

Smart E-VANET For Scheduling and Communication

Welcome Page with Sign Up and Sign In:

The Welcome Page serves as the entry point to our E-VANET platform, inviting users to explore the future of electric mobility. Choose "Sign Up" to become part of our community or "Sign In" if you already have an account.



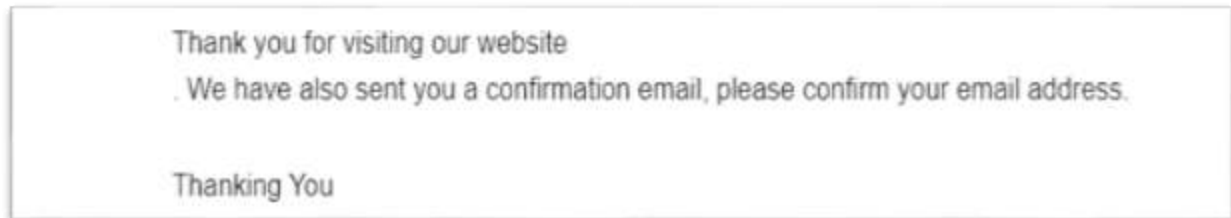
The Creating Account module facilitates user registration for EV owners.

By completing this process, users provide essential details to access seamless services.



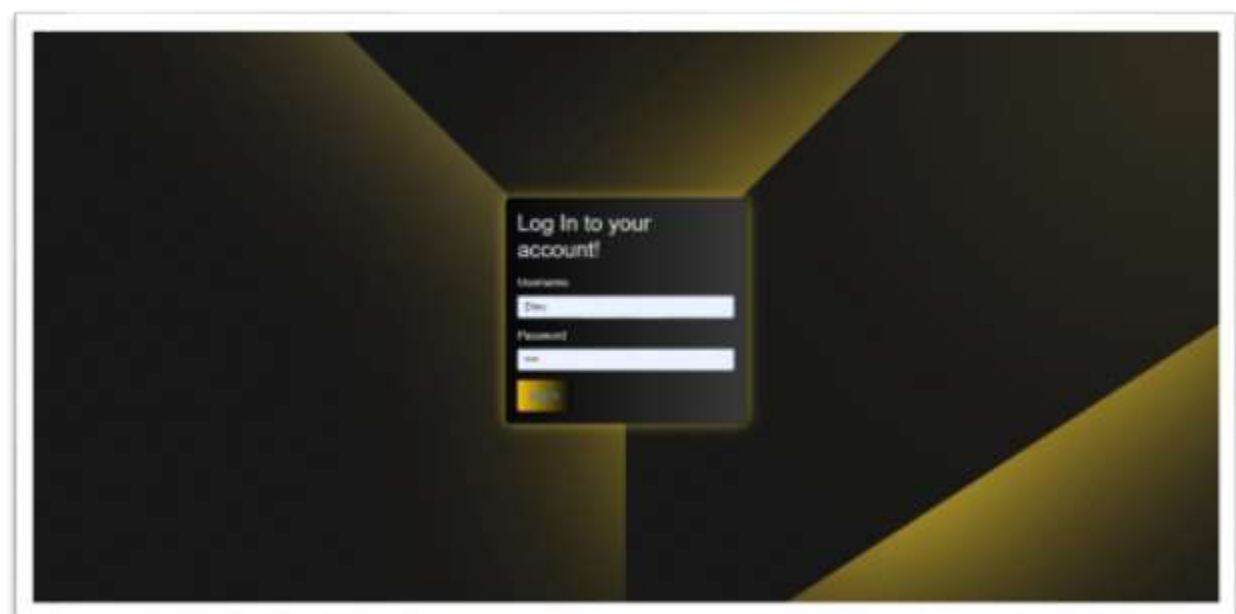
Email Confirmation for EV Owner:

After registering, users receive an email confirmation as part of the account verification process. This email contains a confirmation link, ensuring the security and legitimacy of their E-VANET account.



Redirecting to Login Page after Confirming Email:

Upon successfully confirming their email, users are seamlessly redirected to the Login Page. This step ensures a smooth transition, allowing users to sign in with their verified credentials.



