UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Guilherme de Souza Silva

Desenvolvimento de sistema para gerenciamento e histórico de fichas de pacientes em leitos de UTI

Uberlândia, Brasil

2021

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Guilherme de Souza Silva

Desenvolvimento de sistema para gerenciamento e histórico de fichas de pacientes em leitos de UTI

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Fabiano Azevedo Dorça

Universidade Federal de Uberlândia – UFU

Faculdade de Computação

Bacharelado em Ciência da Computação

Uberlândia, Brasil

2021

Guilherme de Souza Silva

Desenvolvimento de sistema para gerenciamento e histórico de fichas de pacientes em leitos de UTI

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Ciência da Computação.

Trabalho em construção. Uberlândia, Brasil, 18 de Junho de 2021:

Fabiano Azevedo Dorça
Orientador

Maurício Cunha Escarpinati

Maria Adriana Vidigal de Lima

Uberlândia, Brasil
2021

Dedico esse trabalho aos profissionais que acreditam num sistema médico eficiente e acima de tudo que trabalham arduamente para melhoria da área. Também dedico aos meus familiares que sempre me apoiaram nessa trajetória

Agradecimentos

Agradeço primariamente a minha família, que sempre me motivou a continuar meus estudos universitários.

Também agradeço meus amigos José Lemes Netto, Gabriel Franco e Raphael Cardoso, além de meu irmão Gustavo de Souza Silva, que me deram força quando estava para desistir dessa entrega, além de auxiliarem nas avaliações de meu código para assegurar a qualidade deles.

Agradeço também ao doutor Angelo Caetano Fernandes por fornecer os materiais necessários para construção das fichas do sistema.

Ao Prof. Dr. Fabiano Azevedo Dorça, por ter me orientado nesse projeto, e não ter desistido em momento algum, mesmo que eu tenha tido momentos sem aparecer devido ao trabalho.

Também agradeço à todos os professores da FACOM e de outras faculdades da UFU que me ajudaram a trilhar o caminho de cientista da computação.

Resumo

Este trabalho tem como objetivo o desenvolvimento de um sistema de gerenciamento de unidade de tratamentos intensivos de forma a facilitar o trabalho dos médicos que hoje preenchem grandes fichas à mão, consumindo não só uma grande quantidade de papel, como também muito tempo do corpo médico. Para atingir o objetivo proposto, foram feitas análises junto a um médico que expôs suas demandas para construção de um sistema que atendesse de forma correta suas necessidades diárias durante seus atendimentos. Devida a essas necessidades, o projeto foi constituído de uma aplicação web e de outra aplicação móvel, devida a necessidade dos dados só poderem ser coletados próximo ao leito do paciente e do fato de que ao mesmo tempo os médicos precisam de uma visão do paciente antes de ir até seu leito, principalmente nas trocas de turno. Foram utilizadas técnicas de engenharia de software como padrões de projeto MVC e Singleton e diagramas de casos de uso, além de técnicas para gestão de projeto baseadas na metodologia Ágil Kanban que contribuiu para a rápida finalização do sistema.Na seção de desenvolvimento deste trabalho é possível compreender melhor como as técnicas citadas acima foram aplicadas, bem como visualizar os diagrama de caso de uso e as telas que foram geradas a partir desses diagramas. Ao final, uma versão inicial do software foi construída para que a demanda inicial proposta seja atendida, deixando em aberto possibilidades de trabalhos e propostas futuras.

Palavras-chave: UTI, Sistema médico, gestão de hospitais, gerenciamento de UTI.

Lista de ilustrações

Lista de abreviaturas e siglas

UTI	Unidade de terapia intensiva
UX	User Experience - Experiência do usuário
TI	Tecnologia da Informação

Sumário

1 Introdução

O desenvolvimento de software está cada vez mais abrangente, hoje existem desde softwares para realizar pedidos de comida em casa, até outros que auxiliam no sono. Diversas empresas continuam, dia após dia, migrando seus sistemas do papel para modelos atuais de software, escolhendo sempre soluções em que o sistema mantenha uma alta disponibilidade e que facilite seu fluxo de trabalho durante os turnos.

A Covid-19 trouxe um aumento considerável no número de internações nas UTIS e de forma crescente, se vê necessário um sistema com a tecnologia mais nova e que atenda as demandas dos médicos.

Nesse trabalho, foi proposto o desenvolvimento completo de um sistema que facilita nos controles de entrada e saída das Unidades de terapia intensiva de hospitais, tal controle hoje é feito por via de fichas onde os médicos e enfermeiros manualmente fazem o preenchimento e estocam as fichas em armários.

Devido ao fato desse trabalho ser relacionado a um novo produto, não há exatamente uma pesquisa que tenha relação direta à ele, no entanto, o trabalho de Pereira et al. (2012) ajudou com suas justificativas, sobretudo nos pontos onde os autores descrevem as necessidades dos hospitais além das barreiras que são encontradas nesse mercado.

Uma citação bastante importante encontrado em Pereira et al. (2012) que ajuda a justificar a aplicação construída neste trabalho, é: "De acordo com estudos internacionais, estima-se que 46% do orçamento operacional de um hospital seja relacionado com logística: 27% em material e equipamentos e 19% em mão-de-obra. Cada melhoria que um sistema de informação trouxer para os processos logísticos, auxiliará na redução desses custos".

Com essa informação pode-se concluir que o sistema desenvolvido atende ao menos dois requisitos, que são o de economia de recursos, visto que há uma redução no consumo de papel e redução de gasto do time médico em preenchimento de fichas.

Uma fonte de informações e validações muito importante para o desenvolvimento desse sistema foi o Doutor Angelo Caetano Fernandes, que forneceu informações baseadas em seu dia a dia de trabalho em um hospital de Uberlândia. Ele se dispôs a ajudar na criação das fichas médicas, fornecendo os materiais necessários para criação do sistema de forma que atendesse as demandas corretamente do time médico.

Para a construção desse sistema serão utilizados conceitos de engenharia de software para que o desenvolvimento seja padronizado facilitando a manutenção e a futura evolução do sistema. Todo o sistema está sendo desenvolvido com base em estudos das fichas médicas e no entendimento do uso da aplicação por meio dos médicos.

Esse projeto contemplará uma aplicação para dispositivos móveis e uma aplicação para uso em navegadores de internet, cada uma dessas aplicações terá seus objetivos e funcionalidades descritas no desenvolvimento.

A seguir apresento os objetivos desse trabalho.

1.1 Objetivos

Os objetivos deste projeto são:

- Desenvolver um sistema de software para controle de pacientes nas UTIs;
- Utilizar conceitos de experiência da usuário *(user experience)* para que o software seja de fácil utilização;
- Utilizar na prática, técnicas de engenharia de software e padrões de projeto de forma a ter o sistema bem documento e construído;
- Construir um aplicativo móvel e web;
- Aplicar conhecimentos adquiridos no curso, tais como padrões de projeto, programação para internet, programação para dispositivos móveis e banco de dados;
- Realizar uma implantação de homologação para que o sistema possa ser testado e experimentado por médicos em condições reais e pelos avaliadores do trabalho.

1.2 Justificativa

Com o intenso uso das UTIs e com a necessidade de modernização das antigas fichas utilizadas até hoje, esse sistema se faz necessário. A ideia central também é facilitar o controle dos médicos de diferentes turnos dos pacientes em estado grave e que precisam de atenção especial, e assim trazendo melhorias significativas em seus tratamentos, devido a maior atenção dada a eles e não em extensas fichas preenchidas à mão.

1.3 Organização do Trabalho

Na seção atual foram apresentados, como forma de introdução, os objetivos e conceitos iniciais do trabalho.

No capítulo seguinte serão apresentados os referenciais teóricos e os conceitos utilizados para o desenvolvimento do sistema apresentado.

No capítulo 3 serão apresentados trabalhos correlatos, ou seja, sistemas já desenvolvidos para a área médica.

No capítulo 4 serão apresentados os métodos utilizados no desenvolvimento e todas as ferramentas que foram utilizadas.

No capítulo 5 será feita a conclusão e apresentação de futuras melhorias para o trabalho.

2 Referencial Teórico

Apresento neste capítulo conceitos das tecnologias adotadas para elaboração do projeto, além de uma breve discussão sobre a introdução de softwares para gestão de hospitais

2.1 Visão dos sistemas de gestão hospitalares

Hoje em um mundo informatizado, a tecnologia da informação é uma aliada poderosa no quesito salvar vidas, porém deve ser sempre utilizada com o máximo de cuidado, segundo Pereira et al. (2012), a TI e seus sistemas informatizados são essenciais na saúde.

Esses sistemas auxiliam não só na competitividade empresarial em saúde, mas na melhoria do atendimento para a comunidade em um serviço de necessidade básica. Porém, os sistemas precisam garantir a integridade das informações mantidas e fornecidas por eles, a fim de evitar consequências graves, como processos judiciais ou indução ao erro médico.

É vital que um sistema informatizado apresente informações: precisas, completas, em tempo real e útil.

Sabendo dessas informações, Pereira et al. (2012) também diz que hospitais estão investindo fortemente em sistemas de gestão com diversos objetivos, e, este trabalho visa então a gestão eficiente dos leitos de UTI.

Um ponto extremamente importante sobre o desenvolvimento de software com foco na saúde é que por muitas vezes essas aplicações são abrangentes demais, e tendem a não serem bem aceitas pelos usuários finais (médicos/enfermeiros).

Como também dito por Pereira et al. (2012), em uma pesquisa realizada em 2009, município de São Paulo, com o objetivo de verificar como o sistema de informação foi concebido, implantado, operado e mantido para suporte a gestão pública da saúde bucal local, observou-se que dependendo do nível dos colaboradores na empresa, as posturas e percepções são diferentes:

- O nível estratégico (direção) acredita na melhoria do sistema;
- O nível tático (gerência) tem postura positiva, apesar de apresentar desânimo;
- O nível operacional (usuário final) vê as modificações como imposição e só as acata pelo compromisso e obediência.

Por esse motivo, o desenvolvimento desse software se focou nos seguintes princípios da metodologia Ágil:

- Software funcional acima de documentação detalhada;
- Colaboração com os médicos acima de negociação de contratos;
- Ser compreensivo à mudanças.

Dessa forma foi possível desenvolver um software que atenda todos os níveis de colaboradores, tendo um produto final entregue mais rápido e sempre aberto à melhorias.

2.2 Conceitos Utilizados

Nessa seção, serão apresentados conceitos utilizados no desenvolvimento do sistema.

2.2.1 Metodologia Ágil

As metodologias ágil são um conjunto de técnicas adotadas para melhoria do fluxo do desenvolvimento de software, são todas baseadas no Manifesto Ágil, a metodologia utilizada para esse projeto foi a Kanban devido a necessidade de respostas rápidas à mudanças do projeto.

