

SKRIPSI

PERKAKAS COMMAND LINE KIRI



Alfred Aprianto Liaunardi

NPM: 6181801014

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2022

DAFTAR ISI

DAFTAR ISI	iii
DAFTAR GAMBAR	v
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Tujuan	2
1.4 Batasan Masalah	2
1.5 Metodologi	2
1.6 Sistematika Pembahasan	3
2 LANDASAN TEORI	5
2.1 <i>Command Line</i>	5
2.1.1 <i>Command Line Interface</i> dan <i>Graphical User Interface</i>	5
2.1.2 <i>Command Line</i> di Linux	6
2.1.3 <i>Command Line</i> di Windows	7
2.2 KIRI	10
2.2.1 Tampilan	11
2.2.2 API ¹	12
2.3 Fungsi dan <i>Library</i> Bahasa C	17
2.3.1 getopt [1]	18
2.3.2 libcurl [8]	19
2.3.3 cJSON	20
2.3.4 CMake	20
3 ANALISIS	21
3.1 Analisis Aplikasi Sejenis	21
3.1.1 Chrome Web Store Item Property CLI	21
3.1.2 iTunes Search API	23
3.1.3 Uber CLI	24
3.1.4 Google Maps Direction CLI	26
3.2 Analisis API KIRI	27
3.2.1 Search Place	27
3.2.2 Routing	28
DAFTAR REFERENSI	29
A KODE PROGRAM	31
B HASIL EKSPERIMEN	35

DAFTAR GAMBAR

1.1	Tampilan halaman web KIRI	1
2.1	Dua jenis tampilan perangkat lunak	5
2.2	Baris <i>shell prompt</i> terminal di sistem operasi Linux.	6
2.3	Tampang kedua antarmuka <i>command line</i> bawaan di sistem operasi Windows.	8
2.4	Tampilan awal halaman web KIRI	11
2.5	Tampilan akhir halaman web KIRI	12
2.6	Halaman web <i>API Keys</i> KIRI.	13
2.7	Penggunaan API KIRI untuk layanan pencarian lokasi	14
2.8	Penggunaan API KIRI untuk layanan pencarian rute	17
2.9	Penggunaan API KIRI untuk layanan <i>smart direction</i>	18
3.1	Contoh penggunaan perkakas Chrome <i>Web Store Item Property CLI</i>	22
3.2	Contoh penggunaan perkakas <i>iTunes Search API</i>	24
3.3	Contoh penggunaan perkakas Uber CLI (<i>time</i>)	25
3.4	Contoh penggunaan perkakas Uber CLI (<i>price</i>)	25
3.5	Contoh penggunaan perkakas Google <i>Maps Direction CLI</i>	27
B.1	Hasil 1	35
B.2	Hasil 2	35
B.3	Hasil 3	35
B.4	Hasil 4	35

1

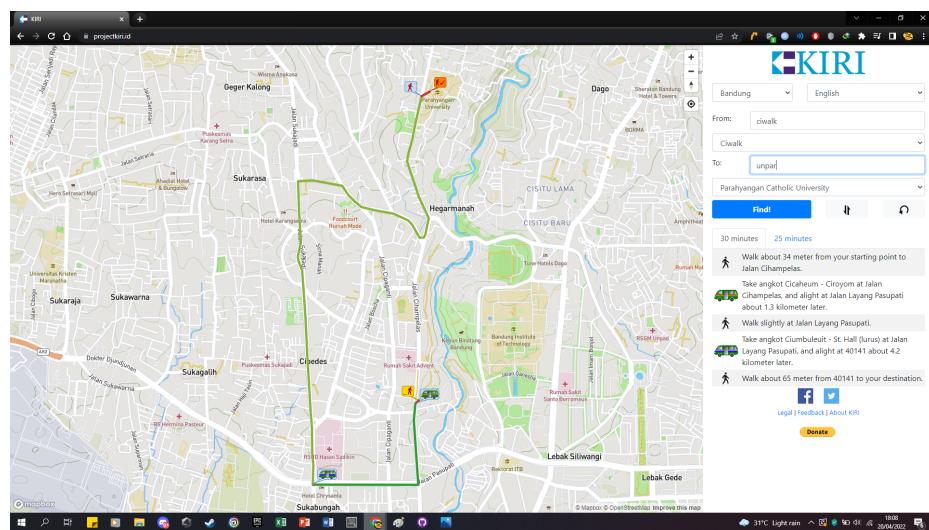
BAB 1

2

PENDAHULUAN

3 1.1 Latar Belakang

- 4 Project KIRI¹ (akan disingkat sebagai KIRI dalam dokumen ini) adalah sebuah perangkat lunak ber-
5 basis web yang dibuat untuk membantu mengurangi efek dari kemacetan. KIRI mengurangi dampak
6 kemacetan dengan membantu penggunanya, baik masyarakat maupun turis, dalam menggunakan
7 salah satu sarana transportasi umum yang ada di Indonesia, yaitu angkutan kota (angkot). Cara
8 KIRI mempermudah penggunaan angkot adalah dengan menunjukkan rute yang akan ditempuh,
9 beserta langkah-langkah yang harus dilakukan oleh pengguna yang ingin berpergian dari satu
10 titik ke titik lain, mulai dari seberapa jauh pengguna harus berjalan untuk menaiki angkot yang
11 bersangkutan, di mana pengguna harus naik atau turun, seberapa jauh lagi pengguna harus berjalan
12 sampai ke titik tujuan, dan seberapa lama estimasi waktu perjalanan yang akan ditempuh. Untuk
13 kebutuhan pembuatan perangkat lunak yang memanfaatkan fitur dari KIRI, tersedia juga REST
14 API KIRI yang dapat digunakan secara praktis. Adapun tampilan dari halaman web ini dapat
15 dilihat di gambar 1.1.



Gambar 1.1: Tampilan halaman web KIRI, yang menunjukkan rute dari Cihampelas Walk ke Universitas Katolik Parahyangan.

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

¹<https://projectkiri.id>

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

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

12 1.2 Rumusan Masalah

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

17 1.3 Tujuan

18 Batasan masalah dalam skripsi ini adalah sebagai berikut:

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

23 1.4 Batasan Masalah

24 Batasan masalah dalam skripsi ini adalah sebagai berikut:

- 25 1. Perangkat lunak dibuat murni dalam bentuk CLI, tanpa tambahan GUI.
26 2. Perangkat lunak yang dibuat tidak menyelesaikan batasan (lokasi tidak terdeteksi, rute tidak
27 berhasil ditemukan, dsb.) yang sudah sejak awal terdapat dalam KIRI.

28 1.5 Metodologi

29 Metodologi yang akan diikuti dalam skripsi ini adalah sebagai berikut:

- 30 1. Melakukan studi dan eksplorasi terhadap fungsi-fungsi yang dimiliki perangkat lunak KIRI
31 serta cara implementasi API KIRI.
32 2. Melakukan analisis dan desain perangkat lunak yang akan dibangun.

²Ubuntu Tutorials - The Linux command line for beginners: 3. Opening a Terminal

- 1 3. Melakukan studi dan eksplorasi terhadap seluruh kemungkinan *library-library* yang memenuhi
2 spesifikasi dalam pembuatan perangkat lunak, berdasarkan analisis dan desain yang telah
3 dilakukan sebelumnya.
- 4 4. Melakukan analisis kebutuhan fitur-fitur perangkat lunak dan melakukan eksplorasi *library*
5 yang dapat digunakan dan memenuhi spesifikasi dalam pembuatan perangkat lunak.
- 6 5. Membangun perangkat lunak berdasarkan rancangan yang sudah dibuat, dengan megimple-
7 mentasikan seluruh modul dan *library* yang telah ditentukan di tahap sebelumnya dalam
8 bahasa C.
- 9 6. Melakukan pengujian fungsional, perbaikan *bug*, serta rekomendasi perbaikan berdasarkan
10 pengujian yang sudah dilakukan.
- 11 7. Menyelesaikan pembuatan dokumen-dokumen yang berkaitan, seperti dokumen skripsi dan
12 dokumentasi perangkat lunak.

13 **1.6 Sistematika Pembahasan**

- 14 Setiap bab dalam skripsi ini memiliki sistematika pembahasan dalam poin-poin sebagai berikut:
- 15 1. Bab 1: Pendahuluan
16 Bab ini berisi latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi peneli-
17 tian, dan sistematika pembahasan.
 - 18 2. Bab 2: Dasar Teori
19 Bab ini berisi pembahasan-pembahasan teoritis mengenai aspek-aspek yang akan dirujuk di
20 dalam skripsi ini, seperti *command line*, bahasa C, dan juga KIRI.
 - 21 3. Bab 3: Analisis dan Perancangan
22 Bab ini berisi pembahasan mengenai rancangan perangkat lunak serta seluruh analisis yang
23 dilakukan terhadap kebutuhan fitur perangkat lunak.
 - 24 4. Bab 4: Implementasi dan Pengujian
25 Bab ini berisi pembahasan mengenai pembuatan perangkat lunak, implementasi seluruh
26 modul-modul yang telah ditentukan di bab 3, serta pengujian fitur-fitur dari perangkat lunak.
 - 27 5. Bab 5: Kesimpulan dan Saran
28 Bab ini berisi kesimpulan hasil pembuatan perangkat lunak dan saran-saran terhadap hasil
29 perangkat lunak yang diberikan selama pengeraaan skripsi.

BAB 2

LANDASAN TEORI

³ 2.1 *Command Line*

⁴ *Command line* (atau *command line interface*) dapat diartikan sebagai tampilan antarmuka/*interface* yang memproses perintah dari pengguna dan meneruskannya langsung ke sistem operasi untuk dijalankan.^[3] Seluruh sistem operasi komputer yang ada memiliki sebuah *command line interface* dalam bentuk *shell*, yang dapat digunakan oleh penggunanya untuk langsung mengakses fungsi atau servis yang disediakan oleh sistem operasi.^[4]

⁹ 2.1.1 *Command Line Interface* dan *Graphical User Interface*

¹⁰ Ada beberapa dari tipe antarmuka yang masih banyak digunakan di zaman sekarang, tetapi dua tipe ¹¹ yang paling banyak muncul adalah *command line interface* dan *graphical user interface*. Perangkat ¹² lunak berbasis *command line* sendiri bisa memiliki berbagai macam tampilan, tetapi semuanya ¹³ selalu mengikuti satu bentuk antarmuka umum. Bentuk yang dimaksud adalah sebuah area/*window* ¹⁴ yang memuat teks berupa perintah-perintah dari user untuk dilakukan oleh komputer, beserta ¹⁵ keluarannya yang juga berupa teks, seperti dapat dilihat pada gambar 2.1a. Jenis perangkat lunak ¹⁶ seperti ini disebut memiliki antarmuka jenis *command line interface* (CLI). Adapun dekorasi visual ¹⁷ yang dimiliki oleh jenis tampilan ini hanya berupa warna pada teks-teks yang ada, tanpa tambahan ¹⁸ gambar apapun. Jika perangkat lunak tersebut memiliki dekorasi dan/atau tombol interaktif berupa ¹⁹ gambar grafis, seperti pada gambar 2.1b, maka perangkat lunak tersebut dikategorikan sebagai ²⁰ perangkat lunak berbasis *graphical user interface*.

```
devasc@labvn: ~/labs/personal/samples
File Edit View Search Terminal Help
devasc@labvn: $ ls -l
total 20
drwxr-xr-x 3 devasc devasc 4096 Dec 14 22:44 Desktop
drwxr-xr-x 2 devasc devasc 4096 Sep 18 2021 Documents
drwxr-xr-x 4 devasc devasc 4096 Sep 18 2021 Downloads
drwxr-xr-x 4 devasc devasc 4096 Dec 14 22:44 Labs
drwxr-xr-x 5 devasc devasc 4096 Jun 17 2020 snap
devasc@labvn: $ cd "labs/personal/samples"
devasc@labvn:~/labs/personal/samples$ ls
txtsample.txt
devasc@labvn:~/labs/personal/samples$ cat txtsample.txt
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Collatio igitur ista te nihil
uvat. Honestia oratio, Socratica, Platonis etiam. Primum in nostrane potestate est, qui
de meminerimus? Duo Reges: constructio interrete. Quid, si etiam lucunda memoria est pra-
eteritorum malorum? Si quidem, inquit, tollerem, sed relinquo. An nisi populari fama?
Quanquam id quidem licet illi existimare, qui legerint. Summum a vobis bonus voluptas
dicitur. At hoc in eo M. Refert tamen, quo modo. Quid sequatur, quid repugnet, vident.
Nam id ipsum absurdum, maximum malum neglegi.
```



(a) Antarmuka perangkat lunak berbasis *command line interface*.

