Rúbrica Evaluación Unidad 3

DESCRIPCIÓN DE LOS NIVELES DE DESEMPEÑO				
Nivel	**Bajo**	**Medio**	**Alto**	**Sobresaliente**
Descripción	Demuestra un dominio limitado del indicador. Su desempeño es impreciso, incompleto e inconsistente.	Demuestra un dominio básico del indicador. Su desempeño es parcialmente preciso y completo.	Demuestra un buen dominio del indicador. Su desempeño es preciso, completo y generalmente consistente.	Demuestra un dominio excepcional del indicador. Su desempeño es preciso, completo y consistente.

INDICADOR	**Niveles de desempeño**			
	Bajo	**Medio**	**Alto**	**Sobresaliente**
3.1.1 Diseña operaciones CRUD, siguiendo el framework definido, para manipular datos en la aplicación.	Define 0 o 1 endpoint funcional. Las rutas no respetan el formato requerido (/api/v1/gestores, /api/v1/empresaTipo), se omite el versionado o se mezclan recursos en un mismo archivo. No existe una separación clara entre recursos ni coherencia REST.	Define los 2 endpoints (gestores y empresaTipo), pero al menos uno presenta problemas importantes en la ruta (por ejemplo parámetros mal definidos, mezcla de query y path, versionado inconsistente) o en la organización de archivos PHP. La estructura REST es parcial.	Define correctamente los 2 endpoints REST en archivos separados, con rutas claras (/api/v1/gestores, /api/v1/empresaTipo), respetando el versionado y la separación por recurso. La estructura de carpetas y archivos es mayormente coherente con el diseño solicitado.	Define de forma ejemplar los 2 endpoints REST, con rutas limpias, versionado correcto, separación por recurso y una estructura de carpetas clara y escalable. La definición de endpoints refleja buenas prácticas REST (nombres en plural, sin verbos en la URL, parámetros bien definidos) y facilita el mantenimiento del código.
	0 puntos	**6 puntos**	**8 puntos**	**10 puntos**
3.1.2 Implementa operaciones CRUD (Crear, Leer, Actualizar y Eliminar) de manera eficiente y precisa, para acceder y	Implementa 5 o menos de los 12 métodos CRUD requeridos (entre ambos endpoints) o la mayoría no funciona (no modifican datos, no retornan JSON válido o usan métodos HTTP	Implementa correctamente entre 6 y 9 métodos de los 12 requeridos. Algunos métodos realizan la acción CRUD correcta, pero presentan errores parciales (por ejemplo, no validan el id, no retornan el	Implementa correctamente entre 10 y 11 métodos de los 12, utilizando el método HTTP correspondiente, recibiendo datos en JSON cuando corresponde y aplicando la	Implementa correctamente los 12 métodos CRUD, uno por cada combinación esperada (GET/POST/PUT/PATCH/DELETE para cada recurso según la pauta). Cada método utiliza el verbo HTTP correcto,

modificar datos en la aplicación.	incorrectos). No se diferencia claramente entre GET/POST/PUT/PATCH/DELETE en PHP.	código HTTP adecuado, o la actualización/eliminación no se persiste correctamente).	operación CRUD adecuada. Los datos se modifican y se devuelven respuestas coherentes en la mayoría de los casos.	recibe/retorna JSON válido, persiste cambios de forma consistente y maneja correctamente los casos de éxito y fallo. La lógica es clara, sin duplicación innecesaria.
	0 puntos	**21 puntos**	**28 puntos**	**35 puntos**
3.1.3 Adapta el framework CRUD a diferentes tipos de datos y estructuras dentro de la aplicación.	Las respuestas JSON se alejan significativamente de los ejemplos entregados. En 3 o menos de los 8 casos esperados (combinación de recursos y operaciones) se respeta la estructura (campos id, nombre, empresa_tipo, activo, y objetos icono y color). Se pierden datos anidados o se devuelven formatos inconsistentes.	Respeta parcialmente la estructura JSON entregada en 4 a 5 de los 8 casos. En algunos endpoints se devuelven correctamente los objetos anidados (empresa_tipo.icono, empresa_tipo.color), pero en otros se simplifica a solo el id o se omiten campos. La adaptación a estructuras anidadas es incompleta.	Respeta la estructura JSON en 6 o 7 de los 8 casos. En general, las respuestas incluyen todos los campos esperados, manteniendo los objetos anidados (icono, color) y el campo activo. Solo se observan detalles menores (por ejemplo, orden de campos o algún nombre ligeramente diferente).	Respeta y adapta exactamente la estructura JSON en los 8 casos; todas las respuestas reproducen la forma entregada en el enunciado, incluyendo objetos anidados (empresa_tipo, icono, color), campos codigo, activo, y mensajes coherentes. Además, la estructura se mantiene consistente en todas las operaciones (GET, POST, PUT, etc.), lo que demuestra una adaptación sólida del CRUD a estructuras complejas.
	0 puntos	**15 puntos**	**20 puntos**	**25 puntos**
3.1.4 Depura errores en la implementación de las operaciones CRUD, para asegurar el correcto funcionamiento de la aplicación.	No realiza validaciones significativas sobre los datos de entrada. No distingue entre entradas válidas e inválidas, ni maneja ids inexistentes. Los errores de ejecución (excepciones, índices inexistentes) se exponen al usuario o detienen el script. No hay controles de seguridad evidentes (por ejemplo, filtrado de entrada, manejo de errores, mensajes genéricos).	Implementa 1 o 2 controles básicos: por ejemplo, valida la presencia de id en algunos métodos o revisa si el JSON está bien formado. Maneja parcialmente algunos errores devolviendo mensajes simples, pero otros errores quedan sin control. La seguridad y depuración es inconsistente entre endpoints.	Implementa al menos 3 controles consistentes: validación de id existente antes de actualizar/eliminar, chequeo de campos obligatorios en POST/PUT, manejo de JSON mal formado, y respuestas controladas ante errores (con códigos HTTP apropiados). Evita exponer mensajes internos de error.	Implementa un conjunto robusto de medidas: validaciones exhaustivas de entrada, verificación de estados (activo/inactivo) antes de cambios, manejo de excepciones centralizado, mensajes de error claros pero no reveladores, y uso coherente de códigos HTTP (400, 404, 409, 500, etc.). Se observan 4 o más mecanismos concretos de seguridad/depuración aplicados en la mayoría de los métodos.
	0 puntos	**6 puntos**	**8 puntos**	**10 puntos**

