BAB III
PERANCANGAN PENELITIAN

3.1 Peralatan Pendukung

Peralatan pendukung dalam pembuatan aplikasi berbasis *website* terdiri dari perangkat keras (*hardware*) dan perangkat lunak (*software*) untuk mendukung berjalannya perancangan dan pembuatan.

3.1.1 Perangkat Keras

Adapun perangkat keras yang dibutuhkan dalam merancang dan membuat *bussines logic* dari *website* perencanaan program *tourism*:

1. *Processor* intel *core* i5
2. *Memory* 8GB.
3. *Hardisk* minimal 250 GB.
4. Monitor, *Keyboard* dan *Mouse*.

3.1.2 Perangkat Lunak

Adapun perangkat lunak yang dibutuhkan dalam merancang dan membuat *bussines logic* dari *website* perencanaan program *tourism*:

1. Sistem operasi *windows* 10
2. *Web Server* : *XAMPP*
3. *Web Browser.*
4. Bahasa Pemrograman : PHP, *Javascript*, HTML, CSS
5. *Database Server* : *MySQL*

3.2 Analisis Kebutuhan

Analisis kebutuhan didapat dari Angket dan wawancara. Berikut merupakan analisis kebutuhan pada aplikasi:

a. Halaman Akun untuk *member*.

b. Halaman Wisata untuk melihat daftar tempat wisata.

c. Halaman Detail Wisata untuk melihat informasi detail tempat wisata.

d. Halaman Hotel untuk melihat daftar hotel.

e. Halaman Detail Hotel untuk melihat informasi detail hotel.

f. Halaman Transportasi untuk melihat daftar Transportasi.

g. Halaman *Login* untuk *member*.

h. Halaman Beranda.

i. Halaman Registrasi untuk non *member* melakukan registrasi.

j. Halaman Buat Rencana untuk membuat rencana berlibur.

k. Halaman Detail Rencana untuk menampikan rencana berlibur yang telah dibuat.

l. Halaman Riwayat Rencana untuk *member* mengelola rencana berlibur yang telah dibuat.

m. Fitur *Map* untuk menampilkan letak koordinat dari tempat wisata dan juga hotel.

n. Fitur *Print* untuk mencetak rencana berlibur yang telah dibuat.

o. Fitur *Likes* untuk mengetahui tempat wisata paling favorit.

