

BAB III

ANALISIS DAN PERANCANGAN

1.1 Analisis

Teknologi berkembang dengan sangat pesat saat ini, tak terkecuali aplikasi pada smartphone (Android). Aplikasi pada android banyak diterapkan di berbagai bidang, karena mudahnya untuk mengakses dan praktis dalam pemakaiannya. Dalam android sudah terdapat banyak sekali aplikasi, khususnya aplikasi pesan instan. Dari berbagai aplikasi pesan instan yang ada pada penelitian terdahulu yang telah penulis kaji, setiap aplikasi pesan instan memiliki kelebihan dan kekurangan masing-masing. Namun, sebagian aplikasi masih belum memiliki fitur untuk pencarian pertemanan, dan tidak ada sama sekali fitur pencarian pertemanan berdasarkan hal yang disukai.

1.1.1 Identifikasi Masalah

Dari pembahasan yang telah dilakukan sebelumnya, berikut beberapa identifikasi masalah yang ditemukan :

- a. Susah untuk mencari teman yang memiliki ketertarikan sama dalam aplikasi pesan instan.
- b. Ketika menemukan orang yang memiliki ketertarikan sama, belum tentu orang tersebut sedang ingin membahas hal itu.
- c. Belum adanya pencarian pengguna secara spesifik menggunakan id pengguna untuk teman yang sudah dikenal.
- d. Pengguna yang tidak menjadi teman dapat mengirimkan pesan ke kita.

1.1.2 Pemecahan Masalah

Berdasarkan identifikasi masalah diatas, berikut solusi pemecahan masalah yang penulis usulkan dengan membuat aplikasi mobile berbasis android menggunakan metode MVVM untuk proses pengembangannya.

Tabel 3.1 Pemecahan Masalah

Permasalahan	Solusi
Susah untuk mencari teman yang memiliki ketertarikan sama dalam aplikasi pesan instan.	Fitur matching, pengguna dapat memilih kategori atau ketertarikan yang diinginkan dan akan bertemu pengguna lain dengan ketertarikan yang sama.
Ketika menemukan orang dengan ketertarikan yang sama, belum tentu orang tersebut sedang ingin membahas hal itu.	Pada fitur matching, ketika pengguna telah menemukan pengguna lain yang memiliki ketertarikan sama, maka akan langsung dibuatkan room chat, jadi bisa langsung memulai obrolan.
Belum adanya pencarian pengguna secara spesifik menggunakan id pengguna untuk teman yang sudah dikenal	Fitur Tambah Teman dengan ID, ketika pengguna memiliki teman yang menggunakan aplikasi ini, maka dapat ditambahkan menjadi teman dengan memasukkan nomor Id nya.
Pengguna yang tidak menjadi teman dapat mengirimkan pesan ke kita.	Fitur pesan instan hanya bisa dilakukan ketika kedua pengguna match atau sudah terdaftar menjadi teman pada aplikasi

1.2 Perancangan

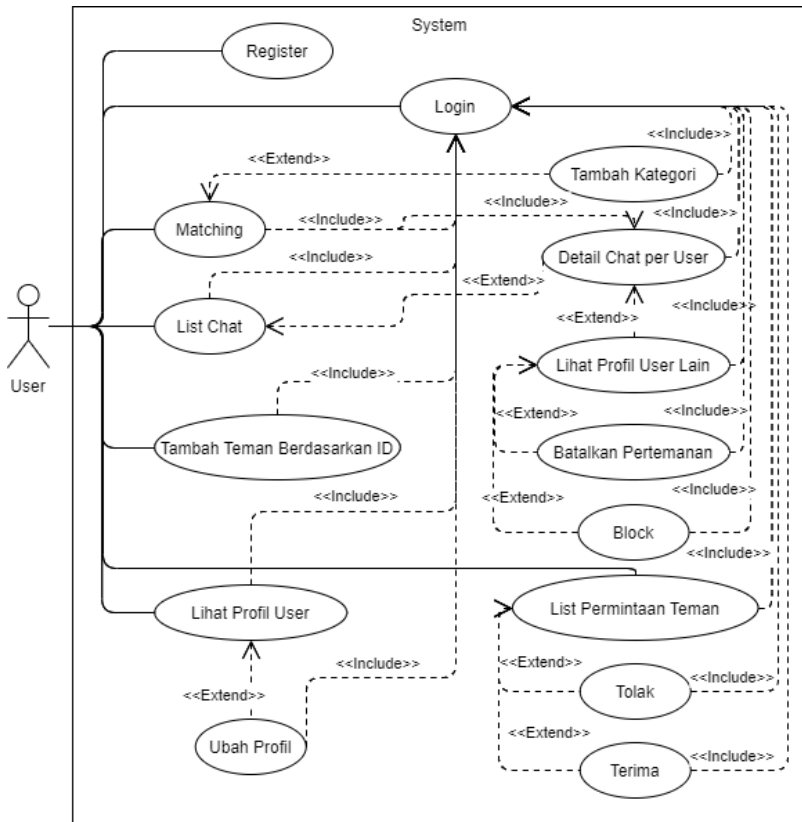
1.2.1 Perancangan Sistem

Dalam pembahasan ini, penulis akan merancang sistem yang akan diterapkan pada aplikasi Message Square. Tabel di bawah menyajikan identifikasi aktor beserta penjelasannya.

