Requirement Analysis

1-) Courses input: We need a json file which is composed of all the courses throughout eight semesters. These courses may also have prerequisite courses. Our program will read this json file to get course information.

2-) Student transcripts: Each student will have their own records file. Those files will have student information, course information (failed, passed, and currently taken classes), GPA and credits information of the student. On the top of this, we need a student class which holds information about students(id and transcript). Additionally, we will have a json writer for student class so that every student will have their own json output file.

3-) Professors: Each course is taught by a professor, professors will have their basic information, such as the courses they teach etc. Some professors can be advisors so that they can control course registrations.

4-) Simulation: We will generate random students to simulate our program. Students can have different numbers of failed and passed classes. Our program will control course registrations that made by the students. It will reject or approve them based on transcript information of the students. For example, let's say there is a course which has a prerequisite course, and a random student failed this prerequisite course. If this student attempts to register to this course, then our program will not allow student to take the course.

Use Cases:

Our customer will see a random simulation cycle in our program. According to customer's desires, we will build and change our software. In an example run our program will do:

1- Read the course data and generate random students in the specified season.

2- Students will have random and unique information fields. They will have passes and failed courses and so on.

3- In our simulation cycle, students will try to register to different courses and our system (aka advisor) will control those attempts.

4- Outputs will be shown on the CLI, they will tell if the attempts are approved or not. If an attempt is failed, then our program will tell the customer the reason of the failure.

Our design and implementation requirements:

In our design, we need try to separate the concerns of our program. Each component will only do things that we described.

We need to apply SOLID principles to our code. Our development cycle will consist of consistent analysis, implementation, and refactoring.

We will do pair-programming sessions with our members and choose the best practices we figure out.