UNIVERSIDADE LUTERANA DO BRASIL

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CAMPUS TORRES

Guilherme Birlem Machado

O desenvolvimento de APIs RESTful é uma prática amplamente utilizada para a criação de sistemas distribuídos e escaláveis. Este texto aborda boas práticas aplicadas no desenvolvimento de uma API RESTful para gerenciar pedidos e fornecedores, destacando princípios como desacoplamento, utilização de padrões HTTP e organização de código.

Boas Práticas em APIs RESTful

APIs RESTful seguem princípios que garantem facilidade de uso, manutenção e integração. No projeto, várias boas práticas foram aplicadas:

1. **Organização por Recursos**
   A API foi estruturada em torno de recursos claros, como `Pedidos` e `Fornecedores`, representados em endpoints específicos:
   - `/api/pedidos`
   - `/api/fornecedores`
2. **Métodos HTTP**
   O uso correto dos métodos HTTP foi seguido:
   - `GET`: Para recuperar informações (listar ou obter um recurso específico).

- `POST`: Para criar novos recursos.
- `PUT`: Para atualizar recursos existentes.
- `DELETE`: Para remover recursos.

3. **Uso de Respostas HTTP Adequadas**
   As respostas foram projetadas para fornecer informações claras:
   - `200 OK`: Operações de sucesso.
   - `201 Created`: Recurso criado com sucesso.
   - `400 Bad Request`: Requisição inválida.
   - `404 Not Found`: Recurso não encontrado.
   - `204 No Content`: Operação realizada sem retorno de dados

```
[HttpPut("{id}")]
0 references
public async Task<ActionResult> Update(int id, Pedido pedido)
{
    if (id != pedido.Id)
        return BadRequest("O ID do pedido não corresponde ao informado na URL.");

    await _pedidoRepository.UpdateAsync(pedido);
    return NoContent();
}
```

   .
4. **Manutenção do Estado Stateless**
   A API foi projetada como *stateless*, garantindo que cada requisição seja independente e contenha todas as informações necessárias.

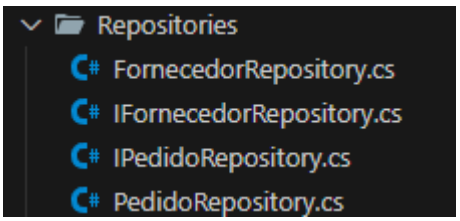
Injeção de Dependência
Para promover o desacoplamento, repositórios foram injetados usando o padrão *Dependency Injection*:

Uso de ORM (Entity Framework)
O Entity Framework foi utilizado para facilitar o mapeamento objeto-relacional e simplificar as operações no banco de dados:

Aplicações das Boas Práticas

Durante o desenvolvimento, as boas práticas foram implementadas para garantir a robustez e escalabilidade da API:

- **Desacoplamento**: Os repositórios (`PedidoRepository` e `FornecedorRepository`) separam a lógica de acesso ao banco da lógica de negócio.




```
public class FornecedorController : ControllerBase
{
    7 references
    private readonly IFornecedorRepository _fornecedorRepository;
```

- **Conformidade com REST**: Os endpoints seguem a semântica REST, tornando a API intuitiva.
- **Validação e Tratamento de Erros**: Respostas claras são retornadas ao cliente para diferentes cenários, como erros de validação e recursos inexistentes.

Conclusão

A adoção dessas boas práticas, como o uso adequado dos métodos HTTP, injeção de dependência e a aplicação de uma arquitetura desacoplada e escalável, são fundamentais para garantir que a API seja robusta, fácil de manter e de integrar com outros sistemas. A utilização do Entity Framework, o design baseado em repositórios e a validação de erros também são elementos cruciais para uma API RESTful de alta qualidade

Foi testado com o inmsonia.