Project Proposal

Name: CS Degree Planner

Project Summary

UIUC students currently have three separate tools to plan their semester plans: course audit to view remaining requirements, course explorer to view prerequisites, and GPA disparity visualizations to gauge difficulty of planned courses. Our project is a web application that allows students to visualize their pending degree requirements, explore options to fulfill them with filters for GPA and their completed prerequisites, and build semester plans that continuously update prerequisite data. When a student adds a course to their plan, the system will automatically recalculate eligibility for dependent courses, flagging ones that cannot yet be taken and showing the earliest semester they become available, so students always know when they can schedule their desired courses.

This platform eliminates the need to interact with all three tools and instead helps students build the best plan to complete their degree requirements. Since it is not feasible to build this application across all majors due to limited data availability, this application will only cover all CS and CS+X majors.

Creative Component

The creative component will be allowing users to filter courses that meet a degree requirement by whether or not they have completed/planned for its prerequisites and for them to view based on that information what semester they would be eligible to take the course.

This is technically challenging as prerequisite data is not nicely stored in any preexisting dataset. Instead, to get access to this information we must:

1. **Use Text Parsing –** The course catalog data (see realness section) has a column labeled as section info. This contains strings that use key phrases such as "Prerequisites" and “Concurrent Enrollment” followed by course codes. We will need to extract this data by recognizing the prefixes following the course codes we want to pull for each course.
2. **Storing dependency information -** Once a user selects that they have taken a certain course or plan to, we will need to store this information in a way that allows us for each course to determine whether the user is eligible to take it based on current planning and what semester they are eligible.

This feature, while technically challenging, creates user value as users do not have to reference outside sources to gauge course eligibility before adding it to their semester plans.

Usefulness

The web app has the following functionality:

Students can…

- track major requirements completion.
- find courses that meet their incomplete requirements.
- view and filter courses by their completed and planned prerequisites to see if and when it is possible to take a course.
- view and filter courses by their overall GPAs and teacher specific GPAs.
- generate semester by semester degree plans based on the previous courses they took and prerequisites for their future courses.

Similar applications:

- University degree audit
    - Does
        - Shows completed and pending requirements
    - Does not
        - Provide prerequisites list
        - Contain information on course difficulty, does not let
        - Allows students to directly form semester plans
- WAF GPA visualizations
    - Does
        - Provides historical GPAs for courses broken down by teacher
    - Does not
        - Allow users to interact data with their course requirements
- Course Explorer
    - Does
        - View courses fulfilling genEDs
        - View prerequisites
    - Does not
        - View courses fulfilling major specific requirements
        - View course difficulty
        - Create semester plans

Our difference:

Our app recognizes that students are often interacting with all three of the above tools, along with a separate semester planning document like a spreadsheet, when determining course plans. **We consolidate the 4 functionalities provided by these separate tools into one comprehensive system, a highly useful tool for students struggling to juggle all the factors that play into course planning.**

Realness – Datasets

We will use the following datasets:

1. **Course Catalog Data:** This will provide us a list of all the courses in UIUC. Under the section info tab, information about prerequisites and concurrent enrollment is given. By parsing that string, we can store the prerequisite and concurrent enrollment information for each course. Additionally, this provides us with credit hour information to ensure students align with the 12-18 credit hour range.
    a. Data Type: CSV file
    b. Number of Rows: 12322
    c. Number of Columns: 26
2. **Gened Dataset**: This dataset will map each course to what GenED it satisfies. This will simplify the logic for GenED requirements for each major, as this data is nicely stored in a preexisting csv.
    a. Data Type: CSV file
    b. Number of Rows: 1061
    c. Number of Columns: 11
3. **GPA Dataset**: This dataset hosts the gpa averages for each course across previous semesters, broken down by teacher.
    a. Data Type: CSV file
    b. Number of Rows: 74600
    c. Number of Columns: 23

We will build the following dataset:

1. Degree Requirements. This dataset will store for all CS and CS+X majors, what the degree requirements are and what courses fulfill them, given that the degree requirement is not a GenED. Since we are dealing with a small size of majors, we will manually build this dataset based on the information needed for our application logic.
    a. We will source this information from this site: https://catalog.illinois.edu/undergraduate/eng_las/statistics-computer-science-bsla s/#degreerequirementstext
    b. We expect two tables with following columns:

i. Course Info
    1. Course Code
    2. Course Name
    3. Degree Requirement Fulfilled
ii. Degree Requirements
    1. Degree Requirement Fulfilled
    2. Minimum Required Credit Hours
    3. Major Attribution (what major this degree requirement for)

Detailed Functionality

User Interactions

- Users will select their major, pending semesters, and check all the courses they have completed so far from a prepopulated list.
- User will be able to view their pending requirements
- User will be able to click each pending requirement and it will go to a new page where they will be able to filter the courses satisfying that requirement either by GPA averages, eligibility from prerequisites completed or planned, and/or by semester based eligibility
    - Example: A student planning CS233 in FA24 will be able to see that while they are eligible for CS341, they won't be eligible till SP25. A student who has not completed or planned for CS233 will show up as not eligible for CS341)
- User can view prerequisites for each course
- User will be able to create their degree plan for each pending semester where they can add each class only if their prerequisites are satisfied and their credit hour requirements are met
- Users can also update/delete courses from their degree plan to test alternative courses.

