🅰️ Reddit Clone - Arquitectura Frontend Angular

🎯 Tecnologías Frontend

- **Angular 17+** (Standalone Components + Signals)
- **Angular Material** (UI Components)
- **Tailwind CSS** (Utility-first CSS)
- **NgRx** (State Management)
- **RxJS** (Reactive Programming)
- **TypeScript** (Strict mode)

📂 Estructura de Proyecto Angular

```
reddit-clone-frontend/
├── src/
│   ├── app/
│   │   ├── app.component.ts
│   │   ├── app.component.html
│   │   ├── app.component.scss
│   │   ├── app.config.ts
│   │   ├── app.routes.ts
│   │   │
│   │   ├── core/                          # 🔧 Core Module
│   │   │   ├── guards/
│   │   │   │   ├── auth.guard.ts
│   │   │   │   ├── guest.guard.ts
│   │   │   │   └── role.guard.ts
│   │   │   ├── interceptors/
│   │   │   │   ├── auth.interceptor.ts
│   │   │   │   ├── error.interceptor.ts
│   │   │   │   ├── loading.interceptor.ts
│   │   │   │   └── retry.interceptor.ts
│   │   │   ├── services/
│   │   │   │   ├── api.service.ts
│   │   │   │   ├── auth.service.ts
│   │   │   │   ├── loading.service.ts
│   │   │   │   ├── error.service.ts
│   │   │   │   ├── theme.service.ts
│   │   │   │   ├── websocket.service.ts
│   │   │   │   └── storage.service.ts
│   │   │   ├── models/
│   │   │   │   ├── api-response.interface.ts
│   │   │   │   ├── user.interface.ts
│   │   │   │   ├── auth.interface.ts
│   │   │   │   ├── pagination.interface.ts
│   │   │   │   └── error.interface.ts
│   │   │   └── utils/
│   │   │       ├── validators.ts
│   │   │       ├── date.utils.ts
│   │   │       ├── url.utils.ts
│   │   │       └── karma.utils.ts
│   │   │
│   │   ├── shared/                        # 🔄 Shared Module
│   │   │   ├── components/
│   │   │   │   ├── header/
│   │   │   │   │   ├── header.component.ts
│   │   │   │   │   ├── header.component.html
│   │   │   │   │   ├── header.component.scss
│   │   │   │   │   └── profile-dropdown/
```

```
│   │   │   │   │   ├── profile-dropdown.component.ts
│   │   │   │   │   ├── profile-dropdown.component.html
│   │   │   │   │   └── profile-dropdown.component.scss
│   │   │   │   ├── sidebar/
│   │   │   │   │   ├── sidebar.component.ts
│   │   │   │   │   ├── sidebar.component.html
│   │   │   │   │   ├── sidebar.component.scss
│   │   │   │   │   ├── community-list/
│   │   │   │   │   │   ├── community-list.component.ts
│   │   │   │   │   │   ├── community-list.component.html
│   │   │   │   │   │   └── community-list.component.scss
│   │   │   │   │   └── trending-topics/
│   │   │   │   │       ├── trending-topics.component.ts
│   │   │   │   │       ├── trending-topics.component.html
│   │   │   │   │       └── trending-topics.component.scss
│   │   │   │   ├── post-card/
│   │   │   │   │   ├── post-card.component.ts
│   │   │   │   │   ├── post-card.component.html
│   │   │   │   │   ├── post-card.component.scss
│   │   │   │   │   └── vote-buttons/
│   │   │   │   │       ├── vote-buttons.component.ts
│   │   │   │   │       ├── vote-buttons.component.html
│   │   │   │   │       └── vote-buttons.component.scss
│   │   │   │   ├── comment-tree/
│   │   │   │   │   ├── comment-tree.component.ts
│   │   │   │   │   ├── comment-tree.component.html
│   │   │   │   │   ├── comment-tree.component.scss
│   │   │   │   │   └── comment-item/
│   │   │   │   │       ├── comment-item.component.ts
│   │   │   │   │       ├── comment-item.component.html
│   │   │   │   │       └── comment-item.component.scss
│   │   │   │   ├── loading-spinner/
│   │   │   │   │   ├── loading-spinner.component.ts
│   │   │   │   │   ├── loading-spinner.component.html
│   │   │   │   │   └── loading-spinner.component.scss
│   │   │   │   ├── error-message/
│   │   │   │   │   ├── error-message.component.ts
│   │   │   │   │   ├── error-message.component.html
│   │   │   │   │   └── error-message.component.scss
│   │   │   │   ├── infinite-scroll/
│   │   │   │   │   ├── infinite-scroll.component.ts
│   │   │   │   │   ├── infinite-scroll.component.html
│   │   │   │   │   └── infinite-scroll.component.scss
│   │   │   │   ├── rich-text-editor/
│   │   │   │   │   ├── rich-text-editor.component.ts
│   │   │   │   │   ├── rich-text-editor.component.html
│   │   │   │   │   └── rich-text-editor.component.scss
```

```
│   │   │   │   └── image-upload/
│   │   │   │       ├── image-upload.component.ts
│   │   │   │       ├── image-upload.component.html
│   │   │   │       └── image-upload.component.scss
│   │   │   ├── directives/
│   │   │   │   ├── lazy-load.directive.ts
│   │   │   │   ├── auto-resize.directive.ts
│   │   │   │   ├── click-outside.directive.ts
│   │   │   │   └── highlight.directive.ts
│   │   │   ├── pipes/
│   │   │   │   ├── time-ago.pipe.ts
│   │   │   │   ├── karma-format.pipe.ts
│   │   │   │   ├── truncate.pipe.ts
│   │   │   │   ├── safe-html.pipe.ts
│   │   │   │   └── member-count.pipe.ts
│   │   │   └── layouts/
│   │   │       ├── main-layout/
│   │   │       │   ├── main-layout.component.ts
│   │   │       │   ├── main-layout.component.html
│   │   │       │   └── main-layout.component.scss
│   │   │       └── auth-layout/
│   │   │           ├── auth-layout.component.ts
│   │   │           ├── auth-layout.component.html
│   │   │           └── auth-layout.component.scss
│   │   │
│   │   ├── features/                    # 🎯 Feature Modules
│   │   │   │
│   │   │   ├── auth/                    # 💪 Auth Module
│   │   │   │   ├── components/
│   │   │   │   │   ├── login/
│   │   │   │   │   │   ├── login.component.ts
│   │   │   │   │   │   ├── login.component.html
│   │   │   │   │   │   └── login.component.scss
│   │   │   │   │   ├── signup/
│   │   │   │   │   │   ├── signup.component.ts
│   │   │   │   │   │   ├── signup.component.html
│   │   │   │   │   │   └── signup.component.scss
│   │   │   │   │   ├── forgot-password/
│   │   │   │   │   │   ├── forgot-password.component.ts
│   │   │   │   │   │   ├── forgot-password.component.html
│   │   │   │   │   │   └── forgot-password.component.scss
│   │   │   │   │   └── oauth-callback/
│   │   │   │   │       ├── oauth-callback.component.ts
│   │   │   │   │       ├── oauth-callback.component.html
│   │   │   │   │       └── oauth-callback.component.scss
│   │   │   │   ├── services/
│   │   │   │   │   └── auth-api.service.ts
```

```
│   │   │   ├── models/
│   │   │   │   ├── login.interface.ts
│   │   │   │   ├── signup.interface.ts
│   │   │   │   └── auth-response.interface.ts
│   │   │   └── auth.routes.ts
│   │   │
│   │   ├── home/                    # 🏠 Home Module
│   │   │   ├── components/
│   │   │   │   ├── home-feed/
│   │   │   │   │   ├── home-feed.component.ts
│   │   │   │   │   ├── home-feed.component.html
│   │   │   │   │   └── home-feed.component.scss
│   │   │   │   ├── create-post-widget/
│   │   │   │   │   ├── create-post-widget.component.ts
│   │   │   │   │   ├── create-post-widget.component.html
│   │   │   │   │   └── create-post-widget.component.scss
│   │   │   │   ├── post-list/
│   │   │   │   │   ├── post-list.component.ts
│   │   │   │   │   ├── post-list.component.html
│   │   │   │   │   └── post-list.component.scss
│   │   │   │   └── feed-filters/
│   │   │   │       ├── feed-filters.component.ts
│   │   │   │       ├── feed-filters.component.html
│   │   │   │       └── feed-filters.component.scss
│   │   │   ├── services/
│   │   │   │   └── home-api.service.ts
│   │   │   └── home.routes.ts
│   │   │
│   │   ├── post/                    # 📝 Post Module
│   │   │   ├── components/
│   │   │   │   ├── post-detail/
│   │   │   │   │   ├── post-detail.component.ts
│   │   │   │   │   ├── post-detail.component.html
│   │   │   │   │   └── post-detail.component.scss
│   │   │   │   ├── create-post/
│   │   │   │   │   ├── create-post.component.ts
│   │   │   │   │   ├── create-post.component.html
│   │   │   │   │   ├── create-post.component.scss
│   │   │   │   │   ├── post-type-selector/
│   │   │   │   │   │   ├── post-type-selector.component.ts
│   │   │   │   │   │   ├── post-type-selector.component.html
│   │   │   │   │   │   └── post-type-selector.component.scss
│   │   │   │   │   ├── community-selector/
│   │   │   │   │   │   ├── community-selector.component.ts
│   │   │   │   │   │   ├── community-selector.component.html
│   │   │   │   │   │   └── community-selector.component.scss
│   │   │   │   │   └── tag-selector/
```

```
│   │   │   │   │   │   ├── tag-selector.component.ts
│   │   │   │   │   │   ├── tag-selector.component.html
│   │   │   │   │   │   └── tag-selector.component.scss
│   │   │   │   │   ├── post-actions/
│   │   │   │   │   │   ├── post-actions.component.ts
│   │   │   │   │   │   ├── post-actions.component.html
│   │   │   │   │   │   └── post-actions.component.scss
│   │   │   │   │   └── post-content/
│   │   │   │   │       ├── post-content.component.ts
│   │   │   │   │       ├── post-content.component.html
│   │   │   │   │       └── post-content.component.scss
│   │   │   │   ├── services/
│   │   │   │   │   └── post-api.service.ts
│   │   │   │   ├── models/
│   │   │   │   │   ├── post.interface.ts
│   │   │   │   │   ├── create-post.interface.ts
│   │   │   │   │   └── post-flair.interface.ts
