Screen Sketches

Group 112 - Duc Khoi Pham, Thanh Mai, Nhat Anh Bui, Kim Sang Huynh

I. Project description

The COM S Schedule is an app that offers a better way for Computer Science students to view, schedule, and plan their degree in a more effective and streamlined way. The main functionality involves automatically building an academic plan from students' inputs (e.g. their degree audit, preferred path, courses, personal goals, etc.); we also offer a convenient toolkit for students to access information, plan courses, schedule classes, among many other features. While our scope for this project only covers the COM S program of study, we aim to develop in such a way as to leave options open for future expansion and upgrades.

II. Actors

A. Students

The students are the main group of users. They can access their scheduled plan, make a new schedule, and access the course lists to look for information on courses. When making a new schedule, they have 2 options: manually input or input degree audit. If they choose to input the degree audit, they will input a PDF of the degree audit. If they choose another option, they will have to answer some questions.

B. Staff

Staff users are allowed to create or modify certain information in the database. This includes but is not limited to: course information, course schedules, degree requirements, descriptions, notes, etc. To create a staff account, there must be additional manual verification carried out by administrators.

C. Administrators

The administrators are the ones who maintain the system and answer access requests (staff). The admin will be able to have access to features of both students and staff. They also have access to logs,

D. Guests

Guests are mostly identical students, but their information is only temporarily stored for as long as the session is active. Some features aren't available, such as preferences or saving plans.

III. Nonfunctional requirements

A. Security

Data security is one of our most important concerns. Authentication is handled by session JWT tokens. The password will be salted and hashed with bcrypt. Additionally, all sensitive communication and operations (credentials, SQL calls) will be done solely by the backend application to minimize security risks and help defend against malicious user actions (e.g. SQL injections, session hijacking, etc.).

B. Performance

While this application isn't time-sensitive in nature, and no components require any real time or instantaneous update operations; we aim for a response time no slower than that of loading a website. The most time consuming operation will likely be the processing of a degree audit and

related computations (e.g. offering recommendations), in which case we set the upper bound to be 1 minute. We will also try to avoid sluggish UI actions, or silent errors.

C. Maintainability

The application follows the MVC design pattern, one of the standard patterns in application development. Extra care is being taken to ensure conformity with well-established OOP principles (encapsulation, modularity, etc.). This greatly facilitates present and future development efforts and minimizes effort spent on debugging and refactoring.

D. Usability

The premise of the application is straightforward and users will already know what they want. The flow of the application will bear resemblance to what students already do to plan their degree; we extend upon this to provide a more streamlined and reliable option for students. Our UI will mostly mirror what students are already familiar with, both intuitively, visually, and functionally; with some novel additions.

E. Portability

The scope of this project only involves an Android application for the frontend. However, the specific platform used by the front end shouldn't be coupled with the rest of the software stack, and theoretically, the API/database backend can support any frontend platform.

IV. Data model

A. `User`: stores authentication and basic information

`uid`: primary key
`username`: username, primary key
`email`: email used for login, unique
`uid`: primary key
`privilege_level`: id for privilege (i.e. type of user)
`pwd_bcrypt_hash`: bcrypt hash value for password
`pwd_bcrypt_salt`: salt value for bcrypt hash

B. `UserPreferences`: stores preferences for each user

`uid`: primary key, foreign key one-to-one reference to `User`
<any configuration options we want to offer>

C. `Student`: stores information relevant to students

`uid`: primary key, foreign key one-to-one reference to `User`
`first_name`: first name
`last_name`: last name
`primary_major`: primary major, this is subtly different from other majors a student may have; there are situations where the primary major may have different ramifications for a student compared to their secondary or tertiary majors.

