Fast Pass
Online Textbook

Chia-Yuan Chang, Eric Yeh,
Kai-Chih Huang, Li-Yuan Hsu,
Po-Han Hou, Xian-Ji Chen

1. Summary of implemetation

Our project is a textbook website. It is used to sell online calculus and finance textbooks. Our stakeholder is Professor Philip Yasskin who is also our customer for this project. His main idea is to use this new website to replace the old one which is currently running. The new website was built by a former student and he designed the framework of the website. Professor Yasskin has multiple thoughts about the website which became our user stories. The demands were:

- ➔ *The login function and it needs to include google login.*
- ➔ *To let readers read the textbook and manage their subscription status.*
- ➔ *To distinguish who can get access to the textbook. There are 3 categories of people can read the book:*
  - ◆ *People who paid for the book*
  - ◆ *Students of Texas A&M and University of South Carolina*
  - ◆ *People who receive a promo code from the book owner*
- ➔ *Let customers buy textbooks on the MyMath website.*
- ➔ *Customers could be able to checkout using PayPal.*
- ➔ *Customers could be able to use promotion code and admin could generate promotion code.*
- ➔ *Let potential customers browse sample chapters of finance textbook*

Due to this is a long-standing project and we have some issues with the legacy code. After discussing with the stakeholders, we decided to complete some of the user demands. We finally completed the login function, the cart function, and to let potential users browse sample chapters and information.

2. User Stories

When our first meeting with the customer, we listed 5 user stories which are:

- ➔ *Create my account and login (3 points)*
  - ◆ *Completed*
- ➔ *Purchase online textbooks (3 points). We found that in this sory there are multiple tasks. One is the action of buying. The second is to method to pay.*
  - ◆ *50 % Missing checkout function.*
- ➔ *View information pages of finance textbooks (3 points)*
  - ◆ *Completed*
- ➔ *View my current subscription status (3 points)*
  - ◆ *Not yet implemented.*
- ➔ *Generate promotion code (3 points)*
  - ◆ *Not yet implemented.*

3. Legacy code and understanding

At the beginning of understanding the legacy code, the current project maintainer seemed unfamiliar with the existing code and just delivered the whole repo to us. Thus, we started following README.md to run up the front-end and back-end sides based on the instructions in README. However, the instructions contain several incorrect scripts without mentioning the required version of each dependency.

There are too many dependency conflicts to resolve quickly, so we discuss them with the project maintainer again. Shockingly, he told us that the codes in our hand were not the latest version, but he also could not find the newest version. And this is the point where we started.

For the refactoring, a massive part of our efforts is to modify the existing codes to fit with the upgraded dependencies to eliminate the dependency conflicts.

4. Scrum masters and product owners

Since we have 6 people and 6 iteration, we all took turns for one time scrum master and one product owner.

Iteration 0. Product owner: Kai Chih Huang	*Scrum master: Po Han Hou*
Iteration 1. Product owner: Chia-Yuan Chang	*Scrum master: Li Yuan Hsu*
Iteration 2. Product owner: Li Yuan Hsu	*Scrum master: Xian Ji Chen*
Iteration 3. Product owner: Po-Han Hou	*Scrum master: Kai-Chih Huang*
Iteration 4. Product owner: Xian Ji Chen	*Scrum master: Eric Yeh*
Iteration 5. Product owner: Eric Yeh	*Scrum master: Chia-Yuan Chang*

5. Iteration summerize

Iteration 0: *We delivered the user stories.*
Iteration 1: *After we got the legacy code we started to dig into it and completed the "View information pages of finance textbooks" user story 3 points.*
Iteration 2: *We implemented the Google login function for the website. However, we couldn't connect to database. At this iteration, we also found the legacy code wasn't correct so we check with stakeholders and found this wasn't the legacy code. In the meantime, we kept trying how to build the connection with database while waiting for the latest version for the backend side.*
Iteration 3: *During this iteration, we had the chance to talk to the former developer about this website. He found the latest version from his side for us. After getting the latest version of the backend side, we implemented the google login api for the server side.*
Iteration 4: *Since we got the latest version, we found a lot of packages and modules are out of date which cause to some functions cannot work properly. We spent plenty time to finish this part. While upgrading the modules, we did successfully connected to the server for the first time.*
Iteration 5: *In this iteration, we noticed that there are some legacy Google login APIs that previous developers implemented, and some conflicts still need to be addressed due to the new versions of TypeORM. On the other hand, due to the*

client feedback, we also started to rearranging the README.md to deliver a comprehensive document for the client and the future developers.

6. Meeting time with customer and what we showed

9/22/2022; *9:30 am – 10:30 am; Online – Basically discussed the user demands.*
10/4/2022; *10:00 am - 11:00; Online – We met Prof Yasskin's assistant, Mathew, and he gave us a walkthrough for the legacy code. This is also when we found the legacy code wasn't latest version.*
10/6/2022; *9:30am -10:30am; Online – We showed the stakeholders about the plan and talk to him about the blockers.*
10/20/2022; *9:30am - 10:30am; Online – We showed stakeholders the process of login frontend side and we told the stakeholders about the difficulties on the server.*
10/27/2022; *9:30am - 10:30am; Online – The stakeholders set up a meeting for us with the former developer.*
10/27/2022; *4:00pm - 5:00pm; Online – We talked with the former developer Tristian and with his help we got the latest version from his side and started to work on it.*
11/4/2022; *9:30am - 10:30am; Online – We did not deliver anything to stakeholders since we implemented the server side of the login at this point but still couldn't connect to the database.*
11/11/2022; *9:30am - 10:30am; Online – We found there are a lot of modules in the legacy code that are out of date and we had to upgrade all of these including the dependencies.*
11/17/2022; *9:30am - 10:30am; Online – We finished upgrading and make the website run again with, finally, the connection between frontend and backend.*
11/23/2022; *9:30am - 10:30am; Online – We talked to stakeholders and he expected us to create a complete version of documentation for the future developers.*
12/01/2022; *9:30am - 10:30am; Online – We told the stakeholders about we finally finished the login function and get rid of all the blockers but did not have time to finish the est of it.*

7. How many branches and releases did you have?

We now have two branches, the main one is the legacy code with some modified and the other one is the one used to deployed.

8. Issues when releasing to Heroku.

The code is packed with docker which is used to run mariaDB. The website was using mariaDB at first. Since we can't run docker on Heroku, we register a mongodb to connect with heroku so that we could put the data into mongodb instead.

9. Issues while using AWS and GitHub and other tools.

When using AWS Cloud9, we need to use yarn build to download all the packages. Since we used the free version of AWS Cloud9. The memory is not big enough to accommodate yarn build . Therefore, when deploying the server, we can only use our own environment in our personal device.

For github, since we didn't push the environment file into github, we have to make sure the documentation mentions the variable or file we should edit or add after pulling the code from github.

10. Other tools we used

We used MongoDB since it's compatible with heroku.

11. Repo contents and the process, scripts.

We have had problems with legacy projects that maintain many deprecated libraries and dependencies. In our project, we provide two branches in our repository. One is the refactored version, which contains upgraded dependencies without conflicts; the other branch is the web application with a brand new back-end framework and database server. Additionally, we provide comprehensive instructions for deployment by running the front-end, back-end, and database server.

12. Links

Github URL:

https://github.com/b06611033/Online_Textbook/tree/deploy

Pivotal Tracker URL:

https://www.pivotaltracker.com/n/projects/2600325

Heroku Url:

https://mymath-website.herokuapp.com/

Presentation:

https://youtu.be/25AvixFjC9Q