

SKRIPSI

**PEMVISUALISASI HASIL PENELITIAN AREA HIJAU
KELURAHAN**



Bosnich Timothy Bonasleng

NPM: 2017730086

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2023**

UNDERGRADUATE THESIS

«JUDUL BAHASA INGGRIS»



Bosnich Timothy Bonasleng

NPM: 2017730086

**DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2023**

LEMBAR PENGESAHAN

PEMVISUALISASI HASIL PENELITIAN AREA HIJAU KELURAHAN

Bosnich Timothy Bonasleng

NPM: 2017730086

Bandung, 8 08 2023

Menyetujui,

Pembimbing

Pascal Alfadian, Nugroho, M.Comp.

Ketua Tim Penguji

Anggota Tim Penguji

«penguji 1»

«penguji 2»

Mengetahui,

Ketua Program Studi

Mariskha Tri Adithia, P.D.Eng

PERNYATAAN

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

PEMVISUALISASI HASIL PENELITIAN AREA HIJAU KELURAHAN

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

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

Dinyatakan di Bandung,
Tanggal 8 08 2023

Meterai Rp. 10000

Bosnich Timothy Bonasleng
NPM: 2017730086

ABSTRAK

«Tuliskan abstrak anda di sini, dalam bahasa Indonesia»

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

ABSTRACT

«Tuliskan abstrak anda di sini, dalam bahasa Inggris»

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

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

KATA PENGANTAR

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

Bandung, 08 2023

Penulis

DAFTAR ISI

KATA PENGANTAR	xv
DAFTAR ISI	xvii
DAFTAR GAMBAR	xix
1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Tujuan	4
1.4 Batasan Masalah	4
1.5 Metodologi	4
1.6 Sistematika Pembahasan	4
2 LANDASAN TEORI	5
2.1 <i>Command-line Interface</i>	5
2.1.1 SCP(<i>Secure Copy Protocol</i>)	6
2.2 Hadoop Distributed File System	6
2.3 <i>Python</i>	7
2.3.1 <i>Pillow (PIL Fork)</i>	7
2.4 <i>Base64</i>	9
2.5 Framework Laravel	9
3 ANALISIS	13
3.1 Proses Pembentukan Gambar	13
3.1.1 Mengunduh File Text	14
3.1.2 Mengkonversi Baris Menjadi Gambar .png	15
3.1.3 Menggabungkan Gambar	16
3.2 Pembentukan Gambar Hasil Segmentasi/Klasterisasi	17
3.3 Analisis Perangkat Lunak	19
DAFTAR REFERENSI	23
A KODE PROGRAM	25
B HASIL EKSPERIMEN	27

DAFTAR GAMBAR

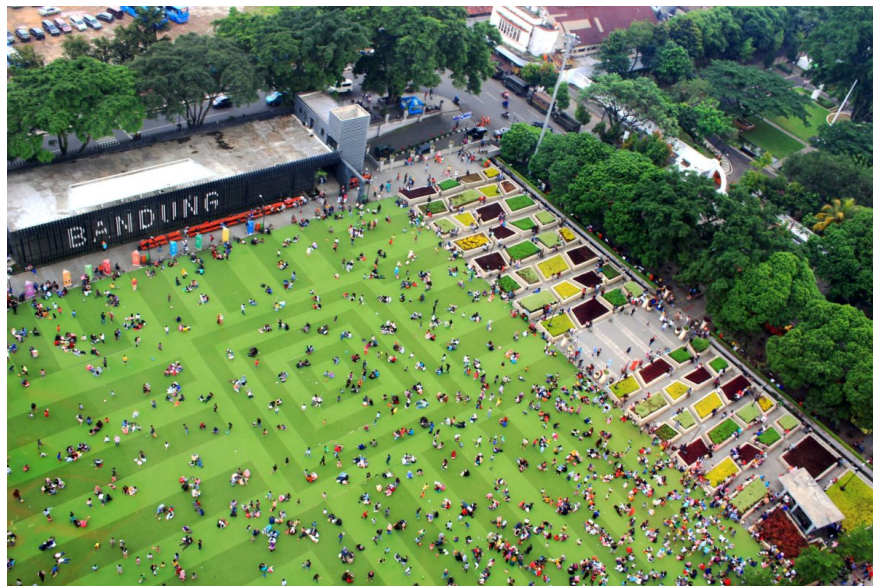
1.1	RTH	1
1.2	Kelurahan Ciumbuleuit	3
2.1	<i>Command Prompt</i>	5
2.2	PHP Artisan Laravel	10
3.1	Gambar seluruh tile dari kelurahan Ciumbuleuit	13
3.2	Contoh gambar kelurahan Ciumbuleuit setiap <i>tile</i>	14
3.3	Data Citra Satelit berupa .txt	15
3.4	Gambar seluruh tile dari kelurahan Ciumbuleuit	16
3.5	Proses pengecekan file Bandung.txt	18
3.6	Diagram <i>Use Case User</i>	19
B.1	Hasil 1	27
B.2	Hasil 2	27
B.3	Hasil 3	27
B.4	Hasil 4	27

BAB 1

PENDAHULUAN

1.1 Latar Belakang

Ruang Terbuka Hijau merupakan suatu ruang terbuka di kawasan perkotaan yang didominasi tutupan lahannya oleh unsur hijau (vegetasi) serta memiliki fungsi antara lain sebagai area untuk rekreasi, sosial budaya, estetika, ekologis dan dapat memberikan nilai ekonomis bagi perkembangan suatu wilayah perkotaan (lihat Gambar 1.1). Definisi RTH sendiri dalam pasal 1 UU No.26/2007 tentang Penataan Ruang adalah area memanjang/jalur dan/atau mengelompok, yang penggunaannya lebih bersifat terbuka, tempat tumbuh tanaman, baik yang tumbuh secara alamiah maupun yang sengaja ditanam. Pada pasal 29 disebutkan bahwa ruang terbuka hijau terdiri dari ruang terbuka hijau publik dan ruang terbuka hijau privat, dimana proporsi ruang terbuka hijau kota paling sedikit 30% dari luas wilayah kota, sedangkan proporsi ruang terbuka hijau publik paling sedikit 20% dari luas wilayah kota.



Gambar 1.1: Contoh Ruang Terbuka Hijau¹

Pemanfaatan citra satelit merupakan sebuah cara agar dapat mengetahui luas RTH pada suatu kota. Citra Satelit adalah gambaran dari permukaan bumi yang didapatkan langsung dari satelit. Oleh karena itu, citra satelit dapat digunakan dalam mengidentifikasi RTH yang mana terdapatnya banyak pepohonan pada suatu wilayah. Perhitungan juga dapat dilakukan pada citra satelit, dan hasil dari perhitungan luas RTH pada suatu wilayah diharapkan dapat memberikan dorongan untuk peningkatan dalam penghijauan agar dapat digunakan oleh pemerintah dalam merancang dan meningkatkan penghijauan di berbagai wilayah di Indonesia.

Penelitian yang dilakukan oleh Juan A. Kusjadi yaitu mengimplementasikan program untuk mengumpulkan, menyiapkan, dan menganalisis data citra satelit kelurahan dari beberapa kota/ka-

bupaten di Indonesia menggunakan Hadoop MapReduce. Data kemudian disimpan pada sistem data lake yang telah dibuat pada Hadoop HDFS. Data hasil analisis dan perhitungan luas RTH juga sudah dilakukan evaluasi dengan nilai sesungguhnya[1]. Data hasil penelitian akan digunakan sebagai penunjang dalam pembuatan halaman web.

Pada Skripsi ini, akan dibangun sebuah halaman web yang interaktif yaitu pemvisualisasian dari hasil penelitian area hijau Kota Bandung[1]. Visualisasi adalah rekayasa dalam pembuatan gambar, diagram atau animasi untuk penampilan suatu informasi dalam penjelasan lain visualisasi adalah konversi data ke dalam format visual atau tabel sehingga karakteristik dari data dan relasi diantara item data atau atribut dapat di analisis atau dilaporkan, dan visualisasi data adalah satu dari yang teknik paling baik dan menarik untuk eksplorasi data. Manusia memiliki kemampuan membangun yang baik untuk menganalisis sejumlah besar informasi yang dipresentasi secara visual. Ia dapat mendeteksi pola umum dan trend, pencilaan dan pola yang tidak umum. Oleh karena itu, dengan dikembangkannya halaman web ini memiliki tujuan agar para pengguna dapat mengetahui informasi yang terdapat pada kelurahan. Informasi yang terdapat pada halaman website berupa nama kelurahan, luas wilayah kelurahan, gambar dari kelurahan/kecamatan, dll. Informasi yang terkumpul akan digunakan untuk mengembangkan halaman web.

