wander

Sprint 1 Retrospective

Team 18

Sai Edupulapati, Akshaya Kumar, Shreya Sanapala, Visa Thongdee, Vidya Vuppala

What Went Well

This sprint, we spent the majority of our time setting up our environment, which included producing UIs and creating and linking our backend. In general, we built the starting features of our project, such as user account creation and modification, profile creation, and linking to Spotify, and also began progress towards the Spotify components of our website (i.e. custom playlist creation).

User Story 1: As a user, I would like to be able to log in with my account.

#	Description	Estimated Time	Owner
1	Create skeleton UI with appropriate text fields and buttons	2 Hrs	Vidya
2	Create an algorithm to retrieve and verify the provided credentials from Firebase	2 Hrs	Akshaya
3	Set up appropriate error messages in case of information mismatch	2 Hr (each)	Akshaya, Vidya

4	Handle site navigation based on action of choice (login, reset, create, etc.)	3 Hrs (each)	Akshaya, Vidya
5	Create and handle logic for logging out	3 Hrs (each)	Akshaya, Vidya
6	Debug and test algorithm with unit testing	2 Hrs (each)	Akshaya, Vidya

Completed:

The user is greeted with a login page that includes the necessary text fields and buttons to input information with. The UI also includes buttons and corresponding redirects to pages that are meant for creating a new account and resetting passwords. Upon entering a username/email and password, the inputs are verified through a Firebase backend. If successful, the user is logged in and sent to the homepage of the website. If unsuccessful (i.e. incorrect username/email, invalid username/email, incorrect password), an error message appears.

User Story 2: As a user, I would like to be able to delete my account.

#	Description	Estimated Time	Owner
1	Set up button and display necessary prompts (confirmation, etc.)	1 Hr	Vidya
2	Create a pop up for users to input their current username, password, etc.	1 Hr	Vidya
3	Display successful confirmation message and navigate them back to homepage	1 Hr	Vidya
4	Implement appropriate logic for deletion of credentials and user data in Firebase	3 Hrs	Akshaya
5	Debug and test to make sure there are no security breaches after deletion	2 Hrs	Sai
6	Debug and test algorithms with unit testing	2 Hrs (each)	Vidya, Akshaya, Sai

Completed:

The user is first required to properly authenticate with their current email and password. After successful reauthentication, the corresponding document associated with the user is deleted in the Firebase backend. Finally, they will be greeted with an error if an attempt to log in again (after deleting the account) is made.

User Story 3: As a user, I would like to be able to create a profile with a username.

#	Description	Estimated Time	Owner
1	Create skeleton UI with appropriate text fields and buttons	2 Hrs	Visa
2	Set up Firebase backend to authenticate validity of user input	2 Hrs	Akshaya
3	Set up appropriate error messages in case of non-unique username	2 Hrs (each)	Akshaya, Visa
4	Set up appropriate error message in case of invalid profile picture file type	2 Hrs (each)	Akshaya, Visa
5	Use Firebase to link profile information with user account details	2 Hrs	Akshaya
^	Debug and test algorithm with unit testing	2 Hrs (each)	Visa, Akshaya

Completed:

The user is able to enter the registration page through a button from the login page and is faced with multiple text fields that are needed to create an account as well as a link/button to go back to the login page if needed. These fields include first name, last name, username, email address, password, and two security questions that will be required when resetting the account's password. After entering a unique username and password that are verified through a Firebase backend, if successful, the user is logged in and sent to the homepage of the website but if unsuccessful (e.g. not unique username/email, invalid password), an error message appears.

User Story 4: As a user, I would like to be able to modify my username.

#	Description	Estimated Time	Owner
1	Create an Edit Profile button	1 Hr	Visa
2	Prompt user with field to edit username contents	2 Hr	Visa
3	Create a save profile button	1 Hr	Visa
4	Update any of the saved changes in the database	3 Hrs	Akshaya
5	Debug and test algorithm with unit testing	2 Hrs (each)	Visa, Akshaya

Completed:

The user is first required to input a valid username, which is checked in the database for any duplicates. If none are found, then the username is successfully changed in the corresponding document in Firebase. Otherwise, an alert/error message is displayed to the user informing them that the username is already being used by a different account.

