## **🔐 Authentication Module**

### **1. Login Flow**

* Login form validates user input.
* On submit, dispatches login action to NgRx store.
* On success:  
  + Stores token/user in localStorage.
  + Dispatches loginSuccess.
  + Redirects to /home.

### **2 . Signup Flow**

* Signup form collects user info.
* Dispatches signup action.
* AuthEffects handles signup in local storage , don’t allow duplicate email ids .
* On success:  
  + Redirects to login

### **3. NgRx Store (auth/store)**

* **Actions:**
  + login, loginSuccess, loginFailure
  + signup, signupSuccess, signupFailure
  + logout
* **Reducer:** Updates auth state (user, token, isAuthenticated)
* **Effects:** Handles API interaction and side-effects (navigation)

4. **Auth Guard**

For checking user authentication.

## **🛒 Product Module**

### **1. Home Page**

* Protected by AuthGuard.
* On init, dispatches loadProducts action.

### **2. NgRx Store (products/store)**

* **Actions:** loadProducts, loadProductsSuccess, loadProductsFailure
* **Reducer:** Stores product list in state.
* **Effects:** Fetches products from [fakestoreapi.com](http://fakestoreapi.com).

## **📦 State Management (NgRx)**

### **Store Modules**

Registered in AppModule:

ts

CopyEdit

StoreModule.forRoot(reducers),

EffectsModule.forRoot([AuthEffects, ProductEffects]),

## **🧭 Routing Configuration**

### **app-routing.module.ts**

ts

CopyEdit

const routes: Routes = [

{ path: '', redirectTo: 'login', pathMatch: 'full' },

{ path: 'login', component: LoginComponent },

{ path: 'signup', component: SignupComponent },

{

path: 'home',

component: ProductListComponent,

canActivate: [AuthGuard],

},

];

## **🎨 UI & Styling**

* All components styled using **Bootstrap 5** classes.
* Form validation feedback via Bootstrap alerts.

## **✅ Application Flow Summary**

graph TD

A[User Visits App] --> B{Authenticated?}

B -- Yes --> D[Redirect to Home]

B -- No --> C[Show Login Page]

C --> E[Login Attempt]

E -->|Success| D

E -->|Fail| C

D --> F[Home Page Loads]

F --> G[Dispatch loadProducts]

G --> H[API Call: fakestoreapi.com]

H --> I[Render Product List]

## **🔄 Logout**

* Clear token from localStorage
* Dispatch logout action
* Redirect to login

## **🧪 Testing & Debugging**

* Use @ngrx/store-devtools for state inspection.
* Add form validators for email/password fields.
* Simulate login with dummy backend or fake service.