# Relatório Final do Projeto: Sistema Sustentável Orientado a Objetos

Equipe:  
- Andrey Antzuk Camargo  
- Guilherme Olimpio  
- Lucca Blumer  
- Luis Vinicius Uliana

Link do GitHub: [**https://github.com/users/Andrey1234-bot/projects/1**](https://github.com/users/Andrey1234-bot/projects/1)**\**

## Introdução

Este relatório apresenta o desenvolvimento do projeto Sistema Sustentável Orientado a Objetos, que visa atender aos requisitos técnicos e contribuir para o Objetivo de Desenvolvimento Sustentável (ODS) escolhido. O documento detalha as etapas do projeto, incluindo justificativa, casos de uso, modelagem, implementação, e uma reflexão final sobre as lições aprendidas.

## Tema e Justificativa

O projeto busca contribuir para o ODS 3: Saúde e Bem-Estar, através do desenvolvimento de um sistema que monitora indicadores de saúde e facilita a comunicação entre usuários e equipes de saúde. O sistema utiliza sensores para coleta de dados e oferece funcionalidades como notificações e recompensas.

## Etapas do Projeto

### Etapa 1: Modelagem Inicial

Artefatos desenvolvidos:

1. Diagrama de Casos de Uso: Identificou os atores principais e descreveu três casos de uso:

* - Registro de Moradores  
  - Monitoramento de Indicadores de Saúde  
  - Geração de Relatórios

2. Requisitos funcionais e não-funcionais foram definidos para nortear o desenvolvimento.

### Etapa 2: Diagrama de Classes de Domínio e Refinamento dos Casos de Uso

O diagrama de classes detalha a estrutura do sistema, incluindo atributos, métodos e relacionamentos entre as classes principais.  
  
 Diagrama de Classes:

* Crie um diagrama UML de classes com as classes principais:
  + **Usuario** (atributos: idUsuario, nome, email, idade, historicoSaude, pontuacao; métodos: cadastrar(), atualizarDados(), visualizarHistorico())
  + **Sensor** (atributos: idSensor, tipoSensor, localizacao, status; métodos: monitorarIndicador(), enviarAlerta())
  + **DadoSaude** (atributos: idDado, tipo, valor, dataHora; métodos: registrar(), atualizar())
  + **EquipeSaude** (atributos: idEquipe, nomeEquipe, email; métodos: receberNotificacao(), emitirRecomendacao())
  + **Recompensa** (atributos: idRecompensa, descricao, pontosNecessarios; métodos: atribuirRecompensa())
  + **RelatorioSaude** (atributos: idRelatorio, dadosConsolidados, dataGeracao; métodos: gerarRelatorio(), visualizar())
  + **Aplicativo** (atributos: versao; métodos: enviarNotificacao(), registrarUsuario())

Diagrama de Classes  
  
  
Diagrama

Descrição gerada automaticamente  
  
Casos de Uso:  
Caso de Uso: Cadastro de Moradores   
 **Ator:** Morador. **Resumo**:O morador se cadastra no sistema para receber informações sobre saúde e bem-estar. **Pré-condição:** O morador tem acesso ao aplicativo e deseja participar do sistema.  **Pós-condição:** O morador estará registrado no sistema e poderá receber notificações e utilizar outras funcionalidades.  
  
  
  
  
  
  
  
Caso de Uso: Monitoramento de Indicadores de Saúde   
  
**Ator:** Sensor, Equipe de Saúde.   
  
**Resumo:** O sistema monitora indicadores de saúde e notifica os moradores e a equipe de saúde.   
  
**Pré-condição:** Os sensores estão instalados e funcionando.   
  
**Pós-condição:** Notificações são enviadas para a equipe de saúde e moradores em caso de anomalias.   
  
**Fluxo Principal**:   
  
O sensor detecta uma anomalia nos indicadores de saúde.  
O sensor envia a informação para o sistema central.  
O sistema notifica automaticamente a equipe de saúde e os moradores da área.  
A equipe de saúde recebe instruções de ação.   
  
  
  
Fluxo Alternativo:   
Se o sensor falhar, um funcionário pode inserir manualmente a informação no sistema e as notificações serão enviadas.

