

**FACULDADE DE ENGENHARIA**

**DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA**

**LICENCIATURA EM ENGENHARIA INFORMÁTICA**

**[Engenharia de Software II]**

**IIIo ano, IIo semestre**

**Sistema de Gestão de Frota de   
Camiões**

**Discentes:**

* Gulele, Dércio
* Jeque, Osvaldo
* Manjate, Andrade Machequele
* Rodrigues, Núria
* Guiruite, Abel Jorge
* Fazir, Wiliam
* Mazembe, Dércio

**Corpo Docente:**

* dr. Vali Issufo
* dr. Bavikha Rugnath
* Monitor: Richaldo Elias

**Maputo Novembro de 2018**



**FACULDADE DE ENGENHARIA**

**DEPARTAMENTO DE ENGENHARIA ELECTROTÉCNICA**

**LICENCIATURA EM ENGENHARIA INFORMÁTICA**

**[Engenharia de Software II]**

**IIIo ano, IIo semestre**

**Sistema de Gestão de Frota de   
Camiões**

**Índice**

[1. Prefácio 2](#_Toc515576556)

[2. Introdução 3](#_Toc515576557)

[3. Glossário 4](#_Toc515576558)

[4. Requisitos 5](#_Toc515576559)

[4.1. Requisitos de utilizadores 5](#_Toc515576560)

[4.2. Requisitos Funcionais (Casos de usos) 5](#_Toc515576561)

[4.3. Requisitos Não Funcionais 5](#_Toc515576562)

[5. Modelos do sistema 6](#_Toc515576563)

[5.1. Diagramas de Classes 6](#_Toc515576564)

[5.2. Diagrama de Sequencia 7](#_Toc515576565)

[5.3. Diagrama de Estados 7](#_Toc515576566)

[5.3.1. Diagrama de Actividades 7](#_Toc515576567)

[6. Interface de Utilizador (Protótipos) 7](#_Toc515576568)

[7. Testes 7](#_Toc515576569)

**Índice de Diagramas**

[Figure 1: Diagrama de casos de uso 10](#_Toc515916696)

[Figure 2: Página de login 11](#_Toc515916697)

[Figure 3: Página inicial 12](#_Toc515916698)

[Figure 4: Registo de viagem 12](#_Toc515916699)

[Figure 5: Diagrama de classes 14](#_Toc515916700)

[Figure 6: 6DC: Viagem 15](#_Toc515916701)

[Figure 7: DC: Carga 1 16](#_Toc515916702)

[Figure 8: DC: Gestor 1 17](#_Toc515916703)

[Figure 9: Manutenção 18](#_Toc515916704)

[Figure 10: Cliente 19](#_Toc515916705)

[Figure 11: Camião 20](#_Toc515916706)

[Figure 12: Motorista 21](#_Toc515916707)

[Figure 13: Funcionário 21](#_Toc515916708)

[Figure 14: Supervisor 22](#_Toc515916709)

[Figure 15: Diagrama de sequência criar viagem 23](#_Toc515916710)

[Figure 16: Registar 23](#_Toc515916711)

[Figure 17: Legenda 24](#_Toc515916712)

[Figure 18:Legenda 24](#_Toc515916713)

[Figure 19: Legenda 24](#_Toc515916714)

[Figure 20: Legenda 25](#_Toc515916715)

[Figure 21: Diagrama de estados do camião 25](#_Toc515916716)

[Figure 22: Alocação do motorista 26](#_Toc515916717)

[Figure 23: Viagem 26](#_Toc515916718)

**Índice de tabelas**

[Table 1: caso de teste - confirmar entrega 29](#_Toc515916746)

[Table 2: caso de teste - registar viagem 30](#_Toc515916747)

[Table 3: Verificar histórico 30](#_Toc515916748)

[Table 4: Verificar solicitação 30](#_Toc515916749)

**Índice de Requisitos**

[Requisito de Utilizador 1 2](file:///C:\Users\richa\OneDrive\Desktop\%5bTemplete%5d%20Documento%20de%20Requisitos.docx#_Toc515574884)

1. Prefácio

[. Deve definir os possíveis leitores do documento e descrever seu histórico de versões, incluindo uma justificativa para a criação de uma nova versão e um resumo de mudanças feitas em cada versão]

1. Introdução

O propósito do sistema desenvolvido e dinamizar e optimizar (armazenando dados que a posterior poderão ser usados para diferentes tomadas de decisão) por meios informáticos o modos vivendo, forma de trabalho, a relação custo-benefício entre o cliente e o prestador de serviços e fundamentalmente a questão tempo que e um recurso imprescindível nos negócios. O sistema terá algumas plataformas integradas como um Web Site.

1. Glossário

A correta interpretação deste documento exige o conhecimento de algumas convenções e termos específicos, que são descritos a seguir:

DC- Diagrama de Classes;

HTML- *HyperText Markup Language,* ou Linguagem de Marcação de Hipertexto.

