# PLANO DE TESTES

**NOME DA EQUIPE:** Koala

**PARTICIPANTES:** Eduardo Izidorio, Kamila Leite, Lucas Anderson, Yhasmim Ferreira

Este modelo pode ser adaptado conforme necessário para atender aos requisitos específicos do projeto.

## 1. Introdução

**Objetivo**: Este documento define o plano de teste e os casos de teste para o Projeto MedBox: Sistema de compartimentos inteligentes, com o intuito de verificar a funcionalidade, desempenho, segurança e confiabilidade dos dispositivos e sistemas implementados.

**Escopo**: Os testes cobrem os principais componentes e funcionalidades do sistema, incluindo a integração de sensores, atuação de dispositivos e a comunicação entre a plataforma ESP32 e o servidor.

## 2. Estratégia de Teste

Metodologia: A metodologia utilizada será baseada em testes manuais e automáticos, com foco em testes funcionais, de integração, de desempenho e de segurança.

Ambiente de Teste:

- Dispositivos: ESP32
- Ferramentas:Arduino IDE

Responsáveis pelo teste: Eduardo Izidorio

## 3. Casos de Teste

**#Caso de Teste 1: Sensor de Distância Ultrassônico (HC-SR04)**

- **ID: CT-001**
- **Descrição:** Validar se o sensor de distância ultrassônico (HC-SR04) mede a distância corretamente e envia os dados para o ESP32.
- **Pré-condição:** Sensor HC-SR04 corretamente conectado ao ESP32 e configuração do código de leitura do sensor.
- **Passos de Teste:**
    1. Ligar o ESP32 e garantir que o sensor HC-SR04 esteja corretamente alimentado.
    2. Testar a medição de distância com o sensor (variando a distância do objeto).
    3. Verificar se a leitura de distância no console serial do :ESP32 corresponde à distância real medida com uma régua ou outro método de medição.
    4. Enviar os dados de distância para o servidor via Wi-Fi (conforme configurado no Caso de Teste 2).
    5. Verificar a recepção e a precisão dos dados de distância no servidor.
- **Resultado Esperado:** O sensor deve medir a distância corretamente, e os dados devem ser enviados sem erros para o servidor.
- **Resultado Real:** Erro nos pinos do código que não estava colocado de maneira correta. Após essa correção, tudo funcionou corretamente.
- **Status:** PASSOU

**###Caso de Teste2: Sensor de Temperatura e Umidade DHT11**

- **ID: CT-002**
- **Descrição:** Validar se o sensor DHT11 mede corretamente a temperatura e a umidade, e envia os dados para o ESP32.
- **Pré-condição:** Sensor DHT11 corretamente conectado ao ESP32 e configuração do código de leitura do sensor.
- **Passos de Teste:**
    1. Ligar o ESP32 e garantir que o sensor DHT11 esteja corretamente alimentado.

2. Testar a medição de temperatura e umidade com o sensor (variando as condições ambientais, como alterar a temperatura ou umidade).
3. Verificar se os valores de temperatura e umidade exibidos no console serial do ESP32 correspondem à medição real com um termômetro e higrômetro de referência.
4. Enviar os dados de temperatura e umidade para o servidor via Wi-Fi (conforme configurado no Caso de Teste 2).
5. Verificar a recepção e a precisão dos dados de temperatura e umidade no servidor.

- **Resultado Esperado:** O sensor deve medir a temperatura e umidade corretamente, e os dados devem ser enviados sem erros para o servidor.
- **Resultado Real:** Atendeu a todos os requisitos
- **Status:** PASSOU

**###Caso de Teste 3: Buzzer e LEDs**

- **ID: CT-003**
- **Descrição:** Validar o funcionamento do buzzer e dos LEDs, garantindo que acionem corretamente conforme comandos do ESP32.
- **Pré-condição:** Buzzer e LEDs corretamente conectados ao ESP32, e código de controle configurado.
- **Passos de Teste:**
    1. Ligar o ESP32 e garantir que o buzzer e os LEDs estejam conectados corretamente.
    2. Testar o acionamento do LED (ligar e desligar) em resposta a comandos do ESP32.
    3. Verificar se os LEDs acendem com a cor e intensidade esperadas quando o comando é dado (por exemplo, LED vermelho acende em caso de erro, LED verde para sucesso).
    4. Testar o acionamento do buzzer, verificando se ele emite som ao ser ativado e desativa quando o comando correspondente é dado.

5. Verificar a resposta do buzzer e dos LEDs ao envio de um sinal do servidor, por exemplo, ativar o buzzer quando um valor de temperatura fora do intervalo permitido for detectado.

