# **RETO 1**

En el área de TI de UTP surgió el requerimiento de realizar un nuevo proceso de autenticación. Nuestro scrum master registro los siguientes requerimientos:

* Nueva funcionalidad de login
* Nueva funcionalidad de registro

|  |  |
| --- | --- |
| **Historia de usuario** | Como usuario nuevo deseo poder registrarme en la web y poder loguearme de manera satisfactoria |
| **Criterio de Aceptación** |  |
| El correo ingresado debe tener un @ para ser válido. | * Dado que ingreso un correo valido   Cuando seleccione registrar  Luego La web debe registrar el usuario |
| La contraseña debe tener como mínimo 5 caracteres | Dado que ingreso una contraseña con 5 caracteres  Cuando se seleccione el botón registrar  Luego la web debe registrar el usuario |
| El correo y la contraseña son requeridos obligatoriamente. | Dado que ingreso el correo y la contraseña  Cuando selecciono “registrar”  Luego la web debe registrar el usuario |
| El sistema debe mostrar un mensaje de error al ingresar campos no válidos. | * Dado que ingreso “liseht.234” como correo   Cuando seleccione registrar  Luego la web debe mostrar mensaje “INVALID”   * Dado que ingreso “admi” como contraseña   Cuando seleccione “registrar”  Luego la web debe mostrar mensaje “INVALID”   * Dado que no ingreso la contraseña   Cuando selecciono “registrar”  Luego la web debe visualizar un mensaje "REQUIRED"   * Dado que no ingreso el correo   Cuando selecciono “registrar”  Luego la web debe visualizar un mensaje "REQUIRED" |

|  |  |
| --- | --- |
| **Historia de usuario** | Como usuario nuevo deseo poder loguearme de manera satisfactoria |
| **Creiterio de Aceptación** | **TestCase**qa-challenge-reto[nombre |
| El correo ingresado debe tener un @ para ser válido. | * Dado que ingreso un correo valido   Cuando seleccione registrar  Luego La web debe registrar el usuario |
| La contraseña debe tener como mínimo 5 caracteres | Dado que ingreso una contraseña con 5 caracteres  Cuando se seleccione el botón registrar  Luego la web debe registrar el usuario |
| El correo y la contraseña son requeridos obligatoriamente. | Dado que ingreso el correo y la contraseña valida  Cuando selecciono “registrar”  Luego la web debe registrar el usuario |
| El sistema debe mostrar un mensaje de error al ingresar campos no válidos. | * Dado que ingreso “liseht.234” como correo   Cuando seleccione registrar  Luego la web debe mostrar mensaje “INVALID”   * Dado que ingreso “admi” como contraseña   Cuando seleccione “registrar”  Luego la web debe mostrar mensaje “INVALID”   * Dado que no ingreso la contraseña   Cuando selecciono “registrar”  Luego la web debe visualizar un mensaje "REQUIRED"   * Dado que no ingreso el correo   Cuando selecciono “registrar”  Luego la web debe visualizar un mensaje "REQUIRED" |

1. Tipos de Pruebas a realizar:
2. Menciona que técnica de diseño de casos de prueba usarías para poder generar dichos casos

* Técnica de Caja Blanca Register:
  + Prueba Parición de equivalencia
    - Email
      * Validus {‘liseht@gmial.com’}
      * No validos {‘liseht’}
      * No valido: {“ “}
    - Password
      * Valido: tamaño(password) >5
      * Valido: tamaño(password) =5
      * No valido: tamaño(password) <5
      * No valido: {“ “}

|  |  |  |  |
| --- | --- | --- | --- |
| Entrada | Tipo de clase | Clases Válidas | ID |
| Email | valida | email contiene “@ | v\_e |
| Email | no valida | email no contiene “@” | nv\_e |
| Email | no valida | email es vació | nv\_e1 |
| password | valido | password.length >5 | v\_p |
| password | valido | Password.length = 5 | v\_p1 |
| password | no valido | Password.length < 5 | nv\_p |
| password | no valido | Password es vacío | nv\_p1 |
|  |  |  |  |

Creamos los test case con todas las combinaciones posibles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CP | Equivalencia | Entradas | | Resultado esperado |
|  |  | email | pass |
| CP1 | v\_e, v\_p | lis@gmail.com | admin1 | SAVED |
| CP2 | v\_e, v\_p1 | lis@gmail.com | admin | SAVED |
| CP3 | v\_e, nv\_p | lis@gmail.com | adm | INVALID |
| CP4 | v\_e, nv\_p1 | lis@gmail.com | “ “ | REQUIRED |
| CP5 | nv\_e, v\_p | lis | admin1 | INVALID |
| CP6 | nv\_e, v\_p1 | lis | admin | INVALID |
| CP7 | nv\_e, nv\_p | lis | adm | INVALID |
| CP7 | nv\_e, nv\_p1 | lis | “ “ | REQUIRED |
| CP5 | Nv\_e1, v\_p | “ “ | admin1 | REQUIRED |
| CP6 | nv\_e1, v\_p1 | “ “ | admin | REQUIRED |
| CP7 | nv\_e1, nv\_p | “ “ | adm | REQUIRED |
| CP7 | nv\_e1, nv\_p1 | “ “ | “ “ | REQUIRED |

