INSTITUTO FEDERAL DO ESPIRITO SANTO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

EMANOEL MARTINS VIEIRA

MVP DE UMA APLICAÇÃO WEB DE TELEMEDICINA PARA UNIDADES BÁSICAS DE SAÚDE UTILIZANDO A PLATAFORMA DE COMUNICAÇÃO TWILIO

Cachoeiro de Itapemirim
2022

EMANOEL MARTINS VIEIRA

MVP DE UMA APLICAÇÃO WEB DE TELEMEDICINA PARA UNIDADES BÁSICAS DE SAÚDE UTILIZANDO A PLATAFORMA DE COMUNICAÇÃO TWILIO

Trabalho de Conclusão de Curso apresentado à Coordenadoria do Curso de Sistemas de Informação do Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientador: DSc. Raul de Souza Brandão

Cachoeiro de Itapemirim

2022

(Biblioteca do Campus Cachoeiro de Itapemirim)

V658m Vieira, Emanoel Martins.

MVP de uma aplicação web de telemedicina para unidades básicas de saúde utilizando a plataforma de comunicação twilio / Emanoel Martins Vieira. - 2022.
105 f. : il. ; 30 cm.

Orientador: Raul de Souza Brandão

TCC (Graduação) Instituto Federal do Espírito Santo, Campus Cachoeiro de Itapemirim, Sistemas de Informação, 2022.

1. Tecnologia da Informação. 2. Comunicação e tecnologia. 3. Medicina. 4. Trabalho. I. Brandão, Raul de Souza. II.Título III. Instituto Federal do Espírito Santo.

CDD: 005.1

Bibliotecário/a: Jacqueline Machado Silva CRB-ES nº 640

MINISTÉRIO DA EDUCAÇÃO
INSTITUTO FEDERAL DO ESPÍRITO SANTO

FOLHA DE APROVAÇÃO-TCC nº 25/2022-CAI-CCSI
Protocolo nº 23151.005222/2022-12

Cachoeiro De Itapemirim-ES, 20 de dezembro de 2022

EMANOEL MARTINS VIEIRA
TÍTULO DO TRABALHO DE CONCLUSÃO DE CURSO

Trabalho de Conclusão de Curso apresentado à Coordenadoria de Sistemas de Informação do Instituto Federal do Espírito Santo, como requisito parcial para obtenção de título de Bacharel em Sistemas de Informação.

Aprovado em 19 de dezembro de 2022

COMISSÃO EXAMINADORA

D.sc. Raul de Souza Brandão
Instituto Federal Do Espírito Santo
Orientador

M.sc. Daniel José Ventorim Nunes
Instituto Federal Do Espírito Santo

M.Sc. Antonio Izo Júnior
Instituto Federal do Espírito Santo

(Assinado digitalmente em 21/12/2022 11:03)
ANTONIO IZO JUNIOR
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO-SUBSTITUTO
CAI-CCSI (11.02.18.01.08.02.13)
Matrícula: 3240051

(Assinado digitalmente em 22/12/2022 09:38)
DANIEL JOSE VENTORIM NUNES
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCLI (11.02.18.01.08.02.06)
Matrícula: 1918045

(Assinado digitalmente em 20/12/2022 16:10)
RAUL DE SOUZA BRANDAO
PROFESSOR DO ENSINO BASICO TECNICO E TECNOLOGICO
CAI-CCSI (11.02.18.01.08.02.13)
Matrícula: 2764324

Para verificar a autenticidade deste documento entre em **https://sipac.ifes.edu.br/public/documentos/index.jsp** informando seu número: **25**, ano: **2022**, tipo: **FOLHA DE APROVAÇÃO-TCC**, data de emissão: **20/12/2022** e o código de verificação: **4350e2cc94**

DECLARAÇÃO DO AUTOR

Declaro, para fins de pesquisa acadêmica, didática e técnico-científica, que este Trabalho de Conclusão de Curso pode ser parcialmente utilizado, desde que se faça referência à fonte e ao autor.

Cachoeiro de Itapemirim, 20 de Dezembro de 2022.

Emanoel Martins Vieira

Dedico esse trabalho primeiramente a Deus e a todos que de alguma forma contribuíram para que o mesmo fosse realizado.

AGRADECIMENTOS

Agradeço primeiramente a Deus, que me deu forças e sabedoria ao longo deste caminho. A minha mãe dona Ilza Santiago Martins, que sempre me deu incentivo, e contribuiu da melhor forma possível, para que eu continuasse estudando, ao meu pai Sr. Joacir Vieira que sempre apoiou,incentivou e aconselhou, a meu amigo Sebastião Ravera que em momentos difíceis prestou seu apoio, a minha namorada Larissa Marvila Dornelas de Souza pela compreensão, e apoio incondicional durante esta etapa. A minha amiga Danielle Rangel Garcia que esteve presente e auxiliou sempre nos trabalhos acadêmicos,ao meu orientador, Dr Raul de Souza Brandão, pela paciência e orientações, que sem dúvidas brilhou e me inspirou neste trabalho, e a todos os professores que se dedicaram e me ensinaram tudo que sei, a todos os envolvidos, muito obrigado.

RESUMO

Telemedicina é o exercício da medicina através das tecnologias de informação e comunicação com o objetivo de prestar atendimento a pacientes e outros profissionais da saúde de forma online. Esta vem se tornando cada vez mais presente na vida dos brasileiros, aos poucos, as pessoas estão podendo ir em busca de assistência medica sem precisar sair de casa, ou se deslocar até uma posto medico, unidade de saúde ou hospital. No Brasil citando especificamente a cidade de Cachoeiro de Itapemirim, no estado do Espirito Santo, é possível ver a implementação da telemedicina, nos serviços de agendamento de saúde, resultados de exames e vigilância sanitária com informações e materiais educativos. Porém não é disponibilizado nenhum serviço de tele consulta, onde o paciente pode fazer uma consulta com o médico, sem estar fisicamente presente. Este trabalho de conclusão de curso tem como objetivo, implementar o produto mínimo viável(MVP) de uma aplicação WEB de telemedicina com triagem em videoconferência feita por um enfermeiro, e tele consulta feita por um médico. E para este fim foi abordada uma metodologia em que abordo desde a analise de aplicativos de telemedicina já existentes, passando por um levantamento de requisitos até chegar a implementação do MVP integrado com a plataforma de comunicação da twilio.

Palavras-chave: Telemedicina, tele consulta, Unidade básica de saúde

ABSTRACT

Telemedicine is the practice of medicine through information and communication technologies in order to provide care to patients and other health professionals online. This is becoming increasingly present in the lives of Brazilians, little by little, people are able to go in search of medical assistance without having to leave their homes, or travel to a medical center, health unit or hospital. In Brazil, specifically citing the city of Cachoeiro de Itapemirim, in the state of Espirito Santo, it is possible to see the implementation of telemedicine, in health scheduling services, test results and health surveillance with information and educational materials. However, no tele consultation service is available, where the patient can make an appointment with the doctor, without being physically present. This course conclusion work aims to implement the minimum viable product (MVP) of a telemedicine WEB application with screening in videoconference made by a nurse, and tele consultation made by a doctor. And for this purpose, a methodology was approached in which I approach from the analysis of existing telemedicine applications, going through a requirements survey until reaching the implementation of the MVP integrated with the twilio communication platform.

Keywords: Telemedicine, tele consultation, Basic health unit

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

MVP - Produto mínimo viável

OMS - Organização mundial da saúde

APS - Atenção primária a saúde

TICs - Tecnologias de informação e comunicação

UBS - Unidade Básica de saúde

SUS - Sistema Único de Saúde

ESF - Estratégia Saúde da Família

USF - Unidades de Saúde da Família

Casaps - Carteira de Serviços da Atenção Primária à Saúde

CONASS - Conselho Nacional de Secretarias de Saúde

CONASEMS - Conselho Nacional de Secretarias Municipais de Saúde

SBMFC - Sociedade Brasileira de Medicina e a Comunidade

ABEFACO - Associação Brasileira de Assistência Familiar e Comunitária

ABO - Associação Brasileira de Odontologia

ATA - American Telemedicine Association

UNICAMP - universidade universidade estadual de campinas

SBIS - sociedade brasileira de informática em saúde

CID 10 – Classificação Internacional de Doenças

CFM - Conselho Federal de Medicina

CFF - Conselho Federal de Farmácia

ITI - Instituto Nacional de Tecnologia da Informação

ICP-Brasil - Infraestrutura de Chaves Públicas Brasileira

Anvisa - Agência de Vigilância Sanitária

MVC - *Model,View,Controller*

API - *Application Programming Interface*

WEBRTC - *Web Real-Time Communications*

REST - *Representational State Transfer*

SDK - *Software Development Kit*

SID - *String Identifier*

SUMÁRIO

1 INTRODUÇÃO

A telemedicina é a prática médica a distância, que permite a realização de consultas, diagnósticos e tratamentos de forma remota. Ela pode ser realizada por meio de videoconferência, telefone, mensagens de texto e outras plataformas de comunicação.

A telemedicina tem sido cada vez mais utilizada como uma forma de facilitar o acesso a cuidados de saúde para pessoas que vivem em áreas remotas ou que têm dificuldade de se deslocar até um consultório médico. Ela também pode ser útil em situações de emergência, quando o paciente não pode se deslocar até um hospital ou centro de saúde.

A pandemia de COVID-19 foi um importante fator que levou o aumento do uso da telemedicina. Com o objetivo de evitar o contágio pelo vírus, muitos pacientes optaram por consultas virtuais em vez de se deslocarem até um consultório médico. Além disso, as restrições de deslocamento impostas pelas autoridades de saúde também levaram a um aumento na utilização da telemedicina.

Mesmo em meio a pandemia, muitas pessoas tiveram dificuldade em ter acesso à telemedicina. e precisaram ir presencialmente a uma consulta com um médico, isso pode ter sido devido a uma série de fatores, como conectividade, falta de informação ou orientação sobre como acessar esses serviços, desconhecimento de sua existência, ou falta de acesso a uma ferramenta de telemedicina.

Dentre todas as possíveis causas citadas, atribuo enfase, a falta de ferramenta de telemedicina, especialmente na atenção primaria da iniciativa publica.

Pensando neste ponto, a construção deste trabalho tem como objetivo, construir uma ferramenta de telemedicina para auxilio da atenção primaria a saúde, com foco na rede pública, tendo como inspiração o fato de que em 1978 a Organização Mundial da Saúde (OMS), em sua Conferência Internacional sobre Atenção Primária à Saúde (APS), em Alma Ata, reafirmou a saúde como um direito humano fundamental, e a preconizou como a mais importante meta social mundial a ser alcançada, a obtenção

do mais alto nível possível de saúde, e para alcançar este objetivo, é requerido um esforço dos setores da sociedade, setores sociais, econômicos, políticos, além do setor da saúde.

Iniciativas privadas, disponibilizam para os seus clientes serviços como o da telemedicina, mais especificamente a tele-consulta, que facilitam o cotidiano quando há a necessidade de uma consultar com um médico, pelo fato da consulta ser remota em video, e em alguns casos por telefone, não existe a necessidade de sair de casa, o que proporciona conforto, facilidade, economia, e uma grande contribuição no cuidado da saúde e qualidade de vida.

Por outro lado, para quem não possui condições financeiras de ter um plano de saúde, esses benefícios não estão ao alcance, quando há necessidade de consultas, por mais simples que seja, existe a necessidade de se deslocar até a unidade de saúde, enfrentar filas, esperar as vezes por horas dependendo da situação, e finalmente ser atendido.

Voltando a destacar o esforço dos setores da sociedade para alcançar um elevado nível de saúde, nos termos do setor politico, o ministério da Saúde possui programas que visam expandir e melhorar a rede de serviços de saúde, sobretudo da Atenção Primária à Saúde, dentre esses programas para esse trabalho, coloca-se em destaque o Programa Telessaúde Brasil Redes, com os seguintes campos de atuação, Tele-consultoria, tele-diagnostico, tele-monitoramento, tele-regulação, e tele-educação.

Tele-consultoria: Consultoria registrada e realizada entre trabalhadores, profissionais e gestores da área de saúde, por meio de instrumentos de telecomunicação bidirecional, com o fim de esclarecer dúvidas sobre procedimentos clínicos, ações de saúde e questões relativas ao processo de trabalho em saúde, podendo ser em tempo real ou por meio de mensagens offline.

Tele-diagnostico: Consiste em serviço autônomo que utiliza as TICs para a realização de serviços de Apoio ao Diagnóstico, como a avaliação de exames à distância, facilitando o acesso a serviços especializados. Busca reduzir o tempo de diagnóstico possibilitando tratamento para complicações previsíveis por meio do diagnóstico precoce.

Tele-monitoramento: Monitoramento a distância de parâmetros de saúde e/ou doença de pacientes por meio das TICs. O monitoramento pode incluir a coleta de dados clínicos, a transmissão, o processamento e o manejo por um profissional de saúde utilizando sistema eletrônico.

Tele-regulação: Conjunto de ações em sistemas de regulação com intuito de equacionar respostas adequadas às demandas existentes, promovendo acesso e equidade aos serviços, possibilitando a assistência à saúde. Inclui também a avaliação e o planejamento das ações, fornecendo à gestão uma inteligência reguladora operacional. A tele-regulação visa fortalecer o atendimento na Atenção Primária em Saúde, permitindo qualificar e reduzir as filas de espera no atendimento especializado.

Tele-educação: Disponibilização de objetos de aprendizagem interativos sobre temas relacionados à saúde, ministrados a distância por meio de TICs, com foco na aprendizagem no trabalho, que por sua vez, ocorre transversalmente em seus campos de atuação.

Esses campos de atuação podem ser vistos aplicados em um núcleo de telessaúde em cada estado, porém em alguns desses estados, até a data deste trabalho, não é possível acessar e fazer o uso deles, o núcleo do estado do Espirito Santo, é um exemplo disso, porém chamo a atenção para um outro fato, que dentre esses campos de atuação, não tem nenhum campo na área da tele-consulta, então este trabalho visa apresentar o desenvolvimento de um MVP de uma aplicação WEB de tele-consulta, como forma de contribuir com uma ferramenta para elevar o nível de saúde conforme proposto pela OMS, a aplicação terá como alvo, as unidades básicas de saúde de Cachoeiro de Itapemirim, no estado do Espirito Santo, que até a data deste trabalho também não possui nenhuma solução na área da saúde em termos de tele-consulta.

2 OBJETIVOS

2.1 OBJETIVO GERAL

Desenvolver o MVP de uma aplicação WEB de telemedicina, que em uma consulta, conecte médico e paciente em uma video conferência.

2.2 OBJETIVOS ESPECÍFICOS

Para atingir o objetivo geral, este trabalho busca responder os seguintes objetivos específicos:
Fazer o levantamento dos requisitos iniciais do MVP
Implementar o MVP da aplicação Web de telemedicina
Fazer a integração da aplicação com a plataforma de video da Twilio

3 REFERENCIAL TEÓRICO

3.1 UNIDADE BÁSICA DE SAÚDE

As unidades básicas de saúde (UBS), fazem parte do Sistema Único de Saúde(SUS), sendo a porta de entrada do individuo nesse sistema, que foi criado como uma forma de assegurar o direito a saúde para os cidadãos, quando necessitarem de atendimento de pouca ou grande complexidade, de forma a garantir o acesso integral, universal e gratuito a saúde a todos brasileiros. A UBS tem como função promover a atenção integral à saúde básica dos indivíduos, estando presente em locais de fácil acesso, o que a deixa próxima do cotidiano da população, desempenhando papel principal na garantia de acesso à saúde de qualidade.Gomes, Pinto e Cassuce (2021)

3.1.1 Atenção Primaria a Saúde

A Atenção Primária à Saúde (APS) é o primeiro nível de atenção à saúde e caracteriza-se por uma gama de intervenções em saúde nos níveis individual e coletivo, englobando promoção e proteção da saúde, prevenção de doenças, diagnóstico, tratamento, reabilitação, redução de danos e manutenção da saúde, com o objetivo de desenvolver uma atenção integral que tenha impacto positivo na situação de saúde das comunidades. É a principal porta de entrada do SUS e o centro de comunicação com toda a rede de atenção do SUS e deve se pautar pelos princípios da universalidade, acessibilidade, continuidade do cuidado, atenção integral, responsabilidade, humanização e equidade. Isso significa que a APS atua como um filtro capaz de organizar o fluxo de poder nas redes de saúde, das mais simples às mais complexas. No Brasil, a atenção primária é desenvolvida com maior grau de descentralização e capilaridade e acontece no local mais próximo da vida das pessoas. Existem várias estratégias governamentais relacionadas, uma delas é a Estratégia Saúde da Família (ESF), que presta atendimento multidisciplinar às comunidades por meio das Unidades de Saúde da Família (USF), por exemplo.Consultas, exames, vacinações, radiografias e outros procedimentos estão à disposição dos utentes da USF. Hoje, existe uma Carteira de Serviços da Atenção Primária à Saúde (Casaps) para auxiliar os gestores comunitários na tomada de decisões e levar o conhecimento da população sobre o que se encontra

na APS. Inclui também outras iniciativas, como o programa Saúde na Hora e Médicos para o Brasil. Saúde (2020b)

3.1.2 Carteira de Serviços da atenção primária a saúde

A Carteira de Serviços de Atenção Primária à Saúde (CaSAPS) é resultado de intensa colaboração e discussão entre o Conselho Nacional de Secretarias de Saúde (CONASS), o Conselho Nacional de Secretarias Municipais de Saúde (CONASEMS), a Sociedade Brasileira de Medicina e a Comunidade (SBMFC).), a Associação Brasileira de Assistência Familiar e Comunitária (ABEFACO), a Associação Brasileira de Odontologia (ABO) e a Secretaria de Atenção Primária à Saúde. Saúde (2020a).

3.2 TELEMEDICINA

A Telemedicina é uma forma de fazer o uso das tecnologias de informação e comunicação na área da saúde, para atendimento e assistência médica a pacientes Maldonado, Marques e Cruz (2016), para Strehle e Shabde (2006) Telemedicina significa “Cura à distancia”, com a utilização das tecnologias de informação e comunicação. E Barbosa, Pereira e Martins (2019) também complementa dizendo que a telemedicina é uma inovação em saúde, que auxilia a areá da saúde usando tecnologias de informação e comunicação para diferentes atividades à distancia relacionadas a saúde. A telemedicina vai além da definição de oferta de serviços de saúde por telecomunicação remota, serviços voltados para a educação e treinamento de profissionais da saúde, também é telemedicina. Maldonado, Marques e Cruz (2016) também afirma que a educação medica a distancia, também faz parte da definição de telemedicina.

Juntamente com o termo telemedicina, também é encontrado o termo telessaúde, que de acordo com a Associação Americana de Telemedicina (ATA) “American Telemedicine Association”. É à conexão do paciente com o hospital, quando não é necessário atendimento presencial, ou quando este não é possível, à telessaúde irá permitir o paciente receber atendimento, consultar, obter informações sobre uma condição ou tratamento, providenciar prescrições e receber um diagnóstico. Marcolino et al. (2013) apresenta a telessaúde como um conceito que é mais abrangente, e multidisciplinar, que envolve o atendimento de cuidado com a saúde, para áreas mais especificas como odontologia, psicologia, enfermagem, fisioterapia e fonoaudiologia. O conceito de

telessaúde também é complementado por Sabbatini (2012) que a descreve como uma área mais ampla, envolvendo, transmissão, armazenamento, interação, disponibilização de tudo que se refira as atividades de saúde. citando como exemplo um site de educação em saúde, que tem serviços interativos de perguntas e respostas para leigos.

3.2.1 Telemedicina no Brasil

No Brasil, a telemedicina ainda é uma atividade relativamente nova, de acordo com Sabbatini (2012), em outros lugares do mundo a telemedicina teve inicio com os primeiros voos espaciais, pois era necessário monitorar os sinais vitais dos astronautas, e para isso era preciso usar telemetria de rádio à longas distancias. O serviço de telemedicina pode ser visto no Brasil pela primeira vez em 1985, ainda de acordo com Sabbatini (2012), quando houve um acidente radioativo na cidade de Goiânia, onde varias pessoas foram contaminadas com césio, os médicos responsáveis pela realização dos laudos, eram professores da universidade universidade estadual de campinas(UNICAMP), e eles fizeram uso de um sistema baseado em computadores de 8 bits, que estavam conectados através da rede nacional de pacotes, os pontos de conexão eram as cidades de Brasília, Goiânia, Rio de Janeiro e Campinas, e através de correio eletrônico, os relatórios eram enviados e recebidos, e constavam neles, a evolução diária das vítimas do acidente.

Mediante a evolução da tecnologia, onde é possível fazer o uso de mais recursos do que um e-mail, de acordo com Sabbatini (2012) o Brasil tem um cenário favorável ao desenvolvimento da telemedicina, já que tem um amplo território, com áreas de difícil acesso, falta de transporte terrestre, e poucos recursos econômicos do governo local para investimento em saúde. Sabbatini (2012) ainda complementa, trazendo um dado levantado pelo Conselho Federal de Medicina, no qual 85% deles estão em apenas 100 cidades, e São Paulo se concentra, quase a metade da porcentagem dos médicos, e mais da metade dos municípios não possuem seus próprios médicos.

