Senior Design Server/Client Development for Project Matching (Phase 2)

Team 18
Client & Advisor: Akhilesh Tyagi

Team Introductions

Haylee Lawrence
Software Engineering

UI Designer & Lead Presenter

MyTien Kien
Software Engineering

Team Organization & Client Interaction

Sanjana Amatya
Software Engineering

Individual Component Design & Report Manager

Alec Elsbernd
Software Engineering

Lead Researcher & Floating Help

Introduction

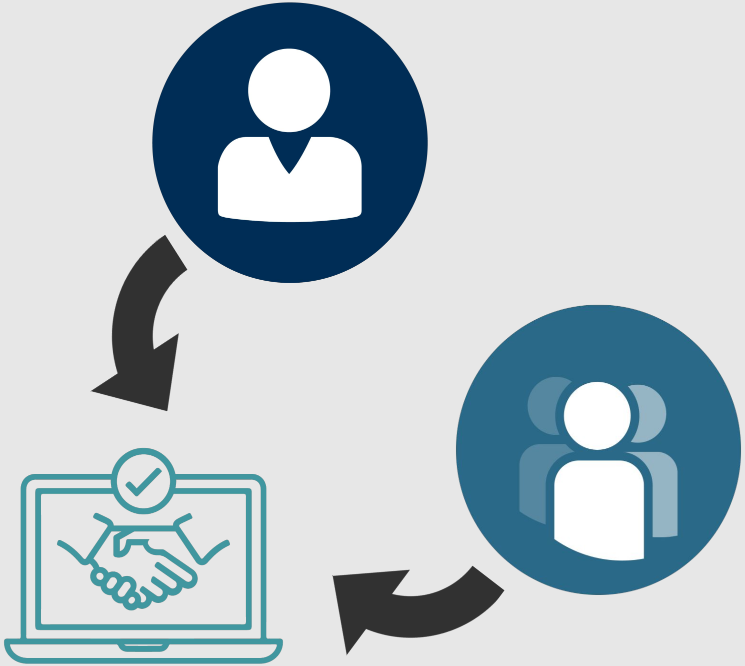
Senior Design Project Matching Phase 2

Currently…

- Matching process time consuming
- Can lead to client/student dissatisfaction

Main Use Cases

- **Clients**: Submit Project Proposals
- **Students**: Input Project Preferences
- **Instructors**: Create optimal Project Groups



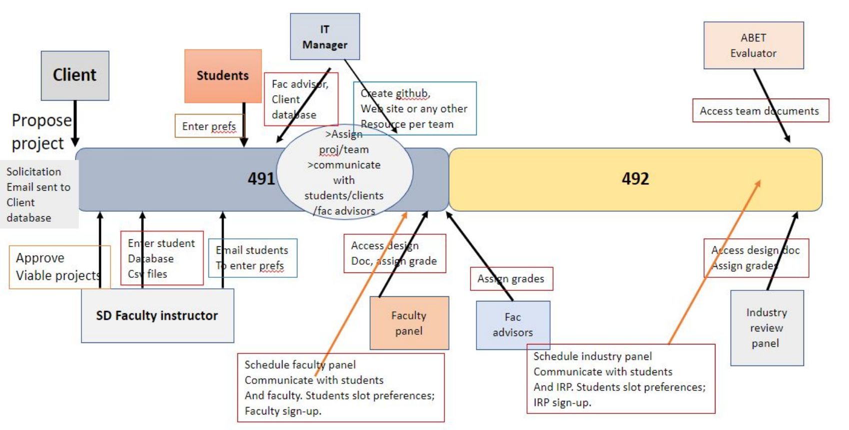
Introduction

What?

- A system that captures the senior design cycle from beginning to end
- Main focus on the project matching system
- Easier experience for everyone involved in the process

What's so unique?

