Universidad Continental

Taller de proyectos 2 – ingeniería de Sistemas e informática

INSTRUCCIONES PARA PROBAR LAS RUTAS API EN PROYECTOS MERN

1. Objetivo

Diseñar e implementar un conjunto exhaustivo de pruebas para las rutas API del proyecto de fin de asignatura, aplicando múltiples metodologías de testing para garantizar la calidad y confiabilidad del backend.

2. Pruebas con Postman

2.1. Crear una o más colecciónes y requests en Postman con nombres significativos.

2.2. Organizar las rutas en carpetas lógicas (Ej: "Autenticación", "Usuarios", "Productos").

2.3. Incluir tests automatizados en el Script Tests de cada request.

2.4. Exportar la colección completa en formato JSON.

Ejemplo de Estructura:

```
Nombre de la aplicación
├── 🔐 Autenticación
│   ├── POST Registrar Usuario
│   ├── POST Login Usuario
│   └── GET Verificar Token
├── 👥 Gestión de Usuarios
│   ├── GET Obtener Perfil
│   ├── PUT Actualizar Usuario
│   └── DELETE Eliminar Cuenta
└── 🛍️ Productos
    ├── GET Listar Productos
    ├── POST Crear Producto
    └── GET Obtener Producto por ID
```

3. Pruebas con cURL

3.1. Crear un archivo pruebas_curl.txt

3.2. Para cada comando, incluir:

a. Título descriptivo

b. Descripción (40-50 palabras)

c. Comando cURL funcional

d. Ejemplo de respuesta esperada

```
Ejemplo de Entrada:
=== CREACIÓN DE USUARIO ===
Descripción: Este comando prueba el endpoint de registro de usuarios,
enviando datos básicos como nombre, email y contraseña. Verifica que el
servidor responda con un token JWT y los datos del usuario creado
exitosamente.

Comando:
curl -X POST http://localhost:5000/api/usuarios/registro \
  -H "Content-Type: application/json" \
  -d '{"nombre": "Juan Perez", "email": "juan@ejemplo.com", "password":
"secure123"}'

Respuesta Esperada:
{"token": "eyJhbGciOiJ...", "usuario": {"id": "123", "nombre": "Juan Perez",
"email": "juan@ejemplo.com"}}
```

4. Mock API con JSON Server

Simular una API no implementada para permitir el desarrollo frontend sin depender del backend completo.

- Crear archivo mock/db.json con datos de prueba
- Configurar JSON Server para simular endpoints
- Incluir relaciones entre datos cuando sea necesario
- Publicar en el repositorio.

5. Pruebas Unitarias con Supertest

Implementar una prueba automatizada para endpoints API usando Jest y Supertest, y publicarlo en el repositorio.

6. Estructura de Entregables

Cada una de los entregables deben cumplir con la convención de nombres para documentos consignados en las instrucciones para gestión de proyectos en GitHub.

- coleccion_postman.json - Colección exportada de Postman
- pruebas_curl.txt - Comandos cURL documentados
- rutas_archivos.txt - Ubicación de implementación del Mock y el Supertest en el repositorio.

Universidad Continental

Taller de proyectos 2 – ingeniería de Sistemas e informática

Rúbrica de evaluación: pruebas de rutas API en proyectos MERN

Criterio/Indicador	Sobresaliente (3)	Suficiente (2)	En desarrollo (1)	Insatisfactorio (0)
Pruebas con Postman: Organización de colecciones en carpetas lógicas; requests con nombres significativos; tests automatizados en Script Tests para cada endpoint.	Colección organizada en ≥3 carpetas lógicas; nombres claros y descriptivos; tests automatizados verifican status, estructura JSON y datos clave en todos los requests.	Colección organizada en 2 carpetas; nombres adecuados; tests automatizados cubren status y respuesta básica en la mayoría de requests.	Organización básica o 1 carpeta; nombres genéricos; tests solo en algunos requests o verifican solo el código de estado.	Sin organización en carpetas; nombres no descriptivos; sin tests automatizados o con errores que impiden ejecución.
Exportación de Postman: Colección exportada en formato JSON; archivo correctamente nombrado y ubicado; estructura	Archivo coleccion_postman.json exportado correctamente; incluye todos los requests, tests y variables; listo para importar y ejecutar.	Archivo exportado con nombre correcto; contiene la mayoría de requests y tests; mínimos errores de estructura.	Archivo exportado pero incompleto o con nombre incorrecto; faltan requests o tests importantes.	Archivo no exportado, corrupto o con estructura incorrecta que impide su uso.

Taller de proyectos 2 – ingeniería de Sistemas e informática

Criterio/Indicador	Sobresaliente (3)	Suficiente (2)	En desarrollo (1)	Insatisfactorio (0)
completa y sin errores.				
Pruebas con cURL: Archivo pruebas_curl.txt con comandos funcionales; incluye título, descripción (40-50 palabras), comando y respuesta esperada para cada endpoint.	Archivo completo con ≥6 comandos; cada uno con estructura clara y descripción detallada; comandos funcionales y respuestas coherentes con la API.	Archivo con 4-5 comandos; estructura básica cumplida; descripciones breves pero claras; comandos mayormente funcionales.	Archivo con 2-3 comandos; estructura inconsistente; descripciones incompletas o comandos con errores.	Menos de 2 comandos; estructura no seguida; descripciones ausentes o comandos no funcionales.
Mock API con JSON Server: Creación de mock/db.json con datos de prueba; configuración de JSON Server; simulación de	Archivo db.json con ≥2 entidades relacionadas; datos realistas y variados; JSON Server configurado y endpoints simulados correctamente.	Archivo db.json con 1-2 entidades y datos básicos; configuración funcional pero sin relaciones complejas.	Archivo db.json con una entidad y datos mínimos; configuración parcial o con errores menores.	Sin archivo db.json o con estructura incorrecta; JSON Server no configurado

Taller de proyectos 2 – ingeniería de Sistemas e informática

Criterio/Indicador	Sobresaliente (3)	Suficiente (2)	En desarrollo (1)	Insatisfactorio (0)
endpoints con relaciones entre datos.				o no funcional.
Pruebas Unitarias con Supertest: Implementación de pruebas automatizadas para endpoints API usando Jest y Supertest; estructura clara y cobertura de casos.	Pruebas implementadas para ≥5 endpoints; cubren éxito, errores y validaciones; código limpio y bien estructurado; ejecución sin fallos.	Pruebas para 3-4 endpoints; cubren casos principales; estructura aceptable y ejecución mayormente exitosa.	Pruebas para 1-2 endpoints; cobertura limitada o con errores en algunos casos.	Sin pruebas implementadas o con errores graves que impiden su ejecución.
Estructura de Entregables: Archivos entregados según convención de nombres; ubicación correcta en repositorio;	Todos los archivos con nombres correctos y ubicados según estándar; rutas_archivos.txt detallado y preciso; estructura fácil de seguir.	Archivos con nombres correctos; rutas_archivos.txt presente pero con detalles mínimos; estructura clara.	Algunos archivos con nombres incorrectos o ubicación errónea; rutas_archivos.txt incompleto o impreciso.	Entregables faltantes; nombres incorrectos; sin rutas_archivos.txt o con información irrelevante.

Universidad Continental

Taller de proyectos 2 – ingeniería de Sistemas e informática

Criterio/Indicador	Sobresaliente (3)	Suficiente (2)	En desarrollo (1)	Insatisfactorio (0)
documentación clara en rutas_archivos .txt.				