Optional

What is Optional?

- Optional<T> is a container object introduced in Java 8.
- It may or may not contain a non-null value.
- Goal: Avoid null checks and NullPointerException.

Why Use Optional?

```
// Bad
String name = user.getName(); // may be null

// Better
Optional<String> name = user.getNameOptional();
```

- Promotes explicit null handling.
- Helps document intent: return value may be missing.
- Reduces bugs from unchecked nulls.

Key Optional Methods

Method	Description
isPresent()	Checks if a value is present
ifPresent(Consumer)	Executes if a value is present
orElse(T)	Returns value or default
orElseGet(Supplier)	Returns value or calls supplier
orElseThrow()	Throws exception if value is missing

Demo – Optional Basics in Java

- Learn how to:
- ✅ Create an Optional
- ✅ Check for presence
- ✅ Use fallback values

Why Learn Optional Before Spring Boot?

- Optional is a modern alternative to null, promoting intentional handling of missing values—essential in real-world apps.
- Encourages readable constructs like orElse(), ifPresent() that is widely used in Spring Boot projects.
- Used in APIs like Spring Data JPA (findById()), Spring Web (REST responses), and configuration properties.

Programming Challenge Instructions

- Create a class Car with a single field licensePlate and a getter method, or reuse a Car class from previous challenges.
- Create an Optional<Car> that contains a Car with license plate "ABC123".
- Use ifPresent() to print the license plate.
- Create an empty Optional<Car> and print the licensePlate if the value is present or "No car available" if the value is absent.