RV College of Engineering®
Mysore Road, RV Vidyaniketan Post,
Bengaluru - 560059, Karnataka, India

Go, change the world®

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Project Report

On

Sustainable electronic goods Recommendation based on Carbon Footprint Insights

Submitted in partial fulfilment of the requirements for the V Semester
ARTIFICIAL NEURAL NETWORK AND DEEP LEARNING
AI253IA

By

1RV22AI007	**Ananth M Athreya**
1RV22AI017	**Gnyan Malliah**
1RV22AI037	**Parth Shukla**

Department of Artificial Intelligence and Machine Learning
RV College of Engineering®
Bengaluru – 560059

January 2025

RV COLLEGE OF ENGINEERING®

(Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Bengaluru– 560059

CERTIFICATE

This is to certify that the project entitled “Sustainable electronic goods Recommendation based on Carbon Footprint Insights” submitted in partial fulfillment of Artificial Neural Networks and Deep Learning (AI253IA) of V Semester BE is a result of the bonafide work carried out by Ananth M Athreya(1RV22AI007), Gnyan Malliah(1RV22AI017) and Parth Shukla (1RV22AI037) during the Academic year 2024-25

Faculty In charge

Date :

Head of the Department

Date :

RV COLLEGE OF ENGINEERING®

(Autonomous Institution Affiliated to Visvesvaraya Technological University, Belagavi)

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Bengaluru– 560059

DECLARATION

We, Ananth M Athreya (1RV22AI007), Gnyan Malliah(1RV22AI017) and Parth Shukla(1RV22AI37), students of Fifth Semester BE hereby declare that the Project titled **"Sustainable electronic goods Recommendation based on Carbon Footprint Insights"** has been carried out and completed successfully by us and is our original work.

Date of Submission: **Signature of the Student**

ACKNOWLEDGEMENT

We are profoundly grateful to our guide, **Dr. Somesh Nandi**, Assistant Professor, RV College of Engineering, for his wholehearted support, valuable suggestions, and invaluable advice throughout the duration of our project. His guidance and encouragement were instrumental not only in the successful completion of the project but also in the preparation of this report.We also extend our special thanks to **Dr. Anupama Kumar** for her invaluable insights, support, and constructive feedback, which significantly contributed to the improvement of our work.

We would like to express our sincere thanks to our Head of the Department, **Dr. Satish Babu**, for his constant encouragement and for fostering an environment of innovation and learning that greatly aided our progress.

We extend our heartfelt gratitude to our beloved Principal, **Dr. K. N. Subramanya**, for his unwavering appreciation and support for this Experiential Learning Project, which motivated us to give our best.

Lastly, we take this opportunity to thank our family members and friends for their unconditional support and encouragement throughout the project. Their backup and motivation were crucial in helping us overcome challenges and successfully complete our work.

ABSTRACT

Carbon footprint refers to the total amount of carbon dioxide(CO2) gas emitted directly or indirectly over the life cycle of a product. Carbon footprint is the measure of whether an organization is environment-friendly. Various approaches have been proposed to measure the carbon footprint of products. Most of these approaches prescribe using an LCA-based framework to assess the carbon footprint. LCA refers to Life Cycle Assessment and involves calculating the emissions due to processes of the product over its entire lifecycle. Electronic products contribute significantly to global carbon emissions. The contribution to the carbon emissions is about 4% which is comparable to that of the airline industry having a contribution of 2-4%.

The different components that are used for sustainability recommendation system are Database,Recommendation and optimization model, Chatbot and Website. Since different modules are integrated, object-oriented methodology will be used. SQL Database is used, as the data about the product should be stored in a Structured Database.Proximal Policy Optimization (PPO) is efficient in learning and optimization stuff and therefore we will be using it to build our recommendation and optimization model. Llama 2 LLM is a free LLM that has a good performance. So, Llama 2 will be used to train the chatbot. Streamlit is used for the front-end.

This project is a comparison tool to help choose products with lower carbon impact and cost, both optimised. The user will be provided, by this project, tips for reducing emissions through smarter energy use, disposal, and production choices. He/she will be encouraged to adopt eco-friendly practices.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1: INTRODUCTION

This chapter provides an overview of the Sustainability Recommender System. It includes the theory and concepts used in the project

1.1 Project Description

Sustainability-focused decision-making is crucial as industries and consumers seek to reduce their carbon footprint. However, identifying eco-friendly products and making sustainable choices can be complex. This project develops an AI-powered Sustainability Recommender System, integrating Reinforcement Learning (PPO), LLama-based chatbot interactions, and carbon footprint analysis using LCA datasets to help users make informed, sustainable purchasing decisions.

The core components of our system include:

- Proximal Policy Optimization (PPO) for optimizing personalized recommendations based on sustainability impact.
- LLama-based chatbot for interactive user queries and eco-friendly guidance.
- PostgreSQL database for efficient product and user data storage.
- Life Cycle Assessment (LCA) dataset to calculate the carbon footprint of products.
- Streamlit-based web application for a user-friendly interface.

By leveraging AI and reinforcement learning, this project aims to empower users to make more eco-conscious purchasing decisions, contributing to a greener future.

Theory and Concept

1. Reinforcement Learning for Sustainable Recommendations (PPO):

Reinforcement Learning (RL) allows the system to dynamically optimize product recommendations by maximizing long-term sustainability impact. The PPO (Proximal Policy Optimization) algorithm is used to train the recommendation model efficiently.

Actor-Critic Model:

- The Actor suggests product recommendations based on the current state.
- The Critic evaluates the sustainability impact of each recommendation.

Reward System:

- Products with lower carbon footprints receive higher rewards.
- Diversity in recommendations is encouraged to prevent bias toward specific brands.

Using PPO ensures that our recommendation system continuously improves through iterative learning, balancing user preferences and sustainability metrics.

2. LLama-Based Chatbot for Sustainable Guidance:

A LLama-powered chatbot enhances user engagement by providing:

- Interactive responses to user queries about sustainable products.
- Explanations of sustainability scores based on carbon footprint calculations.
- Personalized recommendations based on user preferences.

This chatbot acts as an intelligent assistant, guiding users toward more sustainable choices through conversational AI.

3. PostgreSQL for Scalable Data Management:

A PostgreSQL database is used to store and manage:

- User preferences and interaction history to personalize recommendations.
- Product sustainability data, including carbon footprint values from the LCA dataset.
- Chatbot conversation logs to improve future interactions.

PostgreSQL ensures the system can scale efficiently while maintaining high-speed data retrieval for recommendations and chatbot interactions.

4. Carbon Footprint Calculation Using LCA Dataset:

The Life Cycle Assessment (LCA) dataset is used to calculate the carbon footprint of products:

- Each product's environmental impact is assessed based on its entire lifecycle (raw materials, production, transportation, usage, and disposal).
- Carbon footprint values are factored into the recommendation system's reward function, prioritizing low-impact products.

By integrating LCA data, our system ensures scientifically backed sustainability scores for transparent and accurate recommendations.

5. Training Process and Performance Metrics:

The model training process follows these key steps:

1. Initialize environment (EcoProductEnvironment) using sustainability data.
2. Train PPO model using the Actor-Critic architecture.
3. Evaluate recommendations using cumulative rewards and sustainability impact metrics.
4. Select the best model based on training performance.
5. Deploy for real-time recommendations.

Performance is assessed using:

- Cumulative reward (measuring long-term sustainability impact).
- Recommendation accuracy (alignment with user preferences).
- Carbon footprint reduction (LCA-based sustainability assessment).
- Diversity score (ensuring a variety of product options).

6. Streamlit-Based Web Interface:

Users interact with the system through a Streamlit-powered web app, which:

- Displays personalized sustainability recommendations.
- Visualizes carbon footprint data from the LCA dataset.
- Provides chatbot-based sustainability guidance.

