# 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**.

**Duración**: 1 hora 30 minutos.

**Nombre y Apellido:** Lucia Alajardo

Parte teórica

1. Indicar cuales son los objetivos específicos de las pruebas de componentes Objetivos específicos.

* Prevenir la propagación de defectos.
* Generar confianza en la calidad del componente.
* Encontrar defectos en el componente.
* Reducir riesgo.

1. Explique con sus palabras el ciclo de vida de un defecto.

Se le llama “ciclo de vida de un defecto” al proceso que dirige un defecto desde el descubrimiento hasta la solución. Dicho dicho pasa por diferentes estados:

* Inicio.
* Asignado.
* En proceso.
* Corregido.
* En espera de verificación.
* Verificación.
* Cerrado.

1. Explicar con un ejemplo la diferencia entre una **prueba funcional** y una **prueba no funcional**.

**Prueba funcional**, evalúa las funciones que el sistema tendría que realizar, ejemplo si en un campo en el cual deberíamos completar con números, no tendría que permitir el ingreso de letras.

**Prueba no funcional,** evalúa el comportamiento frente a situaciones diferentes, por ejemplo, a una página hacemos un sorteo y calculamos que se va a inscribir x cantidad de personas, y doblar el número, lo esperado es que la página colapse.

1. Leer el siguiente caso de prueba e indicar en base a qué técnica de prueba fue diseñado.

**CASO DE PRUEBA POSITIVO**: porque te permite realizar la acción sin problemas.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Título / Nombre** | **Descripción** | **Precondición** | **Pasos** | | | **Estado de diseño** | **Creado por** |
| **#** | **Acción** | **Resultado esperado** |
| Cambio de estado de pedido | Cambio de estado del pedido llamado "1xPOLLO" del estado "Nuevo" a "Confirmada" | Tener cuenta de usuario admin en ComidaYA | 1 | Ir a la URL:<https://ctd-app-resto.herokuapp.com/> | Visualizar el home de ComidaYA | Finalizado | JG |
| 2 | Ingresar mail (soyadmin@yopmail.com) y contraseña (testqa) | Loguearse en ComidaYA con rol de admin. |
| 3 | Visualizar que en el módulo "Pedidos", el pedido "1xPOLLO" tiene el estado "Nuevo" | Ubicar el registro con el pedido en cuestión |
| 4 | Cambiar el estado del pedido a "Confirmada" | Visualizar que se realiza el cambio del estado del pedido "1xPOLLO" en el listado de pedidos |

1. Mencionar que pruebas se realizan en el ambiente/entorno de **QA**

En el ambiente QA se realizan las siguientes pruebas:

* Pruebas funcionales.
* Pruebas de caso de uso.
* Pruebas de regresión.
* Pruebas de confirmación.
* Pruebas de humo.
* Entre otras.

# **Requerimientos Digital Booking**

**Digital Booking** es nuestro sitio de reservas: [**Ingresá desde acá**](http://fe.deitech.online/)

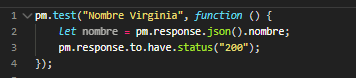
**Requerimientos:**

* El sistema deberá permitir el registro de un tipo de usuario “cliente”.
* Los datos necesarios serán: nombre, apellido, e-mail y contraseña.
* Se debe validar que todos los campos estén completos y tengan el formato correcto:
  + Nombre (de type=”text”)
  + Apellido (de type=”text”)
  + Email (de type=”email”)
  + Password (de type=”password”- min:6 caracteres)
  + Repetir password (de type=”password”- min:6 caracteres)
* El sistema deberá permitir loguear a los usuarios. En caso de que el e-mail o la contraseña sean incorrectos, se debe mostrar un mensaje de error.
* El sistema deberá permitir a los usuarios realizar una reserva. Los usuarios registrados pero no habilitados no pueden realizar reservas.
* Se habilitará un usuario con el link que llega al e-mail de registro.
* Al momento de realizar una reserva, el sistema deberá enviar un mail al usuario logueado con toda la información detallada de su reserva.
* El sistema deberá permitir a los usuarios realizar búsquedas de alojamientos según fecha y lugar.
* Los lugares se mostrarán mediante un listado y debe poder filtrar por palabras claves, mientras que la fecha no debe permitir seleccionar una fecha anterior a la actual y la estadía mínima permitida debe ser de 1 noche.
* En el caso de que no existan alojamientos disponibles para las fechas deseadas, el sistema deberá indicarle al usuario con un mensaje.
* El sistema permitirá la habilitación del usuario mediante el envío de un SMS.
* El sistema deberá permitir agrupar los alojamientos por tipo y también mostrar los más recomendados por los clientes.
* El sistema deberá permitir al usuario agregar alojamientos a favoritos, como así también, quitarlos.
* El sistema deberá permitir al usuario ver las reservas que realizó con los detalles de la misma.
* La aplicación debe ser responsive, es decir, debe adaptarse a las diferentes resoluciones del navegador hasta llegar a la versión móvil.
* La aplicación debe ser capaz de operar adecuadamente con hasta 100.000 usuarios con sesiones concurrentes.

