3. ANALISIS DAN DESAIN SISTEM

Pada bab ini akan dibahas mengenai desain dan analisis sistem mengenai penerapan K-Means Clustering untuk mendapatkan nuansa warna dari gambar yang di-*upload user* untuk diukur kedekatan warnanya terhadap desain *template* yang disediakan aplikasi.

3.1. Analisa Sistem

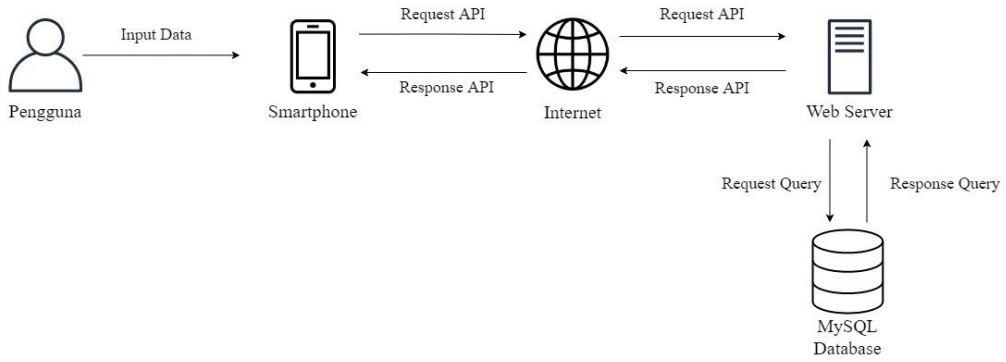
Aplikasi atau situs desain grafis untuk pembuatan konten telah banyak tersedia untuk memudahkan orang-orang yang kurang ahli dalam membuat konten visual. Mulai dari mendesain dari awal sampai dengan menggunakan desain *template* yang telah disediakan. Kehadiran aplikasi atau situs desain grafis cukup membantu karena pemilik bisnis banyak yang ranahnya bukan pada bidang desain.

Banyaknya pilihan desain *template* dengan jenis *output* dan kategori konten yang berbeda-beda dapat menyebabkan *user* mengalami kebingungan dalam menentukan desain *template* mana yang cocok untuk kebutuhan mereka atau bahkan *user* memilih desain *template* yang tidak cocok dengan konten yang dibuat sehingga hasilnya tidak memuaskan.

Berdasarkan penelitian sebelumnya yang dijelaskan pada bab 2.2.1 sistem yang membantu *user* dalam bidang desain grafis dengan cara membantu mengatur *layout* elemen-elemen yang diinginkan *user* menggunakan GANs, masih bergantung pada elemen-elemen yang diinginkan *user*. Apabila *user* tidak dapat menyediakan elemen-elemen yang diinginkan, sistem tersebut juga tidak dapat melakukan pekerjaannya. Sehingga *user* masih harus membuat desain elemen sendiri untuk dijadikan *parameter*. Aplikasi skripsi ini memungkinkan *user* untuk menghasilkan sebuah desain tanpa harus memiliki *skill* desain dengan hanya memilih kategori, memasukkan gambar, dan memasukkan teks yang bersifat opsional.

3.2. Arsitektur Sistem

Untuk mengetahui bagaimana sebuah sistem bekerja, dibutuhkan sebuah desain struktur dari sistem tersebut. Hal ini bertujuan agar pengguna dapat memahami proses yang terjadi. Gambar 3.1 merupakan gambar desain sistem secara keseluruhan dari aplikasi.



Gambar 3.1. Arsitektur Sistem.

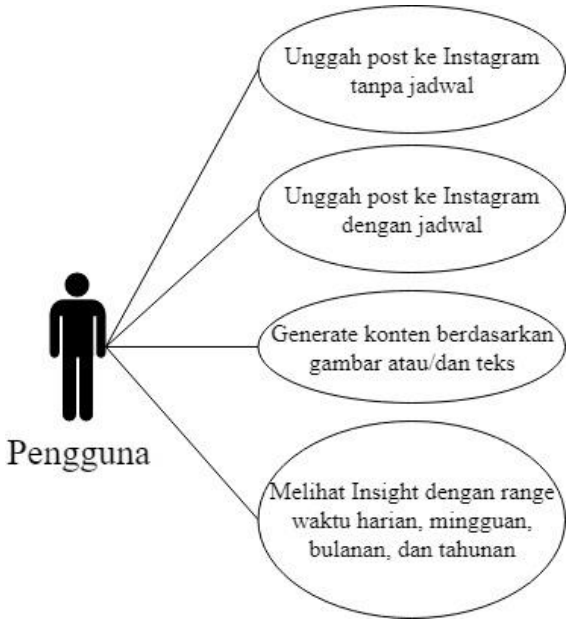
Pertama kali aplikasi dijalankan, aplikasi akan meminta *input* data dari *user* yaitu jenis *output*, kategori konten, gambar, dan teks opsional. Pada saat tombol untuk *upload* data tersebut ditekan, gambar dari user akan di-*upload* ke *server* untuk dideteksi warna dominannya. Apabila proses mendeteksi warna dominan telah selesai, maka data warna RGB hasil pendeteksian akan dikembalikan dalam bentuk JSON.

Setelah data warna RGB telah didapat, *request* API akan dilakukan pada *server* untuk meminta data nuansa warna berbagai desain *template* agar dapat dihitung kedekatan warnanya. Setelah itu, *filename* desain *template* akan dikembalikan pada *smartphone* agar dapat menggabungkan desain *template* dengan teks dan gambar yang dipilih *user*. Untuk mendapatkan koordinat peletakan teks dan gambar dari desain *template* tertentu, *request* API akan dilakukan ke *database* untuk mengambil koordinat-koordinat tersebut.

Setelah *filename* beserta koordinat peletakan teks dan gambar didapatkan, penggabungan teks dan gambar dengan desain *template* dilakukan di *smartphone*. Apabila penggabungan telah selesai dilakukan, *user* dapat menyimpan hasil penggabungan tersebut ke dalam *gallery*.

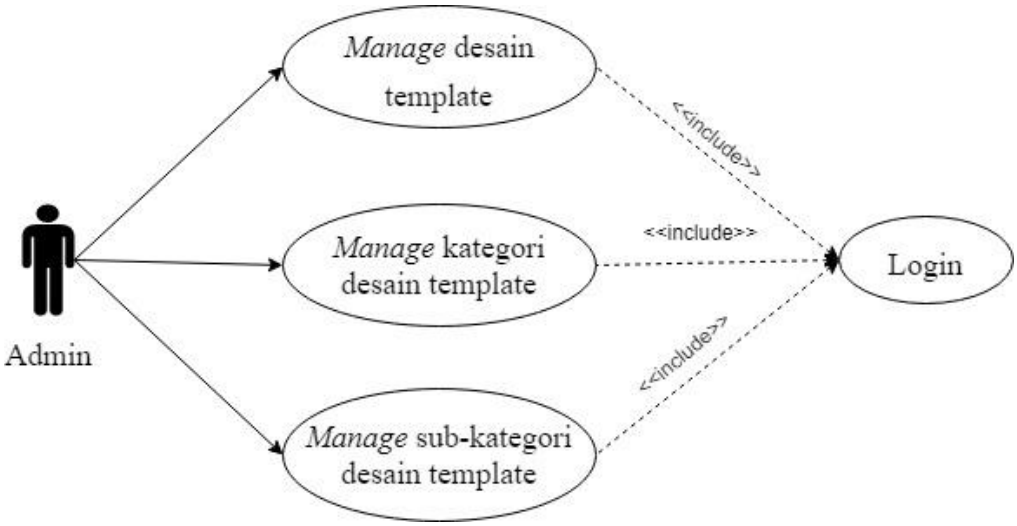
3.3. *Use Case Diagram*

Use case diagram (UCD) bertujuan untuk memberikan gambaran tentang apa saja yang dapat dilakukan oleh *actor* pada sistem yang dibuat. Gambar 3.2 dan Gambar 3.3 merupakan *use case diagram user* dan *admin* dari sistem yang dibuat.



Gambar 3.2. *Use Case Diagram* Pengguna Aplikasi

Use case pada Gambar 3.2 terdapat *actor user*. Pengguna dapat melakukan pembuatan konten berdasarkan teks dan/atau berdasarkan gambar dengan berbagai jenis desain *template* yang siap untuk dijadikan konten.



