Group 3:
Nick Sabia - sabian@my.ccsu.edu
Maciej Bregisz - mbregisz@my.ccsu.edu
Jack Gola - jackgola@my.ccsu.edu
Kobe Onye - kobe.onye@my.ccsu.edu
12/01/21
CS 417

Final Project Submission

How to configure and run the project

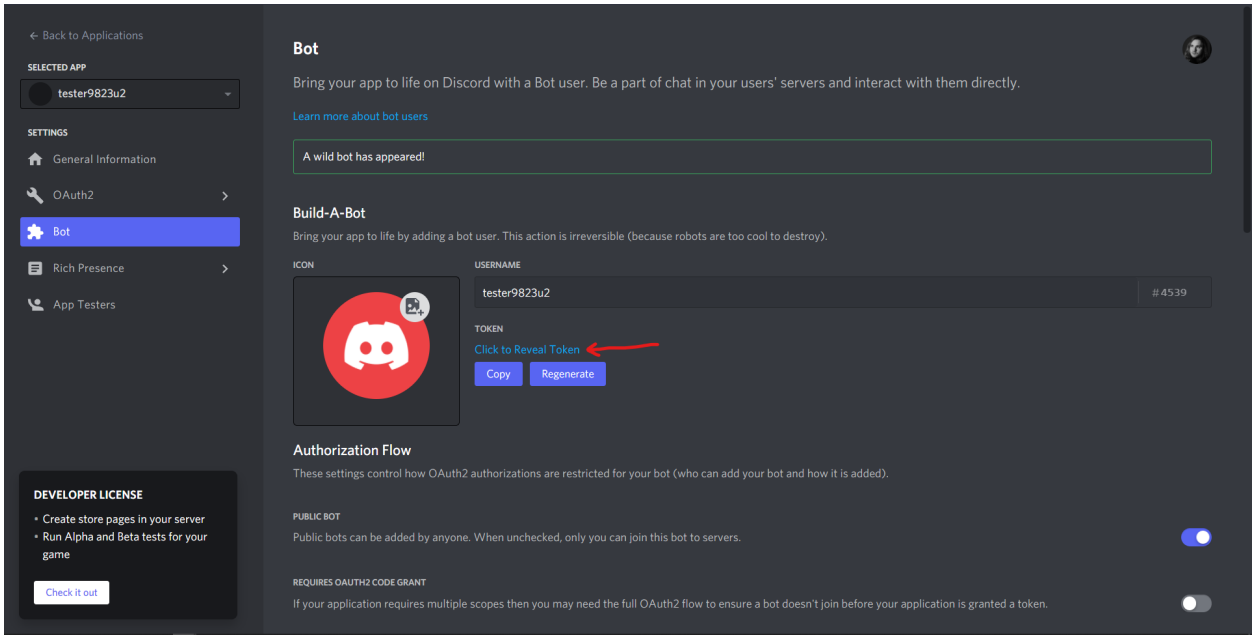
We've packaged the code into a .jar file which will be able to run on Windows and Linux. You can download the release from our group 3 github repo. The file to download is called **racingbot.zip**
https://github.com/CCSU-DesignPatterns-F21/final-project-f21-group3/releases/tag/1.0.1

Since this is a Discord bot, you will need to create a new Discord bot (and you'll need a Discord account). Also, we use MongoDB for a database. All install instructions are outlined below.

To create a discord bot

1. Go to https://discord.com/developers/applications. You may need to sign up for or log into a discord account to see this page.
2. Create a new application.
3. Give the bot a unique name. Ex. "Racing Bot 62948272". These names are like usernames, so if you find this name is already taken at step 5, you'll have to rename it.
4. While on the page where you can see and edit the newly created bot's details, navigate to the bot tab.
5. Click "Add a Bot"

