CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

FACULDADE DE TECNOLOGIA DA PRAIA GRANDE

CURSO SUPERIOR DE TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

CAIO RODRIGO DE OLIVEIRA
GUILHERME MATOS SANTANA
GUSTAVO FERNANDES REIS

LIVE SAVER LOCATOR:
SISTEMA DE CHAMADO E CONTROLE DE AMBULÂNCIAS

PRAIA GRANDE
2024

CAIO RODRIGO DE OLIVEIRA
GUILHERME MATOS SANTANA
GUSTAVO FERNANDES REIS

LIVE SAVER LOCATOR:
SISTEMA DE CHAMADO E CONTROLE DE AMBULÂNCIAS

Relatório Técnico Científico apresentado à Faculdade de Tecnologia da Praia Grande, no Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, 6º ciclo, como Trabalho de Conclusão de Curso de Graduação.

Orientador: Jose Augusto Theodosio Pazetti.

PRAIA GRANDE
2024

Oliveira, Caio
Santana, Guilherme
Reis, Gustavo

Live Saver Locator: Sistema de Chamados e Controle de Ambulâncias / Caio Rodrigo de Oliveira; Guilherme Matos Santana; Gustavo Fernandes Reis – Praia Grande. - 2024.
65f.: 25 il.

Relatório Técnico (Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas) – Centro Estadual de Educação Tecnológica Paula Souza - Faculdade de Tecnologia da Baixada Santista (FATEC Praia Grande), 2024.

Orientador: Jose Augusto Theodosio Pazetti.

1. Sistema de localização em tempo real. 2. Controle de chamados. 3. Aplicação web. 4. Atendimento pré-hospitalar. I Título. II Centro Estadual de Educação Tecnológica Paula Souza – FATEC Praia Grande.

"A tecnologia é melhor quando reúne as pessoas."
-Grupo Ecomp

AGRADECIMENTOS

A realização deste Relatório Técnico só foi possível graças ao apoio e à colaboração das pessoas às quais expresso meus sinceros agradecimentos. Agradeço à minha família, em especial aos meus amigos e integrantes do grupo, pelo apoio incondicional, pela parceria, troca de ideias, momentos de descontração que tornaram essa jornada mais leve e prazerosa e por sempre acreditarem no meu potencial.

Agradeço ao meu orientador, Jose Augusto Theodosio Pazetti, pela orientação, paciência, e por compartilhar seu vasto conhecimento. Sua orientação foi fundamental para o desenvolvimento deste trabalho, sem ele este relatório não teria êxito ou a excelência necessária para a conclusão.

Agradeço também à Prefeitura Municipal de São Vicente, pela oportunidade de estágio e por fornecer os recursos e o ambiente necessários para a realização das pesquisas e práticas fundamentais para este trabalho.

Por fim, a todos que de alguma forma contribuíram para a realização deste trabalho, meus mais sinceros agradecimentos.

RESUMO

O presente trabalho aborda o desenvolvimento e implementação de um sistema *web*[1] de chamado e controle de ambulâncias em tempo real denominado *Live Saver Locator*. O sistema foi desenvolvido para otimizar o controle de chamados de ambulâncias, fornecendo informações sobre os chamados e a localização das ocorrências. O estudo apresenta a importância no atendimento de emergências e como a tecnologia pode ser aplicada para aprimorar esse processo. O *Live Saver Locator* utiliza tecnologias como GPS (*Global Positioning System*)[2] e sistemas de comunicação para rastrear a localização das ambulâncias e disponibilizar essas informações em tempo real para os hospitais que pretendem utilizar o serviço. O Relatório Técnico aborda detalhadamente como foram utilizadas as tecnologias e os desafios enfrentados no desenvolvimento do sistema, incluindo questões relacionadas à precisão da localização. Além disso, são abordadas as etapas de implementação e avaliação do sistema, destacando suas funcionalidades e benefícios como tempo de retorno da ambulância até o hospital, classificados em a caminho, retornando e finalizado, *status*[3] da emergência definidos em grave, urgente e muito urgente, *status* da ambulância categorizado como: disponíveis, ocupadas e inativas para a prestação de serviços de saúde. Por fim, são apresentadas conclusões sobre a eficácia e eficiência do *Live Saver Locator* na melhoria dos serviços do atendimento pré-hospitalar.

Palavras-chave: Chamados; Atendimento; Ambulâncias; Tecnologias.

[1] Traduzido como teia, é o nome dado a internet que conecta os computadores de todo o mundo na rede mundial.

[2] Global Positioning System traduzindo sistema de localização global são importantes instrumentos de localização geográfica e muito utilizados para a localização de pontos específicos da superfície terrestre.

[3] Estado da ocorrência.

ABSTRACT

The present work addresses the development and implementation of a real-time ambulance call and control system called Live Saver Locator. The system was developed to optimize ambulance call control by providing information about calls and the location of incidents. The study highlights the importance of emergency care and how technology can be applied to enhance this process. The Live Saver Locator utilizes technologies such as GPS (Global Positioning System) and communication systems to track the location of ambulances and provide this information in real-time to hospitals intending to use the service. The Technical Report extensively discusses the technologies employed and the challenges faced in system development, including issues related to location accuracy and data security assurance. Additionally, the implementation and evaluation stages of the system are addressed, highlighting its functionalities and benefits such as ambulance response time categorized as en route, returning, and completed; emergency status divided into severe, urgent, and very urgent; ambulance status categorized as available, occupied, and inactive for health service provision. Finally, conclusions are drawn regarding the effectiveness and efficiency of the Live Saver Locator in improving pre-hospital care services.

Keywords: Calls; Care; Ambulances; Technologies.

LISTA DE ABREVIAÇÕES E SIGLAS

HTTP	Hypertext Transfer Protocol
SAMU	Serviço de Atendimento Móvel de Urgência
GPS	Global Positioning System
APH	Atendimento Pré-Hospitalar
RCP	Reanimação Cardiopulmonar
JPA	Java Persistence API
API	Application Programming Interface
ORM	Object Relational Mapper
HTML	Hypertext Markup Language
MVC	Model, View, Controller
JWT	JSON Web Token
SCSS	Sassy CSS
IDE	Integrated Development Environment
CSS	Cascading Style Sheets
WAR	Web Application Archive
SDK	Software Development Kit
UML	Unified Modeling Language
OMG	Object Management Group
SGBD	Sistema de Gerenciamento de Banco de Dados

LISTA DE FIGURAS

SUMÁRIO

1. INTRODUÇÃO

Este trabalho descreve o desenvolvimento de um projeto de aplicação *web* para um Sistema de Chamados e Controle de Ambulâncias, como parte dos requisitos para a conclusão do curso de graduação em Análise e Desenvolvimento de Sistemas na Faculdade de Tecnologia de São Paulo — Unidade Praia Grande.

A partir do conhecimento adquirido no curso, o objetivo foi desenvolver uma aplicação que auxiliasse e otimizasse um serviço existente. Nesse sentido, foi desenvolvido o *Live Saver Locator*, uma aplicação que realizasse o controle de chamados de ambulâncias, centralizando-os com as solicitações efetuadas às ocorrências em tempo real. Esta aplicação *web* é um *software*[4] que oferece interfaces gráficas simples e fáceis para serem utilizadas pelos usuários, é hospedada e executada em um servidor remoto, seguindo o paradigma Cliente/Servidor suportado pelo protocolo de comunicação *HTTP*[5].

O resultado apresenta um projeto técnico, desenvolvido com a ajuda e auxílio do ponto de vista de profissionais da área da saúde, foi realizada uma entrevista anônima conforme apresentado no apêndice B que descrevia não ter um sistema capaz de alertar a chegada das ambulâncias no hospital, assim como a situação da ocorrência. Foi criado um formulário online que obteve como resultado respostas semelhantes para esta situação em que pode ser conferido no apêndice A. Diante dessas necessidades o projeto foi baseado para suprir essas demandas, foram elaboradas duas interfaces de usuário, uma para o hospital e outra para o motorista que será responsável pela ambulância.

No acesso realizado pelo hospital é possível ter uma visão ampla das ambulâncias, o trajeto que será percorrido até o local do acidente, o tempo estimado de retorno da ambulância até o hospital, as notificações de atualizações no estado das ambulâncias e a disponibilidade delas sendo disponíveis quando não estão atendendo algum chamado, ocupadas quando estão atendendo alguma ocorrência ou inativas quando estão passando por manutenção ou algum problema no caminho durante o atendimento. A interface do motorista no sistema foi planejada para ser de rápida funcionalidade, com ícones e funções simples, sendo elas a notificação ao motorista relacionado a ambulância do chamado, a visualização do local e da rota que poderá ser percorrida, a situação da ocorrência, descrição da emergência e ações de atualização da situação da ocorrência a partir da ambulância.

O desenvolvimento de uma aplicação *web* desse tipo requer uma análise das necessidades do cliente e dos usuários finais, juntamente com um planejamento detalhado dos requisitos

[4] Programas e dados que dizem a um computador como executar tarefas específicas.

[5] Hypertext Transfer Protocol (Protocolo de Transferência de Hipertexto em português), é um protocolo de comunicação que permite a transferência de dados na internet.

técnicos e funcionais. Isso inclui a utilização de ferramentas de desenvolvimento *web*, como linguagens de programação, *frameworks*[6] e bancos de dados.

O interesse em desenvolver este sistema se surge do fato de melhorar e monitorar o atendimento em casos de emergência, graças ao uso da tecnologia que proporciona saber a localização exata de uma ambulância, e no cotidiano possivelmente auxiliar no atendimento aos chamados.

O trabalho apresenta os benefícios do sistema e explica o funcionamento de cada componente, assim como demonstra como é desenvolvido o controle e gerenciamento das ambulâncias. Por fim, este estudo busca contribuir para o entendimento e a valorização dos sistemas de localização de ambulâncias como uma ferramenta essencial na modernização dos serviços de emergência médica. Através de uma análise, são esperados *insights*[7] importantes para profissionais da saúde, gestores de serviços de emergência e pesquisadores interessados em melhorar a eficácia e eficiência na qualidade do atendimento em situações críticas.

[6] Frameworks são estruturas de suporte definidas que fornecem funcionalidades pré-moldadas para facilitar o desenvolvimento de software.

[7] Insights são percepções repentinas e profundas que surgem quando uma pessoa compreende algo novo, faz uma conexão relevante ou adquire uma compreensão mais profunda de uma situação, problema, pessoa, ideia ou padrão.

2. DEFINIÇÃO DOS PROBLEMAS E SUAS LIMITAÇÕES

Atualmente, existem maneiras de solicitar atendimento de ambulâncias para atender ocorrências, uma delas é realizada pelo Serviço de Atendimento Móvel de Urgência (SAMU), cujo seu objetivo é chegar o mais rápido possível à vítima, prestando atendimento médico de qualidade no local da ocorrência. De acordo com o Ministério da Saúde, o atendimento do SAMU começa com uma ligação para o 192. É informado a situação da vítima e a localização. Um técnico coleta as informações e as envia para o Médico Regulador, que avalia a situação e decide o melhor atendimento. Ele pode orientar por telefone ou enviar unidades móveis como ambulâncias. A partir do atendimento da solicitação, o médico regulador comunica aos hospitais a emergência com a intenção de reservar leitos para que a ocorrência tenha continuidade. (MINISTÉRIO DA SAÚDE, 2024).

Nesse cenário, o acompanhamento da ambulância à ocorrência da parte do hospital não disponibiliza um controle de local e tempo para atendimento da solicitação, podendo resultar também em inconsistência de informações referente às ocorrências. Buscando solucionar essas limitações, a implementação de uma aplicação *web* que efetue esse controle pode ser uma solução viável. Esta aplicação possibilita manter um acompanhamento da ocorrência, além de manter uma comunicação hospital-ambulância.

2.1 DEFINIÇÕES DO PROBLEMA

Um sistema robusto e seguro que integre diferentes tecnologias, como GPS (*Global Positioning System*)[8], comunicação em tempo real e prontuários eletrônicos, exige expertise técnica e investimentos consideráveis. A segurança de dados dos pacientes também é uma grande preocupação que precisa ser cuidadosamente planejada e implementada.

Após uma entrevista informal, encontrada no apêndice B, realizada de forma anônima com um profissional da saúde do Hospital e Maternidade Mongaguá, o SAMU da região enfrenta desafios que afetam a qualidade do atendimento pré-hospitalar. A falta de acompanhamento em tempo real da ambulância pode afetar a distribuição de profissionais no hospital para atendimento das ocorrências. A comunicação ineficiente entre o hospital e a equipe médica da ambulância e dificuldade em monitorar o desempenho impactam negativamente a eficiência do serviço. Um caso citado por uma auxiliar de limpeza do local que participou da entrevista, foi de um paciente que chegou no local e o hospital não possuía profissionais da saúde capacitados para aquele tipo de ocorrência, sendo designado para outro hospital. Ao desenvolver e implementar um *software web* torna-se um desafio por ser exigido adotar um novo processo de atendimento de solicitações de ambulâncias para que as ferramentas da aplicação sejam aproveitadas ao máximo atendendo e aprimorando as solicitações.

[8] Sistema de Posicionamento Global.

2.2 LIMITAÇÕES DO PROBLEMA

Custos de desenvolvimento, treinamento de pessoal e infraestrutura. É importante garantir a viabilidade econômica do projeto e a sustentabilidade a longo prazo, considerando os recursos disponíveis e o impacto no orçamento da saúde pública.

A introdução de um novo sistema exige adaptação por parte dos profissionais de saúde, que podem ter receio de mudanças em suas rotinas. Treinamentos eficazes e comunicação transparente são essenciais para garantir a adesão ao novo sistema e maximizar seus benefícios.

2.3 SOLUÇÃO E HIPÓTESE

Um sistema integrado e inteligente para o atendimento pré-hospitalar, com recursos como GPS, comunicação em tempo real e integração com prontuários eletrônicos, pode otimizar significativamente o atendimento.

Acompanhar em tempo real a localização da ambulância e o progresso da equipe médica, além de garantir comunicação eficiente e segura entre hospital, equipe e paciente, são alguns dos benefícios.

A otimização do despacho de ambulâncias e a redução de atrasos proporcionam um aumento da eficiência do serviço e reduzir os custos. A redução do tempo resposta às solicitações pode resultar em uma melhoria na qualidade do atendimento.

2.4 OBJETIVO GERAL

O objetivo geral foi desenvolver uma aplicação de controle e chamado pré-hospitalar. Nesse sentido, foi produzido o *software Live Saver Locator*, uma aplicação *web* para o controle e chamados de ambulâncias, integrando-os com as solicitações das ocorrências. Com o uso desse sistema, é possível armazenar dados das atividades efetuadas no pré-atedimento dos acidentes, podendo resultar na facilitação do levantamento de dados do serviço solicitado. O *software* também permite o acompanhamento em tempo real por GPS das ambulâncias atendendo os chamados, além de contar com um *dashboard*[9] com informações gerais sobre as situações das ambulâncias, trajetos em tempo real dos acidentes visualizado por um mapa e controle do tempo estimado de retorno até o hospital.

[9] Um dashboard é uma interface gráfica que exibe informações de forma visual e concisa, geralmente em formato de painéis, gráficos e tabelas. É utilizado para monitorar, analisar e apresentar dados e métricas relevantes de uma organização, processo ou sistema em tempo real, facilitando a tomada de decisões e fornecendo insights de forma rápida e intuitiva.

Inicialmente, como requisitos funcionais[10], além do cadastro de ambulâncias e chamados, o sistema permite acompanhar a posição das ambulâncias em tempo real através de um mapa. Caso estejam operando em um chamado, é possível obter informações sobre o tempo estimado desse chamado, permitindo um controle eficiente entre o hospital e as ambulâncias.

2.5 OBJETIVOS ESPECÍFICOS

- Otimização no atendimento pré-hospitalar centralizando o controle de chamados de ambulâncias.
- Permitir a visualização de tempo estimado do acidente até o hospital.
- Visualização das rotas das ambulâncias e mapa.
- Monitoramento em Tempo Real: A implementação de monitoramento em tempo real, tem sido destacada como uma estratégia eficaz para reduzir tempos de espera e melhorar a eficiência dos serviços. Isso sugere que softwares que contam com localização em tempo real poderiam ser cruciais para otimizar o fluxo de pacientes e reduzir o desvio de ambulâncias.
- Interface fácil e perfil para hospital e ambulância.