Parte práctica

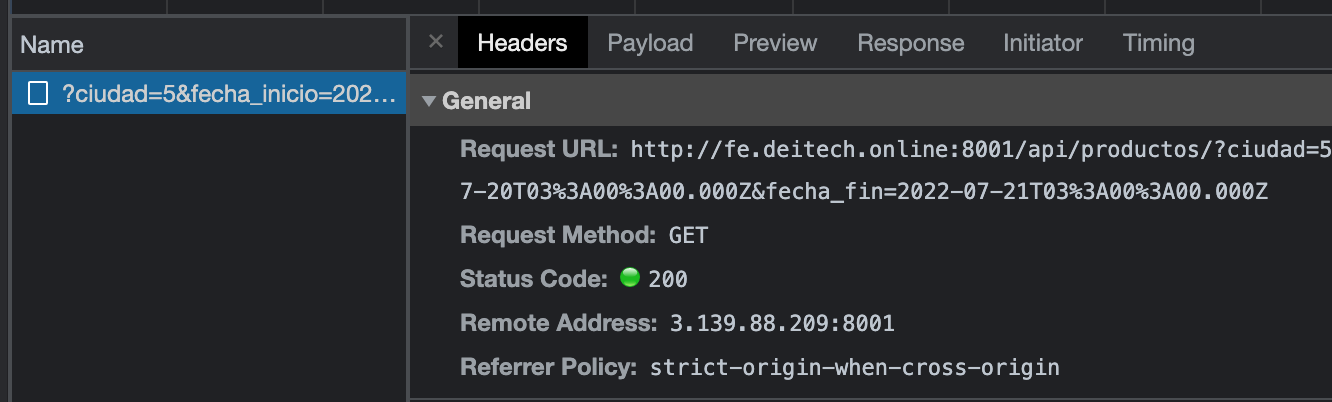
1. ¿Este **test de Postman** es correcto para validar si el contenido devuelto es igual al esperado? Justificar tu respuesta.

Si es correcto para validar que el contenido devuelto es igual que al esperado porque nos muestra el código de respuesta HTTP 200, que significa que actuó como se esperaba.

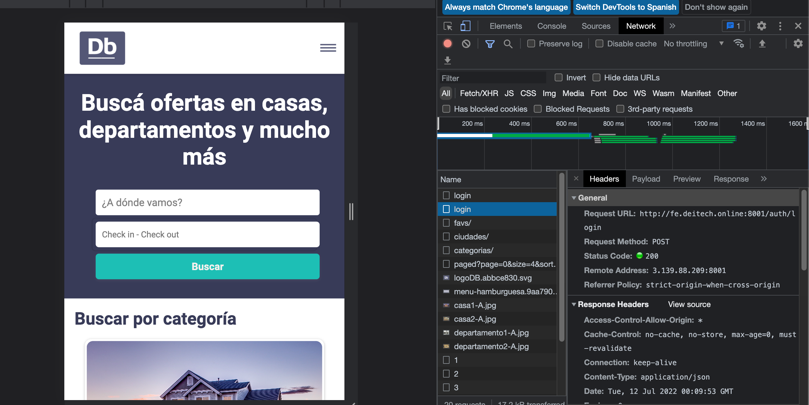


1. Identificar una petición GET y una POST de nuestra app [**Digital Booking!**](http://fe.deitech.online/), (Recomendación: Utilizar la herramienta de desarrollo > DevTools), Explicar brevemente de qué se trata una petición GET.

Get método por el cual realizamos peticiones al servidor. No podemos modificar la información mediante este, y toda información será visible.



En cambio post es más seguro, es un método en el cual se puede enviar datos al servidor, y sirve para crear un registro, por ejemplo en este caso, inicio sesión en booking, y esos datos no son visibles en la url.



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

**Caso de prueba positivo:** hacer una reserva para “Tigre, Buenos Aires”, el día 16/07/22, hasta el 23/07/22. Y que la página me permita la reserva.

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

Defecto en la página de Booking, que no puedo hacer reserva porque no me deja seleccionar los días, y al darle clic a reservar no pasa absolutamente nada.

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

Si busco en Jest que me devuelva el un resultado “true”, debo usar el matcher **.toBeTruthy.**