



Projeto Banco de Dados



Apresentado por: **Matheus Camargo**  
**& Pedro Guaita**



# Banco de Dados



MySQL®

# **MODELO LÓGICO**

# Views

**Lista de views**

# Exemplos de View

# Procedures

**Lista de procedures**

# Exemplos de Procedures



# Triggers

**Lista de triggers**

# Exemplos de Trigger

# Spring Boot

**Java**



# Contexto

O **Spring** nasceu no início dos anos 2000 como uma alternativa ao Java EE, que na época era muito burocrático e difícil de configurar. Ele trouxe uma abordagem mais **leve** e **flexível**, permitindo que desenvolvedores criassem aplicações robustas sem tanta complexidade



# Diferenciais

---

- **Autoconfiguração:** Você não precisa configurar tudo manualmente. O Spring Boot detecta automaticamente as dependências e as configura para você.
- **Servidor embutido:** Diferente do Java EE tradicional, onde você precisava de um servidor como Tomcat ou WildFly, o Spring Boot já traz um servidor embutido, como o Tomcat, Jetty ou Undertow. Isso significa que você pode rodar sua aplicação como um simples arquivo JAR.
- **Starter Packs:** São pacotes de dependências prontos para facilitar a integração com bancos de dados, segurança, mensageria e muito mais.