Tabel 3.2 Identifikasi Aktor

Nama Aktor	Deskripsi
User	Merupakan pengguna aplikasi Message Square.

Setelah aktor sudah diidentifikasi, dilanjutkan dengan menentukan kebutuhan dari aktor tersebut. Berikut kebutuhan dari aktor dalam Use Case Diagram:



Gambar 3.1 Use Case Diagram Aplikasi Android Message Square

Dari gambar diatas kebutuhan tiap aktor dapat dijabarkan sebagai berikut :

Tabel 3.3 Detail Use Case Diagram

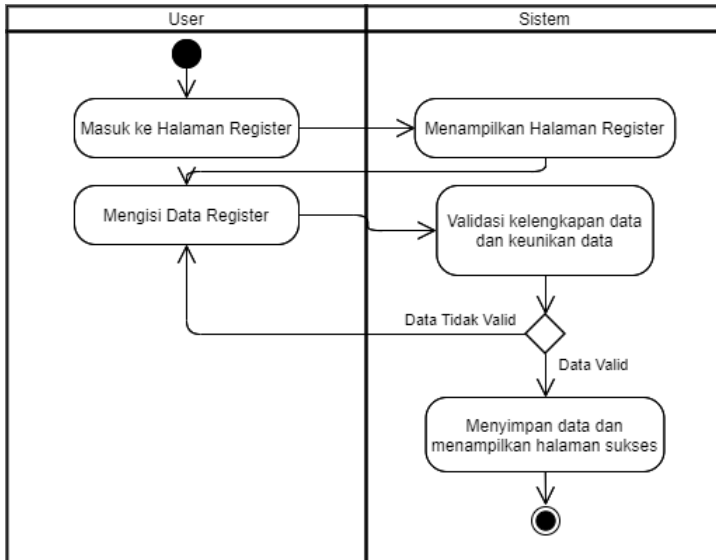
Aktor	Fitur	Deskripsi
User	Dapat melakukan registrasi	User yang belum terdaftar, dapat mendaftar terlebih dulu melalui registrasi untuk mengakses fitur di dalam aplikasi.
	Dapat melakukan login	User yang telah terdaftar, bisa login untuk dapat mengakses fitur aplikasi.
	Dapat Melakukan Fitur Matching	Melakukan pencarian teman berdasarkan kategori yang dipilih.
	Dapat Menambahkan Kategori	User dapat menambahkan kategori jika kategori yang diinginkan ketika ingin matching tidak ada.
	Dapat melihat list chat yang telah dilakukan	User dapat melihat list chat yang telah dilakukan sebelumnya.
	Dapat melihat dan melakukan chat dengan user lain.	User dapat melihat dan melakukan chat dengan user lain setelah matching atau dengan user yang telah menjadi teman.
	Dapat melihat profil teman	User dapat melihat profil yang dimiliki oleh teman.
	Dapat memblokir teman	Jika dirasa teman yang dimiliki bermasalah user dapat memblokirnya.
	Dapat membatalkan pertemanan	Jika dirasa teman yang dimiliki bermasalah user dapat membatalkan pertemanannya.
	Dapat menambahkan teman berdasarkan ID	Untuk user yang ingin menambahkan teman yang sudah dikenalnya, dapat menambahkan teman menggunakan ID nya.

Dapat melihat list permintaan pertemanan	User dapat melihat permintan pertemanan yang ada.
Dapat menerima ajakan permintaan pertemanan	User dapat menerima permintaan pertemanan yang dikirimkan user lain.
Dapat menolak ajakan permintaan pertemanan	User dapat menolak ajakan permintaan pertemanan yang dikirimkan user lain.
Dapat melihat profil	User dapat melihat profil yang dimilikinya.
Dapat mengubah profil	User dapat mengubah profil yang dimilikinya.

Activity diagram digunakan dalam pembuatan rancangan sistem aplikasi Message Square ini. Activity diagram adalah diagram yang menggambarkan alur proses yang berlangsung dalam suatu sistem. Berikut ini merupakan beberapa activity diagram yang menjelaskan setiap prosesnya.

