Instructions

This test aims to assess how you might approach a typical Developer task for FUZED and demonstrate your ability to work with a particular tech stack.

Please treat the test as if you are working on a real task. Keep it simple and have fun with it, spending as little or as much time as you need to demonstrate your abilities. We would like to see how you approach the problem and it doesn't necessarily need to be a complete solution. We are looking for quality over speed. Please document any notes in a readme file.

Please have thoughts about how you would ensure quality, and how this may influence the structure of your code. How can you ensure quality is built in? How would you ensure the solution works? We may ask you about these aspects of your solution.

Deliverables:

- A functional codebase for the mobile application.
- A readme file with:
    - An overview of your solution.
    - Instructions on how to run and test the application.
    - Explanations for any decisions or trade-offs made.
    - Thoughts on how you ensured quality in your solution.
    - Possible improvements you would make if given more time.

The Task

Build a **mobile application** using the **Expo framework** to detect **body points** from the camera. You can use any suitable library to achieve this functionality. The app should:

1. Access the device's camera and display it on the screen.
2. Detect and display body points (e.g. key joints such as the head, shoulders, elbows, knees, etc) in real time using the camera feed.
3. Optionally, overlay a visualization (e.g. skeleton) of the detected body points on the camera feed.

This is a purely technical challenge to show off your ability to implement some functionality on the chosen stack.

Technical Requirements

1. Frameworks/libraries:
    a. Use the **Expo** framework for building the app.
    b. Use any library you find suitable for detecting body points (e.g. **MediaPipe**, **TensorFlow.js**).
2. Code quality:
    a. Write modular and reusable code.
    b. Ensure proper error handling and graceful degradation.
    c. Include comments where necessary to explain complex logic.
3. Testing:
    a. Include a way to validate the functionality, and as a minimum include notes on how you would address quality control from a real-world scenario.

Optional Enhancements

Feel free to add any of the following enhancements to your solution. These are entirely optional and not required to complete the ask, but they can help demonstrate your creativity, attention to detail, and problem-solving abilities. Some examples include:

- Allowing users to switch between front and rear cameras.
- Optimizing the detection process to improve performance, making sure this is running smoothly in real-time.
- Add basic UI for toggling features (e.g. enabling/disabling body points visualization).

Evaluation Criteria

Your submission will be evaluated based on the following:

1. Functionality: does the application meet the requirements and work as expected?
2. Code quality: is the code well-structured, readable, and maintainable?
3. Problem-solving: how effectively did you approach the challenge?
4. Documentation: is the readme comprehensive and clear?
5. Testing: how well did you test and validate your solution?

Submission Instructions

Submit your codebase in a public or private Git repository. If private, please share access with a member of the FUZED team.