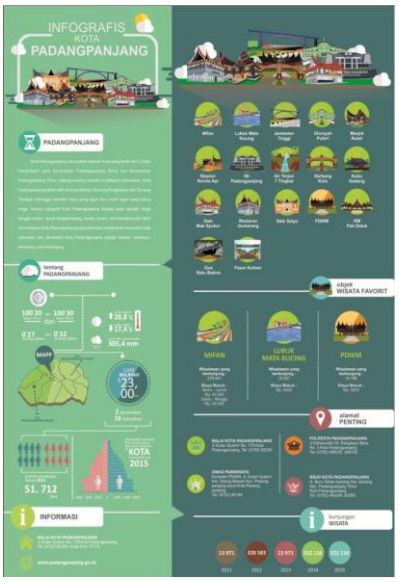
BAB II
LANDASAN TEORI

Bab ini akan membahas mengenai tinjauan pustaka yang berisi pustaka dan hasil penelitian yang pernah dilakukan, isi pustaka berhubungan dengan penelitian ini, kerangka pemikiran, serta landasan teori yang membahas teori-teori dasar pendukung untuk penelitian ini.

2.1 Tinjauan Pustaka

Sebagai pendukung penelitian ini digunakan beberapa tinjauan studi. Tinjauan studi yang digunakan adalah sebagai berikut :

Berdasarkan hasil penelitian dari yang pernah dilakukan (Ghifari, 2018), yang mengambil topik Perancangan infografis Kota Padangpanjang sebagai strategi kreatif promo wisata. Tahapan dari penelitian ini dimulai dari pra perancangan (observasi, wawancara, dokumentasi dan penjaringan ide), proses perancangan (sketsa manual ke digital, tipografi, tata *layout*), dan pasca perancangan (pengaplikasian berbagai media). Hasil penelitian membuktikan bahwa perancangan infografis Kota Padangpanjang menjadi sebuah solusi kreatif dalam mempromosikan wisata. Hasil perancangan infografis tersebut disajikan pada Gambar 2.1.



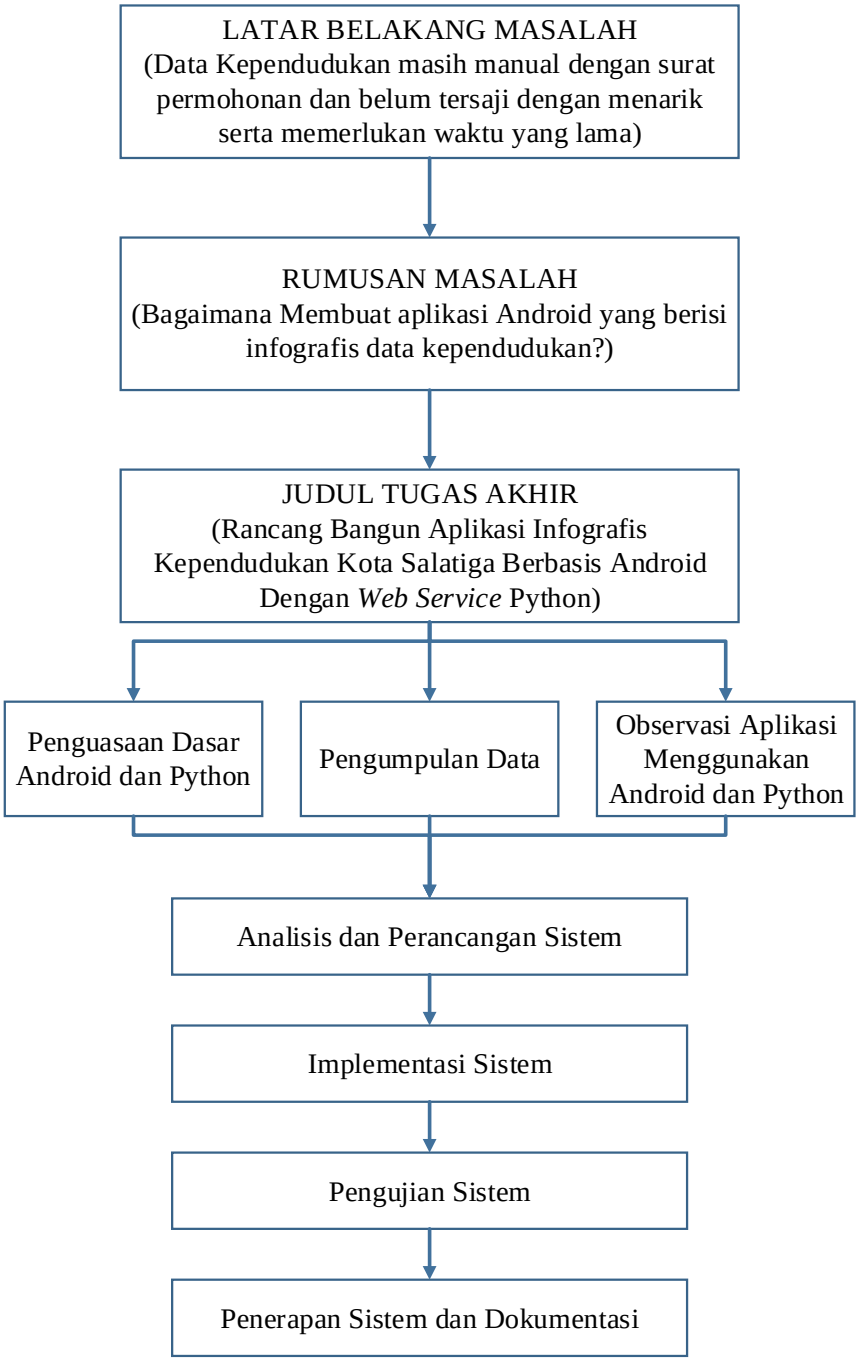
Gambar 2.1. Perancangan Infografis Kota Padangpanjang

