

**"LogiFlex" —** Combining "logistics" with "flexibility," highlighting adaptability in supply chain decisions.

**(Trimester V)**

**Report**

**Submitted by**

**Himanshu Salvekar(2348052)**

**Sayantan Ray (2348057)**

**Swarnasish Banerjee (2348066)**

Content

**[Abstract 2](#_Toc1253370328)**

**[1. Introduction 3](#_Toc728009273)**

**[2. Project study 4](#_Toc1437514352)**

**[3. Literature Review 4](#_Toc1112685314)**

**[4. Survey of Models Used 5](#_Toc614474322)**

[1.1 Models Implemented 5](#_Toc221071431)

[1.2 REINFORCE (Policy-Gradient): 5](#_Toc400831507)

[1.3 Threshold-Based Agent (S-Q Agent): 5](#_Toc118937510)

[1.4 Environment 5](#_Toc1822938860)

**[5 Dataset Description 6](#_Toc2131711918)**

**[6. Modules and Components 6](#_Toc1212522925)**

[6.1 Core Modules 6](#_Toc1400474092)

[ Environment (SupplyDistribution) 6](#_Toc1347293124)

[ Agents 7](#_Toc887961100)

[ Evaluation 7](#_Toc1098344697)

[ Visualization (GraphCreation) 7](#_Toc109892867)

**[7. System Analysis 10](#_Toc133479249)**

**[8. System Design 10](#_Toc1412810475)**

[ System Architecture 10](#_Toc378968446)

[ Module Design 10](#_Toc2033658567)

[ Database Design 11](#_Toc349809917)

**[9. Implementation 11](#_Toc1592533180)**

**[10. Conclusions 11](#_Toc1616463699)**

[ Design and Implementation Issues 11](#_Toc89770896)

[ Advantages and Limitations 12](#_Toc1245928878)

[ Future Enhancements 12](#_Toc213610649)

**[11. Team Responsibilities 12](#_Toc1709003606)**

[ Himanshu Salvekar - Data Analysis & Visualization 12](#_Toc629827417)

[ Sayantan Ray - Algorithm Design & Implementation 13](#_Toc562501456)

[ Swarnasish Banerjee - Simulation Environment & System Integration 13](#_Toc738956898)

Abstract

This report presents the design, implementation, and evaluation of a reinforcement learning (RL)-based supply chain optimization system. The project aims to optimize key supply chain operations, including inventory management, transportation, and delivery scheduling, by leveraging various RL algorithms. We employ an Approximate SARSA agent, a policy-gradient REINFORCE agent, and a threshold-based inventory management agent (S-Q agent) for baseline comparison, within a custom-designed supply chain environment. The system’s performance is evaluated using cumulative reward metrics, policy stability, and convergence rates, with results visualized using graphs for further analysis. The project emphasizes collaboration, with each team member contributing specific modules such as agent design, environment creation, performance evaluation, and testing. The system provides a framework for simulating and optimizing real-world supply chain challenges, demonstrating the potential of RL to improve decision-making in dynamic, cost-sensitive environments.

# Introduction

This report details the contributions of each team member to the various modules implemented in the project. Each module's functionality, purpose, and specific contributions are outlined below. This documentation also provides a theoretical overview of the methods and algorithms used, ensuring a comprehensive understanding of the project’s scope and execution.

**OBJECTIVE**

The primary objective of this project is to develop and evaluate reinforcement learning agents designed to optimize various aspects of supply chain management. These agents aim to minimize operational costs, maximize rewards (profits), and improve decision-making efficiency over extended simulation episodes, ensuring efficient operations even with changing environments.

# **2. Project study**

**Overview**

The study involves designing and implementing reinforcement learning algorithms to optimize decision-making processes in a multi-faceted supply chain environment. The tasks include managing inventory, optimizing transportation routes, and scheduling deliveries. The study also involves:

* **Creating a Custom Simulation Environment:** A simulation model representing real-world supply chain dynamics, including stores, transportation systems, and inventory management systems.
* **Designing and Implementing RL Algorithms:** The algorithms include SARSA (State-Action-Reward-State-Action), REINFORCE (a policy-gradient method), and a custom threshold-based inventory management system (S-Q agent).
* **Evaluating Agent Performance:** Agent performance is measured using several metrics, including cumulative rewards, policy stability, and operational efficiency across simulation episodes.

