
Documentação de Projeto

para o sistema

Desenvolvimento automatizado de aplicações *Back-end for Front-end* (BFF Squared)

Versão 1.3

Projeto de sistema elaborado pelo aluno Marlon Henrique da Silva e apresentado ao curso de **Engenharia de Software** da **PUC Minas** como parte do Trabalho de Conclusão de Curso (TCC) sob orientação de conteúdo do professor Marco Rodrigo Costa, orientação acadêmica do professor José Laerte Xavier e orientação de TCC II do professor (a ser definido no próximo semestre).

08/03/2022

Tabela de Conteúdo

1. Introdução	1
2. Modelos de Usuário e Requisitos	1
2.1 Descrição de Atores	1
2.2 Modelos de Usuários	1
2.3 Modelo de Casos de Uso e Histórias de Usuários	4
2.4 Diagrama de Sequência do Sistema e Contrato de Operações	6
3. Modelos de Projeto	6
3.1 Diagrama de Classes	6
3.2 Diagramas de Sequência	6
3.3 Diagramas de Comunicação	6
3.4 Arquitetura	6
3.5 Diagramas de Estados	6
3.6 Diagrama de Componentes e Implantação.	6
4. Projeto de Interface com Usuário	7
5. Glossário e Modelos de Dados	11
6. Casos de Teste	11
7. Cronograma e Processo de Implementação	11

Histórico de Revisões

Nome	Data	Razões para Mudança	Versão
Criação	08/03/2022	Criação do Documento	1.0
Modificações	17/03/2022	Adição do diagrama de caso de uso e histórias de usuários	1.1
Modificações	22/03/2022	Adição dos modelos de usuário e um projeto de interface	1.2
Modificações	27/03/2022	Adição dos projetos de interface completos	1.3

1. Introdução

Este documento agrega: 1) a elaboração e revisão de modelos de domínio e 2) modelos de projeto para o sistema BFF Squared. A referência principal para a descrição geral do problema, domínio e requisitos do sistema é o documento de especificação que descreve a visão de domínio do sistema. Tal especificação acompanha este documento. Anexo a este documento também se encontra o Glossário.

2. Modelos de Usuário e Requisitos

Nesta seção são apresentados os modelos de usuários e requisitos para o sistema. Tal apresentação é feita pela descrição dos atores (Seção 2.1), descrição dos modelos de usuários (Seção 2.2), descrição de casos de uso de histórias dos usuários (Seção 2.3) e descrição do diagrama de sequências e contrato de operações para o sistema (seção 2.4).

2.1 Descrição de Atores

Os atores identificados para a utilização do sistema são pessoas desenvolvedoras de software, possivelmente com qualificações em *back-end*, *front-end* e *mobile*. As qualificações das pessoas desenvolvedoras podem ser diversas, pois as ações não demandam conhecimento específico e aprofundado em uma determinada tecnologia, mas é necessário que entendam da ferramenta e das suas próprias necessidades. O artefato produzido pelas interações dos usuários com o sistema é utilizado como insumo para que o sistema habilite suas funcionalidades e gere uma aplicação BFF.

2.2 Modelos de Usuários

Para este projeto, seus usuários foram modelados por meio de personas, de acordo com a definição de Courage e Baxter (2005), abordando os seguintes tópicos: identidade, status, objetivos, habilidades e tarefas. As personas exemplificadas nas Tabelas 1, 2 e 3 foram criadas com base em conversas com pessoas desenvolvedoras que demonstraram interesse na proposta do sistema. As características delas foram mescladas com personas fictícias a fim de exemplificar a abrangência do público alvo e de possíveis usuários deste sistema.




 <p>Fonte: freepik. Imagem livre de direitos autorais</p>	<p>Camila Alves, 30 anos, desenvolvedora <i>back-end</i>. Ela trabalha desenvolvendo e dando manutenção em microsserviços de uma empresa de serviços digitais presente em diversas plataformas.</p>
Status	Primária
Objetivos	<p>Aprimorar seus conhecimentos sobre sistemas distribuídos, escalabilidade e arquitetura de software para contribuir mais em seu trabalho e assumir uma posição de liderança técnica.</p>
Habilidades	<p>Construção de APIs (<i>Application Programming Interface</i>), REST (<i>Representational State Transfer</i>), WebSockets, Kafka, GraphQL, bancos de dados, monitoramento de sistemas.</p>
Tarefas	<p>Desenvolver novas funcionalidades para as aplicações já existentes em seu trabalho; revisão e refatoração de códigos; programação em pares; auxiliar novos membros da equipe; sugerir soluções e melhorias para as aplicações do seu time.</p>

Tabela 1. Primeira persona identificada.

