UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

DEPARTAMENTO ACADÊMICO DE ELETRÔNICA

CURSO DE ENGENHARIA ELETRÔNICA

ERIK BENNATON ROLIM MARCON

IGOR IVAN GAUDEDA

**DISPOSITIVO NÃO INVASIVO PARA O MONITORAMENTO DO ESTADO DE SAÚDE DE IDOSOS**

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA

2017

ERIK BENNATON ROLIM MARCON

IGOR IVAN GAUDEDA

**DISPOSITIVO NÃO INVASIVO PARA O MONITORAMENTO DO ESTADO DE SAÚDE DE IDOSOS**

Trabalho de Conclusão de Curso de graduação, apresentado ao Departamento Acadêmico de Eletrônica da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para a obtenção do título de Bacharel em Engenharia Eletrônica

Orientador: Prof. Msc. Daniel Rossato de Oliveira

CURITIBA

2017

ERIK BENNATON ROLIM MARCON

IGOR IVAN GAUDEDA

**DISPOSITIVO NÃO INVASIVO PARA O MONITORAMENTO DO ESTADO DE SAÚDE DE IDOSOS**

Este Trabalho de Conclusão de Curso foi julgado e aprovado como requisito parcial para obtenção do título de Engenheiro Eletrônico pela Universidade Tecnológica Federal do Paraná.

Curitiba, ?? de ?? de 2017.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. ??

Coordenador de Curso

Departamento Acadêmico de Eletrônica

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. ??

Responsável pelo Trabalho de Conclusão de Curso

Departamento Acadêmico de Eletrônica

**BANCA EXAMINADORA**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. ??

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. Msc. Daniel Rossato de Oliveira

Orientador

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Prof. ??

**AGRADECIMENTOS**

Agradecemos ao nosso orientador Prof. Msc. Daniel Rossato de Oliveira, pelo seu apoio e ajuda durante o desenvolvimento do projeto e da elaboração desta monografia. Outrossim, pela sua cobrança, especialmente nestes últimos meses, dado que, sem seu incentivo, supervisão e insistência, jamais teríamos terminado este trabalho dentro do prazo.

// trocar frase

*Tudo dito e nada feito – fito e deito*

Paulo Leminski

**RESUMO**

BENNATON ROLIM MARCON, Erik; GAUDEDA, Igor Ivan. **Dispositivo não invasivo para o monitoramento do estado de saúde de idosos**. 65 f. Trabalho de Conclusão de Curso (Graduação) – Engenharia Eletrônica – Departamento Acadêmico de Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2017.

Este trabalho tem por objetivo realizar uma prova de conceito de um relógio monitor do estado de saúde voltado para pessoas idosas. O projeto visa a prototipação de um sistema de monitoramento e assistência para idosos através de um dispositivo a ser usado no pulso e uma base transreceptora conectada à Internet. As principais funções do dispositivo são o monitoramento de batimentos cardíacos, a detecção de queda e um botão de emergência. Todas as informações coletadas em tempo real pelo dispositivo podem ser acessadas através de uma interface web, alertando familiares e/ou o sistema de saúde sobre possíveis riscos à vida do usuário.

**Palavras-chave:** Idoso; saúde; monitoramento; baixo consumo; internet das coisas;

**ABSTRACT**

BENNATON ROLIM MARCON, Erik; GAUDEDA, Igor Ivan. **Non-invasive life monitoring device for the elderly**. 65 p. Undergraduate Thesis – Electronic Engineering – Department of Electronic, Federal University of Technology - Paraná. Curitiba, 2017.

This thesis aims to develop, as a proof of concept, a health-monitoring watch directed to the elderly population. This project regards the prototyping of a health-monitoring and aiding bracelet and a transceiver base connected to the Internet. Its main functions are the heartbeat rate monitoring, fall detection and an emergency button. All information collected in real-time by the bracelet can be accessed via an web interface, alerting family members and/or the health care system about a possible threat to the user life.

**Keywords**: Elder; health; health monitoring; wearables; ultra-low power; Internet of things (IoT).

**LISTA DE ILUSTRAÇÕES**

[Figura 1 - Diagrama geral do projeto. 12](#_Toc484360888)

[Figura 2 - Diagrama geral do dispositivo de pulso. 13](#_Toc484360889)

[Figura 3 - Diagrama geral da base transceptora. 14](#_Toc484360890)

[Figura 4 - Diagrama geral da aplicação web. 14](#_Toc484360891)

[Figura 5 - Exemplo de WBS utilizado no projeto. 17](#_Toc484360892)

[Figura 6 - Ciclo PDCA 18](#_Toc484360893)

[Figura 7: Cronograma do projeto 20](#_Toc484360894)

[Figura 8 - Tela inicial do aplicativo web. 26](#_Toc484360895)

[Figura 9 - Tela de cadastro. 27](#_Toc484360896)

[Figura 10: Tela de Exibição do aplicativo web (Pessoa monitorada: Igor, sem alarmes ativos) 28](#_Toc484360897)

[Figura 11: Detalhe do histórico do nível de bateria no aplicativo web 29](#_Toc484360898)

[Figura 12 - Primeira parte da página da interface Web Embarcada. 31](#_Toc484360899)

[Figura 13 - Segunda parte da página da interface Web Embarcada. 32](#_Toc484360900)

[Figura 14 - Página de sucesso de conexão da interface Web embarcada. 32](#_Toc484360901)

[Figura 15 - Principais layers utilizadas na comunicação com a internet 34](#_Toc484360902)

[Figura 16 - Módulo Ethernet baseado no controlador W5500 36](#_Toc484360903)

[Figura 17 - Interface Wireless baseada no CC3300. 37](#_Toc484360904)

[Figura 18 - O módulo ESP8266 38](#_Toc484360905)

[Figura 19: Kit de desenvolvimento WEMOS D1 mini (ESP8266) 40](#_Toc484360906)

[Figura 20: Foto da placa access point do Kit de desenvolvimento ez430-Chronos 43](#_Toc484360907)

[Figura 21: Fluxograma geral do firmware do ESP8266 53](#_Toc484360908)

[Figura 22: Fluxograma do estado SCAN 54](#_Toc484360909)

[Figura 23: Fluxograma do estado SETUP\_SERVER 55](#_Toc484360910)

[Figura 24: Fluxograma do estado WAIT\_USER 56](#_Toc484360911)

[Figura 25: Fluxograma do estado CONNECT 57](#_Toc484360912)

[Figura 26: Fluxograma do estado CONFIGURED 58](#_Toc484360913)

[Figura 27: Fluxograma do firmware do Access Point 59](#_Toc484360914)

[Figura 28: Esquemático do circuito da Base 61](#_Toc484360915)

**LISTA DE TABELAS**

[Tabela 1: Comparativo entre as opções de interfaces com a Internet 38](#_Toc484360916)

[Tabela 2: Especificações Técnicas ESP8266 39](#_Toc484360917)

[Tabela 3: Protocolo de Comunicação entre Access Point e ESP8266 45](#_Toc484360918)

[Tabela 4: Protocolo de comunicação entre Access Point e Relógio 46](#_Toc484360919)

[Tabela 5: Campos de dados utilizados no canal do Thingspeak 48](#_Toc484360920)

**LISTA DE ABREVIATURAS, SIGLAS E ACRÔNIMOS**

**SUMÁRIO**

[1. INTRODUÇÃO 9](#_Toc484360921)

[1.1. DELIMITAÇÃO DO TEMA 9](#_Toc484360922)

[1.2. MOTIVAÇÃO 10](#_Toc484360923)

[1.3. PROBLEMAS E PREMISSAS 10](#_Toc484360924)

[1.4. OBJETIVOS 11](#_Toc484360925)

[1.4.1. Objetivo geral 11](#_Toc484360926)

[1.4.2. Objetivos específicos 11](#_Toc484360927)

[1.1. Dispositivo de pulso 11](#_Toc484360928)

[1.4.3. Base transceptora 12](#_Toc484360929)

[1.4.4. Aplicação Web 12](#_Toc484360930)

[1.5. DIAGRAMAS 12](#_Toc484360931)

[1.5.1. Diagrama geral 12](#_Toc484360932)

[1.5.2. Diagrama do dispositivo de pulso 13](#_Toc484360933)

[1.5.3. Diagrama da base transceptora 13](#_Toc484360934)

[1.5.4. Diagrama da aplicação web 14](#_Toc484360935)

[1.6. PROCEDIMENTOS METODOLÓGICOS 15](#_Toc484360936)

[1.6.1 Trabalho de Conclusão de Curso 1 15](#_Toc484360937)

[1.6.2 Trabalho de Conclusão de Curso 2 15](#_Toc484360938)

[1.7. ESTRUTURA DO TRABALHO 15](#_Toc484360939)

[2. PLANEJAMENTO E FERRAMENTAS DE GERENCIAMENTO DE PROJETO 17](#_Toc484360940)

[2.1. Estrutura Analítica de Processos 17](#_Toc484360941)

[2.2. PDCA 17](#_Toc484360942)

[2.3. Gerenciamento de cronograma 19](#_Toc484360943)

[2.4. ANÁLISE DE RISCOS 20](#_Toc484360944)

[2.4.1. Aquisição de batimentos cardíacos errática 20](#_Toc484360945)

[2.4.2. Não detecção de quedas 21](#_Toc484360946)

[2.4.3. Duração da bateria 21](#_Toc484360947)

[2.4.4. Alcance reduzido 21](#_Toc484360948)

[2.4.5. Comunicação com a internet 22](#_Toc484360949)

[2.4.6. Aceitação pelo mercado e usuários 22](#_Toc484360950)

[3. DESENVOLVIMENTO 23](#_Toc484360951)

[3.1. PESQUISAS 23](#_Toc484360952)

[3.2. DESENVOLVIMENTO WEB 24](#_Toc484360953)

[3.2.1. Desenvolvimento do aplicativo web 24](#_Toc484360954)

[3.2.1.1. Tela Inicial 25](#_Toc484360955)

[3.2.1.2. Tela de Cadastro 26](#_Toc484360956)

[3.2.1.3. Tela de Exibição 27](#_Toc484360957)

[3.3. Desenvolvimento da interface gráfica web 29](#_Toc484360958)

[3.4. DESENVOLVIMENTO DA BASE 32](#_Toc484360959)

[3.4.1. Circuito para interface com a internet 33](#_Toc484360960)

[3.4.1.1. A comunicação com a Internet e o Protocolo TCP/IP 33](#_Toc484360961)

[3.4.1.2. Comparativo entre opções de hardware disponíveis 35](#_Toc484360962)

[3.4.1.3. O SoC ESP8266 39](#_Toc484360963)

[3.4.2. Interface com Usuário 40](#_Toc484360964)

[3.4.2.1. Web server 41](#_Toc484360965)

[3.4.2.2. Indicadores Luminosos 42](#_Toc484360966)

[3.4.3. Comunicação com o relógio 43](#_Toc484360967)

[3.4.3.1. Circuito 43](#_Toc484360968)

[3.4.3.2. Protocolo 44](#_Toc484360969)

[3.4.4. Comunicação com a Internet 46](#_Toc484360970)

[3.4.4.1. Princípio de Funcionamento 46](#_Toc484360971)

[3.4.4.2. Thingspeak 46](#_Toc484360972)

[3.4.4.2.1. Utilização dos campos do canal 47](#_Toc484360973)

[3.4.4.2.2. Descrição das requisições utilizadas 49](#_Toc484360974)

[3.4.4.3. IFTTT 50](#_Toc484360975)

[3.4.4.4. NTP server 51](#_Toc484360976)

[3.4.5. Firmware 52](#_Toc484360977)

[3.4.5.1. Firmware do ESP8266 52](#_Toc484360978)

[3.4.5.2. Firmware do Access Point MSP430 58](#_Toc484360979)

[3.4.6. Hardware 60](#_Toc484360980)

[3.5. DESENVOLVIMENTO DO RELÓGIO 61](#_Toc484360981)

[3.5.1. Definição do kit de desenvolvimento 61](#_Toc484360982)

[3.5.2. Implementação do sensor de batimento cardíacos 61](#_Toc484360983)

[3.5.3. Implementação do sensor de queda 62](#_Toc484360984)

[3.5.4. Implementação do alarme para remédios 62](#_Toc484360985)

[3.5.5. Implementação do botão de pânico 62](#_Toc484360986)

[4. RESULTADOS 63](#_Toc484360987)

[5. CONSIDERAÇOES FINAIS 64](#_Toc484360988)

## INTRODUÇÃO

Durante as últimas décadas, devido ao aumento da qualidade e da expectativa de vida, a população idosa vem crescendo de forma constante [6]. Se em 1960 a expectativa de vida do brasileiro era de 54.2 anos, no último levantamento realizado pelo IBGE em 2015, foi constatado que a expectativa de vida média do brasileiro é de 75.5 anos [7].

Durante a senioridade o ser humano requer uma série de atenções e cuidados especiais, devido aos problemas de saúde e complicações típicos da idade avançada [8]. Com os avanços tecnológicos, em especial na área da internet das coisas (*Internet of Things - IOT*), temos a possibilidade de diminuir e até mesmo evitar alguns destes problemas típicos da idade avançada.

Devido às inúmeras mudanças socioeconômicas ocorridas nos últimos 30 anos, tivemos um aumento de 235% no número de idosos que vivem sozinhos entre 1992 e 2012 [2] e é em casa onde 75% dos acidentes com idosos acontecem [3].

Um terço dos atendimentos a lesões traumáticas são para pessoas com mais de 60 anos [4], justamente a faixa de idade mais susceptível a complicações médicas e que necessita de atendimento mais célere.

O rápido atendimento, principalmente para os idosos, é essencial para a diminuição de riscos à vida do paciente e para o aumento de suas chances de recuperação, em caso de infarto - por exemplo - o atendimento deve ocorrer dentro de no máximo 90 minutos [4].

Neste contexto, o aparecimento de uma solução tecnológica para o acompanhamento do estado de saúde geral dos idosos à distância e não invasivo é mera questão de tempo.

### DELIMITAÇÃO DO TEMA

Este projeto tem como objetivo o desenvolvimento e a implementação de um protótipo de solução completa de monitoramento das condições gerais de saúde de uma pessoa de mais idade, oferecendo uma maior segurança ao idoso e, consequentemente, mais tranquilidade para seus familiares e amigos.

O dispositivo contará com detecção de quedas, medidor de batimentos cardíacos, botão de pânico e alarme de horário para tomar os remédios. Caso algum imprevisto ou emergência aconteça, os familiares e o serviço médico urgente será alertado.

O sistema completo consiste num dispositivo móvel, não invasivo, para ser utilizado no pulso e uma base transceptora conectada à Internet. O dispositivo móvel se conectará com a base de forma sem fio e transmitirá todas as informações obtidas em tempo real.

A base, por sua vez, retransmitirá todas estas informações à um serviço *web*, que, de posso de uma chave de acesso, qualquer pessoa autorizada, seja ela um familiar ou o serviço médico urgente, poderá acessar.

### MOTIVAÇÃO

A tecnologia atual permite que seja feito um monitoramento remoto e não invasivo das condições gerais de saúde de uma pessoa, porém o mercado carece de um produto voltado para o público com mais idade e que tenha como principal objetivo promover a tranquilidade para os familiares e para o usuário, mais especificamente soluções de monitoramento das condições gerais de saúde voltado para o salvamento de vidas.

Existem diversos monitores de saúde voltados para a prática de atividades físicas e análise de desempenho esportivo, contudo, não há sequer um produto conhecido visando os cuidados da população com mais idade.

