DEBUG & TESTING

DEBUG & TESTING

- **Que aprenderemos en este apartado?**

1. **Debug**
2. **Testing**
3. **Jest**

Principios de todos los modelos

- Cada actividad de desarrollo debe ser probada
- Ninguna porción del software puede quedar sin ser probada, si ha sido desarrollada tanto de forma en una única fase o iterativa
- Cada nivel de prueba debería ser probado de forma especifica
- Cada nivel de pruebas cuenta con sus objetivos de prueba propios
- Las pruebas llevadas a cabo en cada nivel deben reflejar estos objetivos
- El proceso de pruebas comienza con mucha antelación a la ejecución de pruebas
- Tan pronto como el desarrollo comienza puede comenzar la preparación de las pruebas correspondientes
- También es el caso de las revisiones de documentos comenzando por los conceptos, especificación y el diseño global (en conjunto).

Tipos de Pruebas

- Pruebas unitarias
- Pruebas de integración
- Pruebas de sistema
- Pruebas funcionales
- Pruebas de carga
- Pruebas de estrés
- Pruebas de aceptación

Pruebas unitarias

- Con ellas probamos las unidades del software, normalmente métodos.
- Por ejemplo, escribimos estas pruebas para comprobar si un método de una clase funciona correctamente de forma aislada.
- Por ejemplo, aunque estés probando un método que realice una serie de cosas y al final envíe un correo electrónico a través de un servidor de correo, en una prueba unitaria no tienes que probar que el correo se ha mandado correctamente.
- Buenas pruebas unitarias no irían contra una base de datos, por ejemplo, sino que simularían la conexión.

Pruebas de integración

- En este caso probamos cómo es la interacción entre dos o mas unidades del software.
- Este tipo de pruebas verifican que los componentes de la aplicación funcionan correctamente actuando en conjunto.
- Siguiendo con el caso anterior, las pruebas de integración son las que comprobarían que se ha mandado un email, la conexión real con la base de datos etc.
- Este tipo de pruebas son dependientes del entorno en el que se ejecutan. Si fallan, puede ser porque el código esté bien, pero haya un cambio en el entorno.

PRUEBAS DE SISTEMA

- Lo ideal sería realizar este tipo de pruebas después de las pruebas de integración.

- Aquí se engloban tipos de pruebas cuyo objetivo es probar todo el sistema software completo e integrado, normalmente desde el punto de vista de requisitos de la aplicación.

- Aquí aparecerían las pruebas funcionales, pruebas de carga, de estrés etc

PRUEBAS FUNCIONALES

- En este caso, el objetivo de las pruebas funcionales es comprobar que el software que se ha creado cumple con la función para la que se había pensado.
- En este tipo de pruebas lo que miramos, lo que nos importan, son las entradas y salidas al software. Es decir, si ante una serie de entradas el software devuelve los resultados que nosotros esperábamos.
- Aquí solo observamos que se cumpla la funcionalidad, no comprobamos que el software esté bien hecho, no miramos el diseño del software. Estudiamos el software desde la perspectiva del cliente, no del desarrollador.
- Por eso este tipo de pruebas entran dentro de lo que se llaman pruebas de caja negra: aquí no nos centramos en cómo se generan las respuestas del sistema, solo analizamos los datos de entrada y los resultados obtenidos.
- Una prueba funcional podría ser por ejemplo comprobar que en el formulario de mi página web si relleno un campo de teléfono con zzzz muestre un mensaje de campo no válido, y no me deje registrar.
- Estas pruebas pueden ser manuales, o automatizarse con herramientas. Una de las más conocidas para web es Selenium, en sus distintas variantes.

PRUEBAS DE CARGA

- Las pruebas de carga son un tipo de prueba de rendimiento del sistema. Con ellas observamos la respuesta de la aplicación ante un determinado número de peticiones.

- Aquí entraría por ejemplo ver cómo se comporta el sistema ante X usuarios que entran concurrentemente a la aplicación y realizan ciertas transacciones.

Pruebas de estrés

- Este es otro tipo de prueba de rendimiento del sistema.

- El objetivo de estas pruebas es someter al software a situaciones extremas, intentar que el sistema se caiga, para ver cómo se comporta, si es capaz de recuperarse o tratar correctamente un error grave.

PRUEBAS DE ACEPTACIÓN

- Por último, las pruebas de aceptación se realizan para comprobar si el software cumple con las expectativas del cliente, con lo que el cliente realmente pidió.

JEST

- Jest es una plataforma de testing desarrollada por el equipo de Facebook. Jest es utilizado por Facebook para testear todo su código JavaScript incluyendo las aplicaciones React.

- Una de sus filosofías es proporcionar una experiencia integrada de "configuración cero".

- La integración es muy sencilla ya que viene preconfigurado en el boilerplate create-react-app.

JEST

```
import React from 'react';
import { render } from '@testing-library/react';
import App from './App';

test('renders learn react link', () => {
  const { getByText } = render(<App />);
  const linkElement = getByText(/learn react/i);
  expect(linkElement).toBeInTheDocument();
});
```