EABANK

CCOBJPRG - INF246

EABank System: A Secure E-Wallet Platform for Digital Transactions

Kate Agsiwang

Andrea Aquino

Fiona Recaña

EABank System

Our project is a simulation of a mobile-based e-wallet system, inspired by popular digital wallets like GCash and PayMaya. It is designed to demonstrate how users can perform common financial transactions such as cash in, cash out, transfer, balance inquiry, and currency exchange, all within a secure and user-friendly interface. The goal is to replicate real-life e-wallet features using Java Swing, while also applying object-oriented principles and design patterns in a practical way.

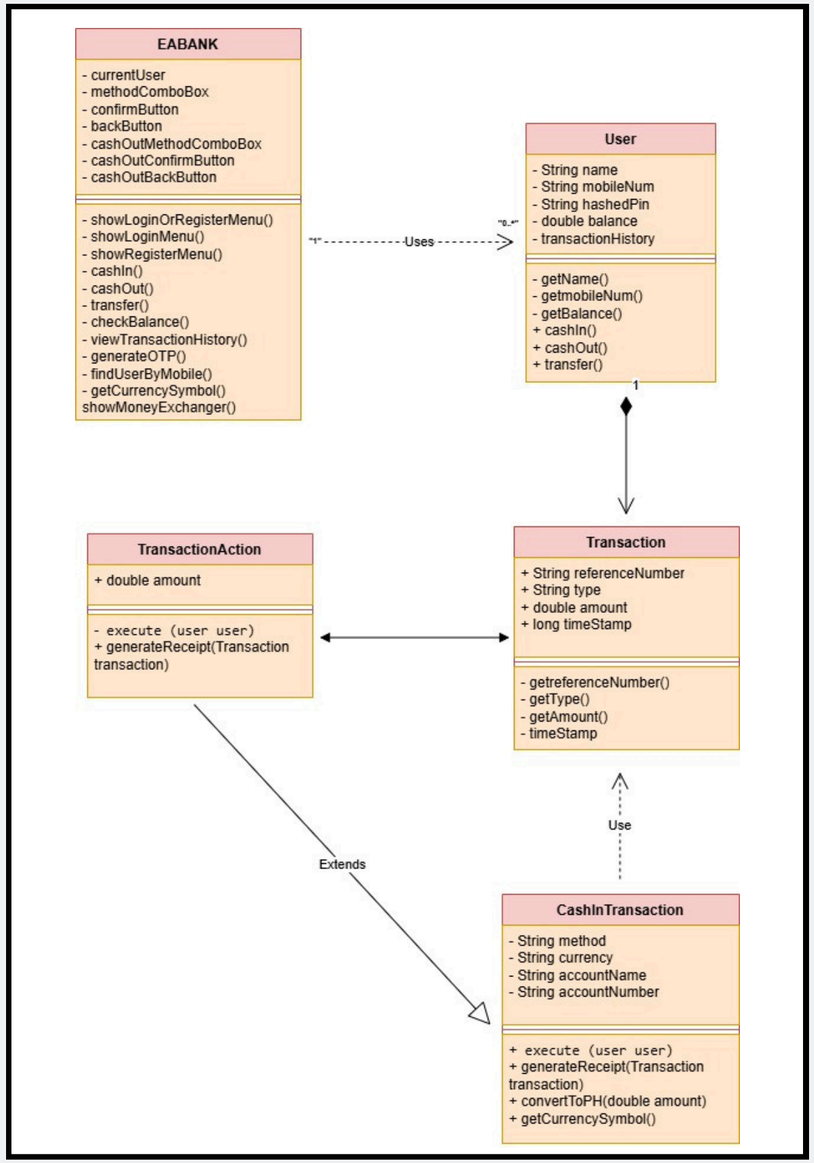
EABANK INTRODUCTION UML & USE CASE DIAGRAM