Assim, o *Live Saver Locator* pode ser uma opção para melhorar o controle de ambulâncias resultando em uma melhoria de qualidade, facilitando o acesso a informações sobre o paciente, permitindo a comunicação em tempo real entre equipe médica e hospital e monitorando a localização das ambulâncias e o status das solicitações em tempo real.

3 JUSTIFICATIVA

A literatura sobre *softwares* de controle de chamados e localização em tempo real de ambulâncias, especialmente em contextos de saúde, aborda várias estratégias e tecnologias para melhorar a eficiência, eficácia e a segurança do atendimento médico de urgência. (MINISTÉRIO DA SAÚDE, 2020).

O objetivo desse projeto é tentar prover uma melhora no atendimento pré-hospitalar, com base em entrevista, formulário, pesquisas sobre artigos e trabalhos acadêmicos foi possível visualizar desfalque em alguns aspectos, um dos principais é justamente a falta de controle do atendimento móvel até o retorno para o hospital. Dessa forma o sistema foi baseado nesse tema, buscando melhorar este pré-atendimento.

[10] Requisitos funcionais são especificações que definem o comportamento que um sistema deve ter. Eles descrevem as funções que o sistema deve realizar, as quais são essenciais para atender aos objetivos e necessidades dos usuários finais.

“Segundo o Ministério da Saúde, a humanização é a valorização dos usuários, trabalhadores e gestores no processo de produção de saúde, por meio da criação de vínculos solidários, da responsabilidade compartilhada e da participação coletiva nos processos de trabalho, objetivando a mudança na cultura da atenção aos pacientes.” (AUTOLAC, s.d.)

O desenvolvimento deste projeto para localização de ambulâncias e controle de chamados visa melhorar o preparo para o atendimento pré-hospitalar, atendendo aos chamados de forma mais rápida e organizada. O sistema possibilita uma visão de localização real das ambulâncias, a situação que ela se encontra e a descrição do chamado de emergência. Essas informações auxiliam o atendimento, podendo salvar a vida em risco de muitas pessoas.

A adaptação nas emergências e a evolução do controle nessas situações poderia vir a ser mais rápida e eficiente com o auxílio de um *software*, que reduza ainda que minimamente, o tempo de preparo para atendimento do paciente. Portanto, nesse contexto a aplicação se torna relevante e futuramente poderia contribuir na área da saúde.

4 METODOLOGIAS

Para a elaboração do documento, foi adotada uma abordagem metodológica que integra a revisão bibliográfica e a pesquisa de levantamento através de entrevistas e formulário.

A metodologia bibliográfica, conforme descrita pela doutoranda Naína Tumelero, é um procedimento de pesquisa que envolve a seleção, análise e interpretação de obras já publicadas sobre o tema de interesse (TUMELERO, 2019). Esta metodologia foi utilizada para informações relevantes sobre o serviço de atendimento de ambulâncias, como são realizados e sobre atendimentos pré-hospitalares. Foram consultados revistas científicas, artigos, publicações de órgãos governamentais e instituições de saúde, construindo assim uma base teórica que orientou as fases do projeto.

Foi realizada uma pesquisa de levantamento, também explicada pela doutoranda Naína Tumelero, que consiste em "um tipo de pesquisa que se realiza para a obtenção de dados ou informações sobre características ou opiniões de um grupo de pessoas, selecionado, em termos estatísticos, como representante de uma população." (TUMELERO, 2019). Esta metodologia foi aplicada para obter insights valiosos sobre a operação diária do serviço de ambulâncias, os principais problemas enfrentados, as necessidades não atendidas e as expectativas dos usuários em relação ao sistema de controle de chamados de ambulâncias. Foram conduzidas entrevistas com profissionais direta e indiretamente envolvidos no serviço, além da disponibilização de um questionário mostrado nos apêndices.

A estratégia adotada no desenvolvimento do sistema *Live Saver Locator* combina abordagens ágeis e tradicionais para criar um processo dinâmico e eficaz. Inicialmente, foram implementadas práticas ágeis, como reuniões diárias (*daily meetings*) [11] e *sprints*[12] do *framework Scrum*[13], para permitir entregas frequentes e adaptáveis. Isso garantiu que a equipe conseguisse lidar com mudanças nos requisitos do projeto de forma flexível. Em paralelo, em momentos-chave, foram incorporados elementos de metodologias tradicionais, como o método espiral, para estabelecer uma base sólida nos requisitos iniciais e facilitar o planejamento detalhado de certas etapas do desenvolvimento, assim realizando a criação de versões do projeto partindo de funcionalidades a serem implementadas conforme necessidades encontradas no sistema. Além disso, houve ênfase em práticas robustas de gerenciamento de projeto, controle de qualidade e testes contínuos. Essa abordagem híbrida permitiu alcançar uma flexibilidade ágil, mantendo a estrutura e definição necessárias para garantir a entrega de um produto de qualidade.

[11] São reuniões diárias de no máximo 15 minutos, utilizadas principalmente em métodos ágeis, como o Scrum. Essas reuniões têm como objetivo otimizar a comunicação e planejar o trabalho do dia, repassando as atividades do dia anterior e definindo o plano para as próximas 24 horas.

[12] Sprints são períodos de tempo limitados, geralmente de uma a quatro semanas, no framework Scrum, durante os quais uma versão incremental e usável de um produto é desenvolvida.

[13] Scrum é um framework ágil para gerenciamento de projetos, frequentemente usado no desenvolvimento de software. Ele enfatiza o trabalho em equipe, a colaboração e a entrega contínua de valor aos clientes.

5 FUNDAMENTAÇÃO TEÓRICA

Para desenvolver um sistema de ambulâncias capaz de garantir a resposta rápida e adequada em emergências, contribuindo significativamente para a redução de mortalidade e morbidade foram necessários a coleta de requisitos funcionais e não funcionais como primeira etapa do processo. Um sistema de ambulâncias deve integrar tecnologias avançadas de comunicação, logística e gestão de recursos. Isso inclui a utilização de GPS para localização precisa dos veículos. Descrevendo os requisitos, eles são separados em dois grupos: funcionais e não funcionais, sendo esses requisitos menos prioritários para a execução do sistema.

Os requisitos são as especificações necessárias para o desenvolvimento de um projeto, que podem ser categorizados em: funcionais e não funcionais. (ESCOLA DNC, 2024). Para esta aplicação foram definidos os seguintes requisitos.

Requisitos Funcionais:

Login

- O sistema deve permitir o login com três tipos de usuários: Administrador, Hospital e Motorista.
- Cada tipo de usuário deve ter acesso a funcionalidades específicas conforme seu perfil.

Gerenciamento de Usuários (Hospital ou Motorista)

- O sistema deve permitir o cadastro de novos usuários, especificando se são do tipo Hospital ou Motorista.
- O sistema deve permitir a alteração dos dados dos usuários cadastrados.
- O sistema deve permitir a exclusão de usuários.
- O sistema deve permitir a consulta de dados de usuários.

Gerenciamento de Ambulâncias

- O sistema deve permitir a consulta de informações sobre ambulâncias.
- O sistema deve permitir o cadastro de novas ambulâncias.
- O sistema deve permitir a edição das informações das ambulâncias cadastradas.
- O sistema deve permitir a exclusão de ambulâncias.

Gerenciamento de Chamados

- O sistema deve permitir o cadastro de novos chamados de emergência.
- O sistema deve permitir a edição das informações dos chamados cadastrados.
- O sistema deve permitir a atribuição de uma ambulância específica a um chamado.

Rastreamento de Ambulâncias

- O sistema deve permitir o rastreamento em tempo real da localização das ambulâncias.

Funcionalidades do Motorista

- O motorista deve ser capaz de realizar login no sistema e selecionar a ambulância que está utilizando.
- O motorista deve ser capaz de alterar a localização da ambulância em tempo real.
- O motorista deve ser capaz de alterar o estado do chamado para as seguintes opções: A Caminho, Retornando, Finalizado.

Requisitos não funcionais:

Notificações

- O sistema deve notificar o Hospital sobre atualizações e mudanças no estado dos chamados.

Acessibilidade

- O sistema é acessível para o motorista pois sua interface é simples e apenas com botões, visando a agilidade do condutor e foco no trânsito.

O sistema foi desenvolvido utilizando tecnologias e ferramentas robustas, segundo informações disponíveis na documentação dos projetos Spring Boot (2024) e Angular (2024), garantindo um sistema confiável, fácil de usar e com bom desempenho. Foram utilizados conceitos de desenvolvimento de *software* como programação orientada a objetos, ORM (*Object Relational Mapper*)[14] e banco de dados relacional.

As escolhas dessas ferramentas, permitem um sistema confiável para o controle de chamados de ambulâncias, oferecendo segurança e eficiência ao hospital. A maioria das tecnologias e ferramentas utilizadas com exceção da *Google Maps* API, são gratuitas ou de código aberto. Conforme relatado pela *Linux Foundation* em 2020, o uso de ferramentas de código aberto apresenta múltiplas vantagens, como acessibilidade e adaptabilidade. Estas ferramentas são soluções de software que trazem uma série de benefícios para os projetos, representando uma alternativa econômica às licenças de software comerciais e possibilitando uma significativa redução de custos para empresas e desenvolvedores. (*LINUX FOUNDATION*, 2020).

[14] ORM é uma técnica que simplifica a interação entre sistemas orientados a objetos e bancos de dados relacionais. O ORM permite que os desenvolvedores trabalhem com objetos e classes em vez de escrever consultas SQL manualmente.

Uma das principais vantagens das ferramentas de código aberto reside na sua flexibilidade e na capacidade de personalização que proporcionam. Ao disponibilizar o acesso ao código-fonte, elas conferem aos desenvolvedores a liberdade para modificar e ajustar os recursos de acordo com as necessidades específicas de cada projeto. Tal flexibilidade permite realizar adaptações precisas e eficazes nas funcionalidades. (*LINUX FOUNDATION*, 2020).

Foram utilizados também os Diagramas UML (*Unified Modeling Language*) que são uma linguagem gráfica padrão usada para visualizar, especificar, construir e documentar os artefatos de um sistema de software. Eles foram desenvolvidos pela OMG (*Object Management Group*) como parte do esforço para padronizar a modelagem de sistemas de software. A UML é amplamente utilizada em todo o mundo por engenheiros de software, arquitetos de sistemas e outros profissionais envolvidos na criação de software para comunicar ideias complexas de maneira clara e concisa.

A UML oferece uma variedade de tipos de diagramas, cada um projetado para representar diferentes aspectos de um sistema. Esses diagramas incluem:

- Diagrama de Classes: Mostra as classes de um sistema, suas relações e atributos. É fundamental para entender a estrutura do sistema e como as classes interagem entre si.

- Diagrama de Sequência: Ilustra como os objetos se comunicam entre si ao longo do tempo. É útil para visualizar fluxos de controle e interações entre objetos durante a execução de um sistema.

- Diagrama de Casos de Uso: Descreve as funcionalidades do sistema do ponto de vista dos usuários finais. Ajuda a identificar quais funcionalidades são necessárias e como elas devem funcionar.

- Diagrama de Componentes: Representa os componentes de um sistema e suas dependências. É usado para organizar o código em partes menores e gerenciar a complexidade.

- Diagrama de Atividades: Semelhante ao diagrama de sequência, mas focado mais nas atividades do sistema do que nos objetos individuais. É útil para modelar processos de negócios ou algoritmos.

- Diagrama de Estados: Mostra os estados possíveis de um objeto e como ele transita de um estado para outro. É crucial para entender o comportamento dinâmico de um sistema.

- Diagrama de Implementação: Mapeia classes para implementações específicas, mostrando como o código real está organizado. É importante para a fase de codificação e manutenção do sistema.

A UML não apenas facilita a comunicação entre diferentes stakeholders em um projeto de software; ela também ajuda a capturar requisitos, a planejar a arquitetura do sistema, a analisar

o design e a documentar o sistema final. Além disso, a UML pode ser usada para gerar código automaticamente, embora essa seja uma aplicação menos comum.

O uso eficaz da UML requer prática e experiência, pois a escolha do tipo de diagrama certo para representar uma determinada informação é crucial para a compreensão clara do sistema. Com o tempo, muitos desenvolvedores e equipes de projeto têm adotado a UML como uma ferramenta essencial em seu arsenal de desenvolvimento de software, reconhecendo seu valor para melhorar a qualidade do software produzido e a eficiência do processo de desenvolvimento.

5.1 DIAGRAMAS DE CASO DE USO

Os Diagramas de Caso de Uso são uma ferramenta essencial na modelagem de sistemas de software, especialmente úteis para representar as interações entre os usuários (atualmente chamados de atores) e o sistema que estão sendo desenvolvido. Esses diagramas são fundamentais para entender o escopo do sistema, as funcionalidades que ele oferece e como essas funcionalidades atendem às necessidades dos usuários. Eles são parte integrante da Linguagem de Modelagem Unificada (UML), que busca padronizar a representação de sistemas de software através de uma variedade de diagramas visuais.

Elementos Principais dos Diagramas de Caso de Uso:

- Atores: São os usuários ou outras entidades externas que interagem com o sistema. Os atores podem ser humanos, organizações, máquinas ou até mesmo outros sistemas.

- Casos de Uso: Representam as funcionalidades ou tarefas que o sistema realiza para atender às necessidades dos atores. Cada caso de uso é descrito de forma a indicar claramente o que o sistema faz e qual é o resultado esperado.

- Relacionamentos: Estabelecem como os atores interagem com os casos de uso. Os relacionamentos podem ser de várias formas, incluindo associação, inclusão (*include*), extensão (*extend*) e herança (generalização).

Importância dos Diagramas de Caso de Uso:

Os diagramas de caso de uso são cruciais para várias etapas do ciclo de vida de um projeto de software:

- Análise de Requisitos: Permite capturar os requisitos do sistema de uma maneira que seja compreensível tanto pelos desenvolvedores quanto pelos stakeholders.

- Design e Arquitetura: Serve como uma referência para o design do sistema, ajudando a garantir que todas as funcionalidades necessárias sejam abordadas.

- Teste: Facilita a identificação de cenários de teste, permitindo que os desenvolvedores verifiquem se o sistema atende aos requisitos especificados.

Como Criar um Diagrama de Caso de Uso:

Criar um diagrama de caso de uso envolve alguns passos básicos:

- Identificar os Atores: Liste todos os usuários ou entidades que interagem com o sistema.

- Identificar os Casos de Uso: Determine as funcionalidades que o sistema precisa fornecer para atender às necessidades dos atores.

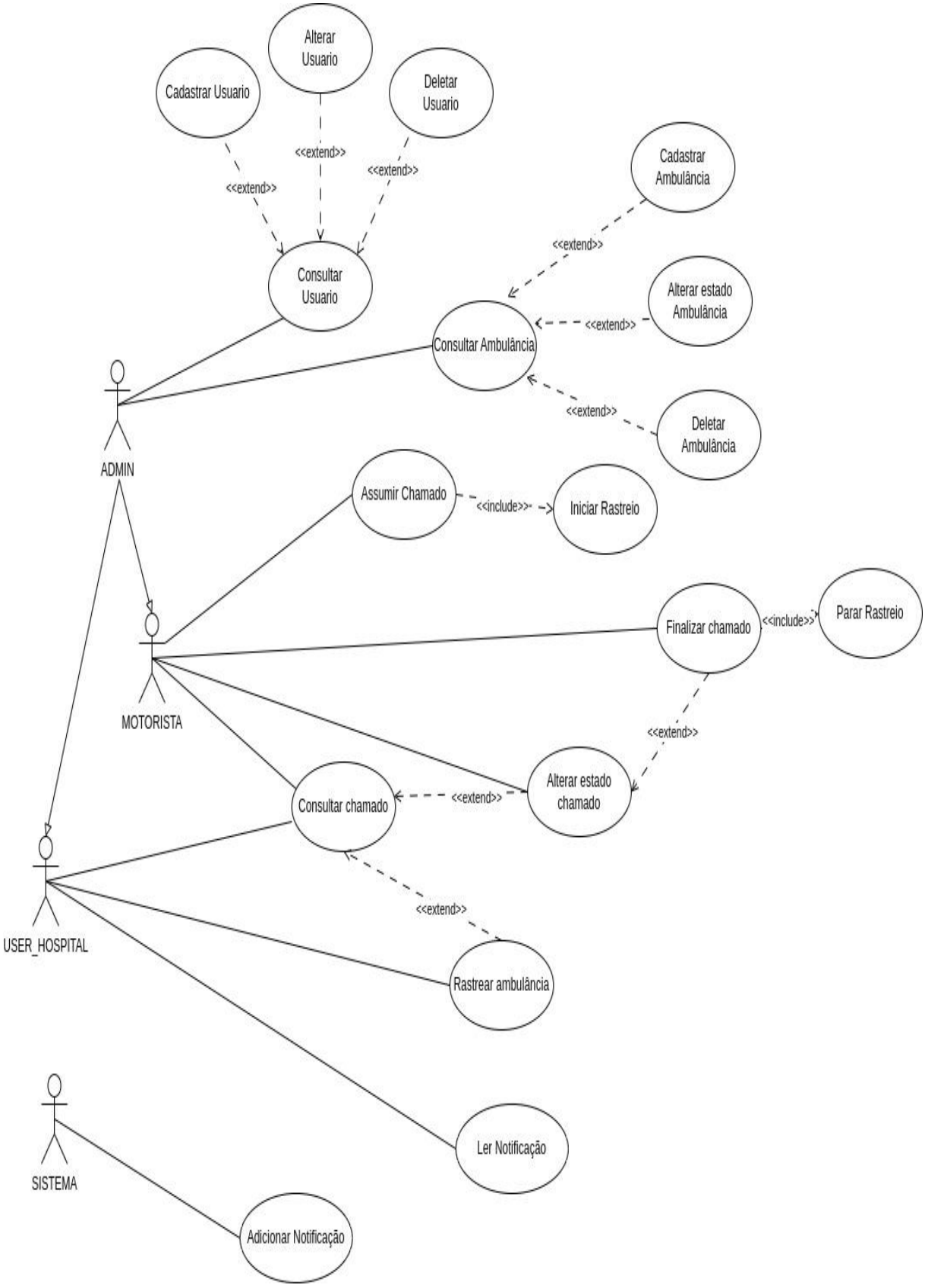
- Desenhar os Relacionamentos: Conecte os atores aos casos de uso relevantes, utilizando os tipos de relacionamentos apropriados para representar as interações.

