Payal's Kitchen Panel wise Breakdown

Great — here's a clear, final **panel-by-panel flow** showing **who does what** and **who sees what**. Read this as the canonical user-permission map for your app — shareable with the team.

Overall actors

- **Visitor** — not logged in (browses only)
- **Customer** — logged-in user who places subscriptions/orders
- **Employee** — delivery/kitchen staff who execute tasks
- **Admin** — full system manager

1) Visitor (public website)

Who they are: any person opening the site without logging in.
Can see / do:

- Home / About / Contact
- Daily Menu (view only)
- Subscription Plans (view only)
- Event Packages (view only)
- Offers / Promotions (view only)
- FAQs, Terms, Gallery
- "Add to cart", "Checkout", "Rate/Feedback" actions are blocked → prompt to **Login / Register Triggers:**
- Clicks on "Order" or "Checkout" → show modal/login page → if register, becomes Customer.

2) Customer (logged-in)

Who they are: registered users with profile, address, payment methods.
Dashboard / Main screens they see:

- Home / Today's Menu
- My Subscriptions (active, expired, renew)
- My Orders (list + detail + tracking)
- Event Orders (bookings & history)
- Cart & Checkout
- Offers/Coupons (apply)
- Profile & Addresses

- Payment History

- Feedback & Complaints

- Notifications (order updates, reminders, offers)

What they can do (actions):

- Register / Login / Logout

- Buy subscription (choose plan, select start date, pay)

- Place custom one-day order (from daily menu)

- Book event package (select package, enter people, special instructions)

- Add/remove items to cart, apply offer, checkout & pay (Razorpay)

- Track order status in real time (Pending → Confirmed → Preparing → Out for delivery → Delivered)

- Cancel order (if allowed by status/time)

- Give feedback & raise complaints for past orders

- View invoices / download receipts

- Manage profile & addresses

What they see in each state:

- **Before subscription**: full plan details, CTA to subscribe

- **After subscription**: auto-menu visibility, auto-scheduling for daily delivery, renewal reminders

- **After order**: order timeline & assigned employee details (if assigned), delivery ETA, payment receipt

3) Employee (staff / delivery)

Who they are: kitchen staff or delivery personnel with limited system access.
Employee Portal / Main screens:

- Login page

- Today's Assigned Tasks (list)

- Task Detail (customer address, order items, special instructions)

- Order History (their completed tasks)

- Status Update controls (Accept / Picked / En route / Delivered)

- Optional: Upload delivery proof (photo/signature)

What they can do (actions):

- Login using employee account

- View only orders/tasks assigned to them

- Update task status (this updates Customer & Admin notifications)

- Mark task notes (e.g., delay, issue)
- Report delivery problems to Admin

What they see:

- Only assigned orders (not full orders of other employees)
- Customer name, contact, address, order items, time slot, special instructions
- Map / directions (optional integration) or simple address + phone

---

4) Admin (superuser)

Who they are: business owner / manager who controls everything.
Admin Dashboard screens:

- Admin Login
- Dashboard / KPIs (revenue, orders today, active subscriptions)
- Customers (list, search, edit, block/unblock)
- Employees (add/edit/assign/attendance)
- Products / Categories / Daily Menu editor
- Subscription Plans (create/edit/delete)
- Event Packages (create/edit/delete)
- Orders (all orders, search, filter by date/status)
- Assign Tasks (assign orders to employees)
- Offers & Coupons (create, schedule)
- Payments (view / reconcile)
- Feedback & Complaints (view, mark resolved)
- Notifications log (email/SMS history)
- Reports (income, subscriptions, event revenue, delivery performance)

What they can do (actions):

- Full CRUD on products, menus, subscriptions, event packages, offers
- Approve/reject event bookings or custom requests
- Assign employees to orders and monitor status
- Create promotions, set validity and discount rules
- View & export finance reports (payments, refunds)
- Respond to complaints or escalate issues
- Manage admin accounts & roles

What they see:

- Global view of system data: all customers, all employees, all orders, KPIs

- Ability to drill into single customer or order history

- Notifications about failed payments, pending approvals, low stock (if implemented)

Example real-life flows (short)

Flow A — Visitor becomes Customer and places subscription

1. Visitor views subscription plans → clicks "Subscribe" → prompted to register/login

2. Registers (email/phone + password + verify OTP) → becomes Customer

3. Chooses plan, enters address, pays → Subscription active → Daily menu access unlocked

4. Admin receives new subscription notification; system logs payment

Flow B — Customer places one-day custom order

1. Customer picks items from Today's Menu → adds to cart → checkout → pays

2. Order created with status **Pending**

3. Admin confirms or auto-accepts; Admin assigns Employee

4. Employee sees task → updates status through lifecycle → Customer notified at each change

Flow C — Customer books event package

1. Customer selects Event Package → enters number of people, date, address → requests quote and pays

2. Admin reviews (if > threshold), confirms booking, assigns team

3. Event order appears on Admin & Employee dashboards; invoices generated

Permission matrix (quick table)

Visitor	Customer	Employee	Admin
View home/menu/plans	All Visitor rights + place order/subscribe	View assigned tasks	All system control
Cannot checkout	Checkout, pay, feedback, complaints	Update order status, delivery proof	Create/edit products, offers, employees
Cannot track	Track orders & subscription	Cannot create offers or edit menus	Manage users and view reports

Notifications — who receives what

- **Visitor** → prompted to register when trying to order

- **Customer** → on registration, payment success, order status updates, subscription renewal reminders, offers
- **Employee** → assigned task notification, reassignment alerts
- **Admin** → new orders, failed payments, complaints, summary daily report

Short implementation notes for your team

- Enforce **role-based routes** and UI guarding in frontend (React) and backend (Express middleware).
- Use JWT for session + refresh tokens.
- Use MongoDB collections you already defined to filter what each role can access.
- Use websockets / polling for real-time order status, or use notification emails with transactional templates.

If you want, I'll convert this into a **one-page visual flow diagram** (PNG/SVG) that you can paste in your PPT or README so the whole team can instantly understand the flows. Want that?