TroveSpace

Sprint 1 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 designing the UI, creating an account system, and achieving smooth communication between the front end and database. We will set up the ability to create Troves, collections, and collectibles while saving detailed features for future sprints.

Scrum Master: Kareem Elhadidi

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

Risks/Challenges: Getting the Firebase set up and working will be our main challenge due to the large amount of data and the number of front end methods needed to exchange that data. Designing the UI will also be a challenge because the complexity of the information will generally require sophisticated styling to make all the pages interactive and clean looking.

Current Sprint Detail

User Story #1

As a user, I would like to open the application from Chrome Desktop web browser (initial setup for frontend and backend).

#	Task Description	Estimated Time	Owner(s)
1	Set up frontend pages and layout	5	Pritesh
2	Set up AngularJS controllers for each module	5	Kareem
3	Set up Firebase to store data objects	5	Adam

Acceptance Criteria:

- Given that the frontend is set up properly, when a user opens the application, then they should see a login screen.
- Given that AngularJS is set up properly, when a user presses a button, then the data is properly passed to and processed by the corresponding controller.
- Given that the backend is set up properly, when a user makes API calls from the frontend, then they can be accepted and returned successfully.
- Given that the database is set up correctly, when a user sends data to be stored, then the data should appear in the database.
- Given that the database is set up correctly, when a user opens the application, then a connection is established with the database and the correct information is displayed.

User Story #2

As a user, I would like to create an account in the application.

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for account creation	3	Pritesh
2	Set up communication with Firebase to allow account authentication	3	Adam
3	Implement error handling for UI	2	Pritesh
4	Perform tests on account creation	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user visits the login page, then they should see fields for entering their account details to create an account.
- Given that the frontend is set up correctly, when a user enters their information and clicks the submit button, then the user's information should be sent to the database.
- Given that the database is set up correctly, when a user submits their information, then their data should appear in the database.
- Given that the frontend is set up correctly, when a user enters incomplete, invalid or already existing credentials, then they should receive an informative error message.
- Given that the frontend is set up correctly, when a user successfully creates an account, they should be logged in and taken to their profile page.

User Story #3

As a user, I would like to login to my account.

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

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user visits the login page, then they should see fields for entering their username and password to log into the application.
- Given that the frontend is set up correctly, when a user enters their information and clicks the login button, then they should be logged in and taken to their profile page.
- Given that the database is set up correctly, when a user sends a valid login request, then the user should be authenticated and allowed to login.
- Given that the database is set up correctly, when a user sends an invalid login request, then the user should be rejected and not allowed to login.
- Given that the frontend is set up correctly, when a user enters invalid or incomplete credentials, then they should receive an informative error message.

User Story #4

As a user, I would like to logout of my account.

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

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user is logged in, then they should see the option to log out in the navigation menu.
- Given that the frontend is set up correctly, when a user clicks the logout button, then they should be logged out and taken back to the login screen.
- Given that the backend is set up correctly, when a user sends a request to log out, then that user is marked as logged out and restricted from sending requests.
- Given that the backend is set up correctly, when a user sends a request for information after they log out, then their request is denied and they are not allowed to see the data.
- Given that the frontend is set up correctly, when a user is logged out, the option to log out should no longer appear in the navigation menu.

User Story #5

As a user, I would like to change my account password.

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

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user is logged in, they should see the option to change their password in the navigation menu.
- Given that the frontend is set up correctly, when a user is not logged in, they should see a forgot password link on the login screen.
- Given that the frontend is set is set up correctly, when a user clicks the change password button, they should be taken a form with fields to enter their current and new passwords.
- Given that the frontend is set up correctly, when a user clicks the forgot password link, they should be prompted to enter their email address to be sent a password reset link.
- Given that the backend is set up correctly, when a user sends a request to change their password, then their password should be updated in the database.

User Story #6

As a user, I would like to delete my account.

