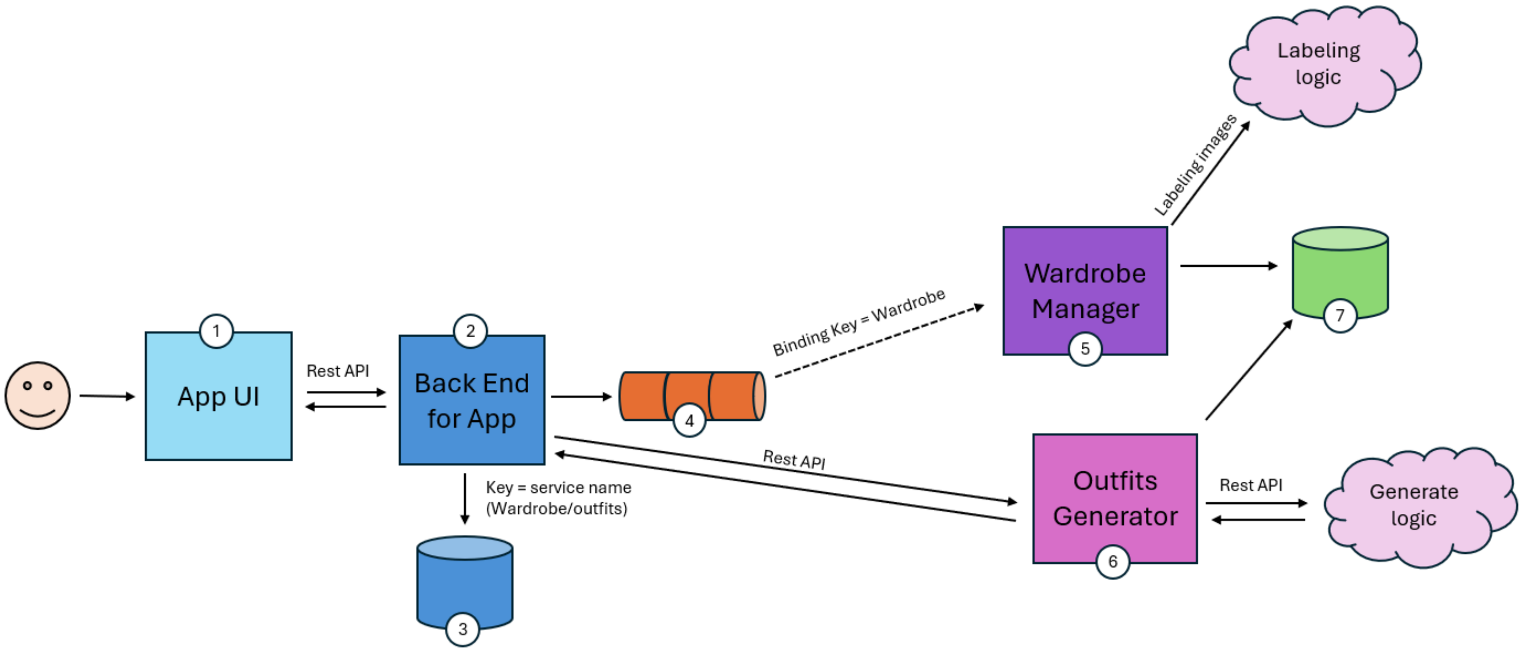
StyleMate: High Level Design

Team: StyleMate
Members: Stav Duck, Tal Dayan

answers the below questions:

1. What are the system's main components?

Main Components Diagram



1) **App UI** –
Will present the functionality of the app, planned to be implemented with React

2) **Back End for App** –
Listen to request (communicate by API) from the UI side.
Will have sign in / login flow using DB with the user's information

