Luka Spaninks

Semester 6

RB03

Version 1

Afbeelding met tekst

Automatisch gegenereerde beschrijving

Preface

The complete design phase of the project SwipeRight can be found in this document. This file will evolve over time and can always be expanded upon.

Inhoud

[1. Techstack 3](#_Toc104475297)

[2. C4 Models 4](#_Toc104475298)

[2.1 System Context 4](#_Toc104475299)

[2.2 Container diagram 5](#_Toc104475300)

[3. Entity Relationship Diagrams 7](#_Toc104475301)

[3.1 Profile Service 7](#_Toc104475302)

[3.2 Matching Service 7](#_Toc104475303)

[3.3 Recommendation Service 8](#_Toc104475304)

[4. User Interface 9](#_Toc104475305)

[4.1 Design System 9](#_Toc104475306)

[4.1.1 Typography 9](#_Toc104475307)

[4.1.2 Colors 9](#_Toc104475308)

[4.2 Mockups 10](#_Toc104475309)

[4.3 Assets 11](#_Toc104475310)

[4.3.1 Logo 11](#_Toc104475311)

[4.3.2 App Icons 11](#_Toc104475312)

[4.3.3 Like & Dislike Buttons 11](#_Toc104475313)

[4.3.4 Default Buttons 11](#_Toc104475314)

# 1. Techstack

The technologies which will be used for the SwipeRight techstack are defined below.

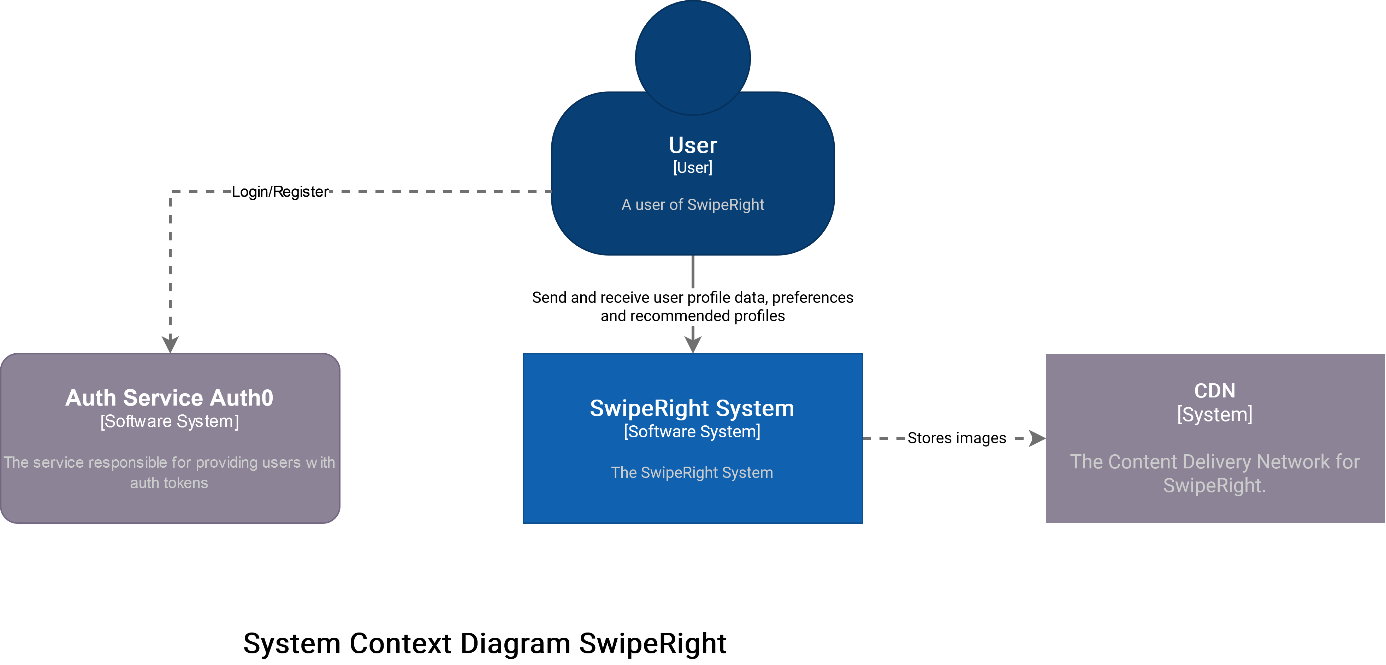
|  |  |  |
| --- | --- | --- |
| Feature | Description | Motivation |
| Security | OAuth 2.0 | Secure and widely used (industry standard) |
| Server script | Go | Performant, Scalability & Nice syntax |
| Server Framework | Gin | Gin is a performant and widely used backend framework for Go. |
| Client Technology | Flutter | Flutter is a performant cross-platform technology, It’s used a lot for native development. |
| Client script | Dart | Dart is the language used in Flutter |
| Database 1 | Apache Cassandra | Performant, Secure & highly scalable |
| Database 2 | Redis | Flexible data structures, performant & simplicity |
| Server OS | Ubuntu | Backend programs should eventually be able to run in every environment |
| Client OS | Android | In the development phase my target OS is android because it is open, it might be extended to IOS in a later stage of production. |

