Abrir o VSCODE

Instalar as extensões

Texto

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente

Digite CTRL+SHIFT+P

Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Uma imagem contendo Retângulo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

2. Configuração do application.properties

No arquivo src/main/resources/application.properties, adicione as configurações para conexão ao banco de dados PostgreSQL:

properties

Copiar código

spring.datasource.url=jdbc:postgresql://localhost:5432/suabase

spring.datasource.username=seu\_usuario

spring.datasource.password=sua\_senha

spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect

**3. Criação das Entidades**

**package com.example.demo.entity;**

**import jakarta.persistence.\*;**

**import lombok.Data;**

**@Data**

**@Entity**

**@Table(name = "tb\_produto")**

**public class Produto {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private Long id;**

**private String descricao;**

**private String nome;**

**private String urlImagem;**

**private Double preco;**

**}**

**package com.example.demo.entity;**

**import jakarta.persistence.\*;**

**import lombok.Data;**

**import java.time.LocalDate;**

**@Data**

**@Entity**

**@Table(name = "tb\_usuario")**

**public class Usuario {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private Long id;**

**private String email;**

**private String nome;**

**private String senha;**

**private String telefone;**

**private LocalDate dataAniversario;**

**}**

**package com.example.demo.entity;**

**import jakarta.persistence.\*;**

**import lombok.Data;**

**import java.time.LocalDate;**

**import java.util.ArrayList;**

**import java.util.List;**

**import com.fasterxml.jackson.annotation.JsonManagedReference;**

**@Data**

**@Entity**

**@Table(name = "tb\_pedido")**

**public class Pedido {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private Long id;**

**@ManyToOne**

**@JoinColumn(name = "id\_usuario")**

**private Usuario usuario;**

**private String status;**

**private LocalDate data;**

**@OneToMany(mappedBy = "pedido", cascade = CascadeType.ALL)**

**@JsonManagedReference  // Define o lado "pai" da relação**

**private List<ItemPedido> itens = new ArrayList<>();  // Inicializando a lista;**

**}**

**package com.example.demo.entity;**

**import com.fasterxml.jackson.annotation.JsonBackReference;**

**import jakarta.persistence.\*;**

**import lombok.Data;**

**@Data**

**@Entity**

**@Table(name = "td\_item\_pedido")**

**public class ItemPedido {**

**@Id**

**@GeneratedValue(strategy = GenerationType.IDENTITY)**

**private Long id;**

**@ManyToOne**

**@JoinColumn(name = "id\_pedido")**

**@JsonBackReference  // Define o lado "filho" da relação**

**private Pedido pedido;**

**@ManyToOne**

**@JoinColumn(name = "id\_produto")**

**private Produto produto;**

**private Integer qtde;**

**private Double preco;**

**}**

**4. Criação dos DTOs**

**package com.example.demo.dto;**

**import lombok.Data;**

**@Data**

**public class ProdutoDTO {**

**private Long id;**

**private String descricao;**

**private String nome;**

**private String urlImagem;**

**private Double preco;**

**}**

**package com.example.demo.dto;**

**import lombok.Data;**

**import java.time.LocalDate;**

**@Data**

**public class UsuarioDTO {**

**private Long id;**

**private String email;**

**private String nome;**

**private String telefone;**

**private String senha;**

**private LocalDate dataAniversario;**

**}**

**package com.example.demo.dto;**

**import lombok.Data;**

**import java.time.LocalDate;**

**import java.util.List;**

**@Data**

**public class PedidoDTO {**

**private Long idUsuario;**

**private String status;**

**private LocalDate data;**

**private List<ItemPedidoDTO> itens;**

**}**

**package com.example.demo.dto;**

**import lombok.Data;**

**@Data**

**public class ItemPedidoDTO {**

**private Long idProduto;**

**private Integer qtde;**

**private Double preco;**

**}**

**5. Criação dos Repositórios**

**package com.example.demo.repository;**

**import com.example.demo.entity.Usuario;**

**import java.util.Optional;**

**import org.springframework.data.jpa.repository.JpaRepository;**

**public interface UsuarioRepository extends JpaRepository<Usuario, Long> {**

**// Método para encontrar um usuário por email**

**Optional<Usuario> findByEmail(String email);**

**}**

**package com.example.demo.repository;**

**import com.example.demo.entity.Produto;**

**import org.springframework.data.jpa.repository.JpaRepository;**

**public interface ProdutoRepository extends JpaRepository<Produto, Long> {}**

**package com.example.demo.repository;**

**import com.example.demo.entity.Pedido;**

**import org.springframework.data.jpa.repository.JpaRepository;**

**public interface PedidoRepository extends JpaRepository<Pedido, Long> {}**

**package com.example.demo.repository;**

**import com.example.demo.entity.ItemPedido;**

**import org.springframework.data.jpa.repository.JpaRepository;**

**public interface ItemPedidoRepository extends JpaRepository<ItemPedido, Long> {}**

**6. Criação dos Services**

**package com.example.demo.service;**

**import com.example.demo.dto.ProdutoDTO;**

**import com.example.demo.entity.Produto;**

**import com.example.demo.repository.ProdutoRepository;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.stereotype.Service;**

**import java.util.List;**

**import java.util.Optional;**

**@Service**

**public class ProdutoService {**

**@Autowired**

**private ProdutoRepository produtoRepository;**

**public List<Produto> listarTodos() {**

**return produtoRepository.findAll();**

**}**

**public Optional<Produto> buscarPorId(Long id) {**

**return produtoRepository.findById(id);**

**}**

**public Produto salvar(ProdutoDTO produtoDTO) {**

**Produto produto = new Produto();**

**produto.setNome(produtoDTO.getNome());**

**produto.setDescricao(produtoDTO.getDescricao());**

**produto.setUrlImagem(produtoDTO.getUrlImagem());**

**produto.setPreco(produtoDTO.getPreco());**

**return produtoRepository.save(produto);**

**}**

**public void excluir(Long id) {**

**if (produtoRepository.existsById(id)) {**

**produtoRepository.deleteById(id);**

**} else {**

**throw new IllegalArgumentException("Produto não encontrado.");**

**}**

**}**

**}**

**package com.example.demo.service;**

**import com.example.demo.dto.UsuarioDTO;**

**import com.example.demo.entity.Usuario;**

**import com.example.demo.repository.UsuarioRepository;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.stereotype.Service;**