│   │   │   │   └── post.routes.ts
│   │   │   │
│   │   │   ├── community/                  # 🏠 Community Module
│   │   │   │   ├── components/
│   │   │   │   │   ├── community-page/
│   │   │   │   │   │   ├── community-page.component.ts
│   │   │   │   │   │   ├── community-page.component.html
│   │   │   │   │   │   └── community-page.component.scss
│   │   │   │   │   ├── community-header/
│   │   │   │   │   │   ├── community-header.component.ts
│   │   │   │   │   │   ├── community-header.component.html
│   │   │   │   │   │   └── community-header.component.scss
│   │   │   │   │   ├── community-nav/
│   │   │   │   │   │   ├── community-nav.component.ts
│   │   │   │   │   │   ├── community-nav.component.html
│   │   │   │   │   │   └── community-nav.component.scss
│   │   │   │   │   ├── community-posts/
│   │   │   │   │   │   ├── community-posts.component.ts
│   │   │   │   │   │   ├── community-posts.component.html
│   │   │   │   │   │   └── community-posts.component.scss
│   │   │   │   │   ├── community-about/
│   │   │   │   │   │   ├── community-about.component.ts
│   │   │   │   │   │   ├── community-about.component.html
│   │   │   │   │   │   └── community-about.component.scss
│   │   │   │   │   ├── community-rules/
│   │   │   │   │   │   ├── community-rules.component.ts
│   │   │   │   │   │   ├── community-rules.component.html
│   │   │   │   │   │   └── community-rules.component.scss
│   │   │   │   │   ├── community-moderators/
│   │   │   │   │   │   ├── community-moderators.component.ts
```

```
│   │   │   │   │   ├── community-moderators.component.html
│   │   │   │   │   └── community-moderators.component.scss
│   │   │   │   ├── community-sidebar/
│   │   │   │   │   ├── community-sidebar.component.ts
│   │   │   │   │   ├── community-sidebar.component.html
│   │   │   │   │   └── community-sidebar.component.scss
│   │   │   │   └── create-community/
│   │   │   │       ├── create-community.component.ts
│   │   │   │       ├── create-community.component.html
│   │   │   │       └── create-community.component.scss
│   │   │   ├── services/
│   │   │   │   └── community-api.service.ts
│   │   │   ├── models/
│   │   │   │   ├── community.interface.ts
│   │   │   │   ├── community-stats.interface.ts
│   │   │   │   ├── community-rule.interface.ts
│   │   │   │   └── moderator.interface.ts
│   │   │   └── community.routes.ts
│   │   │
│   │   ├── user/                    # 👤 User Module
│   │   │   ├── components/
│   │   │   │   ├── user-profile/
│   │   │   │   │   ├── user-profile.component.ts
│   │   │   │   │   ├── user-profile.component.html
│   │   │   │   │   └── user-profile.component.scss
│   │   │   │   ├── profile-header/
│   │   │   │   │   ├── profile-header.component.ts
│   │   │   │   │   ├── profile-header.component.html
│   │   │   │   │   └── profile-header.component.scss
│   │   │   │   ├── profile-nav/
│   │   │   │   │   ├── profile-nav.component.ts
│   │   │   │   │   ├── profile-nav.component.html
│   │   │   │   │   └── profile-nav.component.scss
│   │   │   │   ├── profile-posts/
│   │   │   │   │   ├── profile-posts.component.ts
│   │   │   │   │   ├── profile-posts.component.html
│   │   │   │   │   └── profile-posts.component.scss
│   │   │   │   ├── profile-comments/
│   │   │   │   │   ├── profile-comments.component.ts
│   │   │   │   │   ├── profile-comments.component.html
│   │   │   │   │   └── profile-comments.component.scss
│   │   │   │   ├── profile-saved/
│   │   │   │   │   ├── profile-saved.component.ts
│   │   │   │   │   ├── profile-saved.component.html
│   │   │   │   │   └── profile-saved.component.scss
│   │   │   │   ├── profile-awards/
│   │   │   │   │   ├── profile-awards.component.ts
```

```
│   │   │   │   │   ├── profile-awards.component.html
│   │   │   │   │   └── profile-awards.component.scss
│   │   │   │   ├── profile-stats/
│   │   │   │   │   ├── profile-stats.component.ts
│   │   │   │   │   ├── profile-stats.component.html
│   │   │   │   │   └── profile-stats.component.scss
│   │   │   │   └── profile-sidebar/
│   │   │   │       ├── profile-sidebar.component.ts
│   │   │   │       ├── profile-sidebar.component.html
│   │   │   │       └── profile-sidebar.component.scss
│   │   │   ├── services/
│   │   │   │   └── user-api.service.ts
│   │   │   ├── models/
│   │   │   │   ├── user-profile.interface.ts
│   │   │   │   ├── user-stats.interface.ts
│   │   │   │   └── user-award.interface.ts
│   │   │   └── user.routes.ts
│   │   │
│   │   ├── settings/                    # ⚙️ Settings Module
│   │   │   ├── components/
