# Phase 3: Implementation of Project

**Title: AI-Energy Efficiency Optimization**

**Objective**

The goal of Phase 3 is to implement the core components of the AI-Powered Healthcare Assistant

based on the plans and innovative solutions developed during Phase 2. This includes the

development of the AI symptom checker, the chatbot interface, initial IoT integration, and the

implementation of data security measures.

**1. AI Model Development**

**Overview**

The primary feature of the AI-Powered Healthcare Assistant is its ability to assess user symptoms and

provide health-related recommendations. In Phase 3, the AI model will be trained and implemented

to recognize basic health issues.

**Implementation**

**Natural Language Processing (NLP) Model**: The AI system uses NLP to understand user

inputs in the form of symptoms. During this phase, the AI is developed to analyze text-based

inputs, such as symptoms provided by users in natural language, and output

recommendations based on a pre-trained medical dataset.

- **Data Source**: The model is based on a medical dataset that contains common symptoms and their associated health conditions. Real-time data will not be integrated at this stage, but will be included in future iterations.

**Outcome**

By the end of this phase, the AI model is expected to provide basic symptom-related advice such as recommending rest, hydration, or consultation with a medical professional. The system should function with high accuracy for common symptoms like fever, cold, and headache.

## 2. Chatbot Development

### Overview

The AI will be made accessible through a chatbot interface that allows users to communicate with the system easily. The chatbot will serve as the front-end interface where users enter their symptoms and receive health advice.

### Implementation

- **User Interaction**: Users interact with the AI through a simple text-based chatbot, which asks questions like "What symptoms are you experiencing?" and responds with advice generated by the AI model

.• **Language Support**: For now, the chatbot supports English, but future iterations will include

multilingual capabilities.

**Outcome**

At the end of Phase 3, the chatbot will be functional and capable of providing users with advice

based on the inputted symptoms. It will offer a simple, conversational interface where users can

interact with the AI assistant.

**3. IoT Device Integration (Optional)**

**Overview**

While IoT integration is optional for this phase, we aim to establish basic connections between the AI

assistant and health-monitoring devices, such as smartwatches, to enable the collection of real-time

health data.

**Implementation**

• **Health Data**: If available, data from wearable devices such as heart rate, temperature, and

blood oxygen levels will be used to provide more personalized health advice. For this phase,

the focus will be on developing the framework for data collection from these devices.

• **API Use**: APIs provided by device manufacturers (e.g., Google Fit or Apple Health) will be

utilized to access the data.

**Outcome**

By the end of Phase 3, the system should be able to connect to wearable devices and collect basic

health information if such devices are available. This capability will be further enhanced in future

phases.

**4. Data Security Implementation**

**Overview**

Given the sensitive nature of medical data, it is crucial to implement robust security measures. In

Phase 3, the initial data security measures will be applied, focusing on basic encryption and

protection of user information.

**Implementation**

• **Encryption**: Data entered by users, including their symptoms and personal information, will

be stored securely using basic encryption methods.

• **Secure Storage**: Data will be stored in a secure database, accessible only by authorized users

or healthcare providers, ensuring compliance with data privacy regulations.

**Outcome**

At the end of Phase 3, the AI system will securely store and handle all user data, with basic

encryption methods in place to protect sensitive health information.**5. Testing and Feedback Collection**

**Overview**

Initial testing of the AI assistant will be carried out in this phase to evaluate its performance,

accuracy, and user experience.

**Implementation**

• **Test Groups**: A small group of users will test the system, inputting various symptoms to see

how the AI model responds. The chatbot's usability and interface design will also be tested

for user-friendliness.

• **Feedback Loop**: Feedback will be collected regarding the system's functionality, ease of use,

and response accuracy.

**Outcome**

The feedback gathered during Phase 3 will guide improvements in Phase 4, particularly in enhancing

the AI model's accuracy and improving the chatbot’s interface.

