Inspection Issue Log

Project:	Project-s3952320 PR258	**Origin:**	Requirements, Design, Implementation, Testing
Inspection ID:	BMS-CR-001	**Type:**	Missing, Wrong, Extra, Usability, Performance, Style, Clarity, Question
Meeting Date:	19/09/2024 - 23/09/2024		
Recorder:	Oisin Aeonn (s3952320)	**Severity:**	Major, minor

Defects Found			
Major	36	Minor	31
Defects Corrected			
Major	0	Minor	0

#	Origin	Type	Severity	Date Opened	Location	Java Checklist	Description
1	Implementation	Wrong	Major	19/09/2024	Entire Bank class	MO	The Bank class exhibits low cohesion by handling UI, data management, and business logic, violating SRP. This results in high coupling between core functionalities. Ideally, this should be split into separate modules (e.g., UI, Accounts and Transactions) to improve modularity and maintainability.
2	Implementation	Wrong	Major	20/09/2024	Entire file	MO	Multiple classes (Account, Bank, Task_Driver) are defined in a single file, violating SRP. Separate Account, Bank, and Task_Driver into individual files to improve code organization and maintainability.
3	Implementation	Missing	Major	20/09/2024	Throughout code	CM	The code lacks comprehensive comments throughout. Add clear comments for methods, code segments, loops, variable declarations, and exception handling to improve code documentation and maintainability.
4	Implementation	Missing	Major	22/09/2024	Entire codebase	IO	The codebase lacks unit tests. Create comprehensive unit tests for all methods in Bank and Account classes to ensure code reliability and ease future modifications. (This is what we'll do next).
5	Implementation	Wrong	Minor	20/09/2024	Throughout code	LP	The code has inconsistent indentation and formatting throughout, reducing readability. Standardize code formatting to improve readability and maintainability.
6	Implementation	Wrong	Minor	21/09/2024	Throughout code	VC	The code uses a mix of naming conventions (snake_case and camelCase). Standardize on camelCase for method and variable names throughout the code for consistency and adherence to Java conventions.
7	Implementation	Wrong	Minor	22/09/2024	Lines 3	VC	The class name 'Account' is somewhat generic and non-descriptive. Consider renaming to a more descriptive name like 'BankAccount' to clearly indicate its purpose and improve code clarity.
8	Implementation	Wrong	Major	21/09/2024	Lines 7	VC	The Account class is not declared as final, which potentially allows unintended subclassing. If Account is not meant to be extended, declare it like this: final class Account implements Serializable to prevent inheritance.
9	Implementation	Wrong	Minor	20/09/2024	Lines 9	FD	The class member 'name' in the Account class is declared with default (package-private) access, violating encapsulation principles.
10	Implementation	Wrong	Major	20/09/2024	Lines 10	FD	The class member 'account_number' in the Account class is declared with default (package-private) access, violating encapsulation principles.
11	Implementation	Wrong	Minor	21/09/2024	Lines 10	VC	The variable name 'account_number' uses snake_case, which is unconventional in Java variable naming. Rename to 'accountNumber' to follow Java naming conventions of camelCase to improve code readability.

