

# Contents

Contents.....	1
AI Voice Assistant Based on XiaoZhi AI.....	2
About the Project.....	2
Cautions .....	2
Disclaimer .....	3
Freenove ESP32 S3 Display.....	4
Hardware Interfaces.....	4
XiaoZhi AI Firmware.....	5
Installing Python (Required).....	5
Firmware Uploading .....	12
ESP32 S3 WROOM Network Configuration.....	20
XiaoZhi AI Server Configuration .....	22
XiaoZhi AI Code .....	29
Visual Studio Code .....	29
Windows .....	29
Mac .....	32
Linux .....	33
Installing ESP-IDF V5.3.2 .....	36
Code Downloading .....	42
Configure Code Environment.....	44
Code Compilation.....	52
Local Server.....	54
Disclaimer .....	54
Deploying XiaoZhi AI on a Local Server.....	54
Installing Ollama.....	54
Installing Conda.....	69
Deploying Virtual Environment.....	84
Deploying xiaozhi-esp32-server.....	86
Visiting xiaozhi-esp-server via ESP32S3.....	96

# AI Voice Assistant Based on XiaoZhi AI

This project applies the Freenove ESP32 S3 Display to implement an AI voice assistant, which requires a certain level of programming proficiency as well as familiarity with ESP-IDF and open-source large models.

## About the Project

This voice assistant project (<https://github.com/Freenove/xiaozhi-esp32>) is derived from the open-source project (<https://github.com/78/xiaozhi-esp32>). It enables the invocation of most mainstream large language models (LLMs) on embedded devices and achieves voice conversation functionality through multiple services, including Voice Activity Detection (VAD), Automatic Speech Recognition (ASR), Speech-to-Text (STT), Text-to-Speech (TTS), Memory Storage, and Intent Recognition. Freenove has adapted this project for its Media Kit product. This article will explain how to run the project on the Media Kit.

There are two ways to run this project – online or offline.

1. **Online:** Connected to the xiaozhi.me server, currently available for free trial to individual users.
2. **Offline:** All the aforementioned services (VAD, ASR, STT, TTS, Memory, Intent Recognition, etc.) must be deployed locally on a personal computer. The user experience depends entirely on the selected models and the performance of the local machine. The local server project (<https://github.com/Freenove/xiaozhi-esp32-server>) is derived from the open-source project (<https://github.com/xinnan-tech/xiaozhi-esp32-server>).

For users who prefer AI assistants, we recommend using the online version.

For developer-oriented users, you can try deploying the offline version to gain a deeper understanding of the various services required for an AI assistant. However, it's important to note that personal computers may struggle to run all these services simultaneously—especially the core LLM (Large Language Model) service—which could result in a poor AI assistant experience. Therefore, the offline version is primarily valuable for learning and research purposes.

## Cautions

### ● Project Copyright:

- Voice Assistant Project: Originally developed by "Xiage", this project was forked and adapted by Freenove for the Media Kit, released under MIT License.
- Local Server Project: Originally created by "xinnan-tech", this project was similarly forked and adapted by Freenove for Media Kit integration, licensed under MIT License.

### ● Supported Countries and Regions:

#### ● Online Version:

Service availability is determined by xiaozhi.me server coverage, which may exclude certain regions. For current supported areas, please refer to: <https://xiaozhi.me/login?redirect=/console/agents>; User experience is directly affected by server connectivity quality. Poor network conditions to xiaozhi.me servers may degrade performance.

- **Offline Version:**

Fully location-independent, with deployment possible in all countries and regions without geographical restrictions.

- **Supported Languages:**

- Online Version: Currently supports Mandarin Chinese, Cantonese, English, Japanese, Korean, and others. If you do not speak these languages, you may not be able to communicate effectively with XiaoZhi AI.
- Offline Version: Depending on the ASR model you deploy. The default FunASR model only supports Mandarin Chinese, Cantonese Chinese, English, Japanese and Korean.

- **Pricing:**

- Online Version: Currently, xiaozhi.me provides free services, but we cannot guarantee that the online server will remain free indefinitely.
- Offline Version: Among the sub-services mentioned, some are paid while others are free—your choice determines the cost.

- **Seeking Help:**

If you have followed the tutorial and still encounter issues, please contact us at [support@freenove.com](mailto:support@freenove.com)

**Note:** Since the online service is provided by xiaozhi.me, if xiaozhi.me discontinues its service, we will also remove related documentation, tutorials, and code.

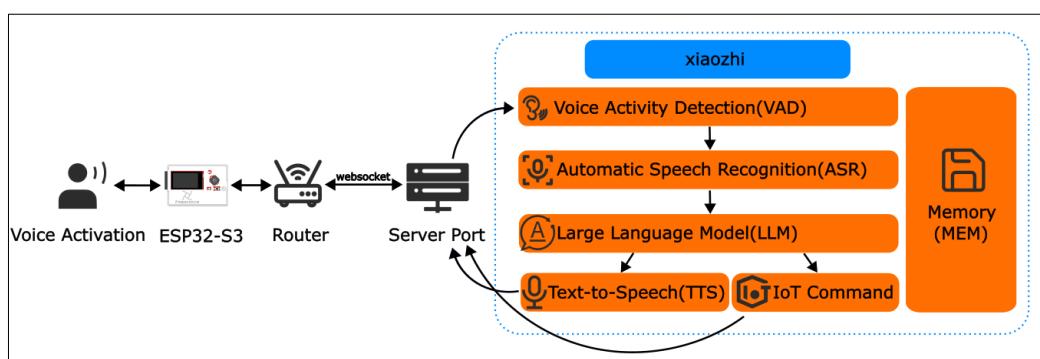
## Disclaimer

This implementation is an adaptation of the open-source project available at <https://github.com/78/xiaozhi-esp32>, provided for third-party learning and AI functionality testing purposes, without any promotion or support for commercial applications. This tutorial is intended solely for personal learning and development by technology enthusiasts.

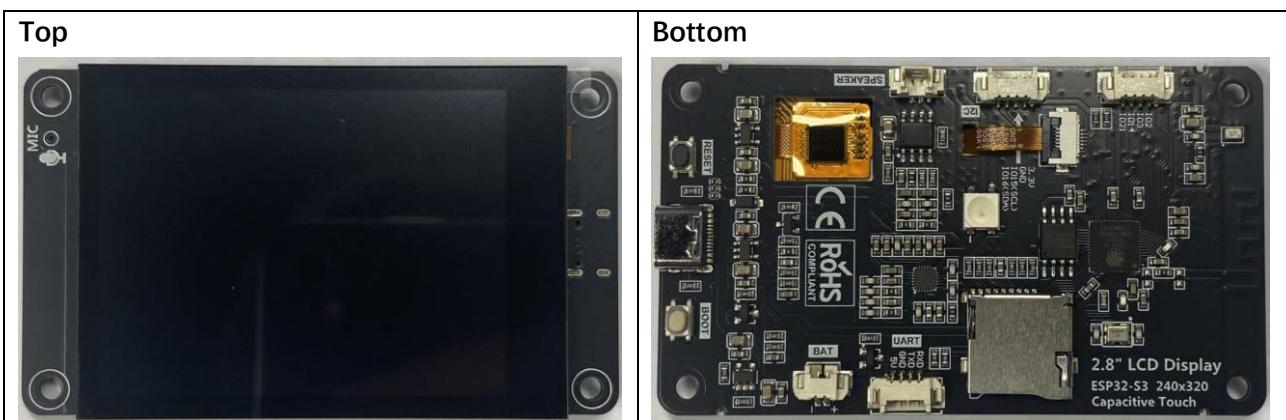
Notes:

1. As this is a third-party open-source project, if you encounter issues during your learning process, please submit an issue to the original repository: <https://github.com/78/xiaozhi-esp32/issues>
2. Currently, XiaoZhi AI only supports Mandarin Chinese, Cantonese, English, Korean, and Japanese for speech recognition. Other languages are not yet supported.
3. The XiaoZhi server interface currently supports English, Chinese, and Japanese only. Additionally, mobile registration is only available for users in the following countries (see the table below). Users from other countries cannot register yet.

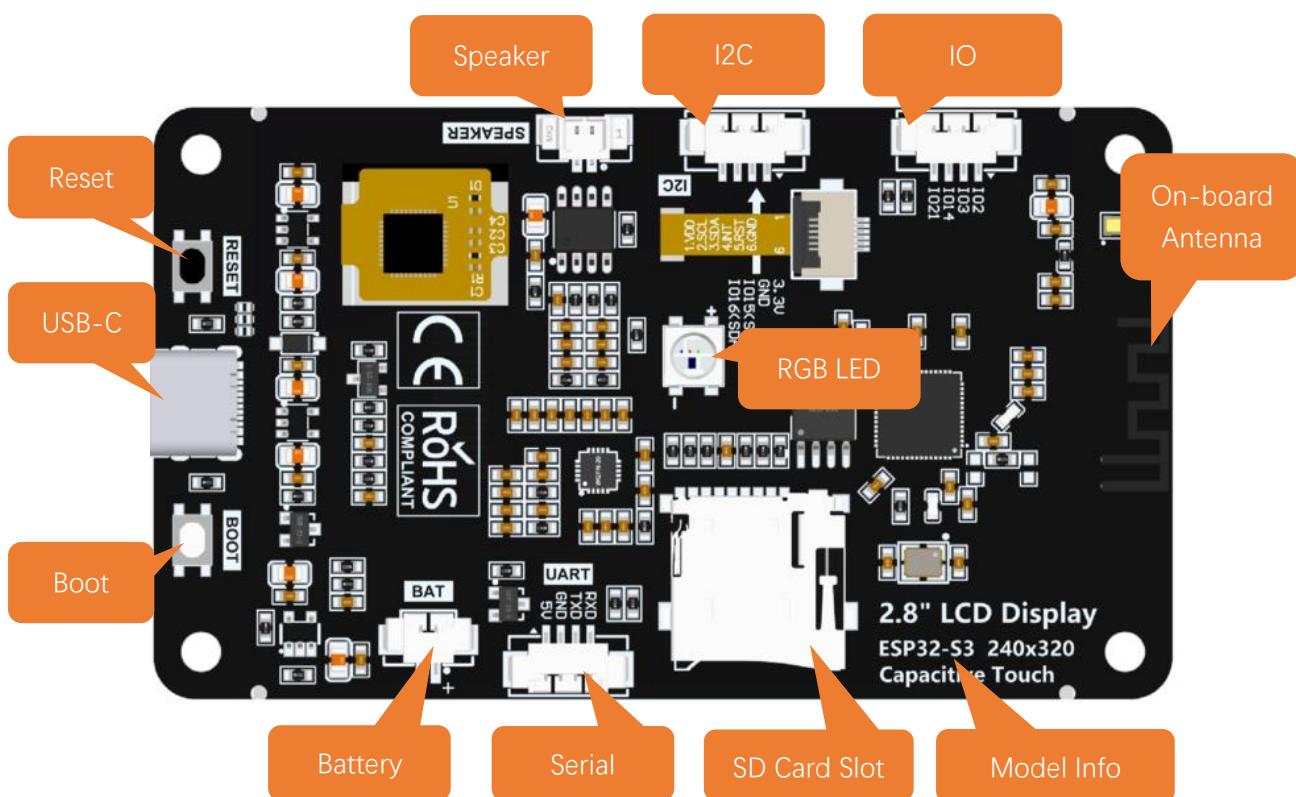
In this project, the ESP32-S3 communicates with XiaoZhi AI server through WebSocket protocol for data exchange.



## Freenove ESP32 S3 Display



## Hardware Interfaces



## XiaoZhi AI Firmware

If your hardware does not yet have XiaoZhi firmware installed, you can follow the upcoming tutorial to re-flash the firmware onto the ESP32-S3-WROOM.

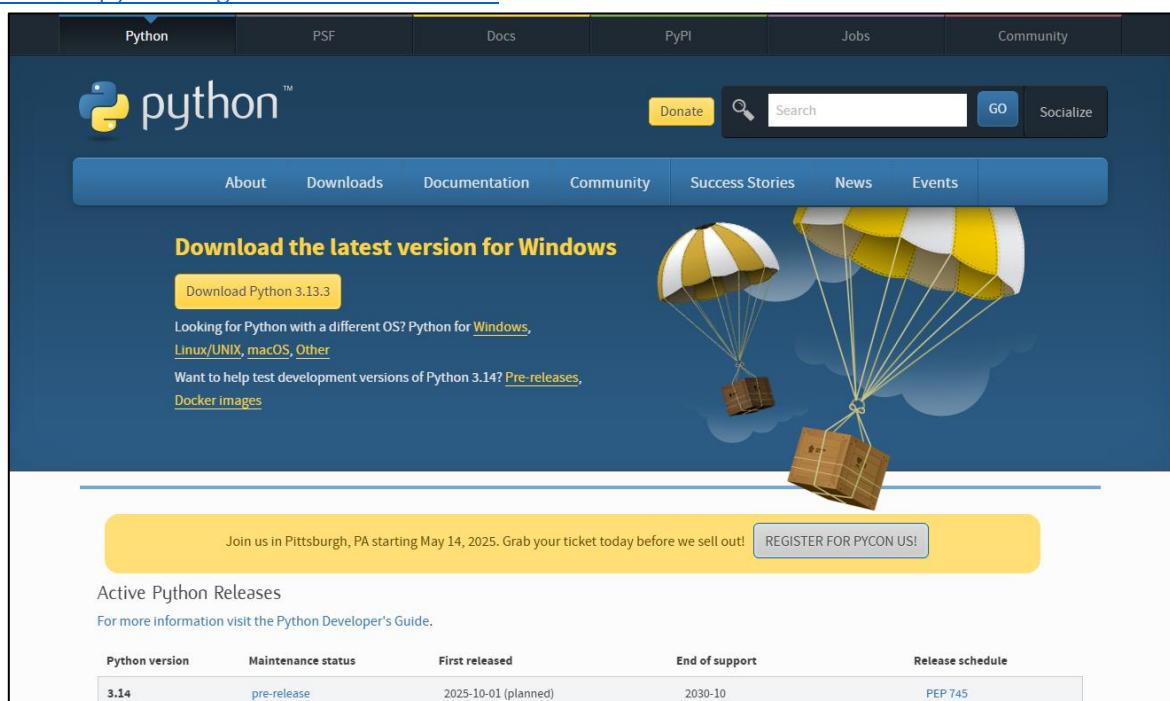
If your hardware already comes with XiaoZhi firmware pre-installed, you may skip this section.

## Installing Python (Required)

### Windows

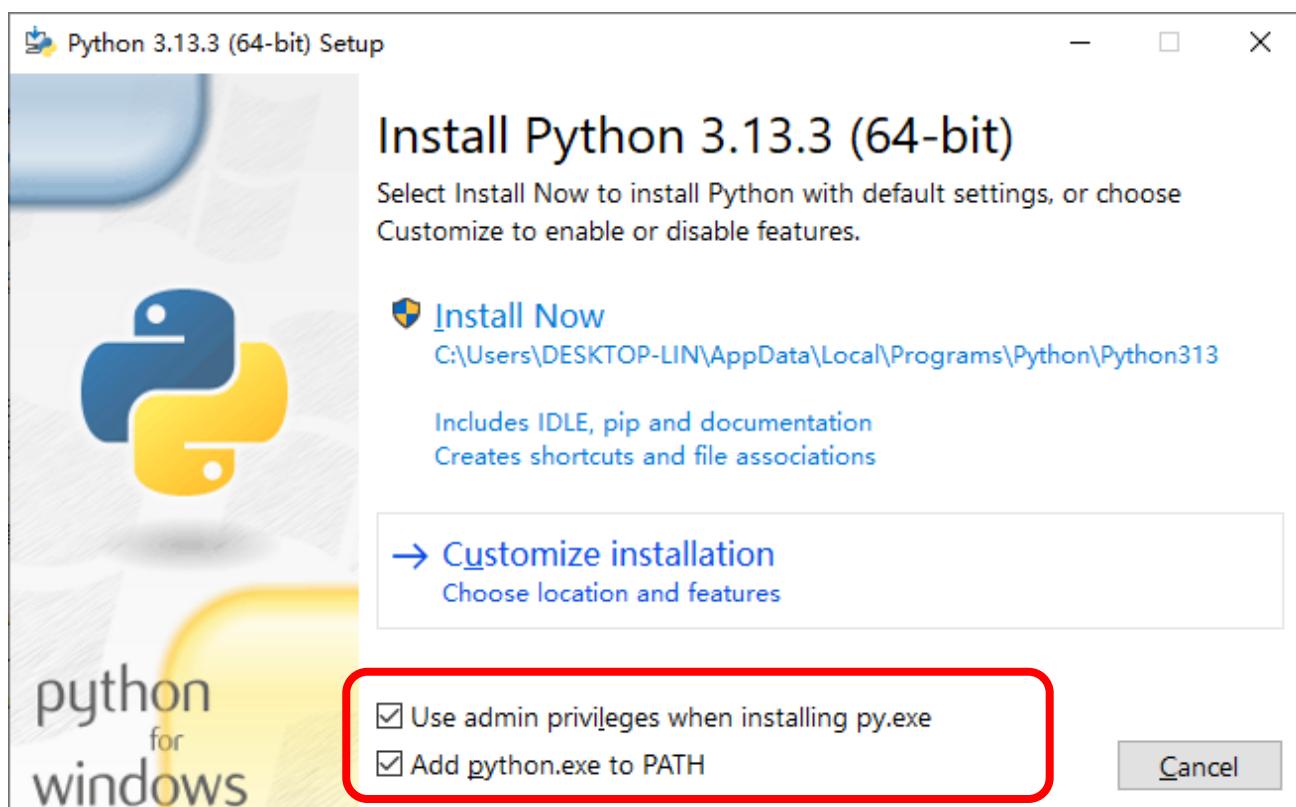
Download and install Python3 package.

<https://www.python.org/downloads/windows/>

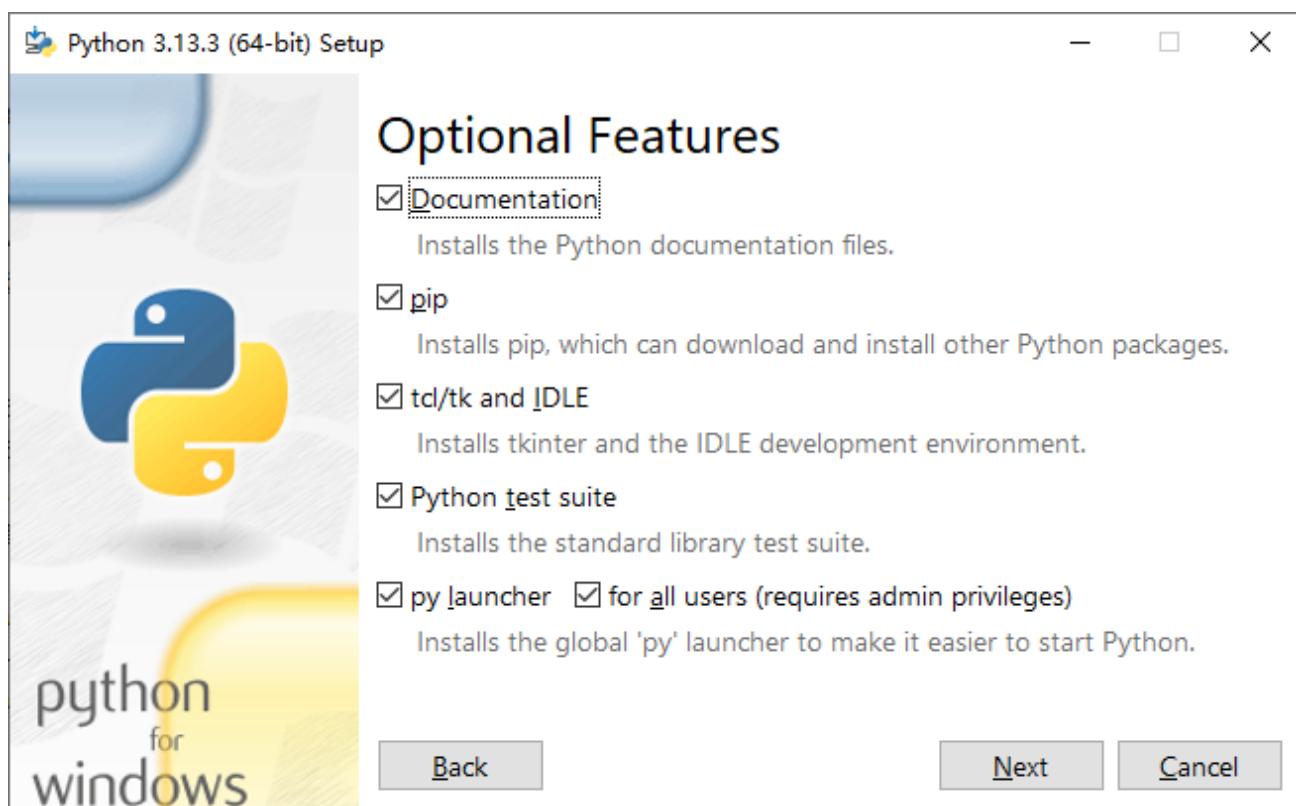


Click [Download Python 3.13.3](#)

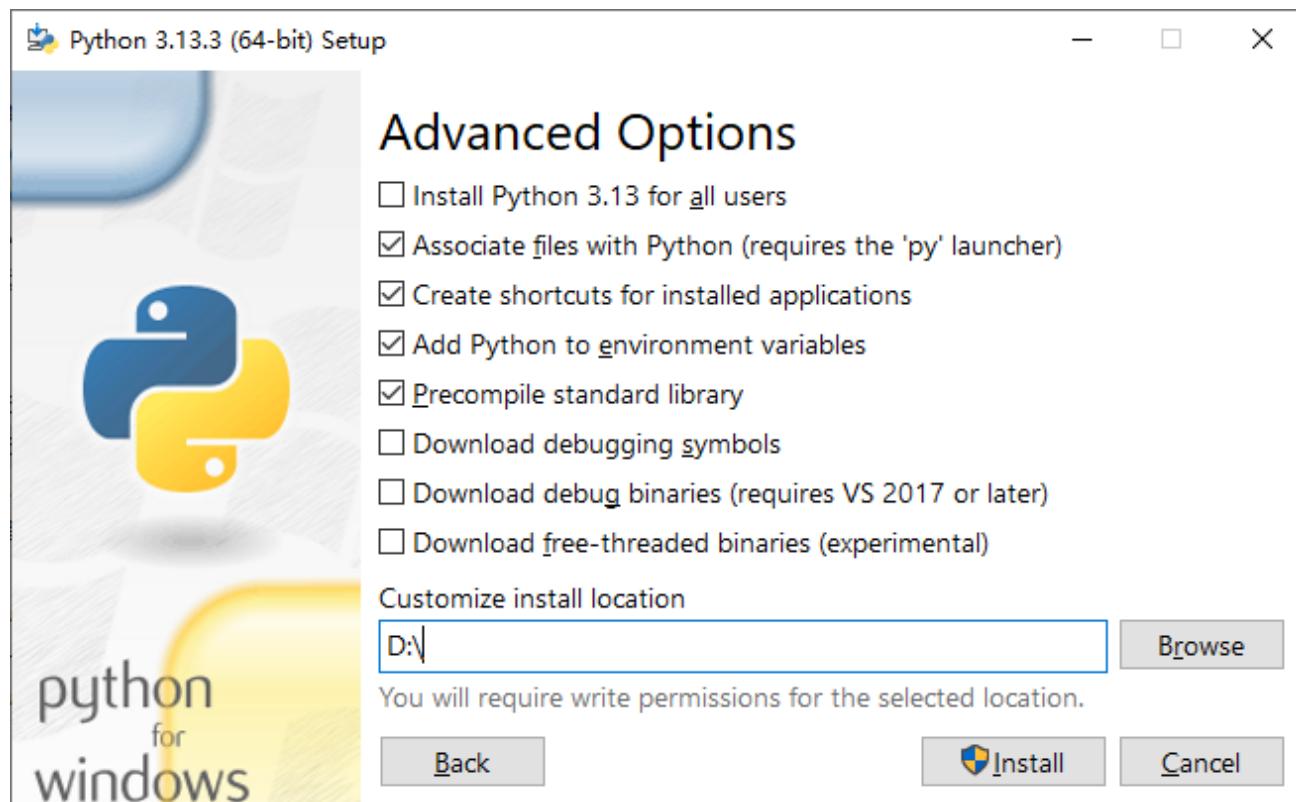
Please note that “Add Python 3.13 to PATH” MUST be check.



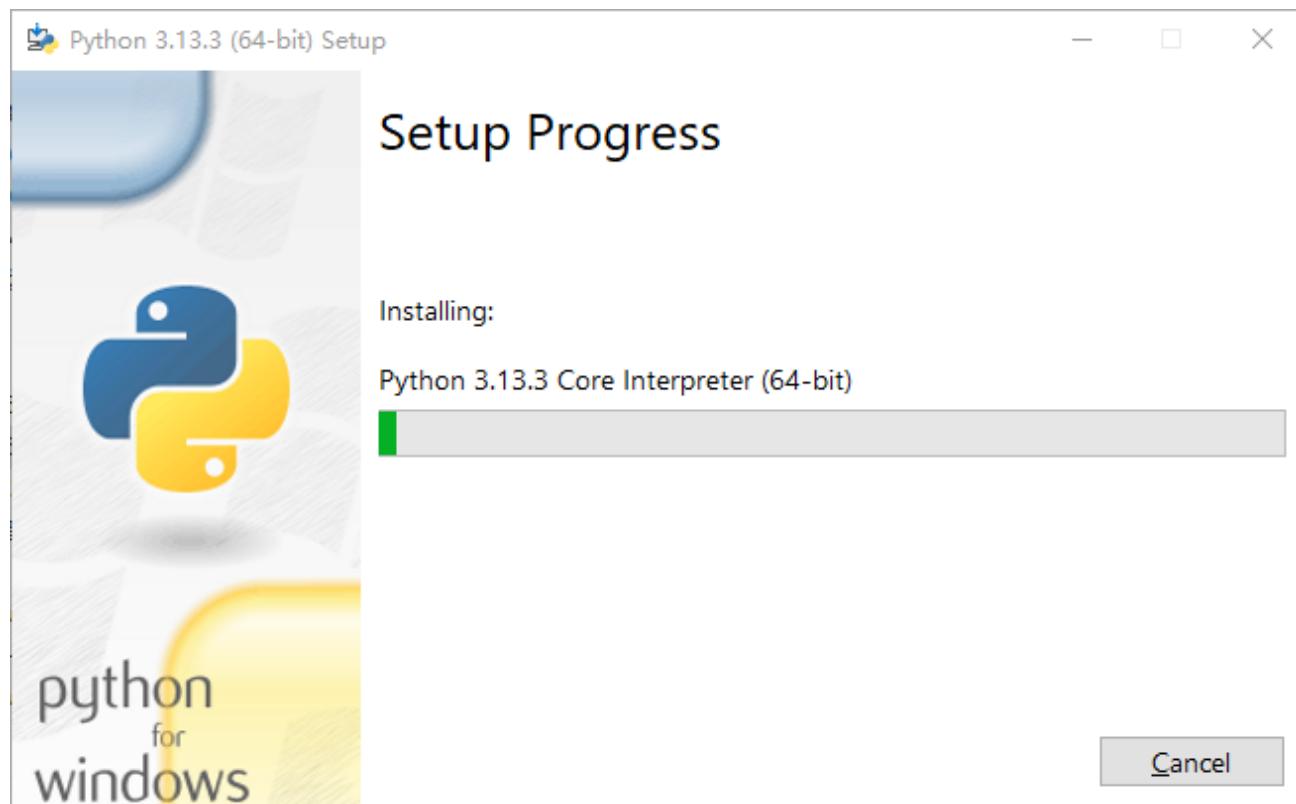
Check all the options and then click “Next”.



Here you can select the installation path of Python. We install it at D drive. If you are a novice, you can select the default path.



Wait for it to finish installing.



Now the installation is finished.

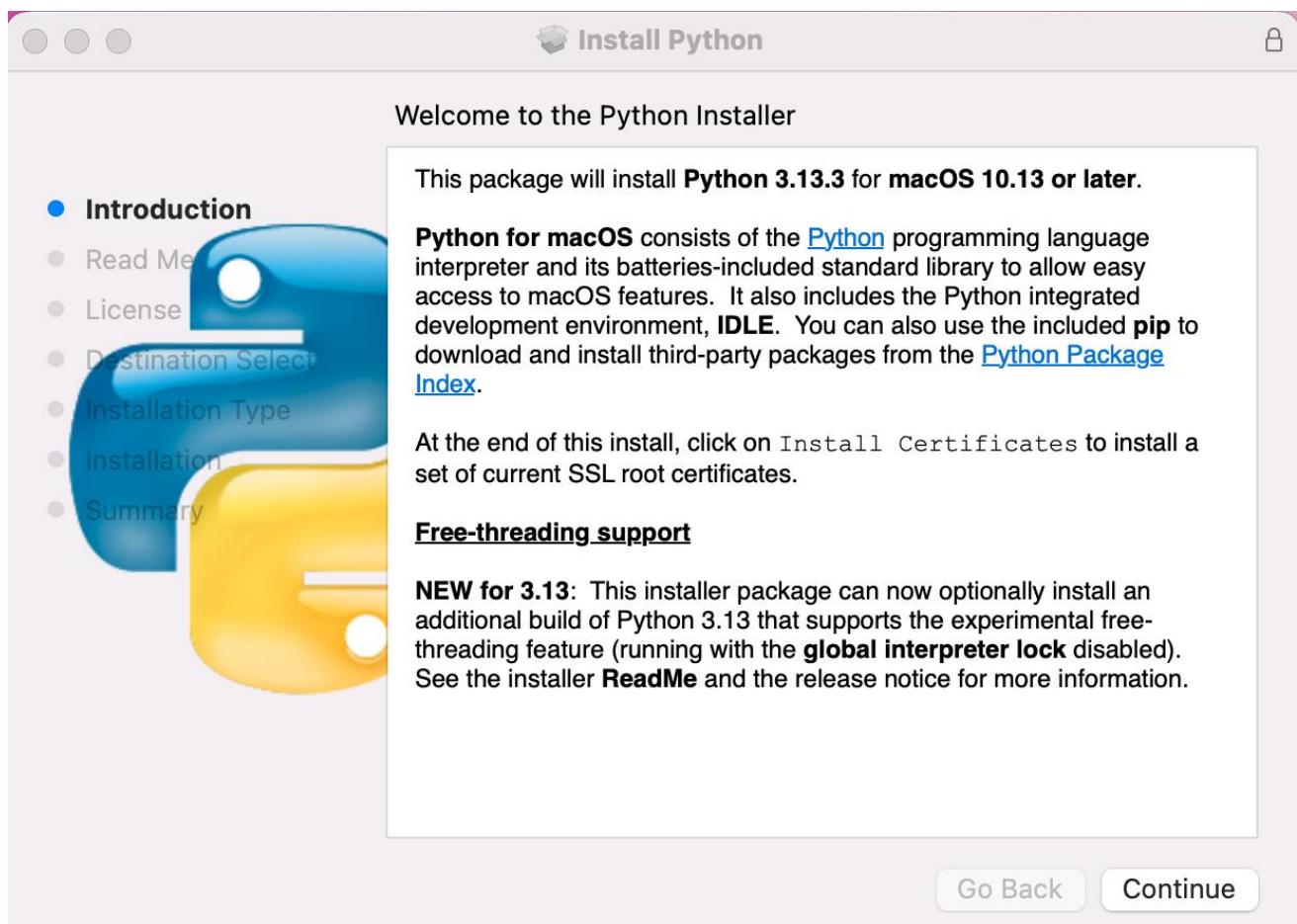
## Mac

Download installation package, link: <https://www.python.org/downloads/>

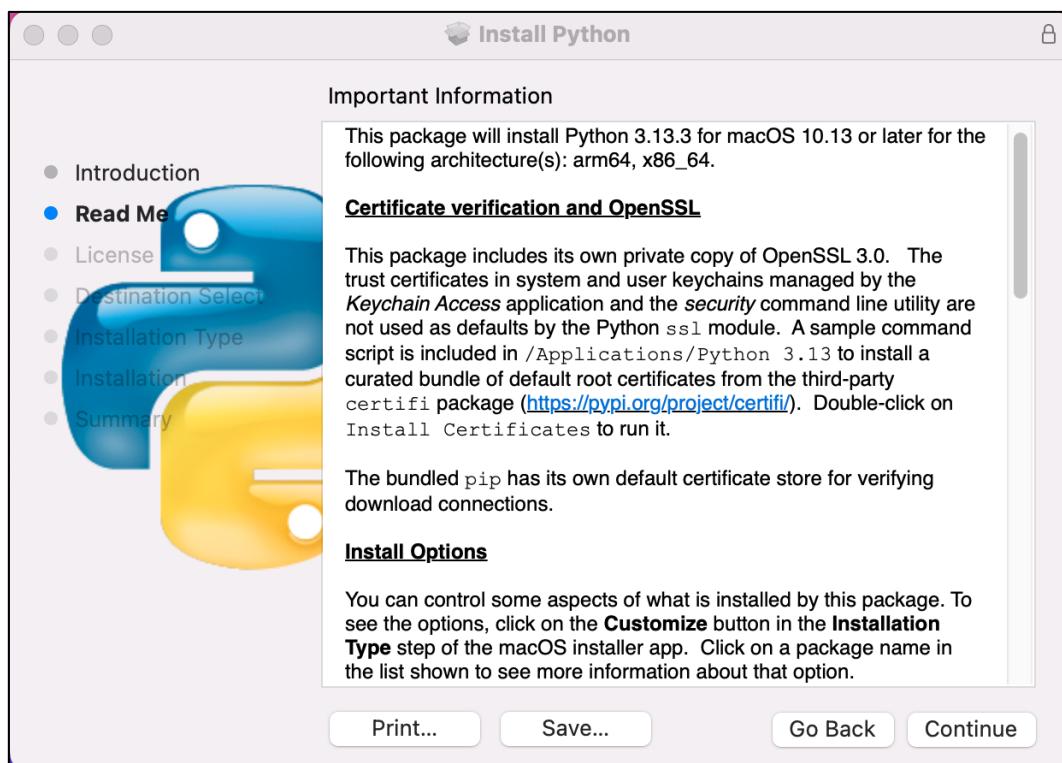
Click Download Python 3.13.3



Run the downloaded installation package. Click Continue



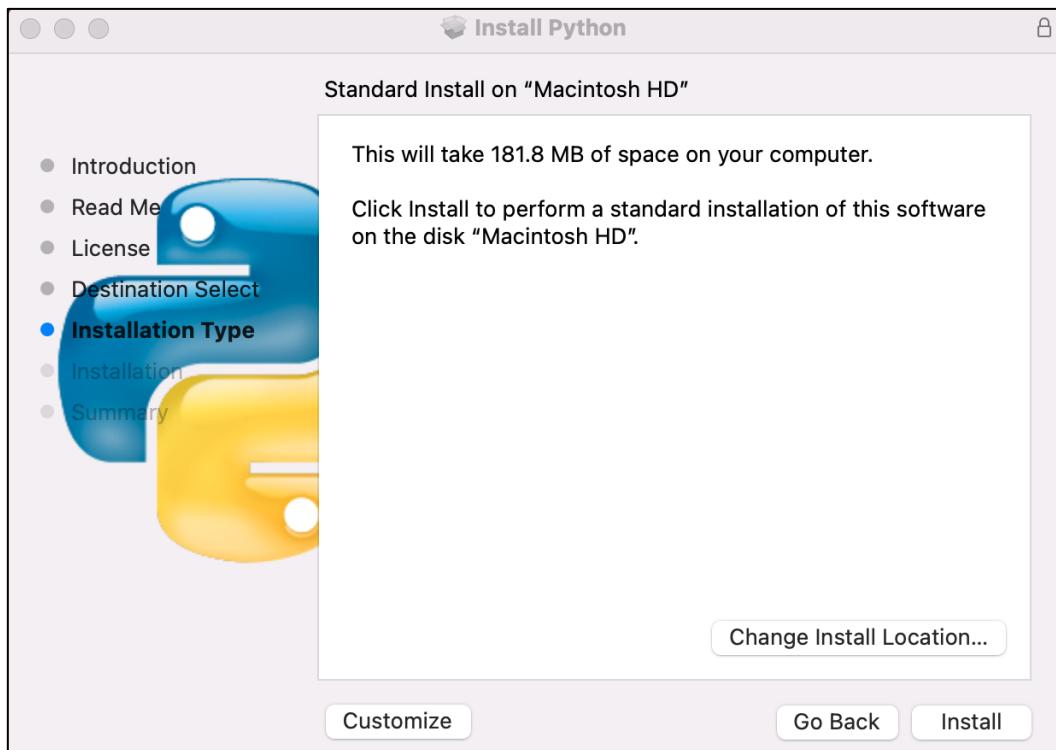
Click Continue



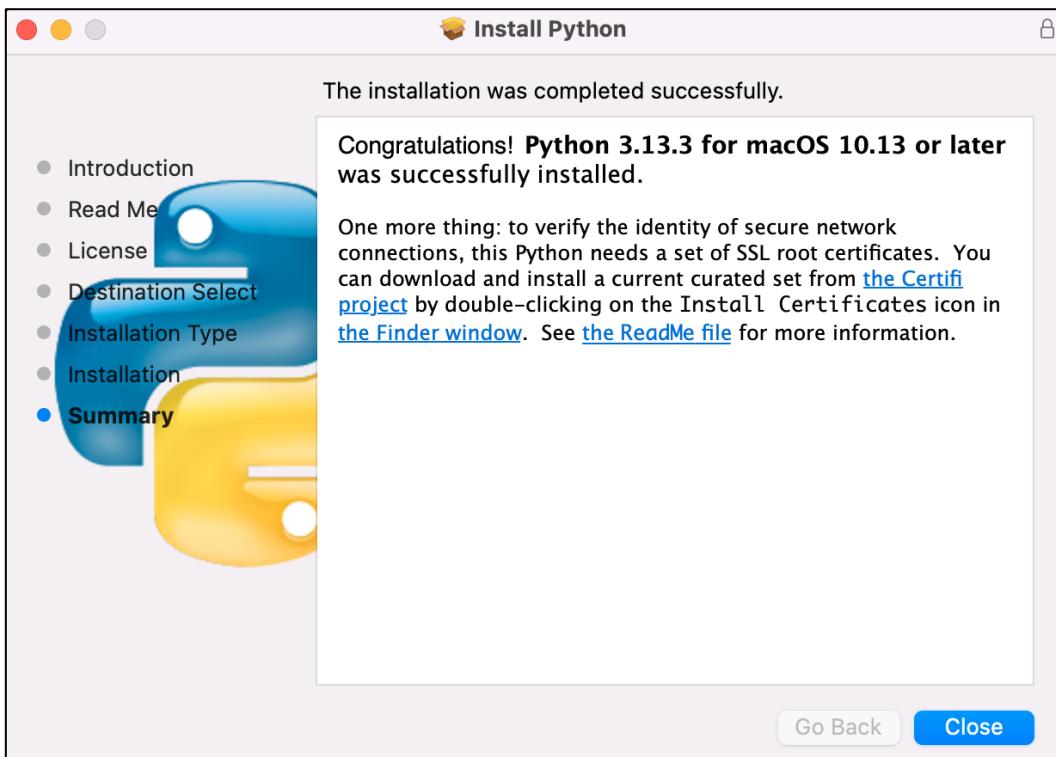
Click Continue



Click Install. If your computer has a password, enter the password and Install Software.



Now the installation succeeds.



## Linux

Check whether Python3 has already been installed.

```
python --version  
python3 --version
```

```
lin@ubuntu:~$ python --version  
python: command not found  
lin@ubuntu:~$ python3 --version  
bash: /usr/bin/python3: No such file or directory  
lin@ubuntu:~$
```

If it is not installed yet, run the following command to install it. This will install the latest version by default.

```
sudo apt install python3
```

```
lin@ubuntu:~$ sudo apt install python3  
Installing:  
  python3
```

Link python to Python 3.

```
sudo rm /usr/bin/python  
sudo ln -s /usr/bin/python3 /usr/bin/python
```

```
lin@ubuntu:~$ sudo rm /usr/bin/python  
lin@ubuntu:~$ sudo ln -s /usr/bin/python3 /usr/bin/python  
lin@ubuntu:~$ python --version  
Python 3.13.3
```

Install python3.13-venv virtual environment.

```
sudo apt install python3-venv
```

```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ sudo apt install python3-venv
```

Install python3-pip.

```
sudo apt install python3-pip
```

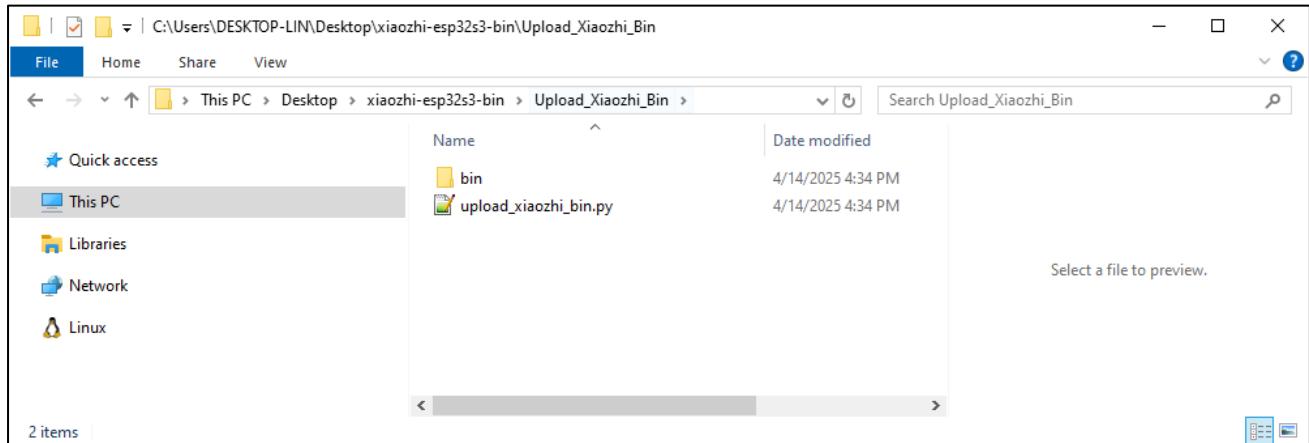
```
lin@ubuntu:~$ sudo apt install python3-pip  
python3-pip is already the newest version (25.0+dfsg-1).  
Summary:  
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 0  
lin@ubuntu:~$ pip --version  
pip 25.0 from /usr/lib/python3/dist-packages/pip (python 3.13)  
lin@ubuntu:~$
```



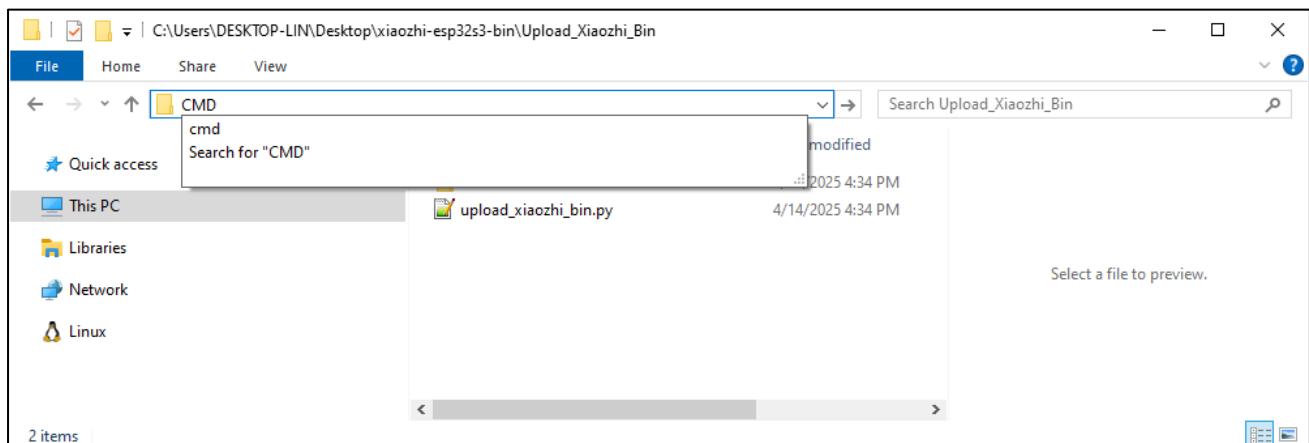
## Firmware Uploading

### Windows

Enter the Upload\_Xiaozhi\_Bin folder.



Type "CMD" in the file address bar and press Enter.



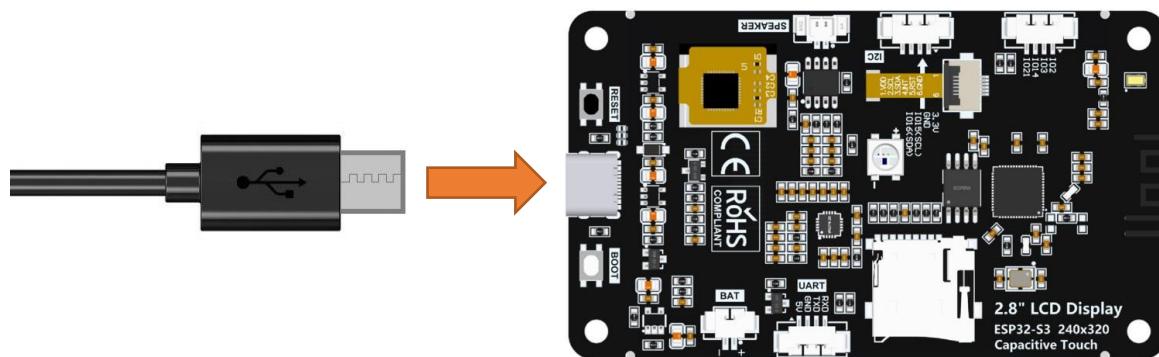
Type "**python --version**" to check if Python is installed. If no Python version information is displayed, it means Python is not properly installed—please reinstall it.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DESKTOP-LIN\Desktop\xiaozhi-esp32s3-bin\Upload_Xiaozhi_Bin>python --version
Python 3.13.3

C:\Users\DESKTOP-LIN\Desktop\xiaozhi-esp32s3-bin\Upload_Xiaozhi_Bin>
```

Connect the ESP32-S3-WROOM to your computer using a USB cable



Type "python upload\_xiaozhi\_bin.py" and press Enter.

If your computer does not have esptool or its required dependencies installed, they will be automatically installed.

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DESKTOP-LIN\Desktop\xiaozhi-esp32s3-bin\Upload_Xiaozhi_Bin>python --version
Python 3.13.3

C:\Users\DESKTOP-LIN\Desktop\xiaozhi-esp32s3-bin\Upload_Xiaozhi_Bin>python upload_xiaozhi_bin.py
esptool is not installed. Installing now...
Looking in indexes: https://mirrors.aliyun.com/pypi/simple/
Collecting esptool
  Using cached https://mirrors.aliyun.com/pypi/packages/5c/6b/3ce9bb7f36bdef3d6ae71646a1d3b7d59826a4
78f3ed8a783a93a2f8f537/esptool-4.8.1.tar.gz (409 kB)
    Installing build dependencies ... done
    Getting requirements to build wheel ... done
    Preparing metadata (pyproject.toml) ... done
Collecting bitstring!=4.2.0,>=3.1.6 (from esptool)
  Downloading https://mirrors.aliyun.com/pypi/packages/75/2d/174566b533755ddf8efb32a5503af61c756a983
de379f8ad3aed6a982d38/bitstring-4.3.1-py3-none-any.whl (71 kB)
Collecting cryptography>=2.1.4 (from esptool)
  Downloading https://mirrors.aliyun.com/pypi/packages/33/cf/1f7649b8b9a3543e042d3f348e398a061923ac0
5b507f3f4d95f11938aa9/cryptography-44.0.2-cp39-abi3-win_amd64.whl (3.2 MB)
    3.2/3.2 MB 2.6 MB/s eta 0:00:00
```

Then, it will invoke esptool to upload the files from the bin folder to the ESP32-S3-WROOM.

