A
Project Report
On
Water Quality Tester

Prepared by

14CE101 Raj Patel
14CE103 Sagar Patel
14CE109 Vicky Patel

Under the guidance of

Prof. Nikita Bhatt

Dr. Gaurang Panchal

Submitted to

Charotar University of Science & Technology
Degree of Bachelor of Technology
in Computer Engineering
CE324 -Software Group Project
Of 6th Semester of B.Tech

Submitted at

U. & P. U PATEL DEPARTMENT OF COMPUTER ENGINEERING

Faculty of Technology & Engineering, CHARUSAT

Chandubhai S. Patel Institute of Technology

At: Changa, Dist: Anand – 38842

Jan-June 2017

CHARUSAT
CHAROTAR UNIVERSITY OF SCIENCE AND TECHNOLOGY

CERTIFICATE

This is to certify that the report entitled "Water Quality Tester" is a bonafied work carried out by **Mr. Raj Patel (14CE101), Mr. Sagar Patel (14CE103)** and **Mr. Vicky Patel(14CE109)** under the guidance and supervision of **Prof. Nikita Bhatt and Dr. Gaurang Panchal** for the subject **Software Group Project (CE324)** of 6th Semester of Bachelor of Technology in **Computer Engineering** at Faculty of Technology & Engineering (C.S.P.I.T.) – CHARUSAT, Gujarat.

To the best of my knowledge and belief, this work embodies the work of candidate himself, has duly been completed, and fulfills the requirement of the ordinance relating to the B.Tech. Degree of the University and is up to the standard in respect of content, presentation and language for being referred to the examiner.

Under the supervision of,

Prof. Nikita Bhatt
Assistant Professor
U. & P. U Patel Dept. of Computer Engg.
C.S.P.I.T., CHARUSAT-Changa.

Dr Gaurang Panchal
Assistant Professor
U. & P. U Patel Dept. of Computer Engg.
C.S.P.I.T., CHARUSAT-Changa.

Dr. (Prof.) Amit Ganatra
Dean,
Faculty of Technology & Engineering
Head, U. & P. U Patel Department of Computer Engineering
C.S.P.I.T., CHARUSAT- Changa, Gujarat.

Chandubhai S Patel Institute of Technology (C.S.P.I.T.)

Faculty of Technology & Engineering, CHARUSAT

At: Changa, Ta. Petlad, Dist. Anand, PIN: 388 421. Gujarat

Abstract

Our application aims to develop a System that would check the quality of water by measuring TDS. This system will allow the user to analyze the present water quality in the purifier. An Android App will notify if the quality get vary from stability. It is Applicable in any remote water plant (Purifier). It is easy to use user interface of an android application. MQTT(Message Queue Telemetry Transport) protocol is used to fast data transfer to cloud and to App.

Acknowledgement

We are highly intended to the following personalities who have helped us throughout our project and without their support this project would never have been completed. We are extremely thankful to them. We are thankful to our Department HOD **Dr. Amit Ganatra,** who has provided us resources from the college and we are also very thankful to our project guide **Asst. Prof. Nikita Bhatt and Dr.Gaurang Panchal** who has provided her time and never ending support. For their support and help we heartily thank them. This project would be incomplete without them.

Table of contents

CHAPTER 1 INTRODUCTION

1.1PROJECT SUMMARY:

- The hardware system will be installed in the purifier system and will upload the data to cloud server.
- Cloud server (Broker) will send the data to its client connected to it.
- An Android app will fetch the data and will give notification about TDS.

1.2PURPOSE:

- To ease the work of the people.
- People can easily check hardness of their water themselves.

1.3SCOPE:

- 12hr scheduled calculation and upload is held.
- Manual refreshment to get present TDS value.
- Needs internet connection.
- High-speed data transmission over MQTT protocol.
- Graphical representation of Analysis over Android App.

1.4OBJECTIVE:

- Applicable in any remote water plant (Purifier).
- No need to check TDS value of water manually.
- Customize TDS value according to the hardness of water.
- Sound with notification when the TDS value fluctuate.
- Easy to use user interface of an android application.

1.5 TECHNOLOGY AND LITERATURE REVIEW:

- **Android Studio** is the official integrated development environment (IDE) for the Android platform.
- Based on JetBrains' IntelliJ IDEA software, Android Studio is designed specifically for Android development.
- **NodeMCU** is an open source IoT platform. It includes firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which is based on the ESP-12 module.

CHAPTER 2 PROJECT MANAGEMENT

2.1 PROJECT PLANNING:

2.1.1 PROJECT DEVELOPMENT APPROACH AND JUSTIFICATION :

We use waterfall method for the development of the system. In the development of the web application we started off with a prototype. After the development of the core prototype, the prototype was used for the project kickoff phase. Using previously build components to plan for the next iteration, we meet and plan for the next iteration to be executed.

