TroveSpace

Sprint 3 Planning Document

Team 13 | https://github.com/baker323/TroveSpace

Adam Baker, Kareem Elhadidi, Rei Orikata, Pritesh Kadiwala

Contents

Sprint Overview

For this sprint, we will focus on implementing notifications, comments, and ratings, as well as allowing users to view the collections of other users. We will also let users view a list of other collectors who currently have each specific collectible in their collection or wishlist. We will give users the option of displaying their email address as part of marking items that are for sale so that they can be contacted. Users will be able to follow troves to see when new items are added, as well as receive notifications when an item in their collection is edited by another user so that they can vote on the changes. We will improve the user experience by automatically removing an item from a user's wishlist when they add it to their collection and allow users to mark duplicate items as well as request troves to be removed.

Scrum Master: Kareem Elhadidi

Meeting Schedule: Mondays, Wednesdays, and Fridays at 3:30PM

Risks/Challenges: A challenge will be figuring out a way to deliver notifications to the correct users so that they can view them when they open the app. We will need to make sure users can track which notifications have been read and provide a way to view them in an organized fashion. We will also need to make sure users have a way to keep their email address private if they choose, but if they want to mark items that they are selling, they will need to be made aware that their email address will be shared.

Current Sprint Detail

User Story #13

As a user, I would like to vote on whether or not an item should be marked as a duplicate

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for marking duplicates	2	Kareem
2	Implement Firebase methods for marking duplicates	3	Adam
3	Implement Firebase methods for removing duplicates	2	Rei
4	Implement error handling for UI	1	Pritesh
5	Perform tests on marking duplicates	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see a button to mark it as a duplicate.
- Given that the frontend is set up correctly, when a user clicks the mark as duplicate button, then they should be prompted to select the name of the similar collectible.
- Given that the frontend is set up correctly, when a user submits the name of the similar collectible, then it should appear next to the mark as duplicate button.
- Given that the frontend is set up correctly, when a user clicks the mark as duplicate button, it should increase the duplicate vote count in the database.
- Given that the database is set up properly, when a user sends a request to mark a duplicate item, then it should be recorded in the database.
- Given that the database is set up properly, when a collectible receives the required amount of votes, then the duplicate should be removed from the database.

User Story #15

As a user, I would like to be notified when an item in my collection is edited by another user

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for notifications	5	Kareem
2	Implement Firebase methods for notifications	5	Adam
3	Implement error handling for UI	3	Pritesh
4	Perform tests on notifications	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user edits a collectible, then all users who currently have the item in their collection should receive a notification.
- Given that the frontend is set up correctly, when a user logs into their account, then they should see a notification icon.
- Given that the frontend is set up correctly, when a user clicks the notification icon, then their notifications will be displayed.
- Given that the frontend is set up correctly, when a user clicks on a notification, then it will be removed from their unread notifications and take them to the collectible page.
- Given that the database is set up properly, when a user sends a request to edit an item, then the notification should be recorded for each corresponding user in the database.
- Given that the database is set up properly, when a user sends a request to mark a notification as read, then it should be recorded in the database.

User Story #18

As a user, I would like to see the number of people who have a specific item in their collection

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for viewing user count	2	Kareem
2	Implement Firebase methods for updating user count	3	Adam
3	Implement Firebase methods for fetching user count	2	Rei
4	Implement error handling for UI	2	Pritesh
5	Perform tests on count functionality	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see the number of people who currently have it in their collection.
- Given that the frontend is set up correctly, when a user adds a collectible to their collection, then the count of people who have it in their collection should increase by one.
- Given that the frontend is set up correctly, when a user removes a collectible from their collection, then the count of people who have it in their collection should decrease by one.
- Given that the database is set up properly, when a user sends a request to increase the count of users with the item in their collection, then it should be recorded in the database.
- Given that the database is set up properly, when a user sends a request to decrease the count of users with the item in their collection, then it should be recorded in the database.

User Story #19

As a user, I would like to see the number of people who have a specific item in their wishlist

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for viewing user count	2	Kareem
2	Implement Firebase methods for updating user count	3	Adam
3	Implement Firebase methods for fetching user count	2	Rei
4	Implement error handling for UI	2	Pritesh
5	Perform tests on count functionality	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see the number of people who currently have it on their wishlist.
- Given that the frontend is set up correctly, when a user adds a collectible to their wishlist, then the count of people who have it on their wishlist should increase by one.
- Given that the frontend is set up correctly, when a user removes a collectible from their wishlist, then the count of people who have it in their wishlist should decrease by one.
- Given that the database is set up properly, when a user sends a request to increase the count of users with the item on their wishlist, then it should be recorded in the database.
- Given that the database is set up properly, when a user sends a request to decrease the count of users with the item on their wishlist, then it should be recorded in the database.

