Team 42

Client: Vicky Thorland-Oster

Adviser: Thomas Daniels

Team Members/Roles

Ryan Tullis – Team Lead

Dillon Gesy – Team Organizer

Nicolas Figueroa Calderas – Frontend Developer

Charles “Joe” Hosier – Frontend Developer

Abdullahi Abdullahi – Backend Developer

sdmay24-42@iastate.edu

Team Website

Revised: 12/4/2023 v9.0.0

Executive Summary

Development Standards & Practices Used

The software practices will follow the standard software project practices. The program should work on most systems and be relatively small in size.

Following the waterfall project management style with integrated things from the agile process will help us with the project's lifecycle.

The project will be tested thoroughly to make sure the final product meets the requirements without interruptions.

Summary of Requirements

- Project will be an executable (.exe)
- The design (UI) will be easy for someone to use
- The user experience (UX) will be a very simple and trivial
- The executable should run easily, even on a lower-end system
- The project should not exceed an hour's worth of runtime
- Differentiate weight of conflict

Applicable Courses from Iowa State University Curriculum

SE 185	SE 309
COM S 227	COM S 311
COM S 228	SE 317
COM S 363	SE 339
SE 319	SE 329

New Skills/Knowledge acquired that was not taught in courses

New Skills/Knowledge acquired that was not taught in courses

Internship experience – managing interns and interning at Wells Fargo

Personal projects – Embedded programming, hosting a server, playing around with Iowa State curriculum concepts on our own

Team

1.1 Team Members

Ryan Tullis – Team Lead

Dillon Gesy – Team Organizer

Nicolas Figueroa Calderas – Frontend developer

Charles “Joe” Hosier – Frontend developer

Abdullahi Abdullahi – Backend developer

1.2 Required Skill Sets for Your Project

Project & team management

Software skills (Java, JavaScript, Python, C, C#, etc)

Teamwork to combine individual components

Documentation for the project

1.3 Skill Sets covered by the Team

Project & team management: Ryan, Dillon

Software skills: Ryan, Dillon, Nicolas, Joe, Abdul

Teamwork to combine components: Ryan, Dillon, Nicolas, Joe, Abdul

Documentation for the project: Ryan, Dillon, Nicolas

1.4 Project Management Style Adopted by the Team

Waterfall project management mixed in with agile processes for linear, yet adjustable work while we develop it throughout both semesters.

1.5 Initial Project Management Roles

Ryan: Interact with the client to learn requirements and set up group meetings

Dillon: Documentation for what happens during the meetings and assign tasks

Nicolas: Design and come up with/document our potential project designs

Joe: Implement individual components for our chosen project design

Abdul: Implement and test individual component

Project Introduction

2.1 Problem Statement

Our goal is to create a class scheduler visualization tool. We want to be able to use scheduling data professors are given in order to show where conflicts may occur with other classes. It's very difficult to make a schedule good for all students, so this approach focuses on the department or professor. The tool should be used as an aid to professors to help submit their schedule of classes. With our approach, we want this solution to be able to be used even after Access+ is gone. The end goal is to at minimum let professors insert their data manually. If there's time, we can also look into getting the data directly from Workday. Furthermore, allowing users of the application to search by class or professor would be beneficial for the user.

2.2 Requirements and Constraints

The **main** requirement for the project is to make a visualization tool for the client.
Moving on to the **subtask** requirements, they entail the following

- Search by professor
- Search by class
- Good GUI for the user, the users are not software engineers
- Display conflicts and their weights with professors/classes
- Be able to see past semesters
- Have a way of inserting data

Functionally, we want the project to be an app. The app needs to be able to collect data and display it visually in order to help schedule future classes.
Our **resource** requirements depend on the application being an executable. For an app, we need to be able to import and organize data locally, which benefits the user by keeping data private.
Our **GUI** needs to be user friendly so anyone can use the class scheduler. The user experience should be easy for newcomers to pick up and use.
The app should only be available privately, therefore will have **no economic or market requirements.**
For **environmental requirements**, our project does use a server. Therefore, there is little to no environmental impact.
The **UI** needs to be displayed in such a way that the user can do what they need to do easily.
Performance-wise, our app needs to be able to run its tools and algorithms within a reasonable time-frame.
Legally, we need to make sure all information in our database is public information - unless inserted by someone manually.
For **maintainability**, it's left up to Iowa State if they want to keep using the application to help schedule their classes. Only bug fixes should need to be done.
Testing requirements entail making sure the app works using different optional functionalities such as importing data and using old data for visualization.

2.3 Engineering Standards

ISO/IEC 25010:2011 - The software made is of quality and has specific characteristics that entails the quality.
ISO/IEC/IEEE 29148:2018 - The software's requirements will be progressively elaborated on through the project's lifecycle. We will get closer to what the client wants over time.

2.4 Intended Users and Uses

The Department of Electrical and Computer Engineering in the College of Engineering at Iowa State University would benefit the most from our project. The project is specifically intended to make it easier to schedule classes within the Department of Electrical and Computer Engineering. Vicky Thorland-Oster, the Assistant Director of Student Services in the Department of Electrical and Computer Engineering, who is our advisor on the project, is currently in charge of creating the schedule for future semesters. However, soon Tina Prouty will be taking over the process. Vicky stated that the current process in place is very time consuming given all the different requirements and restraints on when classes can and cannot be. The only use case for our given project is to visualize different options of schedules for future semesters, given a set of upcoming classes

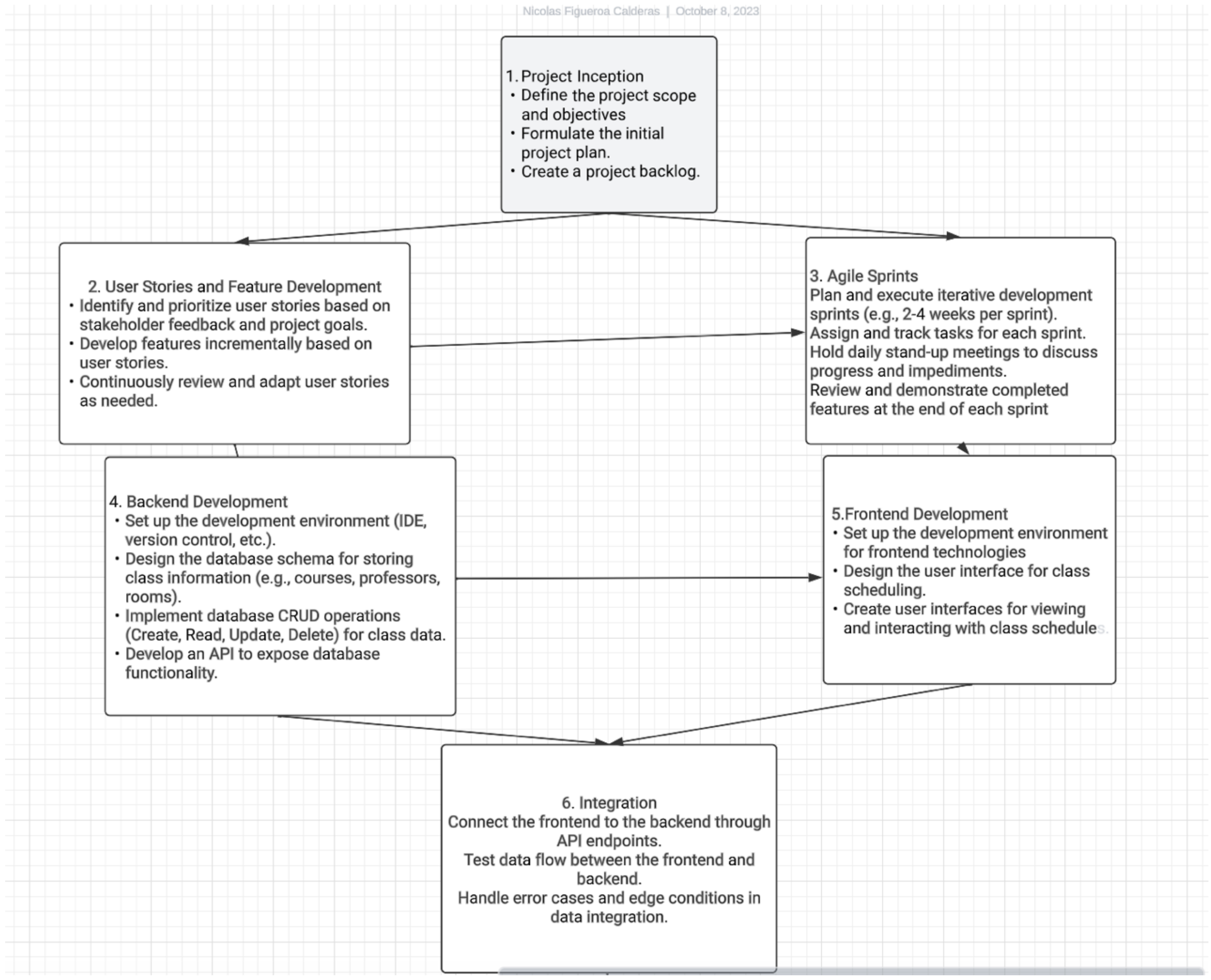
Project Plan

3.1 Project Management/Tracking Procedures

We will use an agile project management style because it is the one that most of us in our group feel comfortable with. We want to have a minimum viable product that we can show at the end of the course, agile will help us to work in sprints and be able to create a product that we can demo. Agile will help us create the base of our project and then allow us to continue to add onto it with additional sprints depending on the time available to us before the end of the course.

We will utilize Gitlab, Trello, and Discord for our project management, with Gitlab and Discord being the main focus.

3.2 Task Decomposition


Nicolas Figueroa Calderas | October 8, 2023
1. Project Inception
• Define the project scope and objectives
• Formulate the initial project plan.
• Create a project backlog.
2. User Stories and Feature Development
• Identify and prioritize user stories based on stakeholder feedback and project goals.
• Develop features incrementally based on user stories.
• Continuously review and adapt user stories as needed.
3. Agile Sprints
Plan and execute iterative development sprints (e.g., 2-4 weeks per sprint).
Assign and track tasks for each sprint.
Hold daily stand-up meetings to discuss progress and impediments.
Review and demonstrate completed features at the end of each sprint
4. Backend Development
• Set up the development environment (IDE, version control, etc.).
• Design the database schema for storing class information (e.g., courses, professors, rooms).
• Implement database CRUD operations (Create, Read, Update, Delete) for class data.
• Develop an API to expose database functionality.
5.Frontend Development
• Set up the development environment for frontend technologies
• Design the user interface for class scheduling.
• Create user interfaces for viewing and interacting with class schedules.
6. Integration
Connect the frontend to the backend through API endpoints.
Test data flow between the frontend and backend.
Handle error cases and edge conditions in data integration.

