2024421

by Matthew Perry

Submission date: 04-May-2025 08:27PM (UTC+0100)
Submission ID: 257772215
File name: 16803_Matthew_Perry_2024421_183770_1739248029.pdf (5.29M)
Word count: 13939
Character count: 80714

Course: BSc (Hons) Electrical and Electronic Engineering

Module: SGEL6000 Independent Project

Assignment: 2. Project Design, Implementation and Evaluation

Name: Matthew Perry

University number: 2024421

Date submitted: 04/05/2025

Word count: 6155

Good Structure

Contents

Acknowledgements 6

1. Executive Summary 7

2. Introduction 8

3. Design 9

3.1 Solution Overview 9

3.2 Physical Layout 10

3.3 Electrical Drawings 10

3.4 Software 12

3.4.1 Arduino 12

3.4.2 Nextion 14

4. Implementation and Management 17

4.1 Procurement 17

4.2 WBS & Project Plan 18

4.3 Construction and Assembly 18

4.3.1 Mechanical 19

4.3.2 IP Rating Considerations 19

4.3.3 Electrical Assembly 20

4.3.4 Simulation 21

5. Testing and Results 22

5.1 Graphical Results 28

5.2 Adjustments 29

6. Evaluation of Project Outcomes 30

6.1 Addressing Constraints 31

6.2 Quality Management and Continuous Improvement 33

7. Conclusions and Recommendations 34

8. Citation List 36

9. Appendix 39

9.1 Final Gantt Chart 39

9.2 Planned and Actual Spending Spreadsheet..... 40
9.3 Employers Statement of Contribution 42
9.4 Design 43
9.5 Software Implementation 51
9.6 Risk Assessment 64
9.7 Practical Implementation 66
9.8 Simulation..... 69
9.9 Testing 72
9.10 Logbooks..... 79

Figure 1; Overview of Solution 9
Figure 2; Nextion Page 0 15
Figure 3; Nextion Page 1 16
Figure 4; Final Gantt Chart 39
Figure 5; Cost Sheet Summary 40
Figure 6; Employers Statement of Contribution 42
Figure 7; Physical Layout Overview 43
Figure 8; Layout Diagram Internal 43
Figure 9; Layout Diagram Front Enclosure 44
Figure 10; Layout Diagram Dosing Pump 45
Figure 11; Eagle Electrical 46
Figure 12; Electrical Diagram 47
Figure 13;12V Perfboard Layout 48
Figure 14;5V Perfboard Layout 48
Figure 15; Transistor Circuit Layout 49
Figure 16; Software Program Structure 51
Figure 17; Gimp Example 53
Figure 18; Powerpoint Nextion Backgrounds 53
Figure 19; Nextion Font Generation 54
Figure 20; Nextion Font List 54
Figure 21; Nextion program.s 55
Figure 22; Arduino Code 1 56
Figure 23; Arduino Code 2 57
Figure 24; Arduino Code 3 58
Figure 25; Arduino Code 4 59
Figure 26; Arduino Code 5 60
Figure 27; Arduino Code 6 61
Figure 28; Arduino Code 7 62
Figure 29; Arduino Issue Before Adjustment 63
Figure 30; Resolved Arduino Issue 63
Figure 31; Completed Construction 66
Figure 32; Perfboards 67
Figure 33; Motor Enclosure 67
Figure 35; Soldering LED Wiring Looms 68
Figure 34; Completed Construction Front 68

Figure 36; Nextion Editor Simulator 70
Figure 37; Temperature and LED Evidence 72
Figure 38; Dosing Time Configuration Evidence 73
Figure 39; Wash Time Evidence 73
Figure 40; Manual Purge Evidence 74
Figure 41; Wash Time Removable Evidence 74
Figure 42; 0V Evidence 75
Figure 43; Password Protection Evidence 75
Figure 44; Replace Chemical Evidence 76
Figure 45; Timer Comparison Evidence 76
Figure 46; Dosing Evidence 77
Figure 47; Temperature Testing Graph 77
Figure 48; Pie Chart Success Rate 78

Table 1; Nextion Timer Descriptions 16
Table 2; Test Plan 28
Table 3; Planned Spending Cost Sheet 40
Table 4; Actual Spending Cost Sheet 41
Table 5; Arduino Pin Out 45
Table 7; Power Supply 49
Table 6; Component Load Schedule 49
Table 8; Electrical Colour Codes 50
Table 9; Nextion Components Addressing 52
Table 10; Risk Assessment 65
Table 11; Arduino Simulation 70
Table 12; Nextion Editor Simulator 71

Acknowledgements

The student would like to express their gratitude to the company for their support with this project. The company provided financial support of £305, which was essential in sourcing key components. The input for the amendment of the FDS specification D1.2 which outlines 2 LED indicators to be replaced with 3 to improve ergonomic performance is appreciated. Lastly, I would like to acknowledge the loan of calibrated testing equipment, including an IsoTech Fast-Cal thermal bath and digital multimeter, which enabled accurate testing of the completed systems performance.

Did your lecturers not support you?

1. Executive Summary

This project focused on the design, development, and testing of a functional sink accessory prototype intended for the use within a medical decontamination department to support the manual cleaning process of endoscopes. The system aimed to automate key elements such as dosing, temperature monitoring, visual alerts, and timing, in line with infection control and safety protocols.

The project followed a structured approach, using the Functional Design Specification (FDS) and managed using a Gantt chart. Key milestones were achieved within ±1 week of their scheduled deadlines, across the project timeline from January to May 2025. Open-source tools were adopted throughout the project, including Arduino IDE and Nextion Editor for software development, and QElectrotech, Eagle, and Microsoft Office for schematic design, documentation, and planning.

Physical construction was carried out using both hand and power tools, which resulted in enclosure modifications that affected the original IP66 rating. As a result, the system's final IP rating could not be verified to meet the intended IP55 minimum. Electrical assembly involved soldering two perf boards, installing DIN-mounted components, and wiring the control system to meet functional and safety requirements all while wearing PPE in line with the risk assessment.

The final prototype was successfully completed under budget, with a total spend of £402 from an available £500. The system achieved an overall test pass rate of 89% based on the defined test plan, which was influenced from the FDS. The project demonstrates successful integration of mechanical, electrical, and software elements within the context of a healthcare engineering application.

This is a really great summary that tells your readers everything they need to know about your project.

2. Introduction

This report will present the design, development, testing and evaluation of the sink accessory prototype intended for the use of hospital decontamination departments. The project will be carried out independently in the aim of improving safety, efficiency and compliance of the manual cleaning process of flexible endoscopes.

The chosen solution (2) automates key stages of the process including accurate dosing, temperature monitoring, timing and alarm indicators. The prototype will be developed using a combination of management techniques, design techniques, software development, practical assembly, simulation and testing. The system will be created with attention to inclusivity, environmental sustainability, and compliance.

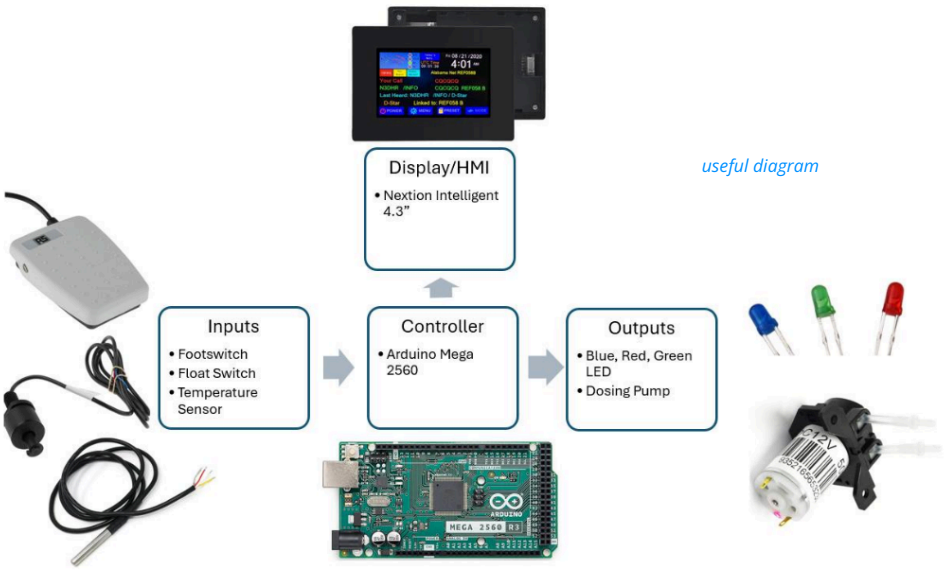
This report will detail the full life cycle of this project including the chosen solution, constraints, design implementation, management techniques, testing outcomes and final evaluation. It concludes with recommendations for improvement and reflection.

A good introduction guides the reader through the report

3. Design

3.1 Solution Overview

Figure 1 displays an overview of the chosen solution that will be developed to meet the functional design specification which was solution 2. It consists of inputs such as a footswitch, float switch and temperature sensor which will be processed by the controller which is an Arduino Mega 2560. The Arduino will communicate with a Nextion HMI which will display process information such as the state of the inputs and offer control. Outputs will be set dependent on the logic that is developed in the Arduino/Nextion program.



useful diagram

Figure 1; Overview of Solution

The physical layout will consist of 3 main components shown in Figure 7 which will meet reference A1 in the functional design specification.

(Perry, 2024)

3.2 Physical Layout

To lay out all the components which will be inside the main enclosure effectively, a Microsoft Word document was created with gridlines. By determining that each block represents 5x5mm for the layout components could be scaled to against the grid. Using the calculation:

$$Original\ Size\ \div 5\ x\ 0.43 = Scaled\ Size$$

- Where 0.43 is the actual size of the square in centimetres.

For example, the enclosure is 241 x 194 x 107mm, therefore for scaling it would be 20.72 x 16.68 x 9.20mm. By applying this to all the components and setting the size of the shapes in Word, an accurate layout was completed which is shown in Figure 8. This method was carried out for the front of the enclosure to ensure that the screen fit and did not obstruct internal components such as the power supplies as shown in Figure 9. Lastly, to adhere to reference A1.5 where all components must be capable of wall mounting and for safety reasons, the pump would need to be housed. This method was useful in this case as the enclosure would be an obscure size as it needed to be small length by width but with a larger depth to account for the motor as shown in Figure 10. Lastly these diagrams will aid as a reference for placement and construction such as cutting or drilling whether that's for mounting or modifying material such as DIN rail.

(Perry, 2024)

3.3 Electrical Drawings

Using QElectroTech, a free open-source electrical design software, a main power electrical diagram was created as shown in Figure 12. The diagram shows 230VAC supply entering the system through a 3-pin plug with a 3A fuse for protection. The live is connected through the panel-mounted swich located in the enclosure. This acts as a safety feature ensuring power is disconnected when the front panel is open. From there the live, neutral and earth conductors are terminated into separate DIN-rail terminal blocks. These terminals then feed two separate DC power supplies which are 12VDC supplying the pump board (see Figure 13) which feeds the 12V pump and Arduino VIN. The 5V board (see Figure 14) is supplied from the Arduino for the 5V outputs and inputs. All grounds are connected to a common point to ensure a stable reference across the system. A dedicated 5V supply is used for the Nextion rather than drawing power from the Arduino as the display requires higher current than the Arduino's onboard regulator can safely provide.

A further in-depth diagram which illustrates electrical connections was designed using Eagle shown in Figure 11. This diagram focuses are on the 12V & 5V connections, including switch operation for the inputs, serial communication and outputs. There are connection identifiers on the diagram which show that there should be a connection made along with numbers that correspond to the Arduino. Table 5 was created to aid this diagram which shows the pin out configuration for the Arduino.

As mentioned, the perf board (Figure 14) is powered by the 5VDC output from the Arduino, which supplies the DS18B20 temperature sensor. The temperatures signal is connected through a 4.7K ohm pull-up resistor to 5V for proper communication using the 1-wire communication protocol. Additionally, 3330-ohm resistors are placed on the cathode legs of the connected LEDs going to ground. These resistors limit the current through each LED to prevent damage and safe operation (Poole, 2022). The diagram also shows the negative side of all input devices such as switches connected to a shared ground rail, additionally the Nextion display is connected here as without a common ground voltage levels between devices can become unreferenced leading to communication errors.

330 Ohm

Figure 13 shows the 12V perf board layout. This shows the VIN from the 12VDC power supply which is used to power the Arduino Mega and the dosing pump. A 2N2222 NPN transistor is used for switching to control the pump. The transistor is driven by a digital output from the Arduino allowing 5V logic to switch the 12V pump circuit.

Why choose the 2N2222

A detailed diagram of this switching circuit has been created using Multisim shown in Figure 15. A 1kΩ base resistor is connected between the Arduino output pin and the base of the transistor which limits the current into the base protecting both components. When the Arduino sends 5V to the base the transistor turns on allowing the current to flow from the 12V supply through the collector to the emitter which is connected to the ground powering the dosing pump (Electronics Tutorials , 2024). Additionally, a 1N4007 flyback diode is placed across the pump terminals to protect the circuit from voltage spikes generated by the pumps inductive load.

Correctly cited

why such a high voltage rated diode?

Lastly, the DC motor inrush current can be as much as 2-3 times its steady state current, a power supply table (Table 7) and a load schedule (Table 6) has been compiled (Amethern 2024). This states the load and supply of the components, ensuring that the load is below for the power supplies and fuses rating. For example, the maximum current draw for the 12VDC supply is 1.25A and at normal operation it draws 0.67A. However, at startup it could reach 1.1A. This shows all the power supplies and fuse has been correctly implemented for safe operation. Colour coding for the wiring has been compiled for reference in the implementation stage shown in Table 8.

Correctly cited

3.4 Software

A flowchart of the software operation was shown in the presentation which is shown in Figure 16. It shows that at idle state, it displays temperature and checks the bottle float switch. However, when the footswitch is activated the pump doses 100ml and the timer starts once the temperature is met and dosing is complete. This operation has now changed for a few reasons as it won't be efficient under certain conditions such as a healthcare professional not realising that the timer has started. Now, the timer will only start once the button is pressed on the Nextion HMI. Lastly, the float switch will trigger a notification on the HMI if the float switch is open.

3.4.1 Arduino

The Arduino code was created on Arduino IDE which is an open-source free software programming application that uses a language similar to C and C++ (Arduino, 2024). Correctly cited

Due to the Arduino and Nextion communicating with each other, events such as buttons and text boxes needed to be documented in a separate excel spreadsheet. The page, ID number and object names were documented as these would need to be declared in the Arduino code as Nextion objects. By doing this, it was possible to track the components and properly address them from the Arduino, which can be seen in Table 9.

The Arduino code is structured to handle all the tasks and hardware states required. The "setup()" function initialises all the hardware components such as inputs or outputs and the Nextion screen. This ensures the system is ready to operate and offers visual cues on the screen.

The program enters the "loop()" function, which is the main driver for all the actions, inside the loop the program begins by checking inputs from the Nextion using the "checkNextion()" function. This function will check for events from the screen such as touch button presses such as countdowns, manual mode and text inputs.

The countdown was implemented within the Arduino code which is controlled through "startCoundown()" function. This is activated through the screen and the time is determined through a specified time from the display and updated in regular intervals. The countdown is checked and updated in the "loop()" to provide feedback and when it reached 0 it displays a message to indicate its completed.

A manual mode has been created to allow Engineers or approved Users to directly control the outputs through the HMI. However, to prevent the system from staying in manual mode completely, the program includes a 15 second auto reset that exits manual mode after the countdown. This ensures that manual mode is turned off and goes back into an automatic state without user intervention.

The code manages the dosing operation via the “checkDose()” function which monitors the foot switch and float switch states. When the footswitch is pressed, and the float switch is high indicating that its safe the program activates the output for the pump for a specified duration. This duration will be determined during commissioning to meet the specified 100ml by the FDS. The function also monitors the float switch to ensure the pump will be turned off or not started if there is no liquid to ensure the pump doesn’t run dry which can cause increased sound levels, motor damage and lead to leaks (Bate, 2018). Correctly cited

Finally, serial monitoring has been implemented for debugging and displaying important characteristics such as temperature, manual mode, countdown status and dosing events. Using the “Serial.print()” and “Serial.println()” commands provide live feedback which makes it easy to verify that the program is working correctly (Söderby, 2025) . The full code is shown from Figure 22 to Figure 28 and comments (//) are added throughout the code to explain the purpose of each section helping to aid understanding of the logic.

3.4.2 Nextion

In addition to the Arduino software, the Nextion required its own software, Nextion Editor, which is an HMI-builder software that uses some pre-made libraries but also requires strict syntax outlined by the Nextion instruction set list. There were different stages while designing this HMI which included creating graphics, implementing timers, buttons and other events.

Firstly, due to the size of the screen being 480x272 background images needed to be created as this can't be done efficiently on the Nextion. Gimp which is a free image manipulation software was used to create the early stages of the HMI development which can be seen in, Figure 17,however, this was a time-consuming solution as amendments were frequent to find the most ergonomical layout. Instead, backgrounds were created on Microsoft power point (Figure 18) and using the snipping tool captured and resized to 480 x 272 px on Microsoft Paint. This was an easier solution where multiples while still maintaining the image quality.

Text that was used on within the Nextion software such as text displays had to be generated using a font generator which can be seen in Figure 19. This will create a file which will then correspond to a font such as style and size that is then imported into a list, then assigning it to the attributes which is shown in Figure 20.

The HMI is comprised of 4 pages, which of 2 are visible and 1 is a keypad. The keypad has been imported for inputting times in temperature offset, dosing and the password. Code implemented in program.s which is the initialising code that will run and states that page 0 will start up which is the homepage shown in Figure 21.

Below shows a revealed image of what the homepage will look like when all notifiers and settings exposed. All controls but manual operation is on this page and comprises of multiple events such as text inputs, buttons, and images which can be referenced back to the Nextion Components table. Text inputs are then entered and read by the Arduino whereas other text such as t_temp and t_remaining are referenced and handled by the Arduino code.

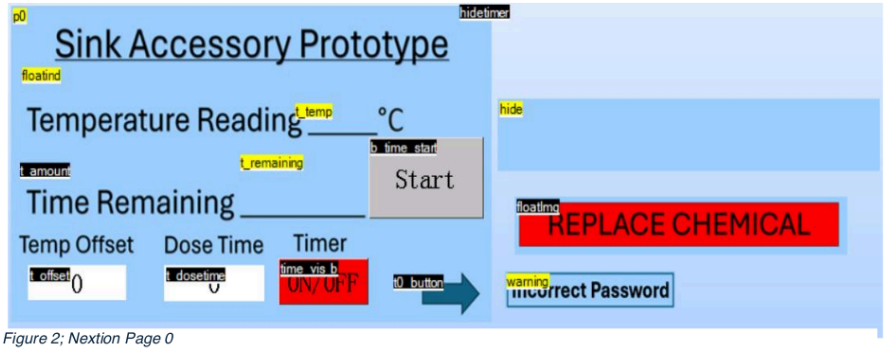


Figure 2; Nextion Page 0

Where the Nextion needed to handle interactive changes to meet the FDS and move between pages, 3 timers were implemented on this page for event handling. Table 1 outlines the names, purpose and the code implemented.

