JobBot Discord Bot

First Week:

I spent the first week creating the Discord Bot, creating a bot.py, creating a virtual environment, and working on setting up the job scraping part of it. All of this was pretty self-explanatory. Creating the bot was just following instructions on the Discord Developer Portal to create it. I created a Discord server to invite it too and that was for the creation of the bot. The bot.py file has all the commands the bot can respond to. I set up a set preference and get preferences command. I also created the database which wasn't crazy hard since it was a single table to hold the user's preferences.

Second week:

The stuff that took the most time so far is trying to figure out how to get the bot to send job openings and scrape the job site. At first my plan was to scrape jobs from both indeed and LinkedIn. Unfortunately, after failing to get Indeed to work, I focused only on just having it scrape LinkedIn. After a lot of trial and error, I was able to make some progress with the bot scraping the job listings. I was able to get it to use the !find_jobs command to search for jobs based on the users' set preferences. I also implemented a way for the user to specify a job title and location by running the command !find_jobs "Job Title" "location". This took the most time out of the whole project because I tried finding different tools on Google that might help me, but they were all not very helpful. After getting the commands to work, I realized it is only pulling the first 5 results of the job search so each time you would run !find_jobs, it would be the same results. I ended up having it pull all the job results and randomly select 5 of them to send in Discord.

All the packages in the code and where I used them:

Built in packages:

os: I used this to load the environment variables specified in the .env file.

time: I used this to pause the execution of scraping the jobs, so it waits for LinkedIn to load completely

random: I used this to select 5 random job listings from the ones it scraped to avoid repetition.

re: I used this to see if the jobs scraped matched the users' preferences. (Specifically, in is_location_match() statements)

json: I used this to store the scraped job data in a JSON format.

Discord Bot Packages:

discord: This handles the bot commands, messages, embeds, and sending the job listings.

discord.ext.commands: Defines the bot commands (for example @bot.command() async def find_jobs(ctx)).

Web Scraping Packages:

selenium: Opens LinkedIn job search, waits until it loads, and extracts the job postings

selenium.webdriver.edge.service.Service: Loads the msedgedriver.exe to open LinkedIn in a browser window.

selenium.webdriver.common.by.By: Finds job listings, titles, companies, and locations.

selenium.webdriver.support.ui.WebDriverWait: Ensures that job listings appear before scraping

bs4: Scrapes LinkedIn job postings after Selenium loads the page.

Database:

psycopg2: Saves the user's preferences into the PostgreSQL database and fetches them when !find_jobs is used.

Environment Variable Package:

dotenv: Loads the LinkedIn authentication cookie and database credentials

Problems I ran into and what I learned:

Some issues I ran into were mainly with the !find_jobs command. At first when I was trying to do indeed, it would show me jobs not related to any of the preferences I set. I never was able to fix that until I completely ditched Indeed. Another error I found was that when the bot sent the results, it would send the job title twice instead of once, I found the issue being that the job title being a clickable link, it was getting added twice in the results. I fixed this by not putting the job title in the embed title and just moved it to the description of the embed box. Not really ideal design wise but it works.

Another issue I was finding is for the longest time, my scraper would return No jobs found every time I would run the command. I couldn't figure it out until I realized that my code was searching for an exact match for location. Since my preference was a small town, there weren't any jobs located there so it would find jobs, see that the location didn't match and responded with unknown. I fixed this by having it not match the job location

with the preferences and instead just pull the results that it gave from the search, this way it would give the user results.

I learned that I shouldn't get frustrated when the code doesn't work and calm down, collect myself, and try again. The more it doesn't work, the more I get frustrated, and the more I mess up the code.