Tianxing Wu

Smart turnstile system

IoT project report

IoT project

2020



CONTENTS

[1 INTRODUCTION 3](#_Toc59557757)

[2 background 4](#_Toc59557758)

[3 implementation 5](#_Toc59557759)

[3.1 Software components 7](#_Toc59557760)

[3.1.1 Database 7](#_Toc59557761)

[3.1.2 Server 8](#_Toc59557762)

[3.1.3 Booking 11](#_Toc59557763)

[3.1.4 Admin tool 12](#_Toc59557764)

[3.1.5 Turnstile 14](#_Toc59557765)

[4 Discussion and conclusion 16](#_Toc59557766)

[References 17](#_Toc59557767)

# INTRODUCTION

The **Internet of things** (**IoT**) describes the network of physical objects— “things”—that are embedded with sensors, software, and other technologies for the purpose of connecting and exchanging data with other devices and systems over the Internet. (1)

IoT is a lightweight solution for collecting, transferring data, by connecting physical objects and make them identifiable, readable, controllable, addressable, and locatable via the Internet (2). By enabling communication for electronic, sensors can therefore interact with other online components in real time. For example, with an addressable IP, an electronic component could be manageable via internet or interact with web server using some APIs provided. Such features IoT system to automate the process of managing and monitoring.

As the covid-19 pandemic is raging through the whole world, IoT enabled technology do helps to prevent further spreading of the virus in a certain extent. NFC is a contact-free data transmission protocols, which provides low speed connection within range of 4 cm (4). This technology is wildly adopted in transportation (travel card), contactless payment.

At the onset of COVID-19, we saw how contactless solutions transitioned from a nice-to-have to a necessity, Stated by Rene Rumberg. (5) On the other hand, indoor environments act perfectly as natural incubators for virus. Therefore, having a system using contactless method to control visitor access, monitoring flow is beneficial for both slowing down the virus spreading and managing visitors.

In this work, an online booking system, a turnstile and a management tool are prototyped, using mainly flask, raspberry Pi and Arduino. The online booking service accepts visitor’s appointment based on the flow estimate. Upon booking, an NFC tag will be registered then passed to the visitors. Visitors can use the tag to enter the premises at the time booked. Every single attempt to enter is logged with temperature measured and photo to ensure the authenticity and the person presenting is not sick (fever).

The NFC tag alone is nearly as cheap as traditional keys but can be contactless. Logging access is also made much easier. Furthermore, the NFC tag can act like switches to activate certain services (lighting, electricity) with a simple swipe.

Note that although the project name is smart **turnstile**, it can also be gate, room door or something else.

This system can be installed to premises where access control is one of necessities. Such places can be gym—the tag could be valid for a month, year or any period defined by the administrator; hotel—photo and temperature are taken at the gate which can be opened with any customer’s tags, but each room door has its corresponding tag; or even metro station which can have temperature measuring feature from this system added compared to the existing one. However, such system could only be useful when visitor could wish leave and reenter a building at will in the middle of the time booked, and this place must not be home unless the temperature is only logged as a reminder but certainly not a key condition for permission.

This report mainly serves as a documentation of how the system is implemented with theory background explained. The rest of this report is divided in to three main parts: background section which presents the tools used. Implementation section where implementation details are explained, and most important results are presented. And lastly the discussion of limitation of the system, comparison with earlier research.

# background

*Server*

In this work, windows server 2016 by azure is used, because the project is tested on windows system when developing. Python flask is the main framework to build the web application as it is featured in rapid prototyping and lightweight. However, it also means that it does not come with any tools providing database abstraction, or other common functions. (6)

