SMART TAXI METER SYSTEM

Electrical and Electronic Engineering

HNDE – Labuduwa

Presented by: Group 05

GROUP MEMBERS

G.W.I. DEVINDI

G.T.N. SAMARAJEEWA

W.T.M.N.U. WIJERATHNA

L.N. WEERASINGHE

E.M.I.S. EGALLA

OVERVIEW

01 INTRO

- Traditional taxi fare systems struggle with real
- Time fuel price changes and accurate distance measurement.
- Our solution uses a rotary encoder to measure distance and calculate fares dynamically.
- Data from up to 10 meters is collected in a central database and updated in real-time according to fuel price fluctuations.
- New meters automatically connect to the server to synchronize and update fare rates.

KEY BENEFITS

- Real-time fare updates
- Accurate distance
- Based fare calculation
- Reliable and fair system for drivers and passengers
- Efficient, scalable, and modernized solution

02 SCOPE

Develop a smart taxi meter system that provides accurate, transparent, and real-time fare calculations reflecting actual distance traveled and fluctuating fuel prices.

03 AIM

To develop a smart taxi meter system that accurately calculates fares in real-time by measuring the distance traveled and adjusting prices according to current fuel costs.

04 PROBLEM STATEMENT

01 Fare calculations are inaccurate as they don't reflect real-time fuel price changes.

02 Fare rate updates are manual, time-consuming, and error-prone.

03 Distance measurement is often inaccurate, causing fare errors.

04 Managing multiple taxi meters independently leads to inconsistent pricing and inefficient fleet management.

05 Updating new meters with current pricing is complicated without a centralized system

06 Lack of transparency leaves passengers unsure if fares are fair and accurate.

05 OBJECTIVES

1. To design a smart meter using a rotary encoder for accurate distance measurement.
2. To implement real-time fare calculation based on distance and current fuel prices.
3. To develop a central database for managing data from multiple taxi meters.
4. To enable automatic synchronization of fare rates when new meters are added.
5. To ensure transparency and fairness in fare charging for both drivers and passengers.
6. To build a scalable system that can support multiple taxis efficiently.

06

SIMILAR SYSTEMS

PickMe / Uber

These ride-hailing apps use GPS-based fare calculation, which adjusts with distance and time, and includes fuel cost indirectly.

Modern Taxi Meters

Some three-wheelers and taxis use electronic meters, but they mostly operate on static rates without real-time fuel price updates.

