Authentication System Setup

☑ Completed

Frontend

- **Login Page** (`frontend/login.html`)
  - Professional split-screen design
  - Form validation with real-time feedback
  - Password visibility toggle
  - "Remember me" functionality
  - Loading states during API calls
  - Alert messages for success/error
  - Backend integration: `POST /api/auth/login`
  - Token storage (localStorage/sessionStorage)
  - Automatic redirect to dashboard on success
- **Signup Page** (`frontend/signup.html`)
  - Professional split-screen design
  - Form validation with password strength indicator
  - Password confirmation
  - Terms of service acceptance
  - Newsletter subscription option
  - Loading states during API calls
  - Alert messages for success/error
  - Backend integration: `POST /api/auth/signup`
  - Automatic redirect to login on success
- **Authentication Styles** (`frontend/css/auth.css`)
  - Modern gradient background
  - Split-screen responsive layout
  - Professional form styling
  - Alert message animations
  - Password strength indicator colors
  - Hover effects and transitions
  - Mobile-responsive design (992px, 576px breakpoints)
  - Accessibility features (focus states)

Backend

- **Authentication Controller** (`backend/controllers/auth.js`)
  - Signup endpoint: Creates user, hashes password, returns JWT
  - Login endpoint: Validates credentials, returns JWT
  - Password hashing with bcrypt
  - JWT token generation (7-day expiration)
  - Email uniqueness validation
  - Error handling
- **User Model** (`backend/models/user.js`)
  - Name, email, password fields
  - Role-based access (user/admin)
  - Bcrypt password hashing middleware
  - Password comparison method
- **Authentication Routes** (`backend/routes/auth.js`)
  - POST /api/auth/signup
  - POST /api/auth/login
  - GET /api/auth/me (protected)

🖊 Testing Instructions

1. Start the Backend Server

```
cd backend
npm install
node server.js
```

Server should start on `http://localhost:5000`

2. Test Signup

1. Open `frontend/signup.html` in browser
2. Fill in:
   - Name: Test User
   - Email: test@example.com
   - Password: Test123!@# (should show "Strong")
   - Confirm Password: Test123!@#
   - Check "I agree to Terms"
3. Click "Create Account"
4. Should see success message and redirect to login

3. Test Login

1. On `frontend/login.html`
2. Enter credentials:
   - Email: test@example.com
   - Password: Test123!@#
3. Optionally check "Remember me"
4. Click "Sign In"
5. Should see success message and redirect to dashboard

4. Verify Token Storage

- Open browser DevTools > Application/Storage
- Check localStorage (if "Remember me") or sessionStorage

- Should see `token` key with JWT value

🔐 Security Features

- ✅ Password hashing with bcrypt (10 salt rounds)
- ✅ JWT tokens with 7-day expiration
- ✅ CORS enabled for cross-origin requests
- ✅ Input validation on frontend and backend
- ✅ Password strength indicator
- ✅ HTTP-only token storage recommendation
- ✅ Bank-level encryption notice to users

📋 API Endpoints

Signup

```
POST http://localhost:5000/api/auth/signup
Content-Type: application/json

{
  "name": "John Doe",
  "email": "john@example.com",
  "password": "securePassword123"
}

Response (201):
{
  "success": true,
  "message": "User registered successfully",
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "user": {
    "id": "...",
    "name": "John Doe",
    "email": "john@example.com",
    "role": "user"
  }
}
```

Login

```
POST http://localhost:5000/api/auth/login
Content-Type: application/json

{
  "email": "john@example.com",
  "password": "securePassword123"
}

Response (200):
{
  "success": true,
  "message": "Login successful",
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "user": {
    "id": "...",
    "name": "John Doe",
    "email": "john@example.com",
    "role": "user"
  }
}
```

🎨 Design Features

- Split-screen layout with gradient background
- Animated password strength indicator
- Real-time form validation
- Loading spinners during API calls
- Auto-hiding alert messages (5 seconds)
- Responsive design for mobile devices
- Professional color scheme matching booking.com style

🚀 Next Steps

1. **Add authentication checks to protected pages**:
    - dashboard.html
    - bookings.html
    - payments.html
2. **Implement logout functionality**:
    - Clear token from storage
    - Redirect to login
3. **Add "Forgot Password" feature**
4. **Implement social login** (Google, Facebook buttons are UI-ready)
5. **Add email verification for new signups**