**PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS  
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA**

**Pós-graduação *Lato Sensu* em Arquitetura de Software Distribuído**

**Renato Pereira da Cruz**

**SISTEMA CONTROLE LOGÍSTICA**

Belo Horizonte

2018

**Renato Pereira da Cruz**

**SISTEMA CONTROLE LOGÍSTICA**

Trabalho de Conclusão de Curso de Especialização em Arquitetura de Software Distribuído como requisito parcial à obtenção do título de especialista.

Orientador(a): Tadeu Faria

Belo Horizonte

2018

*Dedico este trabalho primeiramente а Deus, pоr ser o meu tudo.*

*Aо mеu pai e minha mãе que me deram toda a educação que eu precisava.*

*A minha esposa e filha pela compreensão е a toda minha família.***AGRADECIMENTOS**

Agradeço ao professores da PUC pelas aulas, exercícios e provas que ajudaram nesta jornada.

Agradeço a toda organização da PUC, pelas informações e o excelente serviço prestado.

Agradeço aos amigos de trabalho que contribuíram para o meu crescimento profissional.

**RESUMO**

Este trabalho trata-se de uma solução para um Sistema de Controle Logístico, utilizando dispositivos móveis ou navegadores desktop. Onde o objetivo principal é atingir maior agilidade no processo de coleta, distribuição e entrega de cargas, com priorização das entregas de acordo com horários definidos. Assim há uma economia de combustível e melhor aproveitamento da capacidade do veículo; pois agrupa as entregas por roteirização. Além do monitoramento de entregas. Trazendo assim maior agilidade, pois as consultas são on-line via Internet pelos usuários. O usuário tem uma visão clara e centralizada das informações. Solucionando, assim, os problemas com medidas simples, porém eficazes, otimizando os processos que impactam diretamente na qualidade da prestação do serviço realizado pela empresa.

**Palavras-chave:** Arquitetura de Software. Desenvolvimento. Transporte. Serviços Web. Web Responsiva.

**SUMÁRIO**

[1. Objetivos do trabalho 7](#_qsh70q)

[2. Descrição geral da solução](#_30j0zll) 7

[2.1. Apresentação do problema 7](#_3as4poj)

[2.2. Descrição geral do software (Escopo) 8](#_1pxezwc)

[3. Definição conceitual da solução](#_49x2ik5) 8

[3.1. Requisitos Funcionais 8](#_2p2csry)

[3.2 Requisitos Não-Funcionais 10](#_147n2zr)

[3.3. Restrições Arquiteturais 13](#_3o7alnk)

[3.4. Mecanismos Arquiteturais 13](#_23ckvvd)

[4. Modelagem e projeto arquitetural](#_ihv636) 14

[4.1. Modelo de casos de uso 14](#_32hioqz)

[4.2. Descrição resumida dos casos de uso 17](#_17dp8vu)

[4.3. Modelo de componentes 20](#_1hmsyys)

[4.4. Modelo de implantação 22](#_41mghml)

[4.5. Modelo de dados 24](#_2grqrue)

[5. Prova de conceito / protótipo arquitetural](#_vx1227) 25

[5.1. Implementação e implantação 25](#_3fwokq0)

[5.1.1 Visão lógica 28](#_3fwokq0)

[5.1.2 Módulos compilados e Dependências 31](#_3fwokq0)

[5.2. Interfaces/ APIs 31](#_1v1yuxt)

[6. Avaliação da Arquitetura](#_4f1mdlm) 33

[6.1. Análise das abordagens arquiteturais 33](#_z337ya)

[6.2. Identificação dos atributos de qualidade 34](#_3j2qqm3)

[6.3. Cenários 34](#_2u6wntf)

[6.4. Avaliação 35](#_19c6y18)

[6.5. Resultado 41](#_2xcytpi)

[7. Conclusão 41](#_1ci93xb)

[REFERÊNCIAS](#_3tbugp1) 41

[APÊNDICES](#_28h4qwu) 41

**1. Objetivos do trabalho**

O objetivo geral deste projeto é apresentar um projeto de arquitetura para otimizar o controle logístico de uma transportadora, que vão desde informações gerenciais da empresa, processamento de transações fiscais, status do transporte, em um ambiente web intuitivo e de fácil utilização. Tudo fica registrado no software, dispensando, em alguns casos, a impressão em papel.

Os objetivos específicos são:

1. Criar módulo que possibilite a solicitação de coleta, que interaja com outras transportadoras, que seja responsivo, atendendo tanto desktop quanto usuários com acesso móbile.

2. Cria módulo de controle de tabelas de frete. Que dará apoio à negociação do frete. Possibilitando cadastrar tarifas e realizar simulações.

3. Cria módulo de controle de devolução. Onde o usuário irá registrar manifestações referentes às entregas.

**2. Descrição geral da solução**

**2.1. Apresentação do problema**

Boa parte das transportadoras têm dificuldade em realizar as entregas dentro do prazo. Além da necessidade de prestar um serviço de qualidade e contornar os contratempos que ocorrem diariamente. A falta de informações sobre os procedimentos durante o deslocamento das encomendas também é um problema que precisa ser minimizado. A maioria destes problemas pode ser solucionado com medidas simples, otimizando o tempo dos procedimentos, o que impacta diretamente na qualidade dos serviços prestados. O uso do software resolve o problema da falta de informação e permite o controle de toda a operação desde a coleta até a entrega do produto. E com isso ficar atento a possíveis atrasos na entrega e poder agir rapidamente.

**2.2. Descrição geral do software (Escopo)**

O Sistema de gestão de logística tem funcionalidades, que vão desde as solicitações de coletas, simulador de tarifas, acompanhamento de entregas, geração de manifestações, até integração para geração de CTe. As informações gerenciais através de relatórios permitem um controle rígido dos procedimentos da empresa. É possível, por exemplo, emitir documentos fiscais eletrônicos através do sistema, o que significa que ao emitir uma CTe ou NFS-e tudo ficará registrado no software, dispensando a impressão em papel desses documentos.

**3. Definição conceitual da solução**

**3.1. Requisitos Funcionais**

**Módulo de Coleta**

* Solicitação de coleta do cliente

O usuário deverá realizar a solicitação de coleta de cliente, informando no sistema os dados principais para realizar a coleta.

* Recebimento de coleta de empresas parceiras

Realizar a transferência de uma coleta para outras transportadoras de uma maneira direta.

* Interagir com sistema de faturamento

O sistema irá interagir com um sistema de faturamento.

* Interagir com sistema de frota

O administrador do sistema irá manter o cadastro de frotas no sistema.

**Módulo de expedição**

* Separação das entregas

O sistema deverá permitir ao usuário, separar as entregas por rota.

* Previsão das entregas

O usuário poderá visualizar a previsão de determinada entrega.

* Visualização das entregas

O usuário deverá visualizar no sistema, as entregas previstas por semana ou mês.

**Módulo de entrega**

* Controle dos bens/produtos pendentes de entrega

O usuário deverá ter o controle de todos os produtos pendentes para entrega.

* Visualização/Alteração de status, data e/ou hora

O sistema deverá permitir que o usuário visualize e/ou altere o status das entregas/coletas.

* Avaliação/Satisfação do cliente

Deverá ser possível manter o nível de satisfação do cliente através de uma avaliação.

**Módulo de frete**

* Manter cadastro de fretes

O sistema deverá possibilitar ao usuário, realizar cadastros e/ou atualizações de tarifas.

* Realizar simulações

O sistema deverá possibilitar ao usuário, realizar simulações e calcular valores do frete.

**Módulo de devolução**

* Controle de devolução

Todos os problemas deverão ser registrados no sistema, desde os produtos coletados até os produtos entregues.

**Módulo SAC**

* Controle das solicitações

O funcionário deverá ter a possibilidade de abrir um chamado sobre qualquer dúvida/problema no sistema.

**Módulo de relatório**

* Emitir relatórios

O sistema deverá fornecer relatórios de expedição, pendências, devoluções e rentabilidade.

**Módulo de pagamento**

* Realizar pagamento

O sistema deverá permitir realizar os pagamentos por cartões de créditos.

* 1. **Requisitos Não-Funcionais**
* O sistema deve prover boa usabilidade.

|  |  |
| --- | --- |
| **Estímulo** | O usuário realizando uma solicitação de coleta. |
| **Fonte do Estímulo** | Funcionalidade de solicitação de coleta do cliente. |
| **Ambiente** | Funcionamento normal do sistema. |
| **Artefato** | Módulo de coleta. |
| **Resposta** | O sistema apresenta um padrão no layout, cores e descrição dos botões em todas as telas. |
| **Medida da resposta** | O usuário realizou a solicitação sem maiores dificuldades. |

* O sistema deve suportar ambientes Web responsivos e ambientes móveis.

|  |  |
| --- | --- |
| **Estímulo** | O usuário realizando uma solicitação de coleta. |
| **Fonte do Estímulo** | Funcionalidade de solicitação de coleta do cliente. |
| **Ambiente** | Funcionamento normal do sistema. |
| **Artefato** | Módulo de coleta. |
| **Resposta** | O layout do sistema se adaptou as resoluções e tamanho das telas, ocultando os itens menos importantes de forma a ficar melhor a visualização do usuário. |
| **Medida da resposta** | Adaptação a todas as resoluções, com objetos redimensionados/ocultos de acordo com o tamanho do dispositivo. |

* O sistema deve ser rápido.