(b) Antarmuka perangkat lunak berbasis *graphical user interface*.

Gambar 2.1: Contoh dua jenis antarmuka (*interface*) perangkat lunak.

Selain dari tampilannya sendiri, ada beberapa perbedaan utama lain antara perangkat-perangkat lunak berbasis *command line interface* dengan perangkat lunak berbasis *graphical user interface*.

Adapun perbedaan-perbedaan utama dari kedua jenis antarmuka ini adalah sebagai berikut.[4]

- Penggunaan sumber daya sistem untuk menjalankan perangkat lunak berbasis *command line interface* lebih rendah dibandingkan dengan perangkat lunak berbasis *graphical user interface*.
- Bagi pengguna pemula (atau pengguna awam pada umumnya), perangkat lunak berbasis *command line interface* akan lebih sulit digunakan karena tidak adanya bantuan apapun dalam bentuk visual, sehingga satu-satunya cara untuk tahu bagaimana cara menggunakan fitur-fiturnya adalah melalui dokumentasi perangkat lunak yang ada. Karena alasan yang sama pula, perangkat lunak berbasis *command line interface* lebih sulit untuk dibiasakan penggunaannya.
- Automasi perintah yang bersifat berulang-ulang jauh lebih mudah dilakukan pada perangkat lunak berbasis *command line interface*. Hal ini dikarenakan perangkat lunak berbasis *command line interface* tidak hanya lebih mudah untuk dibuat *script*-nya, tetapi juga lebih efisien untuk digunakan ketika ada banyak sekali perintah yang harus dilakukan pada suatu saat tertentu.

2.1.2 *Command Line* di Linux

Linux merupakan sebuah sistem operasi yang sangat modular, jadi ada banyak sekali *shell* yang dapat dijalankan dan digunakan di dalamnya. Walaupun begitu, ada satu *shell* yang selalu datang ter-*install* di dalam semua sistem operasi Linux, yaitu “*bash*” (GNU *Bourne Again Shell*).[5]

Tampilan

Ketika terminal di Linux dijalankan, akan keluar kotak dialog, beserta sebuah baris. Baris ini biasanya berisi sebuah teks dengan format sebagai berikut.

```
<nama pengguna>@<nama perangkat>:<direktori yang sedang diproses>$
```

Tanda dolar di ujung baris ini menandakan bahwa baris tersebut merupakan baris *shell prompt*, yang merupakan waktu di mana terminal sudah siap menerima masukan dari pengguna untuk diproses. Perlu diingat bahwa di posisi tanda dolar ini, terkadang justru terdapat tanda pagar (#). Tanda pagar di akhir baris *shell prompt* menandakan bahwa terminal tersebut dijalankan dengan tingkat akses *superuser*, yang berarti bahwa entah pengguna masuk ke sistem sebagai user *root*, atau terminal memiliki izin tingkat *superuser/administrator*.[3]



```
drwx----- 5 devasc devasc 4096 Sep 1
devasc@labvm:~$ |
```

(a) *Shell prompt* terminal dengan tingkat izin normal.



```
drwxr-xr-x 3 root root 4096 Sep 1
root@labvm:~# |
```

(b) *Shell prompt* terminal dengan tingkat izin *superuser*.

Gambar 2.2: Baris *shell prompt* terminal di sistem operasi Linux.

1 Navigasi [3]

2 Sama seperti di Windows, Linux menyimpan file-filenya di sebuah struktur direktori yang bersifat
3 hierarkial. Hal ini berarti bahwa file-file tersebut disimpan dalam direktori-direktori (atau *folder*-
4 *folder*) yang tersusun seperti sebuah pohon. dalam arti bahwa satu *folder* bisa jadi berada di dalam
5 satu *folder* lain, atau berisi beberapa *folder* lainnya.

6 Untuk navigasi, terminal Linux memiliki beberapa perintah utama. Adapun perintah-perintah
7 tersebut adalah sebagai berikut.

- 8 • `pwd`

9 `pwd` merupakan singkatan dari *print working directory*, yang berarti bahwa perintah ini akan
10 mengeluarkan *working directory*, atau direktori tempat terminal sekarang sedang bekerja/ber-
11 jalan, sebagai keluaran dari perintah tersebut. Ketika pengguna pertama kali menjalankan
12 terminal, *working directory*-nya selalu merupakan direktori *home* dari perangkat.

- 13 • `ls`

14 `ls` digunakan untuk menghasilkan keluaran berupa isi dari folder yang dispesifikasi. Biasanya
15 digunakan ketika pengguna sudah memasuki folder yang diinginkan, walaupun dengan perintah
16 ini, pengguna bisa saja mengintip isi dari folder manapun di direktori manapun, dengan
17 mengikutkan direktori yang diinginkan sebagai parameter dari perintah tersebut. Adapun
18 Isi dari folder yang diikutkan sebagai parameter tidak hanya berupa folder lain, tetapi juga
19 seluruh file-file yang ada, walaupun untuk file-file yang disembunyikan (nama file diawali
20 dengan tanda titik), perlu ditambahkan opsi `-a` agar file-file tersebut muncul pula dalam
21 keluarannya.

- 22 • `cd`

23 `cd` adalah perintah yang berfungsi untuk mengganti *working directory* dari terminal. Untuk
24 melakukan hal tersebut, perintah yang perlu dimasukkan adalah sebagai berikut:

25 `cd <direktori yang diinginkan>`

26 Direktori yang diinginkan dapat berupa direktori absolut, atau direktori relatif. Perbedaannya
27 adalah direktori absolut selalu dimulai dari folder *root*, mengikuti folder-folder apapun yang
28 ada di antara *root* sampai ke folder yang diinginkan.

29 Sedangkan, direktori relatif selalu dimulai dari *working directory*. Untuk penggunaan direktori
30 relatif, diperlukan dua buah notasi spesial, yaitu titik `(.)`, yang merepresentasikan *working*
31 *directory* sekarang itu sendiri, dan dua titik `(..)`, yang merepresentasikan *parent folder* dari
32 *working directory*.

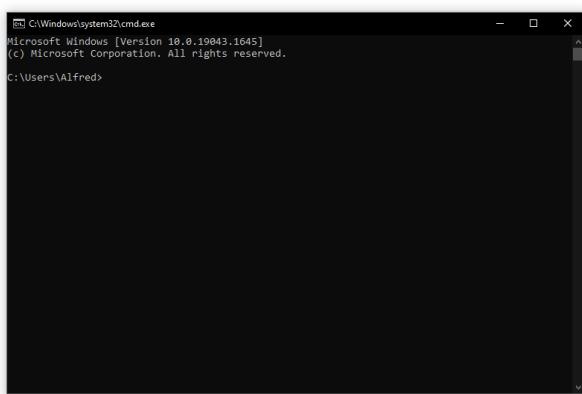
33 2.1.3 Command Line di Windows

34 Cara kerja *command line* di Windows serupa dengan cara kerja *command line* di Linux, dalam
35 arti bahwa untuk bekerja dengan *command line* di Windows, penggunanya juga akan langsung
36 berinteraksi dengan utilitas yang disediakan oleh sistem operasi. *Command line* di Windows juga
37 dapat digunakan untuk hal-hal yang serupa dengan *command line* di Linux, seperti menulis (dan
38 menjalankan) *script*, menjalankan perintah yang diinginkan pengguna secara otomatis, atau melihat
39 status dari sistem operasi.[4]

1 Masih sama dengan Linux, ada banyak sekali command yang bisa digunakan, sehingga susah
 2 untuk menghafal seluruh command-command yang ada—termasuk masukan, parameter-parameter
 3 yang dibutuhkan, serta keluarannya. Untuk melihat dokumentasi, atau penjelasan detail untuk
 4 masukan, keluaran, parameter, serta opsi-opsi dari perintah tertentu, pengguna dapat memasukkan
 5 perintah tersebut, diikuti dengan /?.[4]

6 Tampilan

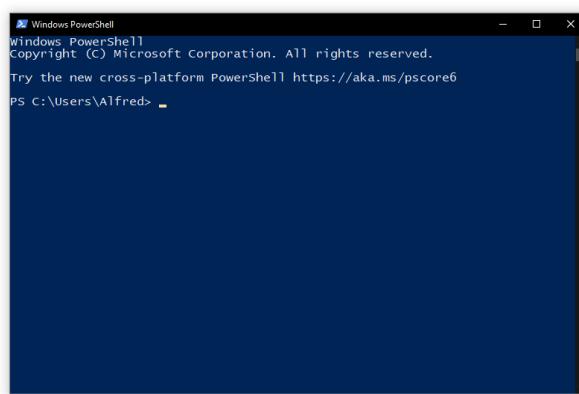
7 Di sistem operasi Windows, ada dua jenis antarmuka *command line*, yaitu cmd (*Command Prompt*)
 8 dan *PowerShell*. Keduanya memiliki tampilan yang kurang lebih sama—hanya saja awalnya cmd
 9 memiliki latar belakang hitam, sedangkan *PowerShell* memiliki latar belakang biru tua, seperti
 10 terlihat di gambar 2.3.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Alfred>
```

(a) Antarmuka Windows *Command Prompt* (cmd)



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Alfred>
```

(b) Antarmuka Winodws *PowerShell*

Gambar 2.3: Tampang kedua antarmuka *command line* bawaan di sistem operasi Windows.

11 Navigasi

12 Untuk navigasi di antarmuka *command line* Windows, ada dua perintah penting yang dipakai ketika
 13 pengguna sedang berurusan dengan file-file dan navigasi dalam direktori sistem. Kedua perintah
 14 tersebut adalah **cd** dan **dir**.

- 15 • **cd (chdir)** [6]

16 **cd** merupakan sebuah perintah yang memiliki tiga fungsi utama, yaitu menampilkan *drive*
 17 tempat sedang *command line* berada (jika pengguna hanya memasukkan **cd** tanpa parameter
 18 apapun), menampilkan direktori tempat *command line* sedang berada (jika pengguna hanya
 19 memasukkan *drive* sebagai parameter, atau fungsi yang paling umumnya, untuk mengganti
 20 *working directory* dari *command line*.

21
 22 Adapun format dari perintah **dir** adalah sebagai berikut.

23 **cd [/d] [<drive>:] [<path>]**

1 Dengan fungsi dari semua opsi dan parameter yang ada sebagai berikut.

2 – /d

3 Opsi yang menandakan bahwa pengguna ingin mengganti *drive* (partisi) dan juga *working*
4 *directory* dari *command line*.

5 – <drive>:

6 Kode huruf dari partisi yang akan diproses.

7 – <path>

8 Direktori yang akan diproses. Parameter ini harus diikutkan beserta kode huruf partisi
9 (tidak dapat berdiri sendiri.)

10 • **dir**

11 **dir** merupakan sebuah perintah yang mengeluarkan/menampilkan sebuah daftar berisi file-file
12 yang ada di suatu direktori, termasuk subdirektori. Jika tidak disertai parameter apapun,
13 perintah ini akan menampilkan label volume dan nomor serial *disk*, dilanjutkan dengan daftar
14 direktori dan file di dalamnya. Untuk file, akan ditampilkan nama beserta ukurannya. Perintah
15 ini juga akan menampilkan jumlah direktori dan file yang didaftarkan, ukuran kumulatifnya,
16 dan sisa dari *disk* yang tidak terpakai (dalam *bytes*).^[7]

17 Adapun format dari perintah **dir** adalah sebagai berikut.^[4]

19 **dir** [<drive:>] [<path>] [<filename>] [/A[[:]<attributes>]] [/B]
20 [/C] [/D] [/L] [/N] [/O[[:]<sortorder>]] [/P] [/Q] [/R] [/S]
21 [/T[[:]<timefield>]] [/W] [/X] [/4]

22 Untuk perintah ini, seperti terlihat di atas, memiliki banyak sekali opsi dan parameter.
23 Tiap-tiap dari parameter tersebut memiliki fungsi tersendiri, yaitu:

24 – /A[[:]<attributes>]

25 Menampilkan file-file dengan atribut tertentu, seperti file yang disembunyikan, file sistem,
26 file *read-only*, dan sebagainya.