LP- Linguagem de Programação;

SO- Sistema Operacional;

SGBD- Sistema de Gestão de Base de Dados;

SGFC- Sistema de Gestão de Frota de Camiões;

PWSGC – Programação Web e Sistema de Gestão de Conteúdos;

PHP- *Hypertext Preprocessor;*

RF – Requisitos Funcionais;

RNF- Requisitos não funcionais;

SQL- Structured Query Language, ou Linguagem de Consulta Estruturada;

1. Requisitos

Nos **requisitos funcionais** preocupamo-nos em descrever as funcionalidades e os serviços do sistema, ou seja, as funções que o sistema deve fornecer para o cliente e como o sistema se comportará em determinadas situações:

* [RF001] O Sistema deve registar clientes e Motoristas (entrada).
* [RF002] O Sistema deve emitir um histórico de viagens (saída).
* [RF003] O Sistema deve passar uma viagem da situação "solicitada" para "confirmada " quando o Gestor tiver feito a confirmação (mudança de estado)
* [RF004] O Gestor, cliente e motorista o podem consultar seus dados no sistema.
* [RF005] O sistema deve permitir ao gestor e o cliente monitorar a viagem, isto e, verificar quanto tempo falta ate ao destino de chegada.
* [RF006] O sistema deve permitir o gestor fazer toda e qualquer alteração de dados existentes na base de dados;
* [RF007] O sistema garantir sempre que o supervisor de um “*check”* para confirmar a partida de um determinado cliente;
  1. Requisitos de utilizadores

[. Deve descrever os serviços fornecidos aos utilizadores.

Segundo Summerville, São declarações, em uma linguagem natural com diagramas, de quais serviços o sistema deverá fornecer a seus usuários e as restrições com as quais este deve operar].

Ex:

Requisito de Utilizador 1

1. O MHC deve gerar relatórios gerenciais mensais que mostram o custo dos medicamentos prescritos por cada clinica durante esse mês.
   1. Requisitos Funcionais (Casos de usos)

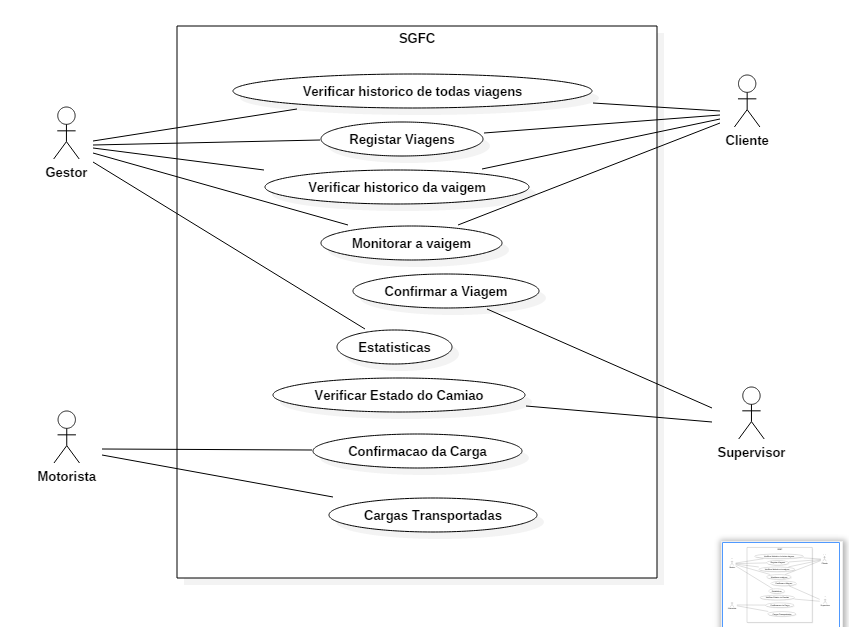


Figure 1: Diagrama de casos de uso

* 1. Requisitos Não Funcionais

Os requisitos não funcionais definem propriedades e restrições do sistema como tempo, espaço, linguagens de programação, versões do compilador, SGBD, Sistema Operacional, método de desenvolvimento, etc. Uma dica importante é que os requisitos não funcionais são geralmente mensuráveis e assim devemos preferencialmente associar uma medida ou referência para cada requisito não funcional.

* [RNF001] O sistema deve imprimir o relatório em até 5 segundos para os caso do uso em que pretendemos visualizar histórico.
* [RNF002] Todos os relatórios devem seguir o padrão de relatórios especificado pelo Gestor;
* [RNF003] O sistema deve ser implementado nas LP’s HTML, PHP, JavaScript e SQL.
* [RNF003] O sistema deve mostrar constantemente a localização do motorista;

Ex:

**[RNF001] Apresentar Tutorial**

Neste tutorial pretende-se indicar as etapas para o processo ***registar uma viagem***.

