Real-Time Supply Chain Visibility Platform Implementation

Yash Bhuse

Project Name: Real-Time Supply Chain Visibility Platform Implementation

Project Objective: To implement a centralized, real-time supply chain visibility platform that enhances tracking of shipments, optimizes inventory management, reduces operational costs, and improves decision-making across the supply chain network.

Project Justification: The company is experiencing inefficiencies due to a lack of transparency in the supply chain, leading to delayed shipments, increased costs, and customer dissatisfaction. A real-time visibility platform will address these issues by providing accurate, timely information, thereby improving operational efficiency and competitive advantage.

Business Problem: The existing supply chain lacks end-to-end visibility, resulting in:

- Delayed shipments due to unforeseen disruptions.
- Inaccurate inventory levels causing stockouts or overstocking.
- Manual tracking processes leading to errors and inefficiencies.
- Inability to make proactive decisions due to outdated information.
- Poor communication between suppliers, carriers, and internal teams.

AS-IS State:

- **Processes:** Manual tracking of shipments and inventory using spreadsheets and emails.
- **Technology:** Disparate systems with limited integration; no centralized platform.
- **Data Management:** Inconsistent data formats; delayed data updates.

- **Communication:** Reactive communication with suppliers and carriers; siloed information.
- **Performance:** High operational costs; low customer satisfaction due to delays.

To-Be State:

- **Processes:** Automated tracking and inventory management with real-time updates.
- **Technology:** Centralized platform integrating all supply chain systems.
- **Data Management:** Standardized data formats; real-time data analytics.
- **Communication:** Proactive collaboration with all stakeholders via the platform.
- **Performance:** Reduced costs; improved customer satisfaction; data-driven decisions.

Project Stakeholders:

- **Project Sponsor:** Chief Operations Officer (COO)
- **Business Analyst**

- **Project Manager**
- **IT Department**
- **Supply Chain Management Team**
- **Warehouse Managers**
- **Suppliers and Carriers**
- **Sales and Customer Service Teams**
- **Customers**
- **Regulatory Compliance Team**

RACI Matrix:

Activity	Responsible	Accountable	Consulted	Informed
Requirements Gathering	Business Analyst	Project Manager	Supply Chain Team, IT Dept.	Stakeholders
System Design	IT Dept.	Project Manager	Business Analyst	Stakeholders
Vendor Selection	Business Analyst	Project Sponsor	IT Dept., Procurement	Stakeholders

Integration Development	IT Dept.	Project Manager	Suppliers, Carriers	Stakeholders
Testing and QA	IT Dept.	Project Manager	Business Analyst	Stakeholders
Training and Deployment	Business Analyst	Project Manager	Supply Chain Team	All Users
Post-Implementation Support	IT Dept.	Project Manager	Business Analyst	Stakeholders

Project In Scope Use Case:

- Implementing a platform for real-time tracking of shipments and inventory.
- Integration with suppliers' and carriers' systems for seamless data exchange.
- Providing dashboards and reports for supply chain analytics.

- Setting up alerts and notifications for delays or inventory issues.
- Training users on the new platform functionalities.

Project Out of Scope Use Case:

- Redesigning physical logistics networks.
- Changing suppliers or carriers.
- Overhauling the existing ERP system.
- Developing in-house hardware solutions.

Business Requirements:

1. **BR1:** The platform shall provide end-to-end visibility of the supply chain.
2. **BR2:** It shall integrate with existing internal systems and external partners' systems.
3. **BR3:** The system shall improve inventory accuracy to 99%.
4. **BR4:** It shall reduce shipment delays by 30% within the first year.

5. **BR5:** The platform shall enhance customer satisfaction ratings by providing accurate delivery estimates.

Functional Requirements:

1. **FR1:** The system must allow users to track shipments in real-time.
2. **FR2:** It must automatically update inventory levels upon shipment receipt.
3. **FR3:** The platform must generate alerts for delays, stockouts, and other exceptions.
4. **FR4:** It must provide role-based access control to protect sensitive data.
5. **FR5:** The system must generate customizable reports and analytics dashboards.

Non-Functional Requirements:

1. **NFR1:** The platform must be accessible 24/7 with 99.9% uptime.
2. **NFR2:** Data updates must reflect in the system within 2 minutes of occurrence.

3. **NFR3:** The system must comply with industry data security standards (e.g., ISO 27001).
4. **NFR4:** The platform should support up to 500 concurrent users without performance degradation.
5. **NFR5:** It must be user-friendly to minimize training requirements.

API Requirements:

1. **API1:** The system must provide RESTful APIs for data exchange with external systems.
2. **API2:** APIs must support secure authentication (e.g., OAuth 2.0).
3. **API3:** The platform must handle API requests and responses in JSON format.
4. **API4:** APIs should support batch data processing for bulk updates.

Integration Requirements:

1. **IR1:** Integration with suppliers' order management systems for purchase orders.

2. **IR2:** Integration with carriers' tracking systems for shipment status updates.
3. **IR3:** Integration with internal ERP and WMS for inventory synchronization.
4. **IR4:** Support for EDI standards to communicate with partners lacking API capabilities.

Database Requirements:

1. **DB1:** The database must store transactional data for at least 5 years.
2. **DB2:** It must support real-time data replication and backups.
3. **DB3:** The database should enforce data integrity and referential constraints.
4. **DB4:** It must be scalable to accommodate data growth over the next 10 years.

Transition Requirements:

1. **TR1:** Develop a data migration plan from legacy systems to the new platform.

2. **TR2:** Provide comprehensive training programs for all user groups.
3. **TR3:** Establish a support system for issues post-implementation.
4. **TR4:** Phase out old systems gradually to ensure business continuity.

Data Dictionary:

- **ShipmentID:** Unique identifier for each shipment.
- **OrderID:** Unique identifier for each customer order.
- **ProductID:** Unique identifier for each product.
- **LocationCode:** Code representing warehouse or distribution center.
- **Status:** Current status of the shipment (e.g., In Transit, Delivered).
- **ETA:** Estimated Time of Arrival for shipments.
- **InventoryLevel:** Current stock level of a product at a specific location.

Project Risks:

1. **Risk1:** Delays in system integration due to incompatible technologies.
    a. *Mitigation:* Conduct a thorough technical assessment during planning.
2. **Risk2:** Data security breaches during data transmission.
    a. *Mitigation:* Implement robust encryption and security protocols.
3. **Risk3:** Resistance to change from employees.
    a. *Mitigation:* Engage stakeholders early and provide change management support.
4. **Risk4:** Budget overruns due to unforeseen technical challenges.
    a. *Mitigation:* Include contingency funds in the budget.

Project Dependencies:

1. **Dependency1:** Availability of APIs from suppliers and carriers for integration.
2. **Dependency2:** Completion of internal system upgrades prior to integration.

3. **Dependency3:** Timely procurement of necessary hardware and software.
4. **Dependency4:** Regulatory approvals for data sharing with external partners.

Project Issues:

1. **Issue1:** Inconsistent data formats received from different partners.
    a. *Resolution:* Implement a data standardization module in the platform.
2. **Issue2:** Limited technical expertise within the team for new technologies.
    a. *Resolution:* Hire external consultants or provide training to the team.
3. **Issue3:** Network latency affecting real-time data updates.
    a. *Resolution:* Upgrade network infrastructure and optimize data transmission protocols.

Project Constraints:

1. **Constraint1:** The project must be completed within 12 months.
2. **Constraint2:** The budget is capped at $2 million.
3. **Constraint3:** The platform must comply with international trade regulations.

Project Assumptions:

1. **Assumption1:** All external partners are willing to participate in system integration.
2. **Assumption2:** No major changes in regulatory requirements during the project timeline.
3. **Assumption3:** Necessary resources (personnel, equipment) will be available as scheduled.

Glossary:

- **API (Application Programming Interface):** A set of protocols for building software applications.
- **EDI (Electronic Data Interchange):** The computer-to-computer exchange of business documents.

- **ERP (Enterprise Resource Planning):** Business process management software.
- **WMS (Warehouse Management System):** Software applications that support warehouse operations.
- **IoT (Internet of Things):** Network of physical objects embedded with sensors and software.
- **RFID (Radio-Frequency Identification):** Technology to encode digital data in RFID tags.
- **Supply Chain Visibility:** The ability to track products in transit from manufacturer to end-user.

GAP ANALYSIS REPORT

Current State (AS-IS):

- **Process:** The supply chain is currently managed through manual tracking methods using spreadsheets and emails, leading to inefficiencies, errors, and delays in communication.
- **Technology:** There is no centralized system to track shipments and inventory. Various supply chain partners (suppliers, carriers, etc.) use different systems, making data consolidation challenging.
- **Data Management:** Data is updated manually and in disparate formats, leading to inconsistent inventory records and shipment statuses.
- **Communication:** Communication between internal teams and external partners (suppliers, carriers) is reactive, often after delays or issues have already occurred.
- **Performance Metrics:** High operational costs, low customer satisfaction, and delays in shipments due to lack of visibility and proactive planning.

Future State (TO-BE):

- **Process:** The supply chain will be managed through an automated, real-time visibility platform that tracks all shipments and inventory levels. Notifications and alerts will trigger proactive action.
- **Technology:** A centralized platform will integrate with internal systems (ERP, WMS) and external partners' systems (suppliers and carriers), providing a single source of truth for supply chain data.
- **Data Management:** Data will be updated automatically in real-time, providing accurate, timely information on shipment statuses, inventory levels, and estimated delivery times.
- **Communication:** Communication with external partners will be proactive, with the system providing real-time notifications for delays, stockouts, and other issues. Collaborative workflows will improve coordination.

- **Performance Metrics:** Reduced operational costs, improved customer satisfaction, on-time delivery, and more efficient inventory management with fewer stockouts or overstocks.

Gap Identified:

1. **Manual Processes vs. Automation:**
    a. **Gap:** The current manual processes for shipment tracking and inventory updates result in errors and delays.
    b. **Impact:** Inefficiency, increased operational costs, and reactive issue resolution.
2. **Disparate Systems vs. Centralized Platform:**
    a. **Gap:** There is no unified platform to consolidate data from multiple systems used by various supply chain partners.
    b. **Impact:** Lack of transparency across the supply chain, making decision-making slow and inaccurate.
3. **Inconsistent Data Formats vs. Standardized Real-Time Data:**

a. **Gap:** Data from different partners is inconsistent and updated manually, causing errors in inventory and shipment records.
b. **Impact:** Inventory inaccuracies and shipment delays lead to customer dissatisfaction.

4. **Reactive Communication vs. Proactive Alerts:**
   a. **Gap:** Current communication with suppliers and carriers is reactive, addressing issues only after they occur.
   b. **Impact:** Delayed issue resolution and missed opportunities to mitigate supply chain disruptions proactively.
5. **Limited Visibility vs. End-to-End Real-Time Visibility:**
   a. **Gap:** Lack of end-to-end visibility prevents the company from accurately tracking shipment status and inventory levels.
   b. **Impact:** Inability to optimize inventory, increased stockouts or overstocking, and missed customer delivery deadlines.

Steps to Cover the Gap:

1. **Automate Shipment and Inventory Management:**
    a. **Action:** Implement the real-time visibility platform to automate shipment tracking and inventory updates.
    b. **Benefit:** Reduce manual errors, improve efficiency, and enable proactive supply chain management.
2. **Implement Centralized Platform for Integration:**
    a. **Action:** Develop a centralized platform that integrates with internal systems (ERP, WMS) and external partners' systems (suppliers, carriers) via APIs or EDI.
    b. **Benefit:** Achieve seamless data consolidation, improve transparency, and facilitate accurate decision-making.
3. **Standardize Data Formats and Ensure Real-Time Updates:**
    a. **Action:** Define and enforce standard data formats (e.g., JSON or EDI standards) and set

up real-time data exchange through APIs with all partners.

b. **Benefit:** Ensure data consistency and accuracy, improving shipment tracking and inventory management.

4. **Set Up Proactive Communication through Alerts and Notifications:**
    a. **Action:** Configure the platform to generate automated alerts for shipment delays, inventory stockouts, and other exceptions.
    b. **Benefit:** Enable proactive issue resolution, reducing operational disruptions and improving customer satisfaction.
5. **Enhance Supply Chain Visibility with Real-Time Dashboards:**
    a. **Action:** Develop real-time dashboards and analytics to provide end-to-end visibility into shipments, inventory, and supply chain performance.
    b. **Benefit:** Improve operational efficiency by allowing better forecasting, planning, and inventory management.

SIPOC ANALYSIS

The SIPOC analysis outlines the key elements of the real-time supply chain visibility project. It helps to identify who the suppliers and customers are, what inputs are needed to execute the project, and the processes involved in delivering outputs that benefit both internal and external customers.

1. **Supplier (S):

- **Internal Suppliers:**
  - IT Department (providing technical infrastructure and integration support)
  - Business Analysts (gathering requirements and defining business processes)
  - Project Management Office (overseeing project timelines and budget)
- **External Suppliers:**
  - Suppliers and Carriers (providing shipment data and inventory information)
  - Platform Vendor (providing the real-time visibility software and support)
  - Regulatory Authorities (supplying compliance standards)

2. **Inputs (I):

- **From Internal Suppliers:**
  - Current shipment and inventory data (from legacy systems)
  - Business requirements and functional specifications
  - Project management documentation (timelines, milestones)
- **From External Suppliers:**
  - Real-time shipment updates from carriers
  - Purchase order and inventory data from suppliers
  - API or EDI documentation for system integration
  - Compliance standards (data security, privacy requirements)

3. **Process (P):

1. Gather business requirements from all stakeholders.

2. Analyze current supply chain processes and identify inefficiencies.
3. Design and configure the real-time supply chain visibility platform.
4. Integrate the platform with internal systems (ERP, WMS) and external systems (suppliers, carriers).
5. Test the platform for data accuracy, real-time updates, and performance.
6. Provide training to users on how to operate the new system.
7. Deploy the platform and monitor performance, ensuring real-time visibility.
8. Continuously improve and optimize the system based on user feedback and performance metrics.

4. **Outputs (O):

- Automated, real-time updates of shipment and inventory data.
- Proactive alerts and notifications for delays, stockouts, and other supply chain exceptions.

- Real-time dashboards and analytics for supply chain performance.
- Improved coordination between internal teams, suppliers, and carriers.
- Enhanced decision-making based on accurate, real-time data.

5. **Customers (C):

- **Internal Customers:**
  - Supply Chain Management Team (using real-time data for decision-making)
  - Warehouse Managers (managing inventory efficiently)
  - Sales and Customer Service Teams (providing accurate delivery information to customers)
  - Finance Team (using data for cost optimization)
- **External Customers:**
  - End Customers (receiving timely and accurate delivery information)

- Suppliers and Carriers (collaborating through integrated systems for smoother operations)
- Regulatory Authorities (receiving compliance reports)

ROOT CAUSE ANALYSIS

Problem:

Despite the implementation of the real-time supply chain visibility platform, users report that shipment tracking updates are delayed by several hours, causing operational inefficiencies, missed delivery deadlines, and customer dissatisfaction.

Root Cause Analysis Using 5 Whys Technique:

1. Why is shipment tracking information delayed in the system?

- The system is receiving updates from external carriers only once every 6-8 hours.

2. Why are updates from external carriers being received infrequently?

- External carriers are sending batch updates instead of real-time updates via API integration.

3. Why are carriers sending batch updates rather than real-time updates?

- Some carriers’ legacy systems are not fully compatible with the API requirements of the new visibility platform, so they resort to batch processing.

4. Why are the carriers' systems incompatible with the API requirements?

- During the integration phase, not all carriers' system capabilities were assessed thoroughly, and their legacy limitations were overlooked.

5. Why were the carriers' system limitations overlooked during the integration phase?

- The initial requirements gathering and stakeholder engagement processes focused mainly on internal systems, without a comprehensive assessment of external partner systems, especially smaller carriers.

Root Cause:

The root cause of the shipment tracking delays is the incomplete assessment of external carriers' system capabilities during the integration phase. The real-time visibility platform’s API requirements were not compatible with all carriers' systems, leading to reliance on batch updates.

Detailed Root Cause Analysis (Ishikawa Diagram):

1. **People:**
    a. Lack of adequate collaboration with all external partners during the initial integration phase.
    b. Insufficient training or technical support provided to the carrier partners to adopt real-time API updates.
2. **Process:**
    a. Incomplete requirements gathering process, focusing primarily on internal systems and

neglecting thorough evaluation of external carrier systems.

b. No fallback mechanism or alternative process in place to handle carriers with legacy systems that are unable to provide real-time data.

3. Technology:

a. Some carriers operate legacy systems that do not support real-time API integration, leading to batch updates.

b. The platform's API design didn't include flexible configurations for carriers with varying technical capabilities.

4. Data:

a. Data exchange occurs in batch processing due to the incompatibility of API standards between the platform and certain carrier systems.

b. No data validation mechanisms to flag outdated or delayed information from carriers, resulting in operational inefficiencies.

5. **External Factors:**
    a. Limited technical resources and investment by smaller carriers to upgrade their systems for real-time API integration.
    b. Regulatory and compliance constraints might also restrict smaller carriers from upgrading their systems.

Steps to Address and Resolve the Problem:

1. **Conduct a Comprehensive Reassessment of Carrier Systems:**
    a. Re-engage all external partners (suppliers and carriers) to assess their system capabilities thoroughly.
    b. Identify which partners are using legacy systems that cannot support real-time API updates.
2. **Implement Flexible API Configurations:**
    a. Redesign the platform's API to support both real-time and batch processing for carriers unable to send real-time updates.

b. Ensure that the platform can differentiate between real-time and batch updates and flag delays to internal users.

3. **Provide Carriers with Technical Support and Solutions:**
    a. Offer carriers technical guidance and support to upgrade their systems for real-time API integration.
    b. Where upgrading is not feasible, work with the carriers to minimize batch update intervals (e.g., from 6-8 hours to 1-2 hours).
4. **Establish Fallback Mechanisms:**
    a. Implement fallback mechanisms, such as manual overrides or alternate data sources, for critical shipments where real-time updates are essential.
    b. Notify internal users when real-time updates are not possible and provide the latest available data with timestamp annotations.
5. **Enhance the Data Validation Process:**

a. Set up data validation rules within the platform to flag outdated or inconsistent information.
b. Provide alerts to users when delayed shipment updates are received, allowing for better operational planning.

6. **Improve Communication with External Partners:**
   a. Establish continuous communication channels with suppliers and carriers to ensure they remain aligned with system requirements and performance expectations.
   b. Set up periodic review meetings to address any ongoing integration issues and update partners on technological advancements.

The delay in shipment tracking updates is a result of incomplete system integration with external carriers, many of whom rely on legacy systems incompatible with the platform's API requirements. Addressing the root cause involves a comprehensive reassessment of external partner systems, flexible API configurations,

technical support, and improved communication with partners. By implementing these solutions, the platform can achieve real-time supply chain visibility across all partners and improve overall operational efficiency.

FUNCTIONAL DECOMPOSITION

Functional decomposition is used to break down the overall system and its requirements into smaller, manageable, and logical components, making it easier to analyze and implement the system effectively. Below is the functional decomposition of the **Real-Time Supply Chain Visibility Platform**.

Level 1: High-Level Function (Main System)

1. **Real-Time Supply Chain Visibility Platform**

Level 2: Major Functional Areas

1. **Shipment Tracking and Management**
2. **Inventory Management**
3. **Integration with External Systems (Suppliers/Carriers)**
4. **Data Processing and Management**
5. **Reporting and Analytics**
6. **Notifications and Alerts**
7. **User Management and Security**
8. **System Administration and Support**

Level 3: Breakdown of Each Functional Area

1. Shipment Tracking and Management

- **1.1. Shipment Data Collection**
  - Collect real-time shipment data from carriers.
  - Collect batch shipment data from carriers with legacy systems.
- **1.2. Update Shipment Status**
  - Automatically update shipment status (In Transit, Delivered, etc.).
  - Record Estimated Time of Arrival (ETA) for each shipment.
- **1.3. View Shipment Details**
  - Provide detailed shipment information (origin, destination, contents).
  - View history of shipment statuses and movements.
- **1.4. Delay and Exception Handling**
  - Identify and flag delayed shipments.
  - Generate exception reports for missed deadlines or other shipment issues.

2. Inventory Management

- **2.1. Inventory Data Collection**
  - Collect real-time inventory updates from warehouses and suppliers.
  - Ensure inventory levels are updated automatically after shipment receipts.
- **2.2. Monitor Inventory Levels**
  - Track current inventory levels for each product in each warehouse.
  - Set minimum and maximum thresholds for stock levels.
- **2.3. Reorder Management**
  - Trigger reorder notifications when stock reaches minimum thresholds.
  - Automate order creation with suppliers based on predefined rules.
- **2.4. Stock Allocation**
  - Allocate stock to customer orders based on real-time inventory levels.

3. Integration with External Systems (Suppliers/Carriers)

- **3.1. API Integration**
  - Enable real-time data exchange via API with external suppliers and carriers.
- **3.2. EDI Integration**
  - Support batch data exchange via Electronic Data Interchange (EDI) for partners with legacy systems.
- **3.3. Data Validation and Transformation**
  - Standardize data from external partners before feeding it into the system.
  - Validate and format data to ensure consistency and accuracy.

4. Data Processing and Management

- **4.1. Data Ingestion**
  - Ingest real-time and batch data into the system.
  - Process external data from APIs and EDI connections.
- **4.2. Data Storage**

    - Store shipment, inventory, and partner data in a central database.
    - Ensure data is stored in structured formats for easy retrieval.
- **4.3. Data Cleansing**
    - Automatically clean and validate data to remove duplicates and errors.
- **4.4. Historical Data Management**
    - Archive historical data for analysis and compliance purposes.
    - Provide users with access to past shipment and inventory data.

5. Reporting and Analytics

- **5.1. Generate Reports**
    - Generate predefined reports (e.g., shipment performance, inventory status).
    - Enable users to customize and create their own reports.
- **5.2. Real-Time Dashboards**

    - Provide real-time dashboards showing shipment status, inventory levels, and key performance indicators (KPIs).
- **5.3. Predictive Analytics**
    - Use predictive models to forecast shipment delays or inventory shortages.
- **5.4. Data Export**
    - Allow users to export data and reports in various formats (Excel, CSV, PDF).

6. Notifications and Alerts

- **6.1. Shipment Delay Alerts**
    - Send real-time alerts for delayed shipments.
    - Notify users of any discrepancies or exceptions in shipment tracking.
- **6.2. Stock Threshold Alerts**
    - Notify users when stock levels fall below the minimum threshold.
    - Trigger automatic reordering alerts.
- **6.3. System Alerts**
    - Send alerts for system issues, data validation errors, or downtime.

- Allow users to customize alert thresholds and types.

7. User Management and Security

- **7.1. User Authentication**
  - Implement multi-factor authentication (MFA) for system access.
  - Support single sign-on (SSO) integration.
- **7.2. Role-Based Access Control**
  - Define roles (Admin, User, Supervisor) and their access rights.
  - Restrict access to sensitive data based on user roles.
- **7.3. Audit Trails**
  - Maintain logs of user activity within the system.
  - Provide audit reports to monitor access and data changes.

8. System Administration and Support

- **8.1. System Configuration**

    - Allow administrators to configure system settings (e.g., thresholds, alert parameters).
- **8.2. Data Backup and Recovery**
    - Implement automated backups and data recovery solutions.
    - Ensure business continuity in case of system failure.
- **8.3. System Monitoring**
    - Continuously monitor system performance and health.
    - Alert administrators for any system failures or performance issues.
- **8.4. Post-Implementation Support**
    - Provide ongoing technical support and user training post-deployment.
    - Offer a helpdesk and knowledge base for troubleshooting.

This functional decomposition breaks down the overall functionality of the Real-Time Supply Chain Visibility Platform into manageable components. Each functional area is further broken down into its

respective features, making it easier to analyze and address specific requirements. This approach ensures that the platform's design and implementation address all necessary business functions while aligning with stakeholder needs.

BUSINESS REQUIREMENT DOCUMENT

Project Name:

Real-Time Supply Chain Visibility Platform Implementation

Project Objective:

To implement a centralized platform providing real-time visibility into shipments, inventory management, and supply chain operations to optimize decision-making, improve operational efficiency, reduce costs, and enhance customer satisfaction.

Project Justification:

The company’s current supply chain management processes are inefficient due to a lack of real-time visibility into shipments and inventory levels, leading to delays, increased costs, and customer dissatisfaction. A centralized platform will address these issues by enabling real-time data exchange,

improving communication, and enhancing supply chain transparency.

Business Problem:

The existing supply chain lacks real-time visibility, resulting in:

- Shipment delays due to unforeseen disruptions.
- Inaccurate inventory levels, leading to stockouts or overstocking.
- Poor communication between internal teams and external partners (suppliers and carriers).
- Inability to make data-driven, proactive decisions.
- High operational costs and low customer satisfaction due to reactive management.

AS-IS State:

- **Processes:** Manual tracking of shipments and inventory using spreadsheets and emails.

- **Technology:** Disparate systems with no centralized platform for supply chain visibility.
- **Data Management:** Inconsistent, manually updated data across systems.
- **Communication:** Siloed communication with external partners, often reactive.
- **Performance:** High costs and inefficiencies due to lack of visibility and real-time tracking.

TO-BE State:

- **Processes:** Automated tracking of shipments and inventory updates.
- **Technology:** A centralized real-time visibility platform integrated with internal and external systems.
- **Data Management:** Standardized, real-time data exchange across all systems.
- **Communication:** Proactive, real-time collaboration with external partners.

- **Performance:** Reduced operational costs, improved efficiency, and increased customer satisfaction through enhanced decision-making.

Project Stakeholders:

- **Project Sponsor:** Chief Operations Officer (COO)
- **Project Manager**
- **Business Analyst**
- **IT Department**
- **Supply Chain Team**
- **Warehouse Managers**
- **External Suppliers and Carriers**
- **Sales and Customer Service Teams**
- **Regulatory Authorities**

RACI Matrix:

Activity	Respo nsible	Accou ntable	Consulted	Infor med

Requirements Gathering	Business Analyst	Project Manager	Supply Chain, IT Dept.	Stakeholders
Platform Design	IT Dept.	Project Manager	Business Analyst, Vendors	Stakeholders
Vendor Selection	Business Analyst	Project Sponsor	IT Dept., Procurement	Stakeholders
Integration Development	IT Dept.	Project Manager	External Suppliers/Carriers	Stakeholders
Testing and QA	IT Dept.	Project Manager	Business Analyst	Stakeholders

User Training and Documentation	Business Analyst	Project Manager	Supply Chain Team	All Users
Post-Implementation Support	IT Dept.	Project Manager	Business Analyst	Stakeholders

Project In-Scope Use Case:

- Implementation of a platform for real-time tracking of shipments and inventory.
- Integration with suppliers' and carriers' systems for seamless data exchange.
- Real-time dashboards for monitoring supply chain performance.
- Notifications for shipment delays, inventory thresholds, and operational exceptions.
- User training and platform adoption support.

Project Out-of-Scope Use Case:

- Redesign of physical supply chain networks (e.g., logistics routes).
- Replacement of existing ERP or Warehouse Management Systems (WMS).
- Development of in-house hardware for tracking (RFID, GPS, etc.).
- Changes to supplier/carrier contracts.

Business Requirements:

1. **BR1:** The platform shall provide real-time visibility for shipments and inventory across all supply chain nodes.
2. **BR2:** It shall integrate with existing ERP and WMS systems as well as external carriers' systems.
3. **BR3:** The platform shall generate real-time alerts for shipment delays and inventory thresholds.
4. **BR4:** It shall improve inventory accuracy to 99% within the first 6 months.

5. **BR5:** The platform shall provide dashboards with key performance indicators (KPIs) for supply chain efficiency.

Functional Requirements:

1. **FR1:** The platform must allow users to track shipment status in real-time.
2. **FR2:** It must automatically update inventory levels based on real-time inputs.
3. **FR3:** The system shall support user roles with customizable access levels (Admin, User, Supervisor).
4. **FR4:** It must provide alerts and notifications for shipment delays, stockouts, and critical supply chain events.
5. **FR5:** The system must allow users to generate custom reports and export data in multiple formats (CSV, Excel, PDF).

Non-Functional Requirements:

1. **NFR1:** The platform must have 99.9% uptime to ensure business continuity.
2. **NFR2:** The system must process data updates in less than 2 minutes after they occur.
3. **NFR3:** The platform shall comply with industry-standard data security protocols (ISO 27001).
4. **NFR4:** It must support up to 1,000 concurrent users without performance degradation.
5. **NFR5:** The system should be designed to minimize user training time by having an intuitive user interface.

API Requirements:

1. **API1:** The platform must provide RESTful APIs for integration with external carrier systems.
2. **API2:** All APIs must support secure authentication mechanisms, including OAuth 2.0.
3. **API3:** APIs must handle real-time data processing and support batch updates for legacy systems.

4. **API4:** APIs must return data in standard formats such as JSON.

Integration Requirements:

1. **IR1:** Integration with ERP and WMS systems to sync inventory data.
2. **IR2:** Integration with external suppliers' and carriers' systems to collect shipment data.
3. **IR3:** Support for EDI (Electronic Data Interchange) for data exchange with legacy partners.
4. **IR4:** Integration with analytics tools for supply chain performance reporting.

Database Requirements:

1. **DB1:** The database must store shipment, inventory, and partner data for at least 7 years for regulatory compliance.
2. **DB2:** Data replication and backup mechanisms must be in place to prevent data loss.

3. **DB3:** The database must enforce data integrity and allow real-time data queries.
4. **DB4:** The system must be scalable to accommodate data growth over the next decade.

Transition Requirements:

1. **TR1:** Develop a phased migration plan from the current manual systems to the new platform.
2. **TR2:** Ensure data migration from legacy systems to the new platform is accurate and complete.
3. **TR3:** Provide comprehensive training to all users (warehouse managers, supply chain team, etc.).
4. **TR4:** Establish a support system for addressing post-implementation issues.

Data Dictionary:

- **ShipmentID:** Unique identifier for each shipment.
- **OrderID:** Unique identifier for each customer order.
- **ProductID:** Unique identifier for each product.

- **LocationCode:** Code for a specific warehouse or distribution center.
- **Status:** Current status of the shipment (e.g., In Transit, Delivered).
- **ETA:** Estimated Time of Arrival for shipments.
- **InventoryLevel:** Current stock level of a product at a specific location.

Project Risks:

1. **Risk1:** Integration challenges with external partners' systems.
    a. *Mitigation:* Conduct a thorough technical assessment of all partners' systems before integration.
2. **Risk2:** Resistance to adopting the new platform from internal and external users.
    a. *Mitigation:* Provide change management support and user training.
3. **Risk3:** Data security breaches during API exchanges.

    a. *Mitigation:* Implement robust encryption and security protocols.
4. **Risk4:** Project delays due to external vendor dependencies.
    a. *Mitigation:* Establish clear timelines and SLAs with all vendors.

Project Dependencies:

1. **Dependency1:** Availability of APIs or data exchange mechanisms from external carriers.
2. **Dependency2:** Completion of internal system upgrades to support integration.
3. **Dependency3:** Timely procurement of hardware and software from third-party vendors.
4. **Dependency4:** Compliance with industry regulations and standards for data handling.

Project Issues:

1. **Issue1:** Some suppliers' systems are not compatible with real-time API integration.

    a. *Resolution:* Support batch processing for legacy systems while encouraging future upgrades.
2. **Issue2:** Inconsistent data formats from different external partners.
    a. *Resolution:* Implement a data normalization process to standardize inputs.
3. **Issue3:** Delays in data transmission due to network issues.
    a. *Resolution:* Upgrade network infrastructure and establish a data transmission monitoring system.

Project Constraints:

1. **Constraint1:** The project must be completed within 12 months.
2. **Constraint2:** The budget is limited to $2 million.
3. **Constraint3:** The platform must be compliant with industry-specific data privacy and security regulations.

Project Assumptions:

1. **Assumption1:** All external partners (suppliers, carriers) will agree to integrate with the platform.
2. **Assumption2:** The required technical resources (personnel, equipment) will be available as scheduled.
3. **Assumption3:** There will be no major changes in regulatory requirements during the project lifecycle.

Glossary:

- **API (Application Programming Interface):** A set of protocols for building and interacting with software applications.
- **EDI (Electronic Data Interchange):** The electronic exchange of business documents between organizations.
- **ERP (Enterprise Resource Planning):** Software that manages business processes in real-time.

- **WMS (Warehouse Management System):** Software that supports the management of warehouse operations.
- **Supply Chain Visibility:** The ability to track the movement of products, materials, and inventory across the supply chain.
- **IoT (Internet of Things):** A network of physical objects embedded with sensors and software for data exchange.

SOFTWARE REQUIREMENT SPECIFICATION

1. Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) document is to detail the functional and non-functional requirements for the Real-Time Supply Chain Visibility Platform. This platform will provide centralized visibility into shipments, inventory levels, and overall supply chain operations, enabling real-time data exchange, improving decision-making, and enhancing efficiency.

1.2 Scope

The Real-Time Supply Chain Visibility Platform will be a centralized system that integrates with internal ERP, WMS, and external partners' (suppliers and carriers) systems. It will provide features such as real-time shipment tracking, automated inventory management, data integration, reporting, and notifications. This document will define the platform's software functionality, performance requirements, interface requirements, and system attributes.

1.3 Definitions, Acronyms, and Abbreviations

- **API:** Application Programming Interface
- **ERP:** Enterprise Resource Planning
- **WMS:** Warehouse Management System
- **EDI:** Electronic Data Interchange
- **SLA:** Service Level Agreement
- **ETA:** Estimated Time of Arrival

1.4 References

- Business Requirements Document (BRD) for Real-Time Supply Chain Visibility Platform
- API Documentation Standards (OAuth 2.0, RESTful APIs)
- Industry Standards (ISO 27001 for security, GDPR for data privacy)

1.5 Overview

This SRS document provides a detailed description of the system's functional and non-functional requirements, interface designs, performance

expectations, data management, security, and operational constraints.

2. System Overview

The Real-Time Supply Chain Visibility Platform aims to centralize and automate the tracking of shipments and inventory. The system will integrate with various internal and external systems to provide real-time visibility across the supply chain, delivering insights and alerts to users and improving operational efficiencies.

3. Functional Requirements

3.1 Shipment Tracking and Management

- **3.1.1 Shipment Data Collection:**
  - The system must collect real-time shipment data from external carriers using APIs.

    - The system must support batch processing for carriers that do not support real-time updates via EDI.
- **3.1.2 Shipment Status Updates:**
    - The platform will automatically update shipment statuses, including in-transit, delayed, and delivered.
    - Users should be able to view detailed shipment information, including origin, destination, content, and ETA.
- **3.1.3 Exception Handling:**
    - The platform must automatically identify and flag delayed shipments.
    - Users should receive real-time notifications for exceptions such as delays or route deviations.

3.2 Inventory Management

- **3.2.1 Inventory Data Collection:**
    - The system will collect real-time inventory data from internal WMS and ERP systems.

    - The system must automatically adjust inventory levels after each shipment is received or dispatched.
- **3.2.2 Monitor Inventory Levels:**
    - The system must track real-time inventory levels across all warehouses and suppliers.
    - The platform must trigger alerts when stock levels fall below predefined thresholds.
- **3.2.3 Reorder Management:**
    - The platform will automatically trigger reorder notifications when inventory reaches minimum levels.
    - Users must be able to set reorder rules based on stock levels, forecast demand, or sales velocity.

3.3 Integration with External Systems

- **3.3.1 API Integration:**
    - The system must support RESTful API integration with external carriers for real-time data exchange.

    - The system must use secure authentication methods (OAuth 2.0) for external data exchange.
- **3.3.2 EDI Integration:**
    - For carriers using legacy systems, the platform must support batch data transfer via EDI.
- **3.3.3 Data Validation:**
    - The system must validate incoming data from external sources to ensure accuracy and consistency before updating the database.

3.4 Data Processing and Management

- **3.4.1 Data Storage:**
    - All shipment, inventory, and partner data must be stored in a structured database.
    - The system must support data retention for at least 7 years for regulatory compliance.
- **3.4.2 Data Cleansing:**
    - The system must have mechanisms to automatically clean, validate, and remove duplicates from incoming data.

- **3.4.3 Historical Data Management:**
    - Users must be able to access and query historical shipment and inventory data for reporting and compliance purposes.

3.5 Reporting and Analytics

- **3.5.1 Real-Time Dashboards:**
    - The platform must provide real-time dashboards displaying shipment status, inventory levels, and key performance indicators (KPIs).
- **3.5.2 Report Generation:**
    - Users must be able to generate custom reports on supply chain performance, shipment history, and inventory levels.
    - The system must support exporting reports in multiple formats (Excel, PDF, CSV).

3.6 Notifications and Alerts

- **3.6.1 Real-Time Alerts:**

    - The platform must send real-time alerts for shipment delays, inventory thresholds, and system exceptions.
- **3.6.2 Customizable Alerts:**
    - Users must be able to configure and customize alerts based on shipment status, inventory levels, and supply chain performance metrics.

3.7 User Management and Security

- **3.7.1 Authentication and Access Control:**
    - The platform must implement multi-factor authentication (MFA) for user login.
    - The system must enforce role-based access control (RBAC) to restrict data access based on user roles.
- **3.7.2 Audit Trails:**
    - The system must log all user activity, including data access, changes, and administrative actions.