27 – /B

28 Menghilangkan *heading* dan ringkasan informasi dari keluaran, atau dengan kata lain,
29 hanya menampilkan file-file dan direktori, tanpa informasi tambahan apapun.

30 – /C

31 Menggunakan separator koma untuk tiap angka ribuan di ukuran file. Jika opsi yang
32 dimasukkan adalah /-C, separator koma justru akan dihilangkan.

33 – /D

34 Menampilkan keluaran dengan format yang lebih lebar. Jika opsi ini diikutkan, keluaran
35 akan ditampilkan dengan urutan berdasarkan kolom.

36 – /L

37 Seluruh teks dalam keluaran akan menggunakan huruf kecil. Jika opsi ini tidak digunakan,
38 keluaran akan mengandung huruf besar dan huruf kecil *mixed case*.

39 – /N

40 Menampilkan daftar dengan format panjang, dengan nama file berada di ujung paling
41 kanan.

- /O[[:]<sortorder>]
Menampilkan daftar direktori yang terurut berdasarkan urutan tertentu, seperti berdasarkan ekstensi file, berdasarkan tanggal dibuat, berdasarkan nama, dan sebagainya. Jika tidak diikutkan tanda minus (-) sebelum huruf O pada perintah, daftar yang muncul akan terurut secara menaik.
- /P
Memberhentikan keluaran selama beberapa waktu singkat (memberi jeda kecil) setelah setiap halaman informasi.
- /Q
Menambahkan informasi mengenai pemilik file dalam keluaran.
- /R
Menampilkan *data stream* alternatif, jika ada.
- /S
Mendaftarkan seluruh file di direktori dan subdirektori yang diproses. Tiap-tiap direktori akan memiliki *header* tersendiri dalam keluarannya.
- /T[[:]<timefield>]
Menspesifikasi *time field* mana yang akan tampil dan digunakan sebagai urutan, jika aturan pengurutan lain tidak ditentukan. *Time field* yang dapat digunakan adalah waktu pembuatan file, kapan terakhir file diakses, dan kapan file terakhir dimodifikasi. Jika parameter ini tidak dispesifikasi, *time field* yang digunakan adalah kapan file terakhir dimodifikasi.
- /W
Menampilkan keluaran dengan format yang lebih lebar. Opsi ini hampir sama dengan /D, hanya saja untuk /W, jika opsi ini diikutkan, keluaran akan ditampilkan dengan urutan berdasarkan baris, dan bukan kolom.
- /X
Menampilkan nama pendek yang dibuat untuk nama-nama file non-8.3. Opsi ini memiliki format tampilan yang sama seperti opsi /N, hanya saja nama pendeknya ditampilkan di keluaran sebelum nama panjangnya.
- 4
Menampilkan angka tahun dengan format angka empat digit.

32 2.2 KIRI

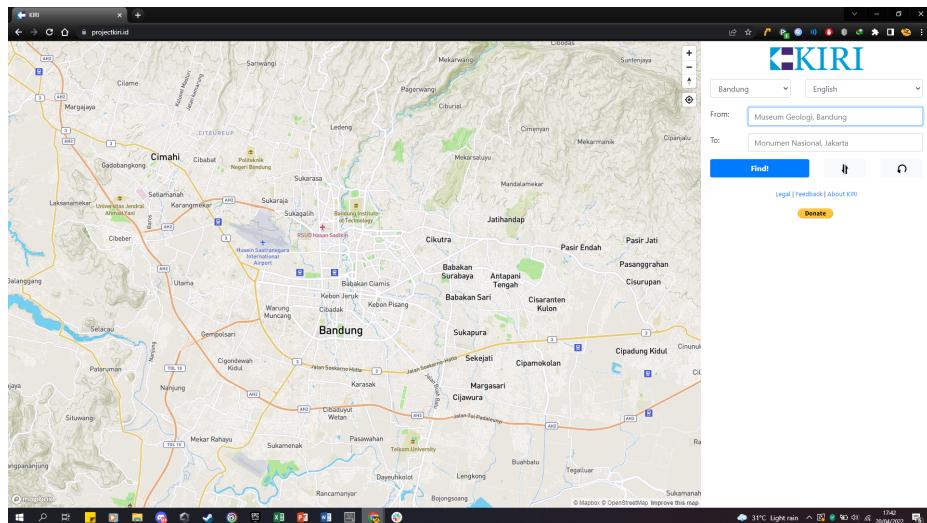
33 KIRI merupakan sebuah perangkat lunak berbasis web yang berfungsi untuk menyelesaikan (atau
34 setidaknya mengurangi) dampak dari masalah-masalah yang dapat diselesaikan oleh transportasi
35 umum/publik di Indonesia, seperti pemanasan global, kemacetan, atau peningkatan harga bensin.
36 Selain itu, turis mancanegara juga memilih untuk menaiki transportasi umum, karena jenis sarana
37 transportasi tersebut tidak hanya jauh lebih murah, tetapi juga memberikan kesempatan yang
38 mudah kepada mereka untuk melihat seluk-beluk dari kota-kota yang mereka kunjungi. Walaupun
39 begitu, banyak masyarakat lokal sendiri yang seringkali masih segan untuk menaiki transportasi
40 publik, umumnya karena transportasi publik dianggap lebih rumit persiapannya dibandingkan
41 dengan metode-metode transportasi privat, seperti menaiki kendaraan pribadi.¹

¹<https://projectkiri.github.io/#about-kiri>

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.

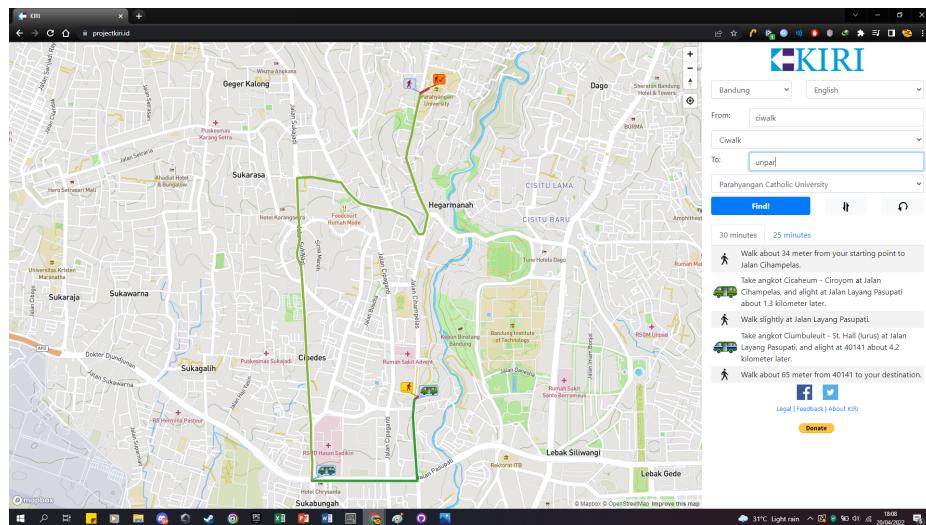
2.2.1 Tampilan

Pada saat pertama kali dibuka, hal pertama yang paling mencolok di halaman awal web KIRI adalah sebuah peta besar di sebelah kiri yang dapat diperbesar ataupun diperkecil. Sedangkan, bagian kanan dari halamannya terdiri atas beberapa bagian. Di bagian paling atas terdapat logo KIRI, beserta sepasang menu *dropdown*—yang pertama merupakan pilihan kota tempat pengguna berada (untuk sekarang hanya tersedia pilihan kota Jakarta dan Bandung), dan yang kedua merupakan pilihan bahasa, entah bahasa Indonesia atau Inggris. Di bawahnya merupakan sepasang menu *dropdown* yang merupakan tempat di mana pengguna memasukkan lokasi awal dan tujuan yang akan diproses oleh KIRI. Terakhir, di bawahnya ada sebuah bagian kosong, yang nantinya akan menjadi tempat di mana KIRI akan meletakkan keluaran dari prosesnya. Adapun tampilan awal dari halaman web ini dapat dilihat di gambar 2.4.



Gambar 2.4: Tampilan awal halaman web KIRI.

Ada dua area yang memiliki perbedaan yang signifikan ketika pengguna sudah memasukkan masukan dan menyuruh KIRI untuk memprosesnya. Bagian yang pertama adalah bagian peta, yang setelah pemrosesan masukan, akan memiliki garis-garis berwarna yang menandakan rute angkot maupun tujuan perjalanan kaki yang harus ditempuh oleh pengguna. Bagian kedua adalah bagian keluaran, yang tadinya kosong, sekarang akan berisi langkah-langkah yang harus ditempuh oleh penggunanya untuk pergi dari lokasi awal ke lokasi tujuan. Spesifiknya, perbedaan-perbedaan ini dapat dilihat di gambar 2.5.



Gambar 2.5: Tampilan halaman web KIRI setelah pemrosesan masukan dari pengguna selesai.

2.2.2 API²

KIRI juga memiliki sebuah API yang dapat digunakan untuk keperluan pengembangan perangkat lunak. API ini menyediakan tiga buah jenis layanan web (*webservice*), yang ketiganya dapat dilakukan dengan mengirim permintaan (*request*) tipe GET melalui API tersebut. Isi dari permintaan yang perlu dikirimkan serta respon dari API yang akan dikembalikan berbeda tergantung dari jenis layanan yang digunakan. Adapun ketiga jenis layanan tersebut adalah pencarian tempat (*search place*), pencarian rute (*routing*), dan *smart direction*.

Search Place

Layanan pencarian lokasi (*search place*) adalah layanan web pada API KIRI yang berfungsi untuk mencari suatu lokasi berdasarkan kata kunci yang diberikan oleh pengguna. Untuk menggunakan layanan ini, pengguna harus mengirim permintaan GET ke alamat <https://projectkiri.id/api>. Adapun permintaan tersebut harus memiliki parameter-parameter seperti terlihat di bawah ini.

- **version**

Kemungkinan nilai: 2

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

- **mode**

Kemungkinan nilai: searchplace

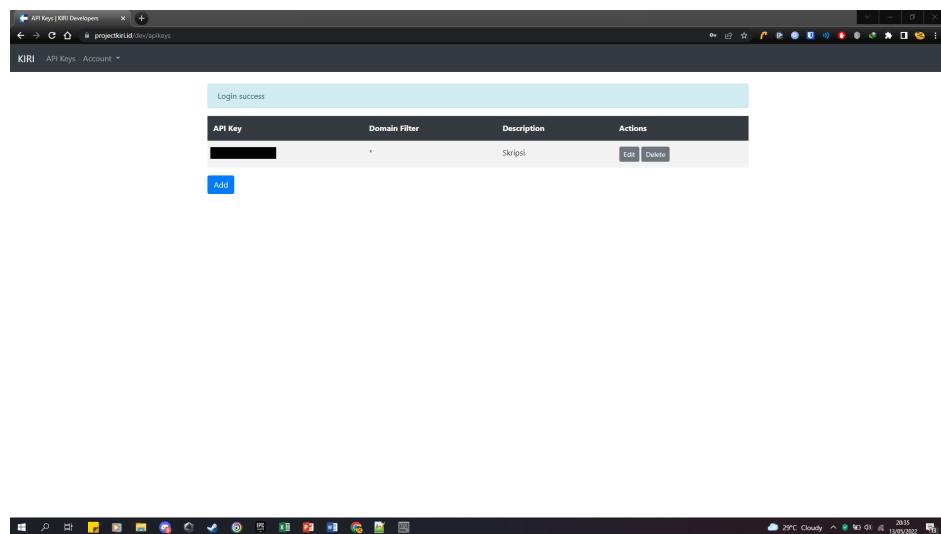
Parameter ini merupakan mode dari servis/jasa API yang akan digunakan oleh pengguna. Untuk penggunaan layanan pencarian lokasi, variabel ini harus diisi dengan **searchplace**.

