@PostMapping  
@Transactional  
public ResponseEntity cadastro(@RequestBody @Valid DadosCadastroPaciente dados, UriComponentsBuilder uriBuilder) {  
 var paciente = new Paciente(dados);  
 pacienteRepository.save(paciente);  
  
 var uri = uriBuilder.path("pacientes/{id}").buildAndExpand(paciente.getId()).toUri();  
 return ResponseEntity.*created*(uri).body(paciente);  
  
}

PostMapping para dizer que vai receber uma requisição do tipo Post

@Transactional = para receber uma transação de escrita.

ResponseEntity é uma classe que vai gerar uma resposta para o navegador

UriComponentsBuilder = gerar o endereço automaticamente

Var paciente = um objeto que recebe os dados do DTO

pacienteRepository = salva os dados do objeto que foi instanciado pela classe/entidade

var uri = é para criar uma variável que contem todos os dados do URL

uriBuilder.path = pega o padrão 127.0.0.1:8080 e adiciona isso aqui “pacientes/{id} ficando 127.0.0.1:8080/pacientes/{id}

buildAndExpand(paciente.getId()).Agora onde é dinâmico no caso do {id} você pega o id que vem la da classe DTO

toUri(); agora você pode criar o endereço com todos esses dados.

Agora sim depois da URL construída você deve retornar esses dados

return ResponseEntity.*created*(uri).body(paciente);

URI = retornar tudo que programamos

Body = retorna os dados do paciente.

No caso aqui estamos criando o endereço pelo ID.