Table 1. RV table for battery

Points: __/1

Requirements	Verification
Battery can supply V_{out}>3.3V.	1. Setup a mock circuit with a 10kΩ resistor (large enough to prevent burning the resistor). 2. Connect the battery to resistor. 3. Measure voltage across resistor and confirm that V>3.3V.

Table 2. RV table for voltage regulator

Points: __/5

Requirements	Verification
Output voltage is within the range of 3.3V±5%.	1. Power on the regulator with 5V supply voltage. 2. Measure voltage between output and ground and confirm that the value is 3.3V±5%.
Voltage regulator can output up to 1A of current.	1. Connect the output to a resistor network with ~3.3Ω total resistance. 2. Power on the regulator with 5V supply voltage. 3. Measure current and confirm that the value is ~1A.

Table 3. RV table for water flow sensor

Points: __/2.5

Requirements	Verification
Functions for supply voltage 5VDC.	1. Connect sensor output to oscilloscope. 2. Connect sensor to 5V power source. 3. Pour water through the sensor. 4. Confirm that the oscilloscope shows output in the form of square waves with 5V amplitude.
Flow rate and frequency relation is described by the mathematical model F=11Q where F is frequency in Hz and Q is flow rate in LPM.	1. Connect sensor output to oscilloscope. 2. Connect sensor to 5V power source. 3. Provide a constant stream of water to the sensor for 30s, storing the water to a bucket. 4. Calculate the frequency of the output square waves from the oscilloscope data. 5. Measure the amount of water in the bucket. 6. Calculate the flow rate from the amount of water and time. 7. Check if the frequency and flow rate obtained fits the mathematical model; if not, update the mathematical model.

Table 4. RV table for ammeter

Points: __/2.5

Requirements	Verification
Functions for supply voltage 3.3V±5% DC.	1. Connect sensor output to oscilloscope. 2. Connect sensor to 3.3V power source. 3. Run current through the sensor. 4. Confirm that the oscilloscope shows output in the form of voltage readings.
Current measurement is accurate up to a 5% margin.	1. Connect sensor output to oscilloscope. 2. Connect sensor to 3.3V power source. 3. Run a 15A current through the sensor. 4. Confirm that the oscilloscope shows an output voltage of 675mV/A±5%.

Table 5. RV table for temperature sensor

Points: __/2.5

Requirements	Verification
Functions for supply voltage 3.3V±5% DC.	1. Connect sensor output to mock microcontroller. 2. Connect sensor to 3.3V power source. 3. Probe the air for its temperature. 4. Run a mock code to check if the sensor outputs any values.
The sensor can differentiate between pump normal operating temperature and pump overheating temperature.	1. Connect sensor output to mock microcontroller. 2. Connect sensor to 3.3V power source. 3. Probe room temperature water to simulate pump in normal operating temperature. 4. Run a mock code and note the output value. 5. Probe boiling water to simulate pump overheating temperature. 6. Run a mock code and note the output value. 7. Confirm that both readings' values are visibly distinct.

Table 6. RV table for vibration sensor

Points: __/2.5

Requirements	Verification
The sensor can differentiate between pump normal operating vibration and pump excessive vibration.	1. Connect sensor output to oscilloscope. 2. Connect sensor to 3.3V power source. 3. Shake the sensor moderately to simulate pump normal operating vibration. 4. Note down the output voltage on the oscilloscope. 5. Shake the sensor harder to simulate pump excessive vibration. 6. Note down the output voltage on the oscilloscope. 7. Confirm that both readings' values are visibly distinct.

Table 7. RV table for microcontroller

Points: __/15

Requirements	Verification
The C program on the microcontroller should be able to process data from the sensor module into a format that is suitable for transmitting data. This will be JSON.	1. Run test C program to check if signal/data from I/O pins can be recognized by the software. 2. Write test data within the C program to check if it can process data into the correct format. 3. Test the C program with sensor input to check if the data is processed correctly.
The C program can transmit data using built-in Wi-Fi capabilities to AWS server.	1. Design and run test C program in Arduino IDE to transmit mock data to AWS server. 2. Check server log of received data to confirm it matches the mock data provided in the code.

Table 8. RV table for SIM module

Points: __/3

Requirements	Verification
The module can establish a GPRS internet connection through the 2G network.	1. Design and run test C program in Arduino IDE to establish internet connection on the ESP32. 2. Run test C program to transmit mock data to AWS server. 3. Check server log to confirm received data matches mock data.

Table 9. RV table for database

Points: __/10

Requirements	Verification
AWS Lambda receives data given that the transmission module successfully sent data.	Send test data from hardware, and log the input received. Check if log matches with test data sent from hardware. Test this after cellular modem is tested.
Data is processed by script on AWS Lambda correctly, and processed data is stored into the database without corruption of data.	1. Create dummy data input within the script and log the processed data. Check if it is successfully processed. 2. Check database table after processed data is inserted to database. Make sure it does not affect past data, and new data is inserted without alteration.
Whenever the database is updated, AWS Lambda script should be executed to update json files in AWS S3 bucket.	Send a single data packet every minute and check if the csv files in AWS S3 bucket updates.

Table 10. RV table for website

Points: __/6

Requirements	Verification
The Javascript component of the website can read and process data from csv files on AWS S3 through ajax in real time.	Upload dummy json file with dummy data to S3. Log the read data on console, and check if data matches with dummy data on csv.
Warnings should be generated through analysis of data read from csv files.	Create dummy json files. One has data satisfying warning conditions, and another does not. Test warning analysis part of code with both of csv files. Check if warnings are generated if only if the csv file has data that satisfies warning conditions.
Data and warnings can be visualised on the website using d3.js.	Given the csv file, check if the website can display all data in graph and highlight the data with warnings.