Caso de Uso: Histórico de Monitoramento  
  
  
**Ator**: Morador.   
  
**Resumo:** O morador pode visualizar seu histórico de saúde e acompanhar sua pontuação de recompensas.   
  
**Pré-condição:** O morador precisa estar cadastrado no sistema e ter dados monitorados.   
  
**Pós-condição:** O morador poderá ver o histórico de monitoramento e a pontuação acumulada.   
  
**Fluxo Principal:**   
  
O morador acessa o aplicativo e faz login no seu perfil.   
  
O morador seleciona a opção "Histórico de Monitoramento".   
  
O sistema exibe todos os dados monitorados e as pontuações acumuladas.   
  
 **Fluxo Alternativo:**   
  
Se não houver dados monitorados, o sistema exibe uma mensagem informando que não há histórico disponível.  
  
  
  
 Caso de Uso: Recompensa   
  
   
  
**Ator:** Morador, Sistema.   
  
Resumo: O morador acumula pontos ou recompensas ao seguir práticas de saúde recomendadas.   
  
  
**Pré-condição:** O morador precisa estar cadastrado no sistema e ter seguido práticas recomendadas.   
  
  
**Pós-condição:** O morador acumula pontos ou recebe uma recompensa baseada em suas atividades.   
  
  
**Fluxo Principal:**   
  
O morador segue uma prática de saúde recomendada.   
  
O sensor valida a prática e registra a ação. O sistema atualiza a pontuação do morador.   
  
O morador pode acessar o aplicativo e ver seus pontos ou recompensas.   
  
  
**Fluxo Alternativo:**  
  
Caso a prática não seja registrada corretamente, o morador não recebe pontos e o sistema envia uma notificação informando o problema.

# Especificação dos Casos de Uso: CADASTRO DE MORADORES: Diagrama Descrição gerada automaticamente MONITORAMENTO DE INDICADORES DE SAÚDE: Diagrama Descrição gerada automaticamente HISTÓRICO DE MONITORAMENTO: Diagrama RECOMPENSA: Diagrama Descrição gerada automaticamente Etapa 3: Modelos Dinâmicos – Diagramas de Interação

Os diagramas de sequência e de comunicação foram criados para modelar o comportamento dinâmico do sistema, detalhando as interações entre objetos nos principais casos de uso.

**Sequencia Morador** Uma imagem contendo Diagrama

Descrição gerada automaticamente **Diagrama de Sequências Monitoramento de Indicadores de Saúde**Diagrama, Esquemático

Descrição gerada automaticamente  
  
   
  
  
  
  
  
  
  
  
 **Diagrama de Sequência Histórico de Monitoramento** Diagrama

Descrição gerada automaticamente

**Diagrama de Sequência Sistema de Recompensas**Diagrama

Descrição gerada automaticamente **DIAGRAMA DE COMUNICAÇÃO: (COLOCAR IMAGEM)**Etapa 4: Diagrama de Atividades e Diagrama de Estados

O diagrama de atividades descreve o fluxo de trabalho do sistema, enquanto o diagrama de estados representa o ciclo de vida dos principais objetos, incluindo estados como Cadastrado, Monitorando, e Inativo.

**Diagrama de Atividades**  
  
  
 Diagrama

Descrição gerada automaticamente

**Diagrama de Estados**  
  
(Imagem aqui)  
  
  
  
  
  
Etapa 5: Implementação do Protótipo

**Protótipo Funcional**O protótipo functional foi implementado utilizando Python. Ele simula funcionalidades como cadastro de moradores, monitoramento de indicadores de saúde, e envio de notificações.

Principais funcionalidades implementadas:

* - Cadastro de Moradores  
  - Monitoramento de Indicadores de Saúde  
  - Notificações de Saúde  
  - Consulta ao Histórico

Instruções de execução e análise do código estão disponíveis no GitHub.  
  
  
**Documentação Técnica  
  
  
Objetivo do Sistema:**

O sistema visa monitorar indicadores de saúde de moradores, como frequência cardíaca e temperatura. O protótipo foi desenvolvido em Python, simulando a coleta de dados de sensores e o envio de notificações em caso de anomalias.