✚ Block chain code :

```

1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.10;
3
4 contract ElectricVehicleCharging {
5
6     struct ChargingStation {
7         uint256 stationId; // station id for the soc location (based on block number + ether + 1 unit)
8         uint256 pricePerKwh; // price per kwh for charging (based on block number + 1 ether + 1 unit)
9     }
10
11     uint256 public thresholdVoltage = 90; // set your threshold value for state of charge (soc)
12     uint256 public averagePricePerKwh; // average price per kwh for charging (based on block number + 1 ether + 1 unit)
13
14     ChargingStation[] public chargingStations;
15     uint256 public numberOfStations = 0;
16     bool public socEntered;
17     bool public stationAdded;
18     uint256 public stationIdCounter;
19
20     // function to enter the state of charge (SOC)
21     function enterSOC(uint256 _soc) external { // @dev soc
22         require(!_socEntered, "State of Charge (SOC) has already been entered.");
23         require(_soc > 90, "Invalid soc value, need to be between 90 and 100.");
24
25         if (_soc > thresholdVoltage) {
26             socEntered = true;
27         }
28     }
29
30 }

```

✚ Output :

Balance: 0. ETH

addChargingStation: 33

enterSOC: 34

averagePricePerKwh: 0: uint256: 7

bookCharging: 0: string: Great deal! You're getting it at an amazing price!

1: uint256: 3

chargingStation: 0: uint256

numberOfStations: 0: uint256: 5

socEntered: 0: bool: true

stationIDCounter: 0: uint256: 0

stationsAdded: 0: bool: true

thresholdVoltage: 0: uint256: 90

Broadcasting Message Feature for EV Owner:

The Broadcasting Message Feature empowers EV owners to share important information with the E-VANET community. Whether it's traffic updates, safety alerts, or community announcements, EV owners can utilize this feature to broadcast messages, fostering real-time communication and collaboration within the E-VANET network.

Receiving Broadcasted Messages:

EV owners seamlessly receive broadcasted messages from their peers through the E-VANET platform. Stay informed about road conditions, events, and community updates as part of our commitment to enhancing communication and safety within the electric vehicle community.

Broadcast Message Sender

You (Profile ID)

Type your message

Send

neha (U246) - 11/30/2023, 3:16:49 PM: Traffic Ahead at nima circle

raj (U248) - 11/30/2023, 3:17:07 PM: road blockage near gate bridge

janisha (U527) - 11/30/2023, 3:17:32 PM: traffic jam near rashtrapati circle

neel (U244) - 11/30/2023, 3:17:48 PM: Go ahead!

Message sent successfully!

Dataset for SOC prediction module :

combined_df.head()

Regenerative Braking Signal	Battery Voltage [V]	Battery Current [A]	Temperature Footwell Co-Driver [C]	Temperature Footwell Co-Driver [C]	Temperature Footwell Driver [C]	Temperature Head Co-Driver [C]	Temperature Head Driver [C]	Temperature Vent right [C]	Temperature Vent central right [C]	Temperature Vent central left [C]	Temperature Vent right [C]	Velocity [km/h]]]
0.0	391.4	-2.29	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.0	391.4	-2.21	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.0	391.4	-2.26	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.0	391.4	-2.30	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
0.0	391.4	-2.30	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1094793 entries, 0 to 1094792
Data columns (total 59 columns):
#   Column                                     Non-Null Count  Dtype
---  -
0   Unnamed: 0                               1094793 non-null  int64
1   Time [s]                                 1094793 non-null  float64
2   Velocity [km/h]                          1078364 non-null  float64
3   Elevation [m]                            1094793 non-null  float64
4   Throttle [%]                             1094793 non-null  float64
5   Motor Torque [Nm]                       1094793 non-null  float64
6   Longitudinal Acceleration [m/s^2]        1094793 non-null  float64
7   Regenerative Braking Signal              1094793 non-null  float64
8   Battery voltage [V]                      1094793 non-null  float64
9   Battery Current [A]                     1094793 non-null  float64
10  Battery Temperature [C]                  10090 non-null    float64
11  max. Battery Temperature [C]             10090 non-null    float64
12  SoC [%]                                  1064000 non-null  float64
13  displayed SoC [%]                       1063999 non-null  float64
14  min. SoC [%]                            1064000 non-null  float64
15  max. SoC [%]                            1063999 non-null  float64
16  Heating Power CAN [kW]                   1094793 non-null  float64
17  Heating Power LIN [W]                    784206 non-null   float64
18  Requested Heating Power [W]              1094793 non-null  float64
19  AirCon Power [kW]                       1094793 non-null  float64
...
57  Temperature Vent right [ C]              622453 non-null   float64
58  Velocity [km/h]]]                       16429 non-null    float64
dtypes: float64(57), int64(2)
memory usage: 492.8 MB

