Sprint Two Retrospective

March 28, 2018

Team: 29

Anjali Malik, Shengqi Wang, Kenan Dominic, Arnav Jain, Sanjit Sama

1. What went well?

We feel that for Sprint 3, our biggest strength was that we were able to incorporate what we learnt in Sprint 1 from our mistakes, and performed at higher standards with clear objectives and expectations. Unlike Sprint 1, we were also able to satisfactorily complete almost all of the user stories with an exception of 1 which was partially incomplete/defective.
Importantly, we were able to reflect on the feedback that was given to us after Sprint 1 review and make needed changes in how we designed, coded, and presented our product.

Additionally, we also made better efforts in communicating with our project coordinator, as well as, conducted regular interactions and meetings with her including a practice demo before our final review. We strongly feel that those efforts were well paid off during our presentation as we were properly aware of the requirements and expectations.

The following are completed tasks that should be highlighted:

- **User Story # 1 :** *I would like to be able to sign out of my account.*

Tasks	Description
1	Implement UI element for user to click on for signing out.
2	Implement server endpoint to remove authIssued and authToken from the database to prevent user from accessing the website without logging back in
3	Implement code to remove the auth token
4	Manual testing for logging out endpoint using POSTMAN, as well as through client interaction

All done

- **User Story # 2 :** *I'd like to be able to change my password.*

Tasks	Description
1	Add UI elements that allow user to create a new password
2	Implement a check in server that will check if the current password provided matches with the old password stored in database.
3	Implement code in JS to check if the current password is not the same as the new password.
3	Implement JS functionality to confirm the two new password boxes are the same
4	Implement endpoint in the server to accept the new passwords, hash them in AES 192 and store the new values in the database
5	Manual testing of the endpoint using POSTMAN. Test through client by changing password and logging again with old and new password.

All done

- **User Story # 4 :** *I would like to be able to scroll through a feed that shows me posts of "Offers" and "Requests"*

Tasks	Description
1	Implement front end to create cards which extract data given by user and display in an elegant manner
2	Implement JS functionality which gets data for posts from database refreshes upon every user interaction
3	Create endpoint in the server to retrieve posts from the database
4	Test endpoint to get all posts using POSTMAN. Test through the client by adding or deleting post, and refreshing page to see all the posts in database are displayed on the page

All done

- **User Story # 5 :** *I would like to be able to make post for "Offer" and "Requests"*

Tasks	Description
1	Create a page where users can write posts and specify different attributes about them including if they are “offers” or “requests”, and press a button for publishing.
2	Implement JS code that can accept user input and send that to the server along with the information about the user who wrote it
3	Implement endpoints in server that can add posts to the database along with their details
4	Test the endpoints in POSTMAN. Test the overall working of the functionality by creating multiple posts of different categories and ensure that their look is consistent with the design

All done

- **User Story # 6 :** *I would like to be able to specify attributes to the “Offer” or “Request” I post.*

Tasks	Description
1	Implement frontend code for creating a post such that it expands upon the category selected
2	Front end cards should display relevant information regarding the category selected
3	Implement JS logic to accept user inputs and selections and send the information to the server
4	Implement endpoints in server to add attributes information to the database ta able
5	Test using client side interactions such as making new post to see that the information is stored in database and displayed on the home page

All done

- **User Story # 7 :** *I would like to be able to sort services on my feed based on the cost of the service.*

Tasks	Description
1	Create a UI element that can specify the price range (slider)
2	Create a UI element that can specify the order in which the posts should be displayed in newsfeed.
2	Implement JS functionality to pass in the order, or lower, and upper bounds to the server
3	Implement server endpoint to sort the database based on the price variable
4	Implement JS to get the sorted list and refresh page with only items in given price range
5	Test endpoints using POSTMAN. Test through client by adding new posts with different amounts of cost specified, and then giving varied price ranges as well as order.

All done

- **User Story # 8 :** *I would like to be able to sort services on my feed based on their posting dates.*

Tasks	Description
1	Create a UI element that can specify a posting date range for relevant posts
2	Create a UI element that can specify the order of date posted in which the posts should be displayed in newsfeed.
2	Implement JS functionality to pass in the user request/input to the server
3	Implement server endpoint to sort the database based on the posting date values, or order
4	Implement JS to get the sorted list and refresh page with only desired items
5	Test endpoints using POSTMAN. Test through client by giving varied date ranges or order.

All done

- **User Story # 9 :** *I would like to filter posts using their categories or type of post.*

Tasks	Description
1	Create a UI element that can choose a category from all categories options
2	Create UI element that lets users choose to filter requests or offers (type)
2	Implement JS functionality to pass in the user category selection or type selection to the server
3	Implement server endpoint to filter the database table rows using the user selection
4	Implement JS to get the filtered items and refresh page with only those items
5	Test endpoints using POSTMAN. Test through client by clicking for filtering posts based on all types and categories

All done

- **User Story # 10 :** *I would like to be able to click interested on a post.*

Tasks	Description
1	Create a UI element for clicking interested on every post card
2	Create endpoints for adding 'likes' to a post and create corresponding notification for the owner of the post
3	Implement functionality in JS that whenever a user click on the interested button, the server is sent that information along with the user who clicked it
4	Implement functionality in JS that whenever a user clicks on the interested button, they should see the change in real time without the need to refresh the page
4	Test the endpoint in POSTMAN. Test the overall working of the functionality by creating clicking multiple posts and checking the change in the DB and on the webpage.

