MODELO DE PLANO DE TESTES

NOME DA EQUIPE: Koala
PARTICIPANTES: Eduardo Izidorio, Lucas Anderson, Yhasmim Ferreira, Kamila Leite

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 Life Vault: Sistema de Controle de Acessos e Monitoramento de Salas Refrigeradas , com o intuito de verificar a funcionalidade, desempenho, segurança e confiabilidade dos dispositivos e sistemas implementados.

Escopo: O projeto Life Vault: Sistema de Controle de Acessos e Monitoramento de Salas Refrigeradas visa desenvolver um sistema que utiliza tecnologia RFID para permitir o acesso a locais restritos de forma ágil, dinâmica e segura. O sistema autentica usuários através de tags RFID e registra acessos. Sensores de temperatura e umidade (DHT11) garantirão a monitorização ambiental, enquanto atuadores como LED RGB e servo motor serão usados para sinalização visual e controle de travas. O acesso será gerenciado via um aplicativo mobile que se comunicará com o sistema por meio do protocolo MQTT.

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, NFC PN532, DHT11, LED RGB
- Ferramentas: VSCODE, AIRDUINO IDE

Responsáveis pelo teste: Lucas Anderson

3. Casos de Teste

 # Caso de Teste 1: Leitura da tag e cartão RFID

● ID: CT-001
● Descrição: Verificar se o sensor está lendo corretamente a tag e o cartão RFID

● Pré-condição: O sensor de leitura está conectado ao ESP32, configurado corretamente, a biblioteca Adafruit PN532 está instalada.
● Passos de Teste:
1. Ligar o dispositivo ESP32.
2. Coletar leituras do RFID 3. Verificar se o id de cada tag está sendo lido corretamente.
● Resultado Esperado: As leituras do cartão e da tag RFID devem ser lidos corretamente, além de ser possível identificar o id de cada tag.
● Resultado Real: O cartão e a tag foram identificados corretamente, mas no primeiro momento o led vermelho não acendeu, pois o jum estava quebrado. Depois, foi realizada a troca e funcionou corretamente.
● Status: PASSOU

#Caso de Teste 2: Funcionamento Sensor DHT11

●ID: CT-002
●Descrição: Verificar se o sensor DHT11 está funcionando corretamente. Para o teste, a temperatura e a umidade devem ser lidas e exibidas no monitor serial.
●Pré-condição: O sensor DHT11 deve estar corretamente conectado ao ESP32, com as portas de dados configuradas corretamente.
●Passos de Teste:

1. Conectar o sensor DHT11 ao ESP32.
2. Ligar o ESP32.
3. Enviar o código para o ESP32.
4. Monitorar a leitura dos dados de temperatura e umidade no monitor serial.

●Resultado Esperado: A leitura de temperatura e umidade será exibida corretamente no monitor serial, com os valores correspondentes às condições ambientais.
● Resultado Real: O sensor DHT11 estava conectado na porta de alimentação errada, por isso não funcionou, foi observado esse erro e, logo foi corrigido.
●Status: PASSOU

#Caso de Teste 3: Funcionamento Servo Motor

● ID: CT-003
● Descrição: Verificar se o servo motor está funcionando corretamente. Para o teste, o motor deve mover-se para uma posição específica, e a movimentação será observada e registrada no monitor serial.
● Pré-condição: O servo motor deve estar corretamente conectado ao ESP32, com a porta de controle configurada corretamente no código.
● Passos de Teste:

1. Conectar o servo motor ao ESP32.

2. Ligar o ESP32.
3. Enviar o código para o ESP32.
4. Monitorar o movimento do servo motor e verificar se ele está se movendo para a posição desejada.
5. Observar os dados de posição e movimentação no monitor serial (se aplicável).

● Resultado Esperado: O servo motor moverá-se corretamente para a posição especificada no código, e o status de movimentação será mostrado corretamente no monitor serial.

● Resultado Real: O servo motor moveu-se corretamente para a posição definida no código e os dados foram registrados corretamente no monitor serial.

● Status: PASSOU

#Caso de Teste 4: Integração Leitura RFID, Sensor DHT11 e Servo Motor

● ID: CT-004

● Descrição: Verificar se o sistema funciona corretamente quando integrado, lendo uma tag ou cartão RFID, coletando dados de temperatura e umidade do sensor DHT11 e, com base nessas leituras, acionando o servo motor. A movimentação do servo motor deve ser realizada quando a tag RFID for lida e as condições do sensor forem atendidas.

