# Senior Design Server/Client Development for Project Matching

Design Document

Team 44

Clients: Jacob Grundmeier, Akhilesh Tyagi

Advisor: Akhilesh Tyagi

Brady ~ Individual component design
Seth ~ Individual component design, Project Management
Kyle ~ Testing, Code reviewer, Git Documents manager
Matthew ~ Database manager
Yunhao Yang ~ Individual component design, Testing
Gavin ~ UI design manager, Code reviewer
Jake ~ CICD manager
Tyler ~ Engineering Standards manager

sdmay22-44@iastate.edu

http://sdmay22-44.sd.ece.iastate.edu/

# Executive Summary

## Development Standards & Practices Used

Agile Practice

HyperText Transfer Protocol (HTTP)

ANSI/ANS 10.3-1995 Standard for Documentation of Computer Software);(ISO/IEC/IEEE 26514)

IEEE Code of Ethics

## Summary of Requirements

Historically, the Senior Design Project Lifecycle has been a manual process requiring significant time, communication and organization from faculty. To significantly increase the efficiency of this process, a defined matching algorithm and structured web-server is needed for client-project proposition, student-project matching, team resource creation and project review scheduling; allowing for optimized team creation and communication.

## Applicable Courses from Iowa State University Curriculum

COM S 311: Introduction to the Design and Analysis of Algorithms

COM S 309: Software Development Practices

COM S 319: Construction of User Interfaces

COM S 363: Introduction to Database Management Systems

## New Skills/Knowledge acquired that was not taught in courses

Laravel, Angular

# Table of Contents

6. Implementation
7. Professionalism
    7.1 Areas of Responsibility
    7.2 Project Specific Professional Responsibility Areas
    7.3 Most Applicable Professional Responsibility Area
8. Closing Material
    8.1 Discussion
    8.2 Conclusion
    8.3 References
    8.4 Appendices
        8.4.1 Team Contract

# List of figures/tables/symbols/definitions

# 1 Team

## 1.1 Team Members

Kyle Kent

Matthew Karr

Tyler Staker

Brady Synstellien

Gavin George

Jake Gudenkauf

Seth Gardner

Yunhao Yang

## 1.2 Required Skill Sets for Your Project

Frontend knowledge, backend knowledge, database building, database connection, agile methodology, client interaction, CICD

## 1.3 Skill Sets Covered by the Team

Frontend knowledge, backend knowledge, database building, database connection, agile methodology, client interaction, CICD

## 1.4 Project Management Style Adopted by the Team

Agile methodology

## 1.5 Initial Project Management Roles

Brady ~ individual component design
Seth ~ individual component design, Project Management (Boards, meetings)
Kyle ~ testing, code reviewer, Git Documents manager
Matthew ~ Database manager
Yunhao Yang ~ individual component design, testing
Gavin ~ UI design manager, Code Reviewer
Jake ~ CICD manager
Tyler ~ Engineering Standards manager

# 2 Introduction

## 2.1 Problem Statement

Historically, the Senior Design Project Lifecycle has been a manual process requiring significant time, communication and organization from faculty. To significantly increase the efficiency of this process, a defined matching algorithm and structured web-server is needed for client-project proposition, student-project matching, team resource creation and project review scheduling; allowing for optimized team creation and communication.

## 2.2 Requirements & Constraints

- Main Functional Requirements
    - 491
        - We need to use a thoroughly researched algorithm to match students to client projects
        - Faculty panel
            - We need to provide a scheduling system for Faculty reviewers and students to make the process more convenient for both
                - After each team is signed up for a panel, a form will be sent to faculty, and each faculty need to sign for a panel
        - We need to allow faculty to easily assign grades to groups using canvas integration
    - 492
        - Industry Review Panel
            - We need to provide a scheduling system for Industry reviewers and students to make the process more convenient for both
        - We need to allow industry reviewers to easily assign grades to groups using canvas integration
- UI Requirements
    - We will need perspectives for all of the following users
        - Students
            - Project Preference Form
        - Teams
            - Manage team files
            - Give access to design docs
            - Schedule review timeslots
        - Faculty Instructors
            - Approve projects
            - Add students to database

                - Send notifications to students
            - System Administrators
                - Manage non-student DB
                - Creation of team resources
            - Faculty Advisors
                - Assign grades to teams
            - Project Clients
                - Propose Projects
            - Faculty Panel
                - Schedule review timeslots
                - Input grades
            - Industry Review Panel
                - Access team design docs
                - Schedule review timeslots
                - Input grades
- Algorithm Requirements
    - Considerations
        - Market research (students)
            - Student polls
            - Possibly client polls
        - Client + faculty advisor
            - Tech skills sets needed by the client are represented
        - Abet Requirements
            - Team diversity
        - Objective function
        - Student Satisfaction function
        - Project Satisfaction function
        - Auction model
    - Constraints
        - Use of an SQL database:
            - MariaDB or MySQL
        - Use of a well documented web-framework
            - Laravel
        - Limited web server memory/resources.
            - Linux Virtual Machine
        - Security for user authentication:
            - Canvas sign in integration
            - OKTA or LDAP integration
    - Quantitative constraints
        - Limited user types.

        - # of students
        - # of Faculty Instructors
        - # Panel Reviewers
        - …
    - Limited development time
        - One semester of Planning
        - One semester of Dev
    - Limited ability for real time features
        - Time between requests and responses

## 2.3 Engineering Standards

- HyperText Transfer Protocol (HTTP)
    - Scheduling and Preference Requests
- File Transfer Protocol (FTP)
    - Upload initial student information/ design doc files
- (ANSI/ANS 10.3-1995 Standard for Documentation of Computer Software);(ISO/IEC/IEEE 26514)
    - We want to make sure our software is understandable and maintainable as we will be passing it on to a system administrator after development has finished

## 2.4 Intended Users and Uses

- The intended users of our project will be anyone involved in the process of assigning groups for Senior Design as well as anyone that is a stakeholder in a senior design group's project.
- This project benefits stakeholders as it speeds up the initial steps of team creations and early communication. It centralizes all data for the course and allows for better organization through the entire senior design lifecycle.
- Use cases:
    - Assigning groups
    - Assisting in group-client interaction
    - Making it easier for groups and stakeholders to communicate
    - Scheduling of faculty and industry panel review

# 3 Project Plan

## 3.1 Project Management/Tracking Procedures

We chose agile for this project. One reason for this choice is we felt agile development best described our plan to create a rough design and implementation as early as possible while creating incremental improvements over time. Most of us also have experience working professionally in agile based team projects. Through the early development process we expect to further define requirements and other functionality features that can be included.

We currently have a Trello board that is actively monitored by each of us. We expect to use Github to facilitate code development on the front and back end. For communication we use Discord when organizing meetings and work times.

