**Money Module**

Linda’s bank account looks good but at a glance, she can’t easily see which transactions are related to each of her locations.

She finds that at the end of the month manually, she’s manually calculating profits. She is currently using a simple application built by her grandchildren, but find that she’s missing cash sales. Linda would like to record a description of the transaction indicating whether or not the transaction was income or expense and more details of what the transaction was for. She usually pay for supplies using her credit card. Her customers pay by cash, credit card or debit card.

Linda needs a system that helps her with her HST calculations. She must be able to query income – expenses.

They would like a more robust system than Excel to manage these details.

Your task is to read the User Story below and create a class diagram and sequence diagrams to support the scenario.

Copy your diagrams to a word file. Analyze your model and outline what the .h files would look like if you generated code from your model. Include pseudo code to indicate what each method would do. Attach this word file and your visual paradigm file to your submission.

**User Story**

As the owner of this business, I would like to record my financial transactions so that I can quickly see how much money I’ve made and what I owe the government in HST. For my purposes, my expenses subtracted from my income.

Acceptance Criteria:

1. Must be able to query transactions by date and provide a total profit and total HST
2. Must be able to query transactions by date and by source of the income.
3. Differentiate between expenses and income – provide a running total of income – expenses
4. Calculate the HST owed to the government by subtracting HST paid on expenses from HST collected on income.

Class Diagram

A diagram of a computer

Description automatically generated with medium confidence

**Use Case Descriptions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name** | **Create an income transaction** | | |
| Triggering Event | Money coming into the business. | | |
| Brief Description | Allows the Owner to record an income transaction. | | |
| Actors | Owner | | |
| Related Use Cases |  | | |
| Preconditions | Owner has opened the Main Menu. | | |
| Post Conditions | Income transaction is saved and added to financial totals. | | |
| Flow of activities | Actor | | System |
|  |  | Requests to add a new income transaction | Displays a calendar  Prompts for description, transaction amount and HST amount  Displays a list of payment types and prompts for selection |
|  |  | Selects the transaction Date  Selects the payment  Enters the transaction description  Enters the transaction amount | Verifies that date is selected  Verifies that description and amount have been entered  Verifies that the payment type was selected.  Calculates HST  Displays income transaction  Prompts to save |
|  |  | Request to save | Saves the transaction and returns to the main menu |
| Exception Conditions | * Owner chooses to cancel adding the transaction | | |

A diagram of a computer program

Description automatically generated with medium confidence

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name** | **Create an expense transaction** | | |
| Triggering Event | Money leaving the business. | | |
| Brief Description | Allows the Owner to record an expense transaction. | | |
| Actors | Owner | | |
| Related Use Cases |  | | |
| Preconditions | Owner has opened the Main Menu. | | |
| Post Conditions | Expense transaction is saved and added to financial totals. | | |
| Flow of activities | Actor | | System |
|  |  | Requests to add a new expense transaction | Displays a calendar  Prompts for description, transaction amount and HST amount |
|  |  | Selects the transaction Date  Enters the transaction description  Enters the transaction amount | Verifies that date is selected  Verifies that description and amount have been entered  Calculates HST  Displays financial transaction  Prompts to save |
|  |  | Request to save | Saves the transaction and returns to the main menu |
| Exception Conditions | * Owner chooses to cancel adding the transaction | | |

A diagram of a project

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **Use Case Name** | **Query Financial Transaction** | | |
| Triggering Event | Owner requires a list of transactions for a date period including Total Income, Total Expenses, Total Profit, Total HST collected, Total HST Paid, HST Owed | | |
| Brief Description | Allows the Owner to retrieve financial totals for a specified date range | | |
| Actors | Owner | | |
| Related Use Cases |  | | |
| Preconditions | Owner has opened the Main Menu. | | |
| Post Conditions | Financial transactions are retrieved, totaled and displayed | | |
| Flow of activities | Actor | | System |
|  | 1. | Requests to query financial transactions | Displays a calendar |
|  | 2. | Selects date range | Verifies that dates are selected  Retrieves transactions in the specified date range  Calculates and displays total income, total expenses, Profit, total HST Owed. Prompts to exit |
|  | 3. | Request to exit | returns to the main menu |
| Exception Conditions |  | | |

A diagram of a project

Description automatically generated

Your tasks:

1. Create a class diagram to support the above case study and Systems Use Case Specifications
2. Create an object level sequence diagram, detailing the Create Income Transaction systems use case specification
3. Create an object level sequence diagram, detailing the Create Income Transaction systems use case specification
4. Create an object level sequence diagram, detailing the Query Financial Transaction system use case specification.
5. Analyze your model and outline what the .h files would look like if you generated code from your model

**.h File**

**IncomeTransaction.h**

Public class IncomeTransaction {

Int incTransID;

String incTransDescription;

Double inctransAmount;

Int incTransdate;

PaymentType paymentType;

create(incTransID,incTransDate,incTransDesc,inctransAmount, paymentTypeID);

getincomeTransaction();

}

**ExpenseTransaction.h**

Public class ExpenseTransaction{

Int expTransID;

String expTransDesc;

Double expTransAmount;

Int expTransdate;

String expTransReceipt;

create(expTransdate, expTransdescription, expTransAmount);

getexpenseTransaction();

}

**PaymentType.h**

Public class paymentType{

Int paymentTypeID;

String paymentTypeDesc;

create();

getpaymentType();

}

**UIController.h**

Public class getPaymentTypes(){}

Public class enterIncomeTransaction(incTransID,incTransDate,incTransDesc,inctransAmount, paymentTypeID){}

Public class saveIncomeTransaction(){}

Public class createExpenseTransaction(){}

Public class enterExpenseTransaction(expTransdate, expTransdescription, expTransAmount) {}

Public class saveExpenseTransaction(){}

Public class getFinancialTransaction(){}

Public class getFinancialTransaction(startDate,endDate) {}

Public class exit(){}

**DomainController.h**

Public class getPaymentTypes() {}

Public class createIncomeTransaction(incTransID,incTransDate,incTransDesc,inctransAmount, paymentTypeID) {}

Public class saveIncomeTransaction(){}

Public class createExpenseTransaction(expTransdate, expTransdescription, expTransAmount) {}

Public class saveExpenseTransaction(){}

Public class getFinancialTransaction(startDate,endDate) {}

Public class calculateHST(){}

Public class calculateTotalIncome(){}

Public class calculateHSTpaid(){}

Public class calculateTotalExpense(){}

Public class calculateTotalHSTOwed(){}

Public class calculateProfit(){}

**EntityManager.h**

Public class get(PaymentType) {}

Public class persist(incomeTransaction) {}

Public class persist(expenseTransaction) {}

Public class get(incomeTransaction,startDate<=date<=endDate) {}

Public class get(expenseTransaction,startDate<=date<=endDate) {}