Uma forma conhecida de aumentar a cobertura e disponibilidade dos serviços de saúde, com qualidade, onde o acesso físico não é possível, é utilizando os serviços de telemedicina, e telessaúde. Sabbatini (2012)

A aplicação da telessaúde pode ocorrer por meio da avaliação por plataformas de videoconferências, prescrição de atividades através de software, monitoramento por uso de pedômetros, acelerômetros e prótese sensibilizada, reabilitação no uso de videoconferências, oxímetro, biofeedback, chamadas telefônicas, e-mails diários, e educação remota, funcionando através de videoconferências, sites, workshop e ferramenta de anotação digital.Costa et al. (2021)

As vantagens da telessaúde ainda de acordo com Costa et al. (2021)

> o potencial da otimização do tempo, custos dos serviços impostos, a possibilidade de aumentar o alcance da equipe de saúde em áreas distantes, ampliar o potencial de reabilitação e permitir diálogos educativos sobre a condição do paciente, o que impactara em uma estimulação de mudanças na vida do mesmo, tendo como objetivo de torná-lo mais saudável e independente.

3.3 APLICAÇÕES WEB

Aplicação WEB de acordo com Azevedo (2018) é qualquer software baseado em web que realize ações (funcionalidades) de acordo com uma interface com o usuário e que normalmente interaja com sistemas de backend. Quando um usuário interage com um web site para realizar alguma ação, por exemplo, fazer login, fazer compras ou acessar o banco, tem-se uma aplicação web. Gonçalves et al. (2005) apresenta mais definições para uma aplicação web, no qual pode ser de uma simples pagina, até um web site completo, tendo neste web site uma implementação de uma lógica de negócio, sendo o estado do negócio alterado conforme o uso, ainda é complementado que para ser uma aplacação web a mesma vai utilizar uma arquitetura distribuída, e irá funcionar em cima de protocolo http, tendo sua estrutura em hipertexto ou hipermídia.

3.4 APLICAÇÃO WEB DE TELEMEDICINA

No que se refere a telemedicina, e telessaúde existe órgãos como a sociedade brasileira de informática em saúde (SBIS) que determinam requisitos que um sistema de tele-consulta deve atender para que este tenha qualidade, esses requisitos, incluem funcionalidades, estrutura, conteúdo, segurança da informação, aderência a legislações, etc.

De acordo com o manual da SBIS na versão 5.1, existem as seguintes categorias que devem seguir os requisitos estabelecidos para certificação. Categoria PEP(Prontuario Eletrônico do paciente), Categoria Telessaúde e Categoria, Categoria prescrição eletrônica, e Categoria de segurança de informação, quando a aplicação WEB não se encaixa em nenhuma das outras categorias. Em cada categoria existem os requisitos para um determinado tipo de serviço médico. Na categoria de prontuario eletrônico de paciente (PEP) existem requisitos para sistemas de consultório individual, Clinicas e ou ambulatórios, Pronto atendimento e internação. Na Categoria telessaúde, existem requisitos para aplicação de tele-consulta, teleinterconsulta e requisitos para triagem, na categoria de prescrição eletrônica tem se os requisitos para receita digital, e na categoria de segurança da informação, tem-se o requisito de segurança da informação. De acordo com Silva (2020) Não existe a obrigação de todos sistemas de telemedicina serem certificados e cumprirem os requisitos da SBIS, nem o conselho federal de medicina, nem a SBIS fazem essa exigência para um sistema, pois se trata de um processo voluntário. Porém tendo uma certificação, tem-se um sistema de qualidade, no qual foi apresentada uma opinião técnica qualificada e imparcial, visando garantir privacidade e a confidencialidade das informações de saúde dos cidadãos, cumprindo a legislação brasileira sobre documentos eletrônicos e melhorando a qualidade dos sistemas de informação em saúde.

3.5 HEURÍSTICAS DE NIELSEN

As heurísticas de Nielsen, é uma forma de avaliar a usabilidade de um sistema, e a citada usabilidade é uma medida que diz o quão facilmente e eficientemente um ser humano consegue fazer uso das funções e dos recursos oferecidos por um produto de alta tecnologia Borges, Rapkiewicz e Feijó (2012), isso se faz necessário pois em qualquer área onde exista um produto, e esse venha ter interação humana, para que ele venha ter sucesso, uma boa experiencia do usuário é um ponto necessário Pressman (2021), quando um software é difícil de ser utilizado, e este estimula o usuário a cometer erros, ou dificulta seus esforços para atingir seus objetivos, quem o utiliza não irá gostar dele, independente da funcionalidade ou conteúdo oferecido, a experiencia do usuário deve ser correta, já que esta molda a percepção do software por parte do usuário Pressman (2021). E buscando uma boa experiencia do usuário, as

heurísticas apresentadas por Nielsen, permitem a avaliação dessa experiencia, dentre essas heurística, destacam-se:

- **Feedback:** Informar continuamente o usuário sobre o que ele está fazendo.

- **Falar a linguagem do usuário:** Terminologia adequada ao usuário e não ao sistema. Buscar organizar as informações de acordo com o modelo mental do usuário.

- **Saídas demarcadas:** Possibilidade de uma tarefa ser cancelada ou uma operação ser desfeita voltando ao estado anterior. Ou seja, o usuário controla o sistema.

- **Consistência:** Fazer a mesma coisa sempre do mesmo jeito, isto é, a mesma operação deve ser apresentada sempre da mesma maneira para facilitar o reconhecimento. O efeito de uma ação deve ser sempre o mesmo.

- **Prevenir erros:**Ter clara as situações que mais provocam erros e modificar a interface para que tais erros não ocorram.

- **Independência da memória do usuário:** A interface deve propiciar que o usuário faça suas escolhas sem necessidade de lembrar comandos específicos.

- **Atalhos:**Fornecer atalhos para acesso a informações com muita profundidade na árvore de navegação. Fornecer abreviações, teclas de função, etc, para usuários experientes poderem executar as funções de modo mais rápido.

- **Dialogo simples e naturais:** A informação apresentada ao usuário deve ser adequada ao momento e contexto, sem excesso de informação nem falta. O acesso às operações deve ser compatível com o modo como o usuário interage.

- **Boas mensagens de erros:** Mensagens de erro que ajudem a resolver problemas, não punindo ou intimidando o usuário, usando linguagem clara e sem códigos.

- **Ajuda e documentação:** O sistema deveria ser de tal forma intuitivo que a necessidade de ajuda ou documentação seria dispensável. Mas sendo necessária deve estar facilmente acessível.

3.6 PRODUTO MINÍMO VÍAVEL - MINIMUM VIABLE PRODUCT

Visando reduzir as dificuldades iniciais e melhorar significativamente as estimativas dos recursos iniciais da criação de um projeto, objetivando uma demonstração ágil de um produto mínimo com valor agregado para o cliente final, é apresentado o conceito do Mínimo Produto Viável (MVP), quando o objetivo é criar um produto de qualidade, de forma ágil e com menor esforço. A ideia do MVP foi originalmente vinculada aos conceitos obtidos por meio da filosofia originada no modelo Toyota de Produção Enxuta Santos, Tiosso e Petrucelli (2019).

Um empreendedor do Vale do Silício que, a partir do estilo Toyota formou uma metodologia baseada no desenvolvimento do cliente, conhecida como Lean Startup, tendo como fim atingir a maior qualidade possível em um produto com o menor esforço possível no desenvolvimento. Estes princípios resultaram no surgimento do movimento Lean Startup, também conhecido no brasil como Startup Enxuta, que se popularizou entre os empreendedores brasileiros.

O MVP não precisa ser um protótipo completo contendo diversas funcionalidades para se tornar um grande produto de sucesso. Assim, um vídeo ou até mesmo uma imagem ilustrativa com o objetivo de testar hipóteses que possam ser respondidas o mais rápido possível por um público alvo, podendo reduzir tempo e esforços para o desenvolvimento do produto final, podem ser considerados um MVP Santos, Tiosso e Petrucelli (2019).

4 METODOLOGIA

Para o desenvolvimento do MVP proposto neste projeto, os seguintes passos foram seguidos:

- Analise de aplicativos de telemedicina já existentes
- Levantamento de requisitos
- Desenvolvimento de casos de uso
- Desenvolvimento do diagrama entidade relacional.
- Implementação do MVP
- Integração do MVP com a plataforma de comunicação da twilio.

4.1 ANALISE DE APLICATIVOS DE TELEMEDICINA JÁ EXISTENTES

4.1.1 HiDoctor

É um software médico desenvolvido pela centralx que é uma empresa focada na produção de softwares para ambientes médicos de alta produtividade. O software hiDoctor objetiva a gestão de clinicas e consultórios, e conta com as funcionalidades de

- **Prontuario:** Ficha de dados pessoais, anamnese, receitas e textos impressos facilmente acessíveis. O programa acompanha também CID-10 e bulário.
- **Agenda:** Agendamento completo, com marcação prática de consultas, confirmação, envio de lembretes, impressão da agenda e balanço, agendamento online através do CatalogoMed, com importação direto para o prontuário e muito mais.

- **Prescrição Médica:**De acordo com o site do software esta funcionalidade permite prescrever uma receita em poucos cliques. O HiDoctor® inclui a EPF, uma enciclopédia com milhares de produtos farmacêuticos para facilitar o preenchimento das prescrições.

- **serviços de SMS para confirmação de consultas:** Nesta funcionalidade as consultas do dia serão confirmadas enviando mensagens de texto para os pacientes agendados. Os pacientes podem confirmar se irão ou não comparecer através de um link enviado na mensagem e o remetente recebe as respostas das confirmações diretamente na agenda.

- **Criação de formulários personalizados para uma determinada especialidade:** Funcionalidade que permite a criar formulários personalizados para facilitar as consultas ou consultar o acervo de formulários feitos por outros médicos de uma determinada especialidade.

- **Atlas de anatomia do corpo humano:**Ilustrações em alta resolução de órgãos e sistemas do corpo humano para importar para o prontuário ou utilizar em sites ou publicações acadêmicas.

- **Chat de comunicação interna:** Ferramenta para comunicação dentro do consultório: para enviar mensagens instantâneas para secretárias e outros médicos da rede local.

- **Teleconsulta:** Ferramenta médica profissional para realizar atendimentos via teleconsulta. O HiDoctor® cria a sala de atendimento virtual e facilita o envio do convite para o paciente.

4.1.2 My Smart Clinic

My Smart Clinic também é uma empresa de tecnologia especializa em software de gestão para clínicas médicas. Conta com uma versão de avaliação gratuita de 7 dias, onde é possível ter acesso a funcionalidades de

- **Agenda:** Onde é possível verificar os pacientes agendados para um determinado médico, sendo possível alterar o status da consulta de agendado para confirmado, paciente no local, desmarcado, ou falta. também é possível incluir um paciente na agenda, selecionar o horário e escolher o procedimento que o mesmo irá realizar.

- **Prontuário:** Na funcionalidade de prontuario é possível ter acesso aos pacientes cadastrados no sistema, um prontuario está associado a um paciente, e para inclusão de um prontuario a um paciente, é necessário clicar no paciente, onde terá acesso a uma lista de fichas, recursos e fichas personalizadas, no qual a tela pode ser vista na figura 11. Os prontuários associados ao paciente, ficam em um lista de prontuários, logo abaixo das fichas. Também pode ser incluso ao prontuário do paciente, fotos, vídeos, simulações, documentos, orçamentos, contratos, galaria e ter acesso ao histórico de atendimento.

- **Tela de consulta** Funcionalidade destinada ao registro de dados de uma consulta com um paciente, esta funcionalidade, é acessada através do prontuário do paciente, os campos desta funcionalidade, embora não mostrados na imagem, são todos do tipo textarea do tipo documento editável, os campos são queixa principal, HPMA, Antecedentes pessoais, medicações em uso, um campo do tipo checkbox para indicar se a pessoa faz uso de tabagismo, outros hábitos, antecedentes familiares, exame físico, hipótese diagnostica e conduta.

- **Inclusão de paciente:** Funcionalidade destinada a cadastro de pacientes no sistema.

- **Consulta online** apesar de se tratar de um software com foco em clinica e atendimento presencial, o My Smart Clinic possui uma funcionalidade de consulta online, porém não é de fácil acesso. As imagens a seguir mostram o procedimento para uma consulta online. **Depois de fazer o cadastro de procedimento de consulta online** Depois de fazer o cadastro de procedimento de consulta online é necessário fazer o agendamento, e em Tipo compromisso selecionar a opção Consulta Online ou Retorno Online e salvar. Então o sistema gerará um link para o acesso à sala virtual.

Além de contar também com as funcionalidades de Videoteca com materiais informativos, serviço financeiro, troca de informação de saúde suplementar, estoque de produtos, relatórios e referencias.

4.1.3 Amplimed

O software da amplimed conta com muitas funcionalidades como o hiDoctor, porém com uma interface bem amigável, como o My Smart Clinic. A analise das funcionalidade será feita com base no que será implementado para este projeto. Na analise dessas soluções em telemedicina, destaquei vantagens e desvantagens, para a construção do mvp da aplicação web desta pesquisa.

	Vantagens	Desvantagens	Funcionalidades
HiDoctor	Completo em funcionalidades	Necessita de download	Prontuário
		Pago e não possui periodo de testes	Agenda
			Prescrição Médica
			serviços de SMS para confirmação de consultas:
			Criação de formulários personalizados para uma determinada
			Atlas de anatomia do corpo humano
			Chat de comunicação interna
			Teleconsulta
My Smart Clinic	Interface amigável	Funcionalidade de consulta online de difícil acesso.	Agenda
			Prontuário
			Tela de consulta(Basica)
			Gerênciamento de cadastro de pacientes
		Pago após o período de 7 dias de teste.	Consulta online(Teleconsulta)
Amplimed	Completo em funcionalidades		Gerenciamento de cadastros de pacientes
			Agendamento de consultas
			Consulta
			Prontuário
	Interface amigável	Pago após 10 dias de teste.	Teleconsulta

Figura 1 – Comparativos das aplicações já existentes no mercado

4.2 VISÃO GERAL DA APLICAÇÃO

Partindo da definição apresentada por Azevedo (2018) no qual é dito que a uma aplicação WEB é qualquer software baseado em web que realize ações (funcionalidades) de acordo com uma entrada de usuário e que normalmente interaja com sistemas de backend, e também da definição apresentada por Maldonado, Marques e Cruz (2016) de que a telemedicina é o uso de tecnologias da informação e comunicação como auxilio as atividades que envolvam saúde, algumas funcionalidades básicas para a implementação de uma aplicação web que visa levar atendimento medico aos pacientes serão descritas.

4.3 TECNOLOGIAS E FERRAMENTAS

A aplicação será construída usando tecnologias como HTML, CSS, Javascript, PHP utilizando o framework Laravel, o pacote spartie, MySQL, e a Plataforma de comunicação da Twilio, a aplicação deverá ser capaz de realizar o cadastro de médicos, pacientes e assistentes de saúde(funcionários da unidade básica de saúde), estes após cadastrados, usarão suas credenciais para fazer acesso ao sistema web, os pacientes poderão acessar a funcionalidade consulta, e através de uma videochamada, farão uma triagem com um enfermeiro, e depois irá aguardar atendimento, o médico inicialmente tera acesso aos pacientes que estão na aplicação aguardando atendimento, irá selecionar um, e chama-lo para atendimento, na ficha da consulta irá os dados da triagem feitas pelo enfermeiro na triagem, no decorrer do atendimento, novas informações poderão ser adicionadas ao histórico médico do paciente, os assistentes de saúde, inicialmente irão prestar assistência ao médico, caso seja necessário levar algum medicamento, ou prestar algum atendimento posterior a consulta do paciente. Com base nisto, em termos de aplicação, já é possível ver o uso de tecnologias de informação e comunicação, para auxilio na prestação de serviços médicos.

4.3.1 Plataforma de prescrições do CFM

Esta aplicação tem um link para plataforma de prescrições do Conselho Federal de Medicina(CFM), uma plataforma gratuita desenvolvida pelo Conselho Federal de Medicina (CFM) em parceria com Conselho Federal de Farmácia (CFF) e o Instituto Nacional de Tecnologia da Informação (ITI) que vale para o todo território nacional. Com ela, a rotina do médico fica mais fácil, já que pode receitar de forma mais ágil e eficiente, nela é possível emitir atestado médico, relatório médico, receita simples, receita antimicrobianos, receita de controle especial, solicitação de exames, laudos e parecer técnico. Através dessa plataforma os pacientes recebem esses documentos médicos por e-mail ou whatsapp. No que se refere a segurança, para validar o médico profissional que fará uso dessa plataforma o procedimento é de acordo com a Portaria nº 467, de 20 de março de 2020, do Ministério da Saúde, nela, o profissional deve usar assinatura eletrônica, por meio de certificados e chaves emitidos pela Infraestrutura de Chaves Públicas Brasileira (ICP-Brasil). Além disso, o uso de dados associados à assinatura do médico deve ser feito de tal modo que qualquer modificação posterior

possa ser detectável, além de observar os requisitos previstos em atos da Agência de Vigilância Sanitária (Anvisa). os médicos dessa aplicação, em uma consulta, irão clicar no botão prescrever, e serão direcionados para plataforma de prescrições do CFM.

4.4 LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos é a etapa do desenvolvimento de sistemas de informação responsável por identificar e modelar as necessidades do negócio a serem atendidas pelos sistemas de informação, e é, portanto, uma atividade cada vez mais relevante em um cenário dinâmico. A engenharia de software se produz através de um conjunto de fases. Cada uma das fases pode envolver métodos,ferramentas e procedimentos, cujas formas de estruturação são citadas como modelo de engenharia de software, independentemente do modelo de desenvolvimento de software, o processo contém três fases genéricas: definição, desenvolvimento e manutenção Junior e Campos (2008).

Segundo Rosa et al. (2017) a engenharia de Requisitos localiza-se na fase de definição, sendo responsável por entender o funcionamento do software, como será a experiência do usuário final e como o sistema irá influenciar nos negócios do cliente. O entendimento correto dos requisitos consiste na etapa mais crítica do desenvolvimento de um software, possuindo influência direta na qualidade final do produto. De acordo com Silva (2020) A fase de levantamento de requisitos deve identificar dois tipos de requisitos: os funcionais e os não-funcionais. Os requisitos funcionais correspondem ao que o cliente quer que o sistema realize, ou seja, as funcionalidades do software. Requisitos funcionais são declarações dos serviços que o sistema deve fornecer. Os requisitos não funcionais são restrições aos serviços ou funções fornecidos pelo sistema. Eles incluem restrições de tempo, restrições no processo de desenvolvimento e restrições impostas por padrões. Os requisitos não funcionais geralmente se aplicam a todo o sistema, não a recursos ou serviços individuais correspondem às restrições, condições, consistência e verificação que devem ser realizadas nos requisitos funcionais.

4.4.1 Requisitos Funcionais

RF01 – CADASTRO DE USUÁRIO E SENHA: representa a funcionalidade de cadastro de usuário para acessar o sistema, podendo ser administrador, paciente, enfermeiro, medico e agente de saúde. As senhas cadastradas serão armazenadas no banco de dados, codificado por algoritmo hash. Os usuários poderão trocar a senha sempre que necessário.

RF02 – LOGIN: corresponde ao acesso dos usuários a ferramenta, contendo os campos de login e senha.

RF03 - CADASTRO DE AGENTE DE SAÚDE: corresponde a funcionalidade de criação e manutenção dos registros dos agentes de saúde.

RF04 - CADASTRO DE ENFERMEIROS: Corresponde a funcionalidade de cadastro e manutenção de registros de enfermeiros.

RF05 – CADASTRO DE UNIDADE BÁSICA DE SAÚDE: representa a funcionalidade de realizar o cadastro da unidade básica de saúde, vinculando cada unidade de saúde a um médico, um medico poderá atuar em mais de uma unidade de saúde.

RF06 - CADASTRO DE AGENDAS: corresponde a funcionalidade de criar um cadastro de agendas para dispor uma organização de horários para atendimento a pacientes. Esta funcionalidade irá trazer a possibilidade de criar agendas por médico tendo horário de inicio e fim da consulta.

RF07 – AGENDAMENTO DE CONSULTAS: corresponde a funcionalidade de realizar agendamentos no sistema, com base na agenda de cada medico, o paciente escolhe os horários disponíveis na agenda do medico, e é incluso em uma fila de atendimento.

RF08 – CADASTRO DE PACIENTES: corresponde a funcionalidade de cadastro e manutenção dos registros do paciente.

RF09 – ATENDIMENTO MÉDICO: corresponde a funcionalidade de atender um paciente, com consulta agendada.

RF10 – CONSULTA: representa a funcionalidade de criar uma videoconferência com o paciente, e gravar dados clínicos de anamnese, conclusão diagnostica, e acesso a prescrição de receitas na plataforma do CFM.

RF11 - TRIAGEM: representa a funcionalidade de uma pré-consulta, realizada por um enfermeiro em videoconferência.

RF12 – IMAGENS E ANEXOS: corresponde a funcionalidade de incluir junto ao atendimento médico um anexo de uma imagem ou documento para complementar o atendimento médico, sendo possível pré-visualizar aos documentos internamente no sistema. Sistema permitirá realizar o download das imagens já anexadas, sendo possível salvar localmente no computador.