│   │   │   │   ├── settings-page/
│   │   │   │   │   ├── settings-page.component.ts
│   │   │   │   │   ├── settings-page.component.html
│   │   │   │   │   └── settings-page.component.scss
│   │   │   │   ├── settings-sidebar/
│   │   │   │   │   ├── settings-sidebar.component.ts
│   │   │   │   │   ├── settings-sidebar.component.html
│   │   │   │   │   └── settings-sidebar.component.scss
│   │   │   │   ├── account-settings/
│   │   │   │   │   ├── account-settings.component.ts
│   │   │   │   │   ├── account-settings.component.html
│   │   │   │   │   └── account-settings.component.scss
│   │   │   │   ├── privacy-settings/
│   │   │   │   │   ├── privacy-settings.component.ts
│   │   │   │   │   ├── privacy-settings.component.html
│   │   │   │   │   └── privacy-settings.component.scss
│   │   │   │   ├── notification-settings/
│   │   │   │   │   ├── notification-settings.component.ts
│   │   │   │   │   ├── notification-settings.component.html
│   │   │   │   │   └── notification-settings.component.scss
│   │   │   │   ├── theme-settings/
│   │   │   │   │   ├── theme-settings.component.ts
│   │   │   │   │   ├── theme-settings.component.html
│   │   │   │   │   └── theme-settings.component.scss
│   │   │   │   └── avatar-editor/
│   │   │   │       ├── avatar-editor.component.ts
│   │   │   │       ├── avatar-editor.component.html
```

```
│   │   │   │   │   └── avatar-editor.component.scss
│   │   │   │   ├── services/
│   │   │   │   │   └── settings-api.service.ts
│   │   │   │   ├── models/
│   │   │   │   │   ├── user-settings.interface.ts
│   │   │   │   │   ├── notification-settings.interface.ts
│   │   │   │   │   └── privacy-settings.interface.ts
│   │   │   │   └── settings.routes.ts
│   │   │   │
│   │   │   └── comment/                    # 💬 Comment Module
│   │   │       ├── components/
│   │   │       │   ├── comment-section/
│   │   │       │   │   ├── comment-section.component.ts
│   │   │       │   │   ├── comment-section.component.html
│   │   │       │   │   └── comment-section.component.scss
│   │   │       │   ├── comment-form/
│   │   │       │   │   ├── comment-form.component.ts
│   │   │       │   │   ├── comment-form.component.html
│   │   │       │   │   └── comment-form.component.scss
│   │   │       │   └── comment-sort/
│   │   │       │       ├── comment-sort.component.ts
│   │   │       │       ├── comment-sort.component.html
│   │   │       │       └── comment-sort.component.scss
│   │   │       ├── services/
│   │   │       │   └── comment-api.service.ts
│   │   │       ├── models/
│   │   │       │   ├── comment.interface.ts
│   │   │       │   └── comment-tree.interface.ts
│   │   │       └── comment.routes.ts
│   │   │
│   │   ├── store/                          # 🗄️ NgRx Store
│   │   │   ├── auth/
│   │   │   │   ├── auth.actions.ts
│   │   │   │   ├── auth.reducer.ts
│   │   │   │   ├── auth.selectors.ts
│   │   │   │   ├── auth.effects.ts
│   │   │   │   └── auth.state.ts
│   │   │   ├── user/
│   │   │   │   ├── user.actions.ts
│   │   │   │   ├── user.reducer.ts
│   │   │   │   ├── user.selectors.ts
│   │   │   │   ├── user.effects.ts
│   │   │   │   └── user.state.ts
│   │   │   ├── post/
│   │   │   │   ├── post.actions.ts
│   │   │   │   ├── post.reducer.ts
│   │   │   │   ├── post.selectors.ts
```

```
│   │   │   │   ├── post.effects.ts
│   │   │   │   └── post.state.ts
│   │   │   ├── community/
│   │   │   │   ├── community.actions.ts
│   │   │   │   ├── community.reducer.ts
│   │   │   │   ├── community.selectors.ts
│   │   │   │   ├── community.effects.ts
│   │   │   │   └── community.state.ts
│   │   │   ├── comment/
│   │   │   │   ├── comment.actions.ts
│   │   │   │   ├── comment.reducer.ts
│   │   │   │   ├── comment.selectors.ts
│   │   │   │   ├── comment.effects.ts
│   │   │   │   └── comment.state.ts
│   │   │   ├── ui/
│   │   │   │   ├── ui.actions.ts
│   │   │   │   ├── ui.reducer.ts
│   │   │   │   ├── ui.selectors.ts
│   │   │   │   └── ui.state.ts
│   │   │   ├── app.reducer.ts
│   │   │   └── app.state.ts
│   │   │
│   │   └── environments/
│   │       ├── environment.ts
│   │       ├── environment.prod.ts
│   │       └── environment.staging.ts
│   │
│   ├── assets/
│   │   ├── icons/
│   │   ├── images/
│   │   ├── fonts/
│   │   └── scss/
│   │       ├── _variables.scss
│   │       ├── _mixins.scss
│   │       ├── _themes.scss
│   │       └── main.scss
│   │
│   ├── styles.scss
│   ├── main.ts
│   └── index.html
│
├── angular.json
├── package.json
├── tailwind.config.js
└── tsconfig.json
```

```
├── tsconfig.app.json
└── README.md
```