# Principais Classes do Sistema

1. **Morador**:
   * **Atributos**: nome, idade, email, histórico, pontuação.
   * **Métodos**: registrar\_indicador(), exibir\_historico(), adicionar\_pontuacao().
2. **Sensor**:
   * **Atributos**: tipo (frequência cardíaca, temperatura, qualidade do ar).
   * **Métodos**: coletar\_dado().
3. **SistemaCentral**:
   * **Atributos**: moradores.
   * **Métodos**: cadastrar\_morador(), encontrar\_morador(), notificar\_anomalia().
4. **Notificacao**:
   * **Métodos**: gerar\_alerta().

# Estrutura do Código

O código é modularizado em classes que interagem entre si. Cada classe tem responsabilidades específicas, como registrar dados, monitorar indicadores e enviar notificações.

Exemplo de código:

python

Copiar código

class Morador:

def \_\_init\_\_(self, nome, idade, email):

self.nome = nome

self.idade = idade

self.email = email

self.historico = []

self.pontuacao = 0

def registrar\_indicador(self, indicador, valor):

# Registra dados de saúde

pass

# Como Executar o Sistema

**1. Passos para execução**:

* + Instale o Python 3.x.
  + Baixe o arquivo sistema\_sustentavel.py.
  + Execute via terminal: python sistema\_sustentavel.py.

1. **Testando o Sistema**:
   * **Cadastro de Moradores**: Adicione moradores com nome, idade e email.
   * **Monitoramento de Indicadores**: Simule dados de saúde (temperatura, frequência cardíaca).
   * **Notificação de Anomalias**: Caso o valor monitorado seja anômalo, o sistema envia uma notificação.

# Exemplo de Saída 1. Cadastro de Morador: Texto Descrição gerada automaticamente com confiança média 2. Monitoramento de Indicadores: Texto Descrição gerada automaticamente com confiança média 3. Notificação de Anomalia:

## Reflexão Final

Durante o desenvolvimento deste projeto, foram enfrentados desafios como a integração entre diferentes componentes e a adaptação às mudanças nos requisitos. A experiência proporcionou aprendizados significativos sobre modelagem, implementação e a importância da comunicação em equipe.

Limitações: O sistema ainda é um protótipo e carece de funcionalidades mais avançadas, como integração em tempo real com sensores físicos.

## Conclusão

O desenvolvimento deste protótipo, "Sistema Sustentável Orientado a Objetos", foi um exercício valioso para entender como tecnologias orientadas a objetos podem ser aplicadas na construção de soluções para problemas reais, alinhadas aos Objetivos de Desenvolvimento Sustentável (ODS). O sistema criado busca melhorar a qualidade de vida de moradores por meio do monitoramento contínuo de indicadores de saúde, com um foco principal na detecção precoce de anomalias e na comunicação eficiente com equipes de saúde.

Ao longo do projeto, a modelagem orientada a objetos foi essencial para organizar e estruturar o sistema, permitindo uma fácil manutenção e expansão. A criação das classes Morador, Sensor, SistemaCentral, e Notificacao demonstrou como as responsabilidades podem ser distribuídas de forma clara e funcional entre os componentes do sistema.

O protótipo foi implementado em Python, utilizando uma interface simples para permitir a interação do usuário com o sistema, mesmo que este seja um protótipo inicial. A simulação da coleta de dados de sensores, a geração de relatórios e a atribuição de recompensas para os moradores foram funcionalidades chave que demonstram o potencial do sistema.

Além disso, a realização de testes mostrou que o sistema cumpre sua função de monitorar indicadores de saúde, registrar dados e gerar notificações automáticas quando os valores monitorados ultrapassam os limites pré-estabelecidos. No entanto, como todo protótipo, o sistema apresenta limitações, como a ausência de integração com sensores reais e a necessidade de uma interface gráfica mais amigável para o usuário final.

Apesar dessas limitações, o sistema tem um grande potencial para evoluir. A próxima fase de desenvolvimento pode incluir a integração com dispositivos reais de monitoramento de saúde, melhorias na análise de dados para gerar recomendações personalizadas, e a implementação de funcionalidades de aprendizado de máquina para prever condições de saúde com base nos dados coletados.

Este projeto proporcionou uma ótima oportunidade de aplicar conceitos teóricos de programação e modelagem em um cenário realista, além de oferecer uma compreensão mais profunda sobre como a tecnologia pode ser usada para apoiar a saúde pública e o bem-estar, um dos pilares dos Objetivos de Desenvolvimento Sustentável.