21 DE ABRIL DE 2022 [0222TDTE1M1C1LAED1021FT]

# Testing I

# **Examen integrador**

Les pedimos que lean atentamente las siguientes consignas y respondan a las preguntas de acuerdo a lo solicitado.

**No se aceptarán links de Drive, solo documentos adjuntos. Caso contrario, el examen no será considerado para su corrección.**

Nota aclaratoria: al enviar el formulario con el adjunto se debe esperar la confirmación del profesor **antes de salir de la sala de Zoom** para garantizar que se recibió correctamente para posterior corrección. Caso contrario, no se recibirá la evaluación y el alumno deberá recuperar esta instancia de evaluación. **Solo se recibirá 1 (un) documento por alumno**.

Recordá que cada ejercicio vale 1 punto.

**Duración**: 1 hora 30 minutos.

**Nombre y Apellido:** *Jairo Rivera López*

Parte teórica

1. ¿Cuál es la diferencia entre las pruebas de **humo** y las pruebas **regresión**?

R/. Mientras que las pruebas de regresión son aplicadas para verificar que un cambio aplicado no ha afectado la integridad de un sistema, las pruebas de humo se aplican para corrobar que un software funciona correctamente antes de avanzar a una etapa de desarrollo más compleja (o antes de ser entregado).

1. Explique con un ejemplo un tipo de **técnica de prueba de caja blanca**.

R/. Se elabora un unit testing para validar que la función “sumar”, de una calculadora, retorna el resultado esperado entre la suma de dos números enteros.

1. Mencionar las principales diferencias entre **debugging** y **testing**.

R/. Mientras que testing encuentra los defectos en un desarrollo, debugging (conociendo dichos defectos) se encarga de resolverlos. El proceso de testing es realizado por el tester y el proceso de debugging es realizado por el desarrollador. Mientras que el proceso de testing puede ser automatizado / manual, el debugg es realizado de forma manual. El proceso de debug es exclusivamente realizado por alguien del equipo interno, mientras que el proceso de testing puede ser llevado a cabo por personas del equipo o personas externas (como el cliente).

1. ¿En qué ambiente aplicarías **pruebas unitarias**? Justifique su respuesta.

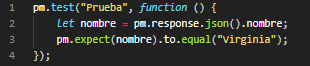
R/: Lo aplicaría en el ambiente DEV, ya que este abarca el proceso de desarrollo en código y es donde se verificaría si un componente se ejecuta correctamente.

1. ¿Cuál es el objetivo de una prueba de **confirmación**?

R/. Verificar que un defecto ha sido resuelto correctamente.

