LKPay

User Manual

INTRODUCTION

This manual provides the minimum hardware specifications, required software installations, and installation and maintenance procedures needed for the operation of the LKPay electronic payment system.

Prior to using the product, please read the manual carefully in order to develop familiarity with the LKPay's functions and utilities.

TERMS

Term	Description
LKPay	The developed cashless payment system for Live Kitchen.
RFID	An acronym for "radio-frequency identification." It is used to track inventory; a reader scans an RFID tag to read data.
NFC	Stands for "near-field communication." This technology is similar to RFID, but NFC devices can both act as a "reader" and "tag."
Raspberry Pi	A less expensive mini-computer but with similar processing power to a PC.
LCD	Shortcut for "liquid crystal display." It is an electronic monitor that uses a screen to present information.

PRODUCT OVERVIEW

Features

1) Tap and Pay Functionality

With its RFID functionality, the MFRC522 card reader is capable of wirelessly scanning cards 3 cm away from its surface at the 13.56MHz frequency.

2) Fast and Quick Checkout

Transactions are quick as contactless cards are scanned by the reader within milliseconds, with two-way data transfer rates of up to 424 kbit/s.

3) Balance Tracking and Cash-in

Balances can be checked and added upon similar to checkout and registration. There is no need to wait in line as the process is almost instantaneous.

4) Easy Registration Process

Registration will only require the user's personal information and his/her ID for input into the cloud database. No need for passwords or two-factor authentication.

5) Secure Digital Transactions

Each contactless card has a 128-bit private key that is used to verify payment. It is unique to every transaction of an individual consumer, ensuring maximum security.

Hardware Specifications

The working prototype uses the following components and materials:

1) Raspberry Pi 3 Model B+*
2) Micro SD card
3) 16x2 LCD module
4) 10k potentiometer
5) MiFare RC522 card reader module
6) USB 18-key numeric keypad
7) Breadboard
8) Jump wires (male - male; male - female)
9) Ethernet cable

*Earlier models of the Raspberry Pi are not recommended to use as they have not yet been tested with the LKPay software.

SETUP & INSTALLATION

The LKPay system was adapted from the Raspberry Pi RFID Attendance System project by Emmet Young at PiMyLifeUp. As a result, the steps for the initial setup presented below is similar to it, with a few modifications. You may find the PiMyLifeUp project link here: https://pimylifeup.com/raspberry-pi-rfid-attendance-system/

Prerequisites

1) Raspberry Pi Setup

To setup your Raspberry Pi with Raspbian, refer to the official setup guide here: https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

Once Raspbian has been setup, update it by opening the terminal and inputting the following commands:

```
sudo apt-get update
sudo apt-get upgrade
```

Next, install the packages **build-essential**, **git**, **python3-dev**, **python3-pip**, and **python3-smbus** by running the following command:

```
sudo apt-get install build-essential git python3-dev python3-pip
python3-smbus
```

2) 16x2 LCD Setup

To setup the 16x2 LCD, have the following ready:

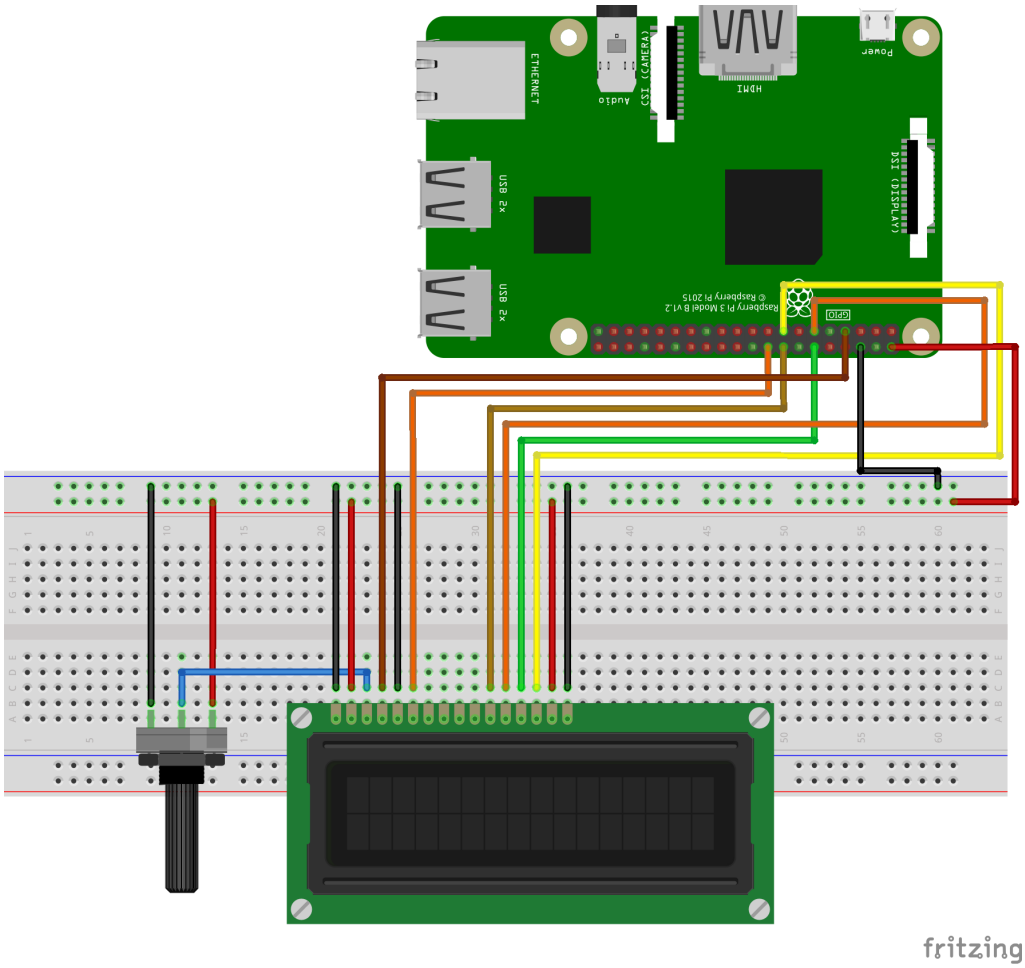
- Breadboard
- 16x2 LCD module
- 10k potentiometer
- 8 pcs. M - M jump wires
- 8 pcs. M - F jump wires

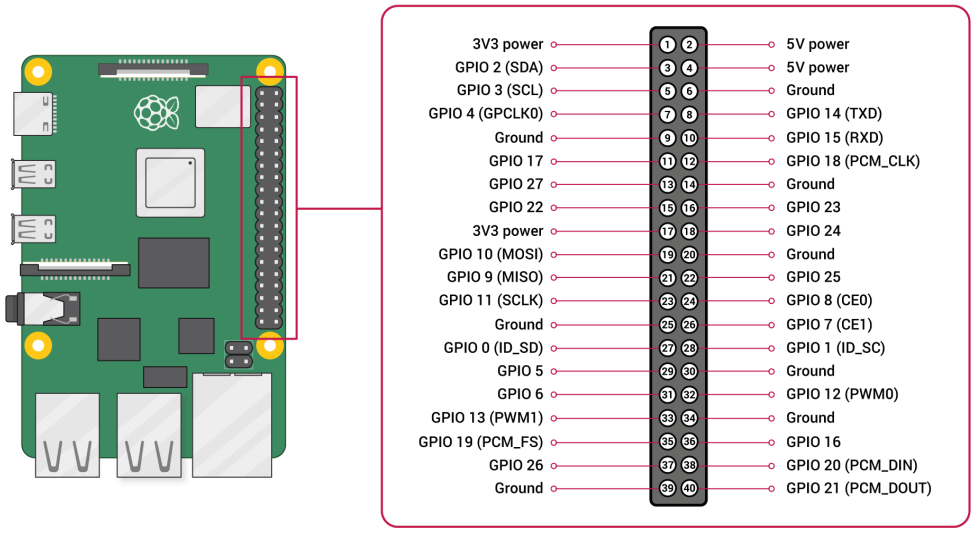
First, the LCD module has to be wired to the Raspberry Pi. For a visual guide, refer to the diagrams on the next page.