1. *Activity Diagram Registrasi*

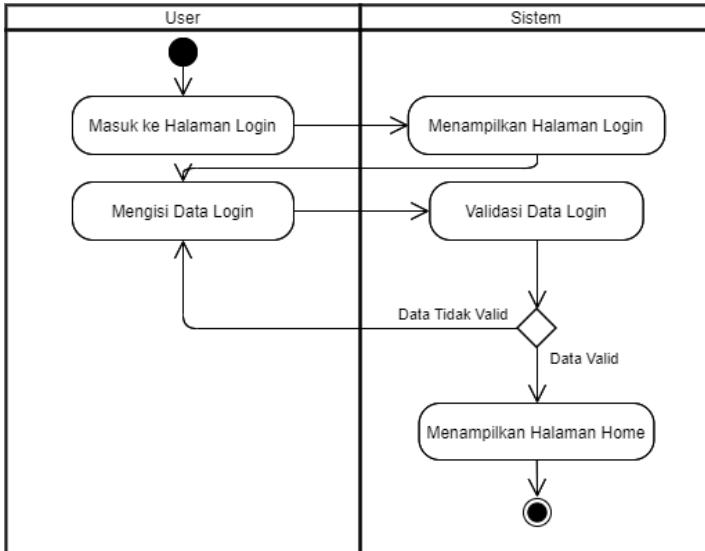
Gambar 3.2 menjelaskan proses bagaimana user melakukan proses pendaftaran pada aplikasi.



Gambar 3.2 Activity Diagram Registrasi

2. Activity Diagram Login

Ketika user sudah memiliki akun, untuk mengakses aplikasi user perlu melakukan login terlebih dahulu. Proses login tertera seperti yang ada di Gambar 3.3 di bawah ini



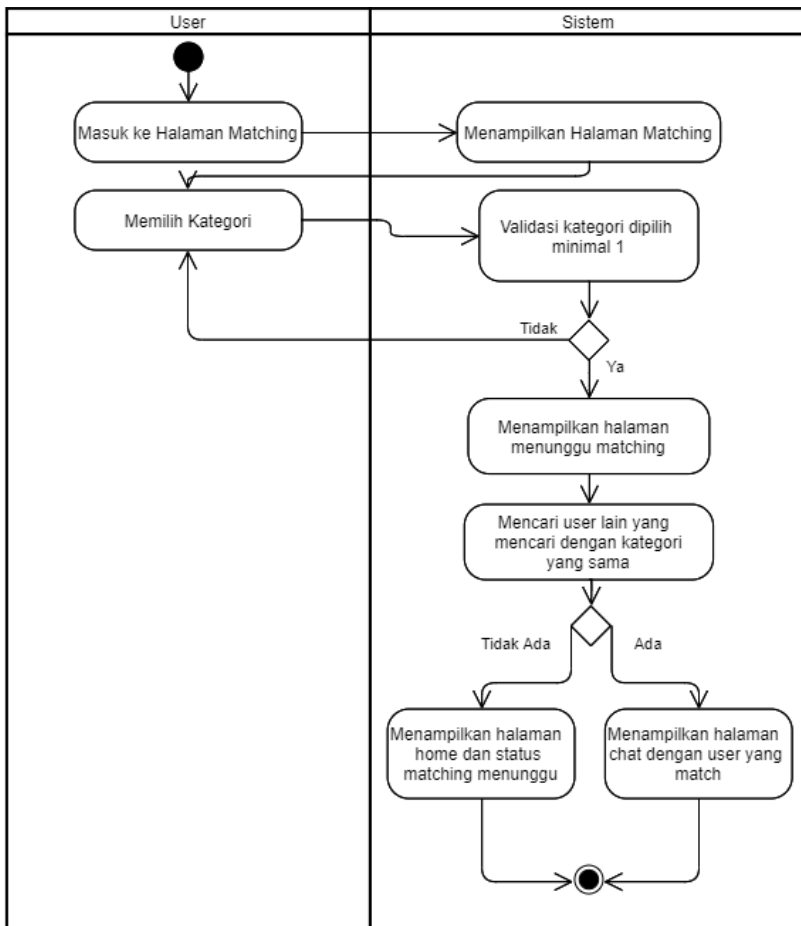
Gambar 3.3 Activity Diagram Login

3. Activity Diagram User

Fitur yang dapat diakses oleh pelanggan adalah sebagai berikut :