User Story #25

As a user, I would like to view another user's collection

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for viewing collections	5	Kareem
2	Implement Firebase methods for fetching collections	5	Adam
3	Implement error handling for UI	3	Pritesh
4	Perform tests on viewing collections	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user clicks on the name of a user from a list of users on the collectible page, then they should see that user's collection.
- Given that the frontend is set up correctly, when a user visits another user's collection page, then they should see all collectibles that the other user has in their collection.
- Given that the frontend is set up correctly, when a user clicks on a collectible in the other user's collection, then they should see the details of that collectible.
- Given that the frontend is set up correctly, when a user visits another user's collection page, then they should see the username of the other user displayed near the top.
- Given that the database is set up properly, when a user sends a request to view another user's collection, then the other user's collection data should be retrieved.

User Story #26

As a user, I would like to view another user's wishlist

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for viewing wishlists	2	Kareem
2	Implement Firebase methods for fetching wishlists	5	Adam
3	Implement error handling for UI	2	Pritesh
4	Perform tests on viewing wishlists	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user clicks on the name of a user from a list of users on the collectible page, then they should see that user's wishlist.
- Given that the frontend is set up correctly, when a user visits another user's wishlist page, then they should see all collectibles that the other user has in their wishlist.
- Given that the frontend is set up correctly, when a user clicks on a collectible in the other user's wishlist, then they should see the details of that collectible.
- Given that the frontend is set up correctly, when a user visits another user's wishlist page, then they should see the username of the other user displayed near the top.
- Given that the database is set up properly, when a user sends a request to view another user's wishlist, then the other user's wishlist data should be retrieved.

User Story #27

As a user, I would like to rate items on a 5 star scale

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for rating items	3	Kareem
2	Implement UI for viewing ratings	2	Kareem
3	Implement Firebase methods for rating items	3	Adam
4	Implement Firebase methods for fetching ratings	2	Rei
5	Implement error handling for UI	3	Pritesh
6	Perform tests on rating items	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see the option to submit a rating out of five stars.
- Given that the frontend is set up correctly, when a user selects a rating out of five stars, then the overall rating of the collectible should be updated.
- Given that the frontend is set up correctly, when a user attempts to change their rating after they have already submitted, they should receive an informative error message.
- Given that the database is set up properly, when a user sends a request to rate a collectible, then their rating should appear in the database.
- Given that the database is set up properly, when a user sends a request to rate a collectible, then the rating count should increase in the database.

User Story #28

As a user, I would like to comment on items

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for commenting	3	Pritesh
2	Implement UI for viewing comments	2	Kareem
3	Implement Firebase methods for commenting	3	Rei
4	Implement Firebase methods for fetching comments	2	Rei
5	Implement error handling for UI	3	Pritesh
6	Perform tests on commenting	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see a place to enter comments.
- Given that the frontend is set up correctly, when a user enters text into the comment box and clicks submit, then their comment should be submitted and displayed to all users.
- Given that the frontend is set up correctly, when a user clicks the upvote button on a comment, then the upvote count should increase by one.
- Given that the frontend is set up correctly, when a user clicks the downvote button on a comment, then the downvote count should increase by one.
- Given that the frontend is set up correctly, when a user clicks the upvote or downvote button again, then their vote should be removed and the count should decrease.
- Given that the frontend is set up correctly, when a user clicks the upvote button after clicking the downvote button and vice versa, then their vote should change.
- Given that the frontend is set up correctly, when a user clicks the reply button on a comment, then they should be able to input a comment as a reply to that comment.
- Given that the database is set up properly, when a user sends a request to submit a comment, then the comment should appear in the database.
- Given that the database is set up properly, when a user sends a request to upvote or downvote a comment, then the corresponding count should increase in the database.

User Story #31

As a user, I would like to have items removed from my wishlist automatically when I add them to my collection

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for removing items	2	Kareem
2	Implement Firebase methods for removing items from wishlist	3	Adam
3	Implement error handling for UI	2	Pritesh
4	Perform tests on removing items	3	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user adds a collectible to their collection that is in their wishlist, then it should be removed from their wishlist.
- Given that the frontend is set up correctly, when a user adds a collectible to their collection that is not in their wishlist, then nothing should be removed from their wishlist.
- Given that the frontend is set up correctly, when a user visits their wishlist after adding an item from their wishlist to their collection, then the item should not appear in their wishlist.
- Given that the database is set up properly, when a user sends a request to add an item to their collection, then the item should be removed from their wishlist in the database.
- Given that the database is set up properly, when a user sends a request to add an item to their collection, then the item should appear in their collection in the database like normal.

User Story #32

As a user, I would like to see a list of all users who own a specific item

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for viewing user list	5	Kareem
2	Implement Firebase methods for fetching user list	5	Adam
3	Implement error handling for UI	3	Pritesh
4	Perform tests on viewing user list	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see a button that will show a list of users who have the item in their collection.
- Given that the frontend is set up correctly, when a user clicks the button, then they should see a list of all users who currently have the collectible in their collection.
- Given that the frontend is set up correctly, when a user removes the item from their collection, then their name should be removed from the list for that collectible.
- Given that the frontend is set up correctly, when a user adds the item to their collection, then their name should be added to the list for that collectible.
- Given that the database is set up properly, when a user sends a request to fetch the list of users who have a collectible in their collection, then the data should be retrieved.
- Given that the database is set up properly, when a user sends a request to remove an item from their collection, then the database should update to reflect that change.
- Given that the database is set up properly, when a user sends a request to add an item to their collection, then the database should update to reflect that change.

