Campus Energy Meter Data Processing and Analysis

SUBJECT: *PROGRAMMING FOR PROBLEM SOLVING USING PYTHON*

SUBMITTED TO: *DR. SAMEER FAROOQ*

COURSE CODE: *ETCCPP102*

MADE BY: *SONAM YADAV*

CLASS: *B.TECH CSE (AI ML)*

SECTION: *B*

ROLL NO: *2501730260*

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my project guide, **Dr. Sameer Farooq**, for his continuous guidance, motivation, and support throughout the completion of this project. His valuable feedback and insightful suggestions consistently helped refine my work at every stage.

I am equally thankful to the **Head of Department** and all the respected faculty members for providing the necessary facilities, resources, and a supportive environment that made this project possible. Their cooperation and encouragement have been truly appreciated.

Lastly, I am grateful to my friends and family for their constant support. Their patience, assistance, and moral encouragement helped me remain focused, especially during the more challenging phases of this project.

TABLE OF CONTENT

INTRODUCTION

Energy consumption across educational campuses has increased significantly due to the expansion of academic blocks, hostels, computer centers, laboratories, and other facilities. Without proper monitoring systems, substantial amounts of electricity often go unnoticed, leading to increased operational costs and negative environmental impacts. Therefore, analyzing electricity usage data is essential for identifying trends, peak loads, and high-consumption areas to support effective energy-saving strategies.

This project presents a Python-based data analysis pipeline designed to examine campus electricity consumption using meter-reading data stored in CSV format. The implementation utilizes **Pandas** for data management, **Matplotlib** for visualization, and **object-oriented programming** to create structured models for buildings and meter readings. The system automatically reads multiple CSV files, cleans and merges them, performs daily and weekly aggregations, generates building-wise statistical summaries, produces visual graphs, and exports both cleaned datasets and summary reports.

Overall, this project strengthened practical skills in handling real-world time-series data, performing resampling operations, working with timestamps, and deriving meaningful insights from raw energy readings. It also demonstrates how Python can be effectively used in energy management and smart-campus applications.

PROBLEM STATEMENT & OBJECTIVES

PROBLEM STATEMENT

Campus authorities generally receive monthly electricity bills but lack a detailed breakdown of building-wise consumption or day-to-day usage patterns. Manually analyzing multiple large CSV meter files is inefficient, prone to errors, and time-consuming. Hence, there is a need for an automated system that can import, clean, summarize, and visually present energy consumption data in a user-friendly format.

OBJECTIVES

The major objectives of this project are:

To develop a Python script that automatically reads all meter-reading CSV files from a designated *data* folder.

To clean and validate timestamp data and remove any invalid rows.

To compute daily and weekly electricity consumption using Pandas resampling features.

To generate building-wise statistical summaries including mean, minimum, maximum, and total kWh.

To apply object-oriented programming principles to organize meter readings and building data.

To produce a visualization dashboard incorporating daily trend lines, weekly bar charts, and scatter plots.

To export cleaned datasets and summary information to CSV and text files for further use.

Each objective is implemented systematically within the project and is documented in the subsequent sections.

Python Lab 5 > data > admin_block_jan.csv > data

```
timestamp,kwh,building
2024-01-01,84.34,Admin_Block
2024-01-02,400.96,Admin_Block
2024-01-03,247.28,Admin_Block
2024-01-04,375.56,Admin_Block
2024-01-05,490.1,Admin_Block
2024-01-06,292.32,Admin_Block
2024-01-07,275.5,Admin_Block
2024-01-08,82.42,Admin_Block
2024-01-09,170.8,Admin_Block
2024-01-10,274.95,Admin_Block
2024-01-11,355.65,Admin_Block
2024-01-12,411.68,Admin_Block
2024-01-13,221.42,Admin_Block
2024-01-14,79.67,Admin_Block
2024-01-15,179.67,Admin_Block
2024-01-16,459.32,Admin_Block
2024-01-17,146.02,Admin_Block
2024-01-18,253.46,Admin_Block
2024-01-19,469.04,Admin_Block
2024-01-20,61.2,Admin_Block
2024-01-21,320.25,Admin_Block
2024-01-22,477.56,Admin_Block
2024-01-23,153.64,Admin_Block
2024-01-24,296.82,Admin_Block
2024-01-25,459.11,Admin_Block
2024-01-26,109.93,Admin_Block
2024-01-27,285.54,Admin_Block
2024-01-28,387.68,Admin_Block
2024-01-29,351.06,Admin_Block
2024-01-30,260.49,Admin_Block
2024-01-31,142.18,Admin_Block
```

METHODOLOGY AND SYSTEM DESIGN

Overall Workflow

The primary workflow of the program is as follows:

Read all CSV files from the data folder.

