Group 4

Systems Specifications

Secure Access Lock System with NFC and Biometrics

California State University, San Bernardino

March 14, 2025 (Spring 2025)

CSE 5408-01

Shunsuke Akiya, Christopher Cao, Rafael Ceja,

Matthew Uy

1. **The Concept**

The Secure Access Locking System with NFC and Biometrics system provides users a simple and intuitive way to protect their valuables using a fingerprint sensor, NFC reader, physical key switch, and LCD screen. All of these components will communicate with the Raspberry Pi, web application, and PCB. When the user uses one of the above methods, a solenoid acting as a lock will unlock for the user with the LCD screen and web application informing the user the lock has successfully unlocked. After a brief period of inactivity, the solenoid will lock itself with the LCD screen and web application updating accordingly. The problem this device solves is users protecting their valuables and adding further security to doors. Users can rest easy knowing all methods to unlock will always be on their person in the form of their smartphone and do not have to solely rely on a physical key. Should the fingerprint sensor or NFC reader fail, users can still use the physical key switch as a failsafe or if they prefer using a more traditional unlocking method. The primary function is to improve the quality of life and security of all users as they go about their daily routine no longer worrying about security.

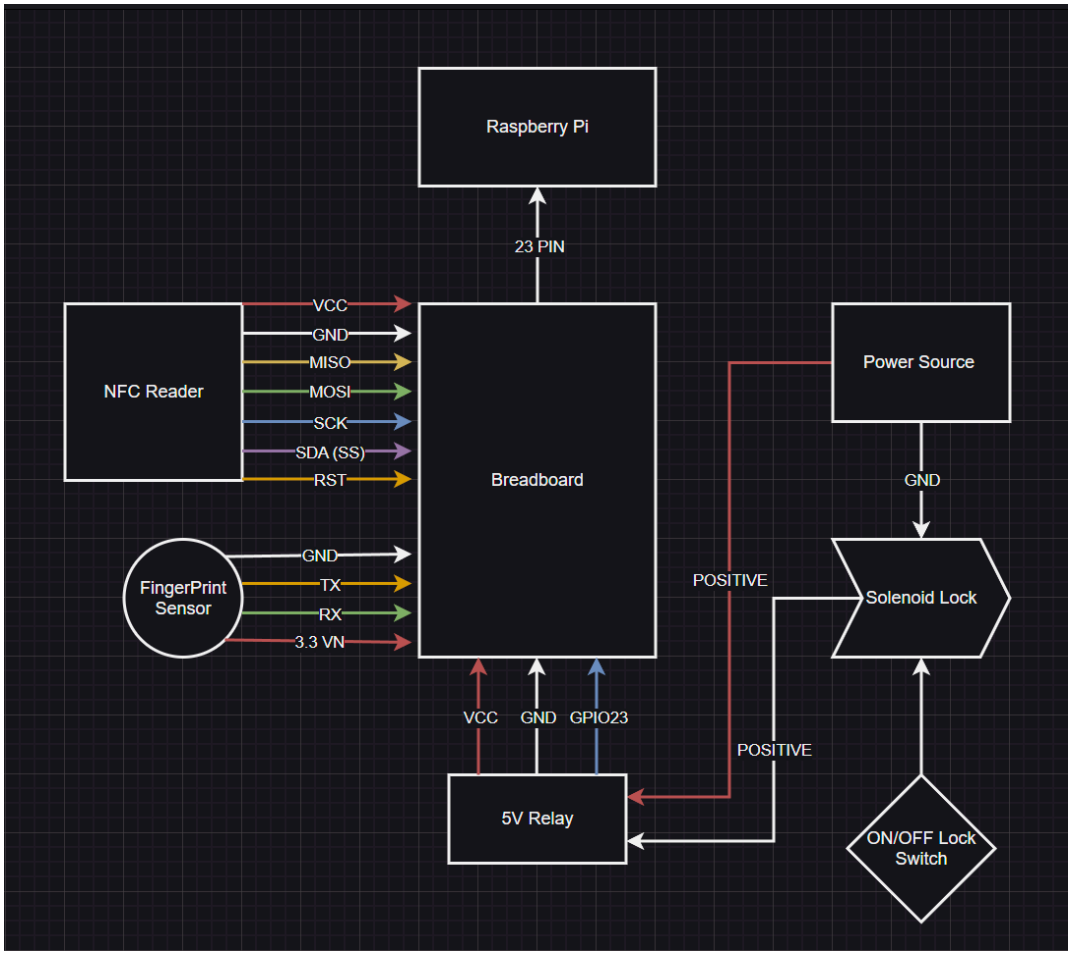
Change(s) from Requirements Specification:

- Revised PCB design

2. **Inputs/Outputs and System Block Design**

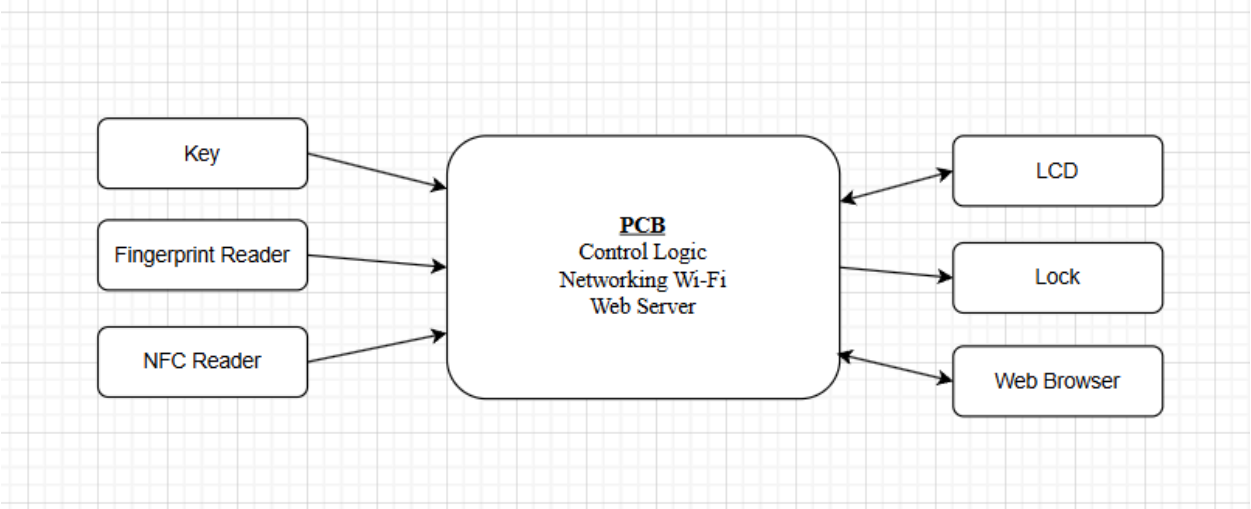
Block Diagram:

OLD BLOCK DIAGRAM:



Inputs/Outputs:

- Raspberry Pi:
  - Inputs: Obtains data from fingerprint sensor, NFC reader, physical keyhole, and HTTP data from web app
  - Outputs: Signal sent to relay to unlock solenoid; feedback given to web app
- Fingerprint Sensor:
  - Inputs: Fingerprint from user, power from Raspberry Pi
  - Outputs: Unique ID assigned to fingerprint to unlock solenoid, sent to Raspberry Pi
- NFC Reader:
  - Inputs: Tag and card, power from Raspberry Pi
  - Outputs: Unique IDs assigned to tag and card to unlock solenoid, sent to Raspberry Pi
- Physical Keyhole:
  - Inputs: Physical key
  - Outputs: Solenoid unlocked manually by user with physical key
- Web App:
  - Inputs: HTTP data status codes on whether sent data successful or not
  - Outputs: Commands sent to Raspberry Pi to accomplish various tasks such as adding a new Wi-Fi connection to Raspberry Pi, pressing a button to unlock and lock the solenoid, etc
- Solenoid Lock:
  - Inputs: Power from relay to function to unlock/lock as needed
  - Outputs: Unlocking and locking
- 3 AA Batteries:
  - Inputs: N/A
  - Outputs: Provides power for solenoid and Raspberry Pi as needed
- PCB (NEW bLOCK DIAGRAM):