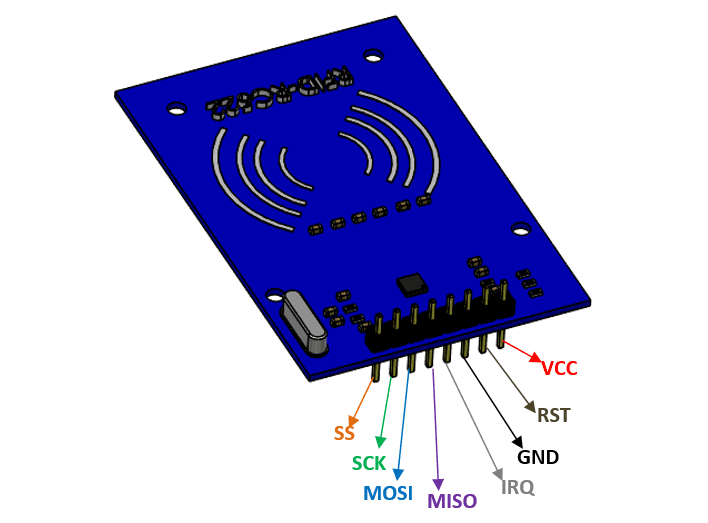
To provide an extra abstraction between database and the web application, pyodbc is then used. The pyodbc modules utilize ODBC interface provided by Microsoft. (7)

*Database*

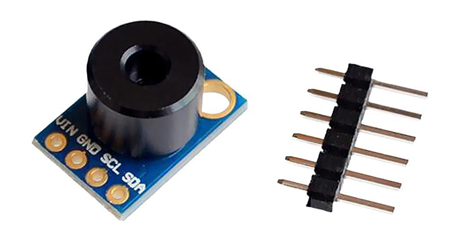
Database used here is the SQL DB provided by Azure. It allows pay-a-you-go payment method which is suitable for prototyping.

*Raspberry Pi*

There are several sensors used in this work but only the most important are introduced.



The **RC522** is a **13.56MHz RFID module** that is based on the **MFRC522 controller from NXP semiconductors**. The module supports I2C, SPI and UART and normally is shipped with a RFID card and key fob. It is commonly used in attendance systems and other person/object identification application. (8,9)



GY-906-BCC is an infrared light thermometer which features in 10 cm range measurement of target, ambient temperature through I2C interface.

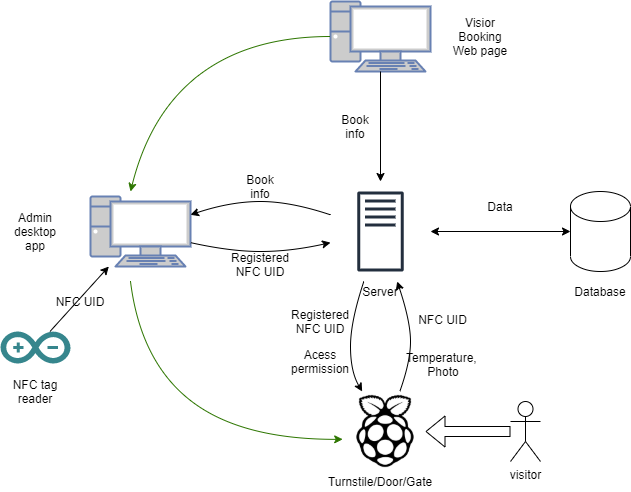
Design of the turnstile is rather simple. (10)

*Admin tool*

To construct the admin tool which can communicate to the server via internet, Tkinter is adopted as it is a python tool for building desktop app and it is the tool the writer is most familiar with. (11)

# implementation

The smart turnstile system consists of a server, and several clients which interact with it in real time.



*Server*

The centralized server hosts Restful APIs using http protocol to serve clients’ needs. It functions as an endpoint for management, visitor, booking which also allows interaction with the database via APIs.

*Database*

The Database communicates with server to store permanent data.

*Booking website*

Visitors could book timeslots prior to their visiting with the booking website provided by the server. It validates and saves information entered by user and the time booked. When validation is complete, the server could save the booking entry to the database.

*Admin desktop app*

For each validated booking, there is a corresponding NFC tag to be assigned. Initially, the booking will be marked as not assigned until the administrator register a NFC card for the booking entry. The management app for admin collects all the unassigned booking by communicating to the server via APIs. When assigning the card, the administrator will be prompted for authentication. The app will transmit the auth info along with the NFC UID read from serial to the server. If the auth matches, and UID is valid, the card will be registered for used and data is saved. Each assigned card corresponds to one booking entry. The server actively searches for any duplicate card used in the database when the API is called.

