Transfer Pathways Tool

DESIGN DOCUMENT

Team Number: 20
Client: Susie DeMoss
Adviser: Dr. Ashraf Gaffar

Team Members/Roles:
Cameron Brecount - Co-Lead on User Interface
Ben Greif - Lead on Testing/Frontend Developer
Curt Lengemann - Lead on Middleware/Database Components
Riess Radtke - Co-Lead on User Interface
Scott Thurston - Co-Lead on Frontend
Luke Turczynski - Lead on API Management/Database Design Development
Cole Weber - Co-Lead on Frontend

Team Email: sdmay22-20@iastate.edu
Team Website: http://sdmay22-20.sd.ece.iastate.edu/

Revised: November 2021/Version 1.0

Executive Summary

Development Standards & Practices Used

- SOLID Design Principles
  - These object-oriented principles will guide our software to use best practices and lead to cleaner, more reliable code. Although PHP is not fully object-oriented, it still allows for numerous object-oriented features such as classes and inheritance; so this standard applies.
- Testing Pyramid
  - The testing pyramid philosophy states that the number of system tests should be less than the number of integration tests, which themselves should be less than the number of unit tests, forming the shape of a pyramid. This standard will apply because this project will use all three methods of testing described in the standard, and it will ensure that our tests run efficiently and maximize test coverage.
- IEEE 610 - Standard Glossary of Software Engineering
  - This standard provides definitions for every software engineering term imaginable. We can refer to this standard to ensure efficient and effective communication within the team.
- IEEE 1016 - Software Design Description
  - Because this project involves creating a software design description, this standard will apply, and will ensure that the information displayed on the relevant diagrams is appropriately conveyed in an understandable fashion.
- IEEE 1233 - System Requirements
  - Because this standard details the methods relating to requirements engineering, this standard will apply to our project. This standard will help us identify the business needs of the students and Admissions staff by determining what is currently in place and how what is in place can be improved upon.

Summary of Requirements

1. Constraints
    - 1.1 The project shall be in the form of an interactive web application.
    - 1.2 The project shall be mobile-friendly.
    - 1.3 The project shall use the Iowa State website template.
    - 1.4 The project shall be able to interact with a Workday backend, or a mocked version.
    - 1.5 Account creation for prospective students shall follow the format of an admissions prospect record.
    - 1.6 The project shall be completed by the end of the 2022 spring semester.
    - 1.7 The project shall be completed within 1,000 person hours.

2. Functional Requirements
    - 2.1 Functional Requirements Relating to All Users
        - 2.1.1 The project shall accept prospective student and administrator (admissions or academic advising staff) account types.
        - 2.1.2 The project shall display different views depending on the account type the user is logged in as.
        - 2.1.3 All project tables containing credits shall display the total number of credits in that table.
    - 2.2 Functional Requirements Relating to Prospective Student Users
        - 2.2.1 The project shall accept as input transfer courses and grades received from other universities.
        - 2.2.2 The project shall use the inputted transfer courses to determine transferability at Iowa State.
        - 2.2.3 The project shall output a four-year plan of the intended major of the prospective student in table format based on the existing four-year plan for that major.
        - 2.2.4 The project shall output a four-year plan of the intended major of the prospective student in flowchart format without prerequisite information based on the existing four-year plan for that major.
        - 2.2.5 While either the flowchart or table view is active, it shall display transferred courses crossed off along with the course name at the student's prior institution.
        - 2.2.6 The prospective student user shall be able to download a .pdf file of the four-year plan of their intended major in table and flowchart format.

2.2.7 The exported .pdf files shall display the date of creation.
2.2.8 The project shall allow account creation for prospective students.
2.2.9 The project shall allow the linking of the inputted transfer courses and majors to a prospective student account.
2.2.10 The project shall allow for this linked data to be changed and deleted.
2.2.11 The project shall allow for guest prospective student users to use the application without creating or signing into an account.
2.2.12 The project shall display, at least once, a disclaimer saying that this is an unofficial evaluation of transfer credits.
2.2.13 The project shall allow all course links to link to more information about the specified course and four-year plan.
2.2.14 The project shall display other resources for prospective students.

2.3 Functional Requirements Relating to Administrator Users
2.3.1 The project shall allow administrator users to view data regarding individual prospective student users.
2.3.2 The project shall allow administrator users to view pertinent aggregate data regarding prospective student users.
2.3.3 The project shall allow administrator users to reach out via email to prospective student users.

3. Nonfunctional Requirements
3.1 The user interface shall be easy to navigate.
3.2 The project shall respond within one second to a four-year plan query.
3.3 The user shall be able to complete a session in under ten minutes.

Applicable Courses from Iowa State University Curriculum

- S E 319
- S E 329
- S E 339
- COM S 228
- COM S 309
- COM S 362
- COM S 363
- COM S 409

New Skills/Knowledge acquired that was not taught in courses

- PHP
- Laravel REST Framework
- Security Testing
- Interfacing with external APIs such as Workday and Okta
- Communication with a client
- Some team members had to learn:
    - Database Connections
    - REST APIs
    - Software Design Principles

Table of Contents

List of figures/tables/symbols/definitions

List of Figures

List of Tables

1 Team

1.1 Team members

1. Curt Lengemann
2. Cole Weber
3. Ben Greif
4. Luke Turczynski
5. Scott Thurston
6. Cameron Brecount
7. Riess Radtke

1.2 Required Skill Sets for Your Project

- UI Design - due to the fact the website should be user friendly for transfer students
    - Mocks ISU Template - due to the fact that they already have a style and a template set up and in use
- Website Design - due to the fact that the client wants a website
    - Mobile Friendly Design - due to the fact that the client has the constraint that the webapp should be mobile friendly
- Databases and Connections - due to the fact that data will be stored will be in a database
- Software Project Management - due to the fact that we will be managing a large, long term software project with many developers and stakeholders
- REST API - due to the fact that the frontend and middleware will be communicating with via a REST API
- Software Design Principles - due to the fact that the website needs to be modular to allow for future use and integration with Workday
- Communication - Due to the number of correspondents we will have.
    - Communication with team members
    - Communication with the TA
    - Communication with client
    - Communication with advisor
- Source Control (GitLab) - We will need a shared source control for the project to easily develop code

1.3 Skill Sets covered by the Team

- UI Design -
    - All
- Website Design -
    - All
- Databases and Connections -
    - Ben Greif
    - Curt Lengemann
    - Scott Thurston
- Software Project Management -
    - All

- REST API –
    - Luke Turczynski
    - Scott Thurston
    - Curt Lengemann
    - Cameron Brecount
- Software Design Principles –
    - Curt Lengemann
    - Cameron Brecount
- Communication –
    - All
- Source Control –
    - All

1.4 Project Management Style Adopted by the team

Waterfall with some Agile elements

1.5 Initial Project Management Roles

- Curt Lengemann - Database Engineer / Middleware Engineer
- Ben Greif - Test Engineer
- Scott Thurston - Frontend Engineer
- Luke Turczynski - Report Manager / API Engineer
- Cole Weber - Meeting Facilitator / Frontend Engineer
- Cameron Brecount - UI/Frontend Engineer
- Reiss Radtke - Meeting Scribe/UI

2 Introduction

The Transfer Pathways Tool is an effort to create a tool with a modern user interface and more features based off the Iowa State TRANSIT system [4].

2.1 Problem Statement

The ISU Office of Admissions has determined that the current user interface for the TRANSIT system [4] is lacking in functionality and usability. Currently, TRANSIT does not have the ability to view four-year plans, which is crucial in determining degree progress and class schedules. Additionally, TRANSIT's user interface is outdated, and its backend will no longer be supported in about two years when ISU transitions from UAchieve to Workday. We are solving this problem by creating a new, more user-friendly version of the TRANSIT system that will be able to interface with the new Workday backend. Called the Transfer Pathways Tool, this application will display a four-year plan of the intended major of the prospective student at Iowa State in both four-year plan tabular and flowchart format. While it will not display the degree audit view present in TRANSIT, its four-year plan view will arguably be more helpful to prospective students, as many users report that the degree audit view is confusing. Functionality for administrators will be expanded to include both individual and aggregate student data to allow for administrators to reach out to potential transfers.

2.2 Requirements & Constraints

1. Constraints
    - 1.1. The project shall be in the form of an interactive web application.
    - 1.2. The project shall be mobile-friendly.
        Rationale: From ISU Admissions, more and more users nowadays are accessing websites via mobile devices.
    - 1.3. The project shall use the Iowa State website template.
        Rationale: This is an official ISU application.
    - 1.4. The project shall be able to interact with a Workday backend, or a mocked version.
        Rationale: ISU Admissions is transferring to a Workday backend in two years, and our application needs to integrate with it.
    - 1.5. Account creation for prospective students shall follow the format of an admissions prospect record.
        Rationale: ISU Admissions wants the prospective accounts to be in their existing standard format.
    - 1.6. The project shall be completed by the end of the 2022 spring semester.
    - 1.7. The project shall be completed within 1,000 person hours.

2. Functional Requirements
    - 2.1. Functional Requirements Relating to All Users
        - 2.1.1. The project shall accept prospective student and administrator (admissions or academic advising staff) account types.
        - 2.1.2. The project shall display different views depending on the account type the user is logged in as.
            Rationale: Prospective student users should not be able to access administrator data.
        - 2.1.3. All project tables containing credits shall display the total number of credits in that table.
            Rationale: It is helpful to students to see totals, so they have an idea of how close they are to graduation

    - 2.2. Functional Requirements Relating to Prospective Student Users
        - 2.2.1. The project shall accept as input transfer courses and grades received from other universities.
        - 2.2.2. The project shall use the inputted transfer courses to determine transferability at Iowa State.
        - 2.2.3. The project shall output a four-year plan of the intended major of the prospective student in table format based on the existing four-year plan for that major.
        - 2.2.4. The project shall output a four-year plan of the intended major of the prospective student in flowchart format without prerequisite information based on the existing four-year plan for that major.
            Rationale: The flowchart is very helpful for more visual audiences.

2.2.5. While either the flowchart or table view is active, it shall display transferred courses crossed off along with the course name at the student's prior institution.

Rationale: The prior course name helps the prospective student understand which course successfully transferred, since they will be more familiar with that course than the ISU equivalent course.

2.2.6. The prospective student user shall be able to download a .pdf file of the four-year plan of their intended major in table and flowchart format.

2.2.7. The exported .pdf files shall display the date of creation.

Rationale: Our client is concerned that students could bring in out of date four-year plans and expect the same courses to transfer.

2.2.8. The project shall allow account creation for prospective students.

2.2.9. The project shall allow the linking of the inputted transfer courses and majors to a prospective student account.

Rationale: So that the user does not have to re-enter information.

2.2.10. The project shall allow for this linked data to be changed and deleted.

2.2.11. The project shall allow for guest prospective student users to use the application without creating or signing into an account.

Rationale: While the ISU Admissions staff would prefer that prospective students create an account, they realize that some might be discouraged from using the tool if they are forced to create an account.

2.2.12. The project shall display, at least once, a disclaimer saying that this is an unofficial evaluation of transfer credits.

2.2.13. The project shall allow all course links to link to more information about the specified course and four-year plan.

Rationale: So the prospective student can learn more about the course and four-year plan.

2.2.14. The project shall display other resources for prospective students.

2.3. Functional Requirements Relating to Administrator Users

2.3.1. The project shall allow administrator users to view data regarding individual prospective student users.

2.3.2. The project shall allow administrator users to view pertinent aggregate data regarding prospective student users.

2.3.3. The project shall allow administrator users to reach out via email to prospective student users.

Rationale: This will allow ISU Admissions staff members to answer any questions the prospective students might have and otherwise assist them.

