This is a fantastic start! Your project already demonstrates solid proficiency in the **MERN** (or rather, MEN + EJS) stack. Implementing a full authentication flow (especially the "Forgot Password" via email) is a significant milestone that many junior developers struggle with. The UI also looks consistent and modern.

To elevate this project from a "class assignment" level to a "hired professional" level, we need to focus on **Real-World Utility**, **System Architecture**, and **Advanced Features (ML)**.

Here is a roadmap to make your Airbnb clone impactful for your resume:

---

1. Functional Improvements (Full-Stack Engineering)

To make this project "resume-ready," it needs to function like a real product.

- **Map Integration (Crucial for Real Estate):**
  - **What:** Instead of just text locations ("Goa, India"), show the homes on an interactive map.
  - **How:** Use **Mapbox** (free tier is generous) or Google Maps API. When a host adds a home, convert their address into coordinates (Geocoding) and store it in MongoDB (`GeoJSON` format).
  - **Why:** It shows you know how to work with 3rd-party APIs and geospatial data.
- **Payment Gateway Integration:**
  - **What:** The "Booking" button should actually process a transaction.
  - **How:** Integrate **Stripe** or **Razorpay** (in Test Mode). You don't need real money; just show that you can handle the secure token exchange between the client, your server, and the payment provider.
  - **Resume Win:** "Implemented secure payment processing using Stripe API."
- **Advanced Search & Filtering:**
  - **What:** Allow users to filter by Price Range, Amenities (AC, Pool, WiFi), and specific dates.
  - **Tech:** This requires complex MongoDB queries using operators like `$gte` (greater than), `$lte` (less than), and `$in`.

---

2. Machine Learning (ML) Integrations

Since you specifically asked about ML, this is the best way to make your project stand out. Since your backend is Node.js, you have two options: use **TensorFlow.js** (run ML in Node) or build a small **Python Flask/FastAPI microservice** (industry standard).

Here are three achievable ML features:

A. Smart Price Predictor (Regression)

- **The Feature:** When a **Host** is adding a home, help them decide the price. "Based on your location (Manali) and amenities (Pool, WiFi), we recommend listing this at ₹3,500/night."
- **The Model:** A simple **Linear Regression** or **Decision Tree**.
- **Data:** Train it on a dummy dataset of house prices (you can get CSVs from Kaggle).
- **Math:** $Price = w_1(Location) + w_2(Bedrooms) + w_3(Amenities) + b$

B. Recommendation System (Content-Based Filtering)

- **The Feature:** On the "Home Details" page, show a section: *"You might also like..."*
- **The Logic:** If the user is looking at a "Cabin in Manali," suggest other cabins or properties in hill stations.
- **Technique:** Use **Cosine Similarity**. Compare the features of the current house vector with other house vectors in your database.

C. Image Classification (Content Moderation)

- **The Feature:** When a host uploads a photo, automatically tag it or check it.
- **Use Case:** If a user uploads a picture of a bedroom, the model detects it and tags it "Bedroom." If they upload something inappropriate, the system rejects it.
- **How:** Use a pre-trained model like **MobileNet** via TensorFlow.js.

---

3. Architecture & Deployment (The "Senior" Touch)

A project running on `localhost:3000` is good, but a deployed project is better.

- **Deployment:** Deploy the backend on **Render** or **Railway** (easier for Node) and the database on **MongoDB Atlas**.
- **Code Quality:**
  - **MVC Pattern:** Ensure your Models, Views, and Controllers are strictly separated.
  - **Validation:** Use a library like **Joi** or **Zod** to validate user inputs (prevent malicious data) before it hits your database.
  - **Error Handling:** Create a global error handling middleware so the app doesn't crash if a query fails.

---

4. How to Present This on Your Resume

Don't just list the technologies. Describe the **Business Value** and **Complexity**.

Project Title: WanderLust (Property Rental Platform with AI Pricing)

- **Full Stack Development:** Engineered a RESTful architecture using **Node.js/Express**, handling authenticated sessions and secure password recovery flows via **Nodemailer**.
- **Machine Learning Integration:** Developed a Python microservice using **Scikit-Learn** to predict optimal rental pricing based on property features, increasing potential host revenue functionality.
- **Database Management:** Utilized **MongoDB** complex aggregations to implement geospatial search, allowing users to find properties within a specific radius.
- **Payment Processing:** Integrated **Stripe API** to handle test-mode transactions, ensuring PCI-compliant data handling.

---

5. Specific UX Feedback based on Screenshots

- **Screenshot 1 (Home Cards):** The cards look great. Consider adding a "Heart" icon directly on the card (top right corner) for quick favoriting without opening details.
- **Screenshot 3 (Email):** The link is raw (`http://localhost...`). In your EJS email template, wrap this in a nice button tag `<a href="..." class="btn">Reset Password</a>` so the user doesn't see the long ugly URL.
- **Screenshot 6 (Host Form):** The "Description" box is small. Use a Rich Text Editor or a larger `<textarea>` so hosts can write detailed descriptions.