**import java.util.List;**

**import java.util.Optional;**

**@Service**

**public class UsuarioService {**

**@Autowired**

**private UsuarioRepository usuarioRepository;**

**// Listar todos os usuários**

**public List<Usuario> listarTodos() {**

**return usuarioRepository.findAll();**

**}**

**// Buscar um usuário por ID**

**public Optional<Usuario> buscarPorId(Long id) {**

**return usuarioRepository.findById(id);**

**}**

**// Buscar um usuário por email**

**public Optional<Usuario> buscarPorEmail(String email) {**

**return usuarioRepository.findByEmail(email);**

**}**

**// Salvar ou atualizar um usuário**

**public Usuario salvarOuAtualizar(UsuarioDTO usuarioDTO) {**

**Usuario usuario = new Usuario();**

**usuario.setNome(usuarioDTO.getNome());**

**usuario.setEmail(usuarioDTO.getEmail());**

**usuario.setSenha(usuarioDTO.getSenha());**

**usuario.setTelefone(usuarioDTO.getTelefone());**

**usuario.setDataAniversario(usuarioDTO.getDataAniversario());**

**return usuarioRepository.save(usuario);**

**}**

**// Excluir um usuário por ID**

**public void excluir(Long id) {**

**if (usuarioRepository.existsById(id)) {**

**usuarioRepository.deleteById(id);**

**} else {**

**throw new IllegalArgumentException("Usuário não encontrado.");**

**}**

**}**

**}**

**package com.example.demo.service;**

**import com.example.demo.dto.ItemPedidoDTO;**

**import com.example.demo.dto.PedidoDTO;**

**import com.example.demo.entity.ItemPedido;**

**import com.example.demo.entity.Pedido;**

**import com.example.demo.entity.Produto;**

**import com.example.demo.entity.Usuario;**

**import com.example.demo.repository.PedidoRepository;**

**import com.example.demo.repository.ProdutoRepository;**

**import com.example.demo.repository.UsuarioRepository;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.stereotype.Service;**

**import java.util.List;**

**import java.util.Optional;**

**@Service**

**public class PedidoService {**

**@Autowired**

**private PedidoRepository pedidoRepository;**

**@Autowired**

**private UsuarioRepository usuarioRepository;**

**@Autowired**

**private ProdutoRepository produtoRepository;**

**public List<Pedido> listarTodos() {**

**return pedidoRepository.findAll();**

**}**

**public Optional<Pedido> buscarPorId(Long id) {**

**return pedidoRepository.findById(id);**

**}**

**public Pedido salvar(PedidoDTO pedidoDTO) {**

**Pedido pedido = new Pedido();**

**// Busca o usuário**

**Usuario usuario = usuarioRepository.findById(pedidoDTO.getIdUsuario())**

**.orElseThrow(() -> new IllegalArgumentException("Usuário não encontrado"));**

**pedido.setUsuario(usuario);**

**pedido.setStatus(pedidoDTO.getStatus());**

**pedido.setData(pedidoDTO.getData());**

**// Constrói os itens do pedido**

**for (ItemPedidoDTO itemDTO : pedidoDTO.getItens()) {**

**Produto produto = produtoRepository.findById(itemDTO.getIdProduto())**

**.orElseThrow(() -> new IllegalArgumentException("Produto não encontrado"));**

**ItemPedido itemPedido = new ItemPedido();**

**itemPedido.setProduto(produto);**

**itemPedido.setQtde(itemDTO.getQtde());**

**itemPedido.setPreco(itemDTO.getPreco());**

**itemPedido.setPedido(pedido);**

**pedido.getItens().add(itemPedido);**

**}**

**return pedidoRepository.save(pedido);**

**}**

**public void excluir(Long id) {**

**if (pedidoRepository.existsById(id)) {**

**pedidoRepository.deleteById(id);**

**} else {**

**throw new IllegalArgumentException("Pedido não encontrado.");**

**}**

**}**

**}**

**7. Criação dos Controllers**

**package com.example.demo.controller;**

**import com.example.demo.dto.ProdutoDTO;**

**import com.example.demo.entity.Produto;**

**import com.example.demo.service.ProdutoService;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.http.HttpStatus;**