Segundo RADIGAN (2021), "A metodologia Kanban se baseia na plena transparência do trabalho e na comunicação em tempo real da capacidade, portanto, o quadro do Kanban deve ser visto como a única fonte de verdade para o trabalho da equipe."

E os principais pontos de vantagem dessa metodologia são:

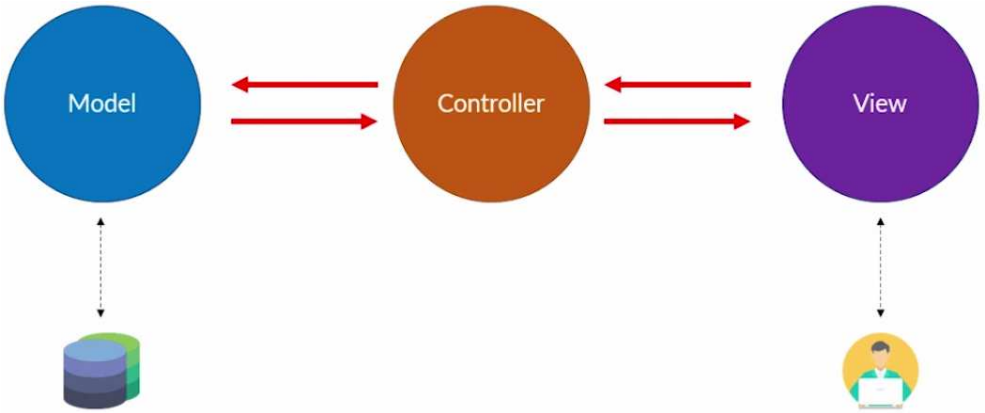
- Uma equipe Kanban concentra-se apenas no trabalho ativo em andamento;
- Limitação de gargalos através de limitação dos trabalhos em andamento;
- Métricas visuais através do quadro;
- E a a entrega contínua (CD) é a prática de lançar trabalhos com frequência para clientes.

2.2.2 Padrões de Projeto

Serão descritos dois padrões de projetos utilizados neste trabalho, sendo eles MVC e o Singleton.

O Padrão de projeto MVC é um acrônimo para *Model View Controller*, é um padrão de projeto onde se separa o software em 3 camadas, como exemplifica a figura 1.

Figura 1 – Diagrama padrão MVC



Fonte: Devmedia ()

A camada *model* onde os modelos e manipulações de dados do banco de dados acontecem, a camada *view* que exibe as informações trazidas da *model*, e a camada *controller*, responsável por fazer a interação entre a model e a view, além das tratativas necessárias para exibição somente dos dados solicitados.

O Padrão de projeto Singleton, é utilizado para garantir a existência de apenas uma instância de uma classe específica, além de criar um ponto de acesso global a essa instância da classe.

2.2.3 Frontend e Backend

As definições de *frontend* e *backend* muitas vezes tendem a confundir as pessoas, são duas áreas de atuação como desenvolver bastante diferentes, mas que hoje em dia com tecnologias como React e NodeJs ganharam maior nível de comunicação.

CRUZ (2017) em tradução de Robbins (2012), define bem a diferença entre *backend* e *frontend*. Ela afirma, em tradução aberta que "Frontend se refere a qualquer aspecto do processo de design que aparece ou se relaciona diretamente ao navegador" e cita algumas atividades, como por exemplo, documento HTML, desenvolvimento do estilo, código Javascript e design de informações em relação à experiência do usuário no site.

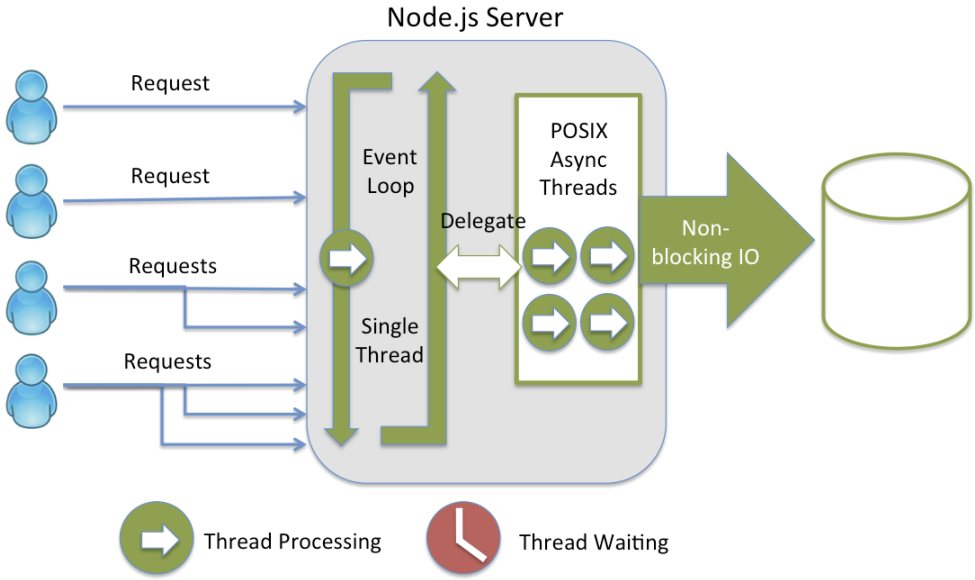
Robbins (2012) relata que o *backend* concentra as atividades que funcionam no servidor, deixando as páginas dinâmicas e interativas. Ela, ainda, cita algumas ações do *backend* tais como organizar a informação no servidor, processar formulários, organizar o banco de dados e utilizar linguagens *server-side*.

2.2.4 NodeJs e Express

Para o *backend* da aplicação foi utilizado o conjunto de duas ferramentas extremamente importantes, a descrição simplificada de cada uma delas vem a seguir. MDS (2021) define NodeJs como um ambiente em tempo de execução de código aberto e multiplataforma que permite aos desenvolvedores criarem todo tipo de aplicativos e ferramentas do lado servidor *(backend)* em JavaScript. Node é usado fora do contexto de um navegador (ou seja executado diretamente no computador ou no servidor).

Uma das funcionalidades mais interessantes do NodeJs é sua alta disponibilidade, segundo JÚNIOR (2015) o Node não trata a requisição como um todo para que ele possa estar disponível para tratar uma nova requisição do cliente, ele simplesmente a recebe, repassa para *threadpool* do servidor e já está disponível novamente para atender outra requisição do cliente, assim como é descrito pela Figura 2

Figura 2 – Funcionamento do NodeJs



Fonte: JÚNIOR (2015)

Segundo MDS (2021) Express é o *framework* Node mais popular e a biblioteca subjacente para uma série de outros *frameworks* do Node. O Express oferece soluções para:

- Gerenciar requisições de diferentes verbos HTTP em diferentes URLs.
- Integrar *"view engines"* para inserir dados nos *templates*.
- Definir as configurações comuns da aplicação web, como a porta a ser usada para conexão e a localização dos modelos que são usados para renderizar a resposta.

- Adicionar novos processos de requisição por meio de *"middleware"* em qualquer ponto da "fila"de requisições.

Além do Express e suas ferramentas já existentes, foi desenvolvido um *middleware* simples para gerenciamento do *token* de sessão do usuário, de forma a permitir o acesso apenas de usuários devidamente autorizados ao sistema e aos serviços disponíveis publicamente.

2.2.5 Banco de dados

No desenvolvimento de aplicações existem diversos mecanismos para armazenamento de dados, neste trabalho foi realizado a utilização de dois banco de dados extremamente diferentes, o MongoDB que utiliza a abordagem *noSQL*, não possuindo então, qualquer linguagem estrutural e PostgreSQL que é fortemente estruturado.

MongoDB é um banco de dados *noSQL* baseado em documentos, o que quer dizer que toda informação é armazenada em documentos JSON como o abaixo na figura 3 MongoDB (2021).

Figura 3 – Exemplo de JSON

Fonte: Site MongoDB

Em conjunto com MongoDB, para que as operações realizadas no banco fossem feitas de forma facilitada e simplificada, foi utilizado o *framework* Mongoose que provê soluções baseadas em esquemas para criação dos modelos da aplicação.

O banco de dados relacional utilizado na aplicação foi o PostgreSQL, como já citado. PostgreSQL, como definido em seu site PostgreSQL (2021), em tradução livre, como "Um banco de dados relacional poderoso com mais de 30 anos de desenvolvimento o que traz forte reputação de estabilidade, robustez e desempenho para ele."

Juntamente ao PostgreSQL, foi utilizado a ferramenta Sequelize, que facilitou na criação dos modelos relacionais e nas consultas SQL.

2.2.6 HTML, CSS, Javascript e Typescript

O HTML ou linguagem de marcação de texto, é o bloco mais simples de um sistema web, ele basicamente define a semântica do que está escrito na tela, e é lido nativamente pelos navegadores web.

O CSS ou Folhas de Estilo em Cascata é utilizado para construir o visual em cima do HTML, é ele que da a aparência aos sistemas web.

Javascript e Typescript são duas linguagens de programação utilizadas em sistema web para criação de comportamentos e funcionalidades, tendo como única diferença entre elas o fato de que Typescript é uma linguagem tipada.

2.2.7 React e React-Native

React e o React-Native são ferramentas desenvolvidas pelo Facebook para desenvolvimento de aplicações web e para dispostivos móveis de alto desempenho.

