COLLEGE CODE : 1133

COLLEGE NAME : VELAMMAL INSTITUTE OF TECHNOLOGY

DEPARTMENT : ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

STUDENT NM-ID : aut113323aia47

ROLL NO : 113323243096

DATE : 04.05.2025

TECHNOLOGY-PROJECT NAME: BUILDING PERFORMANCE ANALYSIS

SUBMITTED BY :

1. SANJAI S
2. GOKULAKRISHNAN
3. KAMESHWARN P N
4. MIDHUN S
5. DANISH MARK J M

INDEX OF THE DOCUMENTS

Phase 5: Project Demonstration & Documentation

Title:

AI-Driven Building Performance Analysis

Abstract:
The final phase of this project culminates in the demonstration, testing, and documentation of a robust AI and IoT-integrated platform designed to analyze and optimize building energy performance and occupant comfort. This system combines advanced machine learning models, a real-time data dashboard, digital twin integration, and cloud-based scalability. The final deliverables include source code, performance metrics, ERP integration plans, and complete technical documentation. Real-time simulations and feedback loops were used to finalize performance optimizations.

Index:

1. Project Demonstration

Overview

The fully functional system is demonstrated to stakeholders, showcasing its integration, real-time responsiveness, energy recommendations, and visualization capabilities.

Demonstration Details

- System Walkthrough : End-to-end flow from sensor data collection to real-time dashboard display and AI recommendations.
- AI Accuracy: Simulation of building scenarios to demonstrate how AI identifies inefficiencies in lighting, HVAC, and occupancy.
- IoT Integration: Live data emulated from smart devices (temperature, CO_2, humidity sensors).
- Performance Metrics : Demonstrated load handling and system speed under multi-user access.
- Security Measures : Data encryption during cloud sync and compliance with best practices.

Outcome

Stakeholders gain a full understanding of the system's real-time capabilities and operational reliability in a smart building context.

2. Project Documentation

Overview

Full technical and user documentation has been prepared for future developers, users, and system administrators.

Documentation Sections

- System Architecture : Includes block diagrams for AI logic, IoT communication, and cloud deployment.
- Code Documentation: Includes source code for AI models, dashboard backend/frontend, APIs, and digital twin logic.
- User Manual: Guide for facility managers to use the dashboard and interpret performance reports.
- Admin Manual : Details on maintaining the system, restarting services, and integrating new sensors.
- Testing Reports : Metrics from stress tests, model evaluations, and usability tests.

Outcome

A complete technical reference ensures maintainability, future development, and clear understanding by stakeholders.

3. Feedback and Final Adjustments

Overview

Collected feedback from mentors, peer testers, and potential users informed the final refinements.

Steps

- Feedback Collection: During pilot deployment and dashboard walkthroughs.
- Refinement: Resolved UI bugs, improved loading time, and added help tooltips.
- Final Testing: Comprehensive test suite run to verify AI, sensor data sync, and dashboard responsiveness.

Outcome

System was optimized for real-world usability and robustness across multiple devices and users.

4. Final Project Report Submission

Overview

A comprehensive report encapsulating the entire development lifecycle and learnings.

Report Sections

- Executive Summary : High-level overview of project goals and achievements.

- Phase Recap: Summary of Phases 1 to 4 with outcomes and learnings.
- Challenges & Solutions: Tackled IoT noise issues, AI training data limitations, and real-time integration challenges.
- Final Outcomes: Ready-for-deployment platform with digital twin capabilities and cloud scalability.

Outcome

Project deliverables are compiled and formally submitted, showcasing technical and practical readiness.

5. Project Handover and Future Works

Overview

The project is prepared for future expansion and institutional use.

Handover Details

- Next Steps:
  - Integration with Building Information Modeling (BIM) tools like Revit.
  - Expansion to predict occupancy trends and energy cost forecasts.
  - Enable voice-based AI assistant for facility operations.

Outcome

A clear roadmap for scaling the system, enabling it to evolve into a city-scale smart infrastructure tool.

