ACTIVIDADES

**Investigación:**

* BraveNewCoin
* JWT
* Jest
* Protractor
* Azure DevOps

**Desarrollo Backend:**

* Pruebas unitarias
* Creación de usuarios
* Autenticación
* Agregar Cripto al usuario
* Listar Cripto del usuario
* Top Cripto (comparando monedas)

**Desarrollo Frontend:**

* Pruebas unitarias
* Maquetar
* Realizar cambio de monedas
* Listado de monedas

**Desarrollo de automatización:**

* Plan de pruebas
* Crear un pipeline (code, build, dev)
* ¿todo se puede automatizar? ¿Todo es sujeto a ser automatizado?

**R.** No. Los casos de prueba automatizables son las pruebas de regresión (después de cada release), Pruebas basadas en riesgo (verificar las funciones críticas), Pruebas complejas (llenar formularios) y Casos de prueba repetitivos (alimentado de una fuente de datos).

**Documentación:**

* Swagger
* Postman
* Supuestos
* Ymeter
* Readme
* Tag “v1.0”

PLAN DE PRUEBAS

## Introducción

En esta estrategia para la realización de pruebas automatizadas se describe el alcance de las pruebas, ambiente, los recursos, las herramientas, los riesgos, planes de contingencia y calendario de ejecución

## Alcance

Se realizarán las pruebas de caja negra (automatizadas) a las funcionalidades seleccionadas durante la planificación de cada sprint.

Las funcionalidades a ser automatizadas serán seleccionadas utilizando los criteios de la lista de chequeo “qué casos de pruebas automatizar”. Anexo.

## Riesgos y planes de contingencia

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Riesgos | Probabilidad de ocurrencia (1-5) | Impacto (1-5) | Severidad (prob\*impacto) | Plan de contingencia |
| 1 | Las funcionalidades no terminadas en el tiempo o sprint estimado no pueden estar en la planificación de automatización | 2 | 5 | 10 | Volver a planificar las funcionalidades a ser automatizadas |
| 2 | Los cambios en las funcionalidades que ya tienen pruebas automatizadas ocasionan trabajo al tener que actualizar los scripts | 3 | 3 | 9 | Estimar el esfuerzo del cambio y volver a priorizar las funcionalidades a ser automatizadas |

## Ambiente y herramientas de pruebas

### Herramientas

|  |  |
| --- | --- |
| Herramienta | Función |
| Selenium WebDriver | API para automatizar sistemas WEB |
| Protractor | Framework de test e2e para Angular |
| protractor-beautiful-reporter | Generación de reportes de protractor |
| Chromedriver | Crear instancias de navegador Chrome |
| Katalon recorder | Grabación de pruebas para automatización |

### Ambiente de pruebas

|  |  |
| --- | --- |
| Navegadores | Chrome |
| Sistemas Operativos | Windows |

## Criterios de entrada y salida

### Criterios de entrada

Las funcionalidades deben haber pasado las pruebas desde DEV hasta QA y haber sido probadas manualmente.

El framework de pruebas está instalado y listo para ejecutarse.

Los defectos mayores encontrados en las pruebas manuales han sido resueltos y cerrados.

### Criterios de salida

Todos los casos de prueba automatizados han sido ejecutados.

Se ha logrado una cobertura suficiente de los requerimientos y funcionalidades probados.

No hay algún defecto sin resolver.

## Planificación y ejecución de las pruebas

|  |  |  |
| --- | --- | --- |
| Sprint | Funcionalidades | Comentarios |
| 1 | Intercambiar los selectores de cryptomoneda, Realizar el cambio de una cantidad de criptomoneda a otra, Tomar la criptomoneda a cambiar desde la tabla de cryptomonedas, Evitar el cambio de criptomoneda cuando no se indica la cantidad |  |

Es necesario que se prueba manualmente la funcionalidad antes de crear las pruebas automatizadas para encontrar defectos mayores.

### Planificación dentro del sprint

Al finalizar el sprint de desarrollo comenzará el sprint de pruebas automatizadas. Siempre después de hacer unas pruebas manuales.

### Planificación de pruebas de regresión

La suite de regresión se ejecutará al final de cada sprint o al realizar un cambio.