- Rotular os Casos de Uso: Use verbos e descrições claras para cada caso de uso, destacando o que o sistema faz e o resultado esperado.

- Revisar e Refinar: Revise o diagrama para garantir que todos os aspectos importantes do sistema sejam representados corretamente e que haja clareza sobre como os atores interagem com o sistema.

Os diagramas de caso de uso são uma ferramenta poderosa para comunicar o propósito e a funcionalidade de um sistema de software de uma maneira que seja acessível a todos os envolvidos no projeto. Ao seguir as diretrizes e práticas recomendadas, é possível criar diagramas de caso de uso que não apenas auxiliem na compreensão do sistema, mas também contribuam significativamente para a qualidade do produto final. Como demonstra a Figura 1:

Figura 1 – Digrama de Caso de Uso



Fonte: Feito pelos autores, 2024.

5.2 DIAGRAMA DE CLASSES

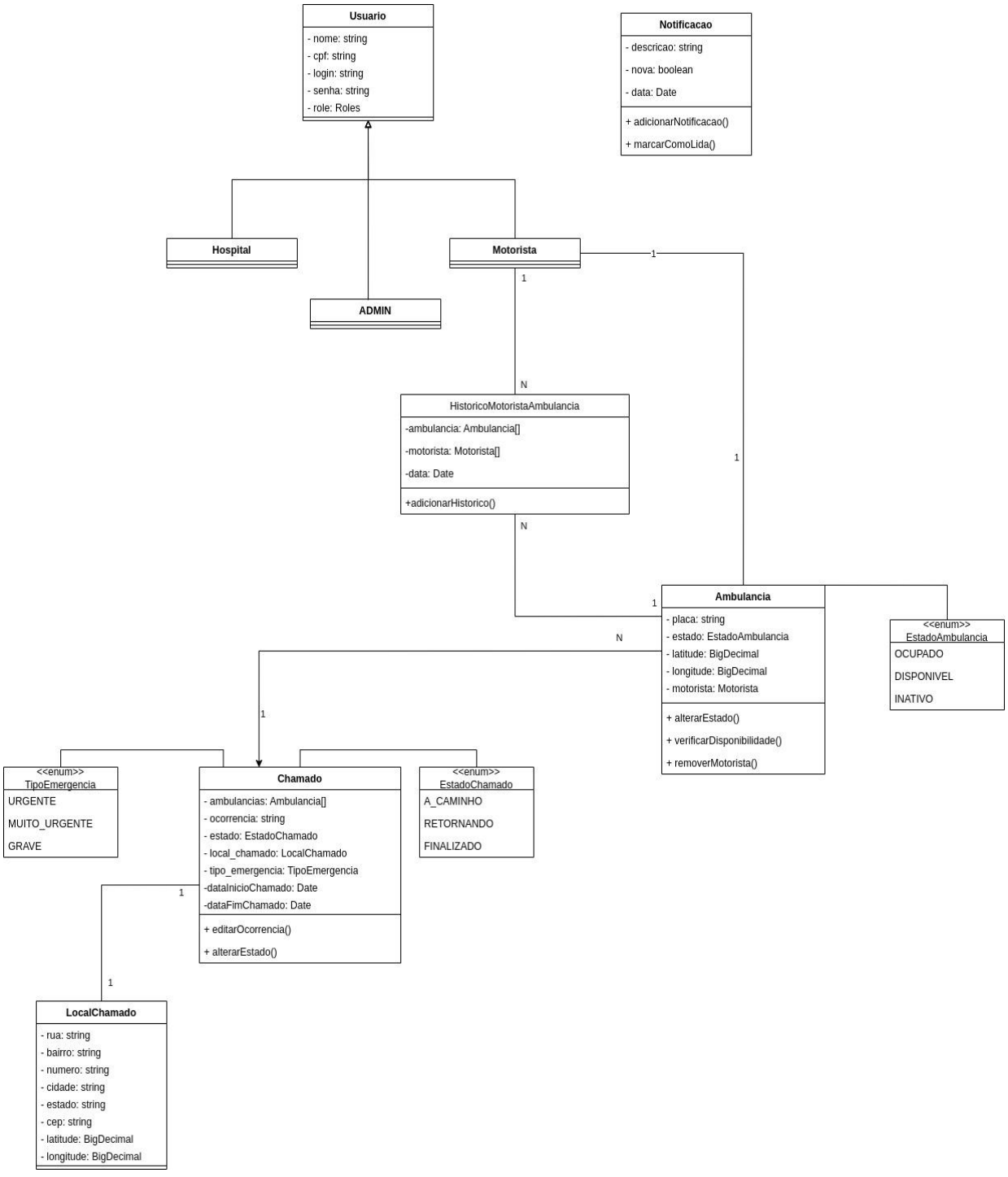
Diagramas de classe são uma representação visual utilizada na programação orientada a objetos para descrever a estrutura e as relações entre as classes de um sistema. Esses diagramas são fundamentais para o entendimento e a organização do design de software, permitindo a visualização clara das entidades (classes), seus atributos, métodos e as conexões entre elas.

As classes são representadas por retângulos divididos em três seções: o nome da classe, seus atributos e métodos. Os atributos são variáveis associadas à classe que armazenam informações sobre os objetos dessa classe, enquanto os métodos são funções que definem o comportamento desses objetos.

Além disso, os diagramas de classe permitem expressar relações entre classes, como herança (onde uma classe deriva funcionalidades de outra), associação (indicando que duas classes estão ligadas de alguma forma), agregação (representando uma relação "todo-parte" sem implicação de propriedade) e composição (uma forma especial de agregação indicando uma relação de propriedade).

Esses diagramas são cruciais tanto para o desenvolvimento inicial de um projeto quanto para a comunicação entre desenvolvedores e stakeholders, facilitando a compreensão dos requisitos do sistema e a tomada de decisões sobre a arquitetura do software. Eles também auxiliam na identificação de problemas potenciais e na otimização do design do sistema antes da implementação do código. Observe a Figura 2:

Figura 2 – Diagrama de Classes



Fonte: Feito pelos autores, 2024.

5.3 DIAGRAMA DE ENTIDADE-RELACIONAMENTO

Um Diagrama Entidade-Relacionamento (DER) é uma representação gráfica utilizada na modelagem de banco de dados relacional para visualizar as entidades (objetos ou conceitos), os relacionamentos entre essas entidades e as propriedades desses relacionamentos. Este tipo de diagrama é fundamental para projetar e entender a estrutura de um SGBD (Sistema de Gerenciamento de Banco de Dados), facilitando a comunicação entre desenvolvedores, analistas de sistemas e usuários finais sobre como os dados serão organizados e acessados.

Componentes Principais:

- Entidades: Representam objetos ou conceitos do mundo real que são relevantes para o domínio do problema sendo modelado. Cada entidade tem atributos, que são características específicas da entidade. Por exemplo, em um sistema de gerenciamento de biblioteca, as entidades podem ser "Livro", "Autor" e "Leitor".

- Atributos: São as propriedades das entidades que fornecem informações detalhadas sobre elas. No contexto dos livros, os atributos podem incluir título, gênero, número de páginas, etc.

- Relacionamentos: Descrevem como as entidades estão conectadas entre si. Um relacionamento pode ser unário (uma entidade está associada a outra), binário (duas entidades estão associadas) ou ternário (três entidades estão associadas). Os relacionamentos também têm suas próprias propriedades, como cardinalidade (quantidade de instâncias de uma entidade que podem estar ligadas a uma instância de outra entidade) e opcionalidade (se o relacionamento deve existir).

Tipos de Relacionamentos:

Existem vários tipos de relacionamentos que podem ser representados em um DER:

- Associação: Indica que duas entidades estão relacionadas de alguma forma. Por exemplo, um livro pode ter um autor.

- Generalização: É usado para indicar uma relação hierárquica entre entidades. Uma entidade pode ser vista como uma generalização de outras entidades mais específicas.

- Especialização: O oposto da generalização, onde uma entidade específica é vista como uma especialização de uma entidade genérica.

- Agregação: Representa uma relação composta por partes e todo. Por exemplo, um departamento pode ser visto como uma agregação de funcionários.

Importância do DER:

O DER é crucial para várias fases do ciclo de vida de um projeto de software, incluindo:

- Análise de Requisitos: Ajuda a identificar e documentar os requisitos do sistema em termos de entidades, seus atributos e como eles se relacionam.

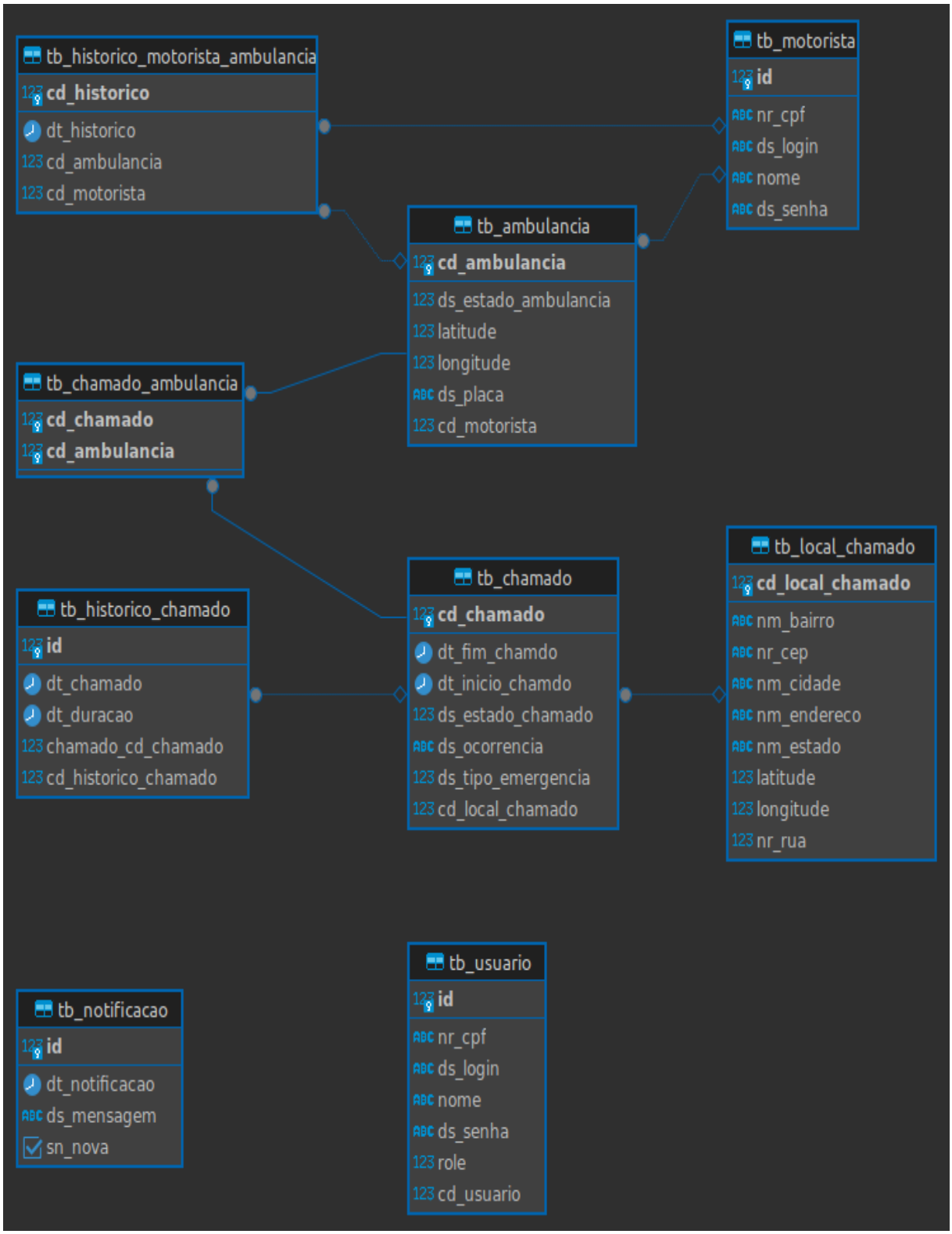
- Design de Banco de Dados: Serve como base para o design físico do banco de dados, definindo tabelas, campos e chaves primárias e estrangeiras.

- Comunicação: Facilita a comunicação entre diferentes stakeholders, pois fornece uma visão clara e concisa da estrutura dos dados.

- Manutenção: Atua como um guia para futuras alterações no esquema do banco de dados, garantindo que as mudanças não quebrem a integridade dos dados.

Em resumo, um Diagrama Entidade-Relacionamento é uma ferramenta poderosa para a modelagem de bancos de dados, permitindo uma representação visual clara e intuitiva das entidades, seus atributos e como elas interagem entre si. Isso ajuda a garantir que o sistema de banco de dados seja bem projetado, eficiente e capaz de atender às necessidades dos usuários e do negócio. Como mostra a Figura 3:

Figura 3– Diagrama de Entidade-Relacionamento



Fonte: Feito pelos autores, 2024.

A**s l**inguagens de programação: Existem diversas linguagens de programação e cada uma com suas particularidades e usos diferentes, graças a elas é possível desenvolver softwares e aplicações web, aplicações para dispositivos móveis e servidores.

> Para se desenvolver uma aplicação web, é necessário utilizar linguagens específicas que são responsáveis pela formatação, estilização e dinamicidade dessa aplicação. A partir disso, a linguagem responsável pela marcação de uma página web é a HTML, que significa HyperText Markup Language, compreendida em português como Linguagem de Marcação de Hipertexto. [...] o HTML5, através de suas novas tags semânticas e da permissão da incorporação de outras tecnologias - como APIs, trouxe aos usuários uma melhor experiência na utilização da Web e uma possibilidade de garantir, de maneira efetiva, a acessibilidade das páginas disponíveis na internet. Ainda de acordo com os autores, a combinação do HTML5, com o CSS e o JavaScript, essa linguagem de marcação passa a ser multiplataforma, o que aumenta o acesso aos mais diversos tipos de sites. (FERRAZ et al. MARQUES et al. SANTOS et al. MOREIRA, 2022)

O sistema foi desenvolvido arquitetado em duas partes: *front-end* e *back-end.* No *back-end,* a aplicação web foi construída com o framework *Spring Boot* como base, adotando o padrão de arquitetura MVC (Model, View, Controller)[15]. O mapeamento do banco de dados *PostgreSQL*[16] foi realizado com JPA para garantir a integração eficiente entre a aplicação e o banco de dados. Para a segurança da aplicação, foi implementada a autenticação utilizando o *Spring Security*[17] em conjunto com tokens JWT (*JSON Web Token*)[18] . As senhas dos usuários foram criptografadas utilizando o *BcryptEncoder*[19]. Além disso, a documentação da API foi gerada automaticamente com o *Swagger*[20].

