CSC309project

Topic:

Creating a web application that allows gamers to offer and receive mentorship for the popular online games League of Legends, Dota2, Overwatch, CS:GO.

Description of web application:

(BeMaster) is a web application that facilitates the interaction between those who wish to improve his or her skillset at a particular game (student/mentee) and those who are willing to offer his or her area of expertise in the same game (coach/mentor). A coach can choose to coach many different games and can decide to offer online mentoring (i.e. watching the user play and informing him/her of his/her mistakes over Skype or Teamviewer), or they can offer offline mentoring: meeting up with mentees face-to-face at an internet cafe to play together in real life. (InsertNameHere) also provides the added security of payments being dealt (for online for offline meet-ups), rather than players needing to pay at the physical location.

Why this web application is important:

In current day, online gaming has evolved to something beyond a mere hobby. The introduction of competitive online gaming has not only paved the road for gamers to be able to earn a living by doing what they love: playing online games, but also bolstered motivation for casual players to improve their skills. On top of this, online gaming communities worldwide have increased in numbers, with many new players demanding coaching services; mentors can meet this increasing demand through coaching via (InsertNameHere).

Skilled players are attracted to the prospect of becoming a mentor as it not only allows them to earn money, but also provides them the opportunity to positively shape and grow the online gaming community that they are a member of. New players will be attracted to coaching as they are presented with the opportunity to learn the game faster and to avoid the unforgiving learning curve that many modern day online games are notorious for having. Existing, casual/less-skilled players who are tired of being stuck at the same rank/elo will want a better player observing their gameplay to catch on mistakes so they can improve.

How it works:

After registering on (InsertNameHere), the user has the option of applying as coach. As a coach, his/her user profile will be displayed to all users on the website as someone who is proficient at a specificly listed game(s) and is offering his services at a price of his/her choosing (by dollars per hour), or at no cost if the coach decides to offer training out of the goodness of his/her heart. As a mentee/student, one can select the game he or she desires training for, select a suitable coach, pay (if there is a price that the coach wants), and send a message to the coach to decide on a time and date to hold the training. After the training, the mentee has the option of leaving a rating on the mentor's profile. Likewise, the mentor can leave a rating on the mentee's profile.

Features & Implementation Details (plan):
- High level overview of the architecture:

We plan to use the MVC architectural pattern for our web application because it is relatively effective and simple in understanding how data is stored, obtained, changed, and transported within the context of our web application. It provides a greater level of organization as it can classify different components of the web application.

- Model:
  The model consists of the database that stores user information. The model is updated when the user inputs information that needs to be stored in the database.
- View:
  The view acts as the visual representation of the model/database. That is, the data is displayed to the user through the front-end. It comprises of javascript, html, and css files.
- Controller:
  The controller connects the model with the view together; a large part of the controller is the node.js server. Another component is the javascript that takes the input entered by the user and inputs it to the database. It also takes information from the database and passes it to the View components (html) to be displayed for the user.

- Database:

We plan to use (InsertDatabaseName) as our database. It will store user authorization details (username and password) as well as profile information of the user (Name, Games, coach or not, profile picture, ratings and reviews).

- Sign up:

The user will register an account through form fields. The information will be entered into a database for authenticating the user (upon future attempts to sign in).

- Login and User Authentication:

After the user signs up, his/her username and password are stored in a database. When he/she attempts to login, the inputted authentication details are checked with that in the database. We also have a further authentication feature via logging in with Facebook and Google Accounts.

- User Profile (Coach/Mentor and Student/Mentee):
  When the user profile is loaded, we retrieve the user data from the database. The data is added to the html page.
- Coach and Student:
  Both have the ability to change their password, profile information, etc. These changes will be saved in the database. The profiles also have reviews that evaluates the effectiveness of a coach or how well a student behaved.
- Coach:
  Have the added feature of a badge that identifies them as a coach. They also have time listings for when they are available for coaching.

- Rating/Review System:
  Mentees, after going through the tutoring session, can give a review/rating (i.e. rating out of 5 and a few sentences) to the mentor. This will be displayed on the mentor's profile. The mentor has the option of replying to any comments/reviews from the mentee. The reviews are stored in the database and are loaded upon profile loading.

- Messaging/Social Network:
  If a potential mentee has any questions for another user, he/she can send a direct/instant message to the user. This will be implemented using Node.js. We also want to implement a broadcasting/bulletin board that allows coaches to update their status. There is also the feature of being able to "friend" a coach. Friends can post comments on each other's profile.

- Searching for coaches on the website:
  After a potential student chooses a game of his or her choosing, all the coaches for the particular game will be listed (loaded from the database). If the student has a particular coach that he/she wants to search for, we can use the database to query for the coach name and display the coach in question.

- Recommending coaches:
  The front page of the website recommends coaches for each game. Generally, the coach with the best reviews or highest ratings. There are other variables that come into effect too (i.e. language spoken, times available for coaching, etc). The database can be queried with a filter to find suitable coaches, based on the user that is currently signed in.

- Searching for nearby coaches available for in-person coaching/mentoring:
  The Google Maps API provides a map. We can use this map to display coaches/mentors that are available for offline coaching. Coaches can toggle their coaching availablity. This will place a marker on the map. The idea is that potential students interested in offline coaching can find coaches in their near vicinity.

- Admin functionaility:
  Have the ability to change passwords, update database, delete users, ban users, etc. There is a specific administrator account. We utilize a flag to test if the admin has logged in or not.

Task Allocation:
  - Database creation/querying: Albert and WeiJun
  - Recommendation System of coaches: Albert and Weijun
  - Messaging/Social Network System/Bulletin Board: TianLu Zhu and Yixiong Chen
  - Searching for coaches/filtering coaches: Albert and Weijun
  - Register and Login System: Tianlun Zhu and Yixiong Chen
  - Create review System: Tianlun Zhu and Yixiong Chen
  - Admin Functionality: Albert and WeiJun
  - Map of nearby Coaches: Tianlun Zhu and Yixiong Chen

Team Member Contact Information:

Name: Yixiao (Albert) Qin

CDF id: g5qinyix

email: albert.qin@mail.utoronto.ca

Name: Yixiong Chen

CDF id: g4chenyk

email: jackkk.chen@mail.utoronto.ca

Name: Tianlun Zhu

CDF id: c4zhutia

email: tianlun.zhu@mail.utoronto.ca

Name: Wei Jun Zeng

CDF id: g4zengwe

email: weijun.zeng@mail.utoronto.ca