### PROBLEMAS E PREMISSAS

As dificuldades do projeto concentram-se na elaboração de um protótipo miniaturizado que possa ser usado no pulso de forma confortável, na obtenção de medidas confiáveis e, não obstante, no alto custo do projeto.

A detecção de queda e o monitoramento dos batimentos cardíacos de forma satisfatória e confiável serão os grandes desafios do projeto. As situações de uso no dia-a-dia estão longe das ideias, muito ruído, muitos movimentos bruscos e diversos fatores que podem atrapalhar na leitura dos sinais.

Diferenciar uma queda de um movimento brusco de braço e um ruído de um batimento cardíaco são apenas alguns dos desafios que serão enfrentados durante o desenvolvimento do projeto.

### OBJETIVOS

### Objetivo geral

Este projeto tem como objetivo desenvolver um protótipo não invasivo para o acompanhamento da saúde de idosos, através do monitoramento de batimentos cardíacos, da detecção de possíveis quedas e da presença de um botão de emergência.

### Objetivos específicos

O projeto consiste em três subsistemas específicos: o dispositivo, a base transreceptora e a aplicação web, cada qual com seus objetivos específicos.

### Dispositivo de pulso

1. Desenvolver a funcionalidade de monitoramento de batimentos cardíacos e detecção de possíveis quedas.
2. Desenvolver a funcionalidade de um botão de socorro, para ser usado em situações de emergência.
3. Desenvolver a funcionalidade de monitoramento de bateria, para indicar o nível atual e quando uma troca deverá ser feita.
4. Desenvolver a funcionalidade de lembrar o usuário a tomar um remédio em horários especificados.
5. Desenvolver o sistema de forma a maximizar a duração da bateria, para reduzir o tempo entre cargas.
6. Desenvolver a comunicação sem fio com a base transceptora, para receber e enviar dados.

### Base transceptora

1. Desenvolver um circuito para a interface com a internet.
2. Desenvolver uma interface com o usuário para indicar o funcionamento do sistema.
3. Desenvolver a comunicação com o dispositivo de pulso.
4. Desenvolver a comunicação com a internet.

### Aplicação Web

1. Desenvolver uma aplicação Web para enviar e receber dados da base transceptora.
2. Desenvolver a funcionalidade de alertar pessoas cadastradas a aplicação web sobre quedas, alterações nos batimentos cardíacos, acionamento do botão de emergência e bateria fraca.
3. Desenvolver a funcionalidade de exibição dos dados provenientes da base.

### DIAGRAMAS

### Diagrama geral

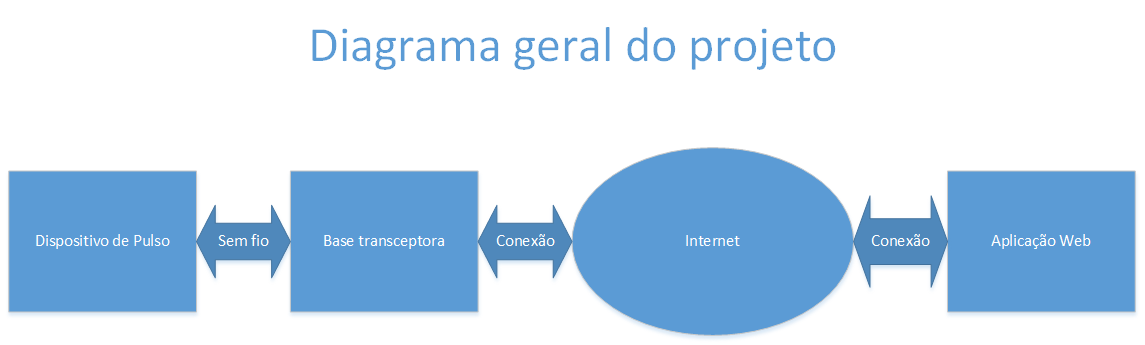


Figura - Diagrama geral do projeto.

Fonte: Autoria Própria

### Diagrama do dispositivo de pulso

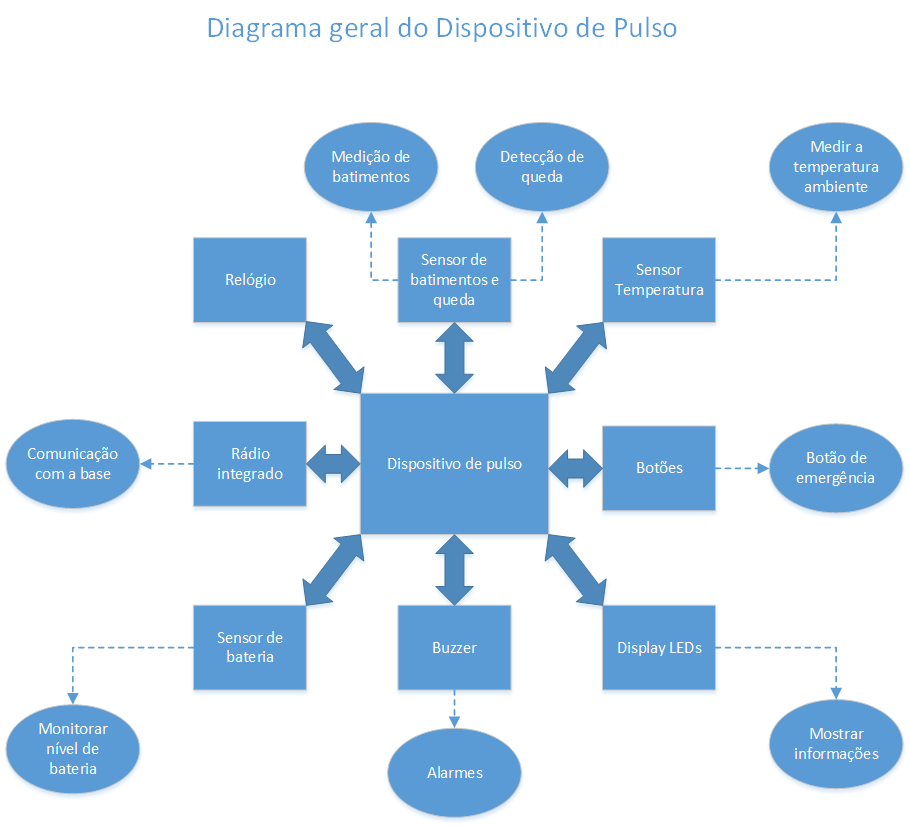


Figura - Diagrama geral do dispositivo de pulso.

Fonte: Autoria Própria

### Diagrama da base transceptora

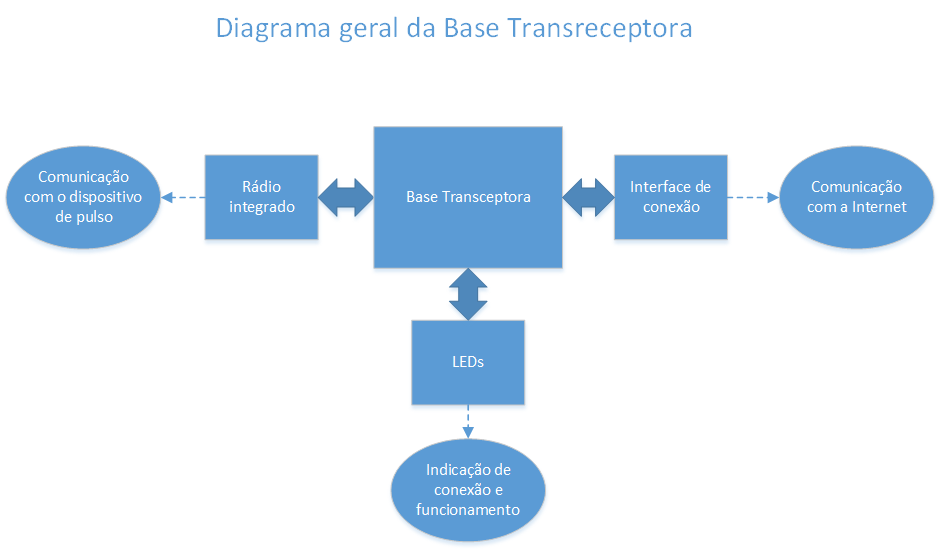


Figura - Diagrama geral da base transceptora.

Fonte: Autoria Própria

### Diagrama da aplicação web

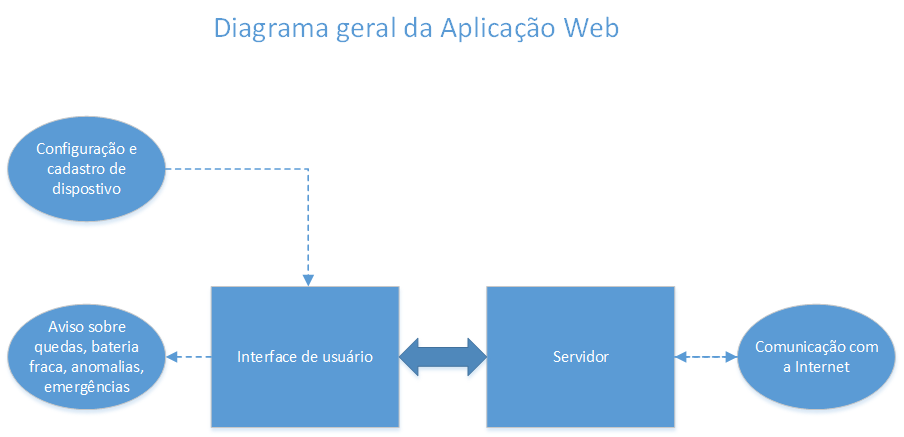


Figura - Diagrama geral da aplicação web.

Fonte: Autoria Própria

### PROCEDIMENTOS METODOLÓGICOS

### Trabalho de Conclusão de Curso 1

1. Delimitação do tema do projeto;
2. Pesquisa por produtos semelhantes existentes no mercado;
3. Planejamento e definição do escopo do projeto;
4. Esboço inicial das tarefas a serem realizadas e do caminho a ser perseguido para a elaboração do projeto;
5. Planejamento e gerenciamento do tempo – desenvolvimento de um cronograma de trabalho;

### Trabalho de Conclusão de Curso 2

1. Prosseguimento do gerenciamento de escopo;
2. Prosseguimento do gerenciamento de tempo;
3. Desenvolvimento da aplicação *web;*
4. Desenvolvimento do protocolo de comunicação entre a base e o relógio;
5. Implementação dos recursos de monitoramento de vida no relógio;
6. Realização de testes;

### ESTRUTURA DO TRABALHO

O presente trabalho está organizado em 5 capítulos, nos quais são apresentados:

**Capítulo 1:** Introdução e delimitação do tema; motivações para a escolha do projeto e objetivos a serem alcançados

**Capítulo 2:** Definição do escopo do projeto, utilização de ferramentas de gestão de projetos.

**Capítulo 3:** Desenvolvimento do projeto, definição do kit de desenvolvimento a ser utilizado e dos sensores.

**Capítulo 4:** Discussão sobre o protótipo obtido, diferenças entre a proposta original e resultado final do trabalho, vantagens e desvantagens.

**Capítulo 5:** Considerações finais acerca do projeto, descrição dos conhecimentos utilizados, identificação de possibilidade de real implementação do projeto no mercado atual. Neste capítulo, não obstante, são apresentadas as sugestões para a melhoria e continuidade do projeto.

## PLANEJAMENTO E FERRAMENTAS DE GERENCIAMENTO DE PROJETO

Nesta etapa foram utilizadas ferramentas de gerenciamento de projetos, focando no gerenciamento de escopo e de tempo. Desta forma, foi possível definir – de forma satisfatória – um cronograma a ser seguido e a divisão de tarefas entre os membros da equipe. Visando desenvolver o melhor protótipo possível, aplicou-se uma técnica de melhoria contínua, utilizada comumente em grandes empresas, visando otimizar as decisões tomadas em cada etapa do projeto.

### Estrutura Analítica de Processos

Analiticamente seguimos a técnica conhecida como WBS (*Work Breakdown Structure*), resumidamente, WBS consiste em uma ferramenta visual de definição do escopo do projeto. Deve-se decompor uma tarefa grande e complexa (de maior hierarquia) em diversas tarefas menores e mais simples de serem entendidas e realizadas.

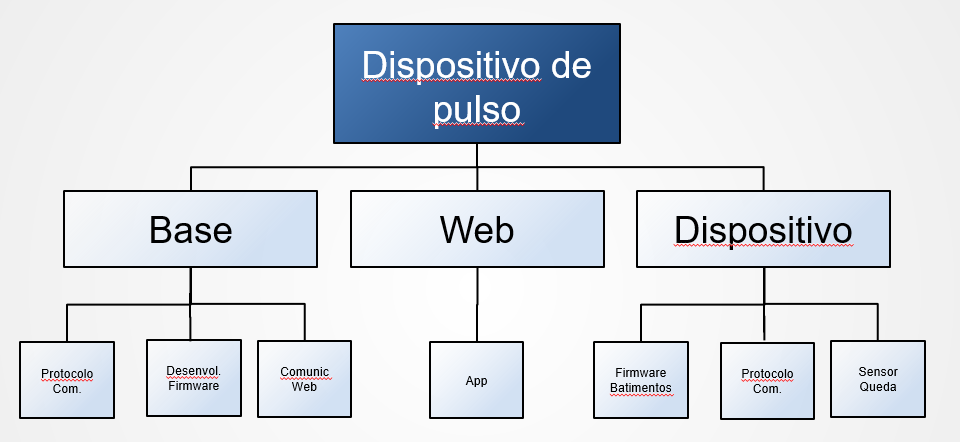


Figura - Exemplo de WBS utilizado no projeto.

Fonte: Autoria Própria

### PDCA

PDCA, do inglês *plan-do-check-act*, é uma técnica de melhoria continua inventada por Walter Stewhart e, posteriormente, modificada por W. Edwards Deming.