Já o *front-end* da aplicação foi desenvolvido utilizando o *framework Angular* com *TypeScript,* seguindo o paradigma de orientação a objetos. Para a interface do usuário, foram utilizados os

[15]O padrão MVC: o Model gerencia os dados, a View renderiza as telas e o Controller decidem qual Model e View utilizar.

[16] PostgreSQL é um sistema de gerenciamento de banco de dados objeto-relacional (ORDBMS) de código aberto, altamente robusto e escalável.

[17] O Spring Security é um framework do ecossistema Spring, utilizado para implementar autenticação e controle de acesso em aplicações Java.

[18] É um formato compacto e seguro para transmitir informações entre partes como um objeto JSON. É comumente usado para autenticação e troca de informações entre sistemas.

[19] BcryptEncoder é um algoritmo de criptografia usado comumente para armazenar senhas com segurança em sistemas de computador. Ele é conhecido por sua capacidade de gerar hashes (representações criptografadas) de senhas que são extremamente difíceis de quebrar, mesmo por meio de ataques de força bruta.

[20] O Swagger é uma estrutura de software que ajuda a projetar, construir, documentar e consumir APIs de maneira eficiente.

componentes do *Angular Material*[21] e *Bootstrap*[22], garantindo um design consistente e responsivo. O roteamento entre os componentes foi realizado com o *Angular Routing*[23], e protegidas com *Angular Guards*[24], proporcionando uma navegação fluida e segura dentro da aplicação. Os estilos foram estilizados utilizando SCSS (Sassy Cascading Style Sheets)[25], seguindo padrões de estilização modernos e organizados.

Para implementar o sistema foi preciso uma busca sobre o processo de atendimento já existente no ambiente hospitalar. O APH (Atendimento Pré-Hospitalar) é o conjunto de medidas tomadas no local da ocorrência de um acidente ou mal-estar súbito, visando à estabilização do paciente e ao seu transporte seguro para um serviço de saúde de maior complexidade. Este serviço fundamental é realizado por equipes capacitadas, geralmente compostas por médicos, enfermeiros, técnicos de enfermagem e socorristas, que utilizam ambulâncias equipadas para atender às diversas necessidades dos pacientes. (MINISTÉRIO DA SAÚDE, 2024)

De acordo com o Corpo de Bombeiros Militar do Distrito Federal (2019), o processo de atendimento pré-hospitalar envolve diversas etapas, desde a recepção da chamada até a entrega do paciente ao serviço de saúde de destino. As principais etapas são:

Recepção da chamada: A chamada é recebida por uma central de atendimento, que deve obter informações sobre a situação do paciente, a localização da ocorrência e o tipo de veículo necessário.

Deslocamento da ambulância: A ambulância mais próxima do local da ocorrência é acionada e se dirige para o local. Durante o trajeto, a equipe de atendimento pré-hospitalar deve se preparar para realizar os procedimentos necessários.

Chegada ao local: Ao chegar ao local da ocorrência, a equipe deve avaliar a situação do paciente e realizar os primeiros socorros. Isso pode incluir medidas como RCP (Reanimação Cardiopulmonar), controle de hemorragias, administração de medicamentos e suporte ventilatório.

Transporte do paciente: O paciente é estabilizado e transportado para o serviço de saúde de destino, seja um hospital, pronto-socorro ou outra unidade de atendimento. Durante o transporte, a equipe deve monitorar o estado do paciente e realizar os procedimentos necessários para garantir sua segurança.

---

[21] O Angular Material é uma biblioteca de componentes de interface de usuário (UI) desenvolvida para Angular, baseada nas diretrizes do Material Design do Google.

[22] Bootstrap é um framework web com código-fonte aberto para desenvolvimento de componentes de interface e front-end para sites e aplicações web

[23] O Angular Routing é um recurso do framework Angular que permite a navegação entre diferentes componentes e visualizações em uma aplicação web.

[24] Os Angular Guards são mecanismos no Angular que protegem rotas, controlando o acesso a elas com base em certas condições.

[25] SCSS (Sassy CSS) é uma extensão do CSS (Cascading Style Sheets) que adiciona funcionalidades avançadas à linguagem CSS.

Entrega do paciente: Ao chegar ao serviço de saúde de destino, a equipe de atendimento pré-hospitalar deve passar informações sobre o estado do paciente à equipe médica do hospital. (Corpo de Bombeiros Militar do Distrito Federal, 2019).

6 PLANEJAMENTO DO PROJETO

A linguagem de programação *Java*[26] foi utilizada como linguagem principal para desenvolver junto com o *framework Spring Boot*[27] e *TypeScript*[28] com framework *Angular*[29].

Segundo informações disponíveis no site do projeto Spring Boot (2024), o framework *Spring Boot* é amplamente reconhecido e adotado para o desenvolvimento de aplicativos *Java*, destacando-se por sua estrutura modular e eficiente. Aliado ao *Angular* e *Typescript*, proporciona uma experiência de desenvolvimento ágil e flexível, permitindo que a equipe se concentre na implementação de recursos personalizados, sob medida para as necessidades do serviço de emergência.

O uso do *Angular* e *Typescript* traz uma camada adicional de eficiência e segurança ao desenvolvimento da aplicação *front-end*[30], garantindo uma experiência de usuário intuitiva e responsiva. Essas tecnologias permitem explorar todo o potencial de personalização e adaptabilidade, resultando em uma solução de localização de ambulâncias altamente customizável e adequada às demandas específicas do serviço.

Regras de Negócio

As regras de negócio para o sistema "Live Saver Locator" são definidas para garantir o funcionamento adequado e seguro do sistema. Elas abrangem as operações de gerenciamento de usuários, ambulâncias e chamados de emergência, bem como as funcionalidades específicas para diferentes tipos de usuários.

Login e Perfis de Usuário:

- O sistema deve permitir login com três tipos de usuários: Administrador, Hospital e Motorista.
- Cada tipo de usuário terá acesso apenas às funcionalidades específicas ao seu perfil.

---

[26] Java é uma linguagem de programação de propósito geral conhecida por sua portabilidade e orientação a objetos.

[27] O Spring Boot é uma ferramenta derivada do Spring Framework, um framework Java baseado nos padrões de projetos, inversão de controle e injeção de dependência.

[28] TypeScript é uma extensão do JavaScript, permitindo que os desenvolvedores escrevam código com tipagem estática, o que pode ajudar a prevenir erros comuns em JavaScript devido à sua tipagem dinâmica.

[29] Angular é um framework JavaScript desenvolvido e mantido pelo Google, projetado para facilitar o desenvolvimento de aplicações web dinâmicas e de alta performance.

[30] O Front-end é a parte visual de sites e aplicações, referindo-se à área das páginas com a qual os usuários podem interagir diretamente. É responsável por desenvolver a interface gráfica utilizando tecnologias base da Web, como HTML, CSS e JavaScript.

- Somente usuários autenticados podem acessar o sistema.

Gerenciamento de Usuários:

- Somente o Administrador pode cadastrar, editar e excluir usuários.
- Usuários do tipo Hospital e Motorista podem consultar suas informações e atualizar seus dados pessoais.

Gerenciamento de Ambulâncias:

- Apenas o Administrador pode cadastrar, editar e excluir ambulâncias.
- Todos os usuários autenticados podem consultar as informações das ambulâncias.

Gerenciamento de Chamados:

- Usuários do tipo Hospital podem cadastrar novos chamados de emergência.
- Os Motoristas podem assumir um chamado e atualizar o estado do chamado (A Caminho, Retornando, Finalizado).
- O sistema deve permitir a atribuição de uma ambulância específica a um chamado de emergência.

Rastreamento de Ambulâncias:

- O sistema deve permitir o rastreamento em tempo real das ambulâncias.
- Os Motoristas devem ser capazes de atualizar a localização da ambulância em tempo real.

Notificações:

- O sistema deve notificar o Hospital sobre atualizações e mudanças no estado dos chamados.
- As notificações devem ser claras e em tempo hábil para permitir respostas rápidas.

Acessibilidade:

- O sistema deve ser acessível, com uma interface simples e intuitiva, especialmente para os Motoristas, para garantir agilidade e segurança no trânsito.

6.1 DESENVOLVIMENTO DO SISTEMA

O desenvolvimento do sistema "Live Saver Locator" segue uma estrutura modular para facilitar a implementação e a manutenção. A seguir estão os principais componentes e suas funcionalidades:

1. Login:

- Implementação de um sistema de autenticação que permite login para Administrador, Hospital e Motorista.
- Controle de acesso baseado em perfis de usuário, garantindo que cada tipo de usuário tenha acesso apenas às funcionalidades específicas.

2. Gerenciamento de Usuários:

- Módulo para cadastro de novos usuários com campos como nome, CPF, login, senha e role.
- Funcionalidades para alteração e exclusão de usuários existentes.
- Interface para consulta de informações dos usuários.

3. Gerenciamento de Ambulâncias:

- Módulo para cadastro de novas ambulâncias com informações como placa, estado, latitude, longitude e motorista.
- Funcionalidades para edição e exclusão de ambulâncias cadastradas.
- Interface para consulta de informações das ambulâncias.

4. Gerenciamento de Chamados:

- Módulo para cadastro de novos chamados de emergência, incluindo informações sobre a ocorrência, local e tipo de emergência.
- Funcionalidades para edição de chamados existentes e atribuição de ambulâncias específicas.
- Controle do estado dos chamados (A Caminho, Retornando, Finalizado) atualizado em tempo real.

5. Rastreamento de Ambulâncias:

- Implementação de rastreamento em tempo real utilizando geolocalização.
- Interface para visualização da localização atual das ambulâncias.
- Funcionalidade para Motoristas atualizarem a localização da ambulância em tempo real.

6. Funcionalidades do Motorista:

- Módulo de login específico para Motoristas.
- Interface para seleção da ambulância utilizada.
- Funcionalidades para alteração do estado do chamado e atualização da localização da ambulância.

7. Notificações:

- Implementação de sistema de notificações que informa o Hospital sobre mudanças no estado dos chamados.
- Funcionalidade para marcar notificações como lidas.

8. Acessibilidade:

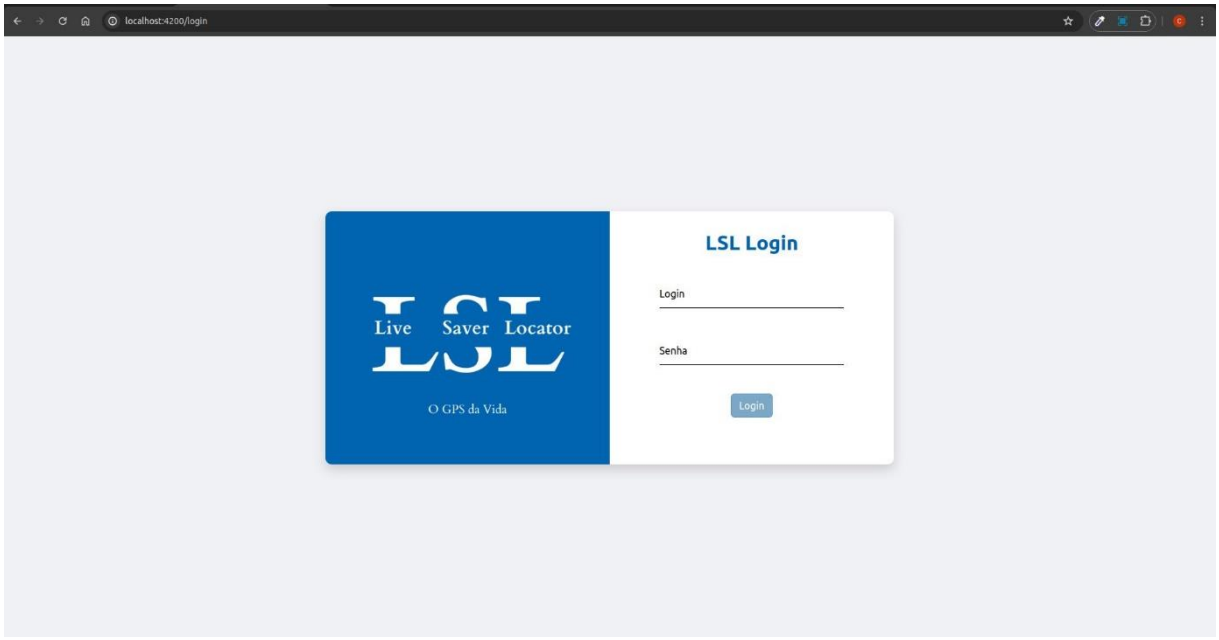
- Desenvolvimento de uma interface simplificada, especialmente para Motoristas, com botões grandes e de fácil acesso.
- Garantia de que todas as funcionalidades essenciais estejam disponíveis com o mínimo de interação necessária para garantir segurança e eficiência.

Para as interfaces foram utilizados conceitos da abordagem cliente servidor, sendo o lado servidor aquele que gerencia recursos e serviços que são fornecidos ao usuário e o lado do cliente aquele que usufrui dos recursos fornecidos pelo servidor. (CONTROLE.NET, 2024). Foram implementadas 2 interfaces principais, sendo elas hospital e motorista. Trabalhando ambas no mesmo *back-end* [31], as interfaces possuem comunicação entre si em relação aos chamados atendidos e as ambulâncias utilizadas pelo motorista.

A interface hospital, deve ser acessada pelos computadores do hospital, acessada por cargos de recepção ou especializados para direcionamento das ambulâncias para chamados. Após realizado o *login* (demonstrado na Figura 4) conta com o módulo de *dashboard* apresentado na Figura 5 e Figura 6, que possui uma visão geral dos chamados e ambulâncias que estão cadastrados no sistema, sendo as ambulâncias podendo ser visualizadas através de um mapa no próprio módulo de *dashboard*.

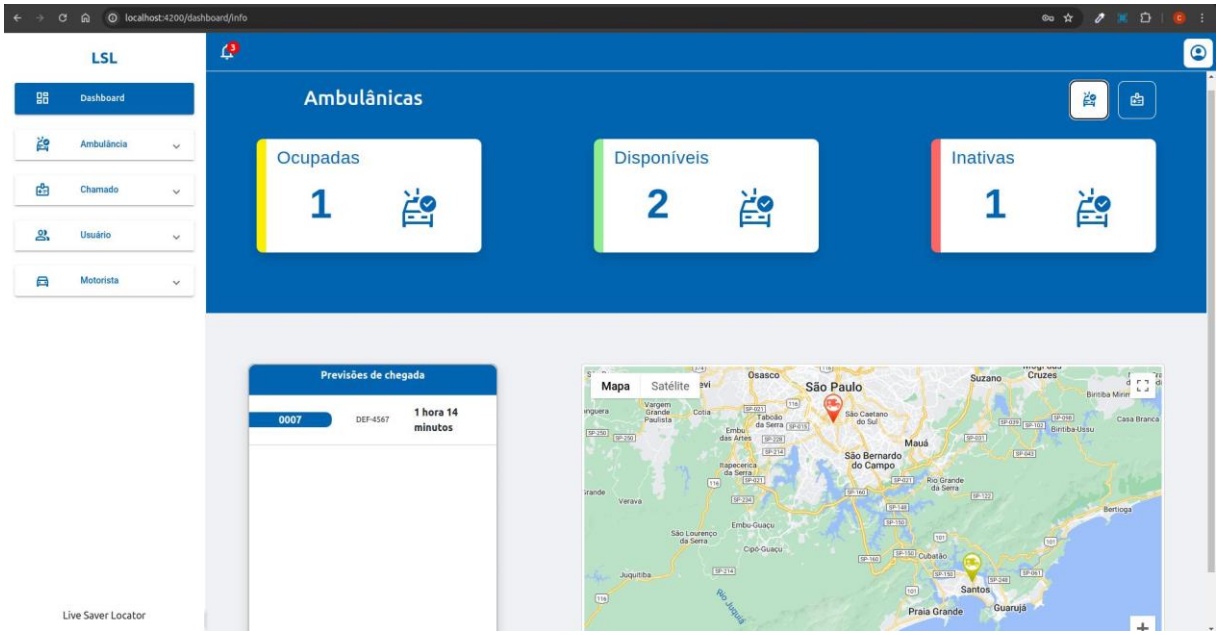
[31] O Back-end é responsável pelos dados da aplicação e pela lógica de negócios que permite que as funcionalidades do Front-end funcionem. O Back-end inclui aspectos como o banco de dados, servidores e segurança, que são essenciais para o funcionamento adequado do projeto.