|  |  |
| --- | --- |
| **Estímulo** | Tela de agendamento de coleta. |
| **Fonte do Estímulo** | Funcionalidade de solicitação de coleta do cliente. |
| **Ambiente** | Funcionamento normal do sistema. |
| **Artefato** | Módulo de coleta. |
| **Resposta** | Dados retornados com sucesso. |
| **Medida da resposta** | Requisições Ajax não ultrapassam o tempo de 500 milissegundos. |

* O sistema deve apresentar manutenção facilitada.

|  |  |
| --- | --- |
| **Estímulo** | Um novo grid precisa ser inserido. |
| **Fonte do Estímulo** | Usuário visualizando o grid. |
| **Ambiente** | Várias telas usam o mesmo grid. |
| **Artefato** | Módulo de expedição. |
| **Resposta** | O mesmo grid precisa estar disponível em várias telas. |
| **Medida da resposta** | Foi preciso inserir apenas uma tag na página. |

* O sistema deve ser simples para testar.

|  |  |
| --- | --- |
| **Estímulo** | Execução de testes no sistema. |
| **Fonte do Estímulo** | Desenvolvedor. |
| **Ambiente** | Ambiente de Desenvolvimento. |
| **Artefato** | Módulo de coleta. |
| **Resposta** | O sistema testou os métodos de CRUD. |
| **Medida da resposta** | O sistema permite executar os testes unitários com apenas um comando. |

* O sistema deve se comunicar com os sistemas dos agentes. Alguns desses sistemas são antigos e desenvolvidos com tecnologia COBOL/CICS

|  |  |
| --- | --- |
| **Estímulo** | Conexão com a API. |
| **Fonte do Estímulo** | Sistemas de requisição. |
| **Ambiente** | Funcionamento normal do sistema. |
| **Artefato** | API disponibilizada para realizar coleta. |
| **Resposta** | A API realizou a agenda de coleta com sucesso. |
| **Medida da resposta** | Informações inseridas na base de dados. |

* O sistema deve operar em qualquer período do dia e da noite.

|  |  |
| --- | --- |
| **Estímulo** | Sistema disponível. |
| **Fonte do Estímulo** | Usuário acessando o sistema |
| **Ambiente** | Funcionamento normal do sistema. |
| **Artefato** | Módulo de coleta e expedição. |
| **Resposta** | O sistema está disponível em qualquer período do dia ou da noite. |
| **Medida da resposta** | Em qualquer período o sistema responde às requisições. |

* O sistema deve apresentar altos padrões de segurança.

|  |  |
| --- | --- |
| **Estímulo** | Acessar o sistema usando criptografia. |
| **Fonte do Estímulo** | Usuário acessando o sistema |
| **Ambiente** | Funcionamento normal do sistema. |
| **Artefato** | Módulo de coleta e expedição. |
| **Resposta** | O sistema deve trafegar os dados criptografados. |
| **Medida da resposta** | O sistema utiliza o protocolo HTTPS. |

* O sistema deve estar disponível 24 horas por dia nos sete dias da semana

|  |  |
| --- | --- |
| **Estímulo** | Instância individual inativa ou indisponível |
| **Fonte do Estímulo** | Agente externo. |
| **Ambiente** | Falha em algum equipamento físico. |
| **Artefato** | Balanceadores de carga. |
| **Resposta** | O sistema é deslocado para as instâncias disponíveis. |
| **Medida da resposta** | O sistema não fica indisponível em caso de falha física. |

**3.3. Restrições Arquiteturais**

* O sistema deve ser desenvolvido em .Net/C# no Background e Angular 2 para o FrontEnd.
* O sistema deve utilizar um serviço de busca de CEP pela internet.
* O sistema deve suportar acessos por tablets e smartphones, Além dos principais browsers, sendo eles: Google Chrome, Mozilla Firefox e Safari.
* O sistema deve utilizar acesso seguro, permitindo a utilização de certificado digital, proveniente de autoridade certificadora reconhecida, instalada no servidor com segurança baseada em criptografia SSL.
* O sistema deve fornecer um serviço para que os parceiros insiram solicitações de coleta
* Todas as integrações devem ser feitas com um serviço de WebAPI.

**3.4. Mecanismos Arquiteturais**

|  |  |  |
| --- | --- | --- |
| **Mecanismo de Análise** | **Mecanismo de Design** | **Mecanismo de Implementação** |
| Integração | WebAPI | Json e XML |
| Build | Gerador de artefato para o servidor | Visual Studio e Angular CLI |
| Gerenciador de pagote | Instalador de módulos | Nuget e NPM |
| Deploy | Implantação em produção | Git |
| Versionamento | Versionamento do código fonte. | GitHub |
| Alta disponibilidade | Balanceamento de carga | NGINX |
| Persistência | Banco de dados relacional | MySQL |
| Persistência | Framework ORM | Dapper |
| Front-End | Interface de comunicação com o usuário | HTML CSS JavaScript |

**4. Modelagem e projeto arquitetural**

**4.1. Modelo de casos de uso**

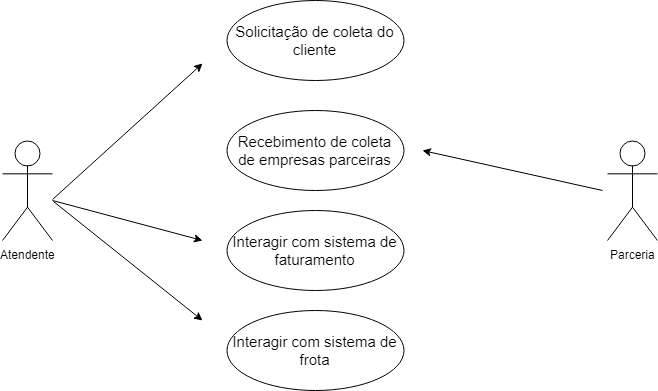


Figura 1.1 – Caso de uso do Módulo de coleta

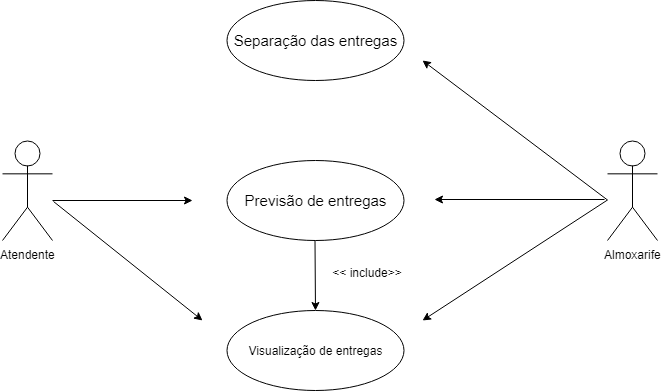


Figura 1.2 – Caso de uso do Módulo de expedição

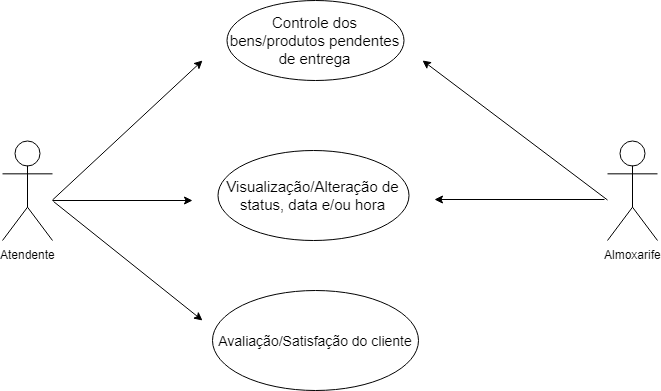


Figura 1.3 – Caso de uso do Módulo de entrega

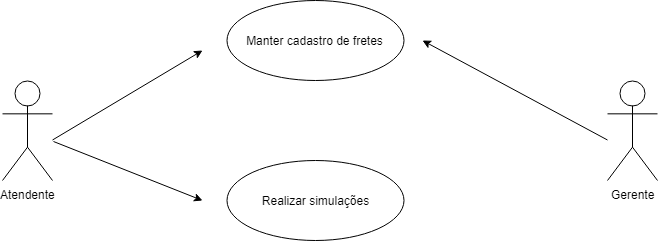


Figura 1.4 – Caso de uso do Módulo de frete

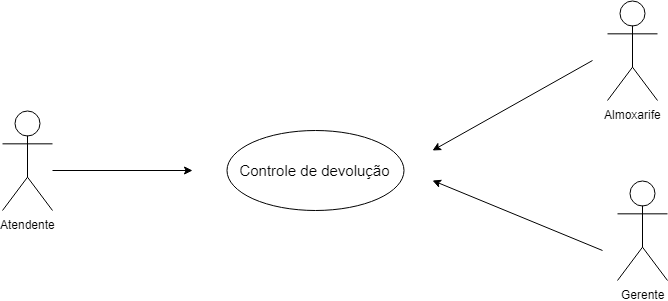


Figura 1.5 – Caso de uso do Módulo de devolução

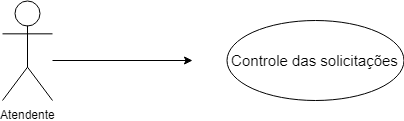


Figura 1.6 – Caso de uso do Módulo de SAC

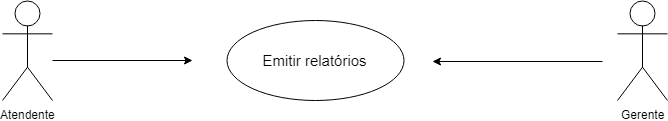


Figura 1.7 – Caso de uso do Módulo de relatório

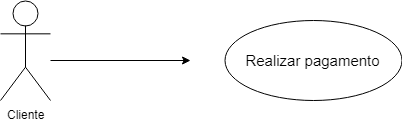


Figura 1.8 – Caso de uso do Módulo de pagamento

**4.2. Descrição resumida dos casos de uso**

**Módulo de coleta**

Solicitação de coleta do cliente

|  |  |
| --- | --- |
| Como atendente | ID: 01 |
| Eu quero realizar uma solicitação de coleta, informando a origem da coleta, o destino final, o valor da nota fiscal e visualizar o valor do frete. | |
| Para já informar o cliente o valor da prestação do serviço. | |
| Prioridade: Alta | Estimativa: 20h |

