**Final Task**

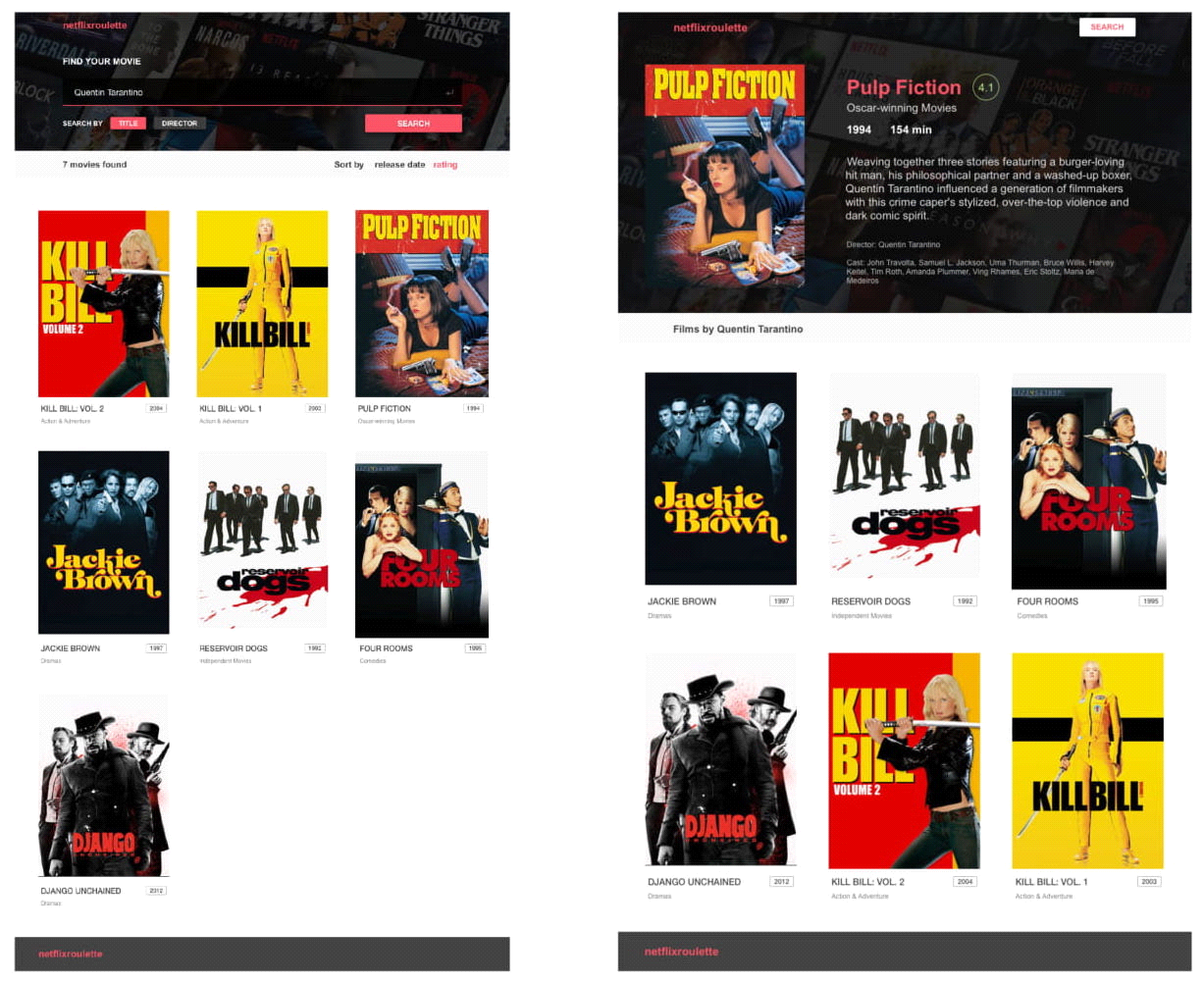
The idea is to create a single page application, which will allow users to search the Netflix Roulette database and save films as favorites.  
  
There will be three pages: main page with search, a page with a particular film and page with favorites. The last one should have footer, list of films that were marked as favorite and header without search.