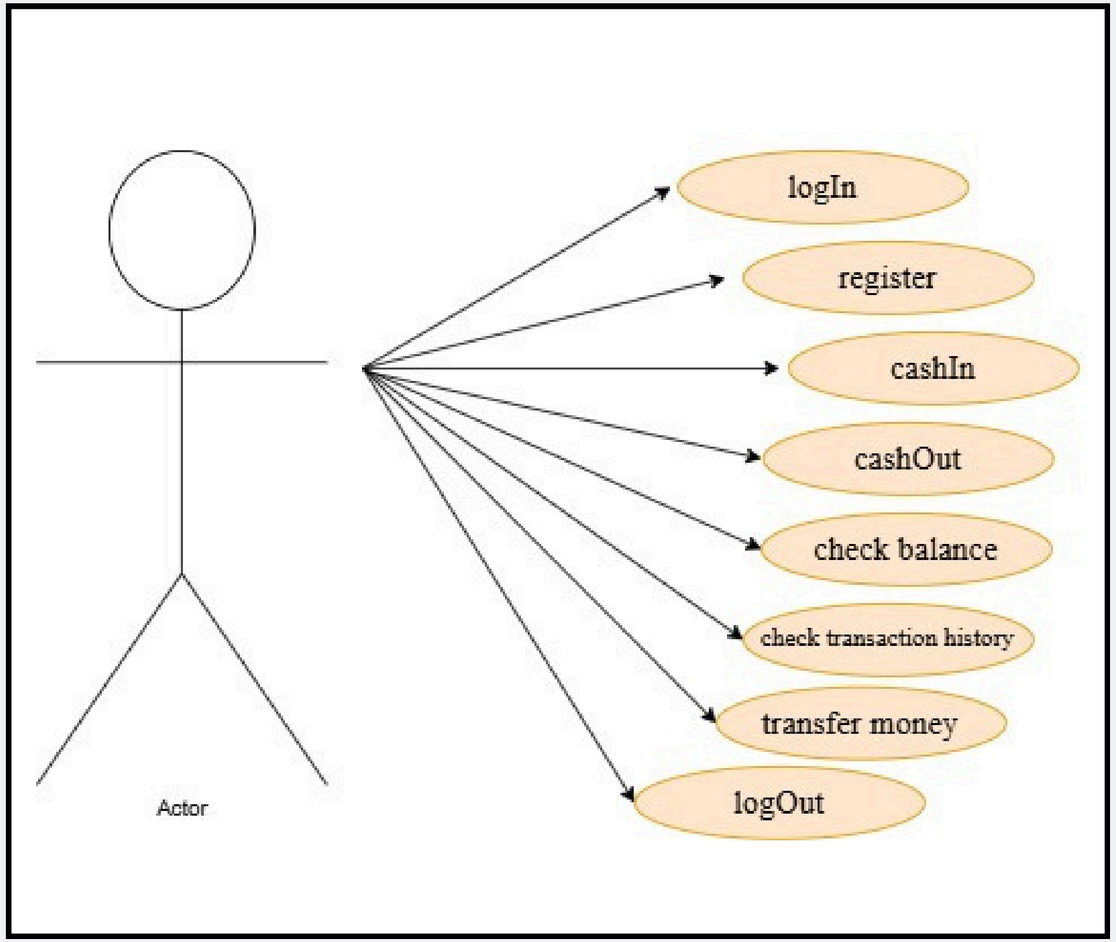
CCOBJPRG - INF246

UML & USE CASE DIAGRAM

UML & USE CASE DIAGRAM

Unified Modeling Language (UML) Diagrams





Use Case Diagram

Design Pattern

Strategy Design Pattern

Our system follows the **Strategy Design Pattern**. The abstract class TransactionAction serves as the strategy interface, defining the common execute() method for all transaction types. Concrete classes such as CashInTransaction implement this interface, each providing specific logic for different types of transactions. The system can dynamically choose and run the appropriate transaction strategy based on the user's selected action. This approach keeps the code clean, organized, and easier to maintain, especially when adding new transaction types in the future.

EABANK | SPECIFICATION | UML & USE CA DIAGRAM | DESIGN PATTE | USER INTERFACE

CCOBJPRG - INF246

User Interface

USER INTERFACE 1:

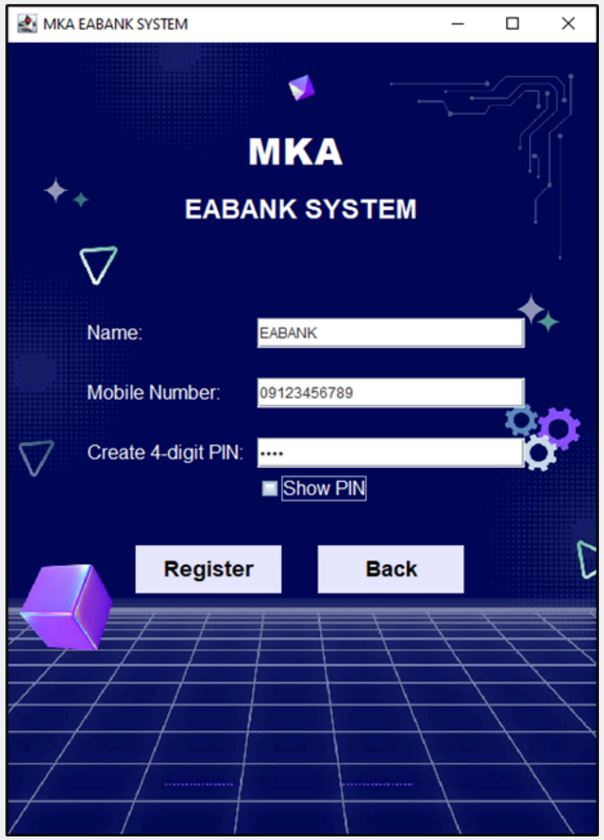
Main Menu Screen

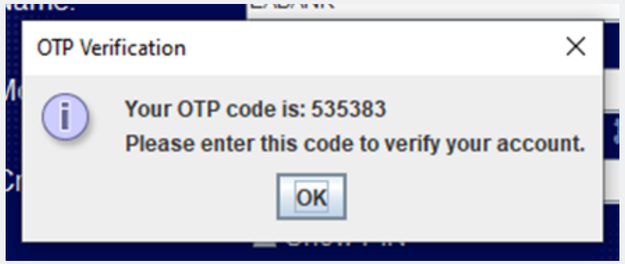
- This is the first screen that appears when the app is opened. It has three buttons: Login, Register, and Exit.
- The Login button is for users who already have an account.
- The Register button is for new users who want to sign up.
- The Exit button closes the app. This screen helps users get started and choose what they want to do next in the system

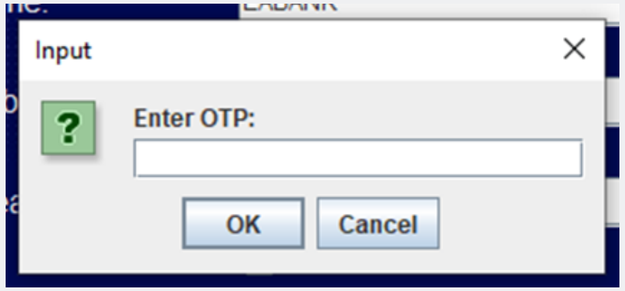
USER INTERFACE 2:

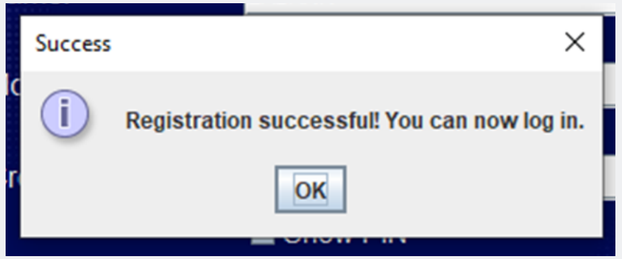
Register Functionality

- It is Used for registering a new account.
- The user fills in their name, mobile number, and creates a 4-digit PIN. They can also choose to show or hide the PIN while typing. After clicking the "Register" button, the system generates a one-time password (OTP), which is shown in a pop-up message on the screen. The user must then enter this OTP in the provided box to confirm their identity. Once the correct OTP is entered, a message appears confirming that the registration was successful.





