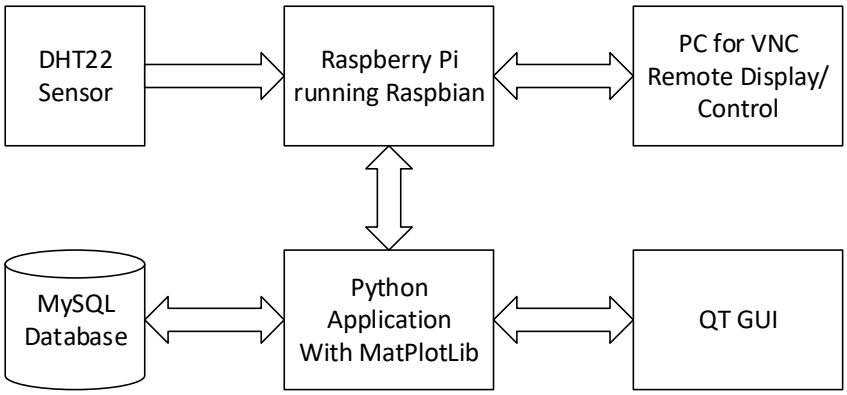
RPi3 Handling Notes:

- As with all electronics, follow ESD guidelines to avoid damaging hardware
- Whenever possible – use the Linux command line or GUI to shut down or reboot the Pi, avoid just pulling power – may corrupt the system
- If you see a yellow lightning bolt on the RPi3 GUI, you are low on power (using 2 power supplies may be needed for the Pi and an attached display for instance)
- Keep a Git or other backup of your SD card and your work – they can and do corrupt

Project 1 – worth 50 Points



- This project will introduce you to a number of technologies – the RPi3, the DHT22 sensor, QT for UIs, Python, the MySQL database, and the MatPlotLib graphics library for Python
- You will be developing a Python application using QT and MySQL on a Raspberry Pi 3 to talk to a DHT22 Temperature/Humidity sensor attached to the Pi; the display of the GUI can be shown on your PC using VNC (or other remote viewing)
- When started, your application should poll the DHT22 every 15 seconds and store the humidity and temperature data in a MySQL database table (humidity, temperature, timestamp)
- The GUI for your application should allow for
    - A button press to get an immediate reading of humidity, temperature, and timestamp from the sensor (this just has to be displayed in the UI, not stored in the database)
    - A button press for a graph of the last 10 temperature entries in the MySQL database (the graph should be created using MatPlotLib, and then the graph image can be shown in a QT image display)
    - A button press for a graph of the last 10 humidity entries in the MySQL database (the graph should be created using MatPlotLib, and then the graph image can be shown in a QT image display)
    - A status line that prints the humidity, temperature, and timestamp from the sensor every time the 15 second timer fires (or loops)
    - The same status line should indicate if the sensor is not connected (when it tries to read sensor data)
    - The GUI should display an alarm text message if the temperature or humidity from any sensor reading are over levels that can be entered on the GUI for each measure
- The program should stop automatically after 30 reads from the sensor

- For 5 bonus points, a button press should change the entire application from displaying any temperature data (in graphs or on the GUI) in degrees C to degrees F and back again
- The design of the QT-based UI is up to you

Getting Started

- Configure RPi3 with SD-based latest version Raspbian Buster OS
  - There's a chance your SD card may have Buster pre-loaded; it also may be blank
  - If you need it, the base Raspbian load is here: https://www.raspberrypi.org/downloads/raspbian/
  - Use "Raspbian Buster with desktop and recommended software"
  - uname -a at a linux command line will tell you your OS version
- Starting with your Pi
  - When you first start your Pi – change the default password for the Pi account (starts as Raspberry)
  - Also set all localization settings to US and your time zone (e.g. Denver)
    - You may need to use raspi-config to enable SSH or VNC access
    - Or you can find the settings in the Raspbian GUI under Pi Configuration/Interfaces
  - You will also want to connect your RPi3's wireless WLAN to a local WiFi access point (you can use the network icon on the GUI menu bar)
  - Finally, from a terminal window, update your Raspbian to the latest build by running:
    - sudo apt-get update
    - sudo apt-get upgrade
  - Interface a Remote Desktop or VNC connection to the RPi3 and Verify Operation
    - For VNC see https://www.raspberrypi.org/documentation/remote-access/vnc/
    - For xdrp/Remote Desktop http://xrdp.org/
    - There are other choices you can explore
- Connect DHT22 Temperature/Humidity Sensor to the Pi
  - You must source the parts below to connect the sensor to the Pi; On-campus, you can use the ECEE Electronics Store
  - You can connect directly from the sensor to the Pi or using a breadboard
  - You will need a 4.7k (or possibly a 10k) resistor across pin 1 and 2 of the DHT22
    - A skilled soldering iron user can solder the resistor across the pins – I used a breadboard
  - **For consistency, you must connect your DHT22 to GPIO4 (Pin #7)**
  - Suggested DHT22 references here https://github.com/adafruit/Adafruit_Python_DHT and https://pimylifeup.com/raspberry-pi-humidity-sensor-dht22/
  - Test your DHT22 with the example Python code from Github Adafruit DHT library or from the tutorial – you should ensure temperature and humidity data are coming from the sensor
- Install and use QT5 to develop a QT UI for your Python application (per class lecture on QT)

Project Delivery

- The code should be yours and your team's work
  - Cite any sources for any Code from the Web; should include the URL of the resource you took it from

    - You may not directly use code from other teams, although they may give you advice or suggestions
    - If someone (students or class staff) helps you on part of your code, give credits in comments and the README and identify which code was provided
- Even though this is a prototype, I'd like to see well-structured Python code
- The project must run natively on an RPi3 development system
- The code must be well commented
    - A typical comment template can be copied (forked) from example.py in my GitHub repo at https://github.com/brmjr9/eid-fall2018
    - Comment/Docstring header for each file identifying the author and file description
    - Comments/Docstrings at any functions or classes including description, input, output
    - Comments for purpose of data structures or complex transactions (usually this is a why and not a how)
- Turn in a GitHub repo link (one submission per team) containing your project with
    - Any code files needed to run the project (not including standard libraries)
    - A README.md (markdown text file) including:
        - Title (i.e. EID Project 1)
        - Names of the developers/students on your team
        - A section called **Installation Instructions**
            - We should be able to follow the instructions to run your project on my RPi3 system (for Project 1,2,3,4 – not the Super-Project)
        - A section called **Project Work**
            - On a multi-person project, include who was responsible for what parts
        - A section called **Project Additions**
            - Describes any features you've added that are above the project requirements

Grading Rubric

- Project must be turned in via GitHub URL
- README.md structured as reviewed above – 10 points (-2 per missing element)
- Demonstration of features by executing project – 25 points
    1. Reading from sensor on demand – 5 points
    2. Reading from sensors every 15 seconds and storing in MySQL, stops after 30 reads – 5 points
    3. Humidity and Temperature MatPlotLib graphs on demand – 5 points
    4. Status line display of incoming sensor data and of disconnected sensor or sensor failure – 5 points
    5. Alarm set, display, and monitor function – 5 points
- For 5 bonus points, a button press should change the entire application from displaying any temperature data (in graphs or on the GUI) in degrees C to degrees F and back again
- Well commented and structured code – 15 points (-2 to 4 for poor commenting, -2 to 4 for poor structure)
- 15% Grade penalty if turned in late (accepted for one week, then 0 points awarded)