

MRP – Monitoramento Remoto de Pacientes em Ambiente Hospitalar Utilizando IOT-LoRa (Interface Gráfica)

Lucca Defelippo de Paula Romanelli, CEFET-MG

Resumo — Com a vinda da pandemia e a superlotação de hospitais, a possibilidade de manter um monitoramento do status do paciente de maneira eficiente e remota é algo cada vez mais essencial. Com a diminuição do tamanho e do custo dos sensores e o aumento de sua eficácia, isso se tornou possível. Devido a esse contexto, este trabalho apresenta uma pesquisa sobre tecnologias necessárias para construir um sistema que captura dados oriundos de um dispositivo vestível (*wearable*), a ser usado em pacientes em hospitais para que os dados deles sejam enviados, através do protocolo LoRa, para armazenamento em um banco de dados e visualização através de uma página web. Este consiste em implementar a estrutura do banco de dados e as tecnologias de construção de uma interface gráfica para a melhor visualização das informações. Para o seu desenvolvimento são apresentadas as tecnologias de *Front-End*, *Back-End*, banco de dados e uma proposta de protótipo de página web para a exibição dos dados.

Palavras-chaves — IoT, Interface Gráfica, Banco de Dados, Paciente.

I. INTRODUÇÃO

O surto da doença causada pelo novo Coronavírus, chamado SARS-CoV-2, foi caracterizada como pandemia, em 11 de março de 2020 pela Organização Mundial de Saúde (OMS). A superlotação das unidades de saúde, a falta de leitos para internação e de equipamentos para cuidados, são problemas que agravam a situação de atendimento e monitoramento hospitalar [1].

A observação dos indicadores das funções vitais de pacientes, como a frequência respiratória, saturação de oxigênio no sangue, pressão sanguínea, frequência cardíaca e temperatura corporal, é uma das práticas mais comuns e importantes em hospitais [2], uma vez que já é conhecida e estudada a relação entre a piora do quadro clínico do paciente e a alteração nestes indicadores horas antes da piora [3]. Portanto, possuir uma maneira constante de monitoramento de alguns sinais vitais dos pacientes de forma não invasiva junto à

maneira de observar tais características remotamente, pode ser de grande auxílio no cotidiano hospitalar.

Uma solução atual para o monitoramento dos sinais vitais dos pacientes, aproveitando-se de um dos grandes avanços tecnológicos dos últimos tempos, é a utilização dos “*wearables*”, que são pedaços de roupas, acessórios ou outras peças vestíveis, compostos por diversos sensores e outros componentes eletrônicos para monitoramento da saúde e aplicação de remédios periodicamente [4].

Algumas empresas de saúde já empregam o monitoramento de pacientes com o uso de “*wearables*”, principalmente os relógios inteligentes disponíveis no mercado como o Apple Watch e Samsung Galaxy Watch, como forma de melhorar a qualidade de vida e diminuir os custos.

O monitoramento remoto e em tempo real de pacientes em casa auxiliou na diminuição da quantidade de casos em que houve a necessidade de intervenção hospitalar (23% menor que em pacientes que não utilizavam) e de casos de readmissão ao hospital após liberação (44% menor chance de retorno ao hospital em 30 dias e 38% menor em 90 dias), o que acarretou em uma diminuição de 11% [5].

Porém, ainda faltam estudos sistemáticos da aplicação do uso destes “*wearables*” em ambientes hospitalares para uma maior gama de idades e quadros clínicos diferentes [6].

A proposta deste trabalho é contribuir com esses estudos propondo um sistema de monitoramento remoto de sinais vitais de pacientes em ambiente hospitalar.

II. OBJETIVO

O objetivo do trabalho, portanto, é a construção de um dispositivo “*wearable*”, com base no conceito de IoT (*Internet of Things*) para monitoramento de alguns sinais vitais dos pacientes e desenvolvimento de uma rede utilizando a tecnologia LoRa (*Long Range*) para envio dos dados coletados na pulseira para um *gateway* e armazenamento em banco de dados. O desenvolvimento de uma interface gráfica para administração e consulta dos dados coletados complementa este sistema.

Devido a sua complexidade e abrangência, o trabalho foi organizado em 3(três) frentes de desenvolvimento, a saber:

Trabalho de Conclusão de Curso enviado em 25 de agosto de 2021 para análise da Banca Examinadora composta pelos Professores José Hissa Ferreira e Alexandre Rodrigues Farias realizado pelo estudante de Engenharia Elétrica no CEFET-MG Lucca Defelippo de Paula Romanelli com o grupo composto

por Vinicius Santana Nascir e Filipe Marigo Soares, juntamente com o Professor Orientador Antony Chiaratti e Professor Túlio Charles de Oliveira.

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

hardware, rede de comunicação e interface gráfica; conforme diagrama de arquitetura apresentado na Fig. 1. Essas frentes de trabalhos foram realizadas em conjunto com os alunos: Vinícius Santana Nascier, para o desenvolvimento do hardware e Filipe Marigo Soares, para o desenvolvimento da rede de comunicação.