D. `StudentProgram`: associates students and programs (majors/minors)

`uid`: primary key, foreign key one-to-one reference to `Student`
`program`: program identifier (e.g. COM S; S E; MATH;...)
`type`: major, minor, certificate
`path`: some programs offer different "paths" a student can follow within a major

E. `Course`: stores information about courses

`id`: course id, primary key

`department`: the program identifier (e.g. COM S; S E; MATH;...)
`number`: course number
`display_name`: course name (e.g. Software Development Practices)
`credits`: credit hours

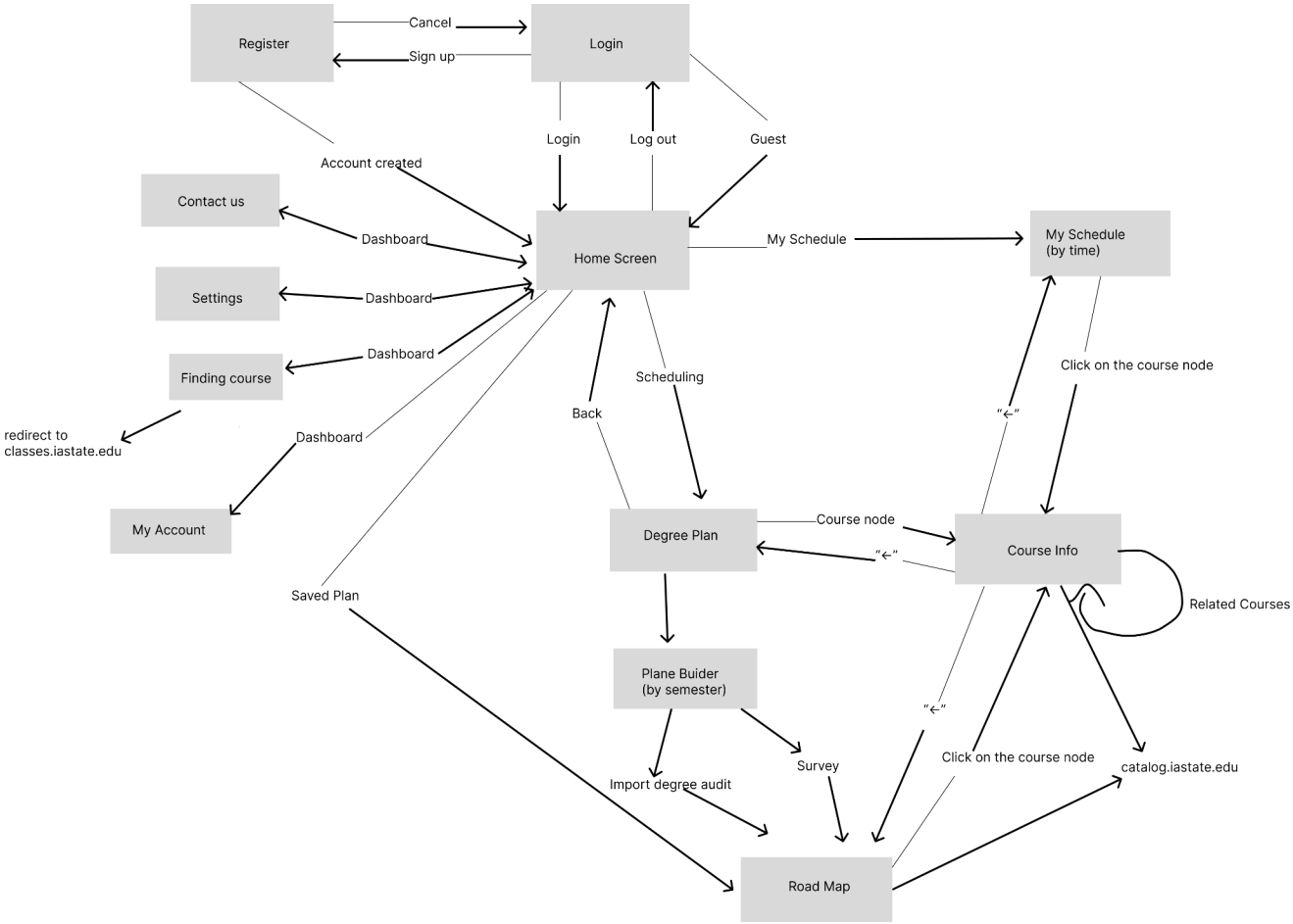
F. `CourseSchedule`: stores time schedules of courses in every semester
`id`: generated id, primary key
`course_id`: course id, foreign key many-to-one reference to `Course`
`year`: the year of this semester
`season`: spring, summer, fall, winter (+ full, first half, second half)
`start_time`, `end_time`: start and end times;
`is_online`: whether it's an online class

