

Domain Driven Design Challenge – Sprint 4

OM Corp.

Matheus Augusto Leite – 99697

Marcelo Hespanhol Dias - 98251

Gabriel Eringer - 99632

Eduardo Tatsuo - 551428

André Sant'Ana - 551575

Yago Marques – 551616

Sumário:

Página 2.....	Sumário
Página 3.....	Descritivo do projeto
Página 4.....	Sobre o produto final
Página 5.....	Release Plan
Página 6.....	Modelo do banco de dados
Páginas 7-10.....	Documentação da API (Endpoints)
Página 11.....	Diagrama de classes UML
Páginas 12-13.....	Procedimentos para executar a API local
Páginas 14-17.....	Protótipo

Descritivo do projeto:

O projeto em questão visa desenvolver um aplicativo de seguro de bicicletas, oferecendo aos usuários uma solução segura, conveniente e confiável para proteger seus meios de transporte. O foco do problema a ser resolvido está na falta de um processo simplificado e eficiente para a contratação de seguros de bicicletas, assim como a automatização do processo de vistoria.

Atualmente, muitos ciclistas enfrentam dificuldades ao buscar proteção para suas bicicletas, pois o processo tradicional de contratação de seguros é complexo, burocrático e muitas vezes dependente de corretores. Além disso, a realização da vistoria para avaliar a condição da bicicleta é um procedimento manual que consome tempo e nem sempre garante a segurança necessária.

Diante desse contexto, o objetivo principal do projeto é oferecer aos usuários uma plataforma intuitiva e eficiente para contratar seguros de bicicletas. Por meio do aplicativo, os usuários poderão realizar todo o processo de contratação de forma direta e simples, eliminando a necessidade de intermediários. Além disso, a automatização da vistoria proporcionará uma experiência mais ágil e segura para o registro das bicicletas, garantindo a integridade do processo.

A justificativa para esse projeto se baseia na crescente popularidade do uso de bicicletas como meio de transporte sustentável e na necessidade de oferecer aos ciclistas uma proteção adequada para seus veículos. Ao simplificar e agilizar o processo de contratação de seguros, o aplicativo tem o objetivo de incentivar mais pessoas a utilizarem bicicletas como meio de transporte, aumentando a segurança e promovendo um estilo de vida saudável.

Além disso, a integração de tecnologias como inteligência artificial, *chatbot* e gamificação permitirá oferecer suporte e engajamento aos usuários, tornando a experiência mais interativa e personalizada. A utilização de *gateways* de pagamento confiáveis e a criptografia dos dados dos clientes garantirão a segurança das transações e a privacidade das informações.

Dessa forma, a entrega do Sprint 2, que compreende a análise de requisitos e a modelagem/projeto do sistema, permitirá validar a compreensão do projeto e estabelecer as bases necessárias para a implementação do aplicativo de seguro de bicicletas.

Sobre o produto final:

O MVP que será entregue no Sprint 4 do projeto concentra-se principalmente no processo de automatização da vistoria de bicicletas por meio de uma API de reconhecimento de imagens com IA. Essa funcionalidade será o cerne da solução proposta, proporcionando maior agilidade e eficiência na análise da condição geral das bicicletas para efeitos de seguro.

O sistema utilizará a tecnologia de *image recognition* para identificar diferentes tipos de bicicletas, modelos, marcas e características. Através da API integrada, será possível avaliar avarias ou danos nas bicicletas. Além disso, os usuários poderão capturar imagens da bicicleta em diferentes ângulos e essas informações serão comparadas com os dados fornecidos no cadastro.

Com base na análise automática realizada pela API, o sistema gerará um relatório de vistoria que será usado para a análise de aprovação. Caso a bicicleta seja aprovada na vistoria, o seguro será emitido.

Embora o MVP se concentre principalmente na automatização da vistoria, vale ressaltar que as demais funcionalidades definidas nas etapas anteriores, como o sistema de pagamento seguro, a autenticação de dois fatores, o *chatbot* para suporte ao cliente e a gamificação do processo de cadastro, estarão presentes na documentação final do projeto.

Essa documentação servirá como uma proposta para a Porto Seguro, oferecendo sugestões e ideias para a melhoria da qualidade do serviço em geral, abrangendo todas as funcionalidades e requisitos definidos nas etapas anteriores do projeto. Assim, o MVP entregue no Sprint 4 será uma etapa crucial para a validação da solução proposta, enquanto as demais funcionalidades serão apresentadas como parte de uma visão mais abrangente e completa do projeto como um todo.