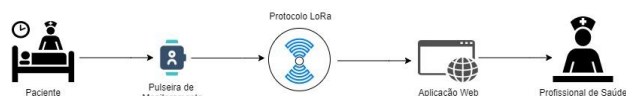


Fig. 1. Monitoramento Remoto de Pacientes em Ambiente Hospitalar utilizando IoT - LoRa

Este artigo trata especificamente a terceira frente de trabalho, que tem por objetivo a construção de uma interface gráfica para que o usuário final tenha acesso aos dados de forma segura e *Mobile Friendly*, isto é, disponíveis em uma página na internet que pode ser acessada pelo computador, em celulares ou tablets; o armazenamento dos dados do sistema em um banco de dados, protegidos por um protocolo de segurança.

III. METODOLOGIA

Este trabalho é de natureza aplicada, visando à construção de conhecimentos para a solução de um problema específico e prevendo uma aplicação prática na área da saúde, e foi desenvolvido em duas etapas. A primeira etapa, apresentada em uma monografia, constituiu-se em um estudo sobre a viabilidade e relevância do projeto e uma revisão bibliográfica dos principais conceitos das tecnologias a serem utilizadas.

Para definir a viabilidade do trabalho e buscando entender as condições atuais de monitoramento de pacientes em hospitais públicos e privados, foi realizada uma pesquisa com os profissionais da área da saúde. Essa pesquisa foi feita por meio de um questionário *online* elaborado com a ferramenta *Google Forms*, montado com questões abertas e fechadas, visando buscar informações sobre 3 (três) aspectos chave.

O primeiro aspecto foi sobre a forma de conectividade a internet dentro do ambiente hospitalar público e privado, e quais dispositivos estão disponíveis para uso coletivo dos profissionais, possibilitando com isso a definição das aplicações, interfaces finais e quais dispositivos devem ter compatibilidade com o sistema proposto.

O segundo aspecto chave foi relacionado à forma com que os profissionais da saúde acessam os prontuários e históricos dos pacientes. É importante entender o acesso a essas informações para manter o padrão adotado, fazendo com que a nova tecnologia seja inserida de forma não invasiva e natural.

Por fim, o terceiro e último aspecto foi relacionado a tecnologia de monitoramento utilizando “*wearables*” com a realidade dos hospitais, ou seja, se é comum tal monitoramento na realidade atual e como a inserção da mesma poderia melhorar nos resultados médicos.

Através dessa pesquisa, foi possível observar que possuir uma forma constante de monitoramento de alguns sinais vitais dos pacientes junto à maneira de observar as características de um paciente remotamente, podem ser de grande auxílio no cotidiano hospitalar.

Definiu-se, então, que o projeto seria constituído por uma pulseira de monitoramento remoto, com as informações protocoladas em LoRaWAN, enviadas por LoRa, para um banco de dados e a construção de uma interface capaz de captar as informações e mostrar para o usuário final.

Na segunda etapa do projeto, que se refere a execução propriamente dita, algumas definições de fluxo do trabalho apresentadas na monografia, embora relevantes para o projeto, tiveram que ser revistas e até mesmo retiradas.

Não haverá utilização de um banco de dados NoSQL para armazenamento dos dados coletados da pulseira. Com isso, o diagrama de fluxo de utilização ficou conforme apresentado na Fig. 2 e o sistema constituído basicamente das seguintes funções:

- cadastro de novos pacientes;
- cadastro de pulseiras no sistema, para obtenção de uma identificação única;
- criação de uma conexão entre o número de identificação da pulseira e o paciente ao qual a ela está designada;
- atualização dos dados dos sensores da pulseira para o banco de dados de forma periódica. Os dados dos sensores usados para monitorar os indicadores das funções vitais são agrupados pelo microcontrolador e enviados, com um formato de mensagem padrão já estabelecido, através do protocolo LoRa;
- acesso aos dados através de uma interface de usuário web, mobile ou utilizando *tags* NFC. Ação pela qual o profissional da equipe médica pode acessar os dados de cadastro do paciente assim como as últimas leituras dos sensores relacionadas àquele paciente.

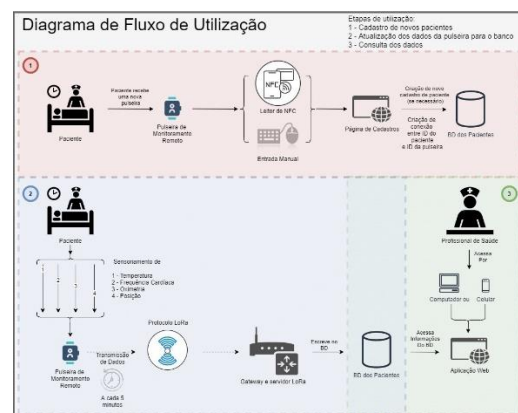


Fig. 2. Diagrama de Fluxo de Utilização do Projeto

IV. METODOLOGIA ESPECÍFICA

Tendo em vista que este documento é focado na parte de obtenção de dados do gateway LoRa e a apresentação em uma página web, a metodologia específica é composta pela construção da estrutura do servidor, da estrutura do banco de dados, da API (*back-end*) e pela exibição dos dados (*front-end*).

