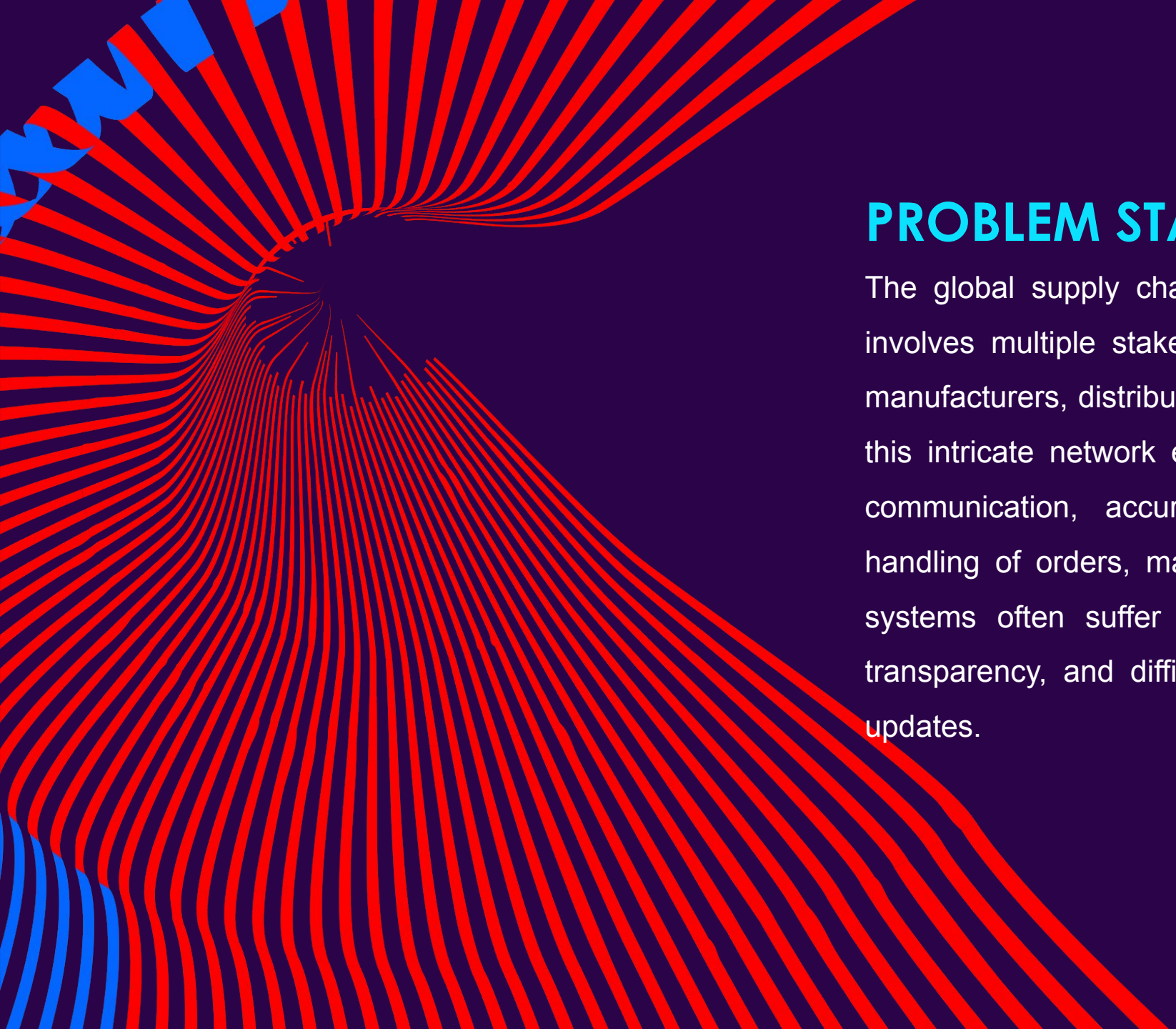
SUPPLY CHAIN MANAGEMENT SYSTEM

Fetch.ai

DESCRIPTION

Welcome to the presentation on our Supply Chain Management System. This project integrates cutting-edge technologies to automate and optimize every stage of the supply chain, from order placement to delivery. By leveraging a multi-agent system and robust smart contracts, we enhance efficiency, ensure transparency, and provide real-time tracking, revolutionizing how supply chains operate.



PROBLEM STATEMENT

The global supply chain process is complex and involves multiple stakeholders including suppliers, manufacturers, distributors, and retailers. Managing this intricate network efficiently requires seamless communication, accurate tracking, and reliable handling of orders, materials, and goods. Current systems often suffer from inefficiencies, lack of transparency, and difficulty in managing real-time updates.

OBJECTIVE

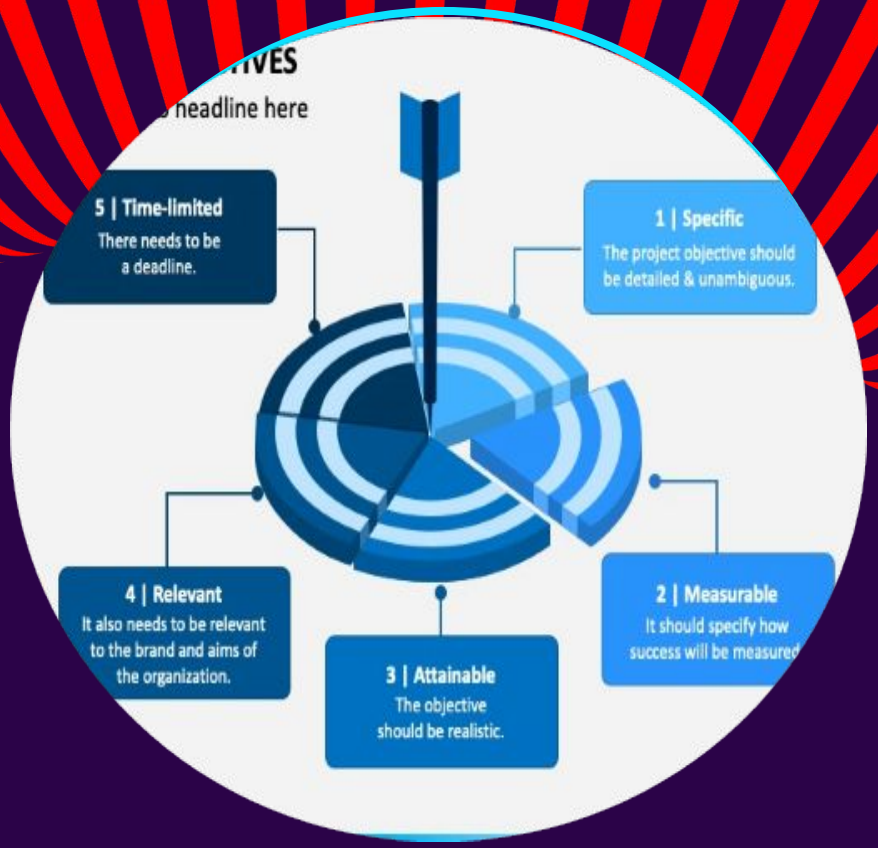
Develop an integrated Supply Chain Management System that automates the entire process from order placement to final delivery, providing a transparent, efficient, and reliable solution. The system should facilitate:

> Order Management : Allow users to place and track orders through a web interface.

> Material Handling : Automate the process of sending and receiving materials between agents (Supplier, Manufacturer, Distributor, Retailer).

> Goods Manufacturing and Distribution : Manage the production and distribution of goods through automated workflows.

> Status Tracking : Provide real-time status updates for orders, materials, and goods at each stage of the supply chain.



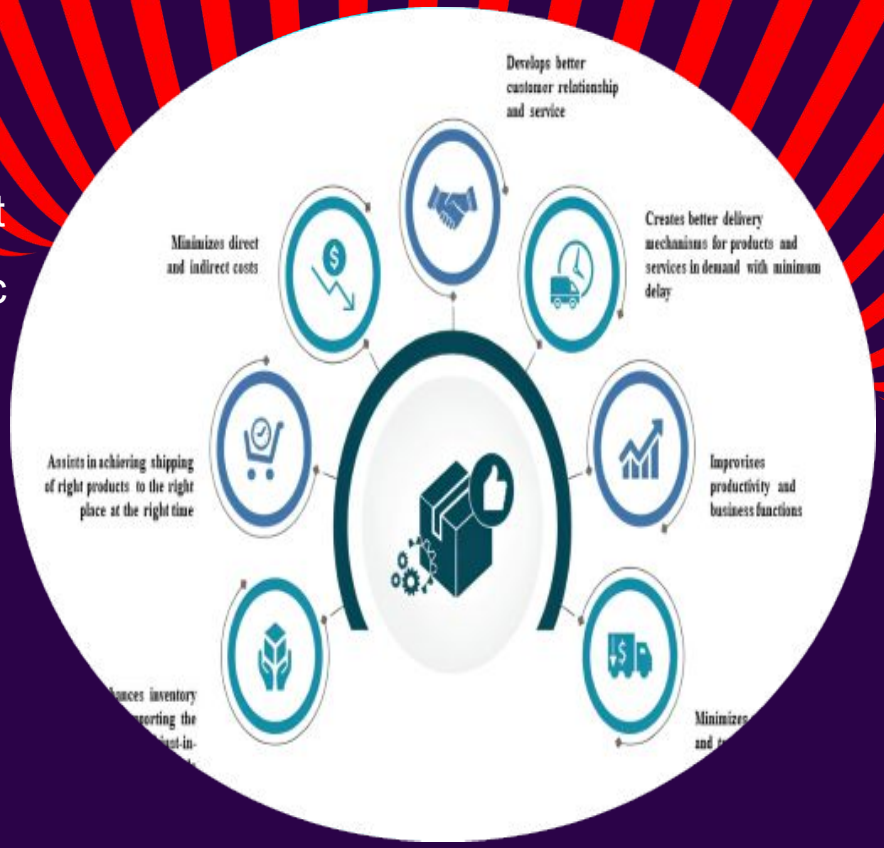
Solution Overview

Backend Implementation : Use a multi-agent architecture to represent different entities in the supply chain, with each agent handling specific tasks and communicating via smart contracts.

Frontend Implementation : Develop a responsive web application using React to interact with the backend, enabling users to place orders, track progress, and view real-time status updates.

Integration : Ensure smooth interaction between frontend and backend, with robust error handling and logging to maintain system reliability.

This suggested solution can be scalable, user-friendly, and capable of handling real-time operations to enhance overall supply chain efficiency.



TECH STACK: FRONT-END

React : For building interactive UIs.

Axios : For handling HTTP requests.

CSS : For styling the application.

Chart.js : For visualizing data with charts.

React-Bootstrap : For responsive UI components.

React Icons : For adding icons to the UI.

Flask-CORS : For enabling API communication across different origins.

TECH STACK: BACK-END

Python : Primary language for backend logic.

Flask : Web framework for handling HTTP requests.

uagents : For agent-based communication and orchestration.

Fetch.AI : For integrating smart contracts and ledger functionalities.

Requests : For making HTTP requests between services.

Logging : For detailed logging and debugging.

Flask-CORS : For enabling cross-origin requests.

FRONTEND IMPLEMENTATION

Overview :

The frontend leverages React to offer a dynamic and intuitive UI for managing the supply chain. It features a modern, glass morphism design to enhance visual appeal and user experience.

Key Features :

> Dashboard : Central hub for managing and monitoring supply chain processes.

> Order Placement : Form for item and quantity input.

> Material Handling : Buttons for sending materials, manufacturing goods, and confirming orders.

> Status Updates : Real-time status and order history display.

Design Highlights :

> Glass Morphism : Elegant glass-like UI elements with frosted glass effects for a modern look.

> Responsive Layout : Adaptable design for different screen sizes, ensuring usability across devices.

UI/UX

User Interface

REACT.JS COMPONENT

Modular components for different functionalities (order placement, status checking).

MATERIAL UI INTEGRATION

Consistent styling with pre-designed components for buttons, input fields, and containers.

GLASS-MORPHISM

Achieved using CSS, creating a frosted glass effect for a modern and visually appealing look.

UI/UX

User Experience

RESPONSIVE DESIGN

Ensures compatibility with different devices and screen sizes.

INTERACTIVE ELEMENTS

Includes buttons, input fields, and dynamic charts for enhanced user interaction.

