**SYSC 3010- Sunny Project Design Components**

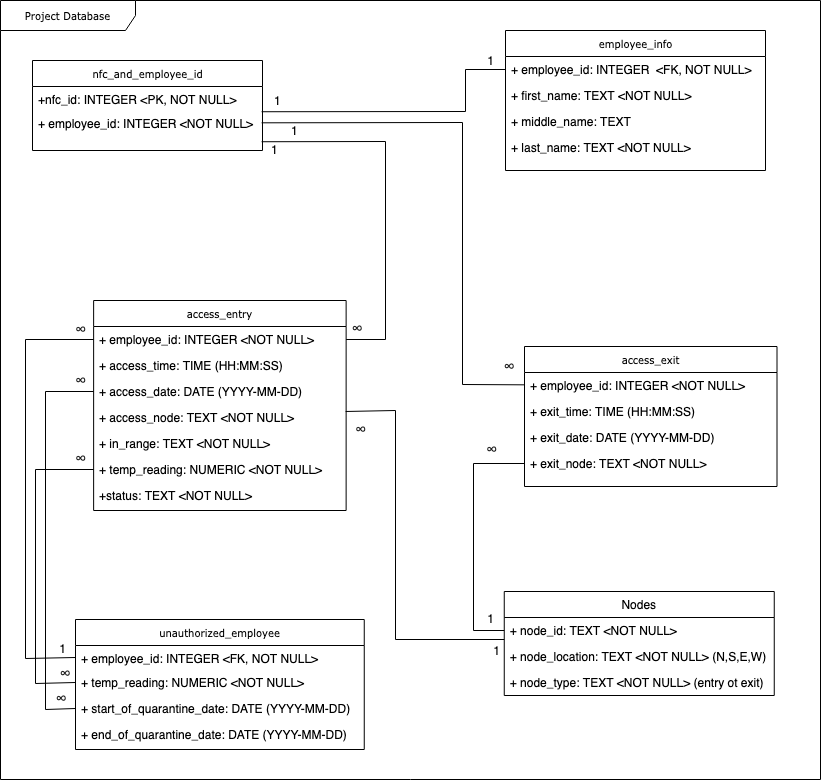
**Problem Statement:**

Our access control system will be designed to meet the needs of employers to look after the safety of their employees. The scope of this project is to provide employers with an access control system that grants access to employees entering the workplace based on their measured body temperature. As studies show that fever (< 39.1 °C) is the most frequent symptom in more than 50% of COVID-19 cases [[1](https://www.cebm.net/covid-19/in-patients-of-covid-19-what-are-the-symptoms-and-clinical-features-of-mild-and-moderate-case/)] our system also blocks access to employees with a body temperature of over 39.1 °C for 14 days to prevent the spread of COVID-19 in the workplace. Additionally, our system adheres to social distancing guidelines implemented by the government by keeping track of the number of people on the premises and restricts employees from entering the building when the maximum threshold for the workplace has been attained. Although not in the scope of the project, our system would work with employers to comply with their standards and constraints to provide their employees access to a COVID-19 free workplace. With many more features and benefits, implementing our system at workplaces around the world successfully helps in providing a COVID-19 free work environment for employees.

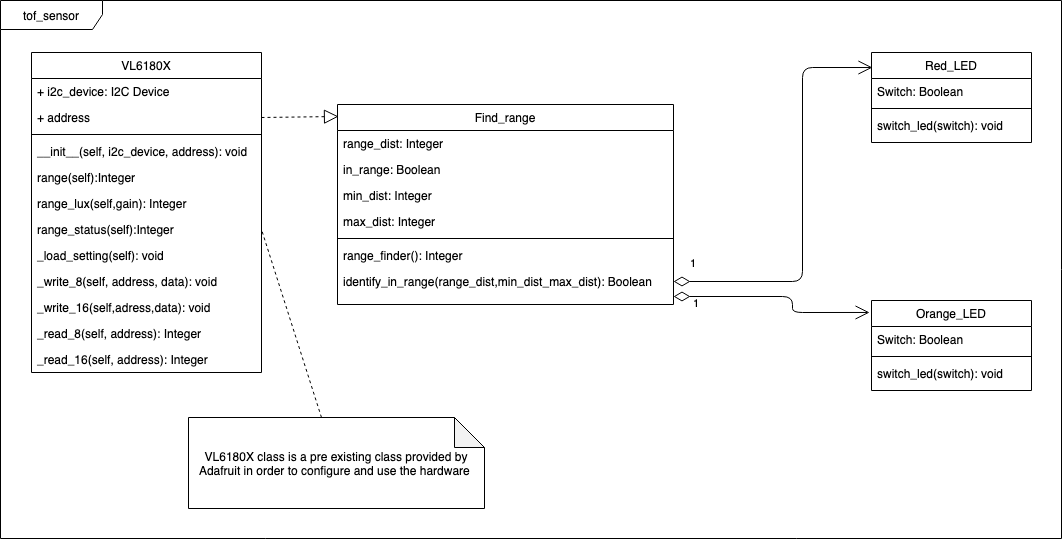
Our access control system is a technology-based solution using a data collection and management system whose primary features and their functionalities include:

* NFC security badge reader
  + Available at each door node at both entry and exit ways
  + Read valid NFC card ID and associate with employee ID
* Infrared temperature sensor
  + Available at each entry way
  + Measure the body temperature of an employee trying to enter the work premises
  + Alert control server in case of high temperature reading
* Time of flight distance sensor
  + Available at each entry way
  + Identify if the employee is standing at an ideal distance from the temperature sensor
* LED display
  + Available at each entry way
  + Display a green light at the entrance if accepting employee entry
  + Display a red circle light at the entrance when control server restricts access due to premises acquiring maximum threshold
  + Display orange light if employee in range of the temperature sensor
  + Display red light if employee not in range of temperature sensor
  + Display flashing green light if control server authorizes employee to access after having temperature recorded
  + Display flashing red light if control server unauthorizes employee from accessing the building after having temperature recorded
* Electronic door lock
  + Available at each door node at both entry and exit ways
  + Unlock only when control server authorizes the employee to enter/exit premises
  + Locked when control server prevents access to employees with recorded fever temperatures
  + Locked at entry ways when control server restricts access due to premises acquiring maximum threshold
* Control server
  + Able to access all door nodes
  + Able to access database and available tables
  + Identifies number of people entering and exiting the premises
  + Initializes access restriction when maximum threshold of premises is acquired
* Database
  + Properly organized tables with appropriate entries for each individual field
  + Properly formatted values for sensor recorded data fields
  + Records of the temperature readings of all employees are accessible for at least 14 days
* GUI
  + Able to add new employee information
  + Able to view details of all employees based on NFC security card user id and employee id
  + Able to approve quarantine period initiated by the control server for individual employees

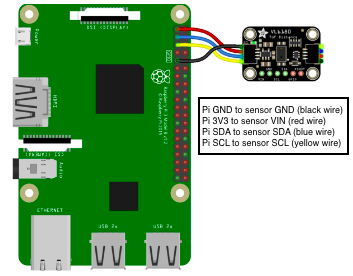
**Database Table schema:**

****

**Class Diagram for Time of Flight sensor (is this also the software interface schematics?):**

****

**Hardware Design:**

****

**Test Demo Plan 1:**

**Hardware requirements:**

* Setup Raspberry Pi’s in headless mode
* Connect sensors to Raspberry Pi’s
* If available, connect LED displays to Raspberry Pi

**Software requirements:**

* Set up Thingspeak channel
* Hard code sensor signal communication to Thingspeak
* Have test stubs which emulate the functionality of the hardware components ready

**Entry access**

Scenario 1: employee with valid NFC card, normal body temperature and ideal distance away from temperature sensor at entry node

Features to be tested:

* Communication between the NFC card reader and Thingspeak
* Communication between control server and Thingspeak
* Communication between Time of Flight range finder and Thingspeak
* Communication between LED display and Thingspeak
* Communication between Temperature sensor and Thingspeak
* Communication between control server and database
* Communication between Thingspeak and the electronic lock at entry node

Test scenario steps:

1. Control server uses Thingspeak to send the LED display a signal to display green light to indicate door node accepting entries
2. NFC reader identifies NFC id card being tapped and notifies control server of access attempt
3. NFC reader identifies tapped NFC card id and reports the id to the control server
4. Control server recognizes employee id based on NFC card id
5. Control server signals the rangefinder to identify if employee in range of temperature sensor through Thingspeak
6. Range finder reports back to the control server with the results and displays orange light
7. When in range, the control server sends a signal to the temperature sensor to measure employee body temperature through Thingspeak
8. Temperature sensor sends the recorded temperature to the control server using Thingspeak
9. Control server saves the recorded temperatures and other required information into the database
10. Control server compares recorded temperature to acceptable temperature range
11. If in acceptable temperature range, control server authorizes employee to enter building through Thingspeak
12. Control server sends a signal to the led display to display a flashing green light to indicate employee’s access has been authorized
13. Control server signals the electronic lock to unlock and adds 1 to number of people on the premises through Thingspeak
14. Control server uploads the additional entry attempt recorded data to database

Expected test scenario result: proper communication between nodes and control server occurred to authorize employee access to the workplace.

