FLARE PATH – ADVANCED VEHICLE FIRE SAFETY AND MONITORING WITH RAPID EMERGENCY DISPATCH SOLUTIONS

R24-058

Status Document-1

Abeywardhana D N – IT21133718

B.Sc. (Hons) Degree in Information Technology specializing in

Information Technology

Department of Information Technology

Sri Lanka Institute of Information Technology

Sri Lanka

May 2024

Group Details

Supervisor – Mr.Nelum Chathuranga Amarasena

Co-supervisor – Mr. Deemantha Nayanajith Siriwardana

External Supervisor – Mr. Onray Sahinda

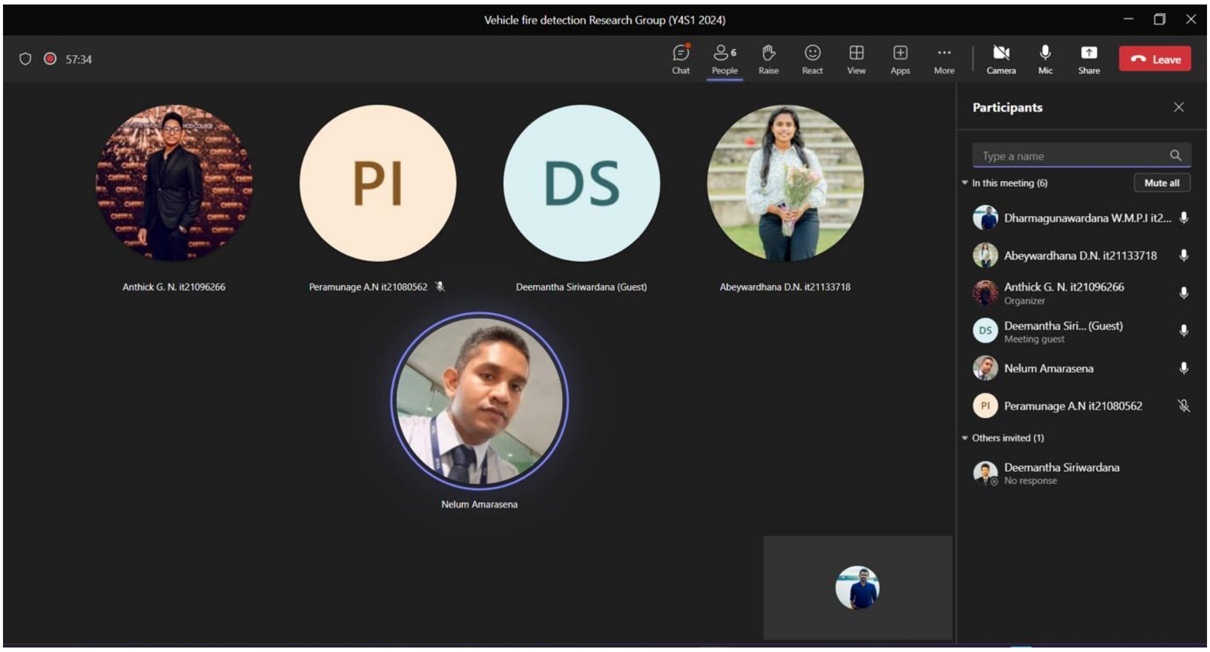
Student Name	Student ID	Contact No	Email Address
Dharmagunawardana W.M.P.I	IT21132346	0772785361	it21132346@my.sliit.lk
Anthick G.N	IT21096266	0779820516	it21096266@my.sliit.lk
Abeywardhana D.N	IT21133718	0714057155	it21133718@my.sliit.lk
Peramunage A.N	IT21080562	0713999266	it21080562@my.sliit.lk

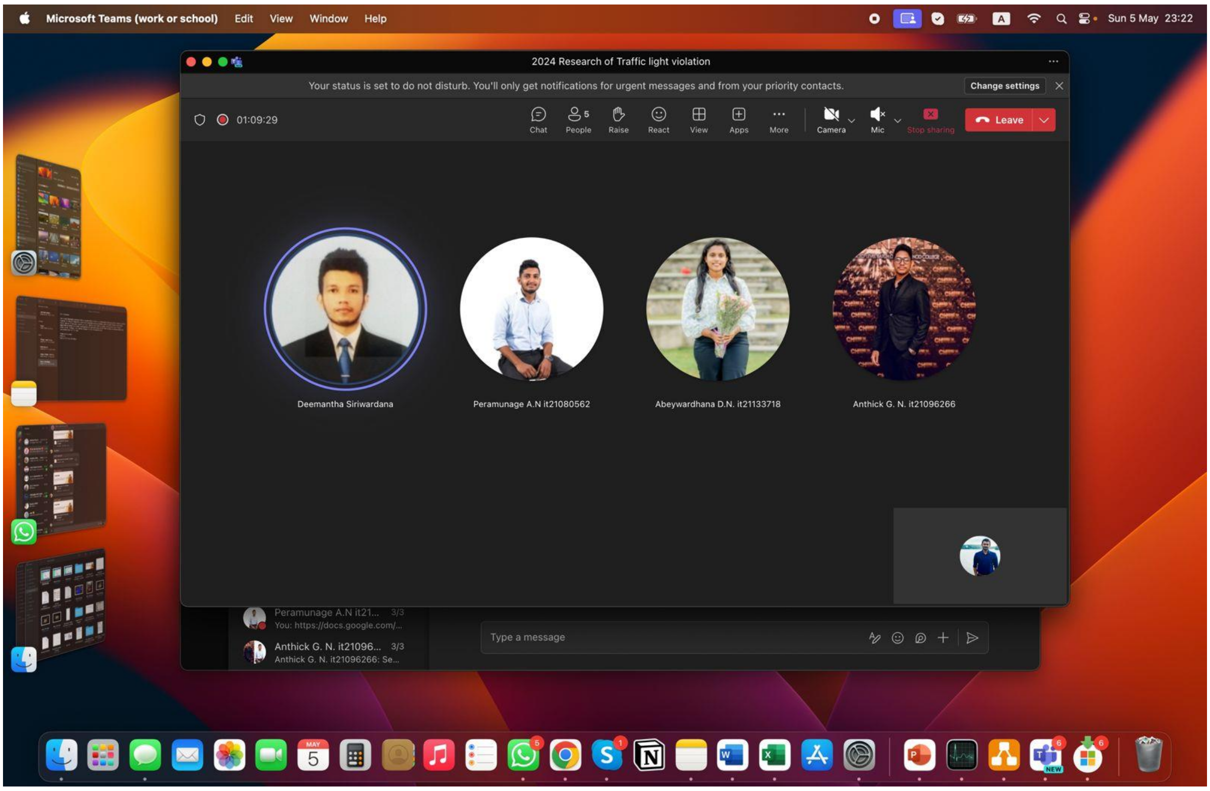
Table of Contents

Meetings & Calls

Meetings with supervisor and co-supervisor

Meeting with both supervisor and co-supervisor about the project progress and improvements that we need to do to our project.

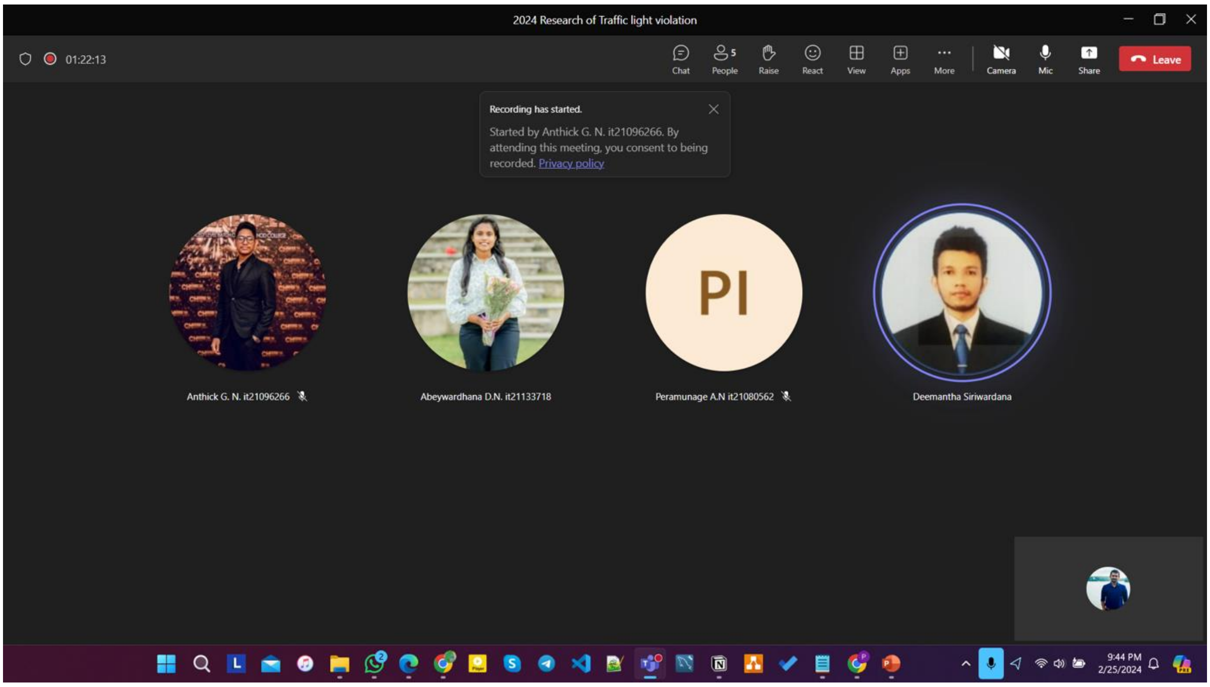




Your status is set to do not disturb. You'll only get notifications for urgent messages and from your priority contacts.

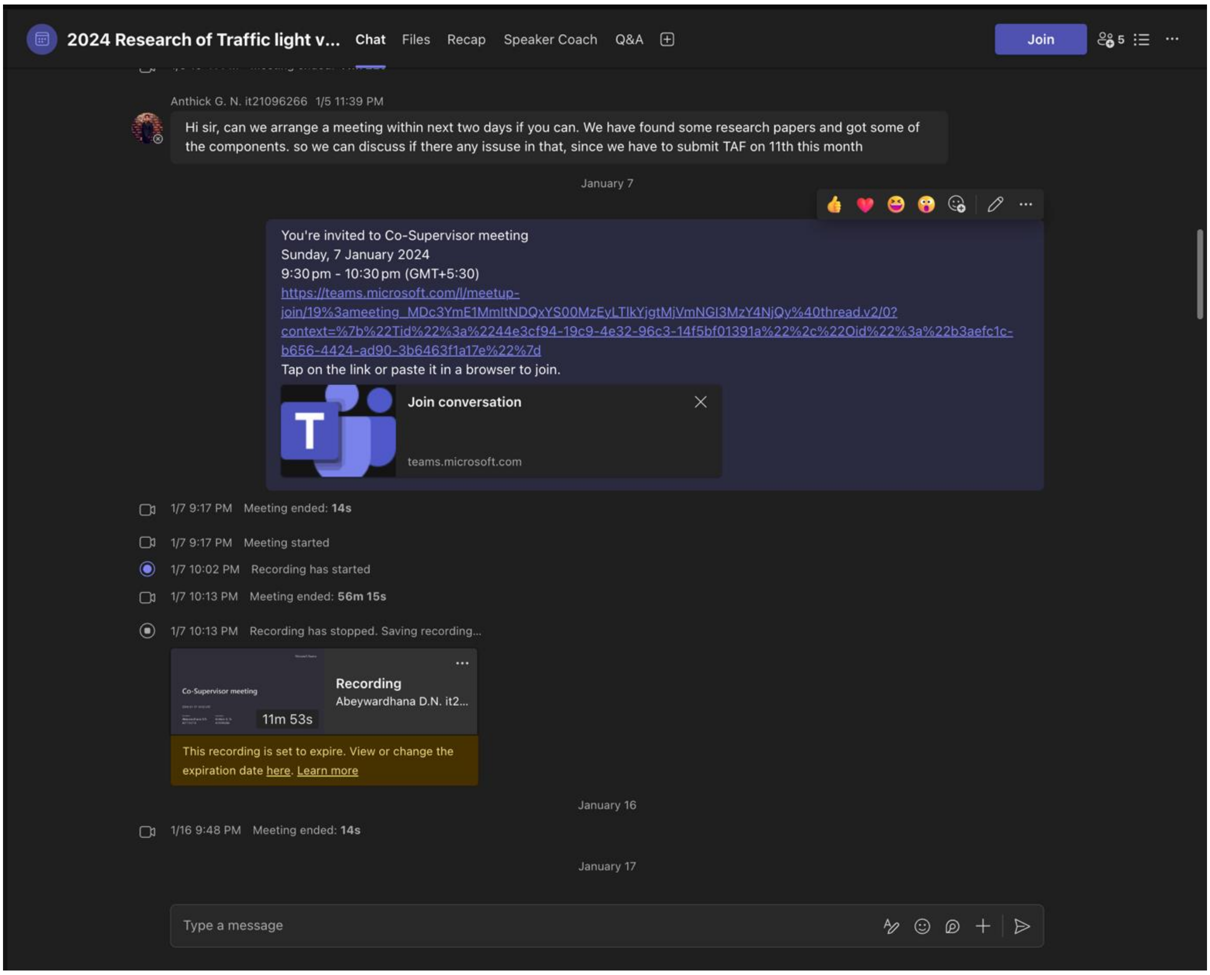
Peramunage A.N it21... 3/3
You: https://docs.google.com/...

