PROJECT DESCRIPTION

BotFly is a project to create a Discord bot using Discord.js. Our bot will provide students and groups with various tools to assist in communication and collaboration. Features will include participation tracking, reminders and alerts and time functions

AlphabetEsq

Casey Edwards | s3776107
Christopher Crawford | s3753150
Dylan Currie | s3793998
Ely Hawkins | s3754457
Melanie Broersen | s3611357

Table of Contents

1 Project Background

1.1 Project Background - Motivation

For some members of the group, the motivation for this particular project came from their desire to further their skills and gain knowledge in the area, particularly JavaScript. This idea allowed those members to work towards their individual goals through realistic hands-on coding experience, while simultaneously assisting in progression towards team goals, aims and intended outcomes.

The motivation behind our main features was the lack of statistical data available on Discord. Many students make use of Discord for collaboration and communication, though there's no way of exporting chat logs, extracting statistics or monitoring participation. Some members found motivation in the idea of using bots to create a solution to a realistic problem.

Our main features are useful to any student with the desire to boost their motivation to achieve study goals. The bot will be especially beneficial for those working on university projects as part of a group, a collective interest of the group.

1.2 Project Background - People (Roles)

Casey Edwards - Senior Software Developer
The senior software developer oversees project work, working with all members to ensure the smooth development and implementation of our Discord.js bot. Responsibilities include assisting all members with the development and implementation of features.

Christopher Crawford - Network Engineer / Software Developer
The network engineer is responsible for assisting in the setup of hosting solution(s) for the purpose of testing our bot over long periods. Our network engineer, like all members, will also assist with the software development of our application.

Dylan Currie - Team Leader / Software Developer
The team leader is responsible for leading the team in a professional manner, overseeing tasks and ensuring outputs are of a desired standard. Responsibilities include managing meetings, motivating and directing team, compiling report submissions and overseeing the completion of assigned coursework tasks. Our team leader, as with all members, will also assist with software development and testing.

Ely Hawkins - Second in Charge / Software Developer
Second in charge is responsible for assisting in the leadership of the team, ensuring all team members conduct work in a professional, collaborative and efficient manner. Responsibilities include assisting in the management of meetings and signing off on completed work ensuring a high standard is met. Second in charge assumes full team leader duties in the absence of our team leader.

Melanie Broersen - Lead Designer & Software Developer

The lead designer is in charge of ensuring all design related tasks are completed to a professional standard. Design related tasks include project demonstrations, presentations, team logos and any end user content. The design leader is responsible for ensuring our application is neat, user-friendly and professional from a design perspective.

1.3 Project Background - People (Blog)

Casey Edwards: Melbourne, VIC, Australia	s3776107@student.rmit.edu.au
Christopher Crawford: Brisbane, QLD, Australia	s3753150@student.rmit.edu.au
Dylan Currie: Port Lincoln, SA, Australia	s3793998@student.rmit.edu.au
Ely Hawkins: Melbourne, VIC, Australia	s3754457@student.rmit.edu.au
Melanie Broersen: Shepparton, VIC, Australia	s3611357@student.rmit.edu.au

1.3.1 Background & Passion in IT

1.3.1.1 Casey Edwards

Back when I was quite young (about 15) I think I really started to get a grasp on my passion for IT related things. This passion came from coding very basic programs in Visual Basic in high school and extended through until after that into doing other things such as modifying video games using scripting and and basic file editing.

While I do not have a specific background in IT professionally my personal passion for IT comes from my past personal experience with coding, and computer hardware components.

This really kicked off when I built my first PC from scratch and I think from then my passion has just remained.

1.3.1.2 Christopher Crawford

My background in IT began three years ago, it was something unexpected as it was not an industry I ever thought of entering. I was studying history at the time whilst working for an ISP in their customer service department. During my time with the ISP, I got well acquainted with the Technical team so began my journey into IT.

From that point, I could not get enough and really threw myself into the industry. Within six months, I had completed the CompTIA Network + and joined the Network Operations team as a junior Network Analyst. I then moved back from the U.K to Australia and moved from Network Support/Administration to Enterprise IT support. Now, I am currently contracting for various Government departments and mainly doing Level 2/3 Desktop/Server support. The reason for undertaking my degree is to better my knowledge in the industry and to reach the goal of System/Network Administrator within the next two years.

My true passion in IT is Networking, an area I want to increase my knowledge. Programming is another element I would like to add as it will be essential to roles, I do in the future.

1.3.1.3 Dylan Currie

I was born in East Gippsland, Victoria, where I lived until age 12. Currently I live on the Eyre Peninsula in South Australia. I have two years of IT work experience, having previously worked as an IT technician for a government agency. During this time, I completed a Certificate III in Information, Digital Media and Technology, along with a Diploma in Leadership and Management. I'm now studying with RMIT as an undergraduate student, completing a Bachelor of Information Technology.

My extensive interest in information technology started at a very young age, having grown up with a father who built computers and created digital games. Throughout my childhood, I was surrounded by family and friends who actively used information technology in unique ways for a variety of different purposes, nurturing my interest in the area. I began coding myself at age 9, starting with basic batch scripts before moving to C/C++ at around 13. In current times, I'm familiar with variety of coding languages; having coded both websites using HTML and small applications using C, C++ and more recently, Java.

1.3.1.4 Ely Hawkins

I am currently studying a Bachelors degree in IT through Open Universities. Since I was young I have always been interested in computers, specifically gaming which has led to an interest and desire to work in the field of programming.

1.3.1.5 Melanie Broersen

I have previously studied certificate III/IV in IT Networking at TAFE, however was unable to finish the course due to unforeseen circumstances. I have had a passion for IT since year 11 VCE when I first took an IT class and realised I was good at something that I also enjoyed. I am currently halfway into my Bachelor of Information Technology.

1.3.2 Strengths & Interests

1.3.2.1 Casey Edwards

I have three main interests when it comes to IT. The first and foremost is my passion for gaming and making video games.
The second is the building and programming of apps and programs that can assist those with disabilities including both mental and physical disabilities.
And lastly I would love to create and manage servers.
The things I am best at though at this time would be my capacity for learning IT related materials such as coding.

1.3.2.2 Christopher Crawford

Networking is certainly my greatest strength and is something I really enjoy. I have extensive experience in hardware, software and operating system support.

As stated before, I really enjoy networking and I have also taken an interest in programming. My interest in programming is mainly aimed at developing my scripting skills to help with my current area of work.

1.3.2.3 Dylan Currie

I have a strong interest in information technology, particularly in the areas of hardware, automation and machine learning. I have some experience coding using a variety of different languages, often creating unique solutions to various coding challenges.

I have some experience relating to management and professional documentation standards, having previously completed a diploma in leadership and management. These skills have proved beneficial in both the planning, documentation, leadership and risk assessment stages of this project.

1.3.2.4 Ely Hawkins

- Picked up programming quickly
- Calm and relaxed under pressure
- Can work individually or in a team
- Interested in programming (specifically game design)

1.3.2.5 Melanie Broersen

I'm a quick learner, organised, and have good attention to detail. I'm interested in learning website coding and design, and also eager to learn how to use many different platforms and programs during my future studies.

1.4 Project Background - Aims & Goals

With our Discord Bot application, we aim to create a valuable automation tool for students who make use of Discord for collaboration. These tools will allow students to monitor participation, set-up deadline reminders and meeting times, and access valuable features encouraging teamwork.

Our aim in this project is to produce a fully functional Discord bot with the intended features, demonstrating our concept in a realistic way.

1.4.1 Research Feasibility

Our first goal is to research the feasibility of our idea. Though early research indicates our idea is feasible, complications with Discord licensing, Discord.js or other unforeseeable issues could render features unfeasible. Our goal is to research into the development of Discord bots, ensuring our team members know about relevant licensing, understand Discord.js and are able to begin development on our application.

1.4.2 Creating a Discord Bot (MVF 1)

Our second goal is creating a basic, barebones Discord bot which successfully connects to servers. This bot won't have any particularly valuable features, though serves as a milestone for our team.

1.4.3 Adding Features - Reminders and Notifiers (MVF 2, 4)

Our next goal is to create a feature which sends reminders at regular intervals. These reminders may relate to an upcoming deadline, scheduled meeting or customised message. This goal will be an achievement for our team and progress towards our overall aim - it will indicate completion of our first main features.

1.4.4 Adding Features - Monitoring Participation (MVF 3, 4, 5)

Our next goal is to add valuable features for automatically monitoring and generating participation statistics on Discord. This tool will monitor messages sent to produce statistics and provide chat logs upon request. These features will ensure our application accurately monitors participation, a core part of our aim.

1.5 Project Background - Scope & Limits

The objective of this project is to produce a functional Discord Bot with the minimal viable features as per designs in *AlphabetEsq-BITS-A2.pdf (**Appendix 1**)*. The software will connect to existing Discord servers, providing users with valuable features for collaboration.

Our project scope includes researching and identifying available technologies, developing and testing. Testing will be done as per testing guidelines in design documentation. No alpha, beta or external testing is planned at this stage.

Development will involve the use of Discord.js to connect with existing Discord servers via the Discord API. Using node.js, our application will run from command line and have no interface, though a user interface may be a future addition. At current times, Discord's existing user interface will be used for interaction with our bot. This will be done using commands in Discord text channels.

Given time restraints, our team will be conducting only limited testing on our application. This testing will involve verifying intended features function as per designs. We will not be making use of third-parties or open-beta testing at this stage, though may consider such things in future projects before any public release.

2 Project Progress - Description

Our project began with two ideas, a discord bot and a 2D computer game. After open group discussions and a majority vote, a discord bot was decided on for the focus of our project. We set out by conducting individual research and working on designs for what would be the key features of our bot.

Once we had core framework in place our team individually began familiarisation with various identified tools and resources, such as Discord.js, Node.js, Visual Studio Code and GitHub. Research showed Visual Studio Code's GitHub integration was a great tool for collaboration, while the program overall is one of the highest ranking amongst JavaScript editors (Heller 2019). Discord.js is one of multiple Discord libraries, though appeared to have the most extensive documentation. Used alongside node.js, Discord.js allows for direct access to the Discord API with minimal coding.

YouTube tutorials as well as written online documentation were shared amongst the group Discord and via Trello (**Appendix 2**), ensuring all team members had access to the same learning materials and resources.

Familiarising ourselves with these resources was our main priority as no team members had prior experience with bot building or JavaScript. Our team identified this as our biggest potential hurdle.

Utilising these tools and resources, we were able to start programming, testing and sharing work on our bot. We made our first GitHub commit on June 28th (AlphabetEsq, 2019a), which was a basic bot with the relevant commands set to respond with a message stating the command is not yet coded.

We initially planned to complete MVFs sequentially (one through five), though as an Agile team, deliverables were completed at varying times as we came across methods or coding techniques which could be utilised in our bot.

Features were completed at varying paces, due to varying coding ability and external commitments of each team member. Our team members all worked together to address this, seeking assistance and input from more competent coders as / when required. This resulted in the completion of MVFs 1, 2 and 5, then later 3 and 4.

During a team meeting, a decision was made to leave extended features until a later project. This decision was made after discussions regarding the work required to complete the various sub features of each MVF. We agreed that our focus should remain on the main features, ensuring the completion of these more important (graded) features.

2.1 MVF1 – Bot Connection

Assuming relevant permissions, bot can be imported and connected to any discord server. Bot shows as 'online' and has a 'bot' tag in the right-hand side user's panel.

Our first minimum viable feature involved creating a bot which would connect to existing Discord servers. This bot serves as the underlying basis for our project.

Now having a rudimentary understanding of Discord.js, node.js and JavaScript, we were able to create a basic Discord bot using Discord.js, a node.js addon which allows access to the Discord API. Following the instructions from a YouTube video (Gaskins 2016) we were able to very quickly create a basic bot that would connect to any discord server, sending a startup message once connected.

This was the first MVF we worked on and completed as it was necessary for the testing of all other minimum viable features.

2.2 MVF2 – Datetime & Nag

Respond to prompts; !dateTime, !nag

The ***!dateTime*** and partially the ***!nag*** command were coded using a combination of techniques used in past courses (such as the 'Date' class) with newly learned techniques and methods unique to Discord.js. This was done very early as it was identified coding these features may be easier than simply mimicking them for the purpose of design demonstrations. The !datetime command was successfully implemented in the first Git commit (Alphabet Esq, 2019a), with a portion of !nag added in the following week (Alphabet Esq, 2019b). This original commit of !nag correctly repeated every three hours, though didn't contain a filter to monitor for only posts by the nagged user. This resulted in the command functioning correctly, though failing to cease when the user posted. Multiple members worked on attempting to resolve this issue, with an ultimate solution implemented (Alphabet Esq, 2019c) after further reading of Discord.js Documentation on message objects (discord.js, 2019).

