CPSC 8740 – Final Project

UI Design and Integration Report

Name: - Sumit Reddy Munnuru

Smart Home Energy Consumption Prediction System

Overview

The goal of the user interface for the *Smart Home Energy Consumption Prediction System* is to provide a simple and intuitive way for users to interact with the underlying machine learning model. The system is designed to predict hourly energy consumption for different household appliances based on environmental and household conditions. This week focused on constructing a high-fidelity UI that enables users to explore predictions, view energy patterns, and interpret model behavior in a straightforward and interactive manner.

The interface is implemented using Streamlit, allowing the application to run as a dynamic dashboard without the need for traditional front-end development. The UI has been structured so that users can easily input parameters, visualize outputs, and navigate through different views of the system.

Design Approach

The design was centered around clarity, usability, and maintainability. Before implementation, the UI layout was sketched out to decide how to organize the input controls, prediction outputs, and visual components. The design was divided into logical sections, each focusing on a different interaction or visualization. Streamlit's built-in components were used for consistency, and custom CSS was added to enhance readability and structure.

The interface follows a multi-tab layout, separating the workflow into four main pages: Overview, Forecast, Analysis, and About. This division helps prevent clutter and provides a clean, focused environment for user interaction. Each tab serves a unique purpose and presents data in a way that aligns with the system's predictive goals.

Technologies Used

1. Languages

Python

Python was used throughout the project for both back-end and front-end logic. All data processing, model loading, feature engineering, and UI logic were implemented using Python.

HTML/CSS (within Streamlit)
Although Streamlit manages most front-end rendering, custom CSS was embedded to improve the appearance of metric cards, spacing, color themes, and visual consistency. This allowed for a cleaner layout and more professional presentation.

---

2. Frameworks and Libraries

Streamlit
Streamlit was selected as the primary UI framework due to its simplicity and tight integration with Python. It enables quick iteration and automatic reactivity when input values change. Widgets such as sliders, dropdowns, date pickers, and time inputs are used throughout the interface.

Pandas
Pandas is used for structuring input data for the model and generating tabular outputs. It is also used to format data for charts displayed in the Forecast and Analysis sections.

Matplotlib / Streamlit Native Charts
Charts such as the hourly energy forecast line graph and appliance comparison bar graph are displayed using Streamlit's charting tools, supported by pandas DataFrames.

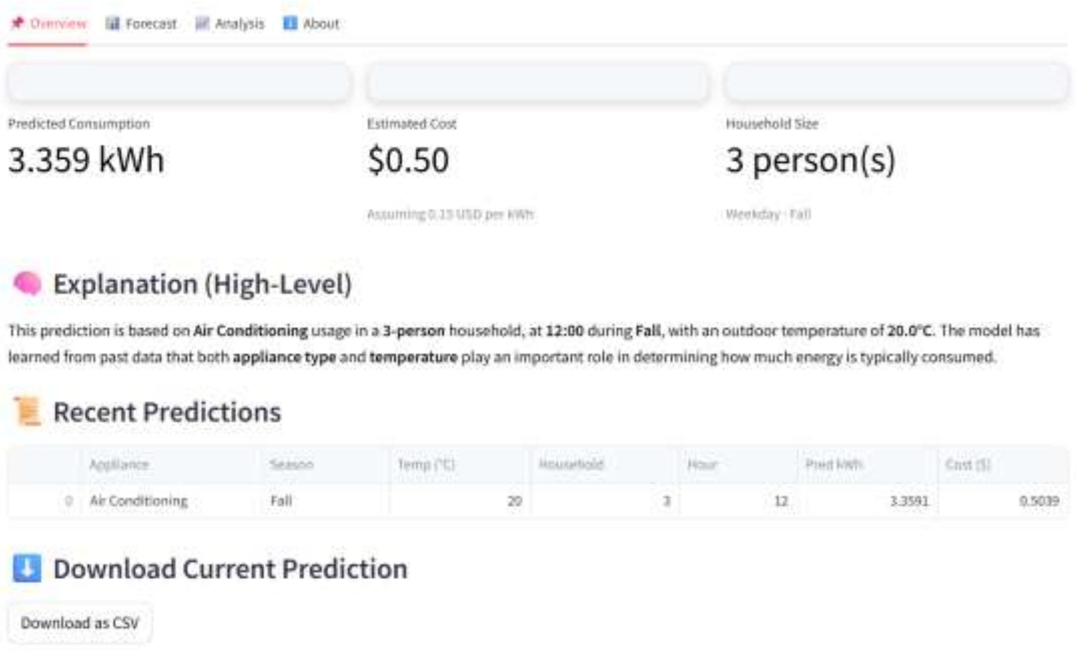
joblib
joblib is used to load the pre-trained machine learning model and the label encoders required for categorical feature transformation.

