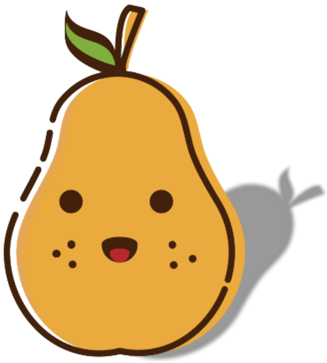
CST8333

Assignment 01



# Pear

## A Discord User Pairing Bot

Kaitlyn Gatineau
041014962

June 9th 2022

All material prepared for this assignment was produced by the author. Material from all third parties has been cited and referenced.

This project is not affiliated, associated, authorized, endorsed by, or in any way officially connected to Discord™ or any of its affiliates.

## Table of Contents

# Introduction

As the community manager of a Discord group called the *Femmes and Thems in Tech*, I oversee, engage with and host events for thousands of spectacular women and nonbinary users who utilize our platform daily. Every week, it is my duty to manually pair individuals up who opt in to “break the ice” for those participating and allow for users to meet someone new every week.

Pear is a tool that was not only created for the sake of a school project but serves as the first iteration of a solution to a real-life problem in our community. Manually pairing users worked on a small scale, but the process needed automation and the type of tool necessary to make this happen did not exist at the time of writing for the Discord platform.

Automating the pairing process using a Discord bot not only saves time for the moderation team, but it allows for accuracy and consistency for when pairings would be posted.

While the bot will be originally utilized to pair people up for networking, it is intended to work as a pairing system for any scenario a Discord admin may deem necessary. Our goal is to create a versatile pairing bot that pairs users for any social/logistical need.

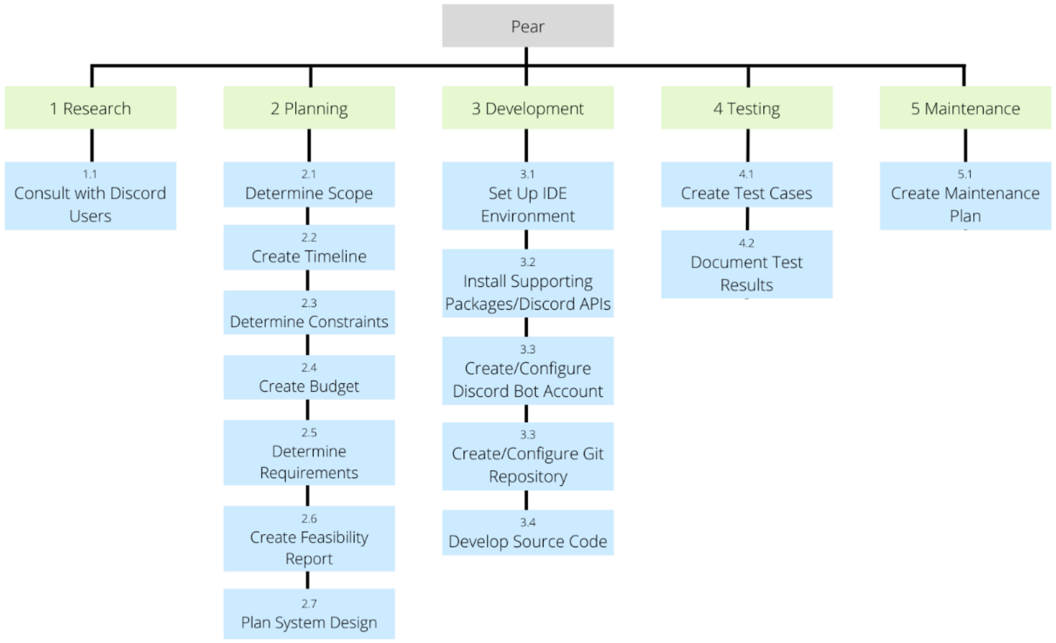
## Approach

This project (in respect to the version being created for the purpose of this course) is being conducted using a modified waterfall lifecycle. The project will be done in three phases, each in respect to the assignment/presentation due at the end of each phase.

The measurement criteria for each phase will be the checklists/rubrics associated with the assignment/presentation. 60% of the source code must be completed by the end of phase two, and 100% completed by the end of phase three. The phase will be deemed successful should the majority of checklist/rubric goals for that phase be met.

## Work Breakdown Structure

Below is a first draft of the WBS for this project. This is subject to change should new needs arise.



## Version Control

Version control will be maintained using a Git repository. The development team plans to utilize the GitHub platform to maintain the code.

## Objective

The overall objective is to create a function bot that works as intended by pairing users upon request. This main objective can be broken down into the following:

**Table 1: Objectives and Business Outcomes**

| No. | OBJECTIVE | BUSINESS OUTCOME | MEASUREMENT CRITERIA |
|---|---|---|---|
| 1 | Bot only pairs users who opt into the pairing system versus all users/online users in the Discord. | Users who wish to be paired are paired, and those who do not wish to be paired are not bothered by the bot. | Users who are opted in are pinged (@ tagged); users who are not are left alone. |
| 2 | Bot is only responsive to commands made by Administrator ranks. | Users can enjoy the pairing system without unnecessary/duplicate pairings triggered by unauthorized users. | User with Administrator rank can trigger the bot via command;<br>Bot does not respond to general user commands. |

| No. | OBJECTIVE | BUSINESS OUTCOME | MEASUREMENT CRITERIA |
|---|---|---|---|
| | | Administrators can rely on the bot to not populate the server with unwanted posts. | |
| 3 | Bot activity is limited to the designated channel. | Users know where to look for the postings with ease. | Bot only triggers from commands and posts in a designated channel. |
| 4 | Bot fetches user IDs from one designated post only. | Users who reacted to unrelated posts in the Discord are not accidently considered in the pairings. | Bot only pairs users who react to a single post.<br>Users who react to other posts are not included. |

## Scope

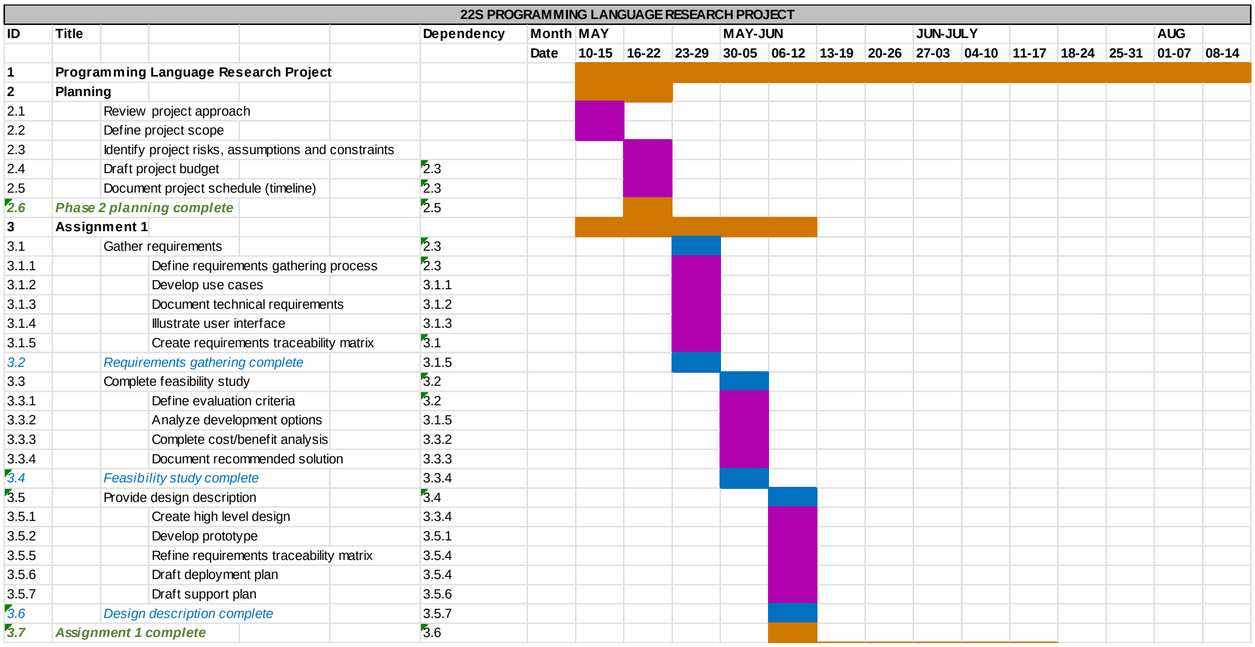
This section describes the features of the bot that will be implemented for the scope of this course. Features beyond the scope are listed as “Activities Out of Scope”.