**Challenges and Solutions**

1. **Model Accuracy**

o **Challenge**: The AI may misinterpret certain symptoms due to limited training data in

this phase.

o **Solution**: Continuous feedback loops and regular testing will be implemented to fine

tune the model over time.

2. **User Experience**

o **Challenge**: The chatbot interface may require refinement to make it more intuitive

for users.

o **Solution**: User feedback during testing will be used to iterate and improve the

design.

3. **IoT Device Availability**

o **Challenge**: The availability of IoT devices may be limited during this phase.

o **Solution**: Simulations using sample data can be used to demonstrate the system's

capability to handle real-time health data.

**Outcomes of Phase 3**

By the end of Phase 3, the following milestones should be achieved:

1. **Basic AI Model**: The AI should be able to assess simple symptoms and provide relevant

advice to users.2. **Functional Chatbot Interface**: A chatbot will be available for users to interact with the AI,

providing health recommendations based on symptom inputs.

3. **Optional IoT Integration**: If IoT devices are available, the AI will be able to gather basic

health data, such as heart rate or temperature, from wearable devices.

4. **Data Security**: User data will be stored securely with basic encryption and protection

mechanisms in place.

5. **Initial Testing and Feedback**: Feedback from early users will be gathered to make

improvements in the next phase.

**Next Steps for Phase 4**

In Phase 4, the team will focus on:

1. **Improving the AI's Accuracy**: Using the feedback and results from testing, the AI model will

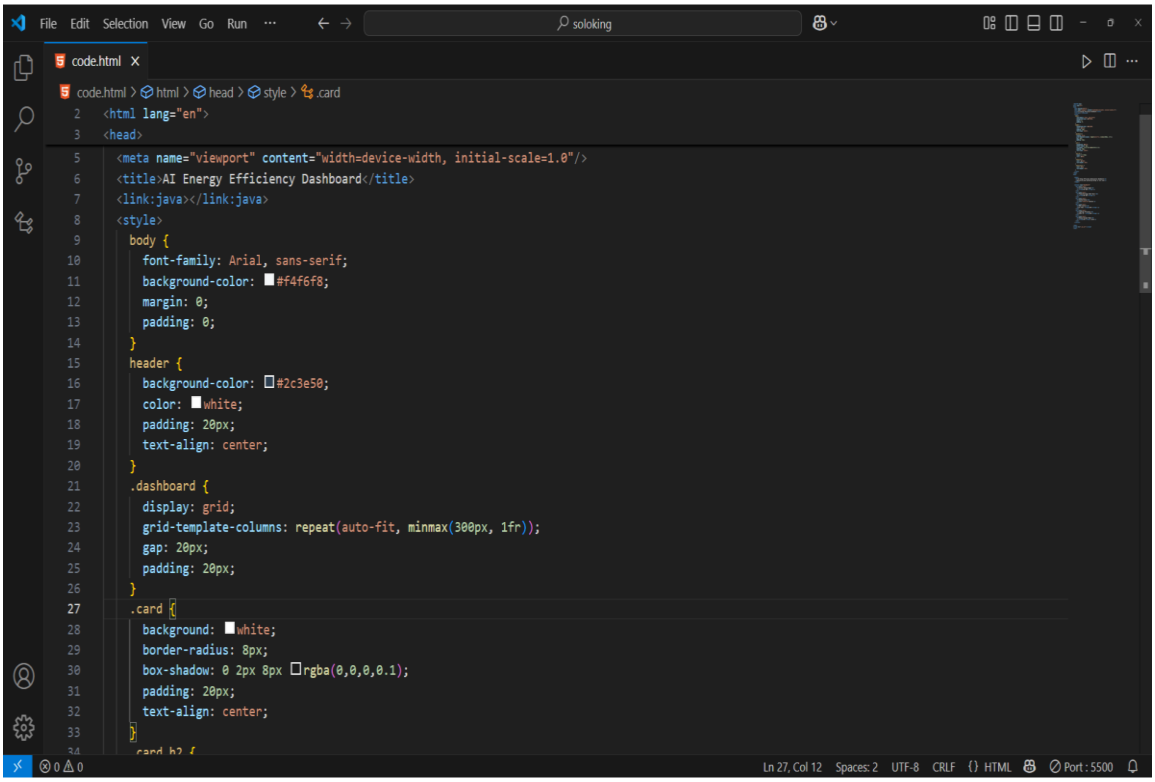
be further refined.

