

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

R24-058

Status Document-2



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

September 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

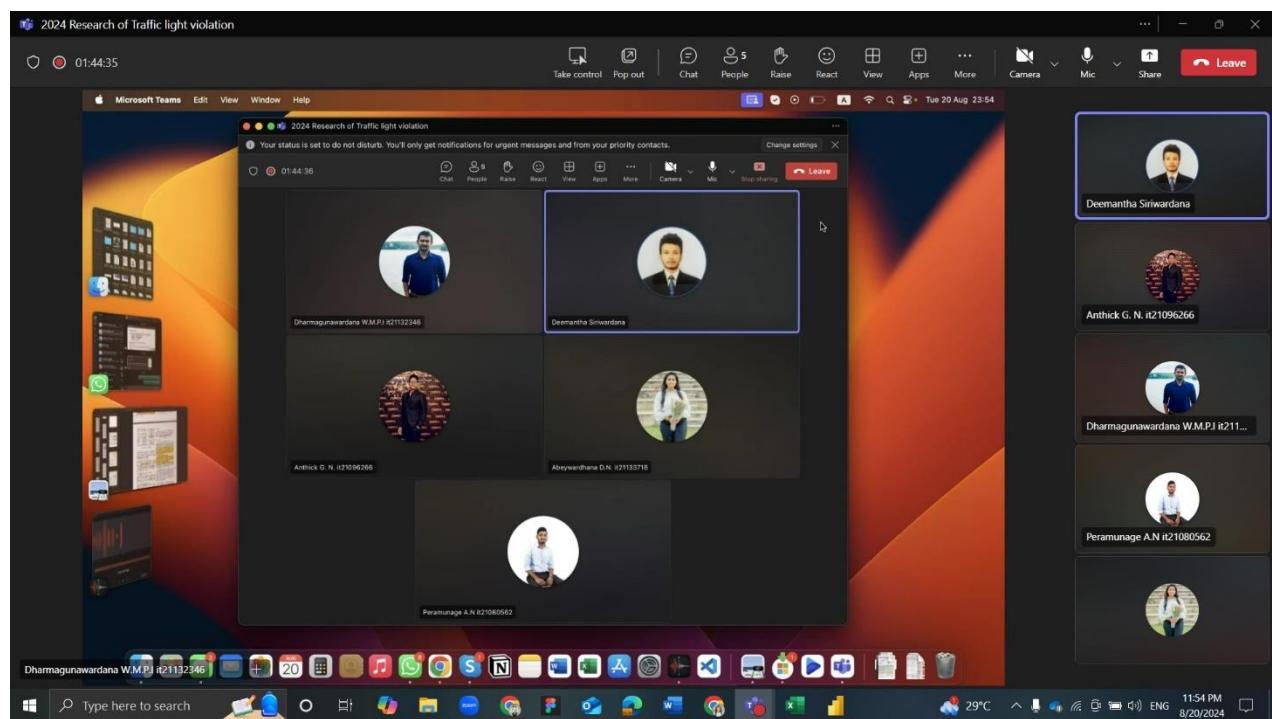
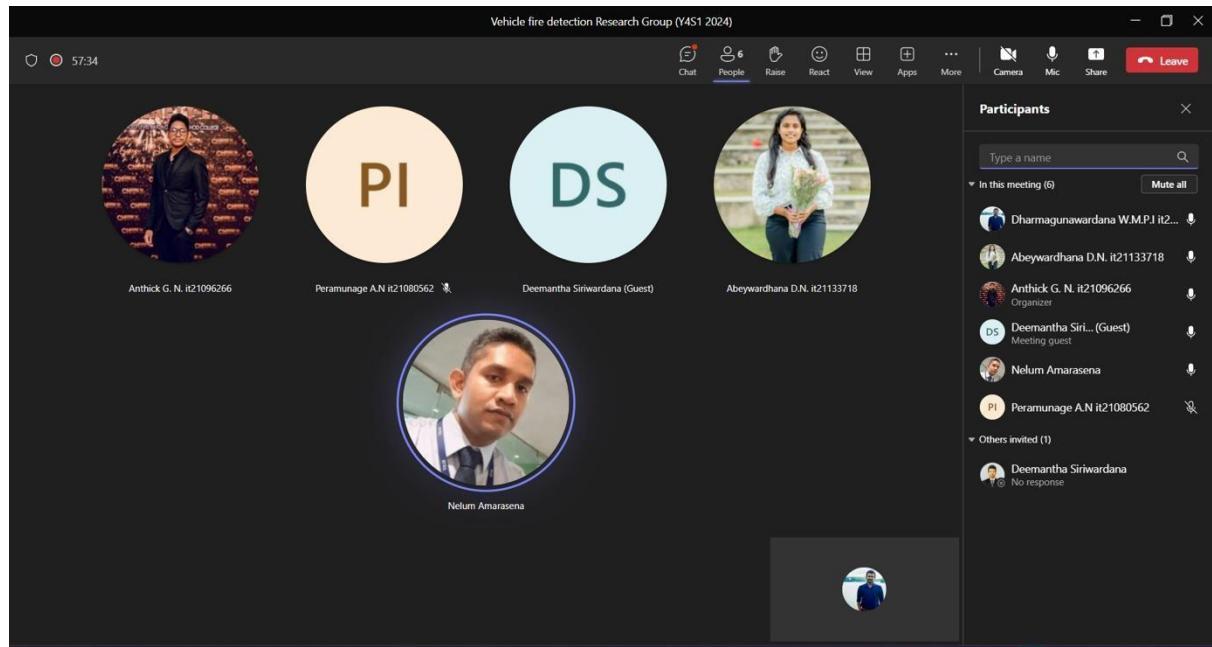
Table of Contents