6. Once you land onto the bot details page, **copy the secret token** and keep it on hand.

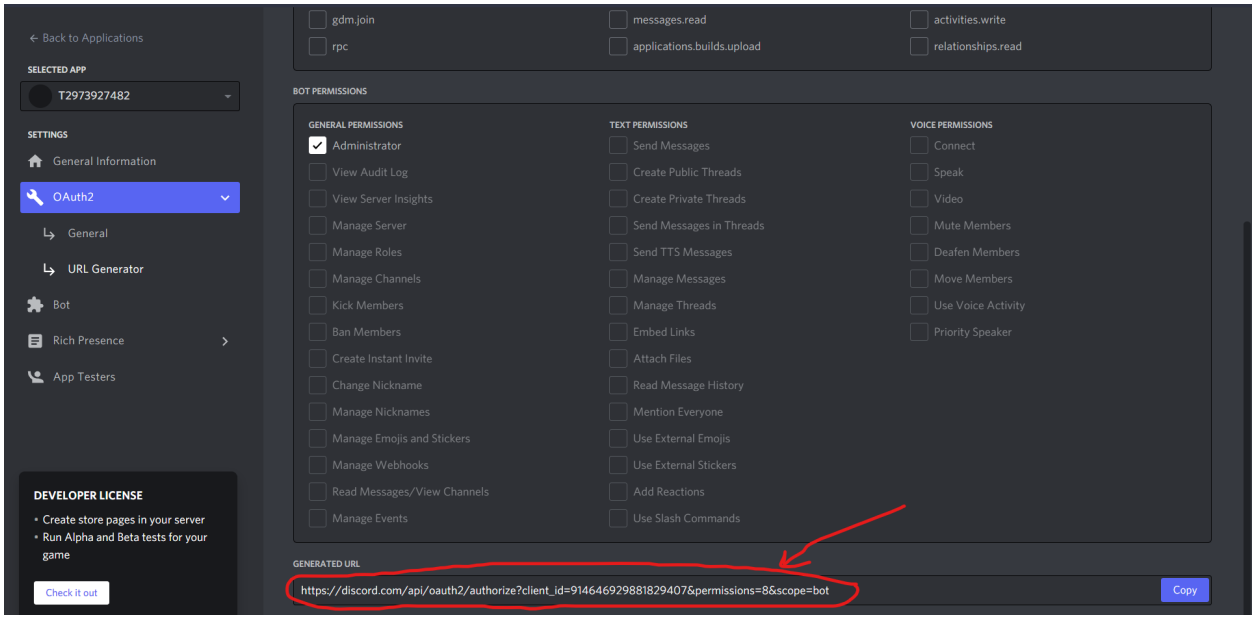


7. Still on the bot details page, scroll down where you see bot privileges. For simplicity, give the bot administrator privileges.

Adding the discord bot to a server

1. Open up Discord and log in. If you don't have it on your system, download and install it here: https://discord.com/
2. Create your own server
3. Navigate back to the https://discord.com/developers/applications page and click on your bot.
4. Navigate to OAuth > URL Generator
5. Check "Bot" under scopes and then check "Administrator" under Bot Permissions.

6. There should be a link that has appeared at the bottom of the page. Copy it and visit it:



7. When you visit that link, you can add the bot to the server of your choice.

What to do with the discord bot secret token

Our code will first look for an encrypted file called AppConfig with all of our keys and other secrets. If it cannot find it, then it will fall back to using the unencrypted config.properties file.

Encrypted version

We don't have direct access to group 1's encryption software, but we supplied an example xml file with the correct structure needed so that when it's encrypted using their software it will work with our bot.

1. Rename the AppConfig.xml.example (***/project-root/AppConfig.xml.example***) to AppConfig.xml. This is where all necessary keys and database info will go.
2. Open up AppConfig.xml and paste the discord secret token into there.
3. For the remaining fields, we need mongodb installed. Instructions on that are found in the next section.

Unencrypted version

1. Rename config.properties.example (***/project-root/config.properties.example***) to config.properties. This is where all necessary keys and database info will go.
2. Open up config.properties and paste the discord secret token into there.
3. For the remaining fields, we need mongodb installed. Instructions on that are found in the next section.