## 3.2 Task Decomposition

1. Frontend Design
    a. Design Initial Project Structure
    b. Create Initial Project Skeleton
    c. Design Permissions Structure and Components
    d. Define Role and Role Specific Components
    e. Define  Role and Role Specific Services
    f. Define  Role and Role Specific Resources
    g. Define Shared Components
    h. Define Shared Services
    i. Define Shared Resources
    j. Create UI Mock-Ups
    k. Implement Basic Role Views Utilizing Permissions
    l. Implement Role Features and Components
    m. Implement UI Mock-Ups
    n. Test  Basic Role Views Utilizing Permissions
    o. Test  Role Features and Components
    p. Test UI

2. Database design
    a. Database Research
    b. Senior Design Database

c. Student Table
d. Group Table
e. ABET table
f. Faculty Table
g. Client Table
h. Project Table
i. Connect the tables
j. Query Database
k. Test Database

3. Algorithm
    a. Market Research
        i. Find out what part of the preference means the most to the student
    b. Determine the parameter
        i. What parameters should be input to the algorithm
    c. Design
    d. Implement
    e. Test
4. Faculty/Client Interaction
    a. Faculty Creates Class
        i. Uploads student list
        ii. Sends out survey link to clients
    b. Clients Will Propose Projects
        i. Fill out project survey on site
        ii. Waits for approval
    c. Faculty Approves projects
        i. Can view proposed projects and approve/decline
            1. Download to Excel
            2. View in site
        ii. Once all reviewed, sends status update to clients
    d. Clients Receive Team List
        i. Once students have stated preferences, clients will get a list of team members
    e. Faculty Review Panel
        i. Log into site and give time preferences
        ii. Sign up for projects to review
        iii. Give comments on site
5. Industry Review Panel
    a. Sign Up For Time Slots

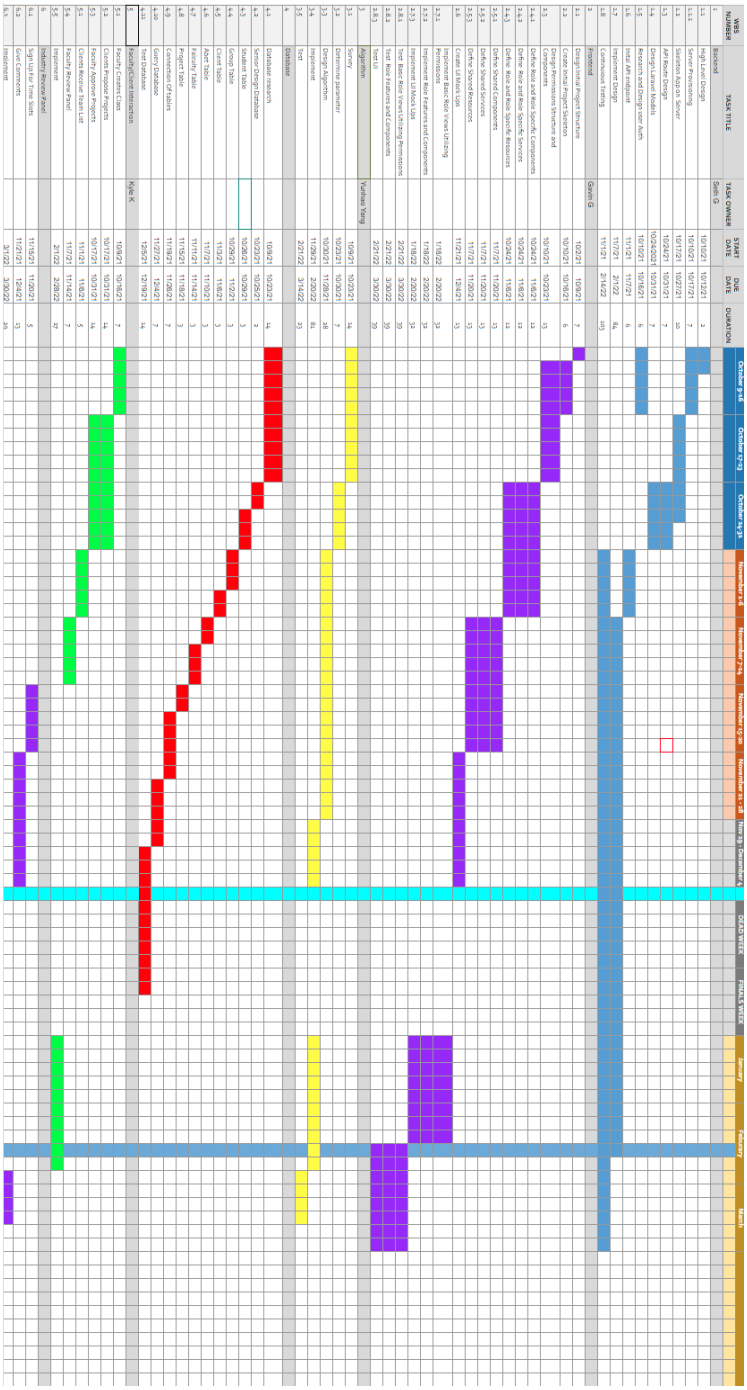
        i. Industry patterns should log into site and give time preferences
        ii. Pick what projects they would like to review
    b. Give Comments
        i. Integration for panel to enter in comments
        ii. Professor can review comments to determine students grades
6. Backend Design
    a. Determine technologies used
    b. High Level Framework Design
    c. Provisioning of server
    d. Skeleton App deployment
    e. Design  of API endpoints
    f. Design of framework components
        i. Model Design
        ii. Controller Design
        iii. Auth Design
    g. Implement Designs using agile methodology
    h. Continuously test implementation

## 3.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

1. Design Documents Completed
    a. Finalized after multiple iterations and ready to be implemented in actual code
2. The backend will be in a launchable state, with 1 or more functional routes
    a. An accepted file structure and design layout will be chosen
    b. Research into laravel debugging and launching procedures will be done
    c. A generalized concept and outline of an algorithm can be run with static data inputs
3. The Frontend will be in a launchable state,
    a. An accepted file structure and design layout will be chosen
    b. Generalized services and components will be implemented
    c. Authentication procedure outlined
4. The database
    a. MariaDB and MySQL will be researched to determine which would be better for the project.
    b. All tables for the Database are created.
    c. Queries to the database are handled in less than a second.
5. Faculty/Clients/Industry

a. Design a flowchart for the two semesters and how each group will interact with it
b. Get feedback from each group on any part that could be improved

## 3.4 Project Timeline/Schedule



## 3.5 Risks and Risk Management/Mitigation

| Risk | Mitigation |
|---|---|
| Over schedule | Hold an in-team meeting to work out a plan to try and distribute tasks better so there is a better chance of finishing. If there's no possible plan to make this happen, then communicate with the client to find out what features to prioritize. |
| Some code changes breaking the project or introducing hard to find bugs | To mitigate this risk, we will be using git with every team member using their own branch. Merges with master are required to be reviewed by at least one other team member before it is approved. |
| Market Research not as successful as desired<br><br>-Not enough responses from surveys | To mitigate this risk, we will reach out to the professor when we send out the surveys to ask him to encourage students/clients to fill out the survey. |

## 3.6 Personnel Effort Requirements

| **Tasks** | **Reference/Explanation** | **Person-Hours** |
|---|---|---|
| Frontend Design | This task involves the initial design of the project structure and skeleton. A major part of this task will be defining and organizing necessary services, components, and role specific views/features. | Design Initial Project Structure - 2h<br>Create Initial Project Skeleton - 2h<br>Design Permissions Structure and Components ~ 10h<br>Define Role and Role Specific Components ~ 5h<br>Define Role and Role Specific Services ~ 5h<br>Define Role and Role Specific Resources ~ 5h<br>Define Shared Components ~ 5h<br>Define Shared Services ~ 5h<br>Define Shared Resources ~ 5h<br>Create UI Mock-Ups ~ 10h |

| | | |
|---|---|---|
| | | Implement Basic Role Views Utilizing Permissions ~ 20h<br>Implement Role Features and Components ~ 20h<br>Implement UI Mock-Ups ~ 20h<br>Test  Basic Role Views Utilizing Permissions ~ 15h<br>Test  Role Features and Components ~ 15h<br>Test UI ~ 5h<br>Total: 135h |
| Database design | This task is about the design and implementation of the database. | Database Research - 12h<br><br>Senior Design Database -2h<br><br>Student Table -4h<br><br>Group Table -4h<br><br>ABET table -4h<br><br>Faculty Table - 4h<br><br>Client Table -4h<br><br>Project Table -4h<br><br>Connect the tables -8h<br><br>Query Database -20h<br><br>Test Database -20h<br><br>**Total:** 86h |
| Algorithm | This task is about designing an algorithm which can properly assign the project to the student. The task involves the market research for the algorithm. And design, implementation and test the algorithm base on the market research | Market Research - 10h<br><br>Determine Parameter - 2h<br><br>Design algorithm - 10h<br><br>Implement algorithm - 40h<br><br>Test algorithm - 5h<br><br>**Total**: 67h |