Recebimento de coleta de empresas parceiras

|  |  |
| --- | --- |
| Como almoxarife | ID: 02 |
| Eu quero que as empresas realizem solicitações de forma automática, sem precisar de uma solicitação por telefone. | |
| Automatizar o processo de abrir solicitação de coleta. | |
| Prioridade: Alta | Estimativa: 15h |

Interagir com sistema de faturamento

|  |  |
| --- | --- |
| Como atendente | ID: 03 |
| Eu quero poder emitir uma CT-e para transportar as mercadorias dos clientes. | |
| Estar de acordo com a legislação. | |
| Prioridade: Alta | Estimativa: 30h |

Interagir com sistema de frota

|  |  |
| --- | --- |
| Como atendente | ID: 04 |
| Eu quero manter cadastro de clientes parceiros e frotas de maneira fácil e intuitiva, sem precisar navegar em muitas telas. | |
| Manter o controle de cadastro no sistema. | |
| Prioridade: Alta | Estimativa: 20h |

**Módulo de expedição**

Separação das entregas

|  |  |
| --- | --- |
| Como almoxarife | ID: 05 |
| Eu quero separar os produtos dos clientes por rotas e lançá-los no sistema. | |
| Planejar a melhor rota para a entrega dos produtos. | |
| Prioridade: Alta | Estimativa: 15h |

Previsão das entregas

|  |  |
| --- | --- |
| Como atendente | ID: 06 |
| Eu quero visualizar e manter o controle de todas as entregas pendentes. | |
| Manter o controle de localidade e status de todos os produtos coletados. | |
| Prioridade: Alta | Estimativa: 15h |

Visualização das entregas

|  |  |
| --- | --- |
| Como atendente | ID: 07 |
| Eu quero visualizar as entregas por períodos, sendo por mês ou semana. | |
| Planejar as entregas conforme as prioridades. | |
| Prioridade: Média | Estimativa: 25h |

**Módulo de entrega**

Controle dos bens/produtos pendentes de entrega

|  |  |
| --- | --- |
| Como almoxarife | ID: 08 |
| Eu quero controlar todos os produtos/bens pendentes de entrega, separando-os por rota, escolhendo o melhor caminho para a entrega. | |
| Para ter o controle de todos os produtos em um só lugar. | |
| Prioridade: Alta | Estimativa: 20h |

Visualização/Alteração de status, data e/ou hora

|  |  |
| --- | --- |
| Como atendente | ID: 09 |
| Eu quero visualizar/alterar os status das entregas em tempo real, e visualizar o histórico dos mesmos com data e funcionário que realizou a alteração. | |
| Para manter os status das entregas e manter o cliente informado. | |
| Prioridade: Alta | Estimativa: 30h |

Avaliação/Satisfação do cliente

|  |  |
| --- | --- |
| Como atendente | ID: 10 |
| Eu quero receber feedback dos clientes e inseri-los no sistema, com uma nota de 0 a 10. | |
| Para manter a satisfação do cliente. | |
| Prioridade: Alta | Estimativa: 15h |

**Módulo de frete**

Manter cadastro de fretes

|  |  |
| --- | --- |
| Como gerente | ID: 11 |
| Eu quero cadastrar/atualizar os preços e tarifas das coletas/entregas, podendo cadastrar vários preços por região/rota e tipo de veículo. | |
| Para consultar os preços das coletas/entregas. | |
| Prioridade: Alta | Estimativa: 25h |

Realizar simulações

|  |  |
| --- | --- |
| Como atendente | ID: 12 |
| Eu quero simular/calcular os preços das entregas e informar aos clientes no momento da solicitação da coleta. | |
| Para informar o cliente, o valor do frete no momento da solicitação. | |
| Prioridade: Alta | Estimativa: 20h |

**Módulo de devolução**

Controle de devolução

|  |  |
| --- | --- |
| Como gerente | ID: 13 |
| Eu quero ter o controle dos produtos/bens que tiveram solicitações canceladas bem como produtos devolvidos. | |
| Para ter o controle de devolução dos produtos ou cancelamentos. | |
| Prioridade: Alta | Estimativa: 25h |

**Módulo de SAC**

Controle das solicitações

|  |  |
| --- | --- |
| Como atendente | ID: 14 |
| Eu quero inserir no sistema manifestações, como reclamações, sugestões e elogios. | |
| Para ter o controle da satisfação do cliente e possíveis problemas/reclamações. | |
| Prioridade: Alta | Estimativa: 25h |

**Módulo de relatório**

Emitir relatórios

|  |  |
| --- | --- |
| Como gerente | ID: 15 |
| Eu quero gerar relatórios, no sistema, das coletas/entregas pendentes e históricos das entregas selecionando um período ou por status. | |
| Para gerar relatórios estratégicos para tomada de decisão. | |
| Prioridade: Alta | Estimativa: 25h |

**Módulo de pagamento**

Realizar pagamento

|  |  |
| --- | --- |
| Como cliente | ID: 16 |
| Eu quero realizar os pagamentos através dos cartões de crédito. | |
| Para realizar pagamentos de forma automática, sem a necessidade de gerar boletos. | |
| Prioridade: Alta | Estimativa: 25h |

**4.3. Modelo de componentes**

O diagrama de componentes abaixo mostra a organização dos componentes e tecnologias que impactam na arquitetura do sistema. Organizados separando segundo suas responsabilidades.

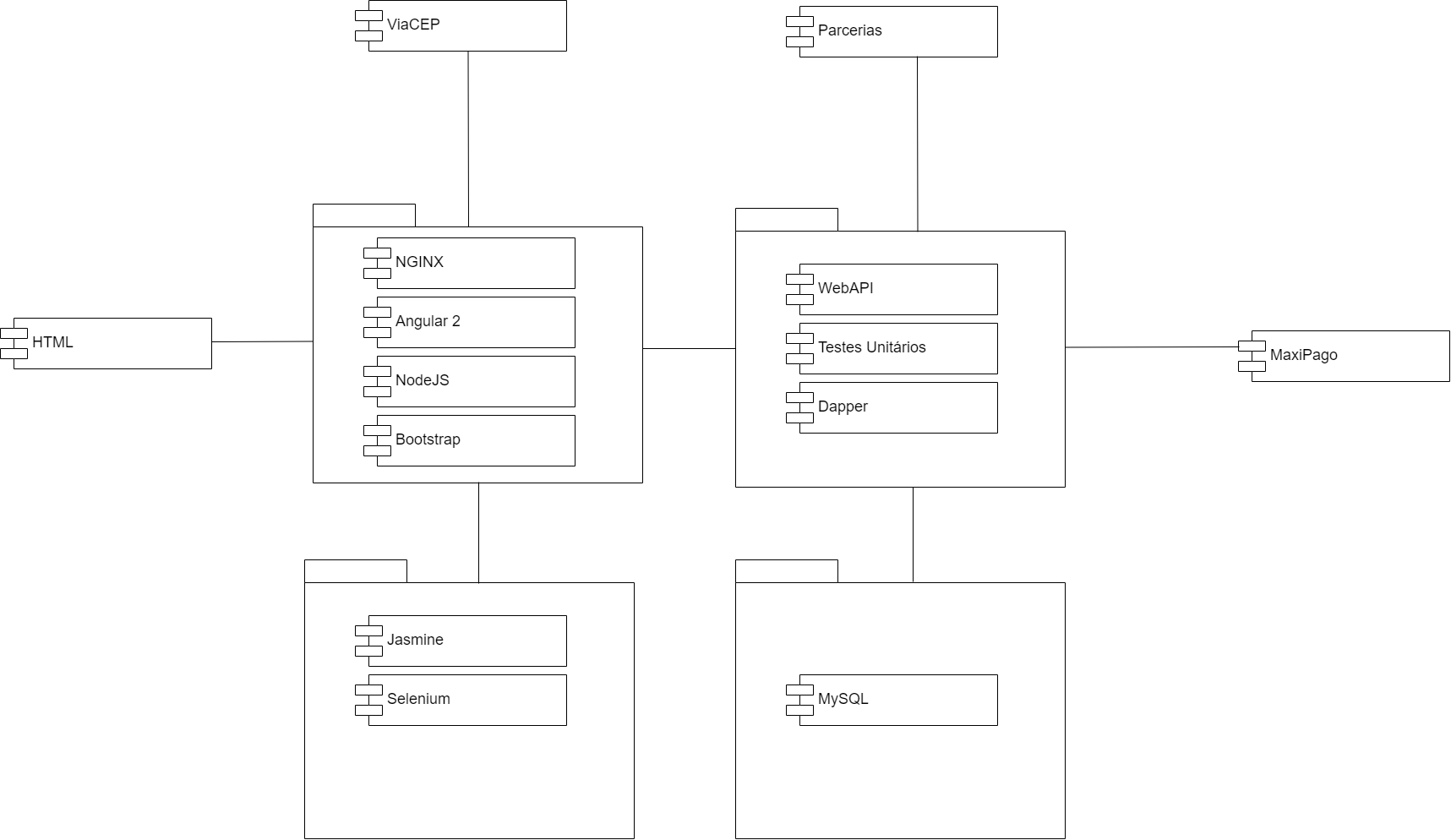


Figura 2.1 – Diagrama de componentes

Na tabela abaixo é detalhado cada componente e sua responsabilidade.