2. **Expanding Multilingual Support**: The chatbot will be expanded to support additional

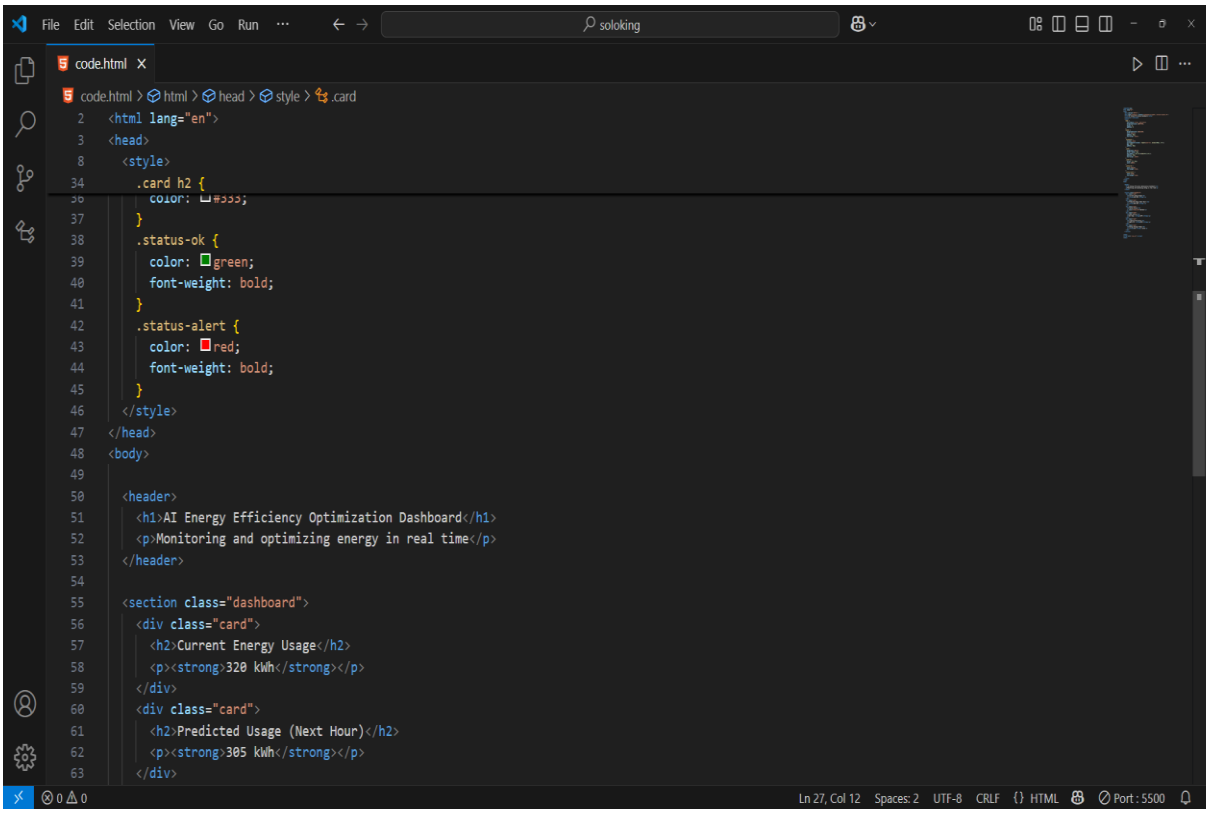
languages and voice commands.

