

GlowUpNow

Overview

- **[System Name]** GlowUpNow
- **[What it is]** A full-stack web application for at-home salon and spa service discovery, booking, and admin operations (service management, analytics, and user oversight).
- **[Primary Users]**
 - **Customers:** Browse services, filter/search, view featured items, and create bookings.
 - **Admins:** Manage services (CRUD + image uploads), monitor bookings via analytics, and view registered users.

Core Capabilities

- **[Public landing]** Category browsing, search and price filters, featured services, testimonials. See `frontend/src/pages/Home.jsx`.
- **[Authentication]** Secure login, role-based navigation, session handling.
- **[Service Management (Admin)]** Create/update/delete services, upload images, filter by category/price/duration, paginate. See `frontend/src/pages/ServiceManagement.jsx`.
- **[Admin Analytics]** Summary KPIs (bookings, revenue, averages), service coverage (bookings per service), bookings overview with search, status filters, pagination, and quick-view modal. See `frontend/src/pages/Dashboard.jsx`.
- **[Admin Users]** Read-only user directory with search and pagination. See `frontend/src/pages/AdminUsers.jsx`.

Architecture & Tech Stack

- **[Frontend]** React (Vite) SPA with Bootstrap styling and custom hooks (`frontend/src/hooks/`).
- **[Backend]** PHP 8 MVC with controllers/models/core under `backend/app/`, routes in `backend/routes/api.php`.
- **[Database]** MySQL/MariaDB. Schema and seed in `database/glowupnow.sql`.
- **[APIs]** JSON over HTTP (REST-ish) via `backend/public/` (e.g., `/api/services`, `/api/bookings`, `/api/analytics/*`).
- **[Hosting/Local]** Designed for XAMPP/Apache; frontend served by Vite dev or static build.

Data Model (High Level)

- **[users]** Accounts with roles (admin/customer).
- **[categories]** Normalized service categories.
- **[services]** Service catalog with price, duration, image, active flag. Includes category_id (normalized); a legacy category string may remain until normalization is completed.
- **[bookings]** Customer bookings linked to users and services; status lifecycle.
- **[booking_status_history]** Audit trail for status changes (who/when).
- **[audit_logs]** Optional event logging with JSON metadata.

Security & Quality

- **[Security]** Password hashing; prepared statements with PDO; role-based route protection (auth/admin middlewares).
- **[Performance]** Composite indexes on bookings by date/status/service/user; service/category indexes for filters.
- **[UX]** Responsive layout, collapsible admin sidebar with tooltips, focus/blur validation feedback, image skeleton loaders, dbl-click quick view modals.

Entity-Relationship Diagram (ERD)

This diagram shows the core entities and their relationships. The system records users, services, and bookings, with a status history audit.

users ||--o{ bookings : makes

users ||--o{ booking_status_history : changes

services ||--o{ bookings : is_booked

bookings ||--o{ booking_status_history : has

categories ||--o{ services : classifies

```
users {  
  int id PK  
  varchar name  
  varchar email UNIQUE  
  varchar role  
  varchar phone  
  datetime created_at  
}
```

```
categories {  
  int id PK  
  varchar name UNIQUE  
  datetime created_at  
}
```

```
services {  
  int id PK  
  varchar name
```

```
varchar category      // legacy text field (to remove during normalization)
int category_id FK    // normalized foreign key
decimal price
int duration_minutes
varchar image_path
tinyint is_featured
tinyint active
datetime created_at
}
```

```
bookings {
int id PK
int user_id FK
int service_id FK
date date
time start_time
time end_time
enum status
varchar notes
datetime created_at
}
```

```
booking_status_history {
int id PK
int booking_id FK
enum old_status
enum new_status
int changed_by_user_id FK
datetime changed_at
}
```

Database Schema Summary

- users: id PK, email UNIQUE, role, phone, created_at.
- categories: id PK, name UNIQUE.
- services: id PK, name, price, duration_minutes, image_path, is_featured, active, category_id FK → categories.id; legacy `category` string retained (see normalization plan).
- bookings: id PK, user_id FK → users.id (CASCADE), service_id FK → services.id; date/time, status enum, notes; indexes on (date,status), (status,date), (service_id,date), (user_id,date).
- booking_status_history: id PK, booking_id FK (CASCADE), changed_by_user_id FK (SET NULL), old_status → new_status, changed_at.
- audit_logs: id PK, user_id FK (SET NULL), action, entity, entity_id, metadata JSON CHECK, created_at.

Key SQL Queries and Explanations

This document lists key queries used by the analytics and management features and briefly explains each, with sample outputs.

Summary Metrics (Admin Analytics)

Location: `backend/app/controllers/DashboardController.php::summary()`

```
```sql
```

```
-- Status counts per date range
```

```
SELECT b.status, COUNT(*) AS cnt
```

```
FROM bookings b
```

```
WHERE b.date BETWEEN :from AND :to
```

```
GROUP BY b.status;
```

```
-- Revenue per category (via services join)
```

```
SELECT s.category AS category, COALESCE(SUM(s.price), 0) AS revenue
```

```
FROM bookings b
```

```
INNER JOIN services s ON s.id = b.service_id
```

```
WHERE b.date BETWEEN :from AND :to
```

```
GROUP BY s.category
```

```
ORDER BY revenue DESC;
```

```
```
```

Sample output:

```
```
```

```
status cnt
```

```
----- ---
```

```
pending 10
```

confirmed 20

canceled 5

completed 7

...

...

category    revenue

-----

Massage    12000.00

Facial     8000.00

...

### Service Coverage

Location: `backend/app/controllers/DashboardController.php::serviceCoverage()`

```sql

SELECT s.id, s.name, COALESCE(COUNT(b.id),0) AS bookings

FROM bookings b

RIGHT JOIN services s ON s.id = b.service_id

WHERE (b.date BETWEEN :from AND :to OR b.id IS NULL)

GROUP BY s.id, s.name

ORDER BY bookings DESC, s.name ASC;

```

Sample output:

...

id   name                      bookings

-- -----

5   Swedish Full Body Massage    12

2   Deep Tissue Massage           8



...

### Bookings List (Filters + Pagination)

Location: `backend/app/controllers/DashboardController.php::bookingsList()`

```sql

```
SELECT b.id, b.date, b.start_time, b.end_time, b.status, b.notes, b.created_at,
s.name AS service_name, s.price, u.name AS customer_name
FROM bookings b
INNER JOIN services s ON s.id=b.service_id
INNER JOIN users u ON u.id=b.user_id
WHERE b.date BETWEEN :from AND :to
AND (:status = '' OR b.status = :status)
AND (
:q = '' OR
b.id = :idq OR
u.name LIKE :uq OR
s.name LIKE :sq
)
ORDER BY b.date DESC, b.start_time DESC
LIMIT :limit OFFSET :offset;
```

...

Sample output:

...

| id | date | start | end | status | customer_name | service_name | price |
|----|------------|-------|-------|---------|---------------|----------------|--------|
| 10 | 2025-11-05 | 14:05 | 15:05 | pending | Allexia Papa | Pedicure (Spa) | 650.00 |

9 2025-10-31 09:00 10:15 confirmed Allexia Papa Deep Tissue Massage 1400.00

...

Indexes Supporting Queries

- `bookings`: `idx_b_date_status`, `idx_b_status_date`, `idx_b_service_date`, `idx_b_user_date`.
- `services`: `idx_services_active`, `idx_services_price`, `idx_services_category`,
`idx_services_category_id`.
- `users`: `email` unique.
- `categories`: `name` unique.

Hooks and Events Integration

This document summarizes the event handlers and custom hooks implemented across the app for rubric alignment.

Custom Hooks (10 implemented)

- **useToggle** —

frontend/src/hooks/useToggle.js

Boolean toggle state. Used for Dense view in AdminUsers.jsx.

- **useForm** —

frontend/src/hooks/useForm.js

Controlled form state with validation and submit helper. Candidate to migrate ServiceManagement.jsx.

- **useFetch** —

frontend/src/hooks/useFetch.js

Async lifecycle wrapper (data, loading, error, reload). Used in AdminUsers.jsx.

- **useInterval** —

frontend/src/hooks/useInterval.js

Interval effect with cleanup. Ready for periodic analytics refresh.

- **useModal** —

frontend/src/hooks/useModal.js

Modal open/close with Escape handling and focus management. Used in ServiceManagement.jsx (delete confirm) and Dashboard.jsx (bookings quick-view).

- **useEventSource** —

frontend/src/hooks/useEventSource.js

SSE wrapper with connect/close and event handlers. Available for real-time features.

- **useAuthHook** —

frontend/src/hooks/useAuthHook.js

Derived auth flags (isLoggedIn, isAdmin) from AuthContext. Optional helper.

- **useClickOutside** — frontend/src/hooks/useClickOutside.js

Detects clicks outside a ref to close popovers/menus/modals. Used in ServiceManagement.jsx.

- **useDebounce** — frontend/src/hooks/useDebounce.js

Debounces a value or callback; helpful for search inputs and typing delays. Used in ServiceManagement.jsx.

- **useLocalStorage** — frontend/src/hooks/useLocalStorage.js
State persisted to localStorage. Used in AdminLayout.jsx to remember sidebar collapsed state.

Event Handlers

- **onClick**
Buttons across dashboards/forms; Edit/Delete actions; Search/Clear controls.
- **onChange**
All form inputs and filters (search, selects, date ranges, per-page).
- **onSubmit**
Forms like Service Management create/update.
- **onKeyDown / onKeyPress**
Enter-to-apply patterns in Service Management and Dashboard filters.
- **onFocus / onBlur**
 - Service Management: Name, Price, Duration show validation feedback after blur; Search highlights on focus.
 - Admin Analytics (Dashboard): From/To date inputs and Bookings search highlight on focus.
- **onDoubleClick**
 - Service rows in Service Management open edit.
 - Bookings rows in Admin Analytics open a read-only quick-view modal.
- **onload**
 - ServiceCard images: skeleton shown until image loads.

File References

- **Service Management** —

frontend/src/pages/ServiceManagement.jsx

- Focus/blur validation on Name/Price/Duration, focus highlight on Search.
- useModal for delete confirm with Escape and outside-click close.
- useClickOutside and useDebounce integrated for UX.
- Double-click on a service row opens edit.

- **Admin Analytics** —

frontend/src/pages/Dashboard.jsx

- Focus/blur highlight on From/To date inputs and Bookings search.
- Quick-view modal (

useModal) opened via double-click on bookings row.

- **Admin Users** —

frontend/src/pages/AdminUsers.jsx

- useFetch for data loading.
- useToggle for Dense table view toggle.

- **Admin Sidebar** —

frontend/src/components/AdminSidebar.jsx

- Title tooltips added to NavLink items for better collapsed UX.

- **Admin Layout** —

frontend/src/components/AdminLayout.jsx

- useLocalStorage persists sidebar collapsed state.

- **Service Card** —

frontend/src/components/ServiceCard.jsx

- onLoad skeleton handling for images; optional Book action.

- **Hooks Index** —

frontend/src/hooks/index.js

- Central exports for all custom hooks.

Login Page

GlowUpNow

Home salon & spa services, on demand

Welcome back

Book massages, facials, nails, and hair services with trusted pros at your home.
Sign in to continue.

Secure

Protected authentication

Fast

Quick, seamless booking

Trusted

Verified professionals

Sign in

Register

Email

you@example.com

Password

Your password

Sign in

New here? Create an account

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

Register Page

GlowUpNow

Home salon & spa services, on demand

Join GlowUpNow

Create your account to book at-home salon & spa services, save preferences, and track your bookings.

Personal

Set your preferences

Tracking

Manage bookings & history

Perks

Access exclusive offers

Create account

Login

Name

Your full name

Email

you@example.com

Password

Create a password (min 8 characters)

Create account

Have an account? Sign in

By creating an account, you agree to our Terms and Privacy Policy.

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

Admin Analytics

GlowUpNow

Dashboard

Services

Bookings

Users

Logout

Signed in as Admin

Admin Analytics

Overview of bookings - 2025-10-20 ~ 2025-10-26

From

20/10/2025

To

26/10/2025

Apply

This Week

Last 7 Days

This Month

Total Bookings

3

Confirmed

0

Pending

1

Canceled

2

Total Revenue

P3100.00

Avg Revenue / Day

P442.86

Avg Duration (mins)

53.3

Status Breakdown

confirmed0

completed0

pending1

canceled2

Drag/hover over chart to preview

Category Breakdown

Facial3

Drag/hover over chart to preview

Revenue by Category

Facial3100

Daily Bookings

10-2010-2110-2

Summary

Total bookings: 3

Average/day: 1.0

Peak day: 2025-10-20 (1)

Top Services

Hydrating Glow Facial2

Classic Deep Cleansing Facial1

Service Coverage

| # | Service | Bookings |
|----|---------------------------------------|----------|
| 3 | Hydrating Glow Facial | 2 |
| 2 | Classic Deep Cleansing Facial | 1 |
| 4 | Acne Clarifying Facial | 0 |
| 7 | Aromatherapy Relaxation Massage | 0 |
| 20 | Bikini Waxing | 0 |
| 1 | Body Scrub | 0 |
| 16 | Brow Wax & Shape | 0 |
| 14 | Classic Eyelash Extensions (Full Set) | 0 |
| 6 | Deep Tissue Massage | 0 |
| 17 | Eyebrow Tinting | 0 |
| 15 | Eyelash Fill (2-3 weeks) | 0 |
| 19 | Full Leg Waxing | 0 |
| 10 | Gel Manicure | 0 |
| 12 | Hair Coloring (Full) | 0 |
| 13 | Keratin Smoothing Treatment | 0 |
| 18 | Lip Waxing | 0 |
| 8 | Manicure (Classic) | 0 |
| 9 | Pedicure (Spa) | 0 |
| 5 | Swedish Full Body Massage | 0 |
| 11 | Women's Haircut + Blow-dry | 0 |

Bookings Overview

Search

All Status

10

Export CSV

| # | Date | Start | End | Status | Customer | Service | Price | Notes |
|---|------------|-------|-------|----------|--------------|-------------------------------|----------|-------|
| 7 | 2025-10-22 | 13:00 | 14:00 | canceled | Allexia Papa | Classic Deep Cleansing Facial | P1100.00 | |
| 8 | 2025-10-21 | 11:00 | 11:50 | pending | Allexia Papa | Hydrating Glow Facial | P1000.00 | |
| 6 | 2025-10-20 | 10:00 | 10:50 | canceled | Allexia Papa | Hydrating Glow Facial | P1000.00 | |

Showing 1-3 of 3

Prev

Page 1 / 1

Next

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

Service Management

GlowUpNow

<

Dashboard

Services

Bookings

Users

Logout

Signed in as Admin

Service Management

Name

Category

Select category

Price

Duration (mins)

Image

Choose FileNo file chosen

Active

Yes

Description

Create Service

Search

Search services

Category

All

Active

All

Per page

10

Price

Min

Max

Duration (mins)

Min

Max

Clear search/filters

| # | Name | Category | Price | Duration | Active | |
|----|---------------------------------|---------------|----------|----------|--------|-----------------------|
| 2 | Classic Deep Cleansing Facial | Facial | ₱1100.00 | 60 mins | Yes | <div>EditDelete</div> |
| 3 | Hydrating Glow Facial | Facial | ₱1000.00 | 50 mins | Yes | <div>EditDelete</div> |
| 4 | Acne Clarifying Facial | Facial | ₱1300.00 | 60 mins | Yes | <div>EditDelete</div> |
| 5 | Swedish Full Body Massage | Massage | ₱900.00 | 60 mins | Yes | <div>EditDelete</div> |
| 6 | Deep Tissue Massage | Massage | ₱1400.00 | 75 mins | Yes | <div>EditDelete</div> |
| 7 | Aromatherapy Relaxation Massage | Massage | ₱1100.00 | 60 mins | Yes | <div>EditDelete</div> |
| 8 | Manicure (Classic) | Hand and Foot | ₱450.00 | 40 mins | Yes | <div>EditDelete</div> |
| 9 | Pedicure (Spa) | Hand and Foot | ₱650.00 | 50 mins | Yes | <div>EditDelete</div> |
| 10 | Gel Manicure | Hand and Foot | ₱700.00 | 60 mins | Yes | <div>EditDelete</div> |
| 11 | Women's Haircut + Blow-dry | Hair | ₱800.00 | 60 mins | Yes | <div>EditDelete</div> |

Showing 1–10 of 20

Prev

Page 1 / 2

Next

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

Booking Management

GlowUpNow

Dashboard

Services

Bookings

Users

Logout

Signed in as Admin

Booking Management

Booking ID

New Status

Confirmed

Update Status

Cancel

#

Date

Time

Customer

Service

dd/mm/yyyy

--:-- --

Name

Service

Status

Per page

All

10

Clear search/filters

| # | Date | Time | Customer | Service | Status | Actions |
|----|------------|---------------------|---------------|-------------------------------|-----------|---------|
| 10 | 2025-11-05 | 14:05:00 - 15:05:00 | Allexia Papa | Acne Clarifying Facial | pending | Edit |
| 9 | 2025-10-31 | 09:00:00 - 10:15:00 | Allexia Papa | Deep Tissue Massage | confirmed | Edit |
| 1 | 2025-10-30 | 09:00:00 - 10:00:00 | Administrator | Body Scrub | canceled | Edit |
| 3 | 2025-10-29 | 09:00:00 - 10:00:00 | Kate Molato | Body Scrub | confirmed | Edit |
| 7 | 2025-10-22 | 13:00:00 - 14:00:00 | Allexia Papa | Classic Deep Cleansing Facial | canceled | Edit |
| 8 | 2025-10-21 | 11:00:00 - 11:50:00 | Allexia Papa | Hydrating Glow Facial | pending | Edit |
| 6 | 2025-10-20 | 10:00:00 - 10:50:00 | Allexia Papa | Hydrating Glow Facial | canceled | Edit |
| 4 | 2025-10-16 | 09:00:00 - 10:00:00 | Kate Molato | Body Scrub | confirmed | Edit |
| 5 | 2025-10-10 | 15:00:00 - 15:30:00 | Allexia Papa | Bikini Waxing | pending | Edit |
| 2 | 2025-10-05 | 09:00:00 - 10:00:00 | Kate Molato | Body Scrub | confirmed | Edit |

Showing 1-10 of 10

PrevPage 1 / 1Next

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

User Management

GlowUpNow

<

Dashboard

Services

Bookings

Users

Logout

User Management

Search

name, email, role

Per page10

DenseOff

Loading...

| # | Name | Email | Role | Phone | Joined |
|---|---------------|-------------------|----------|-------------|---------------------|
| 3 | Allexia Papa | allexia@gmail.com | customer | - | 2025-10-20 01:57:40 |
| 2 | Administrator | admin@gmail.com | admin | - | 2025-10-19 23:44:37 |
| 1 | Kate Molato | kate@gmail.com | customer | 98675645235 | 2025-10-19 22:52:16 |

Showing 1-3 of 3

PrevPage 1 / 1Next

Signed in as Admin

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

User Dashboard

GlowUpNow


DashboardBookingsProfileHi, Alexia PappLogout

Discover Services

Search servicesPriceAll categoriesSearchClear

Featured


Top 6



Classic Deep Cleansing Facial

Deep pore cleansing with exfoliation and mask for radiant skin.


P1100.0060 minsBook



Hydrating Glow Facial

Moisture-rich treatment to nourish dry and dull skin for a dewy finish.


P1000.0050 minsBook



Acne Clarifying Facial

Targeted treatment for acne-prone skin including extraction and antibacterial mask.


P1300.0060 minsBook



Swedish Full Body Massage

Gentle to medium pressure massage to relax muscles and improve circulation.


P900.0060 minsBook



Deep Tissue Massage

Focused, firm pressure to release chronic muscle tension and knots.

P1400.0075 minsBook




Aromatherapy Relaxation Massage

Warm aromatic oils and soothing strokes to calm the mind and body.

P1100.0060 minsBook

All Services


20 results



Classic Deep Cleansing Facial

Deep pore cleansing with exfoliation and mask for radiant skin.


P1100.0060 minsBook



Hydrating Glow Facial

Moisture-rich treatment to nourish dry and dull skin for a dewy finish.


P1000.0050 minsBook



Acne Clarifying Facial

Targeted treatment for acne-prone skin including extraction and antibacterial mask.


P1300.0060 minsBook



Swedish Full Body Massage

Gentle to medium pressure massage to relax muscles and improve circulation.


P900.0060 minsBook



Deep Tissue Massage

Focused, firm pressure to release chronic muscle tension and knots.


P1400.0075 minsBook



Aromatherapy Relaxation Massage

Warm aromatic oils and soothing strokes to calm the mind and body.


P1100.0060 minsBook



Manicure (Classic)

Nail shaping, cuticle care and polish application for neat hands.


P450.0060 minsBook



Pedicure (Spa)

Soak, exfoliation, nail care and hydrating foot massage.


P650.0050 minsBook



Gel Manicure

Long-lasting gel polish cured under LED for durable shine.


P700.0060 minsBook



Women's Haircut + Blow-dry

Professional haircut followed by styling and blow-dry.


P800.0050 minsBook



Hair Coloring (Full)

Full hair color service including consultation and tones.

P2200.00120 minsBook



Keratin Smoothing Treatment

Frizz control and smoothing treatment for sleek hair lasting weeks.

P3500.00150 minsBook

Load more

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

Bookings

GlowUpNow

Dashboard

Bookings

Profile

Hi, Alexia Papa

Logout

Bookings

Select a service from your Dashboard to start booking.

Your Bookings

#

Date

Time

Status

Per page

10

Search

Clear

| # | Date | Time | Service | Status | Actions |
|----|------------|---------------------|-------------------------------|-----------|-----------------------------------|
| 10 | 2025-11-05 | 14:05:00 - 15:05:00 | Acne Clarifying Facial | pending | <div>Edit</div> <div>Cancel</div> |
| 9 | 2025-10-31 | 09:00:00 - 10:15:00 | Deep Tissue Massage | confirmed | <div>Cancel</div> |
| 7 | 2025-10-22 | 13:00:00 - 14:00:00 | Classic Deep Cleansing Facial | canceled | |
| 8 | 2025-10-21 | 11:00:00 - 11:50:00 | Hydrating Glow Facial | pending | <div>Edit</div> <div>Cancel</div> |
| 6 | 2025-10-20 | 10:00:00 - 10:50:00 | Hydrating Glow Facial | canceled | |
| 5 | 2025-10-10 | 15:00:00 - 15:30:00 | Bikini Waxing | pending | <div>Edit</div> <div>Cancel</div> |

Showing 1-6 of 6

Prev

Page 1 / 1

Next

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

My Profile

GlowUpNow

[Dashboard](#)[Bookings](#)[Profile](#)Hi, [Allexia Papa](#)[Logout](#)

My Profile

Name

Allexia Papa

Email

allexia@gmail.com

Phone

Save

GlowUpNow

Home salon & spa services, on demand.

© 2025 GlowUpNow. All rights reserved.

Setup Guide

Requirements

- Node.js 18+
- npm
- XAMPP (Apache + MariaDB/MySQL)
- PHP 8+

Project structure

- Frontend (React): ``frontend/``
- Backend (PHP MVC): ``backend/``
- Database SQL: ``database/ glowupnow.sql``
- Docs: ``docs/``

Backend (Apache)

1. Ensure Apache serves ``backend/public/`` under a URL. With XAMPP htdocs, you can place the project so that this URL works:

- ``http://localhost/ glowupnow/ backend/ public``

2. Confirm by opening: ``http://localhost/ glowupnow/ backend/ public/ api/ health``

Database

1. Open phpMyAdmin: ``http://localhost/ phpmyadmin``
2. Create database ``glowupnow`` if missing.
3. Import ``database/ glowupnow.sql``.
4. Verify tables, indexes and foreign keys are present (see ``database/ glowupnow.sql``).

Frontend

1. Open a terminal in ``frontend/``.
2. Install dependencies: `npm install`

3. Set API base URL if needed:

- Create `.env` in `frontend/`` with: `VITE_API_BASE_URL=http://localhost/``glowupnow/backend/public

4. Run dev server: `npm run dev`

5. Visit the site (as configured by Vite output, usually `http://localhost:5173``).

Admin login

- Default admin user from SQL seed (see `database/``glowupnow.sql`):

- Email: `admin@gmail.com``

- Password: (the hash exists in seed; set/update via app if needed) `/admin123``

File uploads

- Service images are stored under `backend/public/uploads/services/``.

- When creating/updating a service with an image, the backend saves to this folder and sets `image_path`` (e.g. `/uploads/services/<file>.jpg``).

Normalization Plan: `services.category` → `services.category_id`

Goal: enforce strict 3NF by using a foreign key to `categories.id` and dropping the legacy `services.category` (varchar) column after data migration.

Steps

1) Ensure all service categories exist in `categories`

```
INSERT INTO categories (name, created_at)
SELECT DISTINCT s.category, NOW()
FROM services s
LEFT JOIN categories c ON c.name = s.category
WHERE s.category IS NOT NULL AND s.category <> "" AND c.id IS NULL;
```

2) Map services to `category_id` via name

```
UPDATE services s
INNER JOIN categories c ON c.name = s.category
SET s.category_id = c.id
WHERE s.category IS NOT NULL AND s.category <> "";
```

3) Verify there are no unmapped categories

```
SELECT id, name, category, category_id
FROM services
WHERE (category IS NOT NULL AND category <> "")
AND (category_id IS NULL);
```

4) Update the application to use `category_id` exclusively

- Frontend: replace string-based category filters with `category_id` where applicable.

- Backend: service queries should reference `category_id` and `categories.name` via JOIN.

5) Drop legacy column after verification

```
ALTER TABLE services DROP COLUMN category;
```

Rollback (if needed)

```
ALTER TABLE services ADD COLUMN category VARCHAR(64) NULL AFTER name;
```

```
UPDATE services s
```

```
INNER JOIN categories c ON c.id = s.category_id
```

```
SET s.category = c.name
```

```
WHERE s.category IS NULL;
```

Source Code Compilation

This document embeds the core application source files (frontend React and backend PHP). It excludes third-party dependencies and binary assets. For very large files, if content is abbreviated, refer to the original path in the repository.

