Requirements Specification for Study Buddy

1. Introduction

1.1 Purpose of Product

Study Buddy provides a space for university students to organize study groups. After all, “Professor Q is absolutely killing me with these gravitational acceleration questions, and I have no one to work through this with.” To clarify the problem for our imaginary student is not that there are not resources at the university to help solve this problem. She could contact the TA, pay for a tutor, and the university has a big building with plenty of space to work the problem. What this student lacks is what universities have traditionally provided that are now seemingly absent: a community for degree seeking students who are going through similar problems in similar classes. University message boards spontaneous meetups all used to dominate the college sphere. A similar student fifteen years ago might have said “ It's Thursday and the Physics 214 people always meet in the sub to do work through Professor Q’s problems maybe I can check that out.” Various search engines contain a great deal of knowledge, however community and context is ultimately how the human machine is programmed.

1.2 Scope of Product

The scope of this project is that our product will match the view of group based on the student’s selected campus. At this time, the scope of this project will not match based on location, just the student’s declared campus. The student will be able to additionally match with groups within their campus based on classes they are in. Groups can be created and joined by users for specific classes and users will have the ability to join existing groups.

1.3 Acronyms, Abbreviations, Definitions

SB - Study Buddy
UI - User Interface
FE - Front end {The Android App / what the customer sees}
DB - Database {SQAlchemy}

1.4 References

Trello - This is the scrum like board we use to coordinate tasks among team members.
https://trello.com/

2. General Description of Product

Study Buddy is an Android app and consists of a simple two phase product implementation, frontend, and backend.

First, is the Android front end. The code is being developed with android studio to take advantage of the preexisting frameworks and functionality for Android app development. The front end of the application is what the user sees and interacts with. In turn, the user is expected to provide specific information that will make the app more functional. Example of information includes basic info about the individual (name, school) and more functional information (such as classes taken that semester).

Second is the back end is a database which is an SQLAlchemy database that interacts with the front end. The back ends consists of pseudo normalized database schema that allows the application to create views of study groups. Moreover the backend is expected to provide validation and provide security to the information within the app by setting specific rules and context for which that information will be accessed and displayed. Database has numerous tables enumerated in the domain model which provide the apps structure. This is what allows the app to have “memory” so that groups can be created from existing members.

2.1 Context of Product

Study Buddy will be used on college campuses to aid students in facilitating study groups.

2.2 Domain Model with Description

Domain model remains somewhat simplistic and flexible as implementation involves.

1. Database dependency shown is only between User_Login and Database, but in reality every one of these tables is dependent on the database to be able to retrieve and store information.
2. User_Profile remains the central Class for the application:
    1. User should be able to login: and fill out required information with a 1:1 relationship with the User_Login
    2. User_Profile will receive notifications from User_notification. Many notification as * to 1 user profile.
    3. User_Ratings is a composition fo User_Profile as it extends User_profile by adding in the actual ratings. This is necessary because User_ratings will likely be a conglomeration of ratings, which may require more

functions. Also this allows user ratings to be extensible for visualizations later in development.

4. Users will be able to add courses in the userprofile tab. But this information is inherited in to the course table. The implementation here is still under consideration: Relationship is 1 profile many courses.
    1. Do users select from a drop down on courses?
    2. Can they add a new course if they cannot find one?
5. User_Profile is used to aggregate a group. In fact multiple profiles are. Many to many relationship.

2.3 Product Functions (general)

Our product will be able to log users into individual user accounts which will display data based on other users who are signed up on the same campus. Users will be able to create and join groups. Users will be able to accept and reject other users into their groups. Users will be able to rate other users.

2.4 User Characteristics and Expectations

Study Buddy's users are college students who are looking to host and/or take part in study groups. They should be able to utilize basic application functions similar to those in other apps commonly used by this age group. They should also have a school email address that will be used for registration and logging in.

2.5 Constraints

We are constrained to languages and DBs that can be used by Android studio. Because of this constraint, we have chosen to construct our app using Java and SQL since these are languages we are all familiar with. Additionally, we are unable to run Android emulators for the newest Android phones on our system, so we must use an older version of Android in order to run the app on our systems.

2.6 Assumptions and Dependencies

Our system will definitely depend on Java libraries as well as our SQLAlchemy database. At this time, it seems like everything else we will need can be found within Android Studio

3. Functional Requirements

Functional Requirements:

- We want to have SQLAlchemy able to accept any inputs from the front end.
- Generate app splash and login screens.
- DB Table Users and front end & back end provides validation.
- App provides user feedback to show errors or success.
- Groups can be created by user.
- Validation will be done on front/back end and then inserted into DB.

4. System and Non-functional Requirements

4.1 External Interface Requirements (User,Hardware,Software,Communications)

NF.4.1.1 Our product can be viewed on devices and emulators able to run Nougat 7.0. Our emulator is a Nexus 4 running Nougat 7.0.

4.2 Performance Requirements

NF.4.2.1 Our app will need to load pages at a reasonable speed (Less than 3 seconds)

NF.4.2.2 Our app will need to run queries at a reasonable speed (Less than 10 seconds)

4.3 Design Constraints

NF.4.3.1 The UI of our app will be limited to the google design native to android for the sake of simplicity

4.4 Quality Requirements

NF.4.4.1 Users will expect to be given information at a reasonable speed (within 3 seconds or less of clicking a new tab)

NF.4.4.2 Users will expect intuitive UI, which is why we are using the proven model from android

5. Appendices

5.1. Android Studio

Please refer to the following link for more information about Andriod Studio:
https://developer.android.com/studio/intro/index.html

5.2. SQLAlchemy

Please refer to the following link for more information about SQLAlchemy:
https://www.sqlalchemy.org/