PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS

PUC Minas Virtual

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

Projeto Integrado

Relatório Técnico

Aisoftware Tracker

Sistema de controle de frotas

Igor Araújo

Belo Horizonte, Brasil
Agosto, 2022

Projeto Integrado – Arquitetura de Software Distribuído

Sumário

1. Introdução

Locomover-se e transportar cargas sempre foi uma necessidade humana, seja por terra, água ou ar. Na ultima década, principalmente no Brasil a frota de veículos terrestres vem aumentando vertiginosamente. Segundo dados do IBGE a frota de veículos terrestres no Brasil somando todas as categorias cresceu aproximadamente 35 % comparando os números absolutos entre 2010 e 2018, mostrando como é grande e promissor este mercado.

Em 2010 o número de veículos era de 64,8 milhões, já em 2018 este número subiu para mais de 100,7 milhões, em 2021, neste cenário adverso vivido pelo mundo em plena pandemia este número cresceu para 111,4 milhões de veículos.

Focando mais nos veículos de transporte terrestre como carros, caminhões, vans, e etc, temos números consideráveis também. Em 2010 havia cerca de 37,2 milhões de automóveis e 2,1 milhões de caminhões, e em 2021 estes números vão para 59,2 milhões de 2,9 milhões respectivamente.

Com esta crescente vários novos tipos de negócios são desenvolvidos, não só pela possibilidade de levar seus produtos para os mais diversos locais do país, mas também com serviços, além dos bens de consumo. Dado a este crescimento, mecanismos para gerenciar e cuidar destes veículos são cada vez mais necessários, tanto para pessoas físicas, com sistemas de rastreamentos e geolocalização, como principalmente para empresas. Onde podem ter uma visão geral de toda a sua frota, além de trazer mais segurança e informação logística.

O intuito deste projeto de rastreamento controle de frotas para empresas, ajudando na logística e localização em casos de roubo ou mal uso do veículo. Este serviço estará disponível, inicialmente, no mercado nacional, trazendo também um sistema de fácil operação, bonito e de interface limpa. Contendo o mínimo para realizar todas as operações necessárias, de forma direta e simples.,Dado objetivo de sere um sistema simples e direto com suas funcionalidades básicas, trás para o mercado uma maior competitividade e menor custo, refletindo também no preço final para o cliente. E neste documento serão apresentados os requisitos arquiteturais, funcionais e não funcionais e as diagramações da solução para o desenvolvimento da plataforma.

2. Cronograma do Trabalho

A seguir é apresentado o cronograma proposto para as etapas deste trabalho.

Datas		**Atividade / Tarefa**	**Produto / Resultado**
De	**Até**		
01/07/2022	01/07/2022	1. Cronograma do Trabalho	Construção desta tabela
02/07/2022	02/07/2022	2. Contextualização do trabalho	Construção da contextualização deste projeto
03/07/2022	05/07/2022	3. Definição dos requisitos Arquiteturais	Lista dos requisitos Arquiteturais identificados
03/07/2022	05/07/2022	4. Definição dos requisitos Funcionais	Lista dos requisitos funcionais identificados
03/07/2022	05/07/2022	5. Definição dos requisitos Não-funcionais	Lista dos requisitos Não-funcionais identificados
03/07/2022	05/07/2022	6. Definição dos Mecanismos Arquiteturais	Lista dos Mecanismos Arquiteturais identificados
06/07/2022	08/07/2022	7. Construção dos Diagramas de Contextos – Modelo C4	Diagrama de contexto criado no Draw.io e documentado
10/08/2022	10/08/2022	8. Revisão da Etapa 1	Documento Etapa 1 revisado
10/08/2022	10/08/2022	9. Construção do vídeo de apresentação da Etapa 1	Vídeo criado da Etapa 1
10/08/2022	10/08/2022	10. Apresentação em PPT da Etapa 1	PPT
10/08/2022	14/08/2022	11. Publicação no repositório Github Etapa 1	Arquivos produzidos no Github disponíveis abertamente
11/08/2022	15/08/2022	12. Construção dos Diagramas de Contêineres	Diagramas de contêineres
13/08/2022	15/08/2022	13. Construção dos Diagramas de Componentes	Diagramas de componentes
16/08/2022	18/08/2022	14. Desenho dos Wireframes da POC	Protótipos de telas de baixa fidelidade
19/08/2022	30/09/2022	15. Código da aplicação	Aplicação com 3 requisitos implementados
30/09/2022	30/09/2022	16. Publicação no repositório Github Etapa 2	Arquivos produzidos no Github disponíveis abertamente
01/10/2022	03/10/2022	17. Análise das abordagens arquiteturais	Seção do documento produzido
04/10/2022	05/10/2022	18. Cenários	Seção do documento produzido
06/10/2022	07/10/2022	19. Evidências da avaliação	Seção do documento produzido
08/10/2022	09/10/2022	20. Resultados obtidos	Seção do documento produzido
10/10/2022	11/10/2022	21. Avaliação crítica dos resultados	Seção do documento produzido
12/10/2022	13/10/2022	22. Conclusão	Seção do documento produzido

