PONDER

EEE3035: Engineering Professional Studies - Group J
Clean Growth

PROJECT REPORT

V. Berdnikov-Levitsky, J. Ehuriah, K. Harrison, R. Rafky, P. Sodani, P. Soni, M. Taylor, L. Williams

Bachelor of Engineering, University of Surrey
Faculty of Engineering and Physical Sciences (FEPS)
Department of Electrical and Electronic Engineering
Semester 1, 2020-21

Contents

Word count: 19,825

Abstract [V.B.L.]

Whilst numerous large developing countries such as India and Brazil continue to urbanize and industrialize rapidly, their effective protection of the environment sadly remains lacking. Poor waste management, heavy manufacturing and increasing weather volatility have inflicted pronounced and widely-recognised damage upon typical water qualities within such countries. The leakage of hazardous materials and pollutants into drinking water sources, combined with inadequate or non-existent water processing are massively detrimental upon public health and the environment as well as a region's economy and reputation.

By monitoring fluctuations in water quality throughout a water network, the root causes of contamination may be identified. Polluting factories, agricultural run-off, major dumping sites and chemical spillages produce sharp changes in the levels of specific contaminants over large areas as these substances migrate down-stream. Combining this information with knowledge of a river's topography will allow a network of PONDER (POllutioN DEtectoR) aquatic sensor platforms to pin-point pollutant sources within any given waterway. Contemporary methods for monitoring water quality often consist of hand-held meters or bench-top equipment that require human personnel and or watercraft in order for measurements to be conducted [1][2]. Current stationary sensor buoys for monitoring water quality are expensive, typically costing hundreds or thousands of pounds [3][4] and are usually deployed individually. We aim to produce a prototype that could be reproduced and implemented, as a network of sensor buoys, permitting a client to wirelessly monitor several parameters indicative of water quality at numerous points throughout a waterway.

The main innovative aspect of our product is its robustness and functionality despite its low unit price. The device, which shall henceforth be referred to as PONDER, will be a highly-cost effective aquatic sensor platform constructed with off-the-shelf components and will function as an affordable, versatile, unmanned system for the quantification of a body of water's quality.

1

Introduction

The STEM sector has an important role to play in facilitating, practically, the continued and inevitable exponential growth of the world's population and its technology in a manner which inflicts as little irreversible damage as possible upon the environment, human lives, society, culture and our understanding of science itself. Whilst continuing to strive for improved astuteness and efficiency, the science and technology community must constantly monitor and regulate itself to be sustainable above all other goals. Sustainability relates to the nature and quality of the environment, human lives and all other aforementioned entities, being primarily concerned with their maintenance if not continued improvement. Clean growth can therefore be viewed as an essential heuristic exercise in resource management, to the end of enabling the ideally limitless improvement of science and technology without sacrificing their progenitor, humanity, in the process. [V.B.L.]

1.1 UN Sustainable Development Goals [M.T.]

Water quality measurements collected by PONDER will be transmitted to a remote server and stored in a database for further processing. Data gathered on a particular body of water's quality would enable a client to make informed decisions to the end of accomplishing several of the UN's Sustainable Development Goals (SDGs).

- **Goal 6: Clean Water and Sanitation**
    - Over 80% [5] of wastewater resulting from human activities is discharged into rivers or seas without any form of treatment nor pollution filtration whatsoever. [6]

- **Goal 12: Responsible Consumption and Production**
    - The global population's rate of pollution, with respect to time, to rivers and lakes exceeds nature's capacity to reverse such damage. [7]
    - 3 out of 10 people (2.1 billion people, or 29% of the global population) did not have access to a safe drinking-water service in 2015. In addition, 844 million people lacked access to any basic drinking water service whatsoever. [8]

- **Goal 14: Life Below Water**
    - Eutrophication (excess nutrients that result in excessive algal growth and subsequent oxygen depletion).
    - Minerals such as Nitrogen (N), Phosphorous (P), and Potassium (K).
    - Roughly 80% [9] of marine and coastal pollution originates on land. This includes agricultural run-off, pesticides, plastics and raw sewage. [10]

1.2 Developing Countries [M.T.]

The following paragraph, from the UN's research on "Water Quality and Wastewater", presents many of the issues faced by developing countries.

"Due to population growth, accelerated urbanization and economic development, the quantity of wastewater generated and its overall pollution load are increasing globally. Globally, 80% of wastewater flows back into the ecosystem without being treated or reused, contributing to a situation where around 1.8 billion people use a source of drinking water contaminated with faeces, putting them at risk of contracting cholera, dysentery, typhoid and polio. Mainly in low-income areas of cities and towns within developing countries, a large proportion of wastewater is discharged directly into the closest surface water drain or informal drainage channel, sometimes without or with very little treatment. In addition to household effluent and human waste, urban-based hospitals and industries such as small-scale mining and motor garages, often dump highly toxic chemicals and medical waste into the wastewater system. Pollution of groundwater and surface water by agricultural use of untreated or inadequately treated wastewater is a major issue in many developing countries where such irrigation is practised." [11

2

Project Management

2.1 Organization [M.T.]

WhatsApp and Slack groups as well as a Google Drive folder, a GitHub repository, and regular Zoom meetings were utilized in order to facilitate communication and the exchange of electronic files related to the project. The WhatsApp group was actively used to plan meetings, share ideas, and discuss general matters concerning the project. The Slack Group was used in the early stages of the project to share brainstorming ideas and background research. The shared Google Drive folder was used to store all of our work with the exception of our code, which was stored on our GitHub repository to make version control easier [12]. We held online meetings using Zoom, in view of COVID-19.

2.1.1 Roles & Responsibilities [M.T.]

We identified that our project would involve a mixture of hardware and software design. Thus, we split the work among our eight team members according to their strengths and interests. More details on who completed each task are found in figures. 5 and 6 of section 2.2.3: Risk Analysis and is reflected by individuals' individual contributions to this report, as shown by coloured initials.

Team Member	Tasks
V. Berdnikov-Levitsky [V.B.L.]	Research, Documentation, Project Management, Hardware
J. Ehuriah [J.E.]	Research, CAD
K. Harrison [K.H.]	Research, Documentation
R. Rafky [R.R.]	Research, Android App
P. Sodani [P.S$_{odani}$]	Research, Software
P. Soni [P.S$_{oni}$]	Research, Software
M. Taylor [M.T.]	Research, Documentation, Project Management, Hardware
L. Williams [L.W.]	Research, Software Integration

Fig. 1: General Team Member Roles

2.2 Work Plan [M.T.]

2.2.1 Gantt Chart [M.T.] [V.B.L.]

We created a Gantt chart to keep track of tasks and of our progress. Figure. 2 shows the final version of our Gantt Chart. We had initially planned to complete the project by the start of the Christmas holidays, but some tasks, such as the Arduino Firmware and GSM/GPRS communication took much longer than expected. The Gantt Chart shown below reflects these changes in our work plan.

Project PONDER Gantt Chart

TASK NO.	TASK TITLE	RISK LEVEL	TASK OWNER	START DATE	DUE DATE	DURATION / DAYS	% OF TASK COMPLETE
0	**Deadlines / Labs**						
0.1	Design Brief Submission	HIGH	All	N/A	27-10-2020	N/A	100%
0.2	Project Report Submission	HIGH	All	N/A	05-01-2021	N/A	100%
0.3	Project Presentation	HIGH	All	N/A	25-01-2021	N/A	0%
0.4.1	Lab 1 - 10AM-1PM - 16AB04	LOW	Hardware	05-11-2020	N/A	3 hours	100%
0.4.2	Lab 2 - 10AM-1PM - 16AB04	LOW	Hardware	19-11-2020	N/A	3 hours	100%
0.4.3	Lab 3 - 10AM-1PM - 16AB04	LOW	Hardware	03-12-2020	N/A	3 hours	100%
0.4.4	Extra Lab - 10AM-1PM - 16AB04	LOW	Hardware	10-12-2020	N/A	3 hours	100%
1	**Research**						
1.1.1	Water Pollutant Research	LOW	All	19/10/2020	21/10/2020	3	100%
1.1.2	Sensor Choices	MEDIUM	All	19/10/2020	21/10/2020	3	100%
2	**Design - Hardware**						
2.1	Solar Power Calculation	LOW	Hardware	21/10/2020	24/10/2020	4	100%
2.2	Component Choices	LOW	Hardware	21/10/2020	24/10/2020	4	100%
2.3	Hardware Block Diagram	LOW	Hardware	21/10/2020	24/10/2020	4	100%
2.4	Ordering Components	MEDIUM	Hardware	25/10/2020	20/11/2020	26	100%
2.5.1	Enclosure Design Ideas	LOW	Hardware	21/10/2020	14/11/2020	24	100%
2.5.2	Enclosure Build	LOW	Hardware	01/11/2020	03/12/2020	33	100%
2.6	Arduino Firmware	MEDIUM	Hardware	01/11/2020	10/12/2020	40	100%
2.7	PCB/Veroboard Design & Build	MEDIUM	Hardware	15/11/2020	02/01/2021	48	100%
2.8	Software Integration	MEDIUM	Hardware	02/12/2020	02/01/2021	31	100%
3	**Design - Software**						
3.1.1	Basic Visual Design of the App	LOW	Software	19/10/2020	26/10/2020	8	100%
3.1.2	Building App	MEDIUM	Software	26/10/2020	10/12/2020	45	100%
3.2	Software Block Diagram	LOW	Software	19/10/2020	26/10/2020	8	100%
3.3	DB Server Setup	MEDIUM	Software	27/10/2020	04/11/2020	8	100%
3.4	Research Bluetooth Data Transmission	MEDIUM	Software	02/11/2020	10/12/2020	39	100%
3.5	Research GSM/GPRS/GPS Data	HIGH	Software	02/11/2020	10/12/2020	39	100%
3.6	Implement Data Communication	HIGH	Software	23/11/2020	01/02/2021	69	100%
4	**Documentation**						
4.1	Specification Verification Matrix	LOW	All	21/10/2020	27/10/2020	7	100%
4.2	SAGE, EIA, DMP forms	LOW	All	21/10/2020	27/10/2020	7	100%
4.3	Risk Assessment Matrix	LOW	All	21/10/2020	27/10/2020	7	100%
4.4	Project Report	LOW	All	11/12/2020	05/01/2021	25	100%
4.5	Project Presentation	LOW	All	05/01/2021	25/01/2021	21	0%

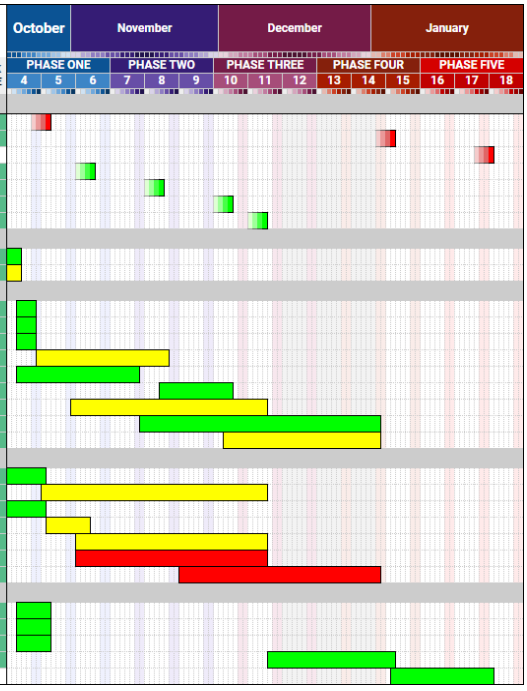


Fig. 2: Revised Project Gantt Chart [M.T.] [V.B.L.]

2.2.2 Meeting Schedule [M.T.]

A detailed log of each of our meetings was kept by Mark, for the benefit of those who could not attend a meeting. This log is available on request.

Meeting Date	Time	Discussion
09/10/2020	13:00-14:00	Project Management
14/10/2020	14:00-15:00	Brainstorming
18/10/2020	13:00-14:00	Research, Sensors
21/10/2020	13:00-14:00	Bill of Materials
25/10/2020	13:00-13:30	Design Brief
04/11/2020	13:00-13:30	Various Tasks
15/11/2020	13:00-13:30	Various Tasks
09/12/2020	13:00-13:30	Various Tasks
02/01/2021	13:00-13:30	Technical Report

Fig. 3: Zoom Meetings

2.2.3 Risk Analysis [M.T.]

We identified the key risks and mitigation measures for each task. We assessed the risk occurrence likelihood and potential severity of outcome and used a typical risk matrix to obtain an overall risk level (Low/Medium/High). Risk mitigation factors were then considered to minimise risk where possible. The highest risk tasks were regarding the cellular data transmission via GSM/GPRS, as the A9G module we chose has very little documentation in English, and this was an area which none of us had much experience in. We mitigated the risk by allocating a lot of time and multiple team members to it, and by also implementing Bluetooth communication as a redundancy. All other tasks had low or medium risk after mitigation factors.

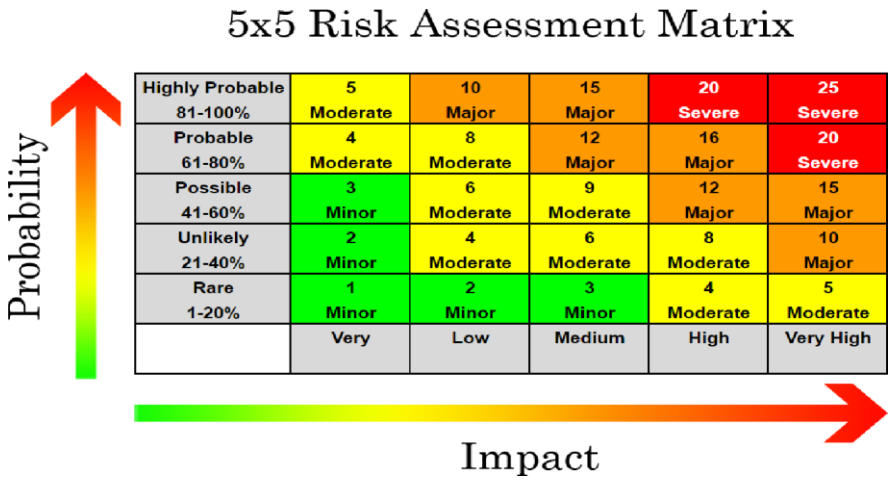


Fig. 4: 5x5 Risk Assessment Matrix

Task No.	Task Title	Description	Est. Time Req.	Task Owner	Additional Notes	Risks	Probability	Impact	Risk Level
1	Research								
1.1.1	Water Pollutant Research	Each team member is assigned a type of water pollution to research and write a short document on. Research should include information on the source, effects, and ways to measure the type of pollution.	3 days	1) Vasily 2) Jordan 3) Kieran 4) Lawrie 5) Mark 6) Pranay 7) Priyam 8) Robby	1) Ammonia 2) Fertilizers such as nitrates and phosphates 3) Detergents and surfactants 4) Oils and hydrocarbons 5) Organic and other solvents 6) Heavy metals such as cadmium, mercury, lead and arsenic 7) Oxygen-depleting substances 8) Sulphur Dioxide	Risk of inaccurate, outdated, or inapplicable information, among others. Minimize risk by referencing relevant and reputable sources. Ensure that information presented is relevant and up to date.	Rare	Very Low	LOW
1.1.2	Sensor Choices	Use Water Pollutant Research to inform choice of sensor to be used in our project. Requires consideration of multiple factors, including how useful the sensor data is, price of sensor, size/weight/form factor constraints, voltage required, analogue/digital pin requirements, ease of integration, ease of acquirement, etc.	3 days	All	Prerequisite to further tasks concerning component selection.	Risk of choice of sensors which later turn out to be incompatible, less robust, inprecise, useless, difficult to integrate, etc. Minimize risk by choosing sensors from reputable vendor with documentation/data sheets available. Check voltage and pin requirements thoroughly.	Unlikely	High	MEDIUM

Fig. 5: Task Descriptions for Task Section 1

2	Design - Hardware								
2.1	Solar Power Calculation	Calculate size of solar panel needed and size of battery needed, taking into account solar irradiance on a horizontal surface in December, and various charging inefficiencies.	4 days	Mark	Must be completed in order to inform choices of components, including: size/number of batteries, size of solar panel, type of battery charger, type of microcontroller, etc	Many of the calculations depend on assumptions. There is a risk that real world use will differ from calculations as a result. For example, we assume that the Arduino NANO consumes 20mA in Idle mode, but this does not include idle current draw from other sensors. However, the NANO can be put into deep sleep mode, reducing current draw substantially.	Rare	Low	LOW
2.2	Component Choices	Make decisions concerning what components to buy, and from what vendors. Consider the price of components, and ensure compatibility between them.	4 days	Mark	The Bluetooth and GSM/GPRS/GPS modules both use UART. Need to choose on or the other. pH probe may be unsuitable for extended immersion	Risk of unforseen compatibility issues between components. Minimise risk by reading data sheets and checking pin and voltage requirements.	Rare	Medium	LOW
2.3	Hardware Block Diagram	Create system overview block diagram, showing top level representation of hardware modules and how they interface with each other	4 days	Mark, Lawrie		No risks involved. Diagram can always be modified to account for errors or changes in design.	Rare	Very Low	LOW
2.4	Ordering Components	Ordering the components chosen from various vendors.	26 days	Mark	Prerequisite to Arduino Prototyping task	Risk of defective/counterfeit products. Minimise risk by purchasing from trustworthy vendors where possible.	Rare	High	MEDIUM
2.5.1	Enclosure Design Ideas	Brainstorm various ideas regarding the design of the product's enclosure, taking into consideration buoyancy, cost, rigidity, waterproofing, stability, etc	24 days	All	Prerequisite to task 2.5.2	Risk of producing poor idea with a design flaw. Mitigate by asking everyone to share ideas and comment on each other's ideas.	Rare	Medium	LOW
2.5.2	Enclosure Build	Design of enclosure in a CAD program, and building of enclosure.	33 days	Mark, Jordan		Some difficulty in ensuring that enclosure is well built and water proof.	Possible	Very Low	LOW
2.6	Arduino Firmware	Writing and testing sketches for each component individually, then integrating them into a single design. Measure power draw and take steps to lower it.	40 days	Mark		Risk of unforseen issues when writing Arduino sketches. Mitigate by assigning multiple people and plenty of time to sort out issues.	Possible	Low	MEDIUM
2.7	PCB/Veroboard Design & Build	Design and build of PCB or veroboard to hold Arduino modules and supporting components.	48 days	Mark, Vasily	Begins in latter stage of Arduino Prototyping. Need to account for shipping time if PCB route chosen.	Risk of mistake with PCB design, or of long shipping time delaying project. Mitigate by having veroboard as a backup option, and considering UK manufacturer if short on time.	Unlikely	Low	MEDIUM
2.8	Software Integration	Integrating hardware and software.	31 days	Lawrie		Risk of problems with integration, particularly in comms. Minimise risk by developing both Bluetooth and GSM/GPRS comms in case one doesn't work.	Unlikely	High	MEDIUM
3	Design - Software								
3.1.1	Basic Visual Design of the App	Create mockup of visual interface (GUI) for mobile application, that the end user will see.	8 days	Robby		No risk. At this stage, design is just a preliminary mockup to give a general idea. It is likely to change in the future.	Rare	Very Low	LOW
3.1.2	Building App	Building mobile application with the ability to present the data from each sensor from multiple devices in a clear and visually pleasing manner.	45 days	Robby		Risk that data is not presented coherently. Significant risk that data transmission functionality of app does not work with Bluetooth and/or GSM/GPRS. Minimise risk by allocating significant time and energy to task.	Possible	Medium	MEDIUM
3.2	Software Block Diagram	Create block diagram representing top level view of software side of the project.	8 days	Pranay, Priyam		No risk. Diagram can be changed at any time if design changes.	Rare	Very Low	LOW
3.3	DB Server Setup	Setup of back-end database server	8 days	Pranay, Priyam		Risk of unforseen issues with storage/formatting/retrieval of data in database. Minimise by considering back up options/redundancies.	Unlikely	Medium	MEDIUM
3.4	Research Bluetooth Data Transmission	Research the use of Bluetooth to transfer data from the device to the mobile app or database.	39 days	Pranay, Priyam	Research may inform choice of communication method to be implemented.	Research may turn up issues regarding compatibility or ease of implementation with regards to our project.	Possible	Medium	MEDIUM
3.5	Research GSM/GPRS/GPS Data Transmission	Research the use of GSM/GPRS to transfer data from the device to the mobile app or database.	39 days	Pranay, Priyam	Research may inform choice of communication method to be implemented. GSM/GPRS+GPS is preferred but more complex.	Research may turn up issues regarding compatibility or ease of implementation with regards to our project. Will need to consider data transmission in the case of poor signal, and cost of SIM card.	Probable	Medium	HIGH
3.6	Implement Data Communication	Implementation of data transmission via Bluetooth or GSM/GPRS. Bluetooth is much easier but greatly limits remote function of device, whereas GSM/GPRS+GPS allows for extended remote use, and GPS positioning, but at the cost of greater complexity.	69 days	Pranay, Priyam, Lawrie	Data communication of some form is a 'must have' feature. The use of a GSM/GPRS+GPS module would allow for remote use and GPS data which are 'should have' features, whereas Bluetooth communication would only satisfy the necessary 'must have' feature.	Risk that data communication via GSM/GPRS proves too difficult to implement. Minimise risk by having the option to implement Bluetooth communication instead.	Possible	Very High	HIGH
4	Documentation								
4.1	Specification Verification Matrix	List of specifications that the finished product must pass. Used to inform our design decisions and prioritise importance of features.	7 days	Kieran		Risk that we neglect to consider a specification necessary for the operation of the device. Very low impact if indeed the case.	Rare	Very Low	LOW
4.2	SAGE, EIA, DMP forms	Completion of Sage Ethical Considerations form, Equality Impact Assessment, and Data Management Plan, to be used as appendices in short design brief submission	7 days	Kieran, Mark, Jordan		Risk that completion of forms brings up issues with regards to ethics, equality, or data management. Minimise risk by designing project in such a way as to avoid such problems.	Rare	Very Low	LOW
4.3	Risk Assessment Matrix	Listing of tasks, and identification of key risks involved with each task using a risk assessment matrix	7 days	Mark		Subjective assessment of risk occurrence likelihood and potential severity of outcome may be incorrect, leading to failure to implement sufficient measures to mitigate risks. Minimise risk by reassessing risks throughout project, and taking action as needed. Make sure to communicate risks effectively.	Rare	Low	LOW

Fig. 6: Task Descriptions for Task Sections 2-4

2.3 Specification Verification Matrix [K.H.]

We considered how our project would address the client brief, and so created a specification verification matrix to concretely set out our project requirements. Please refer to Appendix A.

2.4 Component Ordering Schedule [M.T.]

Most components were ordered by the end of October 2020, and arrived by mid-November, in time for the bulk of our scheduled laboratory sessions. Components for the enclosure were ordered in mid-November after we settled on a design (see section 4.3.1) and arrived by the end of November.

COVID-19 restrictions made working on parts of the hardware difficult. As Mark had most of the hardware in Switzerland, a second A9G module for cellular communication was ordered in December, to facilitate Mark and Lawrie working on it simultaneously. Despite Vasily having the means to manufacture a PCB at home, COVID-19 made it difficult for him to send it to Mark who was staying in Switzerland at the time, and so Mark implemented the circuit on Veroboard instead. The PCB and Veroboard designs can be found in sections 4.6 and 4.7, respectively.

3

Research

3.1 Quantification of Water Quality [M.T.]

Each team member researched a specific type of water pollutant: Ammonia, Fertilizers such as nitrates and phosphates, Detergents and surfactants, Oils and hydrocarbons, Organic and other solvents, Heavy metals such as cadmium, mercury, lead and arsenic, Oxygen-depleting substances, Sulphur Dioxide. The research is presented in the Appendices.

From this research, we determined that many of these pollutants require expensive specialist equipment or lab processes to measure, making them unsuitable for our project. However, despite being unable to detect specific pollutants directly, the presence of many of these pollutants manifest themselves as changes in the water's temperature, TDS, turbidity, pH, Oxidation Reduction Potential, Conductivity, and Dissolved Oxygen. These characteristics can easily be quantified by relatively cheap and accessible sensors, ranging in price from £2-£130 (the most expensive being the Dissolved Oxygen sensor).

For instance, the overuse of fertilizers in farming results in nutrients leaching into the groundwater and making their ways into rivers and bigger bodies of water. The fertilizer runoff contains excessive amounts of nutrients, including minerals like Nitrogen (N), Phosphorous (P), and Potassium (K). These nutrients are normally the limiting growth factor for algae, without which, algae can grow on a massive scale, forming an algal bloom. The algal bloom reduces light penetration, killing plants beneath as they cannot get sunlight to photosynthesize. Eventually, the algal bloom dies and sinks to the bottom of the water, where it is decomposed by bacteria, using up oxygen for respiration. The decomposition depletes the oxygen dissolved in the water, rendering

it toxic, an effect called hypoxia, resulting in the death of most plants and fish, and making the water unsafe for humans to touch in some cases. Surveys showed that 54% of lakes in Asia are eutrophic; in Europe, 53%; in North America, 48%; in South America, 41%; and in Africa, 28%. [13]

Measuring the concentration of Nitrogen, Phosphorus and Potassium directly is difficult, but we could indirectly measure the water's pH and Dissolved Oxygen. The bacterial decomposition process makes the water more carbonic, decreases dissolved oxygen, and decreases the pH level.

On the next page is a table of sensors, their seller, associated cost and whether they will be easy to integrate with an Arduino. It is important to note that there are a few water quality determinants that do not have an electronic method of measurement, and thus have not been included in the table.

