Pulseira para Gerenciamento de Medicamentos

Vinícius Pereira da Silva[1], Ricardo Barroso Leite[2],

[1]Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - Campus Hortolândia (IFSP).
13183-250 – Hortolândia – SP – Brasil

vinicius.silvap@outlook.com, ricardo.leite@ifsp.edu.br

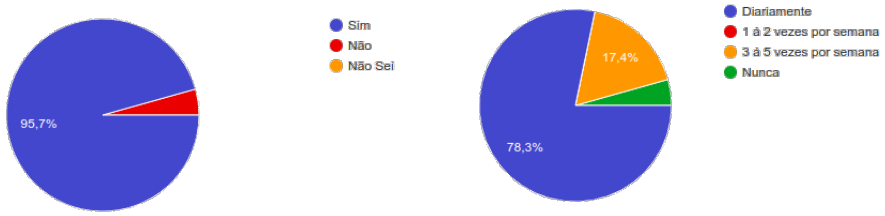
Abstract. *The majority of the elderly people needs to make use of medicines in specific hours of the day, but they suffer from the oblivion caused for an advanced age, that prevents them to remember the schedules and which remedies must take during the day. To solve this problem, this project aims to develop an electronic bracelet for the medical management that will be used for the aged one. Together, an application for mobile devices will be developed managed by a third part, with the purpose to control the schedules and medicines. For the project aspects of usability and low cost will be analyzed, that is critical for the considered public, providing an open and personalized environment, that facilitates future research of the subject.*

Resumo. *A maioria dos idosos precisa fazer o uso de medicamentos em determinados horários do dia mas sofrem com o esquecimento causado pela idade avançada, que os impossibilita de lembrar os horários e quais remédios devem tomar durante o dia. Para resolver esse problema, este projeto visa desenvolver uma pulseira eletrônica para o gerenciamento de medicamentos que será usada pelo idoso. Em conjunto, será desenvolvida uma aplicação para dispositivos móveis administrada por um terceiro, com a finalidade de controlar os horários e medicamentos. Para o projeto serão analisados aspectos de usabilidade e baixo custo, que são críticos para o público considerado, além de propiciar um ambiente aberto e customizável, que facilite pesquisas futuras dentro do tema.*

1. Introdução

Nos últimos tempos nossa sociedade vem sofrendo um grande avanço tecnológico em todos os pilares da civilização e com isso a expectativa de vida das pessoas aumentou expressivamente, o principal elemento dessa evolução é a medicina que com o tempo evoluiu muito e hoje proporciona diversos tratamentos e prevenções a inúmeras doenças, deixando o ser humano mais apto a lidar com esses males e por consequência evitando mortes. Vemos que a população está mais velha, há uma maior presença dos idosos na nossa sociedade o que força o mundo a se adaptar a esse novo contexto. Levando em conta esse cenário é comum essas pessoas sofrerem com os desgastes físicos e mentais causados pela idade avançada, as consequências disso é o uso de medicamentos que visa suprir tais degastes, entretanto surge um problema, o esquecimento dos idosos para com os horários dos medicamentos, esquecer quando e qual ingerir.

Uma pesquisa realizada com 23 pessoas que possuem na sua família idosos que precisam fazer o uso de medicamentos mostra que, 95,7% desses idosos tomam medicamentos de forma contínua sem interrupção sendo 78,3% diariamente, como podemos ver na figura 1 os gráficos com alguns dados levantados pela pesquisa.



(a) Necessidade de medicamentos (b) Frequência de uso dos medicamentos

Figura 1. Pesquisa sobre o uso de medicamentos por idosos.

A Internet das Coisas pode proporcionar a solução desse problema de uma forma simples e não prejudicial ou incômoda para os idosos. ”As aplicações da Internet das Coisas podem melhorar a qualidade de vida especialmente para os idosos. ”(PRESSER et al., 2013, p.19). Essa solução já foi pensada e desenvolvida antes, em uma entrevista ao G1 (Globo), dois alunos, André Rocha Soares e Pedro Henrique Moraes Guizardi mostraram e comentaram sobre o seu projeto voltado a esse problema, segundo Soares e Guizardi (2016, p.1).

> A pulseira é programada por um profissional da saúde, e ela permite que cada medicamento tenha uma cor de identificação, e essa cor é a mesma do pote onde está o remédio. Na hora que o paciente deve tomar o remédio, ela apita, vibra e mostra a cor que o profissional programou.

Embora seja uma ótima solução pessoas idosas teriam dificuldades, os inúmeros potes além da própria pulseira são facilmente esquecidos em cômodos da casa ou até mesmo em outros lugares. Os idosos necessitam de uma tecnologia barata e com toda a tecnologia que temos hoje em dia podemos produzir produtos acessíveis utilizando hardwares e componentes de baixo custo. ”Espero que ela custe bem pouco, para atender, sobretudo, os pacientes do SUS”(SOARES; GUIZARDI, 2016, p.1).

2. Referencial Teórico

Para uma melhor compreensão do projeto e como ele será desenvolvido, as subseções seguintes tem como objetivo explicar os principais componentes, produtos e serviços utilizados. A escolha de ferramentas e produtos usados no projeto foi definida por meio de pesquisas de mercado e experimentos em laboratório.

2.1. NodeMCU

O NodeMCU é um firmware e kit de desenvolvimento que permite a programação de protótipos para a Internet das Coisas (IoT). O firmware utiliza o paradigma event-driven para facilitar o desenvolvimento de aplicações que necessitem acesso à Internet. Além disso, integra módulos de GPIO, 1-Wire, I2C, SPI, PWM, ADC, entre outros, para facilitar o manuseio de módulos baseados no chip ESP8266. (NODEMCU TEAM, 2017). A figura 2 mostra toda a pinagem do NodeMCU.