- **Resultado Esperado:** O buzzer deve emitir som quando ativado e os LEDs devem acender corretamente de acordo com os comandos fornecidos pelo ESP32.
- **Resultado Real:** Um dos LEDs não acendeu durante o teste, ao verificar foi notado que o jumper não estava funcionando corretamente, foi trocado. Após isso ambos os LEDs funcionaram corretamente, assim como o buzzer.
- **Status:**PASSOU

**###Caso de Teste 4: Conexão com Servidor Web**

- **ID: CT-004**
- **Descrição:** Validar se o ESP32 consegue se conectar a um servidor web e enviar/receber dados de forma confiável.
  Pré-condição: Servidor web configurado e ativo, com o ESP32 conectado à rede Wi-Fi.
- **Passos de Teste:**
  1. Ligar o ESP32 e garantir que ele está conectado à rede Wi-Fi.
  2. Verificar a configuração do servidor web (por exemplo, um servidor MQTT local ou na nuvem) para garantir que ele está preparado para receber e responder a requisições.
  3. Enviar uma requisição MQTT do ESP32 para o servidor, usando um método GET ou POST (dependendo da configuração do servidor).
  4. Verificar se a resposta do servidor é recebida corretamente pelo ESP32 (por exemplo, código MQTT 200 de sucesso ou dados em formato JSON).
  5. Testar o envio de dados do ESP32 para o servidor, por exemplo, enviando dados de sensores (como temperatura e umidade) para o servidor via requisição POST.
  6. Verificar no servidor se os dados recebidos do ESP32 estão corretos e são processados de maneira adequada.

7. Realizar um teste de desconexão e reconexão com o servidor, simulando uma perda de rede.
8. Testar a reconexão automática do ESP32 ao servidor após uma falha de conexão.

- **Resultado Esperado:** O ESP32 deve ser capaz de se conectar ao servidor web, enviar e receber dados com sucesso, e reconectar automaticamente em caso de falha de conexão.
- **Resultado Real:** Atendeu a todos os requisitos
- **Status:** PASSOU

**###Caso de Teste 5: Integração de LEDs, Sensor Ultrassônico e Buzzer**

- **ID: CT-005**
- **Descrição:** Validar a integração do sensor ultrassônico com LEDs e buzzer, de modo que os LEDs e o buzzer acionem quando a lembrete do remédio for enviado do app mobile.
- **Pré-condição:** Sensor ultrassônico (HC-SR04), LEDs e buzzer corretamente conectados ao ESP32, e código configurado para ler dados do sensor e controlar os LEDs e o buzzer.
- **Passos de Teste:**
    - Ligar o ESP32 e garantir que o sensor ultrassônico, LEDs e buzzer estão corretamente conectados e configurados.
    - Testar a medição de distância com o sensor ultrassônico e verificar se o valor de distância é exibido corretamente no console serial do ESP32.
    - Configurar o código para acionar os LEDs e o buzzer com base no envio do lembrete pelo app mobile.
    - Exemplo:
    - Dado alerta de mensagem os leds e buzzer são acionados, e só são desligados a aproximação de 5cm do sensor ultrassônico.
    - Verificar se os LEDs acendem corretamente (nas cores definidas) e o buzzer emite os sons esperados conforme a distância medida.

    - Realizar testes de variação de distância (por exemplo, aproximando e afastando objetos) e verificar se a resposta dos LEDs e do buzzer é imediata e correta.
- **Resultado Esperado:** O ESP32 deve acionar os LEDs e o buzzer conforme o recebimento do lembrete no app mobile.
- **Resultado Real:** Atendeu a todos os requisitos
- **Status:** PASSOU

**###Caso de Teste 6: Conexão do app mobile com o servidor na nuvem**

**ID**: CT-006

**Descrição**: Validar a integração do aplicativo mobile com o servidor MQTT, garantindo que os dados sejam enviados e recebidos corretamente, com acionamento de LEDs e buzzer conforme o envio do lembrete de remédio.

**Pré-condição**: O aplicativo mobile deve estar configurado para se conectar ao servidor MQTT, com as credenciais e tópicos corretamente configurados. O dispositivo deve estar conectado ao servidor MQTT, com a assinatura do tópico correspondente.

**Passos de Teste:**

1. **Ligar o servidor MQTT**:
    - O servidor MQTT deve estar funcionando e acessível.
2. **Configurar e enviar o lembrete do app mobile**:
    - Enviar o lembrete de remédio para o tópico MQTT configurado.
3. **Verificar a recepção da mensagem no dispositivo receptor**:
    - O dispositivo (ESP32) deve receber a mensagem do lembrete corretamente.
4. **Acionar LEDs e buzzer**:
    - LEDs devem acender e o buzzer deve emitir som conforme a mensagem recebida.
5. **Testar variação de distância do sensor ultrassônico**:
    - Aproximar e afastar objetos do sensor e verificar a resposta dos LEDs e buzzer.
6. **Testar reconexão ao servidor MQTT**:

- Simular falha de conexão e verificar se o dispositivo reconecta automaticamente.

7. **Verificar o tempo de resposta**:
   - Medir o tempo entre o envio do lembrete e o acionamento dos LEDs/buzzer.