3.3 Rancangan

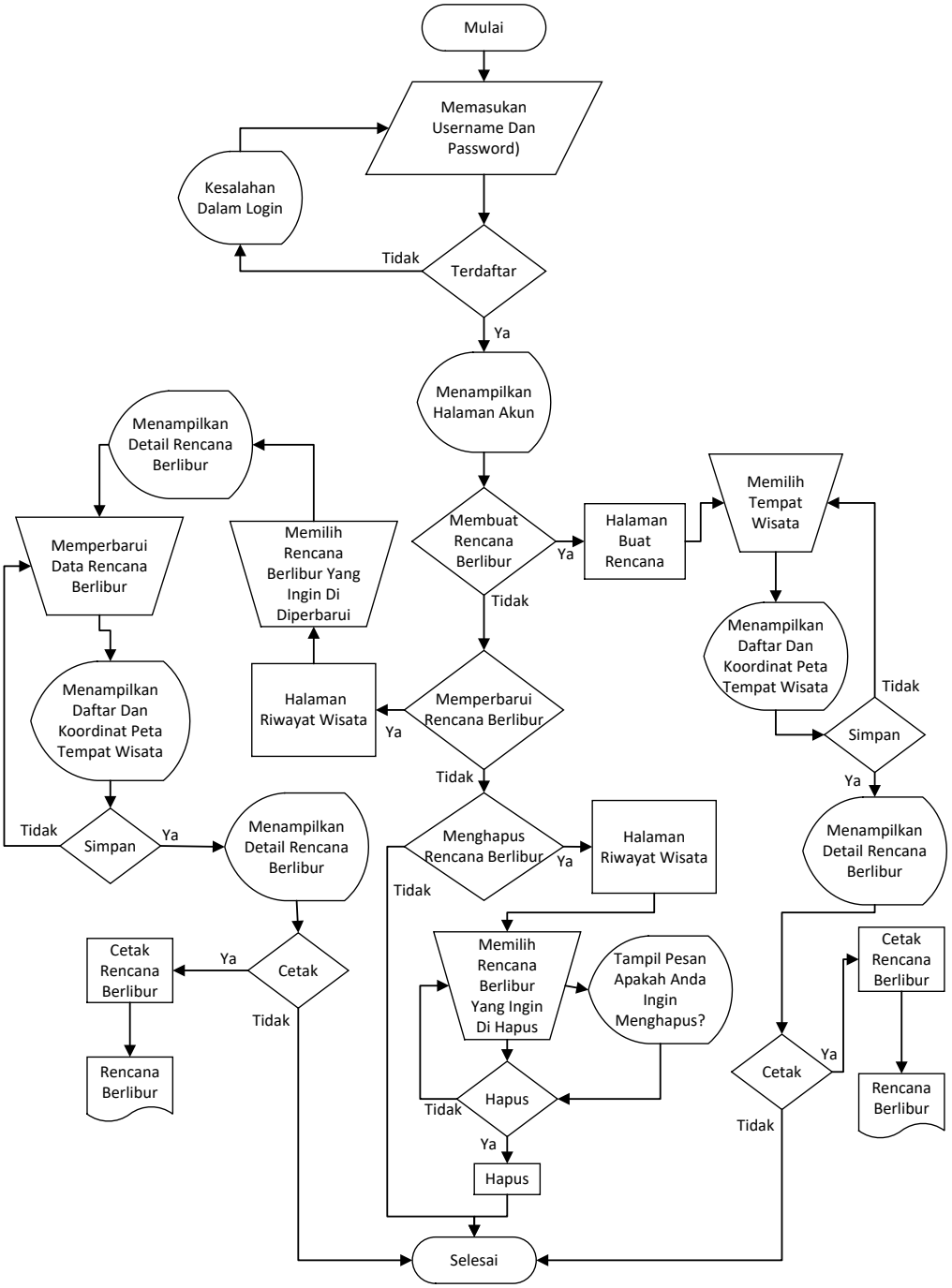
Dalam perancangan *Logic* Aplikasi ini menggunakan metode Bagan Alir atau *Flowchart* dan *United Markup Language* (UML). Model UML yang digunakan dalam pengembangan aplikasi yaitu *Use case* diagram, *Activity* diagram, dan *Class* Diagram.

3.3.1 Flowchart

Berdasarkan analisis kebutuhan dibuatlah *flowchart* untuk membantu perancangan *logic* aplikasi. Berikut penjelasan *Flowchart* tentang Gambar 3.6:

1. Member harus *login* terlebih dahulu agar dapat menggunakan fitur halaman buat rencana. Jika *login* sukses maka akan muncul halaman akun. Namun jika terjadi kesalahan dalam proses *login*, maka akan kembali ke halaman *login*.
2. Ketika *member* selesai *login* maka akan tampil halaman akun. Di halaman ini *member* dapat melanjutkan ke halaman yang sesuai dengan kebutuhannya.
3. Jika *member* memutuskan untuk membuat rencana berlibur maka *member* dapat menuju halaman buat rencana, kemudian memilih tempat wisata, setelah itu akan tampil daftar tempat wisata dan juga koordinat tempat wisata yang telah dipilih. Ketika *member* menyimpan rencana berliburnya maka akan tampil detail rencana berlibur yang telah dibuat. Kemudian *member* dapat memutuskan untuk mencetak atau tidak rencana berlibur tersebut.
4. Jika *member* memutuskan untuk memperbarui rencana berlibur maka *member* dapat langsung menuju ke halaman riwayat wisata, kemudian memilih rencana berlibur yang ingin diperbarui. kemudian tampil halaman detail rencana. proses selanjutnya sama ketika *user* akan membuat rencana baru.