During the coding and implementation of !nag and !dateTime a decision was made to utilise Discord Commando, opposed to generic Discord.js. Considerations were made to the addition of an automated !help command and extra organisational features available in the Commando bot client. Following this change, commands were grouped into relevant 'command groups' in Visual Studio Code and on GitHub.

Compared to our estimations !dateTime was very simple and quick to code due to JavaScript's 'Date' feature. Date is a built-in class of JavaScript, providing similar functionality to the Java variant. The class provides various date related functions, including generating a string of the current date and time.

The !nag command took a little longer to develop than we had initially planned. Our team experienced issues with filtering chat messages via userID. After extensive research and reviewing of Discord.js documentation, we ultimately found that we needed to filter based on *author.id* for what we wished to achieve; not *user.id*.

2.3 MVF3 – Message Counter(s)

Bot is capable of tracking individual user contributions and actions in the Discord chat, providing three feedback options; !count, !teamMembers, !teamPlayer

A decision was made during the implementation process to cache messages locally, rather than utilising loops to pull large amounts of message data from the Discord API. This design decision was made due to Discord's rate limits and to avoid any possibility of going against the discord terms of service. Discord's API rate limits are in place in order to prevent abuse and overload on their servers and as we were working with a potentially infinite amount of data, we decided against it. This had an impact on how we were able to implement MVF3 and MVF5. Specifically, the !count, !teamMembers and !saveChat prompts. These functions may only check the last 100 messages when the cache is empty, with the cache only storing messages once the bot is turned on. The commands were implemented as per designs, with the addition of a short message indicating how many messages were processed. For example, "*(100 messages checked)"* may appear in the output of a command, indicating the last 100 messages were used.

2.4 MVF4 – Extended team functionality

!bestTime, !setMeeting, !dontFreak

The ***!bestTime*** and ***!dontFreak*** commands have been implemented, though the lead developer of these commands is unavailable to provide specific information. Early testing has shown these commands are functional and work as per designs. Full testing is scheduled for completion before August 25th. The developer did note some issues taking in Dates for the JavaScript Date class, though these issues were ultimately addressed through further learning and research.

The ***!setMeeting*** command has been recoded and implemented successfully. Full validation testing of this command has been undertaken and passed. A previous version wasn't tagging all users and sent the final message at the meeting time, opposed to 5 minutes before. The new recoded version functions successfully tags all users using @*everyone* and sends a final message five minutes prior to meetings, as per design specifications.

2.5 MVF5 – Chatlog Exporting

Bot exports/uploads chat history in comma separated values (.csv) format

Feature 5 allows for exporting of all chats sent on the server, uploading a file containing the messages. We originally planned to develop this feature by pulling all messages from the Discord API, though after reviewing Discord terms and conditions relating to rate limits it was concluded this may be non-compliant. Discord API limits such calls to 100 messages, use of repetition to avoid these limits is discouraged.

We later found a new method of coding this feature - we could instead simply write received messages directly to the file. After some research, it was found previously learned skills relating to Java IO were very similar in JavaScript. Through this, code was implemented to produce a comma separated values (CSV) file which messages were stored in as received.

Changes were later made to convert this into an array of messages accessible by all commands, serving as a message cache (AlphabetEsq Git, 2019d). It was identified this array would be valuable in other features, particularly MVF 3. Following these changes, the ***!saveChat*** command now accesses this array, uploading it as a CSV file.

Our initial idea for this feature was to be able to print the entire chat history within a discord channel. Since the alternative implementation only stores messages while it is turned on, the bot would need to be running 24/7 to cache all messages. In the context of our project, we addressed this issue by hosting the bot locally, allowing it to run 24/7 during testing. In a professional scenario, the bot would be hosted online using a virtual private server or on a microcomputer such as a Raspberry Pi, ensuring constant up time.

Our team considered instead pulling the last 100 messages from the channel and presenting these, though an incomplete chat log wouldn't be as beneficial. The idea of having the bot store messages somewhere on shutdown and reconstruct upon restart was considered, though we decided this was well outside the scope and should be left for a later version of the project.

3 Project Progress - Outcomes to Date

All minimum viable features (MVF) have been designed, coded and implemented. Some features have had minor design changes, though were still successfully implemented. MVF 1, 2, and 5 have undergone full validation testing. MVF 4 and 5 have undergone brief testing, with full testing scheduled for completion by August 25. The current demonstrable outcomes for each MVF are displayed in greater detail below.

3.1 MVF1 – Bot Connection

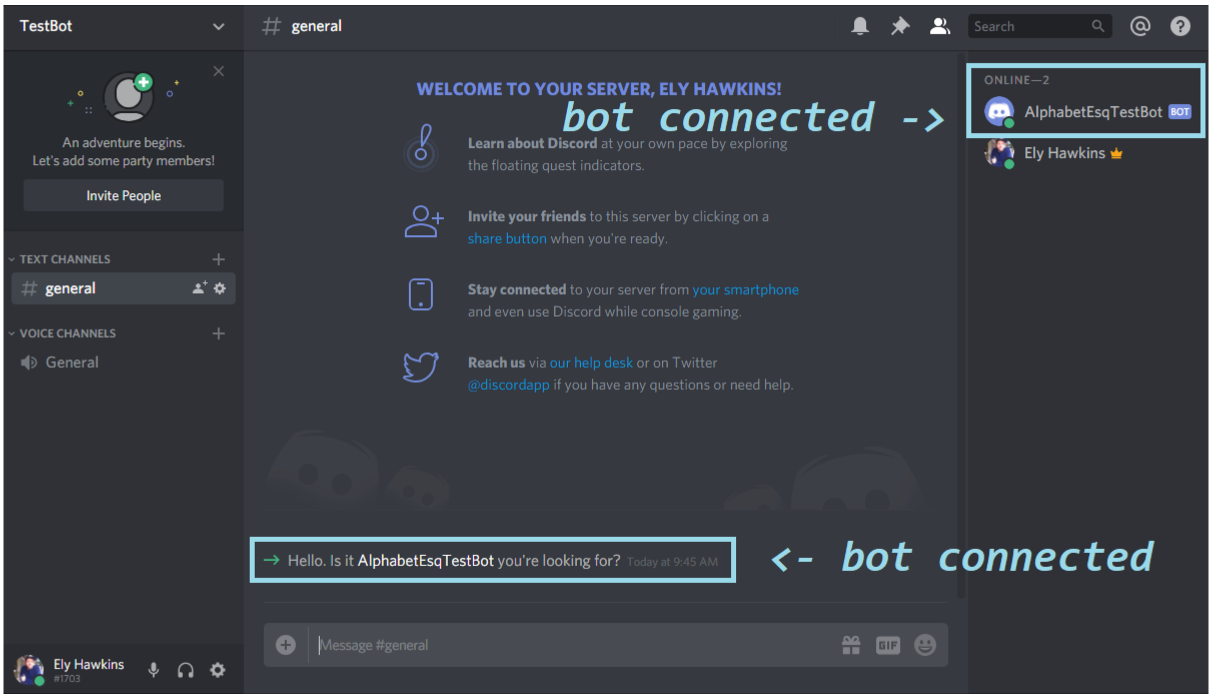
Assuming relevant permissions, bot can be imported and connected to any discord server. Bot shows as 'online' and has a 'bot' tag in the right-hand side users panel.

Implementation of MVF1 was completed and tested successfully. A new bot can be created, started and connected to any existing Discord server that administration rights are held on. The bot shows up as 'online' in the right-side panel and sends an accompanying new user message upon first connection.

This process is repeatable by following a few simple steps:

1. Create a new bot in the Discord Developer Portal, connecting it to any server where administration permissions are held.
2. Select the bot in Discord Developer Portal, click the 'bot' tab, then copy the bot access token.
3. Place the access token in AlphabetEsq bot and running the code.

Following this, the bot will boot up and connect to any servers selected in step 1, showing as 'online' and sending an accompanying connection message. At this point, the bot is connected and ready to respond to commands outlined in MVFs 2-5.


TestBot
general
Search
WELCOME TO YOUR SERVER, ELY HAWKINS!
bot connected ->
ONLINE—2
AlphabetEsqTestBot BOT
Ely Hawkins
An adventure begins.
Let's add some party members!
Invite People
TEXT CHANNELS
general
VOICE CHANNELS
General
Learn about Discord at your own pace by exploring the floating quest indicators.
Invite your friends to this server by clicking on a share button when you're ready.
Stay connected to your server from your smartphone and even use Discord while console gaming.
Reach us via our help desk or on Twitter @discordapp if you have any questions or need help.
<- bot connected
Message #general
Ely Hawkins #1703

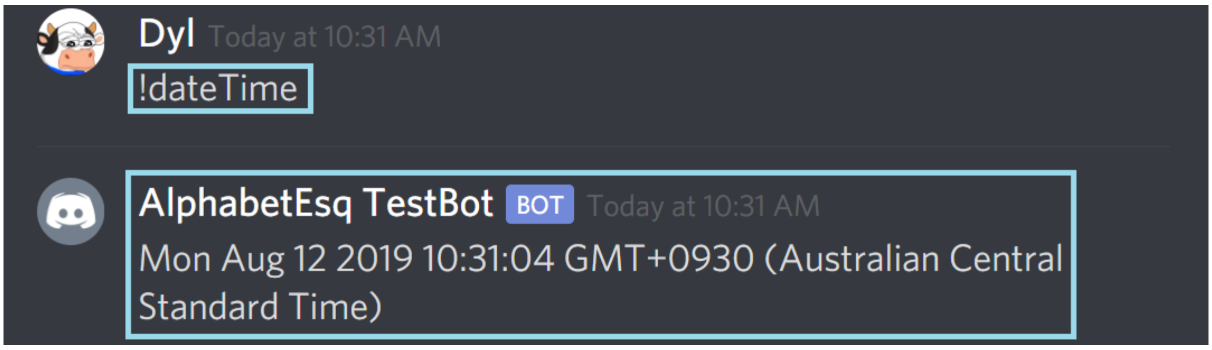
→ Hello. Is it AlphabetEsqTestBot you're looking for? Today at 9:45 AM

***Figure 3.1** - Bot Connection*

3.2 MVF2 – Datetime & Nag

Respond to prompts; !dateTime, !nag

The ***!dateTime*** command has been completed and implemented as per designs. Users of our bot can access this command by simply typing !dateTime, at which point the bot will respond with the current date and time in the bot's local time zone. This command has undergone testing and works as per designs.

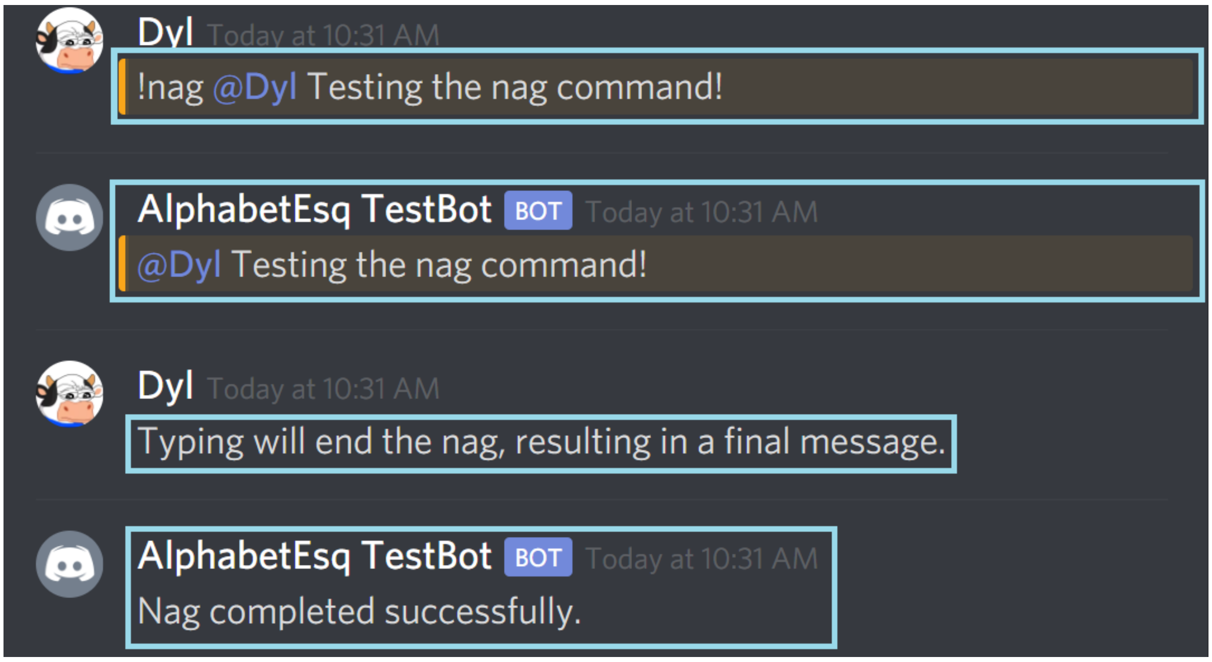


Dyl Today at 10:31 AM
!dateTime

AlphabetEsq TestBot BOT Today at 10:31 AM
Mon Aug 12 2019 10:31:04 GMT+0930 (Australian Central Standard Time)

***Figure 3.2.1** - !dateTime*

The ***!nag*** command has been completed and implemented as per designs, full validation testing has been completed and the command functions as per expectations. Bot successfully tags the mentioned user and sends the accompanying message in intervals until the mentioned user responds.


