College of Engineering

CS Capstone Requirements Document

October 25, 2019

OSU CS Applied Plan Portal

Prepared for

Oregon State University

Dr. Rob Hess _____ Signature _____ Date

Prepared by

Group CS72
The Portal Team

Claire Cahill _____ Signature _____ Date

Jackson Golletz _____ Signature _____ Date

Phi Luu _____ Signature _____ Date

Zachary Thomas _____ Signature _____ Date

Abstract

This document outlines the client requirements for the OSU CS Applied Plan Portal project. This web application will allow students to create custom applied plans, EECS advisors to provide input on proposed custom applied plans, and the EECS Head Advisor to approve plans. This document ensures that both the client and the Portal Team agree on what the final product should achieve and provides methods to verify that the product adheres to these requirements.

CONTENTS

1 INTRODUCTION

The OSU CS Applied Plan Portal aims to be a full-featured, easy-to-use, and aesthetically appealing web application. This application will allow students to create custom applied plans, EECS advisors to provide input on proposed custom applied plans, and the EECS Head Advisor to approve plans.

1.1 Purpose

This document outlines the agreed upon requirements for the OSU CS Applied Plan Portal project.

1.2 Scope

This document describes the complete scope of the OSU CS Applied Plan Portal project. This includes both the front-end and back-end of this web application.

- **Front-end (Web Client):** This is the part of the application that the user interacts with. It is primarily composed of HTML, CSS, and JavaScript. The front-end is further divided into a student interface and an advisor interface.
- **Back-end:** This includes the server, the database, and the parts of the application that help to support the front-end.

1.3 Product overview

1.3.1 Product perspective

The user interface should be easy to navigate on both standard monitors and standard mobile devices. Users should also be able to clearly read text and see all relevant elements on any modern screen. Interactive elements must be easy to select either with a mouse or by touch screen. The most common display ratios that this application must work with are 16:9, 9:16, 8:5, 5:4, 4:3, 3:5, 3:4, and 2:3.

1.3.2 Product functions

The OSU CS Applied Plan Portal has two different user interfaces. The user interfaces that is used depends on if the user is a student or an EECS advisor.

- **Student View:** When a student logs into their account they are recognized as a student are brought to the student homepage.
  - **Homepage:** From the homepage students are able to create new applied plans, view their custom applied plans, and see if they have received any new messages.
  - **Create Plan:** When creating a new plan students select which courses they would like to include in their plan. Invalid courses are filtered out, attempting to select an invalid course will show an error explaining why the course is not applicable. Users may also view detailed information about specific courses before making a selection.
  - **View Plan:** Once a student has created at least one plan they can can view that plan and its status. When viewing a plan students can see any comments that an advisor has left on their plan, they can leave their own comments, modify the courses in their plan, or see the current status of their plan. Each plan will have one of the following statuses: ”Awaiting Initial Review”, ”Awaiting Final Review”, ”Awaiting Student Changes”, ”Rejected”, ”Accepted”.

  - **Direct Message:** Students are able to directly message any of the EECS advisors. Students are also able to see their message history at any time.

- **Advisor View:** When an advisor logs into their account they are recognized as an advisor and they interact with the advisor homepage.

  - **Homepage:** From the homepage advisors are able to see all of the student plans that they are currently working on, as well as notifications for any new messages that they have received. There is also an option to search for plans.
  - **Search:** Advisors can search for plans by any of the statuses: ”Awaiting Initial Review”, ”Awaiting Final Review”, ”Awaiting Student Changes”, ”Rejected”, or ”Accepted”. Advisors can also search for a plan by a students name or student ID number.
  - **View Plan:** When an advisor views a plan they are able to leave comments, mark specific courses as requiring a change, edit the plan, or change the plans status (only the head advisor can change the status of a plan to/from Accepted or Rejected). When an advisor views the plan they can also see if a similar plan (same collection of courses) has been accepted or rejected previously.
  - **Direct Message:** Advisors are able to directly message any of the EECS advisors or any student. Advisors are also able to see their message history at any time.

1.3.3 User characteristics

There are two main user groups.