G. `Privilege`: stores information about courses
`privilege_level`: primary key, 0, 1, 2, 3, for guest, student, staff, and admin
`display_name`: "Student", "Staff", "Administrator"
<booleans on whether they have permissions (read, update, manage, etc.)>

H. `Prerequisite`: stores prerequisites of courses
`course_id`: course id,
`department`: the department identifier (e.g. COM S; S E; MATH;...)
`number`: course number (e.g. 309)
`display_name`: course name (e.g. Software Development Practices)

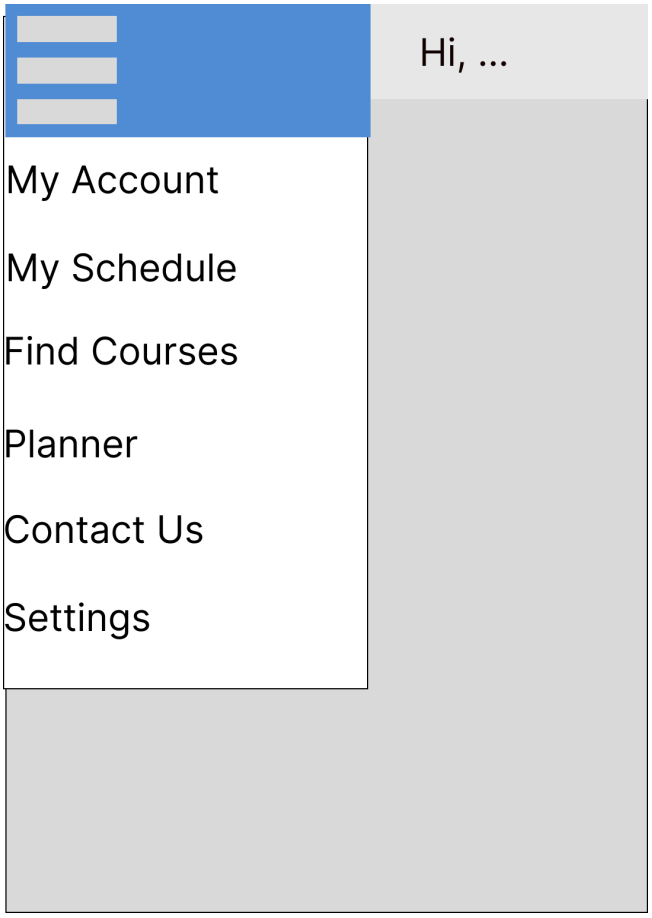
I. `DegreeAudit`: stores degree audits
`id`: generated id, primary key
`uid`: foreign key reference to `Student`
`data`: BLOB object containing the HTML or PDF of the degree audit file
`is_parsed`: whether we have finished parsing the file