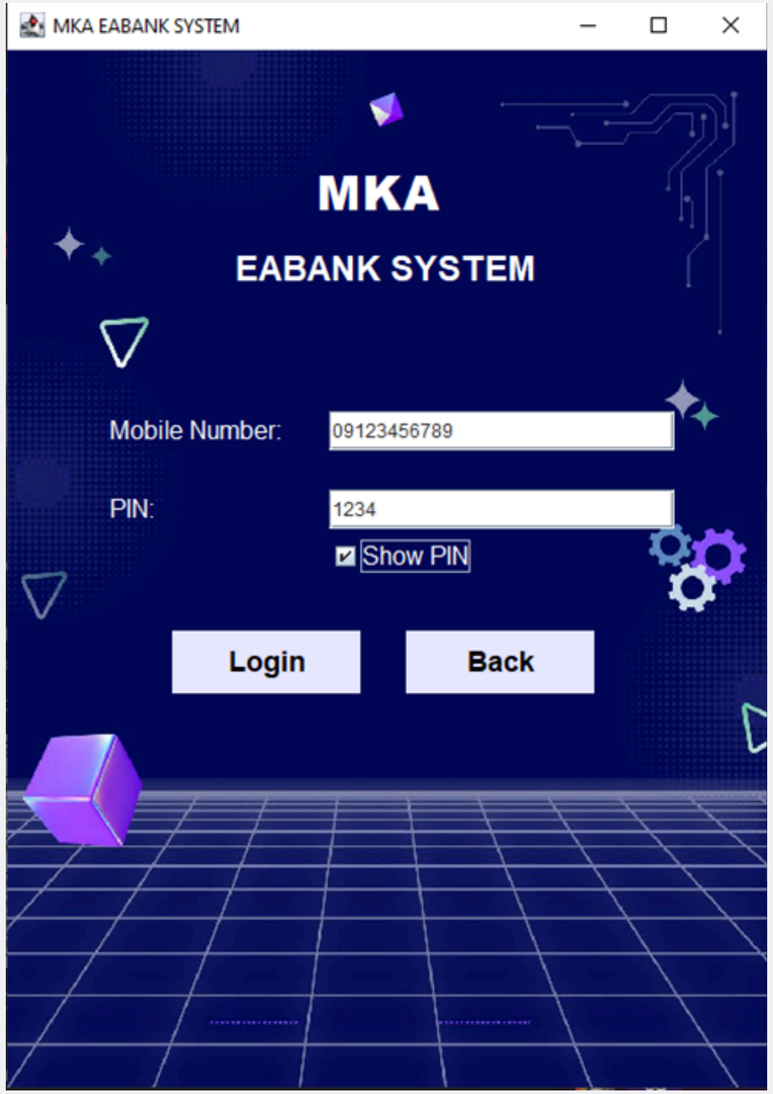




USER INTERFACE 3:

Login Functionality

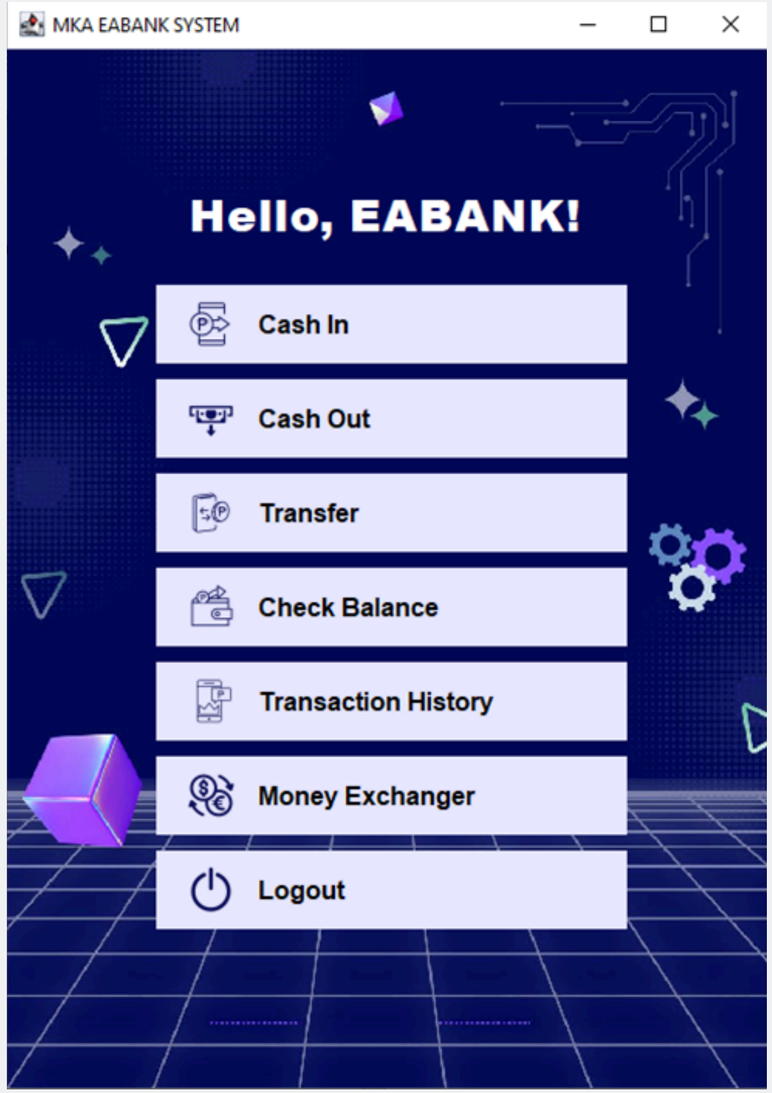
- It is used for logging users account.
- The user is asked to enter their mobile number and 4-digit PIN that they created during registration. There is an option to show or hide the PIN while typing. Once the user enters the correct information and clicks the "Login" button, the system checks if the details are correct.