| | | |
|---|---|---|
| Faculty / Client Interaction | This task is directed on how the faculty and clients will interact with the site. This task includes research broken down into higher level tasks. Finally, a task on how long it will take us to implement everything. | Faculty Creates Class - 4h<br><br>Clients Will Propose Projects - 8h<br><br>Approves projects - 8h<br><br>Clients Receive Team List - 2h<br><br>Faculty Review Panel - 4h<br><br>Implement - 60h<br><br>**Total**: 86h |
| Industry Panel | This task is focused on the industry review panel that occurs at the end of 492. In this task, we have some research to do. Last, an estimate on how long it will take us to implement this feature. | Sign Up For Time Slots - 4h<br><br>Give Comments - 8h<br><br>Implement - 20h<br><br>**Total**: 32h |
| Backend | This task is focused on the design and implementation of the backend web framework. This task will be focused on the creation of API endpoints and providing interaction between the database and the frontend. | High Level Design - 2h<br>Server Provisioning - 2h<br>Skeleton App Deployed on Server- 10h<br>API Route Design- 5h<br>Model Design- 8h<br>User Auth Research- 4h<br>Auth Design- 8h<br>Controller Design- 5h<br>Initial API endpoint- 2h<br>Implement Design- 40h<br>Continuously Test Implementation- 30h<br>**Total**: 116 h |

## 3.7 Other Resource Requirements

1. Time for implementation, debugging, and further development
2. Hardware resources such as memory and processing power for hosting the back and front end
3. Online documentation and material for information on algorithm, front, and back end development
4. Course material from lecture and canvas
5. Team members to assist in design and development
6. Market research, survey students, clients, and industry

# 4 Design

## 4.1 Design Context

### 4.1.1 Broader Context

| 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)<br>Faculty ~<br>Faculty will be affected in terms of time saving. Monitoring the status of proposed projects will be faster than current methods.<br>Clients ~<br>Clients will have an easier time formulateing and proposing projects. They will have a better experience utilizing a more modern project proposal platform<br>Students ~<br>Students will have a better learning experience with senior design. The algorithm side of the project will formulate teams that will be better suited for success.. | The development of a user facing application environment will be a major factor in time saving and user experience. This environment will be developed on a modern framework that can be easily built upon to improve user experience. The algorithm part of the project will improve the students' senior designs experience as team makeups will promote diversity and success through the design and implementation process. |
| 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.<br>The project affects the social values of every person involved in the senior design process, specifically ISU faculty. Being a faculty member means the value of time. This project will take a load off of ISU faculty members, allowing for them to connect more | The implementation of the solution will allow the course faculty to automatically assign teams, relieving a large load of the course's upfront work. This will give faculty more time to connect with students and provide a richer lecture. |

| | | |
|---|---|---|
| | with students, reinforcing the academic culture at ISU | |
| 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.<br><br>Due to the nature of the project being software, there is little to no environmental impact besides the energy required to run our server. | The solution will be implemented on a VM, provisioned on ISU servers. Since this process will be running more than an idle server, more energy will be required from the university, potentially having environmental impacts. |
| 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.<br><br>Due to the increased efficiency of initial course setup, this will decrease the amount of work hours needed from faculty, impacting the cost from the university. | Course Faculty and IT Admins will now spend less time doing work manually, therefore less hours will be billed to the university. |

### 4.1.2 User Needs

Faculty (Professor, Faculty Advisors, Administrators, etc…):

Professors need a way to approve viable projects, add students to a student database and notify students of important updates because they are responsible for doing these tasks. They are the most important role as they are vital for the flow of projects and students into the semester.

Faculty Advisors need a way to assign grades to groups because they are the final decider of team grades.

IT Administrators need a way to interact with the client and faculty database and retrieve team details because they are the role that will manage the client and faculty database and manually create team resources once teams are created.

Faculty Panel Members need a way to schedule panels, communicate with students and faculty because their role is important to review the design of teams at the end of the semester. They then need to communicate with the team they are going to review and the faculty advisors to help assign grades.

Clients (Project creators/proposers)

Clients will need a way to create and/or propose projects for 491. An example of a client user would be an iowa state professor,  corporate sponsor of iowa state, student,  and administrator. The defining characteristic of a client would be a user  looking to develop a solution to a certain problem.

Students (Students Taking 491)

Students will need input on their project preferences because they want to be assigned into a project.

Outside ISU Organization (ABET/Industry)

ABET Evaluators need access to team documents because they need to be able to evaluate teams on their ABET values when the project is being finalized

Industry Panel Members need a way to access design docs and assign grades because they are the final evaluators of the projects and have a strong say on the project's grade after looking over the docs.

### 4.1.3 Prior Work/Solutions

- Our project will be a unique solution to the problem presented by our client
- We will make the algorithm in our project based on the market research. This is a pro over any generic team matching algorithm
- Our web-server will also be using LDAP and possibly OKTA for authentication, making logins specific to the ISU Organization.

### 4.1.4 Technical Complexity

- Frontend (Component):
  - Using modern software engineering principles, a frontend framework will need to be implemented to meet user needs. This will mean meeting industry standards for UI/UX, permissioning/authentication and code standards
- Backend (Component):
  - Using modern software engineering principles, a backend framework and database will need to be implemented. This will require engineering

principles and meeting industry standards in the scope of database design, framework architecture, API endpoint creation and user authentication.

- Algorithm (Component):
  - Though a matching problem of this sense is a NP hard problem, this relies heavily on finding a mathematical model to match users. Examples are bidding algorithms. This requires meeting industry standards for time complexity, ease of use and manipulation of input heuristics/functions

## 4.2 Design Exploration

### 4.2.1 Design Decisions

Design Decisions:

1. Laravel as backend framework
2. Angular as frontend framework
3. ISU provisioned server running on Ubuntu 20.04
4. Apache 2 as http server
5. MySQL as a database language.

### 4.2.2 Ideation

One design decision was what backend framework to use for the project. After an initial discussion with the IT admin on the project it was determined that an SQL based data structure was the preferred option for the backend side of the project. This was decided due to the availability of servers on campus and the lower cost associated with facility the backend on campus versus an alternative cloud option. Once SQL was determined we started to determine potential backend frameworks. The top backend framework options were Laravel, SpringBoot, and MariaDB. SpringBoot was one of our initially preferred options because most of us had prior experience with the framework, Laravel and MariaDB were preferred options because our IT admin for the project recommended the frameworks which worked well with prior 491 projects. Other alternatives that were thought about included ruby on rails and flask, these frameworks were included as options based on the widely available documentation and use in commercial application platforms. The decision was ultimately to go with Laravel because of an interest in learning further about this framework which was the top recommended database by the IT admin on the project.