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for account deletion	1	Pritesh
2	Implement Firebase methods for deleting accounts	2	Adam
3	Implement error handling for UI	1	Pritesh
4	Perform tests on account deletion	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user is logged in, then they should see the option to delete their account in the navigation menu.
- Given that the frontend is set up correctly, when a user clicks the delete account button, then they should be prompted to confirm their action.
- Given that the frontend is set up correctly, when a user confirms their account deletion, then their account should be deleted and they should be taken back to the login screen.
- Given that the frontend is set up correctly, when a deleted user tries to sign in again, they will receive an informative error message and not be allowed to log in.
- Given that the backend is set up correctly, when a user sends a request to delete their account, then their account should be deleted from the database.

User Story #7

As a user, I would like to have multiple distinct collections.

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for creating collections	3	Kareem
2	Implement UI for viewing collections	3	Kareem
3	Implement UI for renaming collections	1	Kareem
4	Implement UI for deleting collections	1	Kareem
5	Implement UI for switching between collections	1	Kareem
6	Implement Firebase methods for creating a collection	3	Adam
7	Implement Firebase methods for fetching a user's collection	2	Adam
8	Implement Firebase methods for renaming a user's collection	2	Adam
9	Implement Firebase methods for deleting a collection	2	Adam
10	Implement error handling for collection creation	1	Kareem
11	Implement error handling for collection renaming	1	Kareem
12	Implement error handling for collection deletion	1	Kareem
13	Perform tests on collection creation	1	Rei
14	Perform tests on collection viewing	1	Rei
15	Perform tests on collection renaming	1	Rei
16	Perform tests on collection deletion	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user visits their profile, they should see the options to view their collections and create a new collection.
- Given that the frontend is set up correctly, when a user clicks the create collection button, then they should be prompted to enter a name and category for the collection.
- Given that the frontend is set up correctly, when a user enters collection information and clicks submit, a new collection should be created and viewable in the user's collections.
- Given that the frontend is set up correctly, when a user enters an invalid collection name, then they will receive an informative error message.
- Given that the frontend is set up correctly and collectibles are implemented, when a user views a collection, they should see all the collectibles that they added to their collection.
- Given that the frontend is set up correctly, when a user views a collection, they should see the options to switch to viewing another collection and delete the collection..
- Given that the frontend is set up correctly, when a user clicks the button to view another collection and chooses a collection, the page should update with that collection's data.
- Given that the frontend is set up correctly, when a user clicks the button to delete a collection, the collection they are currently viewing should be deleted from the database.
- Given that the frontend is set up correctly, when a user views a list of all their collections, the renamed collections are updated and the deleted collections are no longer visible.
- Given that the database is set up correctly, when a user sends a request to create a collection, the new collection appears in the database.
- Given that the database is set up correctly, when a user sends a request to rename a collection, the new name appears in the database.
- Given that the database is set up correctly, when a user sends a request to delete a collection, the collection is removed from the database.
- Given that the database is set up correctly, when a user sends a request to view a collection, then they should be sent the data for that collection.

User Story #8

As a user, I would like to be able to track duplicate items in a single entry.

#	Task Description	Estimated Time	Owner(s)
1	Implement UI for updating duplicates	2	Kareem
2	Implement UI for showing duplicates	1	Kareem
3	Implement Firebase methods for updating the number of duplicates	2	Rei
4	Implement Firebase methods for fetching the number of duplicates	2	Rei
5	Implement error handling for UI	1	Kareem
6	Perform tests on updating duplicates	1	Rei
7	Perform tests on showing duplicates	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views a collectible in their collection, then they should see the number of that item they own.
- Given that the frontend is set up correctly, when a user views a collectible in their collection, then they should see an option to change the number of that item they own.
- Given that the frontend is set up correctly, when a user enters a number and submits, then the page will update to reflect the change.
- Given that the frontend is set up correctly, when a user enters an invalid number, then they should receive an informative error message.
- Given that the database is set up correctly, when a user sends a request to update the number owned of a particular collectible, the new number will appear in the database.