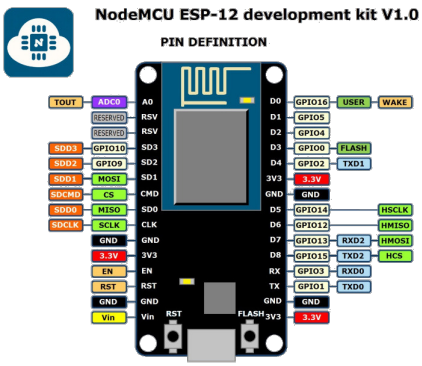


Figura 2. NodeMCU
Fonte: ARDUINING.COM

2.2. ESP8266

ESP8266 é o nome de um sistema embarcado projetado pela Espressiff Systems. O ESP8266 se define como uma solução de redes Wi-Fi autossuficiente se oferecendo como uma ponte entre um microcontrolador pré-existente e a rede com sinal Wi-Fi e que também é capaz de executar aplicações de maneira independente. (KOLBAN, 2015). Observa-se na figura 3 a versão mais popular da família ESP que é o ESP-12E, ele é

integrado juntamente ao NodeMCU - ESP8266.

Figura 3. ESP-12E - Uma das versões mais populares e flexíveis do ESP8266
Fonte: KOLBAN, 2015

2.3. Display OLED

OLED (Organic Light Emitting Diodes) é uma tecnologia de emissão de luz plana, obtida colocando uma série de camadas orgânicas finas entre dois condutores. Quando a corrente elétrica é aplicada, a luz brilhante é emitida. OLEDs são displays que não requerem iluminação de fundo e, portanto, são mais finos e mais eficientes do que os LCD (Liquid Crystal Display) que exigem uma luz de fundo branca. (OLED INFO, 2017). A figura 4 evidencia o display OLED utilizado no projeto.

Figura 4. Display OLED - 128x32 I2C

2.4. Android Studio

O Android Studio é uma IDE de desenvolvimento baseada no IntelliJ Platform, foi desenvolvida pela Google para padronizar e disponibilizar de forma simples e completa todos os recursos presente no sistema operacional Android, contando com uma ferramenta Drag and Drop para a criação de layouts de forma eficiente e fácil. Os aplicativos podem ser desenvolvido usando a linguagem de programação Java ou Kotlin. (ANDROID STUDIO, 2021).

2.5. Arduino

Arduino é uma plataforma eletrônica de código aberto baseada em hardware e software de fácil uso. As placas possuem um microcontrolador que recebe informações de entrada, realiza as instruções programadas e retorna um dado, podendo acionar um motor, um LED, publicar em sites. Para programar a placa usa-se a linguagem de programação Arduino (baseada em Wiring) semelhante ao C++ e utiliza o Arduino Software (IDE) que é baseado em Processing. (ARDUINO, 2021)

2.6. Firebase

O Firebase é um SaaS (Software as a Service) que oferece uma solução completa de softwares e serviços em nuvem para diversas plataformas, tais como, banco de dados em tempo real, API de autenticação, armazenamento de arquivos, dentre muitos outros. É um serviço mantido pela Google e visa facilitar o desenvolvimento de aplicações, remover a responsabilidade da empresa ou desenvolvedor de criar tudo sozinho e atribuir um aspecto de profissionalismo até em pequenos projetos. (GOOGLE FIREBASE, 2021).

2.7. Firebase Realtime Database

É um banco de dados NoSQL hospedado na nuvem. Os dados são armazenados como JSON e são sincronizados em tempo real com os clientes conectados mesmo em multiplataforma, faz parte da suíte Firebase. (GOOGLE FIREBASE, 2021).

2.8. Java

É uma linguagem de programação orientada a objetos e plataforma computacional lançada pela primeira vez pela Sun Microsystems em 1995. Amplamente utilizada para o desenvolvimento de sites e aplicativos. O Java é rápido, seguro e confiável. De laptops a datacenters, telefones celulares à Internet. (JAVA, 2021).

2.9. Bibliotecas Arduino

Para o projeto foi utilizado bibliotecas especificas para cada função da pulseira, essa seção irá detalhar cada uma.

- FS.h - utilizada para que as informações dos alarmes fiquem salvas na memória.
- ESP8266WiFi.h - possibilita a conexão com a interner do ESP8266 com redes sem fio.
- ESP8266TimeAlarms.h - gerencia a criação dos alarmes de forma simplificada.
- ArduinoJson.h - usada para manipular arquivos JSON.
- ESP8266WebServer.h - gera um servidor WEB.
- DNSServer.h - cria um servidor DNS.
- WiFiManager.h - Utilizada junto com o webserver para criar uma página web e obter os parâmetros da conexão wireless.

- Wire.h - Usada para a comunicação utilizando I2C.
- Adafruit_GFX.h - utilizada pelo display SSD1306.
- Adafruit_SSD1306.h - administra os caracteres impressos no display SSD1306.
- FirebaseArduino.h - faz a conexão com o banco de dados Firebase.
- EEPROM.h - Utilizada para gravar dados na memória de forma permanente, mesmo após um desligamento.

3. Metodologia

3.1. Desenvolvimento Incremental

”O desenvolvimento incremental é baseado na ideia de desenvolver uma implementação inicial, expô-la aos comentários dos usuários e continuar por meio da criação de várias versões até que um sistema adequado seja desenvolvido.”(SOMMERVILLE, 2011, p. 21). A Figura 5 ilustra o as etapas e o ciclo seguido no desenvolvimento incremental.