Anthick G. N. it21096... 3/3
Anthick G. N. it21096266: Se...



Recording has started.

Started by Anthick G. N. it21096266. By attending this meeting, you consent to being recorded. Privacy policy



Anthick G. N. it21096266 1/5 11:39 PM

Hi sir, can we arrange a meeting within next two days if you can. We have found some research papers and got some of the components. so we can discuss if there any issuse in that, since we have to submit TAF on 11th this month

January 7

You're invited to Co-Supervisor meeting
Sunday, 7 January 2024
9:30 pm - 10:30 pm (GMT+5:30)
https://teams.microsoft.com/l/meetup-join/19%3ameeting_MDc3YmE1MmItNDQxYS00MzEyLTlkYjgtMjVmNGI3MzY4NjQy%40thread.v2/0?context=%7b%22Tid%22%3a%2244e3cf94-19c9-4e32-96c3-14f5bf01391a%22%2c%22Oid%22%3a%22b3aefc1c-b656-4424-ad90-3b6463f1a17e%22%7d
Tap on the link or paste it in a browser to join.

Join conversation
teams.microsoft.com

1/7 9:17 PM Meeting ended: **14s**

1/7 9:17 PM Meeting started

1/7 10:02 PM Recording has started

1/7 10:13 PM Meeting ended: **56m 15s**

1/7 10:13 PM Recording has stopped. Saving recording...

Recording
Abeywardhana D.N. it2...

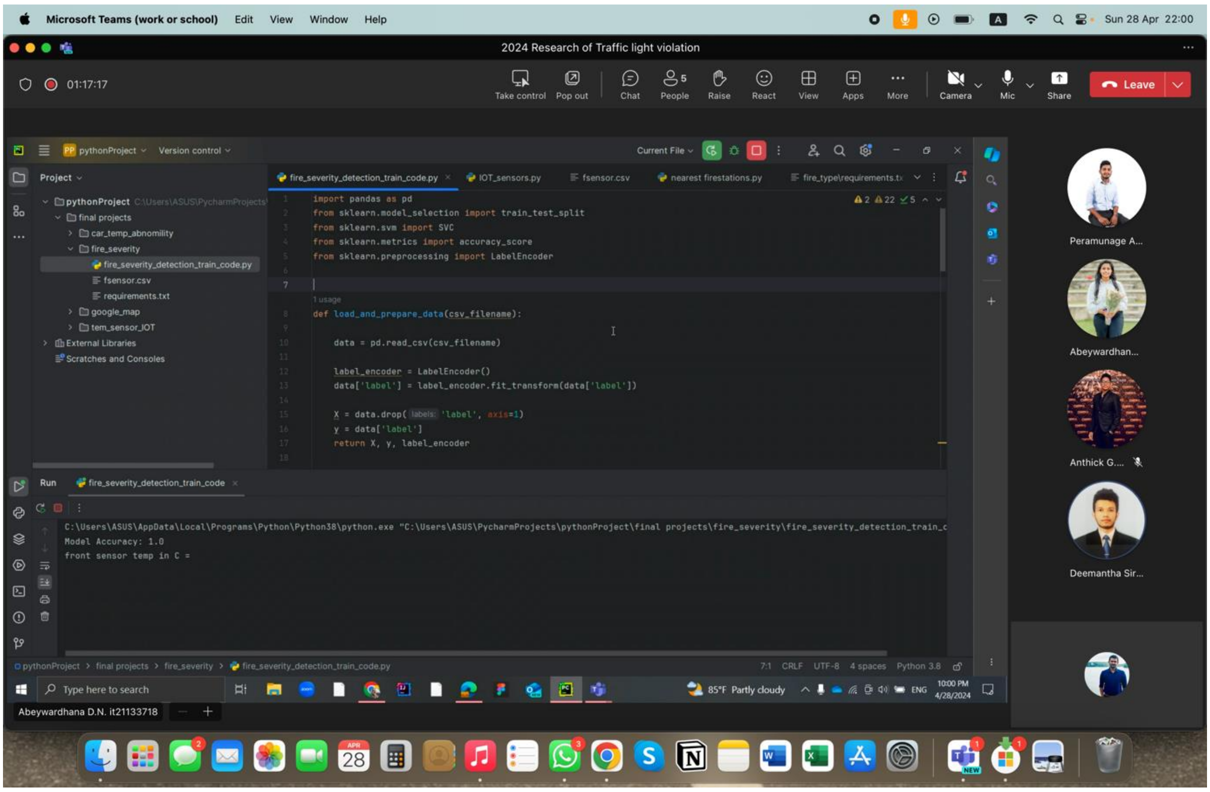
This recording is set to expire. View or change the expiration date here. Learn more

January 16

1/16 9:48 PM Meeting ended: **14s**

January 17

Code review with co-supervisor.


2024 Research of Traffic light violation
01:17:17
Peramunage A...
Abeywardhan...
Anthick G....
Deemantha Sir...
Abeywardhana D.N. it21133718

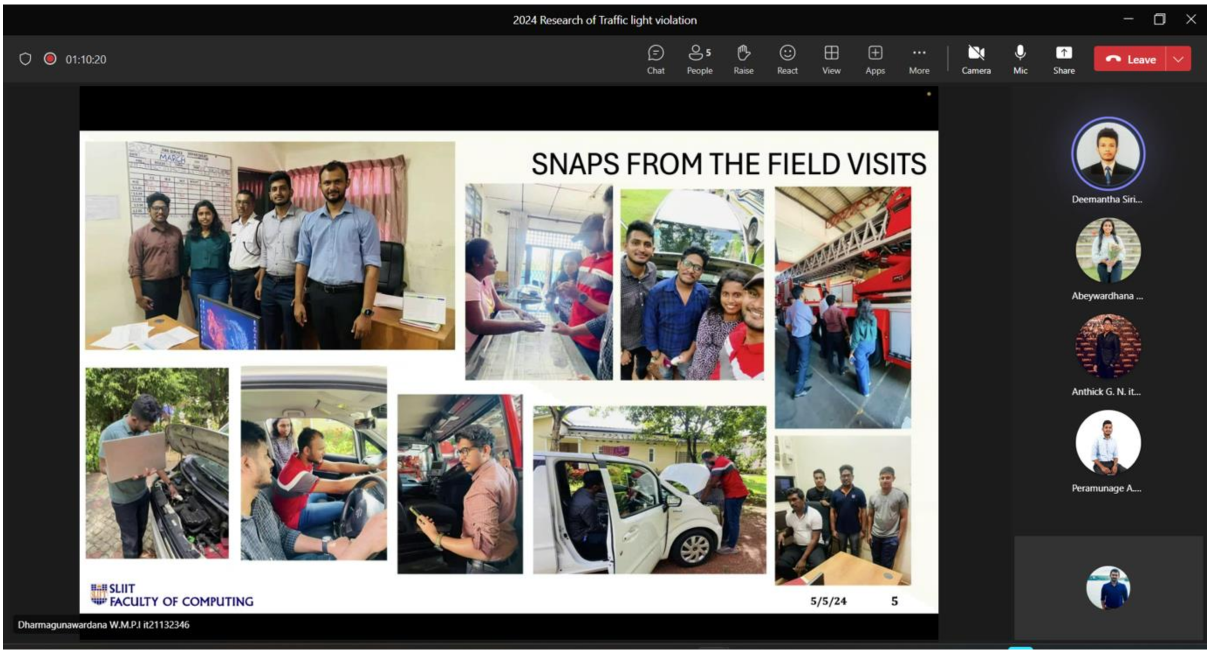
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
from sklearn.preprocessing import LabelEncoder


def load_and_prepare_data(csv_filename):

    data = pd.read_csv(csv_filename)

    label_encoder = LabelEncoder()
    data['label'] = label_encoder.fit_transform(data['label'])

    X = data.drop('label', axis=1)
    y = data['label']
    return X, y, label_encoder
```

