

Actividad de Aprendizaje Semana 12

Indicaciones Generales:

1. El plagio se sanciona con la suspensión o expulsión del estudiante de la Universidad. Reglamento General de Estudios.
2. Sea cuidadoso con su redacción, la cual formará parte de su calificación (aspectos léxicos, sintácticos, semánticos).
3. Lea bien la pregunta o enunciado antes de proceder a su desarrollo, administre su tiempo eficazmente.

Actividad 1.

1. Investigue los siguientes temas según el Grupo asignado.
2. Elaborare una presentación donde contemple los puntos establecidos
3. Adjuntar el archivo a su Canvas, en la tarea "Semana 12– Taller 3".



Fecha límite de presentación: Hoy

Objetivo del Taller

Aplicar pruebas unitarias y de integración con Jest en Node.js, integrando ambas en un flujo de **Integración Continua (CI)** con GitHub Actions. Los estudiantes comprobarán cómo las pruebas se ejecutan automáticamente ante cambios en el código.

Indicaciones Generales

1. Este taller se realiza en **grupos de trabajo (7 grupos)**.
2. Cada grupo deberá crear desde cero:
 - Una **función unitaria propia** con su **prueba Jest**.
 - Una **prueba de integración** combinando su función con una función base común.
 - Un **flujo CI/CD en GitHub Actions** que ejecute ambas pruebas.
3. **No se entrega código base completo**, solo la función principal común.
4. El taller dura 90 minutos.

Parte 1 – Preparación del entorno

1. Crear una carpeta para el laboratorio:
2. `mkdir laboratorio-ci`
3. `cd laboratorio-ci`
4. Inicializar un proyecto Node.js:
5. `npm init -y`
6. Instalar Jest:
7. `npm install --save-dev jest`
8. `npm pkg set scripts.test="jest"`
9. Crear un repositorio nuevo en GitHub y subir el proyecto.

Función base común (usada por todos los grupos)

Archivo: **pedido.js**

```
function calcularTotalPedido(productos, descuento) {
  if (!Array.isArray(productos) || productos.length === 0)
    return "Error: no hay productos en el pedido";

  const subtotal = productos.reduce((acc, p) => acc + p.precio * p.cantidad, 0);
  if (subtotal <= 0) return "Error: monto inválido";

  const total = subtotal - subtotal * (descuento / 100);
  return total;
}

module.exports = { calcularTotalPedido };
```

Descripción:

Esta función calcula el total de un pedido con descuento.

- Si no hay productos o el subtotal es 0, devuelve un mensaje de error.
- El total = subtotal – (subtotal × descuento/100).

Parte 2 – Tarea por grupo

Cada grupo debe **crear su propia función secundaria**, que **utilice calcularTotalPedido()** y agregue una validación o regla de negocio adicional.

Luego crearán su **prueba unitaria** (2 casos: éxito y error).

Grupo	Función a crear	Qué debe validar específicamente
1	verificarLimite()	Rechaza pedidos mayores a S/5000
2	validarDescuento()	Solo permite descuentos entre 0 y 50 %
3	calcularIGV()	Aplica IGV del 18 % al total
4	validarCliente()	Rechaza pedidos sin nombre de cliente
5	verificarStock()	Rechaza productos con cantidad 0
6	generarResumen()	Devuelve texto con cliente, cantidad y total
7	aplicarCupon()	Aplica cupón adicional si el total supera S/3000

Archivos que debe crear cada grupo:

- funcionGrupoX.js (lógica del grupo)
- funcionGrupoX.test.js (prueba unitaria del grupo)

Parte 3 – Prueba de integración

Todos los grupos deben crear una **prueba de integración** donde combinen:

- la función base calcularTotalPedido()
- su propia función secundaria.

Archivo: integracion.test.js

Debe simular un flujo como:

Calcular el total → aplicar la regla del grupo → validar el resultado final (correcto o error).

Cada grupo diseña su caso propio con **al menos un test** que verifique la interacción entre funciones.

Parte 4 – Flujo CI/CD (Integración Continua)

1. Crear carpeta:
2. mkdir -p .github/workflows
3. Dentro, crear el archivo ci.yml con un flujo que:
 - Se ejecute al hacer **push** en la rama main.
 - Instale dependencias y ejecute todas las pruebas Jest.

Los pasos deben incluir:

- checkout del código.
- setup-node con versión 20.
- npm install
- npm test

El pipeline debe aparecer y correr automáticamente en la pestaña **Actions** de GitHub.

Parte 5 – Entregables

Cada grupo debe entregar en Canvas o aula virtual:

1. Captura del flujo **en verde** (todas las pruebas pasan).
2. Captura del archivo ci.yml en GitHub.
3. Evidencia del resultado de las pruebas en consola.
4. Breve reflexión (3 a 5 líneas):
 - Qué validó su prueba unitaria.
 - Qué validó la prueba de integración.
 - Qué tipo de error podría detectar antes del despliegue.

Rúbrica de calificación

Criterios	Destacado (5)	Logrado (4)	En proceso (3)	En inicio (2)	Ausente (0)
Claridad conceptual	Explicación completa, rigurosa y clara de los conceptos técnicos: pruebas unitarias, integración y CI/CD. Usa correctamente la terminología técnica sin errores.	Explicación clara con pequeños errores o imprecisiones conceptuales.	Presenta los conceptos clave, pero con confusión o errores técnicos.	Muestra dificultades para explicar con claridad la relación entre pruebas y CI/CD.	No aborda los conceptos técnicos o confunde completamente los términos.
Aplicación práctica del tema	La práctica funciona correctamente y demuestra dominio del tema: las pruebas unitarias e integración se ejecutan en GitHub Actions sin errores.	El flujo CI/CD funciona con pequeños ajustes o advertencias menores.	La práctica está incompleta o contiene fallos, pero se observa la intención de aplicar el flujo CI/CD.	Solo parte de la práctica se realizó o las pruebas no se ejecutan correctamente.	No presenta actividad práctica ni evidencias funcionales.
Material visual (capturas o evidencias)	Capturas claras y completas del flujo CI/CD, resultados de pruebas y estructura del proyecto. Presentación visual ordenada y bien explicada.	Evidencias adecuadas y alineadas al tema, aunque con menor detalle o estética.	Presenta evidencias básicas o incompletas (una o dos imágenes poco explicativas).	Evidencias mínimas o poco legibles, sin contexto claro.	No presenta material visual o capturas del trabajo realizado.

Trabajo en equipo y distribución de roles	Excelente coordinación del grupo: todos los miembros participaron con roles definidos (codificación, documentación, revisión). Se evidencia comunicación efectiva.	Buena participación general con roles visibles, aunque no todos los integrantes tuvieron el mismo nivel de aporte.	Participación desigual o poco clara; algunos miembros no se integran completamente.	Solo uno o dos miembros realizaron la mayor parte del trabajo; poca colaboración.	No se evidencia trabajo grupal ni colaboración entre miembros.
--	--	--	---	---	--