User Interface Structure

The UI is divided into four main sections that guide the user through the prediction and analysis process:

1. Overview Page

This page appears after a prediction is generated. It displays key information such as predicted energy consumption, estimated cost, and household characteristics. Metric cards were added to present this information clearly. The page also includes a summary paragraph explaining the prediction based on user-selected inputs.



Predicted Consumption
3.359 kWh

Estimated Cost
$0.50
Assuming 0.15 USD per kWh

Household Size
3 person(s)
Weekday · Fall

Explanation (High-Level)

This prediction is based on **Air Conditioning** usage in a **3-person** household, at **12:00** during **Fall**, with an outdoor temperature of **20.0°C**. The model has learned from past data that both **appliance type** and **temperature** play an important role in determining how much energy is typically consumed.

Recent Predictions

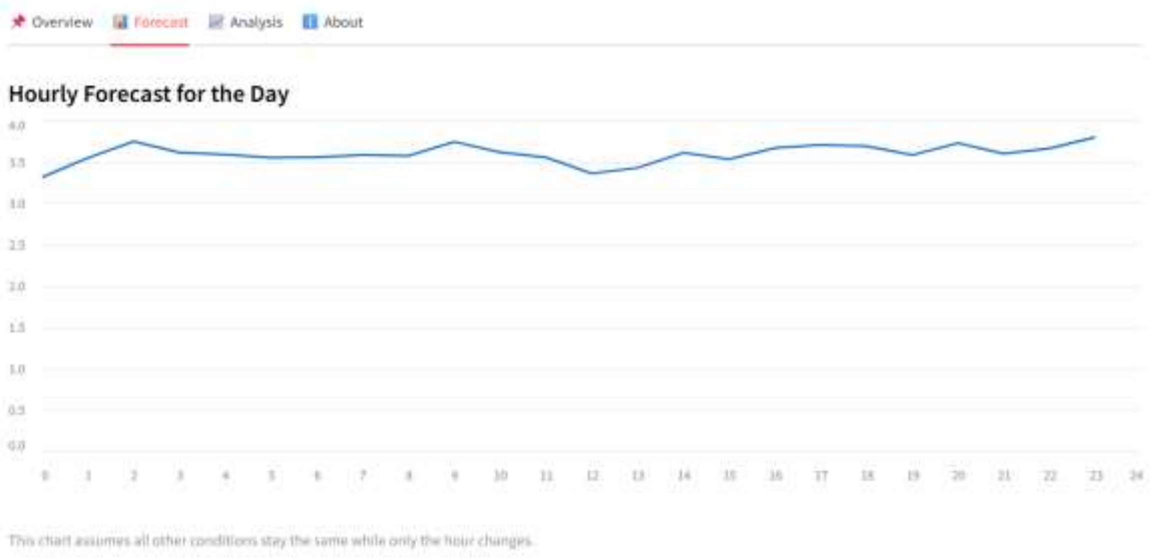
	Appliance	Season	Temp (°C)	Household	Hour	Pred kWh	Cost ($)
0	Air Conditioning	Fall	20	3	12	3.3591	0.5039

Download Current Prediction

Figure 1: Overview Page of the Application

2. Forecast Page

This page presents an hourly forecast visualization for the entire day. The model predicts consumption for each hour under the assumption that all other conditions remain constant. The resulting 24-hour prediction is plotted as a line chart to show patterns in energy demand throughout the day.