- Project matching is an example of a classical assignment problem
- Using Project Matching algorithm
- Overhauled frontend



IT Manager
ABET Evaluator
Client
Students
Fac advisor, Client database
Create github, Web site or any other Resource per team
Access team documents
Propose project
Enter prefs
>Assign proj/team >communicate with students/clients /fac advisors
491
492
Solicitation Email sent to Client database
Approve Viable projects
Enter student Database Csv files
Email students To enter prefs
Access design Doc, assign grade
Assign grades
Access design doc Assign grades
SD Faculty instructor
Faculty panel
Fac advisors
Industry review panel
Schedule faculty panel Communicate with students And faculty. Students slot preferences; Faculty sign-up.
Schedule industry panel Communicate with students And IRP. Students slot preferences; IRP sign-up.

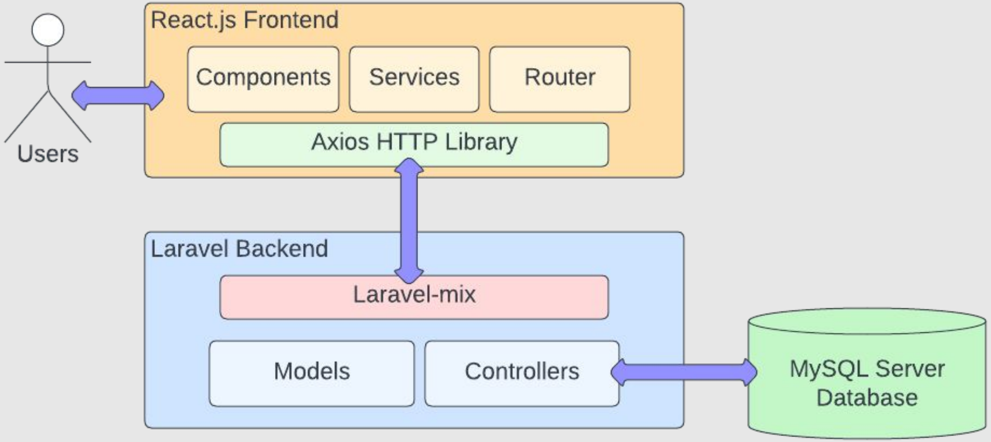
Implementation Architecture

Frontend
- Figma Wireframe
- Layered approach with React components
- React Router capabilities

Backend
- Laravel application
- Contains different packages
- Used axios to call HTTP requests

Database
- Various tables for

    Users
    Projects + preferences
    Groups



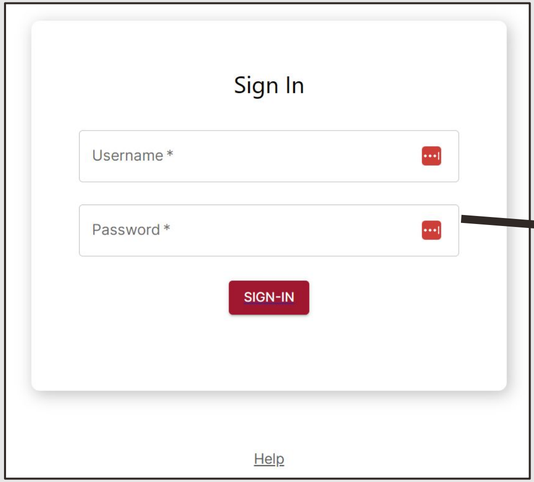
Work Accomplishments

We have accomplished a lot since the beginning of the new semester. In the past few weeks we have

- Created a fully functional Visual Frontend
- Created and deploying the backend
- Set-up Database
- Frontend communicating with the Database
- Researched, Modified, and Coded a Project Matching Algorithm