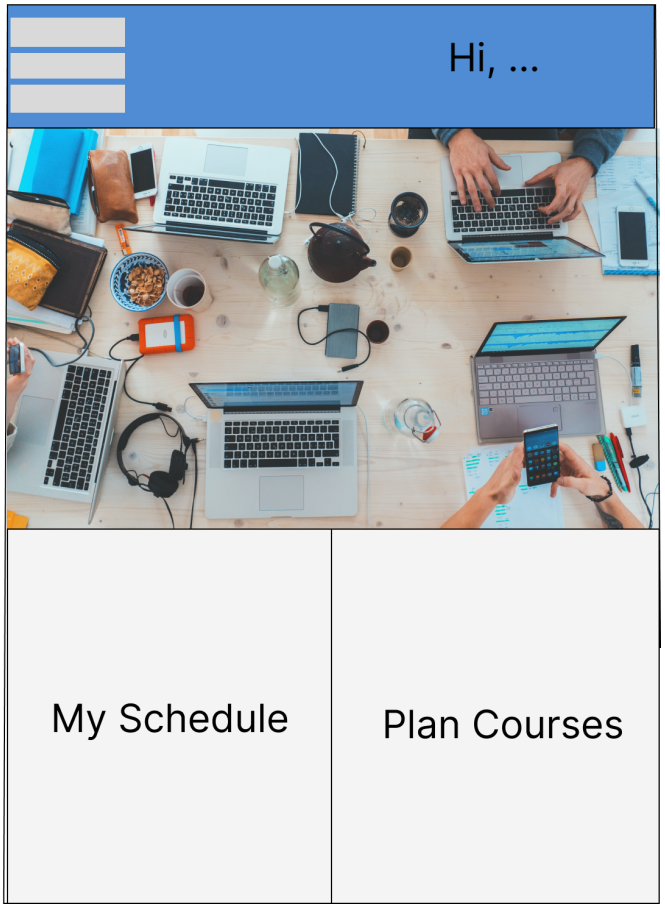
V. Screen flow



VI. Screen sketches

Sketch 1, 2 - Dashboard/Home screen(2 Screens) - Khoi



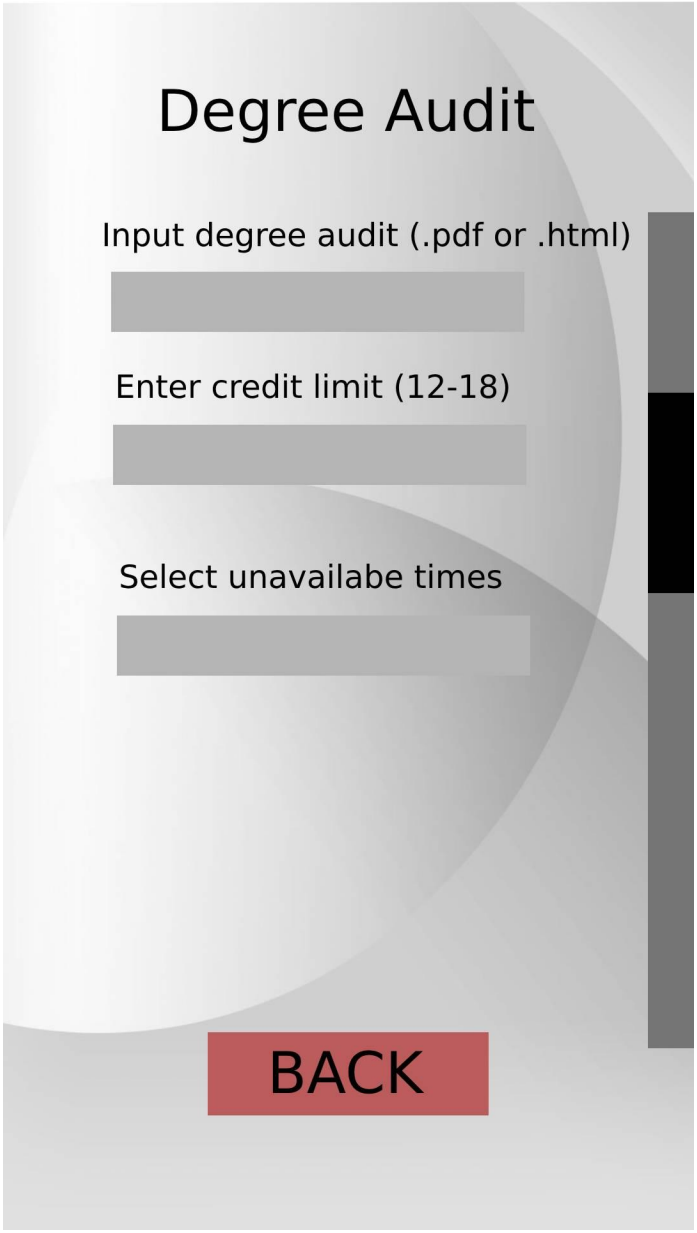


- The DashBoard will be the brain of the App, where you can direct to features. When you click the Top Left Symbol with 3 stripes. It will show the list of features that will direct you to the features you want. There are 6 directions: My account- To see your account information, My schedule- To see your current semester schedule, Find courses- To find the courses that you want, Planner- To build a plan for you, it will show your academic info first then you can continue, Contact us- To contact us for technical issues or being granted staff access, Settings- To change your settings.