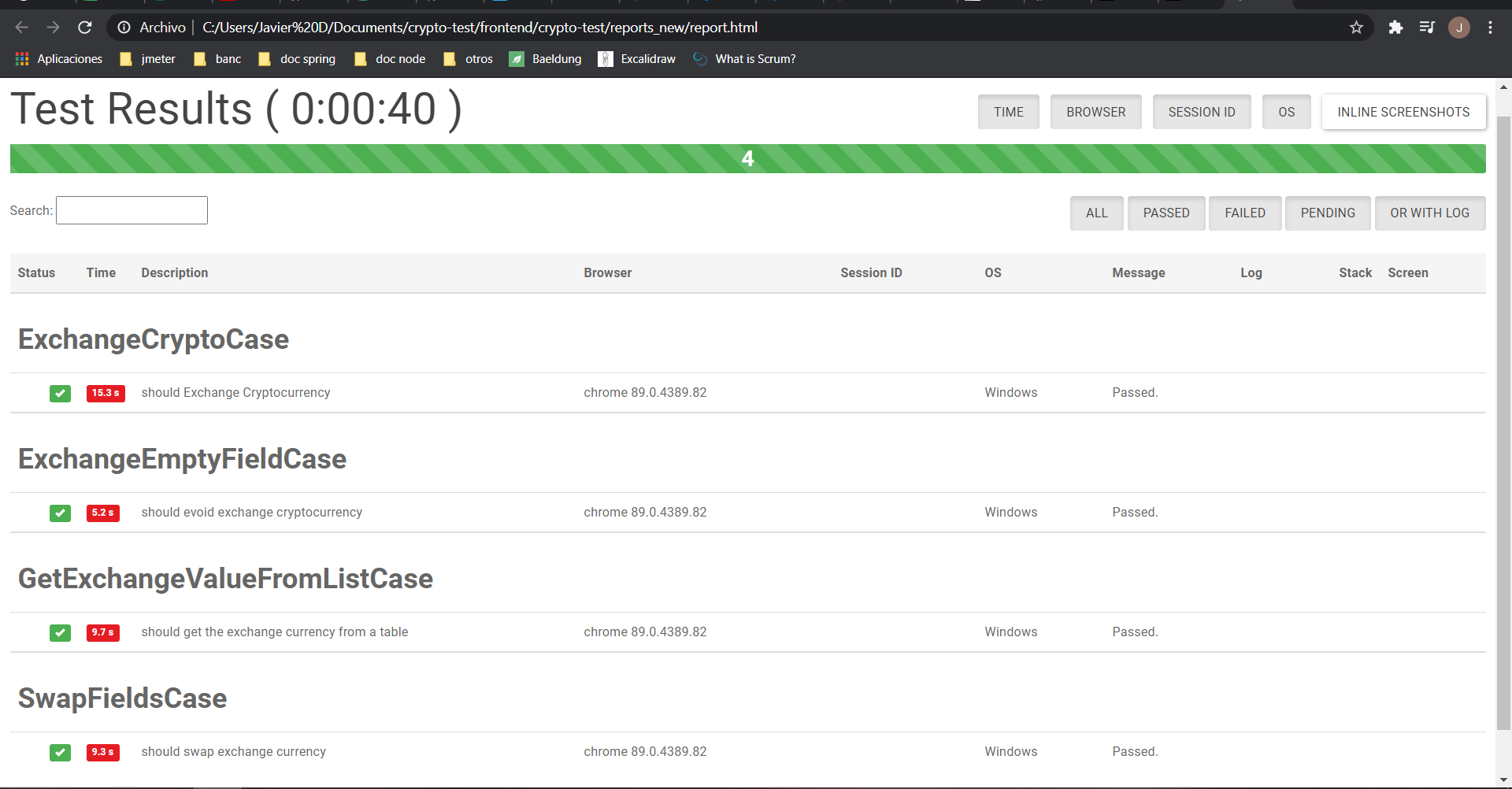
### Funcionalidades a probar

Para la vista de cambio de criptomoneda:

|  |  |  |  |
| --- | --- | --- | --- |
| Funcionalidad | Flujo | Criterio de aceptación | Comentarios |
| No se puede calcular el valor de cambio de monedas sin ingresar la cantidad. | El usuario selecciona las monedas a calcular, omite introducir la cantidad y ejecuta el cálculo. | Debe aparecer en la vista un mensaje de error cuando el usuario no ingresa la cantidad de moneda a calcular. La operación no se ejecuta. | No es una funcionalidad crítica. |
| Se calcula el valor de una cantidad de moneda respecto a otra. | El usuario selecciona las monedas, ingresa la cantidad y ejecuta el cálculo. | Debe darse una respuesta a la operación. | Es la funcionalidad principal de la vista. |
| Se tomarla criptomoneda hacia la que se va a calcular desde la lista de monedas para hacer la operación. | El usuario va a lista de monedas, selecciona la moneda a la que va a cambiar, selecciona la moneda, ingresa la cantidad desde la que va a calcular y ejecuta el cálculo. | Debe seleccionarse la moneda desde la lista y pasar al campo de moneda a la que se va a cambiar. Además, ejecutar la operación y dar una respuesta. | Depende de la funcionalidad de la vista de lista de monedas |
| Se puede intercambiar la selección de monedas que se van a operar. La moneda desde la que se hace el cálculo será ahora la moneda hacia la que se hará la operación. | El usuario selecciona las monedas a operar y ejecuta el intercambio de estas. | La moneda desde la que se hace el cálculo debe estar en la posición de la moneda hacia la que se hace la operación. Igualmente, la moneda hacia la que se hace la operación debe estar en la posición de la moneda desde la que se hará el cálculo. | El intercambio de monedas puede darse n veces y su resultado debe seguir estando dentro del criterio de aceptación. |

## Reporte de pruebas

El reporte de las pruebas se obtendrá a través de protractor-beautiful-reporter. Este Reporte informa sobre los resultados de la ejecución de las pruebas. Incluye las pruebas que pasaron, las pruebas que fallaron, errores encontrados, tasa de éxito y el tiempo transcurrido.



SUPUESTOS

Se usa la base de datos no relacional MongoDB para la solución, pero en una forma de implementación con tablas normalizadas por si se agregan cambios a las funcionalidades, pues un cambio en la base de datos en un futuro podría significar un re trabajo al corregir todas las funcionalidades afectadas.

Los Brave New Coin agregados a la bd tienen un precio aleatorio para facilitar la inserción y simular sus datos, aunque sean tomados de BNC.

El pipeline de Azure DevOps llega hasta las pruebas unitarias del código de backend para el API y la publicación del reporte resultante.

Los idiomas manejados por i18n serán español (es) e inglés (en) se ven al hacer build en dev, no se producirá despliegue de traducciones porque no hay etapa de despliegue a servidor en nube.

El api de BraveNewCoin tiene los precios de las criptomonedas, que se muestran en frontend, pero no se encontró un registro de las monedas (no cripto).

Se agrega el botón de convertir para realizar la acción de calcular el precio de la cantidad de criptomonedas seleccionada cuando el usuario crea que sus parámetros están listos.

El sistema no tiene otros módulos desarrollados en frontend y la modernización del UI no afecta la coherencia en la aplicación.