As quatro etapas do PDCA são:

* A etapa do planejamento (Plan) corresponde à identificação e análise do problema;
* A etapa da execução (Do) corresponde ao desenvolvimento e à implementação de soluções;
* A etapa de verificação ou controle (Check) corresponde à avaliação dos resultados. Se o objetivo foi atingido, continua-se para a etapa de ação. Caso contrário, retorna-se à etapa de planejamento;
* A etapa da ação (Act) corresponde à adoção da solução propriamente;

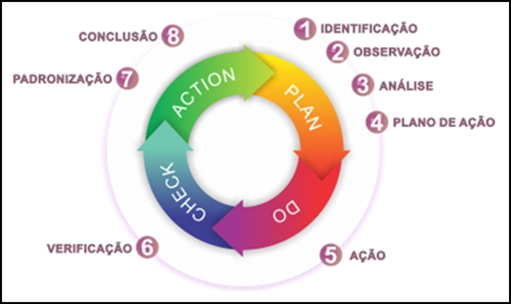
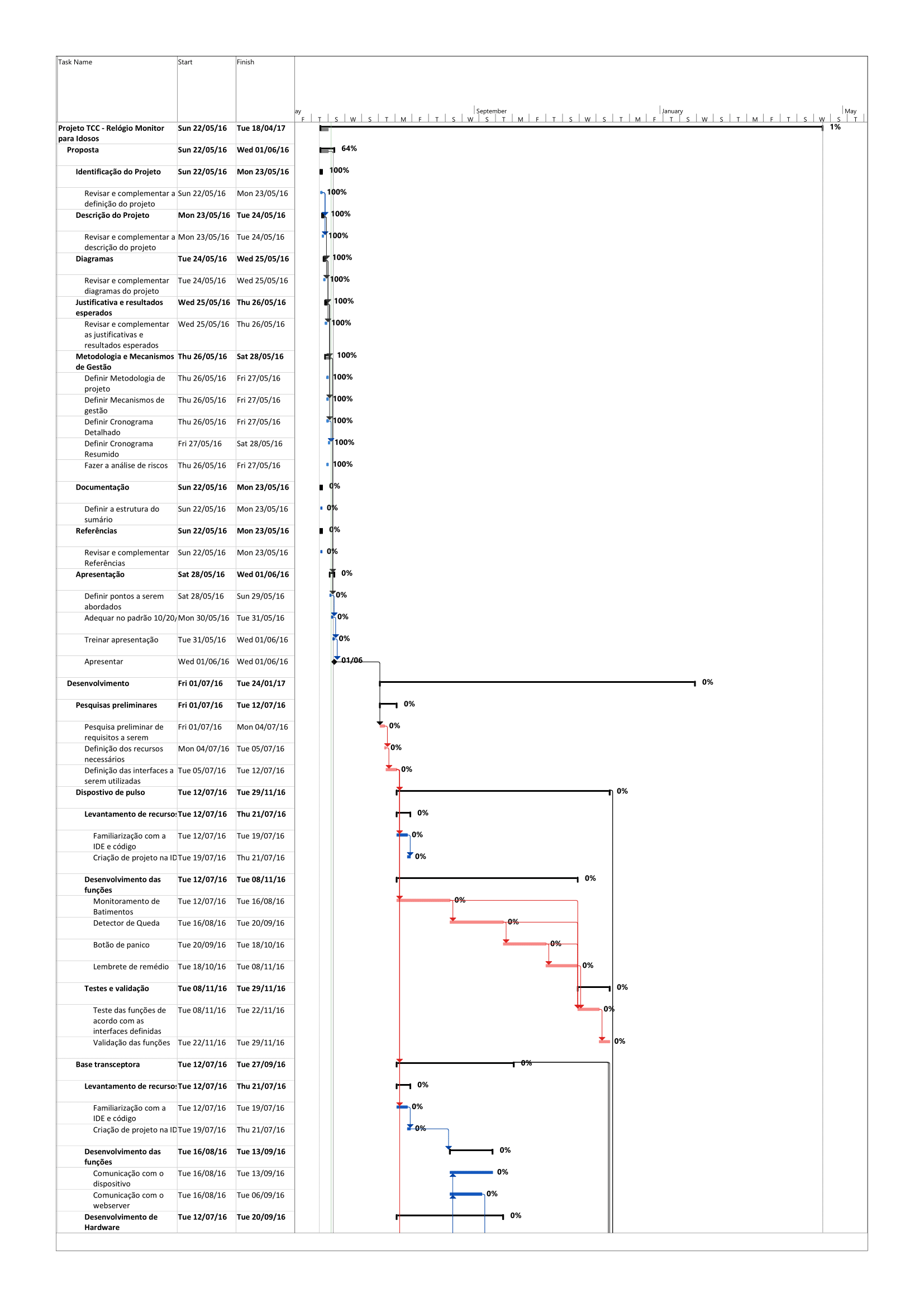


Figura - Ciclo PDCA

Fonte: ?????

Neste projeto, utilizamos as técnicas do PDCA ao longo de todo o seu desenvolvimento, buscando tomar sempre as melhores decisões, otimizar o tempo aplicado ao seu desenvolvimento e garantir que a solução seria a melhor possível.

### Gerenciamento de cronograma



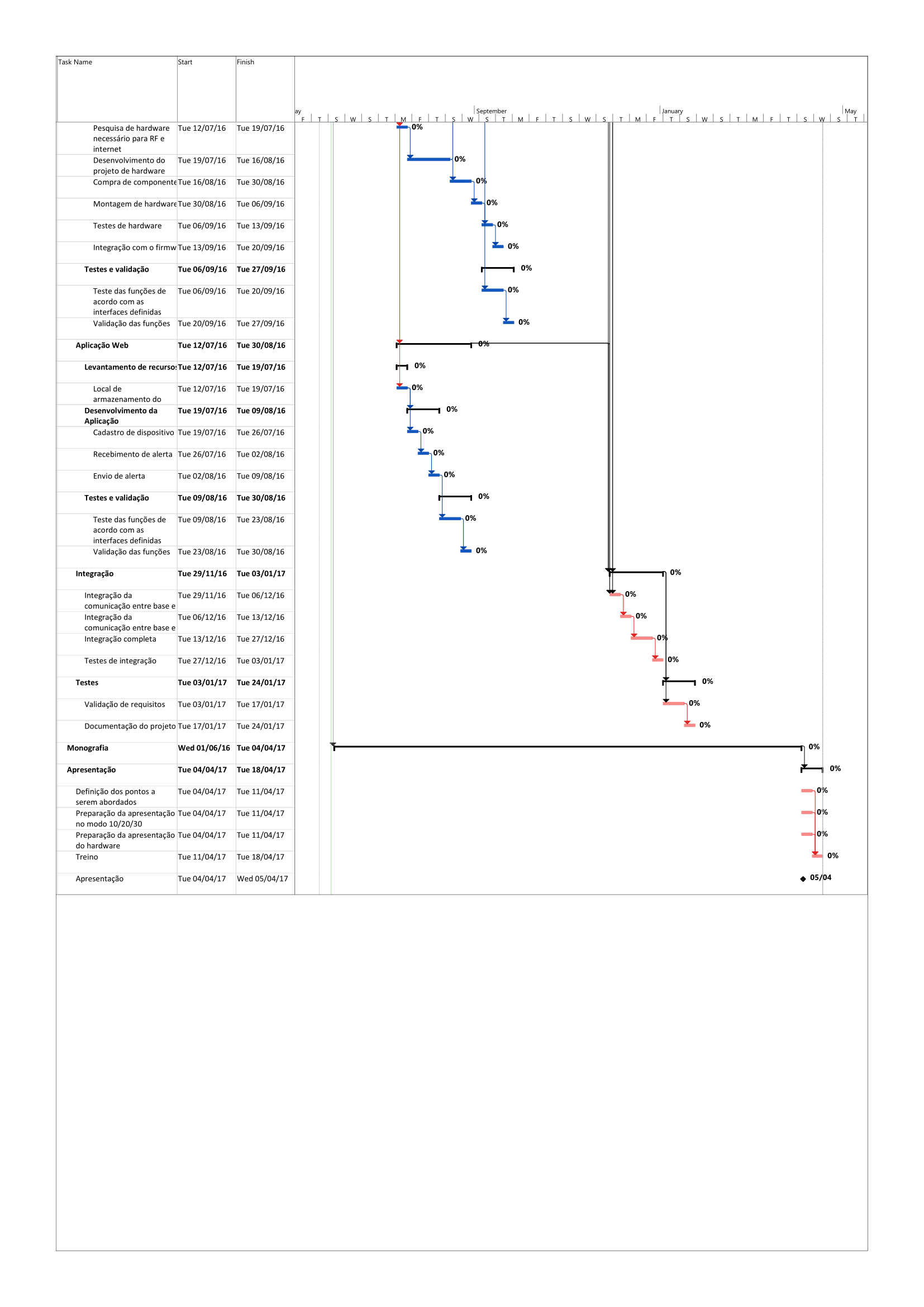


Figura : Cronograma do projeto

Fonte: Autoria própria

### ANÁLISE DE RISCOS

Como todo projeto de engenharia, existem riscos que devem ser considerados no desenvolvimento, sendo os principais mostrados a seguir.

### Aquisição de batimentos cardíacos errática

Devido à exposição do dispositivo a situações cotidianas impossíveis de serem previstas, é possível que existam situações em que o dispositivo venha a adquirir os batimentos cardíacos de maneira errada, causado por vibrações, movimentos, etc. Senso assim, é necessário que haja um algoritmo para detectar se a medição dos batimentos é confiável ou não. Como este fator possui BAIXO RISCO de acontecer, mas como causa um impacto direto na confiabilidade do sistema, é classificado como ALTO IMPACTO.

### Não detecção de quedas

Pelo mesmo motivo de não ser possível prever todas as situações pelas quais o dispositivo irá passar, podem ocorrer situações em que quedas são indevidamente identificadas ou não identificadas. Assim, vários testes devem serem feitos para que a detecção das quedas seja ampla. Este fator atua diretamente sobre a confiabilidade do sistema, tendo BAIXO RISCO e ALTO IMPACTO.

### Duração da bateria

Por ser um dispositivo portátil, é importante que o mesmo tenha uma bateria de longa duração. O projeto visa atingir um nível de consumo de forma a bateria ser trocada apenas em intervalos de pelo menos 3 meses. Se por algum motivo o consumo seja maior que o esperado, o produto terá um atrativo menor, porém sem perder a sua função principal. Por isso, classifica-se como BAIXO RISCO e possui BAIXO IMPACTO.

### Alcance reduzido

Por utilizar de radiofrequência para a comunicação com a base, existe a possibilidade que o alcance do dispositivo não seja suficiente para cobrir uma casa por completo, por exemplo. Isso causaria um impacto na utilização do sistema, pois limitaria o espaço em que o sistema funcionaria de forma completa. Apesar de não impedir o funcionamento, irá o limitar e pode causar consequências mais sérias. Portanto classifica-se como MÉDIO RISCO e MÉDIO IMPACTO.

### Comunicação com a internet

A única forma do dispositivo se comunicar com o mundo externo é através da internet, sendo assim é de extrema importância que a comunicação tenha altos índices de funcionamento. Por ser baixa a possibilidade de não existir comunicação com a internet, classifica-se como BAIXO RISCO, porém como é uma função vital para o projeto, classifica-se como ALTO IMPACTO.

### Aceitação pelo mercado e usuários

Apesar de ser ainda um protótipo, é interessante que o mercado venha a aceitar a ideia, e, portanto, possibilitando o desenvolvimento de um produto final. Por não ser o objetivo do projeto, especificamente, um produto final, classifica-se como BAIXO RISCO e BAIXO IMPACTO.

## DESENVOLVIMENTO

O projeto foi desenvolvido em 4 etapas. Inicialmente, abordou-se o escopo do projeto, os métodos de desenvolvimento, o cronograma a ser seguido e a distribuição das atividades. Em seguida, iniciou-se o desenvolvimento do aplicativo web, depois o desenvolvimento da base de comunicação e, por último, o desenvolvimento do dispositivo móvel não invasivo.

### PESQUISAS

Durante a fase de concepção do projeto foi definido que seria necessária uma interface, acessível através de qualquer dispositivo capaz de acessar páginas da internet, por onde a interação com o usuário do sistema seria feita. A importância de se ter uma interface Web comparado a um programa de computador convencional, é a grande acessibilidade e flexibilidade de dispositivos, sistemas operacionais e custos relacionados ao acesso. Além disso, o uso de dispositivos móveis para acesos a informações tem crescido fortemente, cerca de 81% em 2013, chegando a 1.5 *exabytes* de dados por mês deste mesmo ano (L. YANG et al, 2016). Desta forma, justifica-se a escolha deste tipo de interface com o usuário.

Anterior ao desenvolvimento, foram definidos alguns requisitos indispensáveis para esta aplicação. Estes requisitos envolvem principalmente serviços externos que serão usados pelo aplicativo, sendo definidos a seguir.

Ferramenta de desenvolvimento de aplicativo móvel: desenvolver um aplicativo móvel pode ser facilitado através de uma boa escolha de ferramentas e bibliotecas de programação que simplifiquem o desenvolvimento. Neste caso, foi escolhido a ferramenta “*Bubble Apps*”, sendo sua maior vantagem a facilidade em construir aplicativos simples. Através desta ferramenta, pode ser facilmente integrado um sistema de login, um banco de dados simples, layout e conexão com serviços externos.

Serviço de envio de mensagens de texto: notificar usuários do aplicativo em caso de ocorrência. Foi escolhida a empresa “*Twillio*”, que fornece uma API baseada em requisições onde podem ser enviados SMS gratuitamente em sua versão de testes.

Serviço de envio de mensagens pelo Twitter: notificar usuários do aplicativo em caso de ocorrência. Foi escolhido o serviço “IFTTT”, que também fornece uma API baseada em requisições, de forma gratuita, onde é possível utilizar uma conta existente para enviar mensagens para outros usuários da rede social. No caso deste sistema, as pessoas que são alertadas caso alguma haja alguma ocorrência.

Serviço de envio de e-mail: notificar usuários do aplicativo em caso de ocorrência. Neste caso também foi escolhido o serviço “IFTTT”, que usa uma conta de e-mail existente para enviar mensagens aos usuários do aplicativo.

Serviço de armazenamento: armazenar as informações coletadas pelo dispositivo de pulso. A ferramenta “*ThingSpeak*”, desenvolvida especialmente para a Internet das Coisas, escolhida neste trabalho, é capaz de armazenar dados de sensores e outras informações e tornar esses dados acessíveis facilmente, através de uma API baseada em requisições.

### DESENVOLVIMENTO WEB

O desenvolvimento web presente no projeto consiste nas telas do aplicativo web e em uma interface de configuração embarcada na base. O aplicativo é usado pelos usuários de forma continua, ou seja, é através deste canal que estes acessarão as informações da pessoa monitorada. A interface embarcada é utilizada apenas uma vez, para realizar a configuração inicial do sistema em uma nova pessoa que será monitorada.

### Desenvolvimento do aplicativo *web*

Para o desenvolvimento, foi utilizada a ferramenta “*Bubble Apps”*, que oferece uma interface amigável, com diversas funções já implementadas, além de ser gratuita, observando algumas limitações. Visando a simplicidade no uso do aplicativo, foram desenvolvidas três telas que são suficientes para o funcionamento do sistema.

O uso da ferramenta consiste em criar as páginas do aplicativo, utilizar elementos comuns, como caixas de texto, botões, *links* etc. e criar ações baseadas em eventos. Estas ações envolvem redirecionar para outra página, fazer alterações no banco de dados, modificar elementos das telas, fazer requisições para outros serviços, dentre outras. Os eventos consistem em pressionamento de botões, cliques em *links*, ações periódicas no tempo etc.

### Tela Inicial

Nesta tela, é possível realizar um novo cadastro ou entrar com um usuário já existente. Para realizar o cadastro, basta inserir um endereço de e-mail e a senha a ser utilizada, apertando no botão Cadastrar para concluir. O usuário então será direcionado para a tela de cadastro. Caso este usuário já esteja cadastrado, este será direcionado para tela de exibição, onde é possível ver os dados em tempo real, alertas e as informações de contato.



Figura - Tela inicial do aplicativo web.

Fonte: Autoria Própria

### Tela de Cadastro

A tela de cadastro consiste apenas na entrada do código da pessoa a ser monitorada. Esse código identifica de forma única a pessoa a ser monitorada. É também verificado que o e-mail cadastrado no aplicativo é o mesmo do cadastrado durante a configuração do dispositivo. Dessa forma, é vinculado o usuário atual com a pessoa a ser monitorada. Após a finalização, é carrega a tela de exibição dos dados, que se torna a tela padrão quando se abre o aplicativo.