Contents

- Frontend (React)

- `frontend/src/App.jsx`

- Components

- `frontend/src/components/AdminLayout.jsx`
- `frontend/src/components/AdminSidebar.jsx`
- `frontend/src/components/BookingForm.jsx`
- `frontend/src/components/BookingList.jsx`
- `frontend/src/components/Navbar.jsx`
- `frontend/src/components/ProtectedRoute.jsx`
- `frontend/src/components/ServiceCard.jsx`
- `frontend/src/components/SiteFooter.jsx`

- Contexts

- `frontend/src/context/AuthContext.jsx`

- Hooks

- `frontend/src/hooks/index.js`
- `frontend/src/hooks/useAuthHook.js`
- `frontend/src/hooks/useClickOutside.js`
- `frontend/src/hooks/useDebounce.js`
- `frontend/src/hooks/useEventSource.js`
- `frontend/src/hooks/useFetch.js`
- `frontend/src/hooks/useForm.js`
- `frontend/src/hooks/useInterval.js`

- `frontend/src/hooks/useLocalStorage.js`
- `frontend/src/hooks/useModal.js`
- `frontend/src/hooks/useToggle.js`
- Pages
 - `frontend/src/pages/AdminUsers.jsx`
 - `frontend/src/pages/BookingManagement.jsx`
 - `frontend/src/pages/Bookings.jsx`
 - `frontend/src/pages/Dashboard.jsx`
 - `frontend/src/pages/Home.jsx`
 - `frontend/src/pages/Login.jsx`
 - `frontend/src/pages/ServiceManagement.jsx`
 - `frontend/src/pages/UserDashboard.jsx`
 - `frontend/src/pages/UserProfile.jsx`
- Services
 - `frontend/src/services/analyticsService.js`
 - `frontend/src/services/authService.js`
 - `frontend/src/services/bookingService.js`
 - `frontend/src/services/http.js`
 - `frontend/src/services/serviceService.js`
 - `frontend/src/services/userService.js`
- Backend (PHP)
 - Controllers
 - `backend/app/controllers/AdminUserController.php`
 - `backend/app/controllers/AuthController.php`
 - `backend/app/controllers/BookingController.php`
 - `backend/app/controllers/DashboardController.php`
 - `backend/app/controllers/ServiceController.php`
 - `backend/app/controllers/StreamController.php`

- `backend/app/controllers/UserController.php`
- Core
 - `backend/app/core/Controller.php`
 - `backend/app/core/Database.php`
 - `backend/app/core/Request.php`
 - `backend/app/core/Response.php`
 - `backend/app/core/Router.php`
- Middlewares
 - `backend/app/middlewares/AdminMiddleware.php`
 - `backend/app/middlewares/AuthMiddleware.php`
 - `backend/app/middlewares/CorsMiddleware.php`
- Models
 - `backend/app/models/Booking.php`
 - `backend/app/models/Service.php`
 - `backend/app/models/User.php`
- Services
 - `backend/app/services/TimeSlotService.php`
- Bootstrap / Routes / Entry
 - `backend/bootstrap.php`
 - `backend/public/index.php`
 - `backend/routes/api.php`

Frontend

frontend/src/App.jsx

```
import React from 'react'

import { Routes, Route } from 'react-router-dom'

import Navbar from './components/Navbar'

import Home from './pages/Home'

import Login from './pages/Login'

import Dashboard from './pages/Dashboard'

import ServiceManagement from './pages/ServiceManagement'

import BookingManagement from './pages/BookingManagement'

import UserProfile from './pages/UserProfile'

import AdminLayout from './components/AdminLayout'

import SiteFooter from './components/SiteFooter'

import Bookings from './pages/Bookings'

import ProtectedRoute from './components/ProtectedRoute'

import UserDashboard from './pages/UserDashboard'

import AdminUsers from './pages/AdminUsers'

function App(){
  return (
    <div className="app-shell d-flex flex-column min-vh-100">
      <Navbar />
      <main className="flex-fill">
        <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/login" element={<Login />} />
          <Route path="/dashboard" element={
            <ProtectedRoute role="customer">
```

```
        <UserDashboard />
    </ProtectedRoute>
} />
<Route path="/bookings" element={
    <ProtectedRoute role="customer">
        <Bookings />
    </ProtectedRoute>
} />
<Route path="/profile" element={
    <ProtectedRoute>
        <UserProfile />
    </ProtectedRoute>
} />
<Route path="/admin/dashboard" element={
    <ProtectedRoute role="admin">
        <AdminLayout>
            <Dashboard />
        </AdminLayout>
    </ProtectedRoute>
} />
<Route path="/admin/services" element={
    <ProtectedRoute role="admin">
        <AdminLayout>
            <ServiceManagement />
        </AdminLayout>
    </ProtectedRoute>
} />
<Route path="/admin/bookings" element={
    <ProtectedRoute role="admin">
```



```
    <AdminLayout>
      <BookingManagement />
    </AdminLayout>
  </ProtectedRoute>
} />
<Route path="/admin/users" element={
  <ProtectedRoute role="admin">
    <AdminLayout>
      <AdminUsers />
    </AdminLayout>
  </ProtectedRoute>
} />
</Routes>
</main>
<SiteFooter />
</div>
)
}
```

```
export default App
```

Components

frontend/src/components/AdminLayout.jsx

// AdminLayout.jsx — Wraps admin pages with sidebar and main content; handles sidebar collapsed state.

import React from 'react'

import AdminSidebar from './AdminSidebar'

import useLocalStorage from '../hooks/useLocalStorage'

export default function AdminLayout({ children }) {

const [collapsed, setCollapsed] = useLocalStorage('admin_sidebar_collapsed', false)

return (

<div className={`admin-layout \${collapsed ? 'admin-collapsed' : ''}`>

<AdminSidebar collapsed={collapsed} onToggle={() => setCollapsed(v => !v)} />

<main className="admin-main">

<div className="admin-content container-fluid py-3 py-md-4">{children}</div>

</main>

</div>

)

}

frontend/src/components/AdminSidebar.jsx

// AdminSidebar.jsx — Left navigation for admin area with collapse toggle and tooltips.

```
import React from 'react'
```

```
import { NavLink, useNavigate } from 'react-router-dom'
```

```
import { useAuthContext } from '../contexts/AuthContext'
```

```
function IconDashboard(){
```

```
  return (<svg width="18" height="18" viewBox="0 0 24 24" fill="none"
    xmlns="http://www.w3.org/2000/svg"><rect x="3" y="3" width="8" height="8" rx="1"
    stroke="currentColor" strokeWidth="1.2"/><rect x="13" y="3" width="8" height="5" rx="1"
    stroke="currentColor" strokeWidth="1.2"/><rect x="13" y="10" width="8" height="11" rx="1"
    stroke="currentColor" strokeWidth="1.2"/></svg>)
```

```
}
```

```
function IconServices(){
```

```
  return (<svg width="18" height="18" viewBox="0 0 24 24" fill="none"
    xmlns="http://www.w3.org/2000/svg"><path d="M12 2l2 5h5l-4 3 1 6-5-3 5 3 1-6-4-3h5l2-5z"
    stroke="currentColor" strokeWidth="1" fill="none"/></svg>)
```

```
}
```

```
function IconBookings(){
```

```
  return (<svg width="18" height="18" viewBox="0 0 24 24" fill="none"
    xmlns="http://www.w3.org/2000/svg"><path d="M7 10h10M7 14h4M3 6h18M5 20h14a1 1 0 0 0 1-1V7a1 1 0 0 0-1-1H4a1 1 0 0 0-1 1v12a1 1 0 0 0 1 1z" stroke="currentColor" strokeWidth="1.2"
    strokeLinecap="round" strokeLinejoin="round"/></svg>)
```

```
}
```

```
function IconUsers(){
```

```
  return (<svg width="18" height="18" viewBox="0 0 24 24" fill="none"
    xmlns="http://www.w3.org/2000/svg"><path d="M16 11a4 4 0 1 0-8 0" stroke="currentColor"
    strokeWidth="1.2"/><path d="M3 20a7 7 0 0 1 7-7h4a7 7 0 0 1 7 7" stroke="currentColor"
    strokeWidth="1.2" strokeLinecap="round"/></svg>)
```

```
}
```

```
export default function AdminSidebar({ collapsed=false, onToggle }){
```

```
  const navigate = useNavigate()
```

```

const { logout } = useAuthContext()

return (
  <aside className="admin-sidebar" aria-label="Admin sidebar">

    <div className="admin-sidebar-inner d-flex flex-column" style={{height: '100vh', padding: '1rem 0'}}>

      <div className="px-3 mb-4 d-flex align-items-center justify-content-between">

        <div className="d-flex align-items-center gap-2">

          <div className="brand-title text-white fw-bold d-none d-md-block">GlowUpNow</div>

        </div>

        <button type="button" className="btn btn-sm btn-outline-light sidebar-toggle"
onClick={onToggle} aria-label="Toggle sidebar">

          {collapsed ? (

            <svg width="16" height="16" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg"><path d="M9 6l6 6 6 6" stroke="currentColor"
strokeWidth="1.6" strokeLinecap="round" strokeLinejoin="round"/></svg>

            ) : (

              <svg width="16" height="16" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg"><path d="M15 6l-6 6 6 6" stroke="currentColor"
strokeWidth="1.6" strokeLinecap="round" strokeLinejoin="round"/></svg>

            )}

          </button>

        </div>

      <nav className="nav flex-column px-2" style={{gap:8}}>

        <NavLink to="/admin/dashboard" title="Dashboard" className={{isActive}}=>`admin-link d-flex
align-items-center gap-2 px-3 py-2 ${isActive ? 'active' : ''}`>

          <IconDashboard />

          <span className="link-label">Dashboard</span>

        </NavLink>

        <NavLink to="/admin/services" title="Services" className={{isActive}}=>`admin-link d-flex
align-items-center gap-2 px-3 py-2 ${isActive ? 'active' : ''}`>

          <IconServices />

```

```

        <span className="link-label">Services</span>

    </NavLink>

    <NavLink to="/admin/bookings" title="Bookings" className={{isActive}}=>`admin-link d-flex
align-items-center gap-2 px-3 py-2 ${isActive ? 'active' : ''}`>

        <IconBookings />

        <span className="link-label">Bookings</span>

    </NavLink>

    <NavLink to="/admin/users" title="Users" className={{isActive}}=>`admin-link d-flex align-
items-center gap-2 px-3 py-2 ${isActive ? 'active' : ''}`>

        <IconUsers />

        <span className="link-label">Users</span>

    </NavLink>

    <button className="admin-link d-flex align-items-center gap-2 px-3 py-2 btn btn-link text-start"
onClick={async()=>{ try{ await logout(); navigate('/login', { replace: true }) }catch(e){ navigate('/login', {
replace: true }) } }}>

        <svg width="18" height="18" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg"><path d="M10 17l5-5-5-5" stroke="currentColor"
strokeWidth="1.2" strokeLinecap="round" strokeLinejoin="round"/><path d="M15 12H3"
stroke="currentColor" strokeWidth="1.2" strokeLinecap="round" strokeLinejoin="round"/><path
d="M21 4v16a2 2 0 0 1-2 2h-6" stroke="currentColor" strokeWidth="1.2" strokeLinecap="round"
strokeLinejoin="round"/></svg>

        <span className="link-label">Logout</span>

    </button>

</nav>

<div className="mt-auto px-3 pb-3 text-white small">Signed in as Admin</div>

</div>

</aside>

)
}

```

frontend/src/components/BookingForm.jsx

import React, { useState } from 'react'

import { createBooking } from '../services/bookingService'

export default function BookingForm({ service, onSuccess }) {

const [date, setDate] = useState('')

const [start, setStart] = useState('09:00')

const [notes, setNotes] = useState('')

const [error, setError] = useState('')

const [loading, setLoading] = useState(false)

const WORK_START = '09:00'

const WORK_END = '18:00'

const duration = Number(service?.duration_minutes || 60)

const toSeconds = (hhmm) => {

const [h,m] = (hhmm || '00:00').split(':').map(Number)

return (h*3600) + (m*60)

}

const secondsToHHMM = (secs) => {

const h = Math.floor(secs/3600).toString().padStart(2,'0')

const m = Math.floor((secs%3600)/60).toString().padStart(2,'0')

return `\${h}:\${m}`

}

const hhmmToHhmmss = (hhmm) => `\${hhmm.length===5?hhmm:hhmm.slice(0,5)}:00`

const maxStartHHMM = () => {

const endSecs = toSeconds(WORK_END)

const maxStartSecs = endSecs - (duration*60)

return secondsToHHMM(Math.max(0, maxStartSecs))

```
}
```

```
const todayStr = () => new Date().toISOString().slice(0,10)
```

```
const submit = async (e)=>{
```

```
  e.preventDefault()
```

```
  setLoading(true)
```

```
  setError("")
```

```
  try{
```

```
    // Client-side validation for date/time
```

```
    if (!date || date < todayStr()) { throw new Error('Please pick today or a future date.') }
```

```
    const s = start
```

```
    const eSecs = toSeconds(s) + (duration*60)
```

```
    if (s < WORK_START || s > maxStartHHMM() || eSecs > toSeconds(WORK_END)) {
```

```
      throw new Error(`Please pick a time within working hours (${WORK_START}–${WORK_END}) that  
fits the service duration.`)
```

```
    }
```

```
    await createBooking({
```

```
      service_id: service.id,
```

```
      date,
```

```
      start_time: hhmmToHhmmss(s),
```

```
      notes
```

```
    })
```

```
    onSuccess && onSuccess()
```

```
  }catch(err){
```

```
    setError(err?.data?.error || err?.message || 'Booking failed')
```

```
  }finally{ setLoading(false) }
```

```
}
```

```

return (
  <form onSubmit={submit} className="card p-3 mt-3">
    <h5 className="mb-3">Book: {service.name}</h5>
    {error && <div className="alert alert-danger">{error}</div>}
    <div className="alert alert-info py-2">Accepting bookings: {WORK_START}–{WORK_END}</div>
    <div className="row g-3">
      <div className="col-md-4">
        <label className="form-label" htmlFor="bf_date">Date</label>
        <input id="bf_date" name="date" type="date" className="form-control" min={todayStr()}
value={date} onChange={e=>setDate(e.target.value)} required />
      </div>
      <div className="col-md-4">
        <label className="form-label" htmlFor="bf_start_time">Start Time</label>
        <input id="bf_start_time" name="start_time" type="time" className="form-control" step="60"
min={WORK_START} max={maxStartHHMM()} value={start}
onChange={e=>setStart(e.target.value.slice(0,5))} required />
      </div>
      <div className="col-12">
        <label className="form-label" htmlFor="bf_notes">Notes</label>
        <textarea id="bf_notes" name="notes" className="form-control" rows="2" value={notes}
onChange={e=>setNotes(e.target.value)} />
      </div>
    </div>
    <div className="mt-3 text-end">
      <button disabled={loading} className="btn btn-primary"
type="submit">{loading?'Booking...':'Confirm Booking'}</button>
    </div>
  </form>
)
}

```


frontend/src/components/BookingList.jsx

```
import React, { useEffect, useMemo, useState } from 'react'
```

```
import { listBookings } from '../services/bookingService'
```

```
export default function BookingList({ onSelect, onInteract }) {
```

```
  const [items, setItems] = useState([])
```

```
  const [loading, setLoading] = useState(false)
```

```
  const [error, setError] = useState("")
```

```
  // Filters
```

```
  const [fId, setFId] = useState("")
```

```
  const [fDate, setFDate] = useState("")
```

```
  const [fTime, setFTime] = useState("")
```

```
  const [fCustomer, setFCustomer] = useState("")
```

```
  const [fService, setFService] = useState("")
```

```
  const [fStatus, setFStatus] = useState("")
```

```
  // Pagination
```

```
  const [page, setPage] = useState(1)
```

```
  const [perPage, setPerPage] = useState(10)
```

```
  const load = async () => {
```

```
    setLoading(true); setError("")
```

```
    try {
```

```
      const res = await listBookings({ status: fStatus })
```

```
      const rows = res.data || []
```

```
      setItems(rows)
```

```
    } catch (e) { setError('Failed to load bookings') }
```

```
    finally { setLoading(false) }
```

```
  }
```

```
  useEffect(() => { load() }, [fStatus])
```

```

const filtered = useMemo(()=>{
  return items.filter(b => {
    if (fld !== '' && String(b.id) !== String(fld)) return false
    if (fDate && String(b.date) !== String(fDate)) return false
    if (fTime && !(String(b.start_time | '').startsWith(fTime) ||
String(b.end_time | '').startsWith(fTime))) return false
    if (fCustomer && !String(b.customer_name | '').toLowerCase().includes(fCustomer.toLowerCase()))
return false
    if (fService && !String(b.service_name | '').toLowerCase().includes(fService.toLowerCase())) return
false
    return true
  })
}, [items, fld, fDate, fTime, fCustomer, fService])

```

```

const totalPages = Math.max(1, Math.ceil(filtered.length / perPage))

```

```

const safePage = Math.min(page, totalPages)

```

```

const startIdx = (safePage - 1) * perPage

```

```

const pageltems = filtered.slice(startIdx, startIdx + perPage)

```

```

const fireInteract = ()=>{ if (onInteract) onInteract() }

```

```

const clearFilters = ()=>{

```

```

  setFld(''); setFDate(''); setFTime(''); setFCustomer(''); setFService(''); setFStatus(''); setPerPage(10);
setPage(1); fireInteract(); load()
}

```

```

const onEnter = (e)=>{ if(e.key==='Enter'){ e.preventDefault(); setPage(1); fireInteract() } }

```

```

return (

```

```

  <div>

```

```

    <div className="card p-3 mb-3" style={{ position:'sticky', top:8, zIndex:10 }}>

```

```

<div className="row g-3 align-items-end">

  { /* Row 1: #, Date, Time, Customer, Service (2+2+2+3+3 = 12 on lg) */ }

  <div className="col-6 col-md-3 col-lg-2">

    <label className="form-label">#</label>

    <input type="number" className="form-control" value={fId} onChange={e=>{
setFId(e.target.value); setPage(1); fireInteract() }} onKeyDown={onEnter} />

  </div>

  <div className="col-6 col-md-3 col-lg-2">

    <label className="form-label">Date</label>

    <input type="date" className="form-control" value={fDate} onChange={e=>{
setFDate(e.target.value); setPage(1); fireInteract() }} onKeyDown={onEnter} />

  </div>

  <div className="col-6 col-md-3 col-lg-2">

    <label className="form-label">Time</label>

    <input type="time" className="form-control" value={fTime} onChange={e=>{
setFTime(e.target.value); setPage(1); fireInteract() }} onKeyDown={onEnter} />

  </div>

  <div className="col-12 col-md-6 col-lg-3">

    <label className="form-label">Customer</label>

    <input className="form-control" value={fCustomer} onChange={e=>{
setFCustomer(e.target.value); setPage(1); fireInteract() }} onKeyDown={onEnter} placeholder="Name"
/>

  </div>

  <div className="col-12 col-md-6 col-lg-3">

    <label className="form-label">Service</label>

    <input className="form-control" value={fService} onChange={e=>{ setFService(e.target.value);
setPage(1); fireInteract() }} onKeyDown={onEnter} placeholder="Service" />

  </div>

  { /* Row 2: Status, Per page, Clear (2+2+8) */ }

  <div className="col-6 col-md-3 col-lg-2">

    <label className="form-label">Status</label>

```

```
    <select className="form-select" value={fStatus} onChange={e=>{ setFStatus(e.target.value);
setPage(1); fireInteract() }} onKeyDown={onEnter}>
```

```
    <option value="">All</option>
```

```
    <option value="pending">Pending</option>
```

```
    <option value="confirmed">Confirmed</option>
```

```
    <option value="canceled">Canceled</option>
```

```
    <option value="completed">Completed</option>
```

```
</select>
```

```
</div>
```

```
<div className="col-6 col-md-3 col-lg-2">
```

```
    <label className="form-label">Per page</label>
```

```
    <select className="form-select" value={perPage} onChange={e=>{
setPerPage(Number(e.target.value)); setPage(1); fireInteract() }} onKeyDown={onEnter}>
```

```
    <option value={5}>5</option>
```

```
    <option value={10}>10</option>
```

```
    <option value={20}>20</option>
```

```
    <option value={50}>50</option>
```

```
</select>
```

```
</div>
```

```
<div className="col-12 col-lg-8 d-flex justify-content-lg-end align-items-end">
```

```
    <button className="btn btn-outline-primary" type="button" onClick={clearFilters}>Clear
search/filters</button>
```

```
</div>
```

```
</div>
```

```
</div>
```

```
{loading && <div className="text-muted small mb-2">Loading...</div>}
```

```
{filtered.length === 0 && !loading ? (
```

```
    <div className="text-muted">No results matched your filters.</div>
```

```
) : (
```

```

<div className="table-responsive">
  <table className="table table-striped table-hover">
    <thead>
      <tr>
        <th>#</th>
        <th>Date</th>
        <th>Time</th>
        <th>Customer</th>
        <th>Service</th>
        <th>Status</th>
        <th className="text-end">Actions</th>
      </tr>
    </thead>
    <tbody>
      {pageItems.map(b=> (
        <tr key={b.id}>
          <td>{b.id}</td>
          <td>{b.date}</td>
          <td>{b.start_time} - {b.end_time}</td>
          <td>{b.customer_name}</td>
          <td>{b.service_name}</td>
          <td><span className={`badge text-bg-
${b.status=== 'confirmed'? 'success': b.status=== 'pending'? 'warning': b.status=== 'canceled'? 'danger': 'secondary'}>{b.status}</span></td>
          <td className="text-end">
            <button className="btn btn-sm btn-outline-primary" onClick={()=> onSelect &&
onSelect(b)}>Edit</button>
          </td>
        </tr>
      )
    )}

```

```

        </tbody>
    </table>
</div>
)}

<div className="d-flex justify-content-between align-items-center mt-2">
    <div className="small text-muted">Showing {filtered.length === 0 ? 0 : (startIndex+1)}–
    {Math.min(startIndex+perPage, filtered.length)} of {filtered.length}</div>

    <div className="btn-group" role="group" aria-label="Pagination">

        <button type="button" className="btn btn-outline-secondary btn-sm" disabled={safePage<=1}
        onClick={()=>{ fireInteract(); setPage(p=>Math.max(1, p-1)) }}>Prev</button>

        <span className="btn btn-outline-secondary btn-sm disabled">Page {safePage} /
        {totalPages}</span>

        <button type="button" className="btn btn-outline-secondary btn-sm"
        disabled={safePage>=totalPages} onClick={()=>{ fireInteract(); setPage(p=>Math.min(totalPages, p+1))
        }}>Next</button>

    </div>

</div>
)
}

```

frontend/src/components/Navbar.jsx

// Navbar.jsx — Top navigation; hides brand in admin area and shows responsive burger on non-admin small screens.

```
import React from 'react'
```

```
import { Link, useLocation } from 'react-router-dom'
```

```
import { useAuthContext } from '../contexts/AuthContext'
```

```
export default function Navbar(){
```

```
  const { user, logout } = useAuthContext()
```

```
  const { pathname } = useLocation()
```

```
  const isHome = pathname === '/'
```

```
  const inAdminArea = pathname.startsWith('/admin')
```

```
  return (
```

```
    <nav className={`navbar navbar-expand-lg ${isHome ? 'navbar-dark navbar-hero position-absolute top-0 start-0 w-100' : 'navbar-dark'}`>
```

```
      <div className="container">
```

```
        {!(user && user.role === 'admin' && inAdminArea) && (
```

```
          <Link className="navbar-brand" to="/">GlowUpNow</Link>
```

```
        )}
```

```
        {inAdminArea && (
```

```
          <button className="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#nav">
```

```
            <span className="navbar-toggler-icon"></span>
```

```
          </button>
```

```
        )}
```

```
      <div className="collapse navbar-collapse" id="nav">
```

```
        <ul className={`navbar-nav ms-auto align-items-lg-center gap-2 ${isHome ? 'navbar-hero-links' : ''}`>
```

```
          {!user && pathname !== '/login' && (
```

```

        <li className="nav-item"><Link className={`btn btn-login btn-sm`}
to="/login">Login</Link></li>

    )}

    {user && (

        <>

            {user.role !== 'admin' && (

                <>

                    <li className="nav-item"><Link className="nav-link"
to="/dashboard">Dashboard</Link></li>

                    <li className="nav-item"><Link className="nav-link"
to="/bookings">Bookings</Link></li>

                    <li className="nav-item"><Link className="nav-link" to="/profile">Profile</Link></li>

                </>

            )}

            {user.role === 'admin' && inAdminArea && (

                <></>

            )}

            {!(user.role === 'admin' && inAdminArea) && (

                <li className="nav-item"><span className="nav-link disabled">Hi,
{user.name}</span></li>

            )}

            {!(user.role === 'admin' && inAdminArea) && (

                <li className="nav-item"><button className="btn btn-sm btn-light"
onClick={logout}>Logout</button></li>

            )}

        </>

    )}

</ul>

</div>

</div>

```


</nav>

)

}

frontend/src/components/ProtectedRoute.jsx

```
import React from 'react'
```

```
import { Navigate } from 'react-router-dom'
```

```
import { useAuthContext } from '../contexts/AuthContext'
```

```
export default function ProtectedRoute({ children, role }) {
```

```
  const { user, loading } = useAuthContext()
```

```
  if (loading) return <div className="container py-4"><div className="alert alert-info">Loading...</div></div>
```

```
  if (!user) return <Navigate to="/" replace />
```

```
  if (role && user.role !== role) {
```

```
    const target = user.role === 'admin' ? '/admin/dashboard' : '/dashboard'
```

```
    return <Navigate to={target} replace />
```

```
  }
```

```
  return children
```

```
}
```

frontend/src/components/ServiceCard.jsx

// ServiceCard.jsx — Displays a service with image skeleton, details, and optional Book action.

```
import React, { useState } from 'react'
```

```
import { API } from '../services/http'
```

```
export default function ServiceCard({ service, onSelect }) {
```

```
  const imgSrc = service.image_path ? (service.image_path.startsWith('http') ? service.image_path :  
  `${API}${service.image_path}`) : null
```

```
  const [loaded, setLoaded] = useState(false)
```

```
  return (
```

```
    <div className="card h-100">
```

```
      <div style={{position:'relative', height:180, overflow:'hidden'}}>
```

```
        {!loaded && <div style={{position:'absolute', inset:0, background:'linear-gradient(90deg, var(--gu-  
mist), #f3eef6, var(--gu-mist))', animation:'shimmer 1.2s infinite'}} />}
```

```
        {imgSrc ? (
```

```
          <img src={imgSrc} alt={service.name} onLoad={()=>setLoaded(true)} style={{objectFit:'cover',  
width:'100%', height:'100%', display: loaded? 'block':'none'}} />
```

```
        ) : (
```

```
          <div style={{height:'100%', background:'var(--gu-mist)}} />
```

```
        )}
```

```
      </div>
```

```
    <div className="card-body d-flex flex-column">
```

```
      <h5 className="card-title mb-1">{service.name}</h5>
```

```
      <p className="card-text small flex-grow-1">{service.description}</p>
```

```
      <div className="d-flex justify-content-between align-items-center mt-2">
```

```
        <span className="service-price fw-bold">P{Number(service.price).toFixed(2)}</span>
```

```
        <span className="badge text-bg-secondary">{service.duration_minutes} mins</span>
```

```
      </div>
```

```
    </div>
```

```
    {onSelect && (
```

```
<div className="card-footer bg-transparent border-0 text-end">
  <button className="btn btn-primary btn-sm" onClick={()=>onSelect(service)}>Book</button>
</div>
  )
</div>
)
```

frontend/src/components/SiteFooter.jsx

import React from 'react'

export default function SiteFooter(){

 return (

 <footer className="site-footer p-4 mt-4">

 <div className="container">

 <div className="d-flex flex-wrap justify-content-between align-items-center gap-3">

