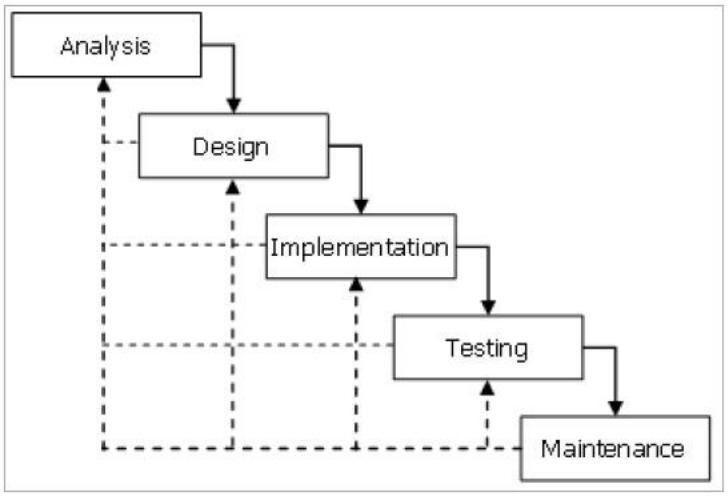
BAB III

METODE PENELITIAN

3.1 Metode Pengembangan Sistem

Pengembangan aplikasi kebudayaan di DIY berbasis android menggunakan model SDLC (*Software Development Life Cycle*). Model SDLC yang dipakai dalam pengembangan sistem ini yaitu model *Waterfall* dari (Bassil, 2012). Tahap-tahap pengembangan sistem menggunakan model *Waterfall* dapat dilihat pada gambar 3.1.



Gambar 3.1 Metode w*aterfall*

3.1.1 *Analysis*

Dalam tahapan ini dilakukan analisis terhadap seluruh kebutuhan software termasuk kegunaan software yang diharapkan oleh pengguna. Informasi ini dapat diperoleh dengan cara wawancara, survey, atau diskusi. Informasi tersebut kemudian dianalisa sehingga mendapatkan dokumentasi kebutuhan pengguna yang akan digunakan dalam tahapan berikutnya.

3.1.2 *Design*

Setelah dianalisa maka peneliti membuat rancangan *interface* dan sistem berdasarkan kebutuhan fungsi *software*. Adapun rancangan *user interface* menggunakan *software Mockup* dan rancangan sistem menggunakan *flowchart* dan beberapa UML (*Unified Modelling Language*) seperti *usecase diagram* dan *class diagram.*

3.1.3 *Implementation*

Pada tahap ini peneliti mengubah *design* menjadi sebuah aplikasi agar fungsi *software* dapat dijalankan. Untuk mengubah desain menjadi sebuah aplikasi peneliti menggunakan *software Android Studio* dengan bahasa pemrogramana *java*. Pengembangan aplikasi ini dilakukan dari awal hingga aplikasi siap dijalankan.

3.1.4 *Testing*

Tahap selanjutnya yaitu *testing*. *Testing* digunakan untuk mengetahui apakah aplikasi yang dikembangkan berjalan sesuai dengan yang diharapkan. *Testing* yang digunakan pada aplikasi kebudayaan di DIY berbasis android menggunakan metode *black box testing*.

3.1.5 *Maintenance*

Tahap terakhir yaitu *maintenance*. Tahap ini digunakan ketika aplikasi sudah jadi dan ingin lebih dikembangkan sekaligus pemeliharaan ketika terjadi *bug*.

3.2 Alat dan Bahan

3.2.1 Perangkat Lunak

Untuk mengembangkan aplikasi *event* kebudayaan di DIY berbasis android, peneliti membutuhkan beberapa perangkat lunak yang mendukung dalam pengerjaan aplikasi *event* kebudayaan di DIY seperti pada table 3.1.

Tabel 3.1 Tabel perangkat lunak

Software	Versi	Fungsi
Android Studio	1.3.2	Sebagai *tools* untuk mengembangkan kode aplikasi dan pembuatan layout.
CPanel	68.0.9	Sebagai *tools* yang akan membantu koneksi ke *database MySQL.*
Google Chrome	57.0.2987.110	Sebagai *browser* untuk membuka *PHPMyAdmin MySQL.*
Microsoft visio	2016	Untuk merancang *flowchart, use case diagram, class diagram*, dan *entity relationship diagram*

3.2.2 Perangkat Keras

Untuk mengembangkan aplikasi kebudayaan di DIY berbasis android, peneliti membutuhkan beberapa perangkat keras yang mendukung dalam pengerjaan aplikasi kebudayaan di DIY berbasis android seperti pada tabel 3.2 dan tabel 3.3.

Tabel 3.2 Tabel perangkat keras laptop

Item	Spesifikasi
Model	Asus Aspire E5-473G
CPU Cores	4 CPU x 2,2 Ghz
Processor Type	Intel(R) Core(TM) i5-5200U
RAM	4 Gb
Memory	500 Gb

Tabel 3.3 Tabel perangkat keras *smartphone*

Item	Spesifikasi
Model	Asus Z007
Operating System	Android OS, v4.4.2 (KitKat)
Processor Type	Dual-core 1.2 GHz
RAM	1 Gb
Memory	8 Gb

3.3 Bahan Penelitian

3.3.1 Wawancara

Wawancara di lakuan dengan cara mewawancarai langsung beberapa pegawai Pemda Dinas Pariwisata Yogyakarta dan Dinas Pariwisata Sleman yang berguna untuk mendapatkan informasi maupun data-data mengenai apa saja yang dibutuhkan dan yang akan di tampilkan untuk sebuah aplikasi yang berisikan informasi seputar *event-event* kebudayaan di DIY.

