

Frontend Engineer - Technical Challenge

Dear candidate,

Thank you again for your application for the position as **Frontend Engineer** (m/f/d).
Today, we want to get a deeper understanding of your knowledge and problem-solving skills.

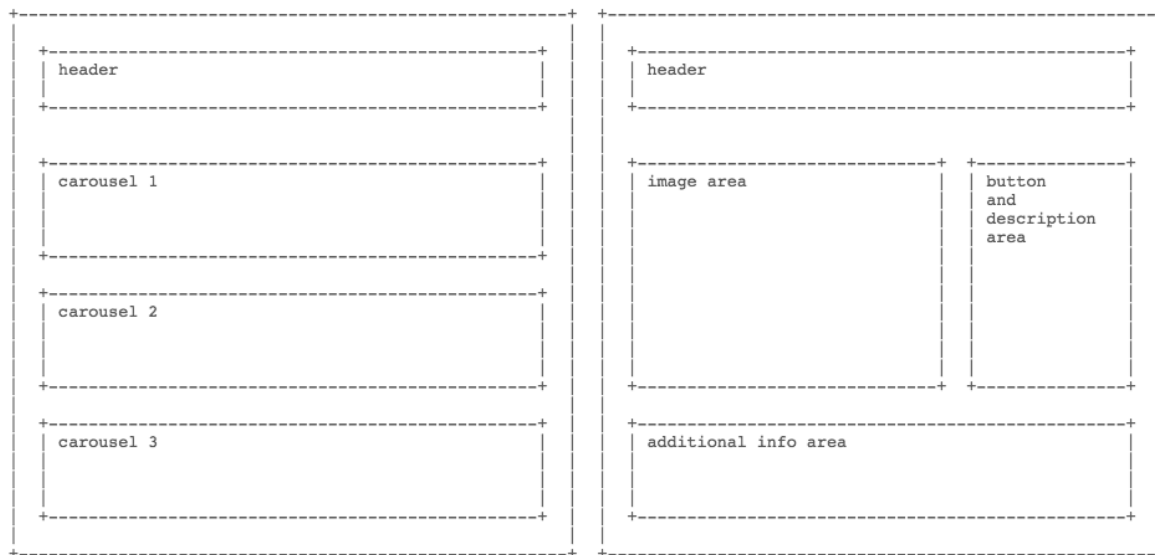
The challenge we would like you to solve is the following.

Problem definitions

Implement a web application to browse film by categories, being able to open a specific film to see its details. To be more specific:

- The homepage should contain 3 carousels with films
- Each carousel is a different category of films
- When clicking on an item, it should go to a detailed page of that item
 - The detail page should include a description, an image and a button
 - The button should trigger an “add to wish list” action
- Depending on the category of the item, the detail page should have:
 - A different font
 - A different button
 - Any other differentiation you think it can be added
- There should be a wish list section where all the items added can be seen



Here you can see a wireframe of what the homepage and the film detail page should contain in terms of components:



Tech stack requirements

Read them very carefully!

- Use React with TypeScript. If you never used TypeScript before, you can also use JavaScript
- Use SCSS for the styling
 - You can split the bundled CSS in different files but...
 - Don't use Styled Components, CSS Modules or Tailwind
- Use Vite for the bundling
- SSR support is required
- The codebase should be handcrafted. The exercise is aimed to understand your abilities to structure a frontend project and your skills in server and client side code management.
 - ❌ You're not allowed to use:
 - Full stack frameworks like Next.js
 - A scaffolding tool like *create-react-app-vite* that already defines the whole project structure for you. ⓘ You can still use *npm create vite@latest*
 - Any other tool that basically defines the whole project structure for you and already provides lots of abstractio

-  You're allowed to use
 - *Routing libraries* like React Router, Tanstack Router or any routing library you prefer
 - *Data loading* libraries like Tanstack Query or any other library you prefer
 - *State management* libraries like Redux, Zustand or any other library you prefer
-  You're allowed, but we give you extra points if you don't use them
 - UI libraries like Shadcn or Radix that gives you component primitives
- You can use any open API for films that you want. We recommend [TheMovieDatabase API](#)

Delivery requirements

- Publish your code to a [git repository](#) and share the link. Our suggestion is to use [GitFront](#)
- Write a [markdown documentation](#) that explain very thoroughly the steps necessary to run the project

Evaluation criteria

- How clean is your code and its reusability
- The complexity of the layout you build
- How clean and ordered are the CSS-like final styles
- How the components are being structured
- If DRY principles have been followed
- How much, what you tested and how
- The SSR strategy implemented