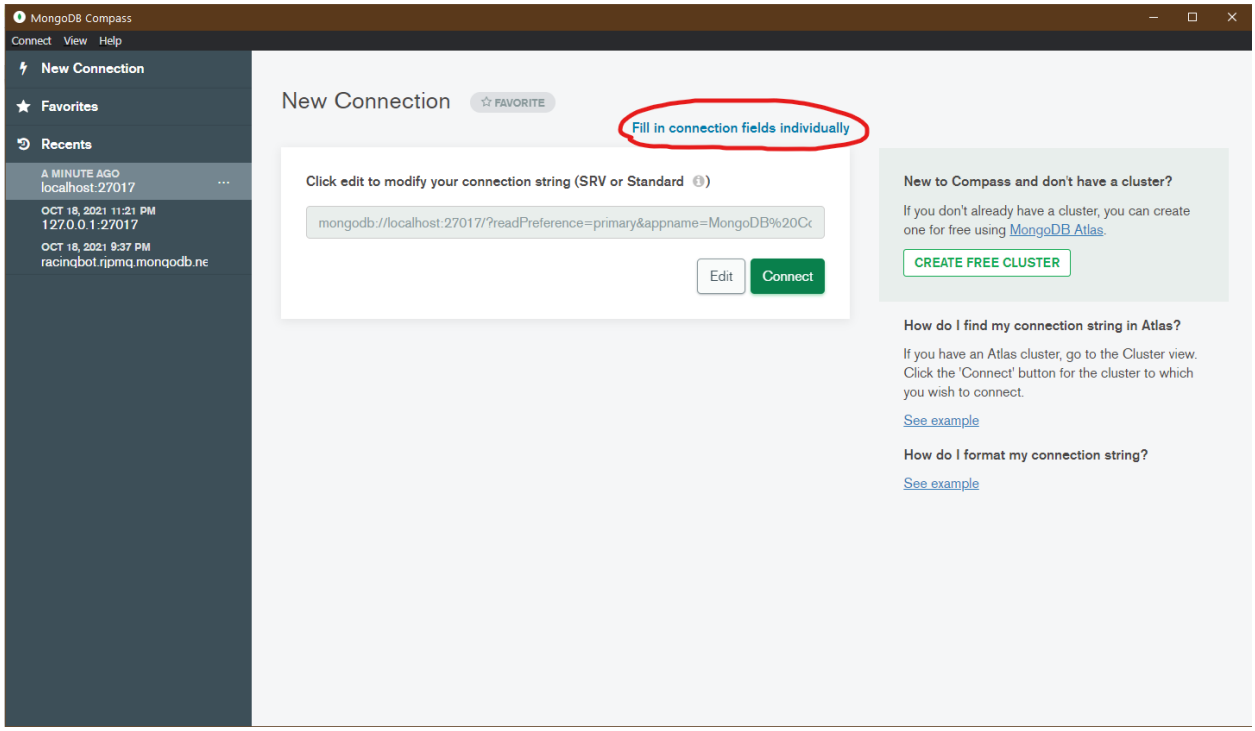
Now in order to use the bot, you must install MongoDB.

Setting up MongoDB for Windows (https://docs.mongodb.com/manual/tutorial/install-mongodb-on-windows/)

1. Download MongoDB https://www.mongodb.com/try/download/community?tck=docs_server
2. Run the installer and complete the setup wizard.
    a. Service Configuration: Make sure that “Install MongoD as a Service” is checked.
3. As a note, you may need to add the bin folder from the mongodb folder into your PATH. (