- **5V** (Physical Pin 2) to breadboard **positive** rail
- **Ground** (Physical Pin 6) to breadboard **ground** rail
- Place the **potentiometer** on the left side of the breadboard
- Place the LCD module to the right of the potentiometer
- **Left pin** of **potentiometer** to the **ground rail**
- **Right pin** of **potentiometer** to the positive rail

Next, connect the LCD display to the Raspberry Pi.

- **Pin 1** of **LCD** (Ground) to breadboard **ground** rail
- **Pin 2** of **LCD** (VCC / 5V) to breadboard **positive** rail
- **Pin 3** of **LCD** (V0) to **middle wire** of the **potentiometer**
- **Pin 4** of **LCD** (RS) to **GPIO4** (Physical Pin 7)
- **Pin 5** of **LCD** (RW) to breadboard **ground** rail
- **Pin 6** of **LCD** (EN) to **GPIO24** (Physical Pin 18)
- **Pin 11** of **LCD** (D4) to **GPIO23** (Physical Pin 16)
- **Pin 12** of **LCD** (D5) to **GPIO17** (Physical Pin 11)
- **Pin 13** of **LCD** (D6) to **GPIO18** (Physical Pin 12)
- **Pin 14** of **LCD** (D7) to **GPIO22** (Physical Pin 15)
- **Pin 15** of **LCD** (LED +) to breadboard **positive** rail
- **Pin 16** of **LCD** (LED -) to breadboard **ground** rail





RPi GPIO Diagram

Source: https://www.raspberrypi.org/documentation/usage/gpio/

Next, install the **Adafruit_CircuitPython_CharLCD** package by opening up the terminal and typing the following command:

```
sudo pip3 install adafruit-circuitpython-charlcd
```

After that, install the **Raspberry Pi GPIO Python library** by running the command:

```
sudo pip3 install RPi.GPIO
```

To test the LCD, run the following command to clone the **Adafruit-CircuitPython-CharLCD** files to the Pi:

```
git clone
https://github.com/adafruit/Adafruit_CircuitPython_CharLCD/tree/master/exam
ples
```

Then, edit the example script by running

```
nano ~/Adafruit_CircuitPython_CharLCD/examples/charlcd_mono_simpletest.py
```

Here, you'll find a section in the first part of the file that looks like this:

```
lcd_rs = digitalio.DigitalInOut(board.D7)
lcd_en = digitalio.DigitalInOut(board.D8)
lcd_d7 = digitalio.DigitalInOut(board.D12)
lcd_d6 = digitalio.DigitalInOut(board.D11)
lcd_d5 = digitalio.DigitalInOut(board.D10)
lcd_d4 = digitalio.DigitalInOut(board.D9)
lcd_backlight = digitalio.DigitalInOut(board.D13)
```

Replace the numbers in the `board.D` parts of the text to be like so:

```
lcd_rs = digitalio.DigitalInOut(board.D4)
lcd_en = digitalio.DigitalInOut(board.D24)
lcd_d7 = digitalio.DigitalInOut(board.D22)
lcd_d6 = digitalio.DigitalInOut(board.D18)
lcd_d5 = digitalio.DigitalInOut(board.D17)
lcd_d4 = digitalio.DigitalInOut(board.D23)
lcd_backlight = digitalio.DigitalInOut(board.D13)
```

After that, save the file by pressing **Ctrl + X** and pressing **Y**. Exit by pressing **Enter**, then put in the following command to run the script:

```
python3
~/Adafruit_CircuitPython_CharLCD/examples/charlcd_mono_simpletest.py
```

The LCD screen should display text. If it is hard to see, try adjusting the contrast with the potentiometer until it is clear.