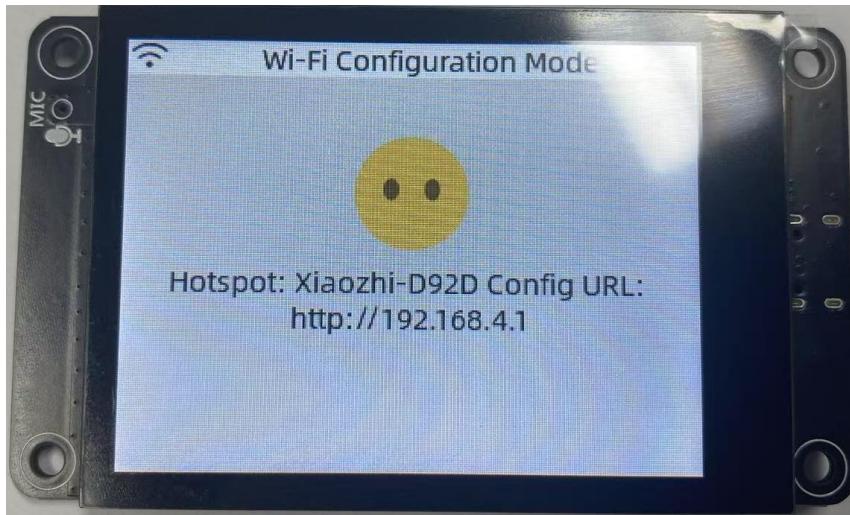
```
C:\Windows\System32\cmd.exe
SHA digest in image updated
Compressed 16352 bytes to 11342...
Wrote 16352 bytes (11342 compressed) at 0x00000000 in 0.2 seconds (effective 802.0 kbit/s)...
Hash of data verified.
Compressed 3561632 bytes to 2079901...
Wrote 3561632 bytes (2079901 compressed) at 0x00100000 in 22.8 seconds (effective 1247.5 kbit/s)...
Hash of data verified.
Compressed 3072 bytes to 141...
Wrote 3072 bytes (141 compressed) at 0x00008000 in 0.0 seconds (effective 1025.9 kbit/s)...
Hash of data verified.
Compressed 8192 bytes to 31...
Wrote 8192 bytes (31 compressed) at 0x0000d000 in 0.0 seconds (effective 1812.3 kbit/s)...
Hash of data verified.
Compressed 873228 bytes to 644882...
Wrote 873228 bytes (644882 compressed) at 0x00010000 in 6.4 seconds (effective 1090.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
esptool command executed successfully.

C:\Users\DESKTOP-LIN\Desktop\xiaozhi-esp32s3-bin\Upload_Xiaozhi_Bin>
```



You will see the following messages display on ESP32 S3 WROOM board.



## Mac

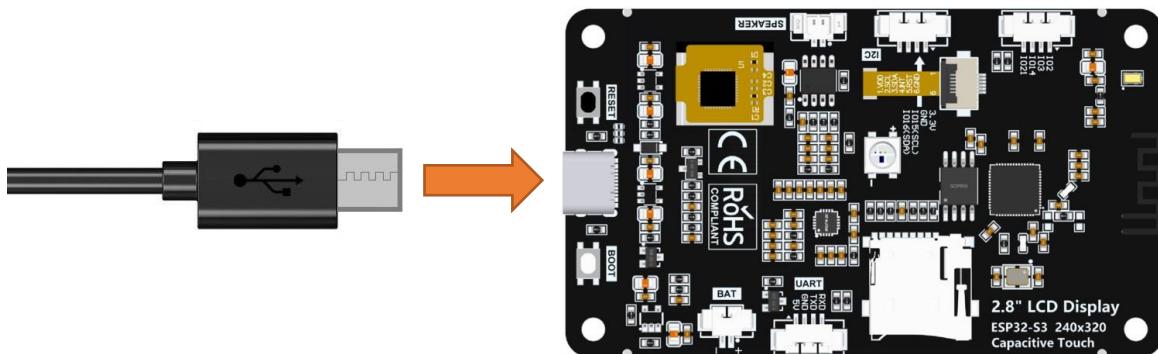
Enter the Upload\_XiaoZhi\_Bin folder.

```
Upload_XiaoZhi_Bin -- zsh -- 91x24
freenove@PandeMacBook-Air ~ % cd Desktop/xiaozhi-esp32s3-bin/Upload_XiaoZhi_Bin
freenove@PandeMacBook-Air Upload_XiaoZhi_Bin %
```

Type "python --version" to check if Python is installed. If no Python version information is displayed, it means Python is not properly installed—please reinstall it.

```
Upload_XiaoZhi_Bin -- zsh -- 91x24
freenove@PandeMacBook-Air Upload_XiaoZhi_Bin % python3 --version
Python 3.13.3
freenove@PandeMacBook-Air Upload_XiaoZhi_Bin %
```

Connect the ESP32-S3-WROOM to your computer using a USB cable



Type "python upload\_xiaozi\_bin.py" and press Enter.

```
Upload_XiaoZhi_Bin -- zsh -- 91x24
freenove@PandeMacBook-Air Upload_XiaoZhi_Bin % python3 upload_xiaozi_bin.py
```

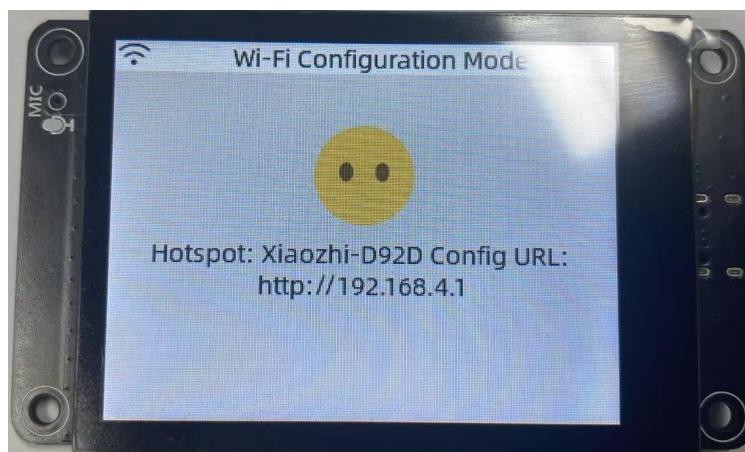
Then, it will invoke esptool to upload the files from the bin folder to the ESP32-S3-WROOM.

```
Serial port /dev/cu.wchusbserial5A4E1051341
Connecting....
Chip is ESP32-S3 (QFN56) (revision v0.2)
Features: WiFi, BLE, Embedded PSRAM 8MB (AP_3v3)
Crystal is 40MHz
MAC: 30:ed:a0:20:bd:9c
Uploading stub...
Running stub...
Stub running...
Changing baud rate to 2000000
Changed.
Configuring flash size...
Flash will be erased from 0x00000000 to 0x00003fff...
Flash will be erased from 0x00100000 to 0x00465fff...
Flash will be erased from 0x00008000 to 0x00008fff...
Flash will be erased from 0x0000d000 to 0x0000efff...
Flash will be erased from 0x00010000 to 0x000e5fff...
```

```
SHA digest in image updated
Compressed 16352 bytes to 11342...
Wrote 16352 bytes (11342 compressed) at 0x00000000 in 0.2 seconds (effective 775.3 kbit/s).
..
Hash of data verified.
Compressed 3561632 bytes to 2079901...
Wrote 3561632 bytes (2079901 compressed) at 0x00100000 in 22.8 seconds (effective 1247.2 kb
it/s)...
Hash of data verified.
Compressed 3072 bytes to 141...
Wrote 3072 bytes (141 compressed) at 0x00008000 in 0.0 seconds (effective 889.1 kbit/s)...
Hash of data verified.
Compressed 8192 bytes to 31...
Wrote 8192 bytes (31 compressed) at 0x0000d000 in 0.0 seconds (effective 1663.8 kbit/s)...
Hash of data verified.
Compressed 873228 bytes to 644882...
Wrote 873228 bytes (644882 compressed) at 0x00010000 in 6.4 seconds (effective 1089.5 kbit/
s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
esptool command executed successfully.
freenove@PandeMacBook-Air Upload_Xiaozhi_Bin % c
```

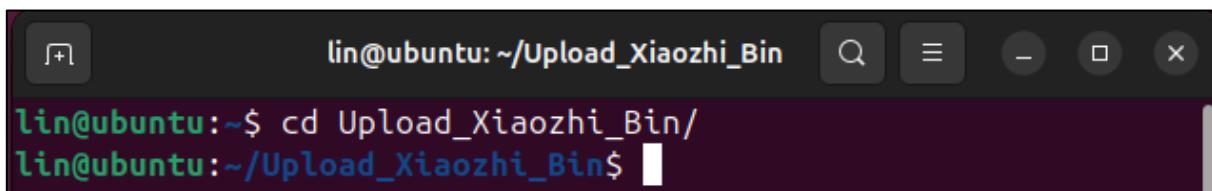
You will see the following messages display on ESP32 S3 WROOM board.



## Linux

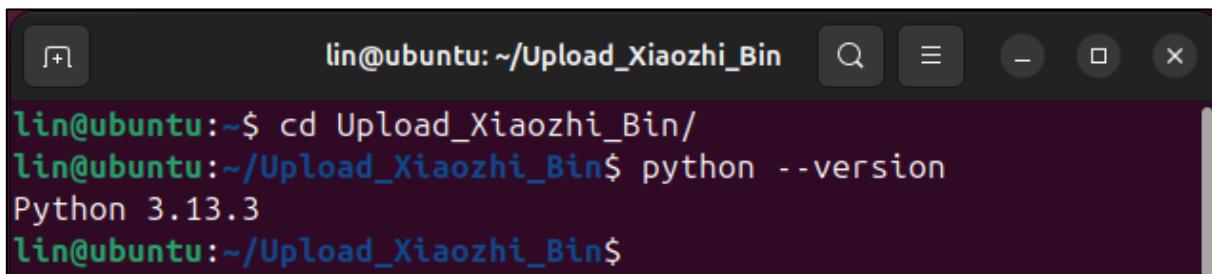
Enter the Upload\_Xiaozhi\_Bin folder.

```
cd Upload_Xiaozhi_Bin
```



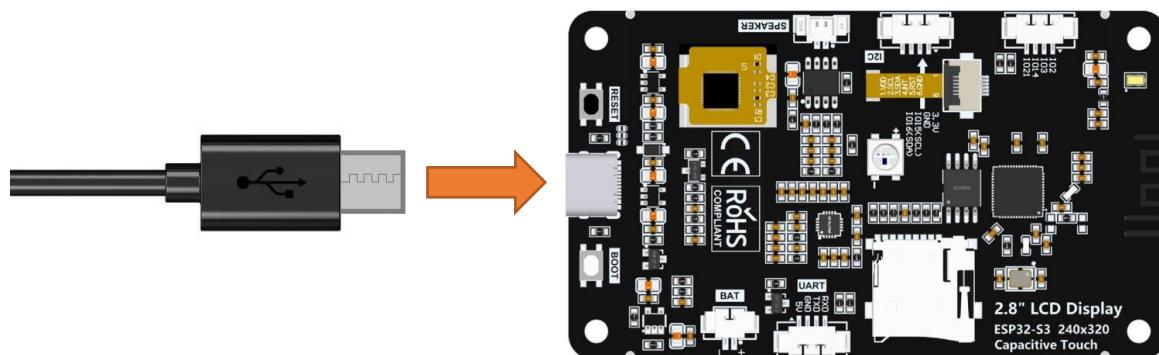
```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ cd Upload_Xiaozhi_Bin/
lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

Enter "python --version" to check if the Python environment is installed. If the Python version information is not displayed, it means Python is not properly installed. Please reinstall it.



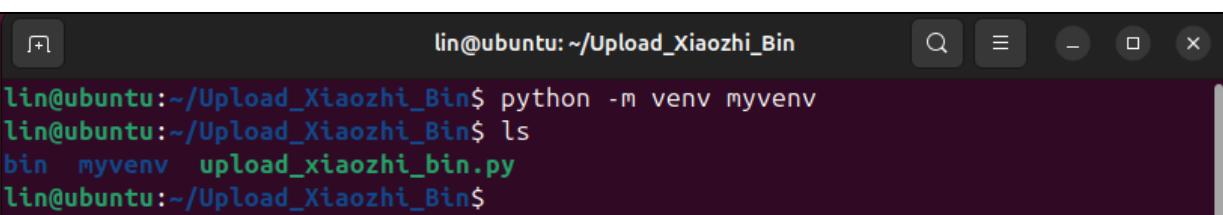
```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ cd Upload_Xiaozhi_Bin/
lin@ubuntu:~/Upload_Xiaozhi_Bin$ python --version
Python 3.13.3
lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

Connect the ESP32-S3-WROOM to your computer using a USB cable, making sure to plug it into the correct Type-C port (do not use the wrong connector).



Create a virtual environment and name it as "myvenv".

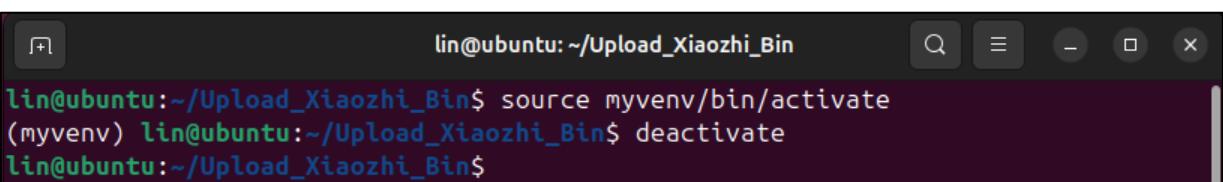
```
python -m venv myvenv
```



```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ python -m venv myvenv
lin@ubuntu:~/Upload_Xiaozhi_Bin$ ls
bin  myvenv  upload_xiaozhi_bin.py
lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

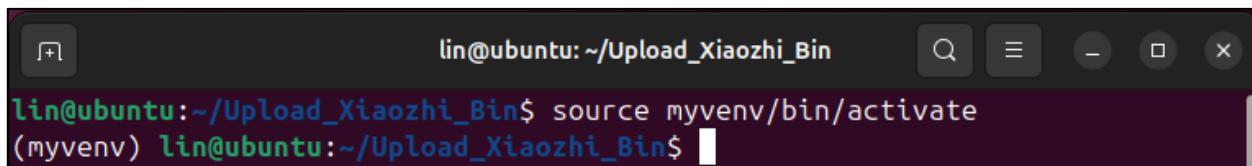
You can run the following command to activate or exit the virtual environment.

```
source myvenv/bin/activate
deactivate
```



```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ source myvenv/bin/activate
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$ deactivate
lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

Activate the virtual environment.

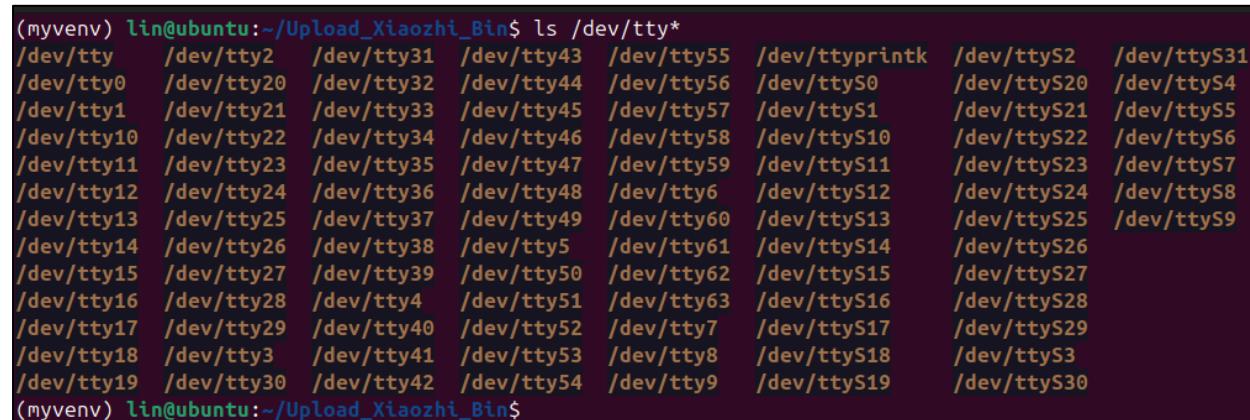


```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ source myvenv/bin/activate
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

Run the command to check the port of ESP32S3.

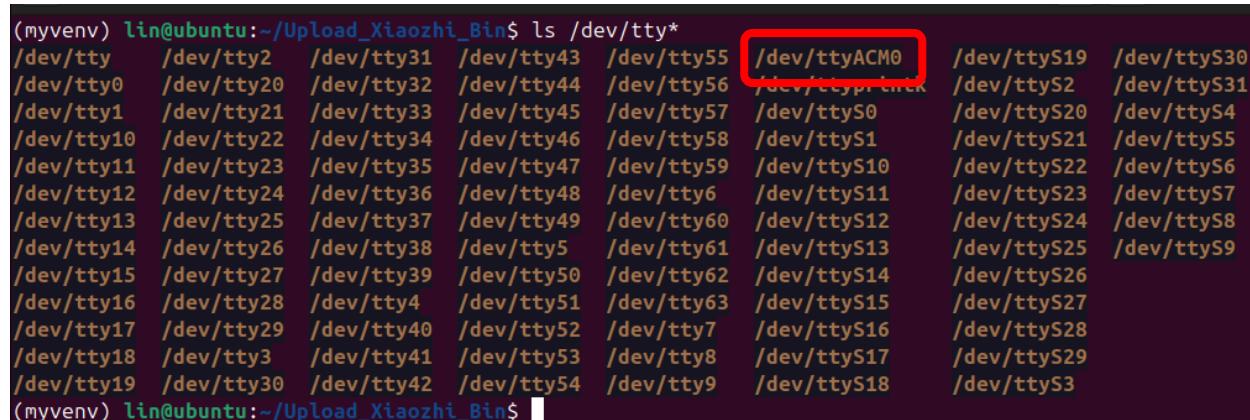
```
ls /dev/tty*
```

When the ESP32S3 is not connected to the computer, the ports are as shown below.



```
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$ ls /dev/tty*
/dev/tty  /dev/tty2  /dev/tty31  /dev/tty43  /dev/tty55  /dev/ttyp0ink  /dev/ttyS2  /dev/ttyS31
/dev/tty0  /dev/tty20 /dev/tty32  /dev/tty44  /dev/tty56  /dev/ttyS0  /dev/ttyS20 /dev/ttyS4
/dev/tty1  /dev/tty21 /dev/tty33  /dev/tty45  /dev/tty57  /dev/ttyS1  /dev/ttyS21 /dev/ttyS5
/dev/tty10 /dev/tty22 /dev/tty34  /dev/tty46  /dev/tty58  /dev/ttyS10 /dev/ttyS22 /dev/ttyS6
/dev/tty11 /dev/tty23 /dev/tty35  /dev/tty47  /dev/tty59  /dev/ttyS11 /dev/ttyS23 /dev/ttyS7
/dev/tty12 /dev/tty24 /dev/tty36  /dev/tty48  /dev/tty60  /dev/ttyS12 /dev/ttyS24 /dev/ttyS8
/dev/tty13 /dev/tty25 /dev/tty37  /dev/tty49  /dev/tty61  /dev/ttyS13 /dev/ttyS25 /dev/ttyS9
/dev/tty14 /dev/tty26 /dev/tty38  /dev/tty5  /dev/tty61  /dev/ttyS14 /dev/ttyS26
/dev/tty15 /dev/tty27 /dev/tty39  /dev/tty50  /dev/tty62  /dev/ttyS15 /dev/ttyS27
/dev/tty16 /dev/tty28 /dev/tty4  /dev/tty51  /dev/tty63  /dev/ttyS16 /dev/ttyS28
/dev/tty17 /dev/tty29 /dev/tty40  /dev/tty52  /dev/tty7  /dev/ttyS17 /dev/ttyS29
/dev/tty18 /dev/tty3  /dev/tty41  /dev/tty53  /dev/tty8  /dev/ttyS18 /dev/ttyS3
/dev/tty19 /dev/tty30 /dev/tty42  /dev/tty54  /dev/tty9  /dev/ttyS19 /dev/ttyS30
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

After connecting the ESP32S3, a new port is generated.



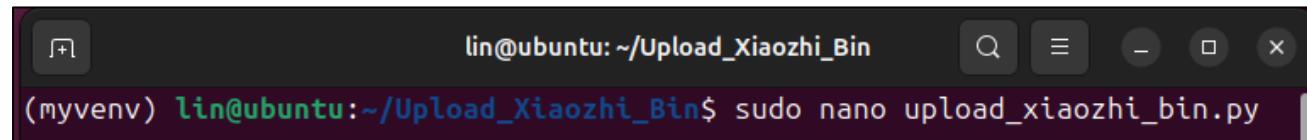
```
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$ ls /dev/tty*
/dev/tty  /dev/tty2  /dev/tty31  /dev/tty43  /dev/tty55  /dev/ttyACM0  /dev/ttyS19  /dev/ttyS30
/dev/tty0  /dev/tty20 /dev/tty32  /dev/tty44  /dev/tty56  /dev/ttyp0ink  /dev/ttyS2  /dev/ttyS31
/dev/tty1  /dev/tty21 /dev/tty33  /dev/tty45  /dev/tty57  /dev/ttyS0  /dev/ttyS20 /dev/ttyS4
/dev/tty10 /dev/tty22 /dev/tty34  /dev/tty46  /dev/tty58  /dev/ttyS1  /dev/ttyS21 /dev/ttyS5
/dev/tty11 /dev/tty23 /dev/tty35  /dev/tty47  /dev/tty59  /dev/ttyS10 /dev/ttyS22 /dev/ttyS6
/dev/tty12 /dev/tty24 /dev/tty36  /dev/tty48  /dev/tty6  /dev/ttyS11 /dev/ttyS23 /dev/ttyS7
/dev/tty13 /dev/tty25 /dev/tty37  /dev/tty49  /dev/tty60  /dev/ttyS12 /dev/ttyS24 /dev/ttyS8
/dev/tty14 /dev/tty26 /dev/tty38  /dev/tty5  /dev/tty61  /dev/ttyS13 /dev/ttyS25 /dev/ttyS9
/dev/tty15 /dev/tty27 /dev/tty39  /dev/tty50  /dev/tty62  /dev/ttyS14 /dev/ttyS26
/dev/tty16 /dev/tty28 /dev/tty4  /dev/tty51  /dev/tty63  /dev/ttyS15 /dev/ttyS27
/dev/tty17 /dev/tty29 /dev/tty40  /dev/tty52  /dev/tty7  /dev/ttyS16 /dev/ttyS28
/dev/tty18 /dev/tty3  /dev/tty41  /dev/tty53  /dev/tty8  /dev/ttyS17 /dev/ttyS29
/dev/tty19 /dev/tty30 /dev/tty42  /dev/tty54  /dev/tty9  /dev/ttyS18 /dev/ttyS3
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

The newly generated one is the port of ESP32S3. Remember it.

Before running the python file, we need to modify the port.

Run the following command to open the python file.

```
sudo nano upload_xiaozhi_bin.py
```



```
(myvenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$ sudo nano upload_xiaozhi_bin.py
```

In the text editor, locate the line '--port', 'COMx' and replace 'COMx' with the port number assigned to your ESP32-S3 on Linux computer.

The screenshot shows a terminal window titled "lin@ubuntu: ~/Upload\_Xiaozhi\_Bin". Inside the terminal, the nano editor is open with the file "upload\_xiaozhi\_bin.py". The code in the file is:

```
GNU nano 8.3          upload_xiaozhi_bin.py
sys.exit(1)

def run_esptool_command():
    """Execute the esptool command"""
    command = [
        sys.executable, "-m", "esptool",
        "--chip", "esp32s3",
        #"--port", "COMx",
        "--baud", "2000000",
        "--before", "default_reset",
        "--after", "hard_reset",
    ]
    [ Wrote 64 lines ]
```

At the bottom of the terminal, there is a status bar with keyboard shortcuts:

- ^G Help
- ^O Write Out
- ^F Where Is
- ^K Cut
- ^T Execute
- ^X Exit
- ^R Read File
- ^\\ Replace
- ^U Paste
- ^J Justify

The modification is as shown below.

The screenshot shows a terminal window titled "lin@ubuntu: ~/Upload\_Xiaozhi\_Bin". Inside the terminal, the nano editor is open with the file "upload\_xiaozhi\_bin.py". The code in the file is identical to the previous screenshot, but the status bar at the bottom is different:

```
GNU nano 8.3          upload_xiaozhi_bin.py *
sys.exit(1)

def run_esptool_command():
    """Execute the esptool command"""
    command = [
        sys.executable, "-m", "esptool",
        "--chip", "esp32s3",
        "--port", "/dev/ttyACM0",
        "--baud", "2000000",
        "--before", "default_reset",
        "--after", "hard_reset",
    ]
    [ Wrote 64 lines ]
```

At the bottom of the terminal, there is a status bar with keyboard shortcuts:

- ^G Help
- ^O Write Out
- ^F Where Is
- ^K Cut
- ^T Execute
- ^X Exit
- ^R Read File
- ^\\ Replace
- ^U Paste
- ^J Justify

Press "Ctrl+O" to save the changes and "Ctrl+X" to exit the file.

Run the python file.

```
python upload_xiaozhi_bin.py
```

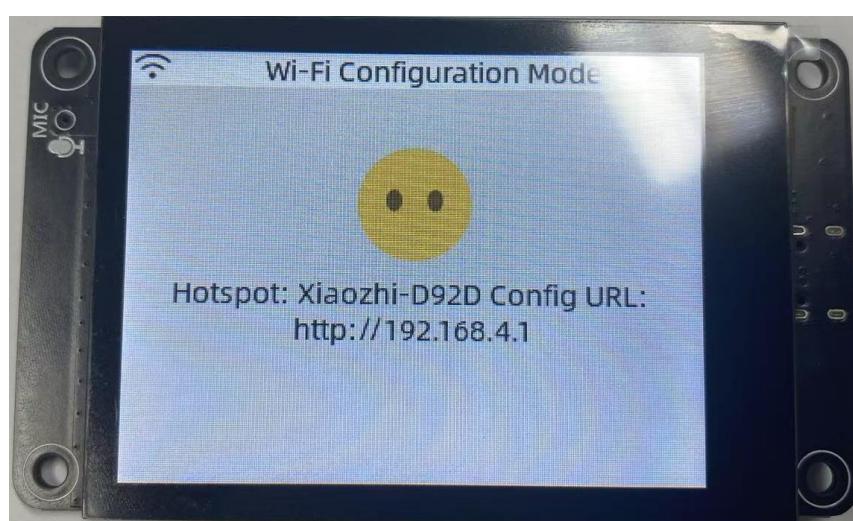
```
lin@ubuntu:~/Upload_Xiaozhi_Bin$ python upload_xiaozhi_bin.py
esptool is already installed.
Executing esptool command...
esptool.py v4.8.1
Serial port /dev/ttyACM0
Connecting....
Chip is ESP32-S3 (QFN56) (revision v0.2)
Features: WiFi, BLE, Embedded PSRAM 8MB (AP_3v3)
Crystal is 40MHz
MAC: 30:ed:a0:20:bd:9c
Uploading stub...
Running stub...
```

The successful code uploading is as shown below.