* Técnica de Caja Blanca Login:
  + Prueba Parición de equivalencia
    - Email
      * Validus {‘liseht@gmial.com’}
      * No validos {‘liseht’}
      * No valido: {“ “}
    - Password
      * Valido: tamaño(password) >5
      * Valido: tamaño(password) =5
      * No valido: : tamaño(password) <5
      * No valido: {“ “}

|  |  |  |  |
| --- | --- | --- | --- |
| Entrada | Tipo de clase | Clases Válidas | ID |
| Email | valida | email contiene “@ | v\_e |
| Email | no valida | email no contiene “@” | nv\_e |
| Email | no valida | email es vació | nv\_e1 |
| password | valido | password.length >5 | v\_p |
| password | valido | Password.length = 5 | v\_p1 |
| password | no valido | Password.length < 5 | nv\_p |
| password | no valido | Password es vacío | nv\_p1 |
|  |  |  |  |

Creamos los test case con todas las combinaciones posibles

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CP | Equivalencia | Entradas | | Resultado esperado |
|  |  | email | pass |
| CP1 | v\_e, v\_p | lis@gmail.com | admin1 | SAVED |
| CP2 | v\_e, v\_p1 | lis@gmail.com | admin | SAVED |
| CP3 | v\_e, nv\_p | lis@gmail.com | adm | INVALID |
| CP4 | v\_e, nv\_p1 | lis@gmail.com | “ “ | REQUIRED |
| CP5 | nv\_e, v\_p | lis | admin1 | INVALID |
| CP6 | nv\_e, v\_p1 | lis | admin | INVALID |
| CP7 | nv\_e, nv\_p | lis | adm | INVALID |
| CP7 | nv\_e, nv\_p1 | lis | “ “ | REQUIRED |
| CP5 | Nv\_e1, v\_p | “ “ | admin1 | REQUIRED |
| CP6 | nv\_e1, v\_p1 | “ “ | admin | REQUIRED |
| CP7 | nv\_e1, nv\_p | “ “ | adm | REQUIRED |
| CP7 | nv\_e1, nv\_p1 | “ “ | “ “ | REQUIRED |

1. Identifica, registra y describe los bugs o mejoras que consideres necesarios .

Incidencias de back:

Issue 1: La validación de email no es correcta

**Descripción:**

**Request URL**:

http://localhost:3000/register

**Request:**

{

"email": "@",

"password": "admin123"

}

**Response**:

{

"code": 201,

"msg": "SAVED"

}

**Result ado** **Actual**:

Permite realizar el registro

**Result ado desperado:**

El servicio debe mostrar mensaje de email incorrecto

Issue 1: La validación de email no es correcta en el registro

**Descripción:**

**Request URL**:

http://localhost:3000/register

**Request:**

{

"email": "@",

"password": "admin123"

}

**Response**:

{

"code": 201,

"msg": "SAVED"

}

**Resultado** **Actual**:

Permite realizar el registro

**Resultado esperado:**

El servicio debe mostrar mensaje de email incorrecto

Issue 2: El servicio de registro no valida si el usuario ya existe

**Descripción:**

**Request URL**:

http://localhost:3000/register

**Request:**

{

"email": "liseht@gmail.com",

"password": "admin123"

}

**Response**:

{

"code": 201,

"msg": "SAVED"

}

**Resultado** **Actual**:

Permite realizar el registro

**Resultado esperado:**

El servicio debe mostrar mensaje indicando que el usuario ya existe

Issue 3: La validación de email no es correcta en el login

**Descripción:**

**Request URL**:

http://localhost:3000/login

**Request:**

{

"email": "@",

"password": "admin123"

}

**Response**:

{

"code": 201,

"msg": "SAVED"

}

**Resultado** **Actual**:

Permite realizar el login

**Resultado esperado:**

El servicio debe mostrar mensaje de email incorrecto

Issue 3: El login no está validando si el usuario existe

**Descripción:**

**Request URL**:

http://localhost:3000/login

**Request:**

{

"email": "liseht@gmail.com",

"password": "admin123"

}

**Response**:

{

"code": 201,

"msg": "SAVED"

}

**Resultado** **Actual**:

Permite realizar el registro

**Resultado esperado:**

El servicio debe mostrar mensaje indicando que el cliente no existeo

1. Implementa los tipos de pruebas automatizadas que consideres necesarios (unitarias, servicios, ui , perfomance), de tal manera que se pueda evaluar los requerimientos solicitados. Se debe crear un proyecto en una carpeta aparte llamada qaChallenge (utiliza el framework de automatización que prefieras y deberas subirlo a tu repositorio).