14/10/2022	19/10/2022	23. Construção do vídeo de apresentação da Etapa 3	Vídeo da etapa 3 disponível
20/10/2022	20/10/2022	24. Publicação no repositório Github Etapa 3	Arquivos produzidos no Github disponíveis abertamente

3. Especificação Arquitetural da solução

Esta seção apresenta a especificação básica da arquitetura da solução a ser desenvolvida, incluindo diagramas, restrições e requisitos definidos pelo autor, tal que permitem visualizar a macroarquitetura da solução.

3.1 Restrições Arquiteturais

Requisitos Arquiteturais são todos os aqueles, funcionais ou não funcionais, que têm impacto direto na arquitetura do sistema. A lista a seguir mostra os requisitos de arquitetura identificados para a implementação inicial.

- [RA01] - Deve ser usado tecnologias abertas (Open source) para o desenvolvimento de toda a plataforma.
- [RA02] - Deve ser usado o serviço de nuvem da Heroku como provedora da infraestrutura necessária para a plataforma.
- [RA03] - Deve ser usado o serviço oAuth0 JWT para autenticação, com a possibilidade inicial de criação de conta diretamente na plataforma por outro usuário que tenha acesso de ADM.
- [RA04] - Deve-se utilizar uma API de um terceiro, Trackmax, para ter as informações de geolocalização e informações da frota.

3.2 Requisitos Funcionais

Requisitos funcionais são todos os requisitos, recursos ou funções que são esperados no processo que o software pode satisfazer.

Em geral, requisitos funcionais representam ações que devem ser executadas pelo sistema, ou seja, requisitos funcionais são “o que o sistema deve fazer”

- [RF01] - Manter Usuários
- [RF02] - Manter Motoristas
- [RF03] - Manter Dispositivos (Veículos)
- [RF04] - Manter Grupos (Clientes)
- [RF05] - Consultar posicionamento de todos os dispositivos no Mapa por latitude e longitude
- [RF05] - Exibir veículos no mapa
- [RF05] - Exibir rota de veículo no mapa
- [RF05] - Exibir lista de veículos com informações basicas, posicionamento e último endereço de posicionamento.
- [RF06] - Consultar e Exportar Relatórios de Rotas
- [RF07] - Consultar e Exportar Relatórios de Eventos
- [RF08] - Consultar e Exportar Relatórios de Resumo

3.3 Requisitos Não Funcionais

Os Requisitos Não Funcionais estão associados às restrições de funcionalidades que **ditam como** o sistema deve fazer. A lista a seguir apresenta os requisitos não funcionais identificados para o desenvolvimento inicial da plataforma.

- [RNF01] - A plataforma deve habilitar a autenticação baseado no modelo JWT diretamente no sistema.
- [RNF02] - O sistema deve ser responsivo web.
- [RNF03] - O sistema deve ser responsivo para todos os tipos de tela (Computadores, Notebooks, Tablets e Smartphones).
- [RNF04] - O sistema deverá utilizar uma API de terceiro para realizar as persistências de informações dos dispositivos rastreadores.
- [RNF05] - O sistema deve operar 24x7.
- [RNF06] - Apenas usuários com permissão de Admin, tem permissão de criar e editar dados do sistema
- [RNF07] - Usuários operacionais, podem apenas acessar dados de leitura.

