Unbox Robotics

Speed Data Management App

May 16, 2024

Introduction

The Speed Data Management App is developed to manage speed data collected from various sources. It provides APIs for inserting data into the SQLite database, clearing the database, and retrieving the latest entry. It features a simple HTML page to visualize the latest speed entry.

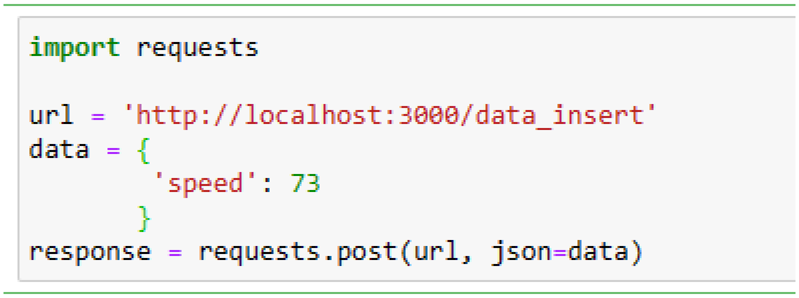
Implementation Details

Architecture Block Diagram : Block Diagram

1. Data Insertion API

The `/data_insert` API endpoint allows inserting speed data along with the timestamp into the SQLite database.

Sample API Call :



```
import requests

url = 'http://localhost:3000/data_insert'
data = {
        'speed': 73
       }
response = requests.post(url, json=data)
```

2. Data Clearing API

The /clear_data API endpoint clears all data entries from the SQLite database. Given our focus on real-time operations, if the database size becomes too large, we have the option to clear the data entirely. Alternatively, we can adjust the configuration to retain only a specified number of rows instead of clearing the entire table.

Sample API Call :

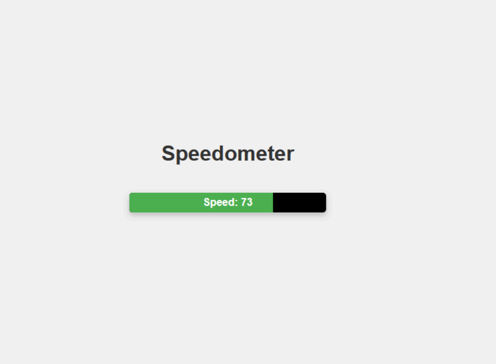
resp = requests.get('http://localhost:3000/clear_data')

3. Latest Entry Retrieval API

The `/latest_entry` API endpoint retrieves the latest entry (timestamp and speed) from the SQLite database in JSON format.

4. Frontend Integration

The `‘/’` endpoint renders a simple HTML template (index.html) to display the latest speed entry. Index.html utilizes the ‘`/latest_entry`’ API to fetch data and display it, refreshing every 1 second. While web sockets could be an alternative, HTTP suffices in this scenario as data arrives every second, minimizing resource wastage.



Speedometer

Speed: 73

.

Conclusion

The Flask Speed Data Management App provides a robust solution for managing speed data. It effectively addresses the requirements outlined in the assignment and offers scalability for future enhancements.