مركز البحرين للدراسات الاستراتيجية والدولية والطاقة
Bahrain Center for Strategic, International and Energy Studies

Analysis of household electricity consumption using optimization technique: A solution for Derasat think tank in Bahrain

A Thesis

By **Ameena Almohanna**

Bachelor of Science in Information & Communication Technology

At **Bahrain Polytechnic**

17/12/2024

Copyright

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

Signature Amena **Name** Ameena Almohanna **Date** 17 December 2024

Approval Signatures

The signatures of the people below document acceptance and approval of the formal thesis. The sponsor representative must have the authority to commit the client's resources to the project. The project manager is empowered by this thesis to proceed with the project as outlined in the charter and project plan.

Table 1: Approval signatures

Position/Title	Signature	Date
Project Manager	Amena Almohanna	17 December 2024
Project Management Office		

Abstract

The modern household experiences rising energy costs, while governments face an increase in electricity demand which makes energy-efficient management essential. In collaboration with the Derasat think tank in Bahrain, an analysis of electricity consumption patterns in a single Bahraini house will be conducted by utilization of an optimization technique to study a collected dataset on an hourly basis without the need for complex installations such as smart meters. Python is used for data analysis and visualization, providing Derasat organization with a scalable and user-friendly solution. Several crucial attributes in the data collection are assessed such as occupancy, outdoor temperature, and main appliances energy usage in Microsoft Excel. In addition, findings, and recommendations on how to reduce electricity consumption will be examined by comparing the original dataset with the optimized dataset. This approach offers a comprehensive insight into peak usage times and possible energy-saving opportunities making it a valuable source. Thus, the Derasat research Institute is interested in this study to discuss energy efficiency and policy development with energy-related companies and authorities.

Acknowledgements

I would like to express my sincere gratitude to my supervisor at Bahrain Polytechnic, Ms. Ancy Cheriyan. Without her encouragement, advice, and valuable feedback, I could not have completed this project. I have greatly benefited from her expertise and guidance in managing this project.

Additionally, I am grateful to the Derasat Research Center who provided me with the opportunity to work on this project. A special thank you to Engineer Sabeeka Khalid Ismaeel and Dr. Abdulla Al Abbasi from Derasat for providing insights and valuable support during the execution process.

I would like to thank Bahrain Polytechnic for sponsoring this research and providing the essential resources to accomplish the project. A special thanks to the course coordinator, Mr. Cyril Anthoni, and my tutor, Mr. Augustine Finucane, for their efforts and aid.

Lastly, I would like to extend my deepest appreciation to my family and friends for their endless kindness and motivation throughout this journey.

Table of Contents

Table of Figures

Table of Tables

List of symbols

Symbol	Description
°C	A Degree Celsius, a unit of temperature measurement.
$\sum$	It is called a summation symbol. It comes from the Greek letter sigma, which is the sum of a sequence of numbers in mathematics.
≤	In mathematics and programming, the "less than or equal to" symbol is used to indicate that one value is less than or equal to another.
∈	The symbol indicates that an item is a member of a set.

List of Abbreviations

Abbreviation	Definition
kWh	Stands for “Kilowatt hours”, a unit of energy which uses one kilowatt of power in one hour.
ID	Short for “**I**dentification **D**ocument”, is a unique value given to distinguish each record within a dataset.
AC	Short name for “**A**ir **C**onditioner”, a device that cools down the temperature.
TV	Stands for “Television”, a device that displays audio and visual broadcasts.
UML	Stands for “**U**nified **M**odeling **L**anguage”, which is used for specifying, visualizing, building, and documenting software artifacts.
ICT	An acronym for "**I**nformation and **C**ommunication **T**echnology", is used for communication, data processing, and sharing information.
pt	Stands for a measurement unit for font sizes.

Introduction

In this thesis, energy efficiency solutions are examined for a Bahraini household, addressing the problem of high energy consumption and its environmental impact due to inefficient use. Due to the high population, economic growth, elevated temperatures and Bahrain's low gas reserves, reducing energy waste is essential. An approach to assessing energy consumption patterns and efficiency improvements is presented by applying optimization algorithms and incorporating visuals from a collected dataset. The report continues with a comprehensive analysis of the issue, goals, previous research and possible solutions.

Project Rationale

Energy utilization has become a significant concern globally, particularly in terms of the sustainability of resources, economic efficiency, and its impact on the environment. Most households are unaware of their energy consumption patterns, resulting in inefficient electricity usage in higher costs. Using less energy for the same results allows devices to remain comfortable and productive. Therefore, the Derasat Research Center for energy studies is searching for ways to efficiently use energy and avoid waste to review it with relevant energy companies and policymakers.

The issue is extremely critical in the Kingdom of Bahrain because the country has limited natural gas reserves, a high population that increases energy consumption and is experiencing rapid economic growth. According to the Sustainable Energy Unit | Kingdom of Bahrain (2017), "Bahrain relies entirely on natural gas which is a limited and diminishing supply that is expected to last for around 15 years. Therefore, the country will have to depend on imported energy and users will have to expect higher electricity bills." (p. 18). Moreover, although Bahrain has a small geographical area, its population grows at an average rate of 7.4%. The number should rise to 2.128 million by 2030, up from 621 thousand in 1999 (Ministry of Information | Bahrain, 2024). Furthermore, energy demand increases as the economy expands. Due to the 3.4% expansion in the non-oil sector, Bahrain's GDP increased by 2.4% YoY during 2023 (Ministry of Finance and National Economy | Bahrain, 2023, p. 7). Compared to other countries, Bahrain consumes more electricity. The energy intensity is 54% higher than the Gulf Cooperation Council (GCC) average, and 88% higher than the world average (Sustainable Energy Unit | Kingdom of Bahrain, 2017). Additionally, temperatures are soaring, the Ministry of Transportation and Telecommunications' Meteorological Department stated in their monthly weather summary that July 2023 was the second hottest July in Bahrain since 1902 (Bahrain News Agency, 2023). Furthermore, Bahrain's carbon dioxide (CO_2) per capita emissions are among the highest in the world and continue to rise. The country's CO_2 emissions in 2013 exceeded the average of high-income countries and were almost five times higher than the global average (Sustainable Energy Unit | Kingdom of Bahrain, 2017). As population and temperature increase, economic development occurs, and industries expand, straining the nation's energy infrastructure.

Promoting energy efficiency, enhancing sustainable living and reducing global warming are the reasons for writing this report. As stated in the Bahrain Economic Vision 2030, which was announced in October 2008 by His Majesty King Hamad bin Isa Al Khalifa, "Protecting our natural environment will include directing investments to technologies that reduce carbon emissions, minimize pollution and promote the sourcing of more sustainable energy" (Sustainable Energy Unit | Kingdom of Bahrain, 2017).

Project Objectives

This thesis seeks to address these issues by reducing household energy consumption and presents a cost-effective algorithm. The aim is to collect appliance energy consumption patterns for one week, 24 hours a day in a single household in Muharraq, Bahrain using Microsoft Excel. Additionally, an optimal enhancement method for appliance usage data will be determined and compared with the original data collection by visualizing the results in Python.

The objectives: -

- Obtain a manual dataset on household electricity consumption every hour for seven days using Microsoft Excel.
- Clean and prepare the collected dataset using Python.
- Analyze the collected dataset using different methods.
- Show the results in an informative visualization.
- Apply an optimization technique to reduce the total electricity utilization and produce an optimized dataset.
- Analyze the optimized dataset.
- Visualize the new dataset.
- Compare the original and enhanced dataset results.
- Provide a reusable and scalable solution.
- Ensure Derasat research organization is satisfied and has meaningful insights to share with interested parties.

Prior Work

A relevant study by Stavrev & Ginchev (2024), employs reinforcement learning techniques to optimize household energy consumption without relying on real household data. Using hypothetical scenarios, these models illustrate various strategies for improving energy efficiency and sustainability. However, behavioral patterns and socioeconomic factors often get ignored, which can affect the accuracy and applicability of their results. As these simulations do not include real applications, how closely they relate to actual consumption behaviors is unclear. By integrating existing data, these simulations might be enhanced in relevance and accuracy.

Hypothesis

The main hypothesis of this thesis is that applying optimization techniques to analyze consumer energy consumption based on true data will result in noticeable savings in electricity usage and bills. By determining the primary appliances and usage patterns that lead significantly to higher electricity utilization, homeowners will make wise decisions and use devices more efficiently. It is expected that the knowledge gained from the data evaluation will contribute to useful suggestions that motivate energy-saving habits, eventually aiding Derasat organization in contributing to greater energy efficiencies and sustainability.

Proposed Solution

The proposed solution involves developing an in-depth energy consumption analysis application to examine electricity usage records from a single Bahraini residence and manage their electrical device usage in Python. This approach will be delivered through the subsequent main elements:

1. **Collection of information**: Hourly electricity utilization data will be collected manually from a Bahraini household for one week to capture real-time appliance usage patterns in Microsoft Excel.

2. **Cleaning of data**: All information gathered will be cleaned and well formatted to remove any errors or nulls, ensuring a reliable dataset.

3. **Analysis of the dataset**: An overall analysis of appliance usage patterns is performed, allowing for peak consumption times to be identified, as well as energy-intensive devices.

4. **Optimization technique:** An advanced optimization technique will be employed to find the ideal way to operate devices in the house.

5. **Analysis of the optimized dataset:** Analysis methods will be conducted on the new environment-friendly dataset to determine whether the results have improved.
6. **Visualization of data**: Consumers will be able to easily understand their energy consumption patterns due to clear and understandable graphical displays, such as charts and graphs.

Description of the Report

This report outlines the critical components of the project, starting with a **Background** section covering the key theories, literature review, and technology selection. The **Design** section discusses algorithm development, data structure layout, and system diagrams in detail. The next section is the **implementation**, which presents tool installation, and analysis of six different methods before and after data enhancement. The **testing** section describes the concepts of functionality, usability, and acceptance. The **results** and **discussion** section summarizes the research findings, accomplishments, obstacles encountered, and future directions. The **LESPI** section presents perspectives on the project from a legal, ethical, social, and professional perspective. Finally, the **Conclusion** highlights the most important lessons learned, and the **Appendices** contain comprehensive instructions and additional design materials.

Background

This project requires a basic understanding of energy consumption analysis principles, the client's background and technologies. The purpose of this section is to provide an overview of the theoretical frameworks and technologies that inform this project.

About the client

Derasat think tank conducts research and insight on policy and strategic topics to address Bahrain's social and economic needs. The organization provides support for initiatives that promote environmental protection, economic stability, and resource efficiency in the Kingdom of Bahrain (Derasat, 2023). Derasat's interest in energy research aligns with Bahrain's broader objectives for energy conservation and sustainability, which is relevant to the current project. By offering comprehensive analytics to support informed decision-making at both the community and national levels, Derasat is helping to address the inefficient use of household energy. As part of this project, Derasat will enhance its understanding of household energy usage patterns and explore possible avenues for promoting efficient energy practices in Bahrain.

Related Theory

Two essential theories are involved in this project, optimization techniques and data analysis. A key role of optimization techniques is to utilize energy efficiently, reduce environmental impact and ensure optimal operating conditions by targeting the maximum or minimum depending on the context. Among these techniques are linear programming, genetic algorithms, and integer programming, each producing solutions to energy-related issues (Alwan, 2016). Moreover, information analysis offers an in-depth view of consumption patterns, enabling researchers to extract insight by cleaning, transforming, and visualizing data. The types include Descriptive Analysis and Inferential Analysis, which identify patterns and trends using statistical measures such as mean, median, standard deviation, and correlation (Jain, 2020). These techniques support decision-making and enhance efficiency across energy and environmental fields.

Project Technology

The project will primarily use Microsoft Excel for data storage and Python programming language for data analysis and optimization approach to achieve its objectives (See Table 1). Python is used via Jupyter Notebook which is an open-source web application that enables data scientists to create and share documents including live code, equations, and multimedia. Jupyter Notebook is the best way to access Python since it divides code into smaller segments and executes cells in any order (Silverio, 2023). This will enable the client to run each code carefully and understand the code better. Python supports many libraries, such as Pandas for data manipulation, NumPy for numerical calculations, Matplotlib to create static, animated, and interactive displays with customization and Seaborn for making statistical graphics more accessible and visually appealing with less code (W3schools, 2024). Energy consumption data can be analyzed and visualized using these tools. According to Rosen and Farsi (2022), "One of the drawbacks of classical optimization methods is that they can yield unsatisfactory results when used to find multiple solutions. This is because they can obtain different solutions for different starting points in multiple runs of the algorithm". However, in this case, the focus is only on minimizing energy usage. In addition, the use of Microsoft Excel as a database is convenient and straightforward but has limitations, for instance, if there is an error in a formula or data cell, all related calculations will be incorrect (Grundy, 2023). Nevertheless, it can be initially used before moving to more advanced databases (Kitto, 2018). Finally, for presenting data, a dashboard using the Microsoft Power BI analytics tool will be created for interactive reports to make data analysis more engaging and accessible (Microsoft, 2024).

Table 1: Technologies images

Technology Name	Technology Logo Image
Microsoft Excel	Figure 1: Microsoft Excel logo
Jupyter Notebook	Figure 2: Jupyter Notebook logo
Python	Figure 3: Python logo
Microsoft Power BI	Figure 4: Power BI logo

Related Work

In Bahrain, several relevant works have been done on energy management. For example, a Bahraini project was implemented by installing solar panels in eight public schools to reduce fossil fuel use and increase energy efficiency. In addition to lowering school electricity costs, this project contributes to the country's goal of increasing renewable energy use to 5% by 2025. It is also a significant step toward greener economies, as it supports broader energy efficiency efforts (Gulf Daily News, 2019). Another example, the work includes integrating a district cooling system (DCS) into government buildings in Bahrain. Using a DCS, chilled water is distributed through underground pipes to multiple buildings, improving energy efficiency and reducing individual cooling systems. Through this innovative approach, government facilities can become more sustainable and efficient while addressing a range of economic and environmental concerns. (Ministry of Works | Bahrain, 2008).

Design

This section outlines the design solution adopted for improving household energy consumption before starting with the implementation phase to show the workflow. It describes how algorithms are designed, and how data structures are chosen, with detailed UML diagrams and flowcharts. The design matches the specifications, as it is of high quality and can be implemented easily by a system developer.

Solution Design

The purpose of this section is to describe all the design methodologies used within the project. This part includes the data collection method, system architecture, configuration diagrams and algorithm layout. Moreover, the appendix section has additional design information.

1. Data Structures

Data structures are specialized formats used to organize, process, retrieve, and store data in computer programs. Controlling data flow across inputs, processes, and outputs allows machines and users to manage and access data efficiently. Among the benefits of data structures are efficient data persistence, resource management, data exchange, and scalability (Brooks, 2021). In this project, data is structured using Microsoft Excel.

Data collection is done manually in a Bahrani house in Arad, Muharraq. Data is recorded hourly in one week from Sunday 13/10/2024, 12:00 a.m. till Saturday 19/10/2024, 11:00 p.m. The dataset holds twenty-three columns and 168 rows. The information includes unique identifiers, date, time, number of people in the house in the current hour, outside temperature in degrees Celsius using the Meteostat website for weather data which is the largest provider of open climate data (Meteostat, 2024), various appliances to understand usage patterns and total energy consumption per hour in kWh.

Table 2: Dataset attributes details

Number of columns	Column Name	Fixed value	Description	Type	Null	Category
1	HouseID	A(Number)	Unique identifier such as “A3”, where “A” is a fixed prefix that refers to one house sample and “3” presents the third row of data, starting with A1 till A168.	object	No	ID
2	Date	-	In a format (DD/MM/YYYY), seven days from Sunday 13/10/2024 till Saturday 19/10/2024.	object	No	Date
3	Time	-	Starting with 12:00 a.m. to 11:00 p.m. on an hourly basis.	object	No	Time
4	Occupancy	-	The number of people staying in the house in a certain hour.	int64	No	Count of people
5	OutdoorTemperature	-	Outdoor temperature in degrees Celsius in Arad, Bahrain using Meteostat	float64	No	Temperature in °C

			climate records (Meteostat, 2024).			
6	AC1	3.4 kWh	Air conditioner x1 brand in highest power.	float64	Yes	Air conditioner
7	AC2	1.83 kWh	Air conditioner x2 brand in medium power.	float64	Yes	Air conditioner
8	AC3	3.4 kWh	Air conditioner x1 brand highest power.	float64	Yes	Air conditioner
9	AC4	2 kWh	Air conditioner x1 brand in normal power.	float64	Yes	Air conditioner
10	AC5	1.98 kWh	Air conditioner x2 brand in normal power.	float64	Yes	Air conditioner
11	AC6	2.08 kWh	Air conditioner x3 brand in normal power.	float64	Yes	Air conditioner
12	AC7	2.12 kWh	Air conditioner x4 brand in medium power.	float64	Yes	Air conditioner
13	AC8	2.12 kWh	Air conditioner x4 brand in medium power.	float64	Yes	Air conditioner
14	TV	0.27 kWh	Television x5 brand.	float64	Yes	Television
15	Refrigeter1	0.5 kWh	Normal refrigerator x6 brand.	float64	No	Refrigerator
16	Refrigeter2	0.5 kWh	Normal refrigerator x7 brand.	float64	No	Refrigerator

17	Refrigeter3	0.3 kWh	Mini refrigerator x8 brand.	float64	No	Refrigerator
18	Freezer	0.0354 kWh	Separate Freezer brand x9.	float64	No	Refrigerator
19	VacuumCleaner	1 kWh	Vacuum Cleaner brand x10.	float64	Yes	Cleaning
20	ClotheWasher	0.41 kWh.	Clothe washer brand x11.	float64	Yes	Cleaning
21	ClothesDryer	2.7 kWh	Clothes dryer brand x12.	float64	Yes	Cleaning
22	WaterDispenser	0.94 kWh	Water dispenser, hot and cold mode, brand x13.	float64	No	Water dispenser
23	TotalUsagekWh	-	Total consumption of all appliances in each hour in kWh.	float64	No	Total energy usage

As shown in the table above, although all AC1, AC3, and AC4 are from the same company brand and model, AC1 and AC3 use the highest power at 3.4 kWh, while AC4 uses normal power at 2 kWh. A total of 17 devices are used in October, a warm month as it transitions from summer to autumn, which means it is generally sunny and dry. The following are the power usage categories for each machine:

- Low consumption appliances: Usually under 1 kWh per day.
- Medium consumption appliances: Approximately between 1 and 2 kWh per day.
- High consumption appliances: More than 2 kWh per day.

Thus, low consumption devices are TV, Refrigeter1, Refrigeter2, Refrigeter3, Freezer, ClotheWasher and WaterDispenser. Medium consumption devices are AC2, AC4, AC5 and VacuumCleaner. High Consumption Devices are AC1, AC3, AC6, AC7, AC8 and ClothesDryer.

2. UML Diagrams

According to Alhumaidan (2012), UML diagrams provide an overview and outline the advantages and disadvantages of the system. Further, UML ensures confidence in system design and reduces the risk of errors. It is concise, easy to understand, and comprehensive, effectively handling complex systems through straightforward concepts and notations. Examples of the UML diagrams are: -

- **Use Case diagram**

UML Use Case diagram visualizes the functionality a system provides and the interactions between actors (individuals or external systems) to achieve desired goals. By focusing on user involvement, they provide a framework that supports requirements analysis and testing, thereby managing system complexity (Alhumaidan, 2012).

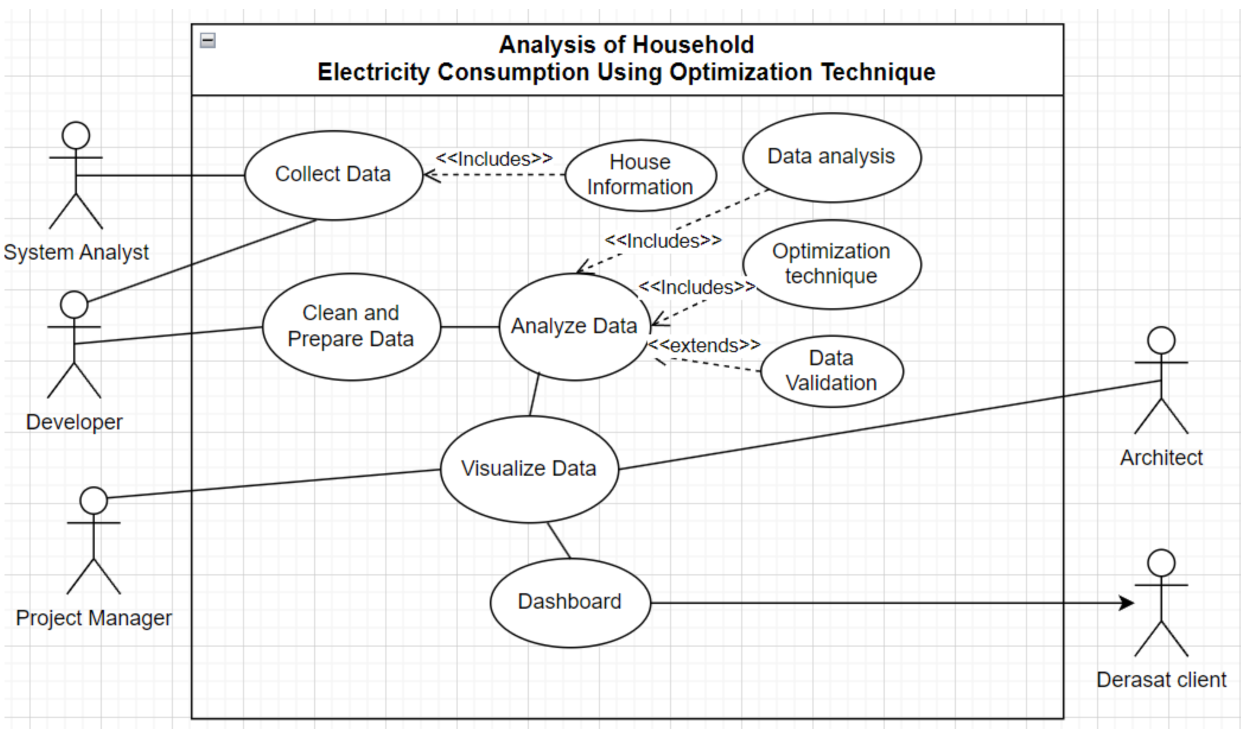


Figure 5: Use Case diagram

The above figure illustrates the project workflow using a Use Case diagram on the Draw.io website. The main actors are the System Analyst, Developer, Project Manager, Architect, and Derasat Client. The actors take part in data collection, preparation, analysis, and visualization. The terms "includes" and "extends" refer to dependencies and additional steps respectively.

- **Sequence diagram**

A sequence diagram is the sequence of messages exchanged between objects in a system over time and visualizes the communication process. Dynamic interactions are modelled using elements such as lifelines, messages, activations, and a time axis, considering both the order of events and the role of the objects. Objects are represented horizontally and time vertically, offering a detailed view of system behaviour, especially when analyzing complicated connections (Alhumaidan, 2012).

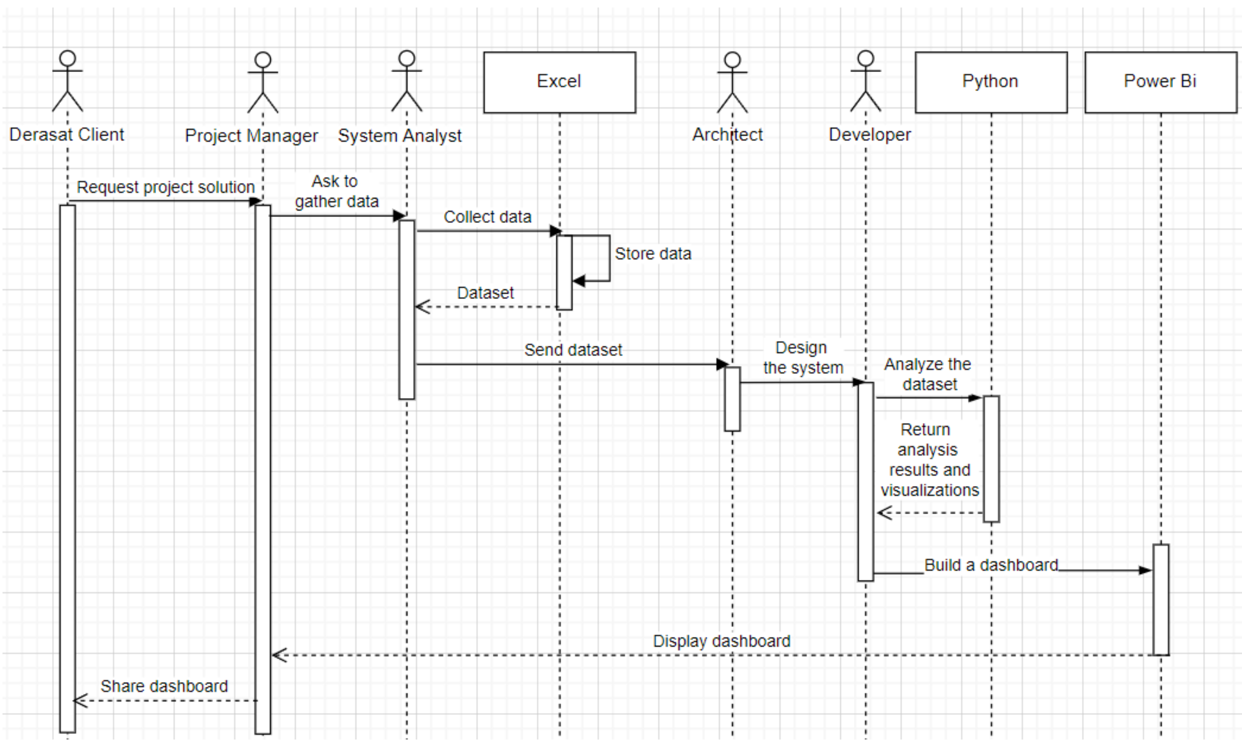


Figure 6: Sequence diagram

The above diagram displays the communication steps, starting with the client's request for a project solution and ending with the Project Manager sharing the results with the customer. Moreover, the "Analyze the dataset" step in Python includes analyzing and visualizing the dataset before and after improving the data collection. The diagram was made using the Draw.io website.

- **Class diagram**

UML class diagram illustrates the classes in a system and their relationships, having notations for classes, attributes, operations, and associations. The class acts as a fundamental building block for object-oriented design, allowing visibility control of attributes and defining relationships, such as many-to-many and one-to-one (Alhumaidan, 2012). The following diagram is made using the StarUML program.

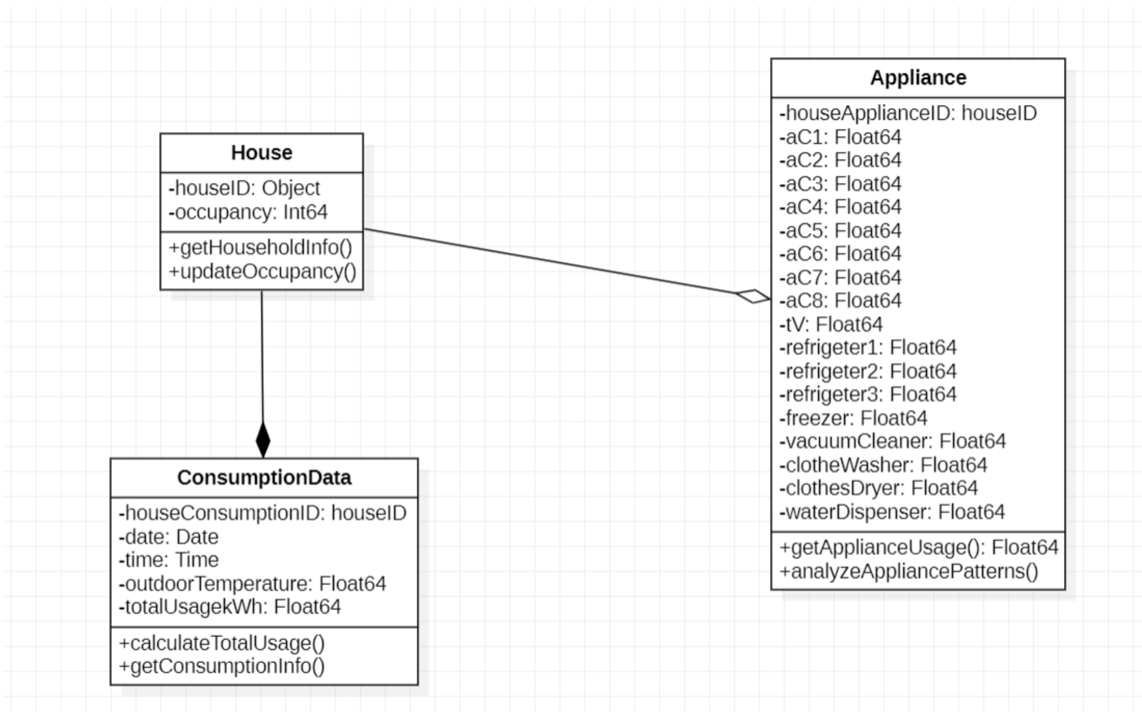


Figure 7: Class diagram

The class diagram figure presents three classes, “House”, “Appliance” and “ConsumptionData”. Each class has attributes and methods with data types, however, the “housed” is the unique identifier of all. The arrows point to the relationships. The "House" class owns the "ConsumptionData" class as part of its data and aggregates several "Appliance" class instances as part of its architecture.

3. Algorithms

This section explains in text and visual form how the project's execution will be handled by demonstrating the sub-tasks of its overall functionality.

- **Flow chart**

A flowchart is an algorithm visually, focusing on clarity and simplicity. It shows a sequence of steps and decisions using shapes such as rectangles, diamonds, and ovals connected by arrows (Lucidchart, 2024). The following flowchart is made using the draw.io website.

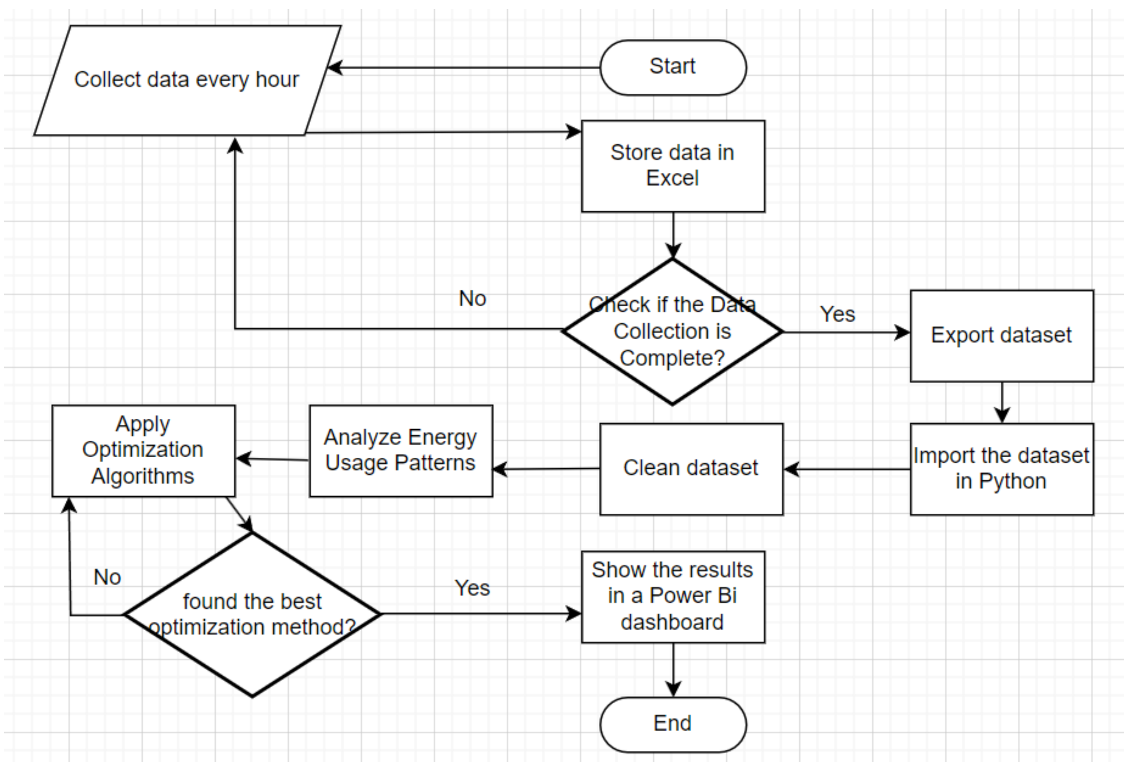


Figure 8: The flow chart

The figure illustrates a data collection and analysis process for energy expenditure in a household. It starts with regular data harvesting, storing data hourly in Excel. After data collection is complete, the dataset is exported and imported into Python. Cleansing and analyzing the data identifies energy consumption patterns. Afterwards, the best optimization method is applied. Lastly, a Power BI dashboard will be created to compare the earlier dataset results with the newly generated results.

- **Pseudocode**

A pseudocode consists of everyday language used to describe an algorithm and resemble a simplified programming language. The code steps can be clarified based on the previously defined flowchart structure (Codecademy, 2024).

```
'''Pseudocode:
# Start
start = True

# Collect data every hour
while start:
  # Store data in Excel
  store_data_in_excel()

  # Check if the Data Collection is Complete
  data_collection_complete = check_data_collection_complete()

  # If data collection is complete, export the dataset
  if data_collection_complete:
    export_dataset()

    # Import the dataset in Python
    import_dataset()

    # Clean the dataset
    clean_dataset()

    # Analyze Energy Usage Patterns
    analyze_energy_usage_patterns()

    # Apply Optimization Technique to Reduce Energy Usage
    apply_optimization_technique()

    # Analyze the Optimized Dataset
    analyze_optimized_dataset()

    # Compare the Results Before and After Optimization
    compare_results_before_after()

    # Show the results in a Power BI dashboard
    show_results()

    # End
    start = False

    '''
```

```
'Pseudocode:\n# Start\nstart = True\n\n# Collect data every hour\nwhile start:\n  # Store data in Excel\n  store_data_in_ex
cel()\n\n  # Check if the Data Collection is Complete\n  data_collection_complete = check_data_collection_complete()\n\n  #
If data collection is complete, export the dataset\n  if data_collection_complete:\n    export_dataset()\n\n    # Import th
e dataset in Python\n    import_dataset()\n\n    # Clean the dataset\n    clean_dataset()\n\n    # Analyze Energy Usage Pat
terns\n    analyze_energy_usage_patterns()\n\n    # Apply Optimization Technique to Reduce Energy Usage\n    apply_optimiza
tion_technique()\n\n    # Analyze the Optimized Dataset\n    analyze_optimized_dataset()\n\n    # Compare the Results Befor
e and After Optimization\n    compare_results_before_after()\n\n    # Show the results in a Power BI dashboard\n    show_re
sults()\n\n    # End\n    start = False\n    \n    '
```

Figure 9: Pseudocode

The figure presents the pseudocode of how energy usage data can be collected and analyzed in a Python file as a comment before coding. An Excel file is used to store the data collected hourly by the program. Upon completion of data collection, the program exports the data to Python for cleanup and analysis. Next, optimization algorithms are applied and analyzed. The outcomes for both the new and old datasets are compared. The results are then displayed on a Power BI dashboard.

4. System Architecture

A system architecture shows how a complex system is organized, what its components are, how they interact, and what behaviour it should show. As it outlines a structure that adapts to evolving needs, it supports scalability, performance, reliability, and security. By aligning technology with business goals, well-defined architecture helps reduce costs, minimize risks, and streamline development and maintenance processes (Alooba, 2022).

- **Topology Diagram**

Topology diagrams show how devices communicate within a network, detailing their structure and connections. There are a variety of topologies, including bus, ring, star, mesh, and hybrid, each with its performance, reliability, and cost advantages and disadvantages. The use of topology diagrams can help diagnose network issues and plan capacity to improve network uptime, efficiency, and productivity (Sunbird, 2021).

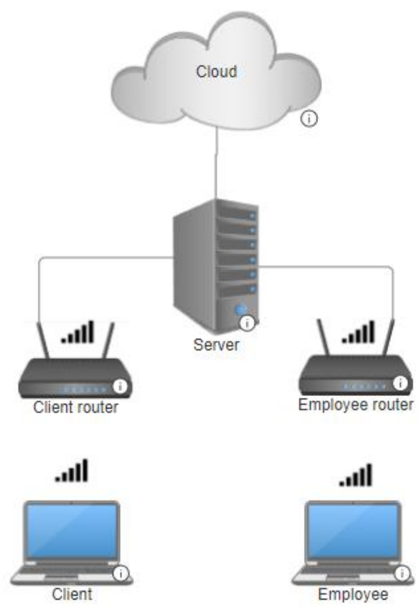


Figure 10: Topology Diagram

The image shows a network diagram with two routers, two devices, a central server and a cloud network. Each router is connected to a laptop with an internet connection, one for an employee and one for a client. The employee is using a laptop to store a dataset locally in Excel and work on Python with connectivity to the internet for data analysis, dashboard creation and sending findings to the client. Finally, the client will receive the final product using an internet connection. The diagram is built with the SmartDraw app.

- **Deployment Diagram**

Deployment diagrams serve as a structural diagram that models the physical aspects of an object-oriented system, representing the configuration of runtime processing nodes and their components. The static deployment view shows the hardware topology and communication paths within the system. In addition to planning the system architecture, deployment diagrams document the installation of software components and show the interaction between the systems and the protocols in use. Modelling requires identifying a list of unique devices, using stereotypes for visual clarity, and accurately depicting the relationships between components and nodes (Visual Paradigm, 2024).

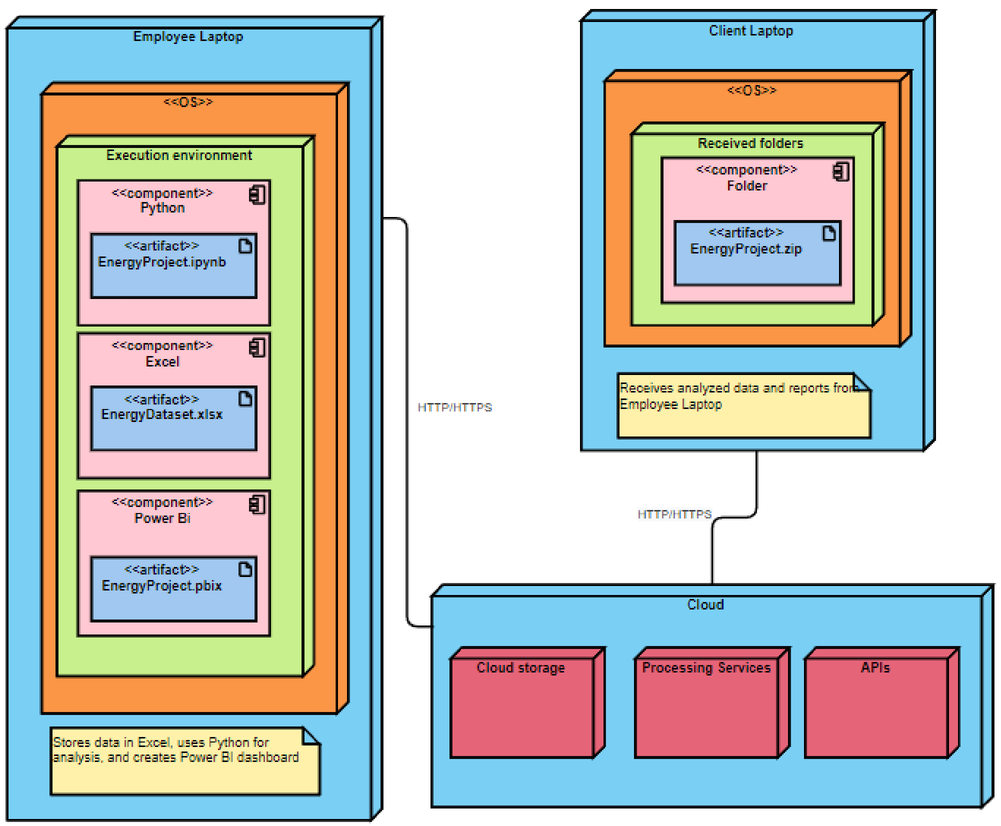


Figure 11: Deployment Diagram

The deployment diagram figure explains a data processing and analyzing system where the **Employee Laptop** works on Python, Excel, and Power BI. Secure data exchange occurs via the Cloud, which provides storage, processing, and APIs. The **Client Laptop** receives the final compressed project folder with the results from the employee via HTTP/HTTPS, ensuring the secure transfer of files. With this setup, employees can process and deliver data efficiently to clients through cloud services. The diagram is created using the online Visual Paradigm.

Implementation

After the dataset is collected and stored in Microsoft Excel, the necessary installation will be done before work begins. Following that, the data collection is imported as a ".xlsx" file into a Jupyter Notebook to use Python (See Appendix). The dataset will be cleaned and analyzed using essential libraries like NumPy, Pandas and Matplotlib for data manipulation, analysis, and visualization. Missing values are handled, and data types are formatted, followed by an analysis of energy consumption and an optimization technique for usage reduction.