Dyl Today at 10:31 AM
!nag @Dyl Testing the nag command!

AlphabetEsq TestBot BOT Today at 10:31 AM
@Dyl Testing the nag command!

Dyl Today at 10:31 AM
Typing will end the nag, resulting in a final message.

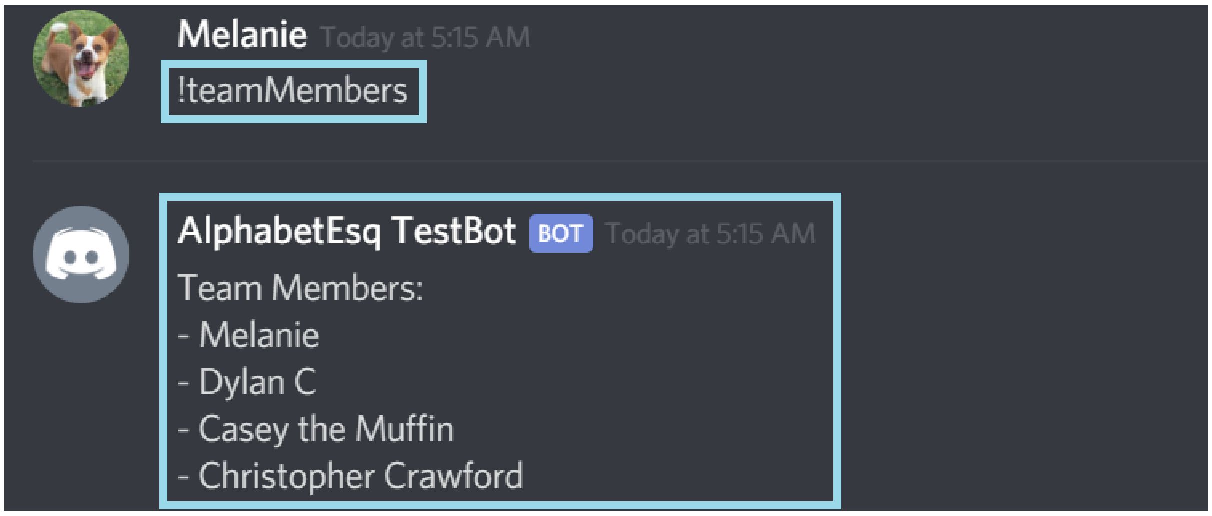
AlphabetEsq TestBot BOT Today at 10:31 AM
Nag completed successfully.

Figure 3.2.2 - *!nag*

3.3 MVF3 – Message Counter(s)

Bot is capable of tracking individual user contributions and actions in the Discord chat, providing three feedback options; !count, !teamMembers, !teamPlayer.

The ***!teamMembers*** command has been completed and implemented, with most functionality working. Users can type "!teamMembers" into the chat window, and the bot will return a list of all the team members that have contributed to the chat in that particular channel, in the accessible history. Accessible history is the greater of 100 messages or the number of messages cached for that channel.

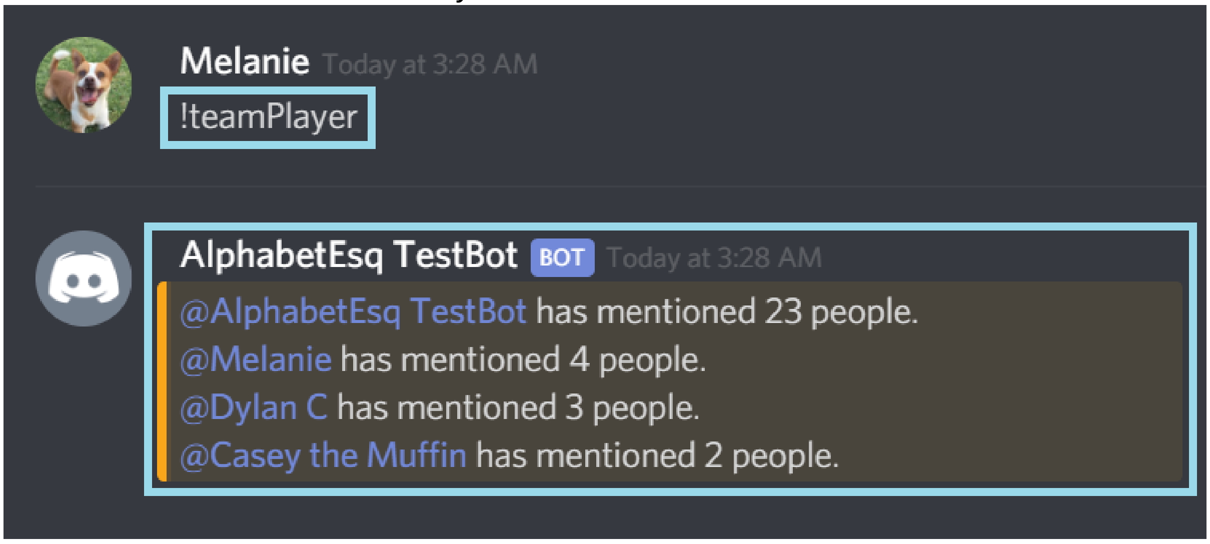

Melanie Today at 5:15 AM
!teamMembers

AlphabetEsq TestBot BOT Today at 5:15 AM
Team Members:
- Melanie
- Dylan C
- Casey the Muffin
- Christopher Crawford

Figure 3.3.1 - *!teamMembers*

Early testing has found this command is unable to include members who have disconnected from the server, as their *member* objects are null upon leaving. Accordingly, no *member.displayName* property is available, resulting in these users not being included. The bot successfully returns a complete list of all *active* members who have been a part of the accessible chat history.

The ***!teamPlayer*** command has also been completed and implemented, mostly as per designs. When users send "!teamPlayer" to the chat, the bot will respond with a list of how many times users have mentioned others in their messages. This count includes the times they have used the tag @*everyone*. Designs were altered slightly with the addition of the message cache, resulting in the bot noting how many messages were checked. This amount will be the greater of the last 100 or the amount of messages cached for that channel. This is yet to be added.



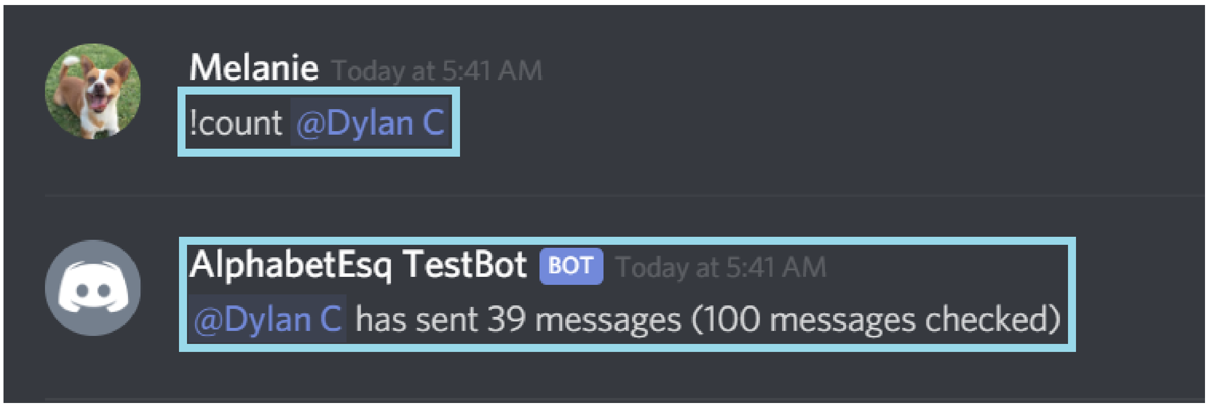
Melanie Today at 3:28 AM
!teamPlayer

AlphabetEsq TestBot BOT Today at 3:28 AM
@AlphabetEsq TestBot has mentioned 23 people.
@Melanie has mentioned 4 people.
@Dylan C has mentioned 3 people.
@Casey the Muffin has mentioned 2 people.

Figure 3.3.2 - !*teamPlayer*

The bot currently includes itself in teamPlayer, something which wasn't originally intended. Due to time constraints, this minor issue was left unaddressed. Our team would have liked to spend extra time on the wording of the output, making it clearer for end users. Some minor changes were made ("*has mentioned x people*" vs "*x user mentions*"), but the information could still be presented in a clearer way.

The command ***!count*** has been completed and implemented, with a slight change on our design. We chose to modify the format of the command from !count(*username*) to !count @*username* as it is more user-friendly, with a handy tag feature included. When the user sends !count @*username* into the text channel, the user is tagged, and the bot will return the mentioned user, along with the total number of messages this member has sent in that channel.


Melanie Today at 5:41 AM
!count @Dylan C

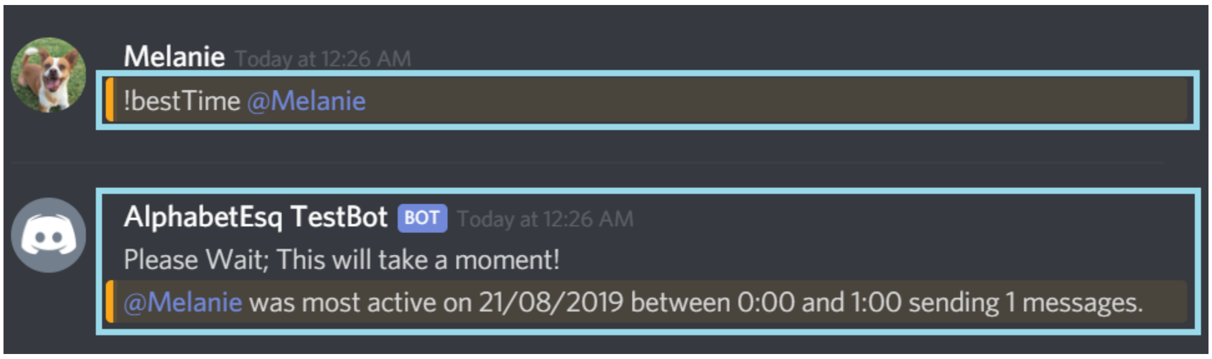
AlphabetEsq TestBot BOT Today at 5:41 AM
@Dylan C has sent 39 messages (100 messages checked)

***Figure 3.3.3 -** !count*

3.4 MVF4 – Extended Team Functionality

!bestTime, !setMeeting, !dontFreak

The command ***!bestTime*** has been completed and implemented, with a slight change in design. Similarly to the !count command, we changed the format of the command from !bestTime(username) to !bestTime @username. When the user types this into the chat, the bot will return the hour of day the mention user has posted the most.

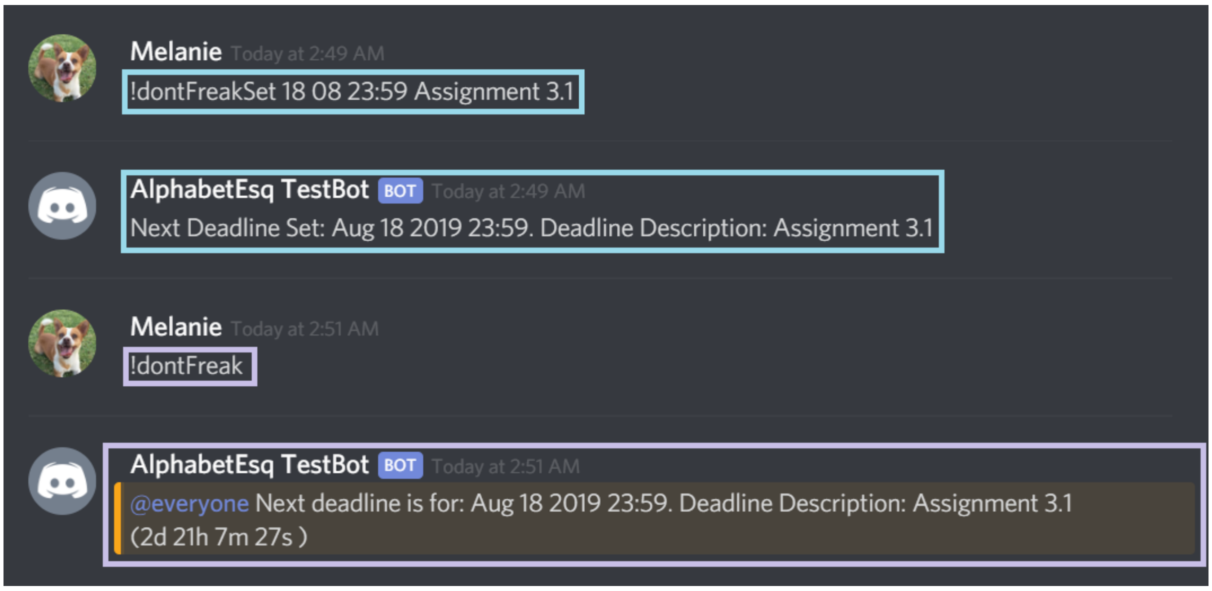

Melanie Today at 12:26 AM
!bestTime @Melanie

AlphabetEsq TestBot BOT Today at 12:26 AM
Please Wait; This will take a moment!
@Melanie was most active on 21/08/2019 between 0:00 and 1:00 sending 1 messages.

