Phase III: Software Design and Modeling
Deadline: April 1st, 2024, 23:59

Software Design and Modeling
Group Name: Studental

Amanda Peza BINF III-B

System Architecture:
Explain how different parts of the system work together. Think of it as describing the big picture of your application - what it does and how it does it.

Studental, is designed as a web-based platform that connects Albanian students with various opportunities such as internships, scholarships, employment, training, and more. The architecture follows a client-server model where the client-side comprises the user interface and interaction components, while the server-side handles data processing, storage, and external integrations.

Client-Side (Frontend -presentation layer):

- FlutterFlow is used to develop the client-side of the application, including the user interface (UI), user interaction components, and frontend logic.
- The Flutter app runs on the users' devices (e.g. web browsers) and communicates with the server-side components through APIs (Application Programming Interfaces).

Server-Side (Backend-application layer):

- Firebase serves as the backend infrastructure for the web application, providing cloud-based services for data storage, authentication, and real-time communication.
- Firebase Firestore is used as the database to store and manage data related to students, internship opportunities, user authentication, and more.
- Firebase Authentication handles user authentication, including sign-up, sign-in, password management, and user session management.

API Communication:

- The communication between the client-side (Flutter app) and the server-side (Firebase) is facilitated through APIs.
- Firebase provides RESTful APIs and SDKs (Software Development Kits) for Flutter that allow the Flutter app to interact with Firebase services securely.
- For example, the Flutter app makes API calls to Firebase Firestore API to retrieve and update data in the Firestore database, and to Firebase Authentication API for user authentication and authorization.

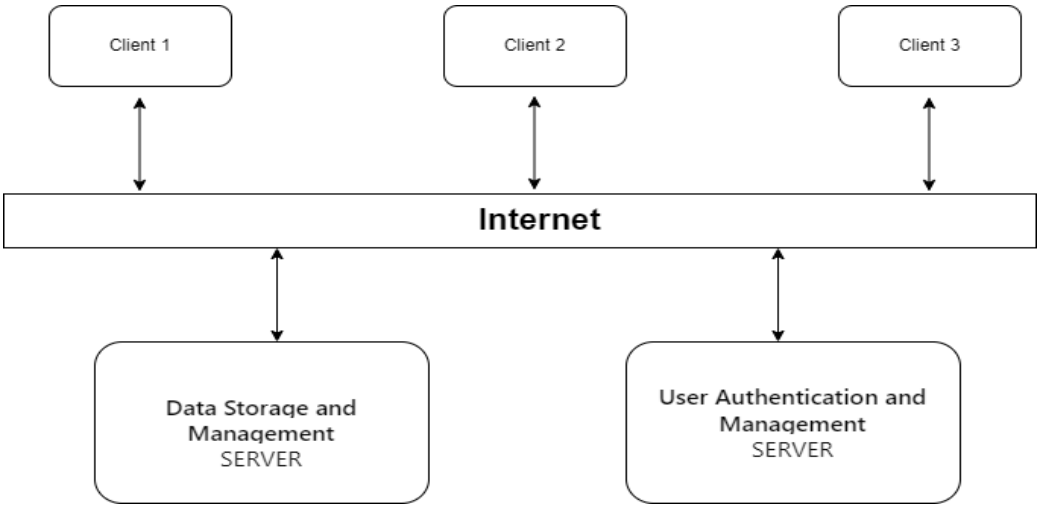
Cloud-Based Services:

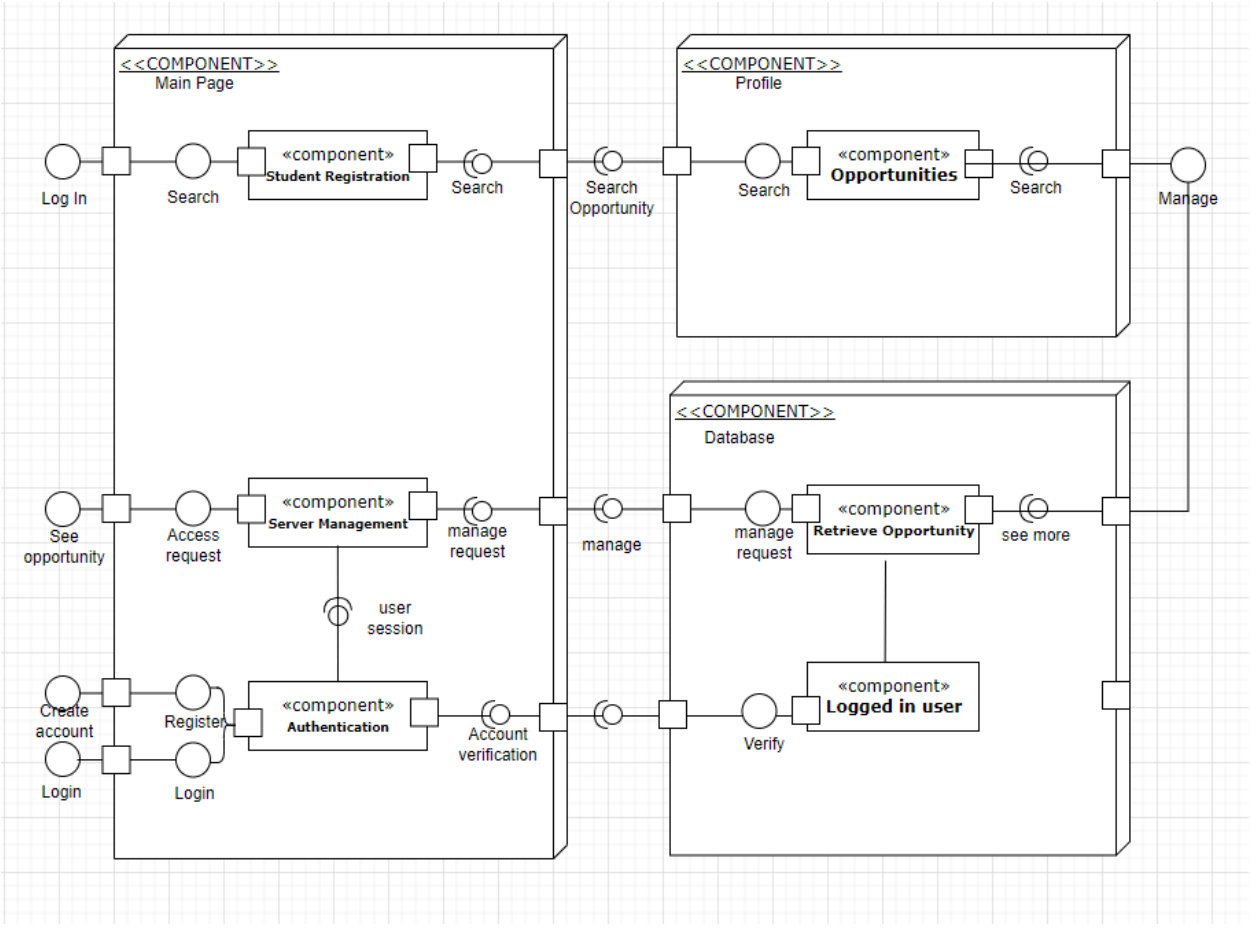
- Since Firebase is a cloud-based platform, it offers scalability, reliability, and real-time synchronization of data across devices.
- Cloud-based services ensure that the application can handle a large number of users, scale resources as needed, and provide a seamless experience regardless of the device or platform used by the users.

Component Diagram:

Draw a picture showing the different parts (components) of your application and how they interact with each other. For example, if your application has a login feature, a component diagram would show how the login component talks to other parts of the system.

Description:In a client–server architecture, the functionality of the system is organized into services, with each service delivered from a separate server. Clients are users of these services and access servers to make use of them. It is used in the Studental app because data in a shared database has to be accessed from a range of locations. The main advantage of this model is that servers can be distributed across a network.