3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Key milestones for the project can help track progress and ensure that the project stays on course. Here are some key milestones for the project based on the dependency graph, along with how to measure progress on each task:

1. **Project Kickoff and Inception (Milestone 1)**

- Tasks: Project Inception (Task 1)
- Measurement:
    - Completion of project scope document.
    - Approval of the initial project plan and backlog.

2. **Backend Development Completion (Milestone 2)**
   - Tasks: Backend Development (Task 2)
   - Measurement:
     - Successful setup of the development environment.
     - Completion of database schema design.
     - Implementation of CRUD operations for class data.
     - Availability of a functional API.

3. **Frontend Development and Integration (Milestone 3)**
   - Tasks: Frontend Development (Task 3) and Integration (Task 4)
   - Measurement:
     - User interface design completion.
     - Implementation of user authentication and authorization.
     - Successful integration of the frontend and backend.
     - Initial testing of data flow.

4. **User Stories and Feature Development (Milestone 4)**
   - Tasks: User Stories and Feature Development (Task 5)
   - Measurement:
     - Prioritized user stories.
     - Incremental feature development.
     - Regular sprint reviews with functional features.

5. **Agile Sprints (Sprint Milestones)**
   - Task: Agile Sprints (Task 6)
   - Measurement:
     - Completion of planned sprint tasks.
     - Daily stand-up meetings held consistently.
     - Review meetings at the end of each sprint with demonstrated features.

6. **Testing and Quality Assurance (Milestone 5)**
   - Task: Testing (Task 7)
   - Measurement:
     - Execution of test cases for frontend and backend.
     - Successful integration testing.
     - Usability testing with feedback and necessary adjustments.

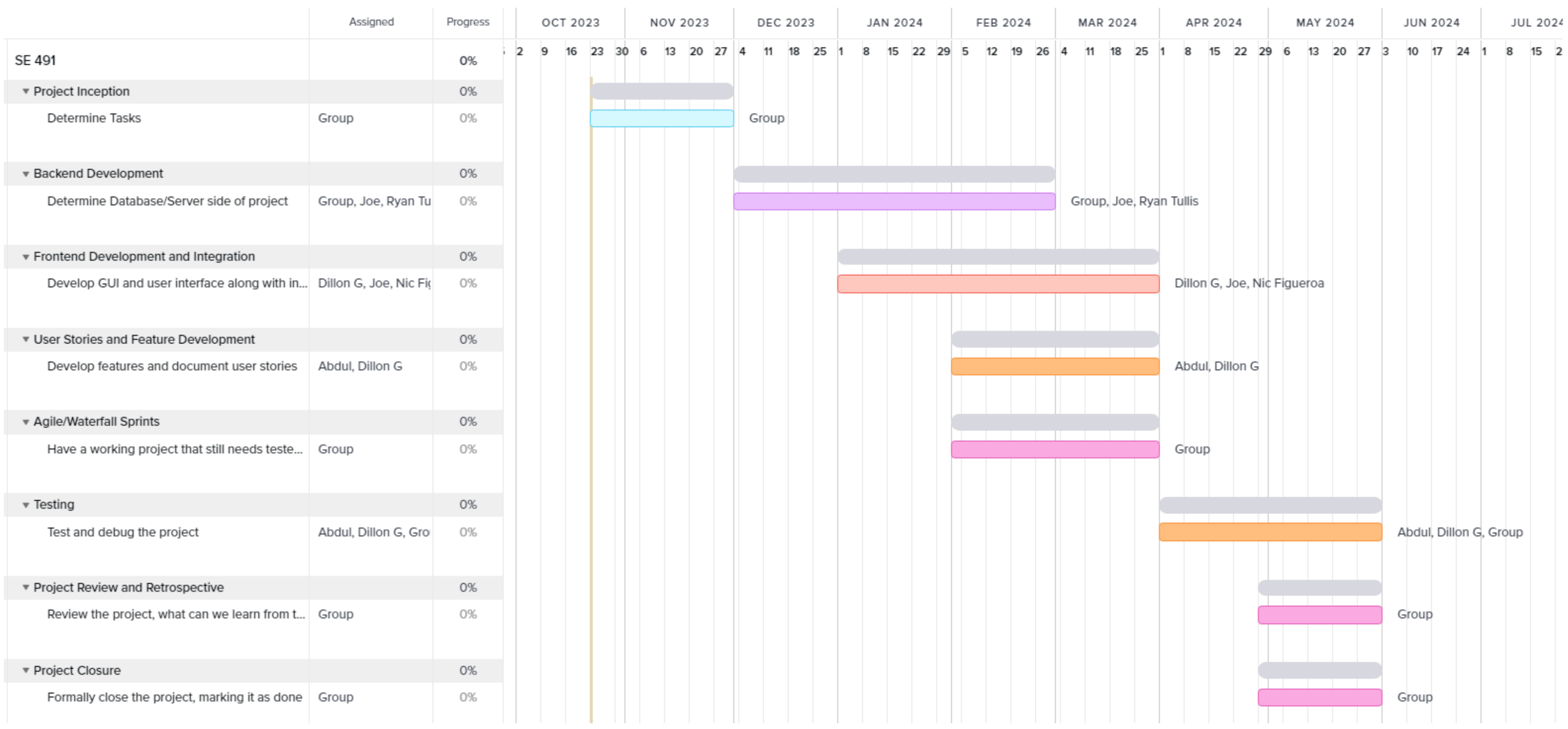
7. **Project Review and Retrospective (Milestone 6)**
   - Task: Project Review and Retrospective (Task 8)
   - Measurement:
     - Successful project review with objectives met.
     - Action items identified in the retrospective for improvement.
     - Archived project files and documentation.

8. **Project Completion (Milestone 7)**
- Measurement:
    - Final stakeholder meeting held.
    - Official closure of the project.
    - Evaluation of the project's success based on feedback and metrics.

3.4 Project Timeline/Schedule

Project: College Class Scheduler

Task	Start Date	End Date	Duration
Project Inception	Oct 2023	Nov 2023	2 months
Backend Development	Dec 2023	Feb 2024	3 months
Frontend Development and Integration	Jan 2024	Mar 2024	3 months
User Stories and Feature Development	Feb 2024	Mar 2024	2 months
Agile/Waterfall Sprints	Feb 2024	Mar 2024	2 months
Testing	Apr 2024	May 2024	1 month
Project Review and Retrospective	May 2024	May 2024	1 month
Project Closure	May 2024	May 2024	1 month



	Assigned	Progress
SE 491		0%
▾ Project Inception		0%
Determine Tasks	Group	0%
▾ Backend Development		0%
Determine Database/Server side of project	Group, Joe, Ryan Tu	0%
▾ Frontend Development and Integration		0%
Develop GUI and user interface along with in...	Dillon G, Joe, Nic Fi	0%
▾ User Stories and Feature Development		0%
Develop features and document user stories	Abdul, Dillon G	0%
▾ Agile/Waterfall Sprints		0%
Have a working project that still needs teste...	Group	0%
▾ Testing		0%
Test and debug the project	Abdul, Dillon G, Gro	0%
▾ Project Review and Retrospective		0%
Review the project, what can we learn from t...	Group	0%
▾ Project Closure		0%
Formally close the project, marking it as done	Group	0%

Fig. 1: A schedule of our project

3.5 Risks and Risk Management/Mitigation

1. **Project Inception**
   - Risk: Can’t agree on project backlog (Probability: Low)
   - Mitigation: Conduct team vote to determine what the project backlog will be

2. **Backend Development**
   - Risk: Unexpected technical challenges in database design (Probability: Medium)
   - Mitigation: Seek out TA help to help process of creating a database design and ask professor for suggestions as well

3. **Frontend Development and Integration**
   - Risk: Integration difficulties between frontend and backend (Probability: High)
   - Mitigation: Seek out TA help to help process of connecting frontend with backend and ask professor for suggestions as well

4. **User Stories and Feature Development**
   - Risk: Changing requirements (Probability: Low)
   - Mitigation: Change user stories and focus more on agile methodology to help keep things on track

5. **Agile Sprints**
   - Risk: Incomplete sprint goals due to underestimated complexity (Probability: Low)
   - Mitigation: Prioritize user stories accurately, and put more time during the week to complete user stories

6. **Testing**
   - Risk: Insufficient time for comprehensive testing (Probability: Medium)
   - Mitigation: Allocate more time for testing in the project plan. Automate repetitive tests to save time.

7. **Project Review and Retrospective**
   - Risk: Incomplete final review (Probability: Low)
   - Mitigation: Plan documentation for college of computer and electrical engineering faculty to use to understand class scheduler.

3.6 Personnel Effort Requirements

Task: Project Inception

Task Description	Estimated Effort (Person-Hours)
Define project scope, objectives, stakeholders	40
Formulate initial project plan	20
Create a project backlog	30
Total Effort	**90 Person-Hours**

Explanation: In the project inception phase, you'll need to spend significant time defining project requirements and goals (40 hours), planning the project (20 hours), and creating a backlog of tasks (30 hours). These efforts are crucial for setting the project's direction.

Task: Backend Development

Task Description	Estimated Effort (Person-Hours)
Set up the development environment	40
Design the database schema	60
Implement database CRUD operations	80
Develop an API to expose database functionality.	100
Total Effort	**280 Person-Hours**

Explanation: Backend development involves setting up the environment (40 hours), designing a robust database schema (60 hours), implementing CRUD operations (80 hours), and creating an API (100 hours), which is a substantial effort.

Task: Frontend Development and Integration

Task Description	Estimated Effort (Person-Hours)
Set up the development environment	40
Design the user interface	80
Implement user authentication and authorization	60
Create user interfaces for class schedules	100
Connect frontend to the backend	80
Test data flow and handle errors	60
Total Effort	**420 Person-Hours**

Explanation: Front-end development and integration require setting up the environment (40 hours), designing a user-friendly interface (80 hours), implementing security features (60 hours), creating user interfaces (100 hours), and ensuring proper integration and testing (80 hours).