12	Implementation	Wrong	Major	20/09/2024	Lines 11	FD	The class member 'pin' in the Account class is declared with default (package-private) access, violating encapsulation principles.
13	Implementation	Wrong	Major	19/09/2024	Lines 12	CN	The code uses double for currency values, which can lead to precision errors in financial calculations. Instead, use long to represent currency in cents, and perform all calculations in this integer format. This approach avoids floating-point precision issues and is widely used in industry for handling currency.
14	Implementation	Wrong	Major	20/09/2024	Lines 12	FD	The class member 'Amount' in the Account class is declared with default (package-private) access, violating encapsulation principles.
15	Implementation	Wrong	Minor	23/09/2024	Lines 19	VC	The variable 'Amount' in the Account class uses PascalCase, which is unconventional for Java variable naming. Rename to 'amount' to follow Java naming conventions of camelCase for improved code readability.
16	Implementation	Wrong	Minor	21/09/2024	Lines 27	VC	The Account constructor initializes Amount with a hard-coded value (1000). We should replace Amount = 1000 + amount; with a named constant private static final long INITIAL_BALANCE = 100000; (which represents $1000.00 in cents). This improves code readability and makes future modifications to the initial balance easier and less error prone.
17	Implementation	Performance	Major	19/09/2024	Lines 74	PE	ArrayList is used for account storage, which provides Θ(n) time complexity for searches. For better performance, especially with a large number of accounts, a HashMap<Integer, Account> should be used, providing Θ(1) access time.
18	Implementation	Wrong	Minor	19/09/2024	Lines 74	VC	The variable 'AL' in the Bank class uses an unclear name. Rename to a more descriptive name like 'accountList' to improve code readability and maintainability.
19	Implementation	Wrong	Major	19/09/2024	Lines 76	FD	The addNewRecord() method is declared as public, which exposes its internal operations. The method should be private to encapsulate the account creation process within the Bank class, adhering to the principle of least privilege.
20	Implementation	Missing	Major	21/09/2024	Lines 76	CF	The addNewRecord() method lacks input validation for the initial deposit amount. We should implement a check to ensure the initial deposit is non-negative. If a negative value is provided, the method should throw an IllegalArgumentException with an appropriate error message.
21	Implementation	Wrong	Minor	23/09/2024	Lines 30 & 50	VC	Variable 'n' in the addNewRecord() method is not descriptive. Rename to 'name' for clarity.
22	Implementation	Missing	Major	20/09/2024	Lines 81	IO	The addNewRecord() method lacks a mechanism to verify the uniqueness of account numbers. This omission could lead to duplicate account numbers, compromising data integrity and potentially causing conflicts in the account management.
23	Implementation	Wrong	Minor	23/09/2024	Lines 35 & 55	VC	Variable 'a' in the addNewRecord() method is not descriptive. Rename to 'accountNumber' for clarity.
24	Implementation	Wrong	Major	21/09/2024	Lines 94	FD	The transfer() method is declared as public, which potentially allows external classes to bypass account validation. We should change to private void transfer() to ensure the principle of least privilege is followed.
25	Implementation	Missing	Major	21/09/2024	Lines 99	IO	The transfer() method doesn't validate the format of account numbers to ensure that account numbers are 8 digits long to maintain data integrity and prevent invalid transfers.
26	Implementation	Wrong	Minor	23/09/2024	Lines 40 & 60	VC	Variable 'p' in the addNewRecord() method is not descriptive. Rename to 'pin' for clarity.
27	Implementation	Wrong	Minor	23/09/2024	Lines 45 & 65	VC	Variable 'am' in the addNewRecord() method is not descriptive. Rename to 'initialDeposit' for clarity.
28	Implementation	Wrong	Major	21/09/2024	Lines 146	FD	Similarly, the withdraw() method is public, which could allow unauthorized access to account funds. Make it private to ensure withdrawals are only performed through the Bank class's public interface.
29	Implementation	Missing	Major	21/09/2024	Lines 152	IO	The withdraw() method lacks validation for the user's PIN. Implement a check to verify the PIN format / length before allowing the withdrawal to proceed.
30	Implementation	Missing	Major	21/09/2024	Lines 158	IO	The withdraw() method doesn't validate if the account number is 8 digits. Implement a check to ensure the entered account number is valid before proceeding with the withdrawal.
31	Implementation	Missing	Major	19/09/2024	Lines 76 to 92	IO	The addNewRecord() method lacks input validation or exceptions for the 8-digit account number. It should check if the length of the a string is 8 and thrown an IllegalArgumentException if not.