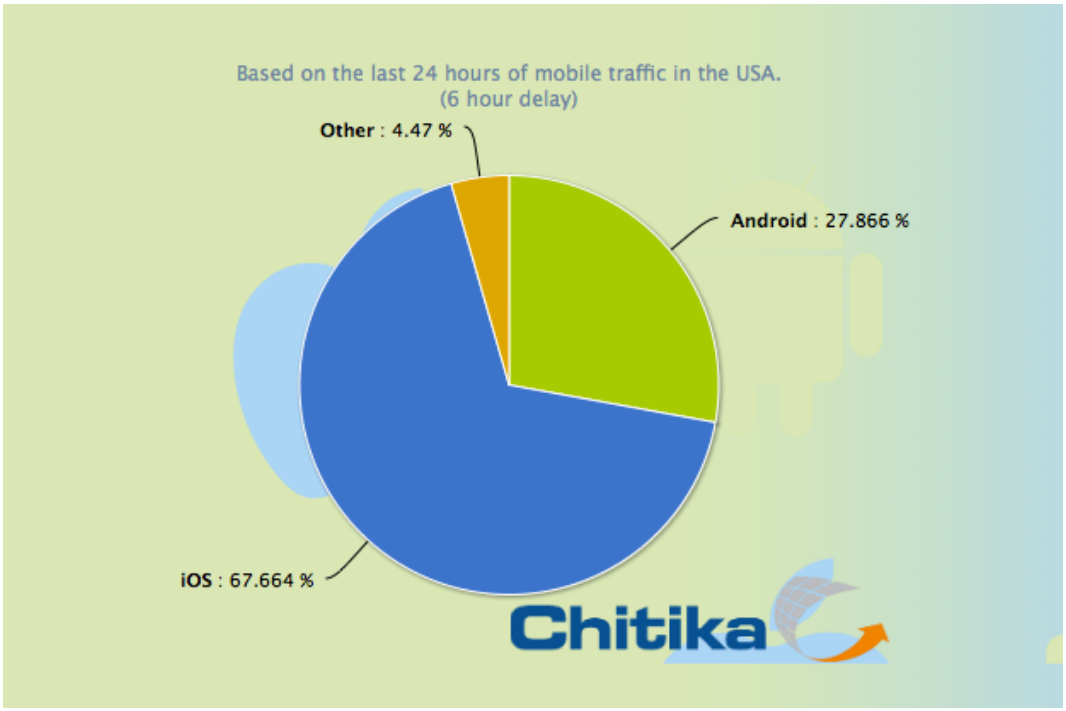
Como dito por Roveda (2021), "React JS é uma biblioteca Javascript para a criação de interfaces de usuário — ou *UI (user interface)*. Criado em 2011 pelo time do Facebook, o React surgiu com o objetivo de otimizar a atualização e a sincronização de atividades simultâneas no *feed* de notícias da rede social, entre eles chat, status, listagem de contatos e outros", "seu funcionamento acontece através do que chamamos de componentes".

React tem sido muito utilizado devido a alto índice de reuso de código por ser baseado na criação e composição de componentes (pequenos módulos de tela, como por exemplo um botão), além da alta performance de renderização das telas.

Diferentemente do modelo padrão de modificação direta ao documento da página, o React faz isso de forma virtual, e se a ferramenta entender que não é necessário, ela não gera uma nova versão da tela ou de uma parte específica dela React (2021).

O React-Native é uma ferramenta exclusiva para desenvolvimento para dispositivos móveis híbrido. ela utiliza o React como base e é capaz de converter a aplicações para uso em dispositivos Android e iOS que hoje são líderes absolutos no mercado *smartphones*, como o gráfico da Figura 4 sugere.

Figura 4 – Mercado de dispositivos móveis em 2012



Fonte: Lunden (2012)

3 Desenvolvimento

Nesta seção será descrito como foram realizados o planejamento e o desenvolvimento das aplicações móvel e web, além do *backend* do sistema.

3.1 Dispositivos Móveis e Web

O sistema construído apresenta duas aplicações distintas, uma móvel e uma navegadores *web*, essa construção foi necessária pois, após uma análise junto ao médico que auxiliou na elaboração do sistema, foi identificado que era necessária uma ferramenta em que o médico pudesse fazer inserções sobre o paciente ainda no leito (móvel), e uma versão mais detalhada para lançamento dos diagnósticos (*web*).

3.2 Metodologia de Desenvolvimento

Conforme já citado no capítulo 2 desse trabalho, o Kanban foi utilizado como metodologia ágil de desenvolvimento, facilitando assim o modelo onde se é possível desenvolver um sistema extremamente mutável para atender os clientes da melhor maneira possível.

O Kanban é uma metologia ágil baseada em um sistema visual, onde tudo o que se precisa fazer para o momento fica exposto em um ambiente para que todos os envolvidos tenham visão das necessidades e do andamento das tarefas atuais do projeto. Essas necessidades tem escopo totalmente maleável e que se adapta ao momento do projeto

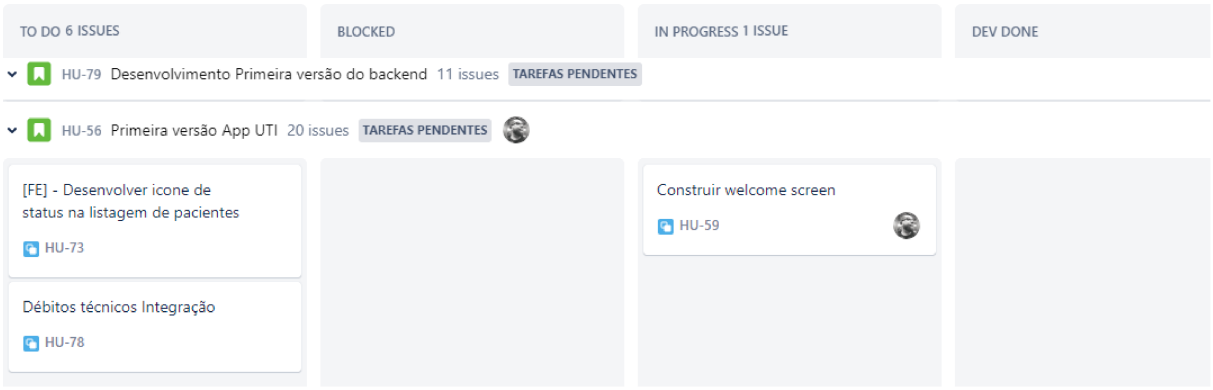
Para agilizar nas entregas do projeto, muitas funcionalidades do sistema que estavam já no quadro Kanban, acabavam retornando para o *backlog* (local onde as tarefas são armazenadas para o desenvolvimento futuro), isso acontecia para que funcionalidades primárias do sistema fossem atendidas antes de outras menos necessárias.

Para esse projeto, foi utilizado o ambiente do site Jira, sistema focado em gestão e gerencia de projetos (Figura 5), onde um quadro Kanban foi criado com as colunas:

- À fazer: tarefas que devem ser desenvolvidas;
- Bloqueados: tarefas que não podem ser executados pois dependem de outras tarefas;
- Em progresso: tarefas que estão sendo desenvolvidas;
- Desenvolvimento finalizado: tarefas que já foram finalizadas, e aguardam por avaliação de código;

- Em avaliação: tarefas que estão em avaliação de código;
- Em teste: tarefas que estão em avaliação de qualidade da funcionalidade;
- Prontas: tarefas que já passaram por todo o fluxo e já estão no ambiente de homologação ou produção.

Figura 5 – Exemplo do quadro Kanban do projeto



Fonte: Do autor

3.3 Requisitos do sistema

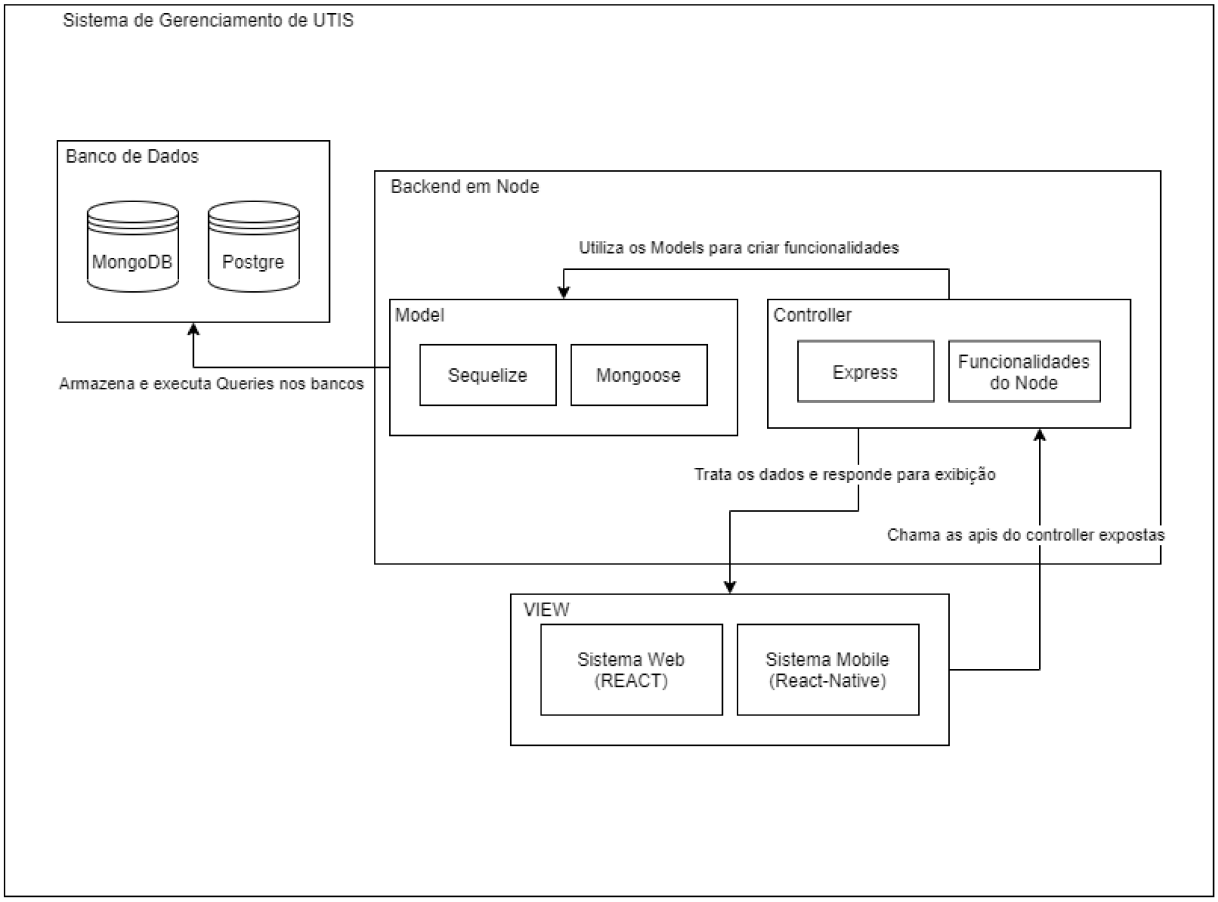
Todos os requisitos do sistema foram colhidos em reuniões informais com o médico já citado anteriormente, poucas informações foram documentadas. Dentre os pontos importantes colhidos e documentados posso destacar, os dados necessários para construção das fichas, a melhor forma para exibição da ficha médica e os tipos de usuários necessários para o sistema.

3.4 Padrões de Projeto

Como citado no capítulo 2, foram utilizados 2 padrões de projeto principais para construção do sistema,

Na definição arquitetural, o sistema se utiliza do padrão MVC, sendo assim, as camadas foram definidas como representando pela Figura 6 a seguir.

Figura 6 – Camadas do MVC



Fonte: Do autor

O uso desse padrão de projeto facilitou na criação das comunicações entre o *backend* e o *frontend*, além de criar uma organização onde se tornou mais fácil dar manutenção.

