Department of Computer Science

B.Tech Computer Science

Academic Year 2020 - 2021

Prediction of Dissolved Oxygen from pH and Water Temperature in Aquaculture Prawn Ponds

Tarun Kantiwal
(COE17B031)
Under
Dr. Munesh Pal Singh

A report submitted to describe my summer intership work
and to present in internship panel

IIITDM Kancheepuram
Department of Computer Science
Vandalur-Kelambakkam Road
Chennai-127
Tamil Nadu
India
T: +91-44-2747 6300
F: +91-44-2747 6301

Abstract

This report describes my internship experience at the Indian Information of Information Technology Design and Manufacturing in Kancheepuram , and it is divided into four chapters.

Chapter One summarizes the projects and It gives us the introduction and need of the project.

Chapter Two dealt with the idea of IoT in the project to send data to cloud and and make use of the data for further calculations.

Chapter Three provides how to and what to do with data over the cloud and project is to handle data to make inferences (Machine Learning or Cleaning data).

Chapter Four Dealt with the Back end or hosting server and make prediction online using API the work done in Nodejs.

Chapter Five In this chapter we will finally done our work for web App and Mobile Application for end user.

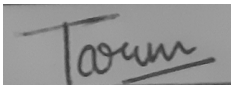
Acknowledgements

Working on this project is very good experience of me. During these five month internship i learned a lot and all my courses made equal and big contribution in completing this internship, Internet of things, Programming, Data cleaning and all maths courses helped me a lot.

I want to help Dr. Munesh Pal Singh to help me throughout the project by providing me knowledge about IoT and keep me calm in while solving error or when i feel stuck in some difficult situation

Further on I want to thank stackoverflow, github and all programming site because these community helped me a lot while i stuck in error regarding cloud, Web App, Mobile App, Tensorflow Model for web and App.

I certify that the work presented in the dissertation is my own unless referenced.

Signature: 

Date: 14/Oct/2020

Contents

Chapter 1

Introduction

A crucial activity in prawn farming is monitoring prawn pond water quality. Variables such as dissolved oxygen (DO), pH, temperature, and salinity are commonly monitored using sensors Temp sensor pH sensor and many other sensor are widely used by small farmers and pond owner to monitor water quality for fishes and agriculture related work. Sensor monitoring comes with challenges such as: purchasing and main- taining sensors is costly, gathering readings with hand-held sensors over many ponds is time consuming, readings can be incorrectly logged, and sensors can fail. Such problems can be mitigated by reducing the number of sensors required. A sensor can be made redundant if its associated variable can be predicted from other sensor data In this study a recurrent neural network (RNN) , a linear regression model,are used to predict dissolved oxygen from pH and water temperature sensor readings.

Aquaculture consists of the set of activities, knowledge and techniques for the breeding of aquatic plants and some species of animals. This activity has a great importance in economic development and food production. Continuous monitoring of the physical, chemical and biological parameters of pond water helps not only to predict and control the negative conditions of aquaculture, but also to avoid environmental damage and the collapse of the production process. The monitoring of physical and chemical variables such as: oxygen, temperature and pH in water are vital to maintain adequate conditions and avoid undesirable situations that may lead to the collapse of aquaculture systems

1.1 Aims and Objectives

The aim of this project is to get the main factor of water Dissolved Oxygen get predict by some given properties of water using IoT solution and Machine Learning Model and made them accessible to end user without any hustle so that they can access in Mobile device like Application of iOS and Android and web App.

Here is the list of the necessary and complete set of objectives that we will need to achieve in order to satisfy our above listed aim using modern day technologies :.

1. Undertook a relevant background study to identify existing work in the area, and to identify appropriate techniques which can be adopted to produce a

solution in this project. like is there any model which can help us to do thing easily

2. In the task first we have and we want data of temp and ph so that we can make a prediction of dissolved oxygen but before making prediction data cleaning we need to done change format like json and other compatible with mobile app or web app.

3. We have to implement one Tensorflow model by which we can make prediction and calculate approximated output given any particular input we have to use Tensorflow Lite (For Mobile App Development) and Tensorflow.js for web app Development.

4. Next thing we need to do is get data from pond in real time and upload all neccessary data like pH and Temp to make prediction using Internet of things example Arduino UNO is perfect to use data and Wifi module to upload data on cloud. Search for free cloud provider is challeging because no one provide free service

5. After getting model ready and data uploaded to cloud we have to make data accessible to end user like farmer and fish farmer so that they can use prediction to make aquatic and agriculture life better. So for this purpose we have to make one Web App and Mobile App with very easy interface so everybody can use it.

6. In Last all the blocks are ready now we have to combine them all and make one final product which can make good predictions.

1.2 Project Approach

The Project will used many new and advanced technology which will make good and better product finally and which will make significant impact on society so we will follow all the above aim and make final product good as all the technology we will use are new so we have to handle thing by learning and doing. Tensorflow is very good for advanced Machine Learning thing and Java Script is good for web and cross platform native mobile application. The problem which we will face is that the data which we are using is having the temperature between 25-31°C and ph is of normal water (6-8) so, making prediction out of this bound will give us error in prediction like for very cold water (o-10 °C) it will give error as we have to collect more data in this range and train the model again but it will not be difficult task as we will get data any time soon after this Lock Down end. We have to host one prediction server which will make online prediction using API.

1.3 Dissertation Outline

Below is the task and the chapter in which detailed description of each task is included chapter wise and summary of the chapter included in Dissertation Outline.

Chapter 1, already talk about what the project is all about where IoT and Machine Learning and Web Model will be used.

Chapter 2, In chapter two we will Talk about Iot and Cloud data handeling.

Chapter 3, Data Computation, cleaning and make inference from data for predictions.

Chapter 4, Talk about Api which will be user for Mobile App and web app.

Chapter 5, One website and Mobile app will be used for Desktop and Mobile Users.

Chapter 6, It shows about out testing and App and web App for are all the things working fine ?

Chapter 7, Conclusion of our Work and Future.

Chapter 2

IoT and Cloud

2.1 IoT and Cloud Introduction

The Internet of Things (IoT) involves the internet-connected devices we use to perform the processes and services that support our way of life. Another component set to help IoT succeed is cloud computing, which acts as a sort of front end. Cloud computing is an increasingly popular service that offers several advantages to IoT, and is based on the concept of allowing users to perform normal computing tasks using services delivered entirely over the internet.

Iot in cloud offers public cloud services can easily help the IoT area, by providing third party access to the infrastructure. Hence, the integration can help IoT data or computational components operating over IoT devices.

IoT devices need a lot of storage to share information for valuable purposes. Iot in cloud, like the StoneFly Cloud Connect to Microsoft Azure or we will use ThinkSpeak a free cloud service for IoT for student projects can provide customers with greater space which can increase as per the users demand. Helping to resolve the storage needs of customers.

The large amounts of data produced by IoT devices need extreme performance to interact and connect with one another. Iot in cloud provides the connectivity which is necessary to share information between the devices and make meaning from it at a fast pace.

Internet Cloud Computing infrastructures help IoT to give meaning to the greater amount of data generated. Users have no worry of buying greater or less storage. They can easily scale the storage as the data generated increases and pay for the amount of storage they consume with Internet Cloud Computing.

2.2 IoT in Project

he aim of this work is design and implements a distributed system for aquaculture water quality care through remote monitoring of dissolved oxygen, pH and temperature. This work will contribute remote monitoring distributed system through what

is known as the Internet of Things to monitoring water quality in ponds. The system is modular, portable, low cost, versatile and allows sharing information through the cloud that can be used for the development and improvement of aquaculture activities.

The system can be implemented in aquaculture farms to be able to monitor in real time the most important physical-chemical variables of the water. With this having a faster response with respect to what actions to take when conditions arise in the water quality of the ponds

A monitoring system for water quality in aquaculture ponds is mentioned in the publication A Mobile Platform for Remote Monitoring of Water Quality. Mobile sensor platform for monitoring ponds. This system consists of the following architecture. It has the sensing node of each pond connected to a sink; this sink sends the information to a mobile application to have a visualization of the data in real time. This information is transmitted via GSM / 3G to the Internet, it can be monitored remotely and the information is stored in a database. In the results the data of the ponds were shown remotely and the measures were corroborated by the transport staff.

