Vehicle Management and Accident Tracking System

Project Description:

The Vehicle Management and Accident Tracking System is a structured software project designed to manage vehicles, their owners, and accident records in an organized way.

It focuses on data validation and accuracy to ensure that all stored information is complete and correct.

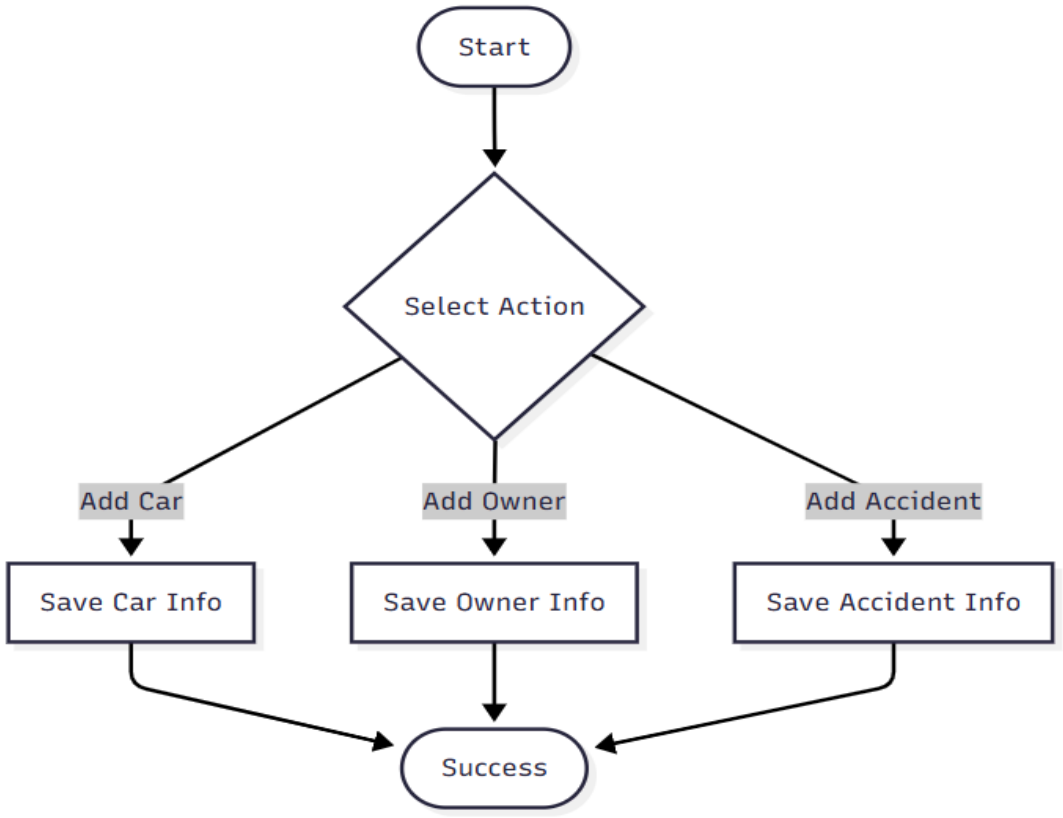
The system includes three main parts:

1. Car Model: Stores essential details about each vehicle such as VIN, plate number, make, model, year, and mileage.

2. Owner Model: Contains information about the vehicle's owner, including personal and contact details.

3. Accident Model: Records of accident data such as date, location, severity, number of injuries, and repair costs.

The goal of this project is to simulate a real-world system that maintains high-quality data through clear validation rules and logical structure.



Model 1: CAR

Serial Number:

- Cannot be empty.
- Must be exactly **17 characters long**.
- Can contain only **uppercase letters (A–Z)** and **numbers (0–9)**.
- Must **not include** the letters **I**, **O**, or **Q**.

Plate Number:

- Cannot be empty.
- Must follow the **Saudi format** — starts with 1–4 digits followed by 1–3 letters.

Make:

- Cannot be empty.
- Must contain **only letters and spaces** (no numbers or special symbols).

Model:

- Cannot be empty.
- May contain **letters, numbers, and spaces only**.