3.2 Sensor Choices [K.H.][M.T.]

There are a few key criteria that were used in order to select which aspects of water quality were chosen, and which sensors were selected to measure them:

1. The selection of sensors needs to measure a broad range of water quality properties, hence try to avoid sensors that measure the same or similar properties to remove unnecessary redundancy.
2. The chosen sensors need to be low-cost so as to not take the total cost of the product over the given budget (£100).
3. The chosen sensors need to be easy to integrate with an Arduino microcontroller.

Below is a table of sensors, their seller, associated cost and whether they will be easy to integrate with an Arduino. It is important to note that there are a few water quality determinants that do not have an electronic method of measurement, and thus have not been included in the table.

Sensor	Seller	Price	Arduino Integration?
Sulphur Dioxide	RS Components	£15.01	Yes
Nitrates	Edulab	£459.64	No
Phosphate	CPC	£75.60	No
Dissolved Oxygen	DFRobot	£129.81	Yes
Turbidity	DFRobot	£7.60	Yes
Hydrocarbon	Frontline Safety	£1.098.00	No
Ammonia	Cool Components	£39.99	No
Laboratory pH	DFRobot	£11.44	Yes
Industrial pH	DFRobot	£43.71	Yes
Temperature	DFRobot	£5.76	Yes
TDS	DFRobot	£6.84	Yes
Electrical Conductivity	DFRobot	£53.69	Yes

Fig. 7: Sensor Choices

Of the aforementioned sensors, we chose to implement temperature, TDS, Turbidity, and pH sensors, given our limited budget of £100. However, our prototype serves as an easily expandable platform, as more sensors can be added depending on the budget.

3.2.1 Temperature [K.H.]

The primary sensor that will be used in this product is the temperature sensor. It is a vital sensor for this product, not necessarily in terms of water quality, but more for operational purposes of the other sensors and components within the device, which can refine their readings based on the temperature.

3.2.2 Total Dissolved Solids (TDS) [K.H.]

TDS (Total Dissolved Solids) indicates how many milligrams of soluble solids are dissolved in one litre of water. In general, the higher the TDS value, the more soluble solids are dissolved in the water, and the less clean the water is. Therefore the TDS value is an indication of the cleanliness of water, which is why the TDS sensor is included in the device.

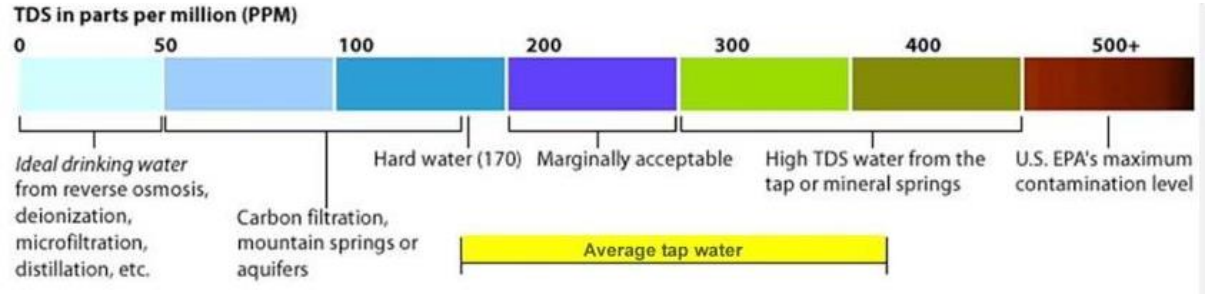


Fig. 8: TDS Graph [26]

3.2.3 Turbidity [K.H.]

The inclusion of this Turbidity sensor will allow the measurement of an important water quality feature, level of turbidity. It is able to detect suspended particles in water by measuring the light transmittance and scattering rate which changes with the amount of total suspended solids (TSS) in water. As the TSS increases, the liquid turbidity level increases.

3.2.4 pH [K.H.]

pH is a vital measurement of water quality as water that is too acidic or too alkali can be dangerous, so it is important that drinkable water has a pH of around 7.

Detailed information on the specifications and use of these four sensors can be found in section 4.2 on Component Selection.

3.3 Data Transmission [K.H.]

In order to comply with specifications given in the initial brief the project must contain:

"An aspect of timed data transmission to a known free secure location."

Therefore, to meet this some method of data transmission must be included in the final product, and as it is not specified as to what method must be used, a few different methods will be explored in this section. There are a few criteria that the chosen method must meet in order for inclusion in the final device. They are as below:

- Must be capable of transmitting data to an external device at a specified time or a repeated transmission at a user set interval.

- Must be easy to integrate with the Arduino microcontroller in order to easily interface the data transmission with the rest of the device (sensors, data storage, etc.).

With these criteria in mind, there are a few options available for inclusion within the device, and they are as below:

- Bluetooth
- Global System for Mobile Communication (GSM)
- General Packet Radio Service (GPRS)

Each of these transmission methods will be examined in more detail later in this section. As stated in the second criteria the data transmission module must be easily to integrate with the Arduino, and there are existing Arduino modules that are available for purchase.

Module Name	Price	Seller	Arduino Compatible?
HC-05 Arduino Bluetooth	£4.60	ST-8078	Yes
A9G GSM/GPRS + GPS	£8.53	DFRobot	Yes

Fig. 9: Arduino Compatibility Table

3.3.1 Bluetooth [K.H.]

Bluetooth is primarily designed for communicating over short distances less than about 10m or around 30 ft. Therefore, if this were to be implemented into the product, it would be used for close range communications with the device, i.e. set up of the device. It is capable of simultaneously sending and receiving wireless signals to other Bluetooth capable devices, so this would make the product easily compatible with modern mobile devices, which would allow an application to be created which would allow the user to easily interface with the product.

The power of the transmitter determines the range over which a Bluetooth device can operate and, generally, these devices are said to fall into one of three classes: class 1 are the most powerful and can operate up to 100m (~330ft), class 2 (the most common device) operate up to 10m (~33ft), and class 3, the least powerful have a range that does not go far beyond 1m (~3.3ft).

3.3.2 Global System for Mobile Communications (GSM) [K.H.]

It is important to note that the most useful thing about the Global System for Mobile Communications (GSM) is that it is an international standard. This means that if you were to travel to any part of the world, GSM is the only type of cellular service available. Integrating this into the product would allow for the device to operate at any point on the globe, provided there is some form of mobile network coverage.

GSM (originally standing for Groupe Spécial Mobile) was first implemented in 1991 and was developed as a digital system using TDMA technology. Using TDMA, a narrow band that is 30 kHz wide and 6.7 milliseconds long is split time-wise into three slots. Narrow band means channels in the traditional sense. Each conversation gets the radio for one-third of the time. This is possible because voice data that has been converted to digital information is compressed so that it takes up

significantly less transmission space. Therefore, TDMA has three times the capacity of an analog system using the same number of channels.

TDMA is the access method used by GSM, its system provides a number of useful features:

- Uses encryption to make phone calls more secure
- Data networking
- Group III facsimile services
- Short Message Service (SMS) for text messages and paging
- Call forwarding
- Caller ID
- Call waiting
- Multi-party conferencing

GSM operates in the 900 MHz band (890 MHz - 960 MHz) in Europe and Asia and in the 1900 MHz (sometimes referred to as 1.9 GHz) band in the United States. It is used in digital cellular and PCS-based systems.

3.3.3 General Packet Radio Service (GPRS) [K.H.]

General Packet Radio Services (GPRS) is a packet-based wireless communication service that promises data rates from 56 up to 114 Kbps and continuous connection to the Internet for mobile phone and computer users. The higher data rates allow users to take part in video conferencing and interact with multimedia web sites and similar applications using mobile handheld devices as well as notebook computers. GPRS is based on the Global System for Mobile Communication (GSM) and complements existing cellular phone connections and the Short Message Service (SMS).

In theory, GPRS packet-based services cost users less than circuit-switched services since communication channels are being used on a shared-used basis, as packets are needed rather than dedicated to only one user at a time. It is also easier to make applications available to mobile users because the faster data rate means that middleware currently needed to adapt applications to the slower speed of wireless systems are no longer needed. As GPRS has become widely available, along with other 2.5G and 3G services, mobile users of virtual private networks (VPNs) have been able to access the private network continuously over wireless rather than through a rooted dial-up connection.

GPRS also complements Bluetooth, a standard for replacing wired connections between devices with radio connections. In addition to the Internet Protocol (IP), GPRS supports X.25, a packet-based protocol that is used mainly in Europe.

4

4.1 Power Budget Calculation [M.T.]

4.1.1 Solar Panel [M.T.]

The goal of the power budget calculation is to ensure that the device will be able to satisfy the requirements laid out in the verification specification matrix (Appendix A) regarding self-sufficient operation. This entails calculating the minimum size of photovoltaic cell and battery needed for a given solar irradiance, power losses, and power draw.

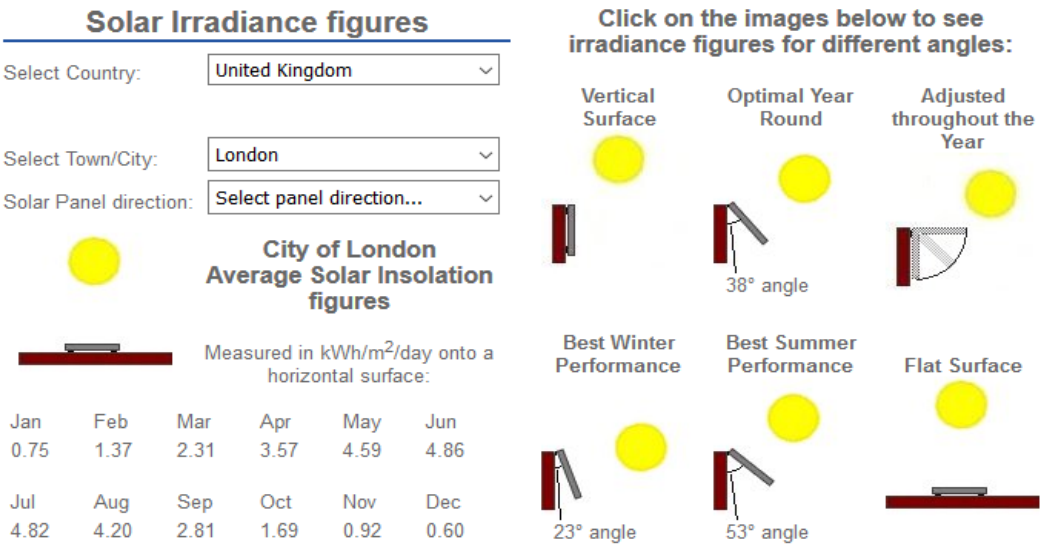


Fig. 10: Solar Irradiance Calculator [27]

We must design our system with the worst case scenario in mind, that being the month of December which receives the least solar radiation. Figure 10 tells us that the average solar insolation (incident solar radiation onto some object) in December, at London's latitude, is only 0.6 kWh/m^2/day onto a horizontal surface. This figure accounts for bad weather.

Figure 11 shows how photovoltaic power potential varies with latitude. Note that generally speaking, using solar irradiance figures at London's latitude (51.509865 °N) as the basis for our calculations guarantees the device's operation at or below that latitude in most of the world. With that in mind, if power consumption can be reduced below that which is estimated in our power budget calculations, the device will be able to operate at even higher latitudes. Please refer to section 4.4.5 on Power Consumption for more details.

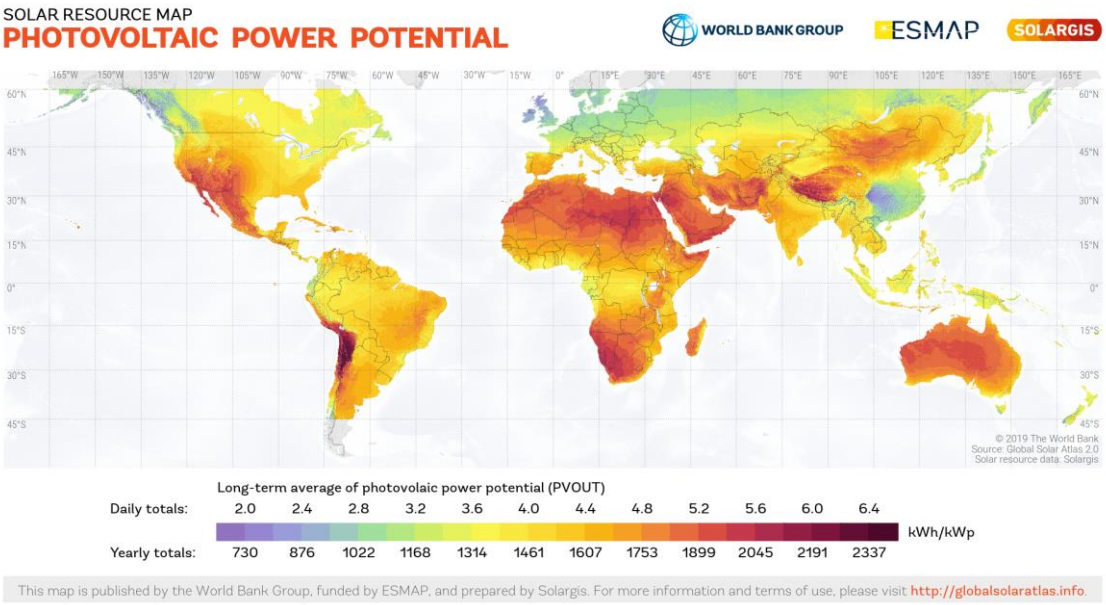


Fig. 11: World's Solar Resource Map [28]

$$0.6\ kWh/m^2/day = \frac{600\ Wh/m^2}{24h} = 24\ W/m^2$$

Assuming solar panel efficiency of 15%:

$$24 \times 0.15 = 3.6\ W/m^2$$

The Solar Power Manager's datasheet [29] states a solar charge efficiency of 73% at 3.7 V 900 mA. Assuming a 27% loss as a result:

$$Usable\ electric\ energy = 0.73 \times 3.6 = 2.6\ W/m^2\ or\ 0.26\ mW/cm^2$$

From various online forum discussions, we estimate the Arduino Nano's idle current draw to be 15mA, and thus an idle power draw of 5*15 = 75 mW. However, note that the Arduino's ATmega328P microcontroller can be put into various sleep modes, which shut down certain unused processes, thereby saving power.

	Active Clock Domains					Oscillators		Wake-up Sources							
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Oscillator Enabled	INT1, INT0 and Pin Change	TWI Address Match	Timer2	SPM/EEPROM Ready	ADC	WDT	Other/O	Software BOD Disable
Idle			X	X	X	X	X(2)	X	X	X	X	X	X	X	
ADC noise Reduction				X	X	X	X(2)	X(3)	X	X(2)	X	X	X		
Power-down								X(3)	X				X		X
Power-save					X		X(2)	X(3)	X	X			X		X
Standby(1)						X		X(3)	X				X		X
Extended Standby					X(2)	X	X(2)	X(3)	X	X			X		X

Notes: 1. Only recommended with external crystal or resonator selected as clock source.
2. If Timer/Counter2 is running in asynchronous mode.
3. For INT1 and INT0, only level interrupt.

Figure 12: Active Clock Domains and Wake-up Sources in different Sleep Modes [30]

The ATmega328P is woken up by one of its interrupt pins sensing an alarm from the Real Time Clock (RTC). This allows us to leave the Arduino in deep sleep for most of the time, conserving power, and only waking up periodically for short periods of time to take a new set of measurements.

We also assume the idle current draw of other components, including the four sensors, SD card reader, RTC, and communication modules to be zero for the purposes of our power budget calculation. In reality, their idle current draws are quite significant, and so measures have to be taken to lower them, such as using a transistor to disable the sensors' voltage rail when not in use. Please refer to section 4.4.5 'Power Consumption' for more details.

Despite having multiple tools to reduce power consumption at our disposal, let us conservatively assume our system's current draw to be 15mA, and its power draw to be 75mW for the time being. We can then calculate the minimum size of solar panel needed:

$$\frac{75\ mW}{0.26\ mW/cm^2} = 288\ cm^2$$

$$\sqrt[2]{288} = 17cm \times 17cm\ minimum\ solar\ panel$$

4.1.2 Li-Ion Battery [M.T.]

Let us now calculate how long a single Samsung 30Q 3.7V 3000mAh Li-Ion battery can power our system for, without being recharged by the solar panel.

The Solar Power Manager converts the Li-Ion battery's 3.7 V to a regulated 5 V output supply, for use by the Arduino and other modules. Its datasheet [31] states that the regulated power supply's efficiency is 89% at 10% load, and 86% at 50% load.

A current draw of 15 mA at 5V, is equivalent to 20 mA at 3.7V. Assuming an 11% efficiency loss in the conversion:

$$\frac{1}{0.89} \times 20 = 23mA$$

Thus, the actual current draw from the battery is 23 mA at 3.7V.
So, given a 3000 mAh capacity battery:

$$\frac{3000\ mAh}{23\ mA} = 144h = 5.5\ days$$

Therefore, a single Li-Ion battery would allow the device to sustain itself for over five consecutive days without recharging its battery with the solar panel.

These power budget calculations now allow us to make informed component selections regarding the size of solar panel (>288cm^2) and size of battery (one 3000 mAh Li-Ion) for our requirements. Note that these calculations depend on our estimated average of 15 mA current draw for the entire system. In section 4.4.5 on Power Consumption, actual power consumption measurements are taken, and various means to reduce them are discussed.

4.2Component Selection [M.T.] [K.H.]

Having completed research on various pollutants, we settled on four sensors that could provide meaningful insight into water quality, whilst remaining within our £100 budget constraint.

For our temperature sensor, we chose the DS18B20 waterproof probe. It is a very common and cheap Arduino-compatible sensor which communicates over a single digital bus, and for which there exists Arduino libraries, such as OneWire. Its claimed measuring range is -55 ~ 110 'C, which is more than enough for our purpose, and its measuring accuracy is +/-0.5'C @ -10~+80'C. It operates on input voltages from 3.0 ~ 5.5V and requires a 4K7 pull-up resistor on the data line. It is available for £2.40 from an eBay vendor. [32]

Figure 13: DS18B20 waterproof temperature probe

It is important to note that while the sensor itself can operate in temperatures up to 125°C, the cable is jacketed in PVC, so it is suggested to keep the maximum temperature under 100°C. As this is a digital sensor there is little signal degradation over long cable distances, while not something that would be present in our device, it is important to note. Furthermore, more than one sensor could be connected to the same bus, as they each have unique IDs.

The DS18B20 sensor is also available from the supplier DFROBOT in the form of a kit, which includes a small adaptor board with a pull-up resistor, with an included JST 2.0mm to PH2.54 female housing connector, for £5.76 [33]. All that the DFROBOT daughterboard does is hold a 4K7 pull-up resistor and more convenient connection points. The cheaper DS18B20 probe form eBay works just as well, and the pull-up resistor can be included on our main PCB.

Our choice of supplier for the TDS and Turbidity sensors was much more limited, as the only supplier with Arduino-compatible sensors that we could find is DFROBOT. We chose to purchase many of our components from them, as they are a reliable supplier, and provide good documentation in their product wikis.

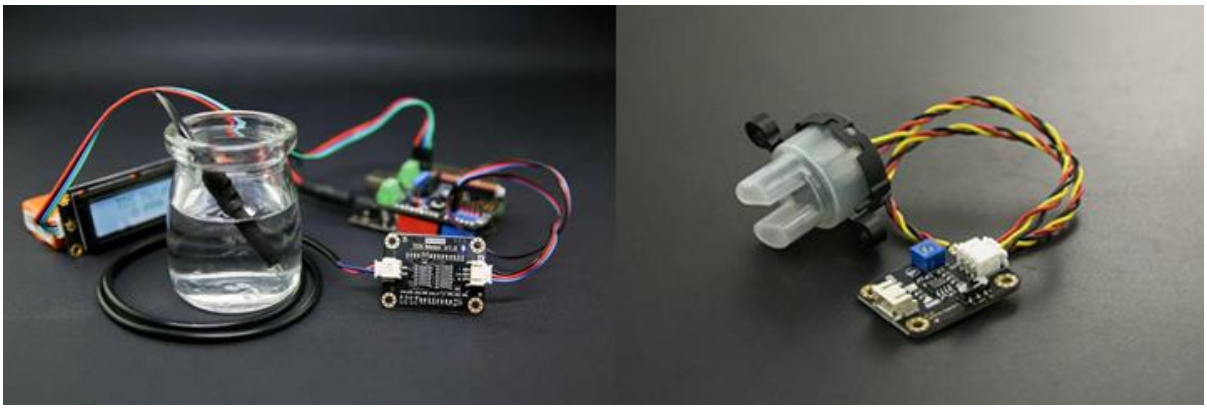


Figure 14: DFROBOT TDS sensor (left) [34], DFROBOT Turbidity sensor (right) [35]

The TDS sensor measures how many milligrams of soluble solids are dissolved in one litre of water, which reflects the cleanliness of the water. The sensor requires an input voltage from 3.3 ~ 5.5V, and has a measurement range of 0 ~ 1000ppm. The probe can not be used in water above 55 degrees centigrade. It has a daughterboard, with a three pin JST 2.0mm to PH2.54 female housing connector cable. The three pins are VCC, GND, and analogue signal output. Thus, the TDS sensor requires just a single analogue pin on the Arduino.

The DFROBOT Turbidity sensor's input voltage is 5V, and output method analogue. Its operating temperature is 5℃~90 ℃, and storage temperature -10℃~90℃. It also has a daughterboard with JST 2.0mm connectors.

Compared with the TDS and Turbidity sensors, we had more choices of pH sensor, as there exists a large number of cheap pH probes on eBay and elsewhere. We chose a laboratory probe from DFROBOT, which was on sale for £11.44.

Figure 15: DFROBOT laboratory [36] (left) and industrial (right) pH sensors [37]

It operates on 5V, has a measuring range of 0-14pH, and an operational temperature range of 0-60 ℃. However, it is only intended for laboratory use, not for extended submerged use. In order to ensure its accuracy, it should be calibrated using a standard solution approximately every six months. When not in use, it should also be kept in a KCL3N solution. The pH electrode needs to be washed with water before use, it is recommended to use deionized water for this. A more expensive pH probe from DFROBOT, costing £43.71, has an industrial electrode suitable for long-term submerged use, and functions otherwise identically to the cheaper laboratory probe.

We had to make a significant compromise due to our £100 budget constraint and chose the laboratory pH probe instead of the industrial probe. The laboratory probe acts as a proof of concept and will work fine in the short-term for our prototype. For a real product, we would strongly suggest that the client consider increasing the budget to about £130 in order to accommodate the added cost of the industrial pH probe.

For our microcontroller, we considered the Arduino family of boards, (UNO, Nano, Pro Mini, etc). A significant advantage of using an Arduino board is the availability of online resources and libraries, and the Arduino IDE. Given the stringent low power draw requirements of our project, we chose to use an Arduino Nano rather than the more popular Arduino UNO development board. The two boards share the same ATmega328P microcontroller, and the same amount of on-board memory (32kB of flash and 2kB of SRAM), with the main differences being that the Nano removes does away with some features (e.g. DC power plug), resulting in a significantly reduced form factor and power draw.

There are even smaller Arduino boards than the Nano, such as the Pro Mini, which exists in 5V and 3.3V variants. However, the Pro Mini does away with some useful features, such as the USB host controller, which requires the use of an FTDI programmer to upload code. Furthermore, there are diminishing returns in terms of power consumption, as the Pro Mini 5V variant consumes almost as much power as the Nano.

The Pro Mini 3.3V variant has lower power consumption but given that our Turbidity and pH sensors can only operate with an input voltage of 5V, we chose a 5V Arduino instead of 3.3V. It would be possible to use a 3.3V Arduino board with 5V peripherals, if a 5V boost converter were used to power the 5V peripherals, but the gains would likely be offset by the inefficiency of the boost converter and doing so would add an unnecessary component and extra cost.

The Arduino foundation makes its PCB designs free and open source, meaning that many cheaper clones of its boards exist on the market. We opted to use an Arduino Nano clone rather than an original Arduino, given our limited budget. The clones are functionally equivalent to the original boards. One of the few differences to note is that our Arduino Nano clone uses the old Bootloader, which necessitates selecting "Tools > Processor > ATmega328P (Old Bootloader)" within the Arduino IDE.

Figure 16: Arduino Nano clone [38], 5V Solar Panel [39]

We also considered some non-Arduino boards, such as the ESP32, and the Teensy 4.1. The ESP32 has more flash memory, and has built in Wi-Fi and Bluetooth, and would also be a suitable choice for our project, despite not needing Wi-FI. The Teensy 4.1 is a much more powerful board with additional features and more I/O, but is more expensive, and we do not need the additional processing power for our project. We ended up choosing an Arduino board because of its price and ease of use with the Arduino IDE without any modifications.

From our power budget calculation in section 4.1, we know that that a 17*17cm or larger solar panel and a single 3.7V 3000 mAh 18650 Li-Ion battery satisfy our requirements.
We chose a 175*172mm 4W 5V 0-800mA polycrystal EVA laminated solar panel, costing £6.24 from an eBay vendor. It satisfies our minimum size requirements, outputs 5V, is lightweight (88g), and suitable for outdoor use.