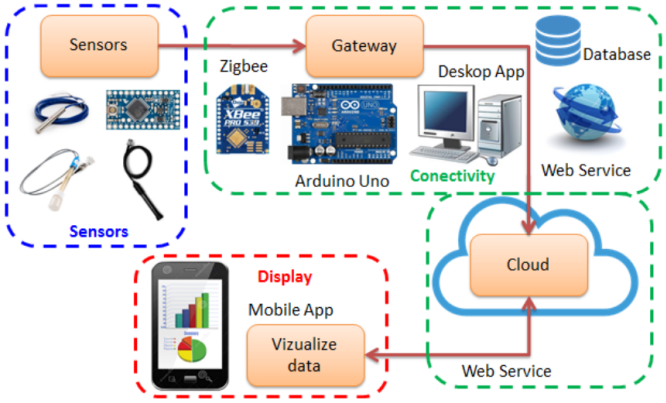
2.3 Work Flow or Connectivity

Here is the diagram which will help us to understand our work flow. Here is the images of all the sensors we will use.

pH Sensor : -

Temperature Sensor : -

Work Flow Diagram :-

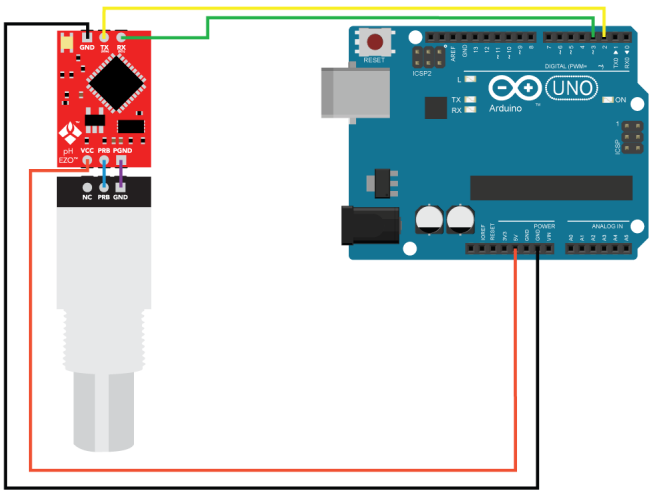


2.3.1 Collect Data From Sensor

In this section we will talk about first task of our workflow collect data from sensor and log it to console of arduino UNO. later we will send it to Cloud (ThinkSpeak for entire Project).

Below is the code of Collecting ph sensor and its connection and console it to COM Port console.

Figure : -



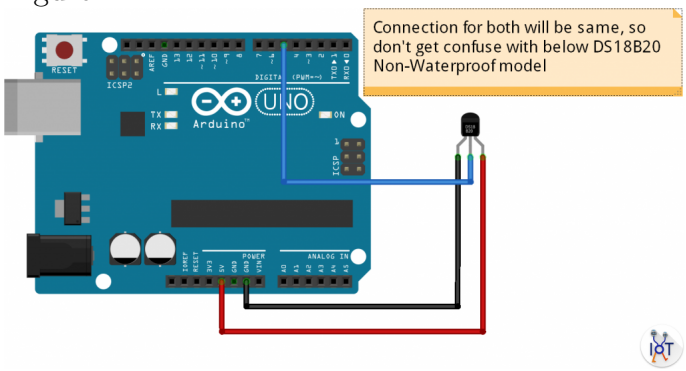
Listing 2.1: Arduino code for Ph sensor data

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
float calibration_value = 21.34;
int phval = 0;
unsigned long int avgval;
int buffer_arr[10],temp;
void setup()
{
 Serial.begin(9600);
  lcd.init();
  lcd.begin(16, 2);
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("    Welcome to      ");
  lcd.setCursor(0, 1);
  lcd.print(" Circuit Digest     ");
  delay(2000);
  lcd.clear();
}
void loop() {
 for(int i=0;i<10;i++)
 {
 buffer_arr[i]=analogRead(A0);
 delay(30);
 }
 for(int i=0;i<9;i++)
 {
 for(int j=i+1;j<10;j++)
 {
 if(buffer_arr[i]>buffer_arr[j])
 {
```

```
33   temp=buffer_arr[i];
34   buffer_arr[i]=buffer_arr[j];
35   buffer_arr[j]=temp;
36   }
37   }
38   }
39   avgval=0;
40   for(int i=2;i<8;i++)
41   avgval+=buffer_arr[i];
42   float volt=(float)avgval*5.0/1024/6;
43   float ph_act = -5.70 * volt + calibration_value;
44   lcd.setCursor(0, 0);
45   lcd.print("pH Val:");
46   lcd.setCursor(8, 0);
47   lcd.print(ph_act);
48   delay(1000);
49 }
```

Below is the code of Collecting temp. sensor and its connection and console it to COM Port console.

Figure : -



Listing 2.2: Arduino code for temp. sensor data

```
1  /********************************************************************/
2  // First we include the libraries
3  #include <OneWire.h>
4  #include <DallasTemperature.h>
5  /********************************************************************/
6  // Data wire is plugged into pin 2 on the Arduino
7  #define ONE_WIRE_BUS 2
8  /********************************************************************/
9  // Setup a oneWire instance to communicate with any OneWire devices
10 // (not just Maxim/Dallas temperature ICs)
11 OneWire oneWire(ONE_WIRE_BUS);
12 /********************************************************************/
13 // Pass our oneWire reference to Dallas Temperature.
14 DallasTemperature sensors(&oneWire);
```

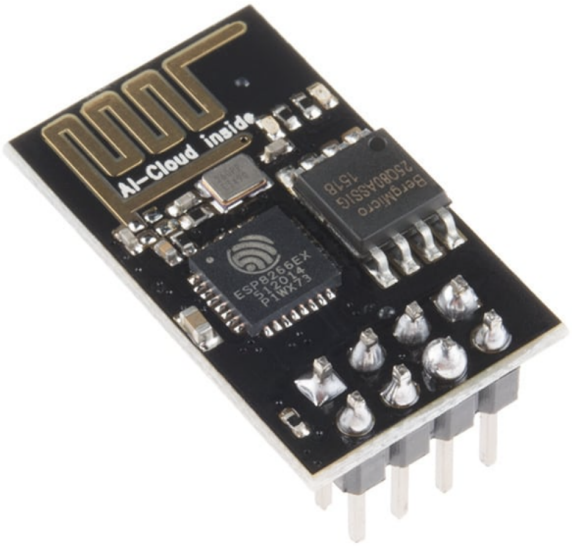
```
15 /*********************************************************************/
16 void setup(void)
17 {
18  // start serial port
19  Serial.begin(9600);
20  Serial.println("Dallas Temperature IC Control Library Demo");
21  // Start up the library
22  sensors.begin();
23 }
24 void loop(void)
25 {
26  // call sensors.requestTemperatures() to issue a global temperature
27  // request to all devices on the bus
28 /*********************************************************************/
29  Serial.print(" Requesting temperatures...");
30  sensors.requestTemperatures(); // Send the command to get temperature r
31  Serial.println("DONE");
32 /*********************************************************************/
33  Serial.print("Temperature is: ");
34  Serial.print(sensors.getTempCByIndex(0)); // Why "byIndex"?
35    // You can have more than one DS18B20 on the same bus.
36    // 0 refers to the first IC on the wire
37    delay(1000);
38 }
```

2.3.2 Send data to cloud (ThingsSpeak)

In this section we will make use of the data which we collected using both the sensor temp and ph and upload that data into the ThingsSpeak cloud which is cloud service offer by Matlab and it is free for student if you use it in limited conditions like minimum request to send data.

And from above section we know that the data from arduino is very fast reading like we get in every millisecond but we will upload data in 1 minute interval in cloud. to make minimum free usage for cloud.

Now Next thing which we will need to do to upload data in cloud is that we will need one wifi module which will connect to near by wifi hotspot and connect to cloud api to send data to channels in cloud. Below is the image of esp module device and code which make a connection of arduino and cloud



ESP8266

Listing 2.3: Arduino code for sending data to cloud

```
1
2  #include "ThingSpeak.h"
3  #include <ESP8266WiFi.h>
4
5  //------- WI-FI details ---------//
6  char ssid[] = "xxxxxxxxx"; //SSID here
7  char pass[] = "yyyyyyyyy"; // Passowrd here
8  //-------------------------------//
9
10 //---------- Channel details ----------//
11 unsigned long Channel_ID = 12345; // Your Channel ID
12 const char * myWriteAPIKey = "ABCDEF12345"; //Your write API key
13 //-------------------------------------//
14
15 const int Field_Number_1 = 1;
16 const int Field_Number_2 = 2;
17 String value = "";
18 int value_1 = 0, value_2 = 0;
19 int x, y;
20 WiFiClient  client;
21
22 void setup()
23 {
24   Serial.begin(115200);
25   WiFi.mode(WIFI_STA);
26   ThingSpeak.begin(client);
27   internet();
28 }
```

