Cruise Ship Management System (CSM)

1. Project Overview

The **Cruise Ship Management System (CSM)** is a web-based application designed to manage services, bookings, and orders on a cruise ship using **Firebase Authentication** and **Cloud Firestore**.

The system follows a **role-based access control model, where each user is redirected to a dedicated dashboard based on their assigned role. All data updates are handled in **real-time** using Firestore listeners.

---

2. Technologies Used

- **Frontend:** HTML, CSS, JavaScript (ES Modules)
- **Backend / Database:** Firebase
  - Firebase Authentication
  - Cloud Firestore (Real-time Database)

---

3. User Roles & Access Control

Each registered user is assigned **one role**, stored in the Firestore `users` collection. Access is enforced both during login and when loading dashboards.

Available Roles

- **Voyager** – Cruise passenger
- **Head Cook** – Catering department
- **Supervisor** – Stationery & retail department
- **Manager** – Booking oversight
- **Admin** – System administrator

If a user attempts to access an unauthorized dashboard, they are automatically redirected to their permitted dashboard.

The following are the accounts for each role:

1. **Voyager**
   - Email -
2. **Head Cook**
3. **Supervisor**
4. **Manager**
5. **Admin**

---

4. Authentication Flow

4.1 Registration

- New users register using the **Register page**.
- During registration, the user **selects their role from a dropdown menu** (Voyager, Head Cook, Supervisor, Manager, or Admin).
- Firebase Authentication creates the user account.
- A corresponding Firestore document is created in the `users` collection containing:
  - Username
  - Email
  - Selected role
  - Account creation timestamp
- After registration, users are redirected to their role-specific dashboard.

4.2 Login

- Users log in using email and password.
- On successful login:
  - The system fetches the user role from Firestore.
  - The user is redirected to the appropriate dashboard.
- If the role is invalid or missing, access is denied.

4.3 Logout

- Available on all dashboards.
- Ends the Firebase session and redirects the user to the login page.

---

5. Dashboards & Functionalities

5.1 Voyager Dashboard

Role: Cruise Passenger

Features

- Browse available services and items via a navigation menu
- **Bookings**:
  - Movies
  - Resorts
  - Salon
  - Spa
  - Medical Consultation
  - Gym
  - Yoga Classes
  - Art Classes
  - Event Hall
  - Banquet Hall
- **Orders**:
  - Snacks
  - Meals

    - Beverages
    - Books
    - Gifts
    - Chocolates

Actions

- Place bookings or orders
- View personal bookings and orders
- Cancel existing bookings

Data Handling

- Writes data to `bookings` and `orders` collections
- Reads only documents matching the logged-in user's UID

5.2 Head Cook Dashboard

Role: Catering Department

Features

- View catering-related orders only

Order Categories

- Snacks
- Meals
- Beverages

Data Handling

- Real-time Firestore listener
- Filters orders based on item category
- Read-only access

5.3 Supervisor Dashboard

Role: Retail & Stationery Supervisor

Features

- View stationery-related orders only

Order Categories

- Books
- Gifts

- Chocolates

Data Handling

- Uses Firestore queries with category-based filtering
- Real-time updates
- Read-only access

5.4 Manager Dashboard

Role: Booking Manager

Features

- View all bookings made by voyagers
- Monitor booking details such as:
  - Item name
  - Category
  - Voyager email
  - Booking time

Data Handling

- Real-time snapshot listener on `bookings` collection
- No modification privileges

5.5 Admin Dashboard

Role: System Administrator

Features

- View all bookings and orders
- Cancel bookings
- Manage system items
- Register new voyager accounts
- Remove voyager accounts

Item Management

- Add new items with category/type
- Delete existing items

User Management

- Register new voyager accounts
- View all voyagers
- Delete voyager accounts

Data Handling

- Full read/write access
- Uses live Firestore listeners for all major collections

---

6. Database Structure (Firestore)

Collections

- `bookings`
  - createdAt
  - itemCategory
  - itemName
  - voyagerEmail
  - voyagerUid
- `items`
  - createdAt
  - name
  - type
- `orders`
  - createdAt
  - itemCategory
  - itemName
  - voyagerEmail
  - voyagerUid
- `users`
  - uid
  - createdAt
  - email
  - role
  - username

---

7. Real-Time Updates

- All dashboards use Firestore `onSnapshot` listeners
- Data updates appear instantly without page refresh
- Demonstrates real-time database capabilities

---

8. Security & Access Control

- Role validation on every dashboard load

- Unauthorized access results in automatic redirection
- Users can only access data relevant to their role

---

9. Notes & Limitations

- Project is a prototype
- Timestamp formats may vary (ISO strings used)

---

10. Conclusion

The Cruise Ship Management System successfully demonstrates:

- Role-based authentication and authorization
- Real-time data handling
- Modular dashboard design
- Practical use of Firebase services