Instructor Frontend

Instructor Logs In

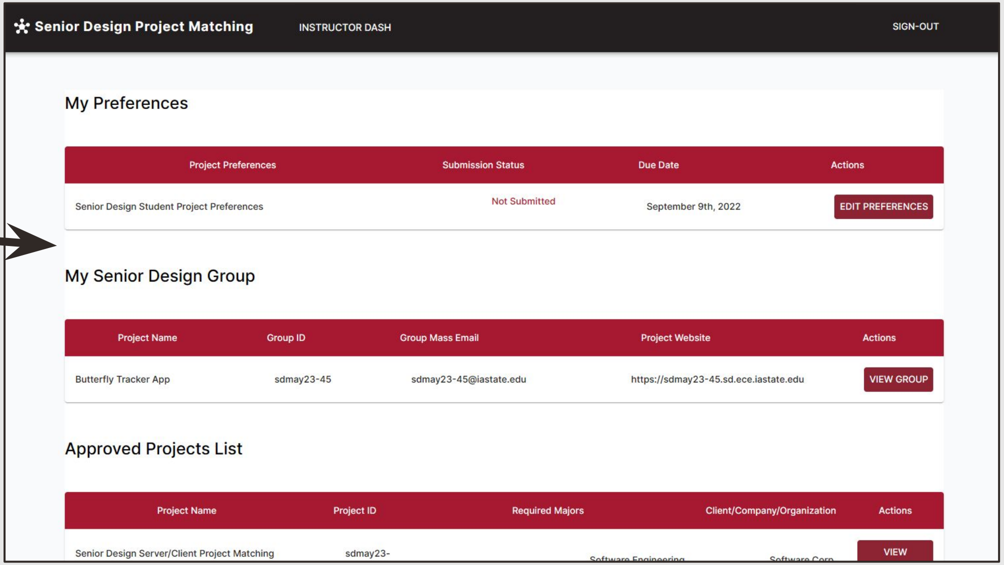


Sign In

Username *

Password *

Dashboard View



My Preferences

Project Preferences	Submission Status	Due Date	Actions
Senior Design Student Project Preferences	Not Submitted	September 9th, 2022	EDIT PREFERENCES

My Senior Design Group

Project Name	Group ID	Group Mass Email	Project Website	Actions
Butterfly Tracker App	sdmay23-45	sdmay23-45@iastate.edu	https://sdmay23-45.sd.ece.iastate.edu	VIEW GROUP

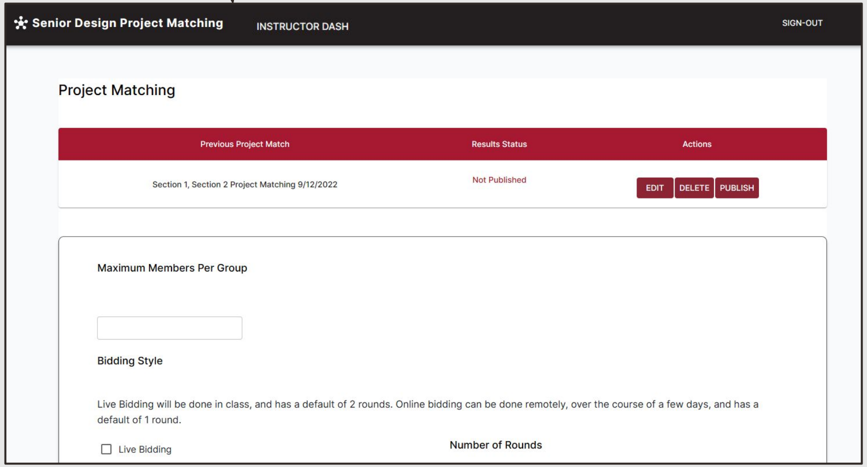
Approved Projects List

Project Name	Project ID	Required Majors	Client/Company/Organization	Actions
Senior Design Server/Client Project Matching	sdmay23-	Software Engineering	Software Corp	VIEW

Instructor Frontend

Dashboard

Project Matching



Project Matching

Previous Project Match	Results Status	Actions
Section 1, Section 2 Project Matching 9/12/2022	Not Published	EDIT DELETE PUBLISH