```
29
30 void loop()
31 {
32   internet();
33   if (Serial.available() > 0)
34   {
35     delay(100);
36     while (Serial.available() > 0)
37     {
38       value = Serial.readString();
39       if (value[0] == '*')
40       {
41         if (value[5] == '#')
42         {
43           value_1 = ((value[1] - 0x30) * 10 + (value[2] - 0x30));
44           value_2 = ((value[3] - 0x30) * 10 + (value[4] - 0x30));
45         }
46       }
47     }
48   }
49   upload();
50 }
51
52 void internet()
53 {
54   if (WiFi.status() != WL_CONNECTED)
55   {
56     while (WiFi.status() != WL_CONNECTED)
57     {
58       WiFi.begin(ssid, pass);
59       delay(5000);
60     }
61   }
62 }
63
64 void upload()
65 {
66   ThingSpeak.writeField(Channel_ID, Field_Number_1, value_1, myWriteAPI
67   delay(15000);
68   ThingSpeak.writeField(Channel_ID, Field_Number_2, value_2, myWriteAPI
69   delay(15000);
70   value = "";
71
72 }
```

2.3.3 Conclusion

Now all data that we will need to make real time prediction is uploaded in cloud now we have to fetch data from this cloud and implement in mobile app and web app but before that we have to train our tensorflow model to make good model which can give us best predictions from our data. So, in the next section we will use Tensorflow Numpy Pandas and scikit to make prediction.

Chapter 3

Machine Learning and Data Cleaning (Or Approach)

In this section we will talk about cleaning data using pandas and convert clean csv data into json using online csv to json converter and we will use sci-kit and tensorflow for two machine learning model Linear Regression and Artificial Neural Network in both Tensorflow and Sci-kit and later we will decide which model to use. In below subsection. first we will write little description of Data Cleaning and ML.

3.1 Data Cleaning

Data cleaning is the process of preparing data for analysis by removing or modifying data that is incorrect, incomplete, irrelevant, duplicated, or improperly formatted. This data is usually not necessary or helpful when it comes to analyzing data because it may hinder the process or provide inaccurate results. There are several methods for cleaning data depending on how it is stored along with the answers being sought. Data cleaning is not simply about erasing information to make space for new data, but rather finding a way to maximize a data set's accuracy without necessarily deleting information. For one, data cleaning includes more actions than removing data, such as fixing spelling and syntax errors, standardizing data sets, and correcting mistakes such as empty fields, missing codes, and identifying duplicate data points. Data cleaning is considered a foundational element of the data science basics, as it plays an important role in the analytical process and uncovering reliable answers. Most importantly, the goal of data cleaning is to create data sets that are standardized and uniform to allow business intelligence and data analytics tools to easily access and find the right data for each query.

3.1.1 Pandas and Numpy

For the purpose of data cleaning we will use Pandas and Numpy as it is very easy to handle data in pandas dataframes and Numpy lib in python is very good library to handle array and manipulate array using advanced array methods.

In this Project we have csv data of water with following properties

STATION CODE,LOCATIONS,STATE,Temp,D.O. (mg/l),PH,CONDUCTIVITY (µmhos/cm),B.O.D. (mg/l),NITRATENAN N+ NITRITENANN (mg/l),FECAL COLIFORM (MPN/100ml),TOTAL COLIFORM (MPN/100ml)Mean,year.

But the problem with this data is that the temp, ph and D.o is having some values which is not in scale and for example ph above 14 and and have value NAN in the temp, ph etc and the which we will need only temperature, ph and dissolved oxygen for our project so we will get all these clean data in new csv.

3.1.2 Machine Learning and Prediction

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. Machine learning focuses on the development of computer programs that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. The primary aim is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

Below is the code of Predictions using Our data using Tensorflow and Sci-kit Learn and data clearning in pandas also done in same code.

Listing 3.1: Linear regression Model Scikit

```
1
2  import numpy as np
3  import pandas as pd
4  import matplotlib.pyplot as plt
5
6  dataset = pd.read_csv("waterdata.csv", encoding= 'unicode_escape')
7
8  dataset.describe()
9
10 dataset["Temp"] = pd.to_numeric(dataset['Temp'], errors='coerce')
11 dataset["Temp"] = dataset["Temp"].replace(np.nan, 0)
12 dataset["PH"] = pd.to_numeric(dataset['PH'], errors='coerce')
13 dataset["PH"] = dataset["PH"].replace(np.nan, 0)
14 dataset["D.O. (mg/l)"] = pd.to_numeric(dataset['D.O. (mg/l)'], errors='
15 dataset["D.O. (mg/l)"] = dataset["D.O. (mg/l)"].replace(np.nan, 0)
16 # dataset["Temp"] = dataset["Temp"].fillna()
17
18 y = dataset["D.O. (mg/l)"]
19
20 X = dataset[["Temp", "PH"]]
21
22 # X["Temp"] = X["Temp"].mean()
```

```
23 # X["Temp"] = X[X["Temp"]==0].mean()
24
25 # X["Temp"].mean()
26 X["PH"].mean()
27
28 dataset=dataset.mask(dataset["Temp"]==0).fillna(dataset["Temp"].mean())
29 dataset=dataset.mask(dataset["PH"]==0).fillna(dataset["PH"].mean())
30 dataset=dataset.mask(dataset["D.O.  (mg/l)"]==0).fillna(dataset["D.O.  (mg
31
32 y = dataset["D.O.  (mg/l)"]
33 X = dataset[["Temp", "PH"]]
34 X["PH"]
35
36 from sklearn.model_selection import train_test_split
37
38 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3
39
40 from sklearn.linear_model import LinearRegression
41
42 lm = LinearRegression()
43
44 lm.fit(X_train, y_train)
45
46 lm.coef_
47
48 predictions = lm.predict(X_test)
49
50 # predictions
51 # 0.07  0.0000032
52
53 X_test.shape
54
55 data = {'longitude':  [30.6],
56         'latitude': [7.5]}
57
58 df = pd.DataFrame (data, columns = ['longitude','latitude'])
59
60 df.shape
61
62 answer = lm.predict(df)
63
64 answer
```

Listing 3.2: Linear regression Model Tensorflow API

```
1
2 # Commented out IPython magic to ensure Python compatibility.
3 import pandas as pd
4 import numpy as np
```

```
import matplotlib.pyplot as plt
import tensorflow as tf
plt.style.use("seaborn-colorblind")
# %matplotlib inline

used_features = ["Temp", "PH","D.O. (mg/l)"]
water = pd.read_csv('waterdata.csv', usecols = used_features, encoding=
# m = pd.read_csv('waterdata.csv', encoding= 'unicode_escape')
# target["D.O. (mg/l)"] = m["D.O. (mg/l)"]
print(water.shape)
water.head()
# target.head()

water["Temp"] = pd.to_numeric(water['Temp'], errors='coerce')
water["Temp"] = water["Temp"].replace(np.nan, 0)
water["PH"] = pd.to_numeric(water['PH'], errors='coerce')
water["PH"] = water["PH"].replace(np.nan, 0)
water["D.O. (mg/l)"] = pd.to_numeric(water["D.O. (mg/l)"], errors='coerc
water["D.O. (mg/l)"] = water["D.O. (mg/l)"].replace(np.nan, 0)

water=water.mask(water["Temp"]==0).fillna(water["Temp"].mean())
water=water.mask(water["PH"]==0).fillna(water["PH"].mean())
water=water.mask(water["D.O. (mg/l)"]==0).fillna(water["D.O. (mg/l)"].m

water

target = water["D.O. (mg/l)"]
features = water.drop('D.O. (mg/l)',axis=1)

features

target

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(
    water, target, test_size=0.33, random_state=42)

# numeric_columns = ["Temp", "PH","D.O. (mg/l)"]
numeric_columns = ["Temp", "PH"]
X_train.drop('D.O. (mg/l)',axis=1, inplace=True)
X_test.drop('D.O. (mg/l)',axis=1, inplace=True)
X_test

numeric_features = [tf.feature_column.numeric_column(key = column) for c
print(numeric_features[0])

linear_features = numeric_features

```

```
training_input_fn = tf.compat.v1.estimator.inputs.pandas_input_fn(x=X_t

eval_input_fn = tf.compat.v1.estimator.inputs.pandas_input_fn(x=X_test,

linear_regressor = tf.estimator.LinearRegressor(feature_columns=linear_f
                                                model_dir = "linear_regr

linear_regressor.train(input_fn = training_input_fn,steps=2000)

linear_regressor.evaluate(input_fn = eval_input_fn)

pred = list(linear_regressor.predict(input_fn = eval_input_fn))
pred = [p['predictions'][0] for p in pred]

prices = (pred)
print(prices)

X_test

y_test

predict_x = {
    'Temp': [30.1],
    'PH': [7.5],
}

def input_fn(features, batch_size=256):
    """An input function for prediction."""
    # Convert the inputs to a Dataset without labels.
    return tf.data.Dataset.from_tensor_slices(dict(features)).batch(10)

pred = linear_regressor.predict(
    input_fn=lambda: input_fn(predict_x))

pred

pred = [p['predictions'][0] for p in pred]

pred
```

Listing 3.3: ANN Model Tensorflow API

```


import pandas as pd
import numpy as np

used_features = ["Temp", "PH","D.O. (mg/l)"]
data = pd.read_csv("waterdata.csv", usecols = used_features, encoding=
```

```
8
9  data
10
11 data["Temp"] = pd.to_numeric(data['Temp'], errors='coerce')
12 data["Temp"] = data["Temp"].replace(np.nan, 0)
13 data["PH"] = pd.to_numeric(data['PH'], errors='coerce')
14 data["PH"] = data["PH"].replace(np.nan, 0)
15 data["D.O. (mg/l)"] = pd.to_numeric(data["D.O. (mg/l)"], errors='coerce
16 data["D.O. (mg/l)"] = data["D.O. (mg/l)"].replace(np.nan, 0)
17
18 data=data.mask(data["Temp"]==0).fillna(data["Temp"].mean())
19 data=data.mask(data["PH"]==0).fillna(data["PH"].mean())
20 data=data.mask(data["D.O. (mg/l)"]==0).fillna(data["D.O. (mg/l)"].mean()
21 target = data["D.O. (mg/l)"]
22
23 from sklearn.model_selection import train_test_split
24 X_train, X_test, y_train, y_test = train_test_split(
25       data, target, test_size=0.33, random_state=42)
26
27 X_train.head()
28
29 train_X = X_train.drop(columns=['D.O. (mg/l)'])
30 train_X.head()
31
32 y_train.shape
33
34 train_y = data[['D.O. (mg/l)']]
35 train_y.head()
36
37 import tensorflow as tf
38 from tensorflow import keras
39 from tensorflow.keras import layers
40 import tensorflowjs as tfjs
41
42 n_cols = train_X.shape[1]
43
44 model = keras.Sequential(
45     [
46         layers.Dense(10, activation="relu", name="layer1",input_shape=(
47         layers.Dense(3, activation="relu", name="layer2"),
48         layers.Dense(1, name="layer3"),
49     ]
50 )
51
52 model.compile(optimizer='adam', loss='mean_squared_error')
53
54 from tensorflow.keras.callbacks import EarlyStopping
55
```

```
56 early_stopping_monitor = EarlyStopping(patience=3)
57
58 model.fit(train_X, train_y, validation_split=0.2, epochs=30, callbacks=
59
60 tfjs_target_dir = "./tfjs"
61
62 tfjs.converters.save_keras_model(model, tfjs_target_dir)
63
64 test_X = X_test.drop(columns=['D.O. (mg/l)'])
65 test_X
66 # test_y_predictions = model.predict(X_test)
67
68 30.6,6.7,7.5
69
70 data = [[30.6, 7.5]]
71
72 # Create the pandas DataFrame
73 df = pd.DataFrame(data, columns = ['Temp', 'PH'])
74
75 # print dataframe.
76 df
77
78 # test_y_predictions = model.predict(test_X)
79 test_y_predictions = model.predict(df)
80
81 test_y_predictions
```

3.1.3 Tensorflow or Sci-kit Which is Best ?

When it's come to which model is best the answer is might complicated because which one is suitable for our App or website is the main reason to think about.

Sci-kit learn is a great library when it comes to simplicity and handling data sets on the other hand the tensorflow is slightly complicated because in tensorflow we have to deal with session to print and make calculation and also it deals with tensors so it is little bit difficult.

But Tensorflow is a great framework for machine learning enthusiast beacuse it provide vast variety of api which handle machine learning task easily. Tensorflow is easy can be used in mobile apps ans Website because of its great community and it is handled by google Below are the framework which are developed by tensorflow community

- Tensorflow (Main Python Framework Widely used)
- Tensorflow.js (Use in Javascript web and app and node application)
- Tensorflow Lite (For Mobile compatibility)

So, Tensorflow is a great option, as it provide great API which we will need in this project in Mobile App and in Web App.

In this chapter we finished all the data handling cleaning, trained the machine learning model and exported them to make inferences from them in real time arduino data. Now we have to include them in React and iOS and Andriod Application.

Chapter 4

API Implementation and Node.js

API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other. Each time you use an app like Facebook, send an instant message, or check the weather on your phone, you're using an API

4.1 Intutive Example (What is API ?)

Imagine you're sitting at a table in a restaurant with a menu of choices to order from. The kitchen is the part of the "system" that will prepare your order. What is missing is the critical link to communicate your order to the kitchen and deliver your food back to your table. That's where the waiter or API comes in. The waiter is the messenger – or API – that takes your request or order and tells the kitchen – the system – what to do. Then the waiter delivers the response back to you; in this case, it is the food.

Here is a real-life API example. You may be familiar with the process of searching flights online. Just like the restaurant, you have a variety of options to choose from, including different cities, departure and return dates, and more. Let us imagine that you're booking you are flight on an airline website. You choose a departure city and date, a return city and date, cabin class, as well as other variables. In order to book your flight, you interact with the airline's website to access their database and see if any seats are available on those dates and what the costs might be.

4.2 What is Node.js ?

Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009 and its latest version is v0.10.36. The definition of Node.js as supplied by its official documentation is as follows : -

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking

I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

4.2.1 Features of Node.js

Following are some of the important features that make Node.js the first choice of software architects.

- Asynchronous and Event Driven All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.
- Very Fast Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.
- Single Threaded but Highly Scalable Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.
- No Buffering Node.js applications never buffer any data. These applications simply output the data in chunks.
- License Node.js is released under the MIT license.

4.3 Tensorflow.js in Node

TensorFlow.js is a JavaScript Library for training and deploying machine learning models in the browser and in Node.js.

There are two module we need to get start with or to make api to make predictions. Two npm packages are below

4.3.1 @tensorflow/tfjs

TensorFlow.js is an open-source hardware-accelerated JavaScript library for training and deploying machine learning models.

- Develop ML in the Browser :- Use flexible and intuitive APIs to build models from scratch using the low-level JavaScript linear algebra library or the high-level layers API.
- Develop ML in Node.js :- Execute native TensorFlow with the same TensorFlow.js API under the Node.js runtime.

- Run Existing models :- Use TensorFlow.js model converters to run pre-existing TensorFlow models right in the browser.
- Retrain Existing models :- Retrain pre-existing ML models using sensor data connected to the browser or other client-side data.

4.3.2 @tensorflow/tfjs-node

TensorFlow.js is an open-source hardware-accelerated JavaScript library for training and deploying machine learning models.

This repository provides native TensorFlow execution in backend JavaScript applications under the Node.js runtime, accelerated by the TensorFlow C binary under the hood. It provides the same API as TensorFlow.js.

sectionLet's Make API
In this module we will use the above two mentioned npm module to make a node api which will use our exported model from chapter 2 and load it in api using Tensorflow.js

Below is the code which is used to make api to get predictions.

Listing 4.1: Node API to get tensorflow predictions

```
1  const app = require('express')()
2  let port = process.env.PORT || 3000;
3  const importData = require("./data.json")
4  const tfn = require("@tensorflow/tfjs-node")
5  const tf = require("@tensorflow/tfjs")
6  const handler = tfn.io.fileSystem("./model/model.json");
7  const bodyParser = require("body-parser")
8
9
10 app.use(bodyParser.json())
11 app.use(bodyParser.urlencoded({ extended: false }))
12
13
14 const model = async(temp, ph) => {
15     var data
16     const model = await tf.loadLayersModel(handler).then(m => {
17         var results = m.predict(tf.tensor2d([temp, ph], [1, 2]))
18         data = results.dataSync()[0]
19         // console.log(data)
20         return data
21     })
22     return data
23 }
24
25 app.get("/:temp/:ph", async (req, res) => {
26     var temp = req.params.temp
```

```
27      var ph = req.params.ph
28      var ans = await model(parseFloat(temp), parseFloat(ph))
29      res.json({ ans })
30  })
31
32
33  app.get("/players", (req, res) => {
34      res.send(importData)
35  })
36
37  app.listen(port, () => {
38      console.log("Server running onn port: ", port)
39  })
```

The API making is itself is the difficult part but what's the most difficult is to host this api on some platform so here comes Heroku which make is easy but hosting on it is also headache so let's get in

4.3.3 Host on Heroku

Heroku is a cloud platform that lets companies build, deliver, monitor and scale apps — we're the fastest way to go from idea to URL, bypassing all those infrastructure headaches.

- Create the App on Heroku :-

```
1       $ heroku create
2       Creating app... done,       fakestuff-farboo-84560
3       https://fakestuff-farboo-84560.herokuapp.com/ | https://git.hero
```

- Set the Node Server Configuration

```
1   $ heroku config:set NPM_CONFIG_PRODUCTION=false
```

- Listen to the Host 0.0.0.0

```
1   $ heroku config:set HOST=0.0.0.0
```

- Run Node in Production Mode

```
1   $ heroku config:set NODE_ENV=production
```

- Tell Heroku to Run "npm run build"

```
1   "scripts": {
2   "dev": "nuxt",
3   "build": "nuxt build",
4   "start": "nuxt start",
5   "generate": "nuxt generate",
```

```
6     "lint": "eslint --ext .js,.vue --ignore-path .gitignore .",
7     "heroku-postbuild": "npm run build"
8   }
```

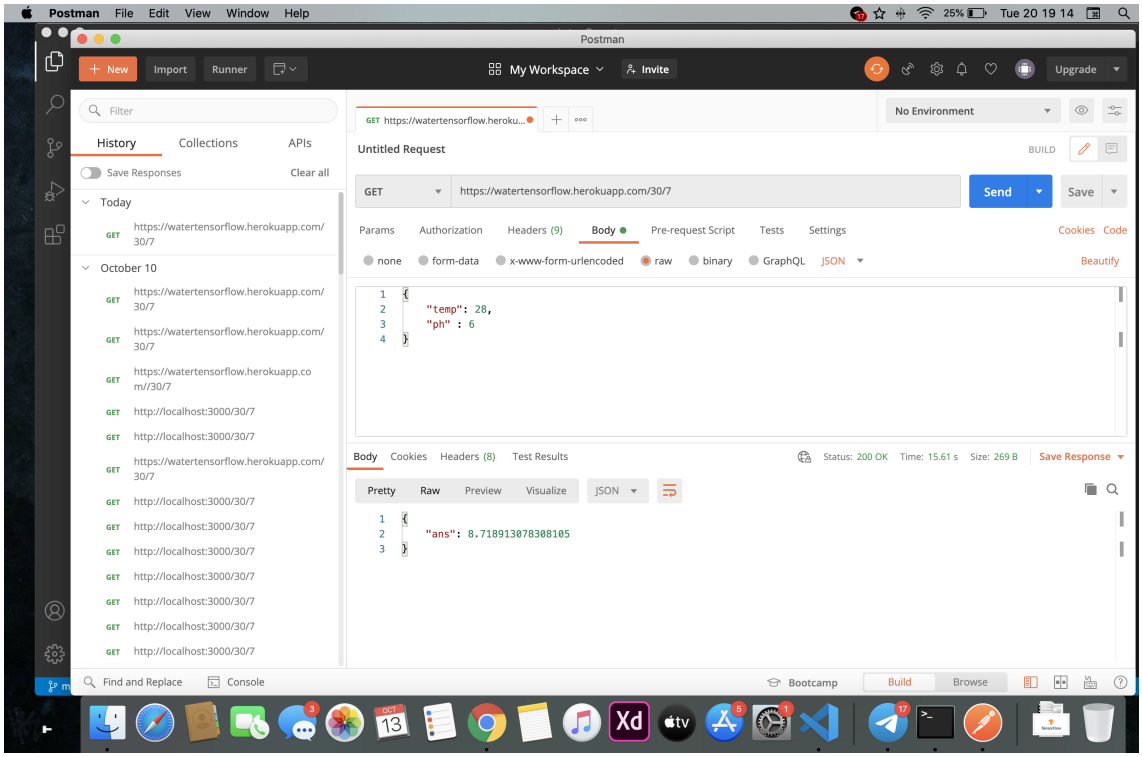
- Create a Procfile for Heroku