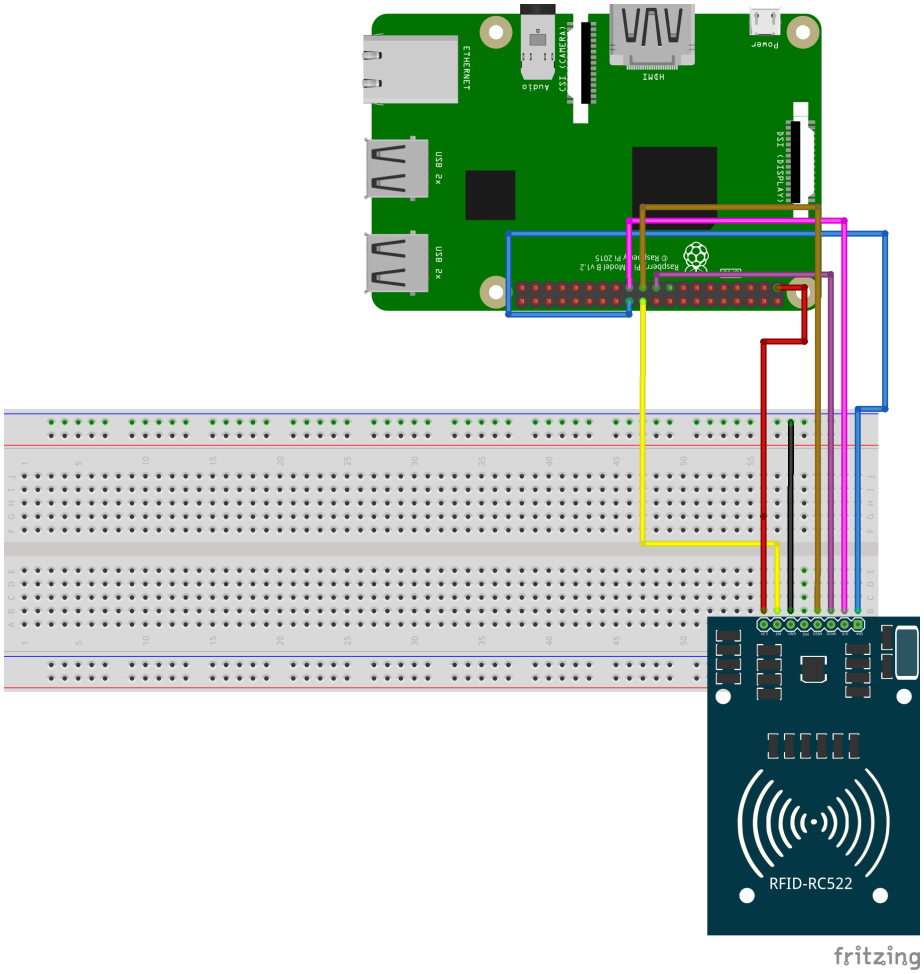
3) MFRC522 Setup

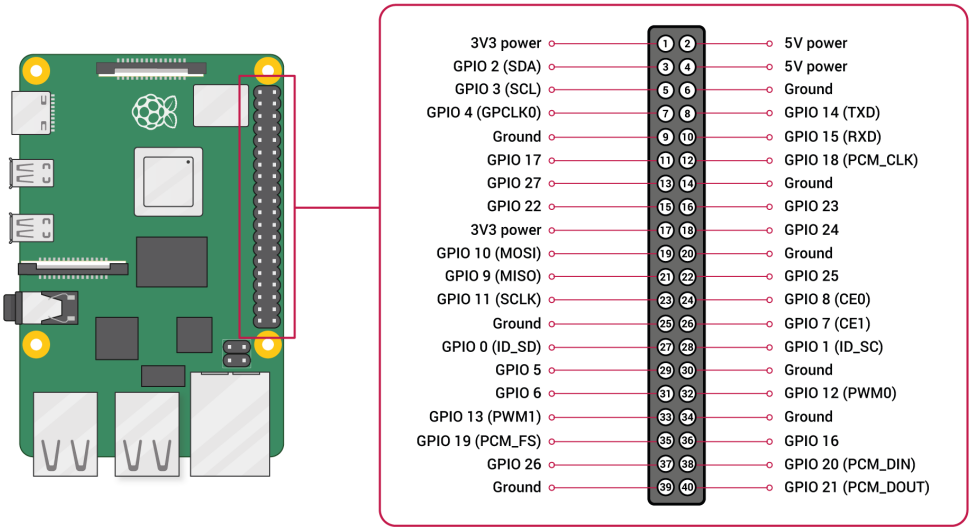
To set up the MFRC522 card reader, have the following ready:

- Breadboard
- MFRC522 card reader module
- 1 pc. M - M jump wire
- 6 pcs. M - F jump wire

Connect the MFRC522 module to the Raspberry Pi. For a visual guide, refer to the diagrams on the next page.

- **SDA** to **GPIO8** (Physical Pin 24)
- **SCK** to **GPIO11** (Physical Pin 23)
- **MOSI** to **GPIO10** (Physical Pin 19)
- **MISO** to **GPIO9** (Physical Pin 21)
- **GND** to breadboard **ground** Rail.
- **RST** to **GPIO25** (Physical Pin 22)
- **3.3v** to **3v3** (Physical Pin 1)





RPi GPIO Diagram

Source: https://www.raspberrypi.org/documentation/usage/gpio/

Next, you need to enable the SPI interface in order to let the Pi communicate with the MFRC522 module.

To do this, run the following command to launch the raspi-config tool:

```
sudo raspi-config
```

A screen should appear with various options. With your **arrow keys**, go down to "**5 Interfacing Options**" and press **Enter**.

Then, use your **arrow keys** to select "**P4 SPI**", and press **Enter**.

Select "**Yes**" to confirm enabling the **SPI interface** and press **Enter**.

Once the SPI Interface is enabled, restart the Pi by entering the following command:

```
sudo reboot
```

To verify that the SPI interface is enabled, run the following command:

```
lsmod | grep spi
```

An entry called "**spi_bcm2835**" should appear, which means that the SPI interface is indeed enabled.

Once that's done, install the spidev library to interact with the MFRC522 interface.

```
sudo pip3 install spidev
```

Then, install the MFRC522 library by using the command:

```
sudo pip3 install mfrc522
```

Now, to test the reader, create a short script by using **nano** through this command:

```
nano ~/pi-rfid/read.py
```

This will open up a text-editor in the terminal. Enter this python code:

```
#!/usr/bin/env python

import RPi.GPIO as GPIO
from mfrc522 import SimpleMFRC522

reader = SimpleMFRC522()
try:
        id, text = reader.read()
        print(id)
        print(text)
finally:
        GPIO.cleanup()
```

When finished, press **Ctrl + X** and **Y** to save the file, then press **Enter** to exit.

You may now test the reader by running the script through the terminal and tapping an RFID card or tag on the reader.

```
python3 ~/pi-rfid/read.py
```

The unique identifier (UID) number of the RFID tag that you tapped should appear in the terminal window.

LKPay System Setup

1) MySQL/MariaDB Database Setup

To setup MySQL/MariaDB (either will be fine) where all the data in the system will be stored, run the following command:

```
sudo apt-get install mysql-server -y
```

Next, to make the server more secure, run the “secure installation” script by running the following command:

```
sudo mysql_secure_installation
```

Enter your own password when prompted to set one, and answer “**Y**” to the other prompts.