- **CS Students:** These are undergraduate students that are comfortable with software and technology. While these students are unlikely to have issues with navigating a web application, they may have limited understanding of the courses and plans that offered to CS students.
- **EECS Advisors:** These are employees of OSU that advise CS students. These advisors are likely to have a firm understanding of software, technology, and the courses and plans that are offered to CS students at OSU.

1.3.4 Limitations

Since the OSU CS Applied Plan Portal will accept and store student data it is critical that FERPA laws are adhered to and that student privacy is a primary focus of this application. Additionally, since we plan to have students test our web application we will need to get approval from the IRB to conduct user testing. The IRB will help ensure that we are conducting humane research and that we are not violating the Common Rule (45 CFR 46), which is put in place by the Department of Health and Human Services.

2 REFERENCES

3 SPECIFIC REQUIREMENTS

This section will provide in-depth descriptions of what the application shall do and how it works to achieve a better user experience for EECS advisors and students.

3.1 External interfaces

External interfaces of the application are its inputs and outputs. The web application shall support the following external interfaces:

1) **Inputs**

- **Mouse events:** This type of input allows the users to navigate to various sections on the web page. The source of this type of inputs can be either a hardware pointing device (primarily on desktops) or the touching and dragging actions by the users (primarily on mobiles). The website shall have pixel-level input accuracy, though it should have a comfortable layout to tolerate a margin of error in user input.
- **Keyboard events:** This type of input allows the users to navigate throughout the web page as long as to submit written responses or text search requests. The source of this type of inputs can be either a hardware keyboard (primarily on desktops) or a software on-screen keyboard (mainly on mobiles).

2) **Outputs**

- **Visual outputs:** This type of output is the primary source of output from the application. Therefore, the application shall have fast response time and shall display the mouse cursor and/or text cursor so the users can know where they are on the web page.

3.2 Functions

The application shall perform the following fundamental actions to process the inputs and produce the output:

1) **Login/Logout/Registration**

- The application shall perform an input validation, including, but not limited to, null-checks on required fields, duplicate checks, and unmatched passwords checks, and shall reject any bad or malicious request.
- The application shall use OSU APIs to compare the provided credentials with existing credentials in the database and determine the validity of the login/logout/registration process.
- If the login/registration process was successful, the application shall redirect the user to the dashboard. Otherwise, the application shall inform the user of the issues and redirect back to the login/registration page.
- If the logout process was successful, the application shall log the user out and redirect to the login page.
- Edge cases are handled in the input validation step, and since this shall be a secured operation, any abnormal request shall be rejected.

2) **Dealing with permissions based on roles**

- The application shall have a role for each type of users. Only administrators shall have permissions to manage the roles.
- The application shall configure the roles such that a student can have access only to his/her Applied plan, and an advisor can have access only to the Applied plans of students they are advising.
- An edge case would be a user having multiple permissions. To prevent this, each user shall only have one role.

- The application shall output a “Forbidden Error” (code 403) response whenever users try to access something they should not.

3) **Creating Applied plans**

- The application shall ensure that the current user is a student or an administrator, otherwise shall output a “Forbidden Error” (code 403) response and stop the operation.
- When a student sends a request to create an Applied plan, the application shall receive it and validate the request. As per the usual rule, the application shall stop the operation and inform the user of the issue if the validator rejects the request.
- The application shall perform a sequence of steps to collect the student’s response to create a draft for his/her Applied plan. This process is quite complicated depending on the type of the student’s Applied plan. What needs to be included in this sequence of steps will be continually discussed and optimized by the EECS advisor that the team works with.
- After a successful creation, the system shall update the student’s profile and show the new plan on his/her dashboard.
- Any failure in the process shall make no changes to the system but redirect the user back to the dashboard and include a feedback message instead.

4) **Viewing Applied plans**

- The application shall ensure that the current user is a student or an advisor or an administrator, otherwise shall output a “Forbidden Error” (code 403) response and stop the operation.
- The application shall allow a student to view his/her—and only his/her—Applied plan.
- The application shall allow advisors to view their students’ Applied plans and give feedback on it.
- When the user sends a request to view an Applied plan, the application shall perform validation. If validation is successful, the system shall fetch the requested Applied plan from the database and display it on the screen.
- An edge case in this operation is a student having no Applied plan. In this case, the application shall display a ”No Applied plan” message on the dashboard.
- Any failure in the process shall make no changes to the system but redirect the user back to the dashboard and include a feedback message instead.