```
C:\Users\ASUS\AppData\Local\Programs\Python\Python38\python.exe "C:\Users\ASUS\PycharmProjects\pythonProject\final projects\fire_severity\fire_severity_detection_train_c
Model Accuracy: 1.0
front sensor temp in C =
```

Group meeting with group members.

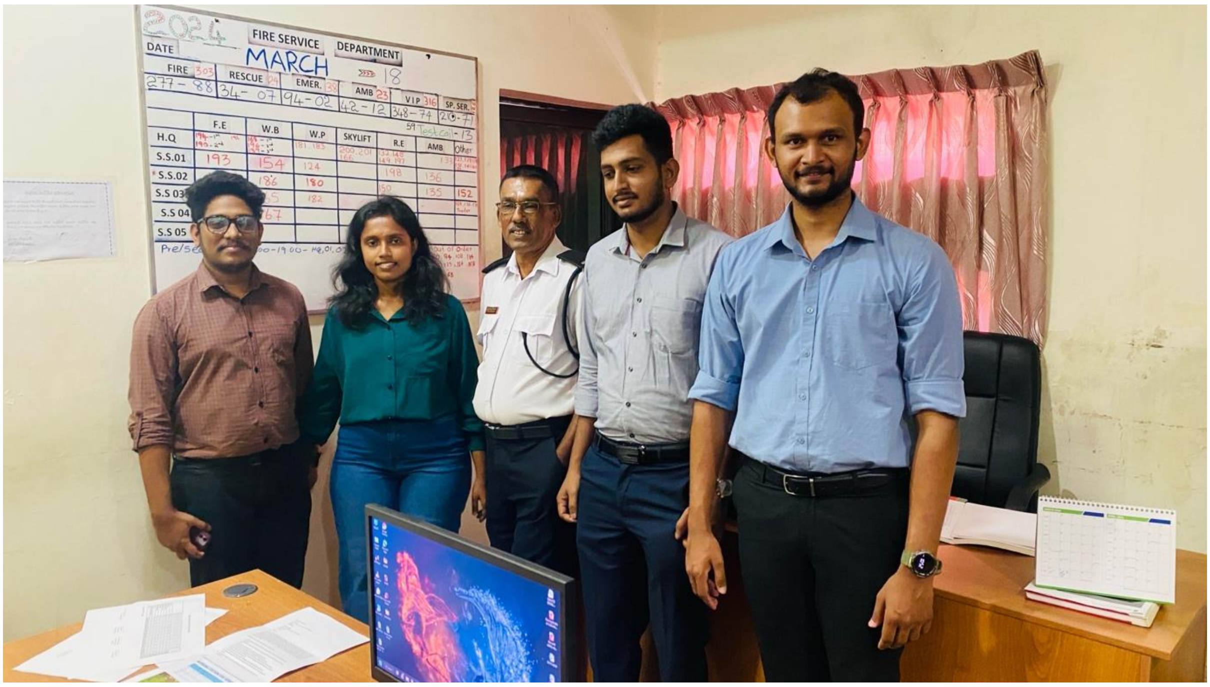
Call with Final Year Research Project
01:05:48
Peramunage A.N it21080562
Abeywardhana D.N. it21133718
Anthick G. N. it21096266

Presentation review with supervisor

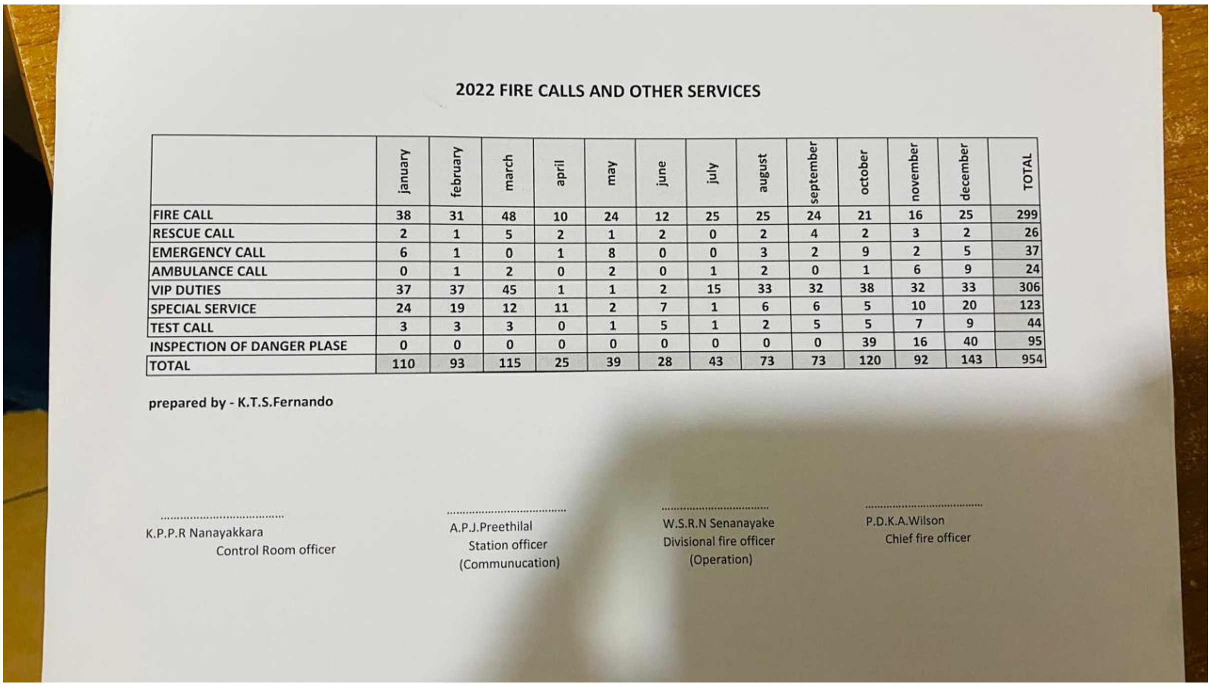


Meetings with Domain Experts

Meeting With Fire department officers and staff. As well we discussed about the domain knowledge and requirements related to fire department.



Mr. Nanayakkara the chief officer of Fire department



2022 FIRE CALLS AND OTHER SERVICES

	january	february	march	april	may	june	july	august	september	october	november	december	TOTAL
FIRE CALL	38	31	48	10	24	12	25	25	24	21	16	25	299
RESCUE CALL	2	1	5	2	1	2	0	2	4	2	3	2	26
EMERGENCY CALL	6	1	0	1	8	0	0	3	2	9	2	5	37
AMBULANCE CALL	0	1	2	0	2	0	1	2	0	1	6	9	24
VIP DUTIES	37	37	45	1	1	2	15	33	32	38	32	33	306
SPECIAL SERVICE	24	19	12	11	2	7	1	6	6	5	10	20	123
TEST CALL	3	3	3	0	1	5	1	2	5	5	7	9	44
INSPECTION OF DANGER PLASE	0	0	0	0	0	0	0	0	0	39	16	40	95
TOTAL	110	93	115	25	39	28	43	73	73	120	92	143	954

prepared by - K.T.S.Fernando

....
K.P.P.R Nanayakkara
Control Room officer

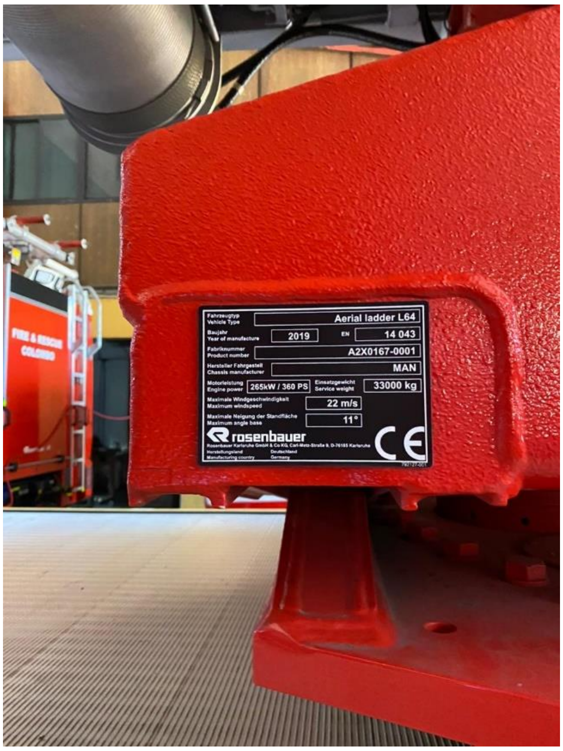
....
A.P.J.Preethilal
Station officer
(Communucation)

....
W.S.R.N Senanayake
Divisional fire officer
(Operation)

....
P.D.K.A.Wilson
Chief fire officer

Manual statistical system of fire department

Snapshots from Field Visit



Fahrzeugtyp
Vehicle Type: Aerial ladder L64

Baujahr
Year of manufacture: 2019

EN: 14 043

Fabriknummer
Product number: A2X0167-0001

Hersteller Fahrgestell
Chassis manufacturer: MAN

Motorleistung
Engine power: 265kW / 360 PS

Einsatzgewicht
Service weight: 33000 kg

Maximale Windgeschwindigkeit
Maximum windspeed: 22 m/s

Maximale Neigung der Standfläche
Maximum angle base: 11°

rosenbauer
Rosenbauer Karlsruhe GmbH & Co KG, Carl-Metz-Straße 9, D-76185 Karlsruhe

Herstellungsland
Manufacturing country: Deutschland
Germany

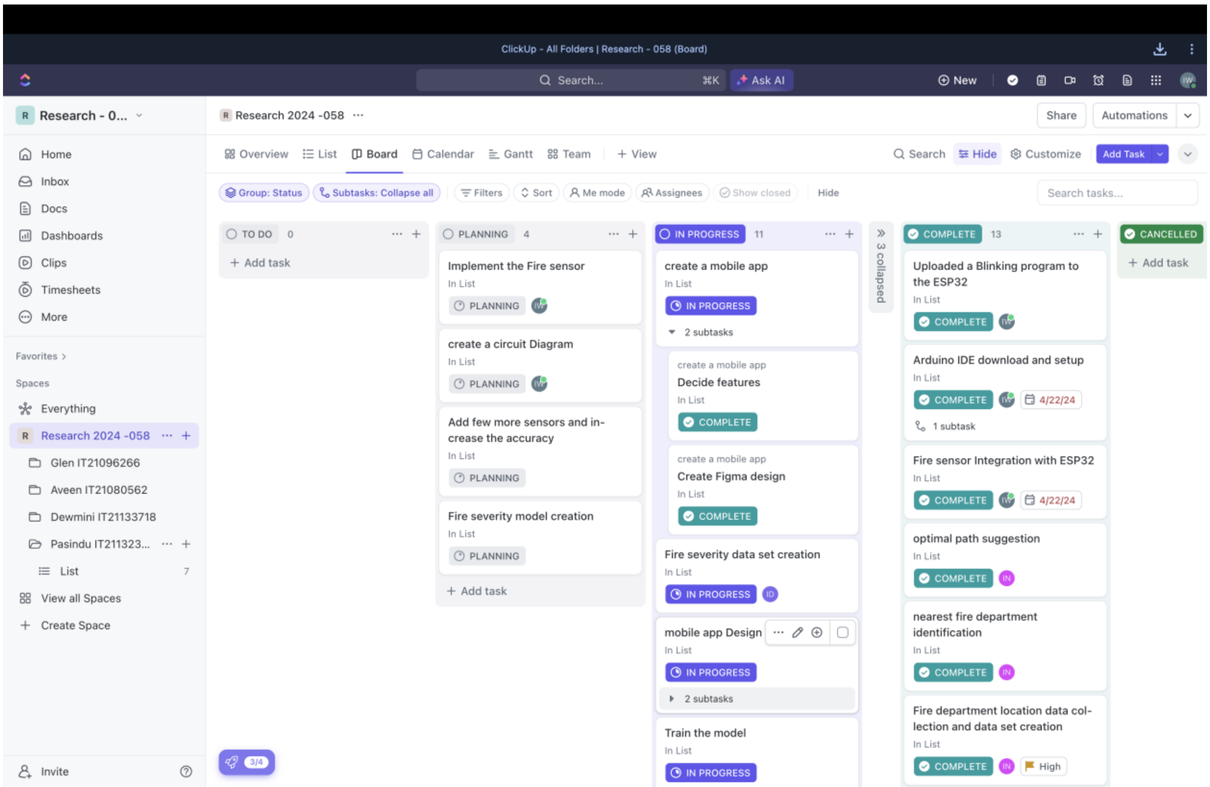
TGS 33.400

MAN
.00 bar

MAN Truck & Bus SE
WMA52WZZ9LL076990
18000 kg
26500 kg
7500 kg
13000 kg
R134a
1.1 kg
DATE

COLOMBO
MUNICIPAL
COUNCIL
FIRE
RESCUE
COLOMBO

Click up Tasks Allocation



TO DO 0

+ Add task

PLANNING 4

- Implement the Fire sensor — In List — PLANNING
- create a circuit Diagram — In List — PLANNING
- Add few more sensors and increase the accuracy — In List — PLANNING
- Fire severity model creation — In List — PLANNING

+ Add task

IN PROGRESS 11

- create a mobile app — In List — IN PROGRESS — 2 subtasks
  - create a mobile app: Decide features — In List — COMPLETE
  - create a mobile app: Create Figma design — In List — COMPLETE
- Fire severity data set creation — In List — IN PROGRESS
- mobile app Design — In List — IN PROGRESS — 2 subtasks
- Train the model — In List — IN PROGRESS

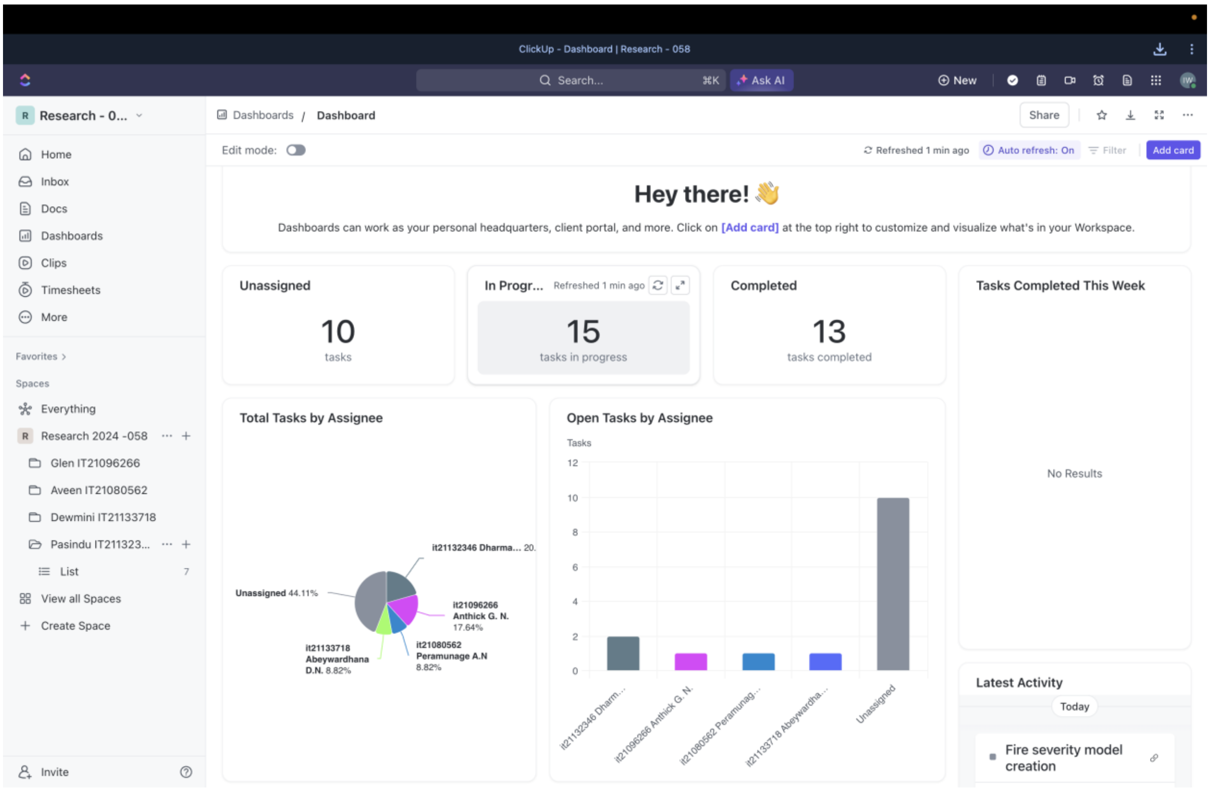
COMPLETE 13

- Uploaded a Blinking program to the ESP32 — In List — COMPLETE
- Arduino IDE download and setup — In List — COMPLETE — 4/22/24 — 1 subtask
- Fire sensor Integration with ESP32 — In List — COMPLETE — 4/22/24
- optimal path suggestion — In List — COMPLETE
- nearest fire department identification — In List — COMPLETE
- Fire department location data collection and data set creation — In List — COMPLETE — High

CANCELLED

+ Add task

Click Up Dashboard



Hey there! 👋

Dashboards can work as your personal headquarters, client portal, and more. Click on [Add card] at the top right to customize and visualize what's in your Workspace.

Unassigned: 10 tasks

In Progr... (Refreshed 1 min ago): 15 tasks in progress

Completed: 13 tasks completed