Name	Purpose	Code

tbutton	This timer will check the numerical input of t0_button at an interval of 50ms. It will check for 3 conditions which are "111" which is the password to move to the next page, "222" which will remove the covers exposing the settings. It will check for any other number to display the incorrect password notifier. Additionally, if "111" is entered it will enter "0" to clear the input. Therefore, a password will be required to move each time its returned to the home page.	if(t0_button.txt=="111") // Password 111
		{
		t0_button.txt="0" // clear the input field
		page 1 // go to Page 1 if correct
		}else if(t0_button.txt=="222") // Password 222
		{
		vis hide,0 // unhide
		vis warning,0 // hide the warning
		t0_button.txt="0" // clear the input
		}else if(t0_button.txt=="0")
		{
		vis warning,0 // hide warning if field is cleared
		}else
		{
		vis warning,1 // show warning if wrong password
		}

float	This timer will check the numerical input of floatind at an interval of 500ms. A number is inserted by the Arduino code where if the float switch input is high, it inserts 1 and low inserts 0. Therefore, the timer is checking the value to then display the low chemical indicator.	if(floatind.txt=="1") // where 1
		{
		vis floatImg,1
		}
		if(floatind.txt=="0") // where 0
		{
		vis floatImg,0}

timernotif	This timer is checking the value of time_vis_b which is a dual state button. If it detects the value is 0 it will hide the cover, but if it detects 1 it will display the cover. This logic will be referenced against the switch “on/off”.	if(time_vis_b.val==0)
		{
		vis hidetimer,0
		}else
		{
		vis hidetimer,1
		}

Table 1; Nextion Timer Descriptions

Page 1 is used for manual control of the outputs which will be used by engineers during commissioning, fault finding or servicing. The b_manual button will be used to initiate a timer within the Arduino code which will then allow the outputs to be switched manually for up to 15 seconds. Adittionally, b0_back also had Nextion commands to return to the previous page which does not require a password.

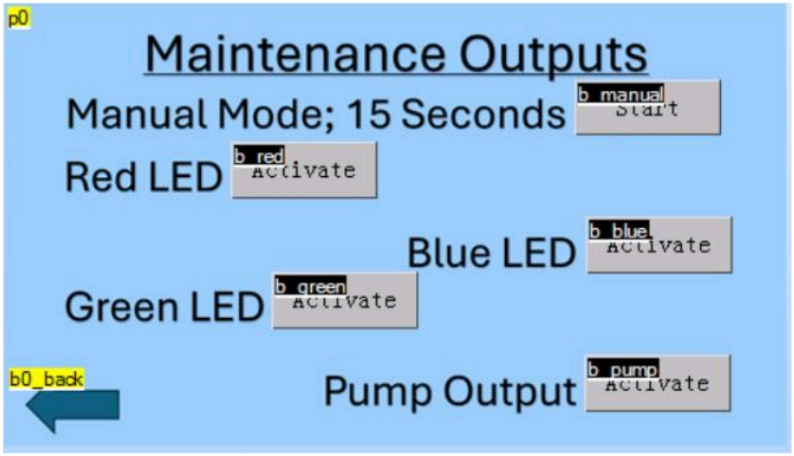


Figure 3; Nextion Page 1

4. Implementation and Management

4.1 Procurement

To track and ensure that the project would be completed within the specified budget and time a sourcing strategy was implemented. Initially, a spreadsheet was created to source the parts for the chosen solution which could be within budget shown in Table 3. A running cost spreadsheet was compiled that listed characteristics which would aid the development as shown in Table 4. This is the table which the parts were bought initially and throughout the project.

From the table, all selected parts had short lead times, ensuring early delivery to support the project schedule. The cost including VAT was recorded for each item to allow accurate tracking of total expenditure and to ensure the project remained within the allocated budget of £500 which is specified in the project proposal.

The URL of each item was included to trace the supplier which supports the traceability of goods and services. This will aid any issues related to faulty components, repeat orders and manage future procurement which is important as this is a prototype.

Once an order was placed, the purchase details were documented in the spreadsheet such as the order date. Once delivered, it was documented and each part was inspected for damage and to verify that it matched the specifications required, as any defects could be recorded and addressed quickly.

4.2 WBS & Project Plan

(Perry, 2024)

Using a work breakdown structure created for combating scope creep and keeping the project aligned with goals referenced with time (Krasteva, 2019). It was broken into four main sectors which were sourcing materials, hardware development, software development and testing.

Correctly cited

Utilising this breakdown structure allowed the student to estimate how long a task would take and create a timeline of events where a Gantt chart could be created. The Gantt chart served as a guide and overview of the project that detailed start and end dates, milestones and dependencies with other tasks (Adobe, 2022).

Correctly cited

As shown in Figure 4, the larger tasks on the left-hand column were specified in the work breakdown structure and the top horizontal row shows the estimated deadline milestone and underneath is the actual completion. It can be said that the timeline was relatively followed due to some tasks running late by a week. Also, the layout shows that when one task ends another starts, however, this was not ideal to follow. Due to hardware and software interacting with each other it made sense to check that they were compatible and communicating with each other. Additionally, simulation ran alongside the software as it was the best way to verify that the code implemented was working in blocks rather than completing the whole program and having to debug errors within a large program.

4.3 Construction and Assembly

Once all the components were delivered and verified against the order/cost sheet, the layout of the enclosure was cross referenced with the digital design to make sure its compliant with the original specifications. This shown that it was mostly accurate however the DIN rail needed to be moved down slightly to accommodate the mounting holes. The physical build was approached methodically combining mechanical and electrical tasks while following safety protocol. Before starting any work, the workspace was clean and free of obstructions, tools were inspected, and the risk assessment was briefed as shown in Table 10. The whole process was conducted in accordance with the Health and Safety at Work Act 1974 (HSE, 2024), with reference to additional relevant regulations.

4.3.1 Mechanical

To mount the electrical components such as terminals and power supplies, a DIN rail was cut to the required length using a hacksaw, following measurements taken with a tape measure to match the mounting points. The rail was secured in a bench vice during cutting and prior to cutting the tools were inspected for defects or damage. The correct PPE such as safety goggles and gloves was worn and edges deburred to avoid cuts which is in line with PUWER regulations (HSE, 2024). Additionally, Perspex was chosen to mount the base of the Arduino and perf boards due to its rigidity, style and electrical insulation. Due to it being available in a standard sheet it was measured and cut to size using a hacksaw and holes were drilled for mounting.

For the front of the enclosure, the Nextion and LEDs needed to be mounted. Measurements were made and ensuring that the screen would be level and cleared internal components an outline was drawn. In each corner a hole was drilled and using a junior hacksaw the void was cut out to insert the screen. Similarly, spacing for the 3 LED bezels were spaced an equal 3rd and using a drill a pilot hole was drilled and then the required size to reduce the risk of cracking or drift. The pump mounting required a larger aperture, so it was drilled using a hole saw attachment which provided a cleaner and more controlled circular cut which is shown in Figure 33. Then the pump was inserted ensuring the seal fit well and two pilot holes were drilled to mount with screws. Holes for the cable glands were measured and drilled located on the bottom left so water or moisture would not enter by gravity and the glands have rubber inserts to form an efficient watertight seal.

4.3.2 IP Rating Considerations

While the enclosure is rated IP66 which provides complete protection against dust ingress and protection against powerful water jets from any direction however this does not remain valid after modifications (BSI, 2013). Drilling the holes for cable glands, LED bezels and mounting displays compromises the original seal unless all holes are resealed using appropriately rated components and materials. To combat this, IP68- rated cable glands were used for both incoming and outgoing wiring, providing a higher ingress protection where a weatherproof seal is required (Index Marine , 2024). The LEDs were also sealed using waterproof adhesive sealant to prevent ingress around the bezels.

The main uncertainty lies with the Nextion display, which despite having a front gasket to mount flush against the panel it does not have a manufacturer specified IP rating. As a result, even if its sealed carefully into position it's not possible to confirm the maintained IP rating post modifications.

4.3.3 Electrical Assembly

To ensure that the electrical circuits were correct, they were first prototyped on a breadboard. This included the transistor-based switching circuit that was tested under load conditions with a Multimeter to verify voltage levels and signal behaviour. Once it was tested and correct the design was transferred to perf boards shown in Figure 32.

Component positions were marked and set out based on the schematic layout. Terminal screw blocks were used to support both a solid connection and ease of future servicing by engineers. The soldering was performed in a well-ventilated environment and aided with a ventilation fan with a charcoal filter. Tips were inspected and cleaned prior to use to avoid oxidation, and flux was applied to ensure clean joints. While soldering components, care was taken to not apply to much heat which may damage the components. After soldering, continuity testing was carried out to check that there weren't shorts. Lastly, excess flux was cleaned off using isopropyl alcohol and a soft brush, while doing so wearing safety glasses and gloves and after use returning the solution in adherence to COSHH guidelines (Chemtronics , 2015).

Correctly cited

Wiring looms for the LEDs and motor were soldered with strain relief and insulated with heat shrink tubing to prevent exposed conductors shown in Figure 35. Jumper bars were installed in the DIN rail terminals to simplify power distribution across common voltage connections.

All terminal connectors and power supplies were fixed to the DIN rail to ensure stability during operation. An earth terminal was installed with conductive metal clips bonding the DIN rail to the system earth meeting the requirement for BS 7671 (IET Wiring Regulations, 18th edition). This will then prevent electric shock or interference under fault on the DIN rail.

Final wiring was carried out in accordance with the wiring diagrams and cables were organised using cable ties and routed to simplify maintenance. Where possible, wire terminations were crimped using insulated ferrules or DuPont connectors for reliable and serviceable connections.

Figure 31 displays the completed construction where all components are shown, and the front panel can be seen in Figure 34.

Lastly, the Nextion program was outputted to a micro-SD card and inserted to flash the screen, and the Arduino was connected directly to the laptop and uploaded.

4.3.4 Simulation

The Arduino program was verified before testing through simulation using the serial monitor function. This was done by using a laptop to run a serial monitor and then pressing touch events on the Nextion such as buttons for manual mode, inserting numerical values and general system operating procedures. The table which displays the simulation feedback can be shown in Table 11.

Additionally, the Nextion program was verified using the Nextion Editor Simulator shown in Figure 36. By running the simulator on a laptop, button presses, and time inserts were carried out with the cursor. From this, button operation could be verified from reading the hex code and referencing it against the Nextion component table shown in Table 12.

5. Testing and Results

Once the build was completed, some simulation was completed regarding the Nextion display, this consisted of testing the movement between pages, testing inputs and accessing settings which could be verified visually and with the output hex code while referencing the Nextion component spreadsheet which can be seen in Figure 36.

Once the construction phase was complete, a structured testing process was carried out to verify that the prototype met the requirements outlined in the Functional Design Specification. Prior to testing, all measurement instruments such as a digital Multimeter and a temperature probe were checked for calibration certificates. By doing this, the instruments will be valid ensuring a complete accuracy of tests (Amplivox, 2024).

Testing was done in a controlled and clean environment using certified equipment to maintain traceability and integrity of results. Each test was designed to be quantitively assess the performance, functionality and safety of the system. Any issues or deviations will comment on & will be discussed later in the project outcomes.

Test Plan						
Test Number	FDS Reference	Description	Expected Result	Actual Result	Pass/Fail	Evidence
1	A1	Press the footswitch.	Dosing sequence starts immediately after pressing the footswitch.	Footswitch pressed & dosing starts.	Pass	N/A
2	B1	Place calibrated probe next to systems temperature probe in a container filled with water at 27 degrees by using a calibrated probe.	The systems temperature reading is within 1 +/- degree of the calibrated probe reading between 26 and 28 degrees.	Set up Isotech fast cal. Place indicated probe with system probe in water at 27 degrees read by indicator.	Pass	N/A
3	B1.1	Fill container with water at 23 degrees referencing a calibrated probe for accuracy. Using the systems digital display alter offsets/potentiometers to match. Check display to ensure system is reading 23 degrees.	The systems temperature probe is calibrated accurately, matching the reading of the calibrated reference probe and on the system displays within +/- 1 degree (Between 22 and 24 degrees).	Water was at 25 degrees. Poured cold water in to reach 23.23 indicated and system read 23.25.	Pass	Figure 37
4	B1.1	Fill container with water at 27 degrees referencing a calibrated probe for accuracy. Using the systems digital display alter offsets/potentiometers to match. Check display to ensure system is reading 27 degrees.	The systems temperature probe is calibrated accurately, matching the reading of the calibrated reference probe and on the system displays within +/- 1 degree (Between 26 and 28 degrees).	Increased water temperature to 27.08 indicated and system read 27.13.	Pass	Figure 37

5	B1.1	Fill container with water at 32 degrees referencing a calibrated probe for accuracy. Using the systems digital display alter offsets/potentiometers to match. Check display to ensure system is reading 32 degrees.	The systems temperature probe is calibrated accurately, matching the reading of the calibrated reference probe and on the system displays within +/- 1 degree (Between 31 and 32 degrees).	Increased temperature to 32.09 indicated and system read 32.06.	Pass	Figure 37
6	B1.3	In the settings page, set the wash timer to 5 minutes, return to home, then back to settings page and set to 10 minutes.	Wash timer begins counting down from 10 minutes after initiation.	Pressed on the time text set to 300 seconds, exited keypad and then re-entered set to 600 seconds. Pressed start counted down from 10:00 to 00:00	Pass	Figure 39
7	B1.4	The dosing timer can be reconfigured to any duration up to 2 minutes.	The timer is set from 0 to 120 at 30 second intervals and the pump doses for the correct time.	Entered password “222” entered 0,30,60,90, 120 and then pressed footswitch. Pump dosed accurately at all durations.	Pass	Figure 38
8	B1.5	Manual purge option to prime pump by pressing the manual purge button.	Press manual purge button, pump switches on and primes pump tubing.	Entered password “111” to access manual page. Pressed start manual activations and then pump held pump output button. Pump activated and tube was primed.	Pass	Figure 40

9	B1.6	Check the wash timer can be removed from the user interface by disabling in the settings page.	The wash timer is removed from the user interface page.	Entered “222” pressed timer On/Off button. Removed the wash timer from the page. On/Off button went from green to red.	Pass	Figure 41
10	C1	Select settings page, password should be requested and enter wrong password.	Entry is denied and a message state that the password is incorrect.	Selected bottom right arrow that allows access to settings or manual page. Entered “05” which resulted in “Incorrect Password” notifier.	Pass	Figure 43
11	C1.1	Power will be automatically disconnected to components when opening the front panel of the control box.	Testing with a multi-meter that the power is isolated and reads 0V.	Removed front panel and heard the switch disconnect. Tested AC common live terminal which shown 0V.	Pass	Figure 42
12	C1.3	System should be able to be isolated at mains power 230V by turning off switched socket.	Testing with a multi-meter should show that the system is not energised (0V).	Turned off switch at socket. Used a multimeter and shown 0V on incoming terminals.	Pass	Figure 42

13	D1.1	Fill a container and insert the dosing lance. Remove the lance and then check the display.	An alarm indicates that the bottle is empty on the digital display.	Removed lance out of container which was full. Looked at the screen and alarm in red indicated " Replace Chemical ".	Pass	Figure 44
14	D1.2	Check red and green indicator for temperature out of limits. Using calibrated reference probe ensure the water is below 23 degrees and then check the indicator. Increment the temperature by 1 degree up until 32 and check the indicator. Lastly, increase the temperature past 32 degrees and check the indicator.	The indicator shows the following: - Below 23 degrees = Red - Between 23-32 degrees = Green - Above 32 degrees = Red	LED shown as follows : 23 degrees or below = Blue Between 23-32 degrees = Green Above 32 degrees = Red	Fail	Figure 37
15	D1.2	Check the service alarm by setting the calibrated date, then amend the current date to a year later in the settings page.	Annual service alarm will appear on user interface.	N/A	Fail	N/A

16	E1.1	Set the wash timer to 5 minutes, press start and record the time separately using a stopwatch.	Recorded and indicated results are within +/- 5% of each other. (Between 04:45 to 05:15 minutes).	Set wash timer to 300 seconds, and Pressed start on system and the stopwatch lap timer at the same time. Stopped lap timer once system reached 00:00. This then shown on the lap timer 04:59:76	Pass	Figure 45
17	E1.2	Set dosing time to achieve 100ml, insert lance into a 500ml cylinder and press footswitch to commence dosing. Record result and repeat the action 3 times.	Recorded results are within +/- 5% of each other. Results read between 95ml to 105ml.	Entered “222” set dose time to 85 seconds. Inserted lance into a container filled with water and pressed the foot switch. After each dose recording the contents on a level surface and pouring back into the dosing bottle. Readings followed: - 1st dose : 100ml - 2nd dose: 100ml - 3rd dose: 100ml	Pass	Figure 46

18	E1.3	Check the repeatability of the float switch by removing and inserting the lance from a container filled with water, while checking the display. Repeat the action 3 times.	A low-level alarm indicator is shown each time after removal and then cleared after re-inserted.	Removed and inserted lance from filled container 3 times and after 1 second of being removed notification stating “Replace Chemical” shown.	Pass	Figure 44

Table 2; Test Plan

5.1 Graphical Results

The test results of the 3 setpoints of temperature calibrated versus the system was plotted on a graph to compare the values shown in Figure 47 which shows the biggest deviation between the calibrated and systems temperature 0.05 of a degree. It was also determined that the system has a pass rate of 89% which can be shown in a pie chart (Figure 48).

5.2 Adjustments

During testing, an issue was identified with the handling of the dosing in relation to the float switch. The float switch correctly triggered the alarm “Replace Chemical” when low however, the safety interlock logic for the dosing sequence was reversed. This meant the system only allowed the pump to activate for a specified time when the container was empty which was not the intended behaviour.

The Arduino code was inspected which shown “if (flowSwitchState == LOW) which can be seen in Figure 29. The code was amended code to “if (floatSwitchState == HIGH) “ which means if there is liquid then carry out sequence shown in Figure 30. This amendment rectified the issue allowing the system to operate as intended.

6. Evaluation of Project Outcomes

Completing the testing, each characteristic in the Functional Design Specification was evaluated against actual performance outcomes. It can be said that many physical and software-based features performed as expected.

Physical A1 to A1.5

The foot switch reliably triggered the pump, the control box components were possible for wall mounting and a Type-G plug was used. Although the dosing pump was an individual item, it was housed efficiently in its own enclosure which has the option for wall mounting making up the main three components which are the footswitch, pump and control box. A1.4 stated that the dosing pump must have a yellow cover near the tubing connections, however this was not met. The original plan was to create a design around the pump front and using a 3D printer with yellow filament create the cover, although time and financial resources did not allow it.

Software B1 to B1.6

The temperature probe could be calibrated using a negative or positive offset and displayed values within ± 1 degree, with the largest deviation of 0.5 at the 27-degree set point. Timers for both wash and dosing were reconfigurable, with the manual purge button being implemented on page 1 which required a password. It could be argued that this option should be on the homepage, however, to prevent misuse of the system it is guarded. Additionally, the removal of the wash timer was available with a dual state button.

Safety C1 to C1.6

Password protection to access the settings page was implemented. Although there was no page change, settings features were only visible and accessible after inputting “222” into the keypad. The design was completed this way because values were appropriately sent and handled to the Nextion due to command handling.