Task: User Stories and Feature Development

Task Description	Estimated Effort (Person-Hours)
Identify and prioritize user stories	40
Develop features incrementally	160
Review and adapt user stories as needed	60
Total Effort	**260 Person-Hours**

Explanation: User stories and feature development involve identifying and prioritizing user stories (40 hours), developing features incrementally (160 hours), and continuously reviewing and adapting user stories (60 hours) to align with project goals.

Task: Agile Sprints

Task Description	Estimated Effort (Person-Hours)
Plan and execute iterative development sprints	120
Assign and track tasks for each sprint	60
Hold daily stand-up meetings	40
Review and demonstrate completed features	80
Total Effort	**300 Person-Hours**

Explanation: Agile sprints involve planning and executing sprints (120 hours), task tracking (60 hours), daily stand-up meetings (40 hours), and feature reviews (80 hours) for iterative development.

Task: Testing

Task Description	Estimated Effort (Person-Hours)
Develop and execute test cases	120
Perform integration testing	80
Conduct usability testing	60
Total Effort	**260 Person-Hours**

Explanation: Testing tasks include developing and executing test cases (120 hours), integration testing (80 hours), and usability testing (60 hours) to ensure the system's quality and usability.

Task: Project Review and Retrospective

Task Description	Estimated Effort (Person-Hours)
Conduct a project review	40
Hold a team retrospective	40
Total Effort	**80 Person-Hours**

Explanation: Project review and retrospectives involve assessing project objectives (40 hours) and discussing team performance and improvements (40 hours).

Task: Project Closure

Task Description	Estimated Effort (Person-Hours)
Archive project files and documentation	40
Conduct a final stakeholder meeting	40
Evaluate the project's success	20
Total Effort	**100 Person-Hours**

Explanation: Project closure tasks include archiving project files and documentation (40 hours), holding a final stakeholder meeting (40 hours), and evaluating the project's success (20 hours).

Total Project Effort: **1,900 Person-Hours**

Project Design

4.1 Design Content

The design content in the project plan is primarily focused on Milestones 2 and 3, which involve the development and integration of the backend and frontend of the project:

1. **Backend Development Completion (Milestone 2):** This milestone includes the design of the database schema, which is a crucial aspect of the project's design. It also involves the successful setup of the development environment, which may include designing the architecture of the backend system. Additionally, the implementation of CRUD (Create, Read, Update, Delete) operations for class data implies designing the API endpoints and associated data structures.

2. **Frontend Development and Integration (Milestone 3):** This milestone encompasses the design of the user interface, which is a key aspect of the project's design content. It also involves the implementation of user authentication and authorization, which includes designing access control mechanisms. The successful integration of the frontend and backend implies the design of the data flow and the interaction between these components.

Overall, the design content in the project plan includes database schema design, API design, user interface design, and system architecture design to ensure the project's successful development and integration.

4.2 Design Complexity

To provide evidence that a project is of sufficient technical complexity, we can evaluate the project against the provided metrics and justify the complexity using specific components, subsystems, and technical principles. Let's consider these **two** metrics:

1. The design consists of multiple components/subsystems that each utilize distinct scientific, mathematical, or engineering principles:

In the project plan described earlier, several components and subsystems are involved, each requiring distinct technical principles:

- **Backend Development (Milestone 2):** This component involves designing the database schema, which requires knowledge of database management and design principles. Additionally, the implementation of CRUD operations necessitates understanding of software engineering and API design.

- **Frontend Development and Integration (Milestone 3):** The frontend development component involves user interface design, which includes principles of user experience (UX) design and user interface (UI) design. The integration task requires knowledge of software architecture and system integration principles.

- **User Stories and Feature Development (Milestone 4):** The development of user stories and features demands a deep understanding of software engineering, software design patterns, and coding practices.

- **Agile Sprints (Sprint Milestones):** The Agile methodology involves various components, including daily stand-up meetings and review meetings, each of which follows specific Agile principles and practices.

- **Testing and Quality Assurance (Milestone 5):** This phase includes the execution of test cases, integration testing, and usability testing, all of which involve distinct testing methodologies and quality assurance principles.

2. The problem scope contains multiple challenging requirements that match or exceed current solutions or industry standards:

The project involves several challenging requirements that can be considered to match or exceed current industry standards:

- The development of a functional API (Milestone 2) is a challenging requirement that needs to meet industry standards for API design and performance.

- User authentication and authorization (Milestone 3) require robust security measures, which are crucial to match industry standards for data protection and user access control.

- Usability testing (Milestone 5) goes beyond basic functionality testing and aims to match industry standards for user experience and usability, which can be highly demanding.

- Regular sprint reviews (Sprint Milestones) with functional features demonstrate a commitment to delivering high-quality software, which matches industry standards for Agile development.

- Project review and evaluation (Milestone 7) based on feedback and metrics aims to exceed industry standards by continuously improving the project based on data-driven insights.

4.3 Modern Engineering Tools

Simple software engineering tools such as databases and IDEs.
SQLite database file - to store new and old data securely and query it in the executable
IDE - connecting the code language to machine code
    Eclipse - Java IDE with a GWT Designer integration
GUI - Google Web Toolkit, uses Java
    GWT Designer - Creation of user interfaces

4.4 Design Context

Area	Description	Examples
Public health, safety, and welfare	Helps the Iowa State University community, specifically instructors but can help students and other faculty as well.	Provides instructors more context when they schedule classes and may help students sign up for classes earlier in the future.
Global, cultural, and social	We reflect all of Iowa State University's standard practices.	All practices revolving around the project should be kept up to standard and will conform to all Iowa State University's practices.
Environmental	The only environmental impact our project has is energy consumption.	If the user doesn't have a device to run the application, they won't be able to use it. In turn, the device requires energy.
Economic	There is no economic impact to this project.	The application will be free and safe to use. There will be no chance of data being leaked due to security flaws.

Public Health, Safety, and Welfare:

N/A

Global, Cultural, and Social:

N/A

Environmental:

N/A

Economic:

- Economic considerations include if the class scheduler can improve the amount of time and resources used to create a schedule without any conflicts.

4.5 Prior Work/Solutions

The projects that have already been done that are similar to ours are other scheduling systems. These systems are similar but try to create an algorithm to actually schedule the classes. Furthermore, they use already-developed algorithms or libraries to help do this. However, we just want to display conflicts graphically. Most of these systems include a UI with some form of data that can be rearranged. This is what we plan to do as well, just using the data given to us by the College of Computer and Electrical Engineering.

4.6 Design Decisions

Our data will be stored in a .sqlite file, which can be queried using Java. Specifically, we will use the JDBC (Java Database Connectivity) API with the SQLite JDBC driver.

We will implement features like class overlap detection, severity tiers for the overlaps, visual image of all the scheduled classes, and a search function for class and professor.

4.7 Proposed Design

We looked at different schedulers to get an idea of how we want to implement different features. We looked at the UX(user experience) of these schedules to see if it was easy to use and on how efficient they were. Some of the schedulers were too complex.

4.7.1 Design 0 (Initial Design)

Design Visual and Description

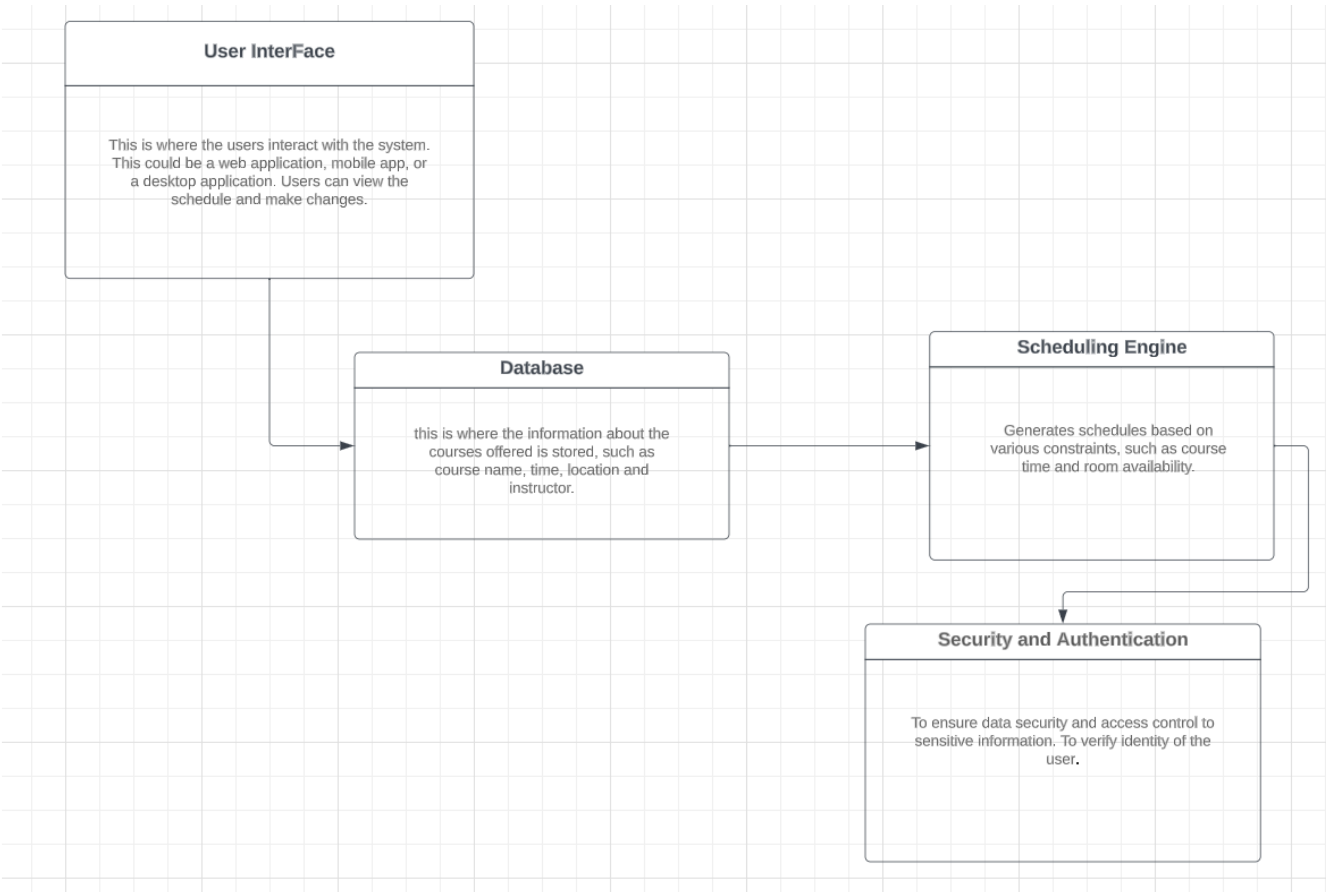

User InterFace
This is where the users interact with the system. This could be a web application, mobile app, or a desktop application. Users can view the schedule and make changes.
Database
this is where the information about the courses offered is stored, such as course name, time, location and instructor.
Scheduling Engine
Generates schedules based on various constraints, such as course time and room availability.
Security and Authentication
To ensure data security and access control to sensitive information. To verify identity of the user.