3. Nonfunctional Requirements

3.1. The user interface shall be easy to navigate.

3.2. The project shall respond within one second to a four-year plan query.

3.3. The user shall be able to complete a session in under ten minutes.
Rationale: If the application takes too long to use, prospective students will be more likely to give up and leave.

2.3 Engineering Standards

- SOLID Design Principles
- These object-oriented principles will guide our software to use best practices and lead to cleaner, more reliable code. Although PHP is not fully object-oriented, it still allows for numerous object-oriented features such as classes and inheritance; so this standard applies.
- Testing Pyramid
    - The testing pyramid philosophy states that the number of system tests should be less than the number of integration tests, which themselves should be less than the number of unit tests, forming the shape of a pyramid. This standard will apply because this project will use all three methods of testing described in the standard, and it will ensure that our tests run efficiently and maximize test coverage.
- IEEE 610 - Standard Glossary of Software Engineering
    - This standard provides definitions for every software engineering term imaginable. We can refer to this standard to ensure efficient and effective communication within the team.
- IEEE 1016 - Software Design Description
    - Because this project involves creating a software design description, this standard will apply, and will ensure that the information displayed on the relevant diagrams is appropriately conveyed in an understandable fashion.
- IEEE 1233 - System Requirements
    - Because this standard details the methods relating to requirements engineering, this standard will apply to our project. This standard will help us identify the business needs of the students and Admissions staff by determining what is currently in place and how what is in place can be improved upon.

2.4 Intended Users and Uses

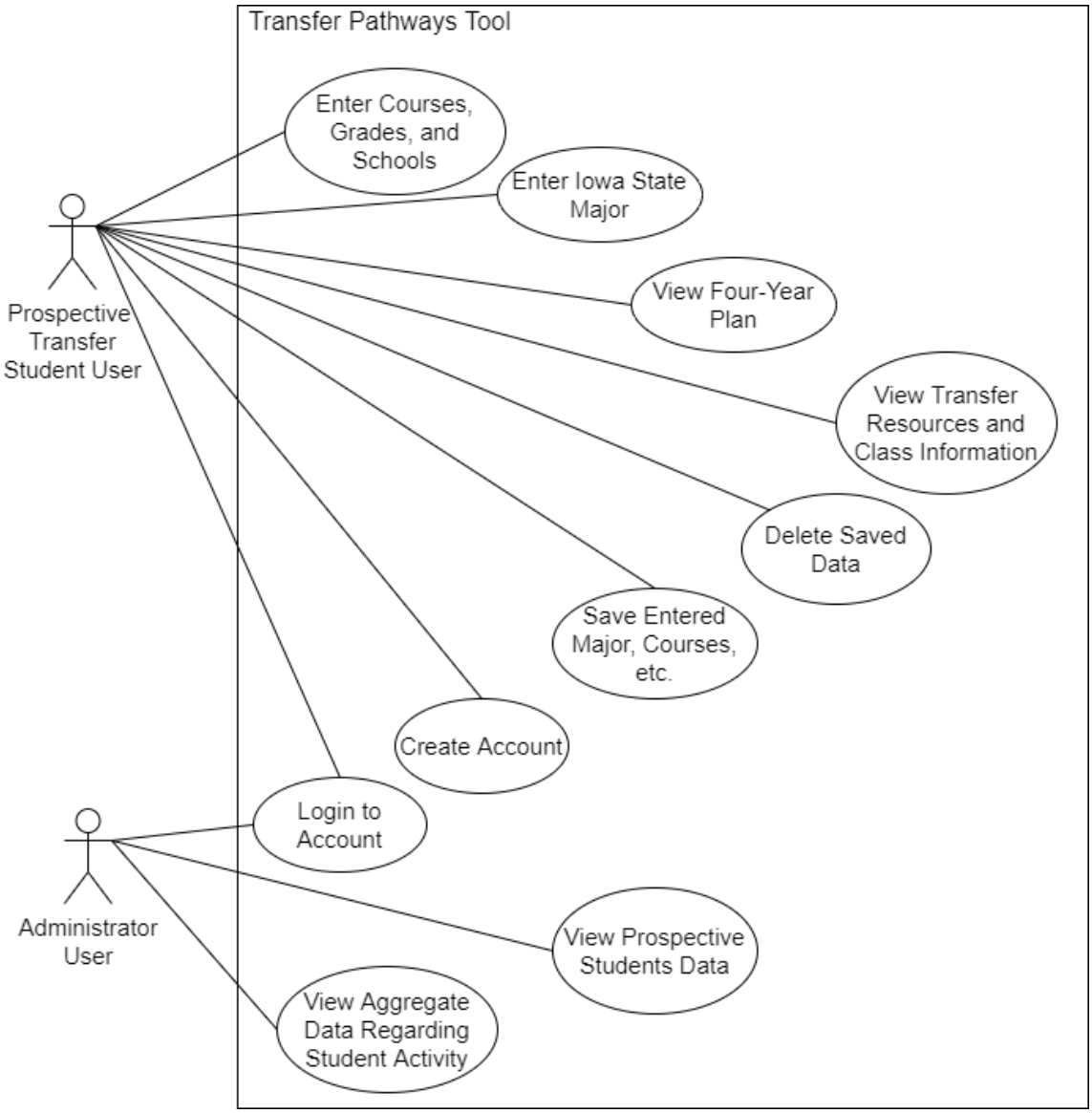

Figure 1 - Use Case Diagram

- Intended User: Prospective student with transfer credits
    - The user should be able to create an account.
    - The user should be able to use the application as a guest.
    - The user should be able to sign into their account.
    - The user should be able to enter their previous courses, grades, and schools.
    - The user should be able to enter their intended major at Iowa State.
    - The user should be able to view a four-year plan of that major with the classes that have successfully transferred crossed off in both a flowchart and tabular format.
    - The user should be able to save intended majors and previous courses, grades, and schools.
    - The user should be able to view, edit, and delete their saved data.

    - The user should be able to view more information regarding their four-year plan and individual classes on the four-year plan.
    - The user should be able to view external transfer resources.
- Persona: Jake
    - Let's imagine a user named Jake. Jake graduated high school one year ago and knew he wanted to go to college but was not sure what to study. So, he attended his local community college for a year to explore colleges and take general education courses and has now decided he wants a degree in Agricultural Engineering at ISU. Jake wants to know which of his general education courses will transfer so he goes to ISU's Transfer Pathway Tools to evaluate his courses. Jake first chooses to create an account so he can return to his information later, then he is then immediately directed to a page to input his courses. After filling in each of the courses he has taken, he selects agricultural engineering to see the number of credits that transfer. Out of curiosity he also selects aerospace engineering but sees that less of his credits transfer and decides to stick with agricultural and remove that major. He then selects the 'view' button to see more information on the flowchart and four-year plan of courses. Happy with the prospect of being an agricultural engineering student, Jake clicks a resource link on the side of the screen which takes him to the ISU's administration application page.
- Intended User: Administrator (ISU Admissions staff or ISU Academic Advisor) user
    - The admin should be able to sign into their account that has elevated rights.
    - The admin should be able to view specific prospective students' saved data.
    - The admin should be able to view aggregate data regarding prospective students.
- Persona: Mary
    - Now let us imagine an administrative user named Mary. Mary works in ISU admissions and is writing a report on interest in different majors from outside schools. She navigates to Transfer Pathway Tools and signs in. This is the first time Mary has visited Transfer Pathways, but she does not need to create an account because she has an account with Okta and is identified as an administrator. Mary is directed to a page with aggregate data where she investigates multiple different interactive graphs to filter by time and major to see traffic regarding prospective students. Happy with the quality of data she has found, Mary uses the data she has found to create an impressive report.
- Persona: John
    - John is another administrator working for ISU, but John is an academic advisor. Like Mary, John can sign in with Okta and when he does, he is directed to the aggregate student data page. From here he knows he is just looking to reach out to potential students, so he uses the navigation to go to the student data page. John knows he wants to reach out to students interested in agricultural engineering, so he queries by that major. He finds a potential student named Jake and clicks his email which opens his default email app so he can send Jake an email.

3 Project Plan

The hybrid (both waterfall and agile) project management method, along with a detailed breakdown of tasks regarding the frontend, middleware, and backend, was used to plan this project.

3.1 Project Management/Tracking Procedures

Our group is utilizing the waterfall project management model with some iterative elements of Agile. We chose this model because we felt like it is more conducive to the structure of the class overall. Our first goal for the semester was to complete a project plan and design document that are of a high quality and can be easily used as a roadmap for development of the project. We felt that we were better equipped to satisfy the goal of developing a quality project if we focused entirely on the project plan and design this semester, which constitutes the requirements and design phases of the waterfall method. Additionally, another goal was to develop relevant skills for the project; our team believes that it is better to develop skills during the requirements and design phases when we have more time than during an agile method with two-week sprints when we are actively trying to resolve issues. Finally, our last goal is to ensure our project will not only function but will also wholly satisfy the needs and desires of the client. We incorporated an iterative approach to design a smooth user experience. Iterating on our screen flow, UI design, and other user-close aspects of our project ensures that we mitigate the major risk of creating a poor user experience. With constant iteration and verification with our client, we are able to meet their needs and desires for this project.

During the first semester, we utilized Discord and Google Docs to track progress. Since we mainly worked in Google Drive for the project plan document and related tasks, it made sense for us to also track progress within the same software. To track progress during the development phase in the second semester, our group will utilize Gitlab issues. Each issue will be assigned a number of effort points, which will be utilized to ensure that all group members are assigned an equal amount of work. These issues will then be placed into an issue board so that the issues can be easily visible and categorized. Finally, Gitlab supports milestones as well; each issue will be tied to a larger milestone. Although Gitlab will be our main source of tracking progress, Discord will be used as well for informal information on progress. Additionally, we are using Gitlab for source control and concurrent development.

3.2 Task Decomposition

- Frontend
    - Create User Interface screen flow diagram
        - Design different components for each page
        - Articulate flow from one page to another
        - Verify UI design with Iowa State UI/UX domain experts
    - Create each individual UI component by picking high risk components first
        - Create UI for entering in transfer courses
        - Create UI for four-year plan table
        - Create UI for four-year plan flowchart - highest risk
        - Create UI for login - lowest risk
        - Create UI for prospective student account creation

        - Create UI for aggregate data screen
        - Create UI for viewing individual student data
    - Create Methods to Communicate with Middleware
- Middleware
    - Integrate JSON parser with Workday API
    - Create methods to follow business logic rules to process data
    - Set up Mocked Workday API
    - Create REST API for middleware
- Database
    - Set up database
    - Integrate database with Mocked Workday API

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

- Frontend
    - F1: UI Components
        - Look for components that are in Bootstrap 5 which can be used for all the functionality users will need to input their courses and output the four-year plans in an efficient UI/UX.
    - F2: Efficient Navigation
        - Be able to efficiently navigate to and from the entering courses page in less than 1 second.
    - F3: Screen Flow Interaction
        - Create screen flow diagrams that provide how each page will interact with each other.
    - F4: Middleware Integration
        - Connect frontend to the middleware using PHP and Laravel
- Middleware
    - M1: Retrieval Efficiency
        - Efficiently retrieving the four-year plans and the coursework in less than 1 second regardless of the number of courses entered.
    - M2: Mockday
        - Setting up a mocked Workday API that retrieves and stores mocked data in our database.
- Database
    - D1: Database with Workday
        - Setting up a database with mocked data in accordance with Workday, using MySQL.

