Will Gillette, Jess Sokolski, Kacey La, Ryan Fitzgerald, Matt Quigley

Dr. Mongan

CS-375: Software Engineering

31 January 2023

Student Access Portal Revamp: Degree Planner Proposal

Ursinus College seeks to provide a liberal arts education, teaching students the necessary tools and skills to excel in many different types of fields. With this mission in mind, the College offers a variety of programs and degrees and thus a plethora of courses. A substantial drawback of this, however, is that it becomes increasingly more challenging to maintain all these courses with the release of new programs or the development of current ones. The College's account management portal, the student access portal, enables students to register for courses, assign courses to future semesters, and view their progress towards core curriculum and degree requirements. Although, there are a lot of underlying issues that make this a frustrating and unproductive process for users, the most prominent being the obscurity of the course assignment feature. The feature enabling students to plan out their degrees is buried deep within the portal, causing many people to be unaware of its existence. Furthermore, the small number of students that have used this system have reported that there are numerous issues with it, such as how it allows them to add courses that are unavailable in certain semesters. Aware that the current system makes it a hassle to effectively plan out one's degree, our mission is to create a more efficient system that resolves all these issues, adds additional features, and is more user-friendly. By creating a scalable architecture that can support a large dataset of courses and effectively

maintain their information, we hope to streamline the current system and ultimately provide students a more productive experience when planning out their degrees. In addition, we look forward to seeing how this tool influences communications between students and advisors since both parties will be more likely to stay on the same page regarding career aspirations. Due to our novel ideas for new features and the tedious user experience with the current system, there is a strong compelling need for our system.

To gather information about user experiences with the current system, we plan to communicate with a variety of stakeholders: Ursinus students, faculty, advisors, and the registrar. With a sample of students, we hope to receive feedback and details about issues on the front-end of the current system so that we can work to resolve them when building our system. On the other hand, faculty and advisors can provide us information regarding what features they use on their end of the system, making us knowledgeable about features we need to implement or any necessary changes. Finally, the registrar has a regularly active administrative role within the system due to having to input final grades and add and drop courses on student accounts, so we hope to gain information about the features on their end and their overall experience as well.

Our development team consists of Will Gillette, Jess Sokolski, Kacey La, Ryan Fitzgerald, and Matt Quigley. We hope to apply our diverse skillsets to many distinct aspects of this project. For instance, a couple of us are knowledgeable about database schemas and backend development, while the remainder of us are experienced with front-end development using HTML, Javascript, and CSS. Kacey will conduct unit testing to ensure that our system remains scalable and the user experience remains smooth. Will and Matt will first create a blueprint for the database schema and then implement the data storage unit on the backend of the application. Ryan will initially spend time getting familiar with React.js before developing the components

for the application's pages, while Jess will learn the fundamentals of CSS to design the elements of our application and create its mobile view. In addition to these individual roles, we hope to come together to help each other progress on others' respective tasks each week. Ultimately, we aim to apply our individual skills to this project while working collectively as a unit and frequently communicating with one another.

Essentially, our system will revamp the student access portal's current features as well as introduce novel ones that facilitate the degree planning process. To fulfill the basic needs of this project, our web application must contain three pages: the landing page, the degree progress page, and the course selection page. Our initial priority will be to gain familiarity with the development tools so that we can begin crafting the landing page, where we will begin implementing user authentication, enabling users to save the states of their degree planner. As a subset of team members begin designing the landing page, the remainder of us will start setting up the backend of the web application, which will display our web pages on the client and handle user requests, data storage, and database queries. After implementing our database schema on the backend, we hope to finalize our work on the frontend and begin conducting some user testing on our development ecosystem. Following our success with this, we plan to focus our efforts on the degree progress page, displaying students' progress towards their majors and core curriculum requirements. Unlike the student access portal, however, we intend to implement functionality such that users can manually choose the core curriculum category that a course fulfills. The current system automatically assigns courses to a random one of their respective categories, causing frustration for users, so we hope that our prototype remedies that problem. After designing the view for the degree progress page, we plan to continue on to the course selection page, allowing students to filter the course catalog and assign courses to semesters in which they

are available. As addressed previously, the current system has a myriad of issues with this, primarily due to the massive size of the course dataset and the lack of maintenance as a result. On the backend, we hope to develop a scalable model that facilitates course updates, enabling administrators to ensure that the application's course catalog remains consistent with Ursinus' official course catalog. After successfully implementing these features, we will have satisfied the needs of the minimum project scope. Below is a tentative timeline outlining our project development plan:

Week Of:	Tasks to Complete:
1/31	➢ Familiarize ourselves with the relevant web development tools and technologies (Javascript, CSS/Tailwind, Bootstrap, React.js, Express.js, Node.js, SQLite/NoSQL, Github Actions, Figma) ➢ Request access to the semester course data excel spreadsheet to understand how Ursinus formats its course data
2/7	➢ **Group 1 (Server Side Crew):** ○ Begin designing a blueprint for the database schema ○ Set up the backend server for the web application **(Node.js/Express)** ○ Decide on a database language while being mindful of storage size ○ Begin implementing the user information database table and any other related database tables **(SQLite/NoSQL?)** ➢ **Group 2 (Client Side Crew):** ○ Create a React.js project ○ Set up and begin designing the landing page **(CSS/tailwind?)** ○ Create a modal form component for user authentication **(React.js)** ○ Implement the Thunk middleware that sends a request to the backend server **(Javascript)**
2/14	➢ **Group 1 (Server Side Crew):** ○ Construct an API endpoint that creates a database entry for new users **(Express.js, SQLite/NoSQL?)** ○ Help out Group 2 as needed ➢ **Group 2 (Client Side Crew):** ○ Finish setting up the middleware if that is not complete