Tasks Completed This Week: No Results

Total Tasks by Assignee

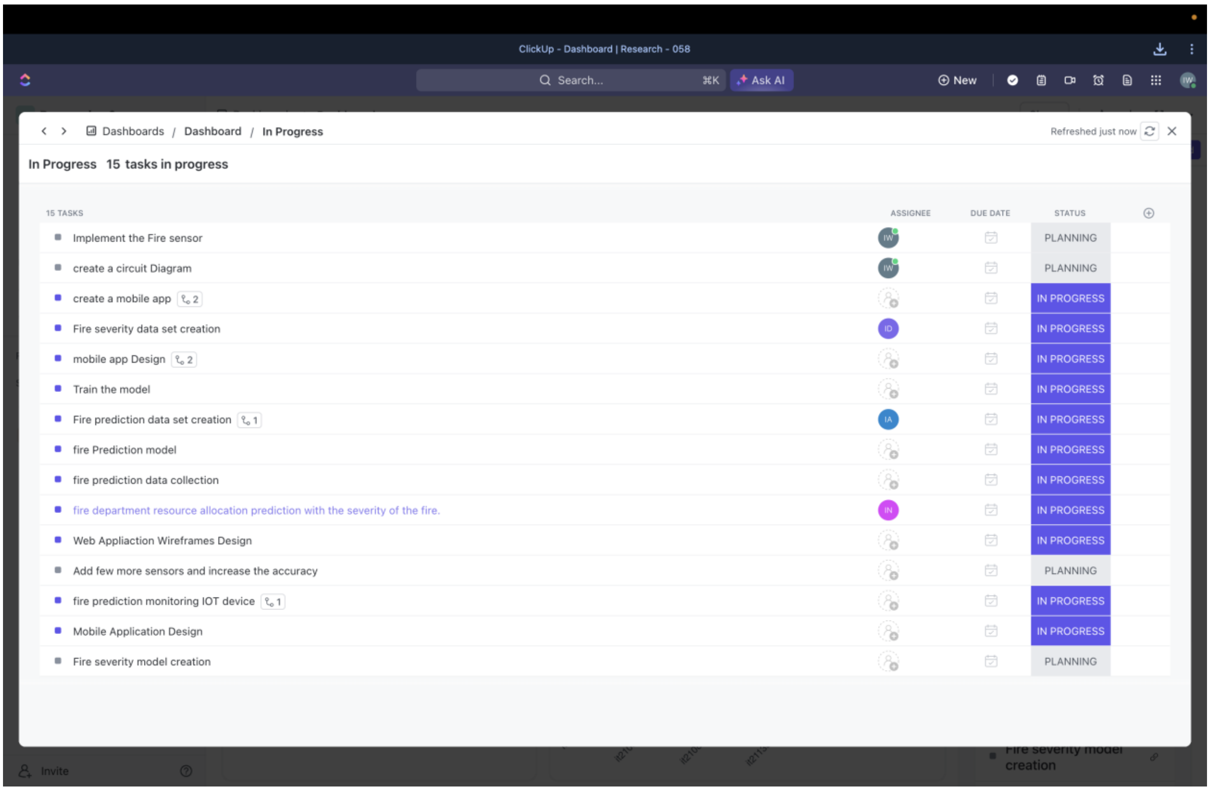
Open Tasks by Assignee

Latest Activity

Today

- Fire severity model creation

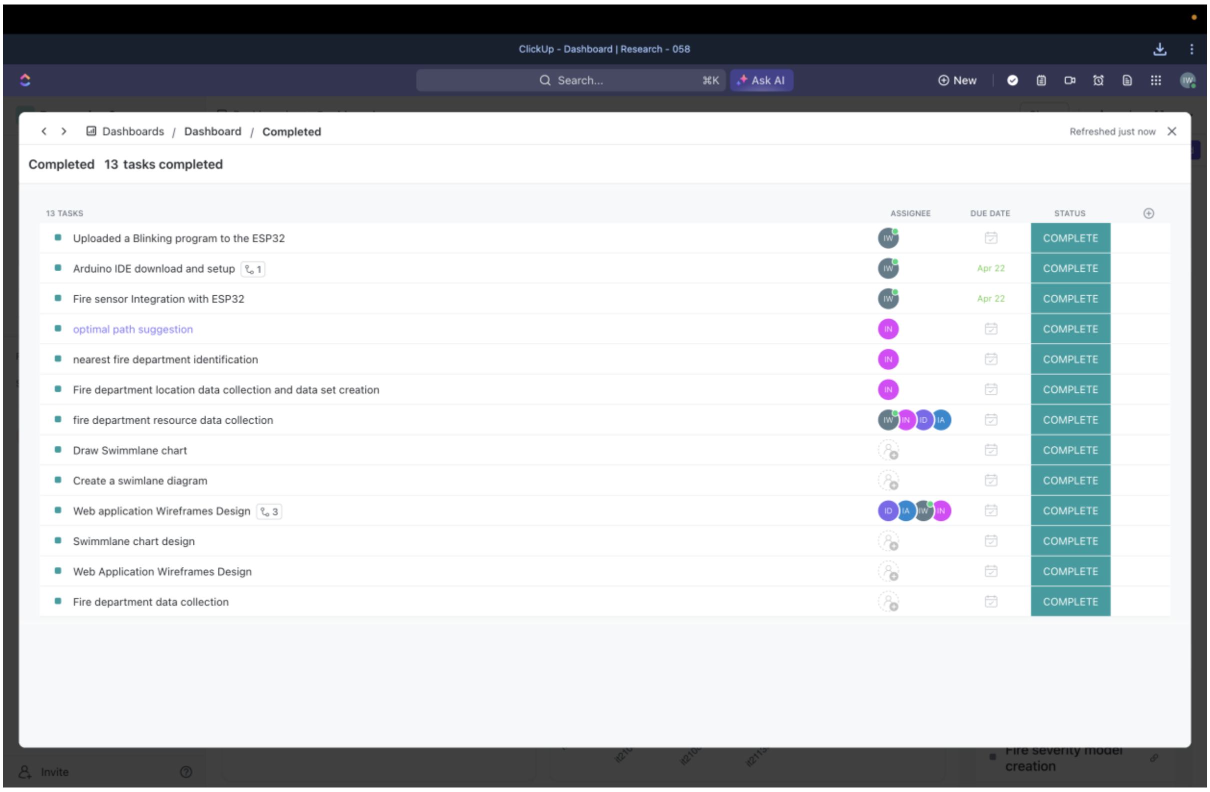
In Progress Tasks


ClickUp - Dashboard | Research - 058
Search...
⌘K
Ask AI
New
Dashboards / Dashboard / In Progress
Refreshed just now
Invite

In Progress 15 tasks in progress

15 TASKS	ASSIGNEE	DUE DATE	STATUS
Implement the Fire sensor	IW		PLANNING
create a circuit Diagram	IW		PLANNING
create a mobile app 2			IN PROGRESS
Fire severity data set creation	ID		IN PROGRESS
mobile app Design 2			IN PROGRESS
Train the model			IN PROGRESS
Fire prediction data set creation 1	IA		IN PROGRESS
fire Prediction model			IN PROGRESS
fire prediction data collection			IN PROGRESS
fire department resource allocation prediction with the severity of the fire.	IN		IN PROGRESS
Web Appliaction Wireframes Design			IN PROGRESS
Add few more sensors and increase the accuracy			PLANNING
fire prediction monitoring IOT device 1			IN PROGRESS
Mobile Application Design			IN PROGRESS
Fire severity model creation			PLANNING

Completed Tasks up to PP1


ClickUp - Dashboard | Research - 058
Search...
⌘K
Ask AI
New
Dashboards / Dashboard / Completed
Refreshed just now
Invite

Completed 13 tasks completed

13 TASKS	ASSIGNEE	DUE DATE	STATUS
Uploaded a Blinking program to the ESP32	IW		COMPLETE
Arduino IDE download and setup 1	IW	Apr 22	COMPLETE
Fire sensor Integration with ESP32	IW	Apr 22	COMPLETE
optimal path suggestion	IN		COMPLETE
nearest fire department identification	IN		COMPLETE
Fire department location data collection and data set creation	IN		COMPLETE
fire department resource data collection	IW IN ID IA		COMPLETE
Draw Swimmlane chart			COMPLETE
Create a swimlane diagram			COMPLETE
Web application Wireframes Design 3	ID IA IW IN		COMPLETE
Swimmlane chart design			COMPLETE
Web Application Wireframes Design			COMPLETE
Fire department data collection			COMPLETE

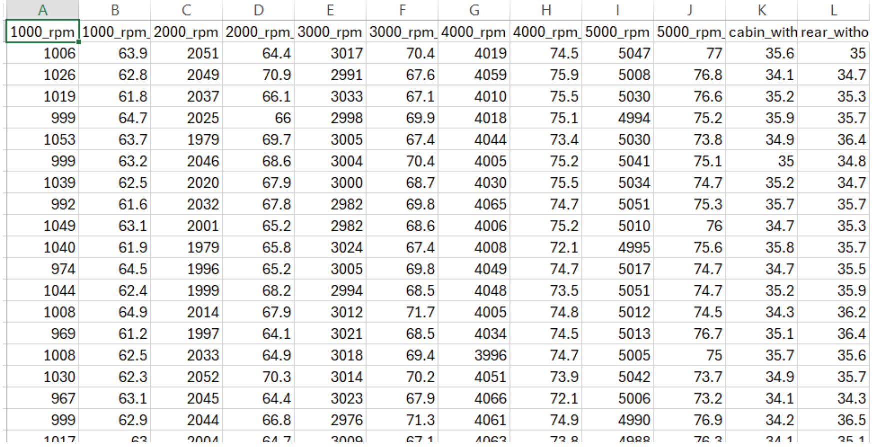
Project Implementation

Data Collection

Collecting data related to temperature and RPM using the thermometer and an RPM gauge.

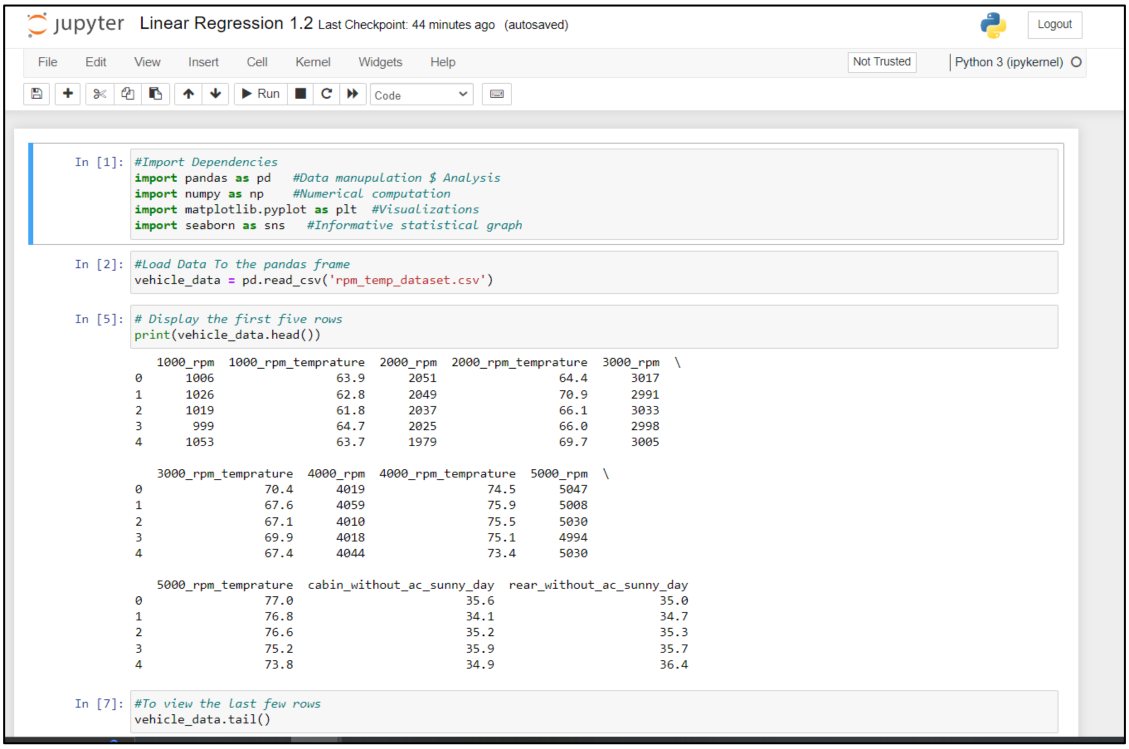
Fire severity Assessment for emergency services.

Data Collection



A	B	C	D	E	F	G	H	I	J	K	L
1000_rpm	1000_rpm_	2000_rpm	2000_rpm_	3000_rpm	3000_rpm_	4000_rpm	4000_rpm_	5000_rpm	5000_rpm_	cabin_with	rear_witho
1006	63.9	2051	64.4	3017	70.4	4019	74.5	5047	77	35.6	35
1026	62.8	2049	70.9	2991	67.6	4059	75.9	5008	76.8	34.1	34.7
1019	61.8	2037	66.1	3033	67.1	4010	75.5	5030	76.6	35.2	35.3
999	64.7	2025	66	2998	69.9	4018	75.1	4994	75.2	35.9	35.7
1053	63.7	1979	69.7	3005	67.4	4044	73.4	5030	73.8	34.9	36.4
999	63.2	2046	68.6	3004	70.4	4005	75.2	5041	75.1	35	34.8
1039	62.5	2020	67.9	3000	68.7	4030	75.5	5034	74.7	35.2	34.7
992	61.6	2032	67.8	2982	69.8	4065	74.7	5051	75.3	35.7	35.7
1049	63.1	2001	65.2	2982	68.6	4006	75.2	5010	76	34.7	35.3
1040	61.9	1979	65.8	3024	67.4	4008	72.1	4995	75.6	35.8	35.7
974	64.5	1996	65.2	3005	69.8	4049	74.7	5017	74.7	34.7	35.5
1044	62.4	1999	68.2	2994	68.5	4048	73.5	5051	74.7	35.2	35.9
1008	64.9	2014	67.9	3012	71.7	4005	74.8	5012	74.5	34.3	36.2
969	61.2	1997	64.1	3021	68.5	4034	74.5	5013	76.7	35.1	36.4
1008	62.5	2033	64.9	3018	69.4	3996	74.7	5005	75	35.7	35.6
1030	62.3	2052	70.3	3014	70.2	4051	73.9	5042	73.7	34.9	35.7
967	63.1	2045	64.4	3023	67.9	4066	72.1	5006	73.2	34.1	34.3
999	62.9	2044	66.8	2976	71.3	4061	74.9	4990	76.9	34.2	36.5
1017	63	2004	64.7	3009	67.1	4063	73.8	4988	76.3	34.1	35.1

Data preprocessing



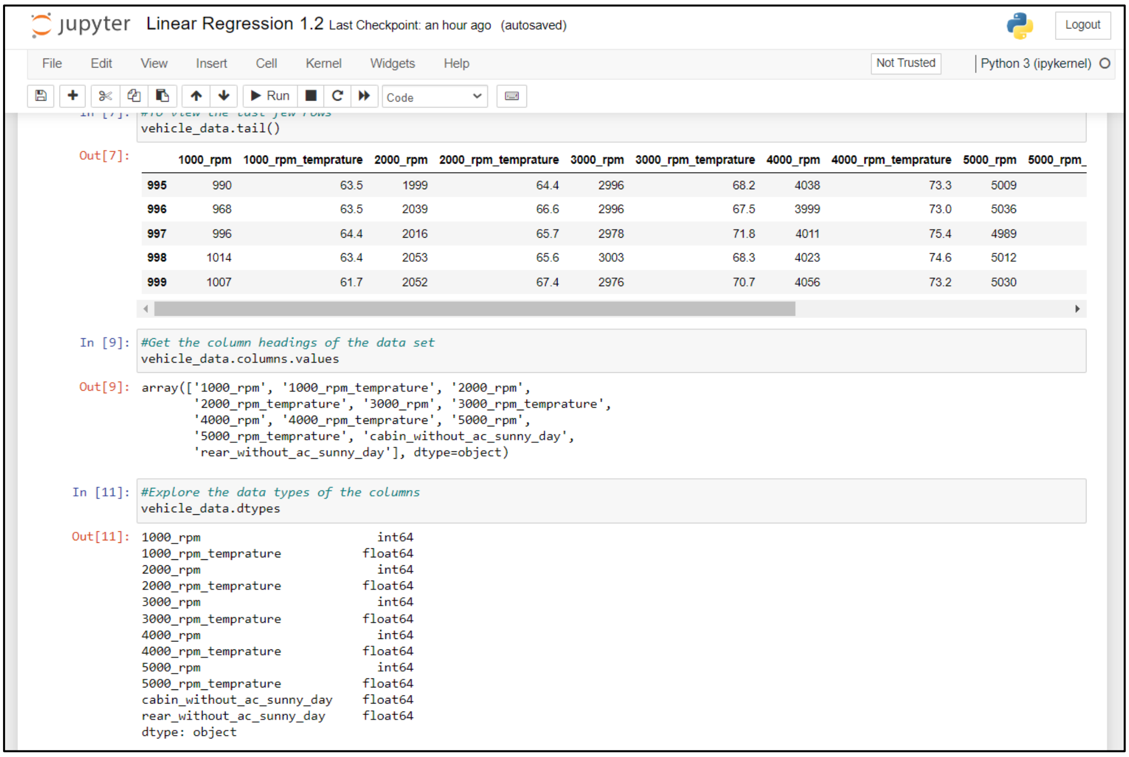
```
In [1]: #Import Dependencies
        import pandas as pd   #Data manupulation $ Analysis
        import numpy as np    #Numerical computation
        import matplotlib.pyplot as plt  #Visualizations
        import seaborn as sns   #Informative statistical graph
```