**Table 2: Project Scope**

| ACTIVITIES IN SCOPE | ACTIVITIES OUT OF SCOPE |
|---|---|
| 1. Assignment 1 | 1. Apply for business financing |
| 2. Fetch only user IDs/usernames of users who “reacted” to a designated post to be entered into the pairing pool to ensure only those who opt in are paired. | 2. Store pairing data in a database schema |
| 3. Randomize the collection order to ensure users are not paired by their place on the “reaction” list. | 3. Ensure no two users are paired multiple weeks in a row |
| 4. Ensure the post can only be triggered by an “Administrator” rank of the Discord server. | 4. Create logic for the bot to post at a designated time without the need to be manually triggered by command. |
| 5. Create a mock Discord server for development with minimum six users for testing. | 5. Purchase a hosting service to maintain high availability uptime without the need for an admin to host the bot on a personal machine |
| 6. Create a GitHub repository to maintain the code | |
| 7. Remove reactions by users who have left the Discord. | |

## Timeline

This section states the timeline for the project deliverables to be completed. Double-clicking on the image below will show the spreadsheet in full.



22S PROGRAMMING LANGUAGE RESEARCH PROJECT

| ID | Title | Dependency | Month | MAY | | MAY-JUN | | | | JUN-JULY | | | | | AUG | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Date | 10-15 | 16-22 | 23-29 | 30-05 | 06-12 | 13-19 | 20-26 | 27-03 | 04-10 | 11-17 | 18-24 | 25-31 | 01-07 | 08-14 |
| 1 | **Programming Language Research Project** | | | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 |
| 2 | **Planning** | | | 🟧 | 🟧 | | | | | | | | | | | | |
| 2.1 | Review project approach | | | 🟪 | | | | | | | | | | | | | |
| 2.2 | Define project scope | | | 🟪 | | | | | | | | | | | | | |
| 2.3 | Identify project risks, assumptions and constraints | | | | 🟪 | | | | | | | | | | | | |
| 2.4 | Draft project budget | 2.3 | | | 🟪 | | | | | | | | | | | | |
| 2.5 | Document project schedule (timeline) | 2.3 | | | 🟪 | | | | | | | | | | | | |
| *2.6* | ***Phase 2 planning complete*** | 2.5 | | | 🟧 | | | | | | | | | | | | |
| 3 | **Assignment 1** | | | 🟧 | 🟧 | 🟧 | 🟧 | 🟧 | | | | | | | | | |
| 3.1 | Gather requirements | 2.3 | | | | 🟦 | | | | | | | | | | | |
| 3.1.1 | Define requirements gathering process | 2.3 | | | | 🟪 | | | | | | | | | | | |
| 3.1.2 | Develop use cases | 3.1.1 | | | | 🟪 | | | | | | | | | | | |
| 3.1.3 | Document technical requirements | 3.1.2 | | | | 🟪 | | | | | | | | | | | |
| 3.1.4 | Illustrate user interface | 3.1.3 | | | | 🟪 | | | | | | | | | | | |
| 3.1.5 | Create requirements traceability matrix | 3.1 | | | | 🟪 | | | | | | | | | | | |
| *3.2* | *Requirements gathering complete* | 3.1.5 | | | | 🟦 | | | | | | | | | | | |
| 3.3 | Complete feasibility study | 3.2 | | | | | 🟦 | | | | | | | | | | |
| 3.3.1 | Define evaluation criteria | 3.2 | | | | | 🟪 | | | | | | | | | | |
| 3.3.2 | Analyze development options | 3.1.5 | | | | | 🟪 | | | | | | | | | | |
| 3.3.3 | Complete cost/benefit analysis | 3.3.2 | | | | | 🟪 | | | | | | | | | | |
| 3.3.4 | Document recommended solution | 3.3.3 | | | | | 🟪 | | | | | | | | | | |
| *3.4* | *Feasibility study complete* | 3.3.4 | | | | | 🟦 | | | | | | | | | | |
| 3.5 | Provide design description | 3.4 | | | | | | 🟦 | | | | | | | | | |
| 3.5.1 | Create high level design | 3.3.4 | | | | | | 🟪 | | | | | | | | | |
| 3.5.2 | Develop prototype | 3.5.1 | | | | | | 🟪 | | | | | | | | | |
| 3.5.5 | Refine requirements traceability matrix | 3.5.4 | | | | | | 🟪 | | | | | | | | | |
| 3.5.6 | Draft deployment plan | 3.5.4 | | | | | | 🟪 | | | | | | | | | |
| 3.5.7 | Draft support plan | 3.5.6 | | | | | | 🟪 | | | | | | | | | |
| *3.6* | *Design description complete* | 3.5.7 | | | | | | 🟦 | | | | | | | | | |
| *3.7* | ***Assignment 1 complete*** | 3.6 | | | | | | 🟧 | | | | | | | | | |

Clicking the icon below will open the full version in Excel. Please note, this Gantt Chart was created using the template provided, and is cited as a source in the *References*.

CST8333 Gantt
Chart - Kaitlyn Gatin

## Milestones and Deliverables

This section lists major milestones/deadlines required by this project.

### Table 4: Project Milestones and Deliverables

| MILESTONE | DATE | DELIVERABLES |
|---|---|---|
| 1. Assignment 1 - Complete | June 9 2022 | Written report and slide presentation |
| 2. Assignment 2 - Complete | July 14 2022 | Written report, slide presentation and 60% of source code |
| 3. Assignment 3 - Complete | August 11 2022 | Written report, slide presentation and 100% of source code |

## Risks

This section mentions the risks that are associated with this project.

**Table 5: Project Risks** (example)

| No. | RISK DESCRIPTION | PROBABILITY (H/M/L) | IMPACT (H/M/L) | MITIGATION |
|---|---|---|---|---|
| 1. | Schedule slippage | M | H | Track project scope and timeline |
| 2. | Discord server outage | L | M | Wait until Discord resolves the issue(s) |
| 3. | Hardware Malfunction | L | H | Back up project to external source; have a spare machine ready |
| 4. | Lack of Knowledge | M | M | Reach out for help from the internet/experienced developers |

## Assumptions

**Table 6: Project Assumptions**

| No. | THE FOLLOWING IS ASSUMNED |
|---|---|
| 1. | Fundamentals of new programming language will be learned and put to use, timely, to complete project |
| 2. | Bot will only run in a single server, via a single host machine at any given time |
| 3. | Tracking previous pairings is not necessary for the scope of this project |

## Technical Constraints

**Table 7: Technical Constraints**

| No. | TECHNICAL CONSTRAINTS |
|---|---|
| 1. | Program requires a user to execute on a host computer for it to function |

## Budget

Below is the proposed budget for the project. The project is measured in “Level of Effort” instead of currency as no money will be spent on this project.

Please note, the Project Budget was created using the template provided, and is cited as a source in the *References*.

**Table 8: Project Budget**

| **ID** | **Title** | **Level of Effort (Days)** |
|---|---|---|
| **1** | **Programming Language Research Project** | 57.20 |
| **2** | **Planning** | 5.00 |
| 2.1 | Review project approach | 1.00 |
| 2.2 | Define project scope | 1.00 |
| 2.3 | Identify project risks, assumptions and constraints | 1.00 |
| 2.4 | Draft project budget | 1.00 |
| 2.5 | Document project schedule (timeline) | 1.00 |
| ***2.6*** | ***Planning Complete*** | 0.00 |
| **3** | **Assignment 1** | 12.50 |
| 3.1 | Gather requirements | 5.00 |
| 3.1.1 | Define requirements gathering process | 1.00 |
| 3.1.2 | Develop use cases | 1.00 |
| 3.1.3 | Describe data requirements | 0.50 |
| 3.1.4 | Document technical requirements | 0.50 |
| 3.1.5 | Illustrate user interface | 1.00 |
| 3.1.6 | Create requirements traceability matrix | 1.00 |
| *3.2* | *Requirements gathering complete* | 0.00 |
| 3.3 | Complete feasibility study | 4.00 |
| 3.3.1 | Define evaluation criteria | 1.00 |
| 3.3.2 | Analyze development options | 1.00 |
| 3.3.3 | Complete cost/benefit analysis | 1.00 |
| 3.3.4 | Document recommended solution | 1.00 |
| *3.4* | *Feasibility study complete* | 0.00 |
| 3.5 | Provide design description | 3.50 |
| 3.5.1 | Create high level design | 1.00 |
| 3.5.2 | Develop prototype | 1.00 |
| 3.5.3 | Refine requirements traceability matrix | 0.50 |
| 3.5.4 | Draft deployment plan | 0.50 |
| 3.5.5 | Draft support plan | 0.50 |
| *3.6* | *Design description complete* | 0.00 |
| ***3.7*** | ***Assignment 1 complete*** | 0.00 |
| **4** | **Assignment 2** | 19.00 |
| 4.1 | Select tools and technology | 7.00 |
| 4.1.1 | Define selection criteria | 3.00 |
| 4.1.2 | Evaluate available tools and technologies | 3.00 |
| 4.1.3 | Select tools and technology | 1.00 |
| *4.2* | *Tools and technology selection complete* | 0.00 |
| 4.3 | Implement source code phase 1 | 12.00 |
| 4.3.1 | Translate design into code (min. 60%) | 3.00 |
| 4.3.2 | Evaluate code | 2.00 |
| 4.3.3 | Identify gaps in requirements and design | 2.00 |
| 4.3.4 | Evaluate implementation of code | 3.00 |
| 4.3.5 | Refine requirements traceability matrix | 2.00 |
| *4.4* | *Phase 1 source code implementation complete* | 0.00 |
| ***4.5*** | ***Assignment 2 complete*** | 0.00 |
| **5** | **Assignment 3** | 20.70 |
| 5.1 | Implement source code phase 2 | 5.00 |