Importing Python Libraries

1. Importing libraries in Python.

```python
# Importing libraries: -
# Import the pandas library as pd for data manipulation and analysis.
import pandas as pd

# Import the numpy library as np for numerical operations.
import numpy as np

# Import the matplotlib library as plt for visualizations
import matplotlib.pyplot as plt

# Importing cm from matplotlib for generating color maps for plotting.
import matplotlib.cm as cm

# Import seaborn as sns for statistics representaions
import seaborn as sns

# Import the datetime library
import datetime

# Import train_test_split to split data into training and testing sets, GridSearchCV for model tuning.
from sklearn.model_selection import train_test_split, GridSearchCV

# Import the statsmodels library as sm, which provides classes and functions for estimating and testing statistical models.
import statsmodels.api as sm

# Import LinearRegression from sklearn.linear_model for building and training linear regression models.
from sklearn.linear_model import LinearRegression

# Import DecisionTreeClassifier for building and training decision tree models.
from sklearn.tree import DecisionTreeClassifier

# Import metrics for evaluating model performance:
# - classification_report: provides detailed precision, recall, and F1-score.
# - confusion_matrix: summarizes classification accuracy.
# - mean_squared_error: measures average prediction error (lower is better).
# - r2_score: indicates how well the model explains data variance (higher is better).
from sklearn.metrics import mean_squared_error, r2_score, classification_report, confusion_matrix

# Import KMeans from sklearn.cluster to perform K-Means clustering, a method for grouping data into clusters.
from sklearn.cluster import KMeans

# Import Axes3D from mpl_toolkits.mplot3d to enable 3D plotting capabilities in matplotlib.
from mpl_toolkits.mplot3d import Axes3D
```

Figure 12: Importing libraries

2. Importing the dataset as “dt” and verifying importation.

```python
# Import the dataset from an Excel file as dt.
dt = pd.read_excel('C:/Users/ameen/OneDrive/Desktop/EnergyProject/EnergyDataset.xlsx')

# Print the first 5 rows of the dataset, a quick overview of the data and to test the code is working.
print("First 5 rows of the dataset:")
print(dt.head())

# Line to seperate codes
print('----------------------------------------------------------------------------------------------------------------')

# Check the structure and data types of each column
print(dt.info())
```

Figure 13: Import the dataset and verification

3. Dataset overview results

```
First 5 rows of the dataset:
  HouseID        Date      Time  Occupancy  OutdoorTemperature  AC1  AC2  AC3  \
0      A1  13/10/2024  00:00:00          4                30.0  3.4  NaN  3.4
1      A2  13/10/2024  01:00:00          4                30.0  3.4  NaN  3.4
2      A3  13/10/2024  02:00:00          4                30.0  3.4  NaN  3.4
3      A4  13/10/2024  03:00:00          4                29.0  3.4  NaN  3.4
4      A5  13/10/2024  04:00:00          4                29.0  3.4  NaN  3.4

   AC4  AC5  ...  TV  Refrigeter1  Refrigeter2  Refrigeter3  Freezer  \
0  NaN  NaN  ... NaN          0.5          0.5          0.3   0.0354
1  NaN  NaN  ... NaN          0.5          0.5          0.3   0.0354
2  NaN  NaN  ... NaN          0.5          0.5          0.3   0.0354
3  NaN  NaN  ... NaN          0.5          0.5          0.3   0.0354
4  NaN  NaN  ... NaN          0.5          0.5          0.3   0.0354

   VacuumCleaner  ClotheWasher  ClothesDryer  WaterDispenser  TotalUsagekWh
0            NaN           NaN           NaN            0.94        11.1954
1            NaN           NaN           NaN            0.94        11.1954
2            NaN           NaN           NaN            0.94        11.1954
3            NaN           NaN           NaN            0.94        11.1954
4            NaN           NaN           NaN            0.94        11.1954

[5 rows x 23 columns]
----------------------------------------------------------------------------------------------------------------
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 168 entries, 0 to 167
Data columns (total 23 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   HouseID             168 non-null    object
 1   Date                168 non-null    object
 2   Time                168 non-null    object
 3   Occupancy           168 non-null    int64
 4   OutdoorTemperature  168 non-null    float64
 5   AC1                 39 non-null     float64
 6   AC2                 14 non-null     float64
 7   AC3                 78 non-null     float64
 8   AC4                 22 non-null     float64
 9   AC5                 57 non-null     float64
 10  AC6                 13 non-null     float64
 11  AC7                 109 non-null    float64
 12  AC8                 84 non-null     float64
 13  TV                  90 non-null     float64
 14  Refrigeter1         168 non-null    float64
 15  Refrigeter2         168 non-null    float64
 16  Refrigeter3         168 non-null    float64
 17  Freezer             168 non-null    float64
 18  VacuumCleaner       6 non-null      float64
 19  ClotheWasher        6 non-null      float64
 20  ClothesDryer        5 non-null      float64
 21  WaterDispenser      168 non-null    float64
 22  TotalUsagekWh       168 non-null    float64
dtypes: float64(19), int64(1), object(3)
memory usage: 30.3+ KB
None
```

Figure 14: Dataset overview results

Data Cleaning and Preparing

The dataset will be cleaned to remove any errors or null fields, ensuring correct analysis. All empty fields will be replaced with a “0.0” value. Moreover, the time attribute will be formatted automatically by Python from 12:00 a.m. and 11:00 p.m. to 00:00:00 and 23:00:00 respectively. As it is collected every hour and minutes and seconds are not needed, it still needs to be formatted to 0 and 23 instead of 00:00:00 and 23:00:00 respectively. Additionally, each attribute will be saved in a variable to be called.

1. Code for data cleaning and preparation.

```
In [123]:
# Dataset cleaning and preparation: -

# Replace empty fields, NaN values with 0.0.
dt = dt.fillna(0.0)

# Check for any remaining missing values
print(dt.isnull().sum())

print('----------------------------------------------------------------------------------------------------------')

# Data formatting for Time attribute insted of 12 AM and 11 PM becomes 0 and 23.

# The lambda function applies strftime('%H') to each time entry, keeping only the hour portion
dt["Time"] = dt["Time"].apply(lambda x: x.strftime('%H'))

# Convert the modified Time column from string format to integer type for further analysis or plotting
dt['Time'] = dt['Time'].astype(int)

# Print dataset to confirm the changes.
print(dt)

print('----------------------------------------------------------------------------------------------------------')


# Check the structure and data types of each column
print("Dataset Info:")
print(dt.info())

print('----------------------------------------------------------------------------------------------------------')

# Declare variables: -
HouseID = dt["HouseID"]
Date = dt["Date"]
Time = dt["Time"]
Occupancy = dt["Occupancy"]
OutdoorTemperature = dt["OutdoorTemperature"]
AC1 = dt["AC1"]
AC2 = dt["AC2"]
AC3 = dt["AC3"]
AC4 = dt["AC4"]
AC5 = dt["AC5"]
AC6 = dt["AC6"]
AC7 = dt["AC7"]
AC8 = dt["AC8"]
TV = dt["TV"]
Refrigeter1 = dt["Refrigeter1"]
Refrigeter2 = dt["Refrigeter2"]
Refrigeter3 = dt["Refrigeter3"]
Freezer = dt["Freezer"]
VacuumCleaner = dt["VacuumCleaner"]
ClotheWasher = dt["ClotheWasher"]
ClothesDryer = dt["ClothesDryer"]
WaterDispenser = dt["WaterDispenser"]
TotalUsagekWh = dt["TotalUsagekWh"]
```

Figure 15: Dataset cleaning and preparation

2. Data cleaning and preparation results.

```
HouseID               0
Date                  0
Time                  0
Occupancy             0
OutdoorTemperature    0
AC1                   0
AC2                   0
AC3                   0
AC4                   0
AC5                   0
AC6                   0
AC7                   0
AC8                   0
TV                    0
Refrigeter1           0
Refrigeter2           0
Refrigeter3           0
Freezer               0
VacuumCleaner         0
ClotheWasher          0
ClothesDryer          0
WaterDispenser        0
TotalUsagekWh         0
dtype: int64
```

Figure 16: Results with no Null values

```
--------------------------------------------------------------------------------------------------------------
    HouseID        Date  Time  Occupancy  OutdoorTemperature  AC1  AC2  AC3  \
0        A1  13/10/2024     0          4                30.0  3.4  0.0  3.4
1        A2  13/10/2024     1          4                30.0  3.4  0.0  3.4
2        A3  13/10/2024     2          4                30.0  3.4  0.0  3.4
3        A4  13/10/2024     3          4                29.0  3.4  0.0  3.4
4        A5  13/10/2024     4          4                29.0  3.4  0.0  3.4
..      ...         ...   ...        ...                 ...  ...  ...  ...
163    A164  19/10/2024    19          4                31.7  0.0  0.0  0.0
164    A165  19/10/2024    20          5                31.3  0.0  0.0  0.0
165    A166  19/10/2024    21          2                30.4  0.0  0.0  0.0
166    A167  19/10/2024    22          2                30.3  0.0  0.0  0.0
167    A168  19/10/2024    23          5                29.7  0.0  0.0  0.0

     AC4   AC5  ...    TV  Refrigeter1  Refrigeter2  Refrigeter3  Freezer  \
0    0.0  0.00  ...  0.00          0.5          0.5          0.3   0.0354
1    0.0  0.00  ...  0.00          0.5          0.5          0.3   0.0354
2    0.0  0.00  ...  0.00          0.5          0.5          0.3   0.0354
3    0.0  0.00  ...  0.00          0.5          0.5          0.3   0.0354
4    0.0  0.00  ...  0.00          0.5          0.5          0.3   0.0354
..   ...   ...  ...   ...          ...          ...          ...      ...
163  0.0  1.98  ...  0.27          0.5          0.5          0.3   0.0354
164  0.0  0.00  ...  0.27          0.5          0.5          0.3   0.0354
165  0.0  0.00  ...  0.27          0.5          0.5          0.3   0.0354
166  0.0  0.00  ...  0.27          0.5          0.5          0.3   0.0354
167  0.0  0.00  ...  0.27          0.5          0.5          0.3   0.0354

     VacuumCleaner  ClotheWasher  ClothesDryer  WaterDispenser  TotalUsagekWh
0              0.0           0.0           0.0            0.94        11.1954
1              0.0           0.0           0.0            0.94        11.1954
2              0.0           0.0           0.0            0.94        11.1954
3              0.0           0.0           0.0            0.94        11.1954
4              0.0           0.0           0.0            0.94        11.1954
..             ...           ...           ...             ...            ...
163            0.0           0.0           0.0            0.94         8.7654
164            0.0           0.0           0.0            0.94         6.7854
165            0.0           0.0           0.0            0.94         6.7854
166            0.0           0.0           0.0            0.94         6.7854
167            0.0           0.0           0.0            0.94         6.7854

[168 rows x 23 columns]
--------------------------------------------------------------------------------------------------------------
```

Figure 17: Time attribute is formatted

```
----------------------------------------------------------------------------------------------------------
Dataset Info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 168 entries, 0 to 167
Data columns (total 23 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   HouseID             168 non-null    object
 1   Date                168 non-null    object
 2   Time                168 non-null    int32
 3   Occupancy           168 non-null    int64
 4   OutdoorTemperature  168 non-null    float64
 5   AC1                 168 non-null    float64
 6   AC2                 168 non-null    float64
 7   AC3                 168 non-null    float64
 8   AC4                 168 non-null    float64
 9   AC5                 168 non-null    float64
 10  AC6                 168 non-null    float64
 11  AC7                 168 non-null    float64
 12  AC8                 168 non-null    float64
 13  TV                  168 non-null    float64
 14  Refrigeter1         168 non-null    float64
 15  Refrigeter2         168 non-null    float64
 16  Refrigeter3         168 non-null    float64
 17  Freezer             168 non-null    float64
 18  VacuumCleaner       168 non-null    float64
 19  ClotheWasher        168 non-null    float64
 20  ClothesDryer        168 non-null    float64
 21  WaterDispenser      168 non-null    float64
 22  TotalUsagekWh       168 non-null    float64
dtypes: float64(19), int32(1), int64(1), object(2)
memory usage: 29.7+ KB
None
----------------------------------------------------------------------------------------------------------
```

Figure 18: Dataset information results

Data Analysis

Analysis of data is the process of evaluating and extracting meaningful insights, associations, and patterns to improve decision-making and efficiency. During this process, data is collected, cleaned, analyzed, and interpreted. Challa (2023) stated that market research shows the data analytics market was valued at USD 41.39 billion in 2022 and is expected to reach USD 346.33 billion by 2030. Moreover, Challa (2023) suggests that data analytics can help businesses streamline operations, increase efficiency, and decrease costs by gaining a better understanding of audience preferences. The following are six data analysis types: -

- **Descriptive Analysis**

A descriptive statistic summarizes and organizes the dataset's characteristics. The three types of descriptive statistics are Distribution, which measures the frequency of values, Central tendency, which calculates the mean, median and mode and Variability which describes how widely the values are spread out (Bhandari, 2020).

- **Inferential Analysis**

Inferential statistics involves interpreting sample data to predict the behaviour of a larger population based on sample data. Researchers may generalize their findings beyond the studied sample by employing statistical methods and probability theory. Analyzing data, making predictions, and drawing meaningful conclusions requires techniques such as correlations and regression analysis. One of the key methods is regression analysis, which examines relationships between variables and predicts results (Emmanuel, 2023).

- **Time Series Analysis**

A time series analysis examines data points collected consistently to identify patterns, trends, and relationships. Variables are analyzed over time to reveal underlying patterns, predict future values, and understand how they behave (Pandian, 2021).

- **Comparative Analysis**

Comparative analysis is a systematic way of identifying similarities, differences, and patterns between two or more variables. It involves assessing each data's strengths, weaknesses, opportunities, and threats (Appinio, 2023).

- **Clustering Analysis**

The cluster analysis method groups items into clusters based on their similarities. This method can be used when relationships within the data are unknown and does not require knowing the number of clusters beforehand. A cluster is formed when items within a cluster are more similar than those within other clusters (Webster, 2020).

- **Classification Analysis**

The classification analysis technique categorizes data based on shared characteristics into diverse groups. The two types of classification analysis are supervised learning, in which algorithms are trained using labelled data, and unsupervised learning, in which clustering is used to discover patterns without labelled data (Samarth, 2023).

Data Analysis and Visualization Before Optimization

1. Descriptive Analysis before optimization

- **Summary statistics**

```
# Descriptive Analysis: -

# Get summary statistics of the numerical columns : mean, median, and standard deviation
print("Summary Statistics:")
print(dt.describe())
```

Figure 19: Summary statistics before optimization code

```
Summary Statistics:
             Time   Occupancy  OutdoorTemperature         AC1         AC2  \
count  168.000000  168.000000          168.000000  168.000000  168.000000
mean    11.500000    3.767857           30.901190    0.789286    0.152500
std      6.942881    0.773616            1.916552    1.439770    0.507297
min      0.000000    1.000000           27.000000    0.000000    0.000000
25%      5.750000    3.000000           29.000000    0.000000    0.000000
50%     11.500000    4.000000           31.000000    0.000000    0.000000
75%     17.250000    4.000000           33.000000    0.000000    0.000000
max     23.000000    6.000000           35.000000    3.400000    1.830000

              AC3         AC4         AC5         AC6         AC7  ...  \
count  168.000000  168.000000  168.000000  168.000000  168.000000  ...
mean     1.578571    0.261905    0.671786    0.160952    1.375476  ...
std      1.700727    0.676714    0.940267    0.557427    1.014991  ...
min      0.000000    0.000000    0.000000    0.000000    0.000000  ...
25%      0.000000    0.000000    0.000000    0.000000    0.000000  ...
50%      0.000000    0.000000    0.000000    0.000000    2.120000  ...
75%      3.400000    0.000000    1.980000    0.000000    2.120000  ...
max      3.400000    2.000000    1.980000    2.080000    2.120000  ...

               TV  Refrigeter1  Refrigeter2   Refrigeter3       Freezer  \
count  168.000000        168.0        168.0  1.680000e+02  1.680000e+02
mean     0.144643          0.5          0.5  3.000000e-01  3.540000e-02
std      0.135058          0.0          0.0  7.238024e-16  7.655602e-17
min      0.000000          0.5          0.5  3.000000e-01  3.540000e-02
25%      0.000000          0.5          0.5  3.000000e-01  3.540000e-02
50%      0.270000          0.5          0.5  3.000000e-01  3.540000e-02
75%      0.270000          0.5          0.5  3.000000e-01  3.540000e-02
max      0.270000          0.5          0.5  3.000000e-01  3.540000e-02

       VacuumCleaner  ClotheWasher  ClothesDryer  WaterDispenser  \
count     168.000000    168.000000    168.000000    1.680000e+02
mean        0.035714      0.014643      0.080357    9.400000e-01
std         0.186132      0.076314      0.460182    1.558959e-15
min         0.000000      0.000000      0.000000    9.400000e-01
25%         0.000000      0.000000      0.000000    9.400000e-01
50%         0.000000      0.000000      0.000000    9.400000e-01
75%         0.000000      0.000000      0.000000    9.400000e-01
max         1.000000      0.410000      2.700000    9.400000e-01

       TotalUsagekWh
count     168.000000
mean        8.601233
std         2.822394
min         2.545400
25%         6.645400
50%         8.270400
75%         9.937900
max        15.445400

[8 rows x 21 columns]
```

Figure 20: Summary statistics before optimization results

According to the summary statistics, the average household occupancy is between three and four people, while outdoor temperatures range between 27°C and 35°C, indicating a warm to hot climate. The use of appliances like refrigerators, freezers and water dispensers appears to be stable since they are constantly running. The overall energy consumption varies significantly, with an average of 8.6 kWh per hour and peaks of 15.4 kWh at certain times, likely caused by high-consuming appliances.

- **Frequency distribution for Occupancy attribute**

```
# Frequency distribution for Occupancy
occupancy_counts = dt['Occupancy'].value_counts()
print("Occupancy Frequency Distribution:")
print(occupancy_counts)
```

Figure 21: Frequency distribution code

```
----------------------------------------------------------------------------------------------------------
Occupancy Frequency Distribution:
4    105
3     34
5     17
2      9
1      2
6      1
Name: Occupancy, dtype: int64
```

Figure 22: Frequency distribution before optimization results

According to the occupancy frequency distribution result, the most common occupancy level is 4 people with 105 occurrences, followed by 3 people with 34 occurrences. Other occupancy levels, however, occur less frequently. The household typically has 3 to 4 occupants.

- **Distribution using box plot for Occupancy, Outdoor temperature and total energy usage.**

```
# Box plot
# Create a figure and subplots with 1 row and 3 columns for displaying multiple box plots side by side
fig, ax = plt.subplots(1, 3, figsize=(15, 6))

# Adjust the spacing between the subplots for clarity and separation
plt.subplots_adjust(wspace=0.5)

# Plot a boxplot for the 'Occupancy' column in the first subplot, setting the color to brown
sns.boxplot(data=dt['Occupancy'], ax=ax[0], color='brown')
ax[0].set_title('Occupancy')  # Set the title for the first subplot
ax[0].set_xlabel('Occupancy')  # Set the label for the x-axis of the first subplot

# Plot a boxplot for 'OutdoorTemperature' in the second subplot, setting the color to red
sns.boxplot(data=dt['OutdoorTemperature'], ax=ax[1], color='Red')
ax[1].set_title('Outdoor Temperature')  # Set the title for the second subplot
ax[1].set_xlabel('Outdoor Temperature')  # Set the label for the x-axis of the second subplot

# Plot a boxplot for 'TotalUsagekWh' in the third subplot, setting the color to green
sns.boxplot(data=dt['TotalUsagekWh'], ax=ax[2], color='Green')
ax[2].set_title('Total Usage (kWh)')  # Set the title for the third subplot
ax[2].set_xlabel('Total Usage (kWh)')  # Set the label for the x-axis of the third subplot

# Display the created box plots on the screen
plt.show()
```

Figure 23: Box plot distribution before optimization code

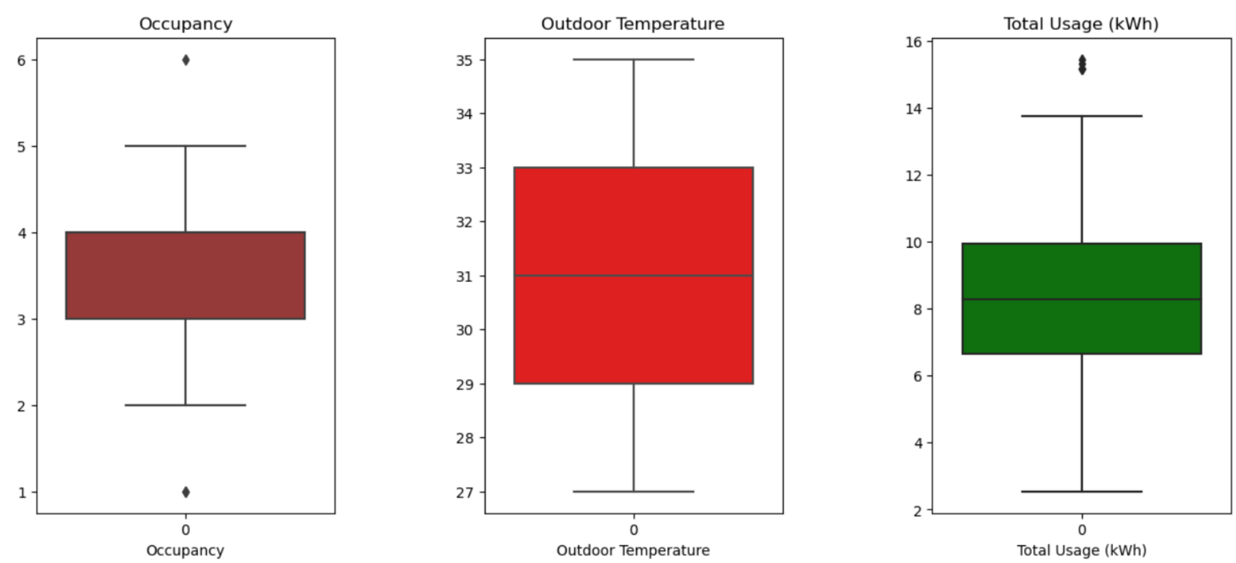


Figure 24: Box plot distribution before optimization result

According to the above figure, the occupancy box plot shows that the median occupancy is around 4. The Occupancy most data ranges from 3 to 4 and the minimum and maximum values are 2 and 5, respectively. Two outliers are present at 1 and 6, suggesting abnormal occupancy counts. Moreover, the Outdoor Temperature box plot shows a median of 31°C, with most temperatures ranging between 29°C and 33°C. The temperature patterns appear stable, with no significant outliers. Finally, based on the Total Usage (kWh) box plot, the median is 8 kWh, with data falling between 6 kWh and 10 kWh. Nevertheless, there are outliers above 14 kWh, showing that some instances consumed significantly more energy.

- **Distribution using scatter plot for Occupancy, Outdoor temperature and time regarding total energy usage.**

```
# Create a pair plot to visualize the relationship between 'Occupancy' and 'OutdoorTemperature'
# with respect to 'TotalUsagekWh'. This scatter plot will help to identify any trends or
# correlations between the number of occupants and the outdoor temperature on energy consumption.

sns.pairplot(dt, x_vars=['Occupancy','OutdoorTemperature','Time'], y_vars='TotalUsagekWh', height = 4, kind = 'scatter')
plt.show()
```

Figure 25: Scatter plot distribution before optimization code

The three scatter plots show the relationship of **total energy usage in kWh with three different factors:**

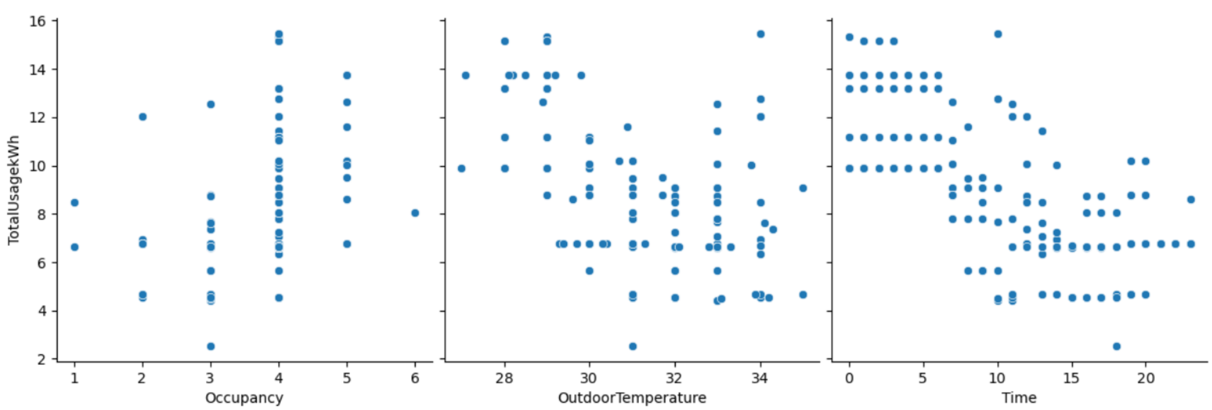


Figure 26: Scatter plot distribution before optimization result

The relationship with occupancy shows that energy usage varies by occupancy level, with most data points clustered around four occupants. However, with the Outdoor Temperature, it displays slightly higher usage around 30°C to 34°C. With the Time plot, energy usage is higher in the early hours, close to zero (Midnight) and decreases over time. In this case, peak energy consumption occurs earlier in the day and decreases later.

- **Minimum and maximum usage details**

```
# Show minimum Energy Usage details

# Find the row with minimum energy usage
min_usage_row = dt.loc[dt['TotalUsagekWh'].idxmin()]

# Print the details for the minimum energy usage
print("Minimum Energy Usage:")
print(f"The hour: {min_usage_row['Time']}")
print(f"Day: {min_usage_row['Date']}")
print(f"Total Usage (kWh): {min_usage_row['TotalUsagekWh']}")
print(f"Temperature: {min_usage_row['OutdoorTemperature']}")
print(f"Occupancy: {min_usage_row['Occupancy']}")

# Minimum Energy Usage: The lowest energy consumption was recorded in the evening at 6:00 PM on October 15th,
# under warm conditions (31°C) with 3 people present.

print('----------------------------------------------------------------------------------------------------------')


# Show maximum Energy Usage details

# Find the row with maximum energy usage
max_usage_row = dt.loc[dt['TotalUsagekWh'].idxmax()]

# Print the details for the maximum energy usage
print("\nMaximum Energy Usage:")
print(f"The hour: {max_usage_row['Time']}")
print(f"Day: {max_usage_row['Date']}")
print(f"Total Usage (kWh): {max_usage_row['TotalUsagekWh']}")
print(f"Temperature: {max_usage_row['OutdoorTemperature']}")
print(f"Occupancy: {max_usage_row['Occupancy']}")

# Maximum Energy Usage: The highest energy consumption occurred in the late morning at 10:00 AM on October 18th,
# with a higher temperature of 34°C and 4 people present.
```

Figure 27: Minimum and maximum usage before optimization code

```
Minimum Energy Usage:
The hour: 18
Day: 15/10/2024
Total Usage (kWh): 2.5454
Temperature: 31.0
Occupancy: 3
----------------------------------------------------------------------------------------------------------

Maximum Energy Usage:
The hour: 10
Day: 18/10/2024
Total Usage (kWh): 15.445400000000001
Temperature: 34.0
Occupancy: 4
```

Figure 28: Minimum and maximum usage before optimization result

The **minimum** energy consumption was 2.5454 kWh, recorded at 18:00 (6 p.m.) on Tuesday, 15/10/2024, at a temperature of 31.0°C with 3 people present. Based on these results, it seems that energy consumption was low during the evenings at a moderate temperature with a typical occupancy level.

On the other hand, the **maximum** energy consumption was 15.4454 kWh, recorded at 10:00 a.m. on Friday, 18/10/2024, with a temperature of 34.0°C and four people inside the house. There was a spike in energy consumption at the weekends during the morning hours, possibly due to increased cooling or appliance usage.

2. Inferential Analysis before optimization

- **Correlation analysis**

```python
# Inferential Analysis

#Correlation Analysis

#Heatmap for correlations matrix
plt.figure(figsize=(26, 16))  # Increase width and height for better visibility

# Create a heatmap using the correlation matrix
sns.heatmap(dt.corr(), annot=True, cmap='RdBu', linewidths=1, linecolor='Brown', vmin=-1, vmax=1)

# Display the heatmap
plt.show()
```

Figure 29: Correlation matrix before optimization code

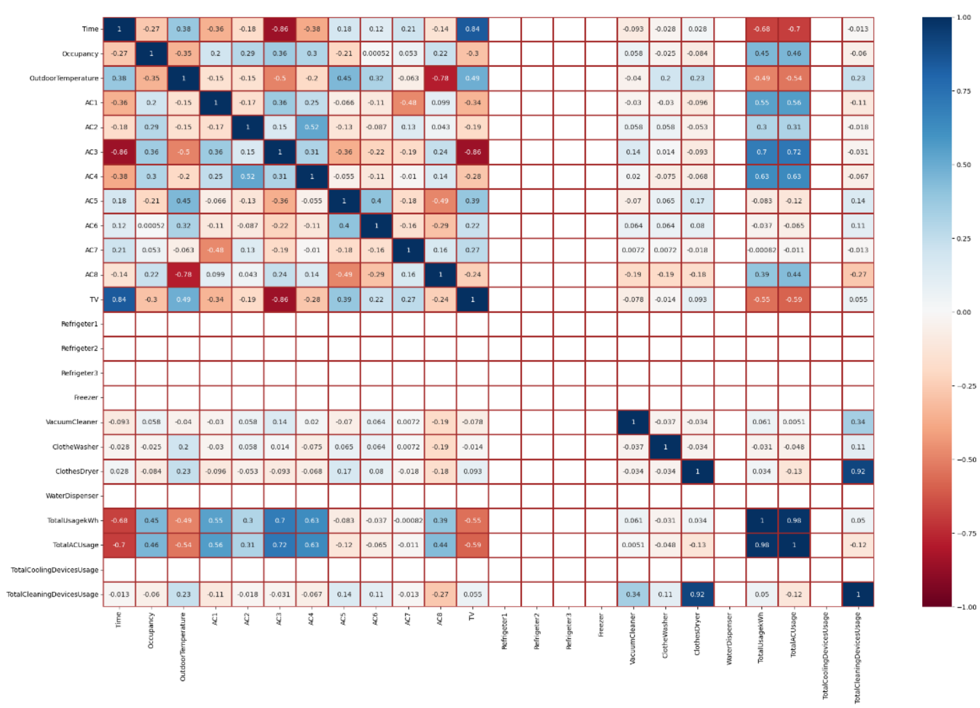


Figure 30: Correlation matrix before optimization result

Correlations equal and above 0.5: -

- **Time and TV:** 0.84.
- **AC1 and TotalUsagekWh:** 0.55.
- **AC2 and AC4:** 0.52.
- **AC3 and TotalUsagekWh:** 0.7.
- **AC4 and TotalUsagekWh:** 0.63.

Correlations equal and below -0.5: -

- **Time and TotalUsagekWh:** -0.68.
- **Time and AC3:** -0.86.
- **OutdoorTemperature and AC3:** -0.5.
- **OutdoorTemperature and AC8:** -0.78.
- **AC3 and TV:** -0.86.
- **TV and TotalUsagekWh:** -0.55.

Correlations that have empty fields because they are constants: -

- **Refrigeter1.**
- **Refrigeter2.**
- **Refrigeter3.**
- **Freezer.**
- **WaterDispenser.**

Based on the correlation analysis that shows the strength and direction of the relationship, TV and time have a strong correlation of 0.84, suggesting that TV usage increases over time, as well as between AC devices like AC1, AC3, AC4 and TotalUsageKWh, indicating that these appliances contribute significantly to the overall energy consumption. In contrast, negative correlations can be found between Time and TotalUsageKWh with (-0.68), Time and AC3 with (-0.86), and TV and TotalUsageKWh with (-0.55), indicating that energy usage decreases as time passes, especially for AC3 and TV. Negative correlation between OutdoorTemperature and AC8 with (-0.78), indicating AC3 usage is inversely related to temperature. Lastly, appliances such as the Refrigerator1, Refrigerator2, Refrigerator3, Freezer, and WaterDispenser do not show any correlation with other variables due to their constant operation.

- **Regression analysis**

Regression analysis between TV and Time: -

- **Independent Variable:** Time.
- **Dependent Variable:** TV.

```python
# Regression model measure relationships between variables
# Highest Positive Relationships: -
# x Independent variable
x = dt['Time']
# y Dependent variable
y = dt['TV']

# Split the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, test_size=0.3, random_state=100)

# Add a constant to the model (intercept)
x_train = sm.add_constant(x_train)

# Fit the OLS regression model
LR = sm.OLS(y_train, x_train).fit()

# Print the summary of the regression model
print(LR.summary())
```

Figure 31: First regression analysis before optimization code

```
------------------------------------------------------------------------------
                            OLS Regression Results
==============================================================================
Dep. Variable:                     TV   R-squared:                       0.691
Model:                            OLS   Adj. R-squared:                  0.689
Method:                 Least Squares   F-statistic:                     257.4
Date:                Mon, 04 Nov 2024   Prob (F-statistic):           4.02e-31
Time:                        20:46:45   Log-Likelihood:                 137.37
No. Observations:                 117   AIC:                            -270.7
Df Residuals:                     115   BIC:                            -265.2
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -0.0431      0.014     -3.154      0.002      -0.070      -0.016
Time           0.0162      0.001     16.045      0.000       0.014       0.018
==============================================================================
Omnibus:                        2.114   Durbin-Watson:                   2.197
Prob(Omnibus):                  0.347   Jarque-Bera (JB):                1.640
Skew:                           0.111   Prob(JB):                        0.440
Kurtosis:                       2.464   Cond. No.                         26.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Figure 32: First regression analysis before optimization result

The first regression analysis indicates a strong relationship between Time and TV, with an R-squared of 0.691. A positive coefficient for Time (0.0162) suggests that TV watching increases over time. The statistical significance of Time as a predictor is confirmed by a low p-value (0.000) and an adjusted R-squared of 0.689.

Regression analysis between AC3 and Time: -

- **Independent Variable:** Time.
- **Dependent Variable:** AC3.

```
# Highest Negative Relationships:

# x Independent variable
x = dt['Time']
# y Dependent variable
y = dt['AC3']

# Split the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, test_size=0.3, random_state=100)

# Add a constant to the model (intercept)
x_train = sm.add_constant(x_train)

# Fit the OLS regression model
LR = sm.OLS(y_train, x_train).fit()

# Print the summary of the regression model
print(LR.summary())
```

Figure 33: Second regression analysis before optimization code

```
------------------------------------------------------------------------------------------------
                            OLS Regression Results
==============================================================================
Dep. Variable:                    AC3   R-squared:                       0.721
Model:                            OLS   Adj. R-squared:                  0.719
Method:                 Least Squares   F-statistic:                     297.7
Date:                Mon, 04 Nov 2024   Prob (F-statistic):           1.07e-33
Time:                        20:46:45   Log-Likelihood:                -153.13
No. Observations:                 117   AIC:                             310.3
Df Residuals:                     115   BIC:                             315.8
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          4.0257      0.164     24.606      0.000       3.702       4.350
Time          -0.2088      0.012    -17.255      0.000      -0.233      -0.185
==============================================================================
Omnibus:                        7.891   Durbin-Watson:                   2.368
Prob(Omnibus):                  0.019   Jarque-Bera (JB):                3.617
Skew:                          -0.149   Prob(JB):                        0.164
Kurtosis:                       2.192   Cond. No.                         26.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Figure 34: Second regression analysis before optimization result

In the second regression analysis, Time has a significant negative relationship with AC3 consumption, with an R-squared value of 0.721. Based on the time coefficient (-0.2088), AC3 consumption decreases over time. Statistical significance is confirmed by a low p-value (0.000) and a high F-statistic (297.7), indicating the reliability of Time as a predictor of AC3 consumption.

Summary of the two regression analysis results

Both regressions indicate strong relationships between Time and the independent variables. In the first model, increased TV usage is associated with increased Time, while in the second model, increased AC3 usage is associated with decreased Time. R-squared values and p-values indicate both models are statistically valid and robust.

- **Covariance analysis**

```
# Calculate the covariance matrix
cov_matrix = dt.cov()
# Create a custom diverging color palette for blue positive and red negative
cmap = sns.diverging_palette(10, 240, as_cmap=True)  # This reverses the colors to make positive blue and negative red
# Set the figure size for the plot to ensure readability
plt.figure(figsize=(20, 17))
# Create a heatmap for visualizing the covariance matrix
sns.heatmap(
    cov_matrix,  # The covariance matrix data
    annot=True,  # Display the covariance values in the heatmap cells
    cmap=cmap,  # Use the modified color map for positive (blue) and negative (red)
    linewidths=1,  # Add lines between cells for better separation
    fmt=".2f",  # Format the displayed values to show 2 decimal places
    center=0,  # Center the color map at zero, so 0.0 appears white
    cbar_kws={"shrink": 0.5}  # Shrink the color bar to fit the plot better
)

# Add a title to the heatmap for context
plt.title('Covariance Matrix Heatmap')
# Display the heatmap plot
plt.show()
```

Figure 35: Covariance matrix before optimization code

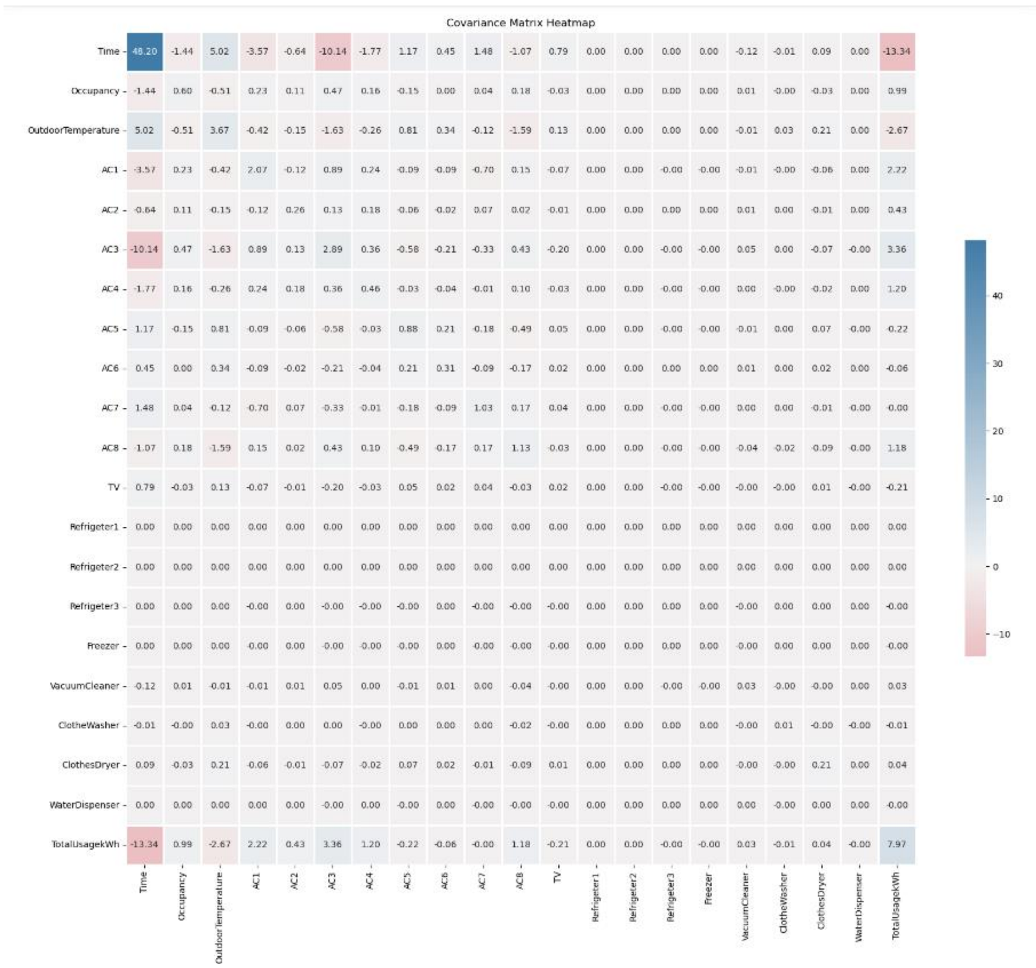


Figure 36: Covariance matrix before optimization result

The covariance matrix shows whether two variables move together positively or negatively. Covariance in blue indicates that variables increase together, while covariance in red indicates that one variable increases as the other decreases. For example, Time and OutdoorTemperature have a covariance of 5.02, showing a direct positive relationship between the two variables. As time progresses, the Outdoortemperature tends to increase. On the contrary, Time and TotalUsageKWh show a negative relationship, with a covariance of -13.34. In other words, total energy usage tends to decrease over time.

3. Time Series Analysis before optimization

```
#Time Series Analysis
# Plot total usage over time to identify trends and patterns.
# Create 'Datetime' column and set it as the index
dt['Datetime'] = pd.to_datetime(dt['Date'] + ' ' + dt['Time'].astype(str) + ':00:00', dayfirst=True)
dt.set_index('Datetime', inplace=True)

# Get unique days and generate a color map for distinct visualization
unique_days = dt['Date'].unique()
colors = cm.rainbow(np.linspace(0, 1, len(unique_days)))  # Create distinct colors for each day

plt.figure(figsize=(14, 7))

# Plot each day's data with a unique color for the lines and points
for i, day in enumerate(unique_days):
    daily_data = dt[dt['Date'] == day]  # Extract data for the specific day
    day_name = pd.to_datetime(day, dayfirst=True).strftime('%A')  # Convert to day name (e.g., 'Sunday')
    plt.plot(daily_data.index, daily_data['TotalUsagekWh'], marker='o', color=colors[i], label=day_name, linestyle='-')

    # If not the last day, connect the end of this day's line to the start of the next with a solid black line
    if i < len(unique_days) - 1:
        next_day_data = dt[dt['Date'] == unique_days[i + 1]]
        if not daily_data.empty and not next_day_data.empty:
            plt.plot(
                [daily_data.index[-1], next_day_data.index[0]],
                [daily_data['TotalUsagekWh'].iloc[-1], next_day_data['TotalUsagekWh'].iloc[0]],
                color='black', linestyle='-'  # Solid black line connecting two different days
            )

# Add plot title and labels for better context
plt.title('Time Series Analysis of Total Energy Usage (kWh) for Each Day')
plt.xlabel('Time')
plt.ylabel('Total Usage (kWh)')
plt.legend(loc='upper right', bbox_to_anchor=(1.15, 1))  # Adjust legend position
plt.grid(True)  # Add grid lines for readability
plt.show()  # Display the plot
```

Figure 37: Time series analysis before optimization code

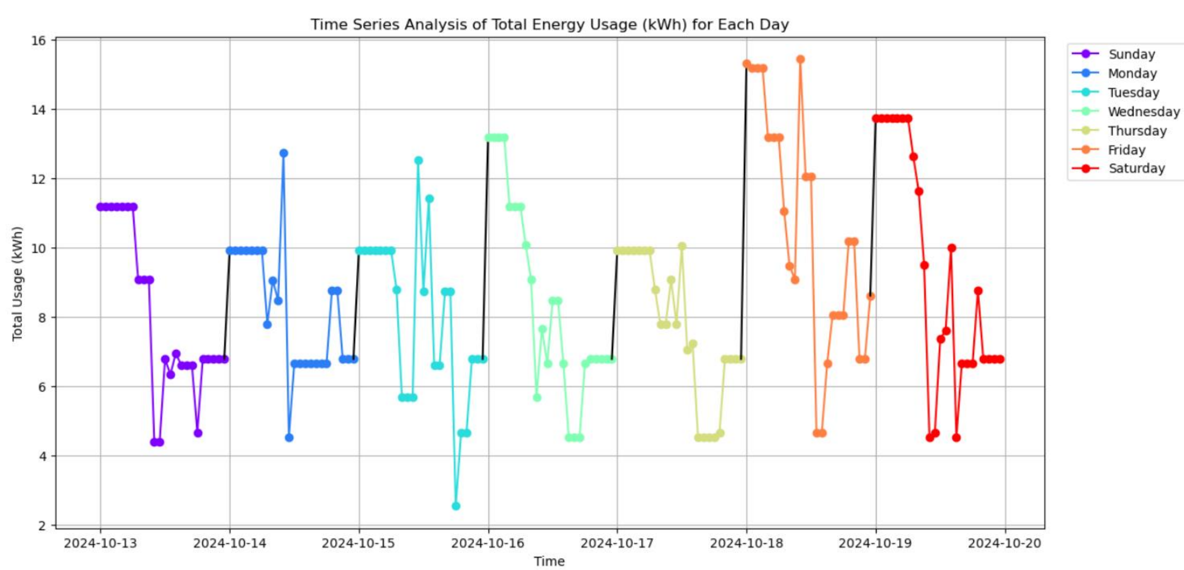


Figure 38: Time series analysis before optimization result

The time series analysis, from October 13 to October 20, 2024, reveals distinct daily patterns in energy usage. However, it generally rises in the morning, falls in the middle of the day, and rises again in the evening. Friday has the highest energy consumption, whereas Tuesday has the lowest. Because Friday and Saturday are the weekends in the Kingdom of Bahrain, people spend more time at home, resulting in higher energy consumption. People usually work on other days, which results in lower electricity consumption.