Power was isolated to the terminals once the panel was opened. This was done through a disconnection pressure switch where the front of the panel pushed in the switch which was mounted on the back piece of the enclosure. Additionally due to the supply being from a Type-G plug power was able to be isolated at the socket. A sticker will be placed on the outside of the panel to indicate a potential electrical hazard on the final prototype, but at the moment with only competent engineers using the prototype, to reduce cost and resources, it will be postponed until release.

Lastly, the control box has an IP rating of IP66, however this has been compromised by modifying it with installing IP66 glands and LED bezels which have been sealed. The main issue that is now caused is the Nextion HMI does not have a specified IP rating by the manufacturer, which means that the IP rating cannot be determined.

Alarms D1.1 to D1.3

The low-level liquid warning was carried out effectively, but handled unconventionally within the software, where at HIGH it would send a “1” to a text box that was checked by a timer and carried out a visibility command.

The LED indicators based on temperature had been amended based on input from the student’s employer. Instead of 2 LED indicators red and green, 3 indicators were used which were blue (cold), green (suitable) and red (hot). This operation worked successfully and will allow users to have a better understanding of the limits due to the representation and connotations with the colours while still displaying the thresholds (Interaction Design Foundation - IxDF, 2021).

Lastly, the overdue service alarm was not implemented as the student struggled to store the data in the internal eeprom of the Arduino. Although a time and date was implemented on a hidden page, it could not be saved due to it utilising the Nextion real time clock functions and not transferring to the Arduino.

Performance E1 to E1.3

The performance specifications were all met with temperature, time and dosing all within tolerances and float switch repeatability being successful.

6.1 Addressing Constraints

All key project constraints outlined during the planning stage were considered & implemented to the best of the student’s ability.

Ethical and Inclusive Design

The project followed guidance from the EDI Engineering Handbook (Royal Academy of Engineering , 2024), by including accessibility features such as a foot switch, high contrast text and background layouts and a user-friendly interface. The amendment to the LED indicators supports a better ergonomic design and can inform the users more effectively about why the water temperature is out of limits at a glance which can support staff in high pressure environments by reducing errors Additionally, this prototype will serve as a tool rather than a replacement which means it will maintain the ethical balance and support workforce wellbeing (Douglas & Raban, 2017).

Environmental Sustainability

The design did implement low power electronics and reconditioned components where possible as it is a prototype. These steps support the circular economy action plan (European Commission , 2023) and ISO 14001 principles. Although it was stated in the constraints that parts would be bought from companies that are ISO 14001 accredited this proved difficult as smaller companies and companies that ship internationally find the accreditation expensive and time consuming (Maiden, 2021), which was applicable when it came to cheaper electronic components such as resistors.

Legal and Commercial

During construction of the design, the health and safety at Work Act 1974 was ensured by conducting a risk assessment, ensuring a clear and safe work environment with the use of suitable PPE and designing the unit to be minimise physical strain by the hands free footswitch and secure mounting points to reduce trip hazards and clutter (HSE, 2024).

The control of substances hazardous to health regulations were addressed by including a float switch which will prevent the users interacting with chemicals when un-necessary. Additionally, during the testing phase, the use of detergents was avoided by replacing it with water for simulation (HSE , 2021). When isopropyl alcohol was used to clean excess flux during soldering, the safety data sheet was reviewed, and the appropriate PPE was worn. Once the task was complete, the chemical was adequately stored with guidance from the recommendations on the data sheet.

BS EN 60204-1:2018 for machinery safety and isolation standards was attempted to be followed by using an IP rated enclosure and cable glands. All terminal blocks were designed to be enclosed and guarded, and in relation to the 230VAC terminals insulated jumper bars were used (BSI, 2021). Colour coding with wiring proved difficult due to preformed wires in addition to 12VDC & 5VDC being used, however, this was combatted by separating the circuits with perf boards and using colours such as black, red, yellow and white while detailing in them in a table.

The prototype will also aim to reduce healthcare associated infections through correct cleaning practices. This has been done by providing accurate temperature monitoring, precise timing and displaying visual and status indicators with comply with standard operating procedures and can be verified through the test plan results. Complying with the National institute for Health and Care Excellence focuses on the end users needs and drive the prototype forward for continuous improvement (NICE , 2021).

Lastly as the project remains in the prototype stage, a full patent was not registered due to early-stage development and financial and legal costs. However, the visual and structural design is protected under the UK unregistered design rights for up to three years after disclosure (HM Government, 2024). This protection prevents others from copying the appearance of the prototype without consent.

6.2 Quality Management and Continuous Improvement

Quality management was implemented through practical structured methods. The FDS was used as a benchmark to evaluate the performance of all features and functions. Test results were directly compared against the FDS and stated expected outcomes in the test plan where any discrepancies were documented and resolved. One example of this was the float switch performing in the reverse desired behaviour which was identified and corrected. The Gantt chart served a tracking sheet to maintain the project timeline while maintaining key milestones aiming for quality outcome.

To support traceability and accountability photographs of each development phase were labelled and stored in the cloud. This approach enabled backup and remote access to evidence and design changes which aligns with best practices in documentation management and quality assurance (British Assessment Bureau, 2024). Additionally, key information such as data sheets, calibration records and safety data sheets were stored digitally and in an organised folders ensuring component origins and the history could be tracked and reviewed if necessary.

Although the project was completed independently an under limited resources, decisions were made to adopt continuous improvement principles, rather than delaying progress due to no access to premium design software such as AutoCAD, open-source tools such as QElectrotech and Eagle were implemented to ensure the integrity and traceability of the design. This outcome proves that effective quality can still be achieved in under resourced conditions.

7. Conclusions and Recommendations

In conclusion, this project is a successful development of a working prototype intended for use in medical decontamination environments. The system was completed within budget, with a total cost of £402 which was below the £500 budget.

Functionality was demonstrated through structured testing with a final pass rate of 89%, validating the design of against the FDS. This pass rate could be argued to be higher as half of the failure rate came from the LED indication layout. This amendment was requested by the employer which entailed the implementation of 3 LEDs instead of 2 for better ergonomic values. Critical elements such as the temperature monitoring, dosing and level indication all performed reliably and within tolerance.

The design placed a strong emphasis on ergonomics, with a good panel layout, high contrast LED indicators and text and a footswitch to minimise strain during operation. Software development and schematic designs were carried out using open-source software including IDE, QElectrotech, GIMP, Microsoft Office platforms and Nextion Editor which provided sufficient functionality while maintaining low development costs.

The practical consisted of cutting, drilling and filing to insert components into the panel and wiring and soldering from an electrical perspective. This was done while wearing the appropriate PPE and carrying out necessary steps in line with the risk assessment.

Serial communication between the Arduino and Nextion display was simulated using the Nextion Editor and Arduino Serial Monitor allowing the button presses, command structure and system response to be verified before full integration.

Testing was carried out using calibrated instruments such as a thermal bath and a digital multimeter ensuring accuracy and reliability of recorded values Documentation and progress tracking were maintained using management techniques such as a Gantt chart, logbooks and regular cloud backups with supported traceability and safeguarded the projects integrity.

Although the prototype successfully met its primary objectives, several areas for future development have been identified. While the current Nextion display performed well with cost, ergonomics and handling, it lacks a manufacturer specified IP rating which poses potential risks. Considering the nature of its application it would be advisable to replace the component that meets an IP rating of IP55 or higher which was specified in the FDS.

Another area for improvement could be the electrical design, DuPont connectors could be improved or replaced with terminal screw connections, with the addition of trunking and shrink wrap. A custom PCB that integrates all the control such as the Arduino handling and 12V transistor circuit could be developed to improve long term reliability and streamline assemblies.

The float switch could be upgraded to a different manufacturer where the suction lance is integrated with the float switch (Reed Electronics AG, 2024) , however this incurs a greater financial cost. Although efforts were made to separate 5V and 12V wiring, with no specifications for colour codes, cable labelling would support better serviceability.

If further funding becomes available, investment into premium electrical design software may improve documentation quality especially for regulatory submission or professional review. Intellectual property should be revisited if the prototype progresses towards commercialisation as a registered design or utility patent would offer stronger protection (Kenton, 2023).

Lastly, the project demonstrates a clear application of engineering practice, continuous improvement and real-world problem solving. It has implemented management techniques, design, digital and practical construction, testing and simulation into a functional product. Despite some limitations it offers a reliable foundation for future development and potential entry into the healthcare market.

Wide Range of Sources

8. Citation List

Correct method of listing.

Adobe, 2022. *What is a Gantt chart? Benefits, how to use them and more.* [Online]
Available at: https://business.adobe.com/uk/blog/basics/what-is-a-gantt-chart
[Accessed 19th April 2025].

Ametherm, 2024. *How to limit DC motor inrush current.* [Online]
Available at: https://www.ametherm.com/blog/inrush-current/dc-motor-inrush-current
[Accessed 14th April 2025].

Amplivox, 2024. *Equipment Calibration*. [Online]
Available at: https://www.amplivox.com/education/knowledge-hub/what-is-calibration-and-why-is-it-important
[Accessed 25th April 2025].

Arduino, 2024. *Arduino/Processing/Python Language Comparison.* [Online]
Available at: https://docs.arduino.cc/tutorials/generic/language-comparison/
[Accessed 12th April 2025].

Bate, E., 2018. *Why Some Peristaltic Pumps Run Dry.* [Online]
Available at: https://sauermanngroup.com/en-GB/insights/why-some-peristaltic-pumps-run-dry
[Accessed 16th April 2025].

British Assessment Bureau, 2024. *10 Tips to Maintain ISO 9001 Certification*. [Online]
Available at: https://www.british-assessment.co.uk/insights/10-tips-to-maintain-iso-9001-certification/
[Accessed 29th April 2025].

BSI, 2013. *BS EN 60529:1992+A2:2013.* 2nd ed. s.l.:BSI.

BSI, 2021. *BS EN 60204-1:2018.* [Online]
Available at: https://knowledge.bsigroup.com/products/safety-of-machinery-electrical-equipment-of-machines-general-requirements-2
[Accessed 29th April 2025].

Chemtronics , 2015. *Safety Data Sheet IPA-ISOPROPYL ALCOHOL.* [Online]
Available at: https://www.phi.com/assets/documents/Safety-Data-Sheets/isopropyl-alcohol.pdf
[Accessed 22nd April 2025].

Douglas, H. & Raban, M., 2017. Applied Ergonomics. *Improving our understanding of multi-tasking in healthcare: Drawing together the cognitive psychology and healthcare literature*, March, pp. 262-273.

Electronics Tutorials , 2024. *Transistor as a switch.* [Online]
Available at: https://www.electronics-tutorials.ws/transistor/tran_4.html
[Accessed 12th April 2025].

European Commission , 2023. [Online]
Available at: https://chatgpt.com/c/6810c00a-1c0c-8002-9360-2fde1db5aaba
[Accessed 26rh April 2025].

HM Government, 2024. *Intellectual Property: an overview*. [Online]
Available at: https://www.gov.uk/intellectual-property-an-overview
[Accessed 15th October 2024].

HSE , 2021. *https://www.hse.gov.uk/coshh/.* [Online]
Available at: https://www.hse.gov.uk/coshh/
[Accessed 29th April 2025].

HSE, 2024. *Health and Safety at Work etc. Act 1974.* [Online]
Available at: https://www.legislation.gov.uk/ukpga/1974/37
[Accessed 24th April 2025].

HSE, 2024. *Provision and Use of Work Equipment Regulations 1998 (PUWER).* [Online]
Available at: https://www.hse.gov.uk/work-equipment-machinery/puwer.htm
[Accessed 19th April 2025].

Index Marine , 2024. *Waterproof Glands*. [Online]
Available at: https://indexmarine.com/products/waterproof-glands/
[Accessed 22nd April 2025].

Interaction Design Foundation - IxDF, 2021 . *What is Color Symbolism?.* [Online]
Available at: https://www.interaction-design.org/literature/topics/color-symbolism
[Accessed 29th April 2025].

Kenton, W., 2023. *Utility Patent: Definition, How It's Issued, Search, and Example*. [Online]
Available at: https://www.investopedia.com/terms/u/utility-patent.asp
[Accessed 30th April 2025].

Krasteva, I., 2019. *What Makes Work Breakdown Structure Important for Projects?.* [Online]
Available at: https://businessmap.io/blog/work-breakdown-structure
[Accessed 19th April 2025].

Maiden, T., 2021. *Is ISO 14001 good for small businesses?*. [Online]
Available at: https://greensmallbusiness.com/is-iso-14001-good-for-small-businesses/
[Accessed 27th April 2025].

NICE , 2021. *Surgical site infections: prevention and treatment.* [Online]
Available at: https://www.nice.org.uk/guidance/ng125
[Accessed 30th April 2025].

Poole, N., 2022 . *Light-emitting Diodes (LEDs).* [Online]
Available at: https://learn.sparkfun.com/tutorials/light-emitting-diodes-leds/how-to-use-them
[Accessed 14th April 2025].

Reed Electronics AG, 2024. *Standard Reed Suction Lances.* [Online]
Available at: https://reed.ch/en/sensor-system-products/suction-lances/standard-suction-lance/
[Accessed 30th April 2025].

Royal Academy of Engineering , 2024. *Why equality, diversity and inclusion matter.* [Online]
Available at: https://raeng.org.uk/edi-engine#:~:text=Why%20equality%2C%20diversity%20and%20inclusion,and%20safe%20solutions%20for%20everyone.
[Accessed 15th October 2024].

RS Components , 2024 . *IP Ratings Explained.* [Online]
Available at: https://uk.rs-online.com/web/content/discovery/ideas-and-advice/ip-ratings
[Accessed 22nd April 2025].

Söderby, K., 2025. *Using the Serial Monitor tool.* [Online]
Available at: https://docs.arduino.cc/software/ide-v2/tutorials/ide-v2-serial-monitor/
[Accessed 16th April 2025].

One more source needed: Perry, 2024, Assignment 1 .....

9. Appendix

9.1 Final Gantt Chart

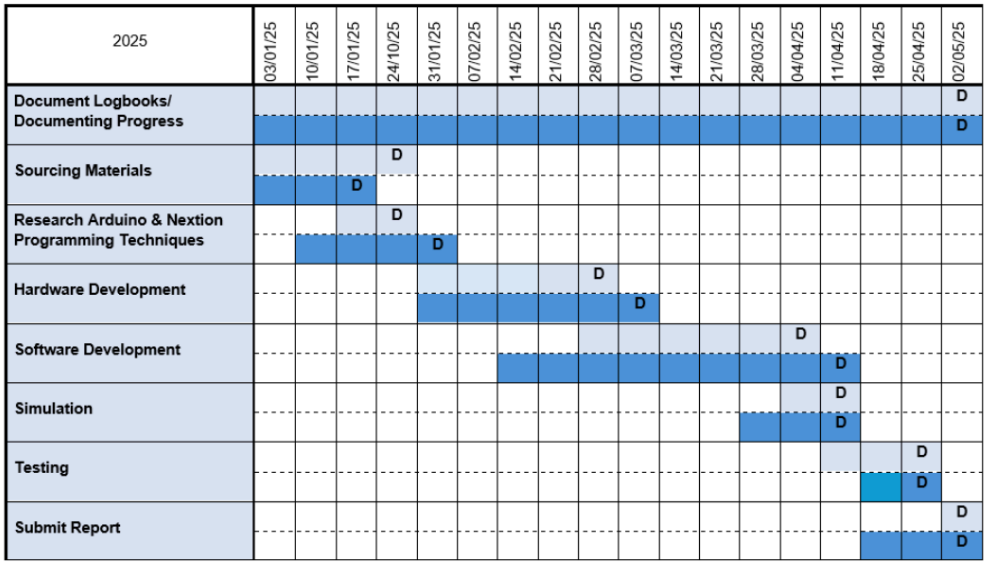


Figure 4; Final Gantt Chart

9.2 Planned and Actual Spending Spreadsheet

Cost Sheet						
Part Description	Quantity	URL	Delivery Time	Cost (exc. VAT)		
Arduino Mega 2560	1	fmBOoqEyAL	1-2 Days	£30.00		
Nextion Intelligent Display	1	ion-43-intellig	1-3 Days	£57.50		
Footswitch	1	om/web/p/foo	0	£24.70		
RGB LED 5 Pack	1	.com/web/p/le	1-2 Days	£1.66		
LED Bezel	2	m/web/p/led-h	1-2 Days	£5.00		
Peristalic Pump	1	-dc-power?srs	2-4 Days	£17.20		
Float Switch	1	/web/p/float-s	1-2 Days	£19.86		
PVC Pipe	1	TE3NTY2ajE	5-7 Days	£5.50		
Panel Switch	1	ducts/schneic	7 Days	£24.79		
24VDC Power Supply	1	/p/din-rail-pov	1-2 Days	£24.72		
Mains Plug	1	/	1 Day	£2.00		
12VDC Power Supply	1	web/p/din-rail-	1-2 Days	£11.74		
Wiring Arduino	1	prototyping-ju	5-7 Days	£11.99		
Enclosure	1	p/p/general-pu	1-2 Days	£50.43		
Flyback Diode 1N4007	10	/1843698?srs	3-5 Days	£2.23		
Transistor 2N222	2	k/product/10x	3-5 Days	£1.42		
Breadboard	1	/	/	/		
Various Resistors	/	/	/	/		
Perfboard	1	636?srsltid=A	/	£4.72		
Temperature Sensor	1	sor-extras?srs	3-5 Days	£3.50		
Miscellaneous	/	/	/	£75		
Total				£373.96	Inc. VAT	£448.75

Table 3; Planned Spending Cost Sheet

Completed Cost Summary	
Budget	£500
Predicted Cost	£434.51
Actual Cost	£402.78
Amount Left	£97.22

Figure 5; Cost Sheet Summary

Running Cost Sheet						
Part Description	Quantity	URL	Delivery Time	Cost (exc. VAT)	Order Date?	Delivered?
Arduino Mega 2560	1	fmBOoqEyAL	1-2 Days	£30.00	09/01/2024	13/01/2025
Nextion Intelligent Display	1	ion-43-intellig	1-3 Days	£57.50	09/01/2024	15/01/2025
Footswitch	1	com/web/p/foo	/	£24.70	On Site	On Site
RGB LED 5 Pack	1	.com/web/p/le	1-2 Days	£1.66	09/01/2024	13/01/2025
LED Bezel	3	m/web/p/led-h	1-2 Days	£7.50	09/01/2024	14/01/2025
Peristalic Pump	1	-dc-power?srs	2-4 Days	£17.20	09/01/2024	13/01/2025
Float Switch	1	/web/p/float-s	1-2 Days	£19.86	09/01/2024	14/01/2025
PVC Pipe	1	CTE3NTY2ajE	5-7 Days	£5.50	09/01/2024	13/01/2025
Panel Switch	1	oducts/schneic	7 Days	£24.79	09/01/2024	13/01/2025
5VDC Power Supply	1	r-15-5/841898	1-2 Days	£10.80	09/01/2024	13/01/2025
Mains Plug	1	63139443656	/	£1.35	On Site	On Site
12VDC Power Supply	1	web/p/din-rail-	1-2 Days	£11.74	09/01/2024	13/01/2025
Wiring Arduino	1	prototyping-ju	5-7 Days	£11.99	09/01/2024	13/01/2025
Enclosure	1	o/p/general-pu	1-2 Days	£50.43	On Site	On site
Flyback Diode 1N4007	10	/1843698?srs	3-5 Days	£2.23	09/01/2024	17/01/2025
Transistor 2N222	2	k/product/10x	3-5 Days	£1.42	09/01/2024	17/01/2025
Breadboard	1	/	/	/	On site	On Site
Various Resistors	/	/	/	/	On site	On Site
Perfboard	1	636?srsltid=A	/	£4.72	On site	On Site
0.25 Wiring Assorted	1	prototyping-ju	1-2 Days	£11.99	28/02/2025	29/02/2025
Enclosure Pump	1	ug-Small-Exte	/	£4.20	In store	29/02/2025
Din Rail Terminal Ends	1	gue/products/1778480-stand		£4.65		
Din Rail Terminals	9	e/products/177	/	£0.96	In Store	29/02/2025
Din Rail Jumper Bar	1	779968-10-wa	/	£5.15	In Store	29/02/2025
1.5 Flex	1	AAAAAD8IdPv	/	/	On Site	On Site
Enclosure 2	1	s/1868488-24	/	£19.74	26/02/2025	29/02/2025
Cable Gland	2	AAAAAD8IdPv	/	£2.07	In store	29/02/2025
Temperature Sensor	2	sor-extras?srs	3-5 Days	£3.50	09/01/2025	17/01/2025
Miscellaneous	/	/	/	£0	/	/
Total				£335.65	Inc. VAT	£402.78

Table 4; Actual Spending Cost Sheet

9.3 Employers Statement of Contribution

DATE: 28TH TH APRIL, 2025

[redacted]

DESCRIPTION: Matthew Perry's, Dosing Control Project.