```
1       web: npm run start
```

- Push Your GitHub Repo to Heroku to Deploy

```
1       $ git add Procfile
2       $ git commit -a -m "Configuration to deploy to heroku"
3       $ git push heroku master
```

After all the thing done we have our website hosting done and now it't time to make a prediction using that API using postman so below is the picture of api which we hosted



As you can see from above image that for the temperature 28 and pH 6 we are getting Dissolved Oxygen amount 8.718913078308105 so here we successfully deployed a machine learning. Now its time to build this or include this API in Our Web App and Mobile App

Chapter 5

Web App and Mobile App

5.1 WebApp

A web application is a computer program that utilizes web browsers and web technology to perform tasks over the Internet.

Web applications use a combination of server-side scripts (PHP and ASP) to handle the storage and retrieval of the information, and client-side scripts (JavaScript and HTML) to present information to users. This allows users to interact with the company using online forms, content management systems, shopping carts and more. In addition, the applications allow employees to create documents, share information, collaborate on projects, and work on common documents regardless of location or device.

5.1.1 How it works?

Web applications are usually coded in browser-supported language such as JavaScript and HTML as these languages rely on the browser to render the program executable. Some of the applications are dynamic, requiring server-side processing. Others are completely static with no processing required at the server.

The web application requires a web server to manage requests from the client, an application server to perform the tasks requested, and, sometimes, a database to store the information. Application server technology ranges from ASP.NET, ASP and ColdFusion, to PHP and JSP.
Here's what a typical web application flow looks like:

- User triggers a request to the web server over the Internet, either through a web browser or the application's user interface
- Web server forwards this request to the appropriate web application server
- Web application server performs the requested task – such as querying the database or processing the data – then generates the results of the requested data

- Web application server sends results to the web server with the requested information or processed data
- Web server responds back to the client with the requested information that then appears on the user's display

5.1.2 Let's Create our App

The technology we will use to build out web app is React app it's very fast and very scalable web framework managed and developed by Facebook. and once it got loaded then everythin is stored and work easily.

Here is the final link of Web App : - `https://watermonitoring.netlify.app/`

To make web app more interactive and knowledgeable we include some best pond animation and Facts about water

Now Lets talk about Code We Will only talk about main code not about animation as it's very difficult to understand those animations so we will first directly jump into the Main code and in last animation code (optional*)

This is the main App.js Code which is starting point of our app

Listing 5.1: Water Monitoring App

```
1
2 import React from 'react';
3 import './App.css';
4 import * as tf from "@tensorflow/tfjs"
5 import Homepage from './Components/Homepage';
6
7 function App() {
8   return (
9     <div id="canvas-wrap">
10      <canvas id="canvas"></canvas>
11      <div id="overlay">
12        <Homepage/>
13      </div>
14    </div>
15  );
16 }
17 export default App;
```

Here is the main Homepage which include graphs between effect on Temperature and D.O and the effect of pH on D.O, Prediction input and facts

Listing 5.2: Homepage of ML app

```
1
```

```
2  import React, { useState } from 'react';
3  import * as tf from "@tensorflow/tfjs"
4  import { Scatter } from 'react-chartjs-2';
5  import d from "../data/water.json"
6  import facts from "../data/facts.json"
7
8
9  const modelUrl = "https://raw.githubusercontent.com/tarun-29/Water-Proje
10 const model = async (temp, ph, setAns) => {
11     console.log("hello babes")
12     return await tf.loadLayersModel(modelUrl).then(m => {
13         if (parseFloat(temp) && parseFloat(ph)) {
14             if (temp === 0 || ph === 0) {
15                 alert("Please enter valid values")
16                 return
17             }
18             else {
19                 var dat = [parseFloat(temp), parseFloat(ph)]
20                 var shap = [1, 2]
21                 var results = m.predict(tf.tensor2d(dat, shap));
22                 //  console.log(results.dataSync())
23                 Promise.resolve(results.dataSync()).then(s => {
24                     setAns(s)
25                 })
26             }
27         }
28         else{
29             alert("Enter numeric value")
30             return
31         }
32
33     })
34 }
35
36 const tempArray= []
37 tempArray.push(d.map(m=>{
38     var obj = {}
39     obj["y"] = m.DO
40     obj["x"] = m.Temp
41     return obj
42 }))
43
44 const pHArray= []
45 pHArray.push(d.map(m=>{
46     var obj = {}
47     obj["y"] = m.DO
48     obj["x"] = m.PH
49     return obj
```

```
50 }))
51
52 const options = {
53     legend: {
54         labels: {
55             fontColor: "white",
56             fontSize: 18
57         }
58     },
59     responsive: true,
60     title: {
61         display: true,
62         fontColor: "white",
63         fontSize: 15,
64     },
65     // tooltips: {
66     //     mode: 'label',
67     // },
68     hover: {
69         mode: 'nearest',
70         intersect: true
71     },
72     scales: {
73         xAxes: [{
74             ticks: {
75                 fontColor: "white",
76                 fontSize: 15,
77             },
78             display: true,
79             gridLines: {
80                 display: false,
81             },
82             scaleLabel: {
83                 display: true,
84                 labelString: 'Temperature',
85                 fontColor: "white",
86                 fontSize: 15
87             }
88         }],
89         yAxes: [{
90             ticks: {
91                 fontColor: "white",
92                 fontSize: 15,
93             },
94             display: true,
95             gridLines: {
96                 display: false,
97             },
```

```
98              scaleLabel: {
99                  display: true,
100                 labelString: 'D.O(mg/L)',
101                 fontColor: "white",
102                 fontSize: 15
103             }
104         }]
105     }
106 }
107
108 const options1 = {
109     legend: {
110         labels: {
111             fontColor: "white",
112             fontSize: 18
113         }
114     },
115     responsive: true,
116     title: {
117         display: true,
118         fontColor: "white",
119         fontSize: 15,
120     },
121     // tooltips: {
122     //     mode: 'label',
123     // },
124     hover: {
125         mode: 'nearest',
126         intersect: true
127     },
128     scales: {
129         xAxes: [{
130             ticks: {
131                 fontColor: "white",
132                 fontSize: 15,
133             },
134             display: true,
135             gridLines: {
136                 display: false,
137             },
138             scaleLabel: {
139                 display: true,
140                 labelString: 'PH',
141                 fontColor: "white",
142                 fontSize: 15
143             }
144         }],
145         yAxes: [{
```

```
146             ticks: {
147                 fontColor: "white",
148                 fontSize: 15,
149             },
150             display: true,
151             gridLines: {
152                 display: false,
153             },
154             scaleLabel: {
155                 display: true,
156                 labelString: 'D.O(mg/L)',
157                 fontColor: "white",
158                 fontSize: 15
159             }
160         }]
161     }
162 }
163
164
165 const data1 = tempArray[0]
166 const data2 = pHArray[0]
167
168 const Temp = {
169     labels: ['Scatter'],
170     datasets: [
171         {
172             label: 'Temp vs D.O(mg/L)',
173             fill: false,
174             backgroundColor: 'rgba(75,192,192,0.4)',
175             pointBorderColor: 'yellow',
176             pointBackgroundColor: '#fff',
177             pointBorderWidth: 1,
178             pointHoverRadius: 5,
179             pointHoverBackgroundColor: 'rgba(75,192,192,1)',
180             pointHoverBorderColor: 'rgba(220,220,220,1)',
181             pointHoverBorderWidth: 2,
182             pointRadius: 2,
183             pointHitRadius: 10,
184             data: data1
185         }
186     ]
187 };
188
189 const ph = {
190     labels: ['Scatter'],
191     datasets: [
192         {
193             label: 'pH vs D.O(mg/L)',
```

```
194             fill: false,
195             backgroundColor: 'rgba(75,192,192,0.4)',
196             // pointBorderColor: 'rgba(75,192,192,1)',
197             pointBorderColor: 'yellow',
198             pointBackgroundColor: '#fff',
199             pointBorderWidth: 1,
200             pointHoverRadius: 5,
201             pointHoverBackgroundColor: 'rgba(75,192,192,1)',
202             pointHoverBorderColor: 'rgba(220,220,220,1)',
203             pointHoverBorderWidth: 2,
204             pointRadius: 2,
205             pointHitRadius: 10,
206             data: data2
207         }
208     ]
209 };
210
211 function Homepage() {
212     console.log(pHArray)
213     const [count, setCount] = useState(0);
214     const [temp, setTemp] = useState(0);
215     const [PH, setPH] = useState(0);
216     const [ans, setAns] = useState(0);
217     return (
218         <div style={{ color: "white", textAlign: 'center', fontSize: 25
219             <div style={{ display: "flex", flexDirection: 'row', justify
220                 <div>
221                     <div style={{ display: "flex", flexDirection: 'colum
222                         <div >
223                             <Scatter data={Temp} options={options} heigh
224                         </div>
225                         <div >
226                             <Scatter data={ph} options={options1} height
227                         </div>
228                     </div>
229                 </div>
230                 <div style={{ display: "flex", flexDirection: 'column',
231
232                     <div style={{ marginTop: 20 }} className="card-form"
233                         <form className="signup">
234                             <div className="form-title">Predictions of I
235                             <div className="form-body">
236                                 <div className="row">
237                                     <input onChange={(e) => setTemp(e.ta
238                                 </div>
239                                 <div className="row">
240                                     <input onChange={(e) => setPH(e.targ
241                                 </div>
```

