Project Development Phase
Code layout and readability - Data Analytics

Date	20 May 2023
Team ID	NM2023TMID17415
Project Name	Project on A Reliable Energy Consumption Analysis System for Energy-Efficient Appliances

Folder structure:

```
app.py
averall.py
energy_analysis.py
    -   data
            -data.csv
            -appliance.csv
    -   static
            (images)
    -   templates
            -index.html
            -eca.html
            -login.html
            -analysis.html
```

Homepage (eca.html)

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Energy Consumption Analysis</title>
    <script>
        function mylink1(){
            window.location.href="analysis.html";
        }
        function mylink2(){
            window.location.href="login.html";
        }
    </script>
    <style>
        table{
  border: 1px;
  border-radius: 10px;
  margin-left: auto;
  margin-right: auto;
}
body {
  background-image: url('ecaimg.webp');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}
tr {
 border: 1px;
 border-radius: 10px;
 height: 200px;
}
td {
  border: 1px;
  border-radius: 10px;
  width: 33.3%;
}
input[type="submit"] {
        width: 90%;
        height: 180px;
        background-color: #1E5128;
        color: white;
        padding: 14px 20px;
        margin: 8px 0;
        border: 1px solid;
        border-color: #1E5128;
        border-radius: 10px;
        cursor: pointer;
        font-weight: bold; font-family: 'Gill Sans'; font-size:30px;
      }
      input[type="submit"]:hover {
        background-color: #4E9F3D;
      }
    </style>
</head>
<body >
   <h1 align="center" style="color:#191A19; font-size: 50px;">Energy
Consumption Analysis</h1>
   <div>
    <table align="left">
        <tr>
            <td><input type="submit" value="Overall Data Analysis"
name="Overall" onclick="mylink1()"></td>
            <td align="right"><input type="submit" value="Individual Data
Analysis" name="Individual" onclick="mylink2()"></td>
        </tr>
    </table>
   </div>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Energy-Login</title>
    <script>
        function validateForm(){
            var x = document.forms["loginForm"]["uname"].value;
            var p = document.forms["loginForm"]["pwd"].value;
            if (
          x == "" &&
          p == ""
        ) {
          alert("Enter Username and Password");
          return false;
        } else if (x == "") {
          alert("Username not entered");
          return false;
        } else if (p == "") {
          alert("Password not entered");
          return false;
        }


    </script>
    <style>
        body {
  background-image: url('log.jpeg');
  background-repeat: no-repeat;
  background-attachment: fixed;
  background-size: cover;
}
      input[type="text"],
      select {
        width: 100%;
        padding: 12px 20px;
        margin: 8px 0;
        display: inline-block;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-sizing: border-box;
      }
      input[type="password"],
      select {
        width: 100%;
        padding: 12px 20px;
        margin: 8px 0;
        display: inline-block;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-sizing: border-box;
      }
      input[type="submit"] {
        width: 50%;
        background-color: #05BFDB;
        color: white;
        padding: 14px 20px;
        margin: 8px 0;
        border: none;
        border-radius: 4px;
        cursor: pointer;
      }
      input[type="submit"]:hover {
        background-color: #00FFCA;
      }
      input[type="reset"] {
        width: 50%;
        background-color: #05BFDB;
        color: white;
        padding: 14px 20px;
        margin: 8px 0;
        border: none;
        border-radius: 4px;
        cursor: pointer;
      }
      input[type="reset"]:hover {
        background-color: #00FFCA;
      }
      div {
        border-radius: 5px;

        padding: 20px;
        height: 100%;
      }
      table{
  border: 1px;
  border-radius: 10px;

  width: 40%;
}
td {
  border: 1px;
  border-radius: 10px;
  width: 50%;
```

```
}
    </style>
</head>
<body style="background-color: #FFA3FD;">
  <div>   <h2 align="right"><a href="eca.html" style="color: rgb(234, 223,
223);">EXIT</a></h2>
  </div>
    <form name="loginForm" action="index.html" method="get">
        <div>
            <table align="left">
                <tr>
                    <th align="center" style="color: #31E1F7; font-size:
50px;" colspan="2">Individual User Analysis</th>
                </tr>
                <tr>
                <th align="center" style="color: #19A7CE;"
colspan="2"><h2>Login Page</h2></th>
                </tr>
                <tr>
                    <td align="center" style="font-family: 'Gill Sans';
color: #97DEFF;font-weight: bolder;">USERNAME:</td>
                    <td><input type="text" name="uname"></td>
                </tr>
                <tr>
                    <td align="center" style="font-family: 'Gill Sans';
color: #97DEFF;font-weight: bolder;">PASSWORD:</td>
                    <td><input type="password" name="pwd"></td>
                </tr>
                <tr>
                    <td style="font-weight: bolder;" align="center"
colspan="2"><input type="submit" value="LOGIN" onclick=" return
validateForm()"></td>
                </tr>
                <tr>
                    <td align="center" colspan="2"><input type="reset"
value="RESET"></td>
                </tr>
            </table>
        </div>
    </form>
```

