

**Department of Electrical Engineering and Computer Science**

**Spring 2021**

**Automated Locker for Curbside Pickup**

**Build Document**

**Professor:** Dr. Hari Kalva

hkalva@fau.edu

**Sponsor:** Mahesh Neelakanta

mahesh@fau.edu

**Group 8 Members:**

**Project Manager:** Jacob Highbridge

jhighbridge2018@fau.edu

**Electrical Engineer:** Brandon Boukzam

bboukzam2014@fau.edu

**Computer Engineer:** Christian Cruz

cruzc2018@fau.edu

**Computer Science:** Yingqi Feng

yfeng2016@fau.edu

**Computer Science:** Tony You

[tyou2017@fau.edu](mailto:tyou2017@fau.edu)

Build Document

# **Hardware Requirements:**

* **AWS Serverless Database:**
  + Computer to access and modify database
  + AWS Http Web Server
    - AWS EC2 Linux Instance, 1GB RAM
* **Locker Kiosk:**
  + Raspberry Pi 3 Model B+
    - 32GB micro-SD
    - Mouse & keyboard
    - Monitor/display that is HDMI compatible
  + Raspberry Pi Camera Module rev 1.3 or similar
    - 15 pin flex cable – at least 127mm in length
  + Raspberry Pi 7” Display with adapter board
    - 15 pin flex cable – at least 76.2mm in length
  + 4 jumper cables – at least 101.6mm in length
  + 12V 6 channel relay
  + 12V to 5V ac to dc converter
  + 5A power supply
  + 6 12v 0.5A Solenoid Locks
  + 6 Metal Locker door latches
* **Admin Interface:**
  + Modern Tablet Computer

# 

# **Software Requirements:**

* **AWS Serverless Database:**
  + Any modern operating system with internet connectivity
  + Access to AWS
  + MySQL Workbench
  + Visual Studio Code
    - Remote Development Extension
* **Locker Kiosk:**
  + Raspberry Pi 3 Model B+
    - Latest Raspberry Pi OS with desktop and recommended software
    - Latest Chromium browser
* **Admin Interface:**
  + Internet browser, preferably Google Chrome

# **Packages/Dependencies:**

**AWS Serverless Database:**

* + AWS Http Web Server
    - PHP 7.3 or higher
    - Phpqrcode
    - PHPMailer
    - QRcode.js
    - Bootstrap

**Locker Kiosk:**

* + Raspberry Pi 3 Model B+
    - pip
    - python3-opencv
    - libqt4-test
    - python3-sip
    - python3-pyqt5
    - libqtgui4
    - libjasper-dev
    - Opencv-contrib-python 4.1.0.25