Scenario 2: employee with valid NFC card, normal body temperature and not in ideal distance range away from temperature sensor at entry node

Features to be tested:

* Communication between the NFC card reader and Thingspeak
* Communication between control server and Thingspeak
* Communication between Time of Flight range finder and Thingspeak
* Communication between LED display and Thingspeak
* Communication between Thingspeak and the electronic lock at entry node

Test scenario steps:

1. Control server uses Thingspeak to send the LED display a signal to display green light to indicate door node accepting entries
2. NFC reader identifies NFC id card being tapped and notifies control server of access attempt
3. NFC reader identifies tapped NFC card id and reports the id to the control server
4. Control server recognizes employee id based on NFC card id
5. Control server signals the range-finder to identify if employee in range of temperature sensor through Thingspeak
6. Range finder reports back to the control server with the results and displays red light to let employee know to stand in ideal range away from the temperature sensor
7. Control server signals electronic lock to stay locked through Thingspeak

Expected test scenario result: proper communication between nodes and control server occurred to let the employee know to stand in an ideal distance range away from the temperature sensor.

Scenario 3: employee with valid NFC card, body temperature over 39.1 °C and in ideal distance range away from temperature sensor at entry node

Features to be tested:

* Communication between the NFC card reader and Thingspeak
* Communication between control server and Thingspeak
* Communication between Time of Flight range finder and Thingspeak
* Communication between LED display and Thingspeak
* Communication between Temperature sensor and Thingspeak
* Communication between the GUI and Thingspeak
* Communication between control server and database
* Communication between Thingspeak and the electronic lock at entry node

Test scenario steps:

1. Control server uses Thingspeak to send the LED display a signal to display green light to indicate door node accepting entries
2. Identify tapped NFC card id
3. Notify control server of access attempt
4. NFC reader recognizes employee id based on NFC card id and reports to control server through Thingspeak
5. Control server signals the range-finder to identify if employee in range of temperature sensor through Thingspeak
6. Range finder reports back to the control server with the results and displays orange light if employee in range of the temperature sensor and red light if employee not in range
7. When in range, the control server sends a signal to the temperature sensor to measure employee body temperature through Thingspeak
8. Temperature sensor sends the recorded temperature to the control server using Thingspeak
9. Control server saves the recorded temperatures and other required information into the database
10. Control server compares recorded temperature to acceptable temperature range
11. If not in ideal temperature range, control server restricts employee from entering the building through Thingspeak
12. Control server sends a signal to the led display to display a flashing red light to indicate employee’s access has been unauthorized
13. Control server sends a quarantine approval notification to the GUI through Thingspeak
14. User approves the quarantine period notification
15. Approval is sent to the control server through Thingspeak from the GUI
16. Control server adds the employee information and recorded data to unauthorized employee table in the database and marks entry access date as the start of quarantine date
17. Using Thingspeak the control server signals the electronic lock to stay locked
18. Control server uploads the additional entry attempt recorded data to database

Expected test scenario result: proper communication between nodes and control server occurred to unauthorize employee access to workplace.

**Exit access**

Scenario 1: employee with valid NFC card and at exit node

Features to be tested:

* Communication between the NFC card reader and Thingspeak
* Communication between the control server and Thingspeak
* Communication between Thingspeak and the electronic lock at exit node

Test scenario steps:

1. NFC reader identifies NFC id card being tapped and notifies control server of exit attempt
2. NFC reader identifies tapped NFC card id and reports the id to the control server
3. Control server recognizes employee id based on NFC card id
4. Control server logs the data recorded for the exit attempt and updates the database
5. Control server signals the electronic lock to unlock and subtracts 1 from the number of people on the premises through Thingspeak