|  |  |
| --- | --- |
| **Componente** | **Descrição** |
| Páginas HTML | Interface do usuário responsável por renderizar o sistema no navegador. |
| NGINX | Servidor de proxy responsável por receber as requisições dos usuários. |
| Angular 2 | Framework web onde será desenvolvido a parte de Front End do sistema. |
| NodeJS | Plataforma JavaScript onde será instalada o módulo de Front End desenvolvido com o Angular 2. |
| Bootstrap | Componente responsável por definir o layout do sistema. |
| ViaCEP | WebAPI de terceiros que permite a consulta de endereços. |
| Parcerias | Interface WebAPI fornecida pelo sistema que permite a inserção de solicitação de coletas. |
| Jasmine | Framework JavaScript que permite fazer os testes automatizados no sistema. |
| Selenium | Extensão do Google Chrome que permite criar scripts que permite realizar teste de interface e simular a navegação dos usuários. |
| WebAPI | Componente/Interface responsável expor os serviços que serão invocados por outro módulo/sistema. |
| Testes unitários | Componente responsável por realizar os testes unitários. |
| Dapper | Biblioteca Micro ORM responsável por realizar o mapeamento objeto relacional. |
| MySQL | SGBD responsável por armazenar os dados da aplicação. |
| Maxipago | API (application programming interface) que permite aos clientes realizarem pagamentos por meio de cartões de créditos. |

**4.4. Modelo de implantação**

Modelo de implantação mostrando como os componentes serão distribuídos em seus devidos servidores.

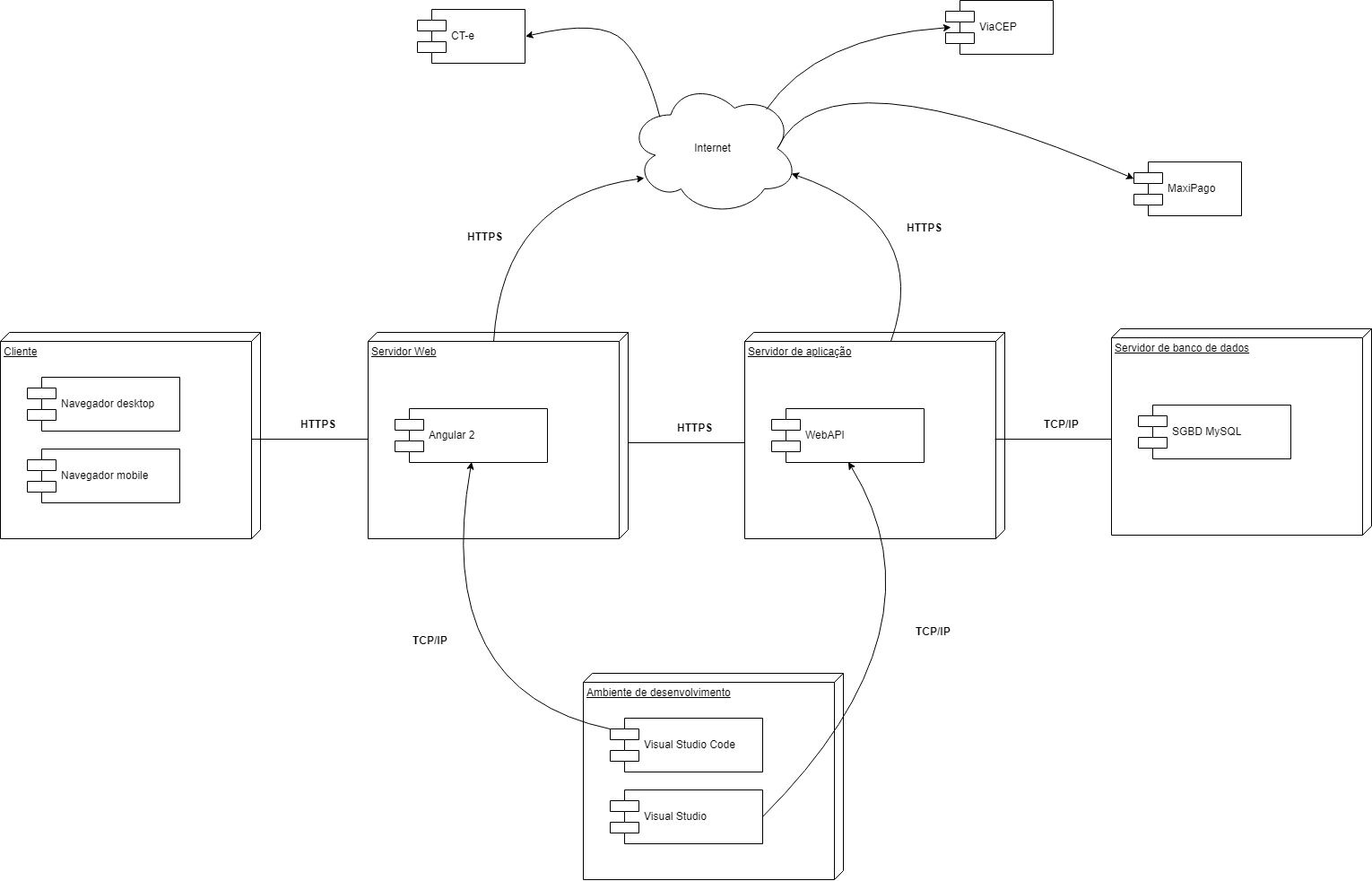


Figura 2.2 – Diagrama de implantação

|  |  |
| --- | --- |
| **Componente** | **Descrição** |
| Navegadores web e móbile | Browser utilizado pelos usuários para interagir com as páginas do sistema, podendo ser navegador web ou mobile. |
| Servidor Web | Servidor responsável pelo sistema que irá interagir com os usuários. |
| Servidor de aplicação | Servidor responsável pelos serviços do sistema e fornecer API para outros módulos/sistemas. |
| Servidor de banco de dados | Servidor onde será instalado o SGBD MySQL, que será responsável por armazenar tos os dados do sistema. |
| Serviços CT-e | API disponibilizada por terceiros que será responsável por gerar as notas fiscais. |
| ViaCEP | Serviço disponibilizado por terceiros para consulta de endereços. |
| Ambiente de desenvolvimento | Ambiente onde serão desenvolvidos os módulos de Front End e Serviços. |

**4.5. Modelo de dados**

Diagrama de classe representando o modelo lógico do sistema.

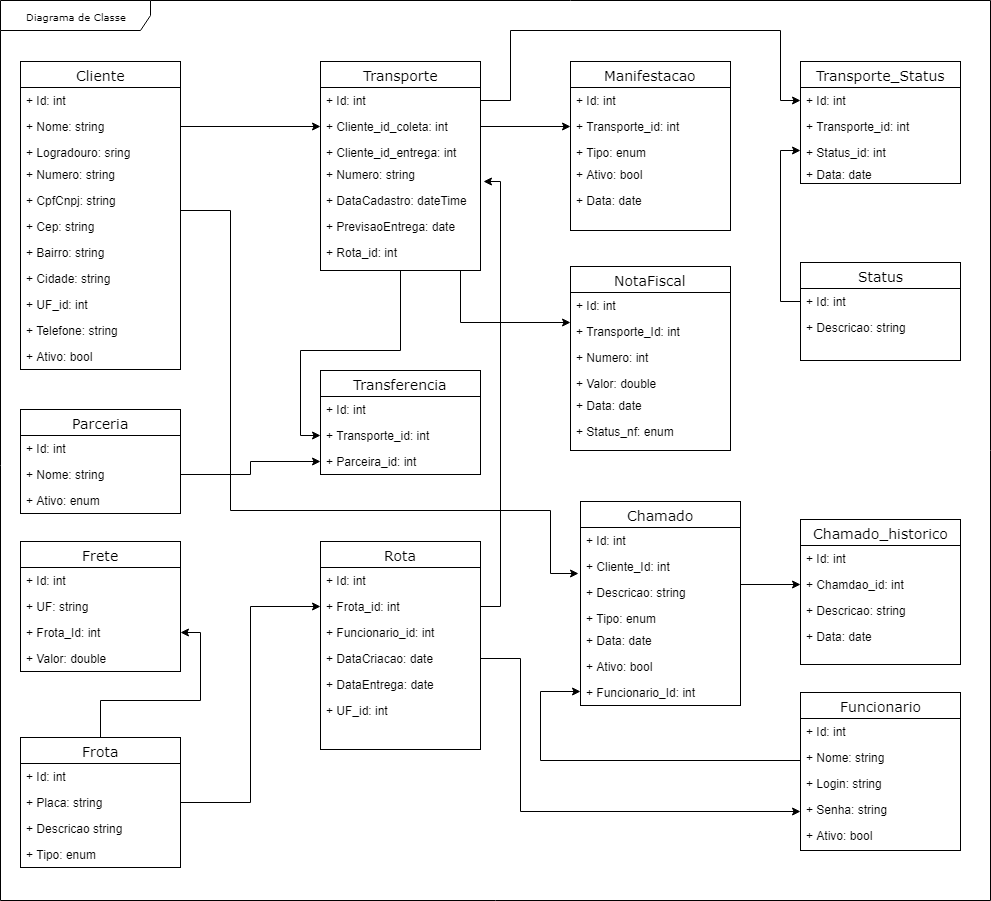


Figura 3.1 – Diagrama de classe

Diagrama de pacote representando os módulos do sistema.