```
In [2]: #Load Data To the pandas frame
        vehicle_data = pd.read_csv('rpm_temp_dataset.csv')
```

```
In [5]: # Display the first five rows
        print(vehicle_data.head())
```

```
   1000_rpm  1000_rpm_temprature  2000_rpm  2000_rpm_temprature  3000_rpm  \
0      1006                 63.9      2051                 64.4      3017
1      1026                 62.8      2049                 70.9      2991
2      1019                 61.8      2037                 66.1      3033
3       999                 64.7      2025                 66.0      2998
4      1053                 63.7      1979                 69.7      3005

   3000_rpm_temprature  4000_rpm  4000_rpm_temprature  5000_rpm  \
0                 70.4      4019                 74.5      5047
1                 67.6      4059                 75.9      5008
2                 67.1      4010                 75.5      5030
3                 69.9      4018                 75.1      4994
4                 67.4      4044                 73.4      5030

   5000_rpm_temprature  cabin_without_ac_sunny_day  rear_without_ac_sunny_day
0                 77.0                        35.6                       35.0
1                 76.8                        34.1                       34.7
2                 76.6                        35.2                       35.3
3                 75.2                        35.9                       35.7
4                 73.8                        34.9                       36.4
```

```
In [7]: #To view the last few rows
        vehicle_data.tail()
```

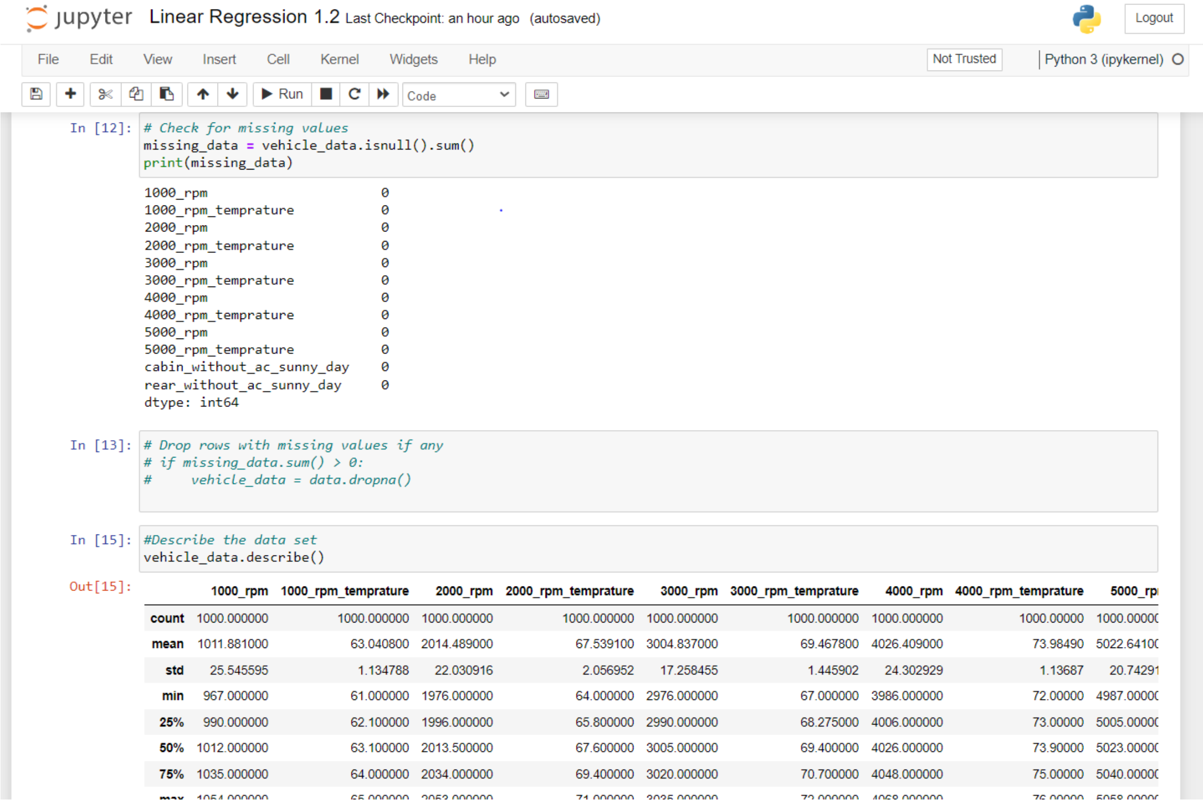


```
In [7]: #To view the last few rows
        vehicle_data.tail()
```

Out[7]:

	1000_rpm	1000_rpm_temprature	2000_rpm	2000_rpm_temprature	3000_rpm	3000_rpm_temprature	4000_rpm	4000_rpm_temprature	5000_rpm	5000_rpm_
995	990	63.5	1999	64.4	2996	68.2	4038	73.3	5009	
996	968	63.5	2039	66.6	2996	67.5	3999	73.0	5036	
997	996	64.4	2016	65.7	2978	71.8	4011	75.4	4989	
998	1014	63.4	2053	65.6	3003	68.3	4023	74.6	5012	
999	1007	61.7	2052	67.4	2976	70.7	4056	73.2	5030	

```
In [9]: #Get the column headings of the data set
        vehicle_data.columns.values

Out[9]: array(['1000_rpm', '1000_rpm_temprature', '2000_rpm',
               '2000_rpm_temprature', '3000_rpm', '3000_rpm_temprature',
               '4000_rpm', '4000_rpm_temprature', '5000_rpm',
               '5000_rpm_temprature', 'cabin_without_ac_sunny_day',
               'rear_without_ac_sunny_day'], dtype=object)

In [11]: #Explore the data types of the columns
         vehicle_data.dtypes

Out[11]: 1000_rpm                        int64
         1000_rpm_temprature           float64
         2000_rpm                        int64
         2000_rpm_temprature           float64
         3000_rpm                        int64
         3000_rpm_temprature           float64
         4000_rpm                        int64
         4000_rpm_temprature           float64
         5000_rpm                        int64
         5000_rpm_temprature           float64
         cabin_without_ac_sunny_day    float64
         rear_without_ac_sunny_day     float64
         dtype: object
```

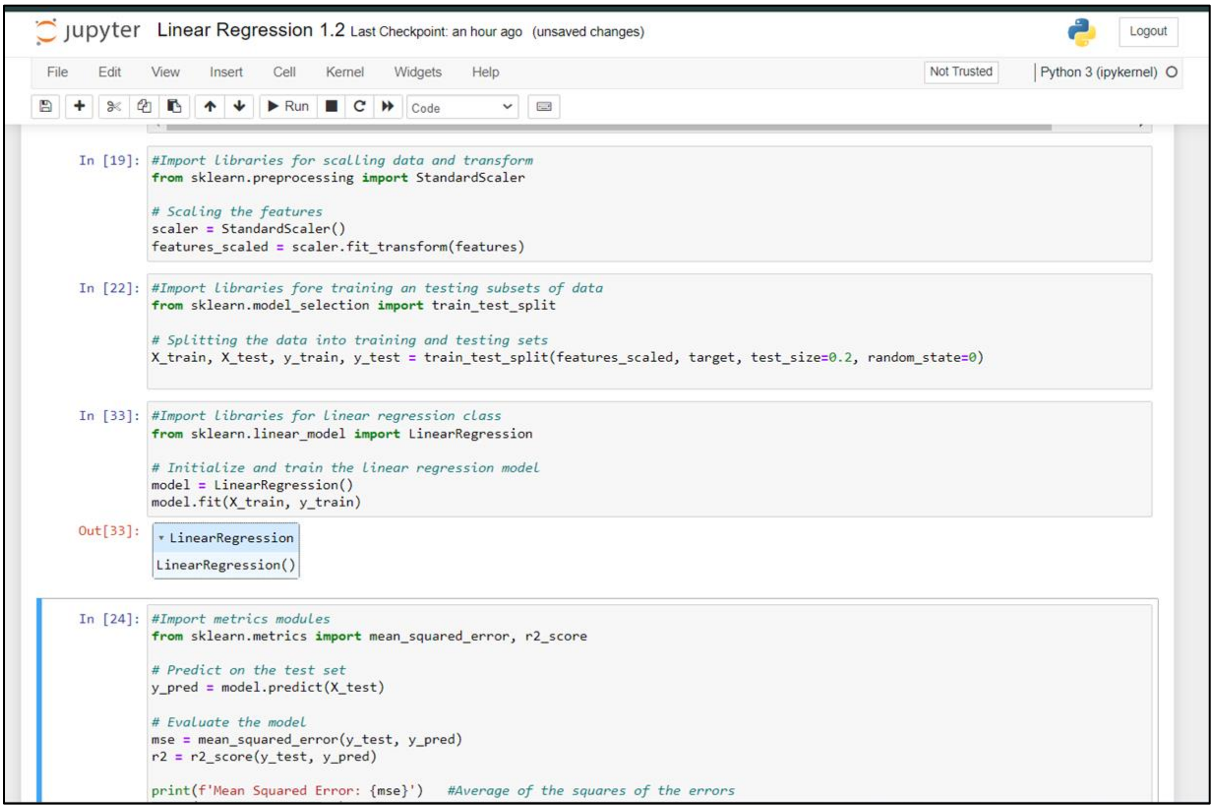


```
In [12]: # Check for missing values
         missing_data = vehicle_data.isnull().sum()
         print(missing_data)

         1000_rpm                      0
         1000_rpm_temprature           0
         2000_rpm                      0
         2000_rpm_temprature           0
         3000_rpm                      0
         3000_rpm_temprature           0
         4000_rpm                      0
         4000_rpm_temprature           0
         5000_rpm                      0
         5000_rpm_temprature           0
         cabin_without_ac_sunny_day    0
         rear_without_ac_sunny_day     0
         dtype: int64

In [13]: # Drop rows with missing values if any
         # if missing_data.sum() > 0:
         #     vehicle_data = data.dropna()

