- **Summary**
  - This project goal is to develop a tracking system for graduate students. Our customer, the advisors of the Computer Science and Engineering Department hope this system will include functions of Adding new students, Editing students, and Searching for students. Our stakeholders of this project will be Karrie Bourquin and the most important needs of them is to edit google spreadsheet and deal with folders in drive. To be precise, the customer owns a spreadsheet that has all the information of all CSE students at TAMU. Our customer hopes this project can load the contents on the spreadsheet on a website.
  - There should be a table which contains all the information on the website. Besides that, the webpage should be able to let the user to manually enter a new student's information and put the information on the spreadsheet as a new row. At the same time, there will be a new folder created for each student. The folder will be the place to contain all the files of a student. The customer also wants to edit a student's information. After the edition, the corresponding information on the google spreadsheet will also be changed.
  - By using App Script, we are able to get all the text on every cell on a spreadsheet. Then it is easy to modify the contents. Also, each folder has a url link, the id of that folder is contained in the url. We can use the app script to get the folder by the link. Then we can do modifications on the folder.
- **User Stories**
  - Story 1: " As an advisor, So that I can load the information of the students from google sheets easily, I want to have the information as a chart"
    - Description: The program will read a google sheet containing the information of the students and display the information as a chart.
    - Points: 1 Implementation status: Finished
  - Story 2: "As an advisor, So that I can click on the name of the students, I want to be able to see more information when clicking on the names."
    - Description: When clicking on a student's name, a dialog will appear containing more detailed information about the students and an edit button.
    - Points: 2 Implementation status: Finished
  - Story 3: "As an advisor, So that I can search for students' information, I want to find a student by his name or UIN"
    - Description: A search bar will be placed on the top of the chart. The user will be able to search for students either by a UIN, a first name or a last name.
    - Points: 2 Implementation status: Finished
  - Story 4: "As an advisor, So that I can give each student a folder in google drive, I want to save all the files in the google drive"

        - Description: Each student will have his/her own folder named by his/her name and UIN in the google drive. And the generated folder links will be added to the google sheet and loaded to the chart on the web page.
        - Points: 2 Implementation status: Finished
        - Changes: At the beginning of the implementation, we created a "add folder" link for each student. However, to simplify the coding process, we decided to give the folders to the students while the google sheet was loading. Therefore, students could have a folder when their information was loaded to the system.
    - Story 5: "As an advisor, So that I can change the loaded google sheet, I want to add students that are not on the google sheet."
        - Description: There will be an add button on the right lower side of the website. When clicking on it, the user will be able to write the UIN, first name, last name and etc. of the students.
        - Points: 2 Implementation status: Finished
        - Changes: The button was initially placed on the top of the web page. When clicking on the button, a dialog would be displayed with the input boxes on it. We later placed the button on the righ lower corner of the web page. When clicking on the button, a box would slide up containing the input boxes.
    - Story 6: "As an advisor, So that I can manage the current students and the former students separately, I want to be able to switch between the current student chart and the former student chart."
        - Description: There will be a button on the top of the search bar named "Switch to former/current student".
        - Points: 2 Implementation status: Finished
    - Story 7: "As an advisor, So that I can change the information of the students, I want to edit the information of the students."
        - Description: There will be an edit button on the pop-up dialog. When clicking on it, the user will be able to change the information of the students and submit by clicking the submit button.
        - Points: 2 Implementation status: Finished
    - Story 8: "As an advisor, So that I can manage the files of the students, I want to upload files to the system."
        - Description: There will be an upload button on the pop-up dialog. When clicking on it, the user will be able to select files from his or her computer and upload them.
        - Points: 2

- Changes: Since we give each student a folder, the user can simply click on the folder link and upload any file to the folder. Therefore, we did not implement this user story.

- **For each scrum iteration, summarize what was accomplished and points completed.**

  Iteration one

  We modified the user stories due to the reason that the previous user stories do not meet the requirements of the customer. After the meeting with the customer on 10/09/2020, we realize that most of the user stories should be changed. Besides that, we feel that each of our previous user stories are so big that it can be divided into multiple smaller stories and we think that by doing so, it will increase our productivity. First of all, we create a website where the google sheets are loaded. The website should be able to load different sheets. Secondly, we deal with the dialog part. When the user clicks a table row, a dialog should appear. The dialog should contain more detailed information about the student with a link to edit information.

  Iteration two

  In iteration two, we first draw a diagram for the student tracking system. On the diagram, the advisor is connected with Delete student, Search student, Add student, Add student and Advisor login. The advisor extends the Delete student, Search student, Add student and Add student. The Add student extends the Add document. At the same time, we finish the search function part. It includes the creative and link folders to students and upload documents. We also do part of the Homepage UI design.