Fig. 2: An early design concept for our application

User Interface: This is where the users interact with the system. This could be a web application, mobile app, or a desktop application. Users can view the schedule and make changes.
Database: This is where the information about the courses offered is stored, such as course name, time, location and instructor.
Security and Authentication: To ensure data security and access control to sensitive information. To verify the identity of the user.
Scheduling Engine: Generates schedules based on various constraints, such as course time and room availability.

Functionality
The class scheduler would run on a computer by the ECpE department. It will schedule the classes and labs offered depending on time and room availability. It satisfies the requirements for this project.

4.7.2 Design 1 (Design Iteration)

Design Visual and Description

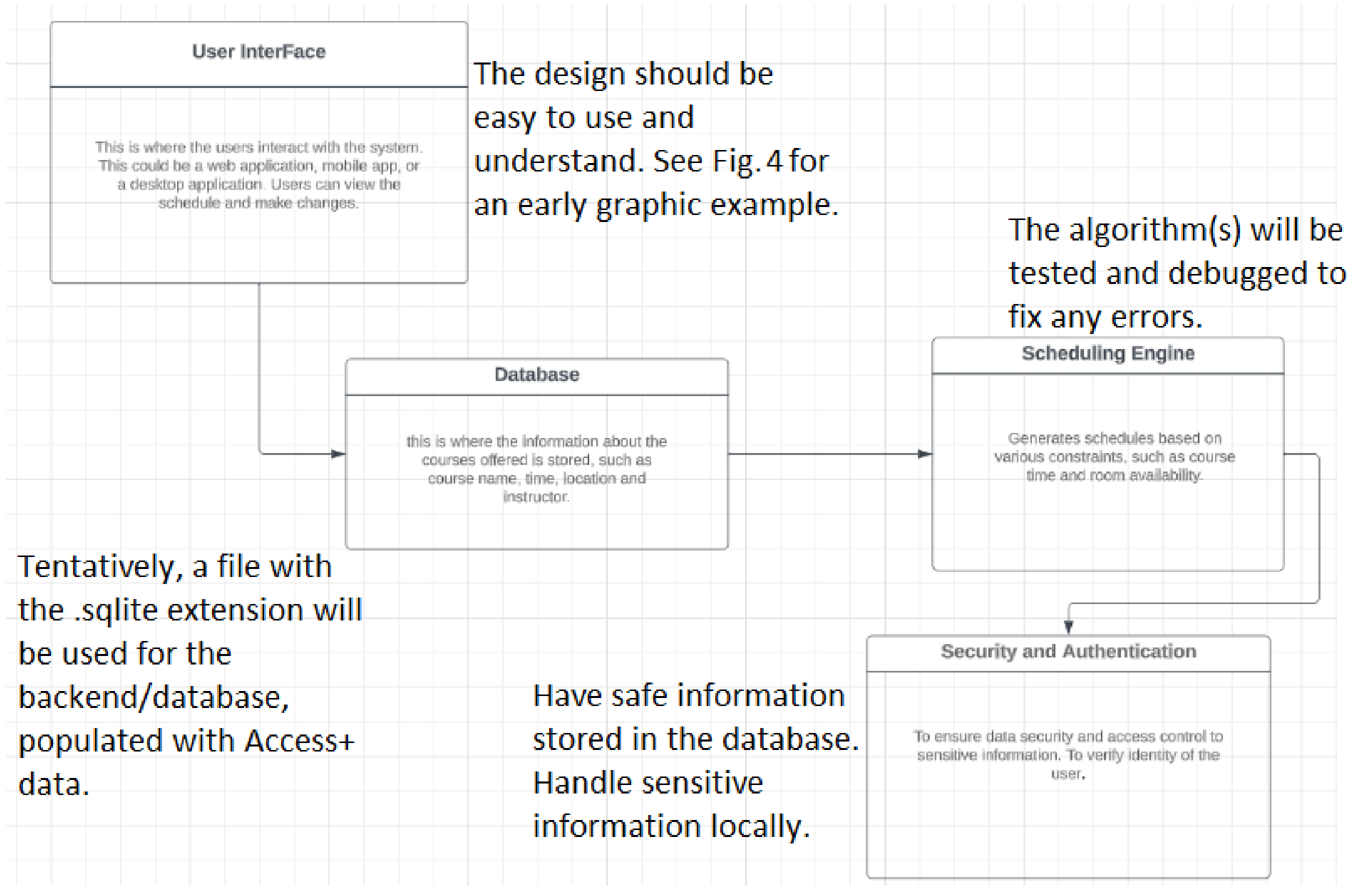

User InterFace
This is where the users interact with the system. This could be a web application, mobile app, or a desktop application. Users can view the schedule and make changes.
The design should be easy to use and understand. See Fig. 4 for an early graphic example.
The algorithm(s) will be tested and debugged to fix any errors.
Database
this is where the information about the courses offered is stored, such as course name, time, location and instructor.
Scheduling Engine
Generates schedules based on various constraints, such as course time and room availability.
Tentatively, a file with the .sqlite extension will be used for the backend/database, populated with Access+ data.
Have safe information stored in the database. Handle sensitive information locally.
Security and Authentication
To ensure data security and access control to sensitive information. To verify identity of the user.

Fig. 3: Our design 0 but evolved iteratively

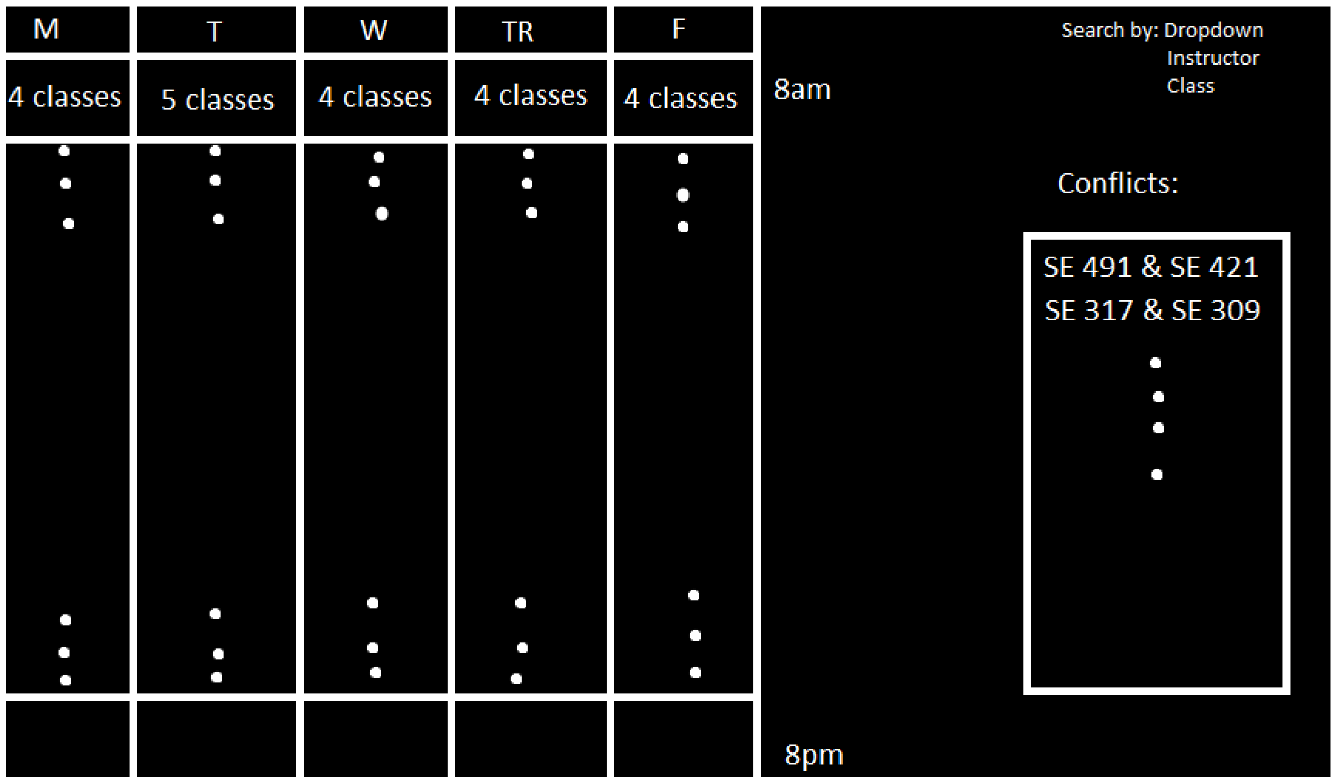


Fig. 4: Early design and rough sketch of a sample interface

The Design 1 goes into more detail on what tools we are using and how we are going to use them.
Using fig. 4, we want to display our data graphically and in an ideal way.
One thing missing from fig. 4 is implementing a way to fix scheduling conflicts.

A majority of design 1 will be the same as design 0, but design 1 will have more specific parts such as the frontend and backend. For example, an agreed upon user interface will be in place, an agreed upon database will be used, and an algorithm for the scheduling engine will be more tested.

4.8 Technology Considerations

While later it is listed as a weakness, not having a server is also a **strength**. By not using a server, we do not have to use dedicated resources for hosting our project. Therefore, we can disregard the hassle of setting up the server. This lessens our work and can keep us focused on the main goal of the project.

Since we are not using a server to host our application, our main **weakness** is the availability of the project. Our current goal of the project is to at least be able to store the project on a flash drive and copy it. However, this is not a good way to share the executable to the instructors.

Some **trade-offs** we had to make could include things such as our tech stack. Not everyone on the team has the same skill set when it comes to different coding languages. While some languages allow us to be more efficient than others, we find it more important to work with languages that everyone knows or can quickly pick up.

4.9 Design Analysis

We predict our design from 4.7 will not work right away. However, it will be a great start to making progress in our project. Changes will most likely need to be made in order to get the project to be functional.
To iterate the design again, while implementing, we will notice some things will be easier to implement over others. Using this, we will come up with more features and ways to deal with conflicts. We also predict we will have to make some trade-offs when it comes to adding and removing features.

