

UNIVERSIDADE PAULISTA – UNIP

GUILHERME CAPUCCI AMBRÓSIO

HENRIQUE DE SOUZA SANTOS MARQUES

IGOR SOUZA DE OLIVEIRA

PEDRO HENRIQUE DESANI HILARIO

RAFAEL CAVALINE GOMES

SISTEMA PARA CADASTRO DE FROTAS E LOCAÇÕES DE VEÍCULOS

**SÃO JOSÉ DO RIO PRETO
2019**

UNIVERSIDADE PAULISTA – UNIP

GUILHERME CAPUCCI AMBRÓSIO

HENRIQUE DE SOUZA SANTOS MARQUES

IGOR SOUZA DE OLIVEIRA

PEDRO HENRIQUE DESANI HILARIO

RAFAEL CAVALINE GOMES

SISTEMA PARA CADASTRO DE FROTAS E LOCAÇÕES DE VEÍCULOS

Trabalho PIM como parte do exigido para obtenção do título de graduação em Análise e Desenvolvimento de Sistemas apresentado à Universidade Paulista – UNIP.

**SÃO JOSÉ DO RIO PRETO
2019**

RESUMO

O projeto tem como objetivo principal desenvolver habilidades e conhecimento a respeito do desenvolvimento de um sistema para realizar o cadastro de frotas e controle de locações de veículos linguagem C# com conexão ao banco na linguagem SQL, onde juntamente com a programação em si, realizar o projeto completo desde o planejamento, levantamento de requisitos, necessidades, abordagem de todos os diagramas apresentados em sala e em pesquisas

Palavra Chave: Linguagem C#, SQL, Desenvolvimento, Diagrama.

ABSTRACT

The main objective of the project is to develop skills and knowledge regarding the development of a system to perform fleet registration and control of vehicle leases C # language with connection to the bank in the SQL language, where together with the programming itself, carry out the project complete from the planning, requirements survey, needs, approach of all the diagrams presented in the room and researches

Key Words: Language C#, SQL, Development, Project, diagrams.

LISTA DE FIGURAS

Imagem 1 – Excel Cadastro de Veiculos	7
Imagem 2 – Sistema Cadastro de Cliente	8
Imagem 3 – Excel Manutenções de Veiculos	9
Imagem 4 – Sistema Manutenções de Veiculos	9
Imagem 5 – Excel Cadastro de Viagens/Locações	10
Imagem 6 – Sistema Cadastro de Viagens/Locações	11
Imagem 7 – Sistema Lançamento de Multas	11
Imagem 8 – Sistema Cadastro de Cliente	12
Imagem 9 – Sistema Cadastro de Garagens	13
Imagem 10 – Sistema Cadastro de Veiculos	13
Imagem 11 – Excel Relatórios	14
Imagem 12 – Sistema Relatórios	15
Imagem 13 – Diagrama de Classe	19
Imagem 14 – Diagrama Entidade Relacionamento	20
Imagem 15 – Diagrama de Sequencia	20
Imagem 16 – Script Criação Tabela cdcliente Banco de Dados MYSQL	21
Imagem 17 – Script Classe DAO.	24
Imagem 18 – Script Classe ClienteDAO.	25
Imagem 19 – Script Classe ClienteBLL.	29
Imagem 20 – Script Classe Fom1 – Formulário Cadastro Cliente.	30
Imagem 21 – Script Classe fmrInicial.	36
Imagem 22 – Script Classe fmrLogin.	38
Imagem 23 – Tela Estoque saída.	39
Imagem 24 – Tela Estoque entrada.	39
Imagem 25 – Lançamento Financeiro de dividas.	40
Imagem 26 – Cadastro de Cliente	41
Imagem 27 – Cadastro de Veiculos.	42
Imagem 28 – Cadastro de Garagem.	43
Imagem 29 – Lançamento de Manutenções.	43
Imagem 30 – Lançamento de Multas para Frota e Cliente.	44
Imagem 31 – Consulta Relatórios.	45

Imagem 32 – Lançamento de Viagens	45
Imagem 33 – Menu do Sistema.	46
Imagem 34 – Tela de Login do Sistema.	47
Imagem 35 – Tela do Assistente de Instalação (1/4)	48
Imagem 36 – Tela do Assistente de Instalação (2/4)	48
Imagem 37 – Tela do Assistente de Instalação (3/4)	49
Imagem 38 – Tela do Assistente de Instalação (4/4)	49

LISTA DE QUADROS/TABELAS

Quadro 1 – Telas de Cadastro/Inclusão	17
Quadro 2 – Cadastrar Viagem/Locação	18
Quadro 3 – Cadastrar Fornecedor	21
Quadro 4 – Relacionamento Fornecedor/Estoque	22
Quadro 5 – Estoque	22
Quadro 6 – Manutenção	22
Quadro 7 – Cadastrar Garagem	22
Quadro 8 – Cadastrar Veiculos	23
Quadro 9 – Cadastrar Cliente	23
Quadro 10 – Lançamento Financeiro	23
Quadro 11 – Lançamento Multas	24
Quadro 12 – Locação Carro/Viagem	24

SUMÁRIO

1 - CENÁRIO DA EMPRESA	6
2 - EMPRESA FROTAS JOAQUIM	6
3 - COMPARATIVO COM A CONCORRÊNCIA	7
3.1 – Tela de Cadastro de veículos	7
3.2- Tela de Manutenções	9
3.3 – Tela Viagens	10
3.4 – Tela de Lançamento de Multas	11
3.5 – Tela de Cadastro de Clientes	12
3.6 – Tela de Cadastro de Garagens	13
3.7– Tela de Cadastro de Veículos	13
3.8 – Tela de relatório	14
4 - REGRAS DE NEGOCIO DO SISTEMA	15
4.1 - Cliente	15
4.2 - Veiculo	15
4.3 - Cobrança	16
4.4 - Contrato	16
4.5 - Seguro	16
4.6 - Estoque	17
5 - CASO DE USO MODELAGEM DO SISTEMA	17
6 - DIAGRAMAS	19
6.1 - Diagrama de Classe	19
6.2 - Diagrama Entidade e Relacionamento	20
6.3 - Diagrama Sequencia – Aluguel de Veículos	20
7 - Linguagem de Definição de Dados	21
7.1 - Script Criação Banco de Dados	21
7.2 - Dicionário de Dados	21
8 - Desenvolvimento C# CRUD de Interface	24
8.1 - Script C# do Sistema	24
9 - MANUAIS DE UTILIZAÇÃO E INTEFACE DE TELAS	39
10.1 - Requisitos Básicos de Instalação	47
10.2 - Instalação do Software	48
11 - CONTRATO DE SERVICO E MANUTENÇÃO DE SOFTWARE DZVOLVE	50
12 – PROMOÇÕES	52
REFERENCIA	53

1 - CENÁRIO DA EMPRESA

A empresa DZVolve, situada na cidade de São José do Rio Preto – SP, atua no ramo de TI, fornecendo soluções de infraestrutura e desenvolvimento de aplicações para toda a região.

Com mais de 5 anos no mercado, atuamos sempre buscando a melhor solução em software para nossos clientes.

A aplicação “Frotas do Joaquim” foi criada a partir da necessidade de controle de frotas para a empresa “Joaquim S/A”, que atua no ramo de alugueis de veículos.

A Joaquim S/A procurou nossa empresa em busca de uma solução sob medida para seu projeto de gestão de frotas, nascendo assim a aplicação “Frotas do Joaquim”, nome no qual a empresa solicitou. A aplicação possibilitou o controle de clientes, veículos disponíveis, manutenções, gastos adicionais, controle de estoque de peças, finanças, entre outras mais funções.

Hoje, a aplicação está entre as melhores do mercado, e ajudando cada vez mais que a empresa “Joaquim S/A” seja pioneira no mercado de aluguel automotivo.

2 - EMPRESA FROTAS JOAQUIM

A Empresa foi criada para atender a necessidade de Frotas do Joaquim sobre o controle de frotas de veículos. Visando sempre a qualidade e máxima performance em aplicação, nossos sistemas são pensados sempre no usuário, visando a facilidade de uso e através de treinamentos, instruir o usuário para maior velocidade de operação.

Objetivo principal é cadastro de frota e controle da empresa juntamente com o sistema de locação de veículos e controle de motorista. Segundo informações disponíveis no relatório de atividades do ANAV o setor de locação de veículos, impactado pelas condições gerais logrou neste ano obter resultados em geral positivos na adequação e inovação das atividades.

3 - COMPARATIVO COM A CONCORRÊNCIA

A aplicação concorrente aqui apresentada, é parametrizada em cima de outro software, no caso, Excel. A aplicação Excel possui custos de licenciamento elevados, levando nossa aplicação a uma vantagem sobre a mesma.

Também notamos que em caso de atualizações da ferramenta Excel, algumas funções da planilha podem deixar de funcionar, visto que a base não foi pensada para a aplicação em si.

Problemas com o pacote Office influenciariam diretamente na aplicação, e em caso de suporte, poderiam surgir tanto problemas com o pacote, quanto com a aplicação, tornando a solução inviabilizável para ambientes de grande porte, visto que não dependeria somente da empresa responsável pela criação da aplicação, a resolução destes problemas.

3.1 – Tela de Cadastro de veículos

Imagem 1 – Excel Cadastro de Veiculos

ID	Tipo de veículo	Marca do veículo	Placa	Possui Kit Gás	Carga máxima permitida (em kg)	Rodagem inicial (em km)	Cor	Ano	Pcto. IPVA	Pcto. Seguro	Pcto. Kit Gás
1	Caminhão	Iveco	DLP3875	Não	5.000	10.000	Azul	2010	Não Pago	Não Possui	Não Possui
2	Caminhão	Scania	KYD2306	Não	2.000	5.000	Vermelho	2011	Pago	Pago	Não Possui
3	Carro	Siena	JIG8745	Sim	250	3.000	Verde	2015	Pago	Não Pago	Não Pago
4	Moto	Harley	HRJ5423	Não	50	200	Amarelo	2017	Não Pago	Pago	Não Possui

Fonte: luz.vc, 2019.

A tela de cadastro de veículos de uma concorrente é inteiramente feita em Excel, propiciando erros ao preencher campos, de forma que a validação não seja tão eficiente quanto em nossa aplicação.

Desta forma, o usuário pode acabar se confundindo ao inserir informações e criando um cadastro problemático para a empresa.

Imagem 2 – Sistema Cadastro de Cliente

A imagem mostra uma janela de software intitulada "Clientes". No topo, há uma barra de ferramentas com ícones para "Salvar", "Editar", "Excluir" e "Cancelar Cadastro". Abaixo, há duas opções de seleção: "Pessoa Física" (selecionada) e "Pessoa Jurídica". O formulário contém vários campos de entrada:

- Nome: [campo de texto]
- CPF/CNPJ: [campo de texto]
- RG: [campo de texto]
- Nº Cnh: [campo de texto]
- Vncd Cnh: [campo de texto]
- Org. Exp: [campo de texto]
- Uf. Exp: [campo de texto]
- Data Nasc.: [campo de texto]
- Cep-Rua/Av: [campo de texto]
- Nº: [campo de texto]
- Bairro: [campo de texto]
- Compl.: [campo de texto]
- Município: [campo de texto]
- Uf: [campo de texto]
- Telefone: [campo de texto]
- Celular: [campo de texto]
- E-Mail: [campo de texto]

Na base da janela, há uma área cinza retangular, provavelmente destinada a uma lista de clientes ou uma visualização de detalhes.

Fonte: Autores, 2019.

Nossa tela de cliente possui toda validação necessária para que o usuário consiga preencher todos os campos de forma correta, visto que a nossa base é totalmente pensada a atender a aplicação, possibilitando que todas as funções sejam validadas corretamente.

3.2- Tela de Manutenções

Imagem 3 – Excel Manutenções de Veículos

ID	Tipo de Veículo	Veículo	Necessidade de Trocas	Km Total Rodado Pelo Veículo	Km de cadastro da última troca de pneus	Troca pneus de quantos em quantos Kms	Rodagem sem troca de pneus	Precisa trocar pneus?	Km de cadastro da última troca de óleo	Troca óleo de quantos em quantos Kms	Rodagem sem troca de óleo	Precisa trocar o óleo?	Km de cadastro da última troca de freios	Troca freios de quantos em quantos Kms
1	Caminhão	Iveco - OLP3875	Alguma troca é necessária	12000	10500	1000	1500	Sim	1400	10000	10600	Sim	12000	4000
2	Caminhão	Scania - KYD2306	Veículo em dia	5000	0	50000	5000	Não	0	10000	5000	Não	4000	4000
3	Caminhão	Siema - JIG8745	Veículo em dia	6100	0	40000	6100	Não	0	10000	6100	Não	5000	4000
4	Moto	Harley - HRU5423	Veículo em dia	200	0	20000	200	Não	0	5000	200	Não	0	-
5	-	-	-	0	-	-	0	-	-	-	0	-	-	-
6	-	-	-	0	-	-	0	-	-	-	0	-	-	-
7	-	-	-	0	-	-	0	-	-	-	0	-	-	-
8	-	-	-	0	-	-	0	-	-	-	0	-	-	-
9	-	-	-	0	-	-	0	-	-	-	0	-	-	-
10	-	-	-	0	-	-	0	-	-	-	0	-	-	-
11	-	-	-	0	-	-	0	-	-	-	0	-	-	-
12	-	-	-	0	-	-	0	-	-	-	0	-	-	-
13	-	-	-	0	-	-	0	-	-	-	0	-	-	-
14	-	-	-	0	-	-	0	-	-	-	0	-	-	-

Fonte: luz.vc, 2019.

A tela de manutenções também apresenta o mesmo problema da tela de clientes, podendo levar o usuário a digitar valores inválidos pela forma na qual foi criada.

Imagem 4 – Sistema Manutenções de Veículos

Manutencoes

Placa: Valor da Manutenção:

Descrição:

Data Inicial: Previsão para entrega:

Salvar Excluir Sair

Fonte: Autores, 2019.

