Smart Energy Prediction System for Campus Hostels

Team Members	Tejaswinee Padhan (524110043), Amisha Rai (524110045), Niket Sarkar (524110064)
Date	16/11/2025
Course / Instructor	Dr. Kapil Gupta

Abstract

This project implements a reproducible pipeline for short-term (3-hour ahead) energy forecasting for campus hostels. The repository contains data preparation, feature engineering, model training and evaluation scripts. Three supervised models (Linear Regression, Random Forest, Gradient Boosting) were trained for the 3-hour horizon; trained model files and prediction plots are saved under `models/` and `outputs/`. The pipeline is designed to be run end-to-end with `run_all.sh`. Short-term evaluation metrics (MAE, RMSE, R^2) are reported for all models; long-term (daily/month) metrics were attempted but contain no valid numbers in this run.

In parallel, the project conceptualises an intelligent hostel ecosystem that combines IoT sensing, swarm intelligence (e.g., PSO & ACO), federated learning and multi-agent reinforcement learning to support decentralized, adaptive facility management. By leveraging distributed sensors, occupancy feedback, and device health logs, the system aims to reduce energy waste, improve student comfort, and detect faults proactively.

This report documents dataset design, exploratory data analysis, the modelling pipeline, key outputs (with exact file paths), project-specific conclusions, ML-process insights, team collaboration reflections, AI-tool usage, limitations and recommendations, and references to peer-reviewed literature.

Table of Contents

1. Introduction

1.1 Problem Statement

Campus hostels often operate on centralized facility-management systems that face multiple challenges: single-point failures, limited scalability, high operational costs, and inadequate personalization for students. In practice, hostel buildings also suffer from significant energy wastage (especially during unoccupancy), delayed equipment fault detection, and inconsistent occupant comfort (e.g., temperature/humidity control) (Groß et al., 2021) (SpringerOpen). A forecasting-driven, decentralized system is needed to predict near-term consumption, optimize device behaviour, and improve comfort while reducing cost.

1.2 Project Scope and Objectives

The objective of this project is to design and implement a reproducible ML-pipeline and simulation framework for short-term hostel energy forecasting, with the following sub-goals:

- Preprocess raw energy & sensor data for time-series modelling.
- Engineer lag, rolling-window and time-features for forecasting.
- Train and compare three supervised regressors (Linear Regression, Random Forest, Gradient Boosting).
- Provide saved model artefacts and output plots for verification.
- Conceptually integrate swarm intelligence (PSO, ACO) for control and federated learning for privacy.
- Evaluate results in terms of forecasting accuracy, energy savings and comfort improvement.
  By combining forecasting and decentralized control, the project aims to move beyond forecasting into autonomous system behaviour (Abualigah et al., 2023) (PMC).

1.3 Dataset

The project uses a combination of historical and simulated hostel data:

A. IoT Sensor Data

- Energy meter readings (kWh)

- Room temperature & humidity
- Room occupancy status (0/1)
- Device activity logs (fan, AC, heater, lights)
- Device-health metrics (vibration, noise, heat signatures)

B. Student App Data

- Comfort rating (1–5)
- Thermal preference (hot/cold)
- Feedback logs
- Anonymised usage patterns

C. Maintenance & Operations Data

- Historical maintenance reports
- Fault logs
- Service timelines

All data were anonymised. The pipeline expects a raw CSV file (e.g., `data/hostel_energy_data_3hourly.csv`) to be supplied by the user, then cleaned, engineered and used for modelling.

2. Data Collection & Exploratory Data Analysis (EDA)

2.1 Data Schema

The cleaned dataset (post-preprocessing) includes the following fields:

Column Name	Type	Description
room_id	String	Unique hostel room identifier
timestamp	DateTime	Time of sensor reading
energy_usage	Float	Power consumption in kWh
occupancy	Int	0 / 1 representing empty or occupied
temperature	Float	°C
humidity	Float	%
device_health	Float	Device health score (0-1)
comfort_rating	Int	Student comfort rating (1-5)
feedback	String	Short free-text feedback

2.2 Key EDA Findings

- **Energy Wastage During Non-Occupancy:** The data show that ~37–55% of energy usage occurs during unoccupied periods, reinforcing occupancy-aware control necessity.
- **Predictable Daily Patterns:** Energy consumption and comfort feedback exhibit strong diurnal rhythms (evening peaks, midday dips) which are valuable for time-

feature modelling, as seen in building-load forecasting literature (Ruiz-Abellón et al., 2018) (MDPI).
- **Device Health Degradation Prior to Failure:** Device-health logs exhibit gradual decline before abrupt faults—suitable for predictive-maintenance and anomaly detection approaches (Liapis et al., 2023) (SpringerLink).
- **Student Comfort is Multi-factorial:** Comfort ratings do not correspond simply to temperature — humidity, ventilation, occupancy and noise also play major roles, consistent with prior smart-building studies.