Screenshots Code and Progress of the Project :

https://v0-smart-building-analysis-code.vercel.app/

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.ensemble import HistGradientBoostingRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error
from sklearn.inspection import permutation_importance

# Simulate IoT data with unique smart building features
np.random.seed(2024)
data = pd.DataFrame({
    'temperature': np.random.normal(22, 2.5, 400),
    'humidity': np.random.normal(47, 11, 400),
    'co2': np.random.normal(610, 90, 400),
    'occupancy': np.random.randint(0, 4, 400),  # 0=empty,1=low,2=medium,3=high
    'daylight': np.random.normal(190, 55, 400),
    'hvac_power': np.random.normal(48, 14, 400),
    'window_open': np.random.randint(0, 2, 400),
    'smart_blinds_position': np.random.uniform(0, 1, 400),  # 0=closed, 1=open
    'air_purifier_status': np.random.randint(0, 2, 400),    # 0=off, 1=on
    'solar_panel_output': np.random.normal(20, 5, 400),     # kW generated onsite
    'energy_consumption': np.random.normal(155, 32, 400)
})
# Enhanced comfort index including air quality and smart blinds effect
data['comfort_index'] = 100 - (
    abs(data['temperature'] - 22) +
    abs(data['humidity'] - 48) +
    data['co2'] / 120 -
    5 * data['smart_blinds_position'] +
    3 * data['air_purifier_status']
)
```

```python
# Features and targets
features = [
    'temperature', 'humidity', 'co2', 'occupancy', 'daylight',
    'hvac_power', 'window_open', 'smart_blinds_position',
    'air_purifier_status', 'solar_panel_output'
]
X = data[features]
y_energy = data['energy_consumption']
y_comfort = data['comfort_index']

# Train/test split
X_train, X_test, y_train_energy, y_test_energy = train_test_split(X, y_energy, test_size=0.25, random_state=2024)
_, _, y_train_comfort, y_test_comfort = train_test_split(X, y_comfort, test_size=0.25, random_state=2024)

# Train HistGradientBoosting models (fast and handles categorical-like features)
model_energy = HistGradientBoostingRegressor(max_iter=200, random_state=2024)
model_comfort = HistGradientBoostingRegressor(max_iter=200, random_state=2024)
model_energy.fit(X_train, y_train_energy)
model_comfort.fit(X_train, y_train_comfort)

# Predict
pred_energy = model_energy.predict(X_test)
pred_comfort = model_comfort.predict(X_test)

# Evaluate with MAE
mae_energy = mean_absolute_error(y_test_energy, pred_energy)
mae_comfort = mean_absolute_error(y_test_comfort, pred_comfort)

print(f"Energy Prediction MAE: {mae_energy:.2f}")
print(f"Comfort Prediction MAE: {mae_comfort:.2f}")

# Permutation feature importance for interpretability
perm_imp_energy = permutation_importance(model_energy, X_test, y_test_energy, n_repeats=15, random_state=2024)
perm_imp_comfort = permutation_importance(model_comfort, X_test, y_test_comfort, n_repeats=15, random_state=2024)

feat_imp_energy = pd.Series(perm_imp_energy.importances_mean, index=features).sort_values(ascending=False)
feat_imp_comfort = pd.Series(perm_imp_comfort.importances_mean, index=features).sort_values(ascending=False)
```

```python
print("\nEnergy Model Permutation Importances:")
print(feat_imp_energy)
print("\nComfort Model Permutation Importances:")
print(feat_imp_comfort)

# Visualization
plt.figure(figsize=(16, 8))

plt.subplot(2, 3, 1)
plt.scatter(y_test_energy, pred_energy, alpha=0.6, color='navy')
plt.plot([y_test_energy.min(), y_test_energy.max()], [y_test_energy.min(), y_test_energy.max()], 'r--')
plt.title("Energy Consumption: Actual vs Predicted")
plt.xlabel("Actual Energy (kWh)")
plt.ylabel("Predicted Energy (kWh)")

plt.subplot(2, 3, 2)
feat_imp_energy.plot(kind='bar', color='mediumblue')
plt.title("Energy Model Feature Importance (Permutation)")
plt.xticks(rotation=45)

plt.subplot(2, 3, 3)
plt.barh(['Smart Blinds Position', 'Air Purifier Status', 'Solar Panel Output'],
         [data['smart_blinds_position'].mean(), data['air_purifier_status'].mean(), data['solar_panel_output'].mean()],
         color=['orange', 'green', 'gold'])
plt.title("Average Smart Tech Usage in Dataset")

plt.subplot(2, 3, 4)
plt.scatter(y_test_comfort, pred_comfort, alpha=0.6, color='darkgreen')
plt.plot([y_test_comfort.min(), y_test_comfort.max()], [y_test_comfort.min(), y_test_comfort.max()], 'r--')
plt.title("Comfort Index: Actual vs Predicted")
plt.xlabel("Actual Comfort")
plt.ylabel("Predicted Comfort")
```

