Smart Home Facing High Electricity Bills

A PROJECT REPORT

Submitted by

Gowripriya R (DL.AI.U4AID24113)
Yaalini R (DL.AI.U4AID24043)
Vepuri Satya Krishna (DL.AI.U4AID24140)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY (B. Tech)
IN ARTIFICIAL INTELLIGENCE AND DATA SCIENCE (AIDS)

Under the guidance of Dr. Vijay Kumar Soni

Submitted to

AMRITA VISHWA VIDYAPEETHAM
AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE
FARIDABAD – 121002

December 2025

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956

SCHOOL OF
ARTIFICIAL INTELLIGENCE
FARIDABAD

BONAFIDE CERTIFICATE

This is to certify that this project report entitled **“Smart Home Facing High Electricity Bills”** is the Bonafide work of **Gowripriya R, Yaalini R, and Vepuri Satya Krishna**, who carried out the project work under my supervision.

Signature

Dr. Vijay Kumar Soni
Assistant Professor
School of AI, Faridabad

AMRITA VISHWA VIDYAPEETHAM
DEEMED TO BE UNIVERSITY UNDER SECTION 3 OF UGC ACT, 1956

SCHOOL OF
ARTIFICIAL INTELLIGENCE
FARIDABAD

DECLARATION BY THE CANDIDATE

I declare that the report entitled **"Smart Home Facing High Electricity Bills"** submitted by me for the degree of Bachelor of Technology is the record of the project work carried out by me under the guidance of **Dr. Vijay Kumar Soni** and this work has not formed the basis for the award of any degree, diploma, associateship, fellowship, title in this or any other University or other similar institution of higher learning.

Gowripriya R (DL.AI.U4AID24113)
Yaalini R (DL.AI.U4AID24043)
Vepuri Satya Krishna (DL.AI.U4AID24140)

ACKNOWLEDGEMENT

This project work would not have been possible without the contribution of many people. It gives me immense pleasure to express my profound gratitude to our honorable Chancellor Sri Mata Amritanandamayi Devi, for her blessings and for being a source of inspiration. I am indebted to extend my gratitude to our Principal, Dr. Lakshmi Mohandas Amrita School of Computing and Engineering, for facilitating us all the facilities and extended support to gain valuable education and learning experience.

I wish to express my sincere gratitude to my supervisor Dr. Vijay Kumar Soni for their personal involvement and constant encouragement during the entire course of this work.

I am grateful to Project Supervisor, Review Panel Members, and the entire faculty of the Department of Computer Science and Engineering, for their constructive criticisms and valuable suggestions which have been a rich source to improve the quality of this work.

Gowripriya R (DL.AI.U4AID24113)
Yaalini R (DL.AI.U4AID24043)
Vepuri Satya Krishna (DL.AI.U4AID24140)

Contents

0.1 Abstract

In modern smart homes, electricity consumption has increased drastically due to the growing number of electrical appliances and their improper usage patterns. Many households operate appliances during peak tariff hours without considering electricity pricing, which leads to excessively high electricity bills. This project proposes a Smart Home Electricity Optimizer that intelligently schedules appliance usage to minimize overall electricity cost while maintaining user comfort.

The system takes appliance details such as power consumption, duration of usage, priority, and category of operation as inputs. Based on these parameters, a priority-based scheduling mechanism is applied using a greedy algorithm combined with Bin Packing constraints. A Max Heap data structure ensures that appliances with higher priority are scheduled first. Appliances are further classified into peak, off-peak, and continuous usage categories to allocate them to appropriate time slots while ensuring total load does not exceed safe limits.

The optimized schedule is stored using a linked list for sequential access and easy display. Java Swing is used to provide a user-friendly graphical interface, enabling dynamic appliance input and validation. The proposed solution demonstrates how data structures and algorithms can be applied effectively to solve real-world energy optimization problems.

0.2 Problem Scenario

Electricity has become a basic necessity in modern households, powering appliances such as air conditioners, washing machines, heaters, refrigerators, and kitchen devices. However, most users operate these appliances without considering electricity tariff variations throughout the day. As a result, multiple high-power appliances may run simultaneously during peak hours, significantly increasing electricity costs.

The lack of an automated scheduling mechanism and priority-based control results in inefficient energy utilization. Users often forget to switch appliances to off-peak hours or are unaware of cost-effective usage patterns. This uncontrolled appliance operation leads to financial strain and unnecessary energy wastage. Hence, there is a need for an intelligent system that can automatically schedule appliances in a cost-efficient manner.

0.3 Objectives of the System

The main goal of this project is to design and implement an optimized appliance scheduling system for smart homes. The specific objectives include:

- Reducing electricity cost by shifting appliance usage to cheaper time slots.
- Scheduling appliances based on priority, power consumption, and duration.
- Implementing a Greedy Bin Packing optimization strategy using efficient data structures.
- Providing a simple and interactive graphical user interface.
- Ensuring scalability for future extensions and enhancements.

0.4 System Requirements / Functional Goals

The system is designed to meet the following functional requirements:

- Accept appliance details including name, power rating, usage duration, and priority.
- Categorize appliances into peak, off-peak, and continuous usage types.
- Use a Max Heap to ensure high-priority appliances are scheduled first.
- Enforce a maximum load constraint (e.g., 3000W) per time slot to prevent overload.
- Generate an optimized appliance schedule using a greedy approach.
- Display the final schedule clearly using a linked list structure.
- Provide error handling and validation through the GUI.

0.5 System Architecture & Environment

The system operates within a modeled environment that mimics real-world electricity usage constraints. This architecture consists of three core components:

0.5.1 The Tariff Environment

The day is divided into four distinct tariff slots, representing the Time-of-Use (ToU) pricing model common in smart grids. This model dictates the optimization logic:

- **Slot 1 (Off-Peak):** Late Night (11 PM - 6 AM) - Cost: 5 (Cheapest)
- **Slot 2 (Normal):** Morning (6 AM - 4 PM) - Cost: 8
- **Slot 3 (Evening):** Early Evening (4 PM - 7 PM) - Cost: 12
- **Slot 4 (Peak):** Prime Time (7 PM - 10 PM) - Cost: 20 (Most Expensive)

0.5.2 Capacity Constraints

To ensure the safety of the household electrical system, the architecture enforces a strict **Maximum Load Limit of 3000 Watts** per time slot. If adding an appliance to a slot exceeds this total capacity, the system must either find an alternative slot or skip the appliance to prevent a circuit overload.

0.5.3 Appliance Modeling

Each appliance is modeled as a distinct object with attributes for Power (Watts), Duration (Hours), Priority (1-10), and Category (Continuous, Peak, Off-Peak). This structured modeling allows the algorithms to make precise decisions regarding cost and placement.

0.6 Data Structures Used

Efficient data structures are crucial for implementing the scheduling logic. The following data structures are used in the system:

- **ArrayList:** Used to dynamically store appliance input components generated through the graphical interface. It allows flexible insertion and removal of appliance data.
- **Max Heap:** Used to store appliances based on their priority. The appliance with the highest priority is always extracted first, ensuring optimal scheduling decisions.
- **Linked List:** Used to store the final optimized schedule in sequential order. This structure allows efficient traversal and easy display of results.

0.7 Algorithms Used

The core scheduling logic relies on two primary algorithmic concepts:

0.7.1 1. Binary Max-Heap (Priority Queue)

A Binary Max-Heap is used to prioritize appliances. In this structure, the parent node is always greater than or equal to its children.

- **Why it is used:** Simple sorting takes $O(N \log N)$ time, but a Heap allows us to continuously extract the maximum priority element in $O(1)$ time after an initial build time of $O(N)$.
- **Function:** This ensures that critical appliances (Priority 9-10) are processed and scheduled *before* low-priority ones (Priority 1-2).

0.7.2 2. Greedy Strategy with Bin Packing (Capacity Constraint)

The system employs a Greedy Algorithm combined with a Bin Packing heuristic to assign slots.

- **Greedy Logic:** The algorithm greedily attempts to assign the appliance to the *cheapest feasible slot* based on its category. For example, an "Off-Peak" appliance will first try to enter Slot 1 (5).
- **Bin Packing (Constraint Check):** Before finalizing the assignment, the system performs a capacity check similar to the Bin Packing problem. It checks if:

$$\text{CurrentLoad}_{\text{slot}} + \text{AppliancePower} \leq \text{MAX\_LIMIT (3000W)}$$

- If the "Bin" (Time Slot) is full, the item is rejected or skipped. This prevents the Greedy algorithm from overloading a single time slot, effectively balancing the load across the schedule.

0.8 System Workflow

The workflow of the system is a multi-stage process that transforms raw user input into a validated, optimized schedule.

1. **Input Phase:** The user interacts with the GUI to add appliances dynamically. Each entry captures the Name, Power (W), Priority (1-10), Duration, and Category.
2. **Validation Modeling:** The system reads the inputs, converts them into 'Appliance' objects, and validates that power and duration are positive integers.
3. **Prioritization (Heap Construction):** All valid appliance objects are inserted into the Max Heap. The Heap automatically reorders them so the most urgent tasks are at the root.
4. **Optimization Loop (The Logic Core):** The system enters a loop to process appliances one by one until the Heap is empty.

5. **Core Logic of Slot Allocation:** For each extracted appliance, the system applies the following decision logic to determine the "Target Slot":

   - **Case A: Continuous (e.g., Fridge)**
     - *Logic:* Must run 24/7.
     - *Action:* The system checks capacity across ALL slots (1, 2, 3, and 4). If it fits in all, it is assigned.
   - **Case B: Off-Peak (e.g., Washing Machine)**
     - *Low Priority (≤ 5):* User is flexible. Target = **Slot 1 (Off-Peak, 5)**.
     - *High Priority (> 5):* User needs it somewhat soon. Target = **Slot 2 (Normal, 8)**.
   - **Case C: Peak (e.g., AC, Heater)**
     - *Low Priority (< 8):* User can wait slightly to save money. Target = **Slot 3 (Evening, 12)**.
     - *High Priority (≥ 8):* User needs it immediately. Target = **Slot 4 (Peak, 20)**.

6. **Capacity Check Assignment:** Once the Target Slot is identified, the system checks if adding the appliance's power exceeds the 3000W limit for that slot. If within limits, the assignment is confirmed; otherwise, it is skipped.

7. **Cost Calculation:** Upon successful assignment, the system calculates the financial cost for that specific appliance using the formula:

$$\text{Cost} = \left(\frac{\text{Power (Watts)}}{1000}\right) \times \text{Duration (Hours)} \times \text{SlotRate ()} \tag{1}$$

8. **Output Generation:** The results (assigned slot, calculated cost, or skipped status) are stored in a Linked List node and rendered onto the Text Area for the user to see.