| | | |
|---|---|---|
| 5.1.1 | Translate design into code (up to 100%) | 1.00 |
| 5.1.2 | Evaluate code | 1.00 |
| 5.1.3 | Identify gaps in requirements and design | 1.00 |
| 5.1.4 | Evaluate implementation of code | 1.00 |
| 5.1.5 | Refine requirements traceability matrix | 1.00 |
| *5.2* | *Phase 2 source code implementation complete* | 0.00 |
| 5.3 | Complete testing | 5.20 |
| 5.3.1 | Refine test plan (traceability matrix) | 0.40 |
| 5.3.2 | Prepare test tracking tools | 0.40 |
| 5.3.3 | Create test environment | 0.40 |
| 5.3.4 | Complete system testing | 0.40 |
| 5.3.5 | Implement build procedures | 0.40 |
| 5.3.6 | Load data into test environment | 0.40 |
| 5.3.7 | Develop test scripts | 0.40 |
| 5.3.8 | Execute test scripts | 0.40 |
| 5.3.9 | Correct defects (if any) | 0.40 |
| 5.3.10 | Document test results | 0.40 |
| 5.3.11 | Obtain approval of testing from Facilitator | 0.40 |
| 5.3.12 | Update support documents (if required) | 0.40 |
| 5.3.13 | Store test data | 0.40 |
| *5.4* | *Testing complete* | 0.00 |
| 5.5 | Complete deployment | 6.00 |
| 5.5.1 | Refine deployment plan | 1.00 |
| 5.5.2 | Prepare production environment | 1.00 |
| 5.5.3 | Complete data conversion activities | 1.00 |
| 5.5.4 | Complete operation and maintenance prep | 1.00 |
| 5.5.5 | Validate code functionality | 1.00 |
| 5.5.6 | Deploy code | 1.00 |
| *5.6* | *Deployment complete* | 0.00 |
| 5.7 | Complete presentation | 4.50 |
| 5.7.1 | Select slide presentation template | 1.50 |
| 5.7.2 | Develop presentation content | 1.50 |
| 5.7.3 | Submit presentation to Facilitator | 1.50 |
| *5.8* | *Presentation complete* | 0.00 |
| ***5.9*** | ***Assignment 3 complete*** | 0.00 |
| ***5.10*** | ***Programming Language Research Project complete*** | 0.00 |

# Requirements

## Introduction

This section lists the use cases and requirements deemed necessary for the success of the project. The requirements are broken down into business, technology, compliance and security.

## Use Cases

This section contains high level use cases to be implemented in the system design.

| | |
|---|---|
| **ID:** UC-01 | **Name:** Create reaction host post |
| **Actor:** Admin User | **Trigger:** "!create" command |
| **System:** Reads command (triggered by prefix "!"), @bot.command() create executes | **Goal:** Bot posts embed message as a response to the command that accepts user reactions |

| | |
|---|---|
| **ID:** UC-02 | **Name:** Pair users/post pair message |
| **Actor:** Admin User | **Trigger: "**!post" command |
| **System:** Reads command (triggered by prefix "!"), @bot.command() pair executes | **Goal:** Bot posts embed message as a response to the command that pairs users who reacted in groups of two |

| | |
|---|---|
| **ID:** UC-03 | **Name:** Delete user reactions no longer on server |
| **Actor:** Admin User | **Trigger: "**!post" command |
| **System:** Reads command (triggered by prefix "!"), @bot.command() pair executes | **Goal:** Delete users who are unable to be paired (due to no longer being a member of the server) before pairing/posting. |

## Business Requirements

The requirements listed below are listed to meet the needs of the business.

**Table 9: Business Requirements**

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
|---|---|---|---|---|
| B-01 | Create reaction collection post | Create the post to collect reactions using "!create! | Functional | Mandatory |
| B-02 | Fetch reaction list from collection post | Gather users who selected the reactions when "!post" is triggered | Functional | Mandatory |
| B-03 | Remove redundant reactions | Remove reactions of deleted users upon "!post" trigger | Functional | Mandatory |
| B-04 | Randomize fetched list | Randomize the list of users upon "!post" trigger | Functional | Mandatory |
| B-05 | Pair users | List the users in groups of two | Functional | Mandatory |
| B-06 | Post pair results | Send an embed message back to the client with the pair results | Functional, Reporting | Mandatory |

## Technology Requirements

The requirements listed below list the hardware, data, interface, application and availability requirements in technology for the application.

**Table 10: Technology Requirements**

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
|---|---|---|---|---|
| T-01 | Physical host | Running machine to host bot | Hardware, Availability | Mandatory |
| T-02 | User Interface | Create a UI that allows for user interaction | Application | Mandatory |
| T-03 | Input/Output – Creating Reaction Post | Ensure bot is triggered by the correct input and generates the correct output | Application | Mandatory |
| T-04 | Input/Output – Creating Pairing Post | Ensure bot is triggered by the correct input and generates the correct output | Application | Mandatory |
| T-05 | Name Format - Mention | Users in the pairing posts must be tagged properly using the @ notation | Data, Interface | Mandatory |

## Compliance Requirements

The requirements listed below are to ensure the project follows the rules and regulations imposed by Discord.

**Table 11: Compliance Requirements**

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
|---|---|---|---|---|
| C-01 | Discord Community Guidelines | Bot must comply with and not promote activity against Discord's Community Guidelines | Discord Rules and Regulations | Mandatory |
| C-02 | Discord Developer Policy | Bot must comply with the API constraints and policy of Discord's Developer Policy | Discord Rules and Regulations | Mandatory |
| C-03 | Discord Privacy Policy | Bot must not obtain sensitive user data without user consent as per the Discord Privacy Policy | Discord Rules and Regulations | Mandatory |
| C-04 | Discord Terms of Service | Bot must not violate the general Discord Terms of Services | Discord Rules and Regulations | Mandatory |
| C-05 | Discord Store Distribution Agreement for Developers | Bot must comply with the Discord Store Distribution Agreement for Developers | Discord Rules and Regulations | Mandatory |

## Security Requirements

The requirements in this section state the security needs for the project.

