

# AutoPrime - Sistema de Concessionária



Instituto de  
computação

Integrantes: Victor Miranda Florido, Felipe Mendes Salles, Guilherme Moreira, Pedro Arthur Nascimento Resende, Gabriel Tavares Gargur

Disciplina Projeto de Software

# Escopo do Sistema

O sistema da concessionária AutoPrime é um software especializada na venda de carros que busca atender às necessidades tanto dos nossos clientes quanto da equipe de funcionários. Nosso objetivo principal é proporcionar uma experiência de compra de veículos simplificada e personalizada, onde oferecemos como funcionalidades principais: a compra de um veículo, agendar test drives, ver preços e descontos.

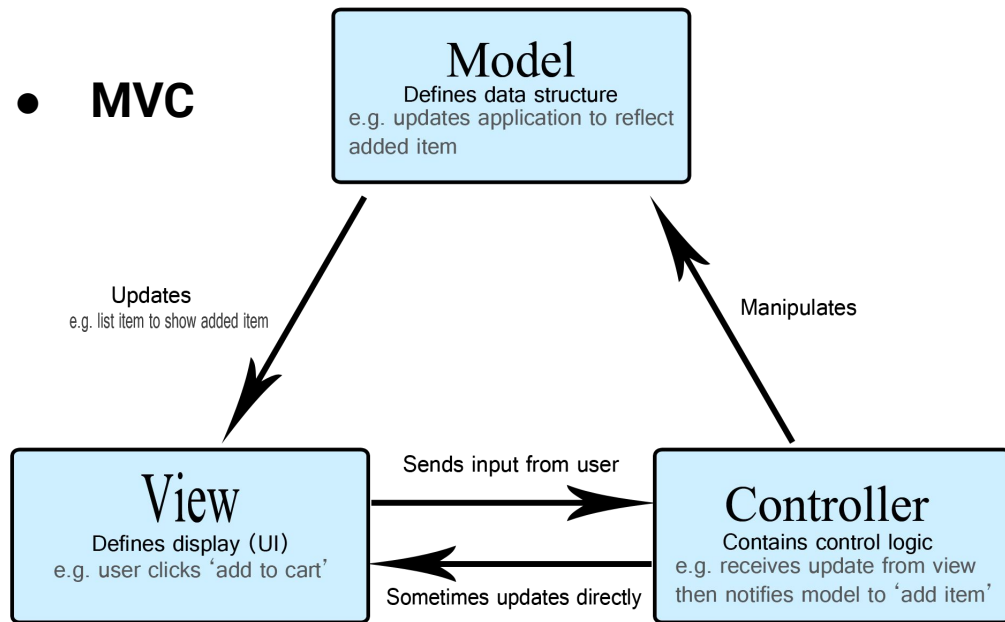


# Requisitos Arquiteturais

- O sistema deve ser de uso fácil e rápido para usuários, sem exigir treinamento prévio (**Usabilidade**).
- O sistema deve permitir o cadastro de clientes.
- O sistema deve fornecer a um único usuário selecionado funções privilegiadas de gerência (**Segurança**).
- O sistema deve permitir o cadastro de funcionários.
- O sistema deve manter registro dos pedidos de cada cliente.
- O sistema deve fornecer e calcular descontos para clientes baseado em suas compras passadas.
- O sistema deve permitir o agendamento de *test drives*.
- O sistema deve gerenciar os modelos de veículos disponíveis e seus respectivos estoques.
- O sistema deve permitir que os usuários vejam as informações e modifiquem as partes de um modelo automobilístico que sejam personalizáveis.
- O sistema deve ser capaz de mostrar as especificações de mais de um veículo ao mesmo tempo para o cliente poder compará-los.

