Hotel Management System

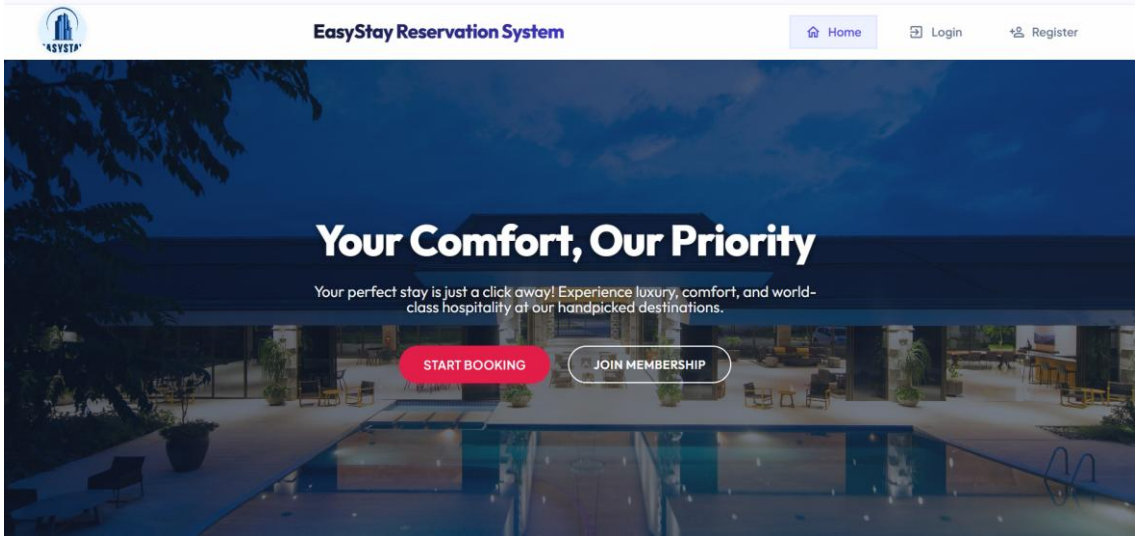
A modern Angular + ASP.NET Core application for hotel reservation management with secure JWT authentication and role-based access control.

Features

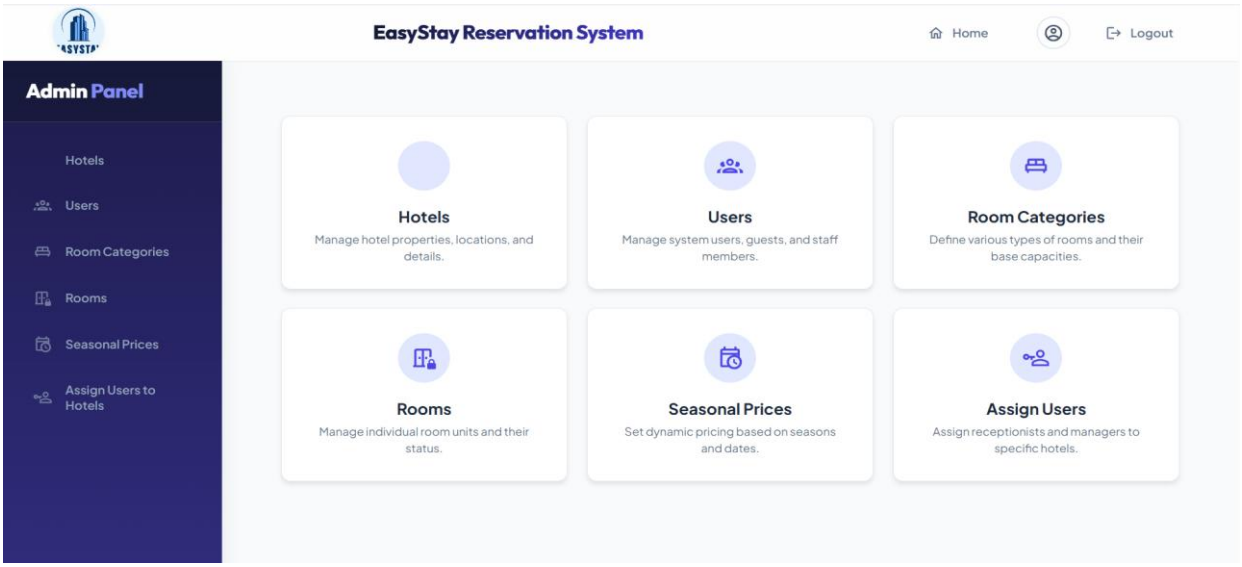
Core Functionality

- Authentication & Authorization: JWT-based login/register with role-based access
- Hotel Management: Browse and manage hotels
- Room Booking: Search available rooms and make reservations
- Room Service: guests can request services; receptionist fulfill, and bill these requests.
- Reservation Management: View and manage bookings
- Billing System: View bills and process payments
- Role-Based Dashboard: Different interfaces for Guest, Receptionist, HotelManager, and Admin
- Recommendation System: Recommends hotels based on previous reservations

