A central media hub for all the things *you* love

CS 40700 - Senior Design Project

SPRINT 3 - RETROSPECTIVE

Team 6

Utkarsh Agarwal *uagarwal@purdue.edu*
Shivangi Chand *chands@purdue.edu*
Amol Moses Jha *jha8@purdue.edu*
Pooja Tewari *tewarip@purdue.edu*

WHAT WENT WELL?

The 3rd and last sprint of the class went the best for us as a team. We were quite successful in completing all the user stories again along with a “If time permits” user story. Our main goals of this sprint on the application end included finishing the ‘Dashboard’ page and the ‘Trending’ page along with allowing the user to search with three more categories (Events, Written/Books, and Audio). A couple more important factors for us this sprint were to make automated testing as well as a secure application, which were both successes as well.

In order to produce an app according to industry standards, it was really important for us as a team to be able to build a secure application by the end of the project. According to our plan, we were able to make the application secure with https. It was the first task that our team strategized on completing and it was the user story that we made it our goal to complete first. We were able to meet these standards as soon as a week into our sprint.

As a team, we were able to establish a good culture for our code reviews. A minimum of two members of our team was always present during a code review and with a healthy amount of discussion as a team, we were able to improve our coding standards as the style of coding began to align for the Frontend and Backend teams. The code quality also improved because both the teams were able to see each other’s code and any code repetition was avoided by making reusable components at various points in the code. Along with this, coding also became more efficient over time as planning was done together by the teams on how to tackle their individual tasks.

Lastly, as we specified in the previous retrospective, we had problems with setting internal hard deadlines last Sprint. However, in this sprint, we were able to adhere to all the deadlines set by the team during the sprint planning. This helped us with staying on track with our development process and finish our Sprint timely.

In all, it was a successful sprint and we were able to achieve all the goals we had set for ourselves.

User Story #26

As a user, I should be able to connect accounts of online audio content providers to the application.

Completed: This story was also completed on the backend. Upon selecting a search result, the user is able to go to say Last FM, for example, and use his/her account to watch the content that was recommended by Mine.

User Story #27

As a user, I should be able to get search results for audio content.

Completed: The UI with the list items being the search results and the categories being unordered list was used. On the backend, the APIs for Last FM was used to obtain the results and sent to the frontend to be displayed. The story was unit tested well.

User Story #28

As a user, I should be able to connect accounts of written content providers to the website.

Completed: This story was tied in with User Story #26 and implemented in the same fashion, where a result can be clicked on and it would take you to the appropriate website for the content. It was unit tested well.

User Story #29

As a user, I should be able to get search results for written content.

Completed: This user story was tied in with the user story #27. The UI with the list items being the search results and the categories being unordered list was used. On the backend, the API for Last FM was used to obtain the results and sent to the frontend to be displayed. The story was unit tested well.

User Story #30

As a user, I should be able to connect accounts of events data providers to the website.

Completed: This story was tied in with User Story #26, 28 and implemented in the same fashion, where a result can be clicked on and it would take you to the appropriate website for the content. It was unit tested well.

User Story #31

As a user, I should be able to get search results for events near me.

Completed: This user story was tied in with the user story #27, 29. The UI with the list items being the search results and the categories being unordered list was used. On the backend, the API for SeatGeek was used to obtain the results and sent to the frontend to be displayed. The story was unit tested well.

User Story #32

As a user, I should be able to browse my previous search terms.

Completed: This user story was one which had to be paid special attention to on the frontend aspect. The UI for this was created by using a reusable component that displayed the previous search terms associated with each category as part of a button on a list. The buttons can be clicked on which takes the individual search term and redirects the user to the search page with the search results of that term and the category associated with it. The frontend was definitely challenging for this, however, with enough brainstorming and code reviews, we were able to successfully create a backend as per our satisfaction. On the backend, the team keeps track of the previously searched terms and then sends a list of the terms associated with the categories of preferences of the user to the frontend. This user story was well tested on the backend and the frontend.

User Story #33

As a user, I should be able to see other trending searches.