5) **Searching Applied plans**

- The application shall ensure that the current user is an advisor or an administrator, otherwise shall output a “Forbidden Error” (code 403) response and stop the operation.
- The application shall allow advisors to search for their students’ Applied plans and perform a viewing action on it.
- When the user sends a request to view an Applied plan, the application shall perform validation. If validation is successful, the system shall search for Applied plans from the database based on the search query and display the result on the screen.
- An edge case in this operation is an empty search result. In this case, the application shall display a ”No Applied plan” message on the dashboard.

- Any failure in the process shall make no changes to the system but redirect the user back to the dashboard and include a feedback message instead.

6) **Retrieves student information**

- The application shall ensure that the current user is an advisor or an administrator, otherwise shall output a "Forbidden Error" (code 403) response and stop the operation.
- The application shall allow an advisor to view students' Applied plans and give feedback on them.
- When the user sends a request to view a student's Applied plan, the application shall perform permission checks on the user and then try to retrieve the student's information from the database.
- Any failure in the process shall make no changes to the system but redirect the user back to the dashboard and include a feedback message instead.

7) **Accepting and rejecting Applied plans**

- The application shall ensure that the current user is an advisor or an administrator, otherwise shall output a "Forbidden Error" (code 403) response and stop the operation.
- The application shall allow advisors to make a decision to accept or reject a student's Applied plan draft.
- When the user accepts or rejects an Applied plan, the system shall make changes to the student's record to update the plan result.
- Any failure in the process shall make no changes to the system but redirect the user back to the dashboard and include a feedback message instead.

3.3 Usability requirements

One of the most important attributes this application must have is usability. To ensure that the application is easy to use, it must satisfy the following qualities:

1) **Learnability:** Students and advisors should take at most 3 hours to learn how to use the external interfaces of the website. All components shall have articulate names and possibly descriptions of what they do.
2) **Friendliness:** The user interface's design shall follow popular design concepts that have been known for creating a friendly environment to the users. The application shall be as interactive as possible and shall give feedback on every operation the user performs.
3) **Productivity:** The application shall surpass the current web-form planner in terms of productivity—the average time spent to build an Applied plan shall be considerably shorter, and the average frustration shall be much less (or even better, none).
4) **Manageability:** The application shall keep track of every student's Applied plan. It shall save all old drafts of a plan and allow the students to continue building the latest version of their plans.

3.4 Performance requirements

- The application shall be scalable and be able to simultaneously serve at least 300 users.
- 95% of the responses shall be less than 3 seconds after the requests are sent.
- 95% of the responses from the back-end shall be asynchronous—meaning the responses will be displayed on the front-end without the user having to reload the web page.

- The average time it takes for a student to complete his/her Applied plan shall be approximately twice faster than that of the current web-form planner.

3.5 Logical database requirements

To retrieve and update information of different entities in the system, the application must store information in relational databases, as shown with the following entities:

1) **Student:** Includes student ID number, first name, last name, date of birth, contact information, major(s), their preferred advisor(s), their Applied plan draft(s), grades, course history, etc. The information about a student is used frequently to determine who are on track on their Applied plan. This data shall be accessible only to the students themselves and to the advisors. Students shall have a many-to-many relationship with advisors, meaning a student can have many advisors, and an advisor can advise multiple students.
2) **Advisor:** Includes advisor ONID, first name, last name, contact information, specialization(s), list of students they are advising, etc. The information about an advisor is used quite often to help students build their Applied plans. This data shall be accessible publicly depending on the property of this entity (e.g. students they are advising should only be visible to the system and the administrators). Advisors shall have a many-to-many relationship with students, as explained in the paragraph above.
3) **Plan:** Includes the corresponding student's ID, plan progress, related drafts, reviewing advisor(s), etc. The information about an Applied plan is used quite often to help the student and the advisor(s) succeed in building the plan. This data shall be accessible only by the student and his/her advisor(s). Plans shall have a many-to-one relationship with students, as a student can build multiple Applied plan, but an Applied plan is associated to only a single student.