**Table 12: Security Requirements**

| NUMBER | NAME | DESCRIPTION | SUBCATEGORY | DRIVER |
|---|---|---|---|---|
| S-01 | Hide Token Data | Ensure critical token data is stored in .env files away from the code | Logical | Mandatory |
| S-02 | Granted Admin Role | Bot must be granted "Administration" privileges on Discord in order to function | Logical | Mandatory |

## Requirements Traceability Matrix

| ID | NAME | DESCRIPTION | TEST CASE ID | SCENARIO | EXPECTED RESULT |
|---|---|---|---|---|---|
| B-01 | Create reaction collection post | Create the post to collect reactions using "!create! | | | |
| B-02 | Fetch reaction list from collection post | Gather users who selected the reactions when "!post" is triggered | | | |
| B-03 | Remove redundant reactions | Remove reactions of deleted users upon "!post" trigger | | | |
| B-04 | Randomize fetched list | Randomize the list of users upon "!post" trigger | | | |
| B-05 | Pair users | List the users in groups of two | | | |
| B-06 | Post pair results | Send an embed message back to the client with the pair results | | | |
| T-01 | Physical host | Running machine to host bot | | | |
| T-02 | User Interface | Create a UI that allows for user interaction | | | |
| T-03 | Input/Output – Creating Reaction Post | Ensure bot is triggered by the correct input and generates the correct output | | | |
| T-04 | Input/Output – Creating Pairing Post | Ensure bot is triggered by the correct input and generates the correct output | | | |
| T-05 | Name Format - Mention | Users in the pairing posts must be tagged properly using the @ notation | | | |
| C-01 | Discord Community Guidelines | Bot must comply with and not promote activity against Discord's Community Guidelines | | | |
| C-02 | Discord Developer Policy | Bot must comply with the API constraints and policy of Discord's Developer Policy | | | |
| C-03 | Discord Privacy Policy | Bot must not obtain sensitive user data without user consent as per the Discord Privacy Policy | | | |
| C-04 | Discord Terms of Service | Bot must not violate the general Discord Terms of Services | | | |
| C-05 | Discord Store Distribution Agreement for Developers | Bot must comply with the Discord Store Distribution Agreement for Developers | | | |
| S-01 | Hide Token Data | Ensure critical token data is stored in .env files away from the code | | | |

| | | | | | |
|---|---|---|---|---|---|
| S-02 | Granted Admin Role | Bot must be granted “Administration” privileges on Discord in order to function | | | |
| UC-01 | Create reaction host post | Bot posts embed message as a response to the command that accepts user reactions | | | |
| UC-02 | Pair users/post pair message | Bot posts embed message as a response to the command that pairs users who reacted in groups of two | | | |
| UC-03 | Delete user reactions no longer on server | Delete users who are unable to be paired (due to no longer being a member of the server) before pairing/posting. | | | |

# Feasibility

## Introduction

The purpose of this section is to evaluate the feasibility of the proposed solution of automating the Discord user pairing process using a bot.

The original goal for this project is to be a fully functional bot that saves pairing data in a schema, runs an algorithm alongside of the pairings against the data saved to pair new people who reacted to a post every week.

During the evaluation process, the team used the criteria listed below.

Evaluation Criteria

- Is the solution technologically feasible?
- Will the solution solve the problem presented?
- Can the client afford the solution?
- Does this solution fit within the given timeframe?
- Is this solution within reach of the development team given their skill/experience level?

It was important to factor in the fact that the development team has other professional and academic commitments alongside of this project. This project is also being used as an educational pillar for the developer(s) as they are writing the logic in a new programming language, which implies that there will be a learning curve for the team.

## Most Feasible Solution

Create an MVP which accepts user commands, pairs users who are a part of a reaction list on a specific message (fetched via ID).

Due to time constraints and the experience level of the development team, implementing the above solution is not feasible for this assignment and a bare minimum MVP has been selected as the scope to ensure the project can be completed within the allotted time frame while ensuring stable functionality.

Economic: Feasible. This will cost less than maintaining a DBMS and server host.

Structural: Feasible. The development team has all tools necessary to build this solution and the Discord APIs allow for the functions proposed to be written.

Operational: Feasible. Middle ground solution between what the client wants long-term and what the development team can offer within the constraints. Pairing functionality is still in tact.

## Alternative Solutions

### Solution 1

Do nothing; continue to pair users manually “as-is” instead of using the bot infrastructure

Economic: Solution is viable to save short term costs, but may lead to a larger pay in manual work down the road.

Structural: N/A

Operational: Solution will not solve the problem of automating the paring, Admins will still need to manually pair users.

### Solution 2

Create logic to assist with pairing outside of Discord (such as a list randomizer web application), but continue to post the pairings manually.

Economic:

Structural: Solution would be feasible, but wouldn’t utilize Discord and their APIs

Operational: Solution is feasible but redundant. A lot of work would be put into creating an application that would require the user to still do manual work afterwards.

### Solution 3

Attempt to create the bot with all of the features originally intended, including the schema and server hosting for high availability.

Economic: Cost will go up, feasibility depends on the budget of the client.

Structural: The use of a DBMS would be required, along with the bot and Discord server/client to save the past pairings.

Operational: The most feasible for the client, as this would eliminate duplicate pairings. The least feasible for the development team as the amount of work to successfully build this solution is outside the scope/timeframe.

## Conclusion

While the team may face their respective challenges, Discord and discord.py both offer ample documentation and support to ensure the solution proposed can be achieved.

The solution chosen will allow the development team time to learn the tools they will be using while still producing a valuable product for the client by the deadline

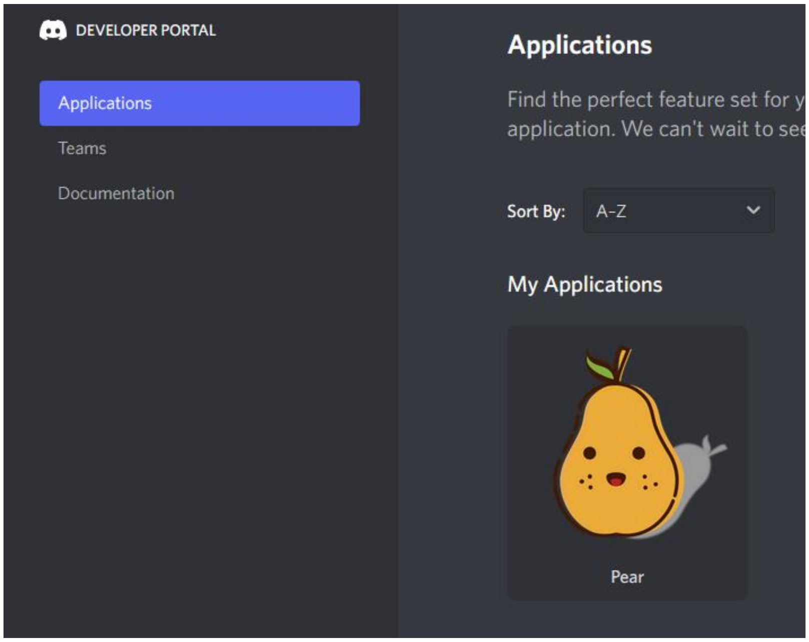
# Design

## Introduction

This section includes information on the high-level system design. Most of the application will depend on Discord REST and Gateway APIs, which serve as the bridge for the bot to communicate with the Discord client, server(s) and database(s).

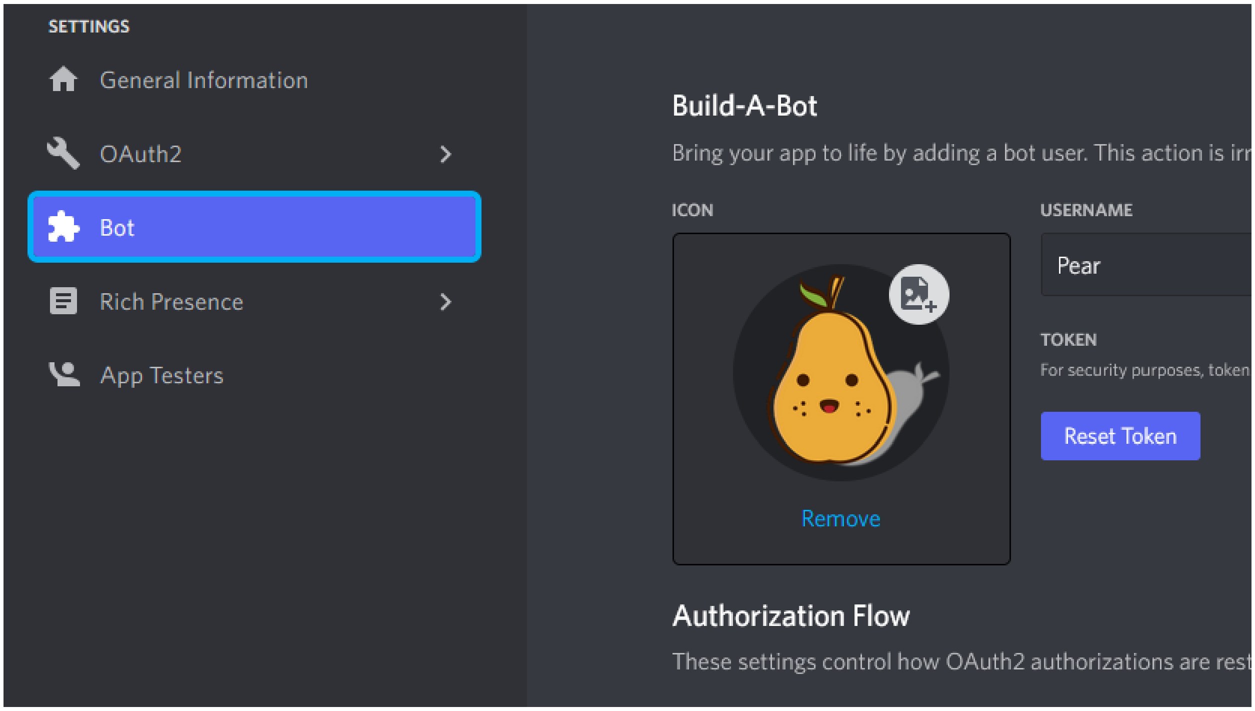
## Creating the Bot Account

Discord requires all bots to be created as an "Application" using their Developer Portal.



