System Requirements Specification Index

For

Personal Finance Management

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

TABLE OF CONTENTS

1 Project Abstract

Personal Finance Management Console is a Java-based console application designed to help users efficiently manage their personal finances. The system allows users to track expenses, categorize transactions, and maintain a record of financial activities. The application enables users to perform CRUD (Create, Read, Update, Delete) operations on expense records, view transactions by category, generate financial reports, and set their monthly income. Users can update, delete, and analyze their spending habits, ensuring effective financial management. The system provides an interactive menu-driven interface, making it simple and user-friendly for managing personal expenses and budgeting effectively.

2 Business Requirements:

Screen Name	Console input screen
Problem Statement	1. User needs to enter into the application. 2. The user should be able to do the particular operations 3. The console should display the menu 1) Add Expense 2) View All Expenses 3) View Expenses by Category 4) Update Expense 5) Delete Expense 6) View Balance 7) Generate Monthly Report 8) Generate Expense by Category Report 9) Set Monthly Income 10) Exit

3 Constraints

3.1 Transaction Constraints

- When adding an expense with a negative or zero amount, the method should throw an **InvalidAmountException** with the message:

  "Expense amount must be positive."

- When trying to update a transaction:

  ➔ If the **index is less than 0** or **greater than or equal to the size of the transactions list**, then the method should throw an **IndexOutOfBoundsException** with the message:

  "Transaction not found at index: [index]."

  ➔ If the **amount is negative or zero**, the method should throw an **InvalidAmountException** with the message:

  "Amount must be positive."

  ➔ If the **category is invalid or does not exist**, the method should throw an **InvalidCategoryException** with the message:

  "Invalid category: [category name]."

- When trying to remove a transaction with an invalid index (i.e., the index is **less than 0** or **greater than or equal to the size of the transactions list**), the method should throw an **IndexOutOfBoundsException** with the message:

  "Transaction not found at index: [index]."

3.2 Common Constraints

- The system should allow users to input and store details of multiple transactions.
- Users should be able to categorize each transaction during input.
- Users should be able to retrieve and filter expenses based on categories.
- Users should be able to update or delete a transaction by specifying its index (and index value must start with 0).
- The system should calculate and display the remaining balance after expenses have been recorded.
- Users should be able to generate a monthly expense report by specifying a month and year.
- Users should be able to generate an expense report categorized by expense type.

4 Template Code Structure

4.1 Package: com.finance

Resources

Class/Interface	Description	Status
PersonalFinanceApp.java(class)	This represents bootstrap class i.e class with Main method, that shall contain all console interaction with the user.	**Partially implemented**

4.2 Package: com.finance.model

Resources

Class/Interface	Description	Status
Transaction (class)	● This class contains all the properties of the Transaction class.	Already implemented.

4.3 Package: com.finance.inventory

Resources

Class/Interface	Description	Status
TransactionManager (class)	● This class contains all the methods which are used to write the business logic for the application ● You can create any number of private methods in the class	Partially implemented.

4.4 Package: com.finance.exception

Resources

Class/Interface	Description	Status
BudgetExceededException (Class)	● Custom Exception to be thrown when an expense exceeds the allocated budget.	Already created.
InvalidAmountException (Class)	● Custom Exception to be thrown when an invalid amount (negative or zero) is entered for a transaction.	Already created.
InvalidCategoryException (Class)	● Custom Exception to be thrown when an invalid or non-existent category is used for a transaction.	Already created.

5 Execution Steps to Follow

1. **All actions like build, compile, running application, running test cases will be through Command Terminal.**
2. **To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal.**
3. **This editor Auto Saves the code.**
4. **If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.**
5. **These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.**
6. **To run your project use command:**

   ```
   mvn clean install exec:java -Dexec.mainClass="com.finance.PersonalFinanceApp"
   ```
7. **To test your project, use the command**

   ```
   mvn test
   ```

8. **You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.**