a. Matching

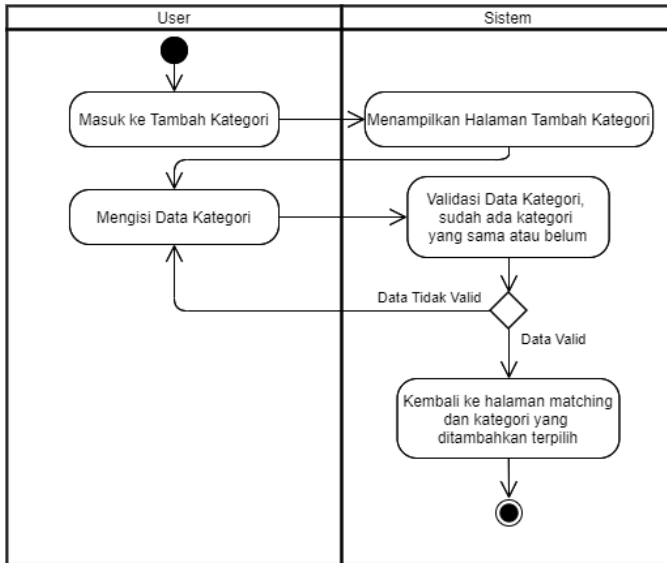
Pada gambar 3.4 dijelaskan alur proses matching, user memilih kategori terlebih dahulu dan sistem akan mencari user lain yang sesuai, dan jika belum ada akan kembali ke halaman home dan proses matching menjadi menunggu.



Gambar 3.4 Activity Diagram Matching

b. Tambah Kategori

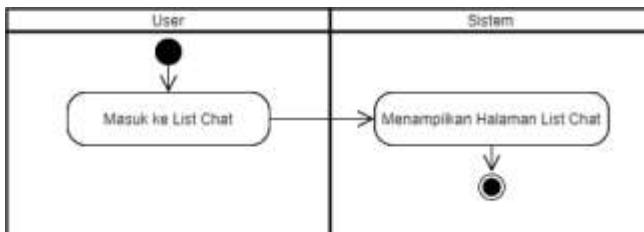
Ketika memilih kategori saat proses matching, dan user tidak menemukan kategori yang diinginkan, user dapat menambahkan kategori sendiri seperti alur yang ada di Gambar 3.5.



Gambar 3.5 Activity Diagram Tambah Kategori

c. List Chat

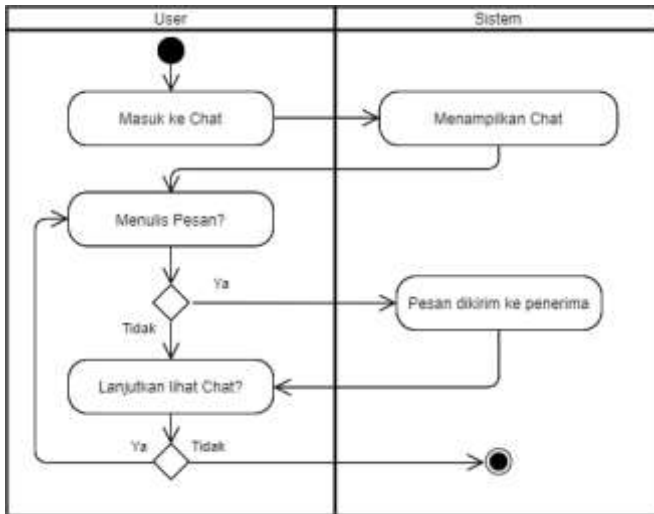
Gambar 3.6 menjelaskan alur bagaimana user melihat list chat yang ada pada aplikasi.



Gambar 3.6 Activity Diagram List Chat

d. Chat dengan User Lain

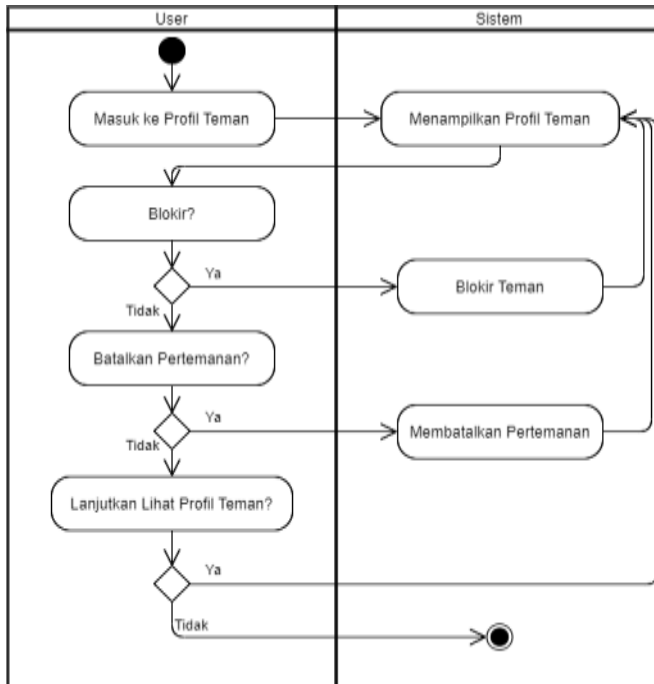
Pada gambar 3.7 dijelaskan alur untuk chat dengan user lain yang sudah menjadi teman.



