Andrew Brown (ajb5384)

Bibartan Jha (bj8355)

Morgan Murrell (mmm6855)

Alina Nguyen (amn2763)

Team E9: Phase 2 Reflection

What did your team learn?

During this phase, we learned how to add paginations to all of our model pages. Before pagination, we had all of our instances on a single model page. Now that we have pagination, our model pages are much more user-friendly and more interactive. We also learned to add features that allowed users to sort and filter instances based on different parameters. Since our website is basketball-related, some of our parameters are current status, position, conference, team, name, and location. This is another way to make our website more user-friendly and more interactive. In this phase, we also learned how to connect our MongoDB through PyMongo without having any issues deploying to Heroku. In Phase I, we had already figured out how to take information from our API's and store it into our MongoDB database for the website. However, in order for MongoDB to work when we deployed to Heroku, we had to allow network access for all IP addresses for our database. While this might be an information security vulnerability, not allowing network access to all IP addresses was the biggest issue for us when we deployed to Heroku. We also learned how to connect to Firebase using Pyrebase. Using Pyrebase was extremely helpful for implementing login/logout features and creating accounts on our website. We also learned how to create frontend and backend tests for our website. We learned how to use different testing tools (such as unittest, Mocha, and Selenium) to test the functionality of our database and our website applications.

Five things you struggled with and need to improve?

A number of our issues during this phase revolved around users interacting with our website if they were logged in to their account that they had created on our website. We struggled with adding favorite players and teams to each user's specific document in our MongoDB database. We also struggled with adding a Favorites button to each instance page that would add information to our MongoDB database when clicked. Our initial idea was to have the page stay the same after the button was pressed. This led to bugs and issues on our website. We were able to resolve the issue by having the user be sent to a confirmation page once he/she pressed the button. We also struggled with creating user authentication with Firebase and JavaScript. To resolve this issue, we switched to Pyrebase. We struggled with syncing pagination with Flask. We resolved this framework by switching to Django framework rather than continuing to use Flask. Our last big issue was creating frontend tests with Mocha. We were not able to check CSS attributes with Mocha so our workaround was to use Selenium for our frontend and GUI tests for our website.

Five things that just worked well?

We had quite a few things work well for us during Phase 2. We were able to add social media links for our teams and players in our MongoDB database very easily. We added more multi-media to our instance pages without much difficulty. We now have a login/logout button based on whether a user is logged in to our website or not. We also can add users to our MongoDB database if someone creates a new account along with updating their specific document when they add a team/player to their favorites list. Implementing MongoDB has been our biggest success in this phase. We were able to add/modify/delete elements in our MongoDB database very easily.

Testing:

Backend: For the backend testing, we used the unittest module. Most of the unit testing was testing the functionality of our MongoDB database. To test our MongoDB database, we would create mockup data and perform CRUD (create, read, update, delete) operations for each database collection to see if the database will work as intended (i.e. does the database store the mockup data in a specific collection). In order to avoid actually writing to our MongoDB database, we would store all of the elements in a collection into an array and then perform the testing. We would check our results using the ASSERT(...) functions to check if the testing performed the way we intended.

GUI: For the GUI testing, we used the Selenium Webdriver tool. The primary goal for our GUI testing was to check the navigation bar, scrolling ability, pagination, log-in/out capabilities, and randomness of the games listed on the front page. For the navigation bar, the Selenium Webdriver clicked through each navigation link and compared the current URL to the expected URL. For the scrolling ability, the webdriver scrolled slowly through the team/players page. For pagination, the webdriver clicked through a few pages on the team/players page and compared the current URL to the expected URL for each new page. For the log-in/out capabilities, the webdriver used ‘seleniumtest@gmail.com’ and ‘password’ to log into an existing account and then log out. The current URL was used to check if it performed the correct actions. Lastly, the front page showcases random NBA games from the 2019 season. This was tested by using the webdriver to compare the first game listed before and after refreshing the page. The results for this test were printed to ‘selenium_test.txt’ for easy-viewing.

Andrew Brown (ajb5384)

Bibartan Jha (bj8355)

Morgan Murrell (mmm6855)

Alina Nguyen (amn2763)

Frontend: For frontend testing, we used Selenium instead of Mocha. We were having issues with Mocha, so we decided to switch to Selenium Webdriver to test our frontend. We believe that using Selenium instead of Mocha would not make much of a difference in terms of the quality of our frontend testing. The main goal for the frontend tests was to have our webdriver test the CSS attributes of various elements on our website. These elements include: the log-in button, homepage table, players model item, players model table, players instance page, and log-in page inputs. To test these elements, we located the elements on our deployed website and compared the received CSS attributes to our expected values. The results for this test were printed to 'selenium_test.txt' for easy-viewing.

Team Github Repo: https://github.com/UT-SWLab/TeamE9

Website Link: https://teame9.herokuapp.com/

Team Info

Group Name: E9

Members: Andrew Brown, Bibartan Jha, Morgan Murrell, Alina Nguyen