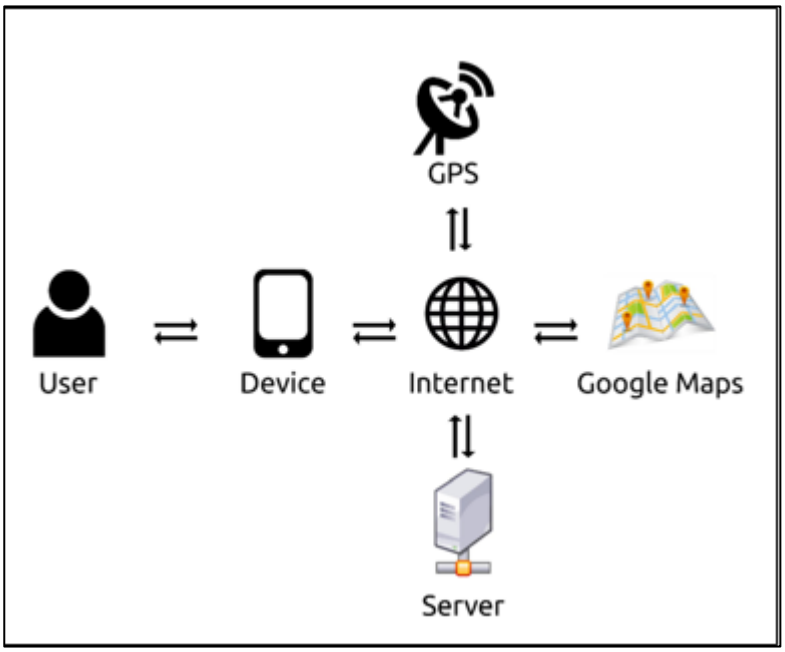
3.4 Analisis Kebutuhan

a. *Activity Splash Screen* sebagai pembuka agar tampilan lebih menarik.
b. *Main Activity* yang menampilkan menu daftar *event,* bantuan, tentang kami.

c. *Activity* daftar menu *event* menampilkan menu *event* Yogyakarta, menu *event* Sleman, menu *event* Bantul, menu *event* Gunung Kidul, menu *event* Kulonprogo.
d. *Activity detail list event* menampilkan daftar *event-event* di Yogyakarta, Sleman, Bantul, Gunung Kidul, Kulonprogo.
e. *Activity list detail event* menampilkan detail salah satu event yang ada di Yogyakarta, Sleman, Bantul, Gunung Kidul, Kulonprogo.
f. *Activity Map* menampilkan lokasi dari *event-event* yang ada di Jogja, Sleman, Bantul, Gunung Kidul, Kulonprogo.
g. *Activity* Galeri menampilkan list gambar atau foto *event* yang ada di Jogja, Sleman, Bantul, Gunung Kidul, Kulonprogo.
h. *Activity View* Galeri menampilkan salah satu gambar atau foto *event* yang ada di Jogja, Sleman, Bantul, Gunung Kidul, Kulonprogo.
i. *Activity* bantuan menampilkan cara menggunakan aplikasi ini.
j. *Activity* tentang kami, menampilkan deskripsi dari *developer* aplikasi.

3.5 Arsitektur Sistem

Gambaran arsitektur sistem yang digunakan dalam Aplikasi Kebudayaan di DIY dapat dilihat pada gambar 3.2.