32	Implementation	Wrong	Major	21/09/2024	Lines 184	FD	Again, the print() method is public, potentially exposing sensitive account information. Change to private void print() and create a public method that returns the account list if needed.
33	Implementation	Wrong	Major	19/09/2024	Lines 194	FD	The load() method in the Bank class is declared as public, potentially allowing unauthorized data loading.
34	Implementation	Wrong	Major	19/09/2024	Lines 213	FD	The save() method in the Bank class is declared as public, potentially allowing unauthorized data saving.
35	Implementation	Wrong	Minor	20/09/2024	Lines 216	MO	The filename for bank records is hardcoded in the load() method. Use a constant for the filename to improve code maintainability and centralize file path management.
36	Implementation	Wrong	Minor	23/09/2024	Lines 230	VC	The class name 'Task_Driver' uses snake_case, which is unconventional for Java class naming. Rename to 'TaskDriver' to follow Java naming conventions of PascalCase for class names.
37	Implementation	Wrong	Minor	20/09/2024	Lines 237	IO	The main method uses a try block but doesn't properly manage the Scanner resource.
38	Implementation	Missing	Major	19/09/2024	Lines 94 to 144	DR	The transfer() method lacks null checks when accessing ArrayList elements. This could lead to NullPointerExceptions if an account reference is null.
39	Implementation	Missing	Major	21/09/2024	Lines 94 to 144	IO	There's no input validation for the sender's PIN in the transfer() method. Add a check to ensure the PIN meets required format/length before proceeding with the transfer.
40	Implementation	Wrong	Minor	20/09/2024	Lines 94 to 144	MO	The transfer() method contains account search logic that violates the DRY principle. Extract this functionality into a separate, reusable method.
41	Implementation	Missing	Minor	20/09/2024	Lines 94 to 144	IO	The transfer() method lacks user feedback for failed transactions. There should be a printed message using System.out of "Transfer failed: Insufficient funds" when a transfer fails, this will provide clear feedback to the user about the reason for the failure.
42	Implementation	Wrong	Minor	23/09/2024	Lines 94 to 144	VC	Variable 's_acc' in the transfer() method uses unconventional snake_case naming. Rename to 'senderAccount' for clarity and to follow Java conventions.
43	Implementation	Wrong	Minor	23/09/2024	Lines 94 to 144	VC	Variable 's_pin' in the transfer() method uses unconventional snake_case naming. Rename to 'senderPin' for clarity and to follow Java conventions.
44	Implementation	Wrong	Minor	23/09/2024	Lines 94 to 144	VC	Variable 'r_acc' in the transfer() method uses unconventional snake_case naming. Rename to 'receiverAccount' for clarity and to follow Java conventions.
45	Implementation	Wrong	Minor	20/09/2024	Lines 268	MO	The filename for bank records is hardcoded in the save() method. Use a constant for the filename to improve code maintainability and centralize file path management.
46	Implementation	Missing	Major	21/09/2024	Lines 135 to 136	IO	The transfer() method lacks validation for the transfer amount. We should implement a check to ensure the transfer amount is positive. If a zero or negative value is provided, the method should throw an IllegalArgumentException with an appropriate error message.
47	Implementation	Wrong	Major	23/09/2024	Lines 94 to 182	CF	The transfer() method has a deeply nested if-else statements, making the code hard to follow.
48	Implementation	Missing	Major	20/09/2024	Lines 40-43, 60, 100, & 152	FD	PINs are stored in plain text, posing a security risk. Implement secure hashing with salt (e.g., using BCrypt or another cryptographic library) and update PIN comparison logic to enhance security.
49	Implementation	Missing	Major	19/09/2024	Lines 146 to 182	DR	The withdraw() method lacks null checks when accessing ArrayList elements. This could lead to NullPointerExceptions if an account reference is null.
50	Implementation	Missing	Major	21/09/2024	Lines 146 to 182	IO	The withdraw() method lacks validation for the withdraw amount. We should implement a check to ensure the withdrawal amount is positive. If a zero or negative value is provided, the method should throw an IllegalArgumentException with an appropriate error message.
51	Implementation	Wrong	Major	23/09/2024	Lines 146 to 182	CF	The withdraw() method has deeply nested if-else statements, making the code hard to follow.
52	Implementation	Wrong	Minor	20/09/2024	Lines 146 to 182	MO	The withdraw() method contains account search logic that violates the DRY principle. Extract this functionality into a separate, reusable method.
53	Implementation	Wrong	Minor	23/09/2024	Lines 146 to 182	VC	Variable 'p_acc' in the withdraw() method uses unconventional snake_case naming. Rename to 'accountNumber' for clarity and to follow Java conventions.
54	Implementation	Wrong	Minor	23/09/2024	Lines 146 to 182	VC	Variable 'p_pin' in the withdraw() method uses unconventional snake_case naming. Rename to 'pin' for clarity and to follow Java conventions.

