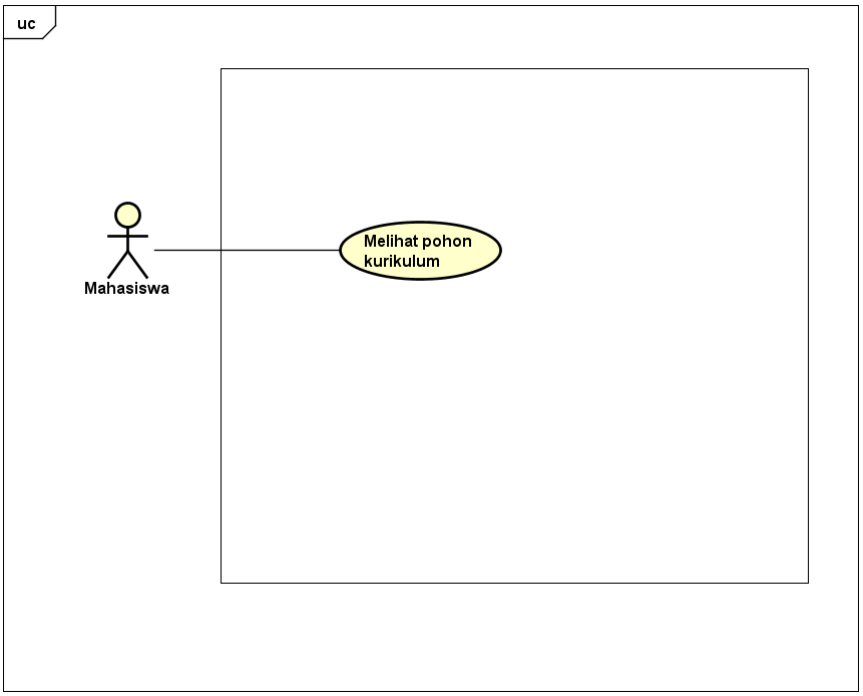
3.4 Spesifikasi Perangkat Lunak yang Dibangun

Berawal dari pengetahuan yang diperoleh melalui studi pustaka yang telah dilakukan, maka selanjutnya menentukan spesifikasi website perangkat lunak yang akan dibangun. Perangkat lunak ini hanya memiliki beberapa spesifikasi, antara lain:

- membaca dari JSON ke dot
- Menggunakan *DOT Language*

- Visualisasi menggunakan *viz.js*
- Menampilkan graf

3.4.1 Use Case dan Skenario



Gambar 3.1: Use Case Pohon Kurikulum

Berikut keterangan dan skenario dari Gambar 3.1:

1. Nama *use case* : Melihat Pohon Kurikulum
   Aktor : Mahasiswa Deskripsi : Aktor melihat isi pohon kurikulum Prakondisi : Aktor belum mengetahui pohon kurikulum baru Skenario normal : Aktor melihat pohon kurikulum Eksepsi : Aktor telah mengetahui pohon kurikulum yang di pakai di kurikulum baru

BAB 4

PERANCANGAN

Pada bab ini akan dibahas mengenai perancangan perangkat lunak yang diimplementasi pada pohon kurikulum.

4.1 Kebutuhan *Input* dan *Output*

Perancangan perangkat lunak pohon kurikulum dengan men *generate* dari JSON ke *DOT*. Input perangkat lunak merupakan kode, nama, prasyarat, sks, semester, dan wajib. Kebutuhan input dan output perangkat lunak:

- *Input*
  Kebutuhan *input* pada pohon kurikulum adalah
  1. **kode**, berisikan kode mata kuliah
  2. **nama**, berisikan nama mata kuliah
  3. **sks**, memberitahukan kepada mahasiswa mata kuliah yang akan diambil memiliki beban berapa banyak.
- *Output*
  Output dari perangkat lunak adalah pohon kurikulum.

