# Exercise: Single Page Application

Problems for exercises and homework for the ["JavaScript Apps" course @ SoftUni.](https://softuni.bg/trainings/3708/js-applications-june-2022)

**Working with Remote Data**

For the solution of some of the following tasks, you will need to use an up-to-date version of the **local REST service**, provided in the lesson’s resources archive. You can [read the documentation here](https://github.com/softuni-practice-server/softuni-practice-server).

## SoftForum

This task is to write functionality for a public forum. This app doesn't need any registration. Every user is able to read and write in the forum, and by creating a post or topic, the user should write also his/her own name.



### Creating topic with post

On the home page there are all topics visible. Also there is a form for creating a new topic with a title, username and post (comment) in it. The button "Post" should create the new topic if all the fields are correctly filled up (and cleans up the input fields after that). The button "Cancel" only clears all the input fields, without sending any request to the server.

Use this URL to create topics: **http://localhost:3030/jsonstore/collections/myboard/posts**



### **Hint**:

You can add an attribute with unique id to each post when creating it.

### Comment on Topic

When the user selects a topic by clicking on it, the app should redirect to a page with all posts (comments) for the selected topic and a form for creating new post (comment).

The structure for the post should be as the following:



After typing in the comment and sending the POST request to the server the page should render updated view with the new post.



You can use this URL: **http://localhost:3030/jsonstore/collections/myboard/comments**



### Edit Movie

Logged-in users should be able to **edit** movies, added by them.

Clicking the [Edit] **button** of a **particular movie** on the Movie Details page should **display** the Edit Movie page inserting the additional information of the movie in the input feelds:



* After a **successful edit** the user should be redirected to the current movie Details page.

### Like Movie

Logged-in users should be able to **like** movie, added by **other** **user**.

**NOTE**: A user should **NOT** be able to like a **movie**, created by **himself**.

Clicking the [Like] **button** of an **movie** (on the Movie Details page) should **make a POST request to the following URL: http://localhost:3030/data/likes with body { movieId: id } ( id – this is the id to the current movie ) .** After **successfully like**:

* [Like] **button** changes to [Liked {number of likes}] **span so** users can‘t **like a movie** multiple times**.**



### Delete Movie

Logged-in users should be able to **delete their** movies.

Clicking the [Delete] **button** of an **movie** (on the Movie Details page) should **delete** the **movie**.

After **successful** **movie delete** the Home pageshould be **shown**