Plano de Testes – Projeto SafeVax V2

1. Introdução

Este documento descreve o plano de testes para o projeto SafeVax V2, visando validar que o sistema atende aos requisitos funcionais, não funcionais e de integração. O objetivo é garantir que todas as funcionalidades – desde a coleta e transmissão dos dados dos freezers até a visualização e notificação – operem de forma segura, eficiente e integrada.

2. Objetivos dos Testes

- **Verificação Funcional:** Garantir que cada requisito funcional seja implementado corretamente e opere conforme esperado.

- **Validação da Integração:** Assegurar que os diferentes módulos do sistema (dispositivos IoT, broker MQTT, servidor Flask, banco de dados, dashboard Streamlit) se comuniquem e interajam corretamente.

- **Avaliação da Segurança e Desempenho:** Testar a eficácia dos algoritmos de compactação (Huffman) e criptografia, bem como a latência na transmissão dos dados.

- **Testes de Usabilidade:** Verificar a interface e a experiência do usuário no dashboard interativo em Streamlit e com as notificações.

- **Testes Manuais e Unitários:** Enquanto os módulos do ESP32 serão validados via testes manuais in loco, os testes unitários serão aplicados no Streamlit.

3. Escopo dos Testes

3.1 - Componentes a Serem Testados

- **Dispositivos IoT e Comunicação via MQTT:**

    - Sensores de temperatura (DHT11).

    - Sensores de estado da porta do container, utilizando sensor de luz infravermelho PIR, instalado na parte superior do container e voltado para a parte móvel da porta.

    - Módulo RFID para identificação de acessos.

    - Envio dos dados (compactados e criptografados) via MQTT.

- **Servidor Flask e Integração com Banco de Dados:**
    - Captação dos dados via MQTT.
    - Processamento: descompactação (Huffman), descriptografia e registro dos dados em banco (SQLite ou PostgreSQL, conforme a versão adotada).
    - Exposição dos dados via API HTTP.
- **Interface de Visualização e Notificações:**
    - Dashboard interativo desenvolvido com Streamlit.
    - Notificações no WhatsApp para recebimento dos dados, emissão de alertas.

Componentes Excluídos dos Testes

- Testes unitários e de integração não serão aplicados aos módulos do ESP32, que serão validados via testes manuais in loco.

4. Estratégia de Testes

4.1 - Tipos de Testes

- **Testes Funcionais (Manuais):**
    - Funcionamento dos sensores (temperatura e sensor PIR para estado da porta).
    - Leitura e registro dos acessos via RFID.
    - Envio e recepção dos dados via MQTT.
    - Processamento no servidor Flask (descompactação, descriptografia e registro dos dados).
- **Testes de Integração (Manuais):**
    - Dispositivos IoT → Broker MQTT → Servidor Flask → API HTTP.
    - Integração entre o servidor e o dashboard (Streamlit).
- **Testes Unitários (Automatizados):**

    - Módulos do dashboard em Streamlit:
        - Validação dos métodos de descriptografia e descompactação.
        - Verificação da lógica de exibição e manipulação de dados.
        - Testes de notificação e alerta no WhatsApp.

- **Testes de Segurança:**
    - Robustez da criptografia aplicada e integridade dos dados durante a transmissão.

- **Testes de Performance:**
    - Medir a latência na transmissão dos dados e a eficácia da compactação, garantindo comunicação em tempo real com baixa latência.

4.2 - Ferramentas de Teste

- **Ambiente de Teste Manual:**
    - Equipamentos com ESP32, sensores (DHT11 e sensor PIR), módulo RFID e broker MQTT configurado.

- **Ambiente de Teste Automatizado:**
    - Frameworks de testes para Streamlit (Python unittest/pytest).

- **Ferramentas de Monitoramento:**
    - Logs e analisadores de pacotes (ex.: Wireshark) para validar a criptografia e a integridade dos dados.

5. Casos de Teste

5.1 - Testes Funcionais

- **TF-01: Monitoramento de Temperatura**
    - **Objetivo:** Verificar se o sensor DHT11 coleta e envia a temperatura em tempo real.
    - **Procedimento:**

1. Simular variações de temperatura no ambiente do freezer.

2. Verificar a exibição dos dados enviados via MQTT.

- **Resultado Esperado:** Dados precisos e atualizados em tempo real.

- **TF-02: Registro de Estado da Porta do Container**

  - **Objetivo:** Confirmar que o sensor de luz infravermelho PIR registra corretamente o estado (aberto/fechado) da porta do container.

  - **Procedimento:**

    1. Simular a abertura e fechamento da porta do container.

    2. Observar se o sensor PIR detecta a alteração de estado imediatamente e se essa informação é enviada ao sistema.

  - **Resultado Esperado:** Cada mudança de estado é registrada e transmitida instantaneamente.

- **TF-03: Emissão de Alerta por Ausência de RFID**

  - **Objetivo:** Validar que, ao abrir a porta do container, se nenhum cartão RFID for identificado em até 10 segundos, o sistema emite um alerta.

  - **Procedimento:**

    1. Abrir a porta do container.

    2. Não apresentar nenhum cartão RFID ao leitor durante 10 segundos.

    3. Verificar se o sistema emite o alerta conforme especificado.

  - **Resultado Esperado:** Alerta emitido imediatamente após o período de 10 segundos sem identificação de RFID.

- **TF-04: Identificação via RFID**

  - **Objetivo:** Validar o registro de acesso dos usuários por meio do RFID.