- When you click anywhere outside the DashBoard, you will return to the home screen.

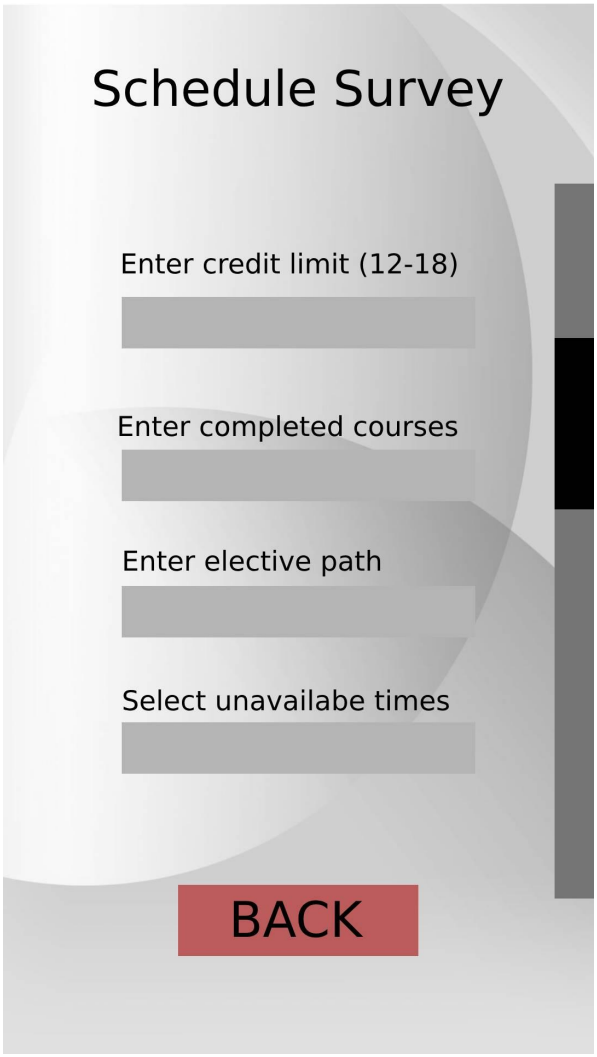
- The Home Screen will have 2 main features: My Schedule and Plan Courses. When clicking on the symbol on the top left, you can access all the offered features that have been shown.