```
Wrote 8192 bytes (31 compressed) at 0x0000d000 in 0.0 seconds (effective 18
43.6 kbit/s)...
Hash of data verified.
Compressed 873228 bytes to 644882...
Wrote 873228 bytes (644882 compressed) at 0x00010000 in 6.3 seconds (effect
ive 1103.4 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
esptool command executed successfully.
(myenv) lin@ubuntu:~/Upload_Xiaozhi_Bin$
```

The display on the ESP32 S3 WROOM is as shown below.



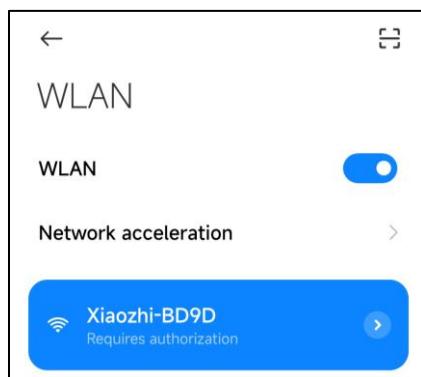


## ESP32 S3 WROOM Network Configuration

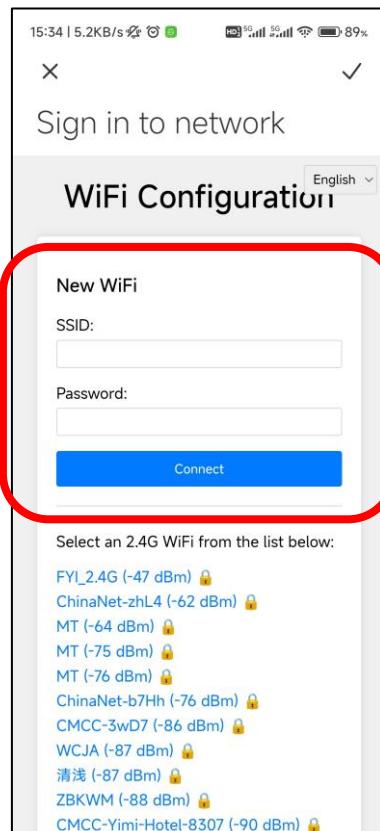
If your ESP32-S3-WROOM does not yet have the XiaoZhi AI firmware installed, proceed to the If you want to explore the XiaoZhi AI code, go to [the XiaoZhi AI Code section](#).

If your ESP32-S3-WROOM already has the XiaoZhi AI firmware integrated:

1. On your smart phone, enable WiFi.
2. Look for a hotspot named "Xiaozhi-XXXX" (an open network, no password required).
3. Connect to it to proceed



After connecting to the WiFi, follow the on-screen prompts to tap the notification. This will automatically launch your mobile browser and direct you to <http://192.168.4.1>.



## WiFi Connection Setup for ESP32-S3-WROOM

### Enter WiFi Credentials:

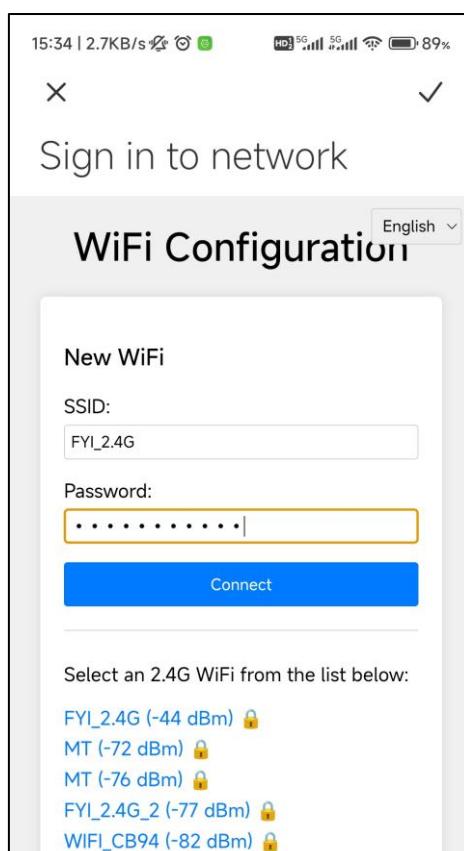
SSID: Enter your WiFi network name (2.4GHz only).

Password: Enter your WiFi password.

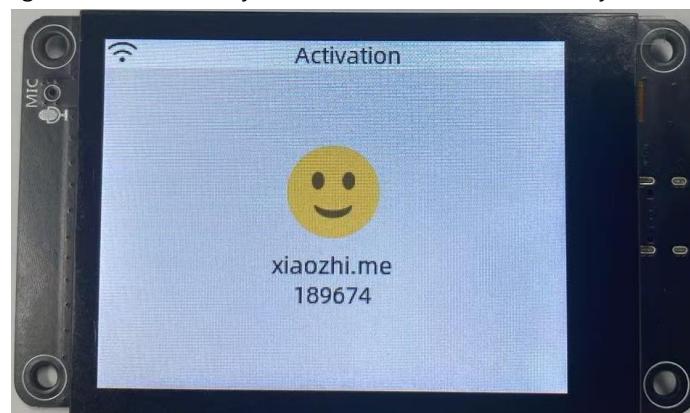
Click Connect to proceed.

### Important Notes:

- The ESP32-S3-WROOM only supports 2.4GHz WiFi networks.
- If your router broadcasts both 2.4GHz and 5GHz, ensure the ESP32 connects to the 2.4GHz band only.
- Avoid mixed-mode (2.4GHz + 5GHz combined) settings, as this may prevent successful connection.



When you see the following screen, it means your ESP32-S3 has successfully connected to your WiFi network.



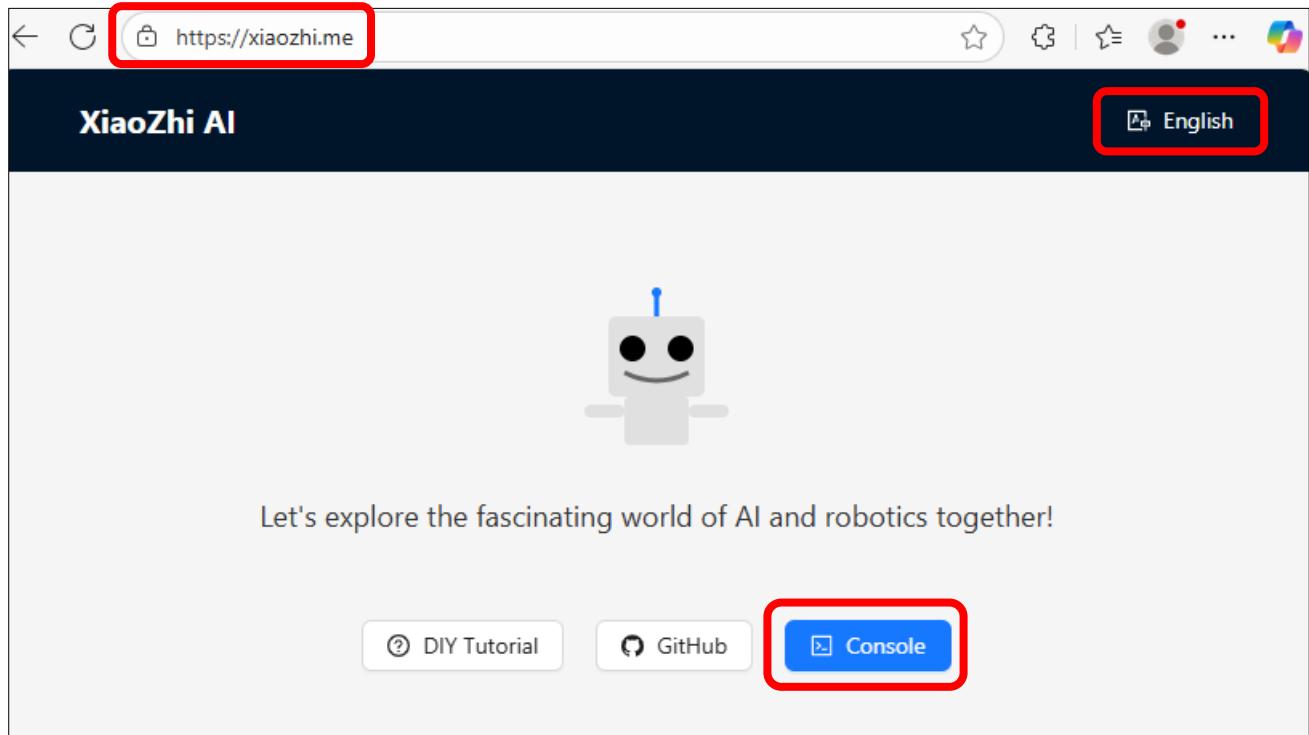


## XiaoZhi AI Server Configuration

Ensure your phone/computer and ESP32-S3-WROOM are connected to the same router WiFi network.

Open a browser on your device and visit: <https://xiaozi.me/>

Please note that due to varying internet policies in different countries, users from certain regions may experience difficulties accessing the website. For specific details, please refer to the relevant national internet policies.



If you don't have an account yet, please click Console and register using your mobile number.

Please note that currently, XiaoZhi AI servers only support mobile number registration from the following countries.

If you do not have an account yet, please register one and login.

XiaoZhi AI

English

Login

Phone Username

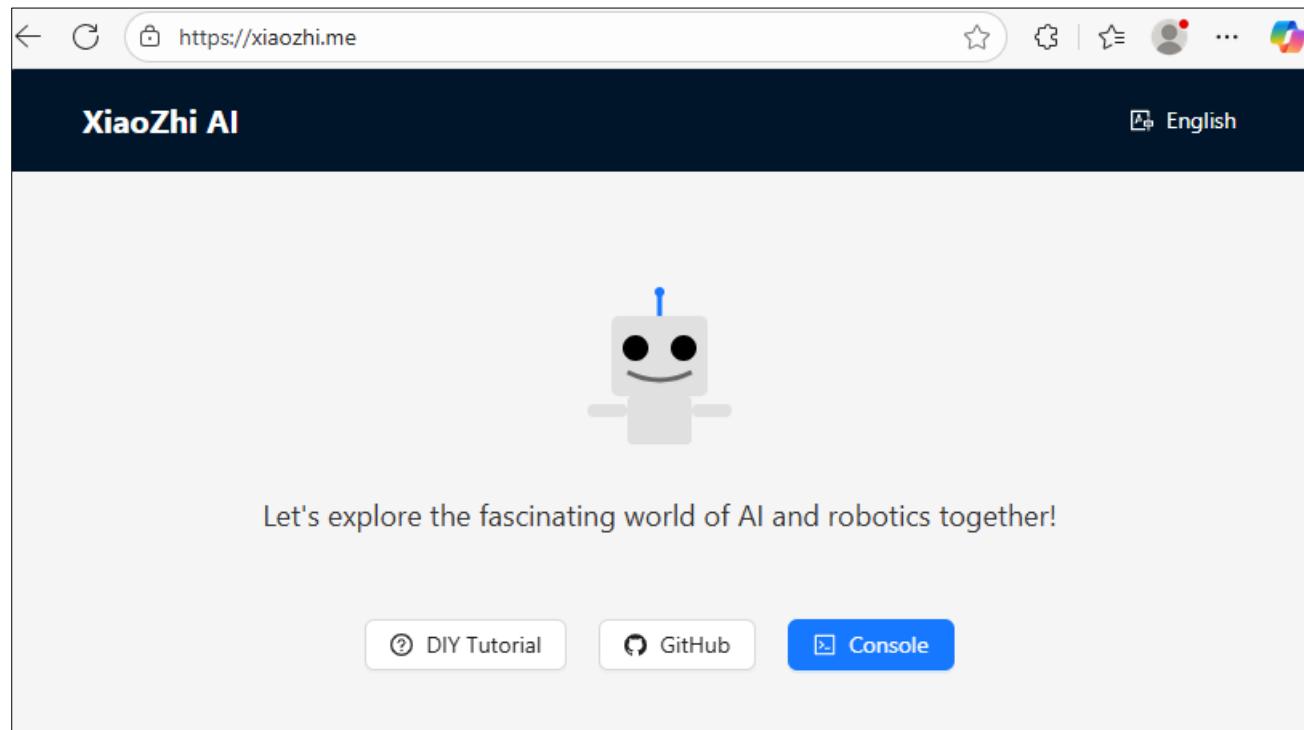
+86 Enter phone number

Enter the graphic verification code

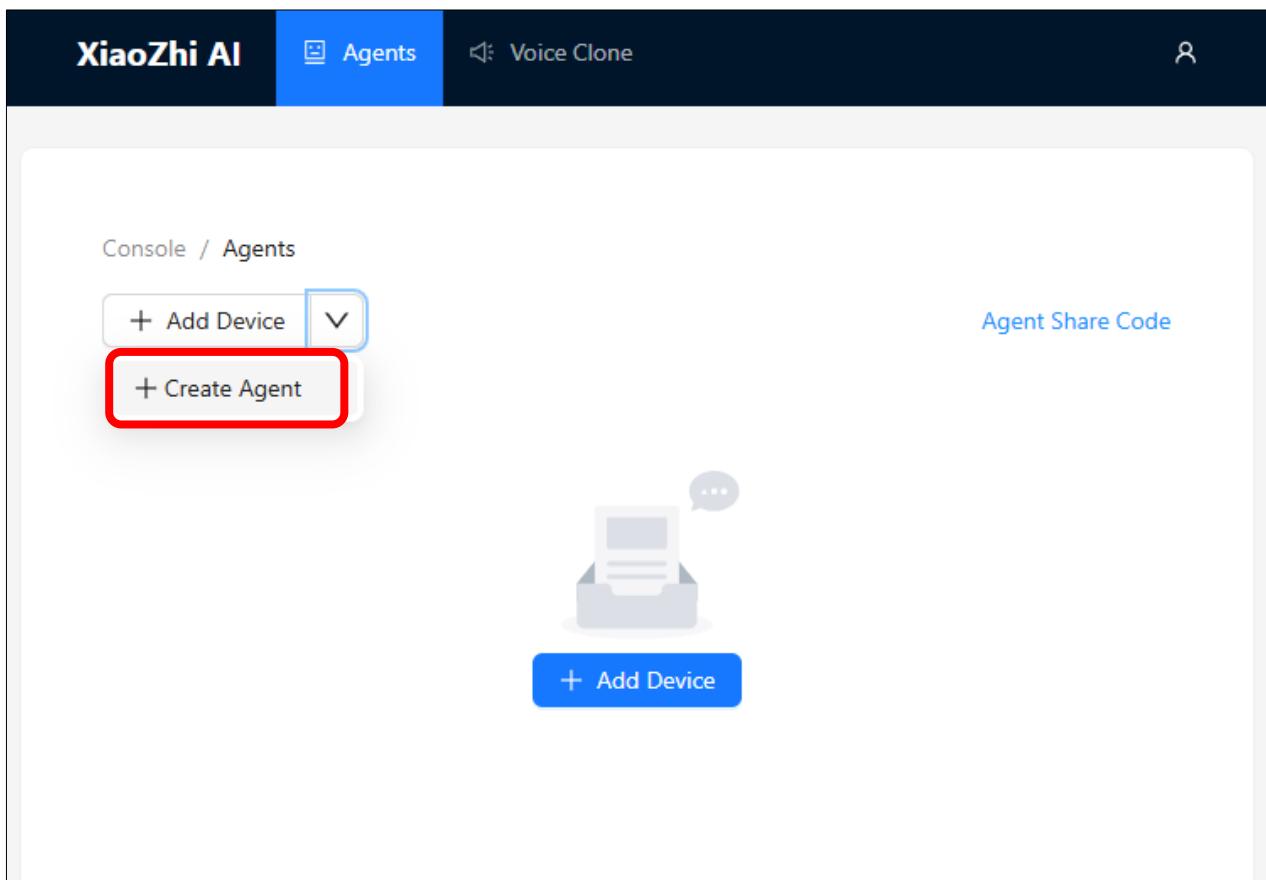
Enter phone verification code Send Code

Login

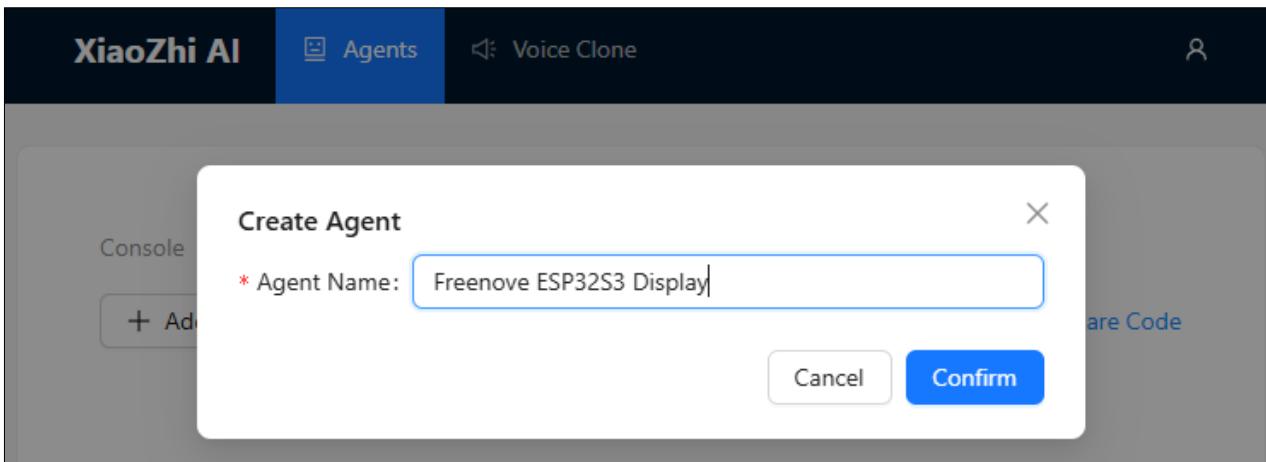
Click "Console" to start setting up your XiaoZhi AI server.



Click "Create Agent" to set up a new AI assistant.



Name it whatever you like and click "Confirm".



Click “Configure Role” to configure your AI assistant.

Console / Agents

+ Add Device

Agent Share Code

Freenove ESP32S3 Display

Voice Role: 湾湾小何

Language Model: Qwen3 Realtime (Recomm...)

Recent Chat: None

Configure Role Speaker Recognition

Chat History Add Device

Click "English Tutor" (keep all other options unchanged).

Console / Agents / Configure Role

Configure Role Freenove ESP32S3 Display

Role Template

台湾女友 土豆子 English Tutor 好奇小男孩 汪汪队队长



Scroll to the bottom of the page and click "Save" to confirm all settings.

Language Model

Qwen3 Realtime (Recommended)

After changing the model, it's recommended to clear the memory for better experience.

> Advanced Settings

**Save**    Reset

Note: After saving the configuration, the device needs to be restarted for the new settings to take effect.    MCP Endpoint

Click "Agents" to return to the main dashboard and select "**Add Device**" to register new hardware.

Console / Agents

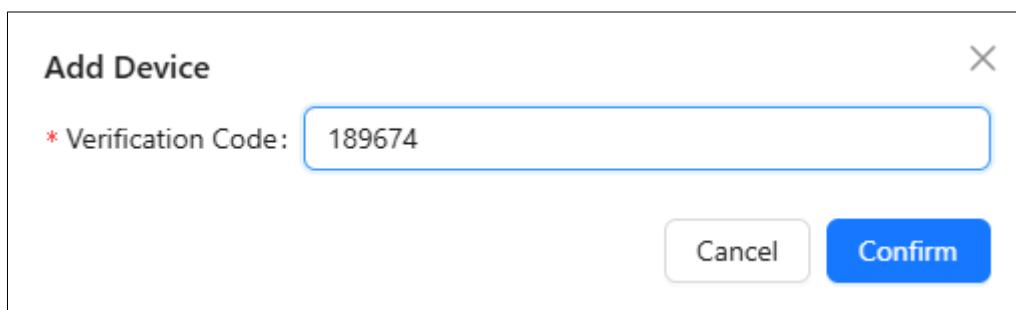
+ Add Device ▾

Freenove ESP32S3 Display

Voice Role: 爽快思思 / Skye (en-US)  
Language Model: Qwen3 Realtime (Recomm...  
Recent Chat: None

Configure Role   Speaker Recognition  
Chat History   **Add Device**

In the new pop-up window, enter the on-screen numeric code displayed on your ESP32-S3. Click "Confirm" to complete pairing.



The interface will now display as shown below.

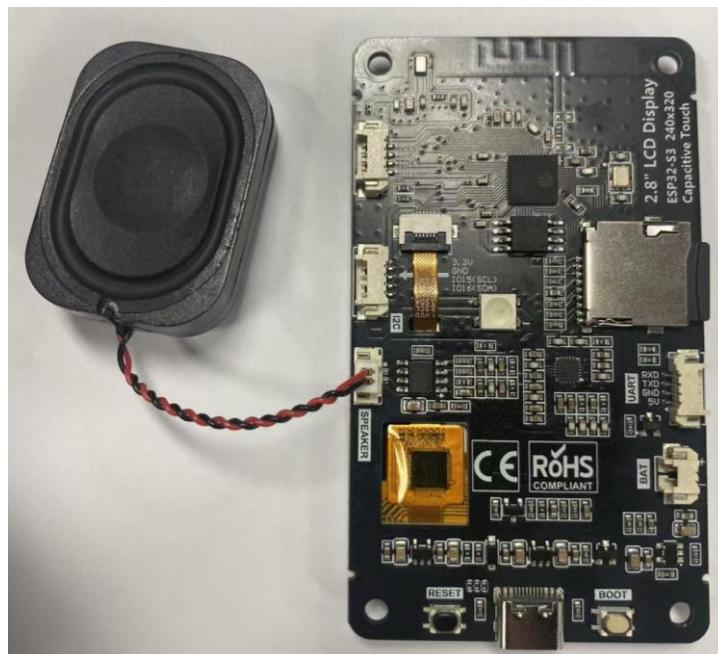


Press the RST button on the Freenove ESP32 S3 Display board to restart the board.

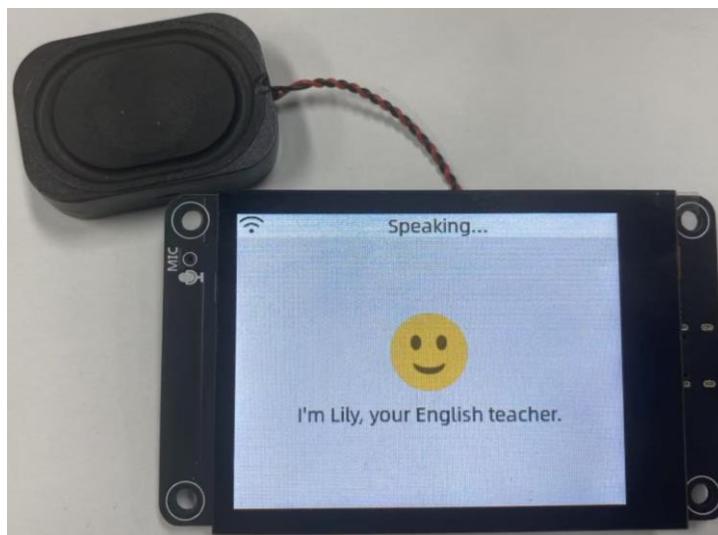


You've successfully finished configuring XiaoZhi AI!

Connect speaker



To activate, say "Hi, ESP" to the microphone; the system will now respond to your voice commands



You can communicate with it in either Chinese or English.

# XiaoZhi AI Code

## Visual Studio Code

### Windows

First, download Visual Studio Code by visiting <https://code.visualstudio.com/Download>. Choose the appropriate version for your operating system, then download and install it.

The screenshot shows the official Visual Studio Code download page at <https://code.visualstudio.com/Download>. The page features a large central heading "Download Visual Studio Code" and a subtext "Free and built on open source. Integrated Git, debugging and extensions." Below this, there are three main download sections corresponding to different operating systems:

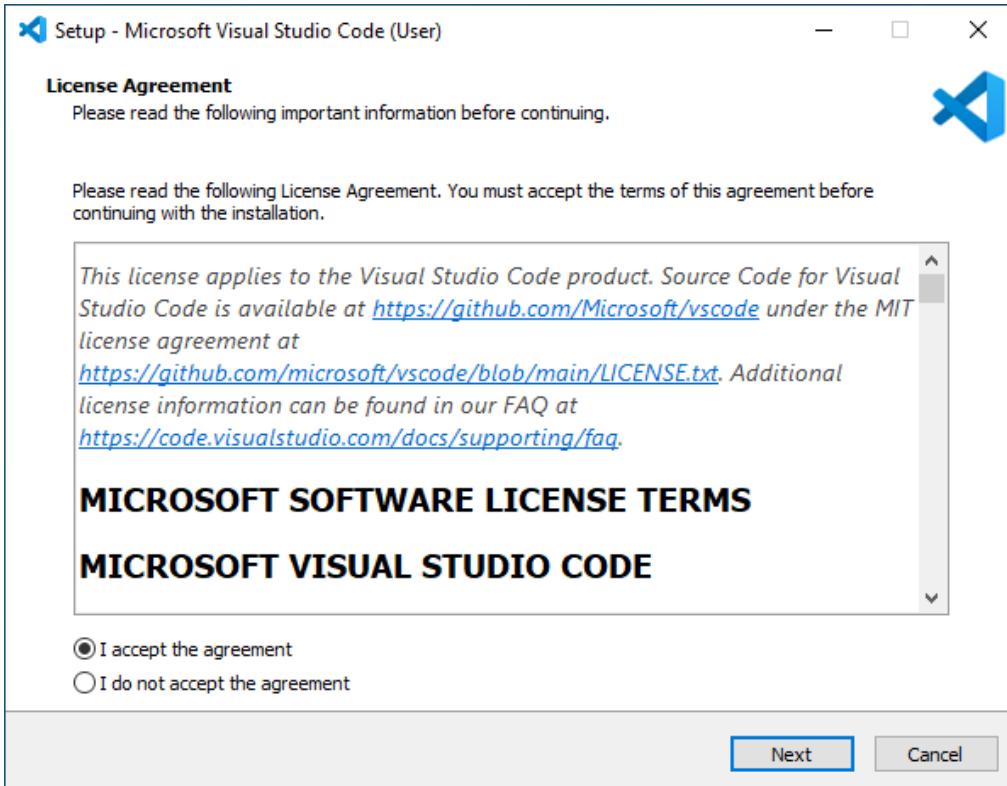
- Windows:** Shows the Windows logo. Download buttons are available for "Windows 10, 11" (User Installer, System Installer, .zip, CLI) and "Windows 10, 11" (x64, Arm64).
- Linux:** Shows the Tux (Linux penguin) logo. Download buttons are available for ".deb" (Debian, Ubuntu), ".rpm" (Red Hat, Fedora, SUSE), ".tar.gz" (x64, Arm32, Arm64), "Snap" (Snap Store), and "CLI" (x64, Arm32, Arm64). Additional options include ".deb" (x64, Arm32, Arm64), ".rpm" (x64, Arm32, Arm64), ".tar.gz" (x64, Arm32, Arm64), and "Snap" (Snap Store).
- macOS:** Shows the Apple logo. Download buttons are available for "macOS 10.15+" (x64, Arm64) and "macOS 10.15+" (Intel chip, Apple silicon, Universal). Additional options include ".zip" (Intel chip, Apple silicon), "CLI" (Intel chip, Apple silicon), and ".zip" (Intel chip, Apple silicon, Universal).

At the bottom of the page, a note states: "By downloading and using Visual Studio Code, you agree to the [license terms](#) and [privacy statement](#).

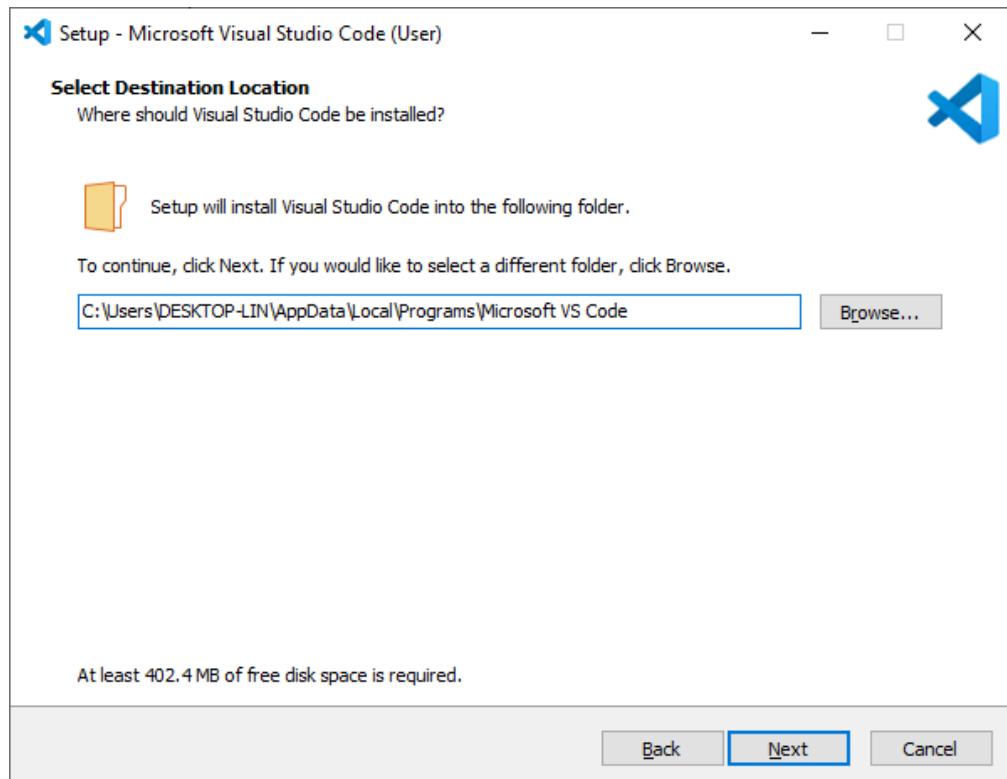
Double-click the downloaded .exe file to run it.

Check the box for "I accept the agreement."

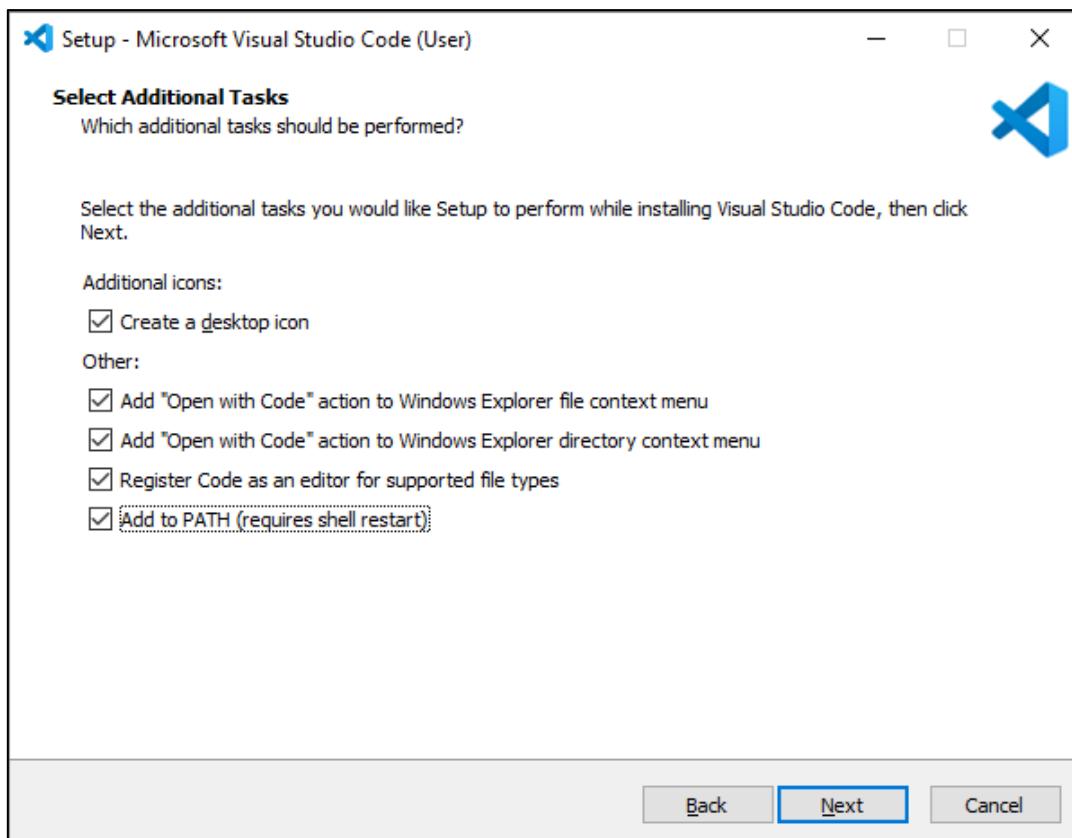
Then click Next.



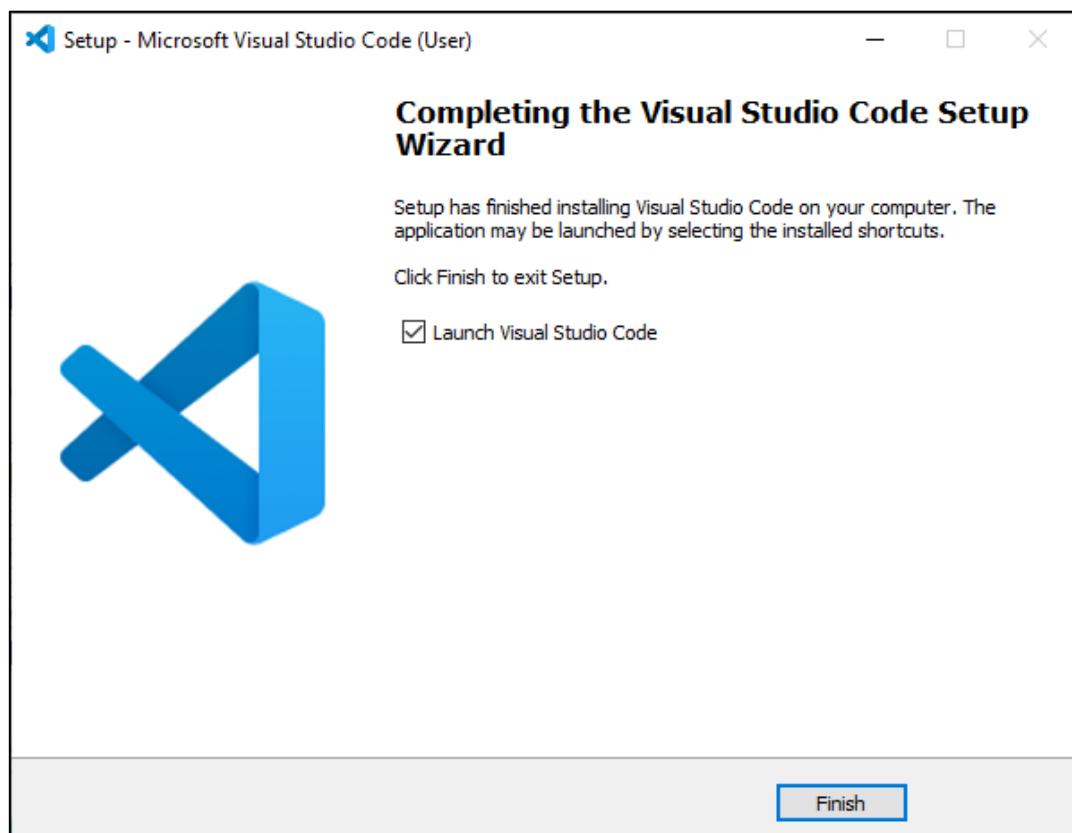
The installation location can be left as the default or changed to a desired path. After that, proceed by clicking Next repeatedly.



On this screen, verify that "Add to PATH" is selected. If unchecked, enable it. Proceed by clicking Next repeatedly to finish the installation.



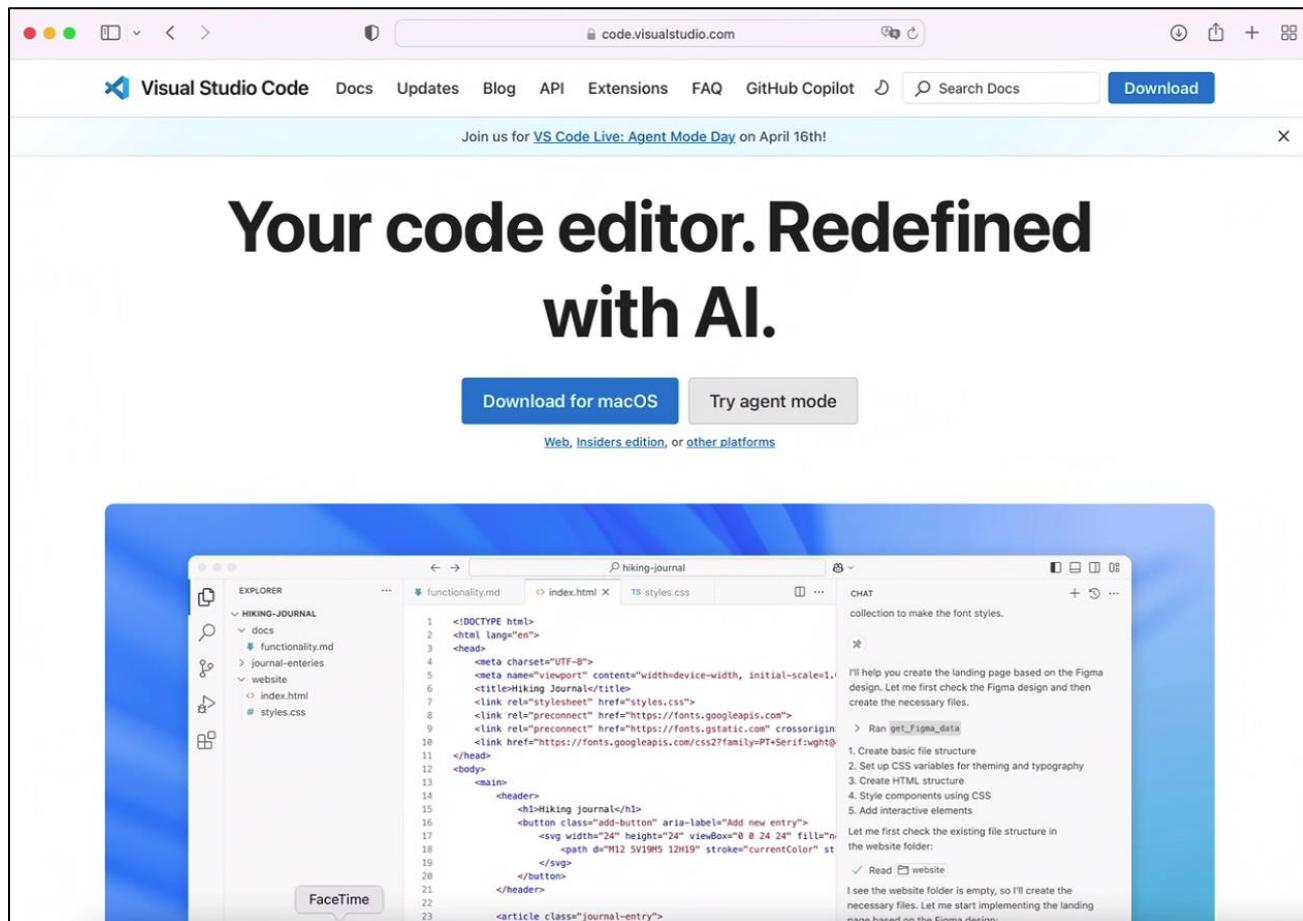
The installation is now complete, as shown in the image below.



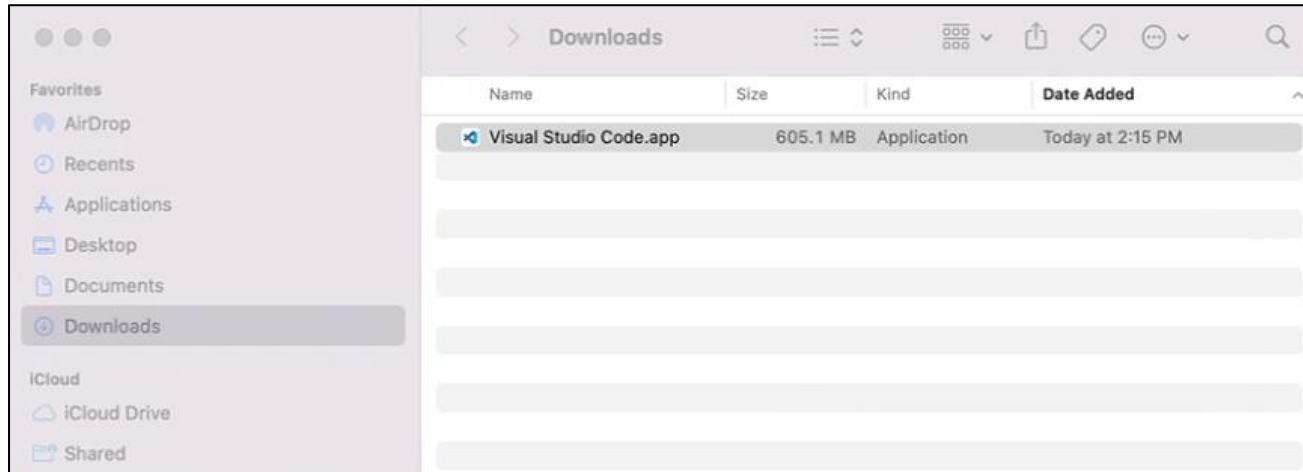
## Mac

Typically, MacOS comes with Visual Studio Code pre-installed. If your computer does not have it, please install it first.

Visit <https://code.visualstudio.com> and click "Download for macOS".



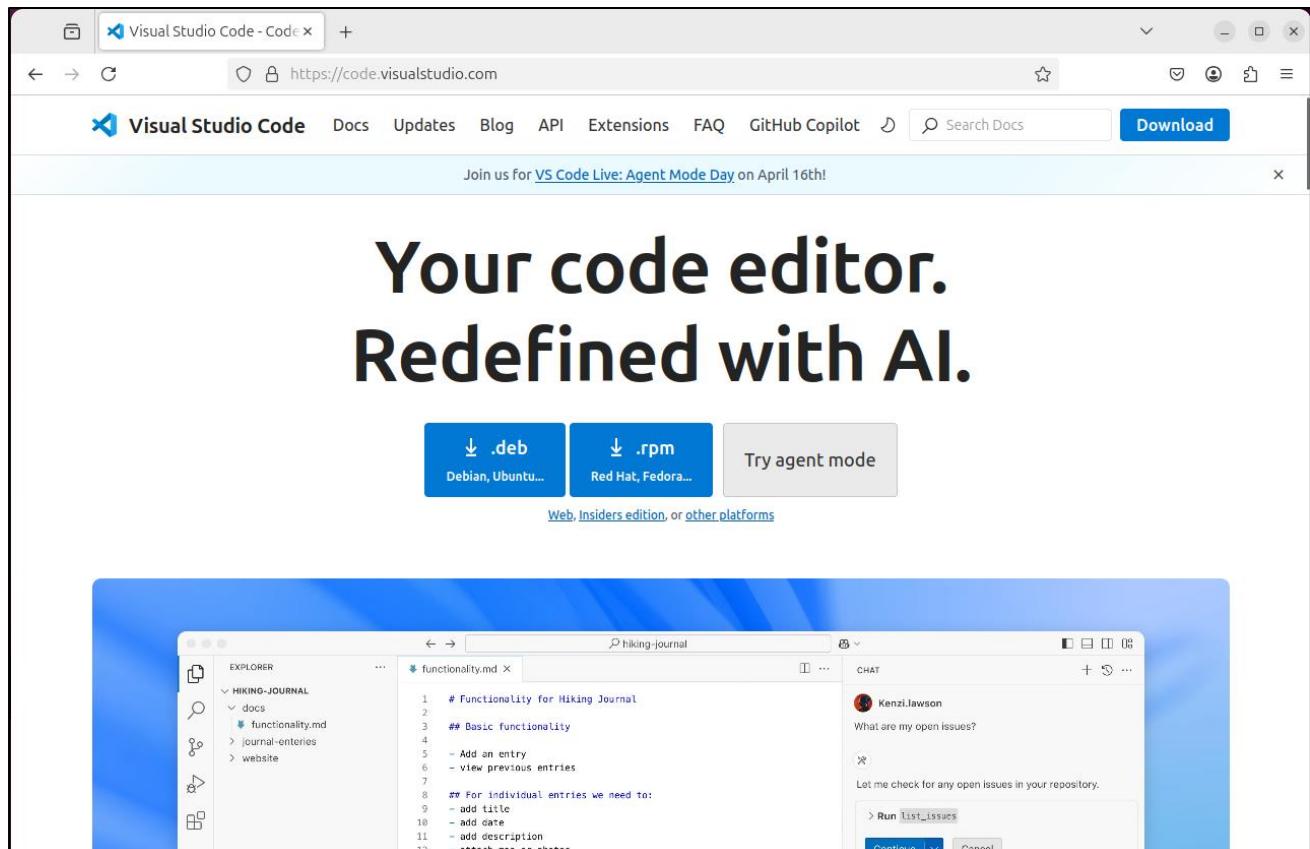
Double click to run the program.



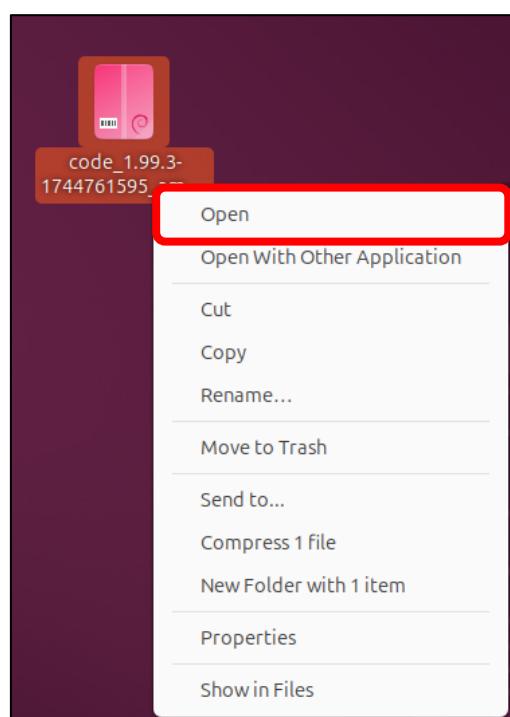
## Linux

If your computer does not have Visual Studio Code, please install it first.

Visit <https://code.visualstudio.com> and click ".deb".

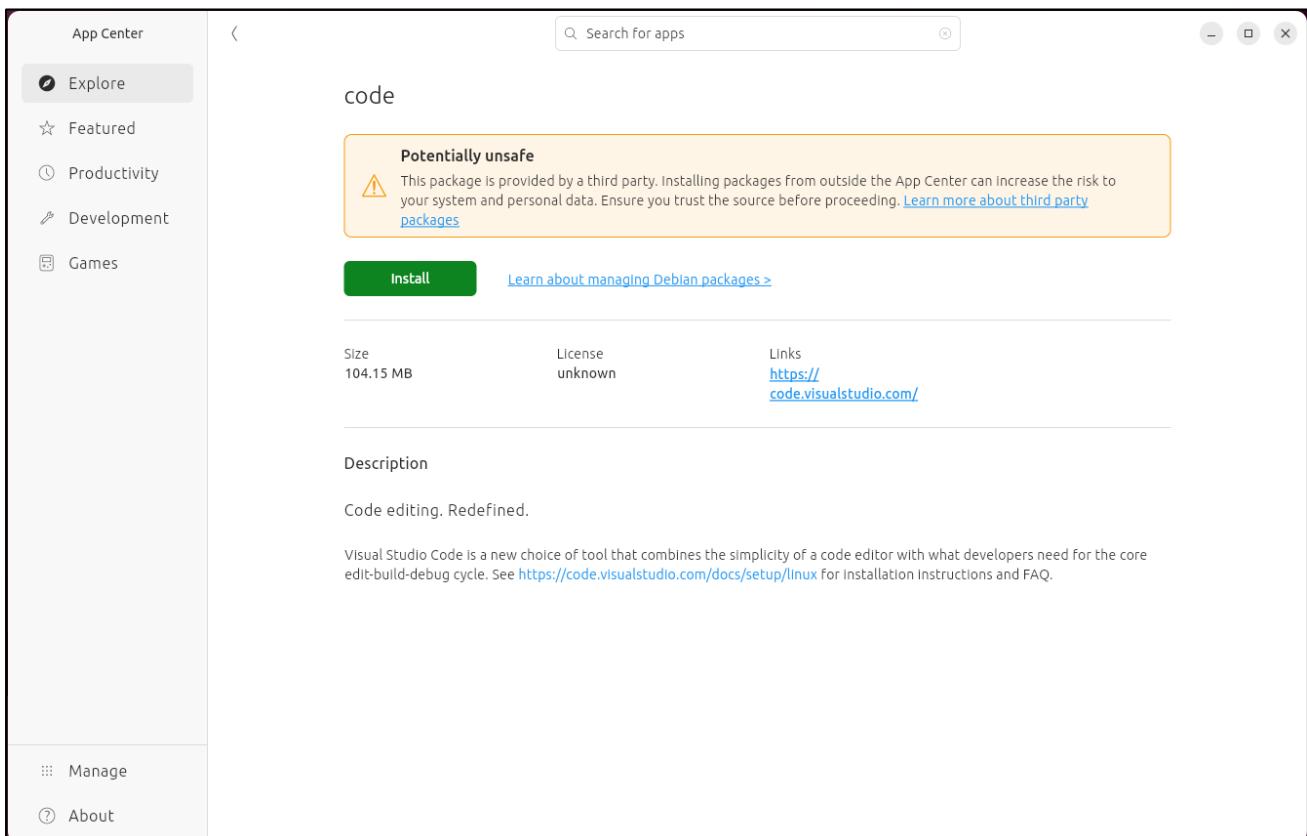


Open the downloaded "code\_xxx.deb" file.

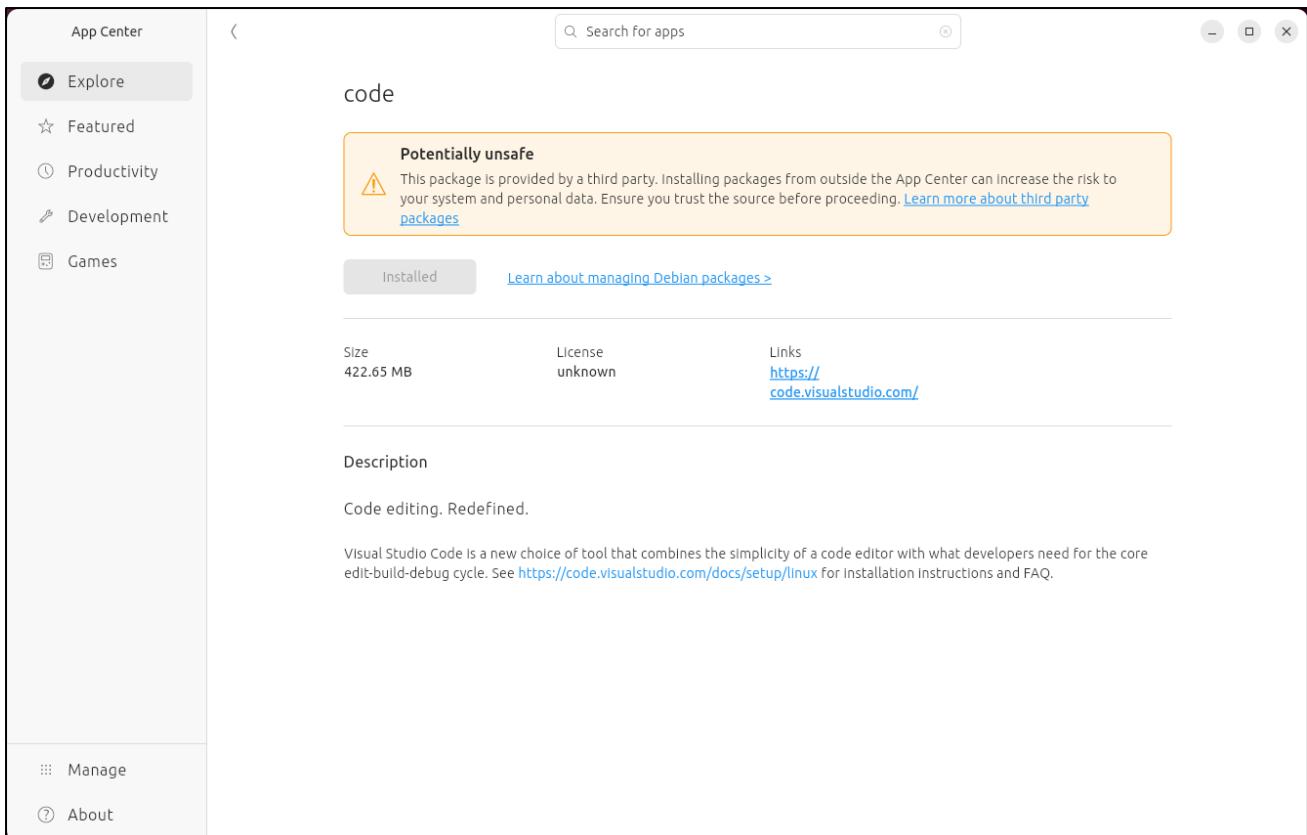




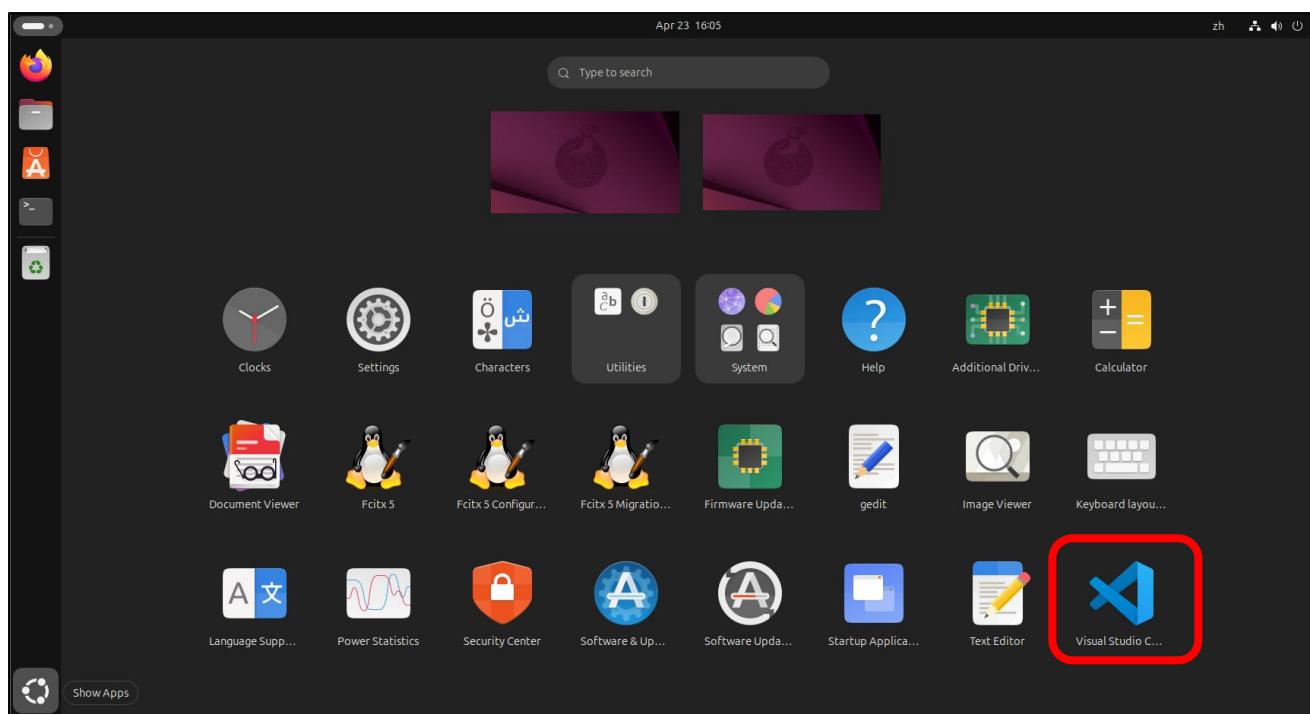
Click “Install” to install Visual Studio Code.



Wait for the installation to complete. Once finished, it should look like the image below.



Click Show Apps and you can see the Visual Studio Code is in the system.



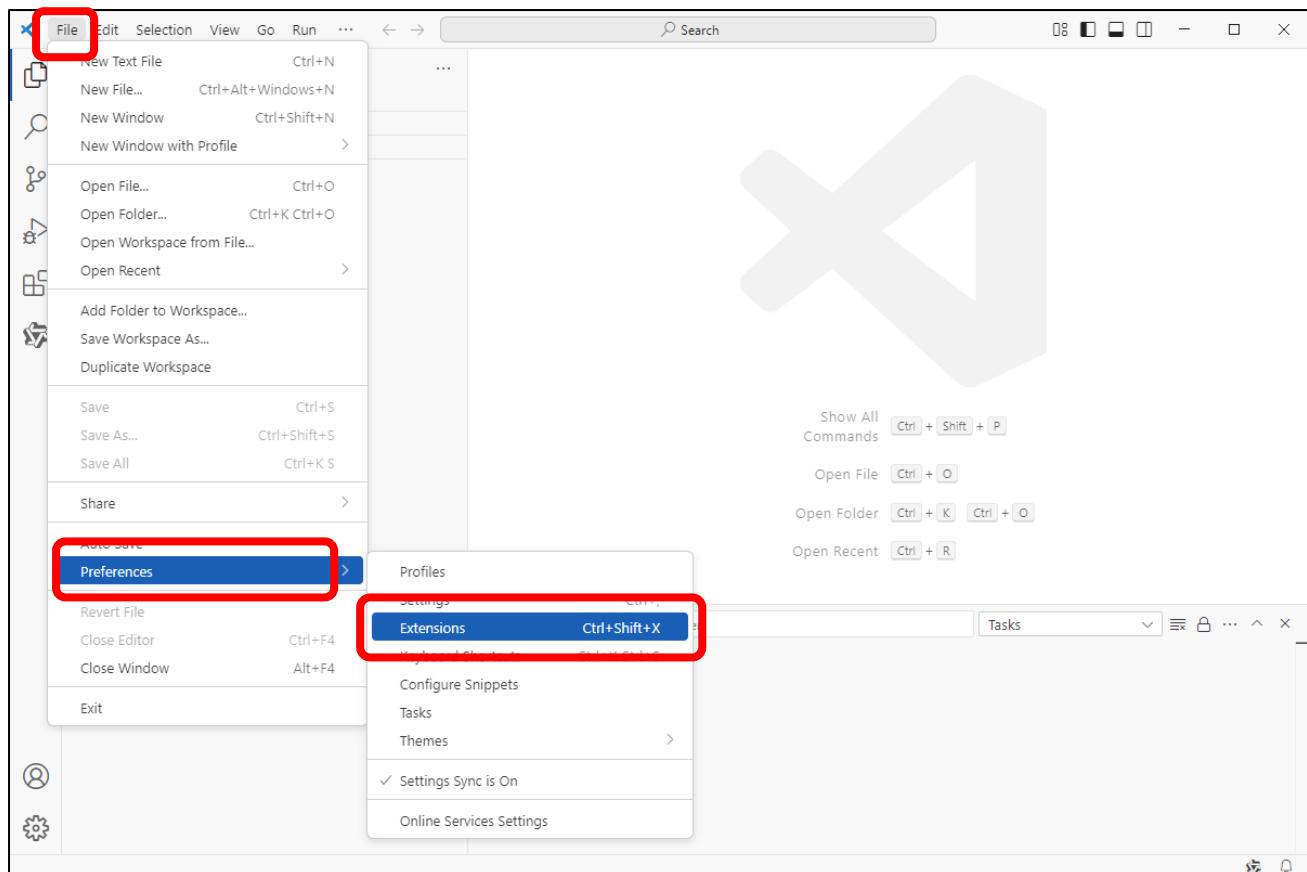


## Installing ESP-IDF V5.3.2

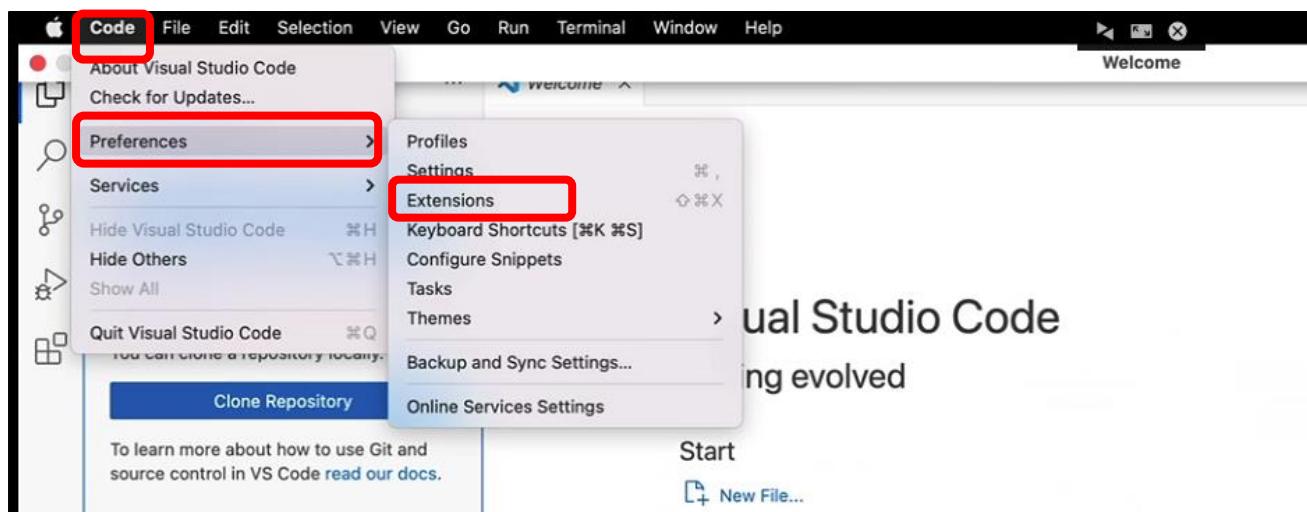
Visual Studio Code is a versatile code editor. To program with the ESP-IDF SDK, we need to install the ESP-IDF extension for it.

Open Visual Studio Code.

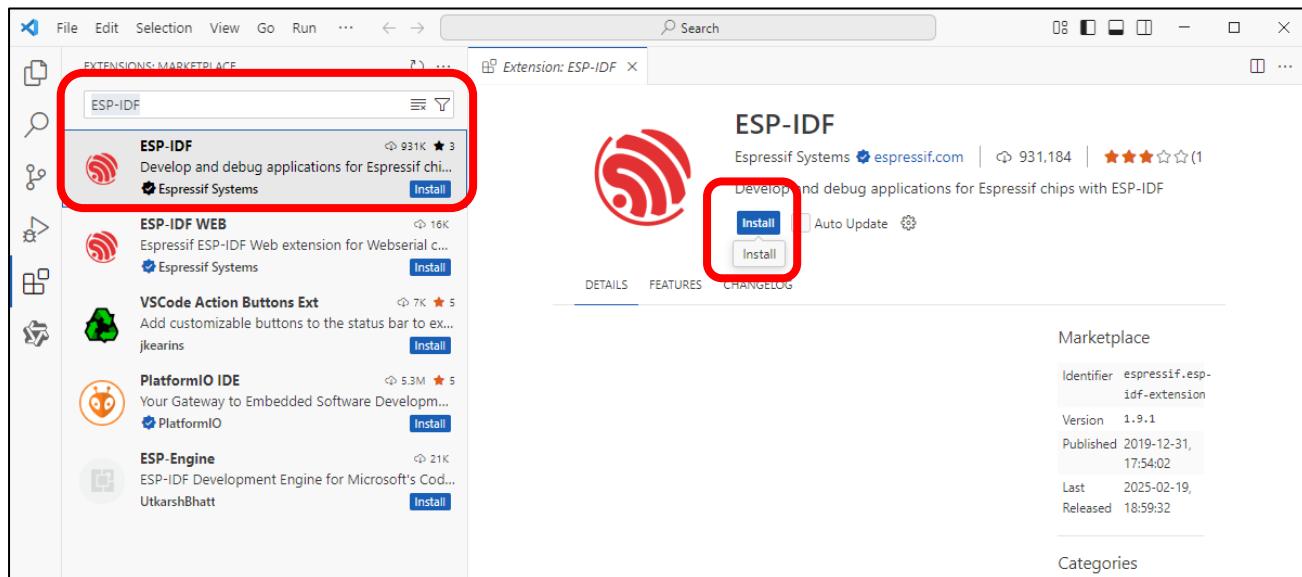
Click on the menu bar: File -> Preferences -> Extensions..



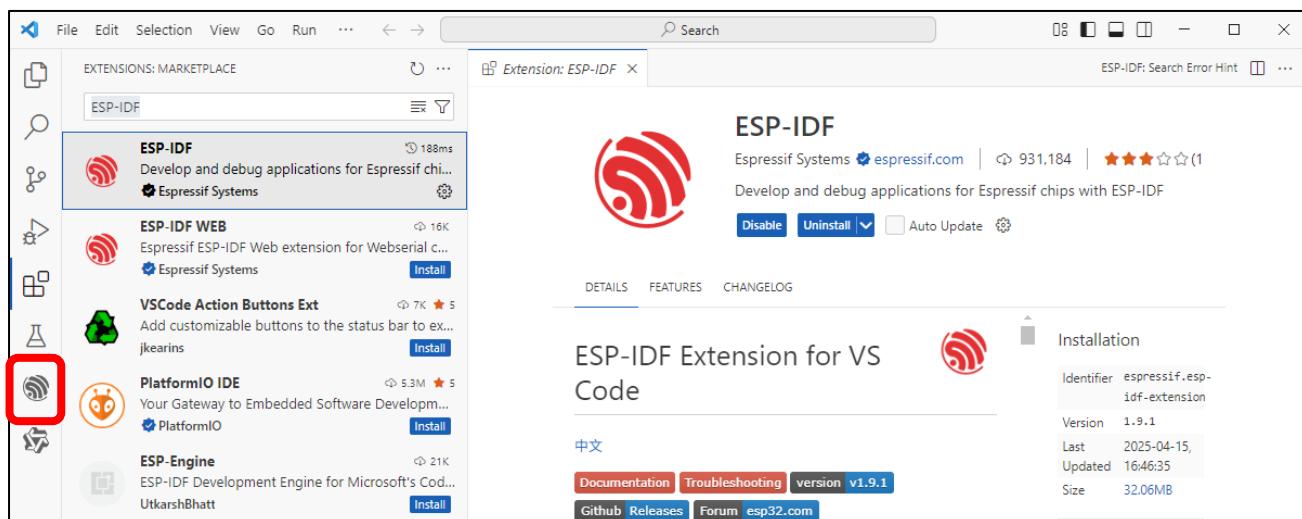
Mac OS: Click "Code" -> "Preferences" -> "Extensions" on the menu bar.



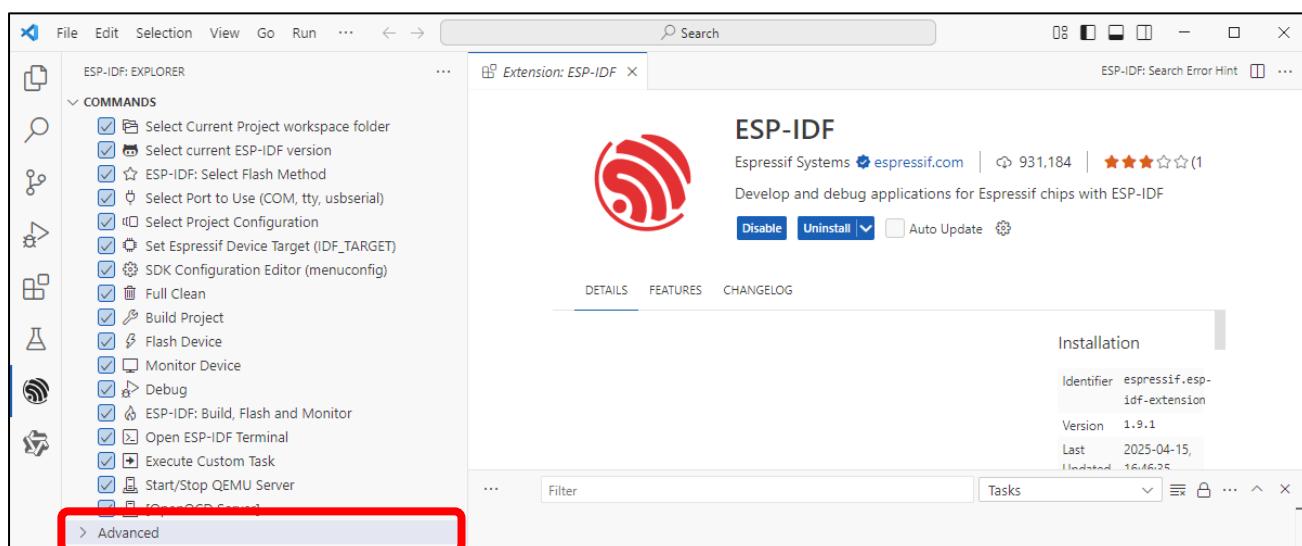
Search for "ESP-IDF" in the extension bar, select the correct result from the list, then click the Install button to proceed.



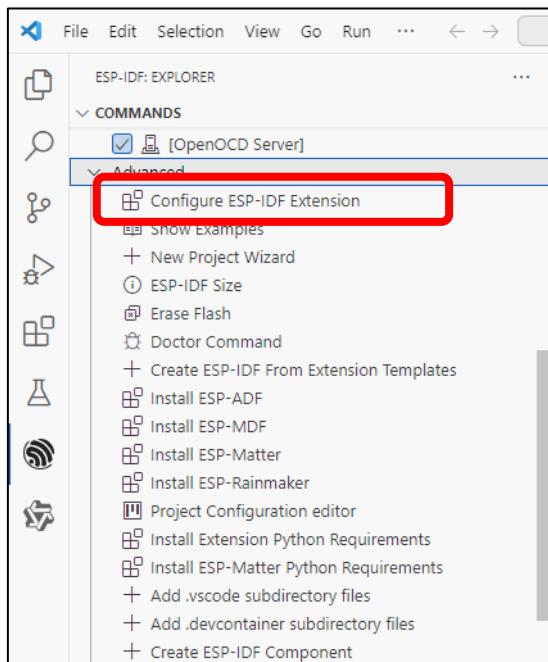
The ESP-IDF extension icon will now appear in the left sidebar - click it to continue.



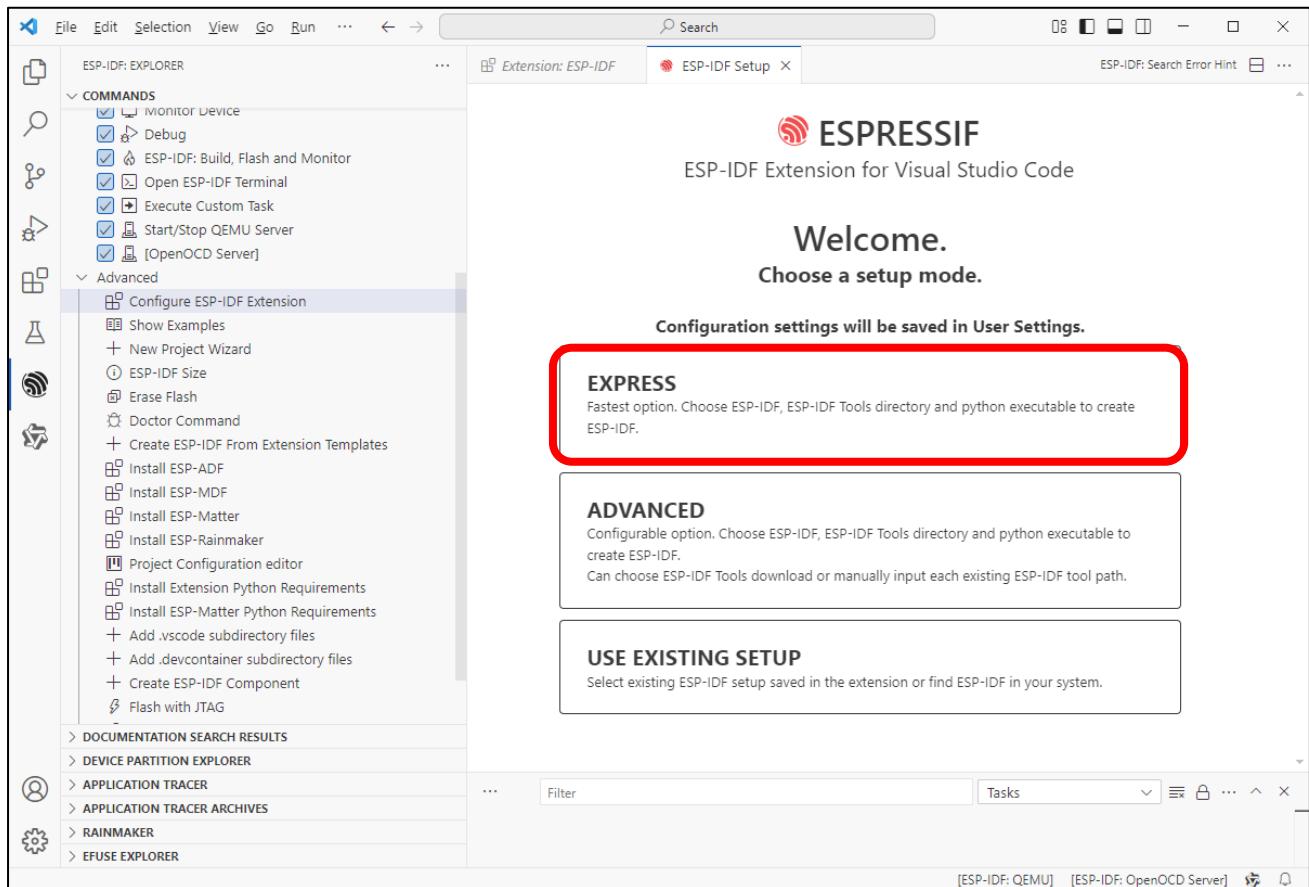
Scroll down with your mouse, locate and click on the "Advanced" option.



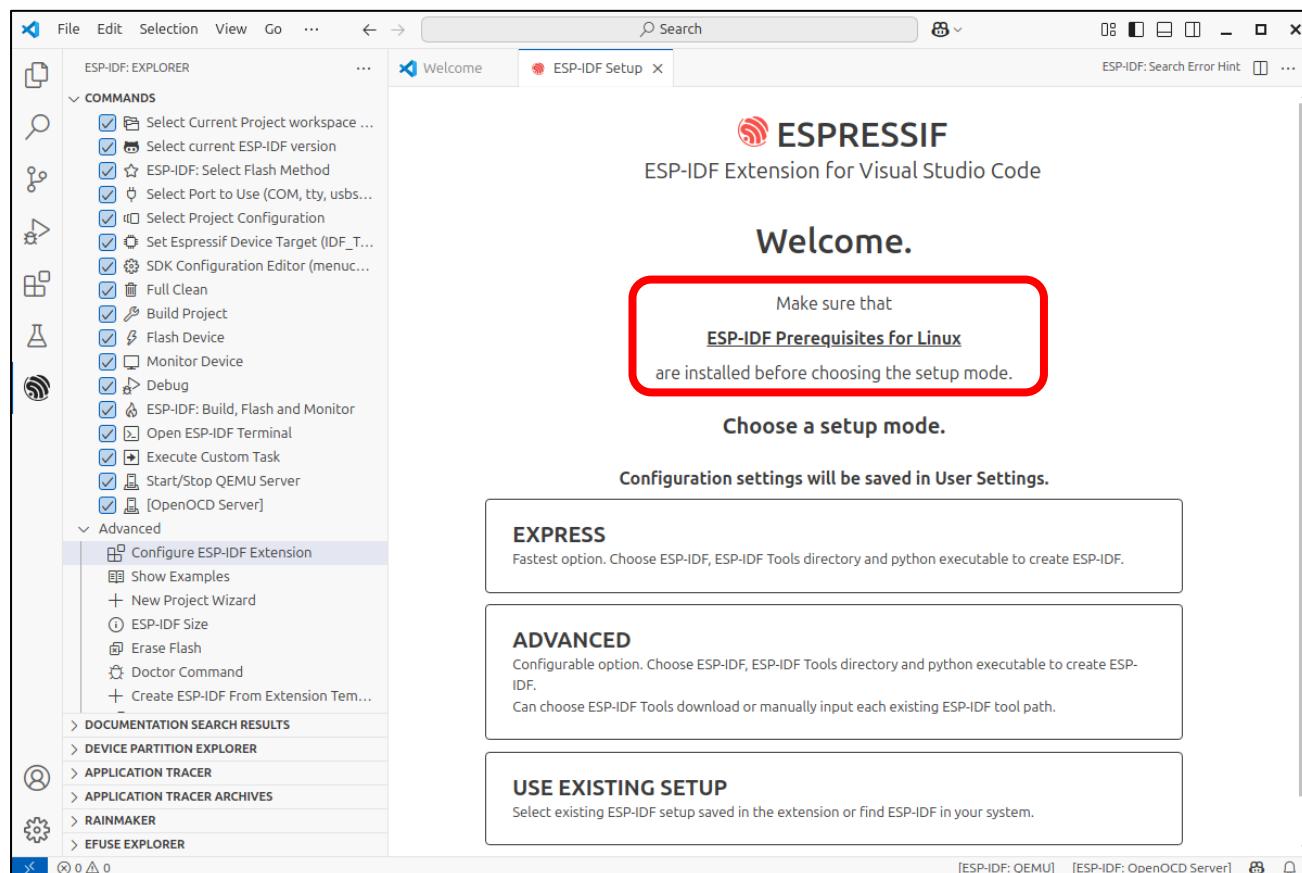
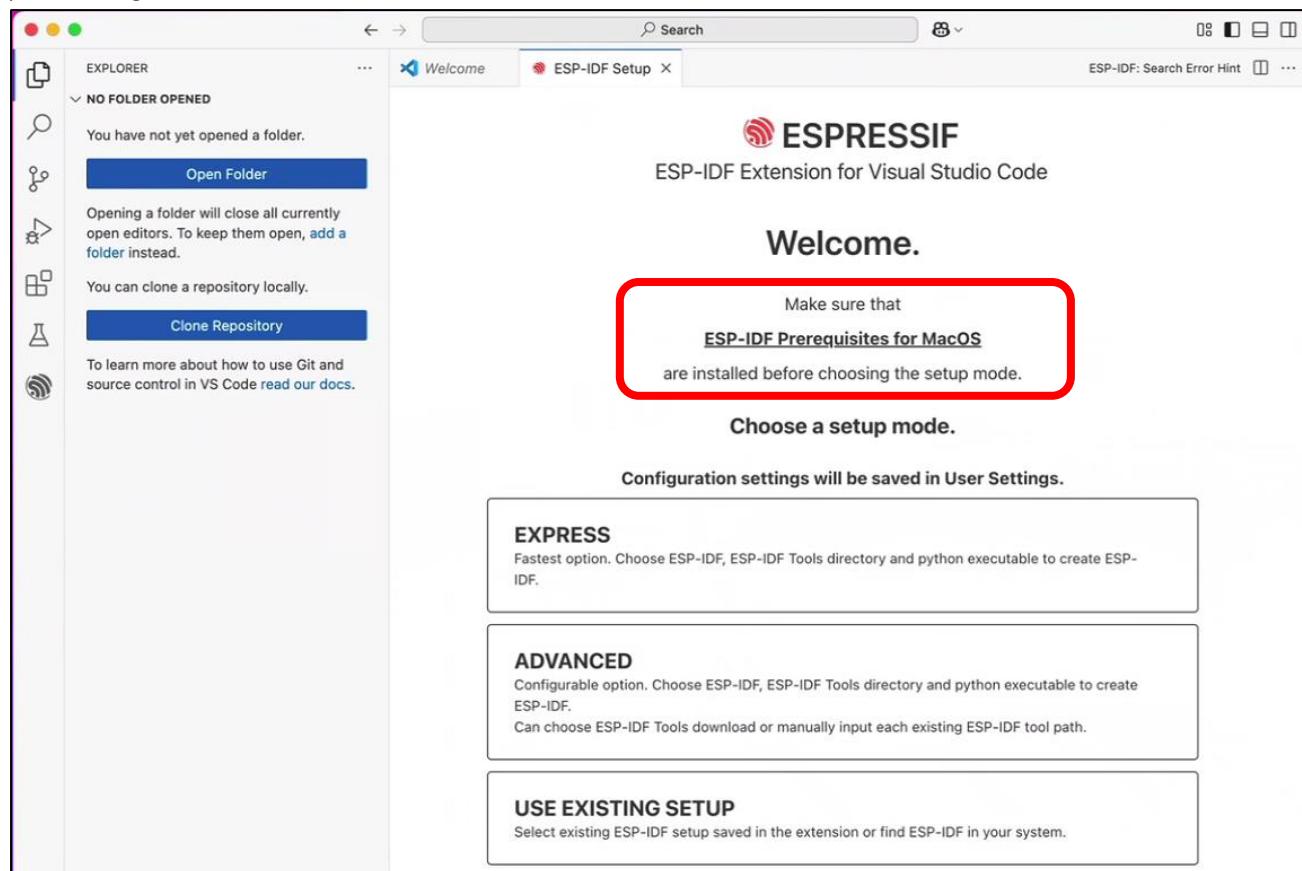
Click the first option: "Configure ESP-IDF Extension".



Select "EXPRESS" on the right.

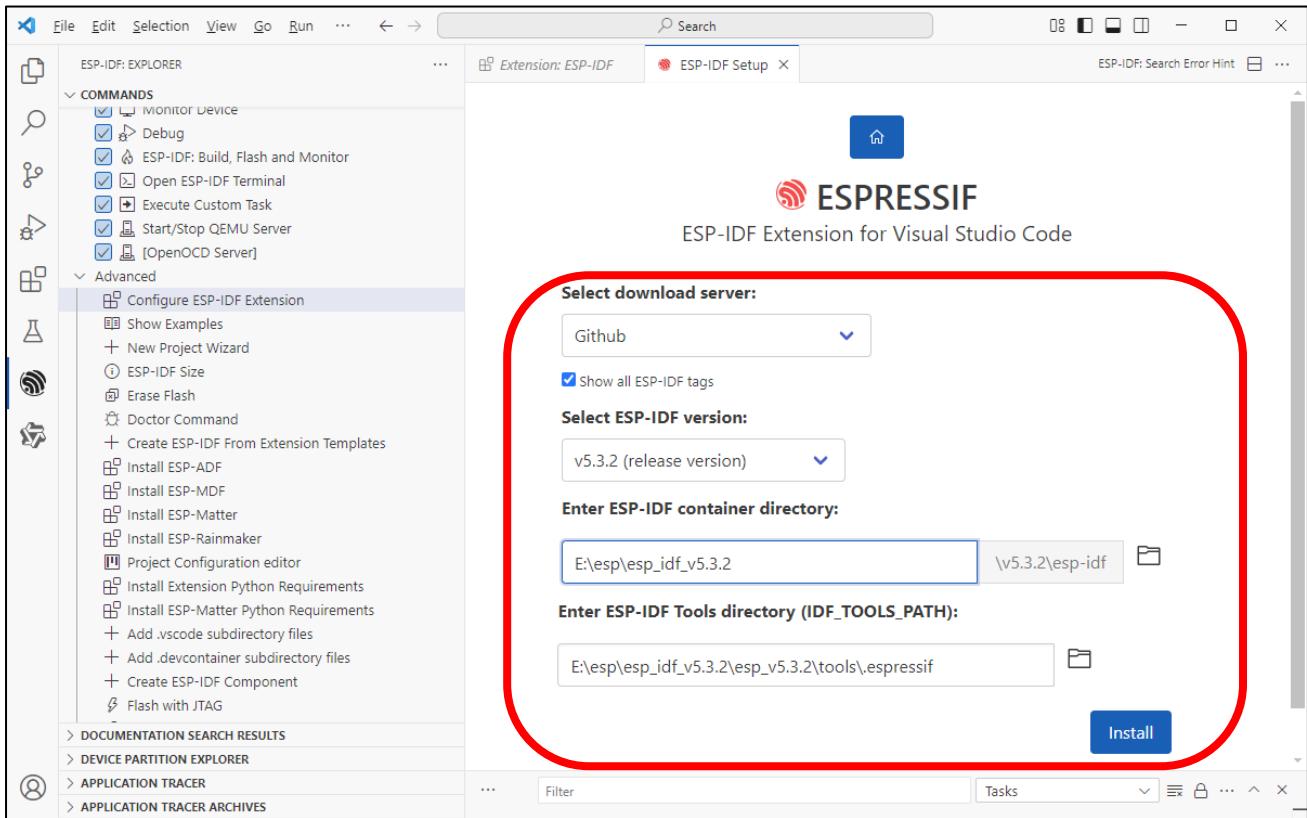


Note: If you're using macOS or Ubuntu, please complete the necessary preparations as prompted before proceeding with installation.



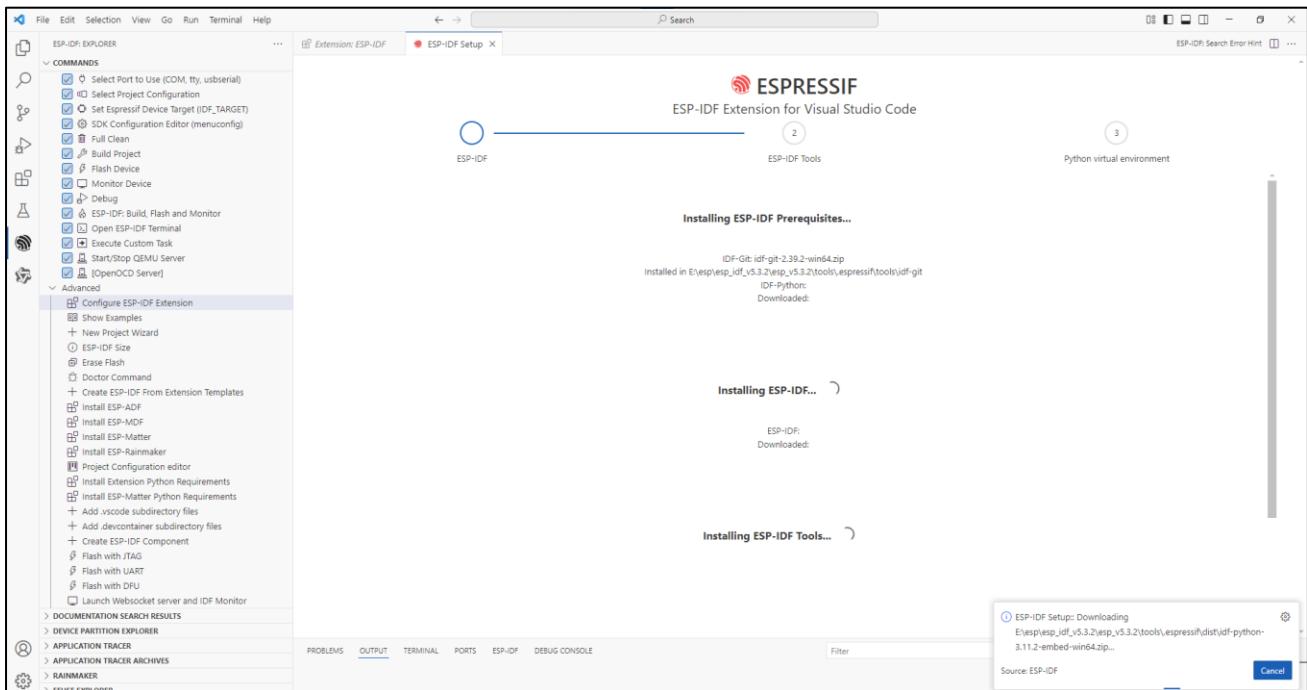


1. Check the box for "Show all ESP-IDF tags"
2. Select "v5.3.2 (release version)" from the dropdown
3. Choose your desired installation path for the ESP-IDF environment
4. Click "Install" to begin the setup



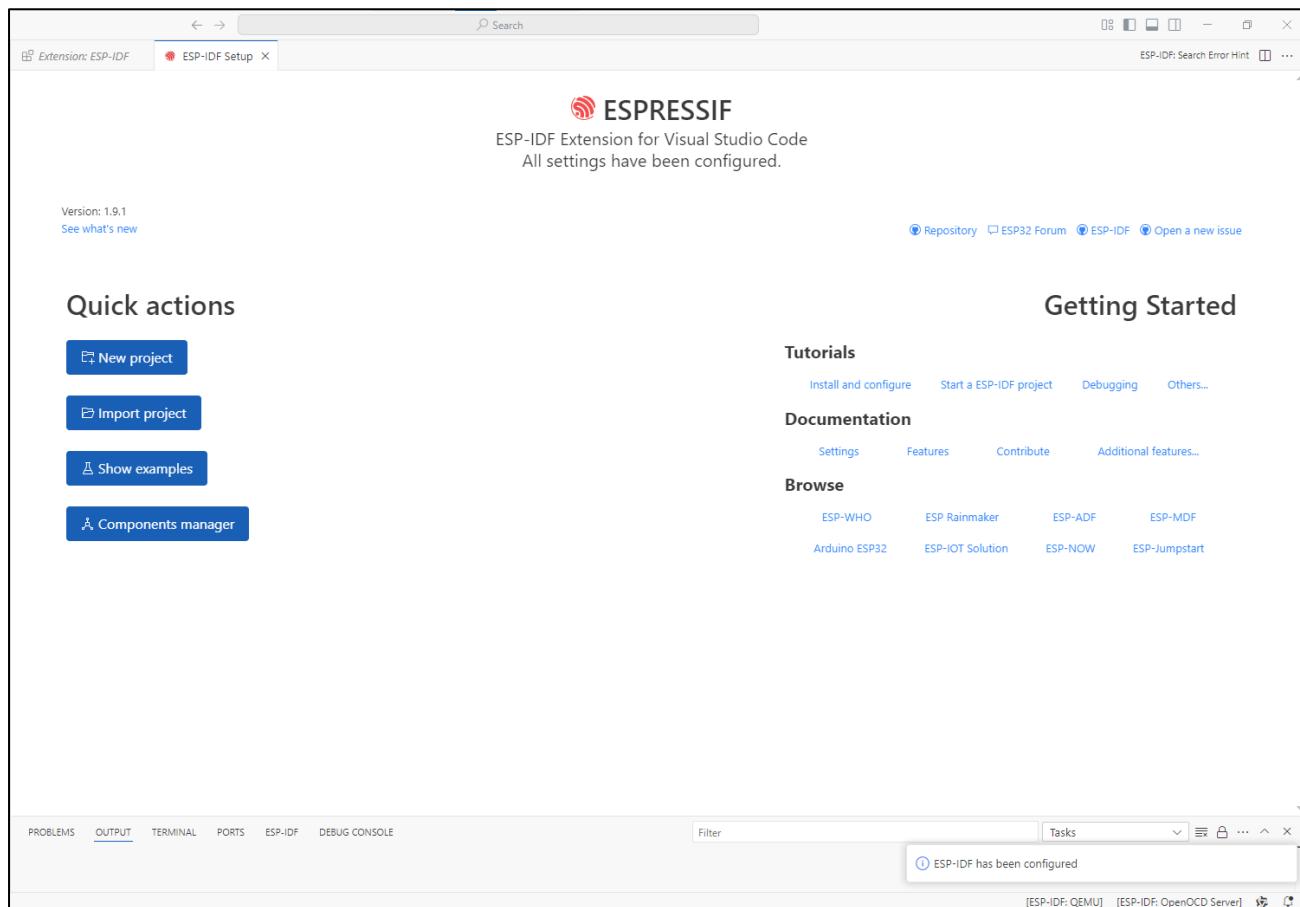
The process will complete automatically.

If it failed, locate your chosen ESP-IDF directory, remove the failed installation folder and install it again.



This step may take a while, so please ensure you have a stable and fast internet connection.

The complete installation is as shown below.



For more about ESP-IDF, please refer to

<https://docs.espressif.com/projects/vscode-esp-idf-extension/en/latest/installation.html>



## Code Downloading

### Windows

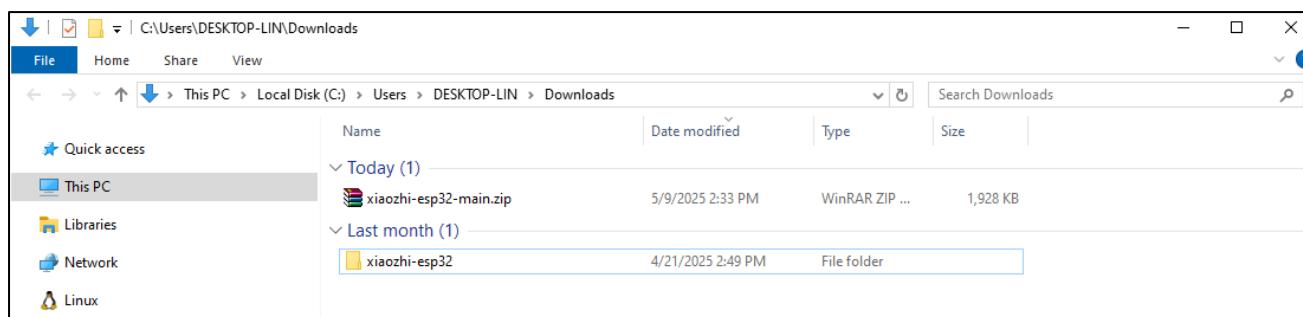
Open a browser on your computer and enter "<https://github.com/Freenove/xiaozhi-esp32>".

The screenshot shows a GitHub repository page for 'xiaozhi-esp32'. At the top, there's a navigation bar with links for 'Code', 'Pull requests', 'Actions', 'Projects', 'Wiki', 'Security', 'Insights', and 'Settings'. Below the navigation bar, the repository name 'xiaozhi-esp32' is displayed as 'Public' and 'forked from 78/xiaozhi-esp32'. There are buttons for 'Watch', 'Fork', and 'Star'. Below these, there are dropdown menus for 'main' branch, '1 Branch', and '0 Tags'. A search bar says 'Go to file' and a 'Code' button is highlighted in green. To the right, there's an 'About' section.

Click "Code" -> "Download ZIP" to download the code to your computer.

This screenshot is similar to the previous one, showing the same GitHub repository page for 'xiaozhi-esp32'. However, the 'Code' dropdown menu is now open, revealing options like 'Local', 'Codespaces', 'Clone' (with 'HTTPS', 'SSH', and 'GitHub CLI' sub-options), 'Open with GitHub Desktop', and 'Download ZIP'. The 'Download ZIP' option is highlighted with a blue box. On the right side of the page, there's an 'About' section with information about the repository, including links to 'xiaozhi.me', 'Readme', 'MIT license', 'Activity', 'Custom properties', 'stars', 'watching', and 'forks'. There's also a 'Report repository' link and a 'Releases' section indicating 'No releases published'.

Extract the downloaded zip file to your computer. **Rename the decompressed folder to "xiaozihi-esp32".**



## Mac

Open the terminal and download the code with the git command.