Next, log in to the server by running the following command. Enter the password you set earlier when prompted.

```
sudo mysql -u root -p
```

Next, create a database with any name that you choose. For example, the following command will be named “lkpay”:

```
CREATE DATABASE lkpay;
```

Now, create a user and a password for it. The following command creates a user named “lkpayadmin” with the password “meridian”. You can set these to any username or password you want.

```
GRANT ALL PRIVILEGES ON *.* TO 'lkpayadmin'@'localhost' IDENTIFIED BY
'meridian';
```

Next, we have to select the “lkpay” database in MySQL by using the “**use**” command.

```
use lkpay;
```

Now, to create the tables we need for the system to store data, put in the following commands:

```
create table Accounts(
    id INT UNSIGNED NOT NULL AUTO_INCREMENT,
    rfid_uid VARCHAR(255) NOT NULL UNIQUE,
    name VARCHAR(255) NOT NULL,
    bal DECIMAL(7,2) NOT NULL,
    registered TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (id)
);

create table Transactions(
    id INT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE,
    rfid_uid VARCHAR(255),
    trans_amt DECIMAL(7,2) NOT NULL,
    acct_netbal DECIMAL(7,2) NOT NULL,
    trans_date TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (id),
    FOREIGN KEY (rfid_uid) REFERENCES Accounts(rfid_uid)
);
```

From this, we can see how all data will be stored into the database.

In the “**Accounts**” table, we have the following columns:

- “**id**” is an integer assigned to each record in the Accounts table.
- “**rfid_uid**” is the numerical UID of RFID tags that are registered in the database.
- “**name**” is the name of the account holder, which is inputted during registration.
- “**bal**” is the current balance of the account holder, stored as a decimal number.

- “**registered**” is the date and time that the user was registered.

In the “**Transactions**” table, we have the following columns:

- “**id**” is an integer assigned to each record in the Transactions table.
- “**rfid_uid**” is taken from the Accounts table as a **Foreign Key**, connecting the two tables. This makes it so each account can be looked up by their transactions, and vice versa.
- “**trans_amt**” is the amount exchanged in each transaction. A positive amount means that the amount was added to the account as balance, while a negative amount means that it was subtracted from the account.
- “**acct_netbal**” is the net balance of the account after the transaction.
- “**trans_date**” is the date and time of the transaction.

2) LKPay Script Installation

Now that the database and the components are working, we can now work on the LKPay Python script.

First, clone the **LKPay repository** by running the following command in the Terminal:

```
git clone https://github.com/andres-dalisay/LKPay
```

Next, open the **LKPay.py script** with nano by running the command:

```
nano /home/pi/LKPay/LKPay.py
```

Here, make sure that the database information in the **db section** of the code (at the beginning of the script) is the same as the **username, password, and database that you set earlier when making the database and user.**

```
db = mysql.connector.connect(
    host="localhost",
    user="lkpayadmin",
    passwd="meridian",
    database="lkpay"
)
```

In this case, the user set in the examples earlier was named “**lkpayadmin**”, with the password “**meridian**”, and the name of the database was set as “**lkpay**”.

After everything is set, press **Ctrl + X** and **Y** to save the file, and exit with **Enter**.

Now, the script is ready to run. Type in the following command to run it:

```
python3 /home/pi/LKPay/LKPay.py
```

OPERATION

---

1) ID Tapping

To scan an ID either for payment or registration, wait for the system prompt to scan. Scanning distance should be at the most 5 cm away from the reader surface in order to ensure optimal read times.

2) User Registration

NOTE: It is recommended to connect a keyboard and a monitor for user registration in order to input the names of the users.

a) Selecting the Registration Option

To select the user registration function, press **0** in the main menu. After that, press **1** in order to continue registration.

b) Placing the ID card

You will be prompted to place an ID card on the reader. If you have already registered using this card, the system will prompt you whether to overwrite or not.

c) Entering personal information

The system will have a prompt to enter the name of the ID holder. Preferably, type it in a “First Name Last Name” format. Press **Enter** when done.

d) Cancelling registration

To cancel registration, press any key other than **Y** during the overwrite or name prompts.

3) Transaction and Payment

a) Selecting the Payment Option

To select the payment function, press **1** in the main menu. After that, press **1** in order to continue payment.

b) Entering the Payment Amount

For the payment input, simply put any non-negative float value, i.e. 1.50 or 100, and press **Enter**. After, you will be prompted once more whether to continue with the transaction. Press **1** to continue.

c) Placing the ID card

You will be prompted to place an ID card on the reader. You will be notified if the transaction was successful when the “deducted funds” and “remaining balance” notifications show on the LCD.

d) Cancelling payment

To cancel payment, press any key other than **1** when prompted to proceed with payment.