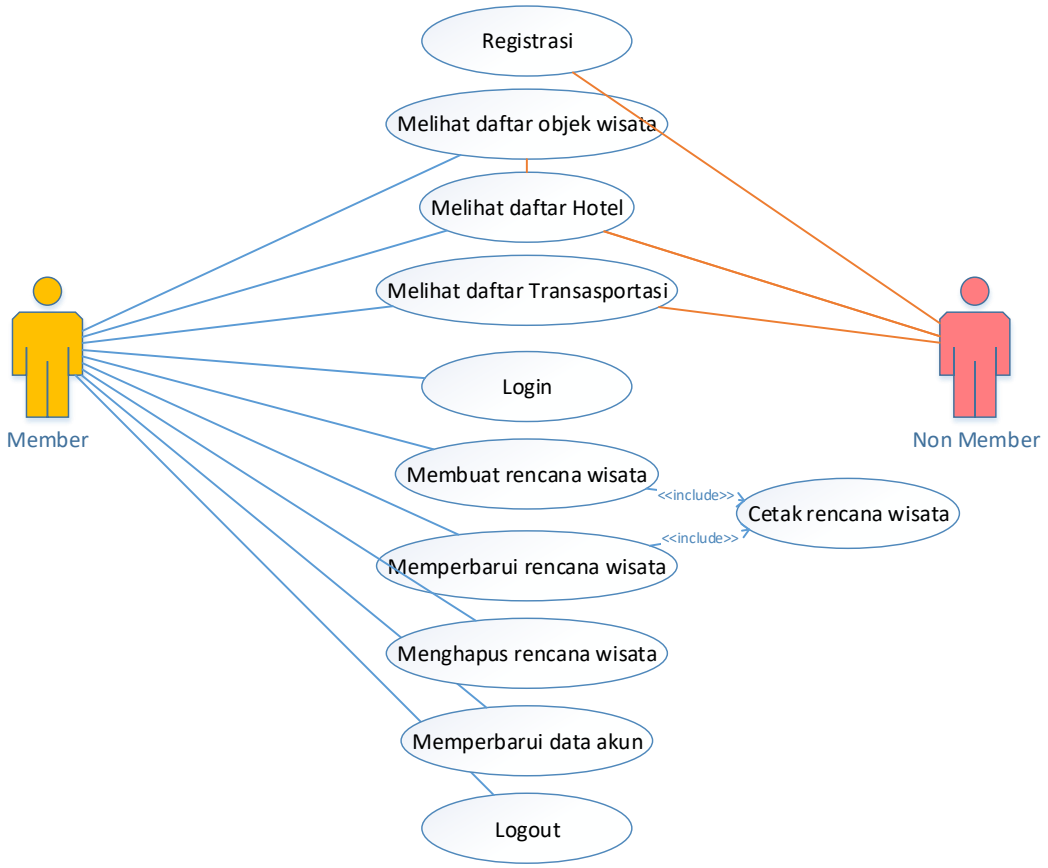
5. Jika *member* memutuskan untuk menghapus rencana berlibur yang sebelumnya telah dibuat maka *member* dapat langsung menuju ke halaman riwayat wisata, kemudian memilih rencana berlibur yang ingin dihapus.



Gambar 3.1 *Flowchart Member* Aplikasi

3.3.2 Use Case Diagram

Berdasarkan analisis kebutuhan dibuatlah *use case* diagram untuk membantu perancangan *logic* aplikasi. Terdapat 2 aktor yaitu *member* dan non *member*. Gambaran *Use Case* diagram yang digunakan dalam aplikasi dapat dilihat pada gambar 3.2.



