Cruise Ship Management System (CSM)

A Web-Based Management and Voyager Platform

---

1. Introduction

The **Cruise Ship Management System (CSM)** is a web-based application designed to streamline operations for cruise ships while providing voyagers (passengers) with a smooth and engaging user experience. It is developed using **JavaScript** for the logic and front-end, **Firebase** for authentication and backend integration, and custom UI styling and responsive design.

This project integrates features for multiple roles: **Admin, Supervisor, Manager, and Voyager**. Each role has its own dashboard, ensuring clear separation of responsibilities and secure access to relevant functionality.

---

2. Objectives

- To build a **multi-role platform** that manages cruise operations efficiently.
- To implement **role-based authentication and access control**.
- To design **interactive dashboards** for each role with intuitive UI/UX.
- To ensure **responsive design** for all devices (desktop, tablet, mobile).
- To integrate essential functionalities such as bookings, ordering, item management, and voyager registration.

---

3. Technologies Used

- **Frontend**: HTML5, CSS3, JavaScript (ES6+).
- **Backend / Database**: Firebase Authentication & Firestore Database.
- **Styling & UI**: Custom CSS, gradient designs, hover animations, responsive grid layouts.

---

4. System Architecture

The system follows a **role-based architecture**. After authentication, users are redirected to dashboards specific to their roles.

Roles and Responsibilities:

- **Admin**
  - Manage catering and stationery items.
  - Register voyagers with secure authentication.
  - Access and monitor voyager lists.
- **Supervisor**

    - Oversee and manage orders placed by voyagers.

- **Manager**
    - Monitor bookings made by voyagers.

- **Voyager (Passenger)**
    - Browse and book cruises, meals, and resort services.
    - Place orders for catering/stationery items.
    - View bookings and orders in personal dashboard.

---

4. Project Structure

The folder structure of the Cruise Ship Management System is as follows:

```
CruiseShipApp/
├── Images/                        #BG image
├── Scripts/                       # JS logic for all pages(includes firebase.js)
├── styles/                        # Styling for all pages
├── dashboard-admin.html           # Admin page
├── dashboard-head-cook.html       # Head Cook page
├── dashboard-manager.html         # Manager page
├── dashboard-supervisor.html      # Supervisor page
├── dashboard-voyager.html         # Voyager page
├── login.html                     # Login page
├── register.html                  # Register page
└── README.md                      # Documentation
```

---

5. How to Run Instructions

Prerequisites

- Firebase project configured with Authentication and Firestore.
- All files should be in their respective folders.

Steps

1. Edit the firebase.js files with your own firebase config.
2. Open `register.html` and register an account with desired role.
3. You will be logged in as your selected role.
4. If you logged in as **Voyager** you can Book/Order amenities and food. You can cancel bookings.

6. Implementation Details

6.1 Authentication

- Implemented using **Firebase Authentication**.
- Supports **secure login, registration, and logout**.
- Role-based redirection: Users are redirected to role-specific dashboards after login.

6.2 Dashboards

Each role has a separate dashboard:

- **Admin Dashboard**
  - Item management (add/remove catering items).
  - Voyager registration with email & password authentication.
- **Supervisor Dashboard**
  - Displays orders in card format (with vertical stacking when screen is smaller).
- **Manager Dashboard**
  - Displays bookings in card format (with responsive design).
- **Voyager Dashboard**
  - Explore cruises, meals, resorts, and other facilities.
  - Place catering or stationery orders.
  - Book services and view confirmed bookings.

---

9. Test Cases

The following table summarizes key test cases executed for the Cruise Ship Management System (CSM):

Test Case ID	Description	Input / Action	Expected Output	Actual Result	Status
TC-01	User Login	Enter valid email & password	Redirected to role-specific dashboard	Works as expected	Pass
TC-02	Invalid Login	Enter wrong credentials	Error message displayed, no login	Works as expected	Pass
TC-03	Voyager Registration (by Admin)	Fill form and submit	New voyager added to database	Voyager appears in Firestore	Pass
TC-04	Add Catering Item (Admin)	Add new item details	Item saved in database	Works as expected	Pass
TC-05	Place Order (Voyager)	Select item and confirm order	Order stored in database and visible in supervisor dashboard	Works as expected	Pass
TC-06	Cancel Order (Supervisor)	Click cancel on an order	Order status updated to "Cancelled"	Works as expected	Pass
TC-07	View Bookings (Manager)	Access bookings page	All voyager bookings displayed	Works as expected	Pass
TC-08	Cancel Booking (Admin)	Select a booking to cancel	Booking status updated in database	Works as expected	Pass

Test Case ID	Description	Input / Action	Expected Output	Actual Result	Status
TC-09	Responsive Design	Resize browser to < 540px	Navbar collapses into hamburger menu, cards adjust layout	Works as expected	Pass
TC-10	Logout	Click logout button	User session cleared, redirected to login page	Works as expected	Pass

---

7. Challenges Faced

- **Responsive dropdown menus**: Needed different behaviors for desktop, mid-size, and mobile screens.
- **Role-based redirection**: Ensuring correct access without exposing unauthorized functionality.
- **Firebase integration**: Configuring secure login and voyager registration.
- **Consistency across dashboards**: Balancing UI/UX for all roles without breaking layouts.

---

8. Results

- Successfully implemented a **multi-role management system**.
- All four dashboards (Admin, Supervisor, Manager, Voyager) are functional and styled consistently.
- Authentication and role-based access control fully operational.
- System is responsive, working across desktop, tablet, and mobile devices.