|  |  |  |
| --- | --- | --- |
| Container | Docker | Sensible default |
| Orchestration | Kubernetes | Sensible default |
| Gateway | Kong | Highly customizable ingress/gateway etc. |
| Monitoring | Grafana & Prometheus | Sensible default |
| Auth | Auth0 | Nice free tier and widely used |

# 2. C4 Models

## 2.1 System Context

The idea is to work with a microservice architecture which is almost completely dependent on internal services. A system like a content delivery network should however be outsourced to an external system though. Vendors like amazon have servers everywhere in the world which helps a great deal with performance. This is a resource I unfortunately do not have access to.



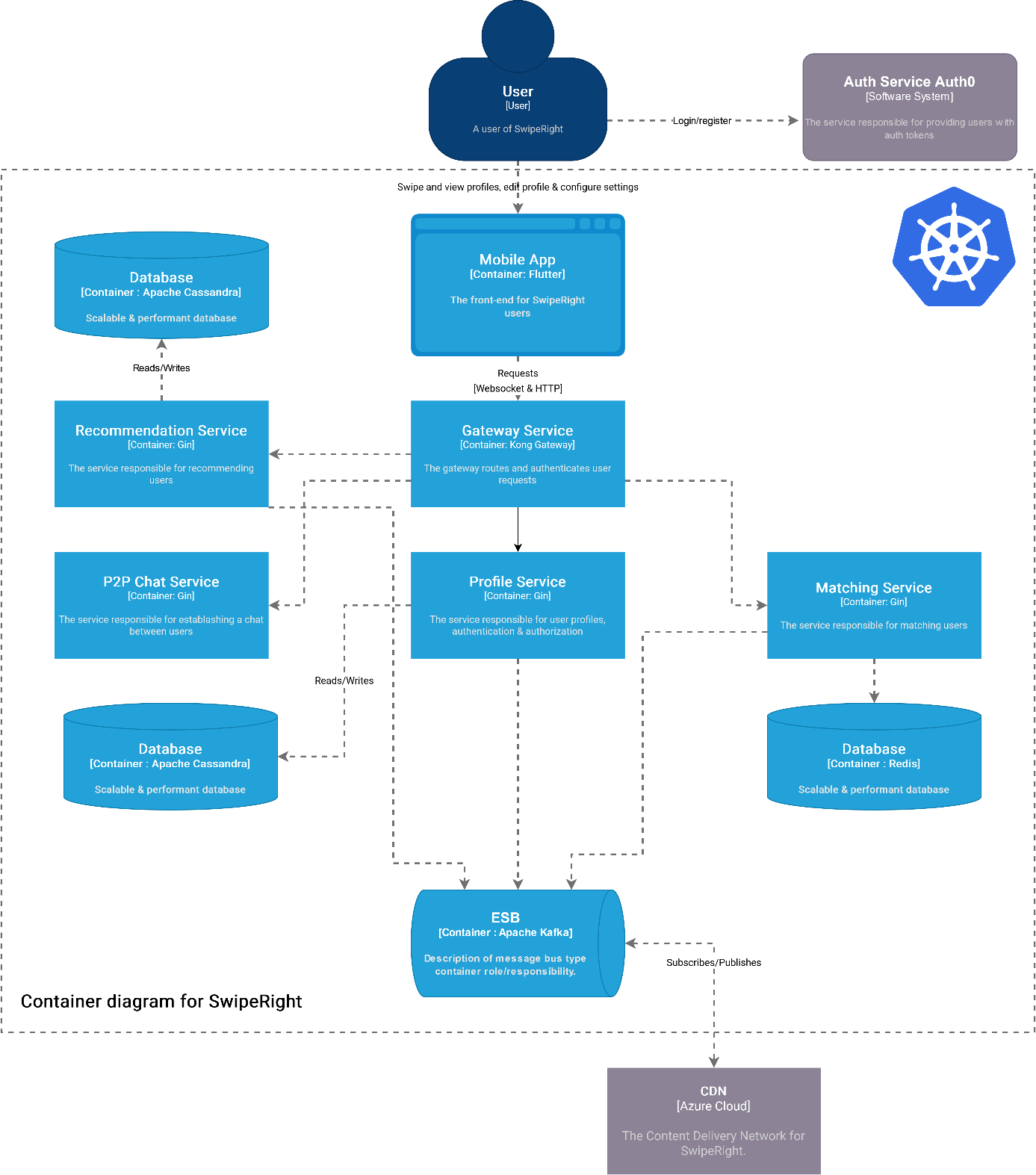