2.1.2 PROJECT EFFORT AND TIME, COST ESTIMATION:

Parameters	**Count**		**Simple**	**Average**	**Complex**		**Total**
No. of user Input	1	X	3	4	6	=	3
No. of user Output	1	X	4	5	7	=	4
No. of Inquires	1	X	3	4	6	=	3
No. of Files	1	X	7	10	15	=	7
external Interface	2	X	5	7	10	=	10

Complexity Weight Factor:

Sr. No.	**Factors**	**Weights**
1.	Does the system require reliable backup and recovery?	**0**
2.	Are data communication required?	**1**
3.	Are there distributed processing functions?	**1**
4.	Is performance critical?	**3**
5.	Will the system run in an existing, heavily utilized operational environment?	**2**
6.	Does the system require online data entry?	**0**
7.	Does the on-line data entry require the input transactions to be built over multiple screens or operation?	**0**
8.	Are the master file updated on-line	**0**
9.	Are the inputs, outputs, files, or inquiries complex?	**1**
10.	Is the internal processing complex?	**1**
11.	Is the code designed to be reusable?	**4**

12.	Are conversion and installation included in the design?	**3**
13.	Is the system designed for multiple installations in different organizations?	**0**
14.	Is the application designed to facilitate change and ease of use by the user?	**4**

Weight	**Degree of Influence**
0	No Influence
1	Incidental
2	Moderate
3	Average
4	Significant
5	Essential

FP Count:

FP = count total*[0.65+0.01 *∑(Di)]

FP = 27* [0.65+0.01*20]

FP = 22.95

Function Point is: 22.95

Line of code (LOC) = FP*53 = 22.95*53 = 1216.35

KLOC = 1.21635

Software Project Type

Type	**a_b**	**b_b**	**c_b**	**d_b**
Organic	**2.4**	**1.05**	**2.5**	**0.38**
Semi-detached	3.0	1.12	2.5	0.35
Embedded	3.6	1.20	2.5	0.32

Effort = a_b*(KLOC) ^b_b

= 2.4*(1.21635) ^1.05

= 3.07 PM

Tdev = c_b * (Effort) ^d_b

= 2.5*(3.07) ^0.38

= 2.17 Months

2.1.3 ROLES AND RESPONSIBILITIES:

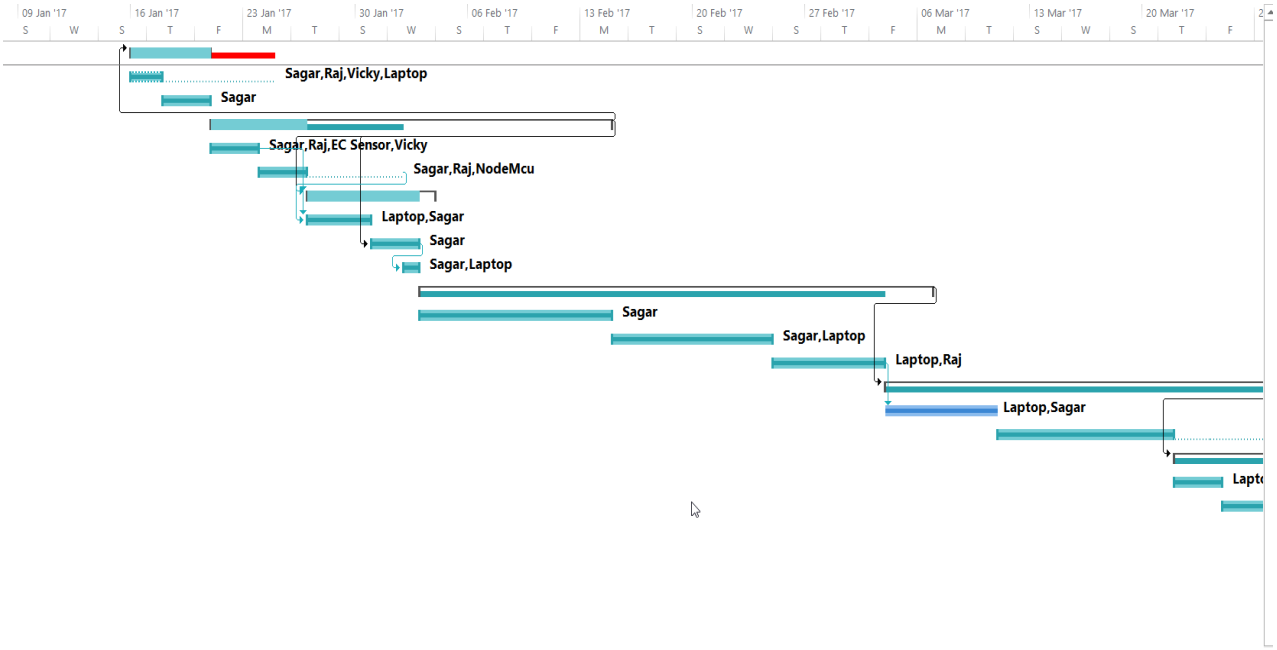
Developer	Roles and Responsibilities
14CE101	Designing, Testing, Documentation
14CE103	Designing, Coding, Requirement gathering
14CE109	Coding,Designing

2.2 PROJECT SCHEDULING (GANTT CHART/PERT/NETWORK CHART):

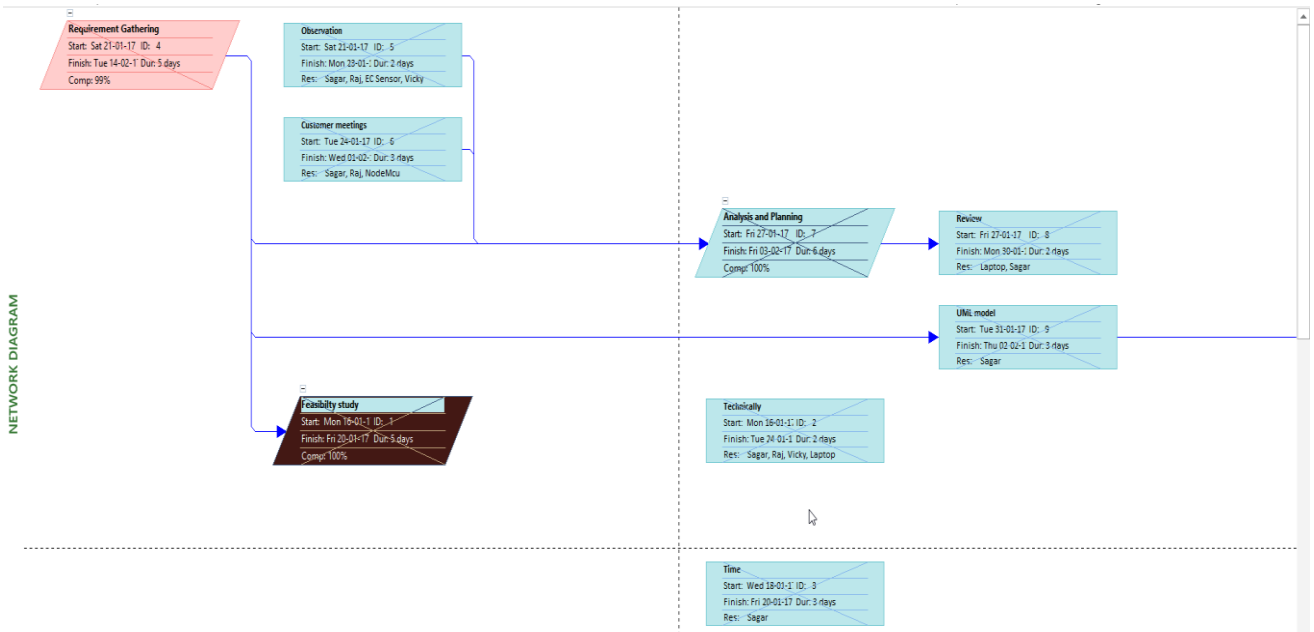
2.2.1 TASK NETWORK REPRESENTATION

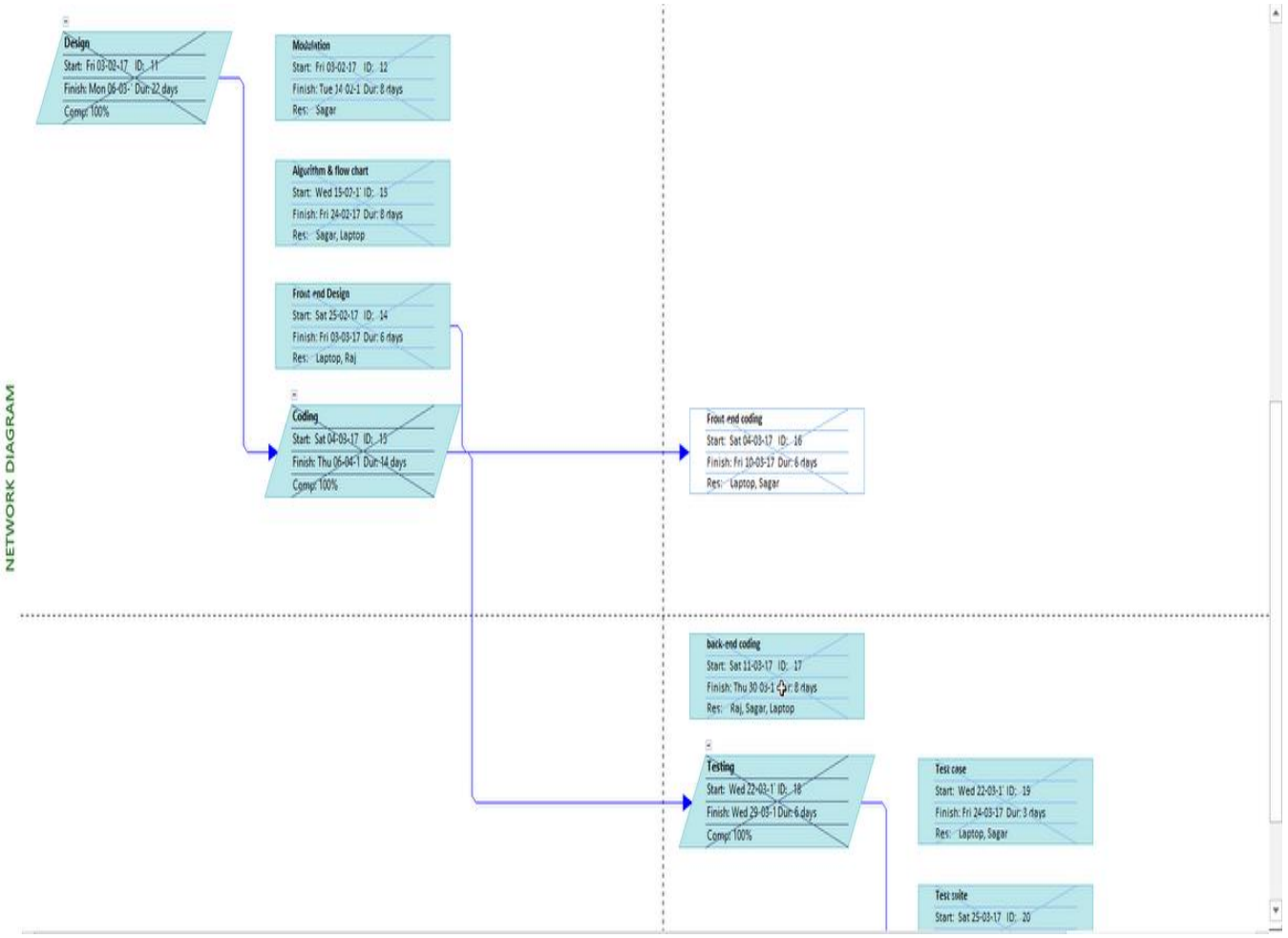
	Task Mode	Task Name	Duration	Start	Finish	% Complete	Predecessors	Resource Names
1		**Feasibilty study**	**5 days**	**Mon 16-01-17**	**Fri 20-01-17**	**100%**	**4**	**Sagar,Laptop,Raj,Vicky**
2		Technically	2 days	Mon 16-01-17	Tue 24-01-17	100%		Sagar,Raj,Vicky,Laptop
3		Time	3 days	Wed 18-01-17	Fri 20-01-17	100%		Sagar
4		**Requirement Gathering**	**5 days**	**Sat 21-01-17**	**Tue 14-02-17**	**99%**		**Sagar,Vicky,Laptop,NodeMcu**
5		Observation	2 days	Sat 21-01-17	Mon 23-01-17	100%		Sagar,Raj,EC Sensor,Vicky
6		Customer meetings	3 days	Tue 24-01-17	Wed 01-02-17	100%		Sagar,Raj,NodeMcu
7		**Analysis and Planning**	**6 days**	**Fri 27-01-17**	**Fri 03-02-17**	**100%**	**4,5,6**	**Sagar,Laptop**
8		Review	2 days	Fri 27-01-17	Mon 30-01-17	100%	5,6	Laptop,Sagar
9		UML model	3 days	Tue 31-01-17	Thu 02-02-17	100%	4	Sagar
10		DFD	1 day	Thu 02-02-17	Thu 02-02-17	100%	9	Sagar,Laptop
11		**Design**	**22 days**	**Fri 03-02-17**	**Mon 06-03-17**	**100%**		**Sagar,Raj,Vicky,Laptop**
12		Modulation	8 days	Fri 03-02-17	Tue 14-02-17	100%		Sagar
13		Algorithm & flow char	8 days	Wed 15-02-17	Fri 24-02-17	100%		Sagar,Laptop
14		Front-end Design	6 days	Sat 25-02-17	Fri 03-03-17	100%		Laptop,Raj
15		**Coding**	**14 days**	**Sat 04-03-17**	**Thu 06-04-17**	**100%**	**11**	**Sagar,Raj,Vicky,Laptop**