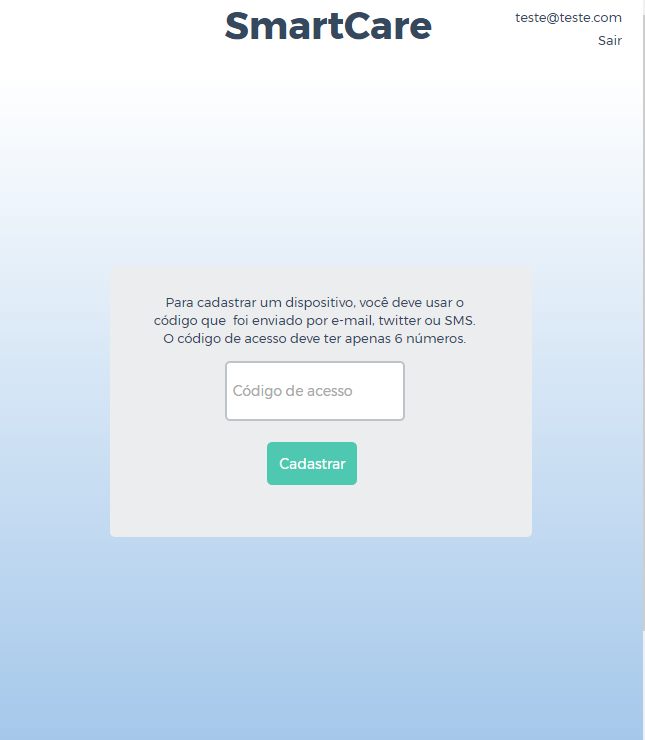


Figura - Tela de cadastro.

Fonte: Autoria Própria

### Tela de Exibição

A tela de exibição é a tela principal de operação do aplicativo. Nesta tela, todas as informações pertinentes à operação do sistema estão presentes. É possível visualizar alertas, as últimas leituras de sensores e os dados atuais de contato, sendo permitido alterá-los caso seja necessário.

A página faz uma atualização dos dados através dos dados armazenados no servidor do ThingSpeak, que foram enviados pela base de comunicação. A forma de obtenção dos dados é através de requisições web executadas a cada 30 segundos, descritas em maior detalhe na seção 3.4.4.2.2.

O usuário pode também alterar as informações de contato, atualizando os campos da página e clicando no botão “Atualizar!”. Quando o botão é pressionado, a página envia uma requisição ao servidor ThingSpeak para atualizar os dados. Na próxima leitura executada pela base, ela verificará a mudança dos dados e atualizará as informações internamente, para que no próximo alerta, por exemplo, as notificações sejam enviadas para os destinatários atualizados.

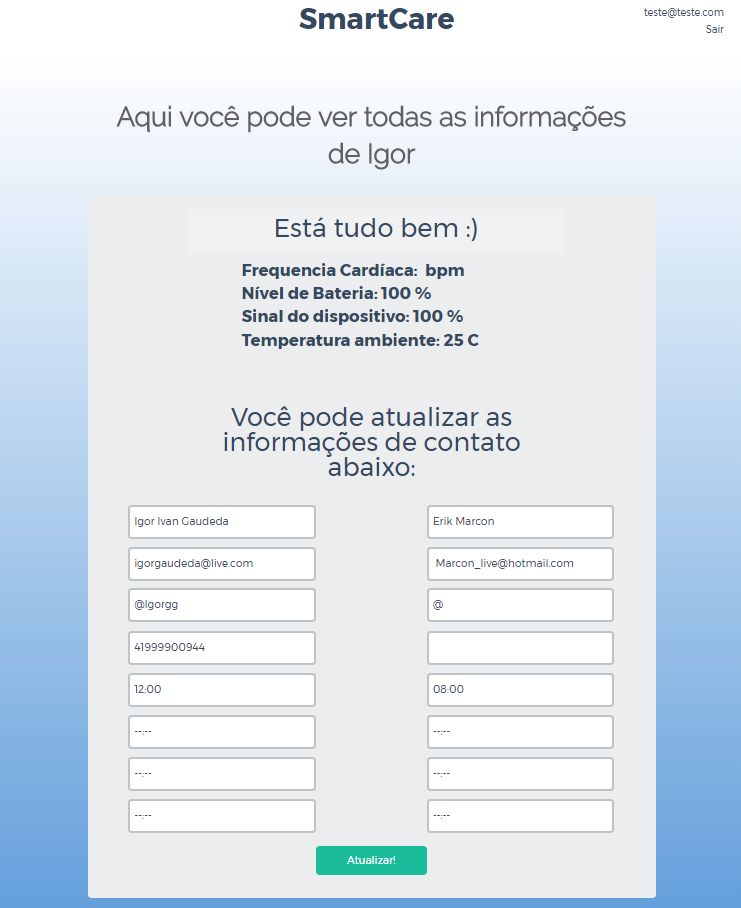


Figura : Tela de Exibição do aplicativo web (Pessoa monitorada: Igor, sem alarmes ativos)

Fonte: Autoria Própria

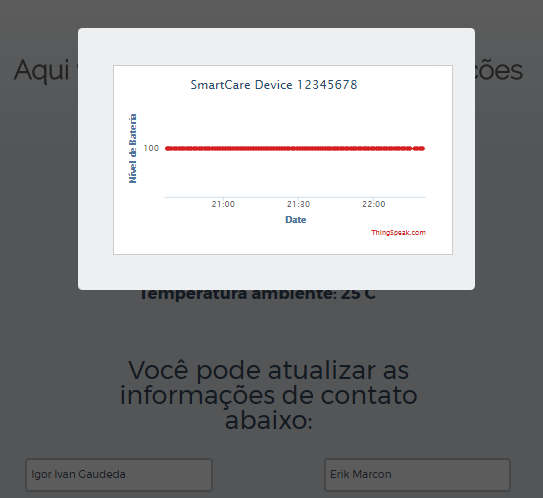


Figura : Detalhe do histórico do nível de bateria no aplicativo web

Fonte: Autoria Própria

### Desenvolvimento da interface gráfica *web*

A tela de cadastro consiste apenas na entrada do código da pessoa a ser monitorada. Esse código identifica de forma única a pessoa a ser monitorada. É também verificado que o e-mail cadastrado no aplicativo é o mesmo do cadastrado durante a configuração do dispositivo. Dessa forma, é vinculado o usuário atual com a pessoa a ser monitorada. Após a finalização, é carrega a tela de exibição dos dados, que se torna a tela padrão quando se abre o aplicativo.

De forma a reduzir a complexidade de operação e configuração do dispositivo, optou-se por fazer uma interface Web embarcada na base transceptora do sistema. Esta escolha pode também ser justificada pela redução de custos, visto que não é necessário um display ou mesmo botões para operar o sistema, além de ser uma forte tendência dentro da indústria da informação e de dispositivos conectados à internet (XIAO & ZENG, p479).

A interface web embarcada possui apenas a função de configuração do dispositivo, sendo executada apenas uma vez, na instalação do sistema na residência da pessoa a ser monitorada. Desta forma, a configuração se torna intuitiva até mesmo para pessoas leigas em informática e tecnologia.

Por se tratar de uma interface embarcada, e, portanto, com recursos limitados, as páginas que compõem esta interface possuem um código simples e enxutos, para economizar memória e processamento por parte do microcontrolador da base. Entretanto, vários recursos visuais foram usados, pois este são carregados e processados no navegador de internet, e não no microcontrolador.

A página principal possui campos a serem preenchidos pela pessoa que está realizando a instalação. Apenas alguns campos são mandatórios, como o nome da pessoa a ser monitorada, o nome do primeiro monitor e o seu e-mail. Os outros campos são opcionais. É possível cadastrar duas pessoas para receberem alertas, além de até oito horários de alarme para que a pessoa monitorada não se esqueça de tomar remédios.

No canto inferior da página, são mostradas as redes Wi-Fi detectadas na residência, sendo que uma delas deve ser selecionada e preenchida corretamente nos campos “Nome da Wi-Fi” e “Senha”.



Figura - Primeira parte da página da interface Web Embarcada.

Fonte: Autoria Própria

Ao clicar no botão “Conectar”, os dados escritos nos campos da página são passados para o web-server embarcado, que irá processar os dados e caso esteja tudo correto e a conexão com a internet for realizada, irá aparecer a tela de confirmação com o código único do dispositivo. Esse código é o mesmo usado para configurar o aplicativo. Caso a configuração não seja bem-sucedida, a página é recarregada.

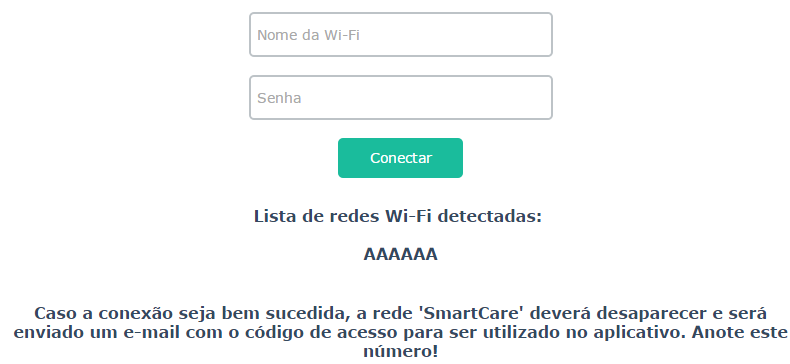


Figura - Segunda parte da página da interface Web Embarcada.

Fonte: Autoria Própria

Os detalhes da implementação do servidor e como as páginas são processadas pelo web-server são mostrados na seção **Erro! Fonte de referência não encontrada.**.



Figura - Página de sucesso de conexão da interface Web embarcada.

Fonte: Autoria Própria

### DESENVOLVIMENTO DA BASE

O desenvolvimento da base de comunicação do projeto se deu seguindo os objetivos definidos na proposta de trabalho. Dentro de cada objetivo, foi necessária a divisão em objetivos menores para facilitar o desenvolvimento. Nesta seção, todo o processo e o resultado final desta parte do projeto é mostrada em detalhes.

### Circuito para interface com a internet

Para cumprir o primeiro objetivo para esta parte do projeto, foi necessário o projeto de um circuito capaz de se conectar à internet e ao mesmo comunicar-se com o relógio. No entanto, como será mostrado, ficaria mais fácil ter um circuito especificamente dedicado apenas para se comunicar com a internet, comunicando-se também com outro circuito que faz a transmissão dos dados para o relógio através de radiofrequência. Primeiramente é necessário entender como a comunicação com a internet ocorre, para então definir um circuito adequado para as necessidades deste projeto.

### A comunicação com a Internet e o Protocolo TCP/IP

A internet é uma rede mundial de dispositivos, mais precisamente uma rede composta por várias outras redes. Não há um controle centralizado, sendo assim, é necessário seguir regras de comunicação para que seja possível a comunicação de uma forma ordenada. Foram definidas camadas, conhecidas como *layers*, que atuam em diferentes etapas do processo de enviar e receber uma informação. Essas camadas são conhecidas como *OSI* (*Open System Interconnected*) *Model*. Este modelo é importantíssimo para simplificar as aplicações do ponto de vista de engenharia, visto que, para enviar um e-mail, por exemplo, não seja necessário saber detalhes da conexão como se é cabeada ou *wireless*.

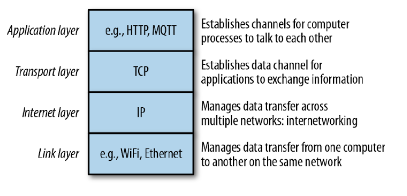


Figura - Principais *layers* utilizadas na comunicação com a internet

Fonte: Rowland et al. p.76

Na *Application layer*, são definidos canais para os programas possam se comunicar entre si. Neste caso vale citar como exemplo o HTTP (*HyperText Transport Protocol*) que é utilizado para enviar e receber páginas da internet, e o FTP (*File Transfer Protocol*), utilizado para a transferência de arquivos.

A *Transport layer* estabelece um canal para que o dados trafeguem de um ponto ao outro com integridade. Em geral, os dados são divididos em pacotes e enviados de forma não ordenada, e ao chegar no destino, são verificados, ordenados e reunidos. Os protocolos mais utilizados nesta camada são o TCP (*Transmission control protocol)* e o UDP (*User datagram protocol*).

Na *Internet Layer* é gerenciado a transferência de dados pelas diversas redes com compõem a internet. Para que seja possível identificar a origem e destino dos dados, é utilizado um número, chamado de endereço IP.

A *Link layer* está relacionada ao aspecto físico da transferência desses dados pela internet. Neste caso estão inclusas as conexões cabeadas, as conexões *wireless*, pela rede celular e rádios proprietários.

Para exemplificar a utilização destas camadas, pode se tomar como exemplo o acesso a uma página da internet por um computador. Neste caso, a *Application layer* do cliente está no navegador de internet, que irá enviar uma requisição por HTTP para um servidor na internet. Essa requisição é acomodada em pacotes na *Transport Layer*, que será então passada para a *Internet layer*, que processará se o pacote chegou no lugar correto ou se deve ser encaminhando a outro endereço. Na *Link layer*, é acessada a interface de comunicação ativa, como a *ethernet* e os dados são enviados. O servidor recebe os dados pela *Link layer*, verifica que é o destino na *Internet layer*, agrupa os pacotes na *tranport layer* e processa a requisição na *Application layer.* A página requisitada é então enviada para o cliente seguindo o mesmo caminho, e ao chegar é mostrada no navegador.

No caso de um dispositivo embarcado conectado a internet, o processamento envolvido nestas *layers* pode se tornar crítico para a aplicação principal. Sendo assim, existem diversas soluções no mercado voltadas a tirar o processamento das *layers* do processador principal. Algumas destas soluções foram consideradas para serem utilizadas neste projeto, visto que o processamento necessário para a aplicação principal seria baixo e não justificaria adquirir um microcontrolador mais poderoso.

### Comparativo entre opções de hardware disponíveis

Conforme a seção anterior mostra, ter um hardware dedicado a processar as informações enviadas e recebidas pela internet pode trazer grandes vantagens, como redução da complexidade do projeto de hardware e firmware, bem como a possível redução de custos e a utilização de tecnologia já estabelecidas no mercado.

Fazendo uma pesquisa por interfaces de conexão com a internet disponíveis, chegaram-se a três opções para o projeto: uma interface *ethernet* baseada no controlador W5500, uma interface *wireless* baseada no controlador CC3000 e uma interface *wireless* baseada no SoC ESP8266.

A interface *ethernet* baseada no controlador W5500 foi é composta por uma placa contendo o controlador, um conector RJ-45 e pinos de IO compatíveis com a plataforma Arduino. O funcionamento da mesma se dá através da interface SPI utilizando um microcontrolador adicional para processar os dados. Dentre seus pontos fortes estão a rápida integração com a plataforma e as suas bibliotecas de utilização de fácil acesso e uso. Seus pontos fracos incluem ser uma solução que necessita de cabeamento, o custo de cerca de U$ 44,95[[1]](#footnote-1), o volume da placa e a necessidade de uma placa Arduino ou outra compatível, visto que esta é apenas uma interface adicional.

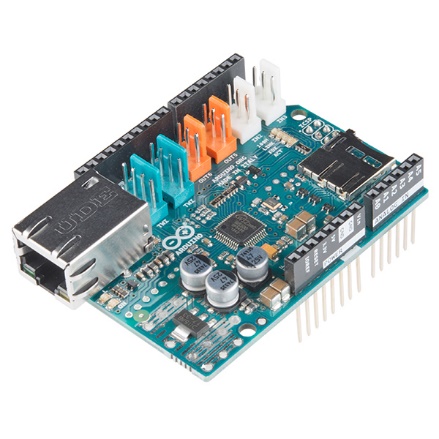


Figura - Módulo *Ethernet* baseado no controlador W5500

Fonte: https://new.sparkfun.com/products/11166

Outra opção considerada, a interface *wireless* baseada no controlador CC3300, também possui uma placa com este controlador, uma antena integrada e pinos de IO compatíveis com a plataforma Arduino. O modo de funcionamento é semelhante a interface *ethernet*, com a principal diferença sendo as funções específicas de conexões sem fio, como procura de redes, autenticação etc. Seus pontos fortes incluem a fácil integração com a plataforma, suas bibliotecas de fácil acesso e uso, além de não necessitar de fios e possuir tamanho reduzido. Seus pontos fracos são o custo relativamente alto, cerca de U$34,95[[2]](#footnote-2), e a necessidade de um microcontrolador externo.

