Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

MARCOS GOMES NOGUEIRA JUNIOR

APLICATIVO PARA ANAMNESE MÉDICA UTILIZANDO SPRING BOOT

Assis/SP
2021

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

MARCOS GOMES NOGUEIRA JUNIOR

APLICATIVO PARA ANAMNESE MÉDICA UTILIZANDO SPRING BOOT

Trabalho de Conclusão de Curso de Análise e Desenvolvimento de Sistemas, do Instituto Municipal de Ensino Superior de Assis – IMESA e a Fundação Educacional do Município de Assis – FEMA, como requisito parcial à obtenção do Certificado de Conclusão.

Orientando: Marcos Gomes Nogueira Junior
Orientador: Dr. Almir Rogério Camolesi

Assis/SP
2021

FICHA CATALOGRÁFICA

N778a GOMES NOGUEIRA JUNIOR, Marcos.

Aplicativo para anamnese médica utilizando spring boot / Marcos Gomes Nogueira Junior. – Assis, 2021.

39 Páginas.

Trabalho de conclusão do curso (Análise e Desenvolvimento de Sistemas). – Fundação Educacional do Município de Assis - FEMA

Orientador: Dr. Almir Rogério Camolesi

1. Sistema Web. 2. Gestão médica.

CDD005.133

APLICATIVO PARA ANAMNESE MÉDICA UTILIZANDO SPRING BOOT

MARCOS GOMES NOGUEIRA JUNIOR

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso Superior de Análise e Desenvolvimento de Sistemas, avaliado pela seguinte comissão examinadora:

Orientador: _____
Dr. Almir Rogério Camolesi

Examinador: _____
Me. Guilherme de Cleva Farto

Assis/SP
2021

DEDICATÓRIA

Dedico este trabalho a minha mãe Marilene Viana Nogueira e meu pai Marcos Gomes Nogueira que sempre estão me dando forças para continuar apesar das dificuldades e sempre me incentivando, a Deus que vem me resguardando e me preparando para enfrentar as dificuldades. Também dedico ao meu orientador Almir Rogério Camolesi por confiar em meu potencial e me aceitar como seu orientado e me incentivar a mim e meus amigos a sempre continuar.

AGRADECIMENTOS

Agradeço a Deus por sempre me acompanhar, me auxiliando e dando capacidade para continuar e me possibilitar ter condições físicas e psicológicas para trabalhar e estudar e por toda sabedoria adquirida ao longo dos anos.

Agradeço aos meus pais Marilene Viana Nogueira e Marcos Gomes Nogueira por sempre me ajudar a alcançar meus objetivos, e por me derem privilégios que me auxiliaram na criação deste trabalho.

Agradeço aos meus amigos de trabalho e faculdade que me ajudaram neste desafio, pela confiança depositada em mim.

Agradeço por último e não menos importante ao meu orientador Dr. Almir Rogerio Camolesi por sempre disponibilizar um pouco do seu tempo para me ajuda a adquirir novos conhecimentos e por estar me acompanhando nesta difícil e estressante caminhada, e também sempre nos motivar a continuar e não desistir neste momento difícil que vemos tendo nos últimos anos.

“Você pode encontrar as coisas que perdeu, mas nunca as que abandonou.”

Gandalf

RESUMO

Os consultórios médicos atualmente precisam realizar suas atividades de maneira segura e com qualidade em suas atividades, e isto se tornou necessário devido ao avanço de uma pandemia que se iniciou no ano de 2020. A Transformação Digital está cada dia mais presente em nosso mundo e diversas tecnologias podem auxiliar nesta tarefa. Para este trabalho foi escolhida a tecnologia Java Spring Boot que permite junto com o banco de dados MongoDB junto ao React a realização do aplicativo de anamnese médica que tem como funcionalidade criar fichas de cadastro médicos de pacientes. O objetivo deste trabalho foi desenvolver de maneira simples e rápida com ênfase nas tecnologias descritas anteriormente um aplicativo para auxiliar os médicos a realizar a anamnese de seus pacientes e que as mesmas possam ser compartilhadas pelos profissionais que fizerem o uso deste aplicativo.

Palavras-chave: Java, Spring boot, Java script, UML, Diagrams, Visual Studio Code, React

ABSTRACT

Medical offices currently need to carry out their activities safely and with quality in their activities, and this has become necessary due to the advance of a pandemic that started in 2020. The Digital Transformation is increasingly gaining popularity our world, and several technologies can assist in this task. For this conclusion work, Java Spring Boot technology was chosen, which, together with the MySQL database with React, allows the realization of the medical anamnesis application that has the functionality to create patient medical records. The objective of this work is to develop, in a simple and fast way, with emphasis on the technologies described above, an application to help physicians perform the anamnesis of their patients and that they can be shared by professionals who use this application.

Keywords: Java, Spring boot, Java script, UML, Diagrams, Visual Studio Code, React

SUMÁRIO

1. INTRODUÇÃO

A Transformação digital que vem ocorrendo nos últimos anos faz com que a tecnologia evolua de maneira acelerada. Antes o que poderia ser feito de forma manuscrita, atualmente pode ser realizada a partir de diversas ferramentas digitais que nos auxiliam de maneira rápida e eficiente. Tanto pessoas como empresas têm visto a importância da tecnologia.

Com o avanço do Covid-19, muitas pessoas passaram a procurar serviços de maneira digital por aplicativos para sua maior segurança e com isso muitos consultórios e hospitais passaram a utilizar tecnologias em seus processos para aumentar o distanciamento social e evitar contato físico.

Em um desenvolvimento de um aplicativo médico as informações e colocação dos dados e muito importante, pois se utilizados da maneira certa, podemos tanto quanto facilitar no atendimento médico e até auxiliar a descobrir novas informações importantes do usuário para um possível pré-tratamento.

Segundo Friedman (2019) prever a evolução das tecnologias atuais é uma aposta mais segura do que seguir padrões mais antigos devido a velocidade que as tecnologias conseguem evoluir de maneira mais rápida do que humanos. Possibilitando uma base para utilizarmos tecnologias que estão mais solidificadas no mercado. Com isso em mente a utilização do Java com Spring Boot não e só uma tecnologia consolidada no mercado, mas com uma vasta comunidade onde possibilita otimização e uma grande fonte de informações permitindo uma fácil manutenção do código fonte.

1.1. OBJETIVO

O principal objetivo com o desenvolvimento desta pesquisa foi produzir uma aplicação web que permite manter as informações e fornecer meios que auxiliem o usuário na criação de fichas de anamnese medica. A aplicação funciona de forma semelhante a uma carteirinha médica e possibilita agilizar atendimentos, pois a aplicação propõe funcionalidades de inserção, remoção, exclusão e edição de dados a partir de uma maneira simplificada para

que não seja difícil de utilizar e muito menos demorado. Foi construída uma interface simples, fluida e sempre priorizando o seu desempenho. A ideia consiste em transformar um processo manual para o digital evitando haver problemas de grafia medica e acelerar o processo de anamnese e evitar contato físico.

1.2. JUSTIFICATIVA

Esta pesquisa cientifica justificou-se pela necessidade crescente em transformar digitalmente a maneira de manter produtividade em atividades que executadas de maneira manual e presenciais, pois, devido ao avanço da Covid-19 no ano de 2020 muitas atividades tiveram uma desaceleração ou até mesmo uma parada total para proteção das pessoas e dificultar o espalhamento do vírus. Por muitas pessoas possuírem a necessidade médica acompanhada e muitas vezes alguns dos processos criar uma aglomeração com a ficha de anamnese medica diminuiria o gargalo médico. Também evitaria deque um paciente e um funcionário entrem em contato físico evitando a contaminação.

1.3. MOTIVAÇÃO

Atualmente, as estruturas para gerenciamento de fichas de anamnese, envolve processos manuais que geram diversas anotações em próprio punho a fim de documentar os dados de cada paciente. No caso de um consultório médico que ao longo dos dias acaba recebe diversos pacientes, manter o ritmo de informações sobre tempo e demasiadamente exaustivo, e cobra das partes envolvidas, causando um desgaste físico e mental. E em certos casos a grafia do funcionário acaba sendo de difícil entendimento para o paciente e farmacêutico.

