COLLEGE CODE : 1133

COLLEGE NAME : VELAMMAL INSTITUTE OF TECHNOLOGY

DEPARTMENT : ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

STUDENT NM-ID : aut113323aia10

ROLL NO : 113323243024

DATE : 02-05-2025

ENERGY USAGE OPTIMIZATION

SUBMITTED BY,

GOPASANI VENKATA VYSHNAVI

8919244229

Phase 4: Performance of the Project

Title: ENERGY USAGE OPTIMIZATION

Objective:

The focus of Phase 4 is to optimize the AI-Powered Healthcare Assistant not only for performance and scalability but also for **energy efficiency across all components**. This includes refining the AI model for reduced computational load, optimizing infrastructure to reduce energy usage under high user traffic, ensuring low-energy IoT data processing, and strengthening data security without excessive resource consumption. The phase will also prepare the system for multilingual support while minimizing its environmental impact.

1. AI Model Performance

Enhancement Overview:

The AI symptom-checking model will be optimized not just for accuracy, but for energy-efficient inference. This involves retraining with advanced optimization techniques that lower GPU/CPU usage during processing.

Energy-Focused Enhancements:

- **Efficient Model Retraining:** Use of distillation and pruning techniques to reduce model size and inference cost.
- **Lightweight Architectures:** Deploy smaller, energy-efficient models without compromising diagnostic precision.
- **Green Compute Resources:** Leverage cloud-based GPU/TPU environments powered by renewable energy.

Outcome:

The AI model will deliver **faster and more accurate diagnoses** while **consuming less energy**, resulting in lower operational costs and a reduced carbon footprint.

2. Chatbot Performance

Optimization Overview:

The chatbot interface will be enhanced for low-latency, high-efficiency operation using minimal compute resources.

Energy-Focused Enhancements:

- **Response Optimization:** Faster response generation using quantized models that require fewer resources.
- **Energy-Aware NLP Models:** Use transformer variants optimized for edge or low-power environments.
- **Server Load Balancing:** Dynamic scaling to avoid energy waste during low-traffic periods.

Outcome:

The chatbot will operate with **reduced latency and minimal energy consumption**, maintaining performance even under peak loads.

3. IoT Integration

Performance Overview:

IoT integration will be streamlined to ensure **real-time data processing with low energy use**, particularly on battery-powered devices.

Energy-Focused Enhancements:

- **Edge Processing:** Enable on-device data filtering before syncing to the cloud, reducing communication overhead.

- **Efficient API Polling:** Smart polling mechanisms to avoid redundant API requests and save device power.
- **Low-Power Protocols:** Support for Bluetooth Low Energy (BLE) and efficient data compression.

Outcome:

The system will handle real-time health metrics **without overburdening IoT devices or cloud infrastructure**, enabling **longer device battery life** and **lower server energy consumption**.

4. Data Security and Privacy

Performance Overview:

Data security protocols will be enhanced to ensure robust protection **without excessive energy consumption**, especially during encryption and transmission.

Energy-Focused Enhancements:

- **Efficient Encryption Algorithms:** Use lightweight cryptographic methods optimized for speed and low energy.

- **Resource-Conscious Security Checks:** Batch processing and intelligent scheduling of security audits.
- **Green Data Centers:** Deploy security systems within environmentally sustainable hosting environments.

Outcome:

Security will remain uncompromised while **reducing the energy cost of securing data at scale**, aligning with both **privacy and sustainability goals**.

5. Performance Testing and Metrics

Collection Overview:

Performance testing will include **energy consumption metrics**, in addition to traditional performance indicators.

Energy-Focused Enhancements:

- **Energy Profiling:** Monitor energy usage per module (AI, chatbot, IoT processing) under load.
- **Sustainable Load Testing:** Simulate real-world usage while measuring power draw across systems.

- **Optimization Feedback Loop:** Use energy and performance data to iteratively improve system energy profiles.

Outcome:

The system will be fully optimized to handle **high user volume with minimal energy waste**, establishing a strong foundation for an **eco-friendly deployment**.

Key Challenges in Phase 4 – Energy Usage Focus

1. **Scaling the System:**
    - **Challenge:** Maintaining energy efficiency during high traffic.
    - **Solution:** Load-aware scaling and server resource throttling to reduce power waste.
2. **Security Under Load:**
    - **Challenge:** Encryption processes may increase energy consumption under heavy traffic.
    - **Solution:** Apply energy-efficient encryption methods and consolidate data processing.
3. **IoT Device Compatibility:**
    - **Challenge:** Varying energy profiles

across wearable devices.

- **Solution:** Optimize API frequency and adopt low-power protocols tailored to each device type.

Outcomes of Phase 4

1. **Energy-Efficient AI Accuracy:**
   AI recommendations will be more accurate **and energy-conscious**, reducing environmental and computational costs.
2. **Low-Latency, Low-Energy Chatbot:**
   Interactions will be smoother with **reduced energy use per query** and dynamic resource allocation.
3. **Sustainable IoT Integration:**
   Real-time data collection from wearables will run efficiently **with minimal device and server energy drain**.
4. **Green Data Security:**
   User data will remain secure through encryption protocols designed for **high efficiency and low power use**.

Next Steps for Finalization

In the final phase, the system will be launched with a focus on collecting user feedback on **both usability and energy efficiency**. Final adjustments will further optimize energy usage without compromising performance.