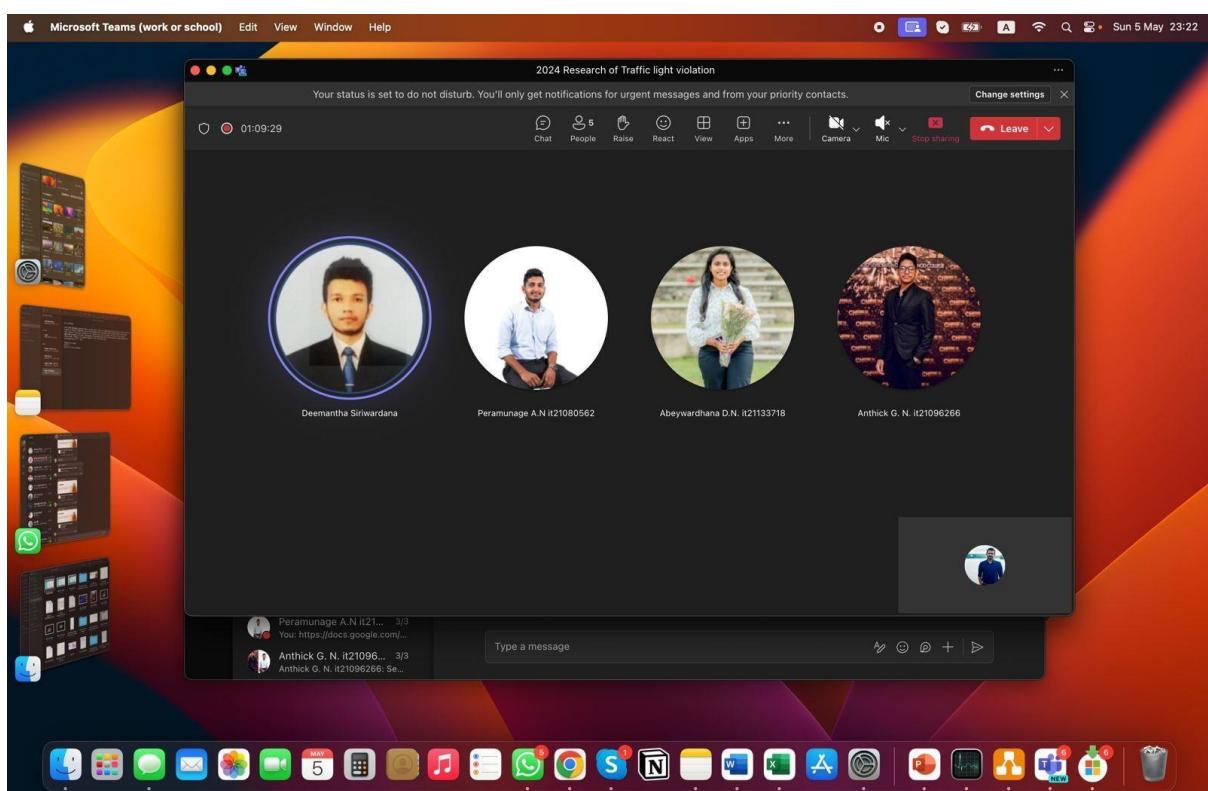
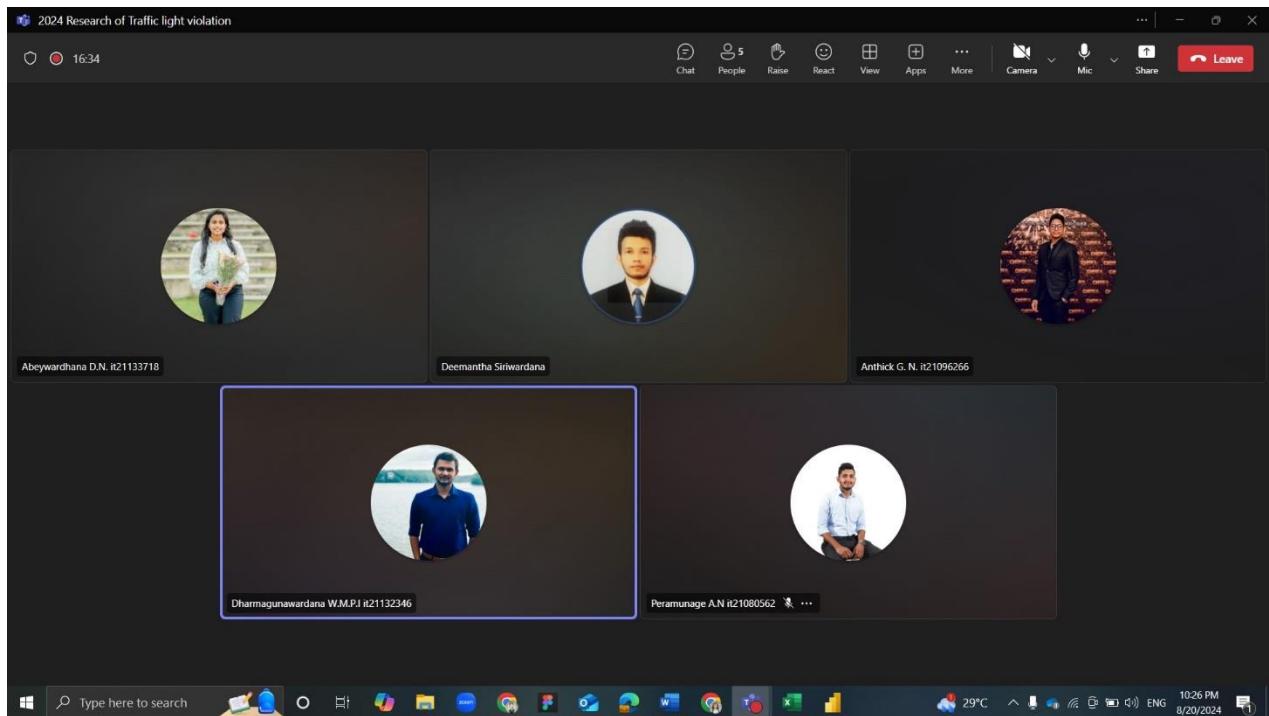
Group Details	2
Meetings & Calls	4
Meetings with supervisor and co-supervisor	4
Meetings with Domain Experts	8
Snapshots from Field Visit	9
Clickup Tasks Allocation	11
ClickUp Dashboard	11
In Progress Tasks	12
Completed Tasks upto PP2	12
Project Implementation	13
Data Collection	13
Fire Severity Assessment for emergency services.....	14
Data Collection & Data Preprocessing.....	14
Model Selection.....	16
Mobile App Wireframes.....	18
Mobile App UI/UX Designs.....	20
Work Breakdown Structure	23