16		Front-end coding	6 days	Sat 04-03-17	Fri 10-03-17	100%	14	Laptop,Sagar
17		back-end coding	8 days	Sat 11-03-17	Thu 30-03-17	100%		Raj,Sagar,Laptop
18		**Testing**	**6 days**	**Wed 22-03-17**	**Wed 29-03-17**	**100%**	**15**	**Sagar,Raj,Laptop**
19		Test case	3 days	Wed 22-03-17	Fri 24-03-17	100%		Laptop,Sagar
20		Test suite	3 days	Sat 25-03-17	Tue 28-03-17	100%		Sagar,Laptop
21		**Final Documantation**	**2 days**	**Sat 29-04-17**	**Mon 01-05-17**	**100%**	**18**	**Vicky,Raj,Laptop,Sagar**
22		Deployment	1 day	Sat 29-04-17	Sat 29-04-17	100%		Raj

2.2.2 GANTT CHART REPRESENTATION



2.2.3 PERT CHART REPRESENTATION




NETWORK DIAGRAM
Design
Start: Fri 03-02-17 ID: 11
Finish: Mon 06-03- Dur:22 days
Comp: 100%
Modulation
Start: Fri 03-02-17 ID: 12
Finish: Tue 14-02-1 Dur: 8 days
Res: Sagar
Algorithm & flow chart
Start: Wed 15-02-1 ID: 13
Finish: Fri 24-02-17 Dur: 8 days
Res: Sagar, Laptop
Front end Design
Start: Sat 25-02-17 ID: 14
Finish: Fri 03-03-17 Dur: 6 days
Res: Laptop, Raj
Coding
Start: Sat 04-03-17 ID: 15
Finish: Thu 06-04-1 Dur: 14 days
Comp: 100%
Front end coding
Start: Sat 04-03-17 ID: 16
Finish: Fri 10-03-17 Dur: 6 days
Res: Laptop, Sagar
back-end coding
Start: Sat 11-03-17 ID: 17
Finish: Thu 30-03-1 Dur: 8 days
Res: Raj, Sagar, Laptop
Testing
Start: Wed 22-03-1 ID: 18
Finish: Wed 29-03-1 Dur: 6 days
Comp: 100%
Test case
Start: Wed 22-03-1 ID: 19
Finish: Fri 24-03-17 Dur: 3 days
Res: Laptop, Sagar
Test suite
Start: Sat 25-03-17 ID: 20