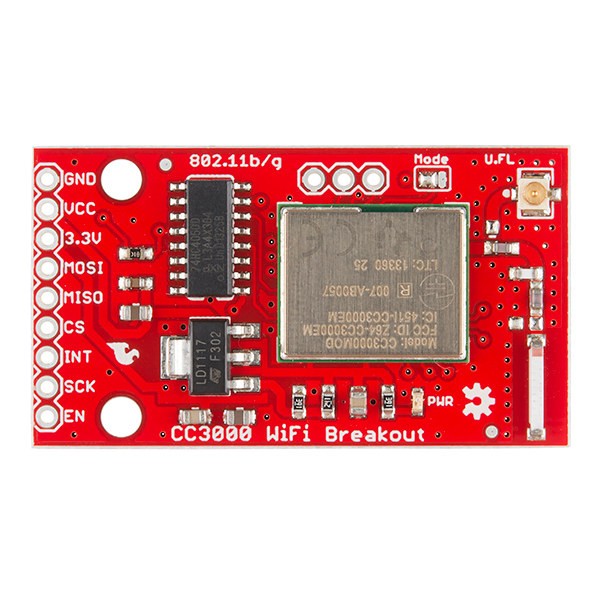


Figura - Interface *Wireless* baseada no CC3300.

Fonte: https://botland.com.pl/moduly-wifi/2276-cc3000-modul-wifi-sparkfun.html

A última opção é a utilização do SoC ESP8266 da empresa Espressif. Este módulo contém um microcontrolador, pinos digitais de IO, a antena e circuitos de radiofrequência, amplificadores e reguladores de tensão e corrente. Por ser um circuito altamente integrado, suas dimensões físicas são reduzidas. O microcontrolador integrado de 32 bits faz o processamento dos dados enviados e recebidos da internet, e ainda disponibiliza cerca de 80% de sua capacidade para o desenvolvedor utilizar em sua aplicação. Este módulo ainda pode ser utilizado com as bibliotecas compatíveis com Arduino, tornando sua utilização mais rápida no projeto. Seus pontos fortes são, então, a fácil integração no projeto, bibliotecas de fácil acesso e uso, não necessitar de fios, possuir tamanho reduzido, além de ter custo baixo, cerca de U$6,95[[3]](#footnote-3) e um microcontrolador integrado. Seus pontos fracos, no entanto, incluem o módulo ter sido introduzido no mercado por uma empresa até então não muito conhecida, tendo o risco de possuir erros de projetos não descobertos ou corrigíveis.

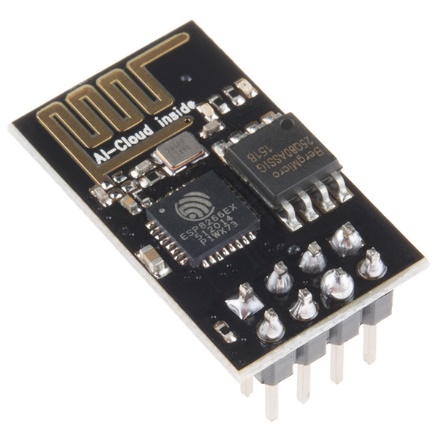


Figura - O módulo ESP8266

Fonte: https://www.sparkfun.com/products/13678

Para ilustrar as principais diferenças entre as opções e evidenciar a escolha tomada foi feita uma tabela comparativa:

|  |  |  |  |
| --- | --- | --- | --- |
| Critério/Opção | Módulo *Ethernet* W550 | Módulo Wireless CC3300 | Módulo SoC ESP8266 |
| Facilidade de integração no projeto | Baixa: necessita de maior espaço físico e cabeamento | Média: necessita de espaço físico e microcontrolador adicional | Alta: Solução altamente integrada |
| Bibliotecas | Fácil acesso | Fácil Acesso | Fácil Acesso |
| Maturidade | Alta | Alta | Média |
| Complexidade de hardware | Média | Média | Baixa |
| Capacidade de processamento | Baixa: Considerando a plataforma Arduino UNO (8 bits) | Baixa: Considerando a plataforma Arduino UNO (8 bits) | Média: Microcontrolador de 32 bits, mas apenas 80% disponível |
| Custo | Alto: U$44,95 | Alto: U$34,95 | Baixo: U$6,95 |

Tabela 1: Comparativo entre as opções de interfaces com a Internet

Considerando as opções e suas características, foi optado por utilizado o módulo baseado no SoC ESP8266, cujos detalhes são mostrados na seção a seguir.

### O SoC ESP8266

A Espressif é uma empresa de semicondutores com base na China, com o foco de produzir soluções altamente integradas e de baixo custo para a internet das coisas (IoT). Em meados de 2014, esta causou uma grande repercussão na internet pelo lançamento do SoC ESP8266, que integrava os recursos necessários para a comunicação *Wi-Fi* juntamente com um microcontrolador, tudo com poucos componentes externos, tamanho reduzido e com custo muito baixo. Desde então, vem lançando mais soluções focando em integração e preço reduzido.

|  |  |
| --- | --- |
| Especificações técnicas do ESP8266[13] | |
| CPU | Tensilica L106 32-bit micro controller |
| Periféricos | UART/SDIO/SPI/I2C/I2S/IR Remote Control |
| Tensão de operação | 2,5~3,6V |
| Corrente de operação | 80 mA |
| Temperatura de operação | -40C até 125 C |
| Encapsulamento | QFN32 (5mm x 5mm) |
| Padrões Internacionais | FCC/CE/TELEC/SRRC |
| Protocolos Wi-Fi | 802.11 b/g/n/e/i |
| Frequência Wi-Fi | 2.4 GHz a 2.4835 GHz |
| Potência máxima transmissão Wi-Fi | 20 dBm |
| Sensitividade máxima recepção WI-Fi | -91 dBm |

Tabela : Especificações Técnicas ESP8266

Neste projeto, foi escolhido um kit de desenvolvimento para ESP8266 da WEMOS ELECTRONICS, cujas vantagens incluem pinos de IO acessíveis, *led*, botões, conexão usb etc. Para facilitar o desenvolvimento, foi utilizado um *bootloader* baseado em Arduino, assim foi possível utilizar diversas bibliotecas já desenvolvidas e testadas para esta plataforma.

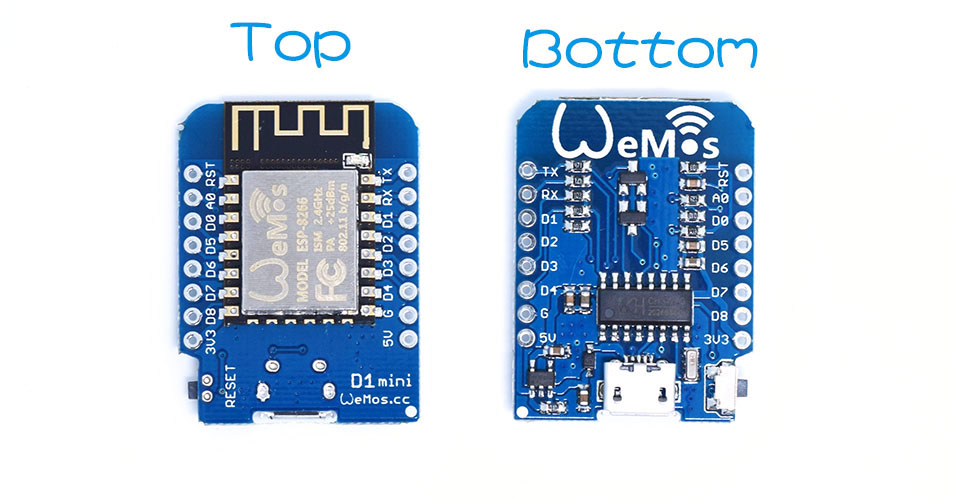


Figura : Kit de desenvolvimento WEMOS D1 mini (ESP8266)

Fonte: <https://www.wemos.cc/product/d1-mini.html>

A placa de desenvolvimento contém 11 pinos de IO digital, podendo ser reconfigurados para interrupção, PWM, I2C e one-wire, além de 1 pino de entrada analógica. Neste projeto, não foram necessários mais pinos ou periféricos externos, visto que toda a comunicação Wi-Fi e processamento já está integrada no ESP8266. Sendo assim, apenas leds indicadores e um botão de *reset* foram necessários externamente.

### Interface com Usuário

No desenvolvimento deste projeto, foi avaliada a necessidade e a aplicação de uma interface com o usuário. Tomando o sistema como um todo, o usuário que é monitorado deve interagir apenas com o dispositivo de pulso. Já os usuários que monitoram, não devem precisar mais do que um aplicativo para o uso diário da aplicação. Neste caso, uma interface física com o usuário fica encarregada apenas de configurar o dispositivo no momento inicial de utilização, não sendo necessário fazer isto mais de uma vez.

Tomando estas informações como base, foi decidido que um simples *web-server*, capaz de exibir uma página simples em um navegador de internet, seria a solução ideal, pois não demanda hardware externo, é fácil de implementar e simplifica o uso do sistema pelo usuário final.

### Web server

Um *web server* é um programa capaz de se comunicação com o meio exterior através do protocolo HTTP, usado para transmitir páginas na internet. Neste projeto, foi implementado um *web server* embarcado, isto é, que funciona em um microcontrolador com recursos reduzidos, utilizando bibliotecas do ESP8266.

Para o funcionamento correto do servidor, é necessário, primeiramente construir as páginas utilizando uma linguagem comum para este fim, como a HTML (*HyperText Markup Language*) e possíveis complementos através de *javascript*. A descrição da página desenvolvida para o *web server* pode ser encontrada na seção 3.3. Pelo fato de o *firmware* do microcontrolador não implementar um sistema de arquivos, as páginas são armazenadas em *strings* na memória.

O servidor é configurado no estado SETUP\_SERVER do *firmware*. Neste estado, é criada uma rede Wi-Fi sem senha que permite ao usuário se conectar e através do navegador acessar as páginas contidas na memória. Após a criação da rede, são atribuídas funções relativas as páginas existentes da memória, ou seja, quando o usuário digita o endereço IP da base, esta deverá retornar a página raiz. Após a configuração das páginas, o estado é mudado para WAIT\_USER.

No estado WAIT\_USER, o *firmware* procede em um laço infinito, sempre checando se houveram novas requisições por parte do usuário conectado. Quando o usuário digita o endereço IP no navegador, é enviada uma requisição que é tratada dentro deste laço. O tratamento correto é executar a função relacionada a página solicitada. O *firmware* trata basicamente duas requisições, sendo uma a página principal (raiz) da aplicação, contendo os campos para preenchimento das informações. Mais detalhes do funcionamento do *firmware* estão na seção 3.4.5.1.

Ao ser pressionado o botão “Conectar”, é enviada uma requisição ao *web server* contendo nos parâmetros as informações dos campos. É importante notar que parte da verificação é feita através da própria página, que é processada no navegador do usuário. Dessa forma os dados recebidos pelo microcontrolador já estão validados. Este apenas irá armazenar os dados na memória para posterior utilização.

### Indicadores Luminosos

Apesar do *web-server* proporcionar todas as funções e informações relacionadas a configuração do equipamento, é necessário também indicar o correto funcionamento da base de comunicação, da comunicação com o relógio e com a internet. Para isso foram adicionados 3 leds ao circuito da base a fim de indicar este funcionamento.

O led indicativo da base de comunicação indica em qual estado de operação a base se encontra. Os estados possíveis são: Configuração, Conexão e Operação. No estado de configuração, a base ainda não foi configurada através do navegador, portanto não há como se conectar ao relógio ou a internet. Neste caso, o led encontra-se apagado. No caso de a base já ter sido configurada, mas houve uma queda de energia e a conexão com a internet e o relógio está sendo realizada novamente, o led fica no estado de conexão, piscando. Caso a conexão ocorra, o led fica acesso, indicando o estado de operação. Caso o relógio ou a internet pare de responder, este led voltará ao estado de conexão.

O led relativo a conexão com a internet fica acesso ao ser realizada uma conexão bem-sucedida com a rede Wi-Fi configurada. Caso a base não esteja conectada a internet, o led fica apagado.

A indicação de comunicação com o relógio também é mostrada por um Led. Caso a comunicação esteja ocorrendo em intervalos normais, o led fica acesso. Caso a comunicação apresente erros consecutivos, o led se apaga. O led piscará durante a inicialização para indicar que o processo de sincronização da comunicação com o relógio está em andamento.

Os leds foram ligados ao ESP8266 apenas através de um resistor, não sendo necessário nenhum *buffer* visto que o microcontrolador consegue fornecer a corrente necessária[13]. O circuito é mostrado na seção 3.4.6.

### Comunicação com o relógio

No kit de desenvolvimento da Texas Instruments ez430-Chronos é fornecido uma placa cuja função é servir de um *access point* (ponto de acesso) para um computador. Dessa forma, é possível comunicar-se através de uma porta USB com o relógio através do computador. No projeto, não é desejado a dependência de um computador para efetuar a comunicação com a internet e o dispositivo. Sendo assim, optou-se por reprogramar o microcontrolador presente nesta placa para comunicar-se através de uma interface serial com o ESP8266, efetivamente tornando o uso do computador desnecessário na aplicação em campo.

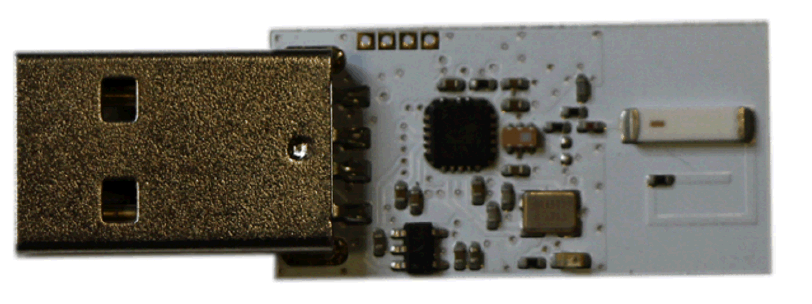


Figura : Foto da placa *access point* do Kit de desenvolvimento ez430-Chronos

Fonte: Manual do kit de desenvolvimento, p63)

### Circuito

O *access point* do kit de desenvolvimento adquirido tem como base o microcontrolador MSP430F5509 conectado a um circuito integrado CC1101, que tem a função de modular o sinal para uma frequência de 915MHz, utilizada na comunicação com o relógio.

O microcontrolador MSP430F5509 possui um *driver* dedicado a comunicação USB para o computador, que no caso deste projeto não seria mais utilizado. Através da inspeção do *datasheet* do microcontrolador e do esquemático da placa, foi constatado que havia uma interface UART disponível para uso, sendo necessário apenas programa-la no firmware e conecta-la ao ESP8266.

Através da comunicação serial com o ESP8266, foi possível separar o processamento dos dados referente a comunicação com o relógio e com a internet. O *access point* comunica-se com o relógio por rádio frequência, salvando os dados internamente. Ao ser requisitado, este repassa os dados ao ESP8266 que enviará para a internet e executará as ações necessárias, como envio de SMS, e-mail, etc. Uma explicação mais detalhada do funcionamento do *firmware* do *access point* é feita na seção 3.4.5.2.

