```
import java.io.*;
import java.util.*;

public class Main {

    private static final String FILE_NAME = "users.dat";
    private static final Scanner sc = new Scanner(System.in);

    public static void main(String[] args) {
        entrypoint();
    }

    public static void entrypoint() {
        System.out.println("Welcome to Ecotrack!");
        System.out.print("Choose: Login / Signup: ");
        String ch = sc.nextLine().trim().toLowerCase();

        if (ch.equals("signup") || ch.equals("sign up")) {
            signup();
        } else if (ch.equals("login")) {
            login();
        } else {
            System.out.println("Invalid choice. Please enter Login or
Signup");
        }
    }

    // ---------- SIGNUP ----------
    public static void signup() {
        String email;
        while (true) {
            System.out.print("Enter your email id: ");
            email = sc.nextLine();
            if (email.endsWith("@gmail.com")) break;
            System.out.println("Please enter a VALID email
(name@gmail.com)");
        }

        System.out.print("Enter your password: ");
        String password = sc.nextLine();

        System.out.print("Enter nickname: ");
        String nickname = sc.nextLine();

        User newUser = new User(email, password, nickname);

        List<User> users = loadUsers();
        users.add(newUser);
        saveUsers(users);

        System.out.println("Signup successful! Please login to continue.");
        login();
    }

    // ---------- LOGIN ----------
    public static void login() {
        String email;
        while (true) {
            System.out.print("Enter email: ");
            email = sc.nextLine();
            if (email.endsWith("@gmail.com")) break;
```

```
            System.out.println("Please enter a VALID email
(name@gmail.com)");
        }

        List<User> users = loadUsers();
        if (users.isEmpty()) {
            System.out.println("No users found. Please signup first.");
            entrypoint();
            return;
        }

        User user = null;
        for (User u : users) {
            if (u.getEmail().equals(email)) {
                user = u;
                break;
            }
        }

        if (user == null) {
            System.out.println("No user found with that email. Please
signup first.");
            entrypoint();
            return;
        }

        int attempts = 0;
        while (attempts < 3) {
            System.out.print("Enter password: ");
            String pw = sc.nextLine();

            if (pw.equals(user.getPassword())) {
                System.out.println("Login successful! Welcome, " +
user.getNickname());

                if (user.getVehicles() == null) {
                    user.setVehicles(recordVehicles());
                }

                saveUsers(users);
                calculate(user.getVehicles());
                return;
            } else {
                attempts++;
                System.out.println("Wrong password. Attempts left: " + (3 -
attempts));
            }
        }

        System.out.println("Too many failed attempts. Waiting 10
seconds...");
        try { Thread.sleep(10000); } catch (Exception ignored) {}
        login();
    }

    // ---------- RECORD VEHICLES ----------
    public static List<Vehicle> recordVehicles() {
        System.out.println("Welcome to Ecotrack! Let's reduce your carbon
footprint");
        System.out.println("Enter your vehicle details:");
```

```
        List<Vehicle> vehicles = new ArrayList<>();

        System.out.print("\nEnter number of vehicles in your household: ");
        int num = Integer.parseInt(sc.nextLine());

        for (int i = 0; i < num; i++) {
            System.out.println("\nDetails of Vehicle " + (i + 1));

            String fuelType;
            double emFact;

            while (true) {
                System.out.print("Enter fuel type (petrol/diesel/cng): ");
                fuelType = sc.nextLine().trim().toLowerCase();

                if (fuelType.equals("petrol")) {
                    emFact = 2.31;
                    break;
                }
                if (fuelType.equals("diesel")) {
                    emFact = 2.68;
                    break;
                }
                if (fuelType.equals("cng")) {
                    emFact = 2.69;
                    break;
                }
                System.out.println("Please enter a valid fuel type.");
            }

            System.out.print("Enter mileage (km/l): ");
            double mileage = Double.parseDouble(sc.nextLine());

            vehicles.add(new Vehicle(fuelType, mileage, emFact));
        }

        System.out.println("Vehicles recorded successfully!");
        return vehicles;
    }

    // ---------- CALCULATE EMISSIONS ----------
    public static void calculate(List<Vehicle> vehicles) {
        System.out.println("\nCalculating today's carbon footprint...");

        double total = 0;

        for (Vehicle v : vehicles) {
            System.out.print("\nEnter distance travelled today (km): ");
            double dis = Double.parseDouble(sc.nextLine());

            double fuelUsed = dis / v.getMileage();
            double emission = fuelUsed * v.getEmissionFactor();
            total += emission;

            System.out.println("Vehicle Emission: " + String.format("%.2f",
emission) + " kg CO2");
        }

        System.out.println("\nTotal Daily Carbon Footprint: " +
String.format("%.2f", total) + " kg CO2");
```

```
        if (total < 3) {
            System.out.println("Low: Great job!");
        } else if (total < 8) {
            System.out.println("Moderate: Try walking/cycling short
trips.");
        } else if (total < 15) {
            System.out.println("High: Consider carpooling or public
transport.");
        } else {
            System.out.println("Very High: Consider EVs or fewer trips.");
        }
    }

    // ---------- LOAD / SAVE USERS ----------
    @SuppressWarnings("unchecked")
    public static List<User> loadUsers() {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream(FILE_NAME))) {
            return (List<User>) ois.readObject();
        } catch (Exception e) {
            return new ArrayList<>();
        }
    }

    public static void saveUsers(List<User> users) {
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream(FILE_NAME))) {
            oos.writeObject(users);
        } catch (Exception ignored) {}
    }
}

// ---------- USER CLASS ----------
class User implements Serializable {
    private String email;
    private String password;
    private String nickname;
    private List<Vehicle> vehicles;

    public User(String email, String password, String nickname) {
        this.email = email;
        this.password = password;
        this.nickname = nickname;
    }

    // getters
    public String getEmail() { return email; }
    public String getPassword() { return password; }
    public String getNickname() { return nickname; }
    public List<Vehicle> getVehicles() { return vehicles; }

    // setter
    public void setVehicles(List<Vehicle> v) { this.vehicles = v; }
}

// ---------- VEHICLE CLASS ----------
class Vehicle implements Serializable {
    private String fuelType;
    private double mileage;
    private double emissionFactor;
```

```
    public Vehicle(String fuelType, double mileage, double emissionFactor)
{
        this.fuelType = fuelType;
        this.mileage = mileage;
        this.emissionFactor = emissionFactor;
    }

    public double getMileage() { return mileage; }
    public double getEmissionFactor() { return emissionFactor; }
```