## 2.2 Container diagram

The container diagram contains a gateway which connects the clients to the service bus. The service bus makes asynchronous communication between all services possible.

The complete stack will run in a Kubernetes cluster and will therefore be easily scalable and has the potential for high availability.

All services will run in docker containers so I don’t have to worry about OS compatibility.

Authentication will be handled by the gateway and the services will authorize requests. The access tokens will be provided by Auth0.



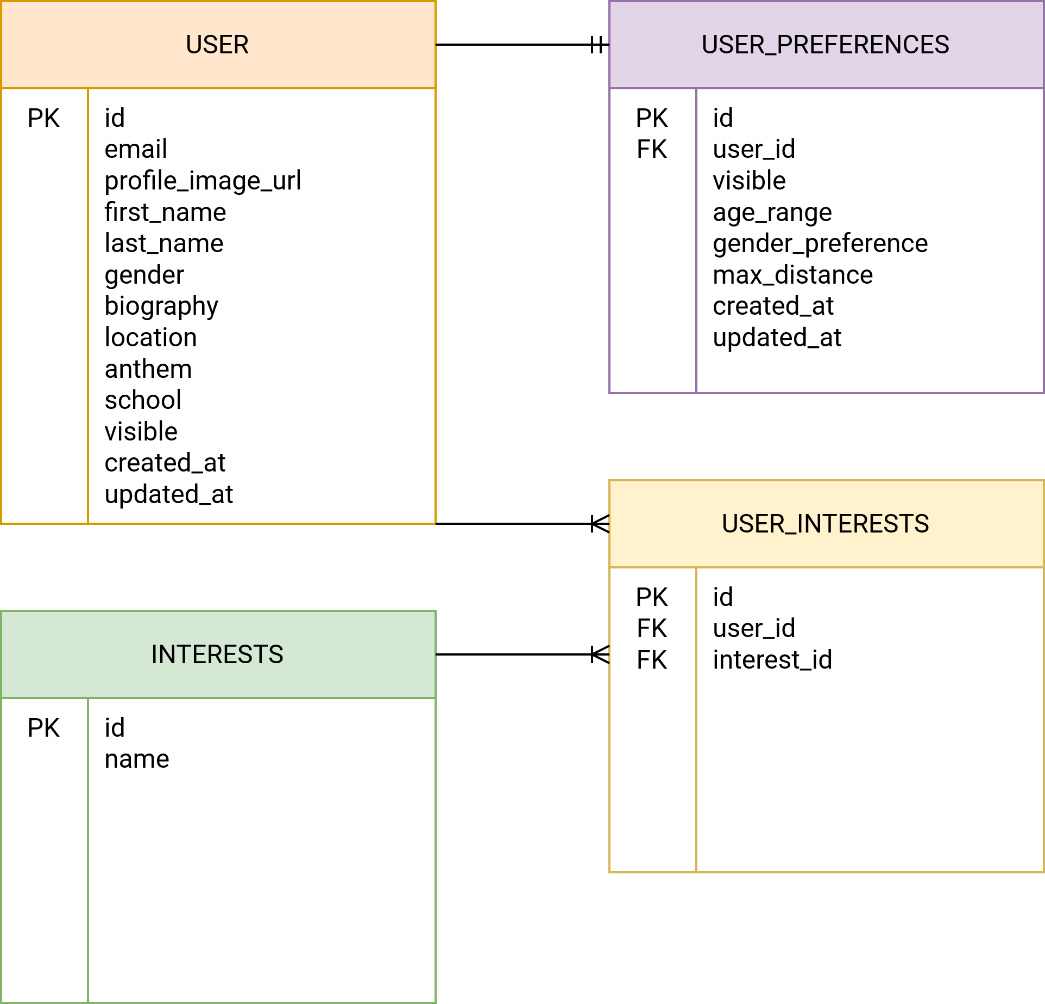
In order to generate URL’s for images I want to use Azure functions in combination with Azure blob storage. The azure function will trigger when a message is send to Kafka with a request to generate a URL. This process will happen entirely asynchronous.