 <div>

 <div className="fw-bold">GlowUpNow</div>

 <div className="small">Home salon & spa services, on demand.</div>

 </div>

 <div className="small">© {new Date().getFullYear()} GlowUpNow. All rights reserved.</div>

 </div>

 </div>

 </footer>

)

}

Contexts

frontend/src/context/AuthContext.jsx

```
import React, { createContext, useContext, useEffect, useState } from 'react'
```

```
import { me as apiMe, login as apiLogin, logout as apiLogout, register as apiRegister } from  
'../services/authService'
```

```
const AuthContext = createContext(null)
```

```
export function AuthProvider({ children }) {
```

```
  const [user, setUser] = useState(null)
```

```
  const [loading, setLoading] = useState(true)
```

```
  useEffect(() => {
```

```
    (async () => {
```

```
      try {
```

```
        const res = await apiMe()
```

```
        setUser(res.user || null)
```

```
      } catch { /* ignore */ }
```

```
      setLoading(false)
```

```
    })()
```

```
  }, [])
```

```
  const login = async (email, password) => {
```

```
    const res = await apiLogin(email, password)
```

```
    setUser(res.user)
```

```
    return res
```

```
  }
```

```
  const logout = async () => {
```

```
    await apiLogout()
    setUser(null)
  }
```

```
const register = async (name, email, password) => {
  const res = await apiRegister(name, email, password)
  setUser(res.user)
  return res
}
```

```
return (
  <AuthContext.Provider value={{ user, loading, login, logout, register }}>
    {children}
  </AuthContext.Provider>
)
}
```

```
export function useAuthContext(){
  return useContext(AuthContext)
}
```

Hooks

frontend/src/hooks/index.js

// hooks/index — Central exports for custom hooks used across the app.

export { default as useToggle } from './useToggle'

export { default as useForm } from './useForm'

export { default as useFetch } from './useFetch'

export { default as useInterval } from './useInterval'

export { default as useModal } from './useModal'

export { default as useEventSource } from './useEventSource'

export { default as useAuthHook } from './useAuthHook'

frontend/src/hooks/useAuthHook.js

// useAuthHook — Derived auth state (isLoggedIn, isAdmin) from AuthContext.

import { useMemo } from 'react'

import { useAuthContext } from '../contexts/AuthContext'

export default function useAuthHook(){

const { user, loading, login, logout, register } = useAuthContext()

const isLoggedIn = !!user

const isAdmin = !!user && user.role === 'admin'

return useMemo(()=>({ user, loading, isLoggedIn, isAdmin, login, logout, register }), [user, loading, login, logout, register])

}

frontend/src/hooks/useClickOutside.js

```
import { useEffect } from 'react'
```

```
export default function useClickOutside(ref, handler){
```

```
  useEffect(()=>{
```

```
    function onMouseDown(e){
```

```
      const el = ref && 'current' in ref ? ref.current : null
```

```
      if (!el) return
```

```
      if (!el.contains(e.target)) { handler && handler(e) }
```

```
    }
```

```
    document.addEventListener('mousedown', onMouseDown)
```

```
    return ()=> document.removeEventListener('mousedown', onMouseDown)
```

```
  }, [ref, handler])
```

```
}
```

frontend/src/hooks/useDebounce.js

```
import { useEffect, useState } from 'react'
```

```
export default function useDebounce(value, delay = 400){  
  const [debounced, setDebounced] = useState(value)  
  useEffect(()=>{  
    const id = setTimeout(()=> setDebounced(value), delay)  
    return ()=> clearTimeout(id)  
  }, [value, delay])  
  return debounced  
}
```

frontend/src/hooks/useEventSource.js

// useEventSource — SSE wrapper with connect/close and event handlers.

import { useCallback, useEffect, useRef, useState } from 'react'

export default function useEventSource(url, opts={}){

const { withCredentials=true, events } = opts

const esRef = useRef(null)

const [connected, setConnected] = useState(false)

const [error, setError] = useState(null)

const open = useCallback((u=url, handlers=events)=>{

if (!u) return

if (esRef.current) { try{ esRef.current.close() }catch{ } }

const es = new EventSource(u, { withCredentials })

esRef.current = es

setConnected(true)

setError(null)

if (handlers) {

Object.entries(handlers).forEach(([name, fn])=>{ if(typeof fn==='function')
es.addListener(name, fn) })

}

es.onerror = (e)=>{ setError(e) }

}, [url, events, withCredentials])

const close = useCallback(()=>{

if (esRef.current) { try{ esRef.current.close() }catch{ }; esRef.current=null; setConnected(false) }
 }, [])

useEffect(()=>()=>{ if(esRef.current) { try{ esRef.current.close() }catch{ } } }, [])

```
return { connected, error, open, close, ref: esRef }  
}
```

frontend/src/hooks/useFetch.js

// useFetch — Generic async lifecycle (data, loading, error, reload).

import { useCallback, useEffect, useState } from 'react'

export default function useFetch(fn, deps = [], opts = {}){

const { immediate = true } = opts

const [data, setData] = useState(null)

const [error, setError] = useState(null)

const [loading, setLoading] = useState(false)

const run = useCallback(async (...args)=>{

setLoading(true); setError(null)

try{

const res = await fn(...args)

setData(res)

return res

}catch(e){ setError(e); throw e }

finally{ setLoading(false) }

}, [fn])

useEffect(()=>{ if(immediate) { run() } }, [immediate, run, ...deps])

return { data, error, loading, reload: run, setData }

}

frontend/src/hooks/useForm.js

// useForm — Controlled form state with validation and submit.

import { useCallback, useState } from 'react'

export default function useForm({ initialValues = {}, validate, onSubmit } = {}) {

const [values, setValues] = useState(initialValues)

const [errors, setErrors] = useState({})

const [touched, setTouched] = useState({})

const handleChange = useCallback((e) => {

const { name, type, checked, value } = e.target

const v = type === 'checkbox' ? !!checked : value

setValues(s => ({ ...s, [name]: v }))

}, [])

const handleBlur = useCallback((e) => {

const { name } = e.target

setTouched(s => ({ ...s, [name]: true }))

if (validate) setErrors(validate(values) || {})

}, [validate, values])

const handleSubmit = useCallback(async (e) => {

if (e && e.preventDefault) e.preventDefault()

const errs = validate ? (validate(values) || {}) : {}

setErrors(errs)

if (Object.keys(errs).length) return

if (onSubmit) return onSubmit(values)

}, [validate, values, onSubmit])

```
const reset = useCallback((next = initialValues) => {  
  setValues(next)  
  setErrors({})  
  setTouched({})  
}, [initialValues])  
  
return { values, setValues, errors, touched, handleChange, handleBlur, handleSubmit, reset }  
}
```


frontend/src/hooks/useInterval.js

// useInterval — Runs a callback every given delay with cleanup.

```
import { useEffect, useRef } from 'react'
```

```
export default function useInterval(callback, delay){  
  const saved = useRef(callback)  
  useEffect(()=>{ saved.current = callback }, [callback])  
  useEffect(()=>{  
    if (delay === null || delay === false) return  
    const id = setInterval(()=> saved.current(), delay)  
    return ()=> clearInterval(id)  
  }, [delay])  
}
```

frontend/src/hooks/useLocalStorage.js

```
import { useEffect, useState } from 'react'
```

```
export default function useLocalStorage(key, initialValue){
```

```
  const read = () => {
```

```
    if (typeof window === 'undefined') return initialValue
```

```
    try{
```

```
      const item = window.localStorage.getItem(key)
```

```
      return item !== null ? JSON.parse(item) : initialValue
```

```
    }catch{ return initialValue }
```

```
  }
```

```
  const [value, setValue] = useState(read)
```

```
  useEffect(()=>{
```

```
    try{ window.localStorage.setItem(key, JSON.stringify(value)) }catch{}
```

```
  }, [key, value])
```

```
  return [value, setValue]
```

```
}
```

frontend/src/hooks/useModal.js

// useModal — Modal open/close with Escape handling and focus management.

```
import { useCallback, useEffect, useRef, useState } from 'react'
```

```
export default function useModal(initial=false){
  const [open, setOpen] = useState(!initial)
  const ref = useRef(null)
  const onKeyDown = useCallback((e)=>{ if(e.key==='Escape') setOpen(false) },[])
  const openModal = useCallback(()=> setOpen(true),[])
  const closeModal = useCallback(()=> setOpen(false),[])
  const toggleModal = useCallback(()=> setOpen(v=>!v),[])
  useEffect(()=>{
    if (!open) return
    const el = ref.current
    const prev = document.activeElement
    if (el && el.focus) { try{ el.focus() }catch{} }
    return ()=>{ if(prev && prev.focus) { try{ prev.focus() }catch{} } }
  }, [open])
  return { open, openModal, closeModal, toggleModal, onKeyDown, ref, setOpen }
}
```

frontend/src/hooks/useToggle.js

// useToggle — Boolean toggle hook.

```
import { useCallback, useState } from 'react'
```

```
export default function useToggle(initial = false) {  
  const [value, setValue] = useState(!initial)  
  const toggle = useCallback((next) => {  
    if (typeof next === 'boolean') { setValue(next); return }  
    setValue(v => !v)  
  }, [])  
  return [value, toggle, setValue]  
}
```

Pages

frontend/src/pages/AdminUsers.jsx

// AdminUsers.jsx — Admin: read-only list of users with search, pagination, and dense view toggle.

```
import React, { useMemo, useState } from 'react'
```

```
import { listUsers } from '../services/userService'
```

```
import { useFetch, useToggle } from '../hooks'
```

```
export default function AdminUsers(){
```

```
  const [page,setPage] = useState(1)
```

```
  const [perPage,setPerPage] = useState(10)
```

```
  const [q,setQ] = useState("")
```

```
  const [dense, toggleDense] = useToggle(false)
```

```
  const { data, loading, error } = useFetch(
```

```
    () => listUsers({ q, page, perPage }),
```

```
    [q, page, perPage],
```

```
    { immediate: true } 
```

```
)
```

```
  const rows = data?.rows || []
```

```
  const total = Number(data?.total || 0)
```

```
  const startIdx = (page-1)*perPage
```

```
  const totalPages = Math.max(1, Math.ceil(total / perPage))
```

```
  const safePage = Math.min(page, totalPages)
```

```
  return (
```

```
    <div className="container py-4">
```

```
      <h2 className="mb-3">User Management</h2>
```

```

<div className="card p-3 mb-3">

  <div className="d-flex flex-wrap gap-2 align-items-end">

    <div className="input-group" style={{maxWidth:360}}>

      <span className="input-group-text">Search</span>

      <input className="form-control" placeholder="name, email, role" value={q} onChange={e=>{
setPage(1); setQ(e.target.value) }} onKeyDown={e=>{ if(e.key==='Enter'){ e.preventDefault();
setPage(1) } }} />

    </div>

    <div>

      <label className="form-label">Per page</label>

      <select className="form-select" value={perPage} onChange={e=>{ setPage(1);
setPerPage(Number(e.target.value)) }}>

        <option value={10}>10</option>

        <option value={20}>20</option>

        <option value={50}>50</option>

        <option value={100}>100</option>

      </select>

    </div>

    <div className="ms-auto d-flex align-items-center gap-2">

      <label className="form-label mb-0">Dense</label>

      <button className={`btn btn-sm ${dense?'btn-primary':'btn-outline-primary'}`}
onClick={()=>toggleDense()} type="button">{dense?'On':'Off'}</button>

    </div>

  </div>

  <div>

    {loading && <div className="text-muted small mb-2">Loading...</div>}

    {error && <div className="alert alert-danger">{error}</div>}

  </div>

</div>

```

```
<div className="table-responsive">

  <table className={`table ${dense? 'table-sm':''} table-striped table-hover m-0`}>

    <thead>

      <tr>

        <th>#</th>

        <th>Name</th>

        <th>Email</th>

        <th>Role</th>

        <th>Phone</th>

        <th>Joined</th>

      </tr>

    </thead>

    <tbody>

      {rows.length===0 ? (

        <tr><td colspan={6} className="text-muted">No users found</td></tr>

      ) : rows.map(u => (

        <tr key={u.id}>

          <td>{u.id}</td>

          <td>{u.name}</td>

          <td>{u.email}</td>

          <td>{u.role}</td>

          <td>{u.phone || '-'}</td>

          <td>{u.created_at}</td>

        </tr>

      ))}

    </tbody>

  </table>

</div>

<div className="card-body d-flex justify-content-between align-items-center">
```

```

    <div className="small text-muted">Showing {rows.length===0?0:(startIdx+1)}–
    {Math.min(startIdx+rows.length, total)} of {total}</div>

    <div className="btn-group" role="group">

        <button className="btn btn-outline-secondary btn-sm" disabled={safePage<=1}
        onClick={()=>setPage(p=>Math.max(1,p-1))}>Prev</button>

        <span className="btn btn-outline-secondary btn-sm disabled">Page {safePage} /
        {totalPages}</span>

        <button className="btn btn-outline-secondary btn-sm" disabled={safePage>=totalPages}
        onClick={()=>setPage(p=>Math.min(totalPages,p+1))}>Next</button>

    </div>

</div>

</div>

</div>

)
}

```


frontend/src/pages/BookingManagement.jsx

import React, { useEffect, useRef, useState } from 'react'

import BookingList from '../components/BookingList'

import { updateBookingStatus } from '../services/bookingService'

export default function BookingManagement(){

const [statusId, setStatusId] = useState("")

const [newStatus, setNewStatus] = useState('confirmed')

const [msg, setMsg] = useState("")

const [err, setErr] = useState("")

const formRef = useRef(null)

const [idEditable, setIdEditable] = useState(false)

const [currentStatus, setCurrentStatus] = useState("")

const [toast, setToast] = useState({ type:"", msg:"" })

const update = async ()=>{

setMsg(""); setErr("")

if (!statusId) { setErr('Please select a booking to update'); return }

if (newStatus === currentStatus) {

setMsg(`No changes. Status is already "\${newStatus}"`.)

// clear form as requested when update is clicked without changes

setStatusId(""); setNewStatus('confirmed'); setIdEditable(false)

return

}

try{

await updateBookingStatus({ id: Number(statusId), status: newStatus })

setMsg('Status updated')

setCurrentStatus(newStatus)

// auto-clear form details after success

```
    setStatusId(""); setNewStatus('confirmed'); setIdEditable(false)
  } catch(e) { setErr(e.data?.error || 'Update failed') }
}
```

```
const scrollToForm = ()=>{
  if (formRef.current) { formRef.current.scrollToView({ behavior: 'smooth', block: 'start' }) }
}
```

```
const handleSelectForEdit = (booking)=>{
  const id = booking?.id
  const st = booking?.status || 'confirmed'
  setStatusId(String(id || ''))
  setNewStatus(st)
  setCurrentStatus(st)
  setIdEditable(true)
  setMsg(""); setErr("")
  scrollToForm()
}
```

```
const handleInteract = ()=>{ setMsg(""); setErr("") }
```

```
useEffect(()=>{ if (msg) setToast({ type:'success', msg }) }, [msg])
```

```
useEffect(()=>{ if (err) setToast({ type:'error', msg: err }) }, [err])
```

```
return (
  <div className="container py-4">
    <h2 className="mb-3">Booking Management</h2>

    <div className="card p-3 mb-3" ref={formRef}>
      <div className="row g-2">
```

```

<div className="col-md-4">

  <label className="form-label" htmlFor="bm_booking_id">Booking ID</label>

  <input id="bm_booking_id" name="booking_id" className="form-control" value={statusId}
disabled={!idEditable} onChange={e=>setStatusId(e.target.value)} onKeyDown={(e)=>{
if(e.key==='Enter'){ e.preventDefault(); update() } } } />

</div>

<div className="col-md-4">

  <label className="form-label" htmlFor="bm_new_status">New Status</label>

  <select id="bm_new_status" name="new_status" className="form-select" value={newStatus}
onChange={e=>setNewStatus(e.target.value)}>

    <option value="pending">Pending</option>

    <option value="confirmed">Confirmed</option>

    <option value="canceled">Canceled</option>

    <option value="completed">Completed</option>

  </select>

</div>

<div className="col-md-4 d-flex align-items-end gap-2">

  <button className="btn btn-primary" onClick={update} disabled={!statusId}>Update
Status</button>

  <button className="btn btn-outline-secondary" onClick={()=>{ setStatusId("");
setNewStatus('confirmed'); setIdEditable(false); setMsg(""); setErr("") }}>Cancel</button>

</div>

</div>

</div>

```

```

<BookingList onSelect={handleSelectForEdit} onInteract={() => { handleInteract(); setToast({type:',
msg:''}) }} />

```

```

{/* Toasts */}

```

```

<div style={{position:'fixed', right:16, bottom:16, zIndex:9999}}>

```

```

{toast.msg && (

```

```
<div className={`toast align-items-center text-white ${toast.type==='error'?'bg-danger':'bg-
success'} show`} role="alert">

  <div className="d-flex">

    <div className="toast-body">{toast.msg}</div>

    <button type="button" className="btn-close btn-close-white me-2 m-auto"
onClick={()=>setToast({type:"",msg:"})}></button>

  </div>

</div>

  )
}
```

frontend/src/pages/Bookings.jsx

import React, { useEffect, useState, useMemo, useRef } from 'react'

import { useLocation } from 'react-router-dom'

import { useAuthContext } from '../contexts/AuthContext'

import { listBookings, cancelBooking, rescheduleBooking } from '../services/bookingService'

import BookingForm from '../components/BookingForm'

import { API } from '../services/http'

import useLocalStorage from '../hooks/useLocalStorage'

import useClickOutside from '../hooks/useClickOutside'

export default function Bookings(){

const { user } = useAuthContext()

const location = useLocation()

const [selected, setSelected] = useState(null)

const [successMsg, setSuccessMsg] = useState("")

const [error, setError] = useState("")

const [bookings, setBookings] = useState([])

const [bookingsLoading, setBookingsLoading] = useState(false)

const [resModalOpen, setResModalOpen] = useState(false)

const [resId, setResId] = useState(null)

const [resDate, setResDate] = useState("")

const [resStart, setResStart] = useState("")

const [resDuration, setResDuration] = useState(60)

const [resLoading, setResLoading] = useState(false)

const [resError, setResError] = useState("")

const [cancelModalOpen, setCancelModalOpen] = useState(false)

const [cancelId, setCancelId] = useState(null)

const [toast, setToast] = useState({ type:"", msg:"" })

// Filters + pagination for "Your Bookings"

```

const [fld, setFld] = useLocalStorage('bookings_fld','')
const [fDate, setFDate] = useLocalStorage('bookings_fDate','')
const [fTime, setFTime] = useLocalStorage('bookings_fTime','')
const [fStatus, setFStatus] = useLocalStorage('bookings_fStatus','')
const [page, setPage] = useState(1)
const [perPage, setPerPage] = useLocalStorage('bookings_perPage',10)

// Derived filtering + pagination for user's bookings
const onInteract = ()=>{ setSuccessMsg(""); setError(""); setToast({type:"",msg:""}) }
const onEnter = (e)=>{ if(e.key==='Enter'){ e.preventDefault(); setPage(1); onInteract() } }
const filtered = useMemo(()=>{
  return bookings.filter(b => {
    if (fld!==' ' && String(b.id)!==String(fld)) return false
    if (fDate && String(b.date)!==String(fDate)) return false
    if (fTime && !(String(b.start_time | ' ').startsWith(fTime) || String(b.end_time | ' ').startsWith(fTime))) return false
    if (fStatus && String(b.status)!==String(fStatus)) return false
    return true
  })
}, [bookings, fld, fDate, fTime, fStatus])

const totalPages = useMemo(()=> Math.max(1, Math.ceil(filtered.length / Number(perPage | 10))),
[filtered.length, perPage])

const safePage = Math.min(page, totalPages)

const startIdx = (safePage - 1) * Number(perPage | 10)

const pageltems = useMemo(()=> filtered.slice(startIdx, startIdx + Number(perPage | 10)), [filtered, startIdx, perPage])

const clearFilters = ()=>{ setFld(""); setFDate(""); setFTime(""); setFStatus(""); setPerPage(10); setPage(1); onInteract() }

const WORK_START = '09:00'

```

```

const WORK_END = '18:00'

const toSeconds = (hhmm) => {
  const [h,m] = (hhmm || '00:00').split(':').map(Number)
  return (h*3600) + (m*60)
}

const secondsToHHMM = (secs) => {
  const h = Math.floor(secs/3600).toString().padStart(2,'0')
  const m = Math.floor((secs%3600)/60).toString().padStart(2,'0')
  return `${h}:${m}`
}

const maxStartHHMM = () => {
  const endSecs = toSeconds(WORK_END)
  const maxStartSecs = endSecs - (Number(resDuration || 60)*60)
  return secondsToHHMM(Math.max(0, maxStartSecs))
}

const todayStr = () => new Date().toISOString().slice(0,10)

useEffect(()=>{
  if (location.state && location.state.selected) {
    setSelected(location.state.selected)
  }
}, [location.state])

const loadBookings = async ()=>{
  if (!user) return
  setBookingsLoading(true); setError('')
  try{
    const res = await listBookings({ user_id: user.id })
    const rows = Array.isArray(res) ? res : (res?.data || [])
  }
}

```

```

    setBookings(rows)
  }catch(e){ setError('Failed to load your bookings') }
  finally{ setBookingsLoading(false) }
}

```

```

useEffect(()=>{ loadBookings() }, [user])

```

```

// SSE: real-time updates to user's bookings list

```

```

useEffect(()=>{
  if (!user) return
  let es
  try{
    const url = `${API}/api/stream/bookings`
    es = new EventSource(url, { withCredentials: true })
  }catch(e){ console.warn('SSE unavailable', e); return }
}

```

```

const handleUpsertById = async (id)=>{
  try{
    const res = await listBookings({ id })
    const rows = Array.isArray(res) ? res : (res?.data || [])
    if (rows.length===0) return
    const row = rows[0]
    if (row.user_id !== user.id) return
    setBookings(prev => {
      const idx = prev.findIndex(b => b.id === row.id)
      if (idx >= 0) {
        const next = prev.slice()
        next[idx] = { ...prev[idx], ...row }
        return next
      }
    })
  }
}

```



```

    }
    return [row, ...prev]
  })
}catch(err){ /* ignore */ }
}

```

```

es.addEventListener('status_changed', (ev)=>{
  try{ const payload = JSON.parse(ev.data); if (payload?.id) handleUpsertById(payload.id) }catch{ /*
ignore */ }
})
es.addEventListener('created', (ev)=>{
  try{ const payload = JSON.parse(ev.data); if (payload?.id) handleUpsertById(payload.id) }catch{ /*
ignore */ }
})
es.addEventListener('ping', ()=>{})
es.onerror = ()=>{ try{ es && es.close() }catch{} }
return ()=>{ try{ es && es.close() }catch{} }
}, [user])

```

```

return (
  <div className="container py-4">
    <div className="d-flex justify-content-between align-items-center mb-3">
      <h3 className="mb-0">Bookings</h3>
    </div>

```

```

    {error && <div className="text-muted small mb-2">{error}</div>}

```

```

    {selected ? (
      <BookingForm

```

```

service={selected}

onSuccess={async () => {
  setSelected(null)

  await loadBookings()

  setSuccessMsg('Booking created successfully!')

  setTimeout(()=>setSuccessMsg(''), 3000)
}}
/>
): (
  <div className="text-muted">Select a service from your Dashboard to start booking.</div>
)}

```

```

{successMsg && <div className="text-muted small mt-3">{successMsg}</div>}

```

```

<div className="mt-4">
  <h4>Your Bookings</h4>
  <div className="card p-3 mb-3" style={{ position:'sticky', top:8, zIndex:10 }}>
    <div className="row g-2 align-items-end">
      <div className="col-6 col-md-3 col-lg-2">
        <label className="form-label">#</label>
        <input type="number" className="form-control" value={fld} onChange={e=>{
setFld(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter} />
      </div>
      <div className="col-6 col-md-3 col-lg-4">
        <label className="form-label">Date</label>
        <input type="date" className="form-control" value={fDate} onChange={e=>{
setFDate(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter} />
      </div>
      <div className="col-6 col-md-3 col-lg-4">

```

```
<label className="form-label">Time</label>

<input type="time" className="form-control" value={fTime} onChange={e=>{
setFTime(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter} />

</div>

<div className="col-6 col-md-3 col-lg-2">

  <label className="form-label">Status</label>

  <select className="form-select" value={fStatus} onChange={e=>{ setFStatus(e.target.value);
setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter}>

    <option value="">All</option>

    <option value="pending">Pending</option>

    <option value="confirmed">Confirmed</option>

    <option value="canceled">Canceled</option>

    <option value="completed">Completed</option>

  </select>

</div>

</div>

<div className="row g-2 align-items-end mt-0">

  <div className="col-6 col-md-3 col-lg-3">

    <label className="form-label">Per page</label>

    <select className="form-select" value={perPage} onChange={e=>{
setPerPage(Number(e.target.value)); setPage(1); onInteract() }}>

      <option value={5}>5</option>

      <option value={10}>10</option>

      <option value={20}>20</option>

      <option value={50}>50</option>

    </select>

  </div>

  <div className="col-12 col-lg-9 d-flex justify-content-lg-end gap-2">

    <button className="btn btn-outline-secondary" onClick={()=>{ setPage(1); onInteract()
}}>Search</button>
```

```

        <button className="btn btn-outline-primary" onClick={clearFilters}>Clear</button>
    </div>
</div>
</div>
{bookingsLoading && <div className="text-muted small">Loading bookings...</div>}
{(!bookingsLoading && bookings.length===0) && <div className="text-muted">You have no
bookings yet.</div>}
{filtered.length===0 && !bookingsLoading ? (
    <div className="text-muted">No results matched your filters.</div>
) : (
    <div className="table-responsive">
        <table className="table table-striped">
            <thead>
                <tr><th>#</th><th>Date</th><th>Time</th><th>Service</th><th>Status</th><th
className="text-end">Actions</th></tr>
            </thead>
            <tbody>
                {pageItems.map(b => (
                    <tr key={b.id} onDoubleClick={()=>{ setResId(b.id); setResDate(b.date);
setResStart((b.start_time | "").slice(0,5)); setResDuration(Number(b.duration_minutes | |60));
setResError(""); setResModalOpen(true) }}>
                        <td>{b.id}</td>
                        <td>{b.date}</td>
                        <td>{b.start_time} - {b.end_time}</td>
                        <td>{b.service_name}</td>
                        <td><span className={`badge text-bg-
${b.status=== 'confirmed'? 'success':b.status=== 'pending'? 'warning':b.status=== 'canceled'? 'danger': 'sec
ondary'}`}>{b.status}</span></td>
                        <td className="text-end">
                            {(b.status=== 'pending' && (Number(b.reschedule_count | |0) < 1 | |
Number(b.cancel_count | |0) < 1)) && (

