UniTrade

Sprint 1 Planning Document

Team 19

Tong An

Yi Dou

Yancheng Guo

Scott Merritt

Yuening Xu

Sprint Overview:

During this first sprint, our main focus will be creating the most basic functions of the app. This will include things like creating a profile, logging in to an already existing profile, authentication, viewing the homepage, selecting an item from the homepage, and creating a new post. Firstly, this involves setting up our development environments (Android Studio, Firebase, and Node.js). Next, we will configure our database through Firebase. From there, we will be able to establish our server functions (also through Firebase) and subsequently connect them to our frontend work in Android Studio to create the core functionality of the app described earlier.

Scrum Master: Tong An

Meeting Plan: Tuesday 2:00-4:00PM, Thursday 3:30-5:30PM.

Risks and Challenges:

The initial challenges will mostly likely come from the initial development setup. This includes establishing git protocols, code standards for the group, setting up the right IDE's for each developer, and making sure everyone is doing their assigned portion of the sprint. Some risks we might face in this sprint may be adhering to the design document, since this will be the first time we can translate our ideas into code. As such, this may require us to backtrack and discuss issues with the whole team often to ensure we are on staying on the right track.

Current Sprint Detail

User Story 1:
As a user, I would like to register a new account.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Set up a server through Firebase	4	Tong An
2	Set up a database through Firebase	4	Tong An
3	Ensure that server is able to properly transfer data to and from the database	3	Tong An
4	Implement ability to store user information in database	3	Tong An
5	Implement a mailer	3	Scott Merritt
6	Set up Android Studio Environment	5	Yi Dou
7	Create login/register page User Interface	3	Yi Dou
8	Unit Test-Implement tests to ensure correct output is given	2	Yuening Xu

Acceptance Criteria:

- Given that the login/register interface is functional, when a user selects the ‘Create an Account’ option, they should be taken to the login/register page to do so.
- Given that the backend to create users is functional, when a request to create an account is made, then the new account will be stored in the database given that there is not an already existing account with the provided email.
- Given that input checker is functional, when the user attempts to give invalid input (like an invalid email address), then they will be notified to try again.
- Given that an account has been created successfully when the user navigates to the next screen, the user should be prompted to the login interface.

User Story 2:
As a user, I would like to log into my account.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Integrate Firebase authentication process	3	Scott Merritt
2	Encrypt the password (and any other snesitive data) before any transfers occur	2	Scott Merritt
3	Implement ability to navigate to the home page if login successfully, or prompt them to try again on unsuccessful attempts	4	Scott Merritt
4	Unit Tests-Implement functionality of checking valid input when users login	2	Yancheng Guo

Acceptance Criteria:

- Given the functionality of login is implemented, when the user attempts to log in with the wrong username or password, then a popup with "Invalid username or password" will be displayed.
- Given the functionality of login is implemented, when the user logs in successfully, then the user will be navigated to the home page.
- Given the functionality of login is well implemented, when the user attempts to log in, the client will use a string checker to examine the input first (for instance, to prevent malicious users from injecting JS code that could be harmful to the system).

User Story 3:
As a user, I would like to logout.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement the functionality in frontend that brings user back to login interface	2	Yi Dou
2	Implement the functionality in frontend that show a popup with success message after logging out successfully	1	Yi Dou

Acceptance Criteria:

- Given the functionality of logging out is well implemented, when users click the “logout” button, then the client would turn back to the login page.
- Given the functionality of logging out is well implemented, when users click “logout” button, then the server will not reply to the frontend until users login again.
- Given the popup message function is well implemented, when a user logs out, the client will notify them

User Story 4:
As a user, I would like to reset my password through email in case I forget it.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement ability to generate a random password and update the database	3	Yancheng Guo
2	Implement ability to send the random password to user's email address	2	Yancheng Guo
3	Implement the interface and functionality to reset password	2	Yancheng Guo

Acceptance Criteria:
- Given that the user class is well defined, the backend will be able to find the user according to the email address that the user provides.
- Given that the proper server functions have been implemented, the server will be able to generate and update a random password to the user.
- Given that the functionality of sending emails is well implemented, the server will send the updated random password to the user’s email address.
- Given the that the interface of login is well implemented, when the user clicks on “forgot my password”, the client will be directed to the “reset password” interface.

User Story 5:
As a user, I would like to retrieve my username if I forget it.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement server function to find username associated with the given email in the database	4	Tong An
2	Implement ability to send the username to the user's email address	4	Scott Merritt
3	Implement the interface and functionality to retrieve username in the frontend	4	Yancheng Guo

Acceptance Criteria:

- Given that the user class is well defined, the backend will be able to find the user according to the email address that the user provides.
- Given that the functionality of sending email is well implemented, the client will send the username to the user’s email address.
- Given the that the interface of login is well implemented, when the user clicks on “forgot my username”, the client will be directed to the “reset password/ retrieve username” interface.