- **region**

Kemungkinan nilai: cgk, bdo, mlg, atau sub

Parameter ini merupakan kode bandara IATA tiga huruf yang merepresentasikan daerah mana tempat lokasi yang ingin dicari berada. Kode yang dapat diproses oleh API ini meliputi **cgk** (Cengkareng/Jakarta), **bdo** (Bandung), **mlg** (Malang), dan **sub** (Surabaya).

²<https://github.com/projectkiri/Tirtayasa/wiki/KIRI-API-v2>



Gambar 2.6: Halaman web *API Keys* KIRI.

1 • **querystring**

2 **Kemungkinan nilai:** *string* apapun dengan panjang minimal satu karakter

3 Parameter ini berisi kata kunci yang akan digunakan untuk menentukan lokasi yang ingin
4 dicari pengguna.

5 • **apikey**

6 **Kemungkinan nilai:** angka heksadesimal 16-digit

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

9 Perlu diperhatikan bahwa salah satu dari parameter yang harus diikutkan dalam pesan tersebut
10 merupakan parameter yang meminta kunci API. Kunci tersebut harus digenerasikan terlebih dahulu
11 sebelum API KIRI dapat digunakan, melalui halaman *API Keys* KIRI,³ yang dapat dilihat di
12 gambar 2.6.

13 Untuk mengakses halaman tersebut, pengguna harus membuat sebuah akun terlebih dahulu.
14 Ketika akun sudah dibuat, maka pengguna baru akan dapat membuat kunci API yang dibutuhkan,
15 sekaligus membuat filter *domain*, yang membatasi di *domain* mana saja kunci tersebut dapat
16 digunakan, serta memberikan deskripsi untuk kunci API tersebut. Kunci ini kemudian dapat
17 digunakan sebagai nilai dari parameter **apikey** yang diperlukan dalam permintaan tadi.

18 Sebelum membahas keluaran dari layanan API ini, perlu ditegaskan dulu apa definisi dari nilai
19 *latitude* dan *longitude* suatu lokasi. *Latitude* merupakan berapa derajat sebuah tempat berada dari
20 garis ekuator, dengan lokasi-lokasi yang berada maksimum 90 derajat di atas ekuator memiliki
21 nilai *latitude* positif, sedangkan lokasi-lokasi yang berada maksimum 90 derajat di bawah ekuator
22 memiliki nilai *latitude* negatif. Sedangkan, *longitude* merupakan berapa derajat lokasi sebuah
23 tempat berada dari garis meridian (bujur) utama Bumi, dengan rentang nilai dari -180 derajat di
24 sisi kiri (barat) meridian utama, hingga 180 derajat di kanan (timur) bujur tersebut.⁴ Kedua nilai
25 ini merupakan salah satu dari dua variabel yang dikembalikan dalam respon API untuk layanan ini,

³<https://projectkiri.id/dev/apikeys>

⁴https://gsp.humboldt.edu/olm/Lessons/GIS/01%20SphericalCoordinates/Latitude_and_Longitude.html



Gambar 2.7: Penggunaan API KIRI untuk layanan pencarian lokasi menggunakan Postman. Gambar ini menunjukkan hasil pencarian lokasi “unpar” di daerah Bandung.

1 dengan variabel lainnya berupa nama dari lokasi yang ditemukan itu sendiri. Adapun respon yang
2 diberikan oleh API akan berupa sebuah objek JSON yang selalu memiliki setidaknya dua variabel,
3 yaitu:

- 4 • **status**

5 **Kemungkinan nilai:** `ok` atau `error`

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

- 8 • **searchresult**

9 Variabel ini merupakan hasil pencarian yang ditemukan oleh layanan API ini. Isi dari variabel
10 ini merupakan objek *array* yang masing-masing memiliki variabel berikut:

- 11 – **placename**

12 Variabel ini berisi nama lokasi yang ditemukan berdasarkan kata kunci yang diberikan
13 oleh pengguna.

- 14 – **location**

15 Variabel ini berisi nilai *latitude* dan *longitude* dari lokasi yang ditemukan dalam pencarian.

16 Contoh dari penggunaan API KIRI untuk layanan ini dapat dilihat di gambar 2.7.

17 ***Routing***

18 Layanan pencarian rute (*routing*) adalah layanan web pada API KIRI yang memiliki fungsi yang
19 sama dengan fungsi utama dari perangkat lunak KIRI sendiri, yaitu menunjukkan rute serta
20 langkah-langkah yang harus ditempuh untuk pergi dari satu lokasi ke lokasi lainnya, dengan
21 menggunakan angkot yang tersedia. Untuk menggunakan layanan ini, pengguna harus mengirim
22 permintaan GET ke alamat <https://projectkiri.id/api>. Adapun permintaan tersebut harus memiliki
23 parameter-parameter seperti terlihat di bawah ini.

- 1 • **version**

2 **Kemungkinan nilai:** 2

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

- 4 • **mode**

5 **Kemungkinan nilai:** **findroute**

6 Parameter ini merupakan mode dari servis/jasa API yang akan digunakan oleh pengguna.

7 Untuk penggunaan layanan pencarian rute dengan angkot, variabel ini diisi dengan **findroute**.

- 8 • **locale**

9 **Kemungkinan nilai:** **en** atau **id**

10 Parameter ini mengatur bahasa apa yang akan digunakan dalam keluaran API nantinya—**en**

11 berarti keluaran akan menggunakan bahasa Inggris, dan **id** berarti keluaran akan menggunakan

12 bahasa Indonesia.

- 13 • **start**

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

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

- 16 • **finish**

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

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

- 19 • **presentation** (opsional)

20 **Kemungkinan nilai:** **desktop**

21 Parameter ini hanya digunakan untuk fitur *backwards compatibility*.

- 22 • **apikey**

23 **Kemungkinan nilai:** angka heksadesimal 16-digit

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

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

- 28 • **status**

29 **Kemungkinan nilai:** **ok** atau **error**

30 Variabel ini manandakan apakah permintaan berhasil diproses atau tidak. Jika permintaan
31 berhasil diproses, variabel ini akan bernilai **ok**, dan jika tidak, variabel ini akan bernilai **error**.

- 32 • **message**

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

- 41 – **steps**

1 **Tipe:** *array*

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

- 4 * Tipe transportasi

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

- 8 * Kode angkot

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

- 12 * *Array latitude dan longitude* lokasi

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

- 15 * Deskripsi langkah

16 Deskripsi langkah yang harus ditempuh, dalam bahasa natural. Bahasa yang digu-
17 nakan tergantung parameter **locale** yang diatur dalam masukan.

- 18 * URL untuk mendapatkan tiket kendaraan

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

- 21 * URL editor rute

22 Tautan untuk melakukan modifikasi rute, jika dimungkinkan. Jika rute tidak bisa
23 dimodifikasi, variabel ini akan berisi **null**.

24 – **traveltime**

25 **Tipe:** *string*

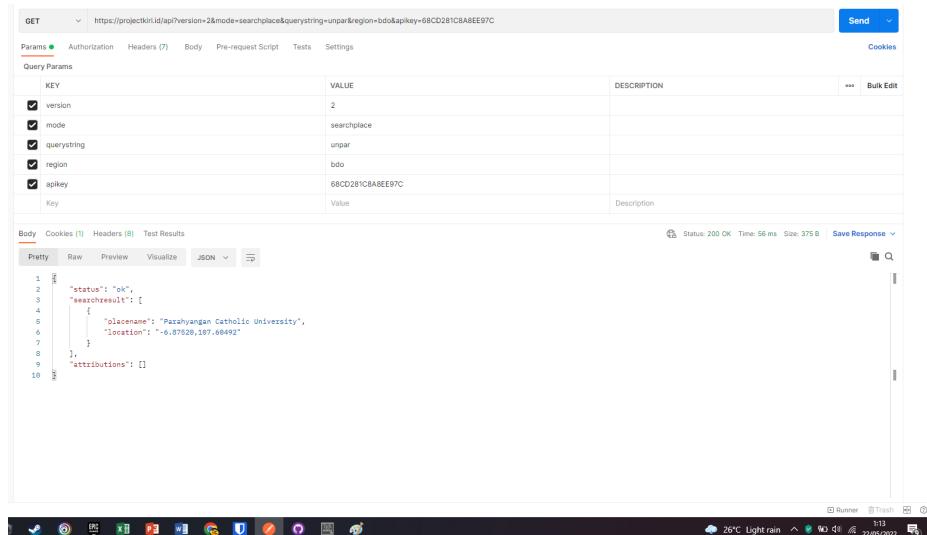
26 Variabel ini berisi estimasi jangka waktu yang diperlukan untuk menyelesaikan langkah
27 tersebut.

28 Adapun gambar 2.8 menunjukkan penggunaan API KIRI untuk layanan pencarian rute dari
29 Cihampelas Walk ke Universitas Katolik Parahyangan.

30 **Smart Direction**

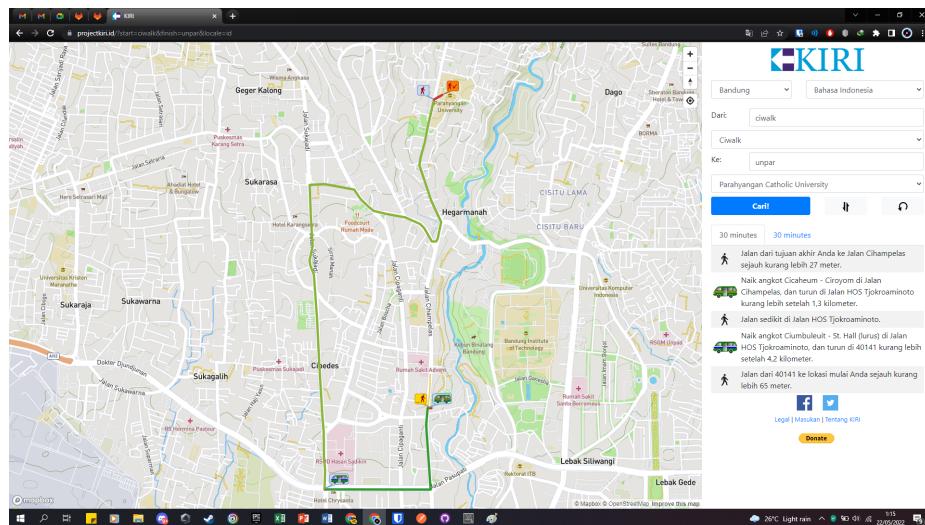
31 Layanan terakhir dari API ini adalah layanan *smart direction*, yang merupakan gabungan dari
32 kedua layanan sebelumnya. Berbeda dengan kedua layanan tadi, yang harus mengakses API secara
33 manual (dengan mengirimkan permintaan GET), layanan ini tidak memerlukan pengguna untuk
34 mengirim permintaan tersebut secara manual—layanan ini sudah otomatis mengirimkan permintaan
35 tersebut. Berbeda dengan kedua layanan sebelumnya pula, layanan ini tidak memerlukan dibuatnya
36 kunci API terlebih dahulu.

37 Layanan ini bekerja dengan menyuruh peramban pengguna untuk langsung mengakses halaman
38 web KIRI yang sudah langsung menunjukkan rute yang perlu ditempuh untuk pergi dari lokasi satu
39 ke lokasi lainnya. Untuk melakukan hal ini, layanan ini memerlukan sebuah URL, yang memiliki
40 format sebagai berikut:



Gambar 2.8: Penggunaan API KIRI untuk layanan pencarian rute menggunakan Postman. Gambar ini menunjukkan hasil pencarian rute dari Cihampelas Walk ke UNPAR.