Gambar 3.3. *Use Case Diagram Admin*

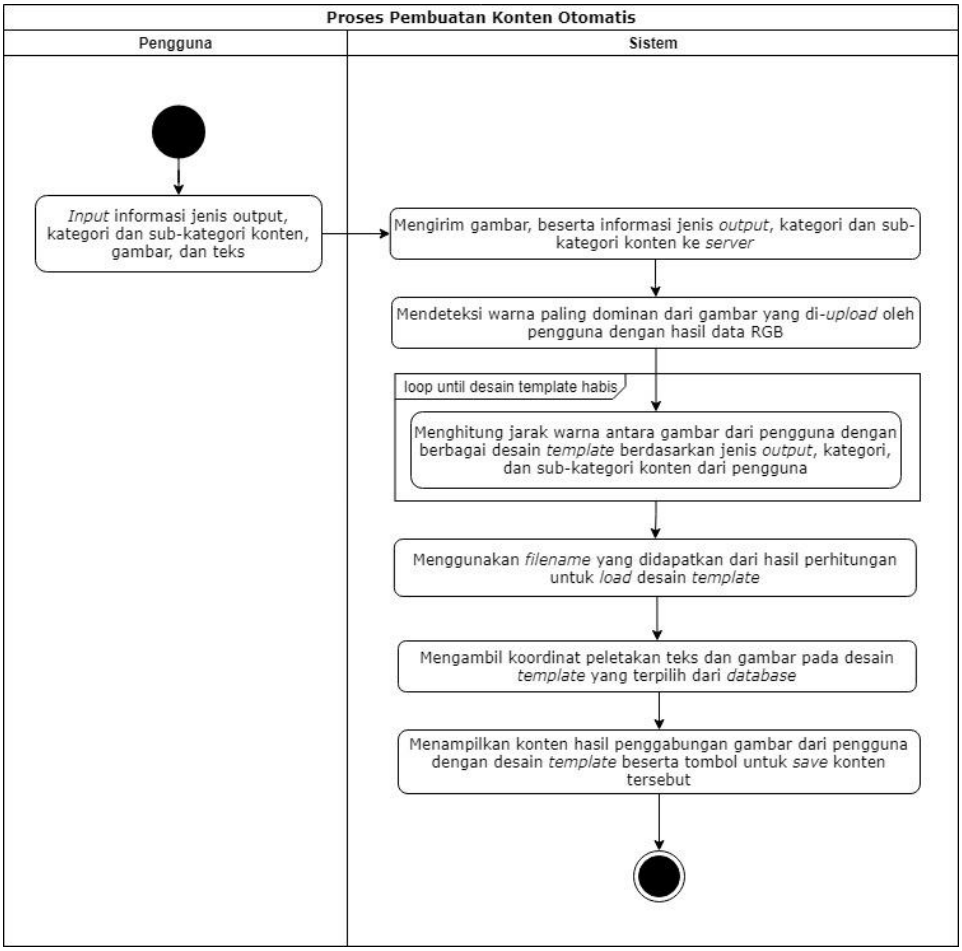
Pada *use case* di atas terdapat *actor admin*. *Admin* dapat memanajemen desain *template*, kategori desain *template*, dan sub-kategori desain *template*, dengan *actions* berupa *add*, *edit*, dan *delete*.

3.4. *Activity Diagram*

Activity diagram dibuat berdasarkan *use case diagram*, untuk menggambarkan *workflow* atau aktivitas dari sebuah sistem atau proses yang dapat dilakukan *actor.*

3.4.1. Proses Pembuatan Konten

Proses pembuatan konten merupakan proses utama pada skripsi ini, karena konten hasil pembuatan konten didapatkan dari proses ini. Data yang berupa jenis *output*, kategori konten, gambar, dan teks didapatkan dari *input* pengguna.



Gambar 3.4. *Activity Diagram* Proses Pembuatan Konten

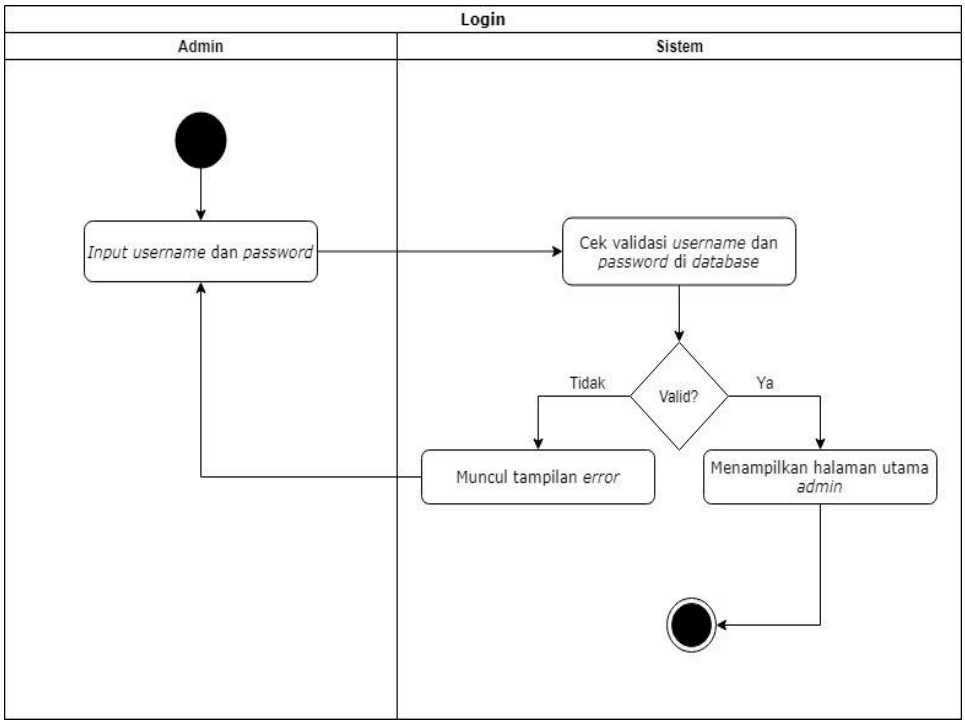
Activity diagram untuk proses pembuatan konten dapat dilihat pada Gambar 3.4. Bermula dengan pengguna memilih jenis *output*, kategori beserta sub-kategori konten, dilanjutkan dengan pengguna diminta untuk melakukan *input* data gambar dan/atau teks untuk dijadikan konten. Gambar yang di-*input* oleh pengguna akan dikirim ke *server* agar dapat ditelusuri warna paling dominannya. Apabila warna paling dominan telah didapatkan, warna tersebut digunakan untuk dihitung kedekatan warnanya dengan desain *template* yang tersedia. Setelah perhitungan selesai, *filename* desain template dikembalikan ke *smartphone* pengguna agar dapat digabungkan dengan gambar dari

pengguna lalu ditampilkan bersamaan dengan tombol *save* dimana tombol tersebut dapat ditekan oleh pengguna untuk menyimpan hasil-hasil penggabungan gambar tersebut ke dalam *gallery*.

Penggunaan Android Volley diterapkan untuk melakukan *request* API dimana API digunakan untuk mengirim data dari pengguna beserta gambarnya ke *server* dan sebaliknya, dan untuk menjalankan *script* Python yang digunakan untuk menjalankan algoritma K-Means Clustering.

3.4.2. Login

Gambar 3.5 menggambarkan proses *login*. *Admin* yang ingin masuk ke dalam sistem harus melakukan *login* melalui aplikasi *android* terlebih dahulu. *Login* membutuhkan data *username* dan *password.* Jika *username* dan *password* terdaftar di *database*, *admin* akan di *redirect* ke halaman utama *admin.* Apabila *username* dan *password* tidak terdaftar, maka akan muncul tampilan *error*.

