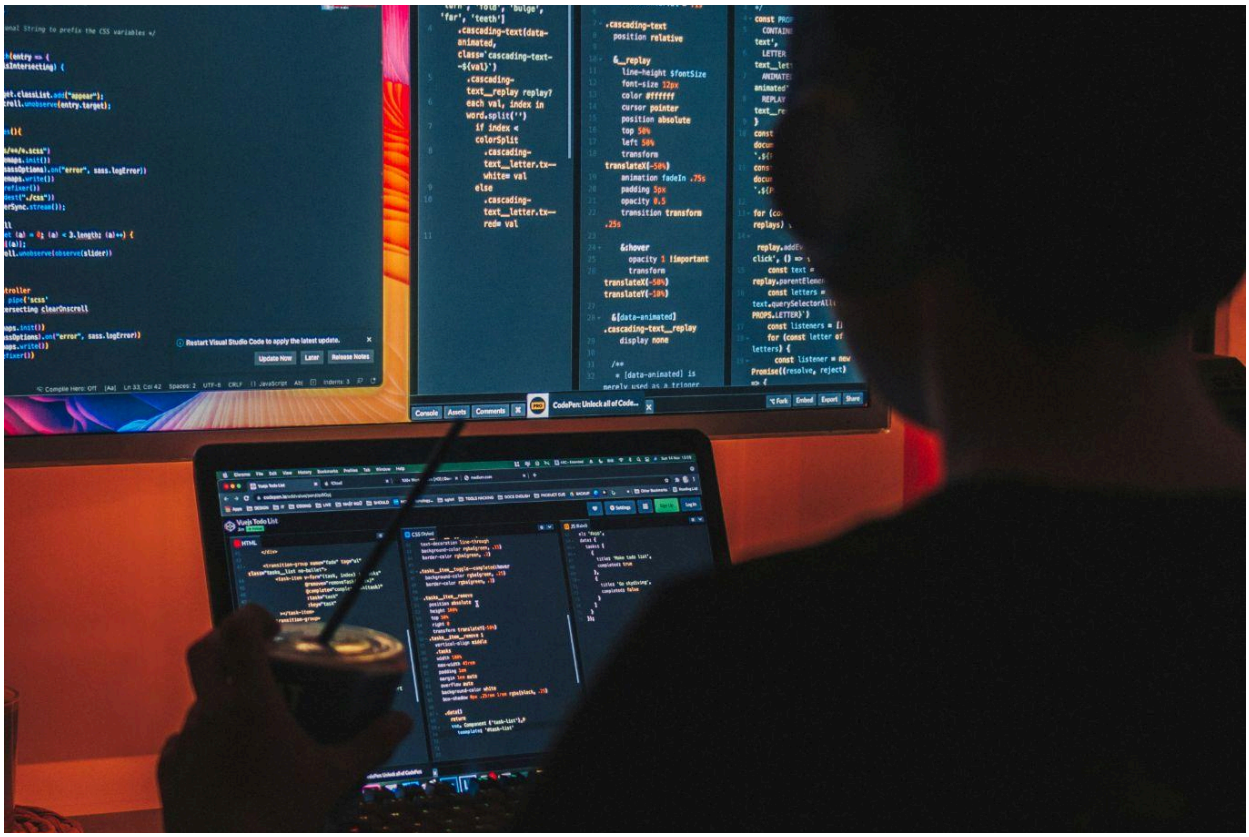

COA TASKFORCE
TAKE HOME CHALLENGE

02.06.2024

—

CODE of AFRICA

PRODUCT TO BUILD: Interactive Photo Gallery

Overview

The purpose of this coding challenge is to assess your skills in HTML, CSS, and JavaScript, as well as your problem-solving abilities. You will be tasked with building an interactive photo gallery based on provided Figma designs. Additionally, you will complete two coding challenges, using javascript, that test your logical thinking and coding aptitude.

Timeline

- **Total Duration:** **3 Days**
- **Final Due Date:** **09th June, 2024**

Requirements

1. **Responsive Design:** The interactive photo gallery should adapt seamlessly to different screen sizes and devices
2. **Figma Design Accuracy:** Match the provided Figma designs pixel-perfectly, including layout, spacing, typography, and visual styles
3. **Hover Interaction:** Implement the hover interaction as shown in the Figma prototype, displaying additional details when a user hovers over a photo
4. **Cross-browser Compatibility:** Ensure consistent performance across modern web browsers, such as Chrome, Firefox, Safari, and Edge
5. **Code Quality:** Write clean, readable, and well-structured code, following best practices and coding conventions for HTML, CSS, and JavaScript

How to work on this challenge

Access the Figma designs:

- Follow this figma link to view the designs in prototype mode: Figma Link
- The designs showcase the interactive photo gallery for both desktop and mobile views
- Use these designs as a visual reference for implementing the gallery