In [15]: #Describe the data set
         vehicle_data.describe()
```

Out[15]:

	1000_rpm	1000_rpm_temprature	2000_rpm	2000_rpm_temprature	3000_rpm	3000_rpm_temprature	4000_rpm	4000_rpm_temprature	5000_rp
count	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.000000	1000.00000	1000.0000
mean	1011.881000	63.040800	2014.489000	67.539100	3004.837000	69.467800	4026.409000	73.98490	5022.6410
std	25.545595	1.134788	22.030916	2.056952	17.258455	1.445902	24.302929	1.13687	20.74291
min	967.000000	61.000000	1976.000000	64.000000	2976.000000	67.000000	3986.000000	72.00000	4987.0000
25%	990.000000	62.100000	1996.000000	65.800000	2990.000000	68.275000	4006.000000	73.00000	5005.0000
50%	1012.000000	63.100000	2013.500000	67.600000	3005.000000	69.400000	4026.000000	73.90000	5023.0000
75%	1035.000000	64.000000	2034.000000	69.400000	3020.000000	70.700000	4048.000000	75.00000	5040.0000
max	1054.000000	65.000000	2053.000000	71.000000	3035.000000	72.000000	4068.000000	76.00000	5058.0000

Model Selection



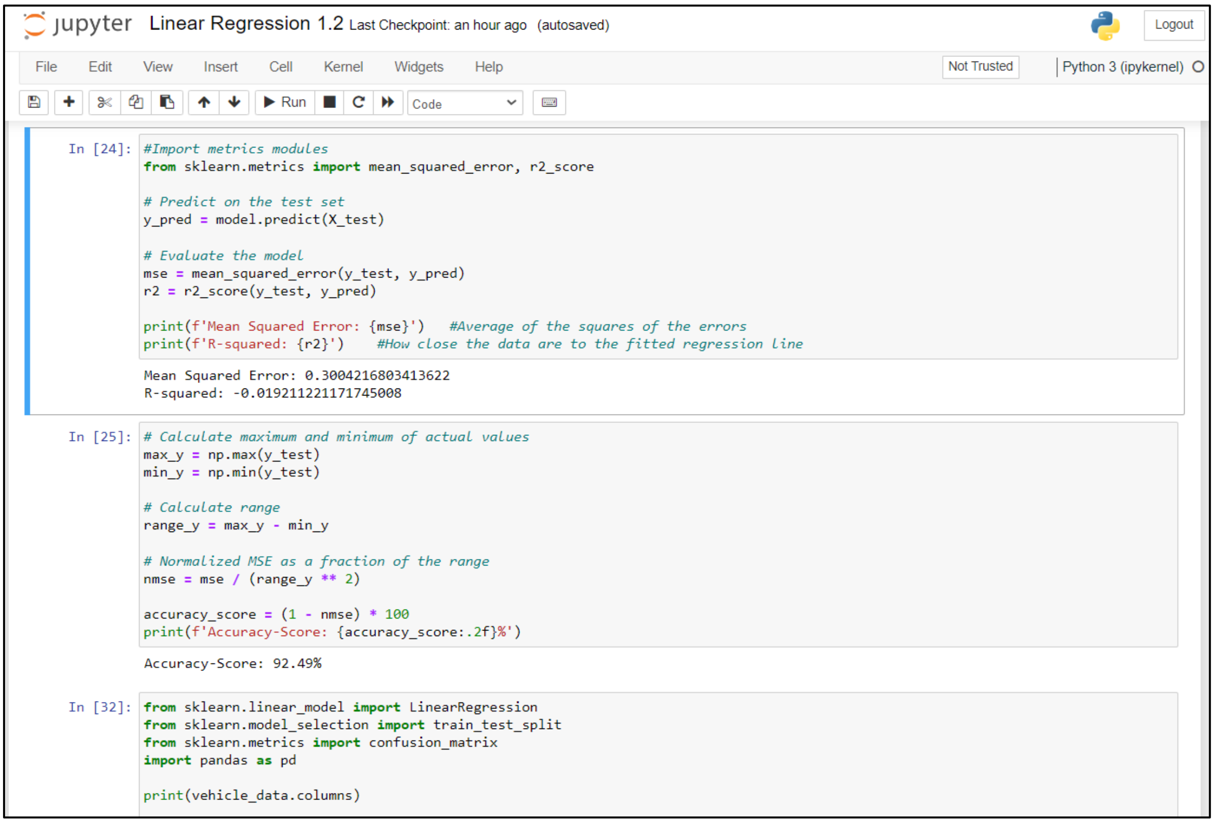
In [19]:
```
#Import libraries for scalling data and transform
from sklearn.preprocessing import StandardScaler

# Scaling the features
scaler = StandardScaler()
features_scaled = scaler.fit_transform(features)
```

In [22]:
```
#Import libraries fore training an testing subsets of data
from sklearn.model_selection import train_test_split

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(features_scaled, target, test_size=0.2, random_state=0)
```

In [33]:
```
#Import libraries for linear regression class
from sklearn.linear_model import LinearRegression

# Initialize and train the linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
```

Out[33]:
```
▾ LinearRegression
LinearRegression()
```

In [24]:
```
#Import metrics modules
from sklearn.metrics import mean_squared_error, r2_score

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')   #Average of the squares of the errors
```



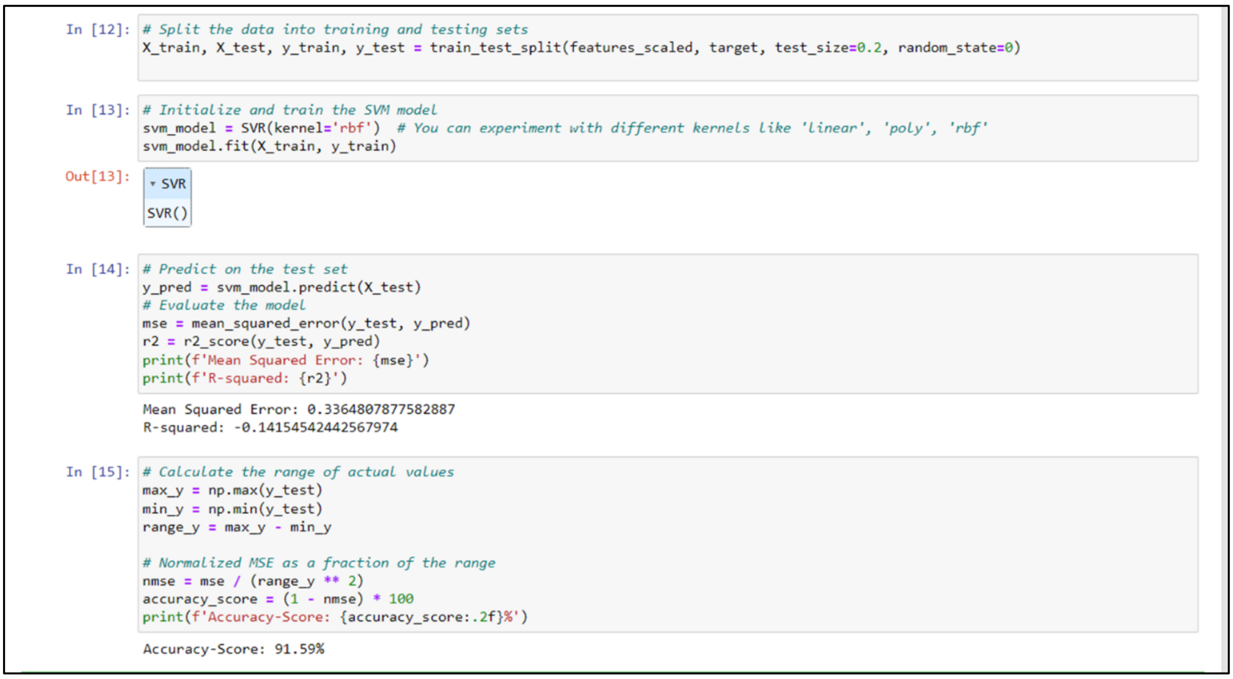
In [24]:
```
#Import metrics modules
from sklearn.metrics import mean_squared_error, r2_score

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f'Mean Squared Error: {mse}')   #Average of the squares of the errors
print(f'R-squared: {r2}')    #How close the data are to the fitted regression line
```

```
Mean Squared Error: 0.3004216803413622
R-squared: -0.019211221171745008
```

In [25]:
```
# Calculate maximum and minimum of actual values
max_y = np.max(y_test)
min_y = np.min(y_test)

# Calculate range
range_y = max_y - min_y

# Normalized MSE as a fraction of the range
nmse = mse / (range_y ** 2)

accuracy_score = (1 - nmse) * 100
print(f'Accuracy-Score: {accuracy_score:.2f}%')
```

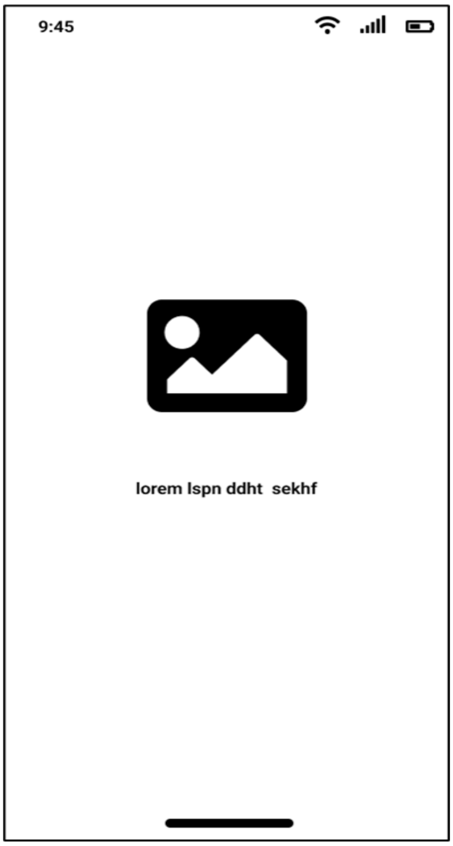
```
Accuracy-Score: 92.49%
```

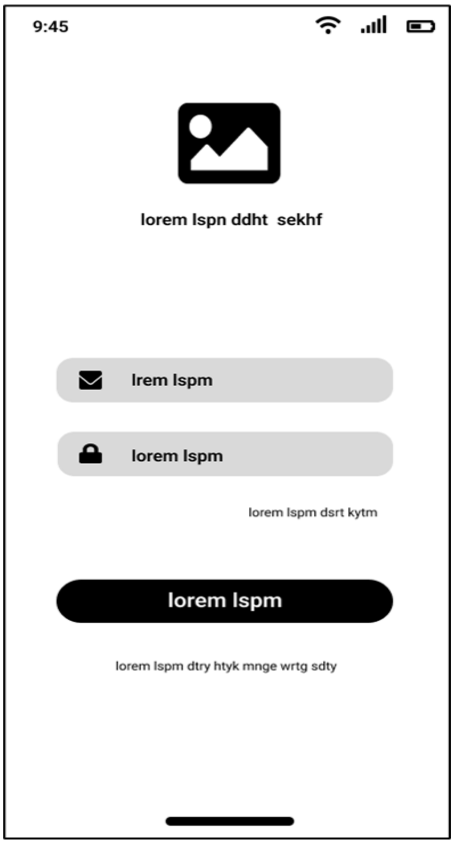
In [32]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import pandas as pd

print(vehicle_data.columns)
```

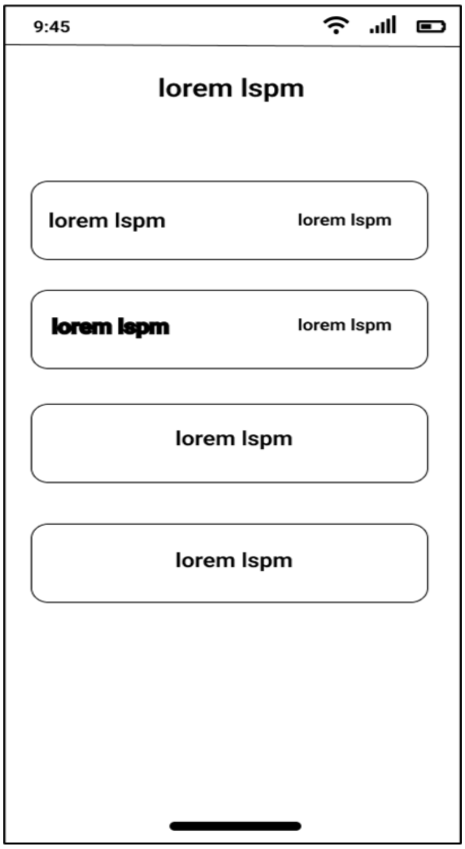


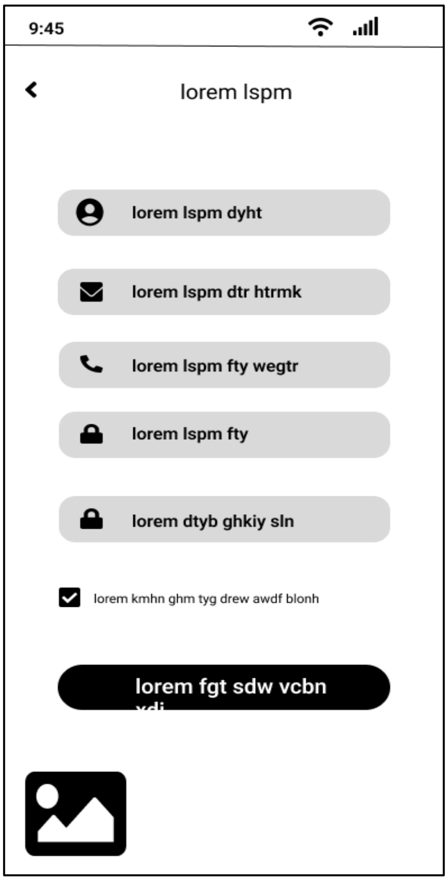
```
In [12]: # Split the data into training and testing sets
         X_train, X_test, y_train, y_test = train_test_split(features_scaled, target, test_size=0.2, random_state=0)

In [13]: # Initialize and train the SVM model
         svm_model = SVR(kernel='rbf')  # You can experiment with different kernels like 'linear', 'poly', 'rbf'
         svm_model.fit(X_train, y_train)

Out[13]: ▾ SVR
         SVR()

In [14]: # Predict on the test set
         y_pred = svm_model.predict(X_test)
         # Evaluate the model
         mse = mean_squared_error(y_test, y_pred)
         r2 = r2_score(y_test, y_pred)
         print(f'Mean Squared Error: {mse}')
         print(f'R-squared: {r2}')

         Mean Squared Error: 0.3364807877582887
         R-squared: -0.14154542442567974

In [15]: # Calculate the range of actual values
         max_y = np.max(y_test)
         min_y = np.min(y_test)
         range_y = max_y - min_y

         # Normalized MSE as a fraction of the range
         nmse = mse / (range_y ** 2)
         accuracy_score = (1 - nmse) * 100
         print(f'Accuracy-Score: {accuracy_score:.2f}%')

         Accuracy-Score: 91.59%
```