Halaman web yang akan dikembangkan harusnya dapat diakses melalui komputer atau laptop. Dalam pengembangan halaman web pemvisualisasi area hijau kota Bandung akan dibantu pembuatannya dengan menggunakan *Framework* Laravel. Penggunaan *Framework* Laravel untuk memudahkan pengembang untuk membangun halaman web, sehingga pengguna yang akan mengakses halaman web akan dimudahkan dalam melihat informasi kelurahan dengan cepat.

Laravel merupakan *framework* PHP yang menekankan pada kesederhanaan dan fleksibilitas pada desainnya. Keunggulan ini didapatkan karena Laravel menggunakan konsep MVC (Model View Controller). Model pada Laravel berguna untuk membantu pengembang berinteraksi dengan database menggunakan *syntax migration* yang merupakan bawaan dari Laravel. Dengan *migration*, pengembangan dapat dengan mudah untuk melakukan modifikasi sebuah database pada sebuah platform secara independen karena implementasi skemas *database* yang direpresentasikan dalam sebuah class. View pada Laravel akan menjadi wadah tampilan website (*frond-end*). Dan Controller yang berfungsi untuk merespon setiap request yang ada pada website sehingga setiap fungsi yang ada akan berfungsi sebagaimana mestinya. Dengan berbagai kemudahan dan fitur yang ada pada Laravel inilah yang membuat pemngembang ingin menggunakannya dalam membangun halaman web pemvisualisasian area hijau kelurahan kota Bandung.

Dalam proses pengembangan halaman web tentu saja dibutuhkan sebuah data. Data yang akan digunakan dalam pembentukan halaman web berupa gambar dari kelurahan di Kota Bandung. Tidak hanya berupa gambar dari kelurahan tetapi juga berupa luas area wilayah untuk mengetahui besar wilayah kelurahan, mengetahui luas wilayah hijau kelurahan, dan melihat kebutuhan area hijau terhadap kelurahan di Kota Bandung. Perhitungan luas wilayah, luas wilayah hijau, dan kebutuhan area hijau telah dilakukan perhitungan untuk setiap kelurahan yang ada.



Gambar 1.2: Kelurahan Ciumbuleuit

Contoh gambar dari hasil penelitian Juan A. Kusjadi dapat dilihat pada gambar 3.1 yang mana merupakan hasil proses pengambilan gambar sebuah kelurahan Ciumbuleuit yang telah disimpan pada Hadoop HDFS. Proses pengambilan gambar dilakukan dengan menggunakan bahasa pemrograman Python². Berbagai macam *library* yang dapat digunakan pada bahasa pemrograman Python dalam membantu pengembangan laman web, diantaranya menggunakan *Python PIL(Pillow)* yang berguna untuk menggabungkan gambar, dan *library* base 64 yang digunakan dalam melakukan peng-*decode*-an teks yang merupakan sebuah *tile* gambar kelurahan.

Hasil dari pemvisualisasi ruang terbuka hijau kelurahan pada kota Bandung akan menjadi sebuah halaman website yang interaktif yang dapat membandingkan kelurahan sesuai dengan masukan oleh pengguna. Dengan dikembangkan halaman website ini maka pengguna dapat memenuhi kebutuhan tempat tinggal bagi masyarakat agar dapat beraktivitas dengan normal dan mendapatkan kadar oksigen yang merata.

1.2 Rumusan Masalah

Berdasarkan deskripsi dan latar belakang yang sudah dibahas bahwa rumusan masalah yang muncul adalah sebagai berikut:

- Bagaimana membuat sebuah halaman website interaktif yang dapat membandingkan data dua buah kelurahan Kota Bandung?
- Bagaimana cara pengguna untuk membandingkan atribut-atribut Citra Satelit dari kelurahan Kota Bandung?
- Bagaimana cara mengekstraksi data citra satelit pada HDFS ke *local directory*?

²Python adalah bahasa pemrograman yang banyak digunakan dalam aplikasi web, pengembangan perangkat lunak, ilmu data, dan *machine learning* (ML). Developer menggunakan Python karena efisien dan mudah dipelajari serta dapat dijalankan di berbagai platform.

1.3 Tujuan

Tujuan dari skripsi ini adalah:

1. Membuat sebuah halaman website interaktif yang dapat membandingkan dua lokasi kelurahan.
2. Pengguna dapat memilih kelurahan untuk sisi kiri dan kanan, untuk membandingkan atributnya.
3. Data citra satelit yang didapatkan akan digunakan untuk memenuhi kebutuhan pada lawan web yang akan dibangun.

1.4 Batasan Masalah

Batasan-batasan masalah untuk penelitian ini adalah sebagai berikut:

1. Penelitian dari data yang sudah matang.

1.5 Metodologi

Metodologi yang akan digunakan dalam pembuatan skripsi adalah:

1. Melakukan survei kepada Fritz H. Hutapea SKom dan Juan A. Kusjadi terkait penenilitiannya
2. Melakukan pengumpulan data hasil penelitian
3. Mempelajari ekstraksi data citra satelit yang disimpan pada HDFS
4. Mempelajari bahasa pemrograman php, html, css dan cara menggunakan *framework* laravel.
5. Mempelajari kebutuhan laman web.
6. Melakukan analisis kebutuhan laman web.
7. Melakukan perancangan antar muka laman web.
8. Membangun laman web berdasarkan *framework* Laravel.
9. Melakukan pengujian pada laman web.
10. Menulis dokumen skripsi.

1.6 Sistematika Pembahasan

Skripsi ini disusun dalam beberapa bab secara sistematis sebagai berikut:

- **Bab 1 Pendahuluan**
Berisikan tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi, dan sistematika pembahasan.
- **Bab 2 Landasan Teori**
Berisikan tentang dasar-dasar dari teori-teori yang digunakan dalam membangun halaman web seperti *Command-line interface*, *Hadoop Distributed File System*, *Python* beserta *library*-nya, Base 64, dan *Framework*.
- **Bab 3 Analisis**
Pada bab ini akan menjelaskan proses pembentukan gambar didalamnya terdapat bagaimana cara pengunduhan teks, pengkonversian baris menjadi gambar, dan menggabungkan gambar. Juga terdapat analisis kebutuhan perangkat lunak.

BAB 2

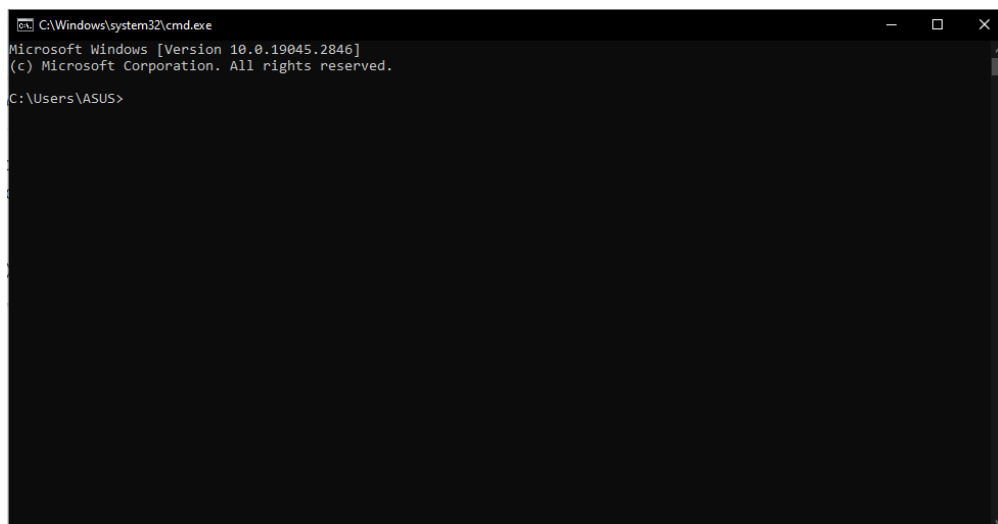
LANDASAN TEORI

Pada bab ini berisikan penjelasan tentang teori-teori yang perlu diketahui sebelum pengembangan halaman web dilakukan.

2.1 *Command-line Interface*

Command-line interface (CLI) merupakan sebuah antarmuka pengguna yang berbasis teks yang digunakan untuk menjalankan program, mengelola berkas-berkas pada komputer, dan dapat berinteraksi dengan komputer¹. *Command-line interface* juga disebut sebagai *command-line user interfaces*, *console user interfaces*, dan *character user interfaces*. *Command-line interface* menerima sebuah perintah yang diinput melalui keyboard perintah yang dipanggil oleh *command prompt* yang dijalankan oleh komputer.

Command-line interface langsung dapat berfungsi ketika sistem komputer dijalankan. *Command-line interface* dapat terbuka di layar kosong dengan *command prompt* lalu perintah-perintah dapat dimasukkan.



