Bank teller

# Operating Systems

12-5-19

# Team members

* + Brandon Rowe
  + Conner Murray
  + Josh Howard



# Description

## Summary

Bank Teller uses a Raspberry Pi modified with a touch screen to display a Java interface, simulating a bank teller system which allows employees and customers to login and manage customer accounts.

The operating system relevance comes from modification of the Raspbian OS to handle the attached touchscreen component.

The bank teller software allows employees or customers to manage customer accounts with full access to basic functions like deposit, withdraw, balance inquiry, manage contact info, password/pin reset and debit card cancellation

The idea was that a customer could walk into a bank and transact without the need of employee assistance even though it would still be available.

The program offers similar functionality to that of an ATM that would be located inside the bank with more functionality. The best way to describe it would be online banking meets ATM.

# project Description

## Extensive

Parts

Raspberry pi is a single-board computer that comes without a pre-installed operating system or peripherals. We first had to flash a micro-sd card with a Linux operating system distribution known as Raspbian. Raspbian is specifically designed to operate the Raspberry Pi as a desktop computer and is based on the Debian Linux distribution. The board only comes with a power charger so the keyboard, HDMI cable and monitor are external hardware that must be purchased in order to use as a desktop. The operating system relevance comes from modifying the Raspbian OS to handle the attached touchscreen.

By default, the raspberry pi display goes through an onboard micro HDMI port to project onto a standard desktop monitor. The touchscreen is purchased separately and connected directly to the onboard GPIO slots. Plugging the screen into the GPIO will not work until the OS is modified to recognize the touchscreen and reset the parameters of the screen output. We removed the installed LCD library by typing the command `sudo rm -rf LCD-show` from the proper directory in the linux terminal and removing the default display settings. We then clone an LCD driver and framework that is specifically designed for the 320 X 480 display we purchased. Next we navigate to the git repository in terminal and type `chmod ./MHS35-show wre`. This allows the touch feature to be used and also project the OS onto the smaller display. The screen was too small to include a software keyboard so we used an external Bluetooth keyboard for typing, though selecting and maneuvering the program was achievable with the touch screen. In summary, our hardware includes the Raspberry Pi, touch screen and Bluetooth keyboard.

Interface

The Bank Teller software was written in java and utilizes a few classes for customer, employee, database/driver and user interface (GUI). Using a text-based database, the system tracks and records changes to customer accounts and transaction history by either an employee or the customer directly. The system has standard ATM functions such as withdraw and balance inquiry. We combine those functions with online functionality by giving users the ability to change their password, debit card pin, contact information as well as the ability to cancel their debit card and deposit funds. Users can login as a customer to manage their account or have an employee login to manage their account for them.

Screenshots.

 

# Conclusion

## reflection

Pros

We were very pleased with the overall project. We successfully connected the touch screen display and raspberry pi for full touch functionality as well as full access to the operating system functions and programs after modifying the OS. We were able to resolve issues with running java programs on the pi. These issues stemmed from Raspbian being unable to recognize multiple classes in Java. We could not call one class from a separate class if they were in different Java files. We were able to compile the source code and graphical user interface into a Java Jar file to run the program on the raspberry pi. In the end, we had a touch screen running our java program with full intended functionality.

Cons

Some areas we would have liked to improve was the graphical interface. The issues with the GUI were related to the OS and touchscreen. After modifying the OS, the actual OS display went lower than the touchscreen height. Meaning that when we ran our program, if we maximized the GUI to fit screen, the bottom part of our program would be cutoff. Given more time we could have anchored the UI to open in the same place from window to window while having it form-fit to the touchscreen size. For our demonstration, we had to move the windows around every time we clicked into a new window and re-adjust to the screen.

# References

Our group wrote the entire project’s code without references or assistance from outside sources. We used our educational knowledge of Java to develop the software and compile into an executable Jar file.

The only assistance we received was from the instruction manual that came with the touchscreen that was purchased. Link to the description below.

<https://www.amazon.com/Raspberry-Touchscreen-320x480-Monitor-Display/dp/B07WQW6H9S>

Link to Source Code

<https://github.com/brandon-rowe/BankTeller>

|  |  |  |
| --- | --- | --- |
| **Team Member** | **Weekly Hours** | **Contribution** |
| Brandon Rowe | 10 hours/week | 33.33% |
| Conner Murray | 10 hours/week | 33.33% |
| Josh Howard | 10 hours/week | 33.33% |

# Appendix

## Photos

 

 

 

## Code

<https://github.com/brandon-rowe/BankTeller>