3.4 Mecanismos Arquiteturais

Os mecanismos arquiteturais representam conceitos técnicos fundamentais que serão padronizados por toda a solução. Eles são refinados durante o projeto em três estados, representados pelas três categorias de Mecanismos Arquiteturais:

- **Mecanismo de Análise**, que dá ao mecanismo um nome, uma descrição resumida e alguns atributos básicos derivados dos requisitos do projeto.
- **Mecanismo de Design**, que são mais concretos e assumem alguns detalhes do ambiente de implementação.
- **Mecanismo de Implementação**, que especifica a exata implementação de cada mecanismo.

Análise	Design	Implementação
Persistência	Micro ORM	Dapper
Persistência	Banco de Relacional	Postgres
Front end	Interface de comunicação com o usuário do painel	ASP.NET, Razor, html5, javascript, css
Front end	Navegador Web	Mozilla Firefox / Google Chrome
Teste de Software	Testes Unitários	XUnit
Autenticação	OAuth0 JWT	Microsoft.AspNetCore. Authentication.JwtBearer
Tratamento de exceções	Camada para tratar as exceções criando interações diferentes para usuários e técnicos	ASP.NET e C#
Build	Linha de comando	Terminal utilizando CLI ASP.NET
Build	Via Pipeline	Heroku pipeline
Deploy	Pipeline	Heroku / Github

4. *Modelagem Arquitetural*

Esta seção apresenta a modelagem arquitetural da solução proposta, de forma a permitir seu completo entendimento visando à implementação da Prova de Conceito (PoC) da plataforma Aisoftware Tracker na seção 5.

Para esta modelagem arquitetural optou-se por utilizar o modelo C4 para documentação de arquitetura de software. Mais informações a respeito podem ser encontradas aqui: https://c4model.com/ e aqui: https://www.infoq.com/br/articles/C4-architecture-model/. Dos quatro níveis que compõem o modelo C4 três serão apresentados aqui e somente o Código será apresentado na próxima seção (5).