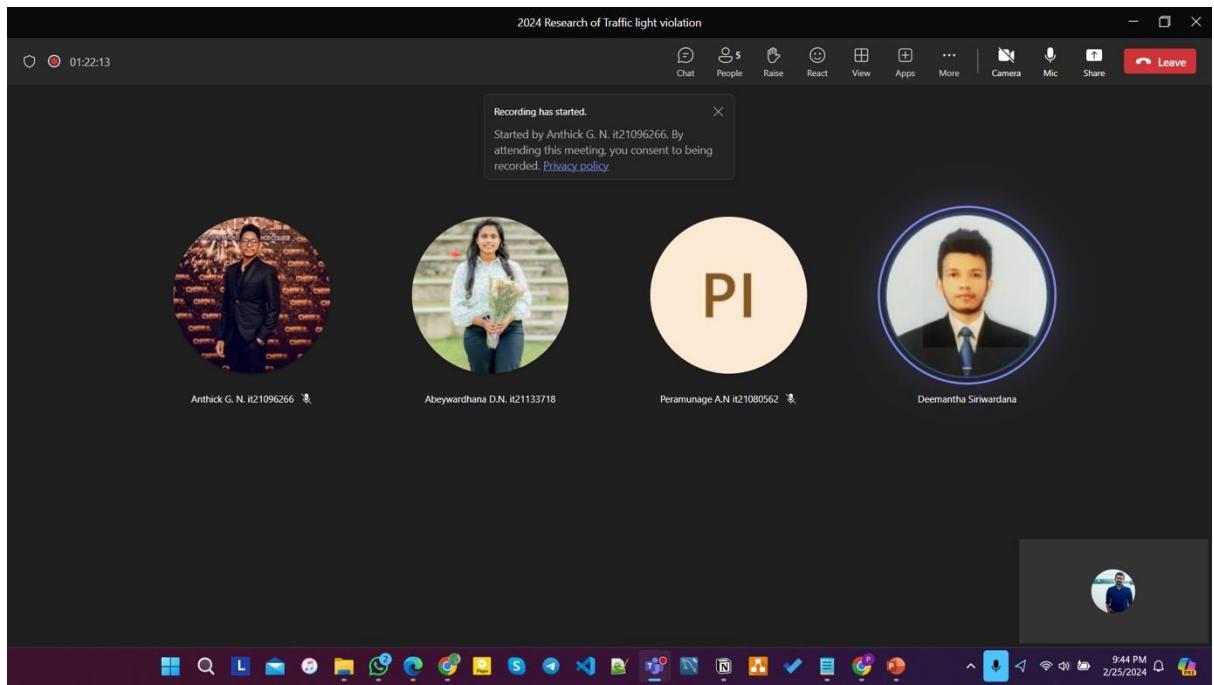
Meetings & Calls

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.







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 issue in that, since we have to submit TAF on 11th this month

You're invited to Co-Supervisor meeting
Sunday, 7 January 2024
9:30pm - 10:30pm (GMT+5:30)
https://teams.microsoft.com/l/meetup-join/19%3ameeting_MDc3YmE1MmtNDOxYS00MzEyLTkYjgtMiVmNGI3MzY4NjQy%40thread.v2/0?context=%7b%22Id%22%3a%2244e3cf94-19c9-4e32-96c3-14f5bf01391a%22%2c%22Oid%22%3a%22b3aeefc1cb656-4424-ad90-3b64631fa17e%22%7d
Tap on the link or paste it in a browser to join.

Join conversation

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...

