

PERKAKAS COMMAND LINE KIRI

ALFRED APRIANTO LIAUNARDI-6181801014

1 Data Skripsi

Pembimbing utama/tunggal: **Pascal Alfadian Nugroho, M.Comp.**

Pembimbing pendamping: -

Kode Topik : **PAN5201**

Topik ini sudah dikerjakan selama : **1 semester**

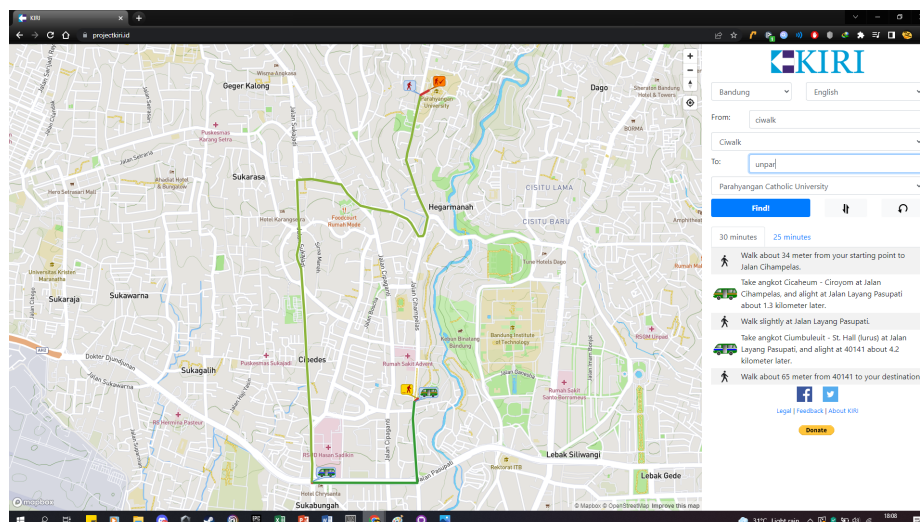
Pengambilan pertama kali topik ini pada : Semester **52 - Genap 21/22**

Pengambilan pertama kali topik ini di kuliah : **Skripsi 1**

Tipe Laporan : **B -** Dokumen untuk reviewer pada presentasi dan **review Skripsi 1**

2 Latar Belakang

Project KIRI¹ (akan disingkat sebagai KIRI dalam dokumen ini) adalah sebuah perangkat lunak berbasis web yang dibuat untuk membantu mengurangi efek dari kemacetan. KIRI mengurangi dampak kemacetan dengan membantu penggunanya, baik masyarakat maupun turis, dalam menggunakan salah satu sarana transportasi umum yang ada di Indonesia, yaitu angkutan kota (angkot). Cara KIRI mempermudah penggunaan angkot adalah dengan menunjukkan rute yang akan ditempuh, beserta langkah-langkah yang harus dilakukan oleh pengguna yang ingin berpergian dari satu titik ke titik lain, mulai dari seberapa jauh pengguna harus berjalan untuk menaiki angkot yang bersangkutan, di mana pengguna harus naik atau turun, seberapa jauh lagi pengguna harus berjalan sampai ke titik tujuan, dan seberapa lama estimasi waktu perjalanan yang akan ditempuh. Untuk kebutuhan pembuatan perangkat lunak yang memanfaatkan fitur dari KIRI, tersedia juga REST API KIRI yang dapat digunakan secara praktis. Adapun tampilan dari halaman web ini dapat dilihat di gambar 1.



Gambar 1: Tampilan halaman web KIRI, yang menunjukkan rute dari Cihampelas Walk ke Universitas Katolik Parahyangan.

Sementara itu, dalam komputer, salah satu dari sekian banyak tipe perangkat lunak adalah *command line*. *Command line* (*command line interpreter*, atau *command line interface*) adalah sebuah perangkat lunak

¹<https://projectkiri.id>

berupa sebuah kotak/*window* yang memuat teks berupa perintah-perintah,² yang menerima masukan dari pengguna dan menjalankannya.[1] Perintah-perintah ini hanya berupa gabungan dari teks and simbol-simbol berupa karakter, tanpa ada tambahan gambar grafis apapun. Singkatnya, tipe perangkat lunak ini bukan merupakan tipe yang paling indah untuk dilihat oleh para pengguna, tetapi jika digunakan dengan tepat, maka jenis perangkat lunak ini bisa menyuruh komputer untuk melakukan banyak sekali perintah-perintah dengan sangat cepat dan sangat efektif.