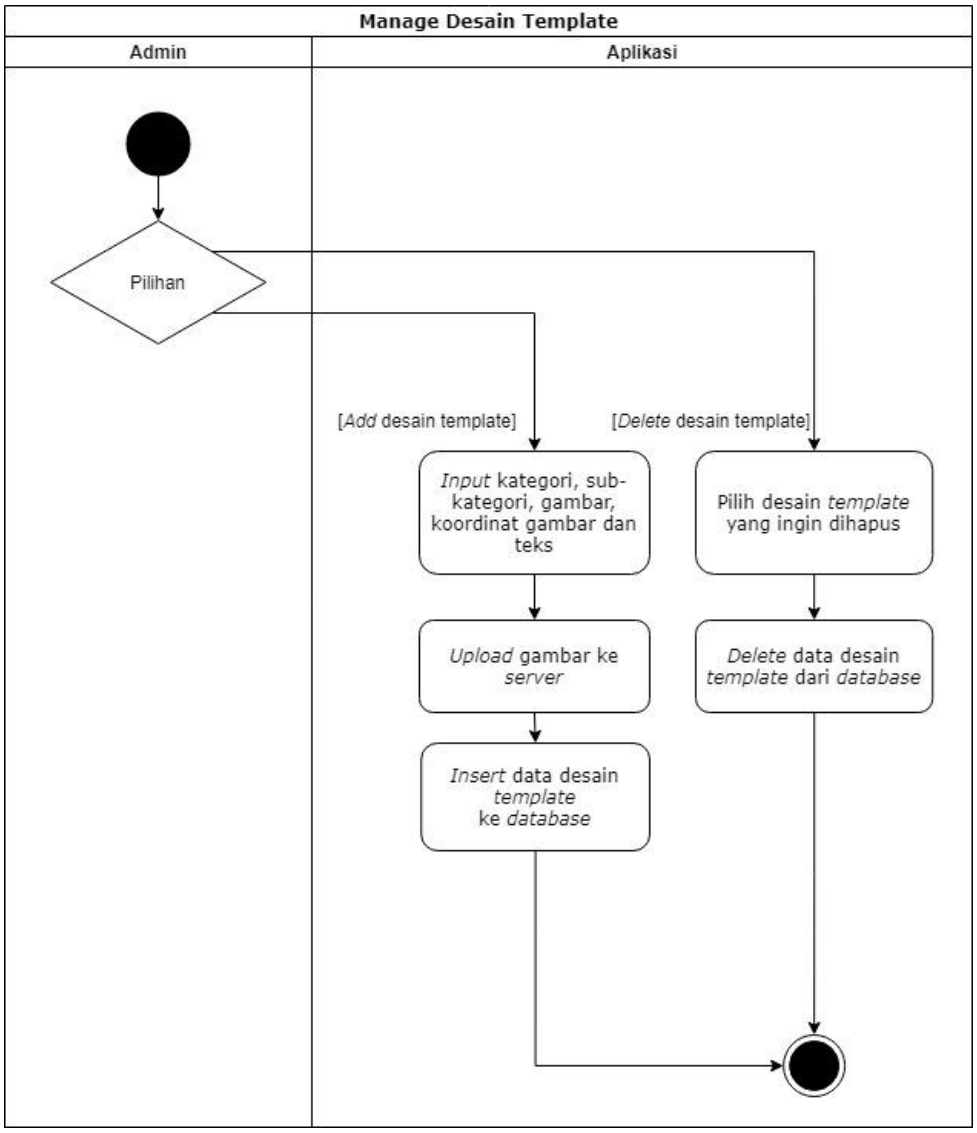


Gambar 3.5. *Activity Diagram Login*

3.4.3. *Manage* Desain *Template*

Gambar 3.6 menggambarkan *workflow* aktivitas untuk memanajemen desain *template. Admin* dapat melakukan tiga *actions* yaitu *add*, dan *delete* desain *template.* Jika *admin* memilih untuk menambahkan desain *template*, aplikasi akan meminta *admin*

untuk memasukkan kategori dan sub-kategori desain *template*, gambar, dan koordinat gambar dan teks. Daftar desain *template* ditampilkan dan *admin* dapat memilih desain *template* mana yang ingin dihapus. Kedua aksi tersebut akan mempengaruhi data di *database*, yaitu penambahan, atau pengurangan data.

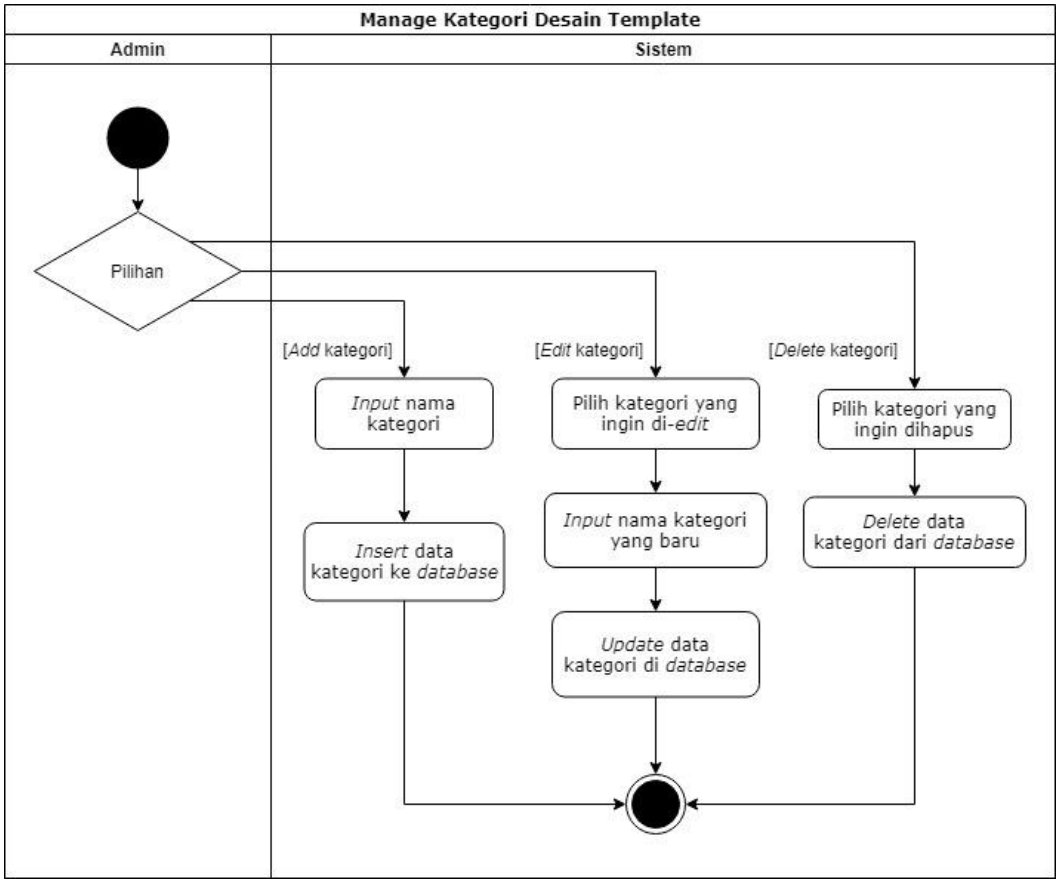


Gambar 3.6. *Activity Diagram Manage* Desain *Template*

3.4.4. *Manage* Kategori Desain *Template*

Gambar 3.7 menggambarkan *workflow* aktivitas untuk memanajemen kategori desain *template* yang disediakan. *Admin* dapat melakukan tiga actions yaitu *add*, *edit*, dan *delete* kategori. Jika *admin* memilih untuk menambahkan kategori, aplikasi akan meminta

admin untuk memasukkan nama dari kategori tersebut. Jika *admin* memilih untuk meng-*edit* kategori yang sudah ada, aplikasi akan menampilkan daftar kategori lalu meminta *update* dari *admin*. Apabila *admin* memilih untuk menghapus kategori yang sudah ada, daftar kategori akan ditampilkan dan meminta *admin* untuk memilih mana yang akan dihapus. Ketiga aksi tersebut akan mempengaruhi data di *database*, yaitu penambahan, perubahan, atau pengurangan data.