User Story 6:
As a user, I would like to view and edit my profile.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement functionality of updating changes to personal profile in the backend	3	Scott Merritt
2	Implement functionality to allow users to change notification settings from the app	3	Tong An
3	Implement interface to allow users to edit their profile	4	Yancheng Guo
4	Unit Test- Implement tests for checking valid input when user editing profile	2	Yancheng Guo

Acceptance Criteria:
- Given that the user class is well defined, the backend will be able to find the user’s profile according to the current user ID.
- Given that the server/database are well implemented, they will be able to update the user’s settings when they submit changes
- Given that the profile class is well defined, the backend will be able to get and set the user’s personal information (i.e. phone number, address, etc.)
- Given that the user login interface is functional, the user will be able to navigate to “my profile” page easily when they wish to look at their profile

User Story 7:
As a seller, I would like to post my items on the app.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement the interface where user can fill out an information form and submit the post	4	Yi Dou
2	Implement the functionality in the backend to create a new post and store it in the database	4	Scott Merritt
3	Implement function to update seller’s list of items	2	Scott Merritt
3	Unit Test-Implement functionality of checking the validity of input information	2	Yancheng Guo

Acceptance Criteria:

- Given that the item class is well defined, the backend will be able to create a new item using user ID, price, category, and detailed description.
- Given that the post class is well defined, the backend will be able to create a new post using date, location, and itemID when a post is created.
- Given that the profile class is well implemented, the backend will be able to store the item into the “my items” list of the user that posted it.
- Given that the home page new post button is functional, the user will be able to see the posting interface when clicking the new post button.
- Given that the input-checker is well implemented, the client will be able to notify the user when they have inputted the information incorrectly.

User Story 8:
As a seller, I would like to set the price of the item I wish to sell.
As a seller, I would like to add details about the item I wish to sell.
(the functionality of these user stories are similar enough we could put them together)

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement the interface where the user can edit and update their post's information	3	Yi Dou
2	Implement the functionality in the backend to update the post's information based on the the user's input	3	Tong An
3	Unit Tests-Implement tests for checking the validity of user's input.	1	Yancheng Guo

Acceptance Criteria:
- Given that the functionality of editing a post is well implemented, then the post's information should be updated in the database when the user submits changes
- Given that the functionality of posting items is well implemented, when the user put invalid inputs, the input-checker will prompt an error message.
- Given that the functionality of editing a post is well implemented, then the user should be redirected to the post page of that post when they have finished editing it.

User Story 9:
As a seller, I would like the ability to delete a post I no longer want listed.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Add delete option to a Post’s interface	2	Yuening Xu
2	Implement backend functionality to update database with deleted post (that is, removing it from the database)	2	Yuening Xu
4	Unit Tests-Implement the functionality of checking the validity of input when user edits the item’s information.	2	Yancheng Guo

Acceptance Criteria:
- Given the functionality of deleting a post is well implemented, a post should be deleted when the seller chooses to delete it.
- Given that the server/database are well implemented, a post should no longer appear in the seller’s “my items” list.
- Given that the functionality of deleting a post is well implemented, the user should be redirected to their profile page when they have deleted their post

User Story 10:
As a user, I would like to see the list of my posted items.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement the ability to request user's posted items from the server to the client	4	Yuening Xu
2	Implement the functionality and interface in frontend where the user can view a list of posted items	4	Yi Dou
3	Implement the functionality that the items that have been sold will have special tag.	2	Yancheng Guo

Acceptance Criteria:

- Given the server/database are well implemented, the server should return the list of posts from a user from the database when the client requests it.
- Given that the Profile interface is well implemented, the user should be able to see a clear list of their posts when viewing their profile
- Given the functionality of seeing the posted items list is well implemented, when a user clicks one of the listed posts, then they will be redirected to the post page for that post
- Given that the server/database are well implemented, the server should be able to retrieve the post page from the database for a selected post from the list of a user's posts

User Stroy 11:
As a user, I would like to add/remove an item to/from a wishlist.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement an 'add/remove' button and the functionality on the interface of item detail page to allow user add item to their wishlist	4	Yuening Xu
2	Implement functionality in backend to store the item into user's wishlist in database	3	Yuening Xu

Acceptance Criteria:

- Given that the add function is well developed, when the user clicks on ”add”, the item will be added to the wishlist.
- Given that the add/remove function is well developed, after the user clicks the “add”, this button will become a “remove” button
- Given that the remove function is well developed, if the user selects to ‘remove’ an item from their wishlist, then that item will be removed
- Given to the backend is well developed, a user’s wishlist will be updated when a user adds/removes an item from it.

