Hostel Finder MVP — Technical Development Roadmap

This document outlines the complete technical development roadmap for the Hostel Finder MVP. The goal is to build, deploy, and launch a functional application in 2 weeks (with a 2-day buffer). This document includes ERD, flow diagrams (described), UI components mapping, backend architecture, security design, and deployment strategy.

# 1. Development Timeline (2 Weeks)

* Week 1: Authentication, User Profiles, Hostel CRUD, Room Management, Basic Search, Safety Modal.
* Week 2: Interaction Logs, Reviews & Reports, Verification Requests, Admin Dashboard, Analytics, Testing & Deployment.
* Buffer (max 2 days): Bug fixing & polish.

# 2. Entity Relationship Diagram (ERD)

Below is the ERD describing entities, attributes, and relationships for the system:

USERS (id, name, email, password\_hash, gender, phone, role, created\_at)  
HOSTELS (id, owner\_id, name, address, description, video\_url, verified, created\_at)  
ROOMS (id, hostel\_id, capacity, beds\_available, rent, facilities)  
REVIEWS (id, hostel\_id, user\_id, rating, comment, created\_at)  
INTERACTION\_LOG (id, user\_id, hostel\_id, type, safety\_confirmed, created\_at)  
REPORTS (id, hostel\_id, user\_id, reason, created\_at)  
VERIFICATION\_REQUESTS (id, hostel\_id, owner\_id, status, created\_at)  
FAVORITES (id, user\_id, hostel\_id, created\_at)  
  
Relationships:  
- USERS → HOSTELS (1–many, owner)  
- HOSTELS → ROOMS (1–many)  
- USERS → REVIEWS (1–many), HOSTELS → REVIEWS (1–many)  
- USERS → INTERACTIONS (1–many), HOSTELS → INTERACTIONS (1–many)  
- USERS → REPORTS (1–many), HOSTELS → REPORTS (1–many)  
- USERS → FAVORITES (1–many), HOSTELS → FAVORITES (1–many)  
- USERS → VERIFICATION\_REQUESTS (1–many), HOSTELS → VERIFICATION\_REQUESTS (1–many)

# 3. System Flow Diagram

High-level flow of how the system works:  
  
1. User signs up → selects role (student/owner/admin) → profile created.  
2. Students: Search → View Hostel Details → Safety Modal → WhatsApp/Call → Interaction logged.  
3. Owners: Register Hostel → Add Rooms → Submit Verification Request → View Stats.  
4. Admin: Approve/Reject Verification → Monitor Reports → Export Analytics.  
5. Safety Modal (female only) ensures trust before interaction.

# 4. UI Components Mapping

Frontend pages and their UI components:

* Landing/Home: SearchBar, FeaturedHostelCard
* Search Results: SearchBar, MapView, HostelCard
* Hostel Detail: HostelHeader, VideoPlayer, FacilitiesList, RoomCard, ContactButtons, ReportButton, ReviewsList, AddReviewForm
* Owner Dashboard: HostelList, OwnerCreateHostelForm, OwnerCreateRoomForm, OwnerStatsPanel, VerificationRequestForm
* Admin Dashboard: AdminReportQueue, VerificationRequestsList, AnalyticsPanel, ExportCSVButton
* Auth & Profile: SignupForm (gender required), LoginForm, ProfilePage

# 5. Backend & Infrastructure

- Framework: FastAPI (Python) or Django REST Framework  
- Database: PostgreSQL (via Supabase)  
- File Storage: Supabase Storage (images/videos)  
- Hosting: Frontend → Vercel/Netlify; Backend → Render/DigitalOcean  
- Email: SendGrid or Gmail SMTP (for verification/notifications)  
- Maps: Google Maps API (preferred) or OpenStreetMap

# 6. Security & Safety Features

- Authentication: JWT tokens, bcrypt password hashing.  
- Safety Modal: Mandatory for female users before hostel interaction.  
- Authorization: Role-based access (Student, Owner, Admin).  
- Reports: Students can report hostels; Admin reviews.  
- Verification Requests: Owners submit → Admin approval required.  
- Data Privacy: Logs stored securely, Supabase storage with access rules.  
- Rate Limiting: Protect APIs from abuse.  
- Backups: Scheduled DB backups.

# 7. Testing Checklist

* Signup/login works, gender required.
* Search → Detail → Safety Modal → Contact flow works.
* Owner CRUD + verification request works.
* Admin can approve verification & see reports.
* Interaction logging works with/without safety confirmation.
* Mobile responsiveness tested.
* Deployment working with live database and storage.