1 Monitoramento de vagas

1.1 Especificação Estrutural

Nessa seção do documento iremos especificar a estrutura do projeto *Park Monitoring* identificando os componentes a serem utilizados, sua função e sua forma de interação com os demais componentes.

- Microcontrolador ESP32: O Microcontrolador é responsável pela orquestração dos componentes do projeto.
  - Leitura dos sensores infravermelho.
  - Implementação da lógica do programa, fazendo com que as medições sejam interpretadas e ações executadas utilizando as diversas interfaces
  - Envio dos registros de eventos via interface Wi-fi integrada ao microcontrolador via protocolo http.
- LED RGB: Permite que com níveis de tensão especificados conseguimos utilizar diferentes cores, ele é formado por um vermelho (R de red), um verde (G de green) e um azul (B de blue - não será utilizado).
- Sensores Infravermelho: O sensor tem como principal objetivo identificar a alteração de estado do objeto monitorado.
  - O sensor escolhido *SHARP GP2Y0A02YK0F* tem como principal característica a medição de distância ao invés de presença. Isso permite flexibilidade na implementação em diferentes ambientes, visto que, podemos calibrar a distância em qualquer ambiente e alterar a lógica adicionando um valor.
  - Características:
    - Alimentação: 5v
    - Corrente: 33mA
    - Saída analógica
- Um decodificador para selecionar o endereço de cada sensor/led que o microcontrolador precisará se comunicar
- Display LCD para simular um painel que indica a quantidade de vagas disponível naquele estacionamento. O display tem o controle de vagas totais vs vagas disponíveis no momento. Esse display é atualizado a cada alteração no estado de um led.
  - Características
    - Comunicação serial via I2C
    - Display 16x2
- Para utilizar o Display LCD precisamos de um módulo I2C IC PCF8574 que fornece expansão de E/S remota de uso geral por meio do relógio serial bidirecional I2C de dois fios (SCL) e dados seriais (SDA).

1.2 Algoritmo de observação de vagas

- De maneira geral o microcontrolador irá fazer uma varredura em cada um dos dos sensores, em um período definido, verificando se houve uma alteração de estado;
- Ao identificar uma alteração de estado o MC irá guardar o tempo em que essa alteração foi detectada e irá manter uma flag para o endereço daquele sensor indicando uma mudança de estados;
- Caso não haja alteração no estado o valor de t_n será atualizado para garantir que o valor do led desse endereço não seja alterada na etapa de mudança de estado;
- Em um segundo momento o MC irá realizar uma segunda varredura nos endereços dos sensores verificando se o tempo passado desde a alteração de estado ultrapassa o valor definido de t_{cs} (tempo de threshold para mudança de estado de uma vaga)
- Caso o tempo desde a alteração de estado detectada seja maior que t_{cs} o MC irá alterar o valor do led e resetar a flag que indica mudança de estado
- Caso o tempo t_n seja menor que t_{cs}, ou seja, ou não se passou tempo o suficiente desde a última alteração de estado ou o valor medido no sensor seja igual o valor representado no led, não será necessário ocorrer nenhuma mudança

Este comportamento descrito está representado no algoritmo a seguir:

```
while true:
    Fazendo uma varredura por todos os enderecos:
        Se (valor_sensor != valor_led): % ou seja sensor esta medindo um estado diferente do
    que ta no led
            se flag_n==0:
                t_n = timestamp.now()
                flag_n = 1
        Se valor_sensor == valor_led: % note que caso seja identificada uma mudanca de estado
    que ja foi identificada anteriormente o valor de t_n nao sera alterado
            t_n = timestamp.now()
            flag_n = 0 % garante que flag de mudanca de estado esteja em zero

    % Para fazer a alteracao de estado da lampada
    Fazendo uma varredura por todos os enderecos:
        se (timestamp.now() - t_n)>= tempo threshold:
            estado_led = estado_sensor
            flag = 0 % reseta flag
    % se (timestamp.now() - t_n) < tempo threshold o estado da led e mantido
```