Já o padrão Singleton foi utilizado para facilitar o acesso as funcionalidades das ferramentas no Nodejs, como sempre se utiliza somente uma instância de cada classe esse padrão facilitou bastante no desenvolvimento, sendo aplicado em Javascript, ele se parece com algo como a seguir:

```
class Database {
  constructor() {
    this.init();
  }

  init() {
    this.connection = new Sequelize(config);
    models
      .map((model) => model.init(this.connection))
      .map(
        (model) => model.associate
```

```
            && model.associate(this.connection.models)
        );
    }
}
//Faz com que seja retornada a mesma instancia
export default new Database();
```

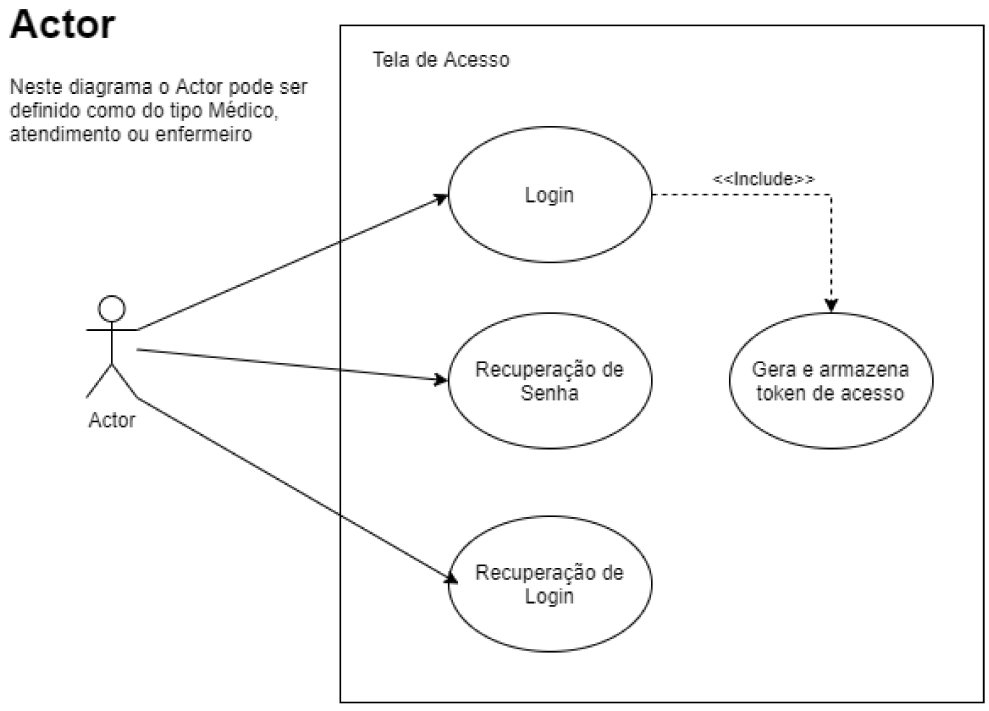
O arquivo acima é importado diretamente no arquivo de entrada, sendo assim, a instância de Database é única e gerada somente no início da aplicação, mas utilizada por toda ela.

3.5 Casos de uso de suas telas

Serão descritos os principais casos de uso do sistema e as fotos da implementação destes via navegador e aplicativo.

Acesso ao sistema: funcionalidades iniciais do sistema, como fazer o login e alterar a senha do usuário. esses casos estão disponíveis para qualquer usuário cadastrado pelo administrador como a Figura 7 demonstra.

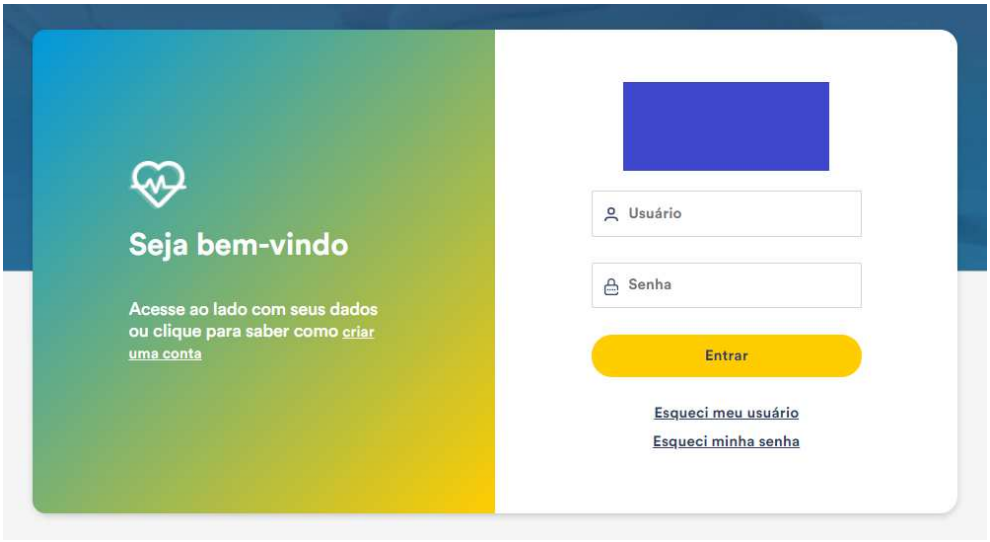
Figura 7 – Caso de uso de acesso - Diagrama



Fonte: Do autor

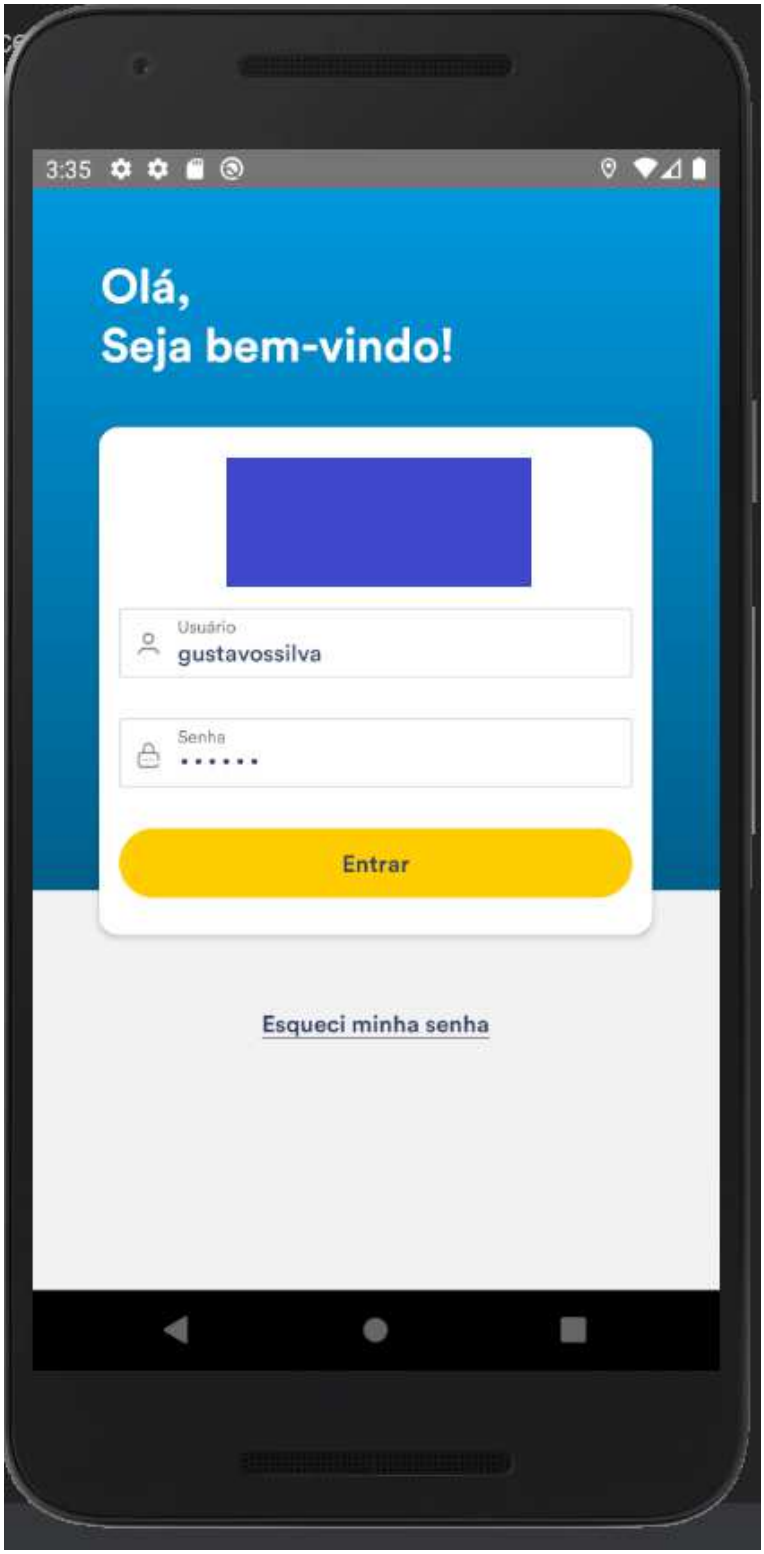
Cada uma dessas funcionalidades pode ser visualizada nas Figuras 8 e 9 abaixo, que são fotos tiradas do sistema final.

