Summary of new user stories, new code/functionality and tests for milestone 2

Note: screenshots of product backlog, sprint backlog and testing our found in other documents in the milestone 2 zip 😊

Now for user stories, we didn’t find it necessary to create any new ones. However, we did rearrange order of our stories and priorities in terms of completing the project. This is evident through Jira where you can see scope changes of the sprint backlog and the product backlog shown. As we finished sprint 1, we realized that some of our user stories needed to be swapped on the list so we can ensure a sprint with the most effective work completed.

New code implementation/Functionality:

In this website, we are allowing three types of users. One is a customer, another one is a publisher account and the third one is an admin. Except admin, customer and publisher needs to open an account through the registration page. We are implementing two microservices for this Milestone 2. Both the different kinds of users are able to login securely throw jwt authentication and are redirected to the “Dashboard” function. For Milestone 2, the dashboard is pretty simple. We have extensively used conditional rendering in order to display different kinds of functions and services for different users. As of now, all 3 users can see the list of books in our store. In order to store images of the books, they are stored in a s3 bucket. Admin and publisher can both add books in the website and view them. Deleting the books requires the admin and the publisher to type in the book id number and them simply press delete. Admins first approve a publisher in the website before he/she can decide to start doing business.

New Tests implemented:

We were able to fully test all parts of the backend and frontend for all implemented features.

We added unit tests for all the microservices we have implemented to test the backend functionality.

Further testing was done using POSTMAN to make sure all post mapping, get mapping, delete mapping and update mapping functions work correctly

We also did extensive testing on the front end, constantly cross referencing our database to ensure all of the features we implemented interact correctly with our backend