4.2 Perancangan Kebutuhan Perangkat Lunak

4.3 Perancangan Antarmuka

BAB 5

IMPLEMENTASI DAN PENGUJIAN PERANGKAT LUNAK

Bab ini terdiri atas dua bagian, yaitu Implementasi Perangkat Lunak dan Pengujian Perangkat Lunak. Bagian implementasi berisi penjelasan lingkungan pengembangan perangkat lunak dan hasil implementasi. Sedangkan bagian pengujian berisi hasil pengujian fungsional terhadap perangkat lunak yang telah dibangun.

5.1 Implementasi Perangkat Lunak

Pada bagian ini akan dibahas mengenai implementasi perangkat lunak yang telah dibangun. Sub bab ini terdiri atas tiga bagian, yaitu lingkungan perangkat lunak, hasil implementasi perangkat lunak, dan Pengujian fungsional.

5.1.1 Lingkungan Implementasi Perangkat Lunak

Dalam proses membangun perangkat lunak ini digunakan spesifikasi perangkat sebagai berikut:

1. Lingkungan Pembangunan

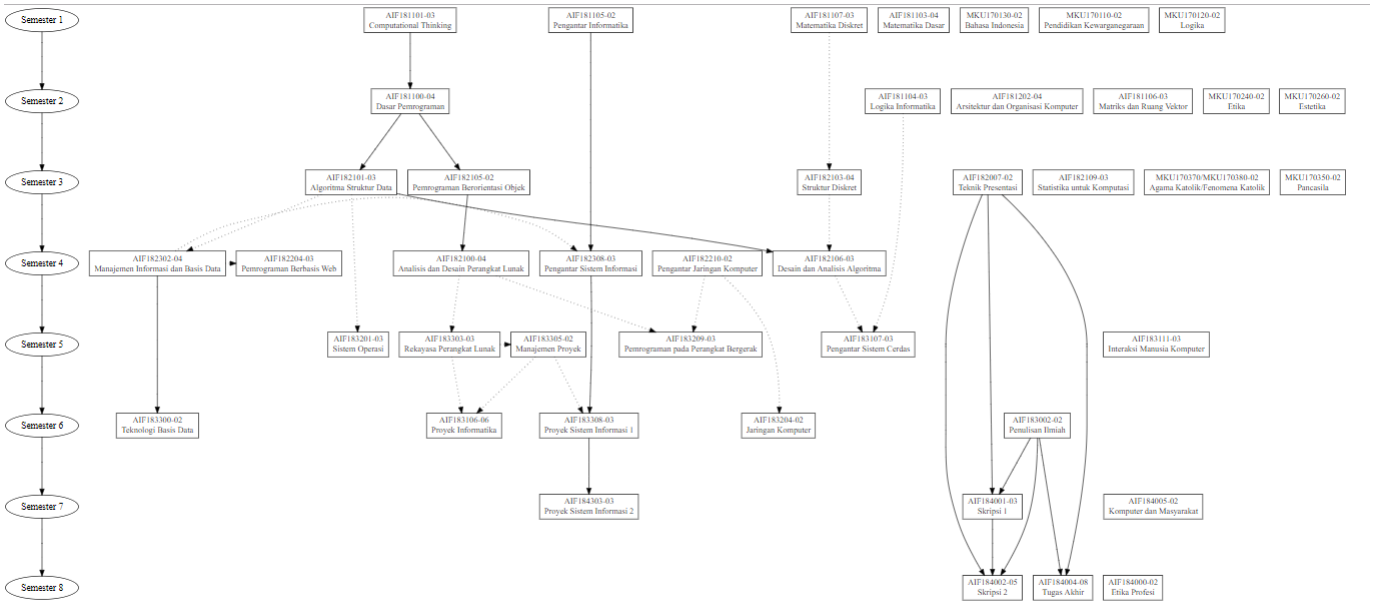
    - Processor : IntelÂő CoreâĎći7-4702MQ 2.2-3.2GHz
    - RAM : 4.00 GB
    - Harddisk : 1TB
    - VGA : NVIDIA GeForce GT 740M
    - Sistem Operasi Komputer : Windows 10 Education 64-bit

2. Lingkungan Implementasi

    - Sistem Operasi Server : *Node.js*
    - Tools : *Visual Studio Code*
    - Bahasa Pemrograman : Javascript
    - Framework : *viz.js*

5.1.2 Hasil Implementasi

Kode program pada perangkat lunak ditulis dalam bahasa pemrograman *javascript* dengan cara men *generate* dari JSON ke *DOT*. Hasil implementasi berupa pohon kurikulum yang visualisasinya menggunakan *viz.js*. Perangkat lunak dapat dilihat seperti gambar 5.1.2 di bawah ini.