Any regular 18850 Li-Ion battery would be suitable. We chose a Samsung 30Q 3000mAh one [40], as it is from a reliable brand, and was well priced. Some 18650 batteries from unknown brands claim to have capacities as high as 5000mAh, but these are highly inflated claims. We also purchased a battery holder for the battery. [41]

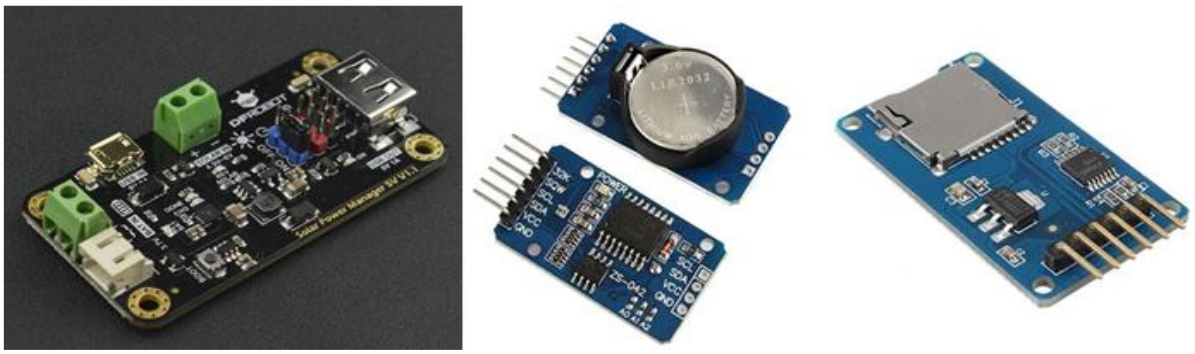


Figure 17: 5V Solar Power Manager [42], DS3231 RTC [43], Micro SD card reader [44]

The Solar Power Manager from DFROBOT serves the dual purpose of charging the Li-Ion battery from the solar panel and providing a 5V regulated output for the Arduino and other components. It features MPPT (Maximum Power Point Tracking) functionality, maximizing the efficiency of the solar panel by adjusting load resistance. The module also employs various protection functions for the battery, solar panel and output. These include over charge/over discharge/over current/reverse connection protection for the battery, short circuit/over current/overheat protection for the 5V output, and reverse connection protection for the solar panel. The module's cost is £6.07. We considered some cheaper alternatives to the Solar Power Manager, such as using a separate cheaper Li-Ion battery charger and boost converter but found that none were suitable for simultaneous charging and discharging of the battery, and none had MPPT.

An integral part of any data logger project is a Real-Time Clock (RTC), without which the Arduino Nano has way of knowing absolute time. It can only keep track of time passed since it was powered on. The function millis() returns the number of milliseconds passed since the Arduino board began running the current program. An RTC module contains a crystal oscillator and its own backup battery (such as a LIR2032 battery). It need only be set up once, after which it can keep track of time, even when disconnected from power thanks to its small backup battery. This allows for all of our sensor data to be time stamped.

Because our project required that the microcontroller be in a deep sleep state the majority of the time to conserve power, an RTC with a programmable alarm must be chosen, as this alarm is used to wake the Arduino up from deep sleep via an interrupt enabled pin. Thus, we chose the DS3231 RTC, which has two programmable alarms (of which we only need one). We can set the alarm such that the device takes measurements at a specific time of month/day/hour/minute, or after a specified time interval (e.g. every minute or 10 minutes). The RTC uses the I2C bus.

An SD card reader is necessary to save data to an SD card. It needs exclusive use of the SPI bus, which is not an issue as we have no other modules using the bus. Data will be saved to the SD card in a CSV (Comma-separated values) file. When taking measurements once per minute, data use is only 60 kB per day, or 21 MB per year. So, an extremely small SD card can be used.

For mobile communications, we have decided to implement two types: Bluetooth and GSM/GPRS. The latter is intended as the primary form of communication, as it allows for completely remote use, whereas Bluetooth acts as a redundant measure, to be used when in close proximity with the device.

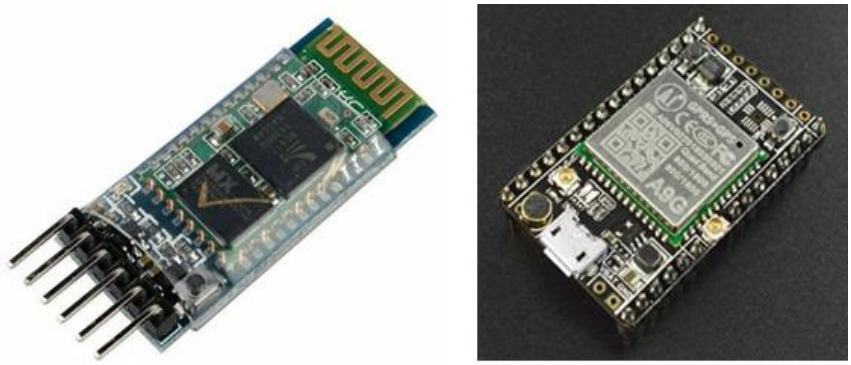


Figure 18: HC-05 Bluetooth module (left) [45], A9G GSM/GPRS+GPS module (right) [46]

For our Bluetooth module, we chose the very popular HC-05. It has an on-board 3.3V regulator chip, so it accepts input voltages between 3.6-6 V. Its Rx and Tx pins connect to the Arduino's Rx and Tx UART pins for serial communication.
For GSM/GPRS communication, we chose the A9G module from DFROBOT. Unfortunately, there are relatively few options available on the market, and the A9G is lacking in good English documentation, making it more difficult to work with. The A9G comes with a GPS receiver too, which we could use to track the device's location and send an alert if it moves away from its intended position, such as if its tether or anchor breaks.

Both the Bluetooth and A9G modules use serial communication (Rx/Tx). The Arduino has native serial support via a piece of hardware called a UART (Universal asynchronous receiver/transmitter) on pins 0 and 1. However, no more than one module can be connected to the Arduino's UART. Thus, to avoid conflicts, one must either ensure that only one module is in use at any given time, or use the SoftwareSerial library to use software to replicate the functionality of the UART on digital pins of the Arduino. Using the SoftwareSerial library comes at the cost of greater memory and flash use. Because the A9G module's power draw is significant, a MOSFET will be used to turn on/off power to it. It will only be turned on periodically to send an update to the database.

Having now selected all of our components, a block diagram overview and a circuit diagram can be created as follows:

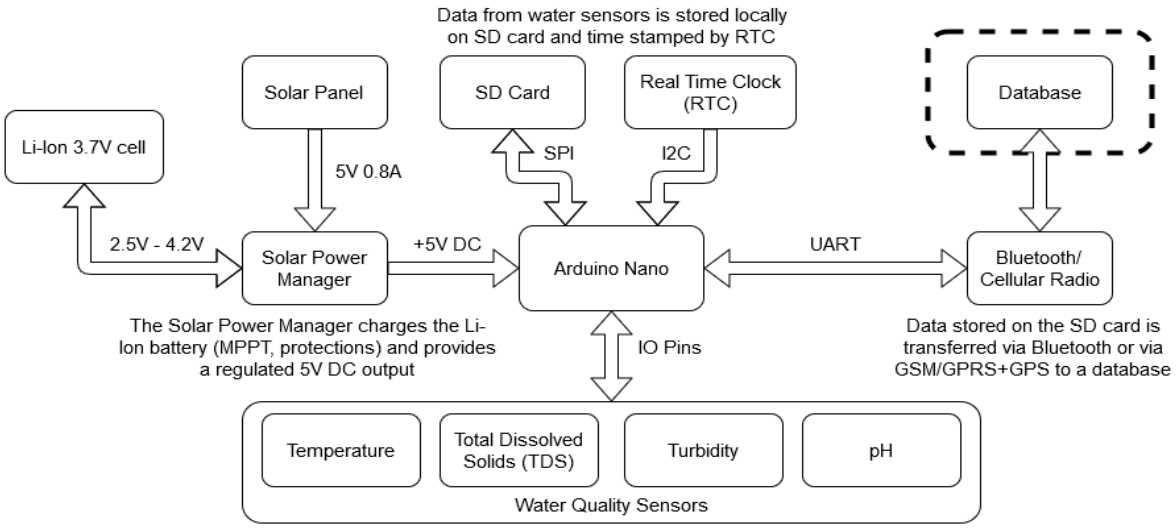


Figure 1: Hardware Diagram

Figure 19: Hardware block diagram [M.T.] [L.W.]

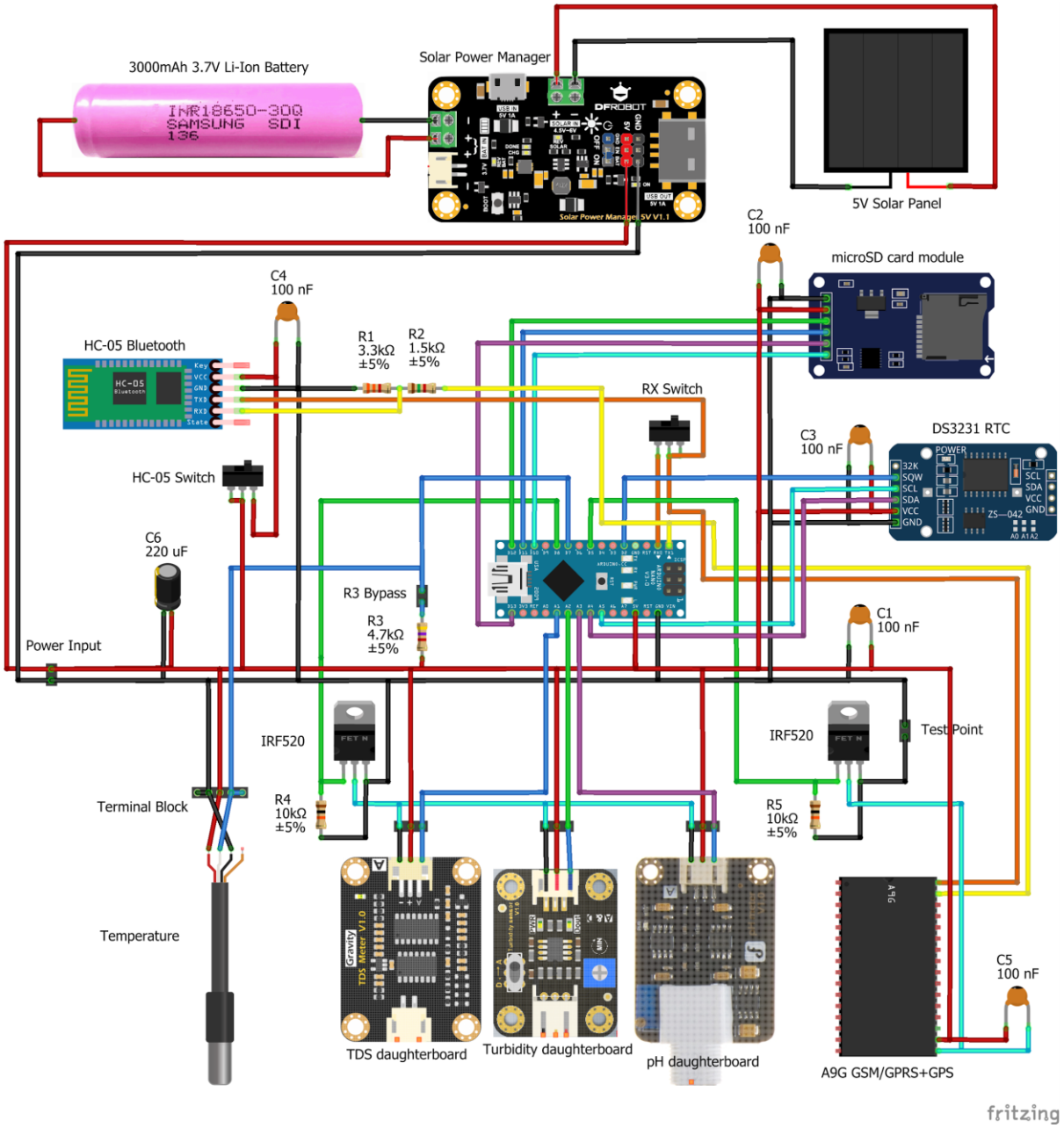


Figure 20: Hardware circuit diagram [M.T.]

Figure 20 is a complete hardware circuit diagram. An HC-05 switch is used to turn Bluetooth pairing on/off. An Rx switch is used to disconnect the HC-05 and A9G's Tx ports from the Arduino's Rx port. These lines must be disconnected while compiling code to the Arduino but can be left connected permanently thereafter. The two IRF520 MOSFETs are used to cut power to peripherals when not in use. The three pin temperature sensor connects to the board via a five pin terminal, such that the user can either solder on a three pin header onto the central pins, or solder on a wider three pin screw terminal, depending on which variant of sensor they have (DFROBOT or other). The R3 bypass allows for the temperature sensor's pullup resistor to be disconnected if using the DFROBOT sensor with its daughterboard. Various capacitors are used to condition the power supply and shunt higher frequencies to ground. Custom Fritzing parts had to be created for the circuit diagram, which are available in our project's GitHub repository.

4.2.1 Bill of Materials [M.T.]

Category	Part	Part Description	Unit	Notes
Sensors	Temperature	Waterproof DS18B20 Sensor Probe	**£2.40**	Uses 1 Digital IN. Input 3.0-5.5V. 4K7 pull-up data
	TDS Sensor	Analog TDS Sensor/Meter for Arduino	**£6.84**	Uses 1 Analogue IN. Input 3.3-5.5V
	Turbidity Sensor	Analog Turbidity Sensor For Arduino	**£7.60**	Uses 1 Analogue IN. Input 5V
	pH Sensor	Analog pH Sensor For Arduino	**£11.44**	Uses 1 Analogue IN. Input 5V
Arduino	Arduino Nano	MakerHawk Nano Microcontroller	**£4.33**	5V IN 16MHz, old bootstrapper
	Real Time Clock	DS3231 RTC I2C with Alarm	**£3.24**	With 2 time of day alarms
	SD Card Reader	Micro SD Card Reader Module	**£1.74**	Needs exclusive use of SPI interface.
Power	Solar Panel	4W 5V 0.8A Solar Panel 175x172mm	**£6.24**	Area: 175x172mm=300 cm^2.
	Power Manager	Solar Power Manager 5V	**£6.07**	MPPT, protections, 5V regulated output
	Li-Ion Battery	Samsung 30Q 3.7V 3000mAh Li-Ion	**£4.69**	Unprotected
	Battery Holder	18650 Li-Ion Battery Holder (1 Battery)	**£0.97**	With leads
Comms	Bluetooth	HC-05 Android Bluetooth Transciever	**£4.60**	Android only. Onboard regulator. UART
	GSM/GPRS+GPS	A9G GSM/GPRS+GPS Module	**£8.83**	Also UART (multiplex with tri-state buffers?)
	Enclosure	5L Rectangular Bucket	**£5.99**	H15cm x W24cm x D21cm
		4x Marine Grade Eye Plates	**£3.99**	Anchor points
		Marine Grade Wire Mesh 30x30cm	**£7.99**	316 grade stainless steel
		20x Self-Adhesive PCB Pillars 12.7mm	**£5.49**	Panel thickness 1.60mm, Hole dia 4.00mm
	MOSFET	IRF520 MOSFET	**£ -**	Available in lab. Used to cut power to sensors.
Total			**£92.45**	

Figure 21: Bill of materials [M.T.]

Note that all prices were correct as of the time of purchase, but that they are subject to change. Most non-DFROBOT components are available from many different vendors, the cheapest of which has been chosen in each case. The major concession made because of the limited budget is the choice of a laboratory pH probe instead of a more robust industrial pH probe. This prototype serves as an expandable platform. More sensors can easily be added, such as Oxidation Reduction Potential, Conductivity, and Dissolved Oxygen sensors), by drilling more holes in the enclosure.

4.3 Enclosure

4.3.1 Design [M.T.]

In the first few weeks of the project, between 21/10/2020 and 14/11/2020, we held brainstorming sessions in our meetings to discuss ideas for our enclosure design. Various team members also created drawings depicting their ideas, a few examples of which are shown below:

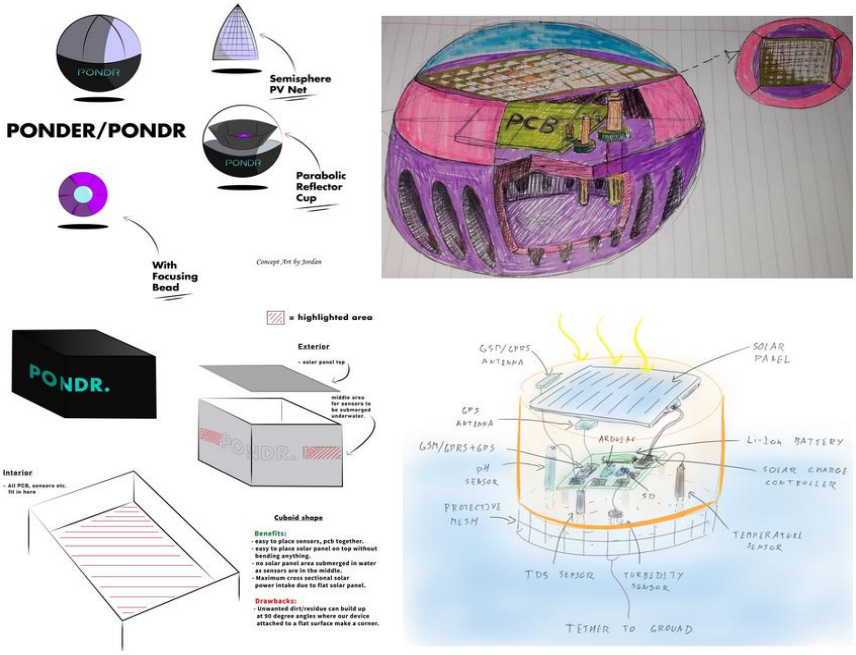


Figure 22: Enclosure ideas [J.E.] [V.B.L.] [M.T.]

In our brainstorming sessions, our initial intention was to 3D print a custom enclosure. The main advantage is that it gives great control and flexibility over the design. However, 3D printing has numerous disadvantages in the context of our project, such as waterproofing, environmental durability, and cost. 3D printed waterproof parts require a great deal of extra care in the design, slicing, and printing stages to ensure that there are no gaps between layers. Furthermore, since our project is intended for extended water immersion, we would have to use a filament that could withstand the long-term effects of biofouling. Since 3D printing results in a porous textured plastic, biofouling was of major concern, and an inherent problem with FDM (Fused Deposition Modelling) 3D printing. FDM is by far the most common and most accessible method of 3D printing, where extruded filament is added in layers. Other methods of 3D printing exist, such as Stereolithography (SLA) which prints an object by shining a UV laser onto a bath of liquid resin to harden the resin layer by layer. SLA can result in higher quality parts but is a less accessible method.

Given that our project's design requirements were not very stringent with regards to the size or shape of the enclosure, we decided that modifying a pre-existing waterproof enclosure to fit our needs would be a better option than 3D printing our own. We considered a few different options, such as using PVC plumbing parts, circular paint buckets, or food containers. In the end, we identified the following rectangular bucket as the most suitable choice:

Figure 23: 5L rectangular bucket (left) [47], oblong eye mounting plates (right) [48]

Its cost of £5.99 fit within our budget. Its dimensions of H15cm x W24cm x D21cm are large enough to fit the 17.5x17.5cm solar panel on the lid, and the 15cm tall pH sensor within. Its intended use is for carp fishing, so it has a rugged construction and a flat watertight lid.

This 5L rectangular bucket was just large enough for our needs without wasting too much space. If we had chosen a circular bucket instead of a rectangular one, it would need a lid diameter of at least (18x2)/sqrt(2)=25cm to fit the square solar panel. The smallest size of circular bucket with the required lid size are 10L buckets, which are twice the volume of a rectangular bucket that fits our requirements.

If this were not a prototype, the capital investment to fabricate custom enclosures to our specifications might be worth it to save having to drill holes in and paint the enclosure. Details of the construction process are in section 4.3.2.

Four oblong eye mounting plates serve as mounting points for the user to use to tether or anchor the device. They are made from a marine grade stainless steel.

We decided to use wire mesh to create a box protecting the four sensor probes on the underside of the enclosure, to avoid the sensors becoming damaged or entangled with debris.

The wire mesh we chose [49] is made from marine grade 316 stainless steel, which is an alloy of steel with added molybdenum making it more resistant to chloride corrosion as opposed to the more common 304 grade steel. We purchased a 30x30cm sheet, from which we make a box by folding and cutting the wire mesh, as shown in section 4.2.2.2.

To avoid drilling more holes to affix the PCB/Veroboard and the sensors' daughterboards to the inside of the enclosure, we used adhesive PCB standoffs. [50]

4.3.2 CAD [J.E.]

To initially visualise our device, the real-life version was modelled in Autodesk Fusion 360. The Design shows the placement of holes for the sensors, eye plates, and solar panel cable.

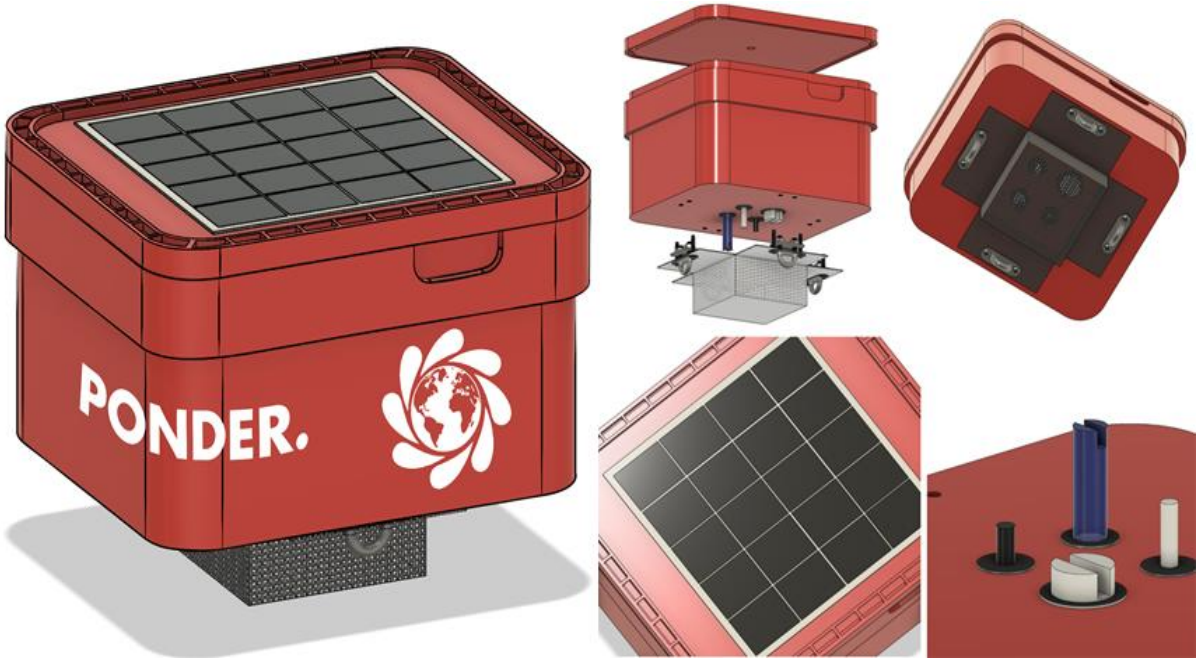


Figure 24: Various views of CAD drawing [J.E.]

4.3.3 Construction [M.T.]

4.3.3.1 Bucket [M.T.]

Four holes were drilled in the bottom of the enclosure, one for each sensor, with diameters as follows: temperature 6mm, TDS 6.5mm, turbidity 22mm, pH 12mm. The sensors are held in place with glue, which also prevents water ingress. A further eight holes were cut to mount the eye plates, which also serve to hold the wire mesh in place. A small hole was cut in the lid to pass through the solar panel's cable. The four sensors' daughterboards are affixed to the inside of the bucket with adhesive PCB pillar mounts. The battery, Solar Power Manager, and PCB/Veroboard are attached to the underside of the enclosure's lid. Having these components on the underside of the lid makes accessibility easier, as the user can remove the lid without any tools by simply disconnecting the four sensor cables, as opposed to needing a screwdriver to detach the solar panel cable from the Solar Power Manager's screw terminal.

Figure 25: Enclosure build before painting and without Veroboard [M.T.]

4.3.3.2 Mesh [M.T.]

A protective wire mesh cage for the four sensor probes on the bottom of the device was fashioned out of a 30x30cm sheet of marine grade stainless steel mesh. This serves to protect the sensitive probes from damage and obstruction. The mesh cage was built by cutting and folding the mesh sheet in a specific manner. The walls of the cage are two layers thick to provide extra structural support for when the device is sitting on the mesh cage. The edges of the mesh cage are folded inwards to prevent fraying. It is affixed to the bottom of the enclosure using the same eight bolts used for the eye plates.

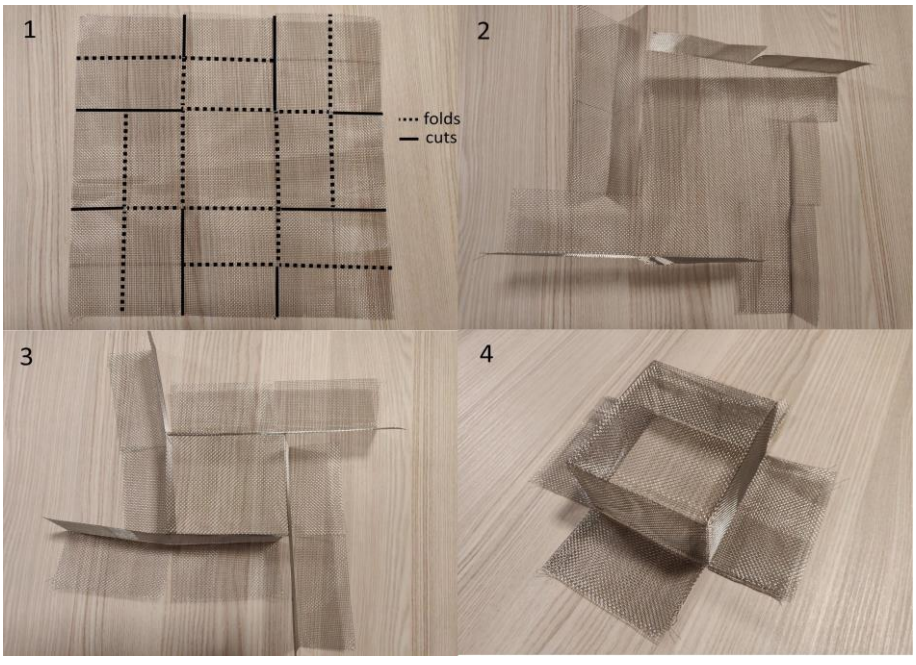


Figure 26: Wire Mesh Building Process [M.T.]