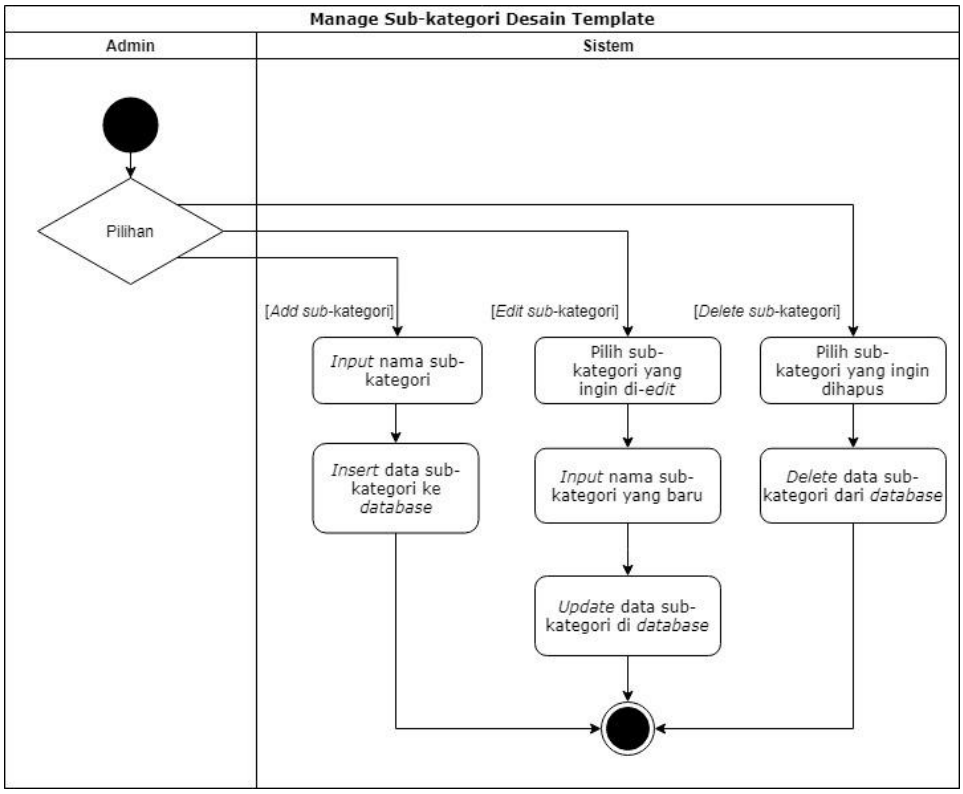


Gambar 3.7. *Activity Diagram* M*anage* Kategori Desain *Template*

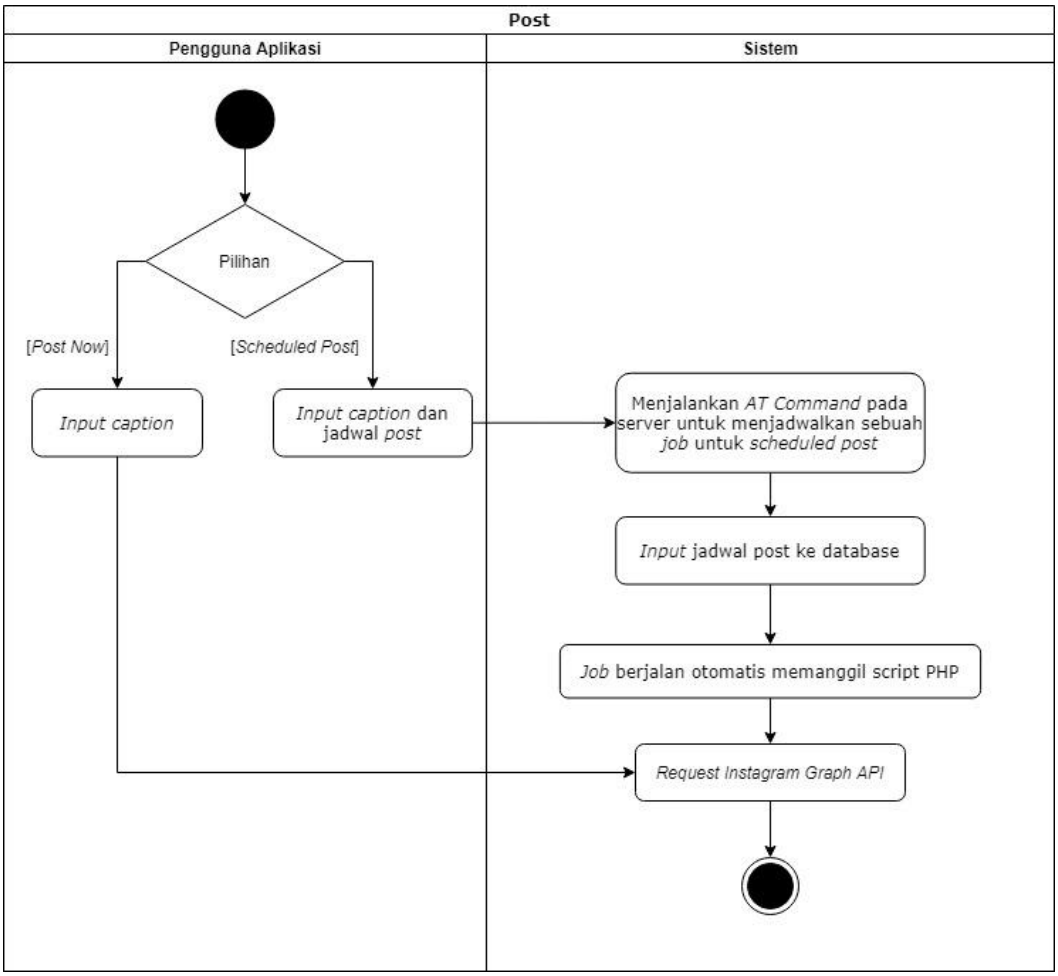
3.4.5. *Manage* Sub-Kategori Desain *Template*

Gambar 3.8 menggambarkan *workflow* aktivitas untuk memanajemen sub-kategori desain *template* yang disediakan. *Admin* dapat melakukan tiga *actions* yaitu *add*, *edit*, dan *delete* sub-kategori. Jika *admin* memilih untuk menambahkan sub-kategori, aplikasi akan meminta *admin* untuk memasukkan nama dari sub-kategori tersebut, dan menentukan golongan kategori mana untuk sub-kategori tersebut. Jika *admin* memilih untuk meng-*edit* sub-kategori yang sudah ada, aplikasi akan menampilkan daftar sub-kategori lalu meminta *update* dari admin. Apabila *admin* memilih untuk menghapus sub-

kategori yang sudah ada, daftar sub-kategori akan ditampilkan dan meminta *admin* untuk memilih mana yang akan dihapus. Ketiga aksi tersebut akan mempengaruhi data di *database*, yaitu penambahan, perubahan, atau pengurangan data.