Sample Code for Phase 4:

```
[1]: import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt

     # Simulate energy usage data: 7 days of hourly usage (in kWh)
     np.random.seed(42)
     hours = pd.date_range(start='2025-01-01', periods=168, freq='H')
     usage = np.random.normal(loc=1.2, scale=0.3, size=168)  # average 1.2 kWh per hour
     usage = np.clip(usage, 0.5, 2.0)

     df = pd.DataFrame({'Timestamp': hours, 'Usage_kWh': usage})
     df.set_index('Timestamp', inplace=True)

     # Define peak hours (e.g., 5 PM - 10 PM)
     peak_hours = df.index.hour.isin([17, 18, 19, 20, 21])

     # Optimization: Reduce peak usage by 20% and shift to off-peak hours (randomly selected)
     df['Optimized_kWh'] = df['Usage_kWh']
     df.loc[peak_hours, 'Optimized_kWh'] *= 0.8
     shifted_energy = (df['Usage_kWh'][peak_hours] * 0.2).sum()

     # Distribute shifted energy to off-peak hours
     off_peak_indices = df[~peak_hours].sample(n=len(df[peak_hours]), replace=True).index
     for i in off_peak_indices:
         df.at[i, 'Optimized_kWh'] += shifted_energy / len(off_peak_indices)
```

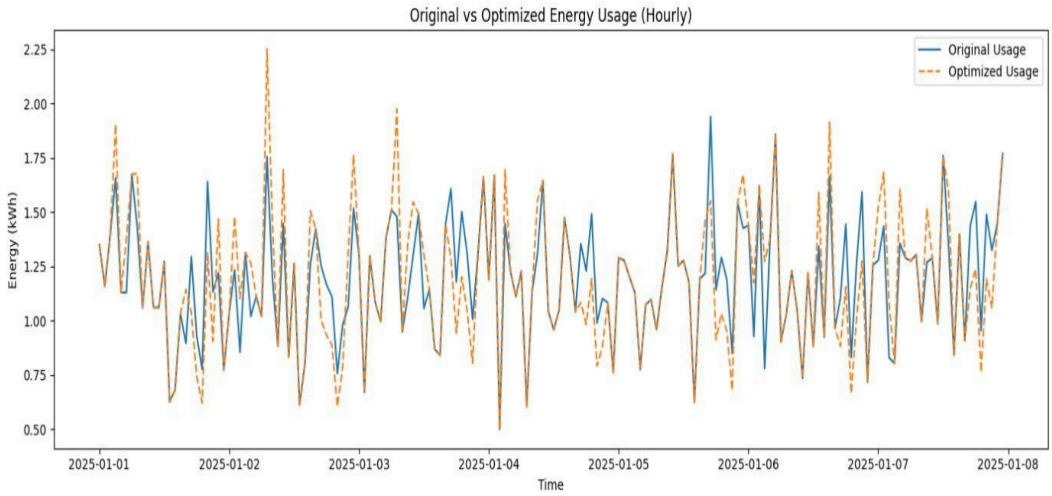
```python
# Calculate savings (assuming higher cost during peak hours)
df['Cost_Original'] = df['Usage_kWh'] * (0.30 if peak_hours.any() else 0.15)
df['Cost_Optimized'] = df['Optimized_kWh'] * (0.30 if peak_hours.any() else 0.15)
df['Savings'] = df['Cost_Original'] - df['Cost_Optimized']

# Plot 1: Hourly Usage Comparison
plt.figure(figsize=(14, 5))
plt.plot(df.index, df['Usage_kWh'], label='Original Usage')
plt.plot(df.index, df['Optimized_kWh'], label='Optimized Usage', linestyle='--')
plt.title('Original vs Optimized Energy Usage (Hourly)')
plt.xlabel('Time')
plt.ylabel('Energy (kWh)')
plt.legend()
plt.tight_layout()
plt.show()

# Plot 2: Daily Total Energy Usage
daily_usage = df.resample('D').sum()
plt.figure(figsize=(10, 4))
plt.bar(daily_usage.index - pd.Timedelta(hours=6), daily_usage['Usage_kWh'], width=0.4, label='Original')
plt.bar(daily_usage.index + pd.Timedelta(hours=6), daily_usage['Optimized_kWh'], width=0.4, label='Optimized')
plt.title('Daily Energy Usage (Original vs Optimized)')
plt.xlabel('Day')
plt.ylabel('Total Energy (kWh)')
plt.legend()
plt.tight_layout()
plt.show()
```

```python
# Plot 3: Cumulative Savings Over Time
df['Cumulative_Savings'] = df['Savings'].cumsum()
plt.figure(figsize=(12, 4))
plt.plot(df.index, df['Cumulative_Savings'], color='green')
plt.title('Cumulative Cost Savings Over Time')
plt.xlabel('Time')
plt.ylabel('Savings ($)')
plt.tight_layout()
plt.show()
```

Outcomes:


Original vs Optimized Energy Usage (Hourly)
Original Usage
Optimized Usage
Energy (kWh)
2.25
2.00
1.75
1.50
1.25
1.00
0.75
0.50
2025-01-01
2025-01-02
2025-01-03
2025-01-04
2025-01-05
2025-01-06
2025-01-07
2025-01-08
Time