4.3.3.3 Water Ingress [M.T.]

Glue was used to prevent water ingress through the thirteen drilled holes. Various types of glue were tested, but the best performing one was found to be simply hot glue, as it does not need to be in between two mating surfaces to work well.

A water ingress test was performed with 1kg of ballast attached to the eye plates. Without this ballast, the device sits very high up in the water, as it only weighs 1kg and so displaces very little water. This is concerning as the device could be overturned by a large wave for instance if it is not well anchored using the 4 available eye plates. There is no convenient way to add ballast within the device itself as there is little available room. To mitigate this risk the user should ensure that the device is properly tethered, and we should recommend that the user adds around 1kg of ballast to the device (attached to the eyelets) to have it sit around half-way submerged. By the Archimedes principle, adding 1kg of ballast to a 1kg device displaces a total of 2kg of water, which means around 50% submersion for a 5L container. (The enclosure is tapered near the bottom, so a 2L is near 50% submersion)

The device was painted orange to fit the colour of other research buoys and to be more visible. If this weren't a prototype, the enclosure could be made from orange plastic instead of having to paint it.

4.4 Arduino Firmware [M.T.] [L.W.]

Note that the full firmware code is available in Appendix J, but excerpts of the code are shown in this section to aid understanding.

4.4.1 Breadboard [M.T.]

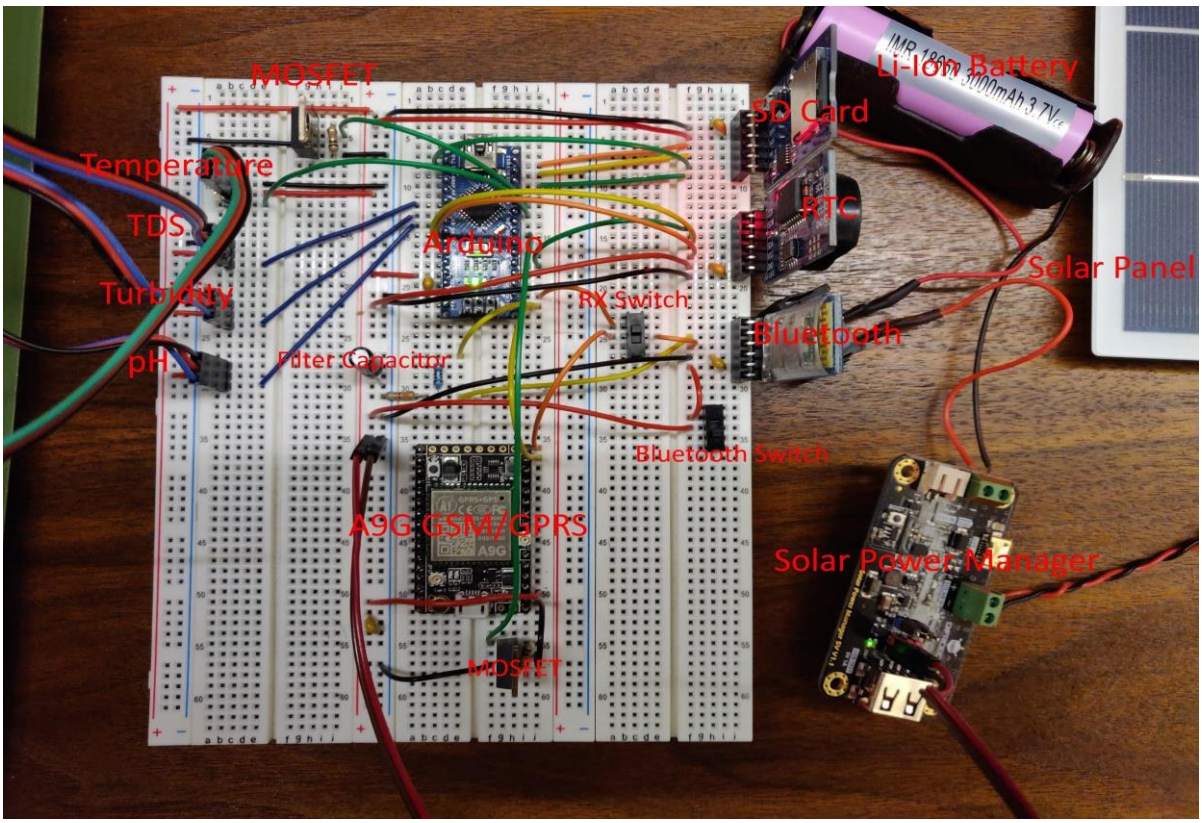


Figure 27: Breadboard implementation [M.T.]

The circuit was initially built on a breadboard, and we encountered a few issues. For instance, initially, the SD card reader wasn't working. We found the issue to be that with the Vcc and GND rails shared between 3 breadboards daisy chained together, there was a significant voltage loss (from about 4.75V to <4V). This is due to breadboards' characteristic capacitance, inductance, and resistance. Thus the SD card reader wasn't receiving enough power. We mitigated the issue by:

1. Connecting modules more directly to a central Vcc and GND rail rather than daisy chaining rails.
2. Adding 220uF electrolytic filter capacitors across the power supply rails to condition the power supply, keeping the voltage throughout the circuit relatively constant.
3. Adding 100nF ceramic capacitors across the power supply rails, and each IC, to shunt high(er) frequency noise to ground.

As a result, the voltage rails were kept closer to 5V, and with less ripple, solving the SD card reader issue. Keeping the sensor voltage rail close to 5V was also very important for the analogue sensors' readings, which depend on receiving precisely 5V as a reference voltage. When powered via the Arduino's USB, the primary power rail is only 4.77V, whereas when powered directly by the Solar Power Manager, it is much closer to 5V. We also created a global variable in the firmware code (#define Vref 4.96) which can be set as the voltage on the sensor power rail to get more accurate readings.

Additionally, a simple switch had to be added to the breadboard to quickly and easily disconnect the serial Rx line from the Bluetooth and A9G modules when uploading code to the Arduino.

4.4.2 SD Card [M.T.] [L.W.]

The Arduino initialises the SD card once in the setup function, checking that the card is present and can be written to. It checks whether a DATA.csv file already exists, and if not, it creates a new file and adds headers to it for "Day, Month, Year, Hour, Minute, Second, Temperature, TDS, Turbidity, pH".

In the loop function, the function writeToCard() is called each time a new set of data is collected. This function checks that the SD card is still present, writes the data to it and to the serial monitor, and closes the file.

```
void writeToCard(){
if(SD.exists("data.csv")) // check the card is still there
{
  CSVfile = SD.open("data.csv", FILE_WRITE);
  if (CSVfile)
  {
    CSVfile.println(dataString); // adds a new line of data
    CSVfile.close(); // close the file

    Serial.println();
    Serial.println(F("Data written to file:")); //the next few lines print data to serial monitor for debugging
    Serial.println(F("(Day,Month,Year,Hour,Minute,Second,Temperature,TDS,Turbidity,pH)"));
    Serial.println(dataString);
    Serial.println();
    Serial.flush();                       // Ensure all characters are sent to the serial monitor
  }
}
```

4.4.3 Real Time Clock (RTC) [M.T.]

The setup function sets the RTC's time to compile time. This only needs to be done once, and in the future, it only sets the time again if it detects that the RTC lost backup power.
Each time a new set of measurements is taken, the ReadTime function is used to create a single string concatenating the date and time with commas separating values.

```
//-----------------------------------------------------------------------------------------------------------
String ReadTime()
{
DateTime now = rtc.now();
return String(now.day(), DEC) + "," + String(now.month(), DEC) + "," + String(now.year(), DEC) + "," +
String(now.hour(), DEC) + "," + String(now.minute(), DEC) + "," + String(now.second(), DEC);
//returns one long string, with commas seperating day,month,year,etc values
}
```

This time string is then concatenated again with the outputs from each sensor, to create the comma separated data string which is written to the SD card.

```
dataString = String(Time) + "," + String(Temp) + "," + String(TDS) + "," + String(Turb) + "," + String(pH);
writeToCard(); // save to SD card
```

The RTC's alarm is also used as an interrupt to wake the Arduino from deep sleep. The function enterSleep takes care of the MOSFETs, turning off power to the sensor rail and A9G. When an interrupt is detected, function alarm_ISR is run. The alarm can be set for a specific time, or for a time delay from the current time.

4.4.4 Sensors [M.T.]

4.4.4.1 Temperature Sensor [M.T.]

The function ReadTemp returns the temperature from the DS18B20 in degrees celsius. The code only depends on the OneWire library, which facilitates communication via a single digital pin, as opposed to some more complex libraries. The temperature sensor sometimes returns an anomalous result of 85 degrees after startup. We solved this issue by simply repeating the measurement if this occurs. We don't expect this simple fix to cause any issues as the PONDER device will never reach 85 degrees celsius in normal use.

4.4.4.2 TDS Sensor [M.T.]

The function ReadTDS makes use of the GravityTDS library. The function compensates for temperature, as the sensor's reading can vary slightly with temperature.

```
//-----------------------------------------------------------------------------------------------------------
float ReadTDS(float Temp)
{
  gravityTds.setTemperature(Temp);  // set the temperature and execute temperature compensation
  gravityTds.update();  //sample and calculate
  float tdsValue = gravityTds.getTdsValue();  // then get the value
  return tdsValue;
}
```

4.4.4.3 Turbidity Sensor [M.T.]

The Arduino reads the Turbidity sensor's reading as an analogue value between 0-1023, and we convert that to a voltage from 0-5V. We then calculate the Turbidity given an equation, specific to this sensor, that maps the voltage to a Turbidity value in NTU.

```
//--------------------------------------------------------------------------------------------------------------
float ReadTurb()
{
  int sensorValue = analogRead(TurbPin);
  float voltage = sensorValue * (VRef / 1024); //Converts the analog reading (0-1023) to a voltage (0-5V)
  float Turb = -1120.4*voltage*voltage+5742.3*voltage-4352.9;
  if (voltage < 2.5) //for extremely dark liquids, the sensor cannot read beyond 3000 NTU, or under 2.5V
  {
    return 3000;
  }
  else if (voltage > 4.2) //A voltage of 4.2V or more means 0 NTU
  {
    return 0;
  }
  else
  {
    return Turb; //Between 2.5V and 4.2V, the sensor measures from 0-3000 NTU
  }
}
```

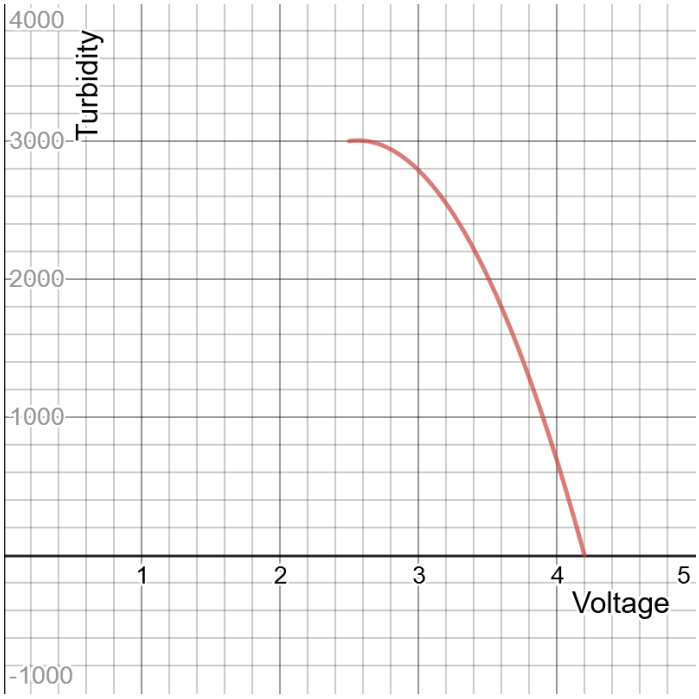


Figure 28: Graph of Turbidity against Voltage [M.T.]

The sensor's measurements have an upper and lower bound of 3000 and 0 NTU, respectively.

4.4.4.4 pH sensor [M.T.]

Much like the Turbidity sensor, we treat the pH sensor as a simple analogue sensor, converting its reading to a pH value using an equation specific to the probe used. The pH value can be calibrated

by placing the probe in a solution of known pH, and setting the variable pHOffset such that the reading is correct.

```
//-----------------------------------------------------------------------------------------------------------
float ReadpH()
{
  int sensorValue = analogRead(pHPin);
  float voltage = sensorValue * (VRef / 1024); //Converts the analog reading (0-1023) to a voltage (0-5V)
  float pH = 3.5 * voltage + pHOffset;
  return pH;
}
```

Note that one of the reasons we implemented this method is that it uses very little dynamic memory and has no library dependencies.

4.4.5 Power Consumption [M.T.]

The Arduino Nano can be powered either via the Vin pin, or via the 5V pin. The Vin pin requires an input voltage of 7V-12V, and uses a linear regulator to drop that to 5V, whereas the 5V pin bypasses the linear regulator, but requires a regulated 5V supply to avoid damaging the Arduino. Powering directly via the 5V pin is evidently much more power efficient than using the linear regulator.

When powering the Arduino with USB, we get only 4.77V on the main voltage rail, and 4.6V on the sensor voltage rail. But using the Solar Power Manager, we get 5.05V on the main voltage rail, and 4.96V on the sensor rail. Thus all measurements below are taken with the SPM.

CLKPR	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08
clock division factor, up to 256	1	2	4	8	16	32	64	128	256
freq, MHz	16	8	4	2	1	0.5	0.25	0.13	0.06
empty sketch, mA	19	16	14.3	13	12.1	11.7	11.5	11.4	11.4
SLEEP_MODE_PWR_DOWN, mA	6.3	6.3	6.3	6.3	6.3	6.3	6.3	6.3	6.3

Figure 29: Clock Speed and Deep Sleep effects on Arduino Nano current draw [M.T.]

Figure 29 shows that reducing Arduino Nano's microcontroller's clock speed from the original 16MHz down to 62.5kHz results in diminishing marginal returns of current draw. By contrast, putting the Arduino into deep sleep mode, reduces current draw to 6.3mA regardless of clock speed. Using deep sleep mode is evidently preferable, given that we can achieve it using the RTC's alarm as interrupts.

Part	Current draw, mA, 5V
Arduino in deep sleep	6.3
SD card reader	1.7
RTC	6.7
Temperature sensor	0.0
TDS sensor	3.7
Turbidity sensor	11.2
pH sensor	12.3

Figure 30: Current draw of individual components [M.T.]

The measured current draw of all core components (Arduino, SD card, and RTC) is 14.7mA at 5V. The sensors draw 27.2mA altogether. The DS18B20 temperature sensor draws no appreciable current as it can operate in parasitic mode. Thus, our goal of 15mA can be achieved by cutting power to the TDS, Turbidity, and pH sensors when not in use. This can be accomplished by using an IRF520 MOSFET, driven by a digital pin from the Arduino. The same can be done for the A9G module, which consumes 60mA in use.

We can arbitrarily choose the time period between measurements, which take less than a second to complete. If measurements are taken every minute, the sensors draw 27.2mA only 1/60th of the time, averaging less than half a milliamp. If the time period between measurements is greater than one minute, the average current draw from non core components is even lower. Thus, we can effectively disregard the power draw from non core components, and state that the device's average power draw is 14.7mA.

	mA	V	mW	Efficiency
current draw 5V	14.7	5	73.5	0.875
battery current draw 3.7V	22.7	3.7	83.99	

Figure 31: Current draw of individual components [M.T.]

Measuring the current draw from the 3.7V Li-Ion battery, we can calculate the Solar Power Manager's boost converter to be 87.5% efficient, which is perfectly within specifications. Given the solar irradiance calculations in section 4.1, we can calculate that the minimum solar panel area is 283cm^2. Given our actual solar panel size of 301cm^2, our power consumption is within our solar budget.

There are multiple methods by which once could reduce power draw even further. We could de-solder the power indicator LEDs on the Arduino Nano, SD card reader, and RTC. The RTC can also be forced to go into low current timekeeping mode, in which it is powered by its LIR2032 battery, by connecting the RTC's VCC to an Arduino digital pin. According to the datasheet, Bit 6 (Battery-Backed Square-Wave Enable) of control register 0Eh, can be set to 1 to force the wake-up alarms to occur when running the RTC from the back up battery alone [51]. One could even avoid using an Arduino Nano and instead build a bare bones board with the ATmega328P processor on its own, without many of the Arduino's peripherals. None of these methods were necessary given that our prototype meets specifications by using deep sleep and MOSFETs, but they could be explored to extend the range of working environments.

4.4.6 Code Optimisation & Memory Use [M.T.]

The Arduino Nano's ATmega328P microcontroller has only 32kB of flash to store the program, and 2kB of SRAM dynamic memory on-board. We very quickly reached the SRAM limit, which caused instability problems when above about 80% usage.

We resolved this issue by reducing our dependence on Arduino libraries wherever possible, as they can take up a lot of memory for functions which we do not need. For instance, many libraries exist to abstract the function of the DS18B20 temperature sensor, but we used only the code we need and reduced our dependency to only the OneWire library.

We also have many Serial.println statements in our code, which print messages to the serial monitor for debugging purposes. The strings in these statements are stored as global variables,

using up valuable dynamic memory. The F() macro creates a pointer to an address in the program memory, thus forcing the microcontroller to store these strings in the 32kB of program memory instead of the 2kB of dynamic memory.

```
Sketch uses 25368 bytes (82%) of program storage space. Maximum is 30720 bytes.
Global variables use 1900 bytes (92%) of dynamic memory, leaving 148 bytes for local variables. Maximum is 2048 bytes.
Low memory available, stability problems may occur.
```

```
Sketch uses 25584 bytes (83%) of program storage space. Maximum is 30720 bytes.
Global variables use 1472 bytes (71%) of dynamic memory, leaving 576 bytes for local variables. Maximum is 2048 bytes.
```

Figure 32: Low memory error (top), Reduced SRAM usage (bottom)[M.T.]

Using F() macros trades an increase in program storage space use from 82% to 83%. for a reduction of dynamic memory use from 92% to 71%. Thus, using these methods, we can reduce our memory use to within our limit.

If we were running low on program storage space, we could use an EEPROM, which is a type of erasable read only memory that can be used to store data which the program uses often but doesn't need to change.

4.4.7 HC-05 Bluetooth Module [M.T.][L.W.]

Data can be transmitted wirelessly to a mobile phone via the HC-05 Bluetooth module. This can be used when the user is in close proximity to the device, and acts as a backup if it is not possible to transmit data via GSM/GPRS. This Bluetooth communication is also useful for debugging, as the user can tell that the device is working properly by looking at the data collected in real time. The user must have physical access to the device, as the Bluetooth HC-05 module must be manually turned on via a switch, as it is normally turned off to conserve power. At present, this switch accessing the switch requires the lid to be removed, but it could easily be replaced by a waterproof latching push button in the lid to improve accessibility [52]. Another push button could be added on the lid to turn the entire device on/off too. At present, the Bluetooth module is configured to transmit data in real time, but it could also be configured to send all data stored on the SD card if it receives a specific AT command from the user.

4.4.8 A9G GSM/GPS Module [L.W.]

The A9G module is used to allow the Arduino to update the database over cellular data. It uses 2 digital pins on the Arduino to act as serial pins, one for transmission, and one for receiving. One disadvantage of using SoftwareSerial to emulate UART pins is that the baud rate is limited; the baud rate used in the A9G is 115200. This means that when a response from the A9G is echoed to the Serial Monitor, there is an occasional data transmission error, changing a character, usually by one decimal value (e.g. '.' Becomes '/'). This error only affects the response and not the commands sent by the Arduino, meaning performance isn't affected, only debugging. The A9G module is controlled using AT Commands, which allow modems to interact with PCs and microcontrollers, this made it difficult to use, due to a lack of English documentation and code libraries. Another issue is that the A9G does not support every standard AT command and has different error codes to most modems, this meant that debugging the A9G involved a large amount of trial and error.

When first coding the A9G module, we designed a loop that checked if it could read the details, such as ID and signal strength, of the SIM card that has been inserted and display it on the serial monitor. At first, it returned errors stating that it was unable to read the SIM card, occasionally displaying the SIM ID. After changing the SIM card to a different carrier, it was able to display the details correctly. The next stage was to test sending a SMS message, to prove that it was

possible to use the SIM card and not just retrieve information from it. This step was not challenging due to the simple syntax of the relevant AT command, the only difficulty was determining the terminating character required when writing the message as it is often not mentioned. Once we discovered that it was ASCII character 26, we were able to send SMS messages.

The final step was to code the modem to make GET requests, this required the cellular data functionality of the SIM. In order to achieve this, first the mobile data must be enabled, next the APN needs to be configured, then that configuration needs to be selected and then the GET request can be made. These steps in theory were simple, it required only 4 commands, however, it appears for cellular data to function, the Real Time Clock (RTC) on the modem needed to be set; this can be done with an additional command that will sync the time with a nearby cellular tower. However, from a restart, it took the A9G roughly 2 minutes, before the RTC could be set, this would mean the A9G should only be turned on relatively infrequently using the MOSFET to save power (such as once an hour). Setting up the APN was difficult as it required finding the APN name for the mobile carrier. Once this was found, when we performed the GET request, we were getting an unspecified error, after doing more research, we found that the APN Vodafone UK use requires login details and the command used in the code for APN configurations didn't have username and password as arguments. We eventually found a command used to configure the active APN and were able to include the login credentials. Another problem we ran into was that despite using the commands to select the APN configuration, the GET request was still returning an unknown error. We found a command that allowed me to get more information about the error codes and the state of the APN, which assisted in the debugging. From this we found that A9G was using the default APN configuration and not the intended one, once we removed the default setting, we were able to complete a GET request to the PHP file and receive an acknowledgement from the site.

```
OK
HTTP/1.1  200  OK
Date: Sun, 03 Jan 2011 01:59:19 GMT
C
```

Figure 33: Acknowledgement from PHP file

4.5 Testing [M.T.]

The device was taken to lake Bret in Switzerland, to take some measurements, and where the photo on this report's front page was taken. The lake is used as a drinking water reservoir for the city of Lausanne.

	A	B	C	D	E	F	G	H	I	J
1	Day	Month	Year	Hour	Minute	Second	Temperat	TDS	Turbidity	pH
2	16	12	2020	13	45	0	13.06	606.99	0	7.92
3	16	12	2020	13	46	0	6.25	739.31	0	7.58
4	16	12	2020	13	47	0	6.06	743.77	0	7.5
5	16	12	2020	13	48	0	5.88	758.19	0	7.43
6	16	12	2020	13	49	0	5.75	812.31	0	7.55
7	16	12	2020	13	50	0	5.63	761.04	0	7.43
8	16	12	2020	13	51	0	5.69	752.86	0	7.33

Figure 34: Data on SD card from lake Bret [M.T.]

Note that the water was quite cold (5.7 °C) in December. The initial temperature measurement is higher as it has not had time to settle to the water temperature yet. The TDS is quite high (700+).

Turbidity is 0, as the water is pretty clear. And pH is around 7.5, but we have doubts concerning the accuracy of the laboratory pH probe as it hadn't been stored or cleaned according to best practices, seeing as the wire mesh prevents easy access to the probe.

Data was collected and stored locally on the SD card, as well as transmitted via the Bluetooth module to a mobile phone, which acted as a serial monitor. This is presented in appendix N.

In addition, a series of tests were performed to demonstrate that all four sensors are working.

1. Submerged in tap water, it measured a pH of 7.48, and a portable pH meter measured 7.2. The pHOffset was set at -0.2 in the Arduino firmware.
2. We added Sulfamic acid (H_3NSO_3) to the water, bringing the pH down to close to 0. PONDER's pH sensor read -0.2 as a result of the phOffset, and the portable pH meter read 1.1 (perhaps a limitation of the meter). PONDER's pH meter seems to be responding to the change in pH, but its accuracy can't be reasonably determined without further tests with different buffer solutions of known pH.
3. Next, we tested the accuracy of the temperature sensor. PONDER read 13.6 °C, whereas a Fluke 62 Mini IR Thermometer read 12°C. Another temperature reading was taken with hot water, with PONDER reading 32.4°C, and the Fluke thermometer reading 33.8. In both cases, PONDER's thermometer seems to be reading high by about 1.5°C. This could easily be resolved by adding a -1.5 offset in the Arduino firmware.
4. With regards to TDS, in clear tap water in Guildford, PONDER read around 300 ppm, whereas in tap water in Switzerland, it read around 500ppm. The higher measurement in Switzerland is likely because the house's plumbing is relatively old. We did not have another measurement device with which to confirm PONDER's results.
5. To check that the turbidity sensor is working, we took a control measurement in clear water, which read 0 NTU, as expected. The Turbidity sensor measurement range is from 0-3000 NTU, with higher NTU meaning more suspended solids. We then diluted some ink cartridges in water. PONDER read 2200 NTU. Without another tool to measure Turbidity, we cannot tell how accurate this measurement is, but it is responding as it should.

Thus, we have demonstrated that all four sensors function as expected. The pH and temperature measurements were independently verified with separate instruments, but we could not do the same for TDS and Turbidity. We ordered a portable TDS meter from China, but it has not arrived in time due to Christmas postage delays.