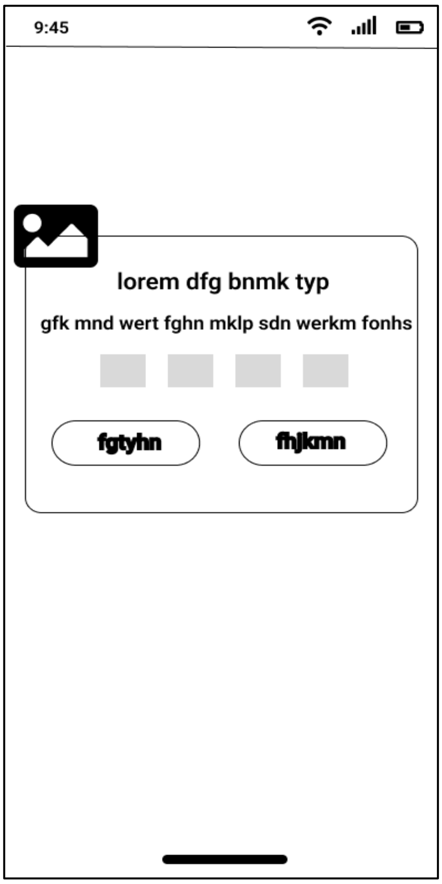
Mobile App Wireframes

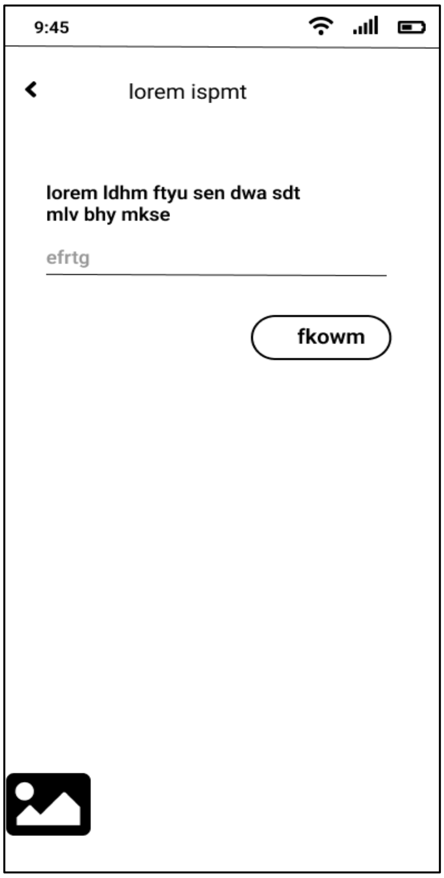


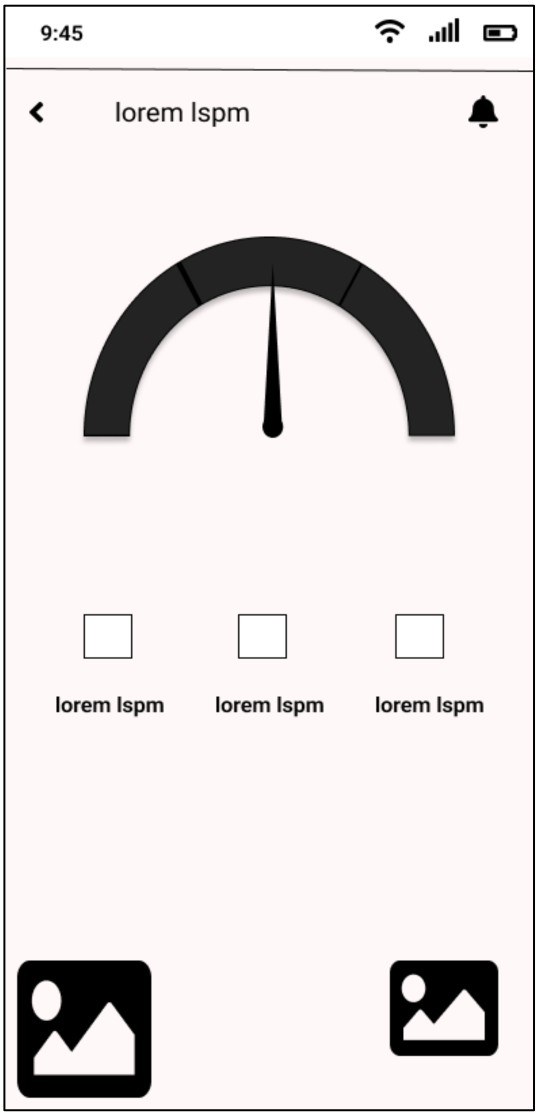


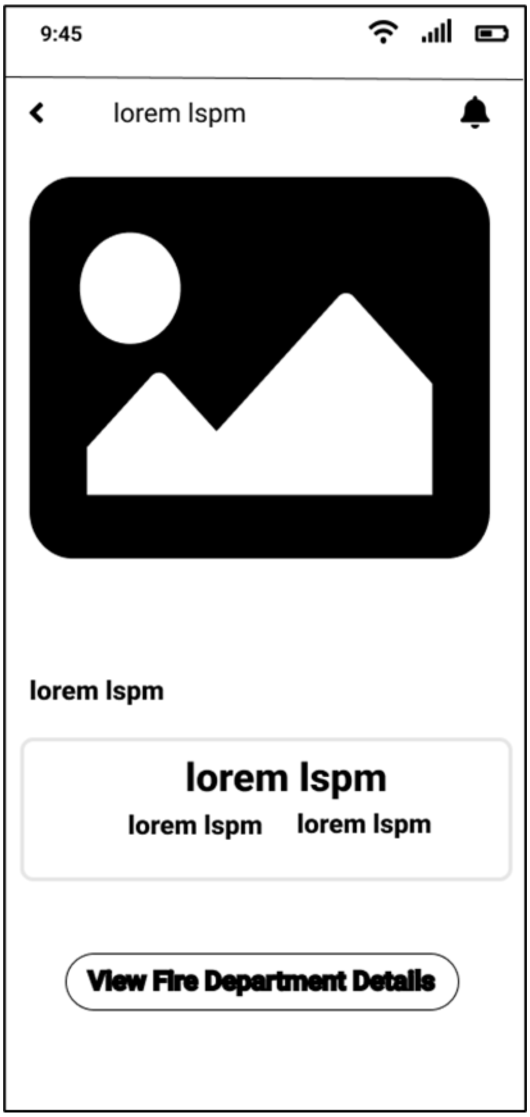




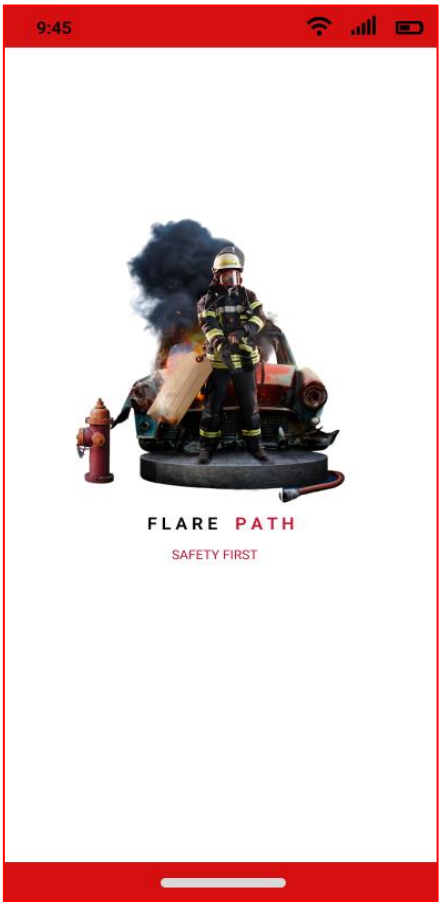


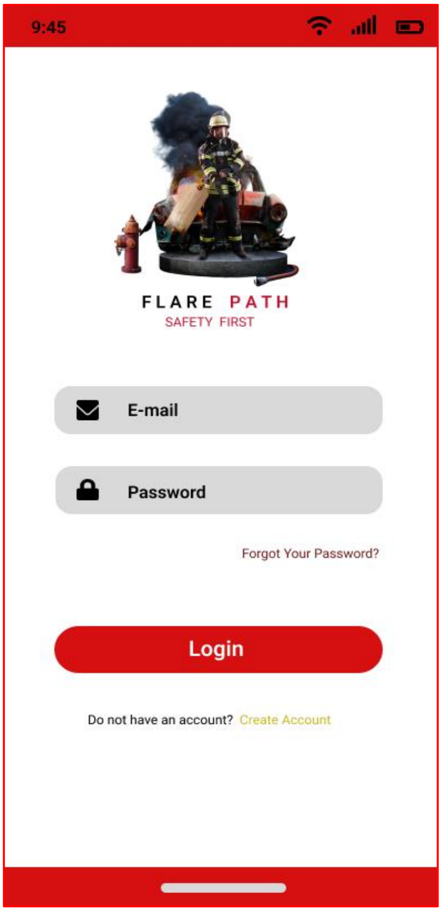



9:45
lorem Ispm
lorem Ispm
lorem Ispm
lorem Ispm

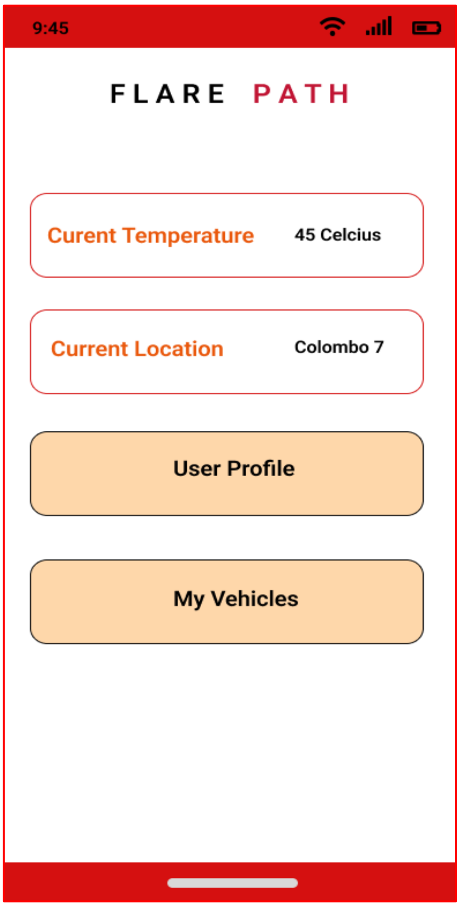

9:45
lorem Ispm
lorem Ispm
lorem Ispm
lorem Ispm
lorem Ispm
View Fire Department Details