All data shall be FERPA-compliant and shall be stored in a stable database management system (DBMS) and shall have backups in case of emergency. Finally, note that the entities and their properties as well as the relationships between the entities are subject to change as the team sees fit as the project moves forward.

3.6 Design constraints

We are restricted to create an interface for desktop PCs and laptops. We are restricted by FERPA to protect the privacy of student records when we are developing this portal. We have to monitor the way we handle data as to not interfere with any of the privacy laws.

3.7 Software system attributes

The portal will include the following attributes in order to ensure the system is user-friendly and maintainable.

1) Reliability - Once it is launched, the system should be available through the EECS website and should be maintained in good quality.
2) Availability - This system should be made available on the OSU EECS website so both students and advisors can access the portal.
3) Security - The portal shall use OSU ONID authentication with the two step Duo login to manage student accounts.

4) Maintainability - The system will be maintained by this team until it is passed off to the EECS advisors and Rob Hess.The portal team will conduct sufficient testing of the final system before passing it off to the advising team.
5) Portability - This portal will be made available to desktop PCs and laptops. If time allows, it will also be accessible through mobile devices.

3.8 Supporting information

This portal solution solves the problem of creating a custom applied plan at Oregon State is complicated and time consuming. The current system is clunky, inefficient, and non-interactive. This will make it easier for students to create their plan and track their progress toward graduation.

4 VERIFICATION

4.1 External Interfaces

As a web application, our product will likely be used with mouse and keyboard in the majority of cases, but we should also take care to opportunistically consider the site's accessibility to impaired users who, for instance, might navigate entirely with the keyboard or while using a screen reader. With this in mind, verification of our external interfaces will overlap considerably with our usability verification.

4.2 Functions

Our testing of functionality will be largely automated, using a system like Jest.js. For a larger, multi-platform application like ours, we believe it is much better to use a testing library instead of unreliable and laborious human testing.

4.3 Usability Requirements

At this point in time, we are unsure of exactly how our product's usability will be evaluated. During our initial client meeting, Dr. Hess proposed possibly conducting randomly sampled, randomly assigned studies to compare students' opinions of the existing form-based solution to those of our replacement. This was the only concrete guidance we received with respect to any form of client evaluation, and was not identified by the client as a requirement, per se. Dr. Hess also noted that in order to conduct such studies, we would need to present our product and testing methodology to Oregon State University's Institutional Review Board (IRB) for approval.

4.4 Performance requirements

Thus far, the client has not yet identified any performance requirements for the project. As a result, we have no ways to outline this type of validation.

4.5 Logical database requirements

We will be leveraging Oregon State University's API significantly to gather data. Because the data we are handling is at least partially confidential student data, FERPA regulations may play a major role in the design and implementation of our database. The client has not given us any further requirements beyond these two considerations, and we have not designed a database on our own at this early stage.

4.6 Design constraints

During our initial meeting, the client suggested that we focus solely on laptop and desktop screen sizes when developing our user interface. We have no other constraints against which to evaluate our solution at this time.

4.7 Software system attributes

We will decide on specifics in terms of availability and reliability as the project proceeds. The robustness of our security will likely be governed by FERPA and any applicable organizational policies.

4.8 Supporting information

We have no additional information to provide at this early stage in the project.

5 APPENDICES

5.1 Assumptions and dependencies interfaces

5.2 Acronyms and abbreviations

- **API:** Application Programming Interface.
- **CFR:** Code of Federal Regulations.
- **CS:** Computer Science.
- **CSS:** Cascading Style Sheets.
- **DBMS:** Database Management System
- **EECS:** Electrical Engineering and Computer Science.
- **FERPA:** Family Educational Rights and Privacy Act.
- **HTML:** Hypertext Markup Language.
- **IRB:** Institutional Review Board.
- **OSU:** Oregon State University.
- **ONID:** Oregon State Network ID.

6 Project Schedule