* Clonar el siguiente repositorio <https://github.com/LisehtBG/qaChallengeTest.git>

git checkout feature/automationApi

abrir el Proyecto Api-automation con IntellIJ IDE

ejecutar la clase runner

abrir el Proyecto challenge con eclipse

ejecutar la clase Login y register

# RETO 2

1. Comenta en base a tu criterio que nivel(es) de pruebas aplicaría para cada ambiente (puedes sugerir otros ambientes u estructura si es que lo consideras) (Coloca tus respuestas en el documento ( qa-challenge-reto[nombre].docx ) )

# RETO 3

### Caso:

Imagina una herramienta web, desarrollada con el objetivo de proporcionar una versión de ancho de banda ligero (se requiere un uso óptimo alrededor de 50 MB) de una herramienta principalmente utilizada por personas de todo el mundo para comprar productos de segunda mano.

Defectos:

1. Los precios no varían cuando una transacción se realiza en diferentes monedas.
2. Las transacciones entre usuarios no se reflejan en un informe de back-end.
3. Hay algunos errores ortográficos en la página de inicio.
4. Las credenciales de usuario no están cifradas en la base de datos.
5. Los foros oficiales de soporte no siempre registran los comentarios del usuario.
6. El ancho de banda necesario para una experiencia de usuario óptima es una velocidad de descarga de 100 MB.

Las prioridades se realizarán en base a las indicadas por negocio. Pero mi orden a corregirse sería la siguiente:

1. El ancho de banda necesario para una experiencia de usuario óptima es una velocidad de descarga de 100 MB.

Es el más prioritario ya que no está cumpliendo con la necesidad del requerimiento: (se requiere un uso óptimo alrededor de 50 MB)

1. Los precios no varían cuando una transacción se realiza en diferentes monedas.

Supongamos que el precio por defecto es $50, y el usuario cambia la moneda a soles. El usuario terminará pagando S/. 50.00 (esto genera una perdida monetaria en la empresa y multiplicándola con la analítica de cantidad de usuarios que realizan compras por día. A mas días sin resolver está incidencia, supone un incremento exponencial de las perdidas.

1. Las credenciales de usuario no están cifradas en la base de datos.

Fácilmente vulnerable para los ataques de injectionSQL. Problema grave de seguridad

1. Los foros oficiales de soporte no siempre registran los comentarios del usuario.

La experiencia de usuario no está siendo la mejor ya que pueden existir usuarios con errores graves en la web y que por motivo de que la data de los comentarios no está siendo procesada, estas incidencias nunca son atendidas. Se pierde la confianza del usuario y a nivel de negocio es algo grave ya que no ayuda a las conversiones.

1. Hay algunos errores ortográficos en la página de inicio.

Afecta a la experiencia de usuario, pero no es bloqueante para que el realice sus procesos frecuentes en la web.

1. Las transacciones entre usuarios no se reflejan en un informe de back-end.

Está sentencia se entiendo como que no se están generando logs de las transacciones de backend.

Esto no afecta directamente a la experiencia del usuario, pero para rastrear ciertas casuísticas que se dan en producción es bueno tener este artefacto para poder realizar un tracking log de algún error.

# RETO 4

### Agile testing

### Caso:

La aplicación web a probar es una plataforma como Mercado Libre. A continuación, podrás encontrar una lista de técnicas y procedimientos utilizados en las pruebas

El equipo de pruebas :

1. Utiliza el mismo entorno que los usuarios finales para realizar las pruebas perfomance.(correcto)

Se recomienda realizar las pruebas en el entorno de producción ya que solo ahí se tendrá las mismas configuraciones del servidor (serán pruebas reales)

1. Las pruebas de humo se ejecutan para cubrir todas las posibles opciones del usuario y se realizan en cada implementación de una nueva versión, todas las pruebas se realizan manualmente para reflejar la misma experiencia del usuario*.(incorrecto)*

Las pruebas de humo se deben realizar para validar el happypath, básicamente si la compilación del escenario principal falla, la funcionalidad debe ser devuelta de inmediato al equipo de desarrollo.

1. Las pruebas de regresión solo cubren los módulos que probablemente se vean afectados por los cambios realizados en la última versión. (correcto)

Las pruebas de regresión verificaran todas las funcionalidades cuyos módulos probablemente se han visto afectados por algún cambio introducido. Se recomienda realizar las siguientes pruebas en ese orden de tiempo:

Smoke test -> Re test -> sanity test -> integration test

1. Los casos de prueba se crean en Excel para que sean fáciles de editar. Los casos de prueba se eliminan después de la implementación de esa versión, por lo que se crean nuevos completos en cada iteración. *(incorrecto)*

Los casos de prueba se deben almacenar en una herramienta de gestión de pruebas: Test Rail, Qmetry, etc. Los casos de prueba nunca se deben eliminar ya que sirven de guía para tener un histórico de los test y sus ejecuciones.

1. Las pruebas comienzan a estar involucradas en el proceso de desarrollo desde la etapa inicial. *(incorrecto)*

Las pruebas inician desde las revisiones (reviews). En este punto se pueden identificar defectos en las especificaciones, defectos en el diseño y/ arquitectura, etc.