Sketch 3 - Degree audit screen -Nhat Anh



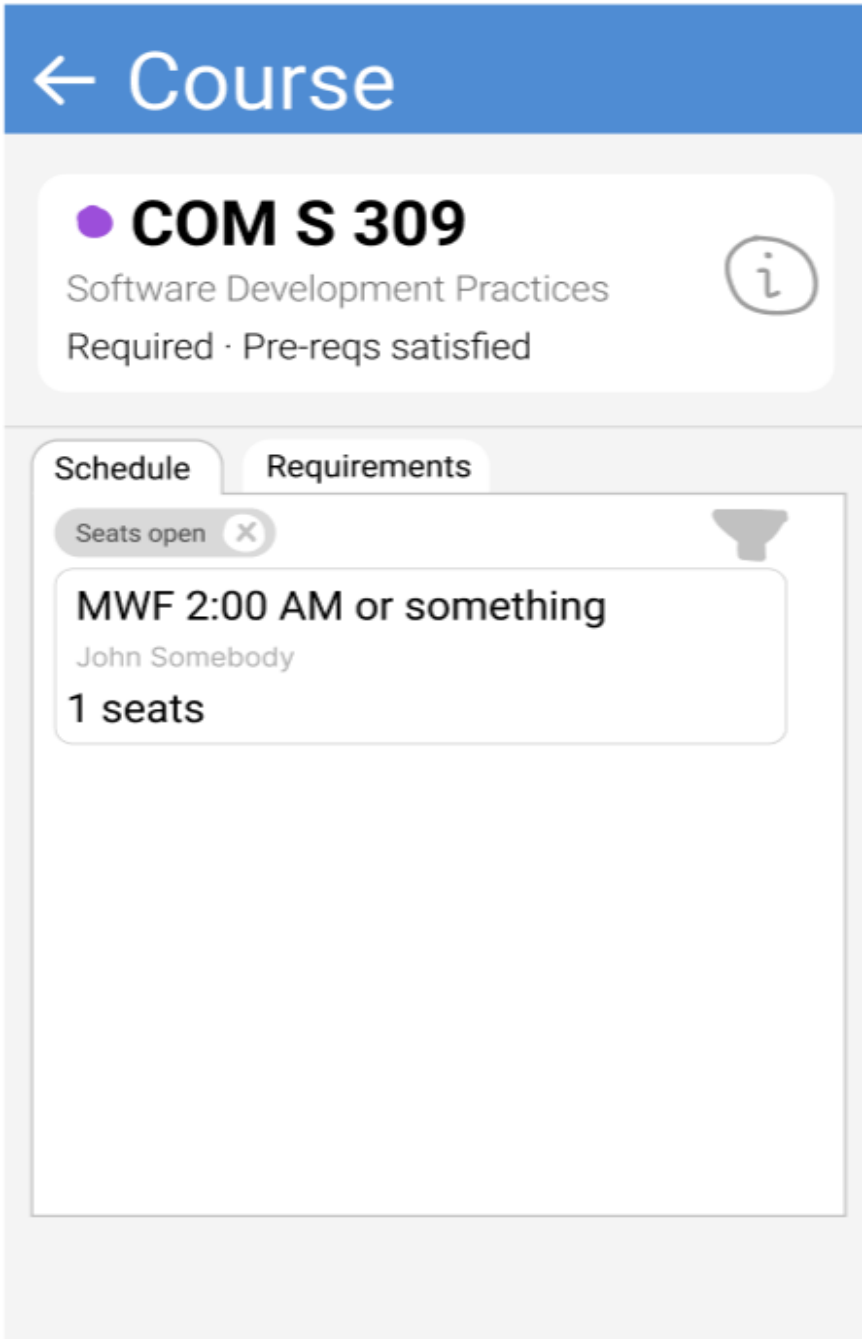
-Users can upload their degree audit in either format. A parser will automatically store information about the courses and plan accordingly
-Further information such as workload, in terms of credit limit, and unavailable time for work, activities etc… is also factored in