### 4.2.3 Decision-Making and Trade-Off

We determined the pros and cons of the backend framework using a weighted matrix table. One of the major criteria was the ease of implementation of a framework. The ease of implementation was determined based on the availability of tutorial-like content on the framework as well as anecdotal reports from other programmers that worked with the framework

on past projects. Another important aspect of the framework was documentation. This was specifically defined as the amount of content as well as the organization and layout of online material that covered the different aspects of the framework. Prior experience was another criteria we compared against which was a scale based on our team members prior knowledge or experience working with the framework. Finally our last criteria was the industry usage of the framework. This was a scale based on reports and statistics of the framework's usage in commercial application platforms within the u.s.

| **Criteria** | **Weighting** | Laravel | SpringBoot | Flask | MySQL | MariaDB |
|---|---|---|---|---|---|---|
| Documentation (Availability and amount of material regarding implementation and information on the framework) | 4 | 5 | 5 | 3 | 4 | 4 |
| Prior Experience (Team members past experience with the framework) | 3 | 2 | 4 | 1 | 5 | 0 |
| Ease of implementation (Framework's ability to interact with other componen | 5 | 5 | 5 | 3 | 3 | 5 |

| | | | | | | |
|---|---|---|---|---|---|---|
| ts) | | | | | | |
| Industry Usage (How common is the Framework in commercial applications) | 4 | 5 | 2 | 4 | 3 | 3 |
| Total | | 71 | 65 | 46 | 58 | 53 |

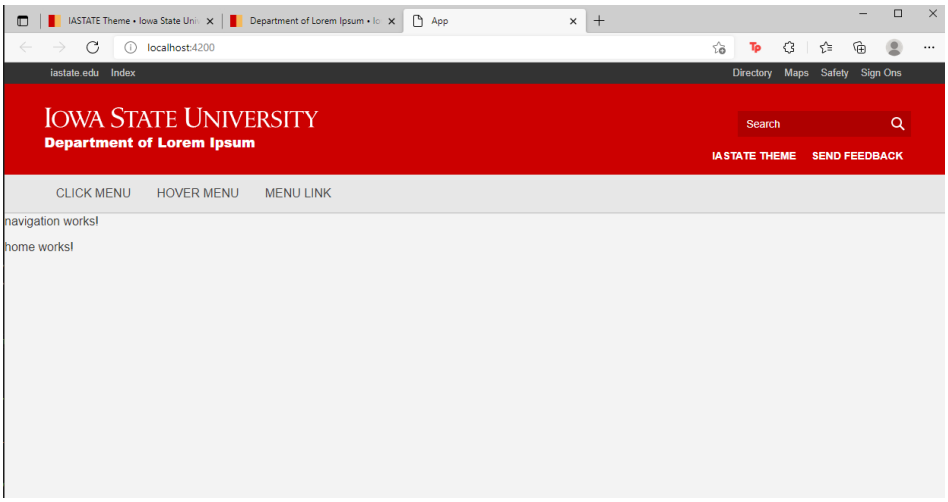
## 4.3 Proposed Design

We have determined the front and back end frameworks, along with the initial designs. For Frontend we have selected angular as the framework and an initial outline of file structure. For the backend we selected Laravel and have secured a server through ETG that can be virtually accessed.

### 4.3.1 Design Visual and Description

Frontend:

- For the frontend UI design, we plan on using the Iowa State theme provided by the university, utilizing the generic nav bar given in the theme to make each view in our design as accessible as possible.



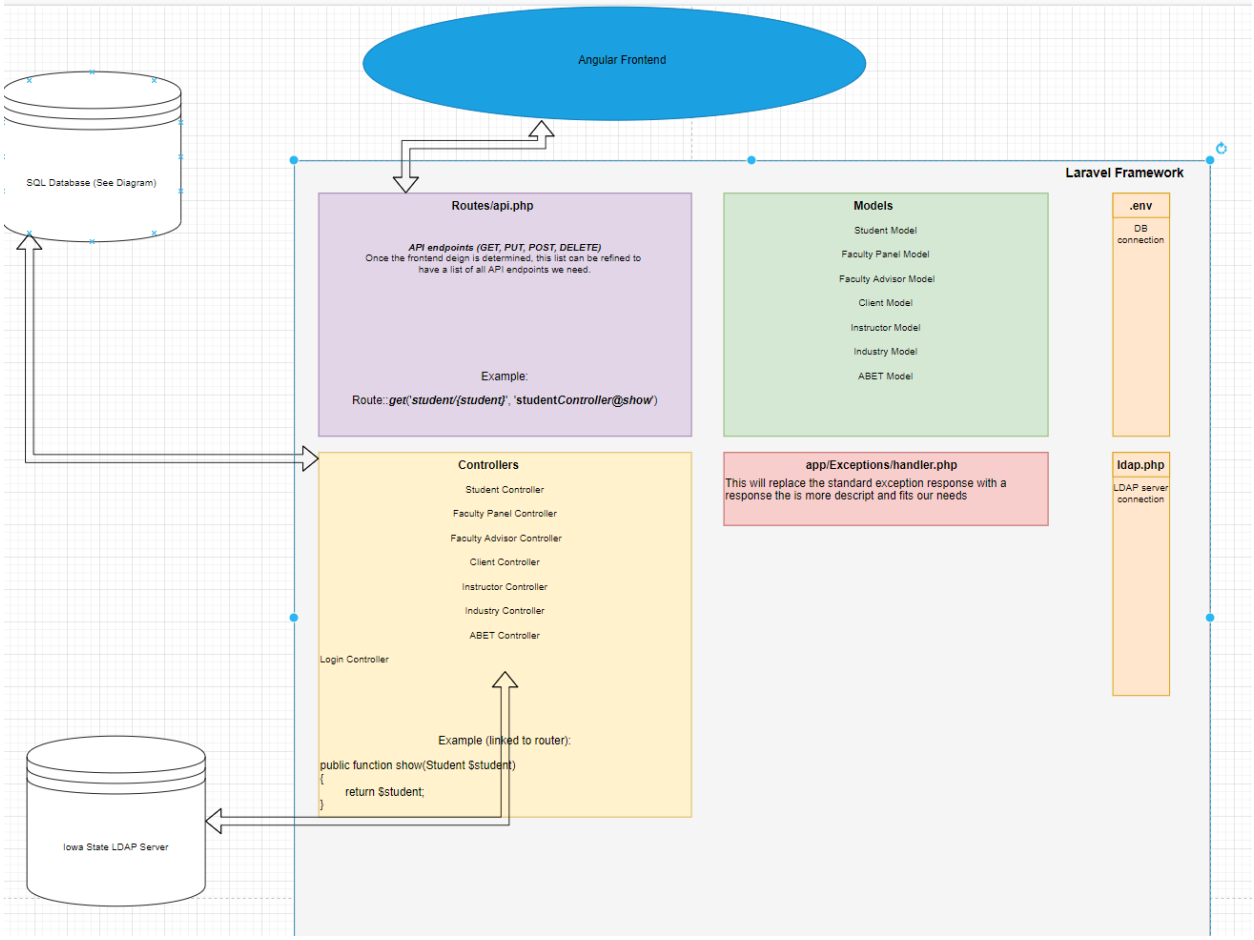
(Figure 4.3.1.1)

- For the frontend architecture, our initial design plan is to utilize angular components and services to create the 8 different views we need. To facilitate communication with the backend, we will have resource services that manage all api calls and then have another set of services that handle the information received from or given to those resources. Our rough initial component diagram shows this concept with different architecture layers for the different responsibilities. Once we plan out more of each view, we will add more components as necessary to this diagram and refine it to show each component's relationship with the others.


(Figure 4.3.1.2)

Backend:

- This implementation is very similar to any other Laravel application. There are going to be models for each user of the application, a controller to access the API endpoints for each user and routes to redirect the calls to the server to its respective controller. There is a connection to the MySQL database and LDAP server for user authentication. We will also implement an exception module to handle exceptions so the frontend has a good idea of the errors. When an endpoint is hit, the router will send the request to the controller where, using the model, data will be queried from the database and send a response back to the frontend.


