PathFinder

1. Team:

Student Name	Student Email	GitHub Username	Student's role
Abdoulfatah Abdillahi	aabdillahi@mail.sfsu.edu	Abdoul1996	Team Lead / Full-Stack
Bryan Escobar	bescobar5@sfsu.edu	Alex3096	Front-End Developer / UX Lead
Omshree Bharodiya	obharodiya@sfsu.edu	OmshreeBharodiya	Back-End Developer / Database Lead
			QA & Research Analyst

2. Problem Statement & Motivation:

- Many CS students, especially first-generation college students, struggle with course planning. Some don't even know that academic advisors are available, while others face limited appointment times, short sessions, or confusion about prerequisites. As a result, they often rely on peers for guidance or make course choices that delay graduation.
- This issue particularly affects freshmen, sophomores, transfer students, and first-year graduate students. The consequences include delayed graduation, financial strain, mismatched courses, and self-doubt, which can even lead students to switch majors or drop out.
- By providing clear, personalized course recommendations, our app will help students match their skill level and interests, satisfy degree requirements, and plan schedules with confidence.

3. Target Users & Context:

- The primary users are **first- and second-year CS students**, **transfer students**, and **first-year graduate students** at SFSU who need guidance in selecting courses and understanding prerequisites.
- The app will be used in **personal academic settings** such as at home, in the library, or while commuting. It will be accessible on both **laptops and mobile devices** with an internet connection, making course planning flexible and convenient.

- Students often struggle with confusing prerequisites and poor course sequencing, which can delay graduation. They need clear, personalized recommendations that balance workload, align with career goals, and keep them on track to graduate on time.

4. Market Research:

Competitor Analysis (existing solutions & limitations)

- University degree planners: helpful for checking requirements but often rigid, provide no personalization, and don't explain why a course is a good next step.
- Generic schedule builders (term planners that assemble timetables): good for time-slot management, but typically ignore prerequisite logic and degree-progress constraints.
- Course- and professor-review sites: surface subjective ratings, yet don't connect opinions to a student's goals or eligibility; information is fragmented across multiple sources.
- Coursewise (reference example): focuses on data-driven guidance; however, tools in this space still tend to lack transparent explanations tied to official degree rules, and may not be tailored to a specific program's prerequisite graph.

User Pain Points (current coping strategies & frustrations)

- Students **piece together information** from catalogs, friends, Discords, and rating sites, which is **time-consuming** and **inconsistent**.
- Difficulty **understanding prerequisites and sequencing**, leading to accidental ineligibility or suboptimal order of courses.
- **Low confidence** when choosing electives; unsure how choices align with **interests, workload tolerance, and graduation timeline**.
- **Advisor access is limited** during peak periods; short appointments don't always resolve planning uncertainty.

Industry/Domain Trends

- Growing demand for **personalized, explainable recommendations** in education technology.
- Increased comfort with **conversational interfaces** (chatbots) to simplify complex decisions.
- Greater emphasis on **transparency and privacy**—students expect to know *why* a suggestion is made and how their data is handled.

Evidence & References

- Example product in this space:
  - **Coursewise**
- University advising pages for degree rules and prerequisites:
  - CS-Degree-Underg
  - DS&AI-Degree-Grad
  - CS-Degree-Grad

- Planned primary research: **10 student interviews** (freshmen, sophomore, transfers, grad) and a short survey to quantify pain points (confidence, time spent planning, errors due to prerequisites).

5. Proposed Solution / Concept:

High-level description of the app:

- Our proposed solution is a **web application** designed to help CS students choose the right courses each term. The app encodes official degree rules and prerequisites, checks which courses each student is eligible to take, and ranks those options based on individual goals such as academic interests, workload tolerance, and timeline to graduation. A conversational chatbot interface provides transparent explanations of why each recommendation is made, citing catalog rules, the student's preferences, and (with consent) aggregated patterns from past CS students. The goal is to reduce confusion, increase confidence in course planning, and support timely degree completion.
- Our application will help students by providing smart course suggestions, ensuring a balanced schedule, and recommending the best courses they can take together.