Every custom build service will contain a metrics endpoint which can be scraped by Prometheus. These metrics can be visualized in a Grafana dashboard.

# 3. Entity Relationship Diagrams

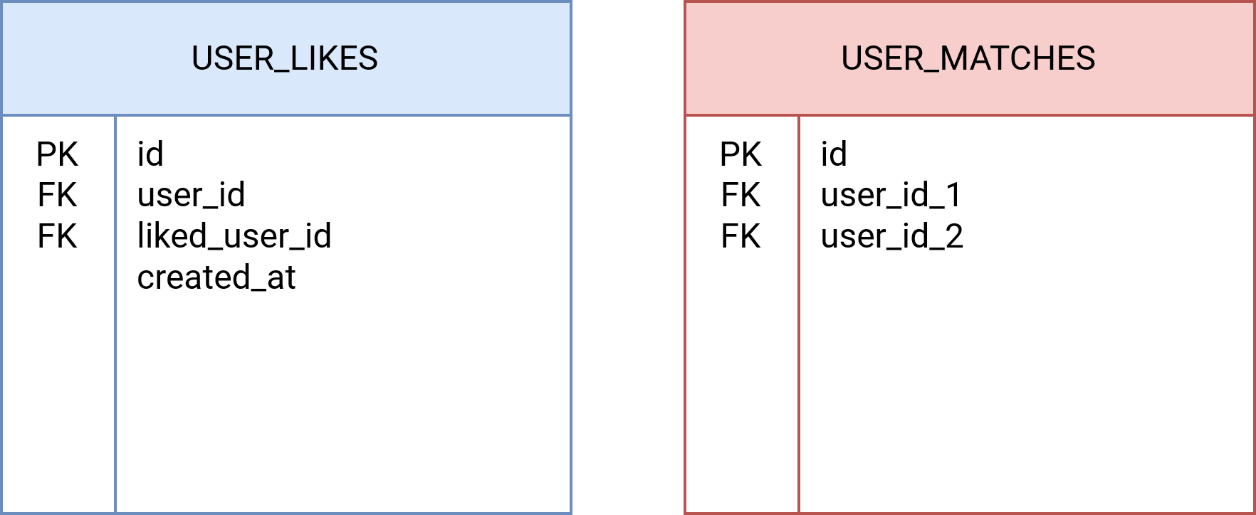
The entities are defined per microservice.

## 3.1 Profile Service

The profile service will store all user related data.

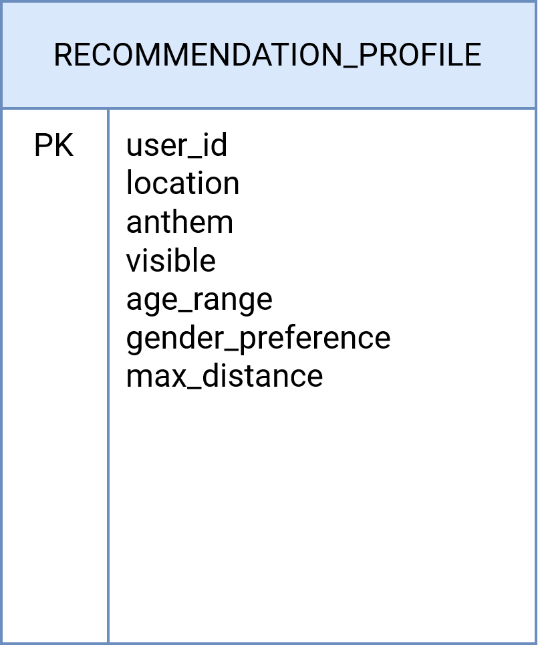
## 3.2 Matching Service

The matching service has the responsibility to keep track of the liked users and needs to update the USER\_MATCHES when two users have liked each other.



## 3.3 Recommendation Service

The recommendation service is in charge of making sure users get other users recommended to them according to preferences.

 Afbeelding met tafel

Automatisch gegenereerde beschrijving

# 4. User Interface

In this chapter the designs and standards are defined.