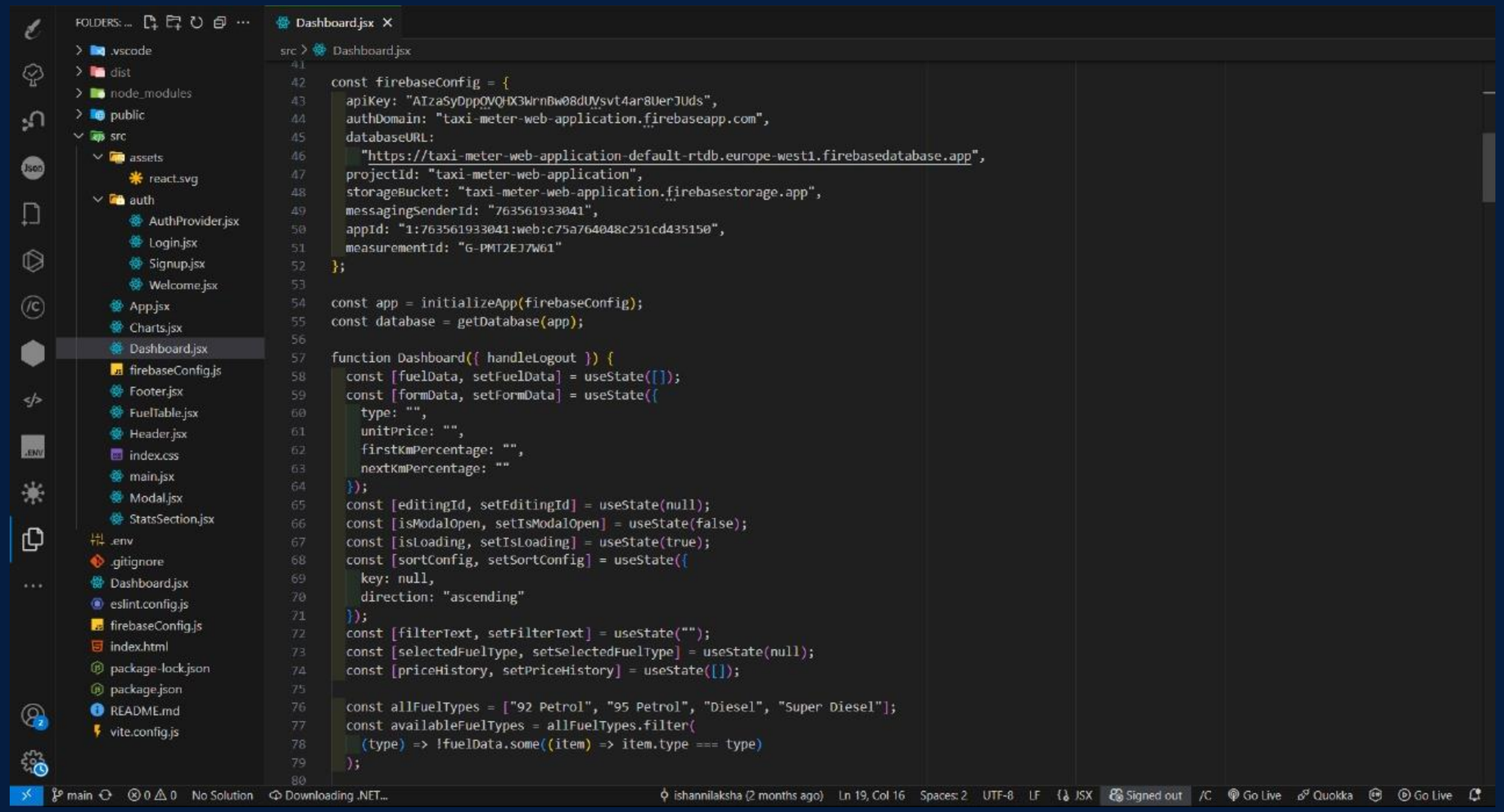
Fleet Management Systems

Used by logistics and transport companies to monitor distance, fuel usage, and performance — but not directly for dynamic fare calculation.

07 COMPONENTS

- Rotary Encoder
- Microcontroller
- Power Supply & Wi-Fi Module
- Humanity Sensor
- Real Time
- Ultrasonic Sensor
- Motor Driver
- LCD Display

08 ARDUINO CODES



08 ARDUINO CODES

Dashboard.jsx

src > Dashboard.jsx

```jsx
const firebaseConfig = {
  apiKey: "AIzaSyDppQVQHX3WrnBw08dUVsvt4ar8UerJUds",
  authDomain: "taxi-meter-web-application.firebaseapp.com",
  databaseURL:
    "https://taxi-meter-web-application-default-rtdb.europe-west1.firebasedatabase.app",
  projectId: "taxi-meter-web-application",
  storageBucket: "taxi-meter-web-application.firebasestorage.app",
  messagingSenderId: "763561933041",
  appId: "1:763561933041:web:c75a764048c251cd435150",
  measurementId: "G-PMT2EJ7W61"
};

const app = initializeApp(firebaseConfig);
const database = getDatabase(app);

function Dashboard({ handleLogout }) {
  const [fuelData, setFuelData] = useState([]);
  const [formData, setFormData] = useState({
    type: "",
    unitPrice: "",
    firstKmPercentage: "",
    nextKmPercentage: ""
  });
  const [editingId, setEditingId] = useState(null);
  const [isModalOpen, setIsModalOpen] = useState(false);
  const [isLoading, setIsLoading] = useState(true);
  const [sortConfig, setSortConfig] = useState({
    key: null,
    direction: "ascending"
  });
  const [filterText, setFilterText] = useState("");
  const [selectedFuelType, setSelectedFuelType] = useState(null);
  const [priceHistory, setPriceHistory] = useState([]);

  const allFuelTypes = ["92 Petrol", "95 Petrol", "Diesel", "Super Diesel"];
  const availableFuelTypes = allFuelTypes.filter(
    (type) => !fuelData.some((item) => item.type === type)
  );
```