Functionality:

(Student)Add/Insert - Student setting up the web application for the first time, the students will submit[major, year, pending semesters, and completed courses)

(Student)Update - When changing their completed semester or degree major we update the database with [year, degree name]

(Student)Delete - When student removes an old completed courses or deletes their major we remove the major associated with the student, removing pending semesters

(Student)Add/Insert - When building future semesters based on their previous semester and pending courses, we insert new data to database with the following: [term, year, max_credits]

(Student)Add/Insert - When the student fills the future planning, they submit[course_id, semsester_id](this is blocked if any of the prerequisites are not met)

(Student)Update - When tuning their search for various courses the student submits[avg_gpa_min, avg_gpa_max]

Admin(Add/Insert) - When adding new courses or refreshing courses we submit[subject, number, title, credits, description, avg_gpa, offered_terms]

Admin(Update/Delete) - When correcting an existing course we submit[subject, number, title, credits, description, avg_gpa, offered_terms]

Low fidelity UI mockup:

https://drive.google.com/file/d/1f8zpV_cXJC-J7w-5WRFqZKf6IdDVUMcH/view?usp=sharing

Teamwork Distribution:

As seen by the below breakdown, some tasks are dependent on the successful completion of Task 1. Therefore, we will follow this distribution of work:

Task 1 dependent tasks Team: Leisha, Bavya

Non Task 1 dependent tasks Team: Ritsika, Rikhita

1. **TASK 1**: Dataset Creation and Cleaning:
    a. Build cleaned dataset mapping each course to its prerequisites/allowed concurrent enrollment
        i. Assigned Person: Leisha
        ii. Backend Tasks: creating dataset and relational model to store information
    b. Build dataset mapping each major to its degree requirements and courses fulfilling it
        i. Assigned Person: Bavya
        ii. Backend Tasks: parsing section info string, storing prerequisite information by course

—----------

2. Onboarding flow
    a. Allow users to provide initial information:

i. Non Task 1 Dependent
    1. Major
        a. Assigned Person: Ritsika
        b. Backend Tasks: update user table with major
    2. Courses completed
        a. Assigned Person: Rikhita
        b. Backend Tasks: update user table with inputted courses, pull course list from dataset
    3. Semesters remaining
        a. Assigned Person: Ritsika
        b. Backend Tasks: update user table with selected semesters

3. Course Views
    a. Tabular view of courses
        i. Non Task 1 Dependent
            1. Columns
                a. Course Code
                b. Course Name
                c. Description
                d. GPA
            2. Filter and Sort
                a. GPA
            3. Assigned Person: Ritsika
            4. Backend Tasks: pull information from Course Catalog Data, calculate gpa for each course, correspond with Course Catalog Data, build filtering logic by gpa, build sorting logic by GPA
        ii. Task 1 Dependent
            1. Columns:
                a. Semester Eligibility
                    i. If eligible, state semester eligibility
                    ii. Else state, not eligible
                b. Filter and Sort
                    i. Filter out courses “not eligible”
                    ii. Filter for courses eligible for given semester
                    iii. Sort courses by semester eligibility
            2. Assigned Person: Leisha
            3. Backend Tasks: pull courses taken by student, calculate eligibility, building filtering based on calculation
4. Semester Planner
    a. Non task 1 dependent
        i. Information

            1. View of planned courses
            2. Information on credit hour thresholds
        ii. Assigned Person: Ritsika
        iii. Backend Tasks: pull courses added by student, pull credit hour information and calculate if student is within threshold
5. Course Information Page
    a. Non Task 1 Dependent
        i. Teacher breakdown of GPA
            1. Assigned Person: Rikhita
            2. Backend Tasks: pull and aggregate data in gpa dataset
    b. Task 1 Dependent
        i. Prerequisites
            1. Assigned Person: Bavya
            2. Backend Tasks: Pull from table
        ii. Concurrent Enrollment
            1. Assigned Person: Bavya
            2. Backend Tasks: pull from table
        iii. Semester Eligibility
            1. Assigned Person: Bavya
            2. Backend Tasks: Pull calculation from tabular view
        iv. Degree Requirement Fulfillment
            1. Assigned Person: Bavya
            2. Backend Tasks: pull from table

We expect this list of assigned tasks to change based on discovered workload. However, this serves to initially guide us.