Employers Statement:

The dosing unit project is an important step for [redacted] as the company looks to expand into the medical decontamination market. Up until now, the company has mainly focused on third-party services like maintenance, validation, and repairs for decontamination equipment. With the development of this product, there's now a chance to move into designing and selling in-house equipment — helping the business grow and offering more long-term stability.

Matthew was given the task of designing and building the dosing unit. To keep the product affordable and compact, he decided to use a custom PCB. Since no one else at the company has experience with PCB design or embedded programming, he had to work independently throughout the design and development stages. The only input given during the project was related to a few design suggestions, such as having three conditions shown by the LEDs, and including 240V terminal blocks in the design to leave options open for ultrasonic integration in the future and allowing him to take testing equipment such as thermocouples and multi-meters.

Despite the limited support, Matthew delivered excellent results. He kept the costs low and the design efficient, making sure the final product would be good value for money. The project has shown that the company has the potential to move beyond just servicing equipment.

The cost that has incurred is a total of £305 excluding V.A.T.

Figure 6; Employers Statement of Contribution

9.4 Design

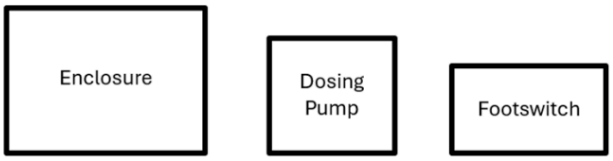


Figure 7; Physical Layout Overview

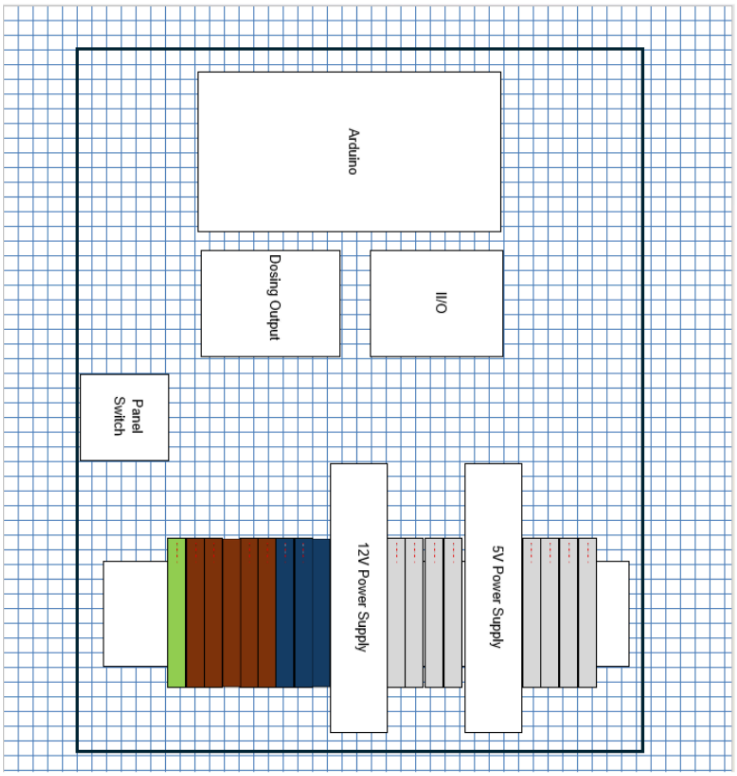


Figure 8; Layout Diagram Internal

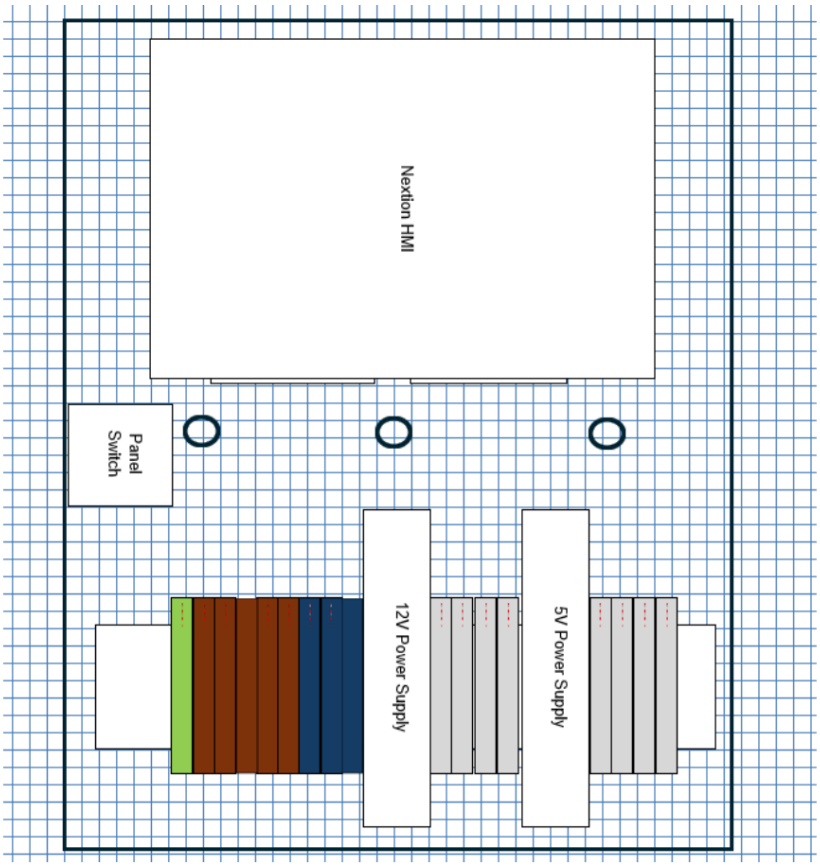


Figure 9; Layout Diagram Front Enclosure

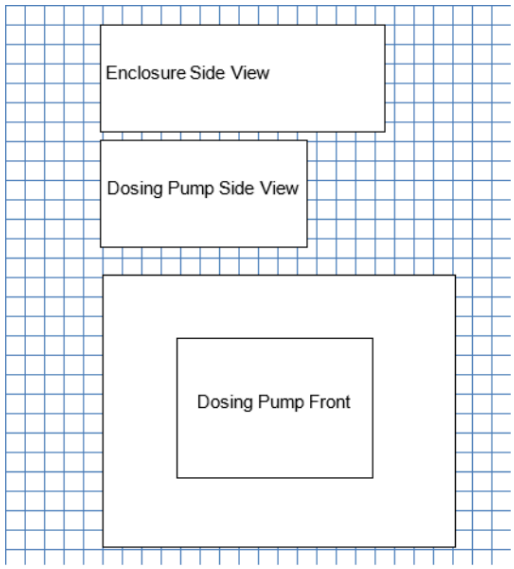


Figure 10; Layout Diagram Dosing Pump

Arduino Pin Out			
Pin	Charcateristic	Connection	Comment
4	Power	5V+	Supply for inputs and LED outputs.
5	Power	GND	
6	Power	GND	
7	Power	VIN	12V+
5	Digital	LED blue	
6	Digital	LED green	
7	Digital	LED red	
8	Digital	Pump output (signal)	This is not a direct output but the signal to switch 12V+.
52	Digital	Foot switch	
53	Digital	Float switch	
16TX2	Communication	RX Nextion	Serial 2
17RX2	Communication	TX Nextion	Serial 2

Table 5; Arduino Pin Out

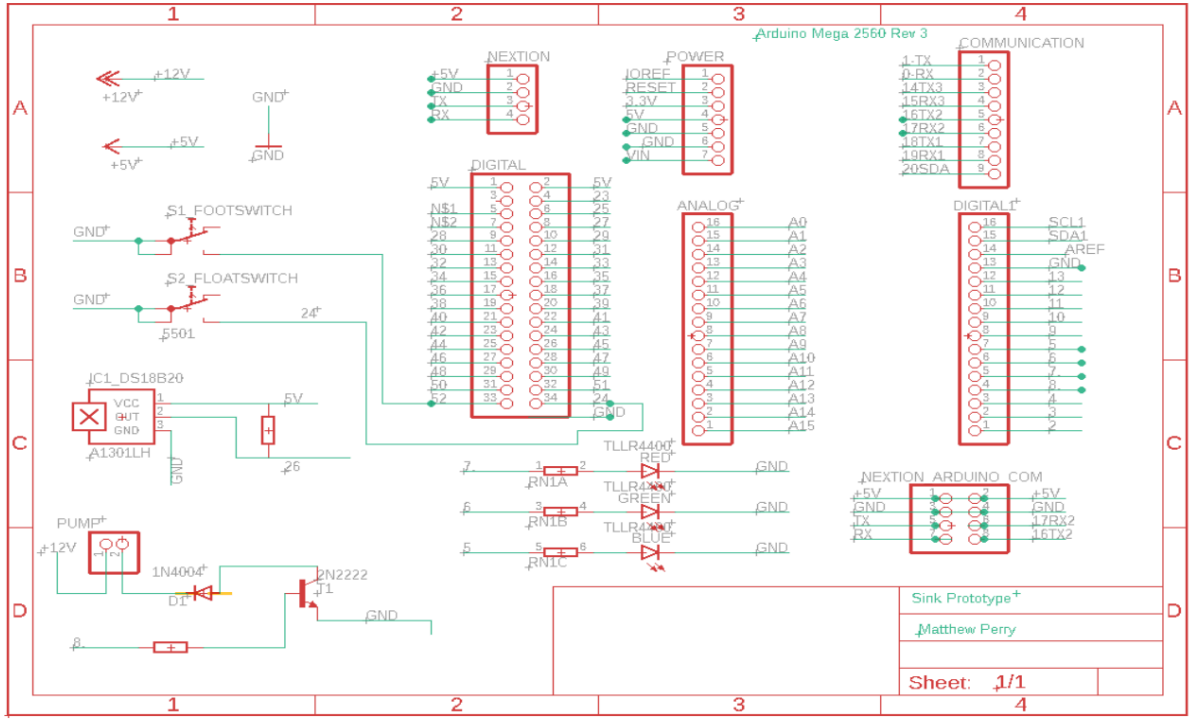


Figure 11; Eagle Electrical

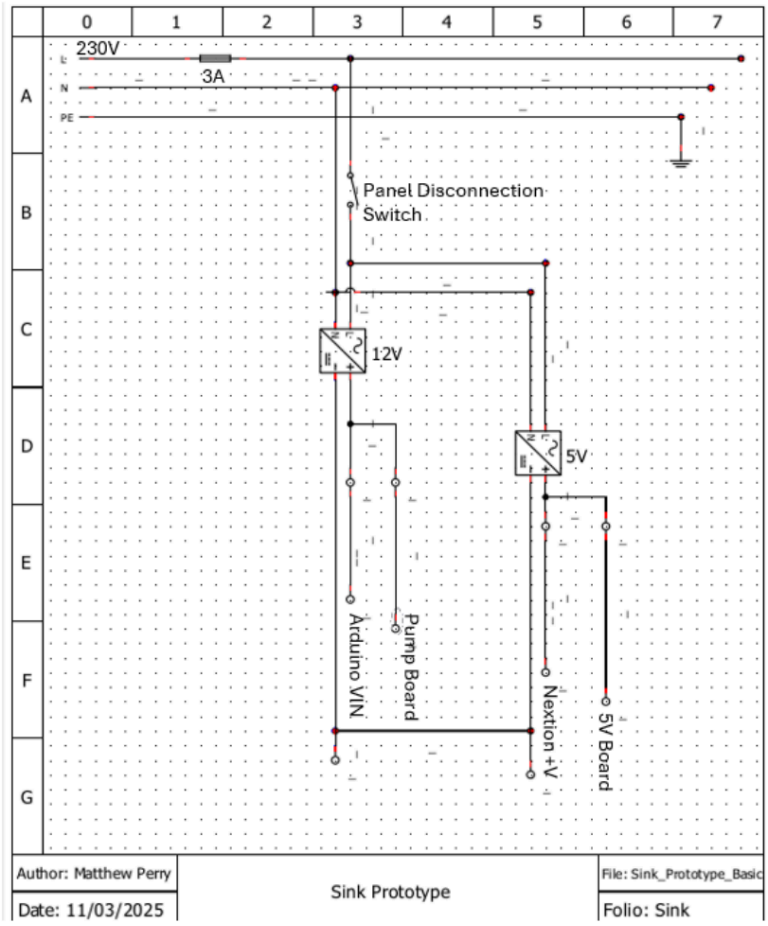


Figure 12; Electrical Diagram

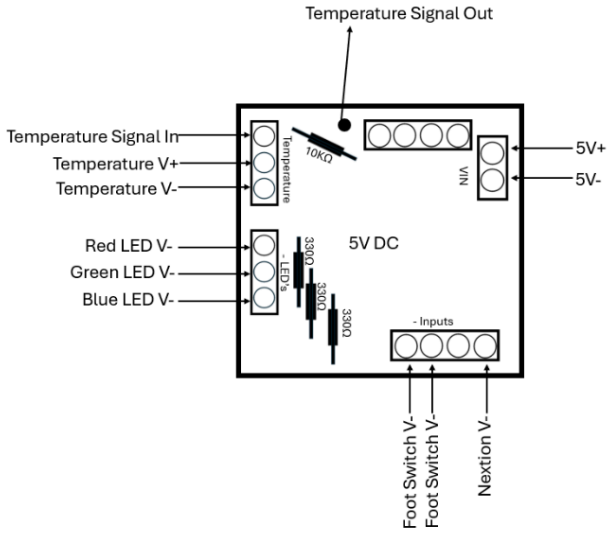


Figure 14;5V Perfboard Layout

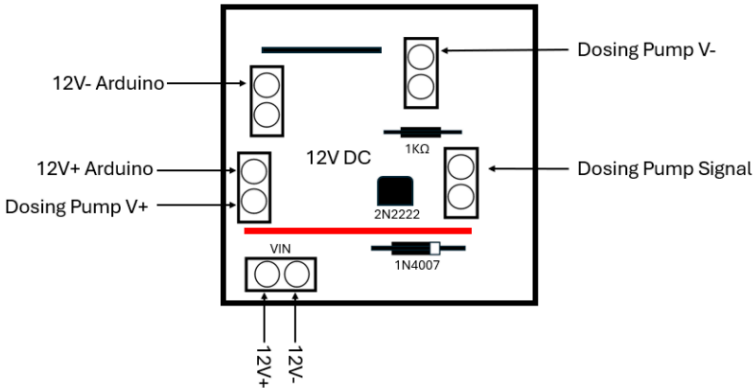


Figure 13;12V Perfboard Layout

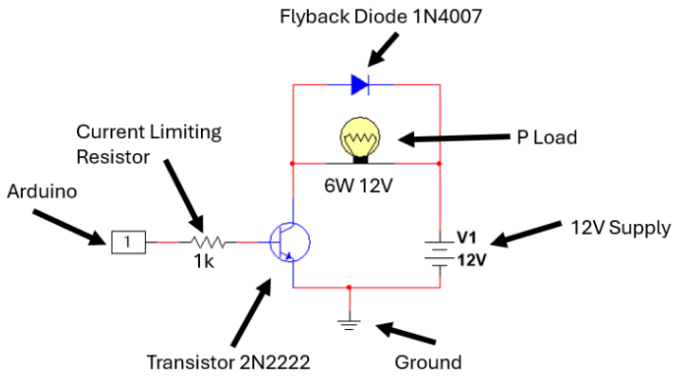


Figure 15; Transistor Circuit Layout

Power Supply					
Component	Voltage	Fuse	Current Draw	Power	Load
Main Fuse	230VAC	3A	-	-	-
Power Supply	12VDC	-	1.25A Max	15W	0.5A
Power Supply	5VDC	-	3A Max	15W	0.67A

Table 7; Power Supply

Component Load Schedule			
Component	Voltage (VDC)	Est Current Draw (A)	Power (W)
Arduino Mega 2560	12V	0.2A	2.4W
Nextion Intelligent 4.3"	5V	0.65A	2.5W
3 x 5V LED (w/330 ohm resistor)	5V	0.027A	0.135W
Footswitch (digital input)	5V	-	-
Float Switch (digital input)	5V	-	-
DS18B20 Temp Sensor	5V	-	-
12V Dosing Pump	12V	0.3A	3.6W

Table 6; Component Load Schedule

Electrical Colour Codes		
230VAC		
Live	Brown	
Neutral	Blue	
Earth	Green &	Yellow
12V & 5VDC		
Positive	Red	
Negative	Black	
Serial Communication		
TX	Yellow	
RX	Blue	