This interactive UI ensures an engaging and user-friendly experience, making sustainability insights accessible.

7. Model Deployment and Real-Time Predictions:

Once trained, the model is deployed to:

- Provide real-time product recommendations via the Streamlit UI.
- Enable chatbot interactions for eco-friendly guidance.
- Continuously update recommendations based on user interactions and sustainability data.

This ensures the system remains responsive, adaptive, and effective in driving sustainable consumer behavior.

1.2 Report Organization

The report is structured to provide a comprehensive understanding of the carbon footprint assessment for electronic products.

It begins with the Introduction, offering an overview of the project's background, significance, and objectives in analyzing the carbon footprint associated with electronic products. It discusses the role of carbon footprint measurement in evaluating environmental sustainability and highlights the impact of electronic products on global carbon emissions.The Project Description elaborates on the scope of the study, methodologies employed, and expected outcomes, setting the stage for further discussions in the report.

The Report Organization section guides readers through the structure of the report, ensuring clarity and logical progression. Following this, the Literature Review delves into previous research, existing measurement techniques, and proposed innovations. It details the tools and technologies used, including the Life Cycle Assessment (LCA) framework, which is widely adopted for measuring carbon footprints.

The Software Requirement Specifications section describes the functional and non-functional requirements of any software tools used for carbon footprint analysis. It also includes external interfaces and design constraints, providing a clear understanding of the system’s capabilities and limitations.

Next, the System Design segment includes architectural design, data flow diagrams, and a detailed description of the methodologies and algorithms employed to assess the carbon footprint. The focus is on using LCA to evaluate emissions at different stages of an electronic product's life cycle.The Implementation section presents key analysis results, supporting screenshots, and discussions on the accuracy and effectiveness of the carbon footprint measurement approach.

This chapter provides an overview of the carbon footprint assessment project, highlighting its importance in evaluating environmental sustainability and the role of advanced assessment techniques like Life Cycle Assessment for accurate measurement. It also outlines the theory and concepts that underpin the project, setting the stage for detailed discussions in later chapters.

The report concludes with a Conclusion, summarizing the findings and emphasizing the significance of carbon footprint assessment in promoting environmental sustainability. The Future Enhancements section suggests potential improvements, such as the integration of artificial intelligence for more accurate predictions and real-time monitoring.

CHAPTER 2: LITERATURE SURVEY

This chapter provides the various research papers referred for the project. The chapter also provides the summary of the information from these papers, and the research gap.

2.1 Literature Survey

[1] outlines methodologies for integrating digital wins into lifecycle assessments to achieve a lower carbon footprint, specifically in manufacturing and product lifecycle management. [2] compares various standards for carbon footprint assessments, analyzing quantification methods and emission treatment approaches. [3] reviews existing carbon footprint estimation methods, focusing on the lifecycle approach. It identifies gaps in current methodologies and suggests improvements for greater precision. [4] explores global research trends in carbon footprint tracking. Emphasizes using technology like AI and blockchain for tracking carbon footprint in complex supply chains. [5] analyzes the environmental impacts of integrated circuit (IC) production, emphasizing carbon footprint and energy demand, and advocates for standardized methodologies to reduce the ecological footprint. [6] contains the LCA of 2 electronic devices- smart watch and a TV remote, and this paper contains the different electronic components these devices have. [7] contains the methodology of LCA of a smartphone, also the data of individual components are given. [8] conducts a Life Cycle Assessment (LCA) comparing a smartwatch and a TV remote by examining their electronic components and evaluating their environmental impacts throughout their lifecycle. [9] presents the life cycle assessment (LCA) methodology for smartphones, detailing the environmental impact of individual components. It aims to inform consumers about the ecological footprint of their devices. [10] analyzes the environmental impacts of integrated circuit (IC) production, emphasizing carbon footprint and energy demand, and advocates for standardized methodologies to reduce the ecological footprint.[11] discusses the prevailing methods used by organizations to estimate the CF of their electronics products and identifies the challenges faced by the electronics industry when adopting these methods in an environment of decreasing product development cycles with complex and diffuse supply chains. [12] describes an LCA method for a Chinese personal computer. [13] provides an original calculation of the LCI data for each electronic components industry. [14] compares LCA variation for 3 electronic products. [15] focuses on embodied carbon emissions of user devices associated with accessing networks, hence tablets, smartphones and feature phones, laptop and desktop PCs, PC displays and customer premises equipment (CPEs). [16] is a study that analyses and compares which electronic product contributes more to the global warming, by analysing the carbon footprint of electronic products. [17] compares the GHG emission of 11 electronic products. [18] proposes a Product Carbon Footprint(PCF) of a Printed Circuit Board(PCB). [19] estimates the carbon footprint (CF) of

smartphones and life cycle costs (LCC) for consumers in scenarios where different material efficiency strategies are implemented in Europe. [20] analyses the carbon footprint of gaming console like PlayStation3.

2.2 Summary of the literature survey

From the research papers, we have obtained the following things:

- The methodology of LCA of electronic goods.
- The emission data of individual components from the LCA done for electronic products.
- Usage of Reinforced Learning for recommendation and optimization system.

The research gaps are:

- The research papers only give the methods of LCA of electronic products, and do not specify a framework for storing the information, and using them further.
- The papers do not give us any sort of optimisation of cost and carbon footprint for recommendation.

Objectives

1. Analyse the data from suppliers, logistics, disposal etc. to extract the carbon footprint of electronic products across their lifecycles
2. Optimise cost and carbon footprint of electronic products.
3. Develop a system the recommends sustainable electronic goods

2.3 Existing and Proposed Systems

Existing Systems

The existing systems of product recommendation prioritize maximizing revenue and profit over other factors. They use collaborative and content-based filtering to recommend the most popular products without taking into consideration their environmental impact.

Their key limitations are :

- Ignores Sustainability : No carbon footprint or environmental impact tracking
- Optimizes Sales and Profit
- Encourages upselling and cross-selling
- Doesn’t personalise for eco-conscious people.

Proposed System

This system shifts the approach from profit-driven to sustainability-first approach. It utilize

2.4. Tools and techniques used

1. Reinforcement Learning for Sustainable Recommendations (PPO)

Reinforcement Learning (RL) allows the system to dynamically optimize product recommendations by maximizing long-term sustainability impact. The PPO (Proximal Policy Optimization) algorithm is used to train the recommendation model efficiently.

Actor-Critic Model:

- The Actor suggests product recommendations based on the current state.
- The Critic evaluates the sustainability impact of each recommendation.

Reward System:

- Products with lower carbon footprints receive higher rewards.
- Diversity in recommendations is encouraged to prevent bias toward specific brands.

Using PPO ensures that our recommendation system continuously improves through iterative learning, balancing user preferences and sustainability metrics.

2. LLama-Based Chatbot for Sustainable Guidance

A LLama-powered chatbot enhances user engagement by providing:

- Interactive responses to user queries about sustainable products.
- Explanations of sustainability scores based on carbon footprint calculations.
- Personalized recommendations based on user preferences.

This chatbot acts as an intelligent assistant, guiding users toward more sustainable choices through conversational AI.

3.Streamlit-Based Web Interface

Users interact with the system through a Streamlit-powered web app, which:

- Displays personalized sustainability recommendations.
- Visualizes carbon footprint data from the LCA dataset.
- Provides chatbot-based sustainability guidance.

This interactive UI ensures an engaging and user-friendly experience, making sustainability insights accessible.

4. Training Process and Performance Metrics

The model training process follows these key steps:

1. Initialize environment (EcoProductEnvironment) using sustainability data.
2. Train PPO model using the Actor-Critic architecture.

3. Evaluate recommendations using cumulative rewards and sustainability impact metrics.
4. Select the best model based on training performance.
5. Deploy for real-time recommendations.