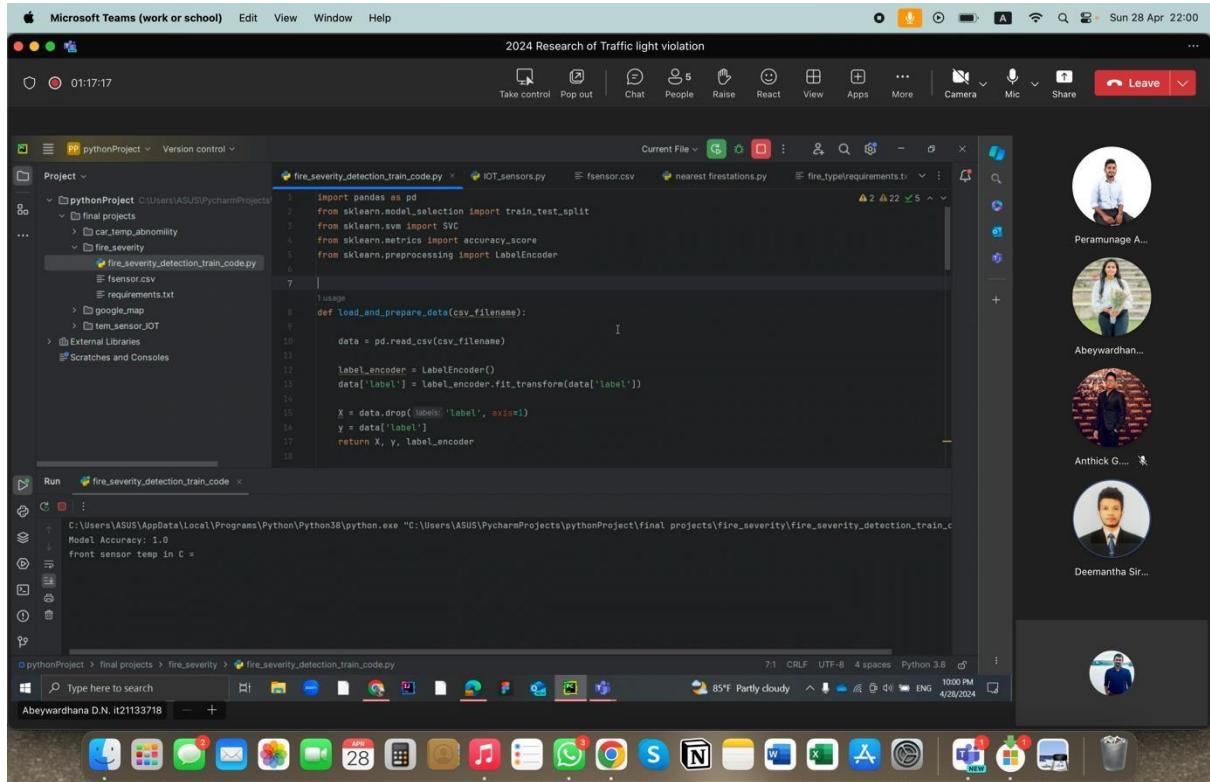
11m 53s

This recording is set to expire. View or change the expiration date [here](#). [Learn more](#)

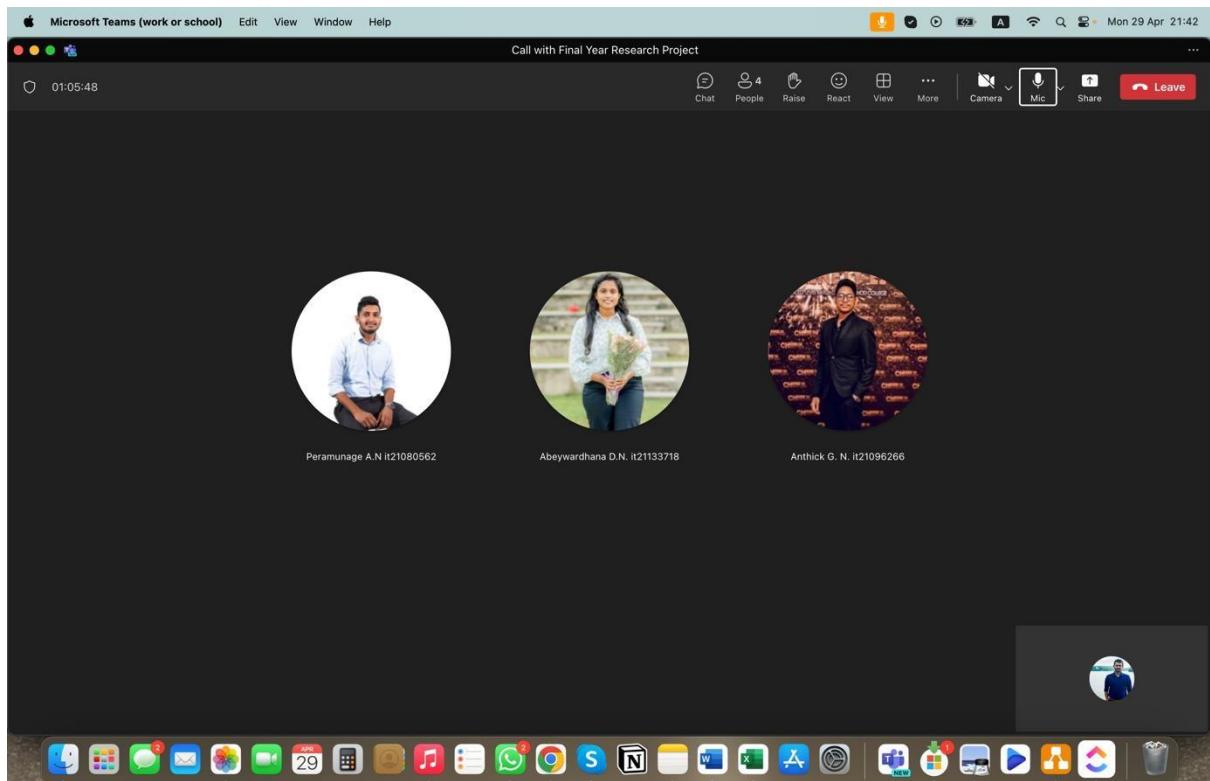
1/16 9:48 PM Meeting ended: 14s

Type a message

Code review with co-supervisor.



Group meeting with group members.



Presentation review with supervisor



Meet Co-supervisor before the demonstration

A screenshot of a Microsoft Teams meeting window. The title bar reads "2024 Research of Traffic light violation". The interface includes standard Microsoft Teams controls like Chat, People, Raise, React, View, Apps, More, Camera, Mic, and Share, along with a "Leave" button. The main area displays a collage of photographs under the heading "SNAPS FROM THE FIELD VISITS". The collage includes images of people in various settings, such as a group in an office, individuals interacting with a vehicle, and a fire truck. On the right side, there is a sidebar with circular profile pictures and names of participants: Deemantha Siri..., Abeywardhana ..., Anithick G. N. it..., and Peramunage A.... At the bottom left, the SLIIT Faculty of Computing logo and the text "Dharmagunawardana W.M.P.I it21132346" are visible. The bottom right corner shows the number "5" and the date "5/5/24".

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
(Communication)

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





Click up Tasks Allocation

ClickUp - All Folders | Research - 058 (Board)

Q Search... HK Ask AI

New | 🌟 | 📁 | 🗃 | 🕒 | 📄 | 🚫 | 🎯 | 📡 | 📈

Share | Automations

R Research - 0... R Research 2024-058 ...

Home | Inbox | Docs | Dashboards | Clips | Timesheets | More

Favorites >

Spaces

Everything

R Research 2024-058 ... +

- Glen IT21096266
- Aveen IT21080562
- Dewmini IT21133718
- Pasindu IT211323... ... +

List 7

View all Spaces + Create Space

Overview List Board Calendar Gantt Team + View

Group: Status Subtasks: Collapse all Filters Sort Me mode Assignees Show closed Hide

Search tasks...