Gambar 3.2 Arsitektur Sistem

Penjelasan tentang gambar 3.2

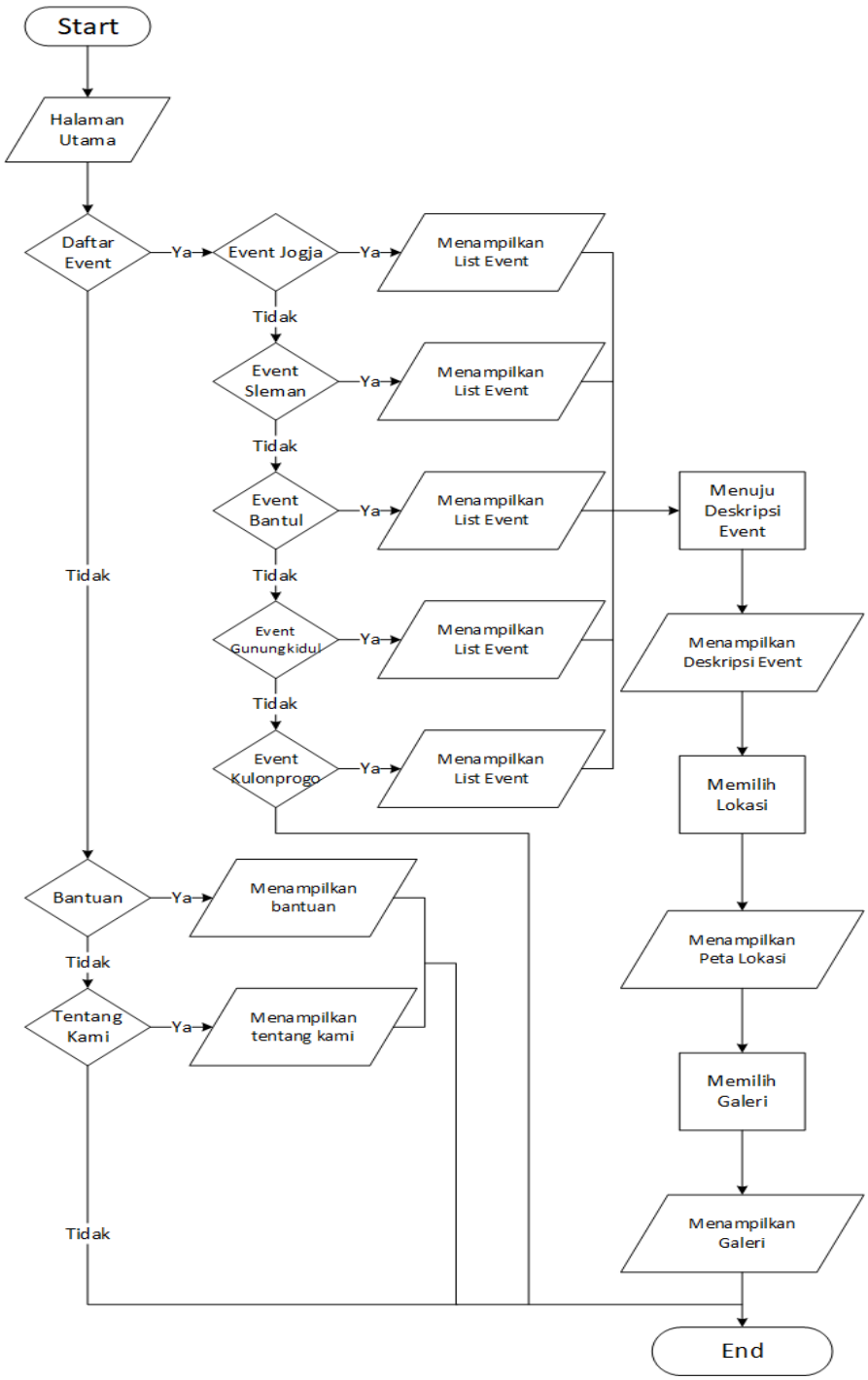
a. User memberikan perintah atau meminta aplikasi untuk menunjukkan *event* kebudayaan yang diinginkan.
b. User memberikan perintah atau meminta aplikasi untuk menunjukkan lokasi *event* kebudayaan dalam peta digital.
c. Aplikasi meminta data *event* kebudayaan dari *database server* melalui koneksi internet.
d. Aplikasi meminta data peta lokasi (*Google Maps*) melalui koneksi internet.

3.6 Rancangan Sistem

Metode yang digunakan dalam perancangan *logic* aplikasi kebudayaan di DIY berbasis android ini adalah *flowchart* dan *Unified Model Language*(UML). Adapun model UML yang digunakan yaitu *usecase diagram* dan *class diagram*.

3.6.1 *Flowchart*

Untuk membantu *logic* aplikasi ini maka dibuatlah *flowchart*. Flowchart dapat dilihat pada gambar 3.3.



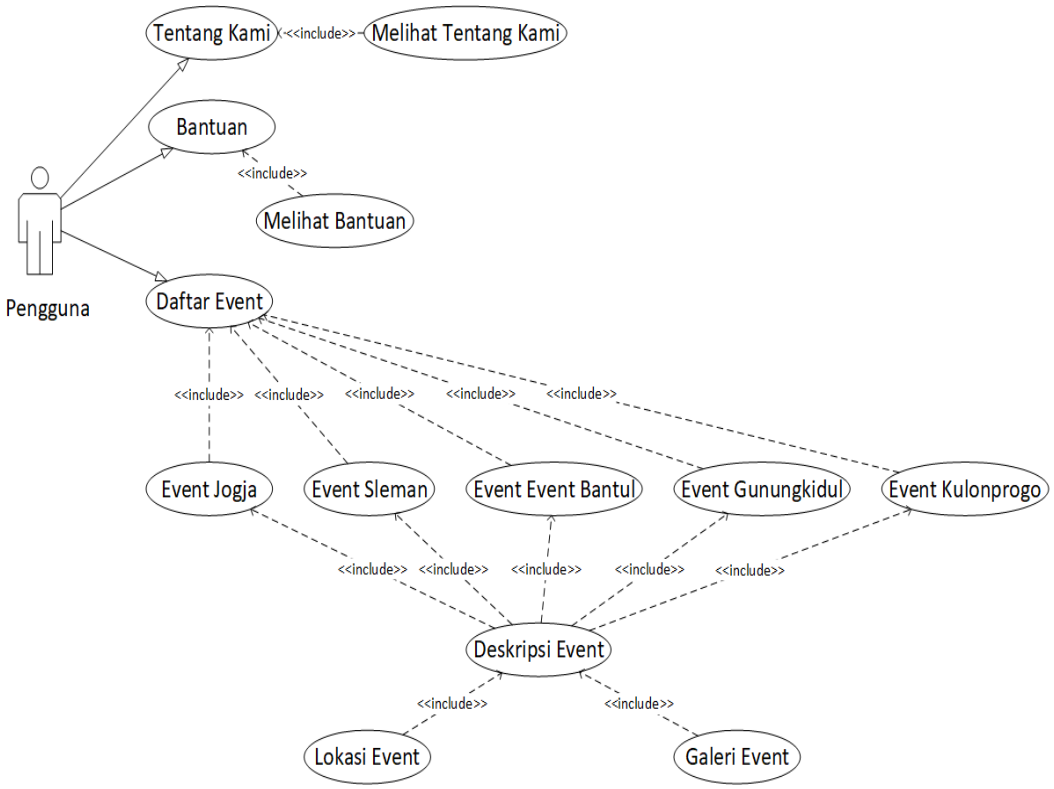
Gambar 3.3 *Flowchart* aplikasi

Adapun penjelasan tentang *flowchart* di atas:

1. Saat membuka aplikasi pengguna akan langsung dihadapkan dengan halaman utama dari aplikasi yang menampilkan *button* daftar event, *button* bantuan, *button* tentang kami.
2. Jika pengguna memilih daftar event maka sistem akan menampilkan menu *event* Yogyakarta, menu *event* Sleman, menu *event* Bantul, menu *event* Gunung Kidul, menu *event* Kulonprogo.
3. Jika pengguna memilih menu *event* Yogyakarta maka akan menampilkan *list event-event* di Yogyakarta.
4. Jika pengguna memilih menu *event* Sleman maka akan menampilkan *list event-event* di Sleman.
5. Jika pengguna memilih menu *event* Bantul maka akan menampilkan *list event-event* di Bantul.
6. Jika pengguna memilih menu *event* Gunung Kidul maka akan menampilkan *list event-event* di Gunung Kidul.
7. Jika pengguna memilih menu *event* Kulonprogo maka akan menampilkan *list event-event* di Kulonprogo.
8. Setelah pengguna memilih event yang diinginkan maka sistem akan menampilkan deskripsi dari event yg dipilih.
9. Jika pengguna ingin melihat lokasi *event* maka sistem akan menampilkan peta lokasi dari *event* yang telah dipilih.
10. Jika pengguna ingin melihat galeri maka sistem akan menampilkan galeri dari *event* yang telah dipilih.
11. Jika pengguna memilih bantuan maka sistem akan menampilkan menu bantuan.
12. Jika pengguna memilih tentang kami maka sistem akan menampilkan menu tentang kami.

3.6.2 *Use Case* Diagram

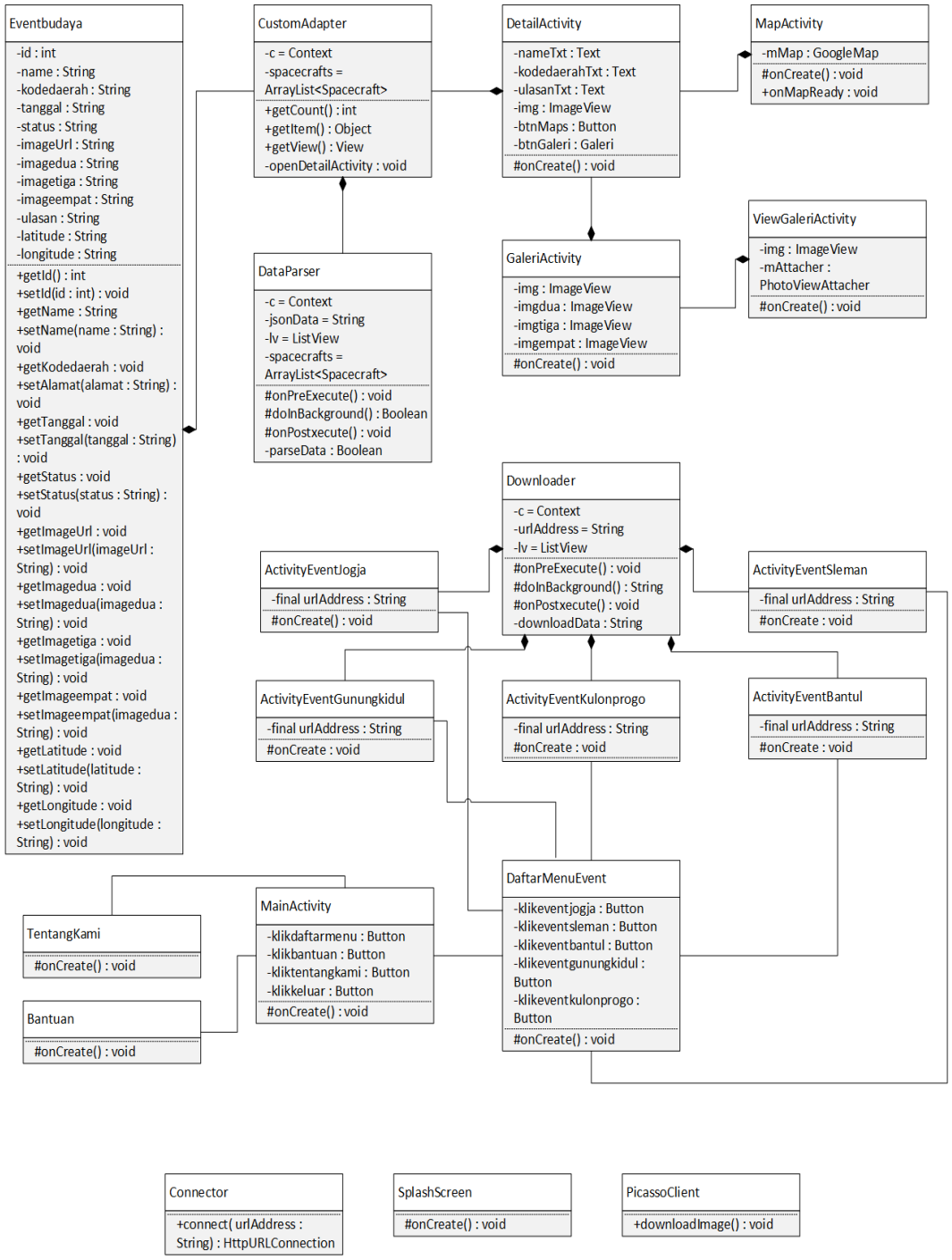
Berdasarkan analisis kebutuhan di atas maka dibuatlah *usecase* diagram untuk membantu dalam pembuatan aplikasi portal culture. Gambar *usecase* diagram bisa dilihat pada gambar 3.4.