Gambar 2.1: *Command Prompt*

Jenis perintah-perintah dari *Command-line interface* akan berisikan :

1. Perintah-perintah dari sistem yang dikodekan sebagai bagian dari antarmuka sistem operasi
2. Program yang dapat dijalankan ketika berhasil dipanggil, dan menjalankan aplikasi yang berbasis teks atau grafis.
3. *batch program*² (*batch files* atau *shell script*) yang merupakan berkas teks berisikan urutan

¹<https://www.techtarget.com/searchwindowserver/definition/command-line-interface-CLI>

²file teks yang berisi serangkaian perintah yang dimaksudkan untuk dieksekusi oleh *command interpreter*

perintah-perintah. Ketika perintah berhasil dipanggil, *batch program* akan menjalankan perintahnya yang mungkin berisikan sebuah perintah sistem dan program yang dapat dieksekusi. Perintah *Command-line interface* yang digunakan antaralain :

2.1.1 SCP(*Secure Copy Protocol*)

Salah satu perintah yang terdapat pada *Command-line interface* yaitu SCP(*secure copy*). SCP memiliki fungsi yang mirip seperti pada perintah *cp(copy)* yaitu untuk menyalin berkas[2]. Perbedaannya yang paling terlihat terletak pada sumber atau tujuan ke *remote host*. Sebagai contoh, jika ingin menyalin sebuah dokumen dari *local directory*(berkas dalam komputer) ke *remote system*, atau dari *working directory* ke *local directory*.

Kode 2.1: Pemanggilan SCP

```
1 C:\Users\Asus>scp ssh i17086@10.100.69.101:Kota_Bandung.txt
```

Kode program 2.1 merupakan contoh penyalinan berkas Kota_Bandung.txt. Berkas tersebut yang tersimpan didalam *remote host* dan disalin ke *local directory* pengguna.

2.2 Hadoop Distributed File System

HDFS (*Hadoop Distributed File System*) merupakan sistem file terdistribusi yang berada pada penyimpanan server dan memiliki banyak kesamaan pada *base storage system*. Sistem penyimpanan terdistribusi ini dapat menyimpan data dalam jumlah yang sangat besar melalui jaringan komputer dengan redundansi bawaan untuk melindungi data. HDFS dirancang untuk pemrosesan yang cepat dan toleran terhadap kesalahan, sehingga memungkinkan pengguna *hardware* pada penyimpanan tidak terkena biaya yang mahal.

HDFS memungkinkan para pengguna untuk menyimpan data kedalam file yang dibagi menjadi beberapa *block*. Karena *Hadoop* dirancang untuk bekerja dengan jumlah data yang besar, ukuran *block* pada HDFS jauh lebih besar daripada yang digunakan oleh *typical relational databases*. Dengan ukuran awal *block* sebesar 128MB, dan dapat dikonfigurasi ukurannya mencapai 512MB.[3]

HDFS memiliki 2 jenis *node*, yaitu *namenode* sebagai *node master* dan *datanode* sebagai *node slave*. Kelebihan utama yang ditawarkan HDFS adalah *scalability* dan *availability* yang dicapai dikarenakan memiliki kemampuan replikasi data dan *fault tolerance*. Dengan adanya kemampuan replikasi data/file, ketika ada kegagalan *software* atau *hardware*, HDFS akan melakukan replikasi ulang blok-blok data pada *node* yang mengalami kegagalan.[4]

Semua perintah HDFS dipanggil menggunakan *script bin/hdfs*. Penjalanan *script "hdfs"* tanpa argumen akan mencetak deskripsi untuk semua perintah.³

Kode 2.2: Perintah HDFS CLI

```
1 C:\Users\Asus>hdfs [SHELL_OPTION] COMMAND [GENERIC_OPTIONS] [COMMAND_OPTIONS]
```

Kode program 2.2 merupakan pemanggilan perintah pada HDFS. Setiap opsi perintah memiliki fungsi untuk menjalankan *script* pada CLI. Penjelasan tiap opsi dijelaskan pada tabel 2.1.

Tabel 2.1: Hadoop memiliki opsi parsing framework yang menjelaskan setiap fungsi kelasnya

COMMAND_OPTION	Deskripsi
SHELL_OPTIONS	kumpulan shell_option yang umum
GENERIC_OPTIONS	kumpulan generic_option yang didukung oleh beberapa perintah
COMMAND COMMAND_OPTIONS	Berbagai perintah dengan opsi

Penggunaan perintah HDFS yang digunakan antara lain :

³<https://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSCommands.html>

1. Penggunaan perintah **dfs**

Perintah **dfs** digunakan untuk menjalankan(*run*) perintah *filesystem* yang didukung oleh *Hadoop*. [**COMMAND_OPTIONS**] dapat dilihat pada *File System Guide*. Contoh pemanggilan **dfs** seperti pemanggilan pada Gambar 2.3

Kode 2.3: Perintah HDFS dfs

```
1 C:\Users\Asus>hdfs dfs [COMMAND [COMAND_OPTIONS]]
```

2. Penggunaan perintah **get**

Perintah **get** digunakan untuk menyalin file HDFS ke *local system*. Gambar 2.4 merupakan contoh yang menunjukkan cara penggunaan perintah **-get** untuk mengunduh file dari HDFS ke *local file system*

Kode 2.4: Perintah HDFS dfs -get untuk mengunduh file HDFS Kota_Bandung.txt ke *local system*

```
1 C:\Users\Asus>hdfs dfs -get /user/18059/geodata/cropped/arcgis/16/Jawa_Barat/Kota_Bandung.txt .
```

3. Penggunaan perintah **-ls**

Perintah **-ls** digunakan untuk menampilkan daftar isi *directory* yang ditentukan oleh *path* yang disediakan oleh pengguna. Gambar merupakan contoh yang menunjukkan cara penggunaan perintah **-ls** untuk melihat isi file HDFS.

Kode 2.5: Perintah HDFS dfs -get untuk mengunduh file HDFS Kota_Bandung.txt ke *local system*

```
1 C:\Users\Asus>hdfs dfs -ls /user/18059/geodata/cropped/arcgis/16/Jawa_Barat
```

2.3 Python

Python adalah bahasa pemrograman yang memulai debutnya pada tahun 1991. *Python* mencakup *object-oriented programming* dan memperkenalkan *syntax* yang membuat banyak *operations* menjadi sangat ringkas dan elegan. Hal yang harus diperhatikan oleh *programmers* baru mengenai *Python* adalah pemakaian spasi(" ") sangat berpengaruh pada arti program yang dikembangkan. Pada proses pengembangan menggunakan bahasa *Python* harus menggunakan *text editor* yang dapat mengenali *syntax*-nya agar memudahkan membuat program sesuai yang diinginkan.[5]

Bahasa pemrograman *Python* Merupakan sebuah bahasa pemrograman komputer yang dikembangkan khusus untuk membuat *source code* yang mudah untuk dibaca. *Pyhton* memiliki *library* yang lengkap sehingga memudahkan seorang *programmer* untuk membuat sebuah aplikasi sesuai dengan keinginan dengan menggunakan *source code* yang terlihat sederhana.

2.3.1 Pillow (PIL Fork)

Python Imaging Library merupakan salah satu *library* yang terdapat pada bahasa pemrograman *Python*. PIL dapat menambahkan pemrosesan gambar ke bahasa pemrograman *Python*. *Library* ini menyediakan *extensive file format*, representasi internal yang efisien, dan memiliki kemampuan yang baik dalam pemrosesan gambar. Pentingnya *library* yang dirancang untuk dapat mengakses data yang disimpan dengan cepat dalam berbagai format piksel. Seharusnya memberikan dasar yang kuat sebagai alat pengolahan gambar⁴.

Python Imaging Library sangat ideal untuk pengarsipan gambar dan aplikasi pemrosesan *batch*. Penggunaan *library* untuk membuat thumbnail, mengonversi antara format file, mencetak gambar, dll. Versi saat ini dapat mengidentifikasi dan membaca sejumlah besar format. Pembantuan dalam penulisan ini sengaja dibatasi dalam format pertukaran dan representasi yang paling umum digunakan.

⁴<https://pillow.readthedocs.io/en/stable/handbook/overview.html>

Python Imaging Library yang rilis saat ini mencakup antarmuka Tk *PhotoImage* dan *BitmapImage*, serta *Windows DIB interface* yang dapat digunakan dengan *PythonWin* dan berbagai macam *toolkits* yang berbasis Windows. Banyak *toolkits* GUI (*Grapiical User Interface*) lainnya yang dilengkapi dengan dukungan PIL. Untuk *debugging*, ada juga metode *show()* yang menyimpan gambar ke disk, dan memanggil utilitas tampilan eksternal.