Project Testing

5.1 Unit Testing

The individual classes for our user interface and our scheduling engine will be tested as we develop each individual unit. We will use the selenium testing framework for the frontend and Junit for the backend.

5.2 Interface Testing

The frontend user interface will be tested as a whole, by itself and with API calls to the backend. The back end scheduling engine will be tested as a whole using Junit.

5.3 Integration Testing

Critical integration paths for our design are our frontend UI and our backend scheduling engine. These will be the most critical of our integration paths and we will test the functions and interfaces, testing for end to end functionality. We will use Selenium and Junit.

5.4 System Testing

System level testing is taking the whole application and testing how different components of the application interact with one another. All of our unit tests, interface tests, and integration tests will be used as building blocks for our system testing. Since we will be testing the entire system's functionality, Selenium is our best available option, since we will be testing straight from the UI and seeing if the functionality is at 100%.

5.5 Regression Testing

Before finalizing new additions and/or changes to the code, all old functionality must be tested with the new additions to ensure the code isn't broken. There will also be test cases for each of the functionality to alert which of the old functionality might be broken.

5.6 Acceptance Testing

We will go over the design, functional and nonfunctional requirements that the client gave us to see if each criteria has been met. The client tested out the product to see if all requirements are met. We will also have users test out the system and give feedback in order to improve the system.

5.7 Security Testing (if applicable)

The data we are using is general information about classes and labs. Any private information will be stored locally on the device, which does not entail any security on our end.

5.8 Results

The results of our testing should make sure the executable successfully implements the requirements in a way the user can understand easily. By testing all features, we can ensure that the software meets the requirements. For our process, we will be using functional testing. Furthermore, since we will all be working on the development process for the project, we will be doing white box testing.

Search by professor	Selenium/Junit testing - all classes pertaining to a specific professor must be displayed
Search by class	Selenium/Junit testing - all class data, such as professors and times, must be displayed
Good GUI for user	Have the client, faculty, and other people use the software to provide a critique of the display. Allow for suggestions so it can be changed
Display conflicts and weights	Selenium/Junit testing - the algorithm(s) we make should have the correct conflict weight, and should be displayed correctly

Search by past semester	Selenium testing/Junit testing - classes, professors, and previous times should be displayed for past semesters, in order to guide a current schedule
Insertion of data	Manual testing, insert single pieces of data, groups of data, single piece of faulty data, and a group of faulty and true data

Project Implementation

6 Implementation

In order to implement our project with a strong user interface/user experience, we need to use a strong GUI for our application. To our knowledge, implementing the backend and filling the data will be a trivial and monotonous task. In contrast, building the frontend and displaying the data graphically will be the most crucial design implementation of our project.

Project Professionalism

7 Professionalism

This discussion is with respect to the paper titled “Contextualizing Professionalism in Capstone Projects Using the IDEALS Professional Responsibility Assessment”, International Journal of Engineering Education Vol. 28, No. 2, pp. 416–424, 2012

7.1 Areas of Responsibility

Area of Responsibility	**Definition**	**Code and Context**
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	SE 8.02 - Refers to the software engineers improving their skills and honing in on quality software, with reasonable cost and time. This code differs as it focuses on the individual, rather than the team.

Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	SE 1.14 - Refers to software engineers promoting maximum quality with the minimum cost. Finding the balance between cost and quality is crucial and generally the same for both codes.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	SE 4.06 - Refers to being honest about reporting to the client/stakeholder any problems with the software or related documents. The two are similar in a way that it means clear, honest communication with the stakeholder is crucial.
Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	SE 4.05 - Refers to keeping confidential information confidential from the public. This reduces the stakeholder/client's risk, as information that should be private needs to stay private. This differs as the SE code is more about the information, rather than the stakeholder.
Property Ownership	Respect property, ideas, and information of clients and others.	SE 4.04 - States that the software engineers should not knowingly use illegally obtained or retained software. By doing this, you respect the property/ideas of others. The difference between the two is that this deals explicitly with using software you shouldn't.

Sustainability	Protect environment and natural resources locally and globally.	SE 1.13 - Refers to identifying and acknowledging any ethical, economic, cultural, legal, and environmental issues formally. The two are relatively the same, but the global scale is more limited and SE 1.13 is more of a general concern.
Social Responsibility	Produce products and services that benefit society and communities.	SE 6.11 - Exercise professional responsibility to society by constructively serving in civic affairs. The code is different because one deals with benefiting society and the other is more about not causing harm to society.

7.2 Project Specific Professional Responsibility Areas

Area of Responsibility	**Definition**	**Application**
Work Competence	Perform work of high quality, integrity, timeliness, and professional competence.	This applies because all members should contribute to the project equally. Our team is performing “medium” well in this area. All work is eventually done, but in a lesser-organized manner.
Financial Responsibility	Deliver products and services of realizable value and at reasonable costs.	This does not apply to our project as we do not have a financial impact. Therefore, our project performance in this area is N/A.
Communication Honesty	Report work truthfully, without deception, and understandable to stakeholders.	This area of responsibility applies to our project due to its group-work nature. While there are no explicit stakeholders aside from the client, we need to report our work to each other in the same way. Our group has a “high” performance in this area, as we communicate our work clearly with each other.

Health, Safety, Well-Being	Minimize risks to safety, health, and well-being of stakeholders.	This does not apply to our project context because there is no risk to safety, health, or well-being of our client. Our performance is N/A.
Property Ownership	Respect property, ideas, and information of clients and others.	This applies to our project, but only minimally. Since we have to be cautious with future class data, we have to respect that information and handle it accordingly. Our group has come up with solutions to handle the data, therefore the performance is high.
Sustainability	Protect environment and natural resources locally and globally.	This does not apply to our project. We have no sort of emission or power usage for a server in our project. Our project performance is N/A.
Social Responsibility	Produce products and services that benefit society and communities.	This area of responsibility applies to our project because it impacts the Iowa State Community. Specifically, it has an impact on the professors in the Electrical and Computer engineering department. By nature of the project, we have a high performance in this area.

7.3 Most Applicable Professional Responsibility Area

The Communication Honesty area of responsibility is the most applicable for our project. It is key that as a group we communicate honestly and thoroughly in order to plan and complete the project. Without Communication Honesty, our ideas would not be able to connect and come together. It is integral to maintain a high standard of this throughout the project's lifespan.

Project Closing

8 Closing Material

The following describes the formal closure of the project.

8.1 Discussion

The formal closure of this project entails that either all requirements were met or tradeoffs have occurred. It is entirely possible some things will not be achieved throughout this project's lifecycle. For our minimum product, we expect to sort/search our data by professor, class, or year. We also expect a functional GUI to assist professors in the scheduling of their classes.

8.2 Conclusion

Our work should be working towards the requirements defined in section 2.2. Our overarching goal is to create an interactive GUI to schedule classes. Our best plan of action to achieving this is by following our schedule, defined in 3.4. If we find the goals are harder to achieve than expected, we can solve the issue in the following 2 ways:

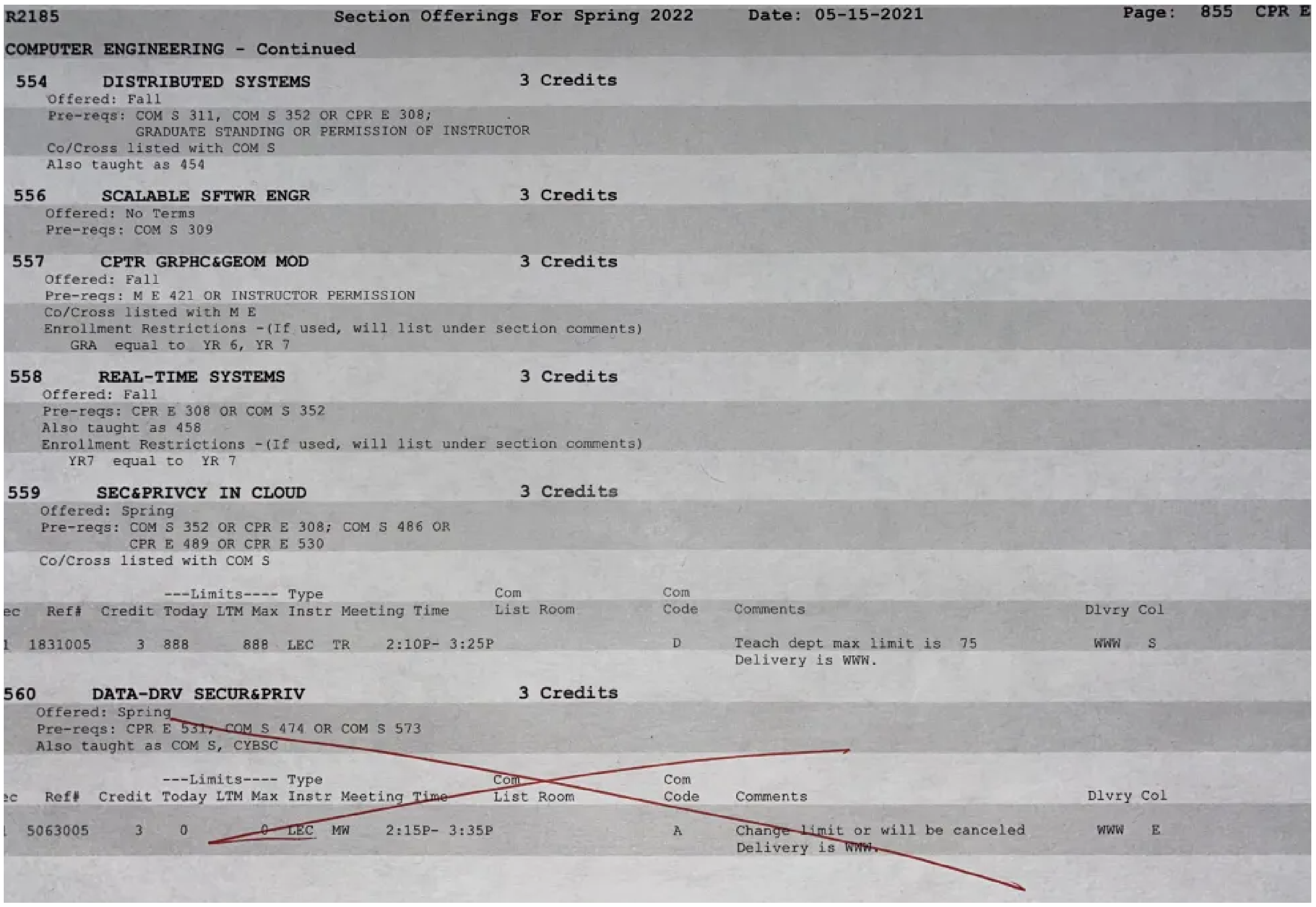
- Put more hours and work into the project
- Modify the requirements and make tradeoffs

In the future, it will be easier to design and plan a project due to our experience. Better planning, familiarizing ourselves with the content better, and more background/routine tasks such as setting up prototypes are some examples that could help achieve the goals in a smoother way.