Gambar 3.7 Activity Diagram Chat

e. Profil Teman

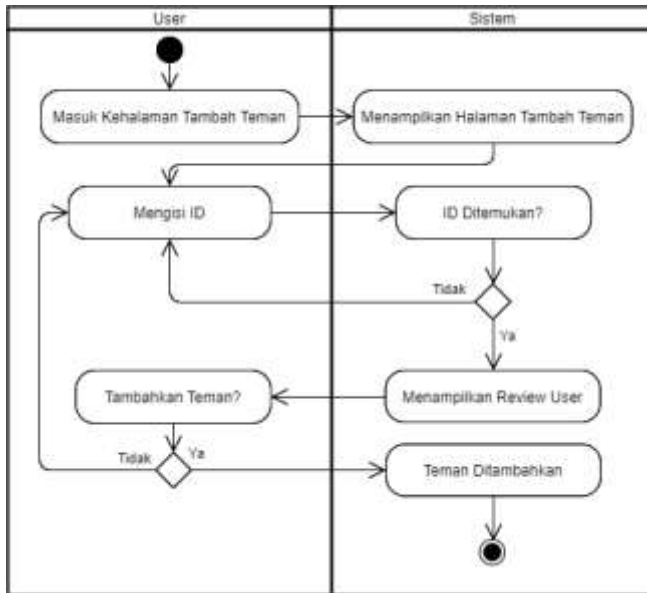
Gambar 3.8 menjelaskan alur yang ada pada halaman profil teman, user bisa melihat, memblokir, dan juga membatalkan pertemanan.



Gambar 3.8 Activity Diagram Profil Teman

f. Tambah Teman Berdasarkan ID

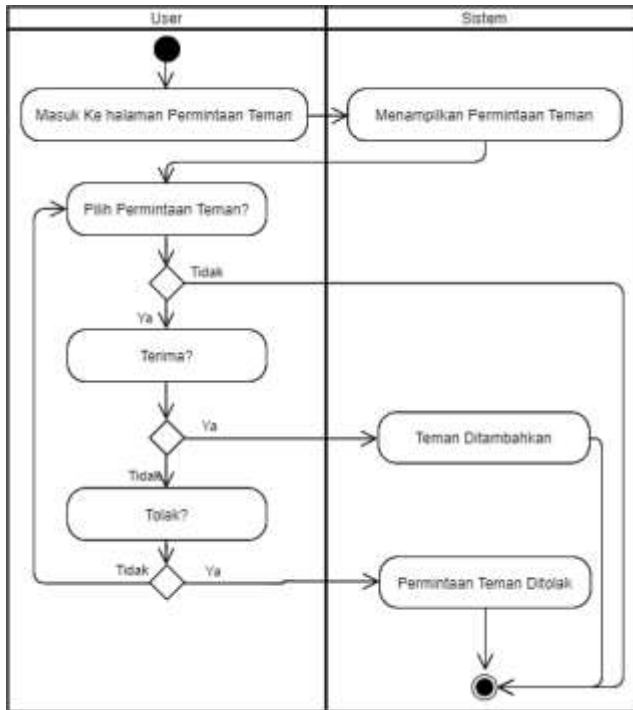
Pada gambar 3.9 dijelaskan alur untuk Menambahkan teman yang sudah dikenal menggunakan ID yang dimiliki.



Gambar 3.9 Activity Diagram Tambah Teman Berdasarkan ID

g. Permintaan Pertemanan

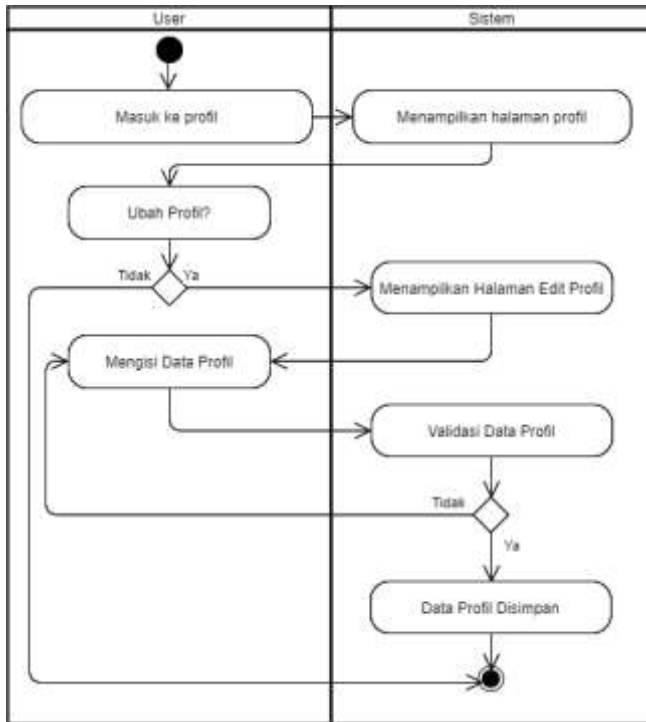
Pada gambar 3.10 dijelaskan alur untuk permintaan pertemanan, baik menampilkan, menerima, dan juga menolak.