55	Implementation	Wrong	Major	23/09/2024	Lines 199 to 205	CF	In the load() method, the while(true) loop doesn't properly handle the end of file, potentially causing an infinite loop if an EOFException is caught and ignored.
56	Implementation	Wrong	Minor	20/09/2024	Lines 194 to 211	IO	The load() method doesn't use modern resource management techniques for file operations, potentially leading to resource leaks. Implementing proper resource handling to ensure files are correctly closed after use is key to reducing this.
57	Implementation	Missing	Minor	20/09/2024	Lines 194 to 211	IO	The load() method doesn't handle IOException separately from other exceptions. Implement specific exception handling for IOExceptions to provide more informative error messages when file operations fail.
58	Implementation	Missing	Minor	20/09/2024	Lines 194 to 211	IO	The load() method does not handle FileNotFoundException separately. Implement specific handling for this exception to provide a clear message when the bank record file is missing, improving error reporting and user experience.
59	Implementation	Missing	Major	20/09/2024	Lines 194 to 227	PE	The program reads and writes the entire account list for each operation, which is inefficient. Would benefit from database or another larger scale storage method.
60	Implementation	Missing	Minor	22/09/2024	Lines 211 to 217	PE	The print() method doesn't implement pagination to store data in chunks which is inefficient.
61	Implementation	Missing	Major	21/09/2024	Lines 213 to 227	IO	The save() method doesn't validate data integrity before writing to the BankRecord.txt file. Add checks to ensure all account data is valid before saving to prevent corruption of the data file.
62	Implementation	Missing	Minor	20/09/2024	Lines 213 to 227	IO	The save() method lacks exception handling for potential I/O errors. Implement error handling to catch IOExceptions, provide informative error messages to the user, and improve overall error reporting.
63	Implementation	Missing	Minor	20/09/2024	Lines 237 to 274	IO	The Scanner object in the main method is not closed after use. Implement proper resource management to prevent potential resource leaks and follow I/O best practices.
64	Implementation	Missing	Major	20/09/2024	Lines 232 to 285	IO	The main method lacks input validation for user inputs. Implement checks to ensure user input is of the expected type before processing, to prevent InputMismatchException and improve user experience.
65	Implementation	Wrong	Major	19/09/2024	Lines 275 to 278	CF	The main method uses overly broad exception handling (catch(Exception e)), which can hide specific issues. Ideally, we should catch and handle specific exceptions separately for more appropriate error handling and recovery actions.
66	Implementation	Missing	Minor	20/09/2024	Lines 275 to 278	IO	User input parsing lacks exception handling for invalid number formats. Implement error handling for NumberFormatException to gracefully manage invalid inputs and provide clear feedback to the user.
67	Implementation	Wrong	Major	22/09/2024	Lines 262 to 292	MO	The main method mixes UI and business logic. Separate these concerns by creating a dedicated UI class that interacts with the Bank class, improving code organization and maintainability.

Java Inspection Checklist Key	
VC	Variable, Attribute, and Constant Declaration Defects
FD	Method Definition Defects
CD	Class Definition Defects
DR	Data Reference Defects
CN	Computation/Numeric Defects
CR	Comparison/Relational Defects
CF	Control Flow Defects
IO	Input-Output Defects
MI	Module Interface Defects
CM	Comment Defects
LP	Layout and Packaging Defects
MO	Modularity Defects
SU	Storage Usage Defects
PE	Performance Defects