Problem Definition :

The problem is to measure energy consumption in a residential or commercial setting to improve energy efficiency, reduce costs, and make informed decisions about energy usage. This involves collecting energy consumption data, preprocessing and cleaning the data, extracting relevant features, developing a predictive model for future consumption, visualizing the data and model outputs, and automating the entire process for real-time monitoring and decision-making.

Design Thinking :

Data Source:

• Identify the data sources: Energy consumption data can come from various sources such as smart meters, IoT sensors, utility bills, or historical records. • Collect and store data: Set up data collection systems to gather real-time or historical energy consumption data. • Ensure data quality: Check for missing values, outliers, and inconsistencies in the data.

Data Preprocessing:

• Data cleaning: Handle missing values, outliers, and duplicate entries by imputing, removing, or transforming data as necessary. • Data normalization: Normalize the data to ensure that all features have the same scale. • Time series decomposition: If working with time series data, decompose it into trend, seasonality, and residual components. • Data aggregation: Aggregate data if necessary (e.g., hourly to daily) to reduce noise and improve model performance.

Feature Extraction:

• Define relevant features: Identify features that can impact energy consumption, such as weather data, occupancy patterns, building characteristics, and historical usage. • Feature engineering: Create new features or transform existing ones to enhance the predictive power of the model.

Model Development:

• Choose an appropriate modeling technique: Select a machine learning or statistical model suitable for energy consumption prediction, such as linear regression, decision trees, neural networks, or time series forecasting methods. • Split the dataset: Divide the data into training,

validation, and test sets for model evaluation. • Train the model: Use the training data to train the chosen model, tune hyperparameters, and optimize performance. • Evaluate the model: Assess the model's performance using appropriate evaluation metrics (e.g., Mean Absolute Error, Root Mean Squared Error, R-squared).

Visualization:

• Data exploration: Visualize the raw data to understand patterns and trends in energy consumption. • Model performance: Create visualizations to showcase how well the model predicts energy consumption compared to actual values. • Real-time monitoring: Develop interactive dashboards or visualizations to monitor real-time energy consumption and alerts.

Source code with explanation

Import Necessary Libraries:

Start by importing the necessary Python libraries. You'll need libraries like Pandas for data manipulation, Matplotlib for visualization, and any specific libraries for working with energy consumption data.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings

warnings.filterwarnings("ignore", category=UserWarning)

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR
from sklearn.metrics import mean_squared_error, r2_score

RED = "\033[91m"
GREEN = "\033[92m"
YELLOW = "\033[93m"
BLUE = "\033[94m"
RESET = "\033[0m"
```

Load the Dataset:

Start by importing the necessary Python libraries. You'll need libraries like Pandas for data manipulation, Matplotlib for visualization, and any specific libraries for working with energy consumption data.

```
df = pd.read_csv("PJMW_hourly.csv")
```

Data Preprocessing:

Data preprocessing is essential to clean and prepare the dataset for analysis. Depending on the dataset, you may need to handle missing values, remove duplicates, and format columns.This include:

- Handling missing data.
- Converting data types.
- Removing outliers.
- Normalizing or scaling data.

```
# DATA PREPROCESSING
# --- Convert the date column to a datetime object
df["Datetime"] = pd.to_datetime(df["Datetime"])

# DATA CLEANING
print(BLUE + "\nDATA CLEANING" + RESET)
# --- Check for missing values
missing_values = df.isnull().sum()
print(GREEN + "Missing Values : " + RESET)
print(missing_values)
# --- Handle missing values
df.dropna(inplace=True)
# --- Check for duplicate values
duplicate_values = df.duplicated().sum()
print(GREEN + "Duplicate Values : " + RESET)
print(duplicate_values)
# --- Drop duplicate values
df.drop_duplicates(inplace=True)
```

Data Analysis

```
# DATA ANALYSIS
print(BLUE + "\nDATA ANALYSIS" + RESET)
```

```
# --- Summary Statistics
summary_stats = df.describe()
print(GREEN + "Summary Statistics : " + RESET)
print(summary_stats)
```

Model Development:

Choose an appropriate modeling technique: Select a machine learning or statistical model suitable for energy consumption prediction, such as linear regression, decision trees, neural networks, or time series forecasting methods.

Split the dataset: Divide the data into training, validation, and test sets for model evaluation.

Train the model: Use the training data to train the chosen model, tune hyperparameters, and optimize performance.

Evaluate the model: Assess the model's performance using appropriate evaluation metrics (e.g., Mean Absolute Error, Root Mean Squared Error, R-squared).

```
# SUPPORT VECTOR MODELLLING
print(BLUE + "\nMODELLING" + RESET)
# Reduce the dataset size for faster training
df = df.sample(frac=0.2, random_state=42)
# Split the data into features (Datetime) and target (AEP_MW)
X = df[["Datetime"]]
y = df["PJMW_MW"]
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)
```

Preprocess the features (Date and Time)

```
# Preprocess the features (Datetime) to extract the day of the year
X_train["DayOfYear"] = X_train["Datetime"].dt.dayofyear
X_test["DayOfYear"] = X_test["Datetime"].dt.dayofyear

# Convert X_train and X_test to NumPy arrays
X_train = X_train["DayOfYear"].values.reshape(-1, 1)
X_test = X_test["DayOfYear"].values.reshape(-1, 1)
# Standardize the data
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
# Create an SVR (Support Vector Regression) model with a linear kernel
svr = SVR(kernel="linear", C=1.0)
```

Train the SVR model

```
svr.fit(X_train_scaled, y_train)
# Predict on the test set
y_pred = svr.predict(X_test_scaled)
```

Evaluate the model

```
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

Plot the actual vs. predicted values

```
plt.figure(figsize=(10, 6))
plt.scatter(X_test, y_test, color="b", label="Actual")
plt.scatter(X_test, y_pred, color="r", label="Predicted")
plt.xlabel("Day of the Year")
plt.ylabel("Energy Consumption (MW)")
plt.title("SVR Model: Actual vs. Predicted")
plt.legend()
plt.grid()
plt.show()
```

Visualization:

• Data exploration: Visualize the raw data to understand patterns and trends in energy consumption. • Model performance: Create visualizations to showcase how well the model predicts energy consumption compared to actual values. • Real-time monitoring: Develop interactive dashboards or visualizations to monitor real-time energy consumption and alerts.

```
# DATA VISUALIZATION
print(BLUE + "\nDATA VISUALIZATION" + RESET)
```

--- Line plot

```
print(GREEN + "LinePlot : " + RESET)
plt.figure(figsize=(10, 6))
sns.lineplot(data=df, x="Datetime", y="PJMW_MW")
plt.xlabel("Datetime")
plt.ylabel("Energy Consumption (MW)")
```

```
plt.title("Energy Consumption Over Year")
plt.grid()
plt.show()
```

--- Histogram

```
print(GREEN + "Histogram : " + RESET)
plt.figure(figsize=(10, 6))
plt.hist(
    df["PJMW_MW"],
    bins=100,
    histtype="barstacked",
    edgecolor="white",
)
plt.xlabel("PJMW")
plt.ylabel("Frequency")
plt.title("Histogram of MEGAWATT USAGE")
plt.show()
```

Conclusion

In conclusion, the measuring energy consumption visualization project has provided valuable insights into the patterns and trends of energy usage within our target environment. By leveraging advanced monitoring technology and data visualization techniques, we have achieved several significant outcomes.