```

```

<>

    {Number(b.reschedule_count | 0) < 1 && (
        <button className="btn btn-sm btn-outline-primary me-2" onClick={()=>{
setResId(b.id); setResDate(b.date); setResStart((b.start_time | "").slice(0,5));
setResDuration(Number(b.duration_minutes | 60)); setResError(""); setResModalOpen(true)
}}>Edit</button>

        )}

        {Number(b.cancel_count | 0) < 1 && (
            <button className="btn btn-sm btn-outline-danger" onClick={()=>{ setCancelId(b.id);
setCancelModalOpen(true) }}>Cancel</button>

            )}

        </>

    )}

    {{(b.status==='confirmed' && Number(b.cancel_count | 0) < 1) && (
        <button className="btn btn-sm btn-outline-danger" onClick={()=>{ setCancelId(b.id);
setCancelModalOpen(true) }}>Cancel</button>

        )}

    </td>

</tr>

)}}

</tbody>

</table>

</div>

)}

<div className="d-flex justify-content-between align-items-center mt-2">

    <div className="small text-muted">Showing {filtered.length === 0 ? 0 : (startIndex+1)}–
{Math.min(startIdx+perPage, filtered.length)} of {filtered.length}</div>

    <div className="btn-group" role="group" aria-label="Pagination">

        <button type="button" className="btn btn-outline-secondary btn-sm" disabled={safePage<=1}
onClick={()=>{ onInteract(); setPage(p=>Math.max(1, p-1)) }}>Prev</button>

        <span className="btn btn-outline-secondary btn-sm disabled">Page {safePage} /
{totalPages}</span>

```

```
        <button type="button" className="btn btn-outline-secondary btn-sm"
disabled={safePage>=totalPages} onClick={()=>{ onInteract(); setPage(p=>Math.min(totalPages, p+1))
}}>Next</button>
```

```
    </div>
```

```
  </div>
```

```
</div>
```

```
{/* Cancel Modal */}
```

```
{cancelModalOpen && (
```

```
<>
```

```
<div className="modal fade show d-block" tabIndex="-1" role="dialog" aria-modal="true">
```

```
<div className="modal-dialog">
```

```
<div className="modal-content">
```

```
<div className="modal-header">
```

```
<h5 className="modal-title">Cancel booking</h5>
```

```
<button type="button" className="btn-close" aria-label="Close" onClick={()=>{
if(!bookingsLoading){ setCancelModalOpen(false); setCancelId(null) } }}></button>
```

```
</div>
```

```
<div className="modal-body">
```

```
<p>Are you sure you want to cancel booking #{cancelId}?</p>
```

```
</div>
```

```
<div className="modal-footer">
```

```
<button type="button" className="btn btn-outline-secondary" onClick={()=>{
if(!bookingsLoading){ setCancelModalOpen(false); setCancelId(null) } }}>Close</button>
```

```
<button type="button" className="btn btn-danger" disabled={bookingsLoading}
onClick={async()=>{
```

```
  const id = cancelId
```

```
  setCancelModalOpen(false); setCancelId(null)
```

```
  try{
```

```
    setBookingsLoading(true)
```

```

        await cancelBooking(id)

        const res = await listBookings({ user_id: user.id })

        const rows = Array.isArray(res) ? res : (res?.data || [])

        setBookings(rows)

        setToast({ type:'success', msg:'Booking canceled' })

    }catch(e){ setToast({ type:'error', msg: (e && e.message) || 'Cancel failed' }) }

    finally{ setBookingsLoading(false) }

  }}>Cancel booking</button>

</div>

</div>

</div>

</div>

<div className="modal-backdrop fade show"></div>

</>

  }}

```

```

{resModalOpen && (
  <div className="modal d-block" tabIndex="-1" role="dialog"
style={{background:'rgba(0,0,0,0.5)}}>
    <div className="modal-dialog" role="document">
      <div className="modal-content">
        <div className="modal-header">
          <h5 className="modal-title">Reschedule Booking #{resId}</h5>
          <button type="button" className="btn-close" aria-label="Close" onClick={()=>{
if(!resLoading){ setResModalOpen(false) } }} />
        </div>
        <div className="modal-body">
          {resError && <div className="alert alert-danger">{resError}</div>}
          <div className="alert alert-info py-2">Accepting bookings: {WORK_START}–
{WORK_END}</div>

```

```

<div className="mb-3">

  <label className="form-label" htmlFor="res_date">Date</label>

  <input id="res_date" name="date" type="date" className="form-control" min={todayStr()}
value={resDate} onChange={e=>setResDate(e.target.value)} />

</div>

<div className="mb-2">

  <label className="form-label" htmlFor="res_start_time">Start Time</label>

  <input id="res_start_time" name="start_time" type="time" className="form-control"
step="60" min={WORK_START} max={maxStartHHMM()} value={resStart}
onChange={e=>setResStart((e.target.value || "").slice(0,5))} />

</div>

<div className="text-muted small">Note: Only pending bookings can be rescheduled.</div>

</div>

<div className="modal-footer">

  <button type="button" className="btn btn-secondary" onClick={()=>{ if(!resLoading){
setResModalOpen(false) } }}>Close</button>

  <button type="button" className="btn btn-primary" disabled={resLoading}
onClick={async()=>{

    const id = resId; const date = resDate; const s = resStart; const dur =
Number(resDuration || 60)

    setResModalOpen(false)

    setResLoading(true); setResError("")

    try{

      // Client validation

      if (!date || date < todayStr()) { throw new Error('Please pick today or a future date.') }

      const eSecs = toSeconds(s) + (dur*60)

      if (s < WORK_START || s > maxStartHHMM() || eSecs > toSeconds(WORK_END)) {

        throw new Error(`Please pick a time within working hours (${WORK_START}–
${WORK_END}) that fits the service duration.`)

      }

      await rescheduleBooking({ id, date, start_time: `${s}:00` })

    }

  }}>Reschedule</button>

</div>

```



```

        // refresh list

        const res = await listBookings({ user_id: user.id })

        const rows = Array.isArray(res) ? res : (res?.data || [])

        setBookings(rows)

        setToast({ type:'success', msg:'Booking rescheduled' })

    }catch(e){ setToast({ type:'error', msg: (e && e.message) || 'Reschedule failed' }) }

    finally{ setResLoading(false) }

    }}>Save</button>

</div>

</div>

</div>

)}

{/* Toasts */}

<div style={{position:'fixed', right:16, bottom:16, zIndex:9999}}>

    {toast.msg && (

        <div className={`toast align-items-center text-white ${toast.type==='error'? 'bg-danger': 'bg-success'} show`} role="alert">

            <div className="d-flex">

                <div className="toast-body">{toast.msg}</div>

                <button type="button" className="btn-close btn-close-white me-2 m-auto"
onClick={()=>setToast({type:"",msg:"})}></button>

            </div>

        </div>

    )}

</div>

</div>

)

```

}

frontend/src/pages/Dashboard.jsx

// Dashboard.jsx — Admin analytics: summary stats, service coverage, bookings overview (filters, CSV, quick-view modal).

import React, { useEffect, useState } from 'react'

import useModal from '../hooks/useModal'

import { getSummary, getServiceCoverage, getAnalyticsBookings } from '../services/analyticsService'

export default function Dashboard(){

const [data,setData] = useState(null)

const [loading,setLoading] = useState(false)

const [error,setError] = useState("")

const [toast, setToast] = useState({ type:"", msg:"" })

const [coverage,setCoverage] = useState(null)

const [covLoading,setCovLoading] = useState(false)

const [boRows, setBoRows] = useState([])

const [boTotal, setBoTotal] = useState(0)

const [boPage, setBoPage] = useState(1)

const [boPerPage, setBoPerPage] = useState(10)

const [boQ, setBoQ] = useState("")

const [boStatus, setBoStatus] = useState("")

const [boLoading, setBoLoading] = useState(false)

const [boQFocus, setBoQFocus] = useState(false)

const [fromFocus, setFromFocus] = useState(false)

const [toFocus, setToFocus] = useState(false)

const [selectedBooking, setSelectedBooking] = useState(null)

const { open: bookingOpen, openModal: openBookingModal, closeModal: closeBookingModal, onKeyDown: bookingKeyDown, ref: bookingRef } = useModal(false)

```

const fmt = (d)=>{
  const y = d.getFullYear();
  const m = String(d.getMonth()+1).padStart(2,'0');
  const day = String(d.getDate()).padStart(2,'0');
  return `${y}-${m}-${day}`
}

const StatCard = ({ label, value, icon }) => {
  return (
    <div className="card h-100">
      <div className="card-body d-flex align-items-center gap-3">
        <div style={{width:40,height:40,borderRadius:8,background:'var(--gu-mist)',display:'flex',alignItems:'center',justifyContent:'center',color:'var(--gu-primary-600)}}>
          {icon}
        </div>
        <div>
          <div className="text-muted small">{label}</div>
          <div className="h4 m-0">{value}</div>
        </div>
      </div>
    </div>
  )
}

```

// Line chart with markers, tooltips, axes, gridlines, and 7-day moving average

```

const LineChart = ({ points, height=160 }) => {
  const m = { top: 8, right: 8, bottom: 24, left: 36 }
  const n = points.length
  const innerH = height - m.top - m.bottom
  const maxVal = points.reduce((mx,p)=> Math.max(mx, Number(p.cnt|0)), 0)

```

```

const safeMax = maxVal > 0 ? maxVal : 1

const innerW = Math.max(260, n>1 ? n*22 : 260)

const width = innerW + m.left + m.right

const x = (i)=> m.left + (n>1 ? (i * (innerW / (n-1))) : innerW/2)

const y = (v)=> m.top + (innerH - (Number(v)/safeMax) * innerH)

const polyMain = n ? points.map((p,i)=> `${x(i)},${y(p.cnt|0)}`).join(' ') : ''

```

```

// 7-day moving average overlay

const avgPoints = points.map((p,i)=>{

  const start = Math.max(0, i-6)

  const win = points.slice(start, i+1)

  const avg = win.reduce((s,q)=> s + Number(q.cnt|0), 0) / win.length

  return { date: p.date, cnt: avg }

})

const polyAvg = n ? avgPoints.map((p,i)=> `${x(i)},${y(p.cnt)}`).join(' ') : ''

```

```

// y-axis ticks (5 levels)

const steps = 4

const stepVal = Math.ceil(safeMax / steps)

const yTicks = Array.from({ length: steps + 1 }, (_,i)=> i * stepVal)

// x-axis ticks ~5 labels

const xTickEvery = Math.max(1, Math.ceil(n / 5))

return (

  <svg width={width} height={height} style={{maxWidth:'100%'}}>

    /* gridlines + y labels */

    {yTicks.map((t,i)=> (

      <g key={`y${i}`}>

```

```

        <line x1={m.left} x2={width-m.right} y1={y(t)} y2={y(t)} stroke="var(--gu-soft)" />

        <text x={m.left-6} y={y(t)} textAnchor="end" dominantBaseline="middle" className="small"
fill="var(--gu-muted)">{t}</text>

    </g>

    )})

    { /* x-axis baseline */ }

    <line x1={m.left} x2={width-m.right} y1={height-m.bottom} y2={height-m.bottom} stroke="var(--
gu-soft)" />

    {points.map((p,i)=> (i % xTickEvery === 0 ? (

        <text key={`x${i}`} x={x(i)} y={height-6} textAnchor="middle" className="small" fill="var(--gu-
muted)">{String(p.date | "").slice(5)}</text>

        ) : null))}

    { /* average overlay */ }

    <polyline points={polyAvg} fill="none" stroke="var(--gu-accent)" strokeWidth="2"
strokeDasharray="4 4" />

    { /* main line */ }

    <polyline points={polyMain} fill="none" stroke="var(--gu-primary)" strokeWidth="2.5" />

    { /* markers with native tooltip */ }

    {points.map((p,i)=> (

        <g key={`m${i}`}>

            <circle cx={x(i)} cy={y(p.cnt | 0)} r="3.5" fill="#fff" stroke="var(--gu-primary)" />

            <title>`${p.date}: ${p.cnt}`</title>

        </g>

    )})

    </svg>

    )

}

```

// Simple donut PieChart with legend

```

const PieChart = ({ data, size=220, inner=0.6, colors }) => {
  const [active, setActive] = useState(-1)

  const total = data.reduce((s,d)=> s + Number(d.value || 0), 0)

  const cx = size/2, cy = size/2

  const r = (size/2) - 8

  const ir = r * inner

  const toXY = (angle, radius)=> [cx + radius*Math.cos(angle), cy + radius*Math.sin(angle)]

  const arcs = []

  let start = -Math.PI/2 // start at top

  const palette = ['var(--gu-primary)', 'var(--gu-accent)', 'var(--gu-primary-600)', 'rgba(73,34,91,.35)', 'rgba(165,106,189,.35)', 'rgba(73,34,91,.2)']

  data.forEach((d, i)=>{
    const val = Number(d.value || 0)

    const slice = total>0 ? (val/total) * Math.PI*2 : 0

    const end = start + slice

    const [sx, sy] = toXY(start, r)

    const [ex, ey] = toXY(end, r)

    const [isx, isy] = toXY(end, ir)

    const [iex, iey] = toXY(start, ir)

    const largeArc = slice > Math.PI ? 1 : 0

    const color = (colors && colors[i]) ? colors[i] : palette[i % palette.length]

    const path = [
      `M ${sx} ${sy}`,
      `A ${r} ${r} 0 ${largeArc} 1 ${ex} ${ey}`,
      `L ${isx} ${isy}`,
      `A ${ir} ${ir} 0 ${largeArc} 0 ${iex} ${iey}`,
      'Z'
    ].join(' ')

    arcs.push({ path, color, label: d.label, value: val, start, end })
  })

```

```

    start = end
  })
  const handleMove = (e)=>{
    if (total<=0) { setActive(-1); return }
    const rect = e.currentTarget.getBoundingClientRect()
    const x = e.clientX - rect.left
    const y = e.clientY - rect.top
    const dx = x - cx
    const dy = y - cy
    const dist = Math.sqrt(dx*dx + dy*dy)
    if (dist < ir || dist > r) { setActive(-1); return }
    let ang = Math.atan2(dy, dx) // [-PI, PI], 0 at +x
    let fromTop = ang - (-Math.PI/2) // rotate so 0 at top
    while (fromTop < 0) fromTop += Math.PI*2
    const frac = fromTop / (Math.PI*2)
    // find slice by cumulative fraction to avoid angle wrap logic
    let acc = 0
    for (let i=0;i<data.length;i++){
      const val = Number(data[i].value || 0)
      const next = acc + (total>0 ? val/total : 0)
      if (frac >= acc && frac < next){ setActive(i); return }
      acc = next
    }
    setActive(-1)
  }
  return (
    <div className="d-flex flex-wrap align-items-center gap-3">
      <svg width={size} height={size} viewBox={`0 0 ${size} ${size}`} onMouseMove={handleMove}
onMouseLeave={()=>setActive(-1)}>

```



```

    {arcs.map((a,idx)=> (
      <path key={idx} d={a.path} fill={a.color} opacity={active===-1 || active===idx ? 1 : 0.5}
stroke="#fff" strokeWidth={active===idx ? 2 : 1} />
    )}}
    {/* center hole to ensure crisp donut if total is 0 */}
    {total===0 && (
      <circle cx={cx} cy={cy} r={r} fill="var(--gu-mist)" />
    )}
  </svg>
  <div className="d-grid small" style={{minWidth: 180}}>
    <div className="mb-1">
      {active>=0 && total>0 ? (
        <strong>{data[active].label}: {Number(data[active].value | 0)}
({Math.round((Number(data[active].value | 0)/total)*100)}%)</strong>
      ) : (
        <span className="text-muted">Drag/hover over chart to preview</span>
      )}
    </div>
    {data.map((d,i)=> (
      <div key={i} className="d-flex align-items-center justify-content-between gap-2">
        <div className="d-flex align-items-center gap-2">
          <span style={{display:'inline-block',width:10,height:10,borderRadius:2,background: ((colors
&& colors[i]) ? colors[i] : ([var(--gu-primary)',var(--gu-accent)',var(--gu-primary-
600)','rgba(73,34,91,.35)','rgba(165,106,189,.35)','rgba(73,34,91,.2)')[i % 6])}} />
          <span style={{fontWeight: active===i ? 600 : 400}}>{d.label}</span>
        </div>
        <span className="text-muted">{Number(d.value | 0)}</span>
      </div>
    )}}
  </div>

```

```

    </div>
  )
}

const computeThisWeek = ()=>{
  const t = new Date();
  const dow = t.getDay(); // 0 Sun .. 6 Sat
  const diffToMon = (dow===0 ? -6 : 1 - dow);
  const mon = new Date(t); mon.setHours(0,0,0,0); mon.setDate(t.getDate()+diffToMon);
  const sun = new Date(mon); sun.setDate(mon.getDate()+6);
  return { from: fmt(mon), to: fmt(sun) }
}

const initWeek = computeThisWeek()
const [from,setFrom] = useState(initWeek.from)
const [to,setTo] = useState(initWeek.to)

const load = async (opts={}) => {
  const useFrom = opts.from ?? from
  const useTo = opts.to ?? to
  setLoading(true); setError("")
  try{
    const res = await getSummary({ from: useFrom, to: useTo })
    setData(res)
  }catch(e){ setError('Failed to load analytics') }
  finally{ setLoading(false) }
}

useEffect(()=>{ load() },[])
useEffect(()=>{ if (error) setToast({ type:'error', msg:error }) }, [error])

```

```
useEffect(()=>{  
  const run = async()=>{  
    setCovLoading(true)  
    try{  
      const res = await getServiceCoverage({ from, to })  
      setCoverage(res)  
    }catch(e){ /* ignore */ }  
    finally{ setCovLoading(false) }  
  }  
  run()  
}, [from, to])
```

```
useEffect(()=>{  
  const run = async()=>{  
    setBoLoading(true)  
    try{  
      const res = await getAnalyticsBookings({ from, to, q: boQ, status: boStatus, page: boPage,  
per_page: boPerPage })  
      setBoRows(Array.isArray(res.rows)? res.rows : [])  
      setBoTotal(Number(res.total) || 0)  
    }catch(e){ /* noop */ }  
    finally{ setBoLoading(false) }  
  }  
  run()  
}, [from, to, boQ, boStatus, boPage, boPerPage])
```

```
const computeLast7Days = ()=>{  
  const end = new Date(); end.setHours(0,0,0,0)  
  const start = new Date(end); start.setDate(end.getDate()-6)
```

```

    return { from: fmt(start), to: fmt(end) }
  }

  const computeThisMonth = ()=>{
    const now = new Date();
    const start = new Date(now.getFullYear(), now.getMonth(), 1)
    const end = new Date(now.getFullYear(), now.getMonth()+1, 0)
    return { from: fmt(start), to: fmt(end) }
  }

```

```

const Bar = ({ label, value, max, color='var(--gu-primary)' }) => {
  const pct = max>0 ? Math.round((value/max)*100) : 0
  return (
    <div className="mb-2">
      <div className="d-flex justify-content-between"><span>{label}</span><span className="text-
muted small">{value}</span></div>
      <div className="bg-light" style={{height:8, borderRadius:4}}>
        <div style={{width:`${pct}%`, height:8, background:color, borderRadius:4}} />
      </div>
    </div>
  )
}

```

```

return (
  <div className="container py-4">
    <div className="d-flex flex-wrap justify-content-between align-items-end gap-2 mb-3">
      <div>
        <h3 className="mb-1">Admin Analytics</h3>

```

```

    <div className="text-muted small">Overview of bookings · {from} → {to}</div>

</div>

<div className="d-flex gap-3 flex-wrap align-items-end">

  <div>

    <label className="form-label" htmlFor="dash_from">From</label>

    <input id="dash_from" name="from" type="date" className="form-control" value={from}
    onChange={e=>setFrom(e.target.value)} onKeyDown={e=>{ if(e.key==='Enter'){ e.preventDefault();
    load() } }} onFocus={()=>setFromFocus(true)} onBlur={()=>setFromFocus(false)} style={fromFocus?{
    boxShadow:'0 0 0 .2rem rgba(111,66,193,.25)', borderColor:'#6f42c1':undefined} />

    </div>

    <div>

      <label className="form-label" htmlFor="dash_to">To</label>

      <input id="dash_to" name="to" type="date" className="form-control" value={to}
      onChange={e=>setTo(e.target.value)} onKeyDown={e=>{ if(e.key==='Enter'){ e.preventDefault(); load()
      } }} onFocus={()=>setToFocus(true)} onBlur={()=>setToFocus(false)} style={toFocus?{ boxShadow:'0 0 0
      .2rem rgba(111,66,193,.25)', borderColor:'#6f42c1':undefined} />

      </div>

    <div className="d-flex align-items-center gap-2">

      <button className="btn btn-outline-secondary" onClick={load}>Apply</button>

      <div className="btn-group" role="group" aria-label="Range presets">

        <button type="button" className="btn btn-outline-primary btn-sm" onClick={()=>{ const
        r=computeThisWeek(); setFrom(r.from); setTo(r.to); load({ from: r.from, to: r.to }); }}>{'This
        Week'}</button>

        <button type="button" className="btn btn-outline-primary btn-sm" onClick={()=>{ const
        r=computeLast7Days(); setFrom(r.from); setTo(r.to); load({ from: r.from, to: r.to }); }}>{'Last 7
        Days'}</button>

        <button type="button" className="btn btn-outline-primary btn-sm" onClick={()=>{ const
        r=computeThisMonth(); setFrom(r.from); setTo(r.to); load({ from: r.from, to: r.to }); }}>{'This
        Month'}</button>

      </div>

    </div>

  </div>

</div>

```

```

{loading && <div className="text-muted small mb-2">Loading...</div>}

{data && (
  <div className="row g-3">
    {(() => {
      const rows = data.by_status || []
      const getCount = (s) => Number((rows.find(r => r.status === s)?.cnt) || 0)
      const confirmed = getCount('confirmed') + getCount('completed')
      const pending = getCount('pending')
      const canceled = getCount('canceled')
      const total = data.totals?.bookings ?? (confirmed + pending + canceled)

      return (
        <>
          <div className="col-6 col-md-3">
            <StatCard label="Total Bookings" value={total} icon={<svg width="18" height="18"
viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg"><path d="M4 4h16v14a2 2 0
0 1-2 2H6a2 2 0 0 1-2-2V4z" stroke="currentColor"/><path d="M4 8h16"
stroke="currentColor"/></svg>} />
            </div>
            <div className="col-6 col-md-3">
              <StatCard label="Confirmed" value={confirmed} icon={<svg width="18" height="18"
viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg"><path d="M20 6L9 17l-5-5"
stroke="currentColor" strokeWidth="2"/></svg>} />
              </div>
              <div className="col-6 col-md-3">
                <StatCard label="Pending" value={pending} icon={<svg width="18" height="18" viewBox="0
0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg"><circle cx="12" cy="12" r="9"
stroke="currentColor"/><path d="M12 7v5l3 3" stroke="currentColor"/></svg>} />
                </div>
                <div className="col-6 col-md-3">
                  <StatCard label="Canceled" value={canceled} icon={<svg width="18" height="18"
viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg"><path d="M6 6l12 12M18 6L6
18" stroke="currentColor"/></svg>} />

```

```

    </div>
  </>
)
})();

```

```

{ /* Revenue summary cards */

{(()=>{

  const rev = data.revenue || { total:0, avg_per_day:0, avg_duration_minutes:0 }

  return (

    <>

    <div className="col-6 col-md-4">

      <StatCard label="Total Revenue" value={`₹${Number(rev.total || 0).toFixed(2)}`} icon={<svg
width="18" height="18" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg"><path d="M4 7h16v10a2 2 0 0 1-2 2H6a2 2 0 0 1-2-2V7z"
stroke="currentColor"/><path d="M4 7l2-3h12l2 3" stroke="currentColor"/></svg>} />

      </div>

      <div className="col-6 col-md-4">

        <StatCard label="Avg Revenue / Day" value={`₹${Number(rev.avg_per_day || 0).toFixed(2)}`}
icon={<svg width="18" height="18" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg"><path d="M4 4h16v16H4z" stroke="currentColor"/><path
d="M8 2v4M16 2v4M4 10h16" stroke="currentColor"/></svg>} />

        </div>

        <div className="col-6 col-md-4">

          <StatCard label="Avg Duration (mins)"
value={`${Number(rev.avg_duration_minutes || 0).toFixed(1)}`} icon={<svg width="18" height="18"
viewBox="0 0 24 24" fill="none" xmlns="http://www.w3.org/2000/svg"><circle cx="12" cy="12" r="9"
stroke="currentColor"/><path d="M12 7v5l3 3" stroke="currentColor"/></svg>} />

          </div>

        </>

      )

    })()
  }
}

```

```

<div className="col-12 col-md-6">

  <div className="card"><div className="card-body">

    <h5 className="card-title">Status Breakdown</h5>

    {(())=>{

      const rows = data.by_status || []

      const by = Object.fromEntries(rows.map(r=> [r.status, Number(r.cnt || 0)]))

      const series = [

        { label:'confirmed', value:Number(by['confirmed'] || 0) },

        { label:'completed', value:Number(by['completed'] || 0) },

        { label:'pending', value:Number(by['pending'] || 0) },

        { label:'canceled', value:Number(by['canceled'] || 0) },

      ]

      const colors = ['var(--gu-primary)', 'var(--gu-primary)', 'var(--gu-accent)', 'var(--gu-primary-
600)']

      const nonZero = series.some(s => s.value>0)

      return !nonZero ? <div className="text-muted">No data</div> : <PieChart data={series}
colors={colors} />

    }}()

  </div></div>

</div>

<div className="col-12 col-md-6">

  <div className="card"><div className="card-body">

    <h5 className="card-title">Category Breakdown</h5>

    {(())=>{

      const rows = data.by_category || []

      const series = rows.map(r=> ({ label: r.category, value: Number(r.cnt || 0) }))

      return series.length===0 ? <div className="text-muted">No data</div> : <PieChart
data={series} />

    }}()

  </div></div>

```


</div>

{/* Revenue by Category */}

<div className="col-12">

<div className="card"><div className="card-body">

<h5 className="card-title">Revenue by Category</h5>

{{()=>{

const rows = data.revenue_by_category || []

if (rows.length===0) return <div className="text-muted">No data</div>

const max = rows.reduce((m,r)=> Math.max(m, Number(r.total_rev | 0)), 0)

return rows.map((r,i)=> (

<Bar key={i} label={r.category} value={Number(r.total_rev | 0)} max={max} color="--gu-primary-600" />

))

}}())}

</div></div>

</div>

<div className="col-12">

<div className="card"><div className="card-body">

<h5 className="card-title">Daily Bookings</h5>

{{()=>{

const pts = data.daily || []

const total = pts.reduce((s,p)=> s + Number(p.cnt | 0), 0)

const days = pts.length || 1

const avg = total / days

let peak = { date: '-', cnt: 0 }

pts.forEach(p=>{ const c = Number(p.cnt | 0); if (c>peak.cnt) peak = { date: p.date, cnt: c } })

return (

```

<div className="row g-3 align-items-start">

  <div className="col-md-8">

    <LineChart points={pts} />

  </div>

  <div className="col-md-4">

    <div className="small text-muted mb-2">Summary</div>

    <ul className="list-unstyled m-0 d-grid gap-1 small">

      <li><strong>Total bookings:</strong> {total}</li>

      <li><strong>Average/day:</strong> {avg.toFixed(1)}</li>

      <li><strong>Peak day:</strong> {peak.date} {{peak.cnt}}</li>

    </ul>

  </div>

</div>

)

})();

</div></div>

</div>

<div className="col-12">

  <div className="card"><div className="card-body">

    <h5 className="card-title">Top Services</h5>

    {{()=>{

      const rows = data.top_services || []

      const max = rows.reduce((m,r)=>Math.max(m, Number(r.cnt || 0)),0)

      return rows.length===0 ? <div className="text-muted">No data</div> : rows.map((r,i)=> (

        <Bar key={i} label={r.name} value={Number(r.cnt || 0)} max={max} />

      ))

    }}

  </div></div>

</div>

```

```

<div className="col-12">

  <div className="card"><div className="card-body">

    <h5 className="card-title">Service Coverage</h5>

    {covLoading ? (

      <div className="text-muted">Loading...</div>

    ) : (!coverage || !Array.isArray(coverage.services) || coverage.services.length===0) ? (

      <div className="text-muted">No data</div>

    ) : (

      <div className="table-responsive">

        <table className="table table-sm">

          <thead><tr><th>#</th><th>Service</th><th className="text-
end">Bookings</th></tr></thead>

          <tbody>

            {coverage.services.map((r,i)=> (

              <tr key={r.id | i}><td>{r.id}</td><td>{r.name}</td><td className="text-
end">{Number(r.bookings | 0)}</td></tr>

            ))}

          </tbody>

        </table>

      </div>

    )}

  </div></div>

</div>

```

```

<div className="col-12">

  <div className="card"><div className="card-body">

    <div className="d-flex justify-content-between align-items-center mb-2">

      <h5 className="card-title mb-0">Bookings Overview</h5>

```

```

<div className="d-flex gap-2">

  <div className="input-group input-group-sm" style={{width:260}}>

    <span className="input-group-text">Search</span>

    <input className="form-control" value={boQ} onChange={e=>{ setBoPage(1);
setBoQ(e.target.value) }} onKeyDown={e=>{ if(e.key==='Enter'){ e.preventDefault(); setBoPage(1) }}
onFocus={()=>setBoQFocus(true)} onBlur={()=>setBoQFocus(false)} style={boQFocus?{ boxShadow:'0 0
0 .2rem rgba(111,66,193,.25)', borderColor:'#6f42c1' }:undefined} />

  </div>

  <select className="form-select form-select-sm" style={{width:150}} value={boStatus}
onChange={e=>{ setBoPage(1); setBoStatus(e.target.value) }}>

    <option value="">All Status</option>

    <option value="pending">pending</option>

    <option value="confirmed">confirmed</option>

    <option value="canceled">canceled</option>

    <option value="completed">completed</option>

  </select>

  <select className="form-select form-select-sm" style={{width:100}} value={boPerPage}
onChange={e=>{ setBoPage(1); setBoPerPage(Number(e.target.value)) }}>

    <option value={10}>10</option>

    <option value={20}>20</option>

    <option value={50}>50</option>

    <option value={100}>100</option>

  </select>

  <button className="btn btn-outline-secondary btn-sm" onClick={()=>{

    const cols = ['ID','Date','Start','End','Status','Customer','Service','Price','Notes']

    const lines = [cols.join(',')].concat(boRows.map(r=>[

      r.id,

      r.date,

      (r.start_time || '').slice(0,5),

      (r.end_time || '').slice(0,5),

```

```

    r.status,
    ""+String(r.customer_name | | "").replaceAll("'", "'")+"",
    ""+String(r.service_name | | "").replaceAll("'", "'")+"",
    Number(r.price | | 0).toFixed(2),
    ""+String(r.notes | | "").replaceAll("'", "'")+"",
  ].join(',')
))

const blob = new Blob([lines.join('\n')], { type:'text/csv;charset=utf-8;' })
const url = URL.createObjectURL(blob)
const a = document.createElement('a')
a.href = url
a.download = `bookings_${from}_to_${to}.csv`
document.body.appendChild(a)
a.click()
a.remove()
URL.revokeObjectURL(url)
}}>Export CSV</button>

```

```
</div>
```

```
</div>
```

```

{boLoading ? (
  <div className="text-muted">Loading...</div>
) : boRows.length===0 ? (
  <div className="text-muted">No data</div>
) : (
  <div className="table-responsive">
    <table className="table table-sm">
      <thead>
        <tr>
          <th>#</th>

```

```

        <th>Date</th>

        <th>Start</th>

        <th>End</th>

        <th>Status</th>

        <th>Customer</th>

        <th>Service</th>

        <th className="text-end">Price</th>

        <th>Notes</th>

    </tr>

</thead>

<tbody>

    {boRows.map((r)=> (

        <tr key={r.id} onDoubleClick={()=>{ setSelectedBooking(r); openBookingModal() }}
style={{cursor:'zoom-in'}}>

            <td>{r.id}</td>

            <td>{r.date}</td>

            <td>{(r.start_time || "").slice(0,5)}</td>

            <td>{(r.end_time || "").slice(0,5)}</td>

            <td>{r.status}</td>

            <td>{r.customer_name}</td>

            <td>{r.service_name}</td>

            <td className="text-end">₹{Number(r.price || 0).toFixed(2)}</td>

            <td>{r.notes}</td>

        </tr>

    )})

</tbody>

</table>

</div>

))

```

```

<div className="d-flex justify-content-between align-items-center mt-2">
  <div className="small text-muted">Showing {(boRows.length===0)?0:((boPage-
1)*boPerPage+1)}-{Math.min(boPage*boPerPage, boTotal)} of {boTotal}</div>

  <div className="btn-group" role="group">

    <button className="btn btn-outline-secondary btn-sm" disabled={boPage<=1}
onClick={()=>setBoPage(p=>Math.max(1,p-1))}>Prev</button>

    <span className="btn btn-outline-secondary btn-sm disabled">Page {boPage} /
{Math.max(1, Math.ceil(boTotal/boPerPage))}</span>

    <button className="btn btn-outline-secondary btn-sm" disabled={boPage>=Math.max(1,
Math.ceil(boTotal/boPerPage))} onClick={()=>setBoPage(p=>p+1)}>Next</button>

  </div>
</div>

{bookingOpen && (
  <>
    <div className="modal fade show d-block" tabIndex="-1" role="dialog" aria-modal="true"
onKeyDown={bookingKeyDown}>
      <div className="modal-dialog">
        <div className="modal-content" ref={bookingRef}>
          <div className="modal-header">
            <h5 className="modal-title">Booking Details</h5>
            <button type="button" className="btn-close" aria-label="Close"
onClick={closeBookingModal}></button>
          </div>
          <div className="modal-body">
            {selectedBooking ? (
              <div className="small">
                <div><strong>ID:</strong> {selectedBooking.id}</div>
                <div><strong>Date:</strong> {selectedBooking.date}</div>
                <div><strong>Time:</strong> {(selectedBooking.start_time || "").slice(0,5)} –
{(selectedBooking.end_time || "").slice(0,5)}</div>

```

```

        <div><strong>Status:</strong> {selectedBooking.status}</div>

        <div><strong>Customer:</strong> {selectedBooking.customer_name}</div>

        <div><strong>Service:</strong> {selectedBooking.service_name}</div>

        <div><strong>Price:</strong> ₪{Number(selectedBooking.price | | 0).toFixed(2)}</div>

        <div><strong>Notes:</strong> {selectedBooking.notes || '—'}</div>

        <div><strong>Created:</strong> {selectedBooking.created_at || '—'}</div>

    </div>

    ) : (

        <div className="text-muted">No booking selected.</div>

    )}

</div>

<div className="modal-footer">

    <button type="button" className="btn btn-outline-secondary"
onClick={closeBookingModal}>Close</button>

</div>

</div>

</div>

</div>

<div className="modal-backdrop fade show"></div>

</>

)}

</div></div>

</div>

</div>

)}

{/* Toasts */}

<div style={{position:'fixed', right:16, bottom:16, zIndex:9999}}>

    {toast.msg && (

```



```
<div className={`toast align-items-center text-white ${toast.type==='error'? 'bg-danger': 'bg-
success'} show`} role="alert">

  <div className="d-flex">

    <div className="toast-body">{toast.msg}</div>

    <button type="button" className="btn-close btn-close-white me-2 m-auto"
onClick={()=>setToast({type:"",msg:"})}></button>

  </div>

</div>

  )
}
```

frontend/src/pages/Home.jsx

// Home.jsx — Public landing page: hero, category search, featured and all services, testimonials.

```
import React, { useEffect, useState } from 'react'
import { useNavigate, Navigate } from 'react-router-dom'
import { listServices } from '../services/serviceService'
import ServiceCard from '../components/ServiceCard'
import BookingForm from '../components/BookingForm'
import { useAuthContext } from '../contexts/AuthContext'
```

```
export default function Home(){
  const { user } = useAuthContext()
  const navigate = useNavigate()
  const [services, setServices] = useState([])
  const [featured, setFeatured] = useState([])
  const [showFeatured, setShowFeatured] = useState(true)
  const [loading, setLoading] = useState(false)
  const [error, setError] = useState("")
  const [selected, setSelected] = useState(null)
  const [successMsg, setSuccessMsg] = useState("")
  const [q, setQ] = useState("")
  const [priceMax, setPriceMax] = useState("")
  const [category, setCategory] = useState("")
  const [toast, setToast] = useState({ type:"", msg:" })
  const [visibleCount, setVisibleCount] = useState(12)

  if (user) {
    return <Navigate to={user.role === 'admin' ? '/admin/dashboard' : '/dashboard'} replace />
  }
}
```

```
const fetchAll = async ()=>{
  setLoading(true)
  setError('')
  try{
    const res = await listServices({ active: 1, sort: 'created_at', order: 'DESC' })
    const rows = res.data || []
    setServices(rows)
    setFeatured(rows.slice(0,6))
  }catch(e){ setError('Failed to load services') }
  finally{ setLoading(false) }
}
useEffect(()=>{ fetchAll() }, [])
```

```
const scrollToFeaturedAnchor = ()=>{
  const el = document.getElementById('featuredAnchor')
  if (el) el.scrollIntoView({ behavior: 'smooth', block: 'start' })
}
```

```
const handleBookNow = () => {
  if (!user) {
    navigate('/login')
  } else {
    scrollToFeaturedAnchor()
  }
}
```

```
const applySearch = async (opts={}) => {
  setLoading(true); setError('')
  try{
```

```

const qV = ('q' in opts) ? opts.q : q
const categoryV = ('category' in opts) ? opts.category : category
const priceMaxV = ('priceMax' in opts) ? opts.priceMax : priceMax
const params = { active: 1, q: qV }
if (categoryV) params.q = `${qV} ${categoryV}`
if (priceMaxV) params.sort = 'price', params.order = 'ASC'
const res = await listServices(params)
let rows = res.data || []
if (priceMaxV) rows = rows.filter(s => Number(s.price) <= Number(priceMaxV))
if (categoryV) rows = rows.filter(s =>
(s.category || '').toLowerCase().includes(categoryV.toLowerCase()))
  setServices(rows)
  setVisibleCount(12)
} catch (e) { setError('Failed to search services') }
finally { setLoading(false) }
}

useEffect(() => { if (error) setToast({ type: 'error', msg: error }) }, [error])
useEffect(() => { if (successMsg) setToast({ type: 'success', msg: successMsg }) }, [successMsg])

return (
  <>
    { /* Hero Section (full-bleed) */ }
    <section className="hero-bleed mb-4">
      <div className="container">
        <div className="hero-landing">
          <div className="hero-left col-md-6">
            <h1 className="display-5">GlowUpNow</h1>

```

```
<p className="fs-5">Professional at-home salon & spa services. Book trusted stylists and therapists for a pampering experience without leaving your doorstep.</p>
```

```
<div className="d-flex gap-2 mt-3 cta-group">
```

```
<button className="btn btn-primary btn-lg" onClick={handleBookNow}>Book Now</button>
```

```
<button className="btn btn-outline-primary btn-lg" onClick={()=>{ setShowFeatured(false); applySearch({ q:', priceMax:', category:' }); scrollToFeaturedAnchor() }}>Explore Services</button>
```

```
</div>
```

```
</div>
```

```
<div className="hero-right col-md-6">
```

```
<div className="hero-media" role="img" aria-label="Home salon service scene" />
```

```
</div>
```

```
</div>
```

```
</div>
```

```
</section>
```

```
<div className="container py-4">
```

```
<h5 className="mb-2">Browse by Category</h5>
```

```
<hr className="section-divider mb-3" />
```

```
<div className="row row-cols-2 row-cols-sm-3 row-cols-md-4 row-cols-lg-7 g-2 mb-4">
```

```
{['Massage','Facial','Hair','Hand and Foot','Waxing','Eyelash','Eyebrow'].map(label => (
```

```
<div key={label} className="col">
```

```
<button className="btn btn-outline-primary rounded-pill w-100" onClick={()=>{ setCategory(label); setShowFeatured(false); applySearch({ category: label }); scrollToFeaturedAnchor() }}>{label}</button>
```

```
</div>
```

```
)))
```

```
</div>
```

```
{/* Trust strip */}
```

```
<div className="trust-strip d-flex align-items-center justify-content-between mb-4">
```

```
<span className="brand">TRUSTED BY CLIENTS IN METRO</span>
```

```
<div className="d-flex gap-3 small">
```

```
<span className="brand">SAFE & HYGIENIC</span>
```

```
<span className="brand">SECURE PAYMENTS</span>
```

```
<span className="brand">VERIFIED PROS</span>
```

```
</div>
```

```
</div>
```

```
{/* Value props */}
```

```
<div className="row g-3 mb-4">
```

```
<div className="col-md-4"><div className="card h-100"><div className="card-body">
```

```
<h5 className="card-title">Home Service Convenience</h5>
```

```
<p className="card-text small text-muted">Professional treatments delivered to your doorstep.  
No traffic, no queues.</p>
```

```
</div></div></div>
```

```
<div className="col-md-4"><div className="card h-100"><div className="card-body">
```

```
<h5 className="card-title">Curated Packages</h5>
```

```
<p className="card-text small text-muted">Save with bundles like Relax & Glow or Full Pamper  
— perfect for self-care days.</p>
```

```
</div></div></div>
```

```
<div className="col-md-4"><div className="card h-100"><div className="card-body">
```

```
<h5 className="card-title">Trusted & Secure</h5>
```

```
<p className="card-text small text-muted">Secure authentication, protected data, and reliable  
booking experience.</p>
```

```
</div></div></div>
```

```
</div>
```

```
<div className="card p-3 mb-3" style={{ position:'sticky', top:8, zIndex:10 }}>
```

```
<div className="d-flex flex-wrap gap-2 align-items-end">
```

```

<div style={{flex:'1 1 320px', minWidth:240}}>

  <input className="form-control" placeholder="Search services..." value={q}
onChange={e=>setQ(e.target.value)} onKeyDown={e=>{if(e.key==='Enter'){e.preventDefault();
setShowFeatured(false); applySearch(); scrollToFeaturedAnchor()}}} />

</div>

<div style={{flex:'0 0 auto', minWidth:160, maxWidth:220}}>

  <select className="form-select" value={priceMax} onChange={e=>{ const v=e.target.value;
setPriceMax(v); setShowFeatured(false); applySearch({ priceMax: v }); scrollToFeaturedAnchor() }}>

    <option value="">Price filter</option>

    <option value="500">Up to ₱500</option>

    <option value="1000">Up to ₱1,000</option>

    <option value="2000">Up to ₱2,000</option>

  </select>

</div>

<div style={{flex:'0 0 auto', minWidth:180, maxWidth:260}}>

  <select className="form-select" value={category} onChange={e=>{ const v=e.target.value;
setCategory(v); setShowFeatured(false); applySearch({ category: v }); scrollToFeaturedAnchor() }}>

    <option value="">All categories</option>

    <option value="Massage">Massage</option>

    <option value="Facial">Facial</option>

    <option value="Hair">Hair</option>

    <option value="Hand and Foot">Hand and Foot</option>

    <option value="Waxing">Waxing</option>

    <option value="Eyelash">Eyelash</option>

    <option value="Eyebrow">Eyebrow</option>

  </select>

</div>

<div className="ms-auto d-flex gap-2" style={{flex:'0 0 auto'}}>

  <button className="btn btn-outline-secondary" onClick={()=>{ setShowFeatured(false);
applySearch(); scrollToFeaturedAnchor() }}>Search</button>

```

```
        <button className="btn btn-outline-primary" onClick={()=>{ setQ(""); setPriceMax("");
setCategory(""); setVisibleCount(12); setShowFeatured(true); fetchAll(); scrollToFeaturedAnchor()
}}>Clear</button>
```

```
    </div>
```

```
</div>
```

```
</div>
```

```
{loading && <div className="alert alert-info">Loading...</div>}
```

```
{/* Anchor for scrolling */}
```

```
<div id="featuredAnchor" />
```

```
{showFeatured && featured.length>0 && (
```

```
  <div className="section-featured mb-3">
```

```
    <div className="d-flex align-items-end justify-content-between mb-2">
```

```
      <h5 id="featured" className="mb-0">Featured</h5>
```

```
      <div className="text-muted small">Top {featured.length}</div>
```

```
    </div>
```

```
    <div className="row g-3">
```

```
      {featured.map(s => (
```

```
        <div key={s.id} className="col-md-3">
```

```
          <ServiceCard service={s} onSelect={user ? setSelected : null} />
```

```
        </div>
```

```
      )})
```

```
    </div>
```

```
</div>
```

```
)}
```

```
<hr className="section-divider" />
```



```

<div className="d-flex align-items-end justify-content-between mb-2">
  <h5 className="mb-0">All Services</h5>
  <div className="text-muted small">{services.length} results</div>
</div>

```

```

{services.length===0 ? (
  <div className="text-muted">No results matched your filters.</div>
) : (
  <div className="row g-3">
    {services.slice(0, visibleCount).map(s => (
      <div key={s.id} className="col-md-4">
        <ServiceCard service={s} onSelect={user ? setSelected : null} />
      </div>
    ))}
  </div>
)}

```

```

{visibleCount < services.length && (
  <div className="d-flex justify-content-center mt-3">
    <button className="btn btn-outline-secondary" onClick={()=> setVisibleCount(v=>
Math.min(services.length, v+12))}>Load more</button>
  </div>
)}

```

```

{selected && user && (
  <BookingForm
    service={selected}
    onSuccess={() => {
      setSelected(null)
    }}
  />
)}

```

```
        setSuccessMsg('Booking created successfully!')
        setTimeout(()=>setSuccessMsg(''), 3000)
    }}
/>
}}
```

```
{/* Testimonials */}
<hr className="section-divider my-4" />
<h5 className="mb-2">What clients say</h5>
<div className="row g-3 mb-4">
    <div className="col-md-4">
        <div className="card testimonial-card h-100"><div className="card-body">
            <div className="small text-muted mb-1">Aira, Makati</div>
            <div className="stars mb-2">★★★★★</div>
            <div className="quote small">“Loved the convenience and the therapist was very
professional. Will book again!”</div>
        </div></div>
    </div>
    <div className="col-md-4">
        <div className="card testimonial-card h-100"><div className="card-body">
            <div className="small text-muted mb-1">Janelle, QC</div>
            <div className="stars mb-2">★★★★★</div>
            <div className="quote small">“Great service! The facial package was exactly what I needed
after a long week.”</div>
        </div></div>
    </div>
    <div className="col-md-4">
        <div className="card testimonial-card h-100"><div className="card-body">
            <div className="small text-muted mb-1">Mico, Taguig</div>
```

```

        <div className="stars mb-2">★★★★★</div>

        <div className="quote small">“Booked a couples massage — punctual, clean, and super
relaxing.”</div>

    </div></div>

</div>

</div>

{ /* Toasts */ }

<div style={{position:'fixed', right:16, bottom:16, zIndex:9999}}>

    {toast.msg && (

        <div className={`toast align-items-center text-white ${toast.type==='error'?'bg-danger':'bg-
success'} show`} role="alert">

            <div className="d-flex">

                <div className="toast-body">{toast.msg}</div>

                <button type="button" className="btn-close btn-close-white me-2 m-auto"
onClick={()=>setToast({type:'',msg:''})}></button>

            </div>

        </div>

    )}

</div>

{ /* Footer removed (handled by global footer) */ }

</div>

</>

)

}

```

frontend/src/pages/Login.jsx

```
import React, { useState } from 'react'

import { useNavigate, Navigate } from 'react-router-dom'

import { useAuthContext } from '../contexts/AuthContext'
```

```
export default function Login(){

  const { login, register, user, loading } = useAuthContext()

  const nav = useNavigate()

  const [email,setEmail] = useState("")
  const [password,setPassword] = useState("")
  const [name,setName] = useState("")
  const [mode, setMode] = useState('login')
  const [error,setError] = useState("")
  const [submitting,setSubmitting] = useState(false)


  const onSubmit = async (e)=>{
    e.preventDefault()
    setSubmitting(true)
    setError("")
    try{
      if(mode==='login'){
        await login(email,password)
      }else{
        await register(name,email,password)
      }
      nav('/')
    }catch(err){
      if (err.status === 422 && err.data?.errors) {
        const messages = Object.values(err.data.errors).filter(Boolean)
```

```

      setError(messages.length ? messages.join(' ') : 'Invalid input')
    } else if (err.data?.error) {
      setError(err.data.error)
    } else if (err.message) {
      setError(err.message)
    } else {
      setError('Authentication failed')
    }
  } finally {setSubmitting(false)}
}

```

```

if (loading) return (<div className="container py-4"><div className="alert alert-info">Loading...</div></div>)

```

```

if (user) return (<Navigate to={user.role === 'admin' ? '/admin/dashboard' : '/dashboard'} replace />)

```

```

return (

```

```

  <div className="container-fluid auth-wrap d-flex align-items-center py-4 py-md-5">

```

```

    <div className="container">

```

```

      <div className="row align-items-center g-4">

```

```

        <div className="col-lg-6 d-none d-lg-block">

```

```

          <div className="auth-left">

```

```

            <div className="mb-3">

```

```

              <div className="fs-4 fw-bold">GlowUpNow</div>

```

```

              <div className="small" style={{opacity:.9}}>Home salon & spa services, on demand</div>

```

```

            </div>

```

```

            <h1 className="display-6 mb-3">{mode==='login' ? 'Welcome back' : 'Join GlowUpNow'}</h1>

```

```

            <p className="mb-4">{mode==='login' ? 'Book massages, facials, nails, and hair services with trusted pros at your home. Sign in to continue.' : 'Create your account to book at-home salon & spa services, save preferences, and track your bookings.'}</p>

```

```

<ul className="list-unstyled m-0 p-0 d-grid gap-2">
  {mode==='login' ? (
    <>
      <li className="d-flex align-items-center gap-2"><span className="badge rounded-pill bg-light text-dark">Secure</span><span>Protected authentication</span></li>
      <li className="d-flex align-items-center gap-2"><span className="badge rounded-pill bg-light text-dark">Fast</span><span>Quick, seamless booking</span></li>
      <li className="d-flex align-items-center gap-2"><span className="badge rounded-pill bg-light text-dark">Trusted</span><span>Verified professionals</span></li>
    </>
  ) : (
    <>
      <li className="d-flex align-items-center gap-2"><span className="badge rounded-pill bg-light text-dark">Personal</span><span>Set your preferences</span></li>
      <li className="d-flex align-items-center gap-2"><span className="badge rounded-pill bg-light text-dark">Tracking</span><span>Manage bookings & history</span></li>
      <li className="d-flex align-items-center gap-2"><span className="badge rounded-pill bg-light text-dark">Perks</span><span>Access exclusive offers</span></li>
    </>
  )}
</ul>
</div>
</div>
<div className="col-lg-5 ms-lg-auto">
  <div className="card auth-card p-4 p-md-5">
    <div className="d-flex justify-content-between align-items-center mb-2">
      <h2 className="h3 m-0">{mode==='login' ? 'Sign in' : 'Create account'}</h2>
      <button type="button" className="btn btn-outline-secondary btn-sm"
onClick={()=>setMode(mode==='login'? 'register': 'login')}>
        {mode==='login' ? 'Register' : 'Login'}
      </button>
    </div>
  </div>
</div>

```

```

</div>

{error && <div className="alert alert-danger">{error}</div>}

<form onSubmit={onSubmit}>

  {mode==='register' && (

    <div className="mb-3">

      <label className="form-label">Name</label>

      <input className="form-control" placeholder="Your full name" value={name}
onChange={e=>setName(e.target.value)} onFocus={()=>{}} onBlur={()=>{}} />

    </div>

  )}

  <div className="mb-3">

    <label className="form-label">Email</label>

    <input type="email" className="form-control" placeholder="you@example.com"
value={email} onChange={e=>setEmail(e.target.value)} onFocus={()=>{}} onBlur={()=>{}} />

  </div>

  <div className="mb-3">

    <label className="form-label">Password</label>

    <input type="password" className="form-control" placeholder={mode==='login' ? 'Your
password' : 'Create a password (min 8 characters)'} value={password}
onChange={e=>setPassword(e.target.value)} />

  </div>

  <div className="d-grid gap-2">

    <button disabled={submitting} className="btn btn-primary" type="submit">{submitting ?
'Please wait...' : (mode==='login' ? 'Sign in' : 'Create account')}</button>

    <button type="button" className="btn btn-outline-primary"
onClick={()=>setMode(mode==='login'? 'register': 'login')}>

      {mode==='login' ? 'New here? Create an account' : 'Have an account? Sign in'}

    </button>

  </div>

  {mode==='register' && (

```

<div className="text-muted small mt-2">By creating an account, you agree to our Terms and Privacy Policy.</div>

}}

</form>

</div>

</div>

</div>

</div>

</div>

)

}

frontend/src/pages/ServiceManagement.jsx

// ServiceManagement.jsx — Admin: manage services (CRUD), filters, pagination, image upload.

```
import React, { useEffect, useState, useRef, useMemo } from 'react'
```

```
import useClickOutside from '../hooks/useClickOutside'
```

```
import useDebounce from '../hooks/useDebounce'
```

```
import useModal from '../hooks/useModal'
```

```
import { listServices, createService, updateService, deleteService, createServiceForm } from  
'../services/serviceService'
```

```
export default function ServiceManagement(){
```

```
  const empty = { id: 0, name: "", category:"", description: "", price: "", duration_minutes: "", image_path:"",  
  active: 1 }
```

```
  const [items,setItems] = useState([])
```

```
  const [form,setForm] = useState(empty)
```

```
  const [loading,setLoading] = useState(false)
```

```
  const [error,setError] = useState("")
```

```
  const [success,setSuccess] = useState("")
```

```
  const [q,setQ] = useState("")
```

```
  const [imageFile, setImageFile] = useState(null)
```

```
  // Filters
```

```
  const [fCategory, setFCategory] = useState("")
```

```
  const [fPriceMin, setFPriceMin] = useState("")
```

```
  const [fPriceMax, setFPriceMax] = useState("")
```

```
  const [fDurMin, setFDurMin] = useState("")
```

```
  const [fDurMax, setFDurMax] = useState("")
```

```
  const [fActive, setFActive] = useState("")
```

```
  // Pagination
```

```
  const [page, setPage] = useState(1)
```

```
  const [perPage, setPerPage] = useState(10)
```

```

const formRef = useRef(null)

// Delete modal state
const [toDelete, setToDelete] = useState(null)

const { open: confirmOpen, openModal, closeModal, onKeyDown, ref: delRef } = useModal(false)

// Toasts
const [toast, setToast] = useState({ type:"", msg:" })

const [nameTouched, setNameTouched] = useState(false)

const [priceTouched, setPriceTouched] = useState(false)

const [durTouched, setDurTouched] = useState(false)

const [qFocus, setQFocus] = useState(false)

useClickOutside(delRef, ()=>{ if (confirmOpen && !loading) closeDelete() })

const load = async()=>{
  setLoading(true); setError("")
  try{
    const res = await listServices({ q, active: " " })
    setItems(res.data || [])
  }catch(e){ setError('Failed to load services') }
  finally{ setLoading(false) }
}

useEffect(()=>{ load() },[])

const submit = async (e)=>{
  e.preventDefault()
  setLoading(true); setError(""); setSuccess("")
  try{

```

```

    const payload = { ...form, price: Number(form.price), duration_minutes:
Number(form.duration_minutes) }

    if (imageFile) {

        const fd = new FormData()

        Object.entries(payload).forEach(([k,v])=>{ if(v!==undefined && v!==null && v!=="") fd.append(k,
String(v)) })

        if (form.id) fd.append('id', String(form.id))

        fd.append('image', imageFile)

        await createServiceForm(fd)

    } else {

        if(form.id){ await updateService(payload) }

        else { await createService(payload) }

    }

    setForm(empty)

    setImageFile(null)

    setSuccess('Saved')

    load()

} catch(err){ setError(err.data?.error || err.data?.errors ? JSON.stringify(err.data.errors) : 'Save
failed') }

finally{ setLoading(false) }

}

```

```

const edit = (s)=> {

    setForm({ ...s })

    setImageFile(null)

    // scroll to form for editing

    setTimeout(()=>{

        if (formRef.current) {

            formRef.current.scrollToView({ behavior: 'smooth', block: 'start' })

        }

    })

```

```

    }, 0)
  }

  const doRemove = async(id)=>{
    setLoading(true); setError(""); setSuccess("")

    try{ await deleteService(id); setSuccess('Deleted'); load() }catch(e){ setError('Delete failed') } finally{
    setLoading(false) }

  }

  const openDelete = (s)=>{ setToDelete(s); openModal() }

  const closeDelete = ()=>{ closeModal(); setToDelete(null) }

  const confirmDelete = async()=>{
    if(!toDelete) return;

    const id = toDelete.id

    closeDelete()

    await doRemove(id)
  }

  // Derived filtering + pagination

  const dq = useDebounce(q, 300)

  const filteredItems = useMemo(()=> items.filter(s => {
    if (dq && !`${s.name} ${s.category}
    ${s.description} | |`.toLowerCase().includes(dq.toLowerCase())) return false

    if (fCategory && s.category !== fCategory) return false

    const price = Number(s.price)

    if (fPriceMin !== "" && price < Number(fPriceMin)) return false

    if (fPriceMax !== "" && price > Number(fPriceMax)) return false

    const dur = Number(s.duration_minutes)

    if (fDurMin !== "" && dur < Number(fDurMin)) return false

    if (fDurMax !== "" && dur > Number(fDurMax)) return false

    if (fActive !== "" && Number(s.active) !== Number(fActive)) return false
  }

```

```

    return true
  }}, [items, dq, fCategory, fPriceMin, fPriceMax, fDurMin, fDurMax, fActive])

  const totalPages = Math.max(1, Math.ceil(filteredItems.length / perPage))

  const safePage = Math.min(page, totalPages)

  const startIdx = (safePage - 1) * perPage

  const pageltems = useMemo(()=> filteredItems.slice(startIdx, startIdx + perPage), [filteredItems,
startIdx, perPage])

  const clearFilters = ()=>{

    setQ(""); setFCategory(""); setFPriceMin(""); setFPriceMax(""); setFDurMin(""); setFDurMax("");
setFActive(""); setPage(1)

    load()
  }

  const onInteract = ()=>{ setSuccess(""); setError(""); setToast({type:",msg:"}) }

  const onEnter = (e)=>{ if(e.key==='Enter'){ e.preventDefault(); setPage(1); onInteract() } }

  useEffect(()=>{

    if (error) setToast({ type:'error', msg:error })

  }, [error])

  useEffect(()=>{

    if (success) setToast({ type:'success', msg:success })

  }, [success])

  return (

    <div className="container py-4">

      <h2 className="mb-3">Service Management</h2>

      <div ref={formRef} className="card p-3 mb-3">

        <form onSubmit={submit} className="row g-3">

```

```

<div className="col-md-4">

  <label className="form-label">Name</label>

  <input className={`form-control ${nameTouched && !form.name ? 'is-invalid':''}`}
value={form.name} onChange={e=>setForm(f=>({...f,name:e.target.value}))}
onBlur={()=>setNameTouched(true)} onFocus={()=>setNameTouched(false)} required />

  {nameTouched && !form.name ? (<div className="invalid-feedback">Name is required</div>)
: null}

</div>

<div className="col-md-4">

  <label className="form-label">Category</label>

  <select className="form-select" value={form.category}
onChange={e=>setForm(f=>({...f,category:e.target.value}))}>

    <option value="">Select category</option>

    <option value="Massage">Massage</option>

    <option value="Facial">Facial</option>

    <option value="Hair">Hair</option>

    <option value="Hand and Foot">Hand and Foot</option>

    <option value="Waxing">Waxing</option>

    <option value="Eyelash">Eyelash</option>

    <option value="Eyebrow">Eyebrow</option>

  </select>

</div>

<div className="col-md-4">

  <label className="form-label">Price</label>

  <input type="number" step="0.01" className={`form-control ${priceTouched && (!form.price
|| Number(form.price)<=0) ? 'is-invalid':''}`} value={form.price}
onChange={e=>setForm(f=>({...f,price:e.target.value}))} onBlur={()=>setPriceTouched(true)}
onFocus={()=>setPriceTouched(false)} required />

  {priceTouched && (!form.price || Number(form.price)<=0) ? (<div className="invalid-
feedback">Enter a positive price</div>) : null}

</div>

<div className="col-md-4">

```

```

<label className="form-label">Duration (mins)</label>

<input type="number" className={`form-control ${durTouched && (!form.duration_minutes
|| Number(form.duration_minutes)<=0) ? 'is-invalid':''}`} value={form.duration_minutes}
onChange={e=>setForm(f=>({...f,duration_minutes:e.target.value}))}
onBlur={()=>setDurTouched(true)} onFocus={()=>setDurTouched(false)} required />

{durTouched && (!form.duration_minutes || Number(form.duration_minutes)<=0) ? (<div
className="invalid-feedback">Enter a positive duration</div>) : null}

</div>

<div className="col-md-4">

<label className="form-label">Image</label>

<input type="file" accept="image/*" className="form-control"
onChange={e=>setImageFile((e.target.files && e.target.files[0]) || null)} />

{form.id && form.image_path ? (<div className="form-text">Current:
{form.image_path}</div>) : null}

</div>

<div className="col-md-4">

<label className="form-label">Active</label>

<select className="form-select" value={form.active}
onChange={e=>setForm(f=>({...f,active:Number(e.target.value)}))}>

  <option value={1}>Yes</option>

  <option value={0}>No</option>

</select>

</div>

<div className="col-12">

<label className="form-label">Description</label>

<textarea className="form-control" rows="2" value={form.description}
onChange={e=>setForm(f=>({...f,description:e.target.value}))} />

</div>

<div className="col-12 text-end">

<button className="btn btn-primary" disabled={loading}
type="submit">{form.id?'Update':'Create'} Service</button>

```

```
    {form.id ? <button type="button" className="btn btn-outline-secondary ms-2" onClick={()=>{
setForm(empty); setImageFile(null) }}>Cancel</button> : null}
```

```
</div>
```

```
</form>
```

```
</div>
```

```
<div className="card p-3 mb-3" style={{ position:'sticky', top:8, zIndex:10 }}>
```

```
<div className="row g-3 align-items-end">
```

```
<div className="col-12 col-md-6 col-lg-3">
```

```
<label className="form-label">Search</label>
```

```
<input className="form-control" placeholder="Search services" value={q} onChange={e=>{
setQ(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter}
onFocus={()=>setQFocus(true)} onBlur={()=>setQFocus(false)} style={qFocus?{ boxShadow:'0 0 0 .2rem
rgba(111,66,193,.25)', borderColor:'#6f42c1' }:undefined} />
```

```
</div>
```

```
<div className="col-6 col-md-6 col-lg-3">
```

```
<label className="form-label">Category</label>
```

```
<select className="form-select" value={fCategory} onChange={e=>{
setFCategory(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter}>
```

```
<option value="">All</option>
```

```
<option value="Massage">Massage</option>
```

```
<option value="Facial">Facial</option>
```

```
<option value="Hair">Hair</option>
```

```
<option value="Hand and Foot">Hand and Foot</option>
```

```
<option value="Waxing">Waxing</option>
```

```
<option value="Eyelash">Eyelash</option>
```

```
<option value="Eyebrow">Eyebrow</option>
```

```
</select>
```

```
</div>
```

```
<div className="col-6 col-md-6 col-lg-3">
```



```

<label className="form-label">Active</label>

<select className="form-select" value={fActive} onChange={e=>{ setFActive(e.target.value);
setPage(1); onInteract() }} onKeyDown={onEnter} onKeyPress={onEnter}>

  <option value="">All</option>

  <option value="1">Yes</option>

  <option value="0">No</option>

</select>

</div>

<div className="col-6 col-md-6 col-lg-3">

  <label className="form-label">Per page</label>

  <select className="form-select" value={perPage} onChange={e=>{
setPerPage(Number(e.target.value)); setPage(1); onInteract() }} onKeyDown={onEnter}
onKeyPress={onEnter}>

    <option value={5}>5</option>

    <option value={10}>10</option>

    <option value={20}>20</option>

    <option value={50}>50</option>

  </select>

</div>

<div className="col-md-6 col-lg-4">

  <label className="form-label">Price</label>

  <div className="d-flex gap-2">

    <input type="number" step="0.01" className="form-control" placeholder="Min"
value={fPriceMin} onChange={e=>{ setFPriceMin(e.target.value); setPage(1); onInteract() }}
onKeyDown={onEnter} onKeyPress={onEnter} />

    <input type="number" step="0.01" className="form-control" placeholder="Max"
value={fPriceMax} onChange={e=>{ setFPriceMax(e.target.value); setPage(1); onInteract() }}
onKeyDown={onEnter} onKeyPress={onEnter} />

  </div>

</div>

<div className="col-md-6 col-lg-4">

```

```

<label className="form-label">Duration (mins)</label>

<div className="d-flex gap-2">

    <input type="number" className="form-control" placeholder="Min" value={fDurMin}
onChange={e=>{ setFDurMin(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter}
onKeyPress={onEnter} />

    <input type="number" className="form-control" placeholder="Max" value={fDurMax}
onChange={e=>{ setFDurMax(e.target.value); setPage(1); onInteract() }} onKeyDown={onEnter}
onKeyPress={onEnter} />

</div>

</div>

<div className="col-md-12 col-lg-4 d-flex justify-content-lg-end align-items-end mt-2 mt-lg-0">

    <button className="btn btn-outline-primary" type="button" onClick={clearFilters}>Clear
search/filters</button>

</div>

</div>

</div>

{/* Toasts */}

<div style={{position:'fixed', right:16, bottom:16, zIndex:9999}}>

    {toast.msg && (

        <div className={`toast align-items-center text-white ${toast.type==='error'? 'bg-danger': 'bg-
success'} show`} role="alert">

            <div className="d-flex">

                <div className="toast-body">{toast.msg}</div>

                <button type="button" className="btn-close btn-close-white me-2 m-auto"
onClick={()=>setToast({type:"",msg:"})}></button>

            </div>

        </div>

    )}

</div>

```

```

{filteredItems.length === 0 ? (
  <div className="text-muted">No results matched your filters.</div>
) : (
  <div className="table-responsive">
    <table className="table table-striped table-hover">
      <thead>
        <tr>
          <th>#</th><th>Name</th><th>Category</th><th>Price</th><th>Duration</th><th>Active</th><th></th></tr>
        </tr>
      </thead>
      <tbody>
        {pageItems.map(s => (
          <tr key={s.id} onDoubleClick={()=>edit(s)}>
            <td>{s.id}</td>
            <td>{s.name}</td>
            <td>{s.category || '-'}</td>
            <td>₹{Number(s.price).toFixed(2)}</td>
            <td>{s.duration_minutes} mins</td>
            <td>{s.active? 'Yes':'No'}</td>
            <td className="text-end">
              <button className="btn btn-sm btn-outline-primary me-2" onClick={()=>edit(s)}
disabled={loading}>Edit</button>
              <button className="btn btn-sm btn-outline-danger" onClick={()=>openDelete(s)}
disabled={loading}>Delete</button>
            </td>
          </tr>
        ))}
      </tbody>
    </table>

```

```

    </div>

  })

  <div className="d-flex justify-content-between align-items-center mt-2">

    <div className="small text-muted">Showing {filteredItems.length === 0 ? 0 : (startIndex+1)}–
    {Math.min(startIndex+perPage, filteredItems.length)} of {filteredItems.length}</div>

    <div className="btn-group" role="group" aria-label="Pagination">

      <button type="button" className="btn btn-outline-secondary btn-sm" disabled={safePage<=1}
      onClick={()=>{ onInteract(); setPage(p=>Math.max(1, p-1))}}>Prev</button>

      <span className="btn btn-outline-secondary btn-sm disabled">Page {safePage} /
      {totalPages}</span>

      <button type="button" className="btn btn-outline-secondary btn-sm"
      disabled={safePage>=totalPages} onClick={()=>{ onInteract(); setPage(p=>Math.min(totalPages,
      p+1))}}>Next</button>

    </div>

  </div>

  {confirmOpen && (

    <>

      <div className="modal fade show d-block" tabIndex="-1" role="dialog" aria-modal="true"
      onKeyDown={onKeyDown}>

        <div className="modal-dialog">

          <div className="modal-content" ref={delRef}>

            <div className="modal-header">

              <h5 className="modal-title">Delete service</h5>

              <button type="button" className="btn-close" aria-label="Close"
              onClick={closeDelete}></button>

            </div>

            <div className="modal-body">

              <p>Are you sure you want to delete "{toDelete?.name}"?</p>

            </div>

            <div className="modal-footer">

```

```
        <button type="button" className="btn btn-outline-secondary" onClick={closeDelete}  
disabled={loading}>Cancel</button>
```

```
        <button type="button" className="btn btn-danger" onClick={confirmDelete}  
disabled={loading}>Delete</button>
```

```
    </div>
```

```
  </div>
```

```
</div>
```

```
</div>
```

```
<div className="modal-backdrop fade show"></div>
```

```
</>
```

```
  })
```

```
</div>
```

```
)
```

```
}
```

frontend/src/pages/UserDashboard.jsx

import React, { useEffect, useState } from 'react'

import { listServices } from '../services/serviceService'

import ServiceCard from '../components/ServiceCard'

import { useNavigate } from 'react-router-dom'

export default function UserDashboard(){

const navigate = useNavigate()

const [services, setServices] = useState([])

const [featured, setFeatured] = useState([])

const [showFeatured, setShowFeatured] = useState(true)

const [visibleCount, setVisibleCount] = useState(12)

const [q, setQ] = useState("")

const [priceMax, setPriceMax] = useState("")

const [category, setCategory] = useState("")

const [loading, setLoading] = useState(false)

const [error, setError] = useState("")

const [toast, setToast] = useState({ type:"", msg:"" })

useEffect(()=>{

const loadServices = async ()=>{

setLoading(true); setError("")

try{

const res = await listServices({ active:1, sort:'created_at', order:'DESC' })

const rows = res.data || []

setServices(rows)

setFeatured(rows.slice(0,6))

}catch{ setError('Failed to load services') }

finally{ setLoading(false) }

```
}
```

```
loadServices()
```

```
},[])
```

```
const handleSelectForBooking = (svc) => {  
  navigate('/bookings', { state: { selected: svc } })  
}
```

```
const scrollToAnchor = ()=>{  
  const el = document.getElementById('ud-featuredAnchor')  
  if (el) el.scrollIntoView({ behavior: 'smooth', block: 'start' })  
}
```

```
const fetchAll = async ()=>{  
  setLoading(true); setError("")  
  try{  
    const res = await listServices({ active:1, sort:'created_at', order:'DESC' })  
    const rows = res.data || []  
    setServices(rows)  
    setFeatured(rows.slice(0,6))  
  }catch{ setError('Failed to load services') }  
  finally{ setLoading(false) }  
}
```

```
const applySearch = async (opts={})=>{  
  setLoading(true); setError("")  
  try{  
    const qV = ('q' in opts) ? opts.q : q  
    const priceV = ('priceMax' in opts) ? opts.priceMax : priceMax
```

```

const catV = ('category' in opts) ? opts.category : category

const params = { active:1, q: qV }

if (priceV) params.sort='price', params.order='ASC'

const res = await listServices(params)

let rows = res.data || []

if (priceV) rows = rows.filter(s => Number(s.price) <= Number(priceV))

if (catV) rows = rows.filter(s => (s.category || '').toLowerCase().includes(catV.toLowerCase()))

setServices(rows)

setVisibleCount(12)

}catch{ setError('Search failed') }

finally{ setLoading(false) }

}

useEffect(()=>{ if (error) setToast({ type:'error', msg:error }) }, [error])

return (

  <div className="container py-4">

    <div className="d-flex justify-content-between align-items-center mb-3" style={{ position:'sticky',
top:8, zIndex:10 }}>

      <h3 className="mb-0">Discover Services</h3>

      <div className="d-flex gap-2">

        <input className="form-control" placeholder="Search services" value={q}
onChange={e=>setQ(e.target.value)} onKeyDown={e=>{ if(e.key==='Enter'){ e.preventDefault();
setShowFeatured(false); applySearch(); scrollToAnchor() } }} />

        <select className="form-select" value={priceMax} onChange={e=>{ const v=e.target.value;
setPriceMax(v); setShowFeatured(false); applySearch({ priceMax: v }); scrollToAnchor() }}>

          <option value="">Price</option>

          <option value="500">≤ ₱500</option>

          <option value="1000">≤ ₱1,000</option>

          <option value="2000">≤ ₱2,000</option>

```



```

</select>

<select className="form-select" value={category} onChange={e=>{ const v=e.target.value;
setCategory(v); setShowFeatured(false); applySearch({ category: v }); scrollToAnchor() }}>

  <option value="">All categories</option>

  <option value="Massage">Massage</option>

  <option value="Hair">Hair</option>

  <option value="Nails">Nails</option>

  <option value="Skin">Skin</option>

  <option value="Package">Package</option>

</select>

<button className="btn btn-outline-secondary" onClick={()=>{ setShowFeatured(false);
applySearch(); scrollToAnchor() }}>Search</button>

<button className="btn btn-outline-primary" onClick={()=>{ setQ(""); setPriceMax("");
setCategory(""); setVisibleCount(12); setShowFeatured(true); fetchAll(); scrollToAnchor()
}}>Clear</button>

```

```

</div>

```

```

</div>

```

```

{loading && <div className="text-muted small mb-2">Loading...</div>}

```

```

{/* Anchor for scrolling */}

```

```

<div id="ud-featuredAnchor" />

```

```

{showFeatured && featured.length>0 && (

```

```

  <div className="section-featured mb-3">

```

```

    <div className="d-flex align-items-end justify-content-between mb-2">

```

```

      <h5 className="mb-0">Featured</h5>

```

```

      <div className="text-muted small">Top {featured.length}</div>

```

```

    </div>

```

```

  <div className="row g-3">

```

```

    {featured.map(s => (
      <div key={s.id} className="col-md-3">
        <ServiceCard service={s} onSelect={handleSelectForBooking} />
      </div>
    ))}
  </div>
</div>
))

<hr className="section-divider" />
<div className="d-flex align-items-end justify-content-between mb-2">
  <h5 className="mb-0">All Services</h5>
  <div className="text-muted small">{services.length} results</div>
</div>

{services.length===0 ? (
  <div className="text-muted">No results matched your filters.</div>
) : (
  <>
    <div className="row g-3">
      {services.slice(0, visibleCount).map(s => (
        <div key={s.id} className="col-md-4">
          <ServiceCard service={s} onSelect={handleSelectForBooking} />
        </div>
      ))}
    </div>
    {visibleCount < services.length && (
      <div className="d-flex justify-content-center mt-3">

```

```

        <button className="btn btn-outline-secondary" onClick={() => setVisibleCount(v =>
Math.min(services.length, v+12))}>Load more</button>

        </div>

    }}

</>

)}

{/* Toasts */}

<div style={{position:'fixed', right:16, bottom:16, zIndex:9999}}>

    {toast.msg && (

        <div className={`toast align-items-center text-white ${toast.type==='error'? 'bg-danger': 'bg-
success'} show`} role="alert">

            <div className="d-flex">

                <div className="toast-body">{toast.msg}</div>

                <button type="button" className="btn-close btn-close-white me-2 m-auto"
onClick={() => setToast({type:"",msg:"})}></button>

            </div>

        </div>

    )}

</div>

</div>

)

}

```

frontend/src/pages/UserProfile.jsx

import React, { useEffect, useState } from 'react'

import { getMe, updateMe } from '../services/userService'

export default function UserProfile(){

const [form,setForm] = useState({ name:"", email:"", phone:"" })

const [loading,setLoading] = useState(false)

const [msg,setMsg] = useState("")

const [err,setErr] = useState("")

const [initial,setInitial] = useState({ name:"", phone:"" })

useEffect(()=>{

(async()=>{

try{

const res = await getMe()

if(res.user){

const next = { name: res.user.name || "", email: res.user.email || "", phone: res.user.phone || "" }

setForm(next)

setInitial({ name: next.name || "", phone: next.phone || "" })

}

}catch(e){ setErr('Failed to load profile') }

})();

},[])

const submit = async (e)=>{

e.preventDefault()

setLoading(true); setMsg(""); setErr("")

try{

const current = { name: form.name || "", phone: form.phone || "" }

```

    if (current.name === (initial.name || '') && (current.phone || '') === (initial.phone || '')) {
      setMsg('No changes')
      return
    }

    await updateMe({ name: current.name, phone: current.phone })
    setInitial(current)
    setMsg('Profile updated')
  } catch (e) { setErr('Update failed') }
  finally { setLoading(false) }
}

return (
  <div className="container py-4">
    <h2 className="mb-3">My Profile</h2>
    {msg && <div className="alert alert-success">{msg}</div>}
    {err && <div className="alert alert-danger">{err}</div>}
    <form onSubmit={submit} className="card p-3">
      <div className="row g-3">
        <div className="col-md-6">
          <label className="form-label" htmlFor="profile_name">Name</label>
          <input id="profile_name" name="name" className="form-control" value={form.name}
onChange={e=>setForm(f=>({...f,name:e.target.value}))} onFocus={()=>{}} onBlur={()=>{}} />
        </div>
        <div className="col-md-6">
          <label className="form-label" htmlFor="profile_email">Email</label>
          <input id="profile_email" name="email" type="email" className="form-control"
value={form.email} disabled />
        </div>
        <div className="col-md-6">

```

```
    <label className="form-label" htmlFor="profile_phone">Phone</label>

    <input id="profile_phone" name="phone" className="form-control" value={form.phone || ""}
onChange={e=>setForm(f=>({...f,phone:e.target.value}))} />

    </div>

  </div>

  <div className="text-end mt-3">

    <button className="btn btn-primary" disabled={loading} type="submit">Save</button>

  </div>

</form>

</div>

)

}
```

Services

frontend/src/services/analyticsService.js

// analyticsService.js — Client wrappers for analytics endpoints (summary, coverage, bookings).

```
import { httpGet } from './http'
```

```
export async function getSummary(params={}){  
  const q = new URLSearchParams(params).toString()  
  return httpGet(`/api/analytics/summary${q?`?${q}`:''}`)  
}
```

```
export async function getServiceCoverage(params={}){  
  const q = new URLSearchParams(params).toString()  
  return httpGet(`/api/analytics/services-coverage${q?`?${q}`:''}`)  
}
```

```
export async function getRightJoinDemo(params={}){  
  const q = new URLSearchParams(params).toString()  
  return httpGet(`/api/analytics/right-join-demo${q?`?${q}`:''}`)  
}
```

```
export async function getAnalyticsBookings(params={}){  
  const q = new URLSearchParams(params).toString()  
  return httpGet(`/api/analytics/bookings${q?`?${q}`:''}`)  
}
```

frontend/src/services/authService.js

```
import { httpGet, httpPost } from './http'
```

```
export async function login(email, password){
```

```
  return httpPost('/api/auth/login', { email, password })
```

```
}
```

```
export async function logout(){
```

```
  return httpPost('/api/auth/logout')
```

```
}
```

```
export async function register(name, email, password){
```

```
  return httpPost('/api/auth/register', { name, email, password })
```

```
}
```

```
export async function me(){
```

```
  return httpGet('/api/auth/me')
```

```
}
```


frontend/src/services/bookingService.js

```
import { httpGet, httpPost, httpPut, httpDelete } from './http'
```

```
export async function listBookings(params = {}){
  const q = new URLSearchParams(params).toString()
  return httpGet(`/api/bookings${q ? `?` + q : ''}`)
}

export async function createBooking(payload){
  return httpPost('/api/bookings', payload)
}

export async function updateBooking(payload){
  return httpPut('/api/bookings', payload)
}

export async function updateBookingStatus(payload){
  const { httpPatch } = await import('./http')
  return httpPatch('/api/bookings/status', payload)
}

export async function rescheduleBooking(payload){
  const { httpPatch } = await import('./http')
  return httpPatch('/api/bookings/reschedule', payload)
}

export async function deleteBooking(id){
  return httpDelete('/api/bookings', { id })
}

export async function cancelBooking(id){
  return updateBookingStatus({ id, status: 'canceled' })
}
```

frontend/src/services/http.js

// http.js — HTTP helpers for API base and JSON/form requests.

```
export const API = import.meta.env.VITE_API_BASE_URL ||  
'http://localhost:glowupnow/backend/public'
```

```
export async function httpGet(path, opts = {}) {  
  const res = await fetch(`${API}${path}`, {  
    method: 'GET',  
    credentials: 'include',  
    headers: { Accept: 'application/json', ...(opts.headers || {}) },  
  })  
  return handle(res)  
}
```

```
export async function httpPost(path, body = {}, opts = {}) {  
  const res = await fetch(`${API}${path}`, {  
    method: 'POST',  
    credentials: 'include',  
    headers: { 'Content-Type': 'application/json', ...(opts.headers || {}) },  
    body: JSON.stringify(body),  
  })  
  return handle(res)  
}
```

```
export async function httpPut(path, body = {}, opts = {}) {  
  const res = await fetch(`${API}${path}`, {  
    method: 'PUT',  
    credentials: 'include',  
    headers: { 'Content-Type': 'application/json', ...(opts.headers || {}) },  
  })  
  return handle(res)  
}
```

```
    body: JSON.stringify(body),  
  })  
  return handle(res)  
}
```

```
export async function httpPostForm(path, formData, opts = {}) {  
  const res = await fetch(`${API}${path}`, {  
    method: 'POST',  
    credentials: 'include',  
    headers: { ...(opts.headers || {} )},  
    body: formData,  
  })  
  return handle(res)  
}
```

```
export async function httpPutForm(path, formData, opts = {}) {  
  const res = await fetch(`${API}${path}`, {  
    method: 'PUT',  
    credentials: 'include',  
    headers: { ...(opts.headers || {} )},  
    body: formData,  
  })  
  return handle(res)  
}
```

```
export async function httpPatch(path, body = {}, opts = {}) {  
  const res = await fetch(`${API}${path}`, {  
    method: 'PATCH',  
    credentials: 'include',
```

```

    headers: { 'Content-Type': 'application/json', ...(opts.headers || {} ) },
    body: JSON.stringify(body),
  })
  return handle(res)
}

```

```

export async function httpDelete(path, body = null, opts = {}) {
  const hasBody = body !== null && body !== undefined
  const res = await fetch(`${API}${path}`, {
    method: 'DELETE',
    credentials: 'include',
    headers: hasBody ? { 'Content-Type': 'application/json', ...(opts.headers || {} ) } : { ...(opts.headers || {} ) },
    body: hasBody ? JSON.stringify(body) : undefined,
  })
  return handle(res)
}

```

```

async function handle(res) {
  const text = await res.text()
  let data = null
  if (text) {
    try { data = JSON.parse(text) } catch { /* non-JSON response */ }
  }
  if (!res.ok) {
    const message = (data && (data.error || data.message)) || text || 'Request failed'
    const err = new Error(message)
    err.status = res.status
    err.data = data
  }
}

```

```
    err.raw = text
    throw err
  }
  return data
}
```

frontend/src/services/serviceService.js

```
import { httpGet, httpPost, httpPut, httpDelete, httpPostForm } from './http'
```

```
export async function listServices(params = {}){  
  const q = new URLSearchParams(params).toString()  
  return httpGet(`/api/services${q ? `?${q}` : ''}`)  
}
```

```
export async function createService(payload){  
  return httpPost('/api/services', payload)  
}
```

```
export async function updateService(payload){  
  return httpPut('/api/services', payload)  
}
```

```
export async function deleteService(id){  
  return httpDelete('/api/services', { id })  
}
```

// FormData versions (also used for updates when including files)

```
export async function createServiceForm(formData){  
  // Controller will treat presence of `id` as update  
  return httpPostForm('/api/services', formData)  
}
```

frontend/src/services/userService.js

// userService.js — Authenticated user profile and admin user listing helpers.

```
import { httpGet, httpPut } from './http'
```

```
export async function getMe(){
```

```
  return httpGet('/api/auth/me')
```

```
}
```

```
export async function updateMe(payload){
```

```
  return httpPut('/api/users', payload)
```

```
}
```

```
// Admin: list all users (supports ?q=&page=&perPage=)
```

```
export async function listUsers(params){
```

```
  const qs = new URLSearchParams(params || {}).toString();
```

```
  return httpGet('/api/admin/users' + (qs ? ('?' + qs) : ''));
```

```
}
```

```
// Admin: update user (role, name, phone)
```

```
export async function updateUser(payload){
```

```
  return httpPut('/api/admin/users', payload);
```

```
}
```

```
// Admin: delete user
```

```
export async function deleteUser(id){
```

```
  return fetch('/api/admin/users?id=' + encodeURIComponent(id), {
```

```
    method: 'DELETE',
```

```
    credentials: 'include',
```

```
    headers: { 'Accept': 'application/json' } }
```

```
}).then(r=>r.json());
```

```
}
```


Backend

Controllers

backend/app/controllers/AdminUserController.php

```
<?php
```

```
// AdminUserController — Admin-only user listing with search and pagination (read-only).
```

```
namespace App\controllers;
```

```
use App\core\Controller;
```

```
use App\core\Request;
```

```
use App\core\Response;
```

```
use App\core\Database;
```

```
class AdminUserController extends Controller {
```

```
    public function index(Request $req, Response $res): void {
```

```
        $pdo = Database::pdo();
```

```
        $q = trim($req->query()['q'] ?? '');
```

```
        $page = max(1, (int)($req->query()['page'] ?? 1));
```

```
        $perPage = min(100, max(5, (int)($req->query()['perPage'] ?? ($req->query()['per_page']  
?? 20))));
```

```
        $offset = ($page - 1) * $perPage;
```

```
        $where = [];
```

```
        $params = [];
```

```
        if ($q !== ''){
```

```
            $where[] = '(name LIKE :q OR email LIKE :q OR role LIKE :q)';
```

```

        $params[':q'] = '%'.$q.'%';
    }

    $whereSql = $where ? ('WHERE '.implode(' AND ', $where)) : "";

    $stmt = $pdo->prepare("SELECT COUNT(*) FROM users $whereSql");
    $stmt->execute($params);
    $total = (int)$stmt->fetchColumn();

    $sql = "SELECT id, name, email, role, phone, created_at
            FROM users $whereSql
            ORDER BY created_at DESC, id DESC
            LIMIT :limit OFFSET :offset";

    $stmt = $pdo->prepare($sql);
    foreach ($params as $k=>$v){ $stmt->bindValue($k, $v); }
    $stmt->bindValue(':limit', $perPage, \PDO::PARAM_INT);
    $stmt->bindValue(':offset', $offset, \PDO::PARAM_INT);
    $stmt->execute();
    $rows = $stmt->fetchAll();

    $res->json([
        'total' => $total,
        'page' => $page,
        'per_page' => $perPage,
        'rows' => $rows,
    ]);
}
}

```


backend/app/controllers/AuthController.php

<?php

namespace App\controllers;

use App\core\Controller;

use App\core\Request;

use App\core\Response;

use App\models\User;

class AuthController extends Controller {

public function register(Request \$req, Response \$res): void {

 \$data = \$req->body();

 \$name = trim(\$data['name'] ?? '');

 \$email = strtolower(trim(\$data['email'] ?? ''));

 \$password = (string)(\$data['password'] ?? '');

 \$errors = [];

 if (\$name === '') \$errors['name'] = 'Name is required';

 if (!filter_var(\$email, FILTER_VALIDATE_EMAIL)) \$errors['email'] = 'Invalid email';

 if (strlen(\$password) < 6) \$errors['password'] = 'Password must be at least 6 characters';

 if (\$errors) { \$res->json(['errors'=>\$errors],422); return; }

 if (User::findByEmail(\$email)) { \$res->json(['errors'=>['email'=>'Email already in use']],422); return; }

 \$hash = password_hash(\$password, PASSWORD_BCRYPT);

 \$userId = User::create(['name'=>\$name, 'email'=>\$email, 'password_hash'=>\$hash, 'role'=>'customer']);

```

    if (!$userId) { $res->json(['error'=>'Registration failed'],500); return; }

    $_SESSION['user_id'] = $userId;
    $_SESSION['role'] = 'customer';
    $res->json(['message'=>'Registered','user'=>User::findById($userId)],201);
}

public function login(Request $req, Response $res): void {
    $data = $req->body();
    $email = strtolower(trim($data['email'] ?? ''));
    $password = (string)($data['password'] ?? '');
    if (!filter_var($email, FILTER_VALIDATE_EMAIL) || $password==='') { $res->json(['error'=>'Invalid credentials'],401); return; }
    $user = User::findByEmail($email);
    if (!$user || !password_verify($password, $user['password_hash'])) { $res->json(['error'=>'Invalid credentials'],401); return; }
    session_regenerate_id(true);
    $_SESSION['user_id'] = (int)$user['id'];
    $_SESSION['role'] = $user['role'];
    $res->json(['message'=>'Logged in','user'=>User::publicUser($user)],200);
}

```

```

public function logout(Request $req, Response $res): void {
    $_SESSION = [];
    if (ini_get("session.use_cookies")) {
        $params = session_get_cookie_params();
        setcookie(session_name(), '', time() - 42000, $params['path'], $params['domain'],
        $params['secure'], $params['httponly']);
    }
}

```

```
}  
session_destroy();  
$res->json(['message'=>'Logged out']);  
}
```

```
public function me(Request $req, Response $res): void {  
    $uid = $_SESSION['user_id'] ?? null;  
    if (!$uid) { $res->json(['user'=>null],200); return; }  
    $user = User::findById((int)$uid);  
    $res->json(['user'=>User::publicUser($user)]);  
}  
}
```

backend/app/controllers/BookingController.php

<?php

namespace App\controllers;

use App\core\Controller;

use App\core\Request;

use App\core\Response;

use App\models\Booking;

use App\services\TimeSlotService;

use RuntimeException;

class BookingController extends Controller {

public function index(Request \$req, Response \$res): void {

 \$filters = \$req->query();

 \$res->json(['data'=>Booking::list(\$filters)]);

}

public function store(Request \$req, Response \$res): void {

 \$data = \$req->body();

 \$errors = [];

 \$userId = (int)(\$data['user_id'] ?? (\$_SESSION['user_id'] ?? 0));

 \$serviceId = (int)(\$data['service_id'] ?? 0);

 \$date = \$data['date'] ?? '';

 \$start = \$data['start_time'] ?? '';

 if (\$userId<=0) \$errors['user_id']='User required';

 if (\$serviceId<=0) \$errors['service_id']='Service required';

 if (!\$date) \$errors['date']='Date required';

```

if (!$start) $errors['start_time']='Start time required';
if ($errors) { $res->json(['errors'=>$errors],422); return; }

$end = TimeSlotService::computeEndTime($serviceId, $start);
// In no-staff mode, enforce general business hours only.
if (!TimeSlotService::isWithinWorkingHours(0, $date, $start, $end)) {
    $res->json(['error'=>'Outside working hours'],422); return;
}
try {
    $id = Booking::create([
        'user_id'=>$userId,
        'service_id'=>$serviceId,
        'date'=>$date,
        'start_time'=>$start,
        'end_time'=>$end,
        'notes'=>$data['notes'] ?? '',
        'status'=>'pending'
    ]);
    $res->json(['message'=>'Created','id'=>$id],201);
} catch (RuntimeException $e) {
    if ($e->getMessage()=='Slot taken') { $res->json(['error'=>'Slot taken'],409); return; }
    $res->json(['error'=>'Create failed'],500);
}
}

public function update(Request $req, Response $res): void {
    $data = $req->body();

```



```

    $id = (int)($data['id'] ?? 0);
    if ($id<=0) { $res->json(['error'=>'Invalid id'],400); return; }
    $ok = Booking::update($id,$data);
    $res->json(['updated'=>$ok]);
}

```

```

public function updateStatus(Request $req, Response $res): void {
    $data = $req->body();
    $id = (int)($data['id'] ?? 0);
    $status = $data['status'] ?? '';
    if ($id<=0 || !in_array($status,['pending','confirmed','canceled','completed'],true)) {
        $res->json(['error'=>'Invalid input'],422); return;
    }
    $ok = Booking::updateStatus($id,$status);
    $res->json(['updated'=>$ok]);
}

```

```

public function destroy(Request $req, Response $res): void {
    $id = (int)($req->query()['id'] ?? ($req->body()['id'] ?? 0));
    if ($id<=0) { $res->json(['error'=>'Invalid id'],400); return; }
    $ok = Booking::delete($id);
    $res->json(['deleted'=>$ok]);
}
}

```

backend/app/controllers/DashboardController.php

<?php

// DashboardController — Admin analytics endpoints: summary, service coverage, bookings list and CSV.

namespace App\controllers;

use App\core\Controller;

use App\core\Request;

use App\core\Response;

use App\core\Database;

class DashboardController extends Controller {

public function summary(Request \$req, Response \$res): void {

 \$pdo = Database::pdo();

 \$from = \$req->query()['from'] ?? date('Y-m-01');

 \$to = \$req->query()['to'] ?? date('Y-m-t');

 \$stmt = \$pdo->prepare("SELECT COUNT(*) cnt FROM bookings WHERE date BETWEEN :from AND :to");

 \$stmt->execute([':from'=>\$from, ':to'=>\$to]);

 \$totalBookings = (int)\$stmt->fetchColumn();

 \$stmt = \$pdo->prepare("SELECT s.name, COUNT(*) cnt

 FROM bookings b

 INNER JOIN services s ON s.id=b.service_id

 WHERE b.date BETWEEN :from AND :to

 GROUP BY s.id, s.name

 ORDER BY cnt DESC


```

        GROUP BY date

        ORDER BY date ASC");

$stmt->execute([':from'=>$from, ':to'=>$to]);

$daily = $stmt->fetchAll();


// Revenue totals and average duration for services booked in range

$stmt = $pdo->prepare("SELECT COALESCE(SUM(s.price),0) AS total_rev,
COALESCE(AVG(s.duration_minutes),0) AS avg_duration

        FROM bookings b

        INNER JOIN services s ON s.id=b.service_id

        WHERE b.date BETWEEN :from AND :to");

$stmt->execute([':from'=>$from, ':to'=>$to]);

$row = $stmt->fetch();

$totalRevenue = (float)($row['total_rev'] ?? 0);

$avgDuration = (float)($row['avg_duration'] ?? 0);

$days = max(1, (int)((strtotime($to) - strtotime($from)) / 86400) + 1);

$avgPerDay = $totalRevenue / $days;


// Revenue by category (no HAVING filter)

$stmt = $pdo->prepare("SELECT COALESCE(NULLIF(s.category,''),'Uncategorized') AS
category, COALESCE(SUM(s.price),0) AS total_rev

        FROM bookings b

        INNER JOIN services s ON s.id=b.service_id

        WHERE b.date BETWEEN :from AND :to

        GROUP BY COALESCE(NULLIF(s.category,''),'Uncategorized')

        ORDER BY total_rev DESC");

$stmt->execute([':from'=>$from, ':to'=>$to]);

$revByCategory = $stmt->fetchAll();

```

```

$res->json([
    'range' => ['from'=>$from,'to'=>$to],
    'totals' => ['bookings'=>$totalBookings],
    'top_services' => $topServices,
    'by_status' => $byStatus,
    'by_category' => $byCategory,
    'daily' => $daily,
    'revenue' => [
        'total' => $totalRevenue,
        'avg_per_day' => $avgPerDay,
        'avg_duration_minutes' => $avgDuration,
    ],
    'revenue_by_category' => $revByCategory,
    'bounds' => [
        'min_date' => $bounds['min_date'] ?? null,
        'max_date' => $bounds['max_date'] ?? null,
    ],
]);
}

```

```

public function bookingsCsv(Request $req, Response $res): void {
    $pdo = Database::pdo();
    $q = $req->query()['q'] ?? '';
    $status = $req->query()['status'] ?? '';
    $from = $req->query()['from'] ?? date('Y-m-01');
    $to = $req->query()['to'] ?? date('Y-m-t');
}

```

```

$where = ["b.date BETWEEN :from AND :to"];

$params = [':from'=>$from, ':to'=>$to];

if ($status !== "") { $where[] = 'b.status = :status'; $params[':status'] = $status; }

if ($q !== "") {

    $where[] = '(b.id = :idq OR u.name LIKE :uq OR s.name LIKE :sq)';

    $params[':idq'] = ctype_digit($q) ? (int)$q : -1;

    $params[':uq'] = '%'.$q.'%';

    $params[':sq'] = '%'.$q.'%';

}

$whereSql = 'WHERE '.implode(' AND ', $where);

header('Content-Type: text/csv; charset=utf-8');

header('Content-Disposition: attachment;
filename="bookings_'.$from.'_to_'.$to.'.csv"');

$out = fopen('php://output', 'w');

fputcsv($out, ['ID', 'Date', 'Start', 'End', 'Status', 'Customer', 'Service', 'Price', 'Notes']);

$stmt = $pdo->prepare("SELECT b.id, b.date, b.start_time, b.end_time, b.status, b.notes,
                        u.name AS customer_name, s.name AS service_name, s.price
                        FROM bookings b
                        INNER JOIN services s ON s.id=b.service_id
                        INNER JOIN users u ON u.id=b.user_id
                        $whereSql
                        ORDER BY b.date DESC, b.start_time DESC");

$stmt->execute($params);

while ($r = $stmt->fetch()) {

    fputcsv($out, [

```

```

        $r['id'], $r['date'], $r['start_time'], $r['end_time'], $r['status'],
        $r['customer_name'], $r['service_name'], number_format((float)$r['price'],2,'.',''),
        $r['notes']
    ));
}
fclose($out);
exit;
}

```

```

public function rightJoinDemo(Request $req, Response $res): void {
    $pdo = Database::pdo();
    $from = $req->query()['from'] ?? date('Y-m-01');
    $to = $req->query()['to'] ?? date('Y-m-t');

    // RIGHT JOIN example (mirrors the LEFT JOIN coverage but uses RIGHT JOIN syntax):
    // Ensures all services appear even if they have zero bookings in the date range.
    $stmt = $pdo->prepare("SELECT s.id, s.name, COALESCE(COUNT(b.id),0) AS bookings
        FROM bookings b
        RIGHT JOIN services s
        ON s.id = b.service_id
        AND b.date BETWEEN :from AND :to
        GROUP BY s.id, s.name
        ORDER BY bookings DESC, s.name ASC");
    $stmt->execute([':from'=>$from, ':to'=>$to]);
    $rows = $stmt->fetchAll();

    $res->json([

```

```

        'range' => ['from'=>$from,'to'=>$to],
        'services' => $rows,
    ]);
}

```

```

public function serviceCoverage(Request $req, Response $res): void {
    $pdo = Database::pdo();
    $from = $req->query()['from'] ?? date('Y-m-01');
    $to = $req->query()['to'] ?? date('Y-m-t');

    // LEFT JOIN: include services with zero bookings in the window
    $stmt = $pdo->prepare("SELECT s.id, s.name, COALESCE(COUNT(b.id),0) AS bookings
        FROM services s
        LEFT JOIN bookings b
        ON b.service_id = s.id
        AND b.date BETWEEN :from AND :to
        GROUP BY s.id, s.name
        ORDER BY bookings DESC, s.name ASC");
    $stmt->execute([':from'=>$from, ':to'=>$to]);
    $rows = $stmt->fetchAll();

    $res->json([
        'range' => ['from'=>$from,'to'=>$to],
        'services' => $rows,
    ]);
}

```



```

public function bookingsList(Request $req, Response $res): void {

    $pdo = Database::pdo();

    $q = $req->query()['q'] ?? '';

    $status = $req->query()['status'] ?? '';

    $from = $req->query()['from'] ?? date('Y-m-01');

    $to = $req->query()['to'] ?? date('Y-m-t');

    $page = max(1, (int)($req->query()['page'] ?? 1));

    $perPage = min(100, max(5, (int)($req->query()['per_page'] ?? 20)));

    $offset = ($page - 1) * $perPage;

    $where = ["b.date BETWEEN :from AND :to"];

    $params = ['from'=>$from, 'to'=>$to];

    if ($status !== '') { $where[] = 'b.status = :status'; $params['status'] = $status; }

    if ($q !== '') {

        $where[] = '(b.id = :idq OR u.name LIKE :uq OR s.name LIKE :sq)';

        $params['idq'] = ctype_digit($q) ? (int)$q : -1;

        $params['uq'] = '%'.$q.'%';

        $params['sq'] = '%'.$q.'%';

    }

    $whereSql = 'WHERE '.implode(' AND ', $where);

    $stmt = $pdo->prepare("SELECT COUNT(*) FROM bookings b INNER JOIN services s ON
s.id=b.service_id INNER JOIN users u ON u.id=b.user_id $whereSql");

    $stmt->execute($params);

    $total = (int)$stmt->fetchColumn();

    $stmt = $pdo->prepare("SELECT b.id, b.date, b.start_time, b.end_time, b.status, b.notes,
b.created_at,

```

```

        s.name AS service_name, s.price, u.name AS customer_name
FROM bookings b
INNER JOIN services s ON s.id=b.service_id
INNER JOIN users u ON u.id=b.user_id
$whereSql
ORDER BY b.date DESC, b.start_time DESC
LIMIT :limit OFFSET :offset");

foreach ($params as $k=>$v) { $stmt->bindValue($k, $v); }
$stmt->bindValue(':limit', $perPage, \PDO::PARAM_INT);
$stmt->bindValue(':offset', $offset, \PDO::PARAM_INT);
$stmt->execute();
$rows = $stmt->fetchAll();

$res->json([
    'total' => $total,
    'page' => $page,
    'per_page' => $perPage,
    'rows' => $rows,
]);
}
}

```

backend/app/controllers/ServiceController.php

```
<?php
```

```
// ServiceController — Public service listing and admin create/update/delete with image uploads.
```

```
namespace App\controllers;
```

```
use App\core\Controller;
```

```
use App\core\Request;
```

```
use App\core\Response;
```

```
use App\models\Service;
```

```
class ServiceController extends Controller {
```

```
    public function index(Request $req, Response $res): void {
```

```
        $q = trim($req->query()['q'] ?? '');
```

```
        $active = $req->query()['active'] ?? null;
```

```
        $sort = $req->query()['sort'] ?? 'created_at';
```

```
        $order = strtoupper($req->query()['order'] ?? 'DESC');
```

```
        if (!in_array($order, ['ASC', 'DESC'], true)) { $order = 'DESC'; }
```

```
        $services = Service::list($q, $active, $sort, $order);
```

```
        $res->json(['data'=>$services]);
```

```
    }
```

```
    public function store(Request $req, Response $res): void {
```

```
        // Accept both JSON and multipart/form-data
```

```
        $data = $req->body();
```

```
        $id = (int)($data['id'] ?? 0);
```

```
        $name = trim($data['name'] ?? '');
```

```

// Defaults: price and duration have sensible fallbacks
$price = isset($data['price']) ? (float)$data['price'] : 500.0;
if ($price <= 0) { $price = 500.0; }
$duration = isset($data['duration_minutes']) ? (int)$data['duration_minutes'] : 60;
if ($duration <= 0) { $duration = 60; }
$desc = trim($data['description'] ?? '');
$category = trim($data['category'] ?? '');
$imagePath = trim($data['image_path'] ?? '');
$isFeatured = (int)($data['is_featured'] ?? 0);

// Handle uploaded file if present
if (!empty($_FILES['image']) && is_uploaded_file($_FILES['image']['tmp_name'])) {
    $root = dirname(__DIR__, 2);
    $uploadDir = $root . '/public/uploads/services';
    if (!is_dir($uploadDir)) { @mkdir($uploadDir, 0775, true); }
    $ext = pathinfo($_FILES['image']['name'], PATHINFO_EXTENSION) ?: 'jpg';
    $safe = preg_replace('/[^\a-zA-Z0-9_-]/', '', strtolower(pathinfo($_FILES['image']['name'],
PATHINFO_FILENAME)));
    if ($safe==='') { $safe = 'service'; }
    $filename = $safe . '-' . time() . '.' . $ext;
    $dest = $uploadDir . '/' . $filename;
    if (!move_uploaded_file($_FILES['image']['tmp_name'], $dest)) {
        $res->json(['error'=>'Failed to upload image'], 500); return;
    }
    // Public path exposed via backend/public base
    $imagePath = '/uploads/services/' . $filename;
}

```

```

$errors = [];

if ($name=== '') $errors['name']='Name required';
if ($price<=0) $errors['price']='Price must be positive';
if ($duration<=0) $errors['duration_minutes']='Duration must be positive';
if ($errors) { $res->json(['errors'=>$errors],422); return; }

if ($id > 0) {
    $current = Service::findById($id);
    if (!$current) { $res->json(['error'=>'Not found'],404); return; }
    // Preserve current image if none provided
    if ($imagePath === '') { $imagePath = $current['image_path'] ?? null; }
    $ok = Service::update($id, [
        'name'=>$name,
        'category'=>$category ?: null,
        'description'=>$desc,
        'price'=>$price,
        'duration_minutes'=>$duration,
        'image_path'=>$imagePath,
        'is_featured'=>$isFeatured,
        'active'=>$data['active'] ?? $current['active'] ?? 1,
    ]);
    if (!$ok) { $res->json(['error'=>'Update failed'],500); return; }
    $res->json(['message'=>'Updated','id'=>$id]);
    return;
}

```

```

$newId = Service::create([
    'name'=>$name,
    'category'=>$category ?: null,
    'description'=>$desc,
    'price'=>$price,
    'duration_minutes'=>$duration,
    'image_path'=>$imagePath ?: null,
    'is_featured'=>$isFeatured
]);
$res->json(['message'=>'Created','id'=>$newId],201);
}

```

```

public function update(Request $req, Response $res): void {
    $data = $req->body();
    $id = (int)($data['id'] ?? 0);
    if ($id<=0) { $res->json(['error'=>'Invalid id'],400); return; }
    $ok = Service::update($id, $data);
    if (!$ok) { $res->json(['error'=>'Update failed'],500); return; }
    $res->json(['message'=>'Updated']);
}

```

```

public function destroy(Request $req, Response $res): void {
    $id = (int)($req->query()['id'] ?? ($req->body()['id'] ?? 0));
    if ($id<=0) { $res->json(['error'=>'Invalid id'],400); return; }
    $ok = Service::delete($id);
    $res->json(['deleted'=>$ok]);
}

```

}

backend/app/controllers/StreamController.php

<?php

// StreamController — Server-Sent Events (SSE) stream for lightweight real-time pings.

namespace App\controllers;

use App\core\Request;

use App\core\Response;

class StreamController {

public function bookings(Request \$req, Response \$res): void {

// SSE headers

header('Content-Type: text/event-stream');

header('Cache-Control: no-cache');

header('Connection: keep-alive');

// Ensure output buffering does not interfere

if (function_exists('apache_setenv')) { @apache_setenv('no-gzip', '1'); }

@ini_set('zlib.output_compression', '0');

@ini_set('implicit_flush', '1');

while (ob_get_level() > 0) { @ob_end_flush(); }

@ob_implicit_flush(1);

// Simple heartbeat loop; real events could be pushed on changes

\$start = time();

\$maxSeconds = 300; // 5 minutes per connection

echo ": connected\n\n"; // comment to open the stream

flush();

while (true) {

if (connection_aborted()) { break; }


```
$elapsed = time() - $start;
if ($elapsed > $maxSeconds) { break; }
echo "event: ping\n";
echo "data: {}\n\n";
flush();
// 25s between pings
sleep(25);
}
// graceful end
echo "event: end\n";
echo "data: {}\n\n";
flush();
}
}
```

backend/app/controllers/UserController.php

<?php

namespace App\controllers;

use App\core\Controller;

use App\core\Request;

use App\core\Response;

use App\models\User;

class UserController extends Controller {

public function index(Request \$req, Response \$res): void {

\$uid = \$_SESSION['user_id'] ?? null;

if (!\$uid) { \$res->json(['error'=>'Unauthorized'],401); return; }

\$user = User::findById((int)\$uid);

\$res->json(['user'=>User::publicUser(\$user)]);

}

public function update(Request \$req, Response \$res): void {

\$uid = \$_SESSION['user_id'] ?? null;

if (!\$uid) { \$res->json(['error'=>'Unauthorized'],401); return; }

\$data = \$req->body();

\$name = trim(\$data['name'] ?? "");

if (\$name=="") { \$res->json(['errors'=>['name'=>'Name required']],422); return; }

\$ok = User::updateProfile((int)\$uid, ['name'=>\$name,'phone'=>\$data['phone'] ?? null]);

\$res->json(['updated'=>\$ok]);

}

}

Core

backend/app/core/Controller.php

```
<?php
```

```
// Base Controller — Common base for controllers; can hold shared helpers.
```

```
namespace App\core;
```

```
class Controller {
```

```
    protected function json($data, int $status = 200): void {
```

```
        http_response_code($status);
```

```
        header('Content-Type: application/json');
```

```
        echo json_encode($data);
```

```
    }
```

```
}
```

backend/app/core/Database.php

<?php

// Database — PDO connection singleton/factory for database access.

namespace App\core;

use PDO;use PDOException;use RuntimeException;

class Database {

private static ?PDO \$pdo = null;

public static function pdo(): PDO {

if (self::\$pdo === null) {

\$host = \$_ENV['DB_HOST'] ?? '127.0.0.1';

\$port = (int)(\$_ENV['DB_PORT'] ?? 3306);

\$db = \$_ENV['DB_DATABASE'] ?? 'glowupnow';

\$user = \$_ENV['DB_USERNAME'] ?? 'root';

\$pass = \$_ENV['DB_PASSWORD'] ?? '';

\$dsn = "mysql:host={\$host};port={\$port};dbname={\$db};charset=utf8mb4";

try {

self::\$pdo = new PDO(\$dsn, \$user, \$pass, [

PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION,

PDO::ATTR_DEFAULT_FETCH_MODE => PDO::FETCH_ASSOC,

PDO::ATTR_EMULATE_PREPARES => false,

]);

} catch (PDOException \$e) {

throw new RuntimeException('DB connection failed: ' . \$e->getMessage());

}

}

return self::\$pdo;

}

}

backend/app/core/Request.php

<?php

// Request — Encapsulates HTTP request data (query, body, files, headers).

namespace App\core;

class Request {

public function method(): string { return strtoupper(\$_SERVER['REQUEST_METHOD'] ?? 'GET'); }

public function path(): string {

 \$uri = \$_SERVER['REQUEST_URI'] ?? '/';

 \$script = \$_SERVER['SCRIPT_NAME'] ?? '';

 \$base = rtrim(str_replace('index.php', '', \$script), '/');

 \$path = parse_url(\$uri, PHP_URL_PATH) ?? '/';

 if (\$base && str_starts_with(\$path, \$base)) { \$path = substr(\$path, strlen(\$base)); }

 return '/' . ltrim(\$path, '/');

}

public function body(): array {

 \$input = file_get_contents('php://input');

 \$contentType = \$_SERVER['CONTENT_TYPE'] ?? '';

 if (str_contains(\$contentType, 'application/json')) {

 \$decoded = json_decode(\$input, true);

 return is_array(\$decoded) ? \$decoded : [];

 }

 return \$_POST ? : [];

}

public function query(): array { return \$_GET ? : []; }

public function headers(): array { return function_exists('getallheaders') ? (getallheaders() ? : []) : []; }

}

backend/app/core/Response.php

<?php

// Response — Minimal JSON response helper.

namespace App\core;

class Response {

public function json(\$data, int \$status = 200): void {

http_response_code(\$status);

header('Content-Type: application/json');

echo json_encode(\$data);

}

}

backend/app/core/Router.php

<?php

namespace App\core;

class Router {

private array \$routes = [];

private array \$middlewares = [];

public function add(string \$method, string \$path, callable \$handler, array \$mw = []): void {

\$this->routes[strtoupper(\$method)][\$path] = \$handler;

if (!empty(\$mw)) { \$this->middlewares[\$method.\$path] = \$mw; }

}

public function dispatch(Request \$req, Response \$res): void {

\$method = \$req->method();

\$path = \$req->path();

\$handler = \$this->routes[\$method][\$path] ?? null;

if (!\$handler) { \$res->json(['error'=>'Not Found','path'=>\$path],404); return; }

\$stack = \$this->middlewares[\$method.\$path] ?? [];

\$next = function() use (&\$handler, \$req, \$res) { call_user_func(\$handler, \$req, \$res); };

while (\$mw = array_pop(\$stack)) {

\$prev = \$next;

\$next = function() use (\$mw, \$req, \$res, \$prev) { \$mw(\$req, \$res, \$prev); };

}

\$next();

}

}

Middlewares

backend/app/middlewares/AdminMiddleware.php

```
<?php
```

```
namespace App\middlewares;
```

```
use App\core\Request;
```

```
use App\core\Response;
```

```
class AdminMiddleware {
```

```
    public function __invoke(Request $req, Response $res, callable $next): void {
```

```
        if (!(isset($_SESSION['user_id']) && ($_SESSION['role'] ?? '') === 'admin')) {
```

```
            $res->json(['error' => 'Forbidden'], 403);
```

```
            return;
```

```
        }
```

```
        $next();
```

```
    }
```

```
}
```

backend/app/middlewares/AuthMiddleware.php

```
<?php
```

```
namespace App\middlewares;
```

```
use App\core\Request;
```

```
use App\core\Response;
```

```
class AuthMiddleware {
```

```
    public function __invoke(Request $req, Response $res, callable $next): void {
```

```
        if (!isset($_SESSION['user_id'])) {
```

```
            $res->json(['error' => 'Unauthorized'], 401);
```

```
            return;
```

```
        }
```

```
        $next();
```

```
    }
```

```
}
```

backend/app/middlewares/CorsMiddleware.php

<?php

namespace App\middlewares;

use App\core\Request;

use App\core\Response;

class CorsMiddleware {

public function __invoke(Request \$req, Response \$res, callable \$next): void {

 \$origins = \$_ENV['CORS_ALLOWED_ORIGINS'] ?? '';

 \$allowed = array_filter(array_map('trim', explode(',', \$origins)));

 \$origin = \$_SERVER['HTTP_ORIGIN'] ?? '';

 // Allow any localhost origin (any port) during development

 \$isLocalhost = \$origin && (

 preg_match('#^http://localhost(?:\d+)?\$#', \$origin) ||

 preg_match('#^http://127\.\d\.\d\.\d(?:\d+)?\$#', \$origin)

);

 if (\$origin && (\$isLocalhost || in_array(\$origin, \$allowed, true))) {

 header("Access-Control-Allow-Origin: {\$origin}");

 header('Vary: Origin');

 header('Access-Control-Allow-Credentials: true');

 }

 // Echo requested headers if provided (preflight), fallback to common defaults

 \$reqHeaders = \$_SERVER['HTTP_ACCESS_CONTROL_REQUEST_HEADERS'] ?? 'Content-Type, Authorization, Accept';

 header("Access-Control-Allow-Headers: {\$reqHeaders}");

```
header('Access-Control-Allow-Methods: GET, POST, PUT, PATCH, DELETE, OPTIONS');
```

```
if ($req->method() === 'OPTIONS') {  
    http_response_code(204);  
    exit;  
}  
$next();  
}  
}
```

Models

backend/app/models/Booking.php

```
<?php
```

```
// Booking model — Data access for bookings with joins to services and users.
```

```
namespace App\models;
```

```
use App\core\Database;use PDO;use PDOException;use RuntimeException;
```

```
class Booking {
```

```
    public static function list(array $filters = []): array {
```

```
        $pdo = Database::pdo();
```

```
        $sql = "SELECT b.id, b.user_id, b.service_id, b.date, b.start_time, b.end_time, b.status, b.notes,  
b.created_at,
```

```
            s.name AS service_name, s.price, u.name AS customer_name
```

```
        FROM bookings b
```

```
        INNER JOIN services s ON s.id = b.service_id
```

```
        INNER JOIN users u ON u.id = b.user_id
```

```
        WHERE 1";
```

```
        $params = [];
```

```
        if (!empty($filters['status'])) { $sql .= " AND b.status = :status"; $params[':status'] =  
$filters['status']; }
```

```
        if (!empty($filters['service_id'])) { $sql .= " AND b.service_id = :sid"; $params[':sid'] =  
(int)$filters['service_id']; }
```

```
        if (!empty($filters['user_id'])) { $sql .= " AND b.user_id = :uid"; $params[':uid'] =  
(int)$filters['user_id']; }
```

```
        if (!empty($filters['from'])) { $sql .= " AND b.date >= :from"; $params[':from'] = $filters['from']; }
```

```
        if (!empty($filters['to'])) { $sql .= " AND b.date <= :to"; $params[':to'] = $filters['to']; }
```

```
        $sql .= " ORDER BY b.date DESC, b.start_time DESC";
```

```
        $stmt = $pdo->prepare($sql);
```

```

$stmt->execute($params);

return $stmt->fetchAll();
}

```

```

public static function create(array $data): int {

```

```

    $pdo = Database::pdo();

```

```

    try {

```

```

        $pdo->beginTransaction();

```

```

        // No-staff mode: lock overlapping bookings for the same date/time window

```

```

        $lock = $pdo->prepare("SELECT id FROM bookings WHERE date = :date AND (

```

```

            (start_time < :end_time AND end_time > :start_time)

```

```

        ) FOR UPDATE");

```

```

        $lock->execute([

```

```

            ':date' => $data['date'],

```

```

            ':start_time' => $data['start_time'],

```

```

            ':end_time' => $data['end_time'],

```

```

        ]);

```

```

        if ($lock->fetch()) { throw new RuntimeException('Slot taken'); }

```

```

        $stmt = $pdo->prepare("INSERT INTO bookings (user_id, service_id, date, start_time, end_time,
status, notes, created_at)

```

```

            VALUES (:user_id,:service_id,:date,:start_time,:end_time,:status,:notes,NOW())");

```

```

        $stmt->execute([

```

```

            ':user_id' => $data['user_id'],

```

```

            ':service_id' => $data['service_id'],

```

```

            ':date' => $data['date'],

```

```

            ':start_time' => $data['start_time'],

```

```

            ':end_time' => $data['end_time'],

```

```

            ':status' => $data['status'] ?? 'pending',

```

```

        ':notes' => $data['notes'] ?? '',
    ]);
    $id = (int)$pdo->lastInsertId();
    $pdo->commit();
    return $id;
} catch (RuntimeException $e) {
    $pdo->rollBack();
    throw $e;
} catch (PDOException $e) {
    $pdo->rollBack();
    throw $e;
}
}

```

```

public static function update(int $id, array $data): bool {
    $pdo = Database::pdo();
    $stmt = $pdo->prepare("UPDATE bookings SET notes=:notes WHERE id=:id");
    return $stmt->execute([':notes'=>$data['notes'] ?? '', ':id'=>$id]);
}

```

```

public static function updateStatus(int $id, string $newStatus): bool {
    $pdo = Database::pdo();
    $stmt = $pdo->prepare("UPDATE bookings SET status=:status WHERE id=:id");
    return $stmt->execute([':status'=>$newStatus, ':id'=>$id]);
}

```

```

public static function delete(int $id): bool {
    $pdo = Database::pdo();
    $stmt = $pdo->prepare("DELETE FROM bookings WHERE id=:id");
}

```

```
        return $stmt->execute([':id'=>$id]);
    }
}
```


backend/app/models/Service.php

<?php

// Service model — Data access for services (CRUD) and related queries.

namespace App\models;

use App\core\Database;use PDO;

class Service {

public static function list(string \$q = "", \$active = null, string \$sort='created_at', string \$order='DESC'): array {

 \$pdo = Database::pdo();

 \$allowedSort = ['created_at','price','name'];

 if (!in_array(\$sort,\$allowedSort,true)) \$sort='created_at';

 \$sql = "SELECT
id,name,category,description,price,duration_minutes,image_path,is_featured,active,created_at FROM
services WHERE 1";

 \$params = [];

 if (\$q !== "") {

 \$sql .= " AND (name LIKE :q1 OR description LIKE :q2)";

 \$params[':q1'] = '%'.\$q.'%';

 \$params[':q2'] = '%'.\$q.'%';

 }

 if (\$active !== null && \$active !== "") { \$sql .= " AND active = :active"; \$params[':active'] =
(int)\$active; }

 \$sql .= " ORDER BY {\$sort} {\$order}";

 \$stmt = \$pdo->prepare(\$sql);

 \$stmt->execute(\$params);

 return \$stmt->fetchAll();

}

public static function create(array \$data): int {

```

    $pdo = Database::pdo();

    $stmt = $pdo->prepare("INSERT INTO services
(name,category,description,price,duration_minutes,image_path,is_featured,active,created_at)
VALUES (:name,:category,:description,:price,:duration,:image_path,:is_featured,1,NOW())");

    $stmt->execute([
        ':name'=>$data['name'],
        ':category'=>$data['category'] ?? null,
        ':description'=>$data['description'] ?? "",
        ':price'=>$data['price'],
        ':duration'=>$data['duration_minutes'],
        ':image_path'=>$data['image_path'] ?? null,
        ':is_featured'=>(int)($data['is_featured'] ?? 0)
    ]);

    return (int)$pdo->lastInsertId();
}

public static function update(int $id, array $data): bool {
    $pdo = Database::pdo();

    $stmt = $pdo->prepare("UPDATE services SET name=:name, category=:category,
description=:description, price=:price, duration_minutes=:duration, image_path=:image_path,
is_featured=:is_featured, active=:active WHERE id=:id");

    return $stmt->execute([
        ':name'=>$data['name'],
        ':category'=>$data['category'] ?? null,
        ':description'=>$data['description'] ?? "",
        ':price'=>(float)($data['price'] ?? 0),
        ':duration'=>(int)($data['duration_minutes'] ?? 0),
        ':image_path'=>$data['image_path'] ?? null,
        ':is_featured'=>(int)($data['is_featured'] ?? 0),
        ':active'=>(int)($data['active'] ?? 1),
        ':id'=>$id
    ]);
}

```

```

    });
}

public static function delete(int $id): bool {
    $pdo = Database::pdo();

    $stmt = $pdo->prepare("DELETE FROM services WHERE id=:id");

    return $stmt->execute([':id'=>$id]);
}

public static function findById(int $id): ?array {
    $pdo = Database::pdo();

    $stmt = $pdo->prepare("SELECT
id,name,category,description,price,duration_minutes,image_path,is_featured,active,created_at FROM
services WHERE id=:id LIMIT 1");

    $stmt->execute([':id'=>$id]);

    $row = $stmt->fetch();

    return $row ?: null;
}
}

```

backend/app/models/User.php

<?php

// User model — Data access for users (auth, profile) and helpers.

namespace App\models;

use App\core\Database;use PDO;

class User {

public static function create(array \$data): ?int {

 \$pdo = Database::pdo();

 \$stmt = \$pdo->prepare("INSERT INTO users (name,email,password_hash,role,phone,created_at)
VALUES (:name,:email,:password_hash,:role,:phone,NOW())");

 \$ok = \$stmt->execute([

 'name'=>\$data['name'],

 'email'=>\$data['email'],

 'password_hash'=>\$data['password_hash'],

 'role'=>\$data['role'] ?? 'customer',

 'phone'=>\$data['phone'] ?? null,

]);

 return \$ok ? (int)\$pdo->lastInsertId() : null;

}

public static function findByEmail(string \$email): ?array {

 \$pdo = Database::pdo();

 \$stmt = \$pdo->prepare("SELECT * FROM users WHERE email = :email LIMIT 1");

 \$stmt->execute(['email'=>\$email]);

 \$row = \$stmt->fetch();

 return \$row ? : null;

}

public static function findById(int \$id): ?array {

```

        $pdo = Database::pdo();
        $stmt = $pdo->prepare("SELECT * FROM users WHERE id=:id");
        $stmt->execute([':id'=>$id]);
        $row = $stmt->fetch();
        return $row ? : null;
    }

    public static function publicUser(?array $user): ?array {
        if (!$user) return null;
        return [
            'id'=>(int)$user['id'],
            'name'=>$user['name'],
            'email'=>$user['email'],
            'role'=>$user['role'],
            'phone'=>$user['phone'] ?? null,
            'created_at'=>$user['created_at'] ?? null,
        ];
    }

    public static function updateProfile(int $id, array $data): bool {
        $pdo = Database::pdo();
        $stmt = $pdo->prepare("UPDATE users SET name=:name, phone=:phone WHERE id=:id");
        return $stmt->execute([':name'=>$data['name'], ':phone'=>$data['phone'] ?? null, ':id'=>$id]);
    }
}

```

Services

backend/app/services/TimeSlotService.php

```
<?php
```

```
namespace App\services;
```

```
use App\models\Service;
```

```
class TimeSlotService {
```

```
    public static function computeEndTime(int $serviceId, string $startTime): string {
```

```
        $service = Service::findById($serviceId);
```

```
        $duration = (int)($service['duration_minutes'] ?? 0);
```

```
        if ($duration <= 0) { $duration = 60; }
```

```
        $start = strtotime($startTime);
```

```
        return date('H:i:s', $start + ($duration * 60));
```

```
    }
```

```
    public static function isWithinWorkingHours(int $staffId, string $date, string $start, string $end):  
    bool {
```

```
        // TODO: Implement using working_hours table; temporary rule 09:00-18:00
```

```
        $min = strtotime('09:00:00');
```

```
        $max = strtotime('18:00:00');
```

```
        $s = strtotime($start);
```

```
        $e = strtotime($end);
```

```
        return $s >= $min && $e <= $max && $e > $s;
```

```
    }
```

```
}
```

Bootstrap / Routes / Entry

backend/bootstrap.php

```
<?php
```

```
// Lightweight bootstrap without Composer.
```

```
// 1) Simple .env loader
```

```
$envFile = __DIR__ . '/.env';
```

```
if (is_file($envFile)) {
```

```
    $lines = file($envFile, FILE_IGNORE_NEW_LINES | FILE_SKIP_EMPTY_LINES);
```

```
    foreach ($lines as $line) {
```

```
        if (str_starts_with(trim($line), '#')) continue;
```

```
        $parts = explode('=', $line, 2);
```

```
        if (count($parts) === 2) {
```

```
            $key = trim($parts[0]);
```

```
            $val = trim($parts[1]);
```

```
            $val = trim($val, "\"' ");
```

```
            $_ENV[$key] = $val;
```

```
            putenv("{key}={val}");
```

```
        }
```

```
    }
```

```
}
```

```
// 2) Minimal PSR-4 autoloader for App\ namespace
```

```
spl_autoload_register(function(string $class){
```

```
    $prefix = 'App\\';
```

```
    if (strncmp($class, $prefix, strlen($prefix)) !== 0) return;
```

```
    $relative = substr($class, strlen($prefix));
```

```
    $path = __DIR__ . '/app/' . str_replace('\\', '/', $relative) . '.php';
```

```
    if (is_file($path)) require $path;
});
```

```
// 3) Error reporting
```

```
$debug = ($_ENV['APP_DEBUG'] ?? 'false') === 'true';
ini_set('display_errors', $debug ? '1' : '0');
ini_set('display_startup_errors', $debug ? '1' : '0');
error_reporting($debug ? E_ALL : 0);
```

```
// 4) Timezone & session
```

```
date_default_timezone_set('Asia/Manila');
session_name($_ENV['SESSION_NAME'] ?? 'GLOWUPSESSID');
if (session_status() === PHP_SESSION_NONE) session_start();
```


backend/public/index.php

<?php

require __DIR__ . '/../bootstrap.php';

use App\core\{Router, Request, Response};

\$origin = \$_SERVER['HTTP_ORIGIN'] ?? '';

\$allowed = array_filter(array_map('trim', explode(',', \$_ENV['CORS_ALLOWED_ORIGINS'] ?? '')));

\$isLocalhost = \$origin && (

preg_match('#^http://localhost(?:\d+)?\$#', \$origin) ||

preg_match('#^http://127\.\d\.\d\.\d(?:\d+)?\$#', \$origin)

);

if (\$origin && (\$isLocalhost || in_array(\$origin, \$allowed, true))) {

header("Access-Control-Allow-Origin: {\$origin}");

header('Vary: Origin');

header('Access-Control-Allow-Credentials: true');

}

// Echo requested headers if provided (preflight), fallback to common defaults

\$reqHeaders = \$_SERVER['HTTP_ACCESS_CONTROL_REQUEST_HEADERS'] ?? 'Content-Type, Authorization, Accept';

header("Access-Control-Allow-Headers: {\$reqHeaders}");

header('Access-Control-Allow-Methods: GET, POST, PUT, PATCH, DELETE, OPTIONS');

if (((\$_SERVER['REQUEST_METHOD'] ?? 'GET') === 'OPTIONS') {

http_response_code(204);

exit;

}

\$router = new Router();

```
$request = new Request();
```

```
$response = new Response();
```

```
require __DIR__ . '/../routes/api.php';
```

```
$router->dispatch($request, $response);
```

backend/routes/api.php

<?php

// routes/api.php — Defines API routes, middlewares, and handlers (auth, services, bookings, analytics, SSE).

use App\core\{Request, Response};

use App\middlewares\CorsMiddleware;

use App\middlewares\AuthMiddleware;

use App\middlewares\AdminMiddleware;

use App\controllers\{AuthController, ServiceController, BookingController, UserController, DashboardController, StreamController, AdminUserController};

\$cors = new CorsMiddleware();

\$auth = new AuthMiddleware();

\$admin = new AdminMiddleware();

\$router->add('GET', '/api/health', function(Request \$req, Response \$res){

 \$res->json(['status'=>'ok', 'time'=>date('c')]);

}, [\$cors]);

// Auth

\$router->add('POST', '/api/auth/register', [new AuthController,'register'], [\$cors]);

\$router->add('POST', '/api/auth/login', [new AuthController,'login'], [\$cors]);

\$router->add('POST', '/api/auth/logout', [new AuthController,'logout'], [\$cors]);

\$router->add('GET', '/api/auth/me', [new AuthController,'me'], [\$cors, \$auth]);

// Services

\$router->add('GET', '/api/services', [new ServiceController,'index'], [\$cors]);

\$router->add('POST', '/api/services', [new ServiceController,'store'], [\$cors, \$auth, \$admin]);

\$router->add('PUT', '/api/services', [new ServiceController,'update'], [\$cors, \$auth, \$admin]);

```
$router->add('DELETE','/api/services', [new ServiceController,'destroy'], [$cors, $auth, $admin]);
```

```
// Bookings
```

```
$router->add('GET', '/api/bookings', [new BookingController,'index'], [$cors, $auth]);
```

```
$router->add('POST','/api/bookings', [new BookingController,'store'], [$cors, $auth]);
```

```
$router->add('PUT', '/api/bookings', [new BookingController,'update'], [$cors, $auth]);
```

```
$router->add('PATCH','/api/bookings/status', [new BookingController,'updateStatus'], [$cors, $auth, $admin]);
```

```
$router->add('DELETE','/api/bookings', [new BookingController,'destroy'], [$cors, $auth]);
```

```
// Users
```

```
$router->add('GET','/api/users', [new UserController,'index'], [$cors, $auth]);
```

```
$router->add('PUT','/api/users', [new UserController,'update'], [$cors, $auth]);
```

```
// Admin users
```

```
$router->add('GET','/api/admin/users', [new AdminUserController,'index'], [$cors, $auth, $admin]);
```

```
// Dashboard
```

```
$router->add('GET','/api/analytics/summary', [new DashboardController,'summary'], [$cors, $auth, $admin]);
```

```
$router->add('GET','/api/analytics/services-coverage', [new DashboardController,'serviceCoverage'], [$cors, $auth, $admin]);
```

```
$router->add('GET','/api/analytics/bookings', [new DashboardController,'bookingsList'], [$cors, $auth, $admin]);
```

```
$router->add('GET','/api/analytics/bookings.csv', [new DashboardController,'bookingsCsv'], [$cors, $auth, $admin]);
```

```
$router->add('GET','/api/analytics/right-join-demo', [new DashboardController,'rightJoinDemo'], [$cors, $auth, $admin]);
```

```
// SSE streams
```

```
$router->add('GET','/api/stream/bookings', [new StreamController,'bookings'], [$cors]);
```

