TradeSwitch App

Store TypeScript Code

Generated: 11/25/2025, 6:44:56 PM

Total Files: 2

Table of Contents

Ø=ÜÁ store

Ø=ÜÄ store\app.reducer.ts

```
import { ActionReducerMap } from '@ngrx/store';
import { AppState } from './app.state';
import { strategyReducer } from '../features/strategy/store/strategy.reducer';
import { reportReducer } from '../features/report/store/report.reducer';
import { userReducer } from '../features/auth/store/user.reducer';

/**
 * Root reducer that combines all feature module reducers.
 *
 * This reducer map combines all feature-specific reducers into a single
 * root reducer for the NgRx store. Each feature module manages its own
 * state slice, and this reducer orchestrates them together.
 *
 * Reducer Structure:
 * - strategy: Handles strategy rule configurations
 * - report: Handles report data and statistics
 * - user: Handles user authentication and profile data
 *
 * Usage:
 * This reducer map is used when configuring the NgRx store in the
 * application root module to provide a unified state management system.
 *
 * Relations:
 * - strategyReducer: From strategy module store
 * - reportReducer: From report module store
 * - userReducer: From auth module store
 * - AppState: Root state interface
 *
 * @constant
 * @type ActionReducerMap<AppState>
 */
export const appReducers: ActionReducerMap<AppState> = {
  strategy: strategyReducer,
  report: reportReducer,
  user: userReducer,
};

```

Ø=ÜÄ store\app.state.ts

```
import { UserState } from '../features/auth/store/user.reducer';
import { ReportState } from '../features/report/models/report.model';
import { StrategyState } from '../features/strategy/models/strategy.model';

/**
 * Root application state interface for NgRx store.
 *
 * This interface defines the complete application state structure,
 * combining all feature module states into a single root state.
 *
 * State Structure:
 * - strategy: Strategy module state (rules, configurations)
 * - report: Report module state (trades, statistics, user key)
 * - user: User authentication and profile state
 *
 * Usage:
 * This state is used by the root reducer to combine all feature reducers
 * and provide a single source of truth for the entire application.
 *
 * Relations:
```

```
 * - StrategyState: From strategy module
 * - ReportState: From report module
 * - UserState: From auth module
 * - app.reducer.ts: Combines all reducers
 *
 * @interface
 */
export interface AppState {
  strategy: StrategyState;
  report: ReportState;
  user: UserState;
}

```