Disciplina: MIC014 – Hands-On BasicDesenvolvimento Orientado a Testes

Atividade: Maker Aula 2

Assunto: Avanço/Melhoria no Projeto de Botão de Pânico

Nome dos participantes: Abrahão Picanço Neres de Oliveira e Luciano dos Santos Nascimento

ARTEFATO 3 - BIG PICTURE

1. Definição do Objetivo: O objetivo central é criar um sistema de emergência simples, onde um botão é usado para enviar um alerta de socorro para um número de telefone específico via WhatsApp, utilizando o ESP32 e a API CallMeBot. Esse sistema pode ser aplicado em várias situações onde a comunicação de emergência seja necessária e o usuário precise de um método de notificação rápida e confiável.

2. Identificação dos Componentes Principais:

- Quem está envolvido no projeto?

Abrahão Picanço Neres de Oliveira e Luciano dos Santos Nascimento

- Quais são as etapas principais do projeto?

Inicialização e Conexão:

Ao ligar o dispositivo, o ESP32 tenta se conectar à rede WiFi usando as credenciais fornecidas. O LED2 começa a piscar para indicar que a conexão está sendo estabelecida. Uma vez conectado, o LED2 fica aceso.

1. Monitoramento do Botão:

   O sistema aguarda o pressionamento do botão. Quando o botão é pressionado, o ESP32 acende o LED1 e começa o processo de envio da mensagem de emergência.

2. Envio da Mensagem:

   O ESP32 faz uma requisição HTTP para a API CallMeBot, enviando o texto configurado ("Socorro!!!!,Me Ajude!!") para o número de telefone especificado.

   Se o envio for bem-sucedido, o ESP32 confirma com o monitor serial e apaga o LED1. Se ocorrer algum erro, o sistema tenta novamente até 3 vezes.

3. Feedback Visual:

   LED1: Indica se a mensagem está sendo enviada (acende) ou se foi enviada com sucesso (apaga).

   LED2: Indica o status da conexão WiFi (pisca enquanto tenta conectar e acende quando a conexão for bem-sucedida).

4. Resiliência e Reconexão:

   Se a conexão WiFi for perdida durante o funcionamento, o ESP32 tenta se reconectar automaticamente. Durante a reconexão, o LED2 pisca, e o sistema retoma a operação quando a conexão for restabelecida.

- **Que ferramentas e recursos serão utilizados?**

1. ESP32 (Placa Principal)

- **Conexão com a Internet**: O ESP32 conecta-se à rede WiFi para permitir o envio de mensagens via internet.
- **Controle de Botão**: O ESP32 monitora um botão (no pino 21), que, quando pressionado, aciona o envio da mensagem.
- **Controle de LEDs**: Dois LEDs são usados para fornecer feedback visual:
  - **LED2 (Pino 2)**: Indica o status de conexão WiFi (pisca enquanto tenta conectar, acende quando conectado).
  - **LED1 (Pino 23)**: Acende enquanto a mensagem está sendo enviada.
- **Envio de Mensagens**: Ao pressionar o botão, o ESP32 faz uma requisição HTTP para a API CallMeBot, enviando uma mensagem de emergência via WhatsApp.

2. CallMeBot API

- **Serviço de Mensagem**: A API é usada para enviar a mensagem de emergência para o número de telefone configurado no código, usando a plataforma WhatsApp.
- **Autenticação e Envio**: A autenticação é feita via uma chave de API única, garantindo que apenas requisições válidas possam ser processadas.

3. WiFi

- **Conexão de Rede**: A estabilidade e a conexão WiFi são essenciais para garantir que a mensagem seja enviada sem falhas. O sistema deve ser capaz de se reconectar automaticamente caso a conexão seja perdida.

- **Quais são os obstáculos a serem superados e as oportunidades?**

Conexão de Internet:

- O sistema depende da estabilidade da conexão WiFi. A perda de conexão pode impedir o envio da mensagem. É importante implementar um mecanismo de reconexão eficiente, como foi feito no código.

Consumo de Energia:

- O ESP32 pode consumir mais energia ao usar o WiFi. Se for necessário operar por longos períodos em locais sem acesso constante à energia, considerar o uso de fontes de energia alternativas (como baterias com gerenciamento de energia) seria importante.

Segurança da Comunicação:

- A API CallMeBot usa um token de autenticação (API Key) que precisa ser mantido seguro. Caso a chave seja comprometida, é possível que mensagens sejam enviadas indevidamente.

Debounce e Interrupções:

- A melhoria do debounce no botão foi uma boa prática para garantir que o botão não dispare múltiplos eventos de envio acidentais. Isso pode ser crucial para garantir a confiabilidade do sistema em situações de emergência.

- Quais são os resultados esperados?

Este projeto visa fornecer uma solução simples e acessível para enviar alertas de emergência via WhatsApp usando um ESP32, um botão e a API CallMeBot. A chave para o sucesso do projeto é garantir que o sistema seja confiável, que a comunicação WiFi seja estável e que o envio de mensagens seja rápido e eficiente. Além disso, o uso de LEDs como indicadores de status e a adição de reconexão automática garantem uma experiência robusta para o usuário final.

O sistema é versátil, podendo ser utilizado em uma variedade de contextos, como segurança pessoal, assistência a idosos, ambientes industriais e sistemas de monitoramento remoto. As melhorias potenciais envolvem a adição de funcionalidades como alertas sonoros, monitoramento remoto e envio para múltiplos destinatários.

Possíveis Melhorias Futuras

1. **Adição de um Alerta Sonoro**:

   Integrar um alarme sonoro (como um buzzer) para fornecer feedback adicional ao usuário, indicando que o alerta foi enviado ou que houve um erro.

2. **Monitoramento Remoto**:

   Incorporar funcionalidades de monitoramento remoto via uma interface web ou aplicativo para que os familiares ou responsáveis possam ver o status do dispositivo e confirmar se o alerta foi enviado corretamente.

3. **Múltiplos Destinatários de Mensagens**:

   O sistema pode ser modificado para enviar mensagens de emergência para múltiplos números, garantindo que diferentes contatos sejam notificados em caso de emergência.

4. **Integração com Sistemas de Emergência**:

   Pode-se integrar o sistema com outros serviços de emergência, como enviar um alerta para um número de emergência local ou para um centro de monitoramento de saúde.

3.Apresenta: