Customer Class Manual Test Case Designs

Test Case 1: `validGetNameTest`

Objective: Verify that the correct name of the customer is returned.

Test Step	Expected Result
Call `customer.getName()`.	Returns `"jay"`.

Test Case 2: `invalidGetNameTest`

Objective: Ensure that an incorrect name does not match the customer's name.

Test Step	Expected Result
Call `customer.getName()`.	Does not return `"jaya"`.

Test Case 3: `validGetAddressTest`

Objective: Verify that the correct address of the customer is returned.

Test Step	Expected Result
Call `customer.getAddress()`.	Returns `"india"`.

Test Case 4: `validGetPhoneNumberTest`

Objective: Verify that the correct phone number of the customer is returned.

Test Step	Expected Result
Call `customer.getPhoneNumber()`.	Returns `"1234"`.

Test Case 5: `validGetUsernameTest`

Objective: Verify that the correct username of the customer is returned.

Test Step	Expected Result
Call `customer.getUsername()`.	Returns `"jay"`.

Test Case 6: `validGetPasswordTest`

Objective: Verify that the correct password of the customer is returned.

Test Step	Expected Result
Call `customer.getPassword()`.	Returns `"jay"`.

Test Case 7: `addAccountTest`

Objective: Ensure that an account can be added to the customer's account list.

Test Step	Expected Result
Create a new customer with no accounts.	Customer has 0 accounts initially.
Add a `CurrentAccount` to the customer using `addAccount()`.	Account is added successfully.
Call `customer.getAccounts()` and verify the size.	Size is 1.

Test Case 8: `applyForLoanTest`

Objective: Ensure loan application is submitted correctly.

Test Step	Expected Result
Call `customer.applyForLoan(4500)`.	Output contains: `"Loan application for 4500.0 submitted."`

Test Case 9: `getLoansTest`

Objective: Ensure that all loans applied for are tracked.

Test Step	Expected Result
Apply for loans with amounts `5500`, `350`, and `500`.	All loans are added successfully.
Call `customer.getLoans()`.	Returns a list of size 4 (including initial loan).

Test Case 10: `getExistingLoansTest`

Objective: Verify that approved loans can be marked as paid and removed from the existing loans list.

Test Step	Expected Result
Approve the first loan using `loans.get(0).approveLoan()`.	Loan is marked as approved.
Pay off the loan using `customer.payOffLoan(loanId, loanAmount)`.	Loan is marked as paid.
Call `customer.getExistingLoans()`.	Returns a list of size 0.

Test Case 11: `payOffLoansNotApprovedTest`

Objective: Ensure loans cannot be paid off if they are not yet approved.

Test Step	Expected Result
Attempt to pay off the first loan (`loans.get(0)`) without approving it.	Output contains: `"Loan is not yet approved!"`

Test Case 12: `payoffLoanNotFoundTest`

Objective: Ensure the system handles cases where a loan ID does not exist.

Test Step	Expected Result
Call `customer.payOffLoan(1656, 3500)` .	Output contains: `"Loan not found."`

Test Case 13: `payoffLoanApprovedTest`

Objective: Verify that approved loans can be paid off.

Test Step	Expected Result
Approve the first loan using `loans.get(0).approveLoan()` .	Loan is marked as approved.
Pay off the loan using `customer.payOffLoan(loanId, loanAmount)` .	Output contains: `"Loan paid off!"` .

Test Case 14: `getOlderLoansTest`

Objective: Verify that loans paid off are moved to the older loans list.

Test Step	Expected Result
Approve the first loan using `loans.get(0).approveLoan()` .	Loan is marked as approved.
Pay off the loan using `customer.payOffLoan(loanId, loanAmount)` .	Loan is marked as paid.

Test Step	Expected Result
Call `customer.getOlderLoans()`.	Returns a list of size 1.

Test Case 15: `validViewAccountTest`

Objective: Verify that account details are displayed correctly.

Test Step	Expected Result
Call `customer.viewAccount()`.	Output contains `"Account Number:"`, `"Balance:"`, and `"Owner:"`.

Test Case 16: `validViewTransactionsTest`

Objective: Verify that transaction history for an account is displayed correctly.

Test Step	Expected Result
Call `customer.viewTransactions()`.	Output contains `"Transaction history for account <account number>"`.

All-DU-Paths Coverage

1. **Getters/Setters**: All relevant getter methods are tested with valid and invalid values.
2. **Account Management**: Tests verify account addition and listing.
3. **Loan Management**: Covers all DU-paths for loan application, approval, payment, and categorization (existing vs. older loans).
4. **Account Actions**: Viewing account details and transaction history ensures all necessary output paths are covered.