Sketch 4 - Survey screen - Nhat Anh



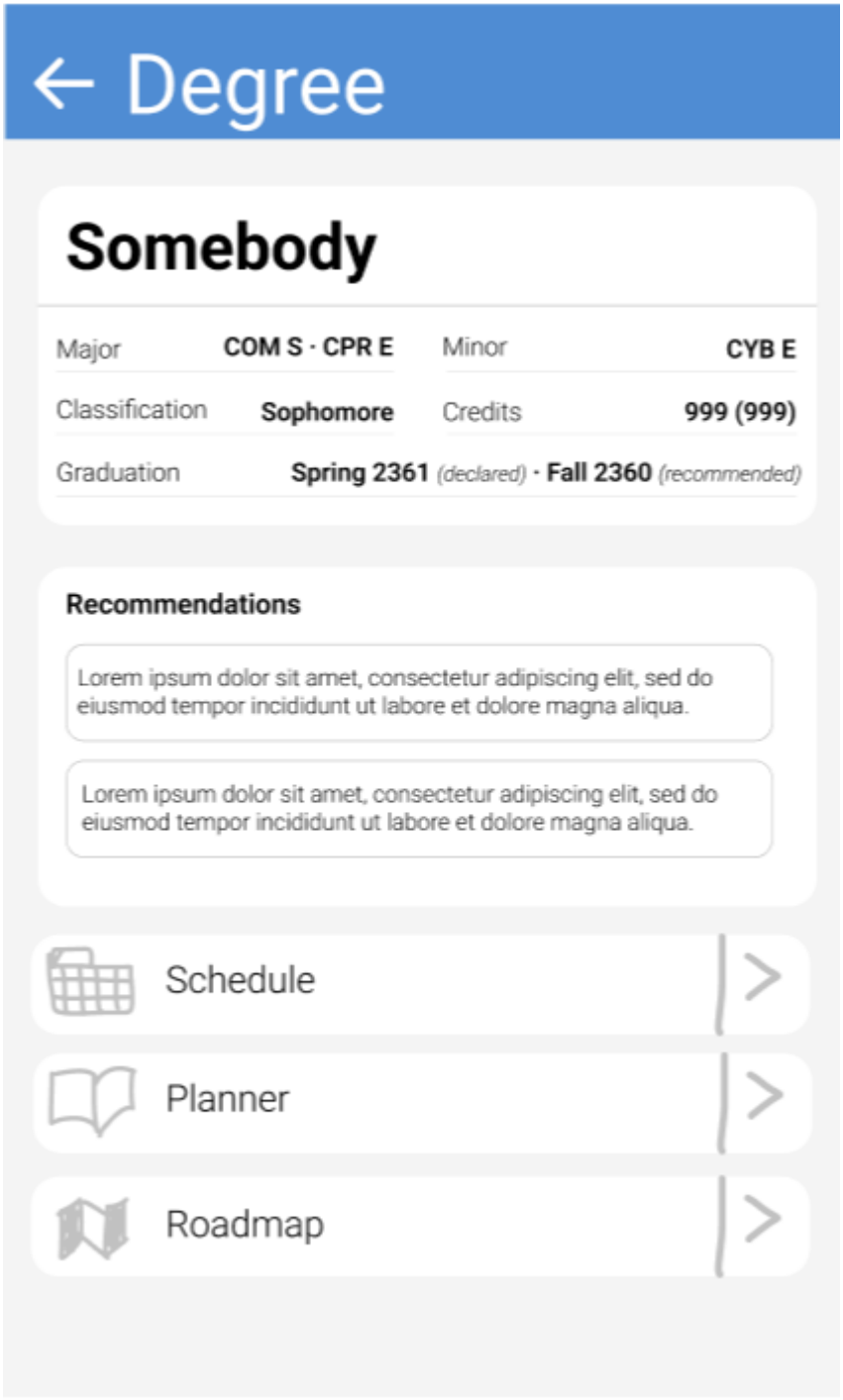
-This screen will ask for a detailed survey about student: completed courses and timings to plan accordingly
-Users can instead skip this step by uploading the degree audit and filling out just the timings

Sketch 5 - Course information screen - Thanh Mai



- This screen displays course information: name, status, time schedule, prerequisites, lecturer, as well as any other information that we may want to include. The data comes from classes.iastate.edu. Most actions here will either redirect to classes.iastate.edu, the course's entry on the university catalog, or back to the planner.
- The schedule tab has time frames for the next term, but can be configured in the filters menu, which gives additional options for filtering time frames. The requirements tab lists the prerequisites of this course, clicking on any goes to that course's information page
- We can also show other student's comments/discussions about the class, if applicable.