CHARTS AND GRAPHS

Interactive charts for visualizing data (e.g., order history, material status).

BACKEND IMPLEMENTATION

Overview :

The backend is structured to handle the core operations of the supply chain management system using Flask for API services and uagents for agent-based processing. It ensures seamless interaction between agents and handles critical supply chain tasks.

Key Features :

> API Endpoints : Manage order placement, material handling, manufacturing, distribution, and status updates.

> Agent Communication : Facilitates communication between Supplier, Manufacturer, Distributor, and Retailer agents.

> Contract Management : Simulates contract interactions to manage and query supply chain states.

BACKEND IMPLEMENTATION

Design Highlights :

> Flask Framework : Provides a robust and scalable API layer.

> uagents Framework : Manages agent-based operations and interactions.

> Logging & Error Handling : Comprehensive logging for debugging and error tracking.

APPROACH AND METHODOLOGY

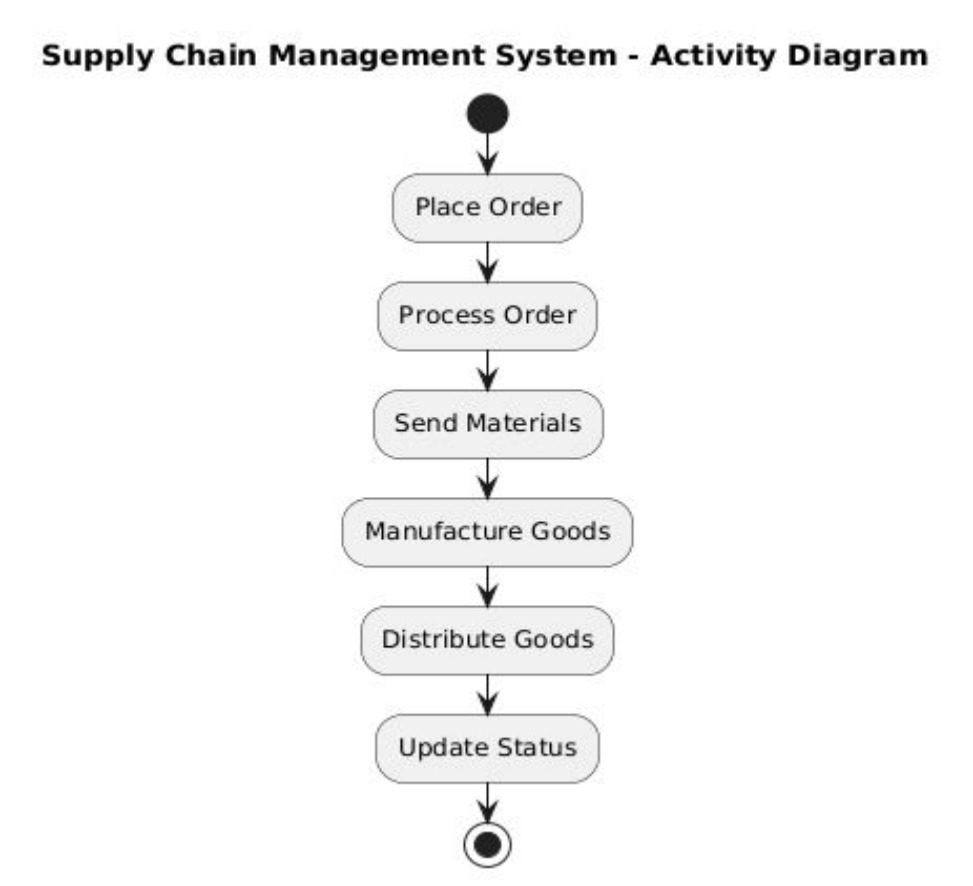
Development Phases

- Phase 1 : Requirement Analysis and Design
- Phase 2 : Backend Development with Uagents
- Phase 3 : Frontend Development with React and Interactive Elements
- Phase 4 : Integration and Testing