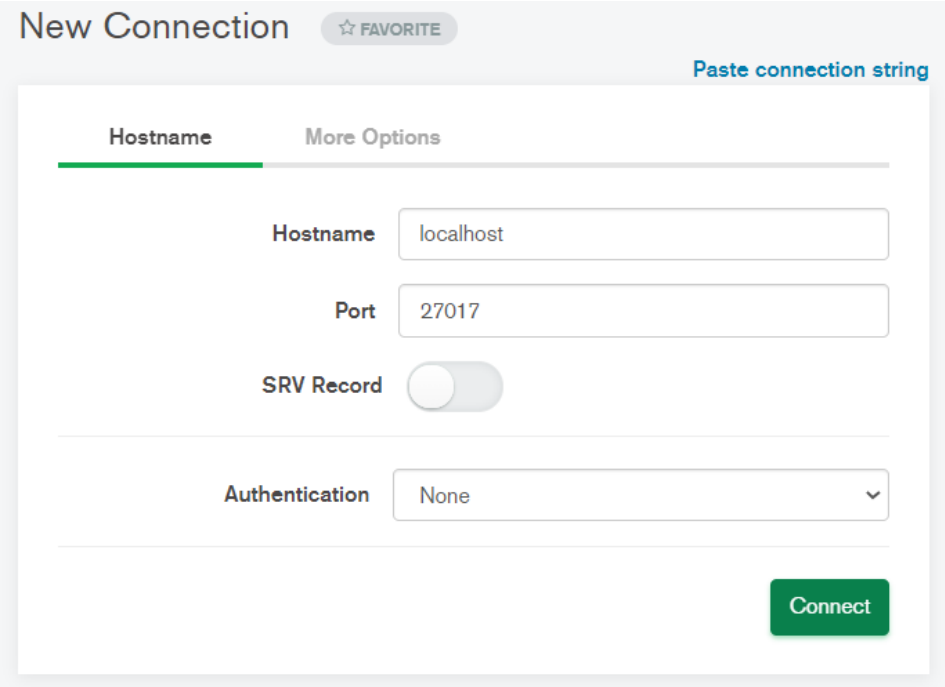
Install mongo compass and create the database

1. Download mongo compass here: https://www.mongodb.com/try/download/compass
2. If you’ve downloaded as a zip, extract the files then run MongoDBCompass.exe.
3. You should see this screen:



Click “Fill in connection fields individually”

4. Fill out the fields as shown here, with the hostname as localhost and the port as 27017, then click connect.



5. Create a new database (call it RacingBot) and add three collections: Users, Shops, and Events. Add the name of the database to config.properties.

NOTE: Since you've started a local server without authentication, you will leave *mongoDBUsername* and *mongoDBPass* blank.

Run the Bat

With Mongo Compass running, the discord bot credentials placed into the config.properties/AppConfig file, we can now run the bot! Within the racingbot.zip from the release in github, there's the **start.bat** file which you can use to start up the bot.

Patterns used and their associated classes

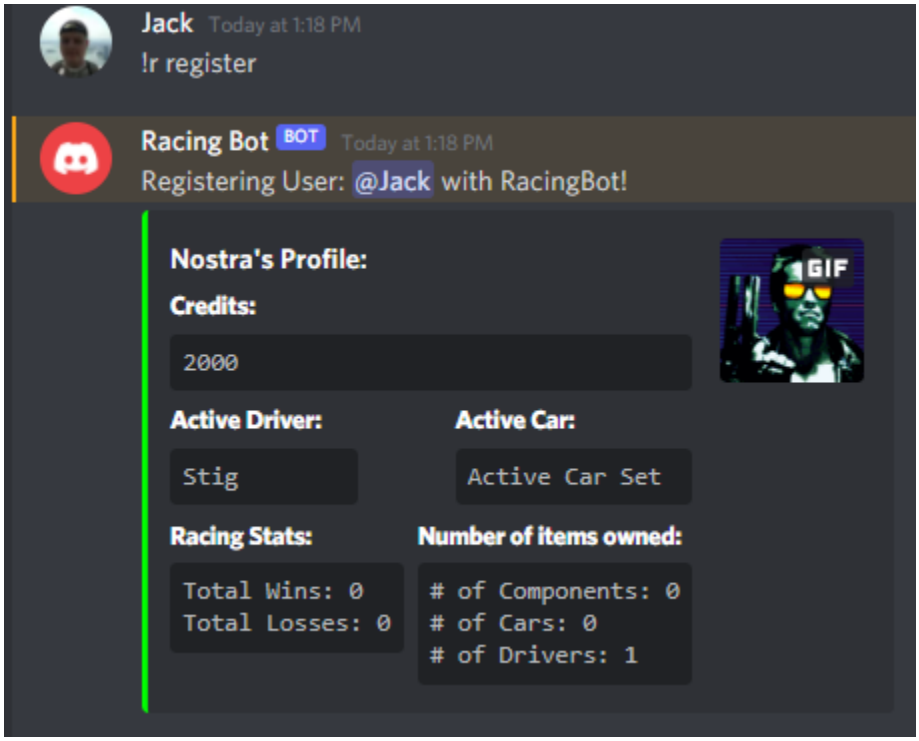
- Singleton
  - ConfigPropertiesHandler
    - AppConfig
    - Graduate Team Module Package: DesignPatterns.utils
  - DBHandler
- Observer
  - GameplayHandler
  - CustomObserver
- Abstract factory
  - Component