Penggunaan kelas *Image*

Dalam penggunaan *Python Imaging Library* terdapat kelas yang paling penting yaitu kelas *Image*, yang didefinisikan dalam modul dengan nama yang sama. Pembuat *instance* dari kelas ini dengan beberapa cara; Baik dengan memuat gambar dari file, memproses gambar lain, atau membuat gambar dari awal. Memuat gambar dari file.

1. Penggunaan fungsi *Image.open()*

Berfungsi untuk membuka dan mengidentifikasi file gambar yang diberikan. Fungsi ini mengidentifikasi sebuah file, tetapi file tetap terbuka dan data gambar tidak terbaca sampai data file gambar tersebut diproses. Memiliki parameter **fp** merupakan nama file yang akan dibuka, **mode** memiliki argumen "r", dan **formats** sebuah daftar format untuk mencoba memuat file. Parameter ini dapat digunakan untuk membatasi format yang akan diperiksa. Dalam penggunaan fungsi dapat dilihat pada kode program 2.6.

Kode 2.6: Pemanggilan fungsi *open()*

```
1 from PIL import Image
2
3 im = image.open("hopper.ppm")
```

Jika pemanggilan fungsi berhasil, fungsi yang dipanggil akan mengembalikan sebuah objek *Image*

2. Penggunaan fungsi *Image.new()*

Berfungsi untuk membuat gambar baru dengan mode dan ukuran yang diberikan. Memiliki parameter **mode** untuk menentukan jenis dan kedalaman piksel dalam gambar seperti mode "L" (8-bit piksel, skala abu-abu), "RGB" (3x8-bit piksel, warna asli), "RGBA" (4x8-bit piksel, warna asli dengan *transparency mask*), dll. Parameter **size** merupakan ukuran dari gambar baru, berisi ukuran horizontal dan vertikal dalam piksel. Parameter **color** memberikan warna apa yang akan digunakan. Biasanya akan langsung diberikan warna hitam.

Kode 2.7: Pemanggilan fungsi *new()*

```
1 from PIL import Image
2
3 im = image.new("hopper.ppm")
```

Jika pemanggilan fungsi pada gambar 2.7 berhasil, maka akan mengembalikan sebuah objek *image*.

3. Penggunaan fungsi *Image.paste()*

Berfungsi untuk menempelkan sebuah objek gambar ke objek gambar lain. Ukuran gambar yang ditempelkan harus sesuai dengan ukuran gambar. Memiliki parameter **im** yang merupakan sebuah objek image atau nilai piksel. Parameter **box** 4-tupel opsional yang memberikan wilayah untuk ditempelkan. Jika 2-tupel digunakan sebagai gantinya, itu diperlakukan sebagai sudut kiri atas. Jika dihilangkan atau tidak ada, objek gambar yang ditempelkan ke sudut kiri atas. Parameter **mask** merupakan sebuah optional *mask* gambar.

Kode 2.8: Pemanggilan fungsi *paste()*

```
1 from PIL import Image
2
3 im = image.new("hopper.ppm")
4 im1 = image.open()
5
6 im.paste(im1, (256, 256))
```

Jika fungsi pemanggilan pada gambar 2.8 berhasil, akan mengembalikan sebuah objek *Image* yang memuat gambar *im1* yang ditempelkan pada gambar baru *im*.

4. Penggunaan fungsi *Image.save()*

Berfungsi untuk menyimpan gambar dengan nama file yang diberikan. Jika tidak memiliki format yang ditentukan, maka format yang akan digunakan ditentukan dari ekstensi penamaan file, jika memungkinkan. memiliki beberapa parameter diantaranya adalah **fp** merupakan nama file yang akan digunakan memiliki tipe data string, parameter **format** merupakan format file yang akan digunakan pada file tersebut, dan **params** merupakan parameter tambahan untuk penulisan gambar.

2.4 Base64

Base64 merupakan sebuah algoritma yang digunakan untuk mengubah tipe data *bytes* menjadi tipe data yang dapat dilihat (dan sebaliknya). Skema pengkodean biner ke teks pada Base64 sebagai persyaratan untuk mengirim tipe data *bytes* melalui jaringan komunikasi yang tidak mengizinkan tipe data biner tetapi hanya tipe data berbasis teks. Data teks yang dihasilkan terdiri dari berbagai karakter yang terdapat pada standar ASCII. Penggunaan kata Base64 berasal dari jumlah karakter ASCII yang digunakan. 64 karakter yang digunakan antara lain adalah 26 karakter a-z *lowercase*, 26 karakter A-Z *uppercase*, ditambah dengan 2 karakter tambahan yaitu karakter tambah "+" dan karakter garis miring "/". Base64 juga sebenarnya memiliki karakter ke 65 yaitu karakter sama dengan "=" yang digunakan sebagai *padding*. Karakter sama dengan ("=") digunakan pada segmen terakhir data biner yang tidak memiliki total 6 *bit*. Keseluruhan karakter yang digunakan pada Base64 disebut juga tabel encoding Base64.

Base64 bekerja dengan cara memotong data biner menjadi segmen-segmen berukuran 6 *bit*. Base64 hanya menggunakan 6 *bit* untuk bisa memenuhi seluruh karakter yang digunakan ($26 = 64$). Masing-masing segmen tersebut kemudian dibaca ke dalam tipe desimal lalu dikonversi ke karakter ASCII. Sebagai contoh konversi data yang berisi 3 buah *byte* yaitu 155, 162, dan 233. Tipe data *byte* diubah menjadi data biner dan diambung menjadi satu yaitu 100110111010001011101001. Kemudian data biner dipotong menjadi segmen yang berisi 6 *bit* menjadi 100110, 111010, 001011, 101001. Masing-masing data dikonversi menjadi desimal, 58, 11, 4 yaitu 381. Terakhir data dikonversikan ke karakter ASCII yang berada pada tabel encoding Base64 menjadi m6Lp. Cara yang sama namun terbalik prosesnya digunakan untuk mendeskripsi data dari Base64 kembali ke tipe data *byte*.^[1]

2.5 Framework Laravel

Framework adalah kerangka kerja yang digunakan oleh developer untuk memudahkan pembangunan aplikasi web yang dapat berupa sekumpulan *library* yang berisi fungsi, *tools*, ataupun *class-class*, dan digunakan sebagai kerangka dalam pembangunan aplikasi web. Umumnya didalam *framework* telah menyediakan solusi untuk dapat mengakses *database*, *authentication*, *templating*, *controls*, dan fungsi-fungsi lainnya/ Dalam penggunaan *framework* diharapkan dapat membuat pengembangan aplikasi menjadi rapi dan bersih, memiliki struktur yang optimal, dan *reusable*.

Laravel adalah *framework* aplikasi web dengan sintaks yang ekspresif dan elegan. Laravel adalah *framework* berbasis PHP yang sifatnya *open source*, dan menggunakan konsep *model-view - controller*. Laravel berada di bawah lisensi MIT *License* dengan menggunakan Github sebagai tempat berbagi code menjalankannya. Laravel berlisensi *open source* yang artinya bebas digunakan tanpa harus melakukan pembayaran. Alamat website resmi dari *framework* Laravel adalah <https://laravel.com>. Kelebihan laravel adalah sebagai berikut:

- Progresif *Framework*

Progresif yang dimaksud adalah *framework* ini dapat bertumbuh bersama developer. Yang artinya dapat diikuti oleh developer baru maupun developer senior dikarenakan terdapat

dokumentasi, panduan, dan tutorial video laravel yang dapat membantu membangun perangkat lunak.

- Komunitas *Framework*

Pada laravel terdapat banyak sekali *packages* terbaik dalam ekosistem PHP. selain itu, ribuan pengembang berbakat dari seluruh dunia telah berkontribusi pada *framework* ini.

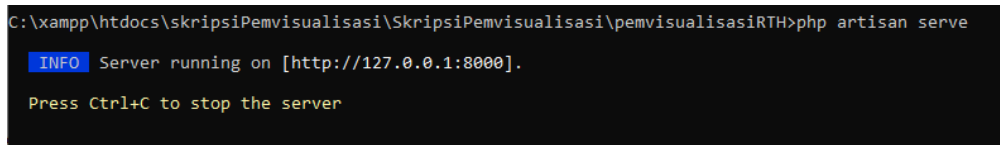
- Berskala *Framework*

Laravel memberikan dukungan sistem cache yang terdistribusi dengan cepat. Faktanya laravel dapat menangani ratusan juta *request* setiap bulan.