## 4.1 Design System

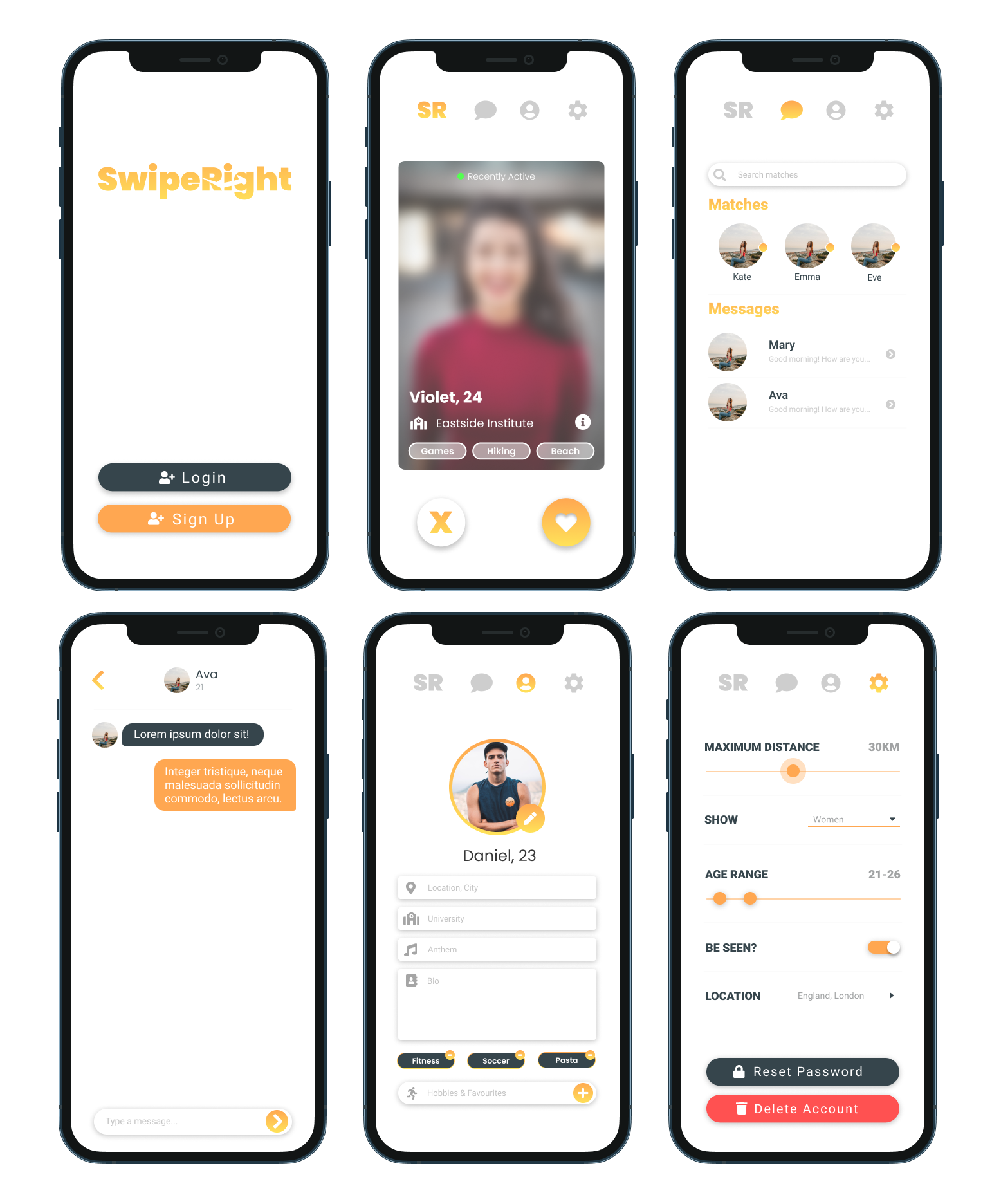
### **4.1.1 Typography**

|  |  |
| --- | --- |
| **Name** | **Font** |
| Logo Font | Poppins |
| App Font | Roboto |
| |  |  | | --- | --- | | Name | **Font Size (px)** | | XS | 12 | | S | 14 | | M | 18 | | L | 20 | | XL | 24 | |  |

### **4.1.2 Colors**

|  |  |
| --- | --- |
| **Name** | **Color** |
| **Logo Gradient Top** | #FFA751 |
| **Logo Gradient Bottom** | #FFE259 |
| **App White** | #FFFFFF |
| **App Dark** | #373737 |
| **App Grey** | #CECECE |
| **App Green** | #54FF51 |
| **App Red** | #FF5151 |

## 4.2 Mockups



*\*\* Mockups are created with* [*Figma*](https://www.figma.com/)

*\*\*\* Pictures used are from* [*Unsplash*](https://unsplash.com/)

## 4.3 Assets

### 4.3.1 Logo



### 4.3.2 App Icons



### 4.3.3 Like & Dislike Buttons



### 4.3.4 Default Buttons

