Technical Specifications Document

Application: Arcade Hoops

Nirvaan Reddy, Sam Donovan, Christopher Cooper, Teresa Tran, Kameron Shahabi, Natalie Shamilian

- Make github repo
- Design front end
    - User+Guest login
    - Transfer data to and from backend through Tomcat servlet requests
- Execute front end : mac OS - Swift
- Servers: front end send info to the server and backend pulls it
- Backend controls network endpoints - similar to REST API - Java
- Backend: Use Java & SQL
- Multithreaded live leaderboard - constantly pulling info from server

Frontend To Do List: (~15-20 hours?)
1) Model user interface after
https://www.figma.com/file/S66XbPX6nsvKrbmUOlsqfX/Arcade-Hoops?node-id=0%3A1
2) Functions to fill in to send/receive data to/from backend
    a) createUser(username: String, email: String, password: String),
    b) attemptLogin(username: String, password: String) -> Bool
    c) addFriend(myUsername: String, friendUsername: String)
    d) deleteFriend(myUsername: String, friendUsername: String)
    e) downloadLeaderboard() -> Array<LeaderBoardEntry>
    f) submitScore(myUsername: String, newScore: Int)
    g+) ??

```
class LeaderBoardEntry {
    scorerUsername: String
    score: Int
}
```

3) Setup RealityKit
    a) Graphically model hoop, ball, and player and make corresponding objects
    b) Keep track of score/how long player has been playing
    c) When time runs out, return to main menu and submit score to server
4) Simple user interface that
    a) renders a button over the RealityKit view
    b) shoots ball in RealityKit physics simulation when is pressed

Back End: Databases (10 hours)
Set up mySQL databases for the usernames, high scores, and passwords
    a) There will be a primary key for all the users with a user_id and a password

b) There will also be a table with a primary key for user id corresponding to the actual usernames
c) Finally there will be a table for each user id corresponding to high scores
d) Users:
    i) User_id
    ii) Username
    iii) Password
e) Scores:
    i) User_id
    ii) Highest_score

High level functions calling upon database

1. getAllHighestScores()
2. addUser(string username, string password, Integer highestScore)
3. updateHighestScore(User) (for individual user)
4. updateHighestScores() (for all users, sends to frontend)
5. deleteUser(User)