Table 8; Electrical Colour Codes

Program Structure

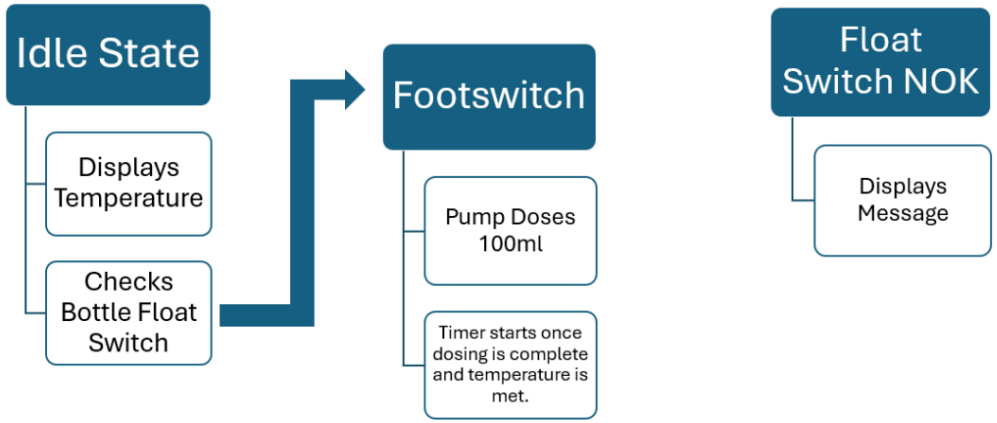


Figure 16; Software Program Structure

Nextion Components				
Page	Object	ID	Object Name	Comment
0	Image	1	p0	Background image
0	Text	4	t_temp	Temperature reading
0	Text	2	t0_button	Next Page if "111" or Settings "222"
0	Text	6	t_remaining	Time remaining
0	Text	13	floatind	Text box for Arduino to insert when chemical is low
0	Picture	11	warning	Incorrect password inserted notifier
0	Picture	10	hide	Hide settings when password is not inputted
0	Picture	12	floatImg	Replace chemical notifier
0	Button	5	b_time_start	Start timer countdown
0	Text	8	t_offset	Temperature offset input
0	Text	7	t_amount	Timer amount
0	Text	9	t_dosetime	Timer for dosing time
1	Image	1	p0	Background image
1	Button	3	b_red	Red LED manual
1	Button	4	b_blue	Blue LED manual
1	Button	5	b_green	Green LED manual
1	Button	6	b_pump	Button for pump manual
0	Dual Button	14	time_vis_b	Button to turn timer display on/off
0	Picture	13	hidetimer	Picture to hide timers
1	Button	7	b_manual	Button to start 15 second manual mode
Global 0	Timer 2 seconds	12	timernotif	Check time_vis_b for 0 or 1 to display timers
Global 0	Timer 50ms	2	tbutton	Timer to check 111 or 222 for t0_button
Global 0	Timer 500ms	14	float	Check floatImg for 1 or 0 then vis command

Table 9; Nextion Components Addressing

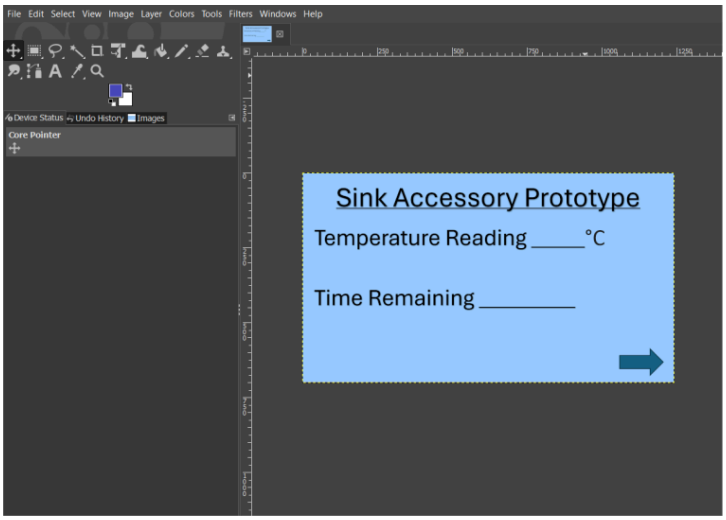


Figure 17; Gimp Example

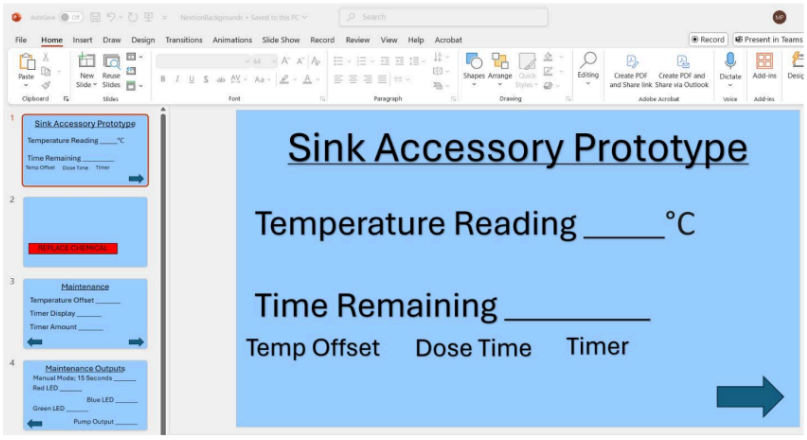


Figure 18; Powerpoint Nextion Backgrounds

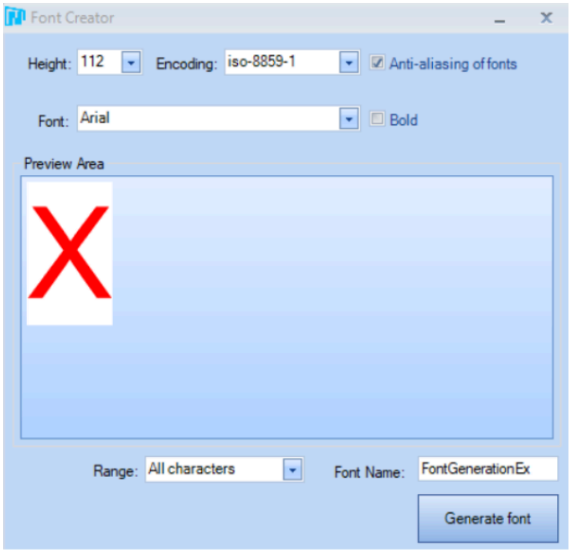


Figure 19; Nextion Font Generation

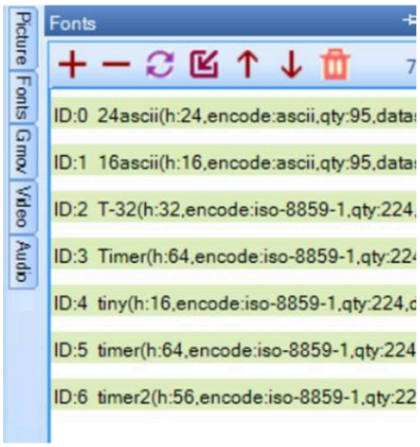


Figure 20; Nextion Font List

Display | Program.s ×

```
//The following code is only run once when power on, and is generally used for global variable definition and power on initialization data
int sys0=0,sys1=0,sys2=0, time_vis=0   //At present, the definition of global variable only supports 4-byte signed integer (int), and other types of global quantity declaration are not
baud=9600//Configure baudrate
dim=100//Configure backlight
recmod=0//Serial data parsing mode:0-Passive mode;1-Active mode
printh 00 00 00 ff ff ff 88 ff ff ff//Output power on information to serial port
page 0                  //Power on start page 0
```

Figure 21; Nextion program.s

```
//Libraries
#include <OneWire.h> // Temperature sensor
#include <DallasTemperature.h> // Temperature sensor
#include <Nextion.h> // Communication for Nextion

// Pin definitions
#define ONE_WIRE_BUS 50 // Temp sensor
const int ledLow = 5; //Blue
const int ledMedium = 6; // Green
const int ledHigh = 7; // Red
const int pumpPin = 8; //Pump signal
const int footSwitchPin = 52; // Footswitch
const int floatSwitchPin = 53; // Floatswitch

// Initialise libraries
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

// Nextion objects where objectname page number, ID
NexText t_temp = NexText(0, 4, "t_temp"); // t_temp (object name), 0 (page
number), 4 (ID)
NexNumber t_offset = NexNumber(0, 8, "t_offset");
NexNumber t_amount = NexNumber(0, 7, "t_amount");
NexText t_remaining = NexText(0, 6, "t_remaining");
NexNumber t_dosetime = NexNumber(0, 9, "t_dosetime");

NexButton b_start = NexButton(0, 5, "b_start");
NexButton b_manual = NexButton(1, 7, "b_manual");
NexButton b_red = NexButton(1, 3, "b_red");
NexButton b_green = NexButton(1, 5, "b_green");
NexButton b_blue = NexButton(1, 4, "b_blue");
NexButton b_pump = NexButton(1, 6, "b_pump");

// State variables tracking manual mode
bool manualMode = false;
unsigned long manualModeStartTime = 0;
unsigned long countdownStartTime = 0;
unsigned long countdownDuration = 0;
bool countdownActive = false;

// Dosing timer and duration
bool doseActive = false;
unsigned long doseStartTime = 0;
unsigned long doseDuration = 0;
```

Figure 22; Arduino Code 1

```
// Temperature offset for Nextion
float tempOffset = 0.0; //Store

// Float switch indicator
NexText floatindText = NexText(0, 14, "floatind"); // Notification on Nextion
unsigned long lastFloatUpdateTime = 0;

// Footswitch edge detection
bool lastFootSwitchState = HIGH; // Remember previous state

// Temperature
void updateTemperature() {
    sensors.requestTemperatures(); // Request
    float temperature = sensors.getTempCByIndex(0);

    Serial.print("Raw Temperature: ");
    Serial.println(temperature);

    long offsetValue;
    if (t_offset.getValue(&offsetValue)) {
        tempOffset = (float)offsetValue; // Read offset from nextion convert
to float
    }

    Serial.print("Temperature Offset: ");
    Serial.println(tempOffset);

    if (temperature == DEVICE_DISCONNECTED_C) {
        Serial.println("Error: Sensor not connected");
        t_temp.setText("Error"); // If fail to read then display Error
        return;
    }

    temperature += tempOffset; // Apply offset to raw sensor data

    Serial.print("Adjusted Temperature: ");
    Serial.println(temperature);

    char tempStr[10];
    dtostrf(temperature, 4, 2, tempStr); // Apply to nextion (4 width 2
decimal)
    t_temp.setText(tempStr);

    if (!manualMode) {
        controlLEDs(temperature); // Automatic mode if not in manual mode for
LED
    }
}
```

Figure 23; Arduino Code 2

```
//LEDs
void controlLEDs(float temperature) {
    const float lowThreshold = 23.0; // Thresholds
    const float highThreshold = 32.0;

    digitalWrite(ledLow, LOW); // Set all to low
    digitalWrite(ledMedium, LOW);
    digitalWrite(ledHigh, LOW);

    if (temperature < lowThreshold) {
        digitalWrite(ledLow, HIGH);
    } else if (temperature <= highThreshold) {
        digitalWrite(ledMedium, HIGH);
    } else {
        digitalWrite(ledHigh, HIGH); //Logic for LEDs
    }
}
// Start countdown for timer on Nextion
void startCountdown() {
    if (!countdownActive) {
        uint32_t timeValue;
        if (t_amount.getValue(&timeValue)) {
            if (timeValue > 0) {
                countdownDuration = timeValue * 1000;
                countdownStartTime = millis();
                countdownActive = true;
                Serial.print("Countdown started for ");
                Serial.print(timeValue);
                Serial.println(" seconds"); // Serial monitor debugging
            }
        }
    }
}
```

Figure 24; Arduino Code 3

```
// Check for touch events (7 bytes 0x65)
void checkNextion() {
    if (Serial2.available() >= 7) {
        uint8_t buffer[7];
        Serial2.readBytes(buffer, 7);

        if (buffer[0] == 0x65) {
            uint8_t pageID = buffer[1];
            uint8_t componentID = buffer[2];
            uint8_t eventType = buffer[3];

            if (pageID == 0 && componentID == 5 && eventType == 1) {
                startCountdown();
            }

            if (pageID == 1 && componentID == 7 && eventType == 1) {
                manualMode = true;
                manualModeStartTime = millis();
                Serial.println("Manual Mode ON");
            }
// Manual control buttons
            if (manualMode) {
                if (pageID == 1) {
                    if (componentID == 3) {
                        digitalWrite(ledHigh, eventType);
                        Serial.println(eventType ? "Red LED ON" : "Red LED
OFF");
                    } else if (componentID == 5) {
                        digitalWrite(ledMedium, eventType);
                        Serial.println(eventType ? "Green LED ON" : "Green LED
OFF");
                    } else if (componentID == 4) {
                        digitalWrite(ledLow, eventType);
                        Serial.println(eventType ? "Blue LED ON" : "Blue LED
OFF");
                    } else if (componentID == 6) {
                        digitalWrite(pumpPin, eventType);
                        Serial.println(eventType ? "Pump ON" : "Pump OFF");
                    }
                }
            }
        }
    }
}
```

Figure 25; Arduino Code 4

```
// Dosing logic
void checkDose() {
    bool currentFootSwitchState = digitalRead(footSwitchPin);
    bool floatSwitchState = digitalRead(floatSwitchPin); // LOW = unsafe
because no liquid

    // Trigger on falling edge & float switch must be HIGH (safe)
    if (!doseActive && lastFootSwitchState == HIGH && currentFootSwitchState
== LOW) {
        if (floatSwitchState == HIGH) {
            uint32_t doseTimeValue;
            if (t_dosetime.getValue(&doseTimeValue)) {
                if (doseTimeValue > 0) {
                    doseDuration = doseTimeValue * 1000;
                    doseStartTime = millis(); // How long to run
                    doseActive = true;
                    digitalWrite(pumpPin, HIGH);
                    Serial.print("Dose started for ");
                    Serial.print(doseTimeValue);
                    Serial.println(" seconds");
                }
            }
        } else {
            Serial.println("Dose Stopped: Float switch in unsafe state.");
        }
    }

    // During dosing stop if time is up or float switch becomes unsafe
    if (doseActive) {
        if ((millis() - doseStartTime >= doseDuration) ||
digitalRead(floatSwitchPin) == HIGH) {
            doseActive = false;
            digitalWrite(pumpPin, LOW);
            Serial.println("Dose stopped (completed or float switch
unsafe).");
        }
    }

    lastFootSwitchState = currentFootSwitchState;
}

void setup() {
    Serial.begin(115200); // Serial monitor
    Serial2.begin(9600); // Nextion where numbers are Baud rates
```

Figure 26; Arduino Code 5

```
// Pin modes
    pinMode(ledLow, OUTPUT);
    pinMode(ledMedium, OUTPUT);
    pinMode(ledHigh, OUTPUT);
    pinMode(pumpPin, OUTPUT);
    pinMode(footSwitchPin, INPUT_PULLUP);
    pinMode(floatSwitchPin, INPUT_PULLUP);


//Initialise
    nexInit();
    sensors.begin();
    t_temp.setText(" ... ");

    Serial.println("Setup complete.");
}

void loop() {
    checkNextion();

    static unsigned long lastUpdate = 0;
    if (millis() - lastUpdate > 5000) { // Update every 5 sec
        updateTemperature();
        lastUpdate = millis();
    }

    // Update float switch displays every 5 sec
    if (millis() - lastFloatUpdateTime >= 5000) {
        if (digitalRead(floatSwitchPin) == LOW) {
            floatindText.setText("1");
        } else {
            floatindText.setText("0"); // sending number to handle Vis command
in Nextion
        }
        lastFloatUpdateTime = millis();
    }

    if (manualMode && (millis() - manualModeStartTime >= 15000)) {
        manualMode = false; // disable manual mode 15 sec
        Serial.println("Manual Mode OFF (auto-reset)");
    }
```

Figure 27; Arduino Code 6

```
// Countdown timer logic
    if (countdownActive) {
        unsigned long elapsedTime = millis() - countdownStartTime; // How much
time since start
        unsigned long remainingTime = (countdownDuration > elapsedTime) ?
(countdownDuration - elapsedTime) : 0;

        if (remainingTime == 0) {
            countdownActive = false;
            t_remaining.setText("00:00");
            Serial.println("Countdown finished"); // stop countdown show 0000
        } else {
            char remainingStr[10];
            unsigned long minutes = remainingTime / 60000;
            unsigned long seconds = (remainingTime % 60000) / 1000;
            snprintf(remainingStr, sizeof(remainingStr), "%02lu:%02lu",
minutes, seconds);
            t_remaining.setText(remainingStr); // convert ms to MM:SS
        }
    }

    checkDose(); // Check footswitch, float switch and start stop pump
}
```

Figure 28; Arduino Code 7

```
void checkDose() {
    bool currentFootSwitchState = digitalRead(footSwitchPin);
    bool floatSwitchState = digitalRead(floatSwitchPin); // High = unsafe because no liquid

    // Trigger on falling edge & float switch must be Low (safe)
    if (!doseActive && lastFootSwitchState == HIGH && currentFootSwitchState == LOW) {
        if (floatSwitchState == LOW) {
            uint32_t doseTimeValue;
            if (t_dosetime.getValue(&doseTimeValue)) {
                if (doseTimeValue > 0) {
                    doseDuration = doseTimeValue * 1000;
                    doseStartTime = millis(); // How long to run
                    doseActive = true;
                    digitalWrite(pumpPin, HIGH);
                    Serial.print("Dose started for ");
                    Serial.print(doseTimeValue);
                    Serial.println(" seconds");
                }
            }
        } else {
            Serial.println("Dose Stopped: Float switch in unsafe state.");
        }
```

Figure 29; Arduino Issue Before Adjustment

```
void checkDose() {
    bool currentFootSwitchState = digitalRead(footSwitchPin);
    bool floatSwitchState = digitalRead(floatSwitchPin); // LOW = unsafe because no liquid

    // Trigger on falling edge & float switch must be HIGH (safe)
    if (!doseActive && lastFootSwitchState == HIGH && currentFootSwitchState == LOW) {
        if (floatSwitchState == HIGH) { // liquid available
```

Figure 30; Resolved Arduino Issue

9.6 Risk Assessment

Risk Assessment					
What are the hazards?	Who might be harmed and how?	What are you already doing to control the risks?	What further action do you need to take to control the risks?	Who needs to carry out the action?	When is the action needed by?
Slips and trips	MP or others in the same area – could fall and injure themselves.	Ensure floor space is clear, no trailing leads and ensure spills are cleaned immediately.	Add signage, ensure if there are trailing leads to make them noticeable and ensure the correct footwear is worn.	MP	At the start of each session of work.
Moving heavy objects	MP – may strain back or drop items causing injuries.	Use proper lifting techniques or ask for assistance with larger items.	Invest in a trolley or lifting aid for larger items. Use steel toe caps.	MP	Before the items are moved.
Cuts & grazes	MP - using tools or fitting some components.	Wear gloves and overalls and use appropriate tools for the correct job.	Ensure safe handling of tools, check tools for damage and ensure first aid kit is readily available.	MP	When hand tools are required.
Flying objects	MP or bystanders – debris from drilling or cutting may impact eyes.	Use safety glasses, ensure material is set in place before drilling or cutting.	Ensure bystanders aren't in the vicinity when working, use shields.	MP	Before use of tools.

Risk Assessment					
What are the hazards?	Who might be harmed and how?	What are you already doing to control the risks?	What further action do you need to take to control the risks?	Who needs to carry out the action?	When is the action needed by?
Dust and Fumes	MP or bystanders – cutting Perspex or soldering.	Work should be completed in a well-ventilated area or wear a mask while cutting. Ventilation fans with a filter (charcoal) should be used to remove solder fumes from workspace.	If possible, use a FFP3 mask.	MP	While cutting materials soldering or any dust production.
Electrocution	MP or Users – Working on electrical circuits or under fault condition.	Ensure power is off during assembly, use insulated tools and ensure by standers don't have access to the panel under fault condition.	Use safety interlocks such as panel switches to disconnect power. Ensure that the panel cannot be energised if not safe.	MP	Before energising.
Power tools	MP - Cuts, abrasions, shock, limbs.	Only trained personnel to use the power tools, use correct PPE that's required and inspect tools before use.	Maintain tools regularly and ensure competency.	MP	Weekly and before each use.