Pada skripsi ini akan dibuat sebuah perangkat lunak berupa perkakas *command line* (*command line tool*) yang dapat menjalankan fungsi-fungsi API dari KIRI. Perangkat lunak ini, seperti jenisnya, akan dibuat murni sebagai perkakas yang dijalankan dari *command line* (terminal, cmd, PowerShell, dll.), dan tampilan akhir dari perangkat lunak akan berupa *command line interface* tanpa tambahan *graphical user interface*. Keseluruhan dari perangkat lunak ini akan dibangun dalam bahasa C.

3 Rumusan Masalah

1. Bagaimana membangun perkakas *command line* yang dapat mengimplementasikan fitur-fitur API KIRI dalam bahasa C?
2. Bagaimana integrasi perkakas *command line* KIRI dapat dilakukan dengan perkakas-perkakas *command line* lainnya di Linux?

4 Tujuan

1. Membangun perkakas *command line* yang dapat mengimplementasikan fitur-fitur API KIRI dalam bahasa C.
2. Melakukan integrasi perkakas *command line* KIRI dengan perkakas-perkakas *command line* lainnya di Linux.

5 Deskripsi Perangkat Lunak

Perangkat lunak akhir yang akan dibuat memiliki fitur minimal sebagai berikut:

- Pengguna dapat memasukkan lokasi awal dan tujuan akhir sebagai masukan dari perangkat lunak.
- Pengguna dapat melihat langkah-langkah yang harus ditempuh dalam perjalanan, mulai dari angkot mana saja yang harus dinaiki, ke mana pengguna harus berjalan kaki untuk bisa mencapai angkot terdekat dari lokasi terakhir pengguna, sampai seberapa jauh pengguna harus berjalan untuk mencapai tujuan akhir.
- Pengguna dapat melihat jarak yang harus ditempuh untuk setiap langkahnya.
- Pengguna dapat melihat seberapa lama waktu perjalanan untuk setiap langkahnya.

6 Detail Perkembangan Pengerjaan Skripsi

Detail bagian pekerjaan skripsi sesuai dengan rencan kerja/laporan perkembangan terakhir :

1. Melakukan eksplorasi fungsi-fungsi perangkat lunak KIRI serta eksplorasi cara implementasi API KIRI.

²[Ubuntu Tutorials - The Linux command line for beginners: 3. Opening a Terminal](#)

Status : Ada sejak rencana kerja skripsi.

Hasil : KIRI merupakan sebuah perangkat lunak berbasis web yang berfungsi untuk menyelesaikan (atau setidaknya mengurangi) dampak dari masalah-masalah yang dapat diselesaikan oleh transportasi umum/publik di Indonesia, seperti pemanasan global, kemacetan, atau peningkatan harga bensin. Selain itu, turis mancanegara juga memilih untuk menaiki transportasi umum, karena jenis sarana transportasi tersebut tidak hanya jauh lebih murah, tetapi juga memberikan kesempatan yang mudah kepada mereka untuk melihat seluk-beluk dari kota-kota yang mereka kunjungi.³

Di halaman web KIRI, pengguna dapat memasukkan input berupa lokasi awal dan lokasi tujuan dan KIRI akan menghasilkan seluruh langkah yang harus ditempuh oleh pengguna untuk sampai ke lokasi tujuan, dengan menggunakan angkot. Keluaran ini sudah meliputi kode angkot mana saja yang harus dinaiki, dan juga seberapa jauh pengguna harus berjalan kaki untuk sampai ke lokasi rute angkot berikutnya.

KIRI juga memiliki sebuah API yang dapat digunakan untuk keperluan pengembangan perangkat lunak. Seluruh permintaan (*request*) yang dilakukan melalui API KIRI harus dilakukan sebagai permintaan tipe GET ke <https://projectkiri.id/api>, beserta parameter-parameter yang dibutuhkan.