4.5 CASOS DE USO

De acordo com Pressman (2021) Um caso de uso conta uma jornada estilizada sobre como um usuário desempenhando um, de uma série de papéis possíveis interage com o sistema sob um conjunto de circunstâncias específicas. A jornada poderia ser um texto narrativo, uma descrição geral das tarefas ou interações, uma descrição baseada em modelos ou uma representação esquemática, em resumo, um caso de uso representa o software ou o sistema do ponto de vista do usuário. E com base nisso fiz os seguintes casos de uso:

1. Cadastrar unidade básica de saúde.
2. Cadastrar usuário e senha
3. Login
4. Cadastrar funcionário
5. Cadastrar agenda
6. Agendar consulta
7. Triagem

8. Consultar
9. Anexar arquivos

Onde faço destaque do caso de uso principal, que é o caso de uso de consultar, e visando facilitar a compreensão do sistema, foi feito o diagrama de caso de uso da aplicação. O diagrama de caso de uso pode ser visto no apêndice A, e sua descrição, pode ser vista no apêndice B.

Caso de uso: Consultar.

Atores: Médico, Paciente

Objetivo: Realizar uma consulta.

Pré condições: Paciente e médicos, precisam estar cadastrados, logados e o sistema de cada um deve ter permissões para acessar câmera e microfone.

- Paciente acessa a opção "consulta"
- Paciente seleciona a opção "minhas consultas"
- O paciente clica na opção cosultar.
- Ao clicar na opção de consultar, o primeiro contato será com o enfermeiro que fará a triagem.
- Depois de passar pela triagem o médico atende o paciente e a consulta começa.
- Médico ou paciente podem finalizar a consulta.

Neste caso de uso, descrevi, a partir do ponto de vista do usuário como é feito o processo de uma consulta.

4.6 DIAGRAMA ENTIDADE RELACIONAL

Após fazer o levantamento de requisitos, e a descrição dos casos de uso com base nos requisitos, foi realizado a modelagem entidade relacional. a modelagem foi feita através do software MySQL WorkBench, e foi necessária para representar a maneira como as entidades/classes colaboraram para atender aos requisitos da aplicação. Essa modelagem resultou em doze entidades.

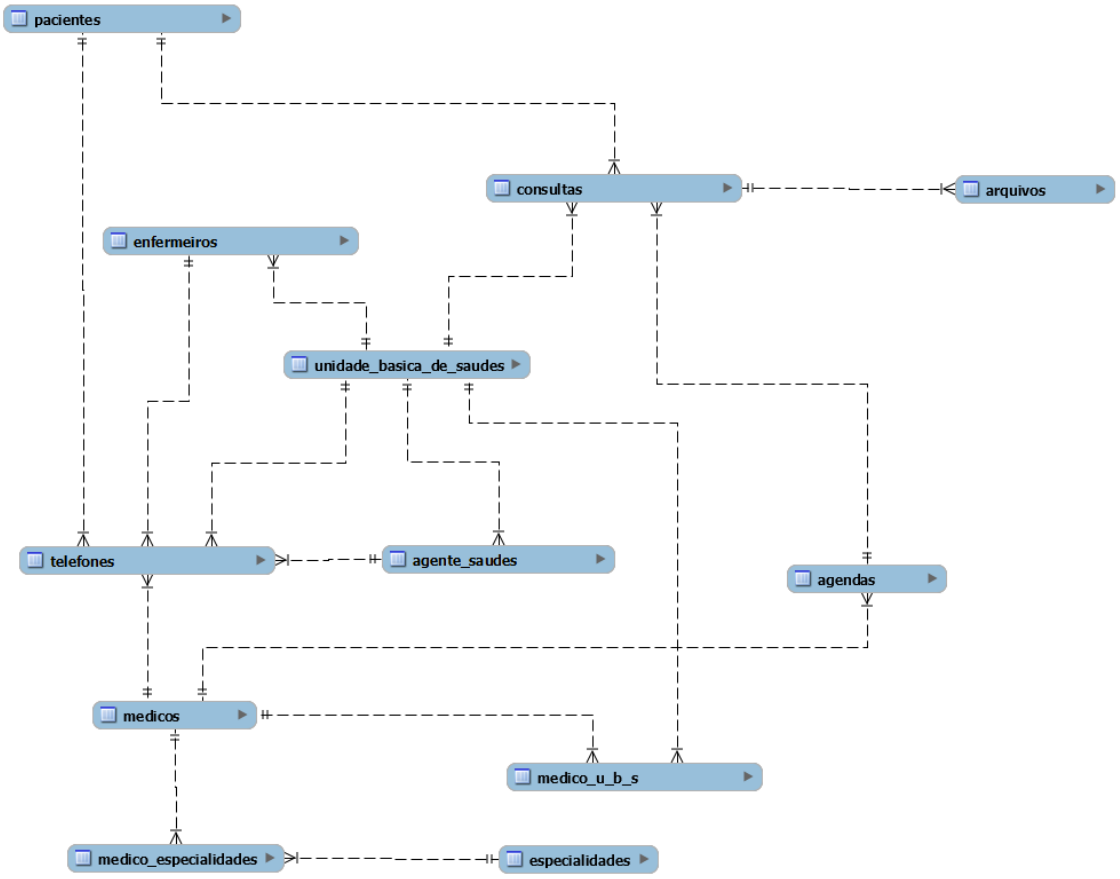


Figura 2 – Diagrama entidade relacional

Onde é possível ver apenas a estrutura de relacionamentos, sendo que o diagrama completo, com atributos estão no apêndice D. Nesse projeto, decidi usar apenas dois diagramas, o de caso de uso junto com sua descrição, e o diagrama entidade relacional, antes de começar a implementação em código.

4.7 IMPLEMENTAÇÃO DO MVP DA APLICAÇÃO WEB DE TELEMEDICINA

Para implementação do MVP desta aplicação foi utilizado o framework Laravel, uma estrutura de aplicação própria para web, que utiliza a arquitetura MVC(Model,View,Controller), e oferece uma forma de criar tabelas, modelos, controladores, entre outros recursos de forma muito facilitada através do comando php artisan, uma vez através do diagrama entidade relacional tendo a estrutura da aplicação, criei os arquivos de modelo, controle, e banco de dados de cada classe através do comando php artisan make:model <Entidade> -mcr, onde as flags mcr, informam ao framework, que ele deve criar uma migração para criar a tabela correspondente a entidade no banco de dados, o contro-

lador, e a assinatura de alguns métodos padrões, que permitem gerenciar, a criação, atualização, leitura, e exclusão de cada entidade da aplicação.

Faço destaque a parte de consultas no qual o código referente a esta sessão se encontra no apêndice F, o controlador da classe consulta ConsultaControler, de acordo com o padrão MVC, retorna uma view/pagina de consulta, essa mesma view é exibida de forma diferente dependendo do perfil do usuário logado, isso é possível graças ao pacote spartie, no qual através da anotação @role é possível definir o papel que tem permissão para acessar a funcionalidade, o papel, é associado a cada usuário, no momento do cadastro, seja paciente, enfermeiro, médico, ou agente de saúde. Outro ponto importante da implementação desta funcionalidade, é o botão de prescrever, ao ser clicado, irá direcionar o médico para a plataforma de prescrições do CFM, a tela desta funcionalidade poderá ser vista no apêndice C.

4.7.1 Integração com a plataforma de comunicação Twilio

Após fazer a implementação do métodos de gerenciamento de cadastro de cada entidade, para que uma consulta e uma triagem com videoconferência fosse disponibilizada na aplicação, foi feita a integração com a plataforma twilio, no qual disponibiliza varias soluções para comunicação, como voz, sms, e-mail, e vídeos, para esta aplicação, estou utilizando o twilio video, que é uma solução que pode ser usada para criar aplicativos de vídeo em tempo real, e gratuitos com o WebRTC Go, a plataforma fornece API REST e SDKs para JavaScript, Android e iOS, para o estado atual deste MVP o foco é o javascript.

Foi necessário entender como funciona uma sala de video na plataforma twilio. O Twilio Programmable Video foi criado no WebRTC, que é uma API elaborada pela World Wide Web Consortion, com essa API é possível adicionar recursos de comunicação em tempo real ao aplicativo.

Ele permite que vídeo, voz e dados genéricos sejam enviados entre pares, permitindo o desenvolvimento de soluções avançadas de comunicação por voz e vídeo. A tecnologia está disponível em todos os navegadores modernos, bem como em clientes nativos das principais plataformas. As tecnologias por trás do WebRTC são implementadas

como um padrão da Web aberto e estão disponíveis como APIs JavaScript normais em todos os principais navegadores. No caso de clientes nativos, como apps Android e iOS, há uma biblioteca disponível com a mesma funcionalidade. O projeto WebRTC é de código aberto e compatível com Apple, Google, Microsoft e Mozilla, entre outros.

Para implementar o twilio video nesta aplicação, foi necessário criar uma conta na plataforma, ter uma aplicação backend, que através de credenciais disponibilizadas pela twilio, através de uma requisição http, gere uma chave que posteriormente será utilizada para acessar uma sala de video, a aplicação backend, foi desenvolvida, e é a mesma que utilizei para gerenciar os cadastros das entidades de modelo da aplicação, para atender a demanda de se conectar com as credenciais da plataforma twilio, e gerar uma chave de acesso a uma sala, tive apenas que implementar uma rota para fazer a requisição, e implementar um controlador, para receber a requisição e devolver uma chave. uma representação deste procedimento poderá ser visto na figura 3.

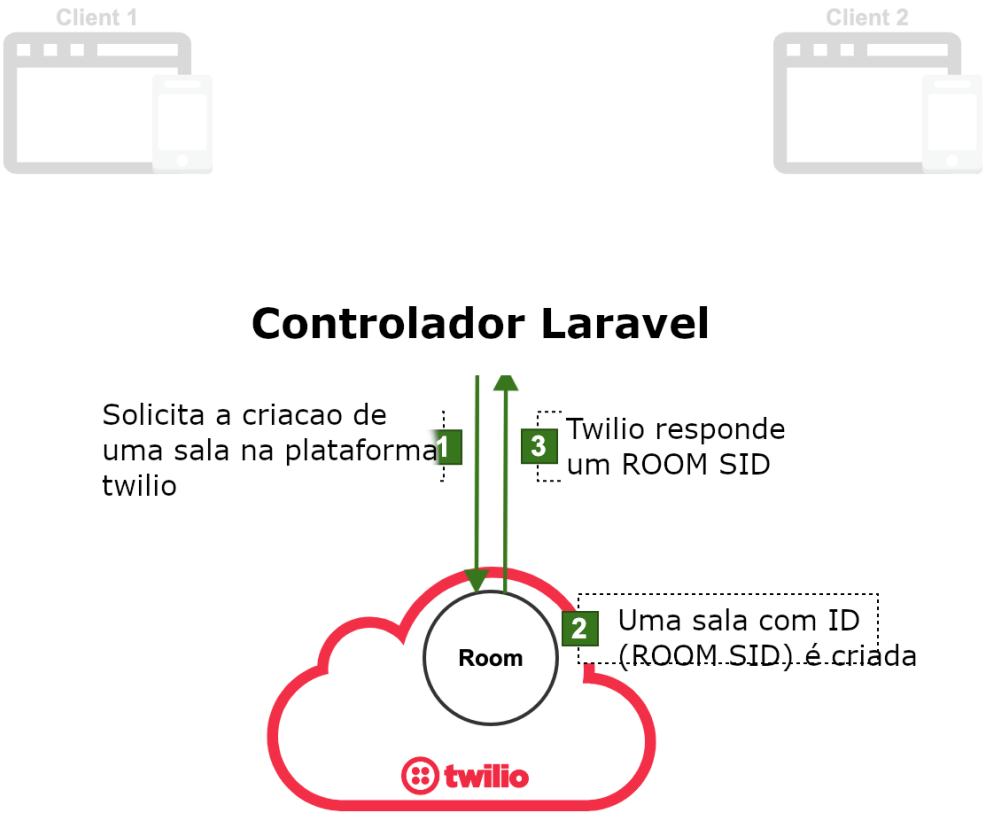


Figura 3 – Criação de uma sala de video na plataforma Twilio

1. Um controlador laravel fará uma requisição ao Twilio para criar uma Sala usando a API REST de Salas.

2. O Twilio irá validar as credenciais de API fornecidas e então criar a sala. no qual terá o estado acompanhada até sua conclusão.

3. O Twilio devolve as informações da sala. Isso inclui o SID: um identificador exclusivo que pode ser usado em solicitações de API posteriores para se referir a sala. Neste ponto, a sala está criada, porém ainda está vazia.

Primeiro cliente obtém um token de acesso Para se conectar a sala criada, é necessário obter um Token de Acesso, isso irá garantir um melhor controle de quem está autorizado a ingressar na sala.

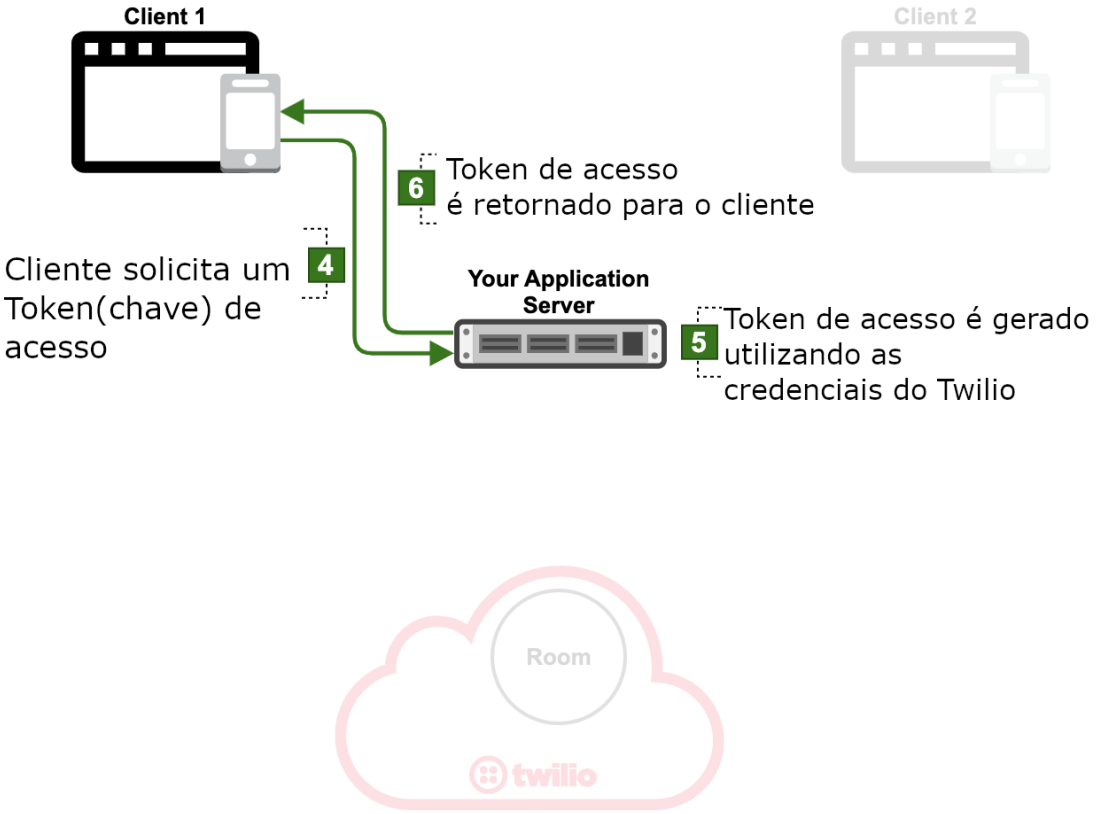


Figura 4 – Primeiro cliente obtém um token de acesso

4. O Cliente 1 solicita um Token de Acesso do Servidor de Aplicações. através de uma

solicitação HTTP.

5. O Servidor de Aplicações, neste caso um controlador laravel utiliza as credenciais da minha conta na Twilio para gerar um Token de Acesso criptograficamente seguro utilizando as bibliotecas auxiliares da Twilio.

6. O Token de Acesso é devolvido ao Cliente 1.

Cliente 1 se conecta a sala

Figura 5 – Cliente 1 se conecta a sala

7. O Cliente 1 se conecta à sala utilizando a interface connect do Twilio Video SDK e autentica-se utilizando o Token de Acesso que foi retornado.

8. O Twilio verifica o Token de Acesso. sendo válido, uma conexão de sinalização é estabelecida entre o cliente e a sala. Neste ponto, o cliente se torna um Participante

e pode publicar e/ou assinar faixas de mídia de outros Participantes.

Cliente 2 se conecta a sala para o segundo cliente se conectar a sala, o processo será idêntico ao que o primeiro cliente usou para se conectar a sala.

1. O cliente 2 irá fazer uma requisição HTTP para o controlador laravel, que utilizando as credenciais do twilio, irá retornar um token de acesso.

2. O controlador laravel, irá gerar e retornar esse token para o cliente2.

3. O cliente2 irá usar esse token, na interface conect do Twilio Video SDK.

4. O Twilio irá verificar o token, e então conectar o cliente2 a sala.

5. Estando na sala o cliente2 se torna um participante, que pode ser chamado de participante2, e então pode publicar e/ou assinar faixas de mídia de outros Participantes.

Faixas de mídia, áudio e video

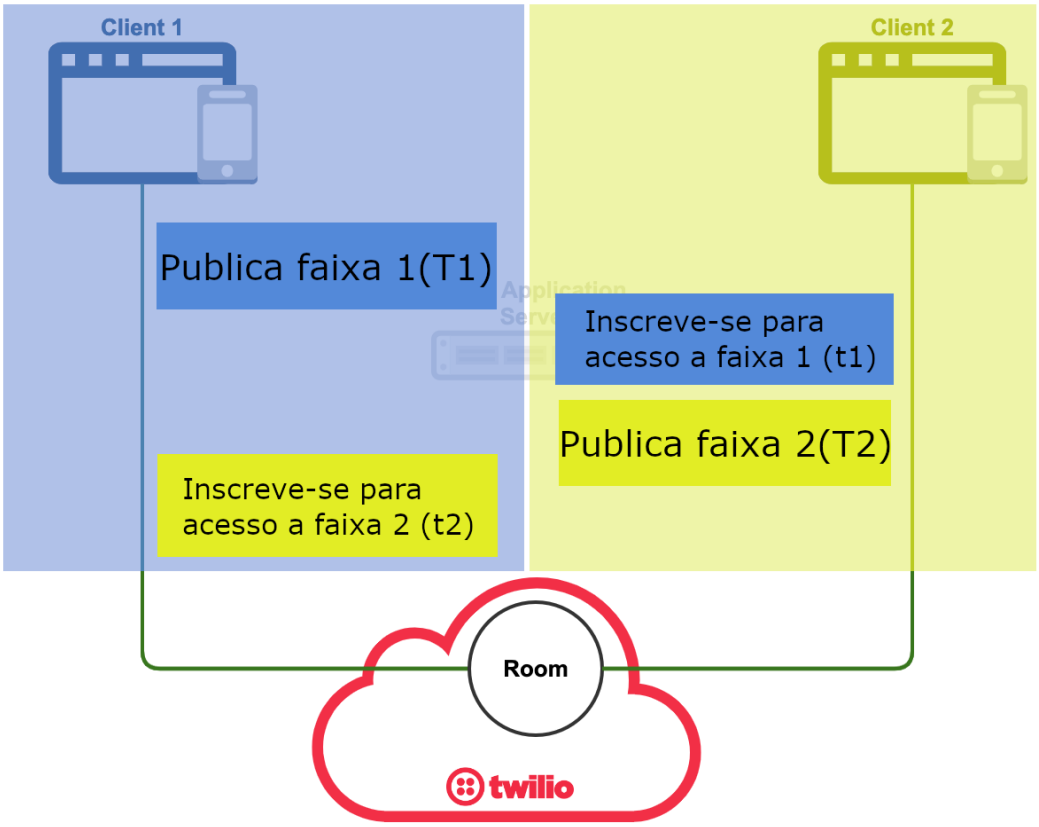


Figura 6 – Faixas de mídia, áudio e video

1. Um Participante da sala pode publicar faixas de áudio, vídeo e dados na sala. Nesse caso, o Cliente 1 publica uma Trilha chamada T1.

2. Os demais participantes da sala são notificados desta publicação da faixa e podem se inscrever na faixa de T1. Neste caso, o Cliente 2 se inscreve na faixa de dados de T1 e tem acesso a ela.