User Story #10

As a user, I would like to add items that do not exist in the TroveSpace database.

#	Task Description	Estimated Time	Owner(s)
1	Implement the UI for creating collectibles	10	Kareem
2	Implement Firebase methods for creating collectibles	10	Rei
3	Implement Firebase methods for adding collectibles to Troves	5	Rei
4	Implement error handling for UI	2	Kareem
5	Perform tests on collectible creation	3	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly and Troves are implemented, when a user views a Trove, they should see the option to create a new collectible.
- Given that the frontend is set up correctly, when a user clicks the button to create a new collectible, they should be prompted to enter item details pertaining to that Trove.
- Given that the frontend is set up correctly, when the user enters information and clicks submit, then the user will be taken to the newly created page for that collectible.
- Given that the frontend is set up correctly, when the user creates a collectible, then the entered information appears on the newly created page for that collectible.
- Given that the frontend is set up correctly, when the user views the Trove page, then the newly created collectible is visible under its respective Trove.
- Given that the frontend is set up correctly, when the user enters invalid or incomplete information, then they will receive an informative error message.
- Given that the database is set up correctly, when the user sends a request to create a new collectible, then the collectible data appears in the database.
- Given that the database is set up correctly, when the user sends a request to view a Trove, then the collectible data appears under that specific Trove in the database.

User Story #11

As a user, I would like to add items that already exist to one or more of my collections.

#	Task Description	Estimated Time	Owner(s)
1	Implement the UI for adding collectibles to collections	3	Pritesh
2	Implement Firebase methods for adding collectibles to collections	3	Adam
3	Implement error handling for UI	2	Pritesh
4	Perform tests on adding collectibles to collections	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user visits a collectible page, then they should see the option to add the item to their collection.
- Given that the frontend is set up correctly, when a user clicks add to collection and chooses an existing collection, then the item is added to their chosen collection.
- Given that the frontend is set up correctly, when a user does not have any collections, then they should receive an informative error message.
- Given that the frontend is set up correctly, when an item is added to a collection and a user views that collection, then that collectible should be visible in that collection.
- Given that the database is set up correctly, when a user sends a request to add an item to their collection, then the new item should appear under that collection in the database.

User Story #12

As a user, I would like to add items that already exist in the database to my wishlist.

#	Task Description	Estimated Time	Owner(s)
1	Implement the UI for adding collectibles to wishlist	1	Pritesh
2	Implement Firebase methods for adding collectibles to wishlist	2	Adam
3	Implement error handling for UI	1	Pritesh
4	Perform tests on adding collectibles to wishlist	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user visits a collectible page, then they should see the option to add the item to their wishlist.
- Given that the frontend is set up correctly, when a user clicks add to wishlist, then the item is added to their wishlist.
- Given that the frontend is set up correctly, when a user already has that item on their wishlist, then they should receive an informative error message.
- Given that the frontend is set up correctly, when an item is added to the wishlist and a user views their wishlist, then that collectible should be visible in the wishlist.
- Given that the database is set up correctly, when a user sends a request to add an item to their wishlist, then the new item should appear under their wishlist in the database.

User Story #23

As a user, I would like to create a new Trove.

