**Program Design**

* C Macros
  + **GOAL:** This section is responsible for stating overall C Macros to be used. These macros mainly serve the purpose of creating a more readable program while also ensuring the upmost efficiency possible including but not limited to, removal of unnecessary variable declaration during run-time and decreased use of memory during run-time.
  + **GUIDE:** No macro serves to be used for any functional purpose other than serving to create a more semantic program.
  + **DEFINITIONS:** Definitions of macros are always to be defined capitalized with “M\_” prefixing the variable to note to the user that such a variable is a macro.
    - #define M\_SYSTEM\_STATE\_IDLE 0
    - #define M\_SYSTEM\_STATE\_DISABLED 1
    - #define M\_SYSTEM\_STATE\_ERROR 2
    - #define M\_SYSTEM\_STATE\_RUNNING 3
* Global Variables
  + **GOAL:** This section is responsible for stating overall program global variables to be used. Global Variables within the program unlike C Macros, global variables can change throughout the lifetime of the program. This allows global variables to be used in a powerful manner that allows the program to know several statuses including the state of the program itself.
  + **GUIDE:** Global Variables are to be used to store the statuses of sensors and or different states of the program.
  + **DEFINITONS:** Definitions of global variables are always to be defined lowercased with “g\_” prefixing the variable to note to the user that such a variable is a global variable.
    - unsigned char g\_system\_state = M\_SYSTEM\_STATE\_DISABLED

**Hardware Design**

* User Facing Statuses
  + **GOAL:** This section defines any devices to be used for the purpose of outputting system messages to the user in any text (LCD) or visual form (LEDs).
  + **GUIDE:** Any output devices are to be used within the appropriate scopes within the main loop of the program.
  + **DEFINITONS:** Definitions of output variables are always to be semantically understandable. It is recommended to follow the format of “o\_” as a prefix following a specific use of the output for the device and then a general device name.
    - unsigned char o\_running\_state\_LED
    - unsigned char o\_error\_state\_LED
    - unsigned char o\_idle\_state\_LED
    - unsigned char o\_disabled\_state\_LED
* System Reading Devices
  + **GOAL:** This section defines any devices to be used for the purpose of reading inputting system devices to the program.
  + **GUIDE:** Any input devices are to be used within the appropriate scopes within the main loop of the program.
  + **DEFINITONS:** Definitions of input variables are always to be semantically understandable. It is recommended to follow the format of “i\_” as a prefix following a specific use of the input for the device and then a general device name.
    - unsigned char i\_status\_water\_sensor
    - unsigned char i\_status\_humidity\_sensor
* System Operation Devices
  + **GOAL:** This section is responsible for stating devices that are not to be used for neither input nor output but general upkeep or functioning of the device. Throughout the program lifetime, these devices behavior changes as the program responds either to user input from either the sensor or user action.
  + **GUIDE:** These devices are to be constantly changed via their defined variable throughout the program lifetime.
  + **DEFINTIONS:** Definitions of system operation devices variables are always to be semantically understandable. It is recommended to follow the format of “sod\_” as a prefix following a specific use of the device and then general device name.
    - unsigned char sod\_vent\_motor
* Wiring Diagram

Testing Outline

Acceptance Criteria:

* The system produces an alert when the water level is below a certain threshold.
  + The system does not produce an alert when the water is above that threshold.
* The system monitors the current air temperature.
  + The system enables or disables a fan when the temperature goes out of bounds of a given range.
* The system displays the current air temperature on the LCD screen.
* The system monitors the current humidity.
* The system displays the current humidity on the LCD screen.
* The system has a user-controlled output vent.
* The system can be toggled on/off by a user.
* The system logs the time and date every time it is turned on or off.

Test Cases:

State-indicating LEDs:

* Disable the system. A yellow LED light should be illuminated, and no monitoring of any sort should occur.
* Induce some sort of error to the system (i.e. water level too low). A red LED should illuminate and an error message should be displayed.
* Enable the system and ensure that a green LED illuminates.
* Set up the system so that it is in the running state (motor should be on) and assert that a blue LED is illuminated.

State:

* Press the power on/off button. The system should immediately enter the idle state.
* Move to a warmer environment, or start the system in a warmer environment. Assert that state changes to running.
* Allow the system to run, ensure that state returns to idle once the temperature drops sufficiently.
* Remove water from the reservoir. Assert that state changes to error state.
* Add water to reservoir. Assert that state changes back to idle.