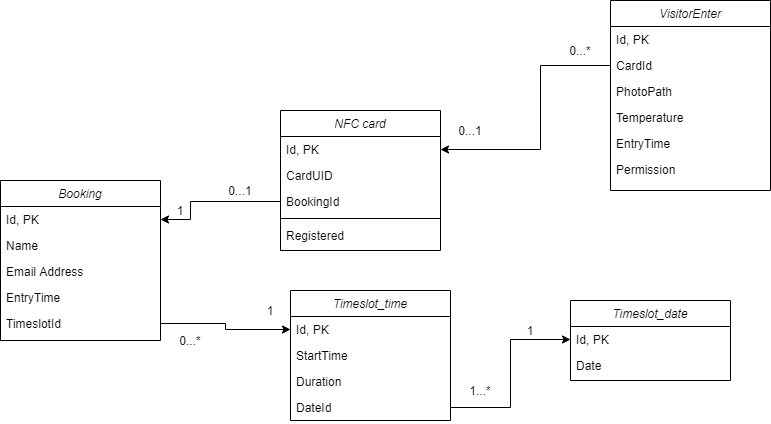
*Turnstile*

To pass the turnstile, the visitor needs to provide a valid NFC tag. The turnstile will read the tag and verify by talking to certain API on server. If card is valid, the turnstile will precede to measure visitor’s body temperature and take a photo. The temperature indicates whether the person is sick (fever) or not. If a permission is passed, the door/Turnstile will allow to be passed.

## Software components

### Database

In this work, SQL database is used.



*Timeslot date/time*

The date and time of timeslots available are saved separately and linked by a one-to-many foreign key. The purpose of doing so is to provide tree like structure when selecting them. One date could have multiple timeslots open, but one timeslot time must have one and only one date.

*Booking*

The booking table stores information of a person’s name, email address, and an entry time column which has datetime data about when the booking is made. Lastly there is a timeslotId column. This column has the timeslot\_time Id, since the relationship between time and date is a many-to-one relationship, each timeslot\_time must have one corresponding date. Therefore, by only refering timeslot\_time, the booking will have enough information about both booked date and time.

*NFC card*

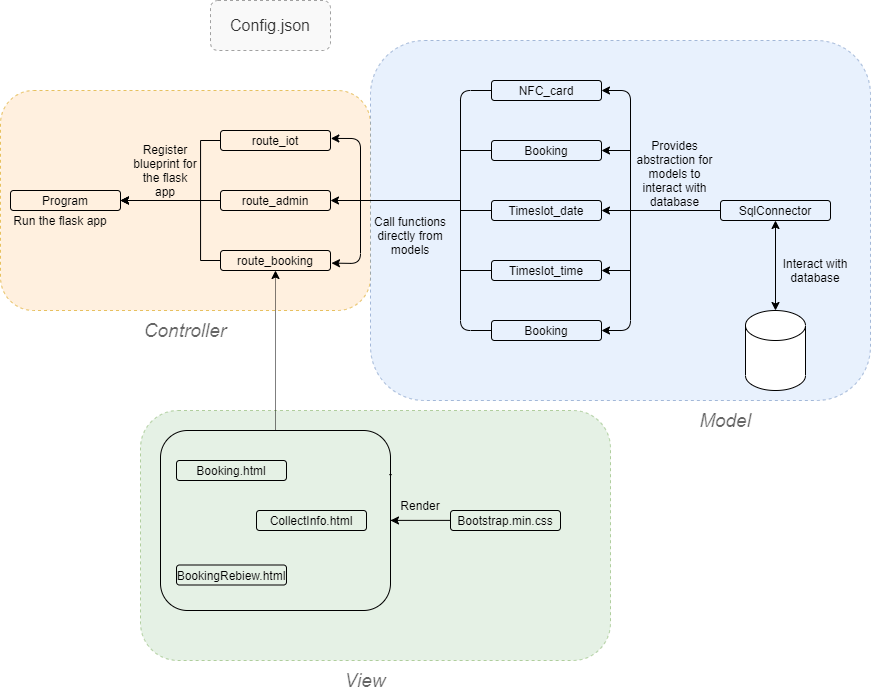
Each booking will have 1 NFC card assigned. An NFC card row is created along with a booking. By default, the card UID is null with Registered = 0 (not assigned). One booking is said to be processed only when the Card UID is assigned to the corresponding card row. With UID assigned, the Registered column will also be set to 1 (assigned).

