# Setting Up Programming Environment

## Step 1: Python & VS Code

### Downloading Python

Download the latest version of python on your computer.

**Link for Latest Version:** https://www.python.org/downloads

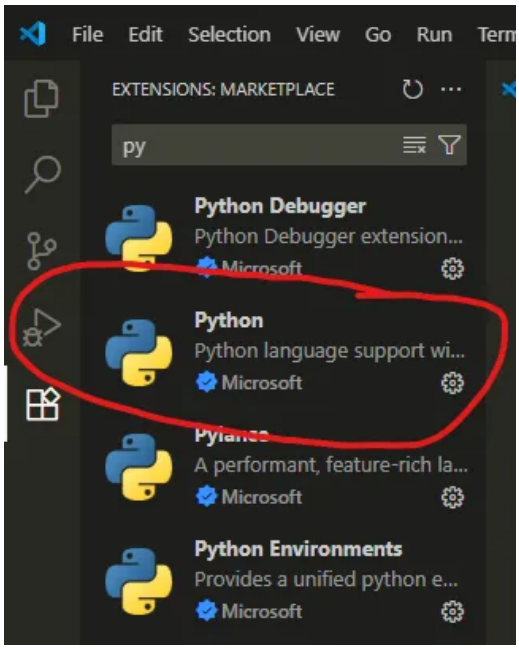
During installation make sure to enable “Add to PATH” as well as remove the character limit.

### VS Code & Python Extension

Download the VS Code editor onto your computer and set up the python extension to begin development.

**Link for Latest Version:** https://code.visualstudio.com/

Make sure to install the Python Extension (Microsoft) from the Extensions Marketplace.




# Step 2: HuggingFace

## Sign Up & Repository

Sign Up and create an account then follow the link to the LLaMA 3 repository. Scroll down and read all the terms and conditions then fill out the information to gain access to the repository.

**MAKE SURE TO USE *"Union College of Union County, NJ"* AS THE ORGANIZATION**

It may take 1-2 hours for you to be granted access.

**Website:** https://huggingface.co

**LLaMA 3 8B Repository:** https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct

## Getting A Token

**Tokens:** https://huggingface.co/settings/tokens

Make sure to keep your Access Code saved somewhere secure.

# Step 3: GitHub & Setting Up

## Sign Up & Repository

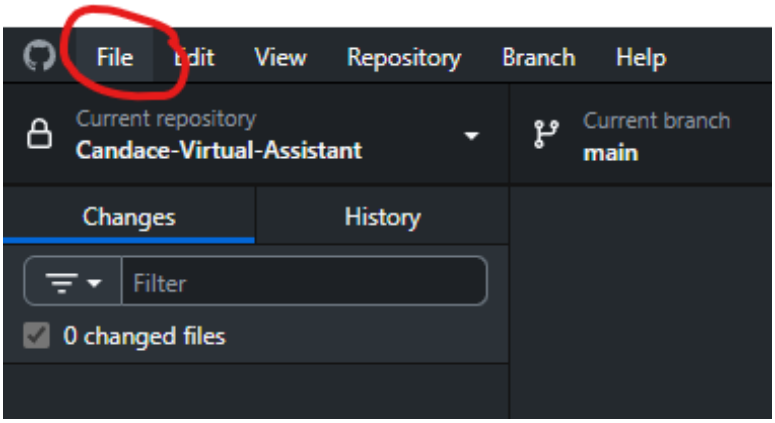
Sign Up and create a GitHub account. Once doing so contact Miguel to be given access to the repository.