**Admin Interface:**

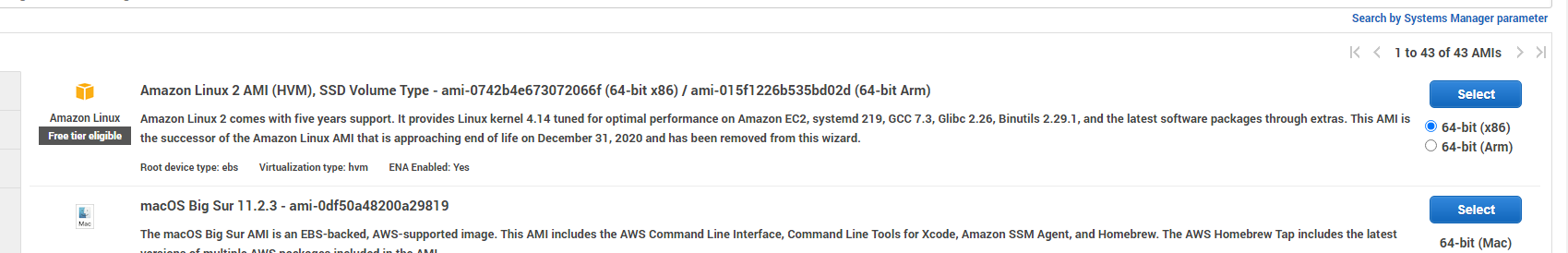
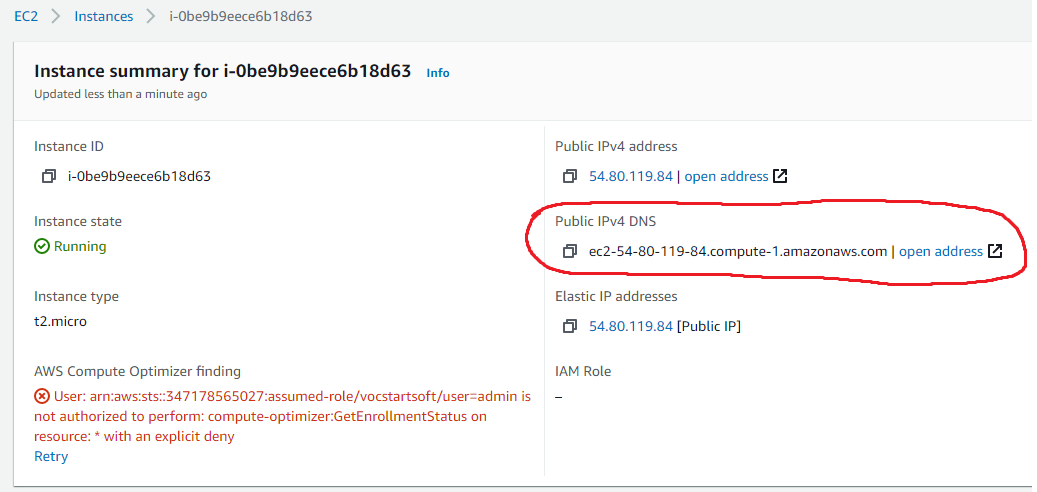
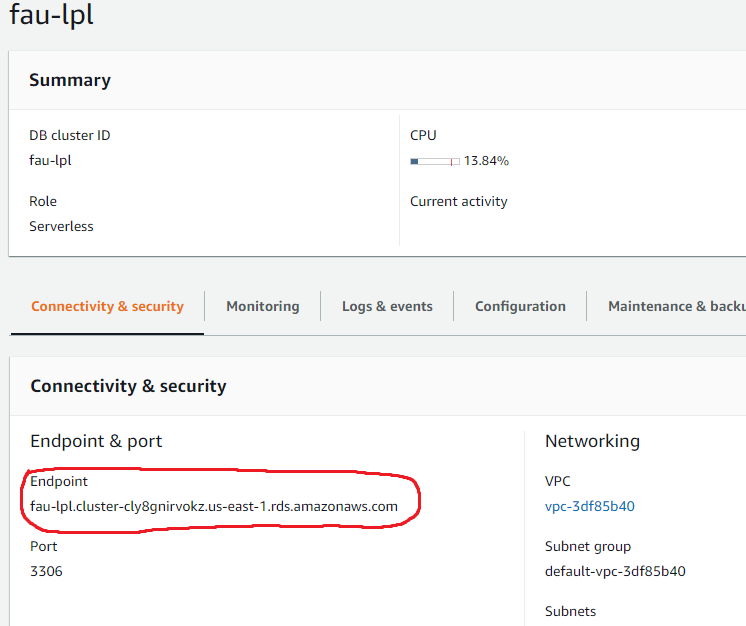
* + None

# **Instructions to Build and Run Software:**

**AWS Serverless Database**

* + - Creating Aurora Serverless Database
      1. Log into AWS and find Amazon RDS
      2. Create Database using these options:
         * Creation Method: Standard Create
         * Engine Type: Amazon Aurora
         * Edition: Amazon Aurora with MySQL Compatibility
         * Capacity Type: Serverless
         * DB cluster identifier: fau-lpl
         * Master Password: (could be anything, we used Group8pass)
         * Minimum capacity: 2GB RAM
         * Max capacity: 122GB RAM
         * VPC: Default
         * VPC Security Group: Create new
         * Additional Config:

Initial database name: lockers

* + - Creating EC2 instance
      1. Navigate to Amazon EC2
      2. Select “Launch Instance”
      3. Select the first instance, Amazon Linux 2
      4. Select t2 micro with 1 vCPU and 1GB RAM
      5. Review and launch, then click launch
      6. Create a new key pair, and name it whatever you want  
         (I named mine “library\_lockers”)
      7. Download the key pair and save it to somewhere safe and memorable
      8. Launch instance
    - Assigning Elastic IP to EC2 instance
      1. In Amazon EC2, find the “Elastic IPs” link under “Network & Security” on the left navigation column
      2. Then click “Allocate Elastic IP address” orange button
         * Don’t change any settings, just allocate
      3. Check off the IP address you just created, and then click “Actions”
         * Associate Elastic IP address
         * Select your current instance, and associate
    - Using MySQL Workbench to Access Database
      1. After downloading MySQL Workbench, press the plus button next to “MySQL Connections” to start a connection
         * Change the Connection Method to Standard TCP/IP over SSH
         * Set the SSH Hostname to be your EC2 Instance IPv4 address
         * Set SSH Username as “ec2-user”
         * On SSH Key File, navigate to your saved key pair
         * Set the MySQL hostname to the Database Endpoint
      2. Now test connection
         * First thing you should see is a quick error message, skip it
      3. Next is a password screen; enter the database password (Ours was Group8pass)
         * Given this, a successful connection is basically guaranteed

If not, then repeat all steps above. (Sorry, I know it can be annoying. Trust me.)

* + - 1. Now, set whichever name you would like to the connection, and then hit OK
      2. Double click the connection to connect to the database.
    - Creating Tables in Database using MySQL Workbench
      1. On the Query 1 Tab, copy this SQL:

use lockers;

CREATE TABLE `admin` (

`aid` int(11) NOT NULL AUTO\_INCREMENT,

`aname` varchar(20) DEFAULT NULL,

`aemail` varchar(45) DEFAULT NULL,

`password` varchar(20) DEFAULT NULL,

PRIMARY KEY (`aid`)

);

CREATE TABLE `lockers\_history` (

`lockerNum` int(11) DEFAULT NULL,

`lockerID` int(11) DEFAULT NULL,

`studentName` varchar(20) DEFAULT NULL,

`email` varchar(20) DEFAULT NULL,

`bookNum` int(11) DEFAULT NULL,

`referenceNum` varchar(255) DEFAULT NULL,

`dateAdded` varchar(30) DEFAULT NULL,

`adminName` varchar(45) DEFAULT NULL

);

CREATE TABLE `lockers\_info` (

`lockerNum` int(11) NOT NULL,

`lockerID` int(11) DEFAULT NULL,

`studentName` varchar(20) DEFAULT NULL,

`email` varchar(30) DEFAULT NULL,

`bookNum` int(11) DEFAULT NULL,

`referenceNum` varchar(255) DEFAULT NULL,

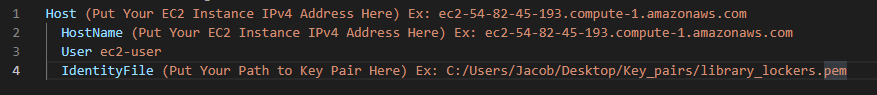
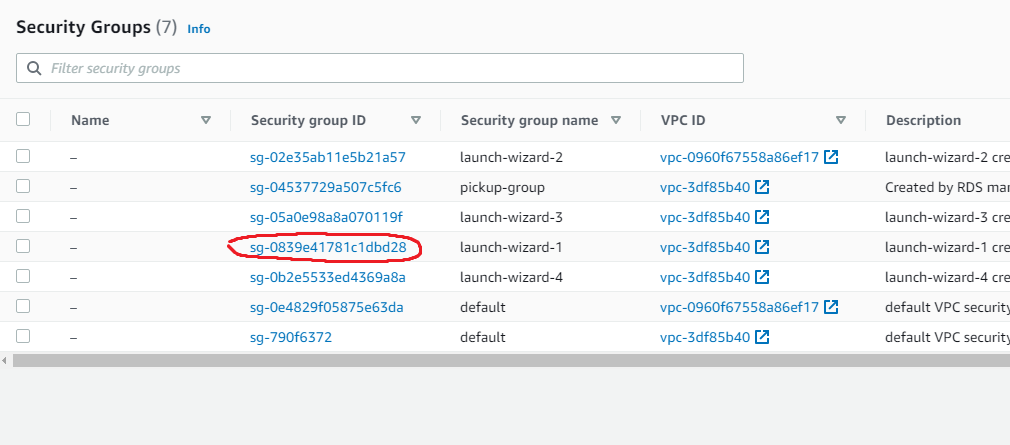
`lockerStat` varchar(7) DEFAULT NULL,

`lockerDate` varchar(45) DEFAULT NULL,

`adminAuth` varchar(45) DEFAULT NULL,

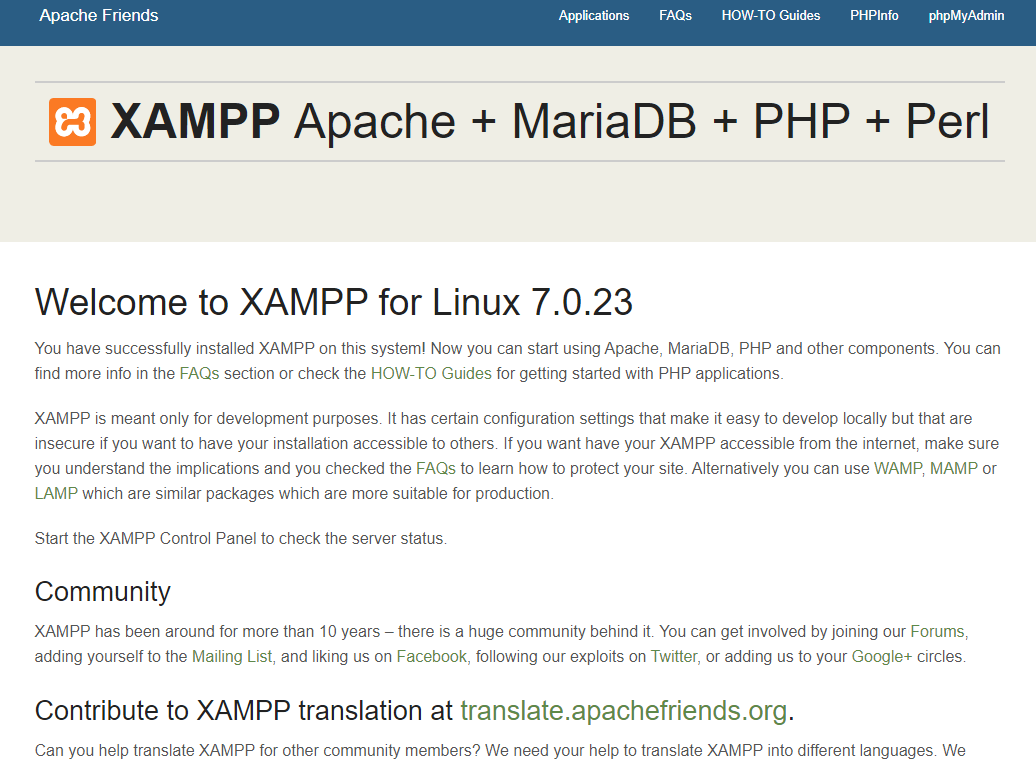
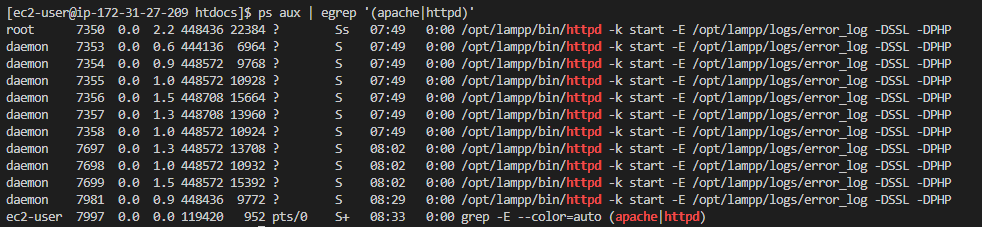
PRIMARY KEY (`lockerNum`)

);

* + - 1. This takes care of everything we need for the database
    - Connecting to EC2 instance through VScode Remote Dev
      1. In Visual Studio Code, search for and download the Remote Development extension from the extensions tab
      2. After installing the extension, press F1 to bring up the command palette and type “open SSH configuration file” then press enter
         * Press enter again to access your config file
         * Follow this template:  
             
           
      3. Once your SSH config file is put together, we save it; it’s time to connect to our instance
         * Press F1 to bring up the command prompt once again
         * Type “connect current window to host” and press enter
         * Now you should be able to see your EC2 IPv4 address to click, and you should click it to connect
         * When the prompt comes up asking which OS the remote platform is running on, click Linux
      4. Given no errors, you have successfully connected to the instance!
         * If you are having trouble, recheck the IPv4 address you typed in the SSH config file
    - Using the Terminal to set up an Apache server with XAMPP
      1. Using the terminal, download XAMPP for 64-bit using this command:  
         wget <https://www.apachefriends.org/xampp-files/7.0.23/xampp-linux-x64-7.0.23-0-installer.run>
      2. Then make an executable:   
         sudo chmod +x xampp-linux-x64-7.0.23-0-installer.run
      3. Run the installation: (enter Y for all questions)  
         sudo ./xampp-linux-x64-7.0.23-0-installer.run
      4. Now, we can start our server using the command:  
         sudo /opt/lampp/lampp start
    - Enabling HTTP Traffic to EC2 Instance
      1. In the Amazon EC2 console, find “security groups” on the left navigation column under “Network & Security”
         * Then, click on the blue Security group ID on your security group for the EC2 instance (most likely launch wizard 1)  
           
         * Then, click edit inbound rules
         * Click Add rule

Type HTTP into the type box, hit enter

Change the source to “Anywhere”

* + - 1. Test connection by entering your elastic IP into the address bar on Google Chrome (Or any other modern browser)
         * If you have completed these last steps correctly, you should see this page:
    - Using the EC2 Instance to Host Website
      1. Open the folder with the address:  
         /opt/lampp/htdocs
      2. From here, copy the src folders “pickup,” and “admin” to the folder (drag and drop)
         * If an error occurs, type “sudo chown -R ec2-user:ec2-user /opt/lampp/htdocs” into the terminal before trying again
      3. After this, we should be good to go, just type in the elastic IP into the browser again, followed by “/admin/index.php” for the admin interface, and “pickup/keypad.php” for the kiosk interface.
    - Updating File Permissions for QRcode Generation
      1. Because of the odd permissions of the Apache server, a QRcode can’t be saved to a file from php until the permissions are updated
      2. To find which user needs to be assigned to the “admin/images” file, we type into the terminal:  
         ps aux | egrep '(apache|httpd)'
      3. It seems clear that we should change the ownership to daemon, so:  
         sudo chown -R daemon:daemon /opt/lampp/htdocs/admin/images
      4. Now, the website should be fully functional