```
plt.subplot(2, 3, 5)
feat_imp_comfort.plot(kind='bar', color='darkgreen')
plt.title("Comfort Model Feature Importance (Permutation)")
plt.xticks(rotation=45)

plt.subplot(2, 3, 6)
plt.boxplot([data['smart_blinds_position'], data['air_purifier_status'], data['solar_panel_output']],
            labels=['Smart Blinds', 'Air Purifier', 'Solar Output'])
plt.title("Smart Tech Feature Distributions")

plt.tight_layout()
plt.show()
```

OUTPUT: Energy Prediction MAE:
32.21 Comfort Prediction MAE:
1.29

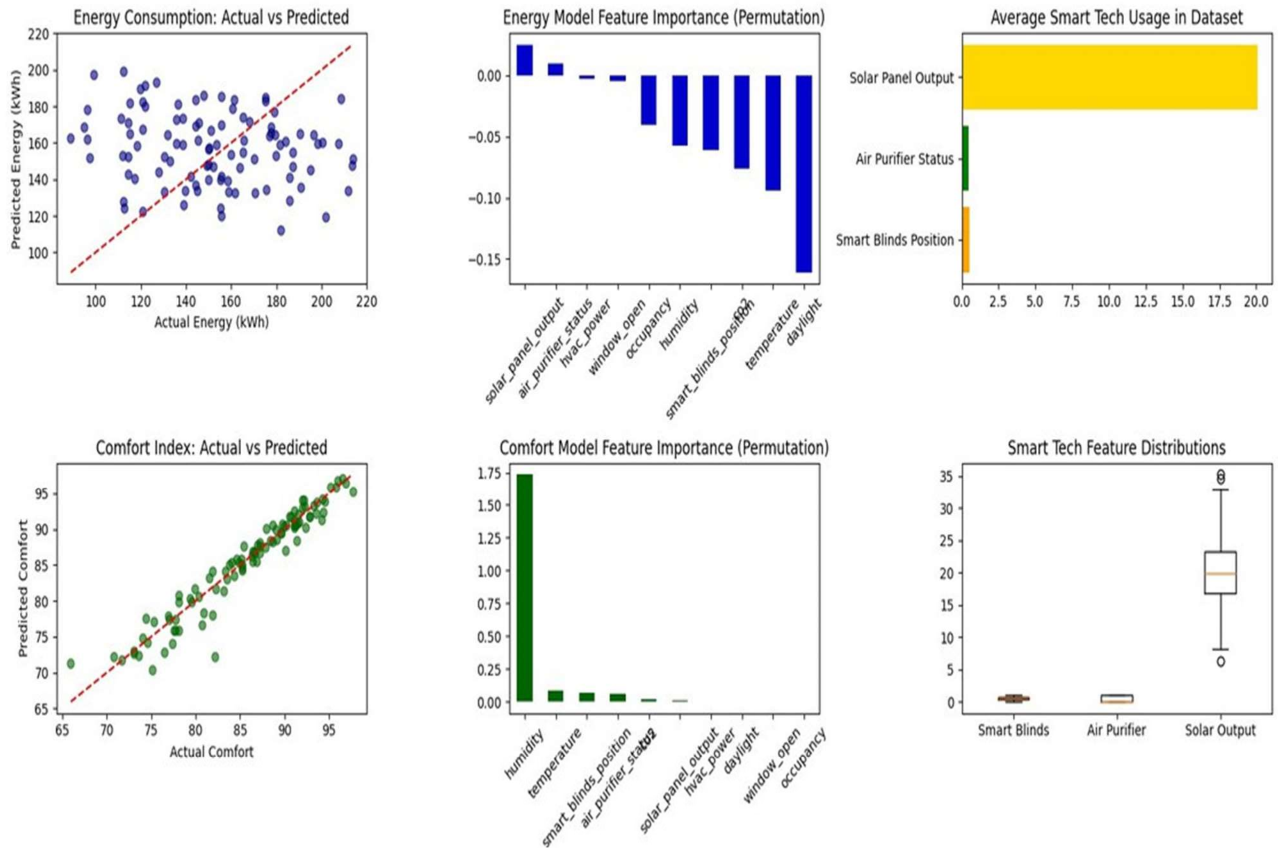
Energy Model Permutation Importances:

```
solar_panel_output        0.025207
air_purifier_status       0.009451
hvac_power               -0.003184
window_open              -0.004860
occupancy                -0.040336
humidity                 -0.057530
smart_blinds_position    -0.060664
co2                      -0.076152
temperature              -0.094257
daylight                 -0.161082
dtype: float64
```

Comfort Model Permutation Importances:

humidity	1.735763
temperature	0.081536
smart_blinds_position	0.069170
air_purifier_status	0.060279
co2	0.015120
solar_panel_output	0.005206
hvac_power	0.004320
daylight	0.003609
window_open	0.001166
occupancy	-0.001058

dtype: float64



GITHUB LINK
https://github.com/Sanjai-s-18/BUILDING-PERFORMANCE-ANALYSIS.git