```
242                                 </div>
243                                 <div className="rule"></div>
244                                 <div className="form-footer" style={{ displa
245                                     <a href="/#" onClick={async () => { con
246                                     <div style={{ color: 'black' }}>{parseF
247                                 </div>
248                             </form>
249                         </div>
250                         {(facts.length <= 100) ? (<div className="card">
251                             <div id="circle"></div>
252                             <h2>Facts</h2>
253                             <p>{facts[count].Fact}</p>
254                             <div className="content">
255                                 <a onClick={(e) => setCount(count + ((Math.
256                             </div>
257                         </div>) : (<div>No Fact</div>)}
258                     </div>
259                 </div>
260             </div>
261         );
262 }
263
264 export default Homepage;
```

Listing 5.3: Background animation

```
1  (function () {
2        'use strict';
3        window.addEventListener('load', function () {
4          var canvas = document.getElementById('canvas');
5
6          if (!canvas || !canvas.getContext) {
7            return false;
8          }
9
10         /********************
11           Random Number
12         ********************/
13
14         function rand(min, max) {
15           return Math.floor(Math.random() * (max - min + 1) + min);
16         }
17
18         /********************
19           Var
20         ********************/
21
22         // canvas
23         var ctx = canvas.getContext('2d');
```

```
24          var X = canvas.width = window.innerWidth;
25          var Y = canvas.height = window.innerHeight;
26          var mouseX = null;
27          var mouseY = null;
28
29          /********************
30            Animation
31          ********************/
32
33          window.requestAnimationFrame =
34            window.requestAnimationFrame ||
35            window.mozRequestAnimationFrame ||
36            window.webkitRequestAnimationFrame ||
37            window.msRequestAnimationFrame ||
38            function (cb) {
39              setTimeout(cb, 17);
40            };
41
42          /********************
43            Wave
44          ********************/
45          var waves = [];
46
47          function Wave(ctx, x, y, r) {
48            this.ctx = ctx;
49            this.init(x, y, r);
50          }
51
52          Wave.prototype.init = function (x, y, r) {
53            this.x = x;
54            this.y = y;
55            this.r = r;
56            this.l = rand(100, 150);
57          };
58
59          Wave.prototype.draw = function () {
60            ctx = this.ctx;
61            ctx.save();
62            ctx.beginPath();
63            ctx.strokeStyle = 'rgb(149, 188, 249)';
64            ctx.arc(this.x, this.y, this.r, 0, Math.PI * 2, false);
65            ctx.stroke();
66            ctx.restore();
67          };
68
69          Wave.prototype.updateParams = function () {
70            this.r += 1;
71          };
```

```
72
73          Wave.prototype.deleteWave = function (i) {
74            if (this.r > this.l) {
75              waves.splice(i, 1);
76            }
77          };
78
79          Wave.prototype.render = function (i) {
80            this.updateParams();
81            this.deleteWave(i);
82            this.draw();
83          };
84
85          /********************
86            Fish
87          ********************/
88
89          var fishes = [];
90          var fishDir = [true, false];
91          var fishColors = ['255, 111, 147', '49, 194, 243', '255, 158, 0
92
93          function Fish(ctx, x, y, r, d, c) {
94            this.ctx = ctx;
95            this.init(x, y, r, d, c);
96          }
97
98          Fish.prototype.init = function (x, y, r, d, c) {
99            this.d = d;
100           this.x = x;
101           this.y = y;
102           this.r = r;
103           this.c = c;
104           this.rad = this.a * Math.PI / 180;
105           if (this.d === true) {
106             this.v = {
107               x: rand(1, 2) * 0.5,
108               y: rand(-1, 1) * 0.5
109             };
110           } else {
111             this.v = {
112               x: rand(-2, -1) * 0.5,
113               y: rand(-1, 1) * 0.5
114             };
115           }
116         };
117
118         Fish.prototype.draw = function () {
119           ctx = this.ctx;
```

```
120           ctx.save();
121           ctx.beginPath();
122           ctx.fillStyle = 'rgb(' + this.c + ')';
123           ctx.scale(2, 1);
124           ctx.arc(this.x / 2, this.y, this.r, 0, Math.PI * 2, false);
125           ctx.fill();
126           ctx.beginPath();
127           if (this.d === true) {
128             ctx.moveTo(this.x / 2 + this.r / 2, this.y);
129             ctx.lineTo(this.x / 2 + this.r + this.r / 2, this.y + this.r
130             ctx.lineTo(this.x / 2 + this.r + this.r / 2, this.y - this.r
131           } else {
132             ctx.moveTo(this.x / 2 - this.r / 2, this.y);
133             ctx.lineTo(this.x / 2 - this.r - this.r / 2, this.y + this.r
134             ctx.lineTo(this.x / 2 - this.r - this.r / 2, this.y - this.r
135           }
136           ctx.fill();
137           ctx.restore();
138         };
139
140         Fish.prototype.updatePosition = function () {
141           this.x -= this.v.x;
142           this.y += this.v.y;
143         };
144
145         Fish.prototype.wrapPosition = function () {
146           if (this.x + this.r + this.r > X) {
147             this.v.x *= -1;
148             this.d = true;
149           }
150           if (this.x - this.r - this.r < 0) {
151             this.v.x *= -1;
152             this.d = false;
153           }
154           if (this.y + this.r > Y) {
155             this.v.y *= -1;
156           }
157           if (this.y - this.r < 0) {
158             this.v.y *= -1;
159           }
160         };
161
162         Fish.prototype.resize = function () {
163           this.x = rand(0, X);
164           this.y = rand(0, Y);
165         };
166
167         Fish.prototype.render = function () {
```

```
168           this.updatePosition();
169           this.wrapPosition();
170           this.draw();
171         };
172
173         /********************
174           Grass
175         ********************/
176
177         // var
178         var grassNum = 200;
179         var grasses = [];
180
181         function Grass(ctx, x, y, w, t) {
182           this.ctx = ctx;
183           this.init(x, y, w, t);
184         }
185
186         Grass.prototype.init = function (x, y, w, t) {
187           this.x = x;
188           this.y = y;
189           this.w = w;
190           this.t = t;
191           this.a = 0;
192           this.rad = this.a * Math.PI / 180;
193           this.c = '255, 255, 255';
194           this.v = {
195             x: Math.cos(this.rad),
196             y: Math.sin(this.rad)
197           };
198           this.xt = this.x + this.w;
199           this.yt = this.y - this.t;
200           this.xb = this.x + this.w + this.w;
201         };
202
203         Grass.prototype.updateParams = function () {
204           this.a += Math.random();
205           this.rad = this.a * Math.PI / 180;
206           this.v.x = Math.cos(this.rad) * 0.3;
207           this.v.y = Math.sin(this.rad) * 0.3;
208         };
209
210         Grass.prototype.updatePosition = function () {
211           this.xt += this.v.x;
212         };
213
214         Grass.prototype.draw = function () {
215           ctx = this.ctx;
```

```
216           ctx.save();
217           ctx.fillStyle = 'rgb(86, 116, 25)';
218           ctx.beginPath();
219           ctx.moveTo(this.x, this.y);
220           ctx.lineTo(this.xt, this.yt);
221           ctx.lineTo(this.xb, this.y);
222           ctx.closePath();
223           ctx.fill();
224           ctx.restore();
225         };
226
227         Grass.prototype.resize = function () {
228           for (var i = 0; i < grassNum; i++) {
229             grasses[i].init(rand(-10, X + 10), Y, rand(2.5, 5), rand(100
230           }
231         };
232
233         Grass.prototype.render = function () {
234           this.updateParams();
235           this.updatePosition();
236           this.draw();
237         };
238
239         for (var i = 0; i < grassNum; i++) {
240           var grass = new Grass(ctx, rand(-10, X + 10), Y, rand(2.5, 5)
241           grasses.push(grass);
242         }
243
244         /********************
245           Bubble
246         ********************/
247
248         // var
249         var bubbleNum = 30;
250         var bubbles = [];
251
252         function Bubble(ctx, x, y, r) {
253           this.ctx = ctx;
254           this.init(x, y, r);
255         }
256
257         Bubble.prototype.init = function (x, y, r) {
258           this.x = x;
259           this.y = y;
260           this.r = r;
261           this.a = rand(1, 10);
262           this.dist = rand(1, 10);
263           this.rad = this.a * Math.PI / 180;
```

