Sri Lanka Institute of Information Technology

KidniFy - A Mobile based Chronic Kidney Disease Patient Care System Using ML and IoT

2023-032

STATUS DOCUMENT - I

Student Name: Perera J.P.M.L

Student ID: IT20226596

Group Details

Supervisor: Ms. Wishalya Vanshanee Tissera

Co-Supervisor: Mr. Samadhi Chathuranga Rathnayake

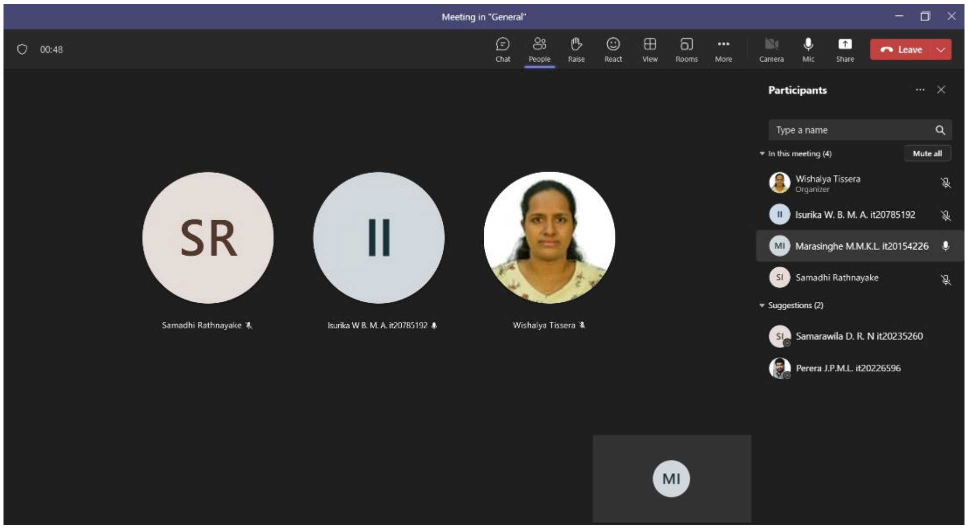
Student Name	Student ID	Contact No	Email Address
Marasinghe M.M.K.L.	IT20154226	0713037712	it20154226@my.sliit.lk
Isurika W.B.M.A.	IT20785192	0701484570	it20785192@my.sliit.lk
Perera J.P.M.L.	IT20226596	0776035479	it20226596@my.sliit.lk
Samarawila D.R.N.	IT20235260	0712421580	it20235260@my.sliit.lk

TABLE OF CONTENTS

1 Teams Meeting 3
1.1 Screenshots of Meetings & Calls 3
1.2 Meeting with the domain expert 4
2 Screenshots of the Tasks by Planner 6
2.1 Chart Overview 6
2.2 Bucket List 7
2.3 Screenshots of GitLab 8
3 Project Implementation 10
4 Gantt Chart 17
5 Work Breakdown Structure 18

1 Teams Meeting

1.1 Screenshots of Meetings & Calls



Meeting in "General"

00:48

Participants

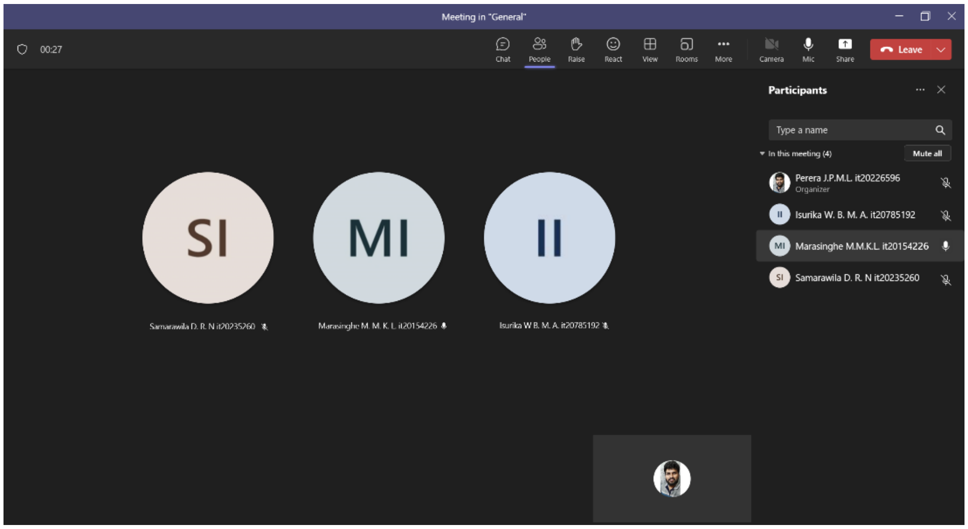
In this meeting (4)

- Wishalya Tissera, Organizer
- Isurika W. B. M. A. it20785192
- Marasinghe M.M.K.L. it20154226
- Samadhi Rathnayake

Suggestions (2)

- Samarawila D. R. N it20235260
- Perera J.P.M.L. it20226596

Samadhi Rathnayake · Isurika W B. M. A. it20785192 · Wishalya Tissera



Meeting in "General"

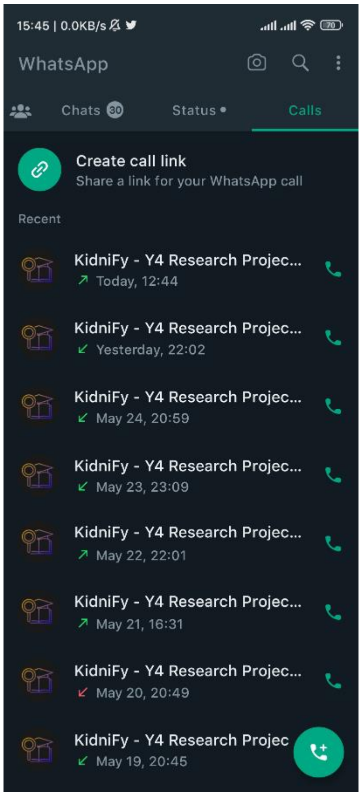
00:27

Participants

In this meeting (4)

- Perera J.P.M.L. it20226596, Organizer
- Isurika W. B. M. A. it20785192
- Marasinghe M.M.K.L. it20154226
- Samarawila D. R. N it20235260

Samarawila D. R. N it20235260 · Marasinghe M. M. K. L. it20154226 · Isurika W B. M. A. it20785192



Create call link
Share a link for your WhatsApp call

Recent

- KidniFy - Y4 Research Projec... ↗ Today, 12:44
- KidniFy - Y4 Research Projec... ↙ Yesterday, 22:02
- KidniFy - Y4 Research Projec... ↙ May 24, 20:59
- KidniFy - Y4 Research Projec... ↙ May 23, 23:09
- KidniFy - Y4 Research Projec... ↗ May 22, 22:01
- KidniFy - Y4 Research Projec... ↗ May 21, 16:31
- KidniFy - Y4 Research Projec... ↙ May 20, 20:49
- KidniFy - Y4 Research Projec ↙ May 19, 20:45

1.2 Meeting with the domain expert

Meeting with Consultant Nephrologist Dr. Pramil Rajakrishnan

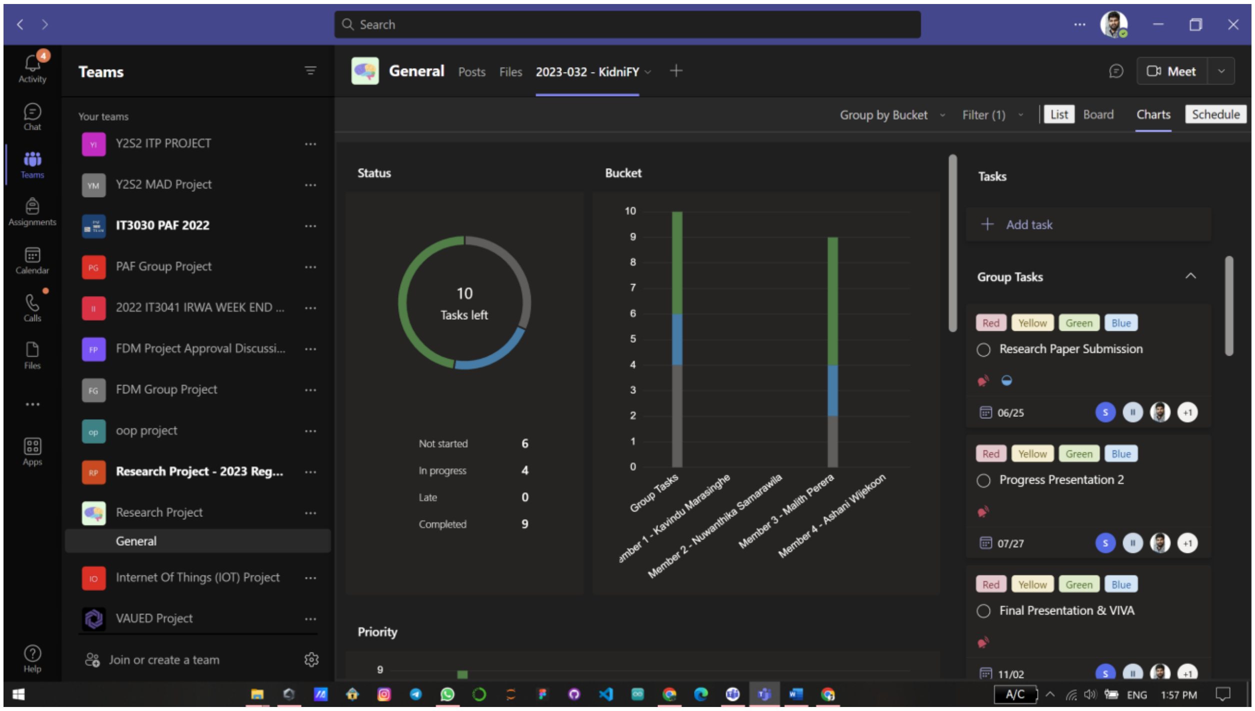
At Kurunegala Teaching Hospital - Kidney dialysis Unit | 2023.04.29