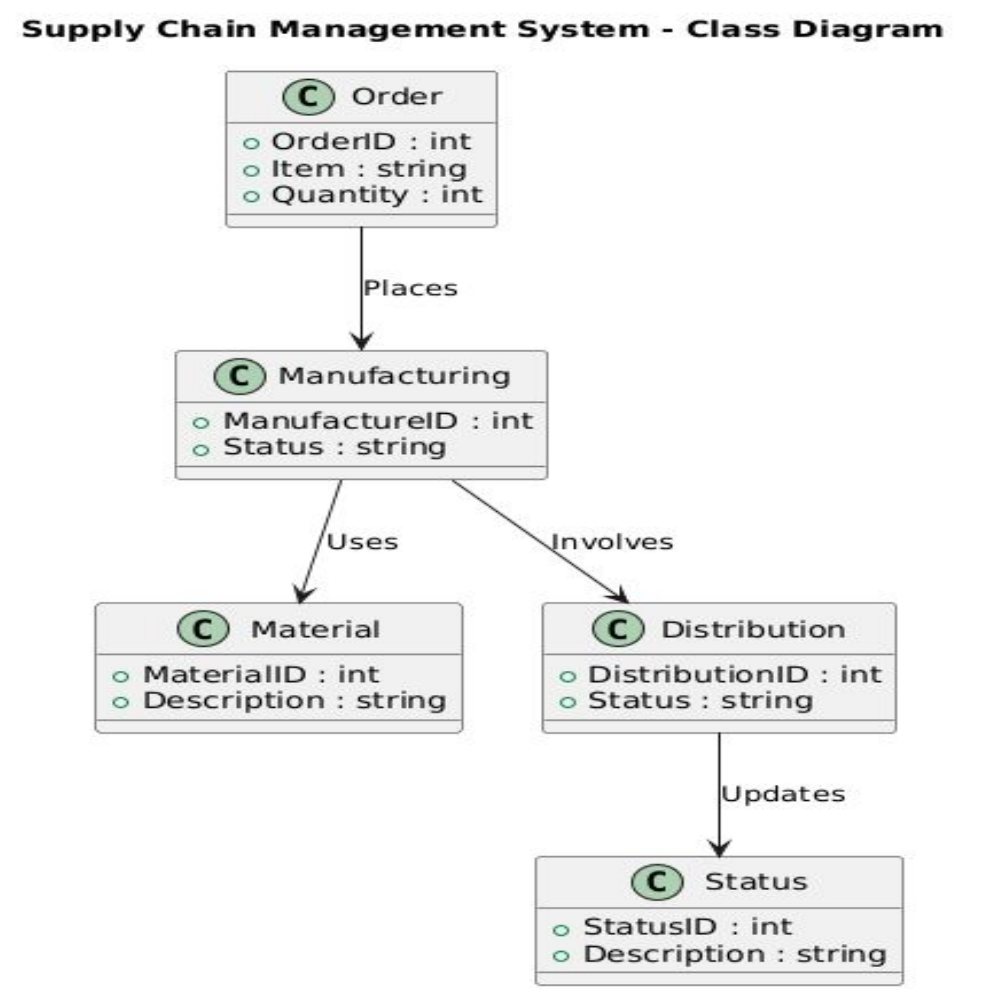
Integration : Ensured seamless communication between frontend and backend through well-defined API endpoints.

Testing : Comprehensive testing of all functionalities to ensure reliability and performance.

