Reverse Logistics SaaS Platform for Sustainable E-commerce Returns

Software Design Specification for

Project Work-1

By

Anushka Paharia
EN22CS301183

Date: Oct 27, 2025

Under the Guidance of: Prof. Mohammad Mazhar and Prof. Roshni Verma

Department of Computer Science & Engineering Faculty of Engineering
MEDICAPS UNIVERSITY, INDORE- 453331
Aug-Dec'25

Contents

List of Figures

1 Introduction

1.1 Purpose

This Software Design Specification (SDS) defines the detailed architecture and system design for the Reverse Logistics SaaS Platform. It enables sustainable product returns through optimized logistics, data-driven decisions, and carbon impact analytics.

1.2 Document Conventions

This document follows IEEE SRS/SDS conventions. Diagrams are included using PlantUML or other UML tools. Headings are hierarchical and numbered (e.g., 1.1, 2.1).

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- Developers: to understand module design, APIs, and logic flow.
- Project Evaluators/Faculty: to assess architectural and algorithmic depth.
- Testers: to identify testable modules and expected outputs.
- End Users (Admin & Retailers): to understand system functionalities.

Suggested reading order:

- Introduction → Overview of goals
- Analysis Model → System behavior & logic
- Design Model → Architecture, database, and interface details

1.4 References

1 Grand View Research. (2024). E-commerce Market Size, Share & Trends Analysis Report. Retrieved September 7, 2025, from https://www.grandviewresearch.com/industry-analysis/e-commerce-market

2 Agrawal, S., & Singh, R. K. (2019). Analyzing disposition decisions for sustainable reverse logistics: Triple Bottom Line approach. Resources, Conservation & Recycling, 150, 104448.

3 Zarbakhshnia, N., Soleimani, H., & Goh, M. (2019). A novel multi-objective model for green forward and reverse logistics network design. Journal of Cleaner Production, 208, 1304–1316.

4 Ramos, T. R. P., Gomes, M. I., & Barbosa-Póvoa, A. P. (2014). Planning a sustainable reverse logistics system: Balancing costs with environmental and social concerns. Omega, 48, 60–74.

5 Agrawal, S., & Singh, R. K. (2020). Forecasting product returns and reverse logistics performance: Structural equation modelling. Management of Environmental Quality: An International Journal, 31(5), 1223-1237.

6 Dutta, P., Mishra, A., Khandelwal, S., & Katthawala, I. (2019). A multiobjective optimization model for sustainable reverse logistics in Indian E-commerce market. Journal of Cleaner Production, 119348.

7 Dabees, A., Barakat, M., Elbarky, S. S., & Lisec, A. (2023). A Framework for Adopting a Sustainable Reverse Logistics Service Quality for Reverse Logistics Service Providers: A Systematic Literature Review. Sustainability, 15(3), 1755.

8 Jain, R., & Kumar, R. (2025). Automated Detection of Fraudulent Returns in E-Commerce: A Machine Learning and Blockchain Approach. ResearchGate. Retrieved September 7, 2025

2 Analysis Model

2.1 Methodology Used

The system is designed using an event-driven, function-oriented approach with the **Waterfall Model**. Ce frontend (React) is component-driven; the backend is implemented as a set of services (Node.js/Express) exposing RESTful APIs. Each service encapsulates a clear responsibility (e.g., Returns Service, Sustainability Engine, ML/Fraud Service, Analytics Service). This approach supports separation of concerns, testability, and incremental development.

2.2 Use Case Diagram and Specification

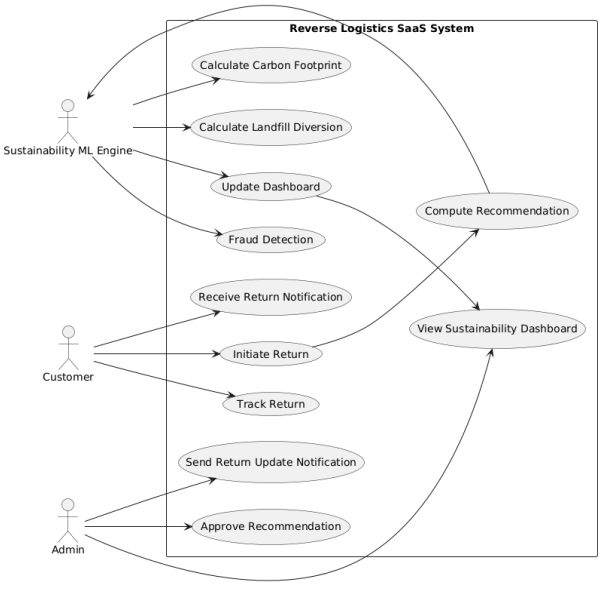

Figure 1: Use Case Diagram

Use Case Specification:

- **Name:** Manage Return Request
- **Actor:** Customer
- **Precondition:** A valid order exists.
- **Primary Flow:** Customer initiates return → System validates → ML recommends → Admin approves.
- **Alternate Flow:** Fraud flagged for manual review.
- **Postcondition:** Return completed or rejected.
- **Termination:** Workflow ends after final decision.

2.3 ER Model

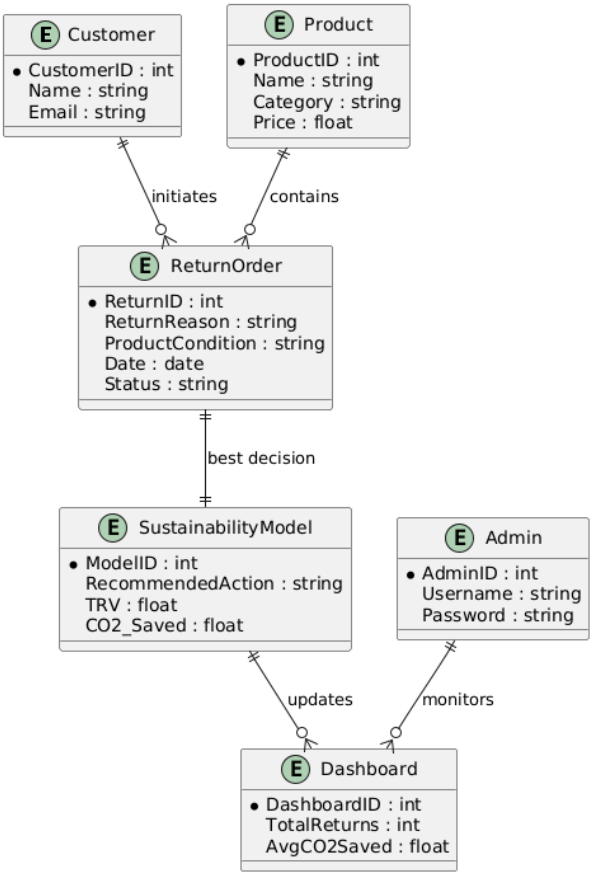

Figure 2: ER Diagram

2.4 Data Flow Diagram (DFD) and Process Specifications

The DFD illustrates data exchange between Customer, Admin, ML Engine, and Database.