# Padrões Arquiteturais e Justificativas

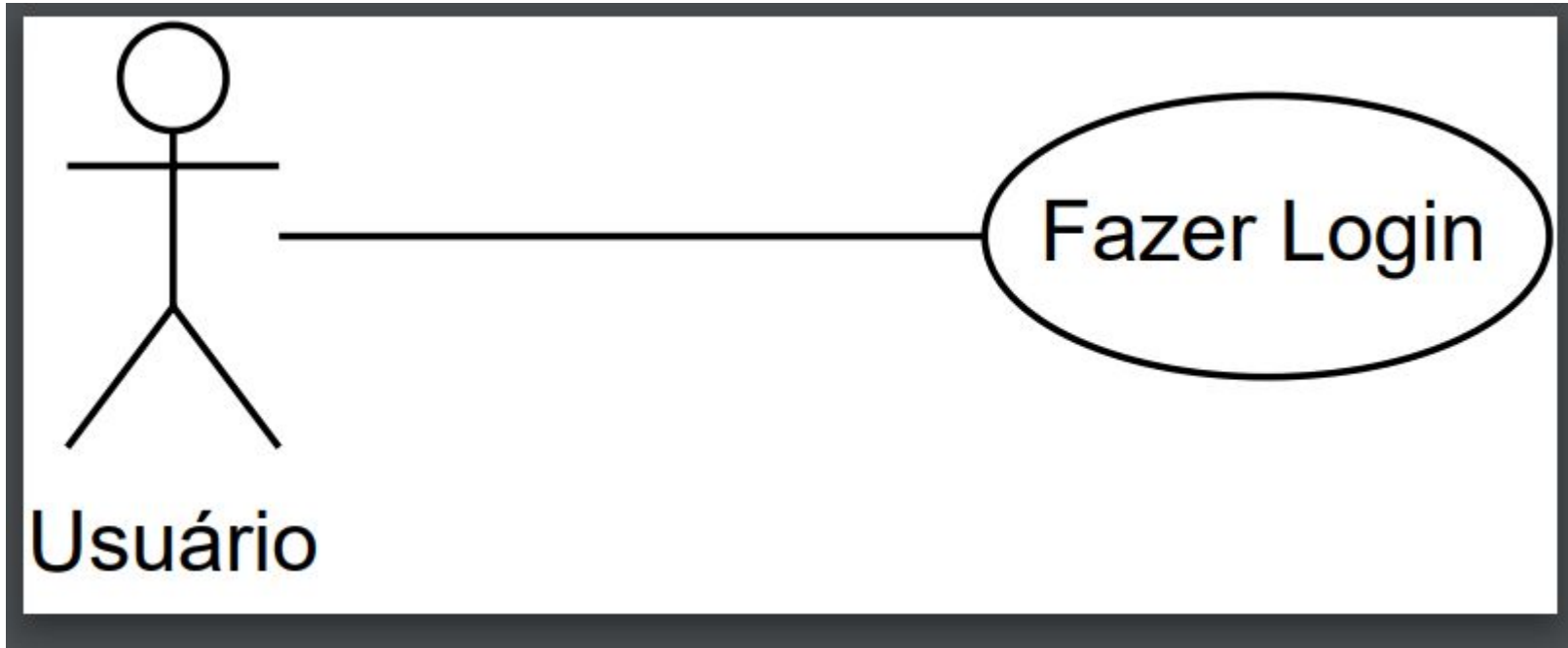
- **MVC**



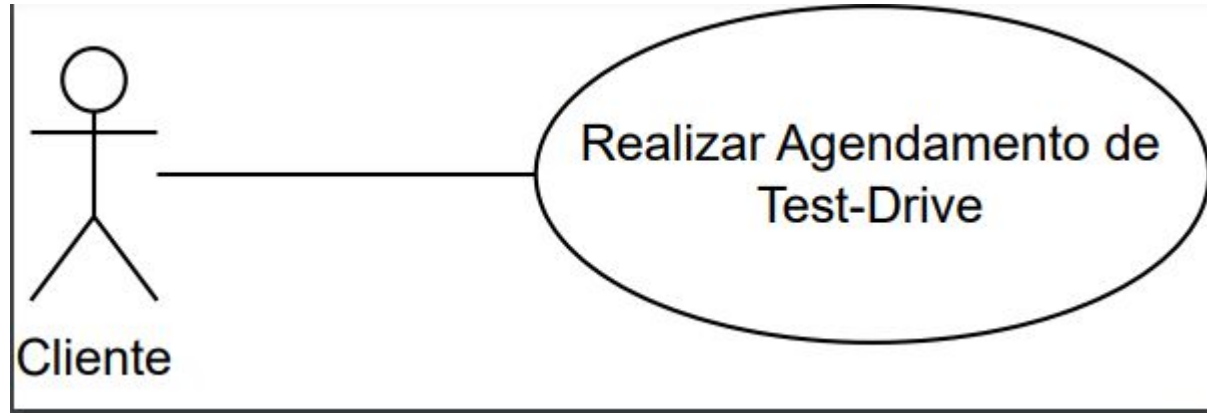
- **Justificativas:**

- Escalabilidade
- Facilidade de Manutenção

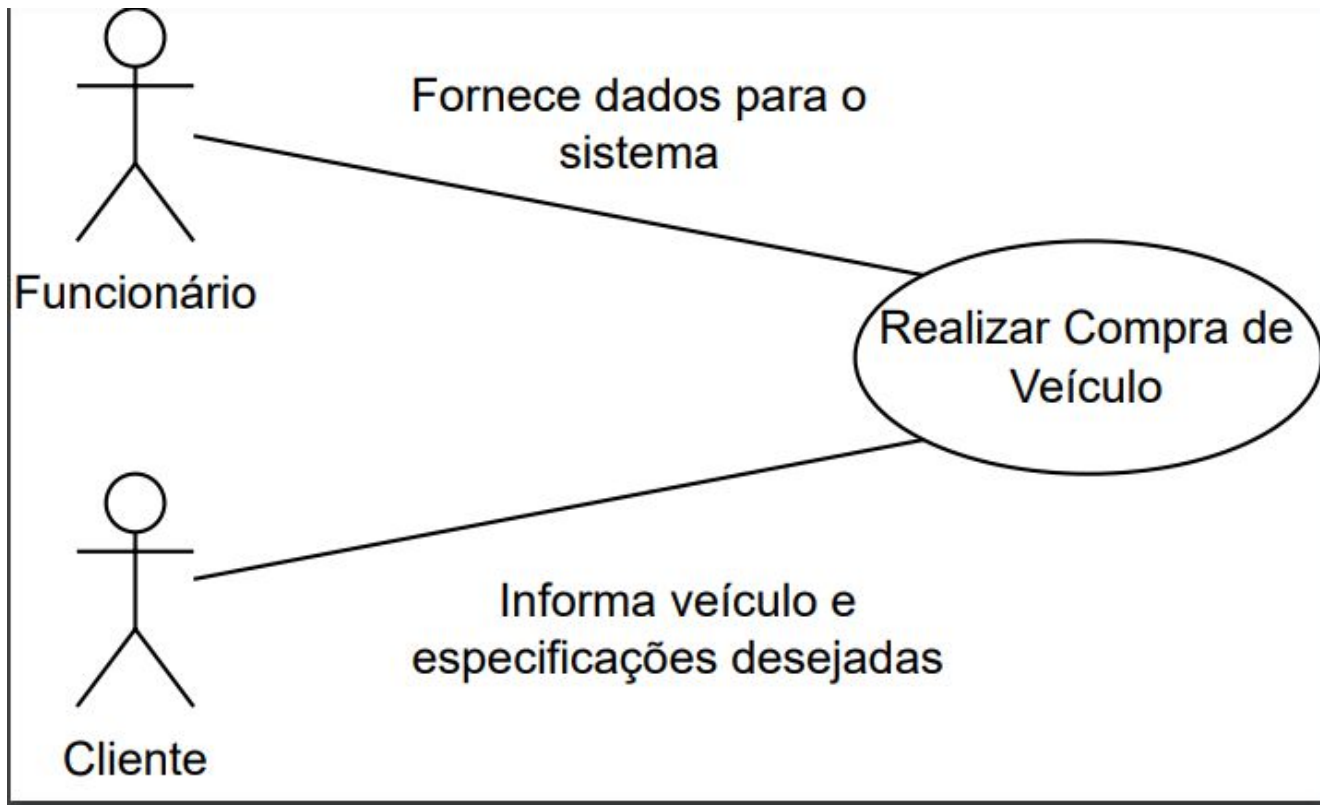
## Casos de Uso



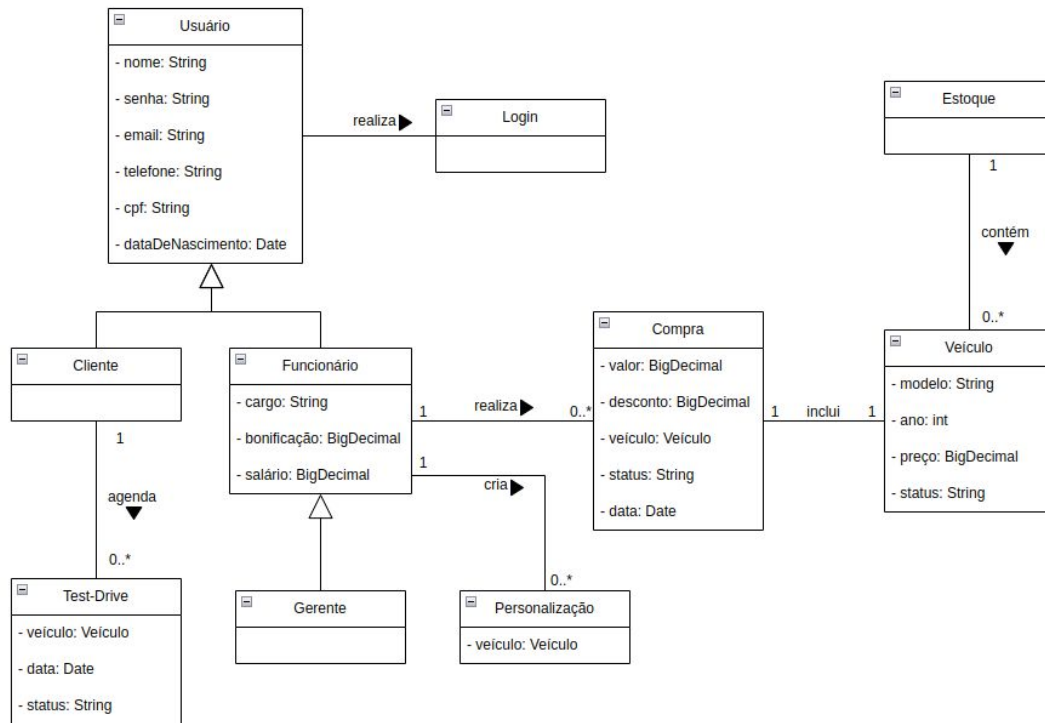
# Casos de Uso



# Casos de Uso

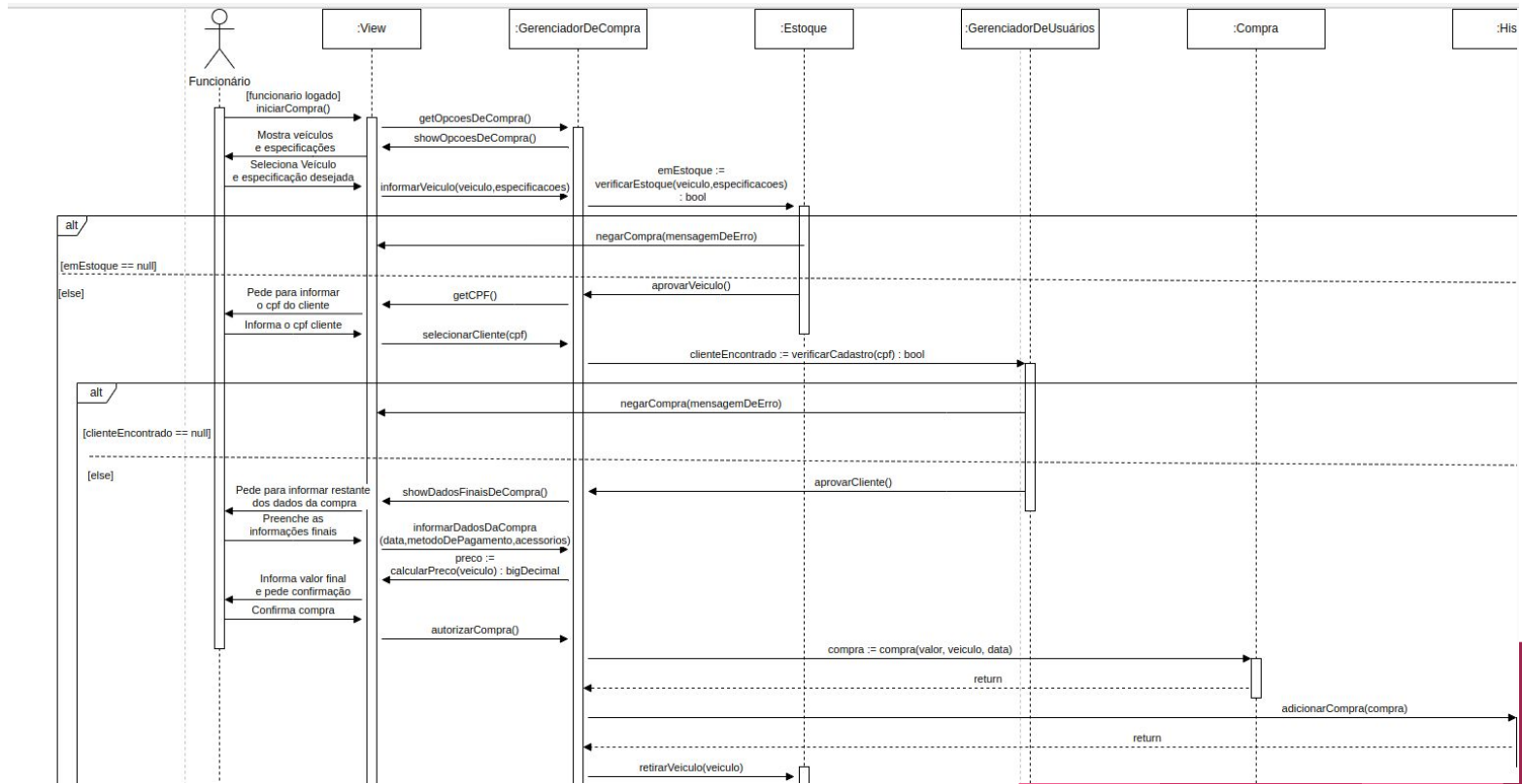


# Modelo Conceitual



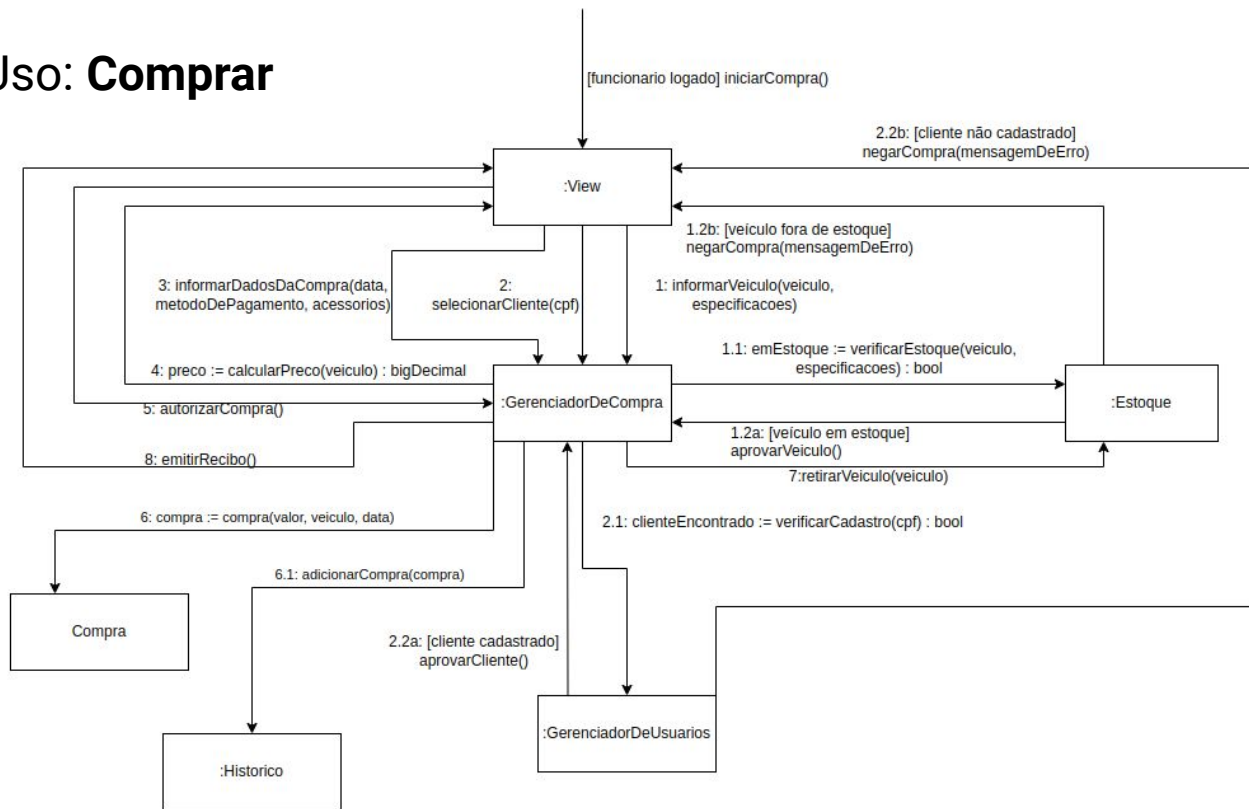


# ● Caso de Uso: Comprar Veículo



# Diagramas de Interação e Estado/Atividade

- Caso de Uso: **Comprar Veículo**



```

classDiagram
    class GerenciadorDeLogin {
        -instance: GerenciadorDeLogin
        -GerenciadorDeLogin()
        +fazerLogin(usuario, senha)
        +getInstance()