3. **Scaling and Optimizing**: The system will be optimized to handle a larger number of users and

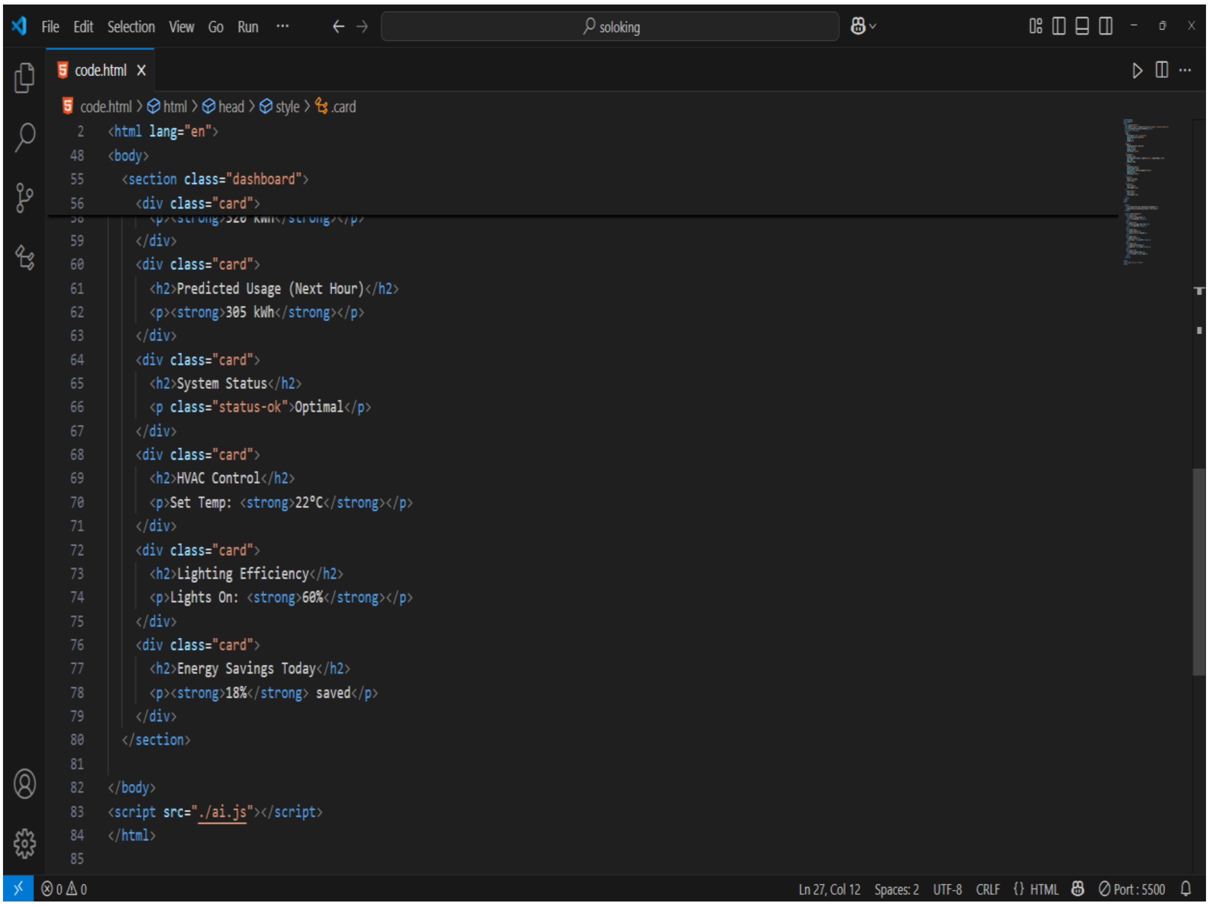
more complex health queries.