Permintaan tersebut harus memiliki parameter-parameter seperti terlihat di bawah ini.⁴

- **version**

Kemungkinan nilai: 2

Parameter ini merupakan tanda bagi API untuk menggunakan protokol versi 2.

- **mode**

Kemungkinan nilai: *findroute*

Parameter ini merupakan mode dari servis/jasa API yang akan digunakan oleh pengguna. Untuk mode *findroute*, jasa yang akan digunakan adalah jasa pencarian rute dengan angkot.

- **locale**

Kemungkinan nilai: *en* atau *id*

Parameter ini mengatur bahasa apa yang akan digunakan dalam keluaran API nantinya—*en* berarti keluaran akan menggunakan bahasa Inggris, dan *id* berarti keluaran akan menggunakan bahasa Indonesia.

- **start**

Kemungkinan nilai: *lat, lng*; dalam bentuk desimal

Parameter ini merupakan nilai *latitude* dan *longitude* dari titik awal perjalanan pengguna.

- **finish**

Kemungkinan nilai: *lat, lng*; dalam bentuk desimal

Parameter ini merupakan nilai *latitude* dan *longitude* dari titik akhir/tujuan perjalanan pengguna.

- **presentation**

Kemungkinan nilai: *desktop*

Parameter ini hanya digunakan untuk fitur *backwards compatibility*.

- **apikey**

Kemungkinan nilai: angka heksadesimal 16-digit

Parameter ini berisi kunci API pribadi yang harus digenerasi terlebih dahulu sebelum API dapat digunakan.

³<https://projectkiri.github.io/#about-kiri>

⁴<https://github.com/projectkiri/Tirtayasa/wiki/KIRI-API-v2>

Sedangkan, respon yang diberikan oleh API berupa sebuah objek JSON yang selalu memiliki setidaknya dua variabel, yaitu:⁵

- **status**

Kemungkinan nilai: `ok` atau `error`

Variabel ini menandakan apakah permintaan berhasil diproses atau tidak. Jika permintaan berhasil diproses, variabel ini akan bernilai `ok`, dan jika tidak, variabel ini akan bernilai `error`.

- **message**

Variabel ini bisa berisi dua macam objek. Jika permintaan dari user tidak berhasil diproses, atau dalam kata lain, terjadi sebuah *error*, maka variabel ini akan berisi string yang merupakan pesan *error* serta alasan spesifik mengapa *error* tersebut terjadi. Di lain sisi, jika permintaan dari user berhasil diproses, variabel ini akan mengalami dua perubahan utama. Pertama, nama variabel ini akan berubah menjadi **routingresults**, dan kedua, isi dari variabel ini akan menjadi sebuah *array* JSON yang merupakan respon dari API KIRI berupa keluaran yang akan dilihat oleh pengguna. *Array* JSON ini sendiri terbagi menjadi beberapa variabel lainnya, yang dapat dilihat di daftar di bawah ini.

- **steps**

Tipe: `array`

Variabel ini merepresentasikan satu buah langkah yang harus ditempuh oleh pengguna. Adapun *array* ini sendiri berisi variabel-variabel berikut:

- * Tipe transportasi

Tipe sarana transportasi yang harus dipakai oleh pengguna. Jika pengguna harus berjalan kaki, variabel ini akan berisi `walk`. Jika pengguna harus menaiki angkot, variabel ini akan berisi `angkot`.

- * Kode angkot

Variabel ini menunjukkan angkot mana yang harus dinaiki oleh pengguna di langkah tersebut. Jika penggunaan angkot tidak dimungkinkan pada langkah ini (pengguna harus berjalan kaki), variabel ini akan berisi `walk`.

- * Array *latitude* dan *longitude* lokasi

Array nilai-nilai desimal *latitude* dan *longitude* dari berbagai titik lokasi yang terdapat dalam rute.

- * Deskripsi langkah

Deskripsi langkah yang harus ditempuh, dalam bahasa natural. Bahasa yang digunakan tergantung parameter `locale` yang diatur dalam masukan.

- * URL untuk mendapatkan tiket kendaraan

Tautan untuk mendapatkan tiket angkutan umum, jika diperlukan. Jika transportasi pada langkah tersebut tidak memerlukan tiket, variabel ini akan berisi `null`.