Gambar 3.10 *Activity Diagram* Permintaan Pertemanan

h. Profil User

Pada gambar 3.11 berikut dijelaskan alur untuk melihat profil dan juga mengubah profil user.



Gambar 3.11 Activity Diagram Profil User

1.2.2 Perancangan Data

Perancangan data pada aplikasi Messages Square menggunakan Firebase Firestore sebagai basis data utama NoSQL yang menyimpan data dalam bentuk dokumen dan koleksi, ditambah dengan Firebase Authentication untuk menyimpan data kredensial user.

3.2.2.1 Struktur Koleksi dan Dokumen

a. Koleksi users

Koleksi untuk menyimpan data pengguna aplikasi Messages Square. Setiap dokumen dari koleksi ini mewakili satu pengguna dengan `userId` sebagai ID dokumen. dokumen dari pengguna ini bisa memiliki sub koleksi `chatRooms` dan `friendRequests`.

Tabel 3.4 Field dari dokumen pada koleksi users

No	Nama Field	Tipe
1	userId	String
2	uniqueId	String
3	fullName	String
4	description	String
5	color	String
6	fcmToken	Array of string
7	fcmTokenList	Array of string
8	friends	Array of string
9	blockedBy	Array of string
10	blockedUsers	Array of string

Tabel 3.5 Field dari dokumen pada koleksi sub koleksi chatRooms

No	Nama Field	Tipe
1	chatId	String
2	senderId	String
3	receiverId	String
4	active	Boolean

Tabel 3.6 Field dari dokumen pada koleksi sub koleksi friendRequests

No	Nama Field	Tipe
1	senderId	String
2	fullName	String
3	color	String
4	timestamp	Timestamp

b. Koleksi chatRooms

Koleksi untuk menyimpan dokumen percakapan antar pengguna. Setiap dokumen chatRooms memiliki sub koleksi chatRoom yang berisi detail pesan dari percakapan tersebut.

Tabel 3.7 Field dari dokumen chatRoom pada koleksi chatRooms

No	Nama Field	Tipe
1	id	String
2	senderId	String
3	receiverId	String
4	text	String
5	photoUrl	String
6	name	String
7	timestamp	Timestamp
8	readTimestamp	Timestamp

c. Koleksi categories

Koleksi untuk menyimpan dokumen kategori yang dapat dipilih pengguna saat melakukan proses *matching*.

Tabel 3.8 Field dari dokumen pada koleksi categories

No	Nama Field	Tipe
1	name	String
2	slug	String
3	description	String
4	timestamp	Timestamp
5	alikeCategories	Array of String

d. Koleksi matchingQueue

Koleksi untuk menyimpan antrian pengguna yang sedang melakukan proses *Matching* dengan kategori yang dipilih.

Tabel 3.9 Field dari dokumen pada koleksi matchingQueue

No	Nama Field	Tipe
1	userId	String
2	matchType	String
3	timestamp	Timestamp
4	categories	Array of String

3.2.2.2 Pendekatan Desain Data

Struktur data dirancang untuk mendukung alur aplikasi yang dinamis dan *real-time*, dengan pendekatan berbasis dokumen dan sub koleksi. Data pengguna, daftar teman, permintaan pertemanan, pesan, dan preferensi kategori dipisahkan dalam koleksi yang terstruktur. Firebase Authentication menjamin setiap pengguna memiliki identitas unik yang konsisten dengan dokumen pada Firestore. Pendekatan ini mendukung kemudahan dalam pencarian data, efisiensi penyimpanan, serta fleksibilitas dalam pengembangan fitur lanjutan seperti sistem matching, pemblokiran, dan pembaruan profil.

1.2.3 Perancangan User Interface / *Mock-up* aplikasi

Perancangan untuk User Interface ini dilakukan sebagai proses mengubah rancangan activity diagram menjadi dalam bentuk desain. Desain yang dihasilkan masih dalam bentuk wireframe.

3.2.3.1 *Wireframe* Registrasi

Merupakan halaman yang digunakan untuk mendaftar ke aplikasi. User diharuskan untuk mengisi *username*, *fullname*, *password* dan konfirmasi *password*.

The image shows a mobile application wireframe for a registration screen. At the top, the status bar displays '9:41' and signal icons. The title 'Register' is centered in a bold font. Below the title are four input fields: 'Username', 'Fullname', 'Password', and 'Re-type Password'. Each field has a light blue border and a small blue icon on the right. A blue 'Register' button is positioned below the input fields. At the bottom, there is a link that says 'Already have an account? Login Now!'. The entire screen is enclosed in a rounded rectangle with a drop shadow.