Core features and functionalities:

- Encodes prerequisites, co-requisites, and degree requirements as a **graph/constraint model**.
- Surfaces **eligible next courses** instantly based on completed and ongoing coursework.
- Provides a **personalized ranking questionnaire** (covering interests, workload preferences, math comfort, etc.).
- Offers a **chat interface** that explains recommendations and answers student questions in natural language.
- Includes **optional indicators** of professor ratings and workload patterns, sourced ethically from publicly available data.

Chosen platform and rationale:

- The app will be developed as a **web application**. A web-based solution was chosen for its accessibility, as students can use it on laptops or mobile browsers without installing additional software. It also enables rapid iteration, easier deployment, and seamless integration with backend services and APIs, making it the most suitable platform for this project.

6. Design Process (User-Centered Design):

- We will apply a **User-Centered Design (UCD)** approach to ensure the app is built around the real needs of CS students.

- **User Research:** We plan to conduct 10 interviews with new CS students and transfer students at SFSU. These conversations will help us understand their challenges with course planning, advisor access, and prerequisites.
- **Ideation:** Based on research findings, we will create user personas that represent typical students, such as a freshman navigating prerequisites, a transfer student aligning past credits, and an upper-division student exploring electives. Personas will guide brainstorming sessions, storyboards, and design sketches.
- **Low-Fidelity Prototypes:** We will develop initial wireframes of the course planner and chatbot to visualize workflows and gather early feedback.
- **High-Fidelity Prototypes:** Next, we will create interactive mockups using **Figma** and build a more functional prototype in **React** to simulate the user experience.
- **Evaluation & Iteration:** Usability testing, heuristic evaluation, and A/B testing will be performed to identify pain points. The app will be refined iteratively until it effectively meets user needs, improves course-planning confidence, and aligns with advising goals.

7. Technical Feasibility:

Usability testing methods (SUS, task completion, A/B, think-aloud):

- The proposed application will be developed as a **web-based platform**. The front end will be implemented using **React** for an interactive and responsive user interface, while the back end will be built with **FastAPI or Flask** to manage logic and API communication. A **PostgreSQL or SQLite database** will store course data, prerequisite structures, and user progress.
- The core system logic will encode **degree requirements and prerequisites as a graph structure**, allowing the app to determine course eligibility dynamically. In addition, we plan to integrate a **GPT-based API** (e.g., OpenAI's GPT models) to power the **chatbot interface**. This model will generate conversational explanations of course recommendations, answer student questions about prerequisites, and provide guidance in natural language.

How findings feed into design iteration:

- The conclusions of these tests will directly inform the iterative design process. Insights into user frustrations, errors, or unclear features will be used to refine the interface, improve chatbot explanations, and adjust the recommendation logic to better meet student needs.

8. Evaluation Plan:

How does the app improve everyday life?

- The app enhances everyday life by streamlining the course planning process and alleviating stress for students. Because it is a web-based platform, it can be accessed from anywhere, making academic planning more convenient and reliable.

What’s innovative/different compared to existing solutions?

- What makes this solution innovative compared to existing tools is its ability to combine degree rules with personalized student preferences while also offering an explainable chatbot for guidance. Unlike static planners that only provide rigid course checklists, this app is dynamic, interactive, and transparent—adapting to each student’s needs and clearly explaining the reasoning behind its recommendations.

9. Expected Contributions :

- We used AI tools in limited ways to support the development of this proposal. ChatGPT was used to assist with brainstorming project ideas, structuring the proposal outline, and refining the clarity of written sections. The final content was reviewed, edited, and approved by all team members. No AI-generated text was included without human oversight.

10. Disclosure & Contributions:

- All members contributed collaboratively and in real time. We worked together during group Zoom sessions, discussing ideas, drafting content, and refining sections simultaneously. Each team member provided input across multiple parts of the proposal, and decisions were made collectively. This project proposal reflects the shared effort and equal participation of the entire team.

11. Google Site:

- This is our google site where you can find our homepage and project proposal !! please read it and enjoy 🙂
  - Link : PathFinder
  - GitHub: pathFinder