# Graduate Student Tracking System

- **Team Members and Roles**
  - Muxi Leng (Scrum Master)
  - Jingwen Han (Product Owner)
  - Chih Chin, Tsai
  - Yuhang Xie
  - Bo Wang
  - Yiwen Tang
  - Libing Zeng
- **Links**
  - GitHub(Documents) : https://github.com/GodBlessWuhan/TrackingSystem
  - GitHub(Code) : https://github.com/qianqiuzhongdu/TrackingSystem
  - Google App Engine deployment: https://orbital-builder-269722.appspot.com/
  - Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2437004
  - Poster Video: https://vimeo.com/415385767
  - Demo Video: https://vimeo.com/415388320

- **Summary**

  The Graduate Student Tracking System is required by department program specialist (stakeholder) to keep track of student documents and degree information, etc. The customer proposed requirements as follows: ensure log into Graduate Student Tracking System for specific people; find students by UIN, name or other identifying information; upload degree plan related to students and manage the status of degree plan; migrate graduate student's information for future use; use TAMU google drive to store the files.

  Our team completed the project according to each demand, based on legacy projects, we improved former interfaces and added more features. After 4 iterations, we successfully meet the customer's requirement and deployed our system on Google App Engine.

- **User Stories and User Interface**

  1. Feature: Log into Graduate Student Tracking System for specific people

  ➔ As a department advisor

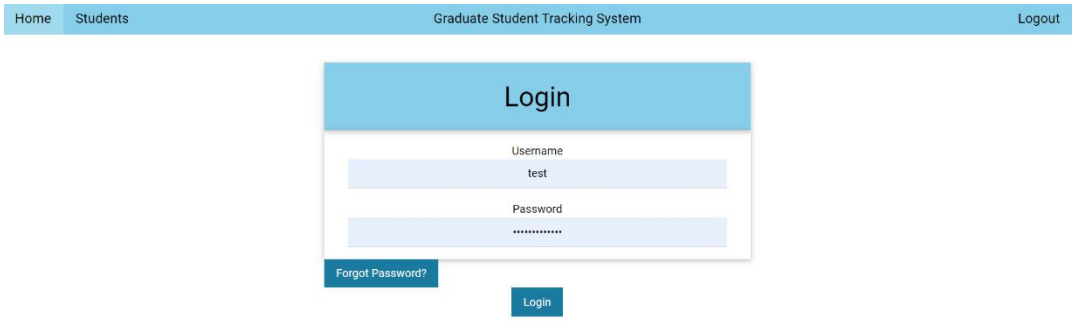
  ❑ So that I can manage and track student status and files safely

  ❑ I want to have a username and password to log into Graduate Student Tracking System

- Details:

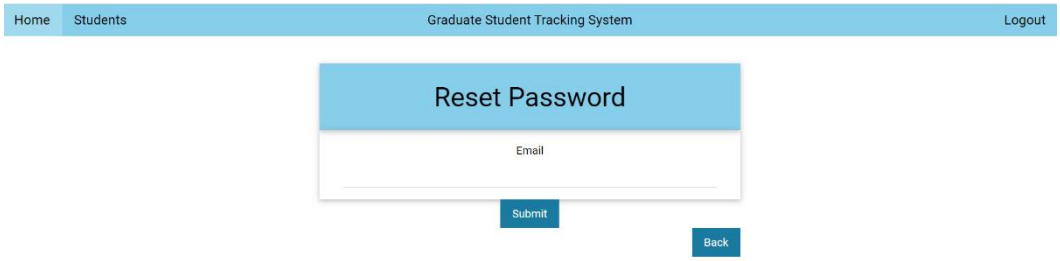
  Interface 1:

  We modified the interface created by legacy project, anyone with authority could be able to log into our system.