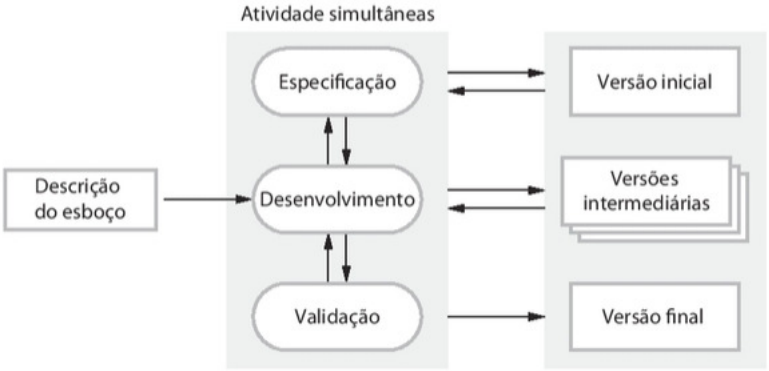


Figura 5. Processo do Modelo Incremental
Fonte: SOMMERVILLE, 2011, p. 22

O modelo incremental é divido em várias atividades de desenvolvimento, a cada atividade o resultado é entregue ao cliente, com base no seu feedback o desenvolvimento pode sofrer mudanças.

> Desenvolvimento incremental reflete a maneira como resolvemos os problemas. Raramente elaboramos uma completa solução do problema com antecedência; geralmente movemo-nos passo a passo em direção a uma solução recuando quando percebemos que cometemos um erro. Ao desenvolver um software de forma incremental, é mais barato e mais fácil fazer mudanças no software durante seu desenvolvimento. (SOMMERVILLE, 2011, p. 22).

Primeiro cria-se uma descrição do esboço do projeto, uma vez definido, passa-se para as especificações necessárias. O desenvolvimento é iniciado e enviado para validação, após o feedback o processo se repete, gerando várias versões do produto até chegar em sua versão final.

3.2. Prototipação

Para complementar o modelo incremental no desenvolvimento do projeto, foi utilizado a prototipação, tanto no hardware como no software. A prototipação é utilizada desde o começo do projeto, ajudando até mesmo nas definições de requisitos, quando os mesmos estão nebulosos. O modelo funciona semelhante ao incremental, em etapas, são desenvolvidos vários protótipos, cada um focado em requisitos específicos, ao final de cada criação são enviados para avaliação e um feedback do cliente.

> Frequentemente, o cliente define uma série de objetivos gerais para o software, mas não identifica, detalhadamente, os requisitos para funções e recursos. Em outros casos, o desenvolvedor se encontra inseguro quanto à eficiência de um algoritmo, quanto à adaptabilidade de um sistema operacional ou quanto à forma em que deve ocorrer a interação homem-máquina. Em situações como essas, e em muitas outras, o paradigma da prototipação pode ser a melhor abordagem. (PRESSMAN, MAXIM, 2016, P. 45).

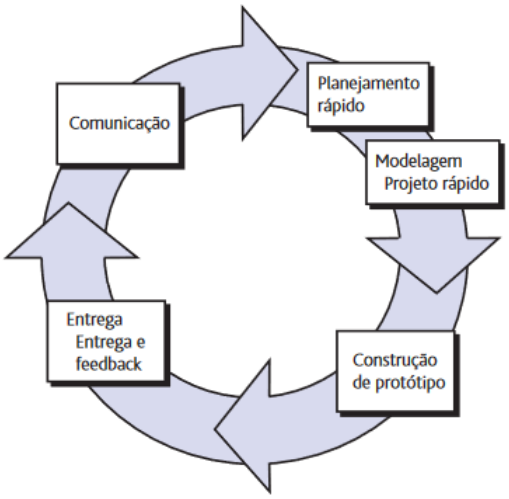


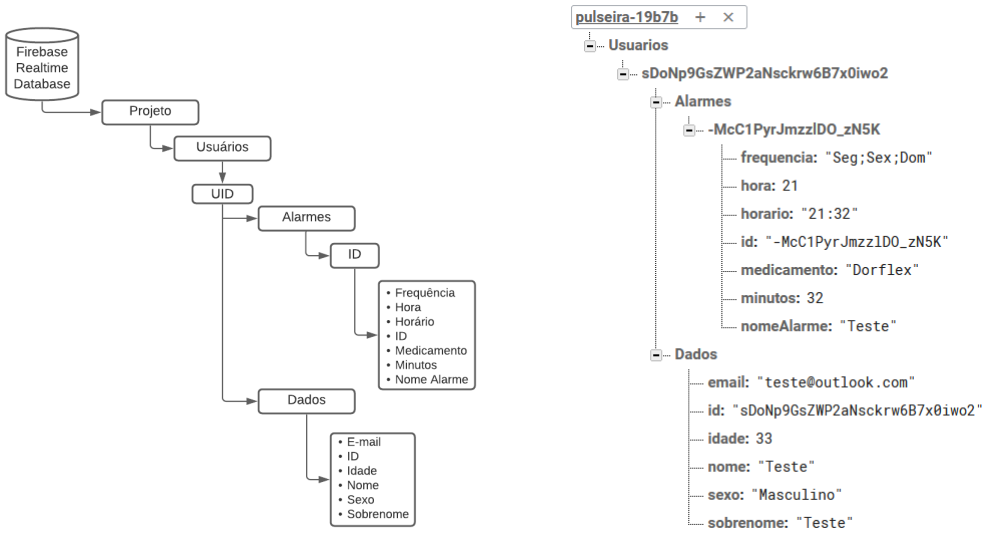
Figura 6. Processo do Modelo de Prototipação
Fonte: PRESSMAN, MAXIM, 2016, p. 45

A figura 6 ilustra todas as etapas da prototipação, começando por comunicação, momento em que os requisitos são levantados e os objetivos do projeto são definidos. Logo após começa a etapa de planejamento rápido e modelagem do projeto rápido, em que uma prototipação é planejada rapidamente e acontece uma modelagem do projeto para representar alguns aspectos do hardware e software ao cliente, ao final desse processo de modelagem, começa a construção de fato do protótipo e a sua entrega para o cliente, que finaliza com um feedback para a equipe.