# 3. ****Literature Review****

Recent advancements in reinforcement learning have shown great promise in tackling logistics problems. Algorithms such as Q-learning, SARSA, and policy-gradient methods (e.g., REINFORCE) are frequently used in optimizing supply chain problems like inventory management, route planning, and dynamic pricing. However, applying these methods to complex, multi-factorial supply chain environments, where many variables interact (e.g., demand fluctuations, transportation constraints, etc.), remains an emerging area of research.

Previous studies have demonstrated the potential of these algorithms in reducing costs and improving decision-making efficiency in real-world applications. Nevertheless, there is a need for further exploration in adapting these algorithms to handle diverse and multi-dimensional supply chain challenges.

# 4. Survey of Models Used

## Models Implemented

**SARSA (State-Action-Reward-State-Action):**

* **Type**: On-policy temporal difference learning.
* **Purpose:** Learns policies by evaluating state-action pairs and updating them based on rewards received after actions are taken.
* **Key Features:** Utilizes a linear function approximator to generalize across states in large state spaces, making it scalable for complex environments.

## REINFORCE (Policy-Gradient):

* **Type**: Monte Carlo policy-gradient algorithm.
* **Purpose**: Uses policy gradient methods to directly learn a policy by maximizing cumulative rewards.
* **Key Features**: It can handle continuous action spaces and utilizes probabilistic action selection.

## Threshold-Based Agent (S-Q Agent):

* **Type**: Heuristic-based agent.
* **Purpose**: Performs inventory management by making decisions based on predefined reorder thresholds.
* **Key Features**: Simplicity and interpretability make it useful for baseline comparisons with more complex agents.

## Environment

The simulation environment models a multi-store supply chain, which includes:

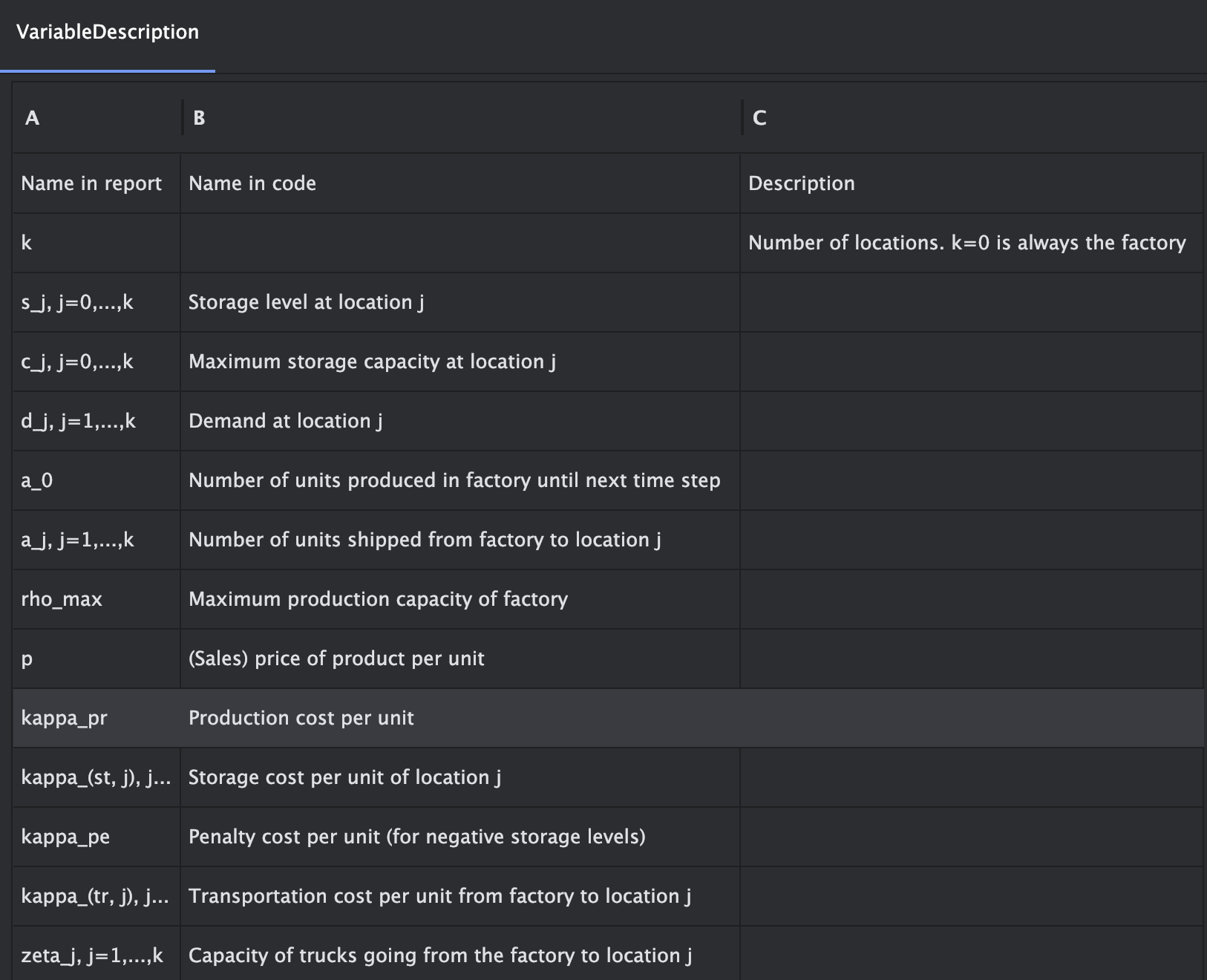
* **Truck Capacities**: Limits on the quantity of goods that can be transported at once.
* **Store Capacities**: The maximum inventory each store can hold.
* **Demand Patterns**: Varying customer demands across stores.
* **Cost Structures**: Includes production costs, storage costs, transportation costs, and penalties for late deliveries or insufficient stock.

The environment operates over multiple time steps (e.g., 24 months), simulating the dynamic nature of supply chains, with the goal of testing agent performance under varying conditions.

# 5 Dataset Description

The supply chain environment itself generates synthetic datasets based on the following parameters:

* **Stores:** The number of stores involved and their respective storage capacities.
* **Truck Capacities:** The maximum load a truck can carry at one time.
* **Demand Patterns:** The fluctuation in customer demands for each store.
* **Costs:** These include production costs, transportation costs, storage costs, and penalties for unmet demand.
* **Rewards:** Agents are rewarded based on their ability to meet customer demands, minimize penalties, and optimize production and transportation costs.



# 6. Modules and Components

## 6.1 Core Modules

## Environment (SupplyDistribution)

This module manages state transitions, reward calculations, and the overall constraints of the supply chain environment. It models dynamic interactions like transportation limitations, store capacity limits, and variable demand.

## Agents

**S-Q Agent:** A simple heuristic-based agent that makes inventory management decisions based on predefined thresholds.

**Approximate SARSA:** This agent utilizes a linear function approximator to learn optimal policies in an on-policy manner.

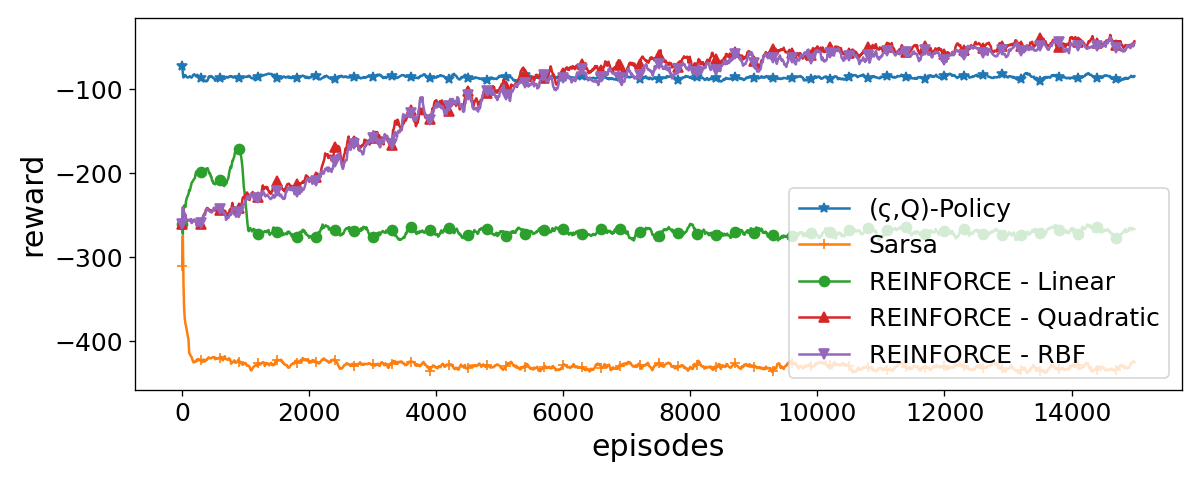
**REINFORCE Agent:** This policy-gradient agent optimizes decision-making by adjusting policy parameters to maximize cumulative rewards.

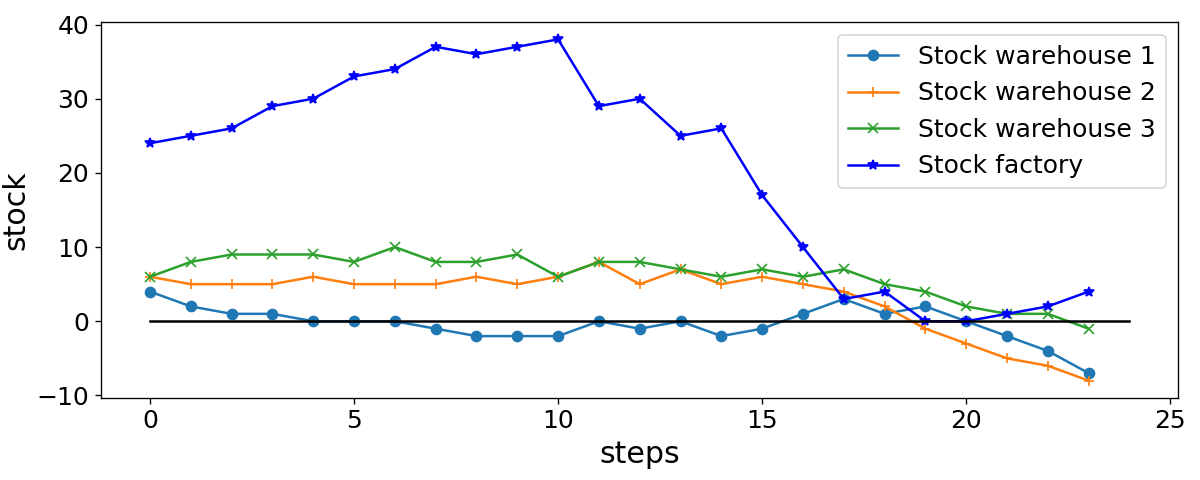
## Evaluation

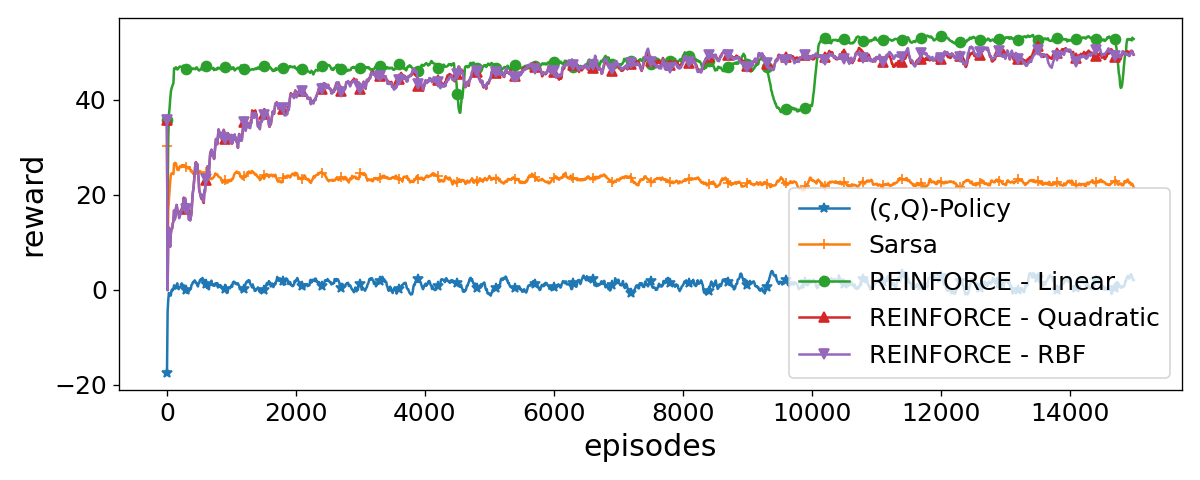
This module evaluates the performance of agents across different episodes by tracking rewards, operational efficiency, and convergence.

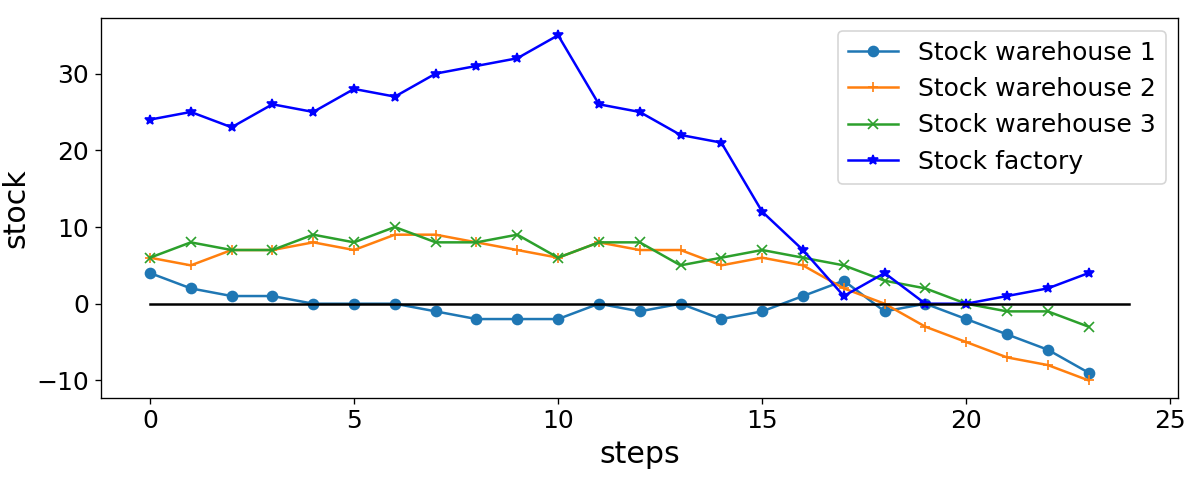
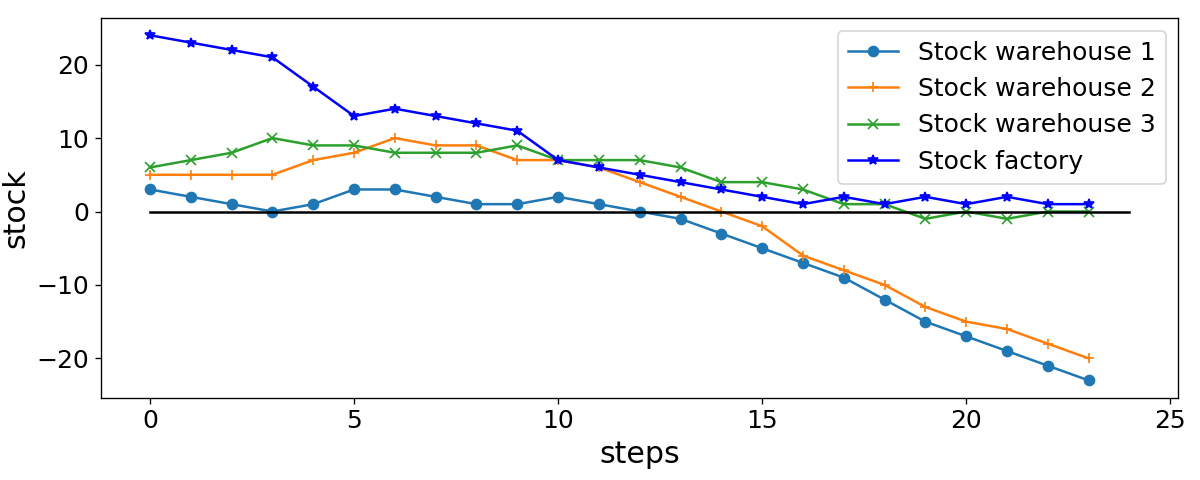
## **Visualization (GraphCreation)**