- 1 <https://projectkiri.id?start=<lokasi awal>&finish=<lokasi akhir>&locale=<locale>>
 - 2 Dapat dilihat bahwa URL tersebut memiliki tiga buah parameter, yaitu:
 - 3 • **start**
Kemungkinan nilai: Nilai *latitude* dan *longitude* lokasi, atau nama lokasi tersebut
Parameter ini berisi lokasi yang ingin digunakan sebagai lokasi mulainya pencarian rute.
 - 4 • **finish**
Kemungkinan nilai: Nilai *latitude* dan *longitude* lokasi, atau nama lokasi tersebut
Parameter ini berisi lokasi yang merupakan tujuan akhir yang ingin dicapai dalam pencarian rute.
 - 5 • **locale** (opsional)
Kemungkinan nilai: id atau en
Menentukan dalam bahasa apa hasil pencarian rutenya akan ditampilkan (bahasa Indonesia atau bahasa Inggris). Jika parameter ini tidak diberikan oleh pengguna, maka bahasa yang akan digunakan adalah bahasa yang terakhir dipakai di halaman web KIRI sendiri.
 - 6 Misalkan pengguna ingin mencari rute dari Cihampelas Walk ke Universitas Katolik Parahyangan, dan menampilkan langkah-langkah yang harus ditempuh dalam rutenya dalam bahasa Indonesia.
 - 7 Pengguna dapat memasukkan URL berikut ke peramban mereka.
 - 8 <https://projectkiri.id?start=ciwalk&finish=unpar&locale=id>
 - 9 Jika URL tersebut sudah dimasukkan ke kotak *link* pada peramban, halaman yang akan ditampilkan akan terlihat seperti pada gambar 2.9.
- ## 21 2.3 Fungsi dan *Library* Bahasa C
- 22 Di bagian ini akan dilakukan analisis terhadap seluruh fungsi bawaan, serta *library-library* bahasa pemrograman C yang akan digunakan dalam pebuatan perkakas ini.



Gambar 2.9: Penggunaan API KIRI untuk layanan *smart direction*, dari Cihampelas Walk ke UNPAR.

1 2.3.1 getopt [1]

2 **getopt** merupakan sebuah fungsi yang dapat mengautomasi pekerjaan-pekerjaan yang berhubungan
3 dengan penerimaan opsi-opsi untuk *command line* berbasis UNIX.

4

5 Fungsi **getopt** dapat dipanggil dengan format sebagai berikut.

6 **getopt (argc, argv, <options>)**

7 Seluruh kode ini dapat dimasukkan ke suatu variabel berupa sebuah karakter yang merepre-
8 sentasikan opsi yang ingin digunakan. **argc** merupakan jumlah argumen yang terdapat dalam
9 masukan, sedangkan **argv** merupakan sebuah *array* yang berisi argumen-argumen tersebut.

10
11 Selain itu, penggunaan **getopt** juga memerlukan penggunaan variabel-variabel tertentu, yang
12 dapat dilihat di daftar berikut.⁵

- 13 • **opterr**

14 Isi dari variabel ini akan memberi signal ke perangkat lunak/perkakas yang menentukan
15 apakah **getopt** akan mengirim pesan ke *error stream* atau tidak. Jika variabel ini bukan
16 bernilai 0, maka pesan *error* akan dikirim. Sebaliknya, jika variabel ini bernilai 0, **getopt**
17 tidak akan mengirim pesan *error* apapun, tetapi tetap akan mengembalikan sebuah karakter
18 tanda tanya (?) sebagai tanda bahwa sebuah *error* telah terjadi.

- 19 • **optopt**

20 Ketika **getopt** menemukan sebuah karakter yang tidak didefinisikan dalam kumpulan opsi,
21 atau sebuah opsi yang tidak disertai argumen yang diperlukan, karakter tersebut akan disimpan
22 di variabel ini.

- 23 • **optind**

24 Variabel ini digunakan oleh **getopt** sebagai indeks untuk *array* **argv**. Jika seluruh argumen

⁵https://www.gnu.org/software/libc/manual/html_node/Using-Getopt.html

1 sudah diproses, nilai variabel ini dapat digunakan untuk menentukan argumen mana yang
2 merupakan arguman tambahan yang tidak terpakai. Nilai dari variabel ini dimulai dari 1.

3 • **optarg**

4 Jika opsi yang sedang diproses memerlukan argumen, variabel ini adalah tempat dimana
5 argumen tersebut akan disimpan.

6 • **<options>**

7 Variabel ini berupa *string* yang menandakan karakter-karakter apa saja yang menjadi opsi
8 yang mungkin dalam perkakas tersebut, beserta tipenya. Jika karakter opsi:

- 9 – Diikuti dengan titik dua (:), maka opsi tersebut memiliki argumen yang bersifat wajib.
- 10 – Diikuti dengan titik dua ganda (::), maka opsi tersebut memiliki argumen yang bersifat
11 opsional.
- 12 – Tidak diikuti apa-apa, maka opsi tersebut merupakan opsi tidak berarguman.

13 **getopt-long**

14 Ada pula versi **getopt** yang memungkinkan perangkat lunak untuk menerima dua jenis opsi—opsi
15 versi pendek berupa sebuah karakter singular, seperti pada **getopt** biasa, dan/atau opsi panjang
16 bergaya GNU, berupa sebuah kata.

17 **getopt-long** juga memiliki seluruh variabel-variabel yang dimiliki oleh **getopt**, hanya saja
18 **getopt-long** memiliki sebuah variabel tambahan berupa struktur, yaitu **long_options**. Variabel
19 ini merupakan sebuah struktur berupa *array* yang berisi beberapa *array* lainnya, di mana *array-array*
20 lain in merupakan masing-masing opsi dari fungsi **getopt-long** tersebut. Tiap-tiap *array* tersebut
21 memiliki variabel-variabel berikut:

22 • **name**

23 Variabel ini merupakan nama panjang dari opsi.

24 • **has_arg**

25 Variabel ini merupakan penanda apakah opsi memerlukan argumen atau tidak. Nilai yang
26 mungkin dalam variabel ini adalah **no_argument**, **required_argument**, atau **optional_argument**.

27 • **flag & val**

28 Kedua variabel ini menandakan bagaimana sebuah opsi akan diberlakukan ketika diterima oleh
29 **getopt-long**. Variabel **flag** dapat diisi dengan penunjuk ke suatu variabel lain yang akan diisi
30 dengan isi dari variabel **val** untuk menandakan bahwa **getopt-long** telah berhasil memroses
31 opsi tersebut. Di lain sisi, jika variabel ini berisi *null pointer*, maka fungsi **getopt-long** akan
32 mengembalikan isi dari variabel **val**.

33 Struktur ini harus diakhiri dengan sebuah *array* tambahan yang seluruh variabelnya bernilai 0.

34 **2.3.2 libcurl [8]**

35 libcurl merupakan sebuah *library* yang berisi fungsi-fungsi yang disediakan dalam bentuk API bahasa
36 C, untuk digunakan oleh aplikasi-aplikasi bahasa C. Libcurl didesain dengan berorientasi transfer
37 (biasanya transfer berkas), tanpa memerlukan para pengguna untuk mengerti protokol-protokol
38 yang digunakan dalam proses pentransferan tersebut. Fitur transfer ini dapat dibuat sesederhana
39 mungkin, dan seluruh aturan dan opsi seputar pemindahan berkas tersebut dapat diatur nantinya
40 secara manual melalui opsi-opsi yang ada.

Untuk mulai melakukan transfer berkas, diperlukan juga sebuah “*easy handle*” yang dapat digunakan untuk satu orang individu/pihak, dengan memanggil fungsi `curl_easy_init()`. Setelah itu, untuk mengatur opsi-opsi yang perlu diatur sesuai kebutuhan pengguna, seperti URL yang dituju, protokol yang ingin dipakai, koneksi ke port spesifik, dan sebagainya,⁶ pengguna harus mengaturnya dengan fungsi `curl_easy_setopt()`. *Handle* yang telah diatur ini dapat digunakan berulang kali dengan konfigurasi yang sama, sampai entah pengguna mengganti konfigurasi opsi-opsinya kembali, atau atau *handle*-nya direset dengan pemanggilan fungsi `curl_easy_reset()`.

2.3.3 cJSON

2.3.4 CMake

⁶https://curl.se/libcurl/c/curl_easy_setopt.html

1

BAB 3

2

ANALISIS

3 3.1 Analisis Aplikasi Sejenis

4 Untuk pembuatan perangkat lunak dalam skripsi ini, ada empat buah perkakas *command line* yang
5 akan diamati sebagai aplikasi sejenis. Dua dari empat aplikasi pertama adalah Chrome *Web Store*
6 *Item Property CLI* dan *iTunes Search API*.

7 3.1.1 Chrome *Web Store Item Property CLI*¹

8 Perkakas *command line* ini merupakan ekstensi dari sebuah aplikasi lain yang memiliki fungsi
9 yang sama, yaitu Chrome *Web Store Item Property*.² Perangkat lunak Chrome *Web Store Item*
10 *Property CLI* ini merupakan perangkat lunak yang akan memanggil fungsi API untuk mendapatkan
11 metadata dari sebuah ekstensi pada *web store* peramban Google Chrome. Perbedaan dari perkakas
12 ini dengan aplikasi dasarnya adalah bahwa perkakas ini dapat digunakan sebagai perkakas *command*
13 *line*, sedangkan aplikasi dasarnya hanya bisa digunakan dalam perangkat lunak lainnya sebagai
14 pemanggil fungsi API.

15 Penggunaan

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

18 `chrome-web-store-item-property <identifier>`

19 Dengan **identifier** berupa ID dari ekstensi yang diinginkan. Jadi, misalkan pengguna mema-
20 sukkan `gighmmiobklfepjocnamgkkbiglidom` sebagai ID yang akan digunakan sebagai *identifier*,
21 maka perkakas ini akan mengembalikan metadata dari ekstensi “AdBlock” sebagai keluarannya.
22 Contoh penggunaan perkakas ini dapat dilihat di gambar 3.1.