Figura 8 – Caso de uso de acesso - Representação no sistema web



Fonte: Do autor

Figura 9 – Caso de uso de acesso - Representação no móvel

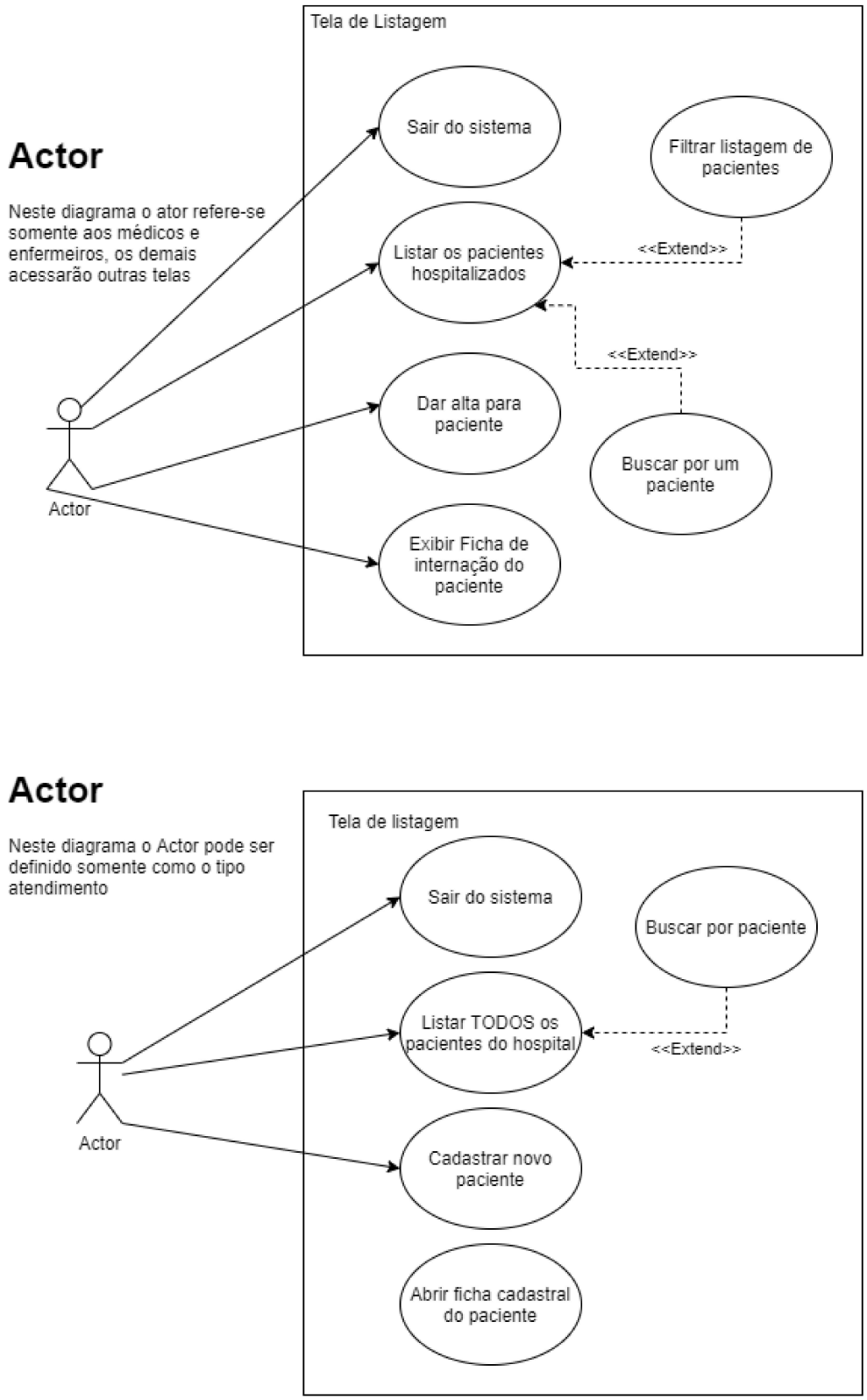


Fonte: Do autor

Listagem de pacientes: Consiste em todas as funcionalidades da tela de exibição dos pacientes, desde abrir mais informações dos pacientes, filtros até executar tarefas como liberar o paciente (dar alta).

Esse caso de uso se divide em duas possibilidades, a visão da equipe médica e a visão da equipe de atendimento. sua representação encontra-se na Figura 10

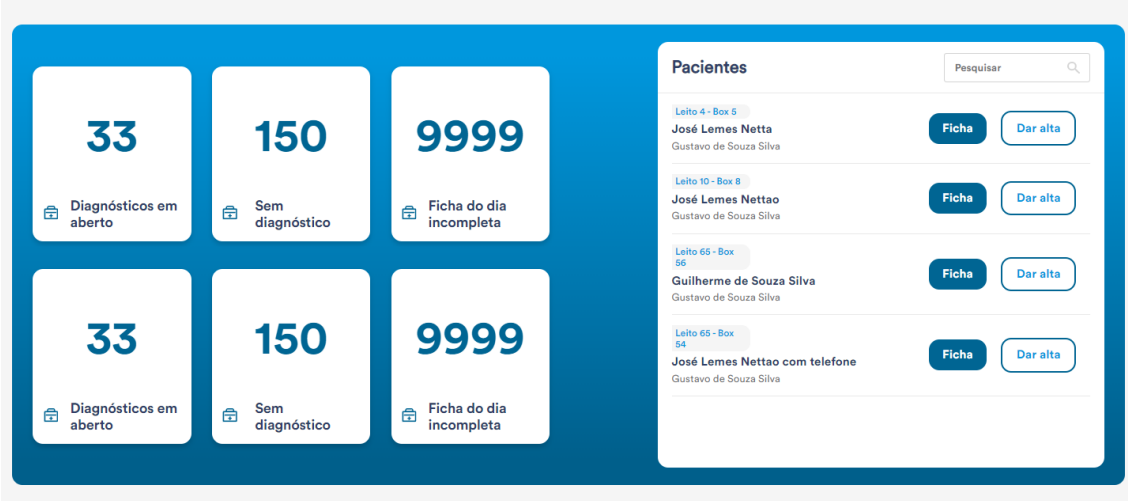
Figura 10 – Caso de uso de listagem - Diagramas



Fonte: Do autor

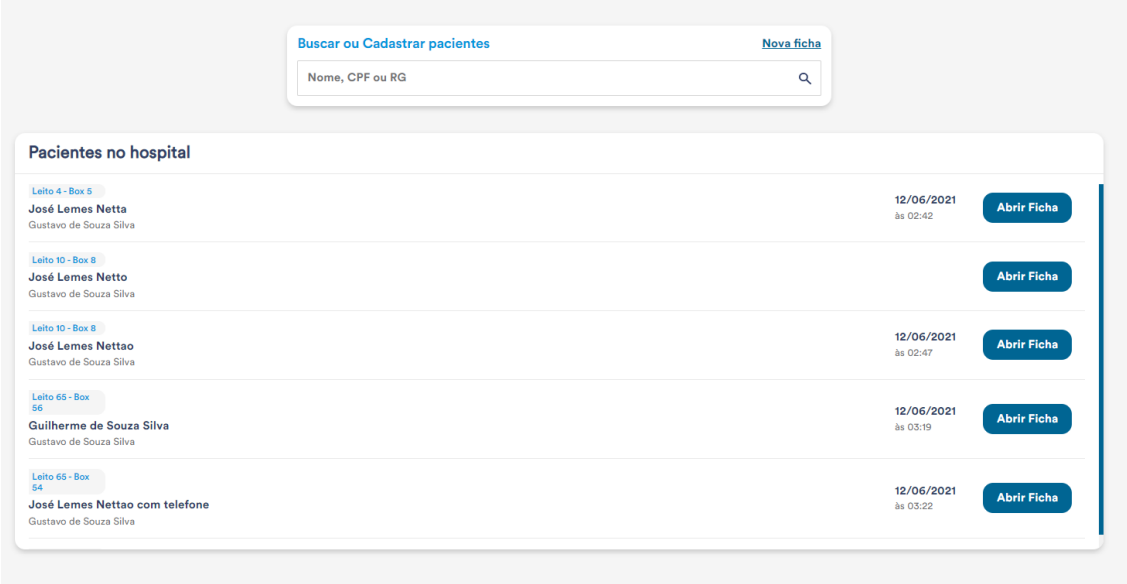
Cada uma dessas funcionalidades pode ser visualizada nas Figuras 11, 12 e 13 que são fotos do sistema.

Figura 11 – Caso de uso de listagem - Representação no sistema web dos médicos



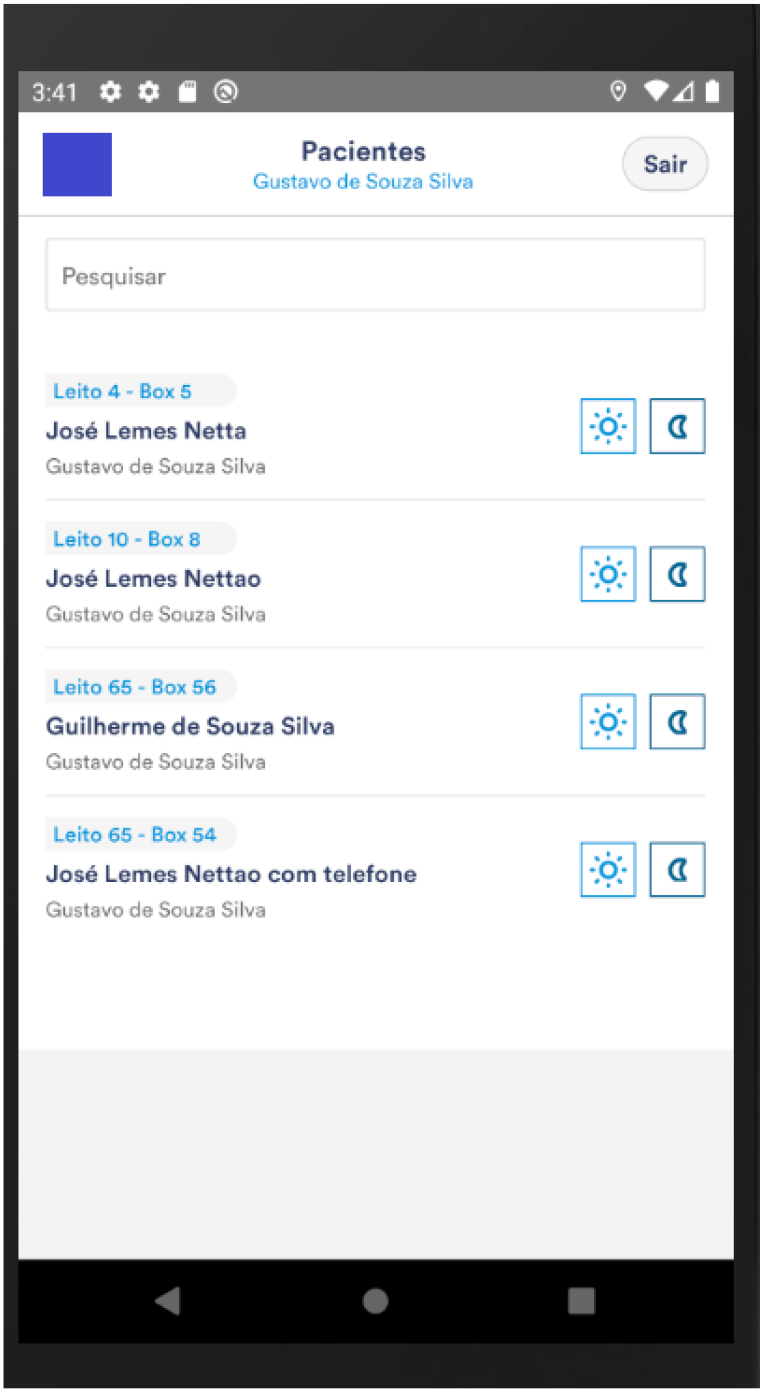
Fonte: Do autor

Figura 12 – Caso de uso de listagem - Representação no sistema web do atendimento