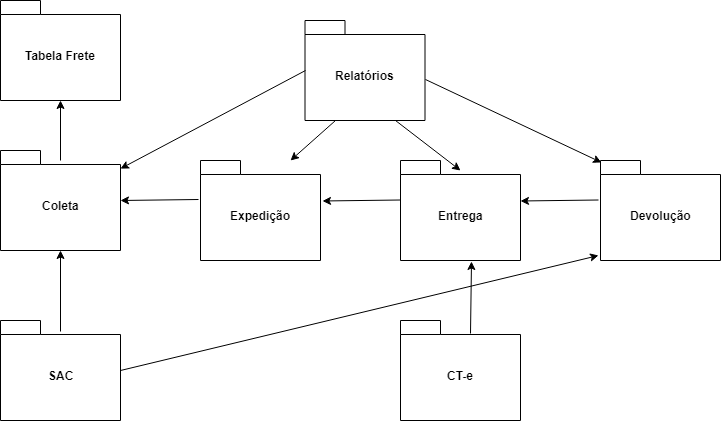


Figura 3.2 – Diagrama de pacote

**5. Prova de conceito / protótipo arquitetural**

**5.1. Implementação e implantação**

A implementação do protótipo visa provar que os requisitos funcionais e não funcionais têm condições de serem atendidos pelo projeto arquitetural definido. O protótipo cobre os seguintes casos de usos:

Solicitação de coleta do cliente do Módulo de coleta.

Manter cadastro do cliente do Módulo de frete.

Separar entregas do Módulo de entrega.

Nessa POC, pretende-se validar os seguintes requisitos não funcionais:

• O sistema deve prover boa usabilidade

O objetivo é proporcionar ao usuário, uma navegação simples, rápida e intuitiva. Acessar as funcionalidades de forma fácil.

Os critérios de aceite para este requisito são:

A tela do sistema deverá ser de fácil de navegar.

O sistema deverá manter um padrão para todas as telas.

• O sistema deve suportar ambientes Web’s responsivos e ambientes móveis.

Hoje em dia as pessoas acessam a internet de dispositivos móveis devido à necessidade de mobilidade das pessoas, assim, o sistema deverá ter um layout responsivo.

Os critérios de aceite para este requisito são:

A tela deverá se adaptar de acordo com a resolução do dispositivo.

O sistema deverá ser acessado pelos principais browsers do mercado.

• O sistema deve ser rápido.

É essencial evitar a espera e frustração dos usuários, quando estão acessando o sistema. Assim o sistema deverá ser rápido e atender às respostas do usuário sem deixá-lo aguardando um período longo.

Os critérios de aceite para este requisito são:

As requisições Ajax não poderão ultrapassar o tempo de 500 milissegundos.

O sistema deverá utilizar SPA (Single Page Application).

• O sistema deve ser simples para testar.

O sistema contará com testes unitários que irá validar os métodos da aplicação, mantendo assim, um código menos sujeitas a erros.

Os critérios de aceite para este requisito são:

Os testes devem ser executados de forma simplificada com apenas um comando.

Os testes deverão ser estruturados conforme as camadas, facilitando assim, as verificações.

• O sistema deve apresentar altos padrões de segurança.

As informações, registradas e guardadas, no sistema são consideradas patrimônios das empresas, assim é imprescindível tentar evitar qualquer tipo de acesso a informações sigilosas.

Os critérios de aceite para este requisito são:

O sistema deverá utiliza o protocolo HTTPS.

As senhas dos usuários deverão ser criptografadas.

**5.1.1 Visão lógica**

O sistema está organizado logicamente em camadas distintas com responsabilidades específicas. Cada camada comporta um conjunto de componentes e objetos que atendem os requisitos da aplicação.

A figura abaixo resume as camadas lógicas do sistema.

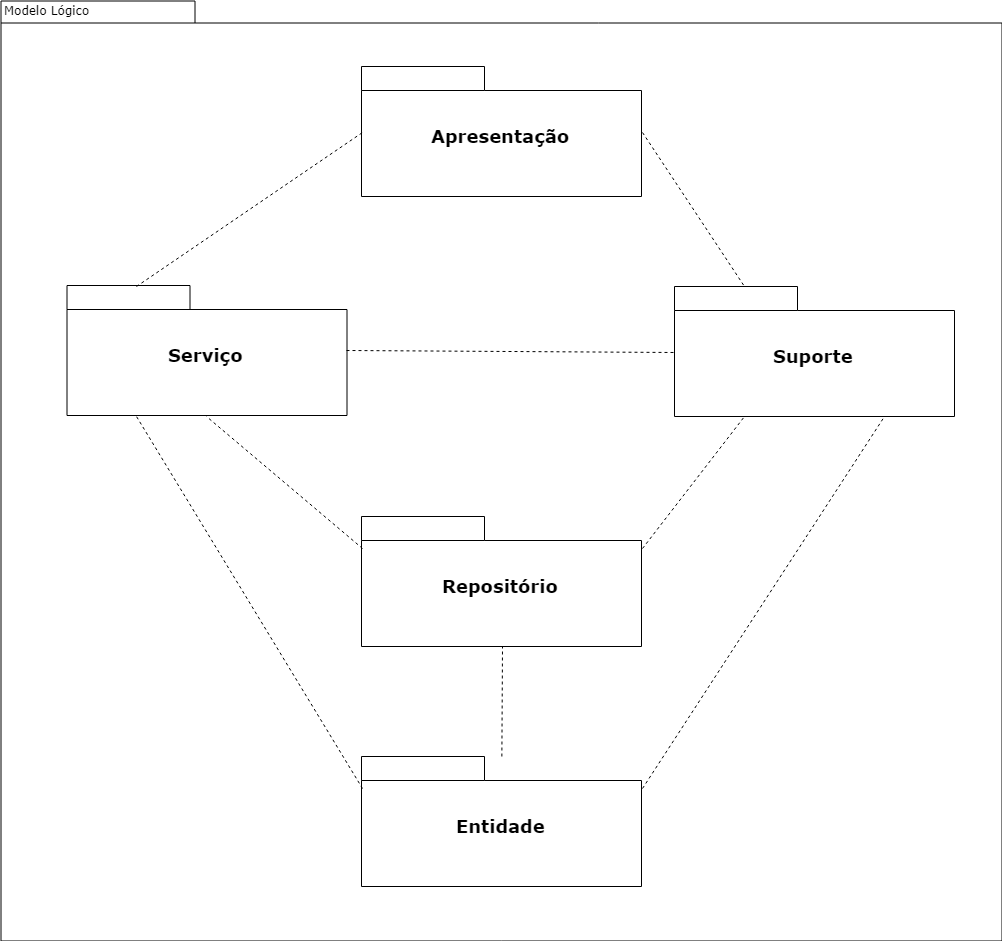


Figura 4.1 – Modelo Lógico

Camada de Apresentação

Essa camada reúne os componentes do sistema responsáveis pela interação dos usuários com o sistema. Os componentes dessa camada, em sua maioria, são controles visuais de apresentação e recuperação de informações por parte do usuário.

Camada de Serviços

Tem responsabilidade de encapsular as regras de negócio, criando um ponto central de compartilhamento de recursos e regras de negócio.

Essa camada se fez necessária para simplificar a interação entre os diversos recursos e diminuir o acoplamento.

Deverá ser projetada para prover o menor acoplamento possível com as demais camadas. Sendo assim, essa camada será implementada em dois módulos distintos, um contendo as interfaces dos componentes, e outro contendo a implementação dos componentes. As demais camadas deverão depender somente do módulo das interfaces dessa camada, e o mecanismo de inversão de controle deverá injetar as dependências.

Camada de Suporte

Nesta camada, serão colocados todos aqueles recursos e objetos da aplicação que não tiverem um padrão associado, e que sirvam somente como ajudantes (helpers) para os demais componentes da arquitetura. Um exemplo é o objeto de transporte de dados.

Camada de Entidades

Representa a camada do aplicativo que contêm as entidades da aplicação, bem como a troca de mensagens que compreendem o negócio do sistema.

Camada de Repositório

Todo o acesso ao banco de dados do sistema, será encapsulado das demais camadas como investimento em adaptabilidade e facilidade de manutenção do software.

Deverá ser projetada para prover o menor acoplamento possível com as demais camadas. Sendo assim, essa camada será implementada em dois módulos distintos, um contendo as interfaces dos componentes, e outro contendo as classes de implementação das interfaces. As demais camadas deverão depender somente do módulo das interfaces dessa camada, e o mecanismo de inversão de controle deverá injetar as dependências.