*Visitor Enter*

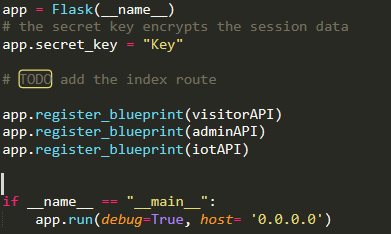
This table logs all the attempts to pass the turnstile regardless of success or fail. The cardId column stores the card used if there is any valid one. Normally, if there is no matching card in the card table, the card Id column will be empty and permission = 0. If there is a matching card in the DB but not valid (expired/not within time) the card Id will be logged but permission will be 0.

The photo taken by the turnstile will be transmitted to the server with HTTP POST request. The photo itself is saved in the file system of the server to optimize performance, but the path is logged to the DB along with the temperature of the visitor.

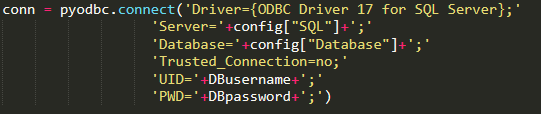
### Server

The web server is construct in a MVC manner using flask. 

The program.py is the main program. It initializes a Flask object with all routes registered using flask blueprint from route\_iot, route\_admin, route\_booking.



In this work, a SqlConnector class is created to add abstraction between the web app and SQL database. The connector initialize connection to database using pyodbc along with username and password pair, server location, database name from config.json file.

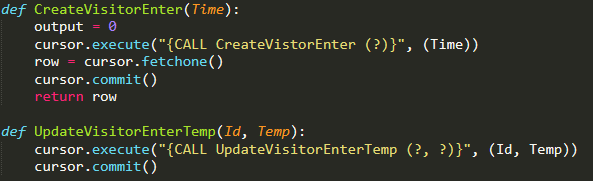


*SqlConnector.py*



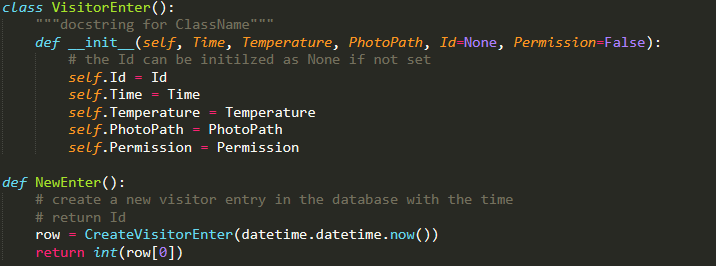
*Config.json*

For each method in the SqlConnector class, it calls predefined stored procedures in the database, if data is modified in the database, the changes will be committed. Otherwise, the raw data returned is captured and return.



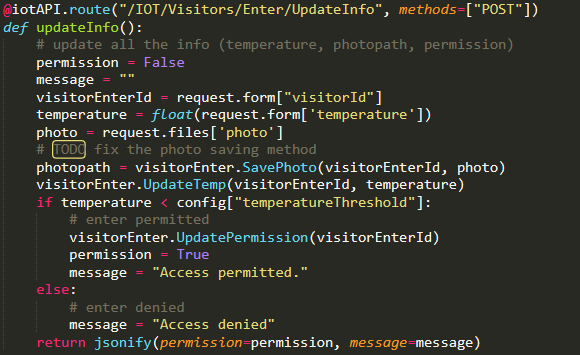
There are in total 5 model python files in this prototype. Each model file has a class to instantiate each model along with numbers of static methods. Those static methods call functions from SqlConnector to achieve certain functionalities. Each model file is aimed to complete all the task related to a model. For example, the visitor enter model contains static method to: create a new visitor enter entry in the database and return the scope Id; update card Id to the entry if there is any.

Unlike methods in SqlConnector which returns raw data, the model methods will have all data processed before return, which means the methods are more specific to one single problem. A model method could use several SqlConnector methods to achieve certain goals, whereas the SqlConnector method merely provide an abstraction to call stored procedures.



These model methods are then used to construct routes. Each route in the flask app provides an endpoint URL to interact.

The routes process incoming request with model methods. sometimes multiple methods are combined to be functional. In below example. The server examines the temperature to grand permission. Several database columns are involved in this case.

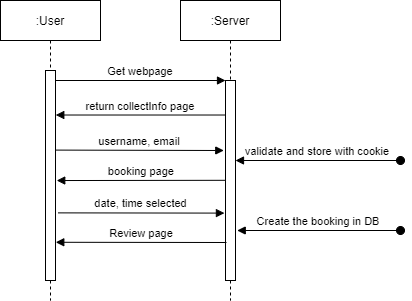


For route\_visitor specifically, there are some html files serves as interactive webpage for booking purpose, and a bootstrap.min.css to render them.



### Booking

The sequence diagram below demonstrates how a booking is made from user.

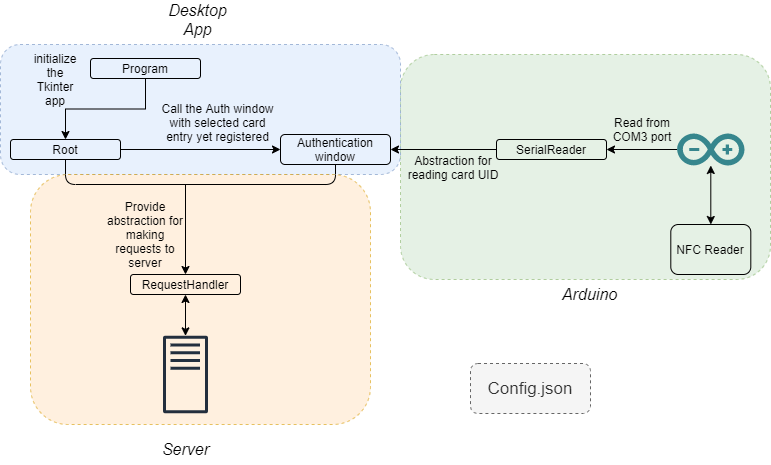


Initially, the user will request a CollectInfo page using HTTP GET. When the users have filled in the information and returned with POST, the username and email will be saved to users’ web browser using cookie. Then users will be redirected to the booking page. Note that, if a user tries to go to the booking page without filling in the email and username, it will be directed back.

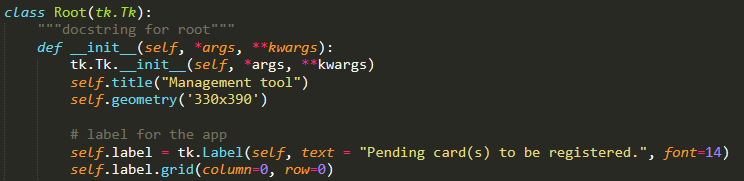
Lastly, by select the date and time from the booking page, the booking will be complete. The entry will be saved in database, and users will be redirected to the review page of the booking.

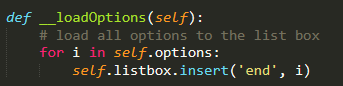
### Admin tool

Admin tool is a desktop application which register the NFC card UID for each booking entry.

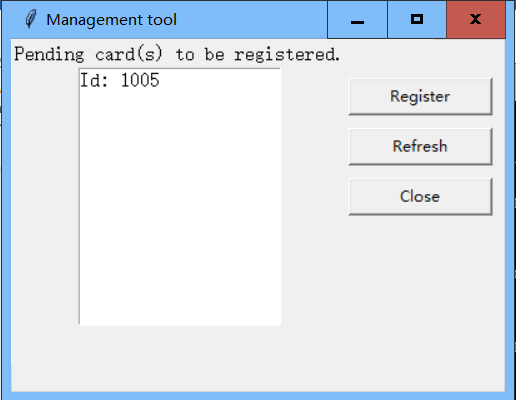


Same as from the web app, the program class in the admin tool is also used to trigger the application. Inside program class, a Tk() of root is created before entering mainloop.

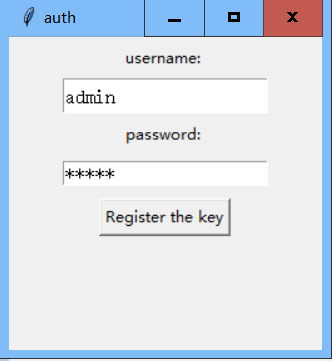




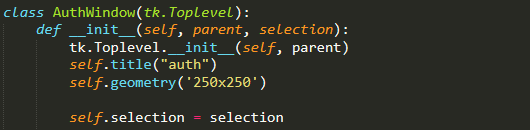
This application typically has two windows. One is root which present all the unassigned card objects with their Ids.



