cluster

Team 15 - Sprint 2 Planning Document

Frederick Li, Shane Li, Shai Mohan, Joseph Singer, David Tong, Andrew Tully

Sprint Overview

Overview

Our first sprint laid out the foundation of our project. For the second sprint, we plan on implementing more functionality for people that have joined clusters. Our frontend team will be making the UI look better and adding functionality. Our backend team will be supplying underlying logic.

If we likened our project to creating a full cake from scratch, we have already completed mixing the batter. This sprint will be allocated for “baking the cake”, which means we will be writing all of the core functionality of the project. Our next sprint will be for “icing the cake” which will add extra features and polish off our final project.

Risks/Challenges

Dependency issues are a potential challenge we might face in this upcoming sprint. Each user story has been split into backend and frontend teams, and many of them depend on each other in some way to complete. Additionally, some new topics are going to be introduced this topic, so another risk we face is finding that we need more time to learn the new topics than we initially anticipated.

Scrum Master: Shane Li
Meeting Schedule: Mondays, Wednesdays, and Fridays from 11:30 a.m. - 12:30 p.m.

Current Sprint Detail

User Story #1

As a cluster member, I would like to be able to have all the capabilities of a logged-in user.

#	Task Description	Hours	Team	Owner
1	Make permissions check for all API calls	2	Back	Andrew
2	Generate different views for cluster members in frontend	7	Front	Joey
3	Send API requests with user permissions	2	Back	Andrew
4	Implement checking user with Spring Security	3	Back	Shane

Acceptance Criteria:

- Given that the frontend can send the API calls and the backend can verify them, when the frontend makes an API call, then the backend can verify that the caller has the appropriate permissions.

User Story #2

As a cluster member, I would like to be able to be able to view restaurant information about my Cluster.

#	Task Description	Hours	Team	Owner
1	Add a restaurant class to backend	2	Back	Shane
2	Provide API calls for sending restaurant information to frontend	2	Back	Andrew
3	Provide database interface for restaurants and link with clusters	4	Back	Fred
4	Implement script to get restaurant information from HungryBoiler	4	Back	Fred
5	Implement API calls to receive restaurant information from the script	2	Back	Andrew
6	Create restaurants page to display all restaurants	8	Front	David
7	Create My Cluster page to display restaurant information	6	Front	Shai

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and restaurant information is returned to the frontend.
- Given that the database interface is set up correctly, when the service layer makes changes to a restaurant, the restaurant information in the database will be updated.
- Given that the frontend is setup correctly, when a user visits the page about their Cluster, the restaurant information will be successfully displayed.

User Story #3

As a cluster member, I would like to be able to be able to view pickup/meeting location for my cluster.

#	Task Description	Hours	Team	Owner
1	Implement API to set/get pickup/meeting location for cluster	2	Back	Andrew
2	Integrate Location class in backend	3	Back	Shane
3	Implement database calls to link location and clusters	8	Back	Fred
4	Implement view in My Cluster and Clusters to show pickup/meeting location	4	Front	Joey
5	Implement API requests on frontend to receive information	3	Front	Shai

Acceptance Criteria

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and location information is returned to the frontend.
- Given that the database interface is set up correctly, when the service layer makes changes to a location, the location information in the database will be updated.
- Given that the frontend is setup correctly, when a user visits the dashboard, their “My Cluster” page, or a specific Cluster page, the location information about the Clusters will be successfully displayed.

User Story #4

As a cluster member, I would like to be able to view information about all other members of my cluster.

#	Task Description	Hours	Team	Owner
1	Implement API call to send member information of a certain cluster	2	Back	Andrew
2	Create view on My Cluster page to show current members	4	Front	David
3	Implement API request to get information in frontend	2	Front	Shai

Acceptance Criteria

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and all member information is returned to the frontend.
- Given that the database interface is set up correctly, when the service layer makes changes , the Cluster's member information in the database will be updated.
- Given that the frontend is setup correctly, when a user visits their "My Cluster" page, all the relevant information about the other members will be displayed.

User Story #5

As a cluster member, I would like to be able to leave ratings for other members of my cluster.

#	Task Description	Hours	Team	Owner
1	Implement API calls to receive ratings from frontend	2	Back	Andrew
2	Integrate ratings class with backend	4	Back	Shane
3	Implement database interface to handle rating requests properly	9	Back	Fred
4	Create views to show ratings throughout the app	5	Front	Joey
5	Implement API request to get information in frontend	4	Front	Shai

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and the rating of a user is updated accordingly.
- Given that the database interface is setup correctly, when the service layer makes changes to a user rating, the user information in the database will be updated.
- Given that the frontend is setup correctly, when a cluster order is successful, then the user will be given an option to rate other users in the cluster.

User Story #6

As a cluster leader, I would like to be able to have all capabilities of a cluster member.

#	Task Description	Hours	Team	Owner
1	Make permissions check for all API calls	4	Back	Andrew
2	Generate different views for cluster leader in My Cluster page	4	Front	David

Acceptance Criteria:

- Given that the backend APIs are implement correctly, when a cluster leader makes a request a regular cluster member makes, then that request is processed successfully.
- Given that the frontend is implemented correctly, when a cluster leader accesses a page, then extra views will appear to allow capabilities only cluster leaders have.

User Story #7

As a cluster leader, I would like to be able to approve/reject users joining cluster.