**5.1.2 Módulos compilados e Dependências**

Uma vez compilados os projetos que comportam o código fonte, deverão ser criados os seguintes módulos de software binários com as respectivas dependências de referência:

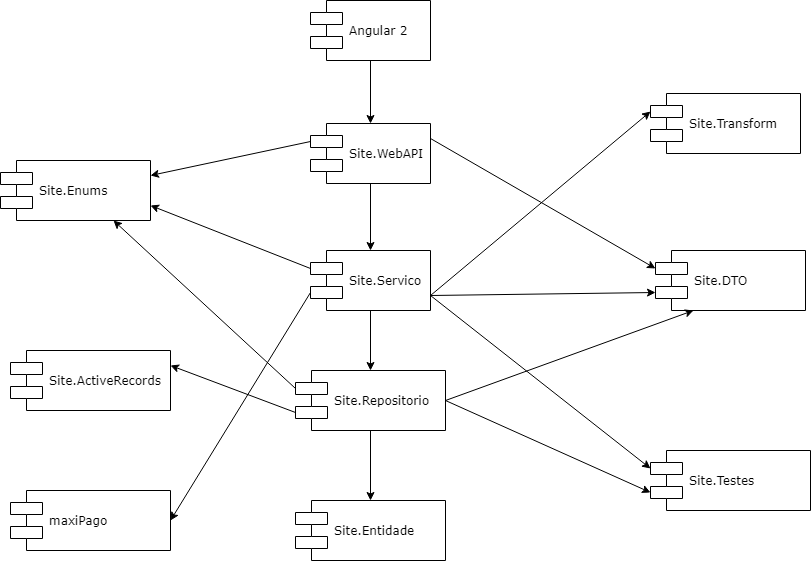


Figura 4.2 – Módulos e dependências

**5.2. Interfaces/ APIs**

**Sessão um: Métodos do Web Service do ViaCep**

Acessando o WebService de pesquisa de CEP

Para acessar o WebService, um CEP no formato de {8} dígitos deve ser fornecido, por exemplo: "01001000".

Após o CEP, deve ser fornecido o tipo de retorno desejado, que deve ser "json", "xml", "piped" ou "querty".

Exemplo de pesquisa por CEP:

https://viacep.com.br/ws/01001000/json/

Veja um exemplo de acesso ao WebService e o retorno:



Figura 5.1 – Exemplo do retorno da requisição ViaCEP

**Sessão dois: Gateway de Pagamentos da MaxiPago.**

Com a integração com os serviços da Maxipago, o sistema conta com uma solução de pagamentos, antifraude e conciliação financeira em uma única plataforma.

url de testes: https://testapi.maxipago.net/UniversalAPI/postXM

url de produção: https://api.maxipago.net/UniversalAPI/postXML

**Sessão três: Integração com a Receita para a emissão de CT-e.**

O Conhecimento de Transporte Eletrônico poderá ser utilizado pelos contribuintes do Imposto sobre Operações Relativas à Circulação de Mercadorias e sobre a Prestação de Serviços de Transporte.

Segue, abaixo, os serviços e suas respectivas url’s:

CteRecepcao: https://cte.fazenda.mg.gov.br/cte/services/CteRecepcao

CteRetRecepcao: https://cte.fazenda.mg.gov.br/cte/services/CteRetRecepcao

CteInutilizacao: https://cte.fazenda.mg.gov.br/cte/services/CteInutilizacao

CteConsultaProtocolo: https://cte.fazenda.mg.gov.br/cte/services/CteConsulta

CteStatusServico: https://cte.fazenda.mg.gov.br/cte/services/CteStatusServico

CteRecepcaoEvento: https://cte.fazenda.mg.gov.br/cte/services/RecepcaoEvento

**Sessão quatro: API para realização de agendamento de coletas por empresas parceiras.**

Url para consumir o serviço: https://projetoj.apphb.com/api/v1/transporte

Exemplo:

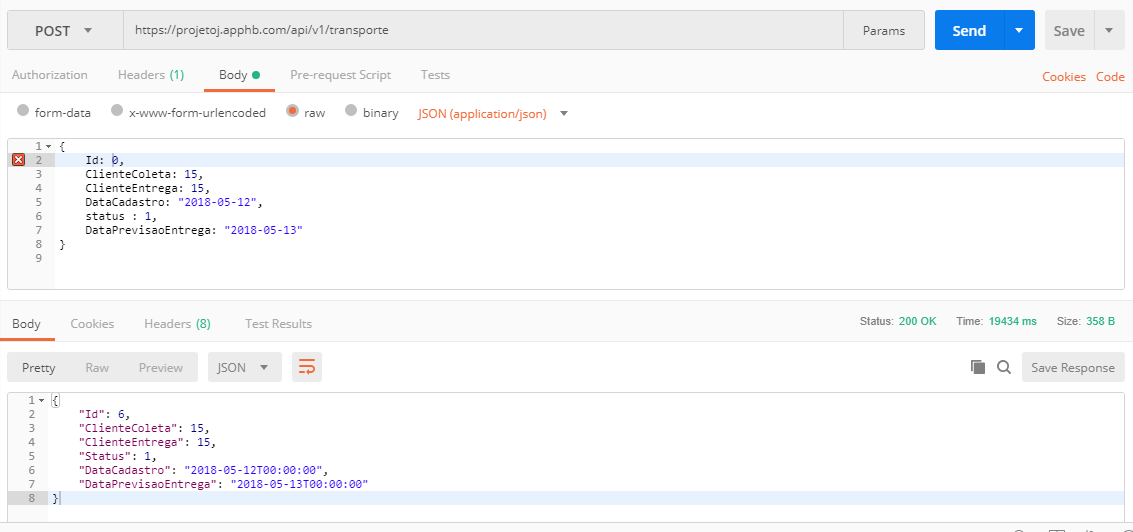


Figura 5.2 – Exemplo da requisição de agendamento

**6. Avaliação da Arquitetura**

**6.1. Análise das abordagens arquiteturais**

A visão lógica descreveu como o sistema é estruturado, na forma de unidades de implementação revelando as classes e estruturas que serão usadas para resolver os problemas de negócio do sistema, e de todos os aspectos que são relevantes para o entendimento do problema.

Os módulos compilados descrevem como o sistema é estruturado em tempo de execução. Essa visão é útil para pensarmos sobre os atributos de qualidade do sistema em tempo de execução, tais como desempenho e confiabilidade.

A visão de implantação descreve como o sistema é mapeado para o hardware. Endereça a parte física do sistema, sua organização física, os computadores, os periféricos e como eles se conectam entre si.

**6.2. Identificação dos atributos de qualidade**

Esta seção sumariza um conjunto de atributos que têm impacto na capacidade do software de manter seu nível de desempenho dentro de condições estabelecidas por um dado período de tempo. Espera-se que cada requisito elencado seja factível, necessário, priorizável, não ambíguo e verificável.

**6.3. Cenários**

**Cenário um:** Ao navegar na tela, o sistema deve apresentar boa usabilidade. A navegação dever apresentar facilidade e o acesso as funcionalidades devem ser bem objetivos para a função que precisa ser realizada, o usuário deve editar um agendamento de coleta sem muito esforço, assim garantindo a agilidade e a usabilidade para fica de acordo com um dos requisitos não funcionais.

**Cenário dois:** Ao realizar o acesso à aplicação através de um dispositivo móvel ou desktop com resolução reduzida, a tela do usuário deverá se adaptar automaticamente, redimensionando seus elementos de acordo com a resolução, estando de acordo com acessibilidade necessária para atender um dos requisitos não funcionais.

**Cenário três:** Ao realizar um acesso em qualquer tela, o sistema deve ter um desempenho aceitável e responder em no máximo 5 segundos a renderização da tela, mostrando um componente na tela informando que o processo está em andamento, e quando o processo finalizar esse componente deve ser escondido, assim atendendo um dos requisitos não funcionais.

**Cenário quatro:** Todo o tráfego na rede deverá ser feito utilizando o protocolo HTTPS. Evitando assim que o sniffing possa ser realizado no sistema para fins com propósitos maliciosos por invasores que tentam capturar o tráfego da rede com diversos objetivos. Além disso, todas as senhas dos usuários deverão ser criptografadas antes de serem salvas no banco de dados.

**Cenário cinco:** Os métodos do sistema deverão ser testados através de testes unitários que visam a diminuir o número de bug’s no software. Para testar a qualidade do código, precisamos, apenas, executar o comando pelo Visual Studio, assim todo o código será testado.

Os atributos foram priorizados e classificados nos quesitos “Importância” e “Complexidade” em alta (A), média (M) e baixa (B) de acordo com as características do requisito.

|  |  |  |  |
| --- | --- | --- | --- |
| **Atributos de Qualidade** | **Cenários** | **Importância** | **Complexidade** |
| Usabilidade | Cenário um: O sistema deve prover boa usabilidade. | M | B |
| Acessibilidade | Cenário dois: O sistema deve suportar ambientes Web responsivos e ambientes móveis. | M | M |
| Desempenho | Cenário três: O sistema deve ser rápido. | A | A |
| Segurança | Cenário quatro: O sistema deve apresentar altos padrões de segurança. | A | A |
| Testabilidade | Cenário cinco: O sistema deve ser simples para testar. | M | B |

**6.4. Avaliação**

A seguir serão apresentadas as evidências dos testes de avaliação. Apresentando as medidas registradas na coleta de dados.

* **Cenário um:**

|  |  |
| --- | --- |
| Atributo de Qualidade: | Usabilidade |
| Requisito de Qualidade: | O sistema deve prover boa usabilidade |
| Preocupação: | Manter um padrão no layout em todas as telas |
| Cenários(s): | Cenário um |
| Ambiente: | Sistema em operação normal |
| Estímulo: | Usuário inserindo/editando um agendamento de coleta no sistema |
| Mecanismo: | Telas simples e objetivas sem muitos componentes a serem carregados possibilitando o servidor de aplicação renderizar de forma rápida os objetos na tela do usuário. Os processos em AJAX permitindo que seja carregado de forma rápida evitando carregar a página inteira novamente e sim apenas o bloco onde se faz necessário alterar as informações. O sistema trabalha com template padrão para todas as telas, assim alterando apenas o conteúdo principal. |
| Medida de Resposta: | O usuário deverá realizar um agendamento em apenas alguns minutos (no máximo 3 minutos), devem mostrar ao usuário um componente que informe que o processo está em andamento. |
| Riscos: | Telas sem muitos estilos e efeitos, fornecendo um layout simplista. |
| Pontos de Sensibilidade: |  |
| Tradeoff: | Um layout performático exige elementos simples, sem muitos efeitos. |

**Evidências do cenário um:**

O sistema mantém um padrão de cores em todas as telas do sistema: azul para incluir, amarelo para editar e vermelho para excluir.