**Locker Kiosk:**

* + Configuring Raspberry Pi 3 Model B+
    - Flash the latest version of Raspberry Pi OS with desktop on a 32GB micro-SD
      1. Download the Raspberry Pi imager directly from their website.
      2. When choosing the OS, choose Raspberry Pi OS with desktop.
      3. Choose the SD that you want to flash the OS on.
      4. Install Micro-SD card in Raspberry Pi micro-SD slot.
    - Installing Raspberry Pi Camera Module rev 1.3
      1. Open the terminal and type the following: ‘sudo raspi-config'. From there go to interface options and set camera to enabled. Following should be a reboot.
      2. Locate the CSI port on the Raspberry Pi located between the HDMI and ethernet; not to be confused with the display port located above the micro-SD slot, and insert one end of the 15-pin flex cable while inserting the other end to the camera module.
    - Installing Raspberry Pi Display
      1. Locate the display port on the Raspberry Pi located above the micro-SD card slot; not to be confused with the CSI port located between the HDMI and ethernet and insert one end of the 15-pin flex cable while inserting the other end to the display driver board.
      2. The display driver board itself requires power, so take two jumper cables and insert them into the 5V and GND pins of the driver board. Then connect those to the corresponding 5V and GND pins on the Raspberry Pi (5V is on pin 2 and GND in on pin 6).
    - Installing Python3
      1. Installing Python3
         * Open the terminal and run the following commands:
         * Sudo apt update
         * Sudo apt install python3 idle3
      2. Test Python3 installation
         * Open the terminal and type ‘Python3 --version’. This should output the version of Python3 that is currently installed.
    - Downloading Chromium Web Driver
      1. Download Chromium Web Driver
         * First check the version of chromium that is installed.
         * Download the version of chromium web driver that is meant for the version of Chromium currently installed at: <https://sites.google.com/a/chromium.org/chromedriver/downloads>
      2. Test Chromium Web Driver Download
         * Below is a sample script that should test the web driver by launching a webpage, loading google.com, then exiting after 5 seconds.  
           
    - Installing RPi.GPIO
      1. Installing RPi.GPIO
         * Open the terminal and run the following command:

Sudo pip install RPi.GPIO

* + - 1. Test RPi.GPIO installation
         * By opening the terminal and entering ‘sudo python3’, Python’s interactive line should appear. Proceed to type ‘import RPi.GPIO as GPIO’ and then GPIO.VERSION. This should output to the terminal the currently installed version of RPi.GPIO.
    - Installing CV2
      1. Installing CV2
         * Open the terminal and run the following command:

sudo apt-get install python3-opencv

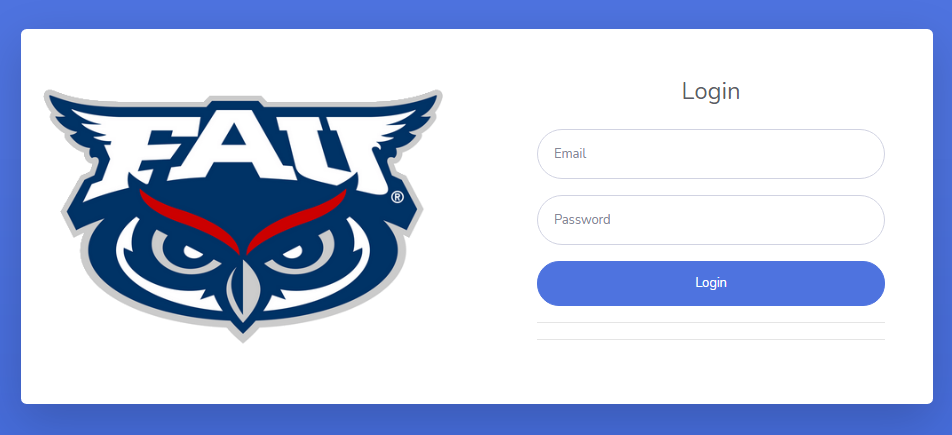
sudo apt-get install libqt4-test python3-sip python3-pyqt5 libqtgui4 libjasper-dev libatlas-base-dev -y