Expected test scenario result: employee successfully exits the building

**Test Demo Plan 2a:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test ID** | **Test Name/ Description** | **Test Setup** | **Expected Result** | **Actual Result** |
| 1 | Orange\_LED function activated | Have LED display set up with the Raspberry Pi connected to monitor | RGB LED displays Orange |  |
| 2 | Red\_LED function activated | Have LED display set up with the Raspberry Pi connected to monitor | RGB LED displays Red |  |
| 3 | Switch between orange and red LED’s | Have LED display set up with the Raspberry Pi connected to monitor | RGB LED initially displays orange and then switches to display red |  |
| 4 | Switch between red and orange LED’s | Have LED display set up with the Raspberry Pi connected to monitor | RGB LED initially displays red and then switches to display orange |  |
| 5 | VL6180X module activated | Have sensor connected to the Raspberry Pi and connect raspberry pi to a display | Display prints out initial reading |  |
| 6 | VL6180X module activated and Range\_Find class used to display multiple range recordings | Have sensor connected to the Raspberry Pi and connect raspberry pi to a display.  Keep moving hand over the sensor to be able to measure range. | Display prints out different readings recorded from moving hand over the sensor |  |

**Test Demo Plan 2b:**

Following features are to be tested in order to test the integrity of the database:

* Control server populating right tables
* Proper formatted data uploaded to appropriate fields
* Records of the temperature readings of all employees are accessible for at least 14 days

Test scenario 1: populating sample test data access\_entry table

Features to be tested:

* Control server populating right tables

Initial setup: set up Raspberry pi connected to monitor keyboard and mouse. Open command prompt on Raspberry Pi.

Test scenario steps:

1. Invoke SQLite by typing this command in the command prompt
   1. sqlite3
2. Open the database by typing this command
   1. .open project\_database.db
3. Populate data to employee access\_entry table
   1. INSERT INTO access\_entry VALUES(19234,time(‘now’),date(‘now’), ”S\_entry”, ”Yes”, 38.0, “Authorized”);
   2. INSERT INTO access\_entry VALUES(19437,time(‘now’,’-12 minutes’),date(‘now’), ”E\_entry”, ”Yes”, 38.3, “Authorized”);
   3. INSERT INTO access\_entry VALUES(19632,time(‘now’,’-1 hour’),date(‘now’), ”N\_entry”, ”Yes”, 39.2, “Unauthorized”);
   4. INSERT INTO access\_entry VALUES(19957,time(‘now’,’-5 minutes’),date(‘now’), ”W\_entry” ,”Yes”, 38.5, “Authorized”);
4. Query data using Select\* command
   1. SELECT \* FROM access\_entry;

Expected test scenario result: table populated with test sample data and query outputs 19234|09:30:23|2020-10-26|S\_entry|Yes| 38.0| Authorized

19437|09:18:23|2020-10-26|E\_entry|Yes| 38.3| Authorized

19632|08:30:23|2020-10-26|N\_entry|Yes| 39.2| Unauthorized

19957|09:25:23|2020-10-26|W\_entry|Yes|38.5|Authorized

Test scenario 2: populating sample test data nfc\_and\_employer\_id table

Features to be tested:

* Proper formatted data uploaded to appropriate fields

Initial setup: set up Raspberry pi connected to monitor keyboard and mouse. Open command prompt on Raspberry Pi.

Test scenario steps:

1. Invoke SQLite by typing this command in the command prompt
   1. sqlite3
2. Open the database by typing this command
   1. .open project\_database.db
3. Populate data to employee access\_entry table
   1. INSERT INTO access\_entry VALUES(19234,time(‘now’),date(‘now’),”S\_entry”,”Yes”, 38.0, “Authorized”);
   2. INSERT INTO access\_entry VALUES(19437,time(‘now’,’-12 minutes’),date(‘now’),”E\_entry”,”Yes”, 38.3, “Authorized”);
   3. INSERT INTO access\_entry VALUES(19632,time(‘now’,’-1 hour’),date(‘now’),”N\_entry”,”Yes”, 39.2, “Unauthorized”);
   4. INSERT INTO access\_entry VALUES(19957,time(‘now’,’-5 minutes’),date(‘now’),”W\_entry”,”Yes”, 38.5, “Authorized”);
4. Query the format of date and time data entries using Select command
   1. SELECT access\_time, access\_date FROM access\_entry;

Expected test scenario result: table populated with test sample data and query outputs

09:30:23|2020-10-26

09:18:23|2020-10-26

08:30:23|2020-10-26

09:25:23|2020-10-26

Test scenario 3: populating sample test data access\_entry table

Features to be tested:

* Records of the temperature readings of all employees are accessible for at least 14 days

Initial setup: set up Raspberry pi connected to monitor keyboard and mouse. Open command prompt on Raspberry Pi.