CHAPTER 3 SYSTEM REQUIREMENTS STUDY

3.1 USER CHARACTERISTICS

- More than in home usage, it will spread more towards industry and organization automation, as the water system will be placed remotely from work flow.
- Need to install the hardware in purifier system and install android app in cell.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

- Processor: Pentium 2.4 GHz or above
- Memory: 256 MB RAM or above
- Hard Disk: 3 GB or above

3.3 ASSUMPTIONS AND DEPENDENCIES:

- User needs to have internet connection.
- The value of TDS will be optimal value.

CHAPTER 4 SYSTEM ANALYSIS

4.1 STUDY OF CURRENT SYSTEM

- There is no current system but there is only TDS meter to check TDS.

4.2 PROBLEM / WEAKNESSES OF CURRENT SYSTEM

- Problem is that, People need to call service man to check TDS and every time so it is not possible to call service man to come and just come and check TDS.

4.3 REQUIREMENTS OF NEW SYSTEM

4.3.1 FUNCTIONAL REQUIREMENTS

- **Connection to APP**

  I/P:-Open the android app and click on connect
  O/P:-Connected, Ready to use.

- **Insert Probe**

  I/P:- Insert Probe in Water
  O/P:- TDS will be displayed on the App screen.

- **Customize TDS value.**

  I/P :- User can set TDS value.
  O/P :- Output will be based on that value.

- **Display Notification**

  →Notifies the Result in Notification bar.

4.3.2 NON FUNCTIONAL REQUIREMENTS

- Usability-Easy to use.
- Reliable- There is less probability of failures.
- Portability- It can be used in different environments as it is on internet.
- Fault tolerance- In case of failure, the system can recover quickly.
- Extensibility- New features can be added to it in the future.

4.4 FEASIBILITY STUDY

4.4.1 DOES THE SYSTEM CONTRIBUTE TO THE OVERALL OBJECTIVES OF THE ORGANIZATION?

- Yes, the system developed contributes to all objects.

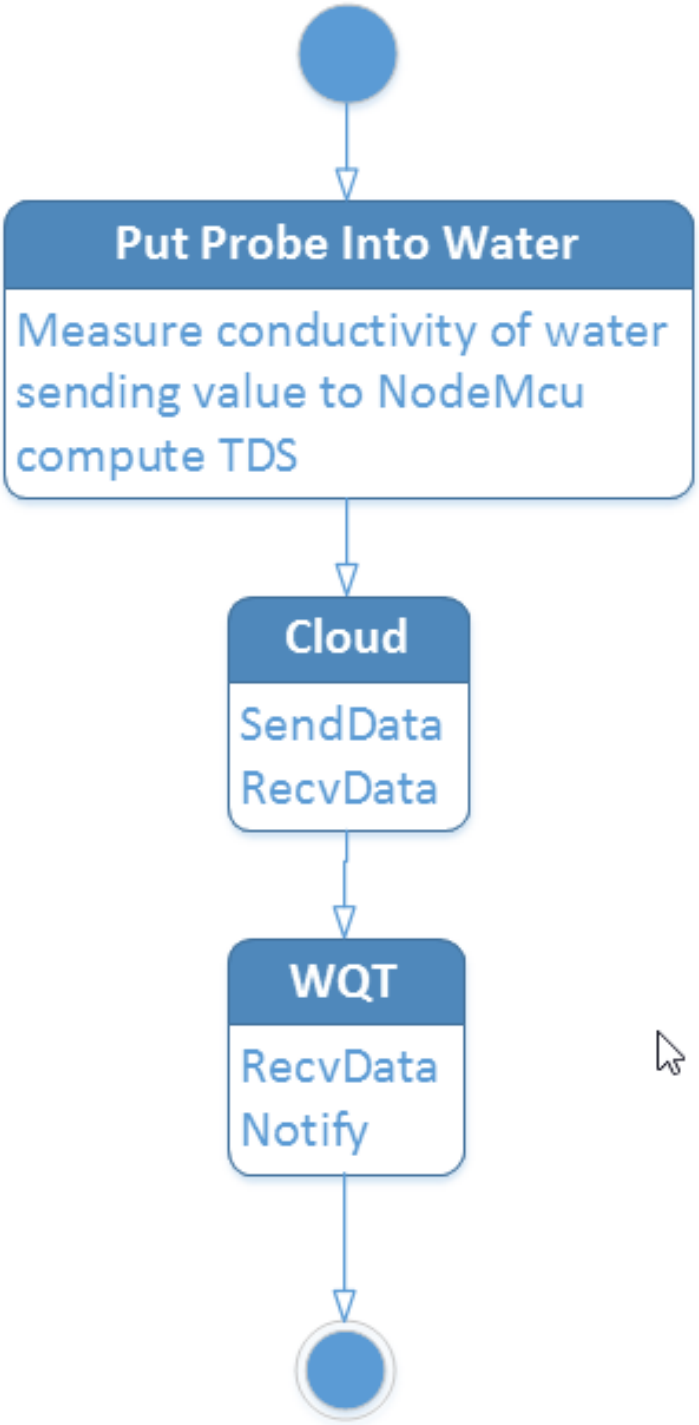
4.4.2 CAN THE SYSTEM BE IMPLEMENTED USING THE CURRENT TECHNOLOGY AND WITHIN THE GIVEN COST AND SCHEDULE CONSTRAINTS?