Figura 4 – Login interface hospital



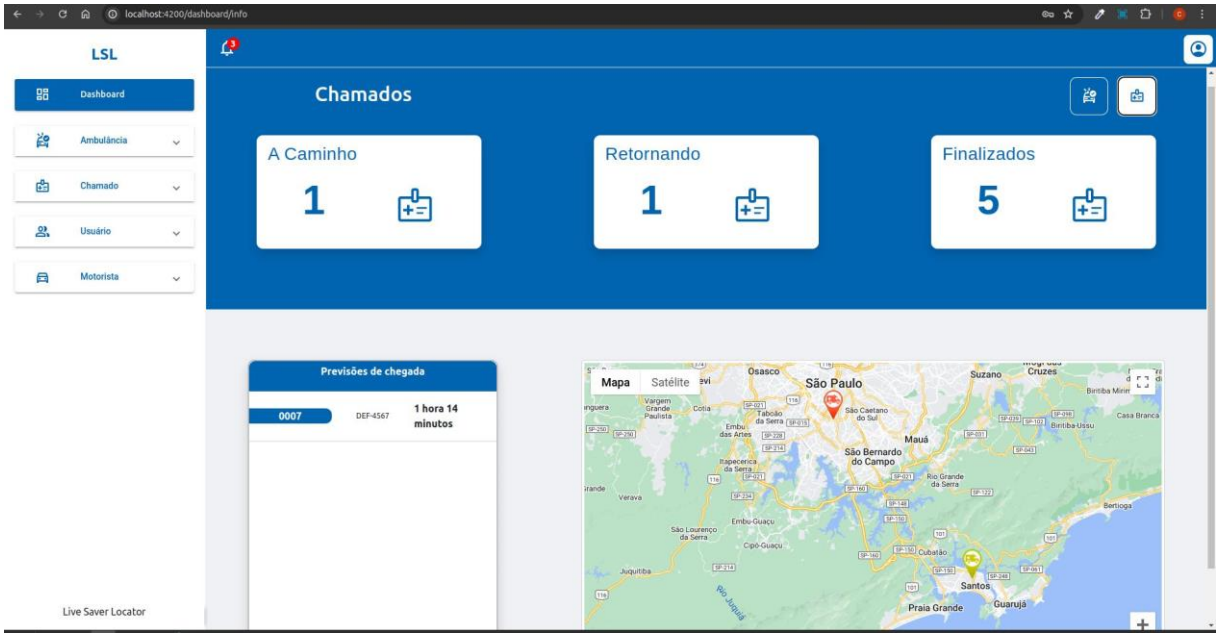
Fonte: Feitor pelos Autores, 2024.

Figura 5 – Dashboard ambulâncias interface hospital



Fonte: Feito pelos Autores, 2024.

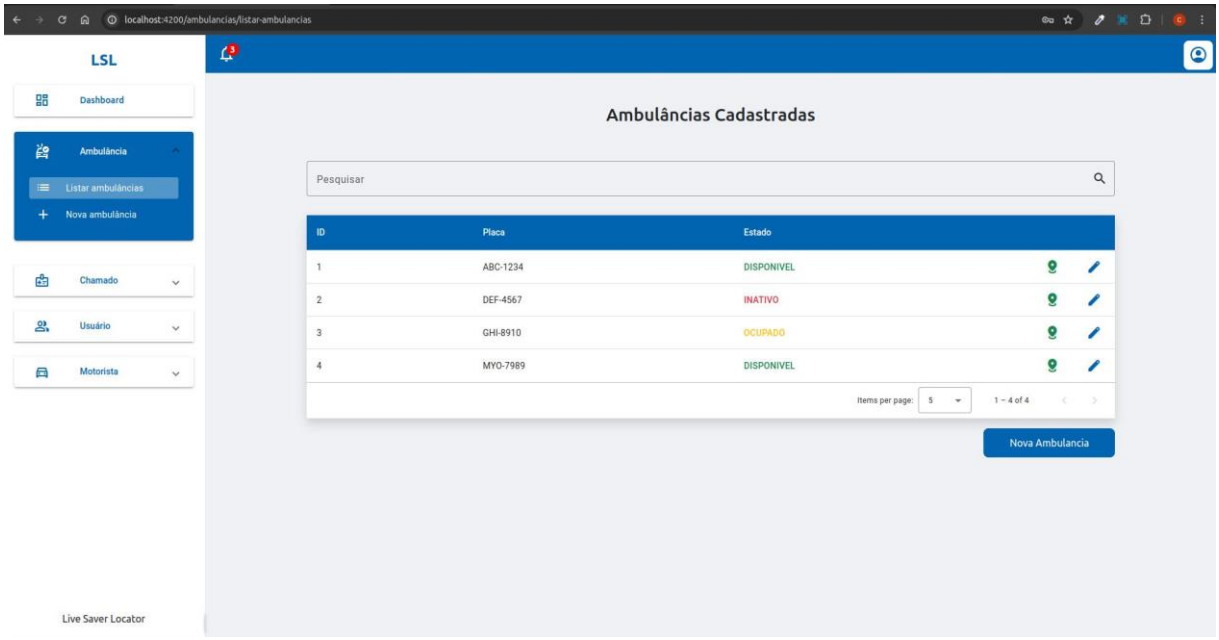
Figura 6 – Dashboard chamados interface hospital



Fonte: Feito pelos Autores, 2024.

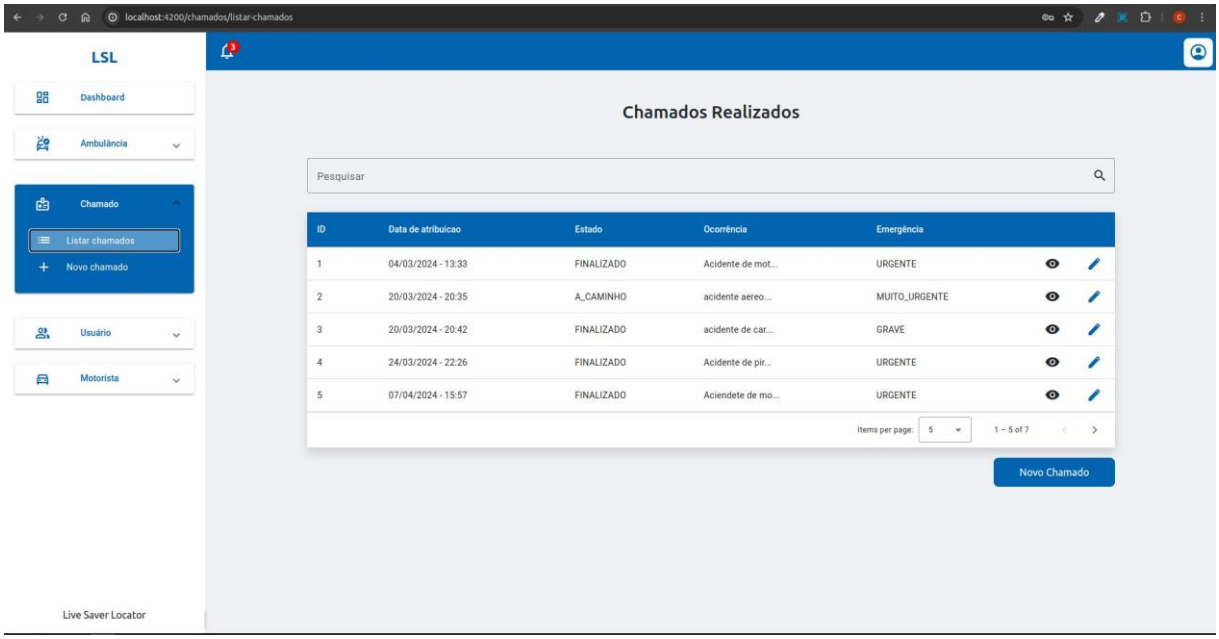
Essas informações são preparadas no *back-end* a partir dos dados mantidos em um banco de dados e lançadas ao *front-end*. Constam também outros 3 módulos de controle, sendo ambulância de acordo com a Figura 7, chamado na Figura 8, usuário na Figura 9 e motorista Figura 10, onde é permitido o cadastro, remoção ou alteração dos dados já cadastrados. No caso do módulo de motorista, é possível também encerrar a sessão de motoristas logados como uma segunda alternativa, para não conflitar futuros *logins*.

Figura 7 - Módulo ambulâncias interface hospital



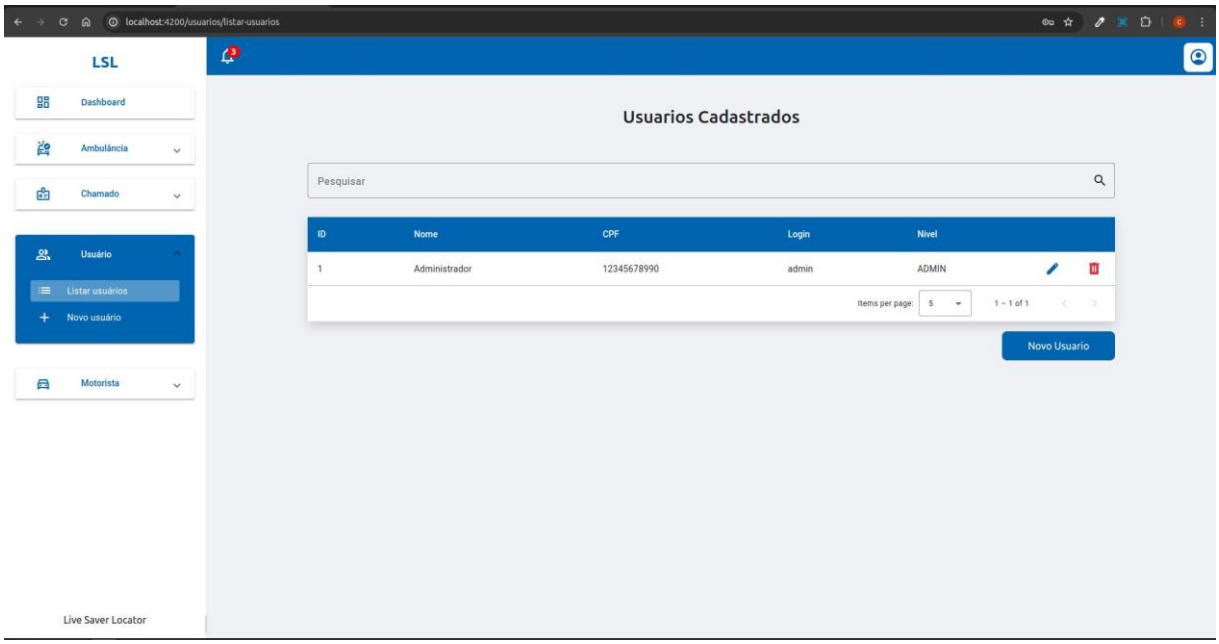
Fonte: Feito pelos Autores, 2024.

Figura 8 - Módulo de chamados interface hospital



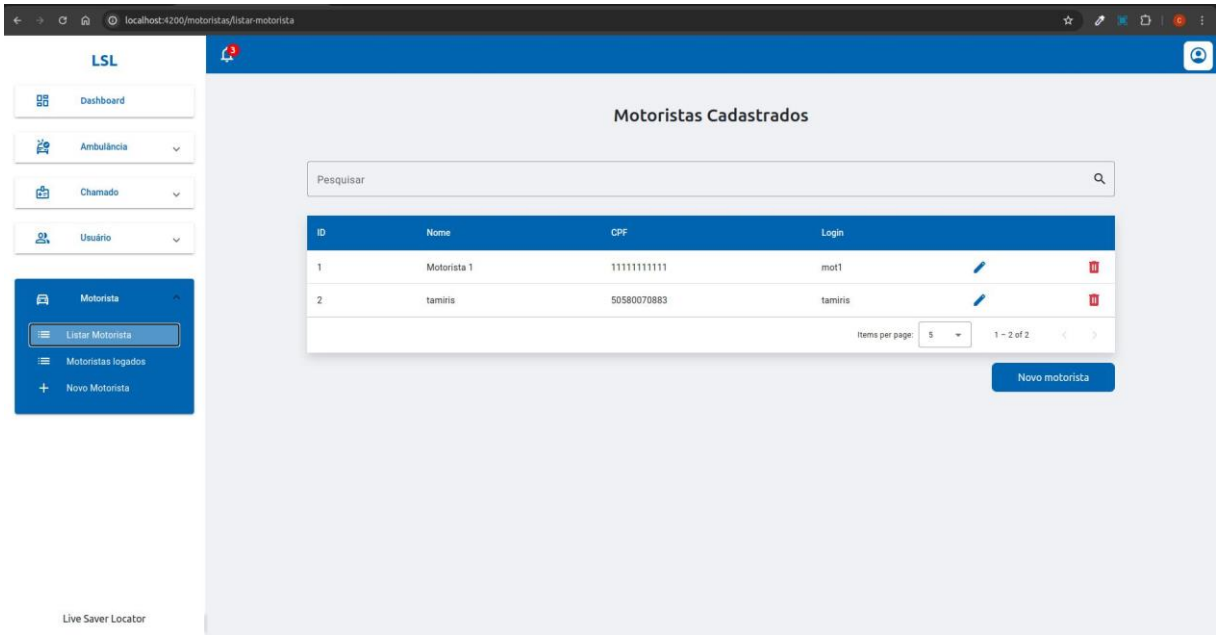
Fonte: Feito pelos Autores, 2024.

Figura 9 - Módulo de usuários interface hospital



Fonte: Feito pelos Autores, 2024.

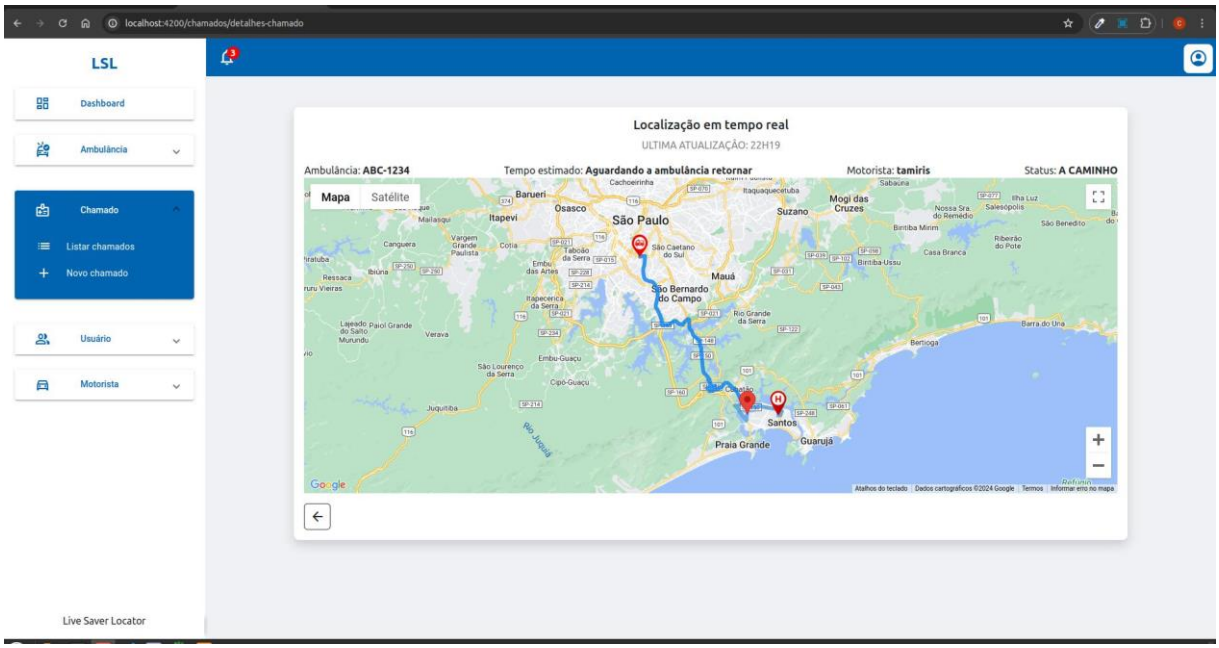
Figura 10 - Módulo de motoristas interface hospital



Fonte: Feito pelos Autores, 2024.