Test scenario steps:

1. Invoke SQLite by typing this command in the command prompt
   1. sqlite3
2. Open the database by typing this command
   1. .open project\_database.db
3. Populate data to employee access\_entry table
   1. INSERT INTO access\_entry VALUES (19234, time(‘now’), date(‘now’), ”S\_entry”, ”Yes”, 38.0, “Authorized”);
   2. INSERT INTO access\_entry VALUES (19437, time(‘now’,’-12 minutes’), date(‘now’,-1 day’), ”E\_entry”, ”Yes”, 38.3, “Authorized”);
   3. INSERT INTO access\_entry VALUES (19632, time(‘now’,’-1 hour’), date(‘now’, ‘-3 days’), ”N\_entry”, ”Yes”, 39.2, “Unauthorized”);
   4. INSERT INTO access\_entry VALUES (19957, time(‘now’,’-5 minutes’), date(‘now’, ‘-7 days’), ”W\_entry”, ”Yes”, 38.5, “Authorized”);
   5. INSERT INTO access\_entry VALUES (19234, time(‘now’), date(‘now’, ‘-13 days’), ”S\_entry”, ”Yes”, 38.0, “Authorized”);
   6. INSERT INTO access\_entry VALUES (19437, time(‘now’,’-12 minutes’), date(‘now’,’-14 days’), ”E\_entry”, ”Yes”, 38.3, “Authorized”);
   7. INSERT INTO access\_entry VALUES (19632, time(‘now’,’-1 hour’), date(‘now’,’-14 days’), ”N\_entry”, ”Yes”, 39.2, “Unauthorized”);
   8. INSERT INTO access\_entry VALUES (19957, time(‘now’,’-5 minutes’), date(‘now’,’-15 days’), ”W\_entry”, ”Yes”, 38.5, “Authorized”);
4. Query data using Select\* command
   1. SELECT \* FROM access\_entry WHERE access\_date > date(‘now’);

Expected test scenario result: table populated with test sample data and query outputs

19437|09:18:23|2020-10-25|E\_entry|Yes| 38.3| Authorized

19632|08:30:23|2020-10-23|N\_entry|Yes| 39.2| Unauthorized

19957|09:25:23|2020-10-19|W\_entry|Yes|38.5|Authorized

19234|09:30:23|2020-10-13|S\_entry|Yes| 38.0| Authorized

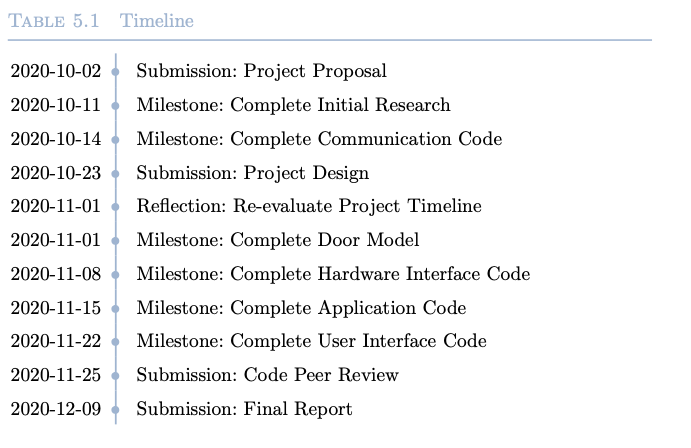
19437|09:18:23|2020-10-12|E\_entry|Yes| 38.3| Authorized

19632|08:30:23|2020-10-12|N\_entry|Yes| 39.2| Unauthorized

19957|09:25:23|2020-10-11|W\_entry|Yes|38.5|Authorized

**Project Update:**

Looking back at the Timeline initiated in the proposal, #insert timeline here#



Our project group has been working really hard in meeting the project timeline. Although things didn’t go exactly as planned in the beginning of the term, our group has successfully met most of the milestones set up so far. As some of the submission dates have been postponed, our team continues to work towards achieving set tasks and hopefully have all the required hardware and software components ready and properly set up for the demo’s. So far, we haven’t had any roadblocks regarding individual assigned tasks. We don’t think re-balancing tasks is necessary as every team member has worked hard towards accomplishing their assigned tasks and has contributed to group submission with good quality work. We would consider ourselves to be well on track to accomplishing our goals set up for the project.