**Resultado Esperado:**

- O sistema deve enviar, receber dados e acionar LEDs/buzzer corretamente.
- O tempo de resposta deve ser inferior a 3 segundos.
- O dispositivo deve reconectar automaticamente após falha.

**Resultado Real:** O teste atendeu a todos os requisitos. O aplicativo mobile enviou o lembrete com sucesso, o servidor MQTT processou a mensagem corretamente, e o dispositivo (ESP32) acionou os LEDs e buzzer conforme esperado. A reconexão foi bem-sucedida após falha de conexão e o tempo de resposta foi inferior a 3 segundos.
**STATUS:** PASSOU

**###Caso de Teste 7: Integração do app com o banco de dados**
**ID:** CT-007
**Caso de Teste:** Integração do App com o Banco de Dados
**Descrição:** Validar a integração do banco de dados com o servidor MQTT na nuvem, garantindo que o ESP32 consiga consumir os dados do banco de dados para a emissão dos alertas de remédios.
**Pré-condição:** O aplicativo mobile deve estar configurado para se conectar ao servidor MQTT e ao banco de dados. O banco de dados deve estar acessível e corretamente configurado para armazenar e fornecer os lembretes. O dispositivo ESP32 deve estar conectado ao servidor MQTT e assinado no tópico correspondente.
**Passos de Teste:**
**1. Ligar o servidor MQTT:** O servidor MQTT deve estar funcionando e acessível.
**2. Conectar o app mobile ao servidor MQTT:** Verificar se a conexão é estabelecida corretamente.
**3. Cadastrar um lembrete de remédio no app mobile:** O lembrete deve ser salvo no banco de dados e enviado para a nuvem.

**4. Consumir os dados do banco de dados pelo ESP32:** O dispositivo deve acessar o banco de dados e receber os lembretes corretamente.

**5. Verificar a recepção da mensagem no dispositivo receptor:** O ESP32 deve processar a mensagem corretamente.

**6. Acionar LEDs e buzzer:** Os LEDs devem acender e o buzzer emitir som conforme a mensagem recebida.

**7. Testar reconexão ao servidor MQTT:** Simular uma falha de conexão e verificar se o dispositivo reconecta automaticamente.

**8. Verificar o tempo de resposta:** Medir o tempo entre o cadastro do lembrete e o acionamento dos LEDs/buzzer.

**Resultado Esperado:** O usuário cadastra o remédio via app mobile, o lembrete é salvo no banco de dados e enviado à nuvem. O dispositivo receptor (ESP32) consome os dados e aciona os LEDs e buzzer corretamente. O tempo de resposta deve ser inferior a 3 segundos. O dispositivo deve reconectar automaticamente após falha.

**Resultado Real:** O banco de dados não estava registrando os remédios corretamente, causando falhas no cadastro dos lembretes. Houve problemas com dependências no desenvolvimento do app, dificultando a comunicação com o banco de dados. Após corrigir e atualizar as versões das dependências, conseguimos estabelecer a comunicação correta, garantindo o registro e envio dos dados ao servidor.

**STATUS:** PASSOU

### 4. Critérios de Aprovação

- Funcionalidade: Todos os casos de teste funcionais devem ser aprovados.
- Desempenho: O tempo de resposta dos dispositivos não deve exceder o limite especificado.
- Segurança: Nenhuma vulnerabilidade crítica deve ser encontrada.
- Resiliência: O sistema deve retomar a comunicação após falhas de rede sem perda de dados.

### 5. Conclusão

- **Resumo dos Resultados**

Durante a execução dos testes do Projeto MedBox, foram identificadas algumas falhas que foram corrigidas ao longo do processo. No teste do Sensor Ultrassônico (CT-001), houve um erro nos pinos do código, mas após a correção, o sensor funcionou corretamente. O Sensor de Temperatura e Umidade (CT-002) atendeu a todos os requisitos sem problemas. No teste do Buzzer e LEDs (CT-003), um LED não acendeu devido a um jumper defeituoso, que foi substituído para resolver o problema.

A Conexão com o Servidor Web (CT-004) ocorreu conforme esperado, garantindo a comunicação do ESP32 com o servidor. Já no teste de Integração do Sensor Ultrassônico, LEDs e Buzzer (CT-005), o sistema respondeu corretamente ao envio do lembrete, acionando os alertas adequadamente.

Na Conexão do App Mobile com o Servidor MQTT (CT-006), todos os requisitos foram atendidos, garantindo que os dados fossem enviados, recebidos e processados corretamente, com tempo de resposta inferior a 3 segundos e reconexão automática após falhas.

Por fim, na Integração do App com o Banco de Dados (CT-007), houve falhas iniciais no registro de lembretes devido a problemas com dependências do aplicativo. Após a atualização das versões e ajustes na comunicação com o banco de dados, os lembretes passaram a ser cadastrados corretamente, garantindo que o ESP32 consumisse os dados e acionasse os alertas conforme esperado.

No geral, todos os testes foram bem-sucedidos após as correções necessárias, garantindo a funcionalidade e confiabilidade do sistema.