Performance is assessed using:

- Cumulative reward (measuring long-term sustainability impact).
- Recommendation accuracy (alignment with user preferences).
- Carbon footprint reduction (LCA-based sustainability assessment).
- Diversity score (ensuring a variety of product options).

2.5 Hardware and Software requirements

Software Requirements

Developer Side:

- OS: Windows 10/11, Ubuntu 20.04+, macOS
- Python 3.8+
- IDE: VS Code, PyCharm, or Jupyter Notebook
- Database: MySQL, PostgreSQL, or SQLite
- LLM Tools: Llama-2, Hugging Face transformers, PyTorch/TensorFlow
- Frontend: streamlit, pandas, numpy, dotenv
- Version Control: Git
- Optional: Docker, langchain

Client Side:

- Modern web browser (Chrome, Firefox, Safari, or Edge)
- JavaScript enabled
- Internet: 5 Mbps minimum connection

Hardware Requirements

1. Developer Side:
    - Processor: Intel i5 (min) / i7 (recommended)
    - RAM: 8GB (min) / 16GB (recommended)
    - Storage: 256GB SSD or 500GB HDD (min) / 512GB SSD or 1TB HDD (recommended)
    - GPU: NVIDIA with CUDA support (for Llama-2 training)
2. Client Side:
    - Any modern device with:
        - Dual-core processor
        - 2GB RAM

- Minimal storage (browser-based)

CHAPTER 3: SOFTWARE REQUIREMENT SPECIFICATIONS

This chapter provides definitions, acronyms, and abbreviations used in the report. Additionally, it presents a general description of the Sustainability Recommender System, including its functional, non-functional, and external interface requirements.

3.1 Introduction

Definitions

- PPO (Proximal Policy Optimization): A reinforcement learning algorithm used for policy optimization, balancing exploration and exploitation to improve recommendation accuracy.
- LLama (Large Language Model Meta AI): An advanced AI chatbot used for conversational interactions, providing sustainability insights and product recommendations.
- PostgreSQL: An open-source relational database management system used to store product data, sustainability scores, and user preferences.
- LCA (Life Cycle Assessment): A methodology used to evaluate the environmental impact of a product throughout its lifecycle, including raw materials, production, transportation, usage, and disposal.
- Streamlit: A Python-based web application framework used to create an interactive UI for users to receive recommendations and interact with the chatbot.

Acronyms

- PPO: Proximal Policy Optimization
- LLama: Large Language Model Meta AI
- LCA: Life Cycle Assessment
- DBMS: Database Management System
- UI: User Interface

Overview

This project focuses on the development of a reinforcement learning-based sustainability recommender system, designed to help consumers make eco-friendly purchasing decisions. It integrates AI-driven recommendation models, conversational AI chatbots, and carbon footprint analysis to guide users toward low-impact products.

By leveraging machine learning, natural language processing, and sustainability data, the system ensures accurate, real-time recommendations while maintaining an intuitive and accessible user experience. To guarantee smooth and efficient execution, specific hardware and software requirements are essential for both training and real-world application.

3.2 General Description

Product Perspective

The Sustainability Recommender System is designed to help consumers make environmentally responsible purchasing decisions by suggesting products based on carbon footprint data and sustainability scores.

The system consists of:

- A reinforcement learning-based recommendation engine that optimizes product suggestions based on sustainability metrics.
- A LLama-powered chatbot that provides interactive sustainability insights and personalized product guidance.
- A PostgreSQL database that stores product attributes, user preferences, and carbon footprint calculations.
- An LCA dataset integration, ensuring that product recommendations are based on real-world environmental impact data.
- A Streamlit-based web application, allowing users to interact with recommendations through a simple UI.

The system is scalable and can be extended to businesses and policymakers looking to integrate sustainability-based decision-making into their workflows.

Product Functions

1.Sustainability-Based Product Recommendations :

- Uses PPO reinforcement learning to optimize recommendations based on user preferences and sustainability factors.
- Suggests low-carbon footprint products to encourage eco-friendly consumption.

2.Interactive Chatbot for Sustainability Guidance :

- LLama-based chatbot answers user queries about product sustainability.
- Provides carbon footprint comparisons between different products.

3.Graphical User Interface :

- Built using Streamlit to ensure a simple and intuitive user experience.

- Displays product recommendations, carbon footprint data, and chatbot interactions in a structured format.

User Characteristics

Primary Users (Consumers):

- Skill Level: General users with no technical expertise.
- Technical Needs: A simple interface for product recommendations and chatbot interactions.
- Goal: To make eco-conscious purchase decisions and reduce personal carbon footprint.

Secondary Users (Sustainability Analysts, Researchers):

- Skill Level: Technically proficient in data analysis and environmental studies.
- Technical Needs: Access to carbon footprint calculations and recommendation system analytics.
- Goal: To evaluate the system's effectiveness in promoting sustainable consumer behavior.

End Users (Retailers, E-Commerce Platforms, Policymakers):

- Skill Level: Advanced users involved in business and sustainability policy.
- Technical Needs: Ability to integrate the system into existing retail/e-commerce platforms for large-scale use.
- Goal: To promote sustainable products, reduce environmental impact, and align business models with eco-friendly standards.

General Constraints

1.Accuracy of Recommendations:

- The system's effectiveness depends on high-quality product sustainability data.
- Errors in LCA dataset calculations may lead to incorrect recommendations.

2.Database Performance & Scalability:

- The PostgreSQL database must efficiently handle large-scale product and user data.
- Queries must be optimized to ensure fast recommendation retrieval.

3.Chatbot Response Accuracy:

- The LLama chatbot relies on pre-trained sustainability knowledge.
- It may require fine-tuning to improve response quality and user engagement.

4.Hardware and Processing Limitations:

- Running PPO training on limited computational resources may increase training time.
- Deploying the chatbot efficiently requires adequate server or cloud resources.

5.Internet Connectivity:

- Users require stable internet access to receive recommendations and interact with the chatbot.
- Offline access is not currently supported for real-time interactions.

Assumptions and Dependencies

1. Availability of Sustainability Data:

- The LCA dataset must be updated regularly to provide accurate carbon footprint insights.
- The system assumes access to a diverse and reliable sustainability dataset.

2.User Access to Web Interface:

- Users need a web-enabled device (laptop, smartphone, tablet) to access the Streamlit-based interface.
- The interface assumes basic digital literacy for navigation.

3.Chatbot Model Optimization:

- The LLama chatbot may require ongoing improvements based on user feedback and new sustainability research.
- Responses will be continuously refined to improve engagement.

4.Integration with Retail Platforms:

- Future expansions may involve API-based integration with e-commerce websites to provide real-time sustainability insights at the point of purchase.

5.Regular Model Training:

- The PPO recommendation model needs periodic retraining as user preferences and sustainability data evolve.
- Training updates will be based on new data collected from user interactions.

3.3 Functional Requirement

1. Introduction

The Carbon Footprint Assessment System is designed to evaluate the environmental impact of electronic products by calculating their carbon footprint using Life Cycle Assessment (LCA) data. The system leverages **Proximal Policy Optimization (PPO)** for optimizing product recommendations based on carbon footprint scores, helping users make sustainable

purchasing decisions. Additionally, a **Llama-based chatbot** is integrated to assist users with queries related to carbon footprint calculations, sustainability practices, and product comparisons.

2. Input

The system requires the following inputs:

- **Product Details:** Users provide electronic product information, including brand, model, and specifications.
- **LCA Data:** Lifecycle Assessment data containing emissions information for different stages of the product's lifecycle (manufacturing, usage, disposal).
- **User Preferences:** Users can specify sustainability priorities (e.g., low energy consumption, eco-friendly materials) to tailor recommendations.
- **Database Inputs:** The system retrieves carbon footprint data from an SQL database containing emission records and sustainability metrics.