Maximum Members Per Group

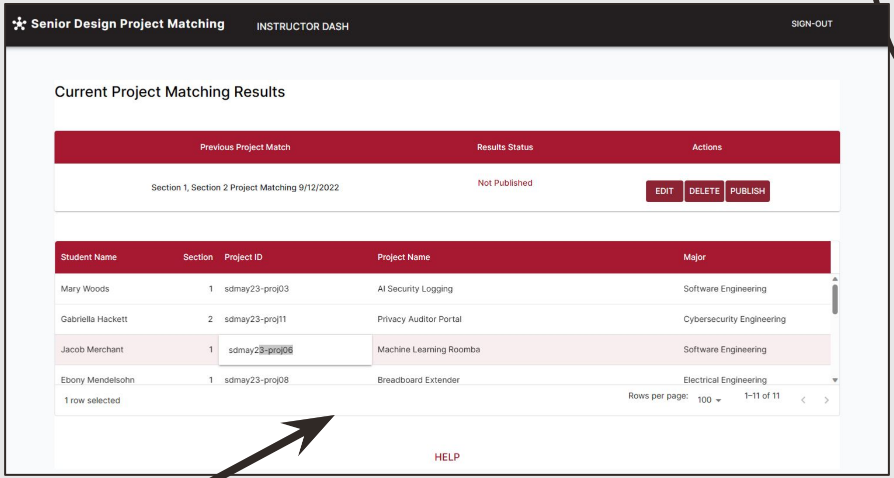
Bidding Style

Live Bidding will be done in class, and has a default of 2 rounds. Online bidding can be done remotely, over the course of a few days, and has a default of 1 round.

☐ Live Bidding

Number of Rounds

Project Matching Results



Current Project Matching Results

Previous Project Match	Results Status	Actions
Section 1, Section 2 Project Matching 9/12/2022	Not Published	EDIT DELETE PUBLISH

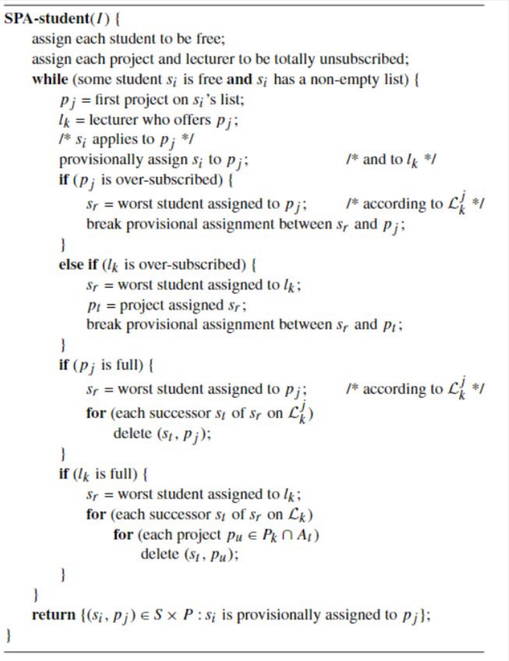
Student Name	Section	Project ID	Project Name	Major
Mary Woods	1	sdmay23-proj03	AI Security Logging	Software Engineering
Gabriella Hackett	2	sdmay23-proj11	Privacy Auditor Portal	Cybersecurity Engineering
Jacob Merchant	1	sdmay23-proj06	Machine Learning Roomba	Software Engineering
Ebony Mendelsohn	1	sdmay23-proj08	Breadboard Extender	Electrical Engineering

1 row selected

Rows per page: 100

1–11 of 11

Algorithm - Paper Version



```
SPA-student($I$) {
    assign each student to be free;
    assign each project and lecturer to be totally unsubscribed;
    while (some student $s_i$ is free and $s_i$ has a non-empty list) {
        $p_j$ = first project on $s_i$'s list;
        $l_k$ = lecturer who offers $p_j$;
        /* $s_i$ applies to $p_j$ */
        provisionally assign $s_i$ to $p_j$;                /* and to $l_k$ */
        if ($p_j$ is over-subscribed) {
            $s_r$ = worst student assigned to $p_j$;        /* according to $\mathcal{L}_k^j$ */
            break provisional assignment between $s_r$ and $p_j$;
        }
        else if ($l_k$ is over-subscribed) {
            $s_r$ = worst student assigned to $l_k$;
            $p_t$ = project assigned $s_r$;
            break provisional assignment between $s_r$ and $p_t$;
        }
        if ($p_j$ is full) {
            $s_r$ = worst student assigned to $p_j$;        /* according to $\mathcal{L}_k^j$ */
            for (each successor $s_t$ of $s_r$ on $\mathcal{L}_k^j$)
                delete ($s_t$, $p_j$);
        }
        if ($l_k$ is full) {
            $s_r$ = worst student assigned to $l_k$;
            for (each successor $s_t$ of $s_r$ on $\mathcal{L}_k$)
                for (each project $p_u \in P_k \cap A_t$)
                    delete ($s_t$, $p_u$);
        }
    }
    return $\{(s_i, p_j) \in S \times P : s_i$ is provisionally assigned to $p_j\}$;
}
```