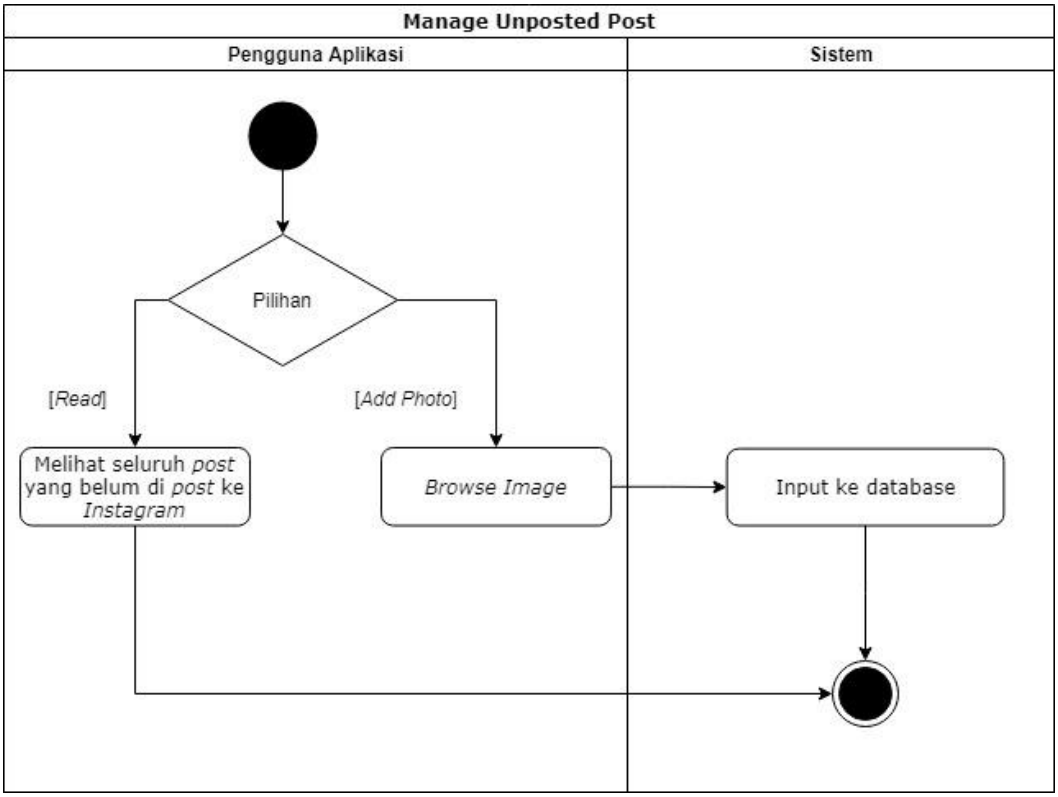


Gambar 3.8. *Activity Diagram Manage* Sub-kategori Desain *Template*



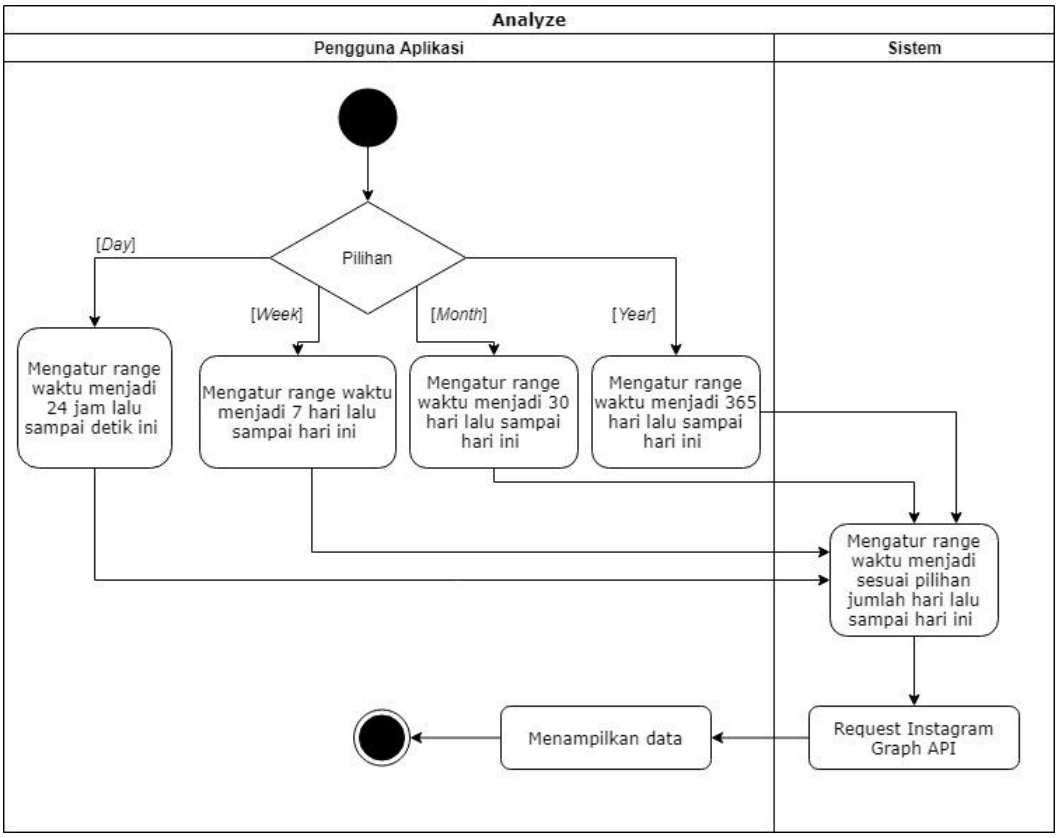
Gambar 3.9. *Activity Diagram Post Now* dan *Scheduled Post*

Gambar 3.9. menggambarkan *workflow* dua aktivitas yang dapat dipilih *user* untuk melakukan *posting*, yaitu *Post Now* dan *Scheduled Post.* Jika *user* memilih untuk *Post Now*, *user* akan diminta untuk *input* teks sebagai *caption* dan sistem akan langsung melakukan *request Instagram Graph API.* Jika *user* memilih *Scheduled Post*, *user* diminta untuk *input* teks sebagai *caption*, dan jadwal *post*. Lalu *input* dari *user* dijadikan *parameter* pada *AT Command* yang dijalankan di *server* untuk menjadwalkan sebuah *job* untuk *scheduled post*. Lalu data gambar *unposted* pada database diperbaharui agar indikator warna pada halaman *preview* ikut berubah. Kemudian *server* akan menjalankan *job* tersebut sesuai dengan jadwal yang telah ditentukan, *job* tersebut adalah menjalankan sebuah *script PHP* untuk melakukan *request Instagram Graph API* untuk melakukan *posting* dan menghapus *post* dari *database server*.



Gambar 3.10. *Activity Diagram Manage Unposted Post*

Gambar 3.10. menggambarkan *workflow* dua aktivitas yang dapat dipilih *user* untuk melakukan *posting*, yaitu *Post Now* dan *Scheduled Post.* Jika *user* memilih untuk *Post Now*, *user* akan diminta untuk *input* teks sebagai *caption* dan sistem akan langsung melakukan *request* ke *Instagram Graph API*. Jika *user* memilih *Scheduled Post*, *user* diminta untuk *input* teks sebagai *caption*, dan jadwal *post*. Lalu *input* dari *user* dijadikan *parameter* pada *AT Command* yang dijalankan di *server* untuk menjadwalkan sebuah *job* untuk *scheduled post*. Lalu data gambar *unposted* pada *database* diperbaharui agar indikator warna pada halaman *preview* ikut berubah. Kemudian *server* akan menjalankan *job* tersebut sesuai dengan jadwal yang telah ditentukan, *job* tersebut adalah menjalankan sebuah *script PHP* untuk melakukan *request Instagram Graph API* untuk melakukan *posting* dan menghapus *post* dari *database server*.

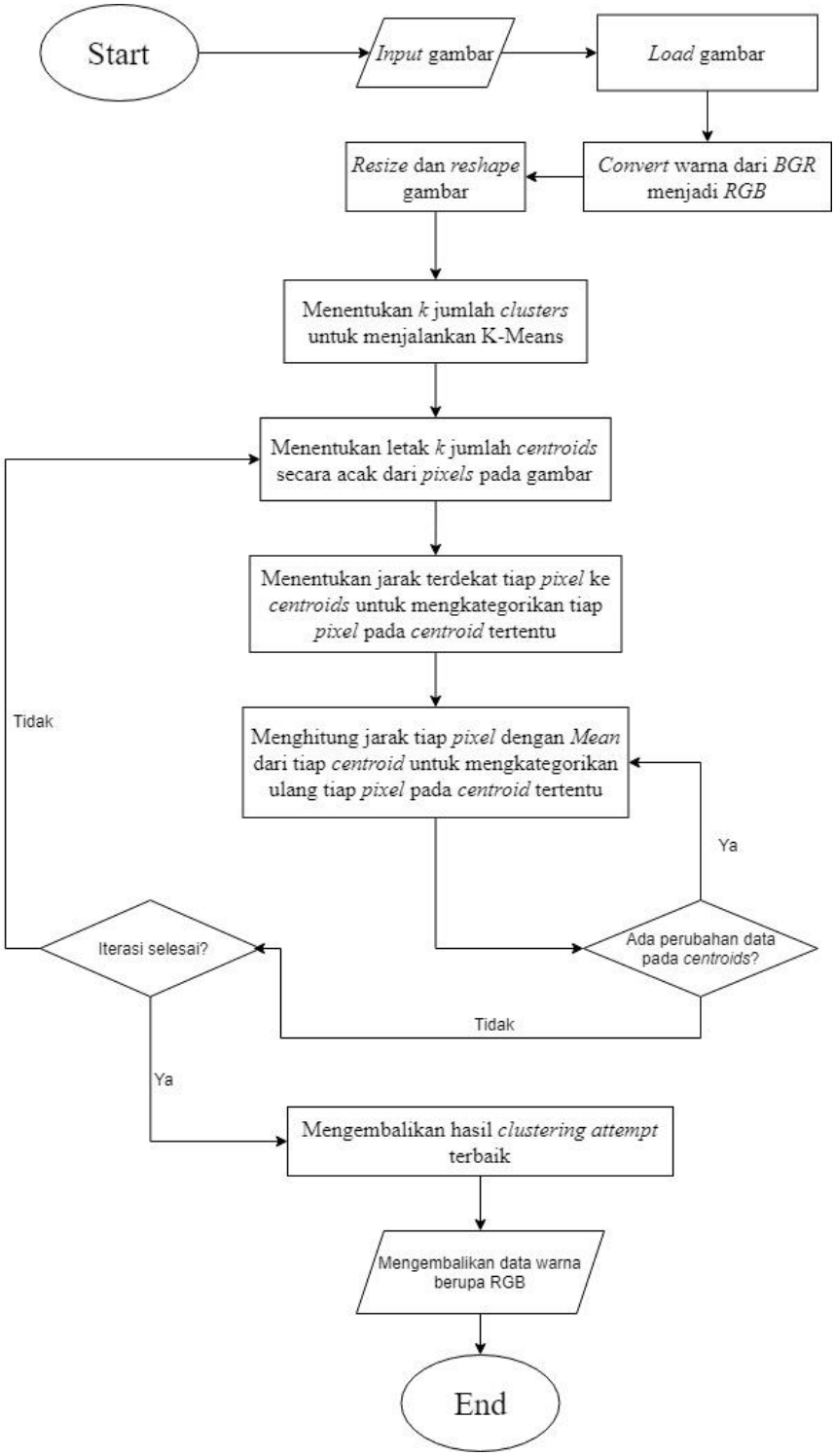


Gambar 3.11. *Activity Diagram Analyze*

Gambar 3.11. menggambarkan *workflow* proses untuk menampilkan jumlah *followers*, *following*, dan total *likes* dan *comments* yang diterima dari seluruh *posts*.

