Sprint Report #3

December 6, 2016

Team Overview

Project

DoorPanes

Members

- Andrew Fagrey
- Jayson Kjenstad
- Samantha Kranstz

Sponsor

- Dr. Jeff McGough
- Dr. Christer Karlsson
- Brian Butterfield

Client Meeting Time

Wednesday at 1:00 p.m.

Sprint Overview

During this sprint we started to focus more on the development of the project. Before we started the actual development, however, we made sure to go through the product backlog and break out each requirement into specific issues in Bitbucket.

Once this was done, we started the actual development. We started developing the web API endpoints, as well as starting to develop the tablet application using Xamarin.

Setbacks

Tablet Application Development Change

Our team had a very major change come up during this sprint. We made the decision to abort the use of Xamarin for our tablet and mobile applications. Early on in the process, development in Xamarin was going ok. Development was being done on the user interface so a user could log in and view the dashboard where the calendar informations will be displayed. It didn't take long into sprint 2 to run into some very major problems. Xamarin was not able to find and link in the needed appcompat files. AppCompat is a set of libraries which allow for areas of the application to be developed in a newer version and at the same time work with older API levels. Due to this problem, we were not able to add anything in our application such as themes, action bars, or navigation drawers. This made it impossible for us to create a user interface at a high enough level needed for this product.

As a team we spent over a hundred man hours trying to figure out what the problem was. We researched what was causing this problem and there were dozens of solutions found which enabled Xamarin to find all the needed AppCompat files. Our team did everything from adding/removing android API levels, chenging target API levels, deleting and redownloading the libraries, and completly removing and redownloading Visual Studio, Xamarin, and Android SDK. After many attempts with this, and probably too many, we as a team decided it was no longer worth our time. Also, there is not a lot of support available for Xamarin so it was difficult from the start. Xamarin discontinued Xamarin Studio for Windows. Their product seems to be going away from Windows and focusing on OS X. This was one of our theories for why the use of Xamarin forms did not work.

We also thought there could be some hardware issue. Two team members had the exact same environment set up besides for the machine it was running on. One member was using Fujitsu T902 and the other was running on a slightly newer model, Fujitsu T725. There were no errors building the Xamarin forms project on the T902 laptop. The project could not build on the T725 tablet with the exact same Visual Studio version, Xamarin version, Android SDK, downloaded API levels, and AppCompat libraries. After going step by step checking every possible factor we could think of, the project still would not run on the T725 computer. This is when we made the final decsion that we would talk to our client about the roadblock that Xamarin was causing our project. We reched out to our client and explained what was happening and it was becoming a major setback. Our client then gave us a few options to move forward from this problem, one being move to native Android development.

We had originally made the decision to use Xamarin so our single project would run across all mobile platforms. However, with all the time spent on getting Xamarin to work correctly, we decided we could use that time to first build the application in native android and later to run on iOS. Starting at Sprint 3 we had to completly restart the application develpment, but we were hopeful that develoing native would be a breeze compared to the struggle with Xamarin. And after just a couple weeks of development the application is already lightyears ahead of what it was during our struggle with Xamarin. As of now we are working hard to catch up on devlopment for the tablet application and hope to be in a good position come to the start of sprint 4.

Other Setbacks

Other setbacks included:

- Trouble understanding how the API endpoints worked
- Getting the database models figured out and having to redo them a few times

Deliverables

- Web API endpoints that could save and retrieve calendar event models.
- A tablet application concept with login screen and calendar event view.

Activities

Team

- Worked to understand concepts.
- Research.

Andrew Fagrey

- Worked with Samantha on understanding and developing the API.
- Worked to understand the JSON storage in the open source calendar framework used in the web application.
- Spent a great deal of time trying to debug the Xamarin issues with Jayson.

Jayson Kjenstad

- Worked on the tablet application.
- Worked on getting Xamarin working.

Samantha Kranstz

- Worked on understanding the web API.
- Developed API endpoints.
- Developed API models.

Work that is carried over into Sprint 3 is as follows:

- Finish calendar event endpoints.
- Creating web application endpoints for creating and saving JSON calendar events.
- Create a solid native Android tablet project.

Backlog

Azure

- Set up Azure
- Create Azure database
- User Authentication
- App Communication

Web Application

- Design Wireframes
- Code the user interface according to wireframes
- Create project and test project
- Create website login screen
- Communicate with Azure
- Create schedule templates
- Create send and receive message system
- Open calendar framework JSON events
- Connect calendar event creation to database
- Load events from database when navigating to dashboard controller
- Remove authorization code

Web API

- Create Professor Model
- Create Office Personnel Model
- Create Calendar Event Model
- Create Student Model
- Model Location
- Create GetCalendarEvent endpoint
- Create GetCalendarEvents endpoint
- Create GetCalendarEventByOwner endpoint

- Create GetCalendarEventsByDate
- Create SaveCalendarEvents endpoint
- Repository Layer
- Serialize data
- Migration on database
- Create project and test project

Tablet Application

- Design Wireframes
- Code the user interface according to wireframes
- Create room login screen
- Design and create splash screen
- Enable tablet to connect to Azure
- Create communication class
- Display a message on tablet screen
- Display schedule on tablet
- Put tablet in kiosk mode
- Move tablet code

Student Mobile App

- Design Wireframes
- Code the user interface according to wireframes
- Create project and test project
- Create app login screen
- Communicate with Azure
- Create send and receive message system
- Allow push notifications on mobile app
- Allow user to view professor and classroom schedules
- Create request form for meeting with instructor
- Create request form for a classroom reservation

Miscellaneous

- Logo
- Learn Azure
- Learn Xamarin
- Connect Visual Studio to Azure
- Look into networking for tablets
- Look into a pre-built calendar framework for displaying calendar events