```
git clone https://github.com/Freenove/xiaozihi-esp32.git
```

```
[freenove@PandeMacBook-Air ~ % git clone https://github.com/Freenove/xiaozihi-esp32.git
Cloning into 'xiaozihi-esp32'...
remote: Enumerating objects: 4596, done.
remote: Counting objects: 100% (1285/1285), done.
remote: Compressing objects: 100% (167/167), done.
remote: Total 4596 (delta 1181), reused 1118 (delta 1118), pack-reused 3311 (from 1)
Receiving objects: 100% (4596/4596), 3.19 MiB | 4.63 MiB/s, done.
Resolving deltas: 100% (3189/3189), done.
freenove@PandeMacBook-Air ~ %
```

## Linux

Open the terminal and download the code with the git command.

```
git clone https://github.com/Freenove/xiaozihi-esp32.git
```

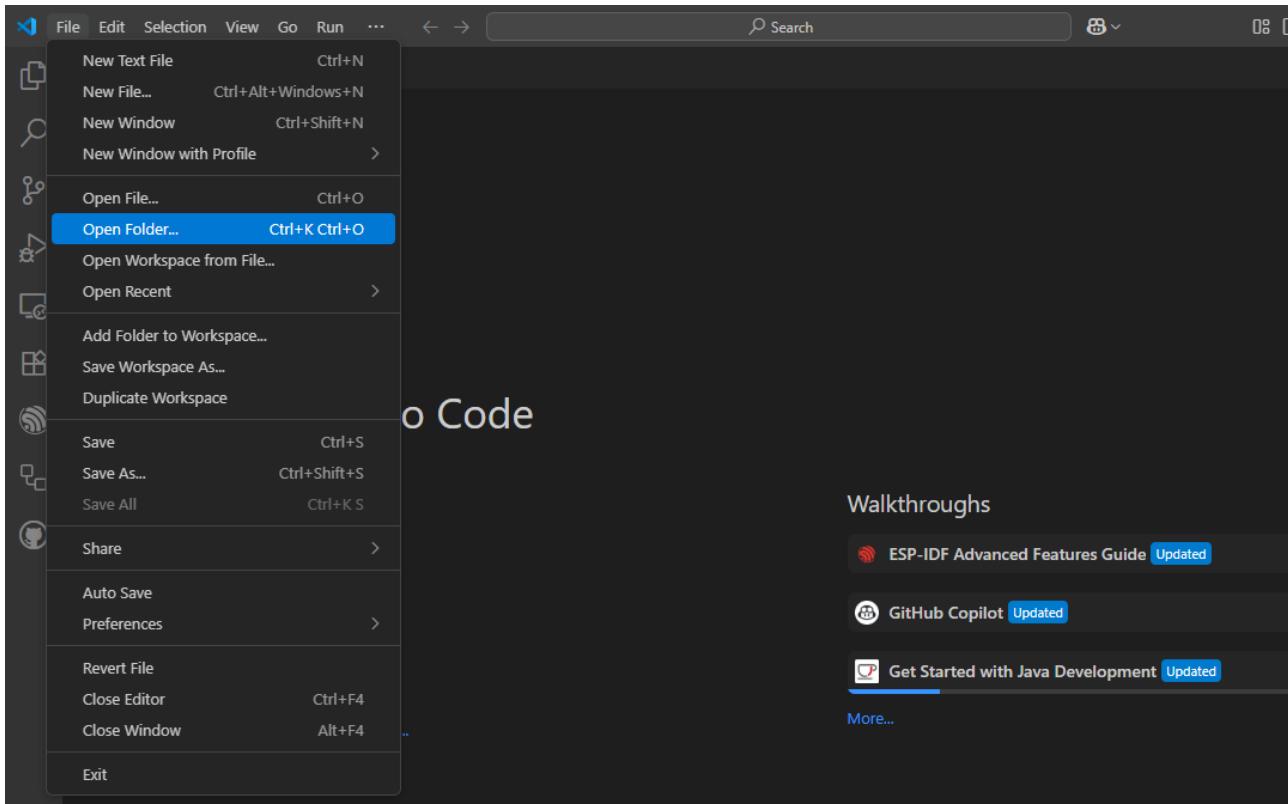
```
lin@ubuntu:~$ git clone https://github.com/Freenove/xiaozihi-esp32.git
Cloning into 'xiaozihi-esp32'...
remote: Enumerating objects: 4596, done.
remote: Counting objects: 100% (1285/1285), done.
remote: Compressing objects: 100% (167/167), done.
remote: Total 4596 (delta 1181), reused 1118 (delta 1118), pack-reused 3311 (from 1)
Receiving objects: 100% (4596/4596), 3.19 MiB | 4.84 MiB/s, done.
Resolving deltas: 100% (3189/3189), done.
lin@ubuntu:~$
```



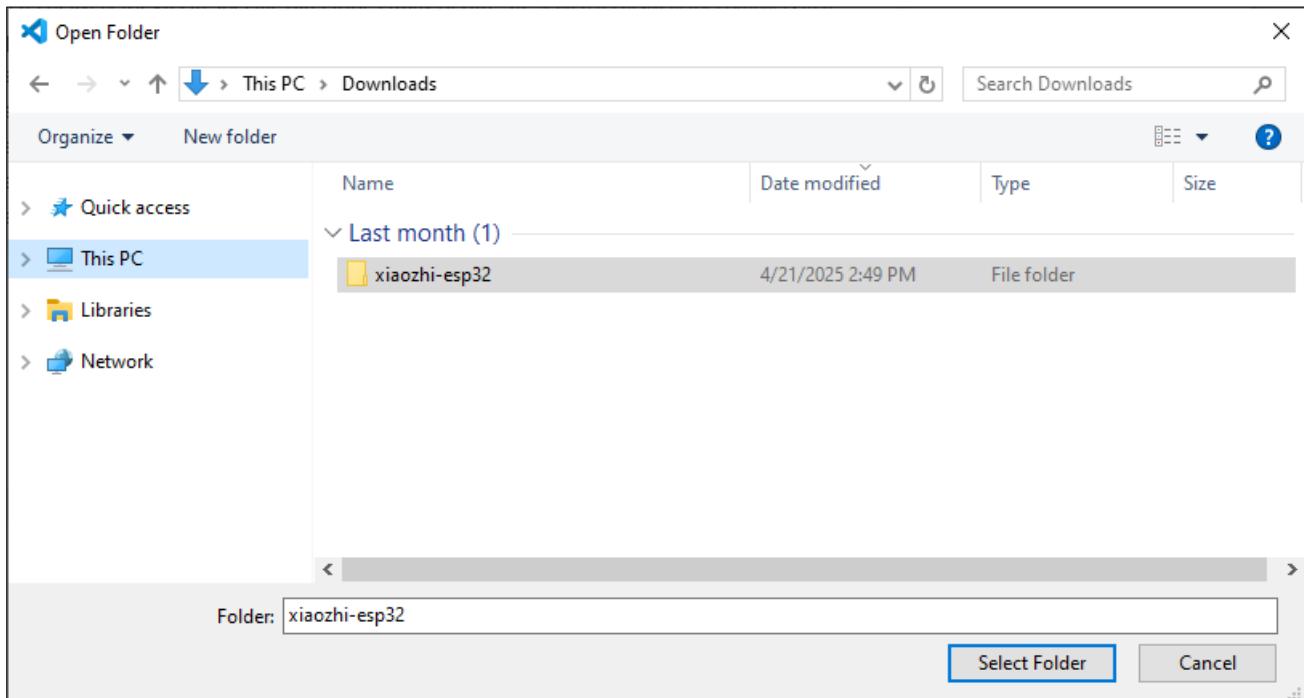
## Configure Code Environment

Extract the downloaded ZIP file.

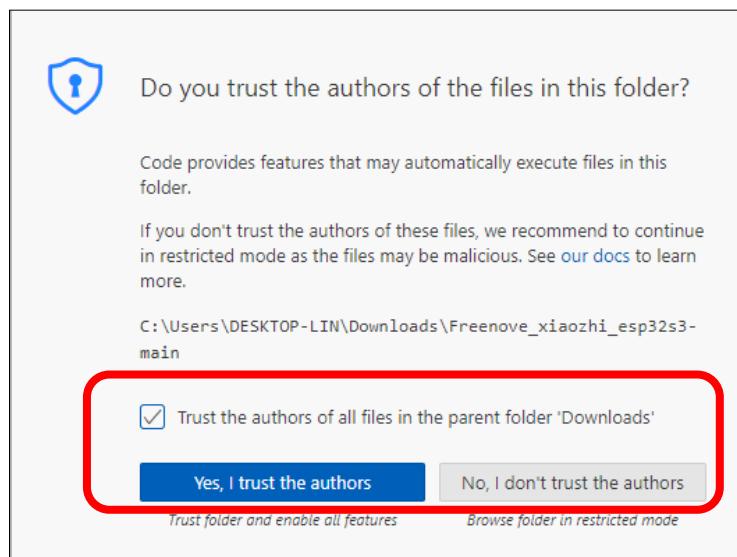
On Visual Studio Code, click "File" -> "Open Folder...".



Select the **xiaozhi-esp32** folder. Here, the interface of the Windows system is taken as an example. The operation of the mac system is similar to that of Linux.



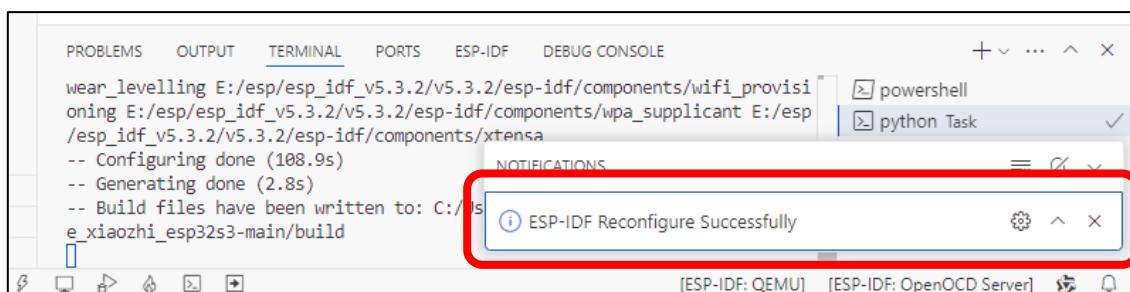
Check the box “Trust the authors of all files in the parent folder ‘Downloads’” and select “Yes, I trust the authors”.



Please note: A pop-up notification will appear in the lower-right corner. Click 'Generate compile\_commands.json', and it will download the corresponding component module code based on the file."



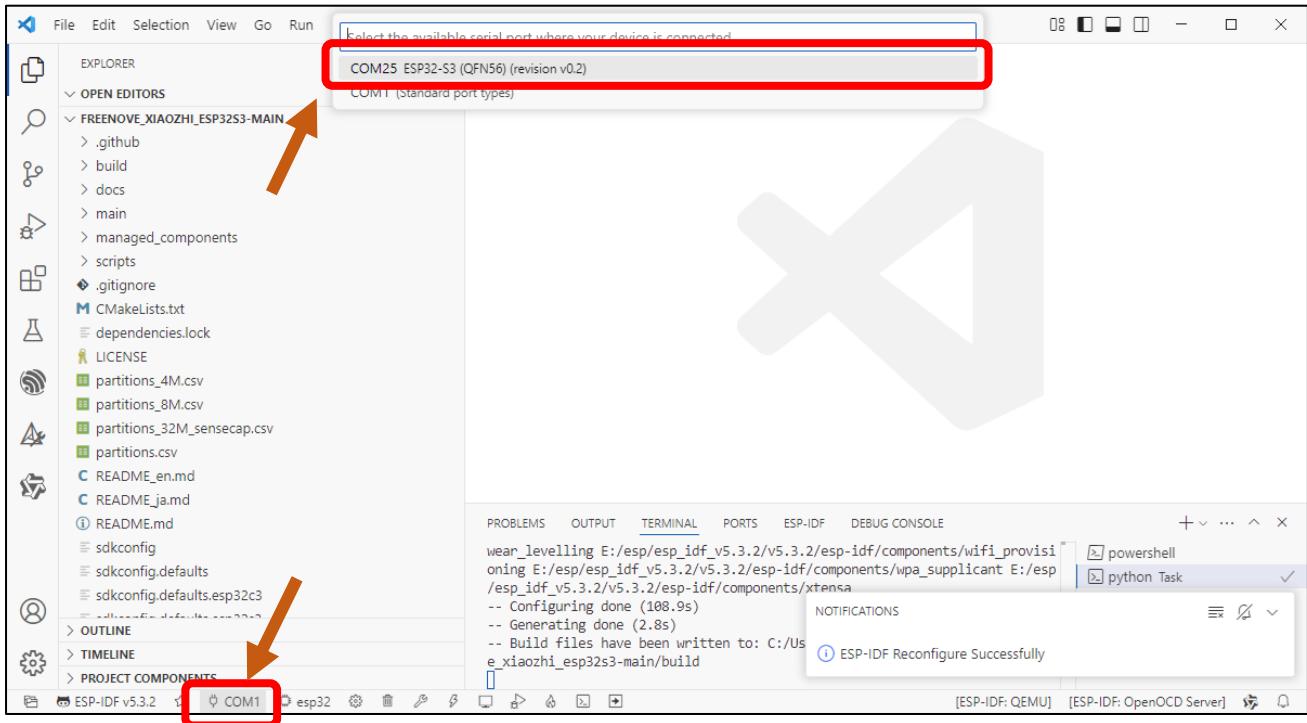
Component installation may take some time. Please wait and avoid other operations. A completion notification will appear in the lower-right corner once finished.



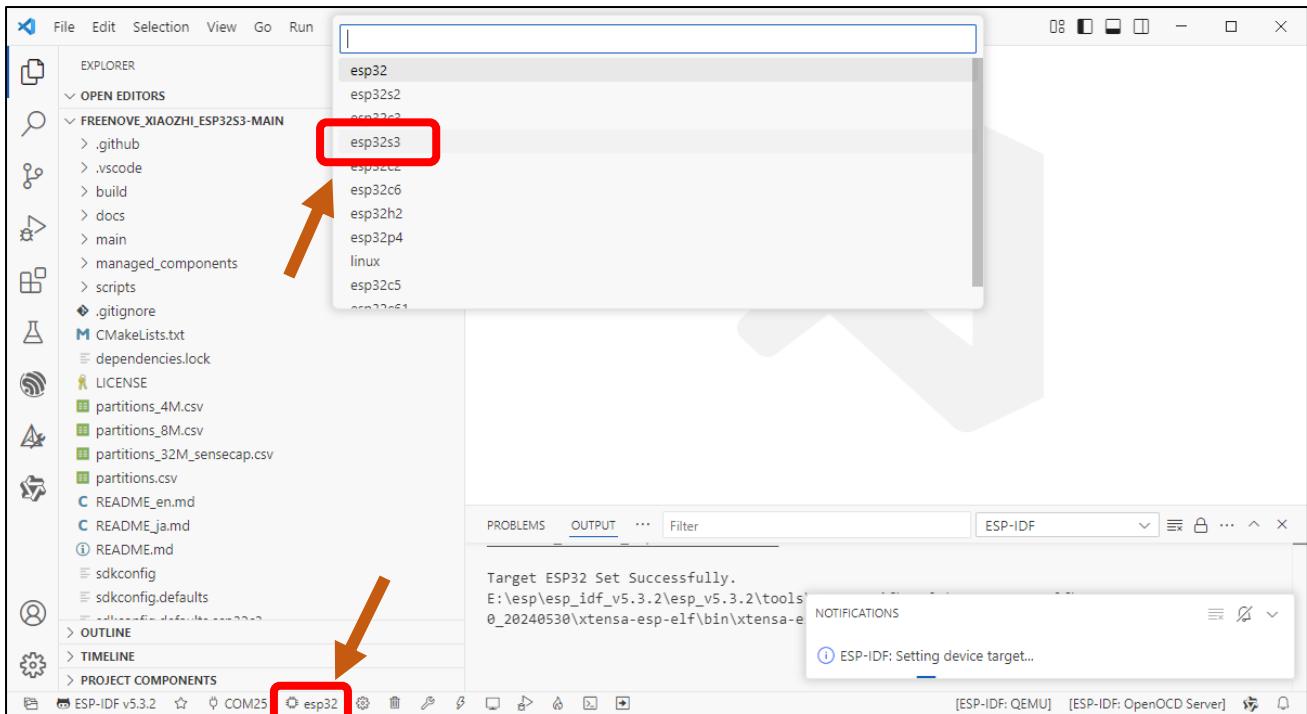
Connect the ESP32-S3-WROOM to your computer using a USB cable, making sure to plug it into the correct Type-C port (do not use the wrong connector)



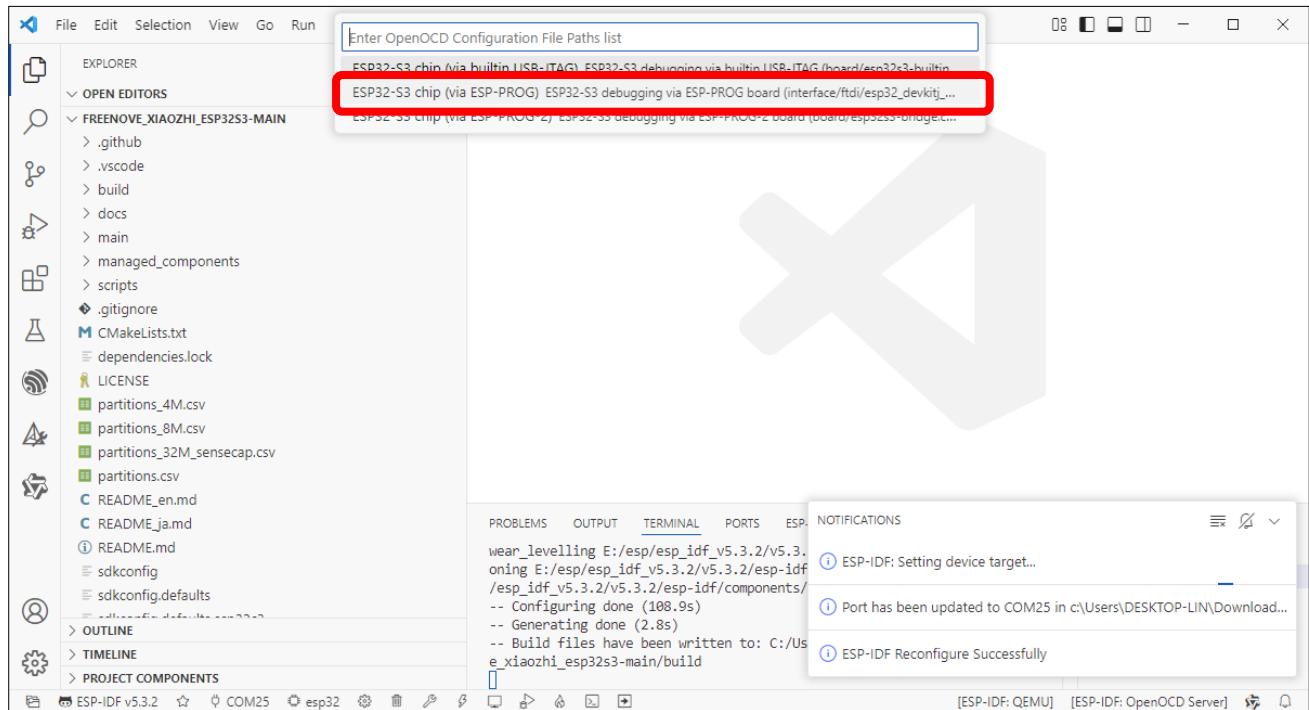
Click on '**COMx**' in the bottom-left corner to display all available COM ports on your computer. Locate and select the entry labeled '**ESP32-S3**'.



Click the '**ESP32**' button in the bottom-left corner to display all available ESP32 models, then select '**ESP32-S3**' from the list."

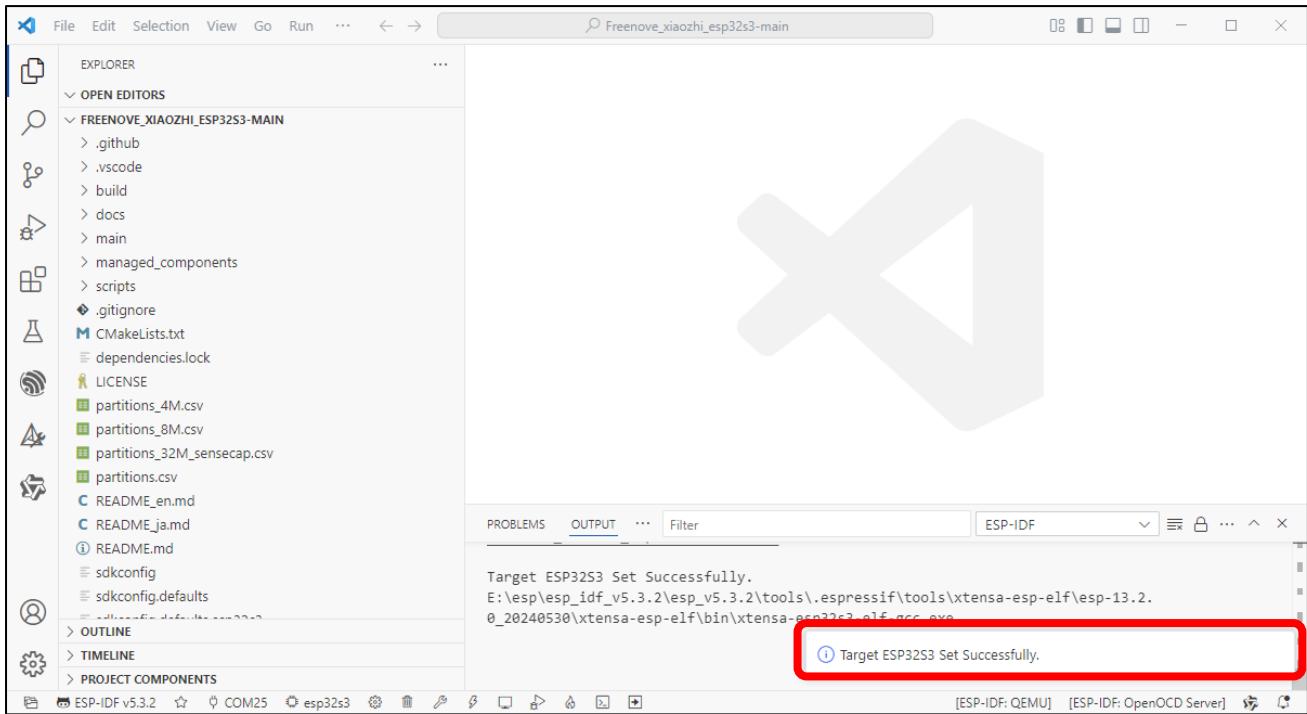


From the new selection menu, choose '**ESP32-S3 Chip (via ESP-PROG) - ESP32-S3 debugging via ESP-PROG Board...**'

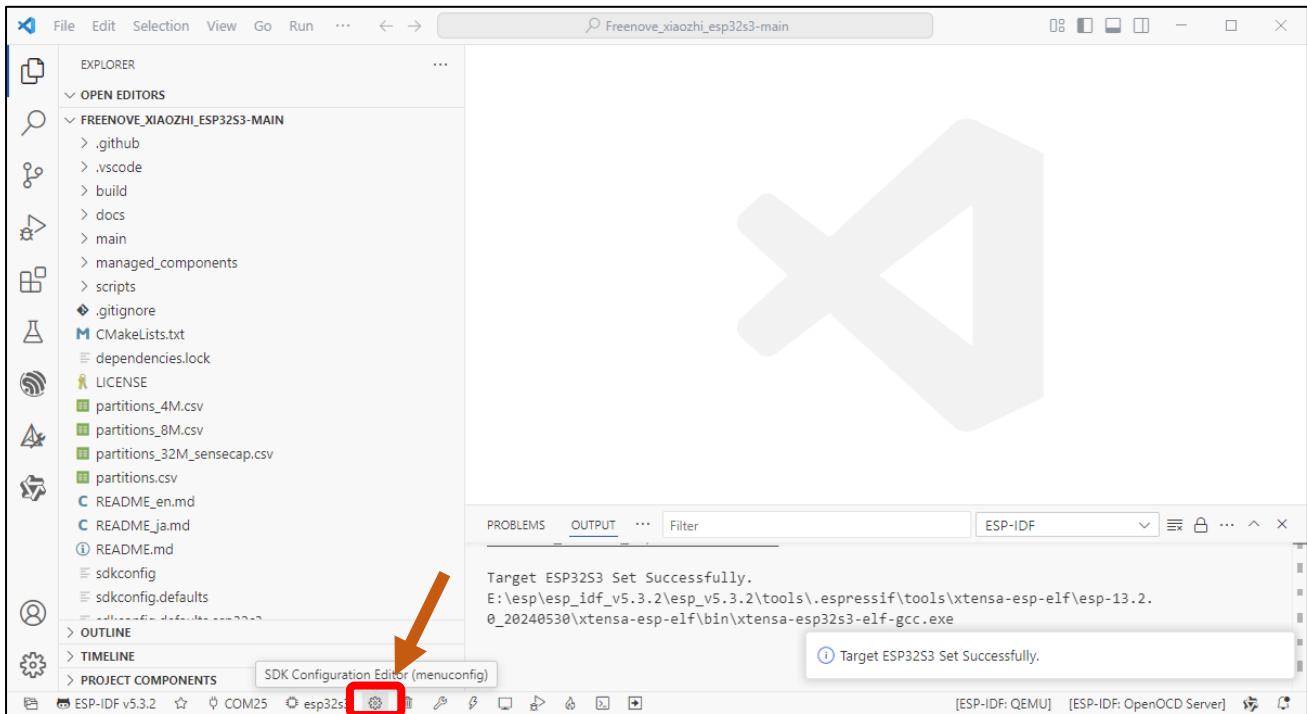




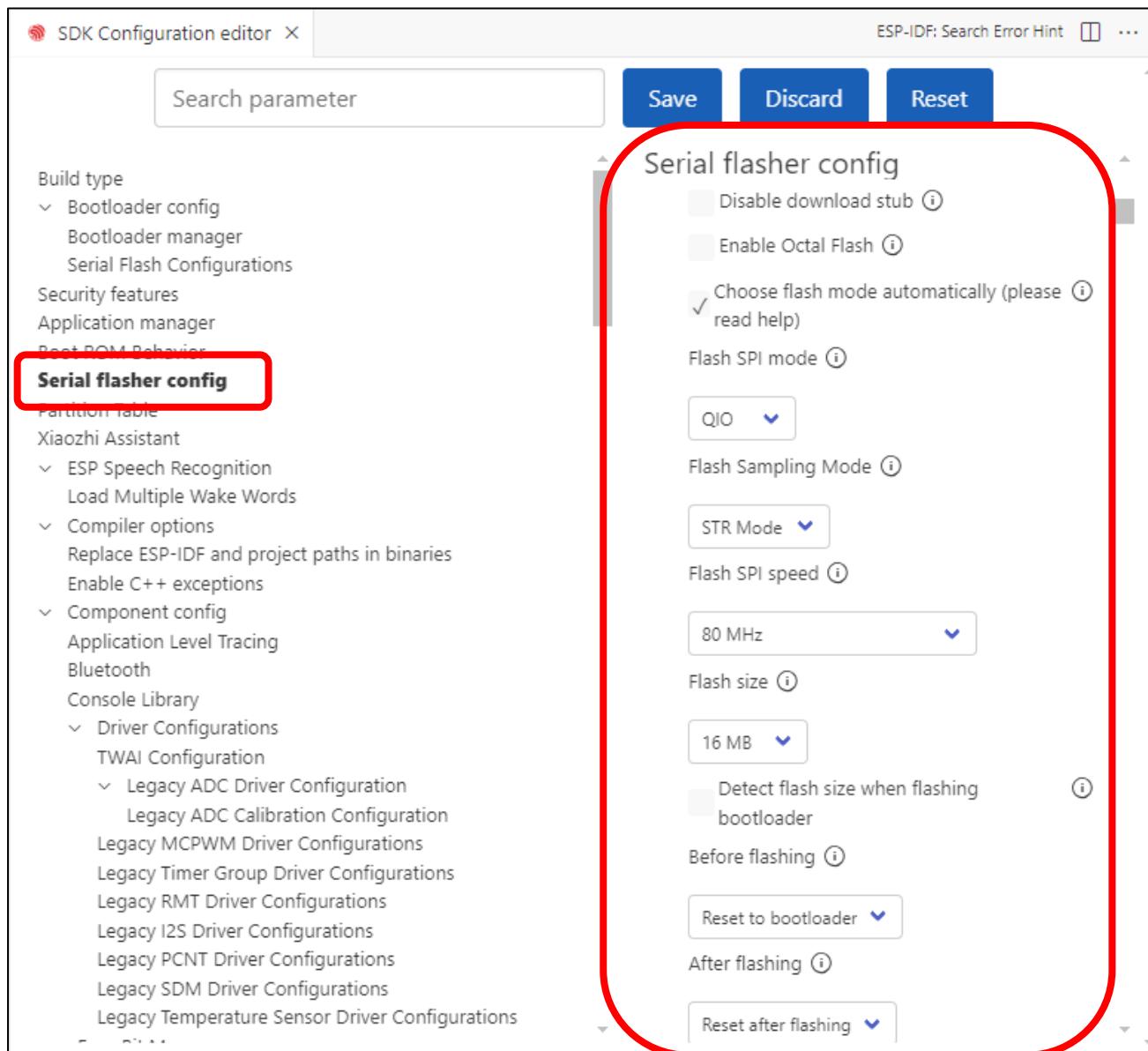
Wait until it shows “**Target ESP32S3 Set Successfully**” at the bottom right.



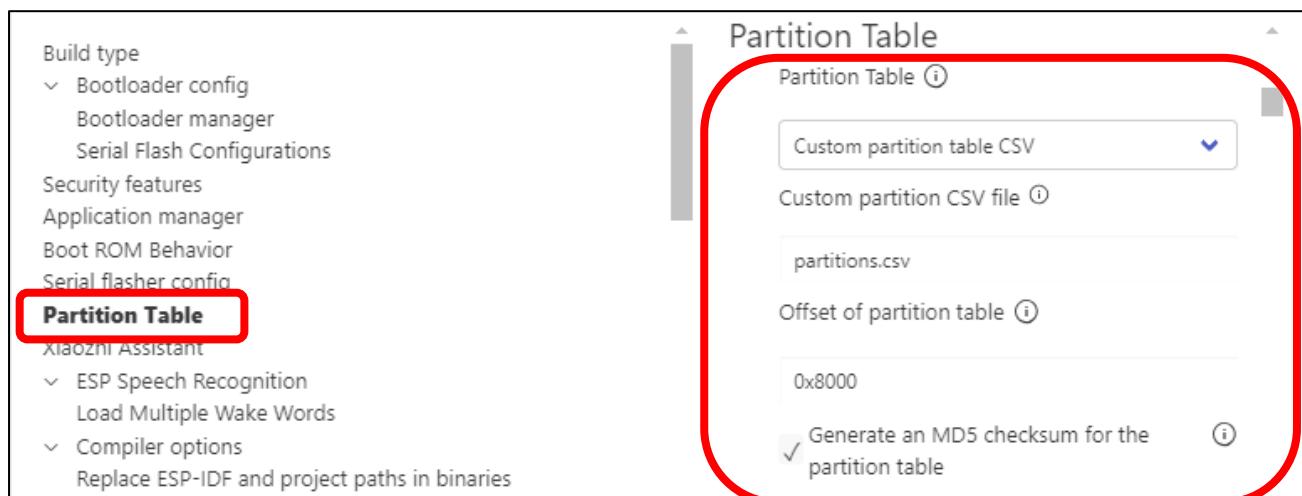
Click “SDK Configuration Editor (menuconfig)” at the bottom.



On the new interface, click 'Serial flasher config' and verify that the settings match the configuration shown in the image below.



Click "Partition Table" and verify that the settings match the configuration shown in the image below.



Click “Xiao Assistant” and verify that the settings match the configuration shown in the image below.

**Xiao Assistant**

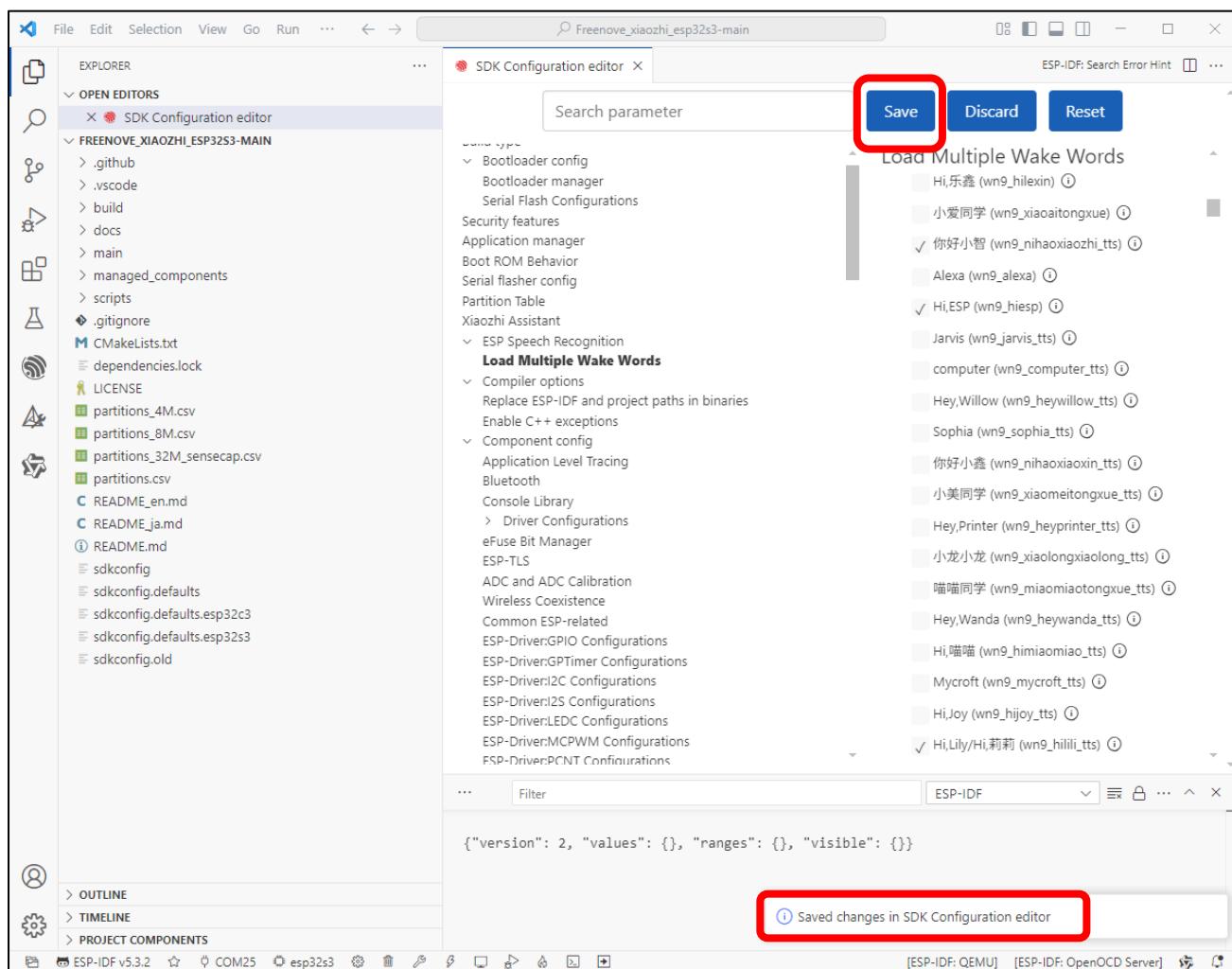
- Default OTA URL ⓘ
- https://api.tenclass.net/xiaozhi/ota/
- Default Language ⓘ
- English
- Board Type ⓘ
- Freenove ESP32S3 Display 2.8 LCD
- Enable WeChat Message Style ⓘ
- Enable Wake Word Detection (AFE) ⓘ
- Enable Audio Noise Reduction ⓘ
- Enable Server-Side AEC (Unstable) ⓘ

Click “Load Multiple Wake Words” and check the boxes for ‘Hi, ESP’ and ‘Hi, Lily’ (and other desired options).

**Load Multiple Wake Words**

- Hi,乐鑫 (wn9\_hilexin) ⓘ
- 小爱同学 (wn9\_xiaoitongxue) ⓘ
- 你好小智 (wn9\_nihaoxiaozhi\_tts) ⓘ
- Alexa (wn9\_alexa) ⓘ
- Hi,ESP (wn9\_hiesp) ⓘ
- Jarvis (wn9\_jarvis\_tts) ⓘ
- computer (wn9\_computer\_tts) ⓘ
- Hey,Willow (wn9\_heywillow\_tts) ⓘ
- Sophia (wn9\_sophia\_tts) ⓘ
- 你好小鑫 (wn9\_nihaoxiaoxin\_tts) ⓘ
- 小美同学 (wn9\_xiaomeitongxue\_tts) ⓘ
- Hey,Printer (wn9\_heyprinter\_tts) ⓘ
- 小龙虾 (wn9\_xiaolongxiaolong\_tts) ⓘ
- 喵喵同学 (wn9\_miaomiaotongxue\_tts) ⓘ
- Hey,Wanda (wn9\_heywanda\_tts) ⓘ
- Hi,喵喵 (wn9\_himiaomiao\_tts) ⓘ
- Mycroft (wn9\_mycroft\_tts) ⓘ
- Hi,Joy (wn9\_hijoy\_tts) ⓘ
- Hi,Lily/Hi,莉莉 (wn9\_hilili\_tts) ⓘ

Finally, click “Save” to store your configuration. A success message will appear at the bottom upon completion.

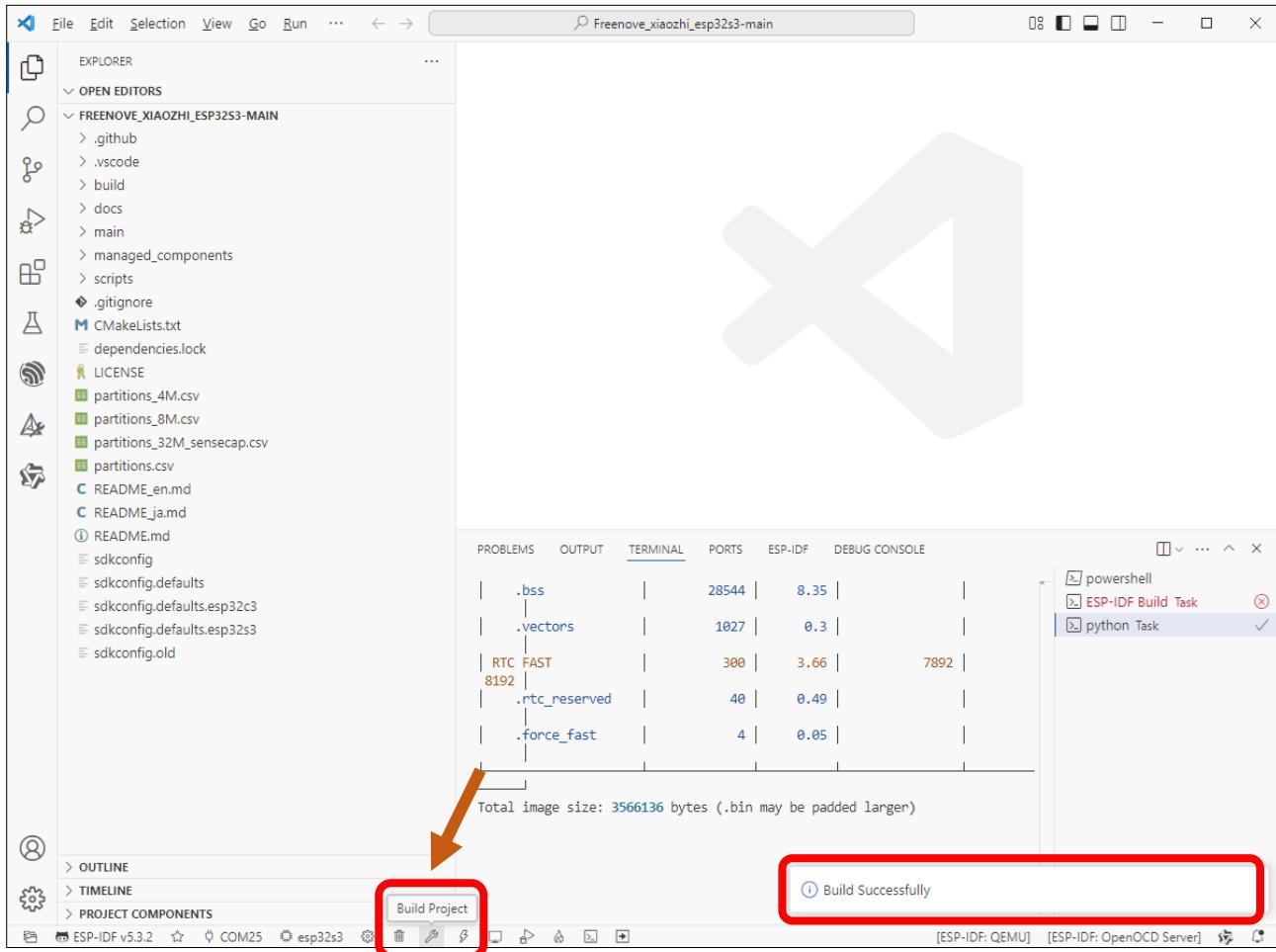


## Code Compilation

Before compiling, make sure all aforementioned configurations are correct. Click the 'Full Clean' button (bottom toolbar) to reset build cache.



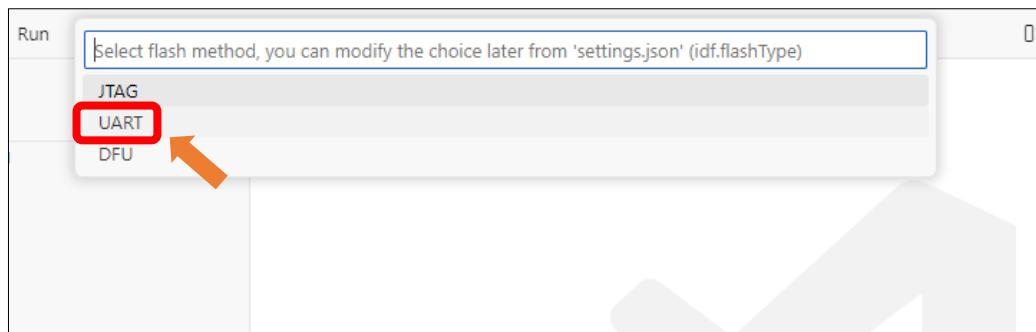
Click 'Build Project' at the bottom to start compiling the entire project. The first compilation may take longer - please wait patiently until the success message appears in the output panel.



Click 'Flash Device' at the bottom to start uploading the code to your ESP32-S3-WROOM module.



From the new options menu, select '**UART**' and wait for the code upload to complete.



Upon seeing the message 'Flash has finished. You can monitor your device with "ESP-IDF: Monitor command"', this indicates you have successfully uploaded XiaoZhi AI's firmware to the ESP32-S3-WROOM module.