SPA Pseudocode

Abraham et al’s matching algorithm aims to match projects by using worker preferences and and requirements.

1. Each worker is given a score based on their preferences for different types of projects, and each project is given a score based on its requirements for different types of workers.
2. Abraham’s algorithm uses a variant of the Gale Shapley algorithm (stable marriage algorithm) to try to match workers to projects by assigning workers to projects that score highly for each other.
3. It does this by using a mathematical formula to calculate the optimal match between workers and projects based on their scores.
    a. The formula takes into account preferences of worker
    b. Requirements of projects
    c. Quality of the match
4. The algorithm is iterative and keeps running to improve the quality of matches over time.

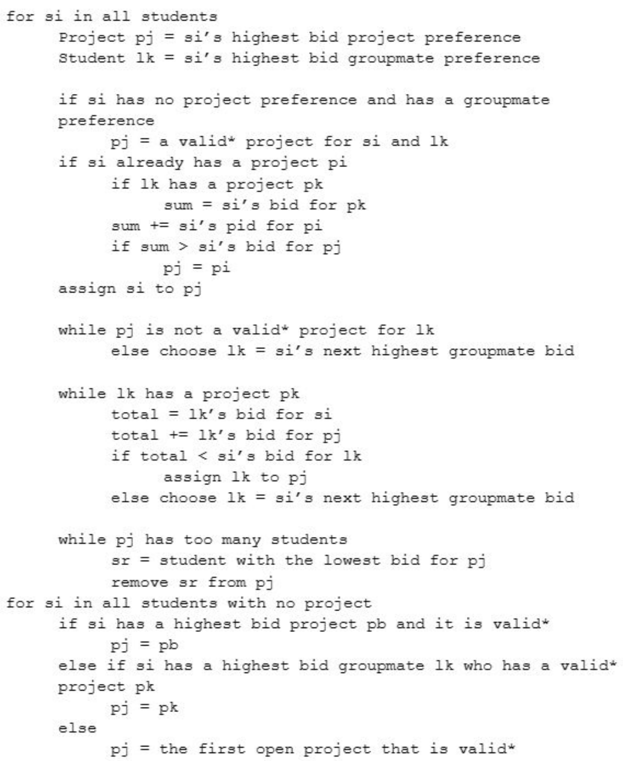
Algorithm - Our Version

What is Different?

- Matches Students based on Groupmates instead of Lecturers
- Extra Complexity – Checks Groupmates Preferences before Grouping with other Students
- Checks if Projects are Valid - allow a Students Major

Implementation

- Coded in Java using Student, Project, and Preferences Classes
- Right now takes Manually inputted Students, Projects, and Preferences
- Outputs the Project Matchings
- For Simplicity
    - 3 Project Preferences and 3 Groupmate Preferences
    - Aim to match 1 pair of Groupmates per Project



```
for si in all students
    Project pj = si’s highest bid project preference
    Student lk = si’s highest bid groupmate preference

    if si has no project preference and has a groupmate
    preference
        pj = a valid* project for si and lk
    if si already has a project pi
        if lk has a project pk
            sum = si’s bid for pk
        sum += si’s pid for pi
        if sum > si’s bid for pj
            pj = pi
    assign si to pj

    while pj is not a valid* project for lk
        else choose lk = si’s next highest groupmate bid

    while lk has a project pk
        total = lk’s bid for si
        total += lk’s bid for pj
        if total < si’s bid for lk
            assign lk to pj
        else choose lk = si’s next highest groupmate bid

    while pj has too many students
        sr = student with the lowest bid for pj
        remove sr from pj
for si in all students with no project
    if si has a highest bid project pb and it is valid*
        pj = pb
    else if si has a highest bid groupmate lk who has a valid*
    project pk
        pj = pk
    else
        pj = the first open project that is valid*
```

Our Project Matching Pseudocode

Abraham et al’s algorithm (in depth)

The objective function is the overall quality of matches, and the constraints ensure that each worker is assigned to only one project and each project is assigned to only one worker.