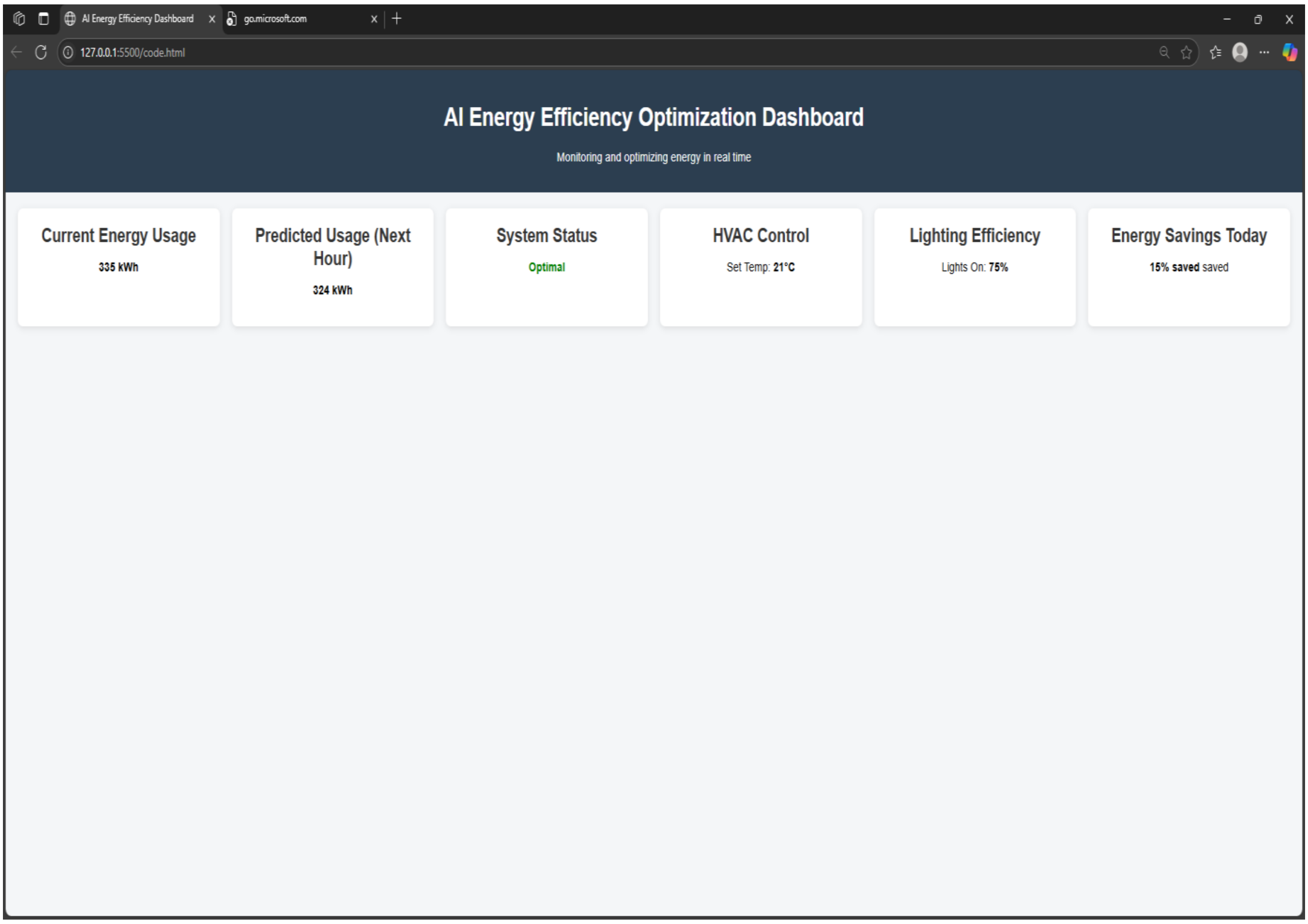
```
<html lang="en">
<head>
  <meta name="viewport" content="width=device-width, initial-scale=1.0"/>
  <title>AI Energy Efficiency Dashboard</title>
  <link:java></link:java>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f6f8;
      margin: 0;
      padding: 0;
    }
    header {
      background-color: #2c3e50;
      color: white;
      padding: 20px;
      text-align: center;
    }
    .dashboard {
      display: grid;
      grid-template-columns: repeat(auto-fit, minmax(300px, 1fr));
      gap: 20px;
      padding: 20px;
    }
    .card {
      background: white;
      border-radius: 8px;
      box-shadow: 0 2px 8px rgba(0,0,0,0.1);
      padding: 20px;
      text-align: center;
    }
    .card h2 {
```