4. Comparative Analysis before optimization

```
# Energy Usage by Appliance Analysis

# List of appliances in your dataset
appliances = [
    "AC1", "AC2", "AC3", "AC4", "AC5", "AC6", "AC7", "AC8", 'Refrigeter1', 'Refrigeter2', 'Refrigeter3', 'Freezer',
    "TV", "VacuumCleaner", "ClotheWasher", "ClothesDryer", "WaterDispenser"
]

# Calculate total energy usage for each appliance
total_usage_per_appliance = dt[appliances].sum()

# Calculate percentage contribution for each appliance
percentage_contribution = (total_usage_per_appliance / total_usage_per_appliance.sum()) * 100

# Plot total energy usage per appliance
plt.figure(figsize=(12, 8))
total_usage_per_appliance.plot(kind='bar', color='red')
plt.title('Total Energy Usage per Appliance')
plt.xlabel('Appliance')
plt.ylabel('Total Usage (kWh)')
plt.xticks(rotation=45, ha='right')  # Rotate x labels for better readability
plt.show()
```

Figure 39: Energy usage by appliance analysis before optimization code

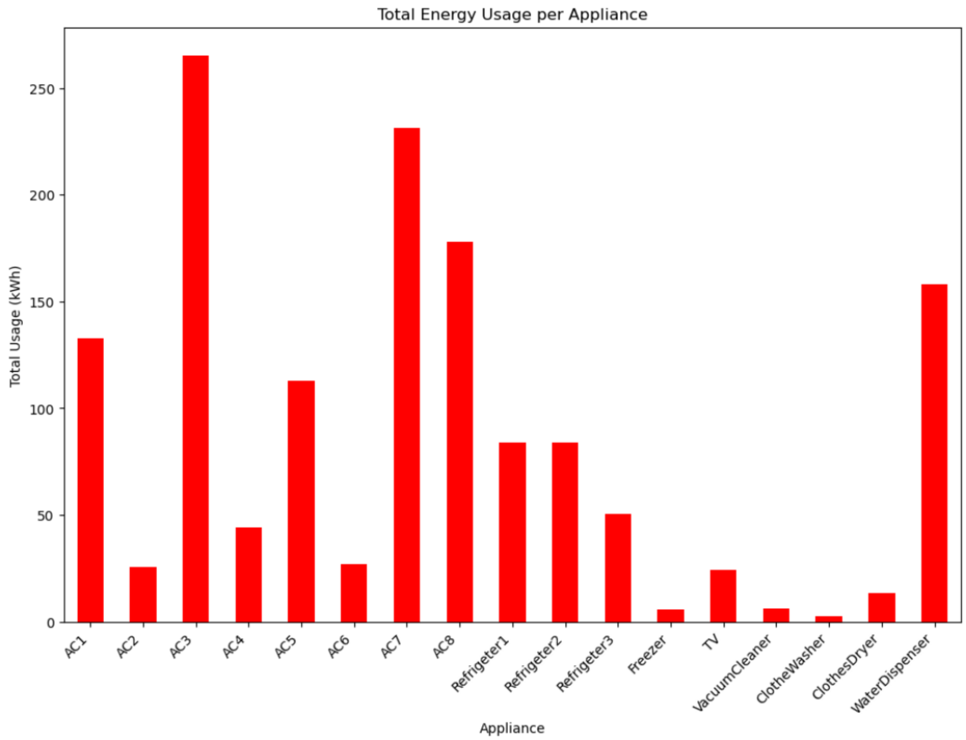


Figure 40: Energy usage by appliance analysis before optimization result

The bar chart compares the total energy usage in kWh for various appliances. Generally, air conditioners use the most electricity, as the highest energy-consuming appliance is AC3, followed by AC7. Appliances such as refrigeter1 and refrigeter2 show moderate energy consumption levels, whereas the freezer, vacuum cleaner, clothes washer, and clothes dryer consume minimal energy.

5. Clustering Analysis before optimization

```python
# Identify clusters
# Create new features by aggregating device usage into broader categories
dt['TotalACUsage'] = (
    dt['AC1'] + dt['AC2'] + dt['AC3'] + dt['AC4'] +
    dt['AC5'] + dt['AC6'] + dt['AC7'] + dt['AC8']
)

# Group refrigerators, freezers, and water dispensers into CoolingDevices
dt['TotalCoolingDevicesUsage'] = (
    dt['Refrigeter1'] + dt['Refrigeter2'] + dt['Refrigeter3'] +
    dt['Freezer'] + dt['WaterDispenser']
)

# Group cleaning devices together
dt['TotalCleaningDevicesUsage'] = (
    dt['VacuumCleaner'] + dt['ClotheWasher'] +
    dt['ClothesDryer']
)

# Define the features for clustering
features = ['TotalACUsage', 'TotalCoolingDevicesUsage', 'TotalCleaningDevicesUsage']
dt_selected = dt[features].dropna()  # Remove any rows with missing values for clustering

# Applying K-Means Clustering
kmeans = KMeans(n_clusters=3, random_state=42)  # Adjust number of clusters as needed
dt_selected['Cluster'] = kmeans.fit_predict(dt_selected)

# Aggregating usage for each device group by hour
hourly_ac_usage = dt.groupby('Time')['TotalACUsage'].sum()
hourly_cooling_usage = dt.groupby('Time')['TotalCoolingDevicesUsage'].sum()
hourly_cleaning_usage = dt.groupby('Time')['TotalCleaningDevicesUsage'].sum()

# Plotting hourly usage for comparison
plt.figure(figsize=(10, 6))
plt.plot(hourly_ac_usage.index, hourly_ac_usage, label='AC Devices', marker='o', color='blue')
plt.plot(hourly_cooling_usage.index, hourly_cooling_usage, label='Cooling Devices', marker='o', color='green')
plt.plot(hourly_cleaning_usage.index, hourly_cleaning_usage, label='Cleaning Devices', marker='o', color='red')

# Labeling
plt.xlabel('Hour of the Day')
plt.ylabel('Total Usage (kWh)')
plt.title('Hourly Energy Usage Patterns by Device Category')
plt.xticks(range(0, 24, 1))  # Show all hours on the x-axis
plt.legend()
plt.grid(True)
plt.show()
```

Figure 41: Clustering analysis before optimization code

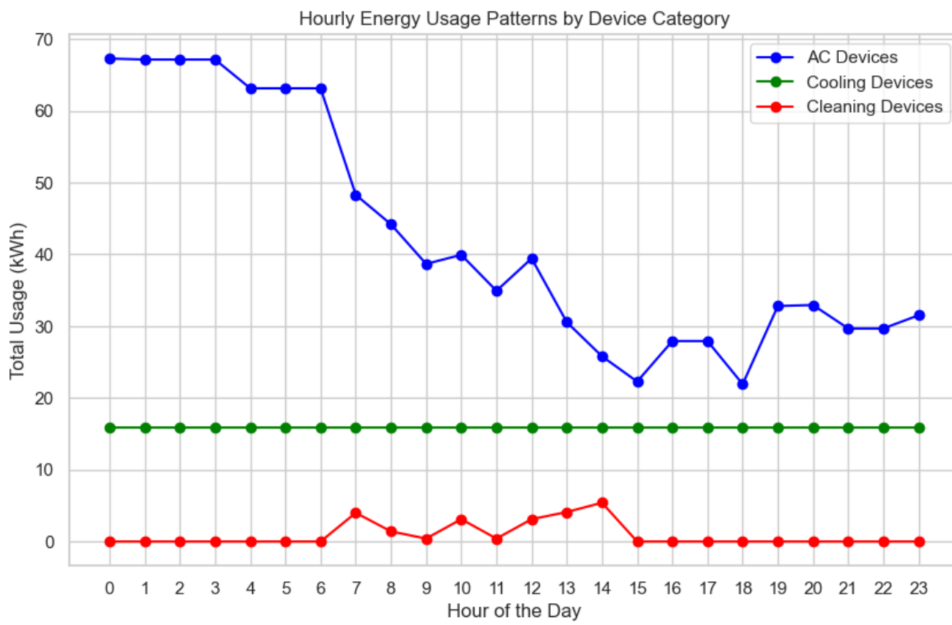


Figure 42: Clustering analysis before optimization result

The line chart shows the hourly energy consumption for three groups: AC, cooling, and cleaning devices. All air conditioners are included in the AC devices group, cooling devices include refrigerators, freezers, and water dispensers, while cleaning devices include vacuum cleaners, clothes washers, and dryer washers. AC devices are the most energy-intensive, peaking approximately at 68 kWh during the early morning hours and gradually decreasing throughout the day, with some fluctuation toward the end of the day. A stable consumption of 15 kWh is maintained throughout the day by cooling devices, indicating continuous operation. The lowest usage is seen with cleaning devices, which have a time range that starts after 6 (6 a.m.) and ends before 15 (3 p.m.), with a small peak around 7 (7 a.m.) and 14 (2 p.m.).

6. Classification Analysis before optimization

- **Classification Report**

```
# Classification Analysis:
# Categorize 'TotalUsageKWh' into 'Low', 'Medium', and 'High' usage levels.
dt['UsageCategory'] = pd.cut(dt['TotalUsageKWh'], bins=[0, 5, 10, dt['TotalUsageKWh'].max()],
                             labels=['Low', 'Medium', 'High'])

# Prepare the feature set (X) with appliance data and relevant variables.
X = dt[['Time', 'OutdoorTemperature', 'AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8',
        'TV', 'Refrigeter1', 'Refrigeter2', 'Refrigeter3', 'Freezer',
        'VacuumCleaner', 'ClotheWasher', 'ClothesDryer', 'WaterDispenser']]
# Define the target variable (y) as the categorized energy usage.
y = dt['UsageCategory']

# Split the dataset into training and testing sets (70% train, 30% test).
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Initialize the Decision Tree Classifier for classification.
classifier = DecisionTreeClassifier(random_state=42)

# Set up a hyperparameter grid for tuning the Decision Tree Classifier using GridSearchCV.
param_grid = {
    'max_depth': [None, 5, 10, 15],  # Varying tree depth for complexity control.
    'min_samples_split': [2, 5, 10],  # Minimum samples required to split a node.
    'min_samples_leaf': [1, 2, 5]  # Minimum samples required for a leaf node.
}

# Perform GridSearchCV for the best combination of hyperparameters.
grid_search = GridSearchCV(estimator=classifier, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# Display the optimal hyperparameters found.
print("Best parameters found: ", grid_search.best_params_)

# Use the best estimator from GridSearchCV to make predictions.
best_classifier = grid_search.best_estimator_
y_pred = best_classifier.predict(X_test)

# Print the classification report summarizing precision, recall, and F1-score.
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

Figure 43: Classification report before optimization code

```
Best parameters found:  {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2}

Classification Report:
              precision    recall  f1-score   support

        High       0.70      0.88      0.78         8
         Low       1.00      0.83      0.91         6
      Medium       0.94      0.92      0.93        37

    accuracy                           0.90        51
   macro avg       0.88      0.88      0.87        51
weighted avg       0.91      0.90      0.90        51
```

Figure 44: Classification report before optimization result

The classification report shows that the model has an overall high performance with a 90% accuracy rate. Notably, the precision, recall, and F1 scores are high in the Medium category, indicating consistency. In the Low category, precision was perfect, whereas in the High category, precision was seventy per cent, but recall was higher.

- **Confusion Matrix**

```
# Create and display a confusion matrix to visualize prediction results.
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, cmap='Blues', fmt='d',
            xticklabels=['Low', 'Medium', 'High'], yticklabels=['Low', 'Medium', 'High'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

Figure 45: Confusion matrix before optimization code

Diagonal cells show the number of instances that were correctly predicted from top-left to bottom-right. In contrast, other cells are incorrectly classified. Shades of blue indicate how many instances regardless of correct or incorrect classification and a darker blue indicates a greater number of instances.

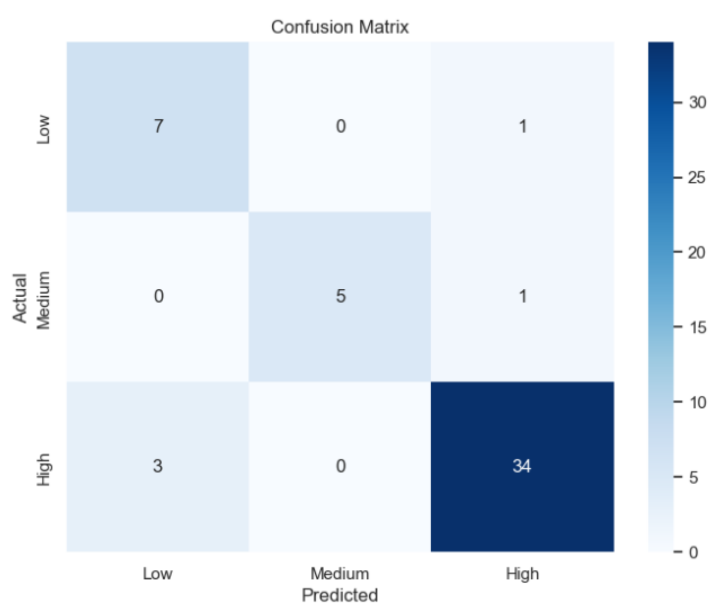


Figure 46: Confusion matrix before optimization result

The confusion matrix shows that the model correctly classified most instances, with seven correct predictions for Low, five for Medium, and 34 for High. Nevertheless, one Low and Medium instance was incorrectly classified as High, and 3 High instances were incorrectly classified as Low.

Data Summarization Before Optimization

```
# Data summarization before optimization
# Loop through each specified appliance to determine daily usage patterns
for appliance in ['AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8', 'TV',
                  'VacuumCleaner', 'ClotheWasher', 'ClothesDryer']:
    print(f"\n{appliance}:")  # Print the appliance name to start its usage summary

    # Iterate through each unique date in the dataset to check appliance usage on each day
    for date in dt['Date'].unique():
        # Get a list of the specific hours when the appliance was running on the current date
        hours_running = dt[(dt['Date'] == date) & (dt[appliance] > 0)]['Time'].tolist()

        # If the appliance was running at any hour(s) on this date, print those hours
        if hours_running:
            print(f"  {date} - Run at hour(s): {', '.join(map(str, hours_running))}")
        else:
            # If the appliance did not run on this date, indicate it was "Not running"
            print(f"  {date} - Not running")

# Calculate and display the total energy usage for each day
# Group data by 'Date' and calculate the sum of 'TotalUsagekWh' for each day to get daily usage totals
total_usage_per_day = dt.groupby('Date')['TotalUsagekWh'].sum()

print("\nTotal Energy Usage for Each Day:")
# Iterate over the calculated daily usage totals and print each date’s total energy consumption
for date, total_usage in total_usage_per_day.items():
    print(f"Date: {date}, Total Usage: {total_usage:.2f} kWh")  # Display usage to two decimal places for clarity

# Calculate and display the total energy usage across all days
# Sum up the total daily energy usage to get a cumulative total for the entire period
total_usage_all_days = total_usage_per_day.sum()
print(f"\nTotal Energy Usage Across All Days: {total_usage_all_days:.2f} kWh")  # Display grand total energy usage
```

Figure 47: Data summarization before optimization code


```
AC1:
  13/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  14/10/2024 - Not running
  15/10/2024 - Not running
  16/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8
  17/10/2024 - Run at hour(s): 10
  18/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 16, 17, 18, 19, 20
  19/10/2024 - Run at hour(s): 14

AC2:
  13/10/2024 - Not running
  14/10/2024 - Not running
  15/10/2024 - Not running
  16/10/2024 - Run at hour(s): 12, 13
  17/10/2024 - Not running
  18/10/2024 - Run at hour(s): 23
  19/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

AC3:
  13/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
  14/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
  15/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11
  16/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
  17/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
  18/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
  19/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

AC4:
  13/10/2024 - Not running
  14/10/2024 - Not running
  15/10/2024 - Not running
  16/10/2024 - Run at hour(s): 0, 1, 2, 3
  17/10/2024 - Not running
  18/10/2024 - Run at hour(s): 0, 1, 2, 3, 10, 11, 12, 15
  19/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

AC5:
  13/10/2024 - Run at hour(s): 12, 13, 14, 15, 16, 17
  14/10/2024 - Run at hour(s): 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
  15/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17
  16/10/2024 - Run at hour(s): 10, 11, 12, 13, 14, 15, 16, 17, 18
  17/10/2024 - Run at hour(s): 12, 13, 14, 15, 16, 17, 18
  18/10/2024 - Run at hour(s): 1, 2, 3, 4, 5, 6, 7, 10, 11, 12
  19/10/2024 - Run at hour(s): 13, 14, 15, 16, 17, 18, 19

AC6:
  13/10/2024 - Run at hour(s): 13, 15, 16, 17
  14/10/2024 - Not running
  15/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17
  16/10/2024 - Not running
  17/10/2024 - Not running
  18/10/2024 - Not running
  19/10/2024 - Run at hour(s): 13, 14

AC7:
  13/10/2024 - Run at hour(s): 10, 11, 12, 18, 19, 20, 21, 22, 23
  14/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  15/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 11, 12, 13, 16, 17, 21, 22, 23
  16/10/2024 - Run at hour(s): 11, 12, 13, 14, 18, 19, 20, 21, 22, 23
  17/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 20, 21, 22, 23
  18/10/2024 - Run at hour(s): 0, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  19/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 7, 8, 11, 12, 16, 17, 18, 19, 20, 21, 22, 23

AC8:
  13/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23
  14/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23
  15/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23
  16/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23
  17/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23
  18/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23
  19/10/2024 - Run at hour(s): 0, 1, 2, 3, 4, 5, 6, 19, 20, 21, 22, 23

TV:
  13/10/2024 - Run at hour(s): 15, 16, 17, 18, 19, 20, 21, 22, 23
  14/10/2024 - Run at hour(s): 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  15/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  16/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  17/10/2024 - Run at hour(s): 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  18/10/2024 - Run at hour(s): 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  19/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23

VacuumCleaner:
  13/10/2024 - Not running
  14/10/2024 - Run at hour(s): 8
  15/10/2024 - Run at hour(s): 7
  16/10/2024 - Run at hour(s): 7
  17/10/2024 - Run at hour(s): 7
  18/10/2024 - Not running
  19/10/2024 - Run at hour(s): 7, 13

ClotheWasher:
  13/10/2024 - Run at hour(s): 12
  14/10/2024 - Run at hour(s): 9
  15/10/2024 - Run at hour(s): 11
  16/10/2024 - Not running
  17/10/2024 - Run at hour(s): 13
  18/10/2024 - Run at hour(s): 8
  19/10/2024 - Run at hour(s): 10

ClothesDryer:
  13/10/2024 - Run at hour(s): 14
  14/10/2024 - Run at hour(s): 10
  15/10/2024 - Run at hour(s): 13
  16/10/2024 - Not running
  17/10/2024 - Run at hour(s): 14
  18/10/2024 - Not running
  19/10/2024 - Run at hour(s): 12

Total Energy Usage for Each Day:
Date: 13/10/2024, Total Usage: 192.87 kWh
Date: 14/10/2024, Total Usage: 196.42 kWh
Date: 15/10/2024, Total Usage: 190.81 kWh
Date: 16/10/2024, Total Usage: 203.24 kWh
Date: 17/10/2024, Total Usage: 184.90 kWh
Date: 18/10/2024, Total Usage: 252.27 kWh
Date: 19/10/2024, Total Usage: 224.50 kWh

Total Energy Usage Across All Days: 1445.01 kWh
```

Figure 48: Data summarization before optimization result

A summary of appliance usage time for all days, including ACs, TV, a vacuum cleaner, a clothes washer, and a clothes dryer, excluding devices like refrigerators, freezers, and water dispensers since they operate continuously, but are included in the total appliance usage calculation. The device name, date, status (running or not running), and hours of runtime are displayed. Moreover, the total amount of energy consumed daily is calculated, showing the least energy consumption on Tuesday 15/10/2024, with a total usage of 190.81 kWh as it is in the middle of the working days in Bahrain, and the highest energy consumption on Friday 18/10/2024 which is a weekend in Bahrain, with a total usage of 252.27 kWh. In total, 1445.01 kWh have been consumed over the seven days.

Optimization Technique

The optimization technique is a set of mathematical methods for finding the most suitable approach to a solution without explicitly assessing all methods. It involves complex, repetitive steps that are managed by computers using basic mathematics and iterative calculations. It is categorized into several types such as Linear Programming (LP) and Integer Programming (IP) based on the problem constraints and structure (Ravindran, Ragsdell, & Reklaitis, 2006).

Choosing the right optimization technique requires the following steps: -

- **Define the System Boundaries**: Determine the system's limits.
  - **Constant variables that run all the time without stopping:** “Refrigeter1” is 0.5, “Refrigeter2” is 0.5, “Refrigeter3” is 0.3, “Freezer” is 0.0354 and “WaterDispenser” is 0.94.
  - **Fixed values for the rest of devices:** “AC1” is 3.4, “AC2” is 1.83, “AC3” is 3.4, “AC4” is 2, “AC5” is 1.98, “AC6” is 2.08, “AC7” is 2.12, “AC8” is 2.12, “TV” is 0.27, “VacuumCleaner” is 1, “ClotheWasher” is 0.41 and “ClothesDryer” is 2.7.
  - **Temperature constraint:** All air conditioning devices are turned off if the outdoor temperature is between 25 and 29°C. The dataset was collected in October during the autumn weather in Bahrain which is considered a pleasant weather, as temperatures and humidity gradually decreases after the summer's extreme heat which exceeds 40°C (Weather Atlas, 2024). Thus, it is possible to close the air conditioner in October.
  - **TV constraint:** TV is limited to four hours of use per day. Otherwise, additional hours will be turned off. According to Statista Research Department (2023), in Saudi Arabia, the average time spent watching television was approximately 5.38 hours, while in Kuwait it was approximately 3.2 hours. As Bahrain's society and media are like neighboring Gulf countries, it is reasonable to estimate that residents watch between three to five hours of television daily.
  - **Vacuum cleaner constraint:** Vacuum cleaner will only work three times per week, every Tuesday, Thursday and Saturday at 7 (7 a.m.). As shown in the previous data summarization result, usually the vacuum cleaner runs at 7 a.m. on Tuesday, Thursday and Saturday. Additionally, Liam Cleverdon, Flooring Expert at Flooring King, suggests vacuuming at least twice a week, especially in heavy-traffic rooms (Lawrence, 2024).
  - **Clothes washer and Clothes dryer constraints:** Both devices will only work every Sunday, Tuesday and Thursday. However, Clothes washer will work on 10 (10 a.m.) and Clothes dryer on 12 (12 p.m.) based on the average of the total weekly hours. According to laundry expert Matt O'Connor, "For an average family, two to three times a week is usually sufficient" (Dickson, 2024).

- **Occupancy constraint:** Devices will be closed depending on the occupancy value. If no one is in the house, which is when occupancy = 0, all devices such as ACs, TV, and vacuum cleaners will be turned off except for clothes washers and dryers, which can be turned off automatically by a timer. Moreover, if the occupancy is one, none or one AC is allowed, and the others will be shut off, and if the occupancy is two, none, one or two ACs are allowed, and all others will be turned off, etc.

- **Specify Performance Criteria:** Select a measurable objective for evaluating solutions, such as "Minimize total energy consumption".
- **Identifying Independent Variables**: Determine which variables can be adjusted to influence the performance of the system such as "AC1", "AC2", "AC3", "AC4", "AC5", "AC6" "AC7", "AC8", "TV", "VacuumCleaner", "ClotheWasher" and "ClothesDryer". Moreover, "TotalUsagekWh" will be calculated again for each hour.
- **Create a system model**: Identify how variables relate and influence performance criteria through a mathematical representation: -
  - **Variables: -**
    - $x_{i,t}$: Binary decision variable for appliance i at time slot t (1=on, 0=off).
    - p_i: Power consumption of appliance i in kilowatt-hours (kWh).
    - T: The set of all time slots in a day (24 for hourly time slots).
    - n: Total number of appliances.
    - $occupancy_t$: Number of people at home during time slot t.
    - $temperature_t$: Outdoor temperature during time slot t.
  - **Objective function (Minimize): -**
    - $\sum_{t=1}^{T} \sum_{i=1}^{n} x_{i,t} \cdot p_i$
  - **Constraints: -**
    - $\sum_{t=1}^{T} x_{TV,t} \leq 4$
    - $x_{AC_j,t} = 0$ for all $j \in \{1, 2, \ldots, 8\}$ and t where $25 \leq temperature_t \leq 29$
    - $x_{VacuumCleaner,t} = 1$ for time slot t at 7 a.m. on specified days
    - $x_{ClotheWasher,t} = 1$ for time slot t at 10 a.m. on specified days
    - $x_{ClothesDryer,t} = 1$ for time slot t at 12 p.m. on specified days
    - $x_{i,t} = 0$ for $i \in \{ACunits, TV, VacuumCleaner\}$ if $\text{occupancy}_t = 0$
    - $\sum_{j=1}^{8} x_{AC_j,t} \leq occupancy_t$

Linear Programming

Linear programming is a mathematical technique that maximizes or minimizes an objective function under constraints to make optimal decisions. Using algorithms and models, the technique solves large-scale problems that would otherwise be too complicated to manage. It is a powerful tool for data-driven strategy and planning due to its ability to optimize resource use and adapt to a variety of fields (Dantzig, 2002). Thus, Linear programming will be the ideal optimization technique for this objective function.

First, PuLP library will be installed for optimization in addition to the earlier Python libraries:

```
# Install the PuLP library, which is a Python library used for linear programming and optimization.
```

```
pip install pulp
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pulp in c:\users\ameen\appdata\roaming\python\python311\site-packages (2.9.0)
Note: you may need to restart the kernel to use updated packages.
```

Figure 49: Install additional library for optimization

Second, linear programming optimization technique will be coded:

```python
# Optimization technique: Linear programming

# Import necessary components from PuLP for linear programming:
# - LpMinimize: Specifies a minimization objective, used to reduce total energy usage.
# - LpProblem: Establishes the optimization model with defined objectives and constraints.
# - LpVariable: Creates decision variables for each device's on/off state in each time slot.
# - LpSum: Calculates the total power consumption, used in the objective and constraint expressions.
from pulp import LpMinimize, LpProblem, LpVariable, lpSum

# Dictionary to define power consumption values (in kWh) for each device we aim to optimize.
power_consumption = {
    'AC1': 3.4, 'AC2': 1.83, 'AC3': 3.4, 'AC4': 2, 'AC5': 1.98, 'AC6': 2.08,
    'AC7': 2.12, 'AC8': 2.12, 'TV': 0.27, 'VacuumCleaner': 1, 'ClotheWasher': 0.41, 'ClothesDryer': 2.7
}

# Dictionary for devices that are constantly on, with their fixed power usage.
always_on_devices = {
    'Refrigeter1': 0.5, 'Refrigeter2': 0.5, 'Refrigeter3': 0.3,
    'Freezer': 0.0354, 'WaterDispenser': 0.94
}

# Assume dt is your DataFrame containing the dataset.
# Extract only the necessary columns from the DataFrame for optimization processing.
original_columns = [
    'HouseID', 'Date', 'Time', 'Occupancy', 'OutdoorTemperature',
    'AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8', 'TV',
    'Refrigeter1', 'Refrigeter2', 'Refrigeter3', 'Freezer',
    'VacuumCleaner', 'ClotheWasher', 'ClothesDryer', 'WaterDispenser'
]
dtOptimize = dt[original_columns]
dtOptimize['DateTime'] = pd.to_datetime(dtOptimize['Date'], format='%d/%m/%Y')

# Function to optimize energy usage for a single row of data.
def optimize_energy(row, daily_tv_usage):
    # Extract relevant parameters for setting constraints.
    outdoor_temperature = row['OutdoorTemperature']
    time_of_day = row['Time']
    day_of_week = row['DateTime'].day_name()

    # Define an optimization model aimed at minimizing energy usage.
    model = LpProblem("Minimize_Energy_Usage", LpMinimize)

    # Create binary decision variables for each device; 1 if the device is on, 0 if off.
    devices = {device: LpVariable(device, cat="Binary") for device in power_consumption}

    # Objective function: Minimize the total energy usage of active devices.
    model += lpSum(devices[device] * power for device, power in power_consumption.items()), "Total_Energy_Usage"

    # Conditional constraints for device operation based on temperature and other conditions:
    # Temperature constraint: AC will be 0 if temperature between 25-29°C range.
    ac_devices = ['AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8']
    if 25 <= outdoor_temperature <= 29:
        # Turn off all ACs if temperature is in the specified range.
        for ac in ac_devices:
            model += devices[ac] == 0
    else:
        # If outside the range, AC usage is the same.
        for ac in ac_devices:
            model += devices[ac] == (1 if row[ac] > 0 else 0)

    # TV constraint: Limit TV usage to 4 hours per day.
    if daily_tv_usage < 4 and row['TV'] > 0:
        model += devices['TV'] == 1  # TV can be on if daily usage is below the limit.
    else:
        model += devices['TV'] == 0  # TV remains off if usage limit is reached.

    # Vacuum cleaner constraint: Only use on specific days and times.
    if day_of_week in ["Tuesday", "Thursday", "Saturday"] and time_of_day == 7:
        model += devices['VacuumCleaner'] == 1  # Vacuum cleaner can be on.
    else:
        model += devices['VacuumCleaner'] == 0  # Vacuum cleaner remains off otherwise.

    # Laundry constraints: ClotheWasher and ClothesDryer operate only on specified days and times.
    if day_of_week in ["Sunday", "Tuesday", "Thursday"]:
        model += devices['ClotheWasher'] == (1 if time_of_day == 10 else 0)
        model += devices['ClothesDryer'] == (1 if time_of_day == 12 else 0)
    else:
        model += devices['ClotheWasher'] == 0
        model += devices['ClothesDryer'] == 0

    # Solve the optimization problem and retrieve results.
    model.solve()
    # Store optimized values for each device, including constant devices that are always on.
    optimized_values = {device: int(devices[device].varValue) * power_consumption[device] for device in devices}
    optimized_values.update(always_on_devices)  # Add constant devices' power usage.
    return optimized_values

# Apply optimization function to each row of data and update daily TV usage.
for date, group in dtOptimize.groupby('DateTime'):
    daily_tv_usage = 0  # Track daily TV usage per day.
    for index, row in group.iterrows():
        # Get optimized energy usage for the row.
        optimized_values = optimize_energy(row, daily_tv_usage)
        # Update daily TV usage counter if TV is on in this row.
        if optimized_values['TV'] > 0:
            daily_tv_usage += 1
        # Write optimized values back into the DataFrame.
        for device, value in optimized_values.items():
            dtOptimize.at[index, device] = value

# Occupancy constraint: Adjust device usage according to occupancy values.
ac_columns = ['AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8']
for index, row in dtOptimize.iterrows():
    occupancy = row['Occupancy']
    # If no one is home, set certain devices to 0 to save energy.
    if occupancy == 0:
        dtOptimize.loc[index, ac_columns + ['TV', 'VacuumCleaner']] = 0.0
    else:
        # Limit the number of active ACs to match occupancy level.
        active_acs = [ac for ac in ac_columns if row[ac] > 0]
        if len(active_acs) > occupancy:
            for ac in active_acs[occupancy:]:
                dtOptimize.at[index, ac] = 0.0

# Calculate the total energy usage for each row and store it in a new column.
usage_columns = [
    'AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8', 'TV',
    'VacuumCleaner', 'ClotheWasher', 'ClothesDryer',
    'Refrigeter1', 'Refrigeter2', 'Refrigeter3', 'Freezer', 'WaterDispenser'
]
dtOptimize['TotalUsagekWh'] = dtOptimize[usage_columns].sum(axis=1)

# Finalize dataset by selecting relevant columns and exporting to an Excel file.
final_columns = original_columns + ['TotalUsagekWh']
dtNewDataset = dtOptimize[final_columns]
dtNewDataset.to_excel("C:/Users/ameen/OneDrive/Desktop/EnergyProject/OptimizedEnergyUsageDataset.xlsx", index=False)
print("Optimized dataset saved as 'OptimizedEnergyUsageDataset.xlsx'")
```

Figure 50: Optimization technique (Linear Programming) code

Finally, check that data was optimized successfully:

```python
# view of the optimized dataset
print(dtNewDataset)

print('----------------------------------------------------------------------------------------------------------------')

# Check the structure and data types of each column for the optimized dataset
print(dt.info())
```

Figure 51: Optimized dataset overview code

```
    HouseID        Date  Time  Occupancy  OutdoorTemperature  AC1  AC2  AC3  \
0        A1  13/10/2024     0          4                30.0  3.4  0.0  3.4
1        A2  13/10/2024     1          4                30.0  3.4  0.0  3.4
2        A3  13/10/2024     2          4                30.0  3.4  0.0  3.4
3        A4  13/10/2024     3          4                29.0  0.0  0.0  0.0
4        A5  13/10/2024     4          4                29.0  0.0  0.0  0.0
..      ...         ...   ...        ...                 ...  ...  ...  ...
163    A164  19/10/2024    19          4                31.7  0.0  0.0  0.0
164    A165  19/10/2024    20          5                31.3  0.0  0.0  0.0
165    A166  19/10/2024    21          2                30.4  0.0  0.0  0.0
166    A167  19/10/2024    22          2                30.3  0.0  0.0  0.0
167    A168  19/10/2024    23          5                29.7  0.0  0.0  0.0

     AC4   AC5  ...   TV  Refrigeter1  Refrigeter2  Refrigeter3  Freezer  \
0    0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
1    0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
2    0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
3    0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
4    0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
..   ...   ...  ...  ...          ...          ...          ...      ...
163  0.0  1.98  ...  0.0          0.5          0.5          0.3   0.0354
164  0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
165  0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
166  0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354
167  0.0  0.00  ...  0.0          0.5          0.5          0.3   0.0354

     VacuumCleaner  ClotheWasher  ClothesDryer  WaterDispenser  TotalUsagekWh
0              0.0           0.0           0.0            0.94        11.1954
1              0.0           0.0           0.0            0.94        11.1954
2              0.0           0.0           0.0            0.94        11.1954
3              0.0           0.0           0.0            0.94         2.2754
4              0.0           0.0           0.0            0.94         2.2754
..             ...           ...           ...             ...            ...
163            0.0           0.0           0.0            0.94         8.4954
164            0.0           0.0           0.0            0.94         6.5154
165            0.0           0.0           0.0            0.94         6.5154
166            0.0           0.0           0.0            0.94         6.5154
167            0.0           0.0           0.0            0.94         6.5154

[168 rows x 23 columns]
----------------------------------------------------------------------------------
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 168 entries, 2024-10-13 00:00:00 to 2024-10-19 23:00:00
Data columns (total 27 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   HouseID                     168 non-null    object
 1   Date                        168 non-null    object
 2   Time                        168 non-null    int32
 3   Occupancy                   168 non-null    int64
 4   OutdoorTemperature          168 non-null    float64
 5   AC1                         168 non-null    float64
 6   AC2                         168 non-null    float64
 7   AC3                         168 non-null    float64
 8   AC4                         168 non-null    float64
 9   AC5                         168 non-null    float64
 10  AC6                         168 non-null    float64
 11  AC7                         168 non-null    float64
 12  AC8                         168 non-null    float64
 13  TV                          168 non-null    float64
 14  Refrigeter1                 168 non-null    float64
 15  Refrigeter2                 168 non-null    float64
 16  Refrigeter3                 168 non-null    float64
 17  Freezer                     168 non-null    float64
 18  VacuumCleaner               168 non-null    float64
 19  ClotheWasher                168 non-null    float64
 20  ClothesDryer                168 non-null    float64
 21  WaterDispenser              168 non-null    float64
 22  TotalUsagekWh               168 non-null    float64
 23  TotalACUsage                168 non-null    float64
 24  TotalCoolingDevicesUsage    168 non-null    float64
 25  TotalCleaningDevicesUsage   168 non-null    float64
 26  UsageCategory               168 non-null    category
dtypes: category(1), float64(22), int32(1), int64(1), object(2)
memory usage: 39.1+ KB
None
```

Figure 52: Optimized dataset overview result

Dataset is optimized successfully using linear programming approaches where all attributes' values remained the same except for "AC1", "AC2", "AC3", "AC4", "AC5", "AC6" "AC7", "AC8", "TV", "VacuumCleaner", "ClotheWasher", "ClothesDryer" and "TotalUsagekWh".

Data Analysis and Visualization After Optimization

1. Descriptive Analysis after optimization

- **Summary statistics**

```python
# Aanalyze the optimized dataset: -

# Descriptive Analysis: -

# Get summary statistics of the numerical columns : mean, median, and standard deviation
print("Summary Statistics:")
print(dtNewDataset.describe())
```

Figure 53: Summary statistics after optimization code

```
Summary Statistics:
             Time   Occupancy  OutdoorTemperature         AC1         AC2  \
count  168.000000  168.000000          168.000000  168.000000  168.000000
mean    11.500000    3.767857           30.901190    0.425000    0.087143
std      6.942881    0.773616            1.916552    1.127806    0.390880
min      0.000000    1.000000           27.000000    0.000000    0.000000
25%      5.750000    3.000000           29.000000    0.000000    0.000000
50%     11.500000    4.000000           31.000000    0.000000    0.000000
75%     17.250000    4.000000           33.000000    0.000000    0.000000
max     23.000000    6.000000           35.000000    3.400000    1.830000

              AC3         AC4         AC5         AC6         AC7  ...  \
count  168.000000  168.000000  168.000000  168.000000  168.000000  ...
mean     0.667857    0.095238    0.577500    0.160952    0.959048  ...
std      1.354846    0.427191    0.902659    0.557427    1.058336  ...
min      0.000000    0.000000    0.000000    0.000000    0.000000  ...
25%      0.000000    0.000000    0.000000    0.000000    0.000000  ...
50%      0.000000    0.000000    0.000000    0.000000    0.000000  ...
75%      0.000000    0.000000    1.980000    0.000000    2.120000  ...
max      3.400000    2.000000    1.980000    2.080000    2.120000  ...

               TV  Refrigeter1  Refrigeter2   Refrigeter3       Freezer  \
count  168.000000        168.0        168.0  1.680000e+02  1.680000e+02
mean     0.045000          0.5          0.5  3.000000e-01  3.540000e-02
std      0.100924          0.0          0.0  7.238024e-16  7.655602e-17
min      0.000000          0.5          0.5  3.000000e-01  3.540000e-02
25%      0.000000          0.5          0.5  3.000000e-01  3.540000e-02
50%      0.000000          0.5          0.5  3.000000e-01  3.540000e-02
75%      0.000000          0.5          0.5  3.000000e-01  3.540000e-02
max      0.270000          0.5          0.5  3.000000e-01  3.540000e-02

       VacuumCleaner  ClotheWasher  ClothesDryer  WaterDispenser  \
count     168.000000    168.000000    168.000000    1.680000e+02
mean        0.017857      0.007321      0.048214    9.400000e-01
std         0.132828      0.054459      0.358636    1.558959e-15
min         0.000000      0.000000      0.000000    9.400000e-01
25%         0.000000      0.000000      0.000000    9.400000e-01
50%         0.000000      0.000000      0.000000    9.400000e-01
75%         0.000000      0.000000      0.000000    9.400000e-01
max         1.000000      0.410000      2.700000    9.400000e-01

       TotalUsagekWh
count     168.000000
mean        5.883912
std         2.904246
min         2.275400
25%         2.275400
50%         6.375400
75%         7.795400
max        13.745400

[8 rows x 21 columns]
```

Figure 54: Summary statistics after optimization result

After applying optimization method, the summary statistics for all attributes remain the same, but for total usage and devices such as ACs, TVs, vacuum cleaners, clothes washers, and clothes dryers. The average total usage decreased from 8.6 to 5.9, and the maximum value changed from 15.4 to 13.7, which indicates a lower energy consumption.