08 ARDUINO CODES

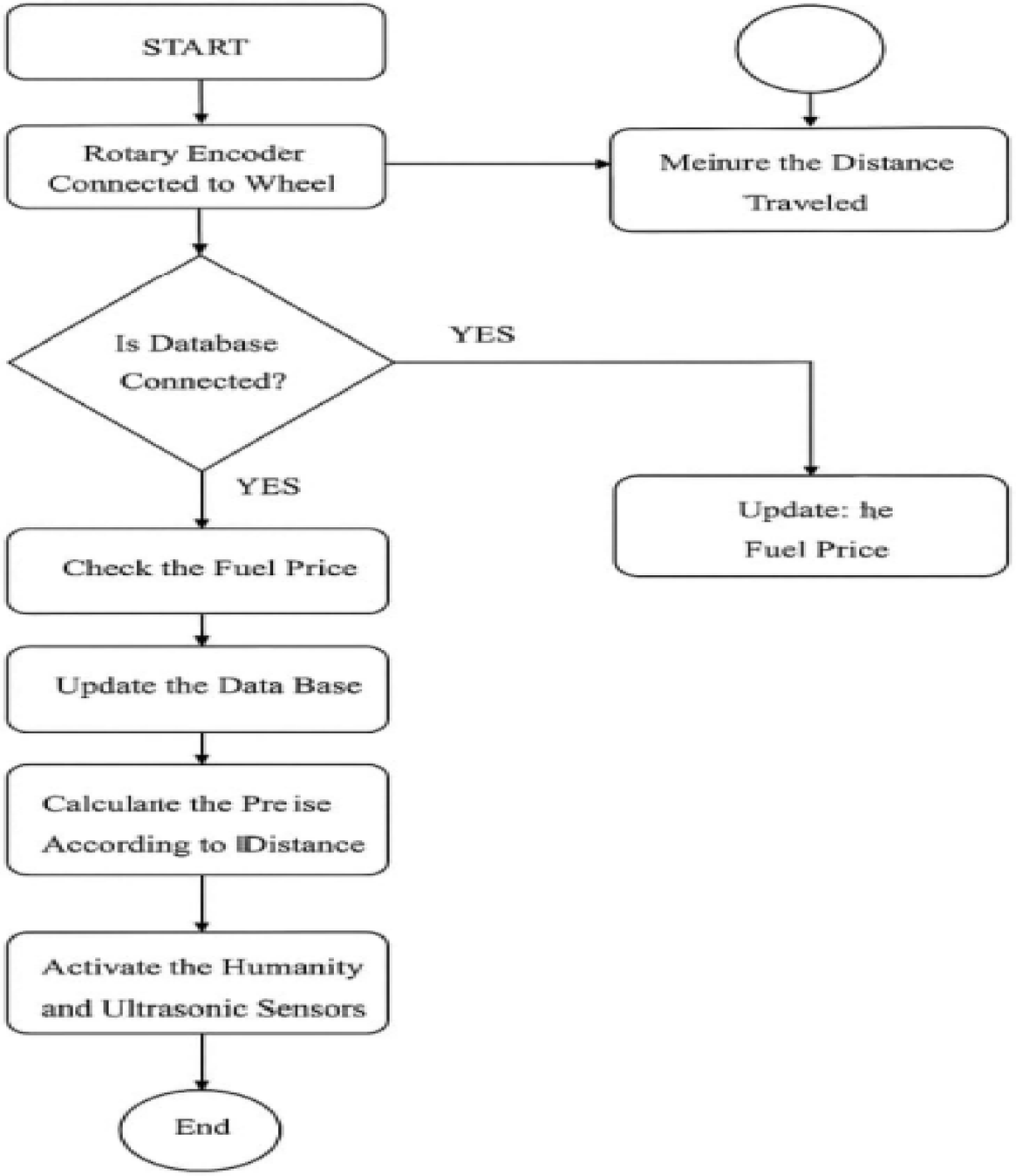
firebaseConfig.js

src > firebaseConfig.js > ...

```javascript
// src/firebaseConfig.js
import { getAnalytics } from "firebase/analytics";
import { initializeApp } from "firebase/app";
import { getAuth, GoogleAuthProvider } from "firebase/auth";
import { getDatabase } from "firebase/database";

const firebaseConfig = {
  apiKey: "AIzaSyDppOVQHX3WrnBw08dUVsvt4ar8UerJUds",
  authDomain: "taxi-meter-web-application.firebaseapp.com",
  databaseURL: "https://taxi-meter-web-application-default-rtdb.europe-west1.firebasedatabase.app",
  projectId: "taxi-meter-web-application",
  storageBucket: "taxi-meter-web-application.firebasestorage.app",
  messagingSenderId: "763561933041",
  appId: "1:763561933041:web:c75a764048c251cd435150",
  measurementId: "G-PMT2EJ7W61"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
const auth = getAuth(app);
const database = getDatabase(app);
const googleProvider = new GoogleAuthProvider();
const analytics = getAnalytics(app);

export { analytics, app, auth, database, googleProvider };
```

09

METHODOLOGY



09

METHODOLOGY

Start

The system initializes when the taxi begins operation.

Rotary Encoder

A rotary encoder connected to the taxi’s wheel measures the actual distance traveled during the ride

Oil Price Data

The system fetches current fuel price data from a central database or external source.

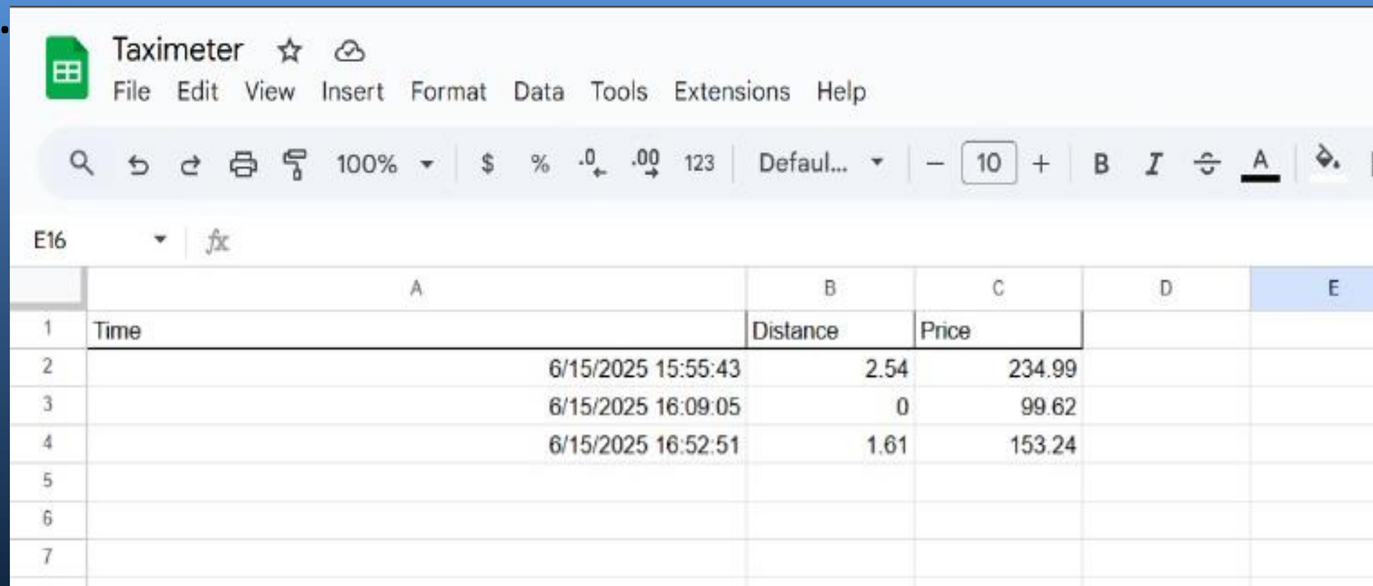
09 METHODOLOGY

Microcontroller

The microcontroller receives input from the rotary encoder and calculates the total distance covered.

