Engineering Challenge - Web FrontEnd

Build the web frontend client of an “Address Book” application

Implementation Details

Your web client should initially display the list of persons from the address book. The user should be able to select a person from the list in order to see more details about that person. Please use the API endpoints from https://randomuser.me/ (Documentation can be found at https://randomuser.me/documentation). Your client should be responsive.

Suggested UX hint:

- User should see the list of persons from the address book
- User should be able to select a person from the list and navigate to the details page
- User should be able to see the first name, last name, phone number on the details page

Deliverables

Please take your time to deliver a quality solution that shows your ability. Include:

- A **README** file that contains:
    - Deployment / running instructions. Assume that we’re running this on a Mac. Bonus points if instructions are for deployments to a distributed cloud infrastructure
    - A summary of the assignment
        - Your overall approach
        - What features you implemented
        - Given more time, what else would you have liked to complete and how long it would have taken you?
        - Given more time, what else would you have done to make the project more robust?
- Production-ready code that:
    - Your code checked into a git repository that can be shared with us (Github, Gitlab, Bitbucket, etc…). We should be able to run the code
    - Follows community standard syntax and style
    - Has no debug logging, TODOs, or FIXMEs
    - Has test coverage to ensure quality and safety

What we look for

- Clean code. Style we’re looking for: concise but descriptive.
- Enough functionality or code to show us your understanding of fundamental development practices

- Test coverage for your code. Bonus point if you are able to perform Test Driven Development
- Bonus points: Code in a *functional*, concise, declarative way. Things we will look for: higher-order functions, function composition, correct use of basic data structures and manipulations (map, reduce, apply, etc.)
- Bonus points for managing asynchronous or concurrent execution through high-level abstractions