Simple Real-Time Database

Modul ini berfokus pada mengirim data **CPU usage** dari komputer ke **Firebase**, kemudian menampilkan data tersebut secara real-time di website menggunakan **Flask**. Aplikasi ini akan di-deploy ke **Vercel**. Anda dapat memodifikasinya untuk mengirim dan menampilkan data lainnya secara real-time.

Modul Hands-On: Visualisasi Data CPU Usage dengan Flask, Firebase, dan Deploy ke Vercel

Prasyarat

1. **Python 3.x** terinstal di komputer. Belum menginstall python dan conda environment? Baca tutorial berikut.
2. **npm** dan **Vercel CLI** terinstal. Cara install npm dan Vercel CLI.
3. **Firebase account** untuk mengakses Firebase Realtime Database.
4. Aplikasi **Flask** akan digunakan untuk menampilkan data di web.
5. **psutil** akan digunakan untuk mengambil data CPU usage.

Langkah 1: Membuat Proyek Firebase

1. **Buat Firebase Project**:
   - Buka Firebase Console dan buat project baru.
   - Setelah project dibuat, buka bagian **Realtime Database** dan klik **Create Database**.
2. **Atur Firebase Realtime Database**:
   - Atur database dalam mode **Test Mode** agar bisa membaca dan menulis tanpa autentikasi selama pengembangan.
3. **Download Firebase Admin SDK**:
   - Buka **Project Settings** di Firebase Console, lalu masuk ke tab **Service Accounts**.
   - Klik **Generate New Private Key** untuk mendapatkan file JSON yang akan digunakan di Python.

Langkah 2: Menyiapkan Lingkungan Python

Anda dapat memilih salah satu dari dua metode untuk membuat dan mengaktifkan virtual environment: menggunakan **Python venv** atau **conda**.

Opsi 1: Menggunakan Python Virtual Environment (venv)

1. **Membuat Virtual Environment**:

Di terminal, buat virtual environment menggunakan `venv`:

```
python -m venv env
```

2. **Mengaktifkan Virtual Environment**:
    - Di Windows:

```
.\env\Scripts\activate
```

- Di macOS/Linux:

```
source env/bin/activate
```

3. **Install Dependency Python**:

Buat file `requirements.txt` dengan isi berikut:

```
Flask==2.0.1
Werkzeug==2.0.1
firebase-admin==5.0.0
psutil
```

Install dependencies yang diperlukan:

```
pip install -r requirements.txt
```

Opsi 2: Menggunakan Conda Virtual Environment (Opsi yang disarankan)

1. **Membuat Conda Environment**:

Jika Anda menggunakan **Conda**, Anda dapat membuat environment baru dengan perintah berikut:

```
conda create --name flask_firebase_env python=3.9
```

2. **Mengaktifkan Conda Environment**:

Aktifkan environment yang baru saja Anda buat:

```
conda activate flask_firebase_env
```

3. **Install Dependency Python**:

Setelah environment aktif, buat file `requirements.txt` dengan isi berikut (jika belum ada):

```
Flask==2.0.1
Werkzeug==2.0.1
firebase-admin==5.0.0
psutil
```

Install dependencies menggunakan `pip` (karena Conda dapat menggunakan pip untuk manajemen paket):

```
pip install -r requirements.txt
```

Dengan kedua opsi ini, Anda dapat menggunakan salah satu metode yang paling sesuai dengan preferensi Anda untuk mengelola virtual environment, baik dengan **venv** atau **Conda**. Setelah environment siap, lanjutkan ke **Langkah 3** untuk membuat script yang mengirim data CPU ke Firebase.

1. **Membuat dan Mengaktifkan Python Virtual Environment**:

Di terminal, buat virtual environment untuk proyek ini:

```
python -m venv env
```

Aktifkan virtual environment: - Di Windows:

```
.\env\Scripts\activate
```

- Di macOS/Linux:

```
source env/bin/activate
```

2. **Install Dependency Python**: Buat file `requirements.txt` dengan isi berikut:

```
Flask==2.0.1
Werkzeug==2.0.1
firebase-admin==5.0.0
psutil
```

Install dependencies dengan perintah berikut:

```
pip install -r requirements.txt
```

Langkah 3: Script Python untuk Mengirim Data CPU ke Firebase

Buat file Python bernama `send_cpu_data.py` untuk mengirim data CPU usage ke Firebase secara real-time.

```
import firebase_admin
from firebase_admin import credentials, db
import psutil
import time

# Inisialisasi Firebase Admin SDK
cred = credentials.Certificate("path/to/your-serviceAccountKey.json")
firebase_admin.initialize_app(cred, {
    'databaseURL': 'https://your-project-id.firebaseio.com/'
})

# Referensi ke node 'cpu_usage' di Firebase
ref = db.reference('cpu_usage')

# Fungsi untuk mengirim data CPU usage ke Firebase
def push_cpu_data():
    while True:
        cpu_percent = psutil.cpu_percent(interval=1)  # Mengambil CPU usage

        data = {
```

```
            'cpu': cpu_percent,
            'timestamp': time.time()  # Menyimpan timestamp
        }

        ref.push(data)  # Mengirim data ke Firebase

        print(f"Data sent to Firebase: {data}")
        time.sleep(5)  # Mengirim data setiap 5 detik

if __name__ == "__main__":
    push_cpu_data()
```

Jalankan script ini:

```
python send_cpu_data.py
```

Script ini akan mengirim data CPU usage ke Firebase setiap 5 detik. Pastikan `path/to/your-serviceAccountKe` ubah sesuai lokasi file JSON Firebase Anda.