3.4 Project Timeline/Schedule

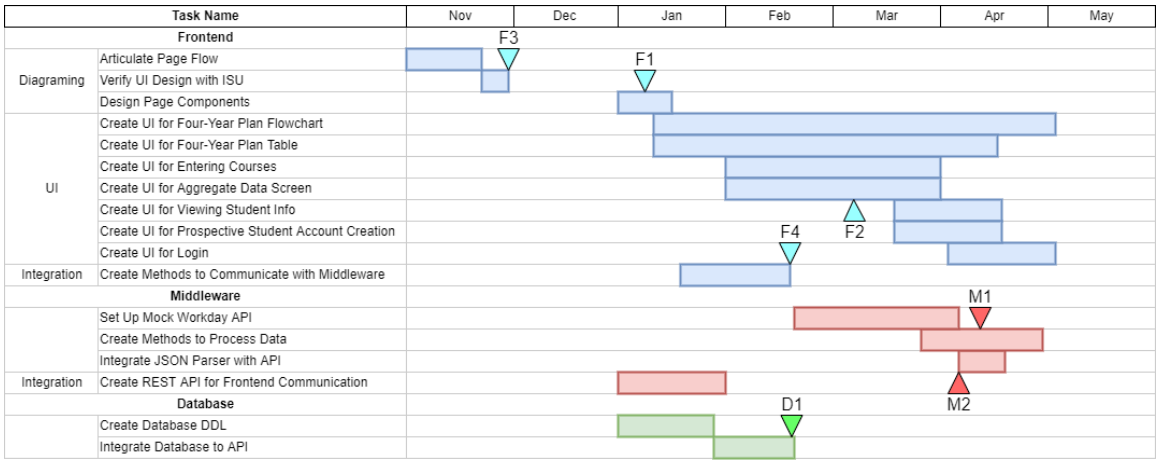

Figure 2 – Project Schedule Gantt Chart

The Gantt chart here is broken down into three major tasks: Frontend, Middleware, and Database, with the milestones indicated by the arrows. These major tasks each have their own subtasks - some of which belong to a group, like Diagramming, UI, and Integration. For this chart, we put the diagramming at the front because it is required for the rest of the Frontend work to be completed. The first semester is for documentation, while the second semester is for implementation. All tasks and subtasks are to be completed by May.

The tasks that are in parallel can be completed this way because none inherently rely on the others, and we have enough group members to accomplish each of these simultaneously.

3.5 Risks And Risk Management/Mitigation

Task	Risks & Probability (In parenthesis)	Mitigation
Create User Interface Screen Flow Diagram	None	None
Create each individual UI component by picking high risk components first	1. Components don't integrate well or make a cohesive experience (0.2)	1. Use PHP for all components to ensure component integration
Connect Frontend to Middleware	1. Frontend and middleware cannot communicate (0.1) 2. Cannot enforce communication speed is faster than a 1 second round trip time desired threshold (0.3)	1. Use a well-known REST API framework such as Laravel to make communication between subsystems easy 2. Research other subsystem communication methods and change to a more performant one

Verify UI design with Iowa State UI/UX domain experts	1. ISU rejects the UI design (0.2)	1. We will have the UI design approved incrementally to have less time reworking components
Integrate JSON parser for Workday API queries	1. The JSON parser could be improperly implemented and modify user data (0.1)	1. Research all available JSON parsers to ensure a quality one
Create methods to follow business logic rules to process data	1. We have a misunderstanding of how Iowa State IT will implement the Workday API which leads to incorrectly processed data(0.1)	1. We will have regular meetings with our domain expert from ISU IT to fix any misunderstandings quickly
Create REST API for middleware	1. REST API cannot handle less than a 1 second round trip time(0.3)	1. Research other process communication methods and change to a more performant one
Set up Mocked Workday API	1. We have a misunderstanding of how Iowa State IT will implement the Workday API (0.1)	1. We will have regular meetings with our domain expert from ISU IT to fix any misunderstandings quickly
Database	1. We have a misunderstanding of how Iowa State IT will implement the Workday API causing our database data to be incorrect (0.1)	1. We will have regular meetings with our domain expert from ISU IT to fix any misunderstandings quickly
Integrate Database and Mocked Workday API	1. The database and API are incompatible or refuse to communicate (0.2)	1. We will plan for the connection to the database during the process of creating the API

Table 1 – Risks and Risk Management/Mitigation

3.6 Personnel Effort Requirements

Task	Man Hours	Explanation
Create User Interface Screen Flow Diagram	50	Usability is one of the major requirements for this project, so it is important to get the design right. We will need to verify the design with the team often to ensure a user-friendly experience.
Create each individual UI component by picking high risk components first	240	4-year plan (flowchart and table): 80, entering courses: 50, login/sign-up: 30, aggregate data: 50, student information: 30. Creating the UI components will take the most amount of time while hooking them up to data providers should be fairly simple.
Connect Frontend to Middleware	30	We will do basic integration early in the project and finish the full integration later on. The full integration of the middleware to the frontend will require more time and effort.
Verify UI design with Iowa State UI/UX domain experts	20	This effort estimation assumes 2 meetings and accounts for any rework from feedback we receive.
Integrate JSON parser for Workday API queries	30	This expected number could go down as we explore the different libraries available for us to use.
Create methods to follow business logic rules to process data	100	This task involves taking in data from different sources such as Workday and processing it to create a student's four-year plan. This can be complicated due to the number of different majors and courses.
Create REST API for middleware	40	We will create basic endpoints early in the project and finish the full integration later on. The full creation of endpoints for the middleware will require more effort.
Set up Mocked Workday API	80	This entails setting up a class to communicate with a database holding mocked Workday data. This class and data will need to be set up in accordance with Workday's API.
Create Database	80	We will keep records on student information, their courses, majors, and so on. We will also need a database for any mocked Workday data we will want to use while building the program.
Integrate Database and Mocked Workday API	40	This entails using PHP to communicate with a MySQL database to store and query data. The necessary functionality of these methods will be tied to the mocked Workday API.

Table 2 – Personnel Effort Requirements

3.7 Other Resource Requirements

We will utilize a virtual machine for the purposes of hosting our web application. The virtual machine will be capable of not only hosting the frontend of the web application but can also host the middleware. Our virtual machine has already been set up by ETG and runs Windows 10. Otherwise, our group will not require any additional resources for this project.

4 Design

The design section of our plan covers all of the components of our program, the functionality of each layer, as well as some of the non-technical factors that we have considered. A number of diagrams and tables are used to describe the different aspects of our design.

4.1 Design Context

This section describes some of the broader concepts in our design, including some external factors to consider, basic user needs and the general components that will be built.

4.1.1 Broader Context

The project is being built for ISU and thus the university stands to be affected. The main communities affected by the project include broadly the scope of college students, but more specifically the subset of those that are interested in attending ISU. By extension from attending ISU, this also has an effect on the community of Ames from the increase in population and monetary spending associated with students living in Ames.

Area	**Description**	**Examples**
Public health, safety, and welfare	How does your project affect the general well-being of various stakeholder groups? These groups may be direct users or may be indirectly affected (e.g., solution is implemented in their communities)	A greater allure to transfer to and thus attend ISU results in more students in Ames. Therefore, incrementally increasing the need for jobs to teach and manage those students as well as providing workers for jobs students often fill.
Global, cultural, and social	How well does your project reflect the values, practices, and aims of the cultural groups it affects? Groups may include but are not limited to specific communities, nations, professions, workplaces, and ethnic cultures.	The project furthers the ubiquitous social expectation for automation of processes and convenience. Additionally, the current international societal standard for automated processes is that they be regularly updated for current visual and functional intuition, which is the goal of this project.

Environmental	What environmental impact might your project have? This can include indirect effects, such as deforestation or unsustainable practices related to materials manufacture or procurement.	An increased usage of an online resource provides the classic advantage of a reduction of reliance on paper documentation. Additionally, the project should reduce the need for in-person meetings with academic advisors, eliminating pollutants caused by needed transportation to attend such meetings. The associated downside is the running of the servers hosting the website and the associated power consumption needed for it. However, as this project is an improvement on an existing website, an increase in power consumption will only occur if a larger server(s) is needed.
Economic	What economic impact might your project have? This can include the financial viability of your product within your team or company, cost to consumers, or broader economic effects on communities, markets, nations, and other groups.	Being an upgrade of an existing project, the only potential economic cost is the need for more server space if the website sees a large influx of traffic. There are many potential benefits for both ISU and students, as well as rippled effects for the economy at large. When students see more benefit in attending ISU from the ability to transfer their courses, ISU will see the economic benefit from an increase in tuition. Students will also be given a better way of evaluating and utilizing the courses they have taken for usage of transfers. Higher utilization of courses taken for degrees can result in a reduction of redundant/useless spending for students. The additional access of funds allows students to spend more on useful spending, either in Ames (as a student) or in their

		local communities, resulting in economic stimulation.

Table 3 – Design Broader Context

4.1.2 User Needs

A **prospective student** needs a way to enter previously completed courses and receive an easy to digest four-year plan in order to successfully plan their course load at Iowa State University.

An **admissions employee** needs the ability to view the data and activity of prospective students in order to reach out to the student to promote Iowa State and help them successfully choose their courses.

4.1.3 Prior Work/Solutions

The other product like ours that exists is the ISU TRANSIT system [4]. It takes classes that a student has taken at other universities as well as a chosen major for the courses to transfer for. It then shows a list of what all the input classes equate to at Iowa State for the chosen major. It also outputs all classes that do not transfer whether it is because the grade the student received for that class was too low, or if there just is no ISU equivalent of that course.

While we have not followed this per se, we have still certainly taken advantage of the general screen flow of the application as well as the process the program undergoes. We are, however, not using the same user interface because the current one is not user friendly. We are also not carrying over the output in its current form as it goes against the requirements our client has set out for us.

Very few other schools have a system like Iowa State's TRANSIT software, Texas A&M has a system called "Howdy", [5] but that can only do one course at a time and generally lacks the complete overall functionality our client is looking for.

ISU IT has shown us the site that Franklin University [6] uses for their transfer students in hopes that the new TRANSIT system we create would be functionally or aesthetically similar, which is its main positive benefit.

4.1.4 Technical Complexity

Our design consists of three main parts that are divided into the frontend, middleware, and miscellaneous backend services. Here are some of the components/subsystems and challenging requirements our team will encounter ([1], [2], [3]):

1. The frontend includes the diagramming of articulating the page flow and components that will make up the user interface and user experience. In industry standards, it matches expectations of communicating how the product will look and flow before implementing it. The scientific principle of harmony, not discord, has been applied here since the development team and our client have been able to agree on how the visuals of the product will look.
2. The frontend and middleware will consist of PHP, Bootstrap, and HTML/CSS which are all web development standards currently being used across many companies and their sites. They all work efficiently and effectively together since they're oftentimes all used together.

This has been applied to the mathematical principle of using the appropriate tools strategically.