- Key → Physical key
- Fingerprint Reader → Sends data to PCB control logic
- NFC Reader → Reads the NFC into the control logic

- PCB: Control Logic, Networking Wi-Fi, and Web Server
    - Control Logic = Informs the lock about status
    - Networking Wi-Fi = communicates together with the lock and the user commands
    - Web server = Allows the user to change settings, holds authentication data, and fetches data with the flask server. Communicates with the Web Browser.
- → LCD → Activates from PCB web server or control logic to activate, can read through both systems
- → Lock: Control Logic and web server will have access for the status of the lock
- → Web Browser → Will communicate with the Networking Wi-Fi and read status activation of the lock, web server can display options and activation. Will send information back to the web browser.

3. <u>Specification of the Blocks</u>

<u>Functionality:</u>

- Raspberry Pi:
    - The Raspberry Pi is the brain of the entire project. All of the programming is done on it through the Thonny IDE and utilizes the Raspberry Pi OS (based on Debian Buster). All of the circuitry and wiring is also done on it, specifically by wiring the components to the 40 Pin GPIO header.
- AA Batteries:
    - The solenoid will utilize about 6 AA batteries, which is used to power the solenoid. The solenoid and batteries will be connected to the Raspberry Pi and a relay, allowing for unlocking and locking capabilities through enabling and disabling the relay.
- Relay:
    - As mentioned in the previous point, the relay's primary use is to allow arbitrary components to be enabled and disabled, in this case the 12V solenoid. By sending an input to the relay through programming (the corresponding GPIO pin that it's plugged into), it enables the solenoid to contract. To this, we can add some base conditions such as "if the fingerprint or NFC reader is true, then send an input to port 5. Else do nothing."
- 12V Solenoid:
    - Acts as a door lock that is unlocked using either the fingerprint sensor, NFC reader or with a physical key hole. It will be controlled using a relay.
- NFC Reader:
    - One of the three ways that will unlock the door. Within the NRC reader, there will be 2 ways to unlock it:
        - Pre Attached NFC Tag
        - Pre Attached NFC Card
    - The UID of the tag and card will already be installed in the lock, so the user will not have to set it up. The only setup that is required is for the phone.
- 5V Fingerprint Sensor:
    - The 2nd of the 4 ways that will unlock the door. The fingerprint sensor can register multiple fingerprints, and will allow the user to unlock the door.

The user's thumb will be used as it is the most comfortable and logical way for the reader. If a registered user scans their fingerprint, the LED around the sensor will light up green, indicating a successful unlock. If not, it will light up red.

- Key Lock:
    - Failsafe used to unlock the door should the NFC reader and/or fingerprint sensor fail.

Technical Specifications:

- Power requirement:
    - As of now, we discovered that a 9V battery was not enough to power the solenoid, so we switched to using AA batteries. After watching some videos with a very similar solenoid setup, we agreed on using 6 AA batteries.
- Communication Protocol:
    - The physical circuitry will be using the NFC protocol for the reader, and for the backend side, we will be using the HTTP protocol for fetching and requesting data.
        - In JavaScript, the fetch() function is used to send and request data. In this case, we will be sending data from the web app to the Raspberry Pi for setups such as adding and removing a fingerprint so that the sensor can store it in memory, pressing a button to lock and unlock the solenoid, adding a new Wi-fi connection for the Raspberry Pi, and adding a new phone for the NFC reader.