All done

- **User Story # 11 :** *I would like to be able to add comments on the discussion board of any open request and offer.*

Tasks	Description
1	Create a UI element for discussion board within frontend code for posts
2	Create endpoints for adding comments to a post and storing users who write comments on posts
3	Update database design to include comments as well as a field to store the user who posted each comment
4	Create endpoint for retrieving comments for a post
5	Implement logic in JS to accept user input for a comment and send that to server
6	Implement JS logic to retrieve comments whenever a post's comment section is viewed
7	Test the endpoints using POSTMAN and test through the client by adding new posts and viewing the changes on the comments section.

All done

- **User Story # 12 :** *I would like to receive a notification if someone interacts with my "Offer" or "Request"*

Tasks	Description
1	Create a UI element that can show their notifications to the user.
2	Implement JS logic to see if other users have clicked "interested" or commented on the post
3	Create endpoint to get notifications for the specific user who is signed in
4	Create function in server to create a new notification every time a user's post is liked or commented on
5	Get the notifications from the server for a user when they click to see their notifications
6	Test endpoint using POSTMAN. Test in real time by liking user post and commenting, to see new notifications displayed in the notifications list

All done

- **User Story # 13 :** *I would like to be able to report a post.*

Tasks	Description
1	Create a UI element that lets the user report a specific post.
2	Implement JS functionality to pass the post ID to the server.
3	Implement endpoint in server for reporting posts that accepts the Post ID, and uses Post ID to store the ID of the user who owns the given post.
4	Initialize the 'REPORTS' table and store all reports in the 'REPORTS' table to be reviewed later.
5	Test the user story by sending reports to the database through client, POSTMAN and checking for its storage in the table

All done

- **User Story # 14 :** *I would like to be able to edit/close my old post.*

Tasks	Description
1	Create a page for where users can see all their active posts.
2	Create UI elements that indicate actions to be performed on a post such as edit/ delete
3	Implement client-side JS to handle either edit or a delete action performed on a post
4	Implement endpoint in server to accept the new attributes for a post in the case of the edit action being performed or to remove the post from the 'POSTS' table in the case of a delete action
5	Test using POSTMAN and through client by deleting or editing a post. Check both UI and database to see the changes

All done

2. What did not go well?

A struggle for this sprint was to determine the time needed to complete each user story in addition to saving buffer time for fixing bugs and adding corner cases to our code. From Sprint 1, our user stories for Sprint 2 were quite work intensive and needed a lot of time for thinking about approaches, designing, and fixing errors on top of actually writing code. One of the issues that this caused, was that we haphazardly dwelled into coding without properly evaluating project feasibility and the design choices that were needed to be made. We did not properly assess our possible approaches, therefore, often ended up needing to change the way we were thinking about the problem. This costed us some time as well as efforts which could have been avoided had we laid out our design plan before moving into the stage for writing code.

Another thing that did not go well was regarding our User story #3 for uploading profile picture. We had assumed that we would use S3 before properly assessing the technology, and other tools that we could use for the problem. We did not seek outside help from CS 307 TAs, which is something we should have done. We realized this during the demo when we were advised on better or easier ways that we could have employed for finishing this user story.

- **User Story # 3 :** *I would like to be able to upload and remove my profile picture*.
    - Although we were able to successfully set up an S3 bucket in AWS, implement endpoint for uploading pictures to the bucket, and write code in JS for selecting and showing an image on the profile page, we failed in writing correct code for communicating with the server to send a POST request. We were not able to send 'form-data' content-type to the server which caused the failure in communication from client to server for this specific feature.

Tasks	Description
1	Add a UI element that allows for user to upload the picture
2	Implement JS functionality to pass the image when done uploading to the Amazon S3 signing endpoint
3	Pass the signed URL to the server's s3sign endpoint to retrieve URL of picture stored in the S3 bucket

3. How should you improve?

Although we felt that we succeeded in multiple ways in this sprint, we still believe there are a number of ways that we can improve, going into Sprint 3. Firstly, we should review our product more often to ensure that the user interface and features as presented for our product are satisfactory. For example, we will be spending more time this sprint on making sure that our UI is supported on different resolutions and screen sizes, a defect that we discovered with our application during Sprint 2.

Further, for this sprint, we need to focus on delegating some amount of time every week on collaboration between different components of the product. Considering the amount of integration between components needed for our project, we must account for a session every week for bugs, defects resolution and cross pollination between multiple files. Given the fair size of our code, we also need to expect to spend some time every week for finding errors and corner cases.