 <p>Fonte: freepik. Imagem livre de direitos autorais</p>	<p>Murilo Santos, 28 anos, desenvolvedor <i>front-end</i>. Ele trabalha com desenvolvimento <i>front-end</i> há 10 anos. Se formou em um curso técnico de TI enquanto cursava o ensino médio e, em seguida, ingressou em um curso de graduação em Sistemas de Informação.</p>
Status	Primário
Objetivos	<p>Encontrar e/ou criar ferramentas para facilitar o desenvolvimento <i>front-end</i>; descobrir novas</p>

	maneiras de resolver problemas comuns no desenvolvimento Web; aprender mais sobre sistemas distribuídos e seus impactos em aplicações Web complexas.
Habilidades	Construção de aplicações <i>front-end</i> ; HTML (<i>HyperText Markup Language</i>); CSS (<i>Cascading Style Sheet</i>); JavaScript; integração com APIs REST e GraphQL; otimização de aplicações Web.
Tarefas	Liderança técnica de <i>front-end</i> Web; revisão e refatoração de código; mentoria de novos colaboradores; pesquisa de novas tecnologias e ferramentas para serem integradas ao processo de desenvolvimento das equipes.

Tabela 2. Segunda persona identificada.

 <p>Fonte: freepik. Imagem livre de direitos autorais</p>	Juliana Ferreira, 23 anos, desenvolvedora <i>mobile</i> . Recém-formada em um curso técnico de programação e desenvolvimento de aplicações móveis. Atua no mercado de trabalho há 3 anos fazendo estágios remunerados, mas foi contratada há pouco mais de 8 meses pela empresa atual.
Status	Secundária
Objetivos	Expandir e aprofundar seus conhecimentos sobre desenvolvimento <i>mobile</i> , Interface de Usuário/Experiência de Usuário (UI/UX, do inglês User Interface/User Experience), consumo de APIs (<i>Application Programming Interface</i>) com REST (<i>Representational State Transfer</i>) e GraphQL.
Habilidades	Desenvolvimento de aplicativos para multiplataformas com React Native e Flutter; desenvolvimento Web com React JS; design e prototipação de interfaces com Figma.

Tarefas	Implementar novas funcionalidades e aprimorar a usabilidade do aplicativo da empresa em que trabalha, disponível para Android e iOS. Uma nova versão deve ser lançada semanalmente com correções e/ou novidades.
---------	--

Tabela 3. Terceira persona identificada.

2.3 Modelo de Casos de Uso e Histórias de Usuários

Nesta seção estão representadas as histórias de usuário definidas para o sistema e na Figura 1 estão ilustrados os casos de uso que representam as ações dos usuários durante a sua utilização. Ambos são a materialização do escopo da aplicação. Para fins de organização, as histórias se encontram numeradas de 1 a 10 e rotuladas com a letra H (de história), não significando uma ordem de prioridade e/ou importância.

1. H01: Gabriel, como desenvolvedor *front-end*, deseja simplificar o fluxo de dados nas aplicações Web;
2. H02: Camila, como desenvolvedora *back-end*, deseja abstrair o versionamentos das *APIs* para os clientes;
3. H03: Pedro, como desenvolvedor *front-end*, deseja abstrair as fontes de dados da aplicação;
4. H04: Murilo, como desenvolvedor *full stack*, deseja uma maneira simples de agregar os dados de diversas fontes para usos específicos do sistema;
5. H05: Paulo, como desenvolvedor *mobile*, deseja uma maneira simples de consumir dados por demanda para o aplicativo;
6. H06: Juliana, como desenvolvedora *back-end*, deseja uma maneira simples de criar uma *API GraphQL*;
7. H07: Henrique, como arquiteto de software, deseja uma maneira simples de criar aplicações BFFs para gerenciar o fluxo de dados entre microsserviços e aplicações clientes;
8. H08: Daniel, como desenvolvedor, deseja automatizar e padronizar a criação de aplicações para o mesmo propósito, mas com diferentes contextos;
9. H09: Giovana, como desenvolvedora, deseja definir novos recursos em uma *API* apenas especificando os dados;
10. H10: Walter, como desenvolvedor *back-end*, deseja gerenciar e testar *APIs* de forma visual, utilizando interfaces dedicadas.