```
264             this.c = '255, 255, 255';
265             this.v = {
266               x: Math.sin(this.rad),
267               y: Math.cos(this.rad)
268             };
269           };
270
271           Bubble.prototype.updateParams = function () {
272             this.a += 1;
273             this.rad = this.a * Math.PI / 180;
274             this.y -= 1;
275           };
276
277           Bubble.prototype.wrapPosition = function () {
278             if (this.x - this.r > X) {
279               this.x = 0;
280             }
281             if (this.x + this.r < 0) {
282               this.x = X;
283             }
284             if (this.y - this.r > Y) {
285               this.y = 0;
286             }
287             if (this.y + this.r < 0) {
288               this.y = Y;
289             }
290           };
291
292           Bubble.prototype.draw = function () {
293             ctx = this.ctx;
294             ctx.save();
295             ctx.beginPath();
296             ctx.globalAlpha = 0.3;
297             ctx.fillStyle = 'rgb(255, 255, 255)';
298             ctx.arc(Math.cos(this.rad) * this.dist + this.x, Math.sin(this
299             ctx.fill();
300             ctx.closePath();
301             ctx.restore();
302           };
303
304           Bubble.prototype.resize = function () {
305             this.x = rand(0, X);
306             this.y = rand(0, Y);
307           };
308
309           Bubble.prototype.render = function () {
310             this.updateParams();
311             this.wrapPosition();
```

```
312           this.draw();
313         };
314
315         for (var i = 0; i < bubbleNum; i++) {
316           var bubble = new Bubble(ctx, rand(0, X), rand(0, Y), rand(1,
317           bubbles.push(bubble);
318         }
319
320         /********************
321           Render
322         ********************/
323
324         function render() {
325           ctx.clearRect(0, 0, X, Y);
326           for (var i = 0; i < grasses.length; i++) {
327             grasses[i].render();
328           }
329           for (var i = 0; i < bubbles.length; i++) {
330             bubbles[i].render();
331           }
332           for (var i = 0; i < fishes.length; i++) {
333             fishes[i].render();
334           }
335           for (var i = 0; i < waves.length; i++) {
336             waves[i].render(i);
337           }
338           requestAnimationFrame(render);
339         }
340
341         render();
342
343         /********************
344           Event
345         ********************/
346
347         // resize
348         function onResize() {
349           X = canvas.width = window.innerWidth;
350           Y = canvas.height = window.innerHeight;
351           for (var i = 0; i < grasses.length; i++) {
352             grasses[i].resize();
353           }
354           for (var i = 0; i < bubbles.length; i++) {
355             bubbles[i].resize();
356           }
357         }
358
359         window.addEventListener('resize', function () {
```

```
360             onResize();
361           });
362
363           canvas.addEventListener('click', function (e) {
364             mouseX = e.clientX;
365             mouseY = e.clientY;
366             var fish = new Fish(ctx, mouseX, mouseY, rand(5, 15), fishDir
367             fishes.push(fish);
368             var wave = new Wave(ctx, mouseX, mouseY, 0);
369             waves.push(wave);
370           }, false);
371
372         });
373       })();
```

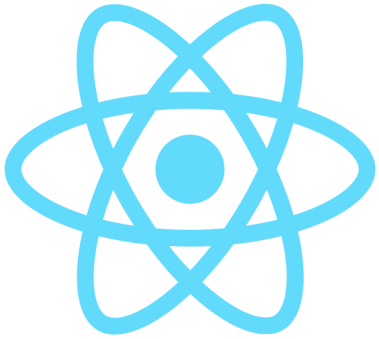
5.1.3 Chart.js

The chart which are plotted in website are done using the best node package manager graph.js and it's very usable the installation command of the library is shown below

Listing 5.4: Installation command graph.js

```
1 npm install chart.js --save
```

Here Our Web App portion is done one interesting point about this web app is that it only require Internet while loading the page once that's it when you make any prediction using this web app then it will predict using making any request to API or over cloud that's the main and very important point of this app. It's only needed to be load once and use anytime anywhere you want without internet(No Backend) (Condition you have to make data by Your self don't fetch from cloud)



5.2 Mobile App React Native

5.2.1 Introduction

- Create native apps for Android and iOS using React :- React Native combines the best parts of native development with React, a best-in-class JavaScript library for building user interfaces.
  Use a little—or a lot. You can use React Native today in your existing Android and iOS projects or you can create a whole new app from scratch.

- Written in JavaScript—rendered with native code :- React primitives render to native platform UI, meaning your app uses the same native platform APIs other apps do.
  Many platforms, one React. Create platform-specific versions of components so a single codebase can share code across platforms. With React Native, one team can maintain two platforms and share a common technology—React.

- Native Development For Everyone:- React Native lets you create truly native apps and doesn't compromise your users' experiences. It provides a core set of platform agnostic native components like View, Text, and Image that map directly to the platform's native UI building blocks.

- Seamless Cross-Platform :- React components wrap existing native code and interact with native APIs via React's declarative UI paradigm and JavaScript. This enables native app development for whole new teams of developers, and can let existing native teams work much faster.

5.2.2 Let's Jump to Code

Let's understand the code behind very simple yet complex machine learning app as the backend api are already completed in chapter 3 not it's time to use that api in react native app the react native app is very different from react app because in web app we didn't use the api to make prediction as tensorflow already included in web app but mobile application are light so we have to make calculation on cloud because it can't perform high computation in mobile phone. So let's get in

Listing 5.5: Mobile App Water Monitoring

```
1
2  import React, { useState } from 'react';
3  import { StyleSheet, Text, View, TouchableWithoutFeedback, Keyboard, Ima
4  import { Input, Button } from 'react-native-elements';
5  import axios from "axios"
6
7  const model = async (temp, ph, setAns) => {
8    console.log("temp, ph")
9    console.log(temp, ph)
10   var url = `https://watertensorflow.herokuapp.com/${temp}/${ph}`
11   axios.get(url).then(data => {
12     console.log(data.data.ans)
13     setAns(data.data.ans)
14   })
15 }
16
17 const fetchData = async (setTemp, setPh, setAns, t, p, setActivity) => {
18   setActivity(1)
19   const temp = "https://api.thingspeak.com/channels/1171829/fields/1.jso
20   const ph = "https://api.thingspeak.com/channels/1171829/fields/2.json?
21   var getTemp = await axios.get(temp)
22   var getPH = await axios.get(ph)
```

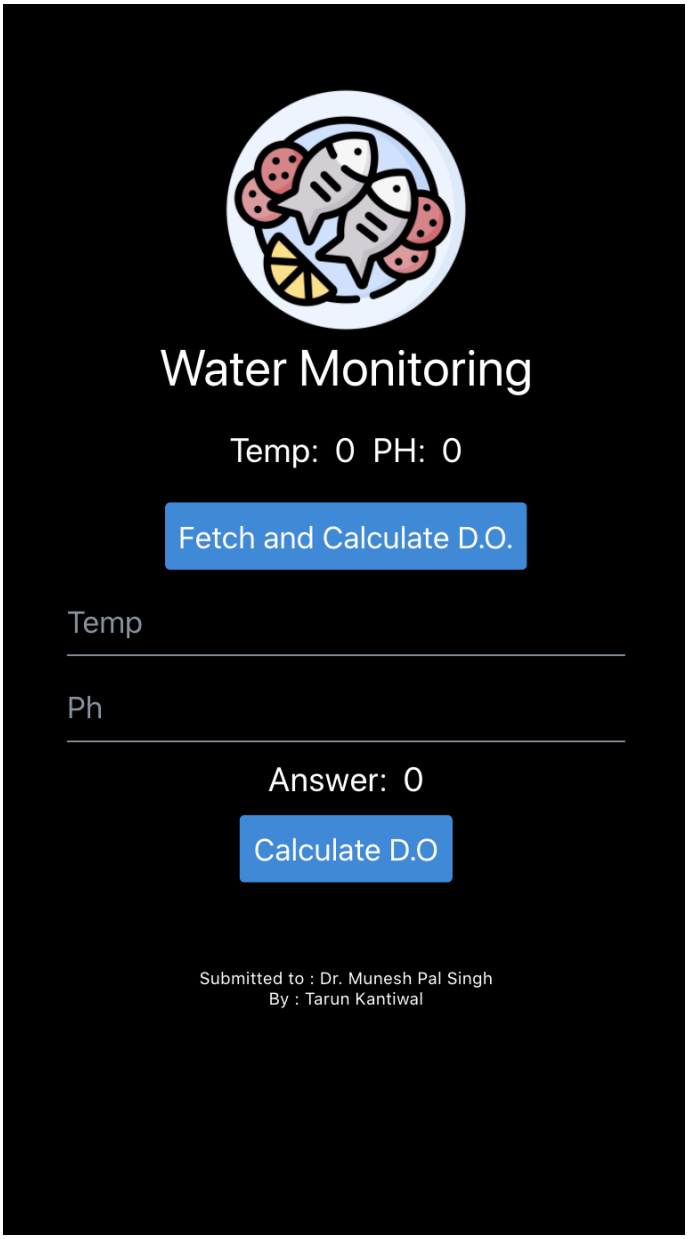
```
23    getTemp.data.feeds.forEach(data => {
24      if (data.field1 !== null) {
25        setTemp(data.field1)
26        t = data.field1
27      }
28    })
29    getPH.data.feeds.forEach(data => {
30      if (data.field2 !== null) {
31        setPh(data.field2)
32        p = data.field2
33      }
34    })
35    await model(t, p, setAns)
36    setActivity(0)
37  }
38
39  const staticData = async (t, p, setAns,setActivity) => {
40    setActivity(1)
41    console.log(typeof (t))
42    if (parseFloat(t) <= 0 || parseFloat(p) <= 0) {
43      alert("Please enter valid Temp and Ph")
44      setAns(0)
45      return
46    }
47    else {
48      setTimeout(()=>{}, 3000)
49      console.log("hello ji")
50      await model(t, p, setAns)
51      setActivity(0)
52    }
53  }
54
55  export default function App() {
56    const [temp, setTemp] = useState(0);
57    const [Ph, setPh] = useState(0);
58    const [tempStatic, setTempStatic] = useState(0);
59    const [PhStatic, setPhStatic] = useState(0);
60    const [ans, setAns] = useState(0);
61    const [activity, setActivity] = useState(0);
62    return (
63      activity === 0 ?
64        <TouchableWithoutFeedback onPress={Keyboard.dismiss} accessible={
65          <View style={styles.container}>
66            <Image
67              source={require("./assets/fish.png")}
68              style={{ height: 150, width: 150 }}
69            />
70            <Text style={{ fontSize: 30, color: 'white', paddingBottom: 20
```