4.6 PCB Design [V.B.L.]

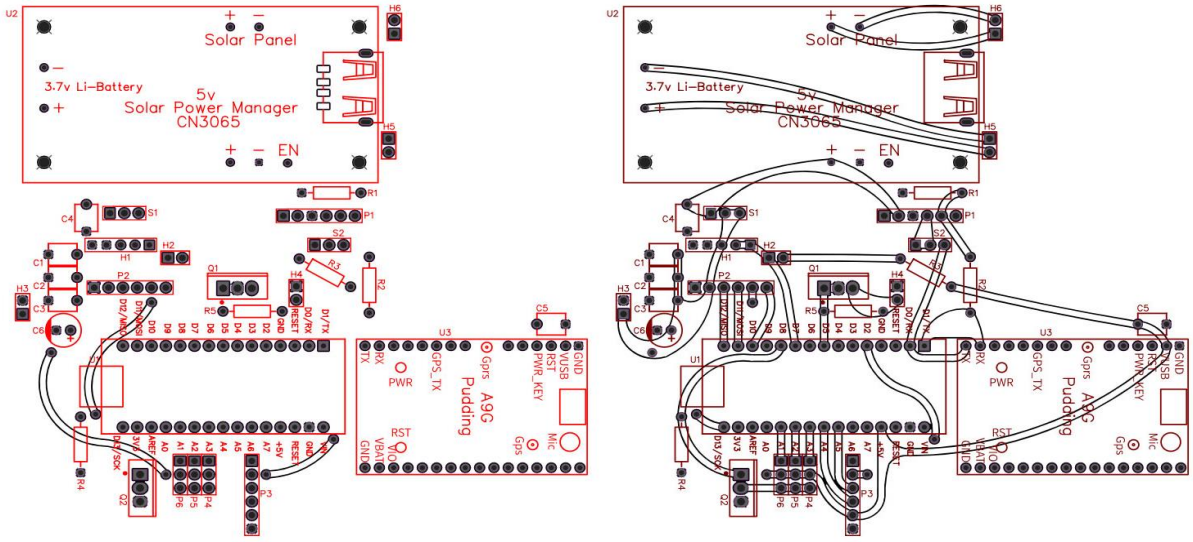


Figure 35: PCB Design [V.B.L.]

Towards the end of the project, time and budget concerns prevented us from implementing this design. Without enough leeway on the budget to order the PCB from a specialist and questions over whether the board would arrive to Mark in time if it were shipped had I used the FR4 PCB board and Ferric Chloride I have in my possession, we went with a Veroboard implementation in the end.

The proposed PCB design uses a 10cm x 12cm (horizontal x vertical) area. It would be possible to reduce this to 10cm x 7cm, were the Solar Power Manager daughter board to be replaced by a 2-pin header. Eliminating the placement of this daughterboard on the PCB would significantly reduce material usage, but for the purpose of a prototype PCB that might need to be moved around and attached/reattached to the casing during construction, testing and evaluation, I decided to add it to the PCB. Since the USB on the Solar Power Manager will remain unused, this left a suitably sized area on the top right of the PCB for the placement of the Li-Ion battery, which could be attached either via a cradle or using adhesive pads.

In the end, a double sided board that used only 6 vias was achieved. In addition, the tracks were all carefully drawn to be curved, with wide areas at junctions. This curvature and wide junction approach was implemented to reduce the risk of uneven heat distribution. One unfortunate aspect, upon reflection, are the relatively long lines exiting the A9G board, which is a high frequency digital communication board. This presents the obvious problem of possible resonance-related noise from these connections in the signals. This was however unavoidable due to both ends of the A9G daughterboard requiring connection to the Arduino Nano in the centre.

Care was taken to ensure that the used ports for the Arduino and the A9G board were accessible from their respective sides of the PCB edge. Additional attention was paid to ensure that the vertically placed daughter boards (to be inserted into the 1x6 female headers seen in the above figure) would have sufficient clearance to the sides, where the boards had significant profiles due to the components mounted upon their surfaces.

Additionally, each coupling capacitor or set thereof is as close to the relevant board(s) as possible.

4.7 Veroboard Design [M.T.]

A Veroboard design was made using the program DIYLC [53] on a 37 by 24 board, a quite common size of Veroboard. The design was implemented without any issues.

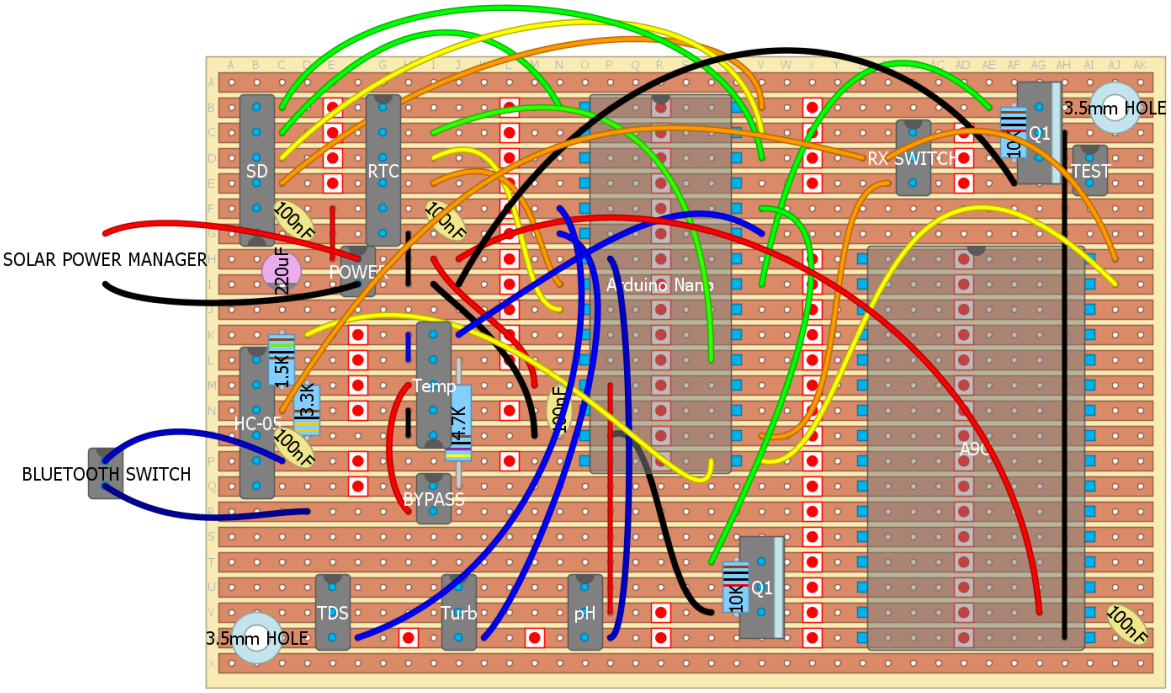


Figure 36: Veroboard design [M.T.]

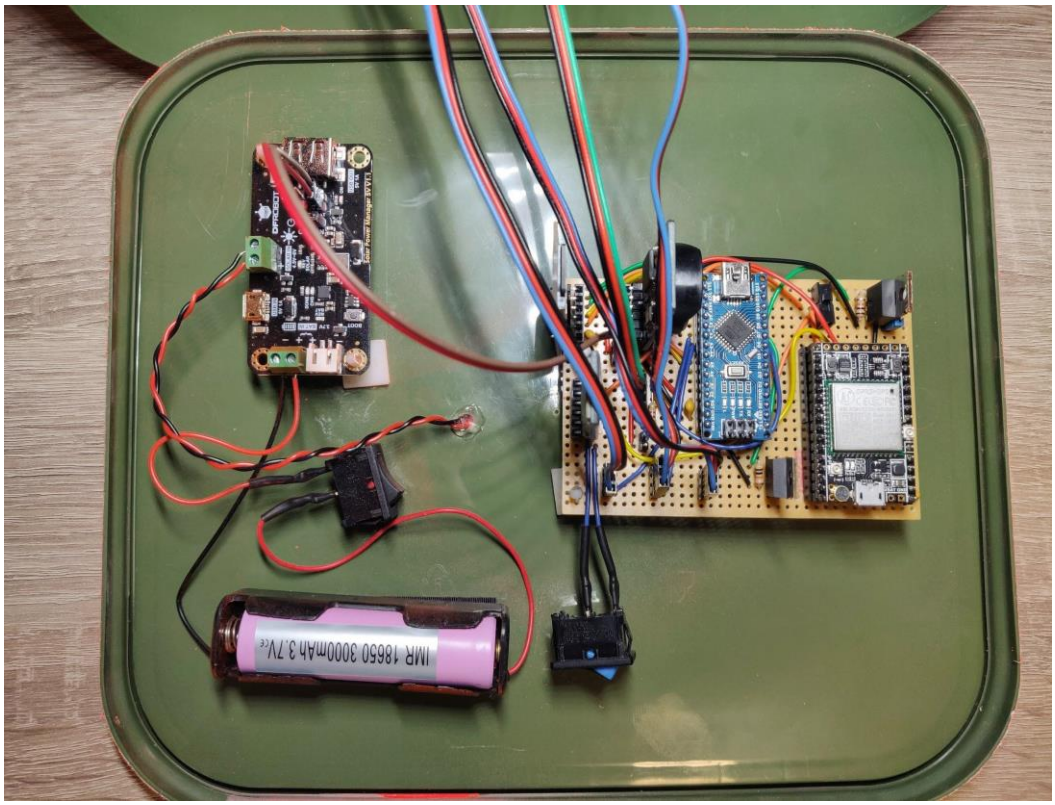


Figure 37: Veroboard implementation [M.T.]

4.8 Software Integration [L.W.] [P.Soni] [P.Sodani]

The integration between software and hardware plays an integral part to this project's success. To ensure that the software and hardware interact smoothly, we must ensure that the data communication between the device and DB (database) server is seamless and provides all the accurate information to be displayed onto the app. It must also verify that the information is being transmitted securely, whilst having back up protocols in case of any unexpected complications.

The Arduino regularly sends HTTP GET requests to the database, updating the database with the current sensor values and location, the rate at which the database is selected in code, and can be increased or reduced depending on the requirements set by the database. The data communication implementation is explored in further detail in section 5.4.

5

Software

5.1 Mobile Application [R.R.]

A companion app was designed to display sensor data in real time along with storing the data for later use. The app would allow for recording of sessions where the data collected by the device within a time frame would be grouped and saved, these sessions would be viewable at any time after recording within the app.

5.1.1 User Interface [R.R.]

Initial UI design plans included a live plotting graph for each pollutant and live alerts for the data being collected, later research found that reading the effects of these pollutants would be more practical and would tell us more info about the potential pollutants present in the liquid. However, the data being sent to the database would be infrequent and in larger batches so the decision was made to scrap the live graphing as the benefits of an infrequently updating graph didn't out-weigh the performance decrease on the app. The same idea was used for the live alerts system planned, as the data received was infrequent batches, all the alerts received would be from hours in the past.

The overall design goals were to make the interface clean and with plenty of relevant data presented at a glance. On the next pages are screenshots of the app in use, with explanations for each. When the app is first opened, this is the screen that is presented to the user.

The app features both a dark and light mode that is tied to the internal system preference. When the user taps the "POWER ON" button a new session begins, and the app begins updating the current session data whenever new data is present in the database. The latest values are shown on the left of each section on the main tab with the minimum and maximum values shown in smaller fonts to the right, these are also updated to be the latest minimum and maximums.
Session duration is shown under the "VIEW ON MAP" button, this displays the time since the "POWER ON" button was turned on and is used as a reference for the time when data was received by the app.

When a session is started, the app will constantly check the database for updates, recording the data as it comes in. The "POWER ON" button switches to a "SHUT DOWN" button which ends the session, and while the session is running the "VIEW ON MAP" button will function to open the default map application installed on the device with parameters that will point to the latest longitude/latitude coordinates.

When pressing "SHUT DOWN" the session ends (regardless of the device current status), averages are calculated for the whole session and all the data is dumped into a text file on the phone which can be viewed in the data history tab.

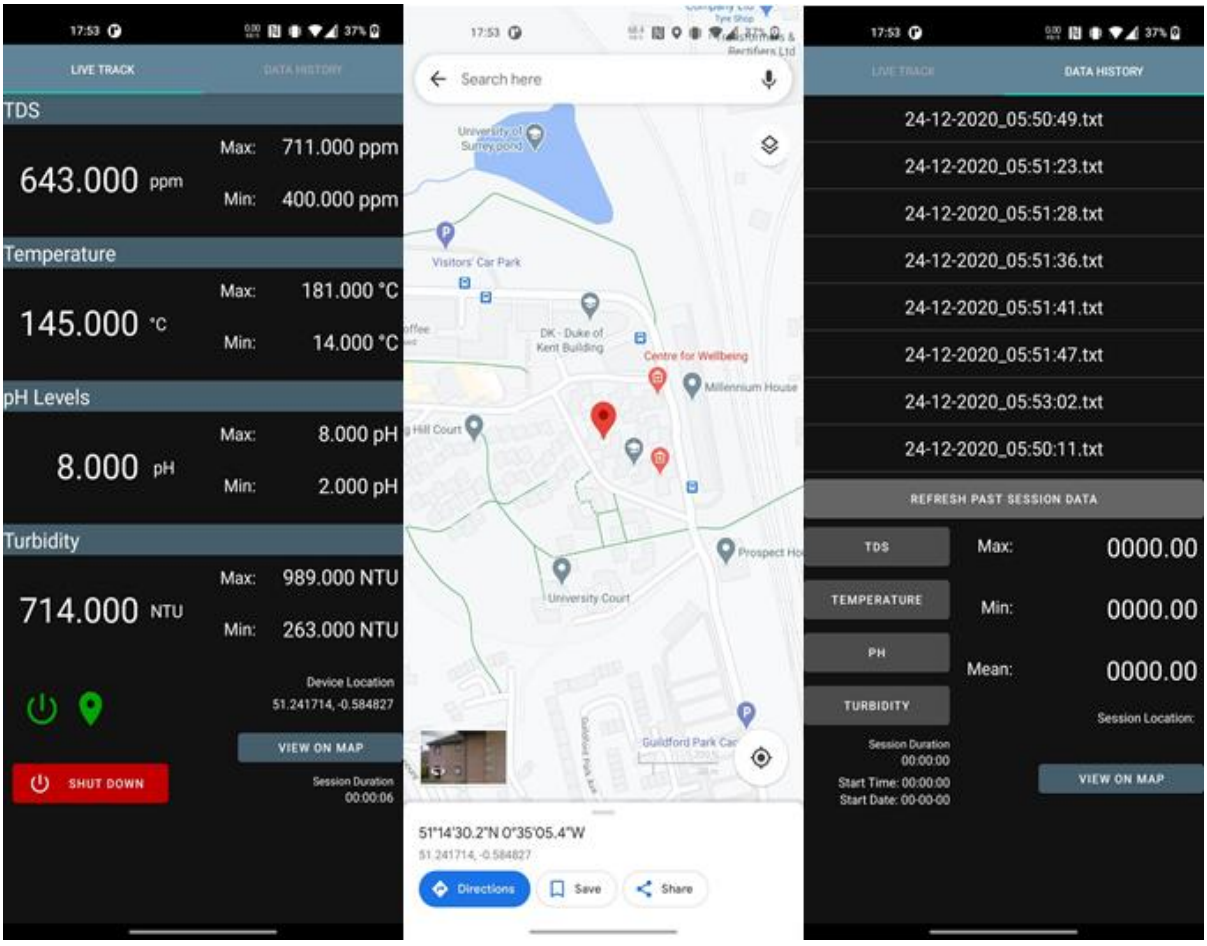


Figure 38: Main Tab Running With Example Data (Left), Google Maps View Of Example Device Location (Middle), Data History Tab With Multiple Session Entries (Right) [R.R.]

The second tab displays data from previous sessions, with each session being a text file with names in the format of "DATE_TIME" (DD-MM-YYYY_HH:MM:SS).

Session data is saved to:
Storage/Internal_Storage/Android/data/com.example.ponderlivetrack/files/documents
all the sessions shown here can be deleted and modified manually by traversing through to the directory, pressing the "REFRESH PAST SESSION DATA" button will clear the list of session data and re-read the directory.

For each session in the list shown, all the data can be seen from within the app with the same "VIEW ON MAP" button that allows the user to see the final location for the session on the default map application.

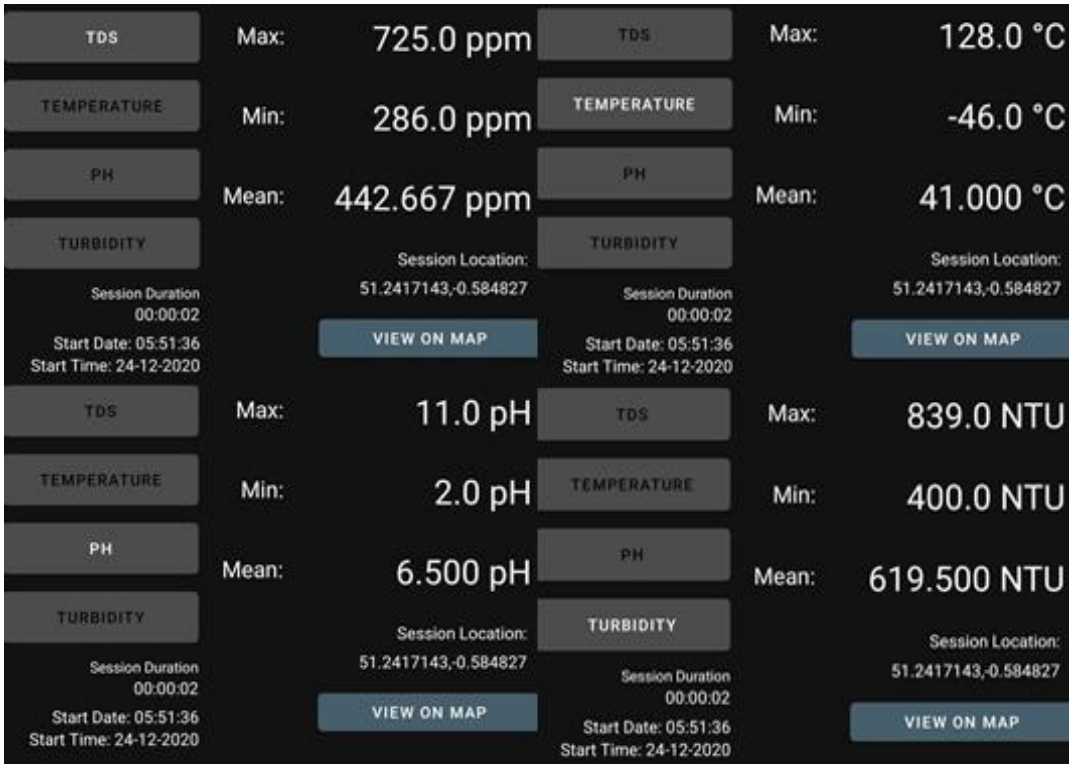


Figure 39: Data History Tab Showing Data For An Example Session [R.R.]

5.1.2 Back-end Programming [R.R.]

The whole app is split into 3 parts: LiveTrackFragment, DataHistoryFragment and the MainActivity. MainActivity sets up the tabs for the Live Tracking and the Data History. LiveTrackingFragment uses a handler loop to constantly check the database and update data with any new data. The handler loop allows the app to loop in the background while not in focus, meaning that the user may use their device for other things while a session is being recorded. Also handles writing of the session file data for the other fragment and visual updating of data. DataHistoryFragment reads all files within a specific directory on the device and all the data on those files to be presented to the user on demand.

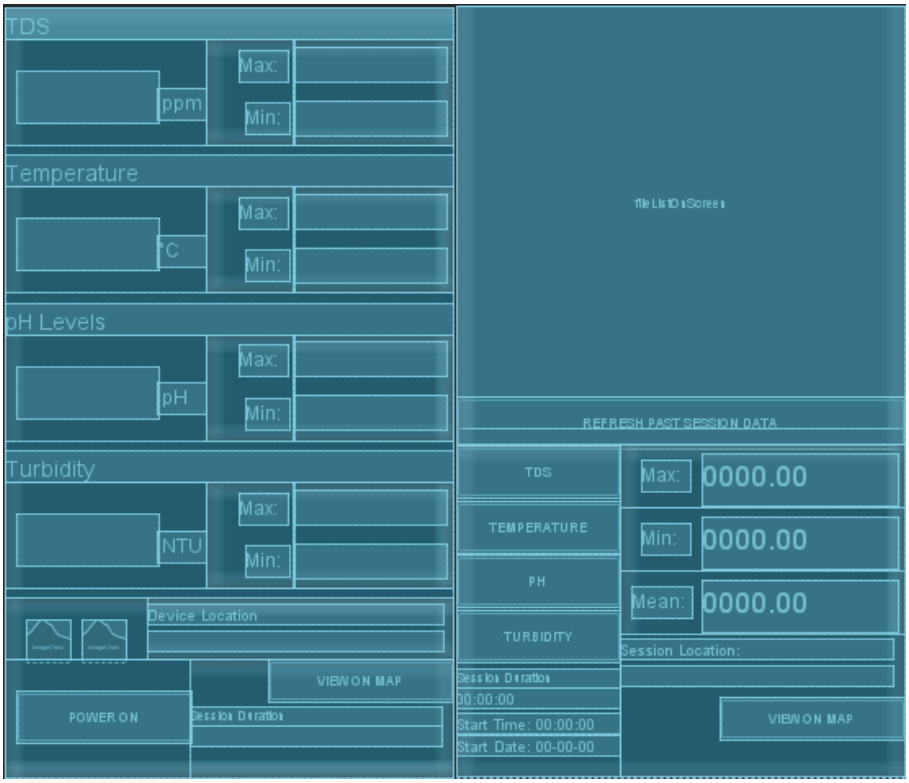


Figure 40: Application Layout Blueprint [R.R.]

The app layout was designed for phones with a 21:9 aspect ratio, however all elements are dynamically resized to fit most other aspect ratios but may not work on extreme ratios. All the elements are set up to be in a hierarchy with boundaries defined by their parents such that elements will never be in the wrong place, but on much smaller resolutions, text sizing issues might arise.

One of the major issues encountered during development was that during a session recording, any time the user would switch fragments (switch tabs or minimise the application) and then reopen the app, the handler loop would start up again without first ending the loop prior to it, therefore creating an exponentially increasing memory demand for the app with each opening and closing.

Here the "onResume" function (this function is called when the app/fragment is opened) has been overridden to first change a global boolean token to true for use in some optimisations and the "happenOnce" function allows the loop to properly close before restarting itself, solving the memory issue. The "onResume" function also allowed for a small optimisation to skip any visual updates from occurring when the app isn't in focus, as these would happen all at once when the app would be resumed without this, causing the app to freeze temporarily when resuming it while it catches up on all the visual updates.

```
override fun onResume() {
    super.onResume()
    /*re-entering fragment from tabbing back into app or swiping back onto frag*/
    isLiveTrackOpen = true
    happenOnce()
}
```

Figure 41: Basic Handler Loop Optimisation [R.R.]

Data stored from previous sessions are stored in a format similar to the image below, making it easier for the app to later read while also retaining readability from the user.

```
Data Collected From Session Starting: 24-12-2020_05:53:02
Session Duration: 00:00:11
TDS:
MAX = 953.0 ppm
MIN = 109.0 ppm
MEAN = 567.364 ppm
TEMPERATURE:
MAX = 181.0 °C
MIN = -139.0 °C
MEAN = 61.727 °C
PH LEVELS:
MAX = 12.0 pH
MIN = 0.0 pH
MEAN = 5.273 pH
TURBIDITY:
MAX = 989.0 NTU
MIN = 263.0 NTU
MEAN = 555.727 NTU
GPS DATA:
LONG = 51.2417143
LAT = -0.584827

----DATA FROM FULL SESSION---- FORMAT -> (TDS,TEMPERATURE,PH,TURBIDITY)

< 402.0 , 49.0 , 6.0 , 467.0 > @ 00:00:01
< 651.0 , 58.0 , 5.0 , 487.0 > @ 00:00:02
< 711.0 , 149.0 , 2.0 , 263.0 > @ 00:00:03
< 547.0 , 14.0 , 2.0 , 567.0 > @ 00:00:04
```

Figure 42: Application Data Storage Example [R.R.]

5.2 Software Block Diagram [P.S$_{odani}$] [P.S$_{oni}$]

The software is split into three parts: Storage, communication and user interface. The storage is done via DB, the communication is handled through SQL libraries which assist in inserting and analysing data in the data tables e.g. temperature sensor values, and a user interface to view the data is handled through an android app, where the information is displayed clearly and the user can interact with the devices.

