

README FILE

Student Name: Shivanshi

Branch: UIC/BCA

Semester: 4th

Subject Name: DBMS

UID: 23BCA10432

Section/Group: BCA-4/b

Submitted to: Mr.Arvinnder Singh







Subject Code: 23CAP-252

SHIP PORT MANAGEMENT SYSTEM

Project Overview

The **Ship Port Management System** is a **database-driven solution** designed to streamline ship, cargo, port, and transaction management. It ensures efficient **scheduling, dock allocation, cargo tracking, employee management, and financial transaction recording** using relational database principles.

Key Features

-  **Ship Management:** Registers ships, their capacity, and ownership details.
-  **Cargo Tracking:** Logs cargo type, weight, and ship assignments.
-  **Port & Dock Utilization:** Stores port details and monitors dock availability.
-  **Scheduling System:** Manages ship arrival and departure timelines.
-  **Employee Records:** Maintains staff working at each port.
-  **Transaction Logging:** Ensures secure financial recording of cargo shipments.

Database Schema

The system consists of **eight tables** ensuring data integrity with **primary and foreign keys**:

Table Name	Primary Key (PK)	Foreign Key (FK)	Constraints
Ship	ShipID	OwnerID	NOT NULL, UNIQUE
Owner	OwnerID	None	NOT NULL, UNIQUE
Port	PortID	None	NOT NULL, UNIQUE
Dock	DockID	PortID	CHECK(Status IN ('Available', 'Occupied'))
Cargo	CargoID	ShipID	NOT NULL, UNIQUE
Schedule	ScheduleID	ShipID, PortID	NOT NULL, UNIQUE
Employee	EmployeeID	PortID	NOT NULL, UNIQUE
Transaction	TransactionID	CargoID, ShipID	CHECK(Amount > 0)



SQL Queries

The system supports **basic SQL operations**, including:

- **Retrieval:** `SELECT * FROM Ship;`
- **Insertion:** `INSERT INTO Ship (ShipID, ShipName, Capacity, OwnerID) VALUES (6, 'Blue Wave', 5500, 3);`
- **Updating:** `UPDATE Ship SET Capacity = 6000 WHERE ShipID = 3;`
- **Deletion:** `DELETE FROM Ship WHERE ShipID = 5;`
- **Aggregations:** `SELECT AVG(Capacity) AS AverageCapacity FROM Ship;`
- **Joins:** `SELECT Ship.ShipName, Owner.OwnerName FROM Ship JOIN Owner ON Ship.OwnerID = Owner.OwnerID;`

System Requirements

- **Database:** MySQL, PostgreSQL, or SQL Server
- **Development Tools:** MySQL Workbench, pgAdmin, or SSMS
- **Operating System:** Windows/Linux/macOS
- **RAM:** 8GB (Minimum), 16GB+ (Recommended)
- **Storage:** SSD (Recommended for fast query execution)

Limitations & Future Enhancements

🔧 Limitations:

- Requires **manual data entry** for initial setup
- **No real-time ship tracking** integration
- Depends on **SQL knowledge for queries**

🔧 Future Enhancements:

- Add **GPS integration** for ship tracking
- Implement **automated notifications** for arrivals/departures
- Enhance **cloud-based scalability** for large-scale operations

Conclusion

The **Ship Port Management System** offers **efficient data handling, enhanced scheduling, and structured maritime logistics management** using relational database principles. While it addresses key aspects of port management, further enhancements can make it **more dynamic and scalable** for larger implementations.

◆ **Developed with structured SQL queries to ensure accuracy and efficiency.** ◆ **A powerful foundation for modernizing port logistics operations.**

★ Contributors & Acknowledgment

Special thanks to our mentors, team members, and database experts for their guidance and contributions!