1. The quality of a match is calculated using the dot product of the worker's preference vector and the project's requirement vector.
2. The preference vector represents the worker's preferences for different types of projects, and the requirement vector represents the project's requirements for different types of workers.

The algorithm uses a probabilistic approach to achieve this goal with the following steps:

1. Initialization: For each project and worker, assign a value of 0.
2. Assignment: For each project, choose a random permutation of the available workers, and assign the first available worker to the project. Continue in this way until all projects are assigned.
3. Improvement: For each worker, calculate the expected value of the project they are assigned to, based on the preferences of the worker and the requirements of the project. If the expected value is higher than the current value of the worker, then reassign the worker to a better project.
4. Steps 2 and 3 are repeated until no further improvements can be made.

Abraham et al's formula

The formula used to calculate the expected value of a worker's assignment is:

$$E[V(w,p)] = \text{sum}_i (p_i * \max(0, w_j - r_{ij}))$$

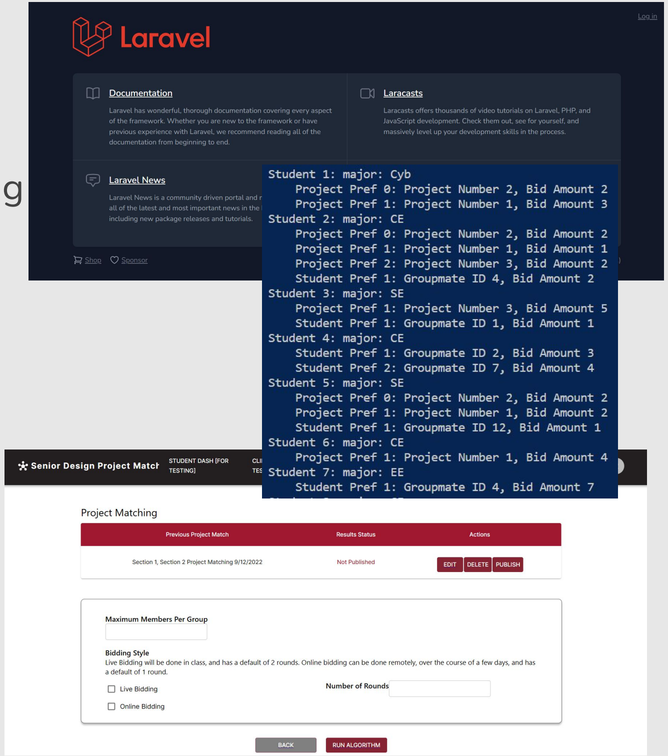
- w is the preference vector of the worker
- p is the probability vector of the project (i.e., the probability of assigning the worker to each project in the current assignment)
- r is the requirement matrix of the projects (i.e., the number of workers required for each project and skill combination)
- i is an index over the projects, j is an index over the skills

This formula calculates the expected value of the worker's assignment as the sum of the product of the probability of being assigned to each project and the maximum of 0 and the difference between the worker's preference for the skill and the project's requirement for the skill.

Works Cited: *On the Power of Randomization in Algorithmic Mechanism Design*. https://viterbi-web.usc.edu/~shaddin/papers/randompower-current.pdf.

Key Contributions

- Haylee
  - Designed, Coded, and Tested Frontend
  - Researched, Coded, and Tested Project Matching Algorithm
- MyTien
  - Connected the backend, frontend, database
  - Backend routing and API
- Sanjana
  - Worked to connect frontend and backend
  - Backend controllers
  - Project Matching Algorithm research
- Alec
  - Previous team backend research
  - Various backend contributions
  - Algorithm research



Documentation

Laravel has wonderful, thorough documentation covering every aspect of the framework. Whether you are new to the framework or have previous experience with Laravel, we recommend reading all of the documentation from beginning to end.