        +entrar(usuario)
    }
    class GerenciadorDeRegistro {
        -instance: GerenciadorDeRegistro
        -GerenciadorDeRegistro()
        +cadastrar(email, nome, cpf, dataNasc, telefone)
        +getInstance()
        +cadastrar(email, nome, cpf, dataNasc, telefone, salario, cargo)
        +informarNovoDado(email, nome, dataNasc, telefone)
        +atualizar()
    }
    class GerenciadorDeAgendamento {
        -instance: GerenciadorDeAgendamento
        -GerenciadorDeAgendamento()
        +criarAgendamento()
        +getInstance()
    }
    class Historico {
        -instance: Historico
        -Historico()
        +exibirErro(mensagemDeErro)
        +getInstance()
        +adicionarAgendamento(agendamento)
        +acessarRegistro()
        +adicionarCompra(compra)
    }
    class GerenciadorDeUsuarios {
        -instance: GerenciadorDeUsuarios
        -GerenciadorDeUsuarios()
        +getInstance()
        +aprovarCliente()
        +cadastraCliente(cliente)
        +getUsuario(usuario, senha)
        +cadastraFuncionario(funcionario)
    }
    class Usuario {
        -nome: String
        -senha: String
        -email: String
        -telefone: String
        -cpf: String
        -dataDeNascimento: Date
    }
    class ValidadorDeInformacoes {
        +validaCliente(email, nome, cpf, dataNasc, telefone)
        +validaSenha(senha)
        +atualizar()
        +validaDado(email, nome, dataNasc, telefone)
        +permitirAdicao()