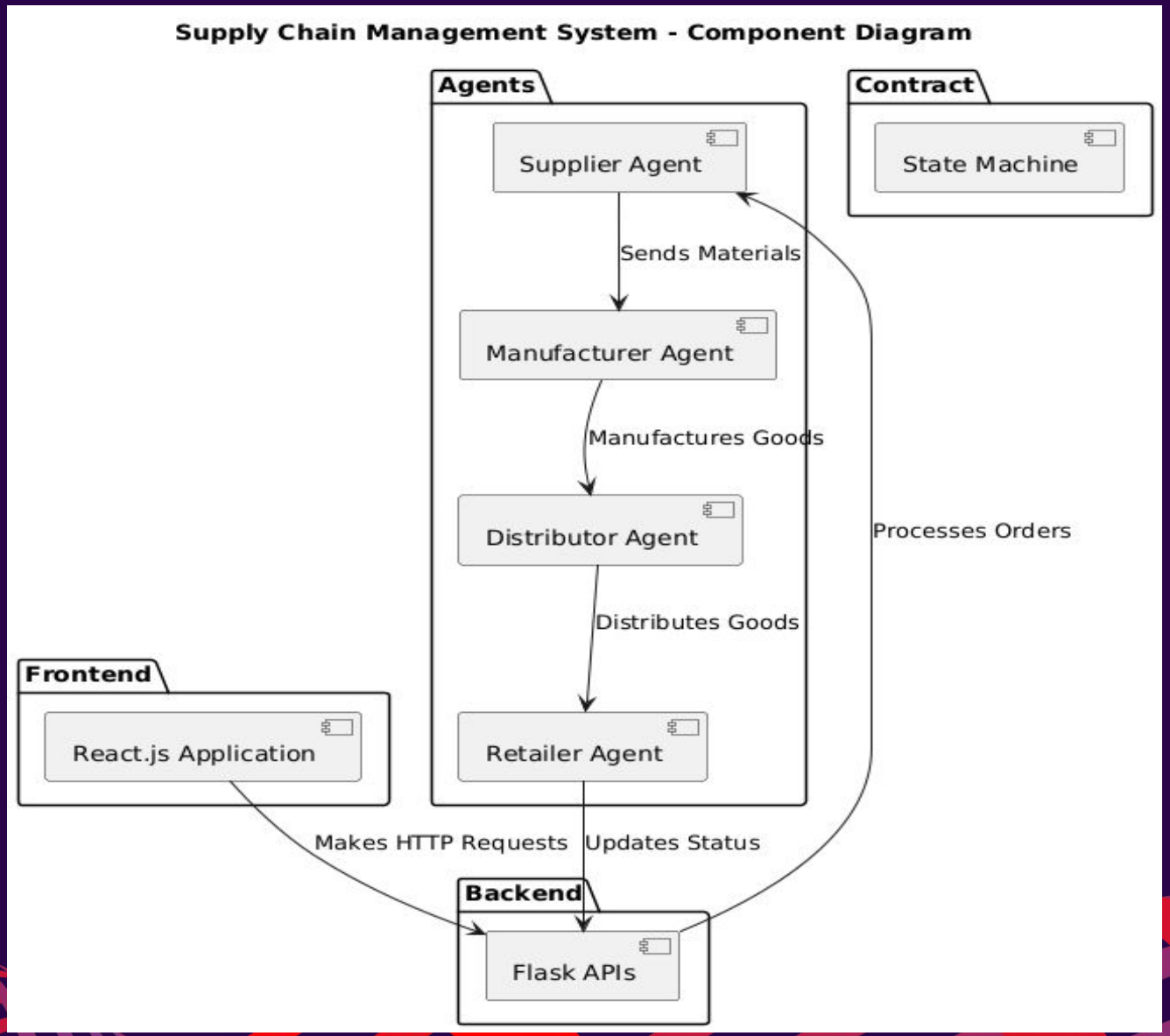
ACTIVITY DIAGRAM



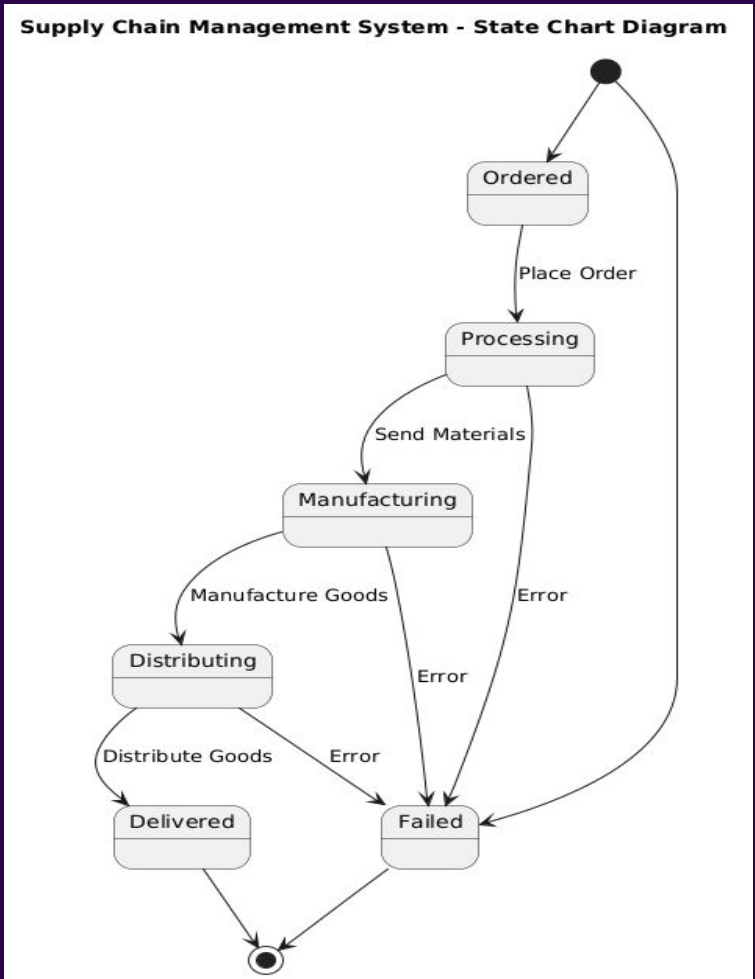
CLASS DIAGRAM



COMPONENT DIAGRAM



STATE TRANSITION DIAGRAM




```
PlantUML 1.2024.6
[From string (line 14) ]

@startuml
title Supply Chain Management System - Sequence Diagram

actor User
participant "React.js Application" as Frontend
participant "Flask APIs" as Backend
participant "Supplier Agent"
participant "Manufacturer Agent"
participant "Distributor Agent"
participant "Retailer Agent"

User -> Frontend : Place Order
Frontend -> Backend : POST /order
Backend -> Supplier Agent : Process Order
 Syntax Error?
```

FEATURES

ORDER MANAGEMENT

Users can place orders through the frontend interface. The order details are sent to the backend, processed, and stored.

- Order history is displayed, showing item names, quantities, and statuses.

MATERIAL HANDLING

Automated processes for sending and receiving materials between agents, enhancing efficiency and reducing manual effort.

FEATURES

MANUFACTURING & DISTRIBUTION

Agents manage the production and distribution of goods, ensuring timely fulfillment and tracking.