TO DO 0

+ Add task

PLANNING 4

+ Add task

IN PROGRESS 11

+ Add task

IMPLEMENT THE FIRE SENSOR

In List

PLANNING

CREATE A MOBILE APP

In List

IN PROGRESS

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

Fire Severity Data Set Creation

In List

IN PROGRESS

MOBILE APP DESIGN

In List

IN PROGRESS

Train The Model

In List

IN PROGRESS

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

Arduino IDE Download and Setup

In List

COMPLETE

Fire Sensor Integration with ESP32

In List

COMPLETE

Fire Department Location Data Collection and Data Set Creation

In List

COMPLETE

High

3 collapsed

Uploaded a Blinking Program to the ESP32

In List

COMPLETE

+ Add Task

Click Up Dashboard

ClickUp - Dashboard | Research - 058

Search... Ask All New Edit mode: Refreshed 1 min ago Auto refresh: On Filter Add card

R Research - 0... Dashboards / Dashboard Share

Home Inbox Docs Dashboards Clips Timesheets More

Favorites > Spaces

Everything Research 2024 - 058 Glen IT21096266 Aveen IT21080562 Dewmini IT21133718 Pasindu IT211323... List View all Spaces Create Space

Invite

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 In Prog... Completed Tasks Completed This Week

10 tasks 15 tasks in progress 13 tasks completed

Total Tasks by Assignee

Assignee	Percentage
Unassigned	44.11%
IIT21133718 Abeywardhana D.N.	8.82%
IIT21080562 Paramunge A.N.	8.82%
IIT21096266 Anthick G. N.	17.64%
IIT21132346 Dharma...	20%

Open Tasks by Assignee

Assignee	Tasks
IIT21132346 Dharma...	20
IIT21096266 Anthick G. N.	17.64
IIT21080562 Paramunge A.N.	8.82
IIT21133718 Abeywardhana D.N.	8.82
Unassigned	10

No Results

Latest Activity Today

Fire severity model creation

In Progress Tasks

The screenshot shows a ClickUp dashboard titled "In Progress" with 15 tasks listed. The tasks are:

- Implement the Fire sensor
- create a circuit Diagram
- create a mobile app [v_2]
- Fire severity data set creation
- mobile app Design [v_2]
- Train the model
- Fire prediction data set creation [v_1]
- fire Prediction model
- fire prediction data collection
- fire department resource allocation prediction with the severity of the fire.
- Web Application Wireframes Design
- Add few more sensors and increase the accuracy
- fire prediction monitoring IOT device [v_1]
- Mobile Application Design
- Fire severity model creation

The tasks are displayed in a table with columns for Assignee, Due Date, and Status. Most tasks are in the "IN PROGRESS" status, while some are "PLANNING".

Completed Tasks up to PP2

The screenshot shows a ClickUp dashboard titled "Completed" with 13 tasks listed. The tasks are:

- Uploaded a Blinking program to the ESP32
- Arduino IDE download and setup [v_1]
- Fire sensor Integration with ESP32
- optimal path suggestion
- nearest fire department identification
- Fire department location data collection and data set creation
- fire department resource data collection
- Draw Swimlane chart
- Create a swimlane diagram
- Web application Wireframes Design [v_3]
- Swimlane chart design
- Web Application Wireframes Design
- Fire department data collection

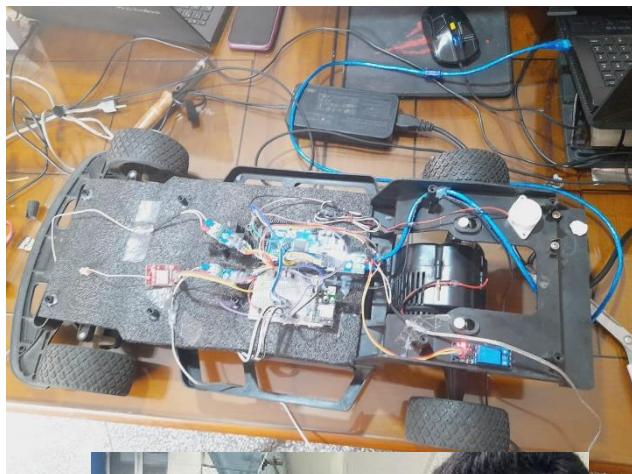
The tasks are displayed in a table with columns for Assignee, Due Date, and Status. All tasks are marked as "COMPLETE".

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	3000	67.1	4063	73.8	4988	76.3	34.1	35.1

The screenshot shows a code editor interface with the following details:

- File Explorer:** Shows files in the FIRE_TYPE directory, including d4.csv, d2.csv, d3.csv, d4.csv, fire_severity.csv, fire_type_detection_main.py, fire_type_detection_train.py, and requirements.txt.
- Code Editor:** The main pane displays the content of d4.csv as a text file. The file contains data points with columns for sensor_front, sensor_mid, sensor_back, and label. The labels include "normal", "front", "back", and "mid".
- Status Bar:** ShowsLn 1, Col 1, Spaces: 4, UTF-8, LF, Plain Text, Go Live, tabnine Starting..., Prettier, 1042 PM, ENG, 9/11/2024.
- Bottom Bar:** Shows various system icons and the taskbar.
- Message:** A floating message box asks if the user wants to install the 'Rainbow CSV' extension from mechatroner for d4.csv, with 'Install' and 'Show Recommendations' buttons.