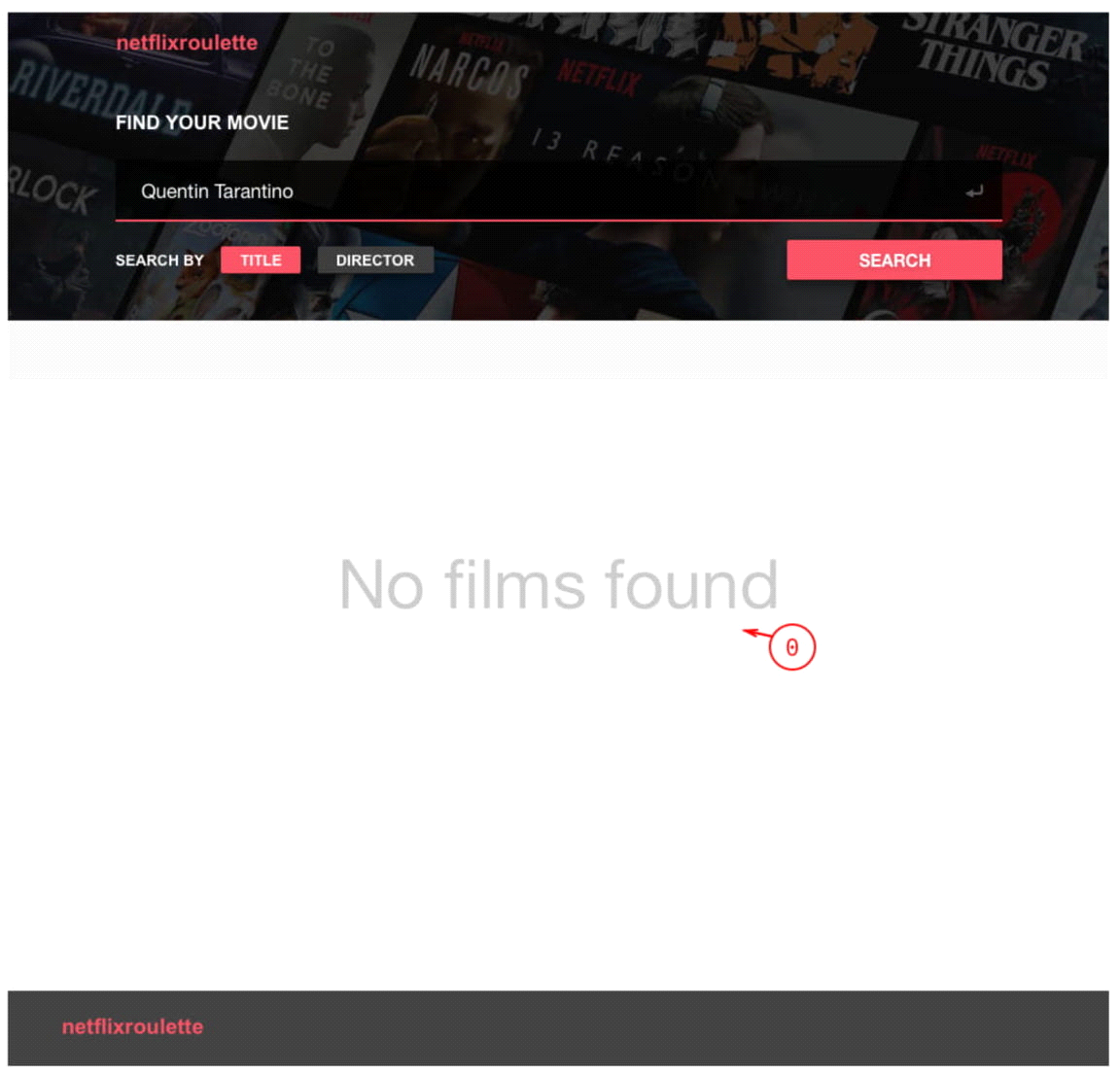
**JSON Server**

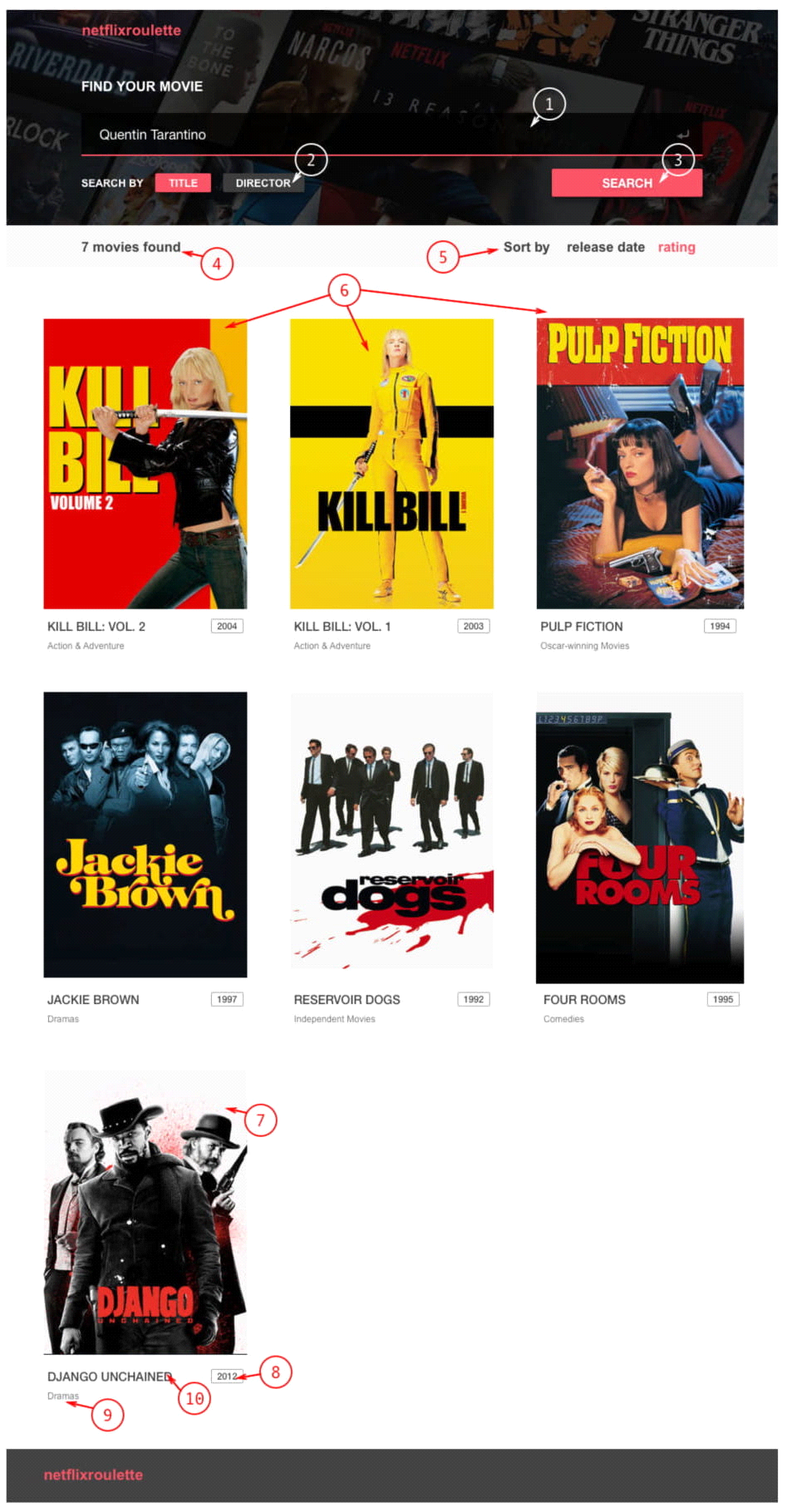
All requests should be performed with AJAX. To mock data and server responses use JSON Server. You can find a documentation here: <https://github.com/typicode/json-server>