Meet the Ward Master of Kidney Dialysis Unit

At Anuradhapura Teaching Hospital | 2023.05.01

2 Screenshots of the Tasks by Planner

2.1 Chart Overview



10
Tasks left

Not started	6
In progress	4
Late	0
Completed	9

Group Tasks

Red Yellow Green Blue
○ Research Paper Submission
06/25

Red Yellow Green Blue
○ Progress Presentation 2
07/27

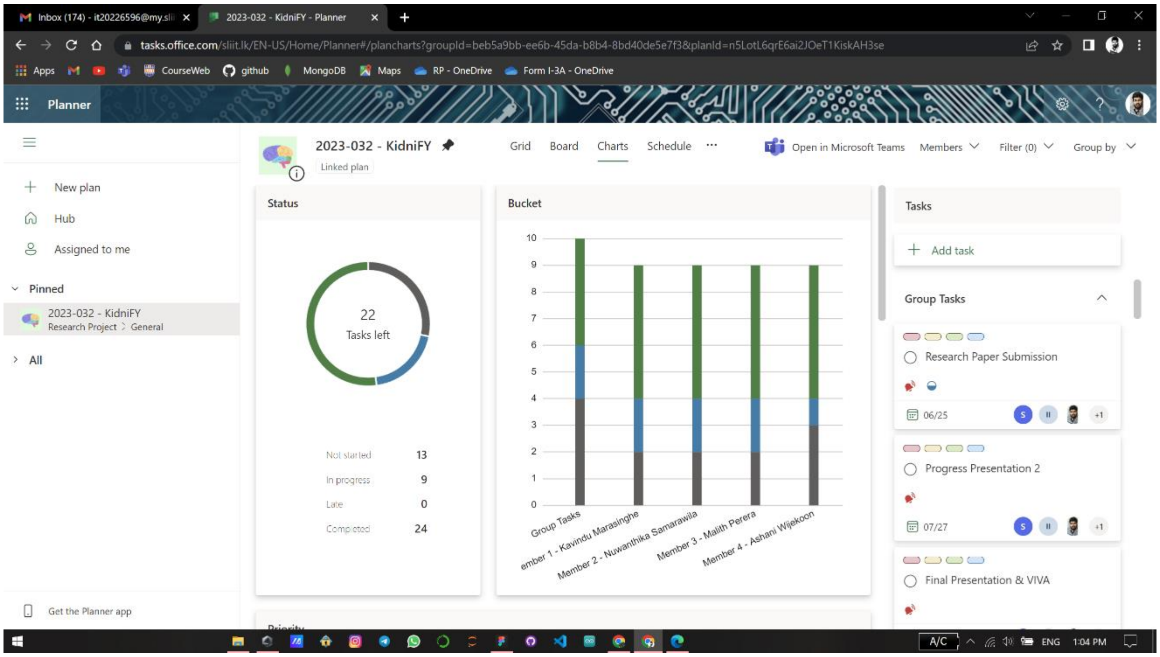
Red Yellow Green Blue
○ Final Presentation & VIVA
11/02

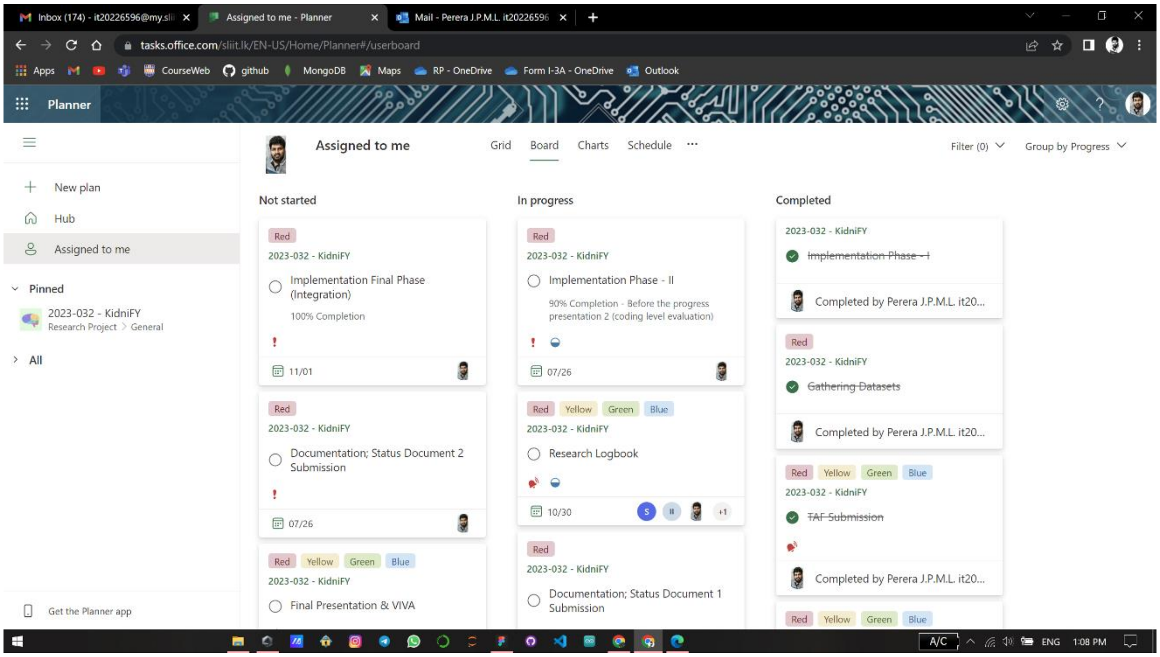
🟩 - To Do

🟦 - In Progress

⬛ - Completed

2.2 Bucket List





Not started

Red
2023-032 - KidniFY
Implementation Final Phase (Integration)
100% Completion
11/01

Red
2023-032 - KidniFY
Documentation; Status Document 2 Submission
07/26

Red Yellow Green Blue
2023-032 - KidniFY
Final Presentation & VIVA

In progress

Red
2023-032 - KidniFY
Implementation Phase - II
90% Completion - Before the progress presentation 2 (coding level evaluation)
07/26

Red Yellow Green Blue
2023-032 - KidniFY
Research Logbook
10/30

Red
2023-032 - KidniFY
Documentation; Status Document 1 Submission

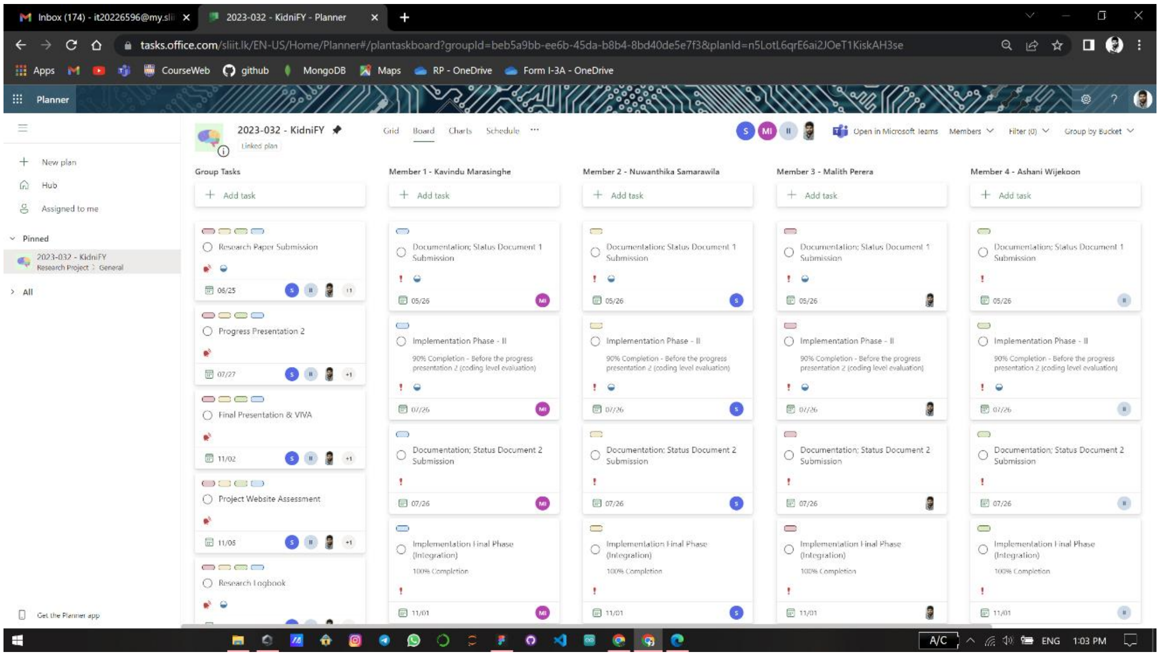
Completed

2023-032 - KidniFY
~~Implementation Phase - I~~
Completed by Perera J.P.M.L. it20...

Red
2023-032 - KidniFY
~~Gathering Datasets~~
Completed by Perera J.P.M.L. it20...

Red Yellow Green Blue
2023-032 - KidniFY
~~TAF Submission~~
Completed by Perera J.P.M.L. it20...