A. Construção da Estrutura do Servidor

A execução desta fase do projeto iniciou-se construindo a estrutura do servidor, para isso foi utilizado a *Amazon Web*

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

Service (AWS), que oferece um serviço de servidor em nuvem. Como nossa aplicação é em caráter de teste e desenvolvimento, conseguimos nos adequar ao plano gratuito da Amazon, que disponibiliza, pelo serviço RDS – *Relational Database*, uma instância do sistema gerenciador de banco de dados Maria-DB, que segue o mesmo padrão do MySQL, e pelo serviço EC2 – *Elastic Computing Cloud*, uma máquina web para subir o código com uma porta de acesso HTTP.

Além desses serviços foi utilizado também o *Elastic Benstalk*, que é um serviço que permite a implementação de aplicativos controlando os serviços de S3 de armazenamento e EC2, sendo assim responsável por subir o servidor de forma automática.

Outra ferramenta importante utilizada nesse processo foi o GitLab que é uma plataforma de DevOps (Desenvolvimento e Operações) onde há uma integração com o *Elastic Benstalk* para enviar as atualizações automaticamente para o servidor de produção. Além disso, foi feito o versionamento do projeto usando o sistema de controle GIT, com duas ramificações, uma em produção com a versão estável do código e uma em desenvolvimento na qual eram aplicadas as modificações e testadas antes de enviar o código para Amazon.

B. Estrutura de Banco de Dados

A estrutura de banco de dados foi desenvolvida no sistema de gerenciamento de banco de dados o MariaDB, que segue o mesmo padrão do MySQL, usando a linguagem de consulta estruturada, SQL – *Structured Query Language*.

Foram criadas 5 (cinco) tabelas para conter os dados do sistema, a saber: usuários (*user_entity*), dados (*data_entity*), dados da pulseira (*pulseira_entity*), pacientes (*paciente_entity*) e conexão entre o paciente e pulseira (*conexão_entity*).

A tabela de usuários contém o e-mail, senha e a função exercida pelo usuário do sistema. A tabela de dados armazena os seguintes dados obtidos através das pulseiras: identificador da pulseira (*node*), data de envio das informações (*time_utc*), dados de controle do LoRa (*seqnp*, *rsi*, *snr*, *senStatus*), o batimento cardíaco (*heart_Rate*), a temperatura corporal (*temperature_degC*), oximetria (*oximetry*) e posição (*position*). A tabela de dados da pulseira contém a identificação dela no sistema (*id*), a identificação LoRa (*node*) e o seu status atual (*status*). A tabela de pacientes contém nome, idade, telefone, contato de emergência, telefone de emergência, sintomas, histórico do paciente, sexo e CPF. Por fim, a última tabela contém a conexão entre o paciente e pulseira com a data de início e a data final da conexão.

Com essas tabelas é possível cruzar a referências e obter os dados para todas as fases de projeto.

C. Construção da API (Back-End)

Para a construção da API, foi utilizado o framework *NestJS* que é uma implementação do software de código aberto Node.js usado no navegador Google Chrome e em outros navegadores. A linguagem escolhida é o *TypeScript*, uma evolução do JavaScript cuja tipagem é forte, o que auxilia na construção do software.

A API deve ser capaz de fazer o CRUD (*create, read, update, delete*) no banco de dados, para isso deve-se utilizar um “mapeador” de objeto relacional (ORM), o escolhido foi o *TypeOrm* por ser o mais maduro disponível para o *TypeScript*, além de ser ofertado pelo framework utilizado. Feita a instalação e importação no projeto, é necessário criar as entidades para cada uma das tabelas de forma a gerar objetos a partir dos dados salvos no banco de dados.

Com os dados no banco de dados relacionados a objeto são criados os *controllers*, responsáveis por abrir as rotas onde as informações são obtidas. Os *controllers* determinam o método HTTP e os parâmetros do método. Por exemplo, o método POST, é usado para criar um dado novo, utilizamos o método pela rota “/pulseira” e ele deve conter como corpo da requisição um JSON com o *node* da pulseira e o seu *status*. A Fig. 3 mostra o exemplo da requisição no software Insomina usado para fazer requisições HTTP.

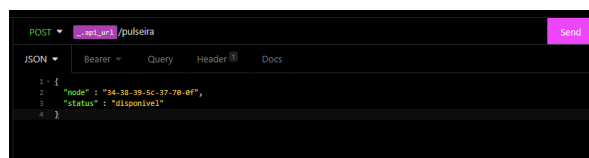


Fig. 3. Exemplo de requisição do sistema no Insomina

Com os *controllers* construídos precisamos criar *services* que contem a regra de negócio. Por meio das regras de negócio conectamos as rotas abertas pelos controladores com as nossas Entidades, para isso usamos modelos chamados DTO, por meio deles os dados são desenvolvidos e criados.