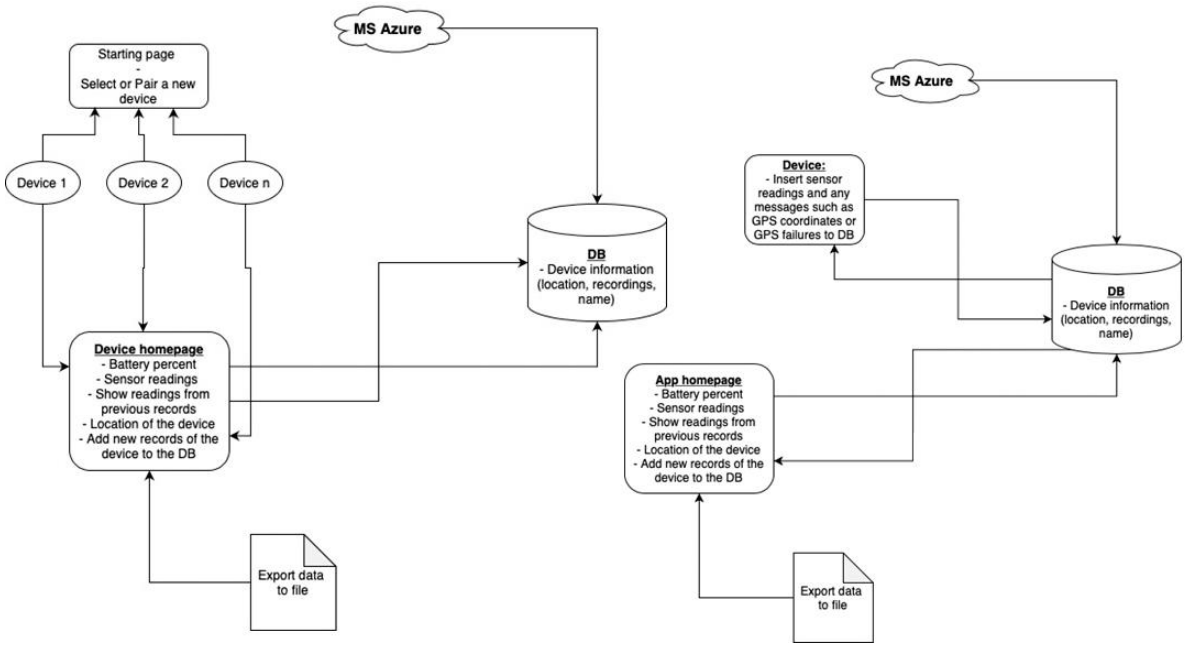


Figure 43: Software Block diagram showing overview of software setup (left), Software Block diagram showing backup software setup (right) [P.S$_{odani}$] [P.S$_{oni}$]

In the diagram above, we can see the architecture of our software setup. The DB is the core element in the architecture, however, we have designed this architecture with the utmost considerations

for unfortunate circumstances, such as the data failing to be stored in the Azure DB. For this we have a local backup storage plan in the form of an SD card, to which the device also stores the data, in the event an upload of the data is interrupted/ incomplete.

To simplify the diagram, the device has a connection to the DB and sequentially uploading data on the respective data tables. The app continuously checks for updates in the DB, and updates the user interface once there is an update to the DB. The user has the ability to export sessions, where they can keep a record of the particular range of dates or time for the sensor data they want.

As a backup to the GSM/GPRS architecture, we also designed an architecture that uses a bluetooth module to transfer the data to the mobile device it connects to. This then gets displayed on the app, which then uses the SQL libraries to store the data in the database.

5.3 Database Server Setup [P.Sodani] [P.Soni]

We hosted the database (DB) server in the cloud, specifically Microsoft Azure cloud services. The choice of this DB server setup is ideal for this project because it is free to use for students. This is due to their credit system, where each student gets 100 credits to use their Azure service for free with 50GB of data. This configuration was perfect for us as we have 8 students in this project, thus theoretically we could run this server for 8-12 months free of charge and have access to a sufficient amount of storage for the incoming data from the device's sensors.

An Azure MySQL server was created with 50GB with two cores and 2GB RAM. The aforementioned configuration fulfils the needs for this project. The following database schema structure was applied:

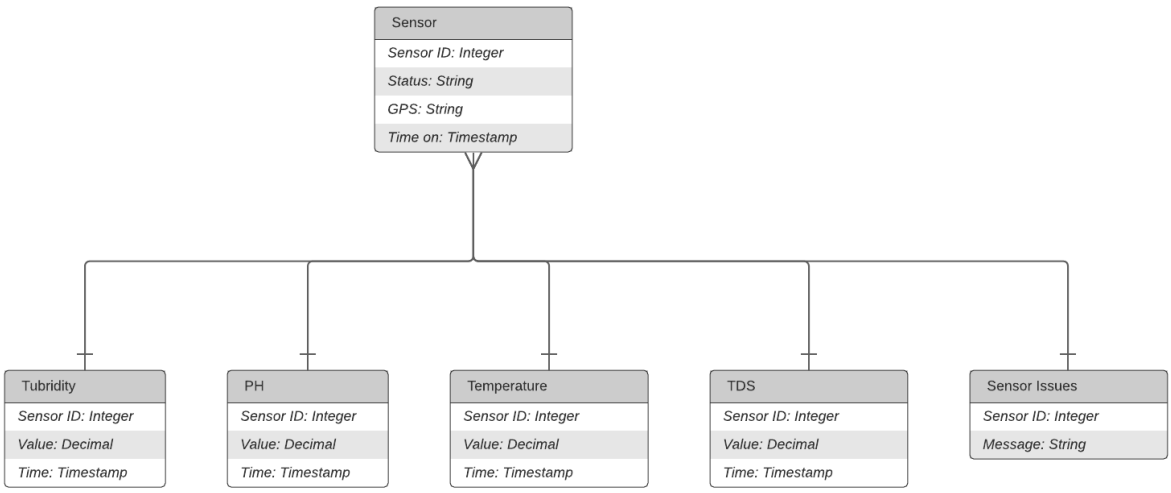


Figure 44: DB UML diagram [P.Sodani] [P.Soni]

Each table was carefully curated to meet the simplistic needs of a database and to meet the one-to-many, many-to-one and many-to-many database relationships. For e.g. one sensor that measures pH values can have multiple pH values linked to the sensor ID.

5.4 Data Communication Implementation [P.Sodani] [P.Soni]

We used the mobile data approach as highlighted in section 5.2. The A9G arduino module has the capability of using GSM/GPRS which can be used to establish a secure connection to the database server hosted on Microsoft Azure, which we will talk more about in section 5.3. The GSM libraries that the module provides is limited, as it can only make GET and POST requests to URLs. Therefore, we hosted a simple web server where the arduino module can make GET requests on demand to a PHP file hosted on the web server. The PHP file contains code which utilises the SQL libraries to initiate a secure connection to the DB and executes SQL queries to insert data into its respective predefined formatted table e.g. a pH value recorded from the device's sensor would be inserted into the pH table in the DB schema. In order to handle different types of data (sensor values, error messages etc.), we separated the SQL queries for the different types of data into separate PHP files. This is an efficient approach as it helps us navigate any issues we may have with the data and it is easy to understand for someone looking into the project with no prerequisite knowledge of how this setup works. The two main security considerations we had for the communication layer are: Getting a secure connection to the DB (SSL certificates) and a secure GET request to the PHP server. The web server was hosted using the web provider called '000webhosts', which by default provides HTTPS hosted websites, which covers the second security consideration for the GET request. For the first security consideration, Microsoft provides Azure certified certificates, which can be used whilst connecting to the DB server (53).

```php
<?php
class dht11{
    public $link='';
    function __construct(){
        $connection = $this->connect();
        $this->storeInDB($_GET['sensor_value'], $_GET['sensor_status'], $connection);
    }

    function connect(){
        $con=mysqli_init();
        mysqli_ssl_set($con, NULL, NULL, "../certs/BaltimoreCyberTrustRoot.crt.pem", NULL, NULL);
        mysqli_real_connect($con, 'ponderdb.mysql.database.azure.com', 'db_admin@ponderdb', '*********', 'ponder', 3306);
        if(mysqli_connect_errno($con)){
            die('Failed to connect to MySQL: '.mysqli_connect_error());
        }
        else{
            printf("MySQL successfully");
        }
        return $con;

    }


    function storeInDB($sensor_value, $sensor_status, $conn){
        date_default_timezone_set("Europe/London");
        $currentTime = date("Y-m-d h:i:s");
        $sql = "INSERT INTO sensor (id, status, timeon) VALUES ('$sensor_value', '$sensor_status', '$currentTime')";
        if(mysqli_query($conn, $sql)){
            echo "Sensor added successfully.\r\n";
        } else{
            echo "ERROR: Could not able to execute $sql for sensor .\r\n" . mysqli_error($conn);
        }
    }
}
$dht11=new dht11()
```

Figure 45: Example PHP code for adding a sensor into the DB [P.Sodani][P.Soni]

On the Arduino, in order to communicate and update the database with the sensor values, it requires correctly configured cellular data access. The A9G cellular module must have the correct Access Point Name (APN) configuration in order to access the cellular functions, as the SIM card being used is a Vodafone Pay Monthly one, the APN name is "wap.vodafone.co.uk" and both the username and password is "wap". Once the configuration is complete and the module can access the internet, a GET request can be made by using the URL of the PHP file, and by

adding the values at the end of the URL, it becomes able to update the database remotely and securely. The full code can be found here: https://github.com/sporadicE/PONDER/tree/main/Database%20Code.

```
SoftwareSerial mySerial = SoftwareSerial(3, 4); //uses digital pins on device to act as additional UART lines
SoftwareSerial gps = SoftwareSerial(8, 9);
String data = "";
void updateSerial();
void updategps();
void setup()
{

  Serial.begin(115200);
  mySerial.begin(115200);
  delay(10000);
  mySerial.println("AT"); //Once the handshake test is successful, it will back to OK
  updateSerial();
  mySerial.println("AT+CMEE=2"); //Gives details for potential error codes
  updateSerial();
  mySerial.println("AT+CGREG?"); //Read SIM information to confirm SIM registration details
  updateSerial();
  mySerial.println("AT+CCLK?"); //Updates RTC with Time from cell tower, NOTE: Upon reset of module, time is reset, needs about
  updateSerial();
  mySerial.println("AT+CGATT=1"); //enables data
  updateSerial();
  //mySerial.println("AT+CGDCONT=1,\"IP\",\"wap.vodafone.co.uk\""); //APN settings in preset slot 1
  updateSerial();
  mySerial.println("AT+CGACT=1,1"); //Enables APN setting in slot 1, first argument is to enable, second is the slot
  updateSerial();
  //mySerial.println("AT+CSTT= \"wap.vodafone.co.uk\",\"wap\",\"wap\""); //configures the APN settings used
  updateSerial();
  mySerial.println("AT+CSTT?"); //displays the APN settings being used
  updateSerial();
  mySerial.println("AT+HTTPGET=\"http://surrey-ponder.000webhostapp.com/insert_measurements.php?sensor_value=123&phValue=7.31&te
  updateSerial();
}
```

Figure 46: Example code for initialising cellular data and sending a GET Request
[P.Sodani] [P.Soni] [L.W.]

Conclusion [M.T.]

PONDER's data logging abilities could prove to be an effective way to improve our knowledge of water pollution sources. This would prove valuable in developing countries, where wastewater discharge is an especially severe problem, preventing access to safe drinking water to over 2.1 billion people, and putting the environment at risk. The prohibitive cost of current commercial water monitoring solutions means that they are only in use in a few key regions.

By contrast, our solution is extremely inexpensive, presenting a low barrier to entry and making it easily scalable to a large network of devices. Our device can also be easily modified to suit the needs of the client by adding more sensors relevant to their intended use case, such as for dissolved oxygen and electrical conductivity. Our devices could help pin-point sources of pollution and help inform our decision making.

PONDER has been designed with sustainability in mind, as its small form factor has almost no appreciable impact on its environment. Its solar panel and battery allow it to be completely self sufficient and autonomous, making it suitable for long term use without any maintenance. Our choice of plastic and marine grade stainless steel makes it highly resistant to the environment.

We think that our development process went very smoothly and that there is little that we would change. We would have liked to be able to be able to use sensors to independently verify PONDER's TDS and Turbidity measurements, and we would also have liked to be able to perform more real-world tests, if it were not for COVID-19.

Appendix A: Specification Verification Matrix [K.H.]

#	Specification	Compliance
1.1	The device should have an aspect of hardware associated with data generation or detection, and recording.	The implementation of sensors into the device which will allow for the monitoring of different aspects of the local water environment.
1.2	The device should have an aspect of timed data transmission to a known free secure location.	Use of a Real Time Clock to timestamp all data. Data transmission via bluetooth or GSM/GPRS to a free web server.
1.3	The device should have an aspect of data display and analysis on a portable device.	The development of a complimentary app will allow all data from the sensor device to be displayed in a user friendly format.
1.4	The design and development of this device should be fully documented with evidence of testing and validation.	The consistent documentation of meeting minutes, research and design documents will allow for proper documentation to be created.
1.5	The device should have a maximum budget of £100.	The price of any components and other costs will be tracked carefully to ensure that the total cost does not go over budget.
2.1	The device should be able to record data related to different aspects of water quality (i.e. pH, TDS, turbidity, temperature, etc.)	The implementation of multiple water quality sensors, such as pH, TDS, turbidity, and temperature, will allow for the device to monitor various different aspects of water quality.
2.2	The device should be able to store recorded data locally as well as transmit data to an external device.	An SD card is used to store recorded data locally. Use of a Bluetooth or GSM/GPRS module to transmit to an external device.
2.3	The device should be able to display recorded data in a presentable and easily understandable way.	The development of an accompanying app will allow data transmitted from the device to be formatted and displayed.
3.1	The device should demonstrate the ability to operate remotely for extended periods of time in a self-sufficient manner.	The use of a solar cell and a Li-Ion battery, designed to suffice when solar irradiation is at its lowest in December. Implementation of other measures to reduce microcontroller idle current.
4.1	The device should be able to remain buoyant to allow a stable test platform.	The enclosure must be designed in such a way as to be buoyant, stable, and with a means to tether the device to a fixed location.

4.2	The device must be waterproofed in order to protect the internal electronics in the device.	Measures must be put in place to prevent the ingress of water into the device, such as to avoid water coming into contact with any electronics.
4.3	The device should be constructed from sturdy materials in order to ensure safe operation in the environment.	Careful consideration must be taken when choosing the materials to make the device from, so as to allow safe operation in the environment.

All the above specifications are met by the final prototype.

Appendix B: Sulphur Dioxide Research [R.R.]

Effects of Sulphur Dioxide Pollution:

- Deforestation around the body of water, and usually within the local area.
- Significantly lowers the PH levels of any bodies of water in the area. This causes water to become highly corrosive towards paints and common building materials.
- Sulphuric waters cause most aquatic life to die
- The gas can irritate eyes, and damage the respiratory tract which leads to more infections
- Rain becomes acidic causing widespread damage to plant life
- Even lightly sulphuric waters can cause a laxative effect which leads to dehydration.

Usually, the presence of sulphuric waters means there's sulphur dioxide air pollution.
The air pollution is mainly caused by fossil fuel burning, and ore extraction contributes to a minor increase in SO2.

Easy Signs of Sulphuric Waters:

- Deforestation
- Light Hazing effect
- Scaly-like build up around pipe and edges of the body of water
- Blackened water
- Rotten egg water smell

Testing for Sulphuric Waters:

A sample of water is taken and mixed with a chemical preservative

Clearing of Sulphuric Waters:

- Filtration via carbon filters
- Manganese oxide coated greensand filter

Appendix C: Nitrates & Phosphates Research [J.E.]

What are fertilisers?

Fertilisers are useful chemicals that are given to plants to enhance the crop yield. Plants are living organisms which source nutrients from fertilisers in order to grow. Given that this is already present within the soil, these amounts are normally inadequate so they rely on an external source to fulfil their maximum growth. The primary nutrients required for plant growth are Nitrogen, Potassium and Phosphorus. [54]

Fertilisers can come in various forms and are considered either organic or inorganic. Organic forms of fertilisers typically come from animal manure, compost, crop residue and biological N fixation. Inorganic fertilisers come in the forms of minerals (nitrogen and phosphorus fertilisers). These are artificially made but produce high volumes of these nutrients for plants. [55]

Effects of Fertiliser in water

Pollution of fertilisers within lakes and streams occur through soil erosion and runoff via rainwater. These eroded soil particles contain nitrates and phosphates which promote microorganism growth like weeds and algae blooms, whilst withdrawing dissolved oxygen from the water. This has a severe impact on the biological equilibrium as fish resultantly begin to suffocate and die, reducing the water quality. Nitrates is high quantities can be a problem for drinking water as they are known to be the cause of methemoglobinemia (blue-baby syndrome) in young children and warm-blooded animals, which interrupts the oxygen intake in the circulatory system. [56]

Algae (cyanobacteria) when in contact with humans can be a health risk with symptoms such as “rashes, nausea and respiratory problems”. Within livestock, cyanobacteria poisoning can be fatal as soon as 20 minutes of indigestion. Typical symptoms experienced are “weakness, staggering, difficulty in breathing, convulsions and, ultimately, death”. [57]

Fertilisers effect on water pH

PH levels measure how acidic or basic water is. According to water quality regulations in the UK, suitable pH levels should range in between 6.5 - 9.5 pH. [58] As algae is produced through photosynthesis, it extracts dissolved carbon dioxide and produced atmospheric oxygen. This reduces the acidity of the water, which in effect raises the water’s pH level. However, the decomposition of dead plants and other material by bacteria depletes oxygen due to bacteria requiring oxygen to respire. This in turn, makes the water more carbonic and decreases the pH level. [59] A suitable pH range for water must be maintained because a pH below 5 can harm fish to the point where calcium levels in female fish would be too low to reproduce, or high pH levels past 9.6 may damage the gills and eyes of fish, eventually causing death. [60]

Appendix D: Oxygen Depletion Research [P.S$_{oni}$]

Microorganisms that live in water feed on biodegradable substances. When too much biodegradable material is added to water, the number of microorganisms increase and use up
the available oxygen. This is called oxygen depletion.

Effects of Oxygen Depleting Water Pollution

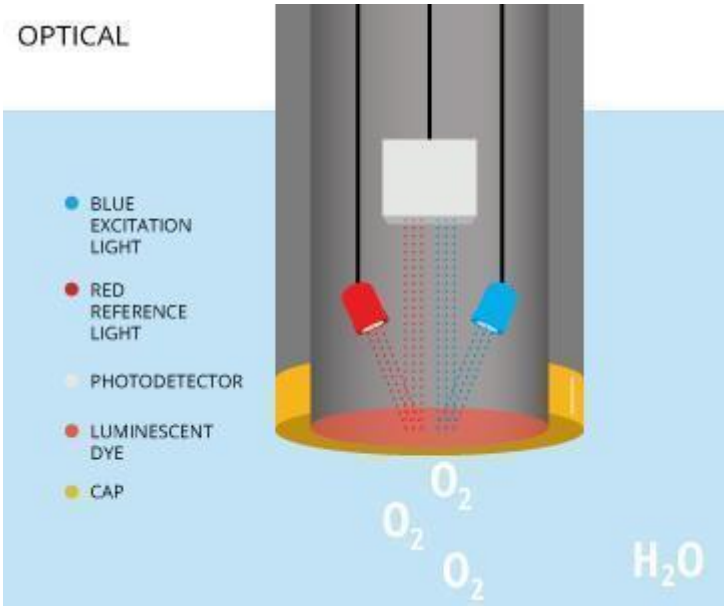
When oxygen levels in the water are depleted, relatively harmless aerobic microorganisms die, and anaerobic microorganisms begin to thrive. Some anaerobic microorganisms are harmful to people, animals and the environment, as they produce harmful toxins such as ammonia and sulphides.

Measure the oxygen levels under water:

Optical dissolved oxygen sensors measure the interaction between oxygen and certain luminescent dyes. When exposed to blue light, these dyes become excited (electrons gaining energy) and emit light as the electrons return to their normal energy state [12]. When dissolved oxygen is present, the returned wavelengths are limited or altered due to oxygen molecules interacting with the dye. The measured effect is inversely proportional to the partial pressure of oxygen [5]. While some of these optical DO sensors are called fluorescent sensors [10], this terminology is technically incorrect. These sensors emit blue light, not ultraviolet light, and are properly known as optical or luminescent DO sensors [11]. Optical dissolved oxygen sensors can measure either the intensity or the lifetime of the luminescence, as oxygen affects both [23].

An optical DO sensor consists of a semi-permeable membrane, sensing element, light-emitting diode (LED) and photodetector [3]. The sensing element contains a luminescent dye that is immobilized in sol-gel, xerogel or other matrix [23]. The dye reacts when exposed to the blue light emitted by the LED [3]. Some sensors will also emit a red light as a reference to ensure accuracy [5]. This red light will not cause luminescence, but simply be reflected back by the dye [7]. The intensity and luminescence lifetime of the dye when exposed to blue light is dependent on the amount of dissolved oxygen in the water sample [23]. As oxygen crosses the membrane, it interacts with the dye, limiting the intensity and lifetime of the luminescence [3]. The intensity or lifetime of the returned luminescence is measured by a photodetector, and can be used to calculate the dissolved oxygen concentration.

The concentration of dissolved oxygen (as measured by its partial pressure) is inversely proportional to luminescence lifetime as shown by the Stern-Volmer equation[5]:



This sensor can be used with a network of other sensors and pass the information from all sensors to the app. We can also try and predict the percent of harmful microorganisms from the oxygen levels in the water.

Appendix E: Organic Solvents Research [M.T.]

What are Organic Solvents?

In chemistry, organic compounds are generally any chemical compounds that contain carbon-hydrogen bonds.[61] Hydrocarbons are organic compounds consisting entirely of hydrogen and carbon.[62]

Many human-made toxic substances are organics compounds. They include household and agricultural pesticides, many solvents, and other household and industrial chemicals. Polychlorinated biphenyls (PCBs), for example, are industrial chemicals that are toxic and carcinogenic. Although banned in the United States in 1977, PCBs persist in the environment, and they accumulate in fish and human tissues when consumed. Some organic compounds are highly bioaccumulative and may be measured in sediment or tissue as well as water.[63]

Not all organic compounds dissolve in water. Requirements to dissolve in water are: hydrogen bonds (F-H , O-H , N-H , Cl-H) between the organic compound and water molecules, a relatively small molecular mass, and small size of alkyl group. In hydrocarbons (alkane , alkene and alkyne), there are only carbon - hydrogen bonds in the organic compound. So polarization is weak to make strong intermolecular forces between water molecules, so they are not soluble in water.[64]

Organic compound type	**Solubility in water**
Alkanes	Insoluble
Alkenes	Insoluble
Alkynes	Insoluble
Benzene	Insoluble
Alkyl halides	Insoluble
Alcohols	Alcohols **upto four carbon atoms** (methanol, ethanol, propanol, butanol) are much soluble in water
Phenol	Insoluble
Aldehyde	Aldehydes with less molecular mass are soluble
Ketone	Aldehydes with less molecular mass are soluble

Carboxylic acid	Carboxylic acids with small alkyl groups are soluble. (formic, ethanoic/acetic, propanoic acids are soluble) All aromatic carboxylic acids are insoluble.
Carboxylic acid chlorides	**Reacts with water** and form soluble HCl and carboxylic acid.
Amides	Amides with less molecular masses (small alkyl groups) are soluble.
Amines	Amines with less molecular masses are soluble.

One way to group chemical contaminants is as Inorganic Contaminants (IOCs) (including nitrate and arsenic), Volatile Organic Contaminants (VOCs), or Synthetic Organic Contaminants (SOCs) [65], the latter of which are man-made compounds used for a variety of industrial and agricultural purposes. This group of contaminants includes pesticides, PCBs, and dioxin.[66]

Effects due to Ingestion

"Organic solvents can be carcinogens, reproductive hazards, and neurotoxins. Carcinogenic organic solvents include benzene, carbon tetrachloride, and trichloroethylene. Organic solvents recognized as reproductive hazards include 2-ethoxyethanol, 2-methoxyethanol, and methyl chloride. Organic solvents recognized as neurotoxins include n-hexane, tetrachloroethylene, and toluene. Many classes of chemicals are used as organic solvents, including aliphatic hydrocarbons, aromatic hydrocarbons, amines, esters, ethers, ketones, and nitrated or chlorinated hydrocarbons. Organic solvents are used in many industries. They are used in paints, varnishes, lacquers, adhesives, glues, and in degreasing and cleaning agents, and in the production of dyes, polymers, plastics, textiles, printing inks, agricultural products, and pharmaceuticals. Millions of U.S. workers are exposed to organic solvents. The level of exposure depends upon the dose, duration, and work being done."[67]

Even at trace levels, chronic ingestion of these compounds causes many different types of diseases including cancer, birth defects, obesity, autism, cardiovascular disease, damaged genes, enzyme dysfunction, damaged mitochondria, damaged DNA, and hormone disruption.[68] They accumulate and persist in organs and fat tissue, and can reach concentrations up to 70,000 times background level.

Pharmaceuticals and personal care products (PPCPs) are emerging pollutants of concern. PPCPs can affect reproductive and developmental processes in fish and wildlife and the insect life they depend on for food. They include a wide variety of synthetic chemicals, such as antibacterial compounds in soaps, hormones, fragrances, cleaning agents, and both over-the-counter and prescription medications. PPCPs likely enter surface waters primarily through end-use rather than manufacturing, either by excretion or disposal by flushing. Low concentrations of PPCPs have been found in the Boise River, which was part of a 2002 US Geological Survey study on pharmaceuticals, hormones, and other organic wastewater contaminants in surface water. The most common PPCPs found in the Boise River were steroids, nonprescription drugs, and insect repellant."[69]

Effects due to decomposition

During the decomposition process of organic compounds, the dissolved oxygen in the receiving water may be used up at a greater rate than it can be replenished, causing oxygen depletion and having severe consequences for the stream biota.[70] This forms part of the mechanism of Eutrophication; when algae die, they decompose and the nutrients contained in that organic matter are converted into inorganic form by microorganisms.[71] Organic effluents also frequently contain large quantities of suspended solids which increase turbidity of the water, and reduces the light available to photosynthetic organisms.[72]

"The presence of a sufficient concentration of dissolved oxygen is critical to maintaining the aquatic life and aesthetic quality of streams and lakes. Determining how organic matter affects the concentration of dissolved oxygen (DO) in a stream or lake is integral to water- quality management. The decay of organic matter in water is measured as biochemical or chemical oxygen demand. Oxygen demand is a measure of the amount of oxidizable substances in a water sample that can lower DO concentrations."[73]

Measuring Levels

Measuring the levels of these organic contaminants requires a Total Organic Compounds (TOC) lab analysis. Most lab analyses measure down to the Parts Per Million (PPM) level, but given that most organic compounds only exist in trace amounts, these tests are often insufficient. Instead, a Qualitative Analysis for TOC measured down to the Parts Per TRILLION level is required; such tests cost thousands of dollars and are only performed by a few laboratories.[74]

Therefore, there currently does not exist a cost-effective and practical means to DIRECTLY measure the amounts of these trace organic contaminants that is suitable for our project.