3. Participantes adicionais podem publicar ou assinar trilhas/faixas. Nesse caso, o Cliente 2 publica uma Trilha chamada T2, que é então assinada pelo Cliente 1.

são essas faixas de dados que permitem que no contexto desta aplicação, o paciente veja video e áudio do médico, e que o médico veja video e áudio do paciente. abaixo é disponibilizado o código do controlado responsável por solicitar a plataforma twilio a criação de uma sala, e gerar o token para que os participantes se conectem a ela.

```php

namespace App\Http\Controllers\API;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Twilio\Jwt\AccessToken;
use Twilio\Jwt\Grants\VideoGrant;


class AccessTokenController extends Controller{
    public function generate_token($nomeSala){

        $accountSid = env('TWILIO_ACCOUNT_SID');
        $apiKeySid = env('TWILIO_API_KEY_SID');
        $apiKeySecret = env('TWILIO_API_KEY_SECRET');

        $sala = $nomeSala;
        $identity = uniqid();

        // Cria o Token de acesso
        $token = new AccessToken(
            $accountSid,
            $apiKeySid,
            $apiKeySecret,
            3600,
            $identity
        );

        // Permissao para acessar o video
        $grant = new VideoGrant($sala);
        $grant->setRoom("$sala");
        $token->addGrant($grant);
```

```
33
34          // Serializa o token em JWT
35          return response()->json([
36              'token' => $token->toJWT()
37          ]);
38
39
40      }
41  }
```

Listing 4.1 – Controler gerador de tokens

Abaixo é disponibilizado o código em javascript responsável por fazer a requisição do token ao controler.

```
1       const Video = Twilio.Video;
2   const sala = document.getElementById('camera').dataset.sala;
3   const url = "/api/access_token/" + sala;
4   const camera = document.getElementById('camera');
5   const btnPrescricao = document.getElementById('prescrever');
6
7   if(btnPrescricao != null){
8     const windowFeatures = "left=1000,top=100,width=820,height
       =720";
9     btnPrescricao.addEventListener('click', function(e) {
10      e.preventDefault();
11        window.open('https://prescricao.cfm.org.br/login', '
       _blank', windowFeatures);
12    });
13  }
14
15  fetch(url)
16    .then(response => response.json())
17    .then(data => {
```

```
18      Video.connect(data.token, {
19        name: `"${sala}"`,
20        audio: true,
21        video: { width: 640 }
22      }).then(room => {
23        room.participants.forEach(participantConnected);
24        room.on('participantConnected', participantConnected);
25
26        room.on('participantDisconnected',
     participantDisconnected);
27        room.once('disconnected', error => room.participants.
     forEach(participantDisconnected));
28      });
29    });
30
31  function participantConnected(participant) {
32    console.log('Participant "%s" connected', participant.
     identity);
33
34    const div = document.createElement('div');
35    div.id = participant.sid;
36
37    const trackSubscribed = (track) => {
38      div.appendChild(track.attach());
39    };
40
41    participant.on('trackSubscribed', trackSubscribed);
42
43    div.innerText = participant.identity;
44
45    participant.tracks.forEach(publication => {
46      if(publication.isSubscribed) {
47        trackSubscribed(publication.track);
```

```
48      }
49    });
50
51    camera.appendChild(div);
52
53    participant.on('trackUnsubscribed', trackUnsubscribed);
54
55  }
56
57  function participantDisconnected(participant) {
58    console.log('Participant "%s" disconnected', participant.
       identity);
59
60    participant.tracks.forEach(trackUnsubscribed);
61    document.getElementById(participant.sid).remove();
62  }
63
64  function trackUnsubscribed(track) {
65    track.detach().forEach(element => element.remove());
66  }
```

Listing 4.2 – trecho responsavel por requisitar o token e criar a sala ao controlador

Os demais trechos de código incluindo a estrutura da aplicação poderão ser encontrados no apêndice E e apêndice F.

4.8 ANÁLISE GERAL DO TRABALHO

Como resultado dessa pesquisa foi possível chegar a uma aplicação MVP, em que é possível conectar em uma sala de videoconferência, primeiramente, enfermeiro e paciente em uma triagem, que é enviada para ficha de consulta do paciente que irá se consultar, essa ficha é acessada pelo médico já com os dados da triagem, e depois médico e paciente, em uma tele-consulta propriamente dita. No qual nessa consulta é possível cadastrar dados clínicos, como anamnese, exame físico, lista de problemas, CID, hábitos de vida e conclusão diagnostica. E após o relato desses dados percebidos pelo medico na consulta com o paciente, o mesmo consegue ser direcionado para a plataforma do CFM para fazer uma prescrição. No Apêndice B, é possível ver as telas da aplicação.

4.9 TRABALHOS FUTUROS

Para trabalhos futuros, é visado algumas melhorias de funcionalidades, como por exemplo, a gravação de dados da triagem e consulta ser feito de forma assíncrona sem precisar recarregar a pagina, pois atualmente ao recarregar a pagina da triagem e da consulta, a câmera do paciente sai da tela do médico, o incremento da aplicação com uma ferramenta de chatbot, também é algo visado, atualmente a triagem é feita por um enfermeiro, então será feita uma pesquisa para incluir uma funcionalidade de chatbot na aplicação, para auxiliar o enfermeiro na triagem.

REFERÊNCIAS

AZEVEDO, D. G. d. *Um estudo sobre ferramentas de busca de vulnerabilidades em aplicações web*. Dissertação (B.S. thesis) — Universidade Federal do Rio Grande do Norte, 2018.

BARBOSA, P. H. F. d. A.; PEREIRA, T. V.; MARTINS, E. F. Telemedicina. EDUERN, 2019.

BORGES, K.; RAPKIEWICZ, C.; FEIJó, A. Usando heurísticas de nielsen para avaliar objetos de aprendizagem e softwares educacionais: um estudo exploratório na área de matemática para ensino superior. In: *Anais do XVIII Workshop de Informática na Escola*. Porto Alegre, RS, Brasil: SBC, 2012. p. 431–440. ISSN 0000-0000. Disponível em: <https://sol.sbc.org.br/index.php/wie/article/view/18736>.

COSTA, T. da et al. Oportunidade para telessaúde em tempos de pandemia: uma revisão integrativa. *Brazilian Journal of Development*, v. 7, n. 11, p. 106419–106432, 2021.

GOMES, M. A. V.; PINTO, V. de O.; CASSUCE, F. C. da C. Determinantes da satisfacao no atendimento das unidades basicas de saude. *Ciência saude coletiva*, Associacao Brasileira de Pos-Graduacao em Saude Coletiva - ABRASCO, v. 26, n. 4, p. 1311, 2021. ISSN 1413-8123.

GONÇALVES, R. F. et al. Uma proposta de processo de produção de aplicações web. *Production*, SciELO Brasil, v. 15, p. 376–389, 2005.

JUNIOR, D. P. d. A.; CAMPOS, R. d. Definição de requisitos de software baseada numa arquitetura de modelagem de negócios. *Production*, SciELO Brasil, v. 18, p. 26–46, 2008.

MALDONADO, J. M. S. d. V.; MARQUES, A. B.; CRUZ, A. Telemedicina: desafios à sua difusão no brasil. *Cadernos de Saúde Pública*, SciELO Brasil, v. 32, 2016.

MARCOLINO, M. S. et al. A rede de teleassistência de minas gerais e suas contribuições para atingir os princípios de universalidade, equidade e integralidade do sus-relato de experiência. *Revista Eletrônica de Comunicação, Informação e Inovação em Saúde*, v. 7, n. 2, 2013.

PRESSMAN, R. S. *Engenharia de software - 9.ed.* McGraw Hill Brasil, 2021. Disponível em: <https://books.google.com/books/about/Engenharia_de_software_9_ed.html?id=FSE3EAAAQBAJ>.

ROSA, L. H. C. et al. Jogos para ensino de levantamento de requisitos de software: uma revisão sistemática de literatura. *RENOTE*, v. 15, n. 2, 2017.

SABBATINI, R. M. A telemedicina no brasil: evolução e perspectivas. *Informatica em Saude: Uma Perspectiva Multiprofissional dos Usos e Possibilidades. Sao Caetano do Sul Yendis Editora*, 2012.

SANTOS, O. G. F. dos; TIOSSO, F.; PETRUCELLI, E. E. Demonstração dos benefícios do minimum viable product na criação de um novo aplicativo móvel. *Revista Interface Tecnológica*, v. 16, n. 1, p. 124–135, 2019.

SAúDE, M. D. *CARTEIRA DE SERVIÇOS DA ATENÇÃO PRIMÁRIA À SAÚDE (CaSAPS)*. 2020. Disponível em: <http://189.28.128.100/dab/docs/portaldab/documentos/casaps_versao_profissionais_saude_gestores_completa.pdf>.

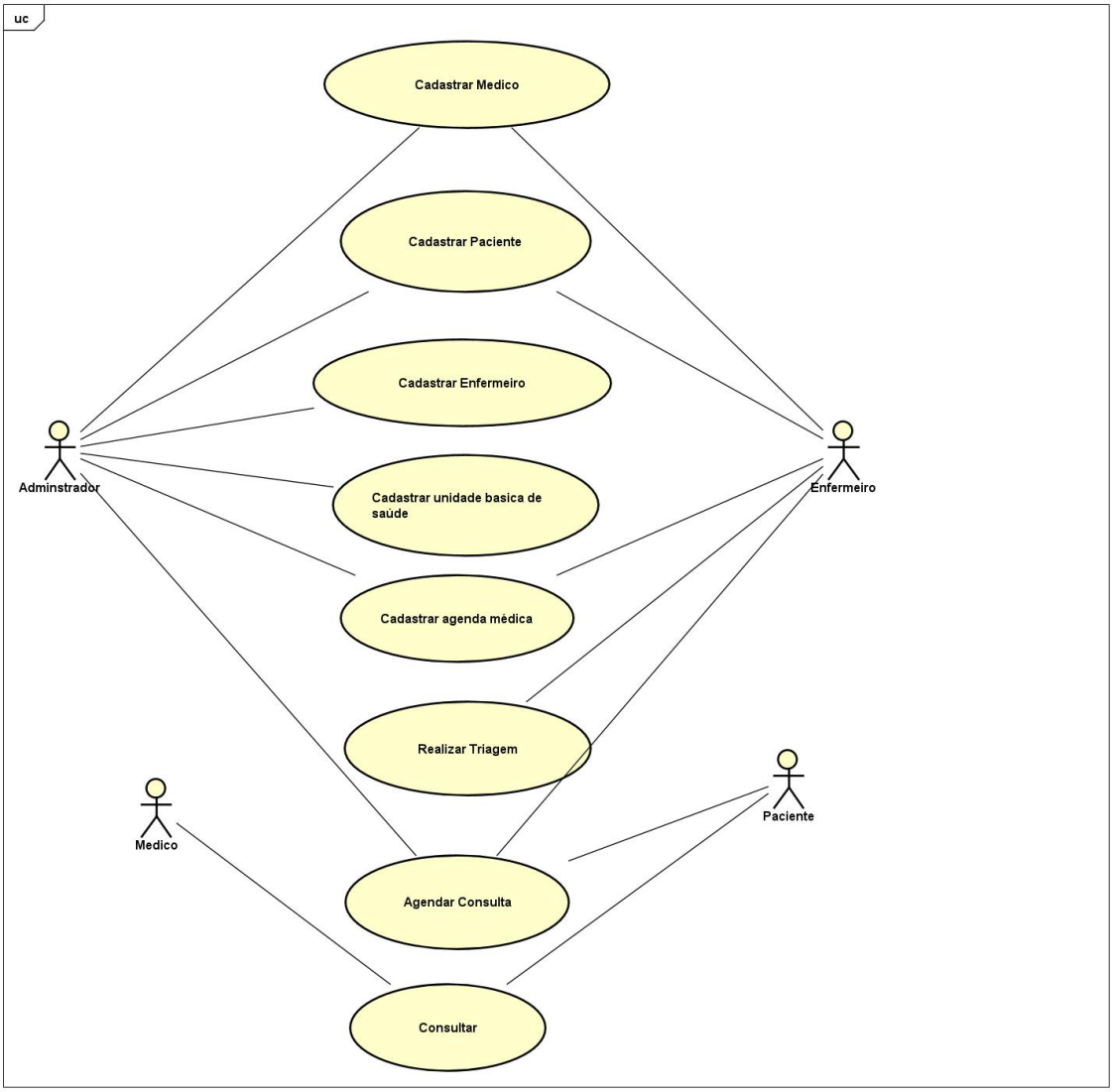
SAúDE, M. da. *O que é Atenção Primária?* 2020. Disponível em: <https://aps.saude.gov.br/smp/smpoquee>.

SILVA, R. C. D. Telemedicina: Requisitos para um sistema de telemedicina com base na certificaÇÃo para sistemas de registro eletrÔnico em saÚde. 2020.

STREHLE, E.; SHABDE, N. One hundred years of telemedicine: does this new technology have a place in paediatrics? *Archives of disease in childhood*, BMJ Publishing Group Ltd, v. 91, n. 12, p. 956–959, 2006.

APÊNDICE A – DIAGRAMA DE CASO DE USO

Figura 7 – Diagrama de caso de uso da aplicação



Fonte:autor

APÊNDICE B – DESCRIÇÃO DOS CASOS DE USO

Caso de uso: Cadastrar unidade básica de saúde.
Atores: Administrador, Sistema
Objetivo: Cadastrar unidades básicas de saúde. **Pré condições:** Permissão para cadastro de unidades básicas no painel administrativo. **Cenário:**

1. O administrador acessa a opção "unidades básicas de saúde."

2. O administrador escolhe a opção "cadastrar unidades básicas de saúde".

3. O sistema exibe um formulário com os campos de nome e endereço.

4. O administrador preenche o formulário e faz a submissão do mesmo.

5. O sistema guarda as informações no banco de dados.

Caso de uso: Cadastrar Médico:
Atores: Enfermeiro, Administrador.
Objetivo: Cadastrar médicos, para que estes usem a aplicação.
Pré condições: Aplicação aberta, ator autenticado **Cenário:**

1. o ator acessa a opção "médicos", e depois "médicos"novamente, depois clica em "cadastrar médico".

2. O Sistema exibe um formulário, com dos dados necessários para cadastrar um médico.

3. O ator submete o formulário devidamente preenchido.

4. O médico é registrado no banco de dados da aplicação

Caso de uso: Cadastrar Especialidade Médica:
Atores: Enfermeiro, Administrador.
Objetivo: Cadastrar especialidades, para que estas sejam associadas aos médicos.
Pré condições: Aplicação aberta, ator autenticado **Cenário:**

1. o ator acessa a opção "médicos", e depois "especialidades".

2. O ator preenche o campo cadastrar especialidade com o nome da especialidade.

3. O ator aperta em cadastrar.

4. A especialidade é registrada no banco de dados.

Caso de uso: Cadastrar Enfermeiro:
Atores: Administrador.
Objetivo: Cadastrar enfermeiros, para que estes usem a aplicação.
Pré condições: Aplicação aberta, ator autenticado **Cenário:**

1. o ator acessa a opção "enfermeiros", e depois "enfermeiros"novamente, depois clica em "cadastrar enfermeiro".

2. O Sistema exibe um formulário, com dos dados necessários para cadastrar um enfermeiro.

3. O ator submete o formulário devidamente preenchido.

4. O enfermeiro é registrado no banco de dados da aplicação

Caso de uso: Login
Atores: Médico, Paciente, Sistema.
Objetivo: Fazer login, e acessar as funcionalidades do sistema.

Pré condições: Aplicação aberta.
Cenário:

1. O ator abre a aplicação

2. O sistema exibe um formulário com os campos "e-mail","senha".

3. O ator preenche os dados.

4. O sistema confirma as informações submetidas pelo ator.

5. O sistema autoriza o autor a fazer acesso as funcionalidades do sistema.

Exceções:

1. O ator preenche dados de e-mail e senha incorretos: O Sistema exibe um modal, avisando ao ator que os dados de acesso estão incorretos.

Caso de uso: Cadastrar agenda
Atores: Enfermeiro, Administrador
Objetivo: Cadastrar agenda medica, para dispor uma organização de horários para atendimento a pacientes
Pré condições: funcionário cadastrado, e autenticados
Cenário:

1. O ator acessa a opção "médicos".

2. O ator acessa a opção "agendas".

3. O ator seleciona "cadastrar"agenda médica,

4. O sistema exibe um formulário com campos de data, hora inicio, e hora fim e seleção de médico.

5. O ator preenche o formulário.

6. O ator clica em cadastrar.

7. O sistema salva a agenda daquele médico no banco de dados.

Exceções:

1. O ator não seleciona nenhum médico: O sistema exibe uma mensagem abaixo do campo de seleção do médico, informando que o um médico deve ser escolhido.

Caso de uso: Agendar consulta
Atores: Paciente, Enfermeiro, Administrador
Objetivo: Agendar consulta com médico.
Pré condições: Paciente, Enfermeiro e Administrador precisa estar cadastrado e autenticado no sistema.
Cenário:

1. Ator acessa a opção "consultas"e depois clica em agendar consultas.

2. O Sistema irá mostrar na tela um formulário, com os campos para selecionar as unidades de saúde, o paciente, a especialidade, e o médico;

3. O ator irá preencher os campos.

4. O Sistema irá mostrar na tela os médicos daquela especialidade naquela unidade de saúde com horários disponíveis.

5. O ator irá selecionar o horário.

6. O ator clica em agendar consulta

7. A consulta então é agendada e armazenada no banco de dados.

Exceções:

1. O ator é o paciente: caso o ator seja um paciente, o sistema não irá exibir o campo para selecionar o paciente no formulário.

Caso de uso: Triagem
Atores: Enfermeiro, Paciente
Objetivo: Realizar uma pré-consulta com o paciente, antes do mesmo se consultar com o médico.
Pré condições: Atores precisam estar cadastrados, logados, e o sistema de cada um deve ter permissões para acessar câmera e microfone.

- Paciente acessar a opção "consultas"e depois "minhas consultas"

- O sistema exibe as consultas agendadas daquele paciente.

- Paciente localiza a consulta agendada, e clica em consultar

- Enfermeiro acessa a opção "consultas"e depois "minhas consultas"

- O sistema exibe as consultas agendadas de todos pacientes.

- Enfermeiro seleciona o paciente agendado para aquele dia e horário.

- A triagem começa, enfermeiro e paciente estão em uma videochamada, e o enfermeiro pode preencher em um formulário chamado triagem, com informações pertinentes.

- Terminando a triagem o enfermeiro clica em gravar dados, e depois finalizar triagem.

- Paciente é retirado da lista do enfermeiro, e adicionado a lista do médico.

Caso de uso: Consultar.
Atores: Médico, Paciente
Objetivo: Realizar uma consulta.
Pré condições: Paciente e médicos, precisam estar cadastrados, logados e sistema de cada um deve ter permissões para acessar câmera e microfone.

- Paciente acessa a opção "consultas"e depois "minhas consultas".

- O sistema exibe as consultas agendadas daquele paciente.

- Paciente seleciona a localiza a consulta agendada, e clica em "consultar".

- Médico acessa a opção "consultas"e depois "minhas consultas"

- O sistema exibe as consultas que já passaram por uma triagem para aquele médico.

- O médico localiza a consulta com o paciente agendado para aquele horário e clica em "realizar consulta".

- Medico e paciente entram em uma videoconferência.

- Medico pode preencher os dados obtidos na conversa com o paciente.

- Médico clica em gravar dados, para gravar os dados da consulta.

- Médico clica em prescrever.

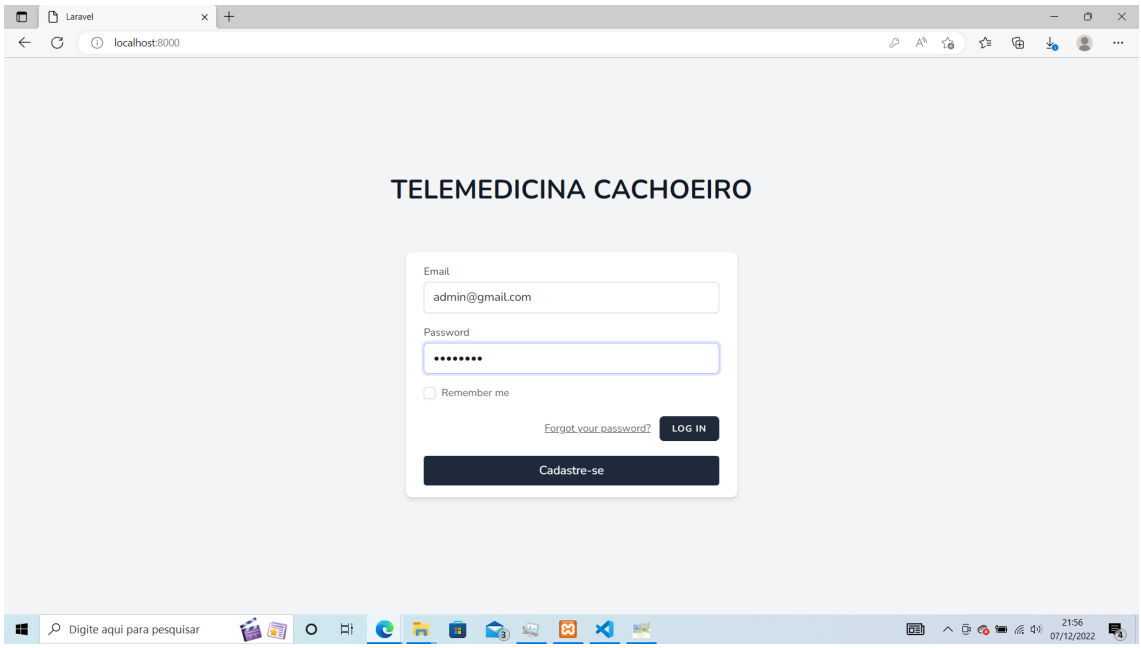
- Médico é direcionado para a plataforma de prescrição do CFM.

- Médico após realizar a prescrição, clica em finalizar consulta.

- Consulta daquele paciente é gravada no banco de dados, e retirada da lista do médico e do paciente.