Gambar 5.1: pohon kurikulum

Pada gambar 5.1.2 terlihat setiap semester memiliki mata kuliah yang berisi kode, sks, dan nama. Lalu mata kuliah yang memiliki prasyarat akan ditunjuk oleh panah. Syarat yang menjadi patokan adalah syarat tempuh, syarat lulus, atau pengambilan secara bersamaan.

5.2 Pengujian Perangkat Lunak

Pada bab ini akan dibahas mengenai pengujian perangkat lunak yang dibangun. Pengujian yang dilakukan adalah pengujian fungsional dan pengujian eksperimental. Pengujian fungsional bertujuan untuk memastikan bahwa seluruh fungsi perangkat lunak yang dibangun berjalan sesuai dengan rencana dan pengujian eksperimental bertujuan untuk mengetahui apa saja *engine* yang dapat dipakai dalam membangun perangkat lunak.

5.2.1 Pengujian Fungsional

Dalam sub bab ini akan dilakukan pengujian fungsional untuk mengetahui fungsi-fungsi yang terdapat pada perangkat lunak dapat berjalan sesuai dengan yang diharapkan. Status pengujian dibagi menjadi dua yaitu "OK" dan "gagal". Di bawah ini Pengujian fungsi pohon kurikulum:

1. Langkah Pengujian : Memanggil fungsi rankSep Hal yang diharapkan : Pada saat memanggil fungsi ranksep keluar node semester dan kode mata kuliah wajib
   Hasil Pengujian : Hasil pengujian node semester satu sampai delapan keluar dan kode mata

kuliah wajib keluar
Status : OK

2. Langkah Pengujian : Memanggil fungsi nodesMatkul Hal yang diharapkan : Pada saat memanggil fungsi nodesMatkul akan keluar label yang berisi kode, sks, dan nama mata kuliah
Hasil Pengujian : kode, sks, dan nama mata kuliah wajib berhasil ditampilkan
Status : OK

3. Langkah Pengujian : Memanggil fungsi edgesMatkul
Hal yang diharapkan : Mata kuliah yang mempunyai prasyarat bisa diketahui melalui petunjuk arah
Hasil Pengujian : Hasilnya mata kuliah yang memiliki prasyarat akan ditunjuk sesuai prasyarat. Jika syaratnya lulus maka garis akan lurus jika syaratnya tempuh garis putus-putus
Status : OK

5.2.2 Pengujian Eksperimental

Pengujian eksperimental dilakukan dengan cara membuat beberapa tugas dalam pengumpulan datanya. Tujuan yang ingin dicapai dalam pengujian eksperimental.

DAFTAR REFERENSI

[1] Nasional, D. P. (2007) *KBBI*, keempat edition. PT Gramedia Pustaka Utama, Indonesia.

[2] Munir, R. (2005) *Matematika Diskrit*, revisi kelima edition. INFORMATIKA, Indonesia.

[3] International, O. K. (2009) opendatahandbook. http://opendatahandbook.org/guide/id/introduction/. 2009.

[4] North, E. K. . S. C. (1996) *Drawing Graphs with Dot*, rfirst edition. ATT Bell Laboratories, Belgium.

[5] Indonesia, C. (2015) Peran lisensi terbuka di bidang data terbuka. https://creativecommons.or.id/2017/07/laporan-global-open-data-index-peran-lisensi-terbuka-di-bidang-open-data-di-tahun-2017/ 2017.