File Edit Selection View Go Run ···
soloking
code.html
code.html > html > head > style > .card
Ln 27, Col 12 Spaces: 2 UTF-8 CRLF {} HTML Port : 5500
⊗ 0 ⚠ 0

```html
<html lang="en">
<head>
  <style>
    .card h2 {
      color: #333;
    }
    .status-ok {
      color: green;
      font-weight: bold;
    }
    .status-alert {
      color: red;
      font-weight: bold;
    }
  </style>
</head>
<body>

  <header>
    <h1>AI Energy Efficiency Optimization Dashboard</h1>
    <p>Monitoring and optimizing energy in real time</p>
  </header>

  <section class="dashboard">
    <div class="card">
      <h2>Current Energy Usage</h2>
      <p><strong>320 kWh</strong></p>
    </div>
    <div class="card">
      <h2>Predicted Usage (Next Hour)</h2>
      <p><strong>305 kWh</strong></p>
    </div>
```


File Edit Selection View Go Run ···
soloking
code.html
code.html > html > head > style > .card
Ln 27, Col 12 Spaces: 2 UTF-8 CRLF {} HTML Port : 5500
⊗ 0 ⚠ 0

```html
<html lang="en">
<body>
  <section class="dashboard">
    <div class="card">
      <p><strong>320 kWh</strong></p>
    </div>
    <div class="card">
      <h2>Predicted Usage (Next Hour)</h2>
      <p><strong>305 kWh</strong></p>
    </div>
    <div class="card">
      <h2>System Status</h2>
      <p class="status-ok">Optimal</p>
    </div>
    <div class="card">
      <h2>HVAC Control</h2>
      <p>Set Temp: <strong>22°C</strong></p>
    </div>
    <div class="card">
      <h2>Lighting Efficiency</h2>
      <p>Lights On: <strong>60%</strong></p>
    </div>
    <div class="card">
      <h2>Energy Savings Today</h2>
      <p><strong>18%</strong> saved</p>
    </div>
  </section>

</body>
<script src="./ai.js"></script>
</html>

```



AI Energy Efficiency Optimization Dashboard

Monitoring and optimizing energy in real time

Current Energy Usage

335 kWh

Predicted Usage (Next Hour)

