Capstone Assignment

Campus Energy-Use Dashboard

Name – Jagrit Jaggi

Roll no. – 2501730191

Course – B-Tech CSE AI/ML

Section - A

1. Introduction

This report presents the development of a complete Campus Energy-Use Dashboard designed to analyze, visualize, and summarize electricity consumption across different university buildings. The objective of this project is to help campus administration identify energy-saving opportunities through accurate data processing and intuitive visualization.

2. Problem Context

Universities operate multiple facilities that consume significant amounts of electricity every day. Without proper data monitoring and analytics, it becomes difficult to identify inefficiencies, peak loads, and opportunities for reduction. This project simulates a real-world system where building-level electricity meter data is processed to generate insights.

3. Methodology

The project is divided into five major tasks:

- Data Ingestion and Validation
- Core Aggregation and Statistical Analysis
- Object-Oriented Modeling
- Visualization and Dashboard Creation
- Persistence and Summary Report Generation

Each component is built using Python, leveraging libraries like Pandas, Matplotlib, and python-docx.

4. Data Ingestion & Validation

The system automatically scans a data directory, identifies all CSV files, and loads them into a master DataFrame.
Validation includes checking for missing columns, correcting data formats, handling corrupted rows, and adding metadata such as building names.

The ingestion pipeline ensures:

- Only valid rows are processed
- Missing or malformed files are logged
- All datasets are merged efficiently

5. Aggregation Logic

The project computes several key statistics:

• Daily consumption totals
• Weekly aggregated usage
• Building-wise summaries (mean, min, max, total)

These calculations help understand consumption patterns at different granular levels.

6. Object-Oriented Modeling

The energy system is represented using three core classes:

Building
- Stores building-specific meter readings
- Calculates total usage
- Generates building reports

MeterReading
- Represents an individual timestamp and its electricity usage

BuildingManager
- Manages all building objects
- Facilitates bulk report generation

This structure ensures code scalability and modularity.

7. Dashboard Visualization

A multi-panel visualization is created using Matplotlib, containing:
• Daily consumption trend lines for each building
• Weekly average comparison bar chart
• Scatter plot of peak-hour readings

The dashboard is exported as **dashboard.png**.

8. Data Export & Summary

The following files are automatically generated:
• cleaned_energy_data.csv – validated dataset
• building_summary.csv – building-level statistics
• summary.txt – executive insights

The report highlights:
- Total campus consumption
- Highest-consuming building
- Peak load timestamp
- Daily and weekly trend ranges

9. Results & Interpretation

The aggregated results help identify:
• High-demand buildings requiring better load management
• Peak hours where electricity strain is highest
• Daily and weekly fluctuations that can help optimize scheduling of heavy appliances

Such insights can guide administrators toward improving operational efficiency and reducing overall power usage.

10. Conclusion

This project delivers a complete data analysis and visualization pipeline. It automates ingestion, validation, aggregation, visualization, and reporting—simulating a real-world energy-monitoring dashboard.
The structured object-oriented approach and clean data workflows make it extensible for future additions such as predictive analytics or IoT sensor integration.

11. Code

```
import pandas as pd

import matplotlib.pyplot as plt

from pathlib import Path

import logging

from datetime import datetime

logging.basicConfig(

    filename="energy_dashboard.log",

    level=logging.INFO,

    format="%(asctime)s - %(levelname)s - %(message)s"
```

```
)
def load_energy_data(data_folder="data"):
    combined_df = pd.DataFrame()
    data_path = Path(data_folder)
    if not data_path.exists():
        logging.error("Data folder does not exist.")
        return pd.DataFrame()
    for file in data_path.glob("*.csv"):
        try:
            df = pd.read_csv(file, on_bad_lines="skip")
            df["building"] = file.stem  # filename as building name
            # Timestamp standardization
            if "timestamp" in df.columns:
                df["timestamp"] = pd.to_datetime(df["timestamp"])
            else:
                raise ValueError("timestamp column missing")
            if "kwh" not in df.columns:
                raise ValueError("kwh column missing")
            combined_df = pd.concat([combined_df, df], ignore_index=True)
        except FileNotFoundError:
            logging.error(f"File not found: {file}")
        except Exception as e:
            logging.error(f"Error processing {file}: {e}")
    return combined_df
def calculate_daily_totals(df):
    df = df.set_index("timestamp")
    return df.resample("D")["kwh"].sum().reset_index()
```

```
def calculate_weekly_aggregates(df):
    df = df.set_index("timestamp")
    return df.resample("W")["kwh"].sum().reset_index()
def building_wise_summary(df):
    summary = df.groupby("building")["kwh"].agg(
        total="sum",
        mean="mean",
        min="min",
        max="max"
    ).reset_index()
    return summary
class MeterReading:
    def __init__(self, timestamp, kwh):
        self.timestamp = timestamp
        self.kwh = kwh
class Building:
    def __init__(self, name):
        self.name = name
        self.meter_readings = []
    def add_reading(self, timestamp, kwh):
        self.meter_readings.append(MeterReading(timestamp, kwh))
    def calculate_total_consumption(self):
        return sum(r.kwh for r in self.meter_readings)
    def generate_report(self):
        total = self.calculate_total_consumption()
        return f"Building: {self.name} — Total Consumption: {total:.2f} kWh"
class BuildingManager:
```