3. The middleware features a JSON parser and methods that will follow business logic to effectively process the data into a more friendly format once we pull what's coming from the Workday API. Using parsers is commonplace professionally since working with an easy-to-use set of data is needed for efficient development. The engineering requirement this aligns with is develop and understand since developing a workable data format and understanding how we're retrieving it in the first place is critical.
4. The backend side of the project is where we integrate with Workday, which is a leading financial and planning system-based software. Working with Workday will require us to mock their API calls due to the fact that the ISU implementation of Workday will not be done for two years. This requirement can be classified under the engineering principle of understanding and the scientific principle of cooperation, not individualism. The entire team is unfamiliar with the Workday API calls being used. Learning about the Workday API will require understanding to work with Workday's documentation and Iowa State Admissions IT to effectively use the API calls.
5. The backend side of the project is where we integrate with Okta, who is currently a leading authentication account software. Much like Workday, Okta requires REST API calls that store the information of user's accounts. This requirement can be classified under the engineering principle of understanding and the scientific principle of cooperation, not individualism. A majority of the team is unfamiliar with the API calls that Okta will provide us and will require understanding and working together with Okta's documentation to effectively use their API calls.

4.2 Design Exploration

This section explores the design decisions our team has come up with while comparing and contrasting what different solutions are best.

4.2.1 Design Decisions

- We will utilize a database in order to mock the Workday API for the backend of our project
- We will utilize a database in order to store user activity and data.
- We will use the PHP language for the frontend of our web application
- We will implement a login using Okta SSO for the purposes of allowing transfer students to save their data for later use
- Users will be provided with input forms to enter grades from their transfer courses

We utilized the lotus blossom to identify potential options for users to save data for later use. We grouped the ideas into three categories, traditional login methods, file-based methods to save data, and other methods. Within these three categories, our team came up with seven feasible options, including not implementing the feature at all, as it is only a soft goal and not a must-have requirement.

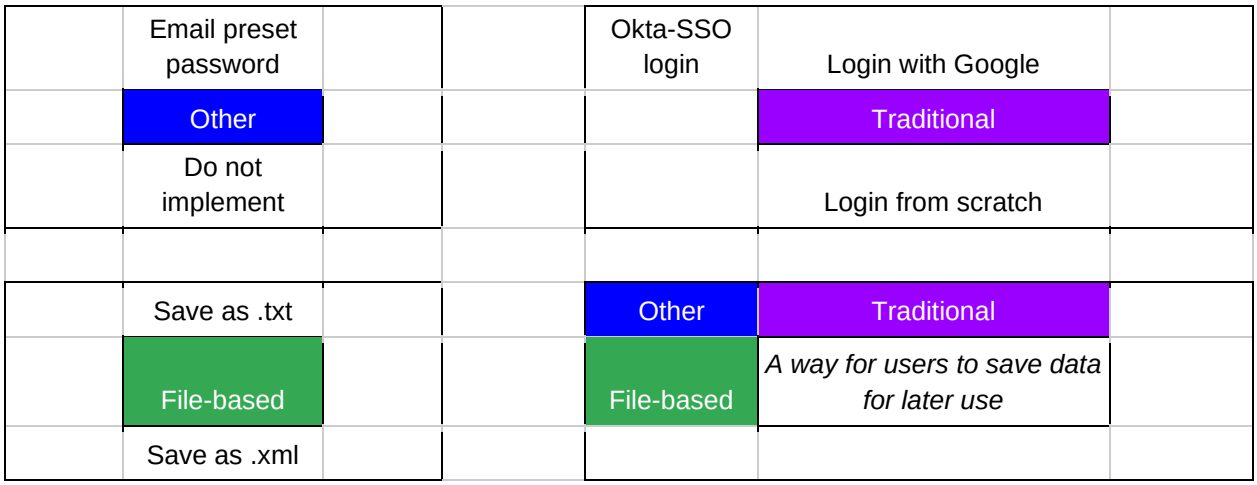

Figure 3 – Lotus Blossom

4.2.2 Decision-Making and Trade-Off

After creating the lotus blossom, our group had an open discussion about each of the potential options. Right away, we decided against the file-based methods of saving data. Because users could easily misplace the file or straight up forget about it, we came to the conclusion that there were better methods of saving data. Also, we decided against not implementing the feature, as we decided that we had the requisite time, along with the desire to meet our client's soft goal. With the remaining four possibilities, we utilized a weighted decision matrix to assist in our decision making:

Selection Criteria	Criterion Weight	Okta-SSO Login		Login with Google		Login from Scratch		Email preset password	
		Score	Total	Score	Total	Score	Total	Score	Total
Team Knowledge	0.25	2	0.5	1	0.25	3	0.75	1	0.25
Recommendation	0.4	5	2	1	0.4	2	0.8	1	0.4
Ease of Use	0.35	4	1.4	4	1.4	3	1.05	2	0.7
Total	1		**3.9**		**2.05**		**2.6**		**1.35**

Table 4 – Weighted Decision Matrix

The selection criteria included prior team knowledge, recommendation from the Admissions Department, and ease of use, as our client indicated that user-friendliness was the main goal for the project. Our team weighted the recommendation the highest, followed closely behind by ease of use. The Okta-SSO login scored high marks for both its recommendation and ease of use, however, it did not receive a perfect five for ease of use to its two-factor authentication feature. The Google login is about as easy to use as the Okta login, however, it was not recommended by Admissions. Our next option was to build a login from scratch, while the team had prior knowledge and experience with this option, it was not recommended by Admissions, although they did say it was possible. Finally, the preset emailed password was an all-around bad option for our project. From the result of the decision matrix, our project had a clear best choice, the Okta-SSO login.

4.3 Proposed Design

This section goes over multiple factors in our proposed design such as the components that will be built, the general functionality of the program and some of our concerns going into development. Much of this is described with the use of diagrams to show how different components connect and the flow of functionality for different users.

4.3.1 Design Visual and Description

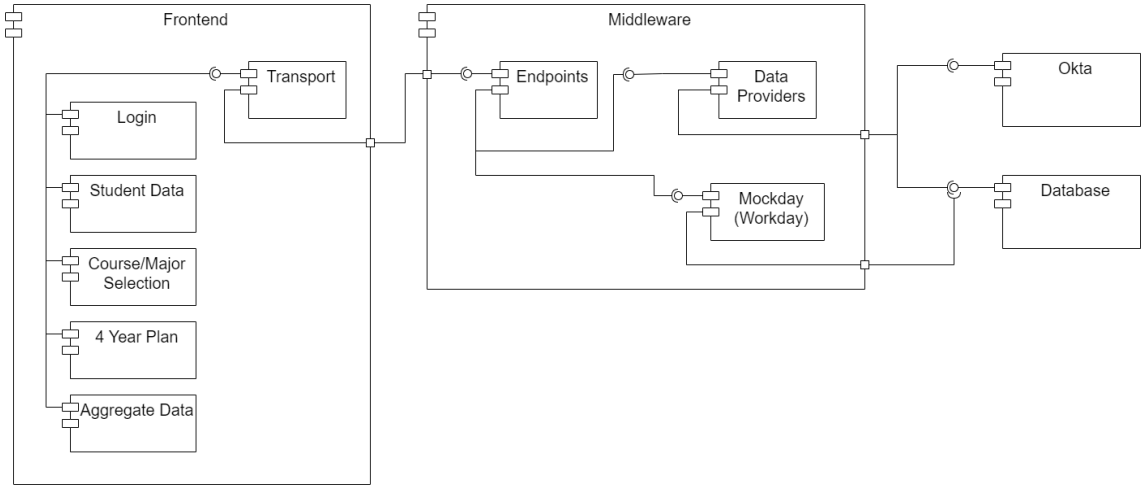

Figure 4 – Component Diagram

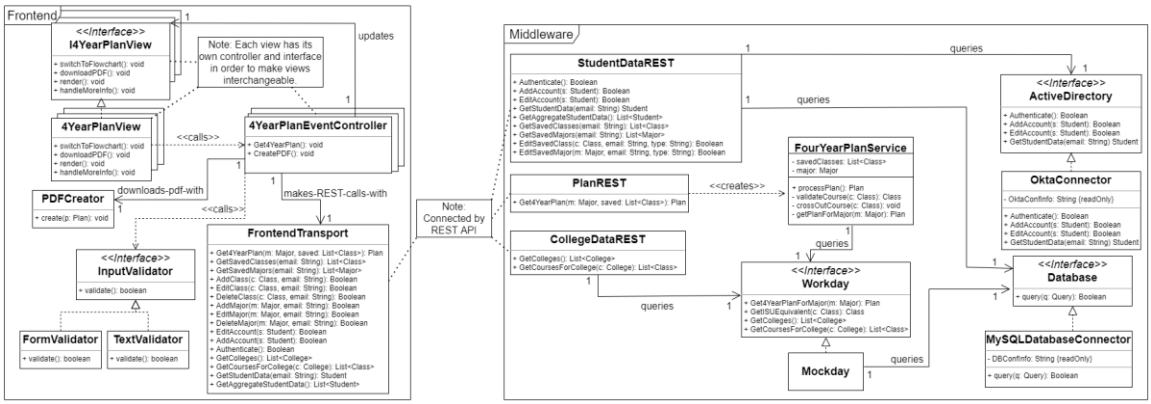

Figure 5 – Class Diagram

The two main components of our system are the frontend and middleware. We will additionally have a database to store user information. The frontend will deal with user interactions and display pertinent information. It will follow a simple MVC-style design. The middleware will host REST endpoints for the various backend services. Since the data sources are subject to change in the next few years, we have a Data Provider component to adapt the data to consistent format. As it stands, we will be mocking the Workday API for this project: Mockday. Mockday will interact with the database for its mocked data. Additionally, the Data Provider will use the database for storing and retrieving user information. While these accounts are provided by Okta, we will need to track our app specific data for each user.

4.3.2 Functionality

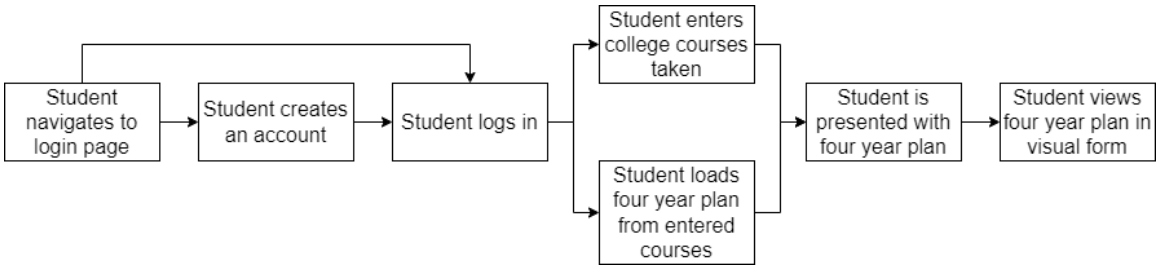

Figure 6 – Transfer Student User Timeline

As the figure above shows, the first step for a transfer student is to navigate to the login page. Then, if they haven't created an account, they can do that next. After they have an account, they log into it. They are then presented with the choice of either entering their college courses or loading a four-year-plan from previously entered courses. They then are shown their four-year-plan for their selected major with the courses that they have already taken crossed out. They also have the option to view a graphical representation of their four-year-plan.

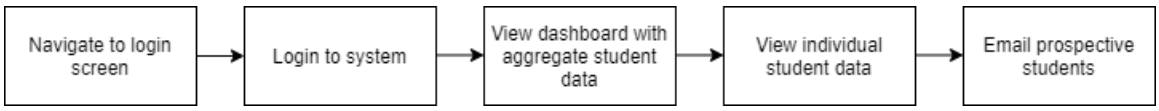

Figure 7 – Admin User Timeline

Admin users such as academic advisors and admissions employees also have to navigate to the login screen. They are able to login to the system using their Okta SSO login. Upon logging in, they are presented with a dashboard of aggregate student data such as what majors are popular for transfer students and what colleges other students are coming from. They can then navigate to a separate page to see individual student data such as who has been using the site recently. They can use that information to email prospective students to help convince them to come to Iowa State.

