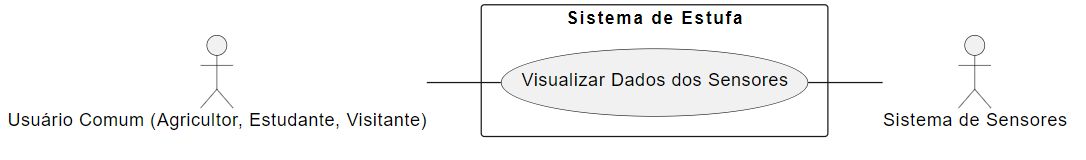
**Casos de Uso**

**Atores**

1. **Usuário Comum (Agricultor, Estudante, Visitante)**
2. **Sistema de Sensores**

**Casos de Uso Principais**

1. **Visualizar Dados dos Sensores**
   * **Descrição**: Permite ao usuário visualizar os dados coletados pelos sensores em tempo real no dashboard.
   * **Ator Principal**: Usuário Comum
   * **Fluxo de Eventos**:
     1. Os sensores coletam dados do ambiente da estufa.
     2. Os dados são transmitidos para o sistema.
     3. O sistema atualiza o dashboard com os dados recebidos.
     4. O usuário visualiza os dados em tempo real no dashboard.
     5. Se os dados estiverem fora dos parâmetros normais, o sistema envia um alerta visual no dashboard.

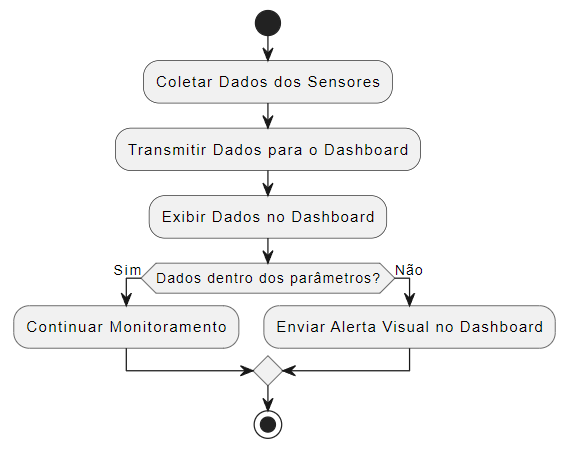


**Diagrama de Atividades**

**Atividade Principal: Visualizar Dados dos Sensores**

**Passos**

1. **Início**
2. **Coletar Dados dos Sensores**
3. **Transmitir Dados para o Dashboard**
4. **Exibir Dados no Dashboard**
   * [Decisão]: Dados dentro dos parâmetros?
     + [Sim]: Continuar Monitoramento
     + [Não]: Enviar Alerta Visual no Dashboard
5. **Fim**



**Modelo Banco de dados**

**Coleção: sensors**

A coleção 'sensors' armazena as leituras dos sensores em tempo real. Cada documento nesta coleção representa uma leitura de sensor e inclui as seguintes informações:

- \_id: Identificador único do documento  
- sensor\_id: Identificador único do sensor  
- type: Tipo de sensor (ex: temperatura, umidade)  
- value: Valor da leitura do sensor  
- timestamp: Data e hora da leitura

**Exemplo de Documento**

{  
 "\_id": "609d1c2f8f1b2c1f8f1b2c1f",  
 "sensor\_id": "temp\_sensor\_01",  
 "type": "temperature",  
 "value": 22.5,  
 "timestamp": "2024-05-16T10:20:30Z"  
}

**Plano de Entregas com Laravel e Vue.js**

**Sprint 1: Configuração Inicial e Implementação do Backend**

**Objetivo**: Configurar o ambiente de desenvolvimento e implementar a API do backend para simulação de dados.

* **História 1**: Configurar o ambiente de desenvolvimento
  + Instalar Laravel via Composer
  + Configurar conexão com banco de dados no arquivo **.env**
  + Configurar ambiente de desenvolvimento local
* **História 2**: Implementar o modelo e a migração para os sensores
  + Criar modelo **Sensor**
  + Criar migração para a tabela **sensors**
  + Executar migração para criar a tabela no banco de dados
* **História 3**: Implementar a API para simulação de dados
  + Criar controlador **SensorController** com métodos **index** e **store**
  + Definir rotas para a API
  + Implementar seeder para simular dados dos sensores

**Sprint 2: Implementação do Frontend**

**Objetivo**: Desenvolver o frontend do dashboard para visualizar os dados simulados dos sensores.

* **História 4**: Configurar o projeto Vue.js
  + Instalar Vue CLI e criar novo projeto
  + Configurar estrutura básica do projeto
* **História 5**: Desenvolver componentes do dashboard
  + Criar componente principal do dashboard
  + Implementar tabela para exibir dados dos sensores
  + Estilizar componentes para melhor visualização
* **História 6**: Integrar frontend com a API do backend
  + Instalar Axios para requisições HTTP
  + Implementar chamada à API para obter dados dos sensores
  + Exibir dados no componente do dashboard

**Sprint 3: Implementação de Alertas e Monitoramento em Tempo Real**

**Objetivo**: Implementar a funcionalidade de alertas e aprimorar o monitoramento em tempo real dos dados dos sensores simulados.

* **História 7**: Implementar sistema de alertas
  + Definir parâmetros para detecção de anomalias
  + Implementar lógica de detecção no backend
  + Integrar alertas visuais no frontend
* **História 8**: Melhorar o monitoramento em tempo real
  + Implementar WebSockets para comunicação em tempo real
  + Ajustar coleta de dados para menor latência
  + Testar e validar monitoramento em tempo real

**Sprint 4: Documentação e Refinamentos Finais**

**Objetivo**: Completar a documentação do projeto e realizar refinamentos finais com base no feedback recebido.

* **História 9**: Completar a documentação do projeto
  + Criar documentação técnica
  + Elaborar guias de usuário
  + Revisar e finalizar documentação
* **História 10**: Refinar funcionalidades existentes
  + Coletar feedback dos usuários
  + Priorizar e implementar melhorias
  + Realizar testes finais de validação