# Estrutura do projeto

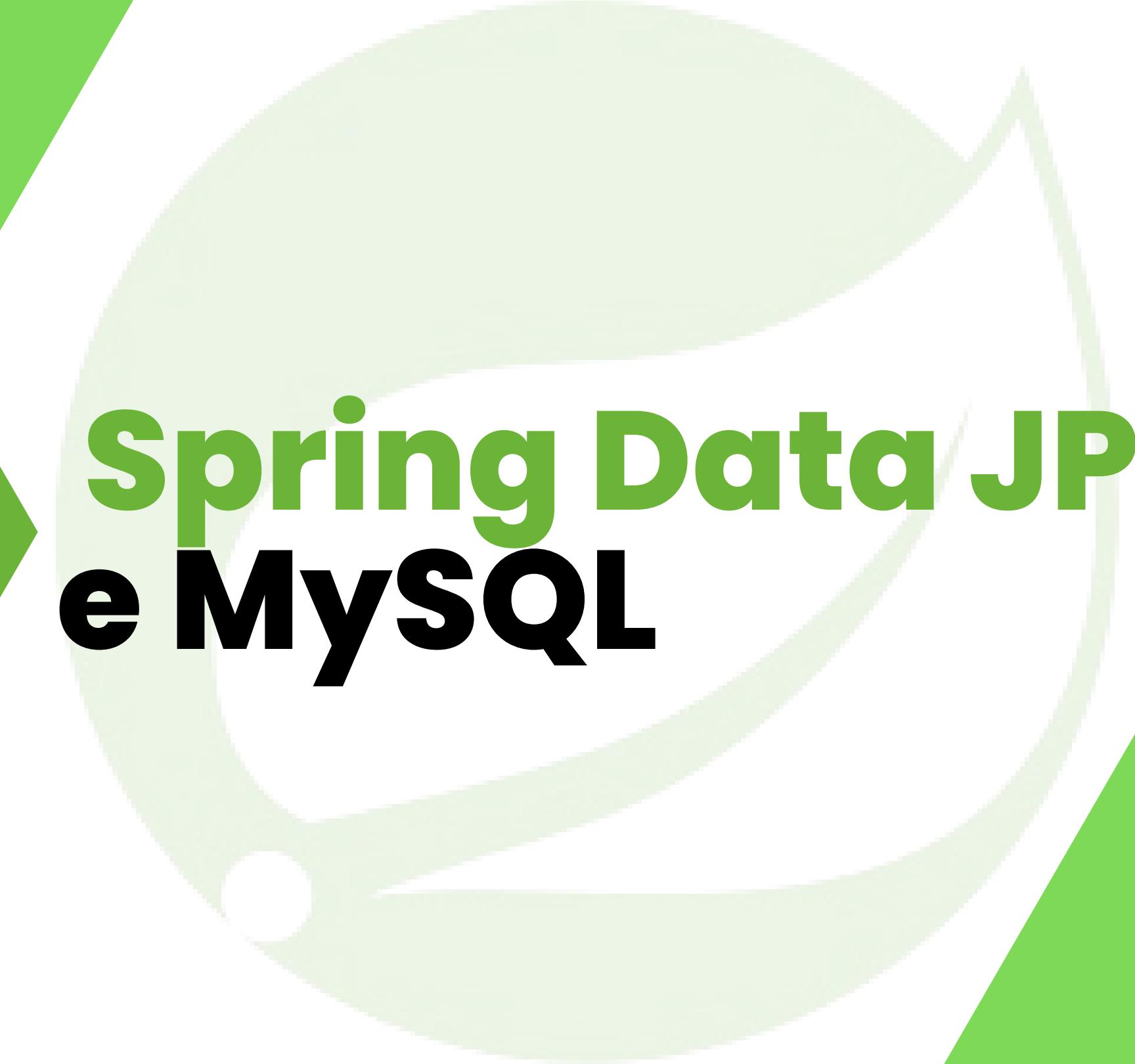
# Arquitetura em camadas

---

-  **controller** → **Controladores REST**
-  **service** → **Regras de negócio**
-  **repository** → **Acesso a banco de dados**
-  **model** → **Modelos de dados**



Persistência  
de Dados



Spring Data JPA  
e MySQL

# O que é JPA?

O **JPA** (**Java Persistence API**) é uma especificação do Java para trabalhar com persistência de dados de forma orientada a objetos. Ele abstrai a complexidade de escrever SQL puro e permite que usemos anotações para mapear entidades no banco de dados



# O que é Hibernate?

O **Hibernate** é a implementação mais popular do JPA. Ele é responsável por traduzir nossas operações Java em comandos SQL



# O QUE É SPRING DATA JPA?

O **Spring Data JPA** é um módulo do Spring que facilita ainda mais o uso do JPA. Com ele, conseguimos reduzir a quantidade de código necessário para realizar operações no banco de dados.



# Configurando Banco de Dados

**Precisamos adicionar a configuração do banco de dados no nosso  
application.properties**

```
spring.datasource.url=jdbc:mysql://localhost:3306/homehero?useSSL=false&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=root
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
server.port=8080
```



# O QUE É ENTIDADE?

- Uma **Entidade** é uma classe Java que representa uma tabela no banco de dados.
- Marcada com **@Entity**.
- Cada atributo representa uma coluna da tabela.
- **@Id**: Indica a chave Primária
- **@GeneratedValue**: Gera o valor automaticamente

```
@Entity  
@Table(name = "cliente")  
@Data  
@NoArgsConstructor  
@AllArgsConstructor  
public class Cliente {  
    @Id  
    @GeneratedValue(strategy = GenerationType.IDENTITY)  
    @Column(name = "cli_id")  
    private Integer id;  
  
    @Column(name = "cli_nome", nullable = false, length = 80)  
    private String nomeCompleto;  
  
    @Column(name = "cli_cpf", nullable = false, unique = true, length = 14)  
    private String cpf;  
  
    @Column(name = "cli_nascimento", nullable = false)  
    private LocalDate dataNascimento;  
  
    @Column(name = "cli_senha", nullable = false, length = 60)  
    private String senha;  
  
    @Column(name = "end_id", nullable = false)  
    private Integer enderecid;  
  
    @Column(name = "cli_email", nullable = false, unique = true, length = 120)  
    private String email;  
  
    @Column(name = "cli_telefone", nullable = false, unique = true, length = 20)  
    private String telefone;  
}
```

# O QUE É ENTIDADE?

- Uma Entidade é uma classe Java que representa uma tabela no banco de dados.
- Marcada com @Entity.
- Cada atributo representa uma coluna da tabela.
- @Id: Indica a chave Primária
- @GeneratedValue: Gera o valor automaticamente



**Python**

The Python logo consists of the word "Python" in a bold, dark blue sans-serif font, centered within a light gray rounded rectangle. To the right of the text is the Python logo icon, which is a yellow and blue abstract shape resembling a pair of interlocking snakes.