Todo o conhecimento adquirido no curso de análise e desenvolvimento de sistemas teve um papel primordial para a resolução do projeto, pois mesmo não utilizando as tecnologias utilizadas dentro do curso foi possível por meio de uma base solida identificar seus conceitos para sua manipulação.

1.4. PERSPECTIVA DE CONTRIBUIÇÃO

Com este aplicativo, apresenta-se como e possível gerar um sistema web profissional de fácil utilização usando Spring Boot e React com design elegante e para conclusão do curso de Tecnologia em Análise e Desenvolvimento de Sistemas.

Este trabalho também contribuiu para a comunidade os pontos positivos e negativos de todas as tecnologias utilizadas dentro deste projeto para que possibilite a outros que tenham acesso a este documento o utilizem como base para possíveis ponto de vista em execução de suas pesquisas utilizando as tecnologias apresentadas a fim de evitar desgastes desnecessários.

1.5. METODOLOGIA

Para o desenvolvimento do projeto foi utilizada a metodologia de trabalho experimental que tem como objetivo testar hipóteses de quem está pesquisando. Inicialmente será realizado o projeto do sistema utilizando-se de ferramenta Diagrams para a elaboração dos diagramas nos quais serão utilizados de base para o desenvolvimento. Para demonstrar foi realizado um estudo de caso que foi feito com base em um médico que precisa criar uma ficha de um paciente e gerar seu relatório. Para criação dos casos de uso, diagramas de classes, diagramas de atividades foi usado a linguagem UML com base no programa Diagrams, nele e possível realizar diversos trabalhos com uma agilidade devido a sua objetividade. Após esta tarefa foi realizada a implementação usando os conceitos de programação orientada a objetos e utilizando da ferramenta Intellij onde está possibilitou o desenvolvimento das funcionalidades utilizado Spring Boot que tem como sua linguagem Java. Para criação do front end foi utilizado React que é uma biblioteca de código aberto do Java Script que tem como foco criança interfaces de páginas web. O Banco de dados escolhido para suportar os dados foi o MySQL. Tal banco foi escolhido devido a possibilidade de uma manipulação mais simples que os bancos NOSQL.

1.6. PÚBLICO ALVO

O aplicativo que foi implementado neste trabalho tem como principal público médicos ou enfermeiros que necessite cadastrar ou atualizar ficha de pacientes de maneira rápida e fácil. Pensando em relação ao aplicativo que permite atender a qualquer pessoa que existe a necessidade de passar por consultas médicas.

1.7. ESTRUTURA DO TRABALHO

O trabalho se encontra dividido em alguns capítulos. O primeiro capítulo foi organizado em seções: introdução, objetivos, justificativas, motivação, perspectiva de contribuição, metodologia, público alvo e estrutura do trabalho

O segundo capitulo possui uma explicação sobre as tecnologias que foram utilizadas para o desenvolvimento deste trabalho. Já no Terceiro capitulo tem como foco em explicar as funcionalidades da arquitetura de web utilizando Spring boot

No quarto capitulo e apresentada a proposta do aplicativo, sua modelagem e explana-se sobre como funciona os principais diagramas de UML que foram utilizados para representar as funcionalidades, o banco de dados, a organização das classes e a arquitetura da aplicação

No quinto capitulo e exibido a conclusão do trabalho onde ira ser explanado o pontos durante o desenvolvimento do projeto e no capitulo seguinte trabalhos futuros onde será apresentado possíveis melhorias para o projeto final.

Por fim, o capitulo seis apresenta-se as referências usadas nesse trabalho.

2. ARQUITETURA MVC

A utilização da arquitetura MVC em diversos projetos vem se dado a necessidade de utilizar de maneira eficiente as tecnologias que nos são apresentadas. A arquitetura em desenvolvimento de software tem alta prioridade em sua organização pois ela definira diversos pontos do projeto em sua criação. Para melhor entendimento sobre a arquitetura MVC observe a imagem.

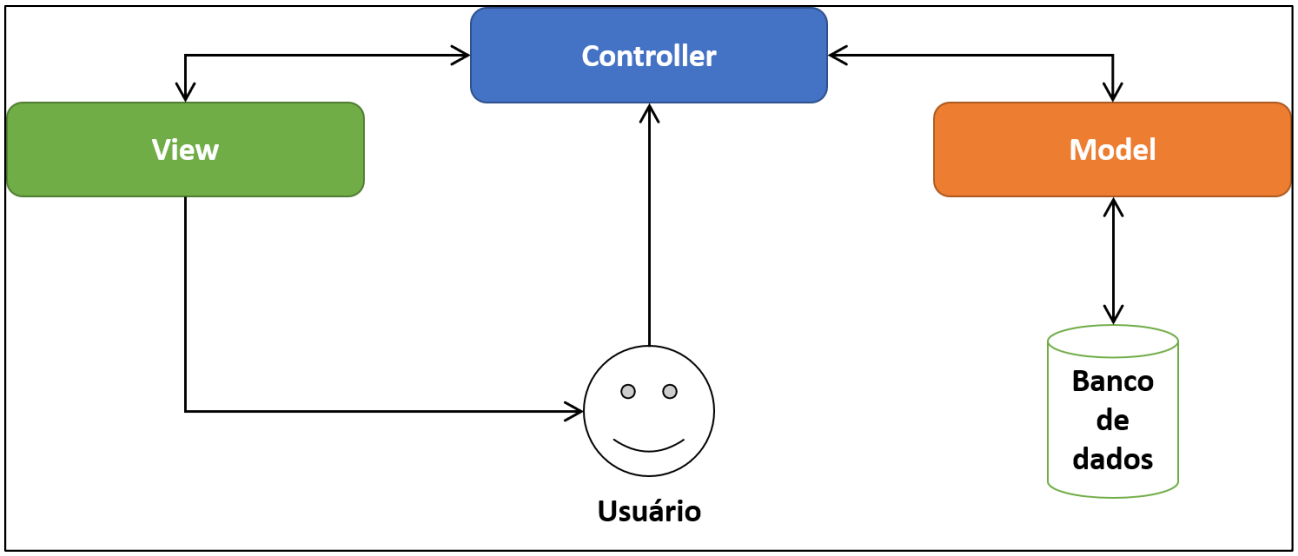

Figura 1: Diagrama de visão geral do sistema MVC

Após observar a Figura 1, verifica-se o uso de 3 camadas para a melhor organização e funcionamento de uma aplicação. Sendo a primeira desta a controller (azul) que tem como objetivo receber as entradas de dados enviadas pelo usuário. Tais entradas são mapeadas e enviadas para as camadas view (verde) e a model (laranja). Por sua vez a camada model tem como objetivo gerenciar um ou mais elementos e informar os status sobre seus elementos e os modifica. Já na camada view é onde ocorre o gerenciamento do *display* que é responsável por apresentar as informações para o usuário por meio de textos e gráficos.

Mas como e apresentado por Marylene(2020) devido a facilidade que o MVC apresenta ele passou a ser utilizado em diversas aplicações de desenvolvimento.

2.1. SISTEMA WEB

O sistema web são hospedados em um servidor tanto em uma empresa terceira como mesmo um servidor local onde qualquer usuário cadastro pode acessar suas ferramentas por meio de um navegador. Um dos motivos de utilização e tirar a necessidade de processamento da máquina física, pois, ao tirar o software da máquina do usuário você faz com que ele não precisa de uma infraestrutura muito grande possibilitando a facilidade no acesso e barateamento nos custos da empresa, somente necessitando de acesso à internet e a independência de plataforma de acesso fazendo com que não precise de uma configuração especifica na máquina está sendo utilizada.