Além da manipulação e criação dos dados, temos que fazer a segurança de forma que apenas usuários certificados sejam capazes de fazer requisições para o nosso sistema. A autenticação foi feita a partir de *Bearer Token*, um conjunto de dados criptografados em que uma chave privada constrói a criptografia e uma chave pública a lê. Para criar o Token usamos o *JSON Web Token (JWT)* que é o padrão da Internet.

D. Exibição dos dados (Front-End)

Para o front-end foi escolhida a plataforma Angular, junto com a linguagem de *design* Material, como uma forma de padronizar o estilo e ter uma biblioteca de componentes para auxiliar a construção.

Utilizando o conceito de componentes do Angular foram desenvolvidas as páginas de login(auth), pacientes, pulseiras, conexões e dados, sendo que cada uma das páginas é responsável por um dos módulos do *back-end*.

O *design* do site, junto com a paleta de cores, teve apenas uma abordagem funcional, que merece um estudo maior, principalmente considerando que em hospitais já existe um padrão de cores e é relevante uma adequação para o sistema.

As páginas possuem uma metodologia de apresentação em tabelas, com botões de ações, sendo possível criar, visualizar, editar e deletar cadastros.

Seguindo a demanda do objetivo teve-se um cuidado com a utilização do software para o uso em tanto em tablets quando

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

em celulares.

V. RESULTADOS E DISCUSSÃO

A seguir é apresentado o resultado da construção de cada componente da interface gráfica e os desafios enfrentados, observando a estrutura adotada na metodologia específica de construção da estrutura do servidor, estrutura do banco de dados, construção da API (*back-end*) e exibição dos dados (*front-end*)

A. Construção da Estrutura do Servidor

O ambiente da *Amazon* foi montado e criada a url <http://tccprod-env.eba-tmqq3bds.us-east-2.elasticbeanstalk.com/>.

O *site* utiliza conexão *http* que não é segura. Para uma aplicação real é necessária a criação de um certificado DNS, que a própria *Amazon* possui e a compra de um domínio para exibição do *site*. Esse processo é relevante em questão de segurança e criptografia dos dados, mas para a produção do trabalho por não envolver dados reais não foi desenvolvida essa parte da estrutura.

A Fig. 4 mostra o serviço *Elastic Beanstalk* rodando a aplicação *Node.js* do projeto.

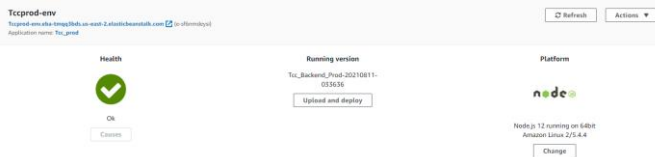


Fig. 4. Serviço *Elastic Beanstalk* rodando aplicação *Node.js*

Além da montagem do ambiente do *site*, foi feita uma página do *Git-Lab* para versionamento do código, isto é, caso encontre-se bugs no código é possível, através dessa página, voltar versões antigas se caso. É uma integração por meio das *pipelines*, na qual o código é verificado se há problemas e automaticamente integrado a *Amazon*, podemos observar na Fig. 5 alguns dos envios do código para a plataforma.

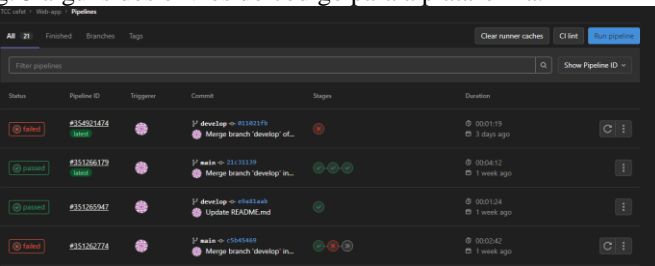


Fig. 5. Código enviado ao *git* onde podemos ver a integração com a *Amazon*

B. Estrutura de Banco de Dados

As tabelas foram criadas sobre a maior instância gratuita para o desenvolvimento, a qual é oferecida pela *Amazon*. Pode-se observar na Fig. 6 que esse tamanho de instância foi suficiente para o desenvolvimento da interface gráfica, pois foram utilizadas poucas conexões e o armazenamento de poucos dados.

Porém, em implementações reais é necessário um estudo de

quantidade de conexões simultâneas, armazenamento e memória a serem usados pelo banco. A própria *Amazon* oferece instâncias maiores e que conseguem fazer mais conexões simultâneas.