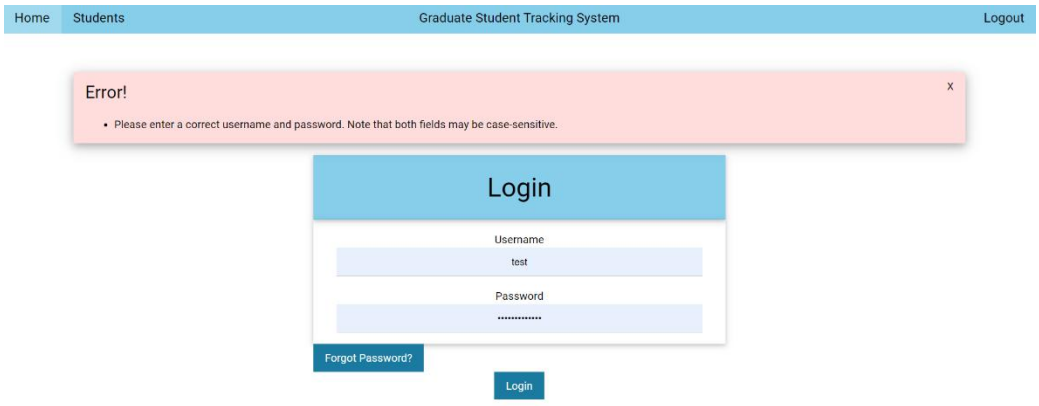

  Interface 2:

  If users forget their password, we offer a way for them to reset their password by sending emails to their registered accounts.




  Interface 3:

  After filling in the login form, if the username didn't match with the password, an error will occur and system will notify the users.




- Points: 1

2. Feature: Search a current student in Graduate Student Tracking System

➔ As a department advisor

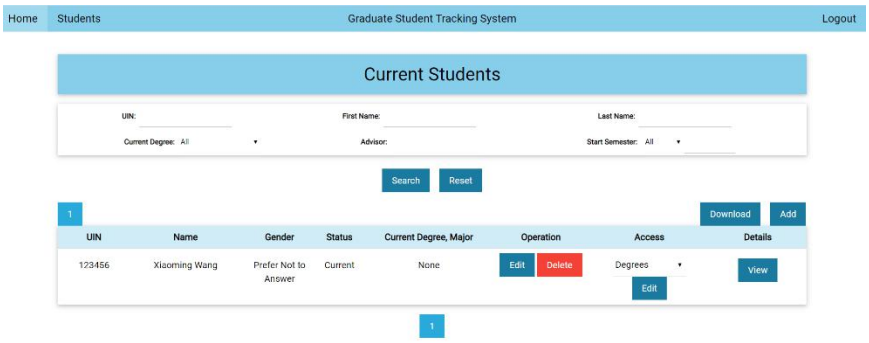
❑ So that I can find the corresponding information and files of a specific student

❑ I want to search a current student by key words, like UIN, lastname, first name, start semester, etc. in Graduate Student Tracking System

- Details:

  Interface 4:

  After login, a search interface will appear with several search requirements, along with the student information in our database.

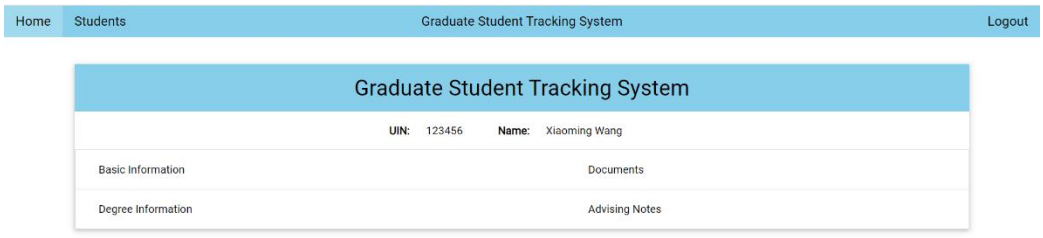



  Interface 5:

  Then we fill some information to search for certain student. But when we can't find the student information, we will have a pop-up page to show the result. Then we can click "go back" button to search for a new student.