Parte práctica

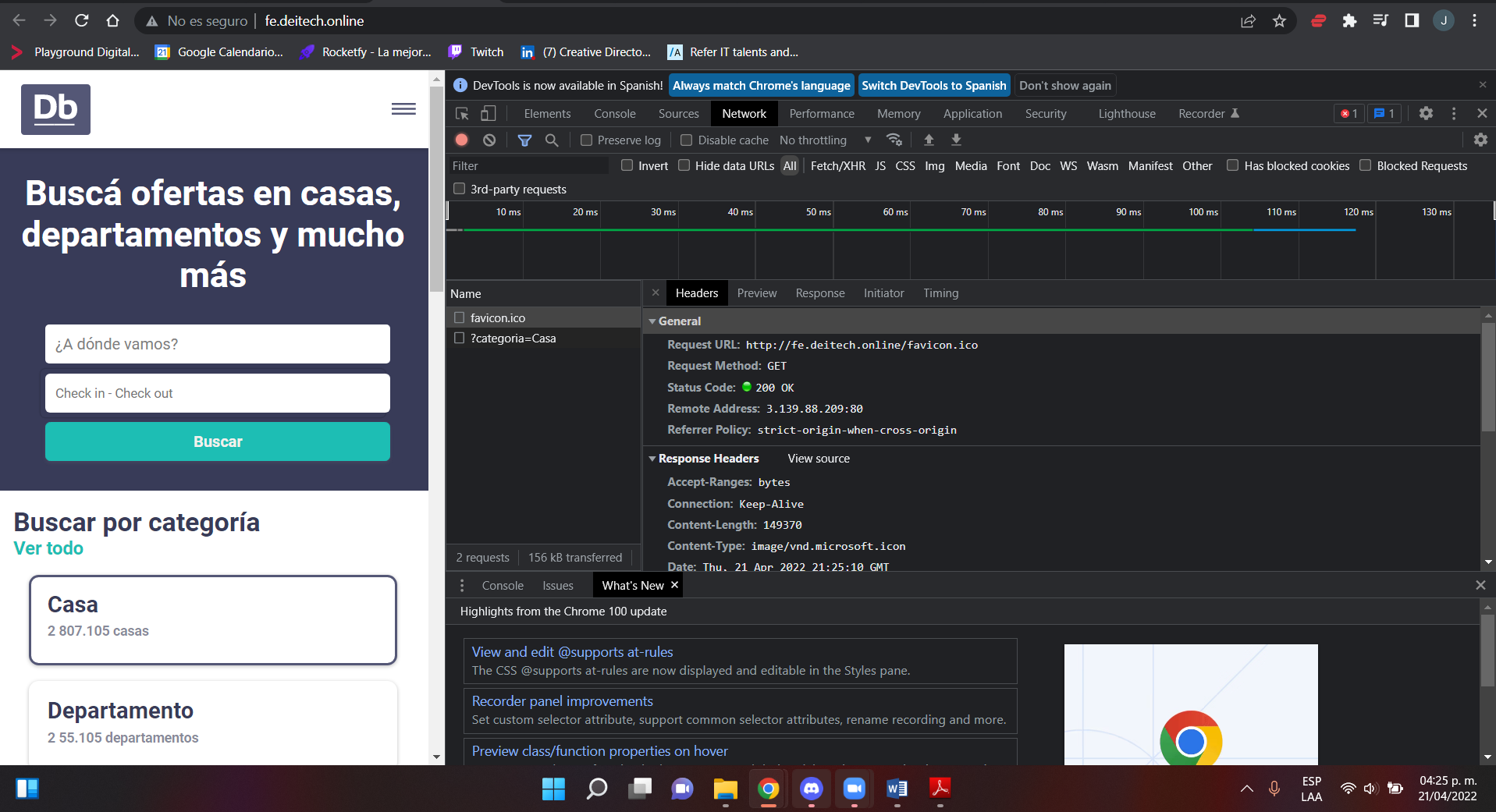
1. ¿Qué nos permite validar el siguiente **test de Postman**?