4.1 *Diagrama de Contexto*

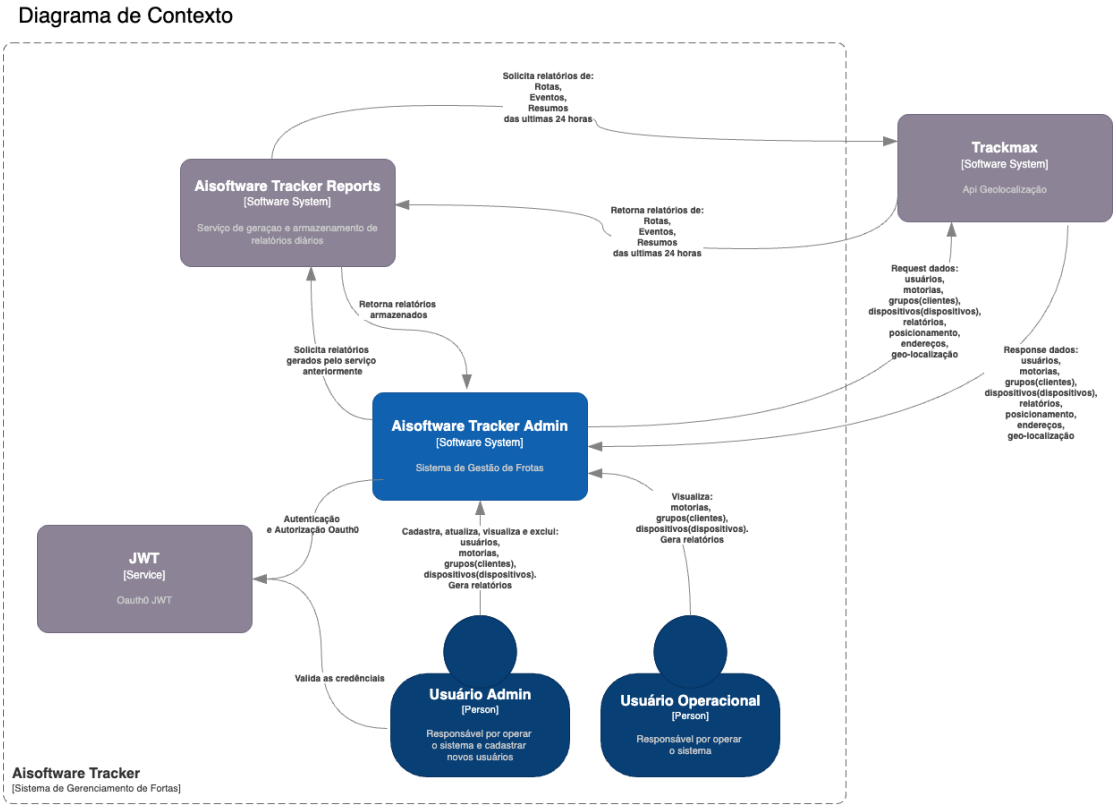

Figura 1 - Visão Geral da Solução Aisoftware Trecker

A figura 1 mostra a especificação o diagrama geral da solução proposta, com todos seus principais sistemas e pessoas envolvidas.

4.2 Diagrama de Container

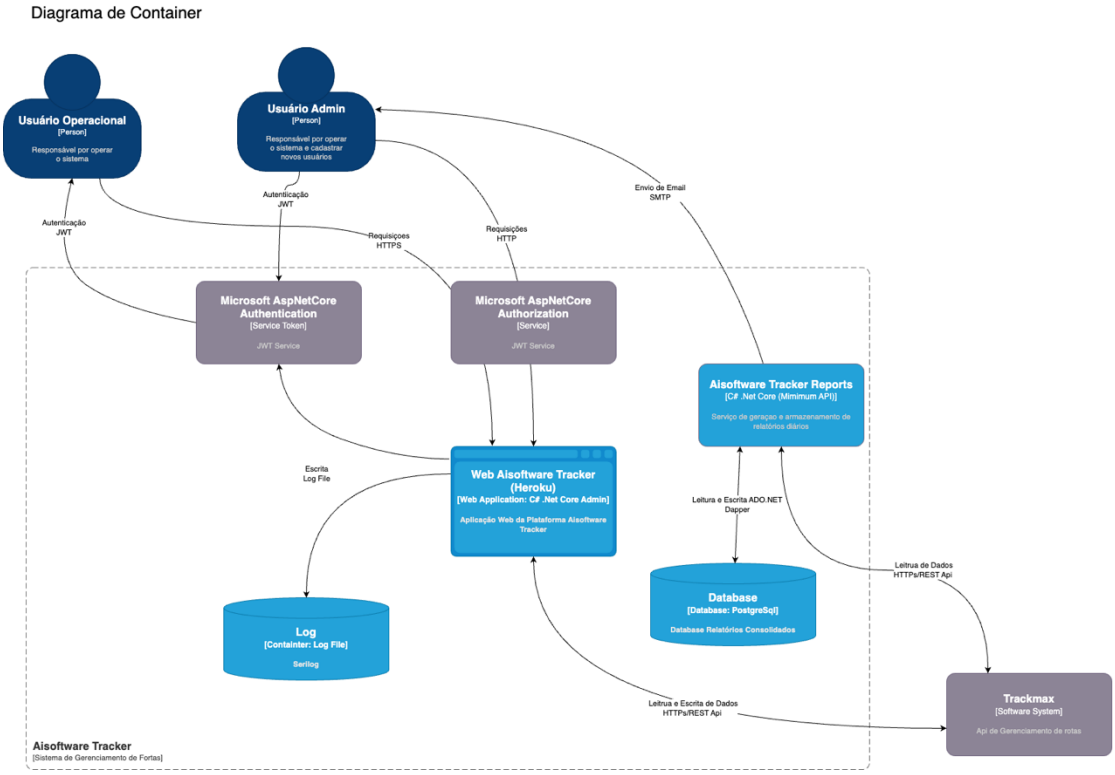

Figura 2 – Diagrama de Containter Aisoftware Trecker

A Figura 2 apresenta os containers da aplicação e a forma como estão organizados, a arquitetura visa manter somente uma fonte de verdade tratando-se dos dados da aplicação como um todo, a fonte de dados oficial é terceira API Trackmax.

O serviço de “Report” consome da API terceira e consolida relatórios no banco de dados para que posteriormente estes relatórios seja enviados via e-mail para os devidos clientes, dado ao seu volume de informações.