### Protocolo

A comunicação entre o *access point* e o ESP8266 ocorre serialmente, sendo o ESP8266 o responsável por requisitar os dados. Este trabalha através de um sistema de *polling*, ou seja, faz a requisição de dados com um intervalo pré-definido e caso haja dados novos a serem enviados, os processa e os envia. Foi então definido um protocolo a ser seguido, com um identificador de comando, os dados, separadores e identificadores de fim de transmissão.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Comandos e Respostas da Comunicação entre *Access Point* e ESP8266 | | | | | | | | | | | | | | | | |
| Comando “Ler dados” ESP8266 -> *Access Point* | | | | | | | | | | | | | | | | |
| 52 | | 0D | |  | | | | | | | | | | | | |
| R | | ‘\r’ | |
| Resposta *Access Point* -> ESP8266 | | | | | | | | | | | | | | | | |
| 52 | XX | | XX | | XX | 2C | XX | XX | XX | 2C | XX | XX | XX | 2C | XX | XX |
| R | Bateria (%) | | | | | , | Sinal (%) | | | , | Temp. ºC | | | , | Bat. Car. | |
| XX | 2C | | XX | | XX | 2C | XX | XX | 2C | XX | XX | 2C | XX | XX | XX | XX |
| bpm | , | | Alarme | | | , | Dia | | , | Mês | | . | Ano | | | |
| 2C | XX | | XX | | 2C | XX | XX | 2C | XX | XX | 2C | XX | XX | 0D |  | |
| , | Hora | | | | , | Minuto | | , | Segundo | | , | Dia sem. | | ‘\r’ |
| Comando “Enviar data” ESP8266 -> *Access Point* | | | | | | | | | | | | | | | | |
| 44 | XX | | XX | | 2C | XX | XX | 2C | XX | XX | XX | XX | 2C | XX | XX | 2C |
| D | Dia | | | | , | Mês | | , | Ano | | | | , | Hora | | , |
| XX | XX | | 2C | | XX | X | 2C | XX | XX | 0D |  | | | | | |
| Minuto | | | , | | Segundo | | , | Dia sem. | | ‘\r’ |
| Resposta *Access Point* -> ESP8266 | | | | | | | | | | | | | | | | |
| 44 | 0D | |  | | | | | | | | | | | | | |
| D | ‘\r’ | |
| Comando “Enviar hora de remédio” ESP8266 -> *Access Point* | | | | | | | | | | | | | | | | |
| 52 | XX | | XX | | 3A | XX | XX | 2C | XX | XX | 3A | XX | XX | 2C | XX | XX |
| H | Hora | | | | : | Minuto | | , | Hora | | : | Minuto | | , | Hora | |
| 3A | XX | | XX | | 2C | XX | XX | 3A | XX | XX | 2C | XX | XX | 3A | XX | XX |
| : | Minuto | | | | , | Hora | | : | Minuto | | , | Hora | | : | Minuto | |
| 2C | XX | | XX | | 3A | XX | XX | 2C | XX | XX | 3A | XX | XX | 2C | XX | XX |
| , | Hora | | | | : | Minuto | | , | Hora | | : | Minuto | | , | Hora | |
| 3A | XX | | XX | | 0D |  | | | | | | | | | | |
| : | Minuto | | | | ‘\r’ |
| Resposta *Access Point* -> ESP8266 | | | | | | | | | | | | | | | | |
| 52 | 0D | |  | | | | | | | | | | | | | |
| H | ‘\r’ | |

Tabela : Protocolo de Comunicação entre *Access Point* e ESP8266

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Comandos e Respostas da Comunicação entre *Access Point* e Relógio | | | | | | | | | | | | | | | | | | | | | | | | | |
| Comando “Enviar Dados” Relógio -> *Access Point* | | | | | | | | | | | | | | | | | | | | | | | | | |
| 52 | XX | XX | XX | | 2C | | XX | | XX | | XX | | 2C | | XX | | XX | | XX | | 2C | | XX | | XX |
| R | Bateria (%) | | | | , | | Sinal (%) | | | | | | , | | Temp. ºC | | | | | | , | | Bat. Car. | | |
| XX | 2C | XX | XX | | 0D | |  | | | | | | | | | | | | | | | | | | |
| bpm | , | Alarme | | | ‘\r’ | |
| Resposta “Enviar dados” *Access Point* -> Relógio | | | | | | | | | | | | | | | | | | | | | | | | | |
| XX | XX | 0D |  | | | | | | | | | | | | | | | | | | | | | | |
| Alarme | | ‘\r’ |
| Comando “Enviar data” Relógio -> *Access Point* | | | | | | | | | | | | | | | | | | | | | | | | | |
| 44 | XX | XX | 2C | | XX | | XX | | 2C | | XX | | XX | | XX | | XX | | 2C | | XX | | XX | | 2C |
| D | Dia | | , | | Mês | | | | , | | Ano | | | | | | | | , | | Hora | | | | , |
| XX | XX | 2C | XX | | X | | 2C | | XX | | XX | | 0D | |  | | | | | | | | | | |
| Minuto | | , | Segundo | | | | , | | Dia sem. | | | | ‘\r’ | |
| Resposta *Access Point* -> ESP8266 (se necessário atualizar data/hora) | | | | | | | | | | | | | | | | | | | | | | | | | |
| 44 | XX | XX | | 2C | | XX | | XX | | 2C | | XX | | XX | | XX | | XX | | 2C | | XX | | XX | |
| D | Dia | | | , | | Mês | | | | , | | Ano | | | | | | | | , | | Hora | | | |
| XX | XX | 2C | | XX | | X | | 2C | | XX | | XX | | 0D | |  | | | | | | | | | |
| Minuto | | , | | Segundo | | | | , | | Dia sem. | | | | ‘\r’ | |
| Comando “Enviar hora de remédio” *Access Point* -> Relógio | | | | | | | | | | | | | | | | | | | | | | | | | |
| 52 | XX | XX | XX | | XX | | 2C | | XX | | XX | | XX | | XX | | 2C | | XX | | XX | | XX | | XX |
| H | Hora | | Minuto | | | | , | | Hora | | | | Minuto | | | | , | | Hora | | | | Minuto | | |
| 2C | XX | XX | XX | | XX | | 2C | | XX | | XX | | XX | | XX | | 2C | | XX | | XX | | XX | | XX |
| , | Hora | | Minuto | | | | , | | Hora | | | | Minuto | | | | , | | Hora | | | | Minuto | | |
| 2C | XX | XX | XX | | XX | | 2C | | XX | | XX | | XX | | XX | | 0D | |  | | | | | | |
| , | Hora | | Minuto | | | | , | | Hora | | | | Minuto | | | | ‘\r’ | |
| Resposta *Access Point* -> ESP8266 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 52 | 0D |  | | | | | | | | | | | | | | | | | | | | | | | |
| H | ‘\r’ |

Tabela : Protocolo de comunicação entre *Access Point* e Relógio

### Comunicação com a Internet

### Princípio de Funcionamento

A base se comunica com a internet inteiramente através do ESP8266 pelo fato de possuir *hardware* dedicado para isto e bibliotecas de fácil uso. A comunicação é feita através de uma conexão Wi-Fi, que neste projeto seria a rede presente na casa da pessoa monitorada. Através principalmente de requisições *web*, a base recebe e envia dados para a internet.

### Thingspeak

O Thingspeak[[4]](#footnote-4) é uma plataforma voltada para a Internet das Coisas (IoT) que permite armazenar dados na nuvem e desenvolver aplicações em IoT. Por ter sido adquirida pela MathWorks, esta plataforma também está integrada ao programa MATLAB, utilizado frequentemente em pesquisas e desenvolvimento. Através desta plataforma, é possível configurar facilmente dispositivos para enviar e receber dados utilizando um protocolo comum, visualizar os dados coletados, programar eventos, dentre outras funções, sem ter a necessidade de configurar servidores ou desenvolver software adicional. A plataforma é gratuita para usuários de teste e possui limitações que podem ser removidas em versões pagas.

A plataforma funciona através de canais. Os canais são conjuntos de dados identificados por um número único e formados por um conjunto de campos, sendo eles o nome do projeto, descrição, oito campos genéricos de dados, um campo de metadados, *tags*, *website*, altitude, latitude, longitude e um vídeo de apresentação.

Para inserir dados em um canal, são utilizadas requisições *web* para o servidor da Thingspeak, através de uma API (*Application Programming Interface*). Estas requisições são capazes de criar novos canais, adicionar e remover dados, excluir um canal etc.

### Utilização dos campos do canal

Por serem genéricos, os campos dos canais do Thingspeak podem ser utilizados da melhora forma para cada tipo de projeto. Neste trabalho, foram utilizados os campos de Identificação, Nome, Descrição, 4 campos genéricos e o campo de metadados. A tabela a seguir mostra sucintamente os campos e suas funções, enquanto que o campo de metadados, mais complexo, é explicado posteriormente.

|  |  |
| --- | --- |
| Campo | Função |
| ID | Armazena a identificação única do canal. É utilizado como o código de acesso no aplicativo. |
| Nome | Descrição do dispositivo. Contém “SmartCare Device <numero>”, onde <numero> é um identificador do conjunto base e relógio. |
| Descrição | Contém o nome da pessoa sendo monitorada |
| Campo 1 | Armazena o nível de bateria (em porcentagem) |
| Campo 2 | Armazena o nível de sinal do relógio (em porcentagem) |
| Campo 3 | Armazena a temperatura ambiente lida pelo relógio (em graus Celcius) |
| Campo 4 | Armazena os batimentos cardíacos lidos (em Batidas por Minuto) |
| Metadados | Contém informações relativas ao contato das pessoas que recebem os alertas, o alerta vigente e os horários de remédio. |

Tabela : Campos de dados utilizados no canal do Thingspeak

O campo de metadados foi escolhido como a solução para armazenar os dados de contato do aplicativo, visto que é possível armazenar qualquer tipo de texto nesse campo. Dessa forma, pode-se alterar as informações pelo aplicativo, que serão lidas pela base de comunicação e atualizadas, assim tendo uma garantia de que caso as informações de contato mudem, o aplicativo continuará exibindo os alertas. A forma de implementar esta solução foi através de um texto contendo caracteres de separação, para identificar posteriormente em quais posições do texto os dados estão. O caractere de separação é a vírgula, e o formato do texto é mostrado a seguir:

<nome do primeiro monitor>,<nome do segundo monitor>,<e-mail do primeiro monitor>,<e-mail do segundo monitor>,<twitter do primeiro monitor>,<twitter do segundo monitor>,<telefone do primeiro monitor>,<telefone do segundo monitor>,<código de alarme atual>,<hora de remédio 1>,<hora de remédio 2>,<hora de remédio 3>,<hora de remédio 4>,<hora de remédio 5>,<hora de remédio 6>,<hora de remédio 7>,<hora de remédio 8>,<campo reservado (0)>

Assim, na configuração do dispositivo, os dados existentes são enviados para o canal na nuvem, que os armazena. Ao cadastrar o canal no aplicativo, este irá recuperar do servidor do Thingspeak os mesmos dados e exibir ao usuário.

### Descrição das requisições utilizadas

Para atingir os objetivos do projeto, foram necessários apenas três tipos de requisições das mais de 15 disponíveis, por isso serão explicadas apenas as mais relevantes.

A criação de um canal pode ser feita manualmente através do site, porém era necessário que isso fosse feito automaticamente durante a configuração do dispositivo. A Thingspeak fornece uma requisição que possibilita a criação de um novo canal, retornando os dados do mesmo caso tenha sucesso. Esta requisição se dá enviando uma requisição HTTP POST para https://api.thingspeak.com/channels, fornecendo como parâmetros a chave de acesso geral e os campos explicitados na seção anterior. Na resposta da requisição, será informado o identificador do canal e as chaves únicas deste canal, utilizadas para a atualização de informações e envio de dados.

No caso de o usuário querer atualizar as informações de contato ou um alarme ocorrer, estas informações devem ser enviadas através de uma requisição específica. Isso se dá através do envior de uma requisição HTTP PUT para https://api.thingspeak.com/channels/CHANNEL\_ID, substituindo CHANNEL\_ID pelo identificador gerado na criação do canal. É necessário fornecer a chave de acesso geral e as informações dos campos a serem atualizados. A resposta indica se a atualização foi realizada com sucesso.

A cada vez que o relógio envia um novo dado, este é enviado ao servidor da Thingspeak para ser armazenado o consultado posteriormente. Para enviar os dados, é utilizada uma requisição específica, do tipo HTTP POST para http://api.thingspeak.com/update, fornecendo a chave de acesso do canal e as informações dos campos a serem atualizadas. A reposta indica se a atualização foi realizada com sucesso.

### IFTTT

IFTTT[[5]](#footnote-5) é o acrônimo para “*If this, then that”* que é traduzido como “se isto, estão aquilo”, premissa desta plataforma online, utilizada para conectar dois serviços online de maneira arbitrária. O funcionamento básico se resumo em conectar um evento específico de alguma plataforma com uma ação, que pode também ser relacionada a alguma plataforma, desde que suportada pela plataforma.

Neste projeto, foram utilizados quatro tipos de eventos que geram uma ação correspondente, sendo elas: enviar *tweet*, enviar SMS, enviar e-mail de emergência e enviar e-mail de configuração.

Para enviar um *tweet*, utilizado para finalizar a configuração do equipamento e também para alertar os usuários sobre uma possível ocorrência com a pessoa monitorada, foi conectado o serviço da rede social *Twitter*, cujo usuário foi criado especialmente para este projeto. O funcionamento se dá primeiramente pelo envio de uma requisição HTTP POST para http://maker.ifttt.com/trigger/send\_tweet/with/key/KEY, substituindo KEY pela chave de acesso da plataforma e fornecendo nos parâmetros da requisição o usuário de destino e a mensagem a ser enviada. Ao receber esta requisição, o servidor IFTTT repassa o usuário e a mensagem para o *Twitter*, que então posta esta mensagem na rede social.

O envio de SMS funciona de maneira análoga, com a diferença que o serviço de mensagens SMS é fornecido pelo *Twillio[[6]](#footnote-6)*. Esta plataforma possibilita o envio de mensagens SMS de maneira gratuita para testes, tendo mais funções nas versões pagas. Para o projeto, o plano gratuito se mostrou suficiente para as necessidades de um protótipo, visto que a quantidade de mensagens é baixa e é adicionado um texto indicando que não é uma versão paga. O serviço foi conectado ao IFTTT e para efetivamente enviar uma SMS, é necessário fazer uma requisição HTTP POST para http://maker.ifttt/com/trigger/send\_sms/with/key/KEY, substituindo KEY pela chave de acesso da plataforma e enviando nos parâmetros da requisição o número de telefone e a mensagem. O servidor IFTTT então repassa o número de telefone e a mensagem para o *Twillio*, que envia a mensagem ao usuário, tanto no momento da configuração quanto no alerta de uma ocorrência.

O envio de e-mail nas situações de configuração e uma emergência ocorre através da conexão com o servidor de e-mail Gmail[[7]](#footnote-7), da Google. Através de uma conta criada especialmente para o projeto, é possível enviar mensagens para os usuário do aplicativo de forma automática. O serviço foi conectado ao IFTTT e para enviar a mensagem, é necessário fazer uma requisição HTTP POST para http://maker.ifttt/com/trigger/send\_email/with/key/KEY, no caso de configuração ou http://maker.ifttt/com/trigger/emergency/with/key/KEY no caso de emergência, substituindo KEY pela chave de acesso da plataforma. Os parâmetros variam de acordo com o tipo de e-mail a ser enviado. Para o e-mail de configuração, são enviados o endereço do destinatário, o nome do usuário e o código de ativação. Para o e-mail de alerta de ocorrência, os parâmetros são o destinatário, o nome da pessoa monitorada e a mensagem de emergência.

