1 Python程序设计#2作业

班级: 305

学号: 2022211683

姓名: 张晨阳

1.1 作业题目

数据文件（graph.txt）是一个全球温度年度异常历史数据。基于Sanic实现一个查询服务，服务包括：

- 按起始和结束年份查询历史数据，查询结果支持多种格式：JSON、XML、CSV（用逗号作为间隔符）。
- 按温度高低进行排序，支持升序和降序两种排序方式。

1.2 作业内容

程序源代码嵌入下方的code block中。

```
from sanic import Sanic, response
from sanic.response import json, html
import csv
import xmltodict

app = Sanic("TemperatureHistory")

def load_data(file_path="E:/WILLIAMZHANG/Python程
设/Homeworks/homework2/graph.txt"):
    data = []
    with open(file_path, "r") as file:
        for line in file:
            if line.strip() and not line.startswith("#"):
                parts = line.split()
                year = int(parts[0])
                no_smoothing = float(parts[1])
                lowess = float(parts[2])
                data.append(
                    {"Year": year, "No_Smoothing": no_smoothing, "Lowess":
lowess})
    return data


temperature_data = load_data()


def query_data(start_year, end_year, order=None):
    filtered_data = [
        item for item in temperature_data if start_year <= item["Year"] <=
end_year]
    if order == "asc":
```

```
        filtered_data.sort(key=lambda x: x["No_Smoothing"])
    elif order == "desc":
        filtered_data.sort(key=lambda x: x["No_Smoothing"], reverse=True)
    return filtered_data

# 主查询页面，带表单
@app.route("/")
async def index(request):
    html_content = """
    <html>
    <body>
        <h2>Temperature Data Query</h2>
        <form action="/query" method="POST">
            <label for="start_year">Start Year:</label>
            <input type="number" id="start_year" name="start_year" value="1880">
<br><br>
            <label for="end_year">End Year:</label>
            <input type="number" id="end_year" name="end_year" value="2022"><br>
<br>
            <label for="order">Order:</label>
            <select id="order" name="order">
                <option value="asc">Ascending</option>
                <option value="desc">Descending</option>
            </select><br><br>
            <label for="format">Format:</label>
            <select id="format" name="format">
                <option value="json">JSON</option>
                <option value="xml">XML</option>
                <option value="csv">CSV</option>
            </select><br><br>
            <input type="submit" value="Submit">
        </form>
    </body>
    </html>
    """
    return html(html_content)

# 处理查询并返回结果
@app.route("/query", methods=["POST"])
async def query(request):
    start_year = int(request.form.get("start_year", 1880))
    end_year = int(request.form.get("end_year", 2022))
    order = request.form.get("order", None)
    data_format = request.form.get("format", "json")

    data = query_data(start_year, end_year, order)

    # 根据用户选择的格式返回数据
    if data_format == "json":
        return json(data)
    elif data_format == "xml":
        xml_data = xmltodict.unparse({"data": {"entry": data}})
        return response.text(xml_data, content_type="application/xml")
    elif data_format == "csv":
```

```
        csv_data = "Year,No_Smoothing,Lowess\n"
        csv_data += "\n".join([f"{item['Year']},{item['No_Smoothing']
                                                  },{item['Lowess']}" for item in
data])
        return response.text(csv_data, content_type="text/csv")
    else:
        return response.text("Unsupported format", status=400)

if __name__ == '__main__':
    app.run(host="0.0.0.0", port=8000)
```

1.3 代码说明

代码结构及说明

1. **依赖库导入**：代码使用了 `Sanic` 框架来创建 Web 服务器，并导入了 `xmltodict` 以支持 XML 格式输出，csv 库用于处理 CSV 数据。

2. **数据加载函数** - `load_data`：

- **功能**：读取 `graph.txt` 文件，过滤注释行，将每行数据解析成字典，包含 `Year`、`No_Smoothing` 和 `Lowess` 三个字段。
- **实现**：函数逐行读取文件，将每行解析为字典并添加到 `data` 列表中。
- **调用**：在程序启动时调用一次，数据保存在 `temperature_data` 中，以便后续查询直接使用。

3. **数据查询函数** - `query_data`：

- **功能**：根据指定的年份范围和排序方式返回过滤后的温度数据。
- **参数**：
  - `start_year`：查询的起始年份。
  - `end_year`：查询的结束年份。
  - `order`：排序方式，可为 "`asc`"（升序）或 "`desc`"（降序）。
- **实现**：过滤年份范围内的数据，按 `No_Smoothing` 列的值升序或降序排列。
- **返回值**：返回包含筛选和排序后的数据列表。

4. **主页路由** - `index`：

- **路径**：/
- **功能**：提供一个 HTML 表单页面，让用户输入查询参数（起始年份、结束年份、排序方式和数据格式）。
- **表单字段**：
  - `start_year`：起始年份。
  - `end_year`：结束年份。
  - `order`：选择升序或降序。
  - `format`：选择返回数据的格式（JSON、XML 或 CSV）。 **返回**：返回一个 HTML 页面，包含可提交查询请求的表单。

5. **查询处理路由** - `query`：

- **路径**：`/query`

- **方法**：POST
- **功能**：处理来自表单的查询请求，根据用户输入的年份范围、排序方式和输出格式来返回相应的数据。
- **过程**：
    1. 获取用户输入的 `start_year`、`end_year`、`order`、`format`。
    2. 调用 `query_data` 函数，传入用户参数，获取过滤和排序后的数据。
    3. 根据 `format` 值选择返回格式：
        - **JSON**：使用 `Sanic` 内置的 `json` 方法返回 JSON 数据。
        - **XML**：使用 `xmltodict.unparse` 将数据转换为 XML 字符串，并以 XML 格式返回。
        - **CSV**：生成 CSV 格式的字符串并返回。
    4. 如果格式不支持，返回状态码 `400` 和错误信息。

6. **程序启动**：

- 在 `__main__` 块中启动 `Sanic` 应用，监听 `0.0.0.0`，端口 `8000`。

使用方式说明

1. 启动代码后访问 `http://localhost:8000/`，加载主页表单。
2. 用户填写查询条件后提交表单，服务根据条件返回所需格式的数据。