Angular Junior/Mid-Level Developer Technical Assessment Challenge

Objective

In this assessment, you will demonstrate your skills in CSS, TypeScript, and Angular. The task is divided into three parts:

- First, you will create a smiley icon using only CSS to showcase your proficiency in CSS and responsive design.
- Next, you will solve a few TypeScript exercises to test your foundational knowledge of typing, writing functions, and working with generics.
- Finally, you will set up an Angular project with Angular Material, where you will implement a dynamic user list using a table with pagination, to demonstrate your expertise in Angular and Angular Material.

Total Time: 4-6 Hours

Technologies:

- HTML5/CSS3
- TypeScript
- Angular 15+
- Angular Material

Part 1: CSS Task - Smiley Icon with Pure CSS

Create a smiley entirely with pure CSS. You may not use any images or SVGs, only HTML and CSS. The smiley should look like the one shown in the image smiley.jpg.

Requirements:

- The smiley should be responsive and adjust its size based on the viewport.
- Use Flexbox or Grid to control the positioning of the eyes and mouth.
- Optional: Use pseudo-elements (::before, ::after) for an extra challenge.

Outcome: An HTML file with embedded CSS code.

Part 2: TypeScript Basics

In this task, you need to demonstrate your knowledge of TypeScript by solving the following smaller exercises:

Task 1: Functions & Typing:

Write a function calculateArea that calculates the area of a rectangle. The function should accept two parameters of type number (width and height) and return the area. This function takes two number arguments and returns their product, representing the area of the rectangle.

Task 2: Interfaces and Objects:

Define an interface User that has the following properties:

- name (string)
- age (number)
- email (optional, string)

Then, create an array of 3 users that uses this interface.

Task 3: Generics:

Write a generic function reverseArray that takes an array as an argument and returns a new array with the elements reversed. The function should work for arrays of any data type (e.g., string[], number[]).

Outcome: A TypeScript file that includes all three sub-tasks.

Part 3: Angular Project with Angular Material

Set up an Angular project where you use Angular Material. In this project, you need to implement the following requirements:

Task 1: Setup and Component Creation

- Create a new Angular project with Angular Material.
- Install Angular Material and set it up. Use the Indigo/Pink theme.
- Create a component called UserListComponent. This component should display a list of users that is loaded from a service.

Task 2: Angular Material Table and Pagination

- Use the Angular Material Table component to create a table that displays the list of users. Each row of the table should show a user's id, name and email.
- Implement pagination (Angular Material Paginator) for the table so that the list of users is spread across multiple pages.

Task 3: Display user details in a dialog

- Extend the existing Angular Material table (from Task 2) with the functionality to open a dialog when a row is clicked, displaying the full details of the user.
- Use the Angular Material Dialog component to implement a modal dialog.
- The dialog should display at least the user's full name, email address, phone number, and date of birth.

Bonus Task (Optional): Search Functionality

- Add a search function that allows filtering by username. Use an Angular Material input field for the search and dynamically filter the list as the user types.

Requirements:

- Use Angular services to provide user data (dummy data can be stored locally in the service; you can use the dataset from users.ts).
- The application should be responsive. The table and pagination should work well on different screen sizes.
- You can use Angular CLI to initialize the project and install Angular Material.

Outcome: A complete Angular project with at least one component (UserList-Component) that displays a list of users with an Angular Material Table and pagination.

Submission Guidelines

Upon completion, send us an email with a link to the assignment as a public GitHub repository with a clear README explaining how to run the application and any additional instructions or notes you want to provide. Ensure that the code is well-documented and follows best practices.

Evaluation Criteria

- CSS/HTML: Clean, structured, and semantic code. Responsiveness and flexibility of the smiley design.
- TypeScript: Correct typing, clean function definitions, and proper use of interfaces and generics.
- Angular: Project structure, correct use of Angular Material, clean implementation of the component(s), and functional pagination.