1-Test

Test de integración y unitarios

```
# 1. Tipos de tests

# 2. Test de Integración con Cypress

# 3. Test Unitarios con Jest
```

```
As a: developer,
  I want: to test end to end my app
  so that: I can be sure of the functions

As a: developer,
  I want: to unit test my app
  so that: I can be sure of the structure
```

1 Test de Integración con Cypress

1.1 Cypress

Cypress

```
  "e2e:shop": "ng e2e shop-e2e --watch",
  "e2e:warehouse": "ng e2e warehouse-e2e --watch",
```

```
yarn e2e:shop
yarn e2e:warehouse
```

1.2 Test e2e

```
GIVEN: the shop web app
  WHEN: user visits home page
```

```
      THEN: should display welcome message
      THEN: should display welcome message from the API
```

apps\shop-e2e\src\support\app.po.ts

```
export const getGreeting = () => cy.get('h1');
```

---

apps\shop-e2e\src\integration\app.spec.ts

```
import { getGreeting } from '../support/app.po';

describe('GIVEN: the shop web app', () => {
  beforeEach(() => cy.visit('/'));
  context('WHEN: user visits home page', () => {
    it('THEN: should display welcome message', () => {
      getGreeting().contains('Hello world');
    });
    // needs the api server to run
    // yarn start:api
    it('THEN: should display welcome message from the API', () => {
      getGreeting().contains('and Welcome to api!');
    });
  });
});
```

---

2 Test Unitarios con Jest

2.1 Jest

Jest

```
  "test:shop": "ng test shop --watch --verbose",
  "test:warehouse": "ng test warehouse --watch --verbose",
  "test:api": "ng test api --watch --verbose",
```

```
yarn test:shop
yarn test:warehouse
yarn test:api
```

2.2 Tests unitarios

2.2.1 Componentes

```
GIVEN: an AppComponent declared in AppModule
  WHEN: the AppModule is compiled
    THEN: should create the component
    THEN: should have a property title with value 'shop'
    THEN: should render 'Hello world' in a H1 tag
```

---

shop: apps\shop\src\app\app.component.spec.ts

```
import { UiModule } from '@a-boss/ui';
import { async, TestBed } from '@angular/core/testing';
import { RouterTestingModule } from '@angular/router/testing';
import { AppComponent } from './app.component';

describe('GIVEN: an AppComponent declared in AppModule', () => {
  describe('WHEN: the AppModule is compiled', () => {
    beforeEach(async(() => {
      TestBed.configureTestingModule({
        imports: [RouterTestingModule, UiModule],
        declarations: [AppComponent]
      }).compileComponents();
    }));

    it('THEN: should create the component', () => {
      const fixture = TestBed.createComponent(AppComponent);
      const app = fixture.debugElement.componentInstance;
      expect(app).toBeTruthy();
    });

    it(`THEN: should have a property title with value 'shop'`, () => {
      const fixture = TestBed.createComponent(AppComponent);
      const app = fixture.debugElement.componentInstance;
      expect(app.title).toEqual('shop');
    });

    it(`THEN: should render 'Hello world' in a H1 tag`, () => {
      const fixture = TestBed.createComponent(AppComponent);
      fixture.detectChanges();
      const compiled = fixture.debugElement.nativeElement;
      expect(compiled.querySelector('h1').textContent).toContain('Hello world');
    });
  });
});
```

2.2.2 Services

```
GIVEN: a GreetingsService
  WHEN: the DataModule is compiled
    THEN: should be created
    THEN: should return an observable when call 'getGrettings()'
    THEN: should return 'Welcome to api!' when call 'getGrettings()'
```

libs\shared\data\src\lib\greetings\greetings.service.spec.ts

```
import {
  HttpClientTestingModule,
  HttpTestingController
} from '@angular/common/http/testing';
import { async, TestBed } from '@angular/core/testing';
import { Observable } from 'rxjs';
import { GreetingsService } from './greetings.service';

describe('GIVEN: a GreetingsService', () => {
  describe('WHEN: the DataModule is compiled', () => {
    beforeEach(() => {
      TestBed.configureTestingModule({
        imports: [HttpClientTestingModule]
      });
    });

    it('THEN: should be created', () => {
      const service: GreetingsService = TestBed.get(GreetingsService);
      expect(service).toBeTruthy();
    });

    it(`THEN: should return an observable when call 'getGrettings()'`, () => {
      const service: GreetingsService = TestBed.get(GreetingsService);
      const greetings$: Observable<any> = service.getGrettings$();
      expect(greetings$).toBeInstanceOf(Observable);
    });

    // Ojo al async para ejectuar las llamadas asíncronas
    it(`THEN: should return 'Welcome to api!' when call 'getGrettings()'`,
async(() => {
      const service: GreetingsService = TestBed.get(GreetingsService);
      service
        .getGrettings$()
        .subscribe(result =>
          expect(result).toEqual({ message: 'Welcome to api!' })
        );
      const httpMock = TestBed.get(HttpTestingController); // mock del backend
para no depender del servidor
      const req = httpMock.expectOne('http://localhost:3333/api'); // esperar a
```

```
que se llame a esta ruta
      req.flush({ message: 'Welcome to api!' }); // responder con esto
      httpMock.verify(); // comprobar que no hay más llmadas
    }));
  });
});
```

> **Blog de apoyo:** Tests unitarios con Jest y e2e con Cypress en Angular
>
> > By Alberto Basalo
>
> Next:

Detección del cambio en Angular

Estrategias de detección del cambio

Técnicas OnPush

Optimización