pip3 install opencv-contrib-python==4.1.0.25

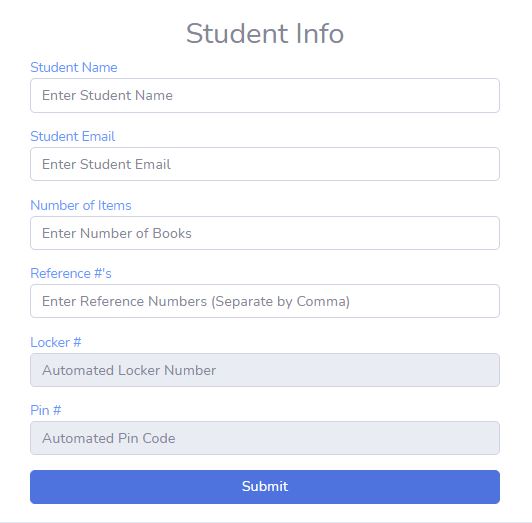
* + - 1. Test CV2
         * By opening the terminal and entering ‘sudo python3’, Python’s interactive line should appear. Proceed to type ‘import cv2’. If everything was installed correctly, the next line should appear. If an error occurs, then the installation was not successful.
    - Connecting Relay
      1. Our system only has support for up to 18 lockers. When initializing the library RPi.GPIO, it is done so using GPIO.setmode(GPIO.BOARD). Meaning the output pins are referenced by the physical pin number. Follow the list below when attaching the relay to the RPi because the index number corresponds to the locker number:   
           
         [15,16,18,19,21,22,23,24,26,29,31,32,33,35,36,37,38,40]  
           
         For example, index 0 (pin 15) corresponds to locker 1 and index 17 (pin 40) correspond to locker 18. It is important to strictly follow this list. Not doing so may result in some lockers not opening.
    - Testing connections
      1. To test the build, execute browser.py. In doing so, this will test the install of all the individual dependencies and packages as well as the connections to the relay. When finished loading and one sees the keypad displayed, proceed to enter a valid pin number/qr code to send a signal to the relay, which in turns opens the locker.
    - Connecting Solenoids and latch
      1. The solenoids should be placed inside of the locker rather than on the door to prevent the need for slack of the wire, also it is the most discreet look.
      2. The locks also must be fitted to a latch this way the door can be held closed when the lock is not active.
      3. The locks should be positioned so that when the door closes the latch will push against the slanted angle of the solenoid, depressing it and sliding behind it.
      4. Adhere with strong Epoxy
    - Wiring Locker
      1. The solenoids will need to be connected to power so they must be soldered to longer wire which is then fed through the system in the most discreet way as possible.
      2. Cover visible wires with tubing or electrical tape to hide and secure them in place.
      3. Connect the wire to the proper relay terminals as well as the power supply.
      4. Connect wire to PCB and feed to the bottom of the locker to allow the plug in to be discrete and low so there is the most length for the AC cord.
    - Test Solenoids
      1. Test each locker by powering one relay at a time to make sure the correct door opens when required.

**Admin Interface:**

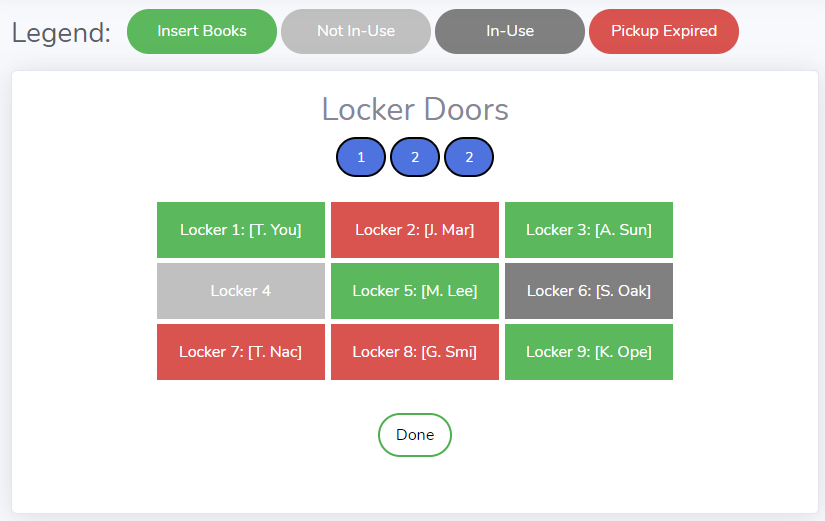
* + Our administrative web page consists of:
    - 1. Login Page