```
</body>
</html>
```

Form.html

```
<html>
<head>
    <title>Energy Consumption Analysis</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f2f2f2;
            margin: 0;
            padding: 20px;
        }

        h1 {
            color: #333333;
        }

        form {
            max-width: 400px;
            margin: 0 auto;
            background-color: #ffffff;
            padding: 20px;
            border: 1px solid #cccccc;
            border-radius: 5px;
        }

        label {
            display: block;
            margin-bottom: 10px;
            color: #333333;
        }

        select,
        input[type="number"] {
            width: 100%;
            padding: 8px;
            font-size: 16px;
            border-radius: 3px;
            border: 1px solid #cccccc;
        }

        input[type="submit"] {
            background-color: #4CAF50;
            color: white;
            border: none;
            padding: 10px 20px;
            font-size: 16px;
            cursor: pointer;
            border-radius: 3px;
        }

        input[type="submit"]:hover {
            background-color: #45a049;
        }
    </style>
</head>
<body>
    <div>   <h3 align="right"><a href="eca.html" style="color:
#000000;">LOGOUT</a></h3> </div>
    <h1 align="center" style="font-family: 'Times New Roman', Times,
serif;">Energy Consumption Analysis</h1>
    <form action="/predict" method="POST">
        <label for="appliance_type">Appliance Type:</label>
        <select name="appliance_type" required>
            <option value="Refrigerator">Refrigerator</option>
            <option value="Air Conditioner">Air Conditioner</option>
            <option value="Washing Machine">Washing Machine</option>
            <!-- Add more appliance types here -->
        </select><br><br>
        <label for="power_rating">Power Rating:</label>
        <input type="number" name="power_rating" required><br><br>
        <label for="usage_pattern">Usage Pattern:</label>
        <input type="number" name="usage_pattern" required><br><br>
        <label for="energy_efficiency_rating">Energy Efficiency
Rating:</label>
```

```
        <select name="energy_efficiency_rating" required>
            <option value="1">Low</option>
            <option value="2">Medium</option>
            <option value="3">High</option>
        </select><br><br>
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

