



COMPLETE FRONTEND TECH STACK GUIDE (Claude-Ready)



Project Context

Build a **production-grade, scalable, mobile-first web application frontend** with:

- Type-safe architecture
 - Fast performance
 - Clean UI/UX
 - Secure authentication
 - Dashboard with charts
 - Admin-ready structure
 - India mobile network friendly
-



Core Stack (MANDATORY)

Language

- **TypeScript (strict mode ON)**
Why:
 - Prevent runtime bugs
 - Better refactoring
 - Required for scalability

tsconfig must include:

```
"strict": true,  
"noUncheckedIndexedAccess": true
```

Framework

- **React 18**
- **Vite (latest)**

Why Vite

- Instant dev server
- Fast HMR
- Smaller bundles
- Better than CRA

Styling

- **Tailwind CSS (latest)**
- Tailwind `class` dark mode strategy

Required Tailwind Plugins

@tailwindcss/forms

@tailwindcss/typography

@tailwindcss/container-queries

Routing

- **React Router v6+**

Must support:

- Public routes
- Protected routes

- Role-based routes
 - Lazy loading
-

State Management Strategy

Global State

Use:

- React Context → auth + user
- Avoid Redux initially

Server State

- TanStack Query (React Query latest)

Must handle:

- caching
 - retries
 - background refetch
 - optimistic updates
 - loading states
-

Forms & Validation

- React Hook Form
- Zod

Rules

- All forms use RHF
 - All schemas in [/schemas](#)
 - Zod shared with backend
-

Charts & Visualization

- **Recharts**

Required charts:

- Line chart
 - Bar chart
 - Pie chart
 - Responsive container
-

Icons

- **lucide-react**
-

Animations

- **Framer Motion**

Use only for:

- page transitions
- modal open
- card hover

- button press

Keep animations subtle.

2 Authentication Architecture (VERY IMPORTANT)

Token Strategy

- ✓ JWT in HTTP-only cookies
 - ✓ Refresh token rotation
 - ✓ Silent refresh
 - ✗ NEVER store tokens in localStorage
-

Required Auth Flow

Login → set cookies → fetch /me → store user in context

Auto Refresh Logic

Frontend must:

1. Detect 401
 2. Call `/auth/refresh`
 3. Retry original request
 4. Logout if refresh fails
-

Protected Routes

Must implement:

- `<ProtectedRoute />`
- `<RoleGuard />`

3 Folder Structure (Claude MUST follow)

```
src/
├── api/
│   ├── client.ts
│   ├── auth.ts
│   └── user.ts
├── app/
│   ├── router.tsx
│   └── providers.tsx
├── components/
│   ├── ui/
│   ├── charts/
│   └── layout/
├── features/
│   ├── auth/
│   ├── dashboard/
│   ├── profile/
│   └── admin/
├── hooks/
├── schemas/
├── lib/
├── styles/
├── types/
├── utils/
└── main.tsx
```

⚠ Feature-based architecture is REQUIRED.

4 API Layer Rules

Fetch Wrapper

Create centralized fetch client:

`src/api/client.ts`

Must handle:

- base URL
- credentials include
- JSON parsing
- 401 refresh logic
- error normalization

TanStack Query Setup

Global QueryClient with:

- `retry: 1`
- `staleTime: 30s`
- `gcTime: 5 min`

5 UI/UX Requirements

Design System

- 8px spacing system
 - rounded-2xl cards
 - soft shadows
 - mobile-first
-

Required Pages

Public

- Landing
- Login
- Signup
- Forgot password

Authenticated

- Dashboard
- Profile
- Settings

Admin (role-based)

- Users table
 - Analytics
-

Navigation

Desktop

- Left sidebar
- Top navbar

Mobile

- Top bar
 - Bottom navigation
-

Performance Requirements

Claude must ensure:

- ✓ Route-based code splitting
 - ✓ React.lazy + Suspense
 - ✓ Image optimization
 - ✓ Memo where needed
 - ✓ Avoid unnecessary re-renders
 - ✓ Lighthouse score target > 90
-

Accessibility (MANDATORY)

Must include:

- semantic HTML
 - aria labels
 - keyboard navigation
 - focus states
 - color contrast AA
-

8 Error Handling Strategy

Use:

- `ErrorBoundary`
 - TanStack Query error states
 - Toast notifications
-

9 Testing Stack

- **Vitest**
- **React Testing Library**
- **MSW**
- **Playwright (optional but recommended)**

Coverage target:

- 80% critical paths
-

10 Deployment

Platform

- **Vercel**

Environment Variables

Must support:

`VITE_API_BASE_URL`

1 1 Code Quality

Mandatory:

- ESLint
- Prettier
- TypeScript strict
- Absolute imports
- No **any** allowed

1 2 Nice-to-Have (if Claude is capable)

- PWA support
- Service worker caching
- Storybook
- Theme switcher
- Skeleton loaders
- Optimistic updates

FINAL INSTRUCTION FOR CLAUDE (IMPORTANT)

When you paste into Claude, end with:

Generate the COMPLETE production-ready frontend codebase following the above architecture. Do not simplify. Use best practices. Ensure the app runs with `npm install` & `npm run dev`.