Gambar 3.4 *Use case* aplikasi

3.6.3 *Class* Diagram

Gambaran *class diagram* dari aplikasi kebudayaan di DIY berbasis *android* bisa dilihat pada gambar 3.5.



Gambar 3.5 *Class* diagram aplikasi

Adapun penjelasan tentang *class* diagram di atas:

1. *Class Eventbudaya* merupakan inisialisasi data yang diambil dari *database*.
2. *Class CustomAdapter* berfungsi mengatur item-item yang ada pada *ListView*.
3. *Class DataParser* berfungsi mengubah data yang di *download* dari server menjadi data yang dapat dibaca oleh aplikasi.
4. *Class DetailActivity* berfungsi sebagai *class* yang menampilkan informasi *event* secara keseluruhan.
5. *Class GaleriActivity* berfungsi sebagai *class* yang menampilkan list foto atau gambar yang ada pada setiap event.
6. *Class ViewGaleri* berfungsi untuk menampilkan detail gambar dari salah satu list atau foto dari setiap event.
7. *Class MapActivity* berfungsi untuk menampilkan peta lokasi *event*.
8. *Class PicassoClient* berfungsi untuk menampilkan gambar dari *url*.
9. *Class Connector* berfungsi sebagai jembatan antara aplikasi dengan server.
10. *Class Splashscreen* berfungsi menampilkan halaman *splashscreen* pada saat aplikasi dibuka.
11. *Class* DaftarMenuEvent berfungsi sebagai *class* untuk menampilkan *class ActivityEventJogja, class ActivityEventSleman, class ActivityEventBantul, class ActivityEventGunungkidul, class ActivityEventkulonprogo*.
12. *Class ActivityEventJogja* menampilkan *event-event* yang ada di Yogyakarta.
13. *Class ActivityEventSleman* menampilkan *event-event* yang ada di Sleman.
14. *Class ActivityEventBantul* menampilkan *event-event* yang ada di Bantul.
15. *Class ActivityEventGunungkidul* menampilkan *event-event* yang ada di Gunung Kidul.
16. *Class ActivityEventKulonprogo* menampilkan *event-event* yang ada di Kulonprogo.
17. *Class Downloader* berfungsi untuk *download* data dari server.
18. *Class MainActivity* berfungsi sebagai *class* untuk menampilkan *class* Bantuan, *class* TentangKami, *class* Keluar.
19. *Class* Bantuan berfungsi menampilkan informasi mengenai cara menggunakan aplikasi.

20. *Class* TentangKami berfungsi menampilkan informasi mengenai *developer* aplikasi.

Adapun penjelasan tentang relasi dari *class* diagram di atas:

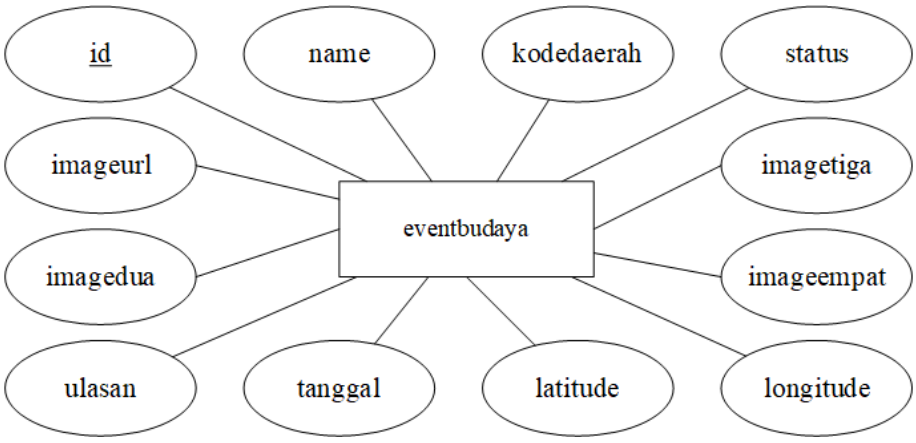
1. *Class Eventbudaya* memiliki relasi *composition* dengan *class CustomAdapter* karena data yang ditampilkan pada *class CustomAdapter* berdasarkan *method getId, getName, getKodedaerah, getImageUrl, getImagedua, getImagetiga, getImageempat, getUlasan, getLatitude, getLongitude* yang ada pada *class Eventbudaya.*
2. *Class DataParser* memiliki relasi *composition* dengan *class CustomAdapter* karena *class DataParser* membutuhkan *import* data dari *class CustomerAdapter* yang mengambil *method getId, getName, getKodedaerah, getImageUrl, getImagedua, getImagetiga, getImageempat, getUlasan, getLatitude, getLongitude* dari *class EventBudaya.*
3. *Class CustomAdapter* memiliki relasi *composition* dengan *class DetailActivity* karena data yang ditampilkan pada *class DetailActivity* di *import* oleh *class CustomAdapter* untuk menampilkan data dengan *method getName, getKodedaerah, getUlasan, getImageUrl, getImagedua, getImagetiga, getImageempat, getLatitude, getLongitude.*
4. *Class DetailActivity* memiliki relasi *composition* dengan *class MapActivity* karena *Class DetailActivity* mengimport data *latitude* dan *longitude* dari *class MapActivity.*
5. *Class DetailActivity* memiliki relasi *composition* dengan *class GaleriActivity* karena *Class DetailActivity* mengimport data img, imgdua, imgtiga, imgempat dari class *GaleriActivity.*
6. *Class GaleriActivity* memiliki relasi *composition* dengan *class ViewGaleri* karena gambar yang ditampilkan pada *class ViewGaleri* berdasarkan *ImageView* dari *class GaleriActivity.*
7. *Class MainActivity* memiliki relasi *association* dengan *class* Bantuan, *class* TentangKami karena *class MainActivity* dapat mengakses data *class* Bantuan,