However, the effect of organic matter decay can be measured by Biological Oxygen Demand (BOD). This measures the amount of oxygen consumed by bacteria and other microorganisms while they decompose organic matter under aerobic (oxygen is present) conditions.[75] Dissolved oxygen sensors, such as this one: https://www.dfrobot.com/product-1628.html are more affordable, but still outside of our £100 budget.

Appendix F: Heavy Metals Research [P.Sodani]

What are Heavy metals?

Heavy metals are a group of metals that have relatively high density and are toxic to living organisms. Examples include Lead (Pb), Arsenic (As), Mercury (Hg) etc. These metals are released into the environment by both natural and anthropogenic sources such as industrial waste, vehicle exhaust, and mining. Most of them are known to be potential carcinogens, which are substances that promote the formation of cancer. Studies have shown that these heavy metals can cause serious long term health problems. Unlike organic pollutants, heavy metals are non biodegradable and can accumulate in living beings. For this reason, suitable methods need to be designed so that they can be removed from the environment as safely and efficiently as possible . [76]

Effects of Heavy metals

Metals can be introduced into an environment as the results of weathering of soils and rocks, from volcanic eruptions and from human activities such as mining, vehicle combustion or the use of metals that contain metal pollutants.

The most common metal pollution in freshwater comes from mining companies. They have an industrial process that uses an acid drainage system that releases heavy metals from ores, as heavy metals are soluble to acid. The acid gets dispersed into the groundwater, which has a high concentration of these metal pollutants.

Metal pollutants are toxic in freshwater due to the acid reducing the pH levels in the water. This allows the metal to become more soluble and mobile. These pollutants lie at the bottom of river sediments for a long period of time. Since they are non-biodegradable, they can cause significant environmental damage. This is dangerous because it can bioaccumulate in living organisms, as there is no easy or natural way to break these heavy metals down into smaller, less harmful components.

Heavy metal poisoning can result from drinking contaminated water, intake via the food chain or high amounts of particle concentration in the air.

From the research above, it is easy to conclude that heavy metals can have a significant effect on the environment, which leads to having an effect on our health and other living organisms. [77]

Measuring Heavy metal Levels

Heavy metals can be measured by Inductively Coupled Plasma (ICP) by Mass spectrometry (MS). The ICP is used to ionise the sample, and the MS separates and quantifies these samples. [78]

Sulphur Dioxide Water Pollution

Effects of Sulphur Dioxide Pollution:

- Deforestation around the body of water, and usually within the local area.
- Significantly lowers the PH levels of any bodies of water in the area. This causes water to become highly corrosive towards paints and common building materials.
- Sulphuric waters cause most aquatic life to die

- The gas can irritate eyes, and damage the respiratory tract which leads to more infections
- Rain becomes acidic causing widespread damage to plant life
- Even lightly sulphuric waters can cause a laxative effect which leads to dehydration.

Usually the presence of sulphuric waters means there's sulphur dioxide air pollution.
The air pollution is mainly caused by fossil fuel burning, and ore extraction contributes to a minor increase in SO2.

Clear Signs of Sulphuric Waters:

- Deforestation
- Light Hazing effect
- Scaly-like build up around pipe and edges of the body of water
- Blackened water
- Rotten egg water smell

Testing for Sulphuric Waters:

A sample of water is taken and mixed with a chemical preservative

Clearing of Sulphuric Waters:

- Filtration via carbon filters
- Manganese oxide coated greensand filter

Appendix G: Detergents & Turbidity Research [K.H.]

What are detergents?

Detergents are organic compounds, of which there are three types: anionic, cationic, and non-ionic. Anionic and cationic have permanent negative or positive charges, attached to non-polar (hydrophobic) C-C chains. Non-ionic detergents have no such permanent charge; instead, they have a number of atoms which are weakly electropositive and electronegative. This is due to the electron-attraction power of oxygen atoms.

There are two kinds of detergents with different characteristics: phosphate detergents and surfactant† detergents. Detergents that contain phosphates are highly caustic, and surfactant detergents are very toxic. The differences are that surfactant detergents are used to enhance the wetting, foaming, dispersing and emulsifying properties of detergents. Phosphate detergents are used in detergents to soften hard water and help suspend dirt in water. [79]

Effects of Detergent

Detergent as an important pollutant has serious risks to the natural ecosystem.They can pass into the wastewater treatment plants and can have bad effects on their performance. Detergent can affect fauna and flora, and they have direct and indirect effects on ecosystems. Eutrophication[2], foaming, and altering parameters such as temperature, salinity, turbidity, and pH are more important, and their effects need to be managed and controlled. Aerobic processes are able to degrade the most of detergents but anaerobic degradation is not possible because of restricted metabolic pathways and toxicity of them. Therefore, production of environment-friendly detergent is an important issue around the world. [80] The presence of higher levels of detergent in water for long periods of contamination, causes more and more depressed levels of bacterial populations. [81]

Measuring Detergent Levels

There are methods that have been developed to determine detergent concentrations, one of these methods include placing a droplet of the testing solution on a piece of Parafilm M and then measuring the contours of the solution using cameras, which is then approximated by ellipses to determine the contact angles.These observed contact angles can be compared with a calibrated standard curve of known detergent concentrations, to determine the concentration of the testing solution. [82]

Practically, for the type of device we are looking to develop, this method of detergent concentration determination is impractical. This is due to the need for multiple cameras observing a single droplet of a test solution, the environment necessary to take these readings is not practical in a small enclosed environment. Other methods may need to be explored in order to be able to determine detergent concentration levels with our desired device.

In terms of a sensor based solution to detergent determination, one method has been developed. This is based around an impedimetric sensor on an interdigitated array with electrode digits located at the bottom of microcapillaries formed in silicon dioxide.The detergent concentration is determined by the adsorption of detergents on the sensor surface, which affects the charge distribution in the electrical double layer. This type of sensor can be used to measure detergent residues starting from 5 ppm even in solutions with high conductivity. [83]

This device was created as a prototype / proof of concept, so there is no way to practically include it within our device. Therefore, my conclusion is that we will not be able to include a practical method of measuring detergent concentration levels with our device.

What is Turbidity?

Turbidity is the measure of relative clarity of a liquid. It is an optical characteristic of water and is a measurement of the amount of light that is scattered by material in the water when a light is shined through the water sample. The higher the intensity of scattered light, the higher the turbidity. Inorganic suspended materials, suspensoids or tripton, reduce light penetration, form adsorption and desorption surfaces, and are able to aggregate with dissolved substances, bacteria, and algae. [84]

Effects of Turbidity

High concentrations of particulate matter affect light penetration and ecological productivity, recreational values, and habitat quality, and cause lakes to fill in faster. In streams, increased sedimentation and siltation can occur, which can result in harm to habitat areas for fish and other aquatic life. Particles also provide attachment places for other pollutants, notably metals and bacteria. For this reason, turbidity readings can be used as an indicator of potential pollution in a water body.

Excessive turbidity, or cloudiness, in drinking water is aesthetically unappealing, and may also represent a health concern. Turbidity can provide food and shelter for pathogens. If not removed, the causes of high turbidity can promote regrowth of pathogens in the water, leading to waterborne disease outbreak, which have caused significant cases of intestinal sickness throughout the United States and the world. Although turbidity is not a direct indicator of health risk, numerous studies show a strong relationship between removal of turbidity and removal of protozoa. The particles of turbidity provide “shelter” for microbes by reducing material has been considered to aid in microbe survival. Fortunately, traditional water treatment processes have the ability to effectively remove turbidity when operated properly. [85]

Measuring Turbidity

Turbidity, as an optical property of water, is one of the more difficult parameters to measure. How murky or opaque water is can be a subjective measurement. Based on the measurement method, different units have been defined to standardize turbidity levels and allow comparisons. Today, there are three modern methods for measuring turbidity, and two for measuring total suspended solids. However, each method has its advantages and limitations:

As turbidity is caused by particles and coloured material in water. It can be measured relative to water clarity, or directly with a turbidity instrument such as a turbidimeter or turbidity sensor. Turbidity sensors may also be referred to as submersible turbidimeters. Water clarity methods involve a secchi disc or tube. They are often quick and inexpensive, but are only as accurate as the person using them. Turbidity meters use nephelometry or other optical scatter-detection techniques for optical technology, but instead of using samples cells, they can be placed directly in the water source to measure turbidity. In addition, turbidity sensors can be used for continuous turbidity measurements. However, when using a meter or a sensor, most turbidity data are not inter-comparable. Turbidity units such as NTU and FNU have “no intrinsic physical, chemical or biological significance”. Thus differences in suspended sediment type (e.g. algae, clay or sand) and

differences in instrument design will alter a turbidity reading. These instruments can be convenient and accurate tools as long as consistency is maintained.

Total suspended solids (TSS) are the main cause of turbidity. The most common, and accurate, method of measuring suspended solids is by weight. To measure TSS, a water sample is filtered, dried, and weighed. The method is the most accurate technique for measuring total suspended solids, however it is also more difficult and time-consuming.

The second method is a recent development by the U.S. Geological Survey. This organization has developed a technique for calculating suspended sediment from acoustic Doppler meter backscatter. While this method is not as accurate as a weight scale, it provides the opportunity for continuous suspended sediment measurements, just as turbidity measurements, however, this requires linear regression modeling and must be re-calculated for each sampling period and location. No standard model exits due to the differences in stream flow, sediment concentration, and particle size. [86]

Of the three methods presented above, the only method that will be practical for our device will be the turbidimeter method. These types of sensors are readily available for a low price and there are boards such as the SEN0189 which can interface effectively with an arduino microcontroller device. [87] In conclusion, the inclusion of this type of sensor would provide beneficial data to our device, in order to rate the quality of a water sample / body.

Appendix H: Hydrocarbons & Oils Research [L.W.]

Hydrocarbons are compounds that are composed of only carbon and hydrogen atoms, they mainly exist in a gaseous or liquid state and are less dense than water, meaning that it floats on bodies of water; hydrocarbons are also insoluble, when mixed with water, they form a layer over the water's surface [93]. Petroleum, also known as crude oil, is a substance composed mostly of various hydrocarbons, with trace amounts of other elements. Petroleum is formed from decaying organic matter, a process that takes millions of years to occur; it can be extracted from reservoirs, some of which are deep below water. Petroleum can be distilled to obtain the hydrocarbons in its mixture, which can be then used for fuel [94].

Oils and hydrocarbons can enter bodies of water through 4 methods, through the consumption of oil, transportation of oil (including spills), extraction or by natural seeps [88]. Since oil does not mix with water, it forms a layer of over the water, this prevents water being oxygenated and causes wildlife in the water to suffocate, causing severe damage to the local ecosystem [89]. The oils can also coat the animal, causing it to be unable to perform respiration or thermal regulation. It takes a small quantity of oil to contaminate a water supply and in sewage treatment, oil is a big risk [91]. Hydrocarbons, such as benzene, are known human carcinogens and can cause irreversible effects on the nervous system [95].

In the EU, under the Dangerous Substances Directive 76/464/EEC, which has now been changed to 2006/11/EC, mineral oil and hydrocarbons are classed as List 1 substances; meaning that water discharged into groundwater, seas, rivers or lakes, only up to 5 or 10mg/l of oil and hydrocarbons are permitted, depending on circumstances [91].

Sensors used to detect oil on top of water often use electromagnetic lasers and sensors to observe the returning laser; there should be a minimal amount of reflection if there is no oil present [89]. Another common method used to detect the amount of oil in water in the field is to use UV fluorescence, it works by using an UV lamp or laser, it then illuminates the water sample, the hydrocarbons within the water then emits a fluorescent light that a sensor can detect [90].

Upon doing research, there does not appear to be a cheap sensor that can be purchased to form a part of our device. Most oil in water detectors are part of their own system and do appear to interface with microcontrollers. It could be possible to use a white LED and a photodetector to act as a low-cost sensor, with the output voltage from the photodetector determining the presence of oil in the water [92]. However, it would require the LED and detector to be just above water level, and the effect of fast-moving water would have to be considered. It is also likely that the sensing circuit would require calibration.

Appendix I: Ammonia Research [V.B.L.]

What is ammonia?

"Ammonia is an inorganic compound composed of a single nitrogen atom covalently bonded to three hydrogen atoms that is an amidase inhibitor and neurotoxin. It is both manufactured and produced naturally from bacterial processes and the breakdown of organic matter. Ammonia is used in many industrial processes, and as a fertilizer and refrigerant. It is characterized as a colorless gas or compressed liquid with a pungent odor and exposure occurs by inhalation, ingestion, or contact." [96]

Typical ammonia concentration in surface waters:

"Water Natural levels in groundwaters are usually below 0.2 mg of ammonia per litre. Higher natural contents (up to 3 mg/litre) are found in strata rich in humic substances or iron or in forests. Surface waters may contain up to 12 mg/litre [96]. Ammonia may be present in drinking-water as a result of disinfection with chloramines.

The presence of ammonia at higher than geogenic levels is an important indicator of faecal pollution. Taste and odour problems as well as decreased disinfection efficiency are to be expected if drinking-water containing more than 0.2 mg of ammonia per litre is chlorinated, as up to 68% of the chlorine may react with the ammonia and become unavailable for disinfection. Cement mortar used for coating the insides of water pipes may release considerable amounts of ammonia into drinking-water and compromise disinfection with chlorine.

The presence of elevated ammonia levels in raw water may interfere with the operation of manganese-removal filters because too much oxygen is consumed by nitrification, resulting in mouldy, earthy-tasting water. The presence of the ammonium cation in raw water may result in drinking-water containing nitrite as the result of catalytic action." [97]

Testing for ammonia:

Not feasible. Ammonia has a P.H. of 11.6 and when considered as a primary pollutant in household aquariums, P.H. measurements allow inference of ammonia levels. Alternatively, single-use testing kits involving chemical re-agents are used, which are simply not feasible for our purposes. The cheapest electronic devices for measuring aqueous are around £200, far above our budget.

Appendix J: Arduino Firmware Code [M.T.]

```
//PONDER Water Quality Monitor - University of Surrey EEE3035 - Engineering Professiona
Studies - Group J Project - 2020/21
//V. Berdnikov-Levitsky, J. Ehuriah, K. Harrison, R. Rafky, P. Sodani, P. Soni, M.
Taylor, L. Williams
//GitHub repository link: https://github.com/sporadicE/PONDER

//Tested for use on Arduino Nano.
//The microcontroller takes measurements from the 4 sensors, goes to sleep for 1 minute
and repeats.

#include <SD.h>
#include <SPI.h>
#include <Wire.h> //I2C library
#include <RTClib.h> //download from https://github.com/adafruit/RTClib
#include <OneWire.h> //library for DS18S20 temperature sensor
#include <GravityTDS.h> //download from https://github.com/DFRobot/GravityTDS
#include <avr/sleep.h>

//pin connections
#define VRef 4.96 //adjust if voltage to sensors is less than 5V
//Note that sensor measurements are off when powering the Arduino with USB, as USB
provides only 4.6V to sensor rail
#define transistorSensors 8 //MOSFET used to cut power to sensor rail when in deep slee
#define transistorA9G 5 //MOSFET used to cut power to A9G

#define TempPin 7 //Temperature sensor on digital pin D7
OneWire ds(TempPin);

#define TDSPin A1 //TDS sensor on analog pin A1
GravityTDS gravityTds;

#define TurbPin A2 //Turbidity sensor on analog pin A2

#define pHPin A3 //pH sensor on analog pin A3
#define pHOffset -0.2 //set offset to calibrate pH sensor to pH7.0 in test solution

//SD card SPI: SCK D13, MISO (mislabelled MOSO) D12, MOSI D11, CS D10
#define CSpin 10 //SD card SPI enable pin
String dataString =""; // holds the data to be written to the SD card
File CSVfile;

//RTC I2C: SCL A5, SDA A4, SQW D2 (interrupts)
RTC_DS3231 rtc;
#define alarmPin 2

//-------------------------------------------------------------------------------------
--------------------------
void setup()
{
  //Changing the clock prescaler reduces ATmega329P's clock speed, thus lowering curren
draw in idle mode.
  //Idle current draw (empty sketch)(in mA): 19 (16MHz), 16 (8MHz), 14.3 (4MHz), 13.0
(2MHz), 12.1 (1MHz). Diminishing returns.
  //In deep sleep state, the microcontroller draws 7.5mA, regardless of clock speed.
  //Since this program spends almost all of its time in deep sleep, we needn't bother
altering clock speed.
  //CLKPR = 0x80;
  //CLKPR = 0x00; //Clock division factor. 0x01 = 2 (8MHz), 0x02 = 4 (4MHz), 0x03 = 8
```

```
(2MHz), 0x04 = 16 (1MHz), etc

  pinMode(CSpin, OUTPUT);
  pinMode(transistorSensors, OUTPUT);
  digitalWrite(transistorSensors, HIGH);
  pinMode(transistorA9G, OUTPUT);
  digitalWrite(transistorA9G, HIGH);

  Serial.begin(9600);
  //while (!Serial); // wait for serial port to connect. Needed for native USB

  //SD card setup
  Serial.println();
  Serial.print(F("Initializing SD card..."); //use of F() macro reduces dynamic SRAM u
  if (!SD.begin(CSpin)) // see if the card is present and can be initialized:
  {
    Serial.println(F("Card failed, or not present"));
    while (1);
  }
  Serial.println(F("card initialized."));

  if(!SD.exists("data.csv")) //checks if there is a data file, if not, creates one and
adds the headers to the file
  {
    SD.open("data.csv");
     dataString = F("Day,Month,Year,Hour,Minute,Second,Temperature,TDS,Turbidity,pH");
    CSVfile = SD.open("data.csv", FILE_WRITE);
    if (CSVfile)
    {
      CSVfile.println(dataString);
      CSVfile.close();
    }
    else
    {
      Serial.println(F("Error writing to file !"));
    }
  }

  //RTC setup
  if (! rtc.begin())
  {
    Serial.println(F("Couldn't find RTC"));
  }
  if (rtc.lostPower())
  {
    Serial.println(F("RTC lost power, let's set the time!"));
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__))); //set to compile time

  }
  pinMode(alarmPin, INPUT_PULLUP);
  rtc.disableAlarm(1);
  rtc.disableAlarm(2);
  rtc.clearAlarm(1);
  rtc.clearAlarm(2);
  rtc.writeSqwPinMode(DS3231_OFF); // Place SQW pin into alarm interrupt mode

  //TDS
  gravityTds.setPin(TDSPin);
```

```
  gravityTds.setAref(VRef);  //reference voltage on ADC, default 5.0V on Arduino UNO
  gravityTds.setAdcRange(1024);  //1024 for 10bit ADC;4096 for 12bit ADC
  gravityTds.begin();  //initialization

  Serial.println(F("void(setup) complete"));
}
//-------------------------------------------------------------------------------------------
-------------------------
void loop()
{
  delay(1000);
  String Time = ReadTime();
  float Temp = ReadTemp();
  float TDS = ReadTDS(Temp);
  float Turb = ReadTurb();
  float pH = ReadpH();

  dataString = String(Time) + "," + String(Temp) + "," + String(TDS) + "," +
String(Turb) + "," + String(pH);
  writeToCard(); // save to SD card

  //Set alarm to wake from deep sleep:
  DateTime now = rtc.now();  // Get current time
  //The alarm can either be set for a number of seconds, minutes, hours, etc from the
current time using now+TimeSpan
  //rtc.setAlarm1(now + TimeSpan(0, 0, 0, 10), DS3231_A1_Second); //every 10 seconds
  //rtc.setAlarm1(now + TimeSpan(0, 0, 1, 0), DS3231_A1_Minute); //every minute

  //Or the alarm can be set explicitly for a specific time using DateTime(Year, Month,
Day, Hour, Minute, Second)
  rtc.setAlarm1(DateTime(0, 0, 0, 23, 59, 59), DS3231_A1_Second); //once a minute, on
the dot (after 1s delay)

  //Alarm modes for alarm 1 (replace above as needed)
  //DS3231_A1_Second         When seconds match (i.e. once a minute)
  //DS3231_A1_Minute         When minutes and seconds match (i.e. once an hour)
  //DS3231_A1_Hour           When hours, minutes, and seconds match (i.e. once a day)
  //DS3231_A1_Day            When day, hours, minutes, and seconds match (i.e. day of
week)
  //DS3231_A1_Date           When date, hours, minutes, and seconds match (i.e. day of
month)

  enterSleep(); //runs function defined below.
}

//-------------------------------------------------------------------------------------------
-------------------------
//function definitions
void writeToCard(){
if(SD.exists("data.csv")) // check the card is still there
{
  CSVfile = SD.open("data.csv", FILE_WRITE);
  if (CSVfile)
  {
    CSVfile.println(dataString); // adds a new line of data
    CSVfile.close(); // close the file

    Serial.println();
```

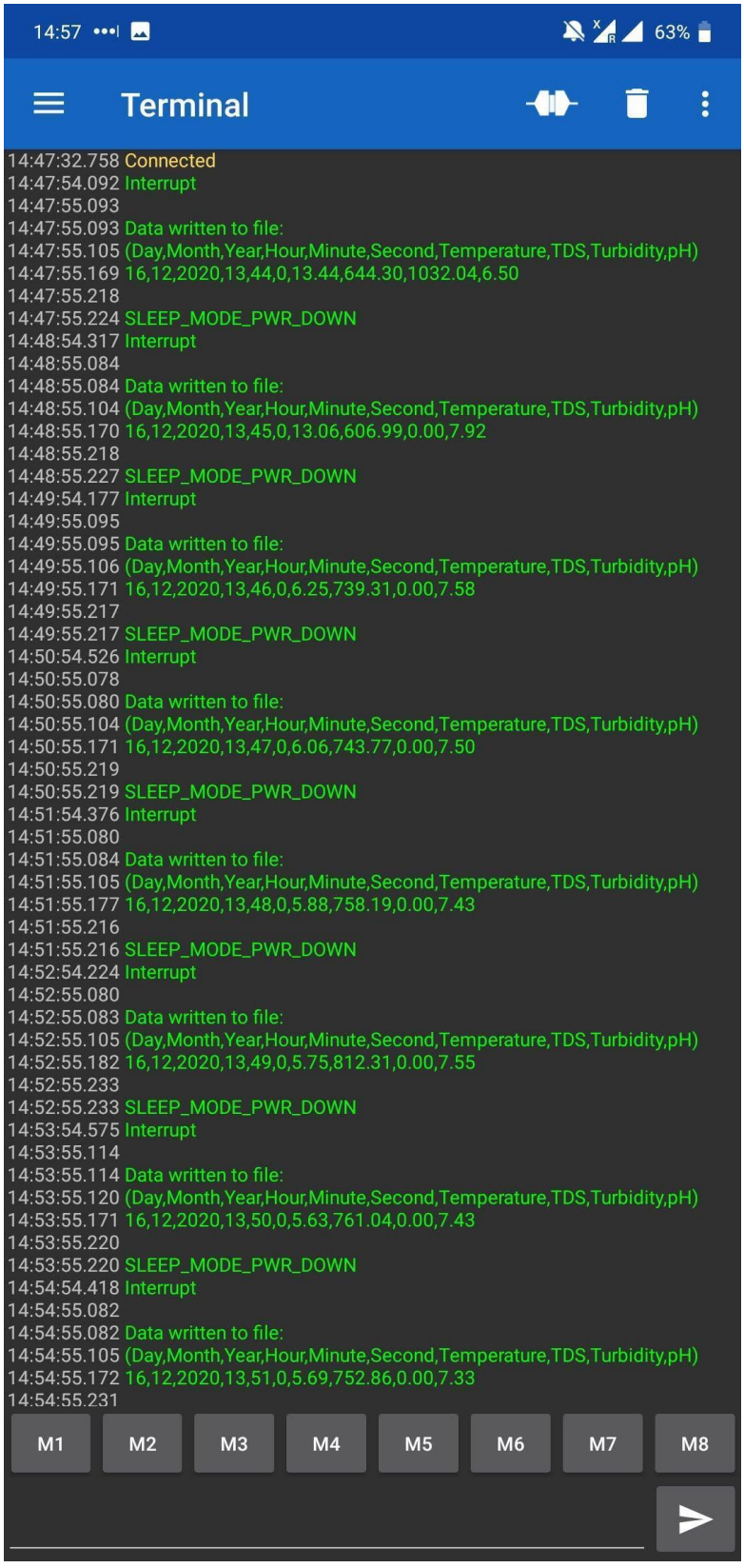
```
    Serial.println(F("Data written to file:")); //the next few lines print data to
serial monitor for debugging
     Serial.println(F("(Day,Month,Year,Hour,Minute,Second,Temperature,TDS,Turbidity,
pH)"));
    Serial.println(dataString);
    Serial.println();
    Serial.flush();                         // Ensure all characters are sent to the
serial monitor
  }
}
else
{
  Serial.println(F("Error opening data.csv"));
}
}

//-------------------------------------------------------------------------------------
-------------------------
String ReadTime()
{
DateTime now = rtc.now();
return String(now.day(), DEC) + "," + String(now.month(), DEC) + "," + String(now.
year(), DEC) + "," +
String(now.hour(), DEC) + "," + String(now.minute(), DEC) + "," + String(now.second(),
DEC);
//returns one long string, with commas seperating day,month,year,etc values
}

//-------------------------------------------------------------------------------------
-------------------------
//code used from DFROBOT wiki.
float ReadTemp()  //returns the temperature from one DS18S20 in DEG Celsius
{
  byte data[12];
  byte addr[8];

  if ( !ds.search(addr))
  {
    //no more sensors on chain, reset search
    ds.reset_search();
    return -1000;
  }

  if (OneWire::crc8( addr, 7) != addr[7])
  {
    Serial.println(F("CRC is not valid!"));
    return -1000;
  }

  if ( addr[0] != 0x10 && addr[0] != 0x28)
  {
    Serial.print(F("Device is not recognized"));
    return -1000;
  }

  ds.reset();
  ds.select(addr);
  ds.write(0x44,1); // start conversion, with parasite power on at the end
```

```
  byte present = ds.reset();
  ds.select(addr);
  ds.write(0xBE); // Read Scratchpad

  for (int i = 0; i < 9; i++) // we need 9 bytes
  {
    data[i] = ds.read();
  }

  ds.reset_search();

  byte MSB = data[1];
  byte LSB = data[0];

  float tempRead = ((MSB << 8) | LSB); //using two's compliment
  float TemperatureSum = tempRead / 16;

  if (TemperatureSum != 85) //Sometimes get anomalous 85 degree measurement after start
  {
    return TemperatureSum;
  }
  else
  {
    return ReadTemp(); //recursive function, repeat measurement until not 85 degrees
  }
}

//--------------------------------------------------------------------------------------
--------------------------
float ReadTDS(float Temp)
{
  gravityTds.setTemperature(Temp);  // set the temperature and execute temperature
compensation
  gravityTds.update();  //sample and calculate
  float tdsValue = gravityTds.getTdsValue();  // then get the value
  return tdsValue;
}

//--------------------------------------------------------------------------------------
--------------------------
float ReadTurb()
{
  int sensorValue = analogRead(TurbPin);
  float voltage = sensorValue * (VRef / 1024); //Converts the analog reading (0-1023) t
a voltage (0-5V)
  float Turb = -1120.4*voltage*voltage+5742.3*voltage-4352.9;
  if (voltage < 2.5) //for extremely dark liquids, the sensor cannot read beyond 3000
NTU, or under 2.5V
  {
    return 3000;
  }
  else if (voltage > 4.2) //A voltage of 4.2V or more means 0 NTU
  {
    return 0;
  }
  else
  {
```

```
    return Turb; //Between 2.5V and 4.2V, the sensor measures from 0-3000 NTU
  }
}

//-------------------------------------------------------------------------------------------
-------------------------
float ReadpH()
{
  int sensorValue = analogRead(pHPin);
  float voltage = sensorValue * (VRef / 1024); //Converts the analog reading (0-1023) t
a voltage (0-5V)
  float pH = 3.5 * voltage + pHOffset;
  return pH;
}

//-------------------------------------------------------------------------------------------
-------------------------
void enterSleep(){
  digitalWrite(transistorSensors, LOW); //turns off input to MOSFET gate, thus cutting
power to sensor rail
  digitalWrite(transistorA9G, LOW); //turns off input to MOSFET gate, thus cutting powe
to A9G

  sleep_enable();                          // Enabling sleep mode
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);   // Setting the sleep mode, in this case full
sleep

  noInterrupts();                          // Disable interrupts
  attachInterrupt(digitalPinToInterrupt(alarmPin), alarm_ISR, LOW);

  Serial.println(F("SLEEP_MODE_PWR_DOWN"));     // Print message to serial monitor
  Serial.flush();                          // Ensure all characters are sent to the serial
monitor

  interrupts();                            // Allow interrupts again
  sleep_cpu();                             // Enter sleep mode

  /* The program will continue from here when it wakes */

  digitalWrite(transistorSensors, HIGH); //turns on MOSFET, returning power to sensor
rail
  digitalWrite(transistorA9G, HIGH); //turns on MOSFET, returning power to A9G
  // Disable and clear alarm
  rtc.disableAlarm(1);
  rtc.clearAlarm(1);

  Serial.println(F("Interrupt"));           // Print message to show we're back
 }

//-------------------------------------------------------------------------------------------
-------------------------
void alarm_ISR() {
  // This runs when SQW pin is low. It will wake up the microcontroller

  sleep_disable(); // Disable sleep mode
  detachInterrupt(digitalPinToInterrupt(alarmPin)); // Detach the interrupt to stop it
firing accidentally
}
```