#	Task Description	Estimated Time	Owner(s)
1	Implement the UI for Trove creation	3	Pritesh
2	Implement the UI for viewing Troves	3	Pritesh
3	Implement the UI for viewing all Troves	3	Pritesh
4	Implement Firebase methods for Trove creation	2	Adam
5	Implement Firebase methods for fetching Trove information	2	Adam
6	Implement Firebase methods for fetching a list of all Troves	2	Adam
7	Implement error handling for UI	2	Pritesh
8	Perform tests on Trove creation	1	Rei
9	Perform tests on Trove viewing	1	Rei
10	Perform tests on viewing all Troves	1	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user views their profile, they should see the option to create a new Trove.
- Given that the frontend is set up correctly, when a user clicks the button to create a new Trove, they should be prompted to enter unique details pertaining to that Trove.
- Given that the frontend is set up correctly, when the user enters information and clicks submit, then the user will be taken to the newly created page for that Trove.
- Given that the frontend is set up correctly, when the user creates a Trove, then the entered information appears on the newly created page for that Trove.
- Given that the frontend is set up correctly, when the user views a list of all Troves, then the newly created Trove is visible under the list of all Troves.
- Given that the frontend is set up correctly, when the user enters invalid or incomplete information, then they will receive an informative error message.
- Given that the database is set up correctly, when the user sends a request to create a new Trove, then the Trove data appears in the database.
- Given that the database is set up correctly, when the user sends a request to view all Troves, then the Trove data appears under the list of all Troves in the database.

User Story #24

As a user, I would like to name the fields in a Trove

#	Task Description	Estimated Time	Owner(s)
1	Implement the UI for naming fields	3	Kareem
2	Implement Firebase methods for naming Trove fields	3	Adam
3	Implement error handling for UI	2	Kareem
4	Perform tests on naming fields	2	Rei

Acceptance Criteria:

- Given that the frontend is set up correctly, when a user creates a Trove, then they should see an option to specify field names that describe items in that specific category.
- Given that the frontend is set up correctly, when a user enters naming information into the fields and clicks submit, then the names are visible when viewing the page for that Trove.
- Given that the frontend is set up correctly, when a user clicks the plus button, then they should be given the option to add additional field names to the Trove.
- Given that the frontend is set up correctly, when a user enters incomplete or invalid information into the fields, then they should receive an informative error message.
- Given that the database is set up correctly, when a user sends a request to name fields of a Trove, then the field names should update under that Trove in the database.

Total Sprint Workload

Developer's Name	**Estimated Time**
Adam Baker	40 Hours
Kareem Elhadidi	38 Hours
Rei Orikata	43 Hours
Pritesh Kadiwala	39 Hours
Total	**160 Hours**

Remaining Backlog

ID	Function Requirement	Hours
9	As a user, I would like to search for items that already exist in the TroveSpace database	30
13	As a user, I would like to vote on whether or not an item should be marked as a duplicate	20
14	As a user, I would like to edit items that already exist in the TroveSpace database	25
15	As a user, I would like to be notified when an item in my collection is edited by another user	15
16	As a user, I would like to vote on whether or not changes made to an item are accurate	10
17	As a user, I would like to be able to see the edit history of an item (time permitting)	30
18	As a user, I would like to see the number of people who have a specific item in their collection	15
19	As a user, I would like to see the number of people who have a specific item in their wishlist	10
20	As a user, I would like to create a new folder for items in a specific category	10
21	As a user, I would like to add an item that is in my collection to a folder	10
22	As a user, I would like to search for items in a specific Trove (a Trove is a category for items, for example: baseball cards, stamps, coins, etc)	15
25	As a user, I would like to view another user's collection	15
26	As a user, I would like to view another user's wishlist	10
27	As a user, I would like to rate items on a 5 star scale	15

28	As a user, I would like to comment on items	15
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
31	As a user, I would like to have items removed from my wishlist automatically when I add them to my collection	20
32	As a user, I would like to see a list of all users who own a specific item	15
33	As a user, I would like to follow a Trove to see when new items are added	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
37	As a user, I would like to search for items within one of my Troves	15
38	As a user, I would like to see the date an item was added to my collection	5
39	As a user, I would like to see the date an item was added to my wishlist	5
40	As a user, I would like to upload an image when creating an item	20
41	As a user, I would like to add details about my item based on the item's Trove	15
42	As a user, I would like to request for a Trove to be removed	5
43	As a user, I would like to mark items in my collection as "for sale" (if time permits)	10

44	As a user, I would like to be able to see all “for sale” listings of a specific item (if time permits)	15

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.