Data preprocessing

jupyter Linear Regression 1.2 Last Checkpoint: 44 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel) O

In [1]: #Import Dependencies
import pandas as pd #Data manipulation & 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_temperature	2000_rpm	2000_rpm_temperature	3000_rpm	3000_rpm_temperature	4000_rpm	4000_rpm_temperature	5000_rpm	5000_rpm_temperature	
0	1006	63.9	2051	64.4	3017	70.4	4019	74.5	5047	77.0	35.6
1	1026	62.8	2049	70.9	2991	67.6	4059	75.9	5008	76.8	34.1
2	1019	61.8	2037	66.1	3033	67.1	4010	75.5	5030	76.6	35.2
3	999	64.7	2025	66.0	2998	69.9	4018	75.1	4994	75.2	35.9
4	1053	63.7	1979	69.7	3005	67.4	4044	73.4	5030	73.8	34.9

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

jupyter Linear Regression 1.2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel) O

In [7]: vehicle_data.tail()

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

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

Out[9]: array(['1000_rpm', '1000_rpm_temperature', '2000_rpm',
'2000_rpm_temperature', '3000_rpm', '3000_rpm_temperature',
'4000_rpm', '4000_rpm_temperature', '5000_rpm',
'5000_rpm_temperature', '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_temperature float64
2000_rpm int64
2000_rpm_temperature float64
3000_rpm int64
3000_rpm_temperature float64
4000_rpm int64
4000_rpm_temperature float64
5000_rpm int64
5000_rpm_temperature float64
cabin_without_ac_sunny_day float64
rear_without_ac_sunny_day float64
dtype: object

jupyter Linear Regression 1.2 Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (ipykernel) O

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

1000_rpm          0
1000_rpm_tempreature 0
2000_rpm          0
2000_rpm_tempreature 0
3000_rpm          0
3000_rpm_tempreature 0
4000_rpm          0
4000_rpm_tempreature 0
5000_rpm          0
5000_rpm_tempreature 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_tempreature  2000_rpm  2000_rpm_tempreature  3000_rpm  3000_rpm_tempreature  4000_rpm  4000_rpm_tempreature  5000_rpm
count    1000.000000        1000.000000  1000.000000        1000.000000  1000.000000        1000.000000  1000.000000        1000.000000
mean    1011.881000       63.040800 2014.489000       67.539100 3004.837000       69.467800 4026.409000       73.98490 5022.64100
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 4987.00000
25%     990.000000      62.100000 1996.000000      65.800000 2990.000000      68.275000 4006.000000      73.00000 5005.00000
50%     1012.000000      63.100000 2013.500000      67.600000 3005.000000      69.400000 4026.000000      73.90000 5023.00000
75%     1035.000000      64.000000 2034.000000      69.400000 3020.000000      70.700000 4048.000000      75.00000 5040.00000
max    1051.000000      65.000000 2055.000000      71.000000 3095.000000      77.000000 4060.000000      76.00000 5060.00000
```

Model

jupyter Linear Regression 1.2 Last Checkpoint: an hour ago (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help

Not Trusted Python 3 (ipykernel) O

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

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

In [22]: #Import Libraries for training and 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
```

The screenshot shows a Jupyter Notebook interface with two sections of code cells.

Section 1: Linear Regression

```

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)

```

Section 2: SVM Model

```

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()