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

✚ Used parameter :

```
NN

combined_df = combined_df.rename(columns={
    'Time [s]': 'Time',
    'Battery Voltage [V]': 'Battery Voltage',
    'Battery Temperature [ C]': 'Battery Temperature',
    'AirCon Power [kW]': 'AirCon Power',
    'Heater Signal': 'Heater Signal',
    'Ambient Temperature Sensor [ C]' : 'Ambient Temperature Sensor',
    'Temperature Head Driver [ C]' : 'Temperature Head Driver',
    'Temperature Vent right [ C]' : 'Temperature Vent right'
})
```

✚ Model:

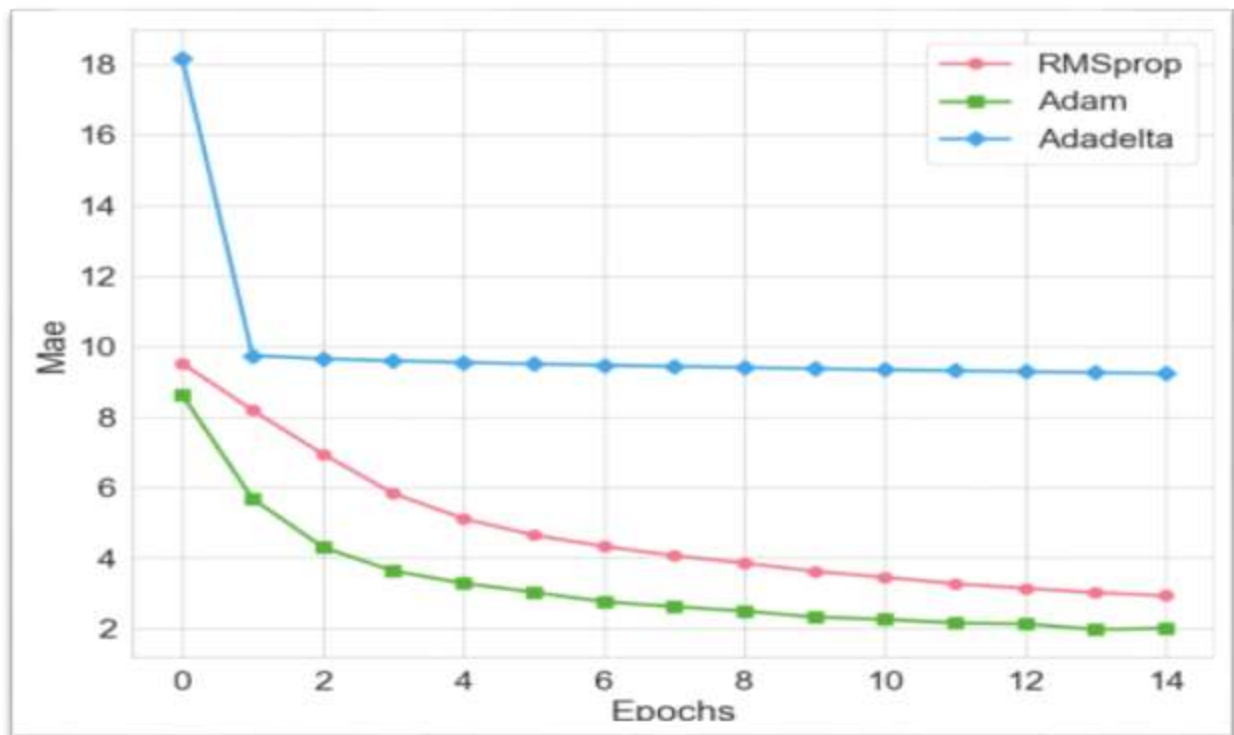
```
model.summary()
```

Model: "sequential"