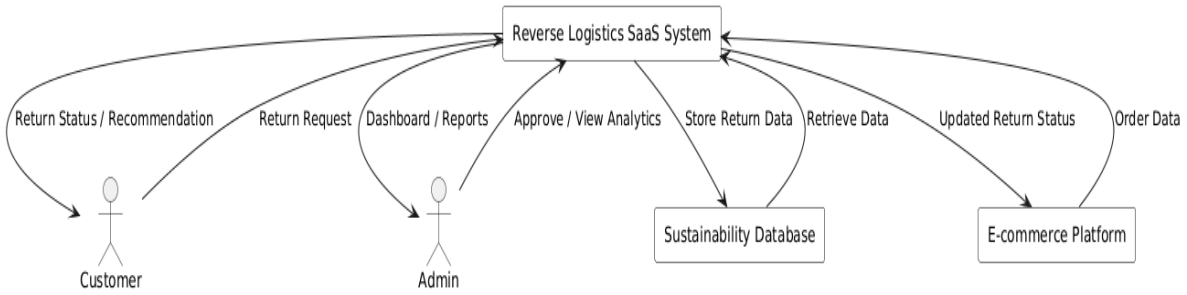

Figure 3: DFD Level 0

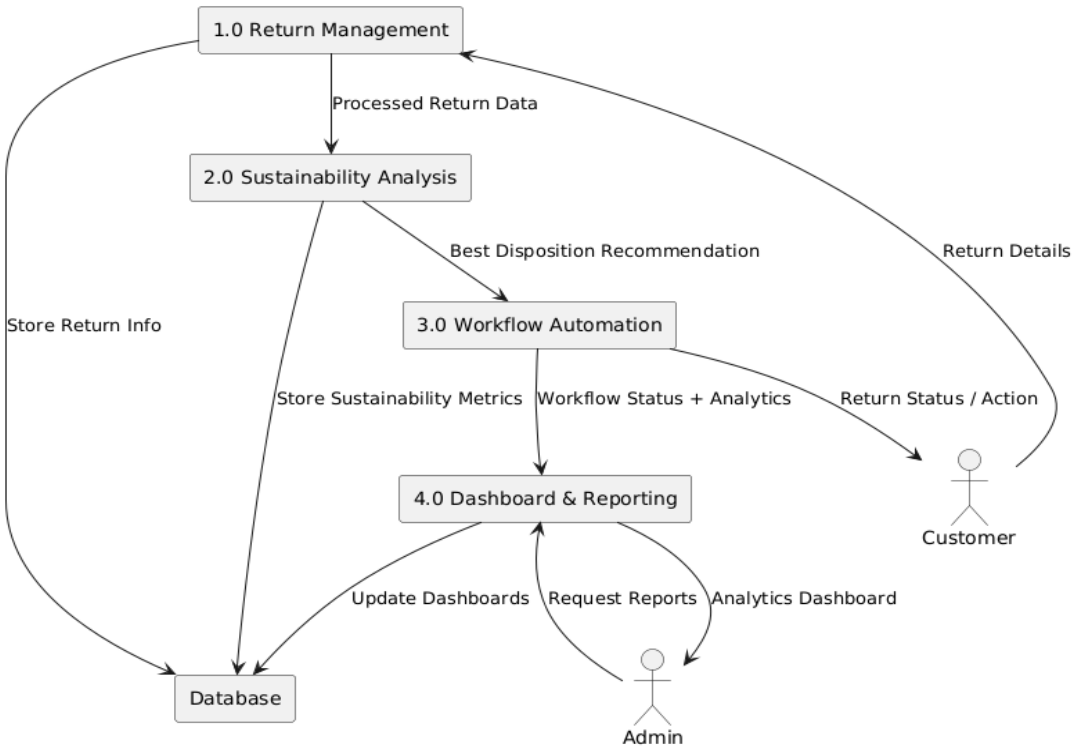

Figure 4: DFD Level 1

2.5 Control Flow Diagram (CFD) and Control Specifications

Control modules include: Return Controller, Fraud Detection, ML Engine, Admin Review, and Notification Manager.

2.6 State Transition Diagram

States: Initiated, Validated, Under Review, Approved, Executed, Completed, and Rejected.

3 Design Model

3.1 Architectural Design

The system follows a three-tier design:

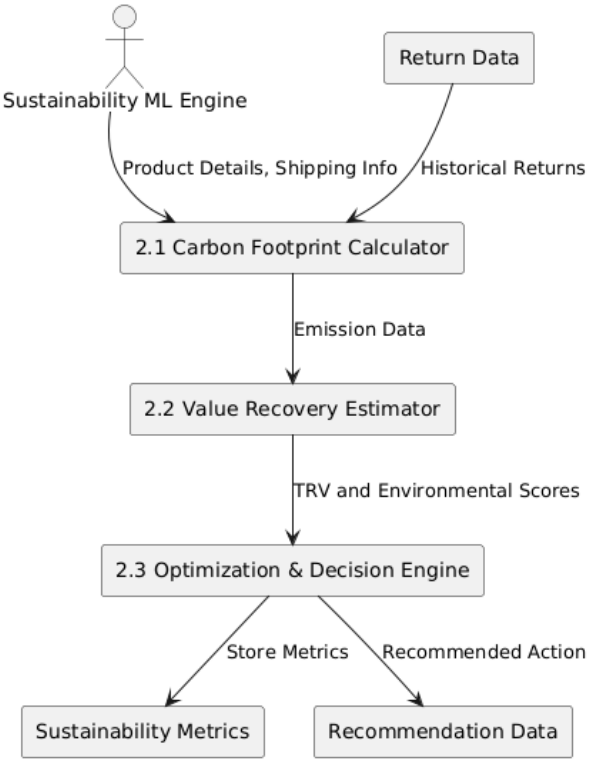

Figure 5: DFD Level 2

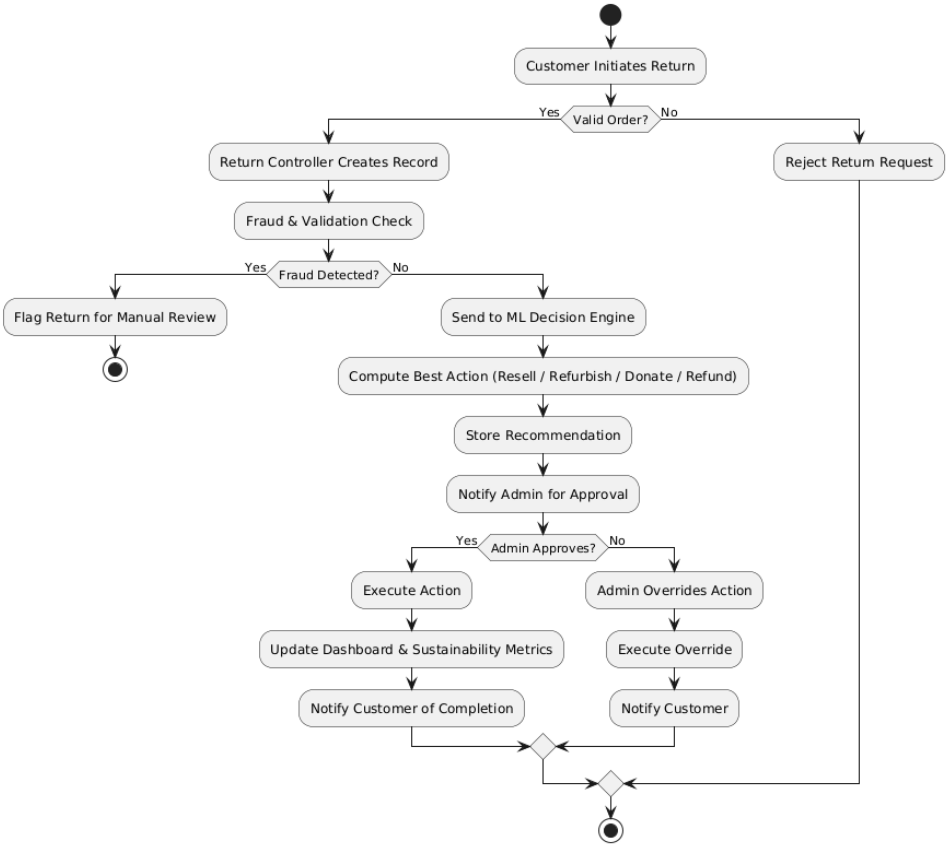

Figure 6: Control Flow Diagram

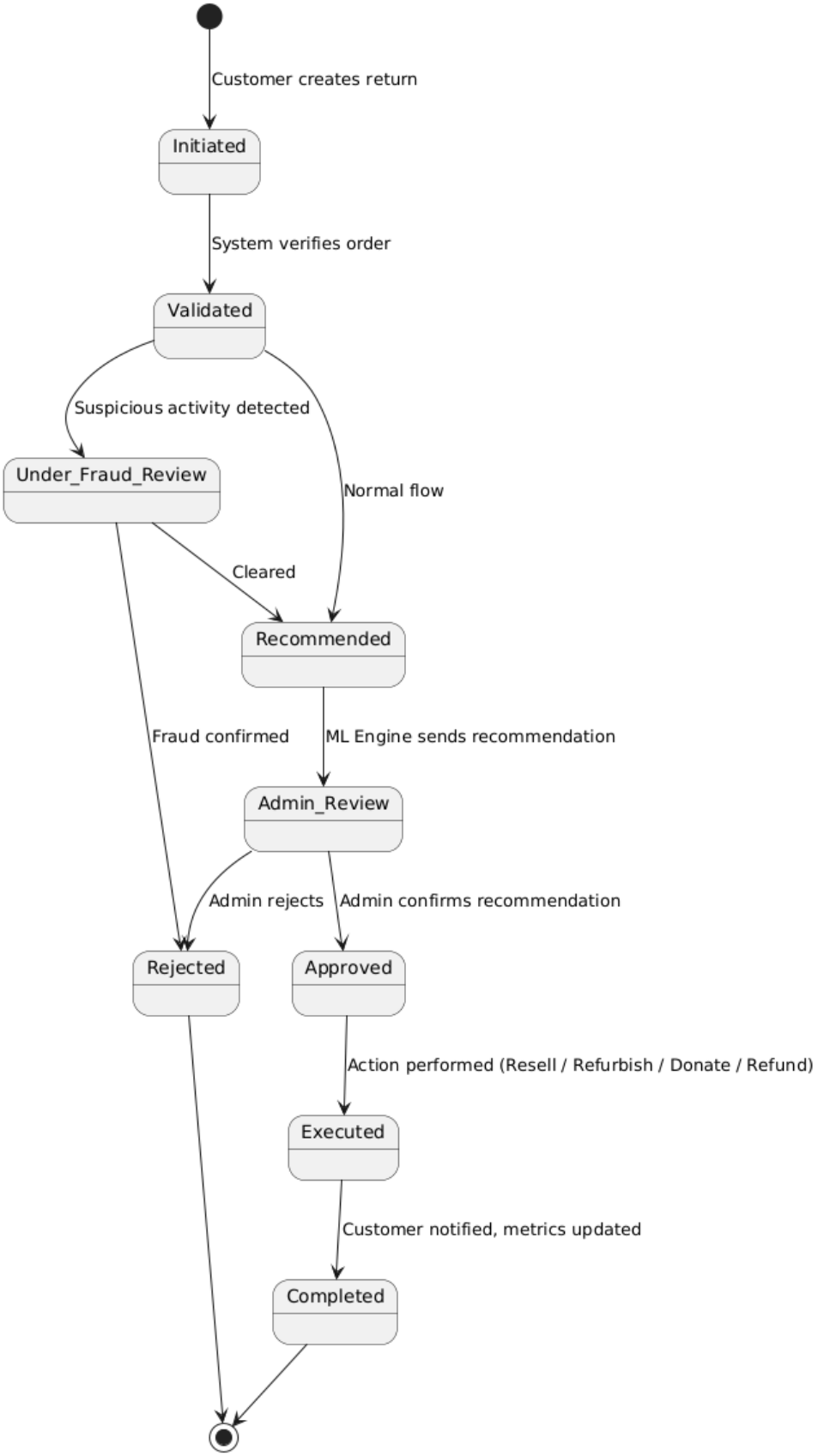

Figure 7: State transition Diagram

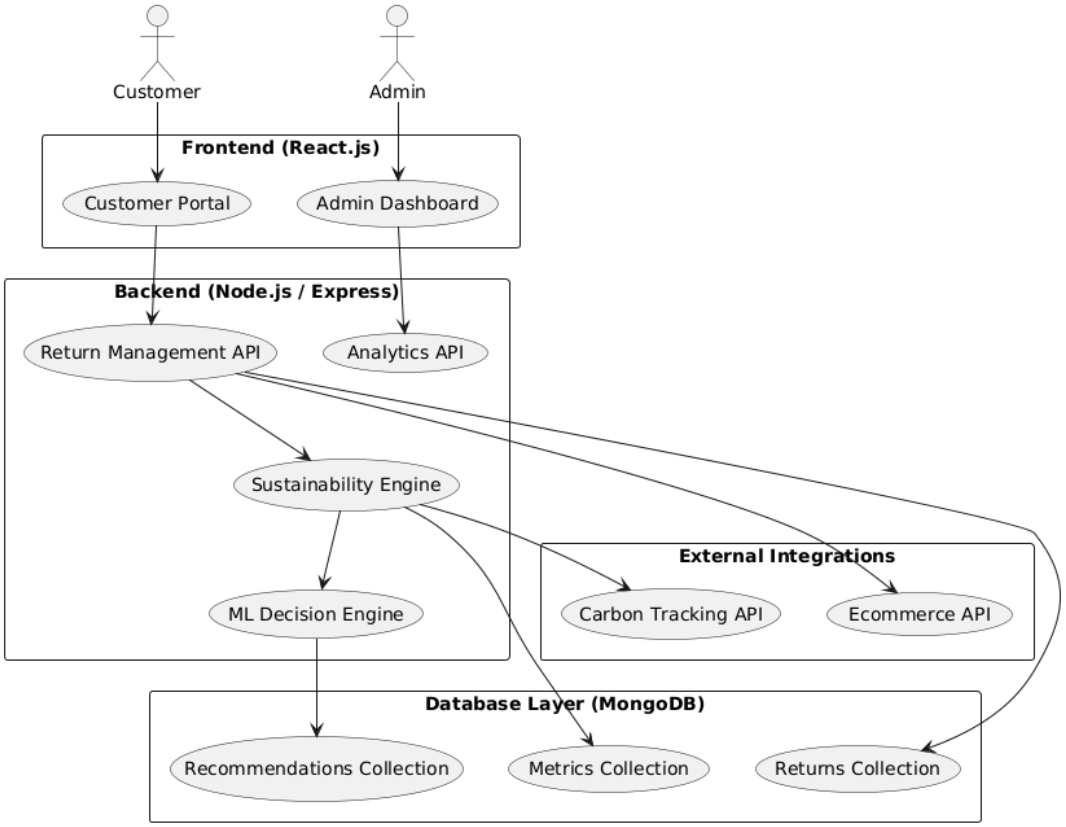

Figure 8: System Architecture Diagram

1. **Frontend:** React.js Customer Portal and Admin Dashboard
2. **Backend:** Node.js/Express REST APIs
3. **Database:** MongoDB
4. **ML Layer:** Python for sustainability computations

3.2 Database Design

The database stores User, Product, ReturnOrder, Recommendation, and SustainabilityMetric entities.