        +cancelarAdicao(mensagemDeErro)
        +validaFuncionario(email, nome, cpf, dataNasc, telefone, salario, cargo)
    }
    class Agendamento {
        -Tipo: String
        -Veiculo: veiculo
        -Status: String
        -Info: String
        -Data: date
        +exibirStatus(agendamento)
    }
    class Registro {
        <<interface>>
    }
    class Compra {
        -Valor: BigDecimal
        -Desconto: BigDecimal
        -Veiculo: veiculo
        -Status: String
        -Data: date
        +exibirStatus(compra)
        +getDetalhes()
    }
    class Estoque {
        -instance: Estoque
        -Estoque()
        +getInstance()
    }
    class Cliente {
        -aracoesSalvas: List<Veiculo1, Veiculo2>
        +email, nome, cpf, dataNasc, telefone)
    }
    class Funcionario {
        -cargo: String
        -bonificacao: BigDecimal
        -salario: BigDecimal
    }
    class Veiculo {
        -modelo: String
        -ano: int
    }

    GerenciadorDeLogin --> Usuario
    GerenciadorDeRegistro --> Usuario
    GerenciadorDeRegistro --> ValidadorDeInformacoes
    GerenciadorDeAgendamento --> Agendamento : cria agendamento
    Historico --> Registro
    GerenciadorDeUsuarios --> Registro
    GerenciadorDeUsuarios --> Compra
    GerenciadorDeUsuarios --> Estoque
    Registro ..> Compra
    Registro ..> Estoque
    Usuario --> Cliente
    Usuario --> Funcionario
    Cliente ..> Cliente : Dependência
    Funcionario ..> Funcionario
    Funcionario ..> Veiculo
    
```

