**Admin Pannel Requirements.**

**🔑 Core Requirements**

1. **Authentication & Authorization**
   * Use JWT (already in backend).
   * Role-based access control (admin, agent, etc.).
   * Admin panel routes should only be accessible to admin.
2. **User Management**
   * View/search all users.
   * Create new users (optional, or only invite).
   * Reset password / deactivate user.
   * Assign roles (admin, agent).
3. **Team Management**
   * CRUD for teams.
   * Manage team\_members (add/remove/change roles).
   * Assign phone numbers to teams.
4. **Telephony & Numbers**
   * View provisioned numbers (phone\_numbers table).
   * Link/unlink numbers to teams.
   * Show metadata (provider, number status).
5. **Call Monitoring**
   * Access calls table:
     + List all calls with filters (team, status, date).
     + View call details (from/to, status, duration).
     + Playback/download recordings (presigned S3 URLs).
6. **Leads Management**
   * CRUD for leads.
   * Search leads by name/phone.
   * See which team owns which leads.
7. **Webhooks & Logs**
   * Monitor Twilio webhooks (success/failure).
   * View call events timeline.
   * Error monitoring from logs.
8. **Security**
   * Admin-only dashboard secured by JWT.
   * Logging of all admin actions (audit trail).
   * Secure file access via presigned S3 URLs.
9. **Analytics (Optional for MVP)**
   * Call volumes per team/user.
   * Agent performance (answered vs missed calls).
   * Recording usage (storage consumption).
   * Lead conversion metrics.
10. **Operations & Settings**

* Environment variables (view only, not editable from UI for safety).
* Observability dashboards (integrate Grafana/ELK or link-out).
* Notifications setup (FCM/APNs keys).

**📊 Tech Choices for Admin Panel**

* **Frontend**: React or Angular.
* **UI Components**: Material UI / Tailwind + shadcn.
* **Backend**: Node.js/Express API.
* **Auth**: JWT stored in httpOnly cookies or local storage.
* **Deployment**: Host on same infra (S3 + CloudFront, Vercel, Netlify).

**✅ MVP Scope for Admin Panel**

If you want to keep it light initially:

1. **Login + Role Check**
2. **Users CRUD**
3. **Teams + Members CRUD**
4. **Calls List + Recording Playback**
5. **Leads CRUD**

**🏗 Project Structure (React + Vite + MUI)**

admin-panel/

 ├── frontend/

 │    ├── package.json

 │    ├── vite.config.js

 │    ├── src/

 │    │    ├── App.jsx

 │    │    ├── main.jsx

 │    │    ├── routes/

 │    │    │    ├── Dashboard.jsx

 │    │    │    ├── Login.jsx

 │    │    │    ├── Users.jsx

 │    │    │    ├── Teams.jsx

 │    │    │    ├── Calls.jsx

 │    │    │    ├── Leads.jsx

 │    │    ├── components/

 │    │    │    ├── Sidebar.jsx

 │    │    │    ├── Navbar.jsx

 │    │    └── lib/

 │    │         └── api.js

**⚙️ Install Dependencies**

# Create project

npm create vite@latest admin-panel --template react

cd admin-panel

# Install deps

npm install @mui/material @mui/icons-material react-router-dom axios

**🌐 Endpoints Mapping**

Your backend already has most endpoints (/auth, /teams, /calls, /leads).  
Admin panel = same API, but with admin role checks.

| **Feature** | **Backend Endpoint** | **Admin UI Screen** |
| --- | --- | --- |
| Auth (login) | POST /auth/login | Login page |
| Users list | GET /users (admin-only) | Users table |
| Create user | POST /auth/register | New User form |
| Assign role | PUT /users/:id/role | Edit User modal |
| Teams list | GET /teams | Teams table |
| Add member | POST /teams/:id/members | Manage team members |
| Remove member | DELETE /teams/:id/members/:userId | Manage team members |
| Phone numbers | GET /teams/:id/numbers | Numbers tab |
| Calls history | GET /calls?teamId=... | Calls dashboard |
| Recording playback | GET /calls/:id/recording | Call detail modal/player |
| Leads list | GET /leads?teamId=... | Leads table |
| Create/Update lead | POST /leads / PUT /leads/:id | Lead form |
| Webhook logs | GET /admin/webhooks (custom) | Logs/Monitoring |