The current design does a nice job of satisfying the requirements for this project. All of the different functional requirements for how the system should behave have been followed. For example, administrator users can email prospective students and view student data. Another example is that transfer students can create an account, view a four-year-plan of their major and view a four-year-plan for their major at a later time. This design is easy to navigate because it is a very natural flow throughout the website. The design also shows that the website will be interactive so that requirement is satisfied as well. Lastly, this design shows a simple user experience so that they can spend less than 10 minutes on the site and not feel frustrated from long user sessions.

4.3.3 Areas of Concern and Development

Concern 1: One of our current concerns is how the four-year plan is going to be put together for each degree program. There are a lot of things to take into consideration such as elective courses. We also have to add in any transfer credits the student may have as that is the whole premise of our project.

Concern 2: Another one of our major concerns is that our mocked Workday should look similar enough to the real Workday so that when our program is integrated with the real Workday, it can

have a smooth transition. Currently, all the specific details on what information Workday will hold and what we will all have access too. If we build our program under the assumption that we can access certain data and it turns out Workday does not hold any of said data, it could require some major adjustments and work for whoever is integrating the two.

Concern 1 Solution: Our current plan for building a realistic four-year plan is to use a base case. Currently, Iowa State provides a base four-year plan for all of its majors. Our plan is to take these base plans and build off of them. We will cross-out any courses that the student may receive transfer credits for. We will still need to keep track of their general education credits. However, using the base four-year plan will make the process easier than building the four-year plan from scratch.

Concern 2 Solution: One thing we have done is meet with a domain expert to get a good idea of what Workday will look like. He was able to give us good information on some of the data we can get from Workday and in what form it will come in. As we continue to meet and discuss with him, we gain a better understanding of how we can build our program. Also, we are building a mock Workday as well as a temporary database to store any of the data we want to mimic for the development of the rest of the project. If there is certain information that Workday does not hold, we will have this database of our own that could hold this extra information. Since we will be starting with a basic amount of data, we want any future developer to have the ability to understand the code and easily add to the program or database. We plan on being very specific and clear in the documentation of our code, should any future developer need to add to our program.

4.4 Technology Considerations

- Workday API
    - Strengths - This is the software the University will be using as its backend API for its websites.
    - Weaknesses - Since the ISU implementation of this API hasn't been created yet, we will have no way to verify what the output of the API will be like, forcing us to rely on ISU IT to implement methods we need.
- Okta API
    - Strengths - Okta provides a quick and easy way for people with Okta accounts to sign in.
    - Weaknesses - We will have to create new Okta accounts for prospective students that will not already have one.
- PHP
    - Strengths - This is the language used by ISU for its websites, so using it here would allow our project to better integrate with the websites that already exist and would allow us to use the University's web development templates.
    - Weaknesses - Not all of our team members have experience with this language, which will add a learning curve to our project completion time.
- PHPUnit
    - Strengths - PHPUnit is the premiere testing software for PHP code, it is incredibly powerful and can get everything done.
    - Weaknesses - Only a few of our team members have experience with PHPUnit.
- Selenium
    - Strengths - Selenium is a powerful framework for testing web applications.

    - Weaknesses - Only a few of our team members have used it in the past.
- Synk
    - Strengths - Synk is a powerful tool for automated securities testing.
    - Weaknesses - No one in our group has used Synk in the past.
- Virtual Machine
    - Strengths - A VM can easily host our Frontend/middleware communication API.
    - Weaknesses - We will need to find a way to host the VM on a university server.

4.5 Design Analysis

The proposed design as in Figure 5 – Class Diagram has been verified and understood by the client and IT professionals from ISU. The visual diagrams have been properly thought out to encompass the larger focus of this project which is the frontend. The middleware has been created in order to properly interact with the 3rd party software that ISU IT and our client want us to use. Using this design, we can utilize ISU's active directory system to authenticate users and integrate with the eventual Workday backend. This design was also created with the intent of ensuring our system is extremely modular and could be integrated with any number of databases and active directory systems. This also ensured that integration with Workday would be easy for ISU IT down the road. The class diagram provides specific methods and classes that will be used for the project that depict the modular design of our system. The timelines provide insight to the flow of how users will use the system and which mimics what's already in place along with the additional requirements to this project. Lastly, the concerns of our design have appropriate solutions that wouldn't take away from the impact our project will have if encountered.

Iterating over the transfer student user timeline design will be important for our project since the user experience is the most important aspect of our project. Since there are many steps to this timeline, we need to make sure each one is well done and verified with respect to the requirements of the project. Like the student timeline, the admin timeline will need to be iterated over since providing ISU Department of Admissions employees with adequate data is crucial to potential prospective students. The class diagram may be modified once development begins to either add more specifics or remove unneeded details. However, we feel as if the current design is relatively satisfactory due to its modular nature and won't have to be altered much.

4.6 Design Plan

The design plan is broken into two main sections to be implemented by the project (the frontend and the middleware) and then a third mocked section acting as the backend of the full solution. For visual reference, see Figure 4 – Component Diagram and Figure 5 – Class Diagram. The goal of this section is to describe the entire design plan with respect to the considerations made such as the use cases, interfaces, and requirements.

The frontend section widely covers the use cases as the frontend is what the user will be interacting with. It is notable that we have decided to address the use cases largely by creating a separate page within the frontend that addresses each case (i.e., a page for sign-in, for entering courses and desired major, viewing aggregate data, etc.). From this breakdown of pages, each page must be able to interact with other pages and to interface with the middleware, so we decided to use the common practice of the Model View Controller architecture (MVC) to support this design plan. As the class diagram shows, each page (view) will have its own controller. All of these

controllers will use some shared classes to support the functionality of the frontend, like the input validator and the transport class. An input validator needed to be considered because of the multiple use cases involving user input. The transport class exists to be the interface between the controllers of the frontend and the middleware. It will make the necessary calls through a REST API.

The middleware provides the functionality in terms of data for the use cases addressed by the frontend, overall, it is needed to interface with external sources to retrieve data. It is also where we made considerations regarding the constraints and requirements such as the speed of retrieving data, which is why there are multiple REST endpoints in order to reduce the issue of creating a bottleneck of REST calls. Some computation may be needed depending on the use case and the associated requirements, such is shown by the four-year plan service class which will need to process the courses and plan based on information queried from Workday. The middleware also notably connects with three data sources, Workday, a database of our own, and Okta. The consideration to include a database of our own came from considering the requirements that may not be satisfied from the other sources, Okta, and Workday. Workday, as has been discussed before, has not been implemented, so our design plan includes how we plan to interface with it (and we will include a mocked version of Workday to demonstrate that interface). Lastly, as we will be needing to query Okta for signing in, we require interfaces for it as well, as has been shown in the class diagram.

5 Testing

To ensure our program is of high quality and running smoothly, we have developed a plan for testing our software from all different angles. Testing will be performed throughout the entire course of development using different tools and software, to ensure each component is fully functional.

5.1 Unit Testing

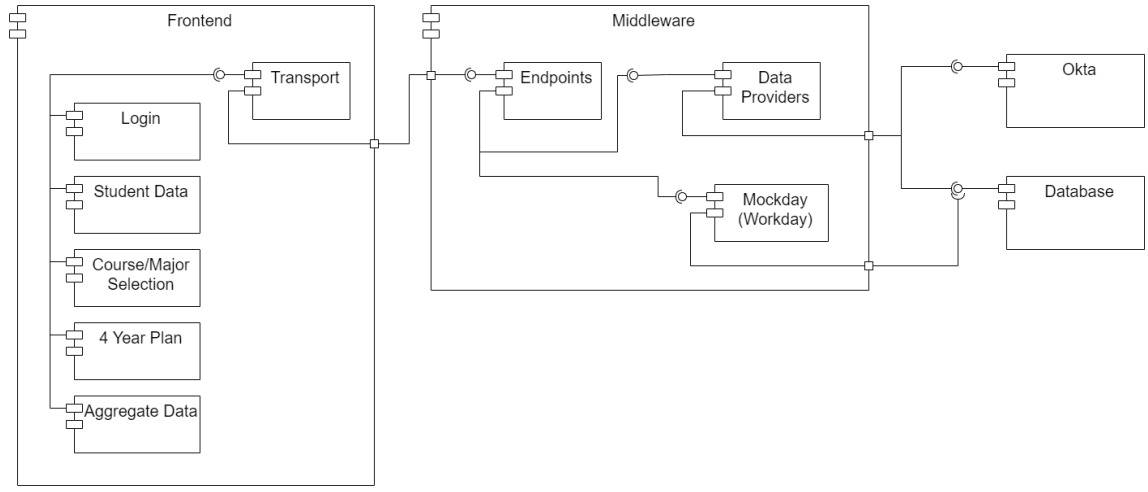

Figure 8 – Component Diagram

The above diagram shows the different components of our design. Each of these components, outside of Okta and the Database since they are external systems, will have unit tests.

The frontend will have unit tests for each of the different view components (Login, Student Data, Course/Major Selection, 4 Year Plan and Aggregate Data), a controller component and a transport component that sends information to the middleware. The view components will be primarily tested using Selenium to verify that all components of the UI are functioning correctly and in the correct location on the screen. We also will test the view components using PHPUnit for unit testing purposes. The controller and transport components will be tested using PHPUnit. Each individual function will be tested given various inputs to ensure that outputs match what we expect them to. The transport component will also be subjected to integration testing with the middleware which will be discussed further later.

The middleware will have unit tests for an endpoints component that is responsible for receiving the data from the frontend and sending the data to the correct spot to be processed or delegating processing to a service class, a data provider component that is responsible for receiving data from external systems, and a Mockday component that is used to simulate a Workday backend until the real Workday backend is connected in a few years. The endpoints and data provider components will both be subjected to integration testing since they communicate with components outside of the middleware. Each individual function for both components will be tested using PHPUnit to verify that data that is received from external sources can be formatted for easy processing later. Any service class the endpoints delegate processing to will also have each function tested using PHPUnit. This is to ensure that the data that it receives is processed correctly. The Mockday component will have some very simple unit tests using PHPUnit to ensure that the data coming out of it is similar to what Workday will eventually provide. It will also have some basic integration tests with the database.

5.2 Interface Testing

For interface testing we will need to test the interaction between the units for frontend and the middleware we create. As the backend will be provided and connected after the scope of our project, interface testing from middleware to a backend will be minimal, limited to connecting to a testing database.

To mock and test calls from the frontend to the middleware we will use tools like Postman to make API calls to the running middleware. This will pretend to be the frontend so we can isolate the call and response. While we cannot do testing with the real intended backend for the project, as it does not exist and is outside the scope of the project, we will ensure the middleware's ability to make such required interfacing. Such interfacing could include making API calls from the middleware and connecting to a database. Testing the API calls from the middleware can once again be manually tested using Postman by setting up a mock server and responses to calls and testing a database connection can be done using a local database like MySQL Workbench.

5.3 Integration Testing

Our design will have two major integration paths, connecting the frontend to the middleware, and the middleware to the backend. These are critical because in the grand scheme of things, our project will not function unless both of those connections are working. We need to get all of our project components integrated or else there will just be several, disconnected parts that

cannot complete our use cases. The integration of these will be tested with tools such as PHPUnit for the frontend/middleware and middleware/backend integration, while Selenium will be used to test the integration of the whole system. We will use a mix of automated and manual testing to ensure that all of the components of our design are connected together and operating properly.