REAL-TIME STATUS

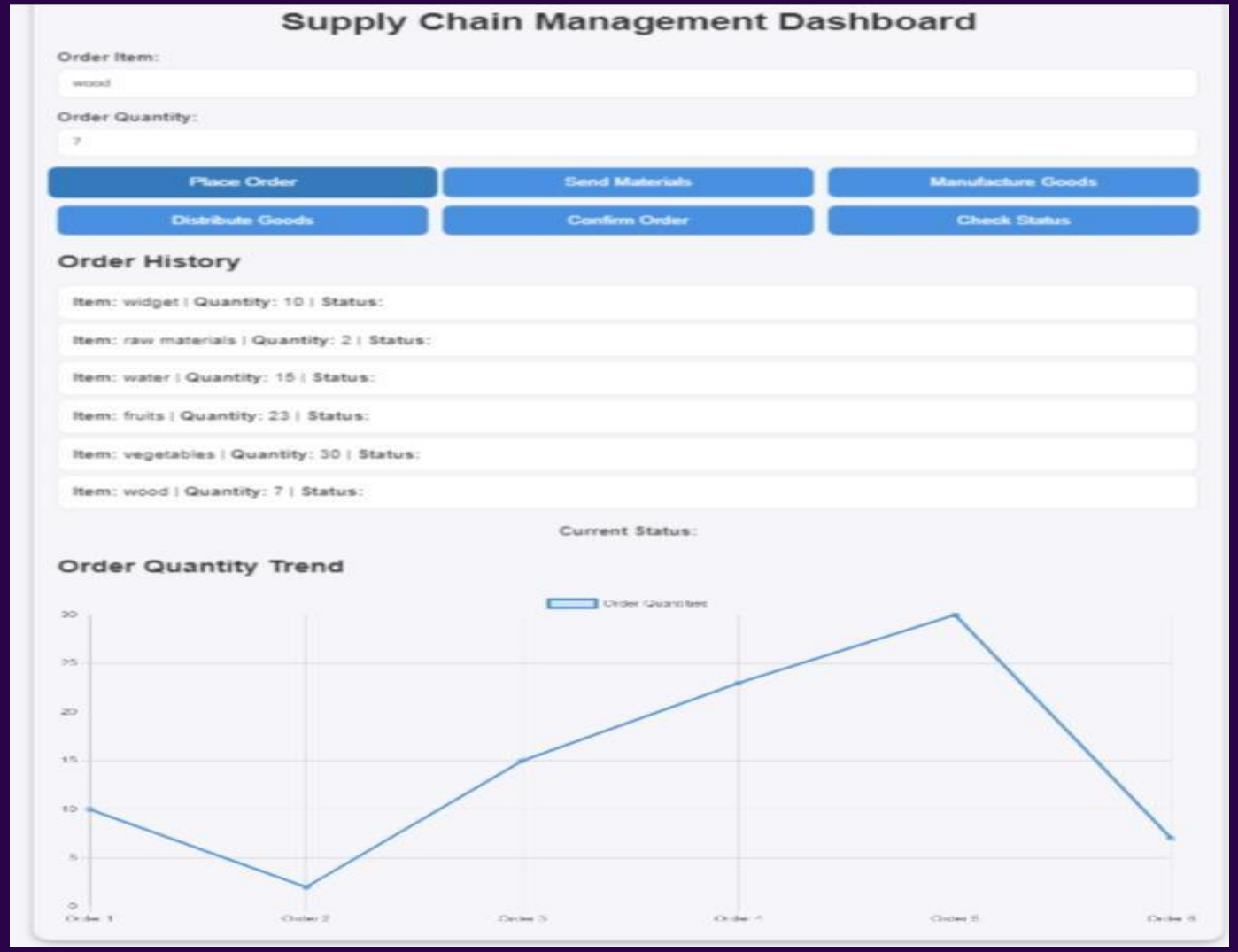
Provides real-time updates on order status, material handling, and other supply chain processes.

OUR COMPETITION GRAPH

CHALLENGES FACED	SOLUTIONS IMPLEMENTED
Difficulties in synchronizing frontend and backend components.	Resolved integration issues by refining API endpoints and frontend logic.
Ensuring robust error handling and logging for smooth operation.	Implemented comprehensive error handling and logging mechanisms to track and address issues effectively.

IMPLEMENTATION

https://fetchai-dapp.netlify.app/



RESULTS & IMPACT

Outcomes :

- Efficient Operations : Streamlined order processing, material flow, and distribution.
- Enhanced Transparency : Real-time updates and clear supply chain visibility.
- Improved Coordination : Seamless interaction among Supplier, Manufacturer, Distributor, and Retailer.

Metrics :

- Reduced Processing Time : Orders handled in minutes, not hours.
- Lower Errors : Minimized manual mistakes through automation.
- Optimized User Experience : Simplified interface for effective supply chain management.

RESULTS & IMPACT

Impact :

- Cost Efficiency : Automated processes reduce operational costs.
- Scalability : Designed for future growth and additional features.

Future Opportunities :

- Advanced Analytics : Potential for predictive insights and integration with other systems.

Visuals :

- Graphs & Charts : Processing time, error rates.
- Screenshots : Key features and user interface.

FUTURE WORK

ADVANCED ANALYTICS

Develop analytics and reporting features for deeper insights into supply chain performance.

System Scalability

Scale the system for larger and more complex supply chains.

Extended Integration

Integrate with additional supply chain partners and platforms for broader applicability.

Blockchain Integration

Explore blockchain technology for added security and decentralization.

CONCLUSION

Summary :

Recap of the project’s objectives, approach, and outcomes. Emphasize key achievements and improvements.

Acknowledgments :

Thank collaborators and Fetch.ai for their support and the opportunity to present this project.

TEAM : UnHoly

M BUDDU UDAY

GitHub:

https://github.com/Morampudi-Buddu-Uday

M POOJITH CHOWDARY

GitHub:

https://github.com/Poojith-Chowdary

N MAHIDEEP

GitHub:

https://github.com/matrix-09

VVS LAXMAN

GitHub:

https://github.com/Vvslaxman

THANK YOU