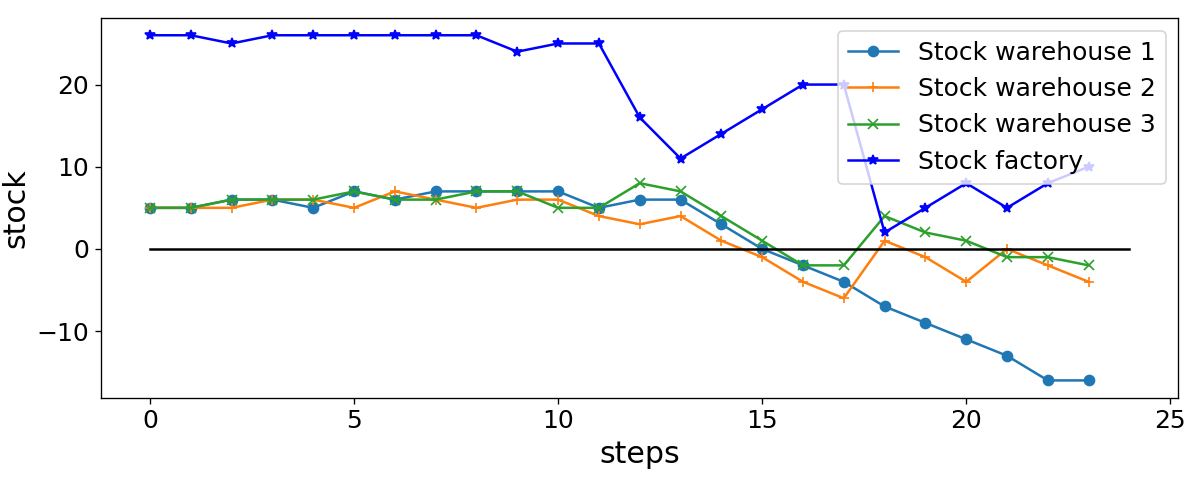
This module generates visual representations of the agent’s performance, providing insight into reward trajectories and policy convergence over time.

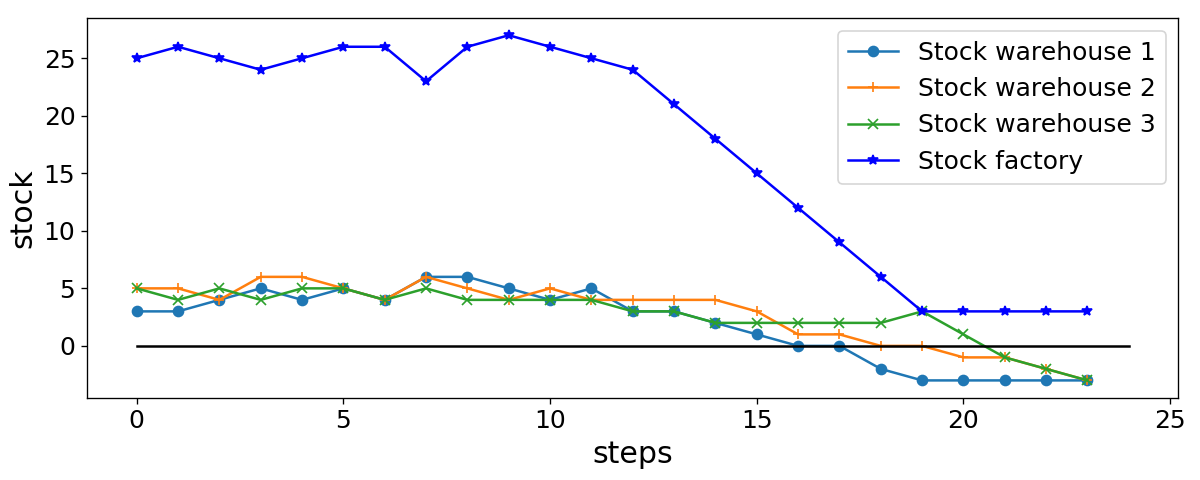












# 7. System Analysis

The system was designed to address several critical factors:

**Operational Efficiency**: The system strives to minimize costs while ensuring timely and efficient supply chain operations.