5.4 System Testing

For system testing, our group plans to rely primarily on automated functional tests. We plan to write these tests using the Selenium framework for PHP, along with the Chrome Developer Tools to assist with web scraping our application's elements. Although we plan to have all webpages touched by at least one functional test, we will focus on three critical features for the majority of our functional tests. These critical features are the ability for the user (a prospective transfer student) to input transfer courses and grades, the ability for the user to view a four-year plan for the Iowa State courses in their intended major (with the courses they have transferred crossed off), and the ability to display different views depending on the account type the user is logged in as. By creating many automated tests for these features, we will be able to ensure our main requirements are satisfied, and that they are not regressed by future code changes. These automated tests are system level tests because they interact with the system as a whole; in our case a web application, from the perspective of a user. In addition to automated tests, manual system testing will be utilized by both the developer and potentially reviewers as well.

5.5 Regression Testing

To ensure that new features or bug fixes do not break the old functionality, our group will primarily rely on GitLab's CI/CD feature. One of the initial steps in our project will be to configure a CI/CD pipeline for our repository. This pipeline will be configured to run all unit tests, integration tests, and system (functional) tests. Unless otherwise specified by the developer, our pipeline will run both when a merge request is created (the merge request will not be able to be merged unless all tests pass), a merge request is updated with a code change, and after a merge request is merged into the master branch. By running all of our tests both before and after merge, our ability to identify and fix regressions will depend on our test coverage. Since we plan to have large unit and integration test coverage along with system (functional) test coverage for our main requirements as mentioned in the prior section, this should be sufficient to identify most regressions. However, if we do identify a regression that was missed by the CI/CD pipeline, we can always write another test to prevent that regression from happening again. Finally, manual testing will be utilized as well by both the author of the merge request and potentially reviewers if there is a high regression probability identified.

The critical features that we need to ensure to not break are the ability for the user (a prospective transfer student) to input transfer courses and grades, the ability for the user to view a four-year plan for the Iowa State courses in their intended major (with the courses they have transferred crossed off), and the ability to display different views depending on the account type the user is logged in as. These features are derived from our project's functional requirements. Most of our project's functional tests will focus on these three critical features.

5.6 Acceptance Testing

Design requirements are met by demonstrating the visual output of the web application and internal processing, thus, having alpha testing can be used to make sure the project is able to

withstand user interaction. Alpha testing is done through both black and white box testing where it looks at both the end user perspective and implementation code. This can be applied to functional requirements where the end user is able to enter their courses and be shown a four year plan all while being signed into their newly created student account. In terms of non-functional requirements, the performance and volume request will be determined by their load times on the page. Other non-functional requirements will be tested according to their criteria.

Beta testing is one of the most effective methods out there which is currently how Iowa State University Department of Admissions tests their finished applications. This phase is usually done near the end of the finished product before the final release. Beta testing is done through black-box testing which means our client wouldn't have access or knowledge about what's going on inside this application. It would be highly beneficial since our client is able to use the application as if she were a student and needed course transfer information. It also will provide insight on our client's user experience and how she would use the product.

5.7 Security Testing

We will perform both vulnerability and security scanning to search for any weaknesses in our software. These scans for weaknesses can be done with Synk, an automated program which will give us better results than manually testing every aspect. This program works with PHP along with any other language we will likely use for development. After determining some of our vulnerabilities, we will perform a risk analysis. This will help us rank the importance of certain security issues and develop a plan to mitigate those risks.

To test the security of our software, we will be performing penetration testing which simulates an attacker. One way of doing this is called abuse/misuse testing, which is testing different use cases and seeing how the software behaves. Another important way we plan on testing security is by static testing. This will involve running these security scans and tests on new code as it is being committed. This will allow us to catch vulnerabilities early on through the development process.

5.8 Results

Our extensive testing strategy will result in a satisfactory application that our client and users will find beneficial. Unit testing will ensure that even the lowest level unit of our system functions properly and acts as a preventative measure against regressions. This assurance will give us confidence that our system will be as bug free as possible to our end user, allowing them to have the most hassle-free experience in figuring out their transfer plan. Interface testing ensures reliable availability of the system by verifying the connections between the frontend, middleware, and backend services. System testing, both automated and manual, allows us to verify the whole stack from the frontend to the backend. We are able to measure the performance of the system end-to-end to ensure we meet our performance constraints. We will use GitLab CI/CD to continuously run our automated tests to prevent regressions. The different acceptance testing phases allow us to verify the design of the system and UI at different stages of development. This way we have confidence that our system provides value to both potential transfer students and admissions employees. Additionally, security testing will ensure transfer student data is only accessible via admin accounts.

The diagram below outlines the flow of implementing a feature and the responsibilities of each team member in the process.

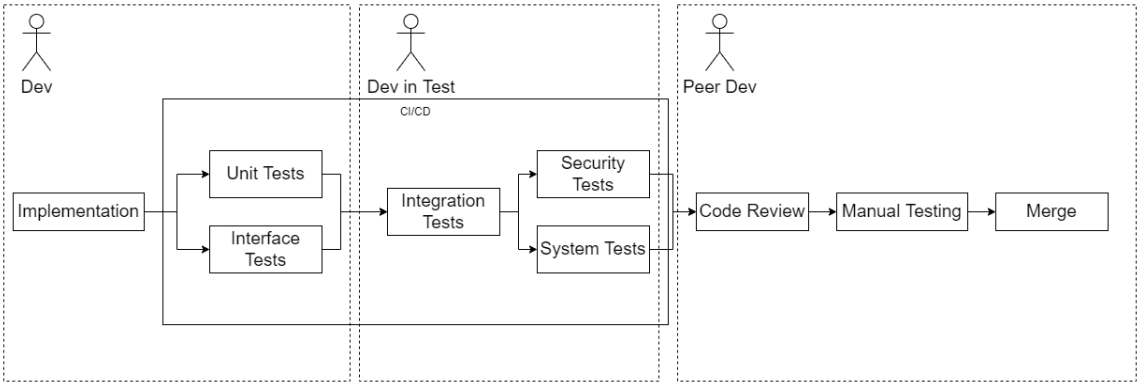

Figure 9 – Testing Flow Diagram

Requirement	**How it is addressed**
Able to interact with a workday backend or mocked version	Interface Testing
Output a four-year plan with courses crossed off	Unit and System/Functional Testing
Accept as input transfer courses from other universities	Interface Testing between middleware and backend, Functional frontend testing
Accept students, advisors (admin), and admissions staff (admin) account types	Security Testing, Unit and System/Functional Testing
Display different views depending on the account type the user is logged in as	Security, Unit, and System/Functional Testing
All project tables containing credits shall display the total number of credits in that table	Unit, and System/Functional Testing

Use the inputted transfer courses to determine transferability at Iowa State.	Unit and System/Functional Testing
Able to download a .pdf file of the four-year plan of their intended major in table and flowchart format.	Unit and System/Functional Testing
Allow for account creation and edits to that account	Interface Testing between middleware and backend, Unit and System/Functional Testing
Allow for linking to outside pages	Unit and System/Functional Testing
Allow administrator users to view data regarding student users	Security, Unit, and System/Functional Testing
Allow administrator users to reach out via email to prospective student users.	Unit, and System/Functional Testing
Respond within 1 second to a four-year plan query	System Testing
User is able to complete a session in under ten minutes	Acceptance Testing in beta phase
User interface shall be easy to navigate	Acceptance Testing in alpha phase

Table 5 – How Testing Addresses Requirements

6 Implementation

One major risk to our project was that the page design and screen flow would lead to a poor user experience and would not be valuable to users. We recognized the severity and probability of such a scenario and switched to a prototyping phase. We began to iterate on possible UI designs, taking inspiration from similar products while improving them with our own ideas.