Add a *source_file* column to each DataFrame and merge them into one dataset.

Convert the timestamp column to a valid datetime format and discard invalid entries.

Compute daily and weekly consumption totals, along with building-wise statistics.

Utilize OOP classes (MeterReading, Building, BuildingManager) to organize data logically.

Generate various visualizations using Matplotlib.

Export cleaned data and summary reports to the output folder.

This workflow ensures a structured and complete approach—from data ingestion to final reporting.

Data Flow

Input:

Multiple CSV files → Combined DataFrame (df)

Processing:

Cleaning and validating timestamps

Resampling for daily and weekly totals

Grouping for building-wise statistics

Creating OOP objects and computing totals

Output:

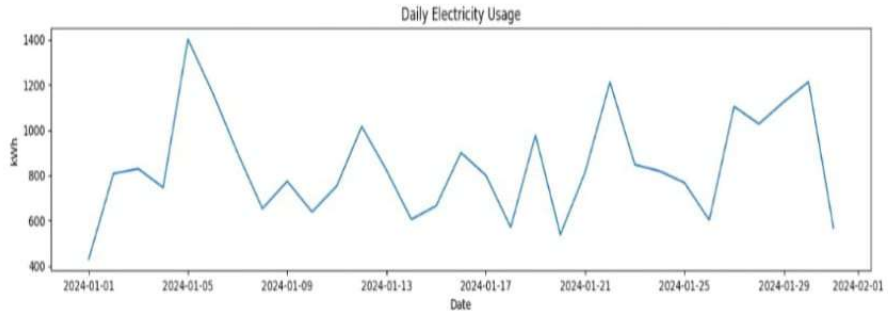
dashboard.png

cleaned_energy_data.csv

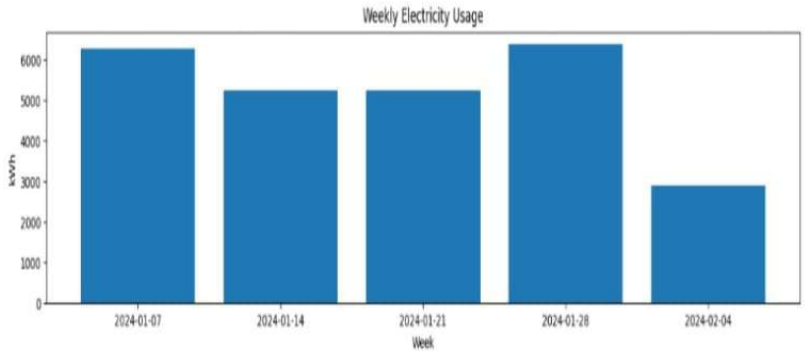
building_summary.csv

summary.txt

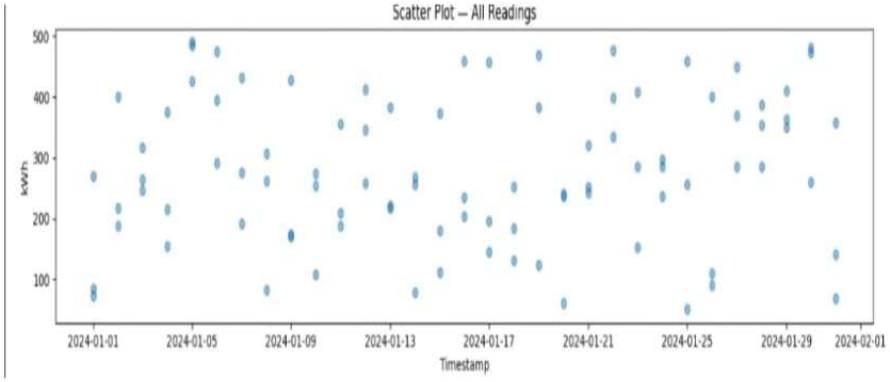
Each output file fulfills a specific purpose: dashboards enable visual interpretation, CSV exports support external tools, and summary text files provide a quick report overview.



❖ **Weekly Electricity Usage**



❖ **Scatter Plot (All Readings)**



IMPLEMENTATION DETAILS

The script begins by importing the necessary libraries: `pandas`, `matplotlib.pyplot`, `os`, and `Path` from `pathlib`. The working directory is printed, and an *output* folder is created to avoid export errors.

Task 1: Data Ingestion and Validation

The data folder is defined as `data_folder = "data"`

All CSV files are retrieved using `Path(data_folder).glob("*.csv")`.

Each file is read using `pd.read_csv()`, after which a `source_file` column is added

If an error occurs while reading any file, it is captured without halting the entire pipeline.

