



Documentação de solução de Arquitetura

Volvo - MDC

Versão 1.1.0





12 de Março de 2021

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	2 de 17
Arquitetura – Volvo MDC			



INFORMAÇÕES SOBRE O DOCUMENTO

Este documento será mantido regularmente pela área de Tecnologia, podendo sofrer alterações conforme necessidade .

DATA	VERSÃO	STATUS	AUTOR	DESCRIÇÃO DAS ALTERAÇÕES
11/03/2021	1.0.0	Elaboração	Marlon Silva	Elaboração do documento.
12/03/2021	1.0.0	Edição	Marlon Silva	Edição do documento.
26/03/2021	1.1.0	Edição	Marlon Silva	Edição do documento.

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	3 de 17
Arquitetura – Volvo MDC			

DESCREVER O PROCESSO DE ARQUITETURA DE PROJETOS VIGENTE NO VOLVO PROJETO MDC.	4
PUBLICO ALVO	4
RESPONSABILIDADES.	4
INFRAESTRUTURA VOLVO – MDC.	4
ESTEIRAS DE DEVOPS – CI E CD	5
PRINCIPAIS TECNOLOGIAS.	5
ARQUITETURA MICROSERVIÇOS	6
ARQUITETURA BACKEND:	7
1. TECNOLOGIA UTILIZADA:	7
2. ESTRUTURAS E RESPONSABILIDADES DE PACOTES	7
3. SWAGGER.	8
PADRÃO GITFLOW	11
FRONT-END.	11
1. TECNOLOGIA UTILIZADA:	11
2. ESTRUTURAS E RESPONSABILIDADES DE PACOTES	12
BANCO DE DADOS.	14
1. PADRÕES E NORMAS TÉCNICAS:	14

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	4 de 17
Arquitetura – Volvo MDC			

OBJETIVO

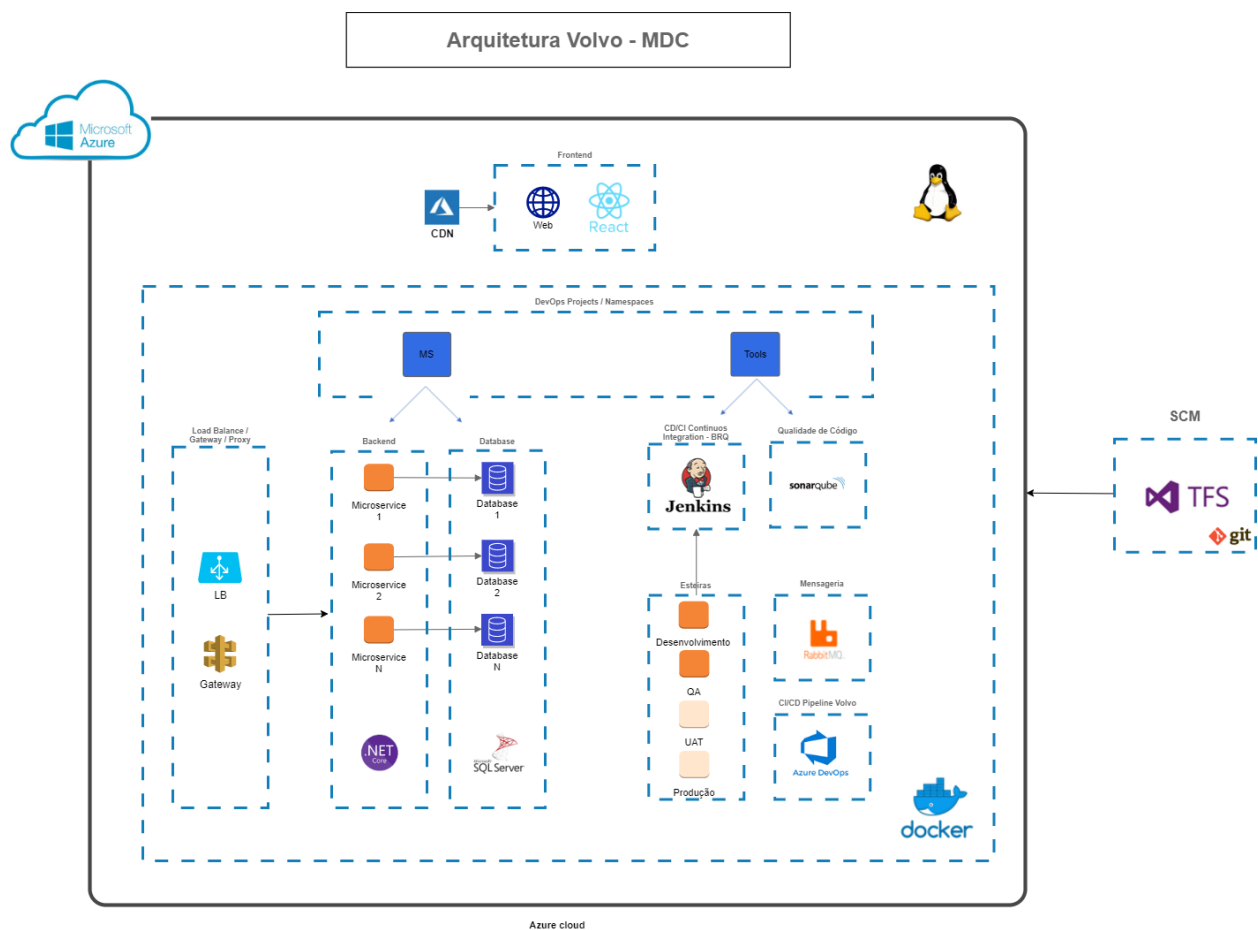
Descrever o processo de arquitetura de projetos vigente no Volvo projeto MDC.



