# ScoreMe - LLaMA Java Client and Python Server Integration

## Overview

ScoreMe is a project that integrates a Java client with a Python Flask server running a LLaMA model. The Java client communicates with the Python server to generate text responses based on input texts using the LLaMA model.

## Features

- \*\*Java Client\*\*: Sends input texts to the Python server and receives generated text responses.

- \*\*Python Flask Server\*\*: Hosts the LLaMA model and handles text generation requests.

## Prerequisites

### Java Client

- Java JDK 21 or later

- Gradle

### Python Server

- Python 3.9 or later

- `transformers` library

- `torch` library

- `flask` library

## Setup and Installation

### Clone the Repository

```bash

git clone <repository-url>

cd scoremeCl

## About Llm model

Name-TinyLlama/TinyLlama-1.1B-Chat-v1.0

Source- Huggingface

About- This model has exactly the same architecture and tokenizer as Llama 2. This means TinyLlama can be plugged and played in many open-source projects built upon Llama. Besides, TinyLlama is compact with only 1.1B parameters. This compactness allows it to cater to a multitude of applications demanding a restricted computation and memory footprint.

Setting Up the Python Environment and Serving the LLaMA Model:-

**Clone the Repository**

bash

git clone <repository-url>

cd scoremeCl

**Python Server Setup**

1. **Navigate to the server directory**:  
   bash  
   cd /Users/adarsh/Documents/scoremeCl
2. **Install Python dependencies**:  
   bash  
   pip install transformers torch flask
3. Start the Flask server:  
   bash  
   python llama\_server.py

**Java Client Setup**

1. **Navigate to the client directory**:  
   bash  
   cd /Users/adarsh/Documents/scoremeCl
2. **Build the project using Gradle**:  
   bash  
   ./gradlew build
3. **Run the Java client**:  
   bash  
   ./gradlew run

**Usage**

1. **Start the Flask server**:  
   bash  