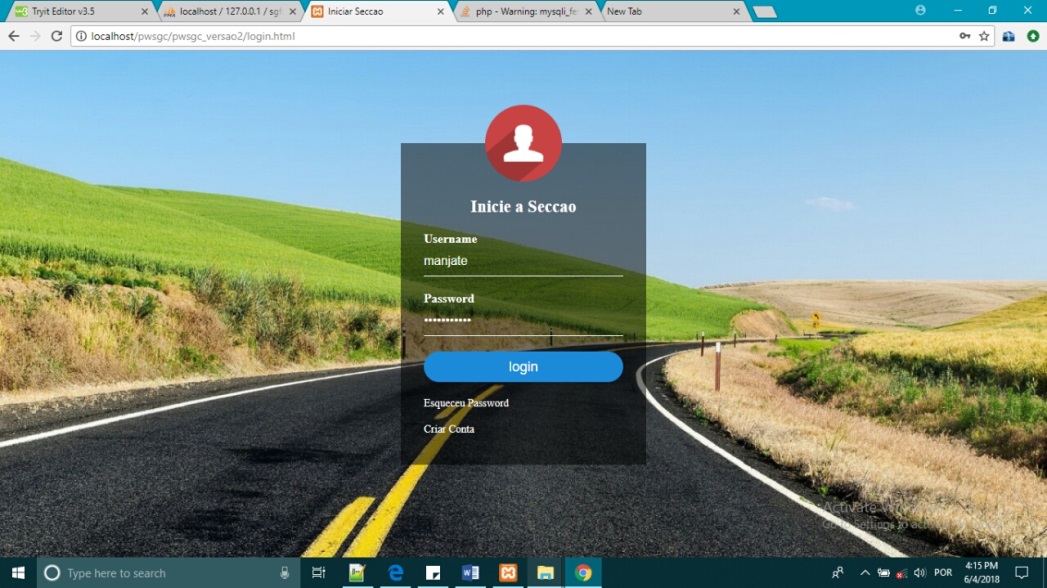


Figure 2: Página de login

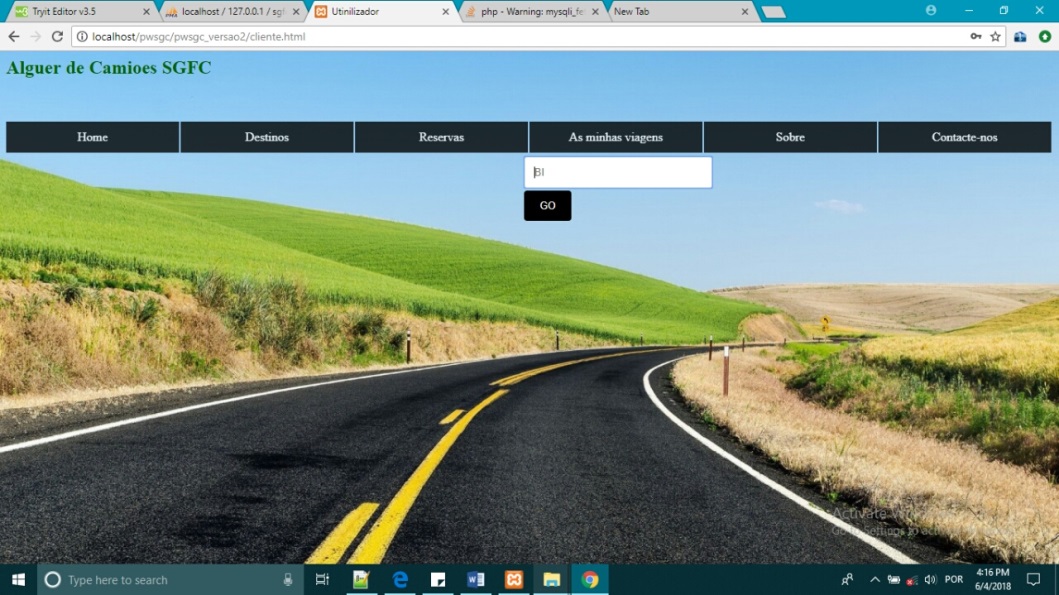


Figure 3: Página inicial

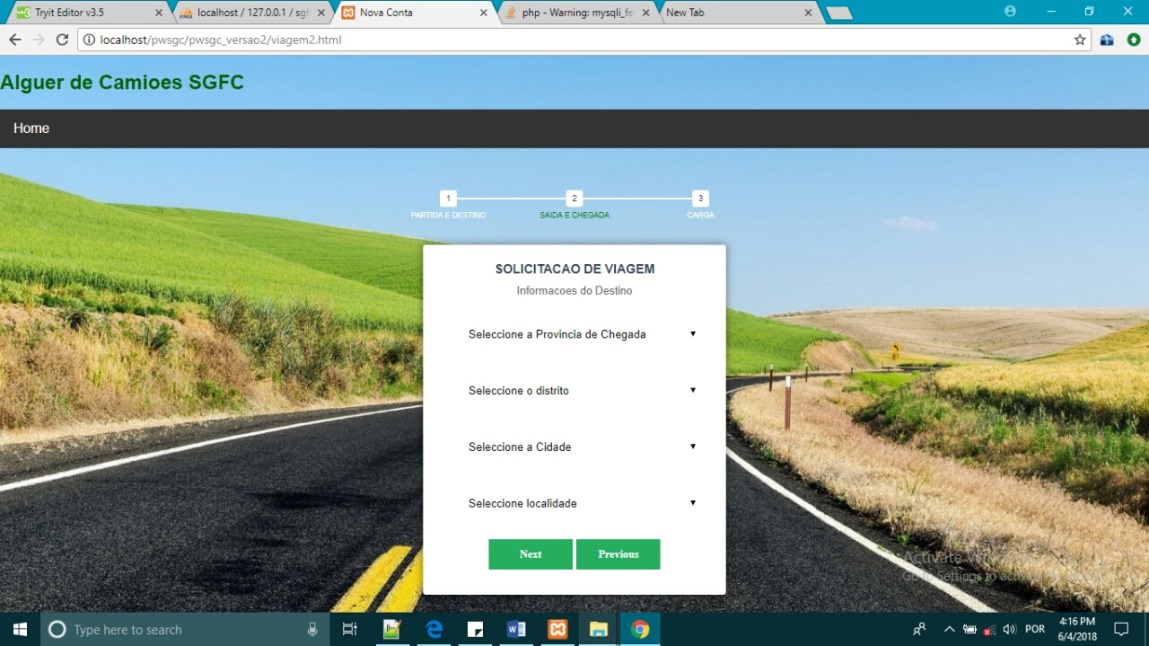


Figure 4: Registo de viagem

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Prioridade:** | **🞎** | **Essencial** | 🗹 | **Importante** | **🞎** | **Desejável** |
|  |  |  |  |  |  |  |

1. Modelos do sistema

Pretende nesta etapa mostar como a classes relacionam-se no sistema, a utilidade de cada método e bem como os seus atributos:

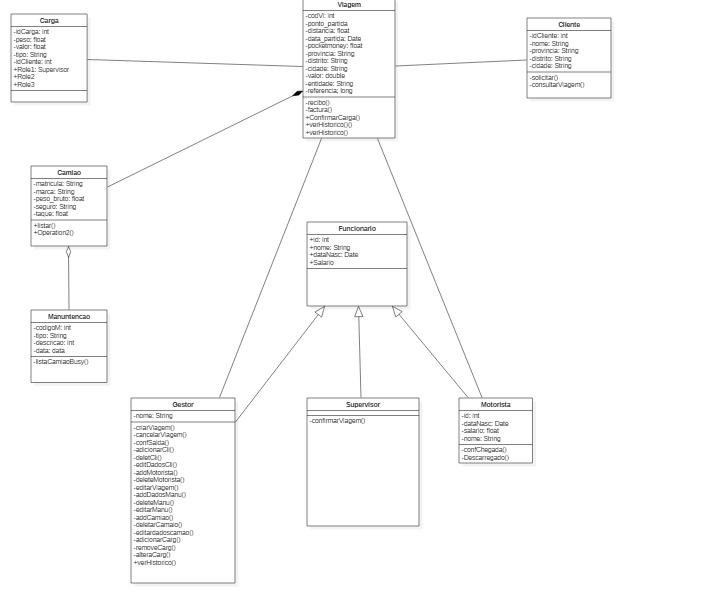


Figure 5: Diagrama de classes

* 1. Diagramas de Classes

Para um melhor entendimento do relacionamento entre os diversos atributos e classes neste ponto ir-se-á descrever o que cada elemento desempenha no sistema:

**[DC001] Viagem**

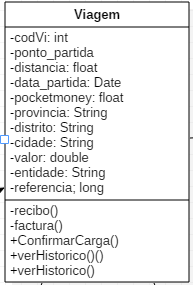


Figure 6: 6DC: Viagem

A classe Viagem, possui os atributos:

codVi – que representa o código cada viagem feito.

ponto\_partida – e local de partida, isto e, o lugar aonde e carregado o camião;

distancia – e a distancia a ser percorrida do ponto de partida ate ao destino final;

data\_partida- data de partida do camião já carregado;

pocketmoney – valor de ajuda de custo que o motorista ira levar durante a viagem.

provincia- pronvicia de destino da carga;

distrito - distrito de destino da carga;

cidade- cidade de destino da carga;

valor – valor que o cliente ira pagar pelo transporte;

entidade- entidade/instituição na qual se efectua os pagamentos;

referencia- conta para a qual se efectuara o pagamento;

recibo()- documento passado ao cliente depois que e confirmado o pagamento

factura() – documento emitido antes do pagamento;

confirmarCarga()- confirmar a carga a ser transportada;

verHistorico() – visualizar as viagens efectuadas.

**[DC002] Carga**

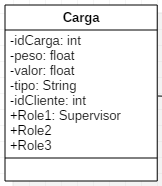


Figure 7: DC: Carga 1

A classe carga tem o seguinte atributos/metodos:

idCarga – código único identificador da carga;

peso- peso da carga a ser transportada;

valor- este atributo refere-se ao valor o qual a carga esta avaliada;

tipo- e a carga/ tipo de carga a ser transportada., Ex. Cimento.

idCliente- o código único identificador que relaciona a carga e o cliente

**[DC003] Gestor**

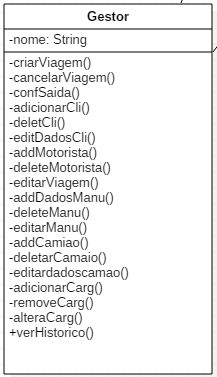


Figure 8: DC: Gestor 1

A classe Gestor possui os seguintes:

nome – e o nome do gestor do sistema e ou da empresa;

Os métodos abaixo são responsáveis por permitir o gestor a:

criarViagem() – criar uma viagem;

cancelarViagem()- cancelar uma viagem, que já havia sido criada;

confSaida()- confirmar a saída/viagem dos camiões;

adicionarCli() – adiciona um novo cliente;

editDados()- alterar dados referentes ao cliente;

addMostorista()- adiciona um motorista novo;

deleteMotorista() – remove um motorista no sistema;

editarViagem() - altera os dados da viagem;

addDadosManu()- estado de um determinado camião em manutenção;

deleteManu()- remove dados de manutenção de um camião;

editarManu()- altera os dados de manutenção;

addCamiao()- adiciona um novo camião no sistema;

deletarCamaio() – remove um camião do sistema;

editardadoscamao()- altera atributos de um camião;

adicionarCarg()- adiciona carga de uma viagem;

removeCarg()- remove uma determinada carga do sistema;

alteraCarg()- altera os atributos de uma carga;

verHistorico()- retorna todas operecoes(viagens) feitas no sistema;

**[DC004] Manutenção**

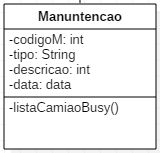


Figure 9: Manutenção

A classe manutenção possui os seguintes atributos/métodos:

codigoM- código que identifica o um camião que foi a manutenção;

tipo- indica o tipo de manutenção que o camiao efectuou. Ex. Reparar radiador;

descricao – Descreve as circunstancias que levaram o camião a manutenção;

data- o dia em que o camião foi a manutenção;

listaCamiaoBusy()- visualiza a lista de camiões em manutenção;

**[DC005] Cliente**

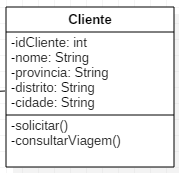


Figure 10: Cliente

A classe Cliente possui os seguintes atributos/métodos:  
idCliente- código único identificador de cada cliente;

nome- nome do cliente;

provincia- província de onde o cliente solicita a viagem;

distrito- distrito de onde o cliente solicita a viagem;

cidade- cidade onde o cliente solicita a viagem;

solicitar() – permite ao solicitar uma viagem;

consultarViagem() – consulta o estado da viagem solicitado;

**[DC006] Camiao**

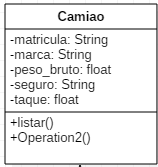


Figure 11: Camião

A classe Camiao possui os seguintes atributos/métodos:

matricula- matricula do camião;

marca- marca do camião;

peso\_bruto- carga máxima que o camião suporta;

seguro- verifica se o camião tem ou não seguro;

taque- indica o numero de litros que o tanque suporta;

listar()- lista todos os camiões;

**[DC006] Motorista**

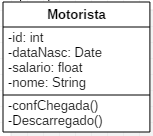


Figure 12: Motorista

A classe Motorista possui os seguintes atributos/métodos:

id- código identificador do motorista;

dataNasc- data de nascimento do motorista.

salario- salario do motorista;

nome - nome completo do motorista

confChegada()- confirmação de chegada do motorista;

Descarregado()- confirmação da entrega da carga;

**[DC007] Funcionario**

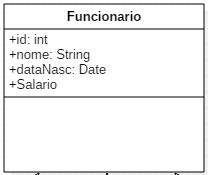


Figure 13: Funcionário

A classe Funcionario e também “pai” das classes Gestor , Motorista e Supervisor possui os seguintes atributos/métodos:

Id - código identificador do funcionário;

salario- salario do funcionário;

nome - nome completo do funcionário;

dataNasc- data de nascimento do funcionário.

**[DC008] Supervisor**

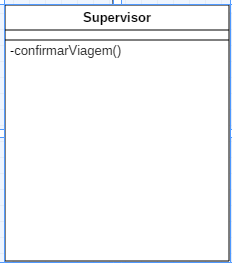


Figure 14: Supervisor

A classe Supervisor possui o seguinte método:

confirmarViagem()- indica se o camiao esta em condições partir ou não.

* 1. Diagrama de Sequencia

[Criar viagem]

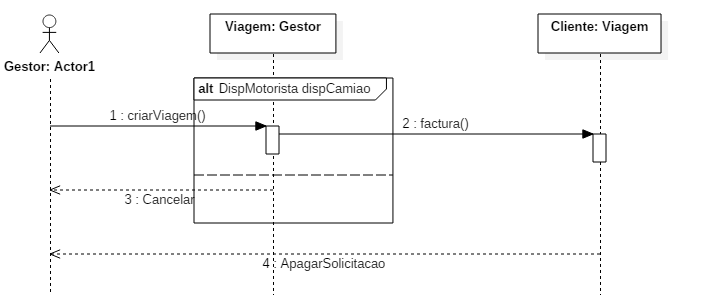


Figure 15: Diagrama de sequência criar viagem

Código: [DS001] Registrar Venda

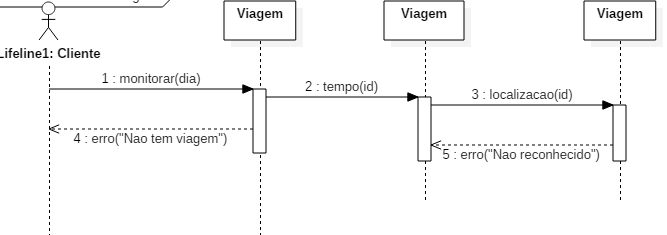


Figure 16: Registar

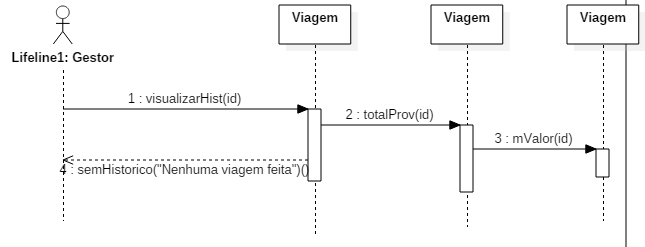


Figure 17: Legenda

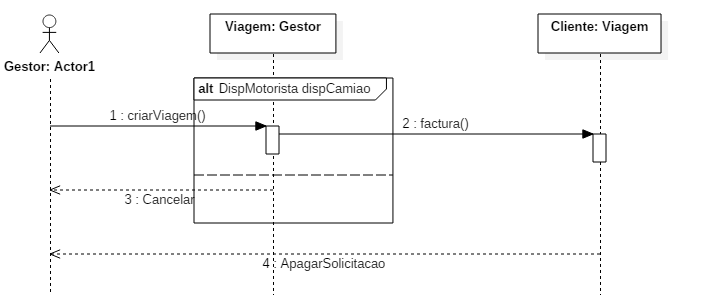


Figure 18:Legenda

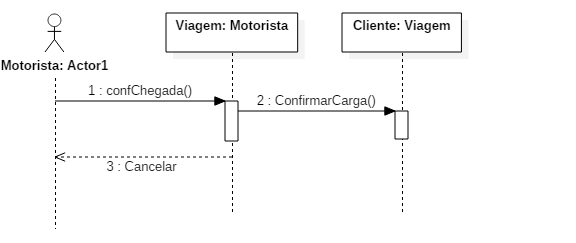


Figure 19: Legenda

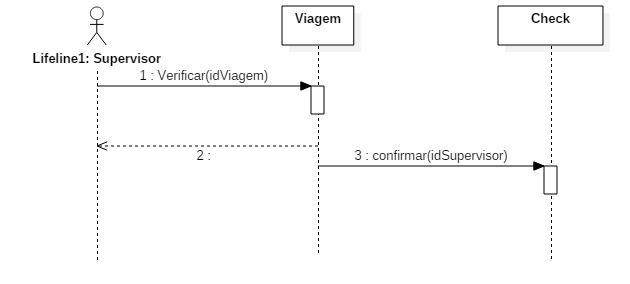


Figure 20: Legenda

* 1. Diagrama de Estados

[Descrição do diagrama a apresentar em seguida]

