**DEPLOY TCP SERVER**

**Note:**

* The Python version must be **3.10**, and **aiohttp** must be **3.11.13**.
* After completing the configuration, please log out of the server using the button in the sidebar.

**I. Setup server**

* **Step 1: Create a Project Folder**

First, create a folder to store your projects. In this guide, we will create a folder named "TCPServerUI" and navigate into it:

|  |
| --- |
| mkdir TCPServerUI  cd TCPServerUI |

* **Step 2: Check if Python 3.10 is installed**

Verify whether Python is installed on your system by running:

|  |
| --- |
| python3 --version |

If Python 3.10 is not installed, use the following commands to install it:

|  |
| --- |
| sudo apt update && sudo apt install python3.10 -y |

* **Step 3: Create a Virtual Environment**

Set up a virtual environment inside the project folder

|  |
| --- |
| python3 -m venv .venv |

* **Step 4: Activate the Virtual Environment**

Activate the virtual environment using the command below

|  |
| --- |
| source .venv/bin/activate |

* **Step 5: Clone the Project Repository**

Since this repository is private, you need to set up an SSH key to authenticate and clone it. Follow these steps carefully.

* + **Open a New Terminal and Generate an SSH Key.**

|  |
| --- |
| ssh-keygen -t rsa -b 4096 -C "your\_email@example.com" |

Replace "your\_email@example.com" with your GitHub email.

When prompted:

* + - Press Enter to accept the default location (~/.ssh/id\_rsa).
    - Optionally, set a passphrase for added security or press Enter to skip
  + **Copy Your Public Key.**

|  |
| --- |
| cat ~/.ssh/id\_rsa.pub |

Copy the output (it starts with ssh-rsa).

* + **Add It to GitHub.**
    - Click the **"add a new public key"** link from your screenshot.
    - In the **Title** field, enter something like "My Laptop SSH Key".
    - In the **Key** field, paste the copied SSH key.
    - Click **Add SSH Key.**

A screenshot of a computer

AI-generated content may be incorrect.

* + Clone the repository.

|  |
| --- |
| git clone [git@github.com](mailto:git@github.com):CloudBurst-Australia/iot-server-dreamsedge.git  cd iot-server-dreamsedge |

* + Checkout to branch **dev\_v1.0.3** to test.

|  |
| --- |
| git checkout dev\_v1.0.3 |

* **Step 6: Install Required Libraries**

Install the necessary dependencies listed in the **requirements.txt** file:

|  |
| --- |
| pip install --no-cache-dir -r requirements.txt |

* **Step 7: Install Redis-server libraries**

|  |
| --- |
| sudo apt update && sudo apt install redis-server -y |

**II. Using tmux to run code**

* **Step 1: Install tmux**

|  |
| --- |
| sudo apt install tmux |

* **Step 2: Creating a New tmux Session**

|  |
| --- |
| tmux new -s vitals-tcp-v1.0.3-session |

Here, **vitals-tcp-v1.0.3-session** is the name of the session. You can replace it with any name.

* **Step 3: Detaching and Reattaching to a tmux Session**

Detach from a session: Press Ctrl + B, then D

List active sessions:

|  |
| --- |
| tmux ls |

Reattach to a session:

|  |
| --- |
| tmux attach -t vitals-tcp-v1.0.3-session |

* **Step 4: Running Code in tmux**

Attach to **vitals-tcp-v1.0.3-session**

|  |
| --- |
| tmux attach -t vitals-tcp-v1.0.3-session |

With the new server, we will run Backend and Frontend separately. So we need to split the terminal to 2 windows: Press **Ctrl + B**, then **Shift + %**

A screenshot of a video

AI-generated content may be incorrect.

To transition between 2 windows: Press **Ctrl + B**, then then or **.**

To run code:

|  |
| --- |
| python3 Backend/main.py |

|  |
| --- |
| python3 Frontend/ui.py |

After that, you can detach tmux: Press Ctrl + B, then D and the script keep running in the background.

**III. Essential tmux Hotkeys**

* **Session Management**
  + tmux new -s <name> – Create a new session
  + tmux attach -t <name> – Attach to a session
  + tmux ls – List all sessions
  + tmux kill-session -t <name> – Kill a session
* **Pane Management**
  + Ctrl + b, then % – Split window vertically
  + Ctrl + b, then " – Split window horizontally
  + Ctrl + b, then Arrow Keys – Move between panes
  + Ctrl + b, then z – Toggle zoom for the current pane
  + Ctrl + b, then x – Close current pane
* **Window Management**
  + Ctrl + b, then c – Create a new window
  + Ctrl + b, then n – Switch to the next window
  + Ctrl + b, then p – Switch to the previous window
  + Ctrl + b, then w – List all windows
  + Ctrl + b, then & – Close the current window

**IV. New updated in this version**

* Initialized a Redis queue to manage the flow of retrieving or updating data in environment.py.
* Initially, it will retrieve data from environment.py, which contains the default settings. Then, the data updates from the UI to the backend will be handled by the queue. This reduces the frequency of access to environment.py, ensuring that the data in environment.py will not be lost for unknown reasons.
* The server has been tested locally for 8 hours, and no errors occurred during the testing.