Nossa tela de manutenções apresenta interface minimalista, proporcionando que o usuário preencha somente dados básicos sobre a manutenção, e após, registrar diretamente em uma base de dados que poderá ser consultada via tela de relatório posteriormente.

3.3 – Tela Viagens

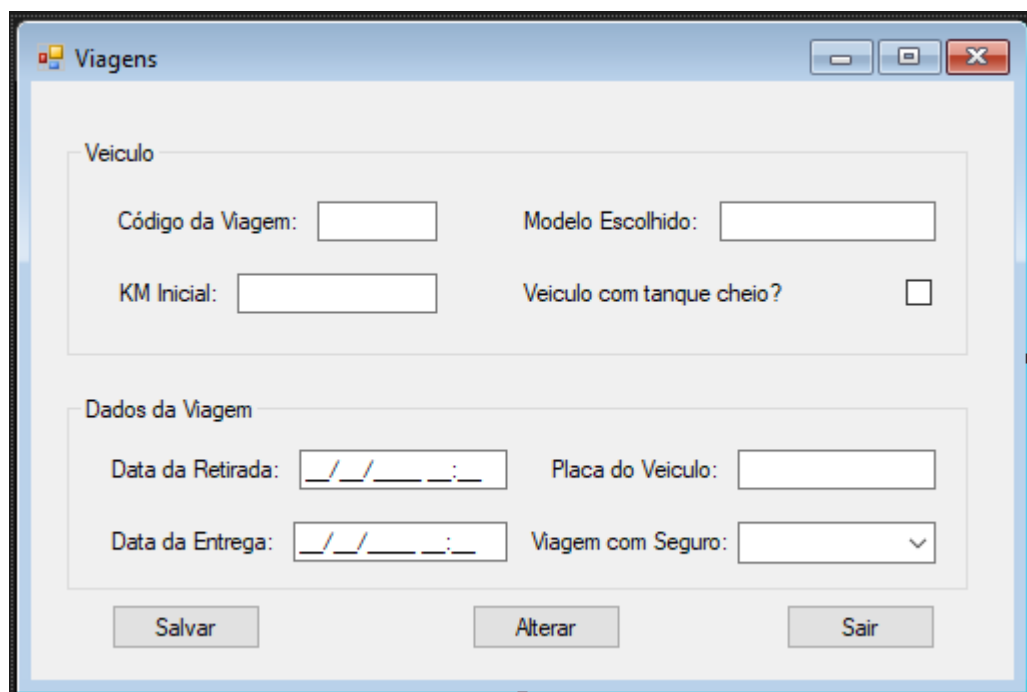
Imagem 5 – Excel Cadastro de Viagens/Locações

		JANEIRO	FEVEREIRO	MARÇO	ABRIL	MAIO	JUNHO	JULHO	AGOSTO	SETEMBRO	OUTUBRO	NOVEMBRO	DEZEMBRO	
Dia	Veículo	Valores pagos em pedágio	Valores pagos em multas	Posto	Combustível	Valor Unitário	Gasto com Combustível	Litros	Outros custos	Peso da carga transportada (Kg)	Km Inicial	Km Final	Km Total	Motorista
2	Iveco - OLP3875	R\$ 100,00	R\$ 0,00	Posto Oeste	Diesel	R\$ 3,08	R\$ 180,00	58,4	R\$ 100,00	3000	10.000	10.550	550	Jarbas
4	Iveco - OLP3875	R\$ 83,30	R\$ 195,23	Posto Leste	Diesel	R\$ 3,22	R\$ 180,00	55,9	R\$ 50,00	5500	10.550	11.200	650	Jarbas
4	Siena - JIG8745	R\$ 58,30	R\$ 130,23	Posto Norte	Gasolina	R\$ 3,90	R\$ 140,00	35,9	R\$ 20,00	200	5.000	5.650	650	Leônidas
6	Siena - JIG8745	R\$ 39,20	R\$ 130,23	Posto Sul	Gasolina	R\$ 3,98	R\$ 140,00	35,2	R\$ 40,00	200	5.650	6.200	550	Leônidas
6	Siena - JIG8745	R\$ 30,00	R\$ 100,00	Posto Sul	Diesel	R\$ 3,90	R\$ 200,00	51,3	R\$ 20,00	251	6.200	6.300	100	Leônidas

Fonte: luz.vc, 2019.

A tela de viagens contém o mesmo erro de todas as outras telas, porém, além disto, contém informações adicionais que poderiam ser incluídas em outras telas, de modo que a interface não seja poluída e demorada para que o usuário preencha.

Imagem 6 – Sistema Cadastro de Viagens/Locações



Viagens

Veiculo

Código da Viagem: Modelo Escolhido:

KM Inicial: Veiculo com tanque cheio? ☐

Dados da Viagem

Data da Retirada: Placa do Veiculo:

Data da Entrega: Viagem com Seguro:

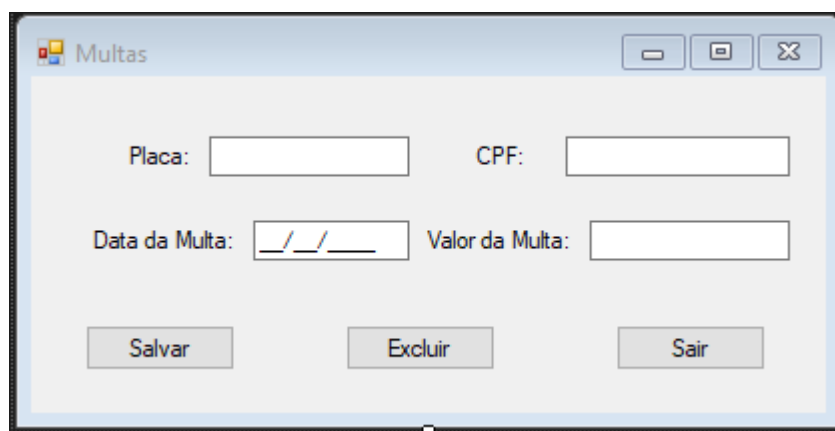
Salvar Alterar Sair

Fonte: Autores, 2019.

Nossa tela apresenta novamente interface totalmente a prova de dados inválidos, e precisa somente de informações básicas para que o cadastro seja efetuado de forma rápida e eficiente.

3.4 – Tela de Lançamento de Multas

Imagem 7 – Sistema Lançamento de Multas



Multas

Placa: CPF:

Data da Multa: Valor da Multa:

Salvar Excluir Sair

Fonte: Autores, 2019.

Dentro da mesma, apresenta tela de multas de forma independente, para que o cadastro seja feito em forma de fluxo, agilizando para que o usuário preencha somente o necessário em cada etapa.

3.5 – Tela de Cadastro de Clientes

Imagem 8 – Sistema Cadastro de Cliente

A imagem mostra uma janela de software intitulada "Clientes". No topo, há duas opções de seleção: **Pessoa Física** (selecionada) e **Pessoa Juridica**. À direita, há ícones e rótulos para **Salvar**, **Editar**, **Excluir** e **Cancelar Cadastro**. O formulário contém os seguintes campos:

- Nome: [campo de texto]
- CPF/CNPJ: [campo de texto]
- RG: [campo de texto]
- Nº Cnh: [campo de texto]
- Vncd Cnh: [campo de texto]
- Org. Exp: [campo de texto]
- Uf. Exp: [campo de texto]
- Data Nasc.: [campo de data]
- Cep-Rua/Av: [campo de texto]
- Nº: [campo de texto]
- Bairro: [campo de texto]
- Compl.: [campo de texto]
- Município: [campo de texto]
- Uf: [campo de texto]
- Telefone: [campo de texto]
- Celular: [campo de texto]
- E-Mail: [campo de texto]

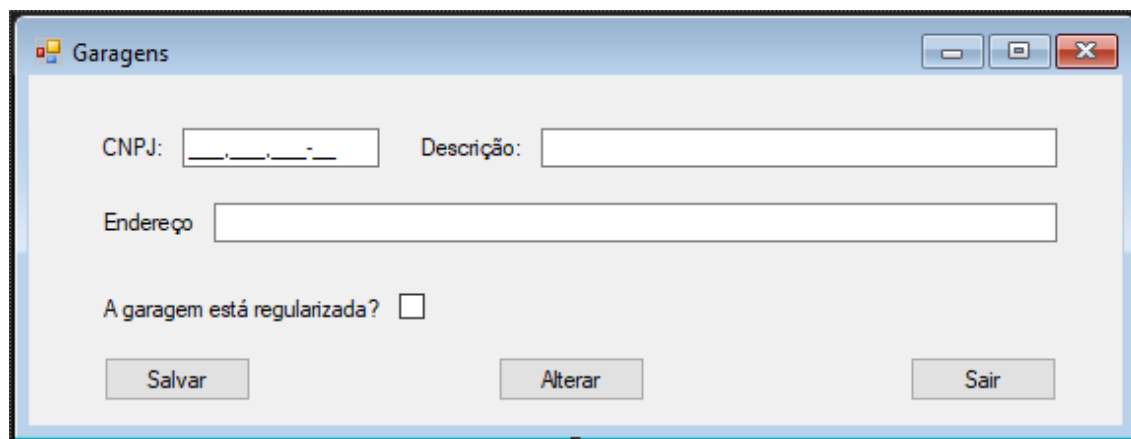
Abaixo dos campos, há uma grande área cinza retangular, provavelmente destinada a uma lista ou visualização de detalhes.

Fonte: Autores, 2019.

Nosso software também apresenta tela completa para cadastro de cliente, no qual fornece informações importantes sobre quem estará utilizando o veículo.

3.6 – Tela de Cadastro de Garagens

Imagem 9 – Sistema Cadastro de Garagens



A screenshot of a software window titled "Garagens". It contains the following fields and controls:

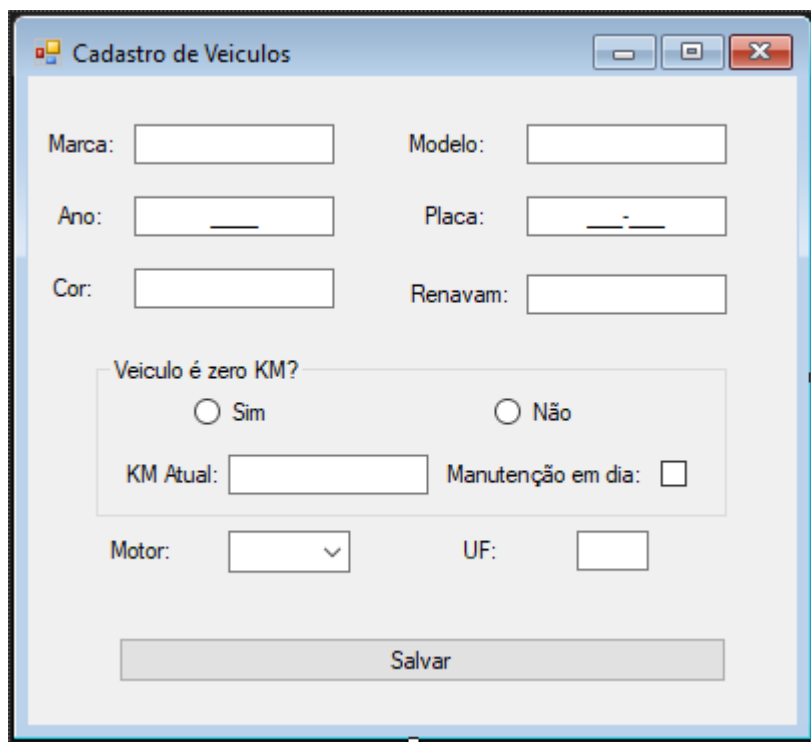
- CNPJ: A text box with a pre-filled format of "____.____.____-____".
- Descrição: A single-line text box.
- Endereço: A single-line text box.
- A garagem está regularizada?: A checkbox.
- Buttons: "Salvar", "Alterar", and "Sair" are positioned at the bottom.

Fonte: Autores, 2019.

Fornecemos cadastro de garagens credenciadas, para caso a empresa deseje parcerias com estacionamentos e afins.

3.7– Tela de Cadastro de Veículos

Imagem 10 – Sistema Cadastro de Veiculos



A screenshot of a software window titled "Cadastro de Veiculos". It contains the following fields and controls:

- Marca: A text box.
- Modelo: A text box.
- Ano: A text box with a pre-filled format of "____".
- Placa: A text box with a pre-filled format of "____-____".
- Cor: A text box.
- Renavam: A text box.
- Veiculo é zero KM?: A section containing two radio buttons labeled "Sim" and "Não".
- KM Atual: A text box.
- Manutenção em dia: A checkbox.
- Motor: A dropdown menu.
- UF: A text box.
- Button: A large "Salvar" button at the bottom.

Fonte: Autores, 2019.

Nosso software fornece cadastro completo para veículos, possibilitando que a empresa tenha total controle das frotas que estão em sua posse.

3.8 – Tela de relatório

Imagem 11 – Excel Relatórios

Mes	January	February	March	April	May	June	July	August	September	October	November	December	Total
Gastos com combustível	R\$640,00	R\$640,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$1.480,00
Gastos com pedágios	R\$110,80	R\$229,50	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$540,30
Gastos com multas	R\$555,69	R\$130,23	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$685,92
Gastos com manutenção	R\$700,00	R\$2.000,00	R\$0,00	R\$2.000,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$4.700,00
Outros gastos	R\$230,00	R\$210,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$440,00
Gasto Total	R\$2.636,49	R\$3.209,73	R\$0,00	R\$2.000,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$7.846,22
Gastos com Categorias													
Carro	R\$1.047,96	R\$567,73	R\$0,00	R\$2.000,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$3.615,69
Caminhão	R\$1.588,53	R\$2.642,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$4.230,53
Moto	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00
Ônibus	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00
	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00
	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00
	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00
	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00	R\$0,00

Fonte: luz.vc, 2019.

Por se tratar de uma aplicação Excel, caso a aplicação seja corrompida ou avariada, todos os dados referentes a relatórios serão perdidos.

