1. **PROJECT PROPOSAL(FILE).**

➔ PROJECT DESCRIPTION

This is a project that looks to focus on giving the international students at ALU Rwanda a fair currency exchange rate that depends on uptodate exchange rates of the money they have in their local currency to RWF.
It is created to help them save the cost of having to get out of school to go exchange the money they have as well as ensure they get the correct exchange amount.

➔ PROBLEM STATEMENT

Brenda is a student at ALU Rwanda where there are over 40 nationalities with different currencies. She want to develop a simple transaction user interface
that will help other student to change their money
to RWF. This will help to save student's time as they won't have to move out of school to
make a simple transaction that requires currency exchange.

➔ PROCEDURES AND METHODS

We will ask the users for some details about them(this info will be stored in a database for easy future transactions), then provide them with a menu of what they want to do, they can either; withdraw money from the phone number they registered with us or exchange manually.
Withdrawal:

This will involve them sending money to us in their local currency form, the money will then be converted to RWF and the amount we should give them displayed on the screen.

Manual;

Here, they go directly to the currency converter, input their currency type, the amount and the amount in RWF we should give them will be displayed on the screen for transparency. They then give us the money and we give them the displayed amount in RWF.

➔ ANTICIPATED OUTCOMES

We expect it to be user friendly and be able to solve the problem of distance and time for a currency exchange transaction.

2. **MAIN : LIST OF CLASSES AND THEIR RELATIONSHIPS(FILE).**

- Main class : brenda_forex()
  - class Customer(brenda_forex)
  - class Transactions (brenda_forex):
    - Withdrawing(self)
  - class Converter(brenda_forex):
    converting(self)
  - class User Interface(brenda_forex)
    user _interface(self)

3. **CUSTOMER INFO (FILE).**

Pseudo code:

Create a brendah_forex class
Define init to declare self, customer_name,customer_phone
Declare self.customer_name
Declare self.customer_phone

Declare self.customer_country
Ask user to input name
Ask user to input their phone number
Ask user to input country
Print welcome message from brendah_forex
Test brendah_forex class

Tests

Test number	Description	Test data	Expected result	Actual result	Pass/Fail
1	User to input their name	Name: JOSEPHINE	The name in uppercase	JOSEPHINE	Pass
2	User inputs their phone number	Phone number: + xxxx	Number should have a correct country code	Phone number starting with a country code	Pass
3	User input their country	Country:country name	Country should be different to rwanda	Country name	Pass
4	User inputs their phone number	Phone number: + xxxxxx	Phone number with a valid country code	Phone number with no country code	Fail
5	User to input their name	Name: JOSEPHINE	The name in uppercase	JOSEPHINE	Pass

4. **<u>TRANSACTION CLASS(FILE).</u>**

Pseudo code:

import brendah_forex class to Transaction file
Create a transaction class
Define init containing customer_name,customer_phone,customer_country,destination of money.
Define withdraw method
Ask user if they want to withdraw(give destination of the money) money or exchange manually
Ask for the amount they want to transact
Store the user info

<u>Tests</u>

Test number	Description	Test data	Expected result	Actual result	Pass/Fail
1	Compare phone number with password attached to it	Phone number: 07xxxxxx Password: *****	Password and name should match	Password and name should match	Pass
2	Compare phone number with password attached to it	Phone number: 07xxxxxx Password: *****	Password and name should match	Password and name should match	Pass
3	Compare phone number with password attached	Phone number:	Password and name should	Password and name should	Pass

	to it	07xxxxxx Password: *****	match	match	
4	Compare phone number with password attached to it	Phone number: 07xxxxxx Password: *****	Password and name should NOT match	Password and name should NOT match	Fail
5	Compare phone number with password attached to it	Phone number: 07xxxxxx Password: *****	Password and name should match	Password and name should match	Pass

5. CURRENCY CONVERTER CLASS(FILE).

Pseudo code:

Create converter file

Import transaction class to converter file

Create converter class

Create converting method

Get transactional information about the user

Get the current conversation rates as per the day updates

```
Algorithm currency convertor ()
INPUT: Currency type and currency value from user.
OUTPUT: Currency value and equivalent US dollar value
{
  Option: = 0;
  Value: = 0;
  Converted: = 0;
  Write ("*** Currency Convertor ***");
  Write ("1. Canadian dollars to US dollars");
  Write ("2. Mexican pesos to US dollars");
  Write ("3. English pounds to US dollars");
  Write ("4. Japanese yen to US dollars");
  Write ("5. French francs to US dollars");
  Write ("Enter the choice:");
```

```
  Read (option);
  Write (“Enter the amount:”);
  Read (value);
  If (option: = 1),then
    Converted: = value * (1/1.468);
  Else if (option: = 2),then
    Converted: = value * (1/9.5085);
  Else if (option: = 3),then
    Converted: = value * 1.6433;
  Else if (option: = 4),then
    Converted: = value * (1/104.92);
  Else if (option := 5),then
    Converted: = value * (1/6.2561);
  Else
    Write (“Wrong Menu Selection”);
  Write (“Entered Value:”, value);
  Write (“US dollar equivalent:”, converted);
}
```

Tests

Test number	Description	Test data	Expected result	Actual result	Pass/Fail
1	Compare phone number with password attached to it	Phone number: 07xxxxxx Password: *****	Password and name should match	Password and name should match	Pass
2	Ask the user to input the amount to transact	Amount: > 0	The amount input is greater than 0	The amount input is greater than 0	Pass

3	User amount input to transact	Amount should be NOT be in RWF	Input amount should be NOT be in RWF	Amount should be NOT be in RWF	Pass
4	Get conversation rates updates per day	Rates should be in %	Rates of conversation per day should be update	Rates of conversation per day should be update	Pass
5	User inputs amount to transact	Amount:	Amount as float values	Amount in words	Fail

6. **USER INTERFACE CLASS(FILE).**

Pseudo code:

Create user_ interface class

Create interface method

Create Input box for all user inputs)

Create Dropdown menu(for the currency)

Create Output

- ➢ Buttons(to show the end of sth)

Tests

Test number	Description	Test data	Expected result	Actual result	Pass/Fail
1	See if the input box is clickable	Click the input box to display the the menu	The dropdown menu is displayed	Dropdown menu is displayed when the input box is clicked	Pass

2	See if the dropdown menu is working	Dropdown menu should have the list of different currency	Selected currency should be displayed in input box	Currency selected in input box	Pass
3	Check if the convert button is working	Button	Converted amount displayed to user	Converted amount displayed to user	Pass
4	See if the input box is clickable	Click the input box to display the the menu	The dropdown menu is displayed	Dropdown menu is displayed when the input box is clicked	Fail
5	Check if the convert button is working	Button	Converted amount displayed to user	Converted amount NOT displayed to user	Fail