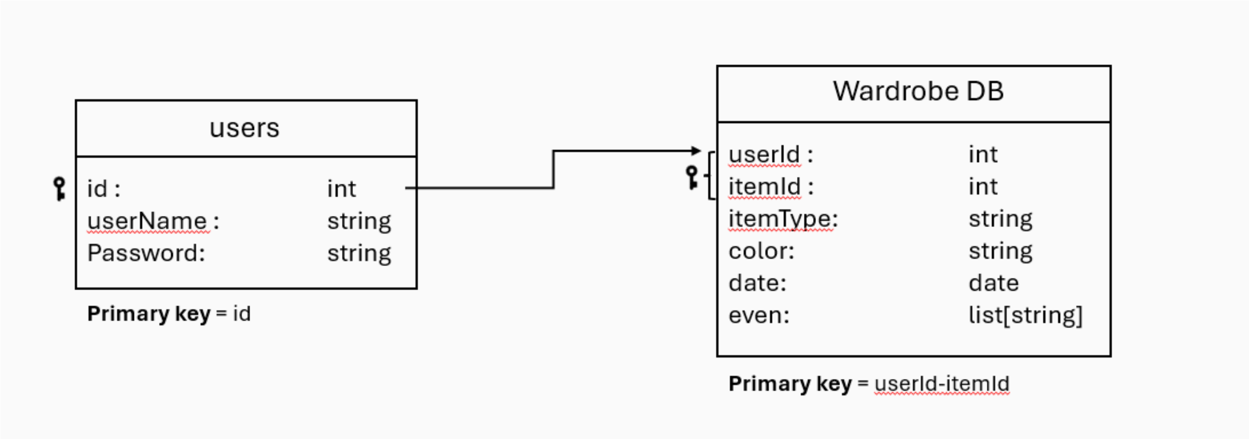
3) **Users Data Base** –
SQL data base contains the user information

4) **Message Broker** –
will transfer the actions to the relevant services, considering using RabbitMQ
5) **Wardrobe Manager** –
store and manage all the user’s clothes, update (if dirty or not) delete or add new items to it’s data.
The user can upload new images of the clothes, the service will be using a Labeling algorithm to get the information from the images and store in the DB.

6) **Outfit Generator** –
The service will communicate with the wardrobe DB and use a generating logic to generate an outfit for the user according to it’s needs.

7) **Wardrobe Data Base** –
After sending the image of an clothing item to the labeling logic the DB will save the object metadata (type: shirt/pants/ shoes, color, date etc..)
planned to be implemented with Mongo DB

Data Model:



users	
id :	int
userName :	string
Password:	string

Primary key = id

Wardrobe DB	
userId :	int
itemId :	int
itemType:	string
color:	string
date:	date
even:	list[string]

Primary key = userId-itemId

API Description:

Using REST API between the app UI and the backend:

- /items/{userId}:
  - GET - get all wardrobe
- /item/{userId}/{itemId}:
  - GET - get specific item
  - DELETE - delete specific item from wardrobe
- /createItem/{userId}:
  - POST - create new item

- /generateOutfit/{userId}:
    - GET - get generated outfit

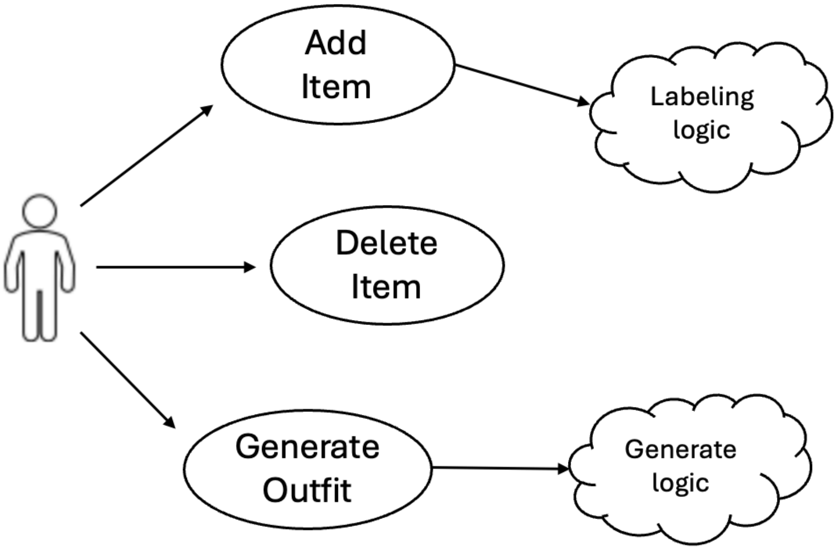
Using messaging queue:

- Sending a message for creating a new item as a json format. It will contain the userId and item’s image.
  ```
  {
          message_id: {int}
          time_stamp: {time}
          user_id: {int}
          Image: {serialized image}
  }
  ```
- Sending a messageas a json format for deleting an item from the wardrobe. It will contain the userId and itemId.
  ```
  {
          message_id: {int}
          time_stamp: {time}
          user_id: {int}
          Item_id: {int}
  }
  ```

Using REST API between the backend and the outfit generator:

- /createOutfit/{userId}
    - GET - get generated outfit

2. How will the main users’ use cases look alike?

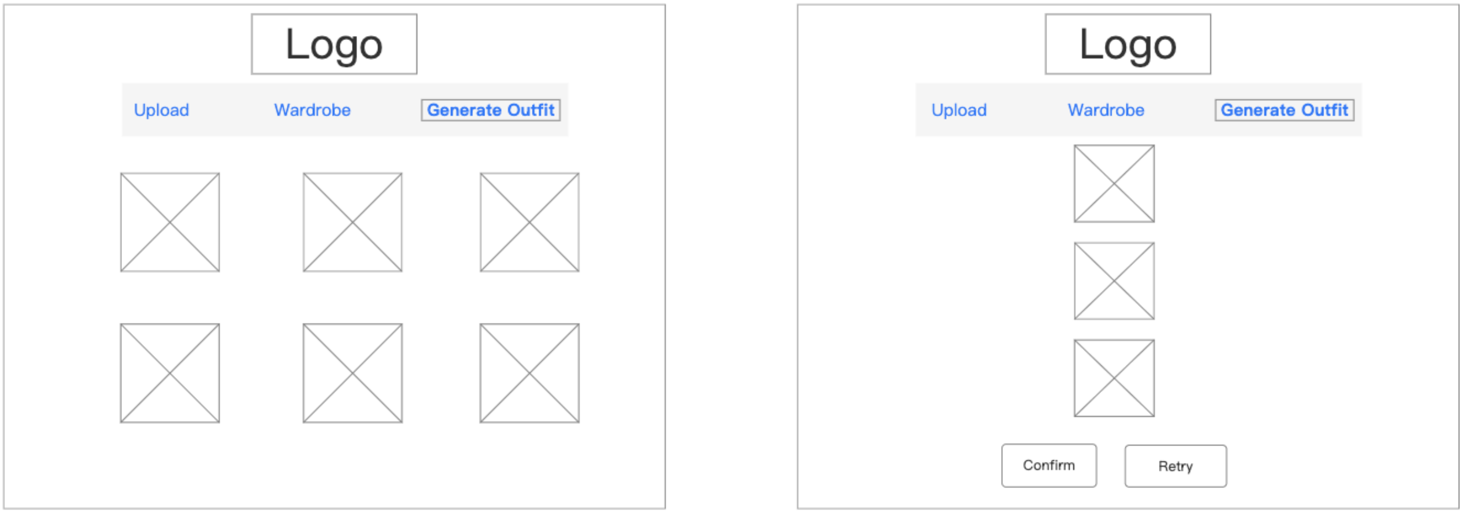


3. What is the Front End (i.e. UI) technology you are aiming to use? does it have its own architecture (e.g. React usually comes with a backed nodeJS behind it, while native JS doesn’t):

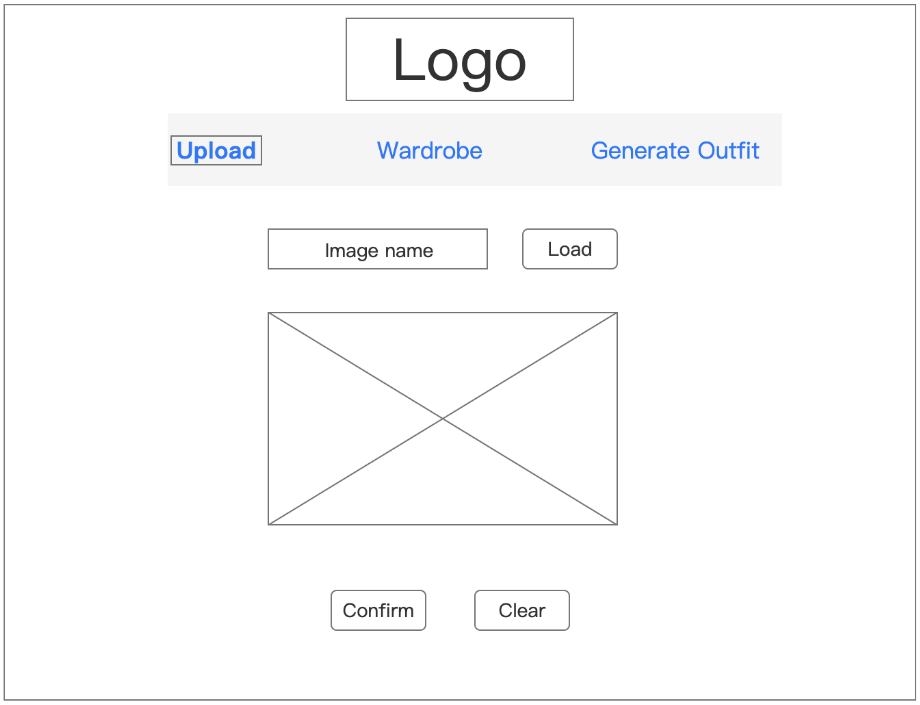
   Planned to be implemented with React.