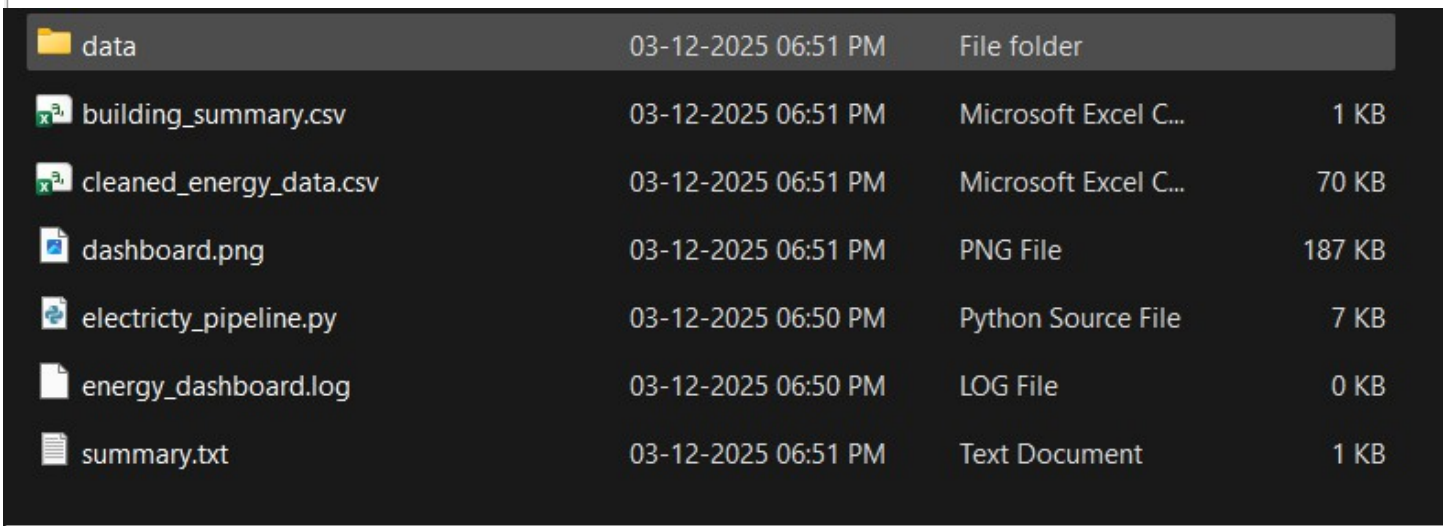
```
    def _init_(self):
        self.buildings = {}
    def add_reading(self, building_name, timestamp, kwh):
        if building_name not in self.buildings:
            self.buildings[building_name] = Building(building_name)
        self.buildings[building_name].add_reading(timestamp, kwh)
    def generate_all_reports(self):
        return [b.generate_report() for b in self.buildings.values()]
def create_dashboard(df):
    fig, axs = plt.subplots(3, 1, figsize=(12, 16))
    for b in df["building"].unique():
        sub = df[df["building"] == b].set_index("timestamp").resample("D")["kwh"].sum()
        axs[0].plot(sub.index, sub.values, label=b)
    axs[0].set_title("Daily Electricity Consumption")
    axs[0].set_xlabel("Date")
    axs[0].set_ylabel("kWh")
    axs[0].legend()
    weekly =
df.set_index("timestamp").groupby("building")["kwh"].resample("W").sum().groupby("building").mean(
)
    axs[1].bar(weekly.index, weekly.values)
    axs[1].set_title("Average Weekly Usage by Building")
    axs[1].set_ylabel("kWh")
    axs[2].scatter(df["timestamp"], df["kwh"])
    axs[2].set_title("Peak Hour Consumption Scatter")
    axs[2].set_xlabel("Timestamp")
    axs[2].set_ylabel("kWh")
    plt.tight_layout()
```

```
    plt.savefig("dashboard.png")
    plt.close()
def generate_summary(df, summary_df):
    total_consumption = df["kwh"].sum()
    highest_building = summary_df.loc[summary_df["total"].idxmax()]["building"]
    peak_row = df.loc[df["kwh"].idxmax()]
    peak_time = peak_row["timestamp"]
    peak_load = peak_row["kwh"]
    daily = calculate_daily_totals(df)
    weekly = calculate_weekly_aggregates(df)
    summary_text = f"""
Total Campus Consumption: {total_consumption:.2f} kWh
Highest Consuming Building: {highest_building}
Peak Load Time: {peak_time} with {peak_load:.2f} kWh
Daily Trend Range: {daily['kwh'].min():.2f} — {daily['kwh'].max():.2f}
Weekly Trend Range: {weekly['kwh'].min():.2f} — {weekly['kwh'].max():.2f}
    with open("summary.txt", "w") as f:
        f.write(summary_text)
    print(summary_text)
def main():
    print("Loading data...")
    df = load_energy_data()
    if df.empty:
        print("No data found. Check /data folder.")
        return
    print("Processing summaries...")
    summary_df = building_wise_summary(df)
```