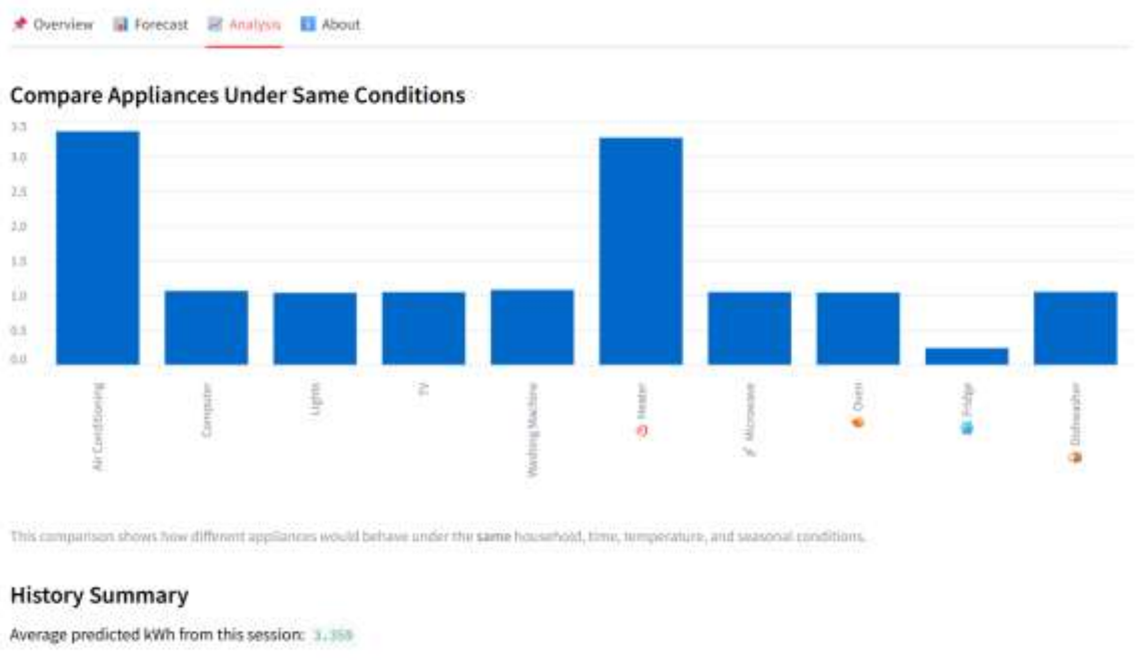


This chart assumes all other conditions stay the same while only the hour changes.

Figure 2: Hourly Energy Forecast Visualization

3. Analysis Page

This section compares energy predictions across all appliances available in the dataset. By holding selected conditions constant, the user can see how different devices contribute to overall energy consumption. A bar chart is used to display these results. The page also includes a session-based history of predictions, allowing users to track inputs and outputs across multiple runs.



Compare Appliances Under Same Conditions

This comparison shows how different appliances would behave under the **same** household, time, temperature, and seasonal conditions.

History Summary

Average predicted kWh from this session: 1.358

Figure 3: Appliance Comparison Chart

4. About Page

Provides a brief summary of the project, the dataset, and the modeling approach. The purpose of this section is to offer background context for users and evaluators reviewing the application.

Features Implemented

Frontend Features

Feature	Description
Input Controls	Sliders, dropdowns, date and time pickers allow users to set appliance type, season, temperature, household size, and time.
Tabbed Layout	Separates the UI into four pages for better organization and navigation.
Metric Cards	Compact displays showing predicted kWh, estimated cost, and additional metadata.

Feature	Description
Forecast Visualization	Displays hourly predictions on a dynamically generated line chart.
Comparison Chart	Side-by-side comparison of energy usage across appliances.
Optional Theme Mode	A light/dark mode toggle was implemented using CSS variables for improved readability.
Download Option	Users can export prediction results in CSV format.

Backend Features

Feature	Description
Feature Engineering	Transforms user input (time, date, season, appliance type) into the numeric features expected by the model.
Model Prediction	Calls the trained Random Forest model to compute predicted energy consumption.
Encoding	Converts categorical variables into encoded form using joblib-loaded encoders.
Session History	Stores past predictions within a session for later review in the Analysis tab.

Challenges Encountered

Several challenges were encountered while integrating the UI with the model:

1. Synchronizing UI Inputs with Model Requirements

The model requires eight specific features in numerical form. Ensuring that the transformations in the UI matched those used during model training required careful replication of preprocessing logic.

2. Layout Management in Streamlit

Streamlit renders components in a top-down structure. Organizing components into separate tabs and arranging visual elements such as metric cards required multiple iterations before the layout felt balanced and readable.

3. Theme Consistency

Implementing dark mode introduced styling inconsistencies due to Streamlit's rerendering behavior. This was resolved by switching to CSS variables rather than attempting to modify HTML classes dynamically.

4. Chart Readability

Charts needed to adapt to theme changes for consistency. Matplotlib settings were adjusted to improve visibility in both light and dark modes.

Next Steps

Future UI improvements may include:

- Enhanced accessibility features, including tooltips and clearer error handling.
- Adding user-selectable forecasting ranges, such as multi-day or weekly predictions.
- Further refinement of spacing and design elements for a more cohesive layout.
- Deployment of the app using Streamlit Community Cloud to support external testing.

Overall, the UI now provides a complete interface that aligns with the system's predictive goals and supports meaningful user exploration of smart home energy patterns.