De acordo com Espinha (2021) seguem alguns dos benefícios de utilizar um sistema web:

- **Otimização:**

Tarefas que possuem a necessidade de uma determinada atenção para sua execução, poder ser facilmente alteradas com um sistema Web, fazendo possibilitando a diminuição de tempo de suas execuções.

- **Economia:**

Os servidores por normalmente serem serviços que podem ser terceirizados os servidores em funcionamento em uma outra empresa especializada possibilita diminuir diversos custos para a empresa que está contratado tanto como gastos para infraestrutura onde os servidores ficariam como energia, espaço e profissionais qualificados para cuidarem do servidor durante 24h devido a possíveis problemas físicos em sua estrutura.

- **Modularidade:**

Um serviço Web possui uma modularidade muito grande podendo ser modificado de diversas maneiras para melhor atender as necessidades da empresa e até mesmo do cliente tornando se um único.

2.2. FERRAMENTAS UTILIZADAS PARA ANÁLISE

Neste tópico irá ser apresentado as ferramentas utilizadas para o método de análise do sistema, a fim de facilitar a implementação.

2.2.1. Diagrams

O diagrams e uma plataforma que possibilita o usuário a criar diversos diagramas para a criação de uma documentação e estruturação de seu projeto de software. E de forma gratuita disponibilizando a utilização de seu software para estudantes.

2.3. FERRAMENTAS UTILIZADAS O DESENVOLVIMENTO

Aqui serão explanadas as ferramentas que foram utilizadas para a implementação do software. Ferramentas que devido a sua vasta gama de utilização e aprendizado.

2.3.1. Intellij IDEA

O Intellij e um ambiente de desenvolvimento escrito em Java para desenvolvimento de software, onde a pertencente de seus direitos e JetBrains. Sendo muito utilizado para programações em Java ele também pode ser utilizado para programações de diversas linguagens tais como HTML, SQL, JPQL e Java Script. Segundo JETBRAINS (2021) diversas funcionalidades de sua IDE podem proporcionar para seu usuário um desenvolvimento mais dinâmico:

- a Geração automática de códigos, recurso que é encontrado em todas IDE’s atuais.
- o Aumento de produtividade pois acabam possuindo diversas ferramentas e plugins que auxiliam durante o desenvolvimento.

2.3.2. Spring Boot

Segundo Alexandre Afonso (2017) o Spring Boot veio com a finalidade de facilitar diversos processos de configuração e criação de aplicações com a intenção de agilizar o processo desenvolvimento, facilitando o processo de modificação de seus módulos para quais finalidades deseja utilizar como web, Persistência e segurança. Onde ele em alguns comandos irá ter sua configuração feita.

2.3.3. MySQL

O MySQL é um Sistema de Gerenciamento de Banco de Dados, que utiliza a linguagem SQL (System Query Language). Segundo TUTIDA (2021), pequenas empresas começaram aderir ao uso do SQL, pois possui servidor confiável, rápido e é de fácil manutenção. Utilizando-o não somente em pequenas aplicações, como também em aplicações críticas.

2.3.4. REACT

React e uma biblioteca conhecida em Java Script sendo usada para a criação de interfaces. Segundo Andrei L. (2019). A Escolha para utilização desta biblioteca se dá ao fato de possuir diversas opções de personalização e por também ser muito popular no mercado possibilitando conseguir achar diversos artigos e informações para o auxílio de manutenção dentro do código.

2.3.5. JAVA

Java é uma linguagem de programação orientada a objetos e desenvolvida pela Sun Microsystems na década de 90. Um dos pontos do Java e que os programas feitos não são executados em código nativos das plataformas e sim são compilados em uma máquina virtual. Um dos pontos que tornou Java popular e por se tratar de uma linguagem de multiplataformas possibilitando a sua maior dissipação pelos desenvolvedores. Segundo Sousa (2017) um dos motivos de Java ter ganhado espaço dentre outras linguagens foi a capacidade de se escrever um código para uma máquina e poder ser utilizado em um tipo

diferente de sistema. Resultando em utilização em diversos sistemas distintos dentre celulares e servidores.

2.3.6. Visual Studio Code

Ambiente de desenvolvimento no Visual Studio Code é um painel criativo que possibilita editar, depurar e compilar códigos. Um IDE (ambiente de desenvolvimento integrado) é um programa repleto de recursos que pode ser usado por muitos aspectos do desenvolvimento de software. Além das ferramentas padrões fornecidos pela maioria dos IDEs, o Visual Studio permite incluir compiladores, ferramentas de auxilio de código, e muitos outros recursos para auxiliar no desenvolvimento de software (MICROSOFT,2021)

2.3.7. Lombok

O Lombok segundo Malaquias (2018) é uma biblioteca Java que teve sua criação com foco em produtividade e redução de código. Utilizando se de anotações para informar nosso compilador. Criando assim uma maneira simples de diminuir a verbosidade do código tornando menos denso.

2.3.8. Spring Security

O Spring Security tem como foco em fornecer uma estrutura de autenticação em Java e dentre outros recursos de segurança para aplicativos. Segundo Afonso (2021) o Spring Security tem como foco tornar a parte de autenticação e autorização mais simples em ser elaborada. Onde possui uma vasta variedade de opções de configurações.

3. MODELAGEM DO APLICATIVO

Pensando na capacidade de facilitar o desenvolvimento do aplicativo, neste Capítulo será exibido ferramentas que são utilizadas nas disciplinas de engenharia de software que auxiliaram no entendimento geral do aplicativo. Para todos os exemplos foram usados o Diagrams que foi citado anteriormente.

3.1. DIAGRAMA DE CASO DE USO

O Diagrama de Caso de Uso é feito no início do projeto, na parte onde há um levantamento de requisitos e analise destes. Tendo em mente que este diagrama será usado durante o processo inteiro de criação do aplicativo. O objetivo deste diagrama e mostrar de forma simplificada para os desenvolvedores e usuários o que o aplicativo irá fazer, mesmo o usuário não tendo tanto conhecimento de desenvolvimento.