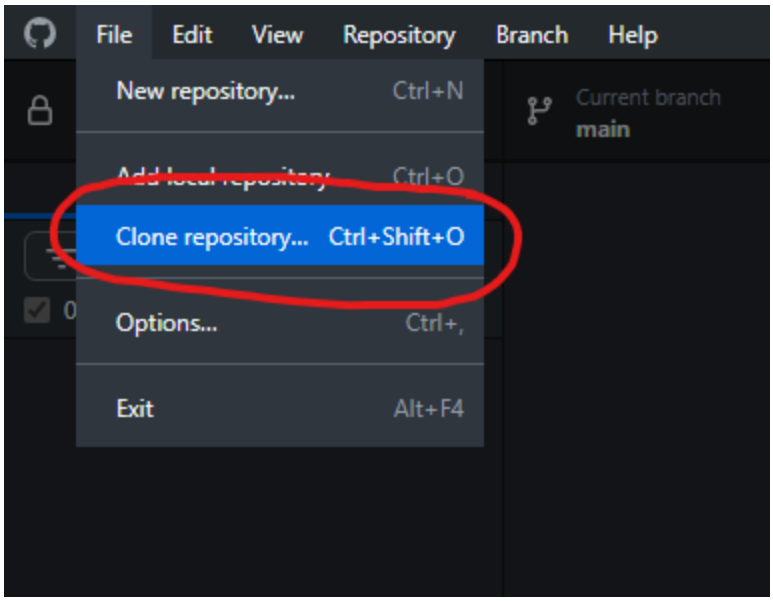
**Website:** https://github.com

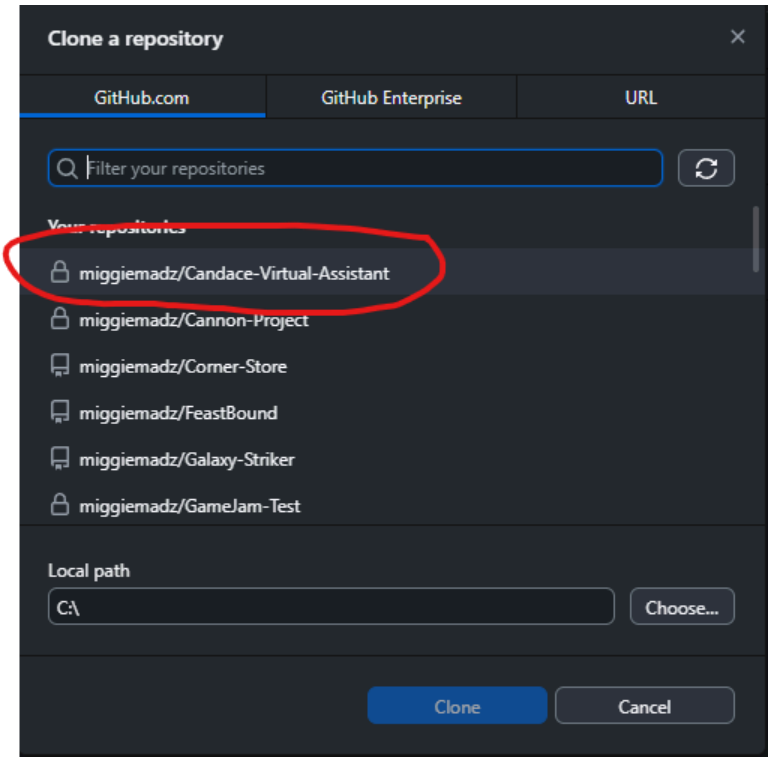
**Contact Email:** miguel.torres@owl.ucc.edu

## GitHub Desktop & Cloning the Repository

Download the GitHub Desktop app on your computer and sign in using your GitHub account. Once doing so you will be able to clone the Candace-Virtual-Assistant repository into a local one to work on. To do so follow these specific steps:






# Step 4: Virtual Environment

After getting the project folder onto VS Code it is now time to set up the Virtual Environment (venv). A venv allows all your work to be saved locally and keeps all

dependencies isolated to your version of the project. This way there aren't any version or permission issues when working as a team.

## Script Permissions

Lets start with enabling permissions for scripts so that there aren't any issues when trying to run the many libraries we may need. Open Windows PowerShell as an administrator and allow it to make changes to your device. Once in it type the following command:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

- **RemoteSigned**: this condition means that you can write scripts locally and they would run no problem
- **-Scope CurrentUser**: this condition means it only applies to your personal project.

## Venv in VS Code

Now open the VS Code Terminal inside your python folder. This can be done by clicking the Terminal tab → New Terminal or you could use the Ctrl + Shift + ` shortcut. This terminal is where the rest of the setup will be done.

In order to create your venv type in the following command into the terminal:

```
# Navigate to your project folder first
cd path/to/project

# Create a virtual environment
python -m venv venv

# Activate it
# On Windows:
venv\Scripts\activate
# On macOS/Linux:
source venv/bin/activate
```

### Notes

- Do not type “path/to/project”, locate the actual project folder that was cloned. It will not be the *Candace-Virtual-Assistant* folder but the *Candace Virtual Assistant* folder. The first one is the GitHub repo’s root folder while the second one is the actual project folder found within the root folder.
- Make sure to type out each command individually of one another to avoid any conflicts, wait for the previous one to complete first.

## Updating Python’s Pip

Making sure that pip is up to date before installing packages to ensure there aren’t any issues. Type the following command into your terminal:

```
python -m pip install --upgrade pip
```

## Adding Libraries

There are a couple libraries you will need in order to get the full use out of LLaMA 3 and Python. In your terminal you are going to type the following commands one after the other:

### PyTorch Ecosystem

```
pip install torch torchvision torchaudio --index-url https://donwload.pytorch.org/whl/cu121
```

### HuggingFace Ecosystem

```
pip install transformers
pip install huggingface_hub
pip install accelerate
pip install sentencepiece
```

***MAKE SURE TO TYPE EACH COMMAND INDIVIDUALLY ONLY AFTER THE PREVIOUS ONE HAS FINISHED INSTALLING***

### Library Uses

- **torch (PyTorch core)**: A deep learning framework. Lets you build, train, and run neural networks like LLaMA 3. This is the foundation everything else sits on.
- **transformers**: HuggingFace's main library for using pretrained LLMs like LLaMA 3. It handles downloading models, tokenizing text, generating responses, and fine-tuning if needed.
- **huggingface_hub**: Lets you connect to the Hugging Face Model Hub where LLaMA 3 lives. Handles downloading models/datasets, uploading your own models, and managing authentication with your Hugging Face account.
- **accelerate**: A utility to make running models faster and easier across different devices (CPU, GPU, multiple GPU's). It helps you train or use models without writing complex low-level device management code.
- **sentencepiece**: A text tokenizer library used by many LLMs. It breaks raw text like "Hello world" into smaller chunks (tokens) that the model can actually understand. Without this, LLaMA 3 couldn't process text properly.

## Flask

Flask is our framework for developing the frontend of *Candace's* sandbox/prototype. Installing it is simple and similar to the previous libraries. In the VS Code terminal type the following commands separate from one another:

```
# The base flask framework
pip install flask

# An extension of flask that allows for future integration with JavaScript
pip install flask-cors
```

### Additional Extensions

- **flask-snippets**: makes debugging and running a lot easier.

## SQLite

SQLite is the DBMS that we will be using as it is extremely simple and lightweight. Luckily there aren’t any needs to install or download anything as Python’s basic library has direct support for SQLite.

## Additional Extensions

- **SQLite Viewer**: Allows us to open .db files directly inside VS Code.