Set up your development environment:

- Choose your preferred code editor or IDE
- Ensure you have the necessary tools and frameworks installed (e.g., HTML, CSS, JavaScript)

Create a public GitHub repository:

- Sign in to your GitHub account or create a new one if you don't have an account
- Create a new public repository for this challenge
- Clone the repository to your local machine

Organize your project:

- Create appropriate directories and files for your code and assets
- Use meaningful names for files and folders to maintain clarity and organization

Implement the interactive photo gallery:

- Start with the HTML structure, creating the necessary elements for the gallery

- Style the gallery using CSS, following the design guidelines from Figma
- Implement interactivity and functionality using JavaScript
- Test your implementation regularly to ensure it works as expected

Solve the coding challenges:

- Read the problem statements carefully and understand the requirements
- Break down the problems into smaller subproblems if necessary
- Write clean, efficient, and well-commented code to solve each challenge
- Test your solutions with various inputs to verify correctness
- Add a folder called "**Challenges**" on your repo where you put the files for the coding challenge. Name first file: '**arrayMap.js**' and second '**stringTransform.js**'

Document your work:

- Include a README file in your GitHub repository
- Provide clear instructions on how to set up and run your project

Commit and push your code:

- Regularly commit your changes to your local repository with descriptive commit messages
- Push your commits to your GitHub repository to keep it up to date

Review and refine:

- Before submitting, review your code for readability, consistency, and best practices

- Run final tests to ensure your photo gallery and coding challenge solutions work as expected
- Make any necessary refinements or optimizations

Tasks

4.1 UI Challenge: Interactive Photo Gallery

- Implement an interactive photo gallery based on the provided Figma designs.
- The gallery should be responsive and work seamlessly on both desktop and mobile devices.
- Implement features such as image thumbnail navigation, full-size image viewing, and any interactions specified in the designs.

4.2 Coding Challenge 1: Array Manipulation

Problem Statement:

Given an array of integers and a target sum, determine if there exists a contiguous subarray within the array that sums up to the target. Return true if such a subarray exists, otherwise return false.

Example:

Input: arr = [4, 2, 7, 1, 9, 5], target = 17

Output: true

Explanation: The subarray [7, 1, 9] sums up to 17, which is equal to the target.

Constraints:

- The array will contain between 1 and 100,000 elements.
- The elements in the array and the target sum will be between -1,000,000,000 and 1,000,000,000.

Expected Time Complexity: O(n), where n is the length of the array.

Expected Space Complexity: O(1).

4.3 Coding Challenge 2: String Transformation

Here's the updated problem statement with real-world word examples:

Problem Statement: Given a string, transform it based on the following rules:

- If the length of the string is divisible by 3, reverse the entire string.
- If the length of the string is divisible by 5, replace each character with its ASCII code.
- If the length of the string is divisible by both 3 and 5 (i.e., divisible by 15), perform both operations in the order specified above.

Example: Input: "Hamburger"
Output: "regrubmaH"
Explanation: The length of the string is 9, which is divisible by 3 but not by 5 or 15. Therefore, the string is reversed, resulting in "regrubmaH".

Example: Input: "Pizza"
Output: "80 105 122 122 97"
Explanation: The length of the string is 5, which is divisible by 5 but not by 3 or 15. Therefore, each character is replaced by its ASCII code, resulting in "80 105 122 122 97".

Example: Input: "Chocolate Chip Cookie"
Output: "eikooCpihCetalocohC"
Explanation: The length of the string is 21, which is divisible by 3 but not by 5 or 15. Therefore, the string is reversed, resulting in "eikooCpihCetalocohC".

Constraints:

- The string will only contain alphanumeric characters and spaces.
- The length of the string will be between 1 and 1000.

Expected Time Complexity: O(n), where n is the length of the string. **Expected Space Complexity:** O(n), where n is the length of the string.

Note: You can assume that the input will always be valid and within the specified constraints. Your solution should handle all possible cases and return the transformed string accordingly.

These examples use real-world words related to food to illustrate the transformations applied based on the divisibility of the string length by 3, 5, and 15.

How to Submit:

Once you have completed the challenges, please share the link to your public GitHub repository via the Google submission link: Google submission link.

In your repository, make sure to include:

- A README file with setup instructions and explanations of your approach.
- The implemented interactive photo gallery.
- Solutions to the two coding challenges.

We will review your submission based on the following criteria

- Code quality, readability, and organization.
- Adherence to best practices and coding standards.
- Functionality and performance of the photo gallery.
- Correctness and efficiency of the coding challenge solutions.
- Clarity and completeness of the README file.

If you have any questions or need further clarification, please don't hesitate to reach out to us at clet@codeofafrica.com. +250 789 823 888

Best of luck with the challenge! We look forward to reviewing your submission,

TASKFORCE TEAM