PUBLICO ALVO

Envolvidos direto ou indiretamente com o processo de desenvolvimento.

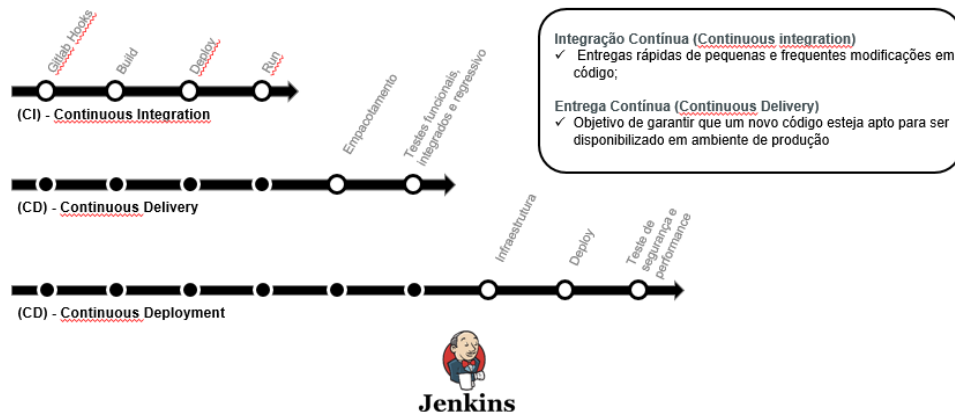
RESPONSABILIDADES

Infraestrutura Volvo – MDC



 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	5 de 17
Arquitetura – Volvo MDC			

Esteiras de DevOps – CI e CD



Principais Tecnologias

Infra:



Frontend





Backend



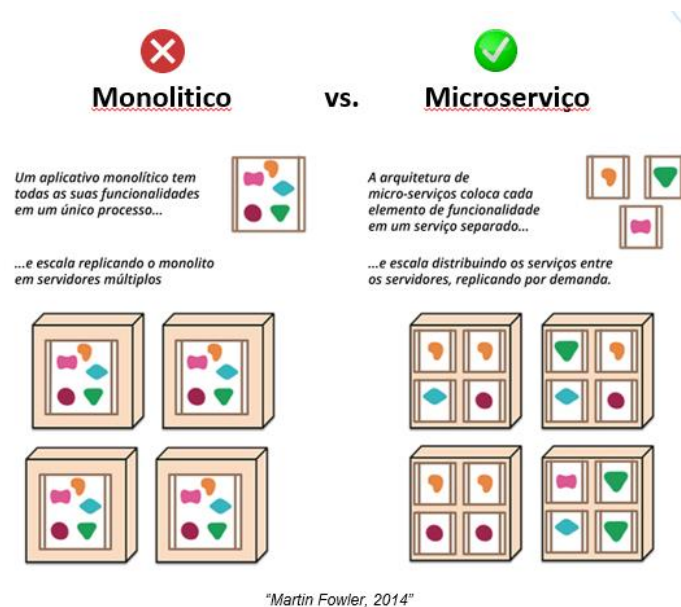
Banco de dados



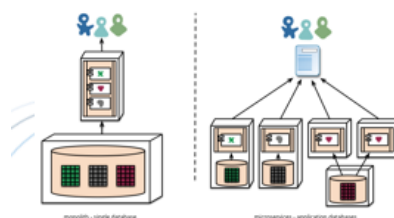
 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	6 de 17
Arquitetura – Volvo MDC			

Arquitetura Microserviços



A arquitetura de microserviços é uma abordagem que desenvolve um aplicativo único como uma suíte de pequenos serviços, cada um executando seu próprio processo e se comunicando através de mecanismos leves, por exemplo uma API com recursos HTTP. Esses serviços são construídos em torno de capacidades de negócios e funcionam através de mecanismos de deploy independentes totalmente automatizados. Há o mínimo possível de gerenciamento centralizado desses serviços, que podem ser escritos em diferentes linguagens de programação e utilizam diferentes tecnologias de armazenamento de dados. Modelagem de microserviço difere da modelagem Monolítica que é construído por uma única unidade centralizada.



Vantagens da arquitetura com microserviços



- Arquitetura individual simples
- Mecanismo de comunicação universal e leve
- Sistemas totalmente independentes
- Serviços coesos e desacoplados
- Facilidade de deploy e testes unitários
- Times multidisciplinares completos
- Ausência de um ponto de falha único
- Entrega contínua

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	7 de 17
Arquitetura – Volvo MDC			

Arquitetura Backend:

O backend encapsula a camada de negócio e a camada de banco expondo apenas uma interface no formato RESTful para o frontend. O frontend deve **orquestrar** as chamadas para realizar uma operação completa (ex. salvar uma tela).

A padrão REST é diferente da RPC, portanto não são aceitos acoplamentos entre serviço e consumidores.

No conceito RESTful, os endpoints do backend são concebidos para interação com um item ou lista (collection), ou seja, eu posso manipular um item ou manipular uma lista ou um item sobre uma lista.

Com esse conceito não podemos criar/interpretar que os endpoints estão manipulando/interagindo com o modelo de persistência (banco de dados), apesar de serem semelhantes.

Em resumo o endpoint RESTful deve abstrair a camada de persistência para o frontend.

1. Tecnologia utilizada:

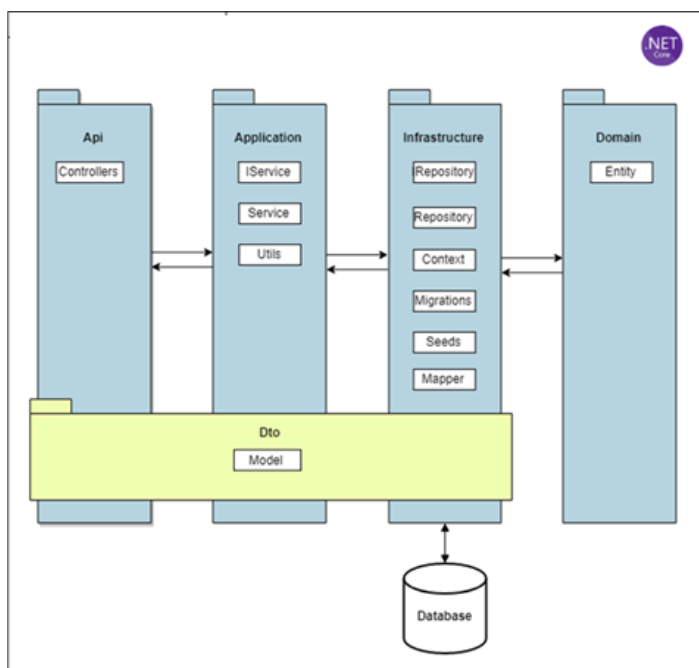
.Net Core 3.1: Responsável pelo back-end da aplicação.