Berdasarkan penelitian Resnatika, dkk. (2018), yang mengambil topik peran infografis sebagai media promosi dalam pemanfaatan perpustakaan. Penelitian ini menggunakan metode kuantitatif desktiptif dengan objek penelitian dilakukan di Perpustakaan Institut Teknologi Bandung. Populasi dari penelitian ini yaitu pengunjung Perpustakaan ITB yang jumlahnya diambil dari total rata-rata pengunjung perpustakaan ITB pada 3 bulan terakhir berjumlah 13.716 orang. Penentuan sampling menggunakan teknik *probability sampling* dengan kategori *simple random sampling*. Hasil penelitian ini dapat ditarik simpulan bahwa infografis sebagai media promosi Infografis sebagai media promosi di Perpustakaan ITB memiliki tiga indikator penelitian, diantaranya daya tarik infografis, kejelasan infografis, dan infografis yang mudah dipahami. Peran infografis sebagai media promosi dalam pemanfaatan perpustakaan dilihat dari aspek kemudahan dipahami dapat disimpulkan bahwa pesan informasi yang terdapat pada infografis mudah dipahami oleh pembacanya. Menggunakan infografis untuk memasarkan konten karena kemudahan dan kecepatan yang dimungkinkan dalam komunikasi. Hal tersebut tidak terlepas dari kemudahan informasi dipahami, kalimat dan bahasa yang mudah dipahami sehingga menjadi sebuah dorongan bagi para pemustaka membaca infografis untuk penambahan informasi seputar layanan perpustakaan ITB.

Berdasarkan penelitian yang pernah dilakukan oleh (Sulihati dan Andriyani, 2016), yang mengambil topik Aplikasi Akademik Berbasis *Mobile* Android pada Universitas Tama Jagakarsa. Peneliti memilih perangkat *mobile* Android dalam mengembangkan aplikasi dikarenakan Android semakin banyak diminati dan juga harga perangkatnya yang bervariasi serta dengan fitur yang cukup memuaskan. Metode *web service* berhasil diterapkan pada aplikasi layanan akademik untuk ponsel android, dengan memanfaatkan JSON(*Java Script Object Notation*) sebagai format pertukaran data yang memungkinkan lintas *platform* tanpa tergantung pada jenis aplikasi yang digunakan di sisi *client*. Penelitian ini menghasilkan sebuah aplikasi yang dapat digunakan untuk mengakses informasi akademik mahasiswa, seperti biodata mahasiswa, KRS, mata kuliah, jadwal, program studi, dan pengumuman.

2.2 Kerangka Pemikiran

Kerangka pemikiran yang menjelaskan alur proses penelitian dapat dilihat pada Gambar 2.2.



Gambar 2.2. Kerangka Pemikiran

Adapun penjelasan dari kerangka pemikiran yang digunakan dalam menyusun tugas akhir ini, dapat diuraikan sebagai berikut :

1) Latar belakang masalah

Data kependudukan digunakan oleh sebagian pengguna dan masih manual melakukan permintaan berupa surat permohonan, sedangkan ada tuntutan kemudahan akses di era keterbukaan informasi oleh seluruh lapisan masyarakat dalam bentuk penyajian yang menarik.

2) Rumusan masalah

Bagaimana membuat aplikasi yang menggabungkan teknologi ponsel pintar Android dan penyajian data kependudukan dalam bentuk infografis di Kota Salatiga?

3) Judul Tugas Akhir

Judul yang sesuai untuk menangani masalah yang dihadapi oleh Disdukcapil Kota Salatiga, Rancang Bangun Aplikasi Infografis Kependudukan Kota Salatiga Berbasis Android Dengan *Web Service* Python.

