# **GridOS: An AI-Powered Smart Energy Management System for Microgrids**

**Technical Report** *VidyutAI Hackathon 2025*

**Team Name:** ZeroEmission

**Abstract**

Synapse Energy is a cloud-native, AI-powered Energy Management System (EMS) designed to address the challenges of grid instability in India, driven by the rise of renewable energy and electric vehicle (EV) adoption. Our platform provides real-time monitoring, intelligent diagnostics, and adaptive energy dispatch to enhance grid reliability and minimize operational costs. The core of our solution is a Reinforcement Learning (RL) agent trained to perform cost-optimal energy dispatch, strategically managing solar/wind generation, battery storage, and grid power. This is complemented by a real-time diagnostics module that monitors EV battery health, detects charging anomalies, and generates actionable alerts. The entire system is visualized through a unified dashboard, providing operators with complete end-to-end visibility and control. Our prototype demonstrates a significant potential for cost reduction and proactive fault management in modern microgrids.

## **1. Introduction**

### **1.1. Problem Statement**

India's rapid transition towards renewable energy and the electrification of transport presents significant challenges for microgrid management. The intermittency of solar and wind power, combined with the high, dynamic loads from EV charging, can lead to voltage fluctuations, frequency instability, and unplanned downtime. Traditional EMS platforms lack the foresight and real-time responsiveness to manage these complex, dynamic systems, forcing operators into a reactive, inefficient maintenance cycle.

### **1.2. Objective**

Our objective, as outlined by the VidyutAI Hackathon 2025 Problem Statement 2, was to design and build a cloud-enabled Smart EMS. The system needed to provide:

* Real-time monitoring of all subsystems (renewables, storage, grid, EV).
* Intelligent diagnostic insights and subsystem health indices.
* An adaptive scheduling and dispatch engine.
* A unified dashboard for visualization and control.

### **1.3. Our Solution: Synapse Energy**

We developed Synapse Energy, a modular platform that addresses these objectives through two core intelligent components:

* An RL-based **Optimal Dispatch Advisor** that minimizes energy costs.
* A data-driven **Diagnostics & Alerting Module** for proactive fault detection, focusing on EV battery systems.

## **2. System Architecture**

Our solution is designed as a cloud-native application with a clear separation between the backend logic and the frontend user interface.

*[Placeholder for System Architecture Diagram]*

* **Data Ingestion Layer:** For this prototype, data is ingested from static datasets (my\_energy\_data.xlsx, ev\_battery\_charging\_data.csv) to simulate real-time IoT streams. A production system would connect to an MQTT broker or a cloud IoT hub.
* **Backend (Python - Flask/FastAPI):**
  + **API Server:** Exposes RESTful APIs for the frontend to consume data.
  + **RL Inference Engine:** Loads the pre-trained stable-baselines3 model (cost\_agent\_v3.zip) to provide real-time dispatch recommendations via an API endpoint (e.g., /api/dispatch).
  + **Diagnostics Module:** Contains the logic (ev\_diagnostics\_module.py) to process simulated EV sensor data, evaluate rules, and generate alerts.
* **Frontend (Dashboard):**
  + A web-based interface built with [mention your framework, e.g., React, Vue, or HTML/JS].
  + Visualizes real-time data using libraries like Chart.js.
  + Displays the RL agent's dispatch plan, health indices, and a live feed of alerts.

## **3. Datasets & Pre-processing**

### **3.1. Energy Dispatch Training Dataset (my\_energy\_data.xlsx)**

* **Source:** A composite dataset providing a 31-day history of Wind+Solar generation and New demand at a 5-minute interval.
* **Purpose:** Used as the primary input for training the Reinforcement Learning agent.
* **Pre-processing:** The data was read using pandas, and columns were renamed to Generation\_kW and Demand\_kW for consistency. The Time column was used to derive the hour of the day for simulating dynamic grid pricing.

### **3.2. EV Diagnostics Dataset (ev\_battery\_charging\_data.csv)**

* **Source:** A detailed time-series dataset of EV charging sessions.
* **Purpose:** Used to develop and validate the EV Diagnostics & Alerting Module.
* **Key Features Utilized:** SOC (%), Voltage (V), Current (A), Battery Temp (°C), Degradation Rate (%), Charging Cycles.