EntityFrameworkCore 3.1: ORM (Mapeador de Objeto Relacional).

SQL Server: Sistema gerenciador de banco de dados objeto relacional (SGBD).

AutoMapper 6.1.0: mapeamento de propriedades de um objeto para outro.

2. Estruturas e responsabilidades de Pacotes



 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	8 de 17
Arquitetura – Volvo MDC			

Descrição, objetivo e responsabilidades de cada camada:

API:

Biblioteca de Classes, responsáveis pelas rotinas de acesso às plataformas baseadas na Web. Esta biblioteca contém Swagger que se trata de uma ferramenta opensource que serve de suporte para entregar uma API bem documentada (gerada automática ou manualmente) que possa ser usada por terceiros.

DTO:

Biblioteca de Classes com os objetos responsável por trafegar os dados entre as camadas. Utilizamos DTOs para representar os dados que queremos que os clientes da Web API recebam.

Application:

Biblioteca de Classes, responsáveis pelos serviços de acesso à camada de Infraestrutura do projeto.

Infrastructure:

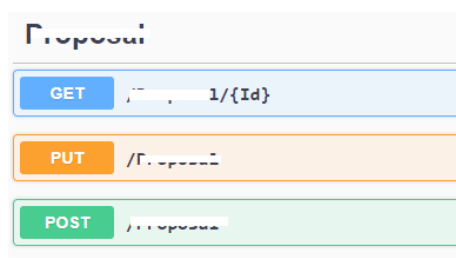
Camada responsável pelo acesso ao Banco de dados e controle de fluxo desses dados, contendo as classes de **Repository** e **Seeds**, além do Contexto de **DataBase** configurado para trabalhar com o Banco de dados **SQL Server**.



Domain:

Biblioteca de Classes, que contém as **Entidades** do projeto, com as representações de domínio de cada uma.

3. Swagger

O contrato das API's estará disponível pelo Swagger: <https://endereco/swagger/index.html>



 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	9 de 17
Arquitetura – Volvo MDC			



Padrões e construção de uma api RESTFul exemplo:

Método	URI	Utilização
GET	/clientes	Recuperar os dados de todos os clientes.
GET	/cliente/{id}	Recuperar os dados de um determinado cliente.
POST	/cliente	Criar um novo cliente.
PUT	/cliente/{id}	Atualizar os dados de um determinado cliente.
PATCH	/cliente/{id}	Atualizar um dado específico de um determinado cliente.
DELETE	/cliente/{id}	Excluir um determinado cliente.

As classes de Controller deverão obrigatoriamente conter apenas os métodos HTTP: GET, POST, PUT, DELETE e PATCH.

GET	Obter os dados de um recurso. O identificador (ID) do recurso é a PK.
POST	Criar um novo recurso.
PUT	Substituir/Atualizar todos os dados de um determinado recurso.
PATCH	Atualizar parcialmente um determinado recurso.
DELETE	Excluir um determinado recurso. O identificador (ID) do recurso é a PK.