    - ChassisComponent
    - TransmissionComponent
    - EngineComponent
    - SuspensionComponent
    - WheelComponent
    - ConcreteComponentFactory
    - ComponentFactory
- Chain of Responsibility
    - RaceTrack
    - TrackNode
    - StraightNode
    - CornerNode
- State
    - Driver
    - DriverState
    - Resting
    - Training
    - Racing
        - Aggressive
        - Normal
        - Defensive
        - Crashed
    - RacePending
    - Completed
        - DNF
        - FinishedRace
        - FinishedTraining
- Iterator
    - Iterator
    - ConcreteIterator
    - Inventory<T extends Unique>
- Decorator
    - Iterator
    - IteratorDecorator
    - QualityFilter<T extends MaterialFilterable>
    - DurabilityFilter<T extends MaterialFilterable>
    - ValueFilter<T extends MaterialFilterable>
    - WeightFilter<T extends MaterialFilterable>
    - ComposureFilter<T extends SkillFilterable>
    - AwarenessFilter<T extends SkillFilterable>
    - DraftingFilter<T extends SkillFilterable>
    - StraightsFilter<T extends SkillFilterable>
    - CorneringFilter<T extends SkillFilterable>
    - RecoveryFilter<T extends SkillFilterable>

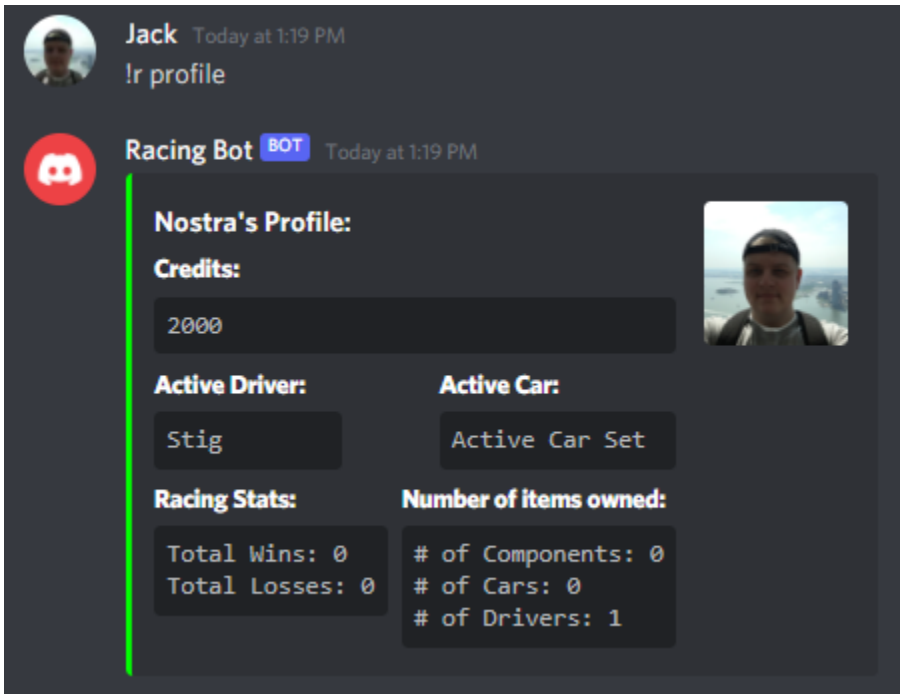
    - ComponentTypeFilter<T extends ComponentFilterable>
- Prototype
    - ICloneable
    - Car
    - Component
- Builder
    - CarBuilder
    - Car
- Strategy
    - Standings
    - SortStandings
    - DriverStandingsQuickSort
    - DriverStandingsMergeSort