Toda manipulação de autenticação e autorização é feito pelo serviços JWT token alocado dentro da aplicação principal que também gerencia os cookies de comunicação com a API terceira.

A rastreabilidade de erros e inconsistências são monitoradas pelos logs do Serilog em arquivos no server e pelo Heroku Metrics.

4.3 Diagrama de Componentes

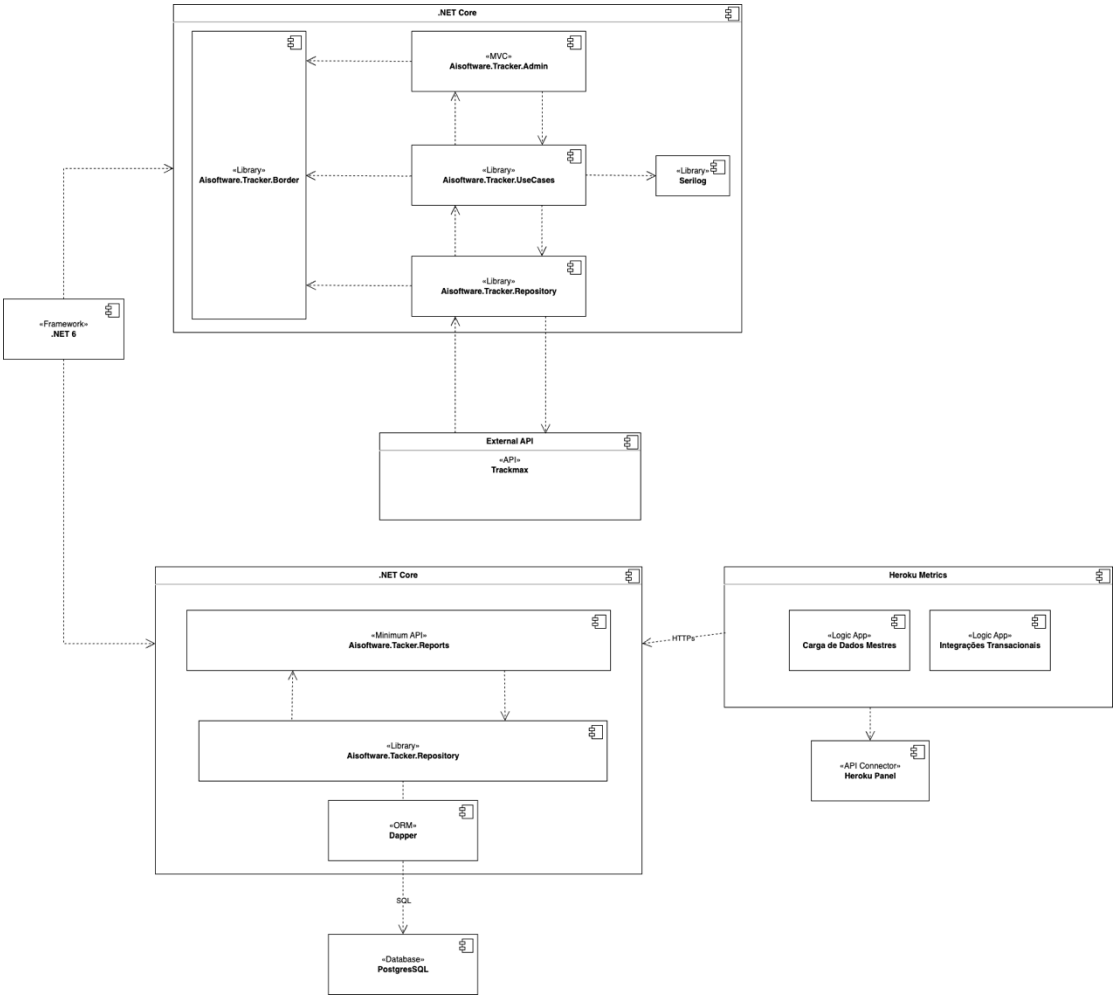

Figura 3 – Diagrama de Componentes Aisoftware Trecker

A Figura 3 apresenta os componentes da aplicação e estes podem ser detalhados como:

- Net 6 – Framework de desenvolvimento de sistemas WEB.
- Aisoftware.Tracker.Admin – Aplicação que disponibiliza a UI para utilização dos usuários.
- Aisoftware.Tracker.UseCases – Lib responsável por conter as regras de negócio da aplicação.
- Aisoftware.Tracker.Repository – Lib responsável pela comunicação externa com as APIs de terceiros.
- Aisoftware.Tracker.Reports – Aplicação responsável por gerar relatórios e os envia para determinados usuários em horários específicos via SMTP
- Dapper – Micro ORM responsável pela comunicação e interações com o banco de dados.
- Database – Banco de dados responsável por armazenar dados de dos relatórios.