LAMPIRAN A

KODE PROGRAM

Listing 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
```

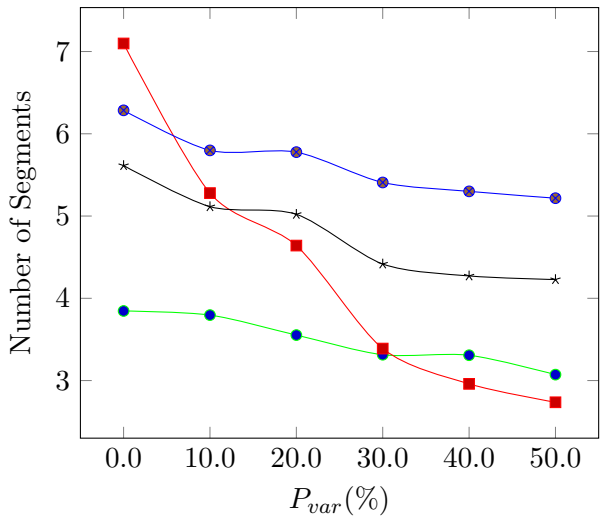
Listing A.2: MyCode.java

```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);
        }
    }

}
```

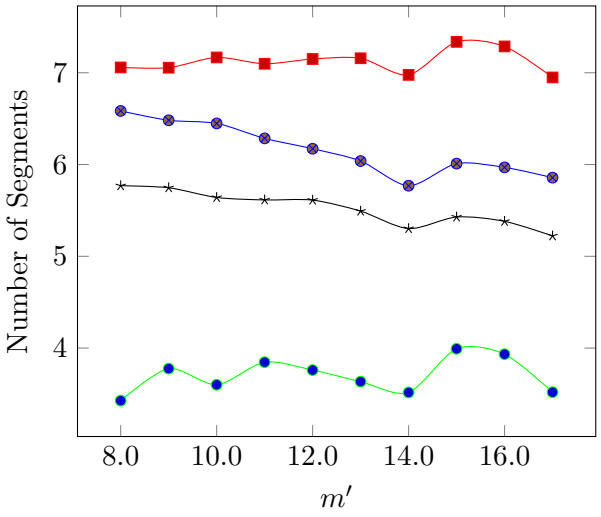
LAMPIRAN B

HASIL EKSPERIMEN

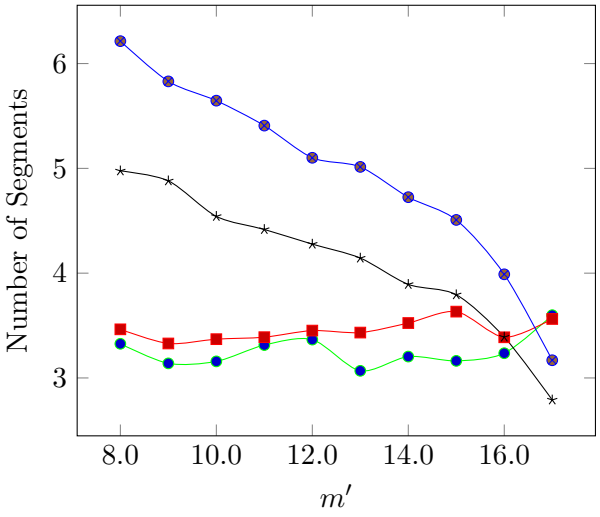
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



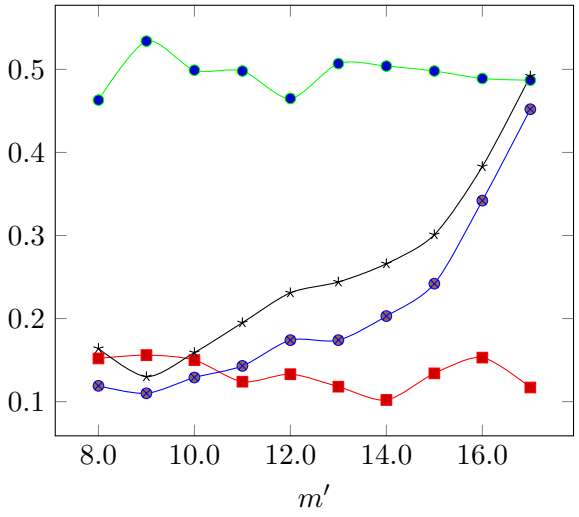
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4