3.1.5 Optimiza el uso del framework CRUD para mejorar el rendimiento y la eficiencia de la aplicación.	No se observa optimización ni reutilización de código (cada método repite lógica). No se proporciona documentación Swagger o solo hay un esbozo sin endpoints funcionales. Se documentan **0 a 3 endpoints** y sin detalles de parámetros ni respuestas.	Reutiliza parcialmente funciones auxiliares (por ejemplo, lectura/escritura de datos), pero aún hay duplicación innecesaria. Implementa una especificación Swagger básica, documentando correctamente entre 4 y 7 endpoints, con rutas y métodos correctos pero con detalles limitados (sin todos los parámetros o ejemplos de respuesta).	Organiza el código CRUD de forma más eficiente: utiliza funciones comunes para validar, leer y responder, reduciendo la duplicación. La documentación Swagger describe correctamente entre 8 y 10 endpoints, incluyendo: ruta, método, parámetros principales y ejemplos básicos de respuesta. Permite probar la mayoría de las operaciones desde Swagger UI.	Estructura el CRUD de forma altamente optimizada, con funciones reutilizables, mínima duplicación de lógica y clara separación de responsabilidades (validación, acceso a datos, respuesta). La documentación Swagger es completa: describe 11 o 12 endpoints, incluyendo: esquemas de datos, parámetros, códigos HTTP, ejemplos de request/response y notas de uso. La API puede ser comprendida y usada solo a partir de la documentación.
	0 puntos	**12 puntos**	**16 puntos**	**20 puntos**
TOTALES	**0 PUNTOS**	**60 PUNTOS**	**80 PUNTOS**	**100 PUNTOS**

Escala de notas según puntaje obtenido

Puntaje	Nota
0.0	1.0
1.0	1.1
2.0	1.1
3.0	1.2
4.0	1.2
5.0	1.3
6.0	1.3
7.0	1.4
8.0	1.4
9.0	1.5

Puntaje	Nota
10.0	1.5
11.0	1.6
12.0	1.6
13.0	1.7
14.0	1.7
15.0	1.8
16.0	1.8
17.0	1.9
18.0	1.9
19.0	2.0

Puntaje	Nota
20.0	2.0
21.0	2.1
22.0	2.1
23.0	2.2
24.0	2.2
25.0	2.3
26.0	2.3
27.0	2.4
28.0	2.4
29.0	2.5

Puntaje	Nota
30.0	2.5
31.0	2.6
32.0	2.6
33.0	2.7
34.0	2.7
35.0	2.8
36.0	2.8
37.0	2.9
38.0	2.9
39.0	3.0

Puntaje	Nota
40.0	3.0
41.0	3.1
42.0	3.1
43.0	3.2
44.0	3.2
45.0	3.3
46.0	3.3
47.0	3.4
48.0	3.4
49.0	3.5

Puntaje	Nota
50.0	3.5
51.0	3.6
52.0	3.6
53.0	3.7
54.0	3.7
55.0	3.8
56.0	3.8
57.0	3.9
58.0	3.9
59.0	4.0

Puntaje	Nota
60.0	4.0
61.0	4.1
62.0	4.2
63.0	4.2
64.0	4.3
65.0	4.4
66.0	4.5
67.0	4.5
68.0	4.6
69.0	4.7

Puntaje	Nota
70.0	4.8
71.0	4.8
72.0	4.9
73.0	5.0
74.0	5.1
75.0	5.1
76.0	5.2
77.0	5.3
78.0	5.4
79.0	5.4

Puntaje	Nota
80.0	5.5
81.0	5.6
82.0	5.7
83.0	5.7
84.0	5.8
85.0	5.9
86.0	6.0
87.0	6.0
88.0	6.1
89.0	6.2

Puntaje	Nota
90.0	6.3
91.0	6.3
92.0	6.4
93.0	6.5
94.0	6.6
95.0	6.6
96.0	6.7
97.0	6.8
98.0	6.9
99.0	6.9

Puntaje	Nota
100.0	7.0