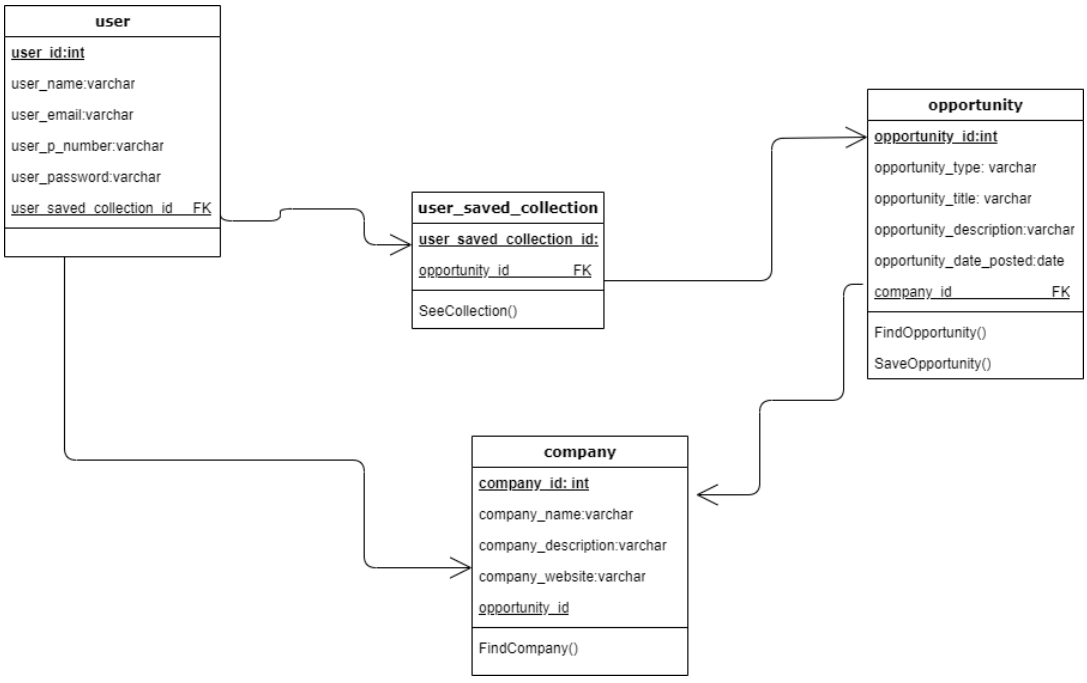




Detailed Design

Class Diagram:

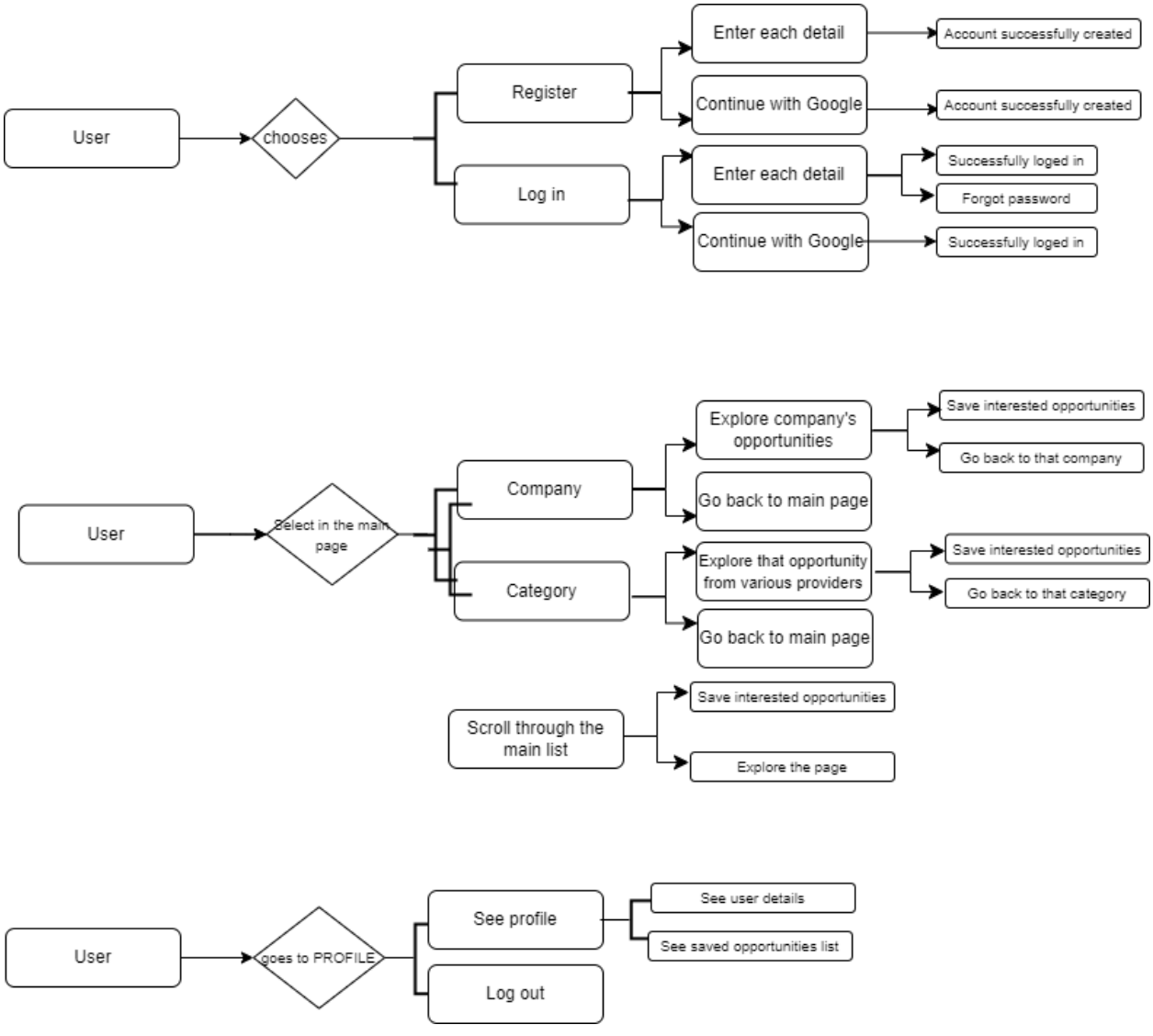
Think of a class diagram as a family tree for your application. It shows the different types of "things" in your application (called classes) and how they relate to each other. For example, if your application deals with cars, a class diagram would show that a Car class might have attributes like color and model, and methods like drive() and park().



