# Discord Bot

## Preface

Discord is a popular social platform that is aiming to replace the likes of Skype, TeamSpeak, Ventrilo and the likes.

It allows the ability to create your own custom **servers** with a range of functionality to offer the users you might invite.

One of those particular functionalities is to add bots to your server that the users can interact with.

Usually these bots do simple things such as playing music, assigning roles to members, deleting messages and so forth.

Discord released an API to create Bots, usable in any language.

From that, many developers have created libraries to interact with this API to make the process even simpler.

## Description

1. Create a discord server
2. Create a discord bot
3. Connect your bot to the server
4. Give your bot the ability to read and write messages
5. Create “Commands” that the bot recognises, and performs an action based on that command.
   1. e.g. **!bot hw**
      1. BotName: Hello World!

Once this is done, your goal is to hook up your previous projects that required input, which normally might have come from an interface or command line, and let the bot be that interface instead, running your projects and outputting the results via messages.

## Expectations

1. Break up your bots functionality into separate classes, one class per command.
   1. Look into ***discord.js-commando***
2. Checking for the specific keywords for each command should not be a bunch of conditional statements, figure out a way of making this a scalable bot.
   1. ***dDiscord.js-commando***
3. At least 5 of your projects integrated into your bot.
4. Integrate at least one API into your bot
   1. Some example ones; <https://www.reddit.com/r/webdev/comments/3wrswc/what_are_some_fun_apis_to_play_with/>
5. **Mocha** installed, unit tests created for each of your projects that you have integrated with your bot.
6. Test coverage set up with codacy, minimum of 50% coverage.

## Part 2

By now you should have a good grasp on issuing commands to your bot.

Team up with other another trainee(s) and host a server where both your bots can reside.

**Try and break each other’s bots**

1. No spamming
2. No physical tampering (Shutting down their bot, editing their code)

Whenever you find a vulnerability in another trainee’s bot, note down the information around it

***Bots owner:***

***Bot command/application:***

***Reproducible steps to break it:***

***How it was identified as broken (Screenshots will suffice):***

***Owner fixed vulnerability(y/n):***

## Deliverable(s) – In order of priority

1. A Github repository link emailed to your trainer, containing your discord bot and your unit tests.
2. A codacy link emailed to your trainer, containing the default JS static analysis that codacy provides for your discord bot repository, as well as test coverage set up.
3. A document detailing all the vulnerabilities you found in other trainee’s bot(s)
4. A document detailing all the vulnerabilities other trainees found in your bots, and how you fixed them.