class TentangKami melalui *Button* klikdaftarmenu, *Button* klikbantuan, *Button* kliktentangkami.

8. *Class* DaftarMenuEvent memiliki relasi *association* dengan *class ActivityEventJogja*, *class ActivityEventSleman*, *class ActivityEventBantul*, *class ActivityEventGunungkidul*, *class ActivityEventkulonprogo* karena *Class* DaftarMenuEvent dapat mengakses data *class ActivityEventJogja*, *class ActivityEventSleman*,*classActivityEventBantul*,*classActivityEventGunungkidul* ,*classActivityEventkulonprogo* melalui *Button* klikeventjogja, *Button* klikeventsleman, *Button* klikeventbantul, *Button* klikeventgunungkidul, *Button* klikeventkulonprogo.
9. *Class Activity Event Jogja*, *class Activity Event Sleman*, *class Activity Event Bantul*, *class Activity Event Gunungkidul*, *class Activity Event kulonprogo* memiliki relasi *composition* dengan c*lass Downloader* karena jika c*lass Downloader dihilangkan maka Class Activity Event Jogja*, *class Activity Event Sleman*, *class Activity Event Bantul*, *class Activity Event Gunungkidul*, *class Activity Event kulonprogo* tidak dapat beroperasi atau mengambil data ke server.

3.6.4 *Entity Relationship* Diagram

Gambaran *entity relationship diagram* dari aplikasi kebudayaan di DIY berbasis *android* bisa dilihat pada gambar 3.6.



Gambar 3.6 *Entity relationship* diagram aplikasi

Adapun penjelasan tentang *entity relationship diagram* di atas:

1. Entitas: eventbudaya.
2. Atribut: Id, name, kodedaerah, tanggal, status, imageurl, imagedua, imagetiga, imageempat, ulasan, latitude, longitude.

3.7 Rancangan Tabel

Pembuatan aplikasi kebudayaan di DIY berbasis android ini menggunakan SQL sebagai bahasa Standart yang digunakan untuk mengakses database. Adapun tabel yang digunakan adalah sebagai berikut:

1. Tabel eventbudaya

Tabel ini digunakan untuk menyimpan informasi event yang ada di Yogyakarta. Struktur tabel spacecraft bisa dilihat pada tabel 3.4.

Tabel 3.4 Tabel eventbudaya

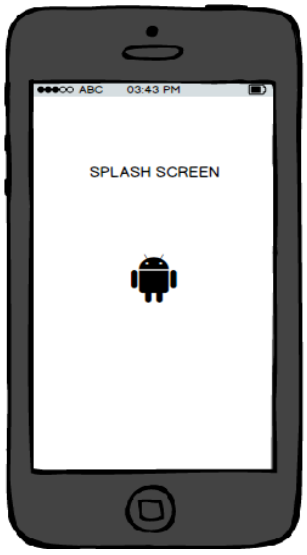
Field	Type	Widht	Definition
id	Int	15	ID event
name	Varchar	100	Nama event
kodedaerah	Varchar	20	Daerah event
tanggal	Varchar	50	Tanggal event
status	Varchar	20	Status event
imageurl	Varchar	100	Nama gambar 1
imagedua	Varchar	100	Nama gambar 2
imagetiga	Varchar	100	Nama gambar 3
imageempat	Varchar	100	Nama gambar 4
ulasan	text		Deskripsi event
latitude	Varchar	50	Latitude Wisata
longitude	Varchar	50	Longitude Wisata

3.8 Rancangan *User Interface*

Perancangan antarmuka merupakan tampilan dari suatu perangkat lunak yang berperan sebagai media komunikasi antara perangkat lunak dan pengguna. Perancangan ini merupakan sebuah penggambaran, perencanaan dan pengaturan dari beberapa elemen yang terpisah ke dalam satu kesatuan yang utuh. Perancangan antarmuka diharapkan memudahkan pengguna dalam melakukan proses interaksi terhadap sistem. Berikut adalah perancangan antar muka yang ada pada aplikasi kebudayaan di DIY berbasis android.