Release Plan:

Sprint 1:

Planejamento/Cronograma: Elaboração de um cronograma organizado para garantir que as entregas sejam realizadas na data correta.

Escopo: Documentação de todas as ideias e conceitos do projeto, garantindo que nada seja esquecido.

Sprint 2:

Análise de Requisitos: Levantamento detalhado dos requisitos funcionais e não funcionais, juntamente com as regras de negócio.

Modelagem de Diagramas: Criação de diagramas, como o diagrama de caso de uso, diagrama de sequência, diagrama de classe, para visualizar e entender o sistema.

Sprint 3:

Implementação: Desenvolvimento das funcionalidades do sistema, como registro de bicicletas, sistema de pagamento, sistema de vistoria automática, *chatbot*, entre outros, conforme os requisitos levantados.

Testes: Realização de testes para verificar a correta implementação das funcionalidades e garantir a qualidade do *software*.

Sprint 4:

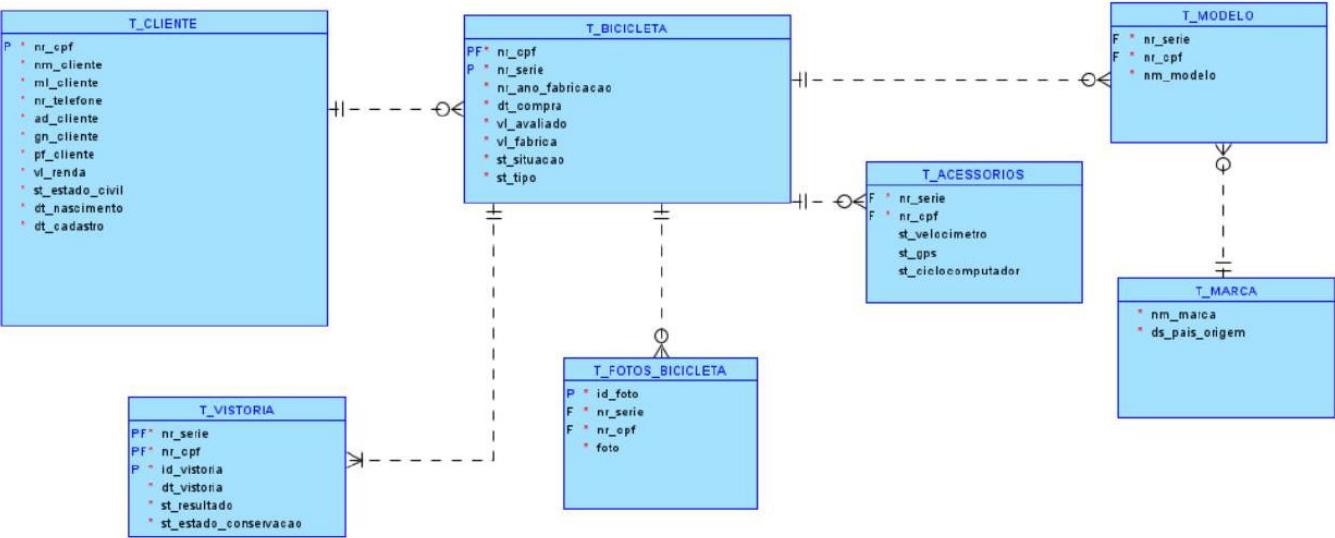
Implantação: Preparação do ambiente de produção e implantação do software em ambiente real utilizando *APIs*.

Testes Finais: Execução de testes adicionais para validar o software em ambiente de produção.

Ajustes Finais: Realização de eventuais ajustes e correções com base nos testes e feedback recebidos.

Lembrando que o detalhamento das funcionalidades e a definição das tarefas específicas a serem realizadas em cada sprint devem ser feitos pela equipe com base nos requisitos e regras de negócio discutidos. O *Release Plan* proposto acima serve como uma diretriz geral para orientar o planejamento do projeto.

Modelo do Banco de Dados:



Documentação da API (Endpoints):

Visão Geral:

Esta API oferece operações de gerenciamento de clientes, bicicletas e seus dados, permitindo a criação, recuperação, atualização e exclusão de registros dos mesmos. É uma API RESTful que opera no banco de dados da Porto Seguro.