- Yes, the system can be implemented using current technology and with given cost and schedule constraints.

4.4.3 CAN THE SYSTEM BE INTEGRATED WITH OTHER SYSTEM WHICH ARE ALREADY IN PLACE?

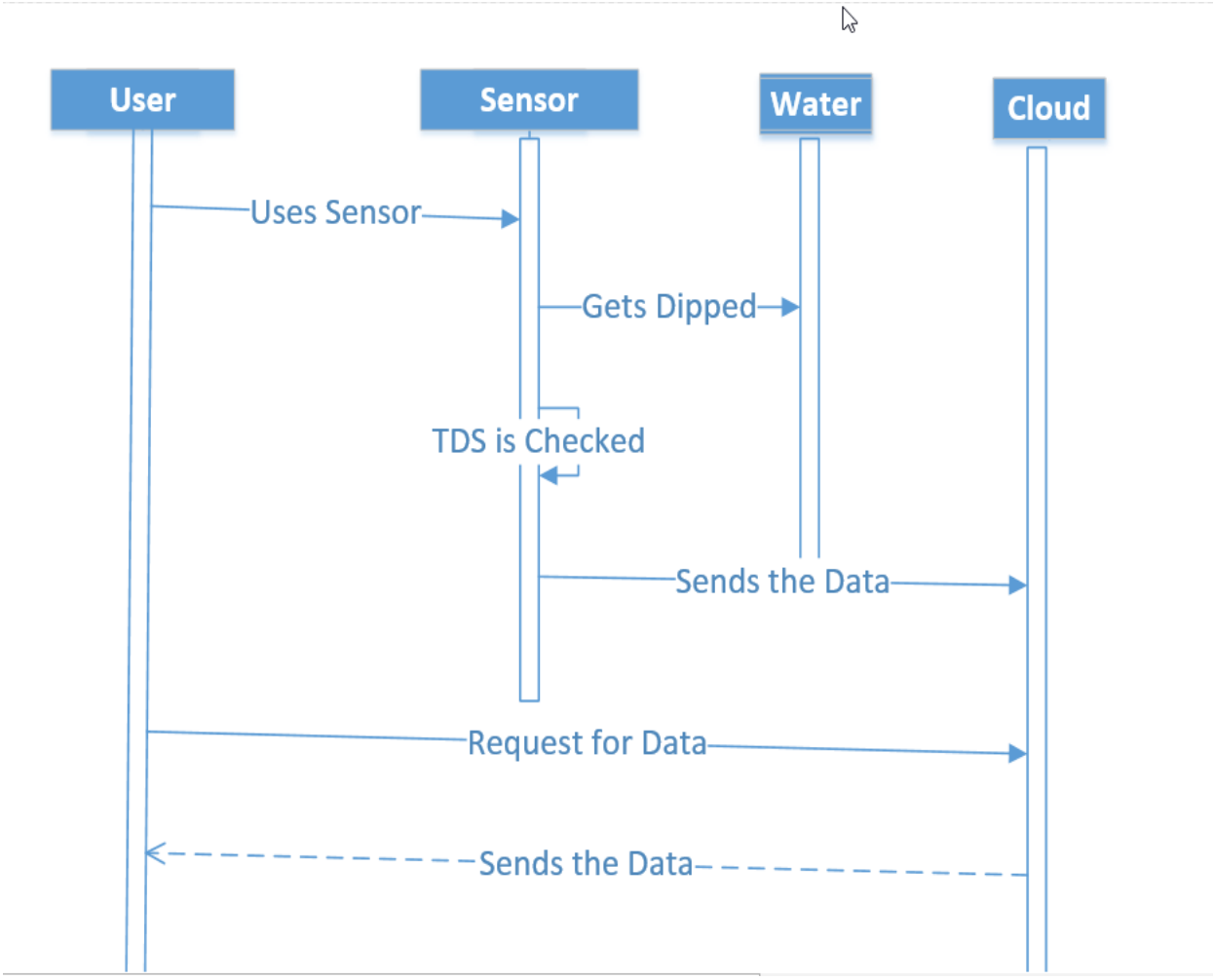
- Yes, the system can be integrated with other systems.