Laracasts

Laracasts offers thousands of video tutorials on Laravel, PHP, and JavaScript development. Check them out, see for yourself, and massively level up your development skills in the process.

Laravel News

Laravel News is a community driven portal and r all of the latest and most important news in the including new package releases and tutorials.

```
Student 1: major: Cyb
    Project Pref 0: Project Number 2, Bid Amount 2
    Project Pref 1: Project Number 1, Bid Amount 3
Student 2: major: CE
    Project Pref 0: Project Number 2, Bid Amount 2
    Project Pref 1: Project Number 1, Bid Amount 1
    Project Pref 2: Project Number 3, Bid Amount 2
    Student Pref 1: Groupmate ID 4, Bid Amount 2
Student 3: major: SE
    Project Pref 1: Project Number 3, Bid Amount 5
    Student Pref 1: Groupmate ID 1, Bid Amount 1
Student 4: major: CE
    Student Pref 1: Groupmate ID 2, Bid Amount 3
    Student Pref 2: Groupmate ID 7, Bid Amount 4
Student 5: major: SE
    Project Pref 0: Project Number 2, Bid Amount 2
    Project Pref 1: Project Number 1, Bid Amount 2
    Student Pref 1: Groupmate ID 12, Bid Amount 1
Student 6: major: CE
    Project Pref 1: Project Number 1, Bid Amount 4
Student 7: major: EE
    Student Pref 1: Groupmate ID 4, Bid Amount 7
```

Project Matching

Previous Project Match	Results Status	Actions
Section 1, Section 2 Project Matching 9/12/2022	Not Published	EDIT DELETE PUBLISH

Maximum Members Per Group

Bidding Style
Live Bidding will be done in class, and has a default of 2 rounds. Online bidding can be done remotely, over the course of a few days, and has a default of 1 round.

□ Live Bidding

□ Online Bidding

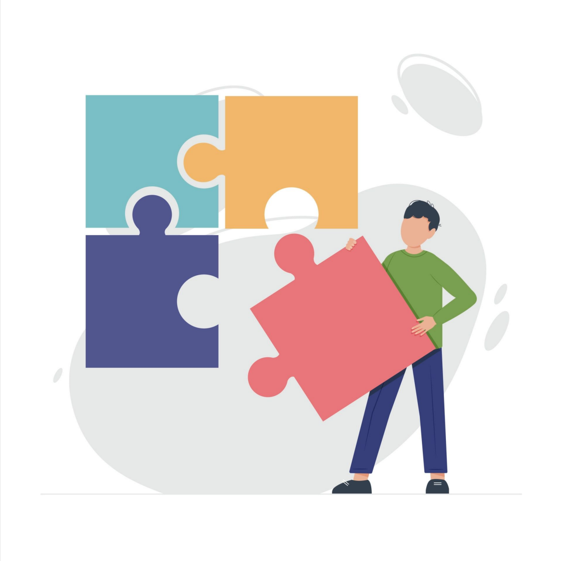
Number of Rounds

Challenges and Solutions

- Learning curve with Laravel
  - Lots of documentation and research
  - Help from our IT Admin
- Connecting Laravel with other components
  - Algorithm and frontend
  - Tutorials, teamwork
- Algorithm and heuristics
  - Research papers
  - Guiding from our advisor
  - Not too familiar with algorithms
- Time and knowledge constraints

Future Work

- Add and allow ABET evaluators access to the website
- Adding more connections from the backend to the frontend
- Implementing a way for Board members and Instructors to sign up for future time slots
- Future implementations of the algorithm
  - Customization for number of groupmates in a project, project skill-level requirements, etc.
  - More bidding iterations to maximize client/student satisfaction
  - Satisfaction % customization (sacrifice student satisfaction to meet skill-level requirements)
  - Dealing with non-ideal conditions (more projects than students)



Conclusion