Team Meeting: ELEN4010 Study Group Coordinator

4 June 2021
Time: 9:50 -12:00
Online: Microsoft Teams

Type of meeting:	Sprint Review and Retrospective
Note taker:	All

Attendees: Tarryn Maggs, Taliya Weinstein, Yasser Karam, Nathan Jones, Basheq Tarifi

Minutes

Agenda item: Sprint Review Session **Presenter:** All

Discussion:

09:50 - 10:00 Scrum Master Introduction: Nathan

Brief overview of the scope covered by the sprint as well as the 12 user stories implemented.

10:00 - 10:10 Tarryn

Completed the following user stories:

- Use the list to navigate between group memberships
- Display list of current group memberships

I created a basic user profile which we can use to navigate between the user's current memberships. The current functionality uses hard-coded membership details and pages, because we have not yet implemented the database/s but the code was designed so that integration with databases will be simpler.

When the profile page loads, the user's membership details can be seen in a table. The details are loaded with the page loading. The process is described as follows: "The client's browser makes a request to the local server to retrieve the information to display in the table. The information is then pushed into the table contents to be displayed."

The table will include a row for each of the memberships. Each row will contain the group's ID and name, as well as the date the user joined the group, the number of users(members) that belong to the group and the number of members that are currently online (in the group's chat). I've included dummy links (non-existent) that will take the user to the specific page when you click on the group name. For proof of concept, I included a button that will change the table of current memberships that is displayed. For future sprints, I've included a dummy button that may be used to allow the user to leave the specified group.

Future improvements and considerations: There is currently no user story that includes the ability to view user details, this should be included and can be implemented by including a call to display the user details (username and password). The whole profile page information needs to be integrated with both databases (user database and group database) so that the information displayed is relevant to the current user. The 'Leave Group' functionality also needs to be included.

10:10 - 10:20 Yasser

Tasks that have been completed, problems encountered and how they were solved, q and a's, what comes next

User Stories Addressed: 'Create study group', and 'Join existing group"

I implemented functionality that allows the user to view, create and join existing groups (hard-coded array). The user currently inputs their username in an 'intermediate' page that leads them to the actual page. In this page, a table is updated with the group created where the user is automatically placed as a member and admin of the group created. Upon creating a group, the user can choose from a (hard-coded) list of members to 'invite'. This automatically adds the

selected users as members to the created group.The table only allows non-members of a group to join groups, and only admin to delete groups (‘delete’ functionality not implemented).

To implement this functionality, the user name has been passed in the URL, to allow the creation and joining of groups to add the current user name as a member/admin. Error checks are involved for the uniqueness of the group name created. End-to-end testing has been implemented using Cypress.

Future considerations: There is no database to allow for persistence and so this shall have to be the basis of the next sprint. The list of users that can be invited, and the existing groups will be loaded from a database. Some styling may be considered in the next sprint (not necessary yet).

10:20 - 10:30 Basheq

As part of the user stories for the sprint, I implemented functionality for non-persistent users to be created and logged in. A user can create an account and sign in using the credentials, which leads to a dummy dashboard page. The user can then log out which returns them to the homepage. Since the basic function was completed early in the sprint, the use of a password with hashing and maintaining a session was also implemented. This was done using the Passport.js module.

A user cannot register if the username is not unique, and incorrect sign in attempts redirect to the login page. Access to the dashboard is dependent on being signed in and access to the login and registration pages is restricted to users who are not signed in. End-to-end testing was performed using Cypress.

This sprint covered the following user stories:

- Create a new account with a username
- Login to an existing account
- Make use of a password in creating and logging into an account (*additional, from future sprint*)
- Be confident that your password is safe via hashing (*additional, from future sprint*)

Future considerations would include additional user details and error messages, as well as integrating persistence in the data by using a database. Another feature would be to display user details after having logged in, and styling pages to be more presentable.

10:30 - 10:40 Nathan

I implemented a basic, non-persistent messaging service using socket.io. Multiple users can enter a group chat area to view and send messages. Basic styling was added using Bootstrap and a short custom css file. I did not make the chat persistent, meaning that new members are unable to see messages from before they joined, and the chat history will be lost between sessions. Given the server-client interactions of this feature, the code is tested almost entirely using End-to-End tests that make use of the Cypress browser environment. However a single function to format the chat messages correctly was tested using Jest.

Difficulties were encountered whilst using Cypress to test multiple users interacting in a single chat since Cypress does not allow for multiple browser sessions. This was solved by simulating another user from within the testing code by opening a port manually and sending messages to the server.

This covers the functionality described by the user story **”Send text messages to a chat area visible to all members”**. Following this, the chat needs to be made persistent using a database so that users can view the chat history after joining the chat between sessions. In addition, events such as votes, meetings and resource sharing need to be integrated with the chat.

10:40 - 10:50 Taliya

I am implementing the functionality to enable a user to select a group from a static list of the groups to which they already belong. Additionally, the user is able to search through the list by typing out which group specifically they would like to be navigated. I also then implemented static pages for each group. Once a user selects a group, they are redirected to the static web page for the group with a dummy heading. End to end testing was also completed using Cypress.io.

These functionalities completed the user stories associated with the **“Search for a Study Group”** covering the product backlog items of the choosing from a static list and searching the static list of group names.

Difficulties encountered during the sprint included being able to setup with the server and route to different pages and being able to add the groups from the static list to the index page when the dropdown menu was selected. These issues were addressed by interfacing with another member who had gotten the server setup working (Nathan) and implementing their recommended changes.

In the next sprint, the logical next user story will be to enable the user to be able to search from the entire list of all groups and leave a group. This will involve the developer story of implementing a database of group names which is linked to a specific user. When the user searches the group database, he should be able to select a new group to join and subsequently leave a group if desired.

10:50 - 11:00 Scrum Master Conclusion: Nathan

All user stories were implemented for this sprint, with a few extra being included as well. Extensions to the backbones of the user stories will be carried out in the next sprint, as well as other required basic functionality.

Action items	**Person responsible**	**Deadline**
✔ Sprint review upload	Nathan Jones	4 June 2021
✔ Sprint planning	Team	4 June 2021

Agenda item: Sprint Retrospective Session **Presenter:** All

Discussion 11:30-12:00:

The integration session took much longer than anticipated and the manner in which the team goes about it should be reconsidered. However, this was the first integration so it may not be the same going forward. A suggestion is for users with similar stories to communicate continuously and integrate their code independently in smaller sessions.

The team should continuously discuss interfaces and models of data being used in order to improve integration and productivity.

Integration of Travis and deploying to Azure also took longer than anticipated since the wrong build service was used. This was fixed using Local Git deployment with Travis and not GitHub Actions.

These issues should be less impactful now that the first release and integration has taken place, which will provide a base for future commits.

10:50-11:00 Sprint Velocity: Nathan

The user stories were only allocated points *after* the first sprint to get a feel for a baseline for the number of story points in each user story going forward.

In the first sprint, 12 user stories were completed, which amounted to 18 user story points. Therefore, using the formula below (see this article), the average sprint velocity after Sprint v1.0 is calculated to be:

$$Avg\ Sprint\ Velocity\ = \frac{story\ points}{number\ of\ sprints} = \frac{18}{1} = 18\ pts/sprint$$