  - **Procedimento:**

1. Aproximar um cartão RFID do leitor.

2. Verificar se o acesso é registrado com os dados corretos.

- **Resultado Esperado:** A identificação do usuário é capturada e enviada corretamente.

- **TF-05: Envio de Dados via MQTT**

  - **Objetivo:** Confirmar que os dados são compactados, criptografados e enviados ao broker MQTT.

  - **Procedimento:**

    1. Verificar os dados enviados a partir dos freezers utilizando um analisador de pacotes.

  - **Resultado Esperado:** Dados enviados no formato esperado, compactados e criptografados.

5.2 Testes de Integração

- **TI-01: Integração do Servidor Flask**

  - **Objetivo:** Verificar a recepção dos dados pelo servidor Flask e seu processamento (descompactação e descriptografia).

  - **Procedimento:**

    1. Enviar dados via MQTT a partir dos dispositivos.

    2. Confirmar que o servidor Flask processa e armazena os dados no banco.

  - **Resultado Esperado:** Dados corretamente registrados e disponíveis via API HTTP.

- **TI-02: Integração do Dashboard (Streamlit)**

  - **Objetivo:** Validar a exibição dos dados no dashboard interativo.

  - **Procedimento:**

    1. Conectar o dashboard à API do servidor Flask.

2. Verificar a atualização em tempo real dos gráficos e tabelas.

- **Resultado Esperado:** Dados apresentados de forma clara e interativa.

- **TI-03: Integração das notificações**
  - **Objetivo:** Testar o consumo dos dados (criptografados e compactados) e o envio de alertas.
  - **Procedimento:**
    1. Simular condições críticas (ex.: temperatura elevada ou porta aberta sem RFID).
    2. Verificar a recepção dos dados no WhatsApp e a emissão dos alertas.
  - **Resultado Esperado:** Notificações enviadas corretamente aos gestores.

5.3 Testes de Segurança

- **TS-01: Validação da Criptografia**
  - **Objetivo:** Confirmar que os dados são criptografados durante a transmissão.
  - **Procedimento:**
    1. Utilizar ferramentas de análise de pacotes para interceptar a comunicação.
    2. Verificar se os dados estão protegidos.
  - **Resultado Esperado:** Nenhum dado em texto claro é visível.

- **TS-02: Teste de Integridade dos Dados**
  - **Objetivo:** Verificar se os dados mantêm sua integridade após a compactação e descompactação.
  - **Procedimento:**
    1. Comparar os dados antes e depois do processo.
  - **Resultado Esperado:** Os dados permanecem íntegros e sem perdas.

5.4 Testes de Performance

- **TP-01: Latência na Transmissão**
    - **Objetivo:** Medir o tempo entre a coleta dos dados e sua disponibilidade via API.
    - **Procedimento:**
        1. Realizar medições em diferentes condições de rede.
    - **Resultado Esperado:** Latência dentro dos parâmetros estabelecidos para atualizações em tempo real.

5.5 Testes Unitários (Criptografia e Streamlit)

- **TU-01: Testes de Descriptografia e Descompactação**
    - **Objetivo:** Validar as funções de descriptografia e descompactação dos dados.
    - **Procedimento:**
        1. Executar testes automatizados utilizando o framework de testes.
    - **Resultado Esperado:** Funções retornam os dados corretamente processados.
- **TU-02: Testes dos Componentes do Dashboard (Streamlit)**
    - **Objetivo:** Verificar que os componentes do dashboard (gráficos, tabelas, etc.) processam e exibem os dados conforme esperado.
    - **Procedimento:**
        1. Utilizar pytest ou outro framework para testar os componentes.
    - **Resultado Esperado:** Componentes apresentam os dados de forma correta e responsiva.

6. Cronograma e Ambiente de Teste

- **Ambiente de Teste Manual:**

    - Equipamentos com ESP32, sensores (DHT11 e PIR), módulo RFID e freezers em ambiente controlado.

    - Broker MQTT configurado para simulação do fluxo completo.

- **Ambiente de Teste Automatizado:**

    - Configuração de testes unitários para o Streamlit em ambiente de desenvolvimento.

- **Cronograma:**

    - Testes Funcionais e de Integração: Durante a implementação e após a integração de cada módulo.

    - Testes Unitários: Após a implementação dos módulos críticos no Streamlit.

    - Testes de Performance e Segurança: Durante a fase de integração e antes da implantação final.

7. Critérios de Aceitação

- **Funcionalidade:** Todos os requisitos funcionais e de integração (incluindo os novos testes para o sensor PIR e a verificação do alerta por ausência de RFID) devem ser atendidos conforme os resultados dos testes.

- **Segurança:** A comunicação deve estar criptografada e os dados devem manter sua integridade durante os processos de compactação/descompactação.

- **Performance:** A latência do sistema deve permitir atualizações em tempo real.

- **Usabilidade:** As interfaces das notificações e do dashboard devem ser intuitivas e responsivas, garantindo uma boa experiência ao usuário.

8. Conclusão

Este plano de testes tem o intuito de assegurar que o projeto SafeVax V2 opere de forma integrada, segura e eficiente, cumprindo todos os requisitos definidos – incluindo a nova abordagem para o monitoramento do estado da porta do container via sensor PIR e a emissão de alertas na ausência de identificação RFID dentro de 10 segundos. As etapas de testes manuais, unitários e de integração garantirão a qualidade e robustez do sistema antes da implantação em ambiente real. Este documento será atualizado conforme novas funcionalidades e requisitos forem incorporados ao projeto.