3.3. Diagrama Banco de Dados

O banco de dados utilizado no projeto foi o Firebase Realtime Database, um banco NoSQL. Sua característica principal é não possuir relacionamento entre entidades e ser

em tempo real, as alterações são feitas imediatamente, o banco trabalha com chave e valor, cada informação (valor) é atrelada com uma chave. O modelo de armazenamento é o JSON (JavaScript Object Notation). Na figura 7 mostra a estrutura geral do banco.
O nó raiz da base é o projeto, abaixo dele temos o grupo chamado usuários, em que todos os usuários do aplicativo são armazenados, cada um possui um UID (User Identification), para todo UID é criado um grupo de alarmes e outro de dados. O grupo alarmes armazena e separa todos os alarmes criados pelo usuário por um ID único gerado automaticamente e seus atributos, sendo eles, frequência, horário, medicamento e o nome personalizado do alarme. No grupo dados é armazenado todas as informações pessoais que aplicativo coleta do usuário, sendo eles, e-mail, UID, idade, nome completo e sexo. .



(a) Atributos e separação de dados

(b) Exemplo com dados

Figura 7. Estrutura banco de dados

3.4. Arquitetura do projeto

Para a comunicação entre ambas as partes do projeto, pulseira e aplicativo Android, faz-se necessário serviços intermediadores para auxiliar na entrega de informações e garantir as funcionalidades da pulseira eletrônica. A figura 8 ilustra como um horário de um determinado medicamento chega até a pulseira.

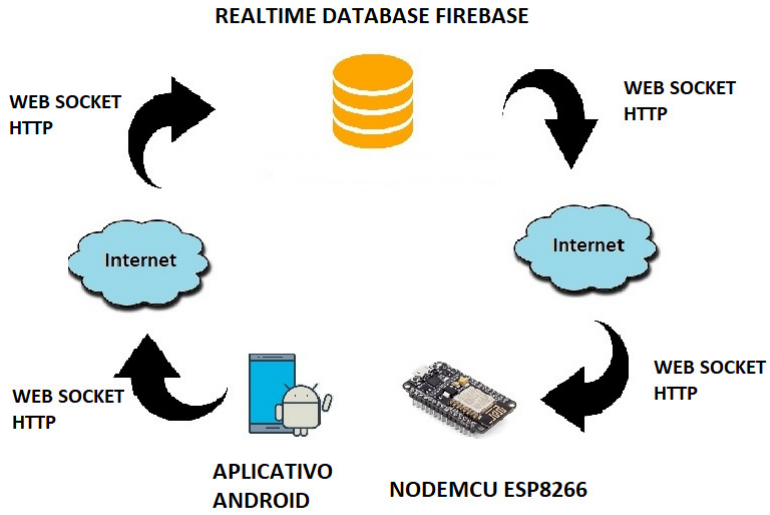


Figura 8. Arquitetura do Projeto

Primeiro passo - usuário utiliza um aplicativo em seu smartphone Android em que será criado uma conta com os seus dados pessoais e alarmes definidos pelo próprio usuário. Todas as informações obtidas pelo aplicativo são enviadas em tempo real para o banco de dados Firebase Realtime Database utilizando a internet como meio de comunicação.

Segundo passo - o banco de dados recebe os dados enviados pelo aplicativo e armazena em uma estrutura NOSQL utilizando JSON, que agrupa os dados por chave e valor. Com todas as informações organizadas, nesse momento o banco está pronto para atender as solicitações enviadas para ele via aplicativo ou pulseira (hardware) utilizando a internet, caso a comunicação com a internet não estiver ativa, o banco faz sua atualização posteriormente.

Terceiro passo - na outra ponta da arquitetura tem-se uma placa, um hardware de baixo custo, chamado NodeMCU, nessa placa estão ligados todos os periféricos necessários para a pulseira funcionar, exemplo, display de OLED que mostra as informações para o usuário. A pulseira utiliza uma conexão com a internet via rede sem fio para receber os dados dos alarmes que devem ser criados, definidos pelo usuário no aplicativo. Uma vez que um alarme foi criado, a pulseira notifica o usuário mostrando o nome do medicamento no horário definido. Para manter os alarmes sempre atualizados a pulseira faz uma checagem de dados a cada cinco minutos na base de dados.

3.5. Caso de uso - aplicativo para dispositivos móveis

A figura 9 demonstra as funcionalidades presentes no aplicativo para dispositivos móveis. O usuário poderá, efetuar seu login no sistema, afim de ter uma experiência

personalizada, recuperar sua senha caso tenha esquecido, cadastrar medicamentos e seus respectivos horários e manter atualizado o seu próprio cadastro.