User Story 12:
As a user, I would like to view my wishlist.

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement the interface in the profile page to display a user’s wishlist	3	Yuening Xu
2	Implement the functionality in frontend that request the user's wishlist from backend	2	Yuening Xu

Acceptance Criteria:

- Given that the interface of wishlist is well implemented, when a user clicks on the wishlist button, then user will be navigated to the interface with a list of items in their wishlist.
- Given that the functionality of wishlist is well implemented, when user clicking on an item in the list, then the user will be navigated to that item’s detail page.
- Given that the server/database are well implemented, then the server will be able to retrieve an item’s detail page from the database when a user requests it.

User story 13:
As a user, I would like to be able to see a post’s detail page

Task Table:

Task Number	Description	Estimated Hours	Task Owner
1	Implement the functionality to have client request post page from server/database	3	Yuening Xu
2	Implement server functionality to retrieve and send post page from the database to the client	2	Tong An
3	Implement a post page for the buyer’s perspective	4	Yi Dou

Acceptance Criteria:

- Given that the interface is well-developed, when users click on a item, they will see the detail page of this item.
- Given that the backend is well implemented, the server will be able to retrieve the detail page from the database when a client requests it
- Given that the interface is well-developed, the user should be able to clearly understand the details of an item when viewing a post page.

Remaining Product Backlog

Functional

1. ~~As a user, I would like to register a new account.~~
2. ~~As a user, I would like to log into my account.~~
3. ~~As a user, I would like to reset my password through email in case I forget it.~~
4. ~~As a user, I would like to retrieve my username if I forget it.~~
5. ~~As a user, I would like to access UniTrade on an Android device.~~
6. ~~As a user, I would like to view and edit my profile.~~
7. As a user, I would like to report technical issues or inappropriate information to the administrator.
8. As a user, I would like to find proper items according to the category and keyword I use.
9. As a user, I would like to be able to see a page with the specific details of an item I've selected.
10. As a user, I would like to change the posts I see based on date, price, or distance.
11. As a user, I would like to see nearby items on the homepage of the app.
12. As a user, I would like to see my purchase/sale history.
13. As a user, I would like to change notification settings.
14. ~~As a user, I would like to add an item to a wishlist.~~
15. ~~As a user, I would like to view my wishlist.~~
16. ~~As a user, I would like to see the list of my items.~~
17. As a user, I would like to know when the price changes for items in my wishlist.
18. As a user, I would like to be able to message a buyer/seller to arrange face-to-face trades, if time permits.
19. As a user, I would like to be able to change the UI style, if time permits
20. As a user, I would like to be notified when another user messages me.
21. ~~As a user, I would like to logout.~~
22. ~~As a seller, I would like to post my items on the app.~~
23. ~~As a seller, I would like to set the price of the item I wish to sell.~~
24. ~~As a seller, I would like to add details about the item I wish to sell.~~
25. As a seller, I would like to include pictures of the item I wish to sell.
26. ~~As a seller, I would like the ability to edit the information about my items.~~

~~27. As a seller, I would like to delete a post I no longer want listed~~

28. As a seller, I would like to receive payments online when the trade is made.
29. As a seller, I would like to be notified when a buyer has purchased my item.
30. As a seller, I would like to submit a request to the buyer for a face-to-face meet up for trading an item, rather than shipping it.
31. As a buyer, I would like to view the reviews of a seller before purchasing an item.
32. As a buyer, I would like to write a review for a seller after completing a trade.
33. As a buyer, I would like to make payments online.
34. As a buyer, I would like to submit a request to the seller for a face-to-face meet up for trading an item, rather than shipping it.
35. As a buyer, I would like to see a seller's profile.
36. As a buyer, I would like to be notified when the seller ships an item.

37. As a buyer, I would like to submit a request to the seller for a face-to-face meet up for trading an item, rather than shipping it.
38. As a buyer, I would like to know the tracking number if I don't choose a face-to-face trade.
39. As an administrator, I would like to see reports from users.
40. As an administrator, I would like to delete inappropriate posts or comments.

Non-functional

1. ~~We must be able to run this app on an Android device, using the tools offered by Android Studio to build it. This includes making the user interface clear and easy to use.~~
2. An administrative program must be able to run on a desktop, allowing an administrator to perform special actions list above
3. The system must have the ability to handle at least 100 users at a time.
4. The app must be able to allow payments through current API (PayPal, initially, but can be expanded to include credit/debit options if time permits).
5. The app must be able to calculate distances between users through the app's API. Distances will be calculated based on locations of buyer and seller. As such, users are required to include some kind of address in their profile for these calculations.
6. ~~Passwords stored in the database will be encrypted using SQL hash functions.~~
7. The response time of the server needs to be less than 800ms.