4) Pengumpulan Data

Semua data yang dibutuhkan dikumpulkan, baik melalui wawancara dengan pengurus barang organisasi perangkat daerah maupun dengan observasi langsung ke beberapa organisasi perangkat daerah di Pemerintah Kota Salatiga.

5) Penguasaan Dasar (Java, Android, Oracle, Python)

Mempelajari semua program (Java, Android, Oracle, Python) yang akan digunakan guna membangun sebuah aplikasi yang diinginkan.

6) Observasi Aplikasi

Mengadakan pengamatan terhadap aplikasi serupa yang sudah ada untuk dijadikan referensi pembuatan aplikasi yang akan dikerjakan.

7) Analisis dan Perancangan Sistem

Terhadap aplikasi yang akan dibangun dilakukan analisis dan perancangan seperti apa dan bagaimana desainnya serta apa saja *content* yang diinginkan. Analisis dan perancangan sistem menggunakan metode UML (*Unified Modeling Language*) yang terdiri dari *Use Case Diagram, Activity Diagram, Class Diagram, Sequence Diagram, Deployment Diagram dan Component Diagram*.

8) Implementasi Sistem

Implementasi sistem merupakan tahapan penerapan sistem berdasarkan hasil analisis dan perancangan sistem. Adapun tahapan dalam implementasi sistem sebagai berikut:

a. Perancangan *database* Oracle

Membuat *database* sesuai dengan kebutuhan sistem dari data-data yang diperoleh.

b. Perancangan *web service* Python

Membuat *web service* yang nantinya akan menjembatani *database* Oracle dengan aplikasi Android.

c. Perancangan aplikasi Android

Merancang aplikasi yang mudah digunakan dan sesuai dengan kebutuhan.

9) Pengujian Sistem

Pengujian aplikasi dilakukan bertujuan untuk mengetahui apabila ternyata masih terdapat kesalahan ataupun kekurangan pada aplikasi yang telah dikembangkan.

10) Penerapan Sistem dan Dokumentasi

Aplikasi telah siap untuk digunakan setelah melewati tahap pengujian dan kemudian membuat dokumentasi dari keseluruhan kegiatan penyusunan Tugas Akhir.

2.3 Landasan Teori

2.3.1 Aplikasi

Aplikasi adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna (Sanjaya, 2015). Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer, tapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna. Contoh utama perangkat lunak aplikasi adalah pengolah kata, lembar kerja, dan pemutar media. Beberapa aplikasi yang digabung bersama menjadi suatu paket kadang disebut sebagai suatu paket atau rangkaian aplikasi (*application suite*).

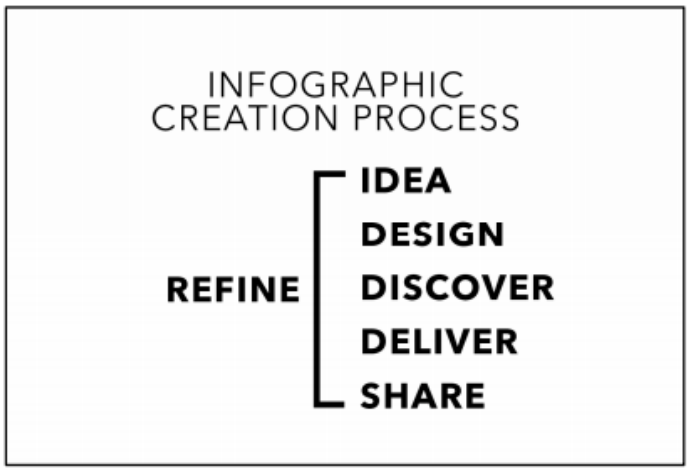
Contohnya adalah Microsoft Office dan Open Office.org, yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam suatu paket biasanya memiliki antarmuka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan setiap aplikasi. Sering kali, aplikasi ini memiliki kemampuan untuk saling berinteraksi satu sama lain sehingga menguntungkan pengguna.

2.3.2 Infografis

Infografis sendiri berasal dari kata infographics dalam Bahasa Inggris yang merupakan singkatan dari information + graphics. Kata ini merujuk pada bentuk visualisasi data yang menyampaikan informasi kompleks kepada pembaca agar dapat dipahami dengan lebih mudah dan cepat (Kementerian Komunikasi dan Informatika, Republik Indonesia, 2018). Bentuk dari infografis adalah sejenis gambar yang memadukan data dengan desain dengan tujuan agar dapat membantu individu dan organisasi berkomunikasi dengan audiens mereka.