É importante observar que o banco de dados da Porto se trata de um banco com tabelas relacionadas entre si - portanto a exclusão de algum item que possua chave estrangeira - apenas é permitida após a exclusão do item filho.

Métodos:

1. Listar Todos:

Método: GET

BASE URL: `http://localhost:8080/portoapi/webapi`

URL: `/clientes` | `/bicicletas` | `/modelos` | `/marcas` | `/acessorios` | `/vistorias` | `/fotos-bicicleta`

Descrição: Obtém uma lista completa dos dados cadastrados no banco de dados.

Resposta de Sucesso: Código 200 (OK)

Exemplo de Resposta:

```
[
  {
    "dtCadastro": "2020-01-01",
    "dtNascimento": "1980-05-20",
    "emailCliente": "joao@email.com",
    "enderecoCliente": "Rua Dante Ribeiro, 123",
    "estadoCivil": "Solteiro",
    "generoCliente": "M",
    "nomeCliente": "João Silva",
    "nrCpf": "12345678901",
    "nrTelefone": "11111111111",
    "renda": 50000.0,
    "tipoCliente": "PF"
  }
]
```

```
    },  
    // Outros clientes...
```

2. Obter cliente, bicicleta ou atributos por CPF, ID ou número de série

Método: GET

URL: /acessorios/{id} | /bicicletas/{nrSerie} | /clientes/{cpf} | /fotos-bicicleta/{id} | /marcas/{id}
| /modelos/{id} | /vistorias{nrSerie}

Descrição: Obtém os detalhes dos acessórios, bicicletas, clientes, fotos-bicicleta, marcas, modelos ou vistorias com base no id, número de série ou CPF fornecidos.

Resposta de Sucesso: Código 200 (OK)

Exemplo de Resposta:

```
{  
  "dtCadastro": "2021-02-28",  
  "dtNascimento": "1992-12-10",  
  "emailCliente": "carlos@email.com",  
  "enderecoCliente": "Rua Crepúsculo, 789",  
  "estadoCivil": "Solteiro",  
  "generoCliente": "M",  
  "nomeCliente": "Carlos Santos",  
  "nrCpf": "11111111111",  
  "nrTelefone": "999999999",  
  "renda": 55000.0,  
  "tipoCliente": "PF"  
}
```

3. Cadastrar um novo cliente, bicicleta, marcas, modelos, vistorias, acessórios ou fotos das bicicletas.

Método: POST

URL: /acessorios/{id} | /bicicletas/{nrSerie} | /clientes/{cpf} | /fotos-bicicleta/{id} | /marcas/{id}
| /modelos/{id} | /vistorias{nrSerie}

Descrição: Cadastrar com os detalhes fornecidos.

Corpo da Requisição:

```
{
```



```
"dtCadastro": "2016-09-15",
"dtNascimento": "1982-09-30",
"emailCliente": "pedro@email.com",
"enderecoCliente": "Av. Paulista, 123",
"estadoCivil": "Casado",
"generoCliente": "M",
"nomeCliente": "Ana Paula",
"nrCpf": "12345678910",
"nrTelefone": "11777777777",
"renda": 75000.0,
"tipoCliente": "PF"
}
```

Resposta de Sucesso: Código 201 (Created)

4. Atualizar um cliente, bicicleta, marca, modelo, vistoria, acessório ou foto das bicicletas.

Método: PUT

URL: /acessorios/{id} | /bicicletas/{nrSerie} | /clientes/{cpf} | /fotos-bicicleta/{id} | /marcas/{id}
| /modelos/{id} | /vistorias{nrSerie}

Descrição: Atualiza os detalhes com base no ID, número de série ou CPF fornecido.

Corpo da Requisição (campos a serem atualizados):

```
{
  "dtCadastro": "2016-09-15",
  "dtNascimento": "1982-09-30",
  "emailCliente": "pedro@email.com",
  "enderecoCliente": "Av. Paulista, 123",
  "estadoCivil": "Casado",
  "generoCliente": "M",
  "nomeCliente": "Ana Paula",
  "nrCpf": "12345678910",
  "nrTelefone": "11777777777",
  "renda": 75000.0,
  "tipoCliente": "PF"
}
```

}

Resposta de Sucesso: Código 200 (OK)

