Testing

By: Puskar Adhikari

Introduction

Software testing is a systematic process of evaluating a software application to verify that it functions correctly, securely, and efficiently according to its specified requirements.

Why?

Why test frontend code?

- Prevent breaking UI logic (state, forms, routing).
- Ensure user interactions work as expected.
- Catch regressions early before deploying.
- Maintain confidence during refactors.

Frontend Testing

- **Functional Testing**
- **Visual Testing (UI Testing/Visual Regression Testing)**
- **Usability Testing**
- **Performance Testing**
- **Browser and Device Compatibility Testing**

Functional Testing

- **Functional Testing:** This verifies that the application's features and functionalities work as intended from a user's perspective. This includes:
  - **Unit Testing:** Testing individual components or functions in isolation.
  - **Integration Testing:** Verifying the interaction between different frontend components and with backend services.
  - **End-to-End (E2E) Testing:** Simulating full user workflows through the application to ensure critical paths work correctly.

Gitlab Repo

Clone this repo:

https://gitlab.com/ait-fsad-2025/labreference/fullstackapplication

Angular

- **Unit testing:** Jasmine + Karma (built-in)
- **E2E testing:** Cypress (Angular 16+ uses Cypress)

Unit Testing

https://angular.dev/guide/testing

Unit Testing

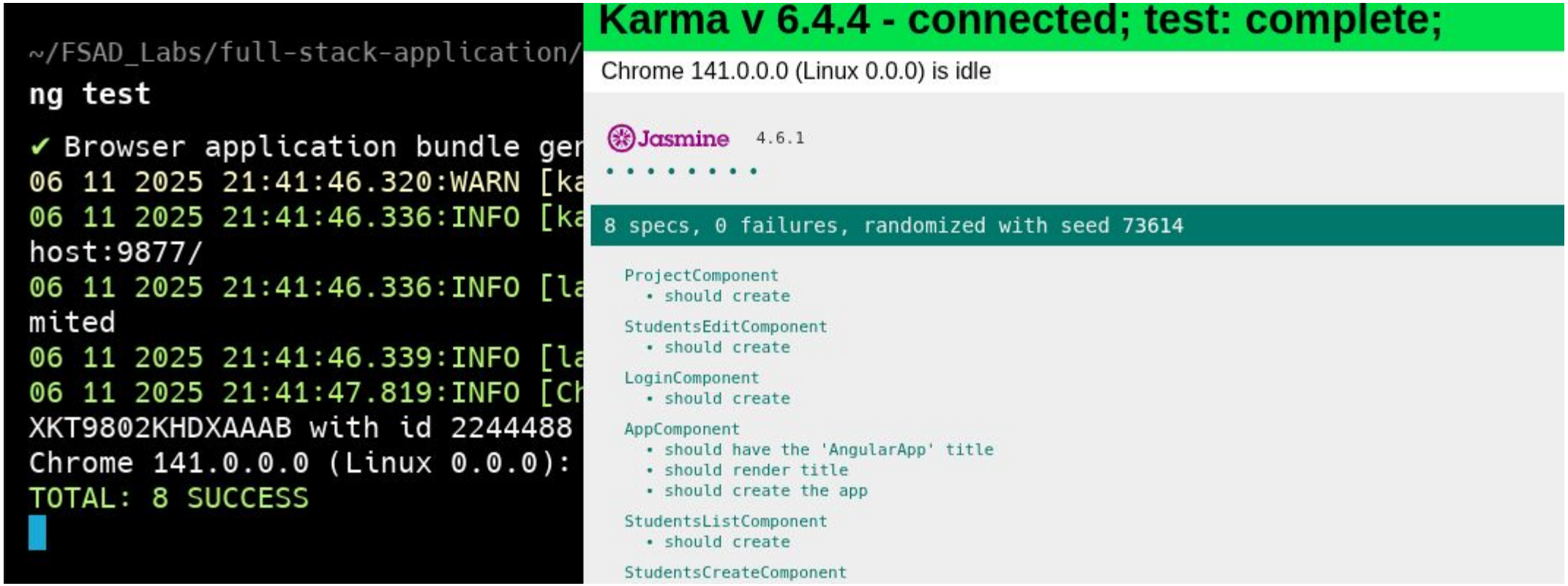
All unit tests are written in .spec.ts file

It follows the format of:

```
describe(‘’) {
    beforeEach()
    it(‘should do this’)
    it(‘has a title’)
}
```

Unit Testing

- cd angularApp/
- ng test



Unit Testing

login.component.spec.ts

```
import { ComponentFixture, TestBed } from '@angular/core/testing';

import { LoginComponent } from './login.component';
import { ReactiveFormsModule } from '@angular/forms';
import { CommonModule } from '@angular/common';
import { By } from '@angular/platform-browser';

describe('LoginComponent', () => {
 let component: LoginComponent;
 let fixture: ComponentFixture<LoginComponent>;

 beforeEach(async () => {
   await TestBed.configureTestingModule({
     imports: [LoginComponent, ReactiveFormsModule, CommonModule]
   })
   .compileComponents();

   fixture = TestBed.createComponent(LoginComponent);
   component = fixture.componentInstance;
   fixture.detectChanges();
 });

 it('should create', () => {
   expect(component).toBeTruthy();
 });

   it('form should be invalid when empty', () => {
   expect(component.loginForm.valid).toBeFalse();
 });

 it('form should be valid when email and password are filled', () => {
   component.loginForm.setValue({ email: 'test@example.com', password: '123456' });
   expect(component.loginForm.valid).toBeTrue();
 });

 it('should show error message when submitting invalid form', () => {
   component.onSubmit();
   expect(component.errorMessage).toBe('Please fill all required fields');
 });

 it('should call onSubmit when form is submitted', () => {
   spyOn(component, 'onSubmit');
   const form = fixture.debugElement.query(By.css('form'));
   form.triggerEventHandler('ngSubmit');
   expect(component.onSubmit).toHaveBeenCalled();
 });
});
```

Coverage Report

- ng test --code-coverage

React

- **React Testing Library:** A library designed to test React components by simulating user interactions and asserting on the rendered output, encouraging tests that resemble how users interact with the application.