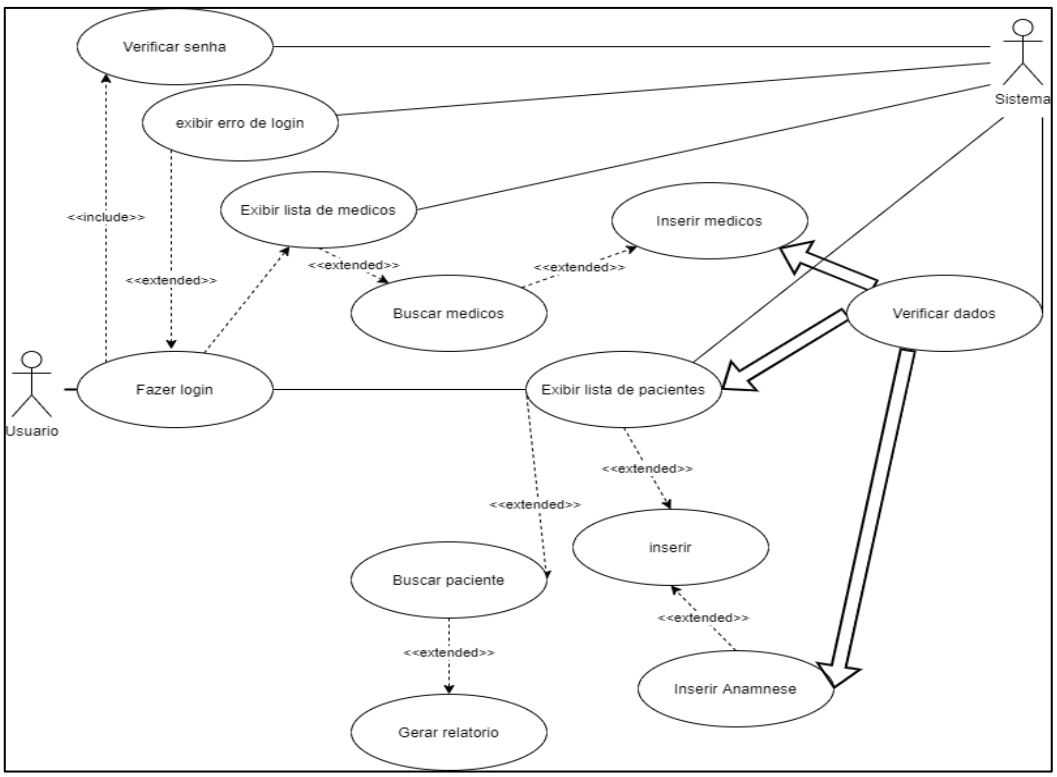

Figura 2: Diagrama de Caso de Uso - Geral

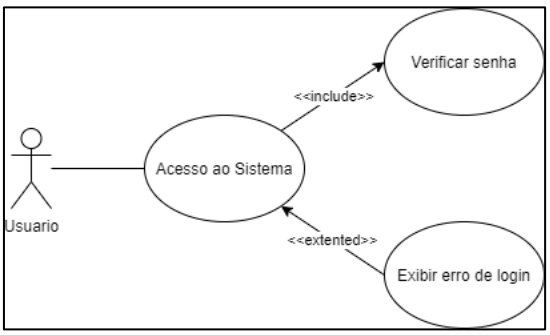

Figura 3: Diagrama de Caso de Uso - Acessar o Sistema

1. **Finalidade / Objetivo:** O usuário realiza o acesso com aplicativo.
2. **Ator:** Usuário.
3. **Evento inicial:** O ator insere as suas informações de acesso para entrar no aplicativo
4. **Fluxo principal:**
    a. O sistema oferece interface gráfica ao usuário para inserir suas informações;
    b. O usurário informa os dados para acessar;
    c. O usuário confirma os dados e seleciona a opção de Acesso ao Sistema(A1).
5. **Fluxo Alternativo:**

    A1 – O Usuário não tem cadastro

    a. O Sistema exibe que não existe informações cadastradas;
    b. O Usuário escolhe a opção cadastrar;
    c. O sistema exibe a interface para o usuário informar seus dados;
    d. O usuário informa seus dados de login e senha;
    e. O usuário seleciona a opção “cadastrar”
    f. O sistema salva as informações no banco de dados;
    g. O sistema exibe as interfaces de acesso ao sistema (login).

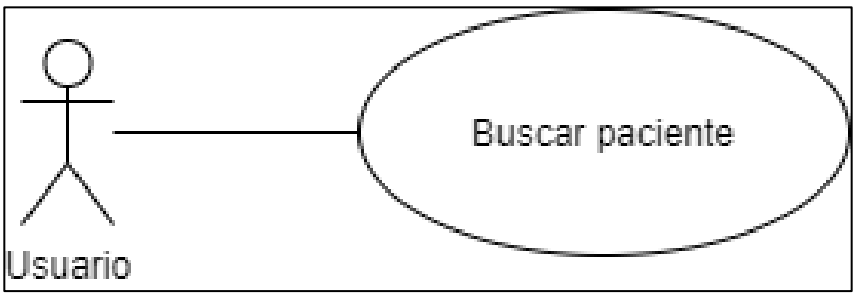

Figura 4: Diagrama de caso de uso – Buscar paciente

1. **Finalidade/Objetivo:** Permite ao usuário recuperar informações de um determinado paciente.
2. **Ator:** Usuário
3. **Evento inicial:** O ator informa no componente de busca, por exemplo, o CPF do paciente.
4. **Fluxo principal:**
    - **a.** O sistema oferece uma interface com um componente de busca.
    - **b.** O usuário informa o CPF do paciente;
    - **c.** O sistema busca no banco os dados;
    - **d.** O sistema exibe o paciente informado;

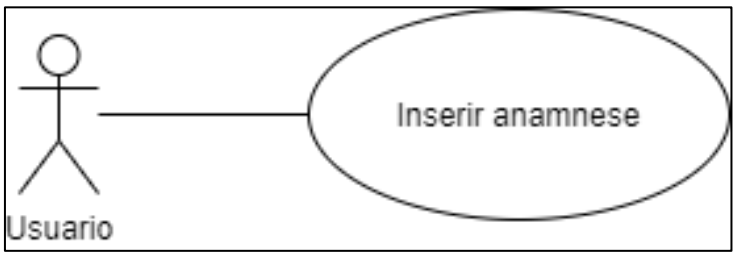

Figura 5: Diagrama de caso de uso - Criar Anamnese

1. **Finalidade/Objetivo:** O usuário consegue cadastrar os dados referentes a uma anamnese
2. **Ator:** Usuário
3. **Evento inicial:** O ator informa no componente os dados de anamnese.
4. **Fluxo principal:**
    - **a.** O sistema oferece uma interface com um componente de criação.
    - **b.** O usuário informa os dados da ficha de anamnese;
    - **c.** O sistema insere no banco os dados;
    - **d.** O sistema exibe a anamnese informada;

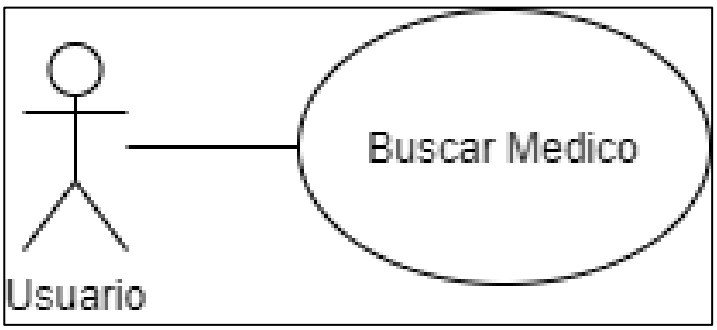

Figura 6: Diagrama de caso de uso - Buscar médico

1. **Finalidade/Objetivo:** Tal funcionalidade permite o usuário recuperar dados de um determinado médico.
2. **Ator:** Usuário
3. **Evento inicial:** O ator informa no componente de busca o CRM do médico.
4. **Fluxo principal:**
    - **a.** O sistema oferece uma interface com um componente de busca.
    - **b.** O usuário informa o CRM do médico;
    - **c.** O sistema busca no banco os dados;
    - **d.** O sistema exibe o médico informado;
    - **e.** O sistema oculta os outros médicos;

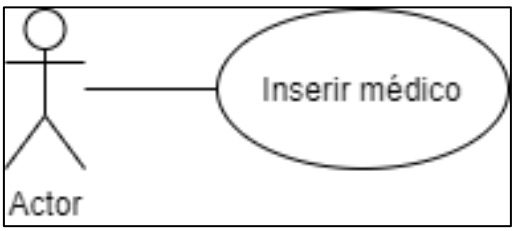

Figura 7: Diagrama de caso de uso - Criar Médico

1. **Finalidade/Objetivo:** O usuário consegue cadastrar os dados de um Médico.
2. **Ator:** Usuário
3. **Evento inicial:** O ator informa os dados do médico;
4. **Fluxo principal:**
    a. O sistema oferece uma interface com um componente para inserir os dados;
    b. O usuário informa os dados do médico;
    c. O sistema insere no banco os dados;
    d. O sistema exibe o médico criado;

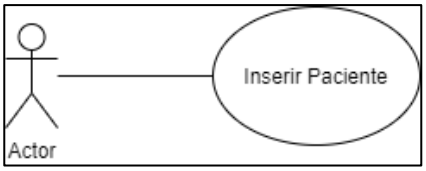

Figura 8: Diagrama de caso de uso - Criar Paciente

1. **Finalidade/Objetivo:** O usuário consegue criar um Paciente.
2. **Ator:** Usuário
3. **Evento inicial:** O ator informa os dados do paciente;
4. **Fluxo principal:**
    a. O sistema oferece uma interface para o usuário informar os dados de Paciente;
    b. O usuário informa os dados do paciente;
    c. O sistema insere no banco os dados;

