Reflection on Code Local Bank GUI

When I updated the LocalBankGUI to handle multiple actions like "Add Account", "Remove Account", "Deposit", "Withdrawal", and "Check Balance" I encountered a few challenges that were essential to making the program work as expected.

1. Account Not Found Errors

Initially I didn’t include validation to check if an account existed before performing a Deposit or Withdrawal. This led to issues when users tried to perform actions on accounts that didn’t exist in the system. The solution was simple: I added a check to ensure that the account number exists in the `HashMap` before proceeding with any transactions.

Error Example:

```
if (!accounts.containsKey(accountNumber)) {
    JOptionPane.showMessageDialog(this, "Account does not exist!", "Error", JOptionPane.ERROR_MESSAGE);
    return;
}
```

2. Deposit/Withdrawal Input Errors

I realized that users could enter non numeric values in the Amount field which caused the program to crash when trying to parse these invalid inputs. To fix this I added a `try-catch` block to handle any invalid input and prompt the user to enter a valid number.

Error Example:

```
try {
    double amount = Double.parseDouble(amountText);
} catch (NumberFormatException ex) {
    JOptionPane.showMessageDialog(this, "Invalid amount. Please enter a valid number.", "Error", JOptionPane.ERROR_MESSAGE);
    return;
}
```

3. Account Duplication

Another issue I faced was with the Add Account action. Users were able to add the same account multiple times causing duplication in the system. I added a check to prevent the addition of an account if it already exists in the `HashMap`.

Error Example:

```
if (accounts.containsKey(accountNumber)) {

    JOptionPane.showMessageDialog(this, "Account already exists!", "Error",
JOptionPane.ERROR_MESSAGE);

    return;

}
```

4. Invalid Balance Input

While adding new accounts users sometimes entered invalid balances (like letters instead of numbers). This caused the program to fail when trying to set the initial balance. The fix was similar to the solution for the Amount field I added input validation to ensure that the Beginning Balance field only accepted valid numbers.

5. Account Removal Error

Finally he Remove Account feature initially allowed users to attempt to remove accounts that didn’t exist which caused confusion. I added a check to verify if the account existed before removing it from the `HashMap`.

Error Example:

```
if (!accounts.containsKey(accountNumber)) {

    JOptionPane.showMessageDialog(this, "Account does not exist!", "Error",
JOptionPane.ERROR_MESSAGE);

    return;

}
```