Infografis berkembang pesat dalam media massa setelah desainer dapat mengkombinasikan antara informasi dari ranah berita ke dalam bentuk visual yang dicetak maupun yang dipublikasikan dalam jaringan internet. Infografis dipublikasikan dalam dua format yakni infografis statis (atau biasa disebut infografis) dan infografis dinamis (*motiongraphic*). Infografis statis berbentuk gambar statis yang berisi tabel, grafis (ilustrasi dan gambar) dan teks. Sedangkan motiongraphic berbentuk video yang berisi komposisi visual bergerak (animasi) dari elemen-elemen infografis dan memiliki alur yang mudah dimengerti.

Dalam menciptakan infografis dibutuhkan 6 proses yaitu ide, desain, penemuan, penyampaian, berbagi, penyempurnaan. (Murphy, 2018). Gambar proses menciptakan infografis ditunjukkan dalam Gambar 2.3.



Gambar 2.3. Proses dalam menciptakan infografis
(Murphy, 2018)

Infografis yang disusun untuk tujuan penyampaian informasi pemerintah biasanya dapat dibagi menjadi empat kategori yaitu: Informasi Harian (*Daily Issue*), Program Pemerintah, Edukasi dan Informasi Publik, dan Tentang Indonesia. Keempat kategori besar ini memerlukan tujuan spesifik yang lebih mendalam dari sekadar memberikan informasi kepada publik. Tiap kategori memerlukan metode tersendiri dalam menyusun konten informasi yang dimuat dalam infografis termasuk segmen spesifik dan gaya visual yang disajikan. Keempat kategori ini juga membutuhkan rangkaian publikasi yang terjadwal agar dapat diukur keberhasilannya mencapai target audien. Penjadwalan inilah yang menggiring pembaca pada repetisi kategori dan memudahkan pembaca untuk mengenal konten informasi apa saja yang akan disajikan oleh akun sosial media yang digunakan untuk menyampaikan informasi pemerintah.

2.3.3 Data Kependudukan

Data Kependudukan adalah data perseorangan dan/atau data agregat yang terstruktur sebagai hasil dari kegiatan Pendaftaran Penduduk dan Pencatatan Sipil. Data Kependudukan terdiri atas data perseorangan dan/atau data agregat Penduduk. Data agregat meliputi himpunan data perseorangan yang berupa data kuantitatif dan data kualitatif. Data Kependudukan yang digunakan untuk semua keperluan adalah

Data Kependudukan dari Kementerian yang bertanggung jawab dalam urusan pemerintahan dalam negeri, antara lain untuk pemanfaatan:

1) pelayanan publik;
2) perencanaan pembangunan;
3) alokasi anggaran;
4) pembangunan demokrasi; dan
5) penegakan hukum dan pencegahan kriminal.

Penyajian Data Kependudukan berskala kabupaten/kota berasal dari Data Kependudukan yang telah dikonsolidasikan dan dibersihkan oleh Kementerian yang bertanggung jawab dalam urusan pemerintahan dalam negeri. Data Kependudukan skala kabupaten/kota diterbitkan secara berkala per semester, yaitu untuk semester pertama yang diterbitkan tanggal 30 Juni dan semester kedua yang diterbitkan tanggal 31 Desember (Republik Indonesia, 2013).

2.3.4 Android

Android adalah sebuah sistem operasi untuk perangkat mobile berbasis Linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam piranti bergerak. Android dirilis perdana pada tanggal 5 November 2007, tidak hanya menjadi sistem operasi di *smartphone*, namun juga pada perangkat lain seperti *wearable device*, tablet PC. Saat ini Android menjadi pesaing utama dari sistem operasi *smartphone* milik Apple Inc. yaitu iOS. Pesatnya pertumbuhan Android disebabkan oleh sistem operasinya memiliki fitur yang sangat lengkap, selain fitur juga dalam hal *tool* pengembangannya, bursa aplikasi Android, serta dukungan yang tinggi dari komunitas Open Source di dunia, sehingga Android berkembang pesat baik dari segi teknologi maupun dari segi jumlah *device* yang ada dunia.

Aplikasi Android dapat dikembangkan dengan berbagai macam tool pengembangan, tool yang resmi dirilis oleh pihak Google adalah Android Studio. Android Studio adalah *Integrated Development Environment* (IDE) untuk mengembangkan aplikasi Android. Android Studio berbasis pada “IntelliJ IDEA”

Java-IDE dari Jetbrains dan diperkenalkan oleh Google. Android Studio ini diumumkan pada Mei 2013. Android Studio menggantikan Eclipse sebagai IDE resmi untuk mengembangkan aplikasi pada *platform* Android dengan template yang tersedia yang memudahkan membuat aplikasi dan kelas baru dan memudahkan dalam menjalankan dan mem-*package*-nya menjadi aplikasi siap *install* (Dawn dan David, 2015). Android Studio memiliki beberapa fitur baru dibandingkan dengan Eclipse, diantaranya adalah :

1) Menggunakan Gradle-based *build* sistem yang fleksibel.
2) Bisa melakukan *build* pada beberapa APK.
3) Layout editor yang lebih bagus.
4) Built-in support untuk Google Cloud Platform, sehingga mudah untuk integrasi dengan Google Cloud Messaging dan App Engine.
5) Import *library* langsung dari Maven *repository*.
6) Diperluas dukungan template untuk Layanan Google dan berbagai jenis perangkat.