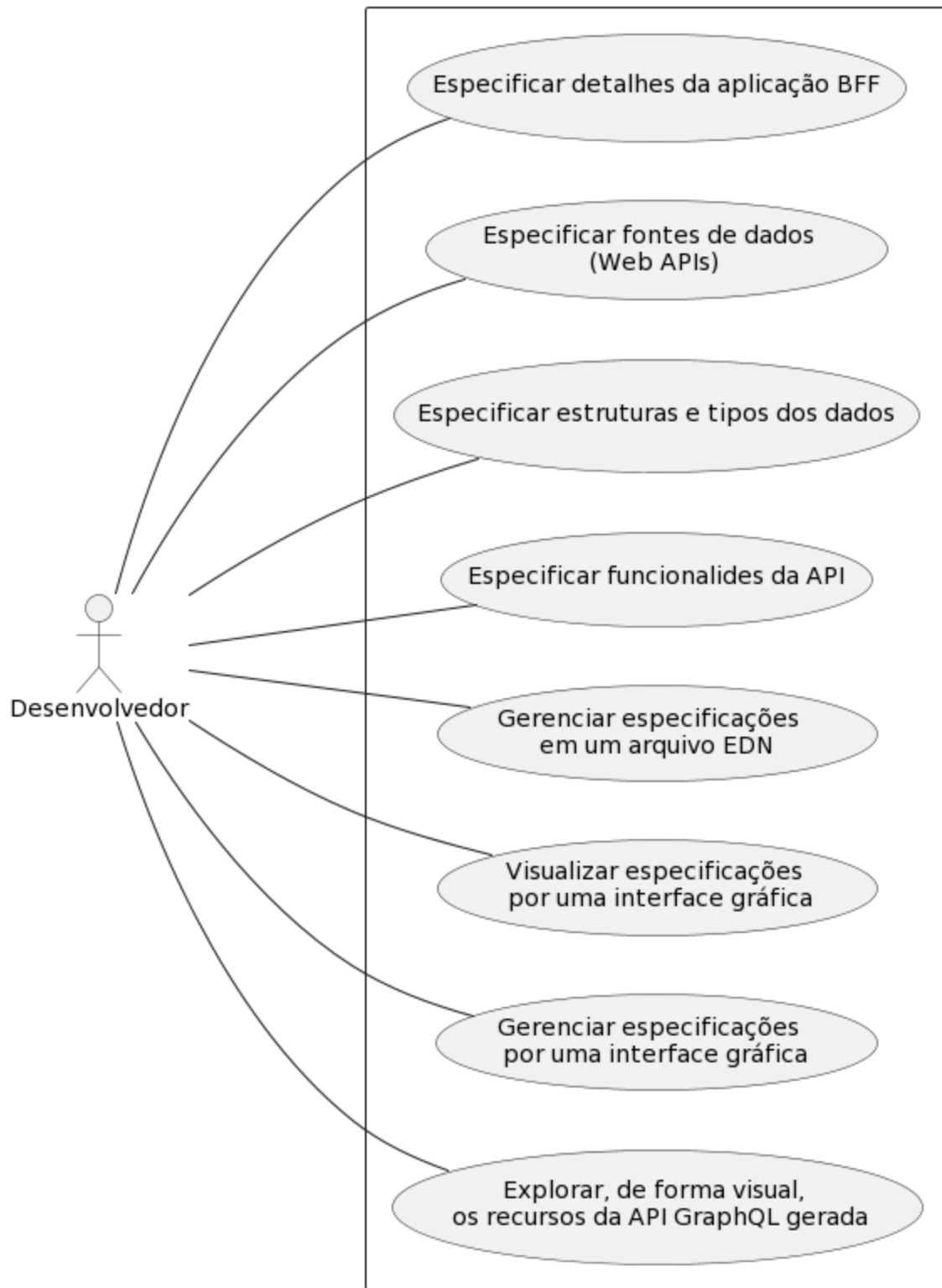


Figura 1. Diagrama de casos de uso do sistema.

