EECS 581: Project 3

Initial Architecture Document

Version 1.2

Table of Contents

Team and Project Overview

Team Number: 30

Team Members: Sophia Jacob, Anna Lin, Kusuma Murthy, Nimra Syed, Nikka Vuong

Project Name: Elysian

Project Synopsis:

With the mobile app, users can create a curated traveling profile, allowing Soleil AI to generate personalized trip recommendations that the user can scroll through.

Architecture Description

Overview:

The mobile app comprises five components: frontend, backend, database, authentication, and the AI recommendation engine. The frontend will act as the interactive Graphical User Interface (GUI) for the users to create their profile, log in, scroll through their generated recommendations, and more. The backend will handle all the business logic and communicate with the AI engine, allowing for the seamless integration between the two components. The database houses the different user profiles with their corresponding user data and preferences. This will be used to query the relevant information needed for the different endpoints in the app. The authentication component ensures best security practices, with the option for the user to validate their credentials via Google Account. Finally, the AI recommendation engine brings the system's core intelligence that creates real-time traveling recommendations that are personalized to the user.

Frontend:

The frontend is built using React Native for an iOS mobile app. With the integration of React Navigation, users can seamlessly move from various pages, such as the Login Page, Curated Recommendation Page, and Account Profile Page. It interacts with the Authentication component to allow users to create and log in or out of their accounts securely. When creating an account, users will be required to answer questions to build their personalized travel profile. These data will be stored by the Database component and later queried by the Backend and AI Recommendation Engine component to generate tailored travel destinations, which are displayed on the Curated Recommendation Page. The Curated Recommendation Page is a scrollable interface showcasing the personalized travel recommendations, each featuring an image and a brief description along with the destination, to help users explore potential trips. To provide a modern, user-friendly experience to the user, the app will utilize TailwindCSS styles and utilities for consistent spacing, color, typography, responsive layouts, and smooth transitions and animations when navigating pages and scrolling through the recommendation feed.

Backend:

The backend for the mobile app is built using Flask and is responsible for integrating the Database and AI Recommendation Engine component. It will use the AI Recommendation Engine component to generate recommendations based on the user's input received from the frontend. Based on the information retrieved from the frontend, the backend will use a Python script to query the right user account logging into the session and return the respective account/profile information back to the frontend. The backend is also responsible for passing the information received from the frontend to the Database and AI Recommendation Engine components so they can perform their respective tasks, such as adding a new entry in the database or generating new recommendations. The backend manages the maintenance of the system's activity, which is not visible to the user on the frontend. Therefore, the backend will include RESTful API endpoints, built with Flask, to help with this smooth communication between the frontend and the other components of the app.

Database:

The database is built using Firebase Realtime Storage. The first database table stores all user-related data, including user profiles, travel preferences, and their recommendation history. This table will be integrated with the Authentication component, using a unique user ID (UID) for secure storing and retrieving of user information. The second database table serves as a centralized repository for all possible locations. This dataset includes attributes relevant to the AI Recommendation Engine component, including location names, countries, and cost estimates. This table will be accessed by both the Backend component and the AI Recommendation Engine component to generate accurate and personalized travel locations.

Authentication:

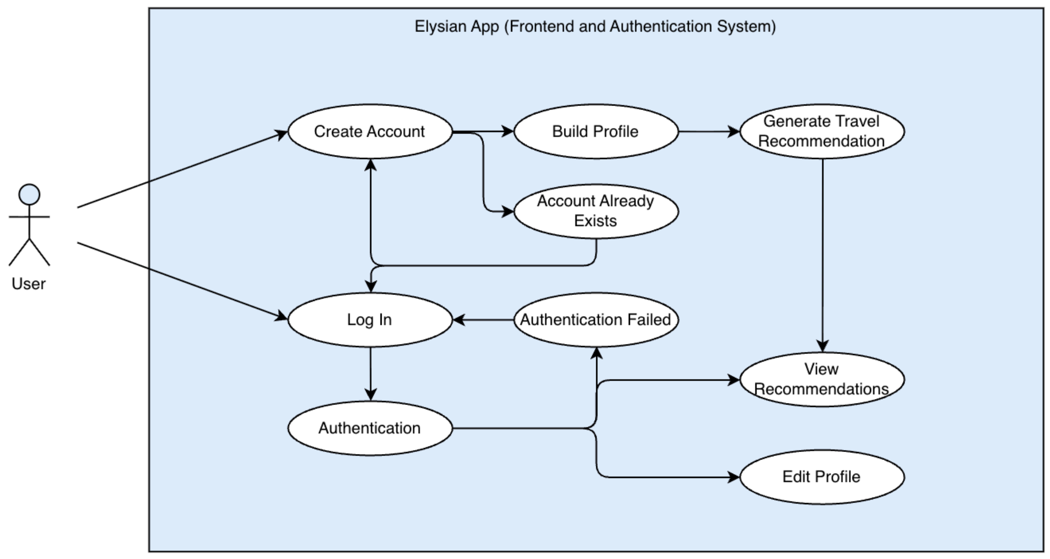
The authentication system for the travel app is built using Firebase Authentication, which provides a secure and user-centric/user-friendly way to manage sign-up and login features. Since the backend is built with Flask, integrating Firebase allows us to offload most of the complex authentication logic(like password hashing, session handling, etc) to a trusted external service. The flow of the authentication is as follows: when a user opens the mobile app for the first time, they can either create an account or log in using their existing credentials and/or Google account. Once the user signs in, Firebase handles the OAuth process in the background and generates a UID (all the UIDs will be stored in Firebase storage as mentioned above) for that person. This UID acts as the main identifier for the user throughout the system like linking their preferences, quiz responses, and AI-generated recommendations together. After a successful login, Firebase returns a secure ID token, which the mobile app stores locally and sends with each request to the Flask backend. The backend uses Firebase’s Admin SDK to verify the token before performing any actions, which ensures that all data access is authenticated and safe.

AI Recommendation Engine:

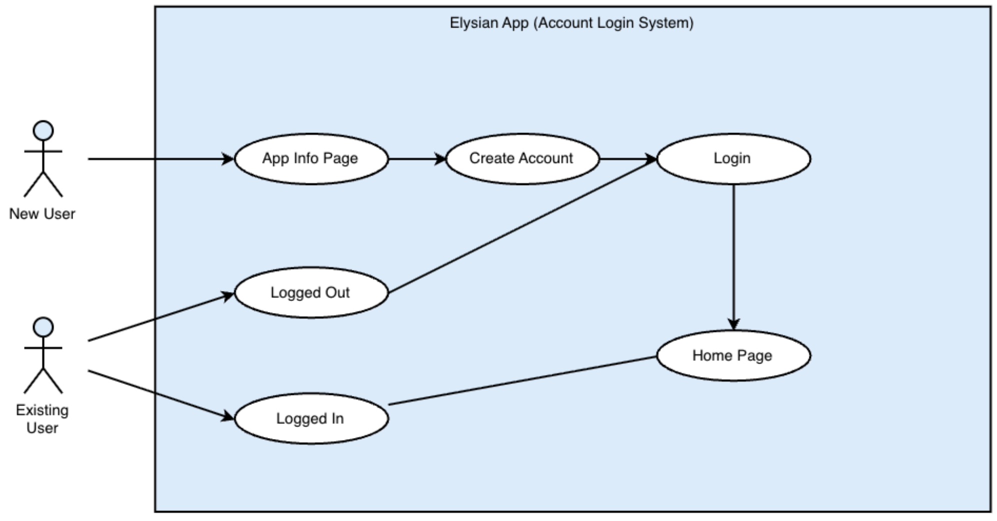
With this mobile app, the team aims to create a tailored AI recommendation engine that will utilize user preferences as inputs to the model and convert them to feature vectors to optimize the different user data and their weights in the model. All the inputs for this model will be stored in the Database component of the system and will be queried when needed in the AI Recommendation Engine. This will allow the model to learn from these inputs to output city locations that fit the most requirements. The engine will be trained with previous datasets of user data and will use best practices when creating the model. To facilitate this process, the mobile app will use TensorFlow to create the custom recommendation engine. Specifically, the app will use the TensorFlow Recommenders (TFRS) library to create a custom model tailored to traveling destinations. It will also utilize TensorFlow Lite, which is Google's framework for running machine learning models on mobile devices, to facilitate the deployment of the engine. Finally, these recommendations will be stored in the Database component in a separate table and will be displayed by the Frontend component once the user fills their profile information.