Inputs/Outputs:

- Raspberry Pi:
    - Inputs:
        - Data from the Fingerprint sensor
        - Data from the NFC reader
        - Input from the Physical keyhole
        - HTTP data sent from the Web app
    - Outputs:
        - Control signal to the relay for enabling/disabling the solenoid lock
        - Feedback to the Web app
    - Interactions:
        - Receives data from the Fingerprint sensor and NFC reader to validate access
        - Sends signal to the relay to unlock the solenoid
        - Communication with the Web app for remote control and other setups
- Fingerprint Sensor:
    - Inputs:
        - Power from the Raspberry Pi
        - Finger placed on sensor
    - Outputs:
        - Fingerprint data sent to the Raspberry Pi
    - Interactions:

        - Sends data to the Raspberry Pi to determine if the fingerprint is recognized or not
- NFC Reader:
    - Inputs:
        - Power from the Raspberry Pi
        - NFC card/tag placed near the reader
    - Outputs:
        - NFC data sent to the Raspberry Pi
    - Interactions:
        - Sends NFC tag data to the Raspberry Pi, which determines if the tag/card is recognized or not
- Physical Keyhole
    - Inputs:
        - Physical key
    - Outputs:
        - Unlocks the solenoid
- Web App
    - Inputs:
        - HTTP status codes on whether the sent data was successful or unsuccessful.
    - Outputs:
        - Sent fetch() commands to the Raspberry Pi for tasks such as adding and removing a fingerprint so that the sensor can store it in memory, pressing a button to lock and unlock the solenoid, adding a new Wi-fi connection for the Raspberry Pi, and adding a new phone for the NFC reader.
    - Interactions:
        - Allows remote management and access of the the door lock
- Solenoid Lock:
    - Inputs:
        - Power from the relay to enable/disable the lock
    - Outputs:
        - Physical locking/unlocking
    - Interactions:
        - Enabling or disabling of the solenoid when relay is activated by the Raspberry Pi
- 6 AA Batteries:
    - Outputs:
        - Power for the system
    - Interactions:
        - Powers the solenoid and/or the Raspberry Pi