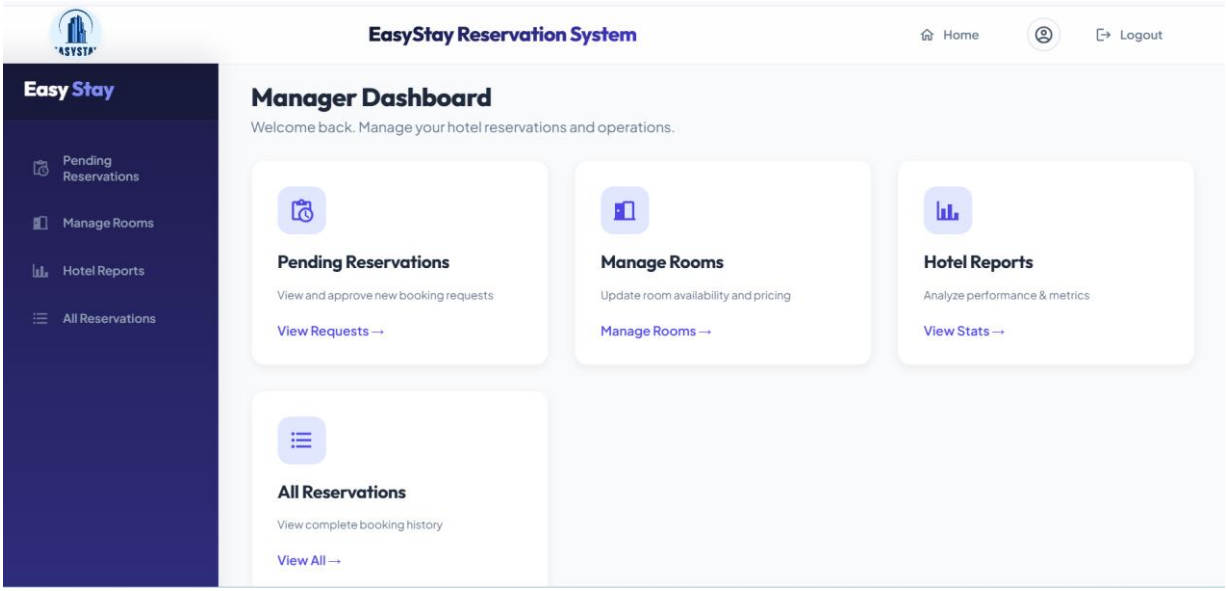
Home page



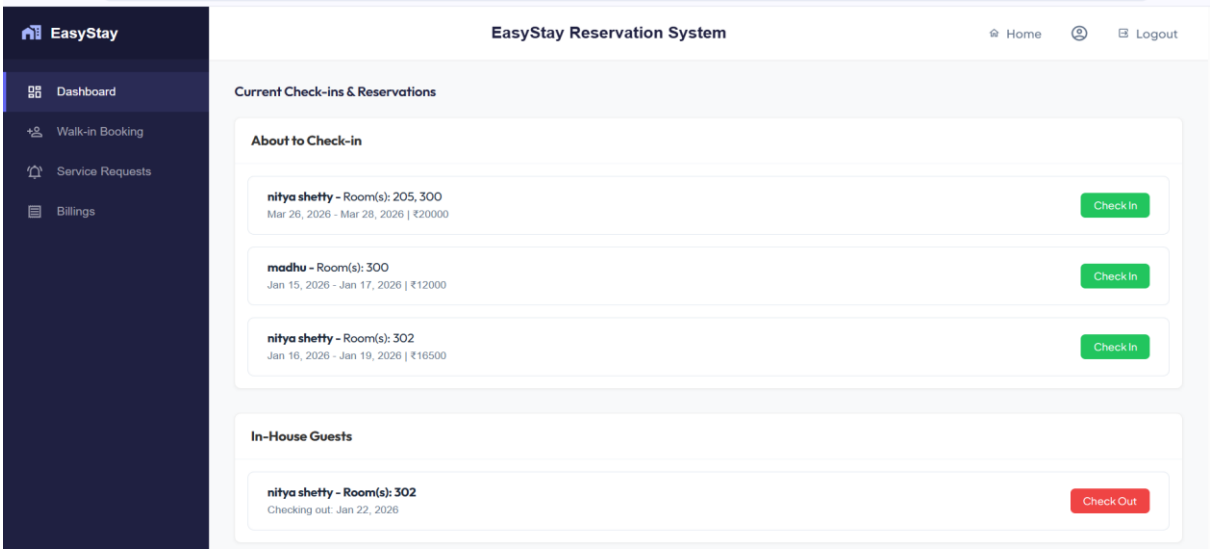
Admin Dashboard

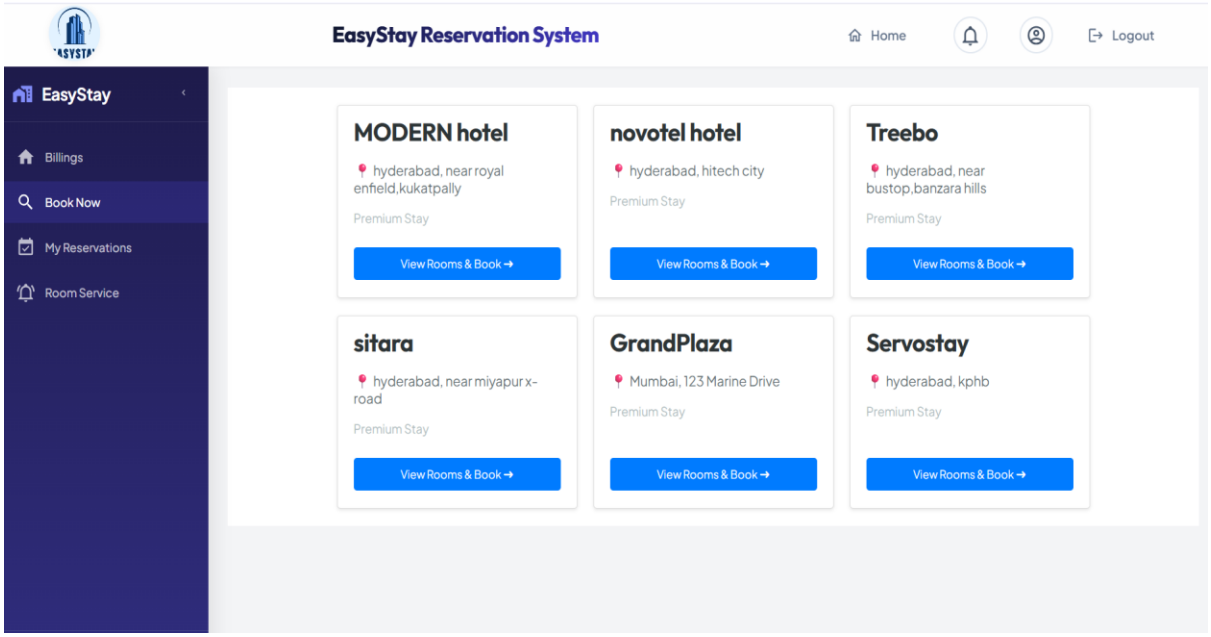


Manager Dashboard



Receptionist Dashboard





User Roles

- Guest: Browse hotels, book rooms, manage reservations, pay bills
- Receptionist: Check-in/out guests, view guest information
- HotelManager: Manage hotels, rooms, and view reports
- Admin: Full system access, user management, role assignment

Technology Stack

Frontend

- Angular 21 (Standalone Components)
- TypeScript
- Reactive Forms
- Angular HTTP Client
- JWT Authentication
- CSS (No SCSS)

Backend

- ASP.NET Core 8 Web API
- Entity Framework Core
- SQL Server
- JWT Bearer Authentication
- Role-Based Authorization
- AutoMapper
- Swagger / OpenAPI
- Background Hosted Services
- Custom styling (no SCSS as requested)

Setup Instructions

Prerequisites
- Node.js (v18 or higher)
- Angular CLI (v21 or higher)
- Backend API running on http://localhost:5254

Installation

1. Clone and navigate to the project
   cd hotel-client

2. Install dependencies
   npm install

3. Start development server
   ng serve --open

4. Start backend server
   cd HotelWebApi

dotnet ef database update

dotnet run

5. Access the application

- Open browser to http://localhost:4200