User Story #33

As a user, I would like to follow a Trove to see when new items are added

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for following troves	5	Kareem
2	Implement Firebase methods for following troves	5	Adam
3	Implement error handling for UI	2	Pritesh
4	Perform tests on following troves	3	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views all troves, then they will be able to click on a button that will show them the troves that they follow.
- Given that the frontend is set up correctly, when a user views a trove, then they should see a follow button.
- Given that the frontend is set up correctly, when a user clicks the follow button, then they should see the trove in their list of followed troves.
- Given that the frontend is set up correctly, when a user is already following a trove, then the follow button should be replaced by an unfollow button.
- Given that the frontend is set up correctly, when a user clicks the unfollow button, then they should no longer see the trove in their list of followed troves.
- Given that the frontend is set up correctly, when a user visits their list of followed troves, then they will be able to click on a trove and see all the collectibles that were added.
- Given that the database is set up properly, when a user sends a request to follow a trove, then the trove should be added to the user's followed troves in the database.
- Given that the database is set up properly, when a user sends a request to unfollow a trove, then the trove should be removed from the user's followed troves in the database.

User Story #42

As a user, I would like to request for a Trove to be removed

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for requesting removal	1	Kareem
2	Implement Firebase methods for requesting removal	2	Rei
3	Implement error handling for UI	1	Pritesh
4	Perform tests on requesting removal	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see an option to request its removal.
- Given that the frontend is set up correctly, when a user clicks the request removal button, then they will receive a message that their request has been submitted.
- Given that the frontend is set up correctly, when a user has already submitted a removal request, then they should be unable to submit another request.
- Given that the database is set up properly, when a user sends a request to remove a trove, then the request should be recorded in the database.
- Given that the database is set up properly, when a database administrator views the database, then he should see how many removal requests were submitted and remove troves based on his discretion and how many users are actively using the trove.

User Story #43

As a user, I would like to mark items in my collection as “for sale”

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for marking “for sale”	2	Rei
2	Implement Firebase methods for marking “for sale”	5	Pritesh
3	Implement error handling for UI	1	Pritesh
4	Perform tests on marking “for sale”	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible in their collection, they should see the option to mark it as “for sale.”
- Given that the frontend is set up correctly, when a user clicks the “for sale” button, then they should receive a message explaining that their email will be shared with other users.
- Given that the frontend is set up correctly, when a user clicks the “for sale” button, then the collectible should be marked as “for sale” and appear under the listings for that item.
- Given that the frontend is set up correctly, when a user unchecks the “for sale” button, then the collectible should be removed from the listings for that collectible.
- Given that the database is set up properly, when a user sends a request to mark a collectible as “for sale,” then the user’s listing should appear in the database.
- Given that the database is set up properly, when a user sends a request to mark a collectible as not “for sale,” then the user’s listing should not appear in the database.

User Story #44

As a user, I would like to be able to see all “for sale” listings of a specific item

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for viewing “for sale” listings	5	Kareem
2	Implement Firebase methods for fetching “for sale” listings	5	Pritesh
3	Implement error handling for UI	3	Pritesh
4	Perform tests on “for sale” listings	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible, then they should see a button that lets them view a list of users who have the item for sale.
- Given that the frontend is set up correctly, when a user clicks the button, then they should see a list of all users who currently have the collectible listed as “for sale.”
- Given that the frontend is set up correctly, when a user clicks a name from the list of users, then they should be taken to that user’s collection page and see their email.
- Given that the frontend is set up correctly, when a user clicks the button to view “for sale” listings and there are no users with the item for sale, then they should receive an error.
- Given that the database is set up properly, when a user sends a request to fetch a list of all users who have a collectible marked as “for sale,” then the data should be retrieved.

Non-Functional

- Application will be adequately responsive and easy to use.
- Application will be fully functional on Chrome (and Firefox if time permits).
- Application will be intuitive to use.
- Application will have a visually appealing UI.
- Application will store the user data safely and securely on Firebase.
- Application will securely transmit data to and from Firebase.
- Application will resize page content based on window size.
- Application will have cleanly organized and well-commented code.
- Application will be maintainable and testable.
- Application will produce useful error messages when required.

Total Sprint Workload

Developer's Name	**Estimated Time**
Adam Baker	40 Hours
Kareem Elhadidi	43 Hours
Rei Orikata	43 Hours
Pritesh Kadiwala	44 Hours
Total	**170 Hours**

Remaining Backlog

ID	Function Requirement	Hours
17	As a user, I would like to be able to see the edit history of an item (if time permits)	30
29	As a user, I would like to offer trades/purchases to other users (if time permits)	25
30	As a user, I would like to rate trades/purchases with other users (if time permits)	20
34	As a user, I would like to add friends (if time permits)	20
35	As a user, I would like to be notified when my friends add a new item to their collection (if time permits)	15
36	As a user, I would like to be notified when my friends add a new item to their wishlist (if time permits)	10