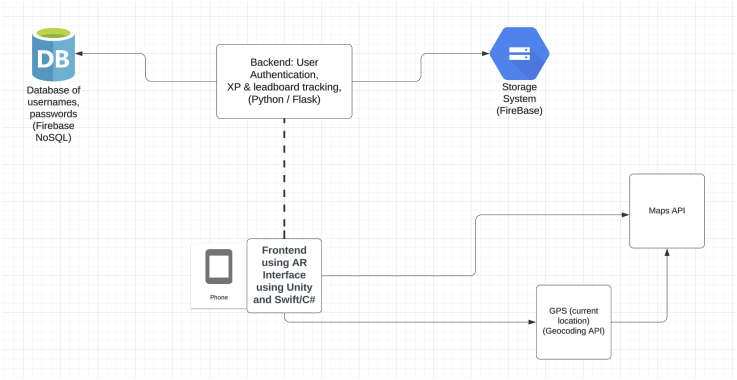
Nick Jackoski
Anas Matar
Kamil Yousuf

Planning Document

1. System Description
    a. Provide an augmented reality (AR) interface that introduces prospective students to the many facets of Rhodes College.
    b. Will provide a basic AR interface (first interface)
    c. Storage for images/assets
    d. Dynamically loaded assets – should not load everything at once
        i. Use GPS to know when assets are loaded
    e. Database inclusions:
        i. Locations
            1. GPS coordinates
            2. Items at location (foreign key)
        ii. Users
            1. Specific id
            2. Username
            3. Password
            4. XP
            5. Discovered items (foreign key)
            6. Discovered locations
        iii. Items
            1. Name
            2. Id
            3. Purpose/association?
2. Architectural Overview
    a. Our architecture comprises four major parts: the frontend, backend, database, and maps API. The frontend is built via Unity's AR IOS kit, it will interact with the backend and the maps API using REST API. The backend consists of a Flask server programmed with Python; it will facilitate an interaction with the front end and the database. The database will hold the user's data, progress, and asset information. The database will also provide links to the virtual items in the storage. Finally, the maps API will provide the minimap functionality and coordinate your location to the quest.
    b.
        1. Firebase (Database)
        2. Firebase (Storage)
        3. Python/Flask (Server/Backend)
        4. Unity IOS AR frontend

Nick Jackoski
Anas Matar
Kamil Yousuf

5. Maps API

c. 

3. Functional Requirements
    a. Information related to specific facets of the college is displayed when those particular buildings are scanned.
        i. A pop-up is displayed describing the building's purpose
    b. Scanning a building will provide the user with experience points (XP)
    c. If a user chooses to log in, they will have access to leaderboards consisting of all who have used the app previously, allowing them to compare XP
    d. A quest system where users can move through the college on a linear path
        i. If they find items outside the desired linear path, users gain additional XP

EPIC 1 - Placing Items
As a LynxUp user **I want** to be able to place items around campus **So that** I can interact with the environment around me

EPIC 2 - Locating items

Nick Jackoski
Anas Matar
Kamil Yousuf

As a LynxUp user **I want** to be able to locate buildings/items throughout campus with facts about the items **So that** I can learn my way around campus

EPIC 3 - Personalized Experience
As a LynxUp user **I want** a personalized experience that is tailored for me **So that** I get the maximum amount of joy out of the game.

EPIC 4 - Quests
As a LynxUp user **I want** to have the option of a fun guided tour of the campus **So that** exploring the campus is enjoyable

4. Non-Functional Requirements
    a. The AR displays should load in no longer than 1 second on the app
    b. The app should function in an offline mode with pre-downloaded campus maps
    c. The app should require no more than 4 steps to set up the AR mode by identifying buildings.
    d. The app must not store the user's location data for security reasons.
5. Technologies and Frameworks
    a. Swift / Xcode for app making, Unity for AR, Firebase SQL for database
    b. GPS to determine relative positioning on campus
6. Minimum Viable Product
    a. The app should have two modes: an AR mode/quest mode, the user can move around the map or campus to discover AR facts about buildings that pop up when the user walks around campus with an added story. A second mode should be integrated with the first with an XP system that involves discovering buildings and geocaches with a leaderboard between peers.
7. Roadmap:
    a. Sprint 1 - app does not crash, AR display
    b. Sprint 2 - AR can identify buildings; tapping a building gives a name and description / provides information

Nick Jackoski
Anas Matar
Kamil Yousuf

c. Sprint 3 - personalization system / FireBase / Database (or maybe single-user [so storing information locally])
d. Sprint 4 - connecting the backend to the frontend; setting up an API through which the frontend can access assets from the FireBase storage