```



```

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

```



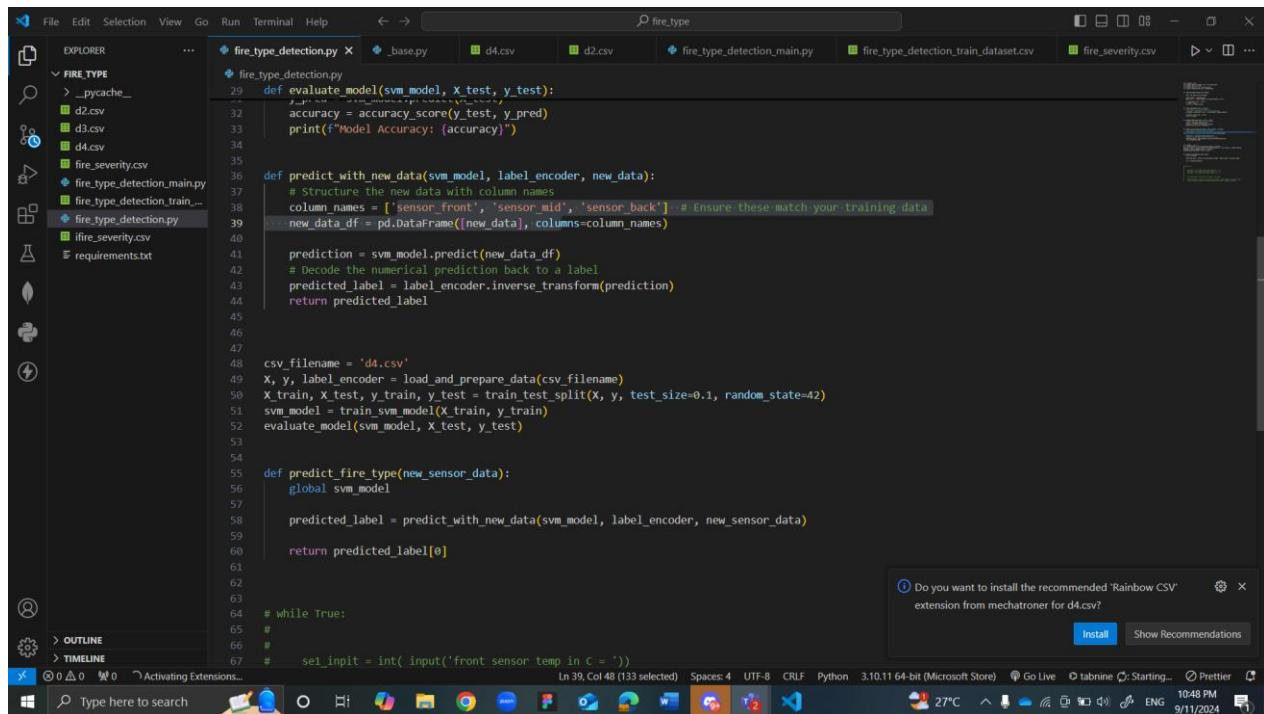
```

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%

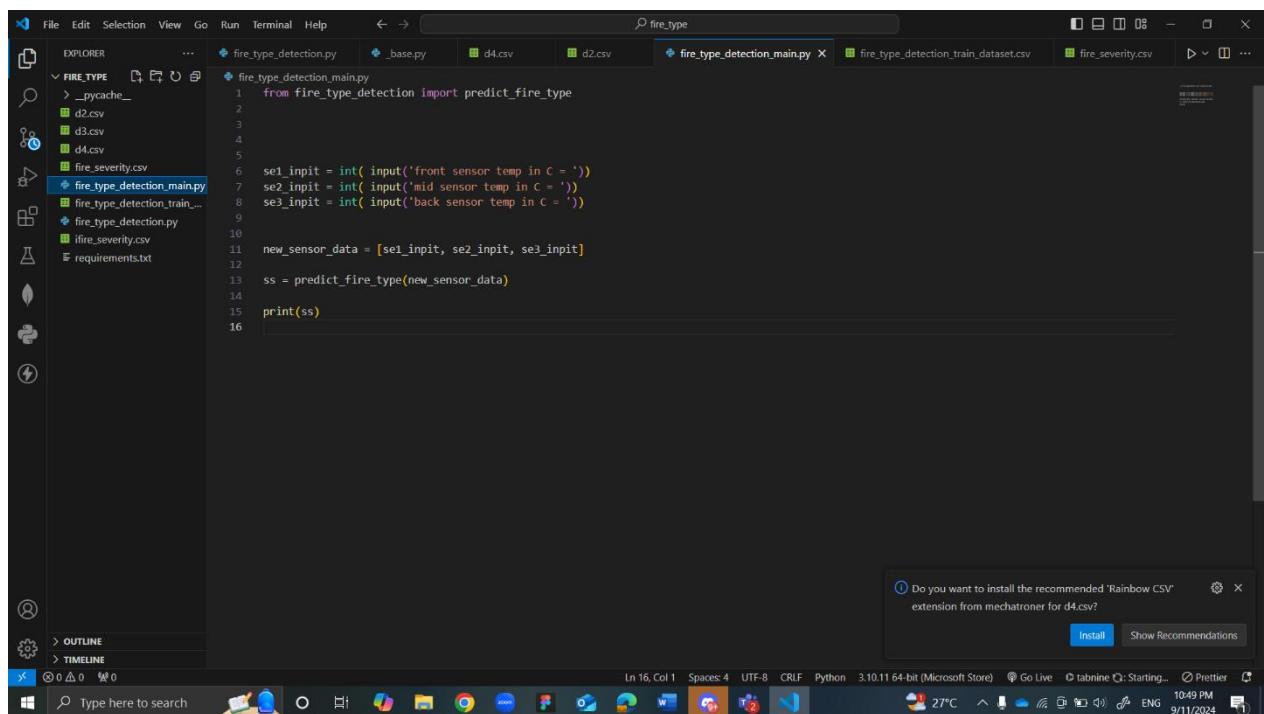
```



```
File Edit Selection View Go Run Terminal Help fire_type_detection.py _base.py d4.csv d2.csv fire_type_detection_main.py fire_type_detection_train_dataset.csv fire_severity.csv

29 def evaluate_model(svm_model, X_test, y_test):
30     accuracy = accuracy_score(y_test, y_pred)
31     print(f"Model Accuracy: {accuracy}")
32
33 def predict_with_new_data(svm_model, label_encoder, new_data):
34     # Structure the new data with column names
35     column_names = ['sensor_front', 'sensor_mid', 'sensor_back'] # Ensure these match your training data
36     new_data_df = pd.DataFrame([new_data], columns=column_names)
37
38     prediction = svm_model.predict(new_data_df)
39     # Decode the numerical prediction back to a label
40     predicted_label = label_encoder.inverse_transform(prediction)
41
42     return predicted_label
43
44
45
46
47 csv_filename = 'd4.csv'
48 X, y, label_encoder = load_and_prepare_data(csv_filename)
49 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.1, random_state=42)
50 svm_model = train_svm_model(X_train, y_train)
51 evaluate_model(svm_model, X_test, y_test)
52
53
54 def predict_fire_type(new_sensor_data):
55     global svm_model
56
57     predicted_label = predict_with_new_data(svm_model, label_encoder, new_sensor_data)
58
59     return predicted_label[0]
60
61
62
63
64 # While True:
65 #
66 #     se1_inpit = int(input('front sensor temp in C = '))
67 #     se2_inpit = int(input('mid sensor temp in C = '))
68 #     se3_inpit = int(input('back sensor temp in C = '))
69
70     new_sensor_data = [se1_inpit, se2_inpit, se3_inpit]
71
72     ss = predict_fire_type(new_sensor_data)
73
74     print(ss)

Ln 39, Col 48 (133 selected) Spaces: 4 UTF-8 CRLF Python 3.10.11 64-bit (Microsoft Store) Go Live tabnine Starting... Prettier 27°C 10:48 PM ENG 9/11/2024
```