Note that the code is also available on Github at https://github.com/sporadicE/PONDER

Appendix K: Bluetooth Serial Monitor at Lake Bret [M.T.]



Appendix L: Ethical Considerations [K.H.]

SAGE-HDR

Response ID	Completion date
640816-640807-66800695	26 Oct 2020, 11:46 (GMT)

1	**Applicant Name**	Kieran Harrison
1.a	**University of Surrey email address**	kh00646@surrey.ac.uk
1.b	**Level of research**	Undergraduate
1.b.i	**Please enter your University of Surrey supervisor's name. If you have more than one supervisor, enter the details of the individual who will check this submission.**	David Carey
1.b.ii	**Please enter your supervisor's University of Surrey email address. If you have more than one supervisor, enter the details of the supervisor who will check this submission.**	david.carey@surrey.ac.uk
1.c	**School or Department**	Electrical and Electronic Engineering
1.d	**Faculty**	FEPS - Faculty of Engineering and Physical Sciences Sciences

2	**Project title**	PONDER

3	**Please enter a brief summary of your project and its methodology in 250 words. Please include information such as your research method/s, sample, where your research will be conducted and an overview of the aims and objectives of your research.**	A river pollution / water quality sensor that would tackle the Clean Growth grand challenge. The objective of this device, is to allow monitoring of water quality in remote areas with a low-cost and user friendly device. It will be a small bouy like device, which will float on the waters surface and use multiple sensors to measure and record different aspects of water quality (e.g. turbidity, temperature, pH, etc.). This data can then be transmitted to an external device, to be displayed on an app. The device will be developed using a Arduino microcontroller to interface the sensors with the communications (bluetooth / mobile) module within the device, that will allow it to communicate the recorded data to an external device. The device will be powered using a lithium ion battery, with connected solar polars on the top side of the device, which will allow it to operate isolated from a mains power supply for an extended period of time. We will be testing this device within electronics labs, and in local rivers and streams, to ensure accurate operation of the device.

4	**Are you making an amendment to a project with a current University of Surrey favourable ethical opinion in place?**	NO

5	**Does your research involve any animals, animal data or animal derived tissue, including cell lines?**	NO

7	**Does your project involve any of the following: human participants (including human data and/or any human tissue*); engineering and/or the physical sciences?**	NO

23	**Declarations**	• I confirm that I have read the University's Code on Good Research Practice and ethics policy and all relevant professional and regulatory guidelines applicable to my research and that I will conduct my research in accordance with these. • I confirm that I have provided accurate and complete information regarding my research project • I understand that a false declaration or providing misleading information will be considered potential research misconduct resulting in a formal investigation and subsequent disciplinary proceedings liable for reporting to external bodies • I understand that if my answers to this form have indicated that I must submit an ethics and governance application, that I will NOT commence my research until a Favourable Ethical Opinion is

		issued and governance checks are cleared. If I do so, this will be considered research misconduct and result in a formal investigation and subsequent disciplinary proceedings liable for reporting to external bodies. • I understand that if I have selected any options on the higher, medium or lower risk criteria then I MUST submit an ethics and governance application (EGA) for review before conducting any research. If I have NOT selected any of the higher, medium or lower risk criteria, I understand I can proceed with my research without review and acknowledge that my SAGE answers and research project will be subject to audit and inspection by the RIGO team at a later date to check compliance

24	**If I am conducting research as a student:**	• I confirm that I have discussed my responses to the questions on this form with my supervisor to ensure they are correct. • I confirm that if I am handling any information that can identify people, such as names, email addresses or audio/video recordings and images, I will adhere to the security requirements set out in the relevant Data protection Policy

Appendix M: Equality Impact Assessment (EIA) [M.T.]

EEE3035 Engineering Professional Studies

Semester 1 2020/21 - Year 3 Group Project

Equality Impact Assessment

Please read the guidelines associated with the completion of EIA before you complete this form. You may find the example EIAs on SurreyLearn useful.

Question	Response
1. Name of the project being assessed; include the group letter	PONDER (Group J)
2. Summary of aims and objectives of the project	To provide an affordable, unmanned and versatile system for the quantification of a body of water's quality. This will take the form of buoys with sensors to measure water temperature, TDS, turbidity, and pH. Data will be viewable using an accompanying mobile application.
3. What involvement and consultation has been done in relation to this project? (e.g. with relevant groups and stakeholders)	None
4. Who is affected by the project?	The end user of the PONDER device.
5. What are the arrangements for monitoring and reviewing the actual impact of the project?	No need for monitoring or review, as no known impacts against any protected characteristic group has been identified.

Protected Characteristic Group	Is there a potential for positive or negative impact?	Please explain and give examples of any evidence/data used	Action to address negative impact (e.g. adjustment to the project)
Disability	No known impacts		
Gender reassignment	No known impacts		
Marriage or civil partnership	No known impacts		
Pregnancy and maternity	No known impacts		
Race	No known impacts		
Religion or belief	No known impacts		
Sexual orientation	No known impacts		
Sex (gender)	No known impacts		
Age	No known impacts		

Evaluation: On the basis of your responses above, please complete the table below with one member of the group signing off the form.

Question	**Explanation / justification**	
Is it possible the proposed project could discriminate or unfairly disadvantage people?	We have not identified any means by which the project could discriminate or unfairly disadvantage people. The PONDER device and accompanying mobile application can be used equally by anyone, regardless of protected characteristics.	
Final Decision	**Tick the relevant box**	**Include any explanation / justification required**
1. No barriers identified, therefore project will proceed.	Tick	No known barriers or impacts were identified. The project can proceed without any changes.
2. You can decide to stop the project because the data shows bias towards one or more groups.		
3. You can adapt or change the project in a way which you think will eliminate the bias.		
4. Barriers and impact identified, however having considered all the available options carefully, there appears to be no other proportionate ways to achieve the aim of the project. Therefore, proceed with caution with the activity knowing the it may favour some people less than others, providing justification for this decision.		

Confirm that this EIA will be included in the project report.	Yes
Date Completed	26 October 2020
Review date (before the final project report)	

Appendix N: Data Management Plan (DMP) [J.E.]

DMP Title:

Project Name: PONDER
Description: This research project aims to analyse the concentration of pollutants in a given water body.
Institution: University of Surrey
Group Name & Letter: J

What data will you collect or create?
We will collect the concentration values for each specific chemical identified by our device.
We will also collect the date and time that each reading was obtained.
The received data will be presented in chronological order on our database with the date and time that these results were measured.
The size of the data collected will most likely be in the range of X KB – Y MB.
Location of the device.

How will the data be collected or created?
The concentration values will be collected by our water quality sensors on our device and displayed on the database system provided in our mobile application.
The date and time will be collected using a real time clock.
This data will be transferred over from the project device to the mobile application via bluetooth or mobile network data.
The GPS data can be collected by making a GET request to one of Google's location APIs.

Documentation and Metadata
What documentation and metadata will accompany the data?

1. Readings within our mobile application that has the UK water quality standards to compare our collected data to.
2. A HELP section of our mobile application detailing how to connect to our project device and transfer information.

Databases will be named according to a pre-agreed convention.
Should the mobile application run into any errors, a README text file would be present to advise you on how to retrieve the data locally stored on our project device.

Ethics and Legal Compliance
How will you manage any ethical issues?
There are no ethical issues in the generation of our results from our device. No personal information or other information apart from the analysis of pollutant concentration within a water body will be collected.

How will you manage copyright and Intellectual Property Rights (IPR) issues?
We do not intend to patent any hardware or software materials we produce during this project. We do not intend to use any product we produce for any monetary gain.

Storage and Backup
How will the data be stored and backed up during the research?

The data collected will be stored locally on an SD card within the device. Transferred data from the project device to a mobile device will be stored within the database on AWS that connects to the mobile application.

How will you manage access and security?
Our mobile application will only ask for permission to access bluetooth, mobile data and the internal storage of a mobile device. All data transmissions from the device to the phone will be secure.

Selection and Preservation
Which data are of long-term value and should be retained, shared, and/or preserved?
Any data obtained during this research may be considered to be of long-term use should another related study require our findings to be helpful.

State the format of the final dataset
Images will be stored as .png
Measured data will be stored in a database system (SQL)
Data in text files will be stored as .txt

Data Sharing
How will you share the data?
The data from the app can be shared to other users as a plaintext via email or any other communication apps i.e. SMS.
Long term, the data can be fetched from the DB using our APIs.

Are any restrictions on data sharing required?
No restrictions are applicable.

Responsibilities and Resources
Who will be responsible for data management?
The software team on this project will be responsible initially, if the device is by any parties interested, they will be responsible for maintaining the DB server, data maintenance etc.

What resources will you require to deliver your plan?
Credit for AWS to host a database server will be required which can be given free of charge to students. SD card for local storage on the device.

Bibliography

[1] https://www.unwater.org/water-facts/quality-and-wastewater/
[2] UN SDG Goal 6: Ensure access to water and sanitation for all:
https://www.un.org/sustainabledevelopment/water-and-sanitation/
[3] https://www.unwater.org/publications/world-water-development-report-2019/
[4] UN SDG Goal 12: Ensure sustainable consumption and production patterns: https://www.un.org/sustainabledevelopment/sustainable-consumption-production/
[5] http://www.unesco.org/new/en/natural-sciences/ioc-oceans/focus-areas/rio-20-ocean/blueprint-for-the-future-we-want/marine-pollution/facts-and-figures-on-marine-pollution/
[6] UN SDG Goal 14: Conserve and sustainably use the oceans, seas and marine resources
https://www.un.org/sustainabledevelopment/oceans/
[7] UN Water Quality and Wastewater: https://www.unwater.org/water-facts/quality-and-wastewater-2/
[8] https://github.com/sporadicE/PONDER
[9] https://en.wikipedia.org/wiki/Eutrophication
[10] https://uk.rs-online.com/web/p/sensor-development-tools/8829020
[12] https://edulab.com/product/nitrate-logger-sensor-edu-logger/
[13] https://cpc.farnell.com/hanna-instruments/hi-713/phosphate-checker/dp/IN07139
[14] https://www.dfrobot.com/product-1628.html
[15] https://www.dfrobot.com/product-1394.html
[16] https://www.frontline-safety.co.uk/honeywell-optima-plus-hydrocarbon
[17] https://coolcomponents.co.uk/products/ammonia-gas-sensor-mq-137
[18] https://www.dfrobot.com/product-1025.html
[19] https://www.dfrobot.com/product-1110.html
[20] https://www.dfrobot.com/product-1354.html
[21] https://www.dfrobot.com/product-1662.html
[22] https://www.dfrobot.com/product-1123.html
[23] http://hanzowater.com/water-related-information/tds-explained/
[24] http://www.solarelectricityhandbook.com/solar-irradiance.html
[25] https://solargis.com/maps-and-gis-data/download/world
[26] https://wiki.dfrobot.com/Solar_Power_Manager_5V_SKU__DFR0559
[27] http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers[-ATmega328P_Datasheet.pdf p34
[28] https://wiki.dfrobot.com/Solar_Power_Manager_5V_SKU__DFR0559
[29] https://www.ebay.co.uk/itm/1-2-5-10-DS18B20-Waterproof-Sensor-Digital-thermometer-Probe-With-PARASITE-MODE/262546255984
[30] https://www.dfrobot.com/product-1354.html
[32] https://www.dfrobot.com/product-1662.html
[33] https://www.dfrobot.com/product-1394.html
[34] https://www.dfrobot.com/product-1025.html
[35] https://www.dfrobot.com/product-1110.html
[36] https://www.amazon.co.uk/dp/B07DDBY7C5
[37]https://www.ebay.co.uk/itm/1X-Portable-4W-5V-Polycrystal-Solar-Panel-Laminated-Board-Component-DIY-Part/283979799211
[38] https://www.ebay.co.uk/itm/30Q-Rechargeable-Battery-3-7V-3000mAh-15A-High-Drain-Li-ion-Batteries/383766108971
[39] https://www.ebay.co.uk/itm/ABS-18650-Power-Bank-Cases-1X-2X-3X-4X-Battery-Holder-Storage-Box-With-Leads-4C/203009804846?ssPageName=STRK%3AMEBIDX%3AIT&var=503215432174&_trksid=p2057872.m2749.l2649
[40] https://www.dfrobot.com/product-1712.html
[41] https://www.ebay.co.uk/itm/ABS-18650-Power-Bank-Cases-1X-2X-3X-4X-Battery-Holder-Storage-Box-With-Leads-4C/203009804846?ssPageName=STRK%3AMEBIDX%3AIT&var=503215432174&_trksid=p2057872.m2749.l2649

[42] https://www.ebay.co.uk/itm/DS3231-High-Accuracy-RTC-Real-time-clock-module-like-DS1307-Arduino-PIC-UK-STOCK/251742357740
[43] https://www.ebay.co.uk/itm/Micro-SD-Card-Reader-SPI-for-Arduino-Raspberry-Pi-UK-Seller/322502815052
[44] https://www.ebay.co.uk/itm/HC-05-Arduino-Android-Wireless-Bluetooth-Serial-5v-Transceiver-Module-MASTER/313074658638
[45] https://www.dfrobot.com/product-2114.html
[46] https://bit.ly/3eRzmFI
[47] https://bit.ly/38Eblk8
[48] https://bit.ly/3kp03m3
[49] https://bit.ly/36oHzgw
[50] http://www.gammon.com.au/forum/?id=11497%20
[51] https://bit.ly/3hCtBNg
[52] http://diy-fever.com/software/diylc/
[53] Microsoft Corp., "Understanding the changes in the Root CA change for Azure Database for PostgreSQL Single server", Microsoft, Feb. 9, 2020, [Online], Available: https://docs.microsoft.com/en-us/azure/postgresql/concepts-certificate-rotation. [Accessed 9 November 2020].
[54] Agricultural Industries Confederation, "What is fertilisers", agindustries.org.uk, 2020
https://www.agindustries.org.uk/resource/what-is-fertiliser.html
(accessed Oct. 20, 2020)
[55] Byju's, "Types of fertilizers", byjus.com, 2020
https://byjus.com/biology/fertilizers
(accessed Oct. 20, 2020)
[56] R. Wiederholt and B. Johnson, "Environmental Implications of Excess Fertilizer and Manure on Water Quality" (NM1281, Revised Oct. 2017)
https://www.ag.ndsu.edu/publications/environment-natural-resources/environmental-implications-of-excess-fertilizer-and-manure-on-water-quality
(accessed Oct. 20, 2020)
[57] C. Stoltenow, "Cyanobacteria Poisoning (Blue-green Algae)" (V1136, Reviewed Dec. 2015)
https://www.ag.ndsu.edu/publications/livestock/cyanobacteria-poisoning-blue-green-algae
(accessed Oct. 20, 2020)
[58] United Utilities, "The pH of drinking water", 2020
https://www.unitedutilities.com/globalassets/documents/pdf/phfactsheet_acc16.pdf
(accessed Oct. 20, 2020)
[59] Lakeaccess.org, (n.d.). Lake Access -- Measuring the Acidity and Alkalinity of Lakes. [online] Available at: http://www.lakeaccess.org/russ/ph.htm [Accessed 3 Feb. 2016].
(accessed Oct. 20, 2020)
[60] Lenntech.com, (n.d.). Effects of acids and alkalis on aquatic life. [online] Available at: http://www.lenntech.com/aquatic/acids-alkalis.htm [Accessed 3 Feb. 2016].
(accessed Oct. 20, 2020)
[61] https://en.wikipedia.org/wiki/Organic_compound
[62] https://en.wikipedia.org/wiki/Hydrocarbon
[63] https://www.deq.idaho.gov/water-quality/surface-water/water-quality-criteria/common-measures/
[64] https://www.chemistryscl.com/advancedlevel/organic/water-soluble-organic-compounds/main.html
[65] https://www.epa.gov/dwreginfo/chemical-contaminant-rules
[66] https://denr.sd.gov/des/dw/SOC.aspx
[67] https://www.cdc.gov/niosh/topics/organsolv/default.html
[68] https://waterworldusa.com/organic-contaminants-in-drinking-water/
[69] https://www.deq.idaho.gov/water-quality/surface-water/water-quality-criteria/common-measures/
[70] https://www.lenntech.com/aquatic/organic-pollution.htm
[71] https://en.wikipedia.org/wiki/Eutrophication
[72] https://www.lenntech.com/aquatic/organic-pollution.htm
[73] https://www.usgs.gov/special-topic/water-science-school/science/biological-oxygen-demand-bod-and-water?qt-science_center_objects=0#qt-science_center_objects

[74] https://waterworldusa.com/organic-contaminants-in-drinking-water/
[75] https://www.usgs.gov/special-topic/water-science-school/science/biological-oxygen-demand-bod-and-water?qt-science_center_objects=0#qt-science_center_objects
[76] Manavi Yadav, Radhika Gupta and Rakesh Kumar Sharma, "Chapter 14 - Green and Sustainable Pathways for Wastewater Purification" 2019. https://www.sciencedirect.com/topics/chemistry/heavy-metal (accessed Oct. 20, 2020).
[77] Lenntech B.V., "Metals in aquatic freshwater, The way freshwater ecosystems deal with an excess of metals.," Lenntech.com, 2020. https://www.lenntech.com/aquatic/metals.htm (accessed Oct. 18, 2020).
[78] AbsoluteResource., "Testing for Metal Contamination in Water & Solid Matrices.," AbsoluteResource.com ,2020. https://www.absoluteresourceassociates.com/services/metals.cfm (accessed Oct. 18, 2020)
[79] Lenntech B.V., "Effects of detergents on aquatic freshwater life," Lenntech.com, 2020. https://www.lenntech.com/aquatic/detergents.htm (accessed Oct. 19, 2020).
[80] S. A. Mousavi and F. Khodadoost, "Effects of Detergents on Natural Ecosystems and Wastewater Treatment Processes: a Review," Environmental Science and Pollution Research, vol. 26, no. 26439–26448, Jul. 2019, doi: 10.1007/s11356-019-05802-x.
[81] I. Effendi, "Detergent Disposal into Our Environment and Its Impact on Marine Microbes," in International Conference on Environment and Technology, 2017, vol. 97, Accessed: Oct. 19, 2020. [Online]. Available: https://iopscience.iop.org/article/10.1088/1755-1315/97/1/012030.
[82] T. C. Kaufmann, A. Engel, and H.-W. Rémigy, "A Novel Method for Detergent Concentration Determination," Biophysical Journal, vol. 90, no. 1, pp. 310–317, Jan. 2006, doi: 10.1529/biophysj.105.070193.
[83] A. Bratov, N. Abramova, A. Ipatov, and A. Merlos, "New Chemical Sensor for Detergents Determination," presented at the 14th International Meeting on Chemical Sensors, Nürnberg/Nuremberg, Germany, 2012, [Online]. Available: https://www.ama-science.org/proceedings/details/1197.
[84] G Likens, M. E. Benbow, T. M. Burton, E Van Donk, J. A. Downing, and R. D. Gulati, Encyclopedia of Inland Waters. Elsevier B.V, 2009, pp. 699–704.
[85] United States Geological Survey, "Turbidity and Water," usgs.gov, 2018. https://www.usgs.gov/special-topic/water-science-school/science/turbidity-and-water?qt-science_center_objects=0#qt-science_center_objects.
[86] Fondriest Inc., "Measuring Turbidity, TSS, and Water Clarity - Environmental Measurement Systems," Environmental Measurement Systems, 2014. https://www.fondriest.com/environmental-measurements/measurements/measuring-water-quality/turbidity-sensors-meters-and-methods/.
[87] W. L. Hakim, L. Hasanah, B. Mulyanti, and A. Aminudin, "Characterization of turbidity water sensor SEN0189 on the changes of total suspended solids in the water," Journal of Physics: Conference Series, vol. 1280, no. 2, 2019, Accessed: Oct. 19, 2020. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1280/2/022064.
[88] How Does Oil Get into the Ocean? 2/11/2015 modified 15/04/2019 https://response.restoration.noaa.gov/about/media/how-does-oil-get-ocean.html#:~:text=There%20are%20four%20primary%20ways,extraction%2C%20and%20transportation%20of%20oil.
[89] Detection of oil in or on water, Control, 22/07/2016, https://www.controlglobal.com/articles/2016/detection-of-oil-in-or-on-water/
[90] Techniques for Measuring Oil in Water, Spectro Scientific 03/05/2016 https://www.spectrosci.com/resource-center/oil-in-water-and-soil/literature/whitepapers/guide-to-measuring-oil-in-water/
[91] Oil in Water Analysis, Awe International, 06/09/2016, https://www.aweimagazine.com/article/oil-in-water-analysis-1260/
[92] Low Cost Optic Sensor for Hydrocarbon Detection in Open Oceans, Lorena Parra, 2015, https://core.ac.uk/download/pdf/41823026.pdf
[93] Physical Properties of Hydrocarbons, Libre Texts, 28/02/2016 https://chem.libretexts.org/Courses/Sacramento_City_College/SCC%3A_Chem_309_-_General_Organic_and_Biochemistry_(Bennett)/Text/08._Organic_Chemistry_of_Hydrocarbons/8.13%3A_Physical_Properties_of_Hydrocarbons

[94] Petroleum, Britannica, https://www.britannica.com/science/petroleum
[95] Petroleum Products in Drinking Water, WHO, 2008, https://www.who.int/water_sanitation_health/dwq/chemicals/petroleumproducts_2add_june2008.pdf
[96] 2019. Ammonia (Compound) – Compound Summary [online] <https://pubchem.ncbi.nlm.nih.gov/compound/Ammonia> [Accessed 20 October 2020]
[97] 1996. Ammonia In Drinking-Water - Background Document For Development Of WHO Guidelines For Drinking-Water Quality. [pdf] Geneva: World Health Organization, p.2. Available at: <https://www.who.int/water_sanitation_health/dwq/ammonia.pdf> [Accessed 20 October 2020].