***Figure 3.4.1 -** !bestTime*

The command ***!dontFreakSet*** has been completed and implemented, with a slight adjustment on initial input and output designs. Entering !dontFreakSet along with the day, month, time, and deadline description, sets the deadline for the task described. The bot then returns a confirmation message back to the user with the due date and description.

The command ***!dontFreak*** has been completed and implemented, with an improvement on initial bot output designs. Inputting !dontFreak will return the previously set deadline, tags everyone in the server and indicates the time left until the deadline.


Melanie Today at 2:49 AM
!dontFreakSet 18 08 23:59 Assignment 3.1

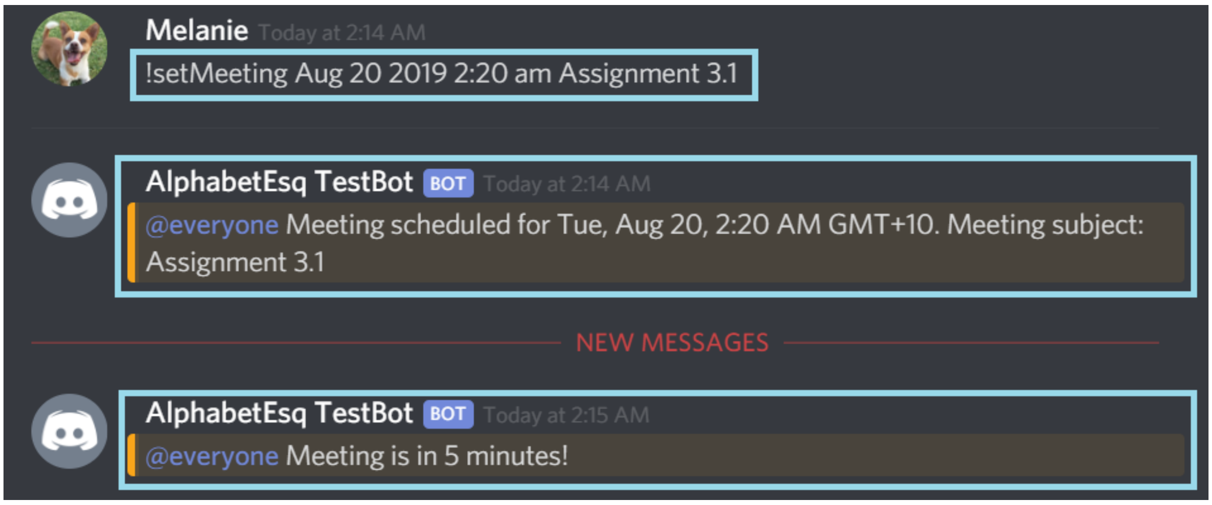
AlphabetEsq TestBot BOT Today at 2:49 AM
Next Deadline Set: Aug 18 2019 23:59. Deadline Description: Assignment 3.1

Melanie Today at 2:51 AM
!dontFreak

AlphabetEsq TestBot BOT Today at 2:51 AM
@everyone Next deadline is for: Aug 18 2019 23:59. Deadline Description: Assignment 3.1 (2d 21h 7m 27s)

Figure 3.4.2 - *!dontFreakSet & !dontFreak*

The command ***!setMeeting*** has been completed and implemented with some adjustments in design. A user can type !setMeeting along with a day, month, time and meeting subject. The bot will store this information and return a confirmation message displaying the date, time and subject of the meeting. The bot will periodically send reminders of the meeting time to the channel at specified intervals. Five minutes before the set time, the bot sends a final message that the scheduled meeting is about to begin.


Melanie Today at 2:14 AM
!setMeeting Aug 20 2019 2:20 am Assignment 3.1

AlphabetEsq TestBot BOT Today at 2:14 AM
@everyone Meeting scheduled for Tue, Aug 20, 2:20 AM GMT+10. Meeting subject: Assignment 3.1

NEW MESSAGES

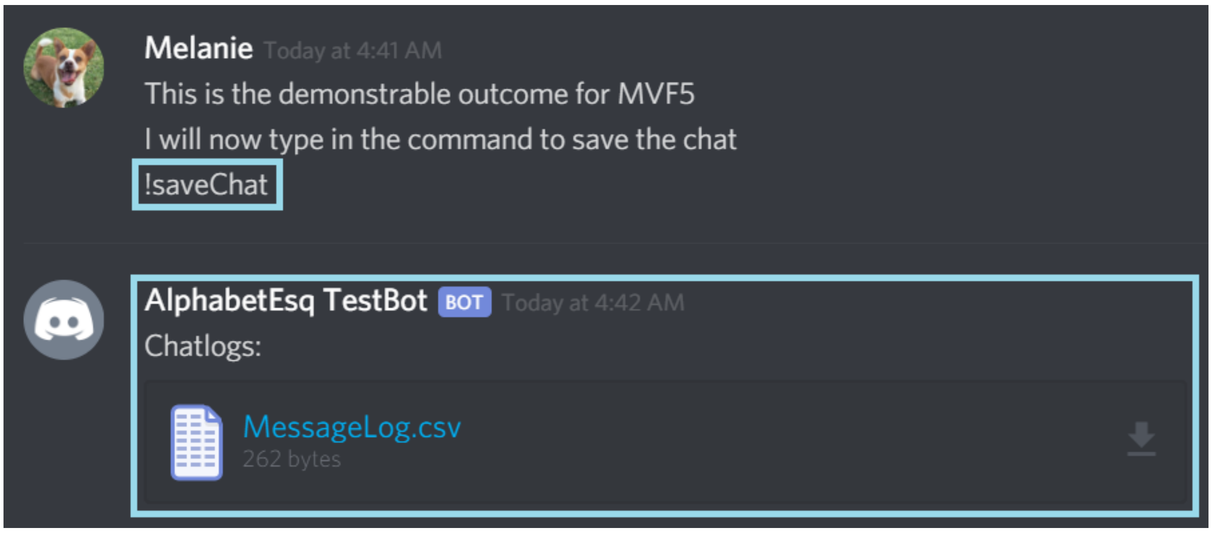
AlphabetEsq TestBot BOT Today at 2:15 AM
@everyone Meeting is in 5 minutes!

Figure 3.4.3 - *!setMeeting*

3.5 MVF5 – Chatlog Exporting

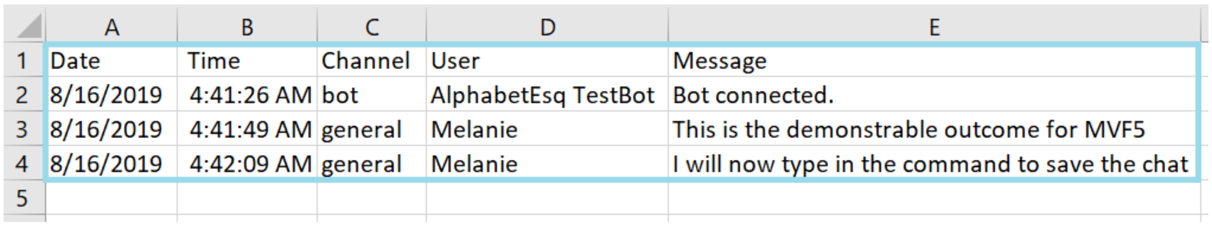
Bot exports/uploads chat history in comma separated values (.csv) format

The ***!savechat*** command was implemented and functions as per designs. Typing the !saveChat command results in the bot successfully uploading a CSV file containing all cached messages since the bot connected. File includes messages from all channels viewable by the bot. Information logged is as per design, with date, time, channel, user and message content contained in the uploaded file.


Melanie Today at 4:41 AM
This is the demonstrable outcome for MVF5
I will now type in the command to save the chat
!saveChat

AlphabetEsq TestBot BOT Today at 4:42 AM
Chatlogs:
MessageLog.csv
262 bytes

Figure 3.5.1 - !saveChat



	A	B	C	D	E
1	Date	Time	Channel	User	Message
2	8/16/2019	4:41:26 AM	bot	AlphabetEsq TestBot	Bot connected.
3	8/16/2019	4:41:49 AM	general	Melanie	This is the demonstrable outcome for MVF5
4	8/16/2019	4:42:09 AM	general	Melanie	I will now type in the command to save the chat
5					

***Figure 3.5.2 –** Exported messageLog.csv file*

4 Project Progress - Scope Creep

With this project, our team aimed to produce a functional Discord bot with various features for aiding student and group collaboration. Our scope included a design document outlining 5 particular features for creation, with information on how each feature would function.

Originally our team had planned to use Discord.js and have no or limited help / user information available. Scope creep was experienced in a minor form when our team came across Discord Commando, an alternative version of Discord.js. This alternative build has functionality for an automated !help command, allowing the user to view user documentation, information and examples on all commands. Our team made the decision to implement this change after a team meeting in Week 6. We felt this change would be beneficial for end-users as it provides ease-of-access to user documentation.

During a team meeting our team decided to not proceed with any extended features, though these features weren’t included in the original scope and were simply intended for if time permits. This decision was made as the team agreed it would be best to instead focus on the main MVF features, putting extra time into ensuring the quality of these features opposed to increasing the quantity of features.

Original plans involved the use of Discord.js for all features requiring access to bulk messages, though it was found this would potentially be against Discord T&Cs. As a result, our team experienced scope creep when we found ourselves also coding a feature to cache messages locally. Such functionality was originally assumed available in Discord.js and accordingly wasn't included in our project scope.

A form of scope contraction was experienced with the !teamMembers command after an issue was noted during preliminary testing. Original designs indicated this command would include members that have left the server. Due to a problem noted with accessing the display name(s) of these users, such functionality would be more difficult than expected. Due to time constraints, it was decided this command would be left as is, noting it will no longer include users who have left the server.

5 Project Progress - Progress

Overall, our features were coded in the time frame outlined in our schedule, though some features are yet to undergo full validation testing. Various issues and setbacks were noted, though features were still coded on time as our team had allocated extra time in expectation of such contingencies.

Our first feature (Bot Connection, ***MVF 1***) was completed while researching and learning features of Discord.js. The feature was ultimately committed to GitHub along with our first commit, prior to officially starting project work. This feature has been tested and test methodology documented, testing was passed successfully.

Our second feature was the !saveChat feature (***MVF 5***), completed after it was found writing to files in JavaScript is very similar to in Java. Multiple members had previous experience in Java and a CSV file generator was quickly created. We planned for this CSV file to serve as a message cache for other commands as well, though later switched to an array of message objects. Multiple changes were made to this feature as we made the switch to a message array, though these changes were minor. The command now pulls data from the message array, generating and uploading a CSV file to the Discord channel. This feature has been successfully tested, passing all validation testing requirements.

Our third feature (***MVF 2***) was similarly started very early, with !dateTime completed and !nag partially coded on the first GitHub commits. !nag was functional, though didn't correctly cease the 3-hour repetition interval upon mentioned user posting. This bug was later corrected in a future commit. These features have both been tested, passing all validation testing requirements.

Around this time our team made the switch to Discord Commando, this switch was with considerations to the extra automated command related features available. As an example, Discord Commando has an automated !help command which provides users with information on all available commands. During this switch, our team added a global 'message array' - a cache of all messages received by the bot. This array is accessible by all features and commands, serving as a way of accessing bulk messages. Previously coded commands were updated to use this array.

Our fourth feature completed was the various message counters (***MVF 3***), these features had some similarities, and all utilised the message cache. As we had multiple team members available to work on this feature the first two commands were completed in a timely manner. During coding of the third command an issue was noted in the !saveChat feature, resulting in less members available for coding of the message counter features. This resulted in some delays, though the coding and implementation was still completed on schedule. As a result of using the message cache, these commands will only process the greater of either the last 100 messages or relevant messages available in the cache. Designs were modified slightly to include *(x messages checked)* in the output. Brief testing was completed successfully on !count and !teamPlayer, though due to the delays noted, full validation testing is yet to be conducted. An issue was noted during the preliminary testing of !teamMembers, where members who have left the server aren't included. It was found that no *member* object is available for users who have left the server, as such *member.displayName* is not accessible. This feature has been left in as is and currently correctly lists all active members that have been a part of the accessible channel history. Due to time constraints, no further time has yet been spent attempting to fix this issue. Our team now plans to complete full validation testing prior to presenting feature demonstrations on August 25, with the changed designs and expectations of !teamMembers in mind.