#	Task Description	Hours	Team	Owner
1	Implement API call to receive requests	2	Back	Andrew
2	Implement logic to properly handle requests	4	Back	Shane
3	Implement API request to get information in frontend	4	Front	Shai
4	Create buttons to allow cluster leader to remove user	3	Front	Joey

Acceptance Criteria:

- Given that the backend is implemented correctly, when the front end makes the API request to approve/reject users, then the user will be properly handled in the backend and removed from the cluster.
- Given that the frontend is implemented correctly, when a cluster leader is on the My Cluster page, then the cluster leader will be given an option to remove certain users from their cluster.

User Story #8

As a cluster leader, I would like to be able to edit information about my cluster.

#	Task Description	Hours	Team	Owner
1	Implement API to allow editing of clusters	2	Back	Andrew
2	Implement logic in service layer	4	Back	Shane
3	Create new page to edit clusters	6	Front	David
4	Implement API request to get information in frontend	4	Front	Shai

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API call, then the information that the cluster leader wishes to be changed will be changed in the backend.
- Given that the database interface is implemented correctly, when the service layer makes changes to the cluster, the database will update the cluster information as well.
- Given that the frontend is implemented correctly, when the cluster leader goes to the My Cluster page, then the cluster leader will have an option to go to another page that allows them to edit cluster information.

User Story #9

As a administrator, I would like to be able to disband clusters.

#	Task Description	Hour s	Team	Owner
1	Implement API requests to disband clusters	2	Back	Andrew
2	Implement logic in service layer to disband clusters	5	Back	Shane
3	Create admin page	6	Front	Joey
4	Add authentication feature in backend to detect whether user in admin	3	Back	Fred
5	Implement API request to send information to backend in frontend	2	Back	Andrew
6	Create buttons to select and disband clusters	4	Front	Joey

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and the cluster is disbanded
- Given that the database interface is setup correctly, when the service layer disbands a cluster the cluster should be removed from the database
- Given that the frontend is setup correctly, when a cluster is removed, it should disappear off the list shown

User Story #10

As a administrator, I would like to be able to view number of clusters active.

#	Task Description	Hours	Team	Owner
1	Implement API call to return the amount of clusters active	2	Back	Andrew
2	Implement logic to serve API with necessary information	3	Back	Shane
3	Create view in admin page to display clusters active	6	Front	David
4	Implement API request to get information in frontend	4	Front	Shai

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and the number of clusters active is increased or decreased
- Given that the database interface is setup correctly, when the service layer makes changes to the number of active clusters, the number of clusters in the database will be updated.
- Given that the frontend is setup correctly, when a cluster number is changed, then the user will be see the updated amount of clusters

User Story #11

As a administrator, I would like to be able to view number of users online.

#	Task Description	Hours	Team	Owner
1	Implement API call to return the amount of users active	3	Back	Andrew
2	Implement logic to serve API with necessary information	3	Back	Shane
3	Create view in admin page to display amount of users active	5	Front	Joey
4	Implement API request to get information in frontend	3	Front	Shai

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and the number of users online is increased or decreased
- Given that the database interface is setup correctly, when the service layer makes changes to the number of online users, the number of users in the database will be updated.
- Given that the frontend is setup correctly, when a user number is changed, then the user will be see the updated amount of users

User Story #12

As a administrator, I would like to be able to view user information.

#	Task Description	Hours	Team	Owner
1	Implement API call to return all user information	3	Back	Fred
2	Implement logic to serve API with necessary information	4	Back	Shane
3	Create view in admin page to display all user information	4	Front	David
4	Implement API request to get information in frontend	2	Front	Shai

Acceptance Criteria:

- Given that the backend is implemented correctly, when the frontend makes an API request, then the request is processed successfully and the user can view user information
- Given that the database interface is setup correctly, when the service layer makes changes to the user information, the user information in the database will be updated.
- Given that the frontend is setup correctly, when a user's information is changed, then the user will be see the updated information

Remaining Backlog

By the conclusion of Sprint 2, we intend to complete 19 of 26 user stories.

As a guest, I would like to be able to:

- ~~view existing clusters.~~
- ~~create an account using Facebook.~~

As a logged-in user, I would like to be able to:

- ~~view my profile.~~
- ~~view and join existing clusters.~~
- sort clusters by newest, distance, or alphabetically by restaurant name.
- view information about a cluster and its members.
- ~~request to start a cluster.~~
  ~~i) specify where I would like to pickup the order.~~
  ~~ii) choose a restaurant to order from.~~
  ~~iii) set minimum cluster size.~~
  ~~iv) set maximum cluster size.~~
  ~~v) set a duration time for the cluster.~~
- ~~log out.~~

As a cluster leader, I would like to be able to:

- ~~have all the capabilities of a cluster member.~~
- ~~approve/reject users joining cluster.~~
- ~~edit information about my cluster.~~

As a cluster member, I would like to be able to:

- ~~have all the capabilities of a logged-in user.~~
- ~~view restaurant information about my cluster.~~
- ~~view pickup/meeting location for my cluster.~~
- ~~view information about all other members of my cluster.~~
- ~~leave ratings for other members of my cluster.~~
- ~~leave a cluster.~~

As a administrator, I would like to be able to:

- ~~disband clusters.~~
- review order history.
- review cluster history.
- ~~view number of clusters active.~~
- ~~view number of users online.~~
- view total number of orders completed.
- ~~view total number of orders not completed.~~
- ~~view user information.~~
- ban users (if time permits).