if (repository.existsByTitulo(serie.getTitulo())) {  
 System.*out*.println("Série já existe no banco de dados.");  
 System.*out*.println("Você pode buscar pelos episodios - OPCAO 2");  
} else {  
 repository.save(serie);  
}  
System.*out*.println(dados);

package br.com.alura.screenmatch.repository;  
  
import br.com.alura.screenmatch.model.Serie;  
import org.springframework.data.jpa.repository.JpaRepository;  
  
public interface SerieRepository extends JpaRepository<Serie, Long> {  
 // primeiro parametro quem é a entidade que estamos persistindo  
 // segundo parametro o tipo do ID que nesse caso definimos como LONG  
  
 boolean existsByTitulo(String titulo);  
}

CLASSE COMPLETA

package br.com.alura.screenmatch.principal;  
  
public class Principal {  
  
 private Scanner leitura = new Scanner(System.*in*);  
 private ConsumoApi consumo = new ConsumoApi();  
 private ConverteDados conversor = new ConverteDados();  
 private final String ENDERECO = "https://www.omdbapi.com/?t=";  
 private final String API\_KEY = "&apikey=6585022c";  
  
 // Criando minha lista de series  
 private List<DadosSerie> dadosSeries = new ArrayList<>();  
  
 private SerieRepository repository;  
  
 private List<Serie> series = new ArrayList<>();  
  
 private String serieBuscada;  
  
 // \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
 public Principal(SerieRepository repository) {  
 this.repository = repository;  
 }  
  
 public void exibeMenu() {  
 var menu = """  
 1 - Buscar séries  
 2 - Buscar episódios  
 3 - Lista de series buscadas  
   
 0 - Sair   
 """;  
  
  
 var opcao = 1;  
 while (opcao != 0) {  
 System.*out*.println(menu);  
 opcao = leitura.nextInt();  
 leitura.nextLine();  
  
  
 switch (opcao) {  
 case 1:  
 buscarSerieWeb();  
 break;  
 case 2:  
 buscarEpisodioPorSerie();  
 break;  
 case 3:  
 buscarListaDeSeries();  
 break;  
 case 0:  
 System.*out*.println("Saindo...");  
 break;  
 default:  
 System.*out*.println("Opção inválida");  
 }  
 }  
 }  
  
  
 **private void buscarSerieWeb() {  
 DadosSerie dados = getDadosSerie();  
 Serie serie = new Serie(dados);  
 // Antes de salvar eu quero ve se existe.  
 if (repository.existsByTitulo(serie.getTitulo())) {  
 System.*out*.println("Série já existe no banco de dados.");  
 System.*out*.println("Você pode buscar pelos episodios - OPCAO 2");  
 } else {  
 repository.save(serie);  
 }  
 System.*out*.println(dados);  
  
 }**  
  
 private DadosSerie getDadosSerie() {  
 System.*out*.println("Digite o nome da série para busca");  
 serieBuscada = leitura.nextLine();  
 var json = consumo.obterDados(ENDERECO + serieBuscada.replace(" ", "+") + API\_KEY);  
 DadosSerie dados = conversor.obterDados(json, DadosSerie.class);  
 return dados;  
 }  
  
 private void buscarEpisodioPorSerie(){  
  
 try {  
// DadosSerie dadosSerie = getDadosSerie();  
 buscarListaDeSeries();  
 System.*out*.println("Escolha uma série pelo nome: ");  
 var nomeSerie = leitura.nextLine();  
  
  
 Optional<Serie> serie = series.stream()  
 .filter(s -> s.getTitulo().toLowerCase().contains(nomeSerie.toLowerCase()))  
 .findFirst();  
  
 if (serie.isPresent()) {  
 var serieEncontrada = serie.get();  
  
 List<DadosTemporada> temporadas = new ArrayList<>();  
  
 Integer totalTemporadas = serieEncontrada.getTotalTemporadas();  
 if (totalTemporadas == null) {  
 System.*out*.println("Titulo não é uma serie ou não tem temporadas");  
 } else {  
 for (int i = 1; i <= serieEncontrada.getTotalTemporadas(); i++) {  
 var json = consumo.obterDados(ENDERECO + serieEncontrada.getTitulo().replace(" ", "+") + "&season=" + i + API\_KEY);  
 DadosTemporada dadosTemporada = conversor.obterDados(json, DadosTemporada.class);  
 temporadas.add(dadosTemporada);  
 }  
 }  
 temporadas.forEach(System.*out*::println);  
  
 List<Episodio> episodios = temporadas.stream()  
 .flatMap(d -> d.episodios().stream()  
 .map(e -> new Episodio(d.numero(), e)))  
 .collect(Collectors.*toList*());  
 serieEncontrada.setEpisodios(episodios);  
 repository.save(serieEncontrada);  
  
 }else {  
 System.*out*.println("Série não encontrada.");  
 }  
 }catch (Exception e){  
 System.*out*.println("""  
 Tem certeza que isso é uma serie?  
 Só podemos localizar uma serie  
 """ + e.getMessage() + "\n");  
 }  
  
 }  
  
 private void buscarListaDeSeries() {  
 series = repository.findAll();  
  
 series.stream()  
 .sorted(Comparator.*comparing*(Serie::getTitulo))  
 .forEach(System.*out*::println);  
 }  
}