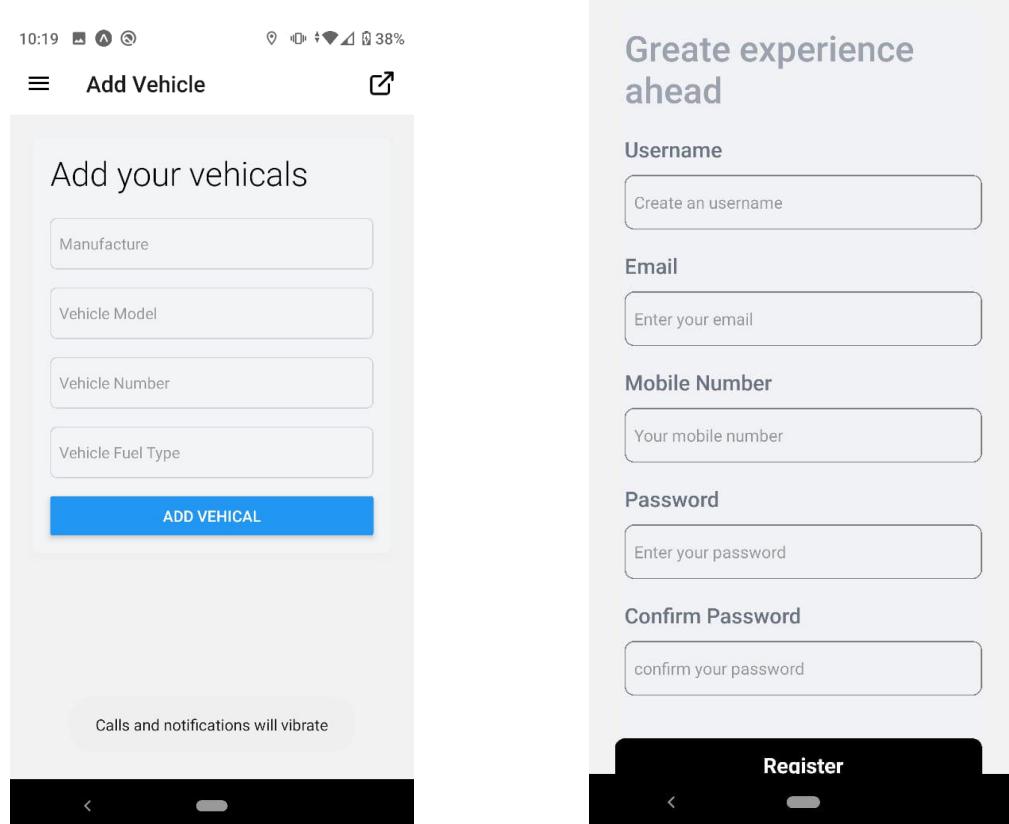
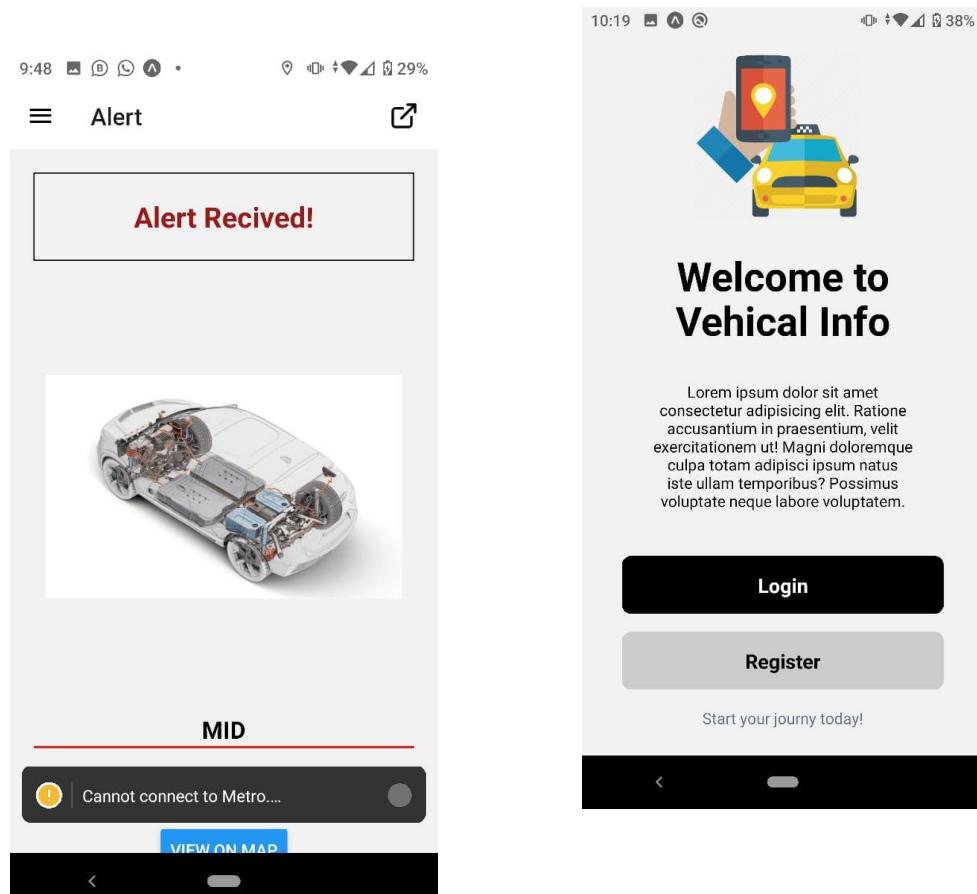


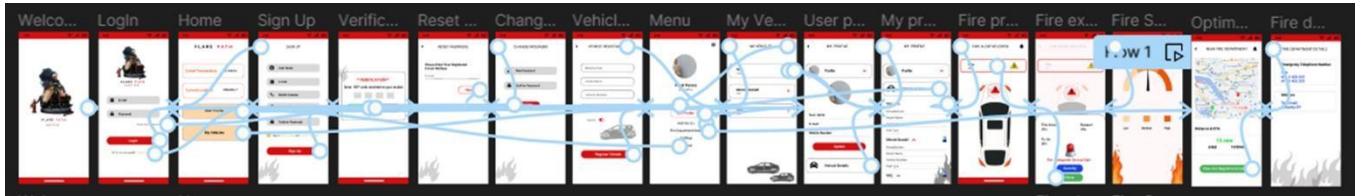
```
File Edit Selection View Go Run Terminal Help fire_type_detection_main.py _base.py d4.csv d2.csv fire_type_detection_main.py fire_type_detection_train_dataset.csv fire_severity.csv

1 from fire_type_detection import predict_fire_type
2
3
4
5
6 se1_inpit = int(input('front sensor temp in C = '))
7 se2_inpit = int(input('mid sensor temp in C = '))
8 se3_inpit = int(input('back sensor temp in C = '))
9
10 new_sensor_data = [se1_inpit, se2_inpit, se3_inpit]
11
12 ss = predict_fire_type(new_sensor_data)
13
14 print(ss)
15
16

Ln 16, Col 1 Spaces: 4 UTF-8 CRLF Python 3.10.11 64-bit (Microsoft Store) Go Live tabnine Starting... Prettier 27°C 10:49 PM ENG 9/11/2024
```

Mobile Application UI/UX Design





UI/UX Designs Flows

Work Breakdown Structure