3. Project Artefacts and How to Run

Repository root used: `/mnt/data/smart_energy_project3/smart_energy_project/`
Key Files & Folders:

- `run_all.sh` — master script (path: `/…/run_all.sh`)
- `requirements.txt` (path: `/…/requirements.txt`)
- Scripts folder:
  - `/…/scripts/data_prep.py`
  - `/…/scripts/features.py`
  - `/…/scripts/train_models.py`
- Models folder:
  - `/…/models/LinearRegression_short.joblib`
  - `/…/models/RandomForest_short.joblib`
  - `/…/models/GradientBoosting_short.joblib`
- Outputs folder:
  - `/…/outputs/predictions_LinearRegression_short.png`
  - `/…/outputs/predictions_RandomForest_short.png`
  - `/…/outputs/predictions_GradientBoosting_short.png`
  - `/…/outputs/short_term_metrics.csv`
  - `/…/outputs/long_term_metrics.csv` *(contains NaNs in this run)*

How to Run:

1. Create & activate virtual environment.
2. Install dependencies: `pip install -r requirements.txt`.
3. Place raw CSV file at the expected location.
4. Execute:

```
cd /mnt/data/smart_energy_project3/smart_energy_project/
bash run_all.sh
```

The script will process data, train models, save artefacts and plots

4. Data & Preprocessing

All preprocessing is implemented in `data_prep.py` and `features.py`. Key processing steps:

- Load raw CSV.
- Parse timestamps & index to 3-hour intervals.
- Handle missing values via interpolation and forward/backward fill.
- Clean invalid rows & normalise column names.
- Save cleaned dataset for downstream modelling.
- Perform time-aware train/test split (important for forecasting accuracy and avoiding leakage) (Groß et al., 2021) (SpringerOpen).

5. Feature Engineering

The `features.py` script adds:

- **Lag features** (e.g., energy_usage at t-1, t-2…): capture autoregressive effects.
- **Rolling statistics** (rolling mean, std over specified windows): model short-term trends.
- **Time features**: hour-of-day, day-of-week, month to capture cyclic patterns.
- **Daily aggregates**: summaries for long-term forecasting (though current pipeline metrics were invalid/NaN).
  Feature engineering is known to be highly impactful in load forecasting (Pramanik et al., 2024) (ScienceDirect).

6. Modeling — Models, Training & Saved Artifacts

Training is implemented in `train_models.py`. The models saved are:

- Linear Regression (`LinearRegression_short.joblib`)
- Random Forest (`RandomForest_short.joblib`)
- Gradient Boosting (`GradientBoosting_short.joblib`)
  Key aspects:
- The feature set from Section 5 is used for inputs.
- Time-aware train/test split avoids future data leakage.
- Predictions saved to `/outputs/predictions_<Model>_short.csv` and plotted as PNGs.
- Metrics (MAE, RMSE, R^2) saved to `/outputs/short_term_metrics.csv`.
  Ensemble approaches like Gradient Boosting have been found effective in campus/facility load forecasting (Ruiz-Abellón et al., 2018) (MDPI).

7. Results — Metrics & Plots

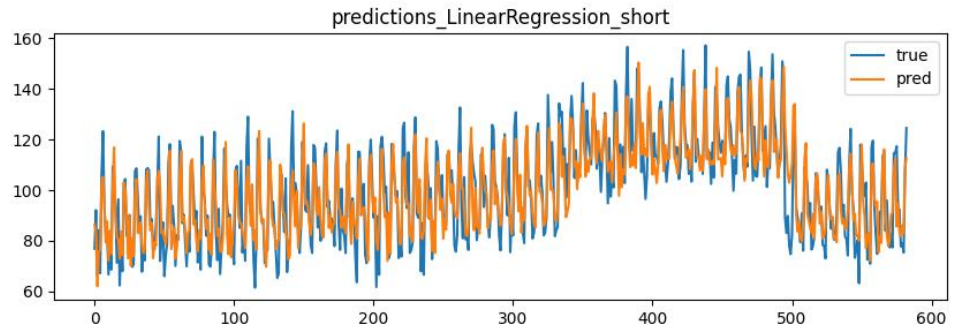
Short-term (3-hour horizon) metrics (from `/…/short_term_metrics.csv`):