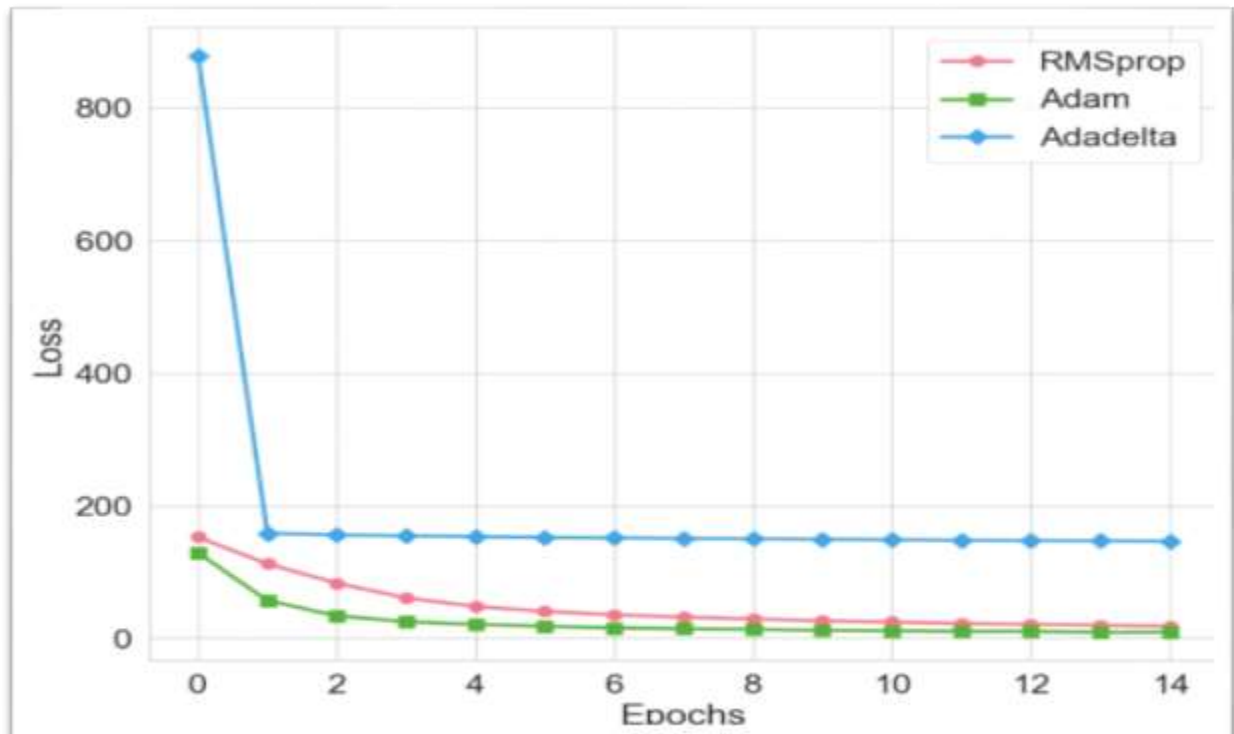
Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 6, 64)	256
max_pooling1d (MaxPooling1D)	(None, 3, 64)	0
flatten (Flatten)	(None, 192)	0
dense (Dense)	(None, 128)	24704
dense_1 (Dense)	(None, 1)	129

=====
Total params: 25089 (98.00 KB)
Trainable params: 25089 (98.00 KB)
Non-trainable params: 0 (0.00 Byte)

✚ Training graph :



✚ Training loss graph :



Adam prediction comparison :