3. Processing

The system executes the following processes:

- **Data Preprocessing:**
    - **Data Cleaning:** Filters incomplete or inconsistent product details to ensure accuracy.
    - **Feature Extraction:** Extracts necessary attributes from product specifications, such as power consumption, weight, and materials used.
    - **Normalization:** Standardizes emission values from LCA datasets to ensure consistent calculations.
- **Carbon Footprint Calculation:**
    - **Lifecycle Emission Aggregation:** Computes total emissions across different lifecycle stages (manufacturing, usage, and disposal).
    - **Weighted Impact Analysis:** Assigns appropriate weights to different emission sources (e.g., higher weight for production phase emissions if they contribute significantly to the footprint).
    - **LCA Methodology Implementation:** Applies predefined carbon footprint assessment methodologies to derive an accurate environmental impact score.
- **Recommendation System:**
    - **PPO-Based Optimization:** Utilizes **Proximal Policy Optimization** to analyze sustainability parameters and optimize electronic product recommendations.
    - **Preference Matching:** Considers user-defined priorities (e.g., energy efficiency, recyclability) to refine recommendations.
    - **Comparative Analysis:** Evaluates multiple electronic products to highlight the most sustainable options.
- **Chatbot Assistance:**

- **Natural Language Processing (NLP):** Processes user queries to extract intent and provide meaningful responses.
- **Knowledge Base Integration:** Retrieves relevant data from sustainability databases to enhance chatbot responses.
- **User Guidance:** Provides explanations on carbon footprint calculations, sustainable product choices, and disposal methods.

4. Output

The system provides the following outputs:

- **Carbon Footprint Report:** Displays the calculated emissions for a given electronic product.
- **Sustainability Score:** Rates products based on their environmental impact.
- **Optimized Product Recommendations:** Suggests sustainable alternatives based on PPO optimization.
- **Chatbot Interaction:** Users receive instant responses to queries regarding sustainability and carbon footprint data.

By analyzing and optimizing the carbon footprint of electronic products, this system aids consumers and organizations in making eco-friendly choices and reducing environmental impact.

3.4 External Interfaces Requirements

1. User Interfaces :

The system requires the following user interfaces for interaction with end users and administrators:

- **Product Recommendations**: Users will interact with a simple web interface to receive product recommendations based on their preferences.
- **Input Preferences**: Users can input their preferences (product category, price range, etc.) through text fields or dropdowns.
- **Display Products**: The recommended products will be displayed in a list, with key details such as name, category, and carbon footprint.
- **Search and Filter**: Users can search for products by name and filter recommendations based on factors like price, category, and carbon footprint.

2. Hardware Interface :

The system requires the following hardware components for operation:

- Client Devices: End-user devices such as smartphones, laptops, and desktops with modern browsers (e.g., Chrome, Firefox, Safari, Edge) are necessary to access the chatbot and product recommendation interface. Devices should be capable of running the Streamlit-based frontend smoothly.

- Server (Backend): A server or cloud-based infrastructure is required for hosting the application, running the machine learning models (Llama-2 for chatbot functionalities), and processing the database queries. This can be either a physical machine or a cloud-based solution such as AWS, Google Cloud, or Microsoft Azure. The server will also manage database operations, handle user requests, and serve recommendations.

- Database Server: A reliable database server (e.g., PostgreSQL) is needed for storing product details, carbon footprint data, and user preferences. This should be optimized for fast querying, scaling, and security.

- Storage Device: Adequate storage is required for saving large datasets such as carbon footprint information, user activity logs, and model artifacts (e.g., pre-trained models or recommendation data). This could be either local storage or cloud-based storage depending on the deployment configuration.

3. Software Interface :

The system needs several external software components for proper functionality:

1. Database Management System (DBMS): The system interfaces with SQL-compatible databases (e.g., PostgreSQL, MySQL) for storing and retrieving product data and user preferences. The DBMS should support fast querying, indexing, and scaling.

2. APIs for External Data: LCA (Life Cycle Assessment) APIs: These APIs provide carbon footprint data and sustainability metrics for various products. The system will interface with external APIs to retrieve updated data related to the carbon footprint and other sustainability-related attributes.

3. Machine Learning Tools:
    - Llama-2 (via Hugging Face): The system will interface with the Llama-2 model via the Hugging Face Transformers library for generating natural language responses and handling user queries.
    - Frameworks for Conversational Flow: Optional integration with frameworks like LangChain or other conversational flow builders may be used for improving the chatbot's ability to understand and process user queries.

3.5 Non-Functional Requirements

1. Performance

- The system should process and return carbon footprint calculations and product recommendations within **2-3 seconds** to ensure a smooth user experience.
- The chatbot should provide near-instant responses to user queries to enhance interaction efficiency.

2. Reliability

- The system should be fault-tolerant and capable of recovering from errors, ensuring uninterrupted service.
- Database queries for retrieving LCA data and emissions records should execute efficiently without significant delays.

3. Usability

- The interface should be intuitive and user-friendly, allowing users to input product details and receive recommendations with minimal effort.
- The chatbot should provide clear and contextually relevant responses to sustainability-related queries.

4. Security

- All user inputs, including product details and preferences, should be securely processed and stored.
- HTTPS encryption should be used for secure data transmission.
- The chatbot should ensure privacy and restrict sensitive data sharing.

5. Maintainability

- The system should follow a **modular architecture** to allow easy updates and integration of new sustainability metrics.
- The recommendation model and LCA database should be designed for scalability, allowing the addition of new product categories without major rework.
- Comprehensive documentation should be maintained to facilitate future enhancements.

3.6 Design Constraints

1. Standard Compliance

a. The system should follow best practices in AI and reinforcement learning, ensuring efficient PPO model training, recommendation accuracy, and chatbot responses.

b. The Streamlit-based UI should adhere to accessibility standards (WCAG 2.1), ensuring inclusivity for all users.

c. Data privacy regulations (e.g., GDPR) must be considered when handling user preferences and interaction history, ensuring that sensitive data is managed securely.

d. The PostgreSQL database structure should comply with ACID principles to ensure data consistency, integrity, and reliability.

2. Hardware Limitations

a. The system must be optimized to run on standard consumer devices, including low-power laptops, desktops, and mobile devices, ensuring accessibility.

b. The PPO-based recommendation model should be efficiently trained on local or cloud-based GPUs, while the LLama chatbot should support real-time responses without excessive computational costs.

c. The database queries should be optimized to ensure minimal latency, even on devices with limited processing power.

d. The client-side interface should remain lightweight, ensuring smooth performance on devices with as low as 2 GB RAM and dual-core processors.

CHAPTER 4: SYSTEM DESIGN

4.1 Architectural Design of the Project

1. Problem Specification

This project aims to develop a sustainability tracker that recommends electronic products to users based on the carbon footprint. Traditional recommendation algorithms do not take into account sustainability metrics. We'll be employing a reinforcement-learning based recommendation system using proximal policy optimization for personalized product suggestion while also considering environmental impact.