Imagem 12 – Sistema Relatórios

Fonte: Autores, 2019.

Nossa tela de relatórios apresenta confiabilidade de que a informação será buscada diretamente de uma base de dados, apresentando estabilidade para que a empresa possa realizar backups e consultas seguras.

4 - REGRAS DE NEGOCIO DO SISTEMA

4.1 - Cliente

RN01 - CNH Valida em todo território nacional;

RN02 - CPF sem restrição;

RN03 - Score/Cartão de Crédito de no mínimo 7% da tabela FIPE do veículo escolhido, como seguro financeiro sobre a locação, devolvido/estornado na entrega do veículo;

RN04 - O cliente só pode alugar um carro por vez, não podendo ter dois alugueis em seu CPF, ou estar cadastrado como motorista auxiliar em outras viagens;

4.2 - Veiculo

RN01 - Reserva de 15 dias antes da locação do veiculo mediante a caução ou seguro financeiro do veículo, como ato de sinal sobre a reserva, período de locação entre 1 a 30 dias;

RN02 - Toda a reserva pode sofrer alterações mediante a atualização da base cálculo do dia;

RN03 - Todo veículo que esteja danificado necessita de um adendo do contrato juntamente com a multa caso o item danificado seja causado pelo locador, caso seja por causas naturais será incluído no valor um reajuste com desconto devido aos danos morais causado;

RN04 - Todo veículo deve ser feito check-list antes/depois da locação, os veículos devem estar todos em condições de uso;

RN05 - Todos os veículos devem ter no máximo 3 anos de uso com base no ano atual;

RN06 - Todos veículos devem ter no máximo 70Mil Km;

RN07 - Veículos que as condições de uso não estejam adequados, ou superior a 3 anos de uso ou 70 Mil KM, devem ser disponibilizados para leilão;

RN08 - Qualquer alteração no cadastro dos veículos só pode ser feita pelo administrador do sistema;

4.3 - Cobrança

RN01 - O aluguel é calculado pela diária de locação, com limite máximo de 3000 km no período máximo de 1 mês (30 dias); caso o KM seja superior, é calculado o valor do aluguel sobre os km excedentes;

4.4 - Contrato

RN01 - Validade do contrato é sobre todo território nacional, caso haja necessidade de alteração sobre abrangência do território nacional/internacional o contrato deve ter autorização das duas partes;

4.5 - Seguro

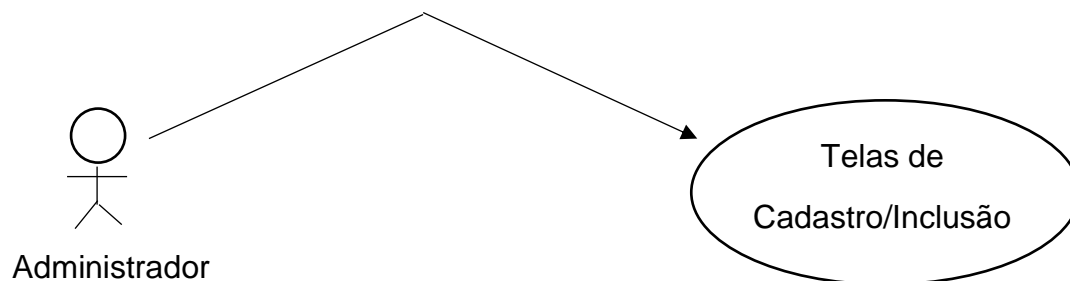
RN01 - Toda locação pode ter no máximo 5 motoristas, mas todos motoristas devem ter seguros individuais;

RN02 - Caso o Cliente deseje utilizar o seguro próprio, todo contrato de aluguel devem ter no o nosso seguro básico para evitar quaisquer transtornos futuros em processos cíveis;

4.6 - Estoque

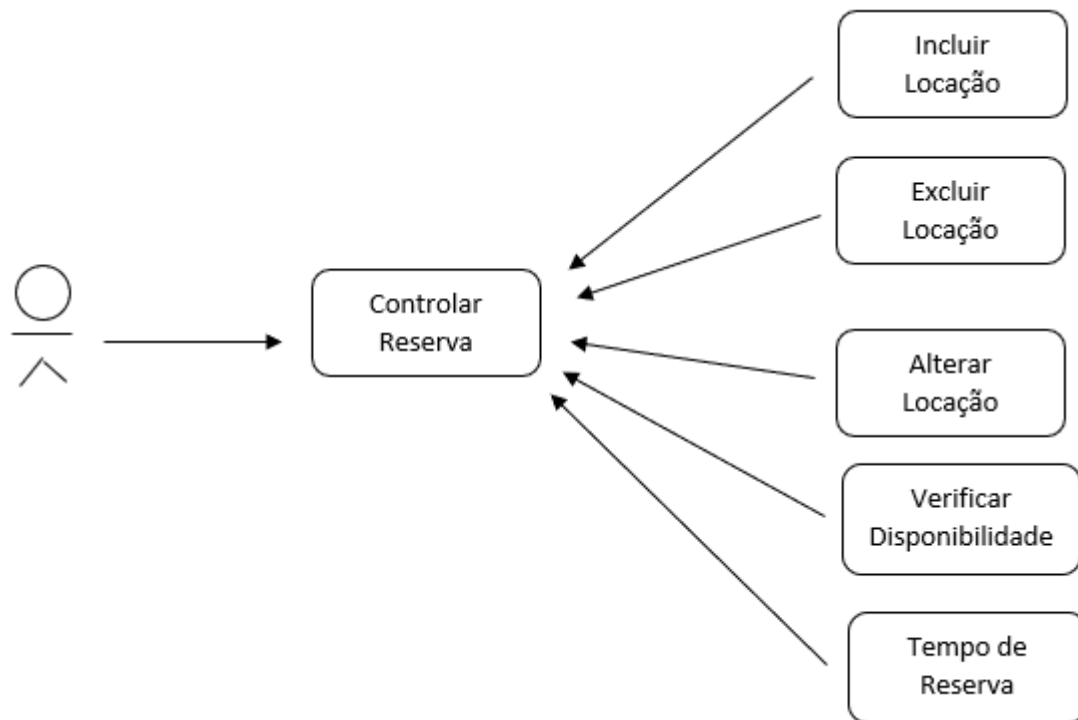
RN01 - A entrada de mercadoria só é permitida caso o cadastro do fornecedor esteja correto, juntamente com a quantidade e valor do produto;

5 - CASO DE USO MODELAGEM DO SISTEMA



Nome USECASE	Cadastrar Cliente
Ator	Administrador
Descrições	O cadastro deverá ser preenchido de acordo com os campos para que não haja nenhum erro de cadastro
Cenário Principal	O sistema mostrará os campos a serem preenchidos; O ator deverá preencher os campos necessários; Após preencher corretamente o cadastro será finalizado;
Cenário alternativo	O Ator não confirma cadastro

Quadro 1 – Telas de Cadastro/Inclusão



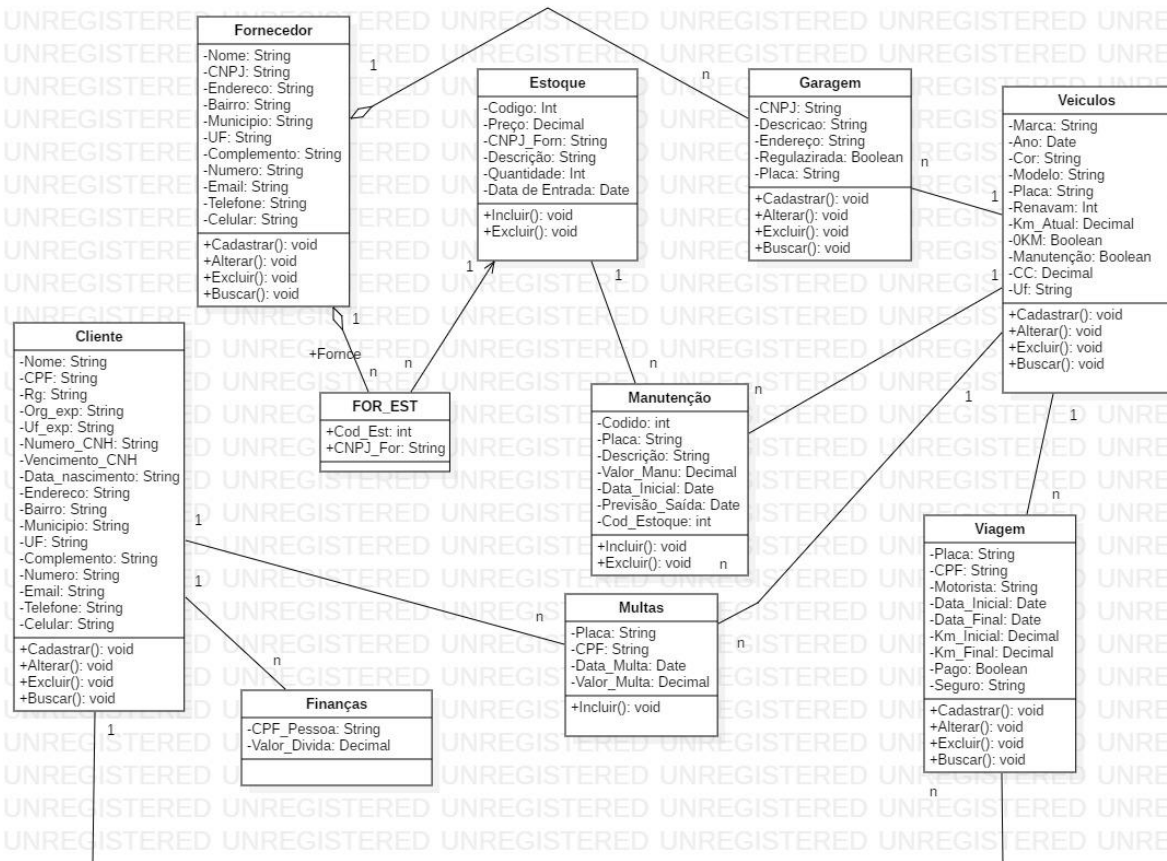
Nome USECASE	Cadastrar Viagem/Locação
Ator	Administrador
Descrição	<p>O sistema mostrará uma interface para o ator preencher os dados da locação;</p> <p>O sistema mostrará uma interface onde todos os campos devem ser preenchidos corretamente pelo ator;</p> <p>O administrador localizará a locação original e irá definir a data de entrega;</p> <p>O sistema mostra uma interface gráfica onde o administrador devesse preencher os campos corretamente</p>
Cenário Principal	O ator deverá preencher todos os campos para que não haja nenhum erro de locação
Cenário Alternativo	Locação não realizada

Quadro 2 – Cadastrar Viagem/Locação

6 - DIAGRAMAS

6.1 - Diagrama de Classe

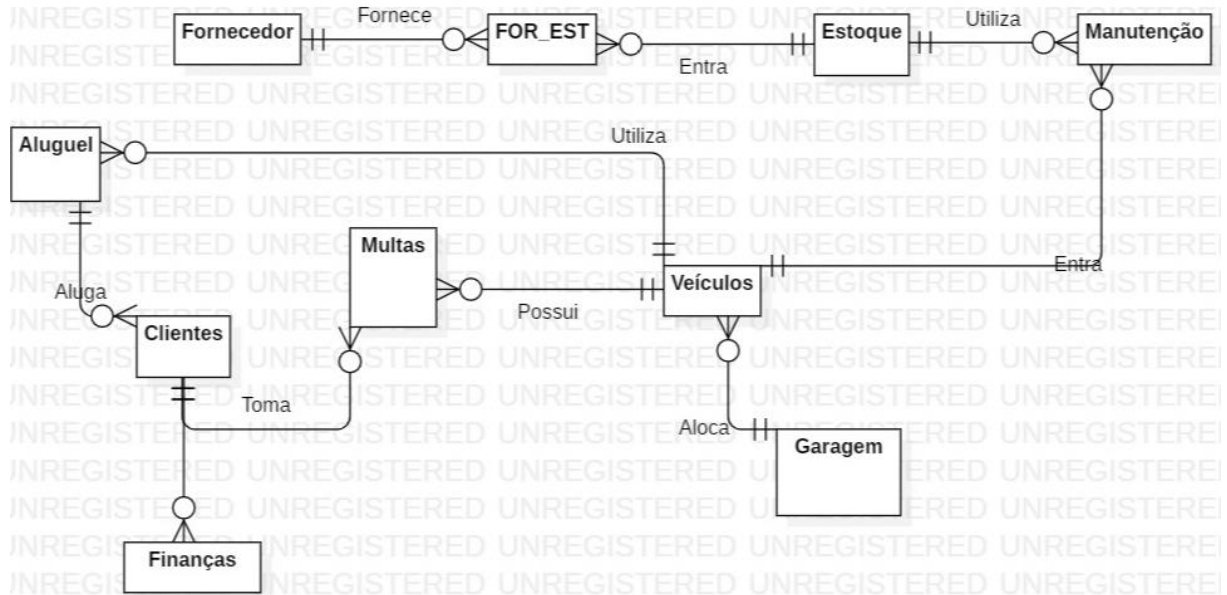
Imagem 13 – Diagrama de Classe



Fonte: Autores, 2019.

6.2 - Diagrama Entidade e Relacionamento

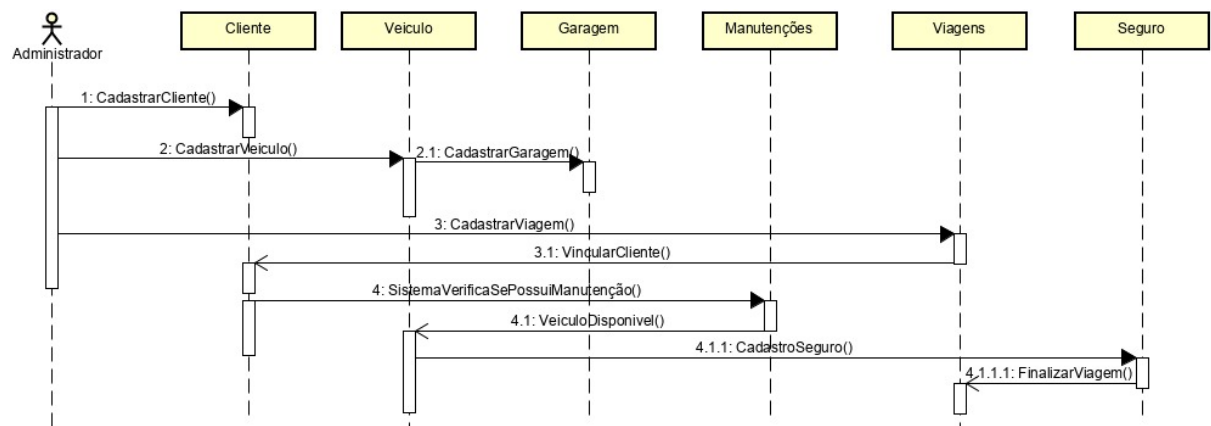
Imagem 14 – Diagrama Entidade Relacionamento



Fonte: Autores, 2019.

6.3 - Diagrama Sequencia – Aluguel de Veículos

Imagem 15 – Diagrama de Sequencia



Fonte: Autores, 2019.

7 - Linguagem de Definição de Dados

7.1 - Script Criação Banco de Dados

Imagem 16 – Script Criação Tabela cdcliente Banco de Dados MYSQL

```

1  CREATE TABLE `cdcliente` (
2      `nome` varchar(50) NOT NULL,
3      `cpf` char(11) NOT NULL,
4      `rg` char(15) NOT NULL,
5      `ORG_exp` char(4) NOT NULL,
6      `uf_exp` char(2) NOT NULL,
7      `cnh` char(11) NOT NULL,
8      `vencimento_cnh` varchar(12) NOT NULL,
9      `nascimento_data` varchar(12) NOT NULL,
10     `endereco` varchar(50) NOT NULL,
11     `municipio` varchar(30) NOT NULL,
12     `uf` varchar(2) NOT NULL,
13     `complemento` varchar(50) NOT NULL,
14     `bairro` varchar(50) NOT NULL,
15     `numero_endereco` varchar(8) NOT NULL,
16     `email` varchar(50) NOT NULL,
17     `telefone` varchar(10) NOT NULL,
18     `celular` varchar(11) NOT NULL,
19     PRIMARY KEY (`cpf`),
20     UNIQUE KEY `cpf_UNIQUE` (`cpf`)
21 ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci

```

Fonte: Autores, 2019.

7.2 - Dicionário de Dados.

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
NOME	STRING	NÃO	NÃO	NÃO
CNPJ	STRING	NÃO	SIM	NÃO
ENDERECO	STRING	NÃO	NÃO	NÃO
BAIRRO	STRING	NÃO	NÃO	NÃO
MUNICIPIO	STRING	NÃO	NÃO	NÃO
UF	STRING	NÃO	NÃO	NÃO
COMPLEMENTO	STRING	NÃO	NÃO	NÃO
NUMERO	STRING	NÃO	NÃO	NÃO
EMAIL	STRING	NÃO	NÃO	NÃO
TELEFONE	STRING	NÃO	NÃO	NÃO
CELULAR	STRING	NÃO	NÃO	NÃO

Quadro 3 – Cadastrar Fornecedor

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
COD_EST	INT	NÃO	SIM	SIM
CNPJ_FOR	STRING	NÃO	SIM	SIM

Quadro 4 – Relacionamento Fornecedor/Estoque

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
CODIGO	INT	NÃO	SIM	NÃO
PREÇO	DECIMAL	NÃO	NÃO	NÃO
CNPJ_FORN	STRING	NÃO	NÃO	NÃO
DESCRIÇÃO	STRING	NÃO	NÃO	NÃO
QUANTIDADE	STRING	NÃO	NÃO	NÃO
DATA DE ENTRADA	DATE	NÃO	NÃO	NÃO

Quadro 5 – Estoque

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
CODIGO	INT	NÃO	SIM	NÃO
PLACA	STRING	NÃO	NÃO	SIM
DESCRIÇÃO	STRING	NÃO	NÃO	NÃO
VALOR MANU	DECIMAL	NÃO	NÃO	NÃO
DATA INICIAL	DATE	NÃO	NÃO	NÃO
PRIVISÃO_SAÍDA	DATE	NÃO	NÃO	NÃO
COD_ESTOQUE	INT	NÃO	NÃO	SIM

Quadro 6 – Manutenção

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
CNPJ	STRING	NÃO	SIM	NÃO
DESCRIÇÃO	STRING	NÃO	NÃO	NÃO
ENDEREÇO	STRING	NÃO	NÃO	NÃO
REGULARIZADA	BOOLEAN	NÃO	NÃO	NÃO
PLACA	STRING	NÃO	NÃO	SIM

Quadro 7 – Cadastrar Garagem

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
MARCA	STRING	NÃO	NÃO	NÃO
ANO	DATE	NÃO	NÃO	NÃO
COR	STRING	NÃO	NÃO	NÃO
MODELO	STRING	NÃO	NÃO	NÃO
PLACA	STRING	NÃO	SIM	NÃO
RENAVAM	INT	NÃO	NÃO	NÃO
KM_ATUAL	DECIMAL	NÃO	NÃO	NÃO
OKM	BOOLEAN	NÃO	NÃO	NÃO
MANUTENÇÃO	DECIMAL	NÃO	NÃO	NÃO
CC	DECIMAL	NÃO	NÃO	NÃO
UF	STRING	NÃO	NÃO	NÃO

Quadro 8 – Cadastrar Veiculos

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
NOME	STRING	NÃO	NÃO	NÃO
CPF	STRING	NÃO	SIM	NÃO
RG	STRING	NÃO	NÃO	NÃO
ORG_EXP	STRING	NÃO	NÃO	NÃO
UF_EXP	STRING	NÃO	NÃO	NÃO
NUMERO_CNH	STRING	NÃO	NÃO	NÃO
VENCIMENTO_CNH	STRING	NÃO	NÃO	NÃO
DATA_NASCIMENTO	STRING	NÃO	NÃO	NÃO
ENDEREÇO	STRING	NÃO	NÃO	NÃO
BAIRRO	STRING	NÃO	NÃO	NÃO
MUNICIPIO	STRING	NÃO	NÃO	NÃO
UF_EXP	STRING	NÃO	NÃO	NÃO
COMPLEMENTO	STRING	NÃO	NÃO	NÃO
NUMERO	STRING	NÃO	NÃO	NÃO
EMAIL	STRING	NÃO	NÃO	NÃO
TELEFONE	STRING	NÃO	NÃO	NÃO
CELULAR	STRING	NÃO	NÃO	NÃO

Quadro 9 – Cadastrar Cliente

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
CPF_PESSOA	STRING	NÃO	SIM	SIM
VALOR_DIVIDA	DECIMAL	NÃO	SIM	SIM

Quadro 10 – Lançamento Financeiro

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
CODIGO	INT	NÃO	SIM	NÃO
PLACA	STRING	NÃO	NÃO	SIM
CPF	STRING	NÃO	NÃO	SIM
DATA_MULTA	DATE	NÃO	NÃO	NÃO
VALOR_MULTA	DECIMAL	NÃO	NÃO	NÃO

Quadro 11 – Lançamento Multas

CAMPO	TIPO	NULO	CHAVE PRIMÁRIA	CHAVE ESTRANGEIRA
CODIGO	INT	NÃO	SIM	NÃO
PLACA	STRING	NÃO	NÃO	SIM
CPF	STRING	NÃO	NÃO	SIM
MOTORISTA	STRING	NÃO	NÃO	NÃO
DATA_INICIAL	DATE	NÃO	NÃO	NÃO
DATA_FINAL	DATE	NÃO	NÃO	NÃO
KM_INICIAL	DECIMAL	NÃO	NÃO	NÃO
KM_FINAL	DECIMAL	NÃO	NÃO	NÃO
PAGO	BOOLEAN	NÃO	NÃO	NÃO
SEGURO	STRING	NÃO	NÃO	NÃO

Quadro 12 – Locação Carro/Viagem

8 - Desenvolvimento C# CRUD de Interface

8.1 - Script C# do Sistema.

Conforme Artigo do publicado pelo Felipe no DEVMEDIA entende -se que se utiliza-se Classes DAO(Data Access Object), para realizar troca de dados entre operações de CRUD e o SGBD(Sistema de Gerenciamento de Banco de Dados). A necessidade de utilizar classes dessa forma é junto com a necessidade de separar logica de negócios com logica de persistência de banco de dados.

Imagem 17 – Script Classe DAO.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
using MySql.Data;
using System.Data;

namespace PIM1
{
    public class DAO
    {
        string conectaBanco = "DATABASE=pim; SERVER=127.0.0.1; UID=root;
        PWD=@morcego1";

        protected MySqlConnection conexao = null;

        public void AbrirConexao()
```

```

    {
        try
        {
            conexao = new MySqlConnection(conectaBanco);
            conexao.Open();
        }
        catch (Exception erro)
        {
            throw erro;
        }
    }

    public void FecharConexao()
    {
        try
        {
            conexao = new MySqlConnection(conectaBanco);
            conexao.Close();
        }
        catch (Exception erro)
        {
            throw erro;
        }
    }
}

```

Fonte: Autores, 2019.

Conforme Imagem abaixo A Classe ClienteDAO é responsável pela persistência no Banco de Dados, utilizando uma mescla de linguagem C# com comando de SQL para persistir, salvando, excluindo, buscando e alterando informações no banco.

Imagem 18 – Script Classe ClienteDAO.

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
using PIM1;
using System.Data;

namespace PIM1
{
    public class ClienteDao : DAO
    {

        MySqlCommand comando = null;

        //METODO DE SALVAR OS ARQUIVOS
    }
}

```

```

public void Salvar(Cliente cliente)
{
    try
    {
        AbrirConexao();

        comando = new MySqlCommand("INSERT INTO CDCLIENTE (NOME, CPF, RG,
        ORG_EXP, UF_EXP, CNH, VENCIMENTO_CNH, NASCIMENTO_DATA, ENDERECO, MUNICIPIO, UF,
        COMPLEMENTO, BAIRRO, NUMERO_ENDERECO, EMAIL, TELEFONE, CELULAR) VALUES (@NOME,
        @CPF, @RG, @ORG_EXP, @UF_EXP, @CNH, @VENCIMENTO_CNH, @NASCIMENTO_DATA, @ENDERECO,
        @MUNICIPIO, @UF, @COMPLEMENTO, @BAIRRO, @NUMERO_ENDERECO, @EMAIL, @TELEFONE,
        @CELULAR)", conexao);

        comando.Parameters.AddWithValue("@nome", cliente.nome);
        comando.Parameters.AddWithValue("@cpf", cliente.CPF);
        comando.Parameters.AddWithValue("@rg", cliente.rg);
        comando.Parameters.AddWithValue("@org_exp", cliente.org_exp);
        comando.Parameters.AddWithValue("@uf_exp", cliente.uf_exp);
        comando.Parameters.AddWithValue("@cnh", cliente.cnh);
        comando.Parameters.AddWithValue("@VENCIMENTO_CNH",
cliente.vencimento_cnh);
        comando.Parameters.AddWithValue("@NASCIMENTO_DATA",
cliente.nascimento_data);
        comando.Parameters.AddWithValue("@ENDERECO", cliente.endereco);
        comando.Parameters.AddWithValue("@MUNICIPIO", cliente.municipio);
        comando.Parameters.AddWithValue("@UF", cliente.uf);
        comando.Parameters.AddWithValue("@COMPLEMENTO", cliente.complemento);
        comando.Parameters.AddWithValue("@BAIRRO", cliente.bairro);
        comando.Parameters.AddWithValue("@NUMERO_ENDERECO",
cliente.numero_endereco);
        comando.Parameters.AddWithValue("@EMAIL", cliente.email);
        comando.Parameters.AddWithValue("@TELEFONE", cliente.telefone);
        comando.Parameters.AddWithValue("@CELULAR", cliente.celular);

        comando.ExecuteNonQuery();

    }
    catch (Exception erro)
    {
        throw erro;
    }
    finally
    {
        FecharConexao();
    }
}

//METODO DE LISTAGEM PARA ALTERAÇÃO E APARECER NO DATAGRIDVIEW

public DataTable listar()
{
    try
    {
        AbrirConexao();
        DataTable dt = new DataTable();
        MySqlDataAdapter da = new MySqlDataAdapter();
    }
}

```

```

        comando = new MySqlCommand("SELECT * FROM CDCLIENTE order by nome",
conexao);

        da.SelectCommand = comando;

        da.Fill(dt);

        return dt;
    }
    catch(Exception erro)
    {
        throw erro;
    }
    finally
    {
        FecharConexao();
    }
}

//METODO DE EDITAR

public void editar(Cliente cliente)
{
    try
    {
        AbrirConexao();

        comando = new MySqlCommand("UPDATE CDCLIENTE SET NOME = @NOME, CPF =
@CPF, RG = @RG, ORG_EXP = @ORG_EXP, UF_EXP = @UF_EXP, CNH = @CNH, VENCIMENTO_CNH =
@VENCIMENTO_CNH, NASCIMENTO_DATA = @NASCIMENTO_DATA, ENDERECO = @ENDERECO,
MUNICIPIO = @MUNICIPIO, UF = @UF, COMPLEMENTO = @COMPLEMENTO, BAIRRO = @BAIRRO,
NUMERO_ENDERECO = @NUMERO_ENDERECO, EMAIL = @EMAIL, TELEFONE = @TELEFONE, CELULAR
= @CELULAR WHERE CPF = @CPF", conexao);

        comando.Parameters.AddWithValue("@nome", cliente.nome);
        comando.Parameters.AddWithValue("@cpf", cliente.CPF);
        comando.Parameters.AddWithValue("@rg", cliente.rg);
        comando.Parameters.AddWithValue("@org_exp", cliente.org_exp);
        comando.Parameters.AddWithValue("@uf_exp", cliente.uf_exp);
        comando.Parameters.AddWithValue("@cnh", cliente.cnh);
        comando.Parameters.AddWithValue("@VENCIMENTO_CNH",
cliente.vencimento_cnh);
        comando.Parameters.AddWithValue("@NASCIMENTO_DATA",
cliente.nascimento_data);
        comando.Parameters.AddWithValue("@ENDERECO", cliente.endereco);
        comando.Parameters.AddWithValue("@MUNICIPIO", cliente.municipio);
        comando.Parameters.AddWithValue("@UF", cliente.uf);
        comando.Parameters.AddWithValue("@COMPLEMENTO", cliente.complemento);
        comando.Parameters.AddWithValue("@BAIRRO", cliente.bairro);
        comando.Parameters.AddWithValue("@NUMERO_ENDERECO",
cliente.numero_endereco);
        comando.Parameters.AddWithValue("@EMAIL", cliente.email);
        comando.Parameters.AddWithValue("@TELEFONE", cliente.telefone);
        comando.Parameters.AddWithValue("@CELULAR", cliente.celular);

        comando.ExecuteNonQuery();
    }
}

```

```

        catch(Exception erro)
        {
            throw erro;
        }
        finally
        {
            FecharConexao();
        }
    }
    //METODO DE EXCLUIR

    public void excluir(Cliente cliente)
    {
        try
        {
            AbrirConexao();

            comando = new MySqlCommand("DELETE FROM CDCLIENTE WHERE CPF=@CPF",
conexao);
            comando.Parameters.AddWithValue("@CPF", cliente.CPF);

            comando.ExecuteNonQuery();

        }
        catch (Exception erro)
        {
            throw erro;
        }
        finally
        {
            FecharConexao();
        }
    }
}
}

```

Fonte: Autores, 2019.

Conforme artigo sobre arquitetura de sistema publicado por Ferreira, Gabs a imagem a seguir utilizada para regras de negócios e controle de exceções ao chamar métodos do CRUD.

Imagem 19 – Script Classe ClienteBLL.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using PIM1;
using System.Data;

namespace PIM1
{
    public class ClienteBLL
    {
        ClienteDao clientedao = new ClienteDao();

        // salvar
        public void salvar(Cliente cliente)
        {
            try
            {
                clientedao.Salvar(cliente);
            }
            catch (Exception erro)
            {
                throw erro;
            }
        }

        // metodo para listar
        public DataTable listar()
        {
            try
            {
                DataTable dt = new DataTable();
                dt = clientedao.listar();
                return dt;
            }
            catch (Exception erro)
            {
                throw erro;
            }
        }

        // editando os dados
        public void editar(Cliente cliente)
        {
            try
            {
                clientedao.editar(cliente);
            }
            catch (Exception erro)
            {
                throw erro;
            }
        }
    }
}
```

```

//excluindo cliente
public void excluir(Cliente cliente)
{
    try
    {
        clientedao.excluir(cliente);
    } catch (Exception erro)
    {
        throw erro;
    }
}
}
}

```

Fonte: Autores, 2019.

Imagem a seguir trata da tela do formulario onde atraves de eventos é chamado os metodos do sistema,

Imagem 20 – Script Classe Fom1 – Formulario Cadastro Cliente.

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using PIM1;

namespace PIM1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
            //chamando o método listar para preencher o datagridview
            listar();
        }

        // metodo de salvar de cliente

        private void salvar(Cliente cliente)
        {
            ClienteBLL clienteBLL = new ClienteBLL();
            cliente.nome = txbnome.Text;
            cliente.CPF = txbcpf.Text;
            cliente.rg = txbrg.Text;
            cliente.org_exp = txb0e.Text;
            cliente.uf_exp = txbUfexp.Text;
            cliente.cnh = txbcnh.Text;
            cliente.vencimento_cnh = txbVcnh.Text;
        }
    }
}

```

```

        cliente.nascimento_data = txbData.Text;
        cliente.endereco = txbendereco.Text;
        cliente.municipio = txbmunicipio.Text;
        cliente.uf = txbuf.Text;
        cliente.complemento = txbcomplemento.Text;
        cliente.bairro = txbbairro.Text;
        cliente.numero_endereco = txbnumero.Text;
        cliente.email = txbemail.Text;
        cliente.telefone = txbtelefone.Text;
        cliente.celular = txbcelular.Text;

        clientebll.salvar(cliente);

        MessageBox.Show("Cliente salvo com sucesso");
        listar();
    }

    // metodo de listagem de cliente

    private void listar()
    {
        ClienteBLL clientebll = new ClienteBLL();
        bdview.DataSource = clientebll.listar();
    }

    //chamando o método editar

    private void editar(Cliente cliente)
    {
        ClienteBLL clientebll = new ClienteBLL();
        cliente.nome = txbnome.Text;
        cliente.CPF = txbcpf.Text;
        cliente.rg = txbrg.Text;
        cliente.org_exp = txbOe.Text;
        cliente.uf_exp = txbUfexp.Text;
        cliente.cnh = txbcnh.Text;
        cliente.vencimento_cnh = txbVcnh.Text;
        cliente.nascimento_data = txbData.Text;
        cliente.endereco = txbendereco.Text;
        cliente.municipio = txbmunicipio.Text;
        cliente.uf = txbuf.Text;
        cliente.complemento = txbcomplemento.Text;
        cliente.bairro = txbbairro.Text;
        cliente.numero_endereco = txbnumero.Text;
        cliente.email = txbemail.Text;
        cliente.telefone = txbtelefone.Text;
        cliente.celular = txbcelular.Text;

        clientebll.editar(cliente);

        MessageBox.Show("Cliente editado com sucesso!");

        listar();
    }

    //Metodo de excluir cliente

```

```

        private void excluir(Cliente cliente)
        {
            ClienteBLL clienteBLL = new ClienteBLL();
            cliente.CPF = txbcpf.Text;

            clienteBLL.excluir(cliente);

            MessageBox.Show("Cliente removido do Banco com sucesso!");

            listar();
        }

```

//método abaixo faz validação na hora que clica no radiobutton Pessoa física e pessoa jurídica

```

private void radioButton1_CheckedChanged(object sender, EventArgs e)
{
    txbnome.Enabled = true;
    txbcpf.Enabled = true;
    txbuf.Enabled = true;
    txbemail.Enabled = true;
    txbmunicipio.Enabled = true;
    txbendereco.Enabled = true;
    txbbairro.Enabled = true;
    txbnumero.Enabled = true;
    txbcomplemento.Enabled = true;
    txbtelefone.Enabled = true;
    txbcelular.Enabled = true;
    txbcnh.Enabled = true;
    txbData.Enabled = true;
    txbOe.Enabled = true;
    txbrg.Enabled = true;
    txbUfexp.Enabled = true;
    txbVcnh.Enabled = true;
}

private void radioButton2_CheckedChanged(object sender, EventArgs e)
{
    txbnome.Enabled = true;
    txbcpf.Enabled = true;
    txbuf.Enabled = true;
    txbemail.Enabled = true;
    txbmunicipio.Enabled = true;
    txbendereco.Enabled = true;
    txbbairro.Enabled = true;
    txbnumero.Enabled = true;
    txbcomplemento.Enabled = true;
    txbtelefone.Enabled = true;
    txbcelular.Enabled = true;
    txbcnh.Enabled = false;
    txbData.Enabled = false;
    txbOe.Enabled = false;
    txbrg.Enabled = false;
    txbUfexp.Enabled = false;
    txbVcnh.Enabled = false;
}

```

//evento ativado quando clica no botao salvar ou solicita alteração, o sistema chama o método testa dados e verifica se pode salvar no banco

```
private void Button2_Click(object sender, EventArgs e)
{
    if (testaDados() == true)
    {
        Cliente cliente = new Cliente();
        salvar(cliente);
    }
    else
    {
        MessageBox.Show("Erro");
    }
}

public bool testaDados()
{
    if (String.IsNullOrEmpty(txbnome.Text) || txbnome.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo Nome", "Nome");
        return false;
    }
    if (String.IsNullOrEmpty(txbcnh.Text) || !txbcnh.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo CNH", "CNH");
        return false;
    }
    if (String.IsNullOrEmpty(txbData.Text))
    {
        MessageBox.Show("Erro no campo Data", "Data");
        return false;
    }
    if (String.IsNullOrEmpty(txbendereco.Text))
    {
        MessageBox.Show("Erro no campo Endereço", "Endereço");
        return false;
    }
    if (String.IsNullOrEmpty(txbmunicipio.Text) ||
txbmunicipio.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo Municipio", "Municipio");
        return false;
    }
    if (String.IsNullOrEmpty(txbtelefone.Text) ||
!txbtelefone.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo Telefone", "Telefone");
        return false;
    }
    if (String.IsNullOrEmpty(txbcpf.Text) || !txbcpf.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo CPF", "CPF");
        return false;
    }
    if (String.IsNullOrEmpty(txbVcnh.Text))
    {

```

```

        MessageBox.Show("Erro no campo Vencimento CNH", "Vencimento CNH");
        return false;
    }
    if (String.IsNullOrEmpty(txbnumero.Text) ||
!txbnumero.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo Número", "Número");
        return false;
    }
    if (String.IsNullOrEmpty(txbbairro.Text))
    {
        MessageBox.Show("Erro no campo Bairro", "Bairro");
        return false;
    }
    if (String.IsNullOrEmpty(txbuf.Text) || txbuf.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo UF", "UF");
        return false;
    }
    if (String.IsNullOrEmpty(txbcelular.Text) ||
!txbcelular.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo Celular", "Celular");
        return false;
    }
    if (String.IsNullOrEmpty(txbrg.Text))
    {
        MessageBox.Show("Erro no campo RG", "RG");
        return false;
    }
    if (String.IsNullOrEmpty(txboe.Text) || txboe.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo Orgão Expeditor", "Orgão Expeditor");
        return false;
    }
    if (String.IsNullOrEmpty(txbUfexp.Text) ||
txbUfexp.Text.All(char.IsNumber))
    {
        MessageBox.Show("Erro no campo UF Expeditor", "UF Expeditor");
        return false;
    }
    if (String.IsNullOrEmpty(txbcomplemento.Text))
    {
        MessageBox.Show("Erro no campo Complemento", "Complemento");
        return false;
    }
    if (String.IsNullOrEmpty(txbemail.Text))
    {
        MessageBox.Show("Erro no campo E-mail", "E-mail");
        return false;
    }
    else
    {
        return true;
    }
}

```

//quando clicar na linha do datagrid view ele carrega as informações nos campos para poder alterar

```

private void Bdview_CellDoubleClick(object sender, DataGridViewCellEventArgs
e)
{
    txbnome.Text = bdview.CurrentRow.Cells[0].Value.ToString();
    txbcpf.Text = bdview.CurrentRow.Cells[1].Value.ToString();
    txbrg.Text = bdview.CurrentRow.Cells[2].Value.ToString();
    txboe.Text = bdview.CurrentRow.Cells[3].Value.ToString();
    txbUfexp.Text = bdview.CurrentRow.Cells[4].Value.ToString();
    txbcnh.Text = bdview.CurrentRow.Cells[5].Value.ToString();
    txbVcnh.Text = bdview.CurrentRow.Cells[6].Value.ToString();
    txbData.Text = bdview.CurrentRow.Cells[7].Value.ToString();
    txbendereco.Text = bdview.CurrentRow.Cells[8].Value.ToString();
    txbmunicipio.Text = bdview.CurrentRow.Cells[9].Value.ToString();
    txbuf.Text = bdview.CurrentRow.Cells[10].Value.ToString();
    txbcomplemento.Text = bdview.CurrentRow.Cells[11].Value.ToString();
    txbbairro.Text = bdview.CurrentRow.Cells[12].Value.ToString();
    txbnumero.Text = bdview.CurrentRow.Cells[13].Value.ToString();
    txbemail.Text = bdview.CurrentRow.Cells[14].Value.ToString();
    txbtelefone.Text = bdview.CurrentRow.Cells[15].Value.ToString();
    txbcelular.Text = bdview.CurrentRow.Cells[16].Value.ToString();
}

private void Button3_Click(object sender, EventArgs e)
{
    if (testaDados() == true)
    {
        Cliente cliente = new Cliente();
        editar(cliente);
    }
    else
    {
        MessageBox.Show("Erro");
    }
}

private void Button4_Click(object sender, EventArgs e)
{
    Cliente cliente = new Cliente();
    excluir(cliente);
}

//ao clicar em cancelar o Cadastro o Sistema limpa os dados

private void Button1_Click(object sender, EventArgs e)
{
    txbbairro.Text = "";
    txbcelular.Text = "";
    txbcnh.Text = "";
    txbcomplemento.Text = "";
    txbcpf.Text = "";
    txbData.Text = "";
    txbemail.Text = "";
    txbendereco.Text = "";
    txbmunicipio.Text = "";
    txbnome.Text = "";
    txbnumero.Text = "";
}

```

```

        txb0e.Text = "";
        txbrg.Text = "";
        txbtelefone.Text = "";
        txbuf.Text = "";
        txbUfexp.Text = "";
        txbVcnh.Text = "";

    }
}
}

```

Fonte: Autores, 2019.

Imagem a seguir trata-se da tela de menu, onde temos os botões para chamar as outras interfaces.