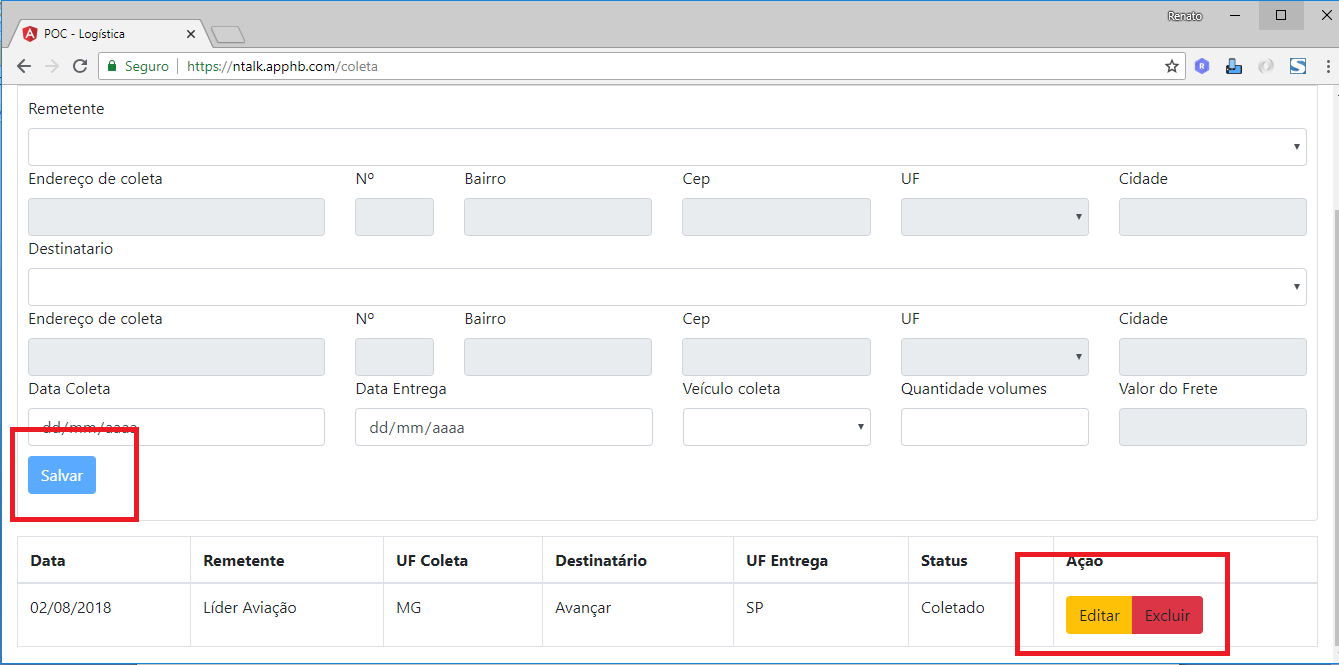


Figura 6.1 – Cores padrões para incluir, editar e excluir.

* **Cenário dois:**

|  |  |
| --- | --- |
| Atributo de Qualidade: | Acessibilidade |
| Requisito de Qualidade: | O sistema deve suportar ambientes Web’s responsivos e ambientes móveis. |
| Preocupação: |  |
| Cenários(s): | Cenário dois |
| Ambiente: | Sistema em operação normal |
| Estímulo: | Usuário acessando o sistema de vários dispositivos com tamanhos e layout’s diferentes. |
| Mecanismo: | Telas que se adaptam ao tamanho dos dispositivos, realocando os objetos na tela. |
| Medida de Resposta: | O usuário, acessando de um dispositivo móbile, consegue realizar todas as tarefas do sistema sem maiores dificuldades. |
| Riscos: | Alguns campos podem ficar maiores/menores dependendo do aparelho. |
| Pontos de Sensibilidade: |  |
| Tradeoff: | Não existe |

**Evidências do cenário dois:**

O sistema se adapta conforme o tamanho dos dispositivos, realocando os objetos na tela para uma melhor visualização.

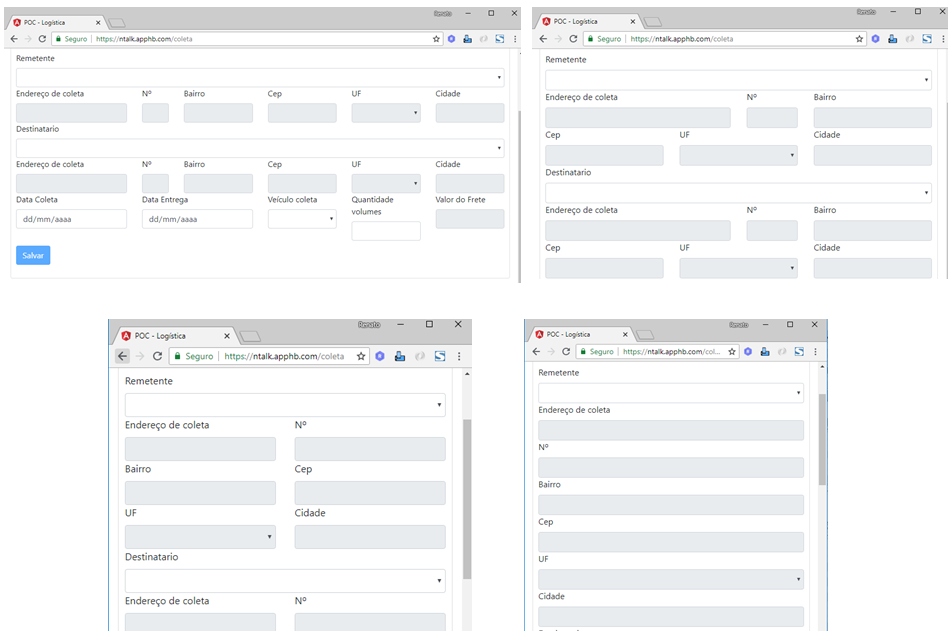


Figura 6.2 – Mesma página acessada por dispositivos de tamanhos diferentes.

* **Cenário três:**

|  |  |
| --- | --- |
| Atributo de Qualidade: | Desempenho |
| Requisito de Qualidade: | O sistema deve ser rápido |
| Preocupação: | Dar respostas rápidas aos usuários, não bloqueando a tela um longo período. |
| Cenários(s): | Cenário três |
| Ambiente: | Sistema em operação normal |
| Estímulo: | Carregar as páginas em até 5 segundos. |
| Mecanismo: | Telas simples sem muitos componentes visuais e apenas os dados indispensáveis para a tela. |
| Medida de Resposta: | A tela terá o carregamento total em até 5 segundos. |
| Riscos: | O tráfego da rede, velocidade da internet, são algumas das situações que não são possíveis de se ter o controle. |
| Pontos de Sensibilidade: | Acesso à internet |
| Tradeoff: | A velocidade do sistema exige que menos dados sejam carregado na tela. |

**Evidências do cenário três:**

Uma requisição ao servidor de aplicação no poderá ultrapassar o tempo de 500 milissegundos.

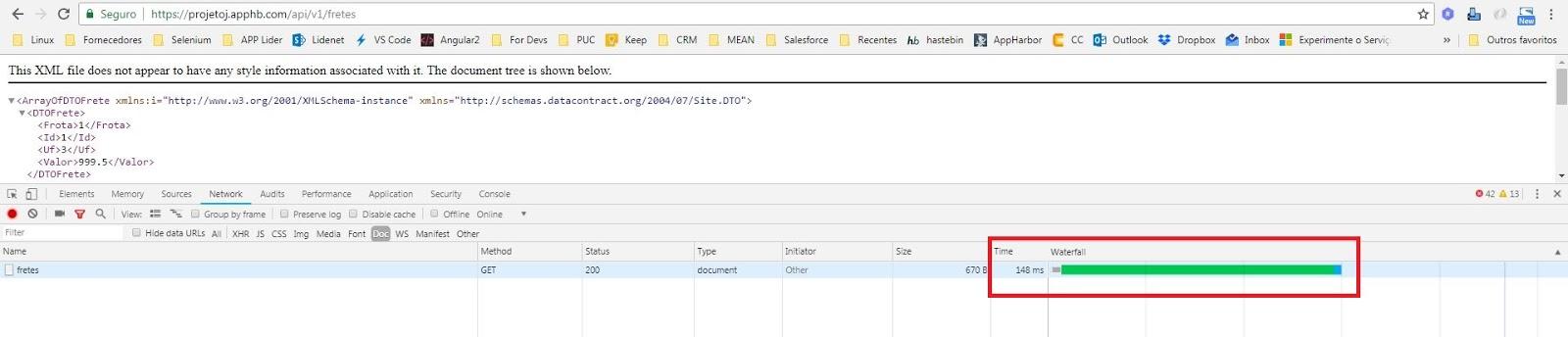


Figura 6.3 – Tempo de resposta de uma requisição.