- **Distribution using box plot for Occupancy, Outdoor temperature and total energy usage.**

```python
# Box plot after optimization
# Create a figure and subplots with 1 row and 3 columns for displaying multiple box plots side by side
fig, ax = plt.subplots(1, 3, figsize=(15, 6))

# Adjust the spacing between the subplots for clarity and separation
plt.subplots_adjust(wspace=0.5)

# Plot a boxplot for the 'Occupancy' column in the first subplot, setting the color to brown
sns.boxplot(data=dtNewDataset['Occupancy'], ax=ax[0], color='brown')
ax[0].set_title('Occupancy')  # Set the title for the first subplot
ax[0].set_xlabel('Occupancy')  # Set the label for the x-axis of the first subplot

# Plot a boxplot for 'OutdoorTemperature' in the second subplot, setting the color to red
sns.boxplot(data=dtNewDataset['OutdoorTemperature'], ax=ax[1], color='Red')
ax[1].set_title('Outdoor Temperature')  # Set the title for the second subplot
ax[1].set_xlabel('Outdoor Temperature')  # Set the label for the x-axis of the second subplot

# Plot a boxplot for 'TotalUsagekWh' in the third subplot, setting the color to green
sns.boxplot(data=dtNewDataset['TotalUsagekWh'], ax=ax[2], color='Green')
ax[2].set_title('Total Usage (kWh)')  # Set the title for the third subplot
ax[2].set_xlabel('Total Usage (kWh)')  # Set the label for the x-axis of the third subplot

# Display the created box plots on the screen
plt.show()
```

Figure 55: Box plot distribution after optimization code

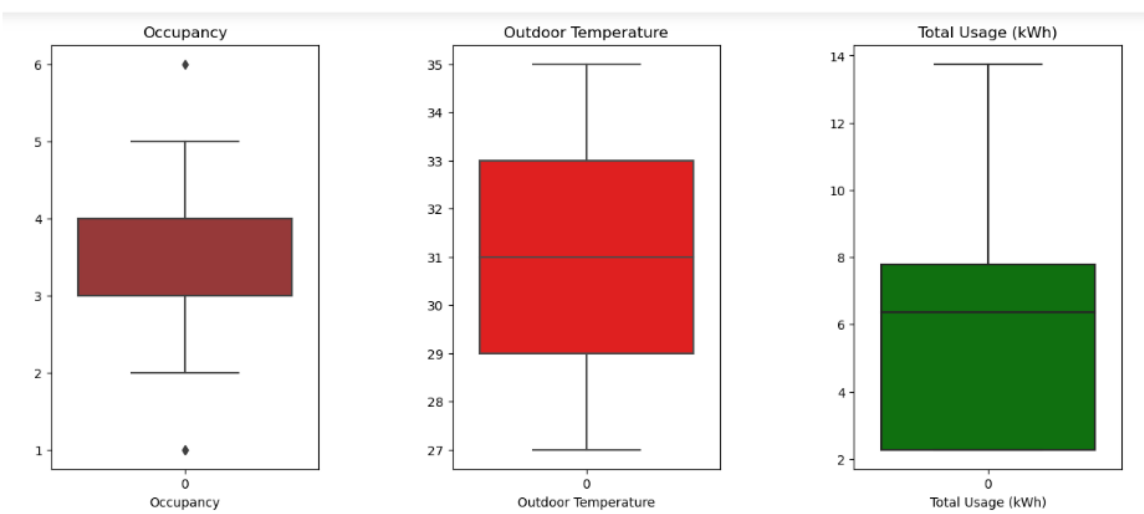


Figure 56: Box plot distribution after optimization result

The box plot distributions for occupancy and outdoor temperature remain unchanged, but the total usage box plot is different since there are no outliers and the average value is around 6 kWh. Furthermore, In the interquartile range (IQR), the 25th percentile became 2.5 kWh and the 75th percentile is around 7.8 kWh.

- **Minimum and maximum usage details**

```
# After optimization
# Show minimum Energy Usage details

# Find the row with minimum energy usage
min_usage_row = dtNewDataset.loc[dtNewDataset['TotalUsagekWh'].idxmin()]

# Print the details for the minimum energy usage
print("Minimum Energy Usage:")
print(f"The hour: {min_usage_row['Time']}")
print(f"Day: {min_usage_row['Date']}")
print(f"Total Usage (kWh): {min_usage_row['TotalUsagekWh']}")
print(f"Temperature: {min_usage_row['OutdoorTemperature']}")
print(f"Occupancy: {min_usage_row['Occupancy']}")

# Minimum Energy Usage: The lowest energy consumption was recorded at 3 AM on October 13th,
# under comfortable conditions (29°C) with 4 people present.

print('-----------------------------------------------------------------------------------------------------------------')


# Show maximum Energy Usage details

# Find the row with maximum energy usage
max_usage_row = dtNewDataset.loc[dtNewDataset['TotalUsagekWh'].idxmax()]

# Print the details for the maximum energy usage
print("\nMaximum Energy Usage:")
print(f"The hour: {max_usage_row['Time']}")
print(f"Day: {max_usage_row['Date']}")
print(f"Total Usage (kWh): {max_usage_row['TotalUsagekWh']}")
print(f"Temperature: {max_usage_row['OutdoorTemperature']}")
print(f"Occupancy: {max_usage_row['Occupancy']}")

# Maximum Energy Usage: The highest energy consumption occurred in the at Midnight on October 19th,
# with a higher temperature of 29.8°C and 5 people present.
```

Figure 57: Minimum and maximum usage after optimization code


```
Minimum Energy Usage:
The hour: 3
Day: 13/10/2024
Total Usage (kWh): 2.2754000000000003
Temperature: 29.0
Occupancy: 4
----------------------------------------

Maximum Energy Usage:
The hour: 0
Day: 19/10/2024
Total Usage (kWh): 13.745400000000002
Temperature: 29.8
Occupancy: 5
```

Figure 58: Minimum and maximum usage after optimization result

The optimized data collection measures the **minimum** energy usage of 2.28 kWh at 3:00 a.m. on Sunday, 13/10/2024, with 4 occupants and a 29°C temperature. Due to the pleasant temperature and the fact that the occupants were sleeping, no devices were used.

In contrast, the **maximum** usage was 13.75 kWh on Saturday, 19/10/2024 at midnight, at a slightly higher temperature of 29.8°C and with five occupants. Due to Saturday being a holiday in Bahrain, people tend to stay up late and use more electronics.

2. Inferential Analysis after optimization

- **Correlation analysis**

```python
# After optimization
# Inferential Analysis

# Correlation Analysis

# Heatmap for correlations matrix
plt.figure(figsize=(20, 14))  # Increase width and height for better visibility

# Create a heatmap using the correlation matrix with scale ranging from -1 to 1
sns.heatmap(dtNewDataset.corr(), annot=True, cmap='RdBu', linewidths=1, linecolor='Brown', vmin=-1, vmax=1)

# Display the heatmap
plt.show()
```

Figure 59: Correlation matrix after optimization code

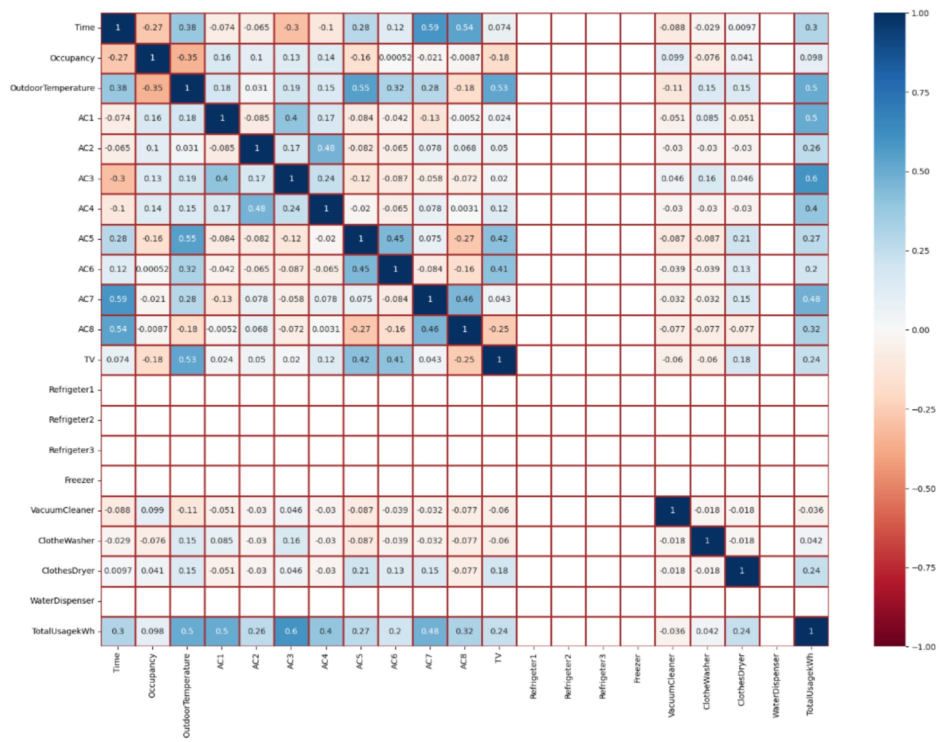


Figure 60: Correlation matrix after optimization result

Correlations equal and above 0.5: -

- **Time and AC7:** 0.59.
- **Time and AC8:** 0.54.
- **OutdoorTemperature and AC5:** 0.55.
- **OutdoorTemperature and TV:** 0.53.
- **OutdoorTemperature and TotalUsagekWh:** 0.5.
- **AC1 and TotalUsagekWh:** 0.5.
- **AC3 and TotalUsagekWh:** 0.6.
- **AC5 and OutdoorTemperature:** 0.55.

Correlations equal and below -0.5: -

- The enhanced dataset shows no strong negative correlations below -0.5, which means that no pair of features has a significant inverse relationship.

Correlations that have empty fields because they are constants: -

- **Refrigeter1.**
- **Refrigeter2.**
- **Refrigeter3.**
- **Freezer.**
- **WaterDispenser.**

A positive correlation between variables indicates patterns of household energy use. Time shows a moderate correlation with AC7 (0.59) and AC8 (0.54), suggesting specific times for AC usage. OutdoorTemperature correlates with AC5 (0.55), TV (0.53), and TotalUsagekWh (0.5), indicating higher temperatures result in more indoor activities and cooling which increase the total energy consumption. The correlation of AC1 and AC3 and TotalUsagekWh implies that the two air conditioners play a role in the overall energy consumption. However, no significant negative correlation exists below or equal to -0.5.

- **Regression analysis**

Regression analysis between AC3 and TotalUsagekWh: -

- **Independent Variable:** AC3.
- **Dependent Variable:** TotalUsagekWh.

```
# Regression model measure relationships between variables
# Highest Positive Relationships: -
# x Independent variable
x = dtNewDataset['AC3']
# y Dependent variable
y = dtNewDataset['TotalUsagekWh']

# Split the dataset into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(x, y, train_size=0.7, test_size=0.3, random_state=100)

# Add a constant to the model (intercept)
x_train = sm.add_constant(x_train)

# Fit the OLS regression model
LR = sm.OLS(y_train, x_train).fit()

# Print the summary of the regression model
print(LR.summary())
```

Figure 61: Regression analysis after optimization code


```
------------------------------------------------------------------------------------------------
                            OLS Regression Results
==============================================================================
Dep. Variable:          TotalUsagekWh   R-squared:                       0.393
Model:                            OLS   Adj. R-squared:                  0.388
Method:                 Least Squares   F-statistic:                     74.53
Date:                Sat, 09 Nov 2024   Prob (F-statistic):           3.91e-14
Time:                        19:54:07   Log-Likelihood:                -262.73
No. Observations:                 117   AIC:                             529.5
Df Residuals:                     115   BIC:                             535.0
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          5.0263      0.242     20.798      0.000       4.548       5.505
AC3            1.3017      0.151      8.633      0.000       1.003       1.600
==============================================================================
Omnibus:                        5.364   Durbin-Watson:                   2.135
Prob(Omnibus):                  0.068   Jarque-Bera (JB):                3.455
Skew:                           0.248   Prob(JB):                        0.178
Kurtosis:                       2.319   Cond. No.                         2.03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

Figure 62: Regression analysis after optimization result

Based on the regression results, there is a significant positive relationship between AC3 usage and TotalUsagekWh. According to the model's R-squared value of 0.393, about 39.3% of the variation in TotalUsagekWh can be explained by AC3 usage alone. For AC3, the coefficient is 1.3017, meaning that for every unit increase in AC3 usage, TotalUsagekWh increases by approximately 1.3. Therefore, AC3 usage is a strong predictor of household energy usage in general.

- **Covariance analysis**

```python
# Calculate the covariance matrix
cov_matrix = dtNewDataset.cov()
# Create a custom diverging color palette for blue positive and red negative
cmap = sns.diverging_palette(10, 240, as_cmap=True)  # This reverses the colors to make positive blue and negative red
# Set the figure size for the plot to ensure readability
plt.figure(figsize=(20, 17))
# Create a heatmap for visualizing the covariance matrix
sns.heatmap(
    cov_matrix,  # The covariance matrix data
    annot=True,  # Display the covariance values in the heatmap cells
    cmap=cmap,  # Use the modified color map for positive (blue) and negative (red)
    linewidths=1,  # Add lines between cells for better separation
    fmt=".2f",  # Format the displayed values to show 2 decimal places
    center=0,  # Center the color map at zero, so 0.0 appears white
    cbar_kws={"shrink": 0.5}  # Shrink the color bar to fit the plot better
)

# Add a title to the heatmap for context
plt.title('Covariance Matrix Heatmap')
# Display the heatmap plot
plt.show()
```

Figure 63: Covariance matrix after optimization code

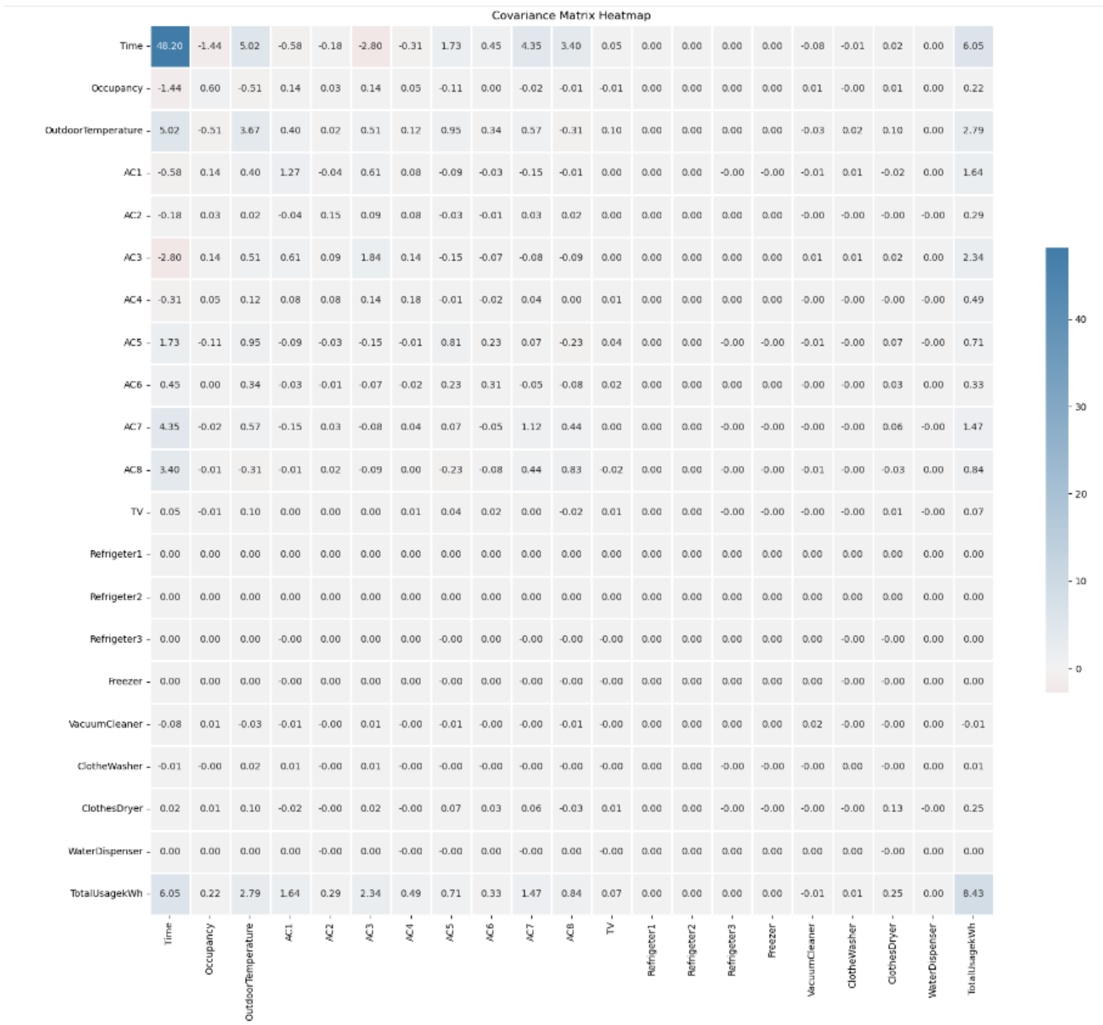


Figure 64: Covariance matrix after optimization result

The covariance matrix heatmap shows positive covariance between time and TotalUsagekWh (6.05). This value indicates an increase in the consumption of electricity as time progresses. Additionally, OutdoorTemperature and Time have a positive covariance value of 5.02, which indicates that the weather is changing over time. Other attributes do not appear to have a significant impact on each other.

3. Time Series Analysis after optimization

```
#Time Series Analysis
# Plot total usage over time to identify trends and patterns.
# Create 'Datetime' column and set it as the index
dtNewDataset['Datetime'] = pd.to_datetime(dtNewDataset['Date'] +
                                          ' ' + dtNewDataset['Time'].astype(str) + ':00:00', dayfirst=True)
dtNewDataset.set_index('Datetime', inplace=True)

# Get unique days and generate a color map for distinct visualization
unique_days = dtNewDataset['Date'].unique()
colors = cm.rainbow(np.linspace(0, 1, len(unique_days)))  # Create distinct colors for each day

plt.figure(figsize=(14, 7))

# Plot each day's data with a unique color for the lines and points
for i, day in enumerate(unique_days):
    daily_data = dtNewDataset[dt['Date'] == day]  # Extract data for the specific day
    day_name = pd.to_datetime(day, dayfirst=True).strftime('%A')  # Convert to day name (e.g., 'Sunday')
    plt.plot(daily_data.index, daily_data['TotalUsagekWh'], marker='o', color=colors[i], label=day_name, linestyle='-')

    # If not the last day, connect the end of this day's line to the start of the next with a solid black line
    if i < len(unique_days) - 1:
        next_day_data = dtNewDataset[dtNewDataset['Date'] == unique_days[i + 1]]
        if not daily_data.empty and not next_day_data.empty:
            plt.plot(
                [daily_data.index[-1], next_day_data.index[0]],
                [daily_data['TotalUsagekWh'].iloc[-1], next_day_data['TotalUsagekWh'].iloc[0]],
                color='black', linestyle='-'  # Solid black line connecting two different days
            )

# Add plot title and labels for better context
plt.title('Time Series Analysis of Total Energy Usage (kWh) for Each Day')
plt.xlabel('Time')
plt.ylabel('Total Usage (kWh)')
plt.legend(loc='upper right', bbox_to_anchor=(1.15, 1))  # Adjust legend position
plt.grid(True)  # Add grid lines for readability
plt.show()  # Display the plot
```

Figure 65: Time series analysis after optimization code

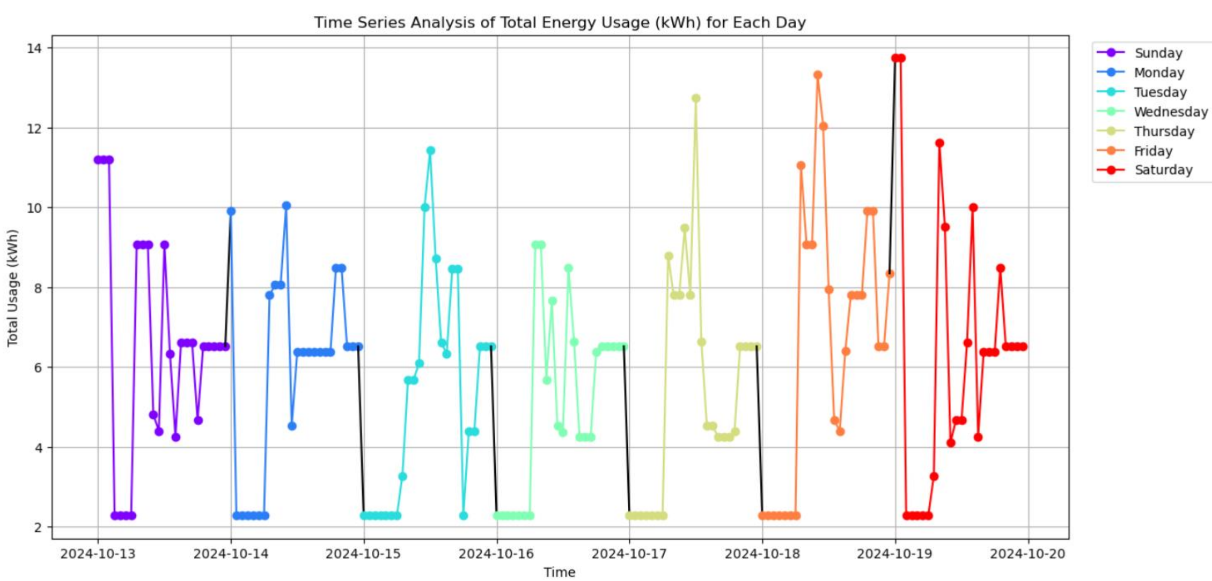


Figure 66: Time series analysis after optimization result

The time series line chart from Sunday to Saturday shows daily fluctuations in total energy consumption (kWh) with peaks varying by day. Higher usage is seen on Saturday (in red) and Friday (in orange) since these days are weekends in Bahrain. Additionally, the drop in energy usage in the middle of each day reflects a time when household activities are minimal.

4. Comparative Analysis after optimization

```
# Energy Usage by Appliance Analysis

# List of appliances in your dataset
appliances = [
    "AC1", "AC2", "AC3", "AC4", "AC5", "AC6", "AC7", "AC8", 'Refrigeter1', 'Refrigeter2', 'Refrigeter3', 'Freezer',
    "TV", "VacuumCleaner", "ClotheWasher", "ClothesDryer", "WaterDispenser"
]

# Calculate total energy usage for each appliance
total_usage_per_appliance = dtNewDataset[appliances].sum()

# Calculate percentage contribution for each appliance
percentage_contribution = (total_usage_per_appliance / total_usage_per_appliance.sum()) * 100

# Plot total energy usage per appliance
plt.figure(figsize=(12, 8))
total_usage_per_appliance.plot(kind='bar', color='red')
plt.title('Total Energy Usage per Appliance')
plt.xlabel('Appliance')
plt.ylabel('Total Usage (kWh)')
plt.xticks(rotation=45, ha='right')  # Rotate x labels for better readability
plt.show()
```

Figure 67: Energy usage by appliance analysis after optimization code

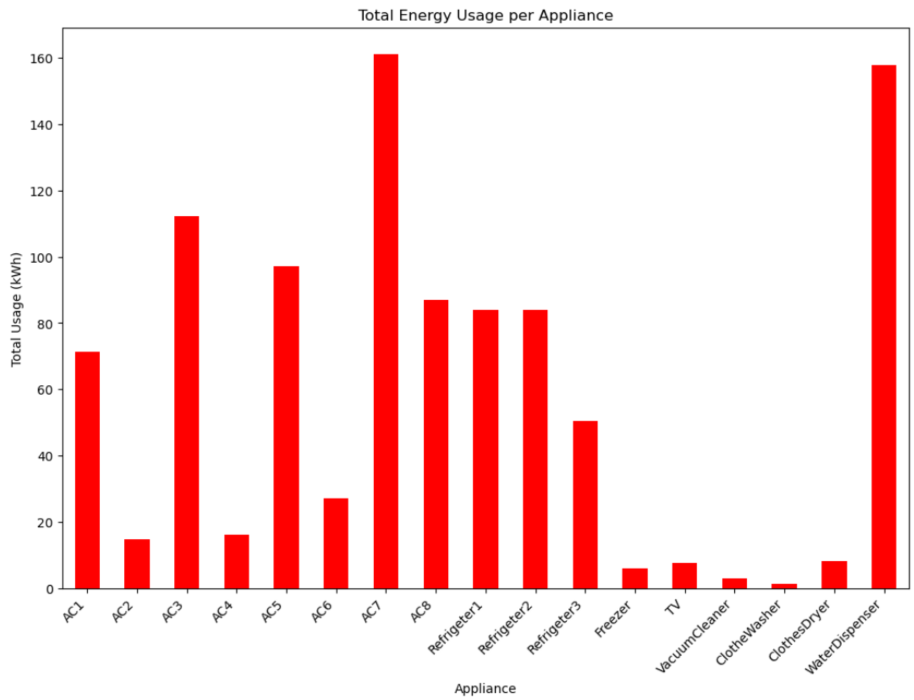


Figure 68: Energy usage by appliance analysis after optimization result

According to the chart, the water dispenser and AC7 consume the most energy. Despite optimizing consumer behavior, air conditioners still contribute the most to total energy consumption. Comparatively, clothes washers and vacuum cleaners still use significantly less energy.

5. Clustering Analysis after optimization

```
# Identify clusters
# Create new features by aggregating device usage into broader categories
dtNewDataset['TotalACUsage'] = (
    dtNewDataset['AC1'] + dtNewDataset['AC2'] + dtNewDataset['AC3'] + dtNewDataset['AC4'] +
    dtNewDataset['AC5'] + dtNewDataset['AC6'] + dtNewDataset['AC7'] + dtNewDataset['AC8']
)

# Group refrigerators, freezers, and water dispensers into CoolingDevices
dtNewDataset['TotalCoolingDevicesUsage'] = (
    dtNewDataset['Refrigeter1'] + dtNewDataset['Refrigeter2'] + dtNewDataset['Refrigeter3'] +
    dtNewDataset['Freezer'] + dtNewDataset['WaterDispenser']
)

# Group cleaning devices together
dtNewDataset['TotalCleaningDevicesUsage'] = (
    dtNewDataset['VacuumCleaner'] + dtNewDataset['ClotheWasher'] +
    dtNewDataset['ClothesDryer']
)

# Define the features for clustering
features = ['TotalACUsage', 'TotalCoolingDevicesUsage', 'TotalCleaningDevicesUsage']
dt_selected = dtNewDataset[features].dropna()  # Remove any rows with missing values for clustering

# Applying K-Means Clustering
kmeans = KMeans(n_clusters=3, random_state=42)  # Adjust number of clusters as needed
dt_selected['Cluster'] = kmeans.fit_predict(dt_selected)

# Aggregating usage for each device group by hour
hourly_ac_usage = dtNewDataset.groupby('Time')['TotalACUsage'].sum()
hourly_cooling_usage = dtNewDataset.groupby('Time')['TotalCoolingDevicesUsage'].sum()
hourly_cleaning_usage = dtNewDataset.groupby('Time')['TotalCleaningDevicesUsage'].sum()

# Plotting hourly usage for comparison
plt.figure(figsize=(10, 6))
plt.plot(hourly_ac_usage.index, hourly_ac_usage, label='AC Devices', marker='o', color='blue')
plt.plot(hourly_cooling_usage.index, hourly_cooling_usage, label='Cooling Devices', marker='o', color='green')
plt.plot(hourly_cleaning_usage.index, hourly_cleaning_usage, label='Cleaning Devices', marker='o', color='red')

# Labeling
plt.xlabel('Hour of the Day')
plt.ylabel('Total Usage (kWh)')
plt.title('Hourly Energy Usage Patterns by Device Category')
plt.xticks(range(0, 24, 1))  # Show all hours on the x-axis
plt.legend()
plt.grid(True)
plt.show()
```

Figure 69: Clustering analysis after optimization code

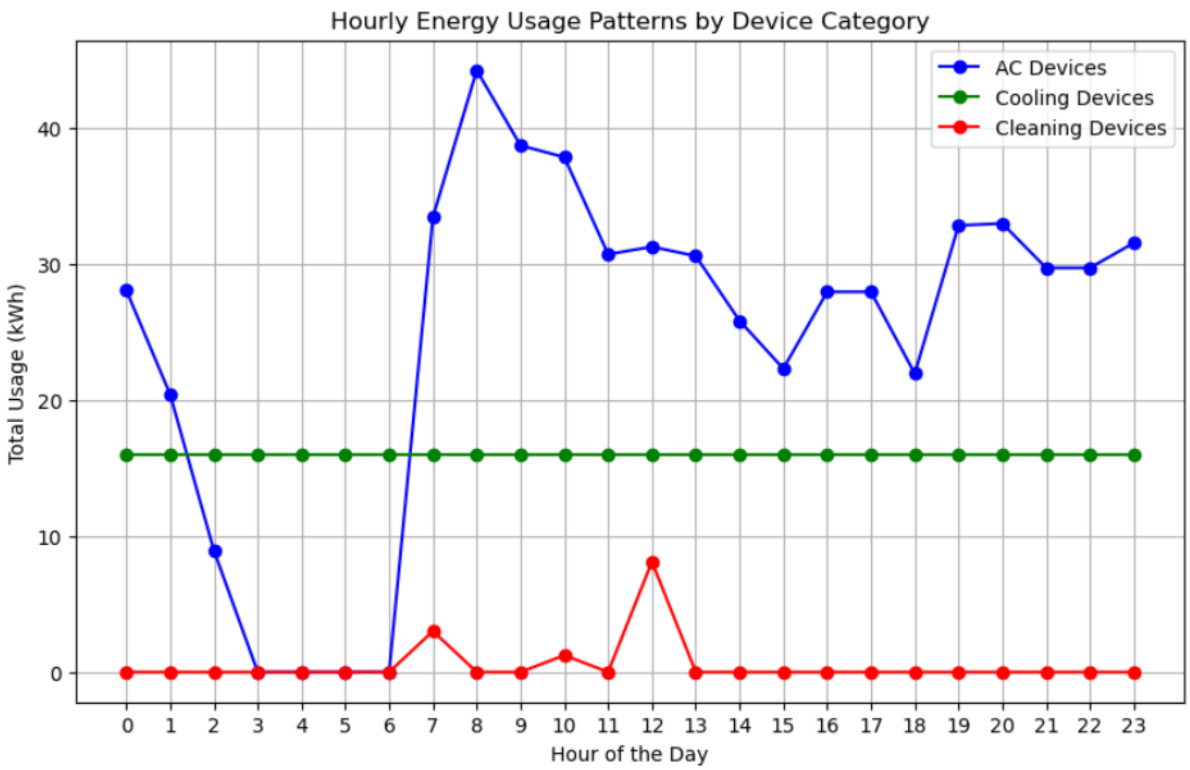


Figure 70: Clustering analysis after optimization result

In the chart, AC devices are shown to have the highest energy consumption, peaking above 40 kWh around 8 a.m. then declining and increasing again in the evening. AC devices are not used between 3 a.m. and 5 a.m. Possibly due to cooler nighttime temperatures or reduced household activity. Throughout the day, cooling devices use around 10 kWh continuously. while cleaning devices are only used occasionally, usually between 6 a.m. and 1 p.m.

6. Classification Analysis after optimization

- **Classification Report**

```
# Classification Analysis:
# Categorize 'TotalUsageKWh' into 'Low', 'Medium', and 'High' usage levels.
dtNewDataset['UsageCategory'] = pd.cut(dtNewDataset['TotalUsagekWh'], bins=[0, 5, 10, dtNewDataset['TotalUsagekWh'].max()],
                             labels=['Low', 'Medium', 'High'])

# Prepare the feature set (X) with appliance data and relevant variables.
X = dtNewDataset[['Time', 'OutdoorTemperature', 'AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8',
        'TV', 'Refrigeter1', 'Refrigeter2', 'Refrigeter3', 'Freezer',
        'VacuumCleaner', 'ClotheWasher', 'ClothesDryer', 'WaterDispenser']]
# Define the target variable (y) as the categorized energy usage.
y = dtNewDataset['UsageCategory']

# Split the dataset into training and testing sets (70% train, 30% test).
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Initialize the Decision Tree Classifier for classification.
classifier = DecisionTreeClassifier(random_state=42)

# Set up a hyperparameter grid for tuning the Decision Tree Classifier using GridSearchCV.
param_grid = {
    'max_depth': [None, 5, 10, 15],  # Varying tree depth for complexity control.
    'min_samples_split': [2, 5, 10],  # Minimum samples required to split a node.
    'min_samples_leaf': [1, 2, 5]  # Minimum samples required for a leaf node.
}

# Perform GridSearchCV for the best combination of hyperparameters.
grid_search = GridSearchCV(estimator=classifier, param_grid=param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)

# Display the optimal hyperparameters found.
print("Best parameters found: ", grid_search.best_params_)

# Use the best estimator from GridSearchCV to make predictions.
best_classifier = grid_search.best_estimator_
y_pred = best_classifier.predict(X_test)

# Print the classification report summarizing precision, recall, and F1-score.
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

Figure 71: Classification report after optimization code

```
Best parameters found:  {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2}

Classification Report:
              precision    recall  f1-score   support

        High       0.50      0.60      0.55         5
         Low       0.83      0.79      0.81        19
      Medium       0.85      0.85      0.85        27

    accuracy                           0.80        51
   macro avg       0.73      0.75      0.74        51
weighted avg       0.81      0.80      0.81        51
```

Figure 72: Classification report after optimization result

The classification report showed an accuracy of 80%. The Medium class performed the best, with precision, recall, and an F1-score of 0.85. The Low class followed with 0.81, while the High class had the lowest F1-score of 0.55 due to lower precision and recall. Overall, the macro average F1-score is 0.74, indicating a balance between classes.

- **Confusion Matrix**

```
# Create and display a confusion matrix to visualize prediction results.
conf_matrix = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, cmap='Blues', fmt='d',
            xticklabels=['Low', 'Medium', 'High'], yticklabels=['Low', 'Medium', 'High'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

Figure 73: Confusion matrix after optimization code

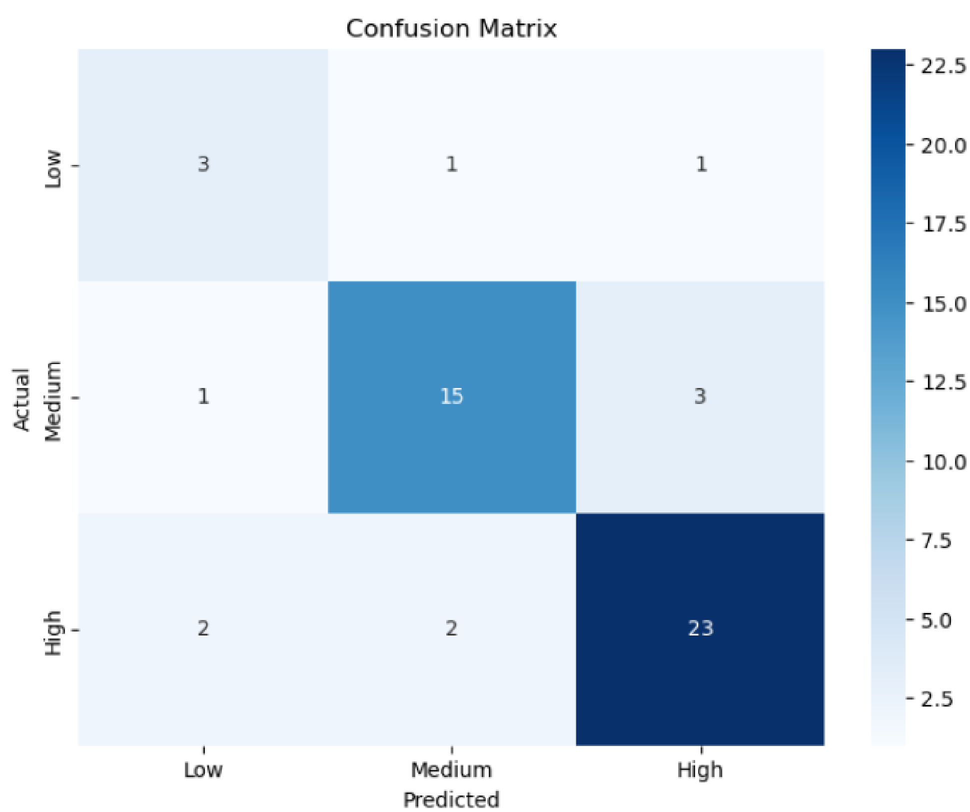


Figure 74: Confusion matrix after optimization result

According to the confusion matrix, 3 instances of the Low class have been correctly classified, while 1 instance has been misclassified as Medium and 1 as High. As for the Medium class, 15 instances were classified correctly, while 1 was classified incorrectly as Low and 3 as High. The High class had the most correct predictions, with 23 instances correctly classified, though two were misclassified as Low and two as Medium. As shown in this matrix, the model is most accurate at classifying High instances.

Data Summarization After Optimization

```python
# Data summarization for the optimized dataset
# Loop through each specified appliance to summarize daily usage patterns.
for appliance in ['AC1', 'AC2', 'AC3', 'AC4', 'AC5', 'AC6', 'AC7', 'AC8', 'TV',
                  'VacuumCleaner', 'ClotheWasher', 'ClothesDryer']:
    print(f"\n{appliance}:")  # Print the name of the appliance being analyzed.

    # Loop through each unique date in the dataset to check appliance usage on each day.
    for date in dtNewDataset['Date'].unique():
        # Identify the specific hours during which the appliance was running on that day.
        hours_running = dtNewDataset[
            (dtNewDataset['Date'] == date) & (dtNewDataset[appliance] > 0)
        ]['Time'].tolist()

        # If the appliance ran during any hours, display the hours it was active.
        if hours_running:
            print(f"  {date} - Run at hour(s): {', '.join(map(str, hours_running))}")
        else:
            # If the appliance was not used that day, indicate it was not running.
            print(f"  {date} - Not running")

# Calculate the total energy usage (in kWh) for each day by summing 'TotalUsagekWh' for each date.
total_usage_per_day = dtNewDataset.groupby('Date')['TotalUsagekWh'].sum()

# Display daily total energy usage.
print("\nTotal Energy Usage for Each Day:")
for date, total_usage in total_usage_per_day.items():
    print(f"Date: {date}, Total Usage: {total_usage:.2f} kWh")  # Format total usage to two decimal places.