4.5 ACTIVITY/PROCESS IN NEW SYSTEM



4.6 SYSTEM ACTIVITY (USE CASE DIAGRAM)

4.7 SEQUENCE DIAGRAM



CHAPTER 5 SYSTEM DESIGN

5.1 SYSTEM APPLICATION DESIGN

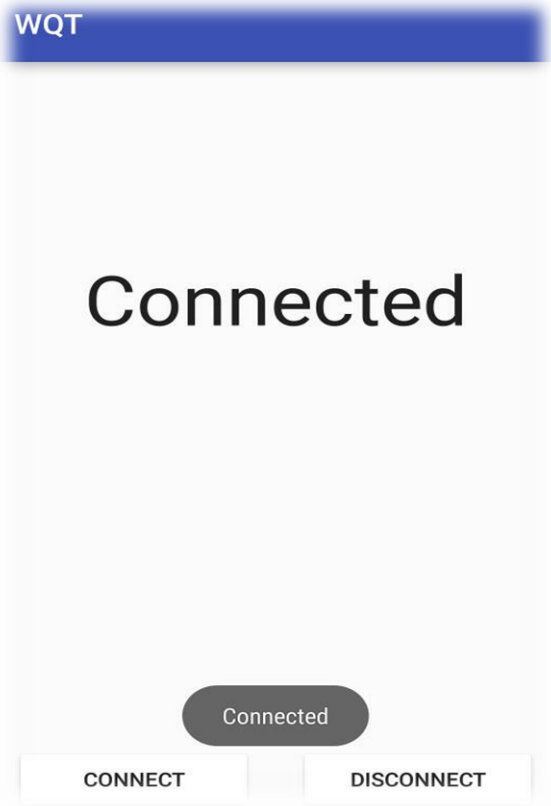
5.1.1. METHOD PSEUDO CODE

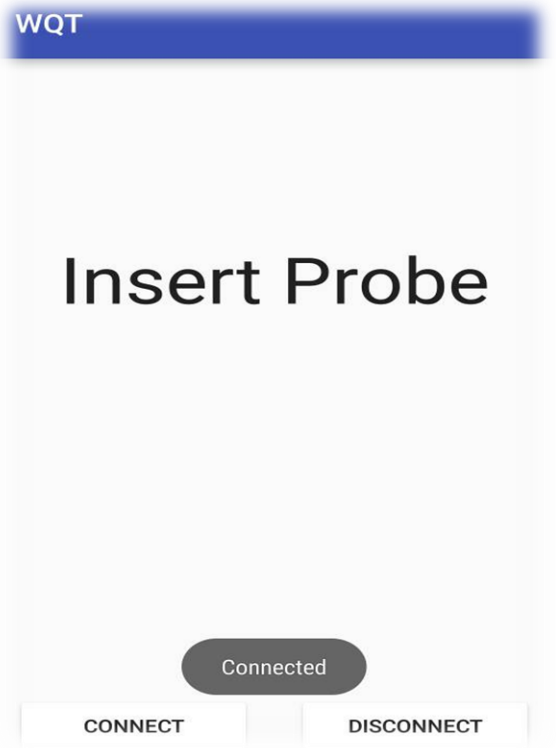
```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>

#define tdspin A0
#define deepsleep  30

#define wifi_ssid "Mi-Fi"
#define wifi_password "md85w0vr0u"

#define mqtt_server "m13.cloudmqtt.com"
#define mqtt_user "nogmsmyx"
#define mqtt_password "cRc2t6dZtU6j"
#define tdstopic "/TDS"

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
  Serial.begin(9600);
  setup_wifi();
  client.setServer(mqtt_server, 14448);
  pinMode(tdspin,INPUT);
}

void setup_wifi() {
  delay(10);
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(wifi_ssid);

  WiFi.begin(wifi_ssid, wifi_password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    if (client.connect("ESP8266Client", mqtt_user, mqtt_password)) {
      Serial.println("\nCONNECTED");
    } else {
      Serial.print("Failed\tState:");
      Serial.print(client.state());
```