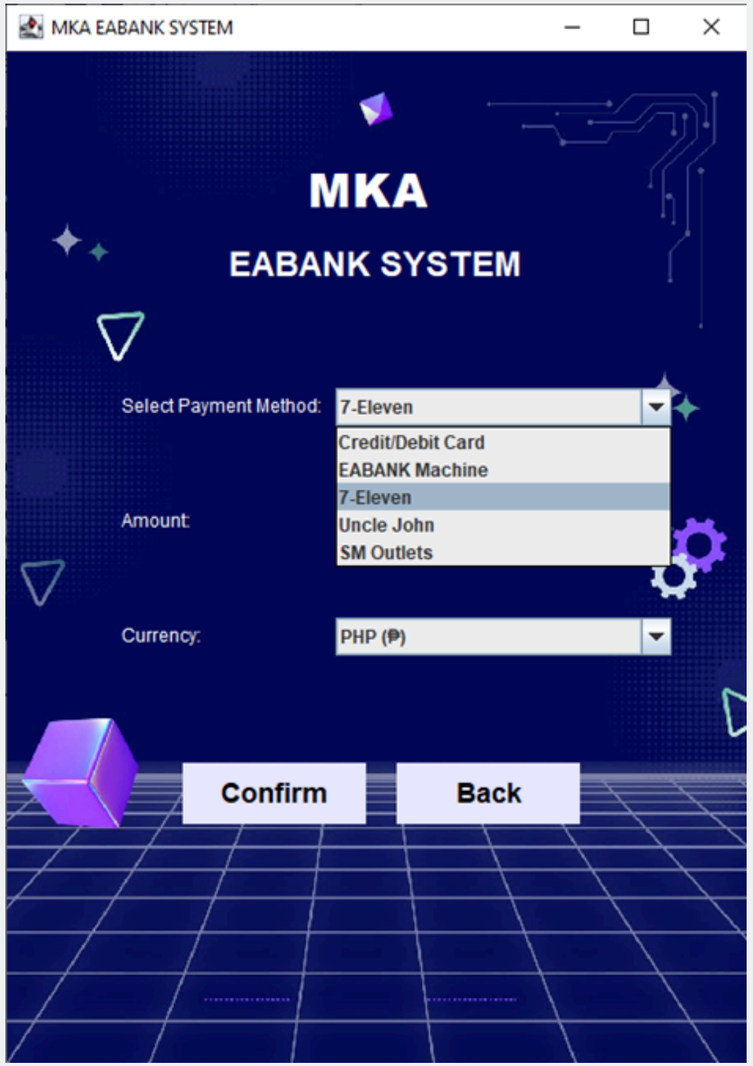


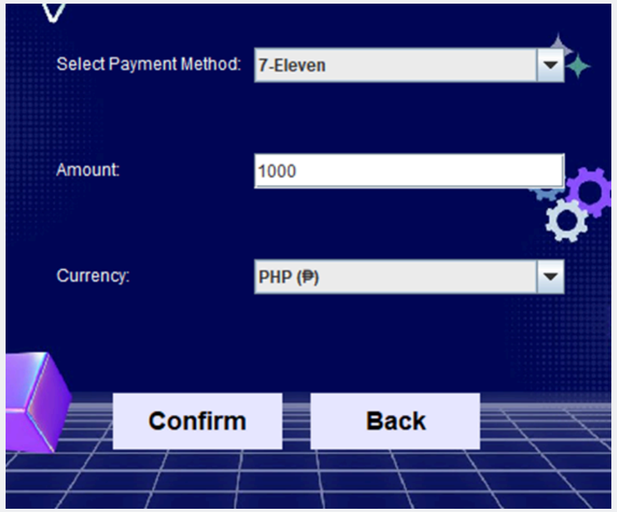
USER INTERFACE 4:

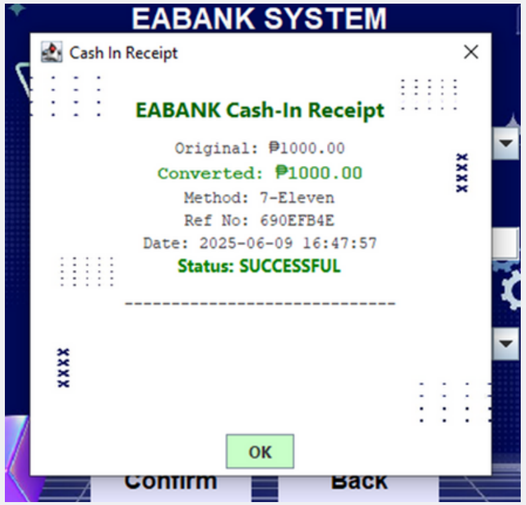
Menu Navigation

- It gives access to important features like Cash In to add money, Cash Out to withdraw money, Transfer to send money to other users, Check Balance to view the current account balance, and Transaction History to see past transactions. It also includes a Money Exchanger feature, which allows users to convert money between different currencies.
- At the bottom, there is a Logout button that safely logs the user out and returns them to the login or register screen.





