8. Testing

8.1 Introduction

- Confirming that the software works as specified in the requirements
- Testing allows to ensure that the function is behaving in an intended way
    - Properties
        - Conciseness
        - Clarity
        - Repeatability
- Red-Green-Refactor Cycle
- Refactoring
    - Changing or updating the structure of code without impacting the functionality
- **Types**
    - *E2E (End-to-End)*
        - Interact with the app - the same way a user would
        - Slow and time consuming
        - Examples - Web Driver JS, Protractor, and Cyprus

- *Integration*
    - Testing how separate parts of the application work together
    - Example - React Testing Library and Enzyme
- *Unit*
    - Process of testing a specific piece of code in isolation
    - Unit is smallest piece of code
    - Practically → Function or Method

8.2 JEST

- JavaScript does not have inbuilt methods → that would allow tests to be written.
- Libraries
    - Jasmine - Mocha - Karma - q Unit
- Allows to test
    - Babel - TypeScript - Node - Angular - Vue
- Code Coverage
    - Higher the code coverage - lower the chances of unidentified bugs
- Mocking
    - Separate code from related dependencies during testing
    - This allows to ensure that unit testing is stand alone
- Jest
    - Comprises of in-built mocking functions
    - Facilitates async code
    - Snapshot Testing
        - To verify that there are no regressions in DOM

8.3 TDD

- Streamlined process of writing code → that will satisfy some requirements
- Traditional Development

    - Requirements → Coding → Testing
- TDD
    - Requirements → Failing Tests → Code → Code Passes Test → Improve Code
    - Minimize regressions
    - Prove new implementation is working
    - Automated test
    - Test implementation
    - Provide deocumentation