📬 Routing Structure

Main Routes (app.routes.ts)

typescript

```typescript
export const routes: Routes = [
  {
    path: '',
    component: MainLayoutComponent,
    children: [
      { path: '', redirectTo: '/home', pathMatch: 'full' },
      { path: 'home', loadChildren: () => import('./features/home/home.routes') },
      { path: 'r/:communityName', loadChildren: () => import('./features/community/community.routes') },
      { path: 'user/:username', loadChildren: () => import('./features/user/user.routes') },
      { path: 'post/:id', loadChildren: () => import('./features/post/post.routes') },
      { path: 'submit', loadChildren: () => import('./features/post/post.routes') },
      { path: 'settings', loadChildren: () => import('./features/settings/settings.routes'), canActivate: [AuthGuard] },
    ]
  },
  {
    path: 'auth',
    component: AuthLayoutComponent,
    children: [
      { path: '', loadChildren: () => import('./features/auth/auth.routes') }
    ]
  },
  { path: '**', redirectTo: '/home' }
];
```

Community Routes (community.routes.ts)

typescript

```typescript
export const COMMUNITY_ROUTES: Routes = [
  {
    path: '',
    component: CommunityPageComponent,
    children: [
      { path: '', redirectTo: 'posts', pathMatch: 'full' },
      { path: 'posts', component: CommunityPostsComponent },
      { path: 'about', component: CommunityAboutComponent },
      { path: 'rules', component: CommunityRulesComponent },
      { path: 'moderators', component: CommunityModeratorsComponent }
    ]
  }
];
```

User Profile Routes (user.routes.ts)

typescript

```typescript
export const USER_ROUTES: Routes = [
  {
    path: '',
    component: UserProfileComponent,
    children: [
      { path: '', redirectTo: 'posts', pathMatch: 'full' },
      { path: 'posts', component: ProfilePostsComponent },
      { path: 'comments', component: ProfileCommentsComponent },
      { path: 'saved', component: ProfileSavedComponent, canActivate: [AuthGuard] },
      { path: 'awards', component: ProfileAwardsComponent }
    ]
  }
];
```

Settings Routes (settings.routes.ts)

typescript

```typescript
export const SETTINGS_ROUTES: Routes = [
  {
    path: '',
    component: SettingsPageComponent,
    children: [
      { path: '', redirectTo: 'account', pathMatch: 'full' },
      { path: 'account', component: AccountSettingsComponent },
      { path: 'privacy', component: PrivacySettingsComponent },
      { path: 'notifications', component: NotificationSettingsComponent },
      { path: 'theme', component: ThemeSettingsComponent }
    ]
  }
];
```

🎨 Component Mapping to SVG Designs

1. Home Feed (home-feed.component)

SVG Plantilla: reddit_home_design

typescript

```typescript
@Component({
  selector: 'app-home-feed',
  standalone: true,
  imports: [CommonModule, PostListComponent, CreatePostWidgetComponent, SidebarComponent],
  template: `
    <div class="flex min-h-screen bg-gray-900">
      <!-- Sidebar Left -->
      <app-sidebar class="w-64 fixed left-0 top-16"></app-sidebar>

      <!-- Main Content -->
      <main class="flex-1 ml-64 mr-80 p-4">
        <app-create-post-widget class="mb-4"></app-create-post-widget>
        <app-post-list [posts]="posts$ | async"></app-post-list>
      </main>

      <!-- Sidebar Right -->
      <aside class="w-80 fixed right-0 top-16 p-4">
        <app-trending-topics></app-trending-topics>
      </aside>
    </div>
  `
})
```

2. Post Detail (post-detail.component)

SVG Plantilla: reddit_post_detail

typescript

```
@Component({
  selector: 'app-post-detail',
  standalone: true,
  imports: [CommonModule, PostContentComponent, CommentSectionComponent, VoteButtonsComponent],
  template: `
    <div class="flex min-h-screen bg-gray-900">
      <main class="flex-1 max-w-4xl mx-auto p-4">
        <!-- Breadcrumb -->
        <nav class="mb-4 text-gray-400">
          <a [routerLink]="['/r', post.communityName]">r/{{post.communityName}}</a> > {{post.title}}
        </nav>

        <!-- Post Content -->
        <article class="bg-gray-800 rounded-lg p-6 mb-6">
          <div class="flex">
            <app-vote-buttons [targetId]="post.id" [voteCount]="post.voteCount"></app-vote-buttons>
            <app-post-content [post]="post" [showFullContent]="true"></app-post-content>
          </div>
        </article>

        <!-- Comments -->
        <app-comment-section [postId]="post.id"></app-comment-section>
      </main>

      <!-- Sidebar -->
      <aside class="w-80 p-4">
        <app-community-sidebar [communityName]="post.communityName"></app-community-sidebar>
      </aside>
    </div>
  `
})
```

3. Create Post (create-post.component)

SVG Plantilla: reddit_create_post

typescript

```
@Component({
  selector: 'app-create-post',
  standalone: true,
  imports: [CommonModule, ReactiveFormsModule, RichTextEditorComponent, CommunitySelector],
  template: `
    <div class="min-h-screen bg-gray-900">
      <header class="bg-gray-900 border-b border-gray-700 p-4">
        <h1 class="text-2xl font-bold text-white">Create a post</h1>
      </header>

      <div class="flex max-w-6xl mx-auto p-4 gap-6">
        <main class="flex-1">
          <form [formGroup]="postForm" (ngSubmit)="onSubmit()" class="bg-gray-800 rounded-lg p-6">
            <!-- Community Selection -->
            <app-community-selector formControlName="communityId"></app-community-selector>

            <!-- Post Type Tabs -->
            <div class="flex gap-2 my-4">
              <button type="button"
                      [class.bg-blue-600]="postType === 'text'"
                      class="px-4 py-2 rounded-lg border">📝 Post</button>
              <button type="button"
                      [class.bg-blue-600]="postType === 'image'"
                      class="px-4 py-2 rounded-lg border">🖼️ Image</button>
              <button type="button"
                      [class.bg-blue-600]="postType === 'link'"
                      class="px-4 py-2 rounded-lg border">🔗 Link</button>
            </div>

            <!-- Title Input -->
            <input formControlName="title"
                   placeholder="Title"
                   class="w-full p-3 bg-gray-700 rounded-lg mb-4">

            <!-- Content Editor -->
            <app-rich-text-editor formControlName="content"></app-rich-text-editor>

            <!-- Tags -->
            <app-tag-selector formControlName="tags"></app-tag-selector>

            <!-- Submit -->
            <div class="flex justify-end gap-4 mt-6">
              <button type="button" class="px-6 py-2 border border-gray-600 rounded-lg">Save Draft</button>
              <button type="submit" class="px-6 py-2 bg-blue-600 text-white rounded-lg">Post</button>
            </div>
          </form>
```

```
        </main>

        <!-- Sidebar with posting guidelines -->
        <aside class="w-80">
          <div class="bg-gray-800 rounded-lg p-4">
            <h3 class="font-bold text-white mb-4">Posting to Reddit</h3>
            <!-- Guidelines content -->
          </div>
        </aside>
      </div>
    </div>
  `
})
```