Table 10; Risk Assessment

9. 7 Practical Implementation

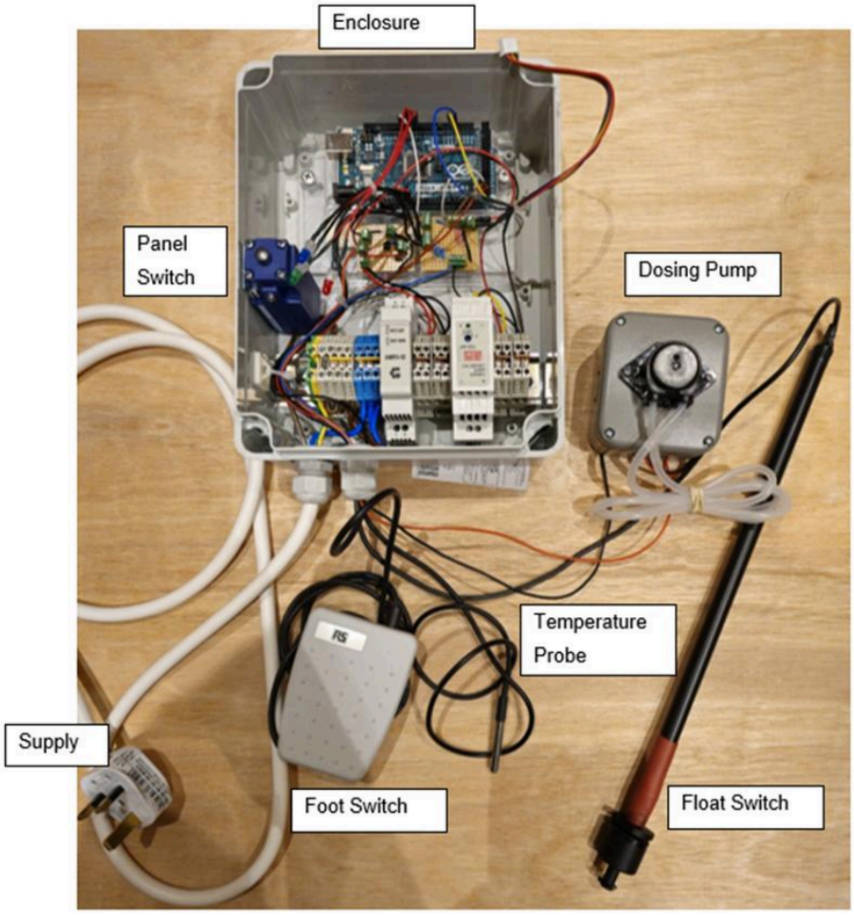


Figure 31; Completed Construction

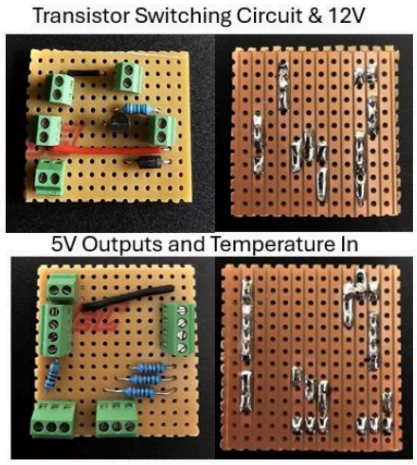


Figure 32; Perf Boards

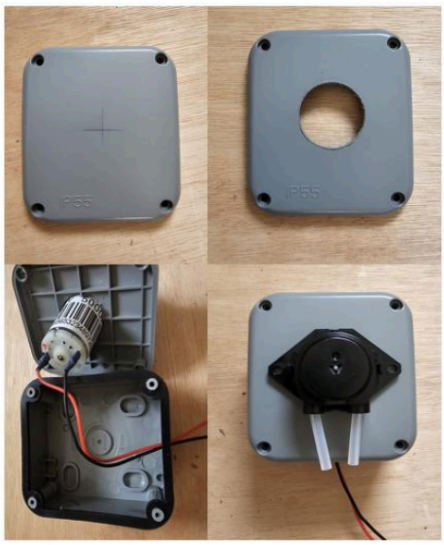


Figure 33; Motor Enclosure

Soldering LED wiring looms



Figure 35; Soldering LED Wiring Looms

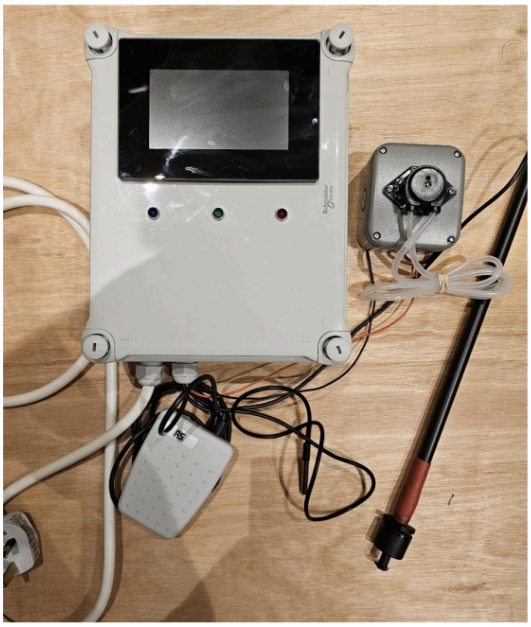


Figure 34; Completed Construction Front

9.8 Simulation

Arduino Simulation (Serial Monitor)	
Initialising	□□□ recvRetCommandFinished ok recvRetCommandFinished ok Setup complete.
Temperature	Raw Temperature: 19.06 recvRetNumber :0 Temperature Offset: 0.00 Adjusted Temperature: 19.06
Temperature with offset +1	Raw Temperature: 19.06 recvRetNumber :1 Temperature Offset: 1.00 Adjusted Temperature: 20.06
Temperature with offset -1	Raw Temperature: 19.06 recvRetNumber :4294967295 Temperature Offset: -1.00 Adjusted Temperature: 18.06
Insert dose time and start, include interlock.	recvRetNumber :120 Dose started for 120 seconds Dose stopped (completed or float switch unsafe)
Manual mode	Manual Mode ON Manual Mode OFF (auto-reset)

Outputs	Red LED ON Red LED OFF Blue LED ON Blue LED OFF Green LED ON Green LED OFF Pump ON
Countdown for 10 seconds.	recvRetNumber :10 Countdown started for 10 seconds recvRetCommandFinished ok recvRetCommandFinished ok Countdown finished

Table 11; Arduino Simulation

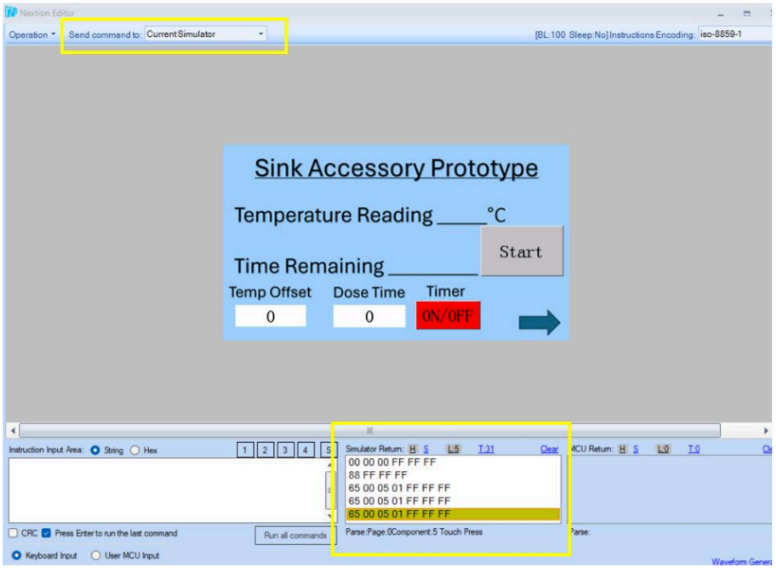


Figure 36; Nextion Editor Simulator

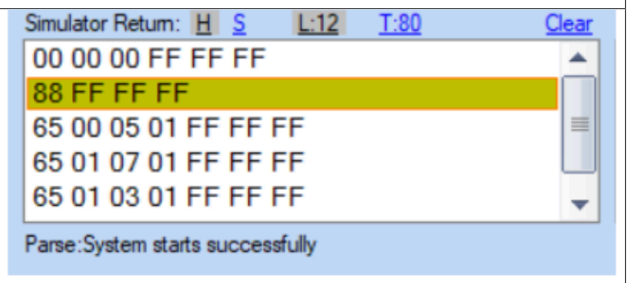
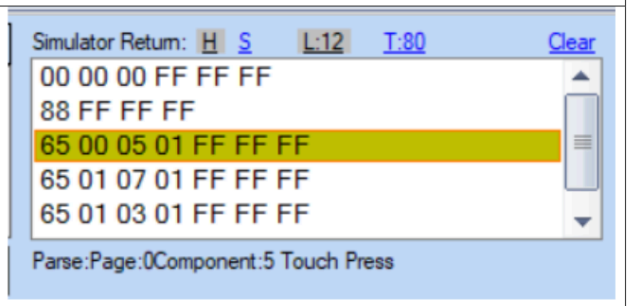
Nextion Editor Simulator	
System Start	
Example of button press.	

Table 12; Nextion Editor Simulator

9.9 Testing



Figure 37; Temperature and LED Evidence

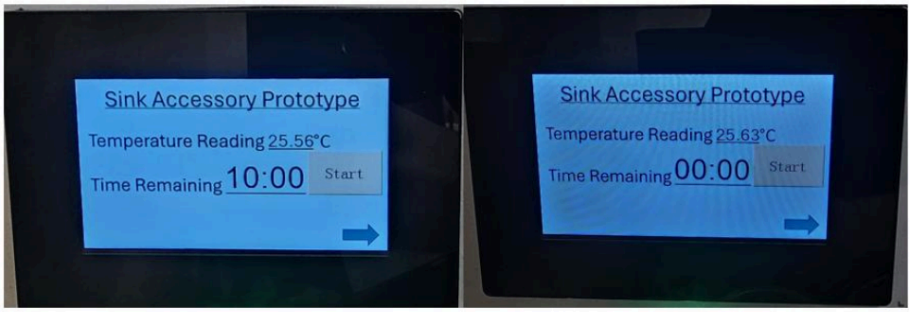


Figure 39; Wash Time Evidence



Figure 38; Dosing Time Configuration Evidence

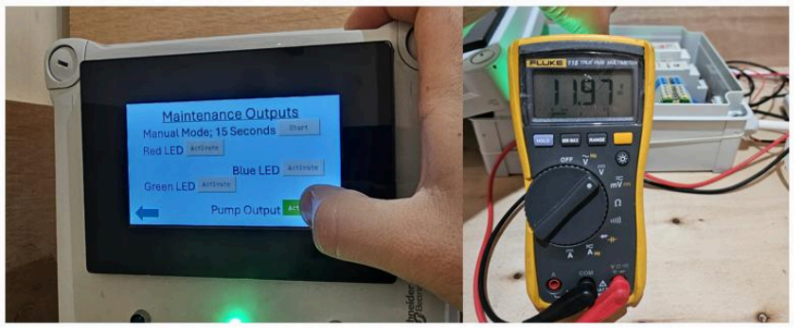


Figure 40; Manual Purge Evidence



Figure 41; Wash Time Removable Evidence

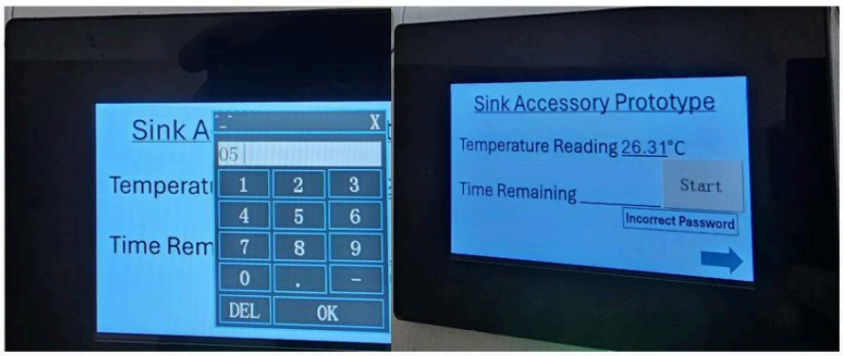


Figure 43; Password Protection Evidence



Figure 42; 0V Evidence



Figure 44; Replace Chemical Evidence

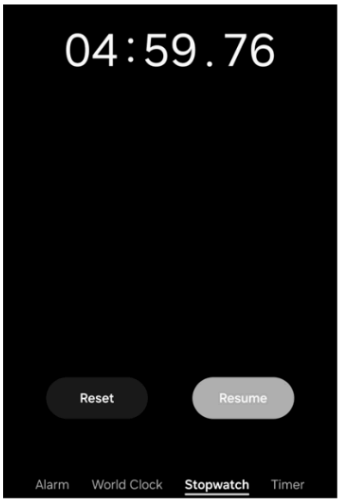


Figure 45; Timer Comparison Evidence

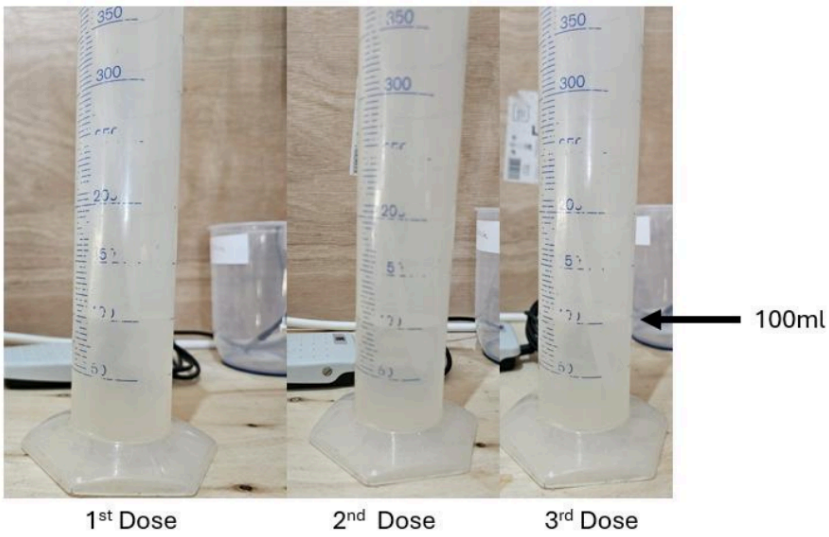


Figure 46; Dosing Evidence

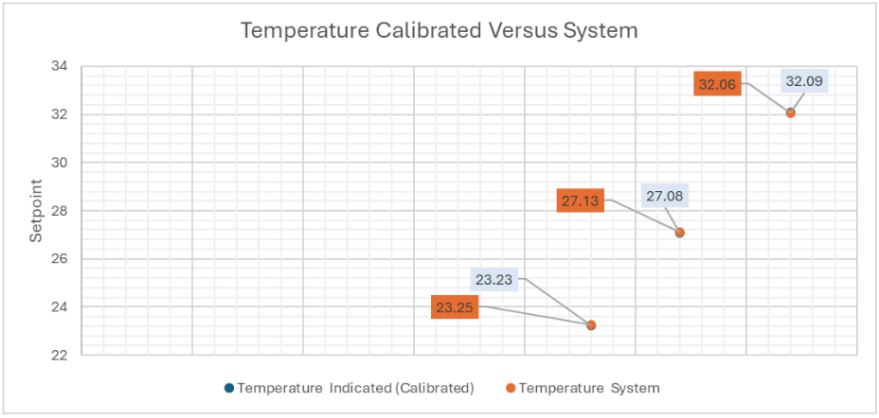


Figure 47; Temperature Testing Graph

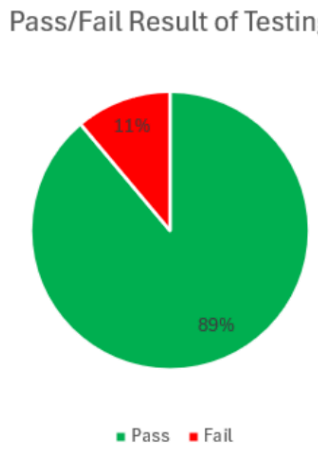


Figure 48; Pie Chart Success Rate

SGEL6000

Independent Project

Logbook

Matthew Perry

Prototype Sink Accessory

2024 - 2025

Summary of Entries

Log Entry	Date	Brief Description of Activity	Duration (hrs)	Accumulated time (hrs)
1	20-09-24	Considering Independent Project Ideas	2	2
2	27-09-24	Discussing Independent Project	2	4
3	04-10-24	Considering New Idea (Project Proposal)	2	6
4	11-10-24	Reading Guidance, briefly investigating 3 methods	2	8
5	18-10-24	Reading relevant standards and task 4	4	12
6	25-10-24	Completing task 4	6	18
7	01-11-24	Half-term, deciding what to include in FDS	2	20
8	08-11-24	Completing task 5	4	24
9	15-11-24	Amending task 6	4	28
10	22-11-24	Research into 3 Solutions	7	35
11	29-11-24	Completing 3 solutions	8	42
12	06-12-24	Completed literature review, WBS,Gantt	8	50
13	13-12-24	Checking assignment	8	58
14	20-12-24	Creating project presentation / Parts	10	68
15	27-12-24	Researching software	10	78
16	03-01-25	Checked parts / Updated list	6	84
17	10-01-25	Amending presentation, Research programming	8	92
18	17-01-25	Project presentation, Programming test 1	13	105
19	24-01-25	Reading assignment brief, Programming test 2	16	121
20	31-01-25	Hardware, Re-order parts	7	128
21	07-02-25	Computer Issue, Eagle Investigation, HMI mock	10	138
22	14-02-25	Eagle, Word Perfboard, Soldering	12	150
23	21-02-25	New Enclosure, Layout Drawing Word	12	162
24	28-02-25	Hardware Construction	15	177
25	07-03-25	Hardware Construction	10	187
26	14-03-25	Confirmation Software/ Sim	12	199
27	21-03-25	Software Development	17	216
28	28-03-25	Software Development	16	232
29	04-04-25	Software Development	18	250
30	11-04-25	Finish Software Development and Sim	20	270
31	18-04-25	Compile Software Pictures, Amendments	8	278

32	25-04-25	Testing, Issue with software	10	288
33	02-05-25	Completing Assignment	22	310

Log Entry Sheet

Log Number	16
Date	03-01-2025
Duration (hrs)	6
Accumulated time (hrs)	84

Record of activities and achievements

Re-checked parts list for price fluctuation and to ensure they are in stock.
12VDC peristaltic pump was out of stock now, sourced new part and updated parts list.
Put in request for parts to be bought.

Added more parts to the list, updated URL links.

Plans for next week

Order parts list once it's been confirmed again by manager, practice presentation.

Signed: 

Signed: R. Davies

Log Entry Sheet

Log Number	17
Date	10-01-2025
Duration (hrs)	8
Accumulated time (hrs)	92

Record of activities and achievements

Amended project presentation after discussion in class. Removed style template and the number of words. Redesigned with more pictures to prompt what to talk about. Practiced presentation on what to talk about on each slide.

First order of parts list approved and ordered on 09/01/2025 , further parts I requested still waiting.

Updated parts log such as order dates.

Looked at open-source software tutorial videos such as eagle, QElectrotech for design schematics.

Plans for next week

Present presentation.

Await and check delivery.

Once confirmed order parts.