Imagem 21 – Script Classe frmInicial.

```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PIM1
{
    public partial class frmInicial : Form
    {
        public frmInicial()
        {
            InitializeComponent();
        }

        //metodos abaixo chama cada tela do Sistema

        private void btnClientes_Click(object sender, EventArgs e)
        {
            //this.Hide();
            Form1 cliente = new Form1();
            cliente.Show();
        }

        private void BtnVeiculos_Click(object sender, EventArgs e)
        {
            //this.Hide();
            frmCadVeiculos veiculos = new frmCadVeiculos();
            veiculos.Show();
        }

        private void BrnFinancas_Click(object sender, EventArgs e)
        {
            //this.Hide();
            frmFinancas financas = new frmFinancas();
            financas.Show();
        }
    }
}

```



```

    }

    private void BtnEstoque_Click(object sender, EventArgs e)
    {
        //this.Hide();
        Entrada eEntrada = new Entrada();
        eEntrada.Show();
    }

    private void BtnGaragem_Click(object sender, EventArgs e)
    {
        //this.Hide();
        frmGaragem garagem = new frmGaragem();
        garagem.Show();
    }

    private void BtnManutencoes_Click(object sender, EventArgs e)
    {
        //this.Hide();
        frmManutencoes manutencao = new frmManutencoes();
        manutencao.Show();
    }

    private void BtnMultas_Click(object sender, EventArgs e)
    {
        //this.Hide();
        Multas multas = new Multas();
        multas.Show();
    }

    private void BtnRelatorios_Click(object sender, EventArgs e)
    {
        //this.Hide();
        frmRelatorio relatorio = new frmRelatorio();
        relatorio.Show();
    }

    private void BtnViagens_Click(object sender, EventArgs e)
    {
        //this.Hide();
        frmViagens viagens = new frmViagens();
        viagens.Show();
    }

    private void BtnSaidaEstoque_Click(object sender, EventArgs e)
    {
        //this.Hide();
        Sidaa eSaida = new Sidaa();
        eSaida.Show();
    }
}
}

```

Fonte: Autores, 2019.

Proxima imagem é sobre a tela de login onde fazemos validações para acessar o sistema juntamente com o metodo para chamar a proxima interface caso esteja correto.

Imagem 22 – Script Classe frmLogin.

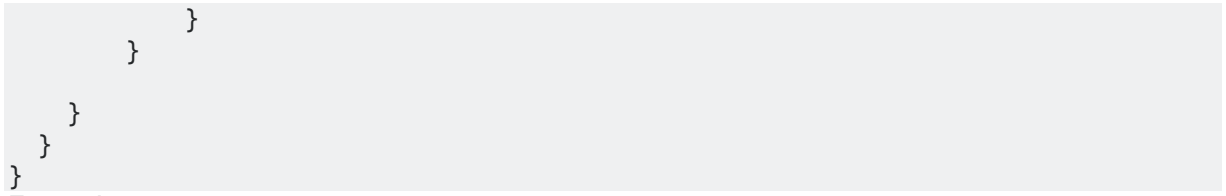
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace PIM1
{
    public partial class frmLogin : Form
    {
        public frmLogin()
        {
            InitializeComponent();
        }

        //bloco de codigo abaixo faz a validação dos textbox da tela de login para poder
        //realizar a entrada no sistema, fizemos a validação simples via IF e ELSE

        private void btnLogin_Click(object sender, EventArgs e)
        {
            if(txtUsuario.Text == "user" && txtSenha.Text == "pass")
            {
                MessageBox.Show("OK");