- The app will automatically reload on file changes

Testing Credentials

Use these test accounts with your backend:

```
// Admin Account
{
  email: "admin@hotel.com",
  password: "Admin@123"
}

// Guest Account
{
  email: "madhu@gmail.com",
  password: "Madhu@123"
}

// Hotel Manager Account
{
 email: "reena@gmail.com",
 password: "Reena@123"
}

// Receptionist Account
```

{
  email: "mohana@gmail.com",
  password: "Mohana@123"
}

API Configuration

The application is configured to connect to the backend API at:
- Base URL: http://localhost:5254
- Swagger URL: http://localhost:5254/swagger (use for API Testing)
- Authentication: JWT tokens stored in localStorage
- Interceptor: Automatically adds Bearer token to requests

Security Features

- JWT Authentication: Secure token-based authentication
- Route Guards: Protect routes based on authentication and roles
- HTTP Interceptor: Automatic token injection
- Platform Checks: SSR-safe localStorage access
- Role-Based Access: Different UI based on user roles

Troubleshooting

Common Issues

1. CORS Errors: Ensure backend API allows requests from http://localhost:4200

2. Token Expiry: Check JWT token expiration and refresh logic

3. Route Access: Verify user roles match required permissions

4. API Connection: Confirm backend is running on http://localhost:5254