Update Central Database

The calculated distance and current fuel price are sent to a central database in real time.



	Time	Distance	Price
2	6/15/2025 15:55:43	2.54	234.99
3	6/15/2025 16:09:05	0	99.62
4	6/15/2025 16:52:51	1.61	153.24
5			
6			
7			

09 METHODOLOGY

Update Fare from Database

The fare is calculated dynamically using the latest distance and fuel price data, then updated on the system.

Display Fare to Customer

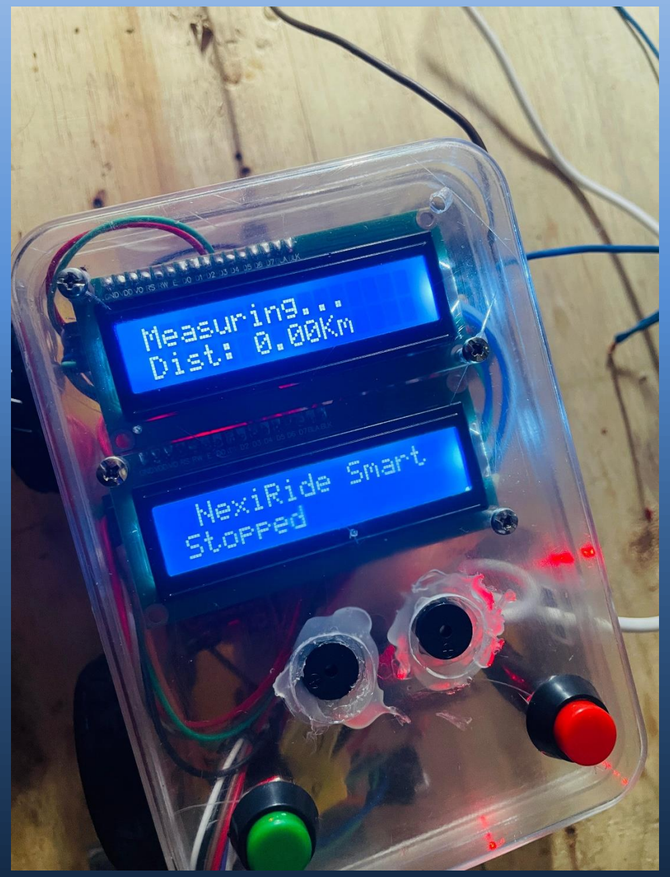
The final fare is displayed on the taxi meter screen for the passenger to see.

09

METHODOLOGY

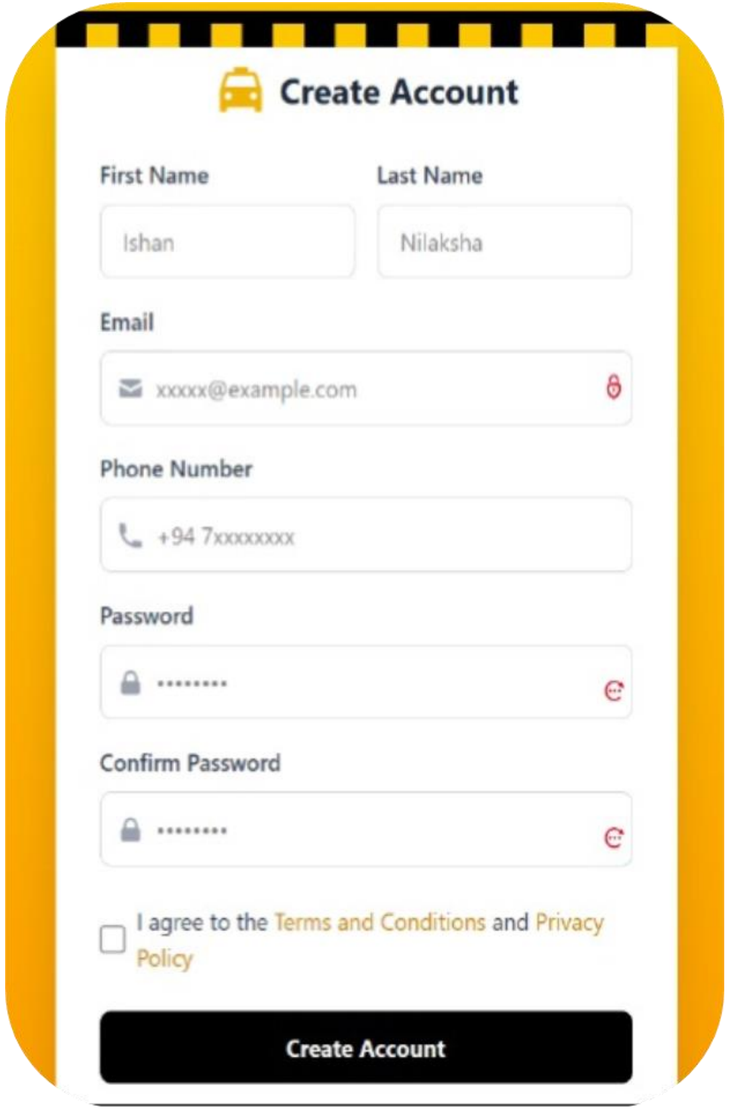
End

The process ends after the ride is completed and the fare is shown.