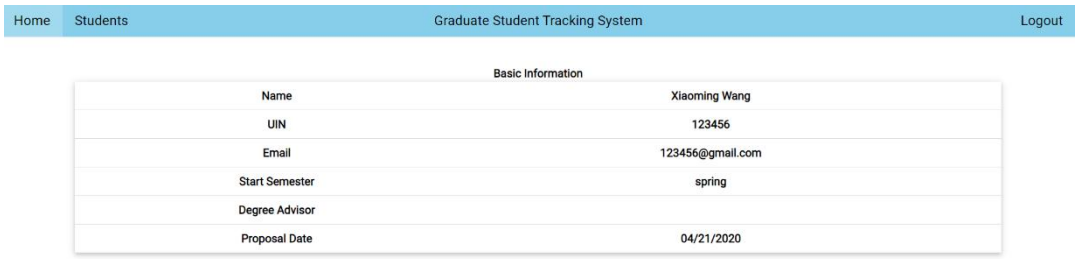

  Interface 6:

  If the information provided matched with data in database, we will get the record of that student. We added a button for every student in the main page, click the button, then page 6 shows the different information components of every student : Basic Information, Documents, Degree Information and Advising Notes.

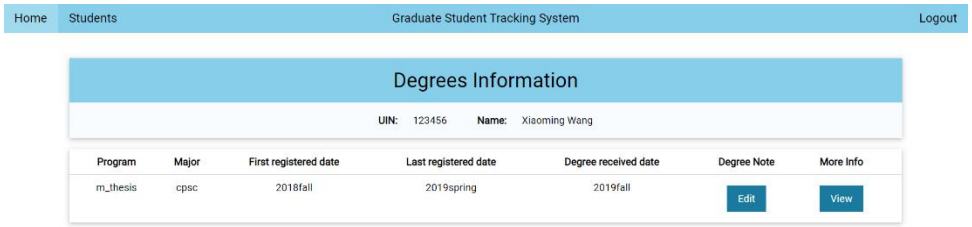



  Interface 7:

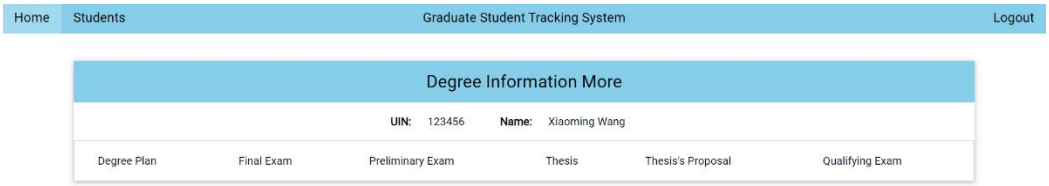
If we click on “Basic Information” button in former page, we will get a page that shows the basic information of a student. Each row represents the specific information of the student like Name, UIN, Email address, Start Semester and Degree Advisor.



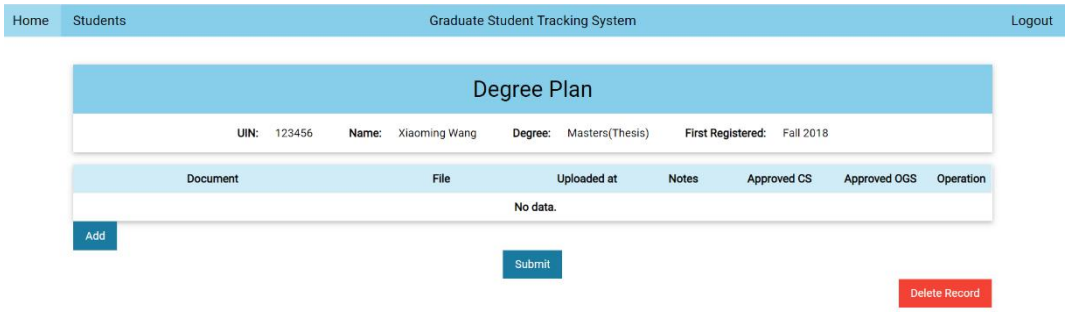

Interface 8:
When we want to check the student's degree information, we will have a pop-up page to show what degree the student has, and each degree would has its own table to show the specific information about the degree, and if we want to know further, we can click the More Information button to check some details.