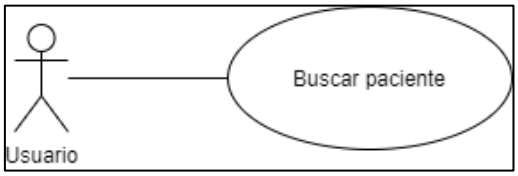

Figura 9: Diagrama de caso de uso – Buscar Paciente

1. **Finalidade/Objetivo:** O usuário consegue buscar os dados de um Paciente.
2. **Ator:** Usuário
3. **Evento inicial:** O ator informa os dados do paciente;
4. **Fluxo principal:**
    - **a.** O sistema oferece uma interface com um componente de busca;
    - **b.** O usuário informa os dados do paciente;
    - **c.** O sistema busca no banco os dados;
    - **d.** O sistema exibe o paciente procurado;

3.2. DIAGRAMA DE CLASSES

O diagrama de classe e um diagrama que tem como público alvo pessoas que estão participando do desenvolvimento do projeto, mas especifico os desenvolvedores, pois segundo Plínio Ventura (2018), o diagrama de classe e apresentação de uma estrutura de banco, ou seja, estrutura das classes interligadas como se fossem utilizadas no aplicativo. Tendo isso em mente facilita o desenvolvedor na preparação do desenvolvimento do aplicativo. Na imagem a seguir, estará sendo exibido o diagrama de classe que contém os principais atributos e métodos que será necessário em cada objeto para aplicação móvel.

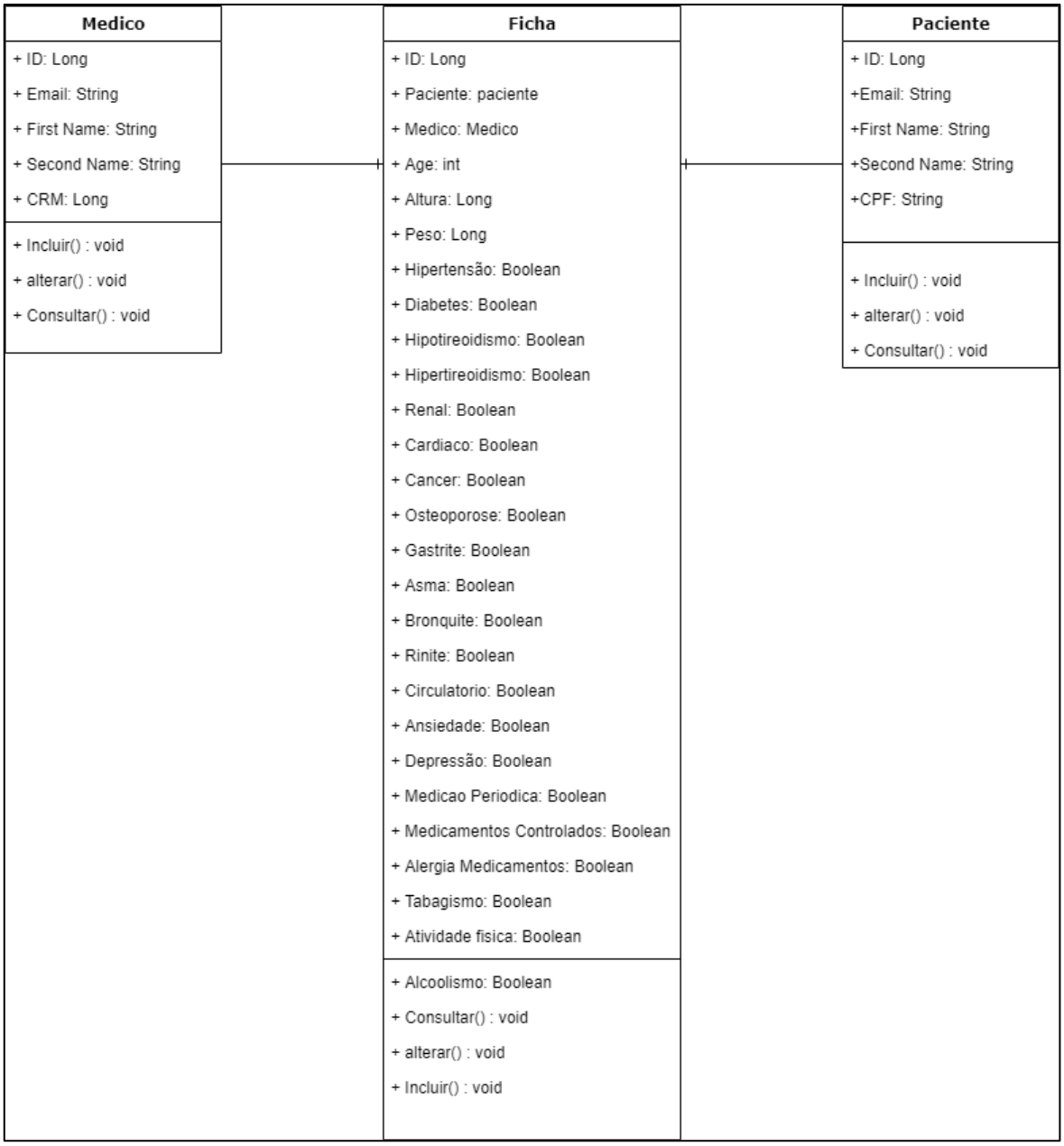

Figura 10: Diagrama de Classes da aplicação

3.3. DIAGRAMA DE ATIVIDADE

Para se comunicar dentro de uma empresa sobre o projeto e necessário mostrar a forma mais clara de como ele funciona, usam-se diagrama de atividades que segundo Lucidchart o foco principal é reunir desenvolvedores e pessoas da área de negócios para entender um processo específico, uma atividade.

Neste diagrama mostra a lógica utilizada no algoritmo, e processos das atividades, processo do negócio entre usuário e sistema e também o esclarecimento dos projetos. Na figura e mostrado o diagrama de atividade do projeto.