2.3.5 Java

Java merupakan bahasa pemrograman yang dikembangkan oleh James Gosling berserta tim di Sun Microsystem pada tahun 1991. Awalnya Java disebut dengan "Oak". Namun pada tahun 1995, nama "Oak" diganti menjadi Java. Bahasa pemrograman Java dirancang untuk menjadi bahasa pemrograman *multi-platform* yang cukup aman dan tangguh. Java memiliki beberapa karakeristik yaitu simpel, berorientasi *object, high performance, multithreaded, dinamis, intepreted,* serta *portabel* (Liang, 2018). Pada mulanya Java biasa digunakan untuk mengembangkan aplikasi berbasis desktop yang lebih dikenal dengan nama J2SE. Kemudian muncul dua versi berikutnya yaitu J2EE yang diarahkan untuk mengembangkan aplikasi skala besar, aplikasi berbasis *network*, dan aplikasi berbasis *web*; dan J2ME yang diarahkan untuk mengembangkan aplikasi pada *device* yang kecil dan terbatas memorinya, misalnya pada perangkat Blackberry dan perangkat berbasis Symbian. Pada tahun 2007, J2ME dijadikan basis untuk mengembangkan sistem operasi Android oleh Google. Pada tahun 2010, perusahaan Oracle membeli Sun

Microsystem dan menjadi perusahaan penyedia layanan untuk bahasa pemrograman Java. Dalam awal masa kepemilikannya, Oracle mengajukan tuntutan pada Google atas penggunaan teknologi J2ME pada *platform* Android, namun mereka dinyatakan kalah di pengadilan.

2.3.6 *Web service*

W3C mendefinisikan *web service* sebagai sebuah *software* aplikasi yang dapat teridentifikasi oleh URI dan memiliki *interface* yang didefinisikan, dideskripsikan, dan dimengerti oleh XML atau JSON dan juga mendukung interaksi langsung dengan *software* aplikasi yang lain dengan menggunakan *message* berbasis XML atau JSON melalui protokol internet. *Web service* adalah sebuah *software* aplikasi yang tidak terpengaruh oleh *platform*, menyediakan *method* yang dapat diakses oleh network (Barry dan Dick, 2018). *Web service* juga akan menggunakan XML untuk pertukaran data, khususnya pada dua *entities bisnis* yang berbeda. Beberapa karakteristik dari *web service* adalah:

1) *Message-based*
2) *Standards-based*
3) *Programming language independent*
4) *Platform-neutral*

Beberapa *key* standard di dalam *web service* adalah: JSON, XML, SOAP, WSDL and UDDI.

2.3.7 JSON

JSON (JavaScript Object Notation) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (generate) oleh komputer (Rischpater, 2015). Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa

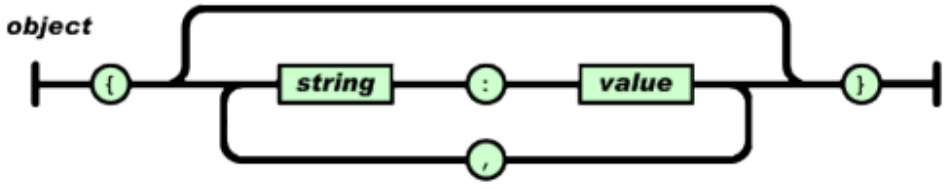
pertukaran-data. JSON terbuat dari dua struktur:

1) Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.
2) Daftar nilai terurutkan (an ordered list of values). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (array), vektor (vector), daftar (list), atau urutan (sequence).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini. JSON menggunakan bentuk sebagai berikut:

1) Objek

   Objek adalah sepasang nama / nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/nilai dipisahkan oleh , (koma). Objek biasanya digunakan untuk menyimpan data tunggal dalam bentuk JSON. Objek dalam JSON disajikan pada Gambar 2.4.



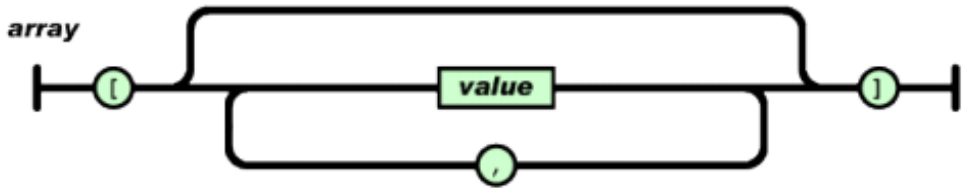
Gambar 2.4. Objek JSON

(Sumber : https://www.web3d.org)

2) *Array* (Larik)

   Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma). Larik dalam JSON dapat digunakan sebagai value

dari JSON object hal ini dapat berguna jika JSON menyimpan data bertingkat. Array dalam JSON disajikan pada Gambar 2.5.