All DataFrames are combined using `pd.concat()`.

Timestamps are converted using `pd.to_datetime(..., errors="coerce")`, and invalid entries (NaT) are removed

`df.head()` is printed to confirm that data ingestion and cleaning were successful.

Task 2: Core Aggregation Logic

Three primary functions handle the aggregation:

`calculate_daily_totals(data)`
Uses `resample("D")` to compute daily kWh totals.

`calculate_weekly_totals(data)`
Uses `resample("W")` for weekly aggregates.

`building_wise_summary(data)`
Uses `groupby("building")` with `agg(["mean", "min", "max", "sum"])` to generate building-level statistics.

The outputs are stored as:

`daily_totals`

`weekly_totals`

`build_summary`

Task 3: Object-Oriented Modelling

MeterReading: Stores timestamp and kWh reading.

Building: Contains a list of readings and computes total consumption.

BuildingManager: Manages multiple buildings and generates consolidated reports.

The script iterates through `df.iterrows()` to populate the objects and then prints building-wise summaries.

RESULTS AND SCREENSHOTS

Python Lab 5 > data > hostel_jan.csv > data

```
timestamp,kwh,building
2024-01-01,73.91,Hostel
2024-01-02,188.98,Hostel
2024-01-03,316.67,Hostel
2024-01-04,155.8,Hostel
2024-01-05,484.24,Hostel
2024-01-06,475.27,Hostel
2024-01-07,431.78,Hostel
2024-01-08,262.55,Hostel
2024-01-09,428.66,Hostel
2024-01-10,109.0,Hostel
2024-01-11,188.93,Hostel
2024-01-12,258.35,Hostel
2024-01-13,383.83,Hostel
2024-01-14,268.62,Hostel
2024-01-15,111.59,Hostel
2024-01-16,204.59,Hostel
2024-01-17,195.99,Hostel
2024-01-18,185.19,Hostel
2024-01-19,124.48,Hostel
2024-01-20,236.71,Hostel
2024-01-21,251.65,Hostel
2024-01-22,398.71,Hostel
2024-01-23,408.38,Hostel
2024-01-24,285.08,Hostel
2024-01-25,257.28,Hostel
2024-01-26,400.2,Hostel
2024-01-27,449.28,Hostel
2024-01-28,353.71,Hostel
2024-01-29,410.22,Hostel
2024-01-30,472.6,Hostel
2024-01-31,68.3,Hostel
```

- Library

Python Lab 5 > data > library_jan.csv > data

```
timestamp,kwh,building
2024-01-01,270.84,Library
2024-01-02,217.57,Library
2024-01-03,264.83,Library
2024-01-04,214.65,Library
2024-01-05,427.06,Library
2024-01-06,395.89,Library
2024-01-07,191.3,Library
2024-01-08,307.68,Library
2024-01-09,174.22,Library
2024-01-10,253.78,Library
2024-01-11,208.84,Library
2024-01-12,345.83,Library
2024-01-13,216.66,Library
2024-01-14,256.59,Library
2024-01-15,373.7,Library
2024-01-16,235.85,Library
2024-01-17,457.89,Library
2024-01-18,131.2,Library
2024-01-19,383.5,Library
2024-01-20,240.07,Library
2024-01-21,241.9,Library
2024-01-22,335.47,Library
2024-01-23,285.31,Library
2024-01-24,236.7,Library
2024-01-25,50.64,Library
2024-01-26,91.52,Library
2024-01-27,369.23,Library
2024-01-28,285.96,Library
2024-01-29,363.27,Library
2024-01-30,479.96,Library
2024-01-31,357.31,Library
```