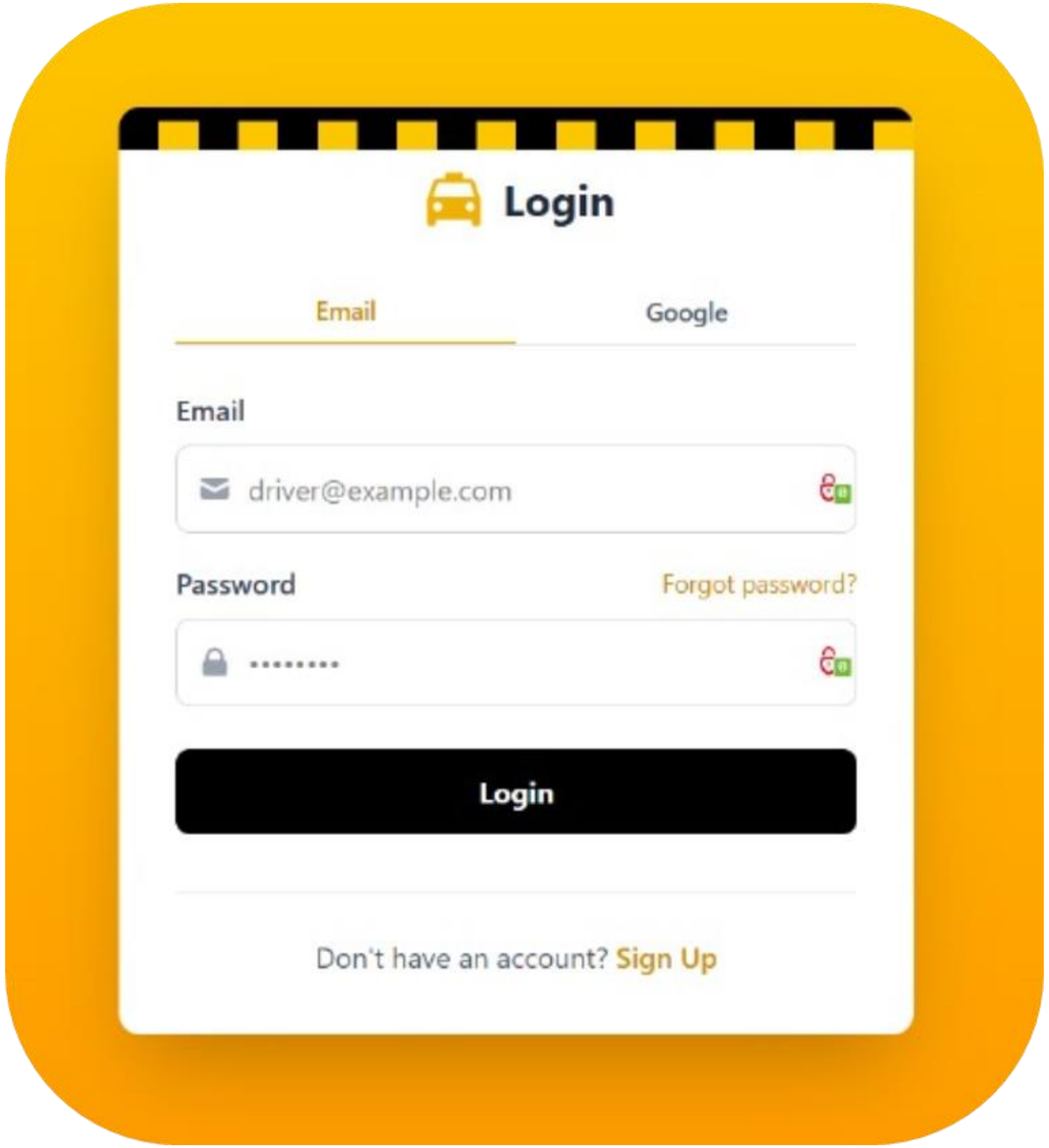


TAXI METER APP

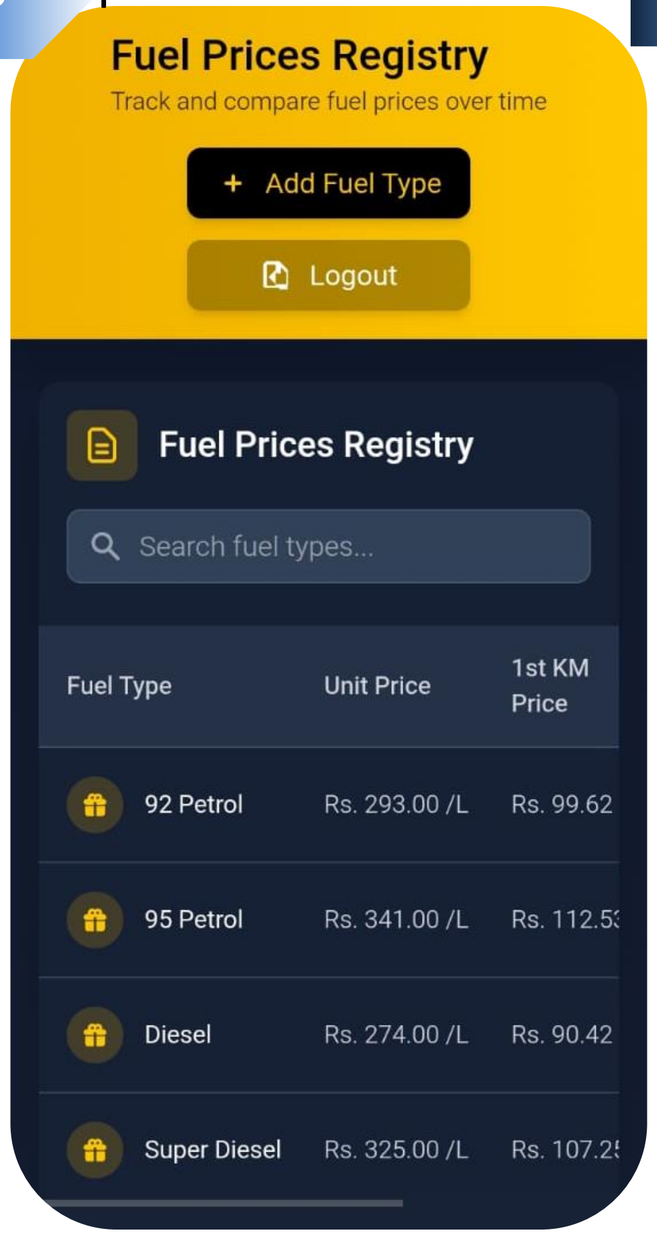
01 Create an Account



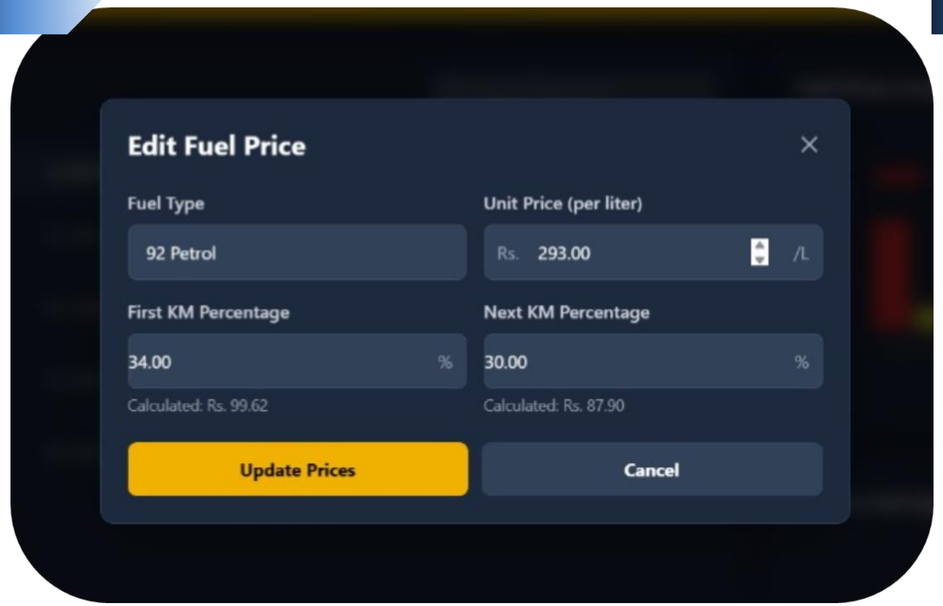
02 Login to the app



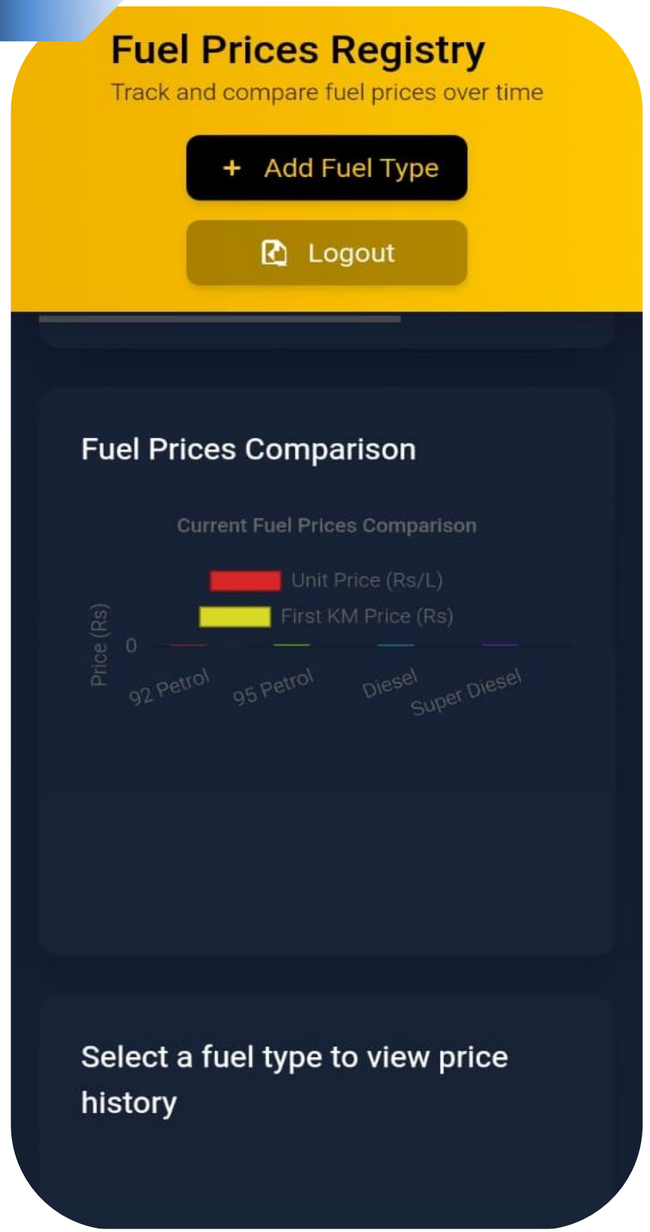
03 Updated Fuel Details



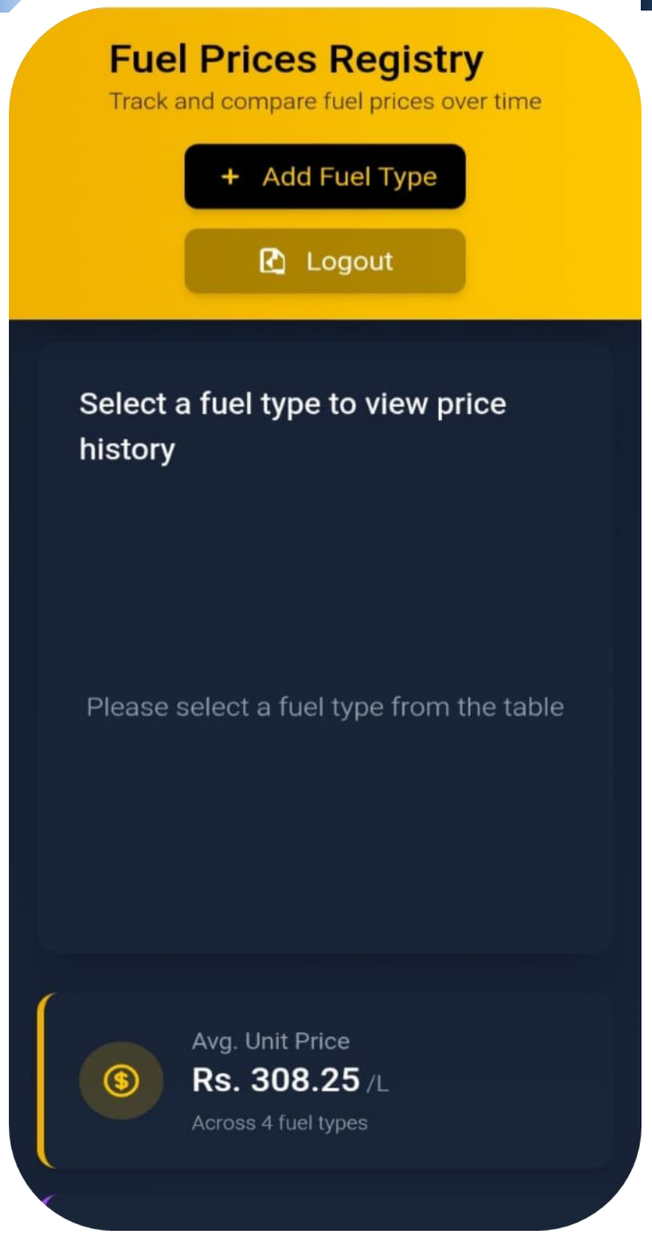
04 Edit Fuel Details



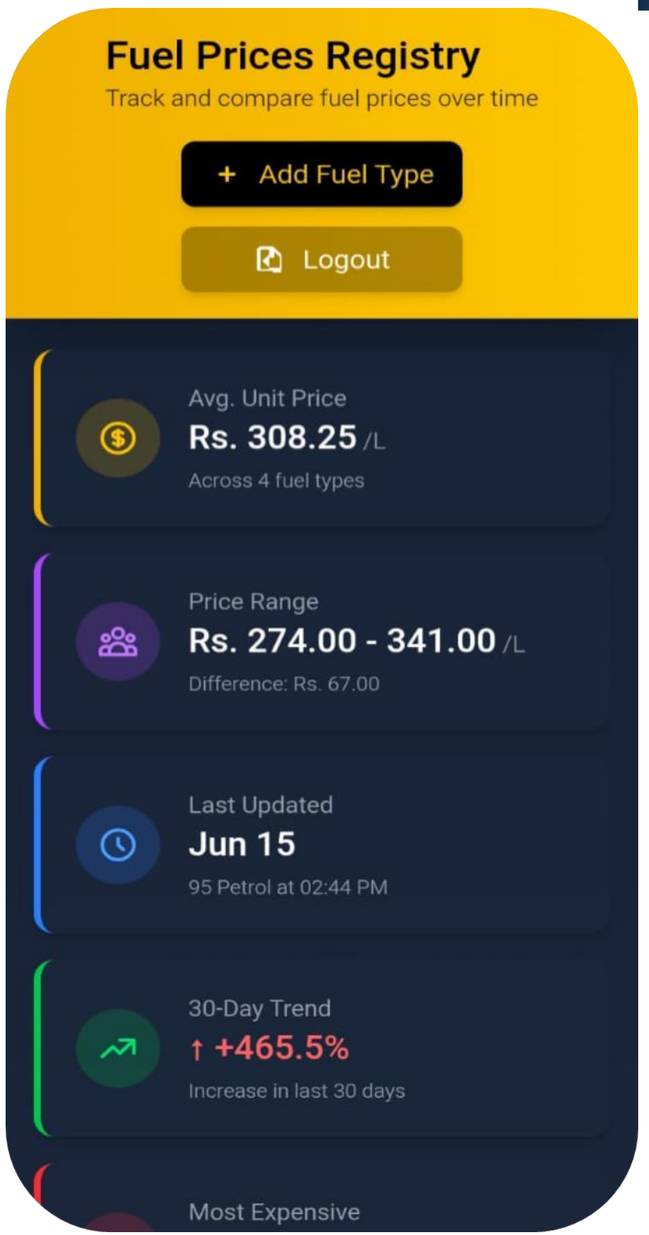
05 Price Comparison



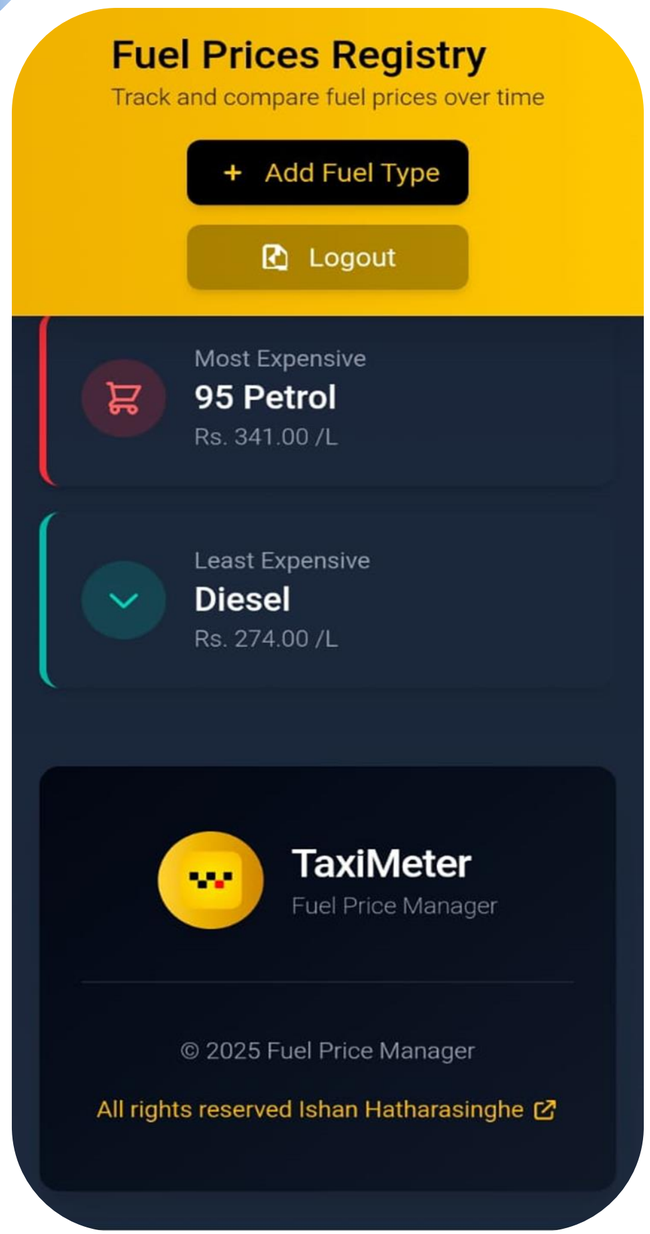