```
71          <View style={{ display: 'flex', flexDirection: 'row', paddingB
72            <Text style={{ fontSize: 20, color: 'white', paddingRight: 1
    {temp}</Text>
73            <Text style={{ fontSize: 20, color: 'white' }}>PH:
    {Ph}</Text>
74          </View>
75          <Button
76            title="Fetch and Calculate D.O."
77            onPress={() => { fetchData(setTemp, setPh, setAns, temp, Ph,
78          />
79          <View style={{ width: 350, paddingTop: 10 }}>
80            <Input
81              placeholder="Temp"
82              onChangeText={value => setTempStatic(value)}
83              inputStyle={{ color: 'white' }}
84            />
85            <Input
86              placeholder="Ph"
87              inputStyle={{ color: 'white' }}
88              onChangeText={value => setPhStatic(value)}
89            />
90          </View>
91          <Text style={{ fontSize: 20, color: 'white', paddingBottom: 10
    {ans}</Text>
92          <Button
93            title="Calculate D.O"
94            onPress={() => { staticData(tempStatic, PhStatic, setAns, se
95          />
96          <View style={{ alignItems: 'center', paddingTop: 50 }}>
97            <Text style={{ fontSize: 10, color: 'white' }}>Submitted to
98            <Text style={{ fontSize: 10, color: 'white' }}>By : Tarun K
99          </View>
100       </View>
101     </TouchableWithoutFeedback >
102     :
103     <View style={{flex:1, alignItems:"center", justifyContent: 'center
104       <Text style={{color: 'whitesmoke', fontSize: 40, paddingBottom:
105       <Text style={{color: 'whitesmoke', paddingBottom: 50, fontSize:
106       <ActivityIndicator size="large" color="#00ff00"/>
107     </View>
108   );
109 }
110
111 const styles = StyleSheet.create({
112   container: {
113     flex: 1,
114     backgroundColor: 'black',
115     alignItems: 'center',
```

```
116       paddingTop: 50
117     },
118   });
```

Now we are ready our app is build Now the most difficult task is to manage version of expo and make apk out of it the most challanging task

React native app by default is cross platform native app for iOS and Andriod but we can't use these app in iOS because of security reason as Apple won't allow iPhone users to download app from external source, you can only use app which are downloaded from app store but we can use this app in android smart phone so i will attach the link to download this apk file and you can install it in your phone and use it. currently we didn't include any authentication as it is in testing mode. click Get the App Image of App below

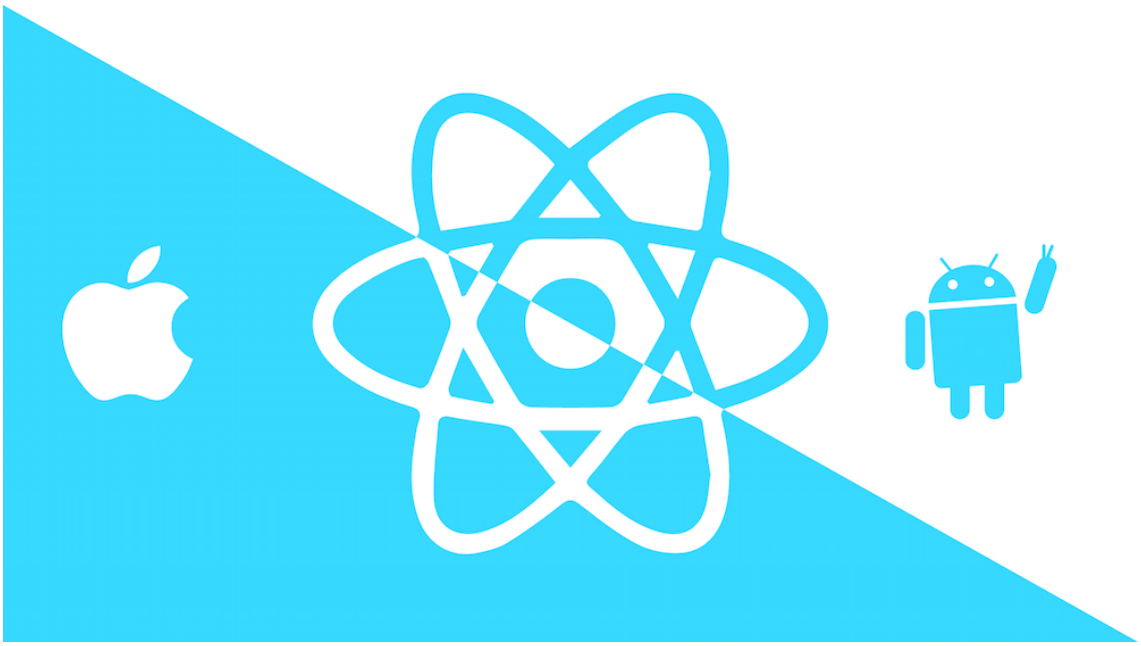


By this we completed the mobile Application which we can use in our mobile to

make prediction. Now we have a machine learning power on our palm.

5.3 What's next ?

Now We have both Mobile App and web App are ready to use we have to take care of testing and what are the things we have to now take care of, errors.



Chapter 6

Testing and Evaluation

By Far our App and web app are working parfectly but we need to take care of some error and make app development and web much more effecient

Below are the Point which we need to be rectify

6.1 Limited data

These App uses water data which is very limited available and the data which is used in this whole project is downloaded from kaggle and that water data has some drawbacks

- The Temperature data is in between normal temperature for example (23-30°C) and if we make prediction below or above these temperature then we for sure will not get good predictions so we need more data of wide variety of temperature.
- And another point is of pH as we know all the water bodies which are pond lake all are not too much acidic and too much basic so when you make prediction for pH like 1-4 or basic solution like 10-14 then it will surely give error the trained model is not train for these temperature and pH ranges so we have to collect more data, to make good predictions

6.2 Data Scaling and Best Model

As the title suggest Data scaling and good model we already using best model for our mobile and web app but as we all know there are chances to minimize test error in machine learning we can improve model and make error minimum but we can't make error go away so we can always do work to reduce error and make our model accuracy very good

And Data scaling says while training data we can scale input to make predictions good this process is also the subset of above paragraph but these are two different task but interlinked so we have to take care of it

Chapter 7

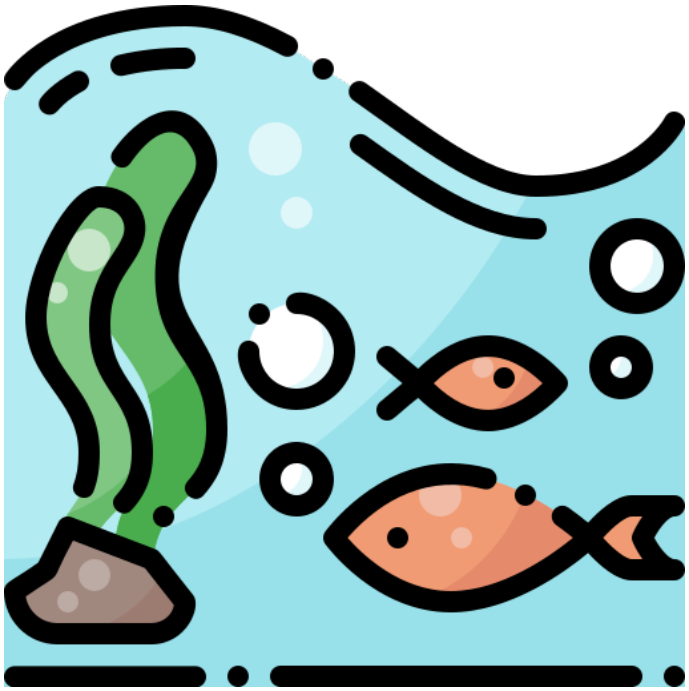
Conclusion

So here we are finishing all our work make a very vast machine learning Model and deployed them in Web App and iOS and android app and connected all the thing to cloud. So here are the things I acheive so far in this project

1. I have completely met my aim and solved the problem
2. The Solution solved the problem as given by Dr. Munesh pal Singh
3. The solution is although best for the certain most common temperature and pH range but for wide variety of data it is getting failed. so we have to rectify it

7.1 Future Work

As we reached the end of our project but there are the task in chapter 6 which we have to do in Future to make this project more successful and great



Github : - Github Tarun