<http://blog.caelum.com.br/rest-principios-e-boas-praticas/>



 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	10 de 17
Arquitetura – Volvo MDC			

Utilização correta dos códigos HTTP

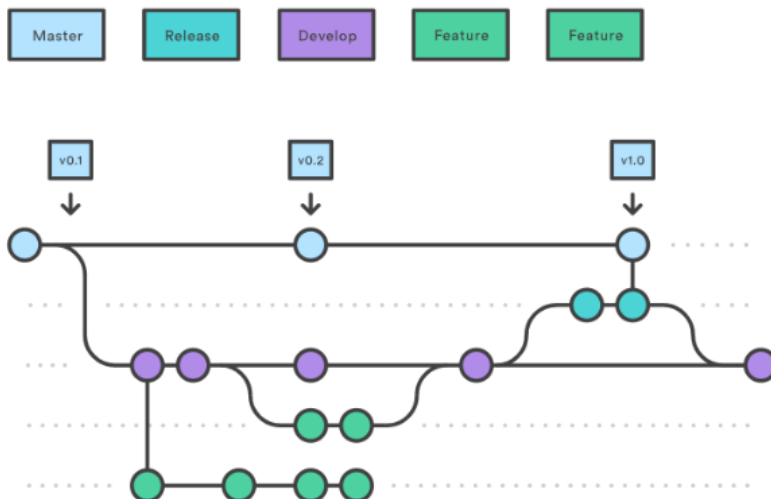
Classe	Semântica
2xx	Indica que a requisição foi processada com sucesso.
3xx	Indica ao cliente uma ação a ser tomada para que a requisição possa ser concluída.
4xx	Indica erro(s) na requisição causado(s) pelo cliente.
5xx	Indica que a requisição não foi concluída devido a erro(s) ocorrido(s) no servidor.

Retornos Válidos:

Status Code	Método HTTP	Descrição
200	GET	Quando o recurso é localizado com sucesso.
201	POST	Quando o recurso é criado com sucesso. *Deve ser retornado o HTTP Header “LOCATION”, contendo a URI para acesso ao recurso via GET.
204	PUT, PATCH e DELETE	Quando retorno de sucesso sem body no response.
412	GET, POST, PUT, PATCH e DELETE	Quando há algum erro na regra de negócio ou técnica.
500	GET, POST, PUT, PATCH e DELETE	Quando houver erro interno ou não tratado/verificado.

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	11 de 17
Arquitetura – Volvo MDC			

Padrão Gitflow



<https://docs.microsoft.com/pt-br/azure/architecture/framework/devops/gitflow-branch-workflow>

Front-end



1. Tecnologia utilizada:

React: Responsável pelos serviços e elementos da aplicação.

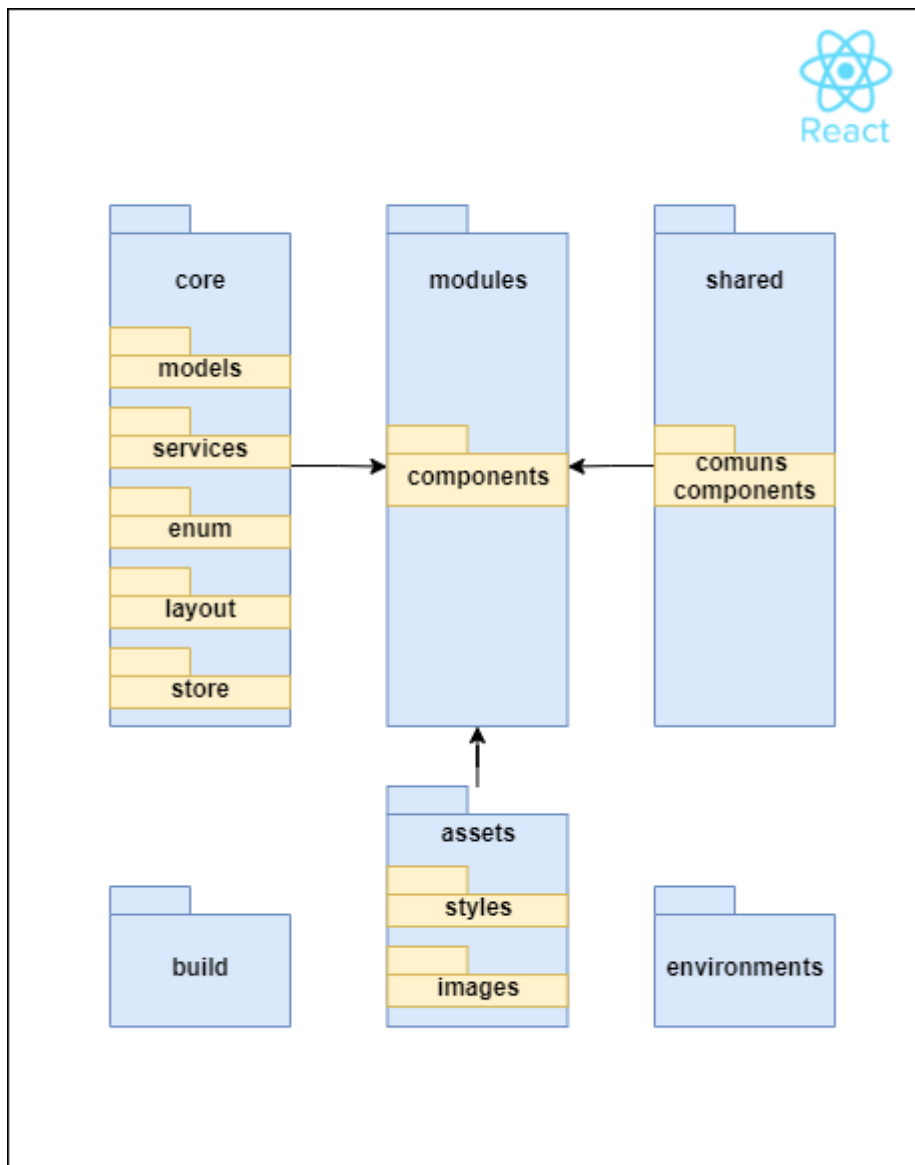
Material UI: Responsável pelo Layout da aplicação