**import org.springframework.http.ResponseEntity;**

**import org.springframework.web.bind.annotation.\*;**

**import java.util.List;**

**import java.util.Optional;**

**@RestController**

**@RequestMapping("/api/produtos")**

**public class ProdutoController {**

**@Autowired**

**private ProdutoService produtoService;**

**@GetMapping**

**public ResponseEntity<List<Produto>> listarTodos() {**

**return new ResponseEntity<>(produtoService.listarTodos(), HttpStatus.OK);**

**}**

**@GetMapping("/{id}")**

**public ResponseEntity<?> buscarPorId(@PathVariable Long id) {**

**Optional<Produto> produto = produtoService.buscarPorId(id);**

**return produto.isPresent()**

**? new ResponseEntity<>(produto.get(), HttpStatus.OK)**

**: new ResponseEntity<>("Produto não encontrado", HttpStatus.NOT\_FOUND);**

**}**

**@PostMapping**

**public ResponseEntity<Produto> salvar(@RequestBody ProdutoDTO produtoDTO) {**

**return new ResponseEntity<>(produtoService.salvar(produtoDTO), HttpStatus.CREATED);**

**}**

**@DeleteMapping("/{id}")**

**public ResponseEntity<?> excluir(@PathVariable Long id) {**

**try {**

**produtoService.excluir(id);**

**return new ResponseEntity<>(HttpStatus.NO\_CONTENT);**

**} catch (IllegalArgumentException e) {**

**return new ResponseEntity<>("Erro ao excluir: " + e.getMessage(), HttpStatus.NOT\_FOUND);**

**}**

**}**

**}**

**package com.example.demo.controller;**

**import com.example.demo.dto.UsuarioDTO;**

**import com.example.demo.entity.Usuario;**

**import com.example.demo.service.UsuarioService;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.http.HttpStatus;**