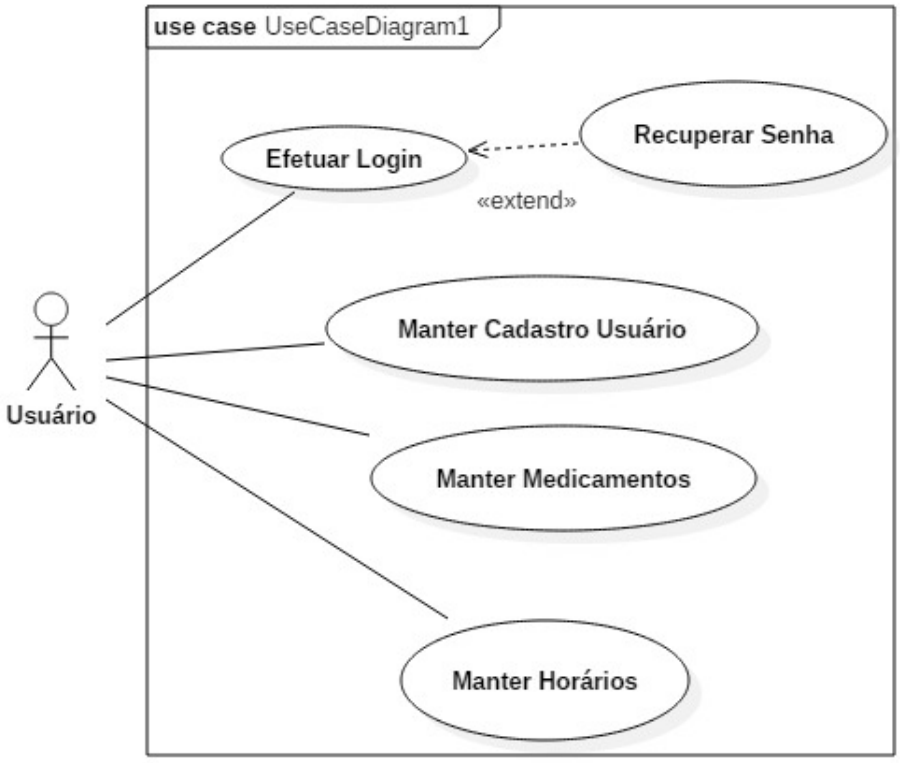


Figura 9. Caso de uso - Aplicativo

Para que todas as ações dentro do aplicativo fiquem disponíveis para uso uma conexão com a internet é necessária, na figura 9 podemos ver o caso de uso e abaixo temos o fluxo básico que o usuário terá em cada etapa.

- Efetuar Login - o usuário tem que fornecer na tela principal seu e-mail e senha cadastrados anteriormente no aplicativo para que o mesmo possa acessar a área exclusiva.
- Recuperar Senha - opção presente na tela principal, utilizada para recuperar o acesso ao aplicativo quando o usuário esquecer sua senha cadastrada, basta fornecer o e-mail de cadastro que será liberado a possibilidade de criar uma nova senha.
- Manter Cadastro Usuário - no primeiro acesso o usuário deverá criar seu cadastro em uma tela dedicada que pode ser acessada através da tela principal, sendo um usuário cadastrado o mesmo pode alterar suas informações pessoais a qualquer momento ou até mesmo excluir sua conta.
- Manter Medicamentos - com o usuário autenticado no aplicativo libera as funções

de adicionar, remover, atualizar e consultar os medicamentos cadastrados nos alarmes.
- Manter Horários - com o usuário autenticado no aplicativo libera as funções de adicionar, remover, atualizar e consultar os horários cadastrados nos alarmes.

4. Desenvolvimento

Nessa seção do projeto será exemplificado e demonstrado os protótipos desenvolvidos.

4.1. Protótipo pulseira eletrônica

Na figura 10 está exemplificado um protótipo da pulseira, em que foi utilizado o NodeMCU, display OLED e a protoboard. Todos os componentes estão interligados por meio de fios, jumpers, acoplados na protoboard e nas saídas da placa NodeMCU. O principal componente é a placa NodeMCU, responsável em alimentar todos os outros componentes, tanto em energia quanto em dados e instruções de comportamento, é através dela que a pulseira terá acesso aos medicamentos e seus respectivos horários presentes no banco de dados. Para mostrar ao idoso o nome do medicamento e o horário é utilizado um display de OLED, que possui alto brilho e tem uma boa nitidez e resolução, tornando possível enxergar durante o período noturno.

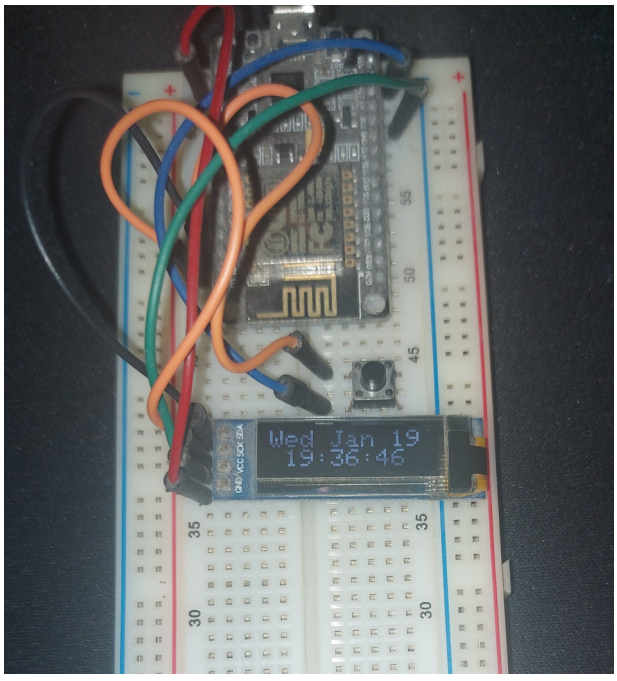


Figura 10. Protótipo pulseira eletrônica

4.2. Software Embarcado Arduino

Nessa seção será mostrado partes importantes do código-fonte utilizado na pulseira (hardware).

4.2.1. Estrutura de dados

Para tratar das informações vindas do banco de dados, foi criado uma Struct com todos os atributos de um alarme definido pelo usuário para ser usado localmente, juntamente com essa estrutura temos uma função que é chamada toda vez que criamos um alarme, ela seta os dados obtidos via passagem de parâmetro em variáveis locais e retorna um objeto alarme com todos os atributos. A figura 11 mostra esse trecho do código.

```
typedef struct {
  int hora;
  int minuto;
  String frequencia;
  String medicamento;
  int alarme_id;
} Alarme;

Alarme SetAlarme(int hora, int minuto, String frequencia, String medicamento, int id) {
  Alarme alarme;
  alarme.hora = hora;
  alarme.minuto = minuto;
  alarme.frequencia = frequencia;
  alarme.medicamento = medicamento;
  alarme.alarme_id = id;
  return alarme;
}
```

Figura 11. Estrutura de dados

4.2.2. Função escreverDisplay()

```
//Função para escrever na tela OLED da pulseira.
void escreverDisplay(String texto) {
  display.clearDisplay();
  display.setTextSize(2);
  display.setTextColor(WHITE);
  display.setCursor(0, 0);
  display.println(texto);
  display.display();
}
```

Figura 12. Função escreverDisplay()

Na figura 12 vemos a função responsável pela impressão dos textos no display OLED da pulseira, recebe como parâmetro um texto em formato de string. Quando chamada faz uma limpeza no display, evitando resíduos, determina o tamanho da fonte, a cor do texto e posiciona ao centro.