4. Login Page (login.component)

SVG Plantilla: reddit_login_page

typescript

```
@Component({
 selector: 'app-login',
 standalone: true,
 imports: [CommonModule, ReactiveFormsModule],
 template: `
  <div class="min-h-screen bg-black flex">
   <!-- Left side with features -->
   <div class="flex-1 flex flex-col justify-center items-center p-8">
    <h1 class="text-4xl font-bold text-orange-500 mb-4">Join the conversation</h1>
    <p class="text-gray-300 text-center mb-8">Connect with communities around your interests</p>

    <div class="space-y-4 w-full max-w-md">
     <div class="bg-gray-800 rounded-lg p-4 flex items-center">
      <div class="w-8 h-8 bg-green-500 rounded-full mr-3"></div>
      <div>
       <p class="text-white font-medium">Share your knowledge</p>
       <p class="text-gray-400 text-sm">Post and comment in communities</p>
      </div>
     </div>
     <!-- More features... -->
    </div>
   </div>

   <!-- Right side with login form -->
   <div class="flex items-center justify-center p-8">
    <div class="w-full max-w-md bg-gray-800 rounded-2xl p-8">
     <div class="text-center mb-6">
      <div class="w-16 h-16 bg-orange-500 rounded-full mx-auto mb-4 flex items-center justify-center">
       <span class="text-white text-2xl font-bold">R</span>
      </div>
      <h2 class="text-2xl font-bold text-white">Welcome back</h2>
      <p class="text-gray-400">Sign in to your account</p>
     </div>

     <form [formGroup]="loginForm" (ngSubmit)="onSubmit()">
      <div class="space-y-4">
       <input formControlName="email"
              type="email"
              placeholder="Email or Username"
              class="w-full p-3 bg-gray-700 rounded-lg text-white">

       <input formControlName="password"
              type="password"
              placeholder="Password"
              class="w-full p-3 bg-gray-700 rounded-lg text-white">
```

```
          <div class="flex items-center justify-between">
            <label class="flex items-center text-gray-300">
              <input type="checkbox" class="mr-2">
              Remember me
            </label>
            <a href="#" class="text-blue-400 text-sm">Forgot password?</a>
          </div>

          <button type="submit" class="w-full bg-blue-600 text-white py-3 rounded-lg font-medium">
            Log In
          </button>

          <div class="text-center">
            <span class="text-gray-400">OR</span>
          </div>

          <button type="button" class="w-full border border-gray-600 text-white py-3 rounded-lg flex items-center just
            <span class="mr-2">G</span> Continue with Google
          </button>
        </div>
      </form>

      <p class="text-center text-gray-400 mt-6">
        Don't have an account?
        <a routerLink="/auth/signup" class="text-blue-400 font-medium">Sign Up</a>
      </p>
    </div>
  </div>
</div>
`

})
```

🔧 Core Services

Auth Service

typescript

```typescript
@Injectable({ providedIn: 'root' })
export class AuthService {
  private readonly API_URL = environment.apiUrl;
  private currentUserSubject = new BehaviorSubject<User | null>(null);
  public currentUser$ = this.currentUserSubject.asObservable();

  constructor(private http: HttpClient, private router: Router) {
    // Check for stored user on app init
    const storedUser = localStorage.getItem('currentUser');
    if (storedUser) {
      this.currentUserSubject.next(JSON.parse(storedUser));
    }
  }

  login(credentials: LoginRequest): Observable<AuthResponse> {
    return this.http.post<AuthResponse>(`${this.API_URL}/auth/login`, credentials)
      .pipe(
        tap(response => {
          localStorage.setItem('token', response.token);
          localStorage.setItem('currentUser', JSON.stringify(response.user));
          this.currentUserSubject.next(response.user);
        })
      );
  }

  signup(userData: SignupRequest): Observable<AuthResponse> {
    return this.http.post<AuthResponse>(`${this.API_URL}/auth/signup`, userData);
  }

  logout(): void {
    localStorage.removeItem('token');
    localStorage.removeItem('currentUser');
    this.currentUserSubject.next(null);
    this.router.navigate(['/auth/login']);
  }

  isAuthenticated(): boolean {
    return !!localStorage.getItem('token');
  }

  getCurrentUser(): User | null {
    return this.currentUserSubject.value;
  }
}
```

API Service

typescript

```typescript
@Injectable({ providedIn: 'root' })
export class ApiService {
  private readonly API_URL = environment.apiUrl;

  constructor(private http: HttpClient) {}

  // Generic HTTP methods
  get<T>(endpoint: string, params?: HttpParams): Observable<T> {
    return this.http.get<T>(`${this.API_URL}${endpoint}`, { params });
  }

  post<T>(endpoint: string, body: any): Observable<T> {
    return this.http.post<T>(`${this.API_URL}${endpoint}`, body);
  }

  put<T>(endpoint: string, body: any): Observable<T> {
    return this.http.put<T>(`${this.API_URL}${endpoint}`, body);
  }

  delete<T>(endpoint: string): Observable<T> {
    return this.http.delete<T>(`${this.API_URL}${endpoint}`);
  }

  // Specific API methods
  getPosts(page: number = 0, size: number = 10): Observable<PagedResponse<Post>> {
    const params = new HttpParams()
      .set('page', page.toString())
      .set('size', size.toString());
    return this.get<PagedResponse<Post>>('/posts', params);
  }

  getPost(id: string): Observable<Post> {
    return this.get<Post>(`/posts/${id}`);
  }

  createPost(post: CreatePostRequest): Observable<Post> {
    return this.post<Post>('/posts', post);
  }

  // Community methods
  getCommunity(name: string): Observable<Community> {
    return this.get<Community>(`/communities/${name}`);
  }

  getCommunityPosts(name: string, page: number = 0): Observable<PagedResponse<Post>> {
    const params = new HttpParams()
```

```
      .set('page', page.toString())
      .set('size', '10');
    return this.get<PagedResponse<Post>>(`/communities/${name}/posts`, params);
  }

  // User methods
  getUserProfile(username: string): Observable<UserProfile> {
    return this.get<UserProfile>(`/users/${username}`);
  }

  getUserPosts(username: string, page: number = 0): Observable<PagedResponse<Post>> {
    const params = new HttpParams()
      .set('page', page.toString())
      .set('size', '10');
    return this.get<PagedResponse<Post>>(`/users/${username}/posts`, params);
  }

  // Vote methods
  vote(targetId: string, targetType: 'post' | 'comment', voteType: 'upvote' | 'downvote'): Observable<VoteResponse> {
    return this.post<VoteResponse>('/votes', { targetId, targetType, voteType });
  }

  // Comment methods
  getComments(postId: string): Observable<Comment[]> {
    return this.get<Comment[]>(`/posts/${postId}/comments`);
  }

  createComment(comment: CreateCommentRequest): Observable<Comment> {
    return this.post<Comment>('/comments', comment);
  }
}
```