Dalam penggunaan laravel memiliki beberapa kekurangan salah satunya yaitu ukuran file yang cukup besar. Di dalam laravel terdapat file yang sifatnya default seperti vendor. File tersebut tidak boleh dihapus sembarangan sehingga ukuran website yang dibuat berukuran cukup besar. Selain itu, dibutuhkan koneksi internet untuk instalasi dan mengunduh *library* laravel, dan PHP minimal versi 5.4 untuk menjalankannya. Berikut adalah dasar-dasar laravel:

1. Artisan

Artisan adalah command line atau perintah yang dijalankan melalui terminal dan disediakan beberapa perintah perintah yang dapat digunakan selama melakukan pengembangan dan pembuatan aplikasi. Salah satu fungsi dari php artisan yaitu “php artisan serve”. Php artisan serve berfungsi untuk membuka website yang telah dibuat tanpa menggunakan web server lokal. Gambar 2.2 merupakan contoh salah satu penggunaan artisan dalam laravel.



```
C:\xampp\htdocs\skripsiPemvisualisasi\SkripsiPemvisualisasi\pemvisualisasiRTH>php artisan serve
INFO Server running on [http://127.0.0.1:8000].
Press Ctrl+C to stop the server
```

Gambar 2.2: PHP Artisan Laravel

2. *Controller*

Controller merupakan suatu proses yang bertujuan untuk mengambil data, menambahkan data, menghapus data, atau mengubah data untuk ditampilkan dalam halaman. Cara membuat *controller* adalah dengan menggunakan *command line* dengan memasukkan "php artisan make controller «nama_controller»". File *controller* nantinya akan otomatis terbuat dan sudah masuk ke folder *controller*.

3. *Routing*

Routing merupakan suatu proses yang dapat memindahkan tampilan halaman ke halaman lain. Dengan menggunakan *routing*, pengguna dapat menentukan halaman yang ingin dikunjungi. Pengaturan *routing* di laravel terletak pada file *web.php*.

4. *Blade View*

Blade adalah *template engine*. Pada dasarnya *Blade* adalah *view* namun dengan menggunakan *Blade* akan mempermudah untuk mengatur tampilan *website* dan menampilkan data. Cara untuk membuat file *view* menjadi file *Blade* adalah dengan menambahkan ekstensi *.blade.php* pada file *view*. Dan cara untuk memanggil file *Blade* sama dengan cara untuk memanggil file *view* biasa.

Setelah melakukan penginstallan Laravel akan terlihat direktori yang berisi aplikasi Laravel dasar. File dan direktori yang terdapat sebagai berikut:[6]

```
app/
bootstrap/
config/
public/
resources/
routes/
storage/
```

```
tests/  
vendor/  
.editorconfig  
.env  
.env.example  
.gitattributes  
.gitignore  
artisan  
composer.json  
composer.lock  
package.json  
phpunit.xml  
readme.md  
server.php  
webpack.mix.js
```

Direktori utama (*root directory*) mengandung folder-folder berikut secara *default*:

- *app*
Berisikan sebagian besar aplikasi saat dijalankan. *Model*, *controller*, *commands*, dan kode domain PHP yang dibuat semuanya akan berada di direktori ini.
- *bootstrap*
Berisi file-file yang digunakan oleh *framework* Laravel saat setiap kali dijalankan.
- *config*
Berisikan semua file konfigurasi aplikasi.
- *database*
Berisi file *database migration* dan *seeds*.
- *public*
Direktori yang ditunjuk oleh server ketika menjalankan aplikasi web. Berisi file *index.php*, yang merupakan *entry point* untuk menangani semua *request* yang masuk ke aplikasi. Didalam folder ini dapat menyimpan beberapa aset dari aplikasi seperti gambar, *JavaScript*, dan *CSS*.
- *resource*
Berisi file *view* dari aplikasi yang dibuat. Didalam folder ini juga terdapat file *language* yang digunakan aplikasi.
- *routes*
Berisi file yang digunakan untuk mendefinisikan semua *route* ke aplikasi. Secara *default* ada tiga file *route* yang disediakan oleh Laravel yaitu *api.php*, *console.php*, dan *web.php*.
- *storage*
Berisi *template Blade* yang dikompilasi, file *session*, file *cache*, dan file lainnya yang dihasilkan secara otomatis oleh Laravel.
- *tests*
Berisi semua file *test* yang dibuat untuk aplikasi.
- *vendor*
Berisikan tempat *Composer* menginstal dependensinya. Direktori ini akan diabaikan oleh Git, karena *Composer* diharapkan dapat berjalan sebagai bagian dari proses implementasi pada server-server jarak jauh.

Direktori utama juga berisi file-file berikut:

- *.editorconfig*
Memberikan instruksi kepada IDE/editor teks tentang standar penulisan kode Laravel (seperti, ukuran indentasi, set karakter, dan apakah harus memotong *whitespace* di ujung baris).
- *.env* dan *.env.example*
Menentukan variabel-variabel lingkungan (variabel yang diharapkan berbeda di setiap lingkungan dan karena itu tidak dimasukkan ke dalam *version control*). *.env.example* adalah

template yang setiap lingkungan harusnya menduplikasi untuk membuat file *.env*-nya sendiri, yang akan diabaikan oleh Git.

- *.gitattributes* dan *.gitignore*
Berisikan file-file untuk pengkonfigurasi Git.
- *artisan*
Berisikan file untuk menjalankan perintah-perintah *artisan* dari *command-line*.
- *composer.json* dan *composer.lock*
Berisikan file konfigurasi untuk *Composer*. *composer.json* dapat diedit oleh pengguna sedangkan *composer.lock* tidak dapat diedit. File-file ini berisi informasi dasar tentang proyek dan juga mendefinisikan dependensi dari PHP.
- *package.json*
File yang berisikan sama seperti *composer.json*, tetapi untuk aset *front-end* dan dependensi dari sistem pembangunan. File ini juga memberi instruksi kepada NPM tentang dependensi berbasis *JavaScript* yang harus diunduh.
- *phpunit.xml*
Berisikan file konfigurasi untuk *PHPUnit* merupakan alat yang digunakan Laravel secara *default* untuk pengujian.
- *readme.md*
Sebuah file *Markdown* yang memberikan pengenalan dasar tentang Laravel. File ini tidak dapat dilihat jika menggunakan instalator dari Laravel.
- *server.php*
Berisikan sebuah server cadangan yang mencoba untuk memungkinkan server yang kurang mampu agar tetap dapat melihat *preview* aplikasi Laravel.
- *webpack.mix.js*
File konfigurasi yang bersifat optional untuk *Mix*. Jika menggunakan *Elixir*, maka akan melihat *gulpfile.js* sebagai gantinya. File-file ini digunakan untuk memberikan instruksi kepada sistem pembangunan tentang cara mengkompilasi dan memproses aset *front-end* aplikasi Laravel.

BAB 3

ANALISIS

Pada bab ini akan dijelaskan mengenai proses pengumpulan data citra satelit yang berada di Hadoop LAB Unpar, pengkonversian data *text* menjadi gambar, dan bagaimana cara penggabungan gambar. Juga akan menjelaskan tentang analisis kebutuhan dalam perancangan perangkat lunak.

3.1 Proses Pembentukan Gambar

Pada Gambar 3.1 merupakan hasil dari penggabungan gambar per *tile*. Langkah-langkah dalam proses pengambilan data berupa gambar citra satelit dari kelurahan di Kota Bandung.



Gambar 3.1: Gambar seluruh tile dari kelurahan Ciumbuleuit

Pertama-tama data yang diambil dari sistem Hadoop yang disimpan pada Hadoop Laboratoium Unpar. Kemudian data yang telah diambil berupa file ".txt" yang setiap baris dari file tersebut merupakan sebuah file gambar berupa *tile* seperti pada gambar(3.2). Kumpulan gambar per *tile* akan digabungkan dengan menggunakan *script*. Penggabungan gambar setiap *tile* akan menghasilkan sebuah gambar dari kelurahan seperti pada gambar 3.1.



Gambar 3.2: Contoh gambar kelurahan Ciumbuleuit setiap *tile*

3.1.1 Mengunduh File Text

Pada penelitian yang telah dilakukan oleh Juan Anthonius Kusjadi menghasilkan data yang telah disimpan pada sistem *data lake* yang telah dibuat pada Hadoop HDFS.[1] Pada proses pengunduhan data harus terlebih dahulu mendaftarkan akun HDFS pada admin laboratoium FTIS UNPAR(Universitas Katholik Parahyangan) agar mendapat akses kedalam sistem penyimpanan HDFS(*Hadoop Distributed File System*). Setelah mendapatkan aksesnya, lalu dapat mengunduh data citra satelit kelurahan per kota dengan perintah seperti pada kode 3.1.