06 Fuel Price History



07 Other Details



08 Other Details



TIME PLAN

- Planning
- Hardware Development
- software Development
- Testing & Calibration
- Finalization

11 COST ANALYZE

COMPONENT	QUANTITY	ESTIMATED COST (LKR)
Arduino Uno Board	1	100
Arduino Mega Board		
L298N Motor Driver	1	330
NEO-6M GPS Module	1	780
ULTRASONIC Sensor	1	195
DHT11 Module	1	175
5V Active Buzzer	1	32
DS3231 RTC Module	1	320
1602 LCD Display (Blue)	2	550
I2C	2	260
Mini Rocker Switch	1	10
Gear Motor (Yellow)	4	460
Wheel Normal	4	280
L293D Motor Shield	1	390
18650 Green 1800MAH-Button Top	2	590
18650*2 Battery Holder	1	50
Other		4000
Total		8000/=

12

FUTURE DEVELOPMENT

- **GPS Module**: Enable real-time location tracking and route-based fare.
- **Load Cell:** Detect passenger presence or package weight for fare adjustment.
- **Wheel Encoder:** Improve distance accuracy and vehicle diagnostics.
- **RFID Reader:** Add driver/passenger authentication and digital payments.
- **Ultrasonic Sensor:** Assist with obstacle detection and passenger presence

13

RESOURCES & REFERENCES

RESOURCES

- Arduino
- MS office
- Java script
- React

REFERENCES

- Arduino project hub
- Instructables
- youtube

14

Q & A

THANK YOU

CONNECT WITH US.

077-843 2168

test01@gmail.com