User Story 5: As a user, I would like to be able to view a profile page with my profile information.

#	Description	Estimated Time	Owner
1	Create a view profile button	1 Hr	Visa
2	Retrieve user’s full name, username, and email address from database	2 Hr	Sai
3	Create UI panel to display the user’s attributes	3 Hrs	Visa
4	Add button to handle the logout functionality	2 Hr (each)	Sai, Visa
5	Add button to navigate to the Edit Profile and Settings functionalities	1 Hr	Visa
6	Debug and test algorithm with unit testing	2 Hrs (each)	Sai, Visa

Completed:

When a user is logged in, they can either click the Profile button from the sidebar dropdown or click on their profile picture on the top right of the screen on the header to access the profile page. The profile page itself displays the user's profile picture, first and last name, username, and email, as well as two buttons to the side to access the edit profile and the settings page. Additionally, at the bottom of the page, there is a Logout button to handle the logout functionality as well.

User Story 6: As a user, I would like to be able to change my password if needed.

#	Description	Estimated Time	Owner
1	Create a skeleton UI with the appropriate text fields and buttons	2 Hrs	Vidya
2	Use Firebase to check validity of new password	1 Hr	Sai
3	Set up appropriate error messages in case of invalid password	2 Hrs (each)	Sai, Vidya
4	Update user account details in Firebase with new password	2 Hrs	Sai
5	Debug and test algorithm with unit testing	2 Hrs (each)	Sai, Vidya

<u>Completed:</u>

The user is first asked to enter their original password which is used to reauthenticate their account. If successful, the program will verify that the two new passwords are matching and if they are acceptable/strong enough, it will change their passwords in Firebase. Additionally, if they attempt to log in again with the old password, they will be greeted with an error.

User Story 7: As a user, I would like to be able to reset my password if needed.

#	Description	Estimated Time	Owner
1	**Create forget password button**	**1 Hr**	**Vidya**

2	Create security questionnaire to confirm identity	4 Hrs	Visa
3	Prompt for new password if the security questions are correctly answered	3 Hr	Sai
4	Change user password in the user database	2 Hr	Sai
5	Debug and test algorithm with unit testing	2 Hrs (each)	Sai, Visa, Vidya

Completed:

If the user forgets their password or wants to reset it, they can do so by clicking the Forget Password button from the home page. This will take them to a page to enter their email to get their account information from the backend. When the Submit button is clicked, the user will be redirected to a page with the security questionnaire that they answered during the registration steps to confirm their identity for security purposes. If the both answers are correct, they will be prompted with a new page to enter the new password they want to change to which will be updated in the account's backend.

User Story 8: As a user, I would like to be able to link my Spotify account to my Wander account.

#	Description	Estimated Time	Owner
1	Create 'Connect to Spotify' button	1 Hr	Vidya
2	Prompt for Spotify login details and permission	3 Hrs	Vidya
3	Connect information from the Spotify API to user profile	4 Hrs	Sai
4	Debug and test algorithm	2 Hrs (each)	Vidya, Sai

Completed:

After the user indicates that they want to authenticate, the user is redirected to a different page where they are asked for their Spotify credentials. Once they are successfully logged in, we are able to retrieve data from them, as well as display a different button for disconnecting from Spotify.

User Story 9: As a user, I would like to be able to remove my Spotify account.

#	Description	Estimated Time	Owner
1	Create remove Spotify connection button	1 Hr	Vidya
2	Remove connection from the Spotify API	3 Hrs	Sai
3	Remove Spotify details from user account details in Firebase database	2 Hrs	Sai
4	Debug and test algorithm	2 Hrs (each)	Sai, Vidya

Completed:

After the user has logged in, if they wish to remove their Spotify account from our website, they are able to click a button which does this for them. In the backend, we are simply throwing away the access token we received back from the Spotify API and deleting any associated music information/song preferences we held previously.

User Story 10: As a user, I would like to be able to take a quiz to determine my music preferences.