5. Excluir um cliente, bicicleta, marca, modelo, vistoria, acessório ou foto das bicicletas.

Método: DELETE

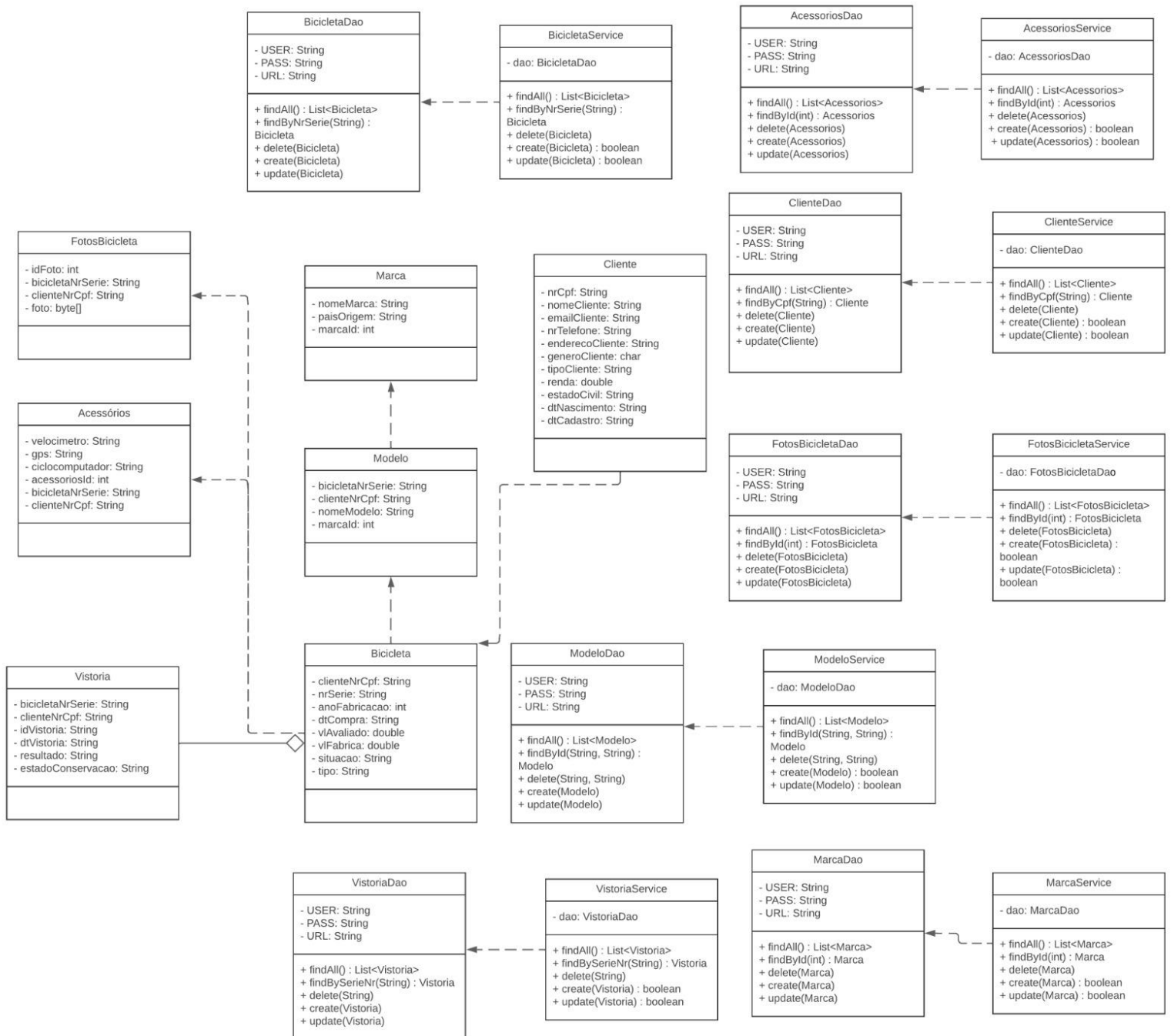
URL: /acessorios/{id} | /bicicletas/{nrSerie} | /clientes/{cpf} | /fotos-bicicleta/{id} | /marcas/{id}
| /modelos/{id} | /vistorias/{nrSerie}

Parâmetros de URL: {id}: O ID do personagem que deseja excluir.

Descrição: Exclui um cliente, bicicleta, marca, modelo, vistoria, acessório ou foto das bicicletas com base no ID, número de série ou CPF fornecido.