2. Block Diagram

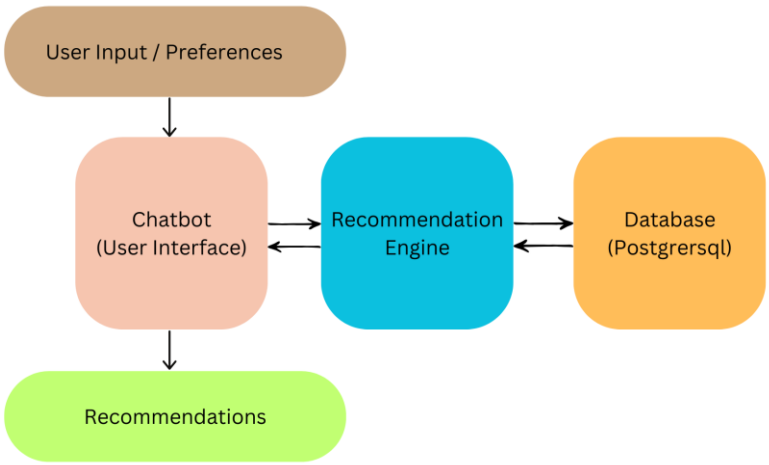

Figure 4.1: Block diagram

3. Module Specification

1. **User Input**
   The user inputs the preferences of the product he/she wishes to buy. It contains specification and category of the product. They can also input their preferred carbon footprint.

2. **Chatbot**
   It is the user interface using which the user interacts. Developed using Llama, an open source Llm developed by Meta. The chatbot is trained by specifying user intents and gradually updating the knowledge base.

3. **Recommendation Engine**
   This module consists of the ML model which is used to predict the recommendations. It takes in input from the Chatbot and fetches the

user history from the database. All these parameters are combined together to make personalized recommendations. The model utilizes the Proximal Policy Optimization(PPO) algorithm, a reinforcement learning algorithm.

4. **Database**
   It consists of all the data about all the recommendable products. The user authentication/identification data is also present. Postgresql is used to develop the database.

5. **Recommendations**
   The output that the user gets from the chatbot. It consists of the products in the range of the user input optimized with sustainability metrics like carbon footprint.

4. Data definition/Dictionary

The database has 11 tables in total, including the mapping tables. The most important tables used for training among these tables were the product, user tables, LCA tables. The data definition for Users Table is as follows :

Column / Feature	**Description**
user_id	Unique identifier for each user
name	Name of the user
password	User's password (Securely stored using hashing)
email	User's email address
phone_number	The phone number of the user
preferences	It consists of a tuple of features of the products that the user wants to buy.

Table 4.1: User table

The data definition for Product Table :

Column Name	**Description**
product_id	The unique identifier for each product
brand	Brand / Manufacturer of the product
model	Specific model of the product
product_category	Category of the product(e.g,.

	Smartphone, Laptop)
price	Price of the product
lca_id	Links the product to its Lifecycle Assessment.

Table 4.2: Product table

The definition for LCA data tables are :

Column Name	**Description**
lca_id	Unique identifier for each LCA record
process_name	Name of the production or manufacturing process
last_update	Date when the data for the LCA record was previously updates
total_carbon_emissions	Total carbon emission associated with the process.

Table 4.3: LCA Data Table

4.2 Data Flow Diagram

Level 0 DFD:

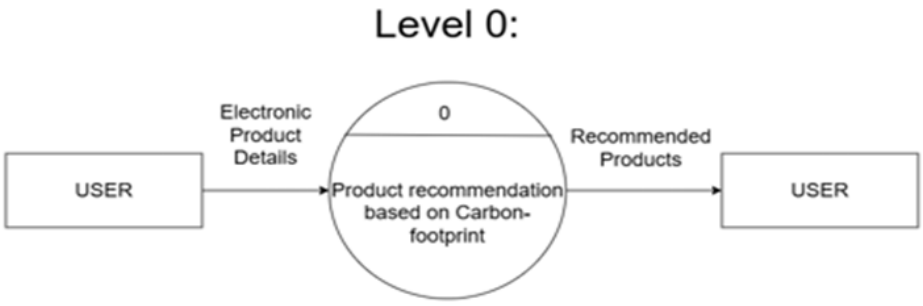

Figure 4.2: DFD 0

1. Main Process (Process 0):
    - Product recommendation based on Carbon-footprint
    - Processes details and generates eco-friendly recommendations
2. External Entities:
    - Input USER:
        - Provides product details and preferences
        - Submits search criteria
    - Output USER:

- Receives recommended products
- Gets carbon footprint results

3. Data Flows:
    - Input Flow:
        - User preferences
        - Search criteria
        - Category and price preferences
    - Output Flow:
        - Product recommendations
        - Carbon footprint scores
        - Product specification

Level 1 DFD :

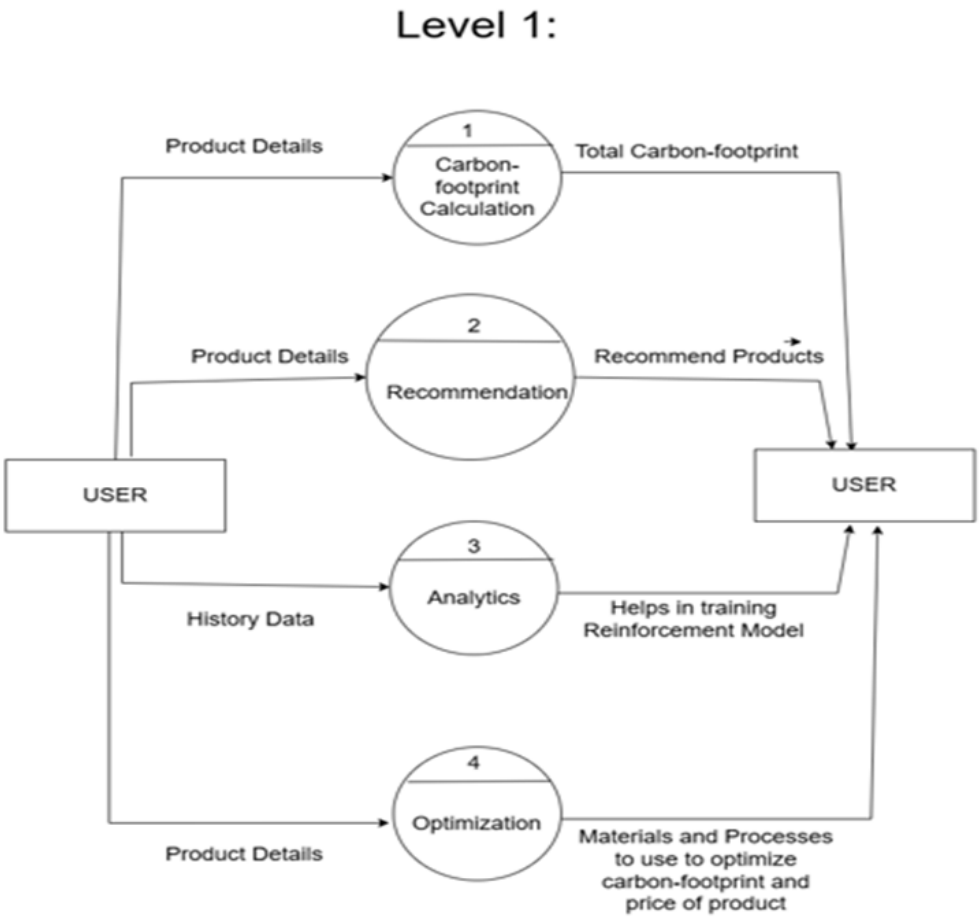

Figure 4.3: DFD 1

1. Process 1: Carbon-footprint Calculation
    - Input: Product Details
    - Output: Total Carbon-footprint
    - Function: Calculates the carbon footprint for products
2. Process 2: Recommendation
    - Inputs: Product Details

- Output: Recommend Products
- Function: Generates product recommendations

3. Process 3: Analytics
   - Input: History Data
   - Output: Helps in training Reinforcement Model
   - Function: Analyzes user interaction data for model improvement
4. Process 4: Optimization
   - Input: Product Details
   - Output: Materials and Processes to optimize carbon-footprint and price of product
   - Function: Suggests optimizations for better sustainability

4.3 Description of the PPO Architecture

Proximal Policy Optimization (PPO) is a reinforcement learning algorithm designed to optimize policies in dynamic environments by balancing exploration and exploitation. It is widely used for training agents in both continuous and discrete action spaces due to its stability and efficiency.