4.2.3. Função setupFirebase()

```
void setupFirebase(){
 //Conectar ao banco de dados Firebase e passar o caminho para consultar os alarmes
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  const ArduinoJson::JsonObject& usuario_alarmes =
    Firebase.get("/Usuarios/"+(String)id_usuario+"/Alarmes")
    .getJsonVariant().asObject();
  int aux_hora;
  int aux_minuto;
  String aux_frequencia;
  String aux_medicamento;
  //Sequência de FOR para percorrer todos os alarmes do usuário no Firebase.
  for (auto kv : usuario_alarmes) {
    const ArduinoJson::JsonObject& dados_alarme = kv.value.asObject();
    dados_alarme.prettyPrintTo(Serial);
    aux_hora = dados_alarme.get<int>("hora");
    aux_minuto = dados_alarme.get<int>("minutos");
    aux_frequencia = dados_alarme.get<String>("frequencia");
    aux_medicamento = dados_alarme.get<String>("medicamento");
    grupo_alarmes[contador_grupo_alarmes] =
      SetAlarme(aux_hora, aux_minuto, aux_frequencia, aux_medicamento, aux_id);
    criarAlarme(grupo_alarmes[contador_grupo_alarmes]);
    contador_grupo_alarmes++;
    auxiliar++;
    aux_id++;
  }
  contador_grupo_alarmes = 0;
}
```

Figura 13. Função setupFirebase()

A figura 13 mostra o código da função chamada setupFirebase, através dela a pulseira se comunica com o banco e obtêm as informações necessárias para criar um alarme. Inicia a conexão com o comando Firebase.begin, com a conexão estabelecida, o comando Firebase.get obtêm um arquivo do tipo JSON com os dados cadastrados para o usuário atual do sistema, logo após, um foreach percorre todos os dados dentro do JSON e cria localmente os alarmes necessários.

4.2.4. Sincronização do Banco de Dados

Para garantir que a pulseira sempre tenha os dados atualizados em tempo real, dentro da função loop() do Arduino foi adicionado duas condicionais de sincronia, se a pulseira estiver com uma conexão de internet ativa e tiver passado um minuto e meio ou mais do relógio interno, específico para essa tarefa, chama-se a função setupFirebase() que atualiza os dados internos com o banco de dados, após finaliza o relógio interno é zerado. A figura 14 mostra o código dessa função.

```
if(WiFi.status()== WL_CONNECTED ) {
      if((millis() - millisFirebase) > 83000){
        setupFirebase();
        millisFirebase = millis();
      }
}
```

Figura 14. Condicionais de sincronia do banco de dados

4.3. Aplicativo para Android

4.3.1. Classe ConfiguracaoFirebase

Na figura 15 é ilustrado a classe responsável pela conexão com o Firebase, seja para manipular dados no Realtime Database ou para autenticar usuário. Quando um método ou uma classe necessita de uma referencia para o banco de dados é feito uma requisição para a ConfiguracaoFirebase que retorna uma instância com o nome referenciaFirebase, caso seja requisitado uma autenticação para ConfiguracaoFirebase é retornado uma instância de autenticação.

```
public class ConfiguracaoFirebase {

    private static FirebaseAuth autenticacao;
    private static DatabaseReference referenciaFirebase;

    public static DatabaseReference getFirebase() {
        if (referenciaFirebase == null) {
            referenciaFirebase = FirebaseDatabase.getInstance().getReference();
        }
        return referenciaFirebase;
    }

    public static FirebaseAuth getFirebaseAutenticacao() {
        if (autenticacao == null) {
            autenticacao = FirebaseAuth.getInstance();
        }
        return autenticacao;
    }
}
```

Figura 15. Classe ConfiguracaoFirebase

4.3.2. Botão salvar alarme

Para criar um alarme dentro do aplicativo, precisa ir até o formulário de cadastro e ao final dele apertar no botão de salvar, quando essa ação é detectada pelo listener btnSalvar os dados do formulários passam por duas validações, para evitar informações incorretas, após a validação, os dados do formulário são enviados para um objeto local do tipo Alarme e é atrelado ao UID do usuário que está realizando o cadastro e por fim esse conjunto de dados é enviado para o banco de dados. A figura 16 mostra o código dessa função.

```
btnSalvar.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        String keyIntent;
        alarme.setHorario(tpHorario.getHour() + ":" + tpHorario.getMinute());
        alarme.setHora(tpHorario.getHour());
        alarme.setMinutos(tpHorario.getMinute());

        definirFrequencia();
        if (editNome.getText().toString().equals("")) {
            editNome.setError("Campo obrigatório!");
        } else if (editMedicamento.getText().toString().equals("")) {
            editMedicamento.setError("Campo obrigatório!");
        } else if (selecionados.isEmpty()) {
            Toast.makeText( context: CriarAlarmeActivity.this,
                    text: "Marque pelo menos um dia!", Toast.LENGTH_LONG).show();
        } else {
            alarme.setNomeAlarme(editNome.getText().toString());
            alarme.setMedicamento(editMedicamento.getText().toString());
            alarme.setFrequencia(selecionados);
            autenticacao = ConfiguracaoFirebase.getFirebaseAutenticacao();
            final String user = autenticacao.getCurrentUser().getUid();

            if (activityPai.equals("com.tcc.pulseira.pulseira.AlarmeAdapter$2")) {
                keyIntent = getIntent().getStringExtra( name: "key");
                alarme.setId(keyIntent);
                cadastrarAlarme(keyIntent, user);
            } else {
                keyIntent = "vazia";
                referenciaFirebase = ConfiguracaoFirebase.getFirebase()
                        .child("Usuarios").child(user).child("Alarmes").push();
                alarme.setId(referenciaFirebase.getKey());
                cadastrarAlarme(keyIntent, user);
```