Gambar 3.2 *Use Case Diagram* Aplikasi

Berikut penjelasan tentang *use case* diagram pada Gambar 3.2:

a. Registrasi : Memungkinkan non *member* untuk melakukan registrasi agar menjadi *member*.

b. Melihat Daftar objek wisata : Memungkinkan *member* untuk melihat kumpulan daftar objek wisata yang dapat kunjungi.

c. Melihat Daftar hotel : Memungkinkan *member* untuk melihat kumpulan daftar hotel.
d. Melihat Daftar transportasi : Memungkinkan *member* untuk melihat kumpulan daftar transportasi yang dapat di pesan.
e. *Login* : Memungkinkan *member* untuk dapat menggunakan halaman buat rencana.
f. Membuat rencana wisata : Memungkinkan *member* untuk membuat rencana berlibur.
g. Memperbarui rencana wisata : Memungkinkan *member* untuk memperbarui rencana berlibur yang sebelumnya telah dibuat.
h. Menghapus rencana wisata : Memungkinkan *member* untuk menghapus rencana berlibur yang sebelumnya telah dibuat.
i. Cetak rencana wisata : Memungkinkan *member* untuk mencetak rencana berlibur.
j. Memperbarui data akun : Memungkinkan *member* untuk memperbarui data akun seperti data diri, *username*, dan *password*.
k. *Logout* : Memungkinkan *member* untuk tidak lagi menggunakan fitur halaman buat rencana.