Interface 9:
This interface shows more degree information like: Degree Plan, Final Exam, Preliminary Exam, Thesis, Thesis Proposal and Qualifying Exam.

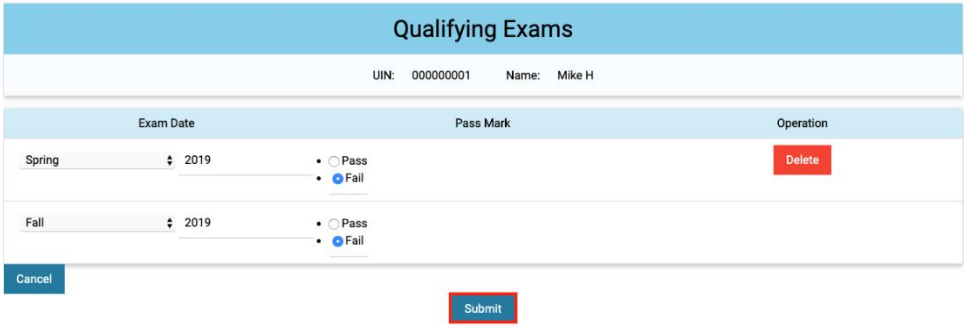



Interface 10:
Based on interface 4, we can get access to degree plan page. This page shows the student’s degree plan. Each row represents the information of this student which is related to their degree plan, like student type, file to be uploaded, committee member, and dates approved by CS and OGAPS.



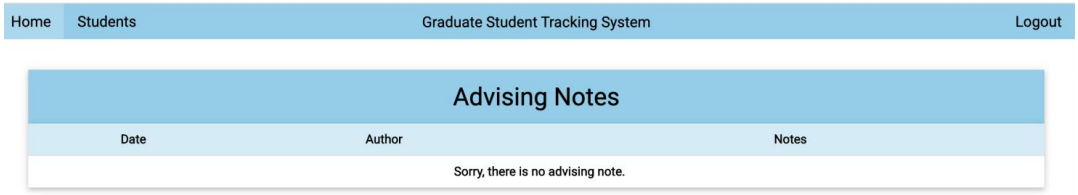

Interface 11:

PhD degrees require students pass qualifying exam in a maximum of 3 times. In order to track qualifying exam of PhD degree, this interface enables administrators to read, add, update, and delete qualifying exam records, which include exam date and pass mark. we can enter this interface from both students.html and degree_info_more.html. This interface is of similar style of degrees.html.




Interface 12:

Based on interface 6, we can get access to advising notes page. This page shows the list of advising notes. Each row in the list contains the date, author, and note content of the note. Also, we can type in notes in the text area and click on "commit" to submit the note.