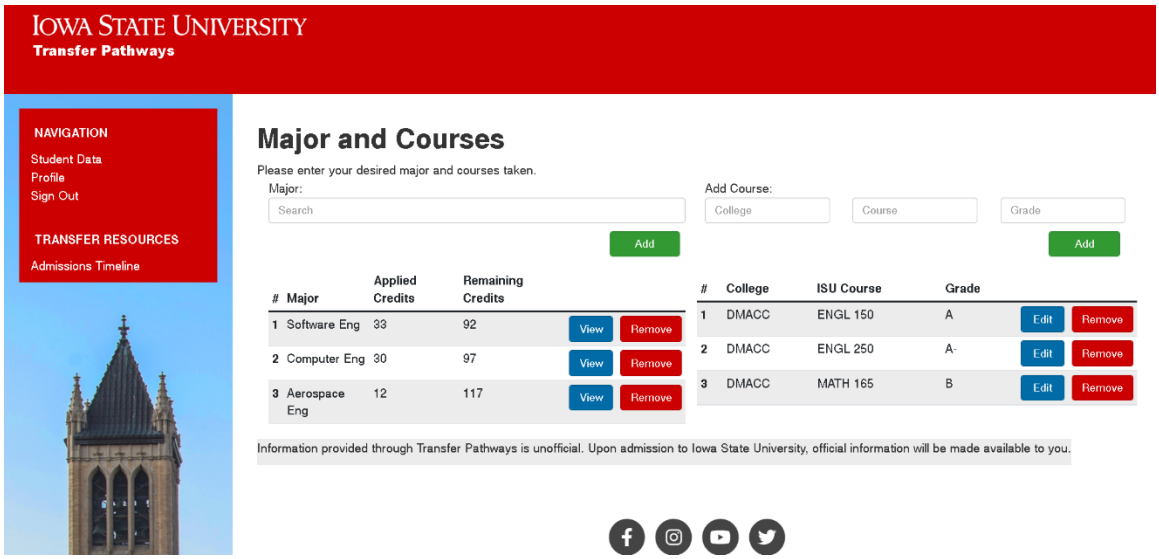

Figure 10 – Major and Courses Prototype Page

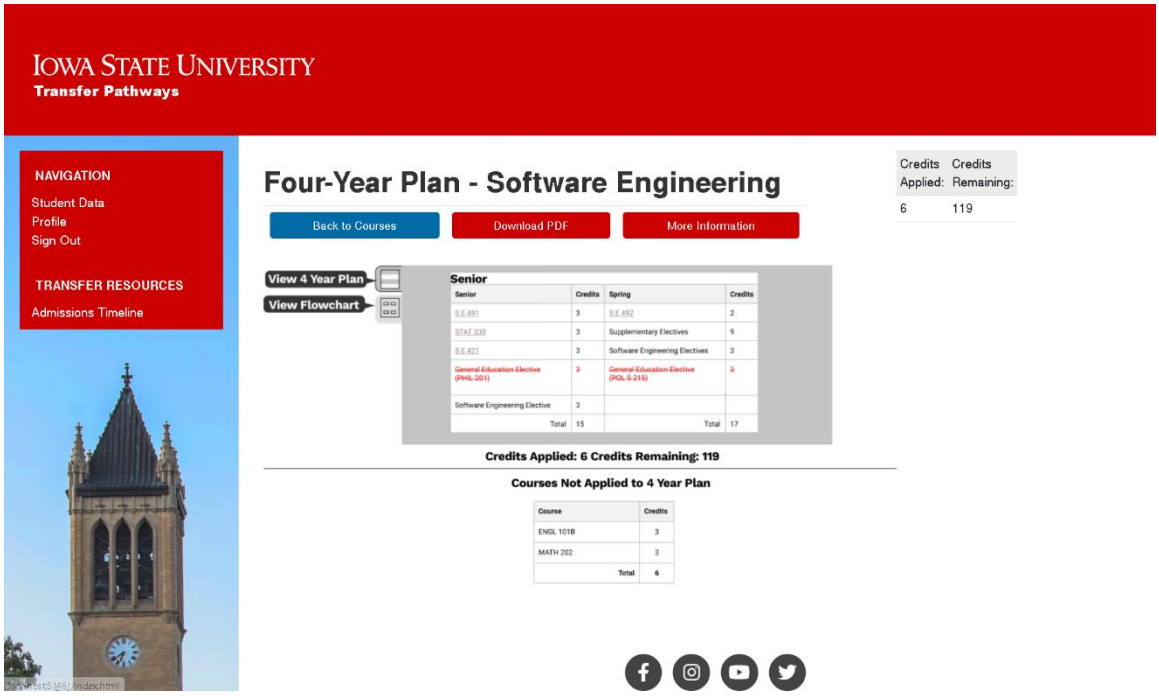

Figure 11 – Four-Year Plan Table Prototype Page

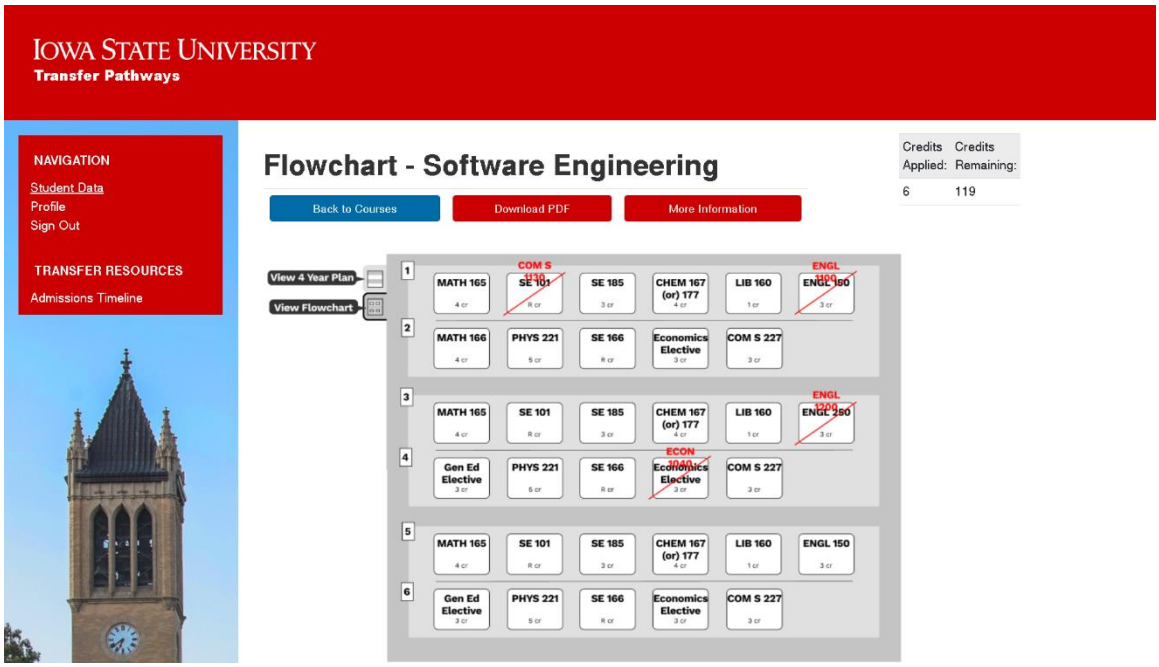

Figure 12 - Four-Year Plan Flowchart Prototype Page

Pages shown above available at [Transfer Pathways Demo • Iowa State University (iastate.edu)]

We have made both a mockup and a prototype of our UI designs shown above. We have shown the mockup to our client and received feedback about the design. The prototype is a plain HTML website with only navigation functionality. The HTML prototype serves as the foundation for the rest of the frontend implementation. We have many UI elements in this prototype that are complete and many that need finalizing. While working on each frontend task outlined in the project plan, we will first complete the relevant UI elements to that task (if not already finished). We will then work on the rest of the task's implementation.

Many of the frontend tasks will require data from the middleware, but it may not be desirable to wait for the middleware tasks to be completed. The same can be said about the middleware waiting on database tasks. To address this, we can simply use dummy data (as opposed to the persisted mock data). With our modular design, swapping out the dummy data with the real data source on either the frontend or middleware will be quick and easy.

7 Professionalism

Professionalism is of utmost importance to our project, and we have looked over the ACM SE Ethics and Professional Practices and thought about how our project relates to this code.

7.1 Areas of Responsibility

Area of Responsibility	Definition	IEEE/ACM SE Ethics and Professional Practices
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence	Product. Software engineers shall ensure that their products and related modifications meet the highest professional standards possible
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs	Client and employer. Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders	Judgment. Software engineers shall maintain integrity and independence in their professional judgment
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders	Client and employer. Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest
Property Ownership	Respect property, ideas, and information of clients and others	Client and employer. Software engineers shall act in a manner that is in the best interests of their client and employer, consistent with the public interest
Sustainability	Protect environment and natural resources locally and globally	Public. Software engineers shall act consistently with the public interest
Social Responsibility	Produce products and services that benefit society and communities	Public. Software engineers shall act consistently with the public interest

Table 6 – Areas of Responsibility

Work Competence

One of the SE code of ethics is product. Simply put, this code requires that engineers focus on building high quality products. Engineers should strive to understand all the specifications of the software they are developing. They should take into account factors from every angle, including environmental, cultural, legal, economic and ethical aspects. They should perform adequate amounts of testing and debugging to ensure they can deliver a high quality, fully functioning product to the client, customer, or employer. This code is very similar to the NSPE version at its base. The main idea of both is to strive for high quality. The SE version is specifically focused on software development, and makes points regarding testing, debugging and other aspects specific to software engineering.

Financial Responsibility

The client and employer codes in the SE code of ethics talk about having your client's or employer's best interests in mind, without going against the public interest. This means staying faithful to who you are working for. Keep confidential information known to only those involved and report mishandling of information or software. Always stay honest when communicating progress, expectations, limitations, and abilities. All focus should be on the client's needs and benefits, within the scope of public interest. This code differs from the NSPE version, as it does not specifically use the word financial or any like term. It rather focuses on integrity to the client and their interest.

Communication Honesty

The parallel principle to Communication Honesty in the SE Code of Ethics is Judgement. Engineers have a responsibility to communicate honestly and openly with coworkers, clients, and other stakeholders. They must engage in all evaluation objectively regardless of the consequences of an unfavorable conclusion. This principle differs from the NSPE version by focusing more on general objectivity within the organization as opposed to communicating with the public.

Health, Safety, Well-Being

The principle most similar to Health, Safety, and Well-Being in the SE Code of Ethics is Client and Employer. Engineers should be beholden to issues of social concern. They are responsible for reporting and acting on known ethical issues. Depending on the product, this could involve sensitive personal information, vital medical data, or plenty of other at-risk information. This could manifest as emphasizing data security, or, on a more sinister note, ensuring that this data is not being sold by the company. This principle differs significantly from the NSPE version because in software, health, safety and well-being do not have as much of a direct effect on decision making, excluding the examples given above.

Property Ownership

The client and employer codes go along best with the idea of property ownership. This area of the SE code of ethics stresses the importance of keeping your client/employer's best interests in mind. This means the information of the client and software being developed is kept confidential. Any communication should be kept honest and non-deceptive, so the client/employer is always informed accurately. Any software and physical technologies being used in development should be used and obtained with proper consent and knowledge from the client. These codes are very similar to the NSPE ideas of acting as a faithful agent to the client.

Sustainability

The ACM code of ethics discusses the importance of designing and developing systems for a safe natural environment and the potential damage it can have to either the local or global environment. This code of ethics was very similar to the NSPE version. However, the NSPE code did include one difference, it provided a reason for sustainability which was to protect the

environment for future generations. This was not directly mentioned in the ACM code of ethics, therefore, it seemed as though it was more so focused on the present impact it has on the environment.

Social Responsibility

The public code in the SE code of ethics is the same as the NSPE code for social responsibility. The SE ethics talk about always keeping the general public in mind when building software. While an engineer should always focus on their clients/employer's best interest, they must do so in the scope of society. They must not build anything that would diminish the quality of life, privacy, or the environment. All publicly released information, testing, and documentation should be honest and understandable. There should be no shortcuts taken when testing or approving different technologies that may have an impact on the society. Engineers should take full responsibility in developing high quality software that will benefit their client without the expense of anyone or anything else.

7.2 Project Specific Professional Responsibility Areas

Area of Responsibility	**Application**	**Performance**
Work Competence	Yes, it'll be necessary for our team members to be competent with the work required to implement the project. The skills and abilities that are applied to the project would be following the same technology standards ISU IT has in place such as PHP, Bootstrap, and HTML. We also need to have a clear idea of what is expected in terms of the user experience and flow according to our client.	High We've been able to gain insight not only from ISU IT but our client as well to understand what is expected of us from the project and its requirements. Although it may have taken more meetings and emails than previously thought, we currently have a clear path and direction to follow. Our team has been familiarizing ourselves with the technology skills that will be required to develop this application such as PHP since a majority of our group members don't have any experience with it.

Financial Responsibility	Yes, our group needs to deliver our Transfer Pathways Tool in a way that provides a high degree of value to the Admissions Department at a low cost.	Medium While we have delivered a prototype of the Transfer Pathways Tool to our stakeholders that will provide the Admissions Department with value, until we actually ramp up development it is difficult to say we have performed high in this area. For cost, we can say that we are performing high, as all that we will require cost-wise is server space.
Communication Honesty	Yes, communication honesty applies to our project because we have multiple stakeholders who expect a good product. Honestly communicating to our stakeholders is definitely an ethical practice in our case, so that our stakeholders have an accurate idea of project progress.	High From very early on in the project, we have maintained a high degree of communication with all stakeholders, including our faculty advisor and our two main contacts from the Admissions Department. We have met frequently with all stakeholders via video chat in addition to email when feasible. Finally, we have been honest about the progress of our work to all stakeholders.
Health, Safety, Well-Being	Yes, this applies to our project because while no one can be physically harmed from the website, psychological harm could be done. If users receive incorrect or misleading four-year plans, they can be hurt by incorrect planning and preparations.	High We have included multiple disclaimers in our UI in order to make sure transfer students understand this information is not final.

Property Ownership	Yes, one item that could potentially be of concern to a user is the ownership of the data they give to our system. Additionally, we may run into ownership issues with the course information from other schools and institutions. We must also be wary of using ISU-owned materials like the Workday API, the ISU web templates, and the Okta API - being sure to credit them in our final product.	Low Any user that submits information to our tool to find their modified four-year plan will have their information accessible by admin users from the department of admissions. Another issue we might encounter would be the possibility of not being allowed to follow ISU's specific logos/images for websites due to copyright regulations.
Sustainability	No, since our project is updating a website using a server that already exists it has no environmental impact.	N/A
Social Responsibility	Yes, this product will benefit society in many ways. It will allow transfer students to more easily be able to plan their course work at Iowa State. It also will allow administrators to learn what students are generally interested in attending Iowa State. All of this will be able to be achieved using an easy-to-understand user interface that any person could be comfortable with.	High We have performed highly in this category. We have done a lot of research and iterated using feedback on a prototype of our website in order to make it as user friendly as possible. Making it easy to use will allow it to appeal to the most people. This allows the greatest number of individuals in society to benefit.