- External Api – Api externa que fornece informações estratégicas para o funcionamento da aplicação.
- Serilog – Serviço responsável por gerar e escrever logs da aplicação.
- Heroku Metrics – Serviço externo que traz informações importantes da vida do container.
- Heroku Panel – Painel de demonstração das métricas coletadas do Heroku.

4.4 *UML do Banco de Dados*

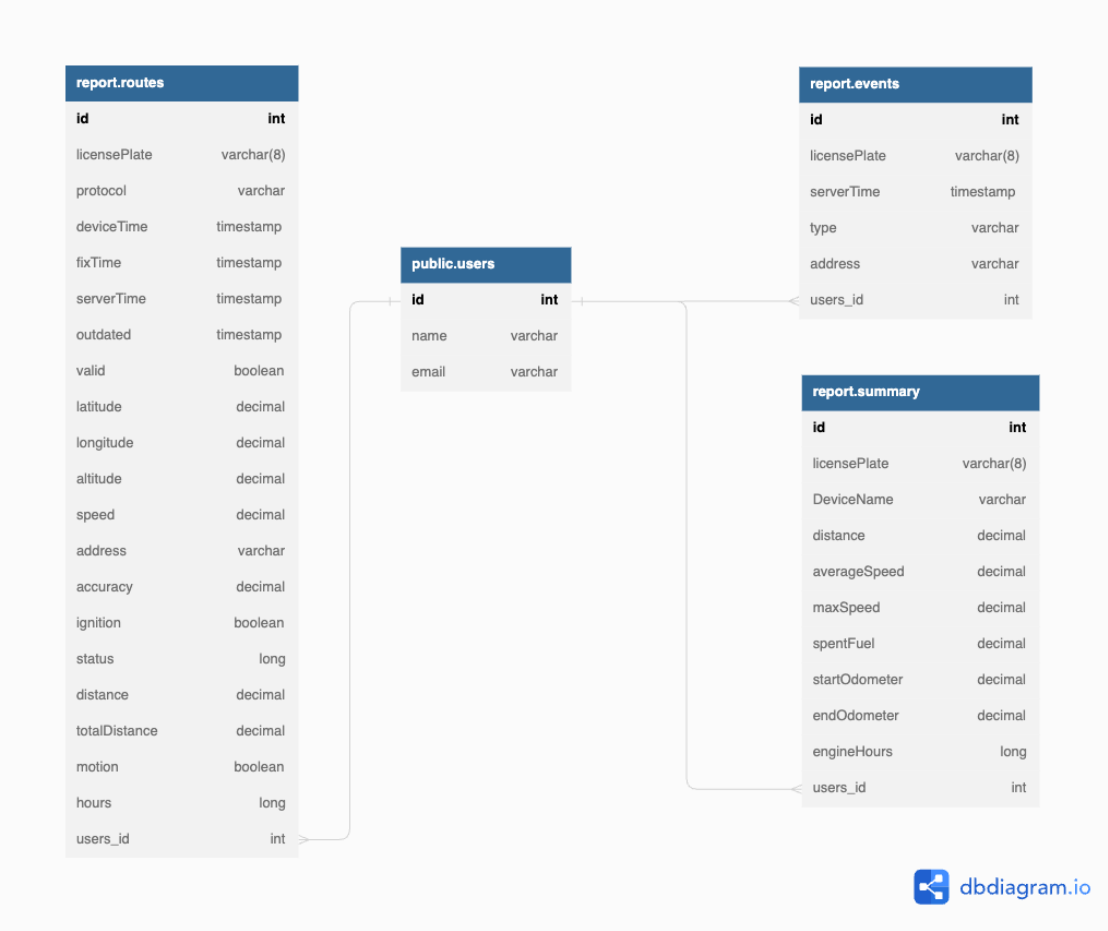

Figura 4 – UML do banco de dados

A figura 4 tem como objetivo mostrar o UML do banco de dados relacionado ao serviço de relatórios, onde o objetivo do mesmo é armazenar o consolidado dos relatórios gerados.