- Points: 3

3. Feature: Upload each student's degree plan file to the system

➔ As a department advisor

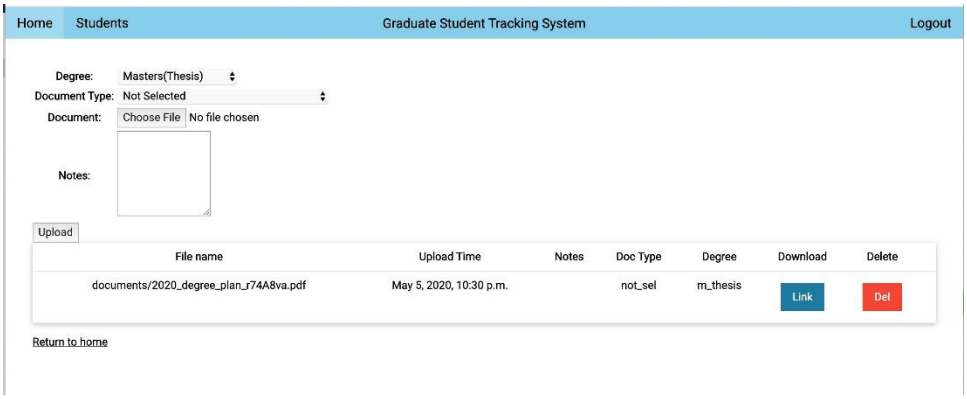
❑ So that I can manage students' degree status

❑ I want to upload degree plans and proposals in specific sections to Graduate Student Tracking System

- Details:

Interface 13:

In order to manage degree plan, proposal and some other important documents of students, we build an interface for users to upload files to the database. This page is based on interface 6, by clicking on 'Document', it will pop. In this page, we can choose the document type, degree type, add notes and upload files. With all information filled in, the data are submitted to the backend. Then click the download link we can download files from the google drive, and also click the del button we can delete the file.




- Points: 2

4. Feature: No worry about the security and legal issues of getting data

➔ As a department advisor

❑ So that I can legally and securely access student's information

❑ I want to Deploy the student tracking system on google API

- Details:

  To deploy a Django app to the Google App Engine, we created a configuration file named app.yaml in the main directory, which does not exist in the legacy code. In the app.yaml file, we specified the version of python, the handlers, the libraries(and their versions), and so on.

- Points: 2

5. Feature: Migration of graduate student's information to another place

➔ As a department advisor

❑ So that I can easily search the current student data but still keep the graduate student information at the other place

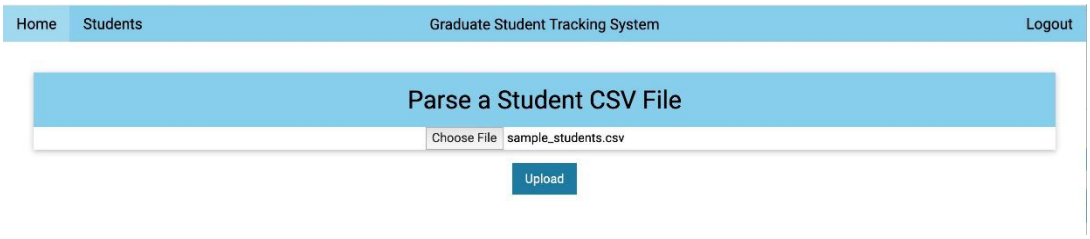
❑ I want to create a new directory to preserve the old data.

- Details:

  Interface 14:

  This page is designed to parse csv file. In the main page, added a "Parse" button. Click this button, page 14 shows up. In which page, click "choose file", then browse a

file in your computer, the system detects if it's a csv file and then added it to our database.




- Points: 2

- **Iteration Accomplishment**

❏ Iteration 0:

The customer addressed her requirements for the whole Graduate Student Tracking System. As a result, we formed a general picture of what this application looks like: First, the data can be searched by student information. Then, for each student, the functions available to advisors include uploading the degree plan, monitoring student preliminary/final exam status, adding a check list of qualifying exams for each semester, managing student's thesis information and typing OGAP approving date. In addition, above information, advisors can download it as a PDF file.

By completing iteration 0, we designed an integrated framework of our system. Moreover, we reflected on the legacy project and formed our logic to improve their work. Everybody in our team made up our minds to deep into Django framework.

Points: 2

❏ Iteration 1:

In the first iteration, we were mainly focused on feature 4: no worry about the security and legal issues of getting data. The Legacy Code can only be deployed on Heroku. To implement the part of the feature "deploy the student tracking system on google api", We refactored a lot of code in many different parts of the project. To be specific, we added a configuration file, changed our database to SQL in cloud and local environment, fixed some bugs caused by the change of Python version, replaced cryptography with pycrypto and imported sendgrid module.