Our fifth feature, extended team functionality (***MVF 4***), was completed alongside the fourth. These features were coded quite quickly as multiple commands use similar coding functionality (ie intervals). Our team had slightly underestimated the complexity of advanced date related functions (particularly taking date/time inputs) but overcame this through extensive learning and research. The feature was ultimately completed and implemented on time, though full validation testing on all commands was delayed and is yet to be completed. Our team now plans to complete full validation testing on !bestTime and !dontFreak prior to feature demonstrations on August 25. There were some issues with an early version of !setMeeting, though these issues were identified, and a later version implemented. The !setMeeting command now functions as per designs and has passed validation testing.

6 Project Progress - Testing

Full validation testing was conducted on Bot Connection (*MVF 1*), !nag and !dateTime (*MVF 2*) and !saveChat (*MVF 5*). The extended team functionality (*MVF 4*) and message counter (*MVF 3*) commands have undergone brief testing, with full validation testing scheduled for completion before 25 August.

6.1 MVF1 – Bot Connection

Assuming relevant permissions, bot can be imported and connected to any discord server. Bot shows as 'online' and has a 'bot' tag in the right-hand side users panel.

Bot connection was validated successfully. Bot can be implemented into any server where administration permissions are held. Bot shows as 'online' in the right-hand side panel and triggers a new-user message upon first connection.

Bot Connection Steps:

1. Create a new bot in the Discord Developer Portal, connecting it to any server where administration permissions are held.
2. Select the bot in Discord Developer Portal, click the 'bot' tab, then copy the bot access token.
3. Place the access token in AlphabetEsq bot and running the code.

Following this, the bot will boot up and connect to any servers selected in step 1, showing as 'online' and sending an accompanying connection message. At this point, the bot is connected and ready to respond to commands outlined in MVFs 2-5.

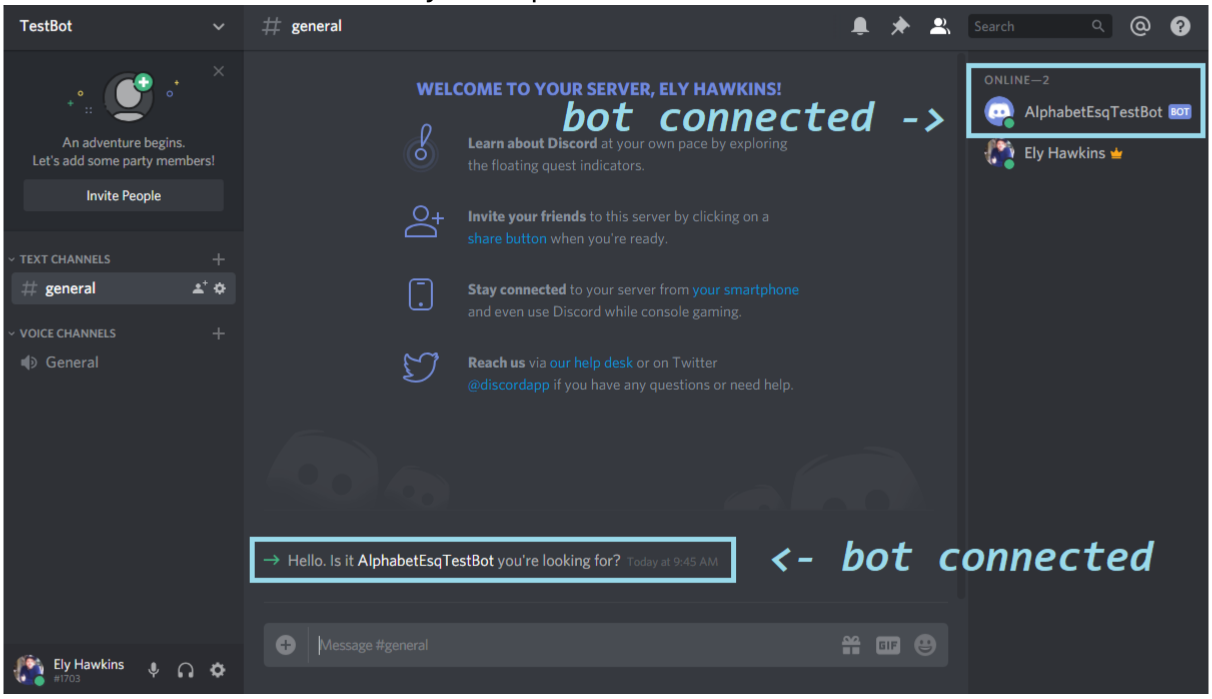

TestBot
general
WELCOME TO YOUR SERVER, ELY HAWKINS!
bot connected ->
Learn about Discord at your own pace by exploring the floating quest indicators.
Invite your friends to this server by clicking on a share button when you're ready.
Stay connected to your server from your smartphone and even use Discord while console gaming.
Reach us via our help desk or on Twitter @discordapp if you have any questions or need help.
An adventure begins.
Let's add some party members!
Invite People
TEXT CHANNELS
general
VOICE CHANNELS
General
ONLINE—2
AlphabetEsqTestBot BOT
Ely Hawkins
Hello. Is it AlphabetEsqTestBot you're looking for? Today at 9:45 AM
<- bot connected
Message #general
Ely Hawkins #1703

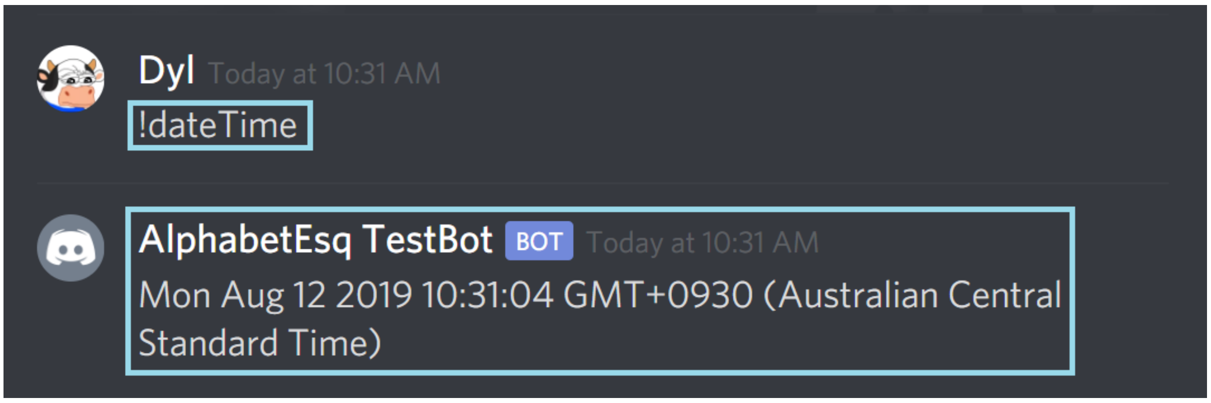
Figure 6.1.1 **-** *Bot Connection*

6.2 MVF2 – Datetime & Nag

Respond to prompts; !dateTime, !nag

Validation tests were conducted and passed. Both commands the !dateTime and !nag commands have been implemented and work as per designs.

The ***!dateTime*** command was successfully tested with different users and multiple time zones. The command results in our Discord bot responding correctly with the date, time and time zone the bot is running in. Date and time is formatted as per designs of *Day Mmm dd yyyy hh:mm:ss GMT+xxxx (Timezone)*. The timezone used is the timezone of the bot, as per designs. When the bot is hosted on a host using a different timezone, the output timezone changes accordingly.



Dyl Today at 10:31 AM

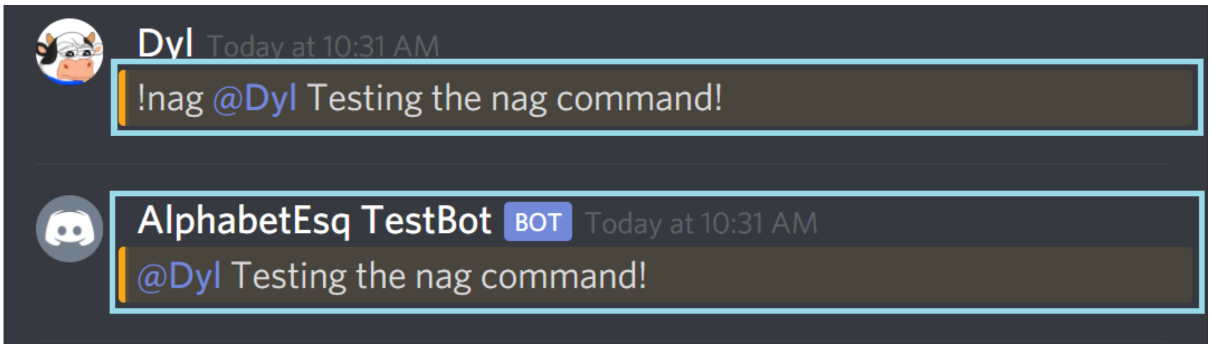
!dateTime

AlphabetEsq TestBot BOT Today at 10:31 AM

Mon Aug 12 2019 10:31:04 GMT+0930 (Australian Central Standard Time)

Figure 6.3.1 - *dateTime*

The ***!nag*** command was tested with different users and demonstrated that an accompanying message will be sent to the mentioned target user at 3 hour intervals.



Dyl Today at 10:31 AM

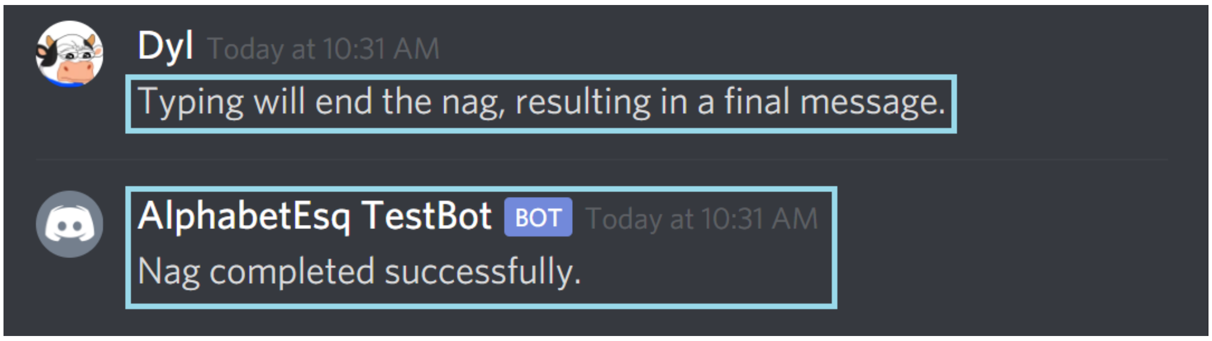
!nag @Dyl Testing the nag command!

AlphabetEsq TestBot BOT Today at 10:31 AM

@Dyl Testing the nag command!

Figure 6.3.2 - *nag start*

Upon the user replying, the bot then sends 'nag completed successfully'. Bot successfully tags mentioned user, sends accompanying message, stops when user posts and sends final message at such time. All validation test criteria met. Full demonstration of this command including evidence of 3 hour intervals will be available in an upcoming video demonstration on 25th August.



Dyl Today at 10:31 AM

Typing will end the nag, resulting in a final message.

AlphabetEsq TestBot BOT Today at 10:31 AM

Nag completed successfully.

Figure 6.3.3 - *nag completed*

6.3 MVF3 – Message Counter(s)

Bot is capable of tracking individual user contributions and actions in the Discord chat, providing three feedback options; !count, !teamMembers, !teamPlayer

All message counter commands have undergone some form of testing, though full validation testing is yet to be documented. Early testing indicates the ***!teamPlayer*** and ***!count*** commands work as per expectations.

An issue was noted with the ***!teamMembers*** command where users that have left the server aren't included in the results. It was found this issue relates to the ***member.displayName*** information not being available for users who are no longer apart of the server. Due to time constraints, it was decided this command would be left as is and will no longer include users that have left. Accordingly, the validation requirement of including members who have left the server in results will be removed. We plan to conduct full validation testing prior to a video demonstration on the 25th of August.

6.4 MVF4 – Extended team functionality

!bestTime, !setMeeting, !dontFreak

Preliminary testing of ***!bestTime*** and ***!dontFreak*** has been conducted and early indications are that commands are functional.

Issues with the ***!setMeeting*** command were noted during the first round of testing. Notably, the command wasn't using @*everyone* and the final message was being sent at the meeting time; not five minutes before. An alternative version was implemented and full validation testing conducted. The command now successfully uses @everyone, repeats every three hours and sends a final reminder 5 minutes before the scheduled meeting.