**import org.springframework.http.ResponseEntity;**

**import org.springframework.web.bind.annotation.\*;**

**import java.util.List;**

**import java.util.Optional;**

**@RestController**

**@RequestMapping("/api/usuarios")**

**public class UsuarioController {**

**@Autowired**

**private UsuarioService usuarioService;**

**// Endpoint para listar todos os usuários**

**@GetMapping**

**public ResponseEntity<List<Usuario>> listarTodos() {**

**List<Usuario> usuarios = usuarioService.listarTodos();**

**return new ResponseEntity<>(usuarios, HttpStatus.OK);**

**}**

**// Endpoint para buscar um usuário pelo ID**

**@GetMapping("/{id}")**

**public ResponseEntity<?> buscarPorId(@PathVariable Long id) {**

**Optional<Usuario> usuario = usuarioService.buscarPorId(id);**

**return usuario.isPresent()**

**? new ResponseEntity<>(usuario.get(), HttpStatus.OK)**

**: new ResponseEntity<>("Usuário não encontrado", HttpStatus.NOT\_FOUND);**

**}**

**// Endpoint para buscar um usuário pelo email**

**@GetMapping("/email")**

**public ResponseEntity<?> buscarPorEmail(@RequestParam String email) {**

**Optional<Usuario> usuario = usuarioService.buscarPorEmail(email);**

**return usuario.isPresent()**

**? new ResponseEntity<>(usuario.get(), HttpStatus.OK)**

**: new ResponseEntity<>("Usuário não encontrado", HttpStatus.NOT\_FOUND);**

**}**

**// Endpoint para salvar ou atualizar um usuário**

**@PostMapping**

**public ResponseEntity<Usuario> salvarOuAtualizar(@RequestBody UsuarioDTO usuarioDTO) {**

**Usuario usuario = usuarioService.salvarOuAtualizar(usuarioDTO);**

**return new ResponseEntity<>(usuario, HttpStatus.CREATED);**

**}**

**// Endpoint para excluir um usuário pelo ID**

**@DeleteMapping("/{id}")**

**public ResponseEntity<?> excluir(@PathVariable Long id) {**

**try {**

**usuarioService.excluir(id);**

**return new ResponseEntity<>(HttpStatus.NO\_CONTENT);**

**} catch (IllegalArgumentException e) {**

**return new ResponseEntity<>("Erro ao excluir: " + e.getMessage(), HttpStatus.NOT\_FOUND);**

**}**

**}**

**}**

**package com.example.demo.controller;**

**import com.example.demo.dto.PedidoDTO;**

**import com.example.demo.entity.Pedido;**

**import com.example.demo.service.PedidoService;**

**import org.springframework.beans.factory.annotation.Autowired;**

**import org.springframework.http.HttpStatus;**

**import org.springframework.http.ResponseEntity;**

**import org.springframework.web.bind.annotation.\*;**

**import java.util.List;**

**import java.util.Optional;**

**@RestController**

**@RequestMapping("/api/pedidos")**

**public class PedidoController {**

**@Autowired**

**private PedidoService pedidoService;**

**@GetMapping**

**public ResponseEntity<List<Pedido>> listarTodos() {**

**return new ResponseEntity<>(pedidoService.listarTodos(), HttpStatus.OK);**

**}**

**@GetMapping("/{id}")**

**public ResponseEntity<?