**Camiao**

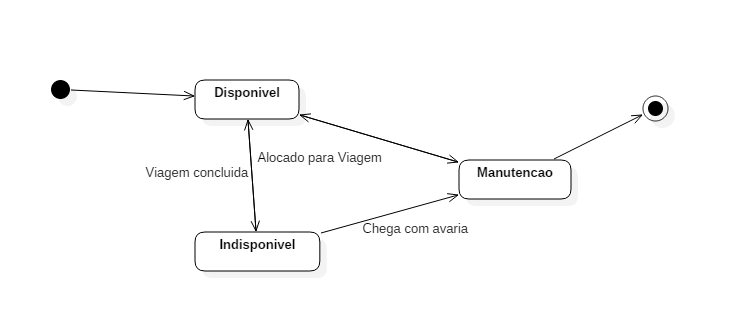


Figure 21: Diagrama de estados do camião

**Alocar Motorista**

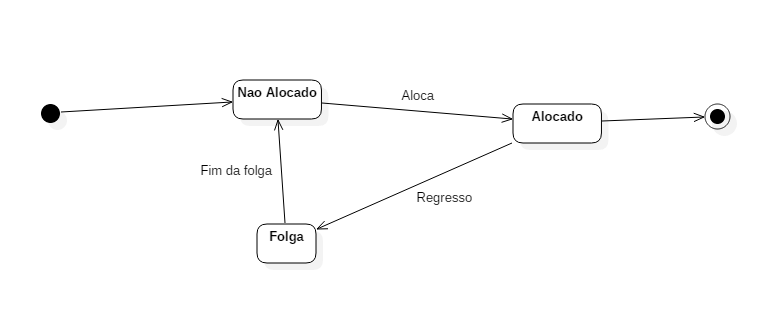


Figure 22: Alocação do motorista

**Viagem**

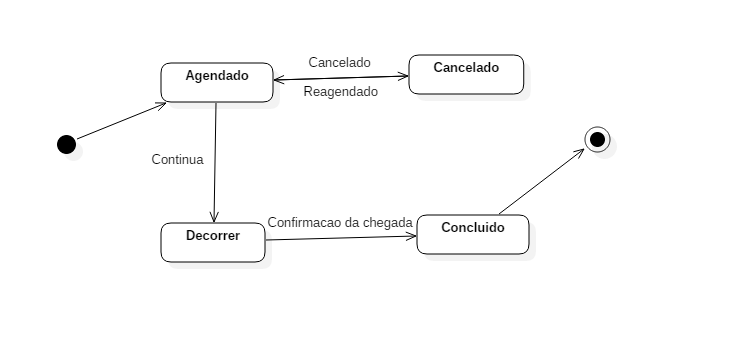


Figure 23: Viagem

Código: [DE001] Venda

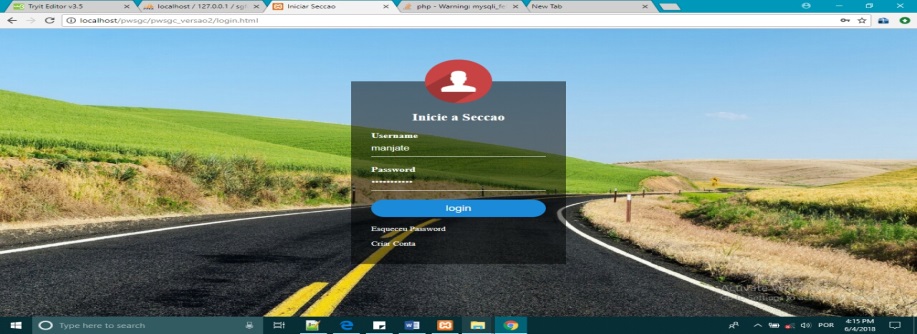
* + 1. Diagrama de Actividades

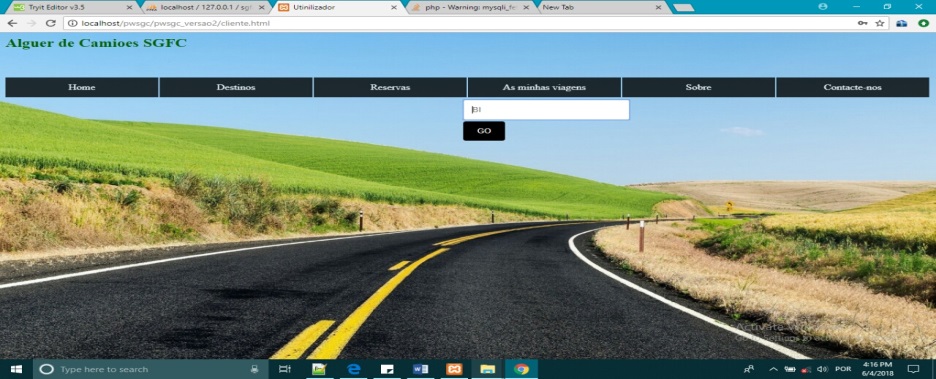
[Descrição do diagrama a apresentar em seguida]

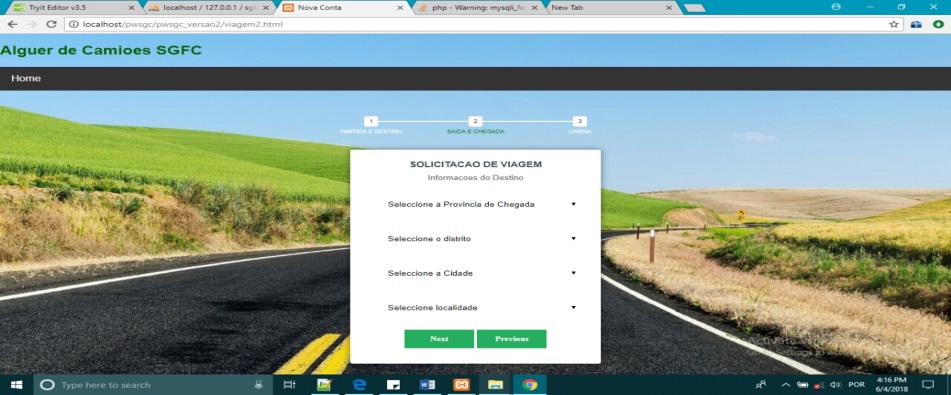
[Diagrama de actividades]

