Sprint Planning Document

Sprint 3

GetGuru

Team 12

Sharoon Srivastava

Ankush Jain

Pranav Punjabi

Vijay Srinivas

Murtuza Kainan

Sprint Overview

The objective of this Sprint document is to assign tasks that need to be carried out in the current Sprint. The tasks will be delegated equally among the teammates. A scrum master will also be assigned to facilitate meetings and coordinate with team members. In this sprint we'll implement the chat features. Additionally, we'll allow the student to view ratings and reviews for any tutor(unfinished from previous sprint). We also plan to allow the users to edit their profiles and deactivate their account if they wish. Lastly, we'd also like to make the UI look much more appealing.

Scrum master: Sharoon Srivastava

Meeting times: 3:00 PM, Tuesday & Thursday for 1 hour and 15 minutes & 5:30 PM, Monday for 30 minutes in Lawson B160.

Risks/Challenges: We will have to learn how the chat features work in mobile apps. None of us have a sound knowledge of it. Also since this is the last sprint, we have to check that our apps work properly for the correct input as well as show a proper error message when the user does something unexpected.

Functional Requirements

1. **As a user, I'd like to receive notifications when contacted via chat**

   Acceptance criteria : The user should receive a notification in-app and outside the app telling him that someone tried to contact him. A preview should be visible if seen in the notification bar.

   Frontend:

   Task 1 – Implement a client side background service to notify the user of any messages via push notifications

- Allow the user to be notified of any incoming message by displaying the app icon in the notification bar. Also give a preview of text message received along with sender name in the notification tray.

- Murtuza
- 5 hours

Backend:

Task1 - Send a signal to the tutor who was contacted by the student.

- Indicate the client by sending the id (primary key) of the tutor concerned so that the client is able to display a notification at the profile of the tutor.
- Ankush and Sharoon
- 5 hours

Task 2 - Send a signal to the student who was contacted by the tutor.

- Indicate the client by sending the id (primary key) of the student concerned so that the client is able to display a notification at the profile of the student.
- Ankush and Sharoon
- 5 hours

2. **As a user, I'd like to view my chat history**

Acceptance criteria : The user should be able to view all his chat history with anyone when he opens the chat.

Frontend:

Task 1- Create an interface for the user to send and receive text messages.

- Use android libraries to build a simple incoming and outgoing message section to differentiate between tutor and student text.
- Pranav
- 3 hours

Task 2- Send the text message to the backend associated with a user id.

- Instantaneous sending of a JSON object enclosing text message, the sender user id, the receiver user id and date and time of message.
- Pranav
- 3 hours

Task 3- Retrieve chat from server

- Display all tutors who have been contacted, arranged in order of recent activity. Populate messages if any user is clicked, by requesting server to return messages.
- Murtuza
- 5 hours

Backend:

Task 1 - Save chat history.

- Save the entire chat in the MySQL table at constant time intervals. Create a table in the MySQL database that stores the chat history for each application user. Store the chat messages for each user at regular time intervals.
- Ankush and Sharoon
- 5 hours

Task 2 - Return the chat when requested for, by the client.

- Receive the id of the user from the client and return the chat history of that particular user from the chat history table in the MySQL database.
- Ankush and Sharoon
- 5 hours

3. **As a student, I'd like to instantly connect to the tutors through chat.**

Acceptance criteria : The student should be able to start a chat with any tutor and send/receive the messages successfully.

Frontend:

Task 1 - Connect to the XMPP server and send the user's id to establish a connection

- Connect the Android client to the XMPP server by establishing a network task.
- Murtuza
- 5 hours

Task 2 - Retrieve and display chat messages in a UI fragment.

- Create a new chat fragment and add to the tabhost activity of the client. Add messages received from the server to the fragment as they arrive.
- Pranav
- 3 hours

Backend:

Task1 - Understand how XMPP server works.

- Write small programs to understand the working of the XMPP server.
- Ankush and Sharoon
- 10 hours

Task2 - Set up an XMPP server for the chat feature.

- Write programs to make sure that the client can communicate with the XMPP server and send POST and GET requests. The POST requests would include storing the chat messages in the database and GET requests would include retrieving the chat history from the database.
- Ankush and Sharoon
- 25 hours

4. **As a user, I'd like to edit my profile**

Acceptance criteria: The user, be it a student or tutor, should be able to edit and save his information whenever wanted, from his profile.

Frontend:

Task 1 - Establish an activity for editing information

- Create an activity for profile editing and establish network tasks to send PATCH requests to the API. Create layouts and UI elements for the aforementioned activities.
- Murtuza
- 5 hours

Backend:

Task1 - Understand how PATCH requests work. Write small programs to get familiar with it.

- Understand the semantics of PATCH requests and how to parse and iterate through the arguments passed in a PATCH request.
- Vijay
- 3 hours

Task2 - Allow the client to send PATCH requests to our server so that the user can edit his information.

- Implement functions which allow the user to edit his information. Iterate through the arguments passed in the PATCH request and replace the matching arguments in the table in the database. Test these functions using curl PATCH requests.
- Vijay
- 7 hours

5. **As a user, I'd like to delete my account**

Acceptance criteria : The user should be allowed to delete his account when necessary. The app should not take him through to the next page when the user tries to sign-in with same email and password. When the user deletes his account, everything related to the tutor should be completely erased from the database.

Frontend:

Task 1 - Enable an option for deleting an account in the edit profile activity.

- Allow user to deactivate/delete account on a button click. Will be providing a button to deactivate with the contingency that all data (favorite tutors, chats) would be lost forever if account is deleted
- Pranav
- 5 hours

Task 2 - Create and bind a button to delete a profile.