Points: 2

❏ Iteration 2:

In iteration 2, we did a lot on feature 2: search a current student in Graduate Student Tracking System. For this feature, we have implemented the searching page with 6 key words (UIN, First Name, Last Name, Current Degree, Advisor, Start Semester) to replace the old page with more than 10 key words. Also, we realized the student information page with a table consisting 5 links: Basic Information, Documents, Degree Information,

Advising Notes, Other Documents. Moreover, we completed the searching function (backend code) and the template of the “Basic Information” page.
Points: 3

❏ Iteration 3:

In this part, we continued with feature 2. We modified the student information page by deleting the “Other Documents” item, so there only exists 4 links. We also implemented the backend code of the “Basic Information” page. Furthermore, “More Information”, “Degree Information” and error page are also completed.
Points: 3

❏ Iteration 4:

In last iteration, we carefully modified feature 1, 3, 5 according to the customer’s requirements. We realized the login page for feature 1, page of Qualifying exam and Advising Notes. For feature 3, we implemented the document page which is designed for uploading and downloading files. Moreover, we implemented a page to upload a csv file containing the student data, parse the file, and store data in the database according to demand of feature 5.
Points: 3

- **Customer Meetings**

  ❏ Meeting 1: 02/27/2020, 15:00 – 16:00, EABA

  In the first meeting, the customer explained her requirements on the Graduate Student Tracking System: a system to manage graduate students’ information. She detailedly explain the features she wants and we ensured that we can understand all her request.

  ❏ Meeting 1: 04/07/2020, 16:00 - 17:00, Zoom

  In this meeting, we showed the customer with the web pages that we have implemented, for example, the student searching page, the page of student information, and so on. Customer is satisfied with our interface, and she mentioned that we need to make some modifications with the user interface we designed. Moreover, we need to realize the features of using TAMU NetId to login to our system and using google drive to store the files.

  ❏ Meeting 4: 05/01/2020, 14:00 - 15:20, Zoom

  In the last meeting, we demonstrated the web pages that we have implemented: the student searching page, the page of student information, basic information, degree information, advising notes information, document and so on. Customer appreciates our interface and mentioned some improvement needed to be done. We promised to complete all the parts before due date.

- **Discussion of Legacy Project**

  The Legacy Code includes several major parts:

❑Login system with functions for users to change password

❑ Upload files, like degree plan, proposals and petitions, and encrypt the uploaded files

❑ Advisors add notes or decisions to the uploaded files

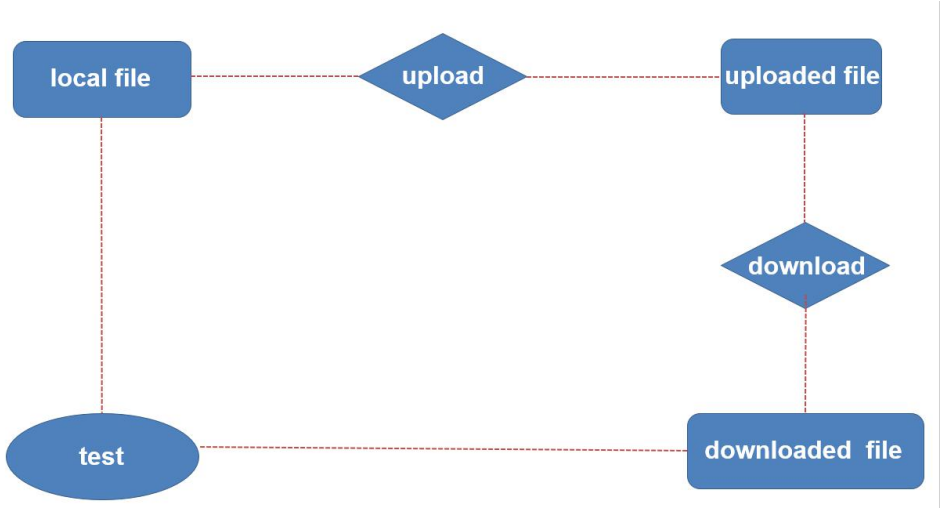
❑ Search a student in the system using various filtering information, like UIN, name