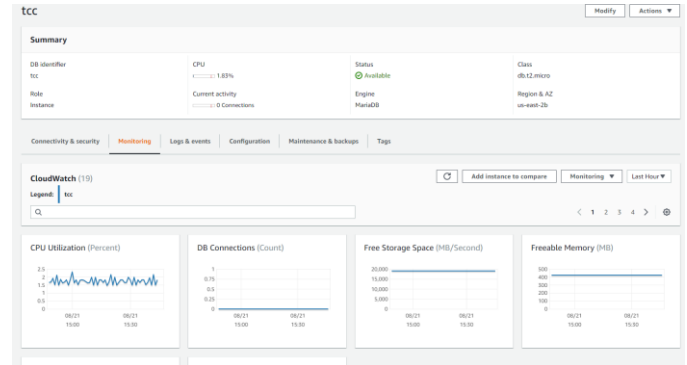


Fig. 6. Instância do banco de dados rodando no serviço *RDS* da *Amazon*.

O diagrama de relação de entidades, Fig. 7, foi feito usando o software *DBeaver*, que conecta no banco de dados. Nele podemos ver as tabelas e os relacionamentos feitos.

A estrutura do banco é feita a partir de *migrations*, ou seja, as alterações que forem feitas nas entidades do projeto irão alterar o banco, sendo assim é muito fácil adicionar ou editar campos diretamente na API. Isso é importante para o desenvolvimento de novas regras de negócio e novas implementações.

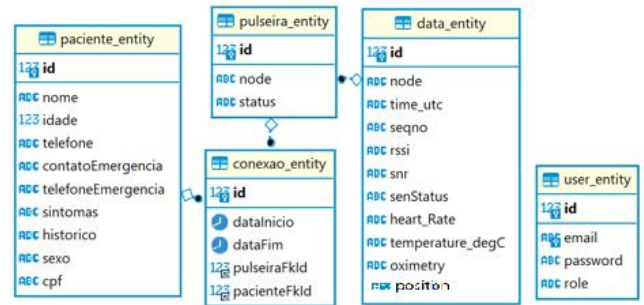


Fig. 7. Diagrama de Relação de Entidades.

C. Construção da API (Back-End)

Os *token* de APIs permitem que o usuário faça a autenticação com aplicativos da nuvem contornando a verificação em duas etapas. Para tanto, foram geradas duas chaves secretas para sua criptografia, ambas com 50 caracteres aleatórios sendo muito difíceis de serem quebradas com força bruta.

Um *token* é utilizado para os usuários acessarem o sistema e o outro para as pulseiras imputarem os dados. O *token* de uma pulseira é expirado em questão de 10 segundos, o suficiente para imputar o dado e nada mais, já os de usuário duram uma hora. Para que não seja necessária a realização do login no sistema de hora em hora, o sistema verifica que o *token* está vencendo e o atualiza, permitindo dessa forma que o usuário tenha uso fluido.

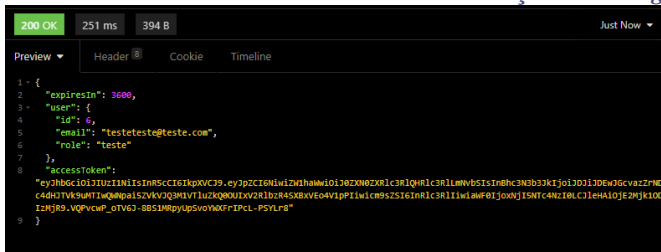


Fig. 8. Exemplo de um token JWT.

Uma aplicação possível para o esquema de usuário que foi feito é dar acesso a uma função específica, como por exemplo, cadastro de pulseira para um usuário do tipo manutenção, cadastro de paciente para usuários enfermeiras e dados apenas para médicos. Essa é uma aplicação que é possível de ser feita, porém não será implementada nesta versão do código.

O projeto é separado em grupos de controle com suas funções específicas. Para facilitar a documentação foi construída uma página usando a ferramenta Swagger, que permite uma rápida visão do sistema. Pelo Swagger podemos ver todas as requisições, quais os parâmetros e o corpo da requisição. As requisições do tipo POST são as criações de novos dados, PUT são para alterações de dados, GET para buscar os dados e DELETE para deletar os dados. No Swagger cada requisição tem os parâmetros, o corpo e o resultado esperado pela resolução, conforme a figura 9.

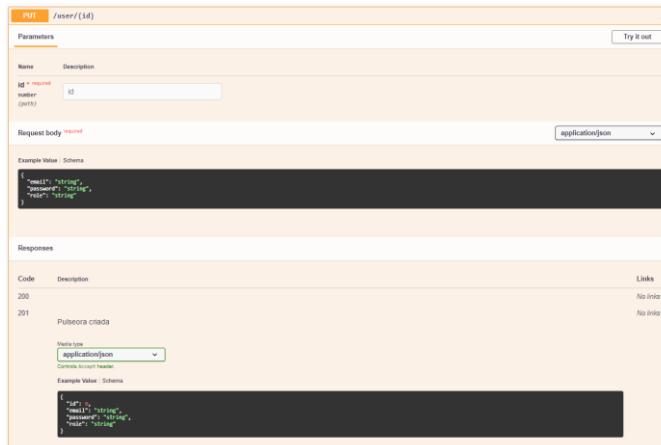
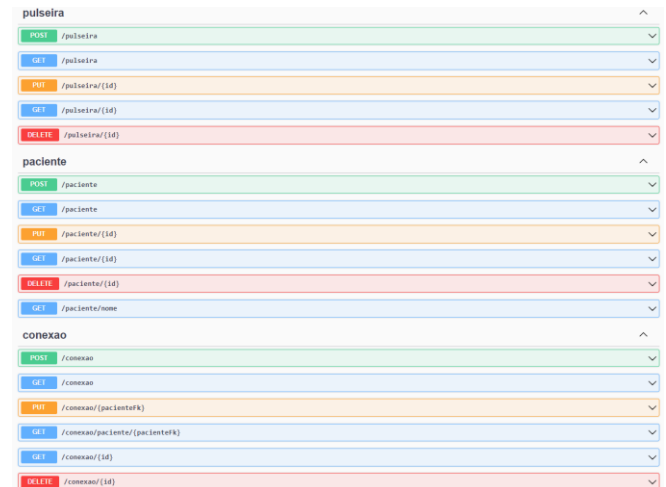