**Scalability:** The architecture is scalable, accommodating multiple stores and varying demand patterns.

**Real-World Constraints:** It takes into account storage limits, production constraints, and penalties, ensuring that the system reflects realistic supply chain dynamics.

# 8. System Design

## System Architecture

The architecture integrates the environment, agents, evaluation, and visualization components. The agents interact with the environment to learn policies and optimize decision-making.

## Module Design

**Environment Module:** Manages supply chain dynamics, state transitions, and reward mechanisms.

**Agent Module:** Implements RL algorithms and policy approximations to optimize decision-making.

**Evaluation Module:** Tracks agent performance and generates visualizations to aid in the evaluation process.

## Database Design

The system stores simulation logs, reward trajectories, and performance metrics in a structured format for further analysis and reporting.

# 9. Implementation

Key Implementation Details

**Environment:** The custom SupplyDistribution environment simulates multi-store supply chains, with varying truck capacities, storage limits, and fluctuating customer demands.

**Agents:**

* **S-Q Agent:** Operates on predefined inventory thresholds and reorders products when stock falls below a threshold.
* **SARSA Agent:** Utilizes linear function approximation for state-action values and updates them via the epsilon-greedy policy.
* **REINFORCE Agent:** Implements policy updates using policy gradients, with different basis functions (Linear, Quadratic, RBF) for flexibility.

**Evaluation**: Each agent’s performance is logged and analyzed based on cumulative rewards and efficiency across episodes.

**Results**: Sample outputs include reward trajectories and performance comparisons between agents. For example, in a medium-complexity environment, the REINFORCE agent outperformed the heuristic S-Q agent in terms of reward maximization.

# 10. Conclusions

## Design and Implementation Issues

Several challenges arose during the design and implementation phases, including:

**Balancing Exploration and Exploitation:** For SARSA, finding the right balance between exploration and exploitation proved challenging, especially in complex environments.

**Hyperparameter Tuning in REINFORCE:** Adjusting the learning rate and other parameters for REINFORCE to achieve optimal convergence required extensive experimentation.

## Advantages and Limitations

**Advantages:**

The RL agents provided adaptive decision-making in dynamic environments, making them suitable for real-time supply chain optimization.

The modular design allowed for easy integration of different agents and environments, facilitating experimentation and improvement.

**Limitations:**

The computational intensity of training these agents can become a bottleneck in large-scale environments.

Hyperparameter tuning, especially in policy-gradient methods like REINFORCE, remains a non-trivial task.

## Future Enhancements

**Real-World Data Integration:** Future work could include the integration of real-world supply chain data to validate the system's performance.

**Advanced Algorithms**: The application of more advanced algorithms like Deep Q-Networks (DQN) and Proximal Policy Optimization (PPO) could further enhance the system’s ability to handle complex scenarios.

**Real-Time Deployment:** Extending the system to enable real-time deployment in actual supply chain operations is an exciting future direction.

# 11. Team Responsibilities

Team Members and Their Responsibilities:

## Himanshu Salvekar - Data Analysis & Visualization

**Responsibilities**:

* Lead the data analysis phase by examining and preparing synthetic datasets for training and evaluation.
* Implement the performance evaluation module, tracking agent rewards and efficiency.
* Design and develop the visualization tools to graph agent performance and decision-making outcomes.
* Oversee the final result presentation for performance metrics across different agents.

## Sayantan Ray - Algorithm Design & Implementation

**Responsibilities:**

* Design and implement the core reinforcement learning algorithms, including SARSA and REINFORCE.
* Develop the custom threshold-based agent (S-Q agent) for baseline comparison.
* Work on the modularization of the agent components, ensuring scalability and ease of testing.
* Conduct algorithm optimization, especially hyperparameter tuning, to enhance convergence and stability.
* Ensure the integration of agents with the environment, enabling smooth interactions and accurate reward calculations.

## Swarnasish Banerjee - Simulation Environment & System Integration

**Responsibilities:**

* Develop and implement the SupplyDistribution environment, simulating real-world supply chain dynamics.
* Manage system architecture and coordinate the interaction between modules (agents, environment, evaluation).
* Oversee the system testing phase, ensuring it handles all edge cases and scales efficiently.