ROKT

JOIN THE TEAM!

Frontend Engineering Take Home Project

Photo search web applicaton

Thank you for taking the time to do our engineering recruitment project. Your time is valuable, and we appreciate that. We designed this exercise to be as realistic to day-to-day work as possible. It's an opportunity for us to gain a deeper understanding of how you think and work. For you, it's a chance to show off your full range of experience and expertise in a comfortable setting.

You should approach this project in the same way you would a real project at work. This is a coding exercise, but it's also an exercise in communication and project management.

How long should this take?

There's no exact time for how long this will take. Some engineers complete it in 2 hours, others take 4. You are not judged on speed of work. We encourage you to take the time that you need, but don't burn yourself out. While important, this is only one step in the interview process.

How to submit your solution

In an effort to prevent rejected or bounced emails due to zip files, we have a Google form for you to upload your solution.

Google form URL: https://forms.gle/g58hyjNVYNyEa8PPA

Complete the form with your email address and a single zip file containing your project. Please ***do not*** include any build artifacts, like node_modules, or sensitive information, like your API key, in the zip file. ***Treat the zip file like you would a public GitHub repo.***

Please make this a single zip file named: *`${yourName}-${roleAppliedFor}.zip`*

Project brief

We need you to build a website to let users browse photos from Pexels (see below notes on getting an API key). There's a ton of competition in the photo space online, so we need to stand out. We need a solution that's lighting fast to load with a snappy, responsive UI. We need to greet site visitors with a selection of curated photos and give them the ability to search our extensive catalog of quality photos from professional photographers.

Minimum requirements

Your solution needs to satisfy all of these. There are a couple specific server/client needs, but beyond those, the implementation is up to you. Use any tools, libraries, and frameworks you think best accomplish the task. (See below user stories for these requirements)

- Display 10 curated photos on the home screen. Use the Pexels curated photos resource to render these photos.
- Provide pagination for Curated photos. Paging should not cause a page refresh. From the client side initiate the needed requests to allow the user to go forward and backward 10 photos at a time
- Provide a photo search. Searching should not cause a page refresh. Consume the photo search resource and display the results to the user, along with pagination, if needed

These are the requirements for an MVP. Beyond these, dig into any and all areas as deep as you like.

Suggested Extras

- Implement a server that wraps the Pexels API, providing one or more of the following features:
  - maintain your application API key on the server, avoid exposing it to the client.
  - server-side rendering
- Write tests however you see fit.
- Placeholder content for images before they're loaded.

Getting a Pexels API Key

To work with the Pexels API, you need an API key. To get a key;

- Create a free Pexels account
  - https://www.pexels.com/onboarding
  - Follow "I want to download"
  - Complete the form. Make sure you use a valid email address
- Confirm your email
- Visit the Image & Video API section of your account
- Provide a description and a URL. These can be fake, feel free to use the examples below or write your own
  - Example description: "I'm using the API for programming practice projects"
  - Example URL: https://example.com

User stories

Your solution needs to satisfy all of these.

As a user visiting the website...

- I can see an initial set of curated photos on the home screen
- I am able to access the the photographer’s name and url if those details are available for every photo
- I can paginate the list of curated photos
- I can use a text input to search for photos I’m interested
- I can see the results of my search in the photo viewing area
- I can paginate search results if needed
- I should not be shown the "previous" paginiation button when there is no previous page
- I can refresh my browser and retain my search query and/or page

As a developer working on the project locally...

- I have access to all project-specific local setup instructions I need to run the project
- I can install any required dependencies with npm or yarn
- I can compile and run the project in one step

How we evaluate your solution

While this is a basic web server and website, there’s a wide range of implementation possibilities. We’re not looking for specific implementation details. We want to see the choices you make. Your preference of tools. And how you approach project requirements.

Here are a few questions we ask when reviewing solutions:

- Did the candidate follow the instructions for submission?
- Does the solution meet the minimum requirements?
- Does the solution satisfy all the user stories?
- Is the project documentation clear?
- Is code readable and concise? does it follow conventions for the chosen language, framework, and libraries?
- Am I able to follow the documentation to get the project running locally without issue?
- If tests are available, am I able to run them locally? Do they pass?
- If we added this repo to our codebase, would other developers be able to work on it by reading the documentation alone?