Fig. 9. Exemplo de requisição no Swagger.

O primeiro grupo é o *auth*, que apresenta as duas requisições de autenticação. O grupo *user* contém as operações relacionadas aos usuários do sistema. Essas requisições são de uso exclusivo do administrador do sistema e os usuários não tem acesso a ela. As requisições do grupo *data* tem o POST para o gateway enviar os dados da pulseira e os GETS que o sistema usa para pegar as informações dos dados.


Fig. 10. Requisições de *auth*, *user* e *data* no Swagger.

As requisições do grupo *pulseira*, *paciente* e *conexao*, apresentadas na figura 11, são as responsáveis pelo consumo e controle das informações das tabelas referentes a esses dados.


Fig. 11. Requisições de *paciente*, *pulseira* e *conexao* no Swagger.

Além das requisições temos os *schemas*, na Fig. 12, que mostram as respostas e corpos das requisições. Para acessar o Swagger basta entrar na url na rota “/api/swagger”.

Para testar a API, foi elaborada uma coleção no software Insomnia, com ele podemos inserir o *token* de autenticação e ter acesso as respostas direto do servidor local e de produção.



Fig. 12. Esquemas com os modelos de interação do Swagger.

D. Exibição dos dados (Front-End)

A primeira página do sistema é a de autenticação, onde tem-se um formulário e um logotipo para que a autenticação dos usuários seja feita, conforme apresentado nas Fig. 13 e 14, sendo uma a versão *web* e uma *mobile*.

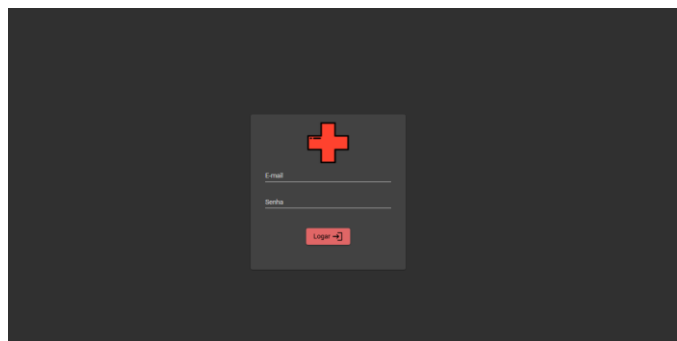


Fig. 13. Página de login versão web.

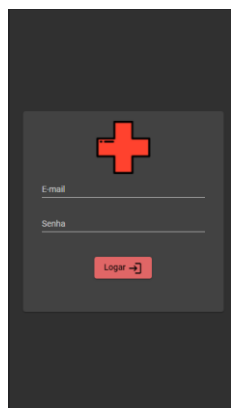


Fig. 14. Página de login versão mobile.

Ao entrar no sistema o usuário é direcionado para a página de dados coletados, onde é mostrado um formulário de busca de paciente. Ao encontrar um paciente a tela mostra dois campos para escolha de data, em que é possível escolher qual o período de dados deseja-se exibir, as Fig. 15 e 16 apresentam o

resultado de uma busca de dados.

Uma melhoria para essa página é a aplicação de um *web-socket* capaz de atualizar os dados em *real time*. Outra questão em que merecem atenção e que para os dados de um paciente antigo embora mantenham-se registrados no banco de dados, não foi feito nenhum trabalho para exibição desse histórico de medições, sendo isso uma sugestão para melhoria do sistema.

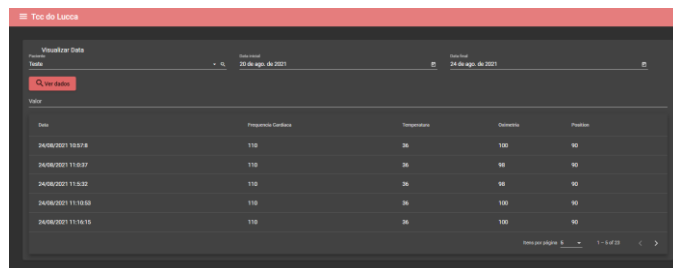


Fig. 15. Página de Dados coletados.