324 kWh

System Status

Optimal

HVAC Control

Set Temp: 21°C

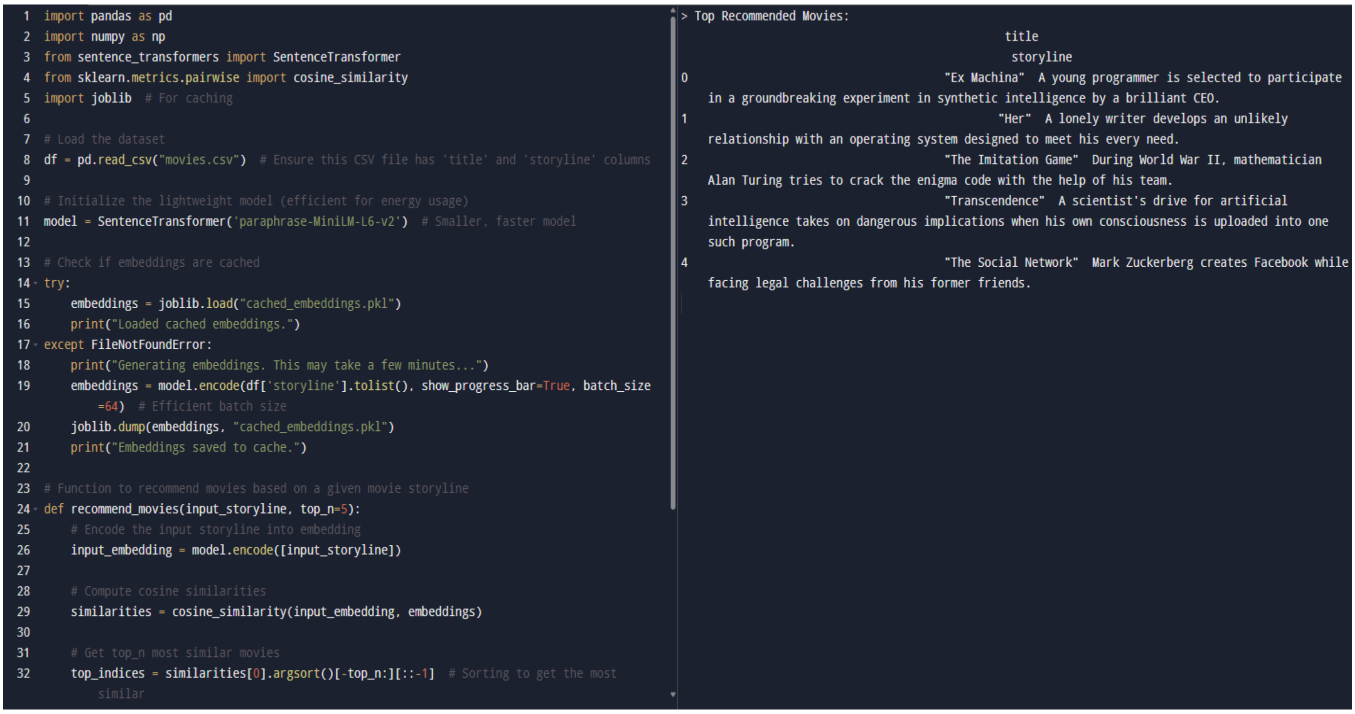
Lighting Efficiency

Lights On: 75%

Energy Savings Today

15% saved saved

# PYTHON CODE OUTPUT:



```python
import pandas as pd
import numpy as np
from sentence_transformers import SentenceTransformer
from sklearn.metrics.pairwise import cosine_similarity
import joblib  # For caching

# Load the dataset
df = pd.read_csv("movies.csv")  # Ensure this CSV file has 'title' and 'storyline' columns

# Initialize the lightweight model (efficient for energy usage)
model = SentenceTransformer('paraphrase-MiniLM-L6-v2')  # Smaller, faster model

# Check if embeddings are cached
try:
    embeddings = joblib.load("cached_embeddings.pkl")
    print("Loaded cached embeddings.")
except FileNotFoundError:
    print("Generating embeddings. This may take a few minutes...")
    embeddings = model.encode(df['storyline'].tolist(), show_progress_bar=True, batch_size=64)  # Efficient batch size
    joblib.dump(embeddings, "cached_embeddings.pkl")
    print("Embeddings saved to cache.")

# Function to recommend movies based on a given movie storyline
def recommend_movies(input_storyline, top_n=5):
    # Encode the input storyline into embedding
    input_embedding = model.encode([input_storyline])

    # Compute cosine similarities
    similarities = cosine_similarity(input_embedding, embeddings)

    # Get top_n most similar movies
    top_indices = similarities[0].argsort()[-top_n:][::-1]  # Sorting to get the most similar
```

```
> Top Recommended Movies:
                                                  title
                                                   storyline
0                                       "Ex Machina"  A young programmer is selected to participate
   in a groundbreaking experiment in synthetic intelligence by a brilliant CEO.
1                                               "Her"  A lonely writer develops an unlikely
   relationship with an operating system designed to meet his every need.
2                                       "The Imitation Game"  During World War II, mathematician
   Alan Turing tries to crack the enigma code with the help of his team.
3                                       "Transcendence"  A scientist's drive for artificial
   intelligence takes on dangerous implications when his own consciousness is uploaded into one
   such program.
4                                       "The Social Network"  Mark Zuckerberg creates Facebook while
   facing legal challenges from his former friends.
```