And another one is authentication window which send HTTP request to remote server with HTTP basic Auth and NFC UID.



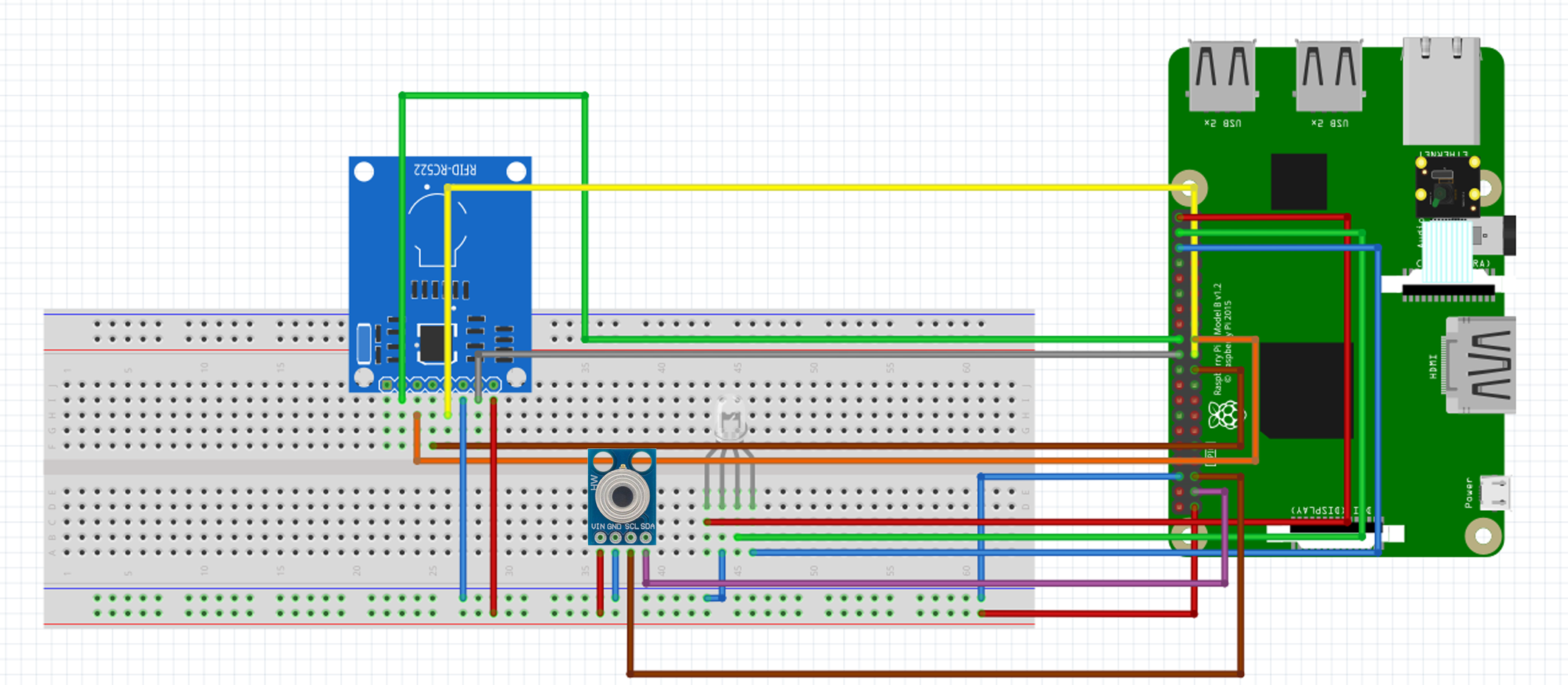
The authentication window class is initialized from within the root class when the register button is pressed, and a selection is made. The selection in root window is one of the parameters required from authentication window’s constructor. Therefore, changing the selection will not affect the functioning of this window.