Fonte: Do autor

Figura 13 – Caso de uso de listagem - Representação no móvel



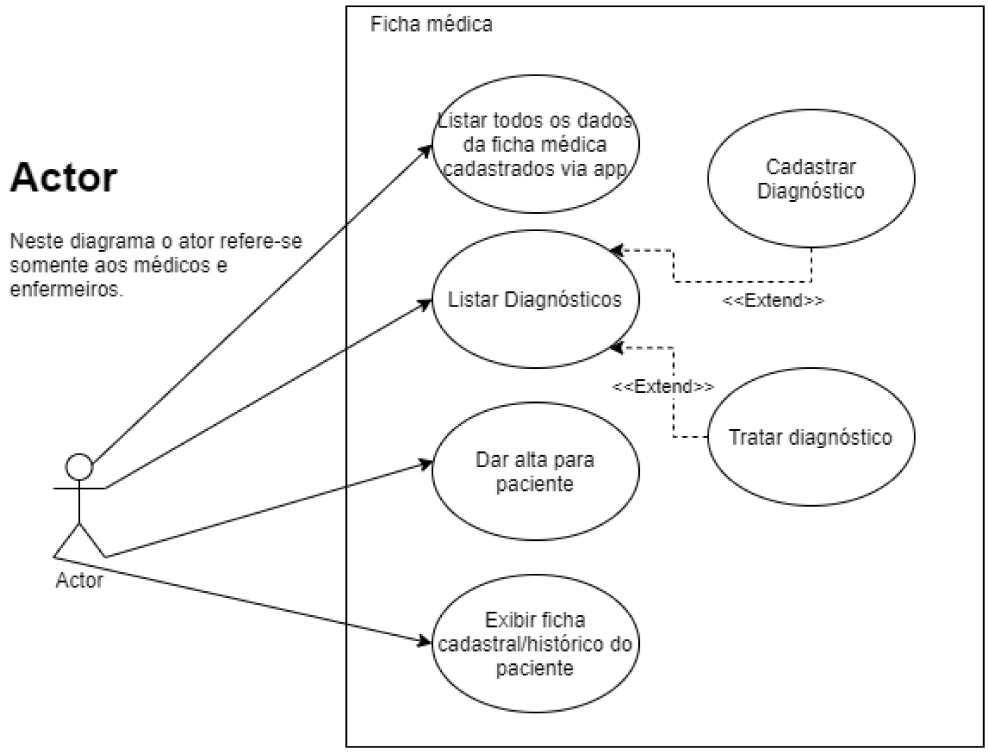
Fonte: Do autor

Vale ressaltar que, usuários de atendimento não tem acesso ao sistema móvel, pois é nele que as fichas médicas são preenchidas e essas informações só podem ser visualizadas e alteradas pela equipe médica do hospital.

Todas as funcionalidades são expostas de maneira a facilitar na visualização das mesmas, dessa forma o usuário não precisa de manuais para utilizar o sistema de forma eficiente.

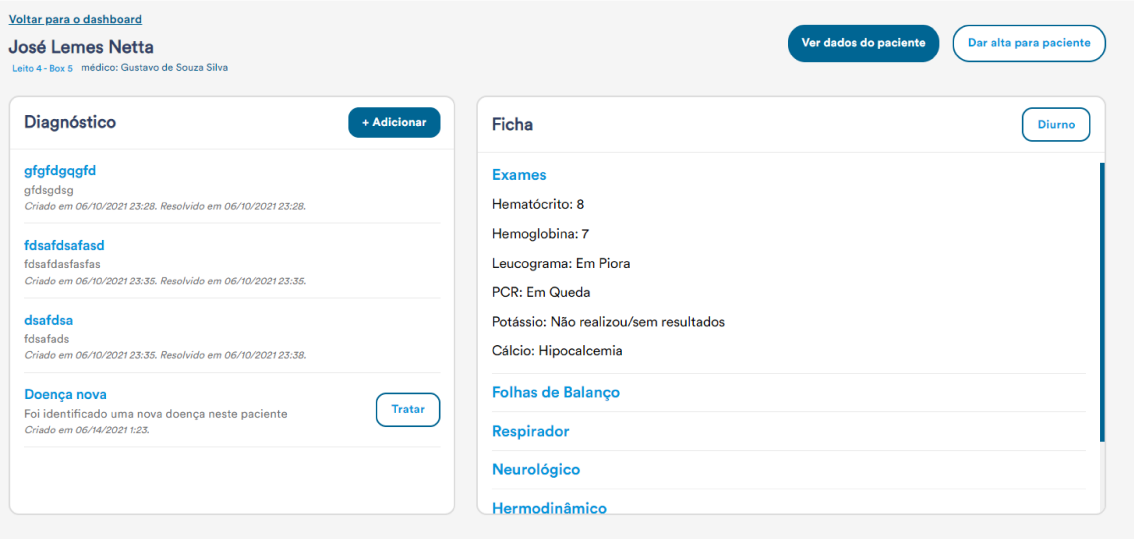
Ficha médica do paciente Web: A ficha médica no uso web é utilizada para consulta de dados médicos do paciente, cadastro e listagem dos chamados diagnósticos, que são observações feitas em cima da ficha observada, além da possibilidade de liberação do paciente quando este estiver recuperado, como é descrito pelo diagrama da figura 14 e representado no sistema desenvolvido pela figura 15

Figura 14 – Caso de uso de ficha médica web - Diagramas



Fonte: Do autor

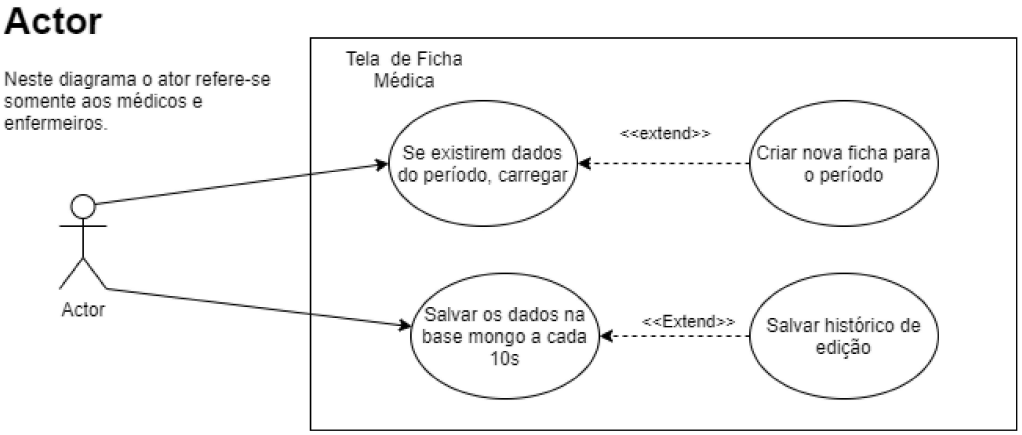
Figura 15 – Caso de uso de ficha médica - Representação no sistema web do atendimento



Fonte: Do autor

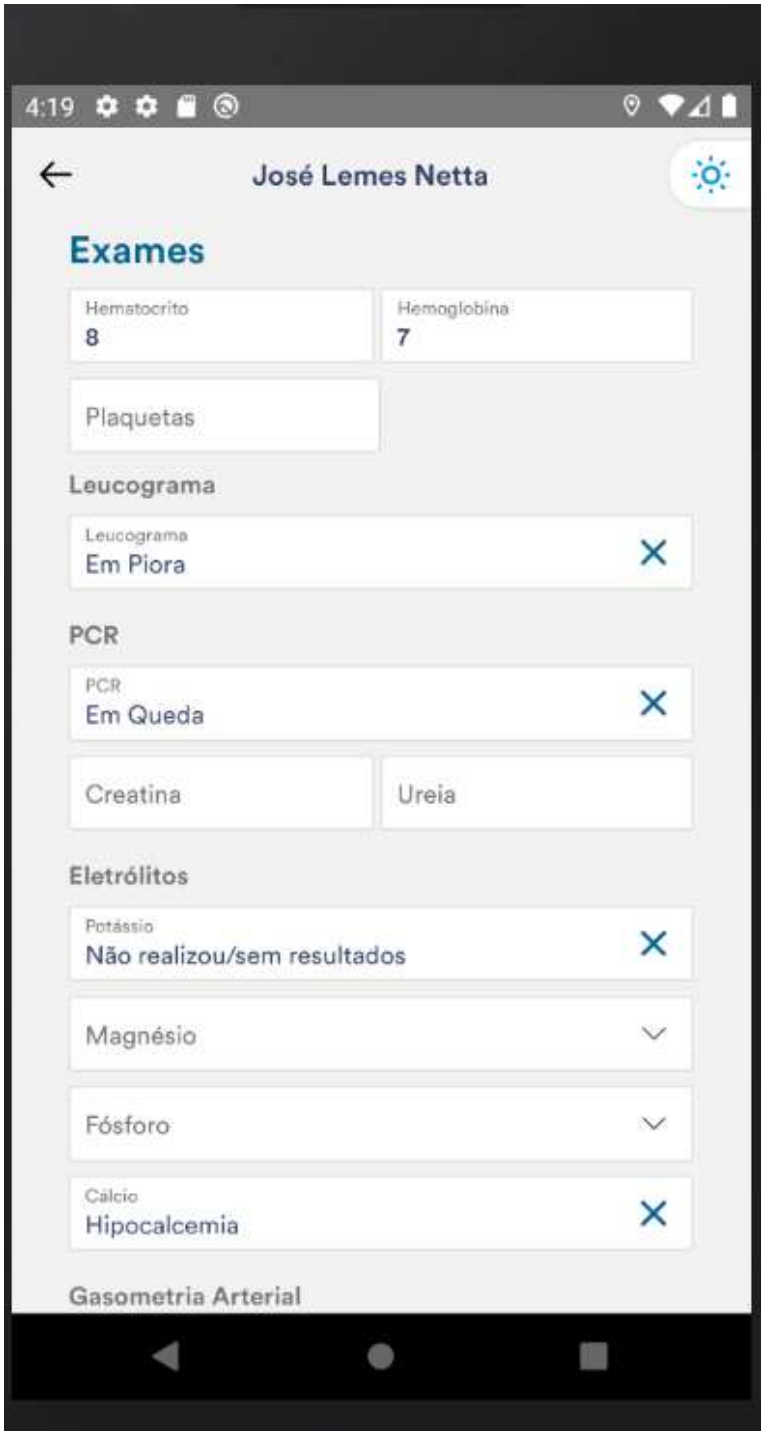
Ficha médica do paciente em dispositivos móveis: Já nessa versão, a ficha médica do paciente é utilizada para fins de cadastro dos prontuários médicos e nada além disso, é nesse caso de uso que o médico ou enfermeiro cadastra as observações feitas do paciente diretamente no leito. O caso de uso dessa funcionalidade é mostrado na figura 16 e sua foto no sistema final na Figura 17