Year:

- Must be a number more than**1980**

Mileage (km):

- Must be a **positive number** or **zero**.
- Represents the total distance driven in **kilometers**.

```
@NotEmpty(message = "Serial number cannot be empty")

@Pattern(  regexp = "^[A-HJ-NPR-Z0-9]{17}$", message = "Serial number must be 17 characters (A–Z, 0–9) without I, O,Q" )

  private String serialNumber;


  @NotEmpty(message = "Plate number cannot be empty")

  @Pattern(  regexp = "^[0-9]{1,4}[A-Za-z]{1,3}$", message = "Invalid plate number format" )
```

```
    private String plateNumber;


    @NotEmpty(message = "Make cannot be empty")
    @Pattern( regexp = "^[A-Za-z ]+$",message = "Make must contain only letters and spaces")
    private String make;


    @NotEmpty(message = "Model cannot be empty")
    @Pattern (regexp = "^[A-Za-z0-9 ]+$", message = "Model may contain letters, digits, and spaces )
    private String model;


    @Min(value = 1980, message = "Year must be greater than or equal to 1980")
       private int year;
    @PositiveOrZero(message = "Mileage (km) must be zero or positive")
    private int mileageKm;
}
```

Model 2: OWNER

ID:

- Cannot be empty.
- Must be at least **2 characters long**.

Full Name:

- Cannot be empty.
- Must be at least **4 characters long**.
- Must contain **only letters and spaces**.

Email:

- Must be a **valid email format** .

Phone Number:

- Cannot be empty.
- Must start with **05**.
- Must be exactly **10 digits long** .

License Number:

- Cannot be empty.
- Must be **6–12 alphanumeric characters** (letters or numbers, no symbols).

Birth Date:

- Cannot be null.
- Must be a **past date** (before today).

```
@NotEmpty(message = "ID cannot be empty")
@Size(min = 2, message = "ID must be at least 2 characters long")
private String id;


@NotEmpty(message = "Full name cannot be empty")
@Size(min = 4, message = "Full name must be at least 4 characters long")
@Pattern(
   regexp = "^[A-Za-z ]+$",
   message = "Full name must contain only letters and spaces")
private String fullName;
```

```
    @Email(message = "Email must be a valid format")
    private String email;

    @NotEmpty(message = "Phone number cannot be empty")
    @Pattern(  regexp = "^05\\d{8}$",  message = "Phone number must start with 05 and contain exactly 10 digits" )
    private String phoneNumber;

    @NotEmpty(message = "License number cannot be empty")
    @Pattern(  regexp = "^[A-Za-z0-9]{6,12}$" , message = "License number must be 6–12 alphanumeric characters )
    private String licenseNumber;

    @NotNull(message = "Birth date cannot be null")
    @Past(message = "Birth date must be in the past")
    private LocalDate birthDate;
}
```

Model 3: ACCIDENT

ID:

- Cannot be empty.

Car Serial Number:

- Cannot be empty.
- Must be exactly **17 characters long**.
- Can contain only **uppercase letters (A–Z)** and **numbers (0–9)**.
- Must **not include** the letters **I**, **O**, or **Q**.

Owner ID:

- Cannot be empty.
- Must be at least **2 characters long**.

Date:

- Cannot be null.
- Must be **today or a past date** (not in the future).

Location:

- Cannot be empty.

Severity:

- Cannot be empty.
- Must be one of: **minor**, **moderate**, **major**, **critical**.

Injuries Count:

Must be a **positive number** or **zero**.

Estimated Cost:

- Must be a **positive number** or **zero**.

Description :

- If provided, must be **≤ 500 characters**.

```
@NotEmpty(message = "ID cannot be empty")
  private String id;

  @NotEmpty(message = "Car serial number cannot be empty")
  @Pattern(
    regexp = "^[A-HJ-NPR-Z0-9]{17}$",
    message = "Car serial number must be 17 characters (A–Z, 0–9) without I, O, Q"
  )
  private String carSerialNumber;
```

```
    @NotEmpty(message = "Owner ID cannot be empty")
    @Size(min = 2, message = "Owner ID must be at least 2 characters long")
    private String ownerId;

    @NotNull(message = "Date cannot be null")
    @PastOrPresent(message = "Date must be today or in the past")
    private LocalDate date;

    @NotEmpty(message = "Location cannot be empty")
    private String location;

    @NotEmpty(message = "Severity cannot be empty")
    @Pattern(
        regexp = "^(minor|moderate|major|critical)$",
        message = "Severity must be one of: minor, moderate, major, critical"
    )
    private String severity;

    @PositiveOrZero(message = "Injuries count must be zero or positive")
    private int injuriesCount;

    @PositiveOrZero(message = "Estimated cost must be zero or positive")
    private double estimatedCost;

    @Size(max = 500, message = "Description must be at most 500 characters")
    private String description;
}
```