# Obrigado!





**Escreva o  
assunto ou  
a sua ideia**

**Elabore rapidamente o que  
você quer discutir.**

# Escreva o assunto ou a sua ideia



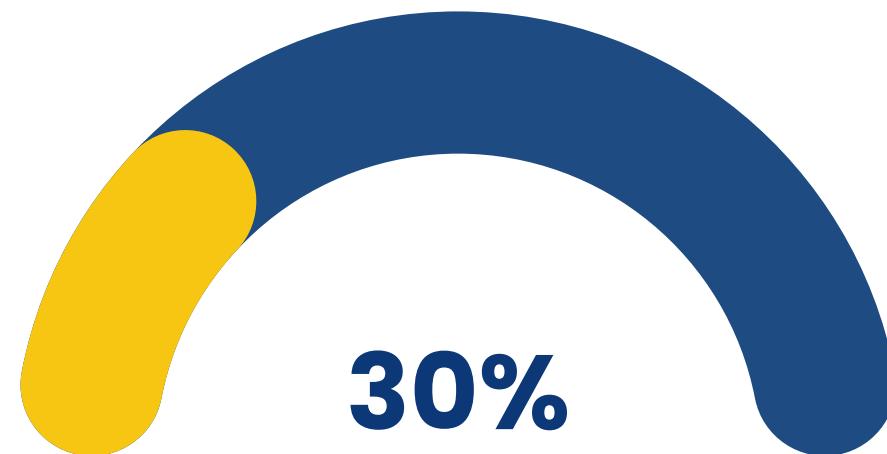
**Adicione um  
ponto principal**



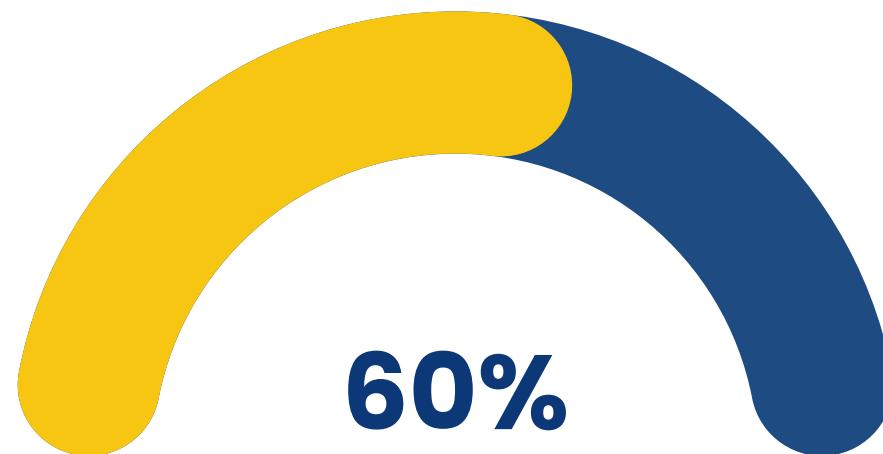
**Adicione um  
ponto principal**



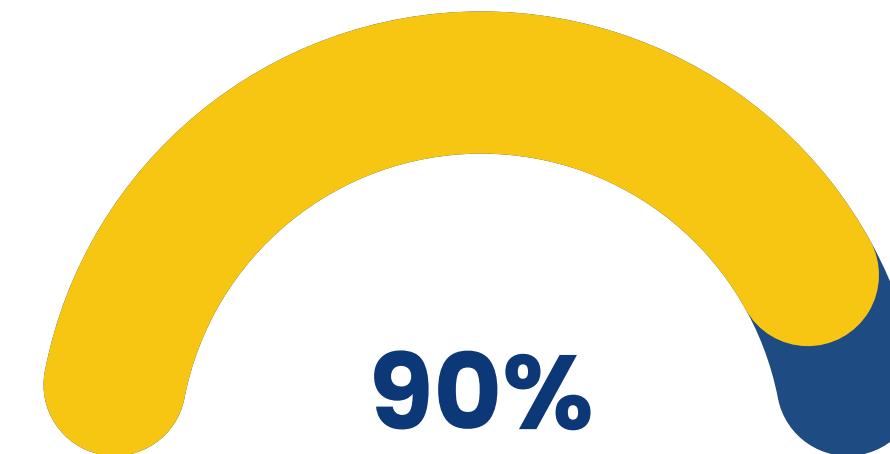
**Adicione um  
ponto principal**



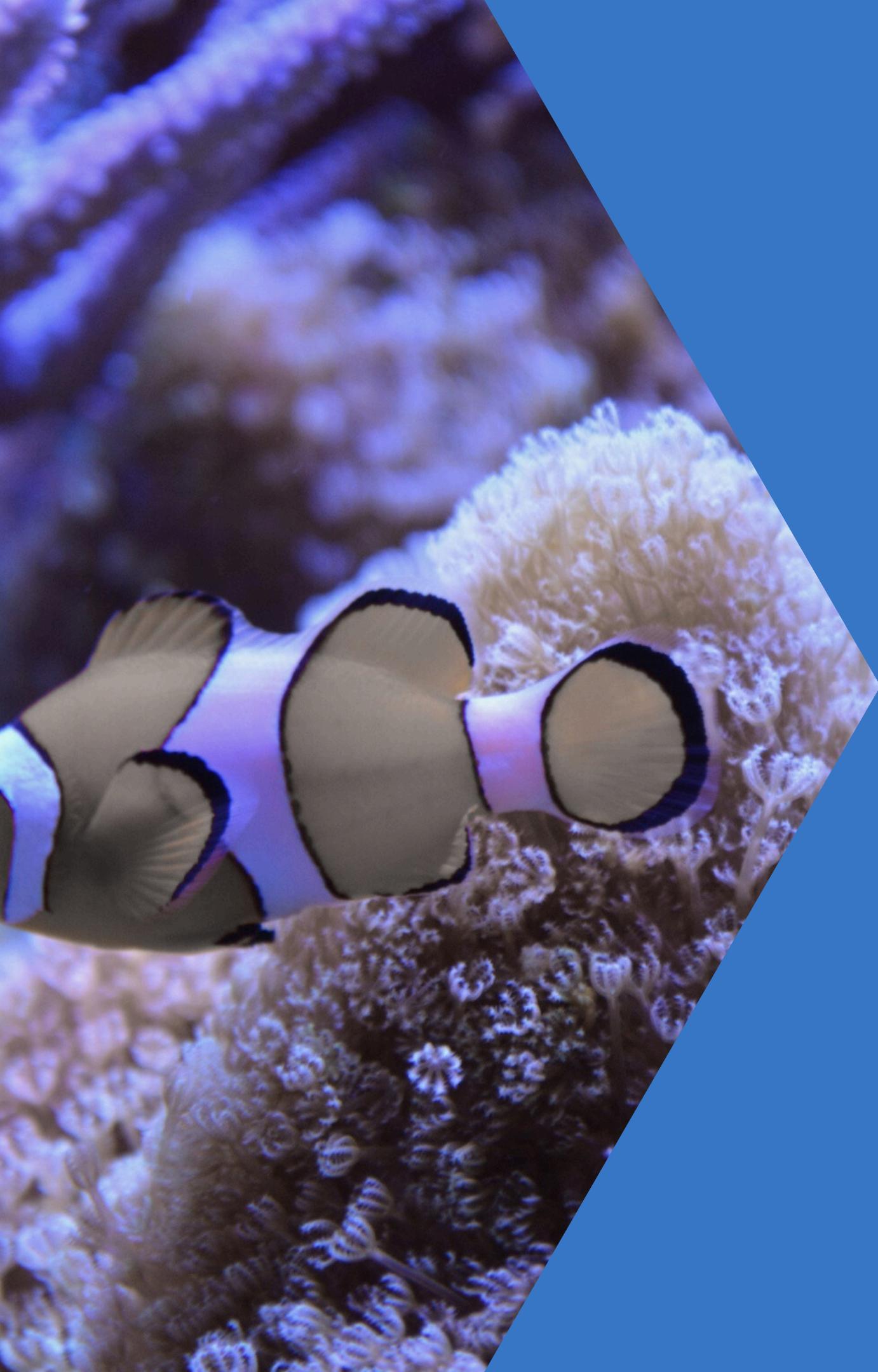
**30%**



**60%**



**90%**



**Adicione um  
título para a  
seção**

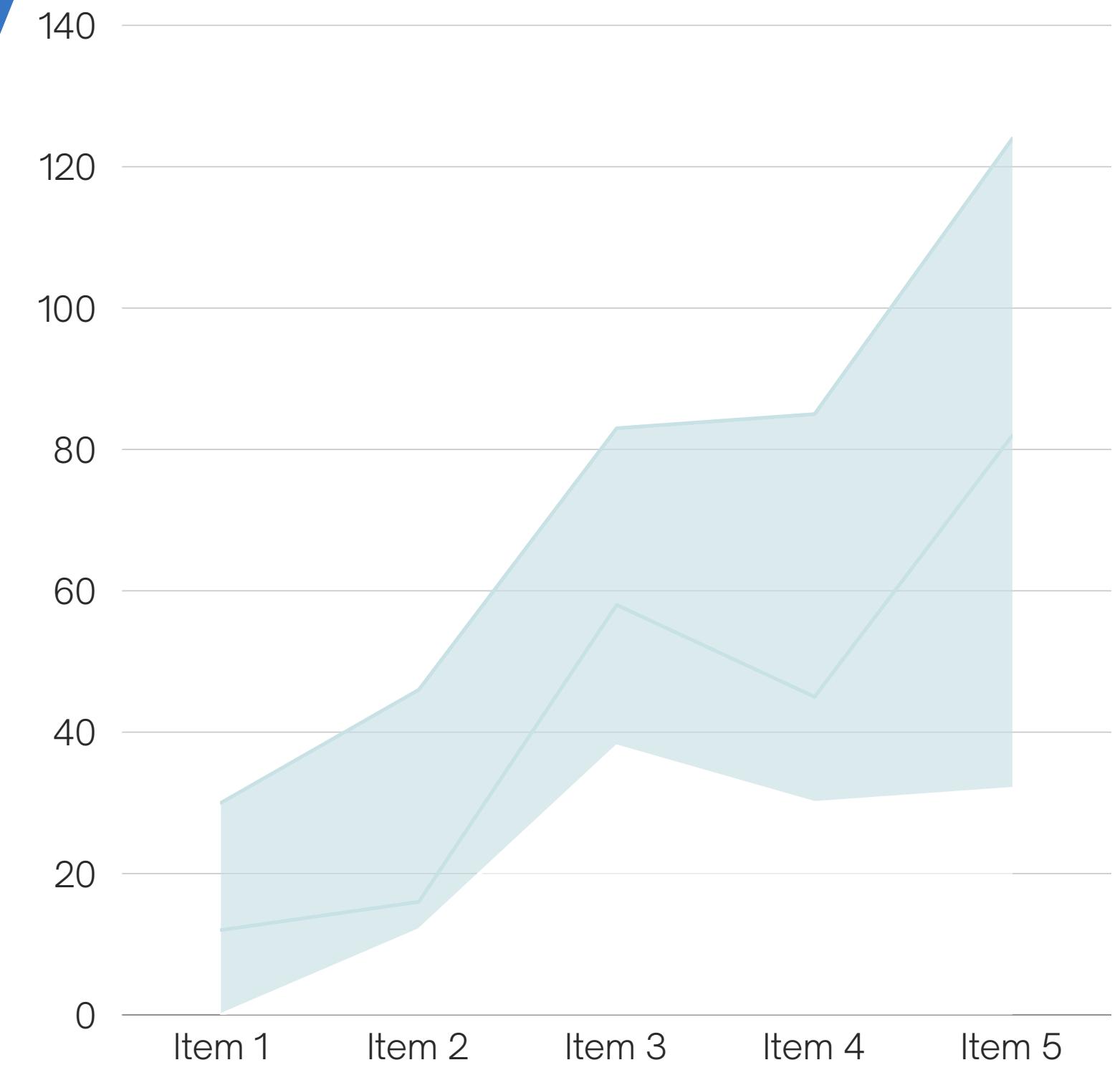
**Escreva  
o assunto  
ou a sua  
ideia**

**Elabore rapidamente o que  
você quer discutir.**



**Escreva o  
assunto ou  
a sua ideia**

**Elabore rapidamente  
o que você quer discutir.**



# **Escreva o assunto ou a sua ideia**



## **Adicione um ponto principal**

Elabore rapidamente o  
que você quer discutir.



## **Adicione um ponto principal**

Elabore rapidamente o  
que você quer discutir.



**Viva o momento.  
Viva para o seu  
propósito. Que cada  
dia seja inesquecível.**

**Elabore rapidamente o que  
você quer discutir.**

A close-up photograph of several fish swimming in clear blue water. The fish are silvery-grey with dark fins and tails. They are positioned in the upper left corner of the slide, partially obscured by a large, semi-transparent yellow arrow shape that points from the top left towards the center.

**Adicione  
um título  
para a  
seção**



# **Escreva o assunto ou a sua ideia**



**Adicione um  
ponto principal**  
**Elabore**  
**rapidamente o que**  
**você quer discutir.**



**Adicione um  
ponto principal**  
**Elabore**  
**rapidamente o que**  
**você quer discutir.**

# Página de recursos

**Use estes elementos  
na sua apresentação  
do Canva.**