2.4 Diagrama de Sequência do Sistema e Contrato de Operações

Nesta subseção é apresentado o diagrama de sequência do sistema e os Contratos de Operações.

Formato para cada contrato de operação

Contrato	
Operação	
Referências cruzadas	
Pré-condições	
Pós-condições	

3. Modelos de Projeto

3.1 Diagrama de Classes

Diagrama de classes do sistema

3.2 Diagramas de Sequência

Diagramas de sequência para realização de casos de uso.

3.3 Diagramas de Comunicação

Diagramas de comunicação para realização de casos de uso.

3.4 Arquitetura

Pode ser descrita com um diagrama apropriado da UML ou C4 Model

3.5 Diagramas de Estados

Diagramas de estados do sistema.

3.6 Diagrama de Componentes e Implantação.

Diagramas de componentes do sistema. Diagrama de implantação mostrando onde os componentes estarão alocados para a execução.

4. Projeto de Interface com Usuário

A seguir estão representados alguns *wireframes* (protótipos) das interfaces gráficas do sistema que serão construídas para a interação visual dos usuários com o sistema proposto.

Na Figura 2 está representada a tela principal do sistema, que será a tela de gerenciamento de *schemas*, na qual serão listados, criados, editados e excluídos todos os modelos de dados que serão tratados pelo sistema. Como todos os outros elementos do sistema a serem definidos pelo usuário, os *schemas* são apenas especificações de dados, com seus atributos e seus respectivos valores.

Header

Schemas APIs Queries Mutations Settings

Enums Interfaces Types Inputs

Type 1 >

Type 2 >

Type 3 >

Type 4 >

Type 5 >

Type 6 >

...

Create new Delete

Name: Input Kind v

Description: Input

Implements: 1 Values: Input

Fields: Add

Name: Input

Description: Input

Type: Select v List Required

Default value: Input

Deprecated? ☐ Input Delete

Name: Input

Description: Input

Type: Select v List Required

Default value: Input

Save

Figura 2. Wireframe da aba de *schemas* do sistema.

Na Figura 3 está representada a tela de gerenciamento de *APIs*, na qual serão listadas, criadas, editadas e excluídas todas as fontes de dados externas que serão tratados pelo sistema. Neste caso, cada fonte de dados é um *endpoint* de uma outra API que será consumida para obter e/ou modificar dados, a depender do funcionamento de cada recurso. Nessa tela são definidas apenas as informações necessárias para realizar uma determinada requisição, a especificação dos dados recebidos e enviados, assim como os parâmetros suportados, é feita diretamente nas abas de *Queries* e *Mutations*.

Header													
<div>Schemas APIs Queries Mutations Settings</div>													
<div>Endpoint 1 ></div> <div>Endpoint 2 ></div> <div>Endpoint 3 ></div> <div>Endpoint 4 ></div> <div>Endpoint 5 ></div> <div>Endpoint 6 ></div> <div>...</div>	<div>Create new Delete</div> <table border="1"><tr><td>Name:</td><td>Input</td><td>Method ></td></tr><tr><td>Description:</td><td colspan="2">Input</td></tr><tr><td>URI:</td><td colspan="2">Input</td></tr><tr><td>Into response:</td><td colspan="2">Input</td></tr></table> <div>Save</div>	Name:	Input	Method >	Description:	Input		URI:	Input		Into response:	Input	
Name:	Input	Method >											
Description:	Input												
URI:	Input												
Into response:	Input												

Figura 3. Wireframe da aba de *APIs* do sistema.

Na Figura 4 está representada a tela de gerenciamento de *Queries*, na qual serão listadas, criadas, editadas e excluídas todas as consultas de dados externas que serão tratados pelo sistema. Neste caso, cada consulta é um recurso da aplicação que será fornecido para obter dados de uma determinada *API*. Nessa tela são definidas apenas as informações necessárias para realizar uma determinada consulta, utilizando modelos de dados e uma *API* previamente definidos. Além disso, também é feita a especificação dos dados recebidos e enviados, assim como os parâmetros suportados.