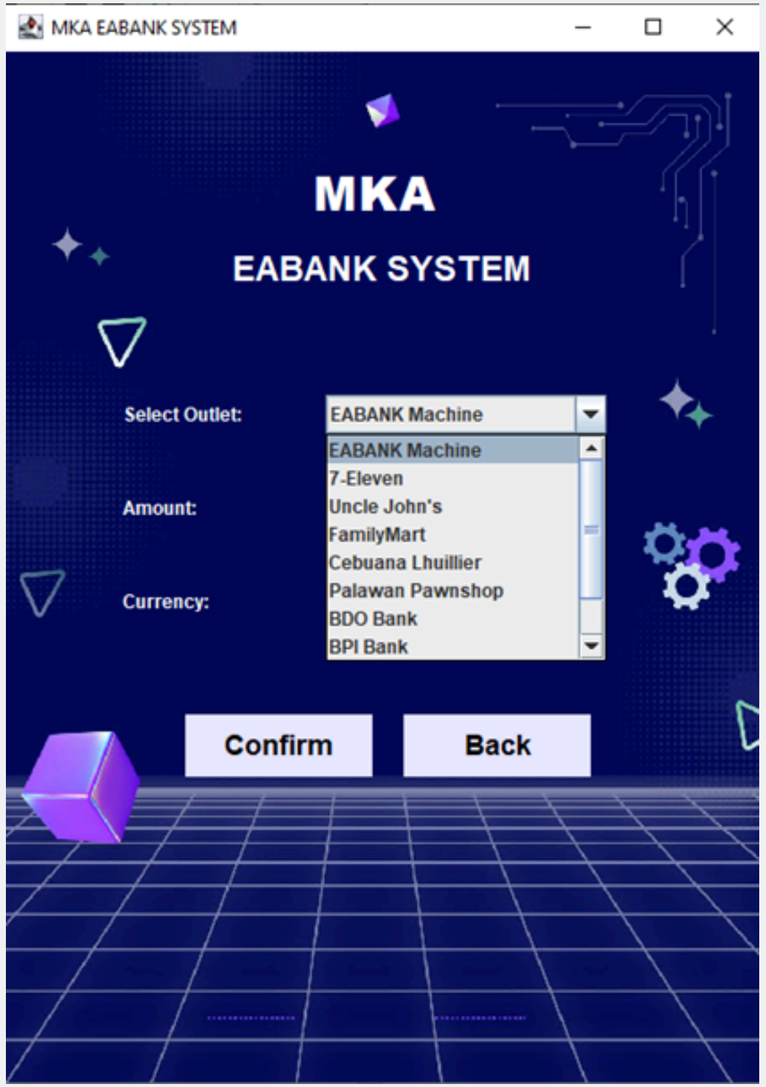


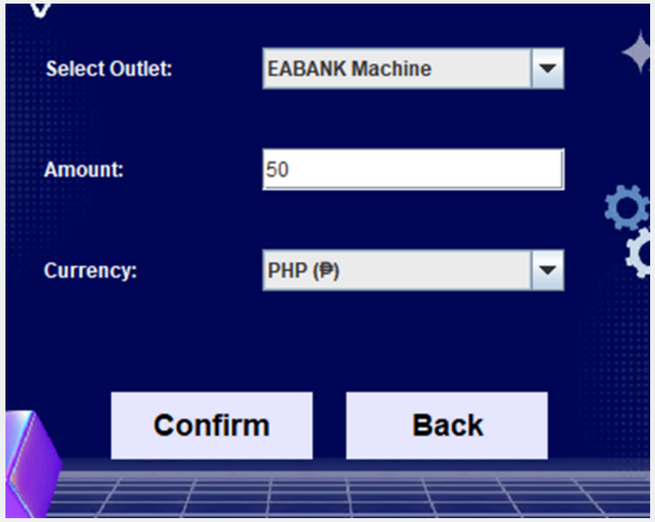


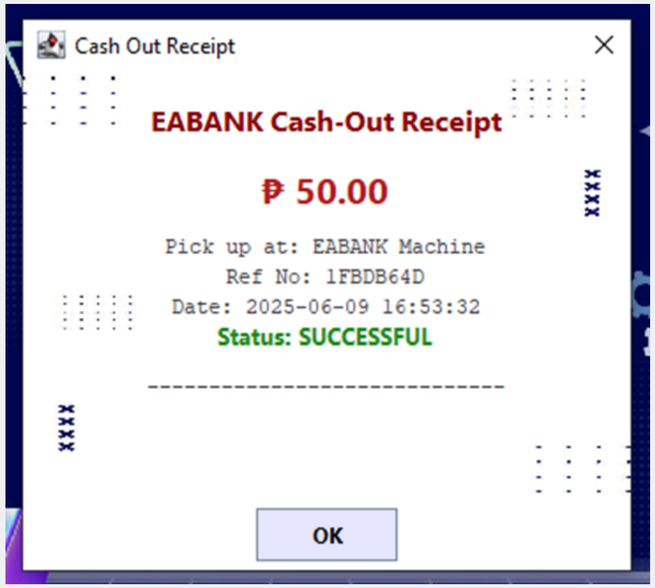
USER INTERFACE 5:

Cash In Functionality

Users can cash in to their account. They enter the amount, choose a payment method like 7-Eleven or credit card, and select the currency (PHP). Users can click "Confirm" to proceed or "Back" to return. After confirmation, a receipt appears showing the amount, payment method, reference number, date, time, and transaction status.



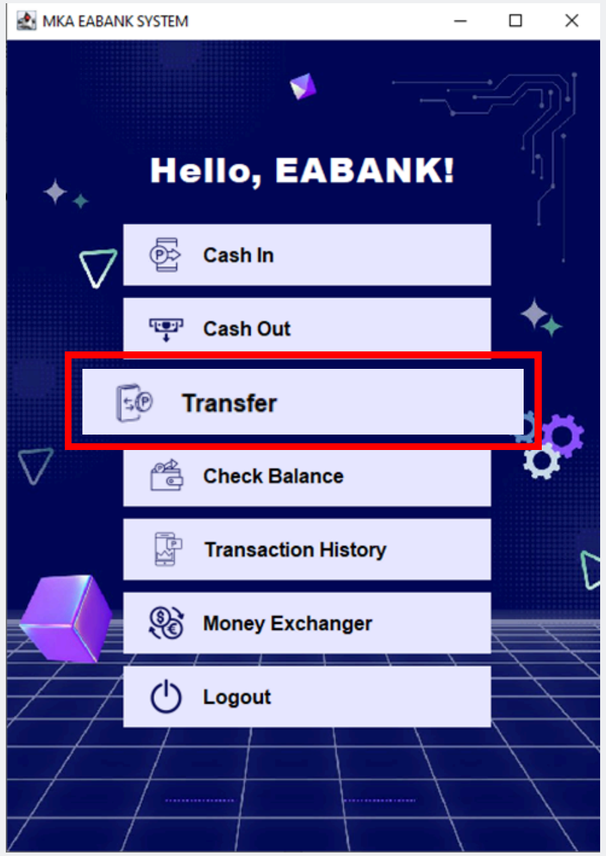


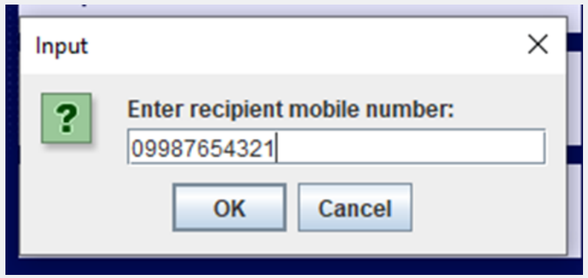


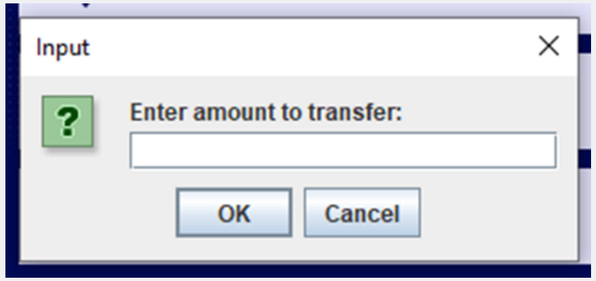
USER INTERFACE 6:

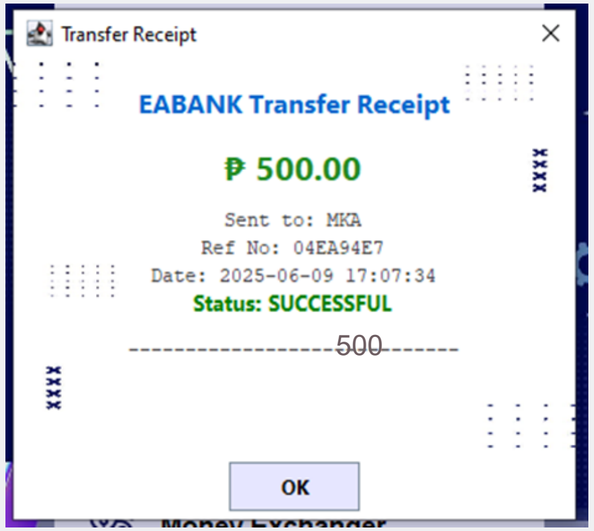
Cash Out Functionality

It handles the withdrawal process in the MKA EABank System. Users select an outlet such as Uncle John's, Family Mart, or 7-Eleven. They then enter the amount, select a currency, and can convert it if needed. After confirming, a digital receipt appears showing the amount, reference number, date, and transaction status, providing a clear and secure cash-out experience.









USER INTERFACE 7:

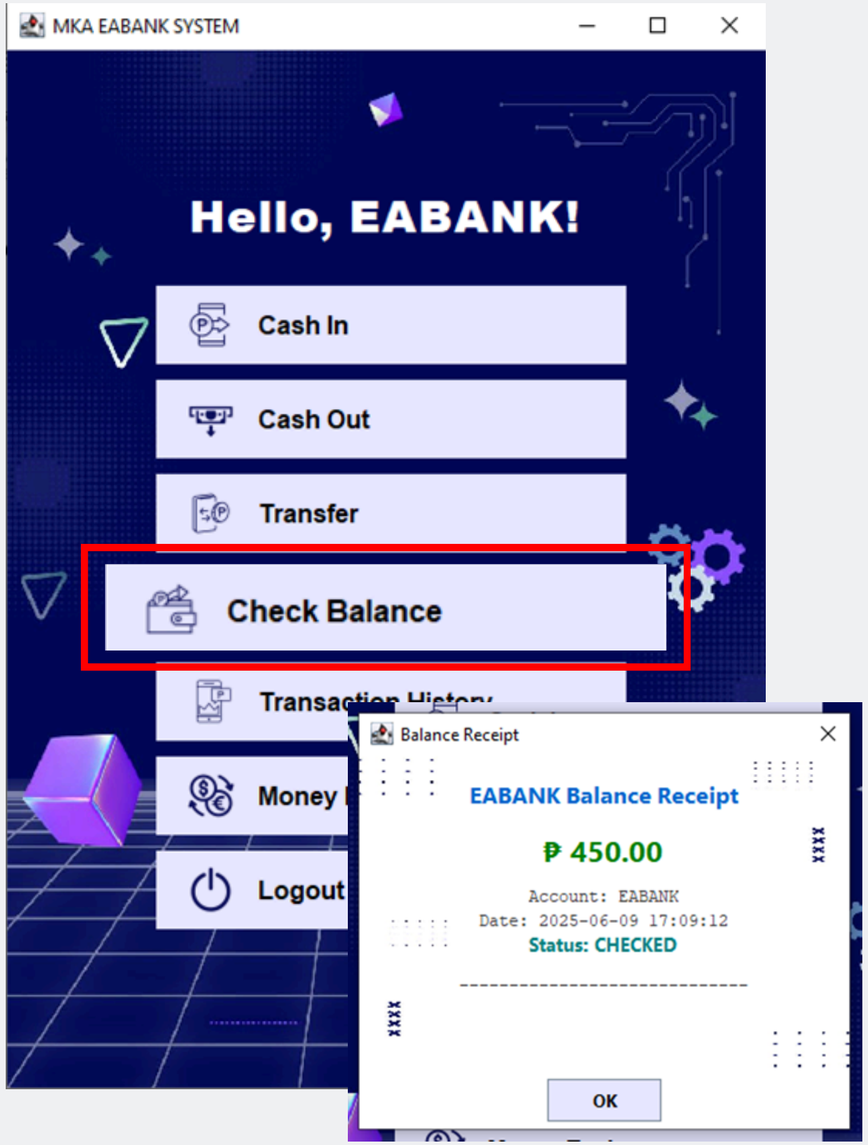
Transfer Funds Functionality

It allows users to send funds to another registered user. The sender enters the recipient's mobile number and the amount to transfer. The system verifies the recipient and checks the sender's balance. If valid, the amount is deducted from the sender and added to the recipient. A receipt is shown with the amount, recipient, reference number, date, time, and transaction status. The transaction is also recorded in both users' histories.