### NTP server

NTP é o acrônimo para *Network Time Protocol*, utilizado para a sincronização de relógios em diversos dispositivos com acesso à internet. Seu princípio de funcionamento parte da utilização de uma referência de tempo muito estável, como por exemplo a utilizada no GPS (*Global Positioning System*). Através do protocolo, é possível receber as informações de tempo de servidores dedicados para este fim, sendo essas informações utilizadas para atualizar a data e hora de um dispositivo.

Neste projeto, a utilização de um servidor NTP se mostrou interessante por ser possível atualizar automaticamente a data e hora do relógio, não necessitando de configuração alguma por parte do usuário, que também poderia ser feita de maneira incorreta. Sendo assim, foram utilizadas bibliotecas de relógio e calendário do microcontrolador, sendo a referência atualizadas através do servidor NTP. Caso a data e hora enviadas pelo relógio estejam defasadas em mais de 5 minutos, o relógio é atualizado automaticamente.

### Firmware

### Firmware do ESP8266

O *firmware* do ESP8266 tem como principal função fazer a ponte entre os dados do relógio, através do *Access Point,* e a internet. Pelo fato do ESP8266 possuir já implementado as funções e *hardware* de comunicação Wi-Fi, o foco principal foi a integração dos serviços da internet com o relógio.

Inicialmente a execução do programa foi dividida em quatro estados principais, cujas funções diferiam entre si, sendo eles um de inicialização, outro de configuração, o de conexão com a internet e o de operação principal. Os dados indispensáveis para a operação do programa, como os dados dos usuários para notificações e as informações dos estados, possuem salvamento na EEPROM do microcontrolador, sendo assim não é necessário reconfigurar os parâmetros de operação quando a base for desligada ou perder energia. A sequência de execução simplificada do *firmware* pode ser vista na Figura 21.

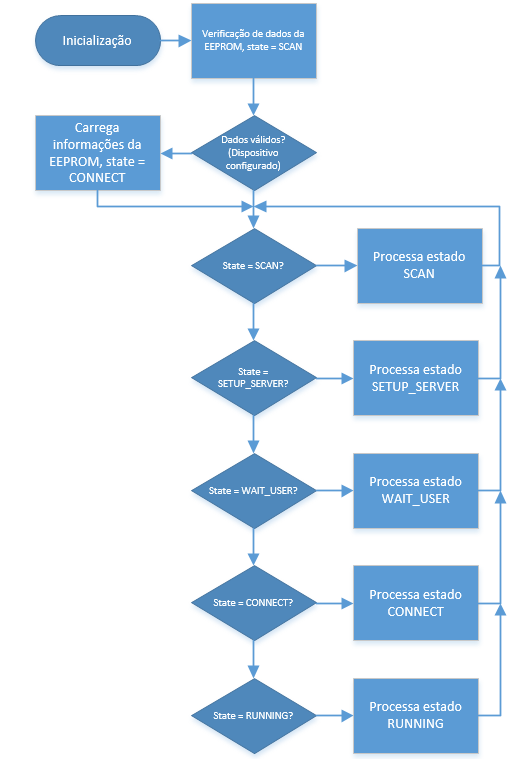


Figura : Fluxograma geral do *firmware* do ESP8266

Fonte: Autoria própria

O estado SCAN é responsável por fazer uma varredura nas redes Wi-Fi disponíveis na região de instalação da base. Para isto, o ESP8266 é colocado no modo AP (*Access Point*) que permite procurar por redes e se conectar a uma rede conhecida. A varredura das redes disponíveis é importante para oferecer ao usuário uma referência no momento da configuração, onde uma rede deve ser selecionada para a conexão definitiva. Ao verificar as redes disponíveis, o *firmware* salva o nome das mesmas na memória para serem recuperadas na exibição da página de configuração. Logo após o estado é mudado para o próximo, SETUP\_SERVER. O Fluxograma deste estado é apresentado na Figura 22.

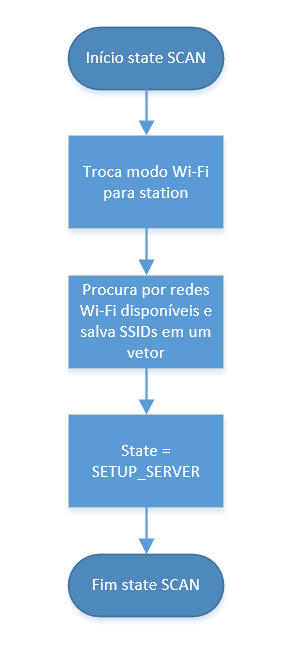


Figura : Fluxograma do estado SCAN

Fonte: Autoria própria

No estado SETUP\_SERVER, o *firmware* já tem salvo o nome das redes Wi-Fi presentes na região da base. A sua principal função é desativar o modo AP, criar uma rede Wi-Fi e iniciar o Web-server utilizado na configuração do dispositivo. A criação de uma rede Wi-Fi própria é feita através da biblioteca do ESP8266, fornecendo o nome da rede e a senha de acesso, neste caso, inexistente para facilitar a configuração. Após a criação da rede Wi-Fi, o web server é iniciado no endereço 192.168.4.1, e então o ESP8266 irá redirecionar todas as conexão neste endereço IP para o web server. Após a correta inicialização, o estado é mudado para WAIT\_USER. O fluxograma deste estado é mostrado na Figura 23.

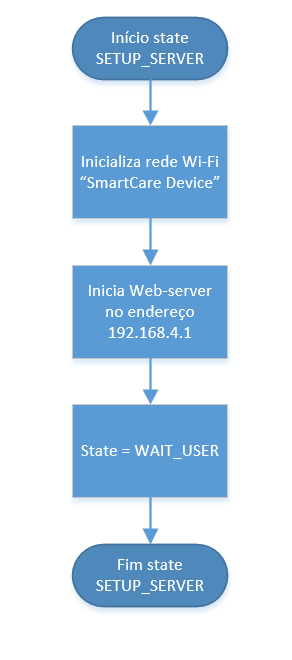


Figura : Fluxograma do estado SETUP\_SERVER

Fonte: Autoria própria

No estado WAIT\_USER, o programa fica em um laço infinito até que seja interrompido por uma ação válida por parte do usuário. Como o *web server* já está funcionando, o *firmware* irá apenas tratar as requisições feitas pelo usuário. Existe uma validação parcial dos dados embutida na página, que garante que eles serão íntegros ao chegar no microcontrolador. Apesar disso, é feita uma verificação e o salvamento no formato necessário ao processamento posterior pelo programa. Quando o usuário clica no botão “Configurar” da página, os dados serão salvos no formato apropriado e o próximo estado irá controlar a conexão com a rede e a criação do canal. Caso estejam incorretos, a página inicial será recarregada. O próximo estado tem denominação CONNECT. A Figura 24 a seguir demonstra o funcionamento do estado WAIT\_USER.

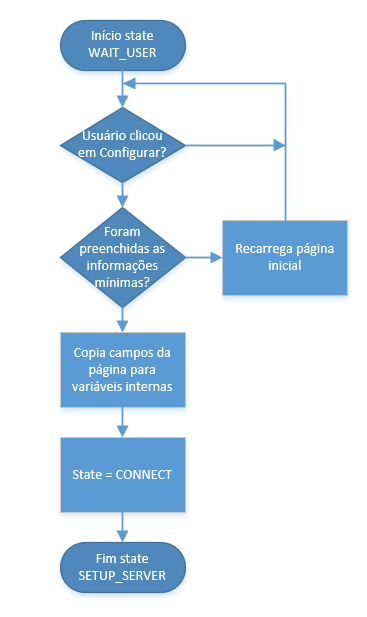


Figura : Fluxograma do estado WAIT\_USER

No estado CONNECT, será feita a conexão com a rede Wi-Fi configurada e a criação do canal de dados do Thingspeak, caso seja necessário. Através das bibliotecas do ESP8266, é feita a conexão com a rede Wi-Fi, sendo suportada inclusive redes protegidas por senha. Em caso de falha, o programa voltará ao estado SETUP\_SERVER, sendo necessário reconfigurar a conexão. Em caso de sucesso, a base irá se conectar ao servidor do Thingspeak e verificar se já existe um canal para este dispositivo. Em caso negativo, será enviada uma requisição para a criação do canal com os parâmetros configurados na página de configuração. Logo após os dados de confirmação forem recebidos, serão verificados e e-mails, SMS e mensagens no Twitter serão enviadas com o número do canal, utilizado para o uso do aplicativo *web*. Caso contrário, será verificada a conexão com o mesmo.

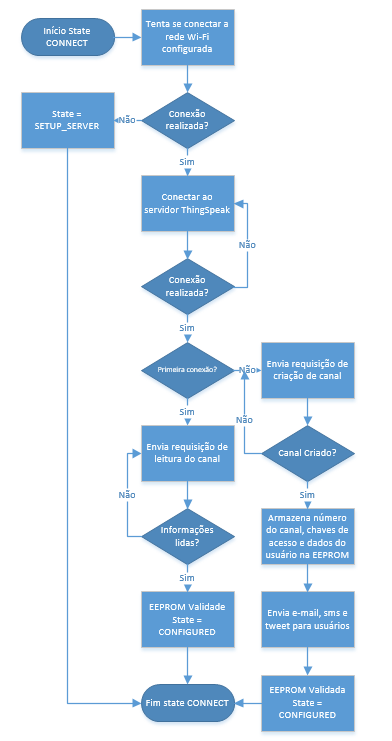


Figura : Fluxograma do estado CONNECT

Fonte: Autoria própria

Após a conexão com o canal acontecer, a EEPROM será considerada como válida pelo *firmware*, logo após a gravação dos dados. O programa irá para o CONFIGURED, que é principal estado de execução. O fluxograma do estado CONNECT é mostrado na Figura 25.

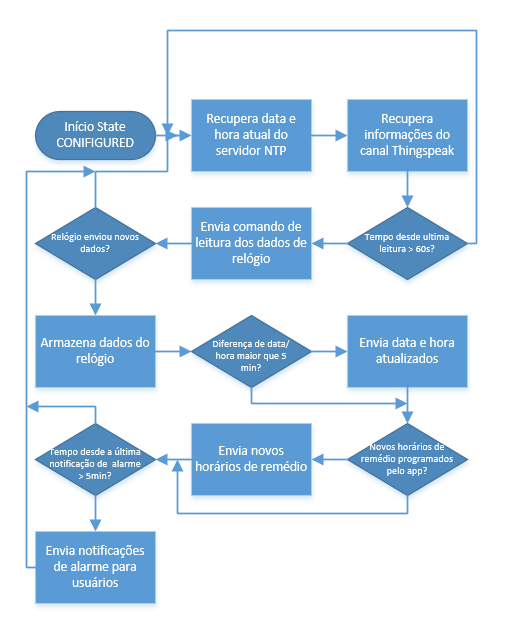


Figura : Fluxograma do estado CONFIGURED

Fonte: Autoria própria

O estado em que o *firmware* deverá ficar na maior parte do tempo em operação é o CONFIGURED. Nele, não é necessária nenhuma intervenção por parte do usuário, pois a função neste caso é repassar os dados do relógio para o servidor na internet. O estado funciona em um laço infinito, iniciando com a atualização da data e hora, vinda do servidor NTP. Com a data e hora atualizadas, é feita uma consulta ao canal do Thingspeak para verificar se houve alguma alteração nos dados de contato ou se houve um pedido para zerar os alarmes do relógio. A cada um minuto, é enviado um comando ao *Access point* requisitando os últimos dados do relógio. Caso o relógio não tenha enviado novos dados, o programa irá para o início do laço. Caso existam novos dados enviados pelo relógio, os mesmo são armazenados na memória e no enviados ao Canal do Thingspeak. Se a diferença de data e hora forem maior que 5 minutos, será enviada a data a hora corretas para o relógio. Existe a possibilidade de novos horários de remédio terem sido programados através do aplicativo. Neste caso estes também serão enviados ao relógio, para que soe um alarme nos horários programados. Logo após, por padrão, é enviada uma notificação a cada 5 minutos, se houver, até que o alarme seja desativado pelo usuário do aplicativo, assinalando que a emergência foi solucionada. O programa então aguarda um minuto para reiniciar o laço de execução. A Figura 26 demonstra o funcionamento deste estado.

### Firmware do Access Point MSP430

O *firmware* desenvolvido para o *Access Point* foi baseado no programa de exemplo da Texas Instruments, que realiza a comunicação com o *firmware* original do relógio. Algumas funções foram retiradas por não ser necessárias para este projeto, como a comunicação USB, entretanto, outra foram adicionadas, como a comunicação serial com o ESP8266 e de data e hora através do RTC (*Real Time Clock*).

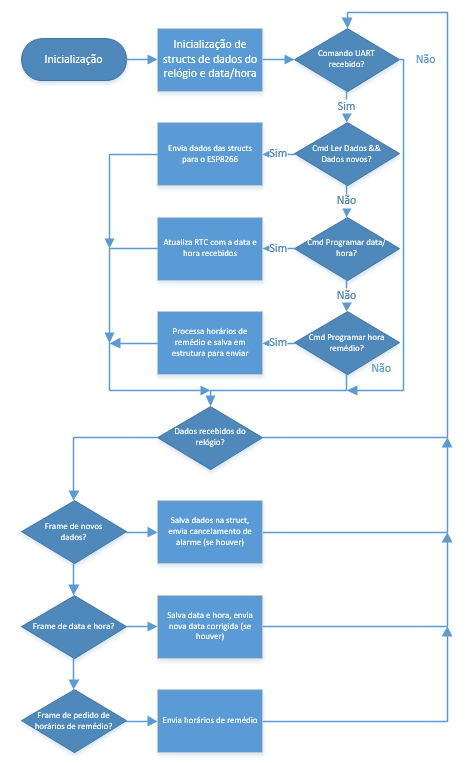


Figura : Fluxograma do *firmware* do *Access Point*

A comunicação serial foi implementada através do estudo dos pinos de IO disponíveis na placa do *Access Point*. O Microcontrolador MSP430F5555, utilizado neste projeto, possui um dos módulos de comunicação serial ligados aos pinos 2 e 3, que possuem acesso dedicado na placa através de um *pad*. O microcontrolador foi configurado para utilizar a comunicação serial nestes pinos, através do *driver* dedicado para este fim, utilizando interrupções para o seu funcionamento.

O módulo de calendário do microcontrolador foi utilizado com o fim de manter a data e hora sempre atualizados. Quando o relógio envia a sua data e hora interna, o *Access Point* grava estes dados e continuar contando o tempo até o ESP8266 enviar o comando de leitura. Assim, a data enviada está sempre atualizada com a do relógio. No caso de não estar atualizada com o servidor de horário global, o ESP8266 enviar a data e hora reais para o *Access Point.* Este agora armazena a data e hora atualizados e corretos, e ao se comunicar com o relógio, envia os dados atualizados, que serão então usados no relógio.

O *firmware* inicia com a inicialização das variáveis globais e dos módulos de comunicação, relógio e comunicação sem fio. Após a inicialização inicia-se um laço infinito onde é verificado se houve comandos dos ESP8266, se é necessário algum processamento, e se houve comunicação sem fio por parte do relógio. O *Access point* funciona principalmente para fazer a ligação entre o relógio com a internet, feita no ESP8266. Os principais processamentos são feitos para salvar os dados a serem enviados e enviar os dados para os destinos corretos. Uma representação em fluxograma do *firmware* pode ser vista na Figura 27.

### Hardware

O hardware desenvolvido para o projeto consistiu principalmente na integração dos kits de desenvolvimento utilizados, como o *Access Point*  da *Texas Instruments*, e do ESP8266. Os únicos adendos realizados foram os *leds* indicadores da base. O circuito é alimentado por uma fonte comercial de 5V. Os circuitos que operam no nível de tensão de 3,3V possuem reguladores internos. O diagrama pode ser visto na figura abaixo.