Navegadores suportados: Google Chrome, Mozilla Firefox

Nome Projeto: volvo-mdc-fe-projeto

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	12 de 17
Arquitetura – Volvo MDC			

2. Estruturas e responsabilidades de Pacotes



 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	13 de 17
Arquitetura – Volvo MDC			

Build:

Pacote onde o build é gerado com os arquivos para o deploy.

Environments:

Pacote onde ficam disponíveis os arquivos com as configurações de cada ambiente e seus respectivos links de API's.

Core

Service: Pacote pelo responsável pela camada de serviços da aplicação, ou seja, é realizado a comunicação direta com os serviços do Backend, além de realizar o controle de mudança de dados de objetos ou atributos.

Models: Pacote responsável pelos arquivos de modelo da aplicação, ou seja, as classes que preenchem os campos das telas da camada de visualização. Além de possuir a responsabilidade de enviar ou receber dados do backend.

Enum: Pacote responsável por configurar valores fixos, ou seja, constantes que poderão ser utilizadas em diversos lugares na aplicação.

Layout: Pacote responsável pelos componentes como Header, Footer, Sidebar ou algum componente que irá ter em todas as telas ou quase todas.

Store: Ducks e Sagas, responsável pela autenticação e autorização da aplicação.

Modules:



Components: Pacote responsável por todos os componentes roteirizados da aplicação.

Ou seja, local onde será criado as telas (camada de visualização), eventos e regras de negócio de cada componente.

Além de possuir a possibilidade de realizar as chamadas para o pacote de serviços.

Shared:

Comuns Components Pacote responsável por componentes que serão compartilhados entre outros componentes, como por exemplo, um Modal, ou seja, componentes que não são roteados.

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	14 de 17
Arquitetura – Volvo MDC			

Banco de dados

1. Padrões e normas técnicas:

Banco de dados

O nome do banco de dados deverá identificar o negócio de Microserviço que está sendo criado e deverá refletir isso para seu nome. Para o nome da base utilizar tudo em minúsculo.

