Joab C. Fernando
2013-36393
CMSC 198 AR-9

CMSC 198 Final Report

The company I went to for my internship is MyPortal Exchange Inc., a start-up company but they have accepted a number of interns over the years. The company I worked for is a project-based company which focuses more on web and mobile development. They try to solve problems presented to them by their clients by offering them cost-effective, interactive, scalable and reliable platforms. MyPortal Exchange Inc., under the company Viventis Search Asia, is headed by Sir Richie Yap and our office is located in Ortigas.

We had been set under a lot of pressure even before the internship started as we were carrying the name of the university with us, so I was really eager to prove my skills as a computer science student from UPLB. On my first day, I was 30 minutes early than my supposed time in. As I entered their office, cozy was the first thing that came into my mind. They were really accommodating to us, interns, and the pressure to prove ourselves seem to lessen because they didn't see us as UPLB students but as we were, interns. They tried to impart with us not only the technical skills but also the non-technical skills needed to survive long in the industry, and they did. The work environment is very relaxed, and as Google have said so, it promotes productivity among the employees.

The interns were divided among their preference; web development or mobile development. I chose web development to further enhance my skills and to gain new ones. The company presented the web development team their own problem in their company—their stacks of leave application papers from their employees. The company has given us the liberty to choose the programming language to use and the implementation of the solution but since they are teaching us Django (Python framework) as the language, we stuck with Django as our language.

Before we started with our real project, Leave Application for the employees, our mentor Ms. Ia Cabatbat asked us with the familiarization of the language by doing simple projects. We were first taught on how to create a project template and after that we were tasked with completing the tutorials given on the Django website which outputs a simple question poll system. At first, it was very hard because Python was not taught in my curriculum but as the time goes by, it became easier because the language became more and more familiar with NodeJS and AngularJS, web programming language that I am very familiar with. Django has a Model-View-Controller structure, so in doing the question poll system, we first did the models for the database, the views for the browser and the controllers to connect the model with our views. The question poll system was fairly easy because in the tutorials you just have to copy and paste the code from the tutorial and it will run smoothly. The real challenge came when our mentor gave us her own specifications for the mini project we're going to make before the real thing.

The task given to me was a simple enrollment system in which there is a list of subjects and a user can choose what subjects to enroll and it computes the total tuition fee based on the subjects the user enrolled in. It was hard at first because I will create my own models, views, and controllers but at the same time it was easy because I can implement it however I wanted it. I

created the model with the implementation I had formed in my head. I had boolean attributes for checkers and foreign keys connected to different classes in my models. The hardest model I had to make was the list of subjects a student has enrolled in. A student is supposed to have an empty list in which subjects could be added into, and it wasn't as easy as it sounds. I had to consult different stackoverflow threads to finally come up with my own answer—a field with one to many relationship with the classes. In creating the views, one problem stood out—how can I bind a subject object to a table row for the user to select it with a checkbox? Binding a whole object with an entire row is easy in a different language but it was very hard to implement in Django. In the end, what I did was to bind only the primary key of that object to the checkbox and when it is submitted I will have a function underneath that loops based on the subjects enrolled and add it to a local variable and I will then again submit the object to the view. It was a work around of the thing I wanted to implement but it works so I was okay with it.

I wanted to implement something I always do in other languages in this enrollment application, in which there is a base html and the only thing that changes is the content inside, which means no more redundant importation of css files, javascripts and images. So, I consulted my mentor personally if this was possible. At first it was very hard to communicate what I wanted to do but as soon as she got what I was trying to say, I was taught on how to do it and the rest was up to me. It was a very enriching experience for me because she was hands-on and passionate to teach me how her code works and cited examples which were implemented in one of their projects in the company.

After the submission of the basic functionalities, Ms. Ia gave us a convention with the directories in our project that involves the right location of the html templates and static files such as css, javascripts and images, and we have to implement it at the enrollment application. It was hard at first because the html file can't find the resources with we were trying to import using the new templating system but after a long debugging it was solved.

When the web development team was familiar with how Django works, we soon started with the Leave Application. We first created the model because we wanted to finalize our model draft before we move on to our views and controllers. The model will serve as the base of our project and if there is one mistake in the model, we will do everything again. It took us a week to finalize our needed models and we didn't get many errors but we kept revising the type of the variables for the convenience of the future users of our application. After finalizing the needed models, we started the authentication system for our leave application. Even in my previous projects, I can't do the authentication system right so I always ask my teammates to do it for me and I would just edit their mistake, so when the opportunity came to be taught how to do a fully working authentication system in Django, I immediately asked my mentor to teach us. I can't believe that it was so simple, you just have to catch the username and password and Django with authenticate it for you but I added the redirection and protection of pages to ensure that the system is secure. I did a fully working authentication system in two days which scared me, so I asked my teammate to test it for me. Turns out, there is an infinite loop when the admin is logged in. Since there is already a user authenticated (admin) another user can't log in but the login page is always rendered and the page keeps on reloading because the authentication system can't verify if a

superuser is logged in. A simple control flow fixed the problem easily, it checks whether a user is a superuser.

Since the basic modules are done, we then started to code the backend routes to get the data from the database and the static user interface. The aim of the user interface is to be as professional looking as it can be but simple at the same time for the users to access our system at the their fingertips. Creating different views for different roles of a user is a tedious task, we have to create an additional tab for the manager, a different module for the admin but we have to ensure that the employee can't access those features. So our source code is long but it's not so long that the code is not readable. It's the best we can do as of now.

Bianca Marie Luna