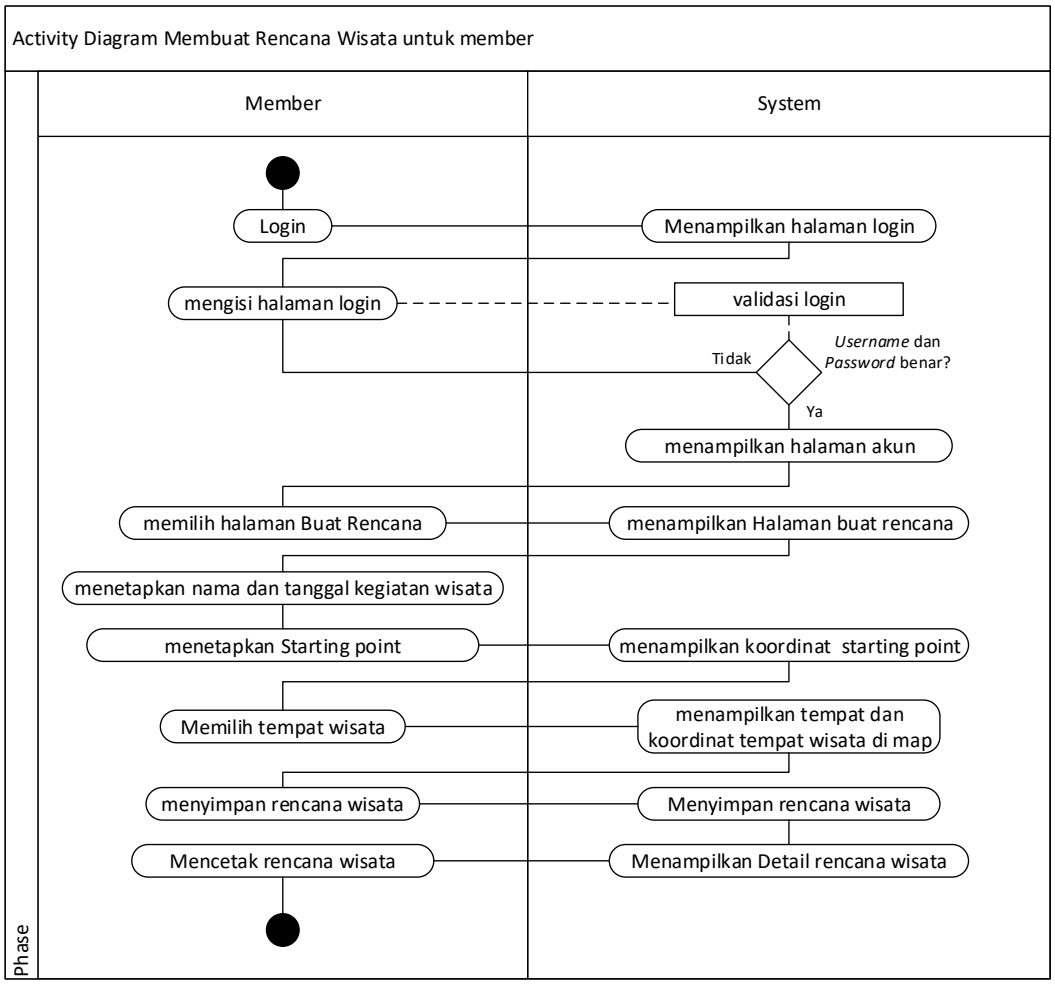
3.3.3 Activity Diagram

Berdasarkan *use case* yang telah dibuat sebelumnya maka dapat diperoleh *activity diagram* berdasarkan aktor yang terlibat dalam *usecase diagram. Activity diagram* dalam aplikasi dibagi menjadi dua bagian yaitu *activity diagram* buat

rencana wisata oleh *member* dan *activity diagram* buat rencana wisata oleh non *member*.

A. Activity Diagram Buat Rencana Wisata Member

Gambaran *Activity Diagram* buat rencana *member* yang digunakan dalam aplikasi dapat dilihat pada Gambar 3.3.



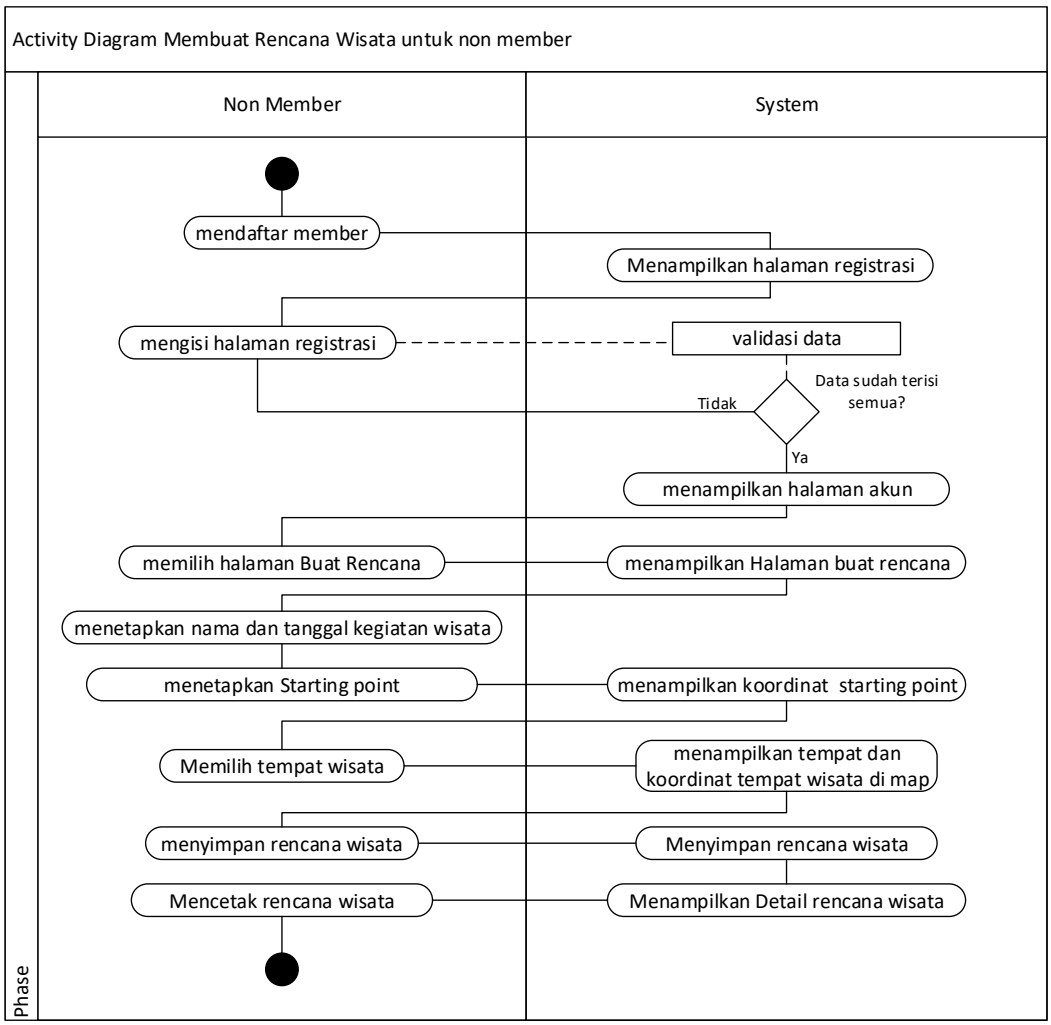
Gambar 3.3 *Activity Diagram* Buat Rencana *Member*