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

25 • **name**

26 Nama dari ekstensi yang dicari metadatanya.

27 • **url**

28 URL halaman web dari ekstensi yang dicari di *web store* Google Chrome.

¹<https://github.com/pandawing/node-chrome-web-store-item-property-cli>

²<https://github.com/pandawing/node-chrome-web-store-item-property>

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

C:\Users\Alfred>chrome-web-store-item-property gighmmpioblk1fepjocnamgkkbiglidom
{"name":"AdBlock _ best ad blocker","url":"https://chrome.google.com/webstore/detail/adblock-%E2%80%94-best-ad-blocker/gighmmpioblk1fepjocnamgkkbiglidom","image":"https://lh3.googleusercontent.com/mgNKV-3VMD556WVuIwSpukQON-iI4Zlqq03ef1jG2B5j9VP7fxr3idIQ_G0JFD/E6041MwvIQ1leDn_80dFLt5VQ=w128-h128-e365-rj-sc0x0xffffffff","version":"4.46.1","price":0,"priceCurrency":"USD","interactionCount":{"UserDownloads":10000000}, "operatingSystem":"Chrome","id":"gighmmpioblk1fepjocnamgkkbiglidom"}
```

Gambar 3.1: Contoh penggunaan perkakas Chrome *Web Store Item Property* CLI.

- **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.

¹ 3.1.2 iTunes Search API³

² Perkakas *command line* ini berfungsi untuk melakukan pencarian melalui API iTunes, sehingga
³ seakan-akan pengguna langsung melakukan pencarian di iTunes sendiri. Hasil pencarian yang
⁴ dilakukan termasuk judul lagu, nama artis, ataupun nama album, dan pengguna dapat memilih
⁵ secara spesifik objek apa yang ingin dicari.

⁶ Penggunaan

⁷ Perkakas ini dapat digunakan melalui *command prompt* dengan cara mengetikkan perintah sebagai
⁸ berikut.

⁹ `itunes-search-api <input> [<options>]`

¹⁰ Dengan **input** berupa nama dari objek yang dicari. Perkakas ini juga memiliki opsi yang masing-
¹¹ masing memiliki parameter tersendiri untuk mempersempit hasil pencarian. Adapun opsi-opsi
¹² tersebut dapat dilihat di daftar di bawah ini.

- ¹³ • **country**

¹⁴ **Kemungkinan nilai:** Kode negara dua huruf

¹⁵ Opsi ini menerima parameter berupa kode negara asal dari album atau artis yang dicari.

- ¹⁶ • **entity**

¹⁷ **Kemungkinan nilai:** song, musicArtist, atau album

¹⁸ Menandakan jenis objek/entitas yang ingin dicari. Opsi ini dapat bernilai **song** untuk pencarian
¹⁹ berbasis judul lagu, **musicArtist** untuk pencarian nama artis, atau **album** untuk pencarian
²⁰ nama album. Jika opsi ini tidak dipakai, objek apapun yang memiliki kemiripan dengan
²¹ **input** dalam salah satu dari ketiga properti ini akan muncul dalam hasil pencarian.

- ²² • **limit**

²³ **Kemungkinan nilai:** Bilangan bulat positif⁴

²⁴ Jumlah hasil pencarian maksimal yang ingin ditampilkan dalam keluaran.

²⁵ Sedangkan, keluaran dari perkakas ini merupakan sebuah objek JSON yang memiliki dua properti
²⁶ utama, yaitu:

- ²⁷ • **resultCount**

²⁸ Properti ini berisi bilangan bulat yang menandakan berapa buah objek yang terdapat dalam
²⁹ hasil pencarian.

- ³⁰ • **results**

³¹ *Array* yang berisi kumpulan objek yang terdapat di dalam hasil pencarian. Objek-objek ini
³² akan dikembalikan berupa sebuah *array* lain yang berisi seluruh properti dari masing-masing
³³ objek. Apa saja properti yang diikutkan dalam *array* tersebut tergantung tipe dari objek
³⁴ dalam hasil pencarian.

³⁵ Adapun contoh penggunaan dan hasil keluaran perkakas ini dapat dilihat di gambar 3.2.

³<https://github.com/awcross/itunes-search-api>

⁴Opsi 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: 1444068937,
      amgArtistId: 748856,
      artistName: 'The Presets',
      collectionName: 'Apocalypso',
      collectionCensoredName: 'Apocalypso',
      artistViewUrl: 'https://music.apple.com/us/artist/the-presets/78745197?uo=4',
      collectionViewUrl: 'https://music.apple.com/us/album/apocalypso/1444068937?uo=4',
      artworkUrl60: 'https://is4-ssl.mzstatic.com/image/thumb/Music14/v4/dd/f7/1c/ddf71c93-692b-2824-606c-226f6eb0cfa/00602517570245.rgb.jpg/60x60bb.jpg',
      artworkUrl100: 'https://is4-ssl.mzstatic.com/image/thumb/Music14/v4/dd/f7/1c/ddf71c93-692b-2824-606c-226f6eb0cfa/00602517570245.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 3.2: Contoh penggunaan perkakas *iTunes Search API*. Gambar hanya memuat satu objek untuk menghemat tempat.

- 1 Selain dua perkakas *command line* tadi, ada dua perkakas lainnya yang bisa digunakan sebagai
- 2 referensi, tetapi tidak dapat dieksplorasi, dikarenakan kedua aplikasi tersebut tidak berhasil
- 3 dijalankan dengan sempurna. Adapun perkakas-perkakas tersebut adalah sebagai berikut.

4 3.1.3 Uber CLI⁵

- 5 Uber CLI merupakan sebuah perkakas *command line* yang dapat digunakan untuk dua fungsi
- 6 utama. Fungsi pertama dari perkakas ini adalah untuk mendapatkan estimasi untuk seberapa lama
- 7 waktu yang diperlukan untuk servis taksi *online* dari Uber untuk mencapai lokasi yang ingin dituju,
- 8 sedangkan fungsi keduanya adalah untuk mengestimasi berapa harga yang harus dibayarkan untuk
- 9 memakai servis tersebut.

- 10
- 11 Fungsi yang pertama dapat dilakukan memanggil perintah dengan format sebagai berikut.

12 **uber time <alamat>**

- 13 **uber** merupakan perintah dasar dari perkakas ini. **time** merupakan parameter yang menandakan
- 14 bahwa pengguna ingin menggunakan servis prediksi waktu dari perkakas ini. Selain itu, pengguna
- 15 harus memasukkan alamat yang ingin dituju sebagai parameter akhir dari perintah yang akan
- 16 digunakan sebagai masukan. Jika sintaksnya sudah benar, perintah tersebut akan bisa diproses oleh
- 17 perkakas dengan cara mengirimkan pesan hasil konversi perintah tersebut ke API Uber. Setelah
- 18 pemrosesan pesan tersebut berhasil, perkakas ini akan menampilkan sebuah keluaran dengan format
- 19 yang dapat dilihat di gambar 3.3. Perlu diperhatikan juga bahwa keluaran yang dihasilkan oleh
- 20 perkakas ini akan meliputi seluruh jenis servis yang disediakan oleh Uber.

⁵<https://github.com/jaebradley/uber-cli>

⁶Gambar diambil dari sumber yang sama dengan footnote 5.

21:00 \$ uber time '25 first street cambridge ma'	
25 First St, Cambridge, MA 02141, USA	
 	
2 min. uberPOOL,uberX	
3 min. uberXL	
5 min. UberBLACK,uberSUV	
7 min. TAXI	

Gambar 3.3: Contoh penggunaan fitur prediksi waktu perjalanan untuk perkakas Uber CLI.⁶

- 1 Sedangkan, untuk memanggil fungsi kedua dari perkakas ini, pengguna dapat dilakukan dengan
- 2 memanggil perintah dengan format berikut.

3 uber price -s <alamat awal> -e <alamat akhir>

- 4 Untuk sintaks ini, **uber** memiliki fungsi yang sama dengan sintaks untuk fungsi pertama dari
 5 perkakas. **price** merupakan penanda untuk perkakas bahwa pengguna ingin menggunakan servis
 6 untuk mengetahui perkiraan harga layanan Uber. Selanjutnya, perkakas akan meminta dua buah
 7 opsi beserta parameternya masing-masing. Pertama, opsi **-s**, berarti *start*, yang akan meminta
 8 sebuah parameter berupa lokasi yang ingin dipakai sebagai lokasi awal perkiraan harga layanan
 9 Uber. Sedangkan opsi **-e**, berarti *end*, akan meminta sebuah parameter berupa lokasi yang ingin
 10 dipakai sebagai lokasi akhir jasa perkiraan harga.

- 11 Adapun keluaran dari fungsi kedua ini dapat dilihat di gambar 3.4. Sama seperti keluaran untuk
 12 fungsi pertamanya, keluaran untuk fungsi kedua perkakas ini juga meliputi seluruh jasa yang
 13 disediakan oleh Uber.

21:00 \$ uber price -s '25 first street cambridge ma' -e '114 line street somerville ma'				
     Surge				
uberPOOL \$3-\$6 1.57 mi. 8 min. 				
uberX \$6-\$9 1.57 mi. 8 min. 				
uberXL \$10-\$13 1.57 mi. 8 min. 				
UberBLACK \$15-\$20 1.57 mi. 8 min. 				
uberSUV \$22-\$28 1.57 mi. 8 min. 				
 25 First St, Cambridge, MA 02141, USA				
 END 114 Line St, Somerville, MA 02143, USA				

Gambar 3.4: Contoh penggunaan fitur prediksi harga perjalanan untuk perkakas Uber CLI.⁷

⁷Gambar diambil dari <https://github.com/jaebradley/uber-cli>

1 Permasalahan

2 Seperti telah dijelaskan di awal bab ini, perkakas ini tidak dapat digunakan. Kesimpulan yang
 3 diambil oleh penulis mengenai alasan perkakas ini tidak dapat dijalankan adalah dikarenakan oleh
 4 penggunaan API dan modul-modul yang telah usang (*deprecated*). Kesimpulan ini diambil oleh
 5 penulis karena dua alasan utama. Pertama, pada awalnya, perkakas ini tidak dapat dijalankan
 6 karena API Google *Maps* yang dipakai mengandung baris kode berikut didalamnya.

7 `exports.placesAutoCompleteSessionToken = require('uuid/v4');`

8 Kode ini merupakan kode yang dipakai untuk mengambil *subpath* dari paket *uuid*, tetapi penggunaannya
 9 sudah berubah untuk versi yang lebih barunya. Akan tetapi, setelah diganti baris tersebut
 10 ke penggunaan versi barunya pun, perkakas ini masih tetap tidak dapat dijalankan—sekarang
 11 perkakas ini justru mengembalikan kode error seperti dapat dilihat di bagian lampiran A. Singkatnya
 12 (seperti tertera di akhir pesan error dalam lampiran tersebut), ini berarti perkakas mencoba
 13 untuk mengakses API Uber dengan menggunakan kredensial OAuth 2.0 yang berlaku untuk versi
 14 sebelumnya dari API tersebut. Permasalahan ini merupakan permasalahan yang juga ditemukan
 15 oleh beberapa pengguna lain, seperti tertera di halaman *GitHub Issues* dari repositori ini.⁸ Oleh
 16 karena hal ini tidak lagi merupakan masalah kode perangkat lunak, maka perkakas ini dianggap
 17 tidak dapat dipakai.

18 3.1.4 Google Maps Direction CLI⁹

19 Google Maps Direction CLI merupakan sebuah perkakas *command line* yang memiliki kegunaan
 20 yang mirip dengan KIRI, hanya saja perkakas ini tidak secara spesifik mengharuskan penggunaan
 21 angkot, atau transportasi umum lainnya. Singkatnya, perkakas ini memiliki fungsi seperti sebuah
 22 GPS. Untuk menggunakannya, pengguna harus memasukkan perintah dengan bentuk sebagai
 23 berikut.

24 `direction <lokasi awal> <lokasi akhir>`

25 Setelah pengguna memasukkan perintah tersebut dengan benar, perkakas ini akan mengirim
 26 permintaan ke API Google *Maps*, di mana jika prosesnya berhasil, keluarannya akan berupa langkah-langkah
 27 yang harus ditempuh untuk sampai ke lokasi akhir, beserta di jarak berapa langkah tersebut
 28 harus diambil, relatif terhadap langkah sebelumnya. Adapun penggunaan dari perkakas ini dapat
 29 dilihat di gambar 3.5.

30 Permasalahan

31 Seperti tertulis di awal bab ini, perkakas ini juga tidak bisa digunakan. Alasan perkakas ini tidak
 32 dapat digunakan lagi-lagi merupakan masalah teknikal, yaitu diperbaruiinya API Google *Maps*.
 33 Lebih spesifiknya, semenjak 2018, Google tidak lagi memperbolehkan penggunaan API Google *Maps*
 34 tanpa kunci API, yang sayangnya tidak hanya mendasari perkakas ini, tetapi juga kunci API ini

⁸<https://github.com/jaebradley/uber-cli/issues/87>

⁹<https://github.com/yujinlim/google-maps-direction-cli>

¹⁰Gambar diambil dari <https://github.com/yujinlim/google-maps-direction-cli>