# Calculate the cumulative energy usage across all days by summing daily totals.
total_usage_all_days = total_usage_per_day.sum()
# Display the overall energy usage across the analyzed period.
print(f"\nTotal Energy Usage Across All Days: {total_usage_all_days:.2f} kWh")
```

Figure 75: Data summarization after optimization code


```
AC1:
  13/10/2024 - Run at hour(s): 0, 1, 2, 7, 8, 9
  14/10/2024 - Not running
  15/10/2024 - Not running
  16/10/2024 - Run at hour(s): 7, 8
  17/10/2024 - Run at hour(s): 10
  18/10/2024 - Run at hour(s): 7, 8, 9, 10, 11, 12, 16, 17, 18, 19, 20
  19/10/2024 - Run at hour(s): 14

AC2:
  13/10/2024 - Not running
  14/10/2024 - Not running
  15/10/2024 - Not running
  16/10/2024 - Run at hour(s): 12, 13
  17/10/2024 - Not running
  18/10/2024 - Run at hour(s): 23
  19/10/2024 - Run at hour(s): 0, 1, 8, 9, 10

AC3:
  13/10/2024 - Run at hour(s): 0, 1, 2, 7, 8, 9
  14/10/2024 - Run at hour(s): 0, 7, 8, 9, 10
  15/10/2024 - Run at hour(s): 8, 9, 10, 11
  16/10/2024 - Run at hour(s): 7, 8, 9, 10
  17/10/2024 - Run at hour(s): 7, 8, 9, 10, 11, 12
  18/10/2024 - Run at hour(s): 7, 8, 9, 10
  19/10/2024 - Run at hour(s): 0, 1, 8, 9

AC4:
  13/10/2024 - Not running
  14/10/2024 - Not running
  15/10/2024 - Not running
  16/10/2024 - Not running
  17/10/2024 - Not running
  18/10/2024 - Run at hour(s): 10, 11, 12, 15
  19/10/2024 - Run at hour(s): 0, 1, 8, 9

AC5:
  13/10/2024 - Run at hour(s): 12, 13, 14, 15, 16, 17
  14/10/2024 - Run at hour(s): 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20
  15/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17
  16/10/2024 - Run at hour(s): 10, 11, 13, 14, 15, 16, 17, 18
  17/10/2024 - Run at hour(s): 12, 13, 14, 15, 16, 17, 18
  18/10/2024 - Run at hour(s): 7, 10, 11
  19/10/2024 - Run at hour(s): 13, 14, 15, 16, 17, 18, 19

AC6:
  13/10/2024 - Run at hour(s): 13, 15, 16, 17
  14/10/2024 - Not running
  15/10/2024 - Run at hour(s): 11, 12, 13, 14, 15, 16, 17
  16/10/2024 - Not running
  17/10/2024 - Not running
  18/10/2024 - Not running
  19/10/2024 - Run at hour(s): 13, 14

AC7:
  13/10/2024 - Run at hour(s): 10, 11, 12, 18, 19, 20, 21, 22, 23
  14/10/2024 - Run at hour(s): 0, 7, 8, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  15/10/2024 - Run at hour(s): 12, 13, 16, 17, 21, 22, 23
  16/10/2024 - Run at hour(s): 13, 14, 18, 19, 20, 21, 22, 23
  17/10/2024 - Run at hour(s): 7, 8, 9, 11, 12, 13, 20, 21, 22, 23
  18/10/2024 - Run at hour(s): 11, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23
  19/10/2024 - Run at hour(s): 0, 1, 8, 11, 12, 16, 17, 18, 19, 20, 21, 22, 23

AC8:
  13/10/2024 - Run at hour(s): 0, 1, 2, 19, 20, 21, 22, 23
  14/10/2024 - Run at hour(s): 0, 19, 20, 21, 22, 23
  15/10/2024 - Run at hour(s): 19, 20, 21, 22, 23
  16/10/2024 - Run at hour(s): 19, 20, 21, 22, 23
  17/10/2024 - Run at hour(s): 19, 20, 21, 22, 23
  18/10/2024 - Run at hour(s): 19, 20, 21, 22, 23
  19/10/2024 - Run at hour(s): 0, 1, 19, 20, 21, 22, 23

TV:
  13/10/2024 - Run at hour(s): 15, 16, 17, 18
  14/10/2024 - Run at hour(s): 8, 9, 10, 11
  15/10/2024 - Run at hour(s): 11, 12, 13, 14
  16/10/2024 - Run at hour(s): 11, 12, 13, 14
  17/10/2024 - Run at hour(s): 12, 13, 14, 15
  18/10/2024 - Run at hour(s): 10, 11, 12, 13
  19/10/2024 - Run at hour(s): 11, 12, 13, 14

VacuumCleaner:
  13/10/2024 - Not running
  14/10/2024 - Not running
  15/10/2024 - Run at hour(s): 7
  16/10/2024 - Not running
  17/10/2024 - Run at hour(s): 7
  18/10/2024 - Not running
  19/10/2024 - Run at hour(s): 7

ClotheWasher:
  13/10/2024 - Run at hour(s): 10
  14/10/2024 - Not running
  15/10/2024 - Run at hour(s): 10
  16/10/2024 - Not running
  17/10/2024 - Run at hour(s): 10
  18/10/2024 - Not running
  19/10/2024 - Not running

ClothesDryer:
  13/10/2024 - Run at hour(s): 12
  14/10/2024 - Not running
  15/10/2024 - Run at hour(s): 12
  16/10/2024 - Not running
  17/10/2024 - Run at hour(s): 12
  18/10/2024 - Not running
  19/10/2024 - Not running

Total Energy Usage for Each Day:
Date: 13/10/2024, Total Usage: 155.84 kWh
Date: 14/10/2024, Total Usage: 143.23 kWh
Date: 15/10/2024, Total Usage: 127.26 kWh
Date: 16/10/2024, Total Usage: 123.15 kWh
Date: 17/10/2024, Total Usage: 129.26 kWh
Date: 18/10/2024, Total Usage: 158.50 kWh
Date: 19/10/2024, Total Usage: 151.26 kWh

Total Energy Usage Across All Days: 988.50 kWh
```

Figure 76: Data summarization after optimization result

The figure above is a summary of the running time for each appliance that was used for seven days such as ACs, TV, vacuum cleaner, clothes washer, and clothes dryer. The day with maximum total energy usage was on Friday, 18/10/2024 at 158.50 kWh as it is a weekend in the country, while the day with minimum total energy usage was on Wednesday, 16/10/2024 at 123.15 kWh which is a working day in Bahrain. In total, 988.50 kWh were used during the week.

Critical Evaluation of the Chosen Optimization Technique

- **State:** The code was successfully implemented using Linear Programming (LP).
- **Reason:** Linear programming techniques minimize household energy consumption by controlling when electronics operate based on defined constraints.
- **Evidence:** The total energy consumption before optimization was 1445.01 kWh. With optimization techniques, it was reduced to 988.50 kWh, saving 456.51 kWh. This represents a reduction of approximately 31.6% or almost one-third of the initial consumption.
- **Alternate:** Smith & Caner Taşkın (2007) reported that Mixed-integer programming (MIP) is another optimization method that incorporates continuous and discrete variables, offering precise, efficient solutions to real-world problems as it can do what the Linear Programming method does but with additional capabilities. Unlike linear programming, MIP allows structured load shifting by enforcing distinct timeslots for each device. Nevertheless, in this case, it is unnecessary as the goal is to minimize the total energy usage per week. Even though distributing the electricity usage will lead to a decrease in the total energy usage of each hour, the total energy usage per day or week will remain the same. Thus, Linear programming is suitable for this scenario.

Testing

This section outlines the testing approaches used to successfully deliver the product. The solution was thoroughly validated to ensure accuracy and efficient operation. Test cases are utilized to demonstrate the process of testing. It also contains experts sharing their experience of using the solution.

Test Plan

Multiple strategies were employed to verify that the optimization code functions well. The following measures were taken to ensure the system functions effectively and is usable: -

- **System functionalities:** Verify that all functionalities of the system, both functional and non-functional, are working as intended.
- **Debugging and Code Review**: Checked the code thoroughly for logic errors in constraints, objectives, and structures.
- **Detailed Verification**: Conducting a step-by-step examination of each constraint to ensure compliance.
- **Performance**: Assessed performance to ensure efficient handling.

Participants

Participants for testing include a mix of an ICT student and client employees to ensure a variety of feedback. The participants' names, ages, gender and their background will be listed.

Table 3: Participants

Name	Age	Gender	Background
Sara Albuainain	21	Female	ICT student at Bahrain Polytechnic with a specialty in Databases, has an experience in data analysis and programming. Sara can assist in code review.
Dr. Abdulla Alabbasi	38	Male	Electrical engineering at Derasat research center, Expert in research that are related to environment and energy fields. Dr. Abdulla Alabbasi can help in the validation of the analysis methods and results.
Eng. Sabeeka Khalid	26	Female	Chemical engineering at Derasat research center, have knowledge in optimization techniques. Eng. Sabeeka Khalid can help in the validation of the linear programming approach.

Functionality Test Cases and Results

A description of the test cases related to the functionality of system parts by inspecting the actual result to ensure the expected outcome is achieved, thus, assigning a "Pass" or "Fail" status.

Table 4: Functionality test cases and results

STEP ID	STEP DESCRIPTION	TEST DATE	EXPECTED RESULTS	ACTUAL RESULTS	PASS / FAIL	ADDITIONAL NOTES
1	Verify that the dataset is clean from any noise	November 20, 2024	Dataset free from any error	Dataset free from any error	Pass	Tested
2	Check the format of all columns	November 20, 2024	Correct column format	Correct column format	Pass	Tested
3	Review the data analysis methods	November 20, 2024	Accurate data analysis	Accurate data analysis	Pass	Tested
4	Check the highest value	November 20, 2024	Maximum value is shown	Maximum value is shown	Pass	Tested
5	Check the minimum value	November 20, 2024	Minimum value is shown	Minimum value is shown	Pass	Tested
6	Confirm math formulas	November 20, 2024	Correct math formulas	Correct math formulas	Pass	Tested
7	Check that the current enhancement technique is suitable for this case	November 20, 2024	An ideal optimization technique is used	An ideal optimization technique is used	Passs	Tested
8	Check the visual graphs	November 20, 2024	Good presentation of data	Good presentation of data	Pass	Tested
9	Make sure a comment is on each line of code	November 20, 2024	Clean code where different developers will easily understand	Clean code where different developers will easily understand	Pass	Tested
10	Compare before and after using optimization method	November 20, 2024	Data is optimized successfully	Data is optimized successfully	Pass	Tested

Acceptance Tests Process and Results

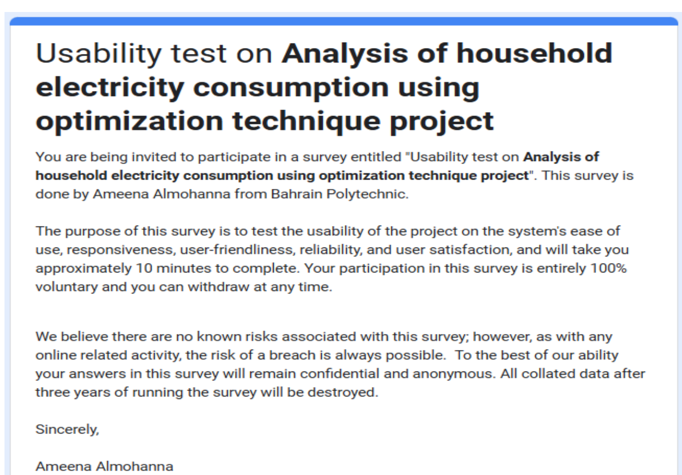
Definition of the acceptance tests conducted by the participants, showing whether certain processes succeeded or failed, along with a reflection on the results. The acceptance tests are extracted from the test cases. Before testing, the volunteers were given a brief overview of the project's problem, objectives, and solution.

Table 5: Acceptance tests process and results

Name	Process	Result	Reflection
Sara Albuainain	Code review	Succeeded	Code is clean and efficient.
Dr. Abdulla Alabbasi	Confirmation of the analysis methods	Succeeded	Analysis is sufficient and results in rich data.
Eng. Sabeeka Khalid	Validation of optimization technique	Succeeded	The choice of optimization method is excellent, and the formulas are correctly written.

Usability Testing Results and Statistics

A usability testing survey using Google Forms was performed to examine participants’ observations on the system's ease of use, responsiveness, user-friendliness, reliability, and user satisfaction.



Usability test on **Analysis of household electricity consumption using optimization technique project**

You are being invited to participate in a survey entitled "Usability test on **Analysis of household electricity consumption using optimization technique project**". This survey is done by Ameena Almohanna from Bahrain Polytechnic.

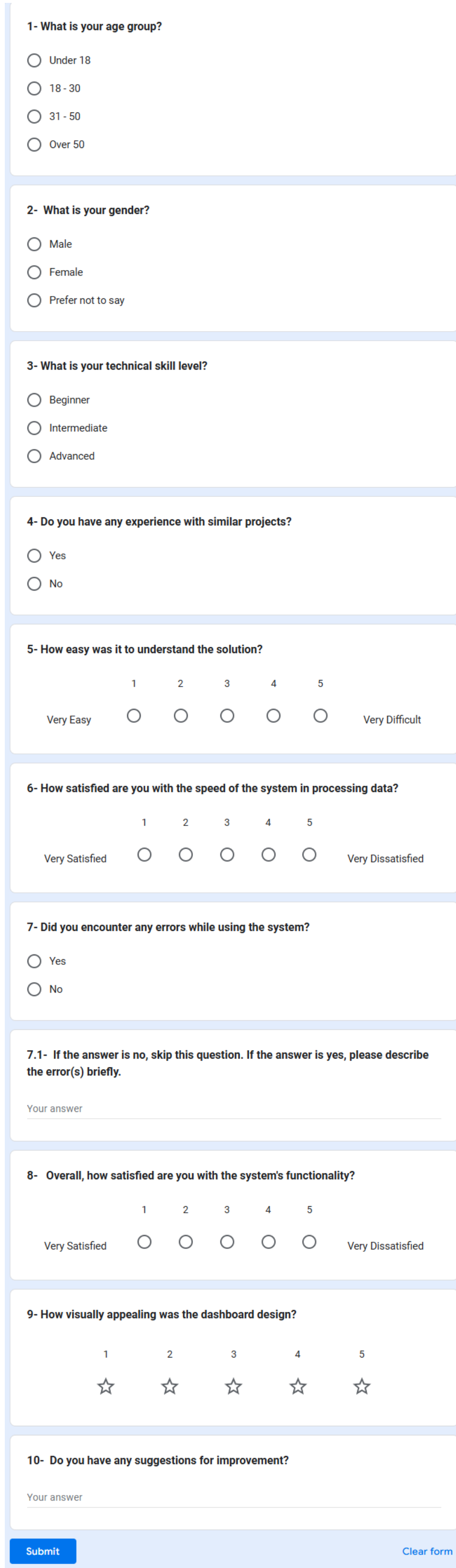
The purpose of this survey is to test the usability of the project on the system's ease of use, responsiveness, user-friendliness, reliability, and user satisfaction, and will take you approximately 10 minutes to complete. Your participation in this survey is entirely 100% voluntary and you can withdraw at any time.

We believe there are no known risks associated with this survey; however, as with any online related activity, the risk of a breach is always possible. To the best of our ability your answers in this survey will remain confidential and anonymous. All collated data after three years of running the survey will be destroyed.

Sincerely,

Ameena Almohanna

Figure 77: Usability testing survey description



1- What is your age group?

- Under 18
- 18 - 30
- 31 - 50
- Over 50

2- What is your gender?

- Male
- Female
- Prefer not to say

3- What is your technical skill level?

- Beginner
- Intermediate
- Advanced

4- Do you have any experience with similar projects?

- Yes
- No

5- How easy was it to understand the solution?

	1	2	3	4	5	
Very Easy	○	○	○	○	○	Very Difficult

6- How satisfied are you with the speed of the system in processing data?

	1	2	3	4	5	
Very Satisfied	○	○	○	○	○	Very Dissatisfied

7- Did you encounter any errors while using the system?

- Yes
- No

7.1- If the answer is no, skip this question. If the answer is yes, please describe the error(s) briefly.

Your answer

8- Overall, how satisfied are you with the system's functionality?

	1	2	3	4	5	
Very Satisfied	○	○	○	○	○	Very Dissatisfied

9- How visually appealing was the dashboard design?

1	2	3	4	5
☆	☆	☆	☆	☆

10- Do you have any suggestions for improvement?

Your answer

Submit   Clear form

Figure 78: Usability testing survey questions

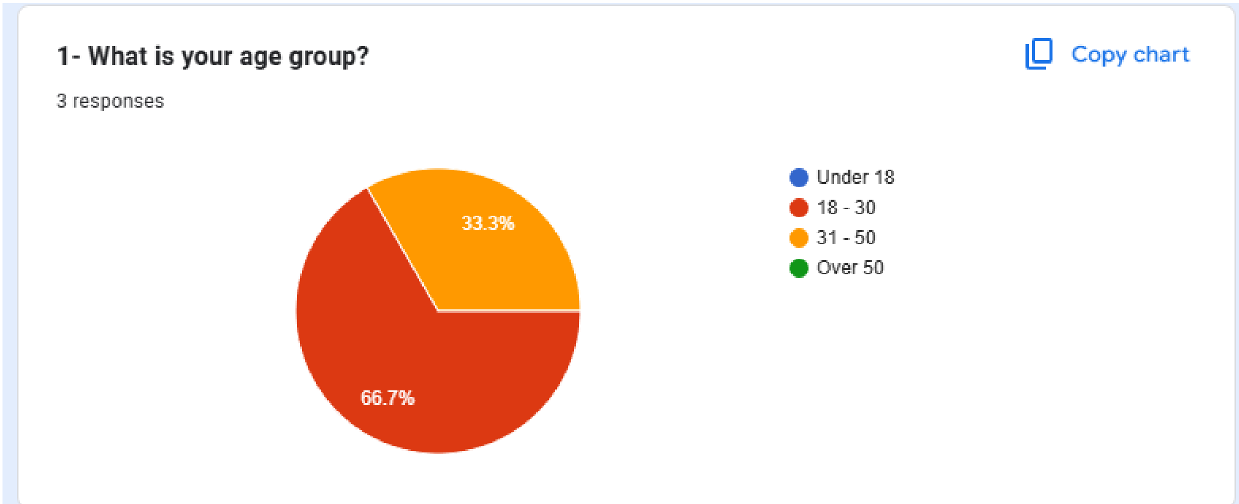


Figure 79: Usability testing survey answer one

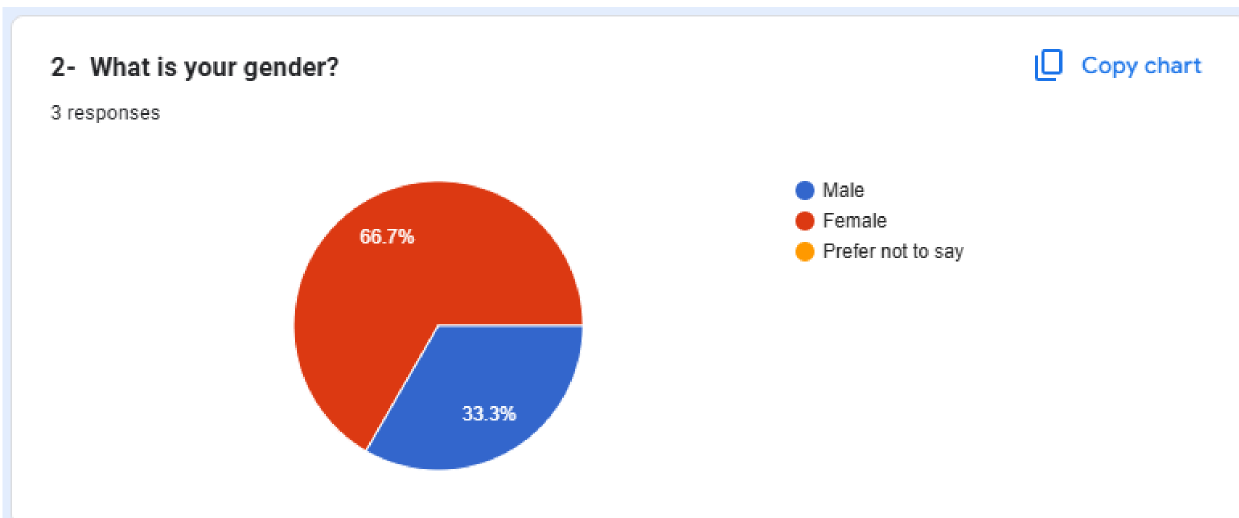


Figure 80: Usability testing survey answer two

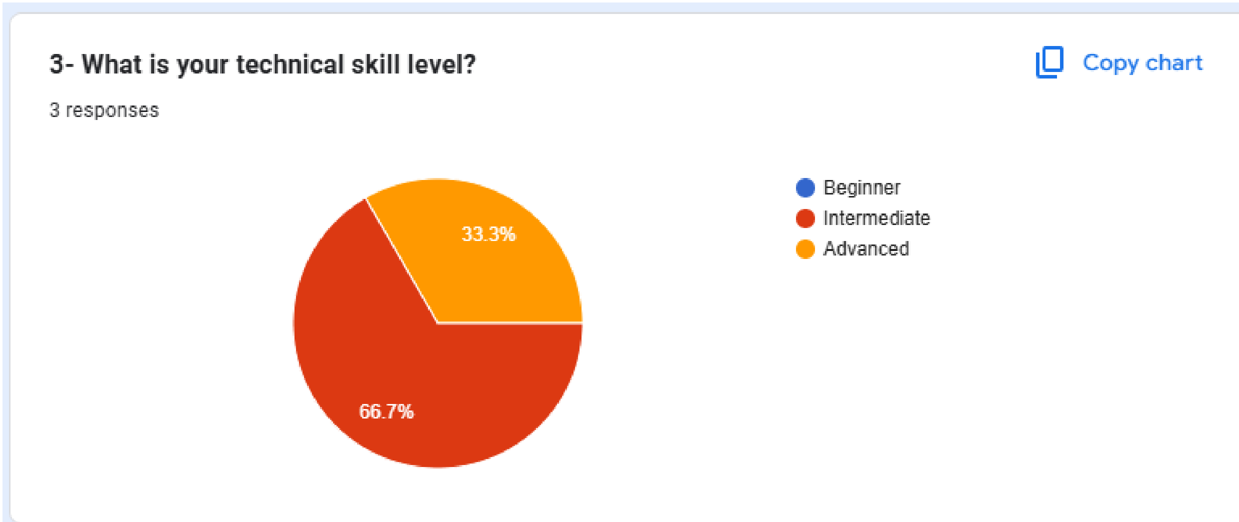


Figure 81: Usability testing survey answer three

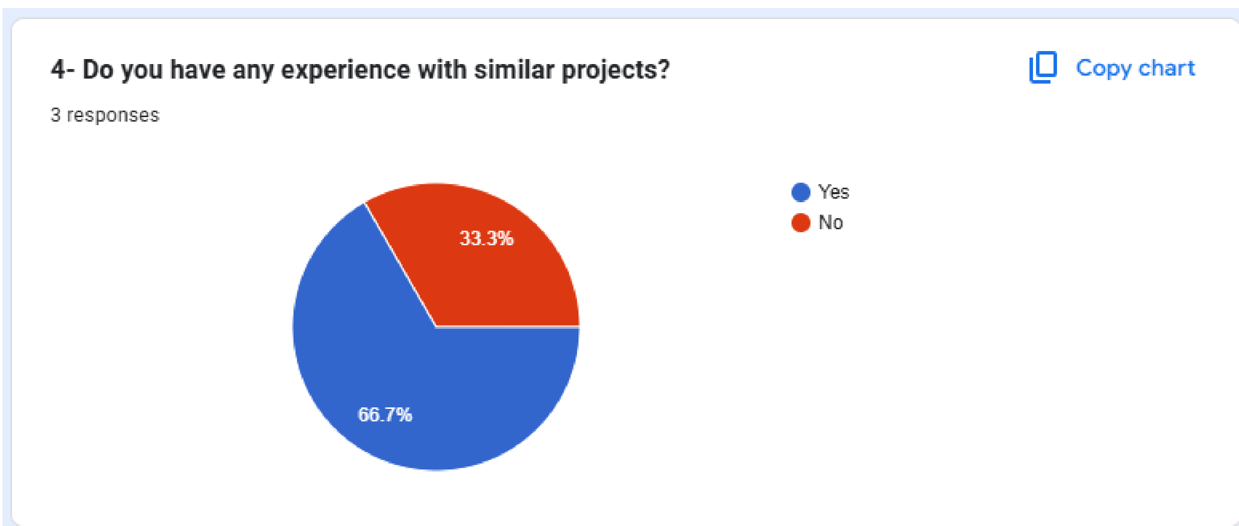


Figure 82: Usability testing survey answer four

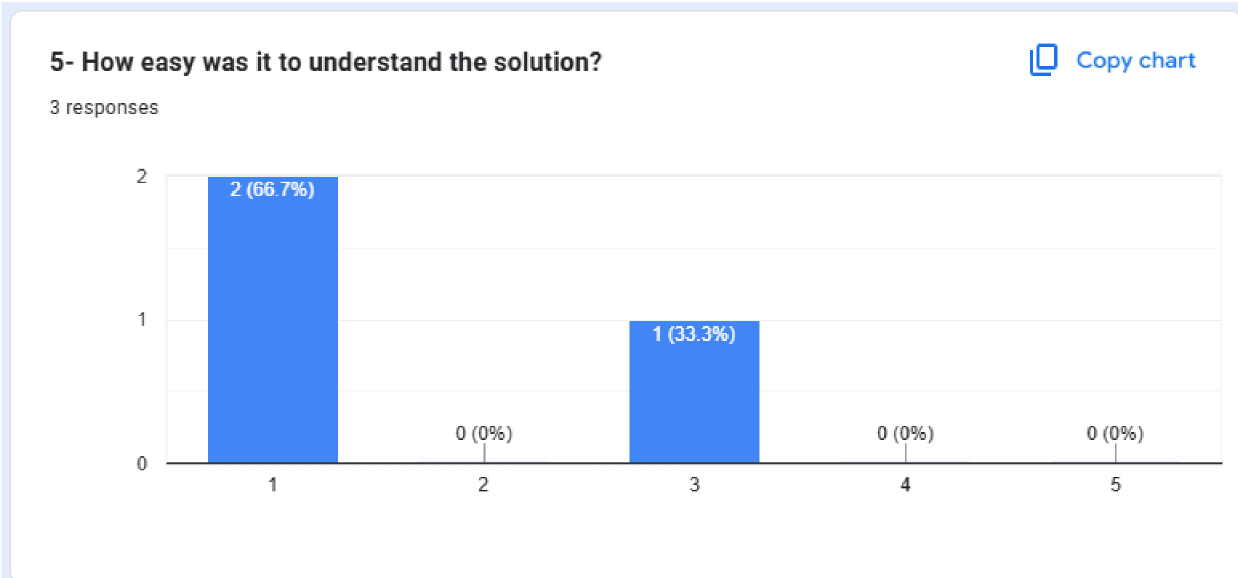


Figure 83: Usability testing survey answer five

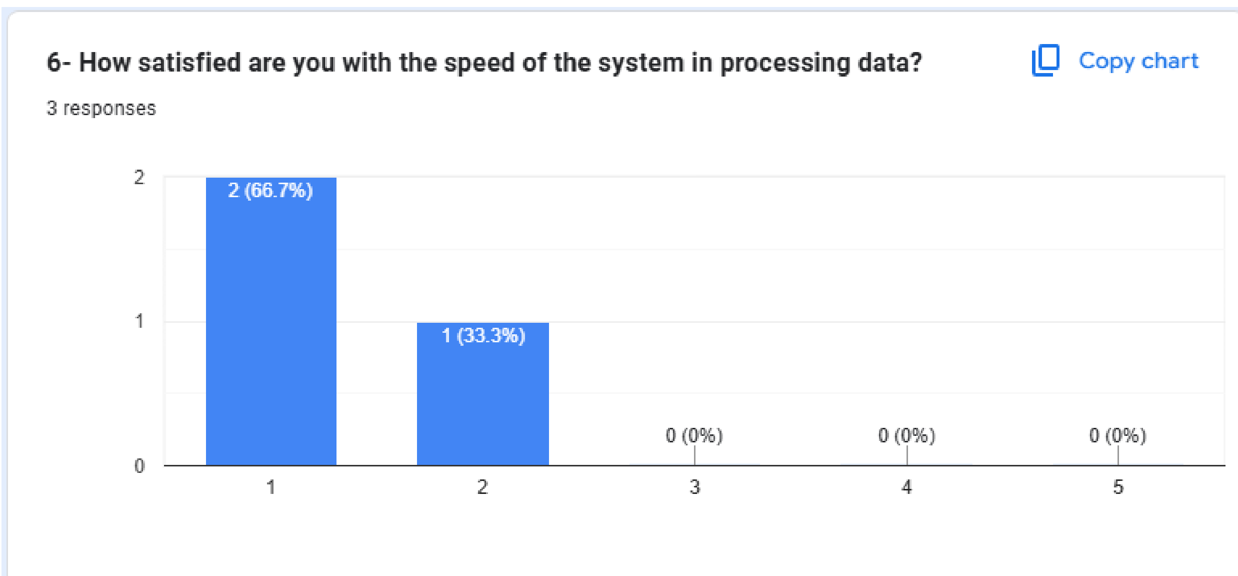


Figure 84: Usability testing survey answer six

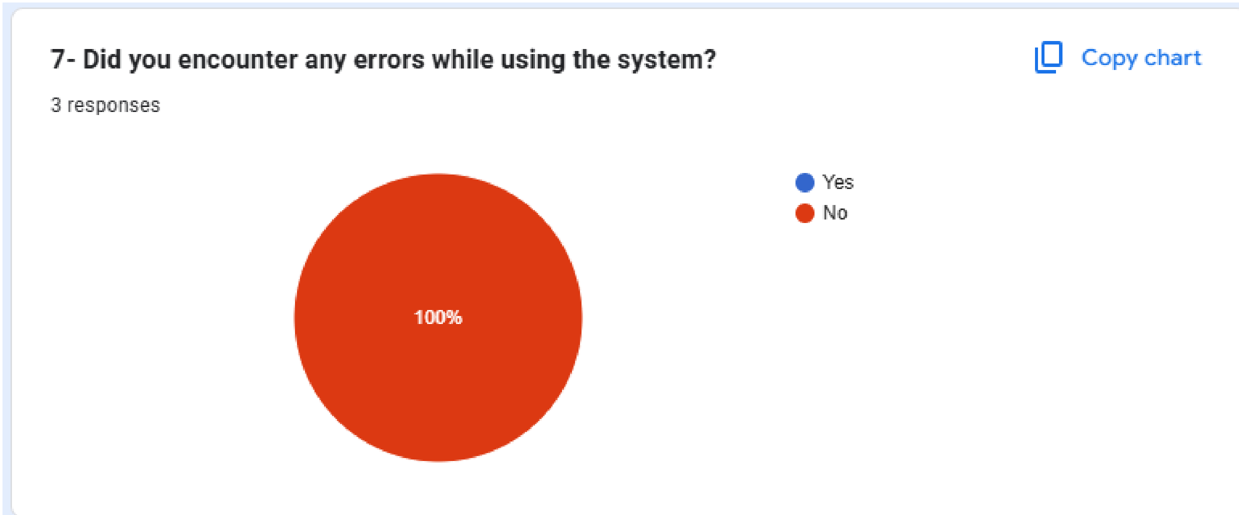
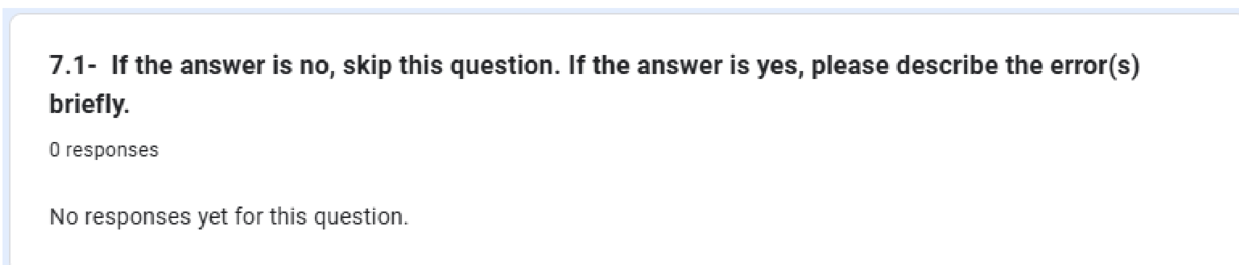


Figure 85: Usability testing survey answer seven


7.1- If the answer is no, skip this question. If the answer is yes, please describe the error(s) briefly.

0 responses

No responses yet for this question.

Figure 86: Usability testing survey extension for answer seven

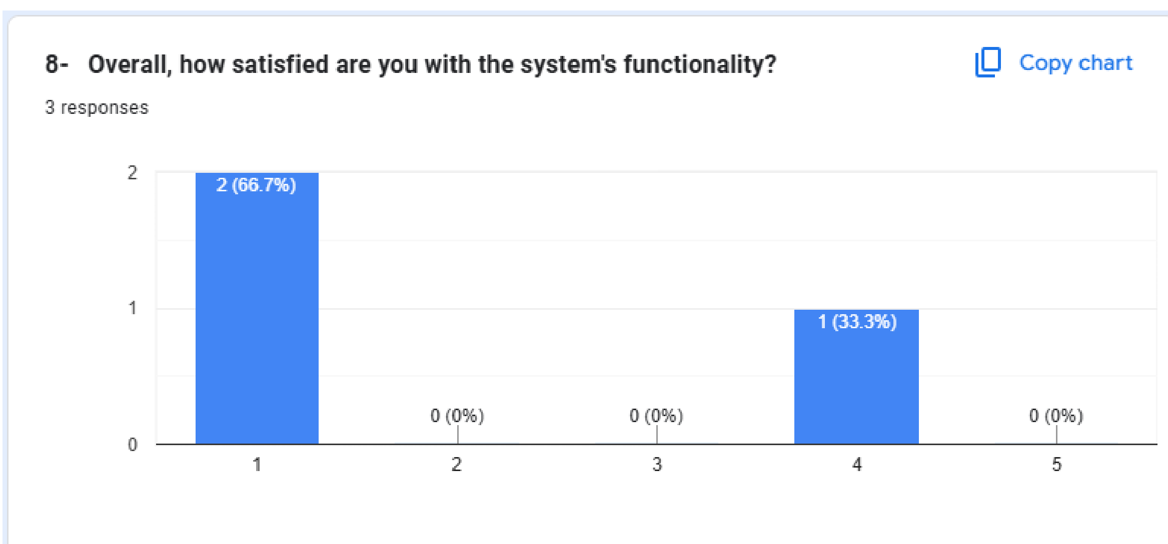


Figure 87: Usability testing survey answer eight

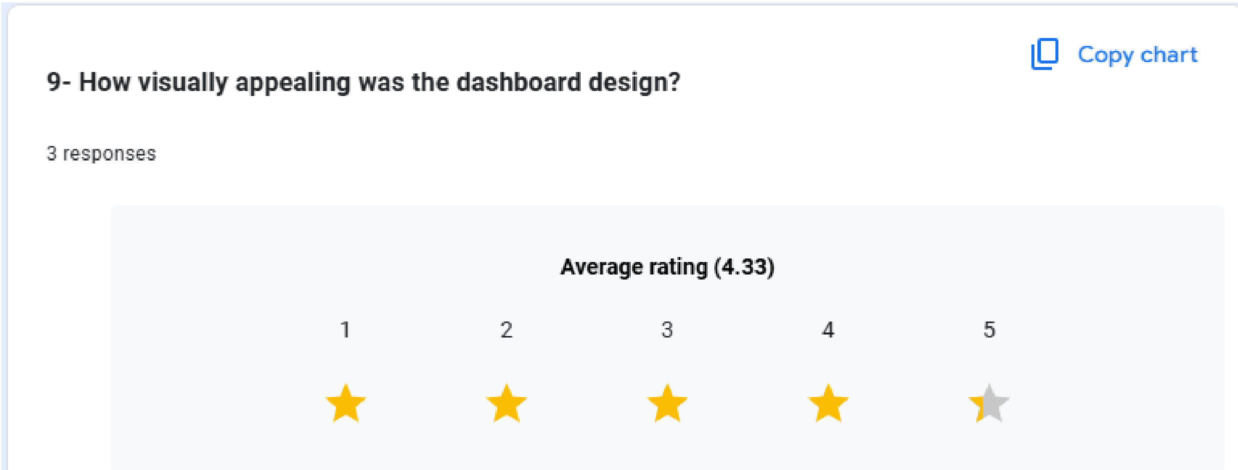
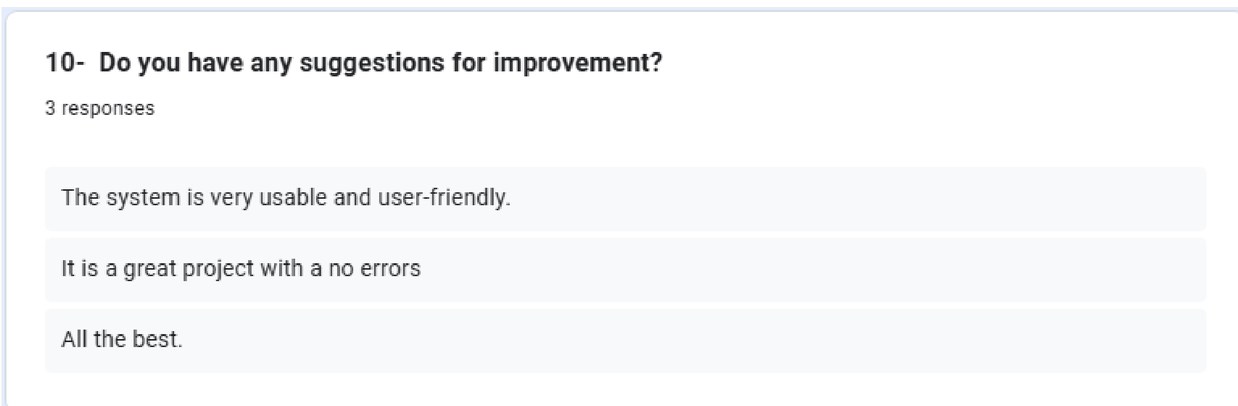


Figure 88: Usability testing survey answer nine


10- Do you have any suggestions for improvement?
3 responses

The system is very usable and user-friendly.

It is a great project with a no errors

All the best.

Figure 89: Usability testing survey answer ten

According to the usability testing survey results, all acceptance tests were passed, verifying that the system met user expectations. Moreover, the results did not show any complaints.

Discussion, LESPI, and Conclusion

Discussion

The discussion section summarizes the project's objectives, the benefits of the energy optimization algorithm, as well as reflections on the experiences and challenges faced during the development of the project. In addition to highlighting potential future advancements, this section also emphasizes the project's importance to Bahrain.

System Functionality

After applying the solution, household energy consumption decreased by 31.6%, it was 1445.01 kWh and became 988.50 kWh, achieving the primary objective of reducing total household energy usage. In this project, the focus was using linear programming optimization technique to minimize unnecessary device usage, by setting operational constraints and optimizing their operation according to occupancy, outdoor temperature, house resident usage pattern and statistics regarding the time to close devices. Certain appliances are automatically deactivated based on constraints. For example, when the weather is pleasant or when there is no one in the house, air conditioning devices are automatically turned off. Adapting appliance operation to household needs and environmental conditions offers an energy-efficient solution to reduce unnecessary consumption of energy. In many ways, this functionality represents a practical methodology for energy management, efficiently utilizing energy as needed by activating electronics only when they are needed and reducing electricity costs at the same time. Implementation of the system complies with the initial design specifications (See Design section) and demonstrates effective integration of constraints to ensure accurate and automated control over household energy consumption through the application of smart controls.

Summary of Achieved Objectives

An evaluation of the project objectives and an explanation weather the system achieved and not achieved those objectives: -

Table 6: Summary of achieved objectives

Number	Objectives	Description	Status
1	Obtain a manual dataset on household electricity consumption every hour for seven days using Microsoft Excel.	A data collection using real household energy usage pattern is collected for one week on an hourly basis and saved in Excel document.	Achieved
2	Clean and prepare the collected dataset using Python	The collected dataset is successfully cleaned from empty cells and well formatted using Python	Achieved
3	Analyze the collected dataset using different methods.	The dataset is analyzed using six different data analysis methods.	Achieved
4	Show the results in an informative visualization.	Different visual representation was used such as line chart, scatter plot and bar chart.	Achieved
5	Apply an optimization technique to reduce the total electricity utilization and produce an optimized dataset.	Linear programming optimization technique was applied, and total energy usage was reduced.	Achieved
6	Analyze the optimized dataset.	The enhanced dataset was analyzed using multiple methods.	Achieved
7	Visualize the new dataset.	The new dataset was visualized using different graphs and charts.	Achieved
8	Compare the original and enhanced dataset results.	A comparison between the two datasets was explained.	Achieved
9	Provide a reusable and scalable solution.	The solution codes can be reused on other projects with minimal modification, and the dataset can include additional households’ data.	Achieved
10	Ensure Derasat research organization is satisfied and has meaningful insights to share with interested parties.	Derasat employees are satisfied with the final solution and can share it with others.	Achieved

Project Issues and Solutions

During the development and implementation of the project, several challenges were encountered. The evaluation and explanation of the factors that contributed to the occurrence of these issues, along with solutions.

Table 7: Project issues and solutions

Number	Issues	Reasons	Method
1	The client did not provide a dataset.	Lack of Bahrani residential energy consumption dataset.	Real-life datasets collected from a Bahraini house.
2	A problem with the execution of the code.	Data caches, memory overloads, variable conflicts, and kernel instability can cause these problems.	Refresh or restart the system.
3	The difficulty of defining constraints.	The dataset was only collected for one week, which resulted in limited observation of patterns	Based on relevant world statistics, realistic constraints were estimated.
4	The occupancy constraint was not working as expected.	It likely because it depends on the model solver output and it may conflict with other constraints, causing the solver to ignore them.	Setting the occupancy constraints outside the model solver.

Backup Plan

A comprehensive backup plan is critical to maintain the integrity and availability of project data and resources in case of unexpected disasters or failures. Project-related information and systems are backed up and can be restored using different strategies, procedures, and technologies. Assuring the continuation of the project requires protecting data and mitigating potential risks. The project documents such as Jupyter Notebooks as (.ipynb) and Excel as (.xlsx) are stored in multiple locations: -

1- Data Backup Strategy

- **Local storage**: Directly stored on the primary working device.

- **OneDrive**: Additional cloud backup, automatically syncing files.
- **Secondary Device**: Project files will be stored on a secondary device in case the primary device fails.
- **GitHub**: Online version control for tracking code changes.
- **Communication with the Client**: All critical files will be emailed to the client to keep them updated.

2- System and Configuration Backup

- **Local Drive and OneDrive**: Configuration files are stored both on OneDrive and locally.
- **GitHub**: Store requirements.txt to recreate the environment in the future.
- **Secondary Device**: If the primary device is unavailable, configuration files will be saved on a second device to prevent downtime.

3- Disaster Recovery Plan

- **OneDrive**: Any device with an internet connection can access the most recent backup.
- **Secondary Device**: Access project files directly from the second device.
- **GitHub**: Restore the repository's code and environment configuration.
- **Client Access**: The client receives regular updates via email, which can be shared again.

4- Roles and Responsibilities

- **Owner of the project**: Maintains backups across all storage locations, verifies file integrity, and updates the client.
- **Client**: Serves as an additional backup recipient via regular email updates.

5- Security and Access Control

- **Local and Secondary Device**: Backups are encrypted, and physical security measures are taken.
- **GitHub and OneDrive**: Authentication and passwords are required for both.
- **Access Control**: Backups can only be accessed by the project owner and authorized personnel. Information that is sensitive is encrypted before it is stored or shared.

6- Incident Response and Communication