Find the perfect feature set for y
application. We can't wait to see

Under the 'Bot' settings, a token will be generated. This token is required to connect the source code to the Discord bot and is only visible when it is created. Should a token reset be needed, the code must be updated with the new token value.



Build-A-Bot

Bring your app to life by adding a bot user. This action is irr

USERNAME

Pear

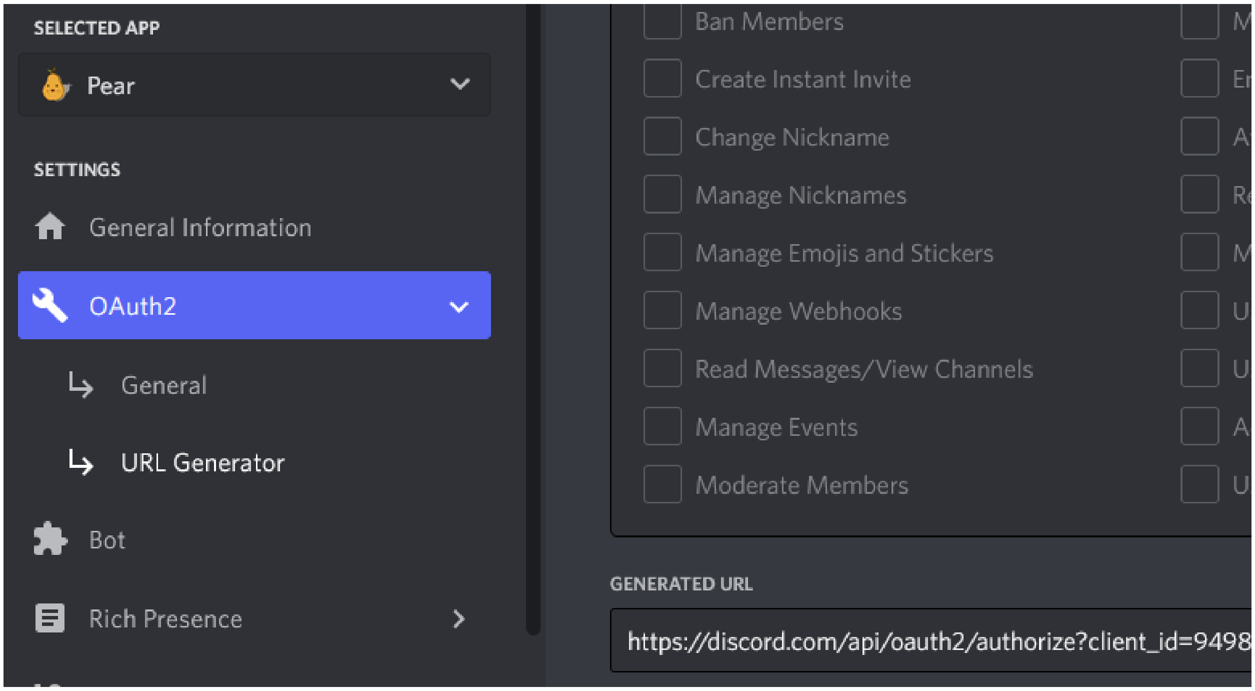
TOKEN

For security purposes, token

Authorization Flow

These settings control how OAuth2 authorizations are rest

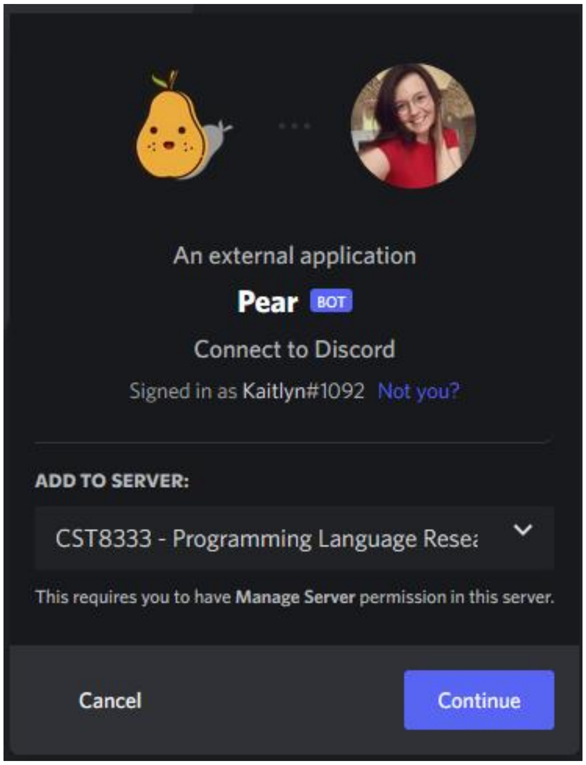
We then utilize the OAuth2 “URL Generator” to create an invite link for the bot that automatically gives it the selected permissions upon joining the selected server.



☐ Ban Members
☐ Create Instant Invite
☐ Change Nickname
☐ Manage Nicknames
☐ Manage Emojis and Stickers
☐ Manage Webhooks
☐ Read Messages/View Channels
☐ Manage Events
☐ Moderate Members

GENERATED URL

https://discord.com/api/oauth2/authorize?client_id=9498



An external application

**Pear** BOT

Connect to Discord

Signed in as Kaitlyn#1092 Not you?

**ADD TO SERVER:**

CST8333 - Programming Language Rese:

This requires you to have **Manage Server** permission in this server.

Cancel

Continue

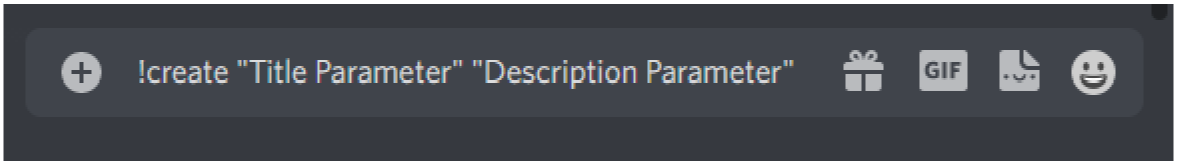
The bot account is then added to the server and the next phase in development can begin.

## User Interfaces

Users are broken into two categories for the purpose of this project.

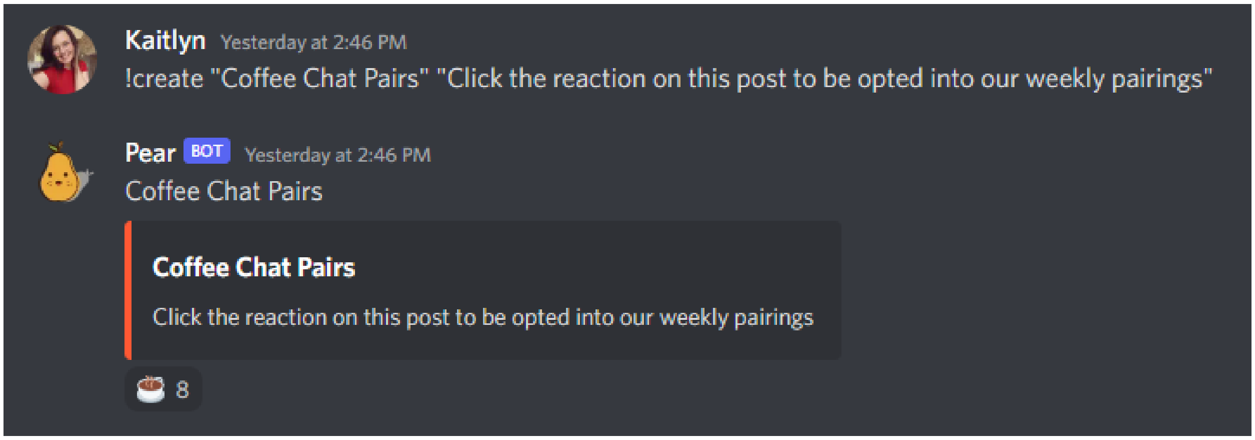
The Admin user will interact with the bot via the Discord desktop, browser, or mobile client after it has been officially added to the server by issuing a command starting with the prefix “!”.