Gambar 2.5. Array JSON

(Sumber : https://www.web3d.org)

Bentuk data JSON objek dan larik dapat saling dikombinasikan untuk mendukung struktur data yang lebih kompleks. JSON mendukung beberapa tipe data untuk menjadi *value* seperti Angka, *String*, *Boolean* dan nilai *Null*.

2.3.8 *Database*

Database adalah kumpulan-kumpulan data-data yang saling terkait (Elmasri dan Navathe, 2016). Definisi *database* yang cukup umum, misalnya, kita dapat mempertimbangkan koleksi kata-kata yang membentuk halaman teks menjadi data terkait dan karenanya untuk terbentuklah *database*. Namun, penggunaan umum dari istilah *database* lebih terbatas.

2.3.8.1 *Database Management System*

Database Management System (DBMS) adalah kumpulan program yang memungkinkan pengguna untuk membuat dan memelihara *database*. DBMS adalah tujuan umum dari sistem perangkat lunak yang memfasilitasi proses-proses mendefinisikan, membangun, memanipulasi, dan berbagi *database* antara berbagai pengguna dan aplikasi. Mendefinisikan *database* meliputi menentukan jenis data, struktur, dan keterbatasan dari data yang akan disimpan dalam *database*. Definisi *database* atau informasi deskriptif juga disimpan oleh DBMS dalam bentuk katalog *database* atau kamus yang disebut meta-data. Membangun *database* adalah proses menyimpan data pada beberapa media penyimpanan yang dikendalikan oleh DBMS. Manipulasi *database* termasuk fungsi seperti *query database* untuk

mengambil data tertentu, memperbarui *database* mencerminkan perubahan dalam *miniworld*, dan menghasilkan laporan dari data. Berbagi *database* memungkinkan beberapa pengguna dan program untuk mengakses *database* secara bersamaan (Elmasri dan Navathe, 2016)

2.3.8.2 SQL

SQL adalah bahasa *database* yang komprehensif yaitu memiliki pernyataan untuk mendefinisikan data, *query*, dan *update*. Awalnya, SQL atau *Structured Query Language* disebut SEQUEL (*Structured Query Language English*) dan dirancang dan diterapkan di IBM Research sebagai antarmuka untuk eksperimental sistem *database* relasional yang disebut SISTEM R. SQL sekarang merupakan bahasa standar untuk DBMS relasional komersial (Elmasri dan Navathe, 2016).

2.3.8.3 Oracle

Tiga dekade yang lalu Larry Ellison melihat ada kesempatan emas saat dia mengerjakan deskripsi *protoype database* relasional. Bersama dua orang pendiri lainnya yaitu Bob Miner dan Ed Oates mereka mengkomersilkan model *database* relasional menggunakan SQL. Perkembangan permrograman yang dikeluarkan oleh produk Oracle sangatlah cepat dan karena perkembangannya tersebut, maka setiap perusahaan banyak menggunakan jasa dari perusahaan Oracle ini. Mulai hanya penggunaan *database*-nya saja, membuat aplikasi dan bahkan support juga untuk aplikasi *web*. Saat ini Oracle telah menjadi standar khusus untuk pembangunan aplikasi dan *database* enterprise. Oracle merupakan sebuah basis data relasional atau RDBMS (*Relational Database Management System*) yang sangat popular dan saat ini banyak digunakan oleh perusahaan perusahaan besar bersekala internasional. Keandalan Oracle dalam melakukan pengolahan dan pemeliharaan data sudah tidak diragukan lagi dikalangan praktisi yang bergelut di dunia pemrograman *database*. Oracle memiliki sistem keamanan yang tinggi sehingga data-datanya dapat terjaga dengan baik (Elmasri & Navathe, 2016).

2.3.9 Python

Python adalah bahasa pemrograman multiguna dengan filosofi perancangan yang berfokus pada tingkat keterbacaan kode. Python diklaim sebagai bahasa yang menggabungkan kapabilitas, kemampuan, dengan sintaksis kode yang sangat jelas, dan dilengkapi dengan fungsionalitas pustaka standar yang besar serta komprehensif (Syahrudin dan Kurniawan, 2018). Python mendukung multi paradigma pemrograman, utamanya, namun tidak di batasi pada pemrograman berorientasi objek, pemrograman imperatif, dan pemrograman fungsional. Salah satu fitur yang tersedia pada Python adalah sebagai pemrograman dinamis yang dilengkapi skrip meski pada praktiknya penggunaan bahasa ini lebih luas mencakup konteks pemanfaatan yang umumnya tidak dilakukan dengan menggunakan bahasa skrip. Python dapat digunakan untuk berbagai keperluan pengembang perangkat lunak dan dapat berjalan di berbagai *platform* sistem operasi, beberapa diantaranya Linux/Unix, Windows, Mac OS X, Java Virtual Machine, OS/2, Palm.

