**🧭 COLLABSPACE — COMPLETE PROJECT ROADMAP**

**⚙️ PHASE 1: FOUNDATION (Setup & Architecture)**

Before any feature exists, we must prepare the skeleton of the app.

1. **Project Setup**
   * Create the solution CollabSpace.sln
   * Create projects:
     + CollabSpace.API (ASP.NET Core Web API)
     + CollabSpace.Application (Business logic layer)
     + CollabSpace.Domain (Entities + Enums)
     + CollabSpace.Persistence (EF Core + Database)
   * Add proper references between projects
2. **Database Setup**
   * Configure EF Core
   * Create AppDbContext
   * Add first migration
   * Apply migrations to local database
3. **Authentication & Authorization**
   * Implement **JWT-based login & registration**
   * Add **Role system** (User, Admin)
   * Protect API routes
4. **Base Entities & Infrastructure**
   * User, Workspace, Project, Message
   * Add repository & service interfaces
   * Centralize common response models (e.g. APIResponse)

**💬 PHASE 2: CORE FEATURES (Real Collaboration)**

These bring CollabSpace to life — the essence of teamwork.

1. **User Management**
   * Register, Login, Logout
   * Update Profile, Upload Avatar
   * View other user profiles
2. **Workspace Management**
   * Create / Edit / Delete Workspaces
   * Invite members (by email or username)
   * Assign roles in workspace (Owner, Editor, Viewer)
3. **Project Management**
   * Within each workspace, users can:
     + Create Projects
     + Add Tasks or Notes
     + Assign tasks to users
     + Track progress (status updates)
4. **Real-Time Communication (SignalR)**
   * Create chat rooms per workspace/project
   * Real-time message broadcasting
   * “User is typing…” indicators
   * Online/offline user presence
5. **Notifications**
   * Real-time alerts for:
     + New messages
     + Task assignments
     + Invitations

**🎨 PHASE 3: ENHANCEMENTS (Make It Shine)**

Let’s polish the edges, add performance, and enhance the UX.

1. **File Sharing**
   * Upload attachments in chats or projects
   * Store securely in a folder or cloud (Firebase, Azure Blob)
2. **Activity Logs**
   * Log workspace actions (who created, updated, deleted)
   * Show activity feed for admins
3. **Search & Filtering**
   * Search for users, messages, projects, tasks
   * Add date and status filters
4. **Error Handling & Logging**
   * Global exception middleware
   * Structured logs (e.g. Serilog)
   * Friendly API error responses

**🧱 PHASE 4: DEPLOYMENT READINESS**

Here, we prepare the app for the world.

1. **Environment Configurations**
   * Separate appsettings.Development.json and appsettings.Production.json
   * Secure keys (JWT secret, DB password, etc.)
2. **CORS & HTTPS Setup**
   * Allow your frontend domain
   * Enforce HTTPS redirection
3. **Publishing**
   * Build in Release mode
   * Generate /publish folder
   * Test locally before upload
4. **SmarterASP.NET Deployment**
   * Upload published files
   * Configure connection strings & environment variables
   * Run final tests

**🌠 OPTIONAL (For Mastery Beyond)**

Once you’ve mastered this stage, you can evolve your app:

* Split into **microservices** (UserService, ChatService, ProjectService)
* Add a **Blazor or React Frontend**
* Add **Docker + Jenkins + CI/CD pipeline**
* Deploy on **Azure or AWS ECS**

**🧩 Summary Timeline (Recommended Order)**

| **Phase** | **Name** | **Focus** | **Priority** |
| --- | --- | --- | --- |
| 1 | Foundation | Setup + Auth + DB | 🔥 Must do first |
| 2 | Core Features | Workspaces, Chat, Tasks | 🌟 Main project |
| 3 | Enhancements | Notifications, Logs, Files | 💎 Optional |
| 4 | Deployment | Hosting + Env config | 🚀 Launch phase |