Parameter one determines the title of the reaction host message, while parameter two determines the description, or text to embody in the embed.



```
!create "Title Parameter" "Description Parameter"
```

Users will then click a reaction (shown as the coffee cup in the screenshot below) to be added to a list.



Kaitlyn Yesterday at 2:46 PM

!create "Coffee Chat Pairs" "Click the reaction on this post to be opted into our weekly pairings"

Pear BOT Yesterday at 2:46 PM

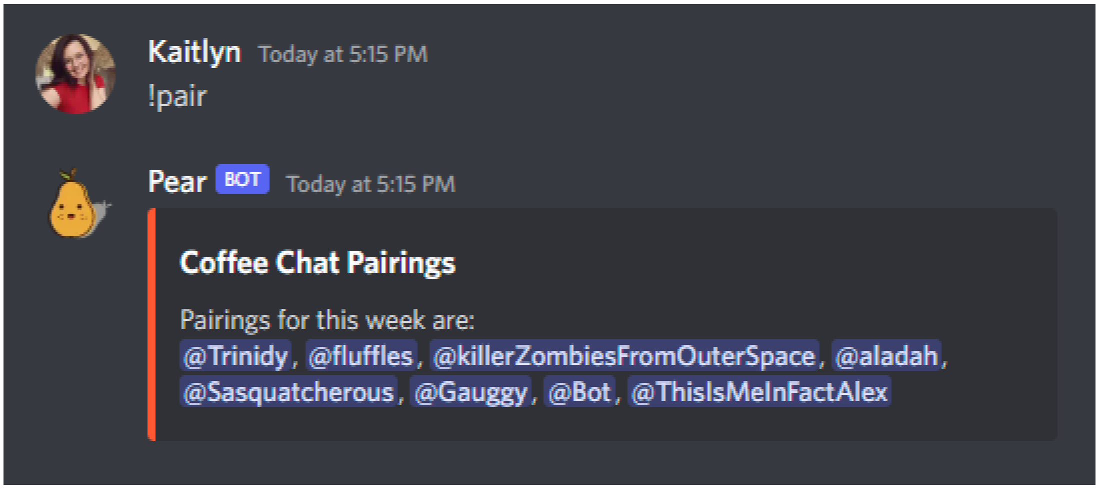
Coffee Chat Pairs

**Coffee Chat Pairs**

Click the reaction on this post to be opted into our weekly pairings

☕ 8

Then, the Admin user will issue the command "!pair" to pair the users that have selected the reaction. Note, this is only a simple prototype and the pairing function is nothing more than a list of users currently. The idea is to pair in groups of two and add function to increase usability and stability.



Kaitlyn Today at 5:15 PM

!pair

Pear BOT Today at 5:15 PM

**Coffee Chat Pairings**

Pairings for this week are:
@Trinidy, @fluffles, @killerZombiesFromOuterSpace, @aladah, @Sasquatcherous, @Gauggy, @Bot, @ThisIsMeInFactAlex

The second type of user is the general user, and they are the users tagged with the "@" symbol in the screenshot above. In Discord, when a user is mentioned using the "@" symbol, it sends a ping directly to the user and notifies them that they have been tagged.

General users will not be able to issue commands to the bot and play a "read only" role with the bot. Their position as stakeholders is to enjoy the benefits of the pairing system itself, however the Admin users decide to utilize the bot.

## Platforms

The bot itself may run anywhere Python can be installed. This includes Apple, Windows and Linux. Users must be able to access the Discord client (which is available on Apple, Windows, Linux, Browser, Android and iOS) to utilize the bot.

## Programming

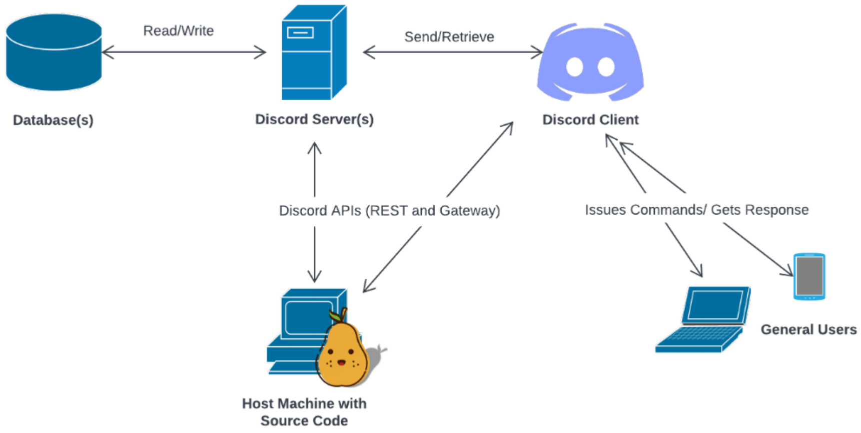
This project has two main functions—one, to create the post to host the reactions from which the bot will pull from, and two, to pair users that reacted to that post and post a message displaying the results.

Using a concept called "cogs" in discord.py will allow us to create a bot utilizing the structure of OOP/ multiple classes instead of creating the bot in one flat bot.py file. This structure will be used to separate the two functions to maintain clean code and better readability.

## System Architecture

This project will be written in Python, using the discord.py library in PyCharm IDE. I have referred to a couple tutorials on how to create a Discord bot in Python and will include them in the references section.

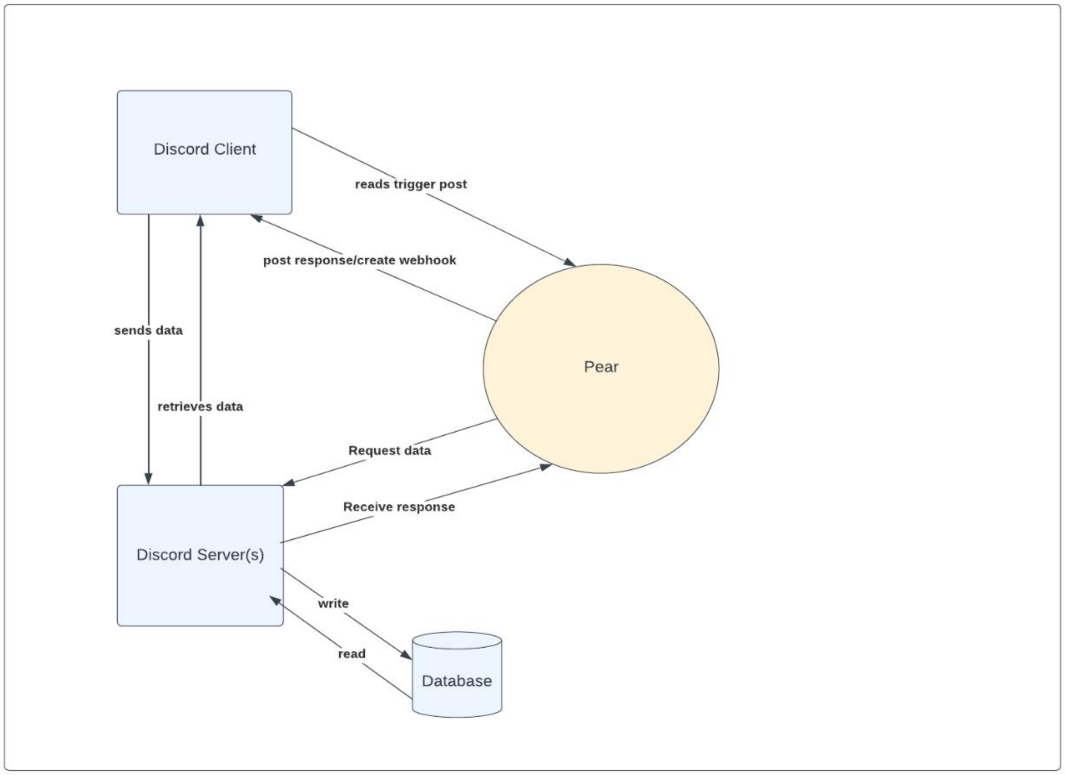
## Network Architecture


Read/Write
Send/Retrieve
Database(s)
Discord Server(s)
Discord Client
Discord APIs (REST and Gateway)
Issues Commands/ Gets Response
General Users
Host Machine with Source Code

