Project Rubric

Your project will be evaluated by one of our senior devs according to the https://github.com/airbnb/javascript styleguide. Please review for detailed project requirements.

App requirement

Create a new nuxt 2/3 application using nuxt-app or nuxi (if you go with nuxt 3), that fetches and filters elements from two different api sources

Dogs api: https://dog.ceo/dog-api/

Cats api: https://docs.thecatapi.com/

Technologies that are mandatory:

Nuxt

Typescript

We recommend tailwind but it's not mandatory

Development Strategy

1. Obtain a Cat api key
    - All api requests for the cats need to be authenticated https://docs.thecatapi.com/authentication
    - You may have some initial concerns with placing your API key directly within your JavaScript source files, so every implementation that would help with security is really appreciated
2. Dog Api Dog Api.
    - Dog api doesn't need an auth key
3. Implement a list view of the set of cats
    - Cats can be filtered by breed using a text field
    - A cat image can be added as a favourite https://docs.thecatapi.com/api-reference/favourites/favourites-post
    - When a cat image is already a favourite, display it differently (hint: a different border or a heart icon)
    - A cat image can be removed from favourites https://docs.thecatapi.com/api-reference/favourites/favourites-delete
4. Implement a list view of the set of dogs
    - Dog images can be filtered by breed using a text field
5. Create a tab view to switch between cats and dogs list view
6. Provide filters options in both components :

    - Both apis are different therefore the filters can vary
    - The filter required is a input search , other filters are well appreciated but not mandatory
7. There should be an initial fetch server side when the page loads so that the user can see a list of cats or dogs depending on which one is being selected by default.
8. Both list views have things in common, therefore using slots or elements that will help to reuse the components are well appreciated
9. Switching between tabs shouldn't re render the list views (hint: vue keep-alive)
10. By clicking on one of the images it should redirect the user to the detail page
    - If the element is a cat it's mandatory to show the image, breed and if the cat is added as a favourite or not
    - If the element is a dog it's mandatory to show only the image and breed

Overall, the application's interface should be intuitive to use and is up to the developer to use the framework of her/his preference. Here are a few examples of how to make the experience straightforward for your user:

- The input text area to filter the pets should be easy to locate
- It should be simple to understand which set of elements are being filtered
- Selecting a pet should redirect the user to a detail page
- The filters, list of results, tab selected (cats or dogs) shouldn't change when the user goes back to the search page (hint: use the store)

Asynchronicity and Error Handling

All data API's used in the application should load asynchronously, and errors should be handled gracefully. In case of error (e.g., a situation in which a third party API does not return the expected result), we expect your webpage to do one of the following:

- A message is displayed that notifies the user that the data cannot be loaded
- There are no negative repercussions to the UI

Note: You should handle errors if the browser has trouble initially reaching the third-party site as well. For example, consider a user using your Pets App, but the user's firewall prevents him/her from accessing the servers.

Code duplication

The functions that fetch the data from the dogs or the cats api need to be abstracted so that we can use one function that can handle both scenarios

Hint: composables and using typescript generics in order to type the responses

Typescript

The components, functions and responses need to be well typed (no anys allowed for those)

Testing

This is not mandatory but it’s well appreciated

Submission

Please send the link to the repository with your solution. A few other things to keep in mind:

1. The node_modules directory may contain thousands of files and should not be contained in the submission. If your project code is on GitHub, you may need to remove tracking from these files (and remove them from your repository) before submitting.
2. Your project *must* include a README with clear instructions for setting up and running your project application code.
3. The master branch is the default Github repository branch. If you wish to submit another branch, you'll need to set it as the new default branch inside your Github
4. If you are having any problems submitting your project, please just reply to the e-mail with which you received these instructions.

Looking forward to seeing your solution, good luck!