- * URL editor rute

Tautan untuk meng-edit rute, jika situasinya memungkinkan. Jika tidak, variabel ini akan berisi `null`.

- **traveltime**

Tipe: `string`

Variabel ini berisi estimasi jangka waktu yang diperlukan untuk menyelesaikan langkah tersebut.

2. Mempelajari perkakas-perkakas *command line* sejenis.

Status : Ditambahkan pada Skripsi 1.

⁵<https://github.com/projectkiri/Tirtayasa/wiki/KIRI-API-v2>

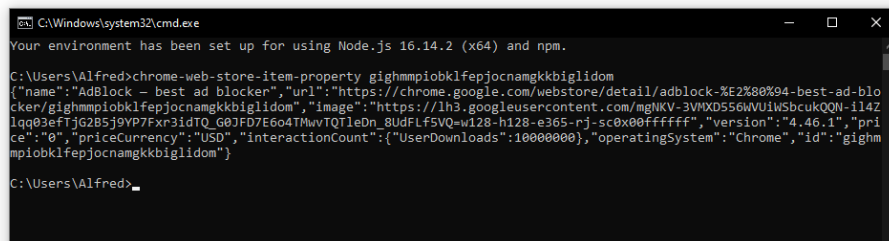
Hasil : Perkakas-perkakas sejenis yang dianalisis meliputi *Chrome Web Store Item Property CLI* dan *iTunes Search API*.

- *Chrome Web Store Item Property CLI*

Perkakas ini dapat digunakan melalui *command prompt* dengan cara mengetikkan perintah sebagai berikut.

```
chrome-web-store-item-property <identifikasi>
```

Dengan *identifikasi* berupa ID dari ekstensi yang diinginkan. Jadi, misalkan pengguna memasukkan `gighmmpiobklfepjocnamgkbbiglidom` sebagai ID yang akan digunakan sebagai *identifikasi*, maka perkakas ini akan mengembalikan metadata dari ekstensi “AdBlock” sebagai keluarannya. Contoh penggunaan perkakas ini dapat dilihat di gambar 2.



Gambar 2: Contoh penggunaan perkakas *Chrome Web Store Item Property CLI*.

Sedangkan, keluaran dari perkakas ini merupakan sebuah objek JSON dengan properti-properti sebagai berikut.

- **name**
Nama dari ekstensi yang dicari metadatanya.
- **url**
URL halaman web dari ekstensi yang dicari di *web store* Google Chrome.
- **image**
Logo (dan ikon *thumbnail*) dari ekstensi yang dicari metadatanya.
- **version**
Nomor versi dari ekstensi.
- **price**
Harga dari ekstensi. Jika ekstensi tidak memiliki harga yang perlu dibayarkan (gratis), properti ini akan bernilai 0.
- **priceCurrency**
Kode mata uang dari harga ekstensi. Jika ekstensi tidak memiliki harga yang perlu dibayarkan, properti ini akan berisi “USD”.
- **interactionCount**
Properti ini berisi interaksi-interaksi pengguna yang tercatat sebagai data di halaman *web store* ekstensi. Pada saat pembuatan skripsi ini, properti ini hanya memiliki satu buah subproperti, yaitu `userDownloads`, yang menandakan berapa kali ekstensi ini telah diunduh oleh pengguna di manapun.
- **operatingSystems**
Menandakan di peramban mana ekstensi versi ini dapat diinstal. Karena ekstensi-ekstensinya berada di *web store* Chrome,
- **ratingValue** (tidak digunakan lagi)
Peringkat yang diberikan oleh para pengguna ekstensi ini. Nilai dari properti ini berupa skala

desimal dari 0.00 sampai dengan 5.00. Di versi terbaru dari perkakas ini, properti ini tidak lagi tersedia dalam keluarannya.

- `ratingCount` (tidak digunakan lagi)

Jumlah pengguna yang telah menilai/memberi peringkat ke ekstensi ini. Di versi terbaru dari perkakas ini, properti ini tidak lagi tersedia dalam keluarannya.