🎨 Tailwind Configuration

tailwind.config.js

javascript

```javascript
module.exports = {
  content: ['./src/**/*.{html,ts}'],
  darkMode: 'class',
  theme: {
    extend: {
      colors: {
        reddit: {
          orange: '#ff4500',
          blue: '#0079d3',
          dark: '#1a1a1b',
          gray: {
            100: '#f8f9fa',
            200: '#edeff1',
            300: '#d7dadc',
            400: '#878a8c',
            500: '#818384',
            600: '#343536',
            700: '#272729',
            800: '#1a1a1b',
            900: '#030303'
          }
        }
      },
      fontFamily: {
        sans: ['Inter', 'system-ui', 'sans-serif']
      },
      spacing: {
        '18': '4.5rem',
        '88': '22rem'
      }
    }
  },
  plugins: [
    require('@tailwindcss/forms'),
    require('@tailwindcss/typography')
  ]
}
```

📱 Responsive Design Strategy

Breakpoints

scss

```scss
// Mobile-first approach
$mobile: 640px;    // sm
$tablet: 768px;    // md
$desktop: 1024px;  // lg
$large: 1280px;    // xl

// Layout adjustments
@media (max-width: $tablet) {
  .sidebar { display: none; }
  .main-content { margin: 0; width: 100%; }
  .mobile-nav { display: block; }
}

@media (max-width: $mobile) {
  .post-card { padding: 1rem; }
  .header { height: 3rem; }
  .vote-buttons {
    flex-direction: row;
    gap: 0.5rem;
  }
}
```

🔄 State Management with NgRx

App State Structure

typescript

```typescript
export interface AppState {
  auth: AuthState;
  user: UserState;
  posts: PostState;
  communities: CommunityState;
  comments: CommentState;
  ui: UIState;
}

export interface AuthState {
  user: User | null;
  token: string | null;
  isAuthenticated: boolean;
  loading: boolean;
  error: string | null;
}

export interface PostState {
  posts: Post[];
  currentPost: Post | null;
  loading: boolean;
  error: string | null;
  pagination: {
    page: number;
    totalPages: number;
    hasMore: boolean;
  };
}
```

🧪 Testing Strategy

Component Testing Example

typescript

```typescript
describe('PostCardComponent', () => {
  let component: PostCardComponent;
  let fixture: ComponentFixture<PostCardComponent>;

  beforeEach(async () => {
    await TestBed.configureTestingModule({
      imports: [PostCardComponent, HttpClientTestingModule],
      providers: [provideMockStore()]
    }).compileComponents();

    fixture = TestBed.createComponent(PostCardComponent);
    component = fixture.componentInstance;
  });

  it('should display post content correctly', () => {
    const mockPost: Post = {
      id: '1',
      title: 'Test Post',
      content: 'Test Content',
      voteCount: 10,
      // ... more properties
    };

    component.post = mockPost;
    fixture.detectChanges();

    expect(fixture.nativeElement.querySelector('.post-title').textContent).toBe('Test Post');
  });
});
```

📋 Development Phases

Fase 1: Core Structure (Semanas 1-2)

1. **Project setup** con Angular 17, Tailwind, NgRx
2. **Auth layout** + **Login/Signup** components
3. **Main layout** + **Header** + **Sidebar**
4. **Basic routing** y guards

Fase 2: Home & Posts (Semanas 3-4)

5. **Home feed** component
6. **Post card** component con voting

7. **Post detail** page
8. **Create post** form

Fase 3: Communities (Semanas 5-6)

9. **Community page** con todas las pestañas
10. **Community header** y sidebar
11. **Community navigation**

Fase 4: User Profiles (Semanas 7-8)

12. **User profile** con todas las pestañas
13. **Profile dropdown** component
14. **Settings pages** completas

Fase 5: Comments & Advanced (Semanas 9-10)

15. **Comment system** anidado
16. **Real-time features** con WebSocket
17. **Infinite scroll** y lazy loading
18. **Mobile responsive** optimization

Fase 6: Polish & Testing (Semanas 11-12)

19. **PWA features**
20. **Testing completo**
21. **Performance optimization**
22. **Documentation**

¿Te parece bien esta arquitectura de Angular? ¿Quieres que detalle algún componente específico o empecemos con la implementación de alguna fase?