EX:

Projeto: volvo-mdc-ms-projeto

Banco: volvo-mdc-db-projeto

Tabelas

- O nome de uma tabela deverá ser sugestivo;
- Deve-se fazer o uso de nomenclatura orientado a objeto;
- Para o nome das tabelas utilizar tudo em minúsculo;
- Para as tabelas compostas deve-se utilizar o caractere "_";
- O nome da tabela deve estar sempre no singular;
- Evite usar abreviações;
- Não utilize acentuações ou caracteres especiais;

Exemplos de nomes para tabelas:

Usuário: Nome da tabela --> usuario (joão, pedro, marcio)

Permissão: Nome da tabela --> permissao (aluno, administrador, professor)



Permissões de usuários: Nome da tabela --> permissao_usuario

Nome dos atributos – Colunas – CAMPOS*

- Não usar preposições;
- Para o nome dos atributos utilizar tudo em minúsculo;
- Usar palavras no singular e sem acentuação;
- Usar nome que identifique e individualize o dado;
- Dar nomes distintos para dados distintos;

Views

- O nome de uma view deverá ser sugestivo;
- Para o nome das views utilizar tudo em minúsculo;

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	15 de 17
Arquitetura – Volvo MDC			

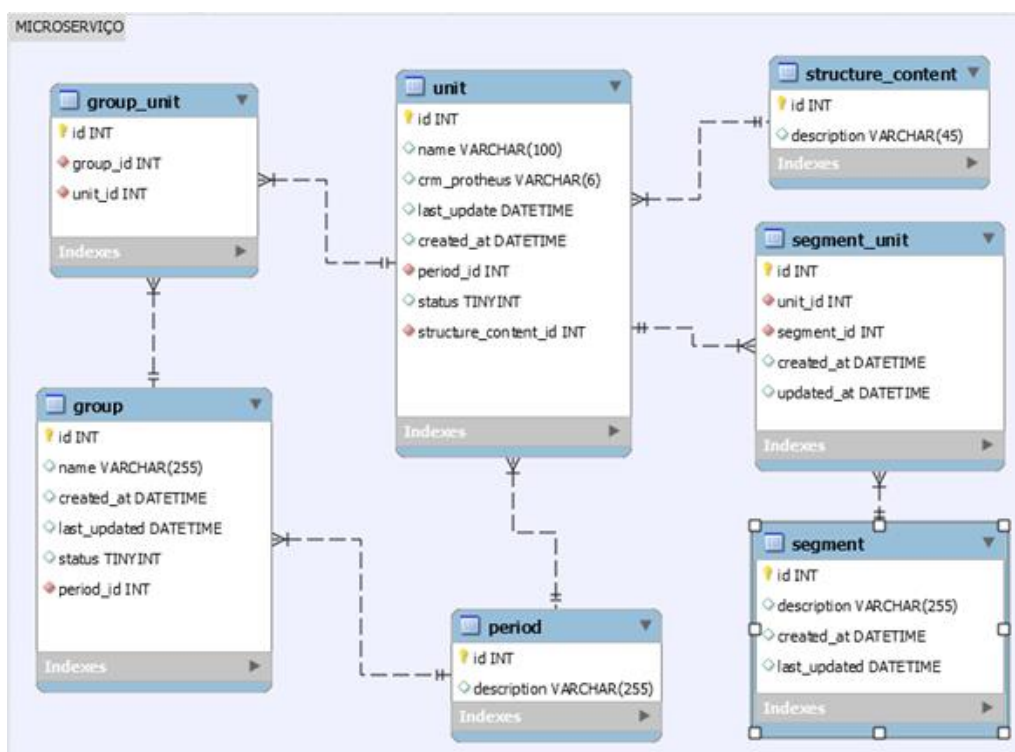
Para o nome de views compostos deve-se utilizar o caractere "_";



O nome da view deve estar sempre no singular iniciados pelo prefixo: "vw_nome_da_view";

Evite usar abreviações;

Não utilize acentuações ou caracteres especiais;



Exemplo:



 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	16 de 17
Arquitetura – Volvo MDC			

BASE NORMATIVA

Governança da tecnologia da informação para desenvolvimento de software.

 	Tecnologia da Informação	Título	Documento de arquitetura Volvo
		Versão	1.1.0
		Edição	12 de março de 2021
		Página	17 de 17
Arquitetura – Volvo MDC			

HISTÓRICO DE ATUALIZAÇÕES

Versão 1.0.0

Criação do Documento

Edição do Documento, alteração do Frontend de Angular para React