Notes

Java (OOP) Basics

```
public class Car {

    String color; // Attributes
    String brand; // Attributes

    public Car(String color, String brand) { // Constructor
        this.color = color;
        this.brand = brand;
    }
    void drive() {
            System.out.println("This " + brand + " is driving.");
    }
    }



// Calling the function
Car myCar = new Car("Blue", "BMW");
myOtherCar.drive();

// OUTPUT
This BMW is driving.
```

Steps:

- Class → Car.
    - Class → It is a BluePrint or Template for creating Objects.
        - It defines a set of properties (attributes) and methods (behaviors) thatthe objects created from the class will have.

- The class doesn'trepresent an actual entity but rather a definition or a concept.

- Color and Brand → attributes of the class Car.

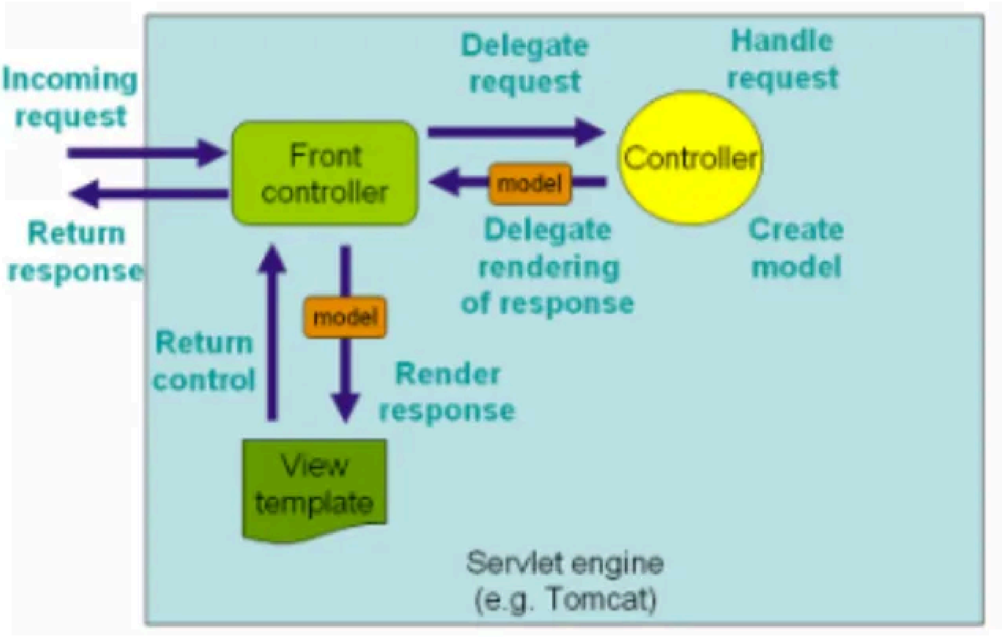
- Constructor (Special Method)
  - It is special method that runs automatically when you create an object new.
  - Here it has 2 parametres Color and Brand.
  - We use the 'this' keyword under the Constructor.
    - `this.color` → means "the color attribute of this object".
    - Without `this`, Java would confuse between the parameter `color` and the attribute `color`.
- Method → drive()
  - The action or the Fucntion the car can do.

The Spring Framework Overview

- Comprehensive Platform for Building Java-based application.
- Spring provides dependency injection and a host of modules (AOP,security, data access, etc.).
- It simplifies the use of design patterns, including MVC, in real-world applications.

Spring MVC (Model/View/Controller)

- It is a specific implementation of the MVC pattern in Java.
  - MODEL→ Java Objects (POJOs) that hold data.
  - VIEW → JSPs Thymeleaf, etc for UI rendering
  - Controller → Annotated with `@controller`and `@RequestMapping` to handle web requests.



- Why Spring Boot?
  - As a framework built on top of spring, designed to make it easier to create stand-alone, production-ready applications.
  - It automates the setup process and allows for rapid prototyting.
    - setup process → Embedding Servers, Starter Dependencies, Auto-Configuration, Convention over Configuration.
  - Support a fasr development cycle enabling teams to move more quickly from design to implementation and testing