Team Final Report

April 29, 2020

Team 3: Owen, Gracie, Abby, Ben, Brad, Katie, Ana

Introduction:

Our project is a gradebook application for the mastery system, modeled after the system used by Davidson professor Dr. Heather Smith. In our application, professors can create a customize a gradebook for a particular course, enter and view student grades, and track overall class progress. Additionally, students can use our application to view their own grades and track their progress in the course.

1. Project Management

- Question 1 - Expected Level of of Effort in Person-Months

What we said: We anticipated our final project will contain 800-1200 LOC and that it would take 1.566 person-months to complete.

Actual results: Our final project contained a little over 1,000 LOC and took less person-months to complete than we had anticipated.

- Question 2 - Overall High-Level Schedule for the Project

What we said: We estimated the project to take approximately 2-3 months to complete and broke it up into 5 sprints over a 12 week period.

Actual results: Due to the interruptions caused by COVID-19, we did not stick to our original high-level schedule as anticipated. Instead of dedicating three sprints to receiving feedback and performing testing, we tested our code throughout the development of the application. We sent our first demo to Dr. Smith for feedback and she responded later than expected but with positive comments. She did express a couple way to improve our application, and we tried to

implement as many of those changes as we could in the past few days. However, we did not recieve continuous feedback from her or from student users as we had originally planned, as it was difficult to get feedback with everyone working remotely.

- Question 4 - Risk Management Plan

What we said: Our initial list of risks included the security and confidentiality of student data, the reliability of student and professor data transactions, scheduling conflicts between team members and professors, and compatibility issues between tools and technologies.

Actual results: By far, the biggest problem that we did not include on our original list of risks was the effect of COVID-19. We did not expect to have to work remotely during the most significant part of the project. This greatly affected scheduling conflicts between team members due to being in different time zones, as well as the inability to meet with Davidson faculty. Nonetheless, our risk mitigation strategies proved very effective, and we were still able to produce a working application.

2. Process Model

In question 1 and 2 of the process model report, we said that we would use the Agile process model with extreme programming for developing our application. However, due to the COVID-19 interruption, we were not able to follow the Agile process very closely. As soon as we began developing, we all had to go home and work remotely. This not only delayed our development, but also made it very difficult to have sprints, stand ups, and other elements of Agile. We ended up following an iterative Waterfall process model.

In question 5, we stated our sequence of events would be as follows: define initial requirements, design the application and architecture, code, test, release and then repeat the process. As I mentioned, we ended up following an iterative waterfall process model and did not iterate on the entire process of defining requirements, designing the architecture, etc. Instead, we iterated in the coding and testing portion of development. That

is, we continued to add more complex features to our application, testing each feature before beginning coding again. While this was not our intended sequence of events, it ended up being successful for the team and the application. Overall, we just didn't have the time to iterate across the entire process like in Agile. Also, our application was straight-forward enough that we could define all of the requirements and the architecture at the beginning of the project.

3. Software Requirements

We have a basic implementation of all our requirements. We have successfully implemented the professor and student side applications using the R packages we planned on using. The interfaces navigate with a sidebar menu and the tabs are organized as planned. While we have implemented these requirements, there is still room to improve the user interface.

We did not have any requirements regarding the visualizations we would include on either interface and we did not decide how we would implement them. We discussed using d3, but decided to use the *echarts* package in R.

4. Software Architecture

We implemented the same set of components that were described in our SAS. We have two interfaces, one for the professor and one for the student. These interfaces are stored within their own R Shiny app and each app communicates with the database.

5. Software Design

In our original software design document, we had 4 components. We had 2.1: Client- R Shiny Application, 2.2: Shiny IO Server, 2.3: DataCats Server, and 2.4 Database. Our actual design is extremely close, if not exactly, what we wrote in this section underneath each

component. We included all four components, as anticipated, and we followed the original design in terms of language, design, and detailed design. We would rate how closely we followed our design an 8.5, because since we never specified what visuals we would include that aspect changed, and the overall UI changed as we created the app. No apparent issues held us back from following our original design, we just did not include enough specificity at the beginning in terms of visuals and UI, and therefore those were the aspects most subject to change as we began coding.

6. Summary

Overall, we think this project was quite successful. The team was able to implement a working prototype of a mastery system gradebook dashboard for both a student side and professor side.

Members of the group may pursue an independent study in application development and deployment for next semester. The team had a lot of fun! We all bonded over our shared interest in software development and education.

We anticipated the project to be somewhat difficult, and it was about as hard as we thought it would be. It was harder to get the whole team up and running on GitHub than we originally anticipated. After this initial challenge, we were able to make changes and teams made progress on their sections.

We would recommend a future team do a similar project. Shiny a great tool for quick software application prototypes.