#	Description	Estimated Time	Owner
1	Prompt the user with a Take Quiz button to initiate the quiz	2 Hr	Shreya
2	Create UI panel that displays a quiz question and answer choices	3 Hrs	Shreya
3	Implement Next and Back buttons that	4 Hrs	Shreya

	navigate through the questions of the quiz		
4	Create a Finalize Quiz Answers button to indicate the end of the quiz and to save the user's answers to the database	5 Hrs (each)	Shreya, Sai
3	Debug and test algorithm with unit testing	2 Hrs (each)	Shreya, Sai

Completed:

When the user clicks on the Take Quiz button, they are presented with the first question of the quiz. Each time they select an answer, they are taken to the next question of the quiz. If the user would like to change their quiz answer, they can use the Previous button to navigate through the quiz. When the user submits their quiz selections, their music preferences are successfully saved to Firebase.

User Story 11: As a user, I would like to be able to see my current favorite music preferences.

#	Description	Estimated Time	Owner
1	Create UI to see current music preferences	7 Hrs	Vidya
2	Be able to update current preferences in the database	3 Hrs (each)	Sai, Vidya
3	Save music preferences to server/database?	4 Hrs (each)	Shreya, Sai
4	Get music preferences from database and display on UI	4 Hrs (each)	Shreya, Sai
5	Debug and test algorithm with unit testing	2 Hrs (each)	Shreya, Sai, Vidya

Completed:

After the user completes their quiz and their data is saved in the database, they are redirected to a results page where they can view their music preferences. After each quiz submission, the user's preferences are successfully fetched from the Firebase database and populated on the screen. The user is able to take the quiz again and update their music preferences from this screen as well.

User Story 12: As a user, I would like to be able to get a music playlist for my journey.

#	Description	Estimated Time	Owner
1	Create UI to show user's playlist	7 Hr	Vidya
4	Embed as a Spotify playlist	1 Hr	Vidya
5	Find library/tools for embedding a playlist	2 Hr (each)	Shreya, Vidya

Completed:

On the website homepage, the user is able to view a Spotify playlist for their current trip with all the Spotify functionalities still available (e.g. playing music, choosing songs, skipping to the next track, scrubbing through the song, etc). If the user clicks on the playlist button, they are redirected to a separate page that only shows the playlist itself in full screen, also with all the Spotify functionalities still available.

User Story 13: As a user, I would like to be able to see my playlist embedded on the website.

#	Description	Estimated Time	Owner
1	Create UI panel to display embedded playlist	1 Hr	Vidya
2	Save embed code from customized playlist	3 Hrs	Shreya
3	Find React package to create a Spotify embedded page	3 Hrs (each)	Shreya, Vidya
4	Link embedded code with React package	3 Hrs (each)	Shreya, Vidya
5	Debug and test algorithm with unit testing	2 Hrs (each)	Shreya, Vidya

Completed:

We made two views for viewing a Spotify playlist. One on the homepage, and a playlist view that shows the expanded version. Since the playlist is embedded, the user is able to view the singer and song name for each song, and they are able to play the song through our website.

User Story 14: I would like to be able to add a profile picture to my account.

#	Description	Estimated Time	Owner
1	Create a skeleton UI with the appropriate text fields and buttons	1 Hr	Vidya
2	Prompt user for profile picture through file upload	2 Hrs	Vidya
3	Use Firebase to check validity of new profile picture file type	1 Hr	Akshaya
4	Set up appropriate error messages in case of invalid profile picture	1 Hr (each)	Akshaya, Vidya
5	Update user account profile picture in Firebase	2 Hrs	Akshaya
6	Debug and test algorithm	2 Hrs (each)	Vidya, Akshaya

Completed:

If a user is logged in, they are able to add a profile picture to their account. The user sees a blank/default profile picture when logging in. After clicking on the 'Upload Profile Picture' button, the user is able to select a file from their device. The function is automatically set to direct the user to only upload image files, but if the user attempts to upload any other file type, an error message appears. Upon upload, the image gets stored in our Firestore backend and is linked to the user based on their unique user ID. The image URL is also saved as part of the user's profile information in Firebase. The image is then updated on the profile page and header.