- **Stakeholder Communication**: Failures are notified to stakeholders via email, with a description of recovery steps and an estimation of the time needed for restoration.
- **Testing regular backups**: backups are tested to ensure that the data is up-to-date and can be accessed when necessary.

Future Work

The following suggestions would be useful in the next release of the project, along with a description of the system modifications that would be useful to add: -

1- Expand the dataset to include data from multiple households:

Information from additional households would be collected to capture a broader range of usage patterns and behaviours to improve the system's accuracy and effectiveness.

2- Add additional observational attributes:

The system can make more precise recommendations for energy optimization by integrating more environmental factors, such as humidity.

3- Extend data collection time:

Detecting seasonal patterns and long-term trends requires monitoring data over time, such as months or even years. A longer timeframe will provide deeper insight into household energy demands and enable more reliable optimization strategies to be developed.

Synopsis of your experience

This project represented my first engagement with Derasat Research Institute, which provided me with valuable insight into energy efficiency and its impact on the nation. Energy optimization techniques introduced me to the importance of combining ICT skills with engineering concepts, which will be crucial for future projects. As a result of this experience, I have gained a deeper understanding of how optimization can improve energy savings.

My experience with this project has helped me gain flexibility in my approach to data analysis, allowing me to compare and interpret the data using a variety of analytical methods. Thus, my problem-solving skills were enriched and my approach to technical challenges was widened. In the future, I will be able to take on more complex projects.

It was an important experience that helped me acquire skills and knowledge that are highly applicable to my career. I have gained a stronger understanding of optimization and data analysis. In addition, I developed project management and technical skills that will prepare me for future roles. With this solution, I am confident that I will be able to contribute to the development of sustainable and efficient systems in Bahrain and beyond.

Bahraini Perspectives

The project contributes to the establishment of a culture of sustainability in the Kingdom of Bahrain, promoting efficient resource use by monitoring and optimizing energy usage following Bahrain's Vision 2030, which is intended to promote eco-friendly development and a shift toward more mindful energy consumption in the Kingdom of Bahrain (Sustainable Energy Unit | Kingdom of Bahrain, 2017).

Considering that Bahrainis have a powerful sense of belonging to the community and caring for the environment, they will probably view this project positively, since the project will promote energy conservation, reduce costs, and align with values of responsibility which is important to Bahrainis. Most households will be willing to adopt energy-saving practices, but some may be concerned about privacy or changing their routines. However, communicating the project's benefits will help build acceptance. The awareness of this project could lead to more widespread adoption of sustainable practices, such as water conservation and recycling. Thus, creating a culture that values and preserves Bahrain's natural resources.

Legal, Ethical, Social, and Professional Issues

A list of issues that were taken into consideration: -

Legally

Keeping data private and secure is a top priority. Since the project collects and stores household energy data, it will comply with Bahrain Personal Data Protection Law No. (30) of 2018, which safeguards the rights and freedoms of individuals and their personal information (Personal Data Protection Authority | Bahrain, 2024). The house resident's personal data is therefore protected and measures such as access control are enforced to prevent unauthorized access.

Ethically

Maintaining user trust requires using collected data only for energy optimization purposes. Hence, it is ethical to avoid misuse of data, such as using it for purposes beyond those originally agreed upon. According to Bahrain's Code of Employee Conduct and Ethics, integrity, responsibility and trust are achieved through clear communication about how data is used and the protection of confidential personal information (Civil Service Bureau | Bahrain, 2014).

Socially

The project promotes environmentally friendly energy practices that positively impact Bahraini society. Bahrain's National Action Charter supports community welfare, sustainable development, and cultural values, which are aligned with the project's emphasis on energy efficiency, social responsibility, privacy, and accessibility (Government of Bahrain, 2023).

Professionally

In the ICT field, maintaining data integrity is vital for establishing trust. Any errors in data processing could damage the project's credibility and affect the reputation of ICT professionalism. Data inaccuracies cause misguided decision-making, inefficient procedures, and a loss of competitive advantage, which can result in damage to the organization's credibility (Niv Sluzki, 2024).

Conclusion

To conclude, this project contributed to a deeper understanding of data analysis and energy optimization for household appliances consumption, offering insightful knowledge to minimize energy utilization. Results show that implementing customized constraints, such as those based on occupancy and temperature conditions, reduces energy consumption without compromising occupants' comfort. Bahrain's environmental goals are being supported by this approach, as well as practical applications of costs savings and sustainability at the household level that can be applied to Bahrain.

However, the success of this project raises new questions, for instance, how additional factors, such as humidity and seasonal changes, might further improve the optimization strategies. Research could be extended to explore the integration of machine learning models that adapt to Bahraini habits over time, allowing for personalized and effective energy-saving techniques. Moreover, collecting a larger dataset over several months or years may provide more information and reliable predictions.

In practical terms, this study highlights the potential for energy conservation within homes and highlights a scalable approach that could benefit not only individual households, but also Bahrain's broader vision of sustainability as well. In the future, algorithms could be modified for households with more complex needs and upgrade the system to help people see real-time energy savings. Not only has this project laid a foundation for obtaining and analyzing household energy usage in the Kingdom of Bahrain, but it has also advanced energy efficiency, encouraging further exploration.

References

Ahmed, Mawahib Eltayeb Ahmed, Ahmed, Q. A., & Asia Adlan Mohamed. (2024). A review of electricity consumption and CO_2 emissions in Gulf Cooperation Council households and proposed scenarios for its reduction. *Arab Gulf Journal of Scientific Research*. https://doi.org/10.1108/agjsr-09-2022-0188

Alooba. (2024). Systems Architecture: Everything You Need to Know When Assessing Systems Architecture Skills. Retrieved November 2, 2024, from Alooba.com website: https://www.alooba.com/skills/concepts/systems-architecture/

Appinio. (2023, October 30). What is Comparative Analysis and How to Conduct It? | Appinio Blog. Retrieved November 4, 2024, from Appinio.com website: https://www.appinio.com/en/blog/market-research/comparative-analysis

Alwan, G. (2016). Optimization Technique. Retrieved October 30, 2024, from ResearchGate website: https://www.researchgate.net/publication/330204775_Optimization_Technique

Alhumaidan, F. (2012). A Critical Analysis and Treatment of Important UML Diagrams Enhancing Modeling Power. Intelligent Information Management, 04(05), 231–237. https://doi.org/10.4236/iim.2012.45034

Bahrain News Agency. (2023, August 7). Met. Dir.: July 2023 second hottest in Bahrain since 1902. Retrieved October 23, 2024, from https://www.bna.bh/en/ website: https://www.bna.bh/en/Met.Dir.July2023secondhottestinBahrainsince1902.aspx?cms=q8FmFJgiscL2fwIzON1%2BDsUasvrtOnCOuOJHD1fmwUY%3D

Brooks, R. (2021, September 16). What are data structures? - North Wales Management School - Wrexham University. Retrieved November 1, 2024, from North Wales Management School - Wrexham University website: https://online.wrexham.ac.uk/what-are-data-structures/

Bhandari, P. (2020, July 9). Descriptive Statistics | Definitions, Types, Examples. Retrieved November 4, 2024, from Scribbr website: https://www.scribbr.com/statistics/descriptive-statistics/

Civil Service Bureau | Bahrain. (2024). Civil Service Bureau | Kingdom of Bahrain. In Civil Service Agency. Retrieved from https://www.csb.gov.bh/MediaManager/Media/Documents/CodeOfConduct-En.pdf

Codecademy. (2024). Pseudocode and Flowcharts. Retrieved November 2, 2024, from Codecademy website: https://www.codecademy.com/article/pseudocode-and-flowcharts

Challa, N. (2023). DATA ANALYTICS AND ITS IMPACT ON FUTURE. Retrieved November 5, 2024, from ResearchGate website: https://www.researchgate.net/publication/371665415_DATA_ANALYTICS_AND_ITS_IMPACT_ON_FUTURE

Dickson, C. (2024, August 6). Should you do laundry every day? Laundry pros share their thoughts. Retrieved November 8, 2024, from homesandgardens.com website: https://www.homesandgardens.com/solved/should-you-do-laundry-every-day

Derasat. (2023, October 17). About Derasat. Retrieved October 30, 2024, from Derasat website: https://www.derasat.org.bh/about-derasat/

Dantzig, G. B. (2002). Linear Programming. Operations Research, 50(1), 42–47. https://doi.org/10.1287/opre.50.1.42.17798

Emmanuel. (2023, April 13). Inferential Statistics: Definition, Types + Examples. Retrieved November 4, 2024, from Formplus website: https://www.formpl.us/blog/inferential-statistics-definition-types-examples

Electricity and Water Authority | Bahrain. (2019). Electricity and Water Tariffs. Retrieved November 21, 2024, from Www.ewa.bh website: https://www.ewa.bh/en/Customer/BillsTariffs/electricity-water-tariffs

Grundy, R. (2023, June 27). Advantages and disadvantages of using spreadsheets for your business. Retrieved October 31, 2024, from Sage Advice United Kingdom website: https://www.sage.com/en-gb/blog/speed-up-business-processes/

Government of Bahrain. (2023). National Action Charter. Retrieved November 12, 2024, from Bahrain.bh website: https://www.bahrain.bh/wps/portal/en/!ut/p/z0/fY69DoIwFEZfpQvzrRKB1Z8BHYQRupBLaeAK3kKpxse3Gme3c5IvXw4oqEAxPqlHT5ZxCl6rpNmcdzLPjlLm1_IkkyLP0m16iIsyhguo_4PwQLdlUXtQ2rI3Lw9Vy3NjOJLY2ocXfjBiJO47e49ki4NDYhKT6Wmdvh1rJJ0JaDrh7Uw6OP8CBeoPCD2g88bBPKr6DUV5FxU!/

Gulf Daily News. (2019, September 30). Solar panels to be installed at eight public schools across Bahrain. Retrieved October 27, 2024, from Zawya.com website: https://www.zawya.com/en/business/solar-panels-to-be-installed-at-eight-public-schools-across-bahrain-vrddo27w

Jain, P. (2020, May 26). What Is Data Analysis: Examples, Types, & Applications. Retrieved October 30, 2024, from Simplilearn.com website: https://www.simplilearn.com/data-analysis-methods-process-types-article

Kitto, J. (2018, June 15). How to migrate from spreadsheets to a DBMS. Retrieved October 31, 2024, from Codebots website: https://codebots.com/continuous-modernisation/how-to-migrate-from-spreadsheets-to-a-dbms

Lawrence, C. (2024, September 27). This is how often you should be vacuuming your floors — from an expert. Retrieved November 8, 2024, from Tom's Guide website: https://www.tomsguide.com/home/this-is-how-often-you-should-actually-be-vacuuming-your-floors

Lucidchart. (2024). What is a Flowchart. Retrieved November 2, 2024, from Lucidchart website: https://www.lucidchart.com/pages/what-is-a-flowchart-tutorial

Microsoft. (2024). Power BI - Data Visualization | Microsoft Power Platform. Retrieved November 1, 2024, from Microsoft.com website: https://www.microsoft.com/en-us/power-platform/products/power-bi/

Meteostat. (2024). The Weather's Record Keeper | Meteostat. Retrieved October 31, 2024, from Meteostat.net website: https://meteostat.net/en/place/bh/al-hadd?s=41150&t=2024-10-17/2024-10-24

Ministry of Information | Bahrain. (2024). Population and demographics. Retrieved October 22, 2024, from https://www.mia.gov.bh/kingdom-of-bahrain/population-and-demographics/?lang=en

Ministry of Finance and National Economy | Bahrain. (2023). Bahrain Economic Report 2023. Retrieved from https://www.mofne.gov.bh/media/td2n1wia/beq_v6_en.pdf

Ministry of Works | Bahrain. (2008, October 25). MINISTRY OF WORKS: DISTRICT COOLING). Retrieved October 27, 2024, from Works.gov.bh website: https://www.works.gov.bh/English/News/mownews/Pages/2008/503.aspx

Niv Sluzki. (2024, August 30). Data integrity strategy. Retrieved November 12, 2024, from Ibm.com website: https://www.ibm.com/think/insights/data-integrity-strategy

Personal Data Protection Authority | Bahrain. (2024). About Personal Data Protection Authority | Kingdom of Bahrain. Retrieved November 12, 2024, from Pdp.gov.bh website: http://www.pdp.gov.bh/en/about-PDPA.html

Python. (2024). Download Python. Retrieved November 3, 2024, from Python.org website: https://www.python.org/downloads/

Pandian, S. (2021, October 23). Time Series Analysis: Definition, Components, Methods, and Applications. Retrieved November 4, 2024, from Analytics Vidhya website: https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-to-time-series-analysis/

Rosen, M., & Farsi, A. (2022). Sustainable Energy Technologies for Seawater Desalination. Elsevier. Available at: https://www.sciencedirect.com/book/9780323998727/sustainable-energy-technologies-for-seawater-desalination

Ravindran, A., Ragsdell, K. M., & Reklaitis, G. V. (2006). Engineering Optimization: Methods and Applications (2nd ed.). John Wiley & Sons. Retrieved from: https://www.wiley.com/en-us/Engineering+Optimization%3A+Methods+and+Applications%2C+2nd+Edition-p-9780471558149

Stavrev, S., & Ginchev, D. (2024). Reinforcement Learning Techniques in Optimizing Energy Systems. MDPI, 13(8), 1459. https://doi.org/10.3390/electronics13081459

Statista Research Department. (2023, September 28). MENA: average TV viewing time by country 2020 | Statista. Retrieved November 8, 2024, from Statista website: https://www.statista.com/statistics/1274099/mena-average-tv-viewing-time-by-country/

Silverio, M. (2023, September 19). What Is Jupyter Notebook? | Built In. Retrieved November 9, 2024, from Built In website: https://builtin.com/software-engineering-perspectives/jupyter-notebook

Samarth, V. (2023, February 11). Professional Certificate in Data Engineering with Microsoft Azure. Retrieved November 5, 2024, from Emeritus India website: https://emeritus.org/in/learn/data-science-classification-analysis/

Sunbird. (2021). What Is a Network Topology Diagram? Retrieved November 2, 2024, from Sunbirddcim.com website: https://www.sunbirddcim.com/glossary/network-topology-diagram

Sustainable Energy Unit | Kingdom of Bahrain. (2017). The Kingdom of Bahrain National Energy Efficiency Action Plan (NEEAP). Retrieved from https://www.undp.org/sites/g/files/zskgke326/files/2022-09/seu_neeap_report_2017.pdf

Smith, J., & Caner Taşkın, Z. (2007). A Tutorial Guide to Mixed-Integer Programming Models and Solution Techniques. Retrieved from https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=3dba4ce1e3265e480da88f76d00e53909e775513

Visual Paradigm. (2024). What is Deployment Diagram? Retrieved November 2, 2024, from Visual-paradigm.com website: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-deployment-diagram/#:~:text=Deployment%20diagrams%20is%20a%20kind,(topology%20of%20the%20hardware).

Weather Atlas (2024, May 24). Yearly & Monthly weather - Bahrain. Retrieved November 8, 2024, from Weather Atlas website: https://www.weather-atlas.com/en/bahrain-climate

W3schools. (2024). Data Science & Python. Retrieved October 30, 2024, from W3schools.com website: https://www.w3schools.com/datascience/ds_python.asp

Webster, W. (2020, August 24). What Is Cluster Analysis? Overview and examples. Retrieved November 4, 2024, from Qualtrics website: https://www.qualtrics.com/experience-management/research/cluster-analysis/

Appendices

Appendix I: System and Client Manuals

The project includes two manuals, one for the client's administrator and one for the client’s researcher.

First, the final product will be in a zipped folder containing all work files:

1. Right-click on the zipped folder “202000270_Final_Product”.

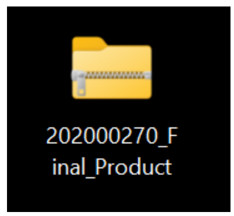


Figure 90: The zipped folder “202000270_Final_Producut”

2. Click on “Extract All…”.

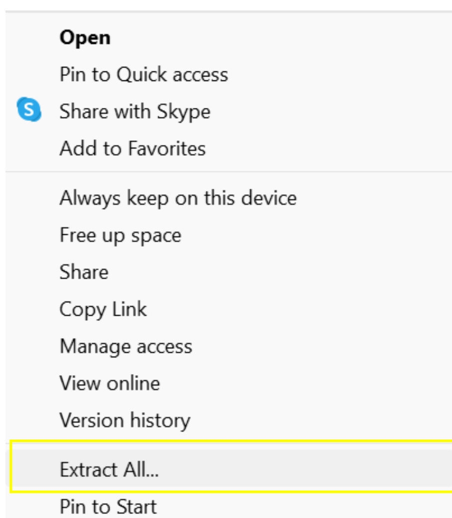


Figure 91: Clicking on “Extract All…”

3. The following window will be displayed, click on “Extract” button.

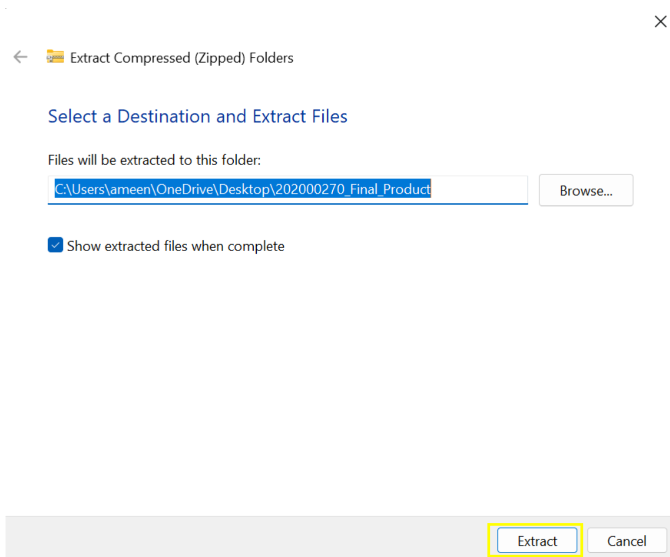


Figure 92: Clicking on the “Extract” button

4. Click on the extracted folder “202000270_Final_Product”.

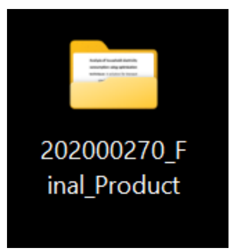


Figure 93: The folder “202000270_Final_Product”

5. Click on the “FinalProduct” folder.

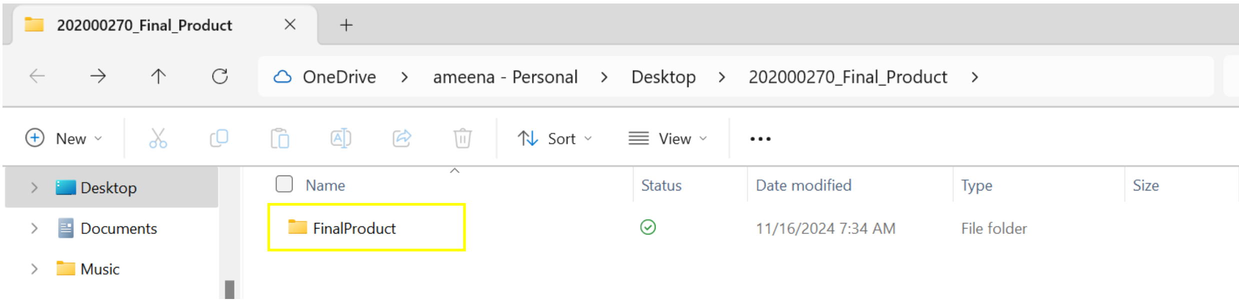


Figure 94: Inside the “202000270_Final_Product” folder

6. Inside the “FinalProduct” folder, the “EnergyProduct” and the “EnergyDashboard” folders.

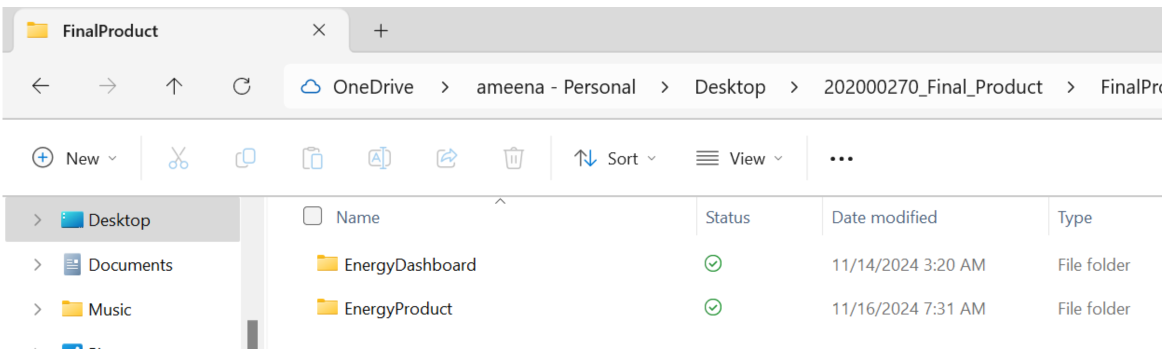


Figure 95: Inside the “FinalProduct” folder.

Manual One

This manual is for the client's administrator to manage and access all the following files inside the “EnergyProduct” folder.

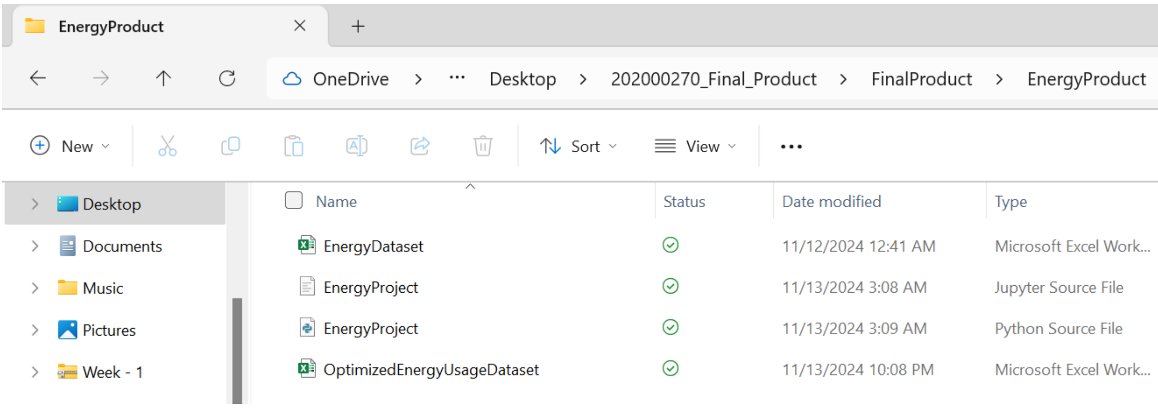


Figure 96: Inside the “EnergyProduct” folder

The following are the files:-

- **EnergyDataset.xlsx**: The admin can view and analyze the raw household electricity usage data contained in this Microsoft Excel file. Features of Excel include easy access, a structured format that is easy to understand, and adaptability for adding new data.

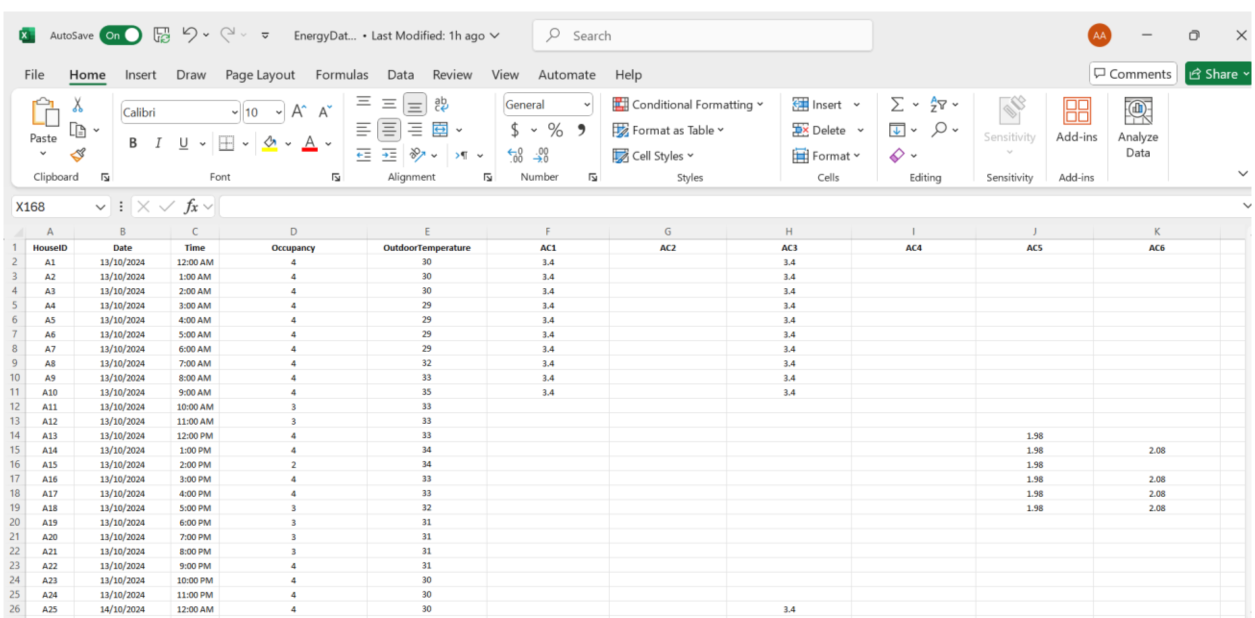


Figure 97: A screenshot of the collected dataset

- **EnergyProject.ipynb**: Data analysis and optimization code are included as a main Jupyter Notebook source file. It is possible to divide the code into different cells and handle errors that occur.

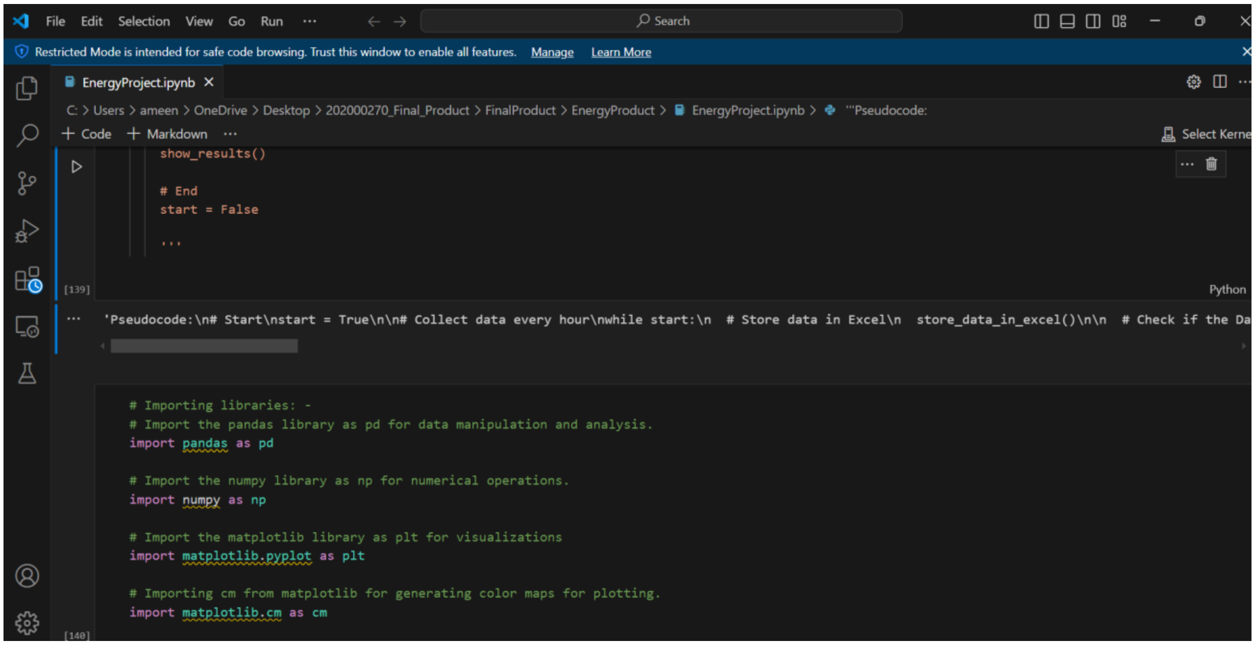


Figure 98: A screenshot of the code in (.ipynb)

- **EnergyProject.py**: Python source file, allows the code to be run without Jupyter Notebook and processes data quickly.

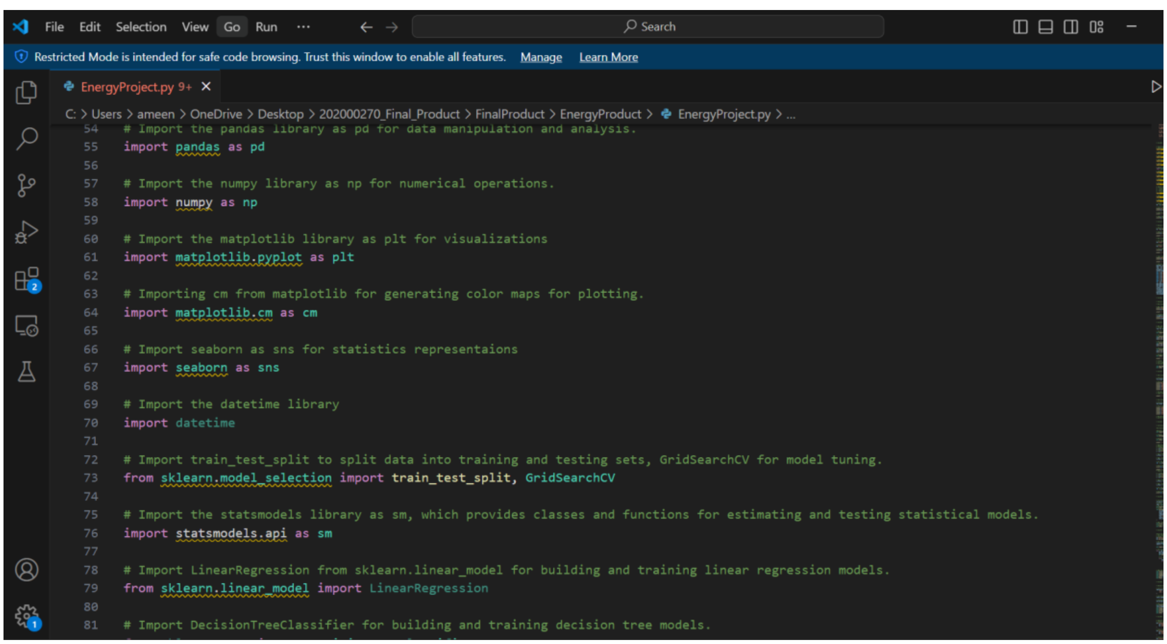
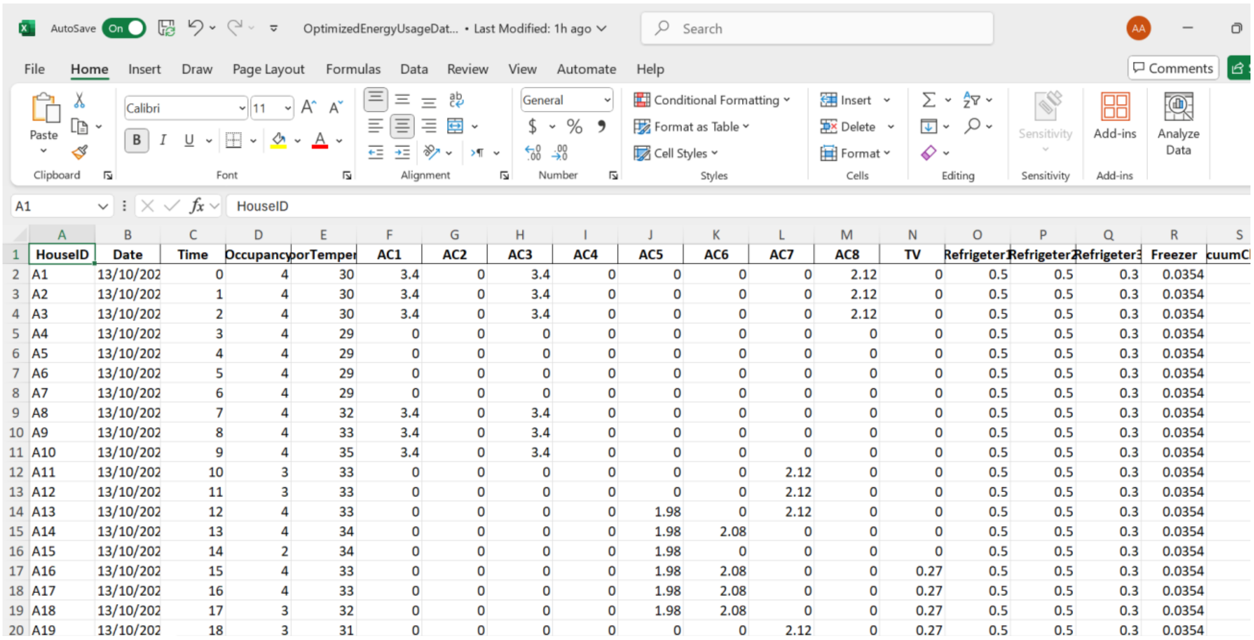


Figure 99: A screenshot of the code in (.py)

- **OptimizedDataset.xlsx**: This Excel file shows the modified data after the optimization process, allowing administrator to compare it with the original. Like the original dataset layout for easy comparison.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	HouseID	Date	Time	Occupancy	porTemper	AC1	AC2	AC3	AC4	AC5	AC6	AC7	AC8	TV	Refrigeter1	Refrigeter2	Refrigeter3	Freezer	cuumCl
2	A1	13/10/202	0	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	
3	A2	13/10/202	1	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	
4	A3	13/10/202	2	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	
5	A4	13/10/202	3	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
6	A5	13/10/202	4	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
7	A6	13/10/202	5	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
8	A7	13/10/202	6	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
9	A8	13/10/202	7	4	32	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
10	A9	13/10/202	8	4	33	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
11	A10	13/10/202	9	4	35	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	
12	A11	13/10/202	10	3	33	0	0	0	0	0	0	2.12	0	0	0.5	0.5	0.3	0.0354	
13	A12	13/10/202	11	3	33	0	0	0	0	0	0	2.12	0	0	0.5	0.5	0.3	0.0354	
14	A13	13/10/202	12	4	33	0	0	0	0	1.98	0	2.12	0	0	0.5	0.5	0.3	0.0354	
15	A14	13/10/202	13	4	34	0	0	0	0	1.98	2.08	0	0	0	0.5	0.5	0.3	0.0354	
16	A15	13/10/202	14	2	34	0	0	0	0	1.98	0	0	0	0	0.5	0.5	0.3	0.0354	
17	A16	13/10/202	15	4	33	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	
18	A17	13/10/202	16	4	33	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	
19	A18	13/10/202	17	3	32	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	
20	A19	13/10/202	18	3	31	0	0	0	0	0	0	2.12	0	0.27	0.5	0.5	0.3	0.0354	

Figure 100: A screenshot of the optimized dataset

Manual Two

In this guide, the client researcher will be able to navigate Power BI and compare old and enhanced datasets inside "EnergyDashboard" folder.

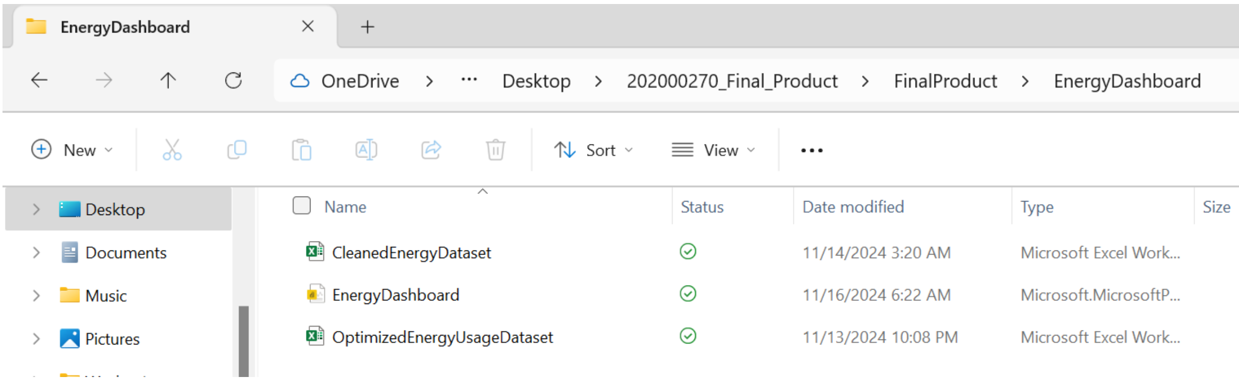


Figure 101: Inside the "EnergyDashboard" folder

The following are the files: -

- **EnergyDashboard.pbix:** Visualize the difference between cleaned and optimized datasets using the Power BI Dashboard. Views can be easily customized, visuals are interactive, and data changes are automatically updated.

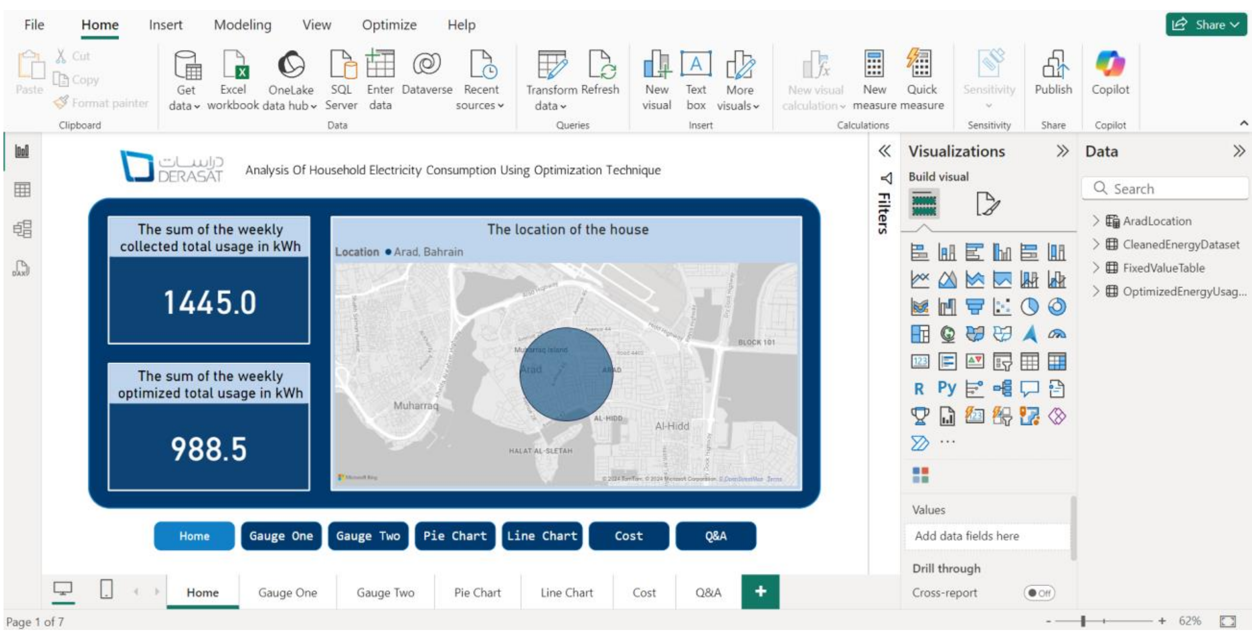
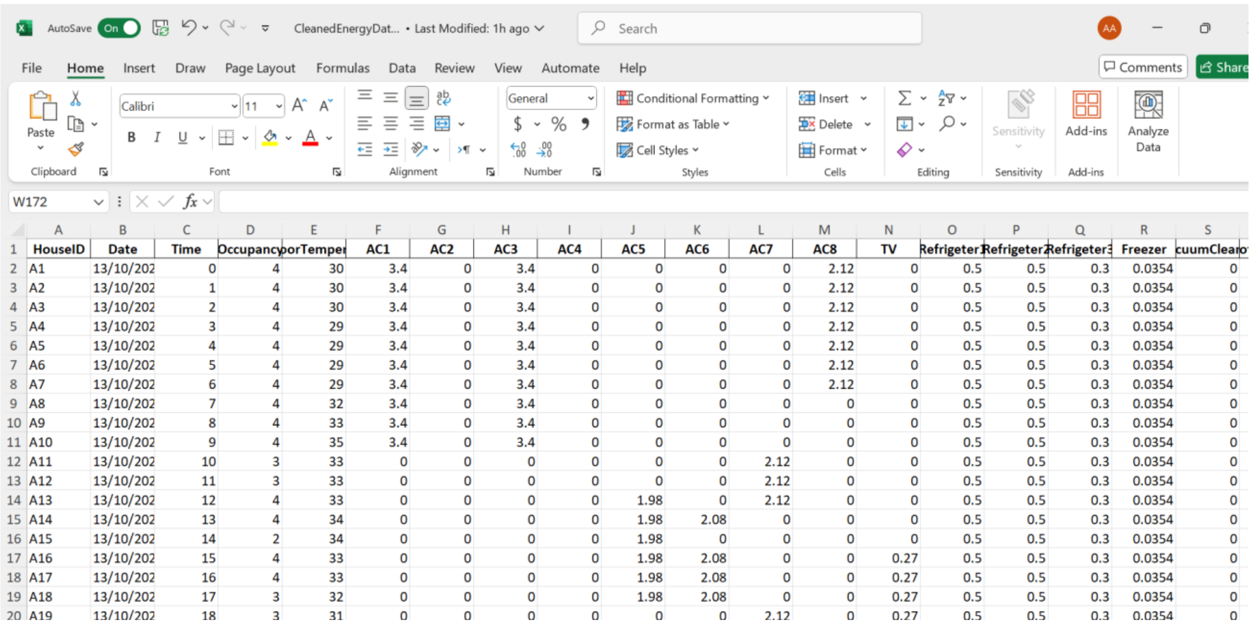


Figure 102: A screenshot of the "EnegyDashboard" file

- **CleanedEnergyDataset.xlsx:** The original imported dataset after cleaning and preparation as shown below, to start working immediately.