Red Yellow Green Blue



2023-032 - KidniFY

Group Tasks
- Research Paper Submission — 06/25
- Progress Presentation 2 — 07/27
- Final Presentation & VIVA — 11/02
- Project Website Assessment — 11/05
- Research Logbook

Member 1 - Kavindu Marasinghe
- Documentation; Status Document 1 Submission — 05/26
- Implementation Phase - II — 90% Completion - Before the progress presentation 2 (coding level evaluation) — 07/26
- Documentation; Status Document 2 Submission — 07/26
- Implementation Final Phase (Integration) — 100% Completion — 11/01

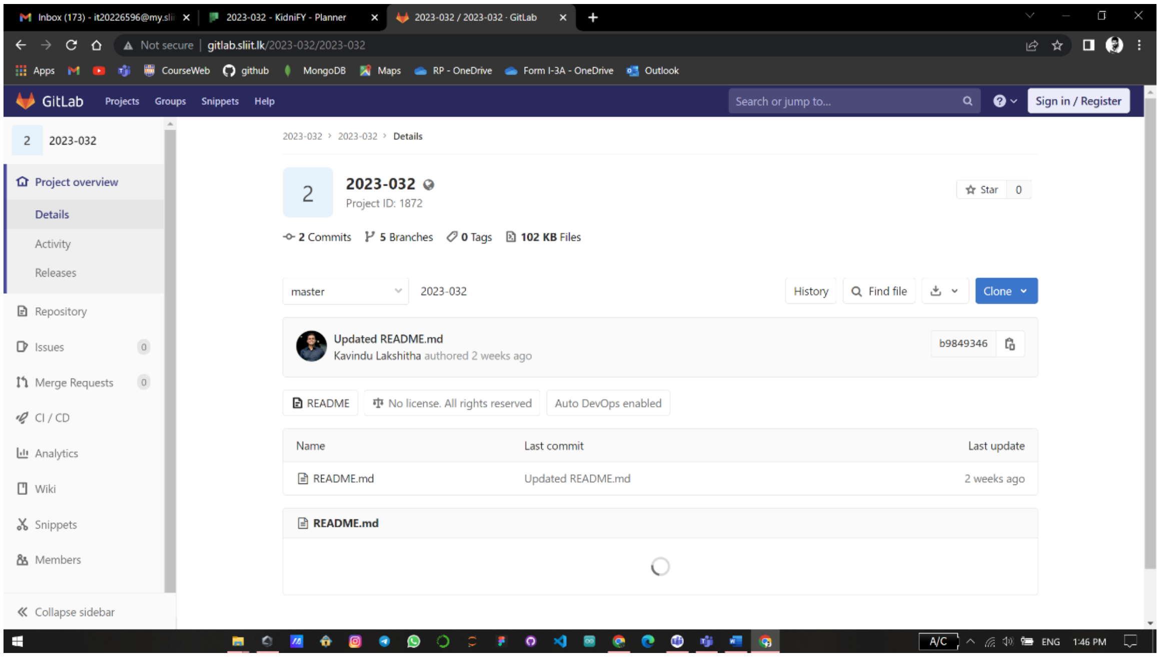
Member 2 - Nuwanthika Samarawila
- Documentation; Status Document 1 Submission — 05/26
- Implementation Phase - II — 90% Completion - Before the progress presentation 2 (coding level evaluation) — 07/26
- Documentation; Status Document 2 Submission — 07/26
- Implementation Final Phase (Integration) — 100% Completion — 11/01

Member 3 - Malith Perera
- Documentation; Status Document 1 Submission — 05/26
- Implementation Phase - II — 90% Completion - Before the progress presentation 2 (coding level evaluation) — 07/26
- Documentation; Status Document 2 Submission — 07/26
- Implementation Final Phase (Integration) — 100% Completion — 11/01

Member 4 - Ashani Wijekoon
- Documentation; Status Document 1 Submission — 05/26
- Implementation Phase - II — 90% Completion - Before the progress presentation 2 (coding level evaluation) — 07/26
- Documentation; Status Document 2 Submission — 07/26
- Implementation Final Phase (Integration) — 100% Completion — 11/01

2.3 Screenshots of GitLab

Repository in GitLab



2023-032
Project ID: 1872

2 Commits · 5 Branches · 0 Tags · 102 KB Files

master · 2023-032

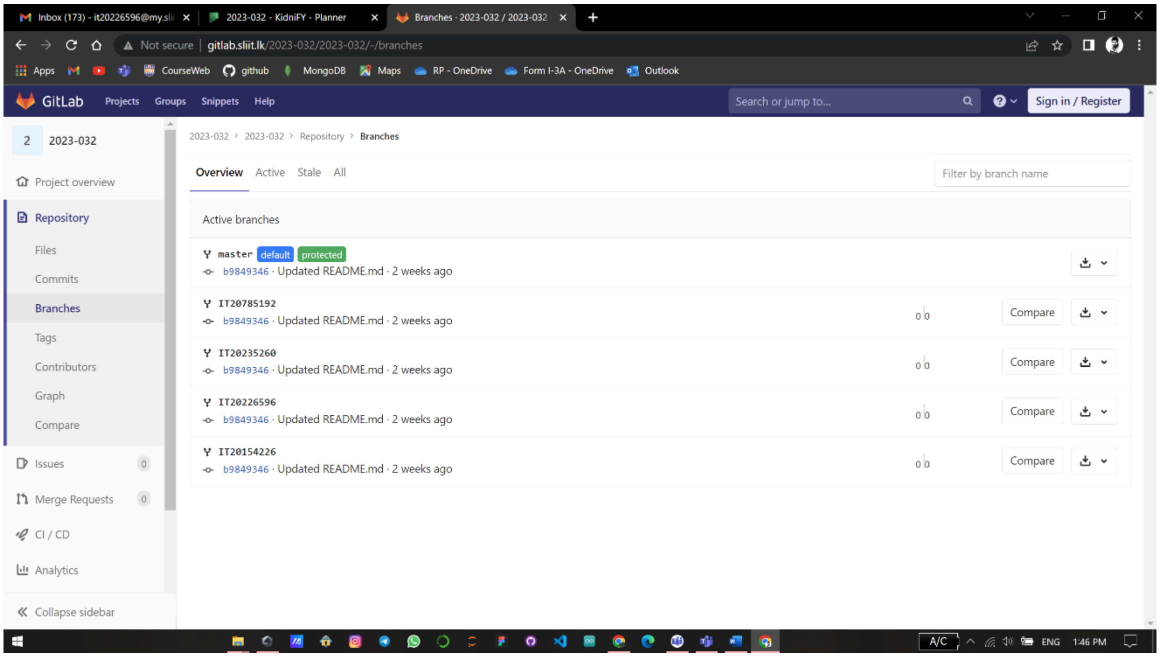
Updated README.md
Kavindu Lakshitha authored 2 weeks ago · b9849346

README · No license. All rights reserved · Auto DevOps enabled

Name	Last commit	Last update
README.md	Updated README.md	2 weeks ago

README.md

Create GitLab Branches & Starting Implementations



2023-032 › 2023-032 › Repository › Branches

Active branches

master default protected
b9849346 · Updated README.md · 2 weeks ago

IT20785192
b9849346 · Updated README.md · 2 weeks ago
0 0

IT20235260
b9849346 · Updated README.md · 2 weeks ago
0 0

IT20226596
b9849346 · Updated README.md · 2 weeks ago
0 0

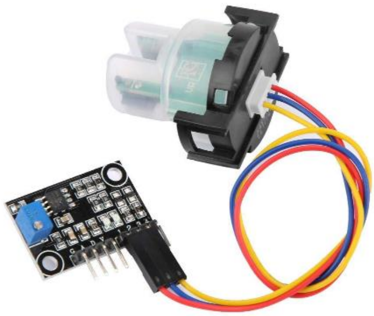
IT20154226
b9849346 · Updated README.md · 2 weeks ago
0 0

3 Project Implementation

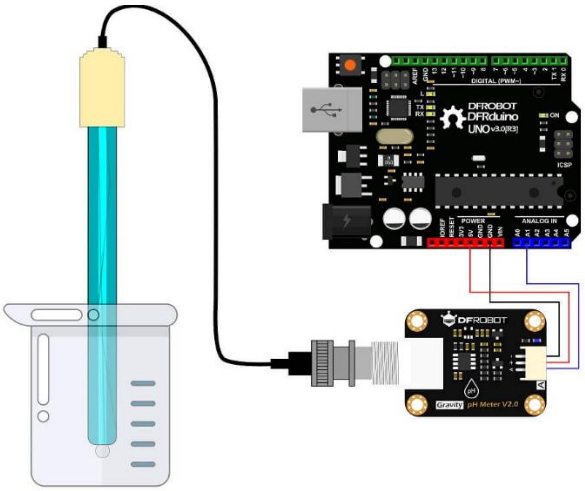
Water Quality Monitoring System for Kidney Patients based on IoT and Machine Learning