❖ **Cleaned energy data**

Python Lab 5 > output > cleaned_energy_data.csv > data

```
timestamp,kwh,building,source_file
2024-01-01,84.34,Admin_Block,admin_block_jan.csv
2024-01-02,400.96,Admin_Block,admin_block_jan.csv
2024-01-03,247.28,Admin_Block,admin_block_jan.csv
2024-01-04,375.56,Admin_Block,admin_block_jan.csv
2024-01-05,490.1,Admin_Block,admin_block_jan.csv
2024-01-06,292.32,Admin_Block,admin_block_jan.csv
2024-01-07,275.5,Admin_Block,admin_block_jan.csv
2024-01-08,82.42,Admin_Block,admin_block_jan.csv
2024-01-09,170.8,Admin_Block,admin_block_jan.csv
2024-01-10,274.95,Admin_Block,admin_block_jan.csv
2024-01-11,355.65,Admin_Block,admin_block_jan.csv
2024-01-12,411.68,Admin_Block,admin_block_jan.csv
2024-01-13,221.42,Admin_Block,admin_block_jan.csv
2024-01-14,79.67,Admin_Block,admin_block_jan.csv
2024-01-15,179.67,Admin_Block,admin_block_jan.csv
2024-01-16,459.32,Admin_Block,admin_block_jan.csv
2024-01-17,146.02,Admin_Block,admin_block_jan.csv
2024-01-18,253.46,Admin_Block,admin_block_jan.csv
2024-01-19,469.04,Admin_Block,admin_block_jan.csv
2024-01-20,61.2,Admin_Block,admin_block_jan.csv
2024-01-21,320.25,Admin_Block,admin_block_jan.csv
2024-01-22,477.56,Admin_Block,admin_block_jan.csv
2024-01-23,153.64,Admin_Block,admin_block_jan.csv
2024-01-24,296.82,Admin_Block,admin_block_jan.csv
2024-01-25,459.11,Admin_Block,admin_block_jan.csv
2024-01-26,109.93,Admin_Block,admin_block_jan.csv
2024-01-27,285.54,Admin_Block,admin_block_jan.csv
2024-01-28,387.68,Admin_Block,admin_block_jan.csv
2024-01-29,351.06,Admin_Block,admin_block_jan.csv
2024-01-30,260.49,Admin_Block,admin_block_jan.csv
2024-01-31,142.18,Admin_Block,admin_block_jan.csv
2024-01-01,73.91,Hostel,hostel_jan.csv
2024-01-02,188.98,Hostel,hostel_jan.csv
2024-01-03,316.67,Hostel,hostel_jan.csv
2024-01-04,155.8,Hostel,hostel_jan.csv
2024-01-05,484.24,Hostel,hostel_jan.csv
2024-01-06,475.27,Hostel,hostel_jan.csv
2024-01-07,431.78,Hostel,hostel_jan.csv
2024-01-08,262.55,Hostel,hostel_jan.csv
2024-01-09,428.66,Hostel,hostel_jan.csv
2024-01-10,109.0,Hostel,hostel_jan.csv
2024-01-11,188.93,Hostel,hostel_jan.csv
2024-01-12,258.35,Hostel,hostel_jan.csv
2024-01-13,383.83,Hostel,hostel_jan.csv
```

❖ **Building summary.csv**

Python Lab 5 > output > building_summary.csv > data

```
building,mean,min,max,sum
Admin_Block,276.6329032258065,61.2,490.1,8575.62
Hostel,285.17903225806447,68.3,484.24,8840.55
Library,279.5232258064516,50.64,479.96,8665.22
```

DISCUSSION AND ANALYSIS

This section interprets major trends observed in the visual outputs:

Daily trends often show weekday peaks and weekend dips.

Weekly summaries simplify comparisons and highlight anomalies or periods of unusually high usage.

Building-wise analysis identifies heavy-consumption buildings such as hostels, academic blocks, or laboratories

Scatter plots reveal base loads and abnormal spikes, helping detect inefficiencies or potential faults.

Overall, the combined analysis supports campus sustainability initiatives and better operational planning.

ADVANTAGES AND APPLICATIONS

Advantages

Automated data ingestion and cleani

Strong error-handling mechanism

Effective use of time-series analysis

Organized OOP-based architecture

Exportable and portable outputs

Applications

Monthly and weekly usage reporting

Identifying priority buildings for energy audits

Supporting green-campus strategies

Educational demonstrations for time-series and data-analysis topics

---

LIMITATIONS AND FUTURE SCOPE

Limitations

Only works with static CSV data

No real-time integration

Limited visualization (non-interactive)

No external variables like weather or occupancy included

Future Scope

IoT and smart-meter integration

Hourly or sub-hourly data analysis

Machine-learning-based consumption forecasting

Web-based interactive dashboards

Cost and carbon-emission estimation

REFLECTION

Working on this project provided a strong practical understanding of data pipelines, especially handling timestamps, resampling, and cleaning real-world datasets. Challenges such as inconsistent data formats and plotting issues helped improve problem-solving and debugging skills. Implementing OOP structures enhanced clarity and reusability in the code.

This project also highlighted how meaningful insights can be derived from simple energy datasets and motivated further interest in advanced areas such as forecasting and IoT-based monitoring.

CONCLUSION

This project successfully demonstrates how Python, Pandas, and Matplotlib can be used to systematically analyze campus electricity consumption. Through automated ingestion, cleaning, aggregation, visualization, and exporting, the system provides a complete workflow for energy analysis. The project also strengthens understanding of time-series data, OOP, and structured reporting—serving as a foundation for future smart-campus energy management systems.