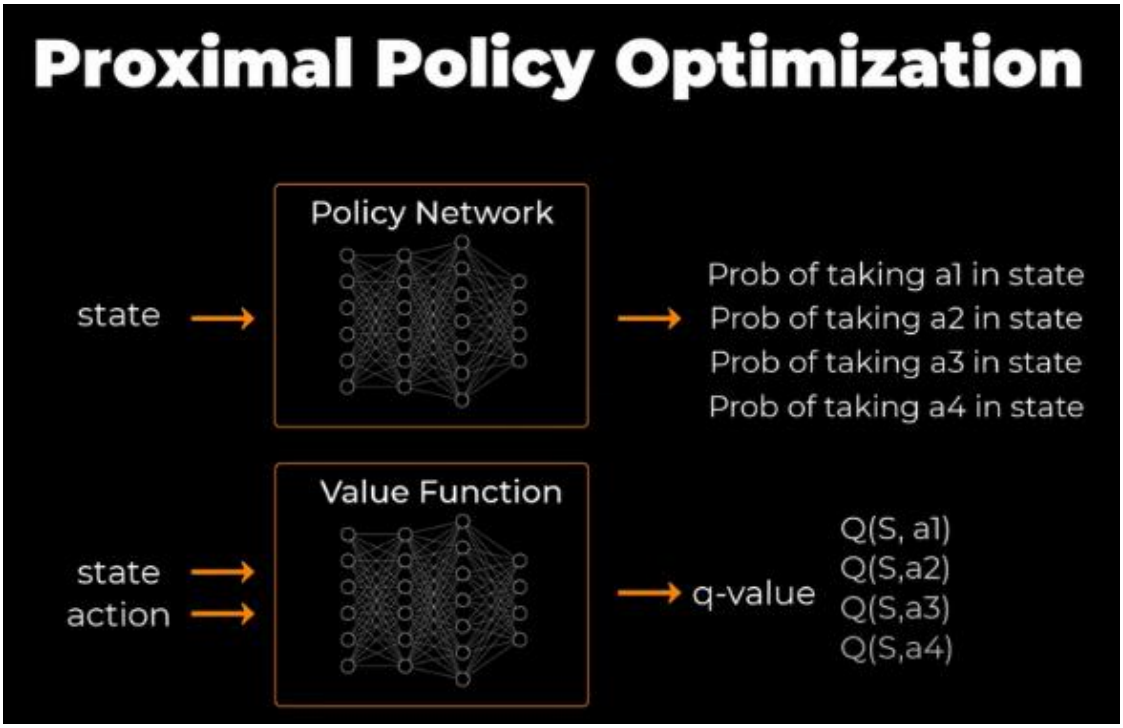

Figure 4.4: Proximal Policy Optimisation

Key Features of PPO:

1. **Policy Optimization Framework:** PPO optimizes policies using a clipped surrogate objective function, ensuring stable and reliable learning by preventing drastic policy updates. This makes the training process more efficient and reduces the risk of performance collapse.

2. **Actor-Critic Architecture:** PPO leverages an actor-critic approach, where the **policy network** (actor) determines actions, and the **value network** (critic) evaluates the expected cumulative reward of a state. This structure improves convergence and policy learning.
3. **Advantage Estimation:** The model uses Generalized Advantage Estimation (GAE) to compute the advantage function, helping to reduce variance while maintaining bias control. This enables PPO to make more stable updates to the policy.
4. **Experience Buffer:** PPO collects and stores agent trajectories (states, actions, and rewards) to perform batch updates. This experience replay mechanism improves sample efficiency and helps refine decision-making.

Implementation Details:

- **Input Layer:** The network takes state observations from the environment and converts them into feature representations for policy and value estimation.
- **Policy Network:** Generates a probability distribution over possible actions, allowing the agent to make decisions based on the current state.
- **Value Network:** Predicts the expected return from a given state, providing a baseline for policy updates and improving stability in learning.
- **Advantage Estimation:** Calculates how much better an action is compared to the average action at a given state, guiding policy updates.
- **Output Layer:** Converts internal representations into either action probabilities (for the policy network) or a scalar value estimating expected returns (for the value network).

Integration Architecture:

- **Frontend:** The user or agent interacts with the environment by taking actions and receiving state observations and rewards.
- **Middleware:** Processes state observations for the PPO model and translates policy outputs into actions for execution.
- **PPO-Core:** Manages policy optimization and value prediction, refining decision-making using collected experience data.
- **Backend:** Stores agent trajectories (states, actions, and rewards) for training and optimization, ensuring the model continually improves.

Benefits of Using PPO:

- **Stability and Efficiency:** The clipped objective function ensures stable updates, preventing drastic policy changes and improving convergence.
- **Scalability:** PPO can be applied to various environments, from robotics to game AI, by fine-tuning hyperparameters.
- **Sample Efficiency:** The experience buffer mechanism allows effective learning with fewer interactions, reducing computational cost.

By integrating the PPO algorithm into the system, agents can learn optimal decision-making strategies in complex environments, making it a reliable solution for reinforcement learning applications.

CHAPTER 5: IMPLEMENTATION

5.1 Code snippets

1. Importing the Libraries

This section imports the necessary libraries for training and evaluating a reinforcement learning-based sustainability tracker.

```
import logging
from src.environment.eco_environment import EcoProductEnvironment
from src.models.actor_critic import ActorCritic
from src.trainer import PPOTrainer
from src.utils.database import init_database
from src.config import TRAINING_CONFIG
```

Figure 5.1: Importing necessary libraries

Core Libraries:

- **logging**: Used for logging messages throughout the training process, making debugging and tracking progress easier.
- **src.environment.eco_environment.EcoProductEnvironment**: Defines the environment in which the agent interacts. This environment models various sustainability factors, simulating real-world ecological impacts and decisions.
- **src.models.actor_critic.ActorCritic**: Implements the Actor-Critic reinforcement learning model, which consists of two main components:
    - **Actor**: Determines the best actions to take based on the current state.
    - **Critic**: Evaluates the actions taken by the Actor, providing a value estimate for policy improvement.
- **src.trainer.PPOTrainer**: Handles the training of the reinforcement learning model using the Proximal Policy Optimization (PPO) algorithm. PPO is chosen for its balance between performance and stability.
- **src.utils.database.init_database**: Initializes the database, ensuring tables and schema are correctly set up to store sustainability tracking data.
- **src.config.TRAINING_CONFIG**: Contains hyperparameters and configuration settings such as learning rate, batch size, and discount factor.

2. Logging Setup

```
def setup_logging():
    logging.basicConfig(
        level=logging.INFO,
        format='%(asctime)s - %(name)s - %(levelname)s - %(message)s'
    )
```

Figure 5.2: Code for logging setup

To track the execution process and potential errors, logging is configured using:

```
logging.basicConfig(
   level=logging.INFO,
   format='%(asctime)s - %(name)s - %(levelname)s - %(message)s'
)
```

This ensures that log messages include timestamps, module names, and severity levels. Proper logging aids in debugging and monitoring the training process effectively.

3. Environment Setup and Model Initialization

1. **Database Initialization**: Before training, `init_database()` sets up the database, ensuring all necessary tables are created and the sustainability data schema is correctly structured.
2. **Environment Creation**:
   - `env = EcoProductEnvironment()`: This initializes the sustainability-based reinforcement learning environment. The environment simulates various eco-friendly scenarios where an AI agent makes decisions to optimize sustainability.
   - The environment provides an **observation space** (possible states) and **action space** (possible decisions the agent can make).
3. **Actor-Critic Model Initialization**:
   - `state_dim = env.observation_space_size`: Determines the number of state variables representing environmental conditions.
   - `action_dim = env.action_space_size`: Defines the number of possible actions the agent can take to promote sustainability.
   - `model = ActorCritic(state_dim, action_dim)`: Initializes the reinforcement learning model with the ability to learn optimal policies based on state-action interactions.

