**Run the server**

1. Open a **new terminal** in VS Code (**Ctrl + ~**).
2. Navigate to the project folder:

cd path/to/distributed\_computing\_system

1. Run the server:

python server.py

1. You should see:

Server listening on 127.0.0.1:65432

**Run the Client**

1. Open a **new terminal** (**do not close the server terminal**).
2. Navigate to the same project directory:

cd path/to/distributed\_computing\_system

1. Run the client:

python client.py

1. You should see something like

Task Result: 499999500000

This is the sum of numbers from 0 to 999999.

**Test Multiple Clients**

1. Open **multiple terminals**.
2. Run python client.py in each terminal.
3. The server will handle multiple tasks in parallel.

**Stopping the Server**

* **Press Ctrl + C** in the server terminal.

**Running on Different Machines**

To run on a **local network** instead of just 127.0.0.1:

1. Find your IP address:

ipconfig # On Windows

ifconfig # On macOS/Linux

1. Change HOST in server.py and client.py to your actual local IP.
2. Run the server on your main computer.
3. Run the client from a different machine in the same network.

## ****GUI Using Tkinter****

This approach uses **Tkinter** to create a **desktop application** where users can input a computation task, send it to the server, and view the result.

### **Install Required Packages**

Tkinter comes pre-installed with Python, but you need to install pillow for better UI elements:

pip install pillow

### **Run the GUI Client**

1. Start the **server** (python server.py).
2. Run the **Tkinter UI client**:

python ui\_client.py

1. Enter a **Python expression** (e.g., sum(range(1000000))) and click **Send Task**.
2. The result will appear below.

## ****Web-Based UI Using Flask and React****

This approach provides a **modern web-based UI** using **Flask for the backend** and **React for the frontend**.

**Install Flask and CORS**

pip install flask flask-cors

Run the Flask server:

python flask\_server.py