The Racing Bot in Action

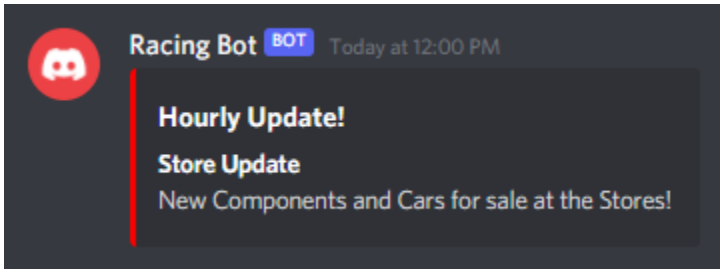
User registering



User Profile



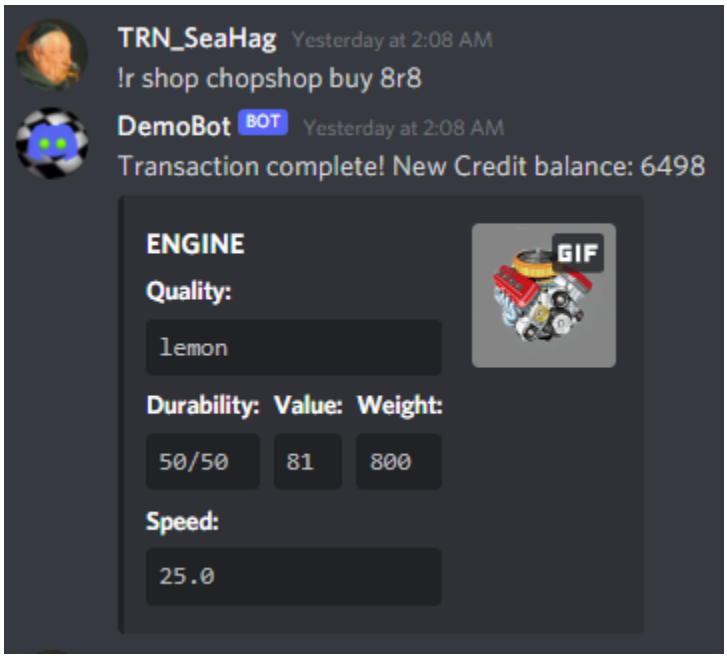
Hourly shop update



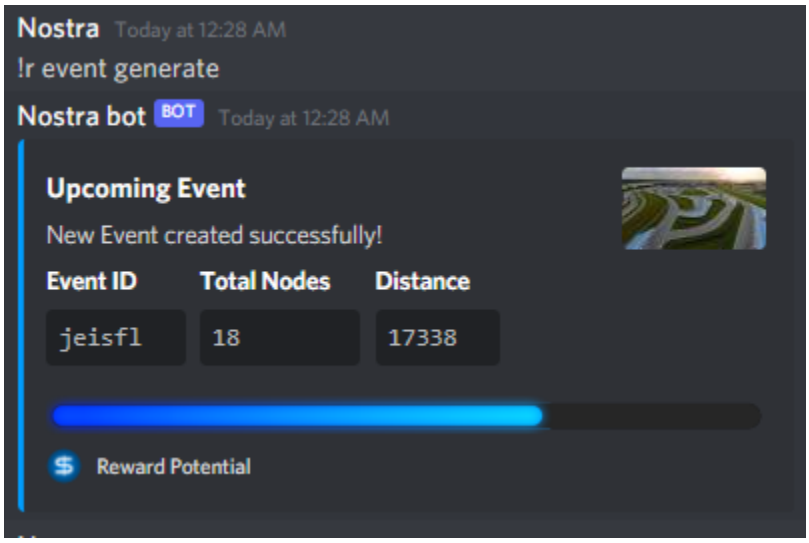
Shop View


DemoBot BOT Yesterday at 2:07 AM
Chop Shop
You never know what you will find at the Chop Shop, stolen catalytic converters, wheels, rims and more! No questions asked, no refunds!
ENGINE
ID: 8r8
ENGINE
Quality: LEMON
Value: 81
Durability: 50
Weight: 800
Speed: 25.0
WHEELS
ID: uwl
WHEELS
Quality: LEMON
Value: 140
Durability: 50
Weight: 800
Braking: 25.0
SUSPENSION
ID: 435
SUSPENSION
Quality: LEMON
Value: 99
Durability: 50
Weight: 800
Handling: 25.0
CHASSIS
ID: ean
CHASSIS
Quality: LEMON