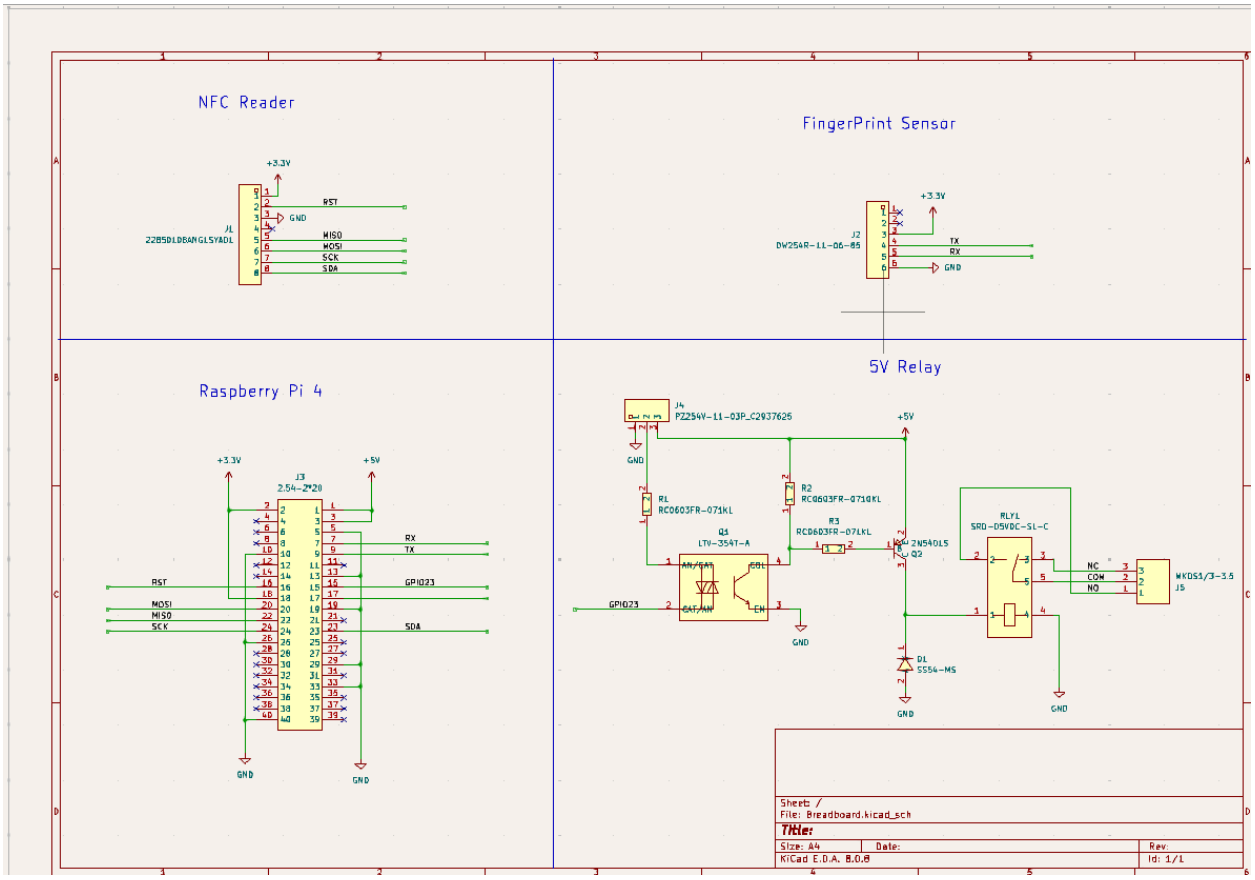
4. **System Description**

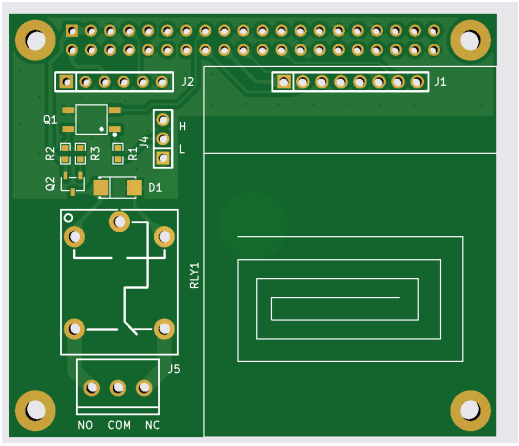
Block Interaction:

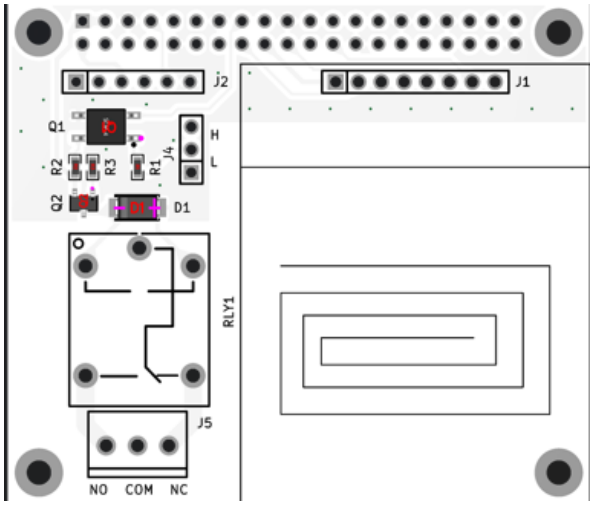
- Raspberry Pi:
    - Communicates with fingerprint sensor and NFC reader to confirm access
    - Once access confirmed, signal sent to relay to unlock solenoid