Além disso, o sistema oferece funcionalidades como rastreamento em tempo real das ambulâncias, tal como rotas e previsões podendo ser visualizadas no *dashboard* no quadro de previsões de chegada e em mapas relacionados aos chamados criados como apresentado na Figura 11, possibilitando a visualização clicando no ícone de visualizar no módulo de chamados.

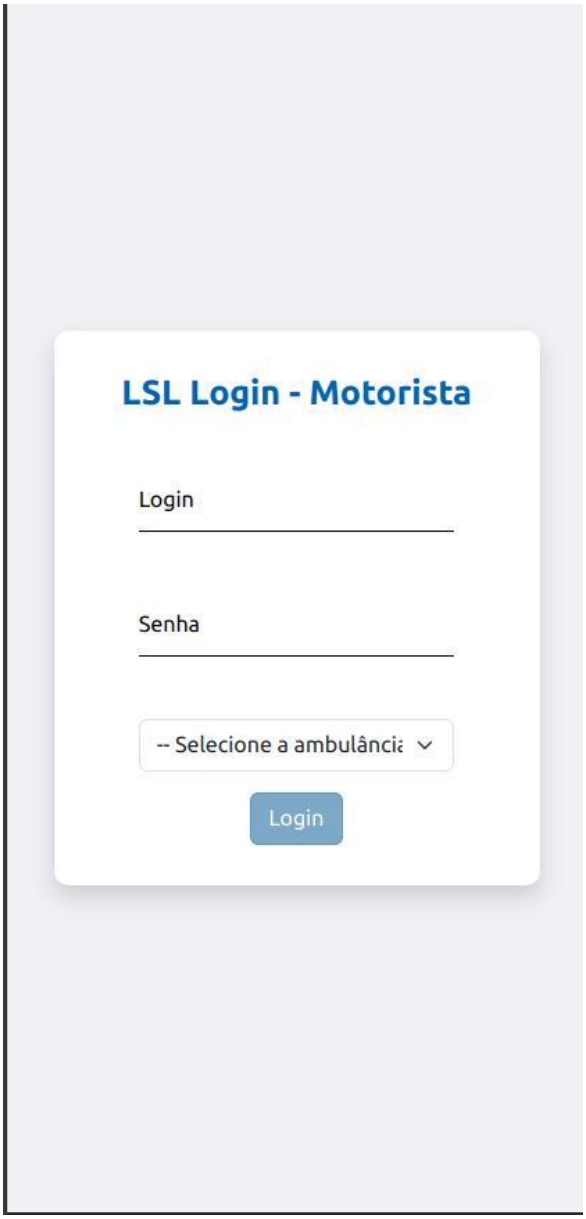
Figura 11 – Mapa de ambulâncias do chamado interface hospital



Fonte: Feito pelos Autores, 2024.

Já a interface motorista, deve ser acessada através de um dispositivo na ambulância, seja um *smartphone*, *tablet* ou monitor com um sistema operacional direcionado para o aplicativo. É pensada para um dispositivo móvel, ainda sim, podendo ser realizado através de um navegador. O login é realizado com as informações do motorista e ambulância que será operada conforme mostra a Figura 12.

Figura 12 – Login interface motorista



Fonte: Feito pelos Autores, 2024.

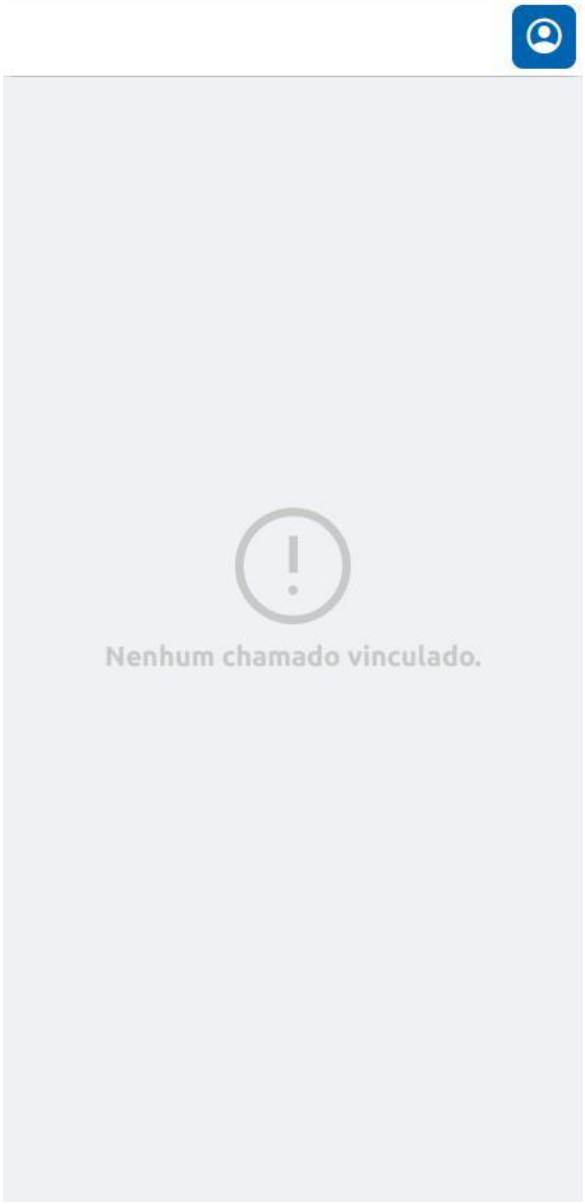
É apresentado um módulo de chamado atual apresentado na Figura 13, onde é descrito as informações do chamado vinculado a ambulância, sendo local, nível de emergência, um mapa com a rota para do local e botões com funções para alterar o estado do chamado. Conforme as alterações do estado do chamado, é notificado ao hospital.

Figura 13 – Chamado atual interface motorista

Fonte: Feito pelos Autores, 2024.

Quando nenhum chamado está vinculado é apresentado uma tela de espera, informando que nenhum chamado foi vinculado, podendo ser visualizado na Figura 14.

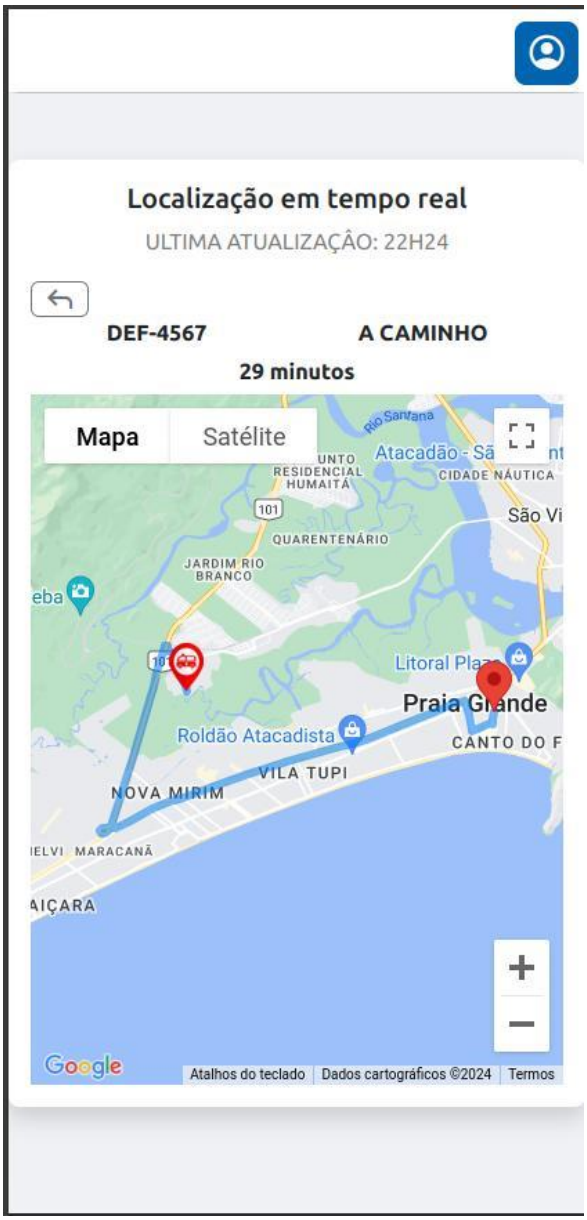
Figura 14 – Nenhum chamado atual interface motorista



Fonte: Feito pelos Autores, 2024.

Além dessas funcionalidades, também é possível visualizar a rota a ser perseguida através de um mapa, demostrado na Figura 15.

Figura 15 – Mapa rota ao chamado interface motorista



Fonte: Feito pelos Autores, 2024.

A equipe de desenvolvimento faz a utilização dos recursos do *framework Spring Boot*, como o *Spring Security* para autenticação de usuários por meios de *tokens*, mapeamento de entidades

através do JPA (*Java Persistence* API)[32] e recursos de geolocalização com *Javascript*[33] e renderização de mapas e rotas com *Google Maps*[34] API (*Application Programming Interface*)[35].

7 TECNOLOGIAS UTILIZADAS NO PROJETO

Tecnologias em tempo real: As tecnologias em tempo real permitem que os usuários interajam com aplicações *web* ou sistemas de forma praticamente instantânea. Isso é de extrema importância para aplicações como chat ao vivo, jogos online, e plataformas de *streaming*[36], onde a resposta rápida é essencial. É importante destacar que o desenvolvimento de aplicações em tempo real requer uma combinação de linguagens de programação, como *JavaScript* para *front-end*, e tecnologias de *back-end* como Node.js, que permitem a comunicação em tempo real entre o servidor e o cliente.

Medidas de segurança cibernética: "A segurança cibernética é fundamental para proteger sistemas e dados contra ameaças online. As medidas de segurança cibernética incluem a criptografia de dados, autenticação forte, *firewalls*, e a implementação de políticas de segurança robustas." (BRAVO, Christian Ali, 2023).

7.1 VISUAL STUDIO CODE

O Visual Studio Code (2024) ou comumente chamado apenas de *VSCode* é um editor de código-fonte gratuito desenvolvido pela Microsoft, distribuído para os principais sistemas operacionais (Windows, macOS, Linux). Possui várias vantagens como sua fluidez, versatilidade e recursos de estilização da interface.

- Diversas extensões e pacotes que dão suporte as linguagens.
- Personalização da interface gráfica.

[32] JPA é uma especificação que permite a persistência de objetos Java em um banco de dados relacional. Ela define um conjunto de conceitos para mapear objetos Java para tabelas de banco de dados e vice-versa, facilitando a manipulação de dados sem a necessidade de escrever consultas SQL manualmente.

[33] JavaScript é uma linguagem de programação amplamente utilizada para desenvolvimento web. Criada inicialmente para adicionar interatividade a páginas da web, ela permite que desenvolvedores criem elementos dinâmicos como animações, formulários interativos e atualizações de conteúdo sem recarregar a página.

[34] O Google Maps é um serviço de mapeamento online desenvolvido pela Google oferecem uma variedade de funcionalidades para ajudar os usuários a navegarem e explorar o mundo.

[35] Uma API (Interface de Programação de Aplicações) é um conjunto de definições e protocolos que permite que diferentes softwares se comuniquem entre si. A API do Google Maps, especificamente, é uma interface oferecida pela Google que permite aos desenvolvedores integrarem funcionalidades de mapeamento e localização em seus próprios aplicativos e websites.

[36] Diferentemente de downloads e uploads tradicionais, o streaming fornece som, vídeo ou outra informação multimídia diretamente ao usuário sem a necessidade de baixar o arquivo completo.

- Integração com APIs e aplicativos *web* na nuvem, além do *Git* que tem um desempenho fluído na plataforma, o que ajuda no versionamento de código facilitando o trabalho com outros desenvolvedores.
- Terminal integrado podendo utilizar de comandos da IDE[37] (*Integrated Development Environment*) sem sair do *software* e sua ferramenta de depuração que é extremamente útil para analisar os erros do código e corrigi-los.

Como é mostrado ícone do *VSCode* na Figura 16 abaixo:

Figura 16 – Visual Studio Code

Fonte: https://uxwing.com/visual-studio-code-icon/

[37] IDE é um ambiente integrado de desenvolvimento, ou seja, um software feito para escrever programas.

7.2 TYPESCRIPT

De acordo com a documentação oficial do *TypeScript* (2024) essa é uma linguagem que verifica erros antes da compilação do código e faz isso com base nos tipos dos valores, o que faz dessa linguagem um verificador de tipo estático. O *TypeScript* aceita qualquer código de *sintaxe*[38] do *JavaScript*, porém ele determinará um erro se o código tiver por exemplo uma operação específica do *JavaScript*, pois ele foi feito para aceitar um programa correto, mas procura o máximo de erros comuns possíveis. Porém o *TypeScript* não altera o tempo de execução de um código *JavaScript* esse é um princípio fundamental da linguagem, podendo assim transcrever um código do *JavaScript* para o *TypeScript* com tranquilidade.

"Grosso modo, uma vez que o compilador do *TypeScript* termina de verificar seu código, ele apaga os tipos para produzir o código 'compilado' resultante. Isso significa que, depois que seu código for compilado, o código JS simples resultante não terá informações de tipo." (TYPESCRIPT DOCS, 2024).

Na Figura 17 é possível visualizar a imagem do ícone:

Figura 17 – TypeScript

Fonte: https://uxwing.com/typescript-programming-language-icon/

[38] Sintaxe é a forma estrutural de se escrever o código, de acordo com suas regras da linguagem.

7.3 ANGULAR

Conforme a documentação descreve o "*Angular* é estrutura de design de aplicativos e plataforma de desenvolvimento para a criação de aplicativos de página única eficientes e sofisticados, construído sobre *TypeScript*" (ANGULAR DOCS, 2024). Esse *framework* possui:

- Tem uma estrutura baseada em componentes para construir sistemas web escaláveis.
- Possui diversas bibliotecas integradas que abrangem uma ampla variedade de recursos, incluindo roteamento, gerenciamento de formulários, comunicação cliente-servidor e outros.
- Um conjunto de ferramentas que facilitam e ajudam os desenvolvedores tornando o trabalho mais rápido e eficiente.

Com este *framework*, você aproveita uma plataforma que pode ser estabelecida em projetos de desenvolvedor único até aplicativos de nível empresarial. O melhor de tudo é que o ecossistema *Angular* consiste em um grupo diversificado de mais de 1,7 milhão de desenvolvedores, autores de bibliotecas e criadores de conteúdo. É recomendável que se tenha conhecimento prévio das linguagens fundamentais de *front-end* como *HTML*, *CSS*[39] e *JavaScript* antes de fazer uso do framework. Seu logo pode ser observado na Figura 18:

Figura 18 – Angular

Fonte: https://www.alura.com.br/artigos/novidades-angular-17

[39] CSS (Cascating Style Sheets) é uma linguagem de marcação utilizada para estilizar os itens de uma página web diretamente em tags do HTML.

7.4 JAVA

“De acordo com a documentação oficial, Java é uma linguagem de programação e plataforma de computação lançada pela primeira vez pela Sun Microsystems em 1995. Ela evoluiu desde seus humildes começos para alimentar uma grande parte do mundo digital atual, fornecendo uma plataforma confiável sobre a qual muitos serviços e aplicativos são construídos. Novos produtos inovadores e serviços digitais projetados para o futuro continuam a depender do Java.” (ORACLE, 2024).

Java é uma linguagem orientada a objetos além de ter sido criada para executar em qualquer ambiente que tiver uma máquina virtual Java instalada, fazendo com que seja portável e independente de uma plataforma. Sua robustez garante uma segurança maior em prevenção de erros, tratamento de exceções, verificação de tipo em tempo de compilação e execução. A imagem do *Java* pode ser vista na Figura 19:

Figura 19 - Java

Fonte: https://logos-world.net/wp-content/uploads/2022/07/Java-Logo.png

7.5 SPRING BOOT

“De acordo com a documentação oficial, o Spring Boot é uma visão orientada adotada da plataforma Spring e de bibliotecas de terceiros para que você possa começar com o mínimo de complicações. A maioria das aplicações Spring Boot requer uma configuração mínima do Spring.

Seus principais recursos são:

- Criar aplicações Spring autônomas
- Incorporar Tomcat, Jetty ou Undertow diretamente (sem a necessidade de implantar arquivos WAR)
- Fornecer dependências 'starter' orientadas para simplificar sua configuração de compilação
- Configurar automaticamente o Spring e bibliotecas de terceiros sempre que possível
- Oferecer recursos prontos para produção, como métricas, verificações de saúde e configuração externalizada
- Absolutamente sem geração de código e sem exigência de configuração XML”

(SPRING FRAMEWORK, 2023).

