OnlyPGs – Unified Phase 1 Product & Technical Specification

Product Name (exact): OnlyPGs
Phase: Phase 1 (Locked Scope)
Goal: A clean, no-noise platform to discover and list real PG (Paying Guest) accommodations in Indian cities.

---

1. Product Philosophy

- Simplicity over features
- Trust through manual verification
- Search-first, conversion-focused (Call / WhatsApp)
- No marketing banners, no ads, no distractions
- Mobile-first, desktop-optimized
- Clean data > feature count

---

2. Phase 1 City Coverage (Locked)

Enabled from Day 1: - Bangalore - Delhi - Noida - Gurgaon - Pune - Hyderabad - Chennai - Mumbai - Indore - Jaipur

Design rule: UI and architecture must not change when new cities are added.

---

3. Core Design Principles

- Minimal visual noise
- Clear information hierarchy
- Fast scanning of listings
- Outdoor-readable (Indian mobile usage)
- Strong CTA emphasis (Call / WhatsApp)

---

4. Color Palette (Locked)

Base Theme

A calm, trustworthy, utility-first palette.

Colors: - `#0D1B2A` – Primary text / headings (deep navy) - `#1B263B` – Secondary text - `#415A77` – Secondary accent / UI emphasis - `#778DA9` – Borders, dividers, subtle highlights - `#E0E1DD` – Main background (warm off-white)

Usage Rules

- Backgrounds: `#E0E1DD`
- Cards: pure white
- Primary text: `#0D1B2A`
- Secondary text: `#1B263B`
- Borders: `#778DA9` at low opacity
- Accent usage: limited to buttons, active filters, highlights

No gradients. No playful colors. No marketing colors.

---

5. Page Inventory (Phase 1)

User-Facing Pages

1. Home / Landing Page
2. Search Results Page
3. PG Details Page
4. Add a PG (Multi-step Form)
5. Submission Success Page

Admin Pages

1. Admin Login Page
2. Admin Dashboard
3. Admin PG Review Page

---

6. Page-Level Design & Behavior

6.1 Home / Landing Page

Goal: Start search immediately.

Desktop: - Header: Logo (left), Add a PG (right) - Hero search: City (mandatory), PG name (optional), Search button - Optional popular localities

Mobile: - Stacked search inputs - Fixed Add a PG button at bottom

---

6.2 Search Results Page

Goal: Fast scanning and filtering.

Filters: - Gender (Boys / Girls / Unisex) - Room Type (Single / Double / Triple / 4-sharing) - Price (activated after room type) - Food (Yes / No)

Desktop: - Left sticky filter sidebar - Right results list (cards)

Mobile: - Filters behind button - Vertical cards - No images in cards

---

6.3 PG Details Page

Goal: Help user decide to contact.

- PG name + area
- Photo gallery (horizontal)
- Room options with fixed prices
- Facilities (icon grid)
- Rules
- Contact section: Call / WhatsApp

Mobile: Sticky bottom bar with Call + WhatsApp

6.4 Add a PG (Multi-Step Form)

Steps: 1. Who are you? (Owner / Manager / Tenant) 2. Basic details 3. Room options & fixed prices 4. Facilities 5. Rules (optional) 6. Photos (3–6) 7. Contact details 8. Review & submit

No login required.

6.5 Submission Success Page

Simple confirmation message:

> "Your PG has been submitted and is under verification."

No further actions.