Gathering IoT Components

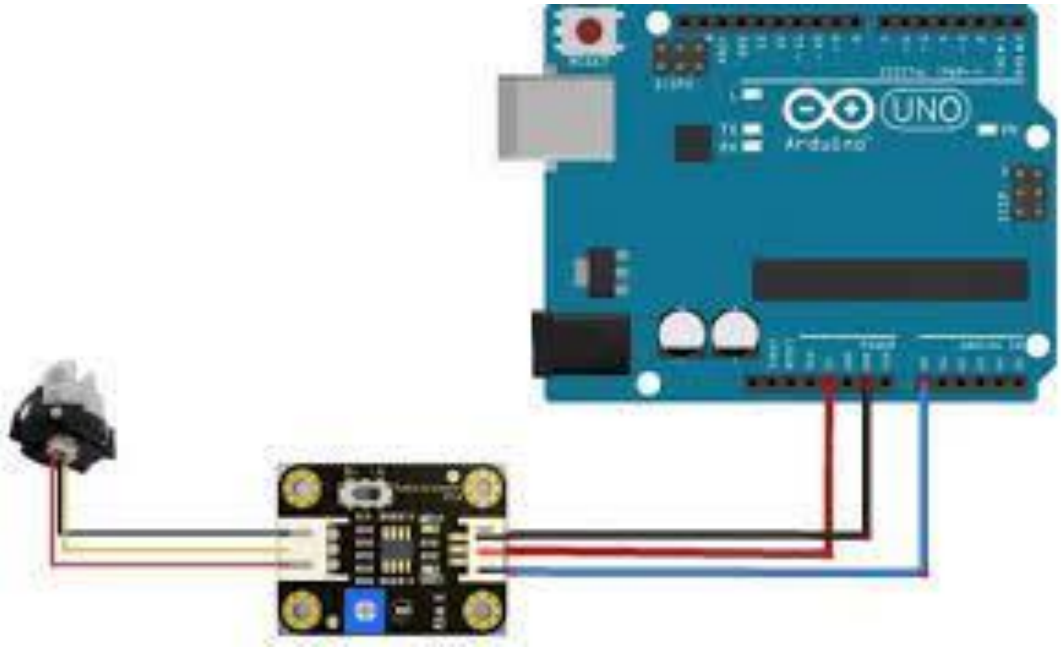
(Arduino Uno Board, Jump wires, pH Sensor , Turbidity Sensor & Temperature Sensor)



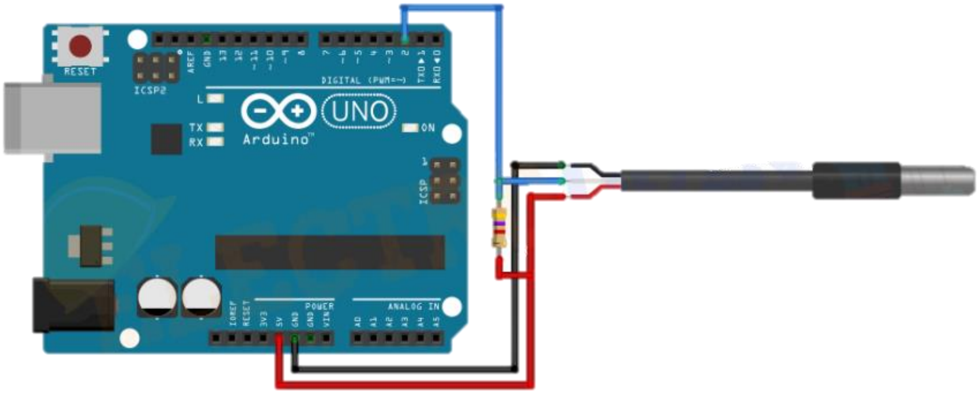
Circuit Diagram for testing the pH Sensor.



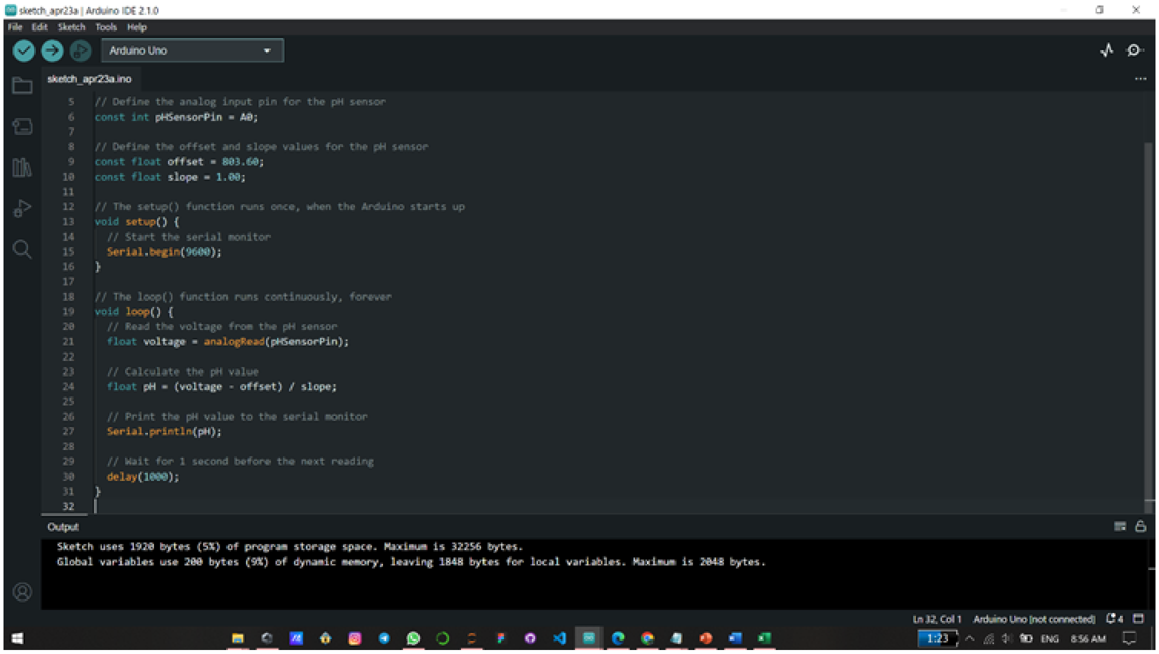
Circuit Diagram for testing the Turbidity Sensor.



Circuit Diagram for testing the Temperature Sensor.



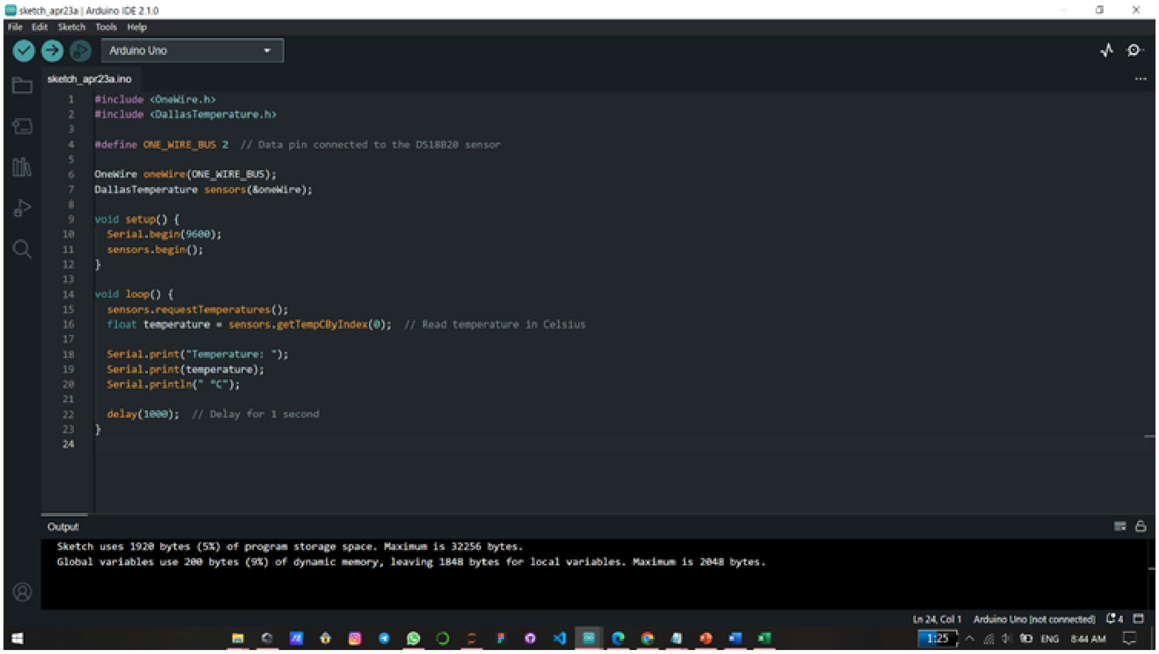
Code for testing the pH sensor.


sketch_apr23a | Arduino IDE 2.1.0
File Edit Sketch Tools Help
Arduino Uno
sketch_apr23a.ino
Output
Ln 32, Col 1 Arduino Uno [not connected] 4
1:23
ENG 8:56 AM

```
// Define the analog input pin for the pH sensor
const int pHSensorPin = A0;

// Define the offset and slope values for the pH sensor
const float offset = 803.60;
const float slope = 1.00;

// The setup() function runs once, when the Arduino starts up
void setup() {
  // Start the serial monitor
  Serial.begin(9600);
}

// The loop() function runs continuously, forever
void loop() {
  // Read the voltage from the pH sensor
  float voltage = analogRead(pHSensorPin);

  // Calculate the pH value
  float pH = (voltage - offset) / slope;

  // Print the pH value to the serial monitor
  Serial.println(pH);

  // Wait for 1 second before the next reading
  delay(1000);
}

```

```
Sketch uses 1920 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 200 bytes (9%) of dynamic memory, leaving 1848 bytes for local variables. Maximum is 2048 bytes.
```