App.py

```
from flask import Flask, render_template, request
import pandas as pd
import numpy as np
import pdfkit
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

app = Flask(__name__)

# Home route
@app.route('/')
def index():
    return render_template('index.html')

@app.route('/predict', methods=['POST'])
def predict():
    # Extract user input from the form
    power_rating = float(request.form['power_rating'])
    usage_pattern = float(request.form['usage_pattern'])
    energy_efficiency_rating =
int(request.form['energy_efficiency_rating'])

    # Generate energy consumption prediction and plot
    energy_consumption = predict_energy_consumption(power_rating,
usage_pattern, energy_efficiency_rating)
```

```
    plot_path = visualize_energy_consumption(power_rating, usage_pattern,
energy_efficiency_rating)

    # Generate suggestions using OpenAI based on the plot
    suggestions = generate_suggestions(power_rating, usage_pattern,
energy_efficiency_rating)

    # Render the result page with energy consumption, plot, and
suggestions
    return render_template('result.html',
energy_consumption=energy_consumption, plot_path=plot_path,
suggestions=suggestions)


# Function to predict energy consumption
def predict_energy_consumption(power_rating, usage_pattern,
energy_efficiency_rating):
    data = pd.read_csv('data/appliance.csv')
    X = data[['power_rating', 'usage_pattern',
'energy_efficiency_rating']]
    y = data['energy_consumption']

    model = LinearRegression()
    model.fit(X, y)

    input_data = pd.DataFrame({'power_rating': [power_rating],
'usage_pattern': [usage_pattern], 'energy_efficiency_rating':
[energy_efficiency_rating]})
    prediction = model.predict(input_data)

    return prediction[0]

# Future trends route
@app.route('/future-trends')
def future_trends():
    data = pd.read_csv('data/appliance.csv')
    x = data['power_rating']
    y = data['energy_consumption']

    # Perform linear regression
    model = LinearRegression()
    model.fit(x.values.reshape(-1, 1), y)

    # Predict future trends
    future_x = np.linspace(min(x), max(x), num=100)
    future_y = model.predict(future_x.reshape(-1, 1))

    # Plot the future trends
    plt.figure()
    plt.plot(x, y, label='Actual')
    plt.plot(future_x, future_y, color='green', linestyle='--',
label='Future Trend')
    plt.xlabel('Power Rating')
    plt.ylabel('Energy Consumption')
    plt.title('Future Energy Consumption Trends')
    plt.legend()

    # Save the plot
    plot_path = 'static/future_trends.png'
    plt.savefig(plot_path)
    plt.close()

    return render_template('future_trends.html', plot_path=plot_path)
def generate_suggestions(power_rating, usage_pattern,
energy_efficiency_rating):
    suggestions = []

    # Check power rating
    if power_rating > 200:
        suggestions.append("Consider using appliances with lower power
ratings.")
    else:
        suggestions.append("Ensure your appliances are energy-efficient.")

    # Check usage pattern
    if usage_pattern < 1.0:
        suggestions.append("Optimize your usage pattern to avoid
unnecessary energy consumption.")
    elif usage_pattern > 2.0:
        suggestions.append("Adjust your usage pattern to reduce energy
consumption during peak hours.")

    # Check energy efficiency rating
    if energy_efficiency_rating == 1:
        suggestions.append("Upgrade to appliances with higher energy
efficiency ratings.")
    elif energy_efficiency_rating == 2:
        suggestions.append("Ensure regular maintenance of your appliances
to maximize energy efficiency.")

    return suggestions


# Function to visualize energy consumption
def visualize_energy_consumption(power_rating, usage_pattern,
energy_efficiency_rating):
    data = pd.read_csv('data/appliance.csv')
    x = data['power_rating']
    y = data['energy_consumption']

    fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(10, 5))

    # Plot the data points and linear regression
    ax1.scatter(x, y, label='Actual')
    ax1.scatter(power_rating, predict_energy_consumption(power_rating,
usage_pattern, energy_efficiency_rating),
                color='red', label='User Input')
    model = LinearRegression()
    model.fit(x.values.reshape(-1, 1), y)
    ax1.plot(x, model.predict(x.values.reshape(-1, 1)), color='orange',
label='Linear Regression')
    ax1.set_xlabel('Power Rating')
    ax1.set_ylabel('Energy Consumption')
    ax1.set_title('Energy Consumption Analysis')
    ax1.legend()

    # Plot the energy consumption trend
    trend_data = pd.read_csv('data/trend.csv')
    trend_x = trend_data['day']
```

```python
    trend_y = trend_data['energy_consumption']
    ax2.plot(trend_x, trend_y, marker='o')
    ax2.set_xlabel('Day')
    ax2.set_ylabel('Energy Consumption')
    ax2.set_title('Energy Consumption Trend')
    ax2.grid(True)
    plt.xticks(rotation=45)
    plt.tight_layout()

    # Save the plot
    plot_path = 'static/plot.png'
    plt.savefig(plot_path)
    plt.close()

    return plot_path

if __name__ == '__main__':
    app.run(debug=True)
```

Overall.py

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import LabelEncoder

# Load the data
data = pd.read_csv("data/data.csv")

# Define valid columns
valid_columns = ["Efficiency", "Appliance"]

# Check if valid columns exist in the dataset
missing_columns = [column for column in valid_columns if column not in
data.columns]
if missing_columns:
    raise ValueError(f"Columns {missing_columns} not found in the
dataset.")
```

```python
# Select only valid columns
data = data[valid_columns]

# Drop rows with missing values if any
data = data.dropna()

# Encode categorical variables using LabelEncoder
label_encoder = LabelEncoder()
categorical_columns = ["Efficiency", "Appliance"]
for column in categorical_columns:
    data[column] = label_encoder.fit_transform(data[column])

# Bivariate Analysis: Scatter plot of Efficiency vs. Appliance
plt.scatter(data["Efficiency"], data["Appliance"])
plt.xlabel("Efficiency")
plt.ylabel("Appliance")
plt.title("Bivariate Analysis: Efficiency vs. Appliance")
plt.savefig("templates/bivariate.png")  # Save the plot as an image

# Univariate Analysis: Histogram of Appliance
plt.hist(data["Appliance"])
plt.xlabel("Appliance")
plt.ylabel("Frequency")
plt.title("Univariate Analysis: Appliance")
plt.savefig("templates/univariate.png")  # Save the plot as an image

# Multivariate Analysis: Correlation Matrix Heatmap
correlation_matrix = data.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap="RdYlGn")
plt.title("Multivariate Analysis: Correlation Matrix")
plt.savefig("templates/multivariate.png")  # Save the plot as an image

# Random Forest Analysis
X = data.drop(["Appliance"], axis=1)  # Features
y = data["Appliance"]  # Target variable

# Create and fit the random forest model
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X, y)
```

```
# Feature Importance
importance = rf.feature_importances_
feature_names = X.columns