8.3 References

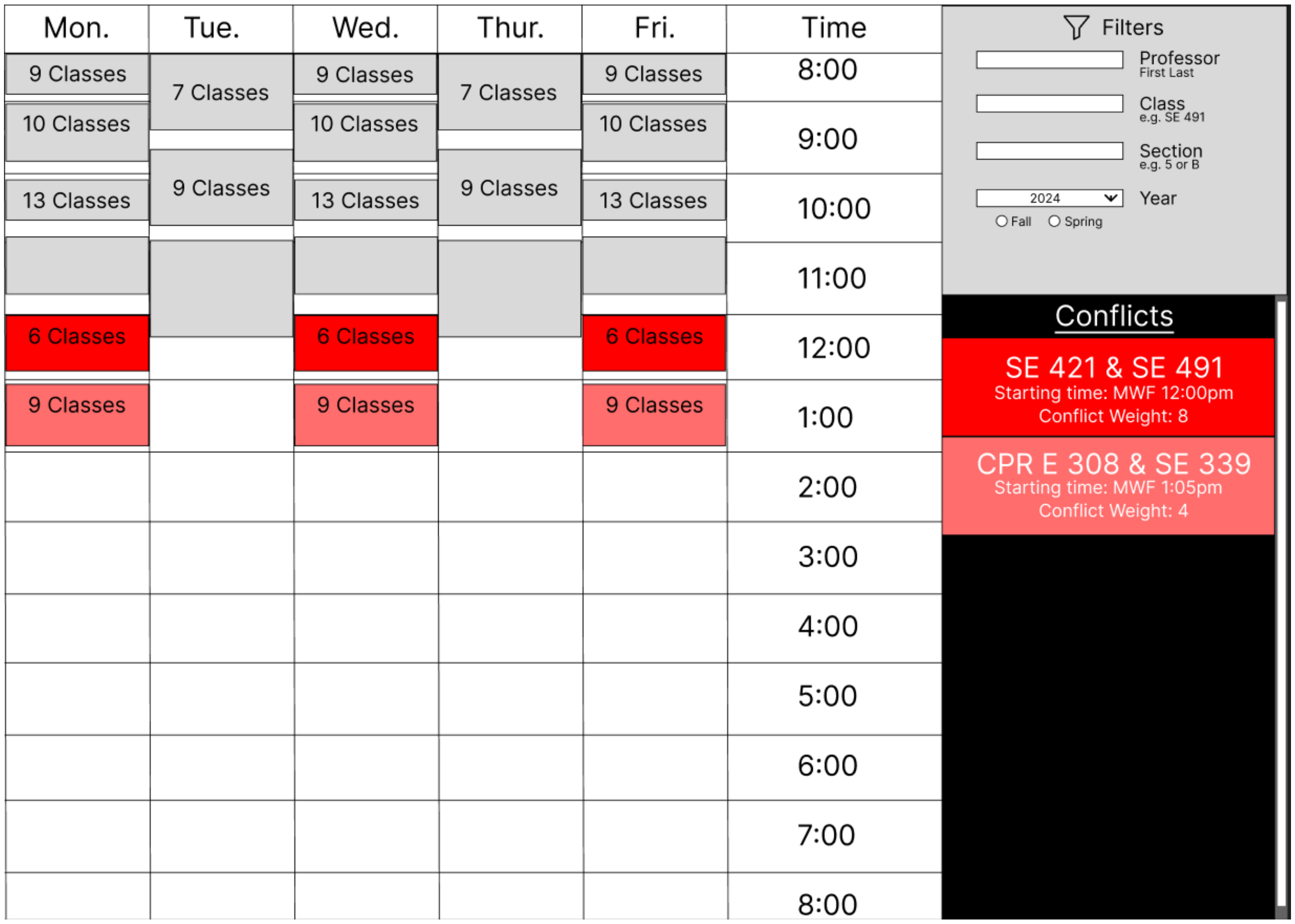
None at this point in time.

8.4 Appendices



```
R2185                        Section Offerings For Spring 2022     Date: 05-15-2021                  Page:  855  CPR E
COMPUTER ENGINEERING - Continued
 554     DISTRIBUTED SYSTEMS                          3 Credits
   Offered: Fall
   Pre-reqs: COM S 311, COM S 352 OR CPR E 308;
             GRADUATE STANDING OR PERMISSION OF INSTRUCTOR
   Co/Cross listed with COM S
   Also taught as 454

 556     SCALABLE SFTWR ENGR                          3 Credits
   Offered: No Terms
   Pre-reqs: COM S 309

 557     CPTR GRPHC&GEOM MOD                          3 Credits
   Offered: Fall
   Pre-reqs: M E 421 OR INSTRUCTOR PERMISSION
   Co/Cross listed with M E
   Enrollment Restrictions -(If used, will list under section comments)
      GRA  equal to  YR 6, YR 7

 558     REAL-TIME SYSTEMS                            3 Credits
   Offered: Fall
   Pre-reqs: CPR E 308 OR COM S 352
   Also taught as 458
   Enrollment Restrictions -(If used, will list under section comments)
      YR7  equal to  YR 7

 559     SEC&PRIVCY IN CLOUD                          3 Credits
   Offered: Spring
   Pre-reqs: COM S 352 OR CPR E 308; COM S 486 OR
             CPR E 489 OR CPR E 530
   Co/Cross listed with COM S

                  ---Limits---- Type                    Com                 Com
ec   Ref#  Credit Today LTM Max Instr Meeting Time      List Room           Code   Comments                          Dlvry Col

1  1831005     3  888      888  LEC  TR    2:10P- 3:25P                      D     Teach dept max limit is  75        WWW   S
                                                                                   Delivery is WWW.

560     DATA-DRV SECUR&PRIV                           3 Credits
   Offered: Spring
   Pre-reqs: CPR E 531; COM S 474 OR COM S 573
   Also taught as COM S, CYBSC

                  ---Limits---- Type                    Com                 Com
ec   Ref#  Credit Today LTM Max Instr Meeting Time      List Room           Code   Comments                          Dlvry Col

1  5063005     3    0        0  LEC  MW    2:15P- 3:35P                      A     Change limit or will be canceled   WWW   E
                                                                                   Delivery is WWW.
```

Fig. 1: An example of what the professors receive when scheduling their classes.



SE 421 & SE 491
Starting time: MWF 12:00pm
Conflict Weight: 8

CPR E 308 & SE 339
Starting time: MWF 1:05pm
Conflict Weight: 4

Fig. 2: A prototype GUI
https://www.figma.com/proto/aBOqXGELgJ1MLCjfMjye2I/Sample-Scheduler-Interface?type=design&node-id=101-2&t=KNYjL2ekRsKfh7k1-0&scaling=min-zoom&page-id=0%3A1

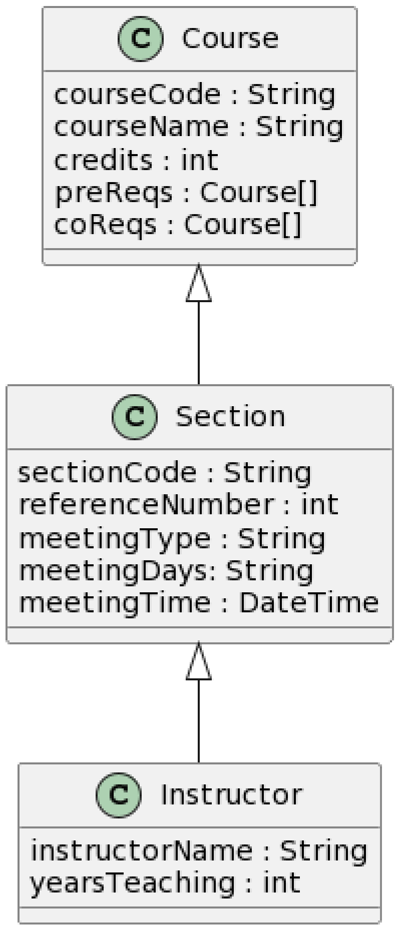
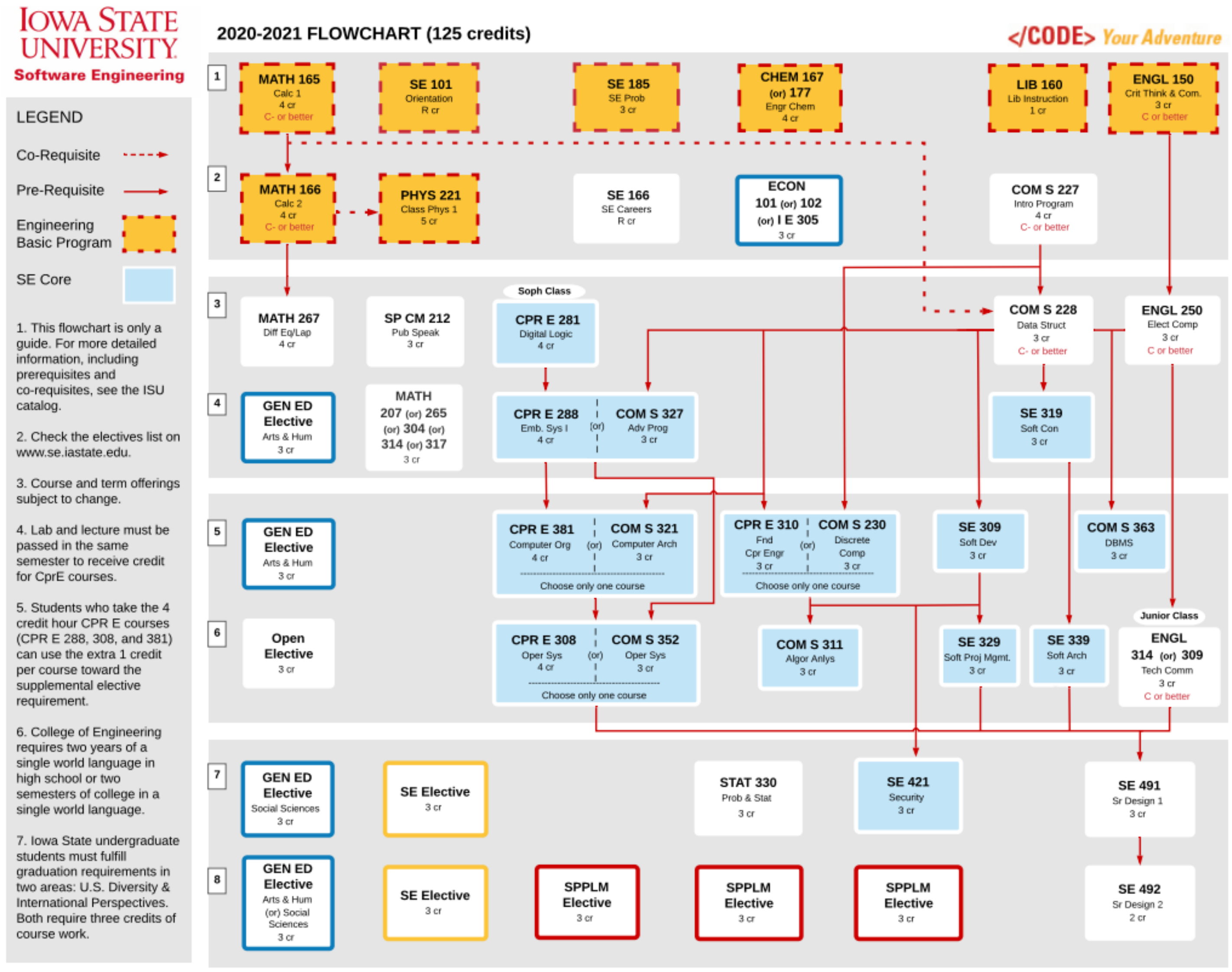