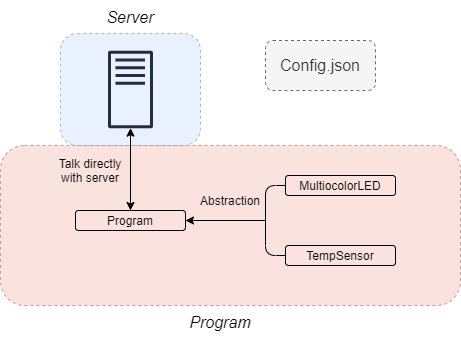
The admin tool highly relies on the server. It requires to get all the unassigned card object, register the key. Therefore, a new class named RequestHandler is added to send request to server and format the responds. It sends HTTP request to the server and return data in a formatted manner.

When registering the key, authentication window will read the COM port output where Arduino is connected using method provided in SerialReader class. The method will scan the com port for output. When card UID is read, the method will return. Then if the username-password pair matches from the server and UID is in correct format, The UID will be registered to use by visitors.

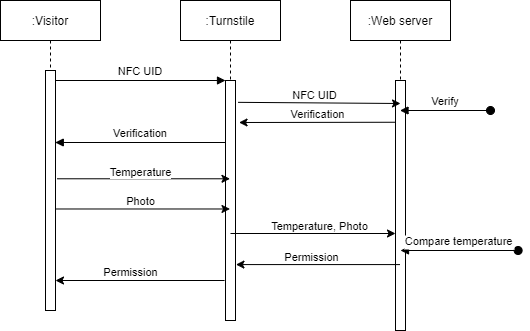
### Turnstile



From hardware’s point of view, the turnstile is constructed with raspberry Pi. It consists of one NFC reader (RC-522), one IR temperature sensor (gy-906-bcc), a cathode multicolor led and Pi camera.



Program.py in this part works as a script. It utilizes function provided by MulticolorLED and TempSensor. Since the script is much simpler compared to other part of the project, it communicates with the server directly.



When entering the turnstile, the turnstile will first ask visitor for the NFC card. As card is swiped, the UID will be compared with entries in the database. If there is a match, the server will further compare the time booked with current time. With all that complete, a verification message will be received by the turnstile. It determines whether the turnstile will precede to measure temperature or start over.

Only if the UID is valid, turnstile will scan the temperature as well as take the visitor’s photo. After that, the temperature is compared with the threshold value set by the web server. When the final verification is passed, turnstile shall allow the visitor to pass.

# Discussion and conclusion

The main result of this work is an online service verify user’s identity with the NFC UID along with temperature. All the process of this system is designed to be as contactless as possible. This system could implement authentication with some policy and flow control at entrance of premises. This system could slow down the spreading of covid-19 in a certain extent.

Since this is just a prototype, there are several limitations of the implementation. First, security. For example, as the admin tool fetches all the non-assigned card entries in the database, there should be some security features to ensure integrity, authenticity, and confidentiality of the data. Some of other services also needs be protected. One possible solution could be using public key. Secondly, privacy, taking photos of visitor may cause some privacy issues, as face recognition technology is wildly adopted, some may collect other people’s face data to impersonate.

References

1. <https://en.wikipedia.org/wiki/Internet_of_things>
2. <https://learn.xamk.fi/pluginfile.php/590888/mod_resource/content/0/A%20Novel%20Secure%20IoT-Based%20Smart%20Home%20Automation.pdf>
3. <https://media.ccc.de/v/30C3_-_5479_-_en_-_saal_6_-_201312291215_-_building_a_safe_nfc_ticketing_system_-_bughardy_-_eagle1753#t=432>
4. [Near-field communication - Wikipedia](https://en.wikipedia.org/wiki/Near-field_communication)
5. [Smart contactless solutions helping tackle COVID-19 and save the environment — e-Estonia (e-estonia.com)](https://e-estonia.com/smart-contactless-solutions-helping-tackle-covid-19-and-save-the-environment/)
6. <https://en.wikipedia.org/wiki/Flask_(web_framework)>
7. https://pypi.org/project/pyodbc/
8. <https://components101.com/wireless/rc522-rfid-module>
9. https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf
10. <https://www.amazon.co.uk/GY-906-BCC-MLX90614ESF-BCC-Temperature-Compensation-Measurement/dp/B07SQZKYM7>
11. https://docs.python.org/3/library/tkinter.html