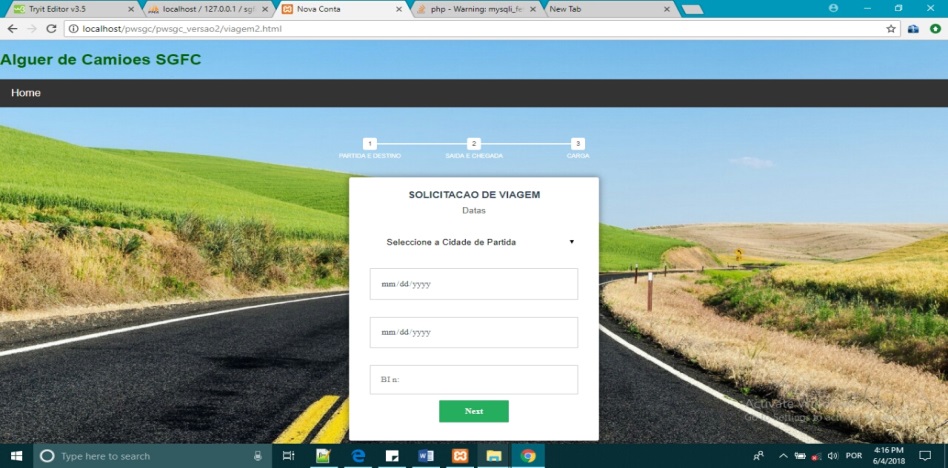
Código: [DA001] Venda

1. Interface de Utilizador (Protótipos)









1. Testes

[Casos de testes]

|  |  |
| --- | --- |
| **Caso de teste** | V 11- **Confirmação da entrega carga** |
| Pre-condições | 1.Login feito pelo motorista.  2.Ter Entrado com o codigo da viagem. |
| procedimentos | 1. O actor (motorista) seleciona a opção confirmar a carga.  2. O sitema deve responder com um campo para inserir o código da Viagem.  3. Deve entrar co o código  4. O sistema apresenta os dados da Viagem e um botão confirmar Entrega.  5. O motorista clica no botão enviar a confirmação. |
| Resultado Esperado | Obter a mensagem “ Confirmado Com Sucesso”  Enviar mensagem de confirmação da carga ao cliente e ao Gestor. |
| Dados de entrada |  |
| Criterios Especiais | Não se aplica |
| Ambiente |  |
| Implementacao | Automatica |
| Interação | Uma vez |

Table 1: caso de teste - confirmar entrega

|  |  |
| --- | --- |
| **Caso de teste** | V 1- **Registar Viagem** |
| Pre-condições | Ter recebido uma notificacão  Ter clicado no botão notificações  A notificação deve ter assunto: Solicitação de viagem |
| procedimentos | 1.O gestor, recebe a notificação e clica no botão “notificação” que carrega uma lista de notificacções.  2. O actor deve abrir as notificações com assunto solicitação.  3.O sistema deve exibir um formulario devidamente preenchido pelo cliente e no final da pagina um botão registar .  4.Ao clicar no botão devera ser disponibilizado uma pagina com as opções de alocação de do camião, motorista e a carga á viagem, e a data de partida.  5.Por fim deve clicar no botão confirmar viagem que regista a viagem na BD(Base de dados) e uma mensagem é enviada ao cliente, supervisor e motorista. |
| Resultado Esperado | Registar Viagem na BD e notificar o cliente, supervisorr e motorista sobre a viagem agendada. |
| Dados de entrada |  |
| Criterios Especiais | Não se aplica |
| Ambiente |  |
| Implementacao | Manual |
| Interação | Uma vez |

Table 2: caso de teste - registar viagem

|  |  |
| --- | --- |
| **Caso de teste** | V2- **Verificar o histórico** |
| Pre-condições | Ter clicado no botão Ver Historico Na Pagina Inicial do Gestor. |
| procedimentos | 1.O actor(Gestor ) deve clicar no botão ver historico localizado no topo da página do lado direito do botão registar.  2.O sistema deve responder redirecionando para outra página, que faz a listagem do historico |
| Resultado Esperado | carregar o historico de todas as viagens. |
| Dados de entrada |  |
| Criterios Especiais | -- |
| Ambiente |  |
| Implementacao | Manual |
| Interação | 1 interação |

Table 3: Verificar histórico

|  |  |
| --- | --- |
| **Caso de teste** | V3- **Verificar a solicitação** |
| Pre-condições | 1.Login feito pelo cliente  2. ter preenchido o formulario de solicitação daviaagem e clicado no botão enviar. |
| procedimentos | 1. Depois de clicado a opção solicitar viagem, o sitema devera responder com um formulario , onde o cliente devera preencher o mesmo com os dados da viagem.  2.Deverá clicar no botão Enviar Solicitação. |
| Resultado Esperado | 1.Obter a Mensagem “ Solicitaçao enviada”;  2. Gestor deve receber a solicitação |
| Dados de entrada | Data de partida e a Carga |
| Criterios Especiais | -- |
| Ambiente |  |
| Implementacao | Manual |
| Interação | 1 interação |

Table 4: Verificar solicitação