Fig. 3: Our UML Diagram for our data


IOWA STATE UNIVERSITY
Software Engineering
2020-2021 FLOWCHART (125 credits)
</CODE> Your Adventure
LEGEND
Co-Requisite
Pre-Requisite
Engineering Basic Program
SE Core
1
MATH 165 Calc 1 4 cr C- or better
SE 101 Orientation R cr
SE 185 SE Prob 3 cr
CHEM 167 (or) 177 Engr Chem 4 cr
LIB 160 Lib Instruction 1 cr
ENGL 150 Crit Think & Com. 3 cr C or better
2
MATH 166 Calc 2 4 cr C- or better
PHYS 221 Class Phys 1 5 cr
SE 166 SE Careers R cr
ECON 101 (or) 102 (or) I E 305 3 cr
COM S 227 Intro Program 4 cr C- or better
3
MATH 267 Diff Eq/Lap 4 cr
SP CM 212 Pub Speak 3 cr
Soph Class
CPR E 281 Digital Logic 4 cr
COM S 228 Data Struct 3 cr C- or better
ENGL 250 Elect Comp 3 cr C or better
4
GEN ED Elective Arts & Hum 3 cr
MATH 207 (or) 265 (or) 304 (or) 314 (or) 317 3 cr
CPR E 288 Emb. Sys I 4 cr (or) COM S 327 Adv Prog 3 cr
SE 319 Soft Con 3 cr
5
GEN ED Elective Arts & Hum 3 cr
CPR E 381 Computer Org 4 cr (or) COM S 321 Computer Arch 3 cr
Choose only one course
CPR E 310 Fnd Cpr Engr 3 cr (or) COM S 230 Discrete Comp 3 cr
Choose only one course
SE 309 Soft Dev 3 cr
COM S 363 DBMS 3 cr
6
Open Elective 3 cr
CPR E 308 Oper Sys 4 cr (or) COM S 352 Oper Sys 3 cr
Choose only one course
COM S 311 Algor Anlys 3 cr
SE 329 Soft Proj Mgmt. 3 cr
SE 339 Soft Arch 3 cr
Junior Class
ENGL 314 (or) 309 Tech Comm 3 cr C or better
7
GEN ED Elective Social Sciences 3 cr
SE Elective 3 cr
STAT 330 Prob & Stat 3 cr
SE 421 Security 3 cr
SE 491 Sr Design 1 3 cr
8
GEN ED Elective Arts & Hum (or) Social Sciences 3 cr
SE Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SE 492 Sr Design 2 2 cr

1. This flowchart is only a guide. For more detailed information, including prerequisites and co-requisites, see the ISU catalog.

2. Check the electives list on www.se.iastate.edu.

3. Course and term offerings subject to change.

4. Lab and lecture must be passed in the same semester to receive credit for CprE courses.

5. Students who take the 4 credit hour CPR E courses (CPR E 288, 308, and 381) can use the extra 1 credit per course toward the supplemental elective requirement.

6. College of Engineering requires two years of a single world language in high school or two semesters of college in a single world language.

7. Iowa State undergraduate students must fulfill graduation requirements in two areas: U.S. Diversity & International Perspectives. Both require three credits of course work.

