Project Report

Isaiah Lleva, Brittany Hill, Tyler Blackmon, Ryan Kokora

Capstone Project Fall 2021

University of West Florida

December 3, 2021

CIS4595 Capstone Project

Dr. Bernd Owsnicki-Klewe

Table of Contents

A. Executive Summary

Our project didn't go as far as we expected, but we got to a point where we feel that we can make applicable modifications in the future. We managed to create a database, save data, and create a user interface in which a user can interact with. We wanted to be able to dynamically create users and classes without having to hard-code them. Unfortunately, we couldn't implement that. This report will outline the accomplishments, the drawbacks, and the remaining plans we have for our application. The topics to be covered for this report are as follows:

- Final Requirements and comparison with the initial requirements
    - This will compare what we have initially required for our project to what we have implemented in the end.
- Final Timeline and comparison with initial timeline
    - This will compare our initial timeline plan with how the timeline ended up occurring.
- Project results compared with expectations
    - This will compare the results of our project with what our initial expectations were.
- Software Evaluation
    - This will take the different software evaluation strategies we chose in the beginning of the semester and describe whether we accomplished them or not.
- Work to be done
    - This will describe the remaining work to be done for the rest of our project.

B. Initial Requirements vs Final Requirements

Here is the list of our initial requirements:

- The ability for students to see the completion status of each SAI
- The ability for professors to login to a faculty portal, filter responses, and see the number of responses received from a given class
- Write user responses to the Firestore Database
- Dynamically create users (students and professors) without having to hard code them
- Dynamically create a list of courses for students without having to hard code them
- Create login information for students and professors
- Implement a clean user interface in which the user can navigate around pages seamlessly
- Implement evaluation forms for courses and facilities
- Allow professors to see a list of responses submitted by students

Here is the list of our final requirements:

- Write user responses to Firestore Database
- Create a Google account for a student and a professor to allow them to login to our application
- Allow professors to see a list of responses submitted by students
- Implement evaluation forms for courses and facilities
- Implement a clean user interface in which the user can navigate around pages seamlessly
- Implement a Google sign in

As shown, we couldn't implement all our initial requirements. Although we tried our best to meet each one of them, we ran into obstacles that prevented us from meeting them.

C. Initial Timeline vs Final Timeline

Our initial timeline is listed below:

- Presentation 1 - September 20, 2021
- Initial GitHub Creation - September 22, 2021
- Firebase setup and sample UI - Estimated October 8, 2021
- Evaluation Form Creation - Estimated October 15, 2021
- Presentation 2 - October 18, 2021
- Alpha site available - Estimated November 1, 2021
- Deployment - Estimated November 26, 2021
- Final Presentation - November 29, 2021

Final Timeline:

- Presentation 1 - September 20, 2021
- Initial GitHub Creation - September 22, 2021
- Firebase setup and sample UI - Estimated October 8, 2021
- Evaluation Form Creation - Estimated October 15, 2021
- Presentation 2 - October 18, 2021
- Get user input to write to database - Estimated November 24, 2021
- Implement Google Sign in - Estimated November 26, 2021
- Final Touch ups – Estimated November 26, 2021
- Final Presentation - November 29, 2021

As expected, we didn't meet our initial timeline, but we did our best. Due to the new version of Angular (Angular 12), we needed extra time to look over new documentation to meet our needs with our Firebase database. There were also some unexpected issues that we encountered that may have also slowed us down, but as mentioned earlier, we tried our best to work around any foreseen or unforeseen issues.

D. Project Results vs. Expectations

Our project results aren't too far from our expectations, although minor changes had to occur. The list of things we have expected that are missing include:

- The ability for students to see the completion status of each SAI
- Professors to login to the faculty portal
- Filter students' responses
- See the number of responses received from a given class
- Dynamically create student and professor users without having to hard code them
- Dynamically create student courses without having to hard code them

Due to the lack of extra time and other issues we had, we couldn't implement it. However, significant ground was covered in the time span we had and managed to accomplish more than we thought.

E. Software Evaluation

Our software evaluation plans are listed below:

- **Source Code Quality**
    - Our source code follows a consistent style. It is also easy to understand and can be tested.
- **Usability, User Interface, and Documentation**
    - Our project was able to achieve this as we had an intuitive user interface, clear labels, and forms that allow users to navigate our website easily.
- **Security**
    - We ensured that our project was secure by using encryption methods in order to make it difficult to breach.
- **Performance**
    - Our application hasn't been tested enough to see if it can handle high volumes without causing a strain on the system.
- **Architecture Quality**
    - Our software follows a strong architecture that allows us to complete necessary tasks and accommodate new needs in the future.
- **Data Quality**
    - Our data represents real-world information.
- **Interoperability**
    - Our application hasn't been tested on other devices such as smartphones. We only have tested it using our own machines.

F. Work to be Done

- Create a way for students to see the completion status of each SAI
- Create a way for professors to login to a faculty portal, filter responses, and see the number of responses received from a given class
- Since our student and teacher users are hard coded, we need to create dynamic login users for multiple students and professors.
- Since our courses are also hard coded, we would have to find a way to make them dynamic as well.