---

7. Admin Pages

7.1 Admin Dashboard

- Summary cards: Total, Pending, Approved, Rejected
- Filters: City, Status
- Search by PG name
- Listings table with actions

7.2 Admin PG Review Page

- Full PG details
- All photos
- Contact info
- Actions: Approve / Reject

---

8. URL Structure

Public Routes

- `/` – Home
- `/search?city=...&pgName=...`
- `/pg/:pgId`
- `/add-pg`
- `/submission-success`

Admin Routes

- `/admin/login`
- `/admin/dashboard`
- `/admin/pg/:pgId`

---

9. Data Architecture

Tech Stack

- Backend: Node.js
- Database: MongoDB Atlas
- Image Storage: AWS S3

---

10. MongoDB Schema (Phase 1)

Pricing Rule (Important – Locked)

❗ **Prices are NOT stored generically at PG level.**
❗ **Every price is tied strictly to a room type for a specific PG.**

This ensures: - No misleading price ranges - Clear comparison across PGs - Accurate filtering by room type

All pricing lives **only** in the `rooms` collection.

Collections

pgs

```
{
  _id: ObjectId,
  pgName: String,
  city: String,
  area: String,
  landmark: String,
  genderType: "boys" || "girls" || "unisex",
  securityDeposit: Number,
  facilities: {
    food: Boolean,
    wifi: Boolean,
    washingMachine: Boolean,
    roomCleaning: Boolean,
    powerBackup: Boolean,
    ac: Boolean,
    attachedWashroom: Boolean
  },
  rules: {
    nightEntryTime: String,
    visitorPolicy: "allowed" || "not_allowed",
    smokingDrinking: "allowed" || "not_allowed"
  },
  photos: [String],
  contact: {
    phone: String,
    whatsapp: String
  },
  status: "pending" || "approved" || "rejected",
  createdAt: Date,
```

```
  updatedAt: Date
}
```

rooms

```
{
  _id: ObjectId,
  pgId: ObjectId,              // Reference to pgs._id
  roomType: "single" | "double_sharing" | "triple_sharing" | "four_sharing",
  price: Number                // Fixed monthly price for THIS room type
}
```

Rules (Strict): - Each room type has exactly ONE fixed price - No min–max ranges allowed - A PG may offer any subset of room types - Minimum 1 room type per PG - Maximum 4 room entries per PG - Prices must always be queried via room type json { _id: ObjectId, pgId: ObjectId, roomType: "single" | "double_sharing" | "triple_sharing" | "four_sharing", price: Number }

```
#### submissions
```json
{
  _id: ObjectId,
  pgId: ObjectId,
  submittedBy: "owner" | "manager" | "tenant",
  submittedAt: Date
}
```

admins

```
{
  _id: ObjectId,
  name: String,
  email: String,
  passwordHash: String,
  createdAt: Date
}
```

---

11. Search & Indexing Rules

Indexes: - `pgs`: `{ city: 1, status: 1 }` - `pgs`: text index on `pgName` - `rooms`: `{ pgId: 1 }` - `rooms`: `{ roomType: 1, price: 1 }`
```

Search flow: 1. Filter rooms by room type + price 2. Collect pgIds 3. Query pgs by city, gender, status=approved 4. Merge results

---

12. Image Storage (AWS S3)

Bucket

- Name: `onlypgs-images`
- Region: `ap-south-1 (Mumbai)`

Rules

- Images stored in S3
- MongoDB stores only public URLs
- 3–6 images per PG
- Max 5MB per image

Folder Structure

```
onlypgs-images/
  pg/
    <pgId>/
      image1.jpg
      image2.jpg
```

Permissions

- Public read via bucket policy
- Upload via restricted IAM user only

---

13. Environment Variables (to be provided separately)

```
MONGODB_URI=
AWS_ACCESS_KEY_ID=
AWS_SECRET_ACCESS_KEY=
AWS_REGION=ap-south-1
AWS_S3_BUCKET=onlypgs-images
```

Credentials are never hardcoded.

---

14. Security & Production Notes

- Restrict MongoDB IPs before launch
- Use dedicated app DB user in prod
- Rotate AWS keys if leaked
- Store secrets in hosting provider env panel

---

15. Phase 1 Success Metrics

- Number of approved PGs
- Searches per day
- Call / WhatsApp clicks
- City coverage quality

Status: Unified Phase 1 PRD – Locked