Como é mostrado ícone do *Spring Boot* na Figura 20 a seguir:

Figura 20 – Spring Boot

Fonte: https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcQpECdGbLgv5jpl_sOzKf5BQfRwIUnCq5YsaBDTVDkTIg&s

7.6 SPRING SECURITY

O *Spring Security* é um *framework* poderoso e altamente personalizável para autenticação e controle de acesso, sendo o padrão de facto para proteger aplicações baseadas no *Spring*. Ele fornece tanto autenticação quanto autorização para aplicações *Java*, permitindo uma fácil extensão para atender a requisitos personalizados.

O *Spring Security* é parte do projeto *Spring* e oferece um sistema de autenticação e autorização de alto nível e altamente personalizável para aplicações *Java*. Ele é a solução oficial para implementar recursos de segurança em aplicações *Spring Boot*, mas também pode ser usado em conjunto com outras *frameworks*.

Além do sistema de autenticação e autorização, o *Spring Security* inclui funcionalidades adicionais que aumentam a segurança das aplicações *Java*, como proteção contra-ataques como *session fixation*, *clickjacking* e *cross* site *request forgery*. Ele também oferece integração com a *Servlet* API e, opcionalmente, com o *Spring Web* MVC, facilitando a instalação através de um starter *Spring Boot*. Observe na Figura 21 seu logo:

Figura 21 – Spring Security

Fonte: https://spring.io/projects/spring-security

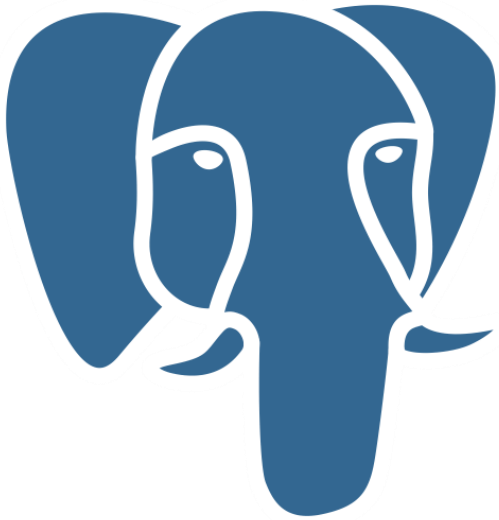
7.7 POSTGRESQL

“O PostgreSQL é um sistema de banco de dados objeto-relacional poderoso e de código aberto que utiliza e estende a linguagem SQL, combinando-a com diversas funcionalidades que armazenam e dimensionam de forma segura as cargas de trabalho de dados mais complexas. As origens do PostgreSQL remontam a 1986 como parte do projeto POSTGRES na Universidade da Califórnia em Berkeley, contando com mais de 35 anos de desenvolvimento ativo na plataforma central.

O PostgreSQL conquistou uma sólida reputação devido à sua arquitetura comprovada, confiabilidade, integridade de dados, conjunto robusto de recursos, extensibilidade e à dedicação da comunidade de código aberto por trás do software, que consistentemente entrega soluções eficientes e inovadoras. O PostgreSQL é compatível com todos os principais sistemas operacionais, é ACID-compliant desde 2001 e possui extensões poderosas, como o popular extensor de banco de dados geoespacial *PostGIS*. Não é surpresa que o PostgreSQL tenha se tornado o banco de dados relacional de código aberto preferido por muitas pessoas e organizações.” (GRUPO GLOBAL DE DESENVOLVIMENTO POSTGRESQL, 2024).

Pode-se ver o ícone do *PostgreSQL* na Figura 22:

Figura 22 - PostgreSQL



Fonte: https://uxwing.com/postgresql-icon/

7.8 GOOGLE MAPS API

De acordo com a documentação oficial, a Google Maps API (2023), que significa **I**nterface de Programação de Aplicativos (API) do Google Maps, é um conjunto de ferramentas que permite a desenvolvedores incorporar funcionalidades do Google Maps em seus sites e aplicativos.

Imagine o Google Maps como um grande banco de dados com informações geográficas. A API funciona como uma ponte, possibilitando que outras aplicações "conversem" com esse banco de dados e utilizem seus recursos.

Usando a Google Maps API, você pode exibir mapas, marcar locais, calcular rotas dentre outras funcionalidades.

A Google Maps API oferece diversas funcionalidades e personalizações, permitindo que desenvolvedores criem experiências ricas e interativas para os usuários. (GOOGLE MAPS PLATFORM, 2024).

Ícone do *Google Maps* na Figura 23 a seguir:

Figura 23 – Google Maps

Fonte: https://uxwing.com/google-map-icon/

7.9 GPS

O GPS ou Sistema de Posicionamento Global, é um sistema de navegação por satélite que fornece informações de localização precisa em qualquer lugar do mundo. Ele foi desenvolvido pelo Departamento de Defesa dos Estados Unidos e entrou em operação em 1973, mas só se tornou totalmente operacional em 1995. O GPS utiliza uma constelação de satélites em órbita para transmitir sinais que são recebidos por dispositivos receptores na Terra, permitindo que eles determinem sua posição com alta precisão.

7.9.1 Integração com o gps

A implementação da integração GPS no projeto é realizada por meio do *TypeScript*. Através do *TypeScript*, é utilizado informações sobre o ambiente do navegador, solicitando permissões necessárias para acessar a geolocalização. Com as informações de geolocalização, é possível determinar coordenadas de latitude e longitude, altitude e velocidade do dispositivo receptor, facilitando a navegação, geolocalização e mapeamento em aplicativos e serviços baseados em localização.

7.10 MICROSOFT AZURE

Segundo a documentação oficial do *Microsoft Azure*: “O Azure é uma plataforma de nuvem projetada para simplificar o processo de compilação de aplicativos modernos. Se você optar por hospedar seus aplicativos inteiramente no Azure ou estender seus aplicativos locais com os serviços do Azure, o Azure ajuda você a criar aplicativos escalonáveis, confiáveis e de fácil manutenção.

O Azure oferece suporte às linguagens de programação mais populares em uso atualmente, incluindo *Python*, *JavaScript*, *Java*, *.NET e Go*. Com uma biblioteca *SDK* abrangente e amplo suporte em ferramentas que você já usa como *VS Code*, *Visual Studio*, *IntelliJ e Eclipse*, o Azure foi projetado para aproveitar as habilidades que você já tem e torná-lo produtivo imediatamente.” (MICROSOFT AZURE, 2024). A Figura 24 mostra a imagem do *Microsoft Azure*:

Figura 24 - Microsoft Azure

Fonte: https://uxwing.com/azure-icon/

7.11 SWAGGER

De acordo com o a fonte sobre o *Swagger* é descrita que “Swagger é um conjunto poderoso e fácil de usar de ferramentas de desenvolvedor de API para equipes e indivíduos, permitindo o desenvolvimento em todo o ciclo de vida da API, desde o design e documentação até o teste e a implantação.

Swagger consiste em uma combinação de ferramentas de código aberto, gratuitas e disponíveis comercialmente que permitem a qualquer pessoa, desde engenheiros técnicos até gerentes de produto inteligentes, construir APIs incríveis que todos adoram.

O Swagger foi desenvolvido pela *SmartBear Software*, líder em ferramentas de qualidade de software para equipes. *SmartBear* está por trás de alguns dos maiores nomes do espaço de software, incluindo *Swagger*, *SoapUI* e *QAComplete*.” (ABOUT SWAGGER, 2024).

Visualize na figura 25 o *Swagger*:

Figura 25 - Swagger

Fonte: https://en.wikipedia.org/wiki/Swagger_%28software%29

8 CONSIDERAÇÕES FINAIS

A ideia de construir um sistema de controle e chamados, inicialmente surgiu através de questões particulares de profissionais da saúde familiares de dois integrantes do grupo. Depois de planejar e discutir sobre essa ideia, foi tomado o primeiro passo em desenvolvimento. Seguindo a coleta de requisitos e uma metodologia bibliográfica e de estudo de campo, o sistema teve um esboço inicial. Após estas pesquisas, foi feita a divisão de tarefas entre os integrantes do grupo pensando nas habilidades mais sólidas de cada um.

O objetivo final deste trabalho chegou na etapa de auxiliar no atendimento pré-hospitalar e futuramente na ampla área da saúde, pois foram feitas anotações de melhorias futuras. Criar uma plataforma para monitorar e ter controle de ambulâncias poderia inovar a forma como é feita a recepção do paciente no hospital e locomoção no trajeto da ambulância até o acidente. Este sistema foi desenvolvido para atender estas questões, desta forma é possível ter uma ferramenta visual que ajuda a ter noção de tempo e preparo para as urgências, a escolha das ferramentas para a criação foi feita para garantir segurança e uma aplicação responsiva para os usuários, projetado para ser visualizado em duas plataformas sendo para a unidade móvel e outra para o hospital, de maneira simples e fácil de ser usada.

Após o término do *Live Saver Locator*, obteve-se o resultado esperado atingindo de maneira concisa a conclusão da idealização inicial. Os requisitos funcionais foram atingidos concluindo o sistema.

REFERÊNCIAS

Profissional da saúde. Enfermeiro. Entrevista concedida ao Caio Rodrigo de Oliveira. Mongaguá, 24 de mai. 2024.

Ministério da Saúde. 2020. **Plano Nacional de Saúde Digital.** Disponível em: https://pesquisa.bvsalud.org/bvsms/resource/pt/biblio-1039752 Acesso em: 14 de abr. 2024.

AUTOLAC. **Atendimento humanizado: o que é e qual a importância na área da saúde?**. Disponível em: https://autolac.com.br/blog/atendimento-humanizado-na-area-de-saude/. Acesso em: 28 de mai. 2024.

TUMELERO, Naína. **Metodologia bibliográfica. Blog da Mettzer**, 23 set. 2019. Disponível em: https://blog.mettzer.com/pesquisa-bibliografica/. Acesso em: 5 jun. 2024.

TUMELERO, Naína. **Pesquisa de levantamento. Blog da Mettzer**, 11 out. 2019. Disponível em: https://blog.mettzer.com/pesquisa-de-levantamento/. Acesso em: 5 jun. 2024.

Corpo de Bombeiros Militar do Distrito Federal. (2007). **Manual de Atendimento Pré-hospitalar**. Brasília: Corpo de Bombeiros Militar do Distrito Federal.

LINUX FOUNDATION. The Linux Foundation. Disponível em: https://www.linuxfoundation.org/. Acesso em: 5 jun. 2024.

ESCOLA DNC**. A importância dos requisitos funcionais e não funcionais no desenvolvimento de software**. Disponível em: https://www.escoladnc.com.br/blog/a-importancia-dos-requisitos-funcionais-e-nao-funcionais-no-desenvolvimento-de-software/#:~:text=Requisitos%20Funcionais%20e%20Requisitos%20N%C3%A3o%20Funcionais%201%20Requisitos,fundamental%20para%20o%20desenvolvimento%20de%20software%20de%20qualidade. Acesso em: 5 jun. 2024.

MINISTÉRIO DA SAÚDE. SAMU 192. Disponível em: https://www.gov.br/saude/pt-br/assuntos/saude-de-a-a-z/s/samu-192. Acesso em: 5 jun. 2024.

ORACLE. **What is Java?**. Disponível em: https://www.java.com/en/download/help/whatis_java.html. Acesso em: 5 jun. 2024.

GOOGLE. Google Maps APIs. Disponível em: https://developers.google.com/maps/apis-by-platform. Acesso em: 5 jun. 2024.

FERRAZ, Raphaela; MARQUES, Victor; SANTOS, Anna; MOREIRA, Hillary. **Cyber Security Information.** 2022. Disponível em: https://github.com/VictorGM01/cyber_sec_info/blob/main/relatorio.md. Acesso em: 29 de fev. 2024.

BRAVO, Christian Ali. **As linguagens de programação mais usadas em cibersegurança**. 2023. Disponível em: https://www.welivesecurity.com/pt/seguranca-digital/as-linguagens-de-programacao-mais-usadas-em-ciberseguranca. Acesso em: 29 fev. 2024.

TYPESCRIPT DOCS. **TypeScript for New Programmer**. Disponível em: https://www.typescriptlang.org/docs/handbook/typescript-from-scratch.html. Acesso em: 01 de mar. 2024.

ANGULAR DOCS. **What is Angular?**. Disponível em: https://angular.io/guide/what-is-angular. Acesso em: 02 de mar. 2024.

ORACLE. **What is java technology and why do i need it?**. Disponível em: https://www.java.com/en/download/help/whatis_java.html . Acesso em: 02 de mar. 2024.

SPRING FRAMEWORK. **Spring Boot**. Disponível em: https://spring.io/projects/spring-boot. Acesso em: 02 de mar. 2024.

SPRING SECURITY. **Spring Security**. Disponível em: https://spring.io/projects/spring-security. Acesso em: 28 de abr. 2024.

GRUPO GLOBAL DE DESENVOLVIMENTO POSTGRESQL. **What is PostgreSQL ?**. Disponível em: https://www.postgresql.org/about/. Acesso em: 02 de mar. 2024.

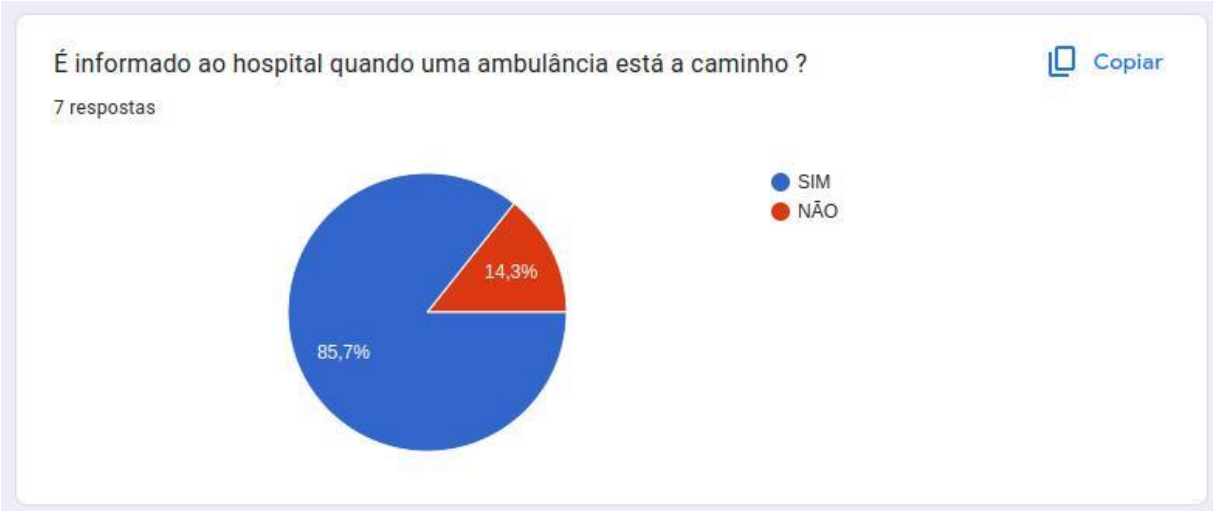
GOOGLE MAPS PLATFORM. **Visão geral**. Disponível em: https://developers.google.com/maps/documentation?hl=pt-br r. Acesso em: 02 de mar. 2024.

MICROFSOFT AZURE. **Visão geral do Azure para desenvolvedores**. Disponível em: https://learn.microsoft.com/pt-br/azure/developer/intro/azure-developer-overview. Acesso em: 05 de mar. 2024.