4. Mocks for main pages:
   This is where you draw and sketch very high-level drawings of the main screens that the main users are expected to work with.
   This needs to give a high-level perspective as to how the system will look.
   Don’t worry, you don’t need to go into details and you are not totally committed to this screen looks in the future
   Power point, Canva, photoshop, figma - are all legit tools for this part..

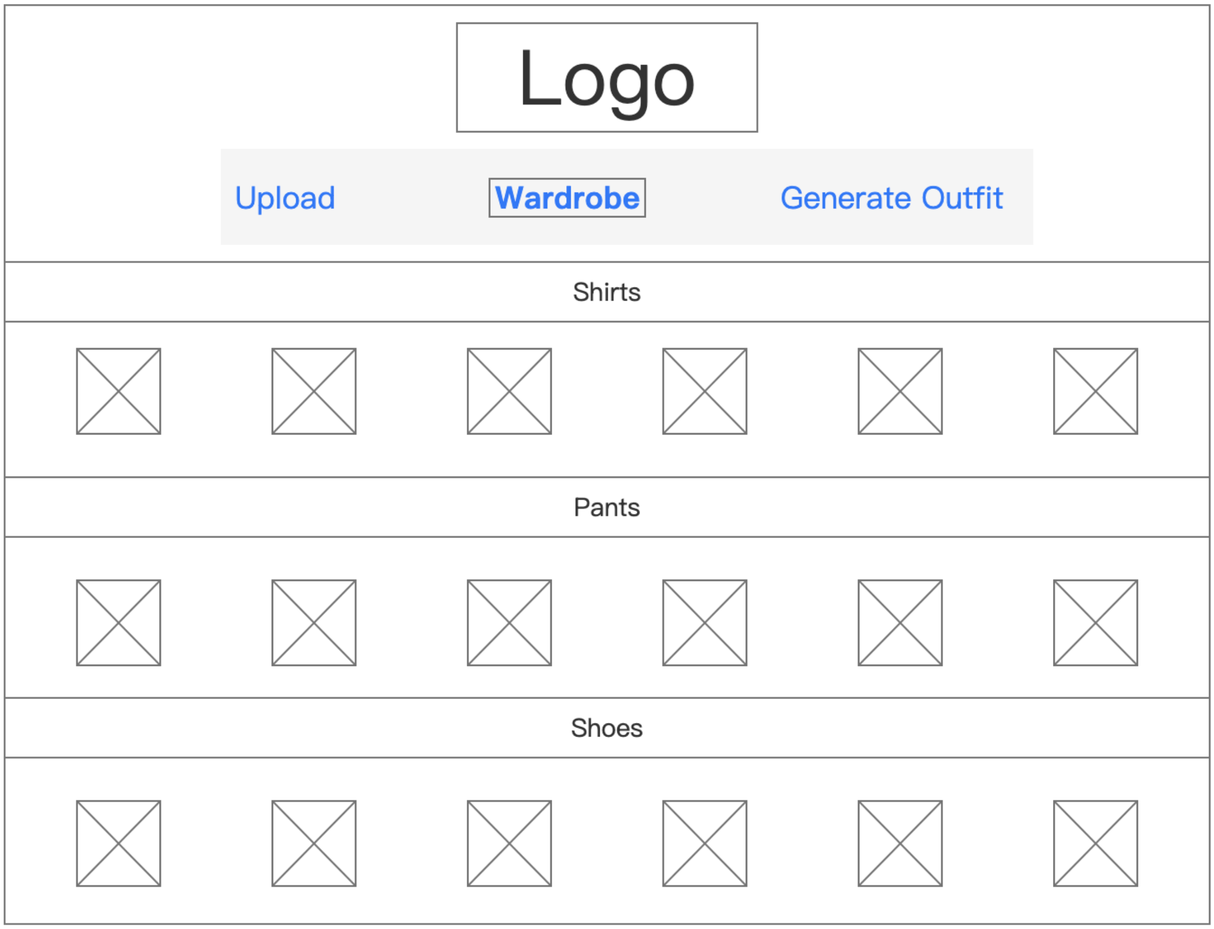
   1. When the user first opens the app, the main screen is the “Generate Outfit” screen:



   2. Upload screen:



3. Wardrobe screen:


Logo
Upload
Wardrobe
Generate Outfit
Shirts
Pants
Shoes