Signed: 

Signed: R. Davies

Log Entry Sheet

Log Number	18
Date	17-01-2025
Duration (hrs)	13
Accumulated time (hrs)	105

Record of activities and achievements

Presented presentation.

All parts have been delivered except for the Nextion Intelligent HMI. Checked to see parts arrived are correct and not damaged; updated parts list.

Prototype 1

Due to being unfamiliar with the Arduino programming style, I carried out a tutorial to familiarise myself with the software and hardware. Once completed and working, I amended the program so it would be more beneficial to the completed project.

This consisted of x3 LED, resistors, Arduino Mega 2560, breadboard, DS18B20 temperature sensor.

The temperature sensor is read by the Arduino, and dependent on the temperature either a cold, OK or hot LED will indicate. Temperature will be displayed on the serial monitor to verify its operation.

This was a beneficial tutorial because it included how to add libraries for sensors to communicate with the Arduino, while showing basic commands, overall layout of a program and debugging techniques.

Bsc3_Example1.ino

```
  1  #include <OneWire.h> // Temperature Sensor Library
 58   // Print temperatures to Serial Monitor
 59   Serial.print("Raw Celsius: ");
 60   Serial.print(rawTempC);
 61   Serial.print(" | Calibrated Celsius: ");
 62   Serial.println(calibratedTempC);
 63
 64   // Control LEDs based on temperature thresholds
 65   if (calibratedTempC < 20.0) {
 66     // Cold temperature
 67     digitalWrite(COLD_LED, HIGH);
 68     digitalWrite(OK_LED, LOW);
 69     digitalWrite(HOT_LED, LOW);
 70   } else if (calibratedTempC >= 20.0 && calibratedTempC <= 32.0) {
 71     // OK working temperature
 72     digitalWrite(COLD_LED, LOW);
 73     digitalWrite(OK_LED, HIGH);
 74     digitalWrite(HOT_LED, LOW);
 75   } else if (calibratedTempC > 32.0) {
 76     // Hot temperature
 77     digitalWrite(COLD_LED, LOW);
 78     digitalWrite(OK_LED, LOW);
 79     digitalWrite(HOT_LED, HIGH);
 80   }
 81
 82   // Wait before the next reading
 83   delay(1000);
 84 }
 85
```

What was shown on Serial Monitor

OK

```
17:37:53.425 -> Raw Celsius: 21.69 | Calibrated Celsius: 22.62
17:37:54.915 -> Raw Celsius: 21.69 | Calibrated Celsius: 22.62
17:37:56.433 -> Raw Celsius: 21.69 | Calibrated Celsius: 22.62
```

Cold

```
17:39:00.940 -> Raw Celsius: 15.75 | Calibrated Celsius: 16.56
17:39:02.443 -> Raw Celsius: 15.75 | Calibrated Celsius: 16.56
17:39:03.966 -> Raw Celsius: 15.75 | Calibrated Celsius: 16.56
17:39:05.425 -> Raw Celsius: 15.69 | Calibrated Celsius: 16.50
17:39:06.940 -> Raw Celsius: 15.69 | Calibrated Celsius: 16.50
```

Hot

```
17:41:22.014 -> Raw Celsius: 34.44 | Calibrated Celsius: 35.63
17:41:23.521 -> Raw Celsius: 34.44 | Calibrated Celsius: 35.63
17:41:25.031 -> Raw Celsius: 34.44 | Calibrated Celsius: 35.63
17:41:26.535 -> Raw Celsius: 34.44 | Calibrated Celsius: 35.63
17:41:28.034 -> Raw Celsius: 34.44 | Calibrated Celsius: 35.63
```

Plans for next week

Start another example which includes the communication between the Arduino & Nextion. Update parts list once Nextion is delivered.

Signed : 

Signed: *R. Davies*

Log Entry Sheet

Log Number	19
Date	24-01-2025
Duration (hrs)	16
Accumulated time (hrs)	121

Record of activities and achievements

Nextion delivered, updated parts list.

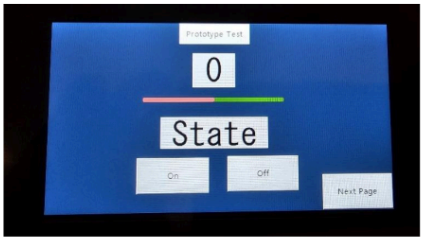
Prototype 2

Followed tutorial on basic Arduino program and communication with Nextion HMI.

This consisted of using the Arduino IDE, Nextion Editor and GIMP (Image manipulation program). GIMP was used to create images to load onto the Nextion Editor software due to it not having a drag and drop functions for backgrounds or aesthetic buttons.

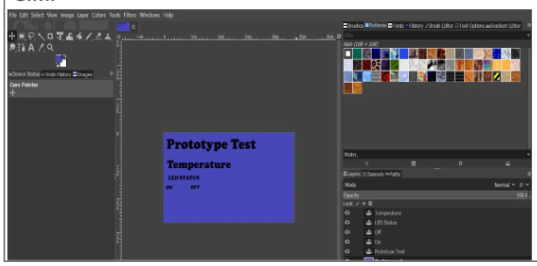
A large portion of this time was spent debugging the communication between hardware devices, including poor grounding on the breadboard which then affected the TX & RX communications.

The example created consisted of two buttons (ON/OFF) on the HMI that switched an LED, and the temperature sensor reading displayed on the HMI.



Then amended it to try it differently again.

GIMP



Arduino IDE

Bsc3_Example3.ino

```
#include "Nextion.h"
#include <OneWire.h>
#include <DallasTemperature.h>

// Define DS18B20 sensor pin
#define ONE_WIRE_BUS 4

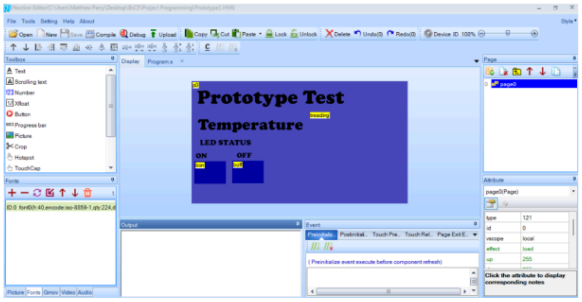
// Initialize OneWire and DallasTemperature libraries
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

// LED pins
const int led1 = 8;
const int led2 = 9;

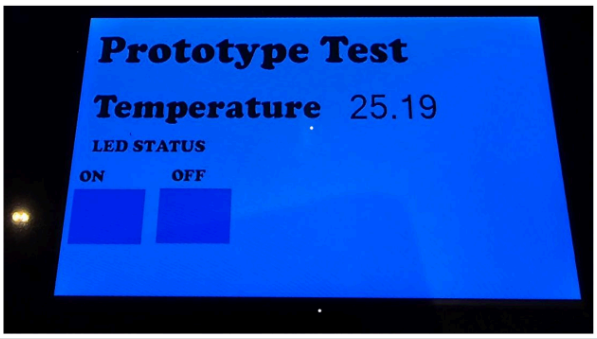
// Nextion objects
NexText tstate = NexText(0, 3, "tstate");
NexButton bon = NexButton(0, 6, "bon");
NexButton boff = NexButton(0, 7, "boff");
NexSlider slider = NexSlider(0, 5, "slider");
NexText tslider = NexText(0, 4, "tslider");
NexText treading = NexText(1, 4, "treading");
NexButton bupdate = NexButton(1, 6, "bupdate");

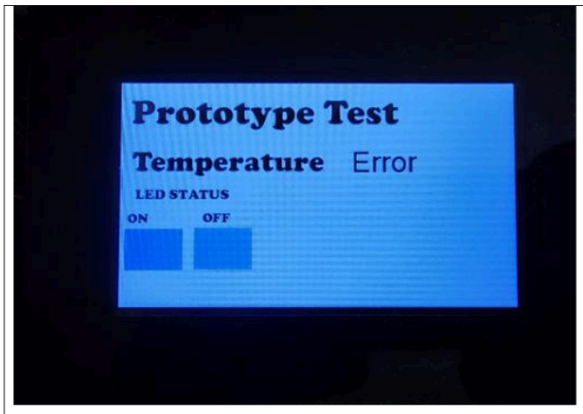
```

Nextion Editor



Physical

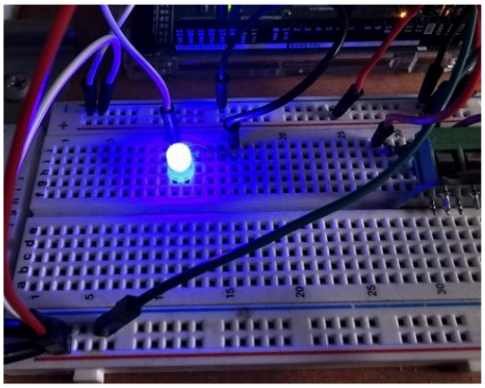




Output Serial Monitor ×

Message (Enter to send message to 'Arduino Mega or Mega 2560' on 'COM4')

```
16:46:16.283 -> recvRetCommandFinished ok
16:46:16.317 -> Temperature updated on Nextion.
16:46:21.802 -> Temperature: 29.75
16:46:21.836 -> recvRetCommandFinished ok
16:46:21.837 -> Temperature updated on Nextion.
16:46:27.317 -> Temperature: 29.56
16:46:27.454 -> recvRetCommandFinished ok
16:46:27.454 -> Temperature updated on Nextion.
```



This was a good example to follow as it allowed me to learn addressing the Nextion components in Arduino, GIMP and general Nextion Editor navigation. This has built a good foundation for my software design stage.

Plans for next week

Hardware design. Design schematics from what was learnt before, figure out a way to design a panel layout.

Signed : 

Signed: R. Davies

Log Entry Sheet

Log Number	20
Date	31-01-2025
Duration (hrs)	7
Accumulated time (hrs)	128

Record of activities and achievements

Although some hardware has already been connected due to carrying out the prototype tests, there has been issues with the breadboard and DuPont connections. This resulted in a poor connection with the temperature sensor which then led to it being damaged and faulty. Investigation into better ways to connect the 5V connections is needed.

Re-ordered temperature sensor and updated parts list. Benefit is temperature sensor is low cost and last delivered within a couple of days.

Plans for next week

Hardware design.

Signed: 

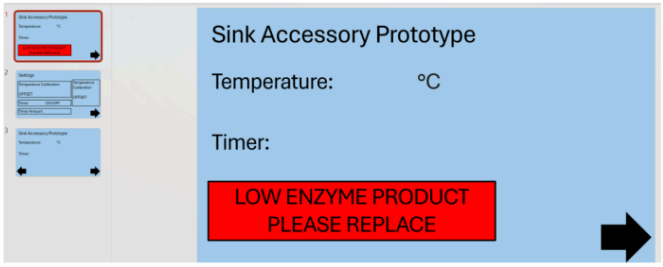
Signed: R. Davies

Log Entry Sheet

Log Number	21
Date	07-02-2025
Duration (hrs)	10
Accumulated time (hrs)	138

Record of activities and achievements

HMI mock up for GIMP software in spare time.



Personal laptop that I use to complete project work faulty, sent for repair.

Investigated how to use Eagle software such as the toolbox and navigation to create a circuit diagram for Arduino side.

Created other design and supporting information, e.g load schedules

Plans for next week

Hardware design. Gather tools and PPE.

Revisit enclosure, although there is one that I currently have on site to reduce business cost and possible waste, it would be better if it was smaller.

Signed: 

Signed: R. Davies

Log Entry Sheet

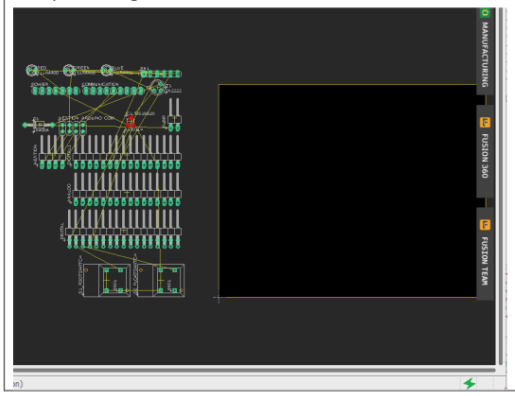
Log Number	22
Date	14-02-2025
Duration (hrs)	12
Accumulated time (hrs)	150

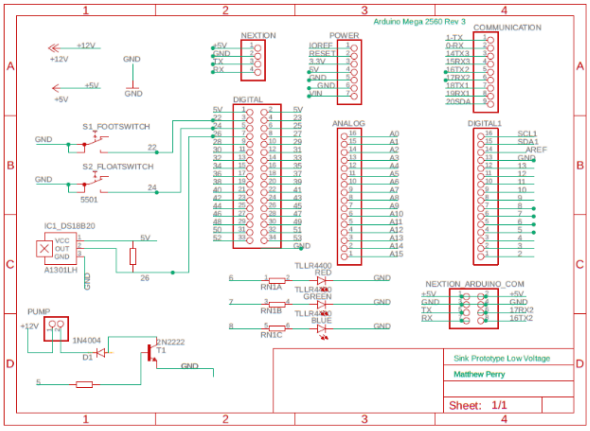
Record of activities and achievements

Investigated switching pump output and inputs such as the 12v pump from 5v input. From this created layout on word.

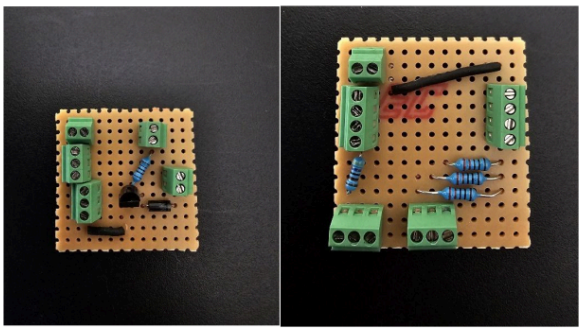
Laptop could not repair but purchased another and backed up all software and files using one-drive.

Completed eagle circuit.





Soldered Perf board.



Plans for next week
Create layout diagram and electrical diagram.

Signed: Signed: R. Davies

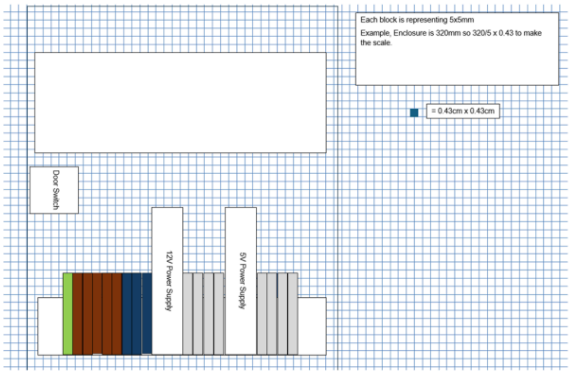
Log Entry Sheet

Log Number	23
Date	21-02-2025
Duration (hrs)	12
Accumulated time (hrs)	162

Record of activities and achievements

Created a scaled layout of the enclosure to see how well everything will fit. The enclosure was used/selected originally because it was already on site and would not cost anything to use, however, it has been decided that it's too large and a smaller enclosure would be fit the needs of the fds better.

Investigated into new enclosures while meeting cost/IP rating. Created hardware layout with enclosure dimensions online, ordered new enclosure personal spend. Due to approval waiting time etc. Updated parts list, running cost.



Plans for next week

Hardware design.

Signed: 

Signed: R. Davies

Log Entry Sheet

Log Number	24
Date	28-02-2025
Duration (hrs)	15
Accumulated time (hrs)	177

Record of activities and achievements

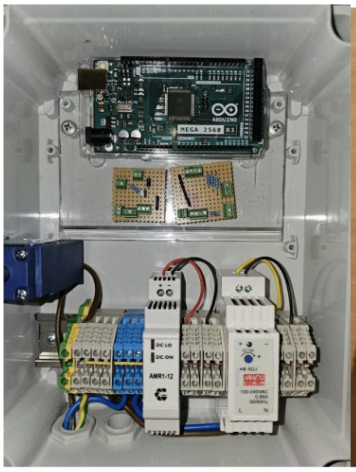
New enclosure arrived and went a bought an enclosure for the 12v pump as the fds states 3 components: footswitch, control box and pump.

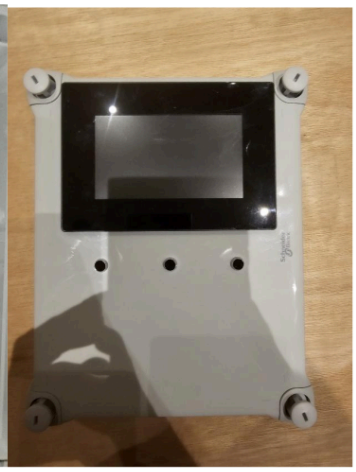
- Cut din rail with hacksaw
- Drilled holes in corners, cut from with junior hacksaw for Nextion
- Cut a piece of pvc to mount Arduino and perf board/PCB.
- Drilled holes for LED bezel.
- Assembled din rail, mounted push switch,
- Drilled and put glands in.
- Wired up power.











Plans for next week
Hardware design, ask for advice about Perfboard; not sure if they look great, but due to being a prototype it may be the best option.

Signed: 

Signed: R. Davies

Log Entry Sheet

Log Number	25
Date	07-03-2025
Duration (hrs)	10
Accumulated time (hrs)	187

Record of activities and achievements

Perfboards are staying as final prototype or next phase may have custom PCB.

Soldered leads and connections for x3 LED and soldered new connections for Nextion VCC, GND, RX,TX.



Wired the rest of the panel in accordance with eagle diagram.

Plans for next week

Complete Hardware, Design, Software Simulation

Signed: 

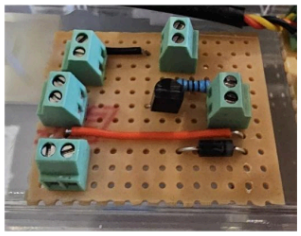
Signed: R. Davies

Log Entry Sheet

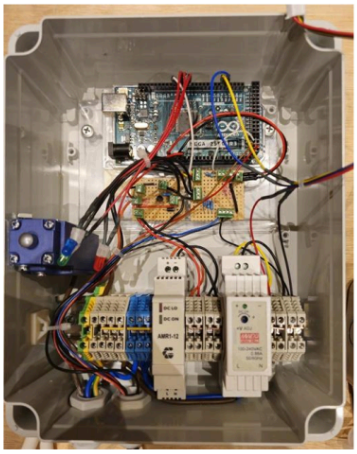
Log Number	26
Date	14-03-2025
Duration (hrs)	12
Accumulated time (hrs)	199

Record of activities and achievements

Using a program to test inputs and outputs that I created on Arduino IDE, the pump output was not activating. Serial plotter showed that footswitch was pressed, and output was on, verified 5V output with multimeter. Created new 12V perf board. Updated the perfboard layouts on Microsoft word.



Wired up new perf board in enclosure



Using serial monitor and basic program to check input and outputs.