Angular Frontend
SQL Database (See Diagram)
Laravel Framework
Routes/api.php
API endpoints (GET, PUT, POST, DELETE)
Once the frontend deign is determined, this list can be refined to have a list of all API endpoints we need.
Example:
Route::get('student/{student}', 'studentController@show')
Models
Student Model
Faculty Panel Model
Faculty Advisor Model
Client Model
Instructor Model
Industry Model
ABET Model
.env
DB connection
Controllers
Student Controller
Faculty Panel Controller
Faculty Advisor Controller
Client Controller
Instructor Controller
Industry Controller
ABET Controller
Login Controller
Example (linked to router):
public function show(Student $student)
{
return $student;
}
app/Exceptions/handler.php
This will replace the standard exception response with a response the is more descript and fits our needs
ldap.php
LDAP server connection
Iowa State LDAP Server

(Figure 4.3.1.3)

Database design:

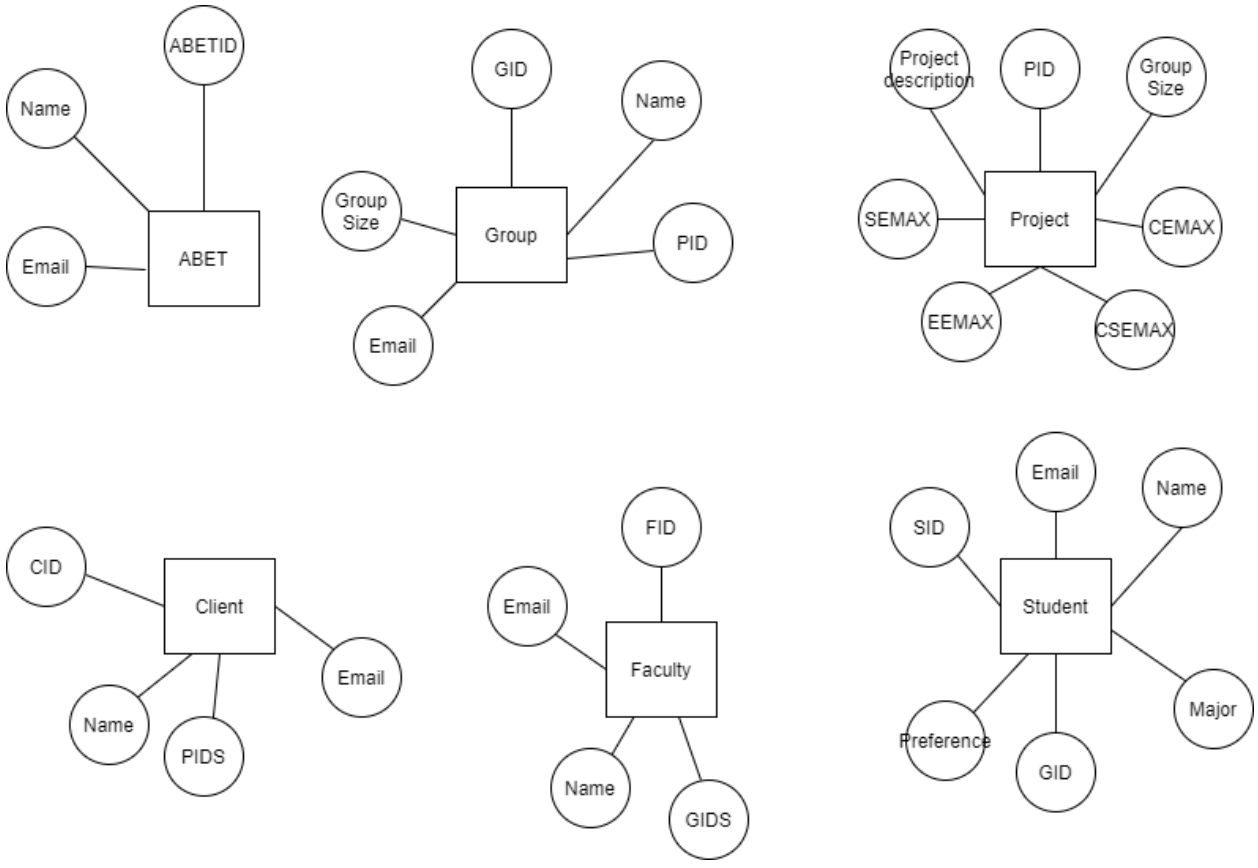
Fields that end with ID are markers for each table to have a unique identifier to call back to a certain line in the table.

Emails and names are included in all of the tables that have to do with people.

The fields ending in max have to do with the max amount of a major can be slotted into the project.

There is a PIDS field for client because clients can propose multiple projects.

There is a GIDS field for faculty because a faculty member could be assigned multiple groups or no groups.




(Figure 4.3.1.4)

### Algorithm

Matching Overview:

For each project, check each student's attribute, and give a score based on matched attributes. Pick the student with the highest score

If found a project is a preferred project of that student, that student can have a bonus score based on the preferred project ranking

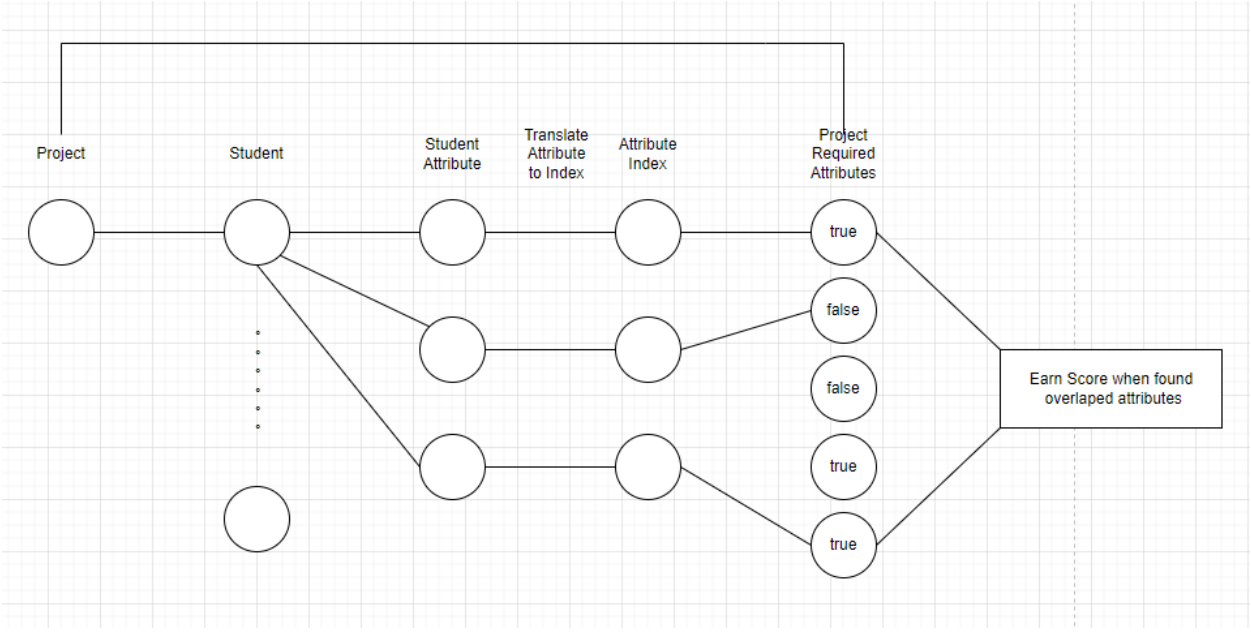
If the student have desired teammate, match the teammate with current project, and then teammate can have a bonus score

Main Matching Data Structure

Project Attribute - A boolean array which contains all of the attributes appeared in the current semester projects. If the value is true, it means the attribute is required by the project

Student Attribute - A string array which contain all of the attributes of the student

Attribute Dictionary - A dictionary that can translate attribute name to attribute index




(Figure 4.3.1.5)

### 4.3.2 Functionality

Our current design clearly shows how we will handle users' roles across our website. How a user will flow between the pages, depending on what roles they have. Additionally, on the backend side of things, we have defined what backend framework we are going to use. Last, we have defined our initial tables that need to be created in our database. Overall, our current design documents provide our team a great starting foundation for our project.

Looking at the non-functional requirements, we have not laid out what constraints we really want to focus on yet. Our current design showcases what the first initial webapp will look like, but does not give any clear guidelines on how responsive or fast we want a user to have. Currently, if you click on the navigation bar you will be redirected to the new page instantaneously. Overall, we could improve our non-functional requirements constraints a bit better.

### 4.3.3 Areas of Concern and Development

One of our initial concerns was handling authentication with users, especially those users who do not have an ISU account/email. As a group, we came up with a solution to use sort of create our own dual authentication service. It may be a bit more complicated, but it is the only solution we have to our problem.