Resposta de Sucesso: Código 204 (No Content)

Diagrama de Classes UML:



Procedimentos para Executar a API Local:

Pré-requisitos:

Antes de iniciar os procedimentos, certifique-se de que os seguintes softwares estão instalados em sua máquina:

- Eclipse IDE
- Apache Tomcat 10.1
- Insomnia

Instruções para Instalação:

Apache Tomcat 10.1:

- Acesse o site oficial do Apache Tomcat.
- Na seção de downloads, escolha a versão 10.1 e baixe o arquivo compactado adequado para o seu sistema operacional.
- Após o download, descompacte o arquivo em um diretório de sua escolha.

Eclipse IDE:

- Acesse o site oficial do Eclipse.
- Na seção de downloads, escolha a versão "Eclipse IDE for Java EE Developers".
- Baixe o instalador correspondente ao seu sistema operacional.
- Execute o instalador e siga as instruções na tela para concluir a instalação.

Insomnia:

- Acesse o site oficial do Insomnia.
- Baixe a versão apropriada para o seu sistema operacional.
- Após o download, instale o Insomnia seguindo as instruções do instalador.

Executando a API:

- Importe o projeto no Eclipse.
- Configure o servidor Tomcat.
- Inicie o servidor.
- Teste os endpoints com o Insomnia.

Importação do Projeto no Eclipse:

- Abra o Eclipse.
- Clique em "File" > "Import".

- Na janela "Import", selecione "Existing Projects into Workspace".
- Clique em "Next".
- Na janela "Select root directory", navegue até a pasta do projeto e clique em "Open".
- Clique em "Finish".

Configuração do Servidor Tomcat:

- Clique com o botão direito no nome do projeto e selecione "Run As" > "Run on Server".
- Na janela "Choose a Run Target", selecione "Apache Tomcat v10.1 Server".
- Clique em "Next".
- Na janela "Select a Server Runtime", selecione "Apache Tomcat v10.1".
- Clique em "Next".
- Na janela "Configure Tomcat Runtime", clique em "Browse" e localize a pasta que descompactou o Tomcat.
- Clique em "Finish".

Inicialização do Servidor:

- O Eclipse iniciará o servidor Tomcat.
- Uma janela será aberta no navegador exibindo a página inicial da aplicação.

Teste dos Endpoints com o Insomnia:

- Abra o Insomnia.
- Crie uma nova coleção com um nome significativo.
- Em seguida, crie uma nova solicitação HTTP para cada endpoint mencionado anteriormente.
- Configure as solicitações conforme as regras fornecidas acima.

Protótipo:



Contrate sua bike com IA

Um novo meio de contratar seguros bike, utilizando inteligência artificial para uma nova experiência!

Ver Seguros



Cursos Gamificados

Venha ver nosso novo curso gamificado criado especialmente para você! Aprenda mais sobre seguros bike conosco de forma gratuita e eficiente.

Acessar





Motorola Baguio - Rockhopper Expert - S456789123
2023-11-13 - R\$ 5999.99
C:\fakepath\brad-ico.png

Atualizar

Cancelar



Motorola Baguio - Rockhopper Expert - S456789123
2023-11-13 - R\$ 5999.99

Atualizar

Cancelar



Motorola Baguio - Rockhopper Expert - S456789123
2023-11-13 - R\$ 5999.99

Atualizar

Cancelar



Motorola Baguio - Rockhopper Expert - S456789123
2023-11-13 - R\$ 5999.99


Atualizar

Cancelar






Cadastro de Seguro

Marca: Digite a marca de sua bike.
Modelo: Digite o modelo de sua bike.
Número de Série: Digite o número de série de :
Data da Compra: 13/11/2023 
Preço: R\$0
Nota Fiscal: Nenhum arquivo escolhido
Foto da Bike: Informe a URL de uma foto d
ADICIONAR CONTRATO



Atualizar Contrato

Marca: Motorola Baguio
Modelo: Rockhopper Expert
Número de Série: S456789123
Data da Compra: 13/11/2023 
Preço: R\$5999,99
Nota Fiscal: Nenhum arquivo escolhido
Foto da Bike: Informe a URL de uma foto d
ATUALIZAR CONTRATO

**EQUIPE OM CORP.**

RM551575

André Sant'ana

RM99632

Gabriel Eringer de Oliveira

RM99697

Matheus Augusto Leite

RM98251

Marcelo Hespanhol Dias

RM551428

Eduardo Tatsuo

RM551616

Yago Marques**WIP**

Disponível somente na versão final do projeto.