Fig. 4: A generic flowchart, used for calculating the weight of conflict

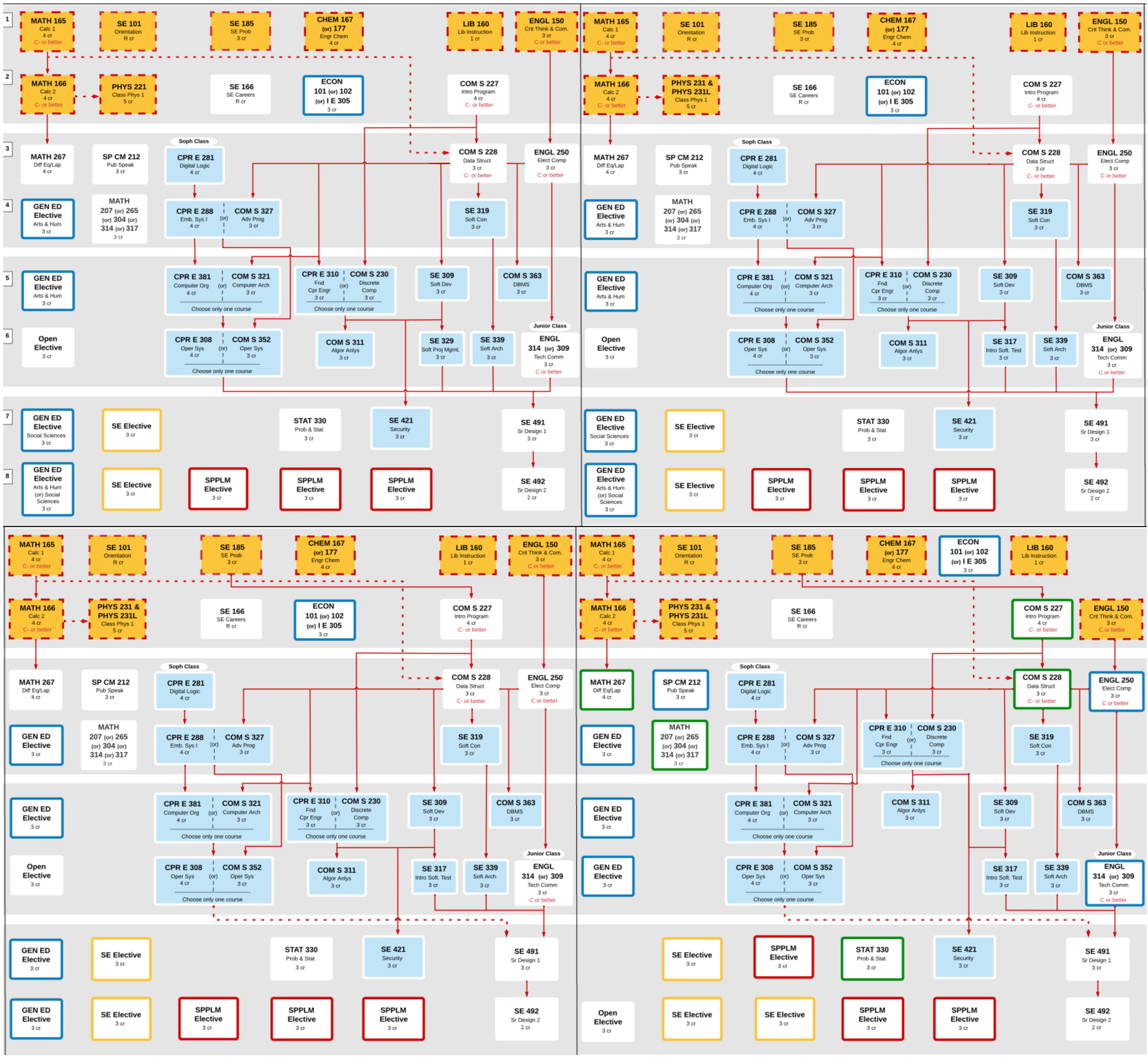

1
2
3
4
5
6
7
8
MATH 165 Calc 1 4 cr C- or better
SE 101 Orientation R cr
SE 185 SE Prob 3 cr
CHEM 167 (or) 177 Engr Chem 4 cr
LIB 160 Lib Instruction 1 cr
ENGL 150 Crit Think & Com. 3 cr C or better
MATH 166 Calc 2 4 cr C- or better
PHYS 221 Class Phys 1 5 cr
SE 166 SE Careers R cr
ECON 101 (or) 102 (or) I E 305 3 cr
COM S 227 Intro Program 4 cr C- or better
MATH 267 Diff Eq/Lap 4 cr
SP CM 212 Pub Speak 3 cr
Soph Class
CPR E 281 Digital Logic 4 cr
COM S 228 Data Struct 3 cr C- or better
ENGL 250 Elect Comp 3 cr C or better
GEN ED Elective Arts & Hum 3 cr
MATH 207 (or) 265 (or) 304 (or) 314 (or) 317 3 cr
CPR E 288 Emb. Sys I 4 cr (or) COM S 327 Adv Prog 3 cr
SE 319 Soft Con 3 cr
GEN ED Elective Arts & Hum 3 cr
CPR E 381 Computer Org 4 cr (or) COM S 321 Computer Arch 3 cr Choose only one course
CPR E 310 Fnd Cpr Engr 3 cr (or) COM S 230 Discrete Comp 3 cr Choose only one course
SE 309 Soft Dev 3 cr
COM S 363 DBMS 3 cr
Open Elective 3 cr
CPR E 308 Oper Sys 4 cr (or) COM S 352 Oper Sys 3 cr Choose only one course
COM S 311 Algor Anlys 3 cr
SE 329 Soft Proj Mgmt. 3 cr
SE 339 Soft Arch 3 cr
Junior Class
ENGL 314 (or) 309 Tech Comm 3 cr C or better
GEN ED Elective Social Sciences 3 cr
SE Elective 3 cr
STAT 330 Prob & Stat 3 cr
SE 421 Security 3 cr
SE 491 Sr Design 1 3 cr
GEN ED Elective Arts & Hum (or) Social Sciences 3 cr
SE Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SE 492 Sr Design 2 2 cr
MATH 165 Calc 1 4 cr C- or better
SE 101 Orientation R cr
SE 185 SE Prob 3 cr
CHEM 167 (or) 177 Engr Chem 4 cr
LIB 160 Lib Instruction 1 cr
ENGL 150 Crit Think & Com. 3 cr C or better
MATH 166 Calc 2 4 cr C- or better
PHYS 231 & PHYS 231L Class Phys 1 5 cr
SE 166 SE Careers R cr
ECON 101 (or) 102 (or) I E 305 3 cr
COM S 227 Intro Program 4 cr C- or better
MATH 267 Diff Eq/Lap 4 cr
SP CM 212 Pub Speak 3 cr
Soph Class
CPR E 281 Digital Logic 4 cr
COM S 228 Data Struct 3 cr C- or better
ENGL 250 Elect Comp 3 cr C or better
GEN ED Elective Arts & Hum 3 cr
MATH 207 (or) 265 (or) 304 (or) 314 (or) 317 3 cr
CPR E 288 Emb. Sys I 4 cr (or) COM S 327 Adv Prog 3 cr
SE 319 Soft Con 3 cr
GEN ED Elective Arts & Hum 3 cr
CPR E 381 Computer Org 4 cr (or) COM S 321 Computer Arch 3 cr Choose only one course
CPR E 310 Fnd Cpr Engr 3 cr (or) COM S 230 Discrete Comp 3 cr Choose only one course
SE 309 Soft Dev 3 cr
COM S 363 DBMS 3 cr
Open Elective 3 cr
CPR E 308 Oper Sys 4 cr (or) COM S 352 Oper Sys 3 cr Choose only one course
COM S 311 Algor Anlys 3 cr
SE 317 Intro Soft. Test 3 cr
SE 339 Soft Arch 3 cr
Junior Class
ENGL 314 (or) 309 Tech Comm 3 cr C or better
GEN ED Elective Social Sciences 3 cr
SE Elective 3 cr
STAT 330 Prob & Stat 3 cr
SE 421 Security 3 cr
SE 491 Sr Design 1 3 cr
GEN ED Elective Arts & Hum (or) Social Sciences 3 cr
SE Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SE 492 Sr Design 2 2 cr
MATH 165 Calc 1 4 cr C- or better
SE 101 Orientation R cr
SE 185 SE Prob 3 cr
CHEM 167 (or) 177 Engr Chem 4 cr
LIB 160 Lib Instruction 1 cr
ENGL 150 Crit Think & Com. 3 cr C or better
MATH 166 Calc 2 4 cr C- or better
PHYS 231 & PHYS 231L Class Phys 1 5 cr
SE 166 SE Careers R cr
ECON 101 (or) 102 (or) I E 305 3 cr
COM S 227 Intro Program 4 cr C- or better
MATH 267 Diff Eq/Lap 4 cr
SP CM 212 Pub Speak 3 cr
Soph Class
CPR E 281 Digital Logic 4 cr
COM S 228 Data Struct 3 cr C- or better
ENGL 250 Elect Comp 3 cr C or better
GEN ED Elective 3 cr
MATH 207 (or) 265 (or) 304 (or) 314 (or) 317 3 cr
CPR E 288 Emb. Sys I 4 cr (or) COM S 327 Adv Prog 3 cr
SE 319 Soft Con 3 cr
GEN ED Elective 3 cr
CPR E 381 Computer Org 4 cr (or) COM S 321 Computer Arch 3 cr Choose only one course
CPR E 310 Fnd Cpr Engr 3 cr (or) COM S 230 Discrete Comp 3 cr Choose only one course
SE 309 Soft Dev 3 cr
COM S 363 DBMS 3 cr
Open Elective 3 cr
CPR E 308 Oper Sys 4 cr (or) COM S 352 Oper Sys 3 cr Choose only one course
COM S 311 Algor Anlys 3 cr
SE 317 Intro Soft. Test 3 cr
SE 339 Soft Arch 3 cr
Junior Class
ENGL 314 (or) 309 Tech Comm 3 cr C or better
GEN ED Elective 3 cr
SE Elective 3 cr
STAT 330 Prob & Stat 3 cr
SE 421 Security 3 cr
SE 491 Sr Design 1 3 cr
GEN ED Elective 3 cr
SE Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SE 492 Sr Design 2 2 cr
MATH 165 Calc 1 4 cr C- or better
SE 101 Orientation R cr
SE 185 SE Prob 3 cr
CHEM 167 (or) 177 Engr Chem 4 cr
ECON 101 (or) 102 (or) I E 305 3 cr
LIB 160 Lib Instruction 1 cr
MATH 166 Calc 2 4 cr C- or better
PHYS 231 & PHYS 231L Class Phys 1 5 cr
SE 166 SE Careers R cr
COM S 227 Intro Program 4 cr C- or better
ENGL 150 Crit Think & Com. 3 cr C or better
MATH 267 Diff Eq/Lap 4 cr
SP CM 212 Pub Speak 3 cr
Soph Class
CPR E 281 Digital Logic 4 cr
COM S 228 Data Struct 3 cr C- or better
ENGL 250 Elect Comp 3 cr C or better
GEN ED Elective 3 cr
MATH 207 (or) 265 (or) 304 (or) 314 (or) 317 3 cr
CPR E 288 Emb. Sys I 4 cr (or) COM S 327 Adv Prog 3 cr
CPR E 310 Fnd Cpr Engr 3 cr (or) COM S 230 Discrete Comp 3 cr Choose only one course
SE 319 Soft Con 3 cr
GEN ED Elective 3 cr
CPR E 381 Computer Org 4 cr (or) COM S 321 Computer Arch 3 cr Choose only one course
COM S 311 Algor Anlys 3 cr
SE 309 Soft Dev 3 cr
COM S 363 DBMS 3 cr
GEN ED Elective 3 cr
CPR E 308 Oper Sys 4 cr (or) COM S 352 Oper Sys 3 cr Choose only one course
SE 317 Intro Soft. Test 3 cr
SE 339 Soft Arch 3 cr
Junior Class
ENGL 314 (or) 309 Tech Comm 3 cr C or better
SE Elective 3 cr
SPPLM Elective 3 cr
STAT 330 Prob & Stat 3 cr
SE 421 Security 3 cr
SE 491 Sr Design 1 3 cr
Open Elective 3 cr
SE Elective 3 cr
SE Elective 3 cr
SPPLM Elective 3 cr
SPPLM Elective 3 cr
SE 492 Sr Design 2 2 cr

Fig. 5: Flowcharts for determining weight of conflict, example of the SE flowcharts

8.4.1 Team Contract

Team Members:

1) Ryan Tullis
2) Nicolas Figueroa Calderas
3) Dillon Gesy
4) Charles “Joe” Hosier
5) Abdullahi Abdullahi

Team Procedures

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

We will meet in the hour before the 491 class every week as well as during TODO for a total of 2 face-to-face meetings a week.

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

For all important communication we will be using email for major updates. For any smaller needs we will communicate via text.

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

For any major decisions that have conflict, we will have a majority vote to make decisions.

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

Dillon will be taking notes during our meetings and will send out an email recapping what was talked about for record keeping.

Participation Expectations

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

Expected for all members to be at every meeting, if something comes up and has to miss a meeting, that’s fine, but the individual who missed should communicate with the team to know what was missed from the meeting.

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

Expect all members to fulfill team assignments and deadlines on time. If there is an issue with an individual potentially missing a deadline, communicate with team members as soon as possible

.

3. Expected level of communication with other team members:

Strive for good communication when necessary, as long as individual work is being completed on time, communication doesn’t need to be incredibly detailed.

4. <u>Expected level of commitment to team decisions and tasks:</u>

All team members should be committed to completing deadlines and targets. Working and communicating to accomplish team goals.

Leadership

1. <u>Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):</u>

Ryan Tullis – Client interaction, team organization
Dillon Gesy – Team organization, individual component design, testing
Joe – Individual component design, testing
Nicolas – Individual component design, testing, high-level manager
Abdul – Individual component design, testing

2. <u>Strategies for supporting and guiding the work of all team members:</u>

Keep everyone responsible and up to date with each other. Making sure all code is reviewed, knowing the requirements, and keeping an open line of communication.

3. <u>Strategies for recognizing the contributions of all team members:</u>

Give everyone some form of valuable work – thank each other for getting work done before the meetings and keep track of where everyone is at.

Collaboration and Inclusion

1. <u>Describe the skills, expertise, and unique perspectives each team member brings to the team.</u>

Ryan & Nicolas Internship experience with management experience
Dillon & Charles Other group projects at Iowa State

2. <u>Strategies for encouraging and support contributions and ideas from all team members:</u>

Talking and good communications, make sure people's ideas are heard

3. <u>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?)</u>

Meet with TA if issue persists past initial conversations from peer-to-peer confrontation.

Goal-Setting, Planning, and Execution

1. <u>Team goals for this semester:</u>

Finish the design part of the project. Get to know everyone in the group. Get a good idea of what we are going to do next semester.

2. Strategies for planning and assigning individual and team work:
At group meetings we will figure out what needs to be done and assign it appropriately.

3. Strategies for keeping on task:
Have frequent group meetings to understand what needs to be done. Using frequent communication is also our goal so we know what is going on at all times.

Consequences for Not Adhering to Team Contract

1. How will you handle infractions of any of the obligations of this team contract?
First try to handle it internally with our group, to find a middle ground.

2. What will your team do if the infractions continue?
We will then bring it up to the TA in order to move forward with the project.

****
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) Ryan Tullis	DATE 9/10/2023
2) Nicolas Figueroa Calderas	DATE 9/10/2023
3) Dillon Gesy	DATE 9/10/2023
4) Charles Hosier	DATE 9/10/2023
5) Abdullahi Abdullahi	DATE 9/10/2023