> buscarPorId(@PathVariable Long id) {**

**Optional<Pedido> pedido = pedidoService.buscarPorId(id);**

**return pedido.isPresent()**

**? new ResponseEntity<>(pedido.get(), HttpStatus.OK)**

**: new ResponseEntity<>("Pedido não encontrado", HttpStatus.NOT\_FOUND);**

**}**

**@PostMapping**

**public ResponseEntity<Pedido> salvar(@RequestBody PedidoDTO pedidoDTO) {**

**return new ResponseEntity<>(pedidoService.salvar(pedidoDTO), HttpStatus.CREATED);**

**}**

**@DeleteMapping("/{id}")**

**public ResponseEntity<?> excluir(@PathVariable Long id) {**

**try {**

**pedidoService.excluir(id);**

**return new ResponseEntity<>(HttpStatus.NO\_CONTENT);**

**} catch (IllegalArgumentException e) {**

**return new ResponseEntity<>("Erro ao excluir: " + e.getMessage(), HttpStatus.NOT\_FOUND);**

**}**

**}**

**}**

**Criando objetos para testar as APIs**

**Exemplo 1: Criar um Usuário**

**json**

**Copiar código**

**{**

**"nome": "Maria Oliveira",**

**"email": "maria.oliveira@example.com",**

**"telefone": "21987654321",**

**"dataAniversario": "1988-07-12",**

**"senha": "senhaSegura123"**

**}**

**Exemplo 2: Criar um Usuário**

**json**

**Copiar código**

**{**

**"nome": "Carlos Andrade",**

**"email": "carlos.andrade@example.com",**

**"telefone": "31912345678",**

**"dataAniversario": "1992-11-25",**

**"senha": "minhaSenhaForte456"**

**}**

**Exemplo 1: Criar um Produto**

**json**

**Copiar código**

**{**

**"nome": "Cadeira Gamer XPro",**

**"descricao": "Cadeira gamer ergonômica com ajuste de altura, inclinação de 180º e apoio para os pés.",**

**"urlImagem": "https://exemplo.com/imagens/cadeira\_gamer\_xpro.jpg",**

**"preco": 899.90**

**}**

**Exemplo 2: Criar um Produto**

**json**

**Copiar código**

**{**

**"nome": "Smartwatch Fit 200",**

**"descricao": "Smartwatch com monitor de batimentos cardíacos, GPS integrado e resistência à água.",**

**"urlImagem": "https://exemplo.com/imagens/smartwatch\_fit200.jpg",**

**"preco": 599.50**

**}**

**Exemplo 3: Criar um Produto**

**json**

**Copiar código**

**{**

**"nome": "Monitor Ultrawide 34\"",**

**"descricao": "Monitor ultrawide de 34 polegadas com resolução QHD e taxa de atualização de 144Hz.",**

**"urlImagem": "https://exemplo.com/imagens/monitor\_ultrawide\_34.jpg",**

**"preco": 1799.00**

**}**

**Exemplo 4: Criar um Produto**

**json**

**Copiar código**

**{**

**"nome": "Teclado Mecânico RGB",**

**"descricao": "Teclado mecânico com switches blue, iluminação RGB e teclas anti-ghosting.",**

**"urlImagem": "https://exemplo.com/imagens/teclado\_mecanico\_rgb.jpg",**

**"preco": 349.99**

**}**

**Exemplo de Objeto JSON para Criar um Pedido com Data**

**json**

**Copiar código**

**{**

**"idUsuario": 1,**

**"status": "PROCESSANDO",**

**"data": "2024-10-06",**

**"itens": [**

**{**

**"idProduto": 101,**

**"qtde": 2,**

**"preco": 150.00**

**},**

**{**

**"idProduto": 102,**

**"qtde": 1,**

**"preco": 299.99**

**}**

**]**

**}**