4) User Cash-in & Load

a) Selecting the Load Option

In order to select the cash-in/load function, press **2** in the main menu. Press **1** in order to continue loading.

b) Entering the Reload Amount

For the reload input, simply put any non-negative float value, i.e. 1.50 or 100, and press **Enter**. After entering you will be prompted once more whether to continue with the transaction. Press **1** to continue.

c) Placing the ID card

You will be prompted to place an ID card on the reader. You will be notified if the transaction was successful when the “added funds” and “current balance” notifications show on the LCD.

d) Cancelling cash-in operation

To cancel reloading, press any key other than **1** when prompted to proceed with the cash-in.

5) Checking User Balance

a) Selecting the Balance Option

In order to select the balance check function, press **3** in the main menu.

b) Placing the ID card

You will be prompted to place an ID card on the reader. After doing so, you will be shown your current balance on the LCD.

MAINTENANCE, TROUBLESHOOTING & OTHERS

In this section, common errors, user questions ,system maintenance procedures, and developer information is discussed and resolved.

1. Common Error Messages

a. Invalid input.

The amount you entered when prompted to enter a payment/load amount is invalid. Make sure that it is a non-negative integer or decimal.

b. User not found.

The ID card has not yet been registered in the database. Register the card before attempting to make transactions with it.

c. Insufficient balance.

The balance of the account is not enough to complete the transaction. Make sure the account has enough balance to pay.

2. Common User Questions

a. Why is my reader not recognized by the Raspberry Pi?

Check the wire connections, and make sure the GND wire is plugged into the breadboard’s Ground rail.

b. Why does my LCD not display anything/have backlight but no text?

Check the wire connections. Try adjusting the potentiometer to control the contrast. If it still doesn't work, check the 5V and Ground lines of the breadboard to see if there is a "gap" between the two sides by column 30. If there is, try bridging the gap by using the jump wires to connect 5V-5V and GND-GND. Test it again.

c. Why is the LKPay software not connecting to the database?

If you aren't already, try and connect to a wifi network and run the script again.

d. Can I register two accounts for one card?

No. The system at its current version can only handle one account per card/tag.

3. Maintenance of Hardware

- Keep the hardware in a cool, dry place.
- Secure the Raspberry Pi in a place where no one can tamper with it.
- For any questions, contact the developer at apadx3304@gmail.com.

4. Developer Information

- You may find the LKPay repository at https://github.com/andres-dalisay/LKPay
- The code is open-sourced under GNU GPLv3 meaning that it can be modified and distributed for commercial and private use. More information on GNU GPLv3 can be found here: https://www.gnu.org/licenses/gpl-3.0.en.html

REFERENCES

Free Software Foundation. (2007, June 29). The GNU General Public License v3.0. Retrieved from https://www.gnu.org/licenses/gpl-3.0.en.html

Kumar, R. (2019, June 01). LCD - What is LCD: Construction and Working Principles of LCD Display. Retrieved from https://www.elprocus.com/ever-wondered-lcd-works/

Raspberry Pi Foundation. (n.d.). GPIO. Retrieved from https://www.raspberrypi.org/documentation/usage/gpio/

What is a Raspberry Pi? (n.d.). Retrieved from https://www.raspberrypi.org/help/what-is-a-raspberry-pi/

Want, R. (2011). Near field communication. *IEEE Pervasive Computing, 10(3), 4–7*. doi:10.1109/mprv.2011.55

Weinstein, R. (2005, August 1). RFID: A technical overview and its application to the enterprise. *IT Professional, 7(3), 27-28*. https://doi.org/10.1109/MITP.2005.69

Young, E. (2019, April 8). Build your own Raspberry Pi RFID Attendance System. Retrieved from https://pimylifeup.com/raspberry-pi-rfid-attendance-system/