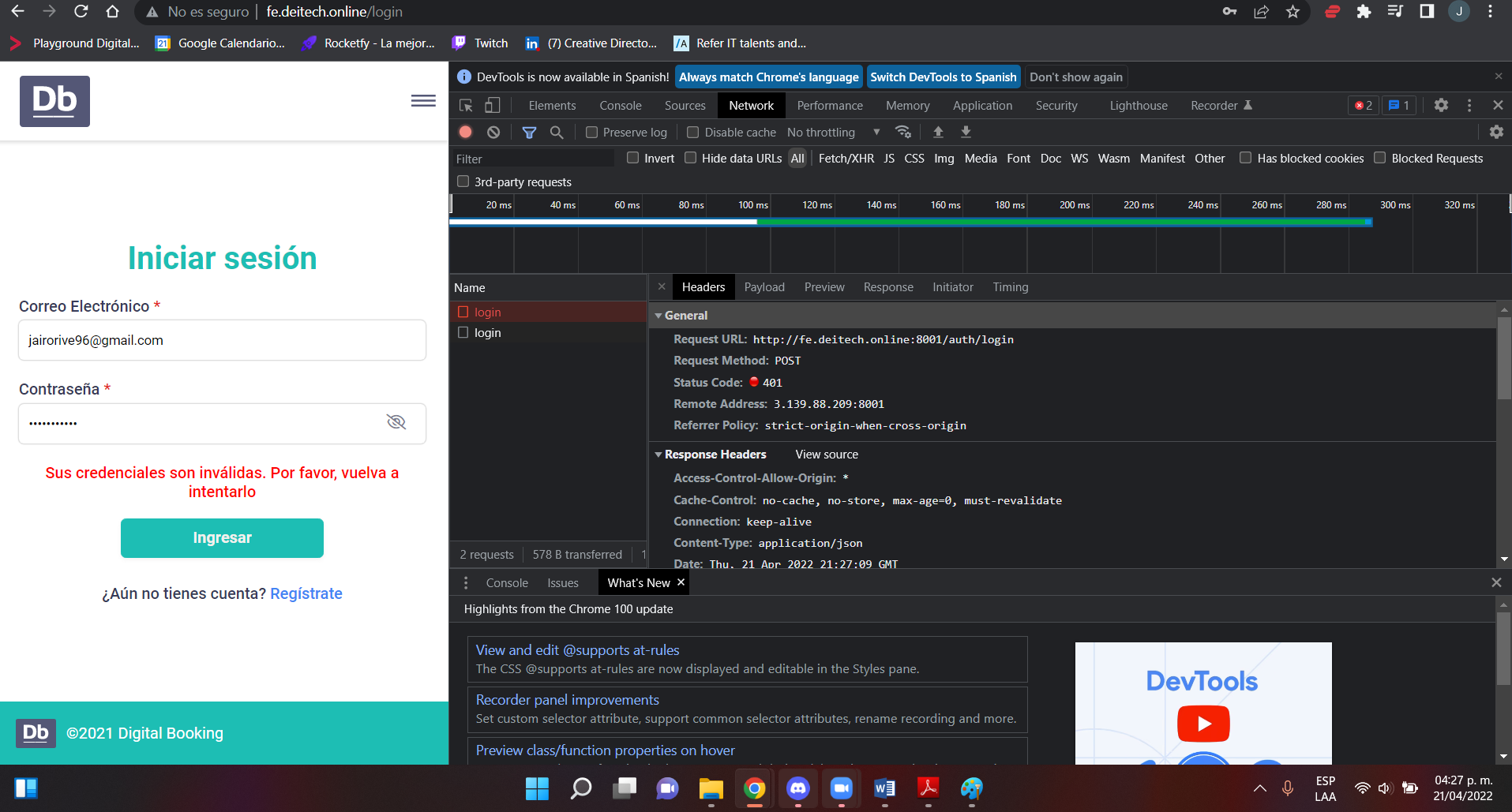


R/. Nos permite validar si la respuesta de una petición a un servidor o API (en este caso un nombre), es igual al esperado (“Virginia”). Para ello, utiliza el método json que resolverá la promesa en un objeto de Javascript (y así poder hacer el test con lo esperado).

1. Identificar una petición GET y una POST de nuestra app [**Digital Booking!**](http://fe.deitech.online/), (Recomendación: Utilizar la herramienta de desarrollo > DevTools) Explicar brevemente de qué se trata una petición POST.

R/. GET:



R/.POST: En la imagen se observa la interfaz de login de la página, donde se ingresan los datos de correo y contraseña. Al darle al botón ingresar, se hace una solicitud post a la API y esta nos devuelve un mensaje verificando si los datos ingresados cumplen o no con el usuario previamente creado. En este caso, arroja un código 401 que significa que las credenciales no fueron debidamente autenticadas:  
  


1. Redactar brevemente **1 caso de prueba** **negativo** que aplicarías en la página de [**Digital Booking!**](http://fe.deitech.online/) (**No se requiere escribirlos en formato de template**).

R/. Un usuario que ingrese datos incorrectos en la sección de “iniciar sesión”, no podrá “ingresar” a la página desde el perfil de usuario registrado.

1. Mencionar **1 defecto** que encuentres en la página de [**Digital Booking!**](http://fe.deitech.online/) (**No se requiere escribirlos en formato de template**).

R/. Una vez completada la reserva, la información de la misma deberá ser enviada al usuario con todos los datos completos y detallados. El email es enviado con la información, excepto la dirección detallada (aparece "null").

1. Si estoy trabajando con **Jest** y quiero validar que el resultado devuelto sea **false**. ¿Qué **matcher** puedo utilizar? Dar un ejemplo de un posible test para cualquier sistema bajo prueba

R/. .toBeFalsy / Se realiza un test en una base de datos donde el registro de una columna de una tabla tiende a ser null, ya que suele no ser ingresada cuando se crea. En este caso, se redactaría un test así:

test('Resultado Registro, () => {

expect(registro.fechaFinal).toBeFalsy();

});