USER INTERFACE 8:

Check Balance Functionality (Balance Receipt)

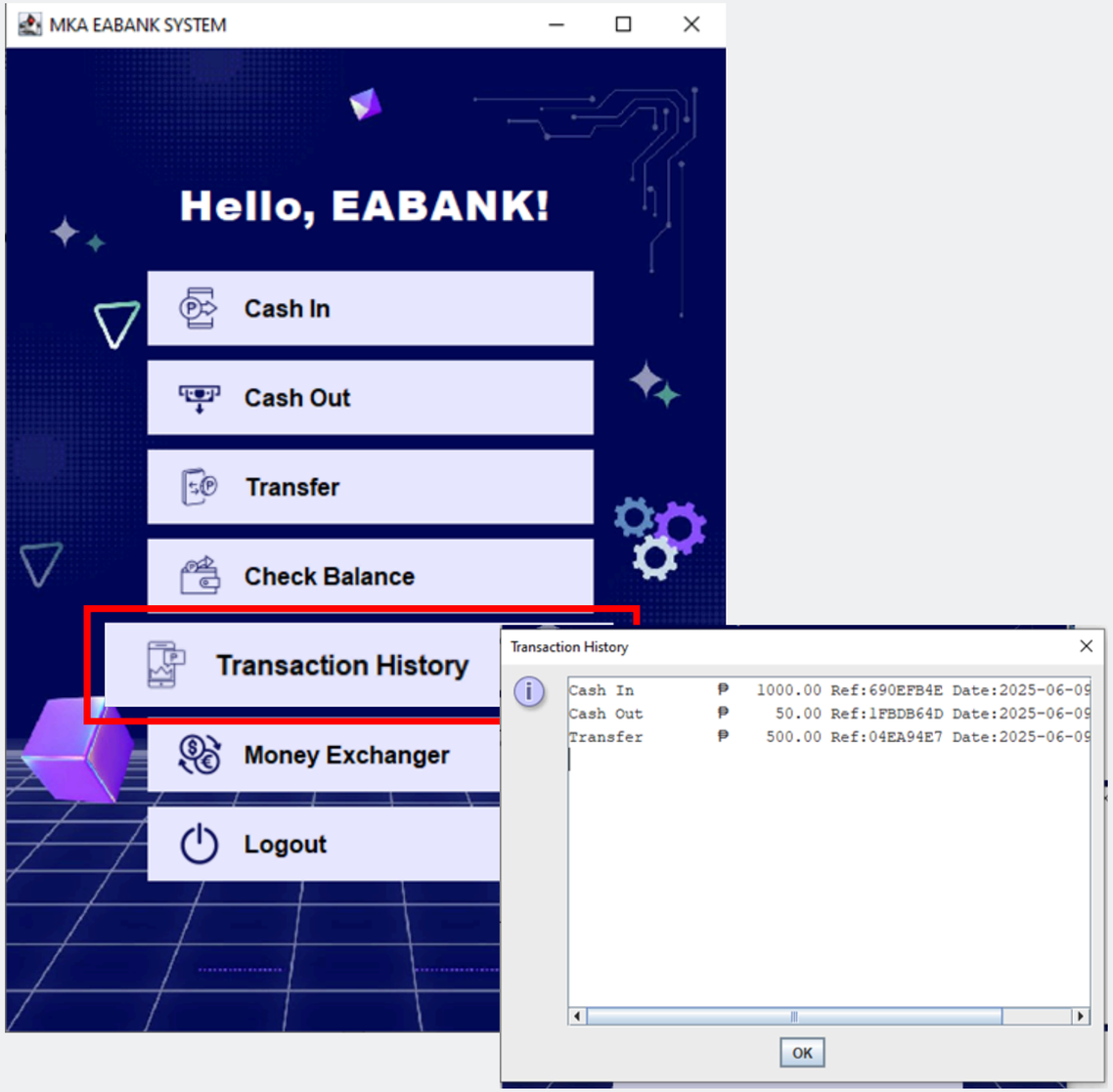
It displays a balance receipt showing the user's current account balance, account name, date, time, and status. It reflects the most recent transactions and can be accessed anytime from the main menu, providing a quick and convenient way for users to view their financial status.

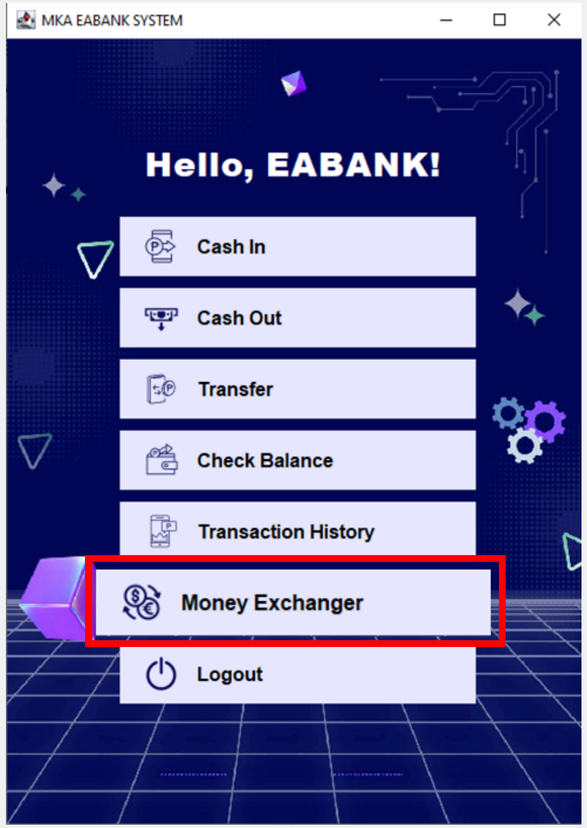


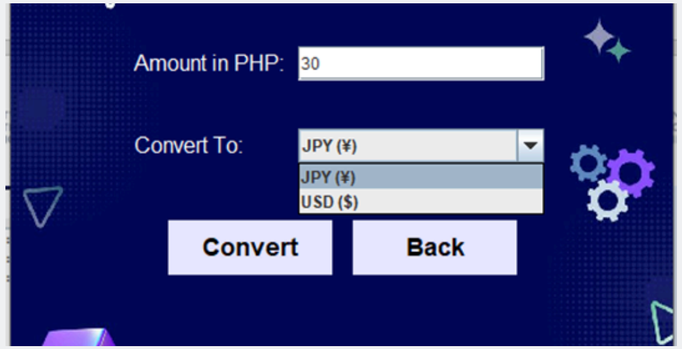
USER INTERFACE 9:

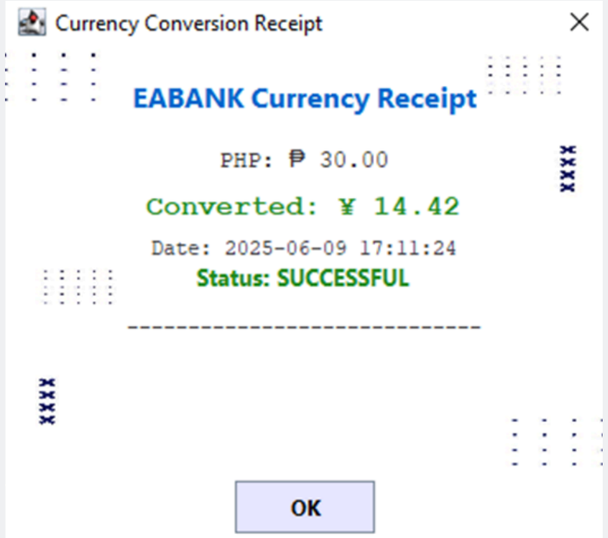
Transaction History Funtionality

It shows a list of the user's past transactions, organized by date and time. Each entry includes the transaction type (e.g., Cash In, Transfer), amount, reference number, and timestamp. If no transactions exist, a message is displayed. This feature helps users monitor account activity and manage their spending or deposits with ease.