❑ Download selected data into excel

However, there are many differences between ours and the legacy project. First, as the legacy project was deployed on Heroku, instead, we figured out a way to deploy it on Google App Engine. Also, we changed the filtering requirements for searching for students, added modules for monitoring student preliminary/final exam status, and added a check list of qualifying exams for each semester. Moreover, our system can manage student's degree plan and present approved dates of CS and OGAPS. Furthermore, we managed data migration by creating a page to upload a csv file containing the student data, parse the file, and store the data in the database.

- **BDD/TDD Process**

  We have conducted unit test for some important functions. For instance, we test the "Upload File" part as the following flow:




  For the most part of this system, BDD method was followed. One of our members conduct testing and tests we conducted are expected to pass.

- **Configuration Management Approach**

  When developing our app, we basically follow the "Agile Lifecycle" approach as our planning and management approach, which includes, continuously refining working but incomplete prototype until customers are happy, with customer feedback on each iteration. And we also write down user stories to validate customer requirements. Pivotal is applied to measure "Velocity" of our progress.

During the development process, we worked individually to develop different user interfaces and committed any progress to the master branch in time.

- **Deployment and Environment Issues**

  ❑Configuration file

  To deploy a Django app to the Google App Engine instead of Heroku, we created a configuration file named app.yaml in the main directory, which does not exist in the legacy code. In the app.yaml file, we specified the version of python, the handlers, the libraries, and so on.

  ❑Database Settings

  The legacy code used sqlite3 as the database. However, the Google App Engine does not support sqlite3. It can support MySQL. What's more, it's better to store data in the Cloud SQL when an app is deployed to the Google App Engine. Therefore, in the settings.py file, we first check whether the running environment is the Google App Engine or the local environment. If the running environment is the Google App Engine, then we connect to the Cloud SQL Instance; otherwise, we connect to the local MySQL database.

  ❑Version of Python and Django

  The Google App Engine only support python2.7 and django1.11, but the legacy code use python3.7 and django2.1.15, so we changed the version of python and Django, and fixed some bugs caused by the change of the versions. For instance, in the url.py file, we should replace function path() and re_path() with url() because path() and re_path() are not supported by django1.11; sel_options.py and 0001_initial.py contain latin letters, which is supported by python3, but for python2.7, we should specify the encoding "latin-1" in the beginning of the two files.

  ❑Modules be Replaced or Added

  We replace cryptography with pycrypto, because Google App Engine does not support cryptography. Besides, we imported sendgrid module and enabled sendgrid API in Google App Engine.

- **Challenges:**

  ❑ How to migrate system platform through Heroku to Google App Engine with many existing incompatible libraries and configuration problems.

  ❑ How to build an 14 interfaces based on customer's requirements and improve code reusability and flexibility.

  ❑ How to parsing csv files in the system, which helps user to upload multiple students information to database easier.

  ❑ How to call Google Drive API within our system, and verify the Tamu Google Drive account.

❑ How to upload and download files between our system and Google Drive.

- **Future works:**
  - ❑ Deploy our system in Tamu CAS platform to make the system more secure, and can be logged in with Tamu UIN.
  - ❑ Develop more features for customer use.
- **Tools**
  - Python
  - Django
  - Cloud9
  - Google App Engine
  - Google Drive
- **Difference between Ruby and Python**

Compared with ruby, the syntax of python is more easier to understand and master. One of the main difference is that in Django the parameters are passed in the render function from the controller to view, in the hash format. The documents of django is more detailed than the documents of rails.