	(Javascript) ○ Set up the React router **(Javascript)** ○ Create another modal form component for user registration, requesting information such as: **First Name, Last Name, Major (dropdown), Anticipated Graduation Date (dropdown) (React.js)** ○ Upon submitting the form, send a request through the middleware to create a new database entry for the user on the server **(Javascript)**
2/21	➢ **Group 1 (Server Side Crew):** ○ Conduct user testing with user registration and authentication ○ Begin constructing the database tables for course data **(SQLite/NoSQL?)** ➢ **Group 2 (Client Side Crew):** ○ Finish designing the landing and registration pages **(React.js, CSS/tailwind?)**
2/28	➢ **Group 1 (Server Side Crew):** ○ Create an API endpoint that fetches course progress data for specified users **(Express.js, SQLite/NoSQL??)** ○ Help out Group 2 as needed ➢ **Group 2 (Client Side Crew):** ○ Begin designing the progress page **(React.js, CSS/tailwind?)** ○ Upon being directed to the progress page, send a request to fetch the user info and course data from the server and store it in the component properties
3/7	➢ **Spring Break:** ○ Both groups conduct user testing on progress page and handle any necessary bug fixes
3/14	➢ **Group 1 (Server Side Crew):** ○ Add an API endpoint that allows the client to update the categories that a student's courses are fulfilling (see pt. 2 of Group 2 tasks) **(Express.js, NoSQL/SQLite?)** ○ Help out Group 2 as needed ➢ **Group 2 (Client Side Crew):** ○ Create a core curriculum category component, and mount one for each course category **(React.js)** ○ Begin implementing the functionality that initially randomly assigns a course to a random one of their respective categories and allows users to drag and drop courses to other categories **(Javascript??)**

	○ Finish designing the progress page **(CSS/Tailwind?, React.js)**
3/21	➢ **Group 1 (Server Side Crew):** ○ Create an API endpoint that receives the course ID and respective semester ID and updates the user's list of courses for the given semester **(Express.js, NoSQL/SQLite??)**. Validate that the course is available before updating database. ○ Help out client side crew as needed ➢ **Group 2 (Client Side Crew):** ○ Create a component that stores course information: course name, category, pre-requisites, credit amount, etc. **(React.js)** ○ Create a semester schedule component that stores the course information components as sub-components **(React.js)** ○ Create an add course button at the bottom of the page, displaying a search page modal component that allows users to search and select courses **(React.js)** ○ Begin designing the course search modal component and handle course selection. Only display courses that are offered in the specified semester. **(CSS/Tailwind?, Javascript)**
3/28	➢ **Group 1 (Server Side Crew):** ○ Make any necessary optimizations to the database tables (normalization is important) ➢ **Group 2 (Client Side Crew):** ○ Add the functionality that allows users to add selected courses to their course list in the respective semester. Be sure to validate that the course is available on the client as well. ○ Conduct user testing for the degree planner page
4/4	➢ **Everyone:** ○ Redesign the view for mobile users by using media queries
4/11	➢ **Time Permitting:** ○ Begin paving the path for any administrative features or aspirant tasks ➢ Otherwise, work on any remaining tasks listed above
4/18	➢ **Everyone:** ○ Conduct final user testing ○ Flesh out any aspects of the client and server

	○ Work on any bug fixes or optimizations ○ Start the project presentation
4/25	➢ Finish the project presentation

Beyond improving the functionality of monitoring degree progress and searching for classes, creating an admin page would be the next big step. This is especially important for the registrar as they need to be able to add and remove students from different courses at the beginning of the semester and make changes to courses as necessary. Another feature that can especially benefit students and advisors is the course recommendation to help students pick classes for their major, minors, and Core Requirements. This improvement allows students to pick their classes with more ease, as students no longer have to search the course catalog for classes that fulfill their requirement. A simple recommendation system might be displaying all the available classes next semester that fulfill a part of their major, minor, or Core Requirements. Following the completion of our minimum viable scope and given any remaining time, we hope to pave the path for the development of these new features.

Currently, there is a platform for which students and advisors can monitor progress and plan a schedule based on the student's major. However, there are issues within the platform that hinder accessibility and usability. The cumulation of these drawbacks results in a lack of fluidity when attempting to plan a degree plan. Our project will redesign the current student portal being used into a cohesive directory and degree planning site that will spark improved discussions with advisors as well as relieve students of stress that comes with planning a degree. Creating a portal with applications that are more visible upon site entry will result in students feeling more confident when checking progress or planning for future semesters. In addition, improved sorting

tools will enable students and faculty to ensure that all major requirements as well as the College's requirements will be finished upon graduation. This project is being built to model how Ursinus students and faculty will utilize the planning sites, but this concept could easily be compatible with other universities. The improvements being made will save the students and advisors time because the new progress tracker will improve clarity on when courses are being offered or if they are still offered. This will in theory save time when meeting with advisors and create room for more meaningful in-person discussion about paths to take to achieve proper completion. Beyond Ursinus, we look forward to seeing how our system impacts the lives of other schools and communities.