Completed: This user story was very similar in terms of the UI to user story #32. The UI for this was created by using a reusable component that displayed the trending terms associated with each category as part of a button on a list. The buttons can be clicked on which takes the individual search term and redirects the user to the search page with the search results of that term and the category associated with it. The frontend was able to reuse the same reusable component from user story #32, which made it rather easy to complete. On the backend, the team keeps track of the most often searched terms app-wide and then sends a list of the terms associated with the categories of preferences of the user to the frontend. Again, this was similar to user story #32. This user story was well tested on the backend and the frontend.

User Story #34

As a user, I should be able to see my most frequent searches.

Completed: This user story was very similar in terms of the UI to user story #32, 33. The frontend was able to reuse the same reusable component from user story #32, 33 which made it easy to complete on the UI end. On the backend, the team keeps track of the most often searched terms by the user and then sends a list of the terms associated with the categories of preferences of the user to the frontend. Again, this was similar to user story #32, 33. This user story was well tested on the backend and the frontend.

User Story #35

As a user, I would like to connect to developers in order to provide meaningful feedback.

Completed: This user story was important because it allows the user to give feedback to the developers and through this feedback, we get an opportunity to improve the application as per the user's needs. The user story's UI was created on the 'Feedback' page, which can be accessed via the 'Accounts' page. The user can just type his/her feedback in a textbox and send it to the developers. The story was unit tested well.

User Story #36

As a developer, I would want the project to be very well tested, with great coverage ensured by a healthy mix of unit, integration, regression and functional tests.

Completed: This user story was important because a good app needs to be well tested. We had ensured testing is there throughout all the sprints. We successfully completed this user story by also adding automated testing on the frontend and finishing all the automated test cases on the backend.

User Story #37

As a user, I would like to use a secure application using https.

Completed: This user story added another layer of security to our application by making both our frontend and backend accessible only using HTTPS. This allows save exchange of information between the two servers. We were able to get this completed and deployed with industry standards in mind.

User Story #38

If time permits, as a user, I would like the user to select the APIs he wants to search from.

Completed: We work with a lot of third-party APIs in this project and we understand that some of these API providers/companies would not be relevant to a particular user. This user story allows users to choose from their sources from a list of our API integrations. We were able to complete this in a fashion that any future additions would automatically be added.

WHAT DID NOT GO WELL?

Even though, we were able to successfully complete our user stories on time, something that did not go well was that we had planned to complete more stories in the first week of the sprint as our team members were busier during the following weeks. It is admittedly hard to schedule times for group work when all of us have widely different schedules. We planned to integrate as we developed but sometimes that wasn't possible to do. It is hard to immediately get integration done - which caused delays when some team members had to wait until necessary prerequisites were being worked on by different team members.

Furthermore, during our sprint review, one of our team members was missing. Every member worked on this sprint equally and tirelessly. However, due to health conditions, all of our team couldn't make it to the review. Given that it was dead week and everyone has a lot of submissions, it is understood that someone can get overworked and exhausted but nonetheless, this is something we would like to avoid.

Finally, unit testing is a concern too. For the frontend, we initially planned to use a React recommended framework such as Jest and Enzyme to create tests, however, the team used Ghost Inspector instead. Ghost Inspector works well and fulfills the requirements, however, using Jest and Enzyme, which are widely used could be better choices. Backend could also benefit from a greater number of test cases, covering more scenarios leading to better code coverage and a more robust application.

HOW TO IMPROVE?

One of the things that we always feel has a chance to improve would be the team efficiency in working together. We plan to aim towards more flexibly scheduled group meetings allowing smaller groups so that group members can have smaller, more focused meetings with the exact people they need to work with in order to get work done. This would allow us to have greater success in scheduling meetings as opposed to trying to find a common time for the entire team at large. Moreover, a targeted approach like this would help us in integrating functionalities faster as well.

Though not a recurring problem, team members missing in the sprint review definitely is an issue we would want to improve. The underlying issue is that during more stressful weeks, sometimes people can be overworked and exhausted. We would like to devise better strategies so that we foresee such weeks and allot work accordingly. This would help us avoid any surprises and be better prepared for sprints.

Testing finally, we firmly believe is a place where we can always improve on. We would like to focus on greater automated testing abilities on the frontend with tooling like Jest and Enzyme, and on the backend to support our growing suite of automated unit tests with more functional, integration and regression testing. Spending more time and care on the testing effort would definitely help us achieve this, and ensure a highly robust and functioning application.