**Test Exploratorio – Safe Wallet**

1. Objetivo del Test Exploratorio:

El objetivo principal es identificar posibles problemas, errores o áreas de mejora en las funciones de login y registro del sistema.

1. Áreas a Testear:

* Login de usuarios.
* Registro de usuarios.

1. Organización del Test Exploratorio:

**Sesión 1: Login**

* Probar diferentes combinaciones de nombre de usuario y contraseña.
* Verificar el comportamiento de la página en caso de credenciales incorrectas.
* Probar el flujo de inicio de sesión exitoso.

**Sesión 2: Registro**

* Completar el formulario de registro con diferentes conjuntos de datos.
* Verificar la validación de datos en el formulario.
* Probar el flujo de registro exitoso.

**Tours:**

Tour 1: Flujo Normal de Login y Registro

* Navegar a través del proceso de inicio de sesión y registro utilizando datos válidos.

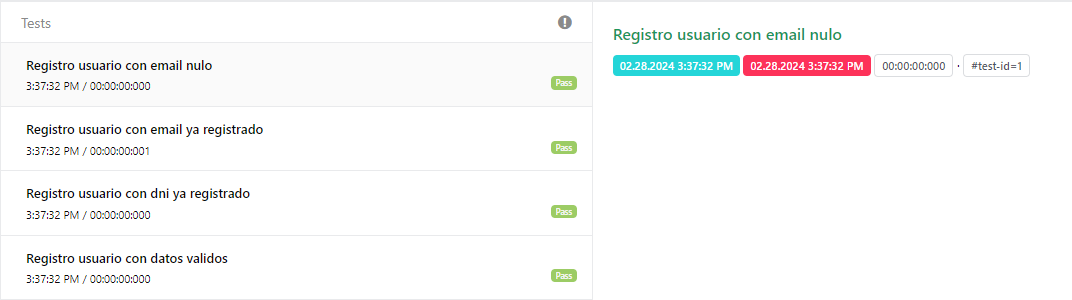
Tour 2: Escenarios de Error

* Explorar diferentes escenarios de error durante el proceso de inicio de sesión y registro, como credenciales inválidas, campos vacíos, etc.

**Escenarios posibles testeados: REGISTER**

Para el registro de usuarios en safewallet se utiliza un microservicio que se encarga de recibir los datos del usuario que se registrara. Este microservicio Guardara todos los datos del usuario y enviara solo los necesarios a dos microservicios utilizando feign

1. SafeWallet-keycloak: Quien tomara algunos datos del usuario para almacenarlos en Keycloak (Solo guarda el correo y el password).
2. SafeWallet-Transfers: Quien tomara algunos datos del usuario para crear la cuenta bancaria y almacenar solo esos datos (cbu, alias, correo, saldo, etc..).

****

1. **Registro usuario con datos validos**
   1. Microservicio: SafeWallet-UserDataFull.
   2. Clase: RegisterTest.
   3. Test: testCreateUser\_Success
   4. Resultado: Passed
2. **Registro usuario con DNI ya registrado.**
   1. Microservicio: SafeWallet-UserDataFull.
   2. Clase: RegisterTest.
   3. Test: testCreateUser\_alreadyRegister\_DNI
   4. Resultado: Passed
3. **Registro usuario con email ya registrado.**
   1. Microservicio: SafeWallet-UserDataFull.
   2. Clase: RegisterTest.
   3. Test: testCreateUser\_alreadyRegister\_EMAIL
   4. Resultado: Passed
4. **Registro usuario con email nulo.**
   1. Microservicio: SafeWallet-UserDataFull.
   2. Clase: RegisterTest.
   3. Test: testCreateUser\_alreadyRegister\_emailNull
   4. Resultado: Passed

**Escenarios posibles testeados: LOGIN**

En Safe Wallet se utiliza un adaptador keycloak desde el frontend para validar si hay un usuario en sesión, de no tener un usuario se envia al SSO de keycloak.

Una vez que el usuario este logeado se utiliza el token que se envía en un encabezado para la comunicación con los microservicios que conforman la billetera virtual. Cada microservicio es responsable de la seguridad de si mismo, traduciendo el token enviado por el Gateway para utilizarlo en el contexto de spring.

Se testearan los endpoints de Keycloack con REST ASSURE que nos permiten logearnos en nuestra aplicación.