Another problem that we thought of comes into play with our algorithm. In order to satisfy an ABET requirement of diversity, we need to have diverse groups. However, one problem is that we do not want to discriminate against people because of their race, gender, or ethnic background. As a group, we have sought guidance from our client on how to handle this situation. As we further develop our algorithm, we may have to ask our client if the proposed solution works.

## 4.4 Technology Considerations

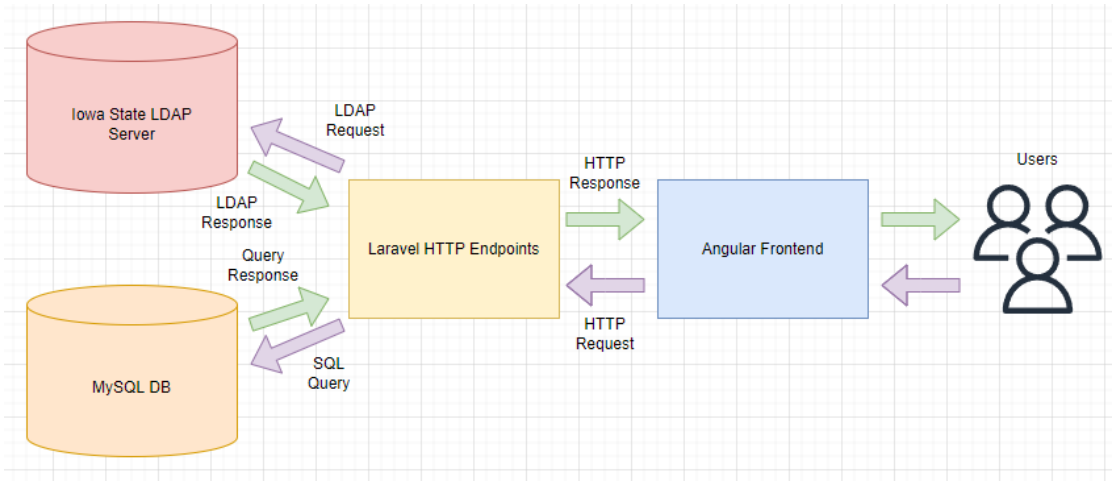
We approach decisions of technology used in the front and backend frameworks using weighted matrix tables. The full scaled matrix table used in the decision process can be seen in section 4.2.3. Our main goal was to determine a technology that would be suitable for every team member. The backend framework was required to be SQL based to follow with iowa state university guidelines established by our technology advisor Jake. The frontend framework was more open to us as the team, a few notable options we considered were React, Angular, and Bootstrap.

## 4.5 Design Analysis

Using all of the above sections to take in consideration, we have established a plan for the actual final project. We compared different frameworks to use on both the frontend and backend. Along with which type of database would be best for our client, tech stack, and knowledge from the team. Additionally, we took into consideration, which would be best in a tech stack and how the stack would run on the Ubuntu VM. Throughout the analysis process we met as a team along with our client and faculty advisor to help us decide which path is the best to go down.

## 4.6 Design Plan

Our design plan has been based off of the high level diagram seen below. We modeled what our system should look like after choosing the technologies our project would use to meet the user's needs. We planned on designing a Laravel application that was able to connect to an ISU authentication server and a database on a VM. We then planned on designing a DB that was able to handle the volume of data we used and designed a MySQL database and the required tables in siad DB to meet data requirements. Next, to interface with the Laravel backend that was serving data, we planned to design an Angular application that was capable of linking the users to the application. Lastly, unconstrained from the design of the web-application, the algorithm was planned to be designed with the capability to read the data from the DB and be able to solve the problem of matching teams; meeting the weighted function requirements.



# 5 Testing

Some unique testing challenges that our design entails are:

- Algorithm analysis with different variations of filled out responses to a form for assigning 491 projects.
- Actions/accessible features for different user types on the front facing application site.
- Unique post data bodies for routes set up on the backend of the application.
- Error handling in the event of invalid data during interaction on the front facing application site or trying to receive response from the backend.

## 5.1 Unit Testing

- Algorithm
    - Enter multiple different groups of data to ensure the algorithm is working as predicted.
    - Input different sets of the data (different student data, and different group requirements) to test the project ranking number generator to ensure the algorithm generates a proper rank number based on the input data.
    - While matching with teammate, make sure the algorithm can find the most proper project based on two students' preferences
- Database
    - Write tests for all of the stored procedures created for the MySQL database. This would be written with the MySQL language inside the database. Schema testing check to make sure all the tables show up in the schema.
- Frontend
    - Each Angular component and service will have its own test file, where we will unit test all methods using Jasmine and Jest.
- Backend
    - Laravel uses the unit testing framework called unitPHP. With this we can test all aspects of the backend functionality. We would test the controller, service, and other various internal backend components.

## 5.2 Interface Testing

- Frontend
    - Within our component test files we can test the Angular forms we create to ensure that data is properly passed into our service and resource calls to the backend.
- Database
    - Stress testing to make sure that when information is added to the database it's not lost nor slows down the database. Testing with Neo4j and postman.

## 5.3 Integration Testing

- Frontend
    - Puppeteer for e2e testing and capturing performance to analyze.
- Backend
    - The testing framework that we choose for the backend is Pest. This will help to ensure that the entirety of the backend produces the correct behavior.

## 5.4 System Testing

Ensure that the integration and unit tests that test the key features of the system requested by the client are successful.

Stress test system to make sure it can handle an expected number of users within our time requirements.

Run the algorithm on an expected set of data and verify that it runs in a timely fashion(we have no constraints) and that it produces a client accepted output.

## 5.5 Regression Testing

By using automated test runners like Jest we can easily ensure that previously written tests still pass and the behavior of methods and components haven't changed.

Some critical features that we need to ensure do not break are the algorithm, the UI input for the algorithm, and extended parameters for the algorithm as these are key features requested by the client.

## 5.6 Acceptance Testing

- Frontend
    - Before we push any live changes to the site, we would want to have a quick meeting with our client or send screenshots of the design. This way if our client wants to change any visual or feature they would have a say before we push the change.
- Backend
    - Group members on the backend sub team, would meet and make sure that the new code changes have the correct functionality. Basically just have everyone on this sub team sign off on the changes.
- Algorithm
    - Explain the entire algorithm to the client step by step and show the code in the meantime.

## 5.7 Security Testing

- Research some common attack patterns and verify that our system can handle and prevent them from working by returning error codes or handling the attack properly.
- Make sure for our frontend components, that if a person is not signed into our site they should not have access to these views.
- Make sure for our backend controller links (requests), that if a person is not signed into our site they should not have access to these requests.

