User Stories and Set-up Assignment

Brendan Byers, Ryan Sisco, Iliana J, Aidan Grimshaw, Yufei Zeng
byersbr, siscor, javieri, grimshaa, zengyu

Contents

1 User Stories

1.1 Add a post

As the user adds a post, his post body is checked for any illegal characters or length, where it is then sent to the server and published.

1.1.1 Corresponding Tasks

1.1.2 UML Sequence Diagram/Spike

1.2 Add a comment

When the user is on the post page, they can add a comment where it is checked for any illegal characters or length, and then it is sent to the server and published.

- Response will be tied to the original post
- Responses arent viewable from the main feed
- Response can be voted on/shared/favorited

1.2.1 Corresponding Tasks

1.3 Like/Dislike

When a user votes, their +1 or -1 is sent to the server, as long as they have not voted on that post already, then the server is updated.

- Posts can receive one vote from the user
- The users vote can be changed
- If a post is upvoted:
  - If upvote is clicked the vote is removed
  - If downvote is clicked the vote becomes a downvote

1.4 Report

When a user clicks the report post button, the post is flagged for review. After X amount of reports, the post is automatically taken down while waiting review.

- When a user reports a post, the post wont appear on the feed for that user

1.5 Delete

When a user wants to delete a post, it is checked if it is their own post, if it is, it is deleted. If it is not, it is checked if they are an admin, if they are, it is deleted. The server will then update based on the above.

1.6 Refresh

If the user swipes down on the screen, the page reacts with an animation and then refreshes the page.

1.7 Sort

when clicked, the page alerts the user of the sorting type and then changes the sorting on the next click. They will be able to cycle through all of the different types.

1.8 Add to Favorites

All favorite posts are shown to the user by date added only.

- After a post is added to favorited, the post on the main feed should show that it is currently in the users favorites list

1.9 Offline Favorites

When a user doesnt have an internet connection, they should still be able to see posts that they have favorited.

- Store them in memory
- Posts will show instead of the no internet page

1.10 Offline Capability

When a user is offline, they should be served a page telling them they have no internet. All urls should load.

- Use a service worker to monitor connections and requests
- If theres no connection show a catch all page (exception is favorites page)

1.11 Browser Compatibility

Web app should work in both chrome and safari.

1.12 View newly created Post

When a post is created by a user, when they go back to the feed they should see it immediately if sorted by new.

1.13 Collapse Post Responses

When viewing responses to a post, user should be able to collapse a response, which will collapse that post and all the children.

- Collapsing can be toggled by clicking on the response again

1.14 Secure HTTPS

All communication should be through an ssl connection.

- Need a certificate from https://letsencrypt.org

1.15 Save Website as App on Phone

Website should behave like a pwa and be able to be saved to a phones home screen.

- Add a web app manifest
- Setup unique icons for the screen etc.

1.16 Responsive Web Page

Content should scale with the size of the viewport/screen

- Default viewport size should be defined in the app manifest

2 The Stories due Next Week

Creating the web app to send and receive data into the server is the most important part of the project. We have two groups, one working on frontend (Aidan Grimshaw, Yufei Zeng, Ryan Sisco), and one on backend (Iliana Javier, Brendan Byers). Communication is very important between the two teams, and getting both parts to work together is the most important part.

The user stories that have the most priority, according to the customer, is Add a Post, Add a Comment, and Like/Dislike. These are needed in order to get content onto the server, and get a starting point for the app. The rest of the features should not be implemented until these 3 are completed. The frontend team will develop the javascript and css according to our mockups. The backend team will help handle importing the data and exporting the data. Error handling will also be done according to the mockups.

The secondary tasks will be completed after the connection between the server and user is more grounded and bug-free. Once that is true, development on the tasks will begin. The backend team will handle receiving data from the user, and the frontend team will develop the different pages needed for the tasks. Error handling this data will not be as difficult, because there is no raw user input to check. Security should be easiest on these as well, with little risk for a dangerous bug here.

The estimated time till completion on the frontend is one week, on 3/1/18. This should give lots of time to create a UI that looks very similar to the mockups. The backend estimated time is 3/1/18 for proper communication between user and server. Security will take longer, with no estimated time due to the long process of investigated and auditing of our system. This means that the app will have some functionality and be considered completed while security is still being updated throughout the project. Getting our system working with data is our top priority, and before all of these user stories can be done, this needs to happen.

This will be done with a simple webpage for testing with the database. Security will have the task of checking user text and possible attacks. This beginning code is the most vulnerable to malicious users, so it is important that it works early on. After the communication is established, the teams will be able to break off and continue development on their tasks. By working this way, we believe that our project will work extremely well, and have a complete backend and frontend that can be swapped if one or the other does not work out, or if hosting becomes an issue.

3 Meeting Report

3.1 Progress Made this Week

This week we worked on gathering User Stories and planning their implementation. We prioritized the 16 user stories into three categories: Due this week, due next week, and reach goals. Each story was further partitioned into a list of corresponding tasks to clarify how it will be implemented. To tackle implementation we created Front End and Back End teams and divided the tasks between them. We also divided the user story UML Sequence diagram creation equally among the team members.

3.2 Plans for Next Week

Next week we will begin to code the highest priority stories, such as Add a Post, Add a Comment, and Like/Dislike. We will be using the tools and coding style we have documented in the past weeks to do so. Communication between the Front and Back End teams will be emphasized to ensure all team members are on the same page and code implementation can be as efficient as possible.

3.3 Team Member Contributions

- User Stories - Brendan Byers, Ryan Sisco, Iliana Javier Aiden Grimshaw
- Corresponding Tasks - Brendan Byers
- Stories Due Next Week - Ryan Sisco
- UML Sequence Diagrams - Ryan Sisco, Aiden Grimshaw, Yufei Zeng, Iliana Javier, Brendan Byers
- Meeting Report - Iliana Javier