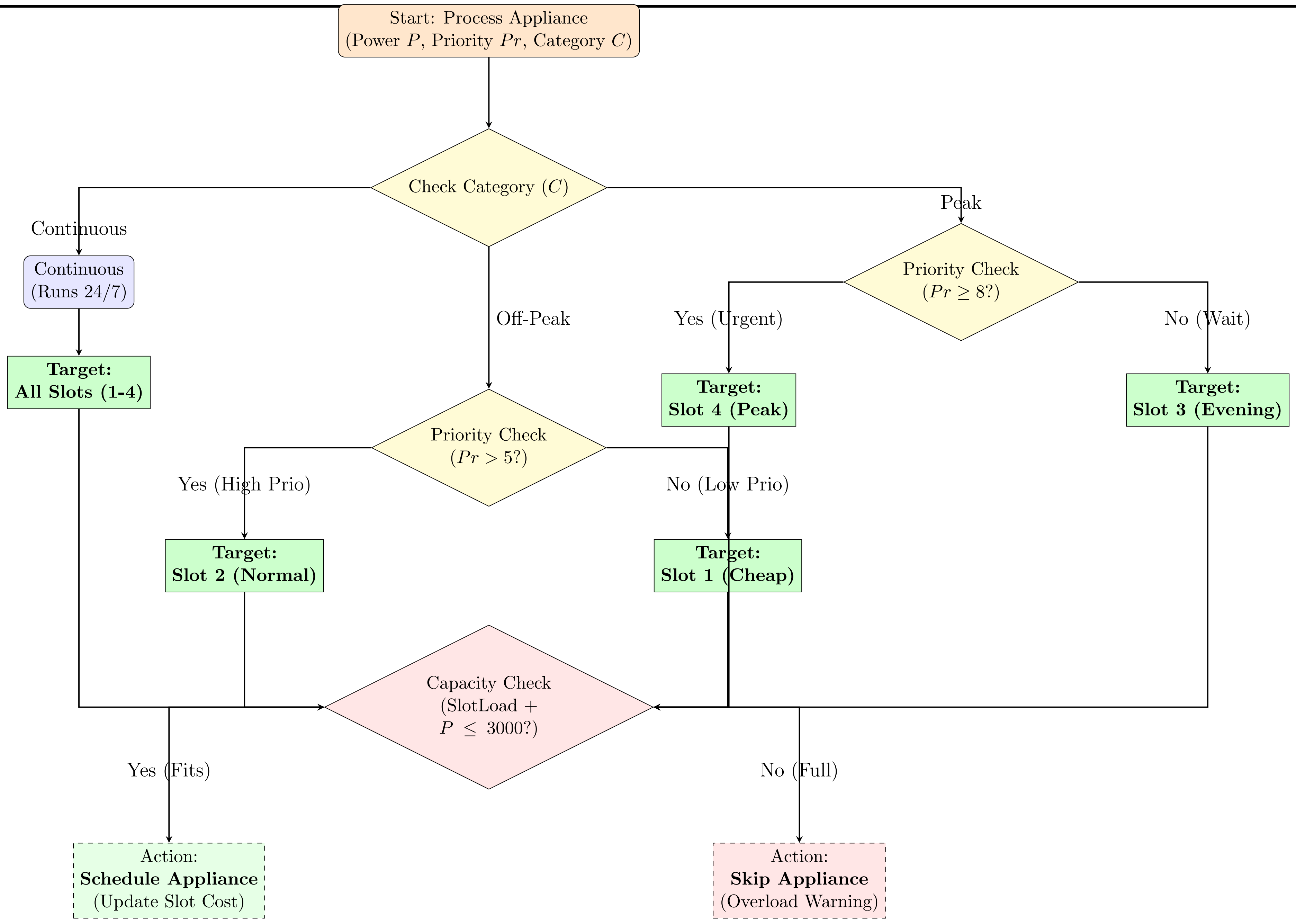
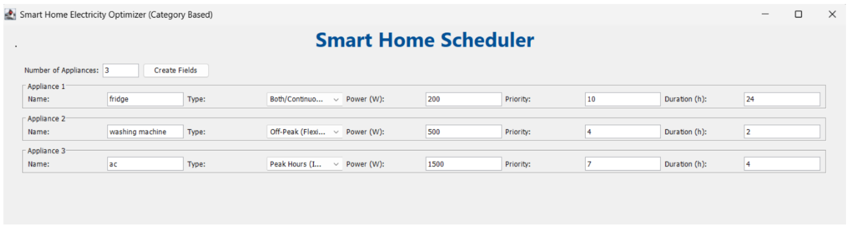


Figure 1: Decision Logic for Slot Allocation & Pricing

0.9 Code Implementation

The system is implemented using Java programming language. Java Swing is used to design the graphical user interface. Separate classes handle appliance modeling, heap operations, linked list scheduling, and user interaction. This modular approach improves code readability, maintainability, and scalability.



Smart Home Electricity Optimizer (Category Based)

Smart Home Scheduler

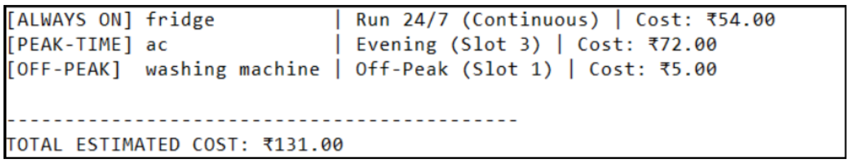
Number of Appliances: 3 [Create Fields]

Appliance 1	Name:	fridge	Type:	Both/Continuo...	Power (W):	200	Priority:	10	Duration (h): 24
Appliance 2	Name:	washing machine	Type:	Off-Peak (Flexi...	Power (W):	500	Priority:	4	Duration (h): 2
Appliance 3	Name:	ac	Type:	Peak Hours (I...	Power (W):	1500	Priority:	7	Duration (h): 4

Figure 2: Graphical User Interface

0.10 Results and Output

The implemented system successfully schedules appliances in a cost-efficient manner. High-priority appliances are allocated appropriate time slots without violating constraints. The optimized schedule reduces peak-hour usage and improves overall energy efficiency.



```
[ALWAYS ON] fridge          | Run 24/7 (Continuous) | Cost: ₹54.00
[PEAK-TIME] ac              | Evening (Slot 3) | Cost: ₹72.00
[OFF-PEAK]  washing machine | Off-Peak (Slot 1) | Cost: ₹5.00

--------------------------------------------
TOTAL ESTIMATED COST: ₹131.00
```

Figure 3: Optimized Appliance Schedule

0.11 Conclusion

This project successfully demonstrates a robust and effective approach to mitigating high household electricity costs through the strategic application of Advanced Data Structures and Algorithms. By shifting away from random appliance usage and adopting a priority-based scheduling model, the system achieves a significant reduction in peak-load consumption and overall expenditure. The core of this optimization lies in the synergy between the Max Heap, which ensures that high-priority tasks are never delayed, and the Greedy Algorithm with Bin Packing constraints, which makes locally optimal choices for time-slot allocation while ensuring circuit safety.

Furthermore, the implementation highlights the practical utility of Linked Lists for maintaining an organized, sequential schedule and ArrayLists for flexible user-input management. Beyond the technical execution, the project proves that even complex real-world problems, such as energy management in a smart home, can be solved with clarity and performance when built upon a foundation of structured data and logical efficiency. Ultimately, this system serves as a scalable model for future energy-saving technologies, providing a clear path toward more sustainable and cost-effective domestic power consumption.

0.12 Future Scope

The system can be enhanced further in several ways:

- Integration with real-time electricity pricing APIs.
- Automatic rescheduling when appliance usage changes dynamically.
- Comparison with advanced optimization techniques such as Dynamic Programming or Genetic Algorithms.
- IoT-based real-time appliance control and monitoring.

0.13 References

[1] T. H. Cormen, et al., *Introduction to Algorithms*, 3rd ed., MIT Press, 2009.

[2] M. A. Weiss, *Data Structures and Algorithm Analysis in Java*, 3rd ed., 2012.

[3] H. Schildt, *Java: The Complete Reference*, 11th ed., 2018.

[4] J. Kleinberg and E. Tardos, *Algorithm Design*, Pearson, 2006.

[5] Oracle, *The Java Tutorials – Swing*, [Online].

[6] IEA, *Energy Efficiency Indicators*, 2020.

Project Source Code

The complete source code for this project is available at:
`https://github.com/SatyaKrishna2811/Adsaa_EndTerm_project/blob/main/adsaEndterm1.java`