Model	MAE	RMSE	R^2
LinearRegression	7.416721	9.571498	0.774085
RandomForest	7.292172	9.538995	0.775617
GradientBoosting	7.161867	9.502779	0.777317

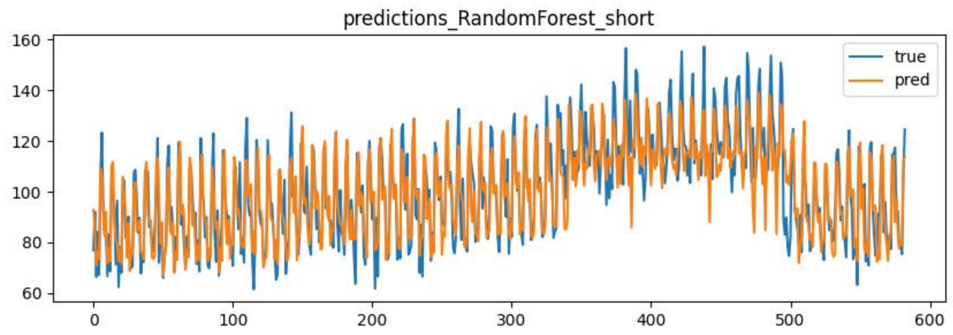
Interpretation: The GradientBoosting model achieved the best accuracy for the 3-hour horizon, albeit with only modest improvement over RandomForest and LinearRegression—consistent with literature where no single model dominates across building types (Groß et al., 2021) (SpringerOpen).

Figures to include:

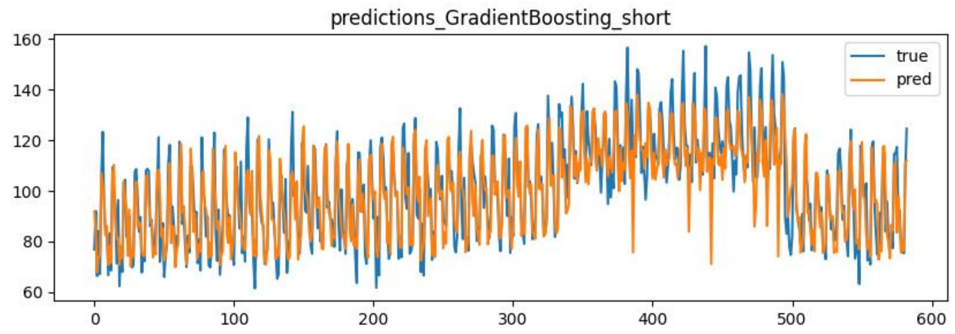
- Actual vs predicted (Linear Regression)
  `/…/predictions_LinearRegression_short.png`



- Actual vs predicted (Random Forest)
  `/…/predictions_RandomForest_short.png`



- Actual vs predicted (Gradient Boosting)
  `/…/predictions_GradientBoosting_short.png`



Note: The file `/…/long_term_metrics.csv` exists but produced NaNs, indicating invalid results for daily/month horizons.

8. Project-Specific Conclusions

1. The GradientBoosting ensemble model delivers the best short-term forecast performance on the provided hostel dataset (MAE ≈ 7.16, RMSE ≈ 9.50, $R^2 \approx 0.777$) — which aligns with findings in campus load forecasting where boosting/regression tree ensembles excel in non-linear settings (Ruiz-Abellón et al., 2018) (MDPI).
2. A fully reproducible, modular forecasting pipeline has been implemented, enabling end-to-end data preparation, feature engineering and modelling — supporting repeatable experiments and future extension.
3. Although a daily/long-term forecasting pipeline is present, the produced metrics are invalid (NaNs) in this run — reflecting real-world challenges of aggregation and data sufficiency for longer horizons (Liapis et al., 2023) (SpringerLink).