Berikut Penjelasan *Activity* Diagram pada Gambar 3.3:

1. Alur dari kegiatan membuat rencana yang dilakukan oleh *member*, yaitu masuk ke halaman *login* terlebih dahulu, kemudian *system* akan menampilkan halaman *login*.
2. Member mengisi *username* dan *password*, kemudian *system* akan melakukan validasi *username* dan *password*. Ketika data yang di masukan benar maka akan muncul halaman akun. Namun apabila *login* gagal maka *member* akan diarahkan kembali ke halaman *login*.
3. Member memilih halaman buat rencana. Kemudian menetapkan nama dan tanggal kegiatan wisata. Member harus terlebih dahulu memilih *starting point* dari rencana berliburnya, kemudian *system* akan menampilkan koordinat *starting point*.
4. Member memilih objek-objek wisata yang ingin dikunjungi. Kemudian *system* akan menampilkan objek wisata yang telah dipilih beserta koordinatnya di peta.
5. Member kemudian menyimpan rencana berlibur ke *database*. Kemudian *system* akan menampilkan detail perjalanan wisata yang telah dipilih oleh *member*.
6. Member dapat mencetak rencana berlibur.

B. Activity Diagram Buat Rencana Wisata Non Member

Gambaran *Activity Diagram* buat rencana *member* yang digunakan dalam aplikasi dapat dilihat pada Gambar 3.4.



Gambar 3.4 *Activity Diagram* Buat Rencana *Non Member*

Berikut Penjelasan *Activity* Diagram pada Gambar 3.4:

1. Alur dari kegiatan membuat rencana yang dilakukan oleh non *member*, yaitu harus mendaftar *member* dengan masuk ke halaman registrasi. Kemudian mengisi data diri lengkap di halaman registrasi. Jika semua data yang di masukan sudah lengkap *system* akan mengarahkan ke halaman akun.

2. Kemudian memilih halaman buat rencana. menetapkan nama dan tanggal kegiatan wisata. terlebih dahulu memilih *starting point* dari rencana berliburnya, kemudian *system* akan menampilkan koordinat *starting point*.
3. Kemudian memilih objek-objek wisata yang ingin dikunjungi. *system* akan menampilkan objek wisata yang telah dipilih beserta koordinatnya di peta.
4. Kemudian menyimpan rencana berlibur ke *database*. *system* akan menampilkan detail perjalanan wisata yang telah dipilih oleh.
5. Mencetak rencana berlibur.

