**Report on Project for CSE350**

This project aims to address some issues we face when our semester results are published.

1. The PDF containing everyone's result is not well formatted.
2. Access to one's result is dependent on having a copy of the PDF. But that should be available online.
3. One has no option to filter his results, i.e. how is he doing in only major courses.
4. If someone has a different programmatic approach to show/work on the data of results, there is no API available.

Our solution to these problems are:

* A relational database system containing data of students, courses and obtained grades.
* A good looking website that will make viewing and filtering results very easy.
* An API for developers to build applications for different platforms with available data.

Some of the tools used to build this project include:

* Python-3.8.1 and Javascript as programming and server-side scripting languages;
* pdfplumber-0.5.25 to scrape the tabulation PDFs provided by the university;
* pandas-1.1.5 to facilititate working with tabular data;
* SQLite-3.31.1 as the RDBMS;
* Django-3.1 as the web framework;
* djangorestframework to build the API;
* xhtml2pdf to convert webpage to PDFs available for download;
* bootstrap as the CSS framework;
* pythonanywhere.com to deploy our web application.

We had some prior experience working with basic Python. So our choice of tools are based on this language.

At first, we divided the workload of the project between us. I, Navid, was tasked to scrape the PDFs and make the data, avoiding inconsistency, available in a relaional database. Sazin was to build the website from scratch, frontend to server-side. API was not included in our plan then. But it came along later and Sazin built it.

I checked and tried some popular PyPi packages to help scrape the PDFs and settled on pdfplumber. It has easy conversion of PDF tables to pandas dataframes. Before populating tables in the database, the data needed some cleaning. I learnt manipulation and insertion of data inside a dataframe. Then I wrote some programming logic and string operations in a jupyter notebook to extract data items and use them to populate new dataframes accordingly. Then we agreed on a database schema.

Sazin was learning the Django framework and Javascript to build the website. He worked quite hard to grasp the concepts of the modern web. He also had to learn HTML and CSS since these are the building blocks for the frontend.

In a meeting with the course teacher, he instructed us to work on an API that would enable us or other developers to use the data to build platform dependent applications later. Sazin was the head behind the API. He had to redesign the database schema so that the code for API was easier and took less time to be built. He learnt django-filter to add filtering to the API and djangorestframework to build the API.

According to the schema, I populated the database tables. I had to use some raw SQL queries alongside Python's sqlite3 package to correct some errors and in the data.

Sazin learnt and used bootstrap CSS framework to build a good looking user interface.

Our web application is deployed at ......pythonanywhere.com. Features of the project include:

1. Easy searching by registration number;
2. view individual results using different filters i.e. only majors, only labs, only 4th semester;
3. view current ranking of students of the same session;
4. an API to provide data related to courses, and grades of students;
5. view the API response as a JSON object;
6. preview and download the output of search and filtering as a PDF.

If we are given the chance, we can extend this project later and make a result and course feedback website for the university.