3.2.1 Data Dictionary

- **User:** user_id, name, email, role
- **Product:** product_id, name, category, price
- **ReturnOrder:** return_id, reason, status, product_id
- **Recommendation:** rec_id, suggested_action, confidence
- **SustainabilityMetric:** metric_id, co2_saved, landfill_diversion, packaging_saved

3.2.2 Normalization

The database schema is normalized up to 3NF:

- **1NF:** Atomic data fields
- **2NF:** Attributes fully depend on primary key
- **3NF:** No transitive dependencies between non-key attributes

3.3 Component Design

Major software components:

1. Return Management Module
2. ML Decision Engine
3. Sustainability Metrics Engine
4. Admin Dashboard
5. Integration Layer

3.4 Flow Chart

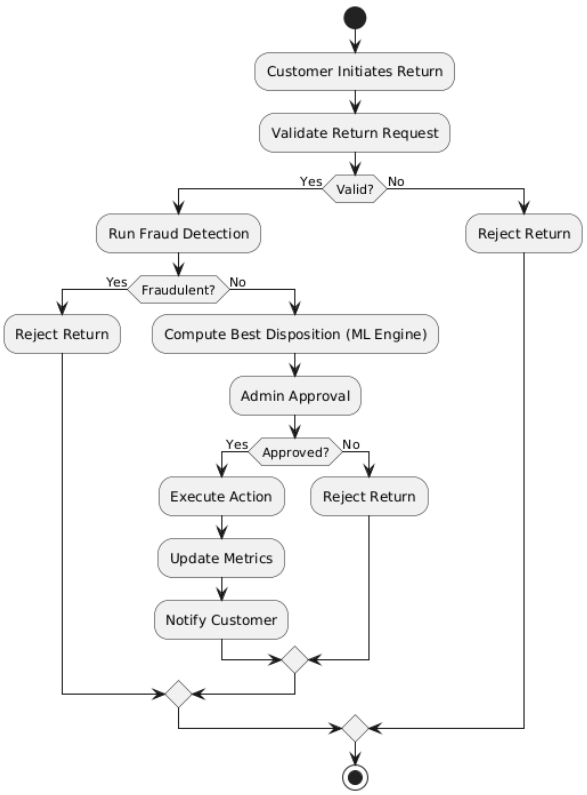

Figure 9: Flow Chart

Appendix B1: Glossary

- ML: Machine Learning
- CO_2: Carbon Dioxide
- SDS: Software Design Specification
- ERD: Entity Relationship Diagram
- DFD: Data Flow Diagram

Appendix B2: To Be Determined List

- Integration with new e-commerce platforms
- Enhanced fraud detection model
- Real-world dataset integration for sustainability computation