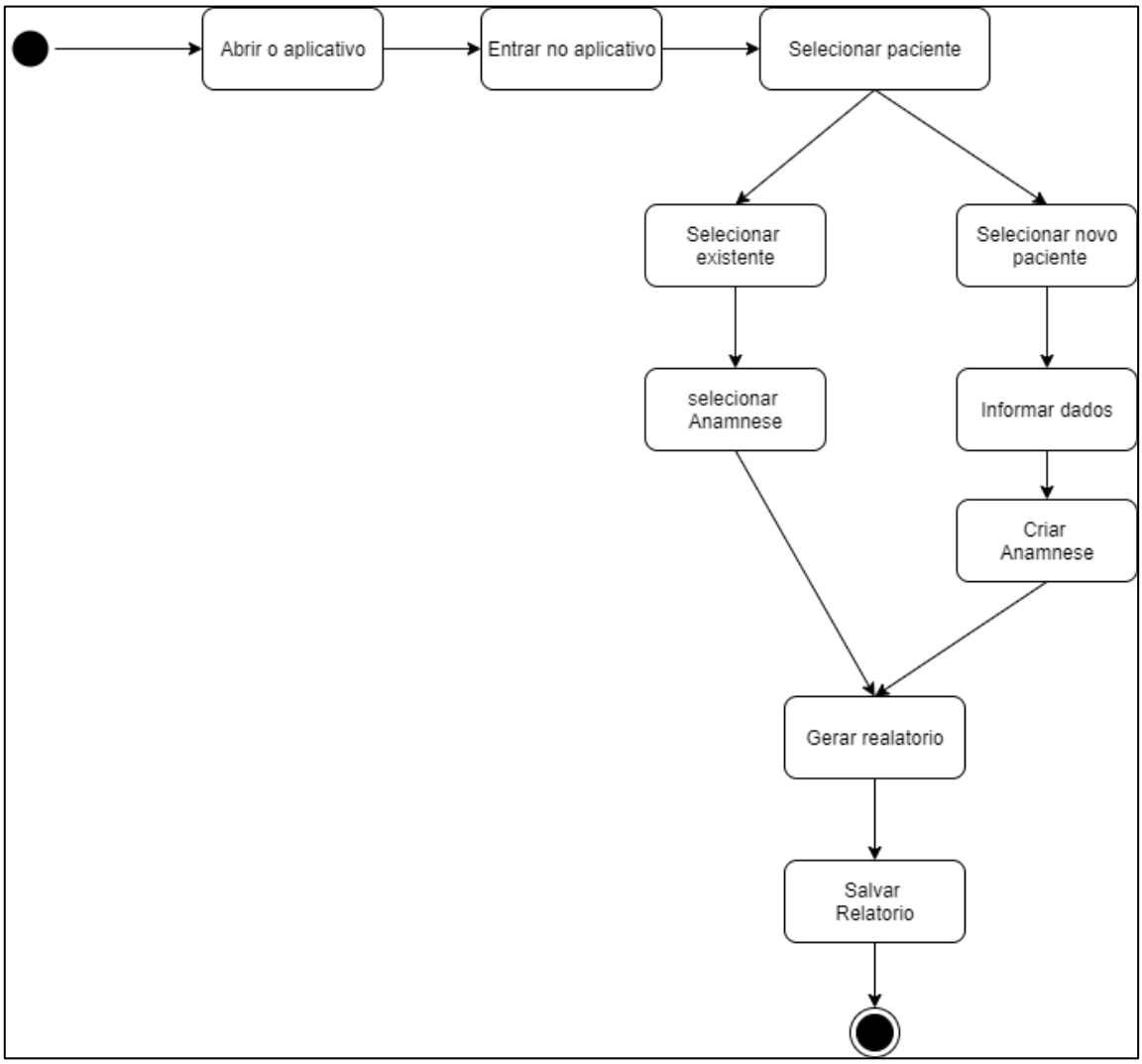

Figura 9: Diagrama de Atividade para inserir despesa ou receita

4. DESENVOLVIMENTO

Neste capitulo será esclarecido o desenvolvimento do trabalho.

4.1. DESENVOLVIMENTO BACK-END

O Desenvolvimento do *back-end* foi feito em Spring Boot com a utilização do Java como linguagem de programação. A finalidade seria que toda regra de negócio fique nessa camada, para o acoplamento, pois caso exista uma necessidade de utilizar outra tecnologia conjunta seja mais fácil, pois não existira a necessidade de escrever o código novamente.

Para que possa existir um melhor aproveitamento de código e de manutenção, o back-end foi separado nas camadas de config, domain, dto, exception, repositorye services.

Config: Nesta camada fica toda a configuração da aplicação, toda vez que a aplicação for iniciada, esta será a camada que primeiro irá ser chamada. Nesta camada poderá conter inserção de dados padrões e configurações de segurança de acesso. Para melhor entendimento desta camada, a Figura 10 demostra um exemplo de inserção de médico.

```
@Override
public void run(String... args) throws Exception {
    Medic medic = Medic.builder()
            .crm(58712L)
            .firstName("Jose")
            .secondName("Pedro")
            .email("joseomedico@medicos.com")
            .build();
    medicRepository.save(medic);
}
```

Figura 10: Inserção de médico

Domain: Será onde todas as classes modelos ou classes de domínios junto aos seus métodos. Para auxiliar no entendimento da camada segue a Figura 11.

```
@Getter
@Setter
@Builder
@Table (name = "medic")
public class Medic {
    @Id
    private Long id;
    private String email;
    private String firstName;
    private String secondName;
    private Long crm;
    }
```

Figura 13 - Ilustração da classe Medic da camada domain

DTO: Esta camada é a abreviação da palavra em inglês *Data Transfer Object* que significa em português objeto de acesso a dados. Ela tem como responsabilidade limitar os dados que serão retornados ao usuário. Podemos utilizar como exemplo uma classe de modelo Medico que tem atributos e-mail, nome, sobrenome e CRM, nesta camada *DTO* podemos criar classes que tenha todos atributos, sendo assim, quando for solicitado uma consulta a dados de Médicos, irá retornar o objeto de classe DTO. A Figura 12 da próxima página, demonstra a classe Medico e seus atributos.

```
@Getter
@Setter
public class MedicResponse {
    private Long id;
    private String firstName;
    private String secondName;
    private String email;
    private Long crm;
}
```

Figura 14 - Ilustração da classe MedicReponse

Repository: Essa camada é responsável por se comunicar com o banco de dados. Na Figura 13

```
@Repository("medicRepository")
public interface MedicRepository extends JpaRepository<Medic, Long> {

    List<Medic> findByCrm (String medic);
}
```

Figura 15 - Ilustração da classe MedicRepository

Services: A camada *services* tem como objetivo realizar todo o tratamento das chamadas de visualização, criação, edição e deleção. Podemos usar um exemplo de um objeto que os usuários solicitaram exclusão e o mesmo não há na base de dados, esta camada irá fazer uma análise se possui o objeto. Na figura a seguir demonstra a utilização da camada services com todas as suas funções.

```
public interface DoctorService {
    public void updateDoctor (MedicUpdateRequest medic);
    public List<Doctor> findAll ();
    public void save (Doctor doctor);
    void deleteById (Long doctorId);
    List<DoctorRecordResponse> findDoctorRecord ();
    Doctor findById (Long doctorId);
}
```

Figura 16 - Ilustração da camada de regra de negócio

Exception: Nesta camada e responsável para fazer o tratamento das exceções. Que ocorrem quando existe erros provenientes de lógica.

```
public class DoctorNotFoundException extends RuntimeException {
    public DoctorNotFoundException (Long id) {
        super ("Doctor not found with id: " + id);
    }
}
```

Figura 17 - Ilustração da camada de exception

4.2. DESENVOLVIMENTO FRONT-END

Para a elaboração do desenvolvimento front-end foi utilizado o *framework* React para o desenvolvimento que utiliza a linguagem de programação Java Script. O Foco e se comunicar com o back-end, para poder obter informações e retornar juntamente com a interface gráfica que a própria ferramenta fornece através de programação.

A separação de camadas foi mais simples contendo as pastas das interfaces e arquivos de serviço e componentes.

O arquivo de serviço tem como objetivo comunicar com a API para obter informações para devolver para o usuário através de métodos HTTP.

```
class MedicsService {
    getDoctor(){
        return axios.get(DOCTOR_API_BASE_URL);
    }
    getDoctorByCrm(id){
        return axios.get(`http://localhost:8080/medics/getAllDoctorByCrm/${id}`);
    }
    createDoctor(doctor){
        return axios.post(`http://localhost:8080/medics/regDoctor`, doctor);
    }
    getDoctorsById(medicsId){
        return axios.get(`http://localhost:8080/medics/findDoctors/${doctorsId}`);
    }
}
export default new DoctorsService()
```

Figura 18 - Ilustração da camada de services

4.3. INTERFACE GRAFICA

Com o objetivo de demonstrar melhor qual foi o resultado do aplicativo finalizado, nos próximos parágrafos há diversas imagens das interfaces gráficas do aplicativo.

Figura 19 – Interface gráfica (Pagina de listagem médica)

Na interface de listagem de médicos (Figura 19), seria onde o usuário poderia procurar ou verificar as informações apresentadas de cada médico e podendo seguir para deletar ou atualizar os dados do médico apresentados na interface do sistema médico.