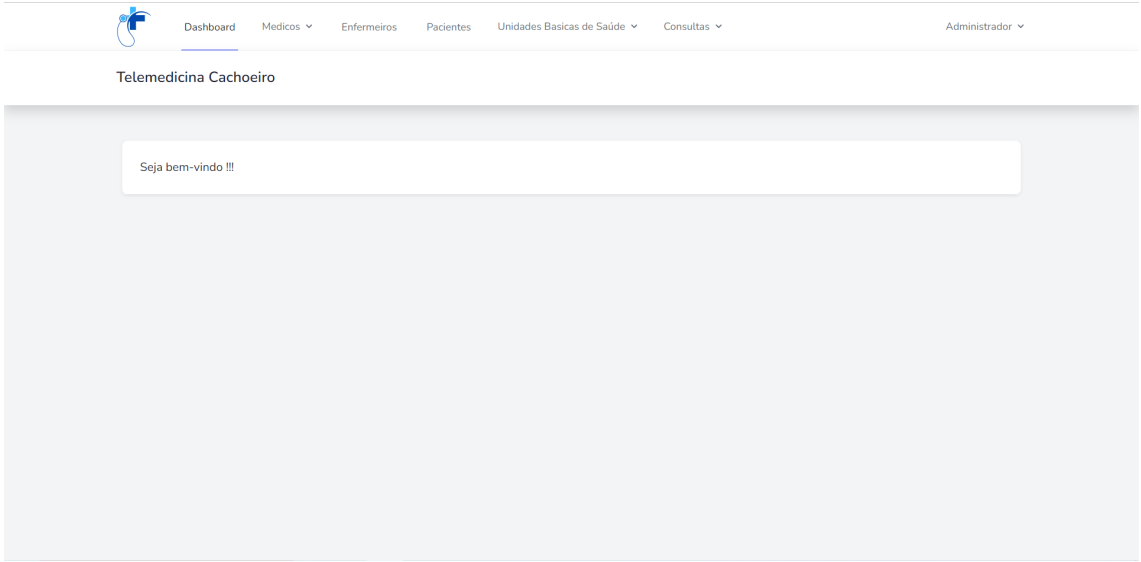
APÊNDICE C – TELAS DA APLICAÇÃO

Figura 8 – Tela de login



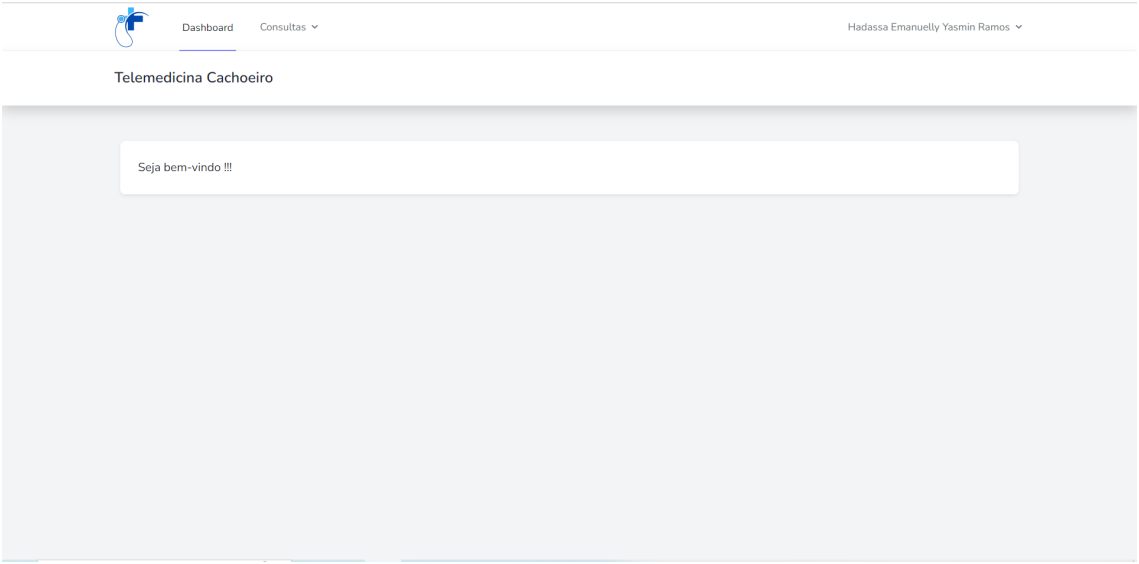
Fonte:Autor

Figura 9 – Tela inicial, logado como administrador



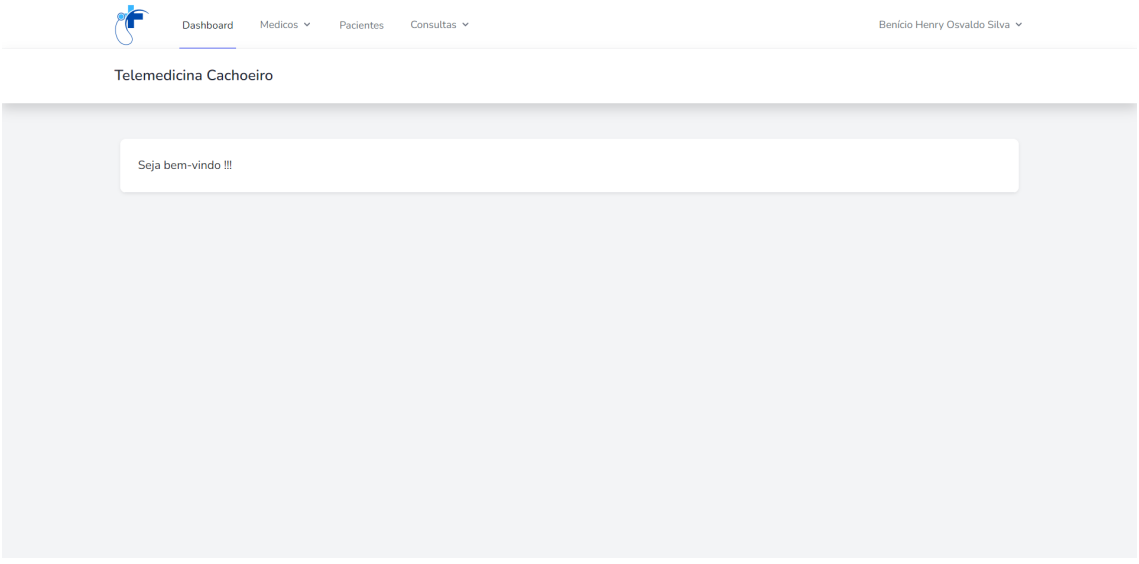
Fonte:Autor

Figura 10 – Tela inicial, logado como paciente



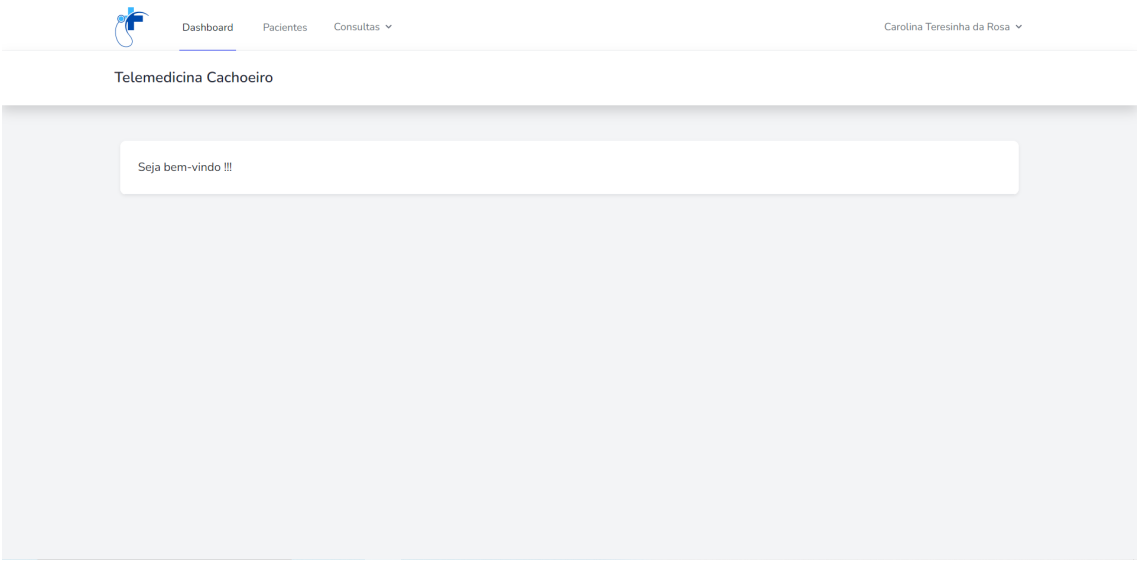
Fonte:Autor

Figura 11 – Tela inicial, logado como enfermeiro



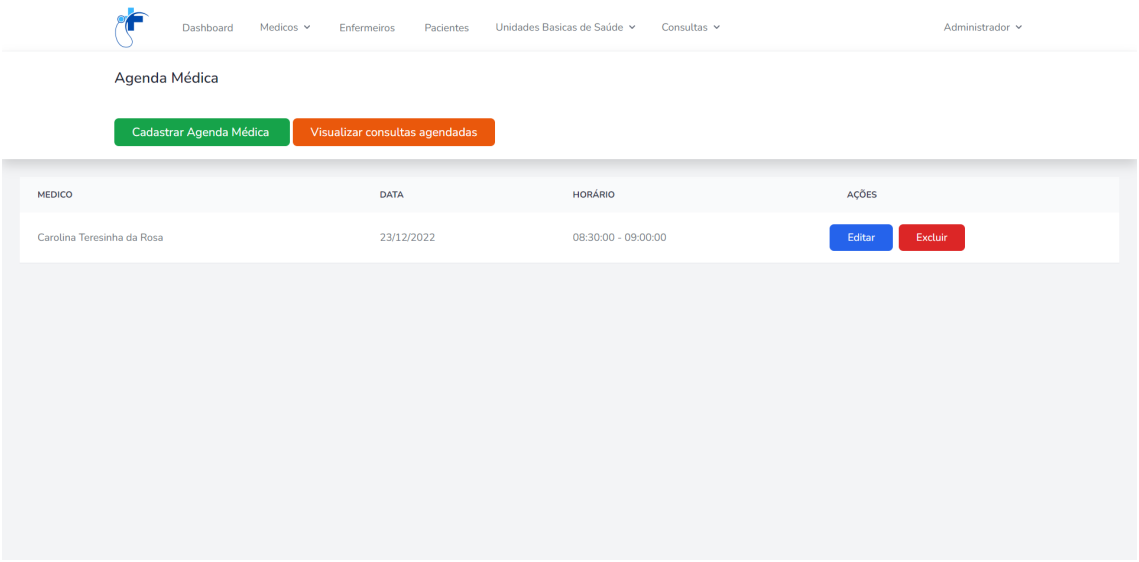
Fonte:Autor

Figura 12 – Tela inicial, logado como medico



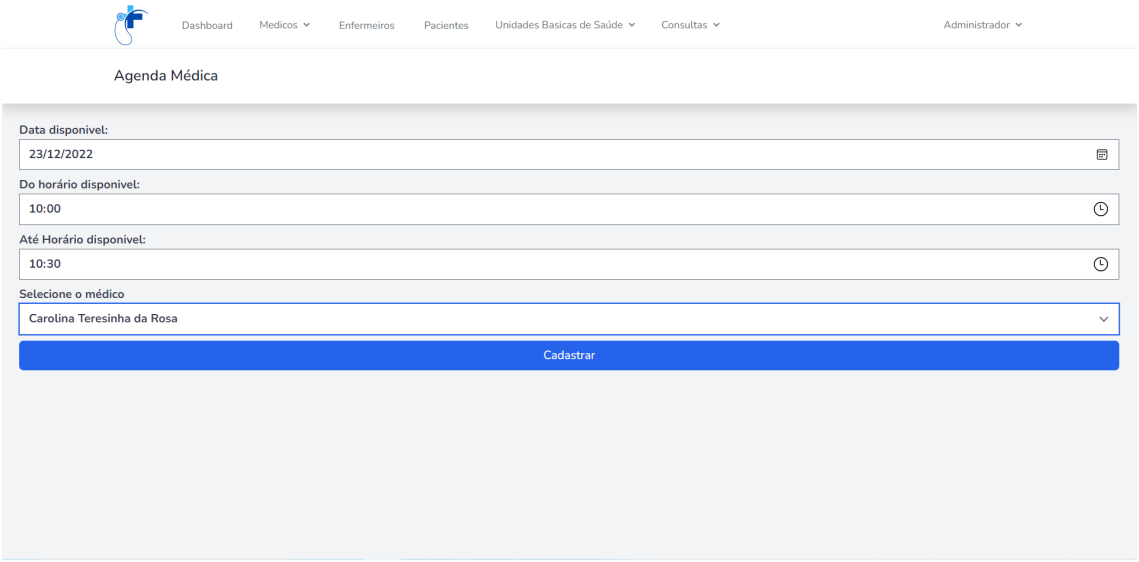
Fonte:Autor

Figura 13 – Gerenciamento de agenda médica, disponível para médico, administrador e enfermeiro



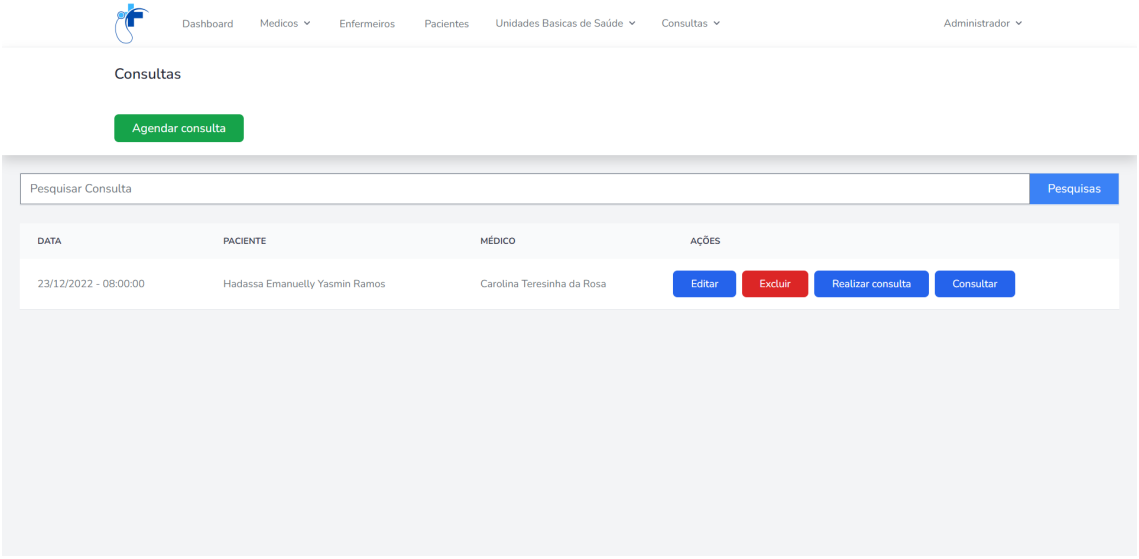
Fonte:Autor

Figura 14 – Cadastro da agenda médica



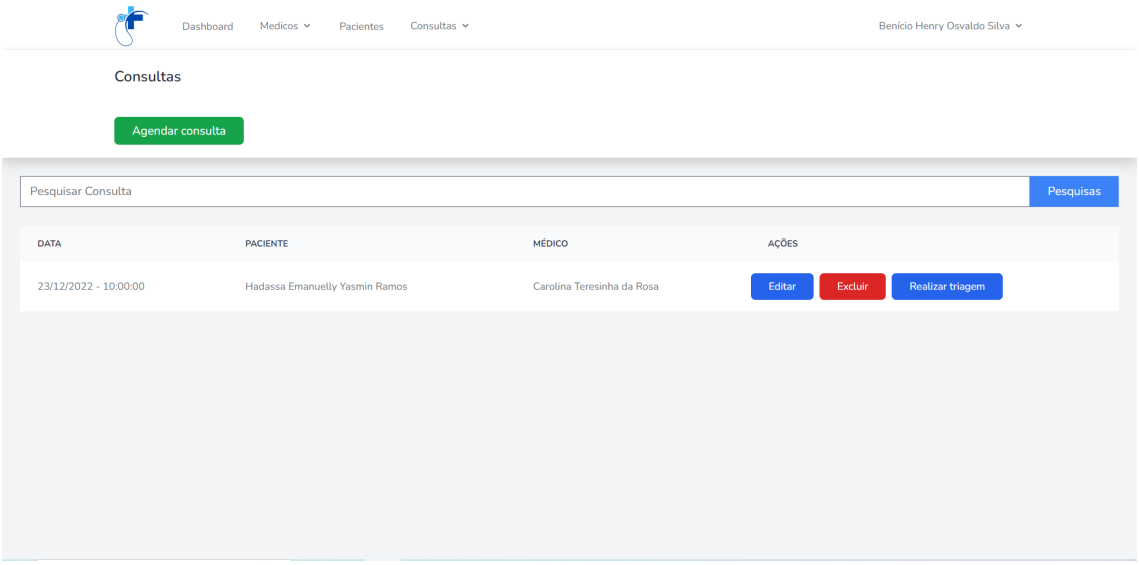
Fonte:Autor

Figura 15 – Tela de gerenciamento de consultas, se o autenticado for medico, somente o botao realizar consulta aparece para ele, caso seja paciente, somente o botão consultar aparece.



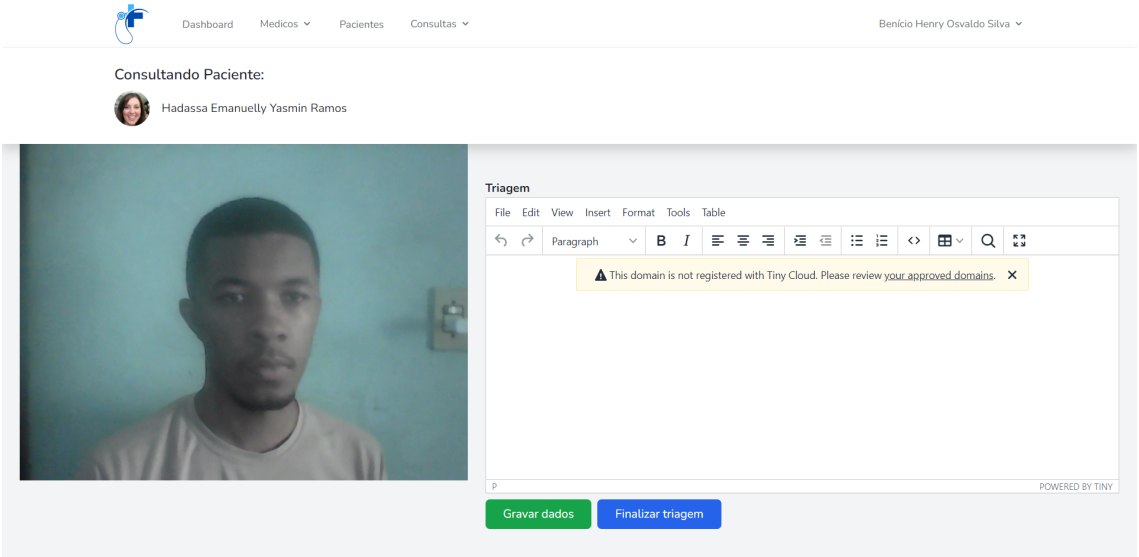
Fonte:Autor

Figura 16 – Tela de gerenciamento de consultas(triagem), do enfermeiro.



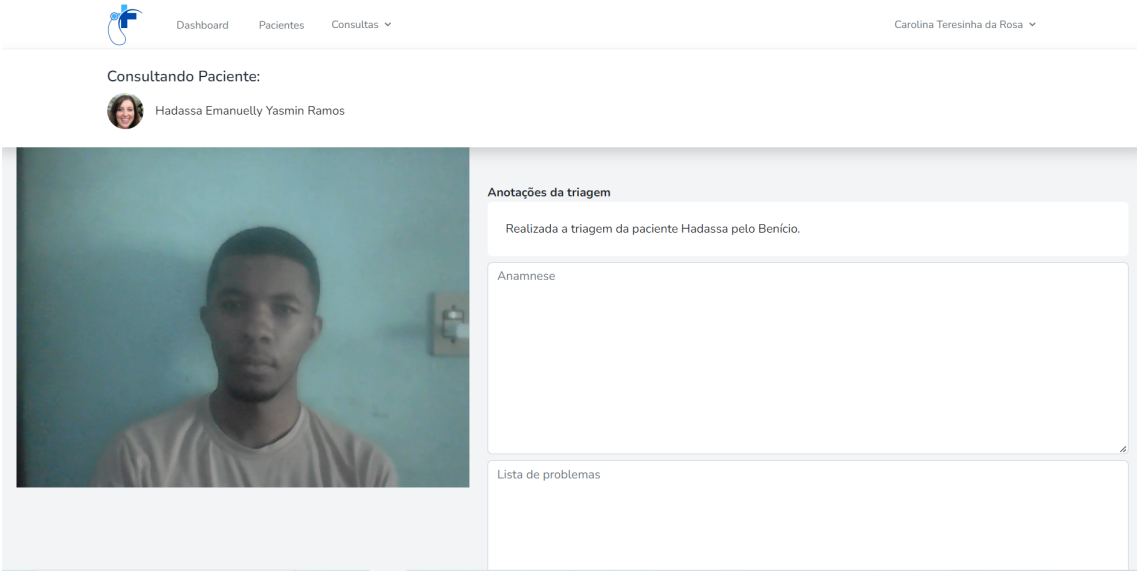
Fonte:Autor

Figura 17 – Tela de triagem do enfermeiro, a imagem do video, vem da camera do paciente.