Full validation testing of ***!bestTime*** and ***!dontFreak*** was delayed due to the availability of team members and issues as noted in Progress report. Our team plans to conduct complete validation testing prior to a video demonstration on 25th August.

6.5 MVF5 – Chatlog Exporting

Bot exports/uploads chat history in comma separated values (.csv) format

Early testing identified an issue with the ***!saveChat*** command where messages were added to the chatlog file in a seemingly random order. It was identified this issue related to the use of a for loop. A bug was also previously noted where extra columns were added if a message contained a comma. These bugs have since been fixed and further testing undertaken, both ensuring commas don't result in issues and checking the order messages are listed in.

The command now functions as per designs. Typing !saveChat results in the bot successfully uploading a CSV file containing all cached messages, in order. File includes messages from all channels viewable by the bot. Information logged is as per design, with date, time, channel, user and message content contained in the uploaded file.

7 Project Progress - Tools & Technologies

7.1 Collaboration Workspaces

7.1.1 Google Docs

Google Docs allowed for live collaboration on documents, ensuring team members could work together collaborating on the creation of the documentation. Throughout this project GoogleDocs was used for the creation, version control and collaboration of documentation. GoogleDocs was essential for online collaboration.

Google Docs Link: Google Drive - Building IT Systems
Google Docs Guide / User Documentation: Google Docs - Docs Editors Help

7.1.2 Trello

Trello is specifically designed along Kanban board principles in order to assist teams in efficient delivery of its objectives. Trello's power comes from instant update across geographically remote users. Our team has made use of Trello to ensure all team members were aware of their responsibilities and assigned tasks. Team members have used this space to collaboratively document progress and provide updates on the status of on-going tasks.

Trello link: AlphabetEsq Trello Board
Trello Guide: Getting Started with Trello

7.1.3 Discord

Discord has been used as our team's primary means of communication throughout this project. Text channels were used for announcements and discussion, with weekly meetings held in voice channels.

***Discord Link*:** AlphabetEsq Discord

7.1.4 GitHub

Our team made use of GitHub for collaborative software development, utilising revision and version control for ease of testing. GitHub integrates well with Visual Studio Code, allowing for automatic version control and syncing. The commits log allowed for easy tracking and logging of contributions.

GitHub link: AlphabetEsq GitHub
Access Guide: AlphabetEsq GitHub Clone Guide
General User Docs: GitHub Guides

7.2 Software

7.2.1 Visual Studio Code

Visual Studio Code is the primary piece of software we have used to build our bot. It is a source-code editor that provides a workspace to build and debug (web) applications. Key features include:

- In-built Git functionality which allowed us to share and sync code to GitHub.
- Allows for the coding of JavaScript outside of the browser.
- Highly customisable with a vast array of extensions.
- Simple user friendly layout and design.

Visual Studio Code Link: https://code.visualstudio.com/
Documentation: https://code.visualstudio.com/docs
Version: 1.35

7.2.2 Discord.js

Discord.js is a node.js module that allows developers to interact with the Discord API, and has served as the core of our project. With extensive features and comprehensive user documentation, Discord.js has been very beneficial for the success of our project. We have made considerable use of Discord Commando which is the official command framework for Discord. Commando had extra functionality for grouping commands and automated features such as the !help command.

Discord.js Link: Discord.js
Documentation: Discord.js Docs
Discord Bot Setup (video): Code a JS Discord Bot: Pt 1 - Getting Set Up
Discord Commando (video): Code a JS Discord Bot: Pt 2 - Setting up Commands
Version: 11.5.1

7.2.3 Git-scm

Git-scm allowed us to integrate Git into Visual Studio Code. Without it we would have needed to use both applications separately. It has given us access to the following Git functions without leaving VS Code:

- Initialize a repository.
- Clone a repository.
- Create branches and tags.
- Stage and commit changes.
- Push/pull/sync with a remote branch.
- Resolve merge conflicts.
- View diffs.

Git-scm Link: https://git-scm.com/
Documentation: https://git-scm.com/doc
Version: 2.22.0

7.2.4 Node.js

Node.js is a JavaScript runtime allowing for JavaScript to be used outside of a browser. When used in conjunction with Visual Studio Code, Node.js allowed us to instantly run and test code with ease. Through the use of Node.js, our team was able to develop and test our Discord bot without any issues. Multiple alternatives are available, such as Javac with Discord Java Library. Our team felt the Discord JavaScript library was the most well documented, as such Node.js was essential.

Node.js Link: Node.js
Documentation: Node.js Docs
Version: 10.16.0 LTS

7.3 Tools

7.3.1 Discord Developer Portal (Discord API)

Discord Developer Portal provides a well documented and extensive set of tools for the development of applications which work with Discord API. It is the most accessible and user-friendly way to develop applications using the Discord API. The portal contains various features allowing for easier collaboration and development of bots. The 'Teams' feature has been used to ensure all members have the required access and permissions to access and control our shared bot.

Discord Developer Portal Link: https://discordapp.com/developers/
User documentation: https://discordapp.com/developers/docs/intro
Version: 6

7.4 Resources

7.4.1 DiscordJS / Discord Bot Documentation

Discord.js has extensive documentation available for developers. Documentation is available on every class, function and variable. We have used this documentation for researching and learning about various features and functions unique to Discord.js.

Discord.js Documentation: Discord JS Docs
Discord.js Tutorial: JavaScript Discord Bot Tutorial
Discord Bot Setup (text): How to make a Discord bot
Discord Bot Setup (video): Code a JS Discord Bot: Pt 1 - Getting Set Up
Discord Commando (video): Code a JS Discord Bot: Pt 2 - Setting up Commands

7.4.2 YouTube

YouTube is host to many video tutorials and resources which has proved helpful in the learning stages of development. Videos are available for setting up Discord bots, tutorials on Discord Commando and general JavaScript tutorials amongst other things.

Discord Bot Setup (video): Code a JS Discord Bot: Pt 1 - Getting Set Up
Discord Commando (video): Code a JS Discord Bot: Pt 2 - Setting up Commands

7.4.3 Lynda

Lynda is a resource with an abundance of information on a wide range of topics. As RMIT students, all team members have unlimited access to Lynda training. Our team had made particular use of training in JavaScript.

JavaScript Tutorial: JavaScript Essential Training

7.4.4 GitHub Learn

GitHub Learn provides various lessons and challenges aimed at allowing new users to learn to use GitHub and its various features. As our team has made extensive use of GitHub throughout our assignment, it was important that all members gained an understanding of the program.

GitHub Learn: Resources to Learn Git

7.4.5 Visual Studio Code Docs

Visual Studio Docs provides various guides for setting up and getting started with Visual Studio Code. As our team has made use of this program for all of our collaborative coding work, a great understanding of its features was essential. The documentation section has guides on various topics, including node.js.

Visual Studio Code Docs: Getting started with Visual Studio Code
Visual Studio Code Setup Guide: Setting up Visual Studio Code
Visual Studio Code Intro Videos: Visual Studio Code Introductory Videos
Visual Studio nodeJS Guide: Build node.js Apps with Visual Studio Code

8 Challenges and Learning

8.1 Difficulties

What have you found easy? What have you found difficult? What was unexpected? In what ways have you been stretched?

8.1.1 Casey Edwards

I found coding the features I was assigned to be a little easier than I was expecting. Having limited knowledge in JavaScript coding, I was expecting learning the language to be a very big learning curve but thus far it has been an easy task.

However, I did find difficulty in finding reliable and properly explained information for Discord.js. This made coding the features I was assigned initially very difficult and tedious, but I have since overcome the limitation.

8.1.2 Christopher Crawford

Considering that I started with the group quite late, I found the transition into the group quite easy. The programming side of the project has been difficult, but I knew this would be difficult going into the project. The unexpected was the demand of the project work. I have been stretched in terms of working and studying full-time.

8.1.3 Dylan Currie

I found working with JavaScript easier than originally expected. I had past experience coding in Java, though wasn't aware how familiar the two languages are.

I found some difficulties in collaborating on documents over the internet. Though GoogleDocs is very helpful with this, there is still a reduced level of communication and collaboration over levels seen in a shared office space or class room.

8.1.4 Ely Hawkins

I found group communication and collaboration easier than expected. Compared to other group projects I was very surprised at how easy it was to communicate and organise within a group.

I found the programming aspect of the assignment harder than I had first anticipated. I think doing well in other programming units had given me a false indication of my skill level.

8.1.5 Melanie Broersen

I have found that JavaScript was quite easy to learn, being so similar to Java which I am pretty familiar with. I found it difficult to not only word things when doing write-ups, but to have other people relying on me to do certain tasks that included write-ups. The responsibility that comes with being part of a team, and to finish my tasks to an agreed upon schedule so that I don't put other team members behind, has been a bit stressful.

I didn't expect myself to be able to step in so much when other members dropped out, went MIA, or were unable to do certain tasks for various different reasons. I've been pushed to talk more than I normally would in group chats, and to finish tasks sooner rather than last-minute.

8.2 Challenges

What challenges were you expecting? What have you done to address these? Have they been overcome?

8.2.1 Casey Edwards

I was expecting myself to be less communicable within the group setting given my mental illness and social anxiety however thanks to the team it has been made easier for me to communicate and give information.

While these challenges are that of a more personal nature I believe the team has helped significantly with this. Their friendly attitude and willingness to work around my illness has allowed me to feel more confident that I’m not letting anyone down when I go absent for more than one or two days at a time.

Because of the above I have also had some challenges keeping to the deadlines set by the group leader Dylan. However, I have overcome these issues with proper communication to my group and getting tasks reassigned when they are something that needs to be done immediately but I am not available to do the task.

8.2.2 Christopher Crawford

The challenges of programming were something that I was expecting in the project. The online tools provided by RMIT, online tutorials and general study. The tools and the knowledge base of the group helped overcome any programming weaknesses.

8.2.3 Dylan Currie

I was expecting some challenges with communication and collaboration, along with challenges in learning JavaScript and Discord.js.

Communication challenges were addressed through agreement on communication expectations, along with the use of tools (ie Trello) to increase communication.

Collaboration challenges were partially addressed through the use of valuable collaboration tools, such as Visual Studio Code used with GitHub, and GoogleDrive. Visual Studio Code, in conjunction with GitHub, made collaboration on code very simple. All members had access to the latest version of code, with revision control done automatically.

GoogleDrive made collaboration on documents easier, though I felt there was still some challenges over sharing physical office or classroom space. Asking for team input wasn’t as simple as standing up and approaching their desk. At times crucial input wasn't received until 12-24 hours later - a huge delay over the minutes input can be sought in when sharing a physical space.

I originally expected Discord.js may be a little complex, though the extensive documentation available made it very easy to use. On a team level, we addressed this challenge by sharing links as we came across valuable information or resources.

8.2.4 Ely Hawkins

I was expecting challenges with group communication and cohesion as it has been an issue with most of my group work in the past.

To address this, I took the initiative to create the group on the canvas forum. This way I could set my ideas and values out from the beginning and try and find people of a similar persuasion.

Tools such as google docs, GitHub, Trello, google drive and discord facilitated what was a relatively smooth group project. Trello was something that kept us on track and provided clear deadlines and structure to the process.

8.2.5 Melanie Broersen

Some challenges that I was expecting were my own contribution to group discussions and coming up with ideas for any aspect of the project. Also, verbal communication in group and mentor meetings, which has always been a challenge for me. I made it easier for myself to talk and contribute in the beginning by having topics and questions prepared.

I also found it a challenge to keep up to date with my assigned tasks, so I have made sure to leave enough time each night to try and get them done. I believe I have overcome the former challenge, however the latter challenge is still being worked on.

8.3 Skills, Experiences and Learning

What are the new skills or experience you have learned and/or developed?

8.3.1 Casey Edwards

In the course of this project I have learned two major things. The first being the extension of my programming abilities; I am still fairly new to programming however have found that I can pick it up with relative ease and learn. In addition to these extensions of my current skills I have also learned about Visual Studio Code which is a piece of software that I had not used prior to this project.

The second thing I have learned is that communication is the most important part of being part of a team. Making sure that my team knows where I'm at, what I'm doing, and when they can expect me to finish particular tasks. This in terms helps me complete my work in a more relaxed fashion as I do not feel as rushed and helps my group by giving them a reasonable expectation of when my work will be ready.

8.3.2 Christopher Crawford

Experiencing tools like Visual Studio, GitHub, and the capabilities of Discord Bot has been a great learning curve. It has allowed me to gain skills using different tool sets and help my understanding of programming requirements.

8.3.3 Dylan Currie

I've further developed my programming skills by gaining experience using JavaScript. I had previous experience with Java, though throughout this project I've developed my knowledge further specifically with JavaScript.

I've gained experience using Visual Studio Code, including setting up projects and integrating GitHub functionality. I've also had valuable experience using Discord.js, the Discord library our team used for Discord API access.