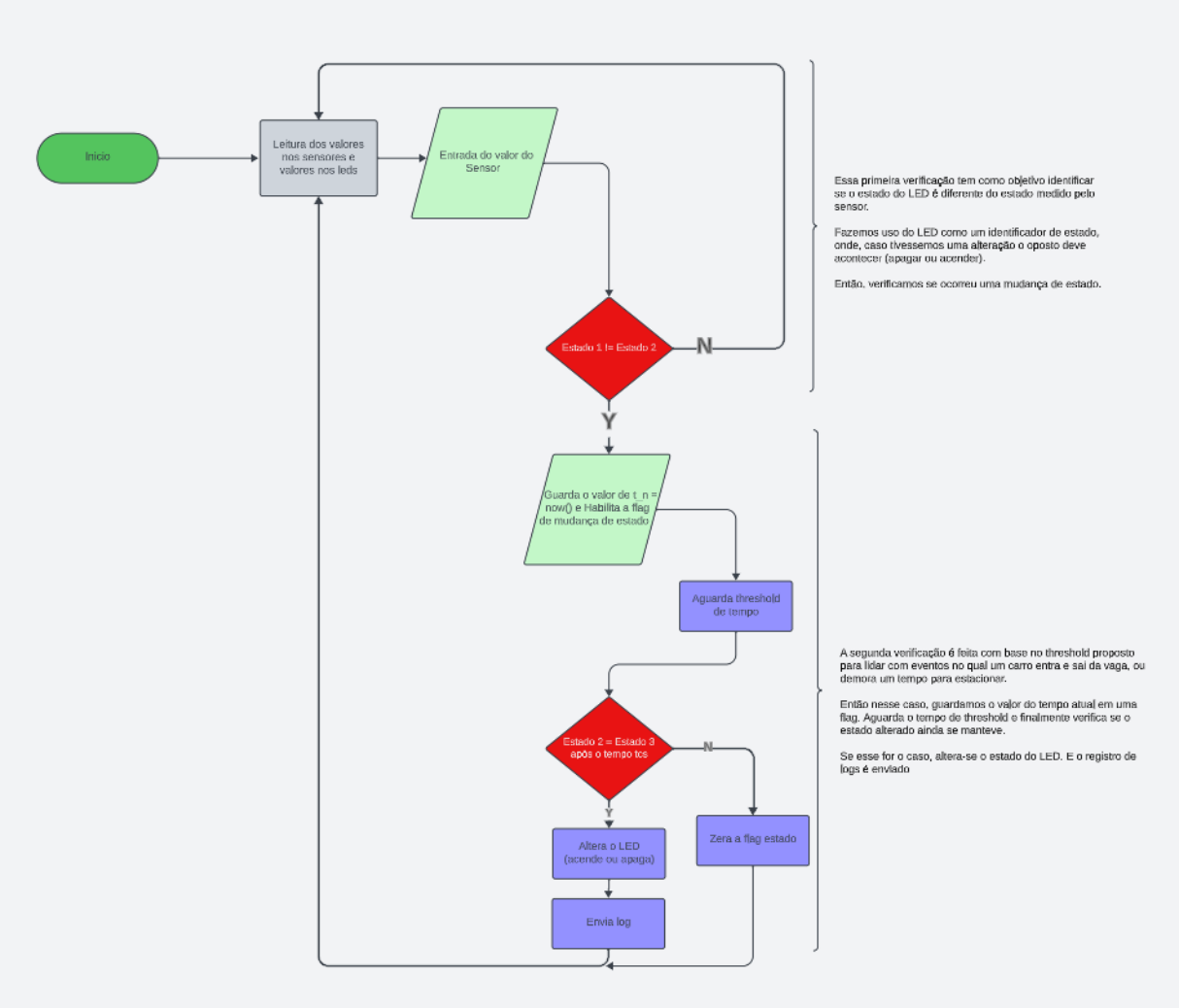

Figura 1: Fluxograma do algoritmo de observação de vagas

2 Envio de dados para análise

Para a realização da análise de fluxo de veículos do sistema, iremos utilizar um software externo, o Splunk Cloud Platform, este software oferece suporte justamente para a análise de dados e geração de relatórios e dashboard para análise de dados de maneira prática. Dessa forma, considerando que o microcontrolador ESP32 escolhido, possui um módulo wifi integrado, a partir de um tempo de coleta de dados t_{dc} definido, o microcontrolador verificaria a relação de vagas naquele instante e bastaria fazer o envio desses dados para o servidor

utilizado do Splunk via protocolo http onde esses dados seriam posteriormente analisados via Splunk Processing Language. De maneira geral temos que o algoritmo utilizado para essa comunicação apresentado na seguir

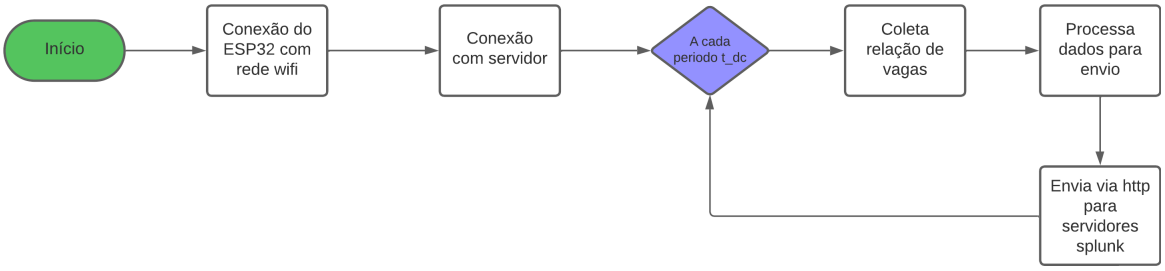

Figura 2: Diagrama de funcionamento do algoritmo de coleta e envio de dados.