5. POC (Prova de Conceito)

Para validar algumas implementações propostas para aplicação foram utilizadas provas de conceito com alguns pontos importantes como SOLID, para um código mais limpo e coeso. E a implementação de Design Patterns que ajudam nesta construção como Singleton, Repository, Injeção de Dependência, AbstractFactory.

5.1 Wireframe

Para apresentação da aplicação foi criado um Wireframe navegável com a ferramenta Miro, onde o mesmo está disponível no seguinte link: https://miro.com/app/board/uXjVPeGQ-7Y=/?share_link_id=352327279950

Onde, para navegar basta clicar nas setas ( ) dentro dos componentes da tela.

5.2 Código da aplicação

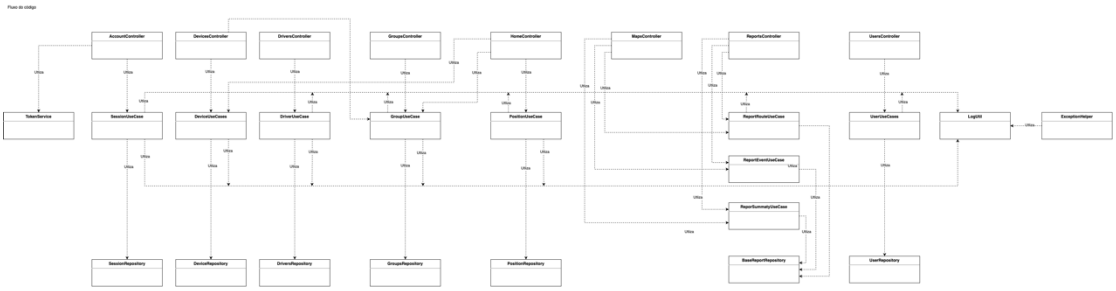

Figura 5 – Código da aplicação

A figura 5 demonstra como estrutura da aplicação foi organizada e como os componentes implementados se relacionam.

- Os componentes implementados são divididos em:
- Controllers: Responsável por intermediar as requisições enviadas pelas Views com as respostas fornecidas pelas models (dentro do Border)
- UseCases: Responsável por concentrar toda a regra de negócio do sistema.
- Repositories: Responsável por realizar a consulta e persistência de dados consumindo uma API
- Borders: Responsável por ter os objetos comuns entre as camadas, sendo a única camada conhecida por todas as outras. Onde neste modelo está os objetos de manipulação de exception e log.

Link aplicação: https://aisoftware-tracker-admin.herokuapp.com/

O código da aplicação está no seguinte repositório no github:

https://github.com/igoraujo/tcc-pos-graduacao#etapa-02

Menu:

- Código da aplicação

Etapa 3 - Pendente

<Conteúdo a ser produzido – Data final 15 de dezembro>

Referências

LOCALIZA. **Evolução da frota de veículos no Brasil.**
Disponível em: https://frotas.localiza.com/blog/frota-de-veiculo.
Acesso em: 10 de agosto de 2022.

IBGE. **Frota de Veículos.**
Disponível em:
https://cidades.ibge.gov.br/brasil/pesquisa/22/28120?indicador=28122&tipo=grafico.
Acesso em: 10 de agosto de 2022.

UNISEPE. **O transporte rodoviário movimentando o Brasil.**
Disponível em: https://portal.unisepe.com.br/blog-unisepe/o-transporte-rodoviario-movimentando-o-brasil/.
Acesso em: 10 de agosto de 2022.

SUSEP. **IVR - Índice de veículos roubados.**
Disponível em:
http://www2.susep.gov.br/menuestatistica/RankRoubo/resp_menu1.asp.
Acesso em: 10 de agosto de 2022.

MINISTÉRIO DA INFRAESTRUTURA. **Frota de veículos - 2021.**
Disponível em: https://www.gov.br/infraestrutura/pt-br/assuntos/transito/conteudo-Senatran/frota-de-veiculos-2021.
Acesso em: 10 de agosto de 2022.