Project Design Phase

1. Introduction

The House Rental Platform is designed to allow users to easily find, view, and rent properties online. The design focuses on delivering a user-friendly, scalable, and secure experience.

---

2. User Personas

Persona	Description	Goals
Renter	A user looking for a house to rent	Find a house quickly, filter by location and price, save favorites, easy booking
Owner	A user listing their property for rent	List properties easily, manage bookings, communicate with renters
Admin	Manages the platform	Moderate listings, manage users, resolve issues

---

3. System Architecture

- **Frontend:**
  - Framework: **React.js + Next.js** (for Server-Side Rendering)
  - UI Library: **TailwindCSS**
  - Key Features: SEO-friendly, fast, mobile-first design
- **Backend:**
  - Framework: **Node.js + Express.js**
  - Authentication: **JWT-based session management**
  - API Routes: RESTful services
- **Database:**
  - **MongoDB Atlas**
  - Collections: Users, Properties, Bookings, Reviews
- **Hosting:**
  - Frontend: **Vercel**
  - Backend: **Render or AWS**

---

4. Database Design (Schema Outline)

Collection	Key Fields
Users	id, name, email, password, userType (renter/owner/admin)
Properties	id, ownerId, title, description, price, location, images, amenities, availability
Bookings	id, propertyId, renterId, startDate, endDate, status
Reviews	id, propertyId, userId, rating, comment

---

5. Component Design (Frontend UI)

Page	Components
Home Page	Search Bar, Featured Listings, Navigation
Property Listing Page	Filters, Property Cards, Pagination
Property Detail Page	Gallery, Amenities List, Booking Form, Reviews Section
Login/Register Page	Forms, OAuth Buttons
Dashboard (User/Owner)	My Bookings, My Listings, Profile Settings
Admin Panel	User Management, Property Approvals, Reports

---

6. API Endpoints

Endpoint	Purpose
POST /api/auth/register	Register a new user
POST /api/auth/login	Login user
GET /api/properties	Get all property listings
GET /api/properties/:id	Get single property details
POST /api/bookings	Create new booking
GET /api/bookings/user/:id	Get user's bookings
POST /api/reviews	Submit a review

---

7. Security Considerations

- Passwords hashed using **bcrypt**
- Secure JWT tokens for authentication

- Input validation on server-side and client-side
- Admin access control

---

8. Design Principles

- **Responsive Design:** Mobile-first using TailwindCSS
- **Performance First:** Lazy loading images, code splitting
- **Scalability:** Clean, modular codebase ready for microservices in future phases
- **User-Centric:** Simple UI, fast load, intuitive navigation