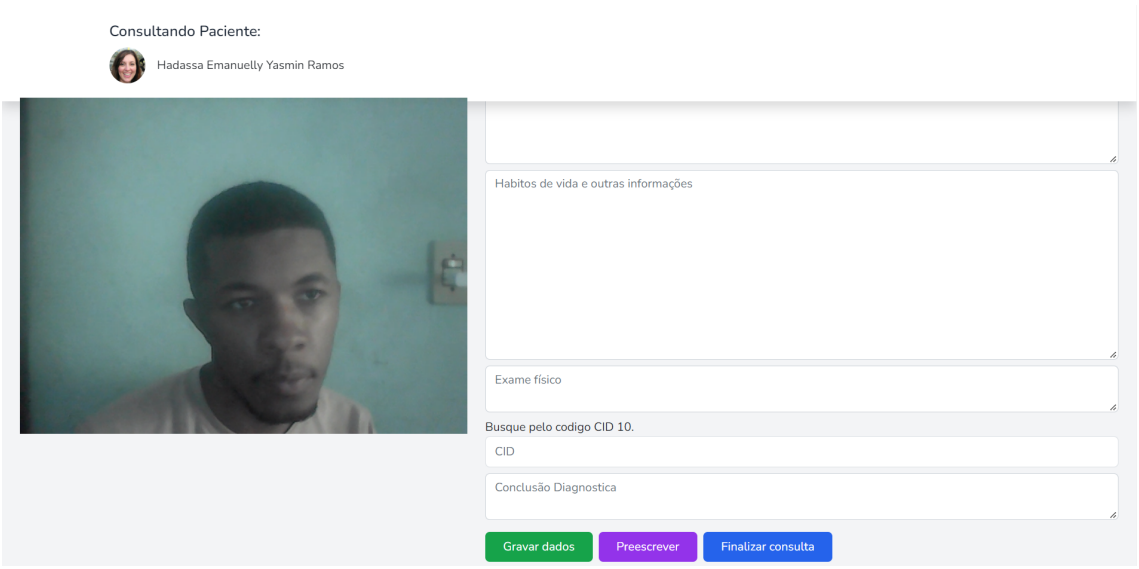
Fonte:Autor

Figura 18 – Tela de consulta do medico, com paciente, a imagem do video, vem da camera do paciente.



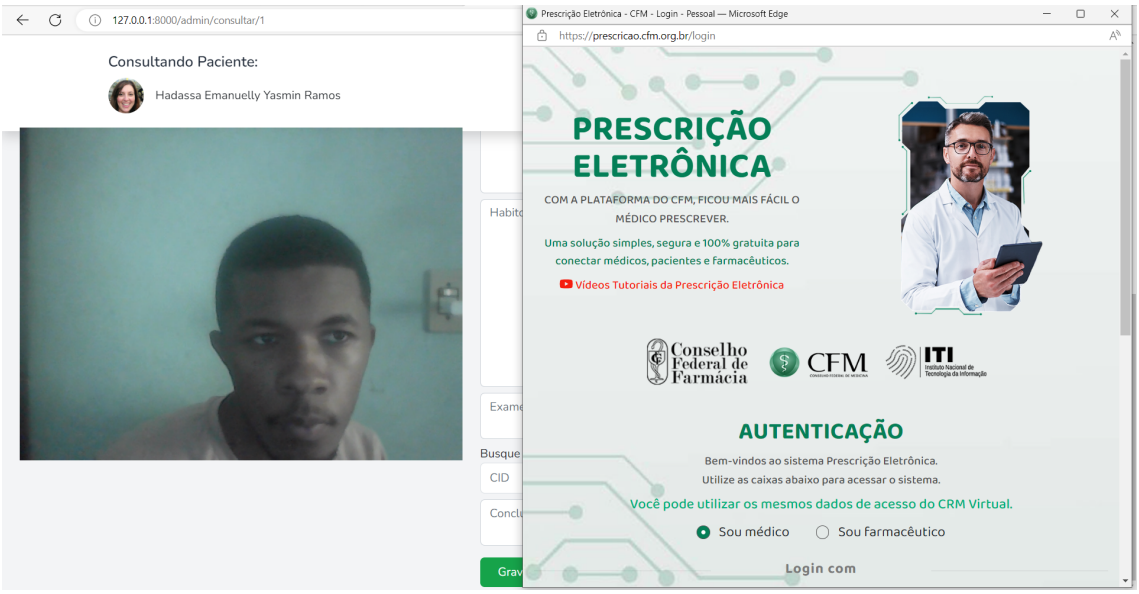
Fonte:Autor

Figura 19 – Tela de consulta do medico, botão de prescrever.



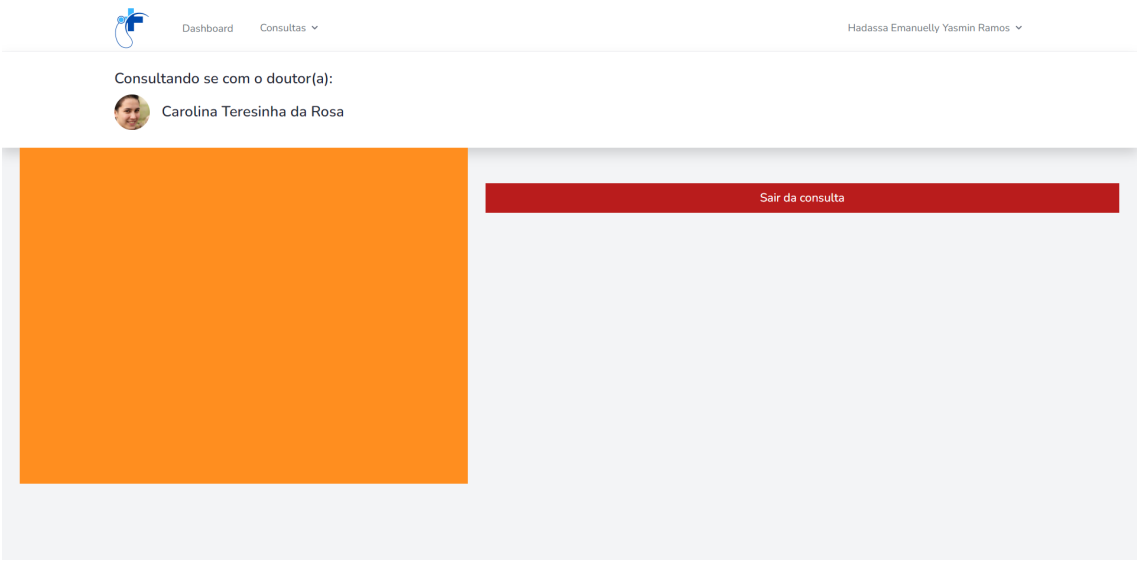
Fonte:Autor

Figura 20 – Tela de consulta do medico, clique no botão prescrever para prescrever na plataforma do cfm.



Fonte:Autor

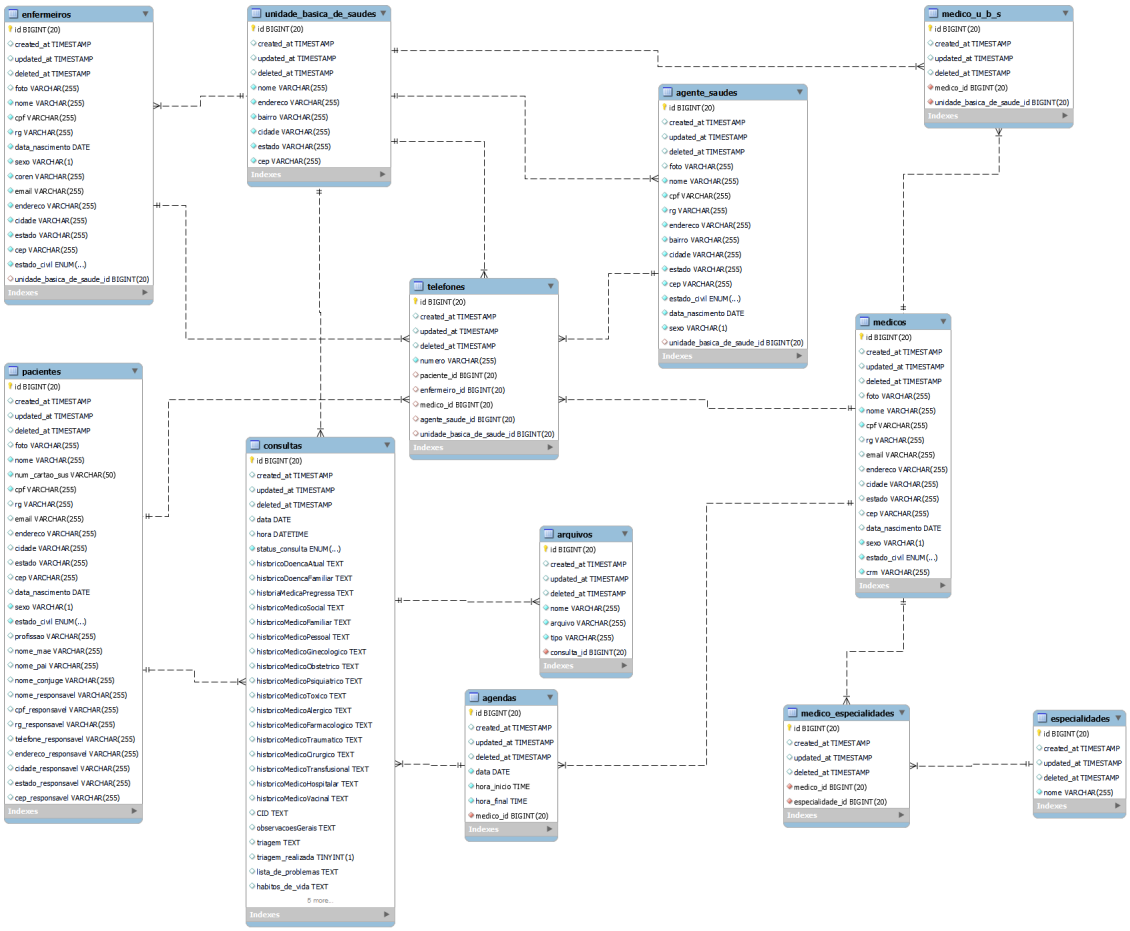
Figura 21 – Tela de consulta do paciente, com o medico, a imagem do video em laranja, vem da camera do médico.



Fonte:Autor

APÊNDICE D – DIAGRAMA DE CLASSES

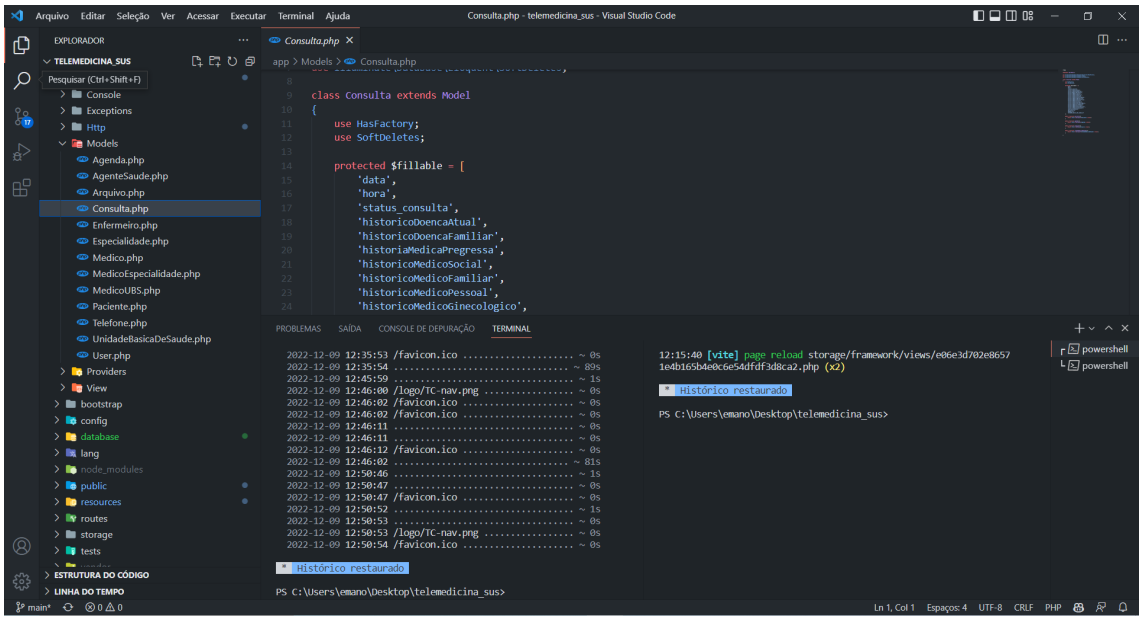
Figura 22 – Diagrama entidade relacional da aplicação



Fonte:Autor

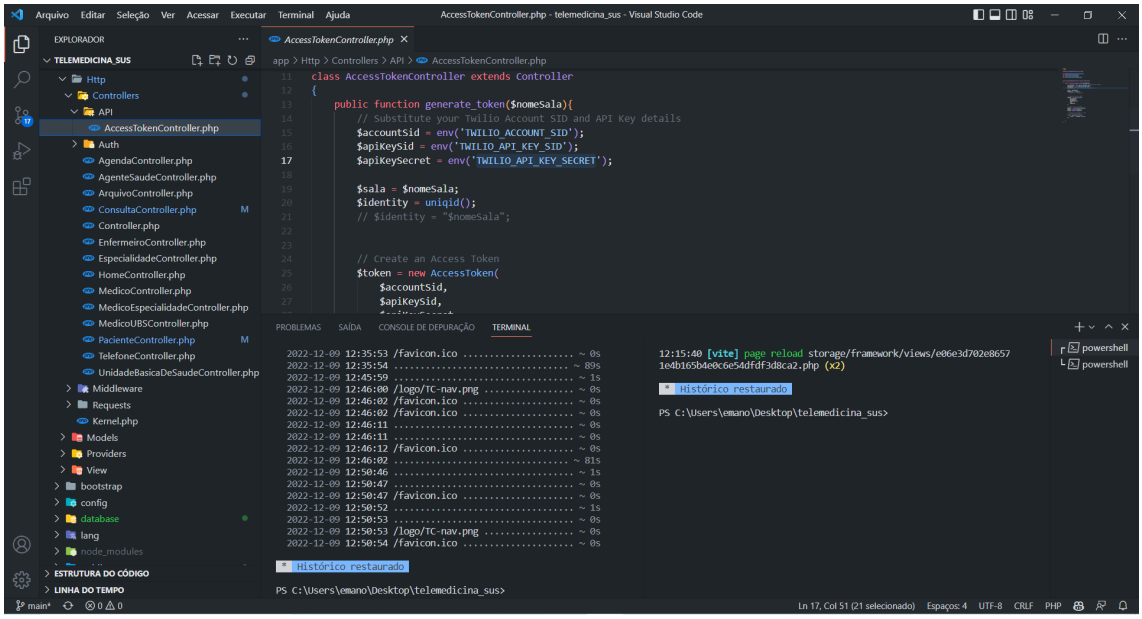
APÊNDICE E – ESTRUTURA DA APLICAÇÃO

Figura 23 – Estrutura do pacote de modelo



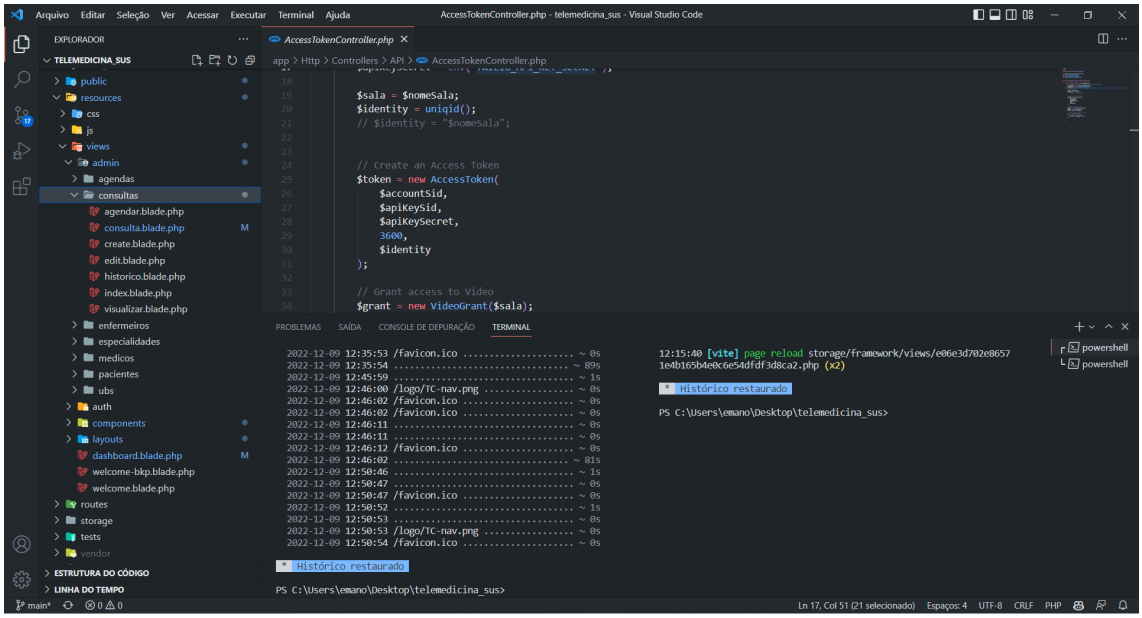
Fonte:Autor

Figura 24 – Estrutura do pacote controler



Fonte:Autor

Figura 25 – Estrutura dos arquivos de visualização



Fonte:Autor

APÊNDICE F – CÓDIGOS DO CONTROLADORES

```php
namespace App\Http\Controllers;

use App\Models\Agenda;
use App\Models\Consulta;
use App\Models\Medico;
use Illuminate\Http\Request;

class AgendaController extends Controller{
    public function index(){
        $agendas = Agenda::join('medicos', 'medicos.id', '=', '
   agendas.medico_id')
        ->whereNotIn('agendas.id', function($query){
            $query->select('agenda_id')->from('consultas');
        })
        ->paginate(6);
        return view("admin.agendas.index", compact("agendas"));
    }
    public function create(){
        $medicos = Medico::all();

        return view("admin.agendas.create", compact("medicos"))
   ;
    }
    public function agendaDoMedico($id){
        $agendas = Agenda::where('medico_id', $id)
        ->whereNotIn('agendas.id', function($query){
            $query->select('agenda_id')->from('consultas');
        })
        ->get();
        return response()->json($agendas);
    }
```

```
30      public function store(Request $request){
31          $regras = [
32              'data' => 'required',
33              'hora_inicio' => 'required',
34              'hora_final' => 'required',
35              'medico_id' => 'required',
36          ];
37
38          $feedback = [
39              'required' => 'O campo :attribute e obrigatorio',
40          ];
41
42          $request->validate($regras, $feedback);
43
44          $agenda = Agenda::create($request->all());
45
46          return redirect()->route('admin.agendas.index');
47      }
48      public function edit(Agenda $agenda){
49          $medicos = Medico::all();
50          return view("admin.agendas.edit", compact("agenda", "
    medicos"));
51      }
52      public function update(Request $request, $id){
53          $regras = [
54              'data' => 'required',
55              'hora_inicio' => 'required',
56              'hora_final' => 'required',
57              'medico_id' => 'required',
58          ];
59
60          $feedback = [
61              'required' => 'O campo :attribute e obrigatorio',
```

```
62          ];
63
64          $request->validate($regras, $feedback);
65
66          $agenda = Agenda::find($id);
67          $agenda->update($request->all());
68
69          return redirect()->route('admin.agendas.index');
70      }
71      public function destroy($id){
72          $agenda = Agenda::find($id);
73          $agenda->delete();
74
75          return redirect()->route('admin.agendas.index');
76      }
77  }
```