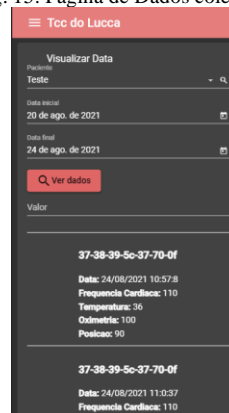


Fig. 16. Página de Dados coletados versão mobile.

As páginas de pulseira, paciente, usuários e conexões seguem um padrão no qual a página inicial exibe todos os dados do sistema em uma tabela. Além dos dados, a tabela exibe a opção de edição e remoção do dado. Outros recursos da página são um filtro que exibe um dado específico e um botão para criação de novas entradas. Esse botão leva a outra página com o formulário para a adição das entradas referentes as páginas. Ao *clicar* em adicionar um dado ou editar, o usuário é redirecionado a um formulário onde ele é capaz de editar e criar os dados, conforme apresentado nas Fig. 17, 18 e 19.

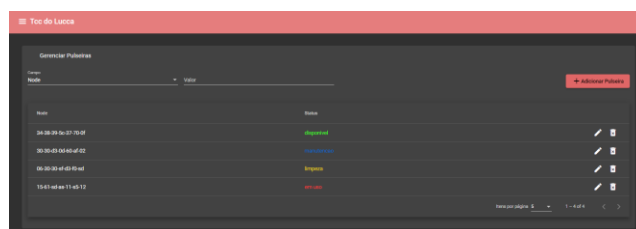


Fig. 17. Tabela de apresentação dos dados gerais.

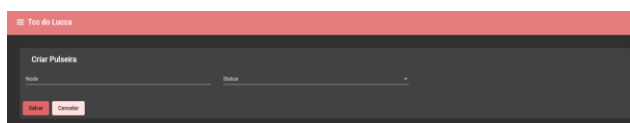


Fig. 18. Formulário de adição, edição de dados gerais.

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

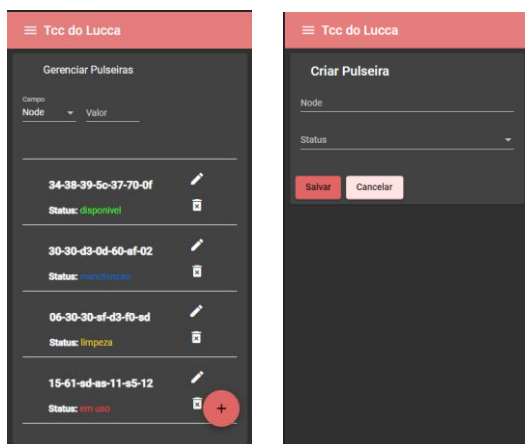


Fig. 19. Tabela de apresentação de dados mobile (à esquerda) e formulário de adição, edição de dados na versão mobile (à direita).

Outro recurso implementado na página é a navegação por um menu lateral, onde ao clicar no ícone de hambúrguer no topo esquerdo, abre-se todas as rotas do sistema, além da possibilidade de desconectar do sistema, conforme apresentado na figura 20.

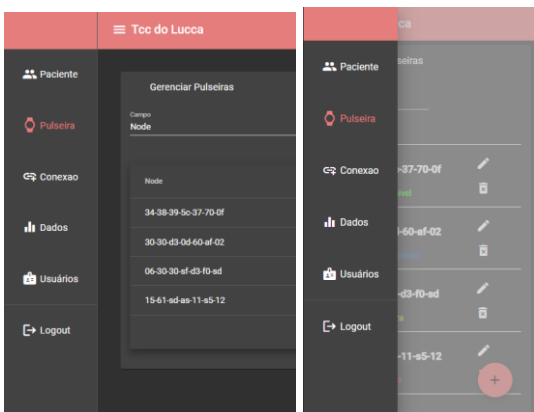


Fig. 20. Side nav versão web (à esquerda) e versão mobile (à direita).

VI. CONCLUSÃO

Um dos objetivos do uso da informática na medicina é dispor informações, onde e quando elas forem necessárias, para os profissionais da área de saúde. A utilização de “wearables” é uma solução atual para o monitoramento dos sinais vitais dos pacientes, aproveitando-se de um dos grandes avanços tecnológicos dos últimos tempos [4].

A superlotação das unidades de saúde, a falta de leitos para internação e de equipamentos para cuidados, são problemas que agravam a situação de atendimento e monitoramento hospitalar com o avanço da pandemia causada pelo novo Corona vírus [1].

A relevância deste trabalho se encontra no próprio contexto em que está presente a pandemia, na dificuldade de acesso a utilização de “wearables” devido a seu custo e na importância que um sistema de monitoramento remoto pode desempenhar em ambientes hospitalares auxiliando no atendimento preditivo de um paciente.