Mobile Application UI/UX Design



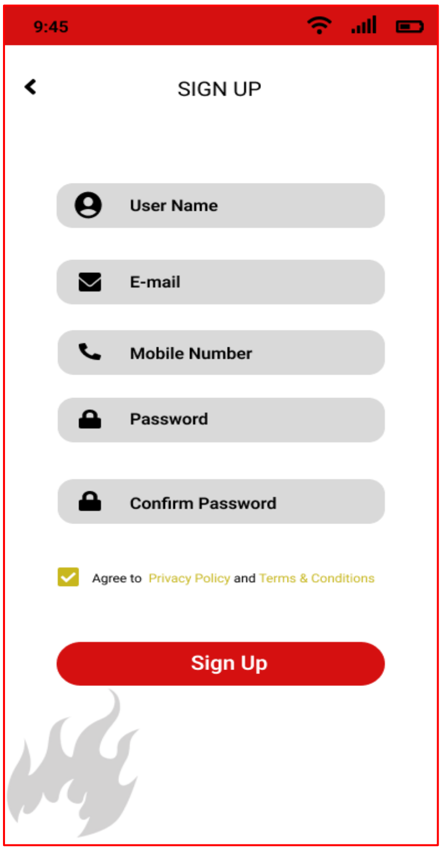


Do not have an account? Create Account

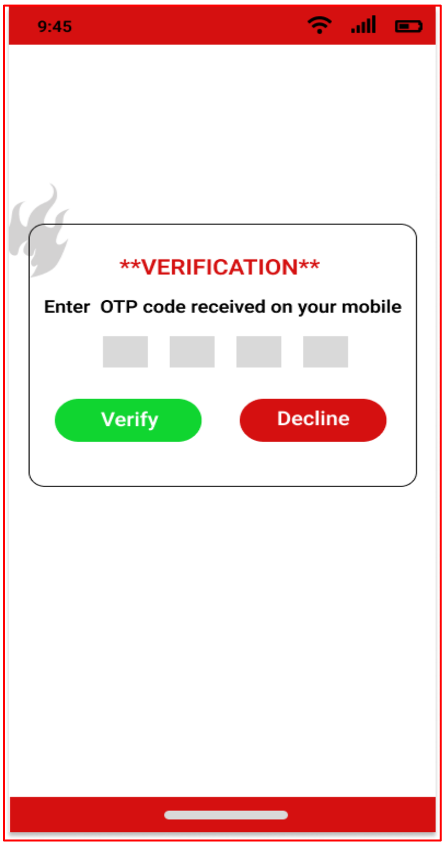


Curent Temperature 45 Celcius

Current Location Colombo 7

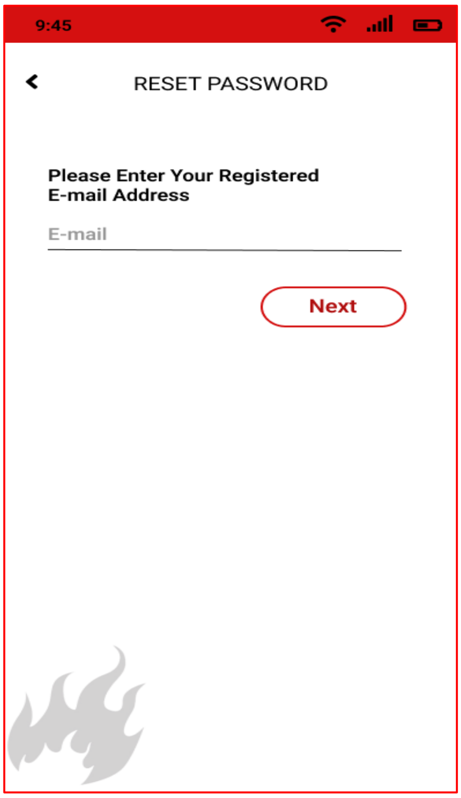


☑ Agree to Privacy Policy and Terms & Conditions



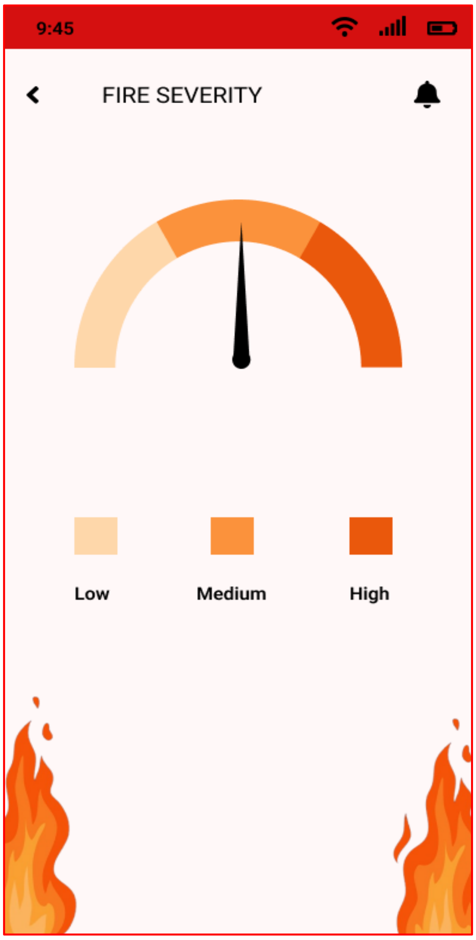
VERIFICATION

Enter OTP code received on your mobile



Please Enter Your Registered E-mail Address

E-mail



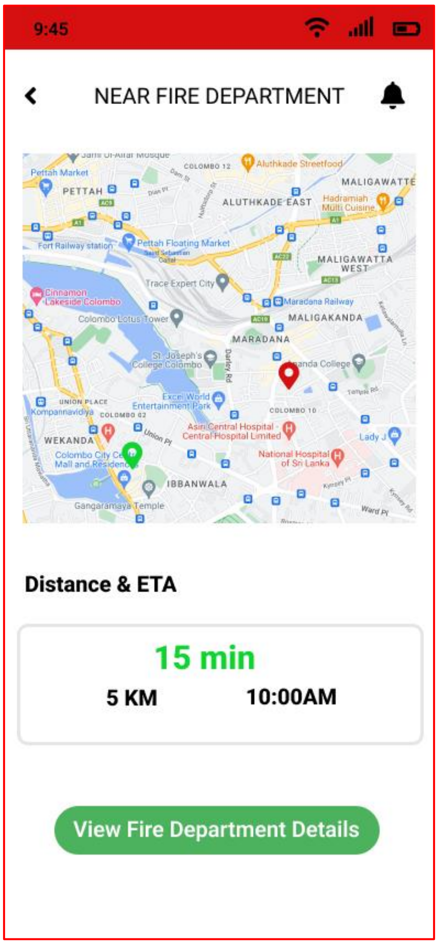
9:45

FIRE SEVERITY

Low

Medium

High



9:45

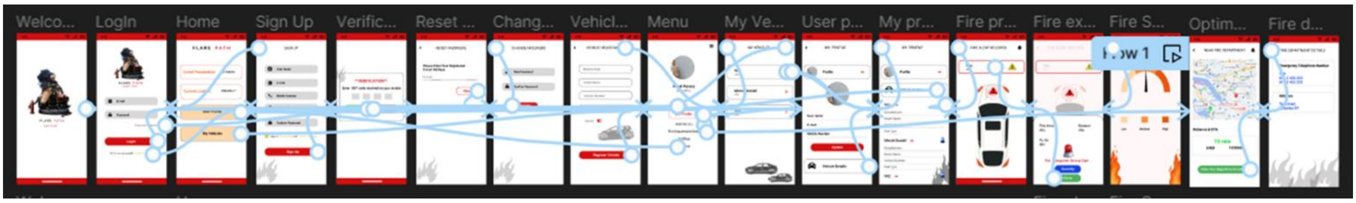
NEAR FIRE DEPARTMENT

Distance & ETA

15 min

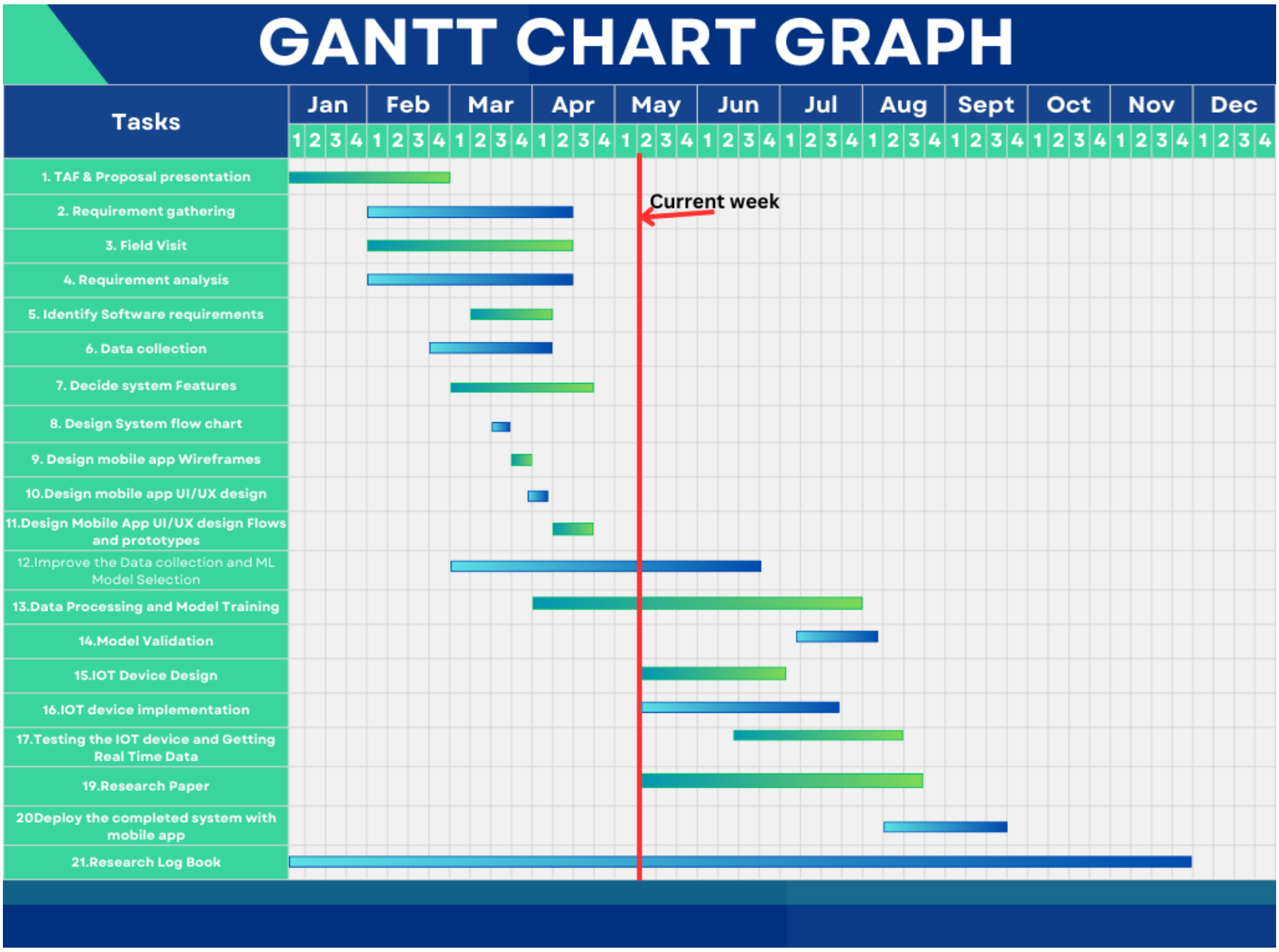
5 KM

10:00AM

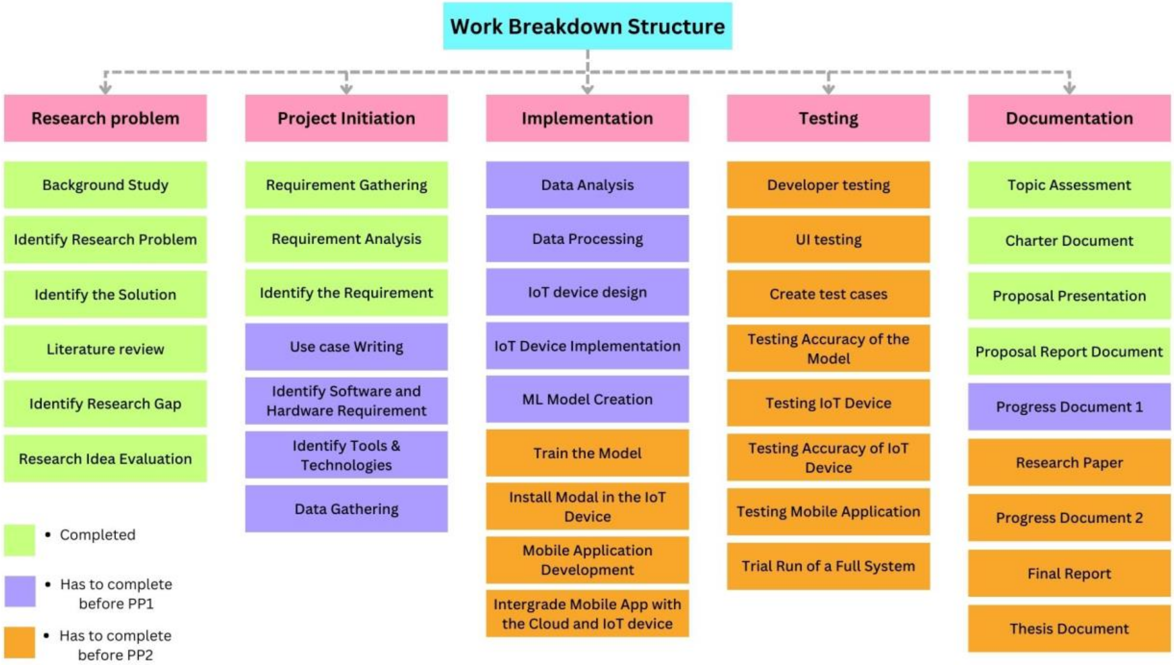


UI/UX Designs Flows

Gantt Chart


GANTT CHART GRAPH
Tasks
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sept
Oct
Nov
Dec
1 2 3 4
1. TAF & Proposal presentation
2. Requirement gathering
3. Field Visit
4. Requirement analysis
5. Identify Software requirements
6. Data collection
7. Decide system Features
8. Design System flow chart
9. Design mobile app Wireframes
10.Design mobile app UI/UX design
11.Design Mobile App UI/UX design Flows and prototypes
12.Improve the Data collection and ML Model Selection
13.Data Processing and Model Training
14.Model Validation
15.IOT Device Design
16.IOT device implementation
17.Testing the IOT device and Getting Real Time Data
19.Research Paper
20Deploy the completed system with mobile app
21.Research Log Book
Current week

Work Breakdown Structure



Work Breakdown Structure

Research problem	Project Initiation	Implementation	Testing	Documentation
Background Study	Requirement Gathering	Data Analysis	Developer testing	Topic Assessment
Identify Research Problem	Requirement Analysis	Data Processing	UI testing	Charter Document
Identify the Solution	Identify the Requirement	IoT device design	Create test cases	Proposal Presentation
Literature review	Use case Writing	IoT Device Implementation	Testing Accuracy of the Model	Proposal Report Document
Identify Research Gap	Identify Software and Hardware Requirement	ML Model Creation	Testing IoT Device	Progress Document 1
Research Idea Evaluation	Identify Tools & Technologies	Train the Model	Testing Accuracy of IoT Device	Research Paper
	Data Gathering	Install Modal in the IoT Device	Testing Mobile Application	Progress Document 2
		Mobile Application Development	Trial Run of a Full System	Final Report
		Intergrade Mobile App with the Cloud and IoT device		Thesis Document

- 🟩 Completed
- 🟪 Has to complete before PP1
- 🟧 Has to complete before PP2