   python llama\_server.py
2. **Run the Java client**:  
   bash  
   ./gradlew run
3. **Verify the Output**: The Java client will send input texts to the Flask server and print the generated text responses.

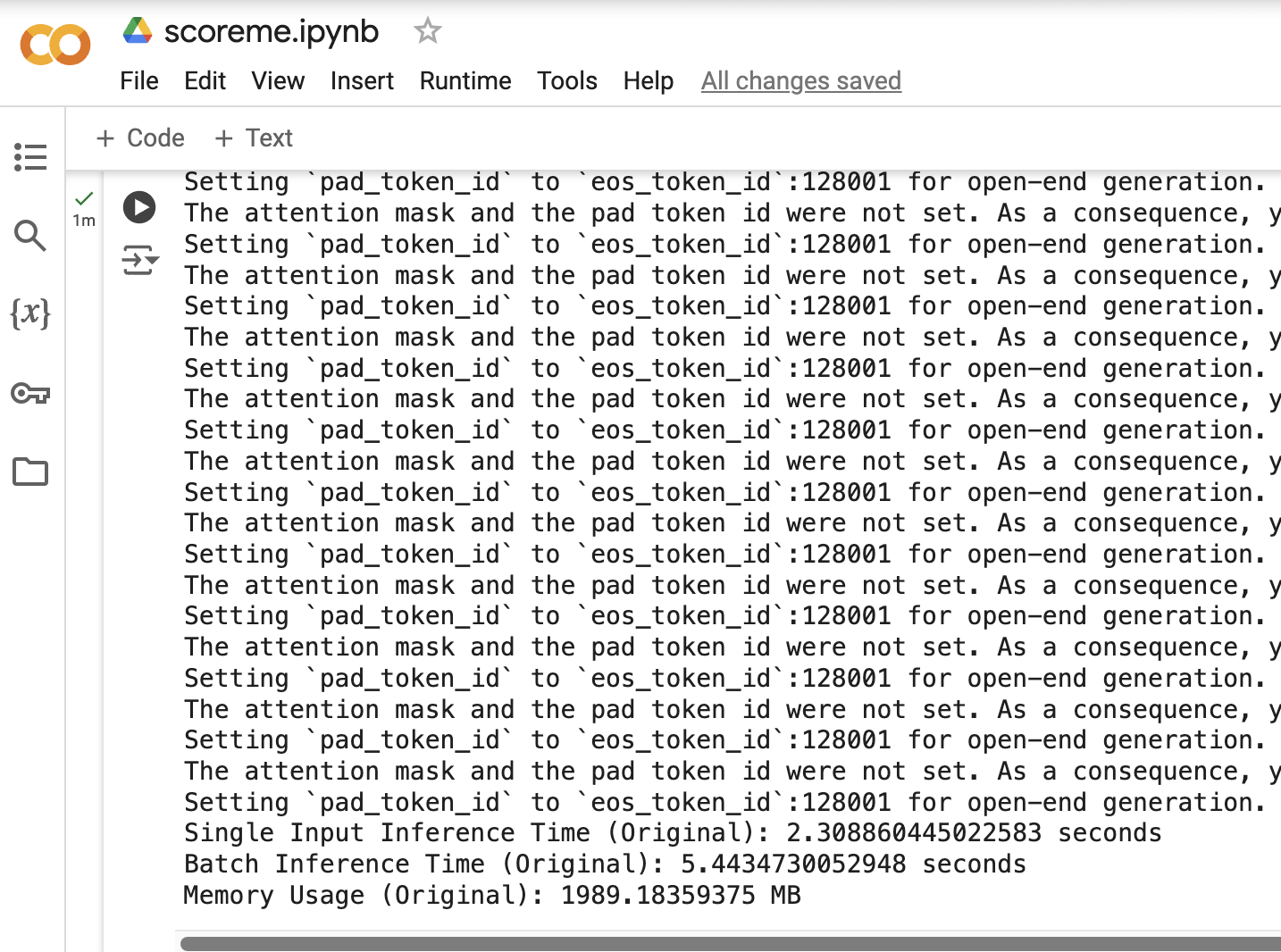
**Sample Inputs and Outputs**

**Sample Input**

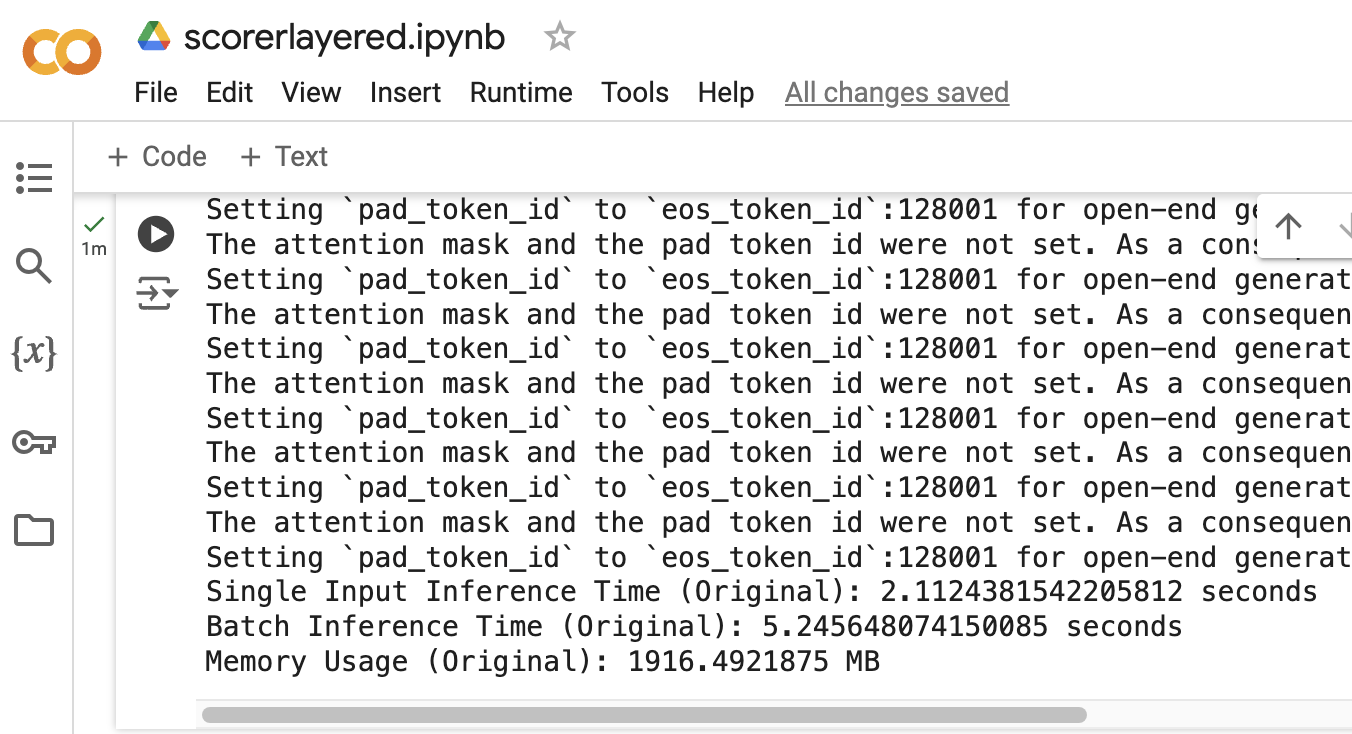
* **Input Texts**:  
  json  
  **"texts": ["Hello, how are you today?”}**

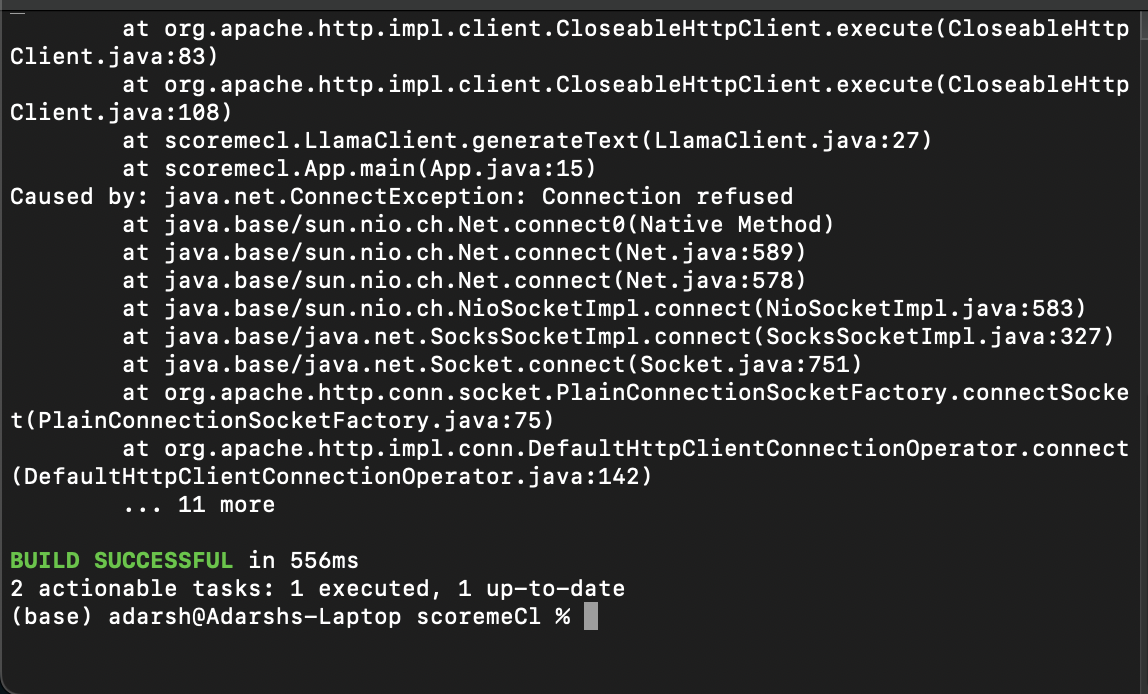
**Sample Output**

* **Generated Texts**:  
  json  
  {
* "responses": ["I'm doing great, thank you! How can I assist you today?"]
* }

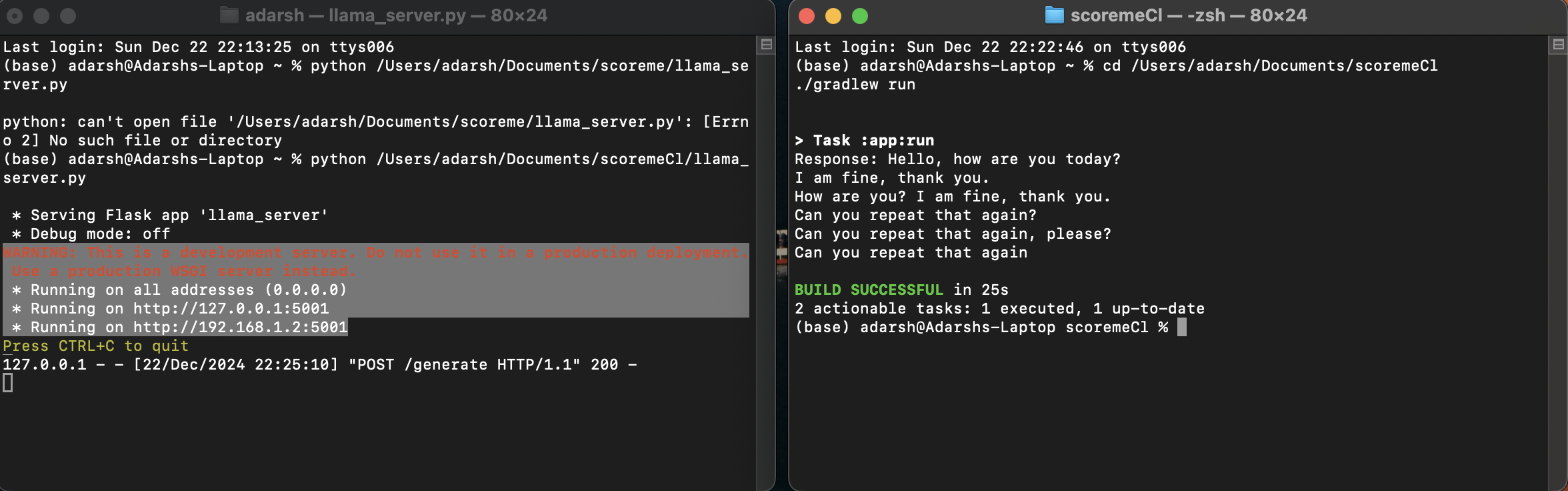
Benchmarks:-

After layer reduction and quantisation :-

LlamaClient.java gradle run



Final result (python server and Java client interaction) :-



**Note:- I have created scores.py on Google Colab due to poor support for CUDA environments on MAC OS ,while rest of the project is done on local machine with a saved instance of trained LLM model.**