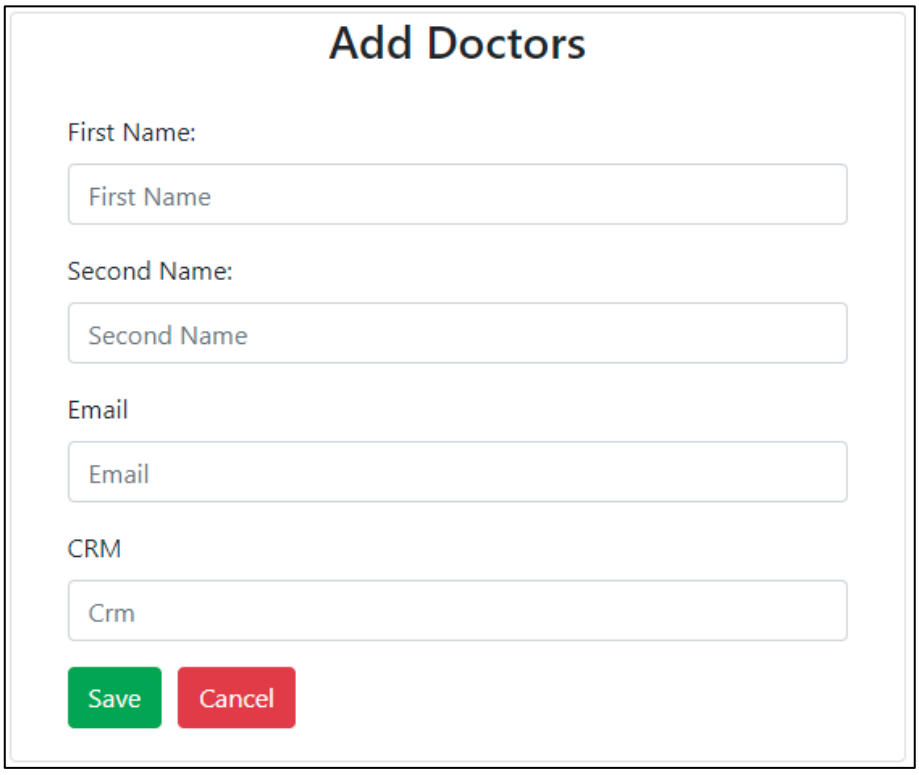

Figura 20 - Interface gráfica (Pagina de cadastro médico)

A interface de cadastro médico (Figura 20) é responsável por receber um novo médico, informando seus dados básicos de cadastro para apresentação no relatório.

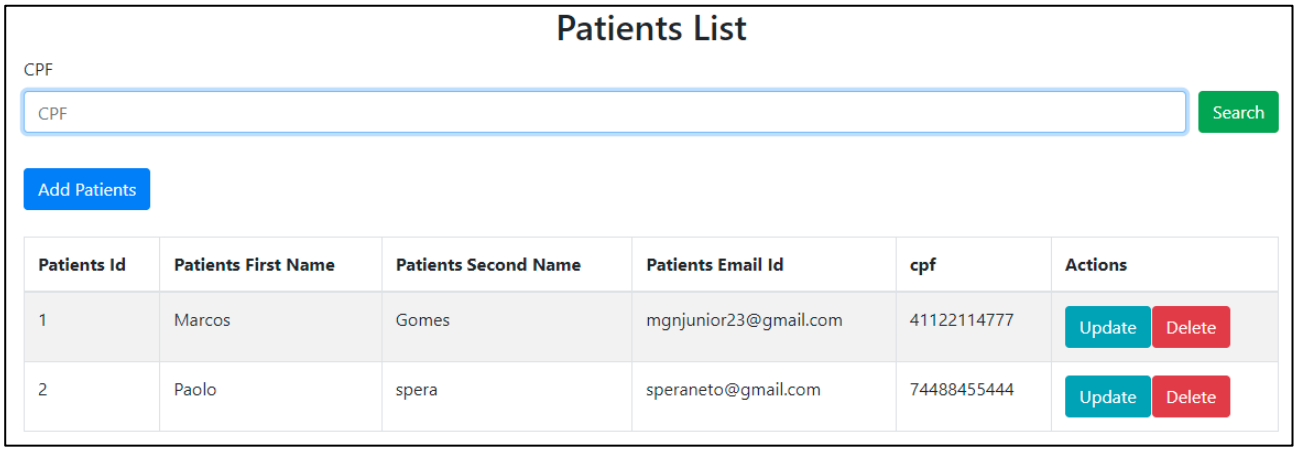

Figura 21 - Interface gráfica (Pagina de listagem paciente)

A interface de listagem de paciente (Figura 21 demonstra um cenário no qual o usuário poderia procurar ou verificar as informações apresentadas de cada paciente e pode escolher para remover ou atualizar os dados do paciente.

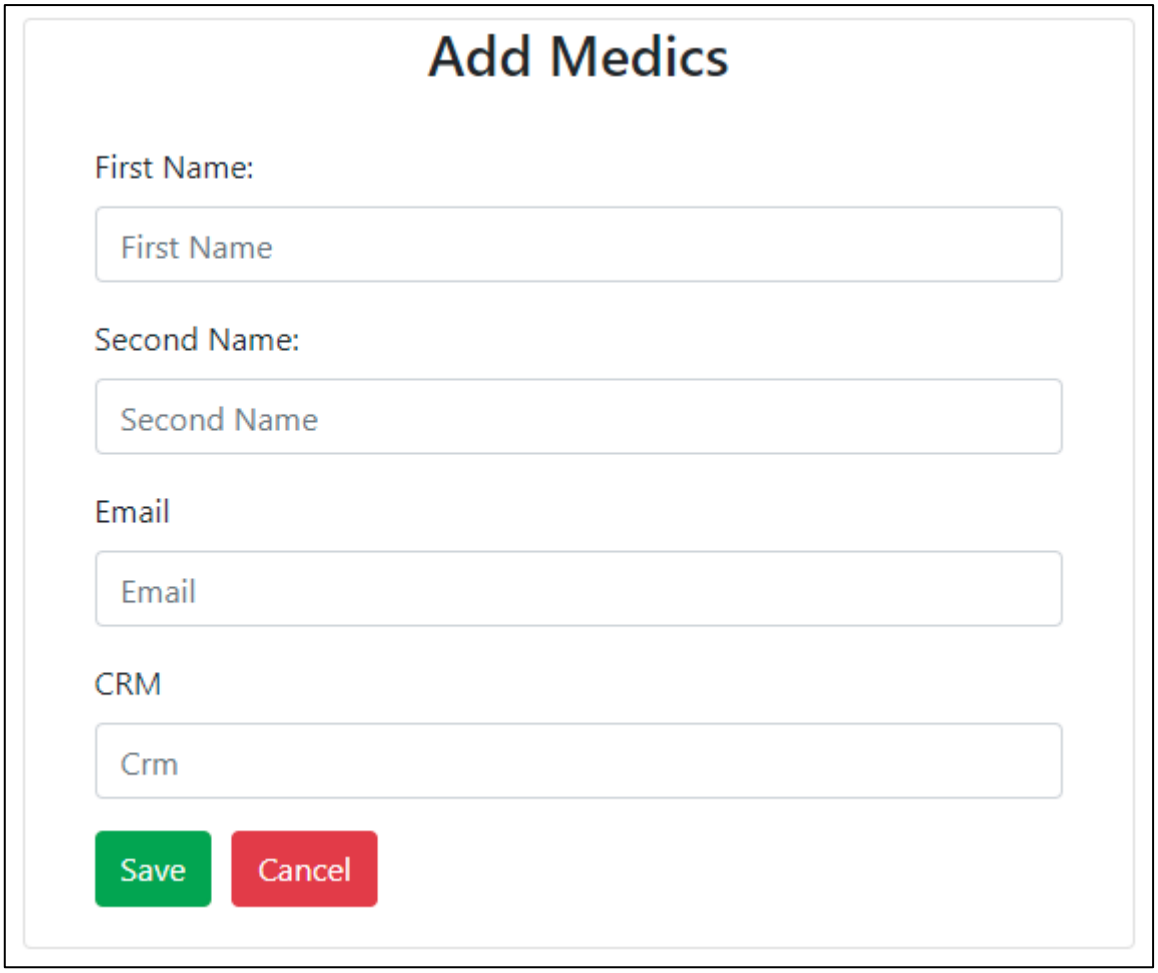

Figura 22 – Interface gráfica (Pagina de cadastro paciente)

A Figura 22 ilustra a interface de cadastro paciente responsável por manipular as informações de um novo paciente e gerar o fluxo de cadastrado de um novo paciente.