```

Route
Bukit Damansara, Kuala Lumpur, Federal Territory of Kuala Lumpur, Malaysia → Petronas Twin Tower, Kuala Lumpur City Centre, 50088 Kuala Lumpur, Federal Territory of Kuala Lumpur, Malaysia
Duration
18 mins
Routes
Head northeast on Jalan Medan Setia 2 (81 m)
Turn right toward Jalan Medan Setia (28 m)
Merge onto Jalan Medan Setia (45 m)
Turn left onto Jalan Setia Murni 1 (0.3 km)
Turn left onto Jalan Setia Murni 3 (53 m)
Turn left onto Jalan BeringinGo through 1 roundabout (1.0 km)
Take the ramp on the right to Jalan Duta/Kuala Lumpur (28 m)
Merge onto Damansara Link/Lebuhraya SPRINT/Sistem Penyiaran Trafik Kuala Lumpur Barat/E23 (0.4 km)
Continue onto Jalan Semantan (0.9 km)
Take the exit on the right toward K. Lumpur/Jln. Parlimen/PWTC (0.6 km)
Merge onto Jalan Tuank Abdul Halim (0.7 km)
Take the exit toward PWTC/Jln. Parlimen/Jln. Tun Razak/Kuching/Dataran Merdeka (0.5 km)
Keep right at the fork, follow signs for Jalan Parlimen/Dataran Merdeka/Merdeka Square/Taman Botani Perdana/Perdana Botanical Garden (0.5 km)
Turn left onto Jalan Parlimen (1.1 km)
Turn left onto Bulatan Data Onn (91 m)
Take the ramp to Jalan Kuching/Route 1 (0.3 km)
Slight left onto Jalan Kuching/Route 1 (0.1 km)
Continue straight to stay on Jalan Kuching/Route 1 (0.7 km)
Take the Jln. Sultan Ismail exit on the right toward Menara KL/KLCC/Ampang/E12 (0.2 km)
Keep right to continue toward Jalan Sultan Ismail (0.3 km)
Continue onto Jalan Sultan Ismail (1.0 km)
Slight left onto the ramp to Ampang (0.4 km)
Continue onto Lebuhraya Bertingkat Ampang - Kuala Lumpur/E12 (0.8 km)
Take exit 1202A-Jln. Tun Razak toward KLCC (0.5 km)
Merge onto Lebuhraya Bertingkat Ampang - Kuala Lumpur/E12Toll road (0.3 km)
Take the exit toward KLCCPartial toll road (0.6 km)

```

Gambar 3.5: Contoh penggunaan perkakas Google Maps Direction CLI.¹⁰

- ¹ tidak bisa didapatkan tanpa membayarkan biaya tertentu. Oleh karena itu, perkakas ini dianggap
- ² tidak bisa lagi dijalankan.

³ 3.2 Analisis API KIRI

- ⁴ API KIRI memiliki tiga buah layanan, yaitu *search place*, *routing*, dan *smart direction*. Diantara
- ⁵ ketiga layanan ini, perlu diingat bahwa *smart direction* merupakan sebuah layanan yang bekerja
- ⁶ dengan langsung membuka halaman web KIRI sendiri, sehingga layanan ini tidak akan digunakan
- ⁷ dalam pembuatan perkakas *command line* untuk skripsi ini.

⁸ 3.2.1 *Search Place*

- ⁹ Layanan pertama dari dua layanan yang tersisa merupakan layanan *search place*. Layanan ini
- ¹⁰ merupakan layanan API KIRI yang berfungsi untuk mencari sebuah lokasi, beserta nilai *latitude*
- ¹¹ dan *longitude*-nya, berdasarkan kata kunci yang diberikan oleh pengguna. Untuk memanfaatkan
- ¹² layanan ini, sebuah permintaan GET harus dikirimkan ke alamat API KIRI. Adapun permintaan
- ¹³ tersebut akan memiliki parameter-parameter sebagai berikut.

- ¹⁴ • **version**

Parameter ini merupakan tanda bagi API untuk menggunakan protokol versi 2. Karena kemungkinan nilai untuk parameter ini hanya satu, maka parameter ini pasti bernilai 2.

- ¹⁷ • **mode**

Parameter ini merupakan mode dari servis/jasa API yang akan digunakan oleh pengguna. Untuk penggunaan layanan *search place*, parameter ini diisi dengan **searchplace**.

- ²⁰ • **region**

Parameter ini mengatur daerah mana tempat lokasi yang ingin dicari berada. Isi dari parameter ini akan diberikan oleh pengguna pada saat pemakaian perkakas.

1 • **querystring**

2 Parameter ini akan berisi kata kunci yang diberikan oleh pengguna untuk mencari lokasi yang
3 ingin ditemukan.

4 • **apikey**

5 Parameter ini akan berisi sebuah nilai kunci API yang sudah digenerasikan sebelumnya, yaitu
6 68CD281C8A8EE97C.

7 **3.2.2 Routing**

8 Layanan kedua dari API ini merupakan fungsi utama dari KIRI sendiri, yaitu pencarian rute dengan
9 menggunakan angkot. Layanan ini akan menerima dua buah lokasi yang ingin dijadikan lokasi
10 awal dan lokasi akhir, dan menunjukkan langkah-langkah yang harus ditempuh untuk menempuh
11 perjalanan tersebut, dengan memanfaatkan jasa angkot yang ada. Sama seperti layanan *search*
12 *place*, layanan ini juga digunakan dengan mengirimkan permintaan GET ke alamat API KIRI.

13 • **version**

14 Sama seperti pada layanan *search place*, parameter ini hanya dapat diisi dengan nilai 2.

15 • **mode**

16 Parameter ini merupakan mode dari servis/jasa API yang akan digunakan oleh pengguna.
17 Untuk penggunaan layanan *routing*, parameter ini diisi dengan **findroute**.

18 • **locale**

19 Isi dari parameter ini akan ditentukan pada saat pemakaian perkakas.

20 • **start**

21 Parameter ini merupakan nilai *latitude* dan *longitude* dari titik awal perjalanan pengguna,
22 yang akan diberikan sebagai masukan pada saat pemakaian perkakas.

23 • **finish**

24 Parameter ini merupakan nilai *latitude* dan *longitude* dari titik akhir/tujuan perjalanan
25 pengguna, yang akan diberikan sebagai masukan pada saat pemakaian perkakas.

26 • **presentation**

27 Parameter ini hanya digunakan untuk fitur *backwards compatibility*. Jika parameter ini ingin
28 digunakan, maka isi dari parameter ini hanya ada satu kemungkinan, yaitu **desktop**. Jika
29 parameter ini tidak dipakai

30 • **apikey**

31 Parameter ini akan berisi sebuah nilai kunci API yang sudah digenerasikan sebelumnya, yaitu
32 68CD281C8A8EE97C.

33 Layanan ini dapat digunakan sebagai lanjutan dari layanan *search place*, terutama karena fitur
34 pencarian rute hanya menerima nilai *latitude* dan *longitude* dari lokasi-lokasi yang akan digunakan
35 sebagai masukan, yang tidak hanya akan menyusahkan pengguna dalam memakai perkakas *command*
36 *line* ini, tetapi juga kedua nilai tersebut juga merupakan salah satu dari keluaran layanan pencarian
37 lokasi. Jadi, dengan menggunakan kedua layanan tersebut secara berlangsung, maka pengguna
38 hanya perlu mencari lokasi tersebut dengan kata-kata kunci, dan pada akhirnya pengguna akan
39 dapat menerima langkah-langkah yang harus ditempuh dalam rute sebagai keluaran akhir dari
40 perkakas.

DAFTAR REFERENSI

- [1] The GNU C Library (2022) cd. https://www.gnu.org/software/libc/manual/html_node/Getopt.html. versi 2.3.5.
- [2] Marsh, N. (2010) *Introduction to the Command Line: The Fat-Free Guide to Unix and Linux Commands*, 2nd edition. CreateSpace, South Carolina.
- [3] Shotts Jr., W. E. (2019) *The Linux Command Line*, 5th internet edition. <https://www.linuxcommand.org/tlcl.php>.
- [4] Mueller, J. P. (2007) *Windows® Administration at the Command Line for Windows Vista™, Windows® 2003, Windows® XP, and Windows® 2000*, 1st edition. Wiley Publishing, Inc., Indiana.
- [5] Matthew, N. dan Stones, R. (2007) *Beginning Linux® Programming*, 4th edition. Wiley Publishing, Inc., Indiana.
- [6] Microsoft Docs (2021) cd. <https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/cd>. versi 29 Juli 2021.
- [7] Microsoft Docs (2021) dir. <https://docs.microsoft.com/en-us/windows-server/administration/windows-commands/dir>. versi 21 September 2021.
- [8] Stenberg, D. (2022) *Everything curl*. <https://everythingcurl.dev/>.

LAMPIRAN A

KODE PROGRAM

Kode A.1: Pesan Error Perkakas *Uber CLI*

```
1 Could not get time estimates:
2   Error: Request failed with status code 401
3     at createError (C:\Users\Alfred\AppData\Roaming\npm\node_modules\uber-cli\node_modules\axios\lib\core\createError.js
4       :16:15)
5     at settle (C:\Users\Alfred\AppData\Roaming\npm\node_modules\uber-cli\node_modules\axios\lib\core\settle.js:18:12)
6     at IncomingMessage.handleStreamEnd (C:\Users\Alfred\AppData\Roaming\npm\node_modules\uber-cli\node_modules\axios\lib\
7       adapters\http.js:202:11)
8     at IncomingMessage.emit (node:events:538:35)
9     at endReadableNT (node:internal/streams/readable:1345:12)
10    at processTicksAndRejections (node:internal/process/task_queues:83:21) {
11      config: {
12        adapter: [Function: httpAdapter],
13        transformRequest: { '0': [Function: transformRequest] },
14        transformResponse: { '0': [Function: transformResponse] },
15        timeout: 0,
16        xsrfCookieName: 'XSRF-TOKEN',
17        xsrfHeaderName: 'X-XSRF-TOKEN',
18        maxContentLength: -1,
19        validateStatus: [Function: validateStatus],
20        headers: {
21          Accept: 'application/json,_text/plain,_*/*',
22          Authorization: 'Token_We0MNCaIpX00F_TUopt4jgL9BzW3bWWt16aYM4mh',
23          'Accept-Language': 'en-US',
24          'User-Agent': 'axios/0.18.1'
25        },
26        method: 'get',
27        baseURL: 'https://api.uber.com/v1.2/estimates',
28        url: 'https://api.uber.com/v1.2/estimates/time',
29        params: {
30          start_latitude: 42.3697672,
31          start_longitude: -71.077356,
32          productId: null
33        },
34        data: undefined
35      },
36      request: <ref *1> ClientRequest {
37        _events: [Object: null prototype] {
38          abort: [Function (anonymous)],
39          aborted: [Function (anonymous)],
40          error: [Function (anonymous)],
41          socket: [Function (anonymous)],
42          timeout: [Function (anonymous)],
43          prefinish: [Function: requestOnPrefinish]
44        },
45        _eventsCount: 6,
46        _maxListeners: undefined,
47        outputData: [],
48        outputSize: 0,
49        writable: true,
50        destroyed: false,
51        _last: true,
52        chunkedEncoding: false,
53        shouldKeepAlive: false,
54        maxRequestsOnConnectionReached: false,
55        _defaultKeepAlive: true,
56        useChunkedEncodingByDefault: false,
57        sendDate: false,
58        _removedConnection: false,
59        _removedContLen: false,
60        _removedTE: false,
61        _contentLength: 0,
62        _hasBody: true,
63        _trailer: '',
64        finished: true,
65        _headerSent: true,
66        _closed: false,
67        socket: TLSSocket {
68          _tlsOptions: [Object],
69          _secureEstablished: true,
70          _securePending: false,
71          _newSessionPending: false,
72          _controlReleased: true,
73          secureConnecting: false,
74          _SNICallback: null,
75          servername: 'api.uber.com',
76        }
77      }
78    }
79  }
80  at ClientRequest.<anonymous> (C:\Users\Alfred\AppData\Roaming\npm\node_modules\uber-cli\node_modules\axios\lib\request.js:84:14)
81  at ClientRequest.emit (node:events:315:20)
82  at IncomingMessage.<anonymous> (C:\Users\Alfred\AppData\Roaming\npm\node_modules\uber-cli\node_modules\axios\lib\request.js:86:12)
83  at IncomingMessage.emit (node:events:315:20)
84  at endReadableNT (node:internal/streams/readable:1345:12)
85  at processTicksAndRejections (node:internal/process/task_queues:83:21)
```

