01 - Rijksmuseum application

Build an application that uses the Rijksmuseum API. The application allows people to search by name of artist (Rembrandt, Vermeer) and see a list of works of art by this artist. Detailed information about the work of art in question can then be displayed.

The application has the following requirements:

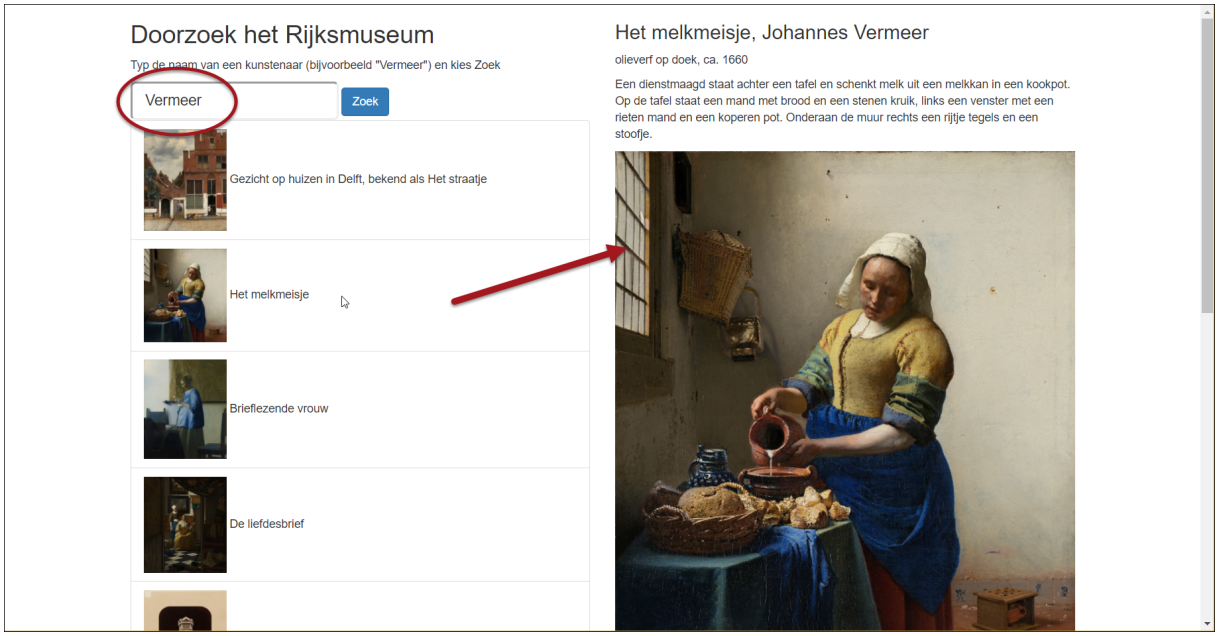
- The app uses Angular as the front-end framework.
- The architecture of the app must comply with the rules of the Angular Style Guide (https://angular.io/guide/styleguide).
- Make sure you have an attractive design. You may use a CSS framework such as Bootstrap, but this is not required. You may also use additional libraries such as Angular Material or PrimeNG. This is also not mandatory.
- The app must be usable on mobile devices (responsive).
- The app contains a search box and a button with sufficient instructions in the user interface.
    - When the visitor fills in an artist and clicks on *Search*, a list of works of art is shown.
    - For each description, a small image of the artwork is shown.
- Selecting a work of art ensures that more details are shown for that work of art.
- Make sure the app is scalable. Use the Single Responsibility principle for components. Use services to communicate with the backend.
- Optional: users can mark a work of art as a Favourite. It is then saved in HTML5 localStorage on its own device

WORKING METHOD:

- Use the Angular CLI to start the project. If necessary, add additional libraries or Angular Modules.
- Sketch out on paper what the layout will look like. Consider desktop and mobile usage (via Chrome DevTools).
- Sign up to use the Rijksmuseum API. This can be done via https://www.rijksmuseum.nl/nl/api and/or http://rijksmuseum.github.io/. There are several steps:
    - Create a new account for *Rijksstudio*. Use your Facebook account to sign in, or sign in via email/password.
    - If necessary, confirm your registration (if you have chosen e-mail/password).
    - Request an API key via My Rijksstudio (https://www.rijksmuseum.nl/nl/rijksstudio/mijn/gegevens). This is done via *Data*, *Advanced data* (bottom). You will receive your key by mail.

    - Read the possibilities for API use on Github pages and check out the examples. A request to request all works of art about Vermeer, for example, looks like this: https://www.rijksmuseum.nl/api/nl/collection?q=vermeer&key=<your_key>&format=json
    - Examine for yourself how detailed information about a work can be requested.
    - Optional: build in *pagination*. For example, the user will first see a list of 10 works of art and links for Next / Previous. The parameters for this are described on Github, under the heading Collection.
- Insert the user interface and logic into the component to call the service.
- Write a service using for example `getArtist(<name>)` and getDetails `(<object-number>)` as methods . Have this service `return` an `Observable<any>`.
    - If you have time, you may write your own artist and art typing/interface (recommended).
    - Expand the service with methods you consider necessary.
    - Also read the documentation, to send parameters to the API you want (for example the addition `imgonly=true`, if you only want results for which an image is also available. See http://rijksmuseum.github.io/ for more parameters).

For example, the application may look like this:



- Finally, optional: generate a distribution-build of your application with `ng build -prod [--aot]` and publish it on the internet. You can for example use free hosting on Github Pages. Examples of this are described on:
    - https://github.com/angular/angular-cli/wiki/stories-github-pages

---

- https://alligator.io/angular/deploying-angular-app-github-pages/
- or use the `angular-cli-ghpages` package directly: https://github.com/angular-buch/angular-cli-ghpages

- Send the URL to the teacher for assessment ;-)

Good luck!

Peter Kassenaar, info@kassenaar.com