3.8.1 Perancangan *Splashscreen*

Perancangan *splashscreen* digunakan untuk tampilan *activity splashscreen* ketika *user* baru membuka aplikasi kebudayaan di DIY. Peracangan *splashscreen* dapat dilihat pada gambar 3.7.



Gambar 3.7 *Splashscreen*

3.8.2 Perancangan *Main Activity*

Main activity dirancang dengan menampilkan 3 *button* yaitu daftar event, bantuan, tentang kami. Perancangan tampilan *main activity* dapat dilihat pada gambar 3.8.

Gambar 3.8 *Main activity*

3.8.3 Perancangan *Activity* Daftar Menu *Event*

Activity daftar menu *event* dirancang dengan menampilkan 5 *button* yaitu *event* Yogyakarta, *event* Sleman, *event* Bantul, *event* Gunung Kidul, *event* Kulonprogo. Disini *user* dapat memilih salah satu lokasi *event* yang dikehendaki. Perancangan tampilan *activity* daftar menu *event* dapat dilihat pada gambar 3.9.

Gambar 3.9 *Activity* daftar menu *event*

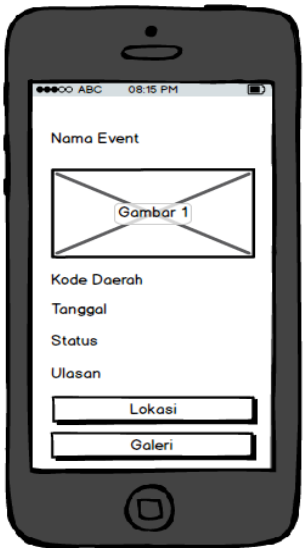
3.8.4 Perancangan *Activity Detail List Event*

Activity detail list event dirancang dengan menampilkan beberapa *image view* dan teks dari masing-masing *event* yang ada di tiap daerah. Disini *user* dapat memilih salah satu *event* yang dikehendaki. Perancangan tampilan *activity detail list event* dapat dilihat pada gambar 3.10.

Gambar 3.10 *Activity detail list event*

3.8.5 Perancangan *Activity List Detail Event*

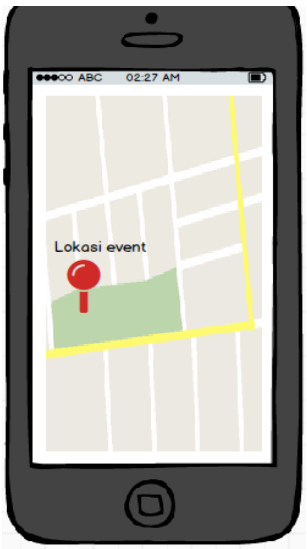
Activity list detail event Jogja dirancang dengan menampilkan *image view* dan teks dari *event* yang ada di Jogja. Disini *user* dapat melihat informasi dan gambar dari seputar event di Jogja yang sudah dipilih. Perancangan tampilan *activity list detail event* Jogja dapat dilihat pada gambar 3.11.



Gambar 3.11 *Activity list detail event*

3.8.6 Perancangan *Activity Map*

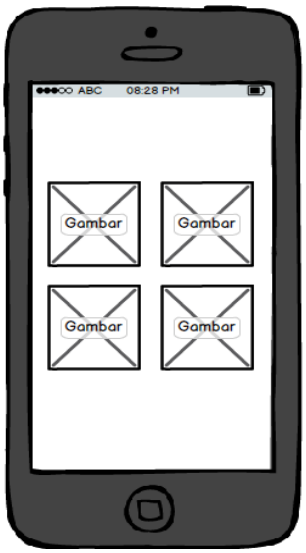
Activity map dirancang dengan menampilkan *map* atau lokasi dari *event* yang dipilih. Perancangan tampilan *Activity map* dapat dilihat pada gambar 3.12.



Gambar 3.12 *Activty map*

3.8.7 Perancangan *Activity* Galeri

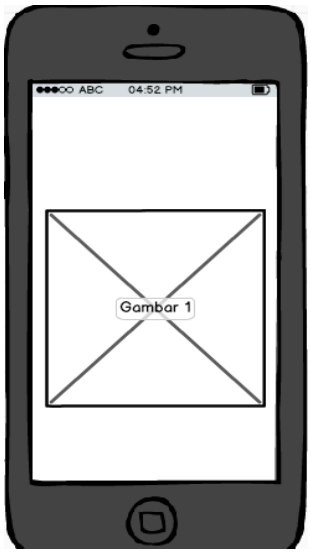
Activity galeri dirancang dengan menampilkan *list* gambar atau foto dari *event* yang dipilih. Perancangan tampilan *Activity* galeri dapat dilihat pada gambar 3.13.



Gambar 3.13 *Activity* galeri

3.8.8 Perancangan *Activity View Galeri*

Activity view galeri dirancang dengan menampilkan *detail* gambar atau foto dari *event* yang dipilih. Perancangan tampilan *Activity view* galeri dapat dilihat pada gambar 3.14.



Gambar 3.14 *Activity view* galeri

3.8.9 Perancangan *Activity* Bantuan

Activity menu bantuan dirancang dengan menampilkan teks yang berisi petunjuk tentang cara menggunakan aplikasi ini. Perancangan tampilan *Activity* menu bantuan dapat dilihat pada gambar 3.15.

Gambar 3.15 *Activity* Bantuan

3.8.10 Perancangan *Activity* Tentang Kami

Activity menu tentang kami dirancang dengan menampilkan teks yang berisi informasi seputar aplikasi dan informasi mengenai *developer*. Perancangan tampilan *Activity* tentang kami dapat dilihat pada gambar 3.16.

Gambar 3.16 *Activity* tentang kami

3.9 Metode Pengujian

Dalam mengembangkan sebuah aplikasi pengujian diperlukan untuk mengevaluasi jalannya aplikasi. Evaluasi digunakan untuk mengetahui apakah aplikasi tersebut sudah memenuhi target perencanaan dan melihat apakah masih ada kekurangannya. Pengujian dilakukan dengan *black box testing* dimana dengan metode ini, pengujian dilihat dari jalannya aplikasi secara fungsional.

Hal-hal yang akan diuji pada aplikasi kebudayaan di DIY yaitu:

1. Aplikasi dapat menampilkan menu daftar *event*.
2. Aplikasi dapat memproses dan mengambil data dari server untuk *event* Jogja.
3. Aplikasi dapat menampilkan *event-event* di Jogja.
4. Aplikasi dapat mengambil data serta menampilkan lokasi *event-event* di Jogja.
5. Aplikasi dapat mengambil data serta menampilkan galeri *event-event* di Jogja.
6. Aplikasi dapat memproses dan mengambil data dari server untuk *event* Sleman.
7. Aplikasi dapat menampilkan *event-event* di Sleman.
8. Aplikasi dapat mengambil data serta menampilkan lokasi *event-event* di Sleman.
9. Aplikasi dapat mengambil data serta menampilkan galeri *event-event* di Sleman.
10. Aplikasi dapat memproses dan mengambil data dari server untuk *event* Bantul.
11. Aplikasi dapat menampilkan *event-event* di Bantul.
12. Aplikasi dapat mengambil data serta menampilkan lokasi *event-event* di Bantul.
13. Aplikasi dapat mengambil data serta menampilkan galeri *event-event* di Bantul.
14. Aplikasi dapat memproses dan mengambil data dari server untuk *event* Gunung Kidul.
15. Aplikasi dapat menampilkan *event-event* di Gunung Kidul.

16. Aplikasi dapat mengambil data serta menampilkan lokasi *event-event* di Gunung Kidul.
17. Aplikasi dapat mengambil data serta menampilkan galeri *event-event* di Gunung Kidul.
18. Aplikasi dapat memproses dan mengambil data dari server untuk *event* Kulonprogo.
19. Aplikasi dapat menampilkan *event-event* di Kulonprogo.
20. Aplikasi dapat mengambil data serta menampilkan lokasi *event-event* di Kulonprogo.
21. Aplikasi dapat mengambil data serta menampilkan galeri *event-event* di Kulonprogo.
22. Aplikasi dapat menampilkan menu bantuan.
23. Aplikasi dapat menampilkan menu tentang kami.
24. Aplikasi dapat keluar program dengan menekan tombol *back* pada halaman utama aplikasi.

3.10 Analisis Data

Metode analisis data yang digunakan pada penelitian ini adalah metode deskriptif. Metode deskriptif adalah suatu metode yang digunakan dalam meneliti suatu objek, kondisi, atau peristiwa. Tujuan dari penelitian deskriptif adalah untuk menggambarkan atau mendeskripsikan fakta-fakta, sifat serta hubungan antara fenomena yang diteliti.

Teknik pengolahan data dapat menggunakan pengukuran dengan skala Likert. Skala Likert adalah suatu skala psikometrik yang umum digunakan dalam kuesioner, dan merupakan skala yang paling banyak digunakan dalam riset berupa survei.

Kriteria jawaban yang dibagikan kepada responden menggunakan kuesioner berupa pengukuran skala Likert. Responden diminta untuk menggunakan sistem informasi secara keseluruhan dengan berhadapan secara langsung. Responden diminta memberikan salah satu pilihan dari jawaban yang telah

disediakan. Ada 5 pilihan jawaban yang diberikan, mulai dari sangat setuju sampai dengan sangat tidak setuju. Data kuantitatif diubah berdasarkan bobot skor satu, dua, tiga, empat, dan lima. Pembagian kategori dan skor pengukuran skala Likert dapat di lihat pada tabel 3.5.

Tabel 3.5 Tabel skor skala likert

No	Kategori	Skor
1	Sangat Setuju	5
2	Setuju	4
3	Cukup	3
4	Tidak Setuju	2
5	Sangat Tidak Setuju	1

Hasil persentase digunakan untuk memberikan jawaban atas kelayakan dari aspek-aspek yang diteliti. Nilai maksimal yang diharapkan adalah 100% dan minimal 0%.

Pembagian kategori kelayakan dapat dilihat pada tabel 3.6.

Tabel 3.6 Tabel kategori kelayakan aplikasi

No	Kategori	Skor
1	Sangat Layak	81% - 100%
2	Layak	61% - 80%
3	Cukup	41% - 60%
4	Tidak Layak	21% - 40%
5	Sangat Tidak Layak	<20%