4. Test with Mock

Reference Demo4 in https://github.com/abigail830/angular-demo

Background

Login.component.html

```
<a [hidden]="needLogin()">Login</a>
```

Login.component.ts

```
import { Component, OnInit } from '@angular/core';
import { AuthService } from '../auth.service';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.scss']
})
export class LoginComponent implements OnInit {
  constructor(private authService: AuthService) { }
  ngOnInit() {
  }
  needLogin() {
    return !this.authService.isAuthenticated();
  }
}
```

auth.service.ts

```
import { Injectable } from '@angular/core';

@Injectable({
  providedIn: 'root'
})
export class AuthService {
  static authenticated: boolean;
  constructor() { }
  isAuthenticated(): boolean {
    return !!localStorage.getItem('token');
  }
}
```

Test with real authService and real localStorage

```
describe('Test with real AuthService and localStorage', () => {
  let component: LoginComponent;
  beforeEach(() => {
    const authService: AuthService = new AuthService();
    component = new LoginComponent(authService);
  });
  afterEach(() => {
    component = null;
  });
  it('should able to hidden login when have token', () => {
    localStorage.setItem('token', '12345');
    expect(component.needLogin()).toBeFalsy();
  });
  it('should able to show login when without token', () => {
    localStorage.removeItem('token');
    expect(component.needLogin()).toBeTruthy();
  });
});
```

Test with mock authService

Keys

- Create a mockAuthService in mock folder which extends AuthService

```
import { Injectable } from '@angular/core';

export class MockAuthService extends AuthService {
  authenticated = false;

  isAuthenticated(): boolean {
    return this.authenticated;
  }
}
```

Then Test with MockAuthService

```
describe('Test with MockAuthService', () => {
  let component: LoginComponent;
  let authService: MockAuthService;

  beforeEach(() => {
    authService = new MockAuthService();
    component = new LoginComponent(authService);
  });
  afterEach(() => {
    component = null;
    authService = null;
  });
  it('should able to hidden login when have token', () => {
    authService.authenticated = true;
    expect(component.needLogin()).toBeFalsy();
  });
  it('should able to show login when without token', () => {
    authService.authenticated = false;
    expect(component.needLogin()).toBeTruthy();
  });
});
```

Test with spyOn

Keys

- spyOn(authService, 'isAuthenticated').and.returnValue(false);
- expect(authService.isAuthenticated).toHaveBeenCalled();
- expect(authService.isAuthenticated).toHaveBeenCalledTimes(1);

```
describe('Test with spyOn', () => {
  let component: LoginComponent;
  let authService: AuthService;
  beforeEach(() => {
    authService = new AuthService();
    component = new LoginComponent(authService);
  });
  afterEach(() => {
    component = null;
  });
  it('should able to show login when without token', () => {
    spyOn(authService, 'isAuthenticated').and.returnValue(false);
    expect(component.needLogin()).toBeTruthy();
  });
  it('should able to hidden login when without token', () => {
    spyOn(authService, 'isAuthenticated').and.returnValue(true);
    expect(component.needLogin()).toBeFalsy();
  });
});
```