- `id`

Properti ini mengandung ID dari ekstensi tersebut. Nilai dari properti ini akan sama dengan ID yang digunakan sebagai parameter masukan perkakas.

- *iTunes Search API*

Perkakas *command line* ini berfungsi untuk melakukan pencarian melalui API iTunes, sehingga seakan-akan pengguna langsung melakukan pencarian di iTunes sendiri. Hasil pencarian yang dilakukan termasuk judul lagu, nama artis, ataupun nama album, dan pengguna dapat memilih secara spesifik objek apa yang ingin dicari.

Perkakas ini dapat digunakan melalui *command prompt* dengan cara mengetikkan perintah sebagai berikut.

```
itunes-search-api <input> [<options>]
```

Dengan *input* berupa nama dari objek yang dicari. Perkakas ini juga memiliki opsi yang masing-masing memiliki parameter tersendiri untuk mempersempit hasil pencarian. Adapun opsi-opsi tersebut dapat dilihat di daftar di bawah ini.

- `country`

Kemungkinan nilai: Kode negara dua huruf

Opsi ini menerima parameter berupa kode negara asal dari album atau artis yang dicari.

- `entity`

Kemungkinan nilai: `song`, `musicArtist`, atau `album`

Menandakan jenis objek/entitas yang ingin dicari. Opsi ini dapat bernilai `song` untuk pencarian berbasis judul lagu, `musicArtist` untuk pencarian nama artis, atau `album` untuk pencarian nama album. Jika opsi ini tidak dipakai, objek apapun yang memiliki kemiripan dengan *input* dalam salah satu dari ketiga properti ini akan muncul dalam hasil pencarian.

- `limit`

Kemungkinan nilai: Bilangan bulat positif⁶

Jumlah hasil pencarian maksimal yang ingin ditampilkan dalam keluaran.

Sedangkan, keluaran dari perkakas ini merupakan sebuah objek JSON yang memiliki dua properti utama, yaitu:

- `resultCount`

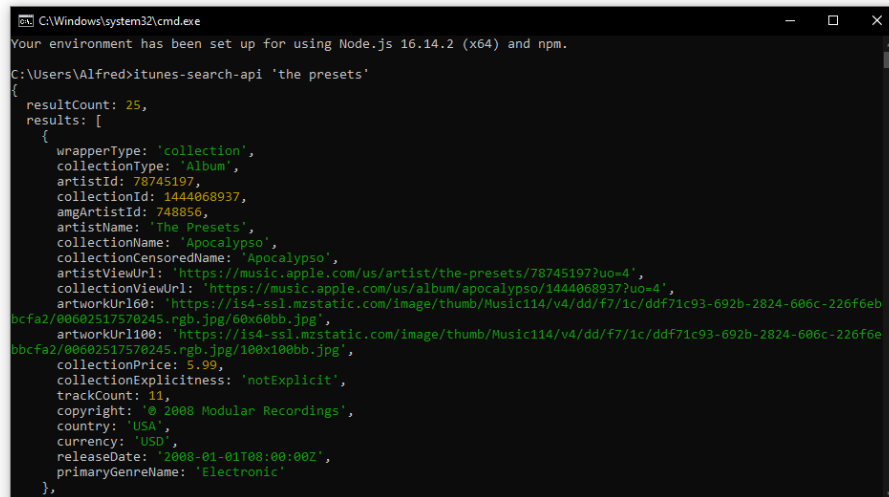
Properti ini berisi bilangan bulat yang menandakan berapa buah objek yang terdapat dalam hasil pencarian.

- `results`

Array yang berisi kumpulan objek yang terdapat di dalam hasil pencarian. Objek-objek ini akan dikembalikan berupa sebuah *array* lain yang berisi seluruh properti dari masing-masing objek. Apa saja properti yang diikuti dalam *array* tersebut tergantung tipe dari objek dalam hasil pencarian.

Adapun contoh penggunaan dan hasil keluaran perkakas ini dapat dilihat di gambar 3.