UML Diagrams

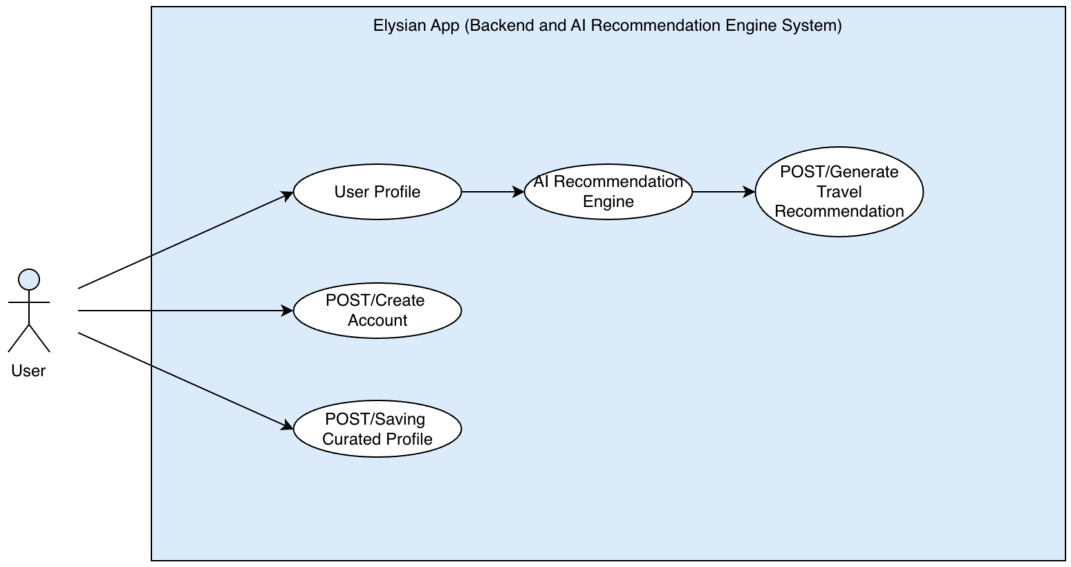
Frontend and Authentication System:



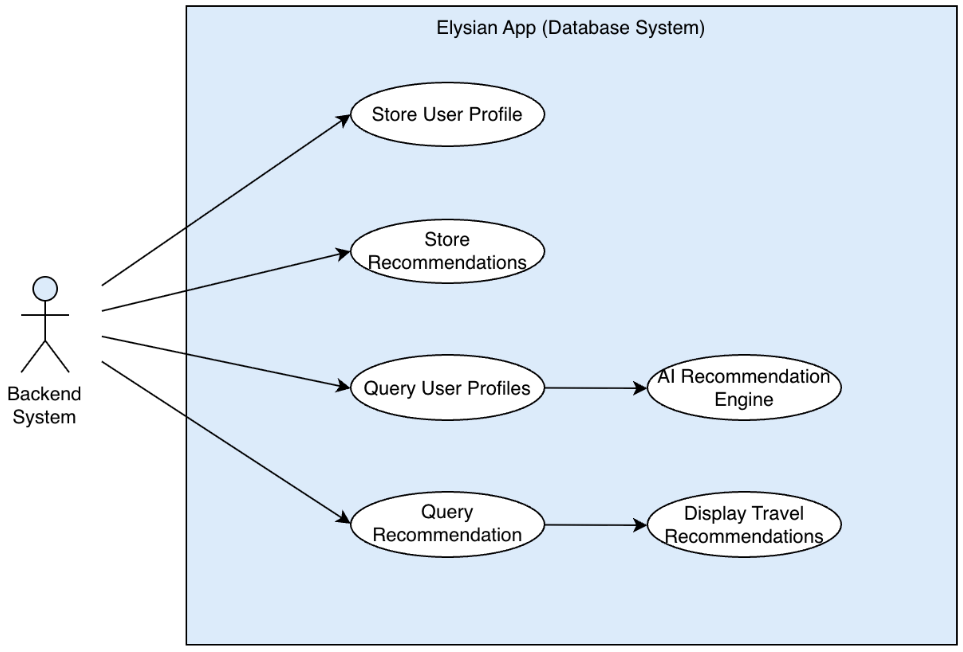
Account Login System:



Backend and AI Recommendation Engine:


Elysian App (Backend and AI Recommendation Engine System)
User Profile
AI Recommendation Engine
POST/Generate Travel Recommendation
POST/Create Account
POST/Saving Curated Profile
User

Database:



**** Note: All our Project Meeting Logs will be housed in the GitHub Repository on the Wiki Page. Please reference it as needed. We also have our Sprint Meetings and tickets created on the GitHub page.**