```
dt.to_excel("C:/Users/ameen/OneDrive/Desktop/EnergyDatasetsDashboard/CleanedEnergyDataset.xlsx", index=False)
```

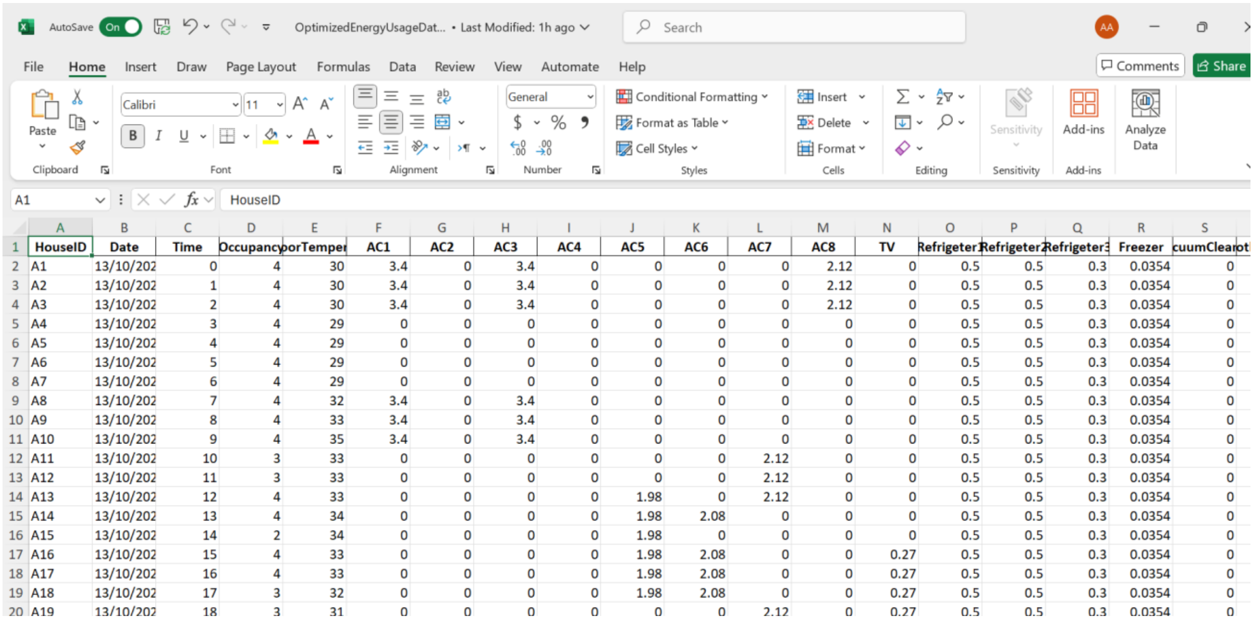
Figure 103: Import the cleaned original dataset in Python



HouseID	Date	Time	Occupancy	porTemper	AC1	AC2	AC3	AC4	AC5	AC6	AC7	AC8	TV	Refrigeter1	Refrigeter2	Refrigeter3	Freezer	cuumClea
A1	13/10/202	0	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A2	13/10/202	1	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A3	13/10/202	2	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A4	13/10/202	3	4	29	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A5	13/10/202	4	4	29	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A6	13/10/202	5	4	29	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A7	13/10/202	6	4	29	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
A8	13/10/202	7	4	32	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
A9	13/10/202	8	4	33	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
A10	13/10/202	9	4	35	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
A11	13/10/202	10	3	33	0	0	0	0	0	0	2.12	0	0	0.5	0.5	0.3	0.0354	0
A12	13/10/202	11	3	33	0	0	0	0	0	0	2.12	0	0	0.5	0.5	0.3	0.0354	0
A13	13/10/202	12	4	33	0	0	0	0	1.98	0	2.12	0	0	0.5	0.5	0.3	0.0354	0
A14	13/10/202	13	4	34	0	0	0	0	1.98	2.08	0	0	0	0.5	0.5	0.3	0.0354	0
A15	13/10/202	14	2	34	0	0	0	0	1.98	0	0	0	0	0.5	0.5	0.3	0.0354	0
A16	13/10/202	15	4	33	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	0
A17	13/10/202	16	4	33	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	0
A18	13/10/202	17	3	32	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	0
A19	13/10/202	18	3	31	0	0	0	0	0	0	2.12	0	0.27	0.5	0.5	0.3	0.0354	0

Figure 104: A screenshot of the cleaned collected dataset

- **OptimizedEnergyUsageDataset.xlsx:** A copy of the previous optimized Excel file with the modified data after the optimization step, allowing researcher to compare it with the cleaned dataset. Same as the original dataset layout for easy comparison.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	HouseID	Date	Time	Occupancy	orTemper	AC1	AC2	AC3	AC4	AC5	AC6	AC7	AC8	TV	Refrigeter1	Refrigeter2	Refrigeter3	Freezer	cuumClear
2	A1	13/10/202	0	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
3	A2	13/10/202	1	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
4	A3	13/10/202	2	4	30	3.4	0	3.4	0	0	0	0	2.12	0	0.5	0.5	0.3	0.0354	0
5	A4	13/10/202	3	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
6	A5	13/10/202	4	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
7	A6	13/10/202	5	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
8	A7	13/10/202	6	4	29	0	0	0	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
9	A8	13/10/202	7	4	32	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
10	A9	13/10/202	8	4	33	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
11	A10	13/10/202	9	4	35	3.4	0	3.4	0	0	0	0	0	0	0.5	0.5	0.3	0.0354	0
12	A11	13/10/202	10	3	33	0	0	0	0	0	0	2.12	0	0	0.5	0.5	0.3	0.0354	0
13	A12	13/10/202	11	3	33	0	0	0	0	0	0	2.12	0	0	0.5	0.5	0.3	0.0354	0
14	A13	13/10/202	12	4	33	0	0	0	0	1.98	0	2.12	0	0	0.5	0.5	0.3	0.0354	0
15	A14	13/10/202	13	4	34	0	0	0	0	1.98	2.08	0	0	0	0.5	0.5	0.3	0.0354	0
16	A15	13/10/202	14	2	34	0	0	0	0	1.98	0	0	0	0	0.5	0.5	0.3	0.0354	0
17	A16	13/10/202	15	4	33	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	0
18	A17	13/10/202	16	4	33	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	0
19	A18	13/10/202	17	3	32	0	0	0	0	1.98	2.08	0	0	0.27	0.5	0.5	0.3	0.0354	0
20	A19	13/10/202	18	3	31	0	0	0	0	0	0	2.12	0	0.27	0.5	0.5	0.3	0.0354	0

Figure 105: A screenshot of the optimized dataset

Appendix III: Detailed Design

A dashboard is created using the Microsoft Power BI desktop to get more insight by comparing before and after data optimization. The dashboard consist of seven pages. The text font type is mainly “Din”, and the font size for headers is 20 pt. The following are the design description for the dashboard elements in detail.

The color scheme inspired by the client logo colors: -

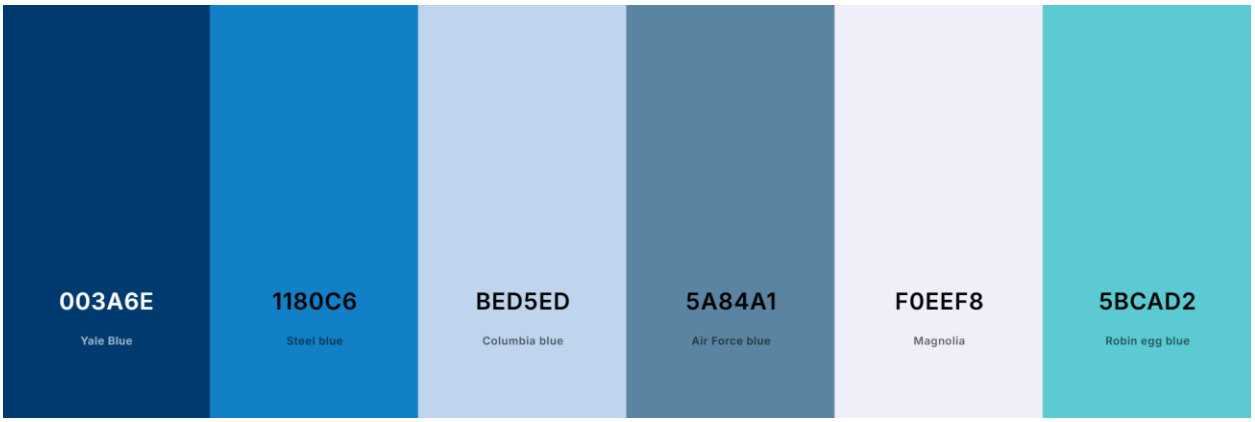


Figure 106: The dashboard color scheme

All pages have navigation buttons: -

Figure 107: A screenshot of the navigation buttons

The state of the navigation buttons:-

1. Up state: -

Figure 108: Navigation button up state

2. Hover state: -

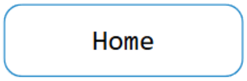


Figure 109: Navigation button hover state

3. Active state: -

Figure 110: Navigation button active state

4. Down state: -

Figure 111: Navigation button down state

- **Home page:** The page consists of two cards that show the total electricity consumption in kWh before and after optimization and a map for the location of the household, with a bubble that shows the area where the house is located.

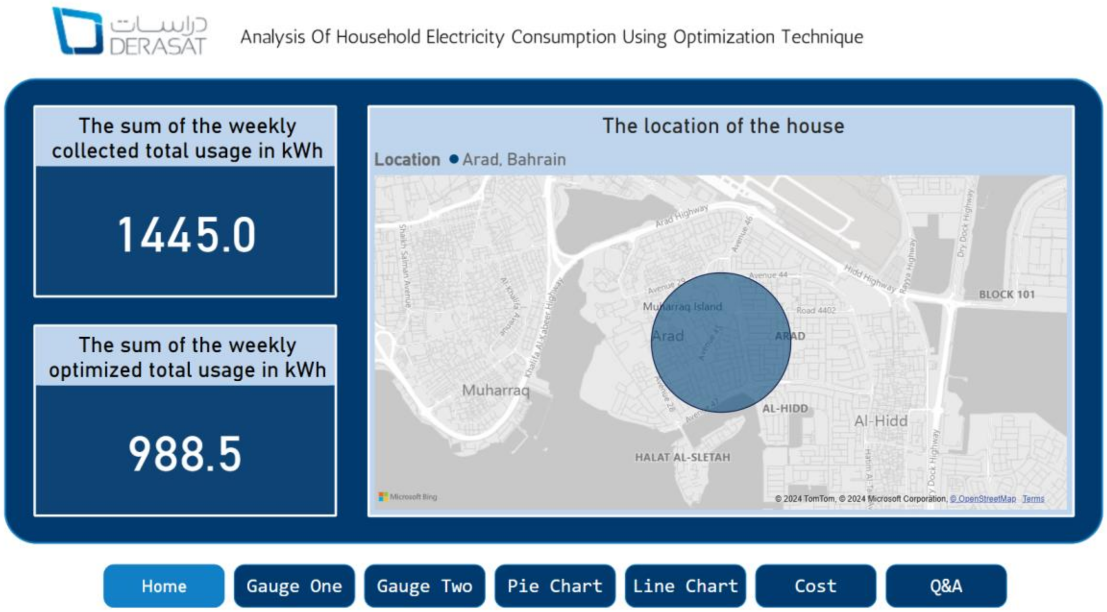


Figure 112: The dashboard “Home” page

- **Gauge One page:** The page includes a gauge for the total energy consumption of AC8 before optimization, which can be affected by four slicers: occupancy, outside temperature, hour of the day and date.

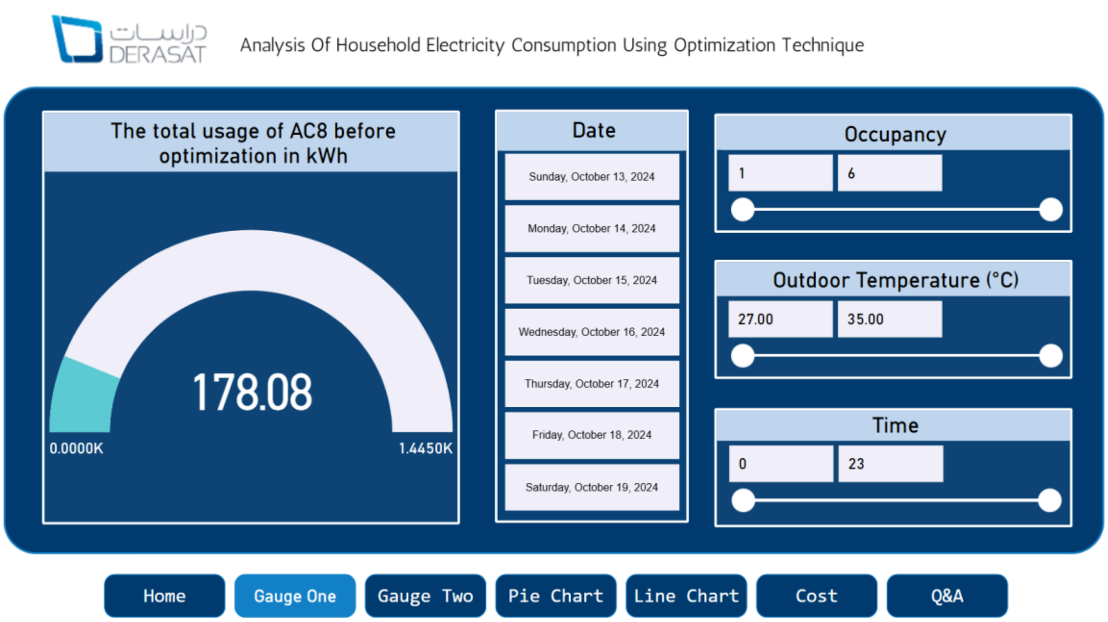


Figure 113: The dashboard "Gauge One" page

The state of the slicer buttons:-

1. Up state: -

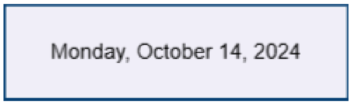


Figure 114: Slicer button up state

2. Down state: -

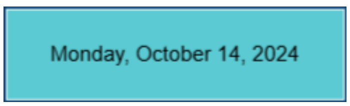


Figure 115: Slicer button down state

An example of how to use the slicers, which changed the total usage of AC8: -

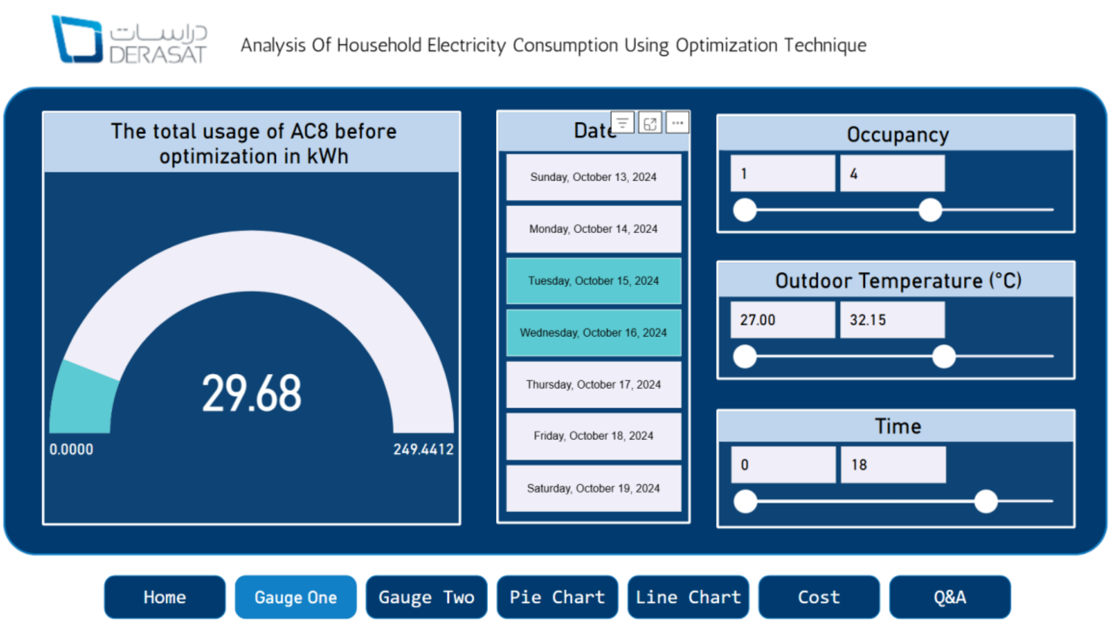


Figure 116: The dashboard "Gauge One" page, after using the slicers

- **Gauge Two page:** The page involves a gauge for the total energy usage of AC8 after optimization, which can also be affected by four slicers: occupancy, outside temperature, hour of the day and date.

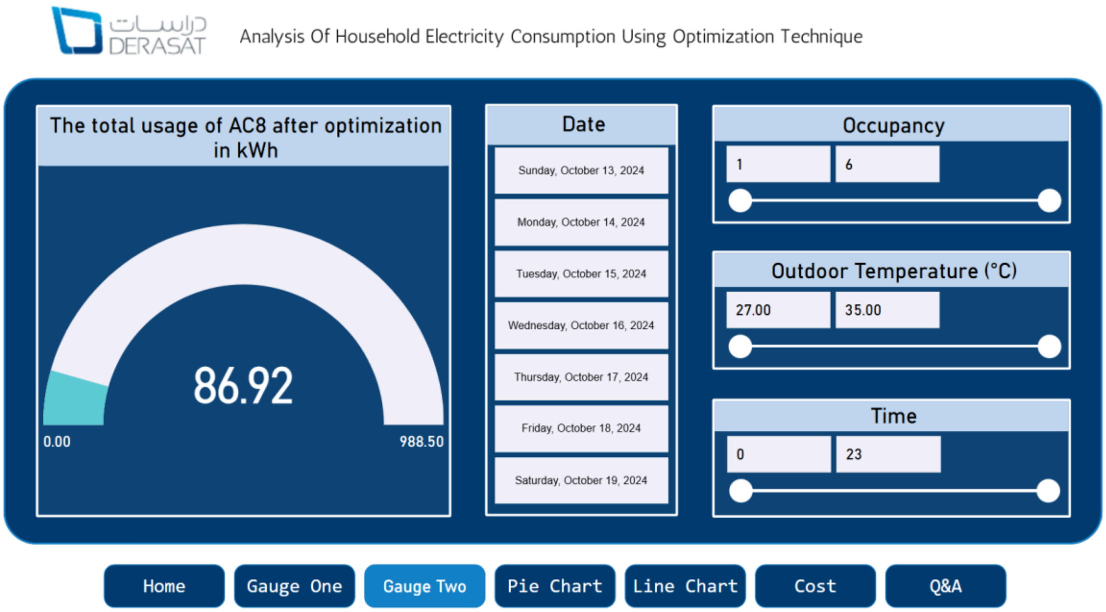


Figure 117: The dashboard "Gauge Two" page

- **Pie Chart page:** A comparison pie charts between the before and after optimization appliance usage of electricity.

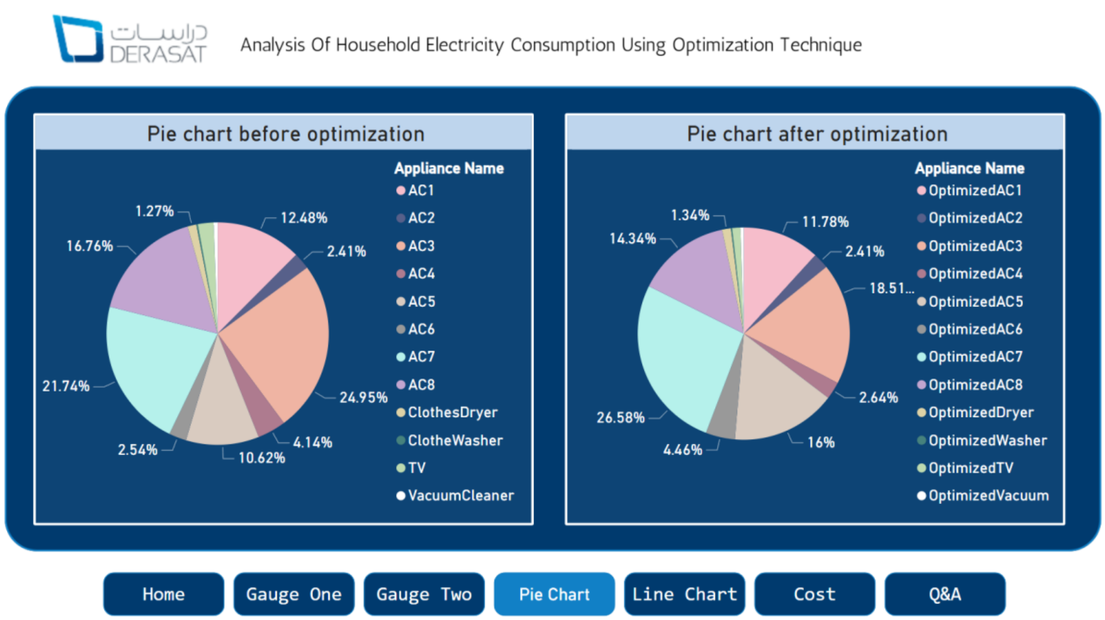


Figure 118: The dashboard "Pie Chart" page

- **Line Chart page:** A comparison of line charts between the two datasets to get insights on the consumption of energy over time.

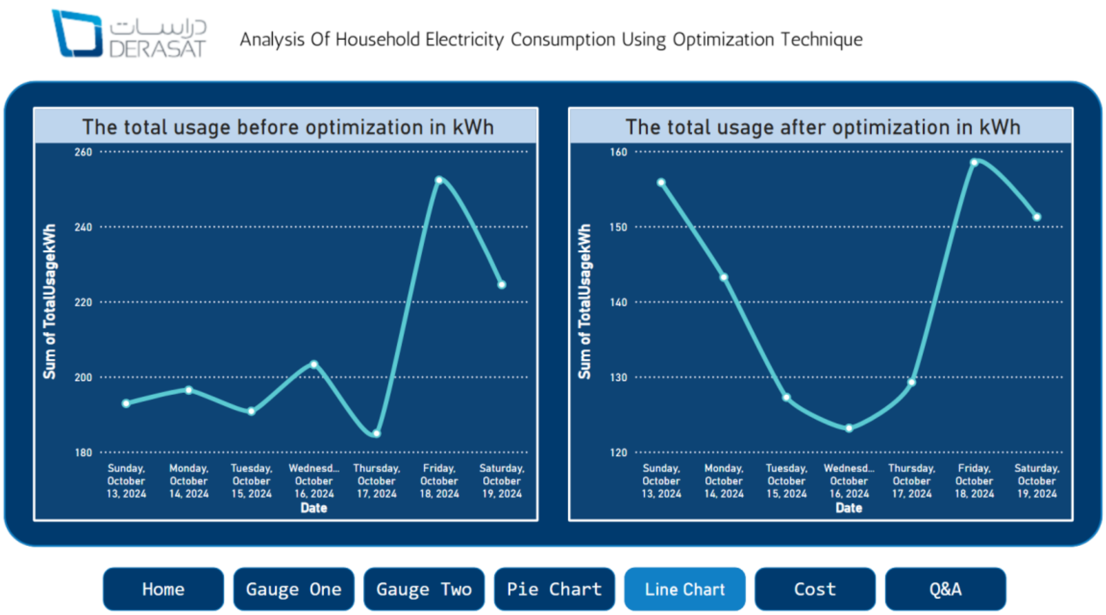


Figure 119: The dashboard "Line Chart" page

- **Cost page:** The page calculates the total cost of energy usage according to the Electricity and Water Authority | Bahrain (2019), depending on the following three categories: -

1. Domestic for Bahrainis with One Account (Subsidy)

Table 8: Cost for Domestic for Bahrainis with One Account (Subsidy)

Consumption Range (kWh)	Monthly Fixed Charge (BHD)	Electricity Usage Charge (BHD/kWh)
First 3,000 kWh		0.003 BHD
3,001 – 5,000 kWh	1.000 BHD	0.009 BHD
Above 5,000 kWh		0.016 BHD

2. Domestic for Bahrainis with More Than One Account, Non-Bahrainis, and Others

Table 9: Cost for Domestic for Bahrainis with More Than One Account, Non-Bahrainis, and Others

Consumption Range (kWh)	Monthly Fixed Charge (BHD)	Electricity Usage Charge (BHD/kWh)
All Consumption	1.000 BHD	0.029 BHD

3. Non-Domestic / Commercial

Table 10: Cost for Non-Domestic / Commercial

Consumption Range (kWh)	Monthly Fixed Charge (BHD)	Electricity Usage Charge (BHD/kWh)
First 5,000 kWh	1.000 BHD	0.016 BHD
Above 5,000 kWh		0.029 BHD

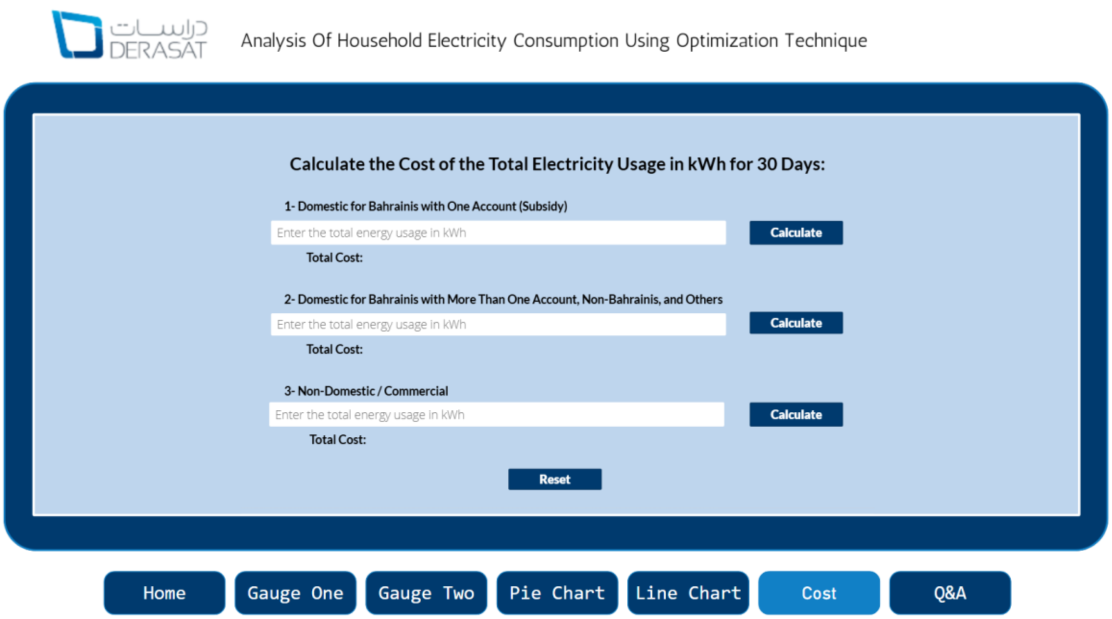


Figure 120: The dashboard "Cost" page

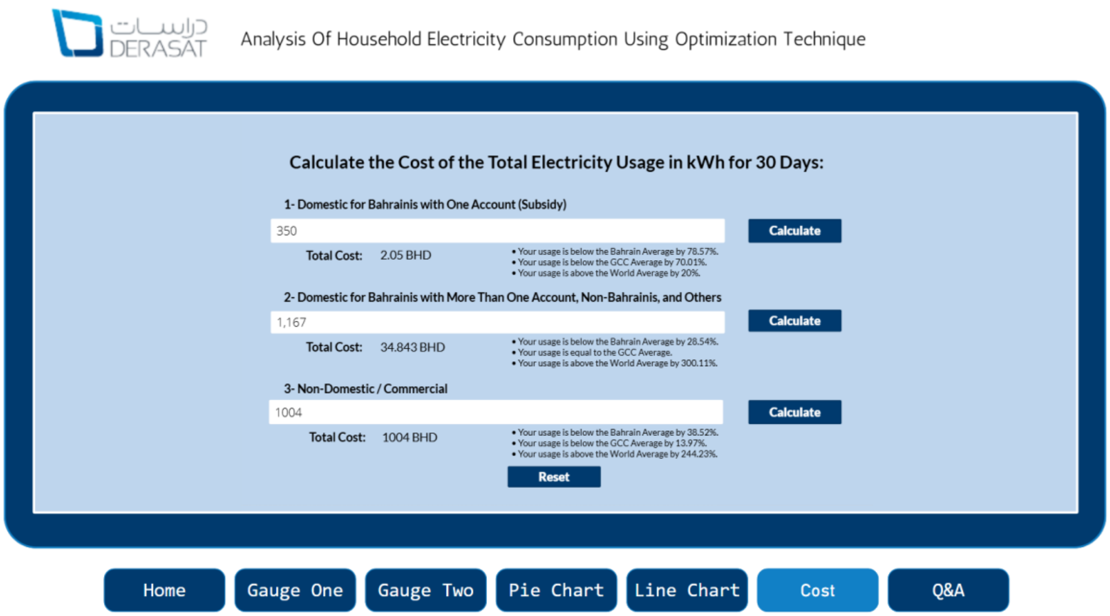


Figure 121: The dashboard "Cost" page, after calculating the total usage cost for each category

The state of the cost page buttons:-

1. Up state: -

Figure 122: Navigation button up state

2. Hover state: -

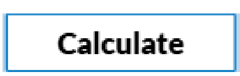


Figure 123: Navigation button hover state

3. Active state: -

Figure 124: Navigation button active state

- **Q&A page:** The page shows an input field where the client can write a question related to the datasets and get instant results.

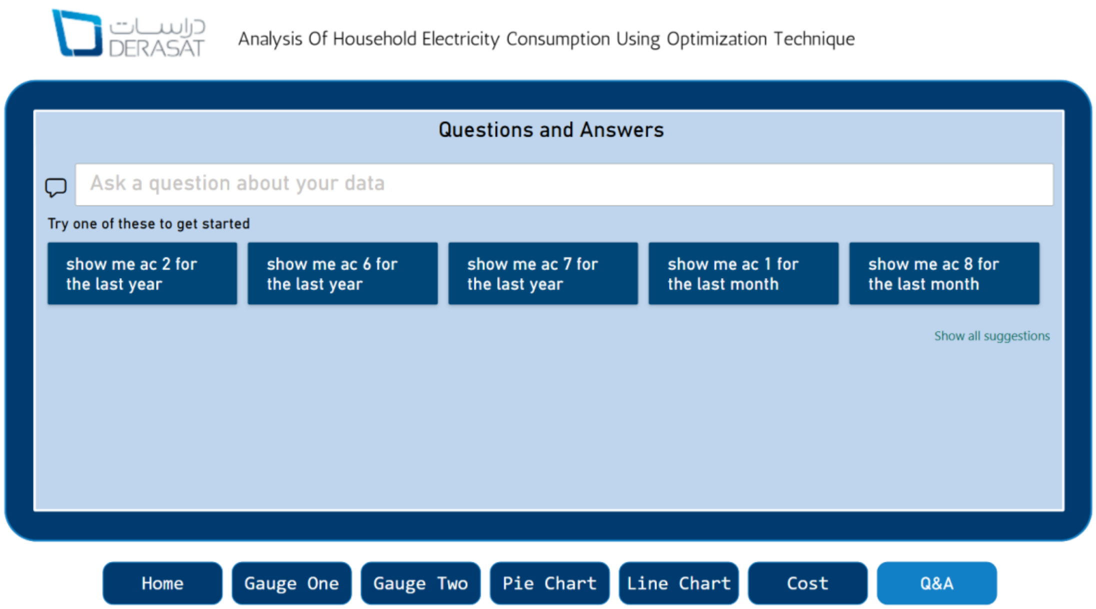


Figure 125: The dashboard "Q&A" page

After clicking on the *"Show all suggestions", more buttons will be displayed: -*

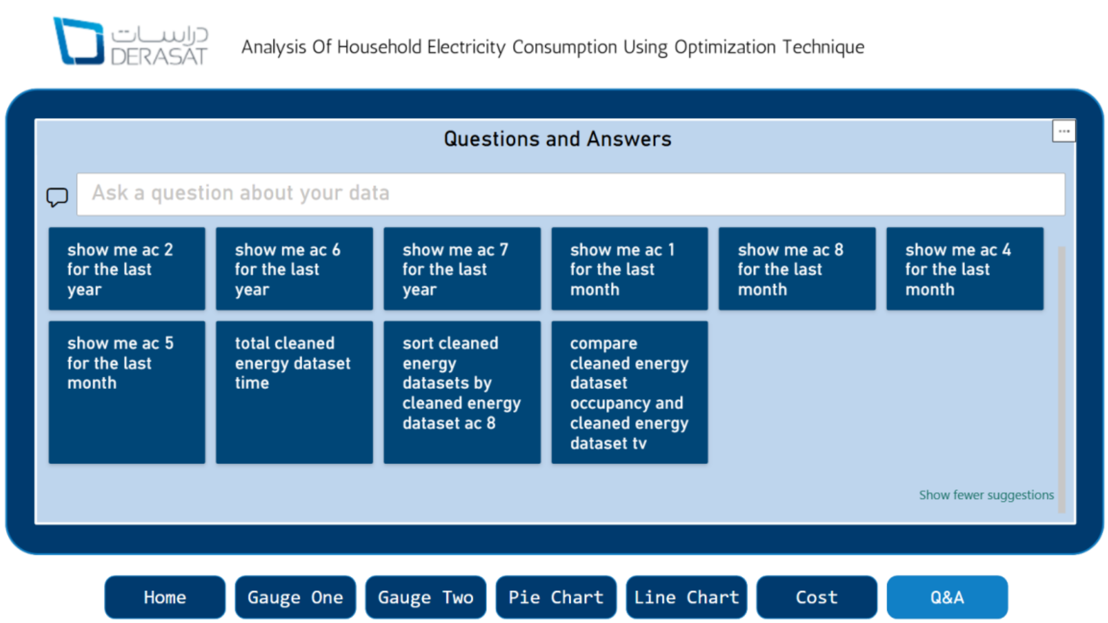


Figure 126: The dashboard "Q&A" page after clicking on the "Show all suggestions"

A red solid underline indicates an error, a blue solid underline indicates an accepted value, and a yellow dotted underline indicates a warning: -

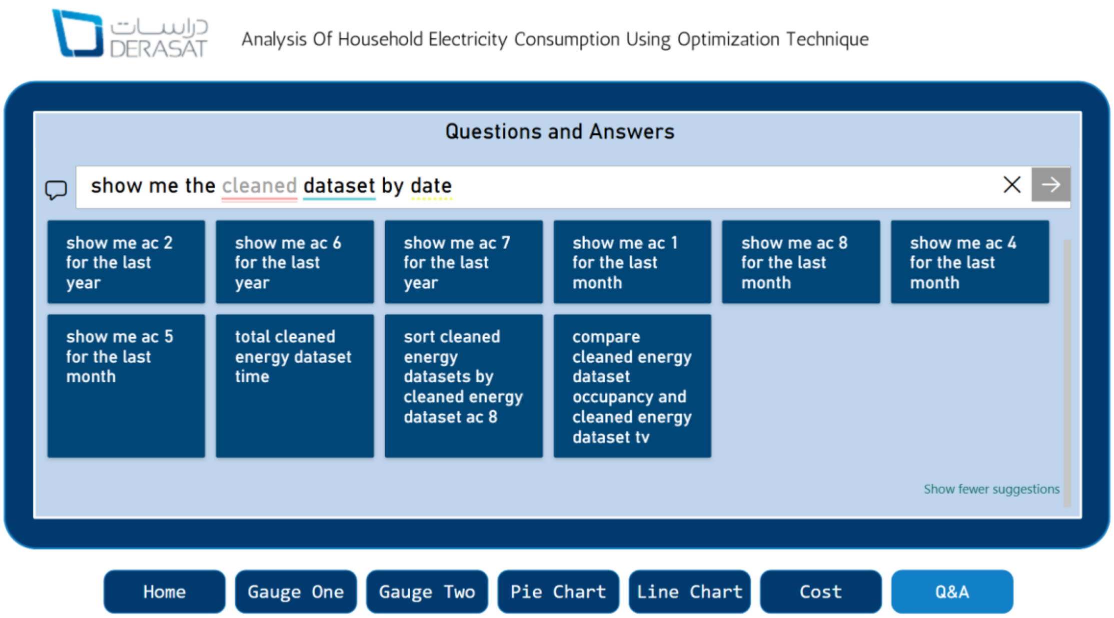


Figure 127: The dashboard "Q&A" underline colors

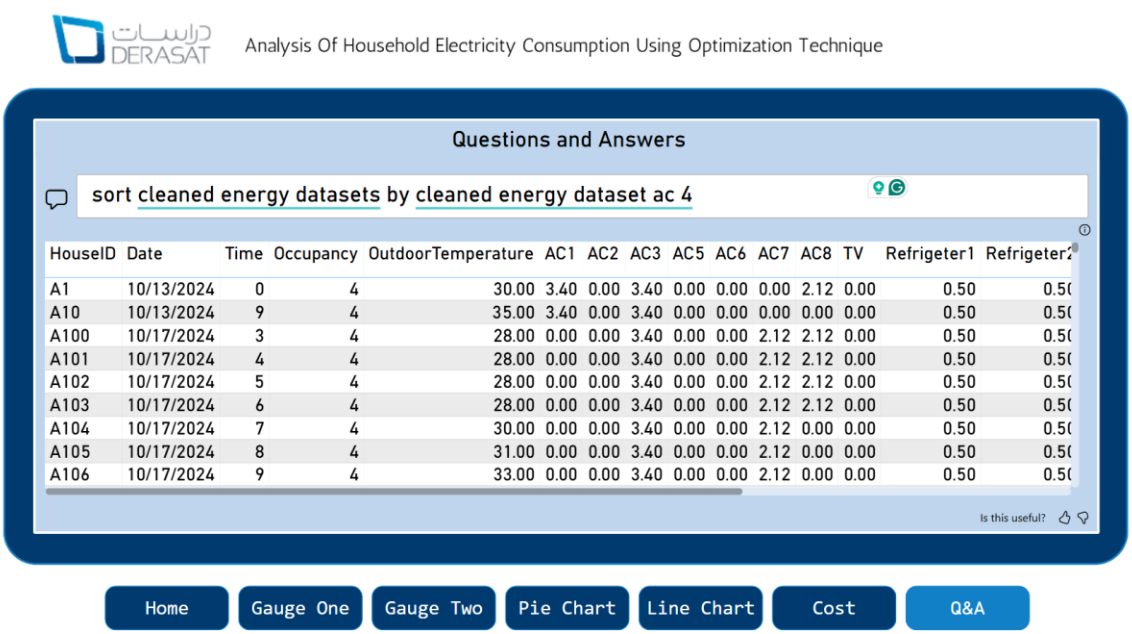


Figure 128: The dashboard "Q&A", example one

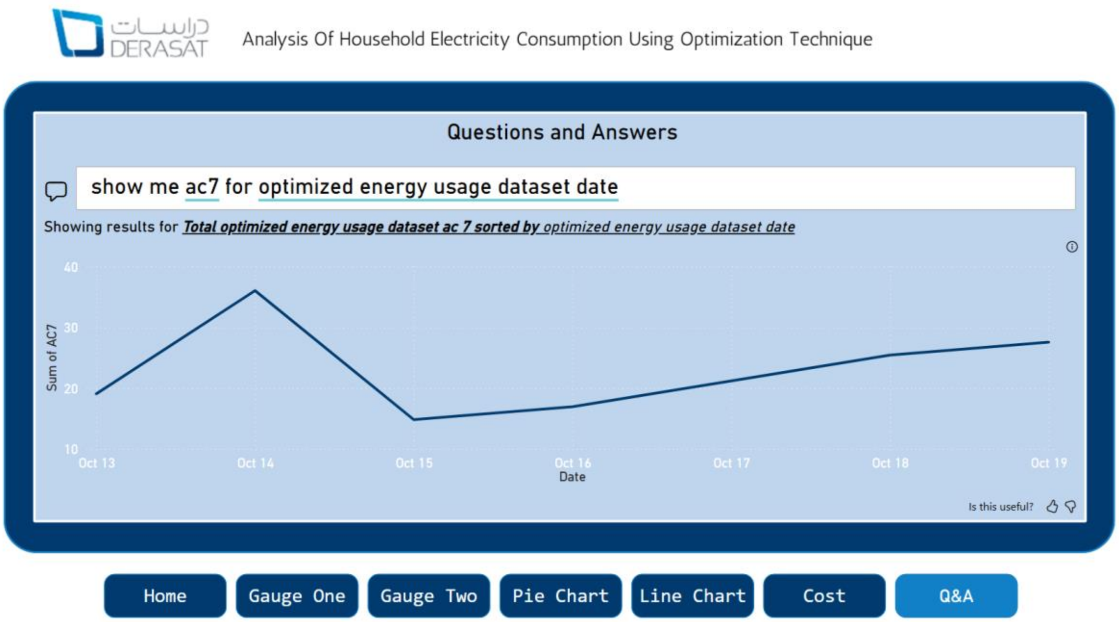


Figure 129: The dashboard "Q&A", example two

Appendix IV: Detailed Implementation and Additional Insights

All configurations and extra information about household consumption over the past months are included in this final part of the thesis.

System Setup

To install Python and Jupyter Notebook: -

1. Start with downloading Python from the Python official website (Python, 2024). Next, open the terminal to verify the installation version. After that, install Jupyter Notebook.

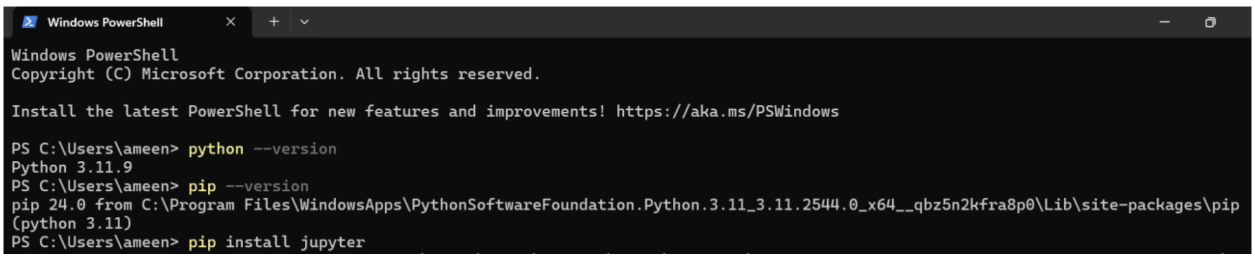


Figure 130: Terminal window for Jupyter Notebook installation

2. Create an empty folder, and name it “jupyter”.

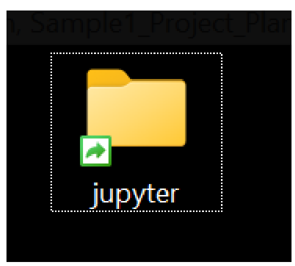


Figure 131: Creation of an empty folder

3. In the “jupyter” folder location, copy the path.

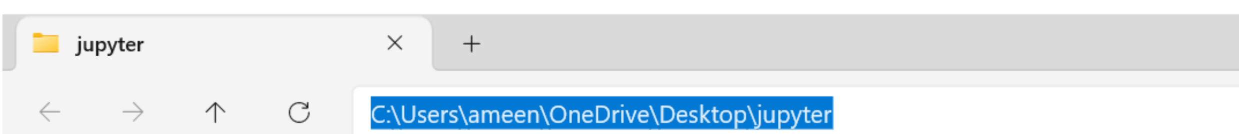


Figure 132: Folder location path

4. In the Terminal, paste the path of the folder location, then write “jupyter notebook”.

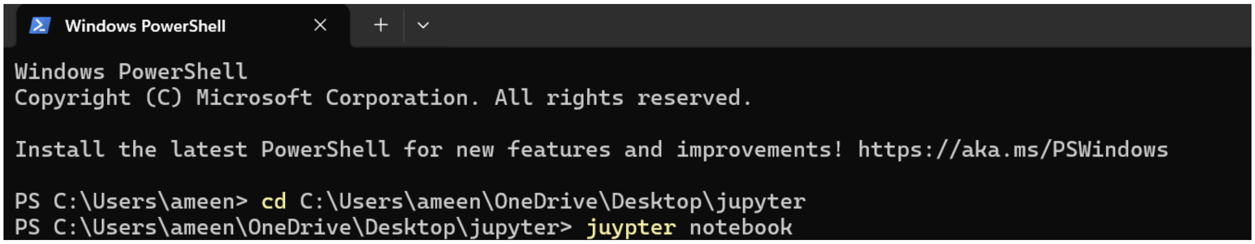


Figure 133: Opening Jupyter Notebook in Terminal

5. The following will pop up.

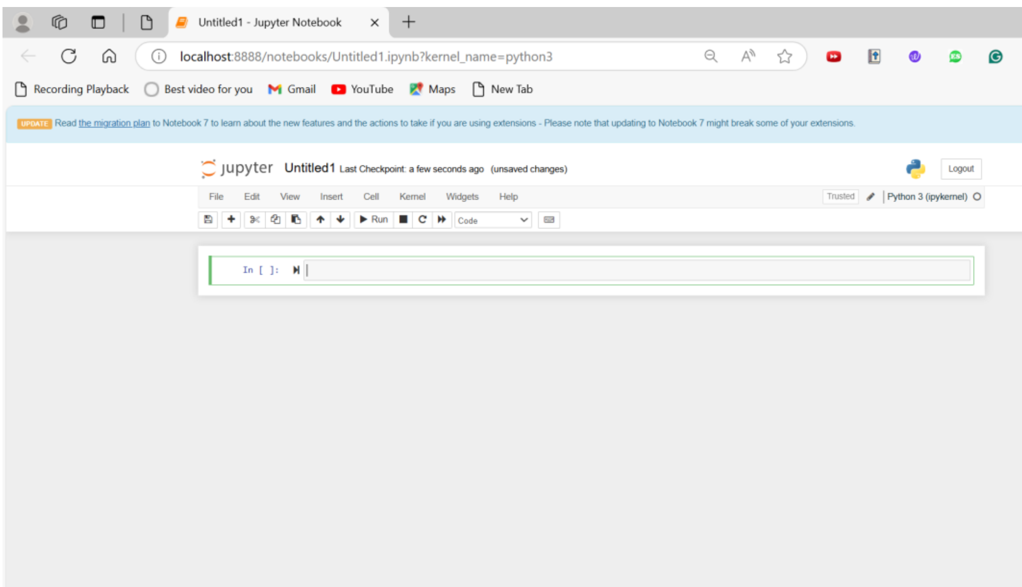


Figure 134: Jupyter Notebook

A survey for the household occupants after showing them the optimized schedule