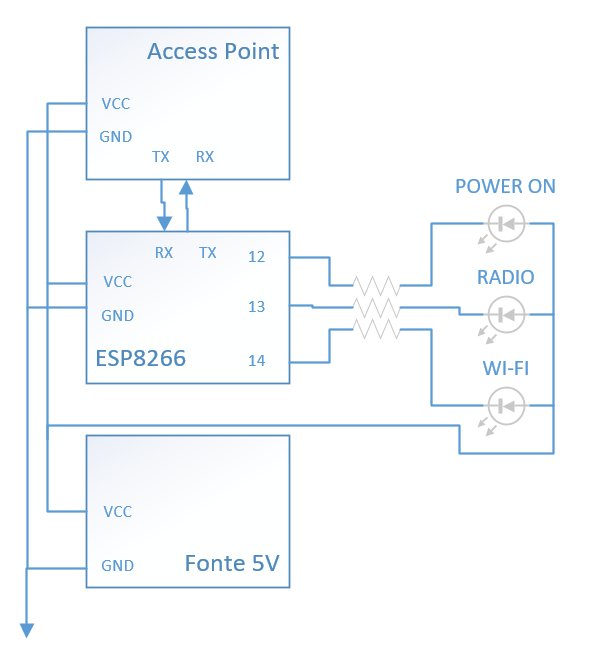


Figura : Esquemático do circuito da Base

Fonte: Autoria própria

### DESENVOLVIMENTO DO RELÓGIO

### Definição do kit de desenvolvimento

### Implementação do sensor de batimento cardíacos

### Implementação do sensor de queda

### Implementação do alarme para remédios

### Implementação do botão de pânico

## RESULTADOS

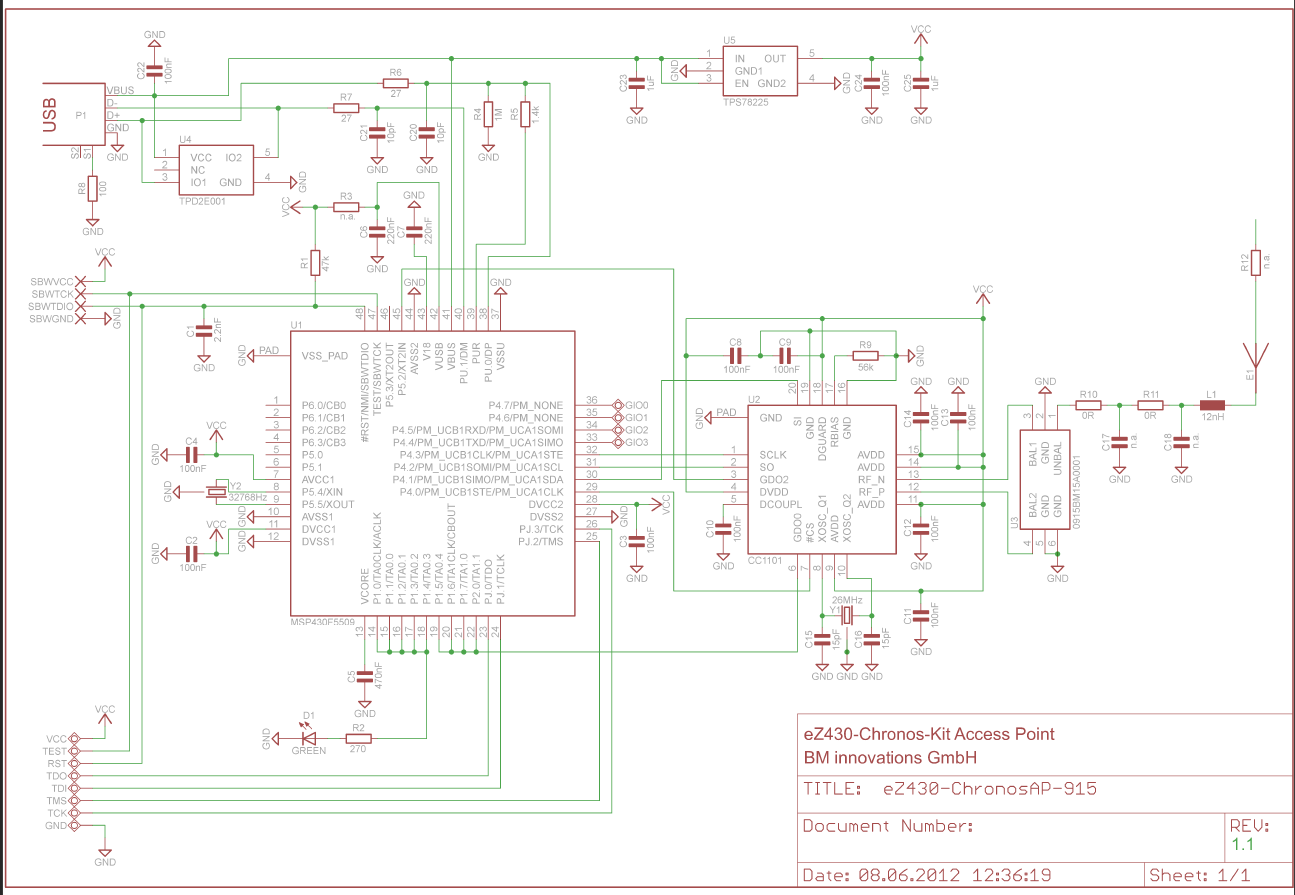
## CONSIDERAÇOES FINAIS

Parte final do texto, na qual se apresentam as conclusões do trabalho acadêmico, usualmente denominada considerações finais. Pode ser usada outra denominação similar que indique a conclusão do trabalho.

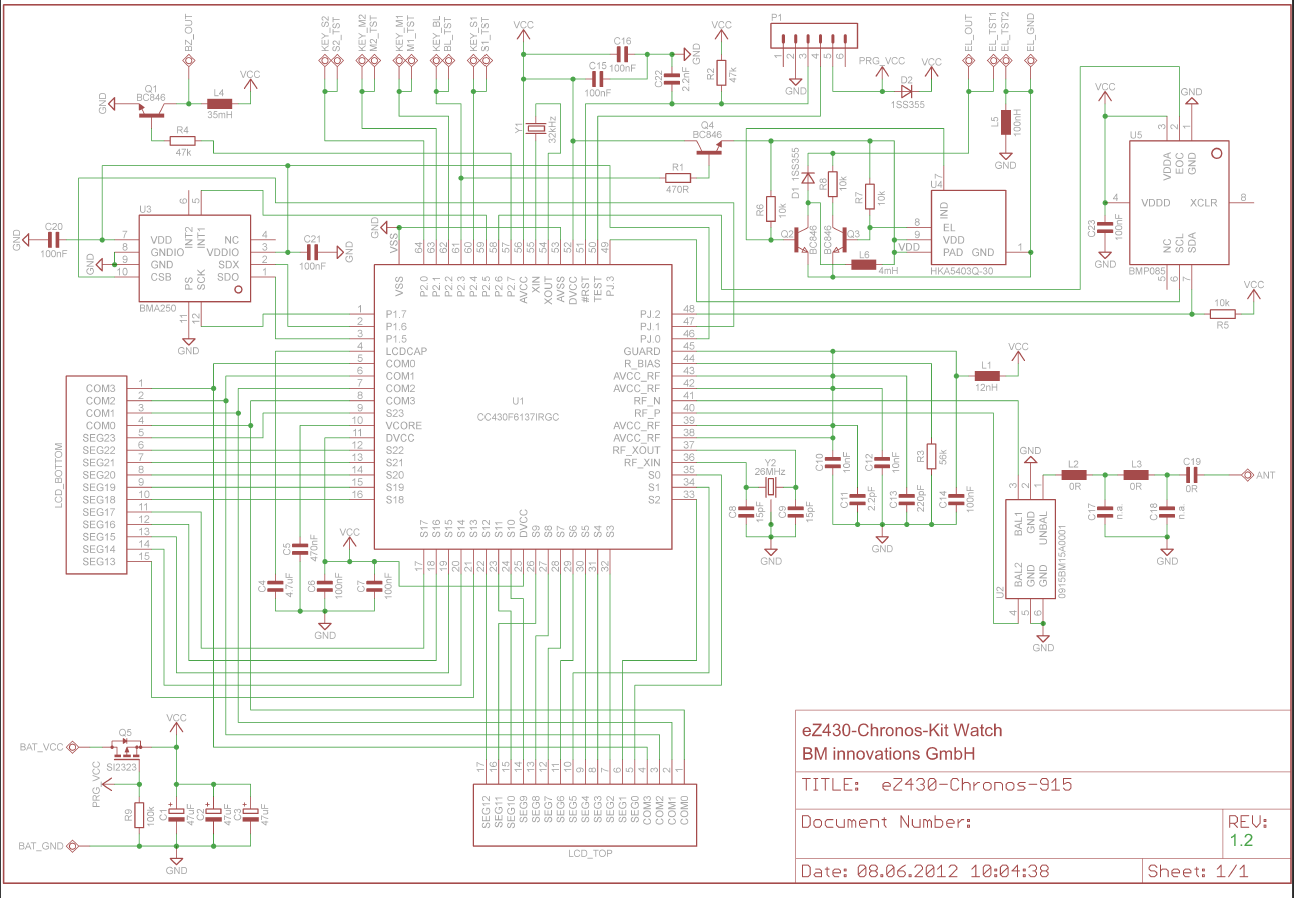
**REFERÊNCIAS**

1. Esp8266. Disponível em http://www.esp8266.com/wiki/doku.php Acesso em 17/08/2016.
2. Cláudia Collucci. Número de idosos que moram sozinhos triplica em 20 anos. Disponível em http://www1.folha.uol.com.br/equilibrioesaude/2013/12/1389765-numero-de-idosos-que-moram-sozinhos-triplica-em-20-anos.shtml Acesso em 31/05/2016.
3. A Crítica. Idosos e crianças são maiores vítimas dos acidentes domésticos. Disponível em http://acritica.uol.com.br/vida/Manaus-Amazonas-Amazonia-Idosos-criancas-maiores-acidentes-domesticos\_0\_1214878524.html Acesso em 01/05/2016.
4. Assessoria de Imprensa Hospital São Luiz. Velocidade do atendimento após o infarto diminui danos. Disponível em http://coracaoalerta.com.br/fique-alerta/velocidade-do-atendimento-apos-o-infarto-diminui-dano Acesso em 31/05/2016.
5. Texas Instruments. Chronos: Wireless development tool in a watch. Disponível em http://www.ti.com/tool/ez430-chronos Acesso em 17/08/2016.
6. Berlindes Astrid Kuchemann. Envelhecimento populacional, cuidado e cidadania: velhos dilemas e novos desafios. Sociedade e Estado, 27:165 – 180, 04 2012.
7. Expectativa de vida do brasileiro ao nascer é de 75,5 anos, diz IBGE. Disponível em http://g1.globo.com/bemestar/noticia/expectativa-de-vida-do-brasileiro-ao-nascer-e-de-755-anos-diz-ibge.ghtml Acesso em 11/05/2017.
8. OLIVEIRA NETTO, A. A. de. **Metodologia da pesquisa científica:** guia prático para a apresentação de trabalhos acadêmicos. 3. ed. rev. e atual. Florianópolis: Visual Books, 2008.
9. UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ. Sistema de Bibliotecas. **Normas para elaboração de trabalhos acadêmicos.** Curitiba: UTFPR, 2009.
10. L. Yang, M. Yuan, W. Wang, Q. Zhang and J. Zeng, "Apps on the move: A fine-grained analysis of usage behavior of mobile apps," IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, 2016, pp. 1-9. Disponível em http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7524464&isnumber=7524326. Acesso em 14/05/2017.
11. J. Xiao and F. S. Zeng, "Design and implementation of embedded Web server," *2012 7th International Conference on Computer Science & Education (ICCSE)*, Melbourne, VIC, 2012, pp. 479-482.  
    doi: 10.1109/ICCSE.2012.6295118  
    *URL:*[*http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6295118&isnumber=6295013*](http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6295118&isnumber=6295013)
12. C. ROWLAND, E. GOODMAN, M CHARLIER, A. LIGHT, A. LUI. “Designnin Connected Products”. O’Reilly Media. 2015.
13. http://espressif.com/sites/default/files/documentation/0a-esp8266ex\_datasheet\_en.pdf

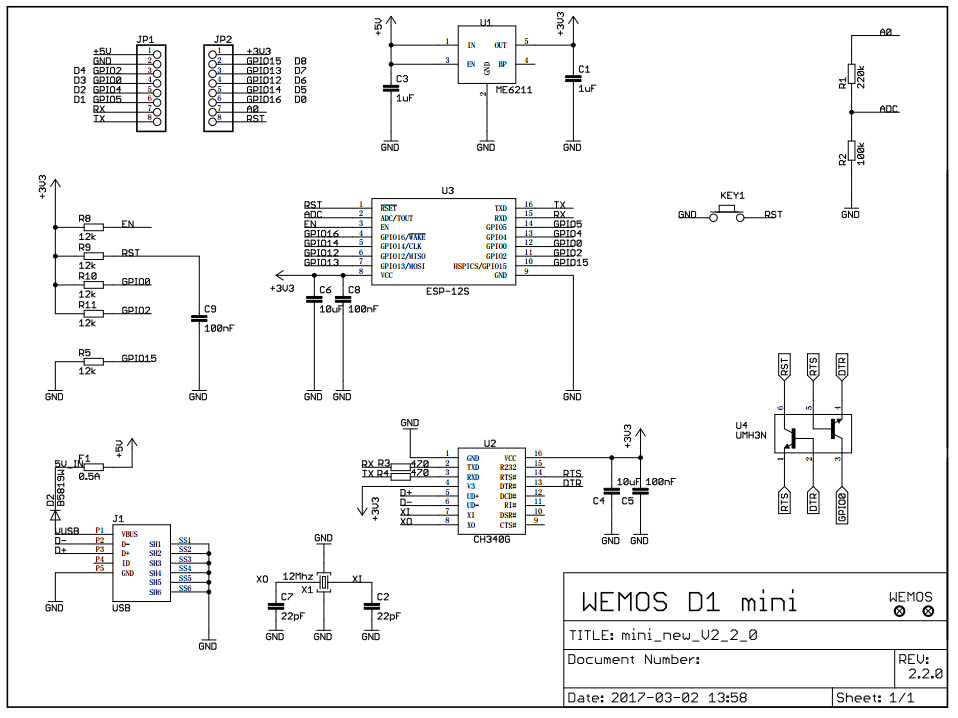
**ANEXO A – Esquemático do circuito *Access Point* da *Texas Instruments*.**



**ANEXO B – Esquemático do circuito *do relógio* da *Texas Instruments*.**



**ANEXO C – Esquemático do circuito *WEMOS D1 mini*.**



1. Preço no dia 21/05/2017 na loja virtual Sparkfun (https://new.sparkfun.com/products/11166) [↑](#footnote-ref-1)
2. Preço no dia 21/05/2017 na loja virtual Adafruit (https://www.adafruit.com/product/1469) [↑](#footnote-ref-2)
3. Preço no dia 21/05/2017 na Loja virtual Sparkfun (https://www.sparkfun.com/products/13678) [↑](#footnote-ref-3)
4. https://thingspeak.com [↑](#footnote-ref-4)
5. http://www.ifttt.com/ [↑](#footnote-ref-5)
6. Htto://www.twillio.com/ [↑](#footnote-ref-6)
7. http://mail.google.com/ [↑](#footnote-ref-7)