CSCE 606 Software Engineering

Team Spongebob: Iteration 2 Project Report

Spring 2021

## **Customer Feedback**

Team Spongebob met with the client on **Friday, April 23rd, 2021 at 5 PM (CST)** on Zoom. We changed our recurring customer meetings from Monday to give Jason and Farnoosh the opportunity to attend. During this meeting we showed the clients the updated app view including the updated home page, about page, and logo, and then briefly discussed the corrections to the backend database. We then prioritized the correction of the header logo, the writing of instructions for teachers during upload, and the providing of a CSV template file to improve the accuracy of a teacher’s uploaded for our next iteration. The clients remarked that the project was coming along well.

## **Major Code Refactoring**

After several days of troubleshooting the legacy code our team decided that further efforts to fix the issues was a poor choice of action. For the following reasons, our team decided to develop a new application:

* Controllers, views, URL paths, and database models were each inspired by different source codes which resulted in limited functionality and code conflicts.
* The current user sign-up and corresponding Authentication form was just a view and did not store any information into the database.
* Due to the existing user sign-up structure, there were no validations in place and users could not be related to any database model.
* The original purposes assigned to each Django app, or controllers, were convoluted and unfocused, which was resulting in complicated controllers, views, and models.
* The code between the database and forms was inconsistent.
* The admin page was unexpectedly working indepently.
* Django documentation gives suggestions in encouraging or discouraging developers to use the best methods. The legacy code was not following Django’s outlined best practices, especially in the Authentication form section.

We created a new application that divides the application based upon functionality, creating a better internal structure. We started from scratch by creating a new models file with a structured database, inheriting from some Django classes. We re-designed the forms, which are now compatible with the database. We were able to then create a functioning database complete with model relations such as teachers to files, a one -to-one relationship.

We have implemented a plethora of new views including the following: profile settings, account view, admin database view, and admin database manipulation. With the help of Django’s allauth package, we added a new authentication system that provided us with an easy way to utilize many account administration features, such as password reset via email. With this, every user has an account page that gives extra information and features to add or edit their information. In addition, all of these features are now accessible from the admin panel. An admin has the ability to sort users and uploaded data as well as perform user administration activities, such as deactivating accounts.

# **User Stories**

## **Completed**

**First User Story - Website Design**

* Feature: Home and About Page Styling & Content
  + As a stakeholder
  + I want the Home and About page to be styled and contain our information
  + So that it will represent and explain the project

The Horizontal ELRC logo was added to the navbar of the website. Also, we added a school background to the home page and about page. HTML with Bootstrap styling was added to both the home and about pages to create a box for text. Since we do not have the home and about pages content yet, the boxes were filled with dummy text for the customer demo to give them a better understanding of what the final product might look like on the page.

**Second User Story - Improved Authentication and Account Experience**

* Feature: User Account Password Recovery
  + As a user
  + I want to be able to reset my password on the login page
  + So I can access my account if I forget my password
* Feature: User Account Settings Page
  + As a user
  + I want to be able to view my account information on a settings page
  + So I can see and change my information
* Feature: User has profile and uploaded files associated with it
  + As a user
  + I want to have a unique page created with my account
  + So that my uploaded files are saved in the database under my name

As seen above, we initially planned to implement each of these features separately. However, when we did the major refactoring of our application, we transitioned to use Django’s allauth package instead of the custom, unreliable existing code. This package provides us with all account and authentication related views, models, and other code, and satisfies the above stories.

## **Changed**

**Iteration 1 First User Story**

* Feature: Log in to website
  + As a teacher
  + I want to log in
  + So that I can access the app under my profile

The implementation code for this story changed. As previously discussed, we transitioned our application to use Django’s allauth package rather than a custom implementation.

## **Incomplete**

* Feature: A&M Branding
  + As a stakeholder
  + I want the webpage to use [Texas A&M branding](https://brandguide.tamu.edu/web/web-color-palette.html)
  + So that it aligns with Texas A&M

This feature was reprioritized by the team due to the need to develop a new application, and reimplement everything including our existing styling, URL paths, views, and more.

* Feature: CSV Template for User Downloads
  + As a teacher
  + I want to be able to download a CSV template
  + So that I can format my data for the AI prediction

This feature was reprioritized by the product owner due to the need to develop a new application, and reimplement everything including styling, URL paths, views, and more.

* Feature: CSV Template for User Downloads
  + As a teacher
  + I want my uploaded CSVs to be analyzed by the legacy python code
  + So that I can receive a better prediction for my students test scores

This feature was also reprioritized by the product owner due to the need to develop a new application, and reimplement everything including styling, URL paths, views, and etc.

## **Testing**

**Changes to Testing Methods:**

While doing increasingly more work on trying to implement the Feature tests, we decided it would be best to scrap creating step definitions corresponding to our scenarios for our acceptance tests. This is because the existing modules for Django, behave and django-behave, have poor and outdated documentation. Trying to make these testing modules work was too time consuming and was diverting attention away from making further progress on the application.

Now, we are using Django’s much simpler TestCase module. Each Python file in the application that contains custom code has a corresponding test file with one or more TestCase classes inside. Our scenarios from our feature files provided guidance for creating the testing for the views, which is how we now implement our acceptance tests. Like what is tested in Cucumber step definitions, our view tests make sure the user is seeing the correct thing on the page when they perform some action. We wrote unit tests for the models, which test the specific functionality of that individual component in the application. We added some additional tests cases for the app.py configuration files. Although there is no code in these, the tests prevent us from accidentally changing the file and messing up our application.

**Running Our Testing:**

Our tests can be run by using the command **python manage.py test tests --verbosity=2**. In addition, we created a workflow for Github’s CI functionality that automatically runs our tests and generates a coverage report when a pull request is created. The configuration file can be found [here](https://github.com/hpatrick5/sp21-csce606-group_project/blob/main/.github/workflows/test_ci.yml). This ensures that tests are always being run and that no features are being broken by the new code.

## Project Software Aids Links

* GitHub repository: <https://github.com/hpatrick5/sp21-csce606-group_project>
* Pivotal Tracker: <https://www.pivotaltracker.com/n/projects/2489771>
* Heroku Deployment: [https://sp21-606-school-district-data.herokuapp.com/](https://sp21-606-school-district-data.herokuapp.com/login/)
* Code Analysis: <https://deepsource.io/gh/hpatrick5/sp21-csce606-group_project/>
* UML Design Diagram: <https://www.gituml.com/viewz/523>
* Test Output: [Here](https://github.com/hpatrick5/sp21-csce606-group_project/blob/3e3b7bc3b71c387e910c6c89a86c1f0c027b1d6e/test_output.txt), and also included in the tar file
* Test Coverage: [Here](https://github.com/hpatrick5/sp21-csce606-group_project/blob/3e3b7bc3b71c387e910c6c89a86c1f0c027b1d6e/test_coverage.html), and also included in the tar file