Kode 3.1: *Command-line* HDFS

```
1 >hdfs dfs -get /user/if18059/geodata/cropped/arcgis/16/Jawa_Barat/Kota_Bandung.txt .
```

Metode yang umum digunakan untuk mengunduh file dari server jarak jauh secara aman adalah dengan menggunakan SCP (*Secure Copy Protocol*). SCP adalah perintah baris perintah yang memungkinkan pengguna untuk mentransfer file antara komputer lokal dan server jarak jauh melalui koneksi yang aman. Dalam pengunduhan file dari server dapat menggunakan perintah 'scp' diikuti oleh alamat sumber file di server dan alamat tujuan pada penyimpanan lokal. Dapat dilihat pada *command-line* 3.2, akan mengunduh file 'Kota_Bandung.txt' dari server HDFS laboratorium FTIS UNPAR ke lokasi yang ditentukan pada penyimpanan lokal. Dengan demikian, File yang telah diunduh dapat dengan mudah digunakan.

Kode 3.2: *Command-line* SCP

```
1 C:\Users\Asus>scp ssh i17086@10.100.69.101:Kota_Bandung.txt
```

Data Kota_Bandung.txt dapat dilihat pada gambar 3.3. Isi dari berkas Kota_Bandung.txt memiliki 9 kolom yang dipisahkan oleh tanda titik koma (";"). Pada kolom pertama diisi dengan nama kelurahan. Pada kolom kedua diisi dengan nama kota. Pada kolom ketiga diisi dengan nama provinsi. Kolom keempat diisi dengan nilai panjang tile untuk kelurahan tersebut. Kolom kelima diisi dengan nilai lebar dari tile untuk kelurahan tersebut. Kolom keenam diisi dengan posisi x koordinat tile untuk kelurahan tersebut. Kolom ketujuh diisi dengan posisi y koordinat tile untuk kelurahan tersebut. Kolom kedelapan diisi dengan ukuran luas per piksel dalam km2 untuk tile pada kelurahan tersebut. Terakhir kolom kesembilan diisi dengan data tile citra satelit dengan format png yang sudah dienkripsi dalam bentuk Base64.[1]



Gambar 3.3: Data Citra Satelit berupa .txt

3.1.2 Mengkonversi Baris Menjadi Gambar .png

Dalam penelitian ini, telah dikembangkan sebuah *script* yang bertujuan untuk mengekstraksi gambar pertile dari data Kota_Bandung.txt. Script yang telah dikembangkan dapat dilihat kode program 3.3.

Kode 3.3: Script Mengekstraksi gambar per tile

```

1  import base64
2
3  file = open("Kota_Bandung.txt", "r+")
4  kordinat_x = 0
5  kordinat_y = 0
6  result = []
7
8  for line in file:
9      file_line = file.readline().split(";", 8)
10     image_data = file_line[8]
11     kordinat_x = file_line[5]
12     kordinat_y = file_line[6]
13
14
15     imgdata = base64.b64decode(image_data)
16     filename = file_line[0] + str(kordinat_x) + str(kordinat_y) + '.png'
17     with open(filename, 'wb') as imgd:
18         imgd.write(imgdata)
19
20 file.close()

```

Script ini menggunakan *library* base64 untuk mengelola data gambar yang disimpan dalam format base64. Pertama, *script* membuka file Kota_Bandung.txt dalam mode pembacaan ('r+'). Variabel kordinat_x dan kordinat_y diinisialisasi ke nilai 0. Selama iterasi berlangsung, data dari file dibaca per baris menggunakan perulangan *for line in file*. Setiap baris diproses dengan membaginya menjadi elemen-elemen dengan pemisah titik koma (;). Data gambar yang terdapat di kolom ke-8 di-decode dari format base64 menggunakan base64.b64decode dan disimpan dalam variabel imgdata. Selanjutnya, nama file gambar ditentukan dengan menggabungkan beberapa elemen, seperti nama kelurahan, kordinat_x, dan kordinat_y, dan diberi ekstensi '.png'.

Gambar yang telah di-decode dan disimpan dalam imgdata lalu ditulis ke dalam file baru dengan nama yang telah ditentukan dalam mode binary ('wb') menggunakan *open* dan *imgd.write*. Terakhir, setelah selesai mengolah semua baris, file sumber Kota_Bandung.txt ditutup menggunakan *file.close()*. Dengan *script* code ini, data gambar dalam format base64 di-extract dan disimpan sebagai file gambar .png dengan nama yang koordinat_x dan koordinat_y. Hasil dari penkonversian gambar dari bari dapat dilihat pada 3.4.



Gambar 3.4: Gambar seluruh tile dari kelurahan Ciumbuleuit

3.1.3 Menggabungkan Gambar

Script yang dikembangkan dalam penelitian ini memiliki tujuan utama untuk menggabungkan sejumlah gambar per tile menjadi sebuah gambar utuh yang merepresentasikan kelurahan. Kode pemrograman dapat dilihat pada kode program 3.4, menggunakan library PIL (*Python Imaging Library*) dengan mengimpor kelas *Image*. Terdapat beberapa variabel kunci yang ditentukan, seperti `nama_kecamatan`, `jumlah_baris`, dan `jumlah_kolom`, yang digunakan untuk mengidentifikasi nama kelurahan serta jumlah baris dan kolom yang digunakan untuk mengatur tata letak gambar per tile.

Kode 3.4: Script Penggabungan Gambar

```

1  from PIL import Image
2
3  nama_kecamatan = 'nama_kecamatan/kelurahan'
4  jumlah_baris = 2
5  jumlah_kolom = 3
6
7  panjang_tile = 256
8  lebar_tile = 256
9
10 panjang_gambar = jumlah_kolom*panjang_tile
11 lebar_gambar = jumlah_baris*lebar_tile
12
13 canvas = Image.new("RGB", (panjang_gambar,lebar_gambar))
14 for y in range(jumlah_baris):
15     for x in range(jumlah_kolom):
16         img = Image.open(nama_kecamatan + str(y) + str(x) + ".png")
17         canvas.paste(img, (x*panjang_tile,y*lebar_tile))
18     canvas.save(nama_kecamatan + ".png")

```

Selanjutnya, variabel `panjang_tile` dan `lebar_tile` menentukan ukuran tile, yang dalam contoh ini adalah 256 piksel. Variabel `panjang_gambar` dan `lebar_gambar` dihitung berdasarkan jumlah kolom dan baris, sehingga ukuran gambar akhir dapat ditentukan. Proses pembuatan gambar dimulai dengan inisiasi variabel `canvas` menggunakan fungsi *Image.new* dengan mode "RGB" dan ukuran gambar sesuai dengan `panjang_gambar` dan `lebar_gambar`. Selanjutnya, terdapat dua *loop* di mana *loop* pertama digunakan untuk mengatur koordinat y, dan *loop* kedua untuk koordinat x. Di dalam *loop-loop* tersebut, variabel `img` digunakan untuk membuka gambar tile yang sesuai dengan koordinatnya dengan menambahkan format file yang sesuai. Gambar yang diakses melalui variabel `img` kemudian disisipkan ke dalam gambar utuh `canvas` menggunakan fungsi *paste*. Dan terakhir, *canvas.save* untuk menyimpan hasil gambar akhir dengan nama sesuai dengan 'nama_kecamatan'. Hasil akhir gambar-gambar per tile menjadi gambar utuh yang merepresentasikan wilayah atau

kecamatan yang diinginkan seperti pada gambar 3.1.

3.2 Pembentukan Gambar Hasil Segmentasi/Klasterisasi

Pada penelitian yang telah dilakukan oleh Juan Anthonius Kusjadi menghasilkan data hasil segmentasi area hijau yang telah disimpan pada sistem *data lake* yang telah dibuat pada Hadoop HDFS.[1] Pengunduhan data gambar citra satelit hasil segmentasi area hijau kelurahan dapat dilakukan dengan kode perintah seperti pada kode 3.5.

Kode 3.5: Pengembalian Data Hasil Segemntasi

```
1 >hdfs dfs -get /user/ifu18059/geodata/result/kmeans-5/arcgis/16/Jawa_Barat/Kota_Bandung.csv .
```

Kode Perintah 3.5 digunakan untuk mengunduh file CSV dari HDFS dan menyimpannya di direktori lokal tempat menjalankan perintah tersebut. Perintah `hdfs dfs -get` merupakan perintah untuk mengunduh file dari HDFS. Perintah `/user/ifu18059/geodata/result/kmeans-5 /arcgis/16/Jawa_Barat/Kota_Bandung.csv` adalah jalur lengkap ke file yang ingin diunduh dari HDFS. File yang dimaksud adalah `Kota_Bandung.csv` yang terletak di direktori `/user/ifu18059/geodata/result/kmeans-5/arcgis/16/Jawa_Barat/` di HDFS. Dan Perintah titik (".") merupakan tempat menyimpan file yang diunduh ke direktori lokal.

