Shipping Port Management System (SPMS) - Requirement Specification Document

1. Introduction

1.1 Purpose

The Shipping Port Management System (SPMS) is designed to streamline port operations by managing ship movements, cargo handling, employee coordination, and financial transactions. It aims to enhance efficiency, security, and transparency in port management.

1.2 Scope

The system will support:

- Real-time ship tracking and berth allocation
- Cargo loading/unloading management
- Employee and crew management
- Port security and compliance handling
- Financial and billing system
- User roles with controlled access

1.3 Stakeholders

- **Port Authorities**: Oversee operations and compliance.
- **Shipping Companies**: Manage ship schedules and cargo handling.
- **Logistics Providers**: Ensure smooth cargo transportation.
- **Customs Officers**: Handle import/export regulations.
- **Dock Workers & Employees**: Execute port operations.
- **Customers**: Track shipments and access billing details.

---

2. Functional Requirements

2.1 User Management

- User authentication (Admin, Employee, Captain, Customer).
- Role-based access control (RBAC).

2.2 Ship & Cargo Management

- Add, update, and delete ship details.

- Assign berths and schedule arrivals/departures.
- Track cargo loading/unloading operations.
- Monitor container storage and movement.

2.3 Employee & Crew Management

- Maintain employee records (name, email, department, country, DOB).
- Assign tasks to employees and dock workers.
- Manage captains and ship crew details.

2.4 Financial Management

- Generate invoices and billing reports.
- Track payments and financial transactions.

2.5 Security & Compliance

- Ensure regulatory compliance for customs and port security.
- Implement logging and auditing features for tracking activities.

---

3. Non-Functional Requirements

- **Scalability**: Handle high traffic and multiple ports.
- **Security**: Implement encryption and secure authentication.
- **Performance**: Ensure real-time tracking and fast query processing.
- **Reliability**: Ensure system uptime with cloud-based hosting.

---

4. Technology Stack

4.1 Database

- MySQL / PostgreSQL for structured data storage.

4.2 Backend

- Node.js (Express) or Python (Django/Flask) for API development.

4.3 Frontend

- React.js / Angular / Vue.js for dashboards and user interfaces.

4.4 Hosting & Deployment

- AWS / Azure / GCP for cloud deployment.
- Docker & Kubernetes for containerized services.

---

5. System Architecture Overview

5.1 Database Schema (ER Diagram Overview)

Entities:

- **Customer** (cust_id, name, email, password)
- **Employee** (emp_id, name, email, empno, dept, description, nativecountry, dob)
- **Captain** (cap_id, name, email, ship_id, shipregno, crewno, shiptype, countryorigin, grosstonnage)
- **TEUS** (teu_id, teucode, destination, portloading, ship_id, shipregno, owner_id)

Relationships:

- One **captain** is assigned to one **ship**.
- One **ship** can have multiple **TEUS containers**.
- One **employee** can work on multiple **ships/cargo operations**.

---

6. Project Roadmap & Timeline

Phase	Task	Timeline
Phase 1	Planning & Requirement Gathering	Week 1-4
Phase 2	System Design & Architecture	Week 5-8
Phase 3	Backend & Database Development	Week 9-14
Phase 4	Frontend & API Development	Week 15-20
Phase 5	Testing & Quality Assurance	Week 21-24
Phase 6	Deployment & Monitoring	Week 25-28

---

7. Conclusion

This document serves as a foundation for the development of the SPMS. Future iterations will refine system features and architecture based on stakeholder feedback.

Next Steps:

- Develop an **ER Diagram** and detailed database schema.
- Define API endpoints for core functionalities.
- Create initial UI wireframes for user interaction.