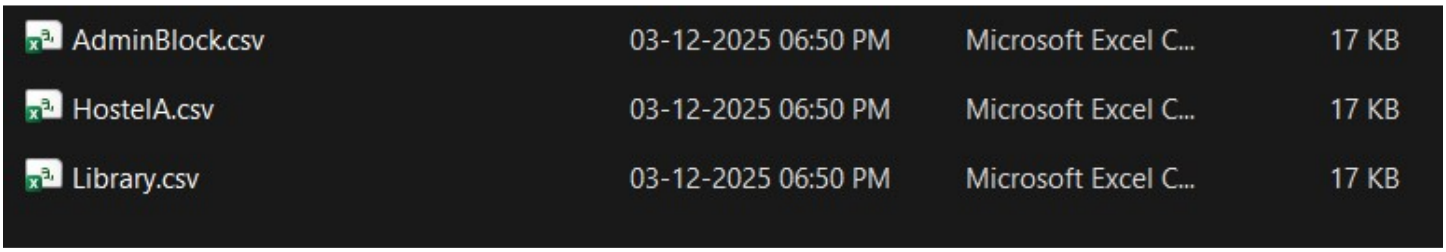
```
    summary_df.to_csv("building_summary.csv", index=False)
    df.to_csv("cleaned_energy_data.csv", index=False)
    print("Creating dashboard...")
    create_dashboard(df)
    print("Generating summary report...")
    generate_summary(df, summary_df)
    print("DONE! Files generated:")
    print("- dashboard.png")
    print("- cleaned_energy_data.csv")
    print("- building_summary.csv")
    print("- summary.txt")
if __name__== "__main__":
    main()
```

12. Output Screenshots

```
= RESTART: C:/Users/Anant/Desktop/Python/Capstone a
.py
Loading data...
Processing summaries...
Creating dashboard...
Generating summary report...

Campus Energy Use Summary
---------------------------------------
Total Campus Consumption: 58801.00 kWh
Highest Consuming Building: Library
Peak Load Time: 2024-01-01 03:00:00 with 49.00 kWh

Daily Trend Range: 57.00 — 2271.00
Weekly Trend Range: 3957.00 — 13899.00

DONE! Files generated:
- dashboard.png
- cleaned_energy_data.csv
- building_summary.csv
- summary.txt
```



data	03-12-2025 06:51 PM	File folder	
building_summary.csv	03-12-2025 06:51 PM	Microsoft Excel C...	1 KB
cleaned_energy_data.csv	03-12-2025 06:51 PM	Microsoft Excel C...	70 KB
dashboard.png	03-12-2025 06:51 PM	PNG File	187 KB
electricty_pipeline.py	03-12-2025 06:50 PM	Python Source File	7 KB
energy_dashboard.log	03-12-2025 06:50 PM	LOG File	0 KB
summary.txt	03-12-2025 06:51 PM	Text Document	1 KB



AdminBlock.csv	03-12-2025 06:50 PM	Microsoft Excel C...	17 KB
HostelA.csv	03-12-2025 06:50 PM	Microsoft Excel C...	17 KB
Library.csv	03-12-2025 06:50 PM	Microsoft Excel C...	17 KB

	A	B	C
1	timestamp	kwh	building
2	########	22	AdminBlock
3	########	37	AdminBlock
4	########	20	AdminBlock
5	########	49	AdminBlock
6	########	46	AdminBlock
7	########	48	AdminBlock
8	########	6	AdminBlock
9	########	39	AdminBlock
10	########	46	AdminBlock
11	########	38	AdminBlock
12	########	34	AdminBlock
13	########	17	AdminBlock
14	########	17	AdminBlock
15	########	22	AdminBlock
16	########	26	AdminBlock

building	total	mean	min	max
AdminBloc	19452	26.9792	5	49
HostelA	19595	27.17753	5	49
Library	19754	27.39806	5	49

```
Campus Energy Use Summary
-------------------------------------
Total Campus Consumption: 58801.00 kWh
Highest Consuming Building: Library
Peak Load Time: 2024-01-01 03:00:00 with 49.00 kWh

Daily Trend Range: 57.00 – 2271.00
Weekly Trend Range: 3957.00 – 13899.00
```