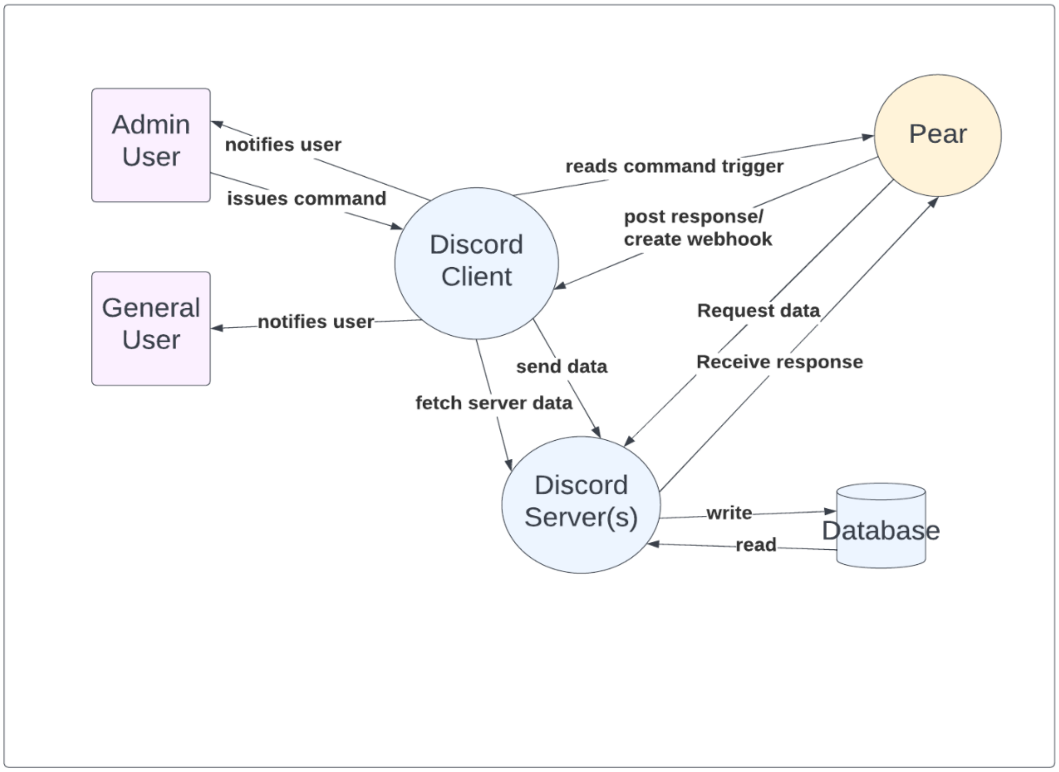
## Process (data flow)

Below are two data flow diagrams (DFDs) representing the high-level logic of data flow for this project.

DFD Level 0



DFD Level 1



### Security

The bot will utilize the built-in security features of Discord’s APIs, OAuth2 and the Discord platform “role” based privileges. All token data will be saved in external .env files to ensure this data is not in plain sight in the primary Python file.

### Availability

The project will have relatively low availability this iteration. It will only be available to users while running on a host machine. The availability will be increased to near maximum should the bot be placed on an external web host, where it can run 24/7.

### Database Schemas

No external schemas will be created for the scope of this project.  All data retrieved will be done via Discord APIs from Discord’s official database(s).

## Test Plan (in process)

This section will house the test plan and test cases, along with their desired and actual results.

### Test Cases

| ID | Artifact | Scenario | Expected result | Result | Automated | Hardware |
|---|---|---|---|---|---|---|
| | | | | | | |

# References

Algonquin College. (2020a). *CST8333 Budget Template*. Algonquin College.

Algonquin College. (2020b). *CST8333 Timeline Template*. Algonquin College.

Algonquin College. (2020c). *CST8333 Assignment 1 Outline*. Algonquin College.

Chi, E. (2020, August 10). *How to Make Discord Bot Commands in Python*. Better Programming; Medium. https://betterprogramming.pub/how-to-make-discord-bot-commands-in-python-2cae39cbfd55

Discord. (n.d.-a). *Discord Developer Portal — API Docs for Bots and Developers*. Discord Developer Portal. Retrieved June 8, 2022, from https://discord.com/developers/docs/intro

Discord. (n.d.-b). *Discord Store Distribution Agreement for Developers (Self-Service)*. Discord Developer Portal; Discord. Retrieved May 31, 2022, from https://discord.com/developers/docs/policies-and-agreements/store-distribution-agreement

Discord. (n.d.-c). *OAuth2*. Discord Developer Portal; Discord. Retrieved June 8, 2022, from https://discord.com/developers/docs/topics/oauth2

Discord. (n.d.-d). *Permissions*. Discord Developer Portal; Discord. Retrieved June 9, 2022, from https://discord.com/developers/docs/topics/permissions

Discord. (2020, July 1). *Discord Developer Policy*. Discord Developer Portal; Discord. https://discord.com/developers/docs/policies-and-agreements/developer-policy

Discord. (2022a, February 25). *Community Guidelines*. Discord. https://discord.com/guidelines

Discord. (2022b, February 25). *Privacy Policy*. Discord. https://discord.com/privacy

Discord. (2022c, February 25). *Terms of Service*. Discord. https://discord.com/terms

Duke, R. (2022). *What is a Work Breakdown Structure*. Workbreakdownstructure.com. https://www.workbreakdownstructure.com/

MiiaHash. (2021, May 3). Microservices architecture (discord bot). *MCMARKET*. https://www.mc-market.org/threads/662952/#post-4826533

Pngtree. (2021). meb style yellow cute pear clipart [Online]. In 阿Go (Ed.), *Pngtree*. https://pngtree.com/freepng/meb-style-yellow-cute-pear-clipart_5867760.html

Rapptz, D. (n.d.). *Cogs*. Discord.py. Retrieved June 9, 2022, from https://discordpy.readthedocs.io/en/stable/ext/commands/cogs.html

Rapptz, D. (2019, September 30). *Welcome to discord.py*. Discord.py. https://discordpy.readthedocs.io/en/stable/

Seewald, D. (2022, April 11). *Send an Embed with a Discord Bot in Python*. Python in Plain English; Medium. https://python.plainenglish.io/send-an-embed-with-a-discord-bot-in-python-61d34c711046

Taneja, N. (2021, August 27). *Architecting discord bot the right way*. DEV Community. https://dev.to/itsnikhil/architecting-discord-bot-the-right-way-383e

Warren, T. (2021, November 17). *Discord is quietly building an app empire of bots*. The Verge. https://www.theverge.com/2021/11/17/22787018/discord-bots-app-discovery-platform

## Appendix A: Module 2 Checklist

Please use the checklist, below, to track your completion of assignment requirements. Please also note that it is possible that not all the items in the checklist will be applicable to your project. With that said, if you answer "No" in the Completed for an item, then please add a brief explanatory note.

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Define/refine the requirements (Planning). | [✓ **Yes**]/No | |
| Conduct project planning. | [✓ **Yes**]/No | |
| Initiate controlling processes. | [✓ **Yes**]/No | |

**Project Approach**

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Review project approach and validate that the modified waterfall project lifecycle, identify the phases of the project, and define project measures of success. | [✓ **Yes**]/No | |
| Initiate engagement with dependent areas. This includes operations, training, support services, data, and documentation management. | [✓ **Yes**]/No | |
| Initiate development of a work breakdown structure. | [✓ **Yes**]/No | |
| Identify Configuration Management Plan, requirements management process, work products, change control, version control, requirements with status tracking. | [✓ **Yes**]/No | |
| Review and revise business objectives, scope, and vision from Concept phase, document the "As-Is" and "To-Be" business processes and any process improvement opportunities. | [✓ **Yes**]/No | |

**User Requirements Definition**

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Identify the project requirements, and decision process. | [✓ **Yes**]/No | |
| Define approach for requirements definition, documenting, communicating requirements and identify log tool. | [✓ **Yes**]/No | |
| Develop and prioritize use cases for portions of the software application to be developed. | [✓ **Yes**]/No | |
| Further define complex use cases (if required) with additional techniques such as data flow diagrams, entity relationship diagrams, state- | [✓ **Yes**]/No | |

| | | |
|---|---|---|
| transition diagrams, and object class and interaction diagrams. | | |

**Technical Requirements Definition**

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Define data requirements. Create data models to document, as needed. | Yes/No | No database schemas will be created on our end for this project. Only data from Discord's official platforms will be utilized. |
| Identify, define, and document technology, infrastructure, and support requirements. Conduct/refine enterprise and data architectural reviews. | [✓ **Yes**]/No | |
| Define the user documentation deliverable types and the training deliverables needed for the application/system. | Yes/No | |
| Elaborate the use cases into the necessary functional requirements. | [✓ **Yes**]/No | |
| Develop user interface prototype. | [✓ **Yes**]/No | |
| Develop and evaluate application/system input and output requirements, screens, menus, reports, and forms. | [✓ **Yes**]/No | |