Beberapa fitur yang dimiliki pemrograman Python adalah :

1) Memiliki kepustakaan yang luas, dalam distribusi Python telah disediakan modul-modul siap pakai untuk berbagai keperluan.
2) Memiliki tata bahasa yang jernih dan mudah dipelajari
3) Memiliki aturan layout kode sumber yang memudahkan pengecekan, pembacaan kembali dan penulisan ulang kode sumber.
4) Berorientasi objek
5) Modular, mudah dikembangkan dengan menciptakan modul-modul tersebut dapat dibangun dengan bahasa Python maupun C/C++.
6) Memiliki fasilitas pengumpulan sampah otomatis, seperti halnya pada bahasa pemrograman Java, Python memiliki fasilitas pengaturan penggunaaan ingatan komputer para pemrograman tidak perlu melakukan pengaturan ingatan komputer secara langsung. Memiliki banyak fasilitas pendukung sehingga mudah dalam pengoperasikannya.

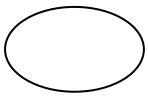
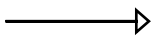
2.3.10 UML

UML (*Unified Modeling Language*) merupakan bahasa *visual* untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks khusus (Rosa A. dan Salahuddin, 2018). Penjelasan tentang masing-masing diagram akan dilakukan pada submodul-submodul berikutnya:

2.3.10.1 *Use Case Diagram*

Use Case Diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungi-fungsi itu (Rosa A. dan Salahuddin, 2018). Simbol *Use Case Diagram* disajikan pada Tabel 2.2.

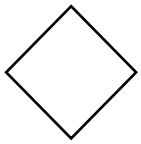
Tabel 2.1. *Simbol Use Case Diagram*

No	Simbol	Nama	Keterangan
1		*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
2		*Use Case*	Fungsional yang disediakan sistem sebagai unit-unit yang saling betukar pesan antar unit atay aktor.
3		*Association*	Komunikasi antara aktor dan *use case* yang berpartisipasi *pada use case* atau *use case* memiliki interaksi dengan *actor*.
4		*Generalization*	Hubungan generalisasi dan spesialisasi antara dua buah *use case* dimana fungsi yang satu fungsi yang lebih umum dari lainnya.

2.3.10.2 *Activity Diagram*

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. *Activity diagram* menggambarkan aktivitas sistem bukan apa yang akan dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Simbol *activity diagram* disajikan pada Tabel 2.3.

Tabel 2.2. *Activity Diagram*

No	Simbol	Nama	Keterangan
1		Status Awal	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2		Aktivitas	Aktivitas yang dilakukan sistem biasanya diawali dengan kata kerja.
3		*Decision*	Asosiasi percabangan dimana jika ada pilian aktivitas lebih atau satu.
4		Penggabungan atau *Join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
		Status Akhir	Status akhir yang dilakukan sistem.

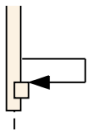
2.3.10.3 *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu (Hendini, 2016). *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut,

proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan. Simbol untuk *sequence diagram* disajikan pada Tabel 2.4.

Tabel 2.3. Simbol *Sequence Diagram*

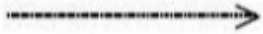
NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Simbol yang Pertama ini bernama *Actor* atau dalam bahasa indonesia nya aktor.
2		*Entity*	*Entity class* merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran sistem.
3		*Boundary*	*Boundary class*, berisi kumpulan kelas yang menjadi *interfaces* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan *form entry* dan *form* cetak.
5		*Message*	Simbol mengirim pesan antar *class.*
6		*Recursive*	Menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
7		*Activation*	Mewakili sebuah eksekusi operasi dari objek.
8		*Lifeline*	Garis yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*

2.3.10.4 *Class Diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut atau properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode atau fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. Class memiliki tiga area pokok yaitu Nama (dan *stereotype*), Atribut, Metode.

Atribut dan metode dapat memiliki salah satu sifat *private* (tidak dapat dipanggil dari luar *class* yang bersangkutan), *protected* (hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya), *public* (dapat dipanggil oleh siapa saja). Simbol relasi dalam *class diagram* disajikan pada Tabel 2.5.

