Project Epsilon

System Design Document

Team member utorid’s:
venturo4, fungcore, siniat, gandhihr, hameed10, liweiyu, louiskob

Table of Contents

CRC Cards

App

Class Name: app	
Parent Class: N/A **Subclasses**: N/A	
Responsibilities: Starts the application, ties all the classes together, renders the HTML.	**Collaborators**: Every class.

Business

Class Name: business	
Parent Class: App **Subclasses**: N/A	
Responsibilities: Handles accepting and rejecting team join requests and showing them in the frontend.	**Collaborators**: Populate Database.

Get Team

Class Name: getTeam	
Parent Class: App **Subclasses**: N/A	
Responsibilities: Displays a team from team ID.	**Collaborators**: N/A.

Populate Database

Class Name: PopulateDatabase	
Parent Class: App **Subclasses**: N/A	
Responsibilities: Initializes database, adds data to db, retrieves data from db.	**Collaborators**: N/A.

Registration

Class Name: Registration

Parent Class: App **Subclasses**: N/A	
Responsibilities: Check if a team exists before creating it, it creates a new team with user credentials.	**Collaborators**: Remove Team, Populate Database.

Remove Team

Class Name: Remove Team	
Parent Class: App **Subclasses**: N/A	
Responsibilities: Remove a user from the team, promote a user in the team.	**Collaborators**: Populate Database.

Software Architecture

Description

Project Epsilon is a Startup Marketplace application in current development. The architecture of this project is a **Three-Tier Architecture**. This means the application will have a presentation tier, a logic or application tier, and a data tier.[1] All of these are running in their own infrastructure. By choosing this architecture, the team is allowed to run the development of these tiers independently and avoid conflict.

Presentation Tier

For the front end of the application, we will be using the **React** framework. React is a javascript library that helps build the graphic user interface of web applications. It is lightweight and scalable[2] and even though it does not have excellent documentation, the community feeds online offer sufficient support.

Application Tier

The application tier houses the project logic and API. We will use the **Flask** framework in Python code. Flask offers the flexibility we need as it is compatible with a lot of other frameworks such as Docker. Allowing for design changes in the future makes it easier to scale this project according to any new requirements that might come in. Flask is also independent from the front end framework and the database management systems, allowing us to choose the most convenient for us.

Data Tier

The chosen database management system is **MySQL**. We are connecting the application and data tiers through the flask-mysqldb[3] library. We chose a relational DMS because the team has had more exposure to this type of database and it appears the most fit for our database schema as we want the data to be more structured.

---

[1] Education, I. C. (2021, April 5). *Three-Tier Architecture*. IBM. https://www.ibm.com/cloud/learn/three-tier-architecture

[2] Lvova, E. (2021, April 5). *React vs. Vue in 2021: Best JavaScript Framework*. Dzone. https://dzone.com/articles/react-vs-vue-in-2021-best-javascript-framework

[3] Flask-MySQLdb. (n.d.). *Welcome to Flask-MySQLdb's documentation! — Flask-MySQLdb 0.2.0 documentation*. Retrieved June 10, 2021, from https://flask-mysqldb.readthedocs.io/en/latest/

Diagram

The following is the System Architecture Diagram of Project Epsilon. It is a three tier architecture as cited above (https://www.ibm.com/cloud/learn/three-tier-architecture).