● Pré-condição:

- O módulo RFID deve estar corretamente conectado ao ESP32, com a porta de dados configurada corretamente.
- O sensor DHT11 deve estar corretamente conectado ao ESP32 e funcionando.
- O servo motor deve estar corretamente conectado ao ESP32 e funcionando.
- O código de controle deve estar configurado para ler a tag RFID, coletar os dados do sensor DHT11 e mover o servo motor conforme as condições específicas.

● Passos de Teste:

1. Conectar o módulo RFID ao ESP32.
2. Conectar o sensor DHT11 ao ESP32.
3. Conectar o servo motor ao ESP32.
4. Ligar o ESP32.
5. Enviar o código de integração para o ESP32, configurando a leitura da tag RFID, coleta de dados do sensor DHT11 e controle do servo motor.
6. Apresentar uma tag ou cartão RFID para o leitor.
7. Verificar se a tag RFID é lida corretamente.
8. Verificar se a temperatura e umidade são lidas corretamente pelo sensor DHT11.
9. Verificar se o servo motor é acionado de acordo com a leitura do sensor DHT11 (ex: movimento do servo motor após leitura correta da tag e condições ambientais específicas).
10. Monitorar o comportamento do sistema e os dados exibidos no monitor serial.

● Resultado Esperado:
  - O sistema deve ler corretamente a tag ou o cartão RFID.
  - O sensor DHT11 deve retornar a temperatura e umidade corretamente.
  - Com base na leitura do sensor DHT11, o servo motor deve se mover para a posição desejada.
  - A movimentação do servo motor deve ocorrer apenas quando a tag RFID for lida com sucesso e as condições ambientais do sensor DHT11 forem atendidas (por exemplo, temperatura acima de um valor específico).
  - Todos os dados e status de operação devem ser exibidos corretamente no monitor serial.

● Resultado Real: O sistema integrou com sucesso a leitura da tag RFID, a coleta de dados do sensor DHT11 e o controle do servo motor. A leitura da tag foi bem-sucedida, o sensor DHT11 retornou valores precisos e o servo motor foi acionado conforme esperado.
● Status: PASSOU

#Caso de Teste 5: Conexão com a Nuvem
● ID: CT-005
● Descrição: Verificar se o sistema consegue estabelecer uma conexão bem-sucedida com a nuvem e enviar dados de forma correta. Para o teste, os dados do sensor (como temperatura e umidade) serão enviados para a nuvem e monitorados em tempo real.
● Pré-condição:
  - O ESP32 deve estar conectado à rede Wi-Fi corretamente.
  - O sistema de nuvem (por exemplo, AWS, Google Cloud, ou outra plataforma) deve estar configurado para receber os dados do dispositivo.
  - O código de integração entre o ESP32 e a nuvem deve estar corretamente configurado e funcional.

● Passos de Teste:
  1. Conectar o ESP32 à rede Wi-Fi.
  2. Enviar o código para o ESP32 com a configuração para a conexão com a nuvem.
  3. Ligar o ESP32 e aguardar a inicialização do sistema.
  4. Monitorar a conexão com a nuvem no monitor serial (verificar se a conexão foi estabelecida corretamente).
  5. Verificar se os dados coletados pelo sensor DHT11 (temperatura e umidade) estão sendo enviados para a nuvem.
  6. Acessar a plataforma de nuvem configurada para verificar se os dados estão sendo recebidos e exibidos corretamente.
  7. Validar se os dados são atualizados em tempo real e se não há falhas na comunicação.

● Resultado Esperado:

- O ESP32 deve se conectar corretamente à rede Wi-Fi.
- O sistema deve estabelecer uma conexão bem-sucedida com a nuvem.
- Os dados de temperatura e umidade do sensor DHT11 devem ser enviados para a nuvem corretamente.
- A plataforma de nuvem deve receber e exibir os dados em tempo real, com atualização constante.
- O status da conexão e os dados enviados devem ser registrados corretamente no monitor serial.

● Resultado Real: A conexão com a nuvem foi estabelecida com sucesso, os dados do sensor DHT11 foram enviados corretamente e exibidos na plataforma de nuvem em tempo real. Não houve falhas na comunicação, e os dados foram atualizados conforme esperado.
● Status: PASSOU

#Caso de Teste 6: Teste de conexão do app com o MQTT
● ID: CT-006
● Descrição: Verificar se o app Flutter consegue estabelecer uma conexão bem-sucedida com o broker MQTT para enviar e receber mensagens corretamente.
● Pré-condição:
As dependências do Flutter devem estar corretamente instaladas.
O broker MQTT (por exemplo, Mosquitto ou outro) deve estar configurado e em execução.
O app Flutter deve estar configurado com as credenciais corretas para o broker MQTT.
● Passos de Teste:

1. Iniciar o app Flutter com as dependências MQTT configuradas.
2. Verificar se o app se conecta corretamente ao broker MQTT com as credenciais configuradas.
3. Verificar se a conexão é estabelecida sem erros no console do app.
4. Enviar uma mensagem de teste para o broker MQTT e monitorar a resposta.
5. Verificar se o broker MQTT recebe e processa a mensagem corretamente.
6. Assinar um tópico no app e verificar se as mensagens enviadas para esse tópico são recebidas corretamente.
7. Testar a desconexão e reconexão do app com o broker MQTT, verificando se a comunicação se mantém estável.