4. Training Process

The training process follows these steps:

1. **Trainer Setup**:
   ```
   trainer = PPOTrainer(
   env=env,
   model=model,
   **TRAINING_CONFIG
   )
   ```
   - The PPOTrainer is responsible for executing the training loop, where the agent interacts with the environment, collects experiences, and updates its policy.
   - Training configurations, such as discount factor and learning rate, are loaded from `TRAINING_CONFIG`.
2. **Starting Training**:
   ```
   trainer.train()
   ```
   - The training process begins, where the AI agent continuously refines its decision-making capabilities through reinforcement learning.
   - The trainer evaluates the model's performance at regular intervals, adjusting policies as needed.
3. **Logging Training Progress**:
   - Success messages are logged after each training step.
   - Any encountered errors are logged using `logger.error()`, ensuring that debugging is straightforward.

5. Dependencies

The following libraries are essential for running the project, as listed in `requirements.txt`:

- `numpy>=1.26.0`: Provides numerical computing capabilities, including matrix operations and array manipulations.
- `torch>=2.0.1`: A deep learning framework used to build and optimize the reinforcement learning model.
- `pandas>=2.0.3`: Used for data analysis and manipulation, particularly for handling sustainability tracking logs and experiment results.
- `sqlalchemy>=2.0.23`: A robust SQL toolkit and ORM (Object Relational Mapper) used for database interactions.
- `psycopg2-binary>=2.9.9`: A PostgreSQL database adapter necessary for connecting and executing queries on the database.
- `python-dotenv>=1.0.0`: Loads environment variables from a `.env` file to manage database configurations and sensitive credentials securely.

6. Model Evaluation and Deployment

After training, the best-performing model is evaluated using test environments to ensure its effectiveness. The deployment process includes:

1. **Evaluating Performance**:
    - The model is tested in different sustainability scenarios to assess its decision-making efficiency.
    - Performance metrics such as reward accumulation and sustainability scores are recorded.
2. **Saving the Model**:
    - The trained model is stored using PyTorch's `torch.save()` method, ensuring it can be reloaded for inference or further training.
3. **Deployment Considerations**:
    - The model is integrated into a real-world sustainability tracking system where it makes continuous decisions to optimize eco-friendly outcomes.
    - Future updates to the model are tracked and versioned using database records and logging mechanisms.

This document outlines the core components of the sustainability tracker, providing a detailed explanation of the system's initialization, training, logging, and model deployment processes. By leveraging reinforcement learning, the system continuously improves its ability to recommend sustainable actions based on real-world data.

5.2 Results and Discussions

The chatbot gives the recommendation of the products, if a valid product category is mentioned, as shown in Figure 5.3

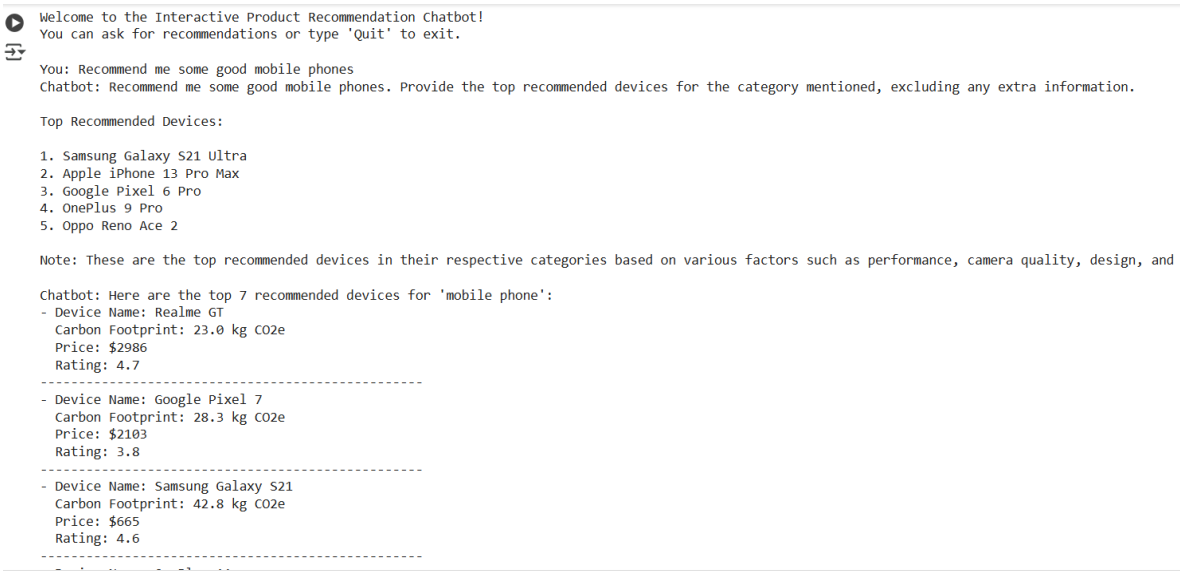

Figure 5.3: The result when a valid category is given

The chatbot tells the user that the category is not valid, if invalid category is entered, as shown in Figure 5.4

```
You: Recommend me some apples
Chatbot: 'Recommend me some apples' is not a valid category. Valid categories are: Laptop, Mobile Phone, Tablet, Smartwat
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
<ipython-input-6-7db9587f4730> in <cell line: 0>()
```

Figure 5.4: The result when an invalid category is entered

```
Enter the product category (select number or type the category name): Laptop

Enter product specifications:
RAM (e.g., 8GB): 16GB
Processor (e.g., Intel i7): AMD
Storage (e.g., 256GB SSD): 512GB

Recommended Products:

Device Name: Asus ZenBook
Category: Laptop
Specifications: Specifications for Asus ZenBook - RAM: 16GB, Processor: AMD, Storage: 512GB
Carbon Footprint: 70.1 kg CO2e
Price: $573
Rating: 3.1
```

```
Device Name: Acer Swift 3
Category: Laptop
Specifications: Specifications for Acer Swift 3 - RAM: 16GB, Processo
r: AMD, Storage: 512GB
Carbon Footprint: 42.9 kg CO2e
Price: $2573
Rating: 3.0


Device Name: HP Spectre x360
Category: Laptop
Specifications: Specifications for HP Spectre x360 - RAM: 16GB, Proces
sor: AMD, Storage: 512GB
Carbon Footprint: 63.2 kg CO2e
Price: $1217
Rating: 4.6
```

Figure 5.5: Product Recommendations Output

CHAPTER 6: CONCLUSION

Conclusion :

The Sustainability Recommender System demonstrates the transformative potential of reinforcement learning (PPO) and AI-driven decision-making in promoting eco-friendly consumer behavior. By integrating a reinforcement learning-based recommendation engine, an LLama-powered chatbot, and carbon footprint analysis using the LCA dataset, the system successfully provides personalized, sustainability-focused product recommendations.

The actor-critic model trained with PPO ensures that recommendations optimize both user preferences and environmental impact, balancing short-term usability with long-term sustainability goals. The PostgreSQL database efficiently manages product attributes, sustainability scores, and user interaction history, ensuring smooth data processing and retrieval. The Streamlit-based web interface enhances accessibility, providing an intuitive platform for users to interact with the system and chatbot.

Compared to conventional recommendation engines, this system stands out for its focus on sustainability, incorporating LCA-based carbon footprint data to guide eco-conscious choices. The LLama chatbot further enhances user engagement by explaining sustainability metrics and offering interactive responses on environmentally friendly alternatives.