Tabel 2.4. Simbol relasi *class diagram.*

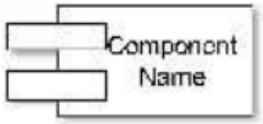
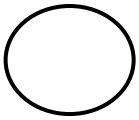
NO	GAMBAR	NAMA	KETERANGAN
1		*Association* atau asosiasi	Relasi antar kelas dengan makna umum
2		*directed association* atau asosiasi berarah	Relasi anar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain.
3		*generalization* atau generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus).s
4		*dependency* atau kebergantungan	Relasi anta kelas dengan makna kebergantungan antar kelas.
5		*agregation* atau agregasi	Relasi antar kelas dengan makna semua-bagian (*whole-part*).

2.3.10.5 *Component Diagram*

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya. Komponen piranti lunak adalah modul berisi kode, baik berisi source code maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil.

Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain (Rosa A. dan Salahuddin, 2018). Simbol-simbol *Component Diagram* disajikan pada Tabel 2.6.

Tabel 2.5. Simbol-simbol *Component Diagram*

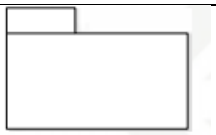
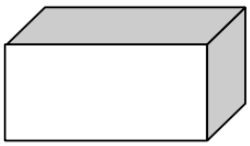
NO	GAMBAR	NAMA	KETERANGAN
1		*Package*	Merupakan sebuah bungkusan dari satu atau lebih komponen.
2		*Component*	Komponen sistem.
3		*Dependency* atau kebergantungan	Kebergantungan antar node, arah panah mengarah pada node yang dipakai.
4		*Link*	Relasi antar *node*.
5		Antarmuka atau *Interface*	Relasi antar komponen

2.3.10.6 *Deployment Diagram*

Deployment atau *physical diagram* menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, *server* atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi *server*, dan hal-hal lain yang bersifat fisikal

Sebuah *node* adalah *server*, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya. Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini. *Deployment Diagram* disajikan pada Tabel 2.7.

Tabel 2.6. Simbol-simbol *Deployment Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Package*	Merupakan sebuah bungkusan dari satu atau lebih *node*.
2		*Node*	Biasanya mengacu pada perangkat keras (*hardware*), perangkat lunak yang tidak dibuat sendiri (*software*). .
3		*Dependency* atau kebergantungan	Kebergantungan antar *node*, arah panah mengarah pada *node* yang dipakai.
4		*Link*	Relasi antar *node*.

2.3.11 Pengujian *BlackBox*

Pengujian menggunakan sekumpulan aktifitas validasi, dengan pendekatan *blackbox testing*. *Blackbox testing* adalah menguji perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program (Rosa A. dan Salahuddin, 2018). Pengujian dimaksudkan untuk mengetahui apakah fungsi-

fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian *blackbox* dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai spesifikasi yang dibutuhkan.

Tujuan dari pengujian adalah untuk menemukan dan memperbaiki sebanyak mungkin kesalahan dalam program sebelum menyerahkan program kepada *customer* (Pressman, 2015). Salah satu pengujian yang baik adalah pengujian yang memiliki probabilitas tinggi dalam menemukan kesalahan. *Blackbox testing* berfokus pada persyaratan fungsional perangkat lunak yang memungkinkan engineers untuk memperoleh set kondisi input yang sepenuhnya akan melaksanakan persyaratan fungsional untuk sebuah program.

2.3.12 Pengujian Kuesioner

Kuisioner adalah suatu teknik pengumpulan informasi yang memungkinkan analis mempelajari sikap-sikap, keyakinan, perilaku, dan karakteristik beberapa orang utama di dalam organisasi yang bisa terpengaruh oleh sistem yang diajukan atau oleh sistem yang sudah ada. Dengan menggunakan kuisioner, penulis berupaya mengukur pengalaman pengguna terhadap apa yang ditemukan dalam sistem (Sugiyono, 2017).

Penelitian ini menggunakan kuesioner untuk melakukan pengujian. Kuesioner dianggap sebagai media penilaian paling tepat untuk menjalankan pengujian usability untuk perangkat lunak. Kuesioner merupakan daftar pertanyaan tertulis yang diberikan kepada subjek yang diteliti untuk mengumpulkan informasi yang dibutuhkan penulis. Menggunakan kuesioner bertujuan untuk mendapatkan penilaian saat proses pengujian.

Kuesioner pada umumnya bermodel tabel. Kuesioner yang terdiri dari baris dan kolom, pada kolom pertama berisi pernyataan yang sesuai dengan kebutuhan penilaian, kemudian kolom selanjutnya berisi tentang skala nilai untuk mengetahui nilai dari setiap pernyataan yang disajikan. Kuesioner biasanya dibentuk dalam skala lima poin dengan model skala Likert untuk memilih jawaban sebagai pengukuran tingkat persetujuan pengguna terhadap pernyataan.