Figura 16. Botão Salvar Alarme

5. Resultados

5.1. Objetivos

Objetivos propostos no início do projeto:

- Promover o cenário de IoT (Internet of Things);
- Alertar o momento certo de consumir um medicamento;

5.1.1. Promover o cenário de IoT (Internet of Things)

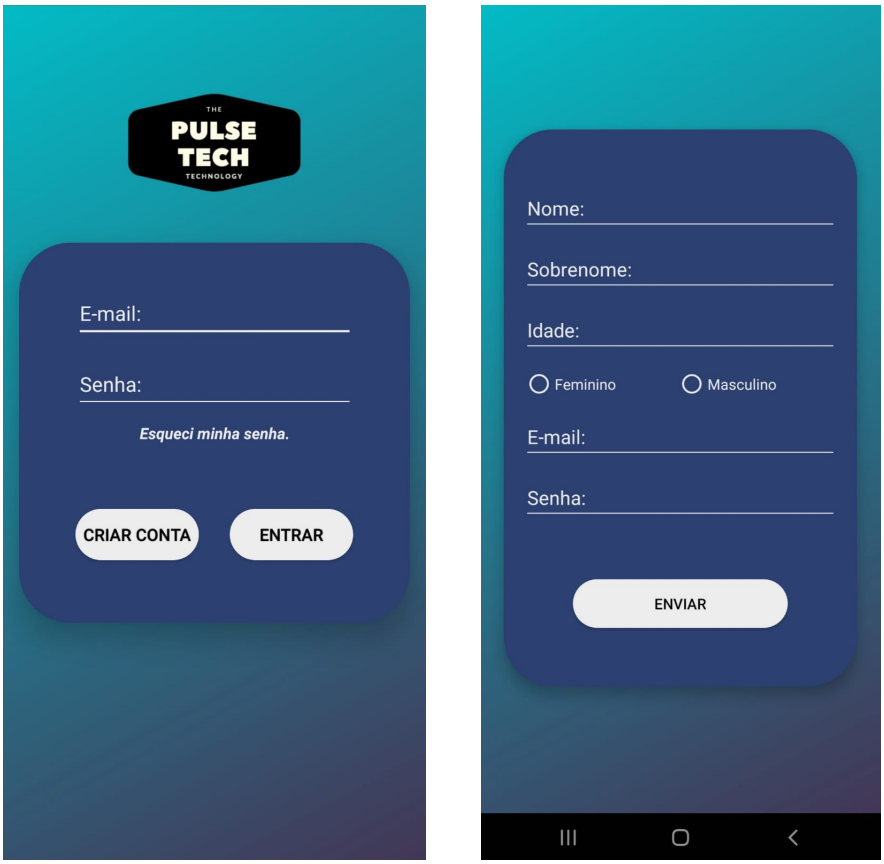
O projeto utilizou várias ferramentas, hardwares e serviços específicos para IoT e todo seu conteúdo ficará disponível para consulta dentro do site do IFSP, dessa forma, fomentando a comunidade de IoT, objetivo alcançado.

5.1.2. Alertar o momento certo de consumir um medicamento

Com o desenvolvimento de um aplicativo para Android e um conjunto de hardwares (pulseira) foi possível criar alarmes personalizados e chegar até o objetivo de avisar o dia e horário em que um medicamento deve ser consumido.

5.2. Aplicativo

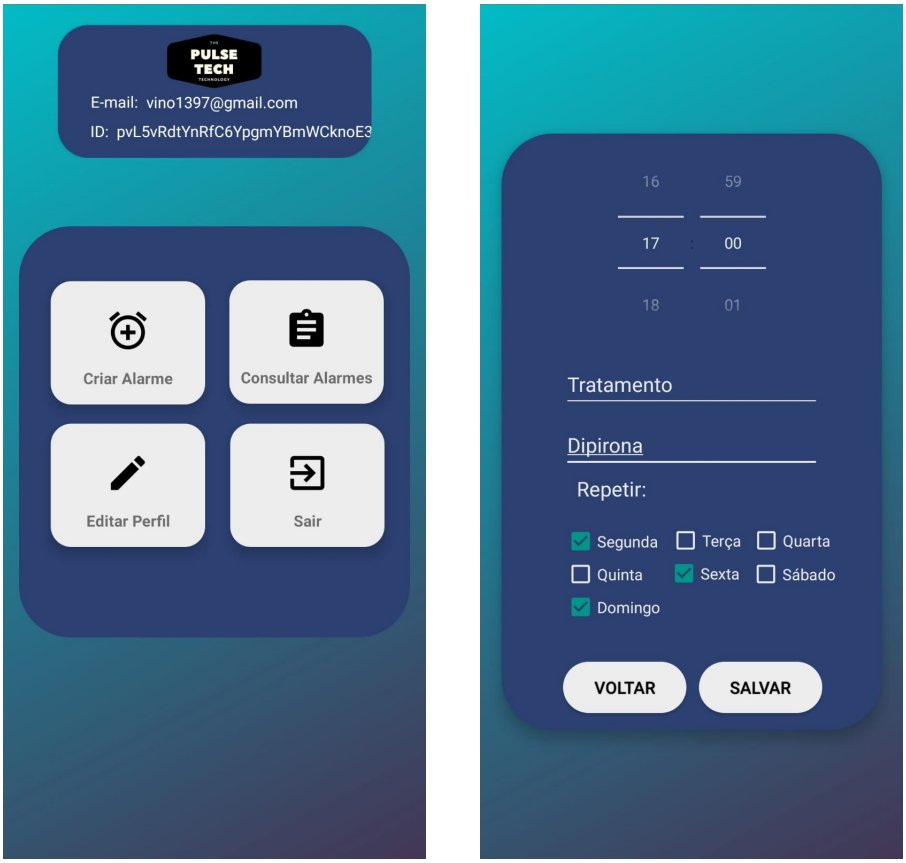
Na figura 17 está representado a tela inicial do aplicativo e a tela para cadastrar os usuários na aplicação.