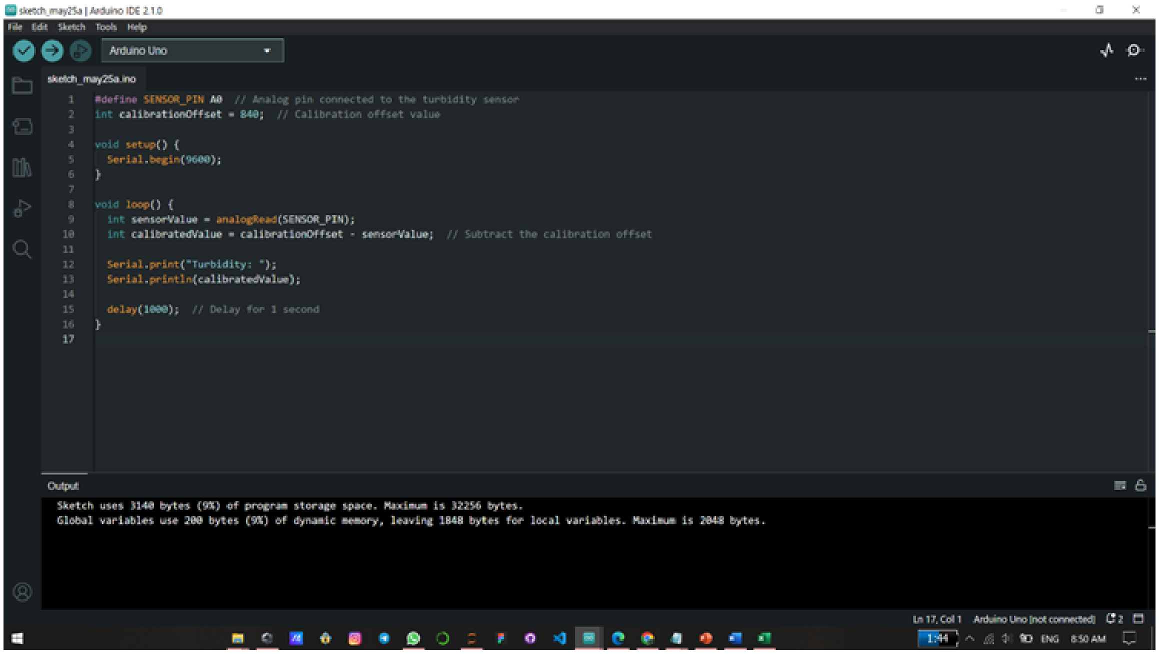
Code for testing the Temperature sensor.


sketch_apr23a | Arduino IDE 2.1.0
File Edit Sketch Tools Help
Arduino Uno
sketch_apr23a.ino
Output
Ln 24, Col 1 Arduino Uno [not connected] 4
1:25
ENG 8:44 AM

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

#define ONE_WIRE_BUS 2  // Data pin connected to the DS18B20 sensor

OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

void setup() {
  Serial.begin(9600);
  sensors.begin();
}

void loop() {
  sensors.requestTemperatures();
  float temperature = sensors.getTempCByIndex(0);  // Read temperature in Celsius

  Serial.print("Temperature: ");
  Serial.print(temperature);
  Serial.println(" °C");

  delay(1000);  // Delay for 1 second
}

```

```
Sketch uses 1920 bytes (5%) of program storage space. Maximum is 32256 bytes.
Global variables use 200 bytes (9%) of dynamic memory, leaving 1848 bytes for local variables. Maximum is 2048 bytes.
```

Code for testing the Turbidity sensor.

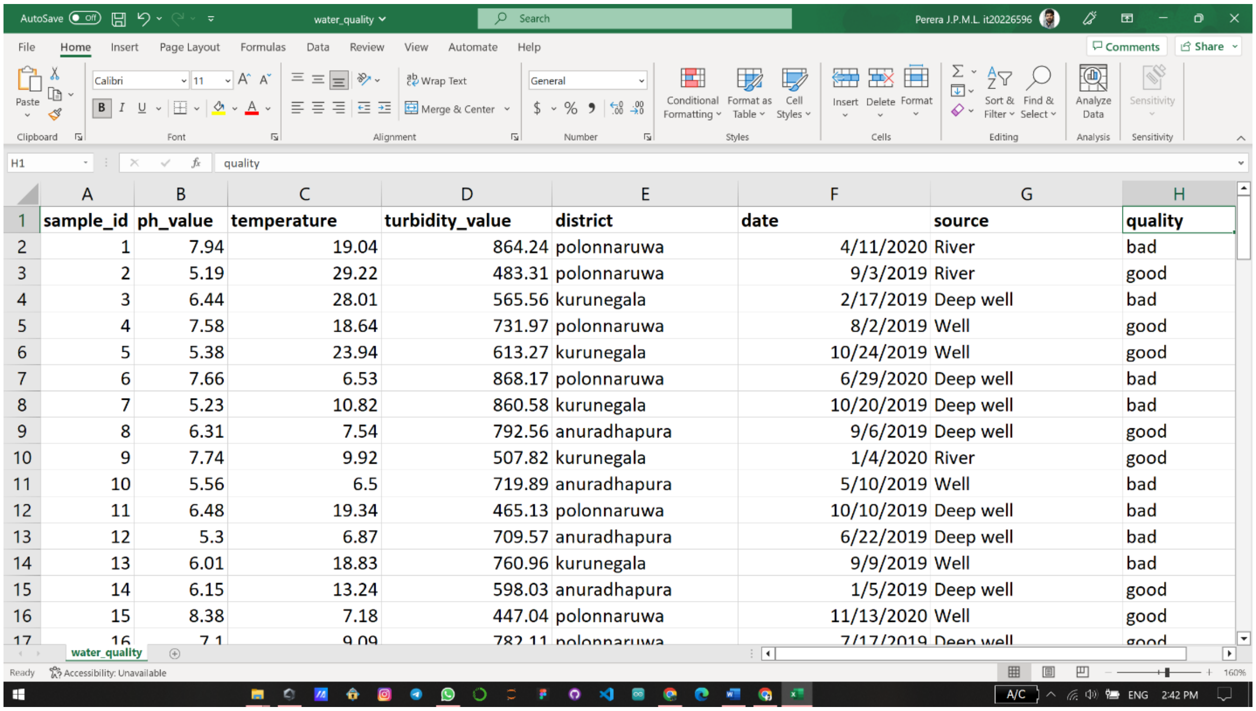

sketch_may25a | Arduino IDE 2.1.0
File Edit Sketch Tools Help
Arduino Uno
sketch_may25a.ino
Output
Ln 17, Col 1 Arduino Uno [not connected]

```
#define SENSOR_PIN A0  // Analog pin connected to the turbidity sensor
int calibrationOffset = 840;  // Calibration offset value

void setup() {
  Serial.begin(9600);
}

void loop() {
  int sensorValue = analogRead(SENSOR_PIN);
  int calibratedValue = calibrationOffset - sensorValue;  // Subtract the calibration offset

  Serial.print("Turbidity: ");
  Serial.println(calibratedValue);

