DP2 2024

Acme Software Factory

Repositorio: https://github.com/DP2-2024-C1-029/Acme-Software-Factory.git

Miembros:

- David Godoy Fernández (davgodfer@alum.us.es)
- Ismael Gata Dorado (ismgatdor@alum.us.es)
- Jaime Varas Cáceres (jaivarcac@alum.us.es)
- José María Portela Huerta (josporhue@alum.us.es)
- Juan José Gómez Borrallo (juagombor@alum.us.es)

Tutor: José González Enríquez

GRUPO C1.029

16/02/2024

Versión 1.0

Índice

Historial de versiones

Fecha	Versión	Descripción	Entrega
16/02/2024	V1.0	Descripción de los conocimientos del testing para la arquitectura WIS	D01

Resumen ejecutivo

Explicamos la experiencia que tenemos con el testing en la arquitectura WIS a través del uso de Junit y Mockito para la realización de pruebas unitarias.

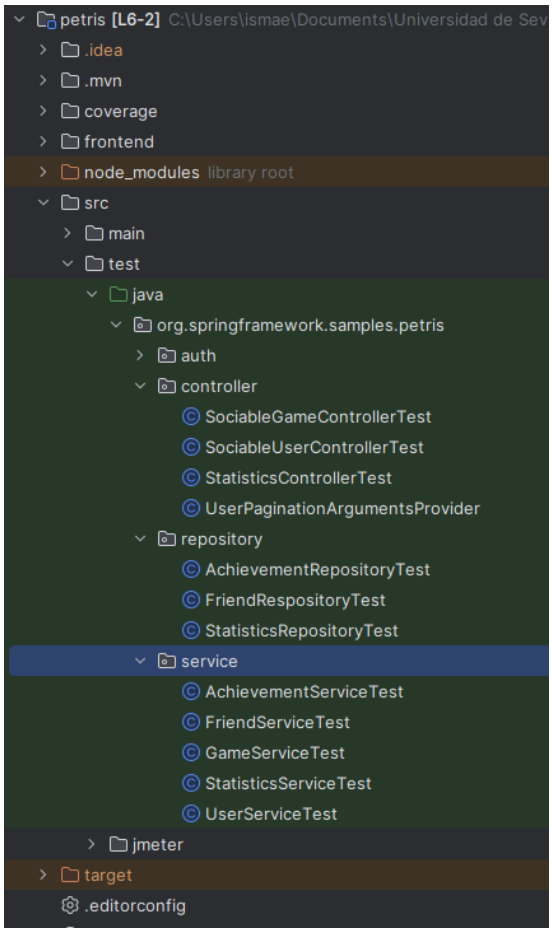
Introducción

En este documento recogemos un breve resumen de lo que conocemos de testing en la arquitectura WIS hasta ahora.

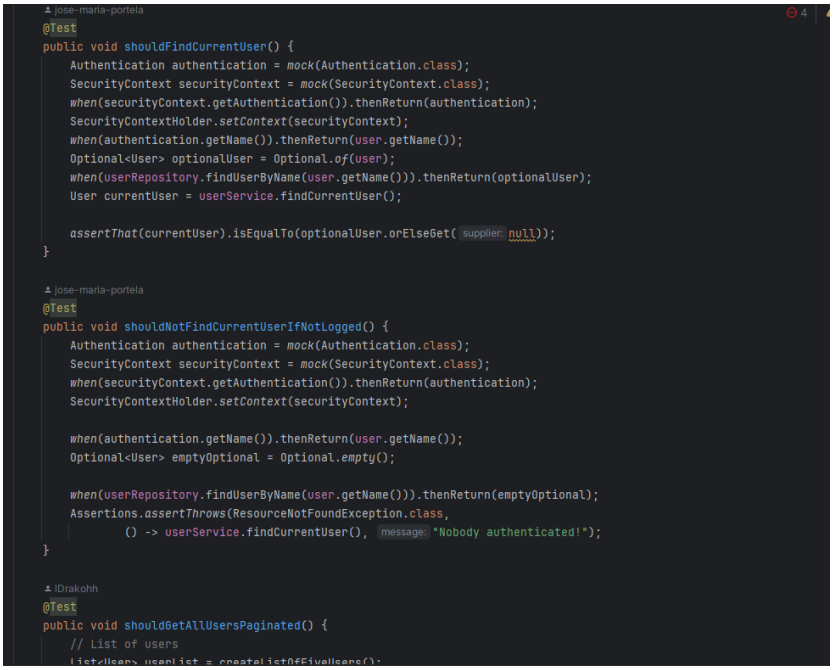
Contenido

Acerca del testing en la arquitectura WIS el equipo tienes algunas nociones y hemos trabajado sobre ella en la asignatura Diseño y Pruebas I.
En dicha asignatura hemos trabajado en su mayoría en pruebas unitarias de los diferentes componentes de la capa de aplicación, para los repositorios, servicios y controladores usando la librería de Junit 5 y Mockito para la simulación de los datos. Aunque no hemos llegado a realizar test de integración o end-to-end.

Adjunto una captura de pantalla de la estructura para los test del proyecto anterior:



Y otra captura de algún test de ejemplo:



```
jose-maria-portela
@Test
public void shouldFindCurrentUser() {
    Authentication authentication = mock(Authentication.class);
    SecurityContext securityContext = mock(SecurityContext.class);
    when(securityContext.getAuthentication()).thenReturn(authentication);
    SecurityContextHolder.setContext(securityContext);
    when(authentication.getName()).thenReturn(user.getName());
    Optional<User> optionalUser = Optional.of(user);
    when(userRepository.findUserByName(user.getName())).thenReturn(optionalUser);
    User currentUser = userService.findCurrentUser();

    assertThat(currentUser).isEqualTo(optionalUser.orElseGet( supplier: null));
}

jose-maria-portela
@Test
public void shouldNotFindCurrentUserIfNotLogged() {
    Authentication authentication = mock(Authentication.class);
    SecurityContext securityContext = mock(SecurityContext.class);
    when(securityContext.getAuthentication()).thenReturn(authentication);
    SecurityContextHolder.setContext(securityContext);

    when(authentication.getName()).thenReturn(user.getName());
    Optional<User> emptyOptional = Optional.empty();

    when(userRepository.findUserByName(user.getName())).thenReturn(emptyOptional);
    Assertions.assertThrows(ResourceNotFoundException.class,
            () -> userService.findCurrentUser(), message: "Nobody authenticated!");
}

lDrakohh
@Test
public void shouldGetAllUsersPaginated() {
    // List of users
```

Conclusiones

En conclusión, podemos confirmar que tenemos unas nociones básicas del testing en la Arquitectura y que a lo largo de la asignatura nos resultara familiar lo cual nos ayudara a entender mejor la asignatura y el proyecto.

Bibliografía

Intencionadamente en blanco.