  Iteration three

  As in the previous user stories, the program would create a folder in google drive for each student during the “Add a student” process. However, when we try to add a group of students by reading an existing google sheet, the folders will not be created automatically. Therefore, an “Add Folders” button is created in order to assign a link of the corresponding folder to each student. When clicking the add folder button, we will imply that we are processing the folders and will return a popup/warning message indicating that the process is finished. At the same time, we add the folder link to the student and create the edit page. We also decide to test the code manually. It has the following property, By adding some new rows into the table (without link) as an import function, we are supposed to see the website able to read these lines and links shows: no folder link. By clicking the “add folder” button we are supposed to see all folders are generated with lastname_firstname_uid and their links at the end of sheets. After reflash the webpage, there should be no more no folder link. By adding a student with a button below with information filled. We are supposed to see a new row added at

bot with the same information we input. By typing some key value in the search bar, we are supposed to get the corresponding filtered table right away when we input the key. By clicking the switching table button, we are supposed to get the different tables (switch from current and former student) displayed.

Iteration four

In this iteration, we Connect Google Sheets and load data to a table in html. On the security part, we improve that g-suite implementations are authorized to only modify the database (google sheets and file folders) rather than asking permission for editing the whole google drive. We move students from one table to another. Then, we deal with a security issue that asks to have permission to access and modify all files on drive instead of shared drive only.As the customer requests, we add the button on the detailed student information page that can move the current student to the former student sheets.

- **Team Role**
  - Scrum Master: Hang Zong
  - Product Owner: Tianxiang Chu
- **List of customer meeting dates, and description of what happened at the meetings, e.g. what software/stories did you demo.**
- Meeting date: Oct. 2.
  - Customers are keeping academic records of graduate students (and also ISS information for international students). The ultimate goal is to turn it from paper folders into electronic folders. The database table of the system should be in a form of google sheet. The columns should be the information and documents of students. Each row should be one person with a single degree. The first question to think about is whether to use one big sheet with a large number of columns or a bunch of small sheets. When talking about the functionality of the system, it should be simple, quick, easy to use, able to run by the end of the semester. The basic two functions should be searching by names and UINs, and adding or updating student information/documents. We don't have to worry about the delete function. Instead, we should think about it as marking a student as a former student. Once the results appear, we should be able to view them and edit them. The code of the system should be mostly using javascript. Dr. Walker showed the sample documents(PhD admission table) at last with links on each row.
- Meeting date: Oct 16.
  - We created the completed brand new User Interface and we showed that document to the customer. We also showed the functionality of the buttons and pop-up dialogues of the first two pages. One thing we needed to do further is that we need to create a folder link and use that folder to store the documentation.
- Meeting date: Oct 30.

    - We showed the functionality of the add-student button at the bottom of the system page. We could enter the basic information of a student to create a new student and set up a new student folder. One could also create a student folder by clicking on the add folder for a student button of one student pop-up dialogue. One thing we needed to work on further is to get the edit function of students information to work.
- Meeting date: Nov. 13
    - We finished the basic functionality and user interface for the system. However, we were faced with a security issue of using google shared drive. Permission to have access and modify all files of the drive (instead of the database) doesn't satisfy the legally acceptable items.
- Meeting date: Nov. 23
    - We made some changes regarding the security issue that asks to have permission to access and modify all files on drive instead of shared drive only. The customer gave us some suggestions on how to improve on that issue based on the previous solutions of the faculty's.
- **Describe any issues you had using AWS Cloud9 and GitHub and other tools.**
    - Security issue. Since the customer wants to modify folders in Google shared drive, we have to give our project permission to access the drive. To achieve this, App Script requires an authorization to access the drive. However, Google API only provides authorization to all of the user's drives, not a specific drive. Due to safety concerns, the customer is not willing to give project permission to access all his drive contents.
    - It is hard to estimate the time for App Script to finish requests on the drive side and spreadsheet side. To be precise, for example, after you add a student, javascript sends a request to App Script. Then Javascript would not wait for the App Script to create folders for that new student and put the link to the spreadsheet. In html and Javascript, it is hard to get the working status of App Script.
- **Test**
    - In order to test our program, we are trying to find some useful tools. However, there are no suitable tools to test our program. After we make a meeting, we decide to test the program by ourselves. We created some test cases to test and record the results.
- **Links**
    - Pivotal Tracker: https://www.pivotaltracker.com/n/projects/2467825
    - GitHub: https://github.com/jjjzy/StudentTrackingSystem
    - Poster and demo video: https://youtu.be/DX-D0NOopOA