  delay(1000);  // Delay for 1 second
}
```

```
Sketch uses 3140 bytes (9%) of program storage space. Maximum is 32256 bytes.
Global variables use 200 bytes (9%) of dynamic memory, leaving 1848 bytes for local variables. Maximum is 2048 bytes.
```

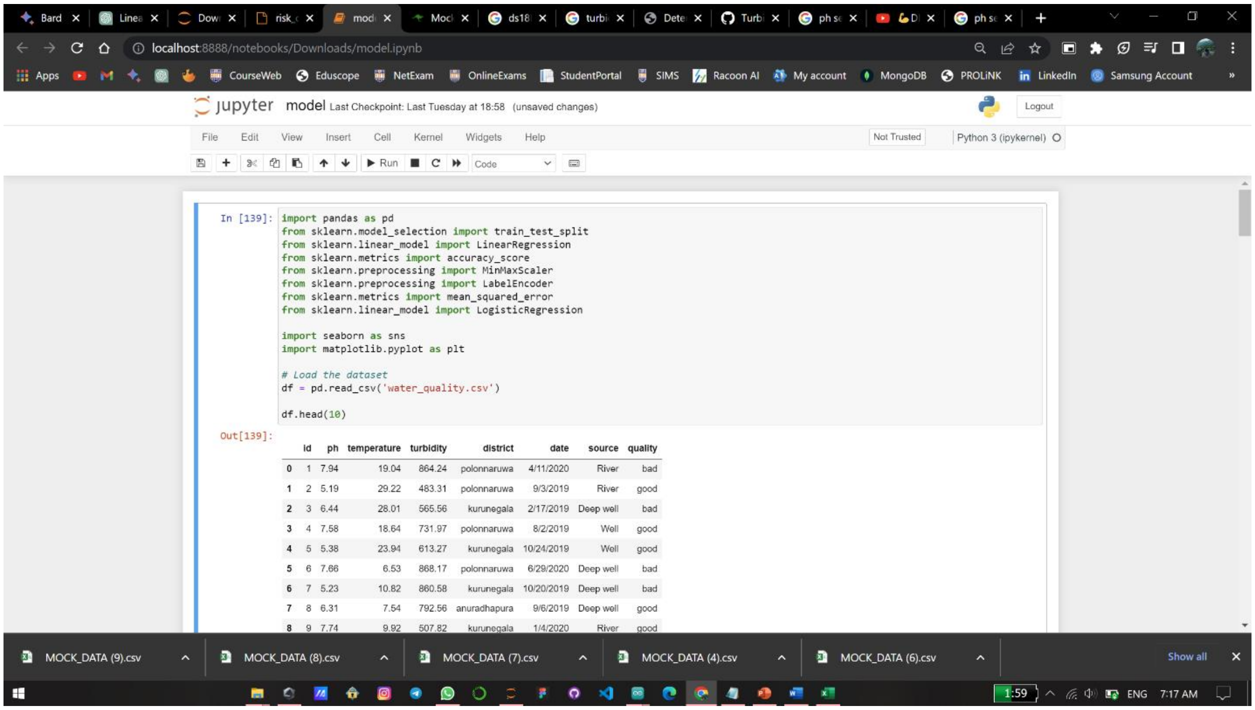
Software solution

Gathered Date Set


AutoSave Off
water_quality
Search
Perera J.P.M.L it20226596
File Home Insert Page Layout Formulas Data Review View Automate Help
Comments Share
H1
quality
water_quality
Ready
Accessibility: Unavailable

sample_id	ph_value	temperature	turbidity_value	district	date	source	quality
1	7.94	19.04	864.24	polonnaruwa	4/11/2020	River	bad
2	5.19	29.22	483.31	polonnaruwa	9/3/2019	River	good
3	6.44	28.01	565.56	kurunegala	2/17/2019	Deep well	bad
4	7.58	18.64	731.97	polonnaruwa	8/2/2019	Well	good
5	5.38	23.94	613.27	kurunegala	10/24/2019	Well	good
6	7.66	6.53	868.17	polonnaruwa	6/29/2020	Deep well	bad
7	5.23	10.82	860.58	kurunegala	10/20/2019	Deep well	bad
8	6.31	7.54	792.56	anuradhapura	9/6/2019	Deep well	good
9	7.74	9.92	507.82	kurunegala	1/4/2020	River	good
10	5.56	6.5	719.89	anuradhapura	5/10/2019	Well	bad
11	6.48	19.34	465.13	polonnaruwa	10/10/2019	Deep well	bad
12	5.3	6.87	709.57	anuradhapura	6/22/2019	Deep well	bad
13	6.01	18.83	760.96	kurunegala	9/9/2019	Well	bad
14	6.15	13.24	598.03	anuradhapura	1/5/2019	Deep well	good
15	8.38	7.18	447.04	polonnaruwa	11/13/2020	Well	good
16	7.1	9.09	782.11	polonnaruwa	7/17/2019	Deep well	good

Import Libraries & load the dataset.

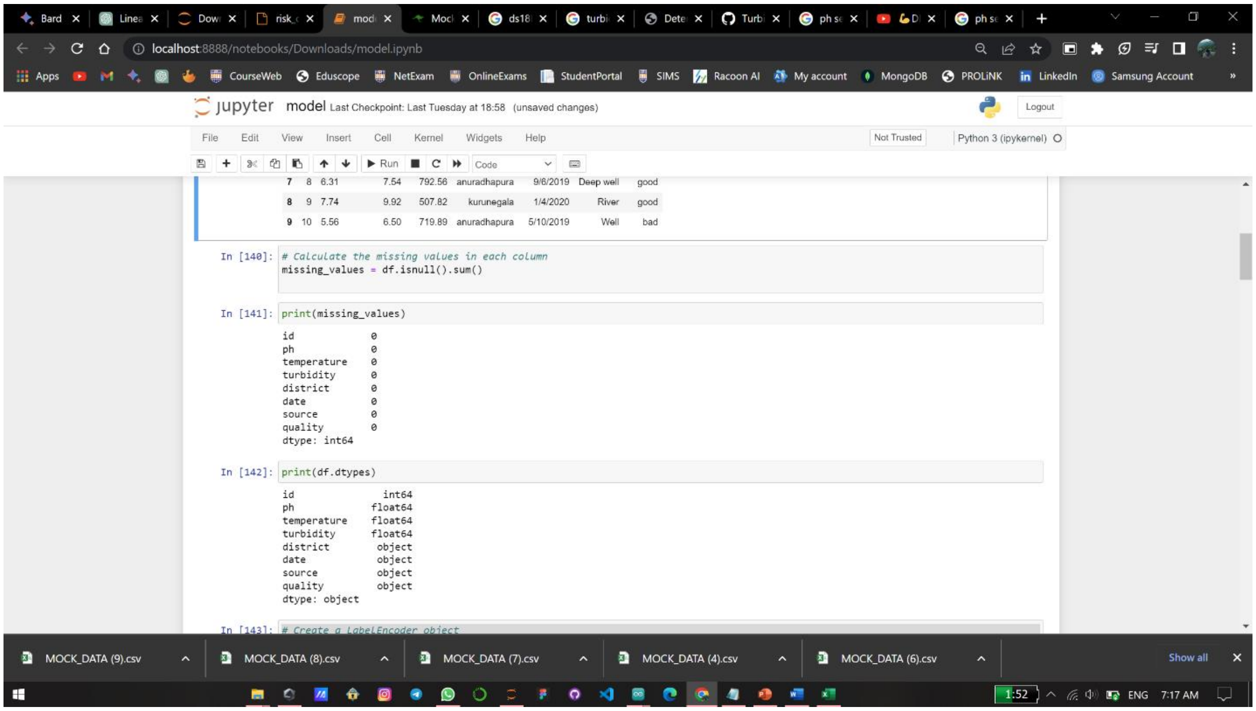


In [139]:
```python
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import mean_squared_error
from sklearn.linear_model import LogisticRegression

import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv('water_quality.csv')

df.head(10)
```

Out[139]:

	id	ph	temperature	turbidity	district	date	source	quality
0	1	7.94	19.04	864.24	polonnaruwa	4/11/2020	River	bad
1	2	5.19	29.22	483.31	polonnaruwa	9/3/2019	River	good
2	3	6.44	28.01	565.56	kurunegala	2/17/2019	Deep well	bad
3	4	7.58	18.64	731.97	polonnaruwa	8/2/2019	Well	good
4	5	5.38	23.94	613.27	kurunegala	10/24/2019	Well	good
5	6	7.66	6.53	868.17	polonnaruwa	6/29/2020	Deep well	bad
6	7	5.23	10.82	860.58	kurunegala	10/20/2019	Deep well	bad
7	8	6.31	7.54	792.56	anuradhapura	9/6/2019	Deep well	good
8	9	7.74	9.92	507.82	kurunegala	1/4/2020	River	good

Checking missing values



7	8	6.31	7.54	792.56	anuradhapura	9/6/2019	Deep well	good
8	9	7.74	9.92	507.82	kurunegala	1/4/2020	River	good
9	10	5.56	6.50	719.89	anuradhapura	5/10/2019	Well	bad

In [140]:
```python
# Calculate the missing values in each column
missing_values = df.isnull().sum()
```

In [141]:
```python
print(missing_values)
```

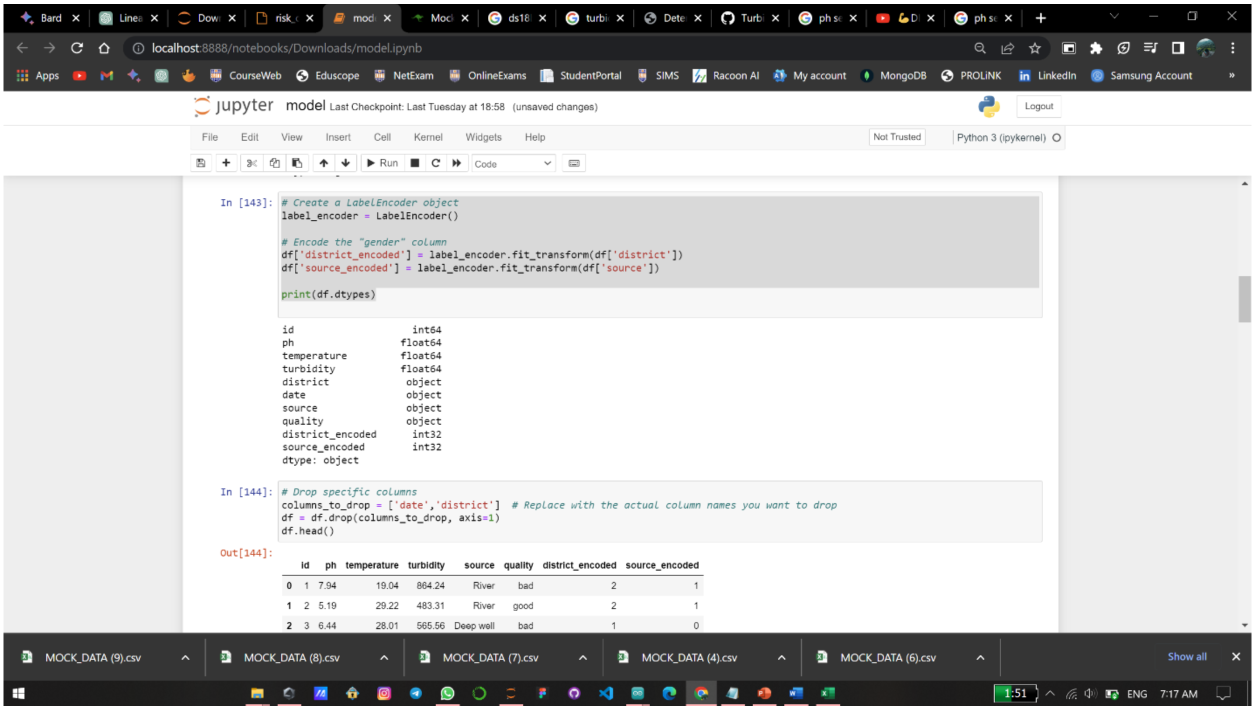
```
id             0
ph             0
temperature    0
turbidity      0
district       0
date           0
source         0
quality        0
dtype: int64
```

In [142]:
```python
print(df.dtypes)
```