```

74|     alpnProtocol: false,
75|     authorized: true,
76|     authorizationError: null,
77|     encrypted: true,
78|     _events: [Object: null prototype],
79|     _eventsCount: 10,
80|     connecting: false,
81|     _hadError: false,
82|     _parent: null,
83|     _host: 'api.uber.com',
84|     _readableState: [ReadableState],
85|     _maxListeners: undefined,
86|     _writableState: [WritableState],
87|     allowHalfOpen: false,
88|     _sockname: null,
89|     _pendingData: null,
90|     _pendingEncoding: '',
91|     server: undefined,
92|     _server: null,
93|     ssl: [TLSSwrap],
94|     _requestCert: true,
95|     _rejectUnauthorized: true,
96|     parser: null,
97|     _httpMessage: [Circular *1],
98|     [Symbol(res)]: [TLSSwrap],
99|     [Symbol(verified)]: true,
100|     [Symbol(pendingSession)]: null,
101|     [Symbol(async_id_symbol)]: 48,
102|     [Symbol(kHandle)]: [TLSSwrap],
103|     [Symbol(kSetNoDelay)]: false,
104|     [Symbol(lastWriteQueueSize)]: 0,
105|     [Symbol(timeout)]: null,
106|     [Symbol(kBuffer)]: null,
107|     [Symbol(kBufferCb)]: null,
108|     [Symbol(kBufferGen)]: null,
109|     [Symbol(kCapture)]: false,
110|     [Symbol(kBytesRead)]: 0,
111|     [Symbol(kBytesWritten)]: 0,
112|     [Symbol(connect-options)]: [Object],
113|     [Symbol(RequestTimeout)]: undefined
114},
115 _header: 'GET /v1.2/estimates/time?start_latitude=42.3697672&start_longitude=-71.077356 HTTP/1.1\r\n' +
116   'Accept: application/json, text/plain, */*\r\n' +
117   'Authorization: Token We0MNCaIpX00F_TUopt4jgL9BzW3bWwt16aYM4mh\r\n' +
118   'Accept-Language: en_US\r\n' +
119   'User-Agent: axios/0.18.1\r\n' +
120   'Host: api.uber.com\r\n' +
121   'Connection: close\r\n' +
122   '\r\n',
123 _keepAliveTimeout: 0,
124 _onPendingData: [Function: nop],
125 agent: Agent {
126   _events: [Object: null prototype],
127   _eventsCount: 2,
128   _maxListeners: undefined,
129   defaultPort: 443,
130   protocol: 'https:',
131   options: [Object: null prototype],
132   requests: [Object: null prototype] {},
133   sockets: [Object: null prototype],
134   freeSockets: [Object: null prototype] {},
135   keepAliveMsecs: 1000,
136   keepAlive: false,
137   maxSockets: Infinity,
138   maxFreeSockets: 256,
139   scheduling: 'lifo',
140   maxTotalSockets: Infinity,
141   totalSocketCount: 1,
142   maxCachedSessions: 100,
143   _sessionCache: [Object],
144   [Symbol(kCapture)]: false
145 },
146 socketPath: undefined,
147 method: 'GET',
148 maxHeaderSize: undefined,
149 insecureHTTPParser: undefined,
150 path: '/v1.2/estimates/time?start_latitude=42.3697672&start_longitude=-71.077356',
151 _ended: true,
152 res: IncomingMessage {
153   _readableState: [ReadableState],
154   _events: [Object: null prototype],
155   _eventsCount: 3,
156   _maxListeners: undefined,
157   socket: [TLSSocket],
158   httpVersionMajor: 1,
159   httpVersionMinor: 1,
160   httpVersion: '1.1',
161   complete: true,
162   rawHeaders: [Array],
163   rawTrailers: [],
164   aborted: false,
165   upgrade: false,
166   url: '',
167   method: null,
168   statusCode: 401,
169   statusMessage: 'Unauthorized',
170   client: [TLSSocket],
171   _consuming: false,
172   _dumped: false,

```

```

173|     req: [Circular *1],
174|     responseUrl: 'https://api.uber.com/v1.2/estimates/time?start_latitude=42.3697672&start_longitude=-71.077356',
175|     redirects: [],
176|     [Symbol(kCapture)]: false,
177|     [Symbol(kHeaders)]: [Object],
178|     [Symbol(kHeadersCount)]: 24,
179|     [Symbol(kTrailers)]: null,
180|     [Symbol(kTrailersCount)]: 0,
181|     [Symbol(RequestTimeout)]: undefined
182|,
183|     aborted: false,
184|     timeoutCb: null,
185|     upgradeOrConnect: false,
186|     parser: null,
187|     maxHeadersCount: null,
188|     reusedSocket: false,
189|     host: 'api.uber.com',
190|     protocol: 'https:',
191|     _redirectable: Writable {
192|       _writableState: [WritableState],
193|       _events: [Object: null prototype],
194|       _eventsCount: 2,
195|       _maxListeners: undefined,
196|       _options: [Object],
197|       _redirectCount: 0,
198|       _redirects: [],
199|       _requestBodyLength: 0,
200|       _requestBodyBuffers: [],
201|       _onNativeResponse: [Function (anonymous)],
202|       _currentRequest: [Circular *1],
203|       _currentUrl: 'https://api.uber.com/v1.2/estimates/time?start_latitude=42.3697672&start_longitude=-71.077356',
204|       [Symbol(kCapture)]: false
205|     },
206|     [Symbol(kCapture)]: false,
207|     [Symbol(kNeedDrain)]: false,
208|     [Symbol(corked)]: 0,
209|     [Symbol(kOutHeaders)]: [Object: null prototype] {
210|       accept: [Array],
211|       authorization: [Array],
212|       'accept-language': [Array],
213|       'user-agent': [Array],
214|       host: [Array]
215|     }
216|,
217|     response: {
218|       status: 401,
219|       statusText: 'Unauthorized',
220|       headers: {
221|         date: 'Fri, 20 May 2022 12:46:31 GMT',
222|         'content-type': 'application/json',
223|         'content-length': '75',
224|         'x-uber-edge': 'e4-dca4:w:85',
225|         'strict-transport-security': 'max-age=31536000',
226|         'x-content-type-options': 'nosniff',
227|         'x-xss-protection': '1; mode=block',
228|         'x-frame-options': 'SAMEORIGIN',
229|         'cache-control': 'max-age=0',
230|         'x-envoy-upstream-service-time': '69',
231|         server: 'ufe',
232|         connection: 'close'
233|       },
234|       config: {
235|         adapter: [Function: httpAdapter],
236|         transformRequest: [Object],
237|         transformResponse: [Object],
238|         timeout: 0,
239|         xsrfCookieName: 'XSRF-TOKEN',
240|         xsrfHeaderName: 'X-XSRF-TOKEN',
241|         maxContentLength: -1,
242|         validateStatus: [Function: validateStatus],
243|         headers: [Object],
244|         method: 'get',
245|         baseURL: 'https://api.uber.com/v1.2/estimates',
246|         url: 'https://api.uber.com/v1.2/estimates/time',
247|         params: [Object],
248|         data: undefined
249|       },
250|       request: <ref *1> ClientRequest {
251|         _events: [Object: null prototype],
252|         _eventsCount: 6,
253|         _maxListeners: undefined,
254|         outputData: [],
255|         outputSize: 0,
256|         writable: true,
257|         destroyed: false,
258|         _last: true,
259|         chunkedEncoding: false,
260|         shouldKeepAlive: false,
261|         maxRequestsOnConnectionReached: false,
262|         _defaultKeepAlive: true,
263|         useChunkedEncodingByDefault: false,
264|         sendDate: false,
265|         _removedConnection: false,
266|         _removedContLen: false,
267|         _removedTE: false,
268|         _contentLength: 0,
269|         _hasBody: true,
270|         _trailer: '',
271|         finished: true,

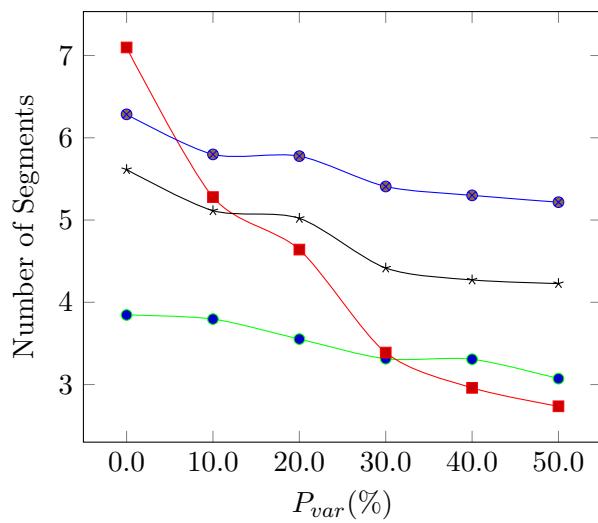
```

```
272     _headerSent: true,
273     _closed: false,
274     socket: [TLSocket],
275     _header: 'GET /v1.2/estimates/time?start_latitude=42.3697672&start_longitude=-71.077356 HTTP/1.1\r\n' +
276       'Accept: application/json, text/plain, */*\r\n' +
277       'Authorization: Token We0MNCaIpX00F_TUopt4jgL9BzW3bWWt16aYM4mh\r\n' +
278       'Accept-Language: en-US\r\n' +
279       'User-Agent: axios/0.18.1\r\n' +
280       'Host: api.uber.com\r\n' +
281       'Connection: close\r\n' +
282       '\r\n',
283     _keepAliveTimeout: 0,
284     _onPendingData: [Function: nop],
285     agent: [Agent],
286     socketPath: undefined,
287     method: 'GET',
288     maxHeaderSize: undefined,
289     insecureHTTPParser: undefined,
290     path: '/v1.2/estimates/time?start_latitude=42.3697672&start_longitude=-71.077356',
291     _ended: true,
292     res: [IncomingMessage],
293     aborted: false,
294     timeoutCb: null,
295     upgradeOrConnect: false,
296     parser: null,
297     maxHeadersCount: null,
298     reusedSocket: false,
299     host: 'api.uber.com',
300     protocol: 'https:',
301     _redirectable: [Writable],
302     [Symbol(kCapture)]: false,
303     [Symbol(kNeedDrain)]: false,
304     [Symbol(corked)]: 0,
305     [Symbol(kOutHeaders)]: [Object: null prototype]
306   },
307   data: {
308     code: 'unauthorized',
309     message: 'Invalid OAuth 2.0 credentials provided.'
310   }
311 }
312 }
```

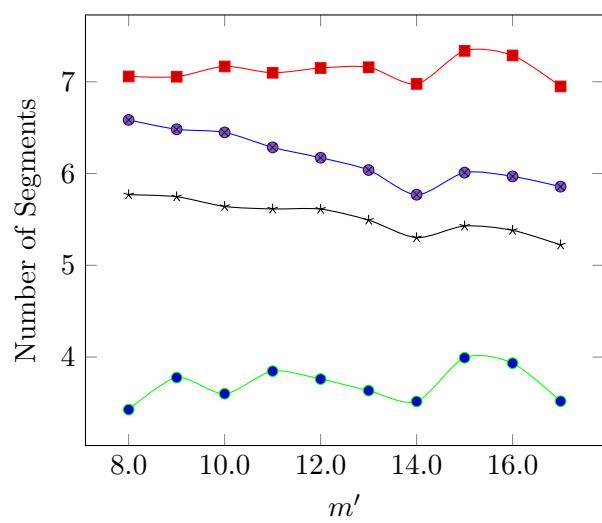
LAMPIRAN B

HASIL EKSPERIMENT

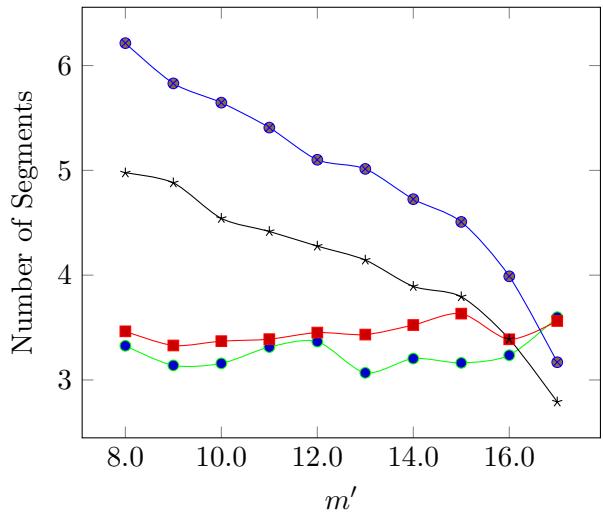
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



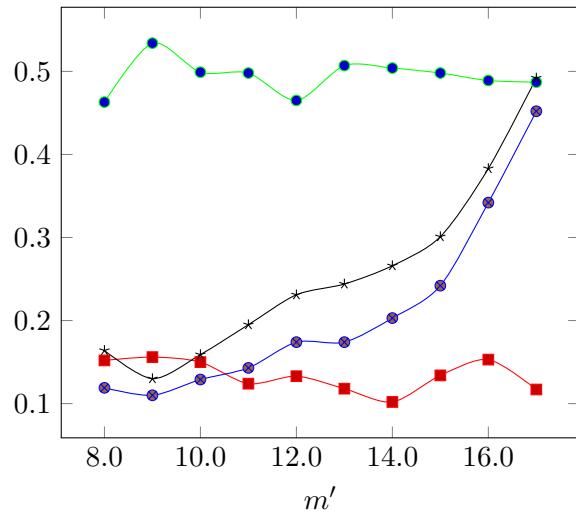
Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4