User Story 15: I would like to be able to modify my profile picture.

#	Description	Estimated Time	Owner
1	Create a skeleton UI with the appropriate text fields and buttons	2 Hrs	Visa
2	Use Firebase to check validity of new profile picture file type	1 Hr	Akshaya

3	Set up appropriate error messages in case of invalid profile picture	1 Hr (Each)	Akshaya, Visa
4	Update user account profile picture in Firebase	2 Hrs	Akshaya
5	Display new profile picture on profile page	2 Hrs	Visa
6	Debug and test algorithm with unit testing	2 Hrs (each)	Visa, Akshaya

Completed:

The functionality here works similarly to User Story 14. Even if a user has a profile picture linked to the account, the user can still upload a new file. Upon upload, Firestore will replace the image linked to the user's id with the new image. Firebase will also replace the image URL in the user's profile information with the new URL. The image is subsequently updated on the profile page UI and website header.

What Did Not Go Well

We were able to achieve most of our Sprint 1 goals as much of our focus was on creating the general UI of the website and implementing basic user profile features. However, we did not have time to fully implement the algorithm for generating a custom playlist.

User Story 12: As a user, I would like to be able to get a music playlist for my journey.

#	Description	Estimated Time	Owner
2	Create playlist in the server using information from the database and Spotify API	2 Hr	Shreya
3	Store current playlist in the database	3 Hr	Shreya
6	Debug and test algorithm with unit testing	2 Hrs (each)	Shreya, Vidya

Not Completed:

Though we did obtain the user's music preferences through the quiz and their Spotify data, we did not connect that data to a generated playlist. We were able to generate a generic playlist using the Spotify API, but we require more time to refine the algorithm to be catered to the user's specific music preferences.

How We Should Improve

During the sprint, our team operated under the unspoken expectation that everyone would complete their tasks to the best of their ability and finish the tasks on time. Although this expectation was an unwritten rule, it should have been more clearly defined and explicitly discussed as a team. This is a problem that can be easily addressed by taking steps towards improving our communication. We should make more of an active effort towards updating each other on our progress towards certain tasks, making it known early on if any one of us is having trouble with a task, and holding each other accountable towards completing tasks as soon as possible.

Another issue that our group faced was the general lack of organization and time management in the first two weeks of Sprint 1. Though we did make progress on the project during every week of the Sprint, most of our tangible results were developed during Week 3. Our group divided up the project by assigning single tasks to multiple people with the intention of lessening the burden of learning a new tech stack. However, this decision only served to cause more confusion in the long run as we did not know which person had already worked on certain tasks. In future sprints, we will definitely assign more tasks to individuals rather than to multiple people. Towards the end of Week 2, our group decided to alleviate this confusion by creating a detailed Notion page (similar to a spreadsheet) to clearly list out each task to be done. In the future, we plan to create a Notion page much earlier in the Sprint to have clear goals and tasks from the get go.

Additionally, during our sprint, a lot of us would work at the same time, and sometimes on the same files. This caused some merge conflicts that impacted many of our features. This happened because we did not divide the tasks in a proper manner and we weren't aware when the main branch was updated. A simple way to fix this is to make sure everyone is working on their own branch and whenever possible working on unique files that others aren't using. Another way to fix this issue is to communicate with the group if someone is updating the main branch, so everyone has access to the most recent version. It is also beneficial to commit often so if we need to revert, we will lose as little code as possible.

During the next sprint, we could improve on our deliverables more by having a clear understanding of what our requirements specifically are. During the current sprint, there were times when we began implementing features and components of our website based simply on the respective acceptance criterias. As a result, there were points during this sprint during which we were confused about which user story a certain criteria was associated with. To that end, we plan on improving on this during the next sprint by continually referencing our Sprint Planning Documents and Product Backlogs. In addition, we hope to be more serious and consistent about our stand-ups so that the end goals for the sprint and the associated deliverables with these user stories are not forgotten.