```
id               int64
ph             float64
temperature    float64
turbidity      float64
district        object
date            object
source          object
quality         object
dtype: object
```

In [143]:
```python
# Create a LabelEncoder object
```

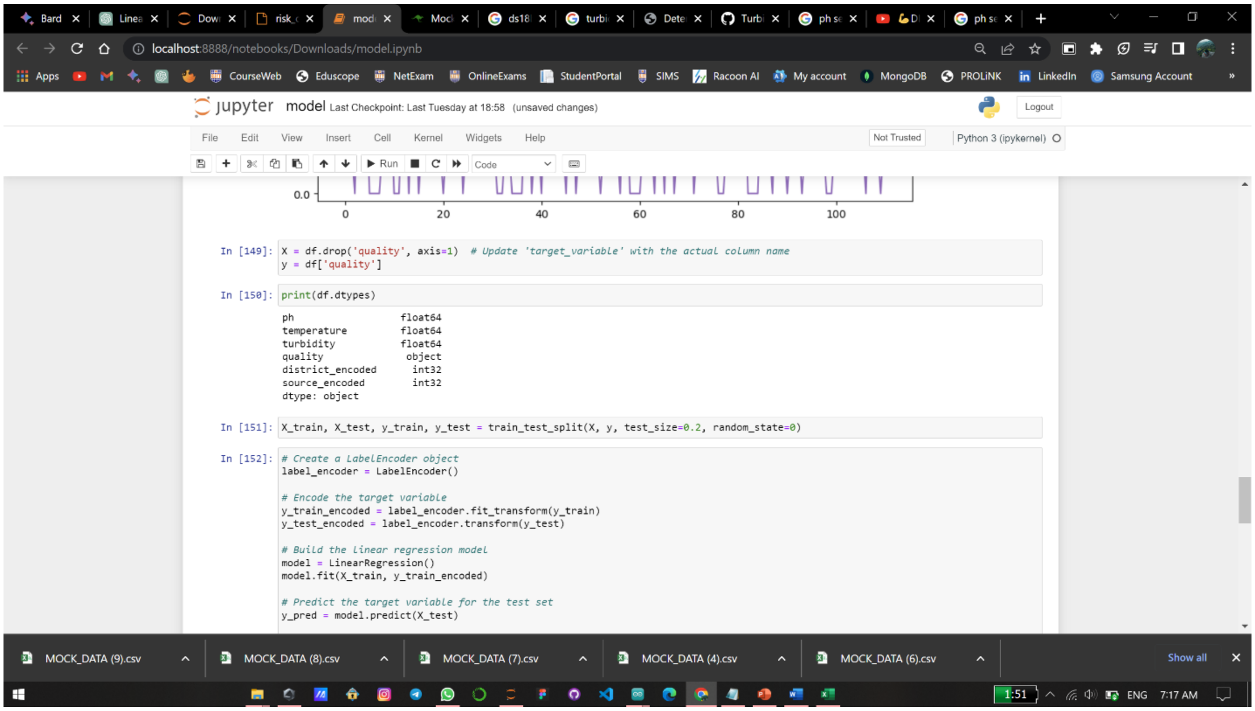
Using Label Encoding



```
In [143]: # Create a LabelEncoder object
          label_encoder = LabelEncoder()

          # Encode the "gender" column
          df['district_encoded'] = label_encoder.fit_transform(df['district'])
          df['source_encoded'] = label_encoder.fit_transform(df['source'])

          print(df.dtypes)

          id                    int64
          ph                  float64
          temperature         float64
          turbidity           float64
          district             object
          date                 object
          source               object
          quality              object
          district_encoded      int32
          source_encoded        int32
          dtype: object

In [144]: # Drop specific columns
          columns_to_drop = ['date','district']  # Replace with the actual column names you want to drop
          df = df.drop(columns_to_drop, axis=1)
          df.head()

Out[144]:
```

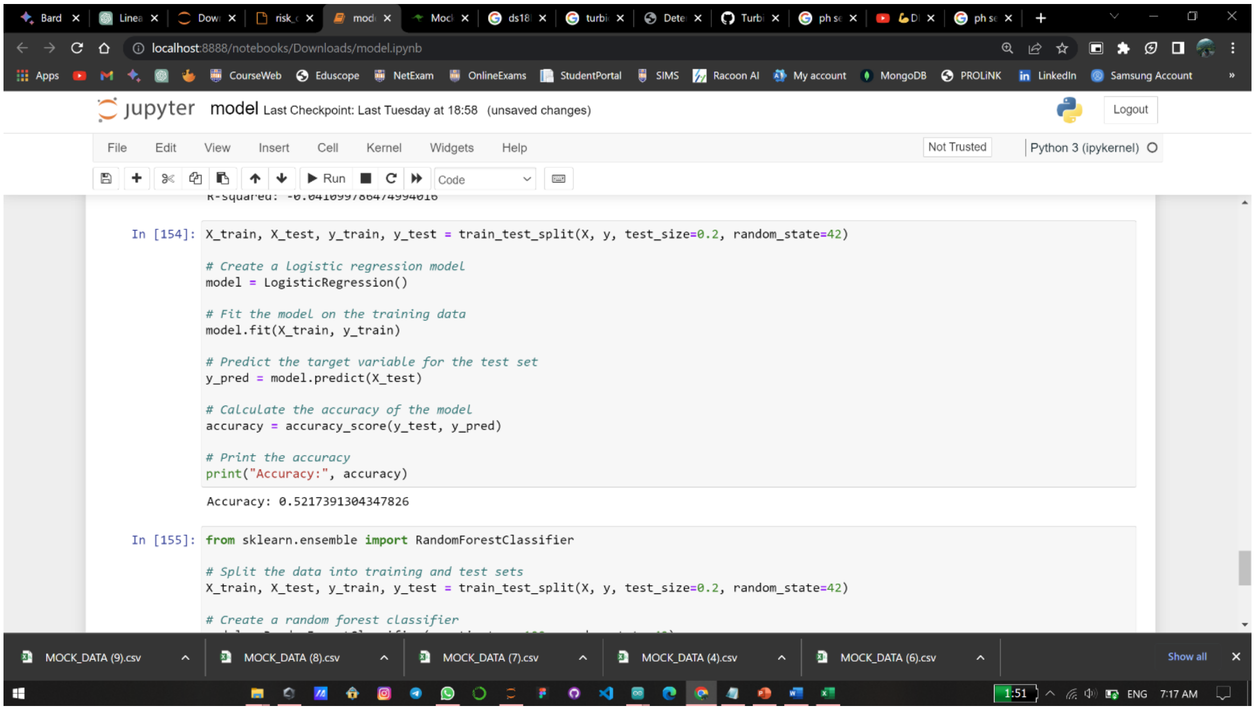
	id	ph	temperature	turbidity	source	quality	district_encoded	source_encoded
0	1	7.94	19.04	864.24	River	bad	2	1
1	2	5.19	29.22	483.31	River	good	2	1
2	3	6.44	28.01	565.56	Deep well	bad	1	0

Split the Dataset.



```
In [149]: X = df.drop('quality', axis=1)  # Update 'target_variable' with the actual column name
          y = df['quality']

In [150]: print(df.dtypes)

          ph                  float64
          temperature         float64
          turbidity           float64
          quality              object
          district_encoded      int32
          source_encoded        int32
          dtype: object

In [151]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

In [152]: # Create a LabelEncoder object
          label_encoder = LabelEncoder()

          # Encode the target variable
          y_train_encoded = label_encoder.fit_transform(y_train)
          y_test_encoded = label_encoder.transform(y_test)

          # Build the linear regression model
          model = LinearRegression()
          model.fit(X_train, y_train_encoded)

          # Predict the target variable for the test set
          y_pred = model.predict(X_test)
```

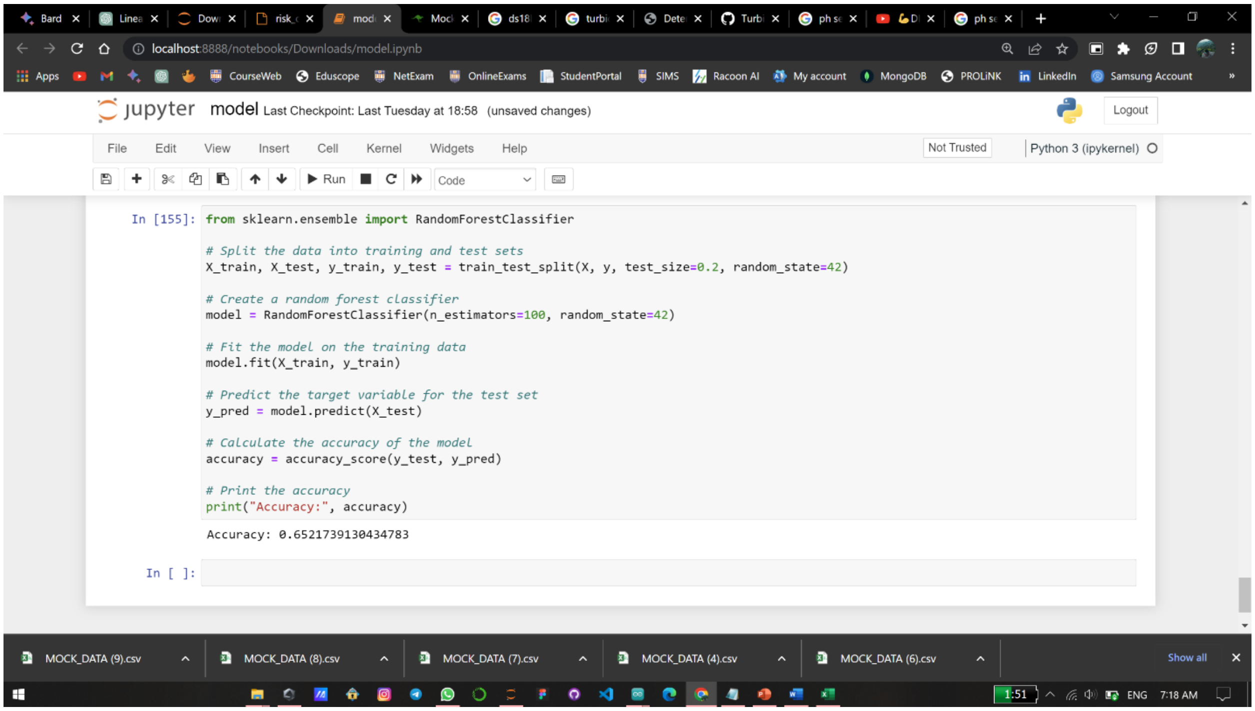
Build the Logistic regression Model & get the accuracy.


localhost:8888/notebooks/Downloads/model.ipynb
jupyter model Last Checkpoint: Last Tuesday at 18:58 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
MOCK_DATA (9).csv
MOCK_DATA (8).csv
MOCK_DATA (7).csv
MOCK_DATA (4).csv
MOCK_DATA (6).csv
Show all
7:17 AM