8.3.4 Ely Hawkins

I now have a basic understanding of the JavaScript programming language. Along with this I have gained experience using Visual Studio Code, GitHub and Trello which will all be valuable to me in the future.

8.3.5 Melanie Broersen

I have learned how to collaborate in a group and have improved my communication skills by a lot. JavaScript is now in my skill set at a basic level, as well as the various programs and platforms we have had to use during this unit, including Discord, Visual Studio Code, GitHub, and Trello.

8.4 Plan Change Reflection

Plans change, did yours? In hindsight, would you make any changes to your plan now? Have things turned out as you expected? What would you do differently if you had your time over again?

8.4.1 Casey Edwards

Being a latecomer to the team certainly made for some interesting restructuring. However, I do believe that aside from this change the team allocation and goals all roughly stayed the same and nothing was drastically changed from the point where I joined the team until now.

8.4.2 Christopher Crawford

My plans did change initially because of some personal issues early in the subject. This meant that I had joined the group quite late and was playing catch up a lot of the time. In hindsight, I would obviously like to have started with a group from the beginning and this way could embed myself more into the group.

8.4.3 Dylan Currie

Our team experienced various role changes and restructuring due to the departure and addition of members in the early stages of our project. The result was myself being appointed team leader and Ely assuming the role of second in charge.

Our aims and goals stayed roughly the same (with the exception of dropping EVFs), with a few minor changes in MVFs. The majority of features turned out as expected, with minor changes in the background. For example, we had originally planned to pull messages directly from the Discord API, but opted to instead cache them locally after reviewing Discord Developer T&Cs.

MVFs were completed out of order, though as an Agile team this was expected. Our team made progress on MVFs as new coding techniques were learnt or Discord.js features found. We often helped each other out with features not assigned to ourselves, working together and sharing coding techniques.

There were some difficulties with how the MVFs were set out. In hindsight, I feel our team would have benefitted from separating each sub-feature into a separate MVF. For example, having a separate MVF defined for each individual command. This would have also helped us better separate leads for each individual command, rather than having one person trying to lead the coding and implementation of up to three commands at one time. This would be beneficial for easier monitoring of tasks and sharing of workloads.

8.4.4 Ely Hawkins

We had 2 team members drop out during assignment 1 which resulted in a restructuring of our team and redefining of our roles.

Personally, I found I had a much lesser role in the programming side of the project as time went on due to time constraints and the fact I was struggling to pick up JavaScript.

In hindsight I feel as though some of the task assigning within the group could have been done better. We tended to assign very large chunks of work which may have benefitted from being broken down into smaller tasks so as to be more achievable.

8.4.5 Melanie Broersen

My plan was to sit back and let the team leader do their thing and let me know what they wanted me to do. That changed when our first team leader left just after another team member left and we were all trying to redefine our roles. A new team leader was appointed, and I ended up standing in a couple of times when they were unable to. It is very unlike me to take charge of something, as I am a very quiet achiever who likes to stay out of the spotlight.

In hindsight, I feel like the organisation and allocation of tasks could have been organised a little better, however, we were hindered by 3 members dropping out and 2 new members joining late. Which is something none of us could predict or help.

If I had my time over again, I would dedicate more time to coding our bot and less time to learning the language. I feel like learning JavaScript only got me so far, but as you start coding you figure out what you need to know specifically, which you can then go and learn in depth.

8.5 Timetable Reflection

Was your timetable realistic? What would you change in your timetable, knowing what you do now? Justify your answer

8.5.1 Casey Edwards

I thought the timetable we had was very realistic but could have used some refinements in the way certain parts were structured in a way where it became difficult to manage time around tasks because they were bigger than you could predict. I think because of this it made it difficult to inform the team what was completed and was not completed.

8.5.2 Christopher Crawford

It is hard for me to comment on the timetable as I joined the group later. However, considering that the group had some many changes early on, I think the group managed very well with the current timetable.

8.5.3 Dylan Currie

Though I believe our timetable was realistic given the amount of time given for each MVF, I would break it down further in future projects. I believe our team could have benefitted by specifically allocating time to research, learning, coding and testing, opposed to just *MVF development* and *MVF validation*. This could have proven beneficial in monitoring progress, allowing us to ensure constant progress is being made to the completion of features.

I believe we slightly underestimated the time required for the development of each feature. In part this was due to our team experimenting with various ways of coding similar functionality, learning as we progressed. As an example, the !setMeeting command was coded three different ways while experimenting with various ways of getting user entered data into a ‘Date’ object. Our team had planned for this in the creation of our schedule, allocating enough time to ensure our project wasn’t jeopardised by delays or contingencies.

8.5.4 Ely Hawkins

We managed to stick to our deadlines for the vast majority of the project which shows we had a good timeline. As mentioned above I believe we could have broken tasks down into more manageable for one person to achieve.

8.5.5 Melanie Broersen
I believe our group timetable was quite realistic, however, my personal timeline was slightly unrealistic, as I had not accounted for all the coursework and written assignment work that I still had to do before I started coding. I had planned to start working on the bot a lot sooner. Knowing about all the extra bits and pieces of work that needed doing, I would try to do them sooner, to leave myself with more time to work on the actual product.

8.6 Tools and Technologies Reflection
Have the tools and technologies worked out as expected? Have any of the risks you identified materialised? Have there been unanticipated events affecting your progress?

8.6.1 Casey Edwards
Visual Studio Code was a good option for our combined work environment. It meshes will with GitHub allowing for easy updates to be added at any time through its own features. Google drive is always an easy way to share files, it allows for the entire group to know what is done and what isn't done and this in turn has the capability to lower the stress of the entire team.

8.6.2 Christopher Crawford
All tools and technologies have worked as expected. However, there is always initially a teething process with these technologies, but eventually get ironed out over time.

8.6.3 Dylan Currie
Visual Studio Code, GitHub and Google Drive all worked as expected. Visual Studio Code GitHub integration was simple and effective - it worked perfectly. Discord.js worked about as expected, allowing us to access the Discord API through a node.js application.

Unforeseeable complications (ie personal events/work commitments/etc) at times impacted meeting our internal deadlines, though all coursework and major project deadlines were still met. We addressed these complications by reassigning work as required, working around any complications that arose.

An issue was identified regarding compliance with Discord T&Cs, particularly regarding rate limits. Discord API limits bulk message pulls to 100 messages, instead encouraging API users to cache messages locally. This had a minor impact on our progress as multiple features required bulk messages. Ultimately, we addressed this issue by adding a local message cache which all commands could access, using bulk message pulling as a backup where less than 100 messages are cached.

8.6.4 Ely Hawkins

The tools and technologies we used all worked as planned. In terms of unanticipated events we had multiple scenarios where busy work and personal lives affected meeting attendance and approaching deadlines. However, information was always shared to absent members and relevant work was reassigned to other members.

8.6.5 Melanie Broersen

All of our tools and technologies have worked out as expected.

9 Project Processes

9.1 Group Project Learning Reflection

What have you learned about group projects? What has worked well in your group? What hasn't worked well?

9.1.1 Casey Edwards

I have learned that it's important to communicate and be up front with team members about what will be done by the deadline and what will not be done by the deadline.

I think the biggest hurdle for me has been using Trello. I personally do not see it as a super-efficient way of keeping track of tasks within a University environment as it quickly becomes over bloated. Because of this I have often forgotten to update my Trello cards and this has led to confusion among the team.

For our team I believe the best communication resource thus far has been Discord. It's a quick and easy way to communicate progress on tasks, ask for updates on the progress of a task and can even be used to generate a good rapport among the team.

9.1.2 Christopher Crawford

Communication is always a key aspect in every group scenario and has been no different in the Group project. The group has had good avenues of communication with Discord and Trello. Coordination of communication is always difficult in a pure online platform. Perhaps the timing of communication could have been better organised, but again a difficult task with different schedules.

9.1.3 Dylan Currie

I've learned how important communication is in group projects. Without communication, things get done twice or confusion surfaces around the status of tasks.

In the early stages of our project, our team made use of Discord to communicate on progress, leaving Trello purely for outlining completed tasks. This at times resulted in confusion on the status of tasks, particularly with the sheer number of messages in the Discord. Though it served as a great informal way to share progress, it wasn't adequate for good, constant communication.

Later in the project our team switched to using both Discord and Trello for communicating progress. By commenting on cards AND sharing in the Discord, all team members were informed of the status of tasks as progress was made. This ensured all members were able to work more collaboratively together and monitor progress on an ongoing basis. This proved successful in ensuring deadlines were met and tasks reassigned where there were issues or complications identified.

9.1.4 Ely Hawkins

I've learned that it is crucial to select a good leader. Utilising the group member most suited to leadership makes for a more structured and well organised experience. In the past, groups I have been a part of have selected for leadership at random which I now see has been to my detriment.

Our communication and organisation have been the things working best for us. Open and frequent communication through discord has meant all team members are constantly up to speed on what needs to be done.

Work/school/life balance has been something i have struggled with and will need to correct for future projects.

9.1.5 Melanie Broersen

Group projects that are specific to online learning, are very difficult. We are all expected to make time to meet every week, but we are not all available at the same time every week. People studying online are assumedly doing so because they have a different schedule to people studying on campus. Our schedules rarely align, making it quite hard for everyone to attend every meeting. This makes it difficult for everyone to stay on the same page and be all aiming for the same goals each week. I have found that group projects are very worthwhile when everyone is doing their part and collaborating well but can be tricky when different people have different expectations for different tasks.

Having Dylan as our team leader has been great for my personal motivation. As a group I think he has definitely helped keep us on track. Our group members usually responded to team and/or personal pings on Discord, and topics that required team input (excluding meeting times) were easily agreed upon. We have all been fairly easy going in our approach to how things are being done, yet still give our opinions when needed.

Trying to all meet up at the same time for voice meetings has been the biggest challenge, but when trying to communicate via text it was sometimes misinterpreted and caused some friction.

9.2 Group Communication Reflection

What were the group's processes for communication? How were they? Have there been any changes since the start of the semester?

9.2.1 Casey Edwards

Discord has been used throughout the project and I believe was very successful but at times when it's busy within the server and everyone is chatting about work it can be hard sometimes to have what you said noticed causing a little confusion within the group.

We also began using Trello as a means to check boxes on certain tasks so if someone isn't around to confirm directly we had a second method of getting an update even from those unavailable. This has been good for our leaders as it allows them to not always be waiting on responses.

9.2.2 Christopher Crawford

Discord and Trello have been the groups means of communication. I think Discord has been far more successful than Trello Board. It is perhaps the live interaction of Discord which the group seems to prefer. Discord has been excellent because it is far easier to use than Trello Board. I believe Trello Board would have a better use in a setting where people can meet up physically.

9.2.3 Dylan Currie

Discord was used throughout the project as our primary communication tool. This provided semi-successful, though there were times where important information was missed due to the sheer volume of messages being sent. This resulted in some confusion on the status of tasks and progress being made.

Later in the project our team started using Trello in addition to Discord when outlining progress, ensuring all relevant information was made available. This allowed us to share progress both semi-formally and semi-informally, ensuring members were kept in the know. This proved beneficial in monitoring the progress of tasks, ultimately allowing our team to address any identified issues at an earlier stage before deadlines.

Our team meetings were helpful for weekly discussions to make formal decisions. I feel it may have been beneficial to also have a daily *stand up* style meeting where team members outline what they plan to work on for the day. Sadly, with the work, life and other study commitments of our team members, such a daily meeting would have proved near impossible. We instead opted for moving Trello cards and sharing our progress / plans via Discord, though I don't feel this was as beneficial as a daily stand up style meeting would prove.

9.2.4 Ely Hawkins

Discord was our primary means of communication. Use of the text channel was crucial for our group as everyone had varying levels of work and study commitments. This meant we were rarely available at the same time. We also made use of discord for our weekly group meetings.

From assignment 2 onwards we started leaving progress comments on relevant Trello cards which was also a very helpful form of communication. Especially when there were multiple people working on one card.

9.2.5 Melanie Broersen

Using Discord as our main point of contact has been very helpful, especially being able to use mentions for specific members as well as the whole team. Every week we try to all meet to check in on our individual and group progress. The weekly meetings weren't always easy to schedule, as not everyone is always available at the initial agreed upon time. This is also due to new members joining and having work at these times some weeks.

We have been commenting on relevant Trello cards since Assignment 2 with our progress on tasks, and recently have moved our Discord group discussion over to our mentor's server as there was some conflict that needed mediating.

9.3 Project Organisation Reflection

From your experience in this project, what is the most important aspect of organising a project of this nature? Is there anything that you would do differently if you were to start again?

9.3.1 Casey Edwards

Honestly, I think allocation is the most important part. To ensure everyone knows what they are doing and is able to complete it by deadlines.

I would break up the tasks differently if I were to do this project again and was leader of organising allocation.

9.3.2 Christopher Crawford

Communication and having the correct tool set to communicate is the most important aspect, I believe. Yes, I would join the group early as it is always difficult trying to catch up in such a big project.

