**PROJECT REPORT**

**TU Bot, A Discord Bot for Professors & Students**

Table of Contents

[Project Abstract 3](#_Toc440365550)

[High Level Requirement 3](#_Toc440365551)

[Conceptual Design 3](#_Toc440365552)

[Background 3](#_Toc440365553)

[Required Resources 4](#_Toc440365554)

## Project Abstract

This project proposes a practical implementation of a multi-purpose Discord bot with the intended use for professors and students to effectively communicate and use meaningful tools in the Discord application. With the hardest parts of quarantine behind us, many students and professors have created Discord accounts and Discord servers to communicate effectively with each other. In the online semesters, many students have picked up the platform and now use it in their everyday life for not just school – as Discord has created a social platform for students to create friendships and properly communicate and collaborate on schoolwork. TU Bot is a concept that strives to be used in all Temple University affiliated servers, with a large scope of helpful tools and commands that would be used for professors and students so that they can make the Discord platform even more convenient for them. Meaningful tools include things like appointment schedulers, academic reminders, categorized/organized notifications, group creators, collaboration toolkits etc. If development time allows it, the bot will strive to web-scrape Canvas for even more uses, perhaps with real-time notifications from Canvas.

## High Level Requirement

This bot requires the user to install the Discord application, create a Discord account, and create a Discord server. The user may use an already existing Discord account or Discord server.

Discord specifically requires Windows 7+, or MacOS 10.11+ on computer devices. On mobile devices, Discord specifically requires iOS 10.0+, or Android 5+. In terms of web browsers, Discord can run on Google Chrome, Firefox 80+, Opera, Microsoft Edge 17+, Safari 11+. A working, stable internet connection is required to utilize all of Discord’s features.

Given an appropriate URL link, any user can add the TU Bot into their Discord server. The user would have to access the URL link and invite the bot to their server. This will give the bot its required permissions to carry out its capabilities without issue. When TU Bot has joined a server, it will host a variety of helpful commands that the user can access with ‘/’ commands in the chat box.

## Conceptual Design

The bot will have a help menu that can be accessed when prompted by a user in the chat box. From there the user can see a full list of commands that can be used by inputting a ‘/’ as an indicator that the bot is taking in a command. These commands will serve many possibilities to make the lives of the student, professor and TA more convenient. This bot is meant to be expanded upon – the list of commands can be added onto as long as implementation allows.

TU Bot, like other Discord bots, are hosted on a user-provided entity. A spare device may have to be used to host the bot 24/7, or perhaps a reliable cloud service. Discord.js is utilized as a foundation for the bot. The entire bot is built using JavaScript. A majority of Discord bots are built on Discord.js, or another official Discord framework. TU Bot will also be built on Discord.js, but it is not open source as it only uses the bare-bones foundation of Discord.js to branch out into its own idea like all other Discord bots.

## Proof of Concept

<https://github.com/cis3296s22/concept-athankim>

## Background

In Discord, bots are the second type of “user” that act as participants in a Discord server. They have become extremely popular, both for the general public who use Discord, and developers who use Discord. Discord bots have the capabilities of responding to user inquiries, web-scraping for useful information, staying up to date in real time with data, etc. The list goes on.

<https://hydra.bot/>

The URL above is the official website for one of the most popular Discord bots, which is used to play music for participating users in a voice chat channel. Although this may seem like a simple feature, the bot features a myriad of helpful commands that accommodate a music player.

<https://gamblingbot.app/home>

The URL above is the official website for another popular Discord bot, which is used to play fun casino minigames with other members of your server, or with yourself. It comes with a sleek and user-friendly UI, along with helpful and easy to understand commands.

The TU Bot does not seek to become a music bot, or a gambling bot. It seeks to become a universally useful tool for any professor or student who hopes to use the Discord application for academic needs at Temple University.

## Required Resources

From the start, anything I need has already been included in the GitHub repository with the importation of Discord.js. In the future, web-scraping may need to be used in the bot to implement helpful features. All of this is possible within the scope of JavaScript.

As for development, all development will be done on the developer’s personal computers. In the future, for stability and longevity, a separate hosting device or cloud server may need to be used/purchased to keep the bot online 24/7.

**Project Design**

**Vision:**

For Temple University's students and staff who need better Discord functionality and workflow in their classrooms, TU Bot is a Discord bot that provides academic tools in the popular application Discord, so that students and staff can take advantage of the application's already useful features and integrations. Unlike other Discord bots that are usually used for entertainment or entirely different purposes, TU Bot seeks to establish an academically rich environment in Discord servers that are created by Temple University students and staff. Our product will automate and ease the use of Discord for the academic environment so that professors can communicate effectively with their students, receive feedback, etc. and so that students can communicate effectively with each other, and be more efficient in their curricular tasks.

**Personas:**

*Professor:* Jane Doe is a 47 year old Professor at Temple University. She teaches multiple online Yoga courses at the University. Teaching multiple sections means she has many students to keep track of. Additionally, Jane is not very tech savvy and often struggles maintaining her online courses. Receiving messages from both students and faculty via email and canvas causes confusion for Jane, and she may miss important messages because of it. She is not opposed to doing class online, she just needs an easier way to keep track of her communications for her courses.