## **4. Methodology**

### **4.1. Reinforcement Learning for Optimal Dispatch**

To achieve adaptive scheduling, we formulated the energy dispatch problem as an RL task.

* **Algorithm:** We selected the **Proximal Policy Optimization (PPO)** algorithm from the stable-baselines3 library due to its stability and performance.
* **State Space (Agent's Inputs):** A 5-element array providing a snapshot of the system:
  1. Battery State of Charge (kWh)
  2. Renewable Generation (kW)
  3. Load Demand (kW)
  4. Hour of the Day (0-23)
  5. Current Grid Price (₹/kWh)
* **Action Space (Agent's Outputs):** A continuous, 2-element array defining the precise power dispatch:
  1. Power to draw from Battery (kW)
  2. Power to draw from Grid (kW)
* **Reward Function:** The agent's motivation was defined by a multi-objective reward function designed to minimize cost while ensuring reliability and battery health. The final reward was calculated as:  
  *Reward = (Off-Peak Charge Bonus) - (Grid Cost Penalty) - (Reliability Penalty) - (Low SoC Penalty)* We used aggressive weighting on the penalties and bonuses to encourage the agent to learn complex, long-term strategies.

### **4.2. Diagnostics & Alerting Module**

For EV battery health, we implemented a rule-based diagnostic engine.

* **Diagnostic Logic:** We developed specific rules to detect common faults:
  + **Overheating:** IF Battery Temp > 45°C (Li-ion) THEN trigger CRITICAL alert.
  + **Over-Voltage:** IF Voltage > 4.2V AND SOC > 90% THEN trigger WARNING alert.
  + **Charging Anomaly:** IF Charging Mode is 'Fast' AND SOC < 80% AND Current < 15A THEN trigger INFO alert.
* **Health Index Calculation:** We created a simple formula to provide a clear health score (0-100%):  
  *Health Index = 100 - Degradation Rate - (Penalty for high Charging Cycles)*

## **5. Results & Validation**

### **5.1. RL Agent Performance**

The agent was trained for 20 chapters. Its performance was validated using TensorBoard, which showed a clear convergence of the mean reward per episode, indicating that the agent successfully learned a stable and optimal policy.

*[Placeholder for your TensorBoard reward graph screenshot]*

### **5.2. Diagnostic Module Validation**

The diagnostics module was tested by simulating the ev\_battery\_charging\_data.csv stream. The system successfully identified and generated alerts for all pre-defined fault conditions present in the dataset.

--- !!! ALERT(S) TRIGGERED !!! ---

[CRITICAL] Battery Overheating: 45.1°C (Threshold: 45.0°C)

>> Recommendation: Stop charging immediately! Risk of thermal runaway.

----------------------------------

### **5.3. Dashboard Visualization**

The final unified dashboard provides a clear, at-a-glance view of the entire system.

*[Placeholder for a high-quality screenshot of your dashboard]*

## **6. Conclusion & Future Work**

### **6.1. Conclusion**

We have successfully developed a functional prototype of an intelligent Energy Management System. Our solution demonstrates the power of Reinforcement Learning for real-time cost optimization and showcases a data-driven approach to EV battery diagnostics. Synapse Energy successfully meets all the core requirements of the hackathon challenge.

### **6.2. Future Work**

The current prototype lays a strong foundation for a production-ready system. Future enhancements would include:

* **Predictive Maintenance:** Evolving the diagnostic module to predict failures before they occur.
* **Vehicle-to-Grid (V2G) Integration:** Expanding the RL agent's capabilities to allow discharging EV batteries back to the grid, creating new revenue streams.
* **MLOps Pipeline:** Implementing an automated pipeline for continuous retraining and deployment of the RL model.
* **Live Market Integration:** Connecting the system to real-time energy markets to optimize dispatch based on live price signals.

## **7. Appendix**

### **7.1. GitHub Repository**

All project code, datasets, and documentation are available at:

https://github.com/Pratik-Kumar-Dhakar/VidyutAI?tab=readme-ov-file#vidyutai