Gambar 3.12 *Wireframe* Registrasi

3.2.3.2 Wireframe Login

Halaman ini digunakan untuk memasuki aplikasi yang memerlukan *username* serta *password* yang telah terdaftar sebelumnya.



Gambar 3.13 *Wireframe* Login

3.2.3.3 Wireframe Tab Chat - Home

Merupakan halaman *tab chat*, pada *home* yang berisi list pesan dari *user* lain yang sudah menjadi teman kita.



Gambar 3.14 Wireframe Tab Chat – Home

3.2.3.4 Wireframe Tab Profil - Home

Merupakan halaman *tab* profil, pada *home* yang berisi detail user kita dan menu untuk mengakses ke fitur yang lain.



Gambar 3.15 Wireframe Tab Profil – Home

3.2.3.5 **Wireframe Detail Chat**

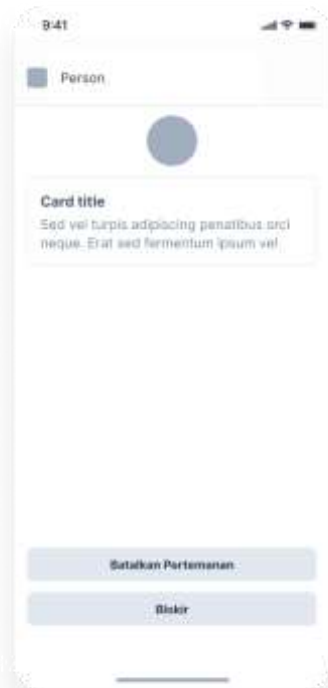
Merupakan halaman detail *chat* yang menampilkan pesan dan mengirim pesan.



Gambar 3.16 *Wireframe Detail Chat*

3.2.3.6 *Wireframe Detail Pengguna Lain*

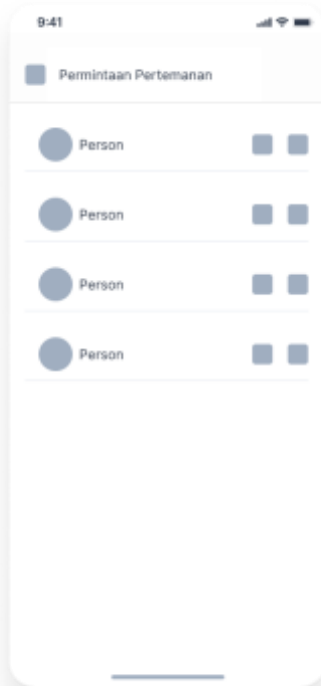
Merupakan halaman detail pengguna lain, dalam halaman ini user bisa membatalkan pertemanan atau memblokir teman tersebut jika sudah berteman, menambahkan pertemanan jika masih belum berteman, menampilkan status telah memblokir jika pengguna memblokir pengguna ini, dan menampilkan status diblokir jika pengguna saat ini telah diblokir pengguna lain tersebut.



Gambar 3.17 *Wireframe* Detail Pengguna Lain

3.2.3.7 *Wireframe List Permintaan Teman*

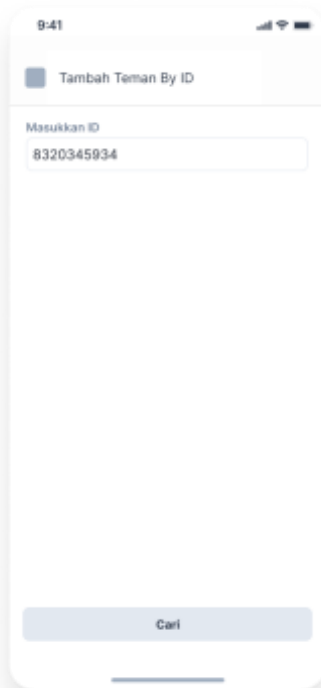
Merupakan halaman yang menampilkan *list* permintaan teman yang didapatkan oleh *user*. Di halaman ini *user* dapat menerima atau menolaknya.



Gambar 3.18 *Wireframe List* Permintaan Teman

3.2.3.8 ***Wireframe*** Tambah Teman Berdasarkan *ID*

Merupakan halaman yang untuk menambahkan teman berdasarkan *ID*. *User* hanya perlu memasukkan *ID* dari temannya untuk menambahkan.