- Create a network task that sends a delete account request to the API. Send user Id to server and request for deletion of database record associated with that ID.
- Pranav
- 5 hours

Backend:

Task 1 - Delete all information about the user from all the MySQL tables upon receiving the request.

- Receive the id of the user to be deleted from the client and delete the users information from all the relevant tables in the mySQL database.
- Vijay
- 3 hours

Task 2 - Make the id available for future user.

- Since the id of a particular user is a primary key, make sure that this id is made available so that it may be used by another user in the future.
- Vijay
- 2 hours

6. **As a user, I'd like to search for a tutor by name.**

Acceptance Criteria : There should be a UI element which allows the user to type in the name of a tutor and successfully return the results.

Frontend:

Task 1 - Modify the UI fragment for searching.

- For a name to be typed in order to search by name, add an edittext UI element to the search fragment.
- Pranav
- 2 hours

Task 2 - Add search parameter for searching by name

- Modify the Network Task class for searching so that the name parameter is included when send a GET request for search.

- Pranav
- 2 hours

Backend:

Task1 - Implement the functionality in the search function to allow searching by name.

- Receive the name of the tutor from the client. Return the tutor with that name in the tutor table of the database. If no name provided, search through all the names and return the list of tutors based on other criteria such as ratings, location and subjects.
- Vijay
- 10 hours

7. **As a tutor, I'd like to unregister myself as a tutor.**

Acceptance criteria: The tutor should be allowed to unregister himself from being a tutor. When done so, the related information from the tutor table should be removed. Also, he should not be able to view his tutor tab anymore, and will have to register as a tutor again

Frontend:

Task 1 - Add a UI element to the edit profile activity in order to enable the option to unregister as a tutor.

- Provide a toggle button to allow user to deactivate tutor account. All data (student chat history, ratings and reviews) would be lost.
- Murtuza
- 2 hours

Task 2 - Create a Network task to send a request to unregister to the API. -

- Send a request to server to set *ifTutor* field to false for a given User ID.
- Murtuza
- 3 hours

Backend:

Task 1 - Get the id of the relevant tutor and set the field *ifTutor* for this user to 0.

- Get the id of the tutor from the client and set the *ifTutor* value in the student table to zero for that particular user
- Vijay
- 1 hour

Task 2 - Clear all the details from his tutor profile.

- Remove the id, location, subjects taught, average ratings and rating count from the tutor table for the user. Remove the ratings and reviews added for this user from the ratings table in the database.
- Vijay
- 4 hours

8. **As a tutor, I'd like to view my own ratings and reviews.**

Acceptance Criteria : There should be a button on the tutor view of the user's profile which allows the user to view all the ratings for him.

Frontend:

Task 1 - Create an activity for viewing ratings

- An activity and layout with an expandable listview will be created for viewing one's own ratings. This activity will be shared when accesses by the tutor or by students that want to view the tutor's ratings.
- Pranav
- 3 hours

Task 2 - Add a UI element to enable viewing of ratings.

- Modify the existing tutor profile fragment to include a button that binds to the view ratings activity so that a tutor may view their ratings.
- Pranav
- 1 hour

Backend:

The backend functionality has been implemented in the previous sprint.

9. **As a student, I'd like to view the ratings and reviews of various tutors**

Acceptance criteria : There should be a button on the profile of the tutor which allows the student to view all the ratings for that tutor.

Frontend:

Task 1 - Create an activity for viewing ratings.

- An activity and layout with an expandable listview will be created for viewing one's own ratings. This activity will be shared when accesses by the tutor or by students that want to view the tutor's ratings
- Pranav
- 1 hour

Task 2 - Add a UI element to enable viewing of ratings.

- Modify the existing tutor display activity to include a button that binds to the view ratings activity so that a tutor may view their ratings.
- Pranav
- 1 hour

Backend:

The backend functionality has been implemented in the previous sprint.

10. **As a user, I'd like to search for a tutor based on a specific location**

Acceptance criteria : there should be a UI element which allows the user to search for tutors based on a specific zip code

Frontend:

Task 1 - Add UI elements to enable searching by location.

- Modify the search option activity to include and edittext field that allows a user to enter zip codes so that search can be performed based on location. Store the changed location preference in the static application manager class.
- Pranav
- 1 hour

Task 2 - Send location to the API

- Add an additional location parameter to the Network task class for searching that extracts the location preference for search from the Application manager class.
- Murtuza
- 5 hours

Backend:

Task 1 - Check whether the parameters passed in the search function are in coordinate format or zip code format. Return the results accordingly.

- If the parameters passed are in coordinate format, convert the coordinates to zipcodes and return the tutors with the matching zip code from the tutor table.
- Ankush and Sharoon
- 5 hours

11. As a developer, I'd like to use facebook's graph API to validate user accounts (If time allows)

Non-Functional Requirements

1. As a developer, I'd like to make the UI appealing
2. As a developer, I'd like my database to be secure by preventing SQL injections using string validations.
3. As a developer, I'd like the amount of allowable downtime per month to be one hour
4. As a developer, I'd like to have a development API server and a production API server
5. As a developer, I'd like to be able to switch the server (development server or productions server) the android app is sending requests to
6. As a developer, I'd like to test my software completely.
7. As a developer, I'd like to handle unexpected requests.
8. As a developer, I'd like to secure the server side API.

Average workload : (in hours)

Sharoon : 30 hours

Ankush : 30 hours

Vijay : 30 hours

Murtuza : 30 hours

Pranav : 30 hours

Total workload : 150 hours