# SOLID e GRASP

## SOLID

- Single Responsibility
- Open-Closed
- Liskov Substitution
- Interface Segregation
- Dependency Inversion

## GRASP

- Especialista
- Criador
- Coesão Alta
- Acoplamento Fraco
- Controlador



# GoF

- Command
- Decorator
- Singleton
- Factory Method
- Observer



# Implementação

- Vamos mostrar um pouco do funcionamento da nossa aplicação...



### Registre o veículo

Modelo:

Ano:

Preço:

REGISTRAR

### Entrar

123.456.789-00

Senha

LOGAR

Não tem uma conta?  
**REGISTRE AGORA!**  
Funcionário novo?  
**JUNTE-SE A EQUIPE!**

### Faça um novo agendamento.

Manutenção

mm/dd/yyyy

Comentários adicionais:

AGENDAR

# Uso de Framework

- Spring Boot
- Thymeleaf
- Hibernate e JPA






# LPS - Linha de Produto de Software




Exemplo:




- Versões diferentes do mesmo carro.




1. Escolha uma

**VERSÃO**

  **TORO ENDURANCE TURBO 270 FLEX AT6 2024**  
R\$ 149.900,00  
 [Mais detalhes](#)

  **TORO FREEDOM TURBO 270 FLEX AT6 2024**  
R\$ 164.190,00  
 [Mais detalhes](#)

  **TORO VOLCANO TURBO 270 FLEX AT6 2024**  
R\$ 178.590,00  
 [Mais detalhes](#)

  **TORO VOLCANO TURBODIESEL 4X4 AT9 2024**  
R\$ 192.390,00  
 [Mais detalhes](#)


Seu carro

**TORO ENDURANCE TURBO 270 FLEX AT6 2024**

R\$ 149.900,00

R\$ 2.282,07 / mês

[Simule as parcelas](#)





# LPS - Características Mandatórias

- Quais características são mandatórias em um Produto Veículo?



Motor



Vidros



Parabrisas

...