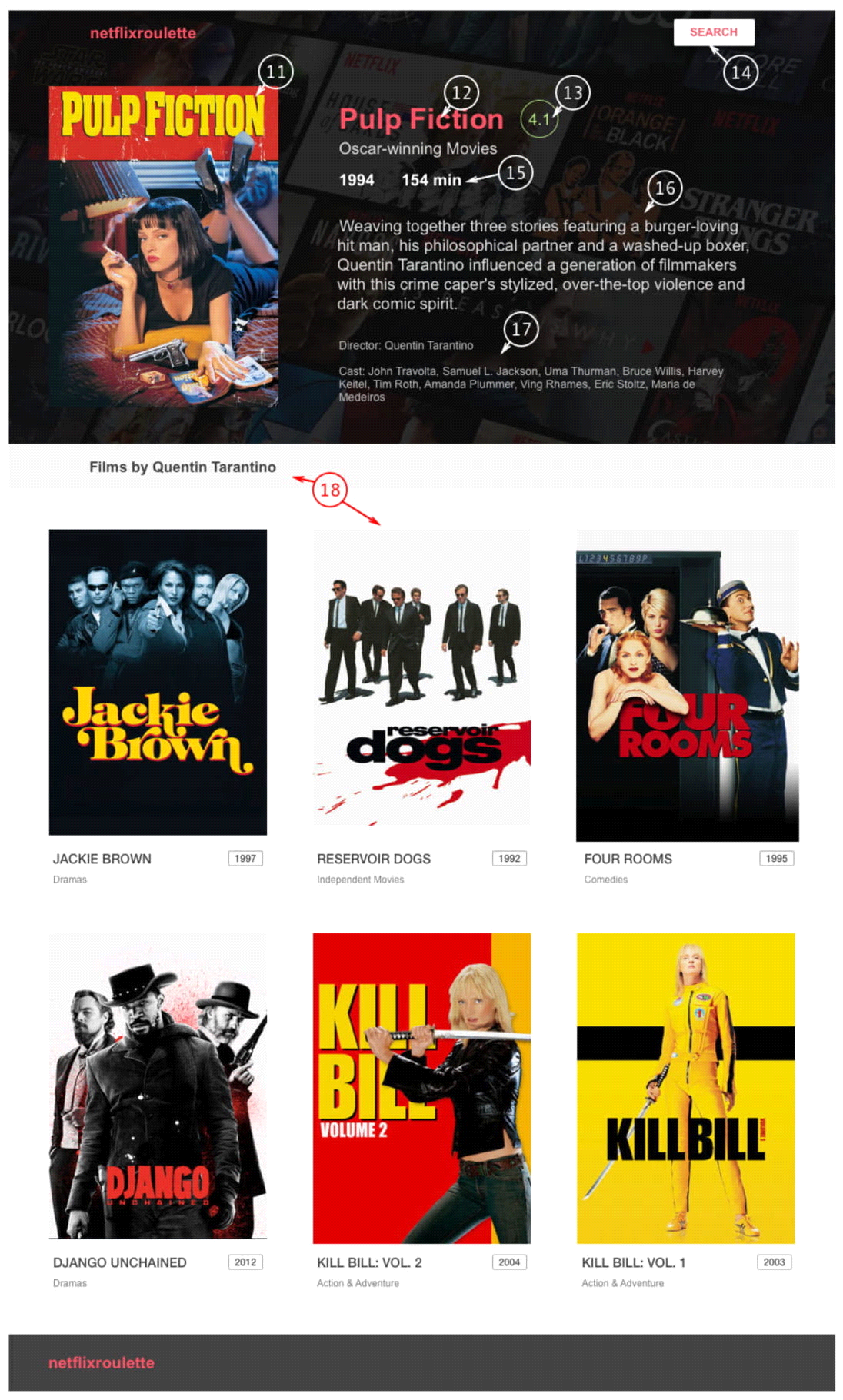
**Interface**

Here are two screens presented with break down. There is also an additional screen with empty state of the main page. Design and captions are available below.









**Captions**

**0. Empty results state**

**1. Search field**

Enter button should work as well

**2. Search filter**

By default «Title» is selected. Click to switch option.

**3. Search button**

**4. Results count**

**5. Results filter**

«release date» is selected by default. Click to switch option.

**6. Results body**

All found items without pagination.

**7. Item image**

URL for the image you will get from the server

**8. Item release date**

**9. Item genre**

**10. Item title**

**11. Film cover**

**12. Film title**

**13. Film rating**

**14. Search button**

Returns user to the main page with search

**15. Film duration and release year**

**16. Description**

**17. Director and cast list**

**18. Films by the same director**

These items are not displayed on design but should be added:

**Star icon** for every film item. Click to mark/unmark item as favorite.

**Link to favorites.** Link on search page directs to page with list of favorite films.

**Task 1**

**Setting up the environment**

*Lectures: Webpack*

For this task you will need to setup a simple NodeJS server. We recommend using the **Express.**

Please install **Webpack** and setup **Babel** for your application. Try to make your configuration simple, as for the small SPA.

**Task 2**

**Create the HTML layout with React components for both pages**

*Lectures: Core Concepts, Components*

You should only layout these pages using JSX. They will be used in future tasks.

Try to use all of the React composition power when performing the task: decompose components into small reusable parts.

**Task 3**

**Create routing for your application.**

*Lectures: Routing*

Link app states between each other with React router.

By default user lands on a new page with empty results state (*caption 0).*

When user performs a new search, you should redirect him to:

**localhost/search/Search%20Query**

When a new user lands on the page with such URL, you should perform search and display results.

When user clicks on a film item, redirect him to: **localhost/film/Attack%20on%20titan**

On switching search type or sorting type you shouldn’t switch any routes.

**Task 4**

**Fetching and managing data with Redux**

*Lectures: Redux*

Make your search component perform real AJAX requests. Connect redux and store all the data in your Redux store.