**Requirements Documentation and Review**

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Prioritize features and/or functional requirements. | [✓ **Yes**]/No | |
| Develop Software Requirements Specification. | Yes/No | This project plan serves as the SRS for the scope of this project |
| Plan and conduct Requirements Walkthrough. Gather feedback and approval for the Software Requirements Specification. | [✓ **Yes**]/No | |
| Plan and conduct User Requirements Walkthrough. Gather feedback and approval for the Software Requirements Specification. | [✓ **Yes**]/No | |

**Test Planning**

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Begin development of Test Plan. Evaluate test risks to determine an appropriate test strategy. Include a high-level test schedule. | Yes/No | Category created – failed to implement this by the deadline. |
| Develop conceptual test cases. | Yes/No | - |

| | | |
|---|---|---|
| Develop initial draft of User Acceptance Criteria. | Yes/No | - |
| Use the test cases to verify use cases, technical requirements, and prototypes. | Yes/No | - |

**Requirements Refinement**

| Activity | Completed | If “No” Please Explain |
|---|---|---|
| Repeat requirements definition steps, as required. Refine SRS [Software Requirement Analysis], if required. | **[✓ Yes/No]** | This is an ongoing process that will be done beyond the scope of Assignment 1 |
| Define/revise the System diagram. | Yes/No | |

**Project Planning**

| Activity | Completed | If “No” Please Explain |
|---|---|---|
| Refine/validate project approach and project lifecycle | **[✓ Yes]**/No | |
| Begin development of the Project Notebook and management approaches | Yes/No | I felt like this was redundant with all of the other |
| Document the project organization. | Yes/No | |
| Establish Communication strategy. | Yes/No | I am the only one working on this project therefore no official strategy plan was needed. |
| Review the project schedule, budget and SRS, sponsors, and other appropriate stakeholders. | **[✓ Yes]**/No | |
| Prepare Project Funding deliverables and review with the funding approval authority. | **[✓ Yes]**/No | |
| Plan for next phase. | **[✓ Yes]**/No | |

**Planning/Funding Approval**

| Activity | Completed | If “No” Please Explain |
|---|---|---|
| Baseline requirements, schedule, and budget and place all configurable documents (e.g., SRS, Project Plan, etc.) in the project repository. | **[✓ Yes]**/No | All but the SRS |
| Document lessons learned. | **[✓ Yes]**/No | |

## Appendix B: Module 3 Checklist

Please use the checklist, below, to track your completion of assignment requirements. Please also note that it is possible that not all the items in the checklist will be applicable to your project. With that said, if you answer "No" in the Completed for an item, then please add a brief explanatory note.

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| A description of the problem to be solved. | [✓ **Yes**]/No | |
| A Project Evaluation Criteria. | [✓ **Yes**]/No | |
| A project briefs. | [✓ **Yes**]/No | |
| An analysis of potential project sites | Yes/No | Not applicable for this project |
| Is this Project Plan technically feasible? | [✓ **Yes**]/No | |
| Analysis of the development options including the "Do Nothing" option. | [✓ **Yes**]/No | |
| Cost/benefit and life-cycle cost information for major decisions. | [✓ **Yes**]/No | |
| Set of conceptual or preliminary drawings (prior to schematic design phase). | [✓ **Yes**]/No | Included preliminary screenshots instead |
| Is this Project Plan operationally Feasible? | [✓ **Yes**]/No | |
| A professionally prepared project budget. | [✓ **Yes**]/No | |
| Recommendation of the project management process to be used. | [✓ **Yes**]/No<br>** labelled 'Most Feasible Solution' | |
| A discussion of scheduling constraints and impact on program and budget. | [✓ **Yes**]/No | |
| Document lessons learned | [✓ **Yes**]/No | |

## Appendix C: Module 4 Checklist

Please use the checklist, below, to track your completion of assignment requirements. Please also note that it is possible that not all the items in the checklist will be applicable to your project. With that said, if you answer "No" in the Completed for an item, then please add a brief explanatory note.

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Initiate Design Activities. | [✓ **Yes**]/No | |
| Determine/Review Project Plan | [✓ **Yes**]/No | |
| Implement Control processes and activities. | [✓ **Yes**]/No | |
| Create mechanisms for tracking, managing, and changing design specs. (Configuration Management). | Yes/No | Redundant for this project, |

| | | |
|---|---|---|
| Define/Refine project roles and responsibilities. | Yes/No | I am working alone on this project, therefore I fulfil all of the roles. |

### High-Level Design

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Establish a design strategy. | [✓ Yes]/No | |
| Create a high-level design based on requirements and identify impact risks. | [✓ Yes]/No | |
| Decompose the system into subsystems and outline the communications. | [✓ Yes]/No | |
| Create Prototype with inputs and outputs. | [✓ Yes]/No | |
| Create logical data models and design data model. | Yes/No | No schemas are created for this project |
| Identify DBMS platform, hardware, operating system. | Yes/No | - |
| Conduct high level architecture design review. | [✓ Yes]/No | |
| Baseline High Level Design Document. | [✓ Yes]/No | |

### Detailed Design

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Create physical databases. | Yes/No | No database for this assignment |
| Create Detailed Design, design system inputs and outputs, user interfaces, and dialogues. | [✓ Yes]/No | |
| Update traceability matrix. | [✓ Yes/]No | |

### Test and Documentation Planning

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Continue development of test plan. | Yes/No | Still ongoing |
| Continue development of documentation approach. | [✓ Yes]/No | |

### Design Approval

| Activity | Completed | If "No" Please Explain |
|---|---|---|
| Obtain design approval. | Yes/No | Obtained project approval via email, but not feedback on the design. |

### Design Phase Clauses

| Activity | Completed | If “No” Please Explain |
|---|---|---|
| Begin development of the Deployment Strategy and Plan, focusing on the Strategy section. | Yes/No | Started, but not completed |
| Begin development of the System/Application Support Document to be used for long-term support issues. | [✓ Yes]/No | |
| Revise Project Schedule and Project Budget as required, and plan for the next phase. | [✓ Yes]/No | Still ongoing |
| Document Lessons Learned. | [✓ Yes]/No | |

## Appendix D: Assignment Grading Rubric

Please find, below, the criteria against which *Assignment 1* will be graded.

| Criteria | Excellent<br>80-100% | Good<br>50-79% | Requires Improvement<br><50% | Points |
|---|---|---|---|---|
| **Assignment Quality** | All information offered is accurate<br>All views are clearly expressed and well explained<br>Contains original ideas, connections, or applications | Most information offered is accurate<br>Most views are clearly expressed and explained<br>Contains mainly original ideas, connections, or applications | Some or no accurate information offered<br>Views are rarely or never clear and require further explanation<br>Many non- original ideas, or unclear connections or applications | **/10** |
| **Assignment Knowledge and Skills Demonstration** | Clear, concise synthesis of course content to demonstrate understanding of topic<br>All ideas are clearly developed, organized logically, and connected with effective transitions<br>Explores ideas, supports points fully using a balance of evidence, and uses effective reasoning to | Evidence of some synthesis of course content to demonstrate understanding of topic<br>Some unified and coherent ideas are developed with effective transitions<br>Supports most ideas with effective examples, and/or references, and details, makes key distinctions | Lack of evidence or weakness in the synthesis of course content to demonstrate understanding of topic<br>Develops and organizes ideas that are not necessarily connected.  Some ideas seem illogical and/or unrelated<br>Presents ideas in general terms; most ideas are inconsistent/unsuppo | **/10** |

| **Criteria** | **Excellent<br>80-100%** | **Good<br>50-79%** | **Requires Improvement<br><50%** | **Points** |
|---|---|---|---|---|
| | make useful distinctions<br>All relevant course and topic links are made | Most relevant course and topic links are made | rted, and reasoning is flawed or unclear<br>Some or no relevant course and topic links are made | |
| **Assignment Structure** | Formatted as per assignment details<br>Structure and format enhance delivery of the information<br>Clear language is used which leads to easy readability<br>Correct grammar and spelling are consistently used | Formatted as per assignment details in most components<br>Structure and format fits well with the delivery of the information<br>Mostly clear language is used with minor readability issues<br>Few or no spelling and/or grammatical errors | Formatting has not been followed<br>Structure and format are unclear and impedes delivery of the information<br>Language used is often unclear which impedes readability<br>Many spelling and grammatical errors | **/5** |
| **Total Points** | | | | **/25** |