ERRATA

These are some future goals, suggestions, and work that can be done to improve the SCADA system

a. **Performance Optimization:** The SCADA system currently is fairly quick with the GUI showing up briefly after the SCADA system starts up, although as the database and in memory storage (Redis) begins to fill up, the GUI sometimes faces issues in loading all the values in time. Unfortunately, this can render the SCADA system useless after an extended period of time and thus should be given high priority.
b. **Removal/Transition from 3rd party Software:** The SCADA system relies on Redis and Postgres in order to communicate with its various subsystems. Having a way for the system to work with internal csv file database or another method that does not require 3rd party would make the SCADA system even more mantinable as Redis and Postgres softwares continue to update.
c. **Watcher Snapshot Action:** Though the Watcher has the capability to write to logs, display warnings, and write to sensors, it would be useful to be able to enable sensors for a short period of time under certain conditions to be able to "snapshot" a particular sensor for that period. This would enable taking in data at a much higher frequency in comparison to taking in data at all times as it would take up far less space in the database. This could be implemented as an additional action with a period, sample rate, and sensor to enable.
d. **Watcher Exit Actions:** It would be useful for the watcher to be able to act differently on the exit of a condition to the entry of a condition. This would allow for writing to registers that keep their values on successive clock cycles and then writing another value or the initial value to them after exiting the condition.
e. **Connect Pi without Internet:** Currently PostMan service can not be run without the use of an internet connection, and so having a way to connect to the Pi on the car through USB or cables, in order to look at the PostMan system at Metzger feid (where the Pi does not have an internet connection) would help address a major use case.
f. **Splash Screen and GUI:** The GUI of SCADA is effective in communicating the sensor information to users, although having led indicators, state boxes, and visually communicating the sensor more effectively would get greater use out of the SCADA system and display. Having a splash screen during reboot would also be an additional feature that could refine the overall restart process for the SCADA system.
g. **Post-processing GUI Missing Functionality**:
    i. Currently, Postman can retrieve and export sensor data from the Postgres database, but it does not retrieve, export, or display logs. Logs generated by the Watcher module are stored in the PostgreSQL database, so retrieving and displaying these records should be straightforward. We did not design an interface

to display them alongside the data, but a possible solution would be displaying them in a separate sheet in the same Excel document as the rest.

ii. Also, it would be nice to add automatic graphing functionality to Postman. The openpyxl python library used for existing export-to-Excel functionality does support creating plots from data cells, so it should be doable without using additional software. https://openpyxl.readthedocs.io/en/stable/

iii. Currently, the process of connecting to the Postgres database is a bit convoluted. Not only do you need to know the IP address of the Pi where the Postgres database is hosted and enter it in the config.yaml file, the Pi and the computer running Postman both need to be connected to the same network such that one can reach the other through SSH. As a solution to this, we recommend two alternate methods of connecting to the database: 1) copy the Postgres database to a USB drive to be plugged into the computer you run Postman on or 2) synchronize the Postgres database with a cloud server. In either case, you will always be connecting to the same location, and the connection process will be greatly streamlined. To get you started, here is an overview of PostgreSQL replication using failover: https://www.enterprisedb.com/postgres-tutorials/postgresql-replication-and-automatic-failover-tutorial