Langkah 4: Membuat Aplikasi Flask untuk Visualisasi Data

1. **Membuat Flask App**: Buat file Python bernama `app.py` yang akan digunakan sebagai aplikasi web dengan Flask.

```
from flask import Flask, render_template, jsonify
import firebase_admin
from firebase_admin import credentials, db
import os

app = Flask(__name__)

# Inisialisasi Firebase Admin SDK
cred = credentials.Certificate("path/to/your-serviceAccountKey.json")
firebase_admin.initialize_app(cred, {
    'databaseURL': 'https://your-project-id.firebaseio.com/'
})

# Referensi ke node 'cpu_usage' di Firebase
ref = db.reference('cpu_usage')
```

```python
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/data')
def get_data():
    snapshot = ref.order_by_key().limit_to_last(10).get()  # Ambil 10 data terakhir
    data = [{'cpu': value['cpu'], 'timestamp': value['timestamp']} for key, value in snaps
    return jsonify(data)

# Tidak perlu app.run() karena Vercel yang akan mengurus servernya
```

2. **Membuat Template HTML**:

Buat folder bernama `templates`, lalu di dalam folder tersebut buat file `index.html`.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Real-Time CPU Usage</title>
    <script src="https://cdn.jsdelivr.net/npm/chart.js"></script>
</head>
<body>
    <h1>Real-Time CPU Usage</h1>
    <canvas id="cpuChart" width="400" height="200"></canvas>

    <script>
        var ctx = document.getElementById('cpuChart').getContext('2d');
        var cpuChart = new Chart(ctx, {
            type: 'line',
            data: {
                labels: [],
                datasets: [{
                    label: 'CPU Usage (%)',
                    data: [],
                    borderColor: 'rgba(75, 192, 192, 1)',
                    borderWidth: 2,
                    fill: false
                }]
```

```
            },
            options: {
                scales: {
                    y: {
                        beginAtZero: true,
                        max: 100
                    }
                }
            }
        });

        function updateChart() {
            fetch('/data')
                .then(response => response.json())
                .then(data => {
                    const timestamps = data.map(item => new Date(item.timestamp * 1000).to
                    const cpuUsages = data.map(item => item.cpu);

                    cpuChart.data.labels = timestamps;
                    cpuChart.data.datasets[0].data = cpuUsages;

                    cpuChart.update();
                });
        }

        setInterval(updateChart, 5000);  // Update setiap 5 detik
    </script>
</body>
</html>
```

Langkah 5: Menyiapkan Vercel untuk Deploy

1. **Install Vercel CLI**: Instal **Vercel CLI** dengan npm:

```
npm install -g vercel
```

2. **Membuat Konfigurasi Vercel**:

Buat file `vercel.json` di root folder proyek Anda:

```
{
  "version": 2,
  "builds": [
    {
      "src": "app.py",
      "use": "@vercel/python"
    }
  ],
  "routes": [
    {
      "src": "/(.*)",
      "dest": "app.py"
    }
  ]
}
```

3. **Ignore File `send_cpu_data.py`**:

Buat file `.vercelignore` di root folder proyek Anda untuk mengecualikan `send_cpu_data.py` dari deployment:

```
send_cpu_data.py
```

Langkah 6: Deploy ke Vercel

1. Di terminal, jalankan perintah berikut untuk deploy aplikasi Flask ke Vercel:

```
vercel
```

2. Ikuti instruksi Vercel CLI, pilih opsi untuk membuat proyek baru dan jalankan deploy.
3. Setelah deployment selesai, Anda akan mendapatkan URL live dari Vercel untuk mengakses aplikasi Flask Anda.

Langkah 7: Menjalankan Aplikasi Secara Lokal

Jika Anda ingin menjalankan aplikasi Flask secara lokal, Anda bisa jalankan perintah berikut:

```
python app.py
```

Buka browser dan kunjungi `http://127.0.0.1:5000/` untuk melihat visualisasi data CPU usage.

Kesimpulan

Anda telah berhasil membuat aplikasi yang mengambil data CPU usage dari komputer Anda, mengirimnya ke Firebase, dan menampilkannya di website yang di-deploy ke Vercel menggunakan Flask. Aplikasi ini memanfaatkan teknologi cloud untuk real-time data visualization.

Tugas `ho1`

Deskripsi Tugas

Cobalah kembali modul diatas di `firebase` dan `laptop` anda masing-masing. Dokumentasikan setiap proses yang anda lakukan. Kemudian, modifikasilah data yang dikirimkan. Saya telah mencontohkan untuk mengirim data `CPU` dan `Memory % Usage`. Anda dapat menambahkan data lainnya seperti `Disk Usage`, `Network Usage`, `GPU Usage`, dan lain-lain, atau bahkan mengirimkan data dari sensor (jika anda punya).

Ketentuan

- Lakukan deployment di `vercel`
- Dokumentasikan setiap proses yang anda lakukan
- Buatlah laporan dengan menggunakan `LaTeX` menggunakan overleaf dan template IF ITERA
- Sertakan screenshot dari setiap proses
- Sertakan referensi (jika ada) dan attachment percakapan dengan AI LLM (apabila ada) pada bagian akhir dari laporan.

Pengumpulan

- Deadline: 4 Oktober 2024 | Pukul 23.59 WIB > Pengurangan nilai 10% untuk setiap jam keterlambatan
- Link Pengumpulan: Google Form