Assignment T3: First Iteration

- TeamShiba
  - Jianing Li (jl5543)
  - Yifei Wang (yw3229)
  - Eurey Noguchi (yn2377)
  - Chiqu Li (cl3895)

Github repository

https://github.com/teamshiba/shiba

User stories and Acceptance Testing (Plan & Process)

Voting: As a user, I want to express my preference by a simple action (like swiping, clicking) so that I can quickly vote for all options.

My conditions of satisfaction are:

- On the matching page (where information and images about an option is presented) on a mobile device, a user can swipe right to ‘like’ or left to ‘dislike’ it.

Test case 1:
When a user enters a match room through a link, he can start voting for the options already in the room one after another.
Sample input: one user clicks the link, enters the room, and swipes left. There’re more than one items in the list.
Expected output: the matching page is displayed after he enters the room. After he swipes, the next item is displayed.

Test case 2:
When a user enters a match room where no items have been added to the list, he cannot vote.
Sample input: one user clicks the link and before that, the host didn’t add any items to the list.
Expected output: the page is empty, with only the “add new item” button in the corner, and an image telling the user to click that button to add items.

Test case 3:
When a user enters a match room where no items have been added to the list, he cannot vote.
Invalid input: when a user enters a matching room, he swipe up.
Expected output: nothing happened.

Bugs found & fixed:
We wrote an API for filtering the unvoted items in a group for a user, but in the testing process we found the “unvoted_by” filtering option not working. It turns out that on the server side, the set difference operation is conducted upon a set of item IDs and a set of item objects.

Consensus: As an attendee of an event, I want to know if my group has a consensus (aka. a match) so that I can see which restaurant the group wants to go to.

My conditions of satisfaction are:

- If all members of that group have ‘liked’ the same restaurants in the list, when anyone clicks the link and enters the room again, the agreed restaurants would be displayed.

Case 1:
When all members of a group agree on the same restaurants, the system should present the matching result to every member.
Sample input: 3 users in the group swipe right for the same restaurant. All attendees stay on that page after they completed their own voting process. All of them have voted for the same restaurants (let’s say “Shake Shack”).
Expected output: the same page shows that matching results after the matching ends on every members’ device. The matching results are the intro of “Shake Shack”.

Case 2:
When members of a group cannot agree on their preferences, the system should notify everyone that no match is successful and present the statistical graph to every member.
Sample input: 3 users in the group swipe right for some restaurant. All attendees stay on that page after they completed their own voting process. They voted for the different restaurants (let’s say user 1 for “Shake Shack”, user 2 for “five guys”, user 3 swipe left for both of them).
Expected output: the same page shows that matching results after the matching ends on every members’ device. The matching results are a corresponding graph.

Bugs found & fixed
The bug we found was in the frontend where swiping left was wired to disliking rather than liking. Therefore, when we were testing the correct result of voting was not shown in the statistics page and no consensus was made where in reality there should have been.

Invite friends/family to match: As an organizer of an event, I want to send a link to my group of friends/family so that they can join the group and vote for their preferences in my particular group.

My conditions of satisfaction are:

- I am able to generate a link for my specific group
- Other users can use this link to join my group

- When other attendees join my group, I should be able to see their names to identify who is currently in the group
- After other attendees have joined the group, they should be able to start swiping
- If the person who received the invitation link is not logged in or registered, they should be redirected to a login/registration page and continue the process to join a group after logging in successfully.

Case 1:
After a user successfully creates a group, they should be able to see an invitation link created that they can copy/paste to share to other people that should join the group. After another user receives the invitation link, they should be able to click on it and be redirected to and join the group that invited them. Once the user joins the group, they should be able to immediately start swiping to indicate their preferences on the restaurants.
Sample input: The organizer of the group creates a group
Expected output: There will be a unique link generated and displayed in the group profile page, which other users can use to join the group.

Case 2:
For each invitation link, no duplicate users should be able to join the same group. For example, if a user clicks on the link to join the group and successfully joins the group, then when they click on the same invitation link as the same user, they should not be able to join the group since they are already in the group.
Sample input: The same user access the same invitation link multiple times
Expected output: The user can only join the same group once and therefore, in the members list in the group profile page, we will only observe one instance of this user.

Case 3:
For a person that has not registered for the app, when they open the invitation link, they should be redirected to the login/registration page. After they finish registration and successfully logged in, they can access the link again to join the group.
Sample input: An unauthenticated user tries to join a group
Expected output: The user is unable to join the group and is redirected to the login page. After the user is logged and accesses the URL, they are able to join the group.

Bugs found & fixed
The API we use to retrieve room information like room names, room numbers was restricted to room members at beginning, this causes the user who wants to join the group can't see the group information. We fixed this bug by removing the restriction so that the user can now see the group information before joining the group.