● Resultado Real:
● Status: PASSOU

#Caso de Teste 7: Recebimento de notificação

● ID: CT-007
● Descrição: Verificar se o sistema consegue enviar e o app consegue receber notificações corretamente, com base em um evento ou condição predefinida.
● Pré-condição:
  - O app Flutter deve estar corretamente instalado no dispositivo.
  - O sistema de envio de notificações (como Firebase Cloud Messaging, OneSignal, etc.) deve estar configurado corretamente.
  - O dispositivo do usuário deve ter permissões para receber notificações habilitadas.
  - O dispositivo deve estar conectado à internet (via Wi-Fi ou dados móveis).
● Passos de Teste:
  1. Garantir que o app Flutter esteja em segundo plano ou fechado.
  2. Realizar uma ação no sistema que deve disparar uma notificação (exemplo: alteração no estado do sistema, sensor acionado, etc.).
  3. Enviar a notificação do backend para o app via o serviço de notificações configurado (como Firebase, por exemplo).
  4. Verificar se o dispositivo recebe a notificação corretamente, seja na tela de bloqueio ou como uma notificação ativa na área de notificações.
  5. Verificar se a notificação contém a informação esperada (como título, corpo da mensagem e/ou dados adicionais).
  6. Abrir o app e verificar se ele responde corretamente ao clique na notificação (se houver alguma ação ou navegação associada à notificação).
  7. Verificar o comportamento do app após o recebimento da notificação (por exemplo, se a notificação altera algum estado dentro do app ou se há alguma atualização no conteúdo).
● Resultado Esperado:
  - O dispositivo deve receber a notificação corretamente, mesmo com o app em segundo plano ou fechado.
  - A notificação deve ser exibida com o conteúdo correto (título, mensagem, etc.).
  - Ao interagir com a notificação, o app deve responder corretamente, realizando a ação associada à notificação (por exemplo, abrir uma tela específica ou atualizar o conteúdo).
  - O status da notificação (como sucesso ou falha no envio) deve ser registrado corretamente no log do sistema.
● Resultado Real:
● Status: PASSOU

#Caso de Teste 8: Criação do banco de dados

● ID: CT-008
● Descrição: Verificar se o banco de dados foi criado corretamente e se a estrutura de tabelas, relações e dados iniciais estão conforme o esperado.
● Pré-condição:
  - O sistema de banco de dados (por exemplo, MySQL, PostgreSQL, MongoDB, etc.) deve estar instalado e em funcionamento.
  - O script de criação do banco de dados e das tabelas deve estar disponível e correto.
  - O acesso ao banco de dados deve estar configurado corretamente (credenciais, permissões de acesso, etc.).
● Passos de Teste:
  1. Executar o script de criação do banco de dados no sistema de gerenciamento de banco de dados (DBMS).
  2. Verificar se o banco de dados foi criado corretamente sem erros.
  3. Validar a criação das tabelas conforme especificado no script (nomes, tipos de dados, e relações entre as tabelas).
  4. Verificar se as tabelas criadas contêm as colunas e chaves primárias/estrangeiras corretamente definidas.
  5. Inserir dados iniciais de teste nas tabelas (se necessário).
  6. Consultar as tabelas para verificar se os dados iniciais estão presentes e corretos.
  7. Testar se as operações básicas de CRUD (Criar, Ler, Atualizar, Excluir) funcionam conforme esperado nas tabelas do banco de dados.
  8. Verificar se o banco de dados está recebendo e armazenando dados conforme as operações realizadas.
● Resultado Esperado:
  - O banco de dados deve ser criado corretamente sem erros.
  - As tabelas e suas relações devem estar de acordo com o especificado no script de criação.
  - Os dados iniciais devem ser inseridos corretamente nas tabelas, caso necessário.
  - As operações de CRUD devem funcionar conforme esperado, permitindo a inserção, leitura, atualização e exclusão de dados.
  - O banco de dados deve estar operando corretamente e realizando as funções de armazenamento conforme o necessário.