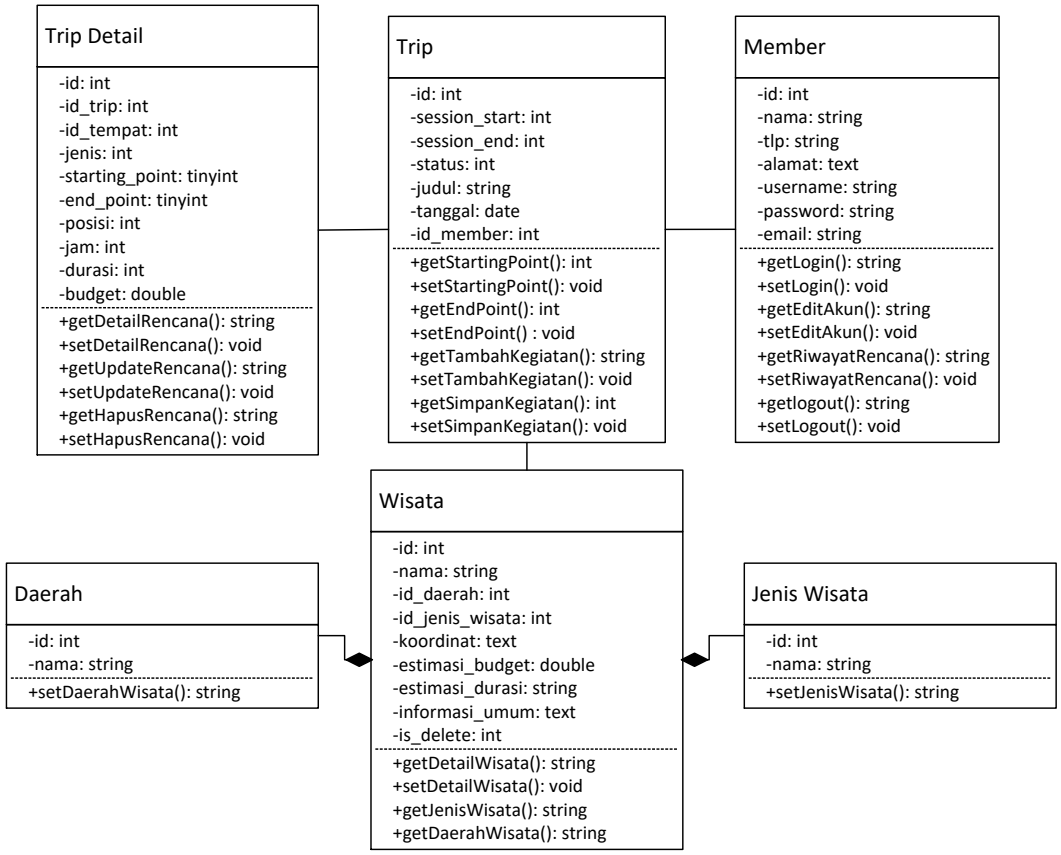
3.3.4 Class Diagram

Class Diagram yang digunakan dapat dilihat pada Gambar 3.5.

Berikut adalah penjelasan *class* diagram pada gambar 3.5:

1. Pada *class* Member terdapat fungsi *login* untuk *member* sehingga dapat menggunakan fitur halaman buat rencana. Terdapat fungsi edit akun untuk memperbarui data. Member juga dapat melihat riwayat rencana berlibur yang telah dibuat. Dan terakhir fungsi *logout*.
2. Pada *class* Trip terdapat fungsi untuk menetapkan *starting point* serta menambahkan tempat kegiatan wisata. Kemudian fungsi untuk menyimpan rencana berlibur.
3. Pada *class* Trip Detail, setelah menyimpan rencana berlibur *member* dapat melihat detail informasi dari rencana berlibur yang telah dibuat. Pada *class* ini juga terdapat fungsi untuk melakukan *update* rencana dan hapus rencana.

4. Pada *class* Wisata terdapat pilihan objek-objek wisata. Dan juga fungsi untuk menampilkan detail informasi dari objek wisata. *Class* ini juga mengambil data dari *class* daerah dan jenis wisata.
5. Pada *class* Daerah berfungsi menyimpan data daerah-daerah tempat wisata.
6. Pada *class* Jenis wisata berfungsi menyimpan data jenis-jenis wisata.



Gambar 3.5 *Class Diagram* Aplikasi

Kelas Member memiliki *association* dengan kelas Trip. Member dapat membuat rencana berlibur dengan cara menetapkan terlebih dahulu *starting point* dari kegiatan wisata dengan menggunakan *method getStartingPoint*(). Kemudian memilih objek-objek wisata dengan *method getTambahKegiatan*(). Kegiatan yang terakhir dipilih akan otomatis menjadi *end point* dari kegiatan wisata dengan

method getEndPoint(). Untuk menyimpan rencana berlibur yaitu menggunakan *method getSimpanKegiatan*().

