← Volver

¡Superado!

Calificación global

7.8/10

Feedback por criterio

Criterio 1: Implementación correcta del modelo Customer

Calificación porcentual: 18 / 20%

Calificación sobre 10: 9

Tipo	Ruta	Mensaje
Error	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/Customer.java	El campo 'edad' no es necesario según los requisitos del modelo 'Customer', que debería tener solo 'id', 'nombre', 'apellido' y 'email'.
Feedback	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/Customer.java	El método 'toString' está sobrescrito adecuadamente proporcionando una representación legible del objeto 'Customer'.

La clase Customer está correctamente implementada con constructor, getters, setters y método toString. Sin embargo, incluye un campo adicional 'edad' que no es parte de los requisitos especificados para el modelo Customer.

Criterio 2: Funcionalidad del repositorio CustomerRepository

Calificación porcentual: 28 / 40%

Calificación sobre 10: 7

Tipo	Ruta	Mensaje
Error	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/CustomersMethods.java	El archivo CustomersMethods actúa como el repositorio, pero no implementa las operaciones CRUD siguiendo exactamente las firmas dadas en el enunciado ('save', 'findById', 'update', 'delete', 'findAll').
Sugerencia	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/CustomersMethods.java	Considera refactorizar la clase CustomersMethods para implementar exactamente las firmas de métodos CRUD especificadas ('save', 'findById', 'update', 'delete', 'findAll') para mejorar la claridad y alineación con el estándar.
Feedback	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/CustomersMethods.java	La estructura de almacenamiento y manipulación de datos en el 'ArrayList<Customer>` es eficiente.

La clase CustomersMethods maneja adecuadamente una lista de objetos Customers y permite manipulación básica de los datos. Sin embargo, su implementación no sigue las firmas de métodos CRUD exactamente como se especifica, lo cual es crucial para el cumplimiento estricto de los requisitos.

Criterio 3: Interacción con el usuario en la clase Main

Calificación porcentual: 32 / 40%

Calificación sobre 10: 8

Tipo	Ruta	Mensaje
Error	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/Principal.java	El método de actualización de cliente no contempla bien la retirada de un problema con 'scanner.nextLine()' que es necesario justo después de usos de 'scanner.nextInt()' para evitar errores de entrada.
Error	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/Principal.java	La forma de manejo de excepciones no está explícitamente implementada para capturar entradas inválidas de forma robusta, por ejemplo, inputs no numéricos al solicitar un entero.
Sugerencia	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/Principal.java	Implementa 'try-catch' alrededor de las operaciones que leen entradas del usuario para manejar mejor las excepciones al ingresar datos no válidos.
Feedback	/workspace/Practica_M1_Java_KZ-main/src/main/java/com/kevinzamora/Principal.java	La interfaz de usuario en consola es clara y las instrucciones de uso están bien especificadas.

La clase Main ofrece una interacción de usuario clara y bien estructurada para realizar operaciones CRUD. No obstante, el código carece de robustez en el tratamiento de entradas inválidas, lo cual es crítico para una buena experiencia de usuario.

Resumen global

El proyecto tiene una base sólida en términos de estructura y funcionalidad, cumpliendo muchas de las expectativas básicas para un CRUD en Java. Sin embargo, hay áreas clave donde no se siguen estrictamente los requisitos específicos, como en la firma de métodos CRUD y gestión de excepciones, dejando espacio para mejoras esenciales. La sobreespecificación en clase Customer y la falta de robustez en el manejo de entradas de usuario son puntos importantes a abordar.

Cerrar proyecto