9.3.3 Dylan Currie

I believe planning and communication are the most important aspects of a project of this nature. Ensuring planning is done in a realistic and professional manner has a major knock-on effect for the later stages of the project.

If we were to do such a project again, I would break down the individual MVF development sections into each command, then each command into the four stages of development. This would allow us to better monitor progress by allocating time for the research & learning, design, coding and testing stages of development, along with more regular deliverables through each command being its own deliverable. Our team made the decision to some-what separate each command as a separate deliverable, though didn't go as far as separating our schedule into the separate stages of development.

9.3.4 Ely Hawkins

Organisation and communication are the most important aspect of a project like this. Especially when it is all being conducted online. Trello was incredible for this as everything was able to be set out in such detail and with a notification function so that it was easy to stay on track. Without this resource we would not have been able to accomplish close to what we did.

The one thing I would change is breaking down assigned work into more manageable tasks. I also would have assigned just one group member to more of the tasks. Often, we had multiple members assigned to a single task and before everyone could get a chance to start it someone else had finished it. This at times lead to inequality of work being done that wasn't necessarily the fault of the person doing less work.

9.3.5 Melanie Broersen

Communication and task allocation. Making sure everyone is on the same page, and are all in agreeance on who is doing what task and by when.

If I could do it all over again, I would make sure everyone, including myself, had a personal weekly to-do list set up from day 1. Since we have had these set up on Trello I have been more aware of what needed doing each week, and having group member-specific tasks from the start would have saved a lot of confusion about what we needed to do, and where we were all currently at.

9.4 Hindsight Reflection

What advice would you give to a group about to embark on a similar project?

9.4.1 Casey Edwards

Communication is always the key to any project getting done on time and to a high quality. When it comes to these kinds of tasks, being silent, not contributing and just doing things in your own time is not ideal. It will leave other group members stressed and possibly end up in not only you getting a low mark but also those you have worked with.

9.4.2 Christopher Crawford

Communication is always the biggest part of group work and having good lines of communication. Understanding each group member's strengths and applying those areas to the group's needs. Each group member will have different scenarios in their lives which is difficult to coordinate in online platform and is something to be mindful of in a group project.

9.4.3 Dylan Currie

Ensure all aspects of project work are broken down into multiple deliverables. Structuring work in this way ensures deliverables each week are clearly outlined and monitorable, encouraging continued progress. Development features can be broken down into research & learning, design, coding and testing. The delivery of these four sections can be easily monitored, ensuring progress is made week on week.

Ensure adequate communication expectations are in place at the start of a project, including outlining how progress will be shared and monitored. A good method of this is using Trello card comments to clearly outline what you've worked on, completed or any progress made. Including any helpful resources in these comments can be very beneficial in assisting other team members in their learning.

9.4.4 Ely Hawkins

My advice would be:

- Break your Trello cards down into small manageable tasks.
- Choose your team leader wisely.
- Be open and constant with communication.
- Set clear standards for communication and hitting deadlines etc.
- Assign tasks based on the skill sets of your team members.

9.4.5 Melanie Broersen

Be organised. Get your team set up as soon as possible and agree on meeting times and project task expectations. Also, make sure to identify the best person for the task of team leader as it will impact the project greatly.

10 Marketing Pitch

We at AlphabetEsq believe we have the most effective student collaboration tools available. Our application, BotFly, provides over 10 different unique features to encourage collaboration, set reminders and monitor participation. By integrating with Discord, a collaboration tool already used by many students, our application can be easily used by a wide variety of people. BotFly can be added to existing Discord servers in as little as 30 seconds, requiring minimal coding experience or knowledge. Setup is as easy as accessing the Discord Developer Portal, creating and connecting a new bot to your server, then starting BotFly.

Currently Discord provides no way of monitoring participation or exporting chatlogs - something which can prove problematic when using it for collaboration. Now, with BotFly, students have a way of exporting chatlogs, generating automatic statistics and other unique features such as setting up meeting or deadline reminders. This is just the start though, we have many future ideas in mind to further encourage and aid collaboration on Discord. Our application could be potentially sold for a premium to universities, students or any other groups making use of Discord for collaboration. In Study Period 1 2019, over 50% of Intro to IT students made use of Discord during their studies. With that level of students using Discord for collaboration, the potential market is huge.

11 Skills and Jobs

11.1 Lead Software / Application Developer

Location: Melbourne, Australia
Work-type: Full time
Salary: $80k - $110k p.a.

The Company:

Our company is a unique, well-founded start-up organisation specialising in the development of tools and technologies for student collaboration. Join our company to experience working with a skilled, dynamic team and be a part of the success of a growing organisation. We're looking for an experienced lead software developer to further our ideas, working on unique, exciting projects.

The Position:

The lead software developer/application developer is responsible for developing applications focused on the needs of end-users. As an application developer, you are self-motivated, self-directing and have extensive experience in Node.js, JavaScript and general application development. You have experience designing feature specifications with end-user experience in mind. You have past experience leading successful small software development teams.

Key Responsibilities / Position Duties

- Assist in the planning, design and specification authoring of feature development
- Serve as a mentor and leader for other software developers
- Work in an Agile manner, with a focus on end-user experience

Skills, Knowledge & Experience

- Bachelor's degree in Information Technology or equivalent
- Excellent communication, documentation and teamwork skills
- Extensive experience in JavaScript development using Node
- Innovative thinking in regard to planning, design and specification authoring of feature development
- Past experience in leadership and management of small teams desired
- Self motivated, self directing and innovative
- Understanding of Agile development and working methodologies

Tech Skills

- Node.js / JavaScript
- Discord.js skills desirable, though not essential

11.2 Junior Software Developer

Location: Melbourne, Australia
Work-type: Full time
Salary: $55k - $65k p.a.

The Company:

Our company is a unique, well-founded start-up organisation specialising in the development of tools and technologies for student collaboration. Join our company to experience working with a skilled, dynamic team and be a part of the success of a growing organisation. We're seeking a junior software developer to work with our experienced lead software developer on unique and exciting projects.

The Position:

The Junior Software Developer works under the Lead Software Developer, assisting with everyday development and coding workload. As a junior software developer, you will have a strong willingness to learn and great communication skills - you're always open to constructive feedback, tips and suggestions. You have some experience in application development, preferably Node.js / JavaScript. You have some experience on developing applications to specifications, with a strong focus on end-user experience. Ideally, you have past experience working in small teams.

Key Responsibilities / Position Duties

- Developing minor and medium features to design specifications
- Contribute to maintaining of features including troubleshooting and bug fixing
- Contribute in planning and review meetings, providing insight and updates
- Willingness to learn and grow responsibilities overtime, receiving mentoring and development from senior staff

Skills, Knowledge & Experience

- Bachelor's in Information Technology or equivalent degree
- Experience or knowledge of Node.js / JavaScript development
- Experience using GitHub, Visual Studio Code and Discord.js desirable
- Works well in small teams with a strong willingness to learn
- Understanding of Agile development methodologies desirable, though not essential
- Innovative thinking to create and implement innovative solutions
- Great communication and documentation skills

Tech Skills

- Node.js / JavaScript application development
- GitHub, Visual Studio Code & Discord.js desirable

11.3 Marketing & Design Specialist

Location: Melbourne, Australia
Work-type: Full time
Salary: $65k - $75k p.a.

The Company:

Our company is a unique, well-founded start-up organisation specialising in the development of tools and technologies for student collaboration. Join our company to experience working with a skilled, dynamic team and be a part of the success of a growing organisation. We're looking for a marketing and design specialist to join our team and grow with us while working on many exciting projects.

The Position:

The marketing and design specialist is responsible for producing professional graphics and digital design materials. You will lead the development and maintaining of webpages (including all design requirements) with the assistance of a junior software developer. Your day-to-day will involve monitoring and analysing of marketing statistics including Google analytics, social media reach and any other available data. You will lead our marketing efforts, including maintaining of commercial social media profiles, designing advertising campaigns and design of digital communications. Occasionally duties involve assisting in the recording and editing of videos, and the design of presentation materials such as slideshows.

Key Responsibilities / Position Duties

- Graphic & digital design - logos, digital and social media advertising, website materials and all associated digital communications and design requirements.
- Project management including planning and implementing website updates
- Utilisation of social media platforms for customer acquisition & retention
- Monitoring and analysis of marketing statistics including Google analytics, social media reach statistics and any other relevant data
- Professional design of slideshow presentations
- Recording and editing of videos to meet specifications

Skills & Knowledge

- Experience designing, developing and maintaining websites
- Experience managing and maintaining commercial social media profiles
- Excellent attention to detail and proof-reading
- Effective communication and time management skills
- Understanding of Agile methodologies desirable, though not essential

Tech Skills

- Website design & development (HTML, PHP, CSS)
- Experience using Adobe Creative Cloud suite

11.4 Technical Help Desk / Customer Service

Location: Melbourne, Australia
Work-type: Full time
Salary: $55k - $65k p.a.

The Company:

Our company is a unique, well-founded start-up organisation specialising in the development of tools and technologies for student collaboration. Join our company to experience working with a skilled, dynamic team and be a part of the success of a growing organisation. We're looking for a help desk / customer service officer to assist customers with utilising our exciting new applications!

The Position:

The help desk / customer service officer is responsible for assisting customers with general and technical enquiries regarding our applications. You will be customer-facing and responsible for troubleshooting and resolving a wide range of general issues, escalating bugs for fixes as required. You will demonstrate excellent interpersonal and communication skills, while being widely adaptable and understanding. We're looking for technical expertise combined with the ability to confidently and professionally assist customers.

Key Responsibilities / Position Duties

- Responding to emails, support tickets and calls in a professional manner
- Assist customers with the setup and troubleshooting of applications
- Guide customers through documented troubleshooting steps
- Document and escalate identified bugs or issues to appropriate team
- Answer customer enquiries or concerns, forwarding as required
- Document and forward feedback, suggestions or requests
- Maintain logs and statistics, including statistics on frequency of issues, bugs, requests, feedback and any other relevant statistics

Skills & Knowledge

- Excellent communication skills across all levels of customer proficiency.
- Excellent and thorough documentation skills
- Works well in teams with on-going, effective communication
- Able to run through troubleshooting steps and procedures, assisting customers in the process of troubleshooting issues
- Excellent phone and email etiquette
- Previous experience in a customer-focused role, particularly a help desk or customer service position
- Information Technology knowledge or skills desired, though not essential

Tech Skills

- Everyday computing skills including word processing / typing
- Information technology support skills desired, though not essential

12 Appendix

12.1 AlphabetEsq-BITS-A2.pdf

AlphabetEsq Trello - Assignment Pt. 2 Submission
Outlines full designs of our Discord bot, including diagrams and demonstrations of all minimum viable features.

12.2 AlphabetEsq Trello Board

AlphabetEsq Trello Board
Trello was used for assigning and monitoring of tasks throughout this project. Our team made use of 'to-do', 'doing' and 'done list's to easily monitor tasks. Comments were used to outlined what had been done, any issues experienced, what still needs to be done and any other relevant information.

12.3 AlphabetEsq GitHub

AlphabetEsq GitHub Repo
GitHub was used for collaboration on coding tasks. Our final code along with information on all previous versions and commits is available.

12.4 AlphabetEsq GoogleDrive

AlphabetEsq GoogleDrive
GoogleDrive was used for collaboration on documents and design materials. Currently contains copies of all past documents, including estimations, design documents and reports. Version history outlines individual contributions and document history. An RMIT Google account is required to access.

13 References

Heller, M. 2019, Review: The 10 best JavaScript editors, InfoWorld, viewed 16 Jun 2019, <https://www.infoworld.com/article/3195951/review-the-10-best-javascript-editors.html>.

Gaskins, A. (2019), Code a JavaScript Discord Bot: Part 1, YouTube, <https://www.youtube.com/watch?v=9CDPw1lCkJ8>.

AlphabetEsq 2019a, GitHub Commits - Added basic bot template, GitHub, <https://github.com/Dyl459/AlphabetEsq/commit/3807a34acf3ca296a9f5e2a44f821679daf62e33>.

AlphabetEsq 2019b, GitHub Commits - Added Nag interval, GitHub, <https://github.com/Dyl459/AlphabetEsq/commit/9b6fe19cf9157557f0d172d27f643b0ec039455d>.

AlphabetEsq 2019c, GitHub Commits - Nag filter is working now, GitHub, <https://github.com/Dyl459/AlphabetEsq/commit/25292eda57edd15ba9e19b6c68d9a7e7fc4e4be1>.

AlphabetEsq 2019d, GitHub Commits - Message logger now stores messages in an array, GitHub, <https://github.com/Dyl459/AlphabetEsq/commit/309b923df490d1d410a6036bd1be1f2d0a73b851>.

Discord 2019, Discord.js Docs - Message, Discord.js, <https://discord.js.org/#/docs/main/stable/class/Message>.