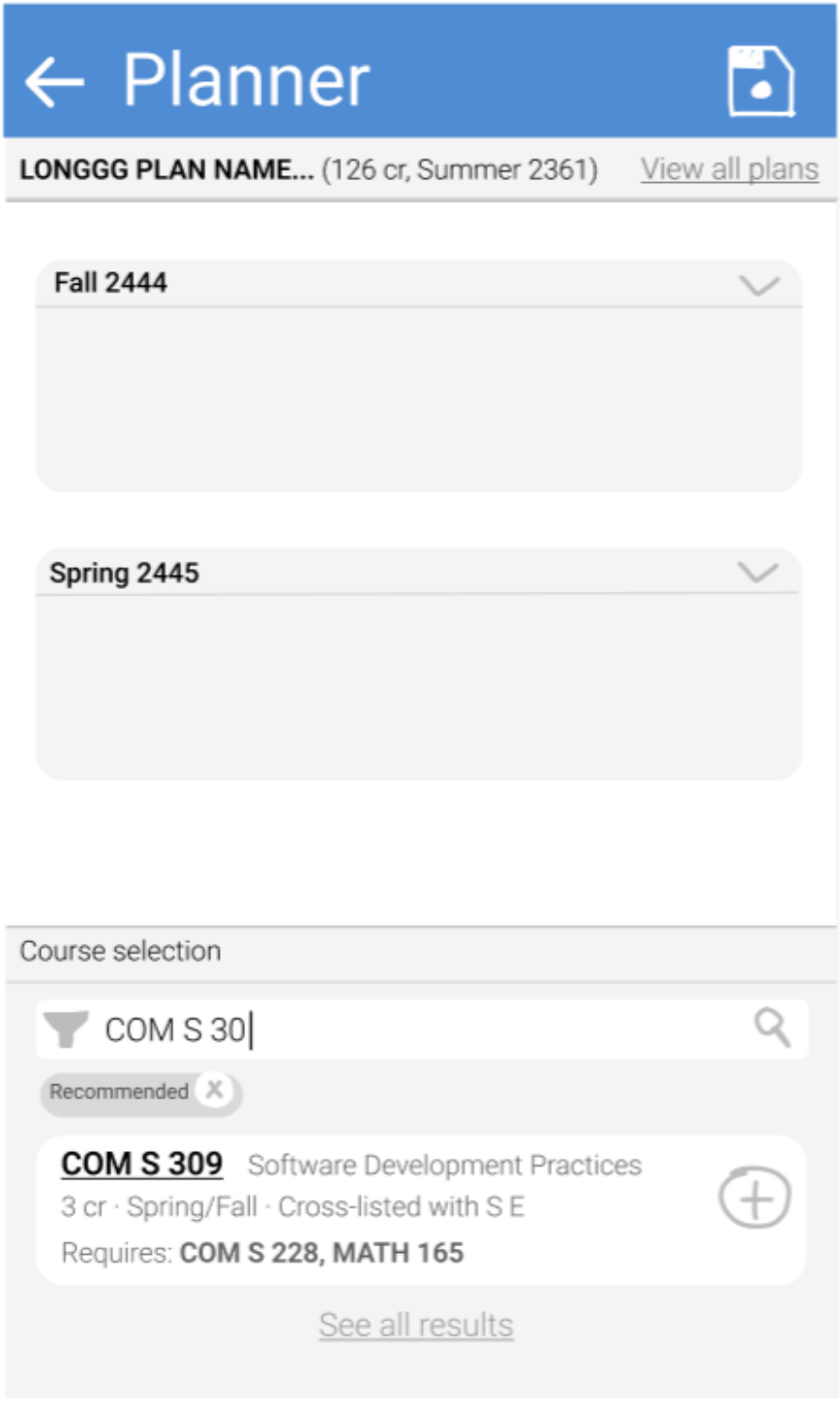
Sketch 6 - Student profile/degree progress screen - Thanh Mai



- This screen displays the student's information: majors, minors, etc.
- It also gives short recommendations for planning, clicking on these brings up either the planner, or the schedule.
- There are also quick links to other parts of the application.

Sketch 7 - Degree planner view - Thanh Mai

- The degree planner screen is one of the more complicated. We have a list of saved plans that the user can access. Pressing the save button saves a new plan.
- The main modal has a list of collapsible accordions, each containing some courses (the exact visuals I haven't decided upon yet, partly because I'm working on the backend, partly because it is quite a complicated component)
- The lower modal is the course selector, it has a search bar, a filter function.
- Tapping on a course in the selector brings up a pop up, allowing the user to decide which term to add the course to. tapping on the course identifier leads to the course information page.
- I would also like a link to the roadmap, but this is a future decision.



Sketch 8 - Roadmap view - <your name>

Sketch 9 - Settings menu - <your name>