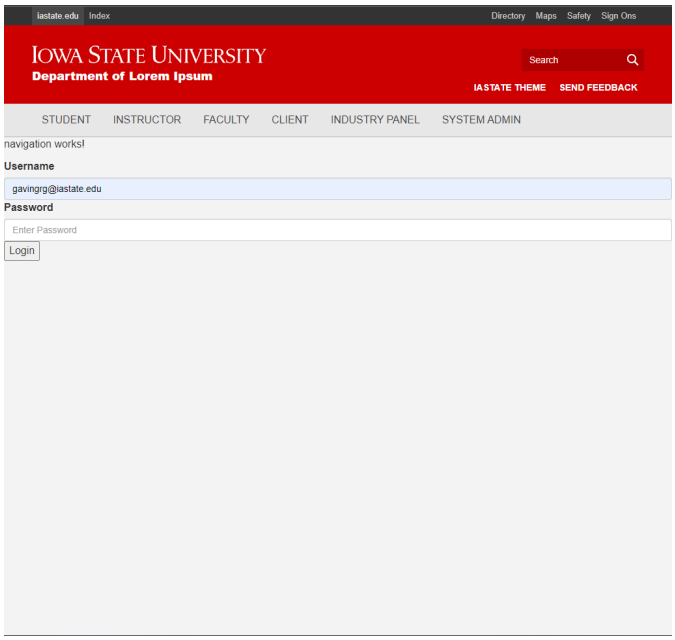
## 5.8 Results

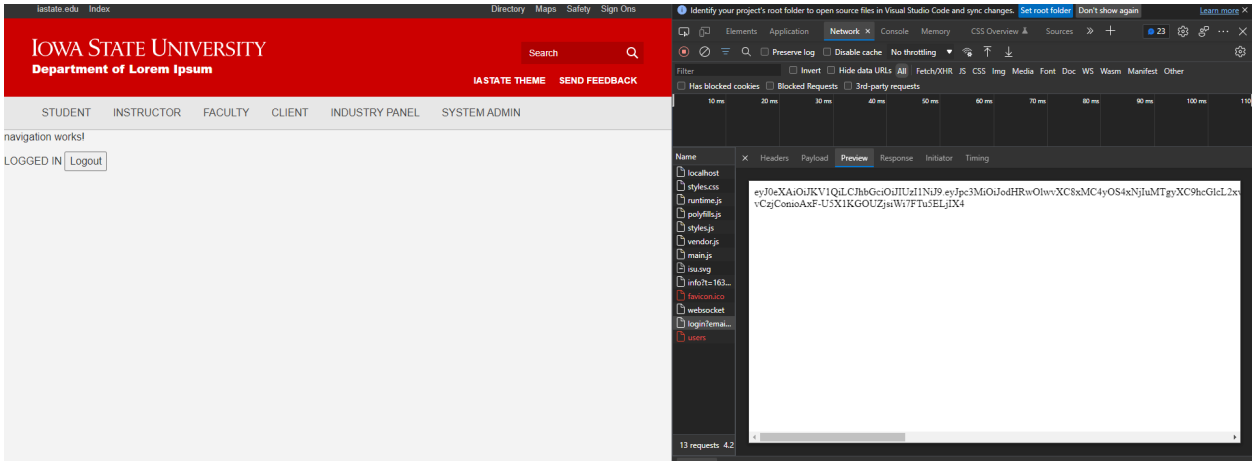
Overall, our project has four main components. Each component has its own testing criteria that it must pass. Looking at the table below, you can see a summary of each component. Each component has their own set of requirements that they must pass to ensure that our project is completely tested.

| Components | Summary narrative |
|---|---|
| Frontend | Users will be able to navigate the site freely with whatever role(s) that they have. Should pass all unit and integration testing. Lastly, should be accepted by the client before any changes go live (big features and UI). |
| Backend | Only internal users have access to requests. Passes all unit and integration tests. |
| Database | All data is returned correctly and precisely. Data return for a query does not take more than a second. Data added to the database is added correctly. |
| Algorithm | Algorithm will be able to generate a most compatible result base on each project's requirements and student's preferences |

# 6 Implementation

https://git.ece.iastate.edu/sd/sdmay22-44



# 7 Professionalism

## 7.1 Areas of Responsibility

| Area of responsibility | Definition | NSPE Canon | IEEE |
|---|---|---|---|
| Work Competence | Perform Work of high quality, integrity, timeliness, and professional competence | Perform services only in areas of their competence; Avoid deceptive acts. | (IEEE #6) Undertake tasks you have the necessary skills to complete or disclose that you lack the skills that could impede progress in the task. IEEE and NSPE are similar in taking tasks you have the skills for and avoiding deceiving others. |
| Financial Responsibility | Deliver products and services of realizable value and at reasonable costs | Act for each employer or client as faithful agents or trustees. | Reject bribery, and avoid conflicts of interest. Make sure to disclose affected parties. (IEEE #2 & #4) |
| Communication Honesty | Report work truthfully, without deception and understandable to stakeholders | Issue public statements only in an objective and trustful manner; Avoid deceptive acts | To be honest and realistic when stating claims based on available data (IEEE #3) |
| Health, Safety, Well-Being | Minimize risks to safety, health, and well-being of stakeholders | Hold paramount the safety, health, and welfare of the public | To avoid injuring others, their property, reputation or employment by false or malicious action. To accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might |

| | | | |
|---|---|---|---|
| | | | endanger<br>the public or the environment.<br>(IEEE #1 & #9) |
| Property Ownership | Respect property, ideas and information of clients and others. | Act for each employer or client as faithful agents or trustees | IEEE does not have a distinct code for this, but Code #9 touches on avoiding injuring others' property, reputation. |
| Sustainability | Protect environment and natural resources locally and globally | N/A | To accept responsibility in making decisions consistent with the safety, health,<br>and welfare of the public, and to disclose promptly factors that might endanger<br>the public or the environment;<br>(IEEE #1)<br>IEEE accepts responsibility for decisions that may endanger/ not protect our natural resources. |
| Social Responsibility | Produce Products and services that benefit society and communities | Conduct themselves honorably, responsibly, ethically, and lawfully so as to enhance the honor, reputation, and usefulness of the profession | To improve the understanding of technology; its appropriate application, and<br>potential consequences.<br>to treat fairly all persons and to not engage in acts of discrimination based on<br>race, religion, gender, disability, age, national origin, sexual orientation, gender<br>identity, or gender expression.<br>(IEEE #5 & #8) |

## 7.2 Project Specific Professional Responsibility Areas

| Area of Responsibility | Does the Area Relate to Our Project? |
|---|---|
| Work Competence | Yes, this code may be one of the most important ones that our project relates to. We want to deliver a high quality project that fits all of our clients needs. Additionally, we want to complete everything related to our project in a timely manner.<br>HIGH - We are utilizing a gantt chart to help reach timeline goals and have adapted an agile like work environment. |
| Financial Responsibility | Yes, this does relate to our project. We may not be getting paid, but we do have a budget to experiment with cloud technologies if we see that it fits. However, we want to be conscious of this and make sure not to get over budget and deliver a great product by the end of the 492.<br>HIGH - We have yet to realize any cost for the project, fulfilling our Financial Responsibility. |
| Communication Honesty | Yes this does relate to our project. It is important for the success of all areas in our project that we report work truthfully and maintain open, objective, and trust worthy dialogue between our team members as well as our client.<br>HIGH - We have been honest and communicative in our team meetings and weekly client meetings. |
| Health, Safety, Well-Being | Yes, this relates to our project as we will need to handle personal information of students and ensure that it is kept private and secure.<br>HIGH - We have not been exposed to sensitive data that could harm stakeholders, but we have roles defined in our design that allows for data to be seen to only those allowed. |
| Property Ownership | Yes this does relate to our project since this requires information from the client to work.<br>N/A- Since we have not gathered any information from users yet. |
| Sustainability | Not applicable since our product has no impact on the environment. |

| | |
|---|---|
| | N/A - This product has no impact on the environment. |
| Social Responsibility | Yes this would relate to our project as the objective is to improve the academic and work lives of the various stakeholders who seek to benefit from an improved senior design sign up/proposal process. HIGH - We are highly focused and driven to determine how our project can improve the lives of the users involved. One such area we have identified is saving our stakeholders time in the proposal and assignment process of a senior design project. |

## 7.3 Most Applicable Professional Responsibility Area

Social Responsibility is a vital area of responsibility within our product. We aim to produce a web-server and algorithm that benefits the community of Iowa State staff and students. Our product is the essence of this Responsibility, being it is a tool to benefit everyone involved in Senior Design.

Our team has demonstrated Social Responsibility through two distinct pieces of the product. The matching algorithm streamlines the decision making process and makes sure everyone in senior design has an equal opportunity to get the project that they will benefit from the most and what project they are most qualified for (IEEE #8). Second, our project will create a web server that will make for easier accessibility for professors, clients, faculty, and students. By creating a web server and inherently centralizing and automating the Senior Design process, we achieve IEEE #10, because it creates an environment friendly for everyone involved with senior design and will greatly impact their experience with senior design.