Figura 16 – Caso de uso de ficha médica móvel - Diagramas



Fonte: Do autor

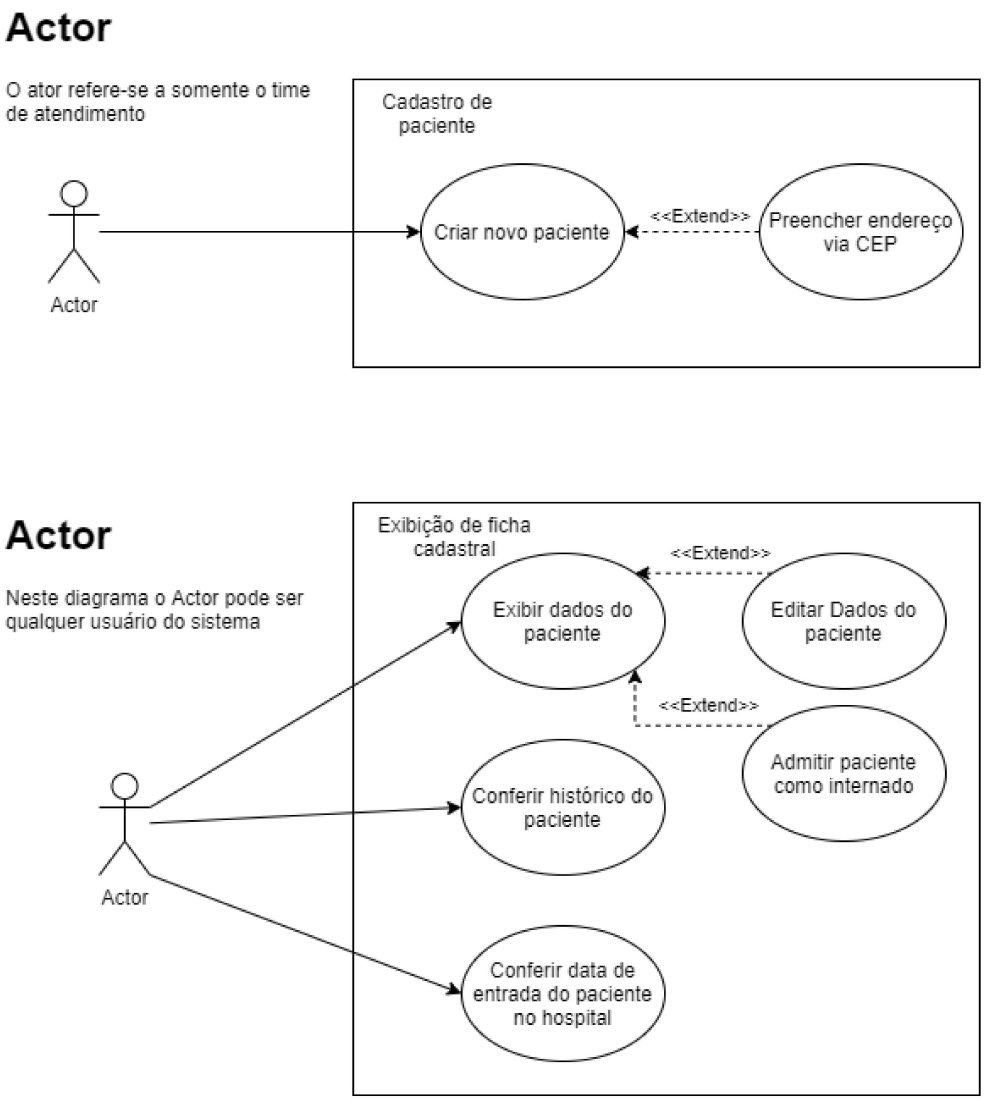
Figura 17 – Caso de uso de ficha médica - Representação no sistema móvel do atendimento



Fonte: Do autor

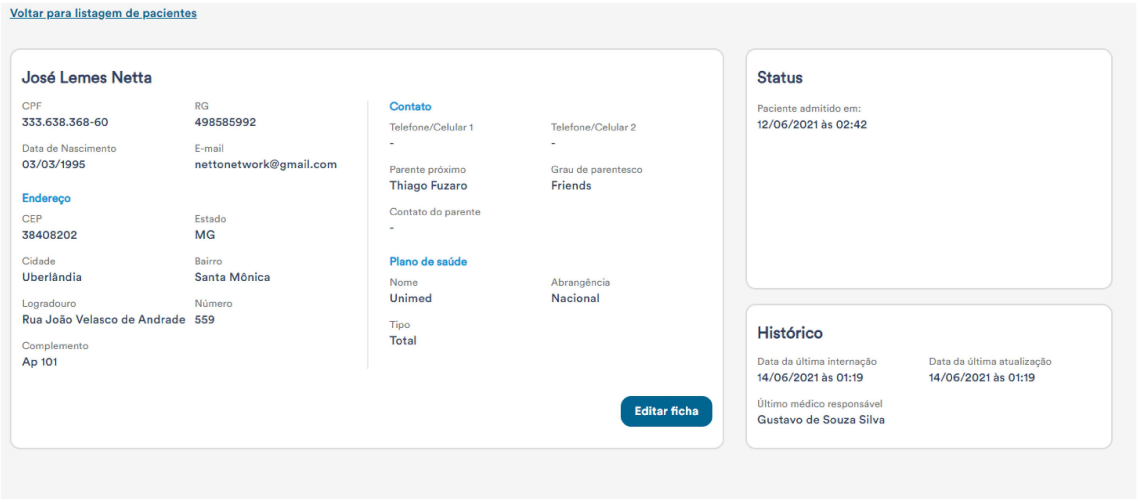
Ficha cadastral do paciente - exibição e cadastro: Esse caso de uso deixa os atendentes, capazes de criar novos pacientes, além de permitir todos os usuários do sistema a visualizar os dados cadastrais de um paciente, juntamente com seu histórico de fichas e expõe a funcionalidade de internação de uma pessoa. A Figura 18 representa o caso de uso, e as Figuras 19 e 20 são a representação no sistema do caso de uso.

Figura 18 – Caso de uso de ficha médica móvel - Diagramas



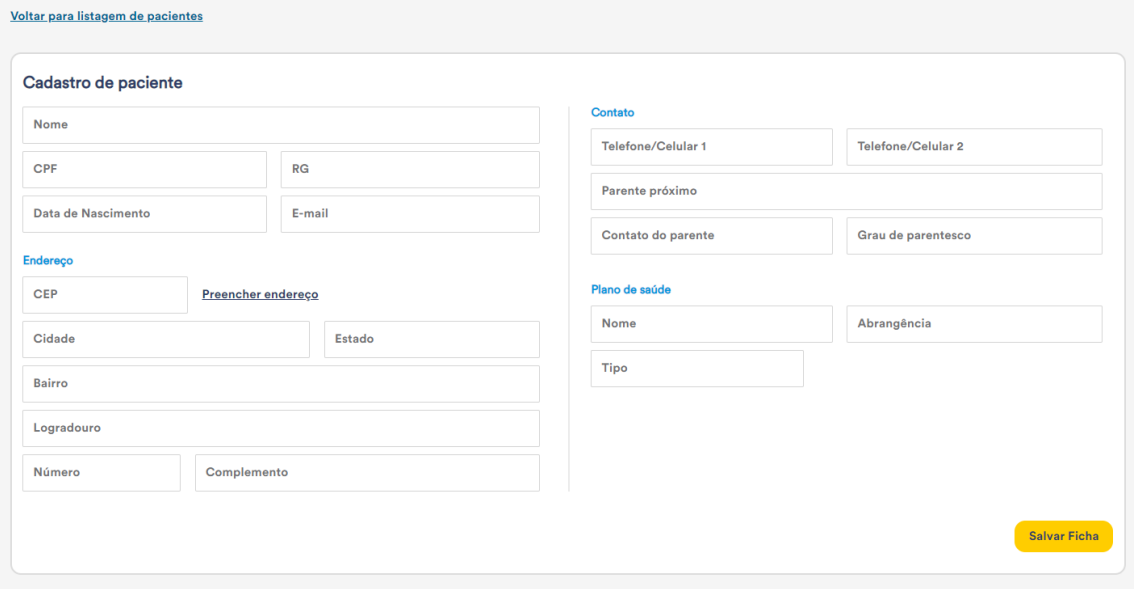
Fonte: Do autor

Figura 19 – Caso de uso de cadastro de paciente - Representação web do cadastro



Fonte: Do autor

Figura 20 – Caso de uso de cadastro de paciente - Representação web da exibição



Fonte: Do autor

Esses foram todos os casos de uso do sistema, uma observação a ser feita sobre os casos, é que para os casos de ficha médica, tanto em dispositivos móveis quanto na web, existe a troca do período da ficha, existindo os períodos do dia e da noite, esses casos não foram especificados para evitar duplicidade de imagens, já que ambos os períodos tem as mesmas funcionalidades, só se diferenciando por um pequeno enumerador cadastrado no banco de dados como DIA e NOITE.

3.6 Diferencial no Armazenamento de Dados

Como exemplificado no caso de uso do aplicativo de celular, a atualização das fichas é automática e constante, basta o usuário abrir a ficha já anteriormente criada de um paciente em seu dispositivo móvel que os novos dados inseridos na ficha são armazenados a cada 10 segundos no servidor.

É por esse motivo que o sistema utiliza dois banco de dados, todos os dados da ficha são colocados no banco de dados noSQL, pois entre esses dados não há relacionamento algum e depois com uma única operação que é realizada apenas uma vez durante a criação da ficha, essa ficha fica ligada ao paciente no banco de dados relacional (Essa ligação é feita somente com identificadores, sem informações pessoais do paciente).

