01000111 01110010 01101111 01110101 01110000 00100000 00110001

Documentation, Tips, and other things

The purpose of this document is to provide future CS and C&BA students with the means to understand what successes and failures were encountered along the way with the ELO project. This information serves as a summary of all the information already encapsulated in the sprints, presentations, and GitHub codes. Note that the work defined in this document is only provided by the “middle-end” and “back-end” teams, respectively.

**Sprint 1**

**Summary**

The team learned the basics of several tools needed for the project. These tools included WordPress, Python, MySQL, Tableau, Excel, and various WordPress plugins. Prior to Sprint 1, there had been work done on this project, so the team spent time catching up on what had already been done/what still needed to be done or reworked. Additionally, the team worked on clarifying exactly what the client wanted/needed. This was accomplished mostly through in-person meetings with both the C&BA teams, and the other CS 495 group working on the front-end of this project.

**Successes**

* Installation and basic management of WordPress
* Set up basics for taking in data in an excel file
* Updating requirements for the project.

**Sprint 2**

**Summary**

Following the planning and detailing of Sprint 1, Sprint 2 was geared towards implementation. During this sprint, the team never had access to a server that was not locally hosted, therefore, sharing work that was strictly on WordPress was difficult. The meetings with the C&BA teams consisted of giving them updates on our progress with the back-end software components. Towards the end of the sprint, it was proposed that we implement “Bama Pulse” into our process, so we needed to begin to learn how to make REST API calls from their backend called “Give Pulse.”

The main goal of this sprint was to create, update, and use an excel file on the locally hosted WordPress servers. However, complications arose from this later detailed in the failures. The idea was to create a form and, upon submission, send data directly to the excel file which is used for the ELO database. Python had a way to query form submissions from the backend side of WordPress (MySQL) but calling this script automatically became an issue.

**Successes**

* Python integration
* The ‘openpyxl’ library provided a way to create and update excel files
* The ‘mysql-connector-python' library provided a way to interact with MySQL databases including WordPress’s developer’s back end. This was used to get form data from Fluent Forms. This could be integrated with ‘openpyxl’ to produce a working database
* JavaScript
* Learned how to get data from Fluent Forms via jQuery script, could display data from each form

**Challenges**

* Python integration
* Running python scripts directly from WordPress is difficult
* Python scripts cannot be called from the JavaScript code provided by the Fluent Forms editor
* JavaScript
* Calling 3rd party scripts was difficult
* Ideally, we would get Fluent Form data directly after a user hits the submit button and then add that data to the Excel file. JavaScript, in the implementation, could not achieve this in WordPress
* Getting a server
* We needed a server to test what we did on our locally hosted networks. This did not happen in Sprint 2

**Steps for Sprint 3**

* Find a way to store excel files securely
* Find a way to run python scripts automatically
* Find a way to access Give Pulse’s API

**Sprint 3**

A couple things happened during this sprint. Data collection switched from being entirely reliant on WordPress and Fluent Forms to being reliant on Bama Pulse (Give Pulse). We got a paid plan for a WordPress server (such that everyone now has access to one server). We needed to figure out a way to get data from Give Pulse via their API. Early on during this sprint we got access to the documentation for the API as well as the information needed to access it. We found a few Python libraries that could easily access REST APIs.

Without the need to create the forms ourselves (for the ELO database at least), we could move away from WordPress’s Fluent Forms.

We also had to start to incorporate Tableau which meant there was a possibility we needed to access their API as well.

**Successes**

* Python Integration
* The ‘base64’ and ‘requests’ libraries made it possible to use Python to connect to a REST API. Using this code, we can access Give Pulse’s API and begin to extract data.
* The ‘pysftp’ library made it possible to use Python to connect and get files from SFTP clients. This was never directly used, but if needed this was an option to move files around on WordPress
* The ‘tableauserverclient’ library provided the easiest way of accessing data sources from tableau online.
* Were able to use boxsdk to store write to an excel file stored in box

**Challenges**

* Givepulse API
  + The get courses error is throwing a ‘500 internal server error’ when being called. We’ve followed all steps according to the documentation and cannot resolve the issue. The other methods work just fine using the same process
* Box SDK
  + Need a token for authentication. Right now, only works with a temporary ‘developer token’ which expires after 60 minutes and you have to manually replace reset it and replace it in the code
  + Need some way to get a permanent token that won’t need to be changes
* Server
  + We were never given access to a university server to host the script on. Moving forward this would ideally be stored on one and would run as a scheduled job that would check givepulse for updates.