Hardware Test			
	Pass/Fail	Evidence	Comments
Enclosure Switch	Pass	Output Serial Monitor × Message (Enter to send message to 'Arduino Mega or recvRetCommandFinished ok Temperature updated on Nextion. Temperature Sensor: 22.12 °C Foot Switch State: Released Float Switch State: Deactivated Pump State: OFF LED State: Blue ON, Green OFF, Red OFF Temperature: 22.19 recvRetCommandFinished ok Temperature updated on Nextion. Temperature Sensor: 22.12 °C Foot Switch State: Released Float Switch State: Deactivated Pump State: OFF LED State: Blue OFF, Green OFF, Red ON Temperature: 22.19	N/A
Nextion Connection	Pass		N/A
Temperature Sensor	Pass		N/A
Red LED	Pass		N/A
Green LED	Pass		N/A
Blue LED	Pass		N/A
Float switch	Pass		N/A
Footswitch	Pass		N/A
Pump	Pass		Initially not functioning, confirmed poor connection, designed new perf board and replaced; Working as intended.

Plans for next week
Software Development.

Signed: 

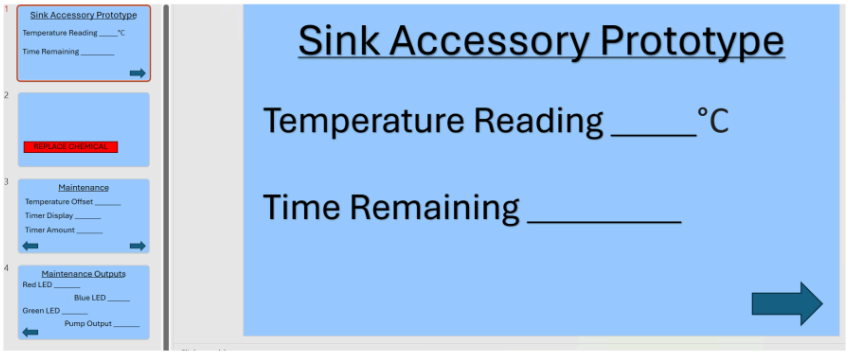
Signed: R. Davies

Log Entry Sheet

Log Number	27
Date	21-03-2025
Duration (hrs)	17
Accumulated time (hrs)	216

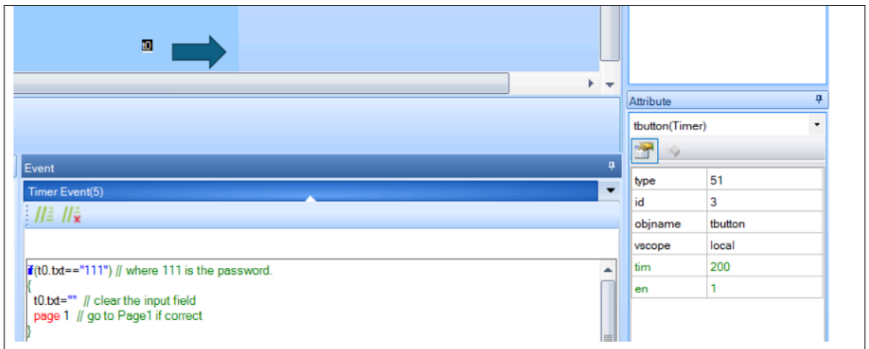
Record of activities and achievements

Created new project within Nextion, using Microsoft power point created pages to use for the background and then resized them with paint 480x272 pixel. Now using this method instead of GIMP.



Added buttons and made them transparent on the arrows for movement of the pages.

First page is the home page and to navigate to the other screens requires a password (111) using a timer to check every 200ms.



Knowing that all the hardware was correctly wired and working from the software/hardware test, I added all the buttons with the correct attributes and started writing the Arduino program.

Firstly, I wanted a page with all manual switching options, including temperature readings (with offset). I attempted to use the same method of programming that I had done in a previous examples I had done, switching the LED on and off with a button. Although this isn't outlined in the FDS it would be useful for maintenance engineers and debugging in the program.

As follows for the red led button:

NexButton sw_red = NexButton(2, 4, "sw_red"); - defining the button name, page number, ID

NexTouch *nex_listen_list[] – declaring it to the touch event list

void sw_redPopCallback(void *ptr) {

Serial.println(" red button pressed turning red LED on");

digitalWrite(ledHigh, HIGH);

} – button call backs with serial monitoring message for confirmation.

sw_red.attachPush(sw_redPopCallback); - registering call backs

This method has proved to work previously, however it gave me many error codes such as invalid commands, invalid attributes and problems compiling with a full program of what I wanted.

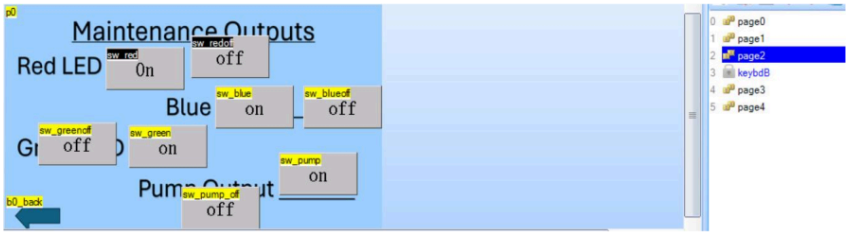
I managed to clear all errors so that it would compile and download, which it did.

However, the serial monitors displayed RX/TX communication error codes but was displaying the commands on the serial monitor such as sw_red. I knew that there was not a communication fault because I was sending information (temperature) and receiving (serial monitor) but no outputs.

I attempted to trouble shoot and amend the Arduino code and nextion touch events for most of the hours for this log with no successful outcome.

After reading the Nextion command and Arduino manual again I found a different way of carrying this out using the GetValue command. This way I can set the HMI to send component IDs on a touch event and the Arduino will read the Value as 0 or 1 which seems like a less complicated way.

I had to make the maintenance page again as I forgot to put the pump output name on PowerPoint then alter on paint to re-size. Once the outputs will switch, I will customise all the buttons and layout.



The current code –

Displays LED correctly on temperature limits.

Shows temperature

Offset option with temperature

Password protected first page

Serial monitor shows touch events.

I have saved the current code if I need to re-try this method, however I am going to start a fresh approach next week.

Plans for next week

New start for some of the Arduino program using a different method.

Signed: 

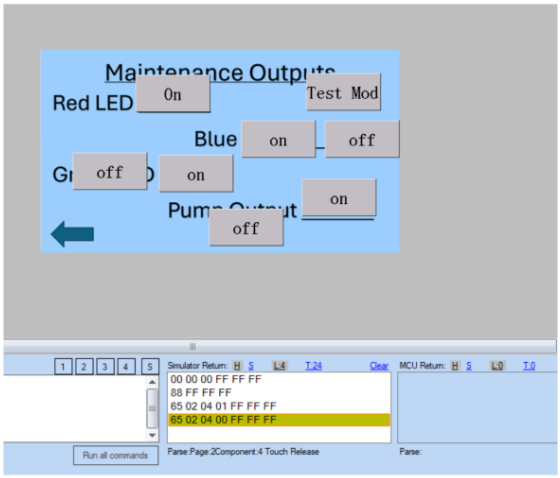
Signed: R. Davies

Log Entry Sheet

Log Number	28
Date	28-03-2025
Duration (hrs)	16
Accumulated time (hrs)	232

Record of activities and achievements

After reading the commands and looking on forums/manuals on the best way to handle was the GetValue or reading what the Arduino directly sends to the Arduino when a button is pressed which is a packet that's at least 7 bytes long as shown below when the red led was pressed.



Button pressed- send component ID, then it shows 65 02 04 00

Where 65 – touch event

02 – pageID

04 – component type

01 – event type so this has been pressed where 00 would be it hasn't been pressed.

Example:

```
void checkNextion() {
    if (Serial2.available() >= 7) {  // Wait for a full Nextion packet
        uint8_t buffer[7];
        Serial2.readBytes(buffer, 7);

        if (buffer[0] == 0x65) {  // 0x65 = Touch event
            uint8_t pageID = buffer[1];
            uint8_t componentID = buffer[2];
            uint8_t eventType = buffer[3];

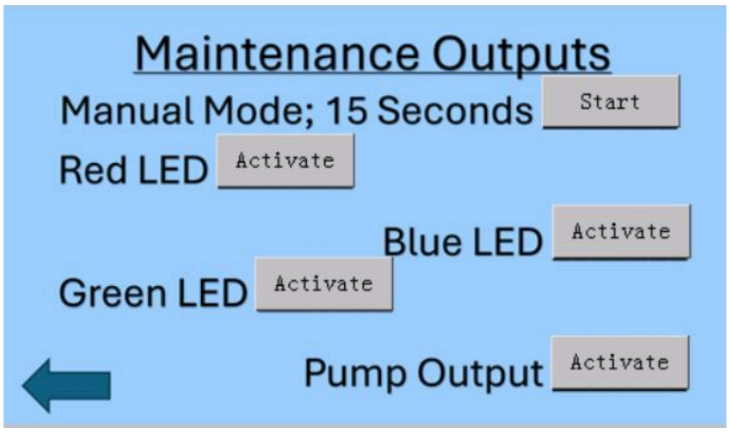
            Serial.print("Page: "); Serial.print(pageID);
            Serial.print(", Component: "); Serial.print(componentID);
            Serial.print(", Event: "); Serial.println(eventType);

            // LED Buttons
            if (pageID == 2) {  // Check for buttons on Page 2
                if (componentID == 4 && eventType == 1) {
                    digitalWrite(ledHigh, HIGH);
                    Serial.println("Red LED ON"); // Button pressed
                } else if (componentID == 9 && eventType == 1) {
                    digitalWrite(ledHigh, LOW);
                    Serial.println("Red LED OFF"); //Button not pressed
```

This works however it times out for manual LEDS because it's still trying to follow the logic based on temperature.

Amended all screens again; Powerpoint, paint, Nextion Editor

I had to use a manual mode button, so when received it will ignore all logic for 15 seconds and then revert. I tried other options recommended online such as printh command, but didn't work so I used the method I have used previously.



Currently working:

All manual mode buttons led outputs based on temperature and temperature reading with offset.

Plans for next week

Time input for dosing, making picture visible when float switch is active, covering timer option dependent on button and general sequence.

Signed 

Signed R. Davies

Log Entry Sheet

Log Number	29
Date	04-04-2025
Duration (hrs)	18
Accumulated time (hrs)	250

Record of activities and achievements

Spent time creating the timer, however unable to get it to operate as intended. At present the timer can be set on the HMI which is sent to the Arduino which is handles the timer. However, it only updates if the HMI is on page 2 and continues to cycle after an input. If not, it uses the max counting value, this is because the Nextion software hasn't changed the ID for the pages, and I am unable to change the attributes. Not sure if this is a fault on the Nextion Editor software.

Amending the program with the timer then stopped the manual mode operation, it was receiving the command but not operating the outputs which made me think that there are conflicting statements.

The Arduino was looking for an event type. Because both buttons are ID 0 and an event type with the timer it was getting confused and restarting the timer to any value, I re arranged the "check Nextion" section.

New home page again.

While amending, I changed the code, and the manual control wasn't prioritised so now if manual mode is on it is only to control the LEDs and pump. I inserted another If statement and ignoring automatic control when manual mode is active, so the countdown and manual timers are separate. The timer is now operational; however, the Arduino will only read the value from page 0 which is an issue because that is currently the home page, and I don't want users to have access to pump timings etc.

Today was the deadline for the software to be completed so, I am going amend the layout of the home page again so it can be inserted. It isn't ideal, but with the deadline coming up I need to focus on getting a completed system.

Plans for next week

Complete software, carry on with simulation;

Timer input for dosing, making picture visible when float is active, covering timer option dependent on switch button and general appearance.

Signed 

Signed R. Davies

Log Entry Sheet

Log Number	30
Date	11-04-2025
Duration (hrs)	20
Accumulated time (hrs)	270

Record of activities and achievements

I have tried multiple ways of sending the inputted value to be read by the Arduino if on page 1 but still unable to get it to operate correctly. I've tried

- Setting variable in different pages to be read by the Arduino.
- Checking for change using timer on Nextion.
- Storing the initial value indefinitely once it read correctly by the Arduino.

The only way that I can set a value it to have the inputs on Page 0, which isn't ideal because then users are able to change settings which the fds stated against. While amending the code I have also caused the temperature offset to stop working which I'm currently trying to debug.

The only option I can think of is a show/hide command. This way I can use a password on a button, once its correct it will hide an image which then allows access to the inputs on page 0. This means I will need to completely re-create the background images and change the attributes of components on the Nextion & Arduino code.

Created a way for this to work as intended while still meeting the FDS all working.

Manual mode is not working again, think it is because I deleted the now redundant pages so component IDs, page numbers etc may change. Checked the hex string from the buttons which have shown that is has changed for example, previously the manual button was

65 02 04 01 but now its 65 01 07 01 (touch event) which verifies the change.

Referenced all buttons against the excel spreadsheet which the Nextion has automatically changed. I have now updated the Arduino code and the excel spreadsheet. Simulation and software seem to be working as intended while meeting the fds.

Software complete, however created date/time for calibration alarm but unable to store values after power loss, after research can store in Arduino eeprom but limited on time, will attempt to resolve if time allows.

Plans for next week

Testing and compile report.

Signed 

Signed R. Davies

Log Entry Sheet

Log Number	31
Date	18-04-2025
Duration (hrs)	8
Accumulated time (hrs)	278

Record of activities and achievements
Unable to add date and time for calibration alarm as I am already one week behind on software and don't have time for debugging etc. Compiling all digital media for report. Requested test equipment, organising a space for testing and getting board to mount system on to prove test for evidence.

Plans for next week
Testing

Signed 

Signed R. Davies

Log Entry Sheet

Log Number	31
Date	25-04-2025
Duration (hrs)	10
Accumulated time (hrs)	288

Record of activities and achievements
Collected test equipment. Set up hot block ran pre-cal. Testing and recording the evidence, inserting into report. Issue with dosing logic, where it would only dose if float switch was low but notified Nextion. Checked Arduino code and rectified code.

Plans for next week
Finish report

Signed 

Signed R. Davies

Log Entry Sheet

Log Number	32
Date	02-05-2025
Duration (hrs)	22
Accumulated time (hrs)	310

Record of activities and achievements
Writing report, amendments, compiled. Completed report.

Plans for next week
Complete presentation 2.

Signed 

Signed R. Davies

2024421

GRADEMARK REPORT

FINAL GRADE

GENERAL COMMENTS

82/100

PAGE 1

PAGE 2

QM **Good Structure**

Report is structured exactly as stated in the assignment brief.

PAGE 3

PAGE 4

PAGE 5

PAGE 6

Text Comment. Did your lecturers not support you?

PAGE 7

Text Comment. This is a really great summary that tells your readers everything they need to know about your project.

PAGE 8

Text Comment. A good introduction guides the reader through the report

PAGE 9

Text Comment. useful diagram

Text Comment. (Perry, 2024)

Text Comment. You should have cited assignment 1 as a source for the FDS here

PAGE 10

Text Comment. (Perry, 2024)

Text Comment. These are great design considerations

PAGE 11

Text Comment. 330 Ohm

Text Comment. Why choose the 2N2222

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

Text Comment. why such a high voltage rated diode?

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

PAGE 12

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

Text Comment. Excel

PAGE 13

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

Text Comment. (Perry, 2024)

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

PAGE 19

PAGE 20

QM **Correctly cited**

Citation method conforms to Harvard standard, well done.

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

PAGE 36

QM

Wide Range of Sources

You have shown good evidence of reading from a wide range of sources.

QM

Correct method of listing.

You have listed your sources correctly according to the Harvard method.

PAGE 37

PAGE 38

Text Comment. One more source needed: Perry, 2024, Assignment 1 .....

PAGE 39

PAGE 40

PAGE 41

PAGE 42

PAGE 43

PAGE 44

PAGE 45

PAGE 46

PAGE 47

PAGE 48

PAGE 49

PAGE 50

PAGE 51

PAGE 52

PAGE 53

PAGE 54

PAGE 55

PAGE 56

PAGE 57

PAGE 58

PAGE 59

PAGE 60

PAGE 61

PAGE 62

PAGE 63

PAGE 64

PAGE 65

PAGE 66

PAGE 67

PAGE 68

PAGE 69

PAGE 70

PAGE 71

PAGE 72

PAGE 73

PAGE 74

PAGE 75

PAGE 76

PAGE 77

PAGE 78

PAGE 79

PAGE 80

PAGE 81

PAGE 82

PAGE 83

PAGE 84

PAGE 85

PAGE 86

PAGE 87

PAGE 88

PAGE 89

PAGE 90

PAGE 91

PAGE 92

PAGE 93

PAGE 94

PAGE 95

PAGE 96

PAGE 97

PAGE 98

PAGE 99

PAGE 100

PAGE 101

PAGE 102

PAGE 103

PAGE 104

PAGE 105

PAGE 106

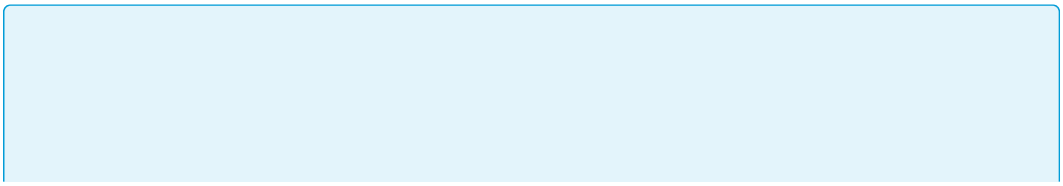
PAGE 107

PAGE 108

PAGE 109

PAGE 110

GRADING FORM: NEW CSG-RUBRIC



MATTHEW PERRY **82**

1. ENGAGEMENT WITH LITERATURE

Section 3: Judicial use and critical evaluation of technical literature in the design of the project.
There is clear evidence of reading and evaluation of sources. (4/5)
Section 8: Extent of reading, range of credible sources and use of Harvard method.
A very wide range of sources have been considered. However, you should also have cited assignment 1 as one of your sources (4/5)

8/10 **8**

2. KNOWLEDGE AND UNDERSTANDING

Sections 3 and 4: Advanced knowledge and critical understanding demonstrated in the design and implementation of the project.
There is excellent evidence of knowledge and understanding through the report (8/10)

8/10 **8**

3. COGNITIVE AND INTELLECTUAL

Section 5: Testing and analysis of results.
Excellent evidence of testing and analysis of results (9/10)
Section 5: Critical evaluation of the project outcomes.
Excellent and thorough evaluation of project outcomes. (8/10)
Section 6: Conclusions and recommendations.
An excellent set of conclusions and recommendations (8/10) **25**

25/30

4. PRACTICAL APPLICATION

Section 4: Deployment method, use of material, tools, equipment and techniques in the construction of the project.
Thorough treatment in the project, excellent attention to detail in the deployment of methods etc.. (9/10)
Section 4: Quality of the final product or system.
The end product is of a high quality. Internal wiring could be neater. (7/10)
Sections 5, 6 and appendix: Management of the design process including use of plans, costing spreadsheet and log book.
There is excellent evidence of understanding of project management processes. (8/10)

24

24/30

5. TRANSFERABLE SKILLS

Section 1: Acknowledgements, executive summary and introduction.
Very good acknowledgements that show the level of independence achieved in the project. An excellent summary that tells the reader everything they need to know about the project. A good introduction guides the reader through the report (4/5)
All Sections: Presentation quality of the report, diagrams, charts, tables, figure numbering, page and section numbering etc..
Of a high standard (4/5)
All Sections: Quality of the writing style, use of punctuation and grammar.
High standard generally (4/5)
Word count limit NOT exceeded (5/5)

17