A survey is being conducted for the household members using Google Forms, It involves four main occupants after viewing the enhanced schedule to evaluate the effectiveness of the solution.

Optimized Energy Usage Feedback Survey

You are being invited to participate in a survey entitled "Optimized Energy Usage Feedback Survey". This survey is done by Ameena Almohanna from Bahrain Polytechnic.

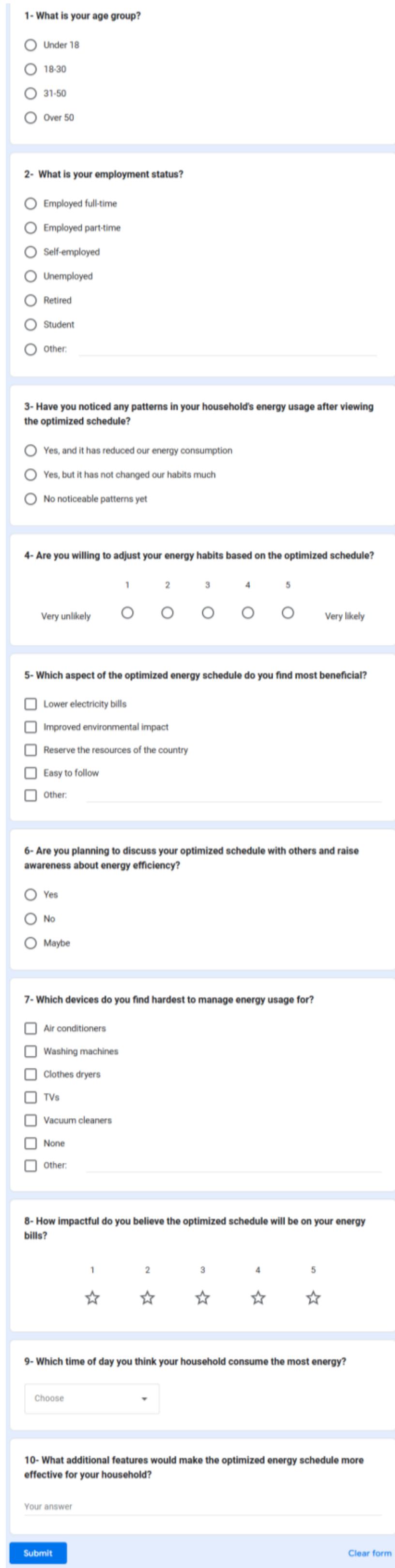
This survey aims to gather feedback on how the optimized energy usage schedule impacts your daily habits and energy consumption preferences, and will take you approximately 10 minutes to complete. Your participation in this survey is entirely 100% voluntary and you can withdraw at any time.

We believe there are no known risks associated with this survey; however, as with any online related activity, the risk of a breach is always possible. To the best of our ability your answers in this survey will remain confidential and anonymous. All collated data after three years of running the survey will be destroyed.

Sincerely,

Ameena Almohanna

Figure 135: Survey description for the household occupants



1- What is your age group?

- Under 18
- 18-30
- 31-50
- Over 50

2- What is your employment status?

- Employed full-time
- Employed part-time
- Self-employed
- Unemployed
- Retired
- Student
- Other:

3- Have you noticed any patterns in your household's energy usage after viewing the optimized schedule?

- Yes, and it has reduced our energy consumption
- Yes, but it has not changed our habits much
- No noticeable patterns yet

4- Are you willing to adjust your energy habits based on the optimized schedule?

Very unlikely 1 2 3 4 5 Very likely

5- Which aspect of the optimized energy schedule do you find most beneficial?

- Lower electricity bills
- Improved environmental impact
- Reserve the resources of the country
- Easy to follow
- Other:

6- Are you planning to discuss your optimized schedule with others and raise awareness about energy efficiency?

- Yes
- No
- Maybe

7- Which devices do you find hardest to manage energy usage for?

- Air conditioners
- Washing machines
- Clothes dryers
- TVs
- Vacuum cleaners
- None
- Other:

8- How impactful do you believe the optimized schedule will be on your energy bills?

1 2 3 4 5

9- Which time of day you think your household consume the most energy?

Choose

10- What additional features would make the optimized energy schedule more effective for your household?

Your answer

Submit

Clear form

Figure 136: Ten survey questions sent to the household occupants

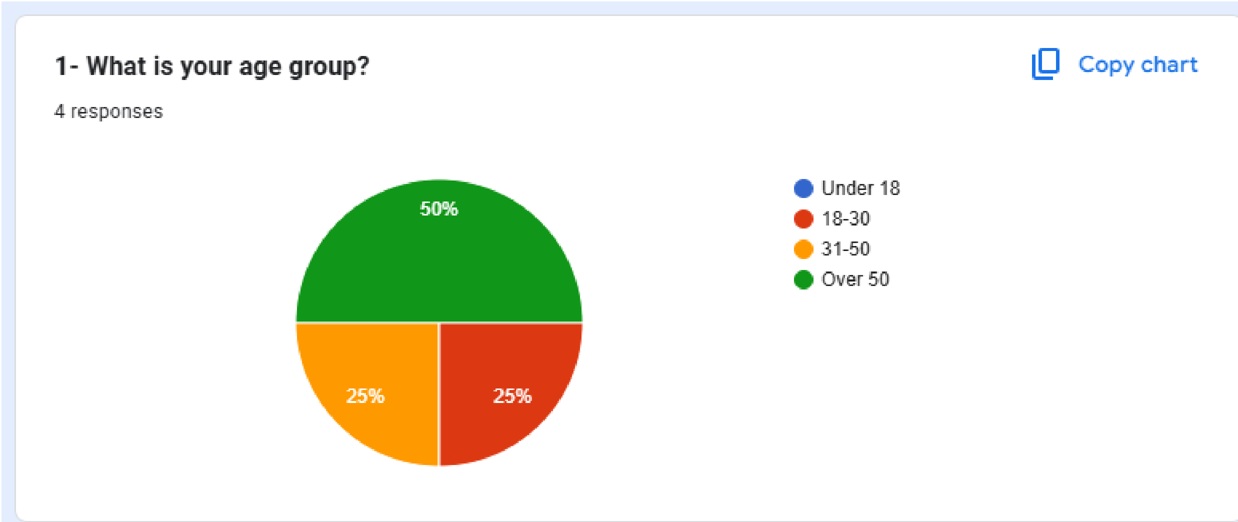


Figure 137: Answers for Question One

The above pie chart shows an age distribution for four respondents. Half of the respondents are over 50, one is between 18-30, and one is between 31-50. None are under 18 years old.

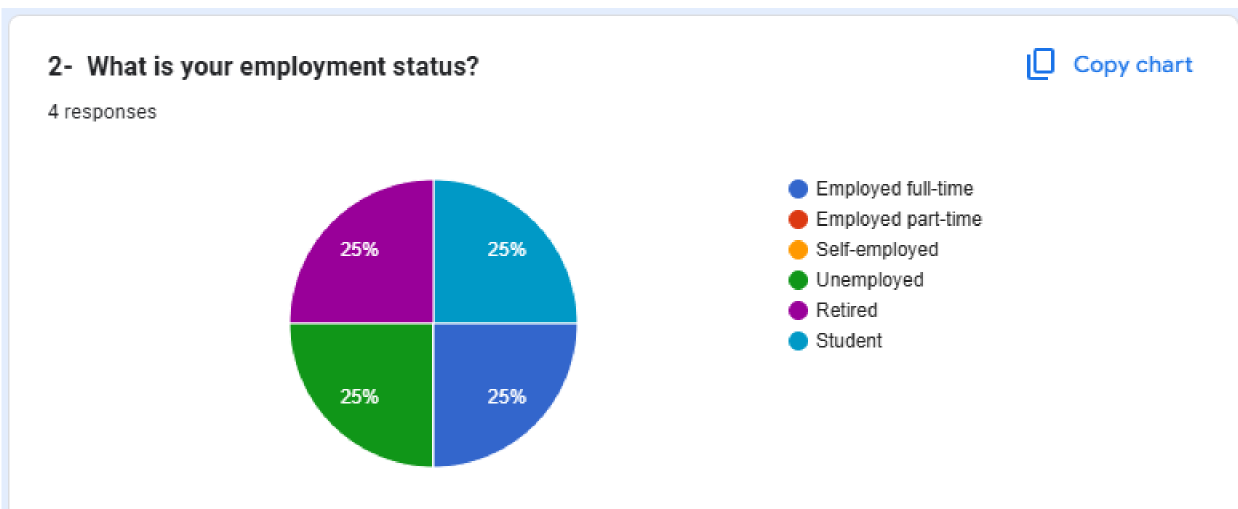


Figure 138: Answers for Question Two

The chart shows employment status, each category receiving an equal share of 25%. The categories are Employed full-time, Unemployed, Retired, and Student.

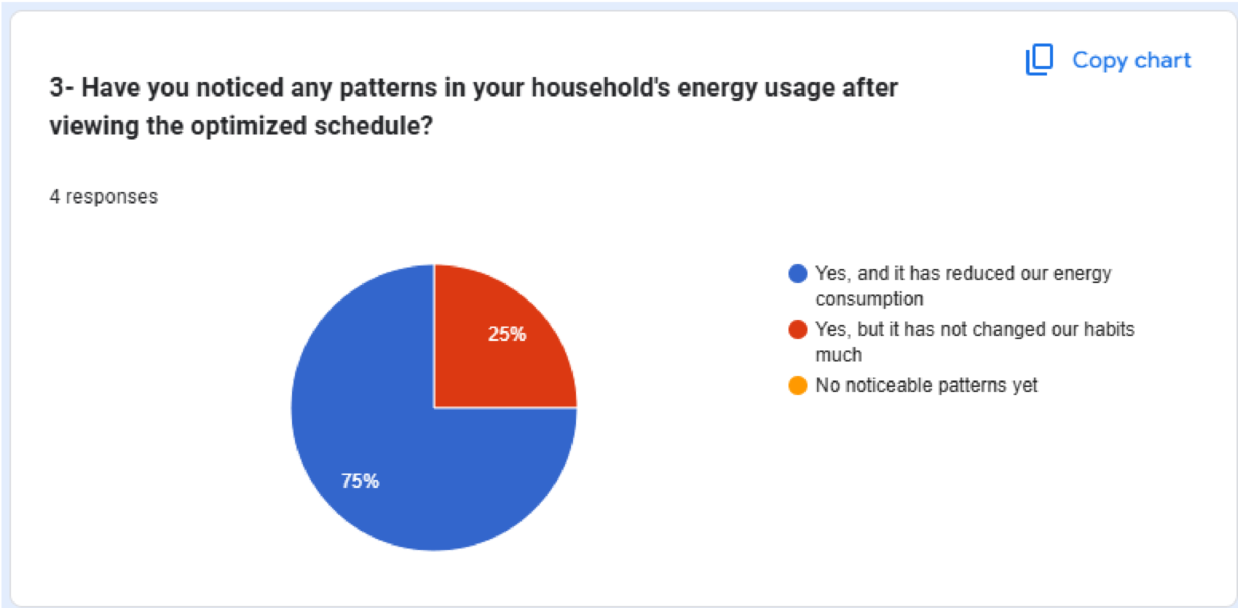


Figure 139: Answers for Question Three

According to the pie chart, seventy-five per cent of respondents noticed patterns in their energy usage after viewing the optimized schedule, while 25 per cent noticed patterns but reported minimal changes.

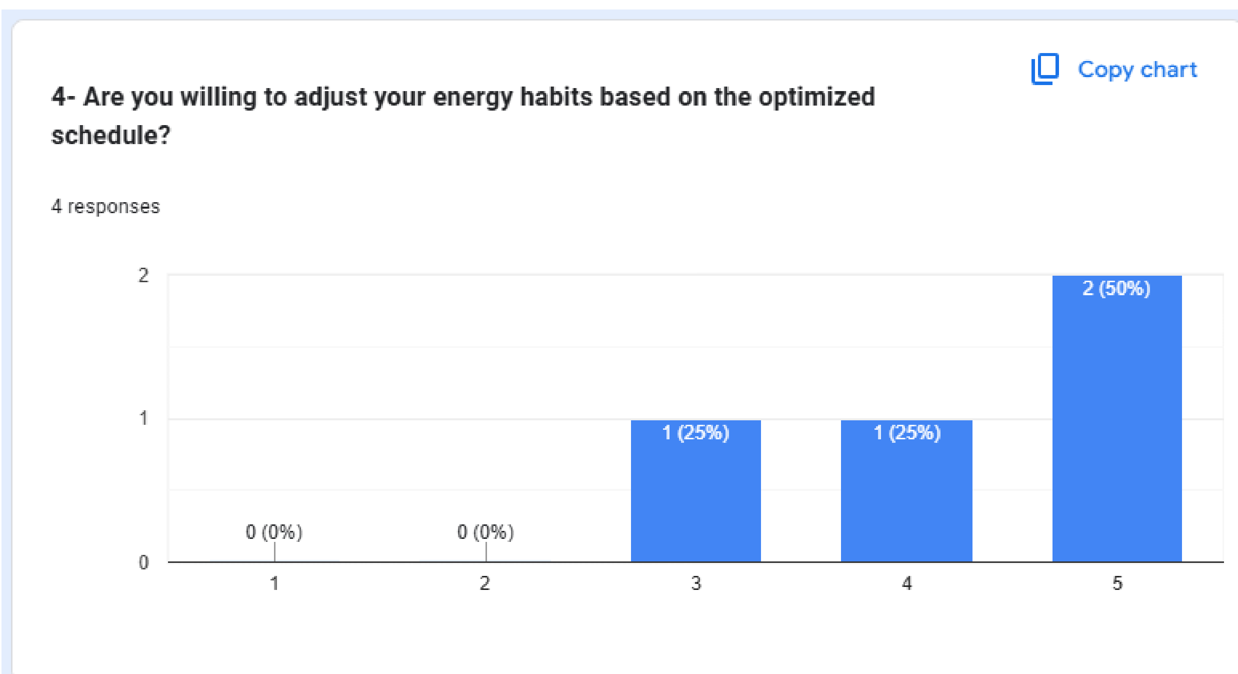


Figure 140: Answers for Question Four

According to the bar chart, respondents are willing to adjust their energy habits based on the optimized schedule. Half of the respondents rated their willingness as 5 (very willing), while 25% rated it as 3 and 4. There were no respondents who rated their willingness below 3.

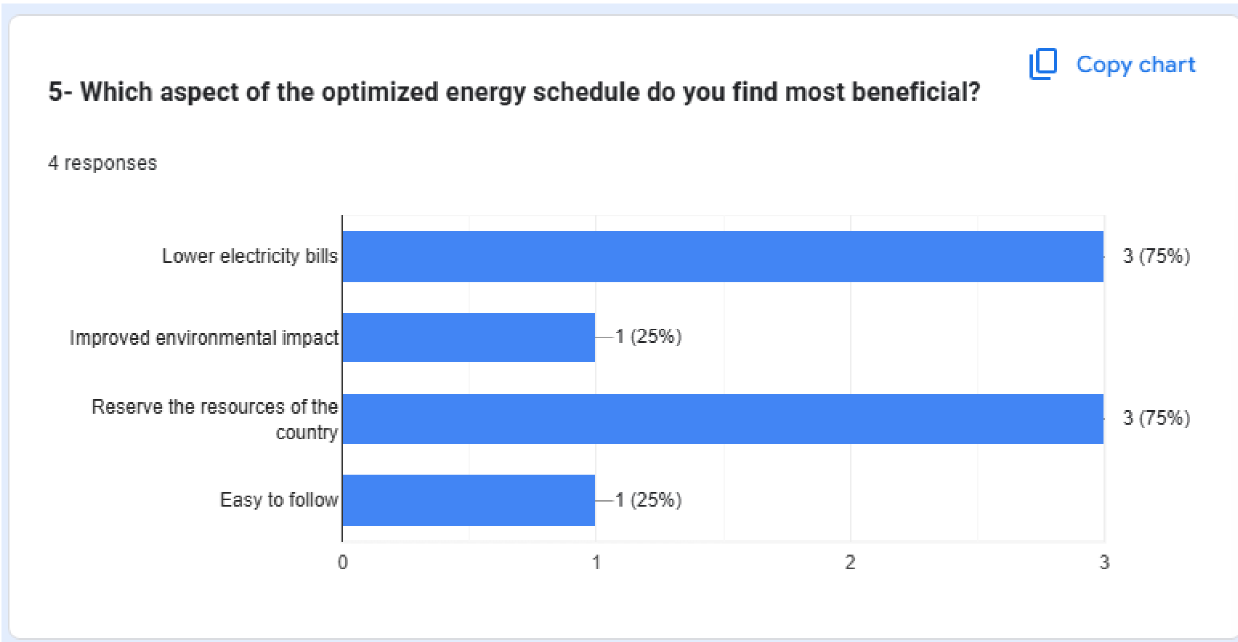


Figure 141: Answers for Question Five

The chart shows what occupants believe they will gain from an optimized energy schedule. Most respondents believe lower electricity bills and reserving country resources are the most beneficial.

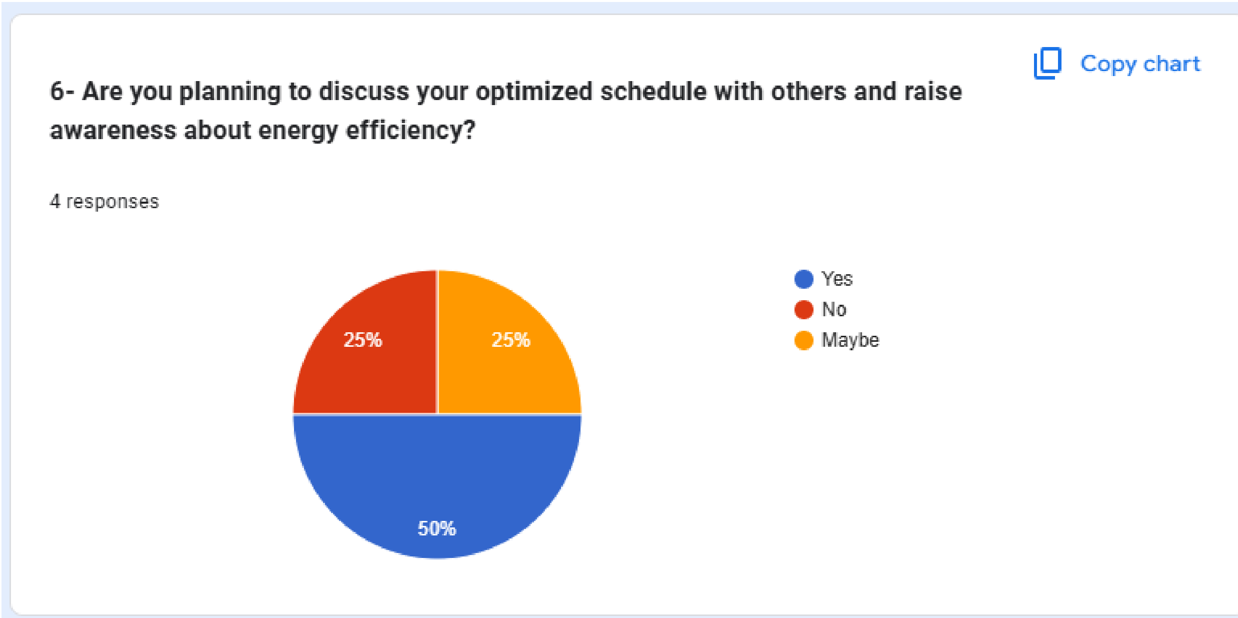


Figure 142: Answers for Question Six

According to the chart, 50% of respondents plan to share their optimized schedule with others to raise awareness about energy efficiency. Meanwhile, 25% are unsure, and another 25% are not planning to share.

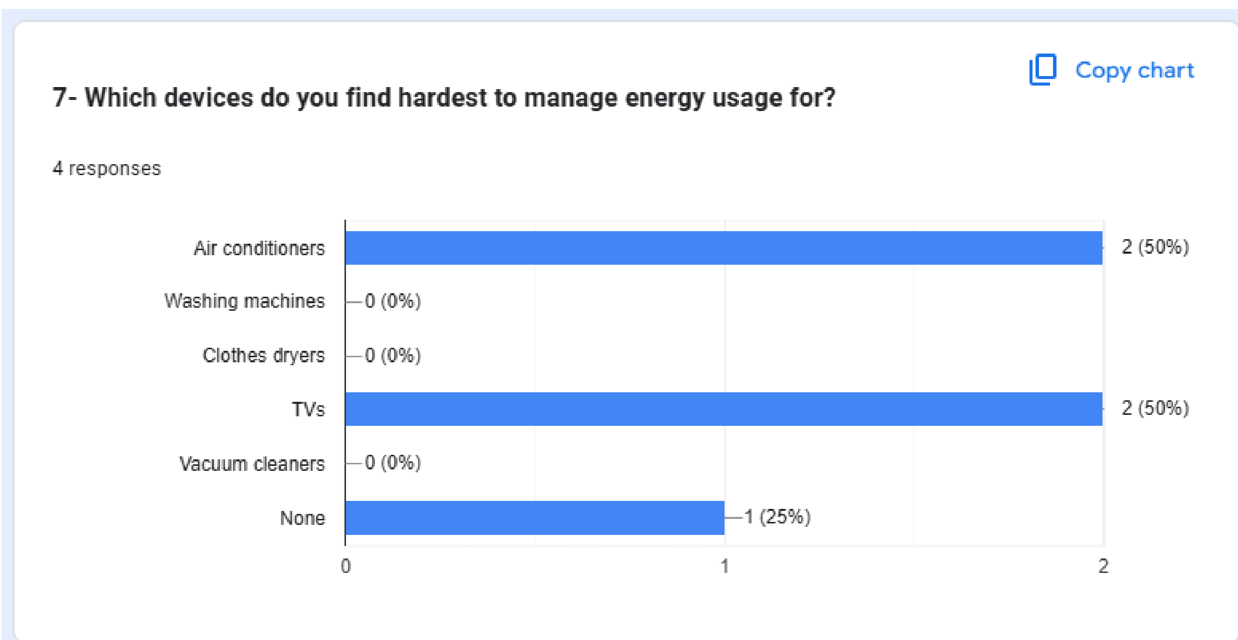


Figure 143: Answers for Question Seven

Based on the chart, 50% of respondents find air conditioners and TVs the most challenging devices to manage. There was one respondent (25%) who indicated that managing energy usage was not a problem.

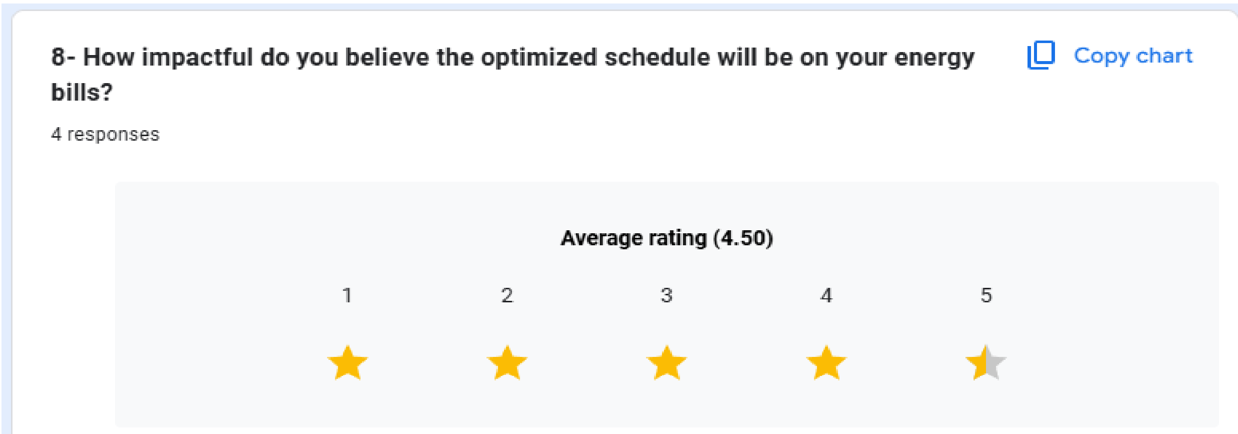


Figure 144: Answers for Question Eight

According to the chart, respondents rated the optimized schedule highly, with a rating of 4.5 on average.

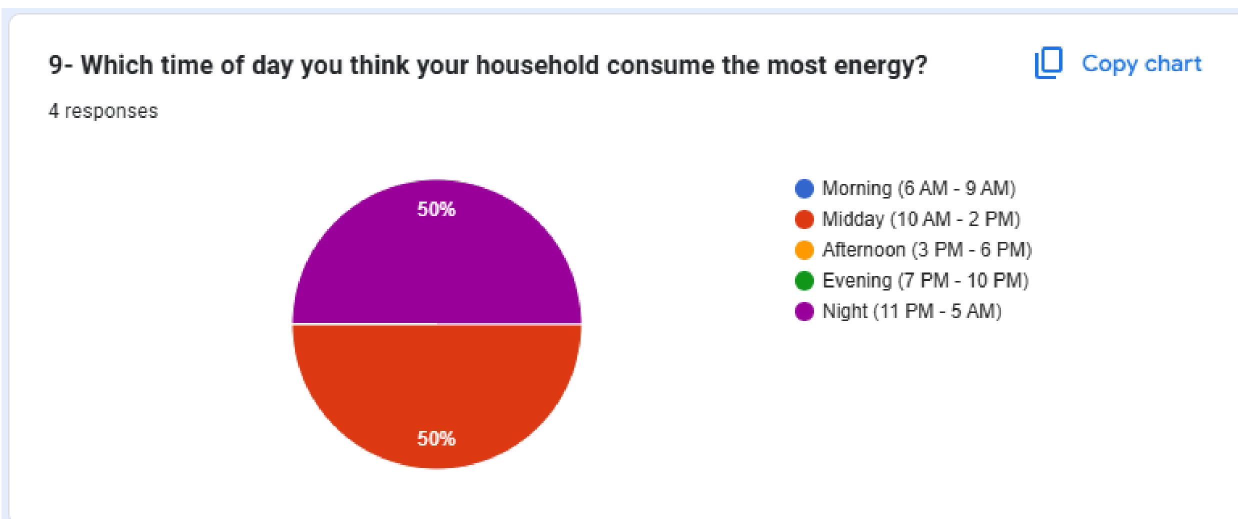


Figure 145: Answers for Question Nine

There are 50% of respondents who believe their households consume the most energy at Midday (10 a.m. - 2 p.m.) and Night (11 p.m. - 5 a.m.)

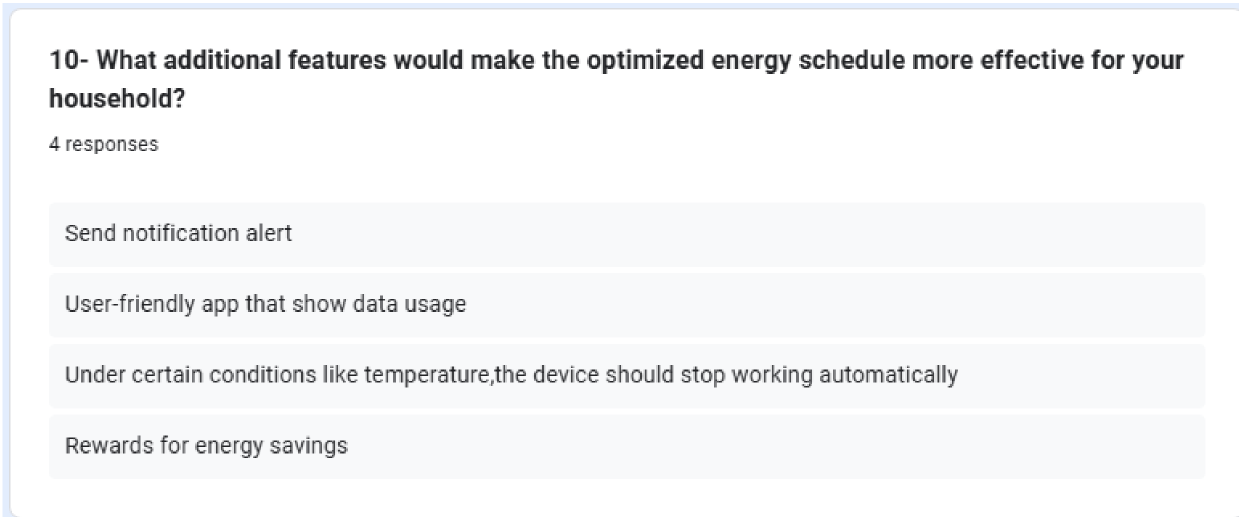


Figure 146: Answers for Question Ten

The responses suggested additional features to make the optimized energy schedule more effective, such as notification alerts, a user-friendly app to display data usage, automatic device shutdown under certain conditions like high temperatures, and rewards for energy savings.

Electricity and Water Authority (EWA) Graph for Additional Insights

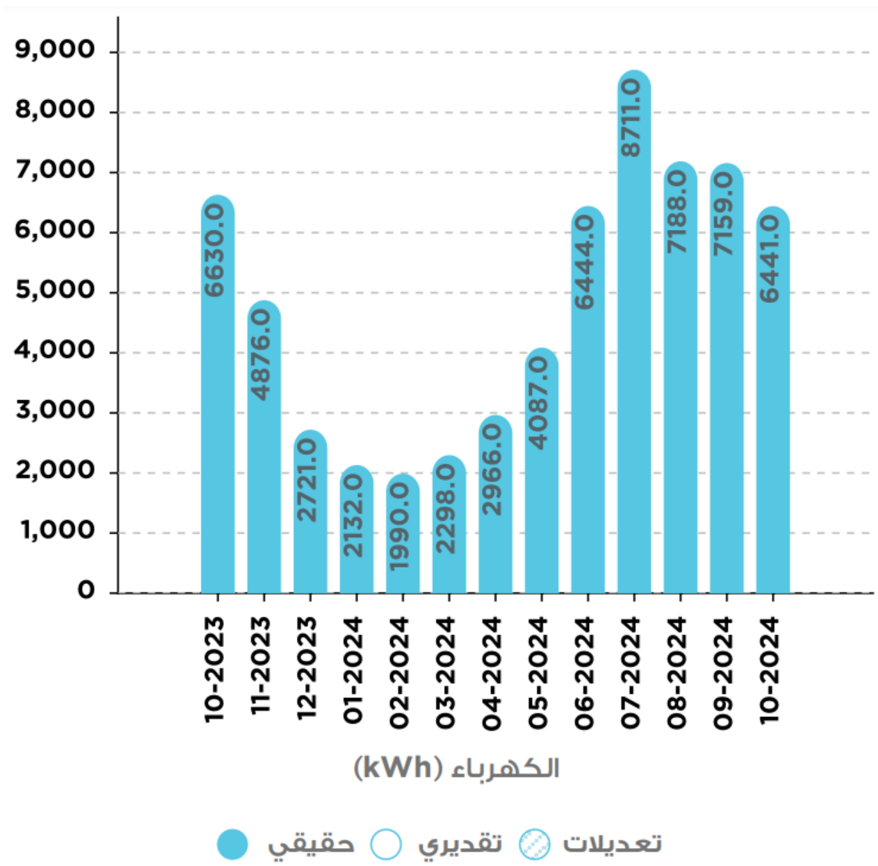


Figure 147: Electricity and Water Authority (EWA) graph for monthly energy usage (kWh) of the household from October 2023 to October 2024

The figure shows the effect of seasonal changes in Bahrain on the consumption of energy. Energy consumption significantly increases during the summer months (July to September) and decreases significantly during the winter months (January to March). It reached its maximum in July 2024 when 8711 kWh were consumed. While the lowest consumption was recorded in February 2024 at 1990 kWh.

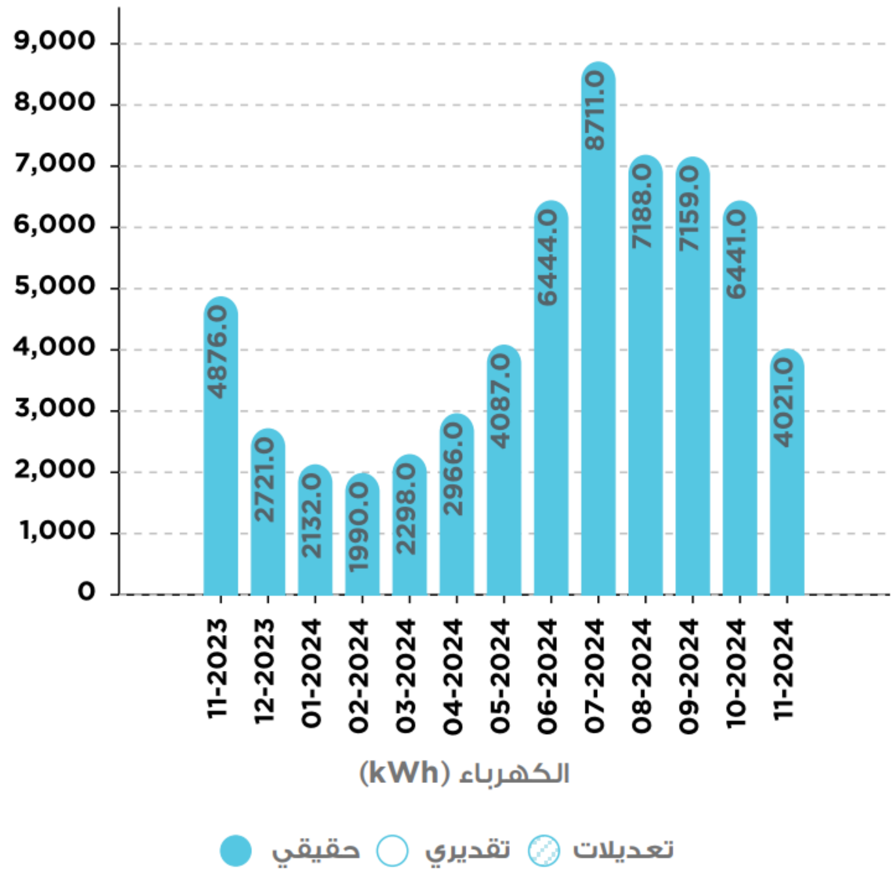


Figure 148: Electricity and Water Authority (EWA) graph for monthly energy usage (kWh) of the household from November 2023 to November 2024

In this chart, the total usage for November 2024 (4876.0 kWh) is shown to have decreased significantly compared to November 2023 (4021.0 kWh) after implementing the optimization method in October 2024 and showing the household occupants the optimized schedule based on their usage pattern. The difference of 855 kWh indicates a 17.53 percent improvement in energy efficiency. This means a change in the resident's behavior pattern after showing the project outcome.

The average household electricity consumption compared to other countries

Table 11: The average household electricity consumption and costs

Month	Usage (kWh)	Cost (BHD)
January	2132.0	7.396
February	1990.0	6.97
March	2298.0	7.894
April	2966.0	9.898
May	4087.0	19.783
June	6444.0	51.104
July	8711.0	87.376
August	7188.0	63.008
September	7159.0	62.544
October	6441.0	51.056
November	4021.0	19.189
December	2721.0	9.163
Total Annual Consumption	56158.0	395.381
Average Monthly Consumption (Total Annual Consumption/12)	4679.83	32.95
Average Daily Consumption (Total Annual Consumption/365)	153.86	1.08
Peak Month	8711.0	87.376
Lowest Month	1990.0	6.97

Table 12: Comparison of the project household electricity consumption in kWh with Bahrain, Gulf Cooperation Council (GCC), and World averages

Factor	Project Household (2023–2024)	Bahrain Average (2021)	GCC Average (2021)	World Average (2021)
Annual Consumption (kWh)	56,158.0	19,597	14,000	3,500

Monthly Average (kWh)	4,679.83	1,633	1,167	291.67
Peak Month Consumption (kWh)	8,711.0 (July)	4,000 (July)	4,250 (Summer)	750 (Summer)
Lowest Month Consumption (kWh)	1,990.0 (February)	1,200	700 (Winter)	250 (Winter)
Electricity Subsidy	Subsidized in Bahrain	Yes	Yes (differs by country)	No (May include tax)

This project's household electricity consumption of 56,158 kWh per year is higher than the average for Bahrain, the GCC, and the world. During the month of July, the peak usage of 8,711 kWh exceeded Bahrain's summer peak of 4,000 kWh, also exceeding global summer averages. With 1,990 kWh consumed in February, the lowest month, consumption is higher than Bahrain and GCC winter averages (Ahmed, Mawahib Eltayeb Ahmed, Ahmed, & Asia Adlan Mohamed, 2024). Therefore, Bahrain houses that consume more electricity than average are considered a useful case study for identifying their pattern and improving their usage.

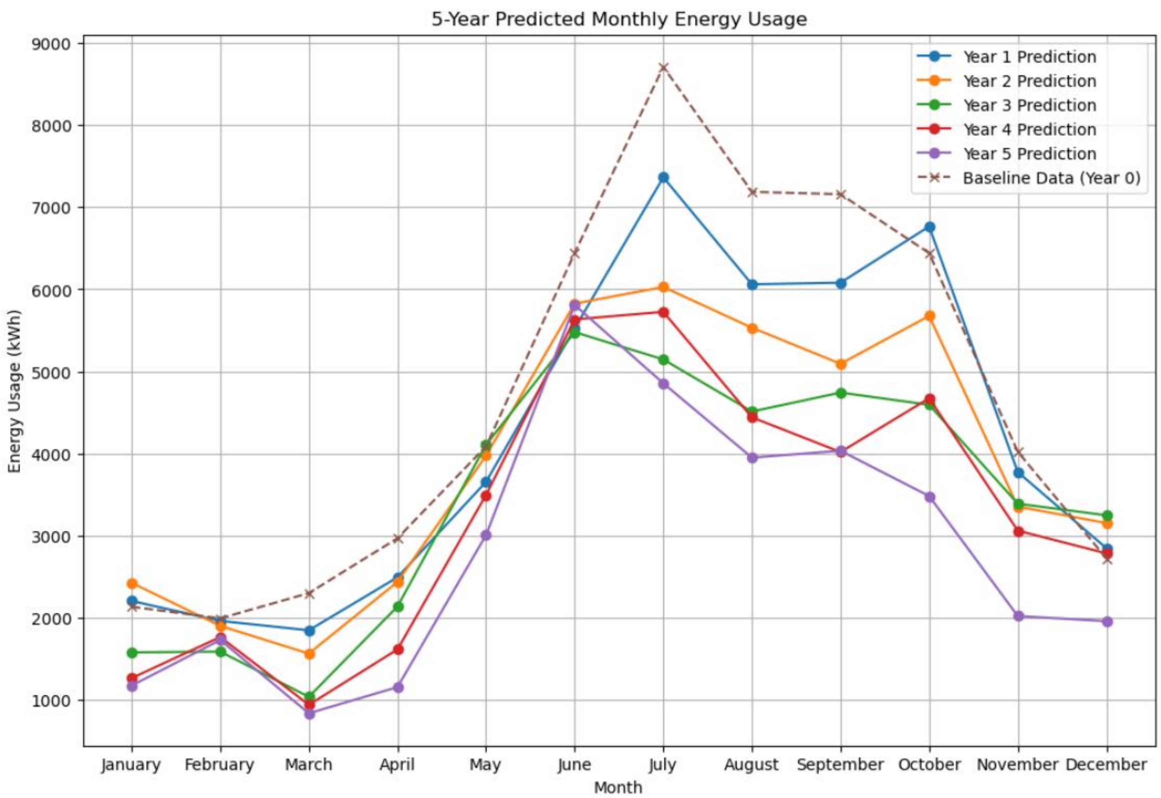


Figure 149: A line chart using Genetic algorithm to optimize energy based on the linear programming optimization schedule and seasonal trends over five years

This chart shows the predicted monthly energy usage using a genetic algorithm over five years. Based on the EWA one-year consumption taken for the sample house in this project as a baseline of 56158.00 kWh, the algorithm gradually reduced consumption to 33992.04 kWh by Year 5. The reduction was based on climate change in Bahrain, and the optimization of scheduled energy usage. While summer months (June to August) show gradual optimization, winter months maintain consistently lower consumption. As a result, the algorithm shows the possibility of reducing energy consumption over time while maintaining necessary usage.

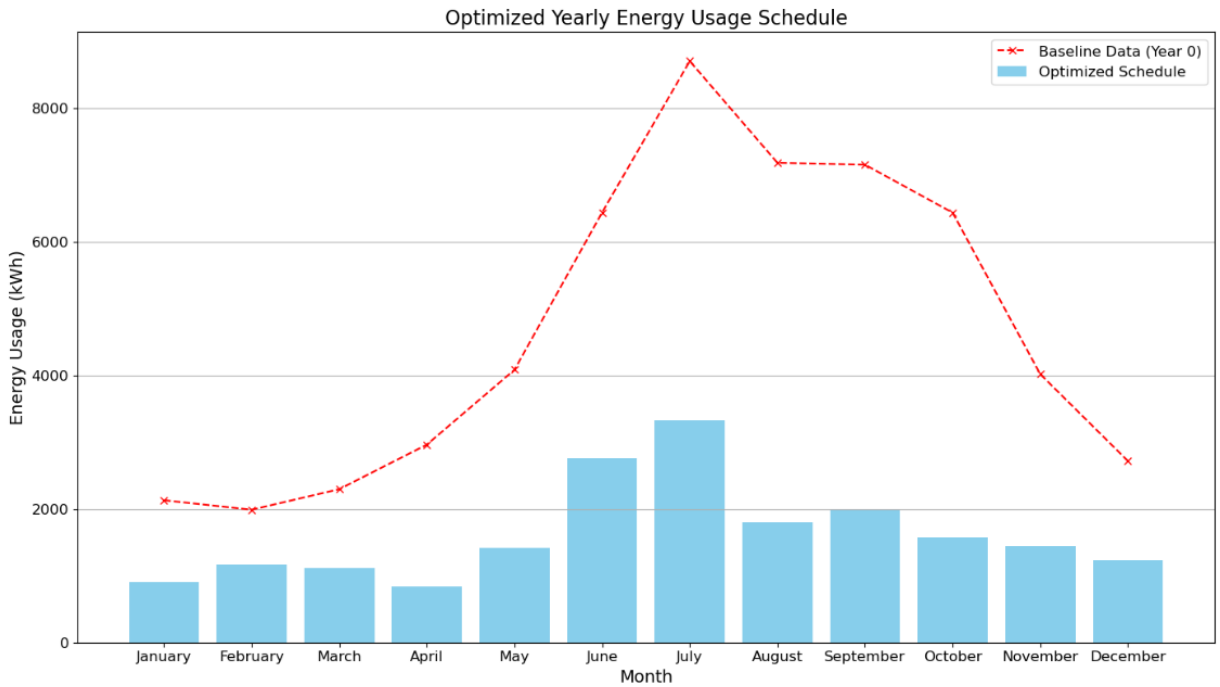


Figure 150: An energy distribution chart using genetic optimization, matching Bahrain's average annual consumption while maintaining a balanced monthly distribution.

Based on Bahrain's average annual energy usage of 19,597 kWh, this chart optimizes the house's energy usage by applying the genetic method. As a result, energy consumption is close to the baseline with a total of 19,597.29 kWh per year. Thus, the chart shows the possibility of reducing peaks, while keeping a balanced usage pattern.