Listing F.1 – Controlador responsavel por gerenciar agendas médicas

```
1   namespace App\Http\Controllers;
2
3   use App\Models\Arquivo;
4   use Illuminate\Http\Request;
5   use Illuminate\Support\Facades\File;
6
7   class ArquivoController extends Controller{
8       /**
9        * Display a listing of the resource.
10       *
11       * @return \Illuminate\Http\Response
12       */
13      public function store(Request $request)
14      {
15          $regras = [
```

```
16              "nome" => "required"
17          ];
18
19          $feedback = [
20              "required" => "O campo :attribute    obrigat rio"
21          ];
22
23          $request->validate($regras, $feedback);
24
25          $arquivo = new Arquivo();
26
27          $arquivo->nome = $request->nome;
28          $arquivo->consulta_id = $request->consulta_id;
29
30          if($request->hasFile("arquivo") && $request->file("
    arquivo")->isValid()){
31              $requestArquivo = $request->arquivo;
32              $extension = $requestArquivo->extension();
33              $arquivoName = $requestArquivo->
    getClientOriginalName() . strtotime("now") . "." .
    $extension;
34              $requestArquivo->move(public_path("consultas/
    $request->consulta_id/arquivos"), $arquivoName);
35
36              $arquivo->arquivo = $arquivoName;
37              $arquivo->tipo = $extension;
38          }
39
40          $arquivo->save();
41      }
42      public function update(Request $request, Arquivo $arquivo){
43
44          $regras = [
```

```
45              "nome" => "required"
46          ];
47
48          $feedback = [
49              "required" => "O campo :attribute    obrigat rio"
50          ];
51
52          $request->validate($regras, $feedback);
53
54          $arquivo = Arquivo::withoutTrashed()->find($arquivo->id
    );
55          $arquivo->update($request->all());
56
57          if($request->hasFile("arquivo") && $request->file("
    arquivo")->isValid()){
58              File::delete(public_path("consultas/$request->
    consulta_id/arquivos/$arquivo->arquivo"));
59              $requestArquivo = $request->arquivo;
60              $extension = $requestArquivo->extension();
61              $arquivoName = $requestArquivo->
    getClientOriginalName() . strtotime("now") . "." .
    $extension;
62              $requestArquivo->move(public_path("consultas/
    $request->consulta_id/arquivos"), $arquivoName);
63
64              $arquivo->arquivo = $arquivoName;
65              $arquivo->tipo = $extension;
66          }
67
68          $arquivo->save();
69
70      }
71
```

```
72      public function destroy(Arquivo $arquivo){
73          $arquivo = Arquivo::withoutTrashed()->find($arquivo->id
    );
74          File::delete(public_path("consultas/$arquivo->
    consulta_id/arquivos/$arquivo->arquivo"));
75          $arquivo->forceDelete();
76      }
77  }
```

Listing F.2 – Controlador responsavel por inserir arquivos em uma consulta

```
1   <?php
2
3   namespace App\Http\Controllers;
4
5   use App\Http\Controllers\API\AccessTokenController;
6   use App\Models\Consulta;
7   use App\Models\Especialidade;
8   use App\Models\Paciente;
9   use App\Models\UnidadeBasicaDeSaude;
10  use App\Models\User;
11  use Illuminate\Http\Request;
12  use Illuminate\Support\Facades\Auth;
13  use Twilio\Rest\Client;
14
15  class ConsultaController extends Controller{
16      public function index(){
17          $mensagem = "";
18          $user = User::find(auth()->user()->id);
19
20          if($user->hasRole('paciente')){
21
22              $paciente = Paciente::where('id', $user->
    userable_id)->first();
```

```
23              $consultas = Consulta::join("pacientes", "pacientes
    .id", "=", "consultas.paciente_id")
24              ->join("agendas", "agendas.id", "=", "consultas.
    agenda_id")
25              ->join("medicos", "medicos.id", "=", "agendas.
    medico_id")
26              ->select("agendas.data AS data_agenda","agendas.
    hora_inicio","agendas.hora_final","medicos.nome as
    medico_nome","pacientes.nome as paciente","consultas.*")
27              ->where('paciente_id', $paciente->id)
28              ->where("consultas.status_consulta","<>","realizada
    ")
29              ->paginate(6);
30          }else if($user->hasRole('medico')){
31              $medico = User::find(auth()->user()->id)->userable;
32              $consultas = Consulta::join("pacientes", "pacientes
    .id", "=", "consultas.paciente_id")
33              ->join("agendas", "agendas.id", "=", "consultas.
    agenda_id")
34              ->join("medicos", "medicos.id", "=", "agendas.
    medico_id")
35              ->select("agendas.data AS data_agenda","agendas.
    hora_inicio","agendas.hora_final","medicos.nome as
    medico_nome","pacientes.nome as paciente","consultas.*")
36              ->where('medico_id', $medico->id)
37              ->where("consultas.status_consulta","<>","realizada
    ")
38              ->where("consultas.triagem_realizada", 1)
39              ->paginate(6);
40          }elseif($user->hasRole('enfermeiro')){
41              $consultas = Consulta::join("pacientes", "pacientes
    .id", "=", "consultas.paciente_id")
42              ->join("agendas", "agendas.id", "=", "consultas.
```

```
agenda_id")
43              ->join("medicos", "medicos.id", "=", "agendas.
medico_id")
44              ->where("consultas.triagem_realizada", 0)
45              ->select("agendas.data AS data_agenda","agendas.
hora_inicio","agendas.hora_final","medicos.nome as
medico_nome","pacientes.nome as paciente","consultas.*")
46              ->paginate(6);
47          }else{
48              $consultas = Consulta::join("pacientes", "pacientes
.id", "=", "consultas.paciente_id")
49              ->join("agendas", "agendas.id", "=", "consultas.
agenda_id")
50              ->join("medicos", "medicos.id", "=", "agendas.
medico_id")
51              ->select("agendas.data AS data_agenda","agendas.
hora_inicio","agendas.hora_final","medicos.nome as
medico_nome","pacientes.nome as paciente","consultas.*")
52              ->paginate(6);
53          }
54
55          return view("admin.consultas.index", compact("consultas
", "mensagem"));
56      }
57
58      public function agendar(){
59
60          $mensagem = "";
61          $ubs = UnidadeBasicaDeSaude::all();
62          $especialidades = Especialidade::all();
63          $pacientes = Paciente::all();
64          return view("admin.consultas.agendar", compact("
mensagem", "especialidades","pacientes","ubs"));
```

```
65      }
66
67      public function store(Request $request){
68          $regras = [
69              'agenda_id' => 'required',
70              'paciente' => 'required',
71              'medico' => 'required',
72          ];
73
74          $feedback = [
75              'required' => 'O campo :attribute    obrigat rio',
76          ];
77
78          $request->validate($regras, $feedback);
79
80          $consulta = new Consulta();
81
82          $idPaciente = Auth::user()->userable_id;
83          $paciente = Paciente::find($idPaciente);
84
85          $consulta->agenda_id = $request->agenda_id;
86          $consulta->paciente_id = $paciente->id ?? $request->
    paciente;
87          $consulta->unidade_basica_de_saude_id = $request->
    unidade_basica_de_saude_id;
88          $consulta->status_consulta = "agendada";
89          $consulta->save();
90
91          $msg = "Consulta agendada com sucesso!";
92
93          return redirect()->route("admin.consultas.index")->with
    ("msg", $msg);
94      }
```

```
95
96      public function update(Request $request, Consulta $consulta
        ){
97          date_default_timezone_set('America/Sao_Paulo');
98
99          $consulta = Consulta::find($consulta->id);
100         $consulta->habitos_de_vida = $request->habitos_de_vida;
101         $consulta->lista_de_problemas = $request->
        lista_de_problemas;
102         $consulta->exame_fisico = $request->exame_fisico;
103         $consulta->conclusao_diagnostica = $request->
        conclusao_diagnostica;
104         $consulta->historicoDoencaAtual =  $request->
        historicoDoencaAtual ? $request->historicoDoencaAtual:NULL;
105         $consulta->historicoDoencaFamiliar =  $request->
        historicoDoencaFamiliar ? $request->historicoDoencaFamiliar:
        NULL;
106         $consulta->historiaMedicaPregressa =  $request->
        historiaMedicaPregressa ? $request->historiaMedicaPregressa:
        NULL;
107         $consulta->historicoMedicoSocial =  $request->
        historicoMedicoSocial ? $request->historicoMedicoSocial:NULL
        ;
108         $consulta->historicoMedicoFamiliar =  $request->
        historicoMedicoFamiliar ? $request->historicoMedicoFamiliar:
        NULL;
109         $consulta->historicoMedicoPessoal =  $request->
        historicoMedicoPessoal ? $request->historicoMedicoPessoal:
        NULL;
110         $consulta->historicoMedicoGinecologico =  $request->
        historicoMedicoGinecologico ? $request->
        historicoMedicoGinecologico:NULL;
111         $consulta->historicoMedicoObstetrico =  $request->
```

```
    historicoMedicoObstetrico ? $request->
    historicoMedicoObstetrico:NULL;
112         $consulta->historicoMedicoPsiquiatrico =  $request->
    historicoMedicoPsiquiatrico ? $request->
    historicoMedicoPsiquiatrico:NULL;
113         $consulta->historicoMedicoAlergico =  $request->
    historicoMedicoAlergico ? $request->historicoMedicoAlergico:
    NULL;
114         $consulta->historicoMedicoFarmacologico =  $request->
    historicoMedicoFarmacologico ? $request->
    historicoMedicoFarmacologico:NULL;
115         $consulta->historicoMedicoTraumatico =  $request->
    historicoMedicoTraumatico ? $request->
    historicoMedicoTraumatico:NULL;
116         $consulta->historicoMedicoCirurgico =  $request->
    historicoMedicoCirurgico ? $request->
    historicoMedicoCirurgico:NULL;
117         $consulta->historicoMedicoTransfusional =  $request->
    historicoMedicoTransfusional ? $request->
    historicoMedicoTransfusional:NULL;
118         $consulta->historicoMedicoHospitalar =  $request->
    historicoMedicoHospitalar ? $request->
    historicoMedicoHospitalar:NULL;
119         $consulta->historicoMedicoVacinal =  $request->
    historicoMedicoVacinal ? $request->historicoMedicoVacinal:
    NULL;
120         $consulta->CID =  $request->CID ? $request->CID:NULL;
121         $consulta->observacoesGerais =  $request->
    observacoes_gerais ? $request->observacoes_gerais:NULL;
122         $consulta->data = date("Y-m-d");
123         $consulta->hora = now();
124         $consulta->triagem = $request->triagem;
125         $consulta->status_consulta = "acontecendo";
```

```
126
127         $consulta->save();
128
129         $msg = "Dados gravados com sucesso!";
130         return redirect()->route("admin.consultar", $consulta->
    id)->with("msg", $msg);
131      }
132
133      public function finalizarConsulta(Consulta $consulta){
134         $consulta = Consulta::find($consulta->id);
135         $consulta->status_consulta = "realizada";
136         $consulta->save();
137
138         $msg = "Consulta finalizada com sucesso!";
139         return redirect()->route("admin.consultas.index")->with
    ("msg", $msg);
140      }
141
142      public function finalizarTriagem(Consulta $consulta){
143         $consulta = Consulta::find($consulta->id);
144         $consulta->triagem_realizada = 1;
145         $consulta->save();
146
147         $msg = "Triagem finalizada com sucesso!";
148         return redirect()->route("admin.consultas.index")->with
    ("msg", $msg);
149      }
150
151      public function destroy($id){
152         $consulta = Consulta::find($id);
153         $consulta->delete();
154
155         $msg = "Consulta exclu da com sucesso!";
```

```
156         return redirect()->route("admin.consultas.index")->with
    ("msg", $msg);
157      }
158
159      public function consultar(Consulta $consulta){
160          $typeUser = Auth::user()->userable_type;
161
162          $sala = $consulta->id.$consulta->paciente_id;
163
164          if($typeUser == "App\Models\Paciente"){
165              $consulta = Consulta::join("pacientes", "pacientes.
    id","=","consultas.paciente_id")
166              ->join("agendas", "agendas.id","=","consultas.
    agenda_id")
167              ->join("medicos", "medicos.id","=","agendas.
    medico_id")
168              ->where("consultas.id", $consulta->id)
169              ->select("pacientes.nome AS paciente_nome","
    pacientes.foto AS paciente_foto","medicos.nome AS
    medico_nome","medicos.foto AS medico_foto","consultas.*")
170              ->first();
171
172              return view("admin.consultas.consulta", compact("
    consulta", "sala"));
173
174          }elseif($typeUser == "App\Models\Medico"){
175
176              $consulta = Consulta::join("pacientes", "pacientes.
    id","=","consultas.paciente_id")
177              ->where("consultas.id", $consulta->id)
178              ->select("pacientes.nome AS paciente_nome","
    pacientes.foto AS paciente_foto" ,"consultas.*")
179              ->first();
```

```
180
181             return view("admin.consultas.consulta", compact("
    consulta","sala"));
182         }elseif($typeUser == "App\Models\Enfermeiro"){
183             $mensagem = "Sala de triagem";
184             $consulta = Consulta::join("pacientes", "pacientes.
    id","=","consultas.paciente_id")
185             ->where("consultas.id", $consulta->id)
186             ->select("pacientes.nome AS paciente_nome","
    pacientes.foto AS paciente_foto" ,"consultas.*")
187             ->first();
188
189             return view("admin.consultas.consulta", compact("
    consulta","sala","mensagem"));
190         }
191     }
192
193     public function historicoPaciente($paciente){
194         $historico = Consulta::join("pacientes", "pacientes.id"
    ,"=","consultas.paciente_id")
195         ->join("agendas", "agendas.id","=","consultas.agenda_id
    ")
196         ->join("medicos", "medicos.id","=","agendas.medico_id")
197         ->where("pacientes.id", $paciente)
198         ->where("consultas.status_consulta", "realizada")
199         ->select("consultas.*", "pacientes.nome AS
    paciente_nome", "medicos.nome AS medico_nome")
200         ->paginate(12);
201
202         return view("admin.consultas.historico", compact("
    historico"));
203     }
204
```

```
205     public function visualizarConsulta($consulta){
206         $consulta = Consulta::join("pacientes", "pacientes.id",
    "=","consultas.paciente_id")
207         ->join("agendas", "agendas.id","=","consultas.agenda_id
    ")
208         ->join("medicos", "medicos.id","=","agendas.medico_id")
209         ->where("consultas.id", $consulta)
210         ->select("consultas.*", "pacientes.nome AS
    paciente_nome", "pacientes.foto AS paciente_foto","pacientes
    .id AS id_paciente","medicos.nome AS medico_nome")
211         ->first();
212
213         return view("admin.consultas.visualizar", compact("
    consulta"));
214     }
215 }
```

Listing F.3 – Controlador responsavel por gerenciar consultas médicas

```
1  namespace App\Http\Controllers;
2
3  use App\Models\Enfermeiro;
4  use App\Models\Telefone;
5  use App\Models\User;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\File;
8
9  class EnfermeiroController extends Controller{
10
11     public function index(){
12         if(request()->search){
13             $enfermeiros = Enfermeiro::where('nome', 'like', '%
    '.request()->search.'%')
14             ->orWhere('coren', 'like', '%'.request()->search.'%
```

```
    ')
            ->paginate(6);
        }else{
            $enfermeiros = Enfermeiro::paginate(6);
        }

        return view("admin.enfermeiros.index", compact("
    enfermeiros"));
    }

    public function create(){
        return view("admin.enfermeiros.create");
    }

    public function store(Request $request)
    {
        $regras = [
            'nome' => 'required|min:3',
            'cpf' => 'required|min:11',
            'rg' => 'required|min:9',
            'data_nascimento' => 'required|min:10',
            'sexo' => 'required|min:1',
            'coren' => 'required|min:3',
            'email' => 'required|min:3',
            'endereco' => 'required|min:3',
            'cidade' => 'required|min:3',
            'estado' => 'required|min:2',
            'cep' => 'required|min:8',
            'estado_civil' => 'required|min:3',
            'telefone' => 'required|min:3',
            'senha' => 'required|min:3',
            'confirma_senha' => 'required|min:3|same:senha'
        ];
```

```
46
47          $feedback = [
48              'required' => 'O campo :attribute    obrigat rio',
49              'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
50              'cpf.min' => 'O campo cpf deve ter no m nimo 11
    caracteres',
51              'rg.min' => 'O campo rg deve ter no m nimo 9
    caracteres',
52              'data_nascimento.min' => 'O campo data de
    nascimento deve ter no m nimo 10 caracteres',
53              'sexo.min' => 'O campo sexo deve ter no m nimo 1
    caracteres',
54              'coren.min' => 'O campo coren deve ter no m nimo 3
     caracteres',
55              'email.min' => 'O campo email deve ter no m nimo 3
     caracteres',
56              'endereco.min' => 'O campo endere o deve ter no
    m nimo 3 caracteres',
57              'cidade.min' => 'O campo cidade deve ter no m nimo
     3 caracteres',
58              'estado.min' => 'O campo estado deve ter no m nimo
     2 caracteres',
59              'cep.min' => 'O campo cep deve ter no m nimo 8
    caracteres',
60              'estado_civil.min' => 'O campo estado civil deve
    ter no m nimo 3 caracteres',
61              'telefone.min' => 'O campo telefone deve ter no
    m nimo 3 caracteres',
62              'senha.min' => 'O campo senha deve ter no m nimo 3
     caracteres',
63              'confirmar_senha.min' => 'O campo confirmar senha
    deve ter no m nimo 3 caracteres',
```

```
64              'confirmar_senha.same' => 'O campo confirmar senha
    deve ser igual ao campo senha'
65          ];
66
67          $request->validate($regras, $feedback);
68
69          $enfermeiro = Enfermeiro::create($request->all());
70
71          if($request->hasFile('foto') && $request->file('foto')
    ->isValid()){
72              $requestImage = $request->foto;
73              $extension = $requestImage->extension();
74              $imageName = md5($requestImage->
    getClientOriginalName() . strtotime("now")) . "." .
    $extension;
75              $requestImage->move(public_path('img/enfermeiros'),
     $imageName);
76              $enfermeiro->foto = $imageName;
77              $enfermeiro->save();
78          }
79
80          $telefone = new Telefone();
81          $telefone->numero = $request->telefone;
82          $telefone->enfermeiro_id = $enfermeiro->id;
83          $telefone->save();
84
85          $user = new User();
86          $user->name = $request->nome;
87          $user->email = $request->email;
88          $user->password = bcrypt($request->senha);
89          $user->userable_id = $enfermeiro->id;
90          $user->userable_type = 'App\Models\Enfermeiro';
91          $user->assignRole('enfermeiro');
```

```
92          $user->save();
93
94          return redirect()->route('admin.enfermeiros.index')->
    with('msg', 'Enfermeiro cadastrado com sucesso!');
95      }
96
97      public function edit(Enfermeiro $enfermeiro){
98          $enfermeiro = Enfermeiro::find($enfermeiro->id);
99          $telefone = Telefone::where('enfermeiro_id',
    $enfermeiro->id)->first();
100         return view("admin.enfermeiros.edit", compact("
    enfermeiro", "telefone"));
101     }
102
103     public function update(Request $request, Enfermeiro
    $enfermeiro)
104     {
105         $regras = [
106             'nome' => 'required|min:3',
107             'cpf' => 'required|min:11',
108             'rg' => 'required|min:9',
109             'data_nascimento' => 'required|min:10',
110             'sexo' => 'required|min:1',
111             'coren' => 'required|min:3',
112             'email' => 'required|min:3',
113             'endereco' => 'required|min:3',
114             'cidade' => 'required|min:3',
115             'estado' => 'required|min:2',
116             'cep' => 'required|min:8',
117             'estado_civil' => 'required',
118         ];
119
120         $feedback = [
```

```
121             'required' => 'O campo :attribute    obrigat rio',
122             'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
123             'cpf.min' => 'O campo cpf deve ter no m nimo 11
    caracteres',
124             'rg.min' => 'O campo rg deve ter no m nimo 9
    caracteres',
125             'data_nascimento.min' => 'O campo data de
    nascimento deve ter no m nimo 10 caracteres',
126             'sexo.min' => 'O campo sexo deve ter no m nimo 1
    caracteres',
127             'coren.min' => 'O campo coren deve ter no m nimo 3
     caracteres',
128             'email.min' => 'O campo email deve ter no m nimo 3
     caracteres',
129             'endereco.min' => 'O campo endere o deve ter no
    m nimo 3 caracteres',
130             'cidade.min' => 'O campo cidade deve ter no m nimo
     3 caracteres',
131             'estado.min' => 'O campo estado deve ter no m nimo
     2 caracteres',
132             'cep.min' => 'O campo cep deve ter no m nimo 8
    caracteres',
133             'estado_civil.required' => 'O campo estado civil
    deve ser preenchido',
134         ];
135
136         $request->validate($regras, $feedback);
137
138         $enfermeiro = Enfermeiro::withTrashed()->find(
    $enfermeiro->id);
139         $enfermeiro->update($request->all());
140
```

```
141         if($request->hasFile('foto') && $request->file('foto')
    ->isValid()){
142             File::delete(public_path('img/enfermeiros/' .
    $enfermeiro->foto));
143             $requestImage = $request->foto;
144             $extension = $requestImage->extension();
145             $imageName = md5($requestImage->
    getClientOriginalName() . strtotime("now")) . "." .
    $extension;
146             $requestImage->move(public_path('img/enfermeiros'),
     $imageName);
147             $enfermeiro->foto = $imageName;
148             $enfermeiro->save();
149         }
150
151         $telefone = Telefone::where('enfermeiro_id',
    $enfermeiro->id)->first();
152         $telefone->numero = $request->telefone;
153         $telefone->save();
154
155         $user = User::where('userable_id', $enfermeiro->id)
156         ->where('userable_type', 'App\Models\Enfermeiro')
157         ->first();
158
159         $user->name = $request->nome;
160         $user->email = $request->email;
161         if($request->senha != null){
162
163             $regras = [
164                 'senha' => 'min:6',
165                 'confirmar_senha' => 'same:senha',
166             ];
167
```

```
168             $feedback = [
169                 'senha.min' => 'O campo senha deve ter no
    m nimo 6 caracteres',
170                 'confirmar_senha.same' => 'O campo confirmar
    senha deve ser igual ao campo senha',
171             ];
172
173             $request->validate($regras, $feedback);
174
175             $user->password = bcrypt($request->senha);
176
177         }
178         $user->save();
179
180
181         return redirect()->route('admin.enfermeiros.index')->
    with('msg', 'Enfermeiro atualizado com sucesso!');
182
183     }
184
185     public function destroy(Enfermeiro $enfermeiro){
186         $enfermeiro = Enfermeiro::withTrashed()->find(
    $enfermeiro->id);
187         File::delete(public_path('img/enfermeiros/' .
    $enfermeiro->foto));
188
189         $enfermeiro->forceDelete();
190         return redirect()->route('admin.enfermeiros.index')->
    with('msg', 'Enfermeiro exclu do com sucesso!');
191     }
192 }
```