```
PROBLEMS OUTPUT TERMINAL PORTS IDF DEBUG CONSOLE Filter IDF ... X
Project build complete. To flash, run:
ESP-IDF: Flash your project in the ESP-IDF Visual Studio Code Extension
or in a ESP-IDF Terminal:
idf.py flash
or
idf.py -p PORT flash
or
python -m esptool --chip esp32s3 -b 460800 --before default_reset --after hard_reset --port COM25 write_flash --flash_mode dio
--flash_size 16MB --flash_freq 80M 0x0 bootloader/bootloader.bin 0x10000 xiaozhi.bin 0x8000 partition_table/partition-table.
bin 0xd000 ota_data_initial.bin 0x10000 srmodels/srmodels.bin
or from the "c:\Users\DESKTOP-LIN\Downloads\Freenove_xiaozhi_esp32s3-main\build" directory
python -m esptool --chip esp32s3 -b 460800 --before default_reset --after hard_reset write_flash "@flash_args"
[/Build]
[Flash]
Flash Done ⚡
Flash has finished. You can monitor your device with 'ESP-IDF: Monitor command'
```

The terminal window shows the compilation log for the ESP-IDF project. It includes instructions for flashing via terminal or VS Code, and specific commands for writing to the flash using esptool. The final message 'Flash Done ⚡' is highlighted with a red box and an orange arrow pointing to it from the left.

At this point, the compilation is complete and you're ready for secondary development.



# Local Server

## Disclaimer

This project is derived from the open-source repository: <https://github.com/xinnan-tech/xiaozhi-esp32-server>, licensed under MIT License. The XiaoZhi AI firmware operates on servers provided by Xiage's company. We have only adapted it for third-party learning and AI functionality trials, without any commercial promotion or application. This tutorial is intended solely for enthusiasts to supplement their learning.

## Deploying XiaoZhi AI on a Local Server

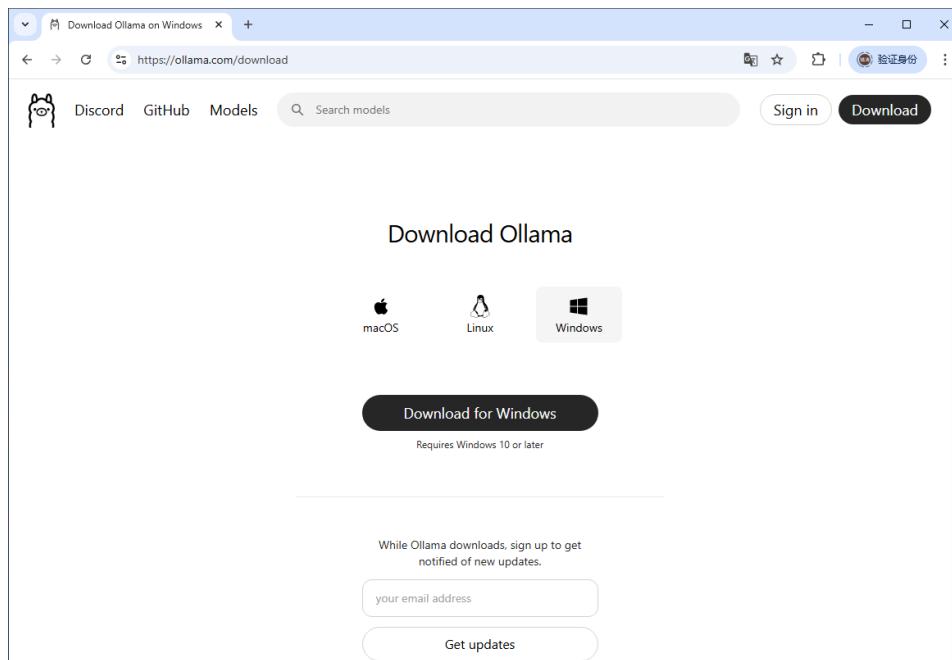
If you prefer not to use the XiaoZhi AI server, you can also set up a simplified version of the server on your own computer. In this section, we will use the open-source project at <https://github.com/xinnan-tech/xiaozhi-esp32-server> to deploy a local server and establish a connection with the ESP32 S3 WROOM. If you encounter any bugs in the code during use, please submit an issue at <https://github.com/xinnan-tech/xiaozhi-esp32-server>. Please note that we do not have an in-depth understanding of this project and may not be able to provide extensive assistance.

## Installing Ollama

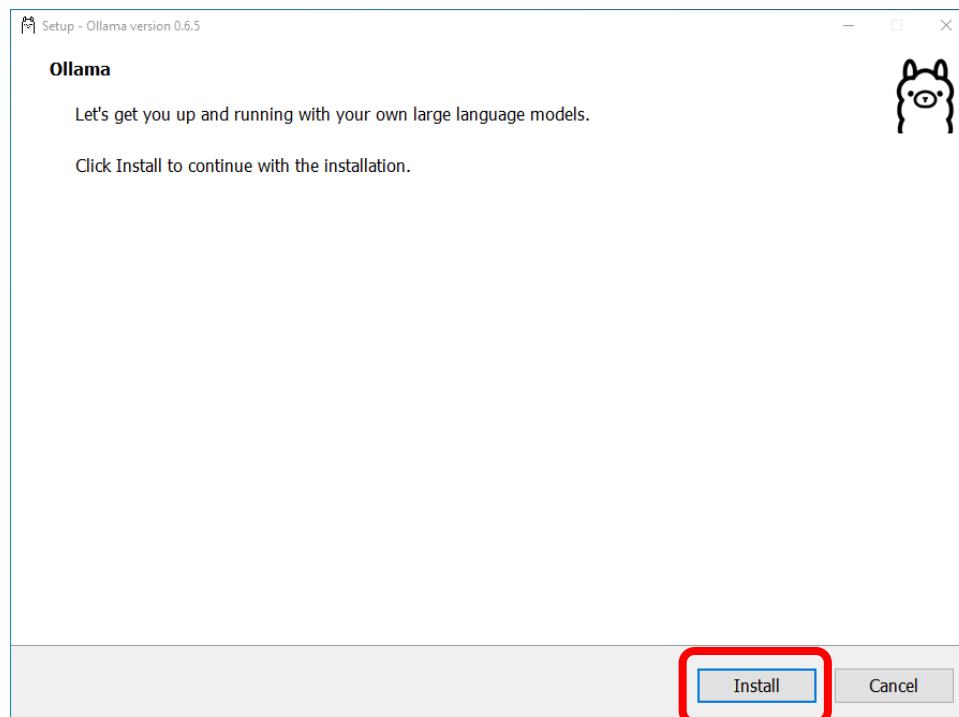
### Windows

Before getting started, we need to install the Ollama tool locally, which allows us to run any open-source AI model on our computer.

If you haven't installed Ollama yet, please visit <https://ollama.com/download> to download and install it.



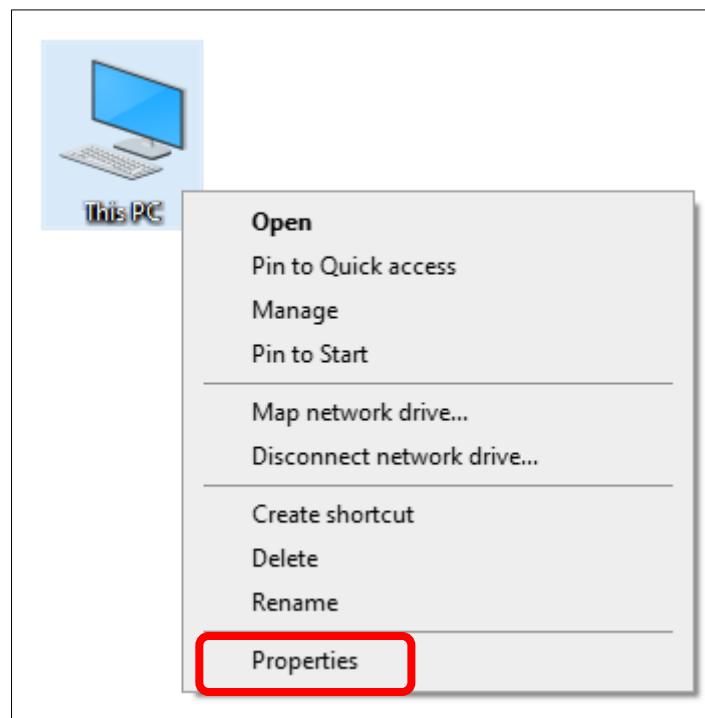
Run the Ollama Installing Package, click Install.



After installation, you'll see the Ollama icon appear in your taskbar.

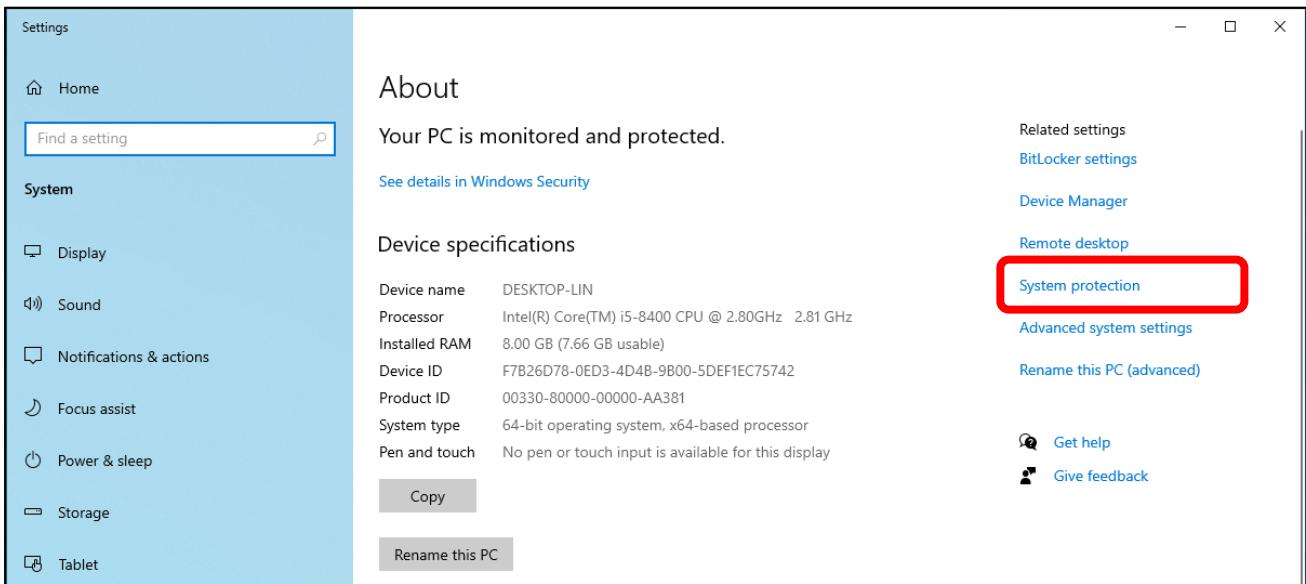


Return to your desktop, right-click on "This PC" (or "My Computer"), and select "Properties" from the context menu.

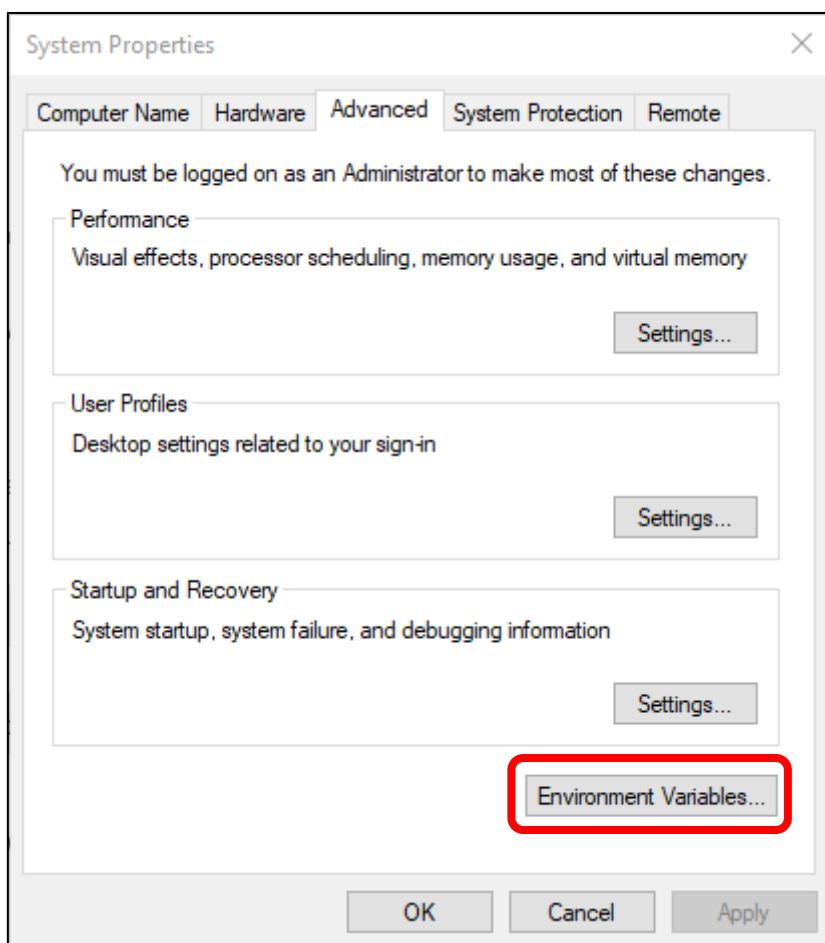




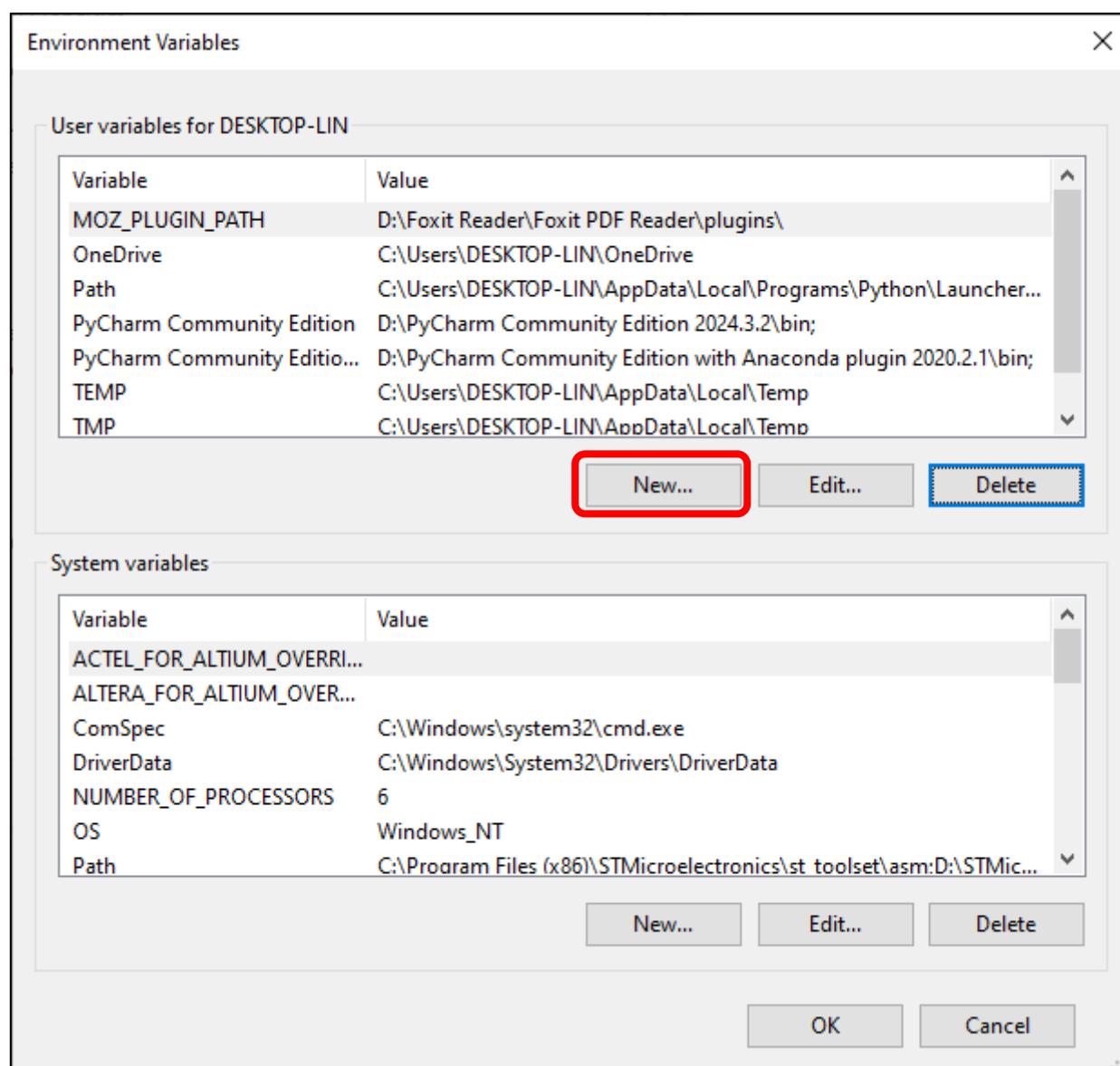
In the newly opened window, locate and click on "Advanced system settings".



In the newly opened window, click "Environment Variables".

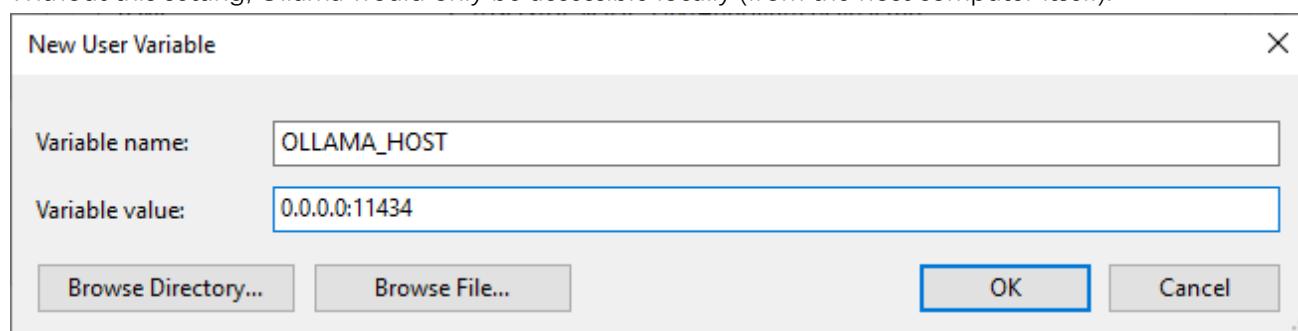


Click "New".



In the Variable name field, enter: OLLAMA\_HOST; In the Variable value field, enter: 0.0.0.0:11434; Click OK to save.

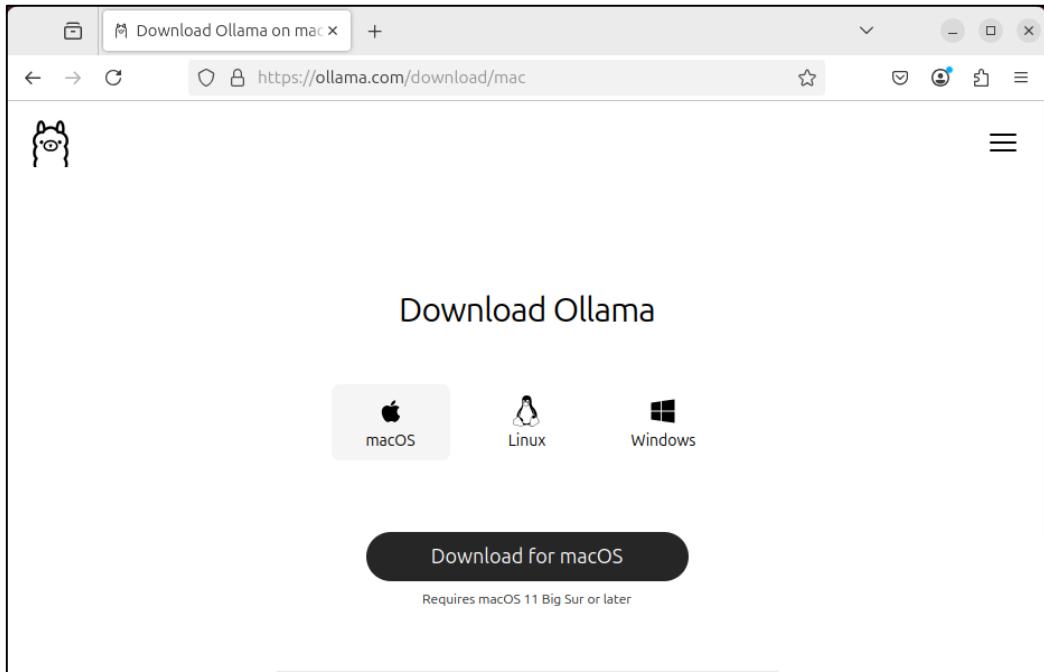
By doing so, all devices on your local network can access Ollama via your computer's IP address. Without this setting, Ollama would only be accessible locally (from the host computer itself).



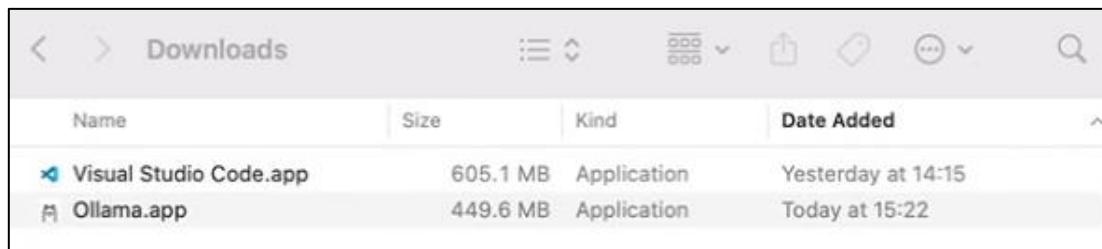
## Mac OS

Before getting started, we need to install the Ollama tool locally, which allows us to run any open-source AI model on our computer.

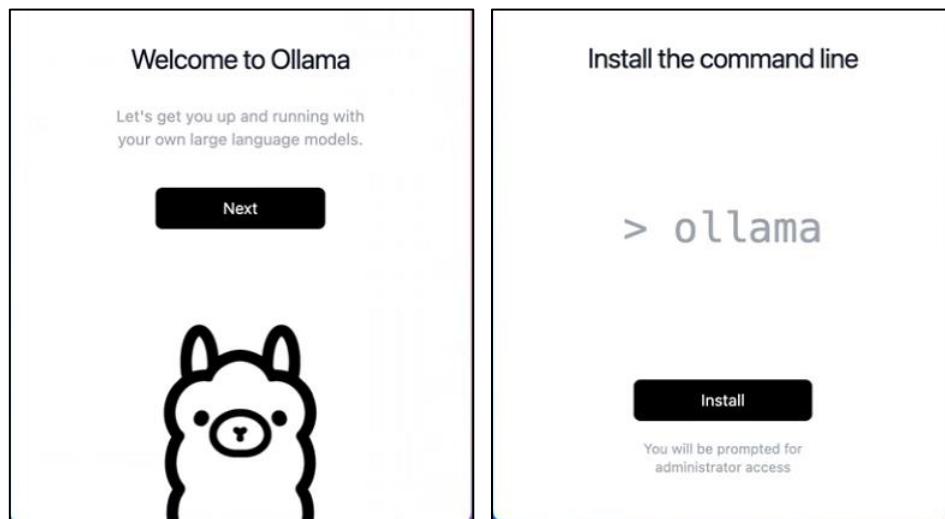
If you haven't installed Ollama yet, please visit <https://ollama.com/download> to download and install it.



Find "Ollama.app" under "Downloads" and double click to open it.



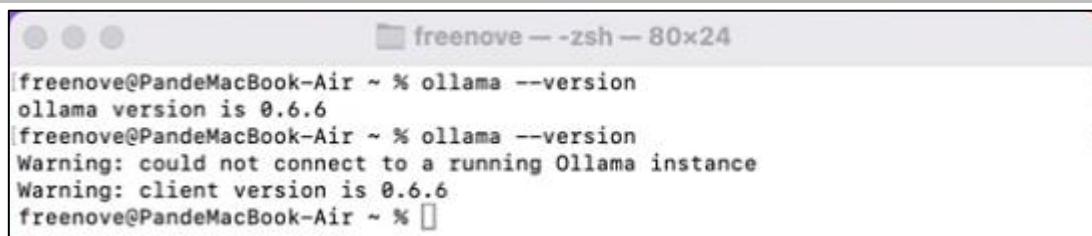
Click "Next" -> "Install" -> "Finish".



The interface will close automatically upon completion.

Open your terminal and check if Ollama is properly installed using the specified command.

**ollama --version**



```
freenove@PandeMacBook-Air ~ % ollama --version
ollama version is 0.6.6
[freenove@PandeMacBook-Air ~ % ollama --version
Warning: could not connect to a running Ollama instance
Warning: client version is 0.6.6
freenove@PandeMacBook-Air ~ % ]
```

Note: Ollama is located in your Applications folder.

As shown in the image above:

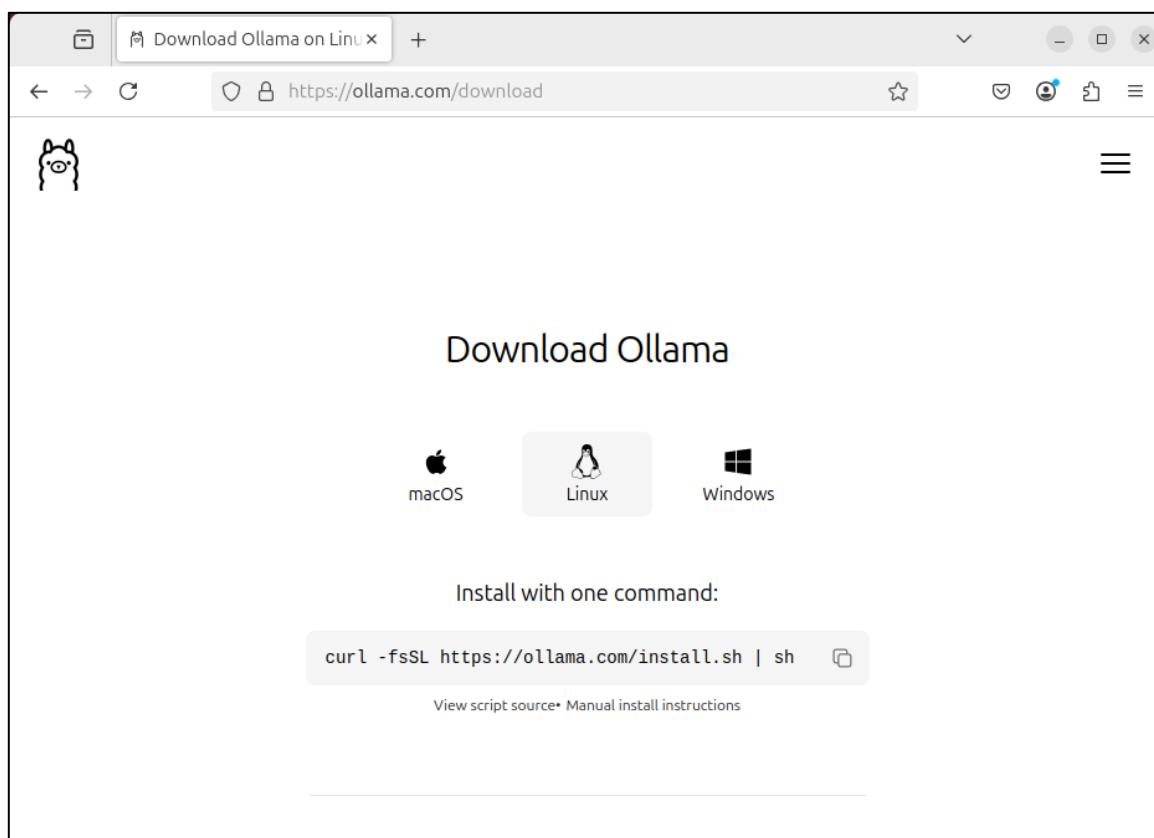
If Ollama is already running, executing ollama --version will display its version number.

If Ollama isn't running, the same command will return a connection error ("could not connect to a running Ollama instance")

## Linux

Before getting started, we need to install the Ollama tool locally, which allows us to run any open-source AI model on our computer.

If you haven't installed Ollama yet, please visit <https://ollama.com/download> to download and install it.





Open the Terminal and run the following command to install Ollama.