                this.Hide();
                frmInicial form = new frmInicial();
                form.Show();
            }
            else
            {
                if(txtUsuario.Text == "user" && txtSenha.Text != "pass")
                {
                    MessageBox.Show("Senha incorreta");
                    txtSenha.Text = "";
                    txtUsuario.Text = "";
                }
                else if(txtUsuario.Text != "user" && txtSenha.Text == "pass")
                {
                    MessageBox.Show("Usuário Incorreto");
                    txtUsuario.Text = "";
                    txtSenha.Text = "";
                }
                else
                {
                    MessageBox.Show("Usuário e senha incorretos");
                    txtUsuario.Text = "";
                    txtSenha.Text = "";
                }
            }
        }
    }
}
```



Fonte: Autores, 2019.

9 - MANUAIS DE UTILIZAÇÃO E INTEFACE DE TELAS

Imagem 23 – Tela Estoque saída.

Fonte: Autores, 2019.

A tela de cadastro de estoque adiciona a possibilidade de o usuário incrementar itens do veículo utilizados na manutenção do mesmo, possibilitando que a empresa possa ter um inventário de itens disponíveis.

Imagem 24 – Tela Estoque entrada.

Fonte: Autores, 2019.

Na tela de entrada, toda vez que uma nova peça ou acessório entrar para inventário, um novo cadastro deve ser realizado.

Código: Inserir código da peça;

Preço: Inserir o preço da peça;

Fornecedor: Inserir o nome do fornecedor;

Descrição: Inserir uma breve descrição do fornecedor;

Quantidade: Inserir a quantidade de peças;

Data de Entrada: Inserir a data de entrada da peça;

Utilizar a tela de saída quando a peça for retirada para ser utilizada em manutenção.

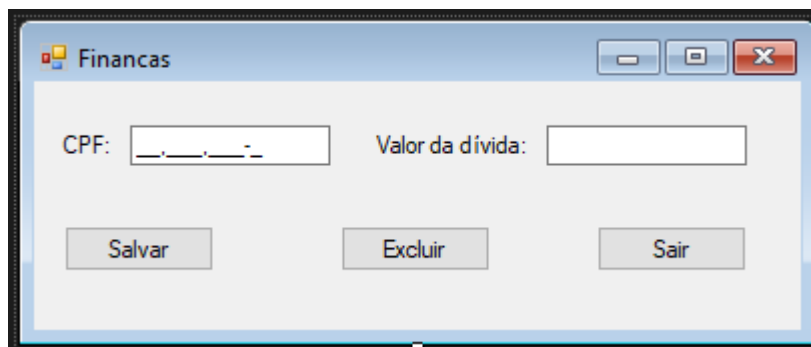
Código: Inserir o código da peça;

Quantidade: Inserir a quantidade retirada da peça;

Descrição: Inserir a descrição da peça;

Data de saída: Inserir a data de saída da peça;

Imagem 25 – Lançamento Financeiro de dividas.



Fonte: Autores, 2019.

A tela de finanças adiciona a possibilidade de adição de finanças para maior controle de dívidas do cliente na aplicação.

CPF: Inserir o CPF do cliente;

Valor da Dívida: Inserir o valor da dívida do Cliente;

Imagem 26 – Cadastro de Cliente

The screenshot shows a window titled 'Clientes' with a standard Windows interface. At the top, there are two radio buttons: 'Pessoa Física' (selected) and 'Pessoa Jurídica'. To the right of these are four icons with labels: 'Salvar' (save), 'Editar' (edit), 'Excluir' (delete), and 'Cancelar Cadastro' (cancel registration). Below this, the form contains several input fields arranged in rows: 'Nome:', 'CPF/CNPJ:', 'RG:', 'Nº Cnh:', 'Vnct Cnh:', 'Org. Exp:', 'Uf. Exp:', 'Data Nasc.:', 'Cep-Rua/Av:', 'Nº:', 'Bairro:', 'Compl.:', 'Município:', 'Uf:', 'Telefone:', 'Celular:', and 'E-Mail:'. At the bottom of the form is a large, empty rectangular area.

Fonte: Autores, 2019.

A tela de cadastro de Clientes adiciona a possibilidade de controle de clientes dentro da aplicação. É nesta tela que se deve ser cadastrado os dados referentes ao cliente, e informar se o veículo será locado a pessoa física ou jurídica.

Nome: Inserir o Nome do cliente ou empresa em caso de pessoa jurídica;

Nº CNH: Inserir número de CNH de motorista;

Data Nasc: Inserir a data de nascimento ou data de abertura de empresa;

Cep-rua/Av: Inserir o endereço com CEP, rua ou avenida de cliente ou empresa;

Município: Inserir município de cliente ou empresa;

Telefone: Inserir telefone de cliente ou empresa;

CPF/CNPJ: Inserir o CPF ou CNPJ do cliente ou empresa;

Vnct CNH: Inserir o vencimento da CNH;

Nº: Inserir o número de complemento/residência;

UF: Inserir o estado do cliente ou empresa;

Celular: Inserir celular do cliente ou empresa;

RG: Inserir o RG caso Pessoa Física seja selecionada;

Org. Exp: Inserir órgão expedidor do RG;

UF Exp: Inserir estado expedidor do RG;

Compl.: Inserir o complemento do cliente ou empresa;

E-mail: Inserir o e-mail do cliente ou empresa;

Imagem 27 – Cadastro de Veiculos.

A imagem mostra uma interface de usuário para o cadastro de veículos. A janela tem o título "Cadastro de Veiculos". Os campos de entrada são:

- Marca: [campo de texto]
- Modelo: [campo de texto]
- Ano: [campo de texto]
- Placa: [campo de texto]
- Cor: [campo de texto]
- Renavam: [campo de texto]
- Motor: [menu suspenso]
- UF: [campo de texto]

Existem também opções de rádio para "Veiculo é zero KM?" (Sim/Não), um campo para "KM Atual" e um checkbox para "Manutenção em dia". Um botão "Salvar" está na parte inferior.

Fonte: Autores, 2019.

A tela de cadastro de veículos adiciona ao software a possibilidade de controle de veículos para a empresa.

Marca: Inserir a marca do veículo;

Ano: Inserir o ano de fabricação do veículo;

Cor: Inserir a cor do veículo;

Modelo: Inserir o modelo do veículo;

Placa: Inserir a placa do veículo;

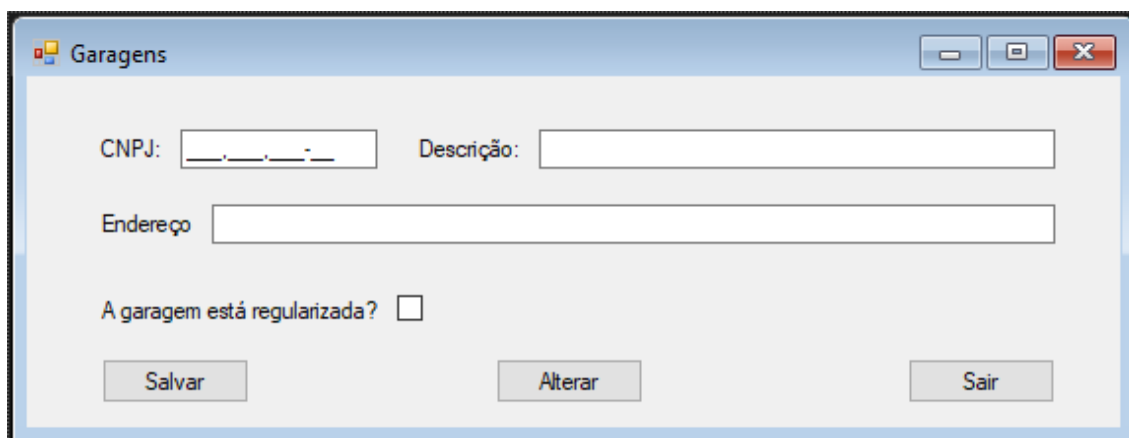
Renavam: Inserir o renavam do veículo;

Inserir dentro do campo "Veículo é zero km?" se o veículo cadastrando é um veículo zero km. Caso não, inserir a KM Atual e marcar se a manutenção está em dia;

Motor: Inserir a cilindrada do motor;

UF: Inserir o estado no qual o veículo foi registrado;

Imagem 28 – Cadastro de Garagem.



Fonte: Autores, 2019.

A tela de cadastro de garagens adiciona ao software a possibilidade de controle de garagens credenciadas.

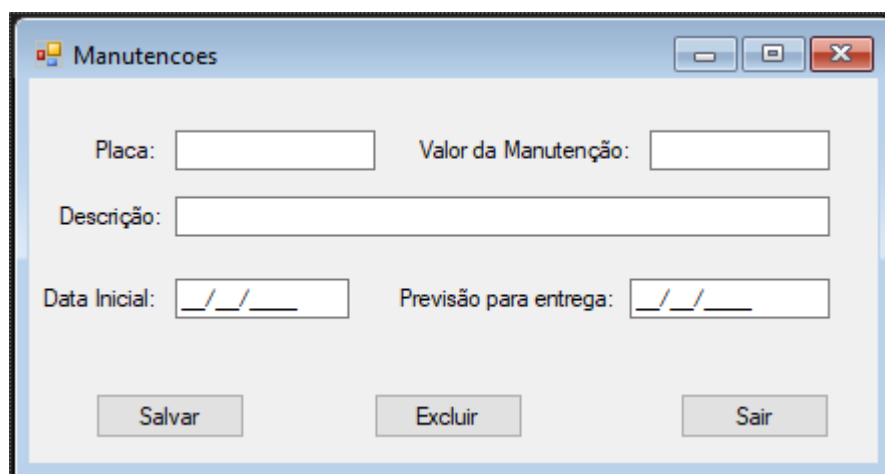
CNPJ: Inserir o CNPJ da garagem;

Descrição: Inserir a descrição da garagem;

Endereço: Inserir o endereço da garagem;

O campo abaixo deve ser checado somente se a garagem obedecer as normas de regularização impostos pelo regulamento da empresa;

Imagem 29 – Lançamento de Manutenções.



Fonte: Autores, 2019.

A tela de manutenções adiciona ao programa a possibilidade de controle de manutenções do veículo para a empresa;

Placa: Inserir a placa do veículo;

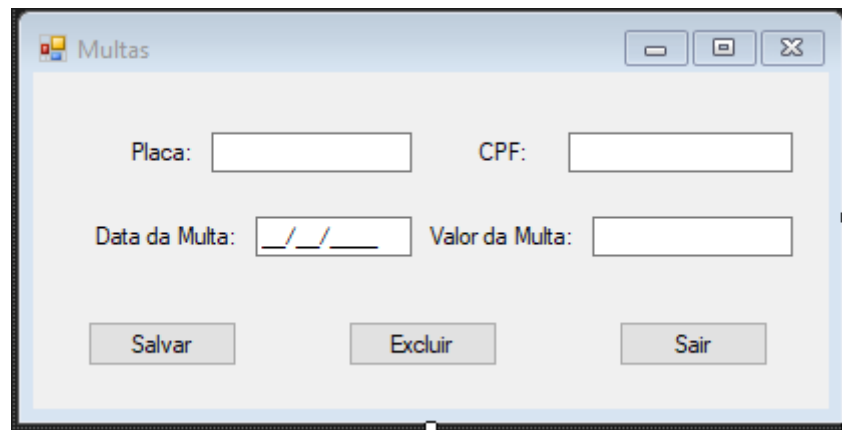
Valor da manutenção: Inserir o valor da manutenção realizada;

Descrição: Inserir a descrição da manutenção. Este campo não deve ser preenchido com o nome do veículo;

Data Inicial: Inserir a data de entrada de veículo na manutenção;

Previsão para entrega: Inserir data prevista para entrega do veículo;

Imagem 30 – Lançamento de Multas para Frota e Cliente.



Fonte: Autores, 2019.

A tela de multas adiciona ao software o controle de multas recebidas pelo condutor.

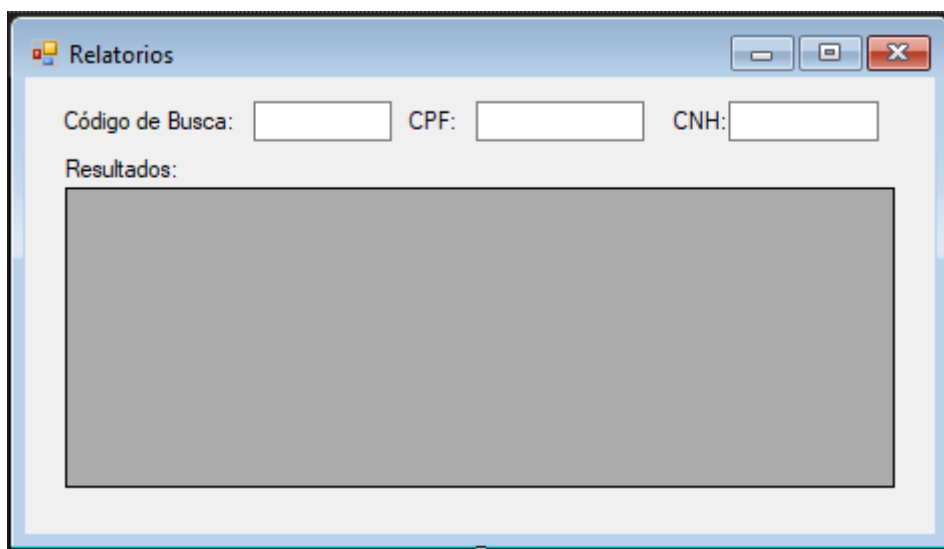
Placa: Inserir a placa do veículo;

CPF: Inserir o CPF do cliente;

Data da multa: Inserir a data que a multa foi aplicada;

Valor da multa: Inserir o valor da Multa;

Imagem 31 – Consulta Relatórios.



The screenshot shows a window titled "Relatorios" with a standard Windows-style title bar. Inside the window, there are three input fields for searching: "Código de Busca:", "CPF:", and "CNH:". Below these fields is a label "Resultados:" followed by a large, empty rectangular box intended for displaying the search results.

Fonte: Autores, 2019.

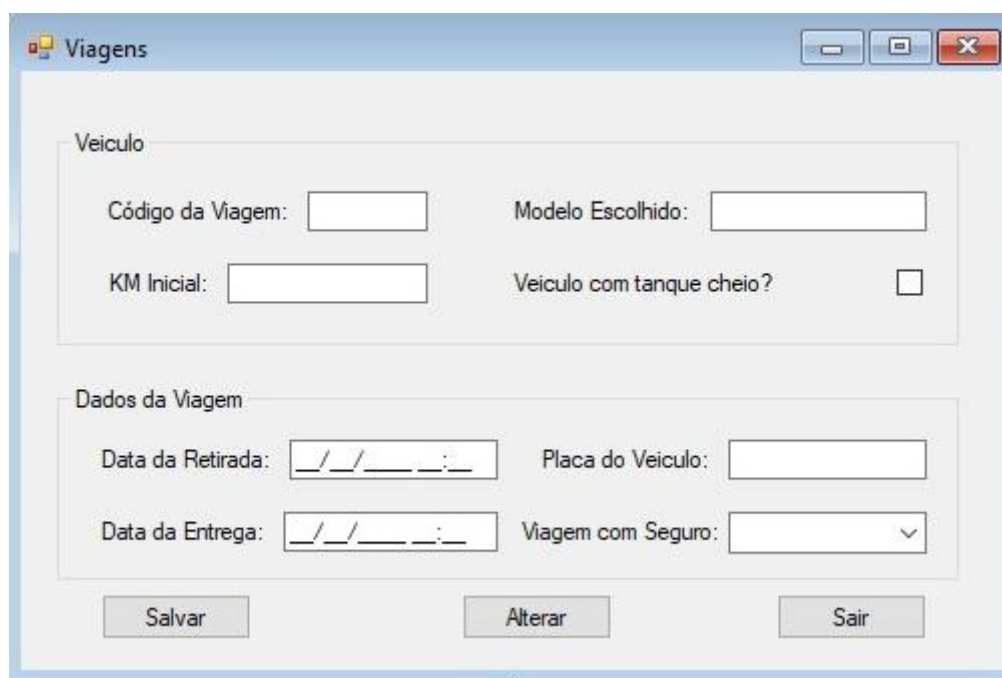
A tela de relatórios adiciona a possibilidade de pesquisa de relatórios para a empresa.

Código de Busca: Inserir caso deseje buscar por ID;

CPF: Inserir se desejar buscar por CPF;

CNH: Inserir caso deseje buscar por CNH;

Imagem 32 – Lançamento de Viagens