4. Non-Functional Requirements

4.1 Performance Requirements

- **4.1.1 System Uptime:**
  - The platform must have an uptime of 99.9% to ensure business continuity.
- **4.1.2 Data Processing Speed:**
  - Real-time data updates must be reflected in the system within 2 minutes of the event occurrence.

4.2 Scalability

- **4.2.1 User Scalability:**
  - The system must support up to 1,000 concurrent users without performance degradation.
- **4.2.2 Data Scalability:**
  - The platform must be scalable to accommodate data growth over the next 10 years, including increased shipment and inventory data.

4.3 Security

- **4.3.1 Data Security:**
  - The system must comply with industry-standard data security protocols, including ISO 27001, to protect sensitive supply chain information.
- **4.3.2 Data Encryption:**
  - All data exchanged between the platform and external systems must be encrypted using industry-standard encryption protocols (e.g., TLS 1.2).

4.4 Usability

- **4.4.1 User Interface:**
  - The platform must have an intuitive user interface that minimizes the learning curve for end-users.
- **4.4.2 Training and Support:**
  - The system should provide in-app tutorials and guides for new users.
  - Dedicated support must be available for post-implementation troubleshooting.

5. System Interface Requirements

5.1 User Interface (UI) Requirements

- **5.1.1 Dashboard Interface:**
  - The platform must provide an interactive dashboard where users can view shipment status, inventory levels, and KPIs in real-time.
- **5.1.2 Report Interface:**
  - The platform must provide an easy-to-use report generation interface where users can create, customize, and export reports.

5.2 External System Interfaces

- **5.2.1 Carrier API Integration:**
  - The system must expose secure APIs for real-time data exchange with external carriers and suppliers.
- **5.2.2 ERP and WMS Integration:**
  - The platform must integrate with internal ERP and WMS systems to synchronize shipment and inventory data.

6. Database Requirements

6.1 Database Structure

- **6.1.1 Transactional Data:**
  - The database must store all transactional data, including shipment status, inventory levels, and partner data.
- **6.1.2 Historical Data:**
  - The system must store historical data for reporting and compliance purposes, with a retention period of 7 years.

6.2 Data Backup and Recovery

- **6.2.1 Backup Frequency:**
  - The system must perform daily backups to ensure data recovery in case of failure.
- **6.2.2 Data Recovery:**
  - The platform must have a data recovery plan to restore operations within 4 hours of a system failure.

7. Transition Requirements (continued)

7.1 Data Migration

- **7.1.1 Data Conversion:**
  - Data from existing manual systems, spreadsheets, and legacy platforms (e.g., ERP, WMS, and external carrier systems) must be accurately migrated to the new real-time visibility platform. This includes historical shipment records, inventory data, and partner details.
  - A data cleansing process must be in place to remove duplicates, resolve inconsistencies, and ensure data accuracy before migration.
- **7.1.2 Data Migration Plan:**
  - A detailed migration plan must be developed to ensure minimal downtime and smooth transition from the old systems to the new platform.
  - The migration must include a comprehensive testing phase to verify data integrity post-migration.

- **7.1.3 Data Validation:**
    - Post-migration, data validation checks must be conducted to ensure that the migrated data is accurate and aligns with the new system's formats and structures.

7.2 User Training

- **7.2.1 Training Sessions:**
    - Training sessions must be conducted for all end-users, including supply chain managers, warehouse staff, and administrative personnel.
    - Training should focus on key functionalities such as shipment tracking, inventory management, reporting, and dashboard customization.
- **7.2.2 Training Materials:**
    - Detailed user manuals, quick start guides, and video tutorials must be provided to help users familiarize themselves with the new platform.

    - In-app support tools, such as tooltips and contextual help, should be embedded within the system to assist users.
- **7.2.3 Post-Deployment Support:**
    - A helpdesk or support team must be available post-implementation to address any issues or questions during the platform adoption phase.
    - Ongoing training should be available for new hires or as the system undergoes updates.

7.3 System Rollout

- **7.3.1 Phased Rollout:**
    - A phased rollout plan should be developed, starting with pilot testing in select locations or departments (e.g., a specific warehouse or region) before company-wide deployment.
    - Feedback from the pilot phase should be collected and incorporated into the final system deployment.
- **7.3.2 Parallel Run:**

    - During the transition, a parallel run (where the old and new systems operate simultaneously) may be implemented to mitigate risks and ensure business continuity.
- **7.3.3 Change Management:**
    - A change management plan must be developed to ensure user buy-in and to minimize resistance to the new system.
    - Clear communication regarding system benefits, training timelines, and expected outcomes should be maintained throughout the transition.

8. Data Dictionary

- **ShipmentID:** Unique identifier for each shipment in the system.
- **OrderID:** Unique identifier for customer orders associated with shipments.
- **ProductID:** Unique identifier for each product being tracked.

- **LocationCode:** Unique code representing a warehouse, distribution center, or storage facility.
- **ETA (Estimated Time of Arrival):** The predicted time at which the shipment is expected to reach its destination.
- **Status:** Current status of the shipment (e.g., In Transit, Delivered, Delayed).
- **InventoryLevel:** Current stock level of a product at a specific location.
- **SupplierID:** Unique identifier for suppliers involved in the supply chain.
- **CarrierID:** Unique identifier for carriers transporting goods in the supply chain.

9. Project Risks

- **Risk 1: Delays in Integration with External Partners**
    - **Impact:** Delayed data exchange with suppliers and carriers could affect real-time visibility and platform effectiveness.

    - **Mitigation:** Conduct early assessments of external systems and establish clear timelines and Service Level Agreements (SLAs).
- **Risk 2: Inaccurate Data Migration**
    - **Impact:** Data inconsistencies during migration may lead to errors in shipment tracking and inventory management.
    - **Mitigation:** Implement thorough data cleansing and validation processes before, during, and after migration.
- **Risk 3: User Resistance to New System**
    - **Impact:** Resistance to adopting the new platform may lead to poor user engagement and system underutilization.
    - **Mitigation:** Provide comprehensive training and clear communication on system benefits to ensure user buy-in.
- **Risk 4: API Failure or Security Breach**
    - **Impact:** External API failures or data breaches during integration could

compromise real-time data availability and security.
    - **Mitigation:** Implement robust encryption protocols and continuous API monitoring to detect and address failures.

10. Project Dependencies

- **Dependency 1:** Availability of APIs or data exchange mechanisms from external carriers and suppliers.
- **Dependency 2:** Internal system upgrades (ERP, WMS) must be completed to support integration.
- **Dependency 3:** Vendor cooperation is required for timely delivery of hardware and software components.
- **Dependency 4:** Compliance with data privacy and security regulations, such as GDPR or industry-specific standards.

11. Project Issues

- **Issue 1: Carrier Systems Incompatibility**
  - Some external carriers may use legacy systems incompatible with the real-time API, necessitating batch processing.
  - **Resolution:** Implement both real-time API integration and batch processing via EDI for legacy systems.
- **Issue 2: Data Inconsistency Across Partners**
  - Different data formats from external partners may cause discrepancies in shipment tracking and inventory records.
  - **Resolution:** Develop data normalization and validation processes to standardize incoming data from all partners.

12. Project Constraints

- **Constraint 1:** The project must be completed within a 12-month timeline to avoid operational disruptions.

- **Constraint 2:** The budget is limited to $2 million, and any scope expansion must be carefully evaluated for impact on costs.
- **Constraint 3:** Compliance with regulatory and industry standards (e.g., ISO 27001, GDPR) must be maintained throughout the project.

13. Project Assumptions

- **Assumption 1:** All external partners (suppliers, carriers) will cooperate in integrating their systems with the new platform.
- **Assumption 2:** The technical resources required for development, integration, and implementation will be available as scheduled.
- **Assumption 3:** No significant changes to regulatory requirements (such as data privacy laws) will occur during the project timeline.