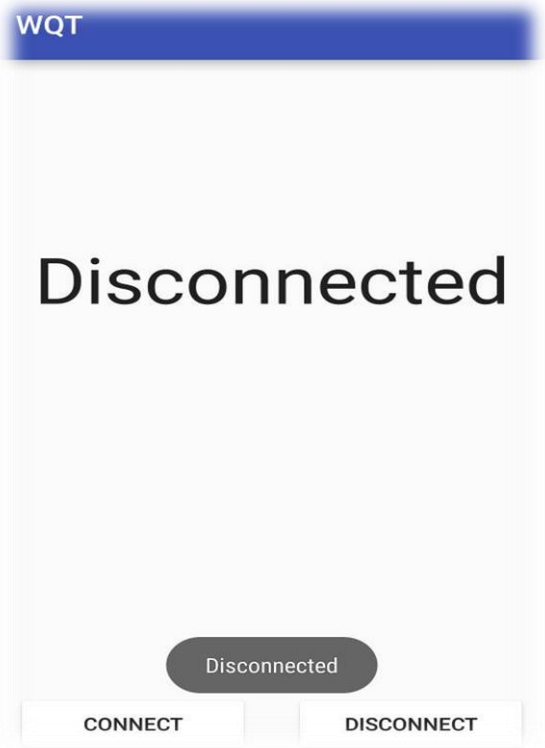
```
      Serial.println("...try again in 5 seconds");
      delay(5000);
    }
  }
}

bool checkBound(float newValue,float prevValue, float maxDiff) {
  return !isnan(newValue) &&
        (newValue < prevValue - maxDiff || newValue > prevValue + maxDiff);
}
long lastmsg = 0;
long now;
float tds = 0;
float diff = 1;

void loop() {
  if (!client.connected()) {
    reconnect();
  }
  client.loop();
    float newtds =analogRead(tdspin);
  float probe = newtds * (5.0 / 1023.0);
  if(probe>=0.1){
    if (checkBound(newtds, tds, diff)) {
      tds = newtds;
      Serial.println("TDS value: ");
      Serial.print(String(tds).c_str());
      client.publish(tdstopic, String(tds).c_str(), true);
    }else{
      Serial.print("\nTDS value: ");
      Serial.println(String(tds).c_str());
    }
       Serial.print("\nClosing MQTT connection...");
    client.disconnect();

    Serial.print("\nClosing Wi-Fi Connection...");
    WiFi.disconnect();

    Serial.print("\n\nEnter in Deepsleep mode for ");
    Serial.print(deepsleep);
    Serial.print(" seconds...\n");
    ESP.deepSleep(deepsleep * 1000000, WAKE_RF_DEFAULT);
    delay(100);

  }else{
      Serial.print("\n\nPlease insert probe...");
      client.publish(tdstopic, "Insert Probe", true);
      while(probe<=0.1){
        now = millis();
        if(now - lastmsg > 10000){
          lastmsg = now;
          Serial.print("\n\nPlease insert Probe...");
          client.publish(tdstopic, "Insert Probe", true);
        }
        newtds =analogRead(tdspin);
        probe = newtds * (5.0 / 1023.0);
        delay(100);
      }
    }
}
```

5.2 INPUT/OUTPUT AND INTERFACE DESIGN

5.2.1 SAMPLES AND INTERFACE







CHAPTER 6 IMPLEMENTATION PLANNING

6.1 IMPLEMENTATION ENVIRONMENT

1. **Creating a hardware system.**
   Using NodeMCU Development board and Arduino IDE, we create a system which check the quality of water by checking the TDS (Total Dissolved Solids) value of water by using the EC sensor which return the analog value of TDS and will queue to the server.
2. **Creating Cloud MQTT broker services.**
   Here we used the CloudMQTT serves powered by amazon web services and create the MQTT Broker service to give a portal to publish and subscribe facility to clients.
3. **Creating an Android App.**
   Android App is created to give a User Interface and to analyse the current Statistics of the water. Which is also notify when the event happen.

6.2 PROGRAM/MODULES SPECIFICATION

1. **Power optimization:**
   The device will send the data at every 6 hours and go to the DEEPSLEEP mode in that 6 hours.
2. **Device status:**
   The hardware will also send the message if the device is not inserted/installed properly. And will notify every 10 minutes.
3. **App Background Services:**
   Notification will also arrive if the device/phone is not running app or in sleep mode i.e. App will also run in background.
4. **Streaming:**
   Continuous streaming of payload will help the user to get the present status of purifier.

CHAPTER 7 TESTING

7.1 TESTING PLAN:

→We are using White Box Testing for this System.

7.2 TEST CASES

Test Suites No: 1

Test Suite Detail: Connection to the system, press **Connect** button and insert the Probe in water.

Test Case ID	Function Name	Test Case (condition)	Expected Results	Actual Result	Pass/Fail
1	-	Press Connect button	Connected	Connected	Pass

Test Suites No: 2

Test Suite Detail: Disconnection to the system, press **Disconnect** button and remove the Probe from water.

Test Case ID	Function Name	Test Case (condition)	Expected Results	Actual Result	Pass/Fail
2	-	Press Disconnect button	Disconnected	Disconnected	Pass

CHAPTER 8 LIMITATION AND FUTURE ENHANCEMENT

8.1 LIMITATIONS:

→TDS value will fluctuate as DIY EC sensor is not measuring the exact value.

→Will not be able to reset as well as work in Deep sleep mode.

8.2 FUTURE ENHANCEMENTS:

→Present water minerals.

→Calculate the acidity (pH) of water resource (Lake, river) to detect chemical linkage from the industries.

→Can be interface into any organization's analytical machine to have an eyes over.

→Stream the data directly to purifier vendors to get interface with system.

CHAPTER 9 CONCLUSION AND DISCUSSION

9.1 PROBLEM ENCOUNTERED AND POSSIBLE SOLUTIONS

- There was a problem in uploading video. So we increased the maximum size that a file can take in web.config file.
- We needed to hide some controls from users that were guest users so we used display none to hide them.

9.2 SUMMARY OF PROJECT WORK

- This system will allow users to get themselves registered as professional or as student. Professionals can add technologies and contents. Students can learn different technologies using the website and also ask queries to professionals. Students can follow professionals. When professionals add content, the followers will receive an email.

CHAPTER 10 REFERENCES

- http://www.arduino.cc/
- http://www.lenntech.com/calculators/conductivity/tds-engels.htm
- http://www.hivemq.com/blog/mqtt-client-library-enyclopedia-paho-android-service
- http://iot.eclipse.org/getting-started#tutorials
- https://www.ibm.com/developerworks/cloud/library/cl-mqtt-bluemix-iot-node-red-app/