# 8 Closing Material

## 8.1 Discussion

This first semester has been a great time for honing our skills as prospective engineers. With all the materials presented above, we have been able to learn while creating this product. We believe the design of this process will leave us with a usable product after we implement it in the next semester. After designing, meeting our defined responsibility areas and compiling user needs, we look forward to moving on with this design.

## 8.2 Conclusion

We continue to be on schedule for a release date at the end of the upcoming spring semester. The majority of this fall semester has been dedicated to creating a design for each of the major project components: web application and algorithm. We were able to successfully establish a design for the full stack of the web application and have begun the early stages of implementation. The algorithm design is in the final stages with an onlooking goal of initial implementation in the upcoming weeks.

## 8.3 References

No References

## 8.4 Appendices

### 8.4.1 Team Contract

**THE TEAM**

**Team Members: Brady Synstelien, Seth Gardner, Yunhao Yang, Matthew Karr, Jake Gudenkauf, Gavin George, Kyle Kent**

**Required Skill Sets for Your Project:** (if feasible – tie them to the requirements)

Coding (Frontend/Backend knowledge), Building and connecting Database, Agile Methodology, Client interaction, Continuous Integration Continuous Deployment (CICD)

**Skill Sets Covered by the Team:** (for each skill, state which team member(s) cover it)

Coding (Frontend/Backend knowledge): Gavin, Tyler, Seth

Building and connecting Database: Gavin,

Agile Methodology: Gavin, Tyler, Seth

Client interaction: Gavin, Tyler, Seth

Continuous Integration Continuous Deployment (CICD), Seth

**Project Management Style Adopted by the Team:**

Agile

**Initial Project Management Roles:**

See leadership section

**Team Name** _____sdmay22-44_____

**Team Members:**

1) ___Gavin George_____ 2) ____Kyle Kent_____
3) ___Brady Synstelien_____ 4) _____Seth Gardner_____
5) ___Yunhao Yang_____ 6) Tyler Staker
7) ___Jake Gudenkauf_____ 8) _____Matthew Karr_____

**Team Procedures**

1. Day, time, and location (face-to-face or virtual) for regular team meetings:
    a. **Regular team meetings will be held weekly on Thursdays at 6:30 p.m virtually on discord.**
2. Preferred method of communication updates, reminders, issues, and scheduling (e.g., e-mail, phone, app, face-to-face):
    a. **We have set up a Discord server that holds most of our communication. Additionally, we can email each other if needed.**
3. Decision-making policy (e.g., consensus, majority vote):
    a. **If we are unable to come to a consensus on an issue we will hold a vote and move forward with whatever decision the majority decides is best for the team.**
4. Procedures for record keeping (i.e., who will keep meeting minutes, how will minutes be shared/archived):
    a. **Meeting minutes will be tracked via the discord application or if need be a group member. We have a shared document for recording meetings with the client.**

**Participation Expectations**

1. Expected individual attendance, punctuality, and participation at all team meetings:
    a. **If we don't specify we can't make a meeting beforehand, we expect all to participate in team meetings at the specified time.**
2. Expected level of responsibility for fulfilling team assignments, timelines, and deadlines:

    a. **We expect all to be involved in assigning and completing necessary tasks and assignments before any predefined deadlines.**
3. Expected level of communication with other team members:
    a. **We expect that each of us are engaged in weekly communication on discord**
4. Expected level of commitment to team decisions and tasks:
    a. **We expect that all of us are involved in critical decisions and generating various tasks for this project.**

**Leadership**

1. Leadership roles for each team member (e.g., team organization, client interaction, individual component design, testing, etc.):
    - **Brady ~ individual component design**
    - **Seth ~ individual component design, Project Management (Boards, meetings)**
    - **Kyle ~ testing, code reviewer, Git Documents manager**
    - **Matthew ~ Database manager**
    - **Yunhao Yang ~ individual component design, testing**
    - **Gavin ~ UI design manager, Code Reviewer**
    - **Jake ~CICD manager**
    - **Tyler ~Engineering Standards manager**

2. Strategies for supporting and guiding the work of all team members:
    a. **Having discord channels specifically to ask questions and get help if you run into issues.**
    b. **Creating comments on Git issues where these discussions can be saved and used to show how we come up with our solutions.**

3. Strategies for recognizing the contributions of all team members:
    a. **Tracking task completion with GitLab issues will allow us to see all contributions team members have made.**

**Collaboration and Inclusion**

1. Describe the skills, expertise, and unique perspectives each team member brings to the team.
    - **Brady has some prior coding experience with typescript, python, HTML/SCSS, postman, GO, EngineX, java, Git, javascript, ionic, and angular.**
    - **Tyler has prior experience with C/C++/C#, Java/JavaScript, HTML/CSS, MySQL, and Unity**
    - **Seth has previous web dev, AWS development, Agile team experience. Has experience with Java, Python, C, Typescript, PHP, HTML, CSS. Has worked with Springboot, Laravel, Angular. Used GitHub, Postman, Azure Dev Ops.**
    - **Yunhao Yang has prior experience with C\C++, C#, Java, Unity, MySQL, JUNIT, and HTML. Has experience of a personal game development project. Used GitHub, and Postman.**

- **Gavin has experience with C\C++, C#, Java, Javascript, Typescript, Angular, HTML/CSS, SQL. Previous agile full stack development on a web application. Used GitHub, Postman, Azure Dev Ops.**
- **Jake has prior experience with C/C++, Java, SQL, JavaScript, HTML, CSS, Springboot, React, Python, Git,**
- **Kyle has previous experience with Javascript/Java, HTML/CSS, React, Springboot, C, MySQL. Agile team experience with continuous development and continuous deployment. Test driven development with Jest, Enzyme, and Mockito testing libraries.**
- **Matthew has previous experience with C/C++, C#, Java, Javascript, HTML, CSS, XML, PHP.**

2. Strategies for encouraging and support contributions and ideas from all team members:
    a. **Make sure everyone feels heard, explore all ideas proposed. Open the decision making process to everyone. Allow everyone to voice their opinion before we move forward with a specific plan. While doing this, we will also keep discussions professional and civil, recognizing that coding can be an emotional experience and that we are all united in trying to achieve the best possible product.**

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?):
    a. **Actively involve everyone in the decision making process, making sure that everyone's perspectives are heard and analyzed. If someone feels like they are falling behind, the team will do their best to assist them in the best way possible.**

**Goal-Setting, Planning, and Execution**

1. Team goals for this semester:
    a. **Develop a well structured plan of how to implement each part of the project. We would like our project to be utilized next year for the Senior Design course to streamline project assignment and development. To accomplish this, we aim to satisfy and exceed the client's requirements.**

2. Strategies for planning and assigning individual and team work:
    a. **Utilize estimated time and tags on gitlab issues to try and balance work and make sure that tasks are given to team members who are able to complete them.**

3. Strategies for keeping on task:
    a. **Using the GitLab milestones to keep track of sprint deadlines and ensure that the tasks are completed on time.**

**Consequences for Not Adhering to Team Contract**

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

a. **As a group we will communicate with the person who commited the infraction and try to come to a satisfactory solution.**

2. What will your team do if the infractions continue?
    a. **If there are some issues that cannot be solved by an in-group meeting, then we will report to the TA to seek guidance on how to come to a proper solution. If the issue is related to any agreement between the team and the client, then we will ask the client for their opinion**

**********

*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) _____Brady Synstelien_____ DATE ____09/22/2021_____
2) Tyler Staker DATE 09/22/2021
3) _____Yunhao Yang_____ DATE __09/22/2021_____
4) ___Gavin George_____ DATE _09/22/2021_____
5) _____Seth Gardner_____ DATE __09/22/2021_____
6) _____Jake Gudenkauf_____ DATE ____09/22/2021_____
7) _____Kyle Kent_____ DATE _09/22/2021_____
8) _____Matthew Karr_____ DATE ____9/22/2021_____