File `Kota_Bandung.csv` berisikan data hasil klasterisasi. Isi dari file `Kota_Bandung.csv` memiliki 9 kolom yang dipisahkan oleh tanda titik koma (";"). Pada kolom pertama diisi dengan nama kelurahan. Pada kolom kedua diisi dengan posisi x tile citra satelit. Pada kolom ketiga diisi dengan posisi y tile citra satelit. Pada kolom keempat diisi dengan nilai panjang dari tile citra satelit pada kelurahan tersebut. Pada kolom kelima diisi dengan nilai lebar dari tile citra satelit pada kelurahan tersebut. Pada kolom keenam diisi dengan data gambar citra satelit yang sudah disegmentasi dengan format png berdasarkan hasil klasterisasi dan di enkripsi menggunakan Base64. Pada kolom ketujuh diisi dengan data gambar citra satelit asli dengan format png dan di enkripsi menggunakan Base64. Pada kolom kedelapan diisi dengan nilai luas area hijau dalam km2. Terakhir pada kolom kesembilan diisi dengan nilai luas area kelurahan.[1]

Proses pengekstrasian file `Kota_Bandung.csv` berbeda dengan proses pengektrasian yang ada pada 3.1.2 dikarenakan format file yang berbeda maka script yang digunakan juga berbeda. Penggunaan script dapat pada file `Kota_Bandung.csv` dilihat pada 3.6.

Kode 3.6: Script Penggabungan Gambar Hasil Klasterisasi

```
1 import base64
2 import csv
3
4 def extract_image_pertile(csv_file):
5     with open(csv_file, 'r') as file:
6         csv_reader = csv.reader(file)
7
8         for row in csv_reader:
9             kelurahan = row[0]
10            kordinat_x = row[1]
11            kordinat_y = row[2]
12            segmented_image_data = row[5]
13            img_data = base64.b64decode(segmented_image_data)
14            filename = f"{kelurahan}{kordinat_y}{kordinat_x}.png"
15
16            with open(filename, 'wb') as img_file:
17                img_file.write(img_data)
18
19
20 if __name__ == "__main__":
21
22     csv_file = "Kota_Bandung.csv"
23     csv.field_size_limit(1000000)
24
25     extract_image_pertile(csv_file)
```

Script Python di atas dirancang untuk memproses data gambar tersegmen dalam format base64 yang terdapat dalam sebuah file `Kota_Bandung.csv`. Script ini bertujuan untuk mengekstrak dan mendekode data gambar tersebut, kemudian menyimpannya sebagai file gambar PNG. Fungsi utama yang terlibat dalam proses ini disebut `extract_image_pertile`, yang akan menerima nama file CSV sebagai parameter. Fungsi tersebut membuka file `Kota_Bandung.csv`, lalu membaca setiap

baris, dan membaca nilai-nilai yang penting seperti kelurahan, koordinat x dan y, serta data gambar tersegmentasi yang dienkripsi dalam base64. Selanjutnya, data gambar tersebut didekode menggunakan *library* base64 dan disimpan sebagai file gambar PNG dengan nama sesuai dengan kelurahan yang terbentuk dari gabungan nilai-nilai kolom tertentu.

Dalam bagian utama script, terdapat pemanggilan fungsi *extract_image_pertile* dengan menyertakan nama file Kota_Bandung.csv yang akan diproses. Sebagai tambahan, script ini mengatur batas ukuran untuk file CSV dengan *csv*. Fungsi *field_size_limit* untuk menangani batasan ukuran default yang ada. Dengan menjalankan script ini, file gambar PNG akan dihasilkan untuk setiap baris. Gambar dari setiap baris tersebut merupakan gambar pertile dari tiap kelurahan.

Setelah mendapatkan gambar per tile langkah selanjut yaitu menggabungkan gambar. Proses penggabungan gambar segmentasi area hijau sama dengan proses penggabungan gambar kelurahan sebelum segmentasi. Pada proses tersebut dapat dilihat pada bab 3.1.3.

Dalam pencarian data gambar hasil segmentasi ditemukan data berupa file Bandung.txt. Setelah melakukan pemrosesan pada data file Bandung.txt, file tersebut merupakan data file dari Kabupaten Bandung. File Bandung.txt juga memiliki beberapa kolom yang menunjukkan nama kelurahan, nama kabupaten, nama provinsi, panjang tile, lebar tile, koordinat tile(x,y), data poligon, dan , tile citra satelit yang di encode menggunakan Base64. Proses pengecekan file tersebut dilakukan dengan menggunakan *script* python yang dikembangkan.

Kode 3.7: Script Pengecekan File .txt

```

1  #cek per satu kelurahan
2  file = open("Bandung.txt", "r+")
3  counter = 0
4  while True:
5      file_line = file.readline().split(";",8)
6
7      if not file.readline():
8          break
9
10     print(file_line[0],file_line[1],file_line[2],file_line[3],file_line[4],file_line[5],file_line[6],file_line[7])
11     counter += 1
12
13     print(counter)
14
15     file.close()

```

Setelah *script* 3.7 dijalankan ternyata terdapat data yang hilang. Sehingga saat melakukan pengekstraksian 3.3 gambar ada beberapa tile yang tidak hilang. Tile-tile yang hilang mengakibatkan gambar tidak dapat digabung menjadi sebuah gambar utuh dari kelurahan kabupaten. Contoh data dari Bandung.txt dapat dilihat pada gambar. Dari gambar tersebut bahwa nama kelurahan kabupaten adalah Kopo, dilanjutkan dengan nama kabupaten Bandung, provinsi Jawa Barat, lebar tile dan panjang tile. Pada kolom koordinat x dan y ada beberapa yang hilang. Data yang hilang adalah nilai x = 0, dan nilai y = 2 tidak terdapat pada kolom.

```

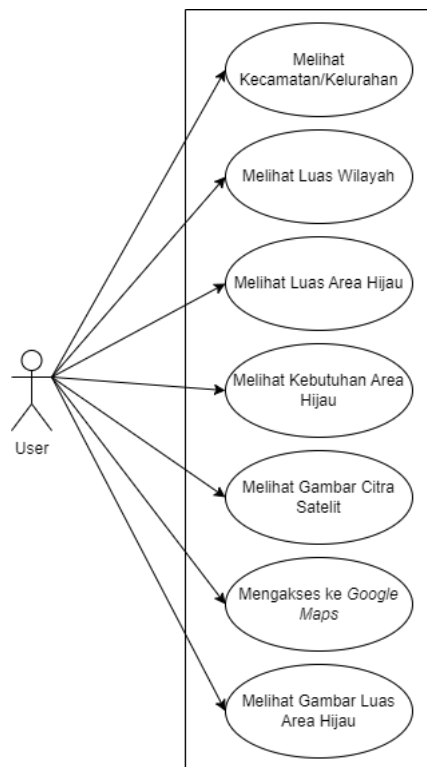
Kopo Bandung Jawa Barat ; ; 0 0
Kopo Bandung Jawa Barat ; ; 0 1
Kopo Bandung Jawa Barat ; ; 0 3
Kopo Bandung Jawa Barat ; ; 1 4
Kopo Bandung Jawa Barat ; ; 1 1
Kopo Bandung Jawa Barat ; ; 3 4
Kopo Bandung Jawa Barat ; ; 3 2
Kopo Bandung Jawa Barat ; ; 3 0
Kopo Bandung Jawa Barat ; ; 2 0
Kopo Bandung Jawa Barat ; ; 2 2
Kopo Bandung Jawa Barat ; ; 4 1
Kopo Bandung Jawa Barat ; ; 4 4
Kopo Bandung Jawa Barat ; ; 4 2

```

Gambar 3.5: Proses pengecekan file Bandung.txt

3.3 Analisis Perangkat Lunak

Proses analisis perangkat lunak merupakan kebutuhan yang memerlukan peranan seorang pengguna untuk menjalankan sebuah perangkat lunak yang akan dikembangkan. Sehingga segala proses sistem dijalankan oleh aktor yang terlibat. Dalam sistem ini hanya memiliki aktor sebagai *user*. Seorang pemangku kepentingan atau pembuat keputusan memegang peranan sebagai *user* itu sendiri. Dalam menggambarkan peranan pengguna terhadap interaksinya dengan sistem, maka dapat dilihat pada diagram *use case* yang terdapat pada Gambar 3.6 berikut.



Gambar 3.6: Diagram *Use Case User*

Pada Gambar 3.6, seorang aktor atau *user* pada sistem berperan dalam memegang akses penuh ke dalam sistem. Dalam hal ini *user* dapat masuk ke dalam sistem yang telah dibangun, dapat memilih kelurahan yang ingin dilihat. Setiap kelurahan yang dipilih *user* dapat melihat luas wilayah, luas area hijau, kebutuhan area hijau, gambar citra satelit/gambar luas area hijau, dan juga dapat mengakses ke halaman *Google Maps* yang merujuk ke lokasi kelurahan yang dipilih.