Statistics table: As an attendee of an event, I want to have a table showing how many people chose each option so that I can know about other people's choices and select an option manually when no exact match could be made in the end.

My conditions of satisfaction are:

- A statistics on how many people chose each option will pop up to every member of the group if the group has finished matching.
- The statistic could also be shown when the user clicks the “Statistics” button during matching

Case 1:
Sample Input: All group members finished swiping all the restaurants.
Sample Output: The user will be prompted to click the statistics button to see the results. Restaurants that everyone agrees will be marked out.

Case 2:
Sample Input: During a match with multiple users, one of the users click the “Statistics” button
Sample Output: The user will see the correct information of how many people choose each option correctly.

Case 3:
Sample Input: No user has voted for anything, one of the users click the “Statistics” button
Sample Output: The statistic graph will show all items with counts of 0.

Edit Profile: As a user, I can go to my user profile page and see my information.
My conditions of satisfaction are:

- When I am in the app, I can go to my user page and update my user name.

Case 1:
When a user signs in to the app, he can go to my profile and update my information. Once he updates the info, his info will permanently change.
Sample input: One user goes to the personal file page, updates the user name.
Sample output: The name of the user changed in the user profile page.

Case 2:
When a user doesn’t login, he cannot update his profile.
Invalid input: A user enters the URL for profile edit in browser without login.
Sample output: Redirected to login page.

Bugs found & fixed
The first time we implemented it, we used “set” operation instead of “updating”, this makes the user object in firestore losing all attributes except for the name. We fixed the bug by changing from “set” to “update”

View Room List: As a user, I want to be able to see a list of groups that I am currently a part or was a part of so that I can immediately access the voting page and the voting statistics.

My conditions of satisfaction are:

- I can find all of my joined groups as a list
- I can see groups that have already ended in a separate list

Case 1:
I am able to see rooms that I have created and I have joined in a list.
Sample input: The user clicks on the add group button to create a group
Sample output: The created group appears in the list in the Rooms tab

Case 2:
If the user has not joined nor created any group, then they should not be able to see any groups in the list.
Simple input: The user is new.
Sample output: The room list is empty and the user cannot see any groups

Bugs found & fixed:
We found out that the groups that were marked complete were not seen in the 'History' tab. This bug was fixed in the backend where there was an issue due to the difference between the boolean value true and the string true.

Login: As a user, I can login to the app so that I can create or join a group and start voting.

My conditions of satisfaction are:

- I can successfully register and login with my gmail in the login page and I will be directed to my group lists.
- I can register and login using my email address

Case 1:
Sample input: One user at the sign up page and sign up with Gmail.
Sample output: This user would sign up successfully and would be successfully directed to the group list page.

Case 2:
Sample input: A user enters the wrong email or password
Sample output: The user is unable to login and is asked to try again

Bugs found & fixed:
When we were testing using nginx to integrate service and the web app, the systems kept raising authentication errors. At that time, we are using a field named "id_token" inside the header of each request. It turns out that such a field would be ignored by nginx. Then we named it "Authorization" instead, which comply with the convention.

Logout: As a user, I can logout from the app so that the group info is no longer accessible.
My conditions of satisfaction are:

- I can click the logout button on my profile page and the app is no longer accessible until I logged in again.

Case1:
Sample input: One user at the user profile page and click logout button.
Sample output: This user would sign out successfully and be directed to the sign up page. He can't access shiba content until logged in.

There are no invalid inputs in this case since either user presses the logout button or not.

Creating a Group: As a user (organizer of a group), I want to create a new group where other users can join to vote.

My conditions of satisfaction are:

- I am able to create a new empty group
- I am able to name my group

Case 1:
Sample input: A user clicks on the create group button and inputs a name for the group.
Sample output: A new group is created and is displayed in the room list.

Case 2:
Invalid input: A user clicks on the create group button but tries to create a group without a name
Sample output: The user is unable to create the group and nothing is displayed on the room list.

Automated testing & code analysis

Where are the test cases?

Submit the link to the folder(s) within your github repository containing all the test cases that are automatically invoked by your unit testing tool.

Service: https://github.com/teamshiba/shiba/tree/main/service/tests

Frontend: Tests are located together inside each directory where the tested files are located.

https://github.com/teamshiba/shiba/tree/main/webapp/src

Where are the configurations?

Submit the link to the file(s) in your repository that configure the build tool and/or package manager and the automated unit testing tool.

https://github.com/teamshiba/shiba/blob/main/docker-compose.yml

It also includes configuration from individual Dockerfiles for each service that it runs.

Where are the reports from style checker & bug finder?

the link(s) to the folder(s) within your repository containing the reports from your automated style checker and automated bug finder.

In most cases, there should be at least two reports stored for each tool, at least one that shows errors and at least one that is clean(er).

https://github.com/teamshiba/shiba/tree/main/reports