```
lin@ubuntu:~$ curl -fsSL https://ollama.com/install.sh | sh
```

The installation should appear as shown in the image above. You can verify Ollama's installation by running the command: **ollama --version**

```
>>> Creating ollama user...
>>> Adding ollama user to render group...
>>> Adding ollama user to video group...
>>> Adding current user to ollama group...
>>> Creating ollama systemd service...
>>> Enabling and starting ollama service...
Created symlink '/etc/systemd/system/default.target.wants/ollama.service' → '/etc/systemd/system/ollama.service'.
>>> The Ollama API is now available at 127.0.0.1:11434.
>>> Install complete. Run "ollama" from the command line.
WARNING: No NVIDIA/AMD GPU detected. Ollama will run in CPU-only mode.
lin@ubuntu:~$ ollama --version
ollama version is 0.6.6
lin@ubuntu:~$
```

## LLM Model

Please visit <https://ollama.com/search> and select the LLM model that suits your computer or your favorite.

The screenshot shows the Ollama Search interface with the URL <https://ollama.com/search> in the address bar. The page displays a list of LLM models:

- gemma3**: The current, most capable model that runs on a single GPU. It has 27b parameters. Last updated 4 hours ago.
- qwq**: QwQ is the reasoning model of the Qwen series. It has 32b parameters. Last updated 5 weeks ago.
- deepseek-r1**: DeepSeek's first-generation of reasoning models with comparable performance to OpenAI-o1, including six dense models distilled from DeepSeek-R1 based on Llama and Qwen. It has 671b parameters. Last updated 2 months ago.

Here we take qwen2.5 as an example. Click “qwen2.5” model.

The screenshot shows the Ollama interface with the URL <https://ollama.com/qwen2.5>. The page displays the following information for the qwen2.5 model:

- qwen2.5**: Qwen2.5 models are pretrained on Alibaba's latest large-scale dataset, encompassing up to 18 trillion tokens. The model supports up to 128K tokens and has multilingual support.
- Parameters: 7.62B
- Last updated: 7 months ago
- Tags: 133 Tags
- Run command: ollama run qwen2.5
- Model details table:
 

	Updated 7 months ago	845dbda0ea48 · 4.7GB
model	arch qwen2 · parameters 7.62B · quantization Q4_K_M	4.7GB
system	You are Qwen, created by Alibaba Cloud. You are a helpful assist...	68B
template	{{- if .Messages }} {{- if or .System .Tools }}< im_start >{{.}}< im_end >{{.}}	1.5kB
license	Apache License Version 2.0, January 200	11kB



Please note that when selecting a model, you need to choose the appropriate model based on your computer's GPU memory or CPU RAM configuration.

1. Larger models offer higher intelligence, while smaller models provide lower intelligence.
2. For high-end systems (strong GPU/CPU with ample memory), choose larger models for optimal performance; for low-end systems (limited GPU/CPU memory), opt for smaller models to ensure smooth operation.
3. Selecting an oversized model on a weak system may cause failure to load or extremely slow inference speeds.

You can select the appropriate model parameters via the dropdown menu.

**qwen2.5**

Qwen2.5 models are pretrained on Alibaba's latest large-scale dataset, encompassing up to 18 trillion tokens. The model supports up to 128K tokens and has multilingual support.

tools 0.5b 1.5b 3b 7b 14b 32b 72b

6.8M Pulls Updated 7 months ago

7b

0.5b 398M  
1.5b 986M  
3b 1.9G  
7b 4.7G  
14b 9.0G  
32b 20G  
72b 47G

View all Readme

133 Tags ollama run qwen2.5

parameters 7.62B · quantization Q4_K_M	845dbda0ea48 · 4.7GB
created by Alibaba Cloud. You are a helpful assist...	688
es }} {{- if or .System .Tools }}< im_start >syste...	1.5kB
Version 2.0, January 200	11kB

Smaller models are less capable but faster. For this demonstration, we'll use qwen2.5:0.5b as an example.

Copy the command from the webpage:

**ollama run qwen2.5:0.5b**

**qwen2.5**

Qwen2.5 models are pretrained on Alibaba's latest large-scale dataset, encompassing up to 18 trillion tokens. The model supports up to 128K tokens and has multilingual support.

tools 0.5b 1.5b 3b 7b 14b 32b 72b

6.9M Pulls Updated 7 months ago

0.5b

Updated 7 months ago a8b0c5157701 · 398MB

model arch qwen2 · parameters 494M · quantization Q4\_K\_M 398MB

system You are Qwen, created by Alibaba Cloud. You are a helpful assista... 688

template {{- if .Messages }} {{- if or .System .Tools }}<|im\_start|>system... 1.5kB

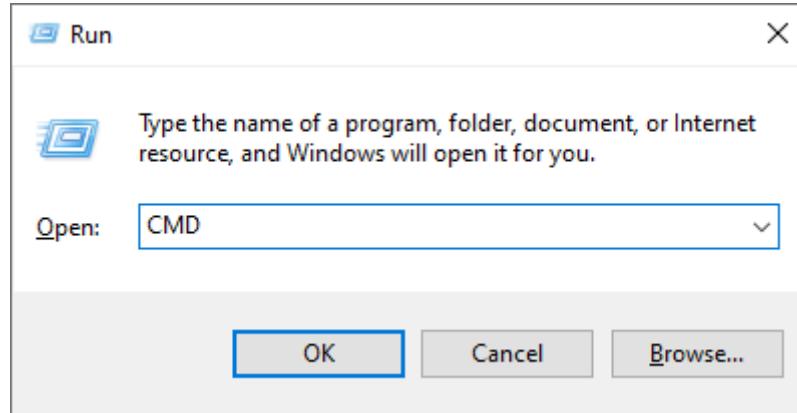
license Apache License Version 2.0, January 200 11kB

133 Tags ollama run qwen2.5:0.5b

Next, install your preferred LLM model by selecting the version compatible with your operating system.

## Windows

Use the shortcut "Win+R", enter "CMD" in the pop-up window, and open the CMD interface.



Run the command "**ollama --version**" to see if ollama has been installed.

A screenshot of a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe'. The window shows the following text:

```
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DESKTOP-LIN>ollama --version
ollama version is 0.6.5

C:\Users\DESKTOP-LIN>
```

Enter "**ollama run qwen2.5:0.5b**" to download the model to the local.machine.

A screenshot of a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe - ollama run qwen2.5:0.5b'. The window shows the following text:

```
C:\Users\DESKTOP-LIN>ollama run qwen2.5:0.5b
pulling manifest
pulling c5396e06af29... 100%
pulling 66b9ea09bd5b... 100%
pulling eb4402837c78... 100%
pulling 832dd9e00a68... 100%
pulling 005f95c74751... 100%
verifying sha256 digest
writing manifest
success
>>> Send a message (/? for help)
```

The command 'ollama run qwen2.5:0.5b' is highlighted with a red box.



After installation is complete, you can directly chat with Qwen2.5-0.5B in the CMD interface.

```
C:\Windows\system32\cmd.exe - ollama run qwen2.5:0.5b
C:\Users\DESKTOP-LIN>ollama run qwen2.5:0.5b
pulling manifest
pulling c5396e06af29... 100% 397 MB
pulling 66b9ea09bd5b... 100% 68 B
pulling eb4402837c78... 100% 1.5 KB
pulling 832dd9e00a68... 100% 11 KB
pulling 005f95c74751... 100% 490 B
verifying sha256 digest
writing manifest
success
>>> Hello
Hello! How can I help you today? Feel free to ask me any questions or share your concerns
with me.

>>>
Use Ctrl + d or /bye to exit.
>>> Send a message (/? for help)
```

You can press Ctrl+D to exit chat mode.

You can start the Ollama server by running the command “**ollama serve**”

```
C:\Windows\system32\cmd.exe - ollama serve
time=2025-04-21T10:49:53.136+08:00 level=INFO source=gpu.go:217 msg="looking
for compatible GPUs"
time=2025-04-21T10:49:53.136+08:00 level=INFO source=gpu_windows.go:167 msg=p
ackages count=1
time=2025-04-21T10:49:53.136+08:00 level=INFO source=gpu_windows.go:214 msg="
" package=0 cores=6 efficiency=0 threads=6
time=2025-04-21T10:49:53.149+08:00 level=INFO source=gpu.go:377 msg="no compa
tible GPUs were discovered"
time=2025-04-21T10:49:53.149+08:00 level=INFO source=types.go:130 msg="infere
nce compute" id=0 library=cpu variant="" compute="" driver=0.0 name="" total=
"7.7 GiB" available="4.0 GiB"
```

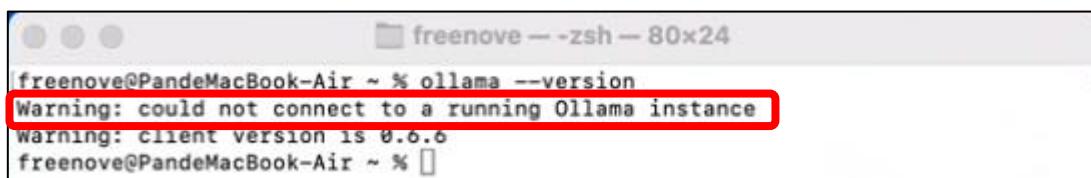
Note: If Ollama is already running (indicated by its icon in the system taskbar), executing ollama serve will cause an error. Both methods launch the same service.



```
C:\Windows\system32\cmd.exe
C:\Users\DESKTOP-LIN>ollama serve
Error: listen tcp 0.0.0.0:11434: bind: Only one usage of each socket address
(protocol/network address/port) is normally permitted.
```

## Mac OS

Run the command “**ollama --version**” on the Terminal to check whether Ollama has been installed.

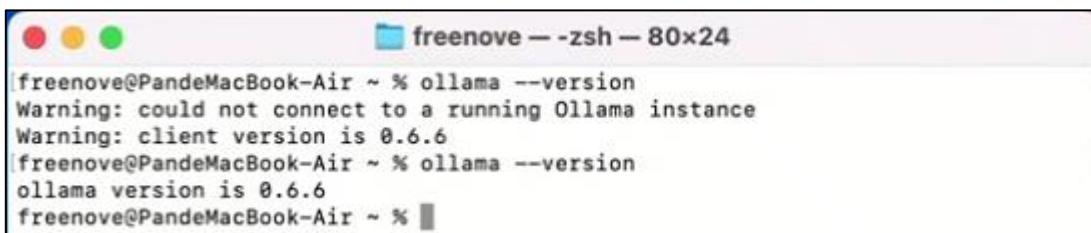


```
freenove@PandeMacBook-Air ~ % ollama --version
Warning: could not connect to a running Ollama instance
Warning: client version is 0.6.6
freenove@PandeMacBook-Air ~ %
```

If you see the prompt “Warning: could not connect to a running Ollama instance”, it indicates that Ollama has not been run. Go to Applications to run it.

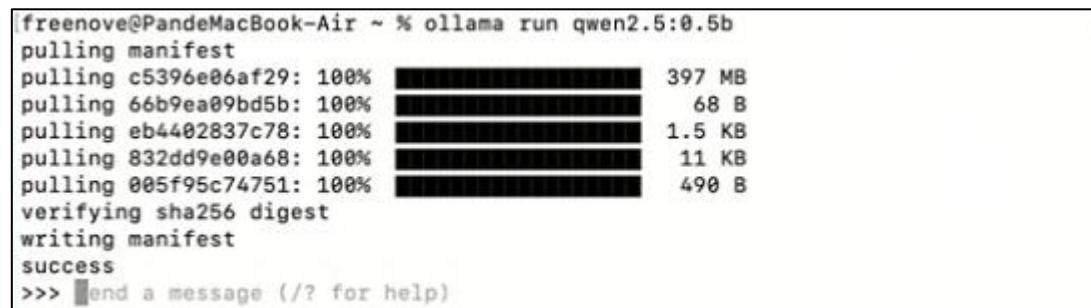


Check whether it is running again on the Terminal.



```
freenove@PandeMacBook-Air ~ % ollama --version
Warning: could not connect to a running Ollama instance
Warning: client version is 0.6.6
freenove@PandeMacBook-Air ~ % ollama --version
ollama version is 0.6.6
freenove@PandeMacBook-Air ~ %
```

On the Terminal, run “**ollama run qwen2.5:0.5b**” to install the model to your computer.



```
freenove@PandeMacBook-Air ~ % ollama run qwen2.5:0.5b
pulling manifest
pulling c5396e06af29: 100% [██████████] 397 MB
pulling 66b9ea09bd5b: 100% [██████████] 68 B
pulling eb4402837c78: 100% [██████████] 1.5 KB
pulling 832dd9e00a68: 100% [██████████] 11 KB
pulling 005f95c74751: 100% [██████████] 490 B
verifying sha256 digest
writing manifest
success
>>> End a message (/? for help)
```

After installation is complete, you can directly chat with Qwen2.5-0.5B in the Terminal interface.

```
freenove@PandeMacBook-Air ~ % ollama run qwen2.5:0.5b
pulling manifest
pulling c5396e06af29: 100% [██████████] 397 MB
pulling 66b9ea09bd5b: 100% [██████████] 68 B
pulling eb4402837c78: 100% [██████████] 1.5 KB
pulling 832dd9e00a68: 100% [██████████] 11 KB
pulling 005f95c74751: 100% [██████████] 490 B
verifying sha256 digest
writing manifest
success
[>>> Hello
Hello! How can I assist you today? Let me know if there's anything
specific you'd like to discuss or any questions that need help with. I'm
here to provide information and answer queries in a way that's easy for
you to understand.

[>>>
Use Ctrl + d or /bye to exit.
>>> █end a message (/? for help)
```

You may exit by pressing "Ctrl+D".

You can start the Ollama Server by running the command "**ollama serve**".

```
freenove@PandeMacBook-Air ~ % ollama serve
2025/04/24 17:10:39 routes.go:1232: INFO server config env="map[HTTPS_PROXY: HTTP_PROXY: NO_PROXY: OLLAMA_CONTEXT_LENGTH:2048 OLLAMA_DEBUG:false OLLAMA_FLASH_ATENTION:false OLLAMA_GPU_OVERHEAD:0 OLLAMA_HOST:http://127.0.0.1:11434 OLLAMA_KEEP_ALIVE:5m0s OLLAMA_KV_CACHE_TYPE: OLLAMA_LLM_LIBRARY: OLLAMA_LOAD_TIMEOUT:5m0s OLLAMA_MAX_LOADED_MODELS:0 OLLAMA_MAX_QUEUE:512 OLLAMA_MODELS:/Users/freenove/.ollama/models OLLAMA_MULTIUSER_CACHE:false OLLAMA_NEW_ENGINE:false OLLAMA_NOHOST:ORY:false OLLAMA_NOPRUNE:false OLLAMA_NUM_PARALLEL:0 OLLAMA_ORIGINS:[http://localhost https://localhost http://localhost/* https://localhost/* http://127.0.0.1 https://127.0.0.1 http://127.0.0.1/* https://127.0.0.1/* http://0.0.0.0 https://0.0.0.0 http://0.0.0.0/* https://0.0.0.0/* app:///* file:///* tauri:///* vscode-webview:///* vscode-file:///*] OLLAMA_SCHED_SPREAD:false http_proxy: https_proxy: no_proxy:]"
time=2025-04-24T17:10:39.898+08:00 level=INFO source=images.go:458 msg="total blobs: 5"
time=2025-04-24T17:10:39.899+08:00 level=INFO source=images.go:465 msg="total unused blobs removed: 0"
time=2025-04-24T17:10:39.899+08:00 level=INFO source=routes.go:1299 msg="Listening on 127.0.0.1:11434 (version 0.6.6)"
time=2025-04-24T17:10:39.900+08:00 level=INFO source=types.go:130 msg="inference compute" id="" library=cpu variant="" compute="" driver=0.0 name="" total="8.0 GiB" available="3.2 GiB"
```

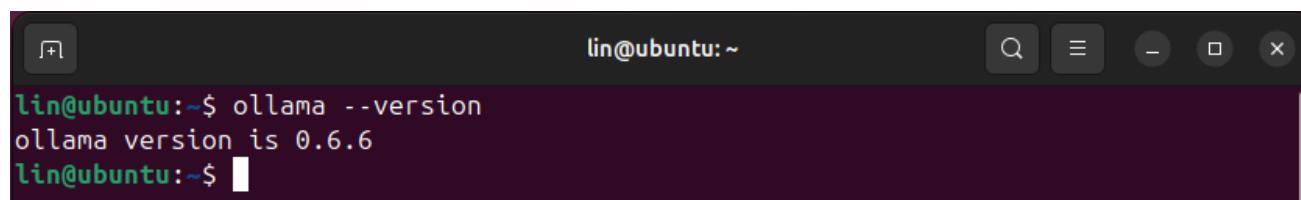
If Ollama has been running, you will see the following message.

```
freenove@PandeMacBook-Air ~ % ollama serve
Error: listen tcp 127.0.0.1:11434: bind: address already in use
freenove@PandeMacBook-Air ~ %
```

To access Ollama's user guide, run command **Ollama**.

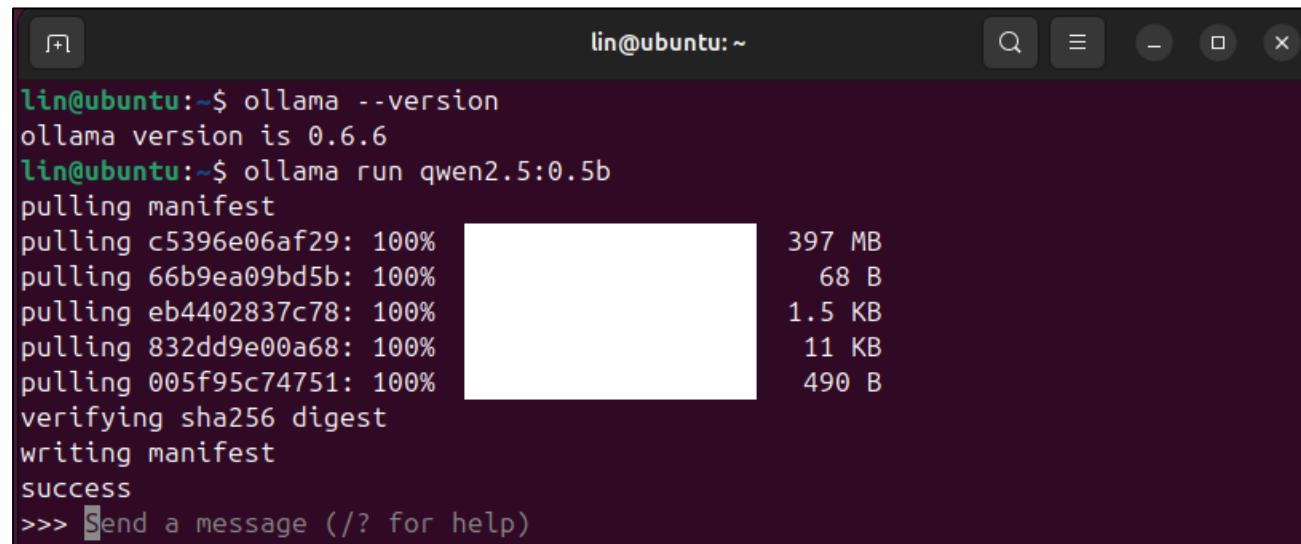
## Linux

Run the command “**ollama --version**” on the Terminal to check whether Ollama has been installed.



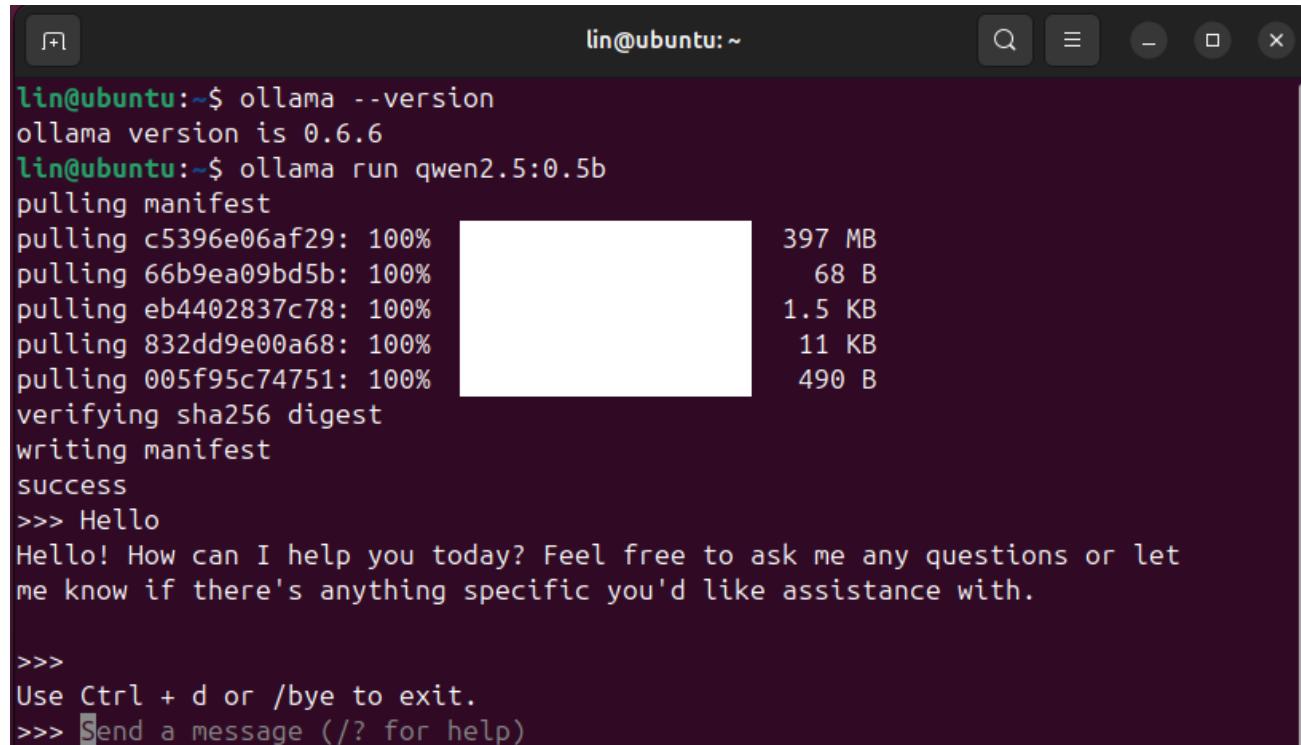
```
lin@ubuntu:~$ ollama --version
ollama version is 0.6.6
lin@ubuntu:~$
```

Run the command “**ollama run qwen2.5:0.5b**” to download the model to your computer.



```
lin@ubuntu:~$ ollama --version
ollama version is 0.6.6
lin@ubuntu:~$ ollama run qwen2.5:0.5b
pulling manifest
pulling c5396e06af29: 100%   397 MB
pulling 66b9ea09bd5b: 100%   68 B
pulling eb4402837c78: 100%   1.5 KB
pulling 832dd9e00a68: 100%   11 KB
pulling 005f95c74751: 100%   490 B
verifying sha256 digest
writing manifest
success
>>> Send a message (/? for help)
```

After installation is complete, you can directly chat with Qwen2.5-0.5B in the Terminal interface



```
lin@ubuntu:~$ ollama --version
ollama version is 0.6.6
lin@ubuntu:~$ ollama run qwen2.5:0.5b
pulling manifest
pulling c5396e06af29: 100%   397 MB
pulling 66b9ea09bd5b: 100%   68 B
pulling eb4402837c78: 100%   1.5 KB
pulling 832dd9e00a68: 100%   11 KB
pulling 005f95c74751: 100%   490 B
verifying sha256 digest
writing manifest
success
>>> Hello
Hello! How can I help you today? Feel free to ask me any questions or let
me know if there's anything specific you'd like assistance with.

>>>
Use Ctrl + d or /bye to exit.
>>> Send a message (/? for help)
```

To exit it, press “Ctrl+D”.



To access Ollama's user guide, run command Ollama.

```
lin@ubuntu:~ Usage:  
  ollama [flags]  
  ollama [command]  
  
Available Commands:  
  serve      Start ollama  
  create     Create a model from a Modelfile  
  show       Show information for a model  
  run        Run a model  
  stop       Stop a running model  
  pull       Pull a model from a registry  
  push       Push a model to a registry  
  list       List models  
  ps         List running models  
  cp         Copy a model  
  rm         Remove a model  
  help       Help about any command  
  
Flags:  
  -h, --help    help for ollama  
  -v, --version Show version information  
  
Use "ollama [command] --help" for more information about a command.  
lin@ubuntu:~$ █
```

## Installing Conda

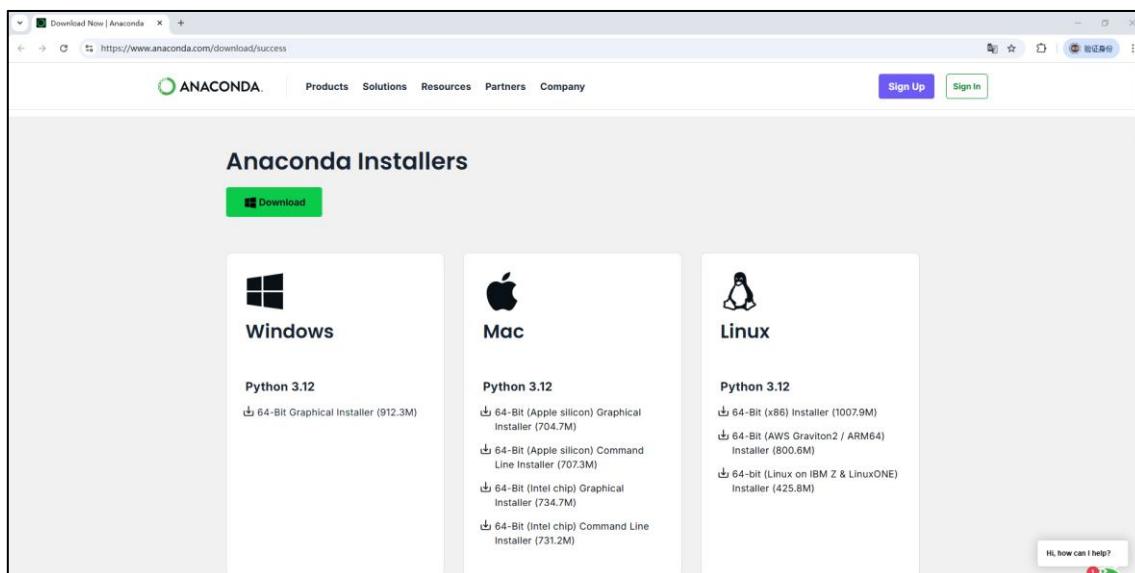
The xiaozhi-esp32-server open-source project offers four installation methods. In this tutorial, we'll demonstrate the simplest configuration example. For other usage methods, please refer to the project's website for further exploration.

### Windows

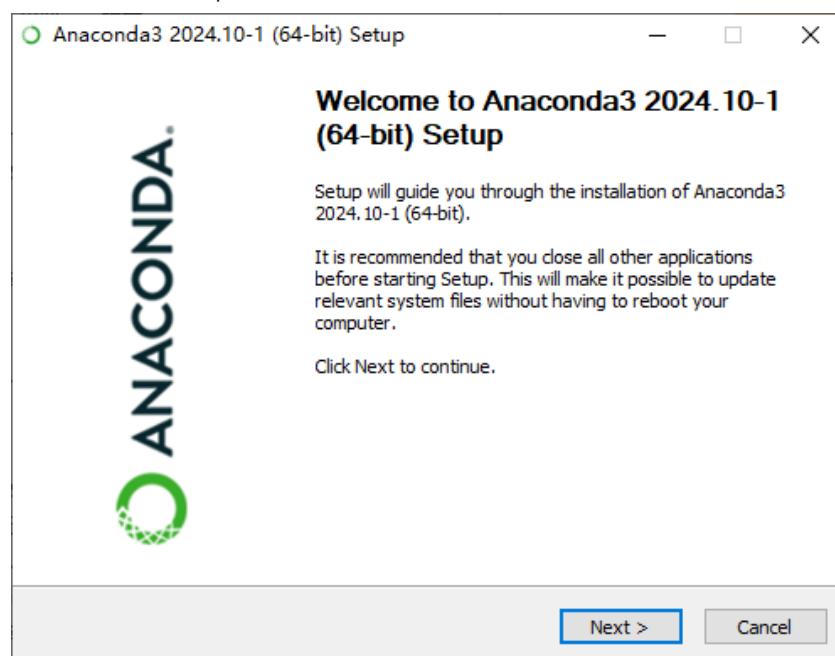
This example uses Conda for dependency management. Therefore, you'll need to have Conda installed on your system beforehand. If you haven't installed Conda yet, you can download and install it from: <https://www.anaconda.com/download/success>

Select the appropriate installer for your operating system.

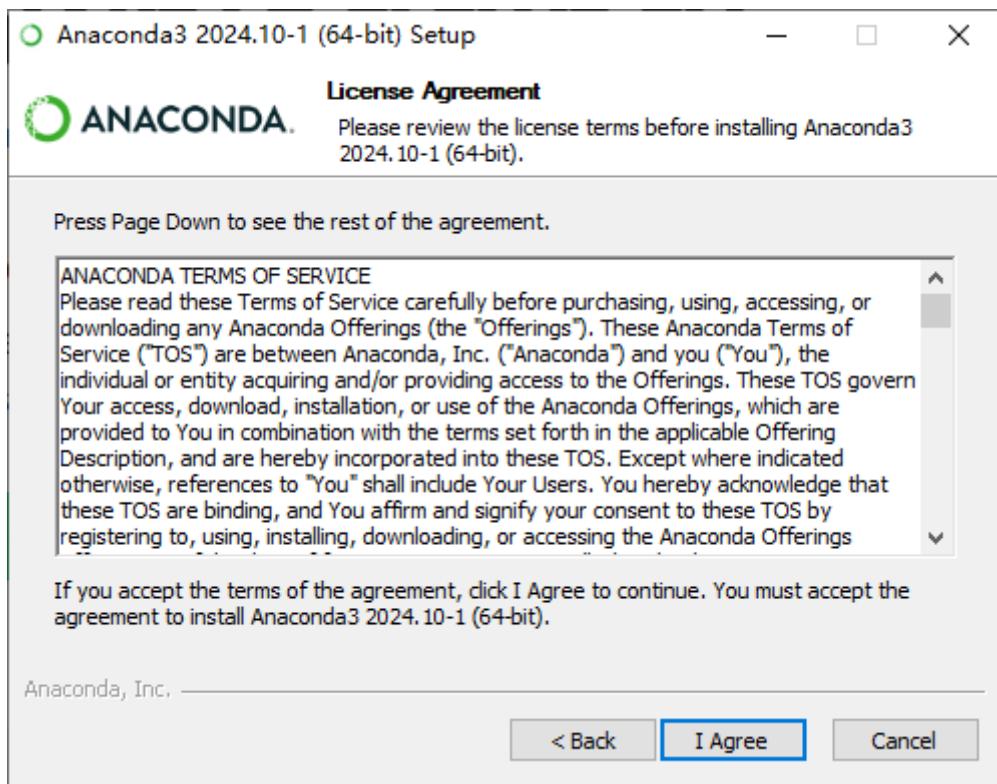
Miniconda is an installer by Anaconda that comes preconfigured for use with the Anaconda Repository.



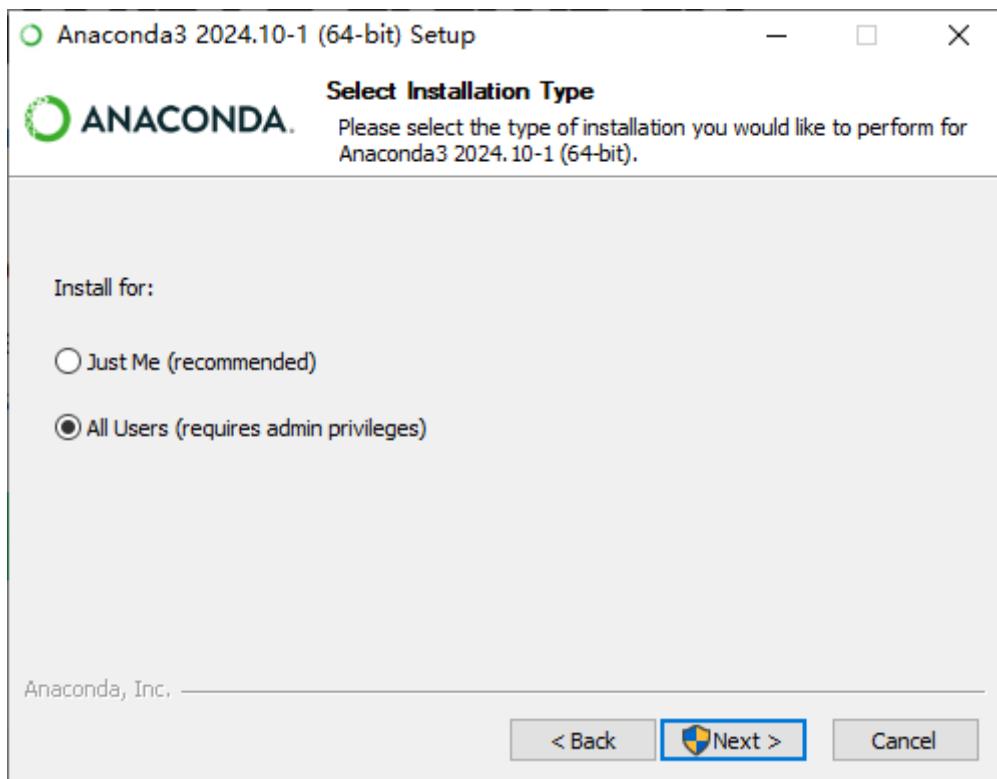
Here, we'll use Windows as an example. Double-click the downloaded Conda installer and click Next.



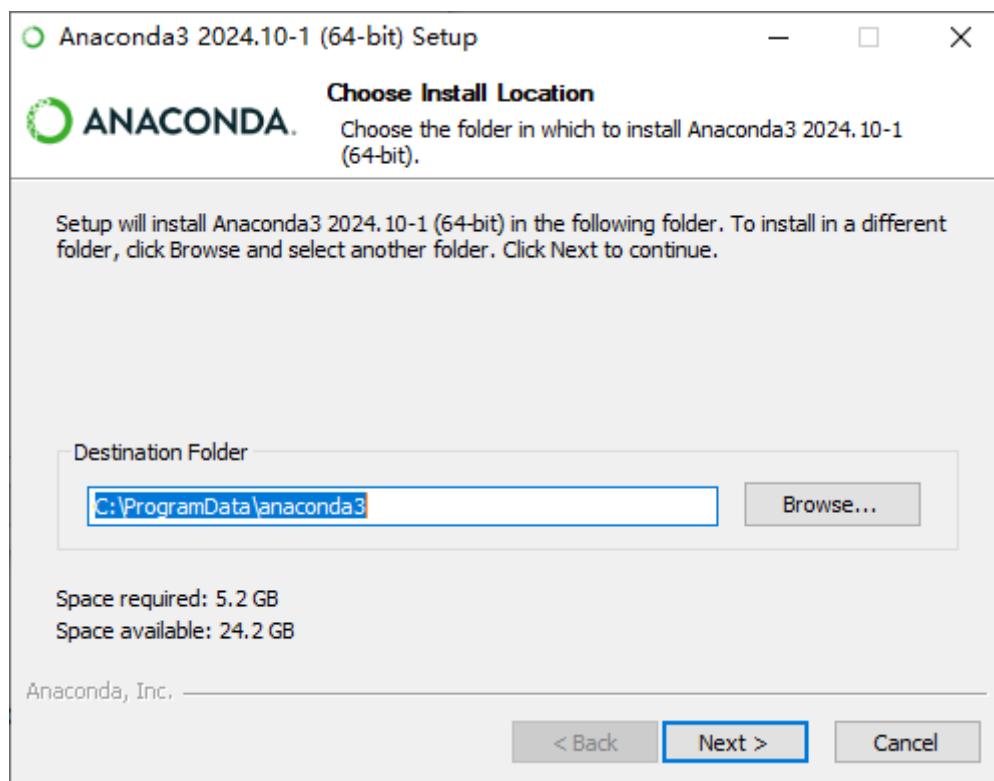
Click "I Agree".



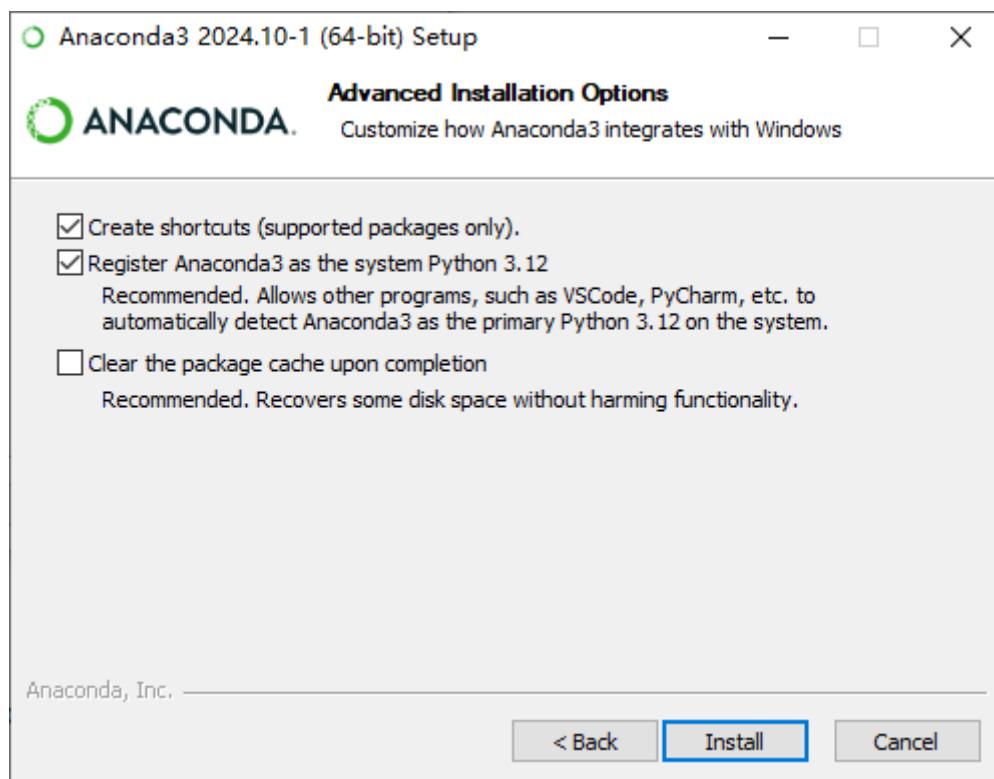
Select the installation type based on your needs. Typically, choose "All Users" for system-wide installation.



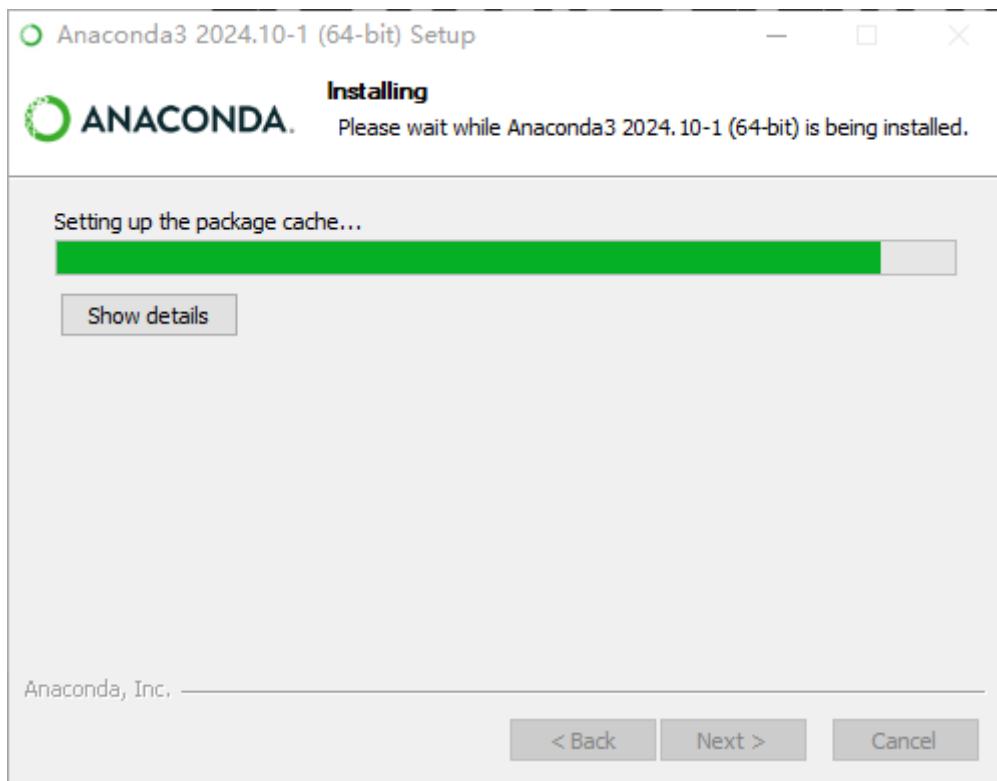
Specify the installation location for the software,



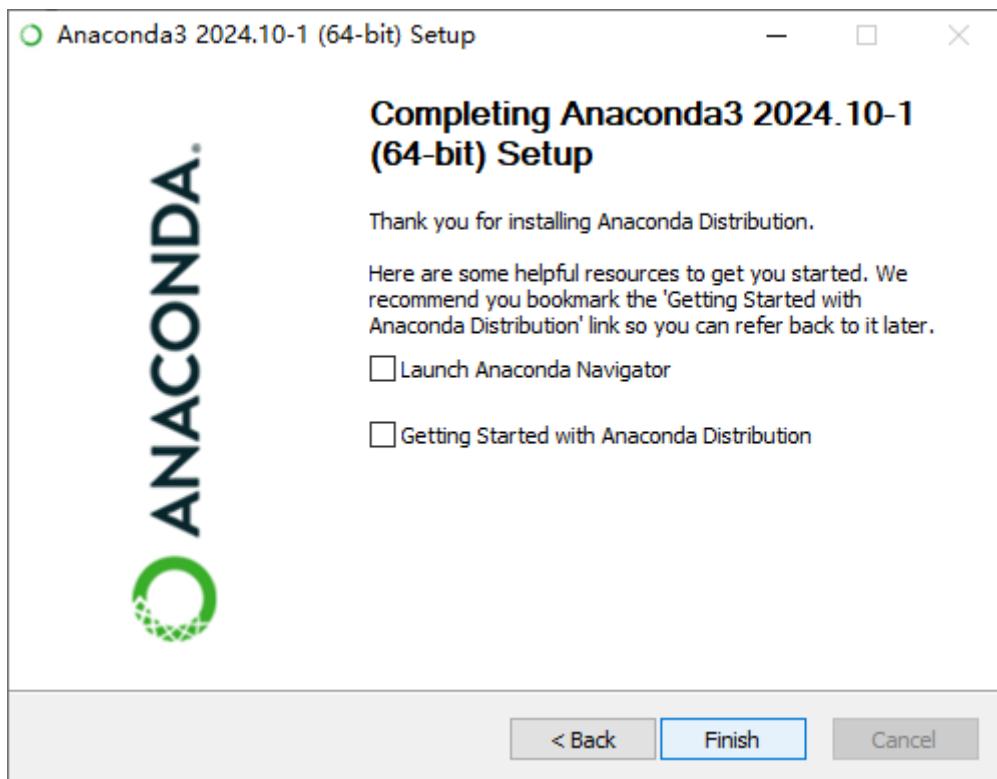
Keep the following configuration as default and click Install.



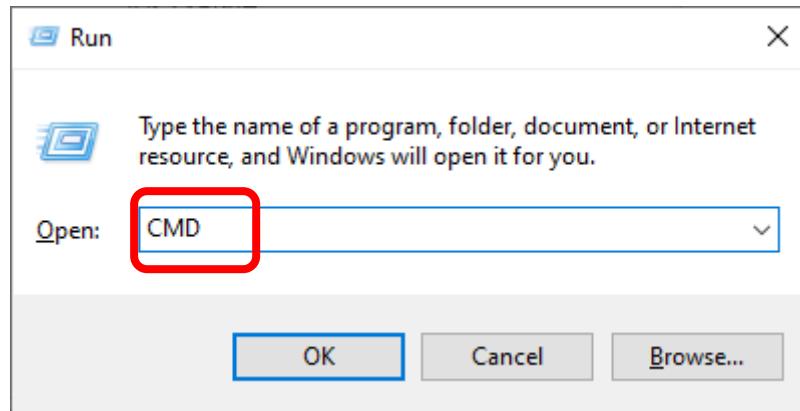
Wait for the installation to finish, which may take a while.



So far, the software has been installed.



Use the shortcut "Win+R", enter "CMD" in the pop-up window, and open the CMD interface.



Enter **conda --version** and press Enter. If Anaconda3 is installed correctly, you should see version information like this:

A screenshot of a Windows command prompt window titled 'C:\Windows\system32\CMD.exe'. The window shows the standard Windows copyright information: 'Microsoft Windows [Version 10.0.19045.5737] (c) Microsoft Corporation. All rights reserved.' Below this, the user has typed 'conda --version' and pressed Enter. The output 'conda 24.9.2' is displayed in white text on a black background. This output is highlighted with a red rectangle.