Berdasarkan diagram *use case* pada Gambar 3.6, berikut adalah daftar skenario untuk setiap *use case*:

1. Use Case: Melihat kelurahan

Actor: Pengguna

Pre Condition: Pengguna telah dapat mengakses website dan berada pada halaman utama website

Post Condition: Pengguna melihat informasi dari kelurahan dipilih

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna menekan pada dropdown kelurahan	Dropdown akan menampilkan daftar kelurahan
Pengguna dapat memilih salah satu kelurahan	Ditampilkan informasi dari kelurahan

2. Use Case: Melihat luas wilayah kelurahan

Actor: Pengguna

Pre Condition: Pengguna berada pada halaman utama website dan telah memilih kelurahan yang ingin dilihat

Post Condition: Pengguna dapat melihat luas wilayah kelurahan yang dipilih.

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna telah memilih kelurahan	Ditampilkan informasi tentang luas wilayah kelurahan

3. Use Case: Melihat luas area hijau kelurahan

Actor: Pengguna

Pre Condition: Pengguna telah dapat mengakses website dan berada pada halaman utama website

Post Condition: Pengguna melihat informasi dari luas area hijau kelurahan yang dipilih

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna telah memilih kelurahan	Ditampilkan informasi tentang luas wilayah kelurahan

4. Use Case: Melihat kebutuhan area hijau

Actor: Pengguna

Pre Condition: Pengguna telah dapat mengakses website dan berada pada halaman utama website

Post Condition: Pengguna melihat informasi dari kebutuhan area hijau kelurahan dipilih

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna telah memilih kelurahan	Ditampilkan informasi tentang kebutuhan area hijau kelurahan

5. Use Case: Melihat gambar citra satelit

Actor: Pengguna

Pre Condition: Pengguna telah dapat mengakses website dan berada pada halaman utama website

Post Condition: Pengguna melihat informasi berupa gambar kelurahan yang dipilih

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna telah memilih kelurahan	Ditampilkan gambar citra satelit kelurahan

6. Use Case: Mengakses ke *google maps*

Actor: Pengguna

Pre Condition: Pengguna telah dapat mengakses website dan berada pada halaman utama website

Post Condition: Pengguna melihat informasi berupa *link googlemaps* dari kelurahan yang dipilih

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna telah memilih kelurahan	Ditampilkan <i>link googlemaps</i> kelurahan

7. Use Case: Melihar gambar luas area hijau

Actor: Pengguna

Pre Condition: Pengguna telah dapat mengakses website dan berada pada halaman utama website

Post Condition: Pengguna melihat informasi dari kelurahan dipilih

Steps:

<i>Actor Actions</i>	<i>System Response</i>
Pengguna memilih <i>button</i> citra satelit	Ditampilkan informasi gambar dari kelurahan
Pengguna memilih <i>button</i> area hijau	Ditampilkan gambar area hijau dari kelurahan

DAFTAR REFERENSI

- [1] Kusjadi, J. A. (2022) Pengumpulan data citra satelit kelurahan dan perhitungan luas area hijau dengan teknologi big data. Skripsi. Universitas Katolik Parahyangan, Indonesia.
- [2] Shotts, W. (2019) *The Linux Command Line*, fifth internet edition edition. A LinuxCommand.org Book.
- [3] Alapati, S. R. (2016) *Expert Hadoop Administration: Managing, Tuning, and Securing Spark, YARN, and HDFS*, 1 edition. Addison-Wesley Professional, US.
- [4] Holmes, A. (2014) *Hadoop in Practice*, 2 edition In Practice. Manning Publications, US.
- [5] John Canning, R. L., Alan Broder (2022) *Data Structures and Algorithms in Python (Developer's Library)*, 1 edition. Addison-Wesley Professional.
- [6] Stauffer, M. (2019) *Laravel: Up and Running: A Framework for Building Modern PHP Apps*, 2 edition. O'Reilly Media.

LAMPIRAN A

KODE PROGRAM

Kode A.1: MyCode.c

```
1 // This does not make algorithmic sense,
2 // but it shows off significant programming characters.
3
4 #include<stdio.h>
5
6 void myFunction( int input, float* output ) {
7     switch ( array[i] ) {
8         case 1: // This is silly code
9             if ( a >= 0 || b <= 3 && c != x )
10                 *output += 0.005 + 20050;
11             char = 'g';
12             b = 2^n + ~right_size - leftSize * MAX_SIZE;
13             c = (--aaa + &daa) / (bbb++ - ccc % 2 );
14             strcpy(a,"hello_$@?");
15         }
16         count = ~mask | 0x00FF00AA;
17     }
18 }
19
20 // Fonts for Displaying Program Code in LATEX
21 // Adrian P. Robson, nepsweb.co.uk
22 // 8 October 2012
23 // http://nepsweb.co.uk/docs/progfonts.pdf
```

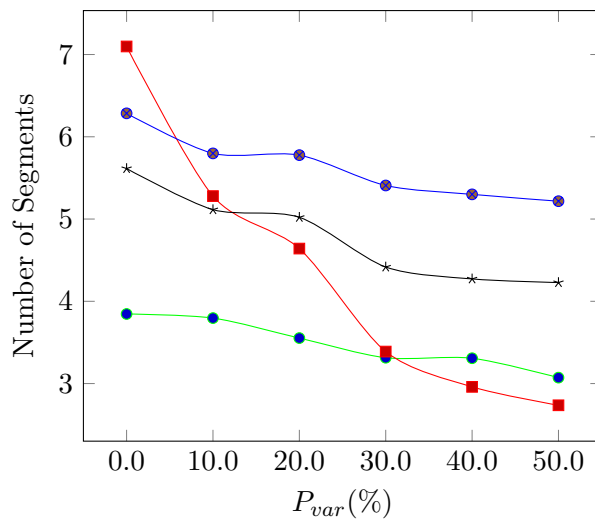
Kode A.2: MyCode.java

```
1 import java.util.ArrayList;
2 import java.util.Collections;
3 import java.util.HashSet;
4
5 //class for set of vertices close to furthest edge
6 public class MyFurSet {
7     protected int id; //id of the set
8     protected MyEdge FurthestEdge; //the furthest edge
9     protected HashSet<MyVertex> set; //set of vertices close to furthest edge
10    protected ArrayList<ArrayList<Integer>> ordered; //list of all vertices in the set for each trajectory
11    protected ArrayList<Integer> closeID; //store the ID of all vertices
12    protected ArrayList<Double> closeDist; //store the distance of all vertices
13    protected int totaltrj; //total trajectories in the set
14
15    /*
16     * Constructor
17     * @param id : id of the set
18     * @param totaltrj : total number of trajectories in the set
19     * @param FurthestEdge : the furthest edge
20     */
21    public MyFurSet(int id,int totaltrj,MyEdge FurthestEdge) {
22        this.id = id;
23        this.totaltrj = totaltrj;
24        this.FurthestEdge = FurthestEdge;
25        set = new HashSet<MyVertex>();
26        ordered = new ArrayList<ArrayList<Integer>>();
27        for (int i=0;i<totaltrj;i++) ordered.add(new ArrayList<Integer>());
28        closeID = new ArrayList<Integer>(totaltrj);
29        closeDist = new ArrayList<Double>(totaltrj);
30        for (int i = 0;i <totaltrj;i++) {
31            closeID.add(-1);
32            closeDist.add(Double.MAX_VALUE);
33        }
34    }
35 }
36 }
```


LAMPIRAN B

HASIL EKSPERIMEN

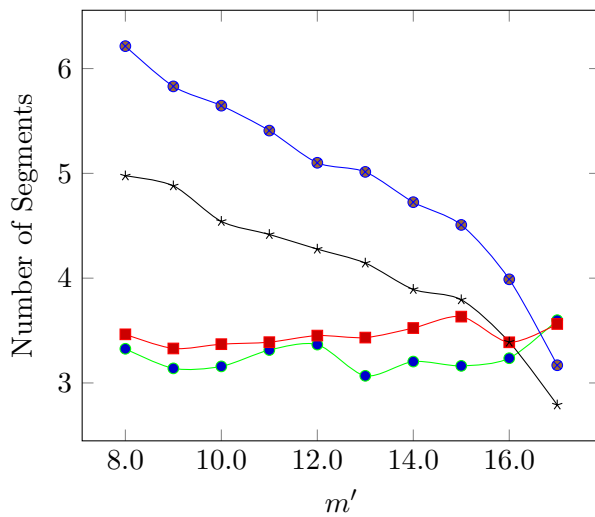
Hasil eksperimen berikut dibuat dengan menggunakan TIKZPICTURE (bukan hasil excel yg diubah ke file bitmap). Sangat berguna jika ingin menampilkan tabel (yang kuantitasnya sangat banyak) yang datanya dihasilkan dari program komputer.



Gambar B.1: Hasil 1



Gambar B.2: Hasil 2



Gambar B.3: Hasil 3



Gambar B.4: Hasil 4