Kelas Trip memiliki *association* dengan kelas Trip Detail. Rencana berlibur yang telah disimpan dapat dilihat detailnya pada kelas trip detail dengan *method getDetailRencana*(). pada kelas trip detail juga terdapat *method getUpdateRencana*() untuk memperbarui rencana dan *method getUpdateRencana*() untuk menghapus rencana yang telah dibuat.

Kelas Trip memiliki *association* dengan kelas Wisata di mana *member* dapat memilih objek wisata yang ingin ditambahkan ke bagian dari rencana berlibur. Pada kelas wisata terdapat *method getDetailWisata*() untuk menampilkan detail informasi objek wisata.

Kelas Wisata memiliki *composition* dengan kelas Daerah dan Jenis wisata, artinya kelas Daerah dan Jenis wisata merupakan bagian dari kelas Wisata. Kelas Daerah dan Jenis wisata tidak dapat berdiri sendiri apabila kelas Wisata tidak ada.

3.4 Metode Pengujian

Pengujian perangkat lunak merupakan suatu kegiatan yang dilakukan untuk memperoleh informasi serta mengevaluasi kualitas dari produk atau layanan yang sedang diuji. Tujuan pengujian dalam pengembangan aplikasi adalah untuk mengetahui apakah aplikasi yang diuji dapat memenuhi kebutuhan *user* dengan mendasari pada rancangan dan pengembangan perangkat lunak.

Metode pengujian yang dipakai dalam pengembangan aplikasi adalah *black box testing*. *Black box testing* atau tes fungsional adalah pengujian yang dilakukan

hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak yang dikembangkan.

Black box testing berfokus pada persyaratan fungsional perangkat lunak. Disebut juga pengujian *behavioral* atau pengujian partisi. *Black box testing* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk suatu program.

Beberapa hal yang diujikan dalam sistem aplikasi yaitu apakah sesuai dengan tujuan awal dikembangkan sistem, tujuan pengembangan sistem adalah:

1. Aplikasi dapat membuat rencana berlibur sesuai dengan keinginan dan kebutuhan *user*.
2. Sistem dapat memperlihatkan letak koordinat dari tempat-tempat wisata yang akan dikunjungi.
3. Sistem dapat menghitung jarak dan waktu dari satu tempat kegiatan ke tempat kegiatan berikutnya.
4. Sistem dapat menyimpan semua *record* dari rencana berlibur yang telah dibuat oleh *user*.
5. Sistem dapat membuat *user* mudah untuk memperbarui rencana berliburnya.
6. Sistem dapat mencetak rencana berlibur.
7. Aplikasi dapat memberikan daftar objek-objek wisata yang dapat dikunjungi oleh *user*.
8. Aplikasi dapat memberikan informasi secara detail dari objek wisata.

9. Sistem dapat memberikan informasi mengenai objek-objek wisata yang paling banyak disukai berdasarkan *user* lainnya..
10. Aplikasi dapat memberikan daftar hotel dan penginapan sehingga memudahkan *user* untuk mencari tempat menginap.
11. Aplikasi dapat memberikan informasi detail dari penyedia layanan hotel atau penginapan.
12. Sistem dapat membuat *user* bisa menuju situs utama dari penyedia layanan hotel atau penginapan sehingga *user* dapat melakukan reservasi.
13. Aplikasi dapat memberikan daftar dan informasi singkat tentang layanan transportasi sehingga memudahkan *user* untuk mencari kendaraan yang dapat digunakan.
14. Sistem dapat membuat *user* bisa menghubungi pihak penyedia layanan transportasi sehingga *user* dapat memesan kendaraan.
15. Aplikasi dapat membuat *user* bisa memperbarui data akunnya.
16. Aplikasi dapat menyediakan layanan untuk *user* mengirimkan pesan kepada *admin*.