* **Cenário quatro:**

|  |  |
| --- | --- |
| Atributo de Qualidade: | Segurança |
| Requisito de Qualidade: | O sistema deve apresentar altos padrões de segurança |
| Preocupação: | Manter as informações inseridas no sistema de forma segura. |
| Cenários(s): | Cenário quatro |
| Ambiente: | Sistema em operação normal |
| Estímulo: | Usuário acessando o sistema através do protocolo HTTPS. |
| Mecanismo: | Utilização de certificados digitais. |
| Medida de Resposta: | Todo o tráfego deverá ser criptografado. |
| Riscos: | Ataques de sniffing na rede. |
| Pontos de Sensibilidade: | Usuários acessando o sistema pelo protocolo HTTP. |
| Tradeoff: | Não existe |

**Evidências do cenário quatro:**

Todos os acessos ao sistema deverão ser pelo protocolo HTTPS e as senhas dos usuários deverão, ser gravadas no banco de dados, criptografadas.

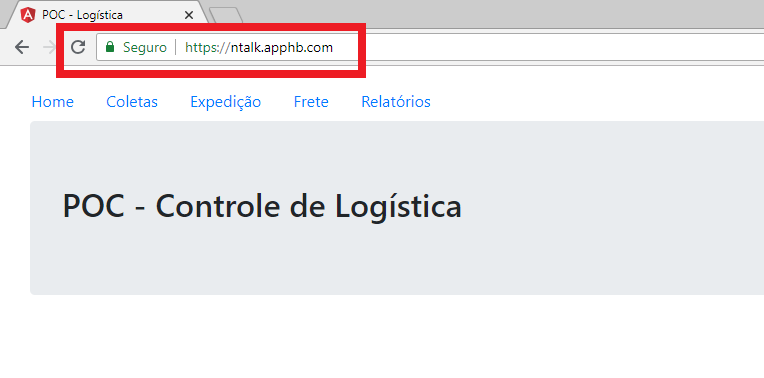


Figura 6.4 – Protocolo HTTPS sendo utilizado.

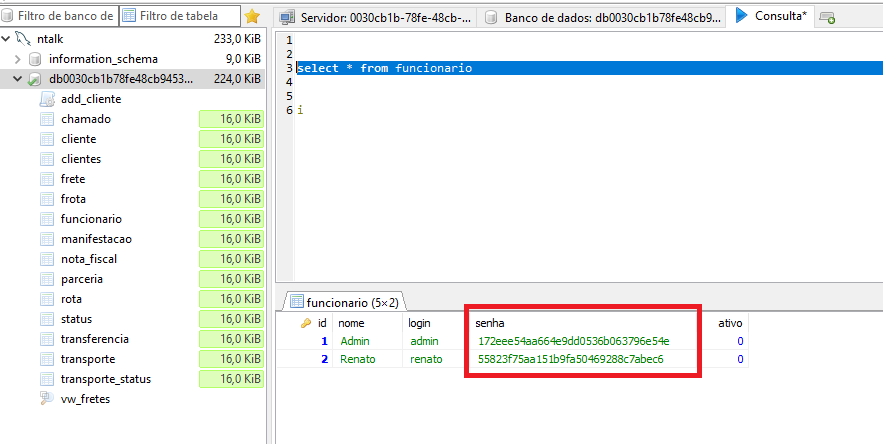


Figura 6.5 – Senhas criptografadas.

* **Cenário cinco:**

|  |  |
| --- | --- |
| Atributo de Qualidade: | Testabilidade |
| Requisito de Qualidade: | O sistema deve ser simples para testar. |
| Preocupação: | Manter a integridade do sistema ao realizar alterações no código. |
| Cenários(s): | Cenário cinco |
| Ambiente: | Sistema em operação normal |
| Estímulo: | Desenvolvedor realizando os testes. |
| Mecanismo: | Testes unitários do Visual Studio. |
| Medida de Resposta: | Testes executados com sucesso. |
| Riscos: | Os testes falharem. |
| Pontos de Sensibilidade: | Disciplina em manter os testes unitários atualizados |
| Tradeoff: | Tempo de desenvolvimento. |

**Evidências do cenário cinco:**

Todos os métodos do sistema são testados com apenas um comando.

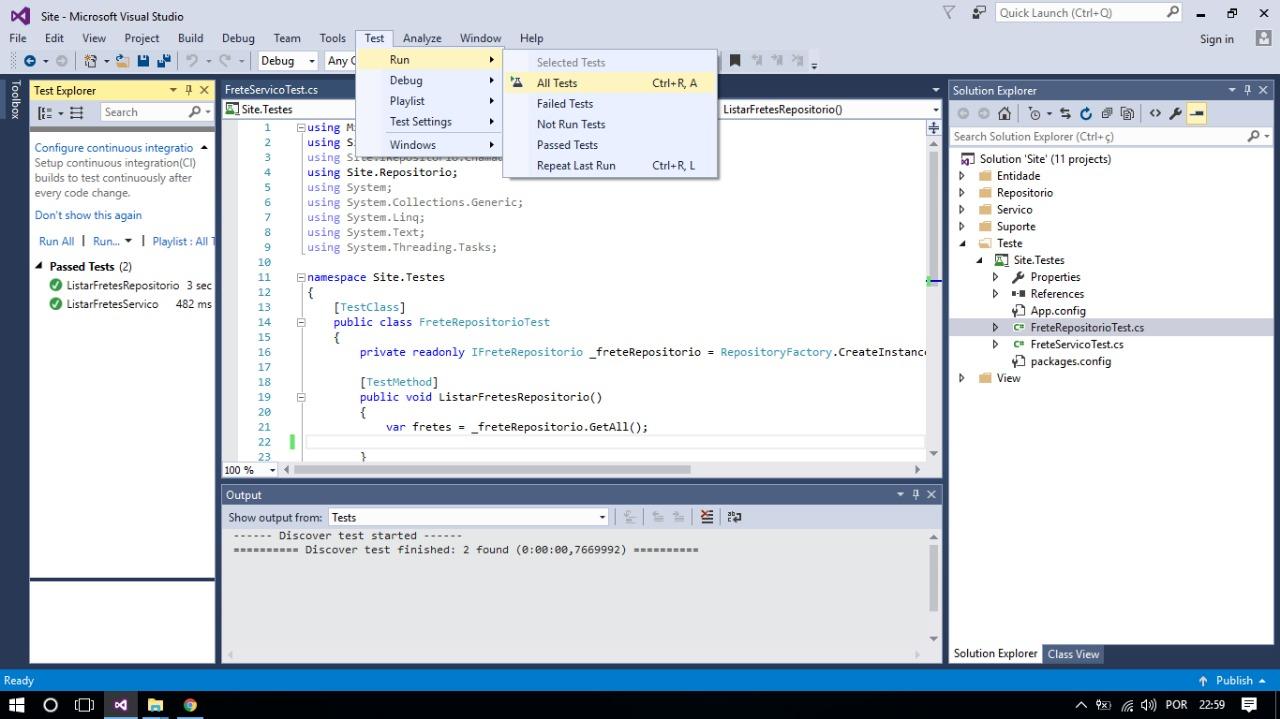


Figura 6.6 – Testes unitários sendo executados.

**6.5. Resultado**

Após validar os atributos de qualidade escolhidos, verifiquei que o projeto arquitetural atende às necessidades exigidas. Foram considerados os seguintes requisitos de qualidade abaixo:

|  |  |  |
| --- | --- | --- |
| **RFN** | **Testado** | **Homologado** |
| Segurança | SIM | SIM |
| Usabilidade | SIM | SIM |
| Acessibilidade | SIM | SIM |
| Performance | SIM | SIM |
| Testabilidade | SIM | SIM |

Dentro desta avaliação, vale ressaltar alguns pontos importantes. As páginas foram desenvolvidas de forma responsivas e em um módulo separado, que facilita a manutenção do sistema.

Os serviços podem ser consumidos através de uma APi desenvolvida de .Net/C#, separando, assim, as regras de negócio da aplicação.

Um ponto de limitação da arquitetura é o desenvolvimento com o framework Angular 2, pois exige uma curva de aprendizado maior em relação aos demais frameworks JavaScript, como o JQuery por exemplo.

Enfim, no geral, a arquitetura atendeu os requisitos exigidos para o desenvolvimento do sistema.

**7. Conclusão**

Este projeto apresentou uma arquitetura para um sistema de controle de logística. Foram apresentados pontos fortes e pontos que podem ser melhorados. No geral, a arquitetura se mostrou apta a atender aos requisitos exigidos.

**REFERÊNCIAS**

Página de documentação do Angular: https://angular.io/docs

CORDEIRO, Gilliard. CDI **Integre as dependencias e contextos do seu código Java**. ISBN: 978-85-66250-18-3, Casa do código, Data publicação: 08/2013

SAMY SILVA, Maurício. **Bootstrap 3.3.5 Aprenda a usar o framework Bootstrap para criar layouts CSS complexos e responsivos**. ISBN: 978-85-7522-460-1, Novatec, Data publicação: 10/2015

Página de documentação de WebAPI da Microsoft: https://docs.microsoft.com/en-us/aspnet/web-api/

Página de documentação do NGINX: https://docs.nginx.com/

**APÊNDICES**

URL do código Front End: https://github.com/renatokim/PocFrontEndControleLogistica

URL do código Back End: https://github.com/renatokim/SistemaControleLogistica

URL da aplicação Front End: https://ntalk.apphb.com/

URL da aplicação Back End: https://projetoj.apphb.com/api/v1

URL video Youtube: https://youtu.be/p8ASePhplmA