14. Glossary

- **API (Application Programming Interface):** A set of protocols that allow different software applications to communicate with each other.
- **EDI (Electronic Data Interchange):** The electronic exchange of business documents in a standardized format between trading partners.
- **ERP (Enterprise Resource Planning):** Software systems that integrate business processes such as supply chain, inventory, and finance.
- **WMS (Warehouse Management System):** A software system designed to optimize warehouse operations, including inventory tracking and management.
- **Real-Time Supply Chain Visibility:** The ability to monitor and track the movement of goods and materials across the entire supply chain in real-time.

15. Appendix

- **A.1:** Detailed Migration Plan (Attachment)

- **A.2:** API Documentation (Attachment)
- **A.3:** User Training Materials (Attachment)

USER STORY AND ACCEPTANCE CRITERIA

User Story 1: Real-Time Shipment Tracking

As a supply chain manager,

I want to track the real-time status of shipments from various carriers,

so that I can monitor delivery progress and take action in case of delays.

Acceptance Criteria:

1. The system shall display the current status of all shipments (e.g., In Transit, Delivered, Delayed) in real-time.
2. The system must automatically update shipment status every time an event occurs (e.g., when a shipment reaches a checkpoint).
3. The user can filter shipments by status, carrier, origin, destination, and date range.
4. If the shipment is delayed, the system must send an automatic notification to the user.
5. The system shall display ETA (Estimated Time of Arrival) for all shipments.

User Story 2: Inventory Level Monitoring

As a warehouse manager,

I want to see real-time inventory levels across all warehouse locations,

so that I can prevent stockouts or overstocking.

Acceptance Criteria:

1. The system shall display inventory levels for each product across all warehouse locations in real-time.
2. The user must be able to filter the inventory view by product, location, and category.
3. The system must trigger a low-stock alert when inventory levels drop below a predefined threshold.
4. The system must allow the user to set custom inventory thresholds for different products.
5. The system must show current inventory alongside forecasted demand for comparison.

User Story 3: Integration with Carrier Systems via API

As a developer,

I want the platform to integrate with external carrier systems using APIs,

so that shipment data is automatically updated in real-time without manual intervention.

Acceptance Criteria:

1. The system must be able to consume real-time shipment data from external carriers using RESTful APIs.
2. The system must authenticate external carriers using OAuth 2.0 or other secure methods.
3. The API must be able to handle both real-time and batch processing for carriers with different data submission methods.
4. In case of API failure, the system must retry the connection up to 3 times and send an error notification if the failure persists.

5. The API integration must support both JSON and XML data formats.

User Story 4: Generate Custom Supply Chain Reports

As a supply chain analyst,

I want to generate customizable reports on shipment performance and inventory levels,

so that I can make data-driven decisions.

Acceptance Criteria:

1. The user must be able to generate reports based on shipment status, carrier, delivery timelines, and inventory levels.
2. The system must provide the option to export reports in multiple formats (Excel, PDF, CSV).
3. The user should be able to schedule recurring reports (e.g., daily, weekly) and receive them via email.

4. The system must allow the user to customize report templates, including choosing specific data fields and filters.
5. The system must generate reports within 30 seconds for up to 1,000 records.

User Story 5: Real-Time Alerts for Delays and Stock Levels

As a warehouse supervisor,

I want to receive real-time alerts for shipment delays and low inventory levels,

so that I can proactively address issues before they escalate.

Acceptance Criteria:

1. The system must send real-time alerts to the user when a shipment is delayed or inventory levels fall below the threshold.
2. The user must be able to configure alert preferences for specific products, locations, or shipment statuses.

3. Alerts must be delivered via email, SMS, or in-app notifications, based on user preference.
4. The alert notification must include relevant details such as shipment ID, location, delay reason, and expected resolution time.
5. Users should be able to acknowledge or dismiss alerts from the notification interface.

User Story 6: Role-Based Access Control (RBAC)

As an administrator,

I want to control user access to the platform's features based on roles,

so that sensitive information is protected, and users only see data relevant to their responsibilities.

Acceptance Criteria:

1. The system must support role-based access control, with predefined roles such as Admin, Manager, and User.
2. Administrators must be able to create and assign custom roles with specific access rights.

3. Users with limited roles should not be able to access or modify data that falls outside their permissions.
4. The system must enforce permissions at both the feature level (e.g., reporting, tracking) and the data level (e.g., shipment status, inventory levels).
5. Audit logs must track any changes to user roles and access settings, including who made the changes and when.

User Story 7: Historical Shipment and Inventory Data Access

As a compliance officer,

I want to access historical shipment and inventory data,

so that I can generate reports for audits and regulatory compliance purposes.

Acceptance Criteria:

1. The system must store shipment and inventory data for at least 7 years.

2. The user must be able to filter historical data by date range, shipment status, inventory levels, and location.
3. The system must provide the option to export historical data in CSV and PDF formats.
4. Historical data must be accessible within 5 seconds of a query for up to 10,000 records.
5. The system must maintain the integrity of historical data, ensuring it cannot be modified once archived.

User Story 8: Data Migration from Legacy Systems

As a developer,

I want to migrate data from the existing manual or legacy systems to the new platform,

so that all historical shipment and inventory records are available in the new system.

Acceptance Criteria:

1. Data from legacy systems must be mapped to the new platform's data structure before migration.

2. A data validation process must ensure that migrated data is accurate, complete, and free of duplicates.
3. The migration process must ensure minimal system downtime and no data loss.
4. A backup of all legacy data must be maintained in case of migration failure.
5. Users must be able to verify the accuracy of migrated data within the new system post-migration.

User Story 9: Dashboard Customization

As a supply chain manager,

I want to customize my dashboard,

so that I can see the information that is most relevant to my role.

Acceptance Criteria:

1. Users must be able to add, remove, and rearrange dashboard widgets (e.g., shipment tracking, inventory levels).

2. The system must allow users to save multiple dashboard configurations and switch between them.
3. Dashboard widgets must update in real-time, reflecting the latest shipment and inventory data.
4. Users should be able to set up dashboard filters (e.g., by location, product, shipment status).
5. The system must allow users to share their customized dashboard views with other team members, based on permissions.

User Story 10: Secure Data Exchange Between Systems

As a security officer,

I want to ensure that all data exchanged between the platform and external systems is encrypted,

so that sensitive shipment and inventory data is protected.

Acceptance Criteria:

1. All data exchanged between the platform and external systems must be encrypted using TLS 1.2 or higher.
2. API keys used for external system integration must be securely stored and managed.
3. The system must support secure authentication protocols, such as OAuth 2.0, for external integrations.
4. Data exchange logs must be maintained for audit purposes, including timestamps and status of the exchange.
5. In the event of a data exchange failure, the system must retry up to 3 times and notify the administrator if the issue persists.

USE CASE

Use Case 1: Real-Time Shipment Tracking

- **Use Case ID:** UC-001
- **Use Case Name:** Real-Time Shipment Tracking
- **Use Case Description:** The system enables the supply chain manager to track the status of shipments in real-time, including shipment details and any delays.
- **Primary Actor:** Supply Chain Manager
- **Supporting Actor:** Carrier System
- **Pre-Condition:** Shipments are initiated, and carrier systems are integrated via API for real-time data exchange.
- **Post Condition:** Shipment statuses are updated in real-time, and any delays are flagged to the user.

Main Flow:

1. **Actor Step:** The Supply Chain Manager accesses the platform's shipment tracking dashboard.
    a. **System Response:** The system displays a list of active shipments along with their current status.