SWAGGER. **About Swagger**. Disponível em: https://swagger.io/about/. Acesso em: 05 de mar. 2024

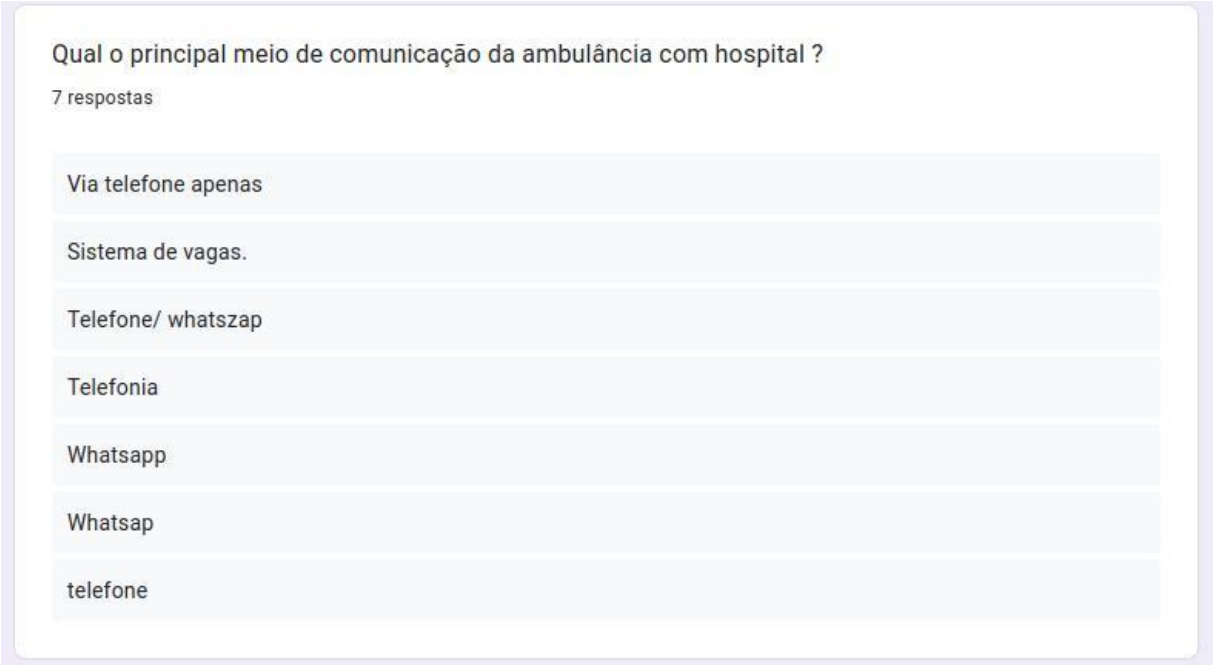
APÊNDICE A – FORMULARIO RELAÇÃO HOSPITAL AMBULÂNCIA

Apêndice A.1 - Questão 1



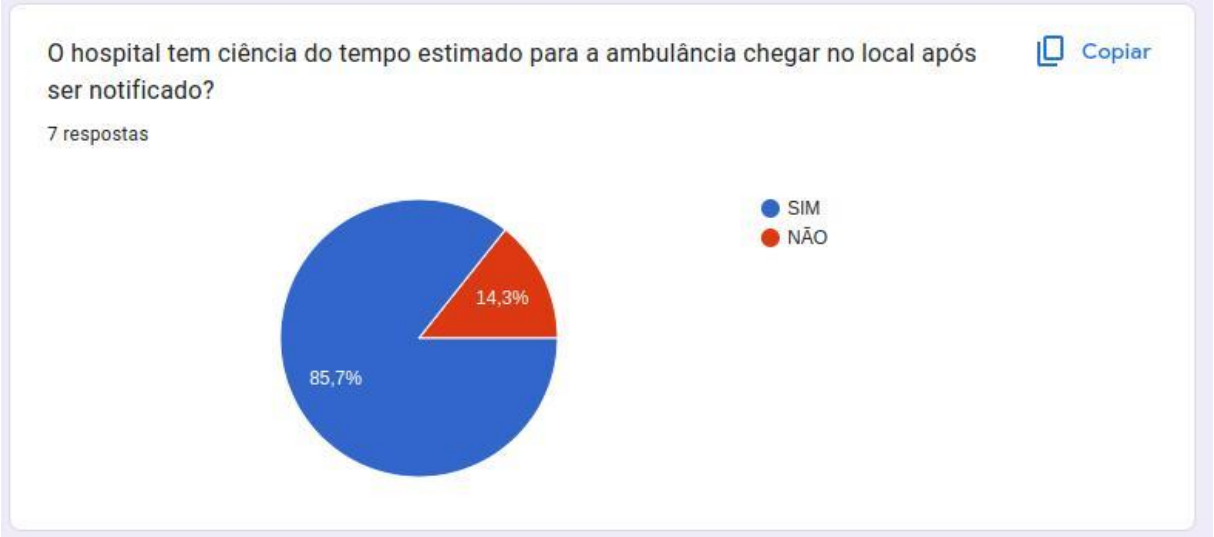
Fonte: Feito pelos Autores, 2024.

Apêndice A.2 - Questão 2



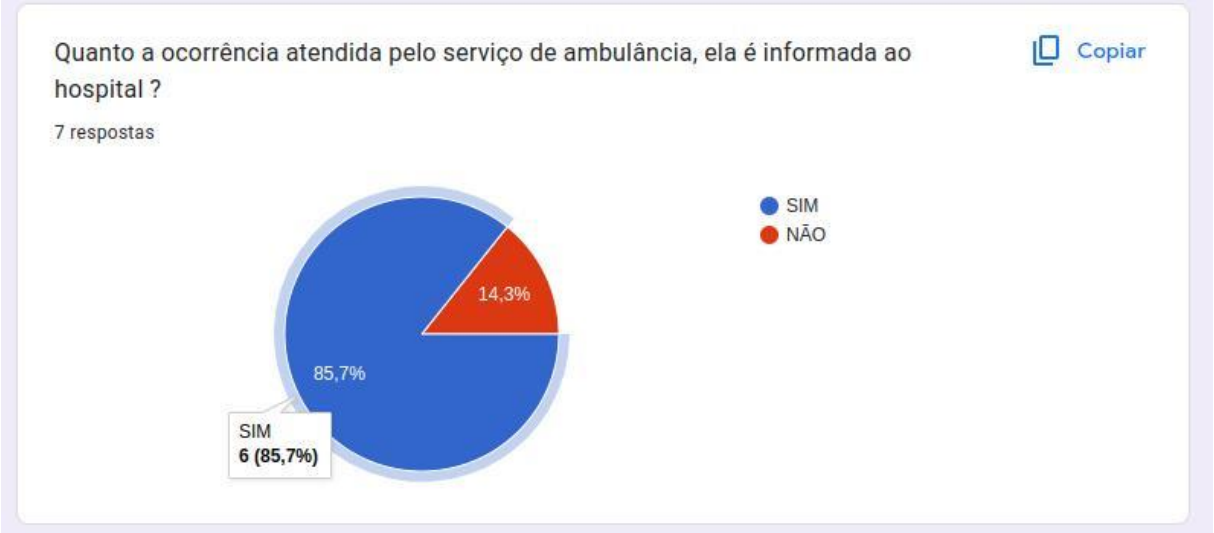
Fonte: Feito pelos Autores, 2024.

Apêndice A.3 - Questão 3



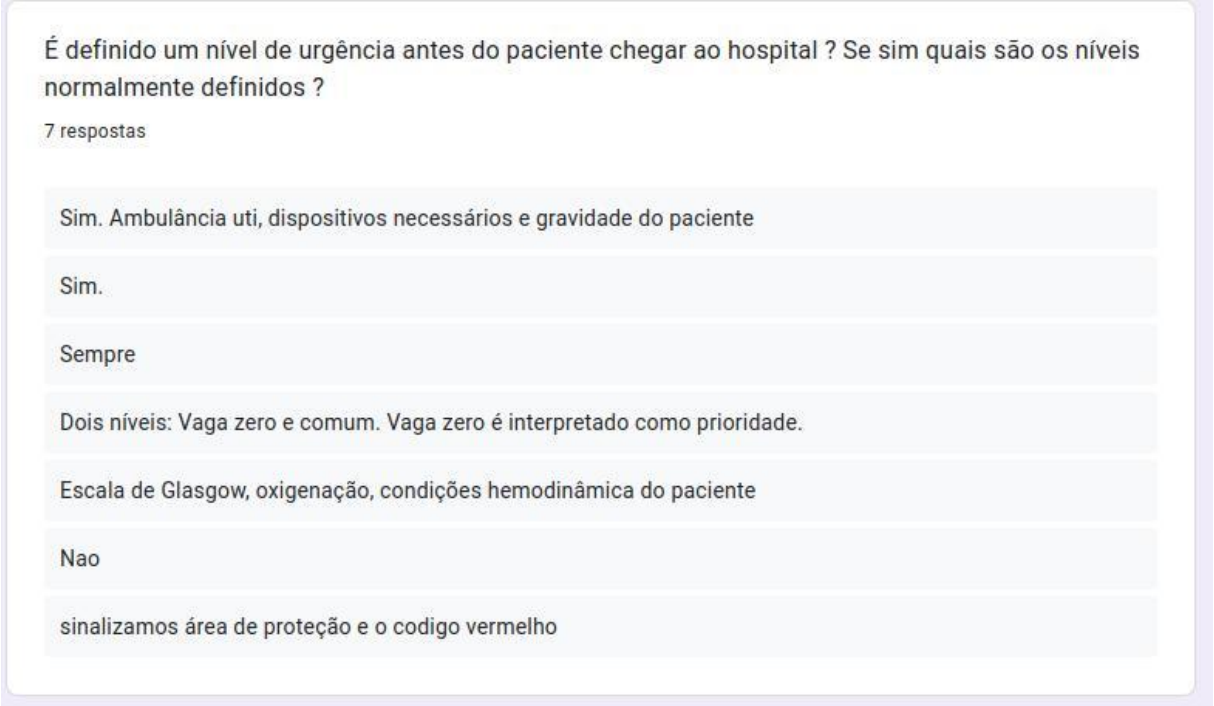
Fonte: Feito pelos Autores, 2024.

Apêndice A.4 - Questão 4



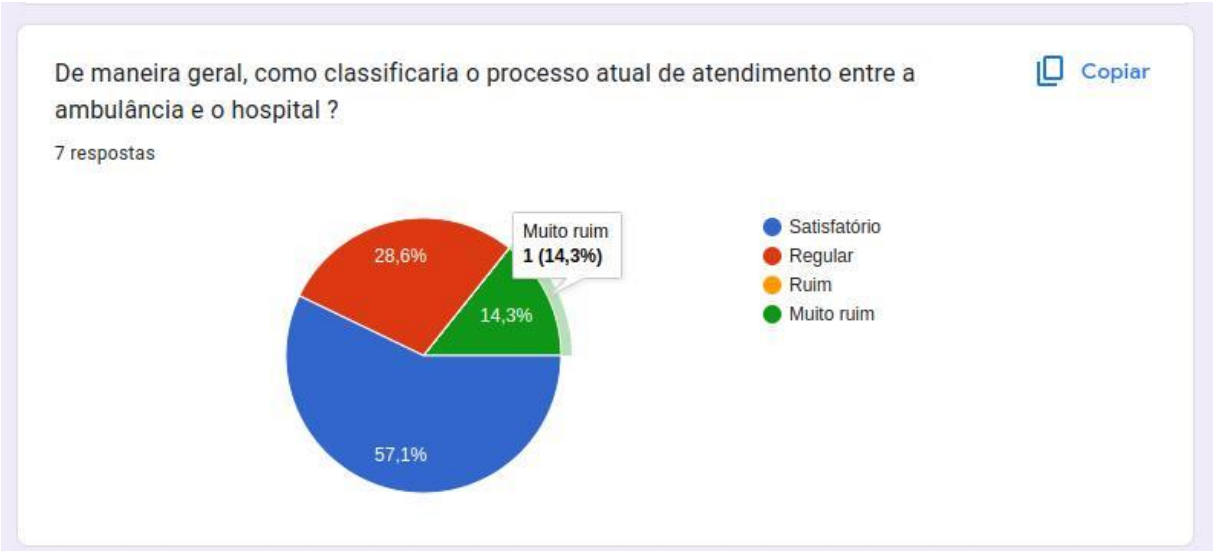
Fonte: Feito pelos Autores, 2024.

Apêndice A.5 - Questão 5



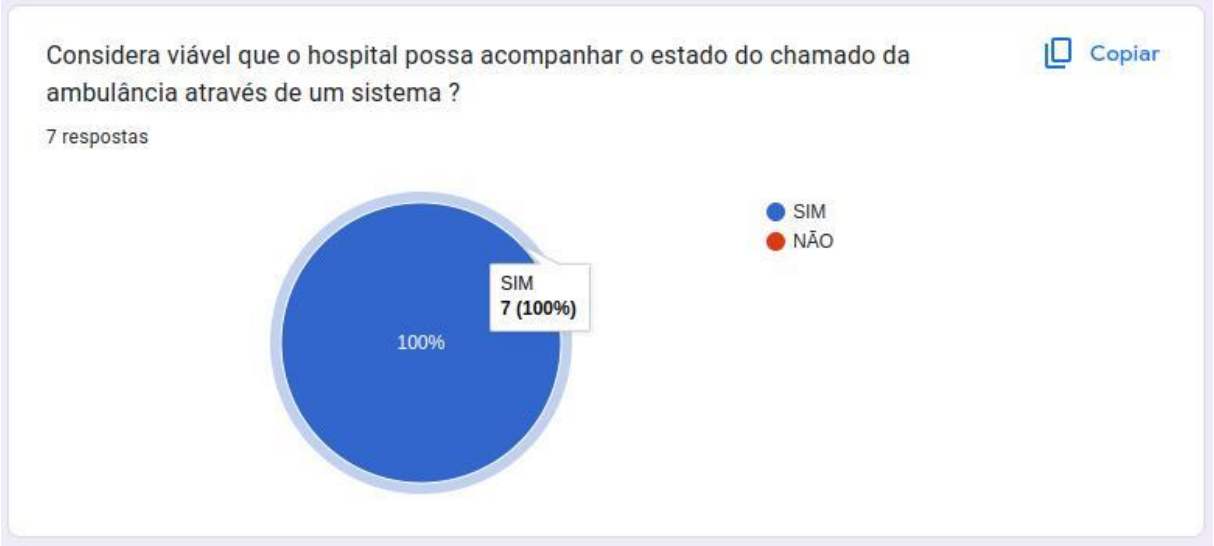
Fonte: Feito pelos Autores, 2024.

Apêndice A.6 - Questão 6



Fonte: Feito pelos Autores, 2024.

Apêndice A.7 - Questão 7

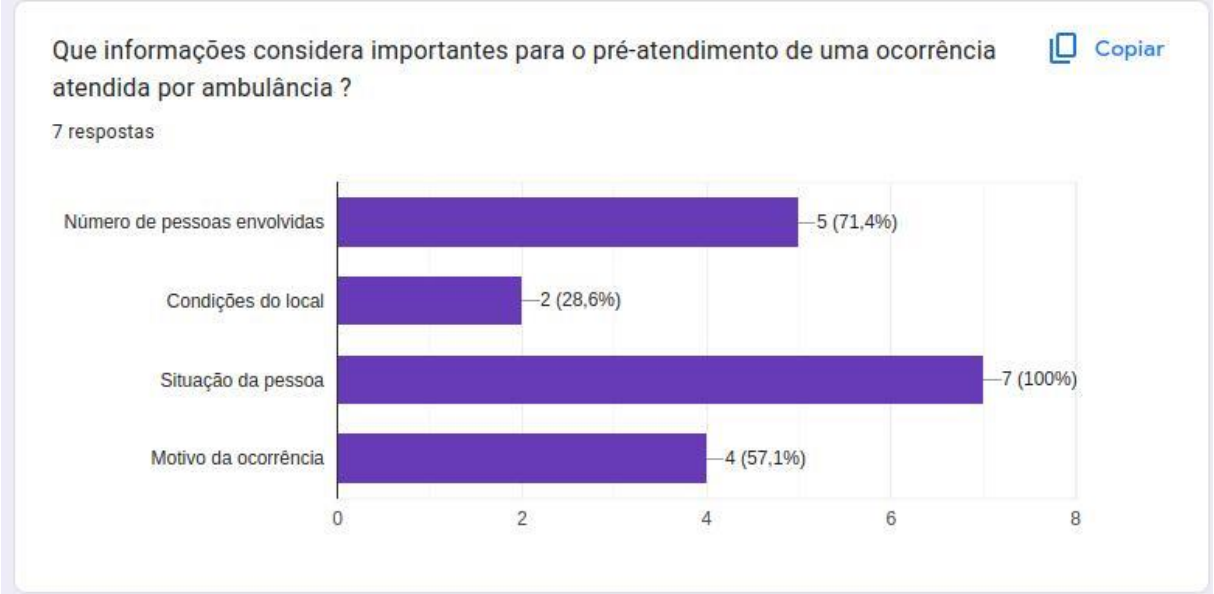


Fonte: Feito pelos Autores, 2024.

Apêndice A.8 - Questão 8

Fonte: Feito pelos Autores, 2024.

Apêndice A.9 - Questão 9



Fonte: Feito pelos Autores, 2024.

APÊNDICE B – DIAGRAMA DE FLUXO DA ENTREVISTA INFORMAL E ANÔNIMA