Building an automated system to measure energy consumption, analyze the data, and provide visualizations is a multi-step project. Here, I'll guide you through the process of analyzing the energy consumption data, creating visualizations, selecting a machine learning algorithm, training the model, and evaluating its performance.

**Part 1: Analyzing the Energy Consumption Data**

**1. Data Exploration:**

- Load the PJM energy consumption dataset from Kaggle.

- Explore the dataset to understand its structure and the available features.

**2. Data Preprocessing:**

- Handle missing data: If there are missing values, decide on an appropriate strategy (e.g., interpolation or removal).

- Convert data types: Ensure that data is in the correct format for analysis.

- Check for outliers: Identify and decide how to handle any outliers.

**3. Descriptive Statistics:**

- Calculate descriptive statistics (mean, median, standard deviation, etc.) to gain insights into the data.

- Plot time series data to visualize trends and patterns.

**4. Feature Engineering (if necessary):**

- Create new features that might be relevant for the analysis, e.g., day of the week, hour of the day, holidays, etc.

**Part 2: Creating Visualizations**

**5. Time Series Plots:**

- Create time series plots of energy consumption to visualize daily, weekly, and seasonal patterns.

- Use line plots, area plots, or heatmap plots for better insights.

**6. Correlation Analysis:**

- Analyze the correlation between energy consumption and external factors (e.g., temperature, holidays, etc.).

- Create correlation matrices and heatmaps.

**7. Distribution Plots:**

- Visualize the distribution of energy consumption data (histograms, kernel density plots).

**8. Geospatial Visualizations(if applicable):**

- If the dataset contains geographical information, create maps to visualize consumption across regions.

**Part 3: Machine Learning for Predictive Analysis**

**9. Machine Learning Algorithm Selection:**

- Choose a suitable machine learning algorithm for your energy consumption prediction task. Time series forecasting models (e.g., ARIMA, LSTM) are common choices.

**10. Data Splitting:**

- Split the dataset into training and testing sets. A common split is 70-30 or 80-20 for training and testing, respectively.

**11. Model Training:**

- Train the selected machine learning model on the training dataset.

**12. Model Evaluation:**

- Evaluate the model's performance on the testing dataset using appropriate metrics (e.g., RMSE, MAE, MAPE for time series data).

**13. Hyperparameter Tuning (if needed):**

- Optimize the model's hyperparameters to improve performance.

**14. Visualize Predictions:**

- Plot the model's predictions against the actual energy consumption data to assess its accuracy.

**15. Interpret Results:**

- Interpret the model's performance and draw conclusions from the analysis.

**16. Documentation:**

- Create a comprehensive document that includes details about data preprocessing, visualizations, machine learning model selection, training, and evaluation.

**17. Sharing for Assessment:**

- Share the documentation and code with stakeholders or reviewers for assessment.

Make sure to use appropriate libraries (e.g., Python with pandas, numpy, matplotlib, seaborn, scikit-learn, TensorFlow/Keras for machine learning) to implement the above steps. This will help you build a robust system for energy consumption analysis and prediction.

Certainly, I can provide a simplified Python program to demonstrate how you can analyze energy consumption data, create visualizations, and perform a basic machine learning task. Keep in mind that this is a simplified example, and a production-ready system would involve more complexity and data processing. The example uses Python, pandas, matplotlib, and scikit-learn.

```python

Import necessary libraries

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model\_selection import train\_test\_split

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error

Load the dataset

data = pd.read\_csv('hourly\_energy\_consumption.csv')

Data Preprocessing

Assuming there's a 'datetime' column representing the date and time

data['datetime'] = pd.to\_datetime(data['datetime'])

data.set\_index('datetime', inplace=True)

Data Exploration

print(data.head())

print(data.describe())

Visualizations

Plot the time series data

plt.figure(figsize=(12, 6))

plt.plot(data.index, data['load'], label='Energy Consumption')

plt.xlabel('Date and Time')

plt.ylabel('Energy Consumption')

plt.title('Energy Consumption Time Series')

plt.legend()

plt.show()

Split data for machine learning

X = data.index.timestamp # Using timestamps as features for simplicity

y = data['load']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

Machine Learning - Linear Regression as an example

model = LinearRegression()

model.fit(X\_train.reshape(-1, 1), y\_train)

Make predictions

y\_pred = model.predict(X\_test.reshape(-1, 1))

Model Evaluation

mse = mean\_squared\_error(y\_test, y\_pred)

print(f'Mean Squared Error: {mse}')

Visualize Predictions

plt.figure(figsize=(12, 6))

plt.plot(X\_test, y\_test, label='Actual')

plt.plot(X\_test, y\_pred, label='Predicted')

plt.xlabel('Date and Time')

plt.ylabel('Energy Consumption')

plt.title('Energy Consumption Prediction vs. Actual')

plt.legend()

plt.show()

```

This program loads the dataset, preprocesses the data, explores it, creates a simple time series plot, and then demonstrates a basic machine learning example using Linear Regression to predict energy consumption. The prediction results are visualized as well. Remember that this is a simplified example, and real-world projects may require more advanced techniques and data preprocessing steps.