USER INTERFACE 10:

Currency Converter (Money Exchanger)

It allows users to convert Philippine Peso (PHP) to other currencies like Japanese Yen (JPY) or US Dollar (USD). Users enter the amount in PHP and select a target currency. After clicking "Convert," a receipt appears showing the original amount, converted value, date, time, and conversion status. This feature helps users check currency values and keep a record of the conversion.

Code Snippet

EABANK | SPECIFICATION | UML & USE CA DIAGRAM | DESIGN PATTE | USER INTER | CODE SNIPPET

USER CLASS

```java
// User Class
static class User {
    private final String name;
    private final String mobile;
    private final String hashedPin;
    private double balance;
    private final ArrayList<Transaction> transactionHistory;

    public User(String name, String mobile, String pin) {
        this.name = name;
        this.mobile = mobile;
        this.hashedPin = hashPin(pin);
        this.balance = 0.0;
        this.transactionHistory = new ArrayList<>();
    }

    private String hashPin(String pin) {
        return Integer.toHexString(pin.hashCode());
    }

    public boolean checkPin(String pin) {
        return this.hashedPin.equals(hashPin(pin));
    }

    public String getName() {
        return name;
    }
```

```java
    public String getMobile() {
        return mobile;
    }

    public double getBalance() {
        return balance;
    }

    public void cashIn(double amount) {
        balance += amount;
        transactionHistory.add(new Transaction(type:"Cash In", amount));
    }

    public void cashOut(double amount) {
        balance -= amount;
        transactionHistory.add(new Transaction(type:"Cash Out", amount));
    }

    public void transfer(User recipient, double amount) {
        this.balance -= amount;
        recipient.balance += amount;
        transactionHistory.add(new Transaction(type:"Transfer", amount));
        recipient.transactionHistory.add(new Transaction(type:"Received", amount));
    }

    public ArrayList<Transaction> getTransactionHistory() {
        return transactionHistory;
    }

    public Transaction getLastTransaction() {
        if (transactionHistory.isEmpty())
            return null;
        return transactionHistory.get(transactionHistory.size() - 1);
    }
}
}
```

tRANSACTION CLASS

```java
// Transaction Class
static class Transaction {
    private final String referenceNumber;
    private final String type;
    private final double amount;
    private final LocalDateTime timestamp;

    public Transaction(String type, double amount) {
        this.referenceNumber = UUID.randomUUID().toString().substring(beginIndex:0, endIndex:8).toUpperCase();
        this.type = type;
        this.amount = amount;
        this.timestamp = LocalDateTime.now();
    }

    public String getReferenceNumber() {
        return referenceNumber;
    }

    public String getType() {
        return type;
    }

    public double getAmount() {
        return amount;
    }

    public String getTimestamp() {
        return timestamp.format(DateTimeFormatter.ofPattern(pattern:"yyyy-MM-dd HH:mm:ss"));
    }

    @Override
    public String toString() {
        return String.format(format:"%-15s ₱%10.2f Ref:%s Date:%s", type, amount, referenceNumber, getTimestamp());
    }
}
```

TRANSACTION ACTION CLASS

```java
abstract class TransactionAction {
    protected double amount;

    public TransactionAction(double amount) {
        this.amount = amount;
    }

    public abstract void execute(EABANKSYSTEM.User user);

    public abstract String generateReceipt(EABANKSYSTEM.Transaction transaction);
}
```

EABANK | SPECIFICATION | UML & USE CA DIAGRAM | DESIGN PATTE | USER INTER | CODE SNIPPET

CASH IN TRANSACTION CLASS

```java
class CashInTransaction extends TransactionAction {
    private String method;
    private String currency;
    private String accountName;
    private String accountNumber;

    public CashInTransaction(double amount, String method, String currency, String accountName, String accountNumber) {
        super(amount);
        this.method = method;
        this.currency = currency;
        this.accountName = accountName;
        this.accountNumber = accountNumber;
    }

    @Override
    public void execute(EABANKSYSTEM.User user) {
        double convertedAmount = convertToPHP(amount, currency);
        user.cashIn(convertedAmount);
    }

    @Override
    public String generateReceipt(EABANKSYSTEM.Transaction transaction) {
        return String.format(
                format:"--- CASH IN RECEIPT ---%nMethod: %s%nOriginal Amount: %s %.2f%nConverted to PHP: ₱ %.2f%nAccount: %s (****%s)*%nRef N
                method, getCurrencySymbol(currency), amount,
                convertToPHP(amount, currency),
                accountName, accountNumber.substring(accountNumber.length() - 4),
                transaction.getReferenceNumber(),
                transaction.getTimestamp());
    }
```

EABANK | SPECIFICATION | UML & USE CA DIAGRAM | DESIGN PATTE | USER INTER | CODE SNIPPET

CASH IN TRANSACTION CLASS

```
    public double convertToPHP(double amount, String currency) {
        return switch (currency) {
            case "PHP (₱)" -> amount;
            case "USD ($)" -> amount * 55.5;
            case "JPY (¥)" -> amount * 0.48;
            default -> 0;
        };
    }

    public String getCurrencySymbol(String currency) {
        return switch (currency) {
            case "PHP (₱)" -> "₱";
            case "USD ($)" -> "$";
            case "JPY (¥)" -> "¥";
            default -> "";
        };
    }
}
```

Project Demonstration