Value: 61
Durability: 50
Weight: 800
Popularity: 1.0
Acceleration Modifier: 1.0
Speed Modifier: 1.0
Handling Modifier: 1.0
Braking Modifier: 1.0
TRANSMISSION
ID: 1vh
TRANSMISSION
Quality: LEMON
Value: 62
Durability: 50
Weight: 800
Acceleration: 25.0
Cars for sale:
r4s
id: r4s | Durability: 50 | Value: 454 | Quality: LEMON | Weight: 4000

Purchasing individual component from Shop

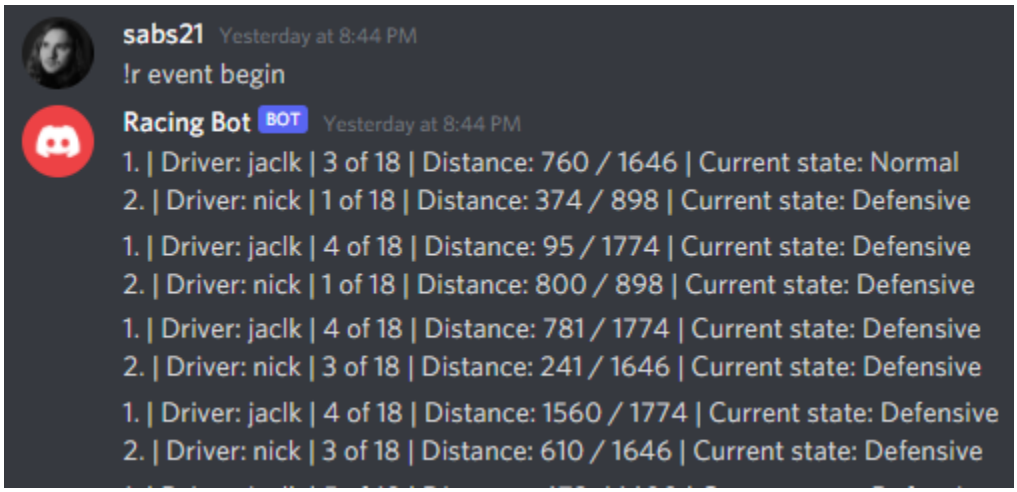


Race event being generated

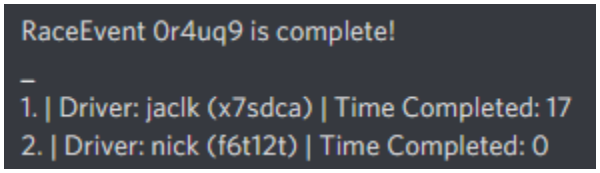


Event view

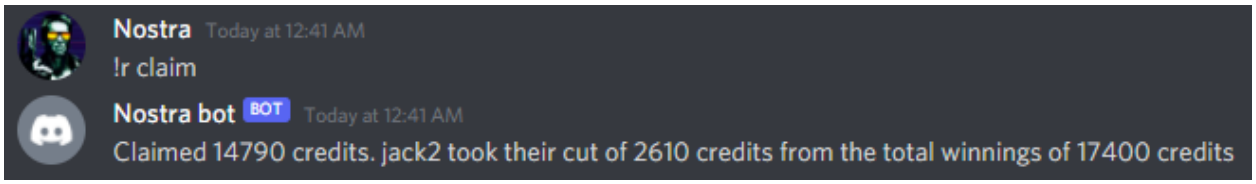
Race in progress (text)



Race end (time completed is 0 because of “Did not finish”)



Claiming credits after race is over



Repairing components after race