# LPS - Características Variáveis (Opcionais e Alternativas)

## Opcionais

3. Escolha os

### KIT OPCIONAIS



**PACOTE TECNOLOGIA**  
R\$ 5.890,00  
[Mais detalhes](#)



**TECNOLOGIA E DESIGN BLACK**  
R\$ 6.500,00  
[Mais detalhes](#)

Seu carro


### TORO VOLCANO TURBO 270 FLEX AT6 2024




## Alternativas

2. Escolha uma


### COR




**SÓLIDAS VERMELHO COLORADO**  
R\$ 0,00



**SÓLIDAS BRANCO AMBIENTE**  
R\$ 1.500,00




**METÁLICAS PRETO CARBON**  
R\$ 2.490,00



**METÁLICAS GRANITE CRYSTAL**

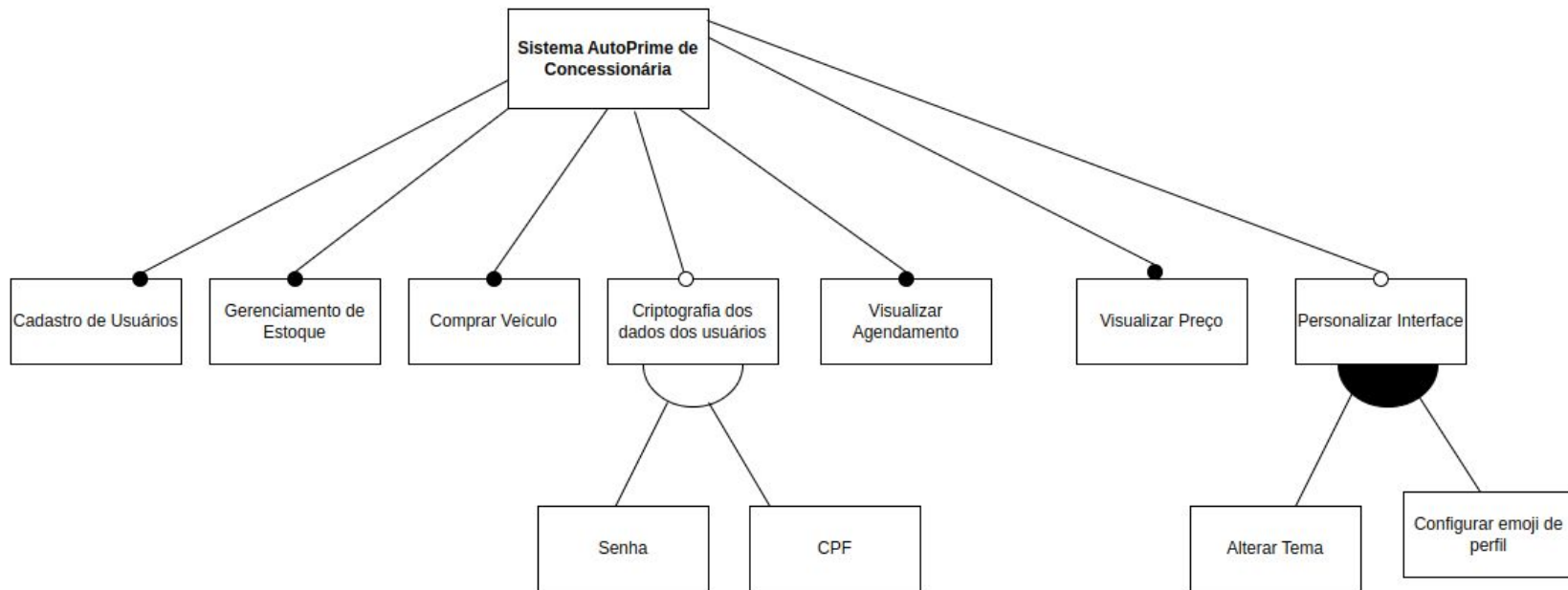
Seu carro

### TORO ENDURANCE TURBO 270 FLEX AT6 2024



# LPS para Sistema de Concessionária

## LPS AutoPrime



# Dificuldades Encontradas

- Adaptação dos Padrões de Projeto a implementação do sistema.
- Dificuldade com as ferramentas que escolhemos para desenvolver o sistema.
- Tempo compatível entre os membros do grupo para a realização de reuniões.



Obrigado :)