Gambar 3.19 *Wireframe* Tambah Teman berdasarkan *ID*

3.2.3.9 ***Wireframe*** Ubah Profil

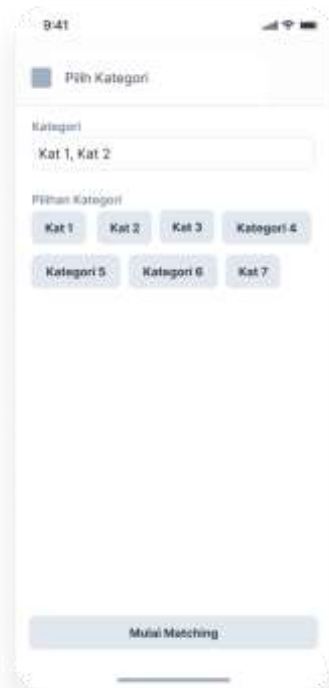
Merupakan halaman untuk mengubah profil *user*. Data yang bisa dirubah hanya *fullname* dan deskripsi saja.



Gambar 3.20 *Wireframe* Ubah Profil

3.2.3.10 *Wireframe* Matching

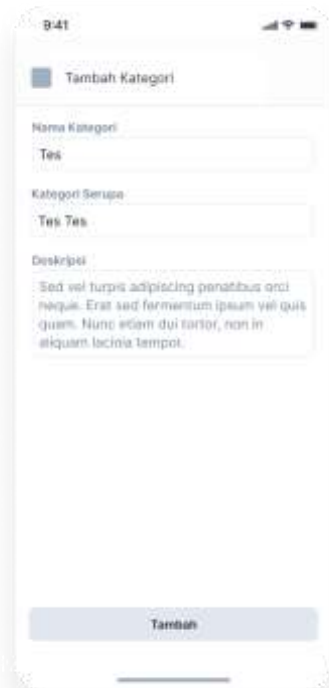
Merupakan halaman untuk memilih kategori ketika akan menggunakan fitur *matching*. *User* dapat memilih lebih dari 1 kategori.



Gambar 3.21 Wireframe Matching

3.2.3.11 Wireframe Tambah Kategori

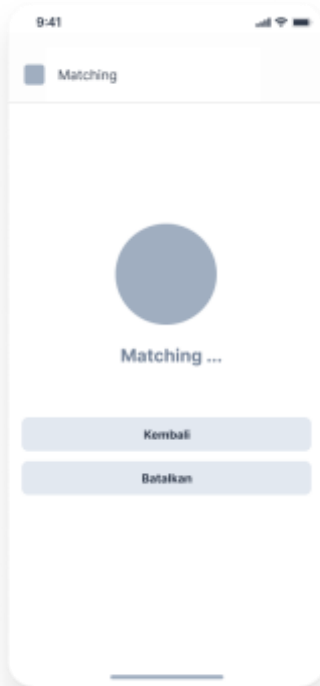
Merupakan halaman untuk menambah kategori ketika kategori yang diinginkan *user* tidak ada.



Gambar 3.22 *Wireframe* Tambah Kategori

3.2.3.12 *Wireframe* Menunggu *Match*

Merupakan halaman menunggu proses matching. *User* bisa kembali dan proses *matching* akan dilakukan hingga bertemu *user* yang *match*, atau *user* dapat membatalkan proses *matching*.



Gambar 3.23 *Wireframe Menunggu Match*

1.3 Rancangan Pengujian

Pada rancangan pengujian ini, peneliti menerapkan dua metode pengujian, yaitu unit testing dan black box testing, sebagai tahapan evaluasi akhir dari pengembangan aplikasi. Unit testing dilakukan untuk menguji logika dari unit-unit terkecil dalam aplikasi, seperti fungsi atau kelas, agar dapat dipastikan bekerja sesuai dengan yang diharapkan. Dalam pengembangan aplikasi ini, arsitektur Model-View-ViewModel (MVVM) digunakan, sehingga unit testing difokuskan pada bagian ViewModel, yang menjadi penghubung antara tampilan (View) dan data (Model). Dengan menguji ViewModel secara terpisah, pengembang dapat memastikan bahwa logika aplikasi berjalan dengan benar tanpa bergantung pada

antarmuka pengguna atau data langsung dari Firebase. Pengujian ini berfokus untuk unit yang ada di ViewModel dan juga Repository.

Sementara itu, black box testing digunakan untuk menguji fungsionalitas aplikasi dari perspektif pengguna, dengan memverifikasi apakah keluaran (output) yang dihasilkan sudah sesuai dengan masukan (input), tanpa mengetahui struktur atau kode di baliknya. Untuk fitur yang akan diuji sesuai dengan fitur yang telah ditetapkan di perancangan sistem sebelumnya.

Kedua metode ini saling melengkapi. Unit testing memberikan kepastian bahwa logika internal aplikasi pada setiap komponen berjalan stabil, sedangkan black box testing memastikan bahwa aplikasi dapat digunakan secara fungsional oleh pengguna. Dengan kombinasi keduanya, pengujian aplikasi menjadi lebih menyeluruh, baik dari sisi teknis maupun pengalaman pengguna.