The screenshot shows a window titled "Viagens" with a standard Windows-style title bar. The window is divided into two main sections. The first section, labeled "Veiculo", contains four fields: "Código da Viagem:", "Modelo Escolhido:", "KM Inicial:", and "Veiculo com tanque cheio?" (which is a checkbox). The second section, labeled "Dados da Viagem", contains four fields: "Data da Retirada:" (with a date-time picker), "Placa do Veiculo:", "Data da Entrega:" (with a date-time picker), and "Viagem com Seguro:" (which is a dropdown menu). At the bottom of the window, there are three buttons: "Salvar", "Alterar", and "Sair".

Fonte: Autores, 2019.

A tela de Viagens adiciona ao software o controle de viagens feitas pelo cliente.

Código da Viagem: Inserir o código de viagem;

Modelo escolhido: Inserir o modelo de veículo escolhido pelo cliente;

KM Inicial: Inserir o KM inicial do Veículo (Momento de saída);

Checar o campo “**Veículo com tanque cheio**” somente se o veículo estiver com tanque cheio no momento de saída;

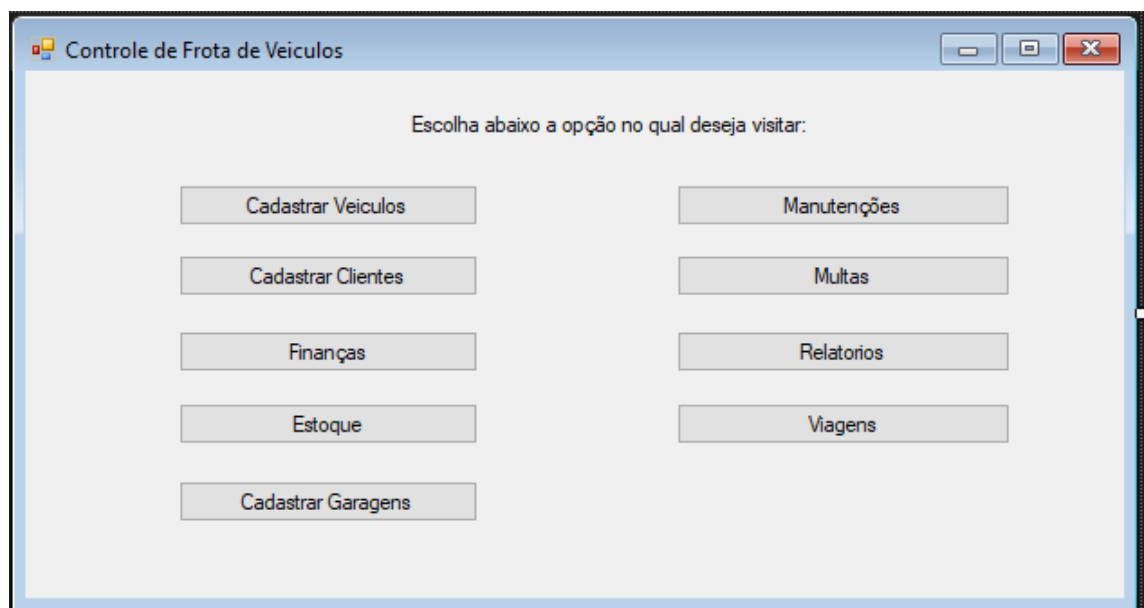
Data da Retirada: Inserir a data de retirada do veículo;

Placa do veículo: Inserir a placa do veículo;

Data da Entrega: Preencher a data de entrega prevista;

Checar o campo “**Veículo com seguro**” somente se o veículo contratado contar com seguro automotivo;

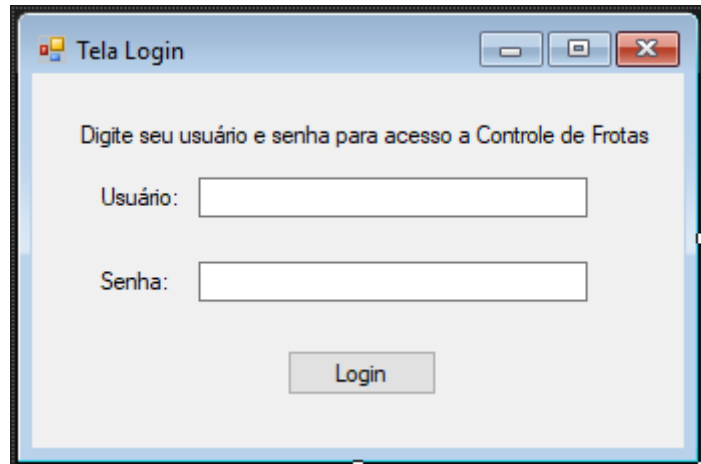
Imagem 33 – Menu do Sistema.



Fonte: Autores, 2019.

A tela inicial é a tela principal do software. É nela que o programa se inicia e que o usuário terá a opção de acesso a todos as features do programa.

Imagem 34 – Tela de Login do Sistema.



Fonte: Autores, 2019.

A tela de login possibilita maior segurança para a empresa com uma camada de segurança em aplicação. Somente usuários autorizados podem acessar o sistema;

Usuário: Digite o usuário;

Senha: Digite a senha;

Após o clique em “**Login**”, o sistema verificará se as informações possuem credenciais para login, caso positivo, sua aplicação iniciará.

10 - SOFTWARE

10.1 - Requisitos Básicos de Instalação

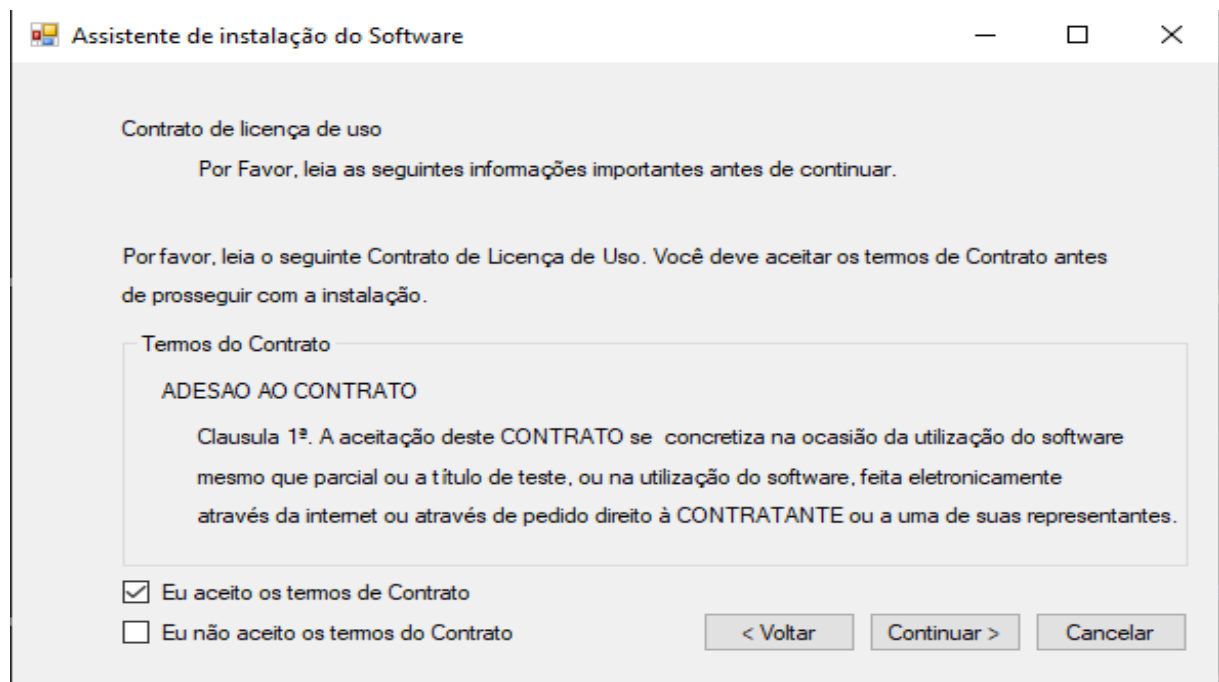
O sistema foi desenvolvido fundamentalmente utilizando a plataforma (Visual Studio e MYSQL).

Os pré-requisitos para o funcionamento são:

- Processador: Intel ou AMD multi core.
- Memoria: 4 GB ou superior.
- HD: 10 GB de espaço livre em disco.
- Sistema Operacional: Microsoft Windows 7 ou superior.
- Gráficos: Controlador Intel

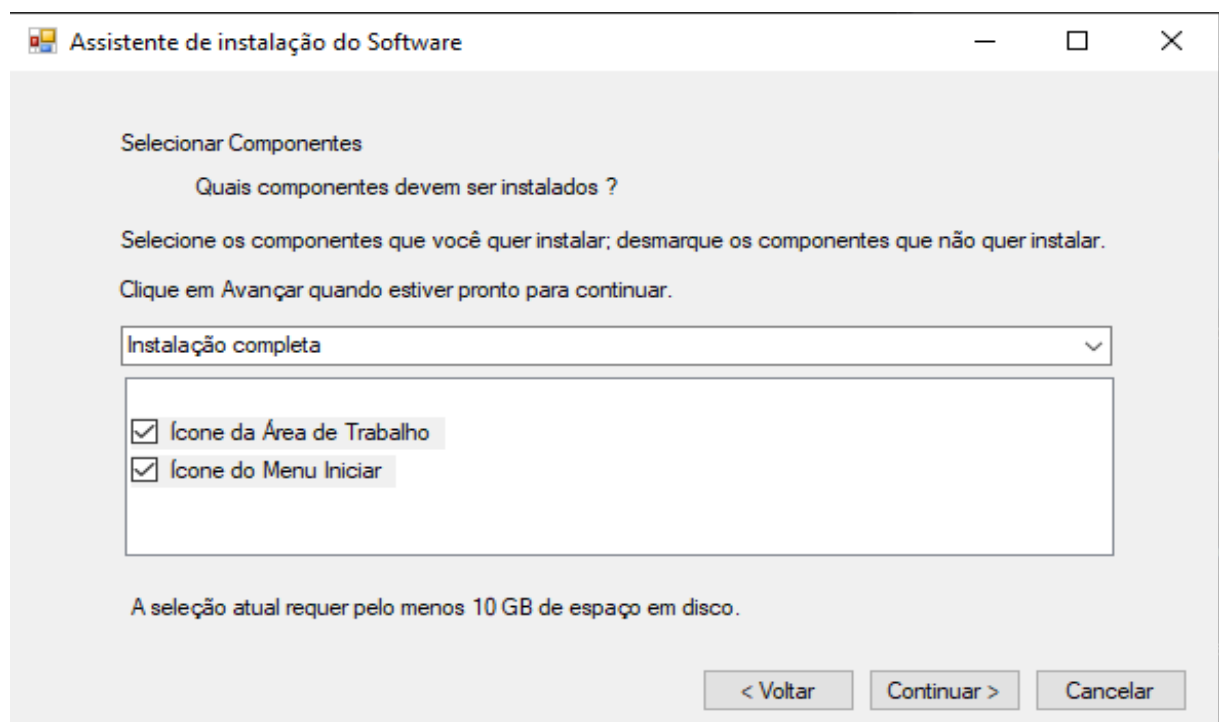
10.2 - Instalação do Software

Imagem 35 – Tela do Assistente de Instalação (1/4)



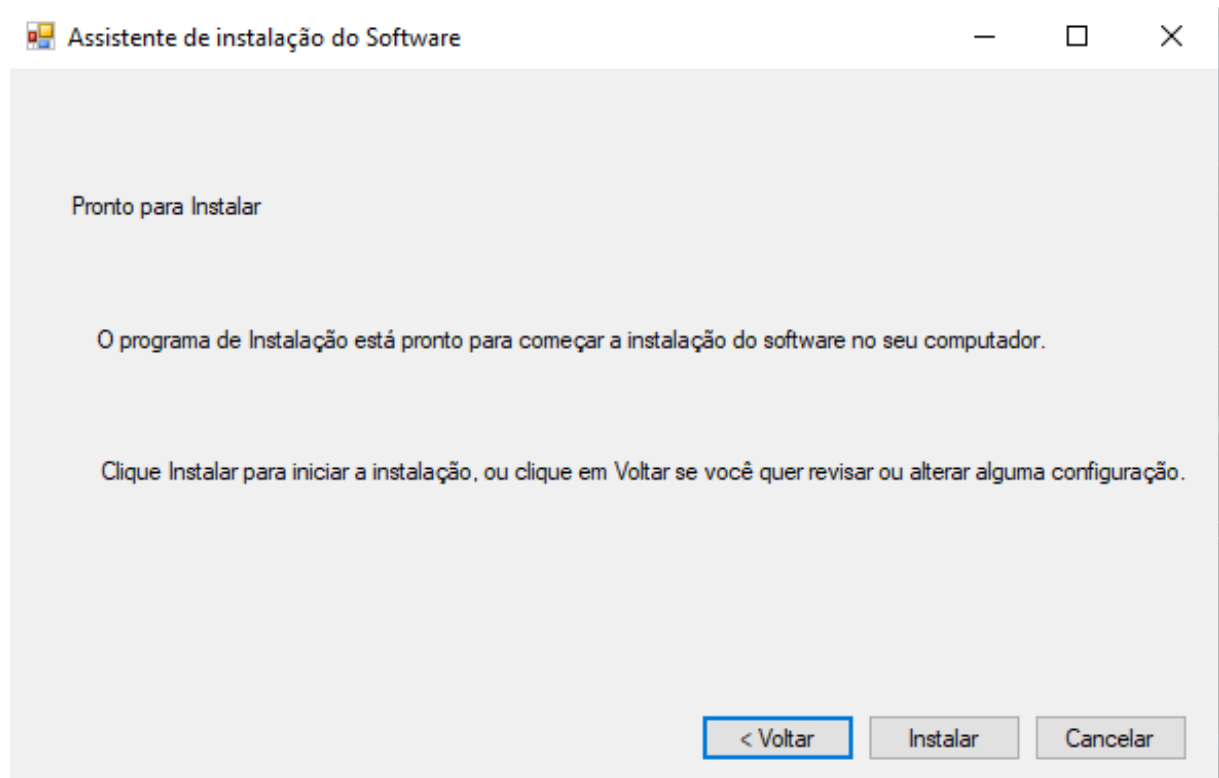
Fonte: Autores, 2019.

Imagem 36 – Tela do Assistente de Instalação (2/4)



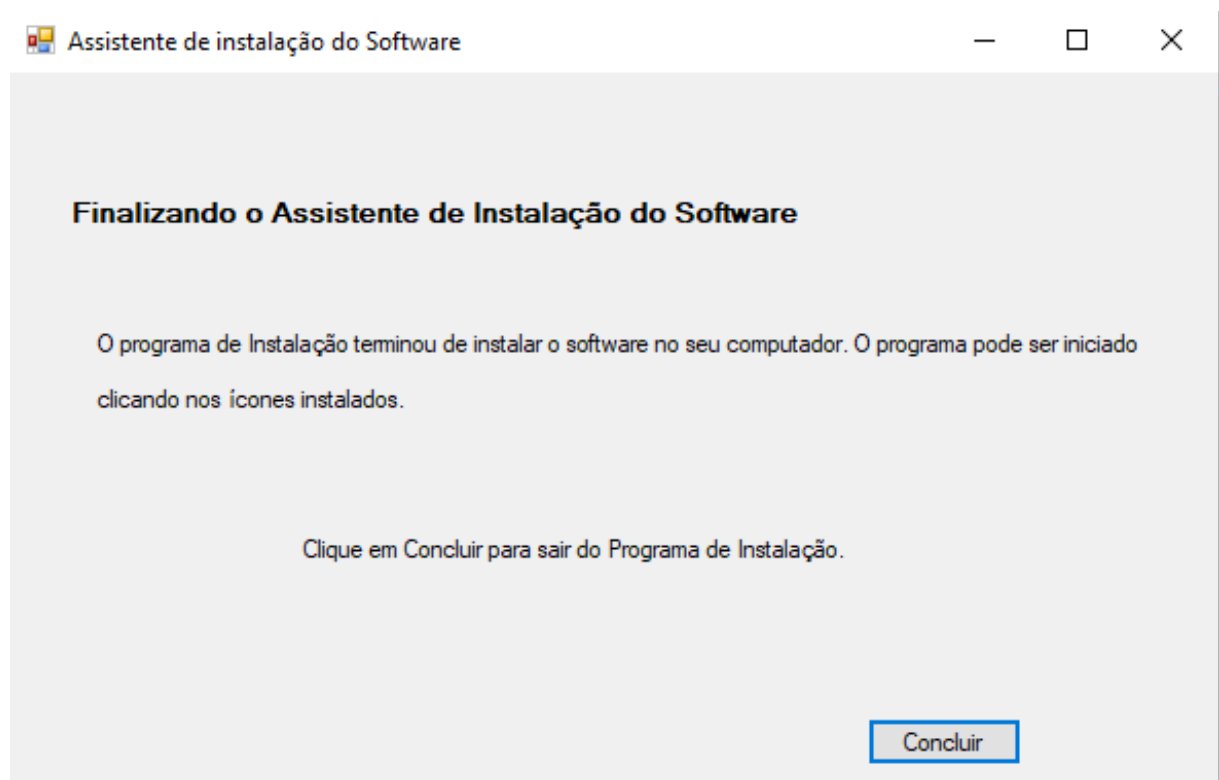
Fonte: Autores, 2019.

Imagem 37 – Tela do Assistente de Instalação (3/4)



Fonte: Autores, 2019.

Imagem 38 – Tela do Assistente de Instalação (4/4)



Fonte: Autores, 2019.

11 - CONTRATO DE SERVICO E MANUTENÇÃO DE SOFTWARE DZVOLVE

EMPRESA:.....

LOCATÁRIO:..... RAZÃO SOCIAL

RESPONSABILIDADE DA CONTRATADA:

1.0 Total orientação quanto à instalação e manutenção dos softwares por e-mail no endereço Dzvolve@gmail.com em 2 dias úteis no horário comercial;

2.0 Fornecimento gratuito das atualizações (upgrades) das versões dos softwares, ficando excluídos desta cláusula os novos programas desenvolvidos pela CONTRATADA;

3.0 Em caso de defeito do software a manutenção do(s) mesmo(s) será efetuada de forma gratuita, ficando excluídos os casos em que o defeito apresentado tiver sido gerado por uso incorreto ou inadequado do software ou problemas do computador do CONTRATANTE.

4.0 Entrega e atualização dos manuais para correta utilização do(s) software(s), será de responsabilidade do CONTRATANTE:

5.0 A realização de back-up de segurança do programa diariamente, sendo que a CONTRATADA não se responsabilizará pelos dados que forem perdidos caso tal orientação não seja seguida pelo CONTRATANTE;

CONTRATO DE MANUTENÇÃO VALOR MENSAL

1.0 O valor referente de R\$3.000(três mil reais) refere-se a o software completo.

2.0 O valor da mensalidade de serviços será pago no mês posterior à prestação do serviço;

3.0 Em caso de atraso de mais de 25 dias no pagamento referente a este contrato os serviços de manutenção e os descontos em produtos e serviços, sendo que após a regularização dos pagamentos em atraso os serviços serão reiniciados;

4.0 Este contrato tem duração de 12 (doze) meses e após este período poderá ser rescindido por qualquer uma das partes a qualquer tempo, desde que tal desistência seja efetivada por escrito à outra parte com o prazo mínimo de 30 dias, ficando o CONTRATANTE responsável pelo pagamento da última parcela entre a notificação da desistência e o término do contrato ;

EMPRESA:

CONTRATANTE:

TESTEMUNHAS:

12 – PROMOÇÕES

Relâmpago: Com esta promoção o cliente sempre ficará atento ao que você está “aprontando na sua loja”.

Ações na linha “nessa data escolhida pelo proprietário, alugue seu carro com 15% de desconto” também despertam no cliente o senso de urgência. Com este desconto se o mesmo deixa para alugar depois, não terá mais o desconto.

Utilizar a inteligência com as métricas do negócio para tornar a ação mais eficiente. Com isso saberemos o dia e a hora de maior movimento e realizar as devidas promoções.

Datas Especiais: Essas datas também são importantes sem dúvida alguma, podendo ir mais além oferecendo descontos no mesmo de aniversário da loja ou do cliente.

Ampliar a estratégia para datas que não comemoram nada. Por exemplo, por que não realizar algumas ofertas próximo ao final do ano? Ou seja, cliente sempre procuram viajar durante este período.

Segmentadas: Um exemplo de promoção segmentada seria o seguinte: a cada 5 automóveis alugados, o 6º sai por uma porcentagem escolhida pela locadora na próxima locação.

Convenio: Um exemplo de promoção convenio seria na seguinte ideologia: a locadora de veículos aluga o carro para empresa “X”, assim ganhando um desconto especial de acordo com o tamanho da frota da empresa.

REFERENCIA

SARQUIS, Carlos. RELATÓRIO DE ATIVIDADES DE 2017. Disponível em <<http://www.anav.org.br/noticias.php>>. Acesso em 24 de abril de 2019.

PLANILHA DE CONTROLE DE FROTA 4.0. Disponível em <<https://luz.vc/planilhas-empresariais/planilha-de-controle-de-frota>>. Acesso em 19 de maio de 2019.

Disponível em <<https://www.devmedia.com.br/forum/contrato-de-manutencao-de-software/136163>>. Acesso em 01 de junho de 2019.

Felipe, DAO Pattern: Persistência de Dados utilizando o padrão DAO. Disponível em <<https://www.devmedia.com.br/dao-pattern-persistencia-de-dados-utilizando-o-padrao-dao/30999>>. Acesso em 04 de junho de 2019.

FERREIRA, Gabs. Voltando no tempo: como eu aprendi a fazer CRUDs com DAL e BLL(ou: Como você não deve fazer CRUDS hoje em dia). Disponível em <<http://gabsferreira.com/voltando-no-tempo-como-eu-aprendi-a-fazer-crud-com-dal-e-bll/>>. Acesso em 04 de junho de 2019.

SCHOTT, MIRCHELL. Creating a Data Access Layer (VB) .Disponível em <<https://docs.microsoft.com/en-us/aspnet/web-forms/overview/data-access/introduction/creating-a-data-access-layer-vb>>. Acesso em 04 de junho de 2019.

SCHOTT, MIRCHELL. Criação de uma camada de lógica de negócios (C#). Disponível em <<https://docs.microsoft.com/pt-br/aspnet/web-forms/overview/data-access/introduction/creating-a-business-logic-layer-cs>>. Acesso em 04 de junho de 2019.