⁶Opsi ini juga menerima bilangan bulat negatif, tetapi menggunakan sebuah bilangan bulat negatif akan menghilangkan pengaruh opsi ini terhadap hasil keluaran.



```

C:\Windows\system32\cmd.exe
Your environment has been set up for using Node.js 16.14.2 (x64) and npm.

C:\Users\Alfred>itunes-search-api 'the presets'
{
  resultCount: 25,
  results: [
    {
      wrapperType: 'collection',
      collectionType: 'Album',
      artistId: 78745197,
      collectionId: 1444868937,
      amgArtistId: 748856,
      artistName: 'The Presets',
      collectionName: 'Apocalypse',
      collectionCensoredName: 'Apocalypse',
      artistViewUrl: 'https://music.apple.com/us/artist/the-presets/78745197?uo=4',
      collectionViewUrl: 'https://music.apple.com/us/album/apocalypse/1444868937?uo=4',
      artworkUrl60: 'https://is4-ssl.mzstatic.com/image/thumb/Music114/v4/dd/f7/1c/ddf71c93-692b-2824-606c-226f6ebbcfa2/00002517570245.rgb.jpg/60x60bb.jpg',
      artworkUrl100: 'https://is4-ssl.mzstatic.com/image/thumb/Music114/v4/dd/f7/1c/ddf71c93-692b-2824-606c-226f6ebbcfa2/00002517570245.rgb.jpg/100x100bb.jpg',
      collectionPrice: 5.99,
      collectionExplicitness: 'notExplicit',
      trackCount: 11,
      copyright: '© 2008 Modular Recordings',
      country: 'USA',
      currency: 'USD',
      releaseDate: '2008-01-01T08:00:00Z',
      primaryGenreName: 'Electronic'
    }
  ]
}

```

Gambar 3: Contoh penggunaan perangkat *iTunes Search API*. Gambar hanya memuat satu objek untuk menghemat tempat.

3. Mempelajari bahasa pemrograman C serta mempelajari dokumentasi-dokumentasi dari seluruh modul yang dibutuhkan untuk pembuatan perangkat lunak.

Status : Ada sejak rencana kerja skripsi.

Hasil : Ada beberapa modul/fungsi bawaan dari bahasa C yang akan digunakan dalam pembuatan perangkat *command line* ini. Salah satu dari fungsi tersebut adalah fungsi `getopt`

4. Melakukan analisis dan desain perangkat lunak yang akan dibangun.

Status : Ada sejak rencana kerja skripsi.

Hasil :

5. Melakukan analisis kebutuhan fitur-fitur perangkat lunak dan melakukan eksplorasi *library* yang dapat digunakan dan memenuhi spesifikasi dalam pembuatan perangkat lunak.

Status : Ada sejak rencana kerja skripsi.

Hasil :

6. Membangun perangkat lunak berdasarkan rancangan yang sudah dibuat, dengan mengimplementasikan seluruh modul dan *library* yang telah ditentukan di tahap sebelumnya dalam bahasa C.

Status : Ada sejak rencana kerja skripsi.

Hasil : Poin ini akan dikerjakan di Skripsi 2.

7. Melakukan pengujian fungsional dan perbaikan *bug*.

Status : Ada sejak rencana kerja skripsi.

Hasil : Poin ini akan dikerjakan di Skripsi 2.

8. Menulis dokumentasi perangkat lunak.

Status : Ada sejak rencana kerja skripsi.

Hasil : Poin ini akan dikerjakan di Skripsi 2.

9. Menulis dokumen skripsi.

Status : Ada sejak rencana kerja skripsi.

Hasil : Dokumen skripsi akan ditulis hingga bab 3 pada Skripsi 1. Pengisian sisa dari dokumen skripsi, serta penyempurnaan bab 1 sampai bab 3, akan dilakukan pada Skripsi 2.

7 Pencapaian Rencana Kerja

Langkah-langkah kerja yang berhasil diselesaikan dalam Skripsi 1 ini adalah sebagai berikut:

- 1.
- 2.
- 3.

Daftar Referensi

- [1] Marsh, N. (2010) *Introduction to the Command Line: The Fat-Free Guide to Unix and Linux Commands*, 2nd edition. CreateSpace, South Carolina.

Bandung, 20/03/2022

Alfred Aprianto Liaunardi

Menyetujui,

Nama: Pascal Alfadian Nugroho, M.Comp.
Pembimbing Tunggal