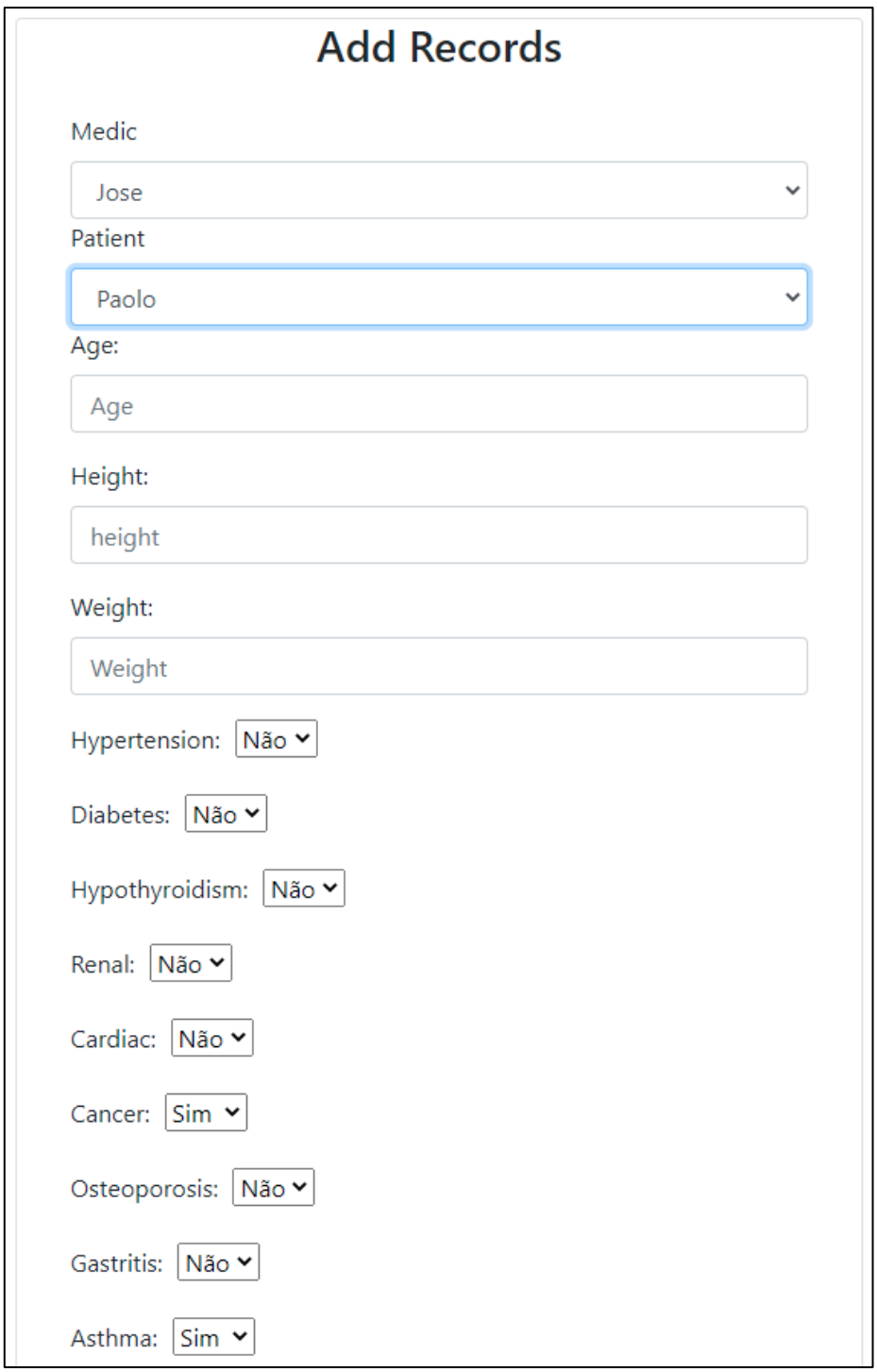

Figura 23 – Interface gráfica (Pagina de cadastro record)

Por fim, temos a interface de cadastro Record (Figura 23). Tal interface permite informar os ados e realizar o cadastro de um record, informando dados relacionados a saúde do paciente e informando paciente e medico para cadastro da ficha de anamnese.

5. CONCLUSÕES

Dentro da realização deste processo houveram adversidades em sua finalização, pois por se tratar de uma tecnologia nova no mercado e consequentemente não existem muitos exemplos em livros e artigos que pudessem agregar no desenvolvimento deste projeto utilizando React. Entretanto, uma boa parte dos objetivos que foram alcançados dentro deste desenvolvimento foram graças a conteúdos achados em inglês, pois proporcionaram uma vasta gama de conteúdo para aprendizado, que colaboraram para a conclusão deste trabalho.

Na parte de desenvolvimento utilizando Spring Boot teve poucas adversidades, devido ao fato de existir muito conteúdo tanto em inglês quanto em português. Podendo encontrar artigos, sites e livros sobre o conteúdo, onde proporcionaram um auxilio em seu desenvolvimento. O Lombok demonstrou sua devida importância para execução deste trabalho devido a sua capacidade de diminuir uma boa parte da verbosidade encontrada em programação na linguagem Java.

Com a finalização deste sistema, espera-se criar uma agilidade em todo o processo de anamnese médica. Visto que quando o processo e executado de maneira manual, o tempo para cadastro dos pacientes era demasiado longo. Além de auxiliar durante o processo gera um distanciamento entre funcionário e paciente. O sistema ainda e muito novo, e com isso existe diversas oportunidades de melhorias que podem ser implementadas que auxiliem ainda mais no gerenciamento médico.

5.1. TRABALHOS FUTUROS

Com o desenvolvimento pude adquirir experiência com este trabalho, com uma revisão da segurança dos dados. Existira a possibilidade de implementar mais funcionalidades, como gráficos, com objetivo de exibir informações de saúde medica dos pacientes, para que o paciente consiga visualizar seus dados via web. Previsão de saúde, através de aprendizado de máquina, de forma que, o cliente possa analisar condições físicas futuras.

6. REFERÊNCIAS

AFONSO, Alexandre. **O que é Spring Boot?.** Disponível em < https://blog.algaworks.com/spring-boot/ >. Acesso em: 14 mar 2020.

ESPINHA, Roberto. **9 Vantagens de utilizar um Sistema web de Gestão de Atividades.** Disponível em <https://artia.com/blog/9-vantagens-de-utilizar-um-sistema-web-de-gestao-de-atividades/>. Acesso em: 12 mar 2021.

JETBRAINS. **Plataforma IntelliJ.** Disponível em < https://www.jetbrains.com/pt-br/opensource/idea/> Acesso em: 12 mar 2021.

MALAQUIAS, Mateus. **Lombok: Escrevendo menos código em Java.** Disponível em <https://medium.com/collabcode/projeto-lombok-escrevendo-menos-c%C3%B3digo-em-java-8fc87b379209/>. Acesso em: 13 mar 2021

MICROSOFT. ***Getting Started*, 2021.** Disponível em < https://code.visualstudio.com/docs> acesso em: 15 mar. 2021.

SOUSA, Elisaudo. **Lombok: Descubra porque a linguagem de programação Java é a número 1 no mundo.** Disponível em <http://www.3way.com.br/descubra-porque-linguagem-de-programacao-java-e-numero-1-no-mundo/>. Acesso em: 01 Agos 2021

TUTIDA, Daniel. **Banco de dados para pequenas empresas e sua importância para os negócios.** Disponível em <https://encontreumnerd.com.br/blog/banco-de-dados-pequenas-empresas>. Acesso em: 12 mar 2021.

VENTURA, Plínio. **O Entendendo o Diagrama de Classes da UML.** Disponível em < https://www.ateomomento.com.br/uml-diagrama-de-classes/ >. Acesso em: 14 mar 2020.

VILLAS, Denis. **O são sistemas web?** Disponível em < http://logicalminds.com.br/o-que-sao-sistemas-web/ >. Acesso em: 19 julho 2021.