Dessa forma, o sistema da ficha ganha em 3 aspectos:

- A alteração e criação de ficha médica não se torna um possível fator bloqueante para os atendentes;

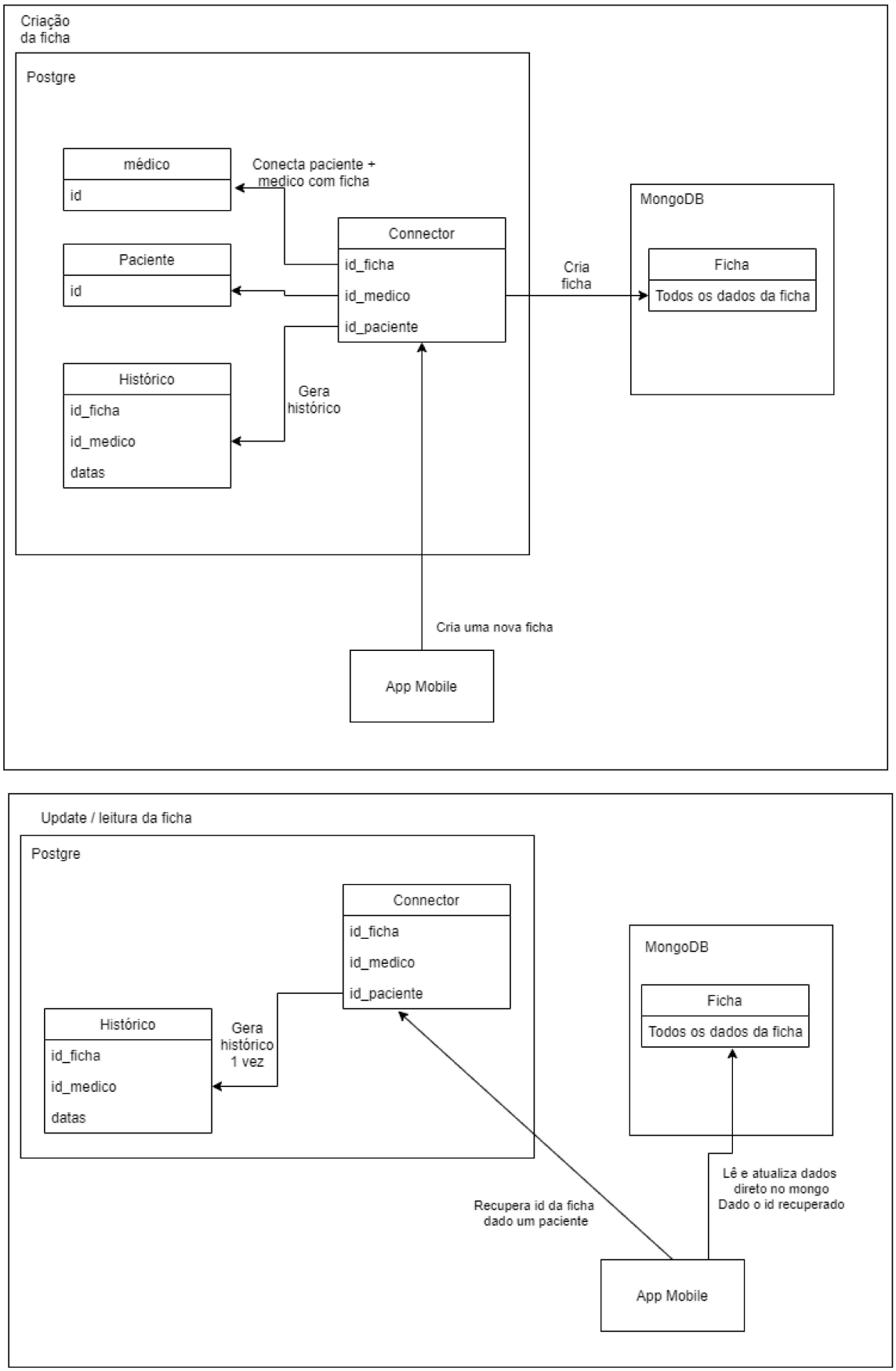
- Os dados no banco não tem vínculo direto com dados pessoais das pessoas, garantindo assim maior segurança dos dados, além de criar a possibilidade da exploração desses dados por uma inteligencia artificial;
- Em diversos momentos vários dados das fichas não serão preenchidos, ao invés de se ter uma linha em uma tabela já criada com todos esses campos iniciados com o valor nulo e com espaço reservado no armazenamento, utilizando MongoDB, esses campos nem existem no arquivo, trazendo alguma economia de espaço.

Outra vantagem importante é a possibilidade da separação do *backend* em dois micro-serviços distintos e que funcionam independentes, aumentando ainda mais a escalabilidade do sistema e reduzindo os custos, já que somente a parte do sistema necessário poderá crescer.

O Diagrama abaixo exemplifica o funcionamento do sistema descrito de armazenamento, a primeira imagem da Figura 21 representa a criação da ficha e suas operações internas, quando uma ficha é criada pelo aplicativo móvel, automaticamente na tabela *connector* se faz a relação entre uma ficha do MongoDB e os dados do paciente e do médico, além do histórico dessa ficha.

Na segunda imagem são exemplificados os processos de atualização e leitura de uma ficha, então, dado um paciente, um período e uma data, o sistema procura a conexão que atenda e retorna somente o identificador da tabela direta no MongoDB, após esse procedimento, todos os dados da ficha são alterados diretamente no MongoDB, sem mais acessos ao banco relacional.

Figura 21 – Diagrama de funcionamento do cadastro/atualização de uma ficha



Fonte: Do autor

3.7 Ficha Médica

A ficha médica foi minimizada no diagrama apresentado pela Figura 21, pois nela existem mais de 200 campos que são possíveis de serem preenchidos, esses campos são categorizados dentro dos seguintes itens: exames, folhas de balanço, monitor multiparamétrico, bomba de infusão, dispositivos, respirador, neurológico, hemodinâmico, respiratório, gastro intestinal, renal, metabólico, infeccioso, nutricional, endócrino, hematológico, pele e mucosas, osteo muscular.

Dentro de cada uma dessas categorias existem diversos campos possíveis de serem preenchidos, sendo nenhum obrigatório. Devido ao número de itens, campos são dispostos em forma de rolagem em uma única tela pelo aplicativo móvel para que a navegação seja simplificada e rápida. A Figura 22 representa mostra alguns dos inúmeros campos esperados na ficha.

Figura 22 – Pequeno trecho de código com dados da ficha

```
269  const renal = {
270    peso: String,
271    corUrina: String,
272    dialise: {
273      realizando: Boolean,
274      uf: Boolean,
275      volume: String,
276      tempoHora: String
277    }
278  }
279  const metabolico = {
280    hidratacao: String,
281  }
282  const infeccioso = {
283    remedios: {
284      amoxicilina: Boolean,
285      azitromicina: Boolean,
```

Fonte: Do autor

Todos os casos de uso do sistema foram então apresentados, bem como as telas desenvolvidas referentes a esses casos no sistema, também foram apresentadas metologias de desenvolvimento de software que colaboraram na gestão do desenvolvimento dessa ferramenta, além disso, foi dada a motivação da escolha do uso de duas ferramentas distintas de banco de dados, bem como elas funcionam no sistema.

Com isso foi construída uma ferramenta altamente escalável, com disponibilidade alta e que funciona quase em tempo real, atendendo os requisitos que um bom sistema médico deve possuir, além de atender os requisitos do já citado médico que contribuiu com o trabalho.

No próximo capítulo serão exibidas as considerações finais e futuras melhorias que podem ser realizadas em trabalhos futuros.

4 Conclusão

O objetivo desse trabalho foi atingido com um sistema que garante alta disponibilidade, além de escrita e leitura rápida dos dados e que se utiliza das últimas tecnologias disponíveis no mercado para isso.

Ao longo do desenvolvimento do sistema, ele foi alterado por diversas vezes, se moldou de acordo com as necessidades expostas pelo médico já citado neste trabalho, assim provando inclusive o bom funcionamento do Kanban, sistema que possibilitou essas alterações no escopo do projeto sem quebra no fluxo normal de trabalho e de forma extremamente rápida.

Dessa forma, tem-se um produto já preparado para produção e que atende perfeitamente as necessidades de um cliente, além de deixar em aberto diversas possibilidades de melhorias futuras.

4.1 Trabalhos Futuros

Como possíveis trabalhos futuros, sugere-se:

- Aprimoramento da usabilidade do sistema, sobretudo, na exibição dos itens da ficha dos pacientes, adicionando visões com gráficos que facilitem ainda mais o diagnóstico dos pacientes.
- Adicionar uma inteligência artificial para predição de estado do paciente, exibindo para os médicos qual paciente provavelmente está em estado mais grave.
- Separação dos *backends* de ficha e usuário/controle, de forma a testar o ganho de desempenho e escalabilidade nesse cenário.
- Implementar o cadastro e leitura de fichas utilizando *websocket* de forma que os dados fiquem disponíveis em tempo real.

Referências

CRUZ, S. C. S. Estudo comparativo entre frameworks de front-end. 2017. Disponível em: <https://www.cin.ufpe.br/~tg/2017-2/scsc-tg.pdf>. Citado na página 14.

DEVMEDIA. *Guia de ASP.NET MVC*. Disponível em: <https://www.devmedia.com.br/guia/asp-net-mvc/38190>. Citado na página 14.

JÚNIOR, A. Falando sobre node.js: Mas que diabos é isso? 2015. Disponível em: <https://tasafo.wordpress.com/2015/07/14/falando-sobre-node-js-mas-que-diabos-e-isso/>. Citado na página 15.

LUNDEN, I. *Real-Time Research: iOS Dominates Over Android When It Comes To Usage, Says Chitika*. 2012. Disponível em: <https://techcrunch.com/2012/04/21/real-time-research-ios-dominates-over-android-when-it-comes-to-usage-says-chitika/>. Citado na página 18.

MDS. Introdução express/node. 2021. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/Express_Nodejs/Introduction>. Citado na página 15.

MONGODB. *The database for modern applications*. 2021. Disponível em: <https://www.mongodb.com/>. Citado na página 16.

PEREIRA, S. R. et al. Sistemas de informação para gestão hospitalar. 2012. Disponível em: <http://www.jhi-sbis.saude.ws/ojs-jhi/index.php/jhi-sbis/article/view/206>. Citado 2 vezes nas páginas 9 e 12.

POSTGRESQL. *PostgreSQL: The World's Most Advanced Open Source Relational Database*. 2021. Disponível em: <https://www.postgresql.org/>. Citado na página 16.

RADIGAN, D. kanban. 2021. Disponível em: <https://www.atlassian.com/br/agile/kanban>. Citado na página 13.

REACT. *Virtual DOM e Objetos Internos*. 2021. Disponível em: <https://pt-br.reactjs.org/docs/faq-internals.html>. Citado na página 17.

ROBBINS, J. N. A beginner's guide to html, css, javascript, and web graphics. 2012. Citado na página 14.

ROVEDA, U. *React: o que é, como funciona e porque usar e como aprende*. 2021. Disponível em: <https://kenzie.com.br/blog/react/>. Citado na página 17.