(a) Tela Inicial (b) Cadastrar novo usuário

Figura 17. Tela inicial e cadastro de usuários.

A figura 18 mostra o dashboard da aplicação, a tela principal para o usuário, juntamente com a tela de criação de alarmes.



(a) Dashboard (b) Criar alarme

Figura 18. Dashboard e cadastro de alarmes.

5.3. Custos

- NodeMCU ESP8266-12 V2 - R$20,00
- Chave Táctil 6x6x5mm 4 Terminais - R$0,20
- Display OLED Branco 128x32 I2C Pixel 0.91 - R$25,00
- 10 Cabos Jumper - R$10
- Protoboard 830 Pontos - R$15,00
- Cabo USB-A x Micro USB, 1 Metro - R$5,00
- Plataforma Firebase - Plano gratuito
- Total - R$75,20

6. Conclusão

Projeto concluído, atingindo seus objetivos conforme citado na seção de resultados. Essa solução foi pensada e executada em laboratório, não houve testes reais com o público-alvo devido ao escopo do projeto, fica como trabalho futuro obter a liberação do comitê de ética para desenvolver testes com os idosos e obter dados realísticos. Todo o hardware utilizado foi montado em uma protoboard, um local para testes de circuitos, uma possível sequência do projeto é imprimir um circuito elétrico em placa definitiva e acoplar em uma pulseira desenvolvida na impressora 3D.

Referências

ARDUINING. **NodeMCU ESP-12 Development Kit V1.0.** Disponível em: <https://arduining.com/2015/08/15/nodemcu-esp-12-development-kit-v1-0/>. Acesso em: 20 nov. 2017.

GOOGLE FIREBASE. **Firebase.** Disponível em: <https://firebase.google.com/?hl=pt-br>. Acesso em: 27 jun. 2021.

GUIZARDI, P. H. M.; SOARES, A. R. **Estudantes criam pulseira que avisa a hora de tomar remédio.** Disponível em: <http://g1.globo.com/espirito-santo/noticia/2016/03/estudantes-criam-pulseira-que-avisa-hora-de-tomar-remedio-no-es.html>. Acesso em: 21 nov. 2017.

ANDROID STUDIO. **Conheça o Android Studio.** Disponível em: <https://developer.android.com/studio/intro?hl=pt-br/>. Acesso em: 27 jun. 2021.

KOLBAN, Neil. **Kolban's Book on ESP8266.** Texas, USA. 2015.

GOOGLE FIREBASE. **Firebase Realtime Database.** Disponível em: <https://firebase.google.com/docs/database>. Acesso em: 27 jun. 2021.

NODEMCU TEAM. **NodeMCU - An open-source firmware based on ESP8266 wifi-soc.** Disponível em: <http://nodemcu.com/index_en.html>. Acesso em: 21 nov. 2017.

OLED INFO. **The OLED Experts** Disponível em: <https://www.oledinfo.com/introduction>. Acesso em: 21 nov. 2017.

PRESSER, M, et al. **Internet das Coisas.** Disponível em: <https://iotcomicbook.files.wordpress.com/2013/10/iot_comic_book_special_br.pdf >. Acesso em: 21 nov. 2017.

SOMMERVILLE, Ian. **Engenharia de Software.** 9. ed. São Paulo: Pearson Prentice Hall, 2011.

Tecnologia e os Idosos. Disponível em: <https://goo.gl/forms/SEb7rxWp0DGjxEa72>. Acesso em: 21 nov. 2017.

ARDUINO. **What is Arduino?** Disponível em: <https://www.arduino.cc/en/Guide/Introduction>. Acesso em: 27 jun. 2021.
JAVA. **O que é a Tecnologia Java e porque preciso dela?** Disponível em: <https://www.java.com/pt-BR/download/help/whatis_java.html>. Acesso em: 27 jun. 2021.

OLIVEIRA, Sérgio de. **Internet das Coisas com ESP8266, Arduino e Raspberry PI.**. [São Paulo]: Novatec, 2017.236 p.

JAVED, Adeel. **Criando projetos com Arduino para a Internet das Coisas.**. [São Paulo]: Novatec, 2017.275 p.

ANGRY TOOLS. **Android Button Maker** Disponível em: <https://angrytools.com/android/button/>. Acesso em: 27 jun. 2021.

MEDIUM. **Listas com RecyclerView**. Disponível em: <https://medium.com/android-dev-br/listas-com-recyclerview-d3f41e0d653c>. Acesso em: 27 jun. 2021.

ALURA. **Criando listas com RecyclerView**. Disponível em: <https://www.alura.com.br/artigos/criando-listas-com-recyclerview>

PRESSMAN, Roger S; MAXIM, Bruce R **Engenharia de Software. Uma abordagem profissional** 8. ed. Porto Alegre: AMGH, 2016.

Documento Digitalizado Público

Artigo final correspondente ao Trabalho de Conclusão de Curso

Assunto: Artigo final correspondente ao Trabalho de Conclusão de Curso
Assinado por: Ricardo Leite
Tipo do Documento: Relatório
Situação: Finalizado
Nível de Acesso: Público
Tipo do Conferência: Documento Digital

Documento assinado eletronicamente por:

- **Ricardo Barroso Leite**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 17/03/2022 14:18:39.

Este documento foi armazenado no SUAP em 17/03/2022. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifsp.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 918672
Código de Autenticação: 979626a845