```
R-squared: -0.04109978647499016
```

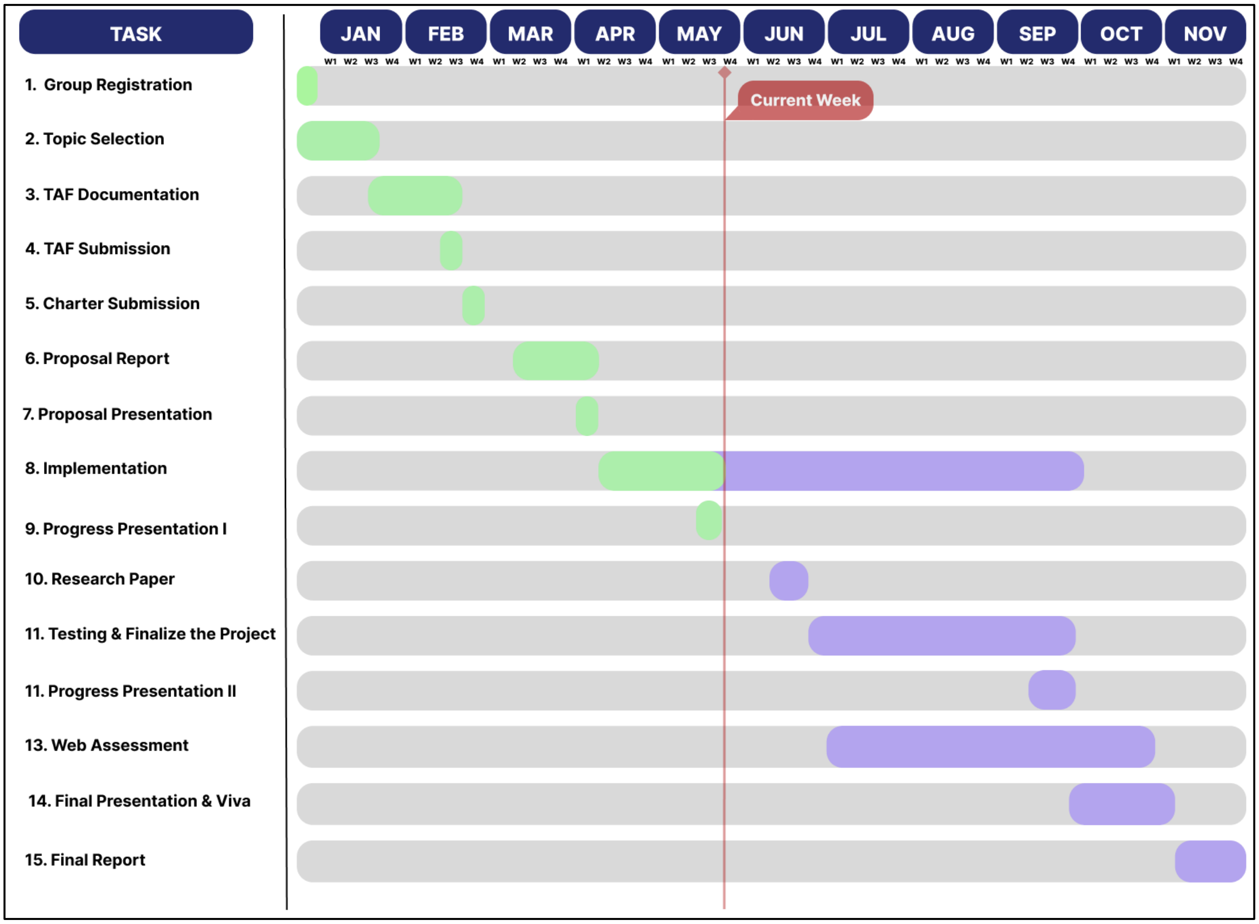
```
In [154]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

          # Create a logistic regression model
          model = LogisticRegression()

          # Fit the model on the training data
          model.fit(X_train, y_train)

          # Predict the target variable for the test set
          y_pred = model.predict(X_test)

          # Calculate the accuracy of the model
          accuracy = accuracy_score(y_test, y_pred)

          # Print the accuracy
          print("Accuracy:", accuracy)

          Accuracy: 0.5217391304347826

In [155]: from sklearn.ensemble import RandomForestClassifier

          # Split the data into training and test sets
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

          # Create a random forest classifier
```

Build the Random Forest Classifier Model & get the accuracy.


localhost:8888/notebooks/Downloads/model.ipynb
jupyter model Last Checkpoint: Last Tuesday at 18:58 (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
MOCK_DATA (9).csv
MOCK_DATA (8).csv
MOCK_DATA (7).csv
MOCK_DATA (4).csv
MOCK_DATA (6).csv
Show all
7:18 AM

```
In [155]: from sklearn.ensemble import RandomForestClassifier

          # Split the data into training and test sets
          X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

          # Create a random forest classifier
          model = RandomForestClassifier(n_estimators=100, random_state=42)

          # Fit the model on the training data
          model.fit(X_train, y_train)

          # Predict the target variable for the test set
          y_pred = model.predict(X_test)

          # Calculate the accuracy of the model
          accuracy = accuracy_score(y_test, y_pred)

          # Print the accuracy
          print("Accuracy:", accuracy)

          Accuracy: 0.6521739130434783

In [ ]:
```

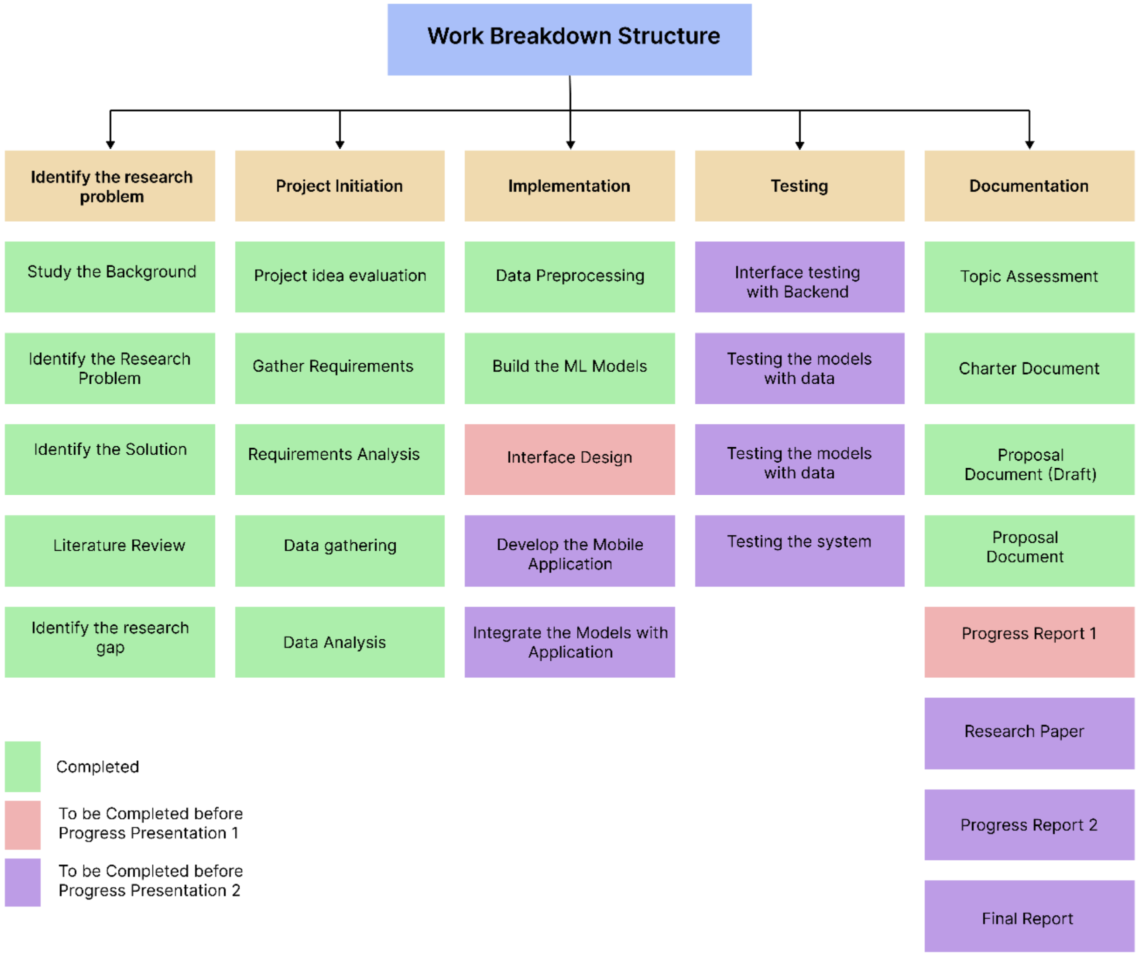
4 Gantt Chart



🟩 - Completed

🟪 - In Progress

5 Work Breakdown Structure



Work Breakdown Structure

Identify the research problem	Project Initiation	Implementation	Testing	Documentation
Study the Background	Project idea evaluation	Data Preprocessing	Interface testing with Backend	Topic Assessment
Identify the Research Problem	Gather Requirements	Build the ML Models	Testing the models with data	Charter Document
Identify the Solution	Requirements Analysis	Interface Design	Testing the models with data	Proposal Document (Draft)
Literature Review	Data gathering	Develop the Mobile Application	Testing the system	Proposal Document
Identify the research gap	Data Analysis	Integrate the Models with Application		Progress Report 1
				Research Paper
				Progress Report 2
				Final Report

🟩 Completed

🟥 To be Completed before Progress Presentation 1

🟪 To be Completed before Progress Presentation 2