Sistema de Monitoramento de Alagamentos - Porto Alegre

Documento Técnico Simplificado - Versão Acadêmica

1. VISÃO GERAL DO SISTEMA

Propósito Este é um sistema acadêmico que simula o monitoramento em tempo real de alagamentos na cidade de Porto Alegre. Ele integra dados simulados de sensores meteorológicos com a análise de imagens por inteligência artificial para identificar pessoas em situação