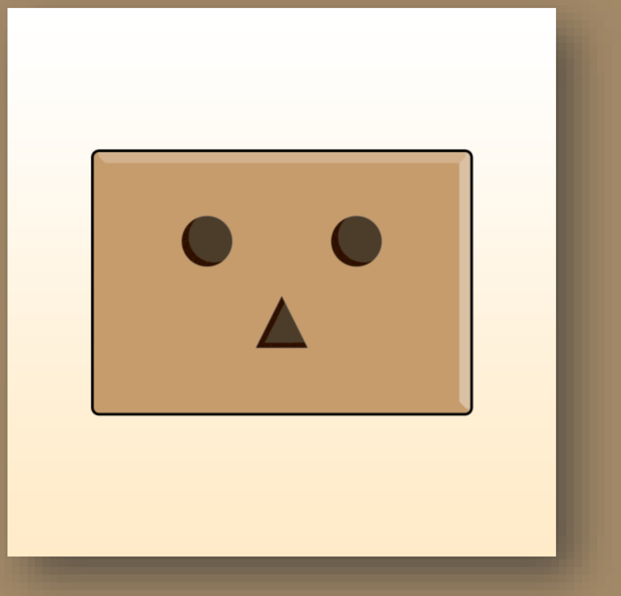

SYSTEM TEST PLAN

COMMANDER

A Discord bot to make server management pain-free

Christian Sauls
Software Engineering
COSC 412-101 @ Towson University

INTRODUCTION

1.1 Definitions and Acronyms

Discord: The popular free proprietary messaging and VoIP application and digital distribution platform for which this project was designed to operate within.

API: Applications Programming Interface. The public-facing endpoint for outside requests to interact with Discord's services.

Bot: A non-human "robot" account created by a regular Discord user to facilitate the automation of certain functions by directly utilizing the Discord API.

Guild or Server: A Discord community space that can be easily created by any Discord user.

Guild Member or Member: A Discord user that is a part of a particular Discord server.

Role: An object that can be assigned to guild members to provide them with a specified set of permissions within that guild. Roles define groups of members within a guild and helps to establish a visual member hierarchy.

Channel: Spaces within a guild that act as containers for certain categories of discussion or other activity. There are different types of channels including text channels, voice channels, and more. For example, an administrator within a server may create a text channel named "videos" with the intent that all videos be posted within that channel. All guilds have a default text channel upon creation with the name "general," and a guild must have at least one text channel at all times. Users can reference channels by prefixing the channel name with "#" (ex: #general).

Commander: The name of this software product. Commander is a Discord bot that interacts with Discord users. Most of its functionalities take place within guilds, as opposed to users directly messaging the bot.

Commander's Playground: The Discord server that was created and used by this project's development team to design, develop and test Commander's functionalities.

1.2 Product Description

Commander is a Discord bot designed to ease the stain on Discord server administrators by offering easy-to-use text commands that carry out otherwise tedious, time-consuming, and complex tasks.

1.3 Objectives

This test plan will ensure complete coverage of all of Commander's functionalities. Testing will be carried out on each command to verify that the desired outcome occurs.

1.4 Constraints and Assumptions

Commander has been developed within a testing environment built specifically for itself: "Commander's Playground." As such, a few of Commander's functionalities rely on specific default member permissions, channels, or roles. Because of this, Commander is not yet ready to offer its full functionality within other guilds, and using these commands will typically result in an error letting the user know that the particular role or channel was not found. If users or roles do not possess the permissions that Commander expects them to, it will either send a reply saying that those permissions are required, or the called function will not behave as expected; for example, if a guild has a "Muted" role and an administrator attempts to use the "mute" command, the targeted user may not be muted if the permissions hierarchy is not in line with Commander's built-in expectations (if that guild does not have a "Muted" role, Commander would reply that it was unable to find a role by that name).

Our next stage of development will be focused on ensuring that these currently-built-in assumptions will be able to be set individually by server owners to accommodate their needs.

With these constraints in mind, please be sure to test Commander's functionalities within Commander's Playground to ensure proper functionality, and to understand how these commands will work globally in the future.

To join Commander's Playground, click on the following link: **https://discord.gg/ZfJGpMU**

1.5 Requirements

For one to test Commander, they first need a Discord account. To ensure that all of Commander's commands can be properly tested, the user must join Commander's Playground and be granted administrative privileges by a developer in the server.

Before Testing

2.1 Enable Developer Mode

Many of Commander's functionalities either require or function optionally with IDs. Each message, channel, guild, user, and more in Discord has a unique ID assigned to it. In order to view these IDs as an end user, you must enable Developer Mode for your account.

The toggle for Developer Mode can be found in the following location:
User Settings > App Settings > Appearance > Developer Mode

With Developer Mode enabled, object IDs are now visible to you by right-clicking on appropriate objects within Discord. You should now be able to see the prompt to "Copy ID" on these objects.

Testing

3.0 Create a Discord Account and Join Commander's Playground

3.1.a automember

Read the message in #how-to-enter, and follow the instructions.
Hint: Type "cardboard" in #enter-here.

You should automatically be assigned the "Member" role, and have access to the rest of the server. This tests Commander's only hidden command – automember – which automatically assigns a specified role when a user enters a secret phrase. You should not be able to re-run this command by typing "cardboard" anywhere else after you have already been given the "Member" role.

3.1.b Obtain Administrator Privileges

For the sake of being able to test all command functionality, have a developer in Commander's Playground grant you administrative privileges.

3.2 !help

In any text channel, type "!help" to execute the help command. Commander should send you a direct message explaining general bot information (including how to run additional commands) and displaying a list of available commands, along with descriptions for what each command does. Commands are categorized into groups representing the base purpose of their functionality.

Additionally, you can type "!help [command name]" (without the brackets) to receive detailed instructions on that command's usage and functionality.

This command satisfies the requirements of offering a help command and full documentation.

3.3 Run All Commands

Run any command of your choosing. Commander should not send any error messages. If you are disallowed from running a command and provided with a reason as to why, that is a normal response; some commands can only be ran by the sole owner of the bot, and some commands may only be ran within guilds.

Go down the list of available commands, executing them to see if its outcome matches what the command description says. All commands are fully self-documented through the "!help" command and do not require any explicit definitions or usage information within this document.

The "Mod" group of commands satisfies the requirement to offer commands that facilitate administrative actions such as kicking members, muting members, banning users, deleting messages in bulk, and more. The commands in this group are privileged, meaning that the user executing the command must have the appropriate guild permissions or else an error is shown to the user.