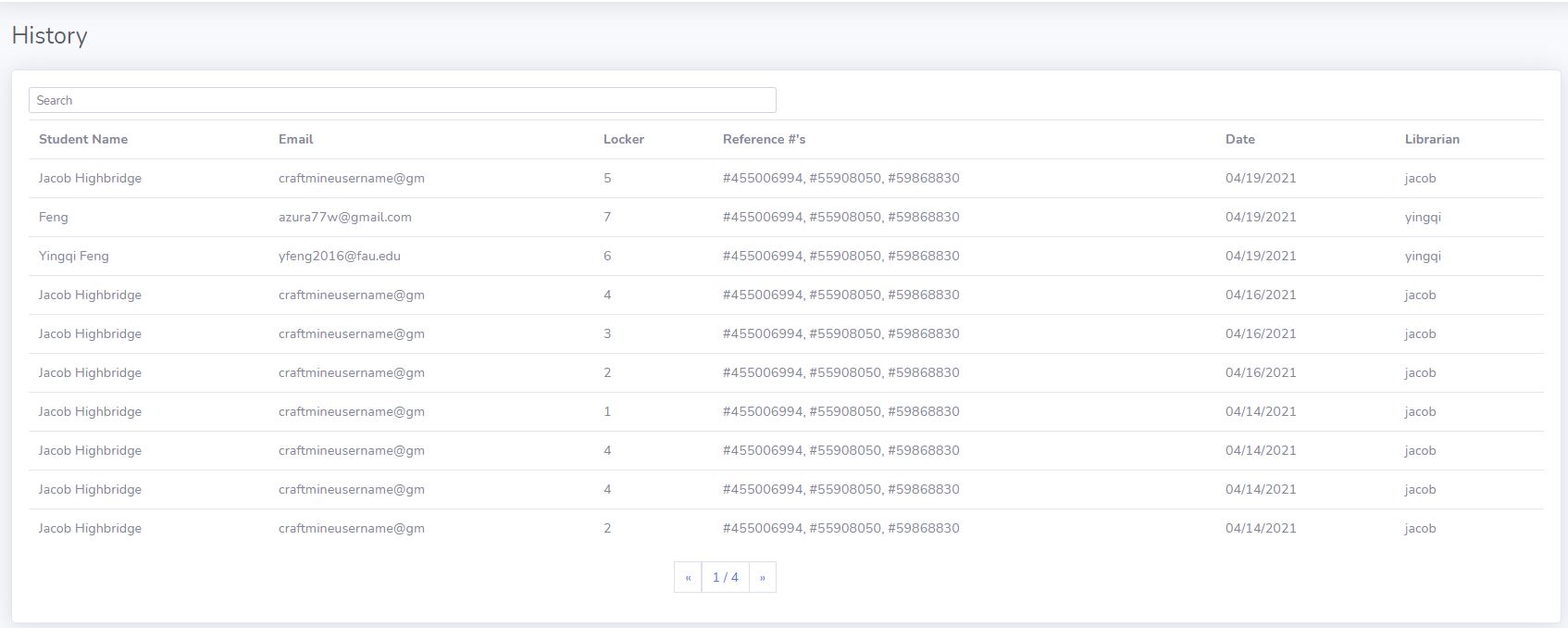
* + - 1. Student Form Page



* + - 1. Locker Access Page



* + - 1. History Page



* + Since it is a web page, the majority of the code, both front end and back, consists of HTML/CSS, PHP and Javascript. In order to access and edit the source code for the web page a Language compatible IDE such as Brackets or Visual Studios is required.

In terms of Visual Studios:

* + - 1. Make sure you have the Admin Folder downloaded and located somewhere you can access.
      2. Open Visual Studios.
      3. Press Ctrl + O and then find the Admin Folder and highlight all the files that you want to open to edit.
      4. Once the files are open, changes can be made to any HTML/CSS/PHP/Javascript code.