Software Developer Course Assessment

Quantitative Assessment Practice -3

Course Name: Programming with JavaScript

Current Week: (2024/07/15)

## Introduction:

The purpose of this assessment is to help us understand how the class is doing in terms of the review material that we have covered during the previous couple of weeks. The only purpose of this assessment is for us to improve our approach to review and ensure that what we’re currently doing is an effective strategy. Completion of this assessment is mandatory - if you don’t submit a solution, it will be marked as incomplete. If you do submit a solution, it will be marked as complete, as you will receive full marks.

Again, the goal here is to help you all in the best way that we can, so please do be honest when answering the questions related to how long it took, which resources you used, etc. And please ensure that you do your own work – don't just copy off a friend to get it done, earnestly do your best with it. If you can’t get it completely working, give us what you have. While it will be graded, the grade will not count against you, it’s just a way for us to see where everybody is, and to know which concepts, if any, we, as a class, may be struggling with.

Deadline: You will have until the end of the day on **Saturday July 27,2024 (4:00pm)** to submit your assessment solutions. Please ensure you answer all the questions outlined in the instructions portion of this document as well in your submission.

Instructions: Your name: \_Anhelina Romanchuk\_

You are allowed to complete the assessment problems below in whatever way you can but please answer the following questions/points as part of your submission:

1. How many hours did it take you to complete this assessment? (Please keep try to keep track of how many hours you have spent working on each individual part of this assessment as best you can - an estimation is fine; we just want a rough idea.)

Answer: [Around 48-50 hours]

1. What online resources you have used? (My lectures, YouTube, Stack overflow etc.)

Answer: [Lectures, the react practice helped a lot because it was similar to QAP, ChatGPT for gallery component]

1. Did you need to ask any of your friends in solving the problems. (If yes, please mention name of the friend. They must be amongst your class fellows.)

Answer: [Did everything alone]

1. Did you need to ask questions to any of your instructors? If so, how many questions did you ask (or how many help sessions did you require)?

Answer: [No, but used their instructions from lessons]

1. Rate (subjectively) the difficulty of Making this all! from your own perspective, and whether you feel confident that you can solve a similar but different problem requiring some of the same techniques in the future now that you’ve completed this one.

Answer: [It was hard. I kind of understand what I need to do but it was hard. I understand how the components connect to each other but the structure of React is hard to figure out. Was really hard to connect gallery to other components. Did not know how to send correct photos with a click of the button. In the end I did it and I do understand what I wrote. I think it will be hard for me to recreate the same project, but it will go faster for sure]

## Project Name: Dog Image Gallery

Overview: Create project in React “Dog Image Gallery” which should be a responsive React application that allows users to explore images of various dog breeds. Using the Dog CEO API, users can select a dog breed and specify the number of images they want to view. The application dynamically fetches and displays the images based on user input, making use of modern React hooks (useState and useEffect) to manage state and side effects.

## Features:

Responsive Design: Make sure that the application is styled to be responsive, ensuring a pleasant user experience on different devices.

Breed Selection: Users can select a dog breed from a dynamically populated dropdown list.

Image Quantity Input: Users can specify the number of images to load (between 1 and 100).

Dynamic Image Fetching: On form submission, the application fetches the specified number of images for the selected breed and displays them in a gallery format.

Hover Effect: Images in the gallery have a hover effect, enhancing the visual appeal.

Technologies Used:

React: For building the user interface.

CSS: For styling the components.

Dog CEO API: The Dog API (<https://dog.ceo/dog-api/documentation/>) is a free web service that uses data from the Stanford Dogs Dataset (<http://vision.stanford.edu/aditya86/ImageNetDogs/>). This dataset contains images and information about 120 breeds of dogs and is used for machine learning and artificial intelligence training.

Components:

App.js:

The main component that holds the entire application.

Manages state for the selected breed, number of images, and fetched images. ***[Hint: You might need three state variables, on each to hold breed, number of images, and fetched images – and thus you might need three functions here to handle each of them which you might pass as props to <BreedSelector/> component and handle them]***

Renders the BreedSelector and ImageGallery components.

BreedSelector.js:

A form component for selecting a dog breed and the number of images to fetch.

Fetches the list of dog breeds using the Dog CEO API and populates a dropdown list. **[You need to use useEffect() here to fetch all the breeds and then take out their names by using object.keys(){do some research on this function}.**

Handles form submission to fetch images for the selected breed and number of images.

ImageGallery.js:

A component that displays the fetched images in a gallery format. ***[ The images number will be passed from App.js and here images will be mapped and produced with the amount sent by App]***

Applies styling to ensure the images are displayed nicely with hover effects.