The practical benefits of this system are significant. By offering real-time recommendations and sustainability insights, it empowers consumers to make informed choices, reducing their carbon footprint. The system's scalability makes it suitable for integration with e-commerce platforms, corporate sustainability initiatives, and policy-driven environmental programs.

In a broader context, this project illustrates the growing role of AI and reinforcement learning in sustainability efforts. As consumer awareness of environmental impact continues to rise, AI-driven solutions like this recommender system will play a crucial role in guiding responsible consumption. Future enhancements, such as expanded product categories, integration with industry sustainability policies, and improved chatbot intelligence, will further solidify its impact.

This project marks a significant step toward a more sustainable future, demonstrating how technology, data-driven insights, and AI-powered optimization can drive meaningful environmental change.

CHAPTER 7: FUTURE ENHANCEMENTS

Future Enhancements

To further enhance the Sustainability Recommender System, several key improvements can be implemented:

1. Model Improvement

- Exploring more advanced reinforcement learning techniques, such as Twin Delayed Deep Deterministic Policy Gradient (TD3) or Soft Actor-Critic (SAC), could improve policy optimization and stability.
- Fine-tuning the PPO model with multi-objective reinforcement learning to balance sustainability, cost, and user preference more effectively.
- Enhancing the actor-critic network architecture with attention mechanisms to focus on the most impactful sustainability factors when making recommendations.

2. Dataset Expansion and Refinement

- Expanding the LCA dataset to include more product categories, industry-specific sustainability data, and lifecycle stages to improve the accuracy of carbon footprint calculations.
- Integrating real-time sustainability data sources, such as supply chain emissions tracking, to ensure recommendations reflect current market trends.
- Addressing biases in sustainability scoring by refining the dataset with regional and industry-specific environmental impact assessments.

3. Real-Time Performance and Optimization

- Optimizing PPO training by implementing parallelized training environments, reducing convergence time for faster model updates.
- Reducing database query latency by indexing sustainability data and optimizing PostgreSQL queries for large-scale deployments.
- Implementing caching mechanisms for frequently recommended products to speed up response times in the Streamlit web app.

4. Deployment and User Interface Enhancements

- Enhancing the Streamlit UI with dynamic visualizations of sustainability scores, product comparisons, and carbon footprint trends.
- Improving the LLama chatbot’s conversational abilities by fine-tuning it on domain-specific sustainability knowledge for more accurate and insightful responses.

- Adding user feedback loops, allowing consumers to rate recommendations and refine preferences, helping the system learn and adapt over time.

5. Cross-Platform Support and Integration

- Developing an API-based recommendation engine, allowing e-commerce platforms and sustainability-focused businesses to integrate real-time product suggestions.
- Deploying the system on a cloud-based architecture for scalability and accessibility, enabling users to access sustainability insights from any device.
- Implementing multilingual support for the chatbot and web interface, ensuring global accessibility and usability.

By implementing these enhancements, the Sustainability Recommender System can become more accurate, scalable, and user-friendly, ultimately empowering consumers, businesses, and policymakers to make more informed eco-conscious decisions.

CHAPTER 8: BIBLIOGRAPHY

[1] He, Bin, and Hangyu Mao. "Digital twin-driven product sustainable design for low carbon footprint." Journal of Computing and Information Science in Engineering 23.6 (2023): 060805.

[2] Gao, Tao, Qing Liu, and Jianping Wang. "A comparative study of carbon footprint and assessment standards." *International Journal of Low-Carbon Technologies* 9.3 (2014): 237-243.

[3] Pandey, Divya, Madhoolika Agrawal, and Jai Shanker Pandey. "Carbon footprint: current methods of estimation." Environmental monitoring and assessment 178 (2011): 135-160.

[4] C. C. Sudarshan, N. Matkar, S. Vrudhula, S. S. Sapatnekar and V. A. Chhabria, "ECO-CHIP: Estimation of Carbon Footprint of Chiplet-based Architectures for Sustainable VLSI," *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Edinburgh, United Kingdom, 2024, pp. 671-685, doi: 10.1109/HPCA57654.2024.00058.

[5] Zhang, Tianwei, et al. "Life cycle assessment (LCA) of circular consumer electronics based on IC recycling and emerging PCB assembly materials." *Scientific Reports* 14.1 (2024): 29183.

[6] J. Malmodin and N. Lövehagen, "A Methodology for Simplified LCAs of Electronic Products," *2024 Electronics Goes Green 2024+ (EGG)*, Berlin, Germany, 2024, pp. 1-12, doi: 10.23919/EGG62010.2024.10631258.

[7] W. Qiang and Z. Zhongli, "Reinforcement learning model, algorithms and its application," *2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC)*, Jilin, China, 2011, pp. 1143-1146, doi: 10.1109/MEC.2011.6025669.

[8] W. F. Álvarez-Castañeda, L. A. Martínez-Tejada and D. Y. Riscanevo-Espitia, "Carbon Footprint meter prototype due to power consumption: Case of study Colombia," *2016 IEEE ANDESCON*, Arequipa, Peru, 2016, pp. 1-4, doi: 10.1109/ANDESCON.2016.7836220.

[9] T. Pirson, T. Delhaye, A. Pip, G. Le Brun, J. -P. Raskin and D. Bol, "The Environmental Footprint of IC Production: Meta-Analysis and Historical Trends," *ESSDERC 2022 - IEEE 52nd European Solid-State Device Research Conference (ESSDERC)*, Milan, Italy, 2022, pp. 352-355, doi: 10.1109/ESSDERC55479.2022.9947198.

[10] T. Pirson, T. Delhaye, A. Pip, G. Le Brun, J. -P. Raskin and D. Bol, "The Environmental Footprint of IC Production: Meta-Analysis and Historical

Trends," ESSDERC 2022 - IEEE 52nd European Solid-State Device Research Conference (ESSDERC), Milan, Italy, 2022, pp. 352-355, doi: 10.1109/ESSDERC55479.2022.9947198.

[11] Vasan, Arvind, Bhanu Sood, and Michael Pecht. "Carbon footprinting of electronic products." Applied energy 136 (2014): 636-648.

[12] Duan, Huabo, et al. "Life cycle assessment study of a Chinese desktop personal computer." Science of the total environment 407.5 (2009): 1755-1764.

[13] Ueno, Takayoshi, Toru Shiino, and Hiroshi Onishi. "Evaluation of electronic components in life cycle assessment." Journal of Material Cycles and Waste Management 1 (1999): 25-32.

[14] J. Valkama and M. Keskinen, "Comparison of simplified LCA variations for three LCA cases of electronic products from the ecodesign point of view," 2008 IEEE International Symposium on Electronics and the Environment, San Francisco, CA, USA, 2008, pp. 1-6, doi: 10.1109/ISEE.2008.4562923.

[15] Lövehagen, Nina, et al. "Assessing embodied carbon emissions of communication user devices by combining approaches." Renewable and Sustainable Energy Reviews 183 (2023): 113422.

[16] Sloma, Marcin. "Carbon footprint of electronic devices." Electron Technology Conference 2013. Vol. 8902. SPIE, 2013.

[17] Teehan, Paul, and Milind Kandlikar. "Comparing embodied greenhouse gas emissions of modern computing and electronics products." Environmental science & technology 47.9 (2013): 3997-4003.

[18] Yung, Winco KC, Subramanian Senthilkannan Muthu, and Karpagam Subramanian. "Carbon footprint analysis of printed circuit board." Environmental Carbon Footprints. Butterworth-Heinemann, 2018. 365-431.

[19] Wojciechowski, Hubert, and Roman Domański. "Assessment of the product carbon footprint of office equipment across the entire life cycle." Economics and Environment 89.2 (2024): 757-757.

[20] Mayers, Kieren, et al. "The carbon footprint of games distribution." Journal of Industrial Ecology 19.3 (2015): 402-415.