Listing F.4 – Controlador responsavel por gerenciar enfermeiros

```php
namespace App\Http\Controllers;

use App\Models\Especialidade;
use Illuminate\Http\Request;

class EspecialidadeController extends Controller{
    public function index(){
        $especialidades = Especialidade::paginate(6);
        return view("admin.especialidades.index", compact("especialidades"));
    }
    public function medicoDaEspecialidade($idEspecialidade){
        $especialidade = Especialidade::find($idEspecialidade);
        $medicos = $especialidade->medicos;
        return response()->json($medicos);
    }
    public function store(Request $request){
        $regras = [
            'nome' => 'required|min:3'
        ];

        $feedback = [
            'required' => 'O campo :attribute    obrigat rio',
            'nome.min' => 'O campo nome deve ter no m nimo 3 caracteres'
        ];

        $request->validate($regras, $feedback);

        Especialidade::create($request->all());
        return redirect()->back()->with("msg", "Especialidade cadastrada com sucesso!");
    }
```

```
31      public function edit(Especialidade $especialidade){
32          $especialidade = Especialidade::find($especialidade->id
    );
33          return view("admin.especialidades.edit", compact("
    especialidade"));
34      }
35      public function update(Request $request, Especialidade
    $especialidade){
36          $especialidade = Especialidade::withTrashed()->find(
    $especialidade->id);
37          $especialidade->update($request->all());
38
39          return redirect()->route("admin.especialidades.index")
    ->with("msg", "Especialidade atualizada com sucesso!");
40      }
41      public function destroy(Especialidade $especialidade){
42          $especialidade = Especialidade::withTrashed()->find(
    $especialidade->id);
43          $especialidade->forceDelete();
44
45          return redirect()->back()->with("msg", "Especialidade
    exclu da com sucesso!");
46      }
47  }
```

Listing F.5 – Controlador responsavel por gerenciar especialidades médicas

```
1  namespace App\Http\Controllers;
2
3  use App\Models\Especialidade;
4  use App\Models\Medico;
5  use App\Models\Telefone;
6  use App\Models\User;
7  use Illuminate\Http\Request;
```

```
8  use Illuminate\Support\Facades\DB;
9  use Illuminate\Support\Facades\File;
10 use Illuminate\Support\Facades\Hash;
11
12 class MedicoController extends Controller{
13     public function index(){
14
15         if(request()->search){
16             $medicos = Medico::where('nome', 'like', '%'.
       request()->search.'%')
17             ->orWhere('crm', 'like', '%'.request()->search.'%')
18             ->paginate(6);
19         }else{
20             $medicos = Medico::paginate(6);
21         }
22
23         return view("admin.medicos.index", compact("medicos"));
24     }
25     public function create(){
26         $especialidades = Especialidade::all();
27         return view("admin.medicos.create", compact("
       especialidades"));
28     }
29
30     public function store(Request $request){
31         $regras = [
32             'nome' => 'required|min:3',
33             'cpf' => 'required|min:11',
34             'sexo' => 'required',
35             'estado_civil' => 'required',
36             'crm' => 'required|min:3',
37             'telefone' => 'required|min:3',
38             'email' => 'required|min:3',
```

```
39              'senha' => 'required|min:3',
40              'confirma_senha' => 'required|same:senha',
41          ];
42
43          $feedback = [
44              'required' => 'O campo :attribute    obrigat rio',
45              'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
46              'crm.min' => 'O campo crm deve ter no m nimo 3
    caracteres',
47              'telefone.min' => 'O campo telefone deve ter no
    m nimo 3 caracteres',
48              'email.min' => 'O campo email deve ter no m nimo 3
     caracteres',
49              'cpf.min' => 'O campo cpf deve ter no m nimo 11
    caracteres',
50              'sexo.required' => 'O campo sexo    obrigat rio',
51              'estado_civil.required' => 'O campo estado civil
     obrigat rio',
52              'senha.min' => 'O campo senha deve ter no m nimo 3
     caracteres',
53              'confirma_senha.same' => 'O campo confirmar senha
    deve ser igual ao campo senha',
54          ];
55
56          $request->validate($regras, $feedback);
57
58          $medico = new Medico();
59          $medico->nome = $request->nome;
60          $medico->cpf = $request->cpf;
61          $medico->rg = $request->rg;
62          $medico->email = $request->email;
63          $medico->endereco = $request->endereco;
```

```
64          $medico->cidade = $request->cidade;
65          $medico->estado = $request->estado;
66          $medico->cep = $request->cep;
67          $medico->data_nascimento = $request->data_nascimento;
68          $medico->sexo = $request->sexo;
69          $medico->estado_civil = $request->estado_civil;
70          $medico->crm = $request->crm;
71
72          if($request->hasFile('foto') && $request->file('foto')
    ->isValid()){
73              $requestImage = $request->foto;
74              $extension = $requestImage->extension();
75              $imageName = $requestImage->getClientOriginalName()
    ."."$extension;
76              $requestImage->move(public_path('img/medicos'),
    $imageName);
77              $medico->foto = $imageName;
78          }
79
80          $medico->save();
81
82          $telefone = new Telefone();
83          $telefone->numero = $request->telefone;
84          $telefone->medico_id = $medico->id;
85          $telefone->save();
86
87          if($request->especialidades){
88              foreach($request->especialidades as $especialidade){
89                  DB::table('medico_especialidades')->insert([
90                      'medico_id' => $medico->id,
91                      'especialidade_id' => $especialidade
92                  ]);
93              }
```

```
94          }
95
96          $user = new User();
97          $user->name = $request->nome;
98          $user->email = $request->email;
99          $user->password = Hash::make($request->senha);
100         $user->userable_id = $medico->id;
101         $user->userable_type = Medico::class;
102         $user->assignRole('medico');
103         $user->save();
104
105
106         return redirect()->route('admin.medicos.index');
107     }
108     public function edit(Medico $medico){
109         $medico = Medico::find($medico->id);
110         $telefone = Telefone::where('medico_id', $medico->id)->
    first();
111         $especialidades = Especialidade::all();
112         $especialidadesMedico = DB::table('
    medico_especialidades')
113         ->where('medico_id', $medico->id)
114         ->join('especialidades', 'especialidades.id', '=', '
    medico_especialidades.especialidade_id')
115         ->select('especialidades.*',"medico_especialidades.*")
116         ->get();
117
118         return view("admin.medicos.edit", compact("medico", "
    telefone", "especialidades", "especialidadesMedico"));
119     }
120     public function update(Request $request, Medico $medico){
121         $medico = Medico::withTrashed()->find($medico->id);
122         $medico->update($request->all());
```

```
123
124         if($request->hasFile('foto') && $request->file('foto')
    ->isValid()){
125             File::delete(public_path('img/medicos/'.$medico->
    foto));
126             $requestImage = $request->foto;
127             $extension = $requestImage->extension();
128             $imageName = $requestImage->getClientOriginalName()
    .strtotime("now").".".$extension;
129             $requestImage->move(public_path('img/medicos'),
    $imageName);
130             $medico->foto = $imageName;
131         }
132     }
133     public function destroy(Medico $medico){
134         File::delete(public_path('img/medicos/'.$medico->foto))
    ;
135
136         $medico = Medico::withTrashed()->find($medico->id);
137         $medico->forceDelete();
138     }
139 }
```

Listing F.6 – Controlador responsavel por gerenciar médicos

```
1 namespace App\Http\Controllers;
2
3 use App\Models\Medico;
4 use App\Models\MedicoUBS;
5 use App\Models\UnidadeBasicaDeSaude;
6 use Illuminate\Http\Request;
7
8 class MedicoUBSController extends Controller{
9     public function index(){
```

```
10      $medicosUbs = MedicoUBS::join("medicos","medicos.id","=","
    medico_u_b_s.medico_id")
11          ->join("unidade_basica_de_saudes","
    unidade_basica_de_saudes.id","=","medico_u_b_s.
    unidade_basica_de_saude_id")
12          ->select("medicos.nome as nome_medico","
    unidade_basica_de_saudes.nome as nome_ubs","medico_u_b_s.id
    AS id_medico_ubs")
13          ->paginate(6);
14          return view("admin.ubs.associar_medico.index", compact(
    "medicosUbs"));
15      }
16
17      public function create(){
18          $medicos = Medico::all();
19          $ubs = UnidadeBasicaDeSaude::all();
20
21          return view("admin.ubs.associar_medico.create", compact
    ("medicos", "ubs"));
22      }
23
24      public function store(Request $request){
25          $regras = [
26              'medico_id'=> 'required',
27              'unidade_basica_de_saude_id' => 'required'
28          ];
29
30          $feedback = [
31              'required' => 'O campo :attribute     obrigat rio'
32          ];
33
34          $request->validate($regras, $feedback);
35
```

```
36          MedicoUBS::create($request->all());
37
38          return redirect()->route('admin.medicosUBS.index')->
    with("msg", "M dico associado com sucesso!");
39      }
40
41      public function edit(MedicoUBS $medicoUBS){
42          $medicos = Medico::all();
43          $ubs = UnidadeBasicaDeSaude::all();
44
45          return view("admin.ubs.associar_medico.edit", compact("
    medicos", "ubs", "medicoUBS"));
46      }
47      public function update(Request $request, MedicoUBS
    $medicoUBS){
48          $regras = [
49              'medico_id'=> 'required',
50              'unidade_basica_de_saude_id' => 'required'
51          ];
52
53          $feedback = [
54              'required' => 'O campo :attribute     obrigat rio'
55          ];
56
57          $request->validate($regras, $feedback);
58
59          $medicoUBS->update($request->all());
60
61          return redirect()->route('admin.medicosUBS.index')->
    with("msg", "M dico associado com sucesso!");
62      }
63      public function destroy(MedicoUBS $medicoUBS){
64          $medicoUBS->delete();
```

```
65
66          return redirect()->route('admin.medicosUBS.index')->
    with("msg", "M dico desassociado com sucesso!");
67      }
68  }
```

Listing F.7 – Controlador responsavel por associar um médico a uma UBS

```
1  namespace App\Http\Controllers;
2
3  use App\Models\Paciente;
4  use App\Models\Telefone;
5  use App\Models\User;
6  use Illuminate\Http\Request;
7  use Illuminate\Support\Facades\File;
8
9  class PacienteController extends Controller{
10
11      public function index(){
12          $pacientes = Paciente::paginate(6);
13          return view("admin.pacientes.index", compact("pacientes
    "));
14      }
15      public function create(){
16          return view("admin.pacientes.create");
17      }
18      public function store(Request $request){
19          $regras = [
20              'nome' => 'required|min:3',
21              'cpf' => 'required|min:11',
22              'sexo' => 'required',
23              'estado_civil' => 'required',
24              'email' => 'required|min:3',
25              'senha' => 'required|min:3',
```

```
26              'confirma_senha' => 'required|same:senha',
27          ];
28
29          $feedback = [
30              'required' => 'O campo :attribute    obrigat rio',
31              'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
32              'email.min' => 'O campo email deve ter no m nimo 3
     caracteres',
33              'cpf.min' => 'O campo cpf deve ter no m nimo 11
    caracteres',
34              'sexo.required' => 'O campo sexo    obrigat rio',
35              'estado_civil.required' => 'O campo estado civil
     obrigat rio',
36              'senha.min' => 'O campo senha deve ter no m nimo 3
     caracteres',
37              'confirma_senha.required' => 'O campo confirma
    senha    obrigat rio',
38              'confirma_senha.same' => 'O campo confirma senha
    deve ser igual ao campo senha',
39          ];
40
41          $request->validate($regras, $feedback);
42
43          $paciente = new Paciente();
44          $paciente->nome = $request->nome;
45          $paciente->cpf = $request->cpf;
46          $paciente->sexo = $request->sexo;
47          $paciente->estado_civil = $request->estado_civil;
48          $paciente->email = $request->email;
49          $paciente->rg = $request->rg;
50          $paciente->endereco = $request->endereco;
51          $paciente->cidade = $request->cidade;
```

```
52          $paciente->estado = $request->estado;
53          $paciente->cep = $request->cep;
54          $paciente->data_nascimento = $request->data_nascimento;
55          $paciente->profissao = $request->profissao;
56
57          if($request->hasFile('foto') && $request->file('foto')
    ->isValid()){
58              $requestImage = $request->foto;
59              $extension = $requestImage->extension();
60              $imageName = md5($requestImage->
    getClientOriginalName() . strtotime("now")) . "." .
    $extension;
61              $requestImage->move(public_path('img/pacientes'),
    $imageName);
62              $paciente->foto = $imageName;
63              $paciente->save();
64          }
65
66          $telefone = new Telefone();
67          $telefone->numero = $request->telefone;
68          $telefone->paciente_id = $paciente->id;
69          $telefone->save();
70
71          $user = new User();
72          $user->name = $request->nome;
73          $user->email = $request->email;
74          $user->password = bcrypt($request->senha);
75          $user->userable_id = $paciente->id;
76          $user->userable_type = 'App\Models\Paciente';
77          $user->assignRole('paciente');
78          $user->save();
79
80          return redirect()->route('admin.pacientes.index');
```

```
81
82      }
83      public function edit(Paciente $paciente){
84          $paciente = Paciente::find($paciente->id);
85          $telefone = Telefone::where('paciente_id', $paciente->
    id)->first();
86          return view("admin.pacientes.edit", compact("paciente",
     "telefone"));
87      }
88      public function update(Request $request, Paciente $paciente
    ){
89          $regras = [
90              'nome' => 'required|min:3',
91              'cpf' => 'required|min:11',
92              'sexo' => 'required',
93              'estado_civil' => 'required',
94              'email' => 'required|min:3',
95          ];
96
97          $feedback = [
98              'required' => 'O campo :attribute    obrigat rio',
99              'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
100             'email.min' => 'O campo email deve ter no m nimo 3
     caracteres',
101             'cpf.min' => 'O campo cpf deve ter no m nimo 11
    caracteres',
102             'sexo.required' => 'O campo sexo    obrigat rio',
103             'estado_civil.required' => 'O campo estado civil
     obrigat rio',
104         ];
105
106         $request->validate($regras, $feedback);
```

```
107
108         if($request->senha){
109             $regras = [
110                 'senha' => 'required|min:3',
111                 'confirma_senha' => 'required|same:senha',
112             ];
113
114             $feedback = [
115                 'senha.min' => 'O campo senha deve ter no
    m nimo 3 caracteres',
116                 'confirma_senha.required' => 'O campo confirma
    senha    obrigat rio',
117                 'confirma_senha.same' => 'O campo confirma
    senha deve ser igual ao campo senha',
118             ];
119
120             $request->validate($regras, $feedback);
121
122             $user = User::where('userable_id', $paciente->id)
123             ->where("userable_type","App\Models\Paciente")
124             ->first();
125
126             $user->password = bcrypt($request->senha);
127             $user->save();
128         }
129
130         if($request->telefone){
131             $telefone = Telefone::where('paciente_id',
    $paciente->id)->first();
132             $telefone->numero = $request->telefone;
133             $telefone->save();
134         }
135
```

```
136         $paciente->update($request->all());
137
138         if($request->hasFile('foto') && $request->file('foto')
    ->isValid()){
139             File::delete(public_path('img/pacientes/' .
    $paciente->foto));
140             $requestImage = $request->foto;
141             $extension = $requestImage->extension();
142             $imageName = md5($requestImage->
    getClientOriginalName() . strtotime("now")) . "." .
    $extension;
143             $requestImage->move(public_path('img/pacientes'),
    $imageName);
144             $paciente->foto = $imageName;
145             $paciente->save();
146         }
147
148         return redirect()->route('admin.pacientes.index')->with
    ('msg', 'Paciente atualizado com sucesso!');
149      }
150
151      public function destroy(Paciente $paciente){
152         $paciente = Paciente::find($paciente->id);
153
154         if($paciente->foto){
155             File::delete(public_path('img/pacientes/' .
    $paciente->foto));
156         }
157
158         $user = User::where('userable_id', $paciente->id)
159         ->where("userable_type","App\Models\Paciente")
160         ->first();
161
```

```
162             $user->delete();
163
164             $paciente->delete();
165         }
166 }
```

Listing F.8 – Controlador responsavel por gerenciar pacientes

```
1  namespace App\Http\Controllers;
2
3  use App\Models\Especialidade;
4  use App\Models\MedicoUBS;
5  use App\Models\UnidadeBasicaDeSaude;
6  use Illuminate\Http\Request;
7
8  class UnidadeBasicaDeSaudeController extends Controller{
9      public function index(){
10         $ubs = UnidadeBasicaDeSaude::paginate(6);
11         return view("admin.ubs.index", compact("ubs"));
12     }
13     public function create(){
14         return view("admin.ubs.create");
15     }
16     public function store(Request $request){
17         $regras = [
18             'nome'=> 'required|min:3',
19             'endereco' => 'required|min:3',
20             'bairro' => 'required|min:3',
21             'cidade' => 'required|min:3',
22             'estado' => 'required|min:2',
23             'cep' => 'required|min:3'
24         ];
25
26         $feedback = [
```

```
27              'required' => 'O campo :attribute    obrigat rio',
28              'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
29              'endereco.min' => 'O campo endereco deve ter no
    m nimo 3 caracteres',
30              'bairro.min' => 'O campo bairro deve ter no m nimo
     3 caracteres',
31              'cidade.min' => 'O campo cidade deve ter no m nimo
     3 caracteres',
32              'estado.min' => 'O campo estado deve ter no m nimo
     3 caracteres',
33              'cep.min' => 'O campo cep deve ter no m nimo 3
    caracteres'
34          ];
35
36          $request->validate($regras, $feedback);
37
38          UnidadeBasicaDeSaude::create($request->all());
39
40          return redirect()->route('admin.unidadesBasicasDeSaude.
    index')->with("msg", "Unidade B sica de Sa de cadastrada
    com sucesso!");
41      }
42      public function edit(UnidadeBasicaDeSaude
    $unidadeBasicaDeSaude){
43          return view("admin.ubs.edit", compact("
    unidadeBasicaDeSaude"));
44      }
45      public function update(Request $request,
    UnidadeBasicaDeSaude $unidadeBasicaDeSaude){
46          $regras = [
47              'nome'=> 'required|min:3',
48              'endereco' => 'required|min:3',
```

```
49              'bairro' => 'required|min:3',
50              'cidade' => 'required|min:3',
51              'estado' => 'required|min:3',
52              'cep' => 'required|min:3'
53          ];
54
55          $feedback = [
56              'required' => 'O campo :attribute    obrigat rio',
57              'nome.min' => 'O campo nome deve ter no m nimo 3
    caracteres',
58              'endereco.min' => 'O campo endereco deve ter no
    m nimo 3 caracteres',
59              'bairro.min' => 'O campo bairro deve ter no m nimo
     3 caracteres',
60              'cidade.min' => 'O campo cidade deve ter no m nimo
     3 caracteres',
61              'estado.min' => 'O campo estado deve ter no m nimo
     3 caracteres',
62              'cep.min' => 'O campo cep deve ter no m nimo 3
    caracteres'
63          ];
64
65          $request->validate($regras, $feedback);
66
67          $unidadeBasicaDeSaude->update($request->all());
68      }
69      public function destroy(UnidadeBasicaDeSaude
    $unidadeBasicaDeSaude){
70          $unidadeBasicaDeSaude = UnidadeBasicaDeSaude::
    withTrashed()->find($unidadeBasicaDeSaude->id);
71          $unidadeBasicaDeSaude->forceDelete();
72      }
73      public function especialidadeMedicosDaUbs($idUBS){
```

```
74          $especialidades = Especialidade::join("
    medico_especialidades", "medico_especialidades.
    especialidade_id", "=", "especialidades.id")
75              ->join("medico_u_b_s", "medico_u_b_s.medico_id", "=
    ", "medico_especialidades.medico_id")
76              ->join("unidade_basica_de_saudes", "
    unidade_basica_de_saudes.id", "=", "medico_u_b_s.
    unidade_basica_de_saude_id")
77              ->where("unidade_basica_de_saudes.id", "=", $idUBS)
78              ->select("especialidades.*")
79              ->distinct()
80              ->get();
81
82          return response()->json($especialidades);
83      }
84
85  }
```

Listing F.9 – Controlador responsavel por gerenciar unidades basicas de saúde

O código completo do MVP desta aplicação pode ser encontrado no gitHub: https://github.com/emanooel/telemedicina$_{s}$*us*