2. **Actor Step:** The manager selects a specific shipment to view detailed information.
    a. **System Response:** The system retrieves and displays shipment details (origin, destination, ETA, current status).
3. **Actor Step:** The manager clicks on the 'Track Shipment' option to view real-time movement.
    a. **System Response:** The system displays the real-time location of the shipment, including a map view and any checkpoints reached.
4. **Actor Step:** The manager applies filters (e.g., by carrier, date range).
    a. **System Response:** The system updates the shipment list based on the selected filters.
5. **Actor Step:** The manager sets alerts for any shipment delays.
    a. **System Response:** The system confirms the alerts and will notify the manager if a delay occurs.

Alternate Flow:

- **If the carrier system does not support real-time updates** (uses batch processing via EDI):
  - The system will retrieve and display the latest batch data instead of real-time data.
  - The system will display a notification that real-time updates are not available for this carrier.

Exception Flow:

- **If the shipment data fails to load due to API failure**:
  - The system will display an error message: "Unable to retrieve shipment data."
  - The system will prompt the manager to retry or notify support.

Additional Requirements:

- **Functional Requirements:**
  - The system must update shipment status every time an event occurs.

    - Real-time location tracking for shipments must be displayed on a map.
- **Non-Functional Requirements:**
    - Data should be updated in less than 2 minutes after the shipment status changes.
- **Database Requirements:**
    - Shipment status, location, and ETA should be stored in the shipment table.
- **Technical Requirements:**
    - API integration with carrier systems using RESTful APIs or EDI for batch updates.

Use Case 2: Inventory Level Monitoring

- **Use Case ID:** UC-002
- **Use Case Name:** Inventory Level Monitoring
- **Use Case Description:** The system provides real-time monitoring of inventory levels across different warehouse locations.
- **Primary Actor:** Warehouse Manager
- **Supporting Actor:** Warehouse Management System (WMS)

- **Pre-Condition:** The WMS is integrated with the platform and provides real-time inventory data.
- **Post Condition:** Inventory levels are updated in real-time, and alerts for low stock levels are triggered.

Main Flow:

1. **Actor Step:** The Warehouse Manager logs into the system and navigates to the inventory dashboard.
    a. **System Response:** The system displays current inventory levels for each product and warehouse.
2. **Actor Step:** The manager filters the inventory list by product category.
    a. **System Response:** The system updates the inventory list based on the selected category.
3. **Actor Step:** The manager clicks on a specific product to view detailed stock information.
    a. **System Response:** The system displays stock levels across all locations, including minimum and maximum thresholds.

4. **Actor Step:** The manager sets low-stock alerts for specific products.
    a. **System Response:** The system saves the alert configuration and will notify the manager when inventory falls below the set threshold.
5. **Actor Step:** The manager exports the current inventory data for reporting.
    a. **System Response:** The system generates and exports the data in the selected format (Excel, CSV, PDF).

Alternate Flow:

- **If inventory data is not updated due to WMS system issues**:
    - The system will display the last available inventory data and a warning that real-time updates are not available.

Exception Flow:

- **If the inventory details for a specific product fail to load**:

- The system will display an error message: "Unable to retrieve inventory data."
- The system will allow the manager to retry or contact support.

Additional Requirements:

- **Functional Requirements:**
  - Real-time inventory updates must be displayed.
  - Alerts for low stock must be configurable by product and warehouse.
- **Non-Functional Requirements:**
  - System must update inventory data within 2 minutes of changes.
- **Database Requirements:**
  - Inventory levels and thresholds must be stored in the inventory table.
- **Technical Requirements:**
  - Integration with the WMS via API or batch processing for legacy systems.

Use Case 3: Generate Custom Supply Chain Reports

- **Use Case ID:** UC-003
- **Use Case Name:** Generate Custom Supply Chain Reports
- **Use Case Description:** The system allows the supply chain analyst to generate custom reports on shipment and inventory performance.
- **Primary Actor:** Supply Chain Analyst
- **Supporting Actor:** None
- **Pre-Condition:** Historical shipment and inventory data are stored in the system.
- **Post Condition:** A custom report is generated and available for export in various formats.

Main Flow:

1. **Actor Step:** The Supply Chain Analyst accesses the reporting section of the platform.
    a. **System Response:** The system displays available report templates and customization options.

2. **Actor Step:** The analyst selects the report type (e.g., shipment performance, inventory trends).
    a. **System Response:** The system loads the selected report template with default settings.
3. **Actor Step:** The analyst customizes the report by selecting specific fields (e.g., date range, carrier, product category).
    a. **System Response:** The system updates the report preview based on the selected filters.
4. **Actor Step:** The analyst clicks on "Generate Report."
    a. **System Response:** The system generates the report and displays it in the browser.
5. **Actor Step:** The analyst exports the report in the desired format (Excel, CSV, PDF).
    a. **System Response:** The system downloads the report in the selected format.

Alternate Flow:

- **If the analyst schedules a recurring report**:

    - The system allows the analyst to set up a report schedule (e.g., daily, weekly).
    - The system sends the scheduled report via email to the analyst on the specified schedule.

Exception Flow:

- **If the report generation takes too long or fails**:
    - The system displays an error message: "Report generation failed. Please try again later."
    - The system allows the analyst to retry or change the report parameters.

Additional Requirements:

- **Functional Requirements:**
    - The system must support report generation based on shipment and inventory data.
    - Reports must be exportable in multiple formats (Excel, CSV, PDF).
- **Non-Functional Requirements:**

    - Reports must be generated within 30 seconds for up to 1,000 records.
- **Database Requirements:**
    - Historical shipment and inventory data must be stored for report generation.
- **Technical Requirements:**
    - The system must allow users to schedule reports for automated delivery.

Use Case 4: Real-Time Alerts for Delays and Low Stock

- **Use Case ID:** UC-004
- **Use Case Name:** Real-Time Alerts for Delays and Low Stock
- **Use Case Description:** The system sends real-time alerts for shipment delays and low inventory levels based on predefined thresholds.
- **Primary Actor:** Warehouse Supervisor
- **Supporting Actor:** None
- **Pre-Condition:** Shipment and inventory data are being updated in real-time.

- **Post Condition:** The warehouse supervisor receives real-time alerts for any delays or low stock levels.

Main Flow:

1. **Actor Step:** The Warehouse Supervisor configures alerts for shipment delays and low stock levels.
    a. **System Response:** The system saves the alert configuration.
2. **Actor Step:** The system detects a delay in a shipment.
    a. **System Response:** The system sends a real-time alert to the supervisor via email and in-app notification.
3. **Actor Step:** The system detects that inventory for a specific product has fallen below the predefined threshold.
    a. **System Response:** The system sends a low-stock alert to the supervisor.
4. **Actor Step:** The supervisor views the details of the delay or low stock notification.

a. **System Response:** The system displays shipment details or inventory levels, along with the reason for the alert.

Alternate Flow:

- **If the supervisor configures custom alert preferences**:
  - The system allows the supervisor to configure specific parameters (e.g., only receive alerts for high-priority shipments or certain products).
  - The system sends alerts based on the custom preferences.

Exception Flow:

- **If the alert system fails to send notifications**:
  - The system logs the failure and retries sending the alert.
  - If the alert still fails, the system sends an error notification to the administrator.

Additional Requirements:

- **Functional Requirements:**
  - The system must support real-time alerts for delays and low stock levels.
  - Users must be able to customize alert preferences.
- **Non-Functional Requirements:**
  - Alerts must be sent within 1 minute of detecting a delay or low stock.
- **Database Requirements:**
  - Alert settings and thresholds must be stored in the user preferences table.
- **Technical Requirements:**
  - The system must support sending alerts via multiple channels (email, SMS, in-app).

THANK YOU