Header																																									
<div>Schemas APIs Queries Mutations Settings</div>																																									
<div>Query 1 ></div> <div>Query 2 ></div> <div>Query 3 ></div> <div>Query 4 ></div> <div>Query 5 ></div> <div>Query 6 ></div> <div>...</div>	<div>Create new Delete</div> <table><tr><td>Name:</td><td colspan="3">Input</td></tr><tr><td>Type:</td><td>Select</td><td><input type="radio"/> List</td><td><input type="radio"/> Required</td></tr><tr><td>Source:</td><td colspan="3">Select ></td></tr><tr><td>Args:</td><td colspan="3"><div>Add</div></td></tr><tr><td>Name:</td><td colspan="3">Select ></td></tr><tr><td>Type:</td><td>Select</td><td><input type="radio"/> ></td><td>Delete</td></tr><tr><td>Name:</td><td colspan="3">Select ></td></tr><tr><td>Type:</td><td>Select</td><td><input type="radio"/> ></td><td>Delete</td></tr><tr><td>Name:</td><td colspan="3">Select ></td></tr><tr><td>Type:</td><td>Select</td><td><input type="radio"/> ></td><td>Delete</td></tr></table> <div>Save</div>	Name:	Input			Type:	Select	<input type="radio"/> List	<input type="radio"/> Required	Source:	Select >			Args:	<div>Add</div>			Name:	Select >			Type:	Select	<input type="radio"/> >	Delete	Name:	Select >			Type:	Select	<input type="radio"/> >	Delete	Name:	Select >			Type:	Select	<input type="radio"/> >	Delete
Name:	Input																																								
Type:	Select	<input type="radio"/> List	<input type="radio"/> Required																																						
Source:	Select >																																								
Args:	<div>Add</div>																																								
Name:	Select >																																								
Type:	Select	<input type="radio"/> >	Delete																																						
Name:	Select >																																								
Type:	Select	<input type="radio"/> >	Delete																																						
Name:	Select >																																								
Type:	Select	<input type="radio"/> >	Delete																																						

Figura 4. Wireframe da aba de *Queries* do sistema.

Na Figura 5 está representada a tela de gerenciamento de *Mutations* (que é igual à tela de gerenciamento de *Queries*), na qual serão listadas, criadas, editadas e excluídas todas as modificações de dados externos que serão tratados pelo sistema. Neste caso, cada modificação é um recurso da aplicação que será fornecido para enviar dados para uma determinada *API*. Nessa tela são definidas apenas as informações necessárias para realizar uma determinada modificação, utilizando modelos de dados e uma *API* previamente definidos. Além disso, também é feita a especificação dos dados enviados e recebidos, assim como os parâmetros suportados.

Header

Schemas

APIs

Queries

Mutations

Settings

Mutation 1

Mutation 2

Mutation 3

Mutation 4

Mutation 5

Mutation 6

...

Create new

Delete

Name:

Input

Type:

Select

V

List

Required

Source:

Select

V

Args:

Add

Name:

Select

V

Type:

Select

V

Delete

Name:

Select

V

Type:

Select

V

Delete

Name:

Select

V

Type:

Select

V

Delete

Save

Figura 5. Wireframe da aba de *Mutations* do sistema.

Na Figura 6 está representada a tela de *Settings*, na qual serão definidas todas as configurações de servidor necessárias para executar a aplicação nos ambientes desejados, expondo uma *API GraphQL* com os recursos definidos nas outras telas.

Header

Schemas

APIs

Queries

Mutations

Settings

Name:

Input

Env:

Select

V

Description:

Input

Host:

Input

API Path:

Input

Port:

Input

IDE Path:

Input

CORS:

Input

Save

Figura 6. Wireframe da aba de *Settings* do sistema.

5. Glossário e Modelos de Dados

Deve-se apresentar o glossário para o sistema. Também apresente esquemas de banco de dados e as estratégias de mapeamento entre as representações de objetos e não-objetos.

6. Casos de Teste

Uma descrição de casos de teste para validação do sistema.

7. Cronograma e Processo de Implementação

Uma descrição do cronograma para implementação do sistema e do processo que será seguido durante a implementação.