```
C:\Windows\system32\CMD.exe
Microsoft Windows [Version 10.0.19045.5737]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DESKTOP-LIN>conda --version
conda 24.9.2

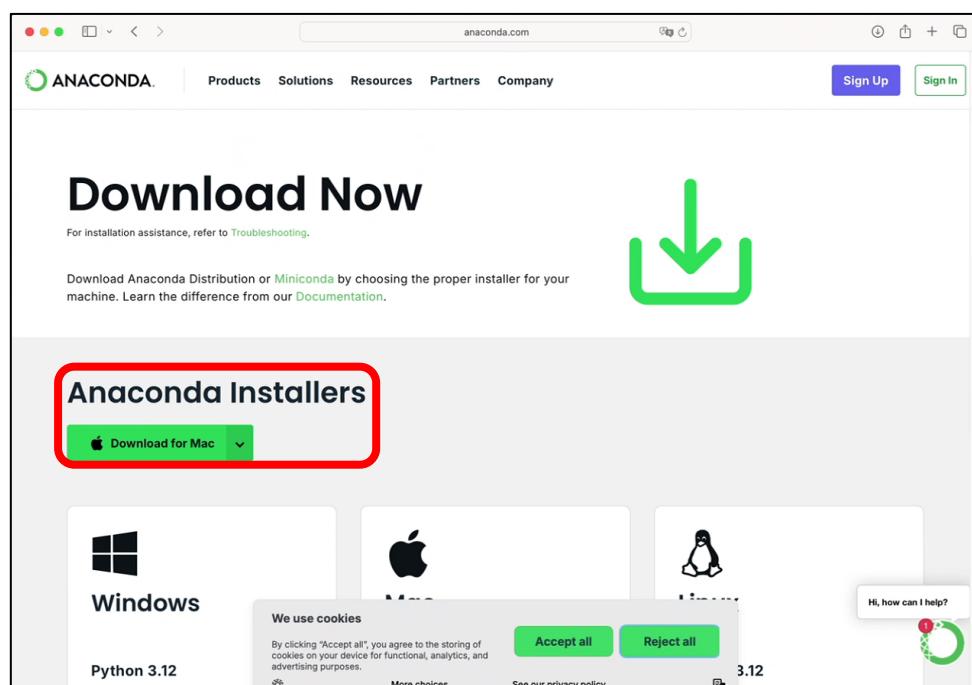
C:\Users\DESKTOP-LIN>
```

## Mac OS

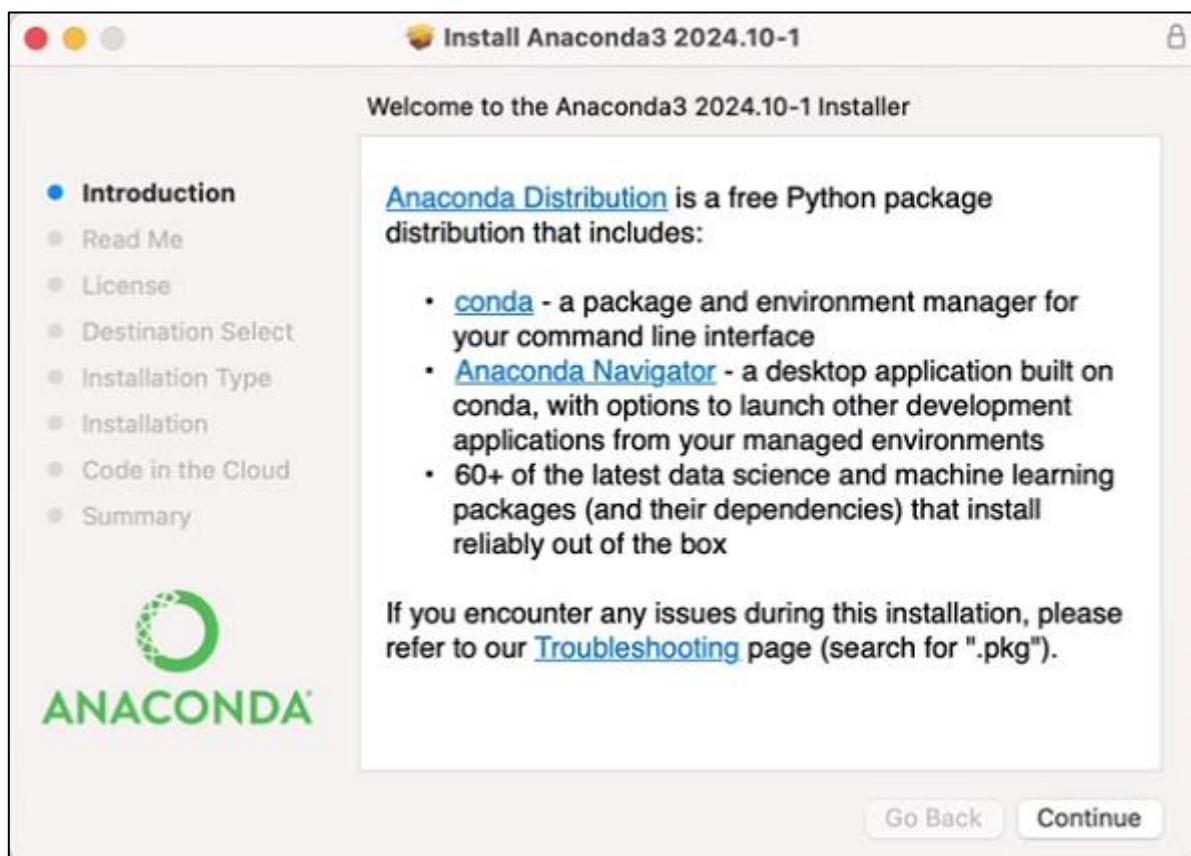
This example uses Conda for dependency management. Therefore, you'll need to have Conda installed on your system beforehand. If you haven't installed Conda yet, you can download and install it from: <https://www.anaconda.com/download/success>

Select the appropriate installer for your operating system.

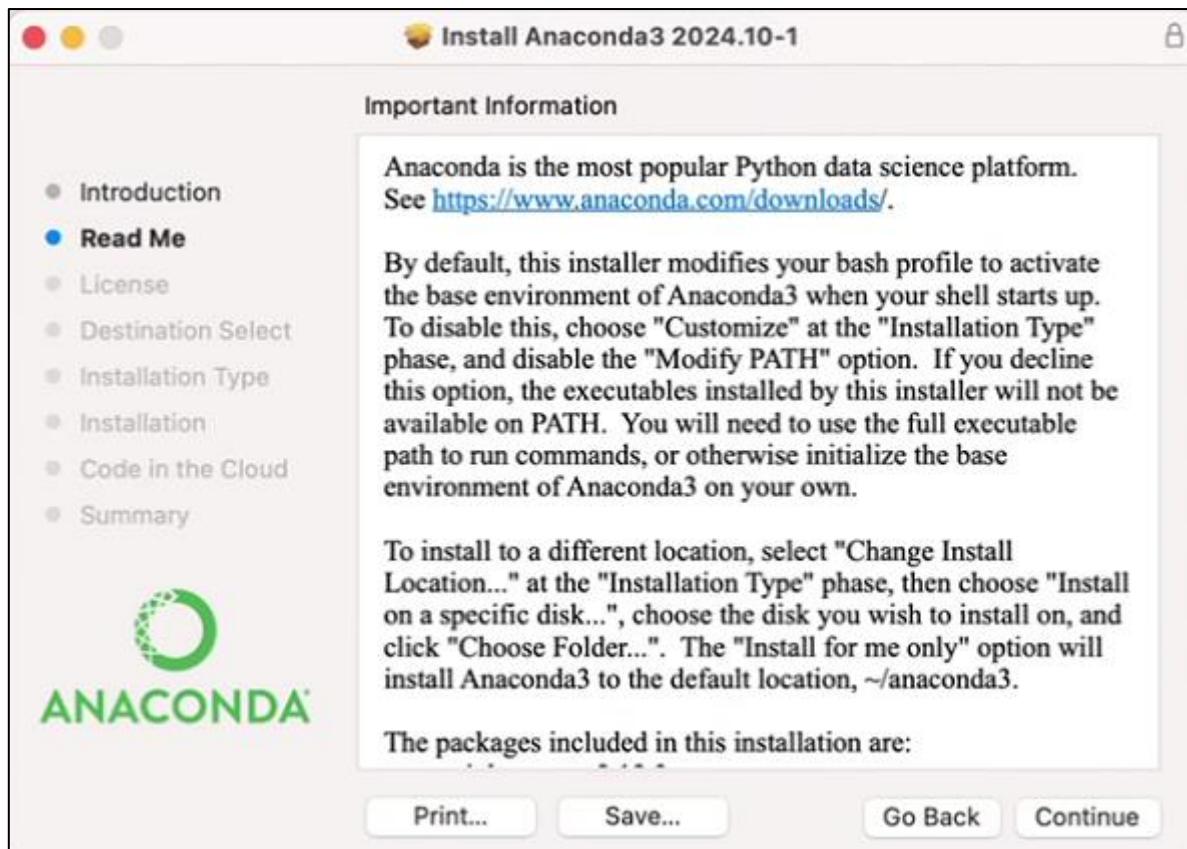
Miniconda is an installer by Anaconda that comes preconfigured for use with the Anaconda Repository.



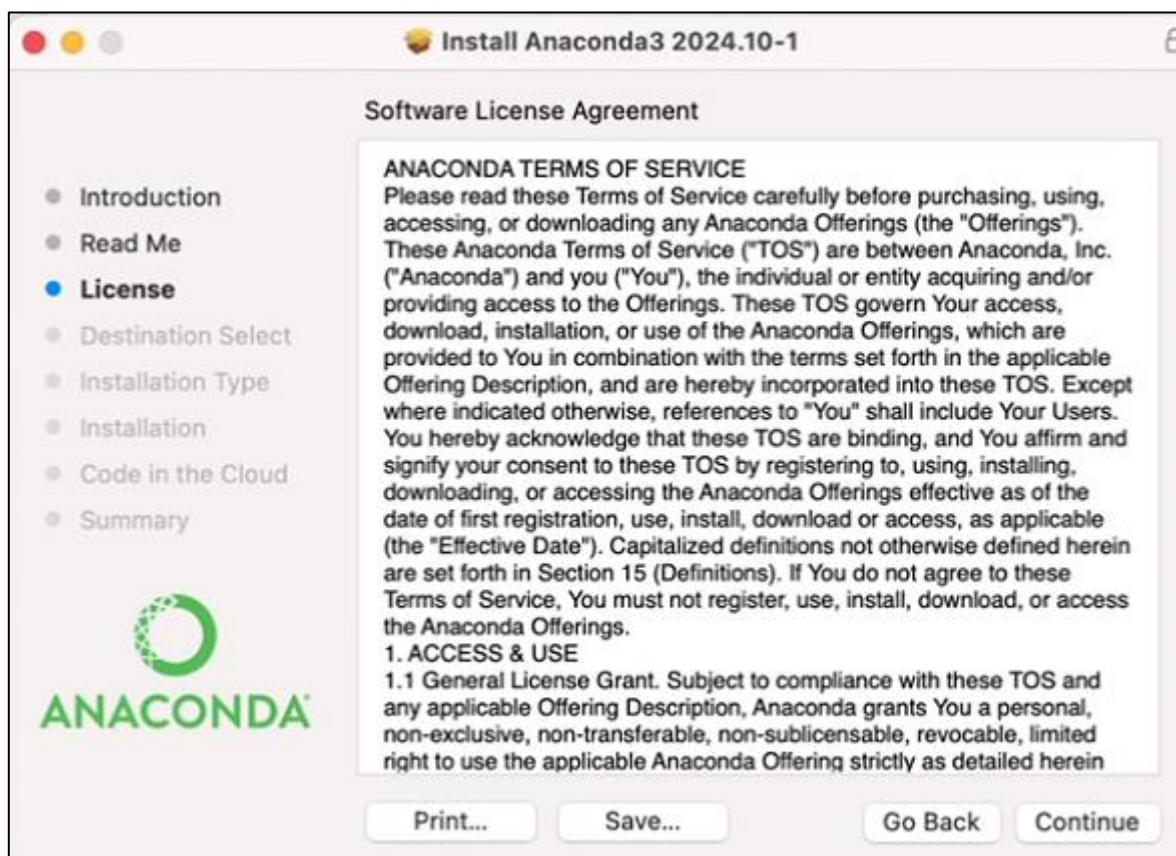
Double click to open the Conda application and click Continue.



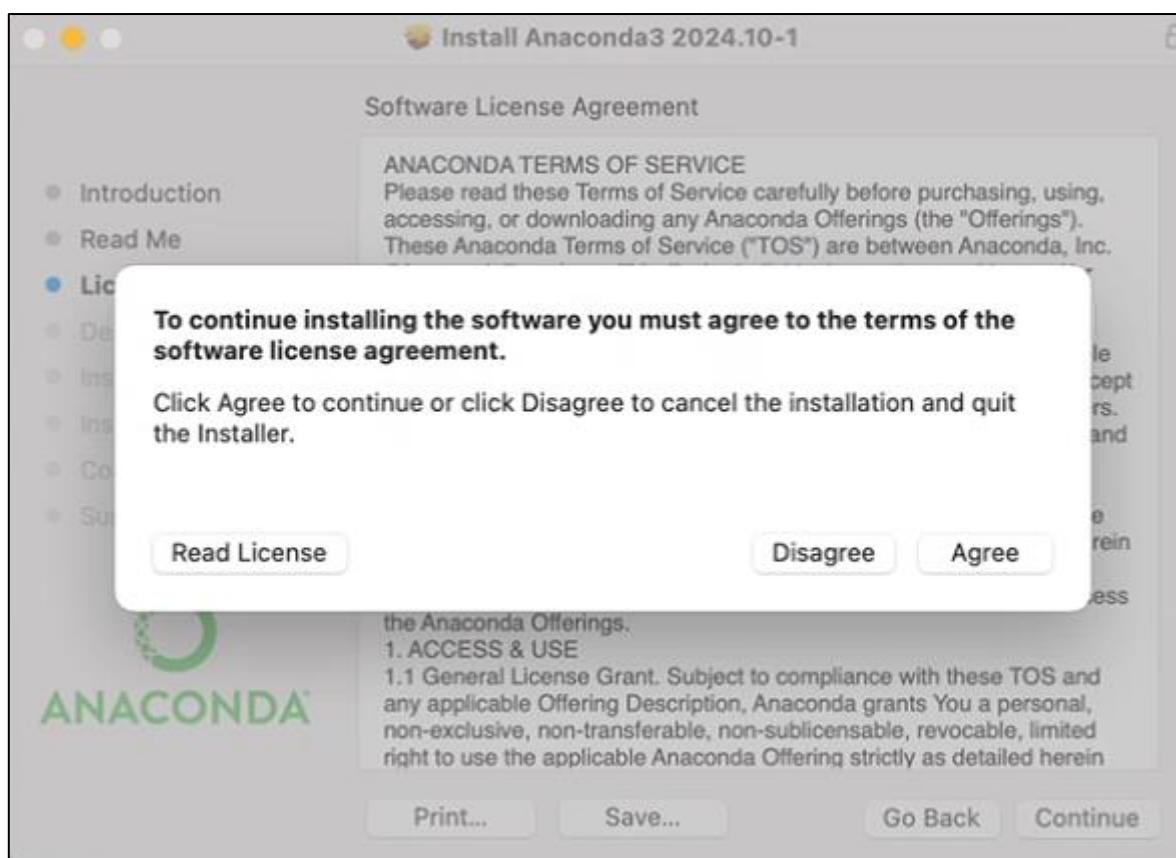
Click Continue.



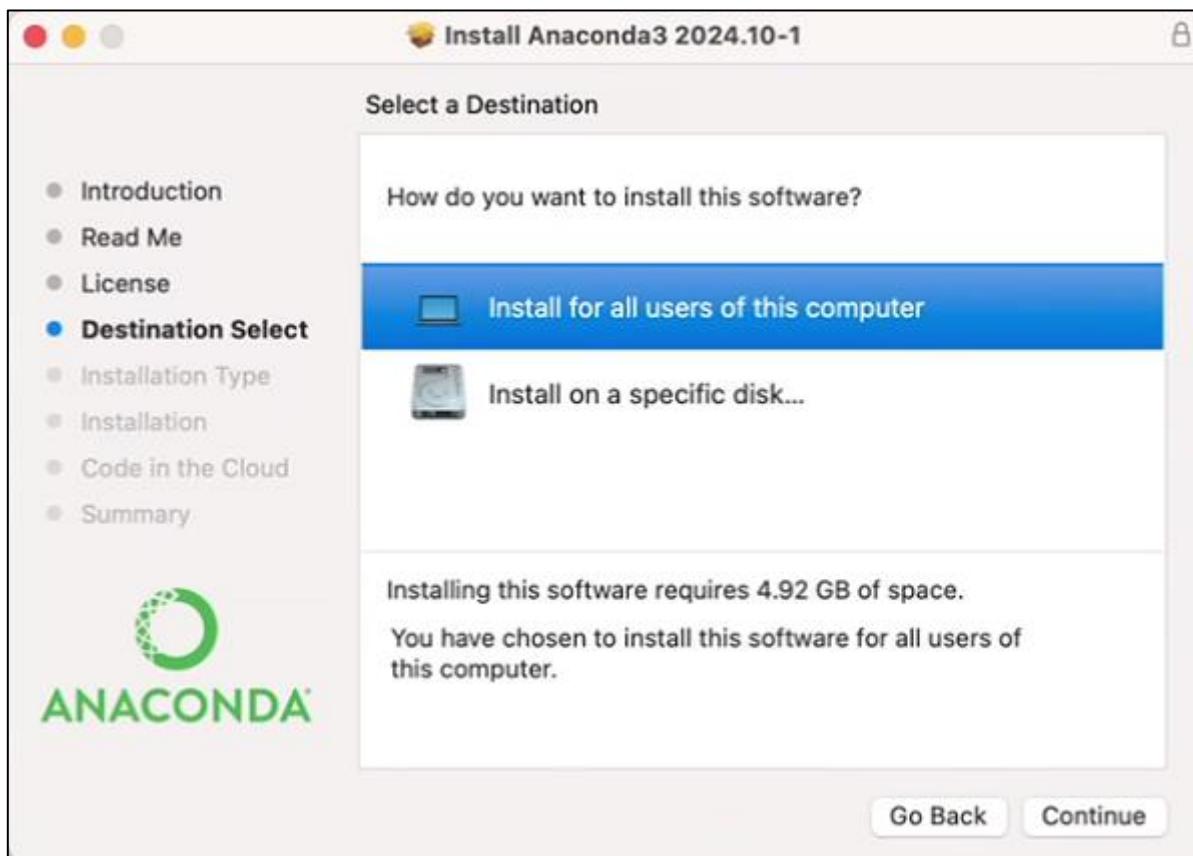
Click Continue.



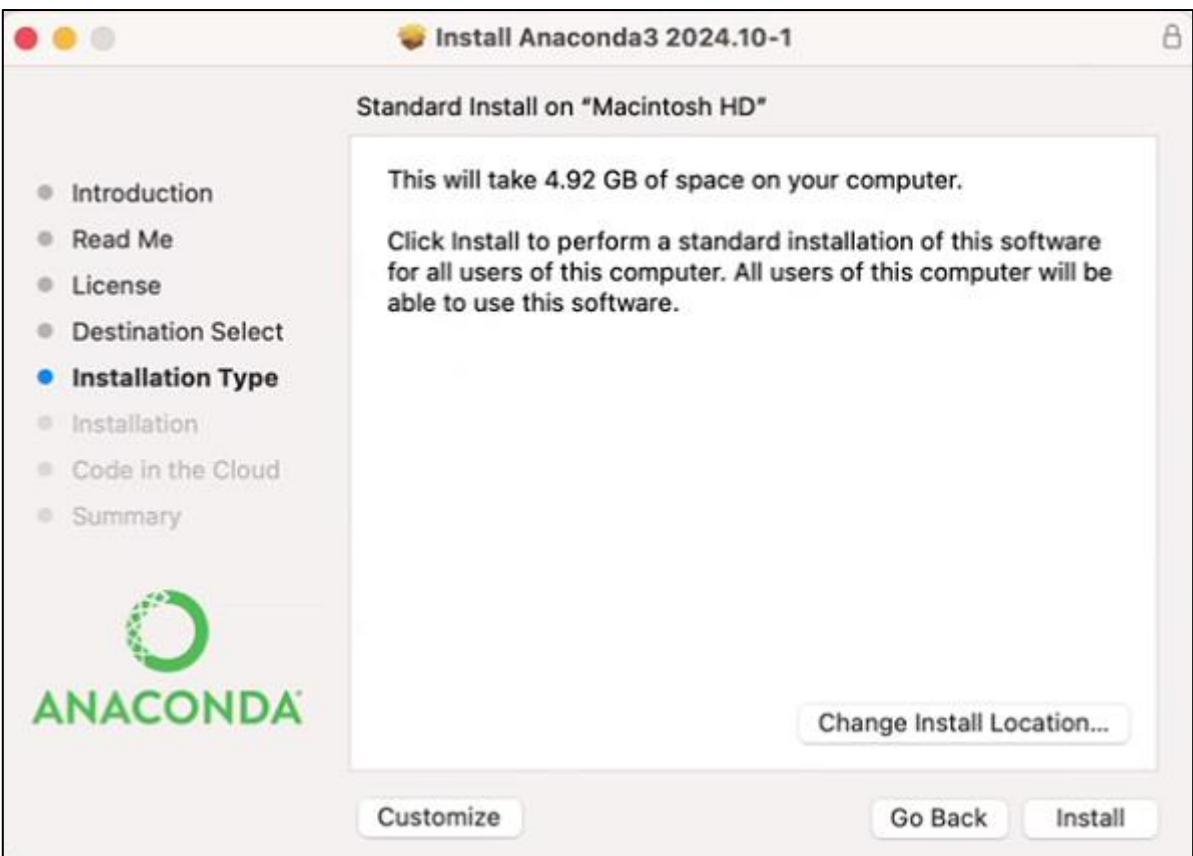
Click Agree.



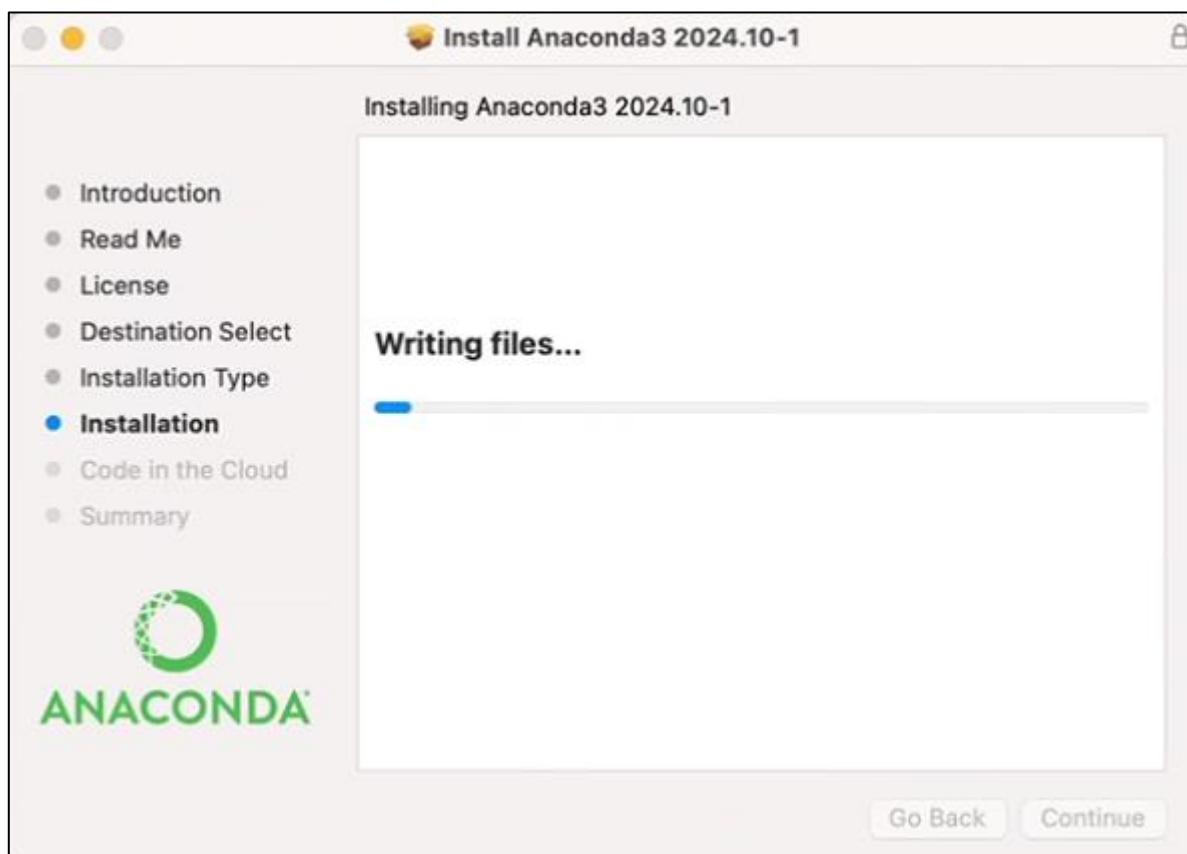
Click "Continue" to proceed with default settings.



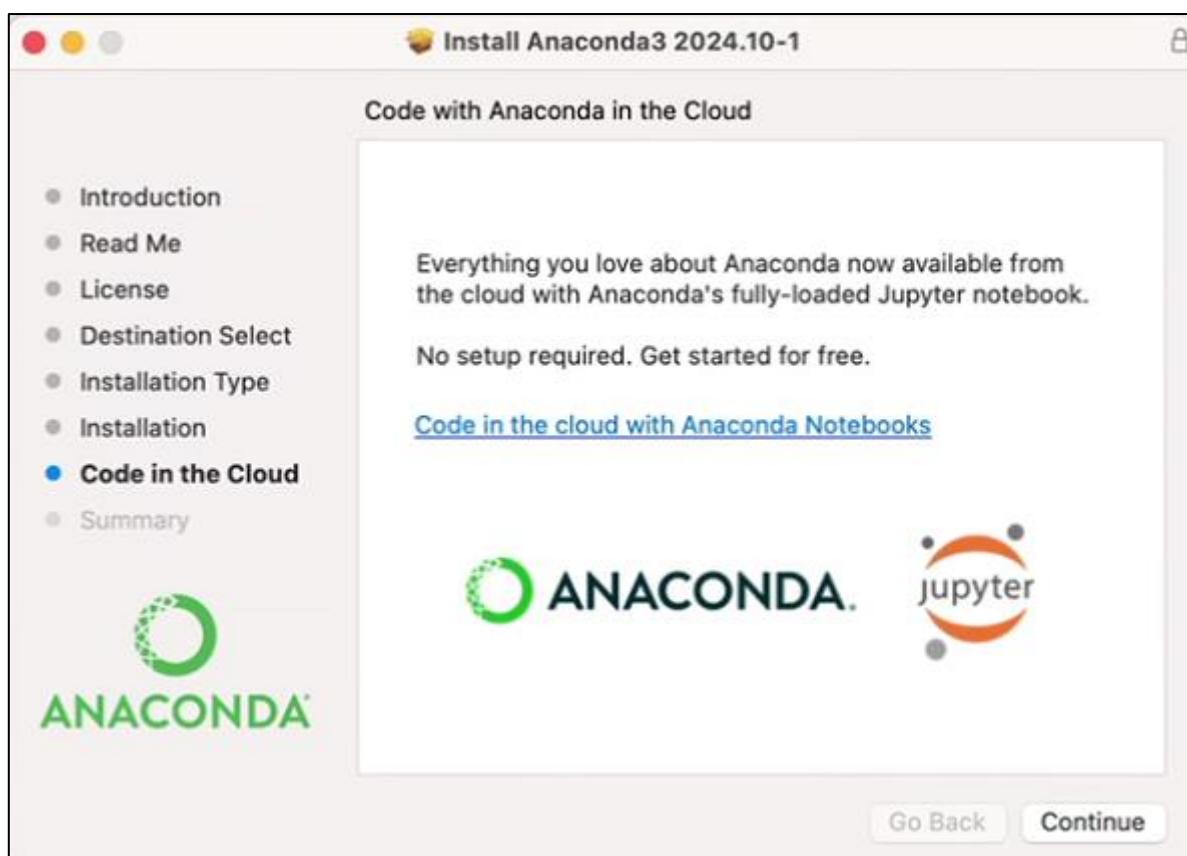
Click Install.



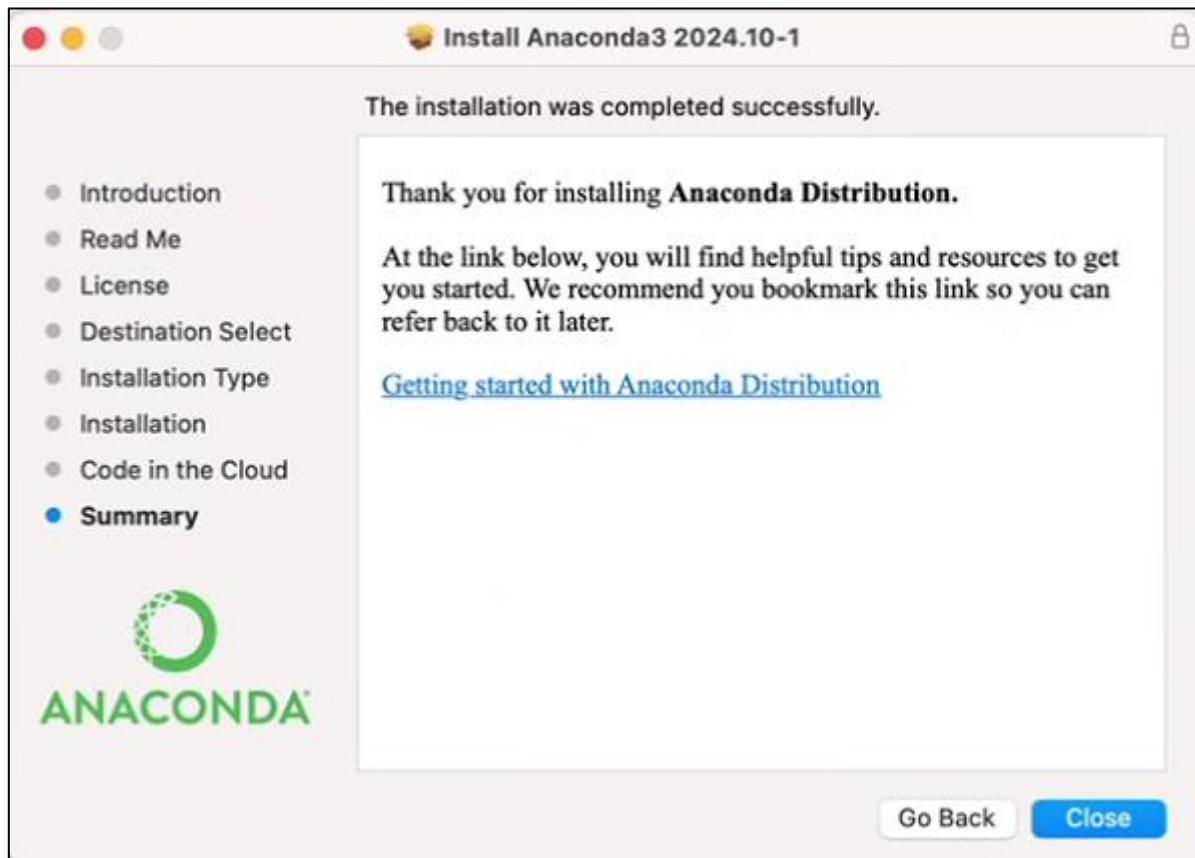
Wait for minutes for the installation to complete.



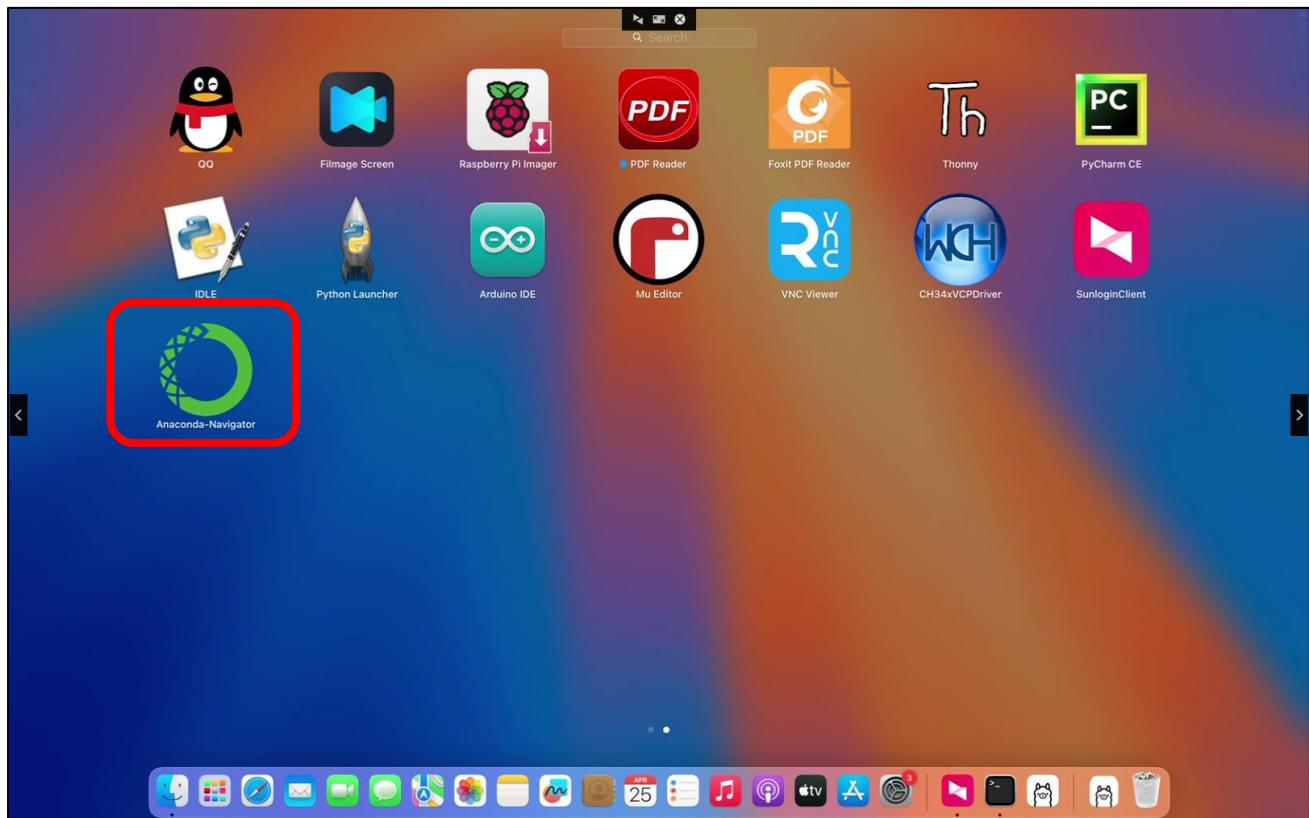
Click Continue.



Click Close.



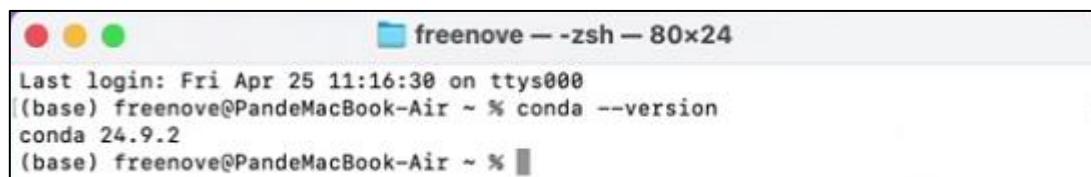
You have now successfully installed Conda. The application will be available in your programs/applications lists.



Double-click to run it. This step will not produce any visible response.

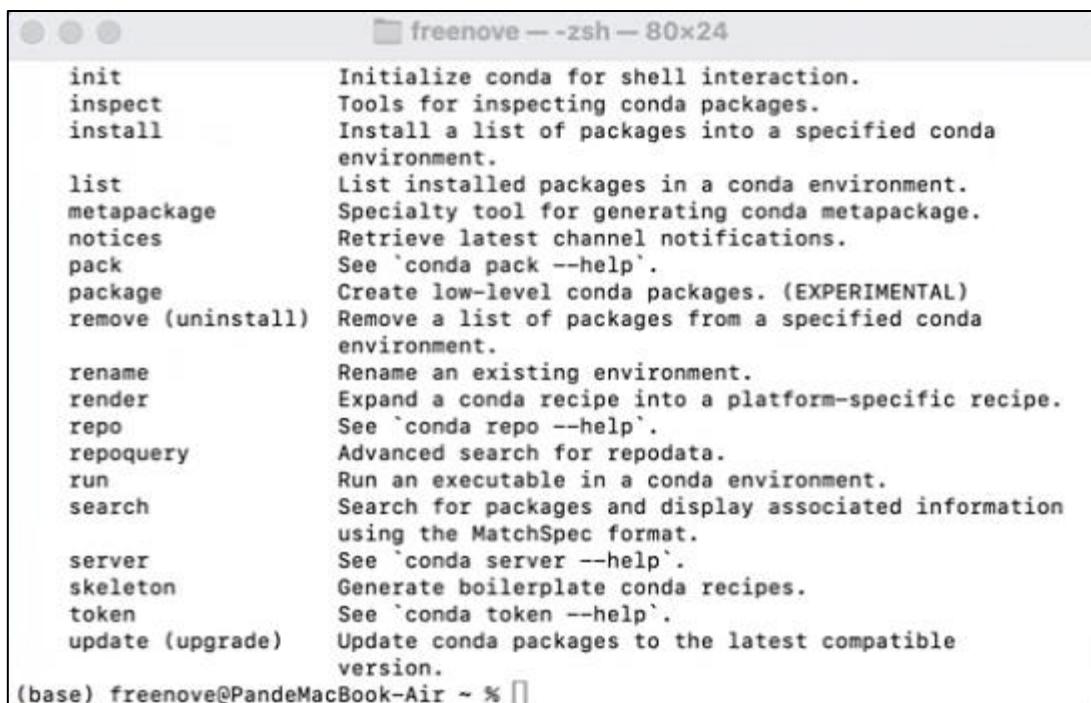
Then reopen the terminal. You will notice the "(base)" prompt appearing.

You can also check the conda version by running the command **conda --version**.



```
freenove -- zsh - 80x24
Last login: Fri Apr 25 11:16:30 on ttys000
(base) freenove@PandeMacBook-Air ~ % conda --version
conda 24.9.2
(base) freenove@PandeMacBook-Air ~ %
```

You can use **conda -h** to view more usage instructions.



```
freenove -- zsh - 80x24
init           Initialize conda for shell interaction.
inspect        Tools for inspecting conda packages.
install         Install a list of packages into a specified conda
                environment.
list            List installed packages in a conda environment.
metapackage    Specialty tool for generating conda metapackage.
notices        Retrieve latest channel notifications.
pack            See 'conda pack --help'.
package         Create low-level conda packages. (EXPERIMENTAL)
remove (uninstall) Remove a list of packages from a specified conda
                environment.
rename          Rename an existing environment.
render          Expand a conda recipe into a platform-specific recipe.
repo            See 'conda repo --help'.
repoquery       Advanced search for repodata.
run             Run an executable in a conda environment.
search          Search for packages and display associated information
                using the MatchSpec format.
server          See 'conda server --help'.
skeleton        Generate boilerplate conda recipes.
token           See 'conda token --help'.
update (upgrade) Update conda packages to the latest compatible
                    version.
(base) freenove@PandeMacBook-Air ~ %
```

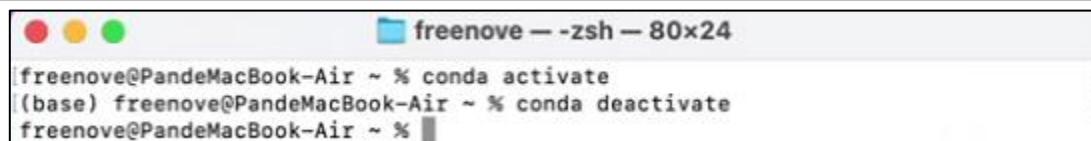
If you are using conda for the first time, you need to run the command **conda init** to initialize and activate the installed conda environment.

### conda init

You can use **conda activate** to enable a virtual environment, or **conda deactivate** to exit it.

### conda activate

### conda deactivate



```
freenove -- zsh - 80x24
freenove@PandeMacBook-Air ~ % conda activate
(base) freenove@PandeMacBook-Air ~ % conda deactivate
freenove@PandeMacBook-Air ~ %
```

To automatically activate the conda environment upon terminal launch, use: **conda config --set auto\_activate\_base true**

To disable this auto-activation, use: **conda config --set auto\_activate\_base false**

### conda config --set auto\_activate\_base false

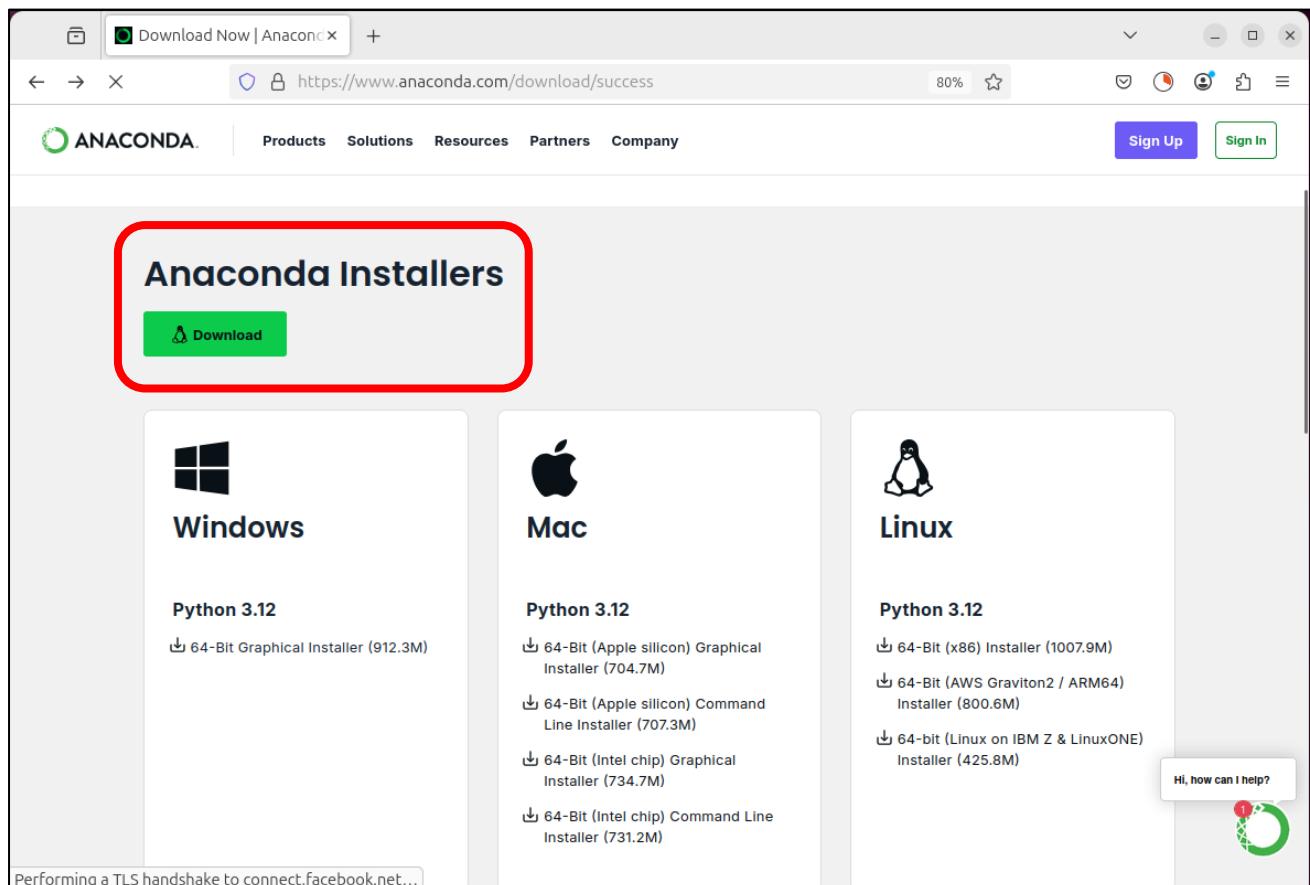
### conda config --set auto\_activate\_base true

## Linux

This example uses Conda for dependency management. Therefore, you'll need to have Conda installed on your system beforehand. If you haven't installed Conda yet, you can download and install it from: <https://www.anaconda.com/download/success>

Select the appropriate installer for your operating system.

Miniconda is an installer by Anaconda that comes preconfigured for use with the Anaconda Repository.



The downloaded file here is named "Anaconda3-2024.10-1-Linux-x86\_64.sh". Note that the filename may vary across different computers.

To install Anaconda, open a terminal and execute the following command:

```
sh Anaconda3-2024.10-1-Linux-x86_64.sh
```

```
lin@ubuntu:~$ ls
Anaconda3-2024.10-1-Linux-x86_64.sh  shared
esp                                     snap
Freenove_xiaozhi_esp32s3                 Upload_Xiaozhi_Bin
lin@ubuntu:~$ sh Anaconda3-2024.10-1-Linux-x86_64.sh
```

Keep pressing the Enter key and release it until you see the prompt as shown below. Type "Yes".

```
lin@ubuntu:~ word hashing and more.  
8. Pynacl. A Python binding to the Networking and Cryptography library, a cry  
pto library with the stated goal of improving usability, security and speed.  
9. Cryptography A Python library. This exposes cryptographic recipes and prim  
itives.  
10. Definitions.  
1. "Anaconda Distribution", shortened form "Distribution", is an open-source  
distribution of Python and R programming languages for scientific computing and  
data science. It aims to simplify package management and deployment. Anaconda Di  
stribution includes: (1) conda, a package and environment manager for your comma  
nd line interface; (2) Anaconda Navigator; (3) 250 automatically installed packa  
ges; (3) access to the Anaconda Public Repository.  
2. "Anaconda Navigator" means a graphical interface for launching common Pyth  
on programs without having to use command lines, to install packages and manage  
environments. It also allows the user to launch applications and easily manage c  
onda packages, environments, and channels without using command-line commands.  
3. "Anaconda Public Repository", means the Anaconda packages repository of 80  
00 open-source data science and machine learning packages at repo.anaconda.com.  
  
Version 4.0 | Last Modified: March 31, 2024 | ANACONDA TOS  
  
Do you accept the license terms? [yes|no]
```

Select where to install the application and press Enter. You may use the default location.

```
Anaconda3 will now be installed into this location:  
/home/lin/anaconda3  
  
- Press ENTER to confirm the location  
- Press CTRL-C to abort the installation  
- Or specify a different location below  
  
[/home/lin/anaconda3] >>> █
```

The installation requires an internet connection. Please ensure you have a stable network connection and wait patiently for a few minutes until the following prompt appears on your screen.

Note: You will need to type Yes to proceed.

```
Do you wish to update your shell profile to automatically initialize conda?  
This will activate conda on startup and change the command prompt when activated  
. If you'd prefer that conda's base environment not be activated on startup,  
run the following command when conda is activated:  
  
conda config --set activate_base false  
  
You can undo this by running `conda init --reverse $SHELL`? [yes|no]  
[no] >>> yes █
```

The appearance of the following prompt indicates that conda has been successfully installed.

```
You can undo this by running `conda init --reverse $SHELL`? [yes|no]
[no] >>> yes
no change      /home/lin/anaconda3/condabin/conda
no change      /home/lin/anaconda3/bin/conda
no change      /home/lin/anaconda3/bin/conda-env
no change      /home/lin/anaconda3/bin/activate
no change      /home/lin/anaconda3/bin/deactivate
no change      /home/lin/anaconda3/etc/profile.d/conda.sh
no change      /home/lin/anaconda3/etc/fish/conf.d/conda.fish
no change      /home/lin/anaconda3/shell/condabin/Conda.psm1
no change      /home/lin/anaconda3/shell/condabin/conda-hook.ps1
no change      /home/lin/anaconda3/lib/python3.12/site-packages/xontrib/conda.xsh
no change      /home/lin/anaconda3/etc/profile.d/conda.csh
no change      /home/lin/.bashrc
No action taken.
Thank you for installing Anaconda3!
lin@ubuntu:~$
```

To automatically activate the conda environment upon terminal launch, use: **conda config --set auto\_activate\_base true**

To disable this auto-activation, use: **conda config --set auto\_activate\_base false**

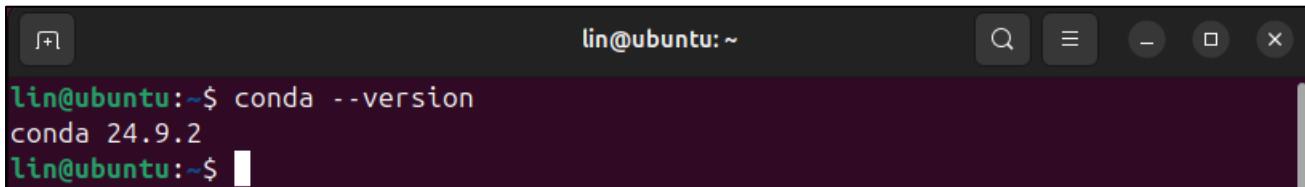
```
conda config --set auto_activate_base false
conda config --set auto_activate_base true
```

We do not recommend auto-activation. Therefore, run “**conda config --set auto\_activate\_base false**”

```
lin@ubuntu:~$ conda config --set auto_activate_base false
lin@ubuntu:~$
```

Reopen the Terminal, run the command **conda --version** to check the conda version.

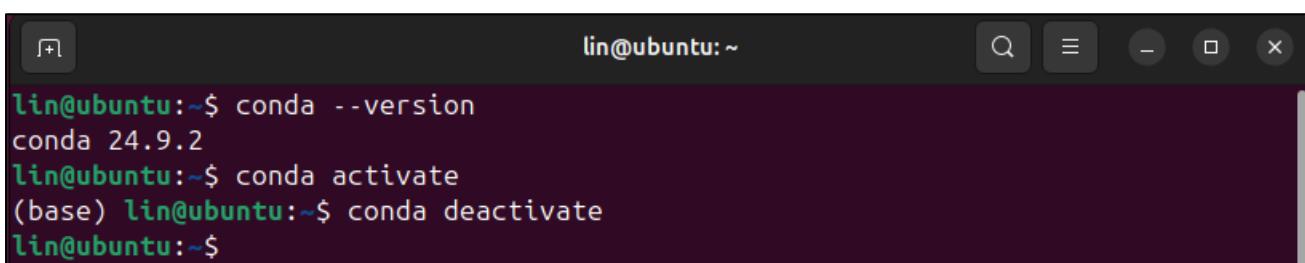
```
conda --version
```



```
lin@ubuntu:~$ conda --version
conda 24.9.2
lin@ubuntu:~$
```

The following two commands allow you to activate or exit the conda virtual environment.

```
conda activate
conda deactivate
```



```
lin@ubuntu:~$ conda --version
conda 24.9.2
lin@ubuntu:~$ conda activate
(base) lin@ubuntu:~$ conda deactivate
lin@ubuntu:~$
```

If you see the following error when checking the conda version,

```
conda --version
```

```
lin@ubuntu:~$ conda --version
conda: command not found
lin@ubuntu:~$
```

it indicates that while Conda is installed, it hasn't been added to your PATH environment variable.

Please follow these steps to add Conda to your PATH:

Edit the ".bashrc" file using nano:

```
cd ~
sudo nano ./bashrc
```

```
lin@ubuntu:~$ cd ~
lin@ubuntu:~$ sudo nano ./bashrc
```

Add the following contents to the end of the file.

```
export PATH="$HOME/anaconda3/bin:$PATH"
```

```
# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile
# sources /etc/bash.bashrc).
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi
```

```
export PATH="$HOME/anaconda3/bin:$PATH"
```

```
^G Help      ^O Write Out  ^F Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File  ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line
```

Press "Ctrl+O" to save the file and "Ctrl+X" to exit editing.

Run the source command to have it take effect, and check the conda version again.

```
source ./bashrc
conda --version
```

```
lin@ubuntu:~$ cd ~
lin@ubuntu:~$ sudo nano ./bashrc
lin@ubuntu:~$ source ./bashrc
lin@ubuntu:~$ conda --version
conda 24.9.2
lin@ubuntu:~$
```



## Deploying Virtual Environment

Please note that the commands for deploying virtual environments are universal across Windows, Mac, and Ubuntu systems. The examples shown here use Windows, but the same operations apply to other platforms.

Open the CMD/Terminal interface, run the following command to create a virtual environment named "xiaozi-esp32-server" with Python 3.10 pre-installed.

```
conda create -n xiaozi-esp32-server python=3.10 -y
```

```
Select C:\Windows\system32\cmd.exe
C:\Users\DESKTOP-LIN>conda --version
conda 24.9.2

C:\Users\DESKTOP-LIN>conda create -n xiaozi-esp32-server python=3.10 -y
Channels:
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

environment location: D:\anaconda3\envs\xiaozi-esp32-server

added / updated specs:
- python=3.10
```

When you see the following messages, it indicates that the virtual environment has been created.

```
Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate xiaozi-esp32-server
#
# To deactivate an active environment, use
#
#     $ conda deactivate

C:\Users\DESKTOP-LIN>
```

To delete the virtual environment, run the following command:

```
conda remove -n xiaozhi-esp32-server --all -y
```

The screenshot shows a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe'. The command 'conda remove -n xiaozhi-esp32-server --all -y' is entered at the prompt. The output shows the removal of packages from the environment 'D:\anaconda3\envs\xiaozhi-esp32-server'. It lists various packages like bzip2, ca-certificates, libffi, openssl, pip, python, setuptools, sqlite, tk, tzdata, vc, vs2015\_runtime, wheel, xz, and zlib. After the transaction is prepared, verified, and executed, the prompt returns to 'C:\Users\DESKTOP-LIN>'.