# Plotting feature importances
plt.barh(range(len(importance)), importance, align="center")
plt.yticks(range(len(feature_names)), feature_names)
plt.xlabel("Feature Importance")
plt.ylabel("Features")
plt.title("Random Forest: Feature Importance")
plt.savefig("templates/feature_importance.png")  # Save the plot as an
image

# Generate HTML file (same code as before)
html_content = f'''
<html>
<head>
    <title>Data Analysis Plots</title>
</head>
<body>
    <h1>Bivariate Analysis: Efficiency vs. Appliance</h1>
    <img src="templates/bivariate.png" alt="Bivariate Analysis">

    <h1>Univariate Analysis: Appliance</h1>
    <img src="templates/univariate.png" alt="Univariate Analysis">

    <h1>Multivariate Analysis: Correlation Matrix</h1>
    <img src="templates/multivariate.png" alt="Multivariate Analysis">

    <h1>Random Forest: Feature Importance</h1>
    <img src="templates/feature_importance.png" alt="Feature Importance">
</body>
</html>
'''

# Save HTML file (same code as before)
with open("templates/analysis.html", "w") as file:
    file.write(html_content)
```

Future.html

```html
<!DOCTYPE html>
<html>
<head>
    <title>Future Energy Consumption Trends</title>
</head>
<body>
    <h1>Future Energy Consumption Trends</h1>
    <img src="{{ plot_path }}" alt="Future Trends Plot">
</body>
</html>
```

Overall.py

```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import LabelEncoder

# Load the data
data = pd.read_csv("data/data.csv")

# Define valid columns
valid_columns = ["Efficiency", "Appliance"]

# Check if valid columns exist in the dataset
missing_columns = [column for column in valid_columns if column not in
data.columns]
if missing_columns:
    raise ValueError(f"Columns {missing_columns} not found in the
dataset.")

# Select only valid columns
data = data[valid_columns]
```

```
# Drop rows with missing values if any
data = data.dropna()

# Encode categorical variables using LabelEncoder
label_encoder = LabelEncoder()
categorical_columns = ["Efficiency", "Appliance"]
for column in categorical_columns:
    data[column] = label_encoder.fit_transform(data[column])

# Bivariate Analysis: Scatter plot of Efficiency vs. Appliance
plt.scatter(data["Efficiency"], data["Appliance"])
plt.xlabel("Efficiency")
plt.ylabel("Appliance")
plt.title("Bivariate Analysis: Efficiency vs. Appliance")
plt.savefig("templates/bivariate.png")  # Save the plot as an image

# Univariate Analysis: Histogram of Appliance
plt.hist(data["Appliance"])
plt.xlabel("Appliance")
plt.ylabel("Frequency")
plt.title("Univariate Analysis: Appliance")
plt.savefig("templates/univariate.png")  # Save the plot as an image

# Multivariate Analysis: Correlation Matrix Heatmap
correlation_matrix = data.corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap="RdYlGn")
plt.title("Multivariate Analysis: Correlation Matrix")
plt.savefig("templates/multivariate.png")  # Save the plot as an image

# Random Forest Analysis
X = data.drop(["Appliance"], axis=1)  # Features
y = data["Appliance"]  # Target variable

# Create and fit the random forest model
rf = RandomForestRegressor(n_estimators=100, random_state=42)
rf.fit(X, y)

# Feature Importance
importance = rf.feature_importances_
```

```
feature_names = X.columns

# Plotting feature importances
plt.barh(range(len(importance)), importance, align="center")
plt.yticks(range(len(feature_names)), feature_names)
plt.xlabel("Feature Importance")
plt.ylabel("Features")
plt.title("Random Forest: Feature Importance")
plt.savefig("templates/feature_importance.png")  # Save the plot as an
image

# Generate HTML file (same code as before)
html_content = f'''
<html>
<head>
    <title>Data Analysis Plots</title>
</head>
<body>
    <h1>Bivariate Analysis: Efficiency vs. Appliance</h1>
    <img src="templates/bivariate.png" alt="Bivariate Analysis">

    <h1>Univariate Analysis: Appliance</h1>
    <img src="templates/univariate.png" alt="Univariate Analysis">

    <h1>Multivariate Analysis: Correlation Matrix</h1>
    <img src="templates/multivariate.png" alt="Multivariate Analysis">

    <h1>Random Forest: Feature Importance</h1>
    <img src="templates/feature_importance.png" alt="Feature Importance">
</body>
</html>
'''

# Save HTML file (same code as before)
with open("templates/analysis.html", "w") as file:
    file.write(html_content)
```