Jane Doe is a frequent user of social media and enjoys using Direct Messaging. She likes the idea of using one central application to maintain her communications with her students, TA’s, and other professors. She is interested in using the TU Discord Bot as it would give her easy access to all her communications related to Temple University. She also likes the idea of being able to directly message groups of students at once in order to send announcements to a certain class section.

*Student:* Ryan is a senior at Temple University. He takes 5 classes during a semester and when he is not focused on schoolwork, he works part-time. Because he is so busy, he has trouble keeping track of his assignments and appointments. However, he frequently uses Discord, and would find it much easier to stay on top of his work if he could get notified of assignment due dates from Discord. TU Bot would be very convenient for him as he would never lose track of his assignments. He would be notified on Discord of any upcoming assignments/appointments, and since he is a frequent Discord user, he would be more likely to see these notifications than the ones on canvas or his school email.

*TA:* Jill is a 24 year old, CIS, graduate student here at Temple University. She is a very diligent student and always finishes her work several days before the deadline. Jill cares gravely about making her resume as best as it can be, therefore, she became a TA to further develop herself in her field. However, as a computer science student, she knows the challenges of working in groups, and as a TA, she finds it challenging to communicate with students.

To Jill, communication is key when working in a group for research and projects. As a graduate student, she understands the importance of reaching out for help in a timely manner. TU-Bot allows Jill to easily and quickly communicate with the students she is assisting. She can send students announcements and save resources for her students to use. When students work in groups for projects, Jill uses TUbot to keep students organized in their own separate teams and ensure their progress.

*Student Group Leader:* Matthew, age 19, is an English major sophomore at Temple University. Born and raised in Lancaster Pennsylvania before enrolling in Temple to live on campus. Back in Lancaster Matthew worked at the local library where they became familiar with the technology to manage a library. His father works for a large technology company and from which he had access to computers his whole life. Over the years Matthew has a handful of experience with technology and computers.

From drawing the short straw Matthew has become the leader of a semester-long group project that counts for the majority of everyone's grade. On top of that, he is drowning from his homework with other classes and is finding it hard to keep up with both. From Matthew’s experience in technology, he recognizes that it can make things easier to organize. He is particularly interested in TU Bot for team organization and structure for this project.

*Admin Level Person:* John Smith is a 50 year old professor at Temple University and a high-up official in the CIS department. Every week, John creates a newsletter describing current events in the CIS field, job/internship opportunities, etc. He wishes that more students participated in these newsletters, though, because they are highly valuable resources. Some of the resources are time-sensitive - so even if people read them eventually, they may be out of luck as employers can sometimes only give a one-day window for applicants. John will send separate email blasts for these occasions, but not everyone checks their email every hour, let alone every day, or they just don’t give them attention due to their abundance.

With TU Bot, John is able to take advantage of the extremely popular Discord application to let students know of time-sensitive opportunities, as students are on Discord frequently - usually with notifications on. His routine is creating an announcement himself, and a time gate on it so that TU Bot automatically notifies certain students based on their chosen preferences. And now John is able to see live reactions on his newsletters, as it is extremely easy for students to instantly message him on Discord - where he can have conversations with students in real time. If his Discord inbox is overflowing, he can open up an appointment system on TU Bot so that students can arrange a time to talk with him all inside of the Discord application.

**Backlog of features to be implemented:**

* Appointment scheduler
* Categorized/organized notifications via DMs
* Group creator
* Schedule sync
* Canvas integration?
* Message blaster
* Channel creator for groups
* Role assigner
* Implement / commands

**Diagram

Description automatically generatedUML Diagram**

The following UML diagram represents the development architecture of TU Bot. The ‘main’ part of the program is index.js, which houses the creation and execution of TU Bot. This file implements the intended helpful features of the bot, shown in the bottom box of the index.js node. index.js implements various node modules within the node\_modules folder represented in the diagram. The main component implemented here is discord.js which provides the foundation of the bot and allows index.js to implement commands and access the bot. discord.js is a node library API wrapper for Discord that makes it easier to create a Discord bot in Node.js/JavaScript. An instance of Client is created in the node\_modules compass as well, and Client allows for connection to the Discord application.

**Project Progress**

**Week 2 Progress**

**Sprint Goal:** Implement basic features for the bot and satisfy Scrum Week 2 rubric requirements.

**Backlog Features**

* Yes/No Poll Creator
* Multiple Choice Poll Creator
* Academic Reminders

**Tasks in Sprint Task Status at End of Sprint Assigned To**

Create UML Diagram FINISHED Amira and Erin

Yes/No Poll implementation FINISHED Tommy

Multiple Choice Poll FINISHED Tommy

implementation

Academic Reminders FINISHED Ryan

implementation

Create Week 2 Project Progress FINISHED Amira

Create Executable that executes FINISHED Athan

w/o inputting into terminal

Create New Release Athan