Sequence Diagrams:
Sequence diagrams show the order in which things happen in your application. They're like step-by-step instructions for how different parts of your application interact with each other to accomplish a task. For example, a sequence diagram for ordering food online would show the steps involved, like selecting items, adding them to the cart, and checking out.

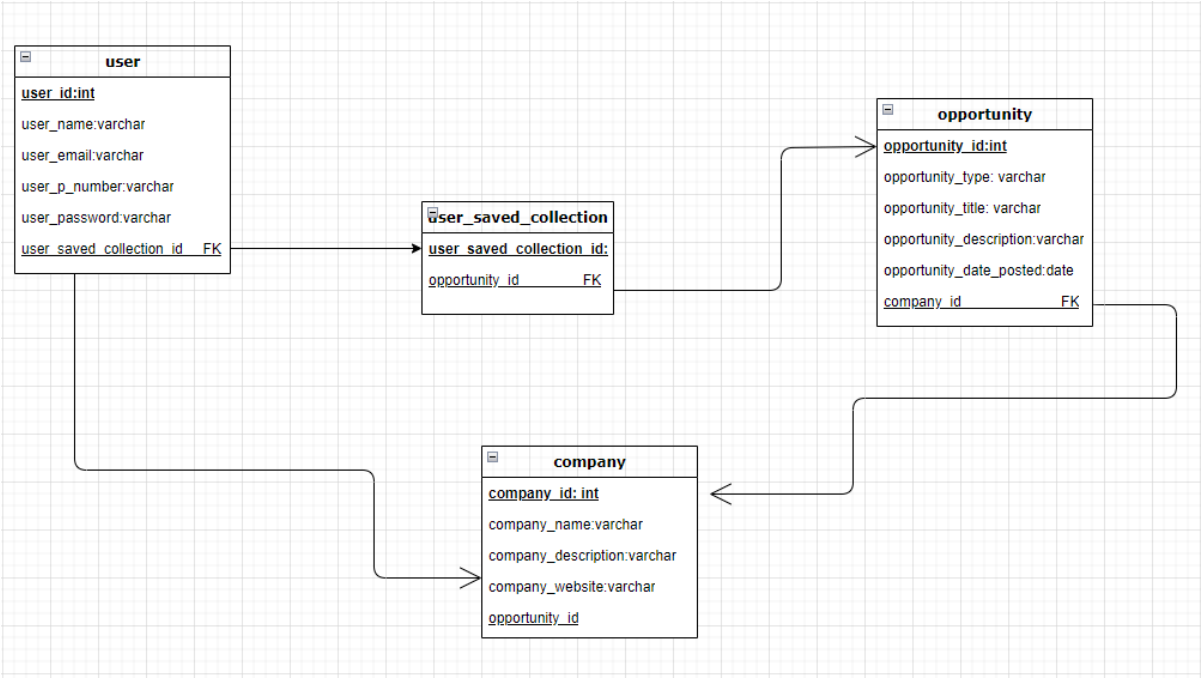
Use Case Diagram:
A use case diagram shows the different ways people (or other systems) can use your application. It's like a map of all the different things your application can do. For example, a use case diagram for a music streaming app might show that users can search for songs, create playlists, and listen to music.

Activity Diagrams:
Activity diagrams show the flow of activities in your application. They're like flowcharts that show the steps involved in completing a task. For example, an activity diagram for booking a flight might show the steps involved, like searching for flights, selecting one, and entering passenger information.



Database Design:

Explain how you've organized the data in your application. This includes things like what tables you have in your database, how they're related to each other, and how you've made sure your data is organized efficiently.



Modeling

State Diagrams:

State diagrams show the different states that an object in your application can be in, and how it transitions between those states. They're like maps of all the possible "statuses" your application can be in. For example, a state diagram for a light switch might show that the switch can be in the "on" or "off" state, and how it transitions between them.

-