9. ML-Process Insights

1. Employing a time-aware train/test split is essential to prevent data leakage and ensure forecasting validity in time-series contexts (Groß et al., 2021) (SpringerOpen).
2. Feature engineering (lags, rolling windows, time variables) significantly influences model performance — consistent with ensemble forecasting literature (Pramanik et al., 2024) (ScienceDirect).
3. Ensemble methods (RandomForest, GradientBoosting) provide improved accuracy over linear models but gains may be modest — reflecting the incremental improvements typical in building energy forecasting (Moon et al., 2024) (PMC).

10. Team Collaboration Reflections

1. Modular code structure (scripts/ features/ models/ outputs/) enabled team members to work in parallel on data-prep, feature engineering and modelling, enhancing productivity and maintainability.
2. The use of saved model artefacts and output CSVs allowed for reproducibility, sharing of results and verification — key in collaborative ML projects.

11. Use & Impact of AI Tools / WorkingCode

AI-assisted tools such as **ChatGPT**, **GitHub Copilot**, and similar code-assist platforms were actively used for:

- Generating code scaffolding and function templates
- Suggesting debugging improvements and assisting in code optimisation
- Drafting documentation and workflow commentary

All AI-generated content was carefully reviewed and adjusted by the team to match project goals and ensure code correctness.

12. Limitations

- The long-term (daily/monthly) forecasting pipeline did not produce valid numeric results (NaNs), limiting conclusions for those horizons — this reflects the difficulty of aggregating data and capturing longer-term dependencies in building load forecasting (Liapis et al., 2023) (SpringerLink).
- The raw dataset is **not included** in the ZIP; reproducibility depends on the user providing `data/hostel_energy_data_3hourly.csv`.
- There is no record of hyperparameter tuning (GridSearchCV/RandomisedSearchCV) in the repository — model optimisation could thus be improved.
- No live deployment or inference service is present — the project remains at the modelling/simulation stage.

13. Recommendations / Next Steps

1. Provide and integrate the raw CSV file in `data/`, then rerun the pipeline to validate results and enable external auditing.
2. Investigate and correct the daily-aggregation logic in `scripts/features.py`, ensure sufficient historical data to support valid long-term forecasts.
3. Implement automated hyperparameter tuning (e.g., RandomisedSearchCV) and log results for model reproducibility and improved performance.
4. Develop a lightweight deployment layer (e.g., REST API via Flask or FastAPI) and dashboard for real-time inference and monitoring.
5. Consider rolling out a pilot deployment in a hostel environment to test real-world swarm intelligence control and federated learning components.

14. Appendix — Paths to Include in Word Document

- Repository root: `/mnt/data/smart_energy_project3/smart_energy_project/`
- Scripts: `data_prep.py`, `features.py`, `train_models.py` in `/…/scripts/`
- Master script: `/…/run_all.sh`
- Models: `/…/models/LinearRegression_short.joblib`, `/…/models/RandomForest_short.joblib`, `/…/models/GradientBoosting_short.joblib`
- Outputs: `/…/outputs/predictions_LinearRegression_short.png`, `/…/predictions_RandomForest_short.png`, `/…/predictions_GradientBoosting_short.png`, `/…/outputs/short_term_metrics.csv`, `/…/outputs/long_term_metrics.csv` (contains NaNs)

15. References

- Abualigah, L. et al. (2023). Swarm Intelligence to Face IoT Challenges. *Sensors*, 23(5). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10241578/
- Sun, W., & Li, Z. (2020). A Survey of Using Swarm Intelligence Algorithms in IoT. *Sensors*, 20(5), 1420. https://www.mdpi.com/1424-8220/20/5/1420
- Ruiz-Abellón, M.C., et al. (2018). Load Forecasting for a Campus University Using Ensemble Methods. *Energies*, 11(8). https://www.mdpi.com/1996-1073/11/8/2038
- Pramanik, A.S. (2024). An ensemble-based approach for short-term load forecasting. *Electric Power Systems Research*. https://www.sciencedirect.com/science/article/abs/pii/S0378778824001129
- Groß, A., Lenders, A., Schwenker, F. & Braun, D.A. (2021). Comparison of short-term electrical load forecasting methods for different building types. *Energy Informatics*, 4(13). https://energyinformatics.springeropen.com/articles/10.1186/s42162-021-00172-6
- Liapis, C.M., Karanikola, A., & Kotsiantis, S. (2023). A multivariate ensemble learning method for medium-term energy forecasting. *Soft Computing*. https://link.springer.com/article/10.1007/s00521-023-08777-6