Table 7 – Project Specific Responsibility Areas

7.3 Most Applicable Professional Responsibility Area

Communication honesty is crucial to our project for two main reasons. First, the project will be a public, outward facing representation of ISU to prospective students. For this reason, we must have consistent, open, and honest communication with ISU Admissions to ensure we accurately portray the image ISU wishes to project. Additionally, we want the project to portray accurate and honest communication about transferring courses to prospective students as well. We have had multiple meetings with our contact in ISU Admissions including meetings to discuss

visual appeal, flow of information, and representation of information to largely great approval and constructive suggestions and feedback.

For the second reason, the project is being created with the intention of merging with a backend being created by ISU IT at a later date. While many details about such backend implementation have not yet been determined, this still means we need to be open about how we make decisions on our frontend so when ISU IT does their implementation, they know how to make the connection as well as tell us about any decisions they know will not function properly. Like meetings with ISU Admissions, we have had multiple meetings with ISU IT as well to learn all that we can, create an understanding, and set up a consistent channel of communication for while we work in the future.

8 Closing Material

The planning and prototyping done this semester and laid out in this document will serve as a comprehensive guide for implementation of the project next semester.

8.1 Discussion

Our work this semester has resulted in what we feel is an effective and detailed plan to build the Transfer Pathways Tool. We have put together diagrams to show the different components, screen flow, class structures and testing procedures. Each component will be built following the designs we have put in this plan, and each will be properly tested before being implemented with the whole system. Our designs will allow us to jump into development and work efficiently to build each component.

We have also considered many non-technical factors that are a part of the plan we have developed. Our plan has a set of ground rules to ensure we are communicating so everyone is on the same page, everyone is on track with their work, and we are all putting in the effort to build a quality program. We have included some of the IEEE standards we will follow, as well as how we will remain professional and ethical during development.

We have also developed a simple prototype of the program to provide our client with a visual representation of what the program will look like and how it will function. The prototype along with our design plan, hits all of the requirements that have been put in place by our client. When we finish development, our client will be presented with a fully functional web-based program that will allow potential transfer students to see how their courses transfer to Iowa State, and what their semesters would look like should they choose to attend. Our finished program will be properly and clearly documented to be ready for implementation with Workday when ISU IT is finished with their side of development.

8.2 Conclusion

Our main goals for this semester were to complete a high-quality project plan that can be easily used as a roadmap for project development, to develop relevant skills for the project, to improve our soft skills, and to ensure our project will meet the client's desires. We believe that the work we have done and the design we have created has achieved these goals.

First, our project plan is extremely detailed, high quality and will lead to successful software development next semester. We have listed all required tasks for our project and a timeline of when we will complete them. This will help ensure that our development process is as structured and efficient as possible. We also have made sure that all requirements have been met and all risks have been mitigated by having frequent communication with our client and ISU IT. They both have approved our plan and are excited to see us work through it next semester.

Next, we have developed many relevant skills for the project this semester. All team members have been learning the different technologies and frameworks we plan to use for development next semester in order to increase efficiency during development. We believe that we have a good understanding of these technologies and are excited to use them next semester.

Third, we have done numerous different tasks to improve our soft skills, specifically communication. We have had many meetings with ISU IT, our client, and our faculty advisor in order to ensure that our project was as high quality as it could be. During the meetings with our client, we collected many important requirements for the usability of the project. During the meetings with ISU IT, we learned about the technical requirements of the project, specifically connecting with the Workday backend. All of these meetings culminated in our team creating a successful prototype in which all stakeholders agreed that our work was well done and ready for implementation.

Lastly, we have ensured that our project will meet our client's desires through our verification that we met all of her requirements by creating a prototype for her. She verified that this was the product that she was looking for and thought the user experience would be good.

To conclude, all of our main goals for this semester have been met. We have had an extremely successful semester and hope to carry this success into implementing the project next semester. We will follow our thorough project plan in order to ensure that next semester is a success as well.

8.3 References

[1]K. Gnatek, "Engineering principles: putting our values into practice," Medium, Jul. 30, 2020. https://medium.com/taxfix/engineering-principles-putting-our-values-into-practice-4bbc140d4fa2 (accessed Nov. 22, 2021).

[2]"Taylor Principles of Scientific Management: Meaning, Definition," BYJU'S. https://byjus.com/commerce/taylor-principles-of-scientific-management/ (accessed Nov. 21, 2021).

[3]"Standards for Mathematical Practice | Common Core State Standards Initiative," Common Core State Standards Initiative, 2019. http://www.corestandards.org/Math/Practice/ (accessed Nov. 21, 2021).

[4]"Iowa State University TRANSIT," TRANSIT. https://transit.iastate.edu/ (accessed Nov. 22, 2021).

[5]"Howdy," Howdy. https://howdy.tamu.edu/uPortal/p/tce-ui.ctf1/max/render.uP (accessed Nov. 22, 2021).

[6]"MyTransfer Credit | College Credit Transfer Check Tool | Franklin.edu," Franklin University. https://www.franklin.edu/transferring-credit/estimate-your-transfer-credit/transfer-credit-tool (accessed Nov. 22, 2021).

8.4 Appendices

8.4.1 Team Contract

Team Members:

1) Curt Lengemann 2) Cole Weber

3) Scott Thurston 4) Ben Greif

5) Cameron Brecount 6) Riess Radtke

7) Luke Turczynski

Team Procedures

Day, time, and location (face-to-face or virtual) for regular team meetings:

Day	Time	Location	Attendees
Friday (as arranged)	11 am (as arranged)	Virtual (Webex)	Team + Client
Friday	2 pm	Virtual (Webex)	Team + TA
Friday	2:30 pm (as arranged)	Virtual (Discord)	Team
Sunday	1 pm (as arranged)	Virtual (Discord)	Team
Tuesday	1 pm (as arranged)	Virtual (Webex)	Team + Advisor

Table 8 - Meetings

2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):

- Discord chat

3. Decision-making policy (e.g., consensus, majority vote):

- Majority vote

4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):

- Meeting minutes will be saved by Reiss and shared with the group via Discord
- Webex meetings will be recorded by Scott to have as a future reference

Participation Expectations

1. Expected individual attendance, punctuality, and participation at all team meetings:

- Team members will attend all meetings with the TA
- Team members will attend all meetings unless advanced notification is given
- Team members will be punctual unless advanced notification is given

2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

- Team members will take an active role in each assignment. Completing a fair portion of the work.
- Team members must meet course deadlines. Team deadlines may be flexible if advanced notice is given.
- Team members will complete all work at least 24 hours prior to due dates.

3. Expected level of communication with other team members:

- Team members will react or otherwise respond to Discord messages or emails within 24 hours.
- Team members will take an active role in meetings and not be distracted by other work.
- Team members will pay attention to discussion of other areas of the project regardless of if the topic is their direct responsibility.

4. Expected level of commitment to team decisions and tasks:

- Each team member should be committed to the project and group giving their best effort at all times
- Each team member should try to spend at least 9 hours per week on this project

Leadership

1. Leadership roles for each team member (e.g., team organization, client interaction,

individual component design, testing, etc.):

- Curt Lengemann - Database/Middleware Component
- Ben Greif - Test Engineer / Testing
- Scott Thurston - Frontend Component / TRANSIT Liaison
- Luke Turczynski - Report Manager / API Design / API Testing / Team Organization
- Cole Weber - Meeting Facilitator / Frontend Component / Client Interaction
- Cameron Brecount - UI/Frontend Component
- Reiss Radtke - Meeting Scribe/UI

2. Strategies for supporting and guiding the work of all team members:

- Actively communicating and asking questions to all team members about technical and design issues.
- Using and actively participating in the “help me please” channel we have set up on our discord for support.
- Require 1 developer code review and passing CI prior to master merge.

3. Strategies for recognizing the contributions of all team members:

- Constructive criticism will be given to ensure team members don’t feel as though their contributions are lacking.
- A team member of the week will get a special colored name on discord for fun.

Collaboration and Inclusion

1. Describe the skills, expertise, and unique perspectives each team member brings to the

team.

- Curt Lengemann - Skilled at web development, using SQL and NoSQL databases, using Git, software design and project management. Interned at Workiva and Transamerica as a Software Engineer which gives me a unique perspective on software development.
- Ben Greif - Skills include web development, functional testing, and setting up CI/CD. Internships include two Quality Assurance internships at Workiva. I bring a quality and testing-focused perspective.
- Scott Thurston - Interned at Principal Financial and Marshalltown Co., skills attained include Web Application Development, Database Management and Development for SQL and NoSQL, and UI Design.
- Luke Turczynski - Interned at Collins Aerospace. Experience in APIs, Backend, web development, and UI design.
- Cole Weber - Work experience includes Iowa State Department of Admissions and Buildertrend. Experience in frontend and backend development in web and mobile development. My unique perspective is investigating if new technologies should be incorporated into projects.
- Cameron Brecount - Work experience includes 3D Astronaut Simulation research project, UI and frontend development at Workiva.
- Riess Radtke - Work experience at ISU Dining in a management position. Skills include UI design, SQL database management, mobile application development

2. Strategies for encouraging and support contributions and ideas from all team members:

- Team members will respect and give due consideration to ideas presented by other team members, the client, the TA, and the advisor.

3. Procedures for identifying and resolving collaboration or inclusion issues (e.g., how will

a team member inform the team that the team environment is obstructing their

opportunity or ability to contribute?)

- Team members will be respectful and try to understand the perspectives of others when bringing up or resolving issues.
- If a team member is unresponsive to the requests of others regarding issues, additional actions may be taken. (i.e. Consulting TA or professor)

Goal-Setting, Planning, and Execution

1. Team goals for this semester:

- Complete a project plan that is of a high quality and can be easily used as a roadmap for development of the project.
- Develop relevant skills for the project.

- Improve soft skills such as communication with clients.
- Ensure our project will not only function but will also wholly satisfy the needs and desires of the client.

2. Strategies for planning and assigning individual and teamwork:

Design:

- Group meeting to discuss what documentation needs to be done for the week and distributing equally to team members
- Assign team members sections of documentation through Google Documents comments

Software:

- Gitlab tickets will be created for tasks with a point system for their complexity.
    - We will try to balance effort points between all team members.
- Group members will be able to accept tickets of any area of interest to them.
- Unwanted tickets will default to the chief of the respective area based on effort points.

3. Strategies for keeping on task:

- Tickets will have a completion date that should be met.
- Tickets can be rolled into Epics for larger groups of tasks with more distant deadlines.
- Group members can post in discord for help so that anyone in need of assistance can reach out to other group members for help.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?

- The first step will be to reach out to the person infringing on the agreement to inform them of what has been going on and bring their attention to the concern.

2. What will your team do if the infractions continue?

- Complaints will be directed to the TA/Professor so that they can take the appropriate next steps.

****

a) I participated in formulating the standards, roles, and procedures as stated in this contract.

b) I understand that I am obligated to abide by these terms and conditions.

c) I understand that if I do not abide by these terms and conditions, I will suffer the

consequences as stated in this contract.

1) _____Curt Lengemann_____ DATE ____9/12/21_____

2) _____Cole Weber_____ DATE ____9/12/21_____

3) _____Luke Turczynski_____ DATE ____9/12/21_____

4) _____Scott Thurston_____ DATE ____9/12/21_____

5) _____Cameron Brecount_____ DATE ____9/12/21_____

6) _____Ben Greif_____ DATE ____9/12/21_____

7) _____Riess Radtke_____ DATE ____9/12/21_____