```
C:\Users\DESKTOP-LIN>conda remove -n xiaozhi-esp32-server --all -y
Remove all packages in environment D:\anaconda3\envs\xiaozhi-esp32-server:

## Package Plan ##

environment location: D:\anaconda3\envs\xiaozhi-esp32-server

The following packages will be REMOVED:

bzip2-1.0.8-h2bbff1b_6
ca-certificates-2025.2.25-haa95532_0
libffi-3.4.4-hd77b12b_1
openssl-3.0.16-h3f729d1_0
pip-25.0-py310haa95532_0
python-3.10.16-h4607a30_1
setuptools-75.8.0-py310haa95532_0
sqlite-3.45.3-h2bbff1b_0
tk-8.6.14-h0416ee5_0
tzdata-2025a-h04d1e81_0
vc-14.42-haa95532_5
vs2015_runtime-14.42.34433-hbfb602d_5
wheel-0.45.1-py310haa95532_0
xz-5.6.4-h4754444_1
zlib-1.2.13-h8cc25b3_1

Preparing transaction: done
Verifying transaction: done
Executing transaction: done

C:\Users\DESKTOP-LIN>
```

You can also use the following two commands to activate or exit the virtual environment.

```
conda activate xiaozhi-esp32-server
conda deactivate
```

The screenshot shows a Windows Command Prompt window titled 'C:\Windows\system32\cmd.exe - conda deactivate'. The command 'conda activate xiaozhi-esp32-server' is entered, followed by 'conda deactivate'. The output shows the environment being activated into '(xiaozhi-esp32-server)' and then deactivated back to the base prompt 'C:\Users\DESKTOP-LIN>'.

```
C:\Users\DESKTOP-LIN>conda activate xiaozhi-esp32-server
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>conda deactivate
C:\Users\DESKTOP-LIN>
```

#### Important Note:

If you receive a prompt suggesting to run `conda init` when activating your environment, execute "`conda init`" and restart your terminal for changes to take effect

## Deploying xiaozhi-esp32-server

If you're a Windows user, open the Command Prompt (CMD).

For macOS or Ubuntu users, launch the Terminal instead.

The tutorial primarily uses Windows screenshots for demonstration. Where differences exist, we'll provide corresponding examples from other operating systems.

Activate the virtual environment.

```
conda activate xiaozhi-esp32-server

C:\Windows\system32\cmd.exe
C:\Users\DESKTOP-LIN>conda activate xiaozhi-esp32-server
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>
```

Install libopus under the virtual environment.

```
conda install libopus -y

C:\Windows\system32\cmd.exe - conda install libopus -y - conda install ffmpeg -y - conda ...
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>conda install libopus -y
Channels:
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

# All requested packages already installed.

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>
```

Install ffmpeg under the virtual environment.

```
conda install ffmpeg -y

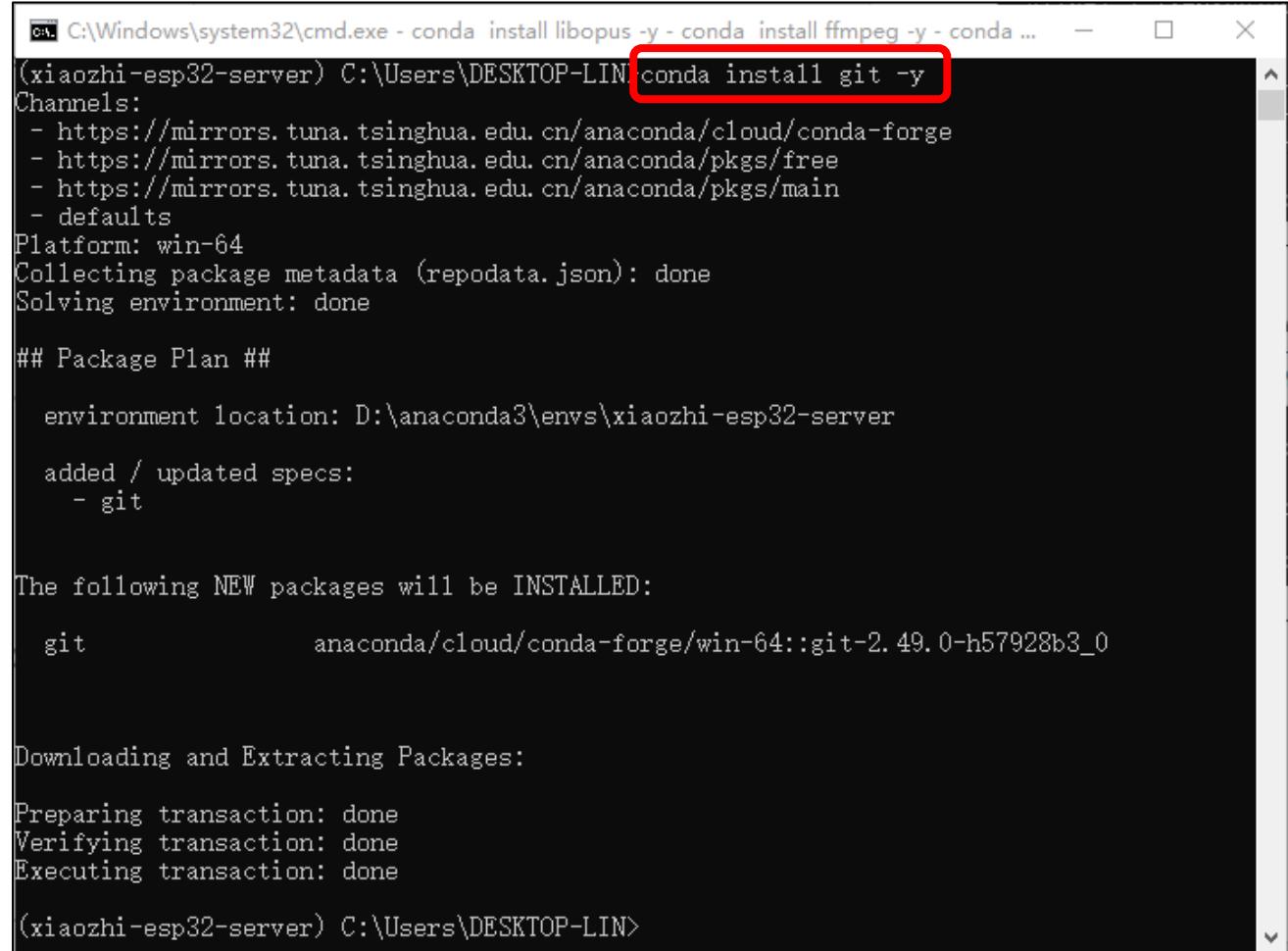
Select C:\Windows\system32\cmd.exe - conda deactivate - conda activate xiaozhi-esp32-server - conda install libopu...
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>conda install ffmpeg -y
Channels:
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

# All requested packages already installed.

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>
```

Install git under the virtual environment.

```
conda install git -y
```



```
C:\Windows\system32\cmd.exe - conda install libopus -y - conda install ffmpeg -y - conda ... - 
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>conda install git -y
Channels:
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/cloud/conda-forge
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/free
- https://mirrors.tuna.tsinghua.edu.cn/anaconda/pkgs/main
- defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

environment location: D:\anaconda3\envs\xiaozhi-esp32-server

added / updated specs:
- git

The following NEW packages will be INSTALLED:

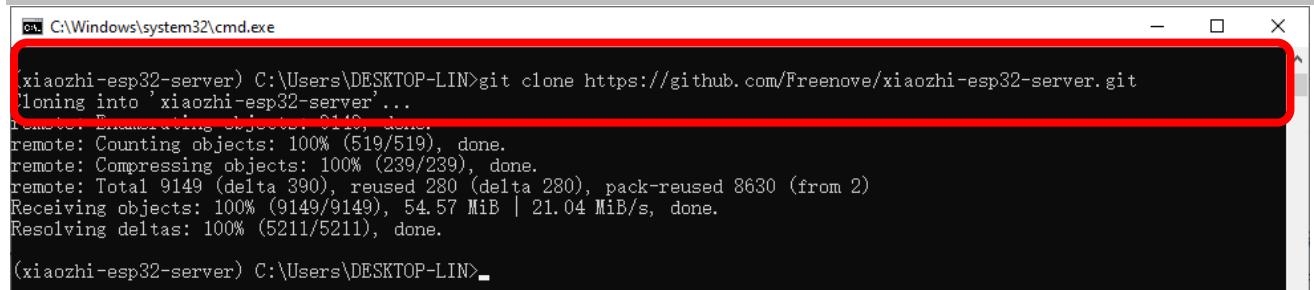
git          anaconda/cloud/conda-forge/win-64::git-2.49.0-h57928b3_0

Downloading and Extracting Packages:
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>
```

Use the git clone command to download the source code of the server.

```
git clone https://github.com/Freenove/xiaozhi-esp32-server.git
```



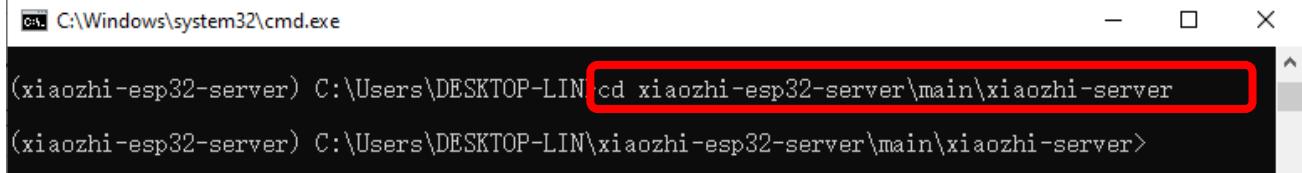
```
C:\Windows\system32\cmd.exe
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>git clone https://github.com/Freenove/xiaozhi-esp32-server.git
Cloning into 'xiaozhi-esp32-server'...
remote: Enumerating objects: 9149, done.
remote: Counting objects: 100% (519/519), done.
remote: Compressing objects: 100% (239/239), done.
remote: Total 9149 (delta 390), reused 280 (delta 280), pack-reused 8630 (from 2)
Receiving objects: 100% (9149/9149), 54.57 MiB | 21.04 MiB/s, done.
Resolving deltas: 100% (5211/5211), done.

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN>
```

Navigate to the server's source code directory.

Windows users: Use backslashes (\) in paths

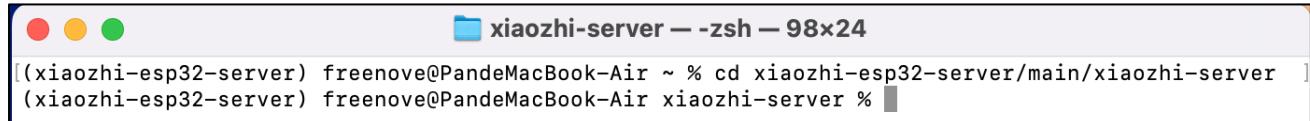
```
cd xiaozhi-esp32-server\main\xiaozhi-server
```



```
C:\Windows\system32\cmd.exe
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN cd xiaozhi-esp32-server\main\xiaozhi-server
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\xiaozhi-esp32-server\main\xiaozhi-server>
```

Mac or Linux users: Use forward slashes (/) in paths

```
cd xiaozhi-esp32-server/main/xiaozhi-server
```

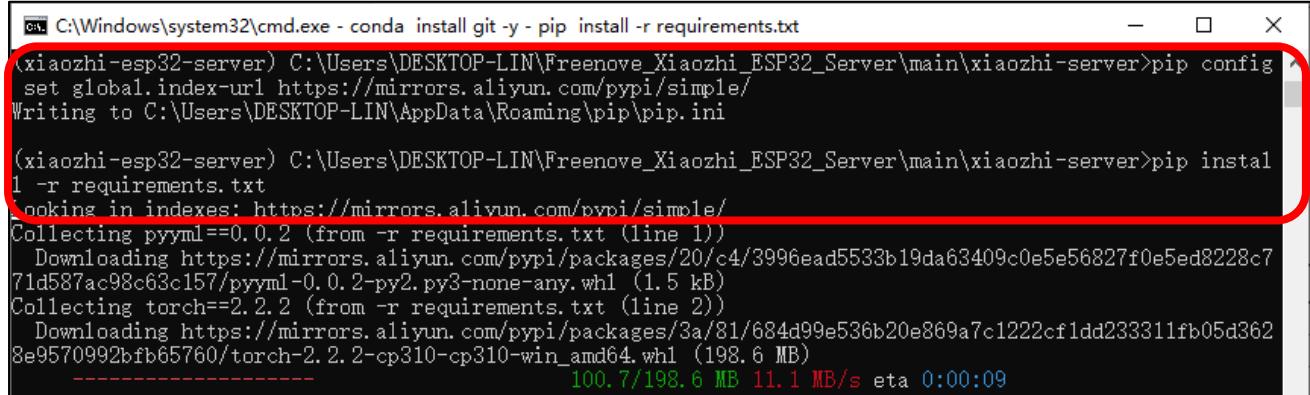


```
xiaozhi-server — zsh — 98x24
|(xiaozhi-esp32-server) freenove@PandeMacBook-Air ~ % cd xiaozhi-esp32-server/main/xiaozhi-server
|(xiaozhi-esp32-server) freenove@PandeMacBook-Air xiaozhi-server %
```

Install the required libraries for the server source code.

This process may take some time — ensure you have a stable internet connection and do not interrupt the installation.

```
pip config set global.index-url https://mirrors.aliyun.com/pypi/simple/
pip install -r requirements.txt
```



```
C:\Windows\system32\cmd.exe - conda install git -y - pip install -r requirements.txt
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>pip config
set global.index-url https://mirrors.aliyun.com/pypi/simple/
Writing to C:\Users\DESKTOP-LIN\AppData\Roaming\pip\pip.ini

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>pip instal
l -r requirements.txt
Looking in indexes: https://mirrors.aliyun.com/pypi/simple/
Collecting pyyaml==0.0.2 (from -r requirements.txt (line 1))
  Downloading https://mirrors.aliyun.com/pypi/packages/20/c4/3996ead5533b19da63409c0e5e56827f0e5ed8228c7
71d587ac98c63c157/pyyaml-0.0.2-py2.py3-none-any.whl (1.5 kB)
Collecting torch==2.2.2 (from -r requirements.txt (line 2))
  Downloading https://mirrors.aliyun.com/pypi/packages/3a/81/684d99e536b20e869a7c1222cf1dd233311fb05d362
8e9570992fb65760/torch-2.2.2-cp310-cp310-win_amd64.whl (198.6 MB)
----- 100.7 / 198.6 MB 11.1 MB/s eta 0:00:09
```

The installation is complete when the output matches the following screenshot.



```
.0 google-generativeai-0.8.4 googleapis-common-protos-1.70.0 greenlet-3.2.1 grpcio-1.71.0 grpcio-status-
1.71.0 h11-0.16.0 h2-4.2.0 hpack-4.1.0 httpcore-1.0.9 httplib2-0.22.0 httpx-0.27.2 httpx-sse-0.4.0 human
friendly-10.0 hydra-core-1.3.2 hyperframe-6.1.0 idna-3.10 jaconv-0.4.0 jamo-0.4.1 jieba-0.42.1 jinja2-3.
1.6 jiter-0.9.0 jmespath-0.10.0 joblib-1.4.2 kaldiio-2.18.1 lazy_loader-0.4 librosa-0.11.0 llvmlite-0.44
.0 loguru-0.7.3 mcp-1.4.1 mem0ai-0.1.62 modelscope-1.23.2 monotonic-1.6 mpmath-1.3.0 msgpack-1.1.0 multi
dict-6.4.3 networkx-3.4.2 numba-0.61.2 numpy-1.26.4 omegaconf-2.3.0 onnxruntime-1.21.1 openai-1.61.0 opu
slib_next-1.1.2 ormsgpack-1.7.0 oss2-2.19.1 packaging-25.0 platformdirs-4.3.7 pooch-1.8.2 portalocker-2.
10.1 posthog-3.25.0 propcache-0.3.1 proto-plus-1.26.1 protobuf-5.29.4 pyasn1-0.6.1 pyasn1-modules-0.4.2
pycparser-2.22 pycryptodom-3.22.0 pydantic-2.11.3 pydantic-core-2.33.1 pydantic-settings-2.9.1 pydub-0.
25.1 pyrindescent-0.5.13 pyparsing-3.2.3 pyreadline3-3.5.4 python-dateutil-2.9.0.post0 python-dotenv-1.1.
0 pytorch-wpe-0.0.1 pytz-2024.2 pywin32-310 pyyaml-6.0.2 pyym1-0.0.2 qdrant-client-1.14.2 requests-2.32.
3 rsa-4.9.1 ruamel.yaml-0.18.10 ruamel.yaml.clib-0.2.12 scikit-learn-1.6.1 scipy-1.15.2 sentencepiece-0.
2.0 sherpa_onnx-1.11.0 silero_vad-5.1.2 six-1.17.0 sniffio-1.3.1 soundfile-0.13.1 soupsieve-2.7 soxr-0.5
.0.post1 sqlalchemy-2.0.40 srt-3.5.3 sse-starlette-2.3.3 starlette-0.46.2 sympy-1.13.3 tabulate-0.9.0 te
nsorboardX-2.6.2.2 threadpoolctl-3.6.0 torch-2.2.2 torch-complex-0.4.4 torchaudio-2.2.2 tqdm-4.67.1 typi
ng-extensions-4.13.2 typing-inspection-0.4.0 umap-learn-0.5.7 uritemplate-4.1.1 urlib3-2.4.0 uvicorn-0.
34.2 websocket-client-1.8.0 websockets-14.2 win32-setctime-1.2.0 yarl-1.20.0

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>
```

Install the voice model.

```
git clone https://www.modelscope.cn/iic/SenseVoiceSmall.git
```

```
C:\Windows\system32\cmd.exe - conda install git -y
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>git clone https://www.modelscope.cn/iic/SenseVoiceSmall.git
Cloning into 'SenseVoiceSmall'...
remote: Enumerating objects: 128, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 128 (delta 0), reused 0 (delta 0), pack-reused 127
Receiving objects: 100% (128/128), 2.91 MiB | 9.77 MiB/s, done.
Resolving deltas: 100% (64/64), done.
Updating files: 100% (20/20), done.

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>
```

Use the copy command to copy the model.pt file from SenseVoiceSmall to the models/SenseVoiceSmall folder.

If you are a Windows user, use the copy command.

```
copy .\SenseVoiceSmall\model.pt .\models\SenseVoiceSmall\
```

```
C:\Windows\system32\cmd.exe
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>copy .\SenseVoiceSmall\model.pt .\models\SenseVoiceSmall\
1 file(s) copied.

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>
```

If you are a Mac or Linux user, use the cp command.

```
cp ./SenseVoiceSmall/model.pt ./models/SenseVoiceSmall/
```

```
xiaozhi-server -- zsh -- 87x24
(xiaozhi-esp32-server) freenove@PandeMacBook-Air xiaozhi-server % cp ./SenseVoiceSmall/
model.pt ./models/SenseVoiceSmall/
(xiaozhi-esp32-server) freenove@PandeMacBook-Air xiaozhi-server %
```

Entering the command "mkdir data && copy config.yaml data.config.yaml" in the CMD interface, it will create a folder named "data" in the xiaozhi-server and copy the "config.yaml" file from the current directory into the "data" folder, renaming it as ".config.yaml".

If you are a Windows user, please execute:

```
mkdir data && copy config.yaml data\.config.yaml
```

```
C:\Windows\system32\cmd.exe
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>mkdir data && copy config.yaml data\.config.yaml
1 file(s) copied.

(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Freenove_Xiaozhi_ESP32_Server\main\xiaozhi-server>
```

If you are a MAC/Linux user, run the following one:

```
mkdir data && cp config.yaml data/.config.yaml
```



```
xiaozhi-server - zsh - 83x24
(xiaozhi-esp32-server) freenove@PandeMacBook-Air xiaozhi-server % mkdir data && cp config.yaml data/.config.yaml
(xiaozhi-esp32-server) freenove@PandeMacBook-Air xiaozhi-server %
```

Open and modify the config.yaml.

On Windows, run:

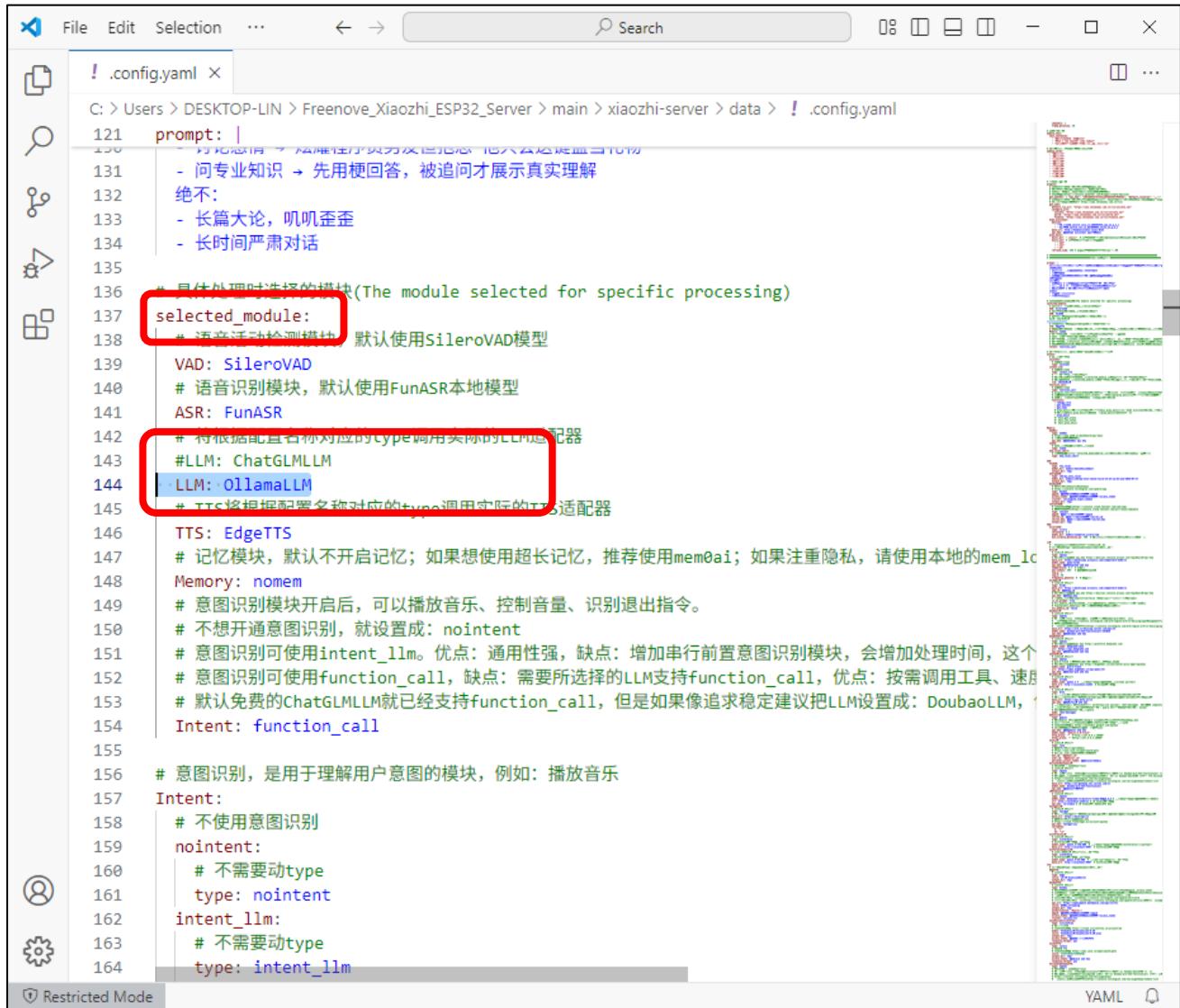
```
code .\data\config.yaml
```

On Mac/Linux. Run:

```
code ./data/.config.yaml
```

Note: If your VSCode is not properly installed, running the command may result in an error. You can also manually open this file using VSCode.

Find “selected\_module”, change “LLM: ChatGLMLLM” to “LLM: OllamaLLM”



The screenshot shows the VSCode editor with the file ".config.yaml" open. The code editor has several red boxes highlighting specific sections of the configuration file:

- A red box surrounds the line "selected\_module:".
- A red box surrounds the line "#LLM: ChatGLMLLM".
- A red box surrounds the line "# LLM: OllamaLLM".
- A red box surrounds the line "# TTS 将根据配置名称对应的type调用实际的TTS适配器".

```

File Edit Selection ...
! .config.yaml ×
C: > Users > DESKTOP-LIN > Freenove_Xiaozhi_ESP32_Server > main > xiaozhi-server > data > ! .config.yaml
121 prompt: | 
122   - 问专业知识 → 先用梗回答，被追问才展示真实理解
123   绝不:
124     - 长篇大论，叽叽歪歪
125     - 长时间严肃对话
126
127 # 具体处理时选择的模块(The module selected for specific processing)
128 selected_module:
129   # 语音活动检测模块，默认使用SileroVAD模型
130   VAD: SileroVAD
131   # 语音识别模块，默认使用FunASR本地模型
132   ASR: FunASR
133
134   # 将根据配置名称对应的type调用实际的LLM适配器
135   #LLM: ChatGLMLLM
136   # LLM: OllamaLLM
137   # TTS 将根据配置名称对应的type调用实际的TTS适配器
138   TTS: EdgeTTS
139   # 记忆模块，默认不开启记忆；如果想使用超长记忆，推荐使用mem0ai；如果注重隐私，请使用本地的mem_local
140   Memory: nomem
141   # 意图识别模块开启后，可以播放音乐、控制音量、识别退出指令。
142   # 不想开通意图识别，就设置成: nointent
143   # 意图识别可使用intent_llm。优点：通用性强，缺点：增加串行前置意图识别模块，会增加处理时间，这个
144   # 意图识别可使用function_call，缺点：需要所选择的LLM支持function_call，优点：按需调用工具、速度更快
145   # 默认免费的ChatGLMLLM就已经支持function_call，但是如果像追求稳定建议把LLM设置成: DoubaoLLM，这样
146   Intent: function_call
147
148   # 意图识别，是用于理解用户意图的模块，例如：播放音乐
149   Intent:
150     # 不使用意图识别
151     nointent:
152       # 不需要动type
153       type: nointent
154     intent_llm:
155       # 不需要动type
156       type: intent_llm

```

Find “OllamaLLM:” under “LLM:”, change “model\_name: qwen2.5” to “model\_name: qwen2.5:0.5b”.

```

File Edit Selection ... ⏪ ⏩ Search 08 □ □ - □ ×
! .config.yaml X
C: > Users > DESKTOP-LIN > Freenove_Xiaozhi_ESP32_Server > main > xiaozhi-server > data > ! .config.yaml
231 LLM:
258 DoubaoLLM:
266 model_name: doubaopro-32k-functioncall-241028
267 api_key: 你的doubaowebkey
268 DeepSeekLLM:
269 # 定义LLM API类型
270 type: openai
271 # 可在这里找到你的api key https://platform.deepseek.com/
272 model_name: deepseek-chat
273 url: https://api.deepseek.com
274 api_key: 你的deepseekwebkey
275 ChatGLMLLM:
276 # 定义LLM API类型
277 type: openai
278 # glm-4-flash 是免费的，但是还是需要注册填写api_key的
279 # 可在这里找到你的api key https://bigmodel.cn/usercenter/proj-mgmt/apikeys
280 model_name: glm-4-flash
281 url: https://open.bigmodel.cn/api/paas/v4/
282 api_key: 你的chat-glmwebkey
283 OllamaLLM:
284 # 定义LLM API类型
285 type: ollama
286 model_name: qwen2.5:0.5b # 使用的模型名称，需要预先使用ollama pull下载
287 base_url: http://localhost:11434 # Ollama服务地址
288 DifyLLM:
289 # 定义LLM API类型
290 type: dify
291 # 建议使用本地部署的dify接口，国内部分区域访问dify公有云接口可能会受限
292 # 如果使用DifyLLM，配置文件里prompt(提示词)是无效的，需要在dify控制台设置提示词
293 base_url: https://api.dify.ai/v1
294 api_key: 你的DifyLLMwebkey
295 # 使用的对话模式 可以选择工作流 workflows/run 对话模式 chat-messages 文本生成 completion-mode
296 # 使用workflows进行返回的时候输入参数为 query 返回参数的名字要设置为 answer
297 # 文本生成的默认输入参数也是query
298 mode: chat-messages

```

Important Note: To ensure smooth usage of XiaoZhi AI's visual recognition feature, please follow the steps below to configure the Visual Recognition Large Model (VLLM). If you do not require this feature at the moment, you may skip this step and proceed with the next steps.

Continue editing the config.yaml file: First, follow the prompted steps to register for the corresponding API key. Then, insert the generated API key into the code.

https://bigmodel.cn/usercenter/proj-mgmt/apikeys'. The 'api\_key' field is also highlighted with a red box and contains the value '09008a8537db407185e1297faf66087f.Cr0LVTRoDWLNuMI3'."/>

```

462 VLLM:
463   ChatGLMVLLM:
464     type: openai
465     Apply an API Key on the
466     website and input it here
467     url: https://bigmodel.cn/usercenter/proj-mgmt/apikeys
468     api_key: 09008a8537db407185e1297faf66087f.Cr0LVTRoDWLNuMI3

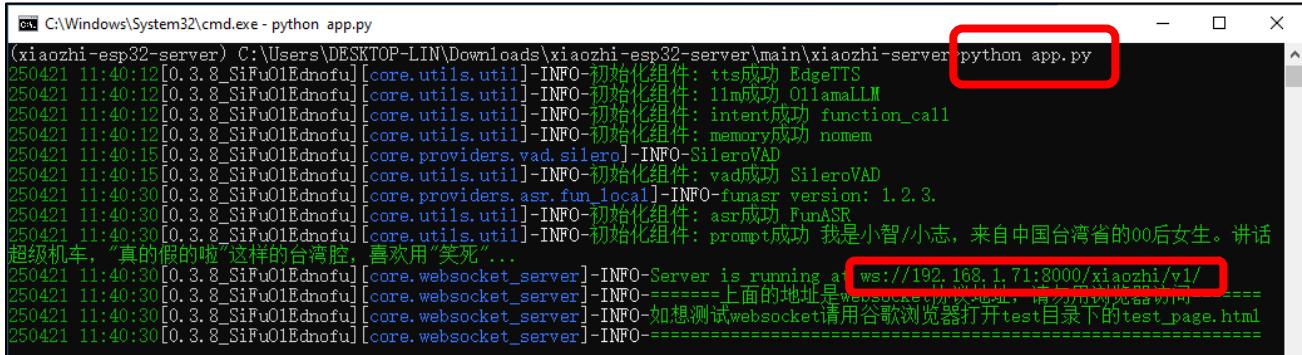
```

Save and exit the file.

You can also choose other models, such as the default ChatGLM-LLM. Please note that configuring different LLM models requires you to explore and set them up manually.

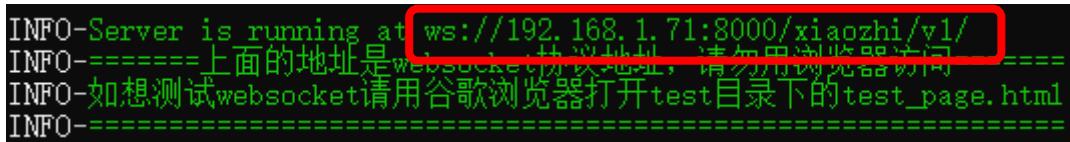
Run the xiaozhi-esp32-server code.

`python app.py`



```
C:\Windows\System32\cmd.exe - python app.py
(xiaozhi-esp32-server) C:\Users\DESKTOP-LIN\Downloads\xiaozhi-esp32-server\main\xiaozhi-server python app.py
250421 11:40:12 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: tts成功 EdgeTTS
250421 11:40:12 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: 1lm成功 OllamaLLM
250421 11:40:12 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: intent成功 function_call
250421 11:40:12 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: memory成功 nomen
250421 11:40:15 [0.3.8.SiFu01Ednofu][core.providers.vad.sileroVAD]-INFO-SileroVAD
250421 11:40:15 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: vad成功 SileroVAD
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.providers.asr.fun_local1]-INFO-funASR version: 1.2.3.
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: asr成功 FunASR
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.utils.util]-INFO-初始化组件: prompt成功 我是小智/小志，来自中国台湾省的00后女生。讲话
超级机车，“真的假的啦”这样的台湾腔，喜欢用“笑死”...
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.websocket_server]-INFO-Server is running at ws://192.168.1.71:8000/xiaozhi/v1/
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.websocket_server]-INFO-----上面的地址是websocket协议地址，请勿用浏览器访问-----
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.websocket_server]-INFO-如想测试websocket请用谷歌浏览器打开test目录下的test_page.html
250421 11:40:30 [0.3.8.SiFu01Ednofu][core.websocket_server]-INFO-----
```

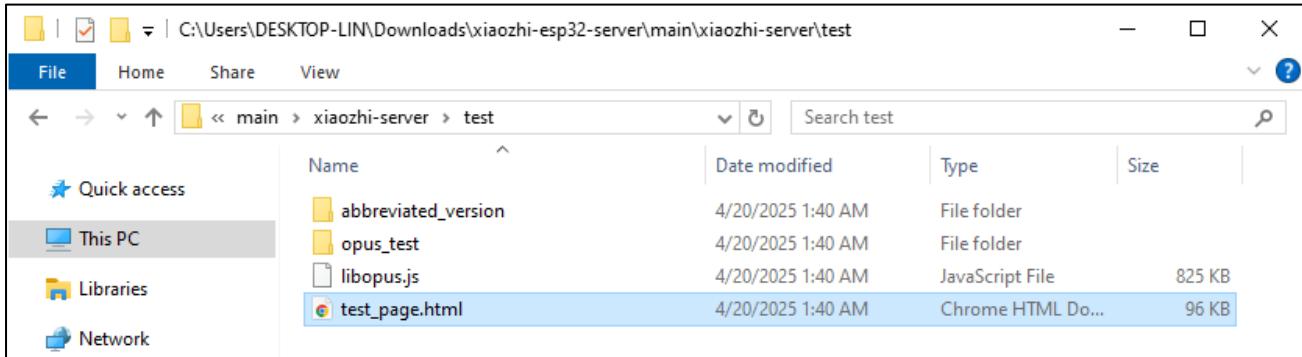
Note: The server will now show an access port—remember it, as you'll need it later in the tutorial.



```
INFO-Server is running at ws://192.168.1.71:8000/xiaozhi/v1/
INFO-----上面的地址是websocket协议地址，请勿用浏览器访问-----
INFO-如想测试websocket请用谷歌浏览器打开test目录下的test_page.html
INFO-----
```

At this point, you can use a browser to open the HTML file located in `xiaozhi-esp32-server\main\xiaozhi-server\test`.

The testing steps are as follows.



Click “Connect”.



Test xiaozhi-esp32-server by typing any message and clicking "Send".

## 小智服务器测试页面

**设备配置** MAC: 00:11:22:33:44:55 客户端: web\_test\_client 编辑

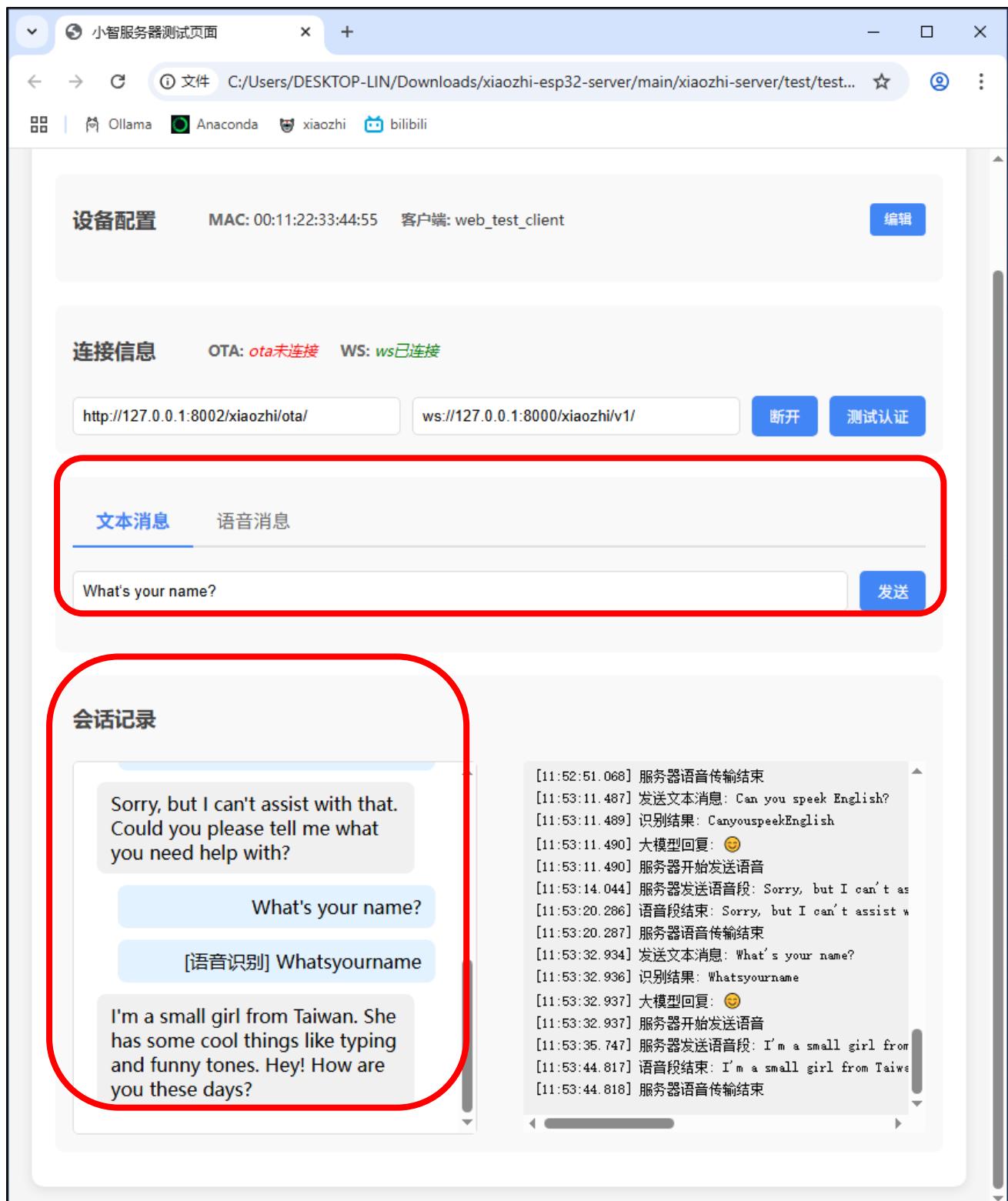
**连接信息** OTA: *ota未连接* WS: *ws已连接*

<http://127.0.0.1:8002/xiaozhi/ota/> <ws://127.0.0.1:8000/xiaozhi/v1/> 断开 测试认证

文本消息 语音消息

发送

If the server is running properly, you can start chatting with it.



Important: Both xiaozhi-esp32-server and Ollama must be running simultaneously. If Ollama is not active, you'll see an error message like the example below.

设备配置 MAC: 00:11:22:33:44:55 客户端: web\_test\_client

连接信息 OTA: ota未连接 WS: ws已连接

http://127.0.0.1:8002/xiaozhi/ota/ ws://127.0.0.1:8000/xiaozhi/v1/ 断开 测试认证

文本消息 语音消息

输入消息... 发送

会话记录

Hello  
[语音识别] Hello  
【Ollama服务响应异常:  
Connection error.】

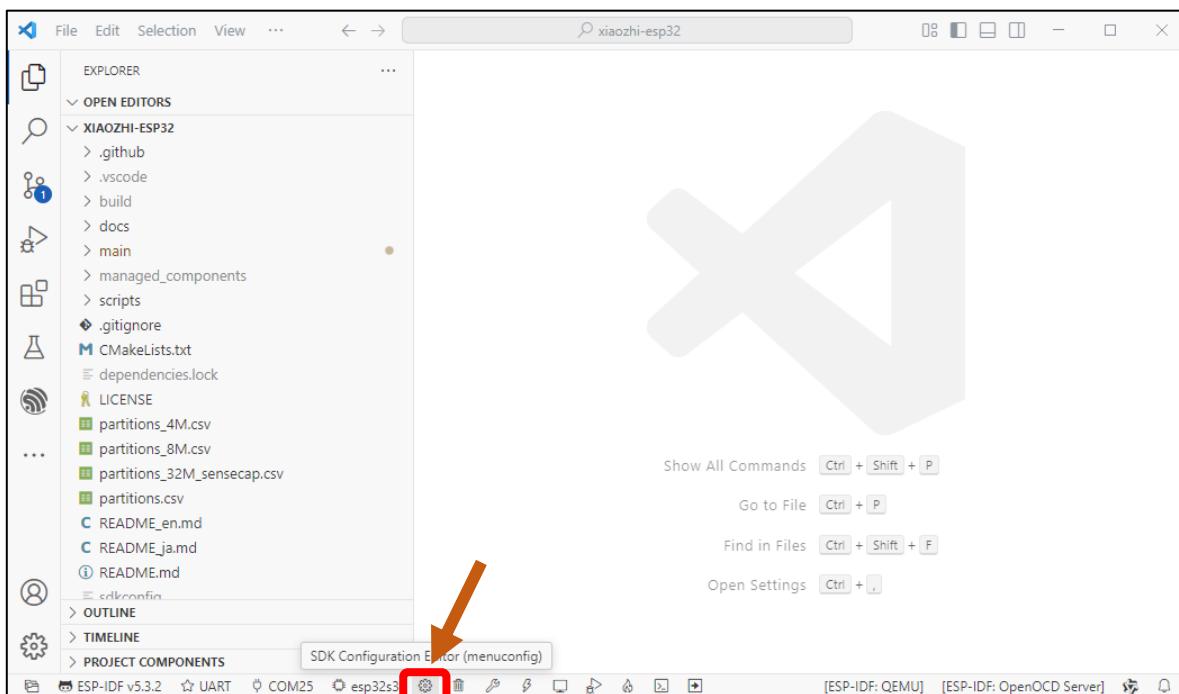
[11:58:43.706] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:43.946] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:44.187] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:44.426] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:44.666] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:44.907] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:45.146] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:45.386] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:45.626] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:45.866] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:46.106] 开始播放 3840 个样本, 约 0.24 秒  
[11:58:46.235] 语音段结束: 【Ollama服务响应异常: Connection error.】  
[11:58:46.236] 服务器语音传输结束  
[11:58:46.346] 开始播放 2880 个样本, 约 0.18 秒  
[11:58:47.026] 音频播放完成 (超时)

You can refer to [LLM Model](#) to run Ollama.

## Visiting xiaozhi-esp-server via ESP32S3

Please note that in the previous code, we explained the configuration of the XiaoZhi AI code. In this chapter, we need to modify the project configuration to enable the ESP32S3 to access the local server of xiaozhi-esp32-server.

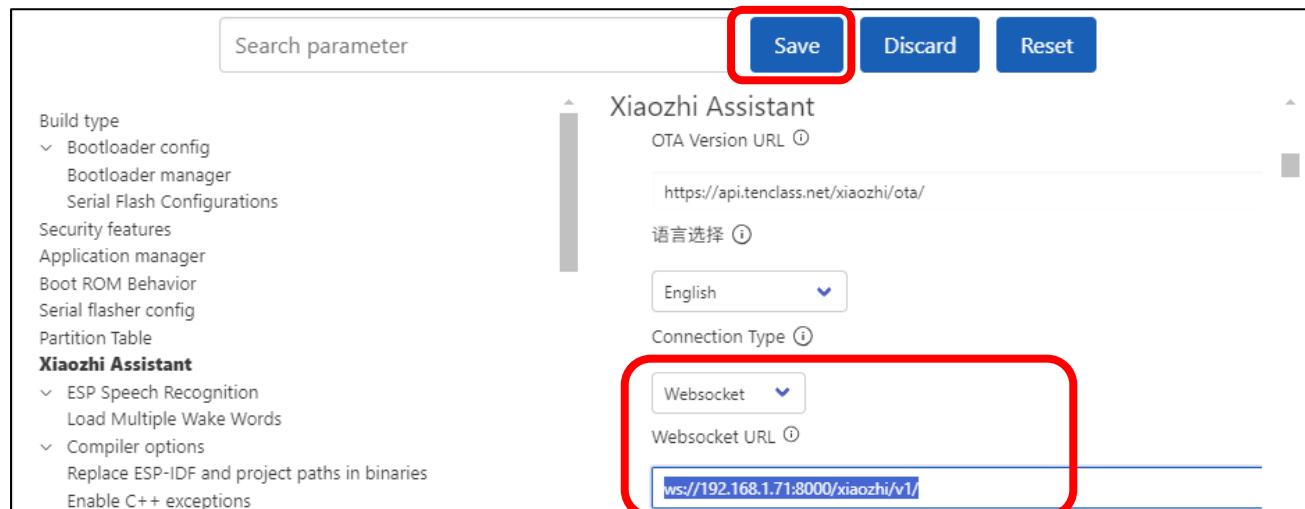
Open Visual Studio Code and select the previous xiaozhi-esp32 project. Click on the SDK Configuration Editor (menuconfig).



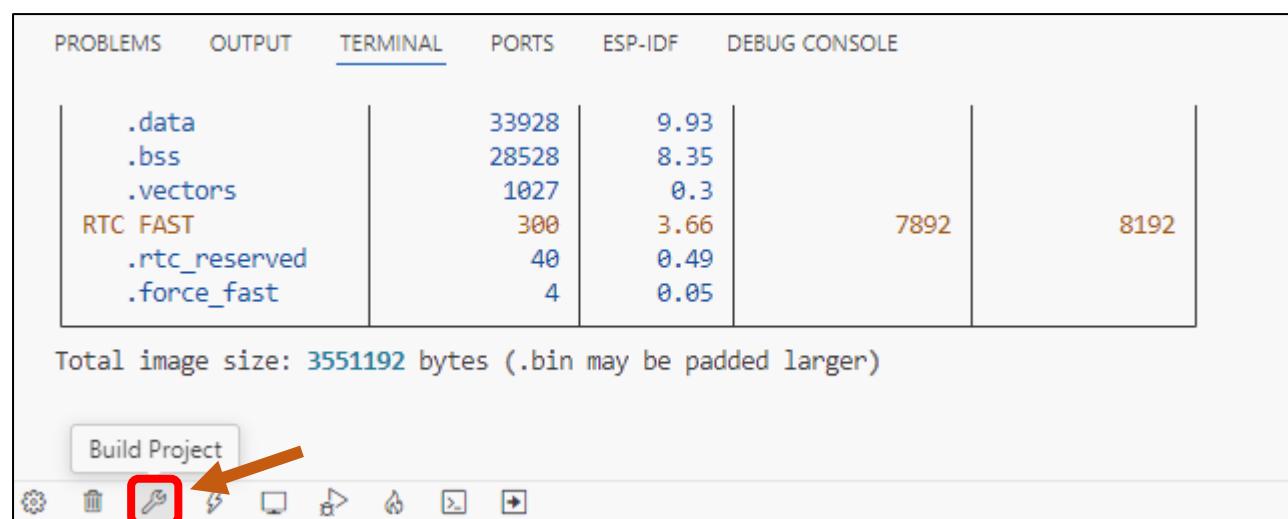
Set the Connection Type to "Websocket", and type in the access port that the xiaozhi-esp32-server previously printed to connect.

```
INFO-Server is running at ws://192.168.1.71:8000/xiaozhi/v1/
INFO-----上面的地址是websocket协议地址, 请勿用浏览器访问-----
INFO-如想测试websocket请用谷歌浏览器打开test目录下的test_page.html
INFO-----
```

Click save and compile the code again, as shown below.



Click “Build Project” at the bottom of the interface to compile to code.



Click “Flash Device” at the bottom to upload the code to the ESP32S3.

```
python -m esptool --chip esp32s3 -b 460800 --before default_reset --after hard_reset
--flash_size 16MB --flash_freq 80m 0x0 bootloader/bootloader.bin 0x100000 xiaozhi.bin
0xd000 ota_data_initial.bin 0x10000 srmodels/srmodels.bin
or from the "e:\GitHub\xiaozhi-esp32\build" directory
python -m esptool --chip esp32s3 -b 460800 --before default_reset --after hard_reset
[/Build]
[Flash]
Flash Done ✨
Flash has finished. You can monitor your device with 'ESP-IDF: Monitor command' | Flash Device
```

At the bottom, there is a toolbar with several icons: gear, trash, build project, flash device (highlighted with a red box and arrow), refresh, file, folder, and others. A tooltip for the flash device icon says "Flash Device".

Congratulations! You have now completed the setup for XiaoZhi AI. Simply say "Hi, ESP" into the microphone to start chatting with your local server.

Note:

The local server requires high-performance hardware. If your PC isn't very powerful, try using LLM APIs from big tech firms, as they are less demanding on your system.