    - Remote control by communicating with web app
- Fingerprint Sensor:
    - Fingerprint sent to Raspberry Pi whether received fingerprint is recognized
    - 1 of 3 methods to unlock solenoid
- NFC Reader:
    - Tag and/or card data sent to Raspberry Pi to determine if they are recognized
    - 1 of 3 methods to unlock solenoid
- Physical Keyhole:
    - Unlocks solenoid through physical key
    - 1 of 3 methods to unlock solenoid
- Web App:
    - Connects with solenoid for remote access
- Solenoid Lock:
    - Unlocks/locks when relay activated by Raspberry Pi
    - Powered by 6 AA batteries
- 6 AA Batteries:
    - Powers solenoid lock

Detailed Schematics and PCB Design:







5. System Analysis

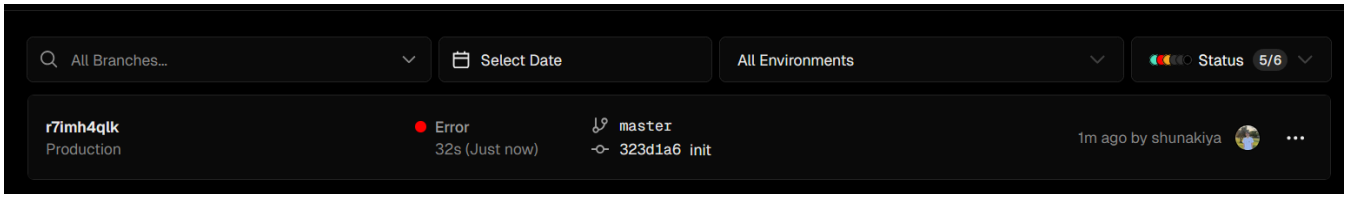
Testing and Results (Figures, Tables, Graphs):

Testing with Solenoid Lock	**Result**
Voltage: 1 - 9v Battery	Fail
Voltage: 9 - 1.5v Battery (AA Battery)	Success
Multiple Testing for Fingerprint/NFC	Fail →Success
Solenoid Activation with Code	Fail

OLD TABLE:

NEW TABLE:

WEB + PCB Testing	**Result**
Failed deployment because code issue	404 ERROR
Flask server not hosting correctly	Fail
Server status	Success
PCB creation	Success, but need new iteration
Website	Fail → Success
Door handle	Configuration Problems (Fail)



Error message ^

Analysis of Key Metrics:

- Performance
  - System performs as expected according to voltage. Solenoid receives power from the battery which then allows unlocking/locking. However, there were problems in achieving the power needed for solenoid.
  - Website is working well, we are waiting for the door handle and PCB where we can understand how we can connect all pieces together
- Accuracy
  - Device and commands are accurate, Raspberry Pi is able to communicate with the breadboard which then can read the NFC reader or Fingerprint Sensor that can activate the solenoid lock.
  - Website is now reading accurately and is able to hold user values and their data
- Efficiency
  - Device is efficient, uses as minimal power necessary to activate solenoid lock. Although lower power/power source may deteriorate the system because it will draw a higher current, the cost of equipment like batteries are easily replaceable and items are cost efficient.
  - Website is efficient and able to load properly while storing data. Efficient while using security measures and IP address.

Problem Areas:

- Singular 9V battery not powerful enough to power solenoid
  - Replaced with 3 AA batteries housed in casing
- One AA battery exploded during testing
  - Immediately replaced
- Website not working
  - Problem with the host server not connecting
- Door handle problems
  - Problem with finding a door handle that can properly fit the things we want to do
    - Solution: Looked through amazon and found door handle with lock implemented into it
- PCB
  - Creation of PCB is both time consuming and resource consuming
    - We have started early in order to counter these extensive issues, the board has come earlier and we have started to work on it earlier to order another version