Assim, a interface gráfica, objeto deste documento é utilizada

para cadastro, consulta e análise dos sinais vitais de um paciente, oriundos dos dados coletados de uma pulseira. Essa interface é composta por uma estrutura de banco de dados, API de segurança, *back-end*, *front-end* e estrutura de servidor em nuvem, e é parte integrante do sistema de monitoramento remoto de sinais vitais de pacientes em ambiente hospitalar utilizando IoT-LoRA, obtido durante a realização deste projeto de formação acadêmica.

O desenvolvimento dessa interface gráfica teve como preocupação primordial a facilidade de acesso aos dados de forma segura, disponível em uma página que pode ser acessada pelo computador, celular ou *tablets* na internet, além de atender aos propósitos estabelecidos.

Os trabalhos em conjunto servem como prova de conceito do sistema criado, com o projeto de um protótipo de placa de circuito impresso capaz de ler os indicadores de funções vitais de pacientes e envia-los utilizando protocolo LoRa; com uma estrutura de servidor central já implantada, que trata dados recebidos pelo servidor LoRa e os adequa ao padrão esperado para armazenamento; e com o desenvolvimento de uma infraestrutura capaz de utilizar os dados reunidos e exibi-los em uma página *mobile-friendly*.

Um ponto importante a ressaltar foi a falta de testes e homologação do sistema, impossibilitando, portanto, a validação dos processos construídos. Porém, através da forma como o software foi desenvolvido é possível aumentar sua estrutura para receber novos usuários e dados, além de construir novas regras de negócio.

Portanto, fica como sugestão para pesquisas futuras a integração deste sistema com sistemas de prontuários eletrônicos existentes, bem como algumas melhorias e ajustes, tais como: acrescentar o sensor de frequência respiratória para monitoramento; utilização de um banco de dados NoSQL para armazenamento dos dados coletados da pulseira; desenvolvimento de um *web-socket* para atualização dos dados em *real time*; A.

Essas melhorias visam o aumento da qualidade e robustez do produto final, permitindo assim sua implementação em um ambiente hospitalar para homologação.

AGRADECIMENTOS

Agradeço primeiramente a Deus pela oportunidade e pelo conhecimento adquirido na realização desta atividade de formação acadêmica.

Sou grato aos meus parceiros de grupo Vinicius e Filipe, ao meu orientador Prof. Anthony Chiaratti, ao coorientador Prof. Tulio Charles e a meu amigo João Victor Cardoso pelo suporte durante o período de pesquisa e produção deste trabalho.

Agradeço também a minha tia, Dra. Ana, que me auxiliou com sua expertise na área da medicina.

E finalmente, agradeço aos meus pais, irmão, namorada e meus amigos, por todo o incentivo e apoio.

Centro Federal de Educação Tecnológica de Minas Gerais – CEFET-MG

REFERÊNCIAS

- [1] Gallasch, Cristiane Helena et al, “*Prevenção relacionada à exposição ocupacional do profissional de saúde no cenário de COVID-19*”, Revista Enferm UERJ, Rio de Janeiro, 2020; 28:e49596, DOI: <http://dx.doi.org/10.12957/reuerj.2020.49596> Disponível em: <https://www.e-publicacoes.uerj.br/index.php/enfermagemuerj/article/view/49596/33146> Acessado em: 30 de março de 2021
- [2] Brekke, I. et al, “*The value of vital sign trends in predicting and monitoring clinical deterioration: A systematic review*”, Publicado: 15 de janeiro de 2019 <https://doi.org/10.1371/journal.pone.0210875> Disponível em: <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0210875> Acessado em: 20 de março de 2021.
- [3] Kaase, J. et al. “*A comparison of antecedents to cardiac arrests, deaths and emergency intensive care admissions in Australia and New Zealand, and the United Kingdom - the ACADEMIA study*”. Em: Resuscitation. Disponível em: <https://pubmed.ncbi.nlm.nih.gov/15325446/>. Acessado em: 20 de março de 2021.
- [4] Gouche, Catherine, G. e JONES, J.. “*Wearables, wearing, and the rhetorics that attend to them*”. Em: Rhetoric Society Quarterly 46(3), pp. 199–206, publicado em 02 de junho de 2016. Disponível em: <https://doi.org/10.1080/02773945.2016.1171689> Acesso em: 20 de março de 2021.
- [5] Maeng, D. et al “*Can telemonitoring reduce hospitalization and cost of care? A health plan's experience in managing patients with heart failure*”. Em: Population Health Management 17(6). PMID: 24865986, pp. 340–344., dezembro 2014. DOI: 10.1089/pop.2013.0107. Disponível em: <https://pubmed.ncbi.nlm.nih.gov/24865986/> Acesso em: 20 de março de 2021.
- [6] Lu, Tsung-Chien et al. “*Healthcare Applications of Smart Watches. A Systematic Review*”. Em: Applied Clinical Informatics 7, pp. 850–869. 14 de setembro de 2016. Disponível em: <https://www.thieme-connect.de/products/ejournals/pdf/10.4338/ACI-2016-03-R-0042.pdf> Acesso em: 20 de março de 2021.