**Store**

Finance Store

Guest (portfolio and template), selected portfolio.

**Action Documentation**

PROFILE ACTIONS

**Create new portfolio.**

Purpose: Purely creates a new portfolio with no tickers.

Instances:

1. Guest creates new portfolio from screener or portfolio.
2. Auth user creates new portfolio from screener or portfolio.

Implementation:

1. Guest -> load to profile in redux (& therefore local storage) OR (done)
2. Auth user -> save to DB and load to profile as well.

**Get User Portfolios (For Display user specific portfolios NOT template.)**

Input: none. Purpose: retrieve all saved **user** portfolios and corresponding tickers.

Purpose: Retrieves all user portfolios for display.

Instances:

1. Guest gets portfolio in Export or Portfolio page.
2. Auth user gets portfolio in Export or Portfolio page.

Implementation:

1. Guest -> NO need to dispatch anything. Data already in redux user portfolios.
2. Auth user -> get from DB. Save to user portfolios in redux.

**Add to portfolio** – add tickers in a portfolio in user’s profile.

Input: name, tickers, and profile.

Instances:

1. Guest adds tickers to portfolio.
2. Auth user adds tickers to portfolio.

Implementation:

1. Guest

Checks:

* If name is in user portfolios in profile in redux. Else, he is trying to edit template, therefore invalid.
* If ticker already exists in that portfolio in redux.

Update user portfolio in redux ONLY.

1. Auth user

Checks:

- If name is a user-specific portfolio. Else, he is trying to edit template, therefore invalid. -- - Check if ticker already exists in that portfolio.

**Dispatch get user portfolios** to get new portfolios from DB and update user portfolios in redux.

\*If edit failed due to editing template, allow user to create new portfolio with tickers from current template.

**Remove from portfolio** – removes tickers from portfolio.

Input: name, tickers, and profile.

Instances:

1. Guest removes tickers to portfolio.
2. Auth user removes tickers to portfolio.

Implementation:

1. Guest

Checks:

* If name is in user portfolios in profile in redux. Else, he is trying to edit template, therefore invalid.

Update user portfolios in redux ONLY.

1. Auth user

Checks:

- If name is a user-specific portfolio. Else, he is trying to edit template, therefore invalid.

**Dispatch get user portfolios** to get new portfolios from DB and update user portfolios in redux.

PORTFOLIO ACTIONS

**Set Portfolio** –Inputs: portfolio name. (Retrieve template OR own).

Purpose: Retrieves tickers based on portfolio name and gets tickers data. Updates redux portfolio.

Instances:

1. Guest selects a portfolio. (from template or own)
2. Auth user selects a portfolio.

\*If page load and selected portfolio is null, both will ‘select’ a default template on page load.

Implementation:

1. Guest -> first try get from redux (own). If cannot be found, means it is a template portfolio -> get from DB.
2. Auth user -> get from DB (template or own).

**FLOW: ADD/REMOVE FROM CURRENTLY SELECTED portfolio**

1. Add or remove portfolio (name). Updates user portfolios. If auth, use DB as source of truth.
2. Set portfolio. Updates selected portfolio.

AFTER: DO THE EXACT SAME FOR TEMPLATES. Auth user will have their own templates stored in DB. Guests will have templates stored in guest templates (redux/local storage). Both have access to default templates. BUT may not edit.

**Authentication Flow & Persisting of state across refresh.**

If we use server-side session, you can store authenticate (or not) and store user info and user movement in session.

If we use JWT, you can only use it for authentication. User movement is not important so we store it in session storage.

Session storage store ALWAYS:

* Storage -> Filters (screener).
* Token -> JWT token. Linked to redux auth.

Session Storage/redux BEFORE sign in.

* Storage -> Guest portfolios

ON login, we save guests stuff into DB and reset redux state.

Make requests to DB directly AFTER sign in.

* User portfolios.
* User templates.

AFTER sign-in,

Store in Redux but not in session storage:

* Auth User Portfolios

CHECK whether user signed in.

Check redux for token. If exists, we assume signed in else if request fails, we sign user out by

If no token, not signed in.

Important: To react to changes, we should always get and set redux. DO NOT get or set directly to local storage! Local storage is just used as persistent storage and will always be linked to redux state.

**Rule**

For auth users, always use DB as the source of truth. For guest, always use redux as the source of truth.

AUTH ACTIONS

**Login**

Implementation:

1) Store JWT token in local storage

2) Store guest portfolio into user DB. NOT DONE!

3) Clear guest portfolio in local storage.

4) Navigate to portfolio.

**Logout**

**Reset**

Resets redux state. Used in login and logout.

When you go to any tab.

Sort sector.

Delete any company. Cannot delete properly.