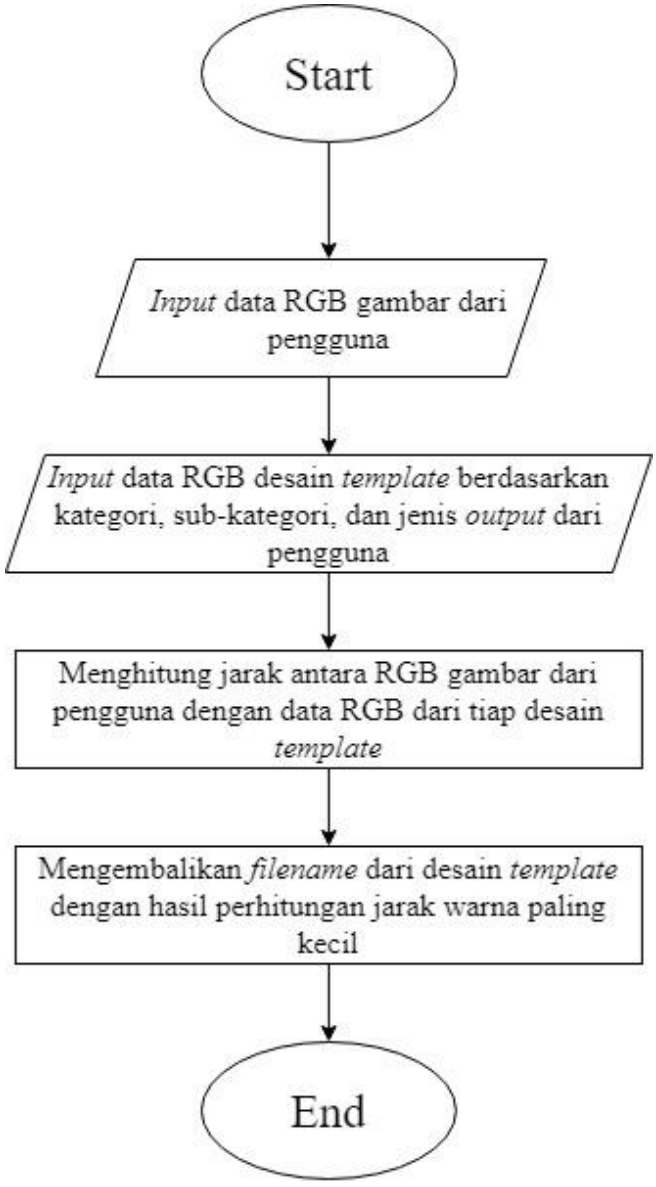
3.5. *Flowchart*

Flowchart pada Gambar 3.12 merupakan proses segmentasi warna menggunakan K-Means Clustering yang ada pada *activity diagram* pada Gambar 3.4. dan *flowchart* pada Gambar 3.13 menjelaskan mengenai alur proses segmentasi warna gambar dari pengguna dan perhitungan jarak antara warna gambar dari pengguna dengan desain *template* yang ada pada *activity diagram* pada Gambar 3.4.



Gambar 3.12. *Flowchart* K-Means Clustering

Flowchart pada Gambar 3.12 merupakan proses segmentasi gambar dari pengguna menggunakan metode K-Means Clustering. Sistem mendapatkan hasil segmentasi warna gambar dari pengguna yang akan digunakan untuk menghitung kedekatan warna dengan berbagai desain *template.*

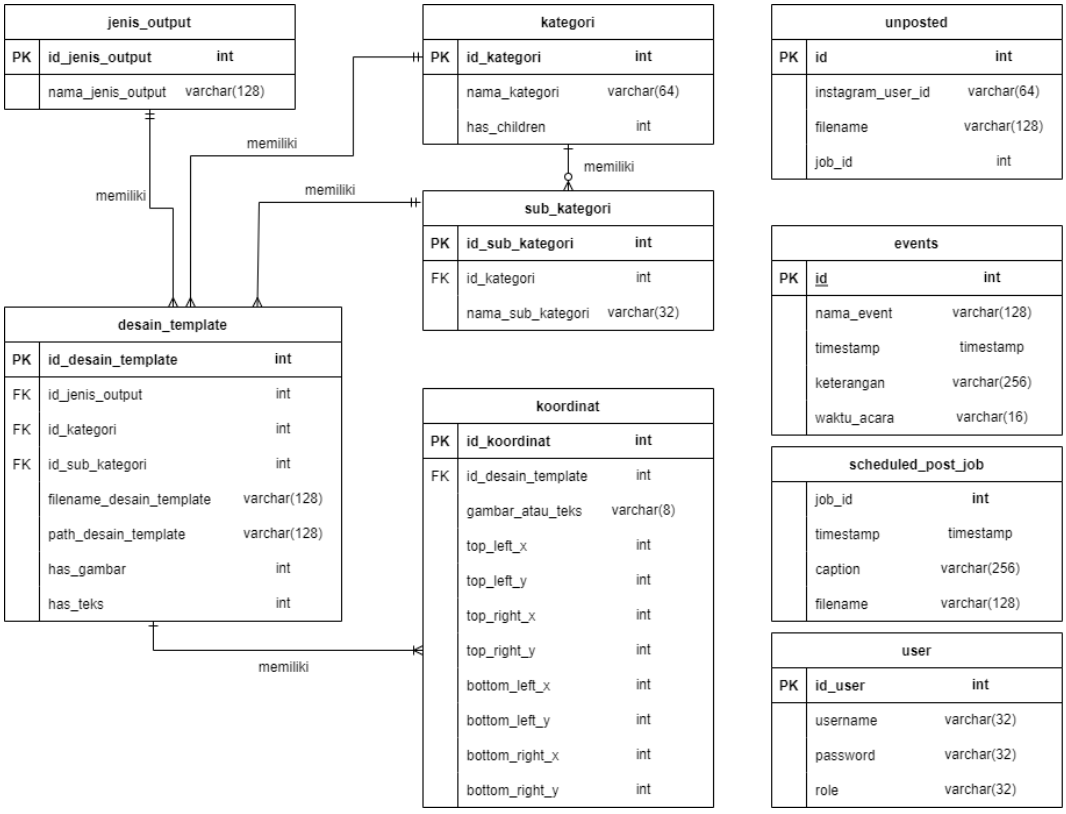


Gambar 3.13. *Flowchart* Perhitungan Jarak Warna dengan Euclidean Distance

Flowchart pada Gambar 3.13 merupakan proses perhitungan jarak warna antara gambar dari pengguna dengan berbagai desain *template* untuk mendapatkan jarak yang paling kecil.

3.6. *Entity Relationship Diagram*

Sebuah *database* sangat diperlukan untuk menyimpan data yang akan digunakan oleh aplikasi maupun *server*. *Entity Relationship Diagram* (ERD) dibuat untuk memahami relasi antar tabel yang ada. Gambar 3.14 merupakan ilustrasi *entity relationship diagram* dari sistem aplikasi.



Gambar 3.14. *Entity Relation Diagram*

3.6.1. Penjelasan Struktur Tabel

Berikut adalah penjelasan struktur data dari setiap tabel yang ada di dalam *database* aplikasi. Berdasarkan *entity relationship diagram* (ERD) yang sudah didesain seperti pada Gambar 3.14, dibuatlah *database* dengan rincian tabel yang digunakan seperti di bawah.

Tabel 3.1.

Desain Tabel jenis_output

Nama	Tipe Data	Key	Deskripsi
id_jenis_output	int(11)	Primary key	Id tabel jenis_output auto increment
nama_jenis_output	varchar(128)	-	Nama jenis output

Desain tabel jenis_output pada Tabel 3.1 digunakan untuk menyimpan data jenis *output* konten yang disediakan. Data yang disimpan berupa nama jenis *output*.

Tabel 3.2.

Desain Tabel kategori

Nama	Tipe Data	Key	Deskripsi
id_kategori	int(11)	Primary key	Id tabel kategori auto increment
nama_kategori	varchar(64)	-	Nama kategori
has_children	int(11)	-	Penentu kategori memiliki sub-kategori atau tidak

Desain tabel kategori pada Tabel 3.2 digunakan untuk menyimpan data kategori konten yang disediakan. Data yang disimpan berupa nama kategori.

Tabel 3.3.

Desain Tabel sub_kategori

Nama	Tipe Data	Key	Deskripsi
id_sub_kategori	int(11)	Primary key	Id tabel kategori auto increment
id_kategori	int(11)	Foreign key	Id primary key tabel kategori
nama_sub_kategori	varchar(32)	-	Nama sub-kategori

Desain tabel sub_kategori pada Tabel 3.3 digunakan untuk menyimpan data sub-kategori dari kategori yang telah dipilih. Data yang disimpan berupa nama sub-kategori.

Tabel 3.4.

Desain Tabel desain_template

Nama	Tipe Data	Key	Deskripsi
id_desain_template	int(11)	Primary key	Id tabel kategori auto increment
id_kategori	int(11)	Foreign key	Id primary key tabel kategori
id_sub_kategori	int(11)	Foreign key	Id primary key tabel sub_kategori
filename_desain_template	varchar(128)	-	Filename desain template yang tersimpan di dalam server
path_desain_template	varchar(128)	-	URL dari desain template yang ada di dalam server
has_gambar	int(11)	-	Jumlah gambar yang dapat dipasang di template
has_teks	int(11)	-	Jumlah teks yang dapat dipasang di template

Desain tabel desain_template pada Tabel 3.4 digunakan untuk menyimpan data desain *template*. Data yang disimpan berupa *filename,* dan *path* URL dari desain *template*.

Tabel 3.5.

Desain Tabel koordinat

Nama	Tipe Data	Key	Deskripsi
id_koordinat	int(11)	Primary key	Id tabel koordinat auto increment
id_desain_template	int(11)	Foreign key	Id primary key tabel desain_template
gambar_atau_teks	gambar(8)	-	Berisi “gambar” atau “teks” untuk menentukan koordinat yang disediakan berlaku untuk gambar atau teks
top_left_x	int(11)	-	Koordinat atas kiri sumbu X
top_left_y	int(11)	-	Koordinat atas kiri sumbu Y
top_right_x	int(11)	-	Koordinat atas kanan sumbu X
top_right_y	int(11)	-	Koordinat atas kanan sumbuY
bottom_left_x	int(11)	-	Koordinat bawah kiri sumbu X
bottom_left_y	int(11)	-	Koordinat bawah kiri sumbu Y
bottom_right_x	int(11)	-	Koordinat bawah kanan sumbu X
bottom_right_y	int(11)	-	Koordinat bawah kanan sumbuY

Desain tabel koordinat pada Tabel 3.5. digunakan untuk menyimpan data koordinat posisi gambar atau teks dari tiap desain *template*. Data yang disimpan berupa

empat titik koordinat dalam bentuk X dan Y dengan satuan ukur *pixel,* dan teruntuk gambar atau teks koordinat tersebut berlaku.

Tabel 3.6.

Desain Tabel events

Nama	Tipe Data	Key	Deskripsi
id	int(11)	Primary key	Id tabel events auto increment
nama_event	varchar(128)	-	Nama acara
timestamp	timestamp	-	Timestamp tanggal dan waktu dalam bentuk unix
keterangan	varchar(256)	-	Keterangan yang menjelaskan acara
waktu_acara	varchar(16)	-	Jam dimulainya acara

Desain tabel *events* pada Tabel 3.6. digunakan untuk menyimpan data *events*. Data yang disimpan berupa nama acara, *timestamp*, keterangan acara, dan waktu acara.

Tabel 3.7.

Desain Tabel unposted

Nama	Tipe Data	Key	Deskripsi
id	int(11)	Primary key	Id tabel image auto increment
instagram_user_id	varchar(64)	-	User Id dari akun Instagram yang login facebook di aplikasi
filename	varchar(128)	-	Nama file gambar yang akan di-post
job_id	int(11)	-	Job id yang didapatkan

			ketika menjadwalkan AT command

Desain tabel *unposted* pada Tabel 3.7. digunakan untuk menyimpan data *image* yang akan di-*post*. Data yang disimpan berupa *Instagram user id*, *filename*, dan *job id*.

Tabel 3.8.

Desain Tabel user

Nama	Tipe Data	Key	Deskripsi
id	int(11)	Primary key	Id tabel user auto increment
username	varchar(32)	-	-
password	varchar(32)	-	-
role	varchar(32)	-	Role yang mengatur hak akses *user*

Desain tabel *user* pada Tabel 3.8. digunakan untuk menyimpan data *user* yang telah terdaftar. Data ini digunakan untuk proses autentikasi *login user*.

Tabel 3.9.

Desain Tabel scheduled_post_job

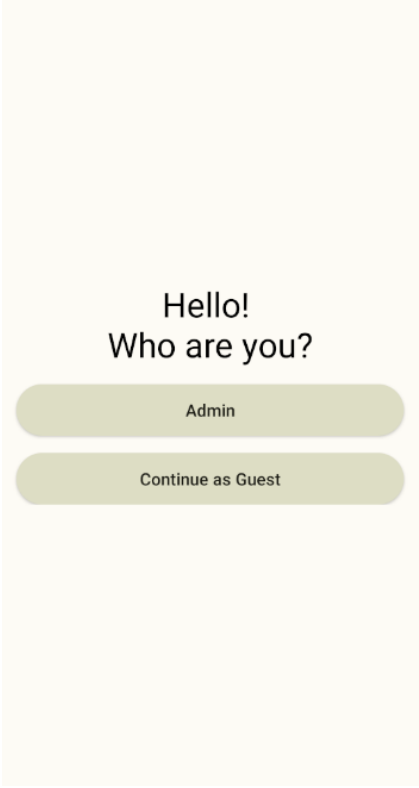
Nama	Tipe Data	Key	Deskripsi
job_id	varchar(8)	-	Job Id yang didapatkan ketika AT command dijalankan
timestamp	varchar(128)	-	Jadwal scheduled post
caption	varchar(128)	-	Teks yang digunakan untuk caption

			Instagram
filename	varchar(128)	-	Nama file gambar yang akan di-upload

Desain tabel *scheduled post job* pada Tabel 3.9. digunakan untuk menyimpan data scheduled_post_job yang telah terdaftar. Data ini digunakan untuk menyimpan jadwal *Instagram Post* beserta *job id* dari *server*.

3.7. Desain *Interface* Aplikasi

Berikut ini dijelaskan mengenai desain *interface* dari aplikasi yang akan dibuat. Desain terdiri dari halaman *home*, *login*, *create design*, *upload* gambar dan teks, hasil, serta halaman *admin* untuk melakukan manajemen data. Gambar 3.15 merupakan tampilan *home* ketika pengguna membuka aplikasi. Terdapat dua pilihan untuk mengakses aplikasi, yaitu sebagai *admin* atau sebagai *user* (*guest*).



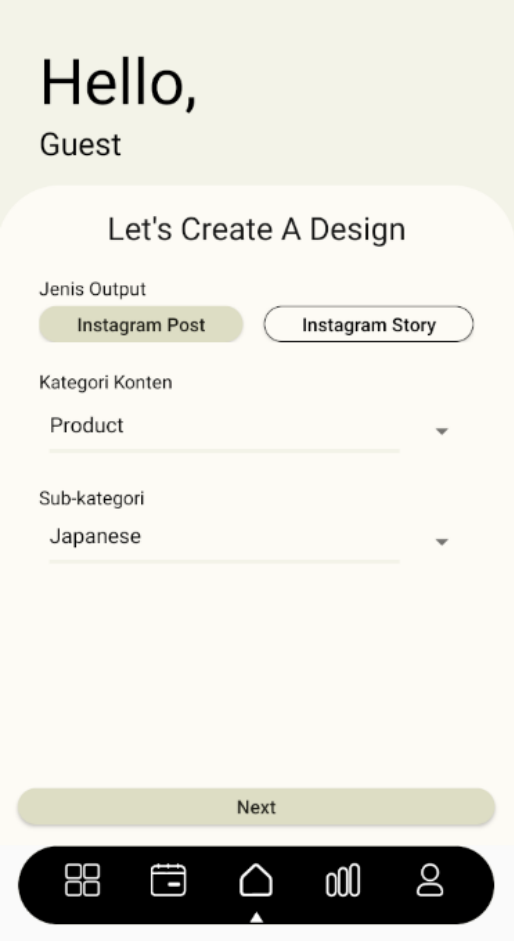
Gambar 3.15. Desain Tampilan *Home* Aplikasi

Apabila pengguna merupakan *user* biasa (*guest*), maka pengguna diarahkan ke halaman aplikasi untuk menampilkan halaman *create design*. Apabila pengguna

merupakan *admin*, maka pengguna akan diarahkan ke halaman *admin* untuk melakukan manajemen data. Sebelum aplikasi mengarahkan pengguna ke halaman *admin*, pengguna diminta untuk memasukkan *username* dan *password* pengguna. Apabila *username* dan *password* terdaftar di dalam *database*, maka pengguna diarahkan ke halaman *admin*.

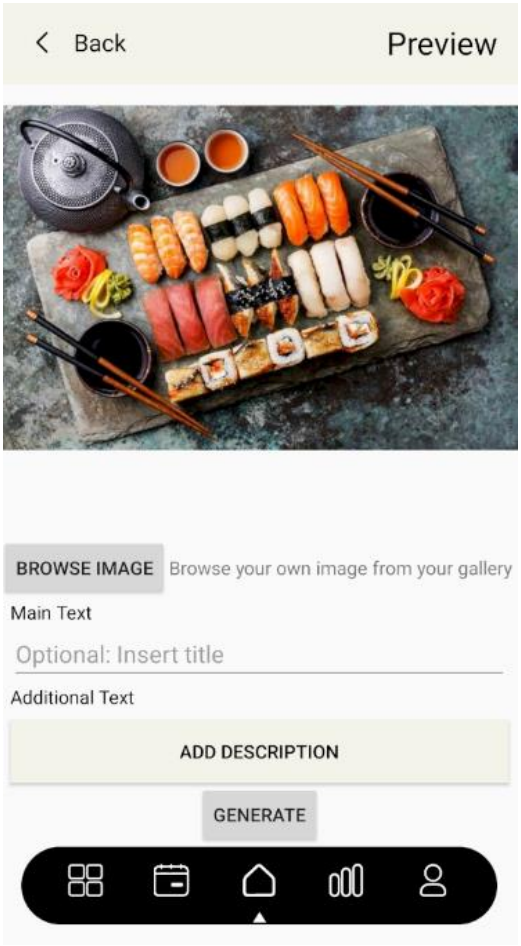
3.7.1. Desain *Interface* Untuk User

Gambar 3.16 merupakan tampilan aplikasi kepada pengguna, menampilkan halaman yang meminta jenis *output*, kategori konten, dan sub-kategori untuk pembuatan konten. Sub-kategori hanya dapat dipilih jika kategori konten memiliki sub-kategori.



Gambar 3.16. Desain Tampilan Halaman Pertama Fitur Pembuatan Konten

Apabila tombol *next* pada bagian bawah halaman ditekan, pengguna akan diarahkan ke halaman pembuatan konten yang kedua. Gambar 3.17 merupakan tampilan pembuatan konten kedua.



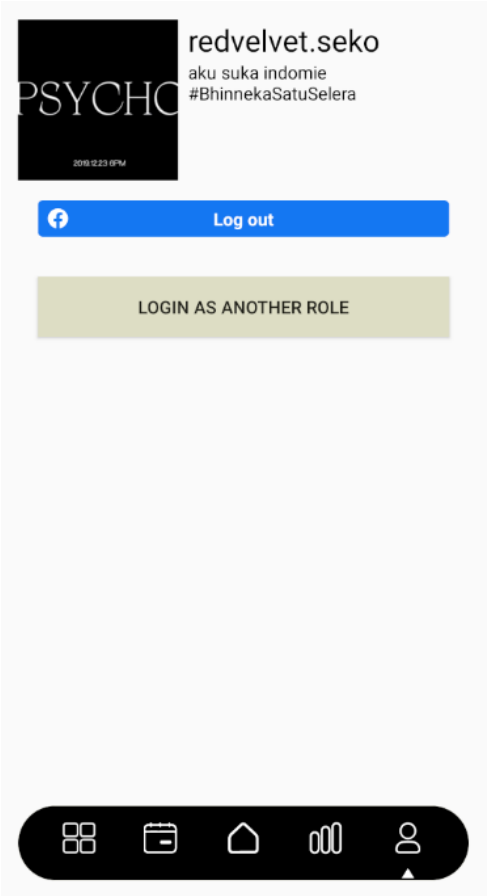
Gambar 3.17. Desain Tampilan Halaman Kedua Fitur Pembuatan Konten

Pada halaman ini, terdapat tombol *browse image* untuk memilih gambar dari *gallery user*. Kolom teks utama yang bersifat opsional pun disediakan untuk dijadikan judul pada desain *template* tertentu yang membutuhkan teks judul. Pengguna pun dapat menambahkan satu teks tambahan untuk dijadikan deskripsi pada desain *template* tertentu.

Apabila tombol *generate* pada bagian bawah halaman ditekan, gambar dan jumlah teks dikirim ke *server*. Lalu *user* diarahkan ke halaman hasil pembuatan konten. Gambar 3.18 merupakan tampilan halaman hasil pembuatan konten.

Gambar 3.18. Desain Tampilan Halaman Hasil Fitur Pembuatan Konten

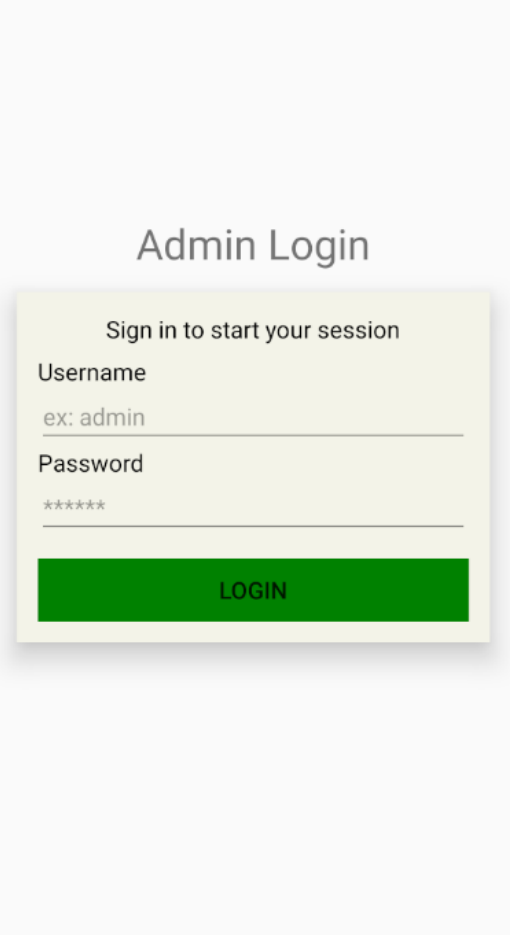
Pada halaman ini, disediakan tombol *save* untuk menyimpan hasil fitur pembuatan konten ke dalam *gallery smartphone user*. *User* diarahkan kembali ke halaman utama jika tombol *save* ditekan.



Gambar 3.19. Desain Tampilan Halaman *Profile*

Gambar 3.19. merupakan tampilan halaman *profile.* Tombol yang berwarna biru merupakan tombol dari *Facebook SDK* yang berfungsi untuk *login* dan *logout* akun *Facebook*. Gambar pada paling kiri atas halaman merupakan *profile picture* akun *Instagram Business* yang telah disambungkan dengan *Facebook Page* pada akun *Facebook user*. Teks *"*willypede" merupakan *username* dan teks dibawahnya merupakan *biography* akun *Instagram Business*. Tombol *Login as Another Role* merupakan tombol untuk pindah ke halaman *Home* untuk memulai ulang aplikasi.

3.7.2. Desain *Interface* untuk Admin



Gambar 3.20. Desain Tampilan Halaman *Login Admin*

Pada halaman ini, *user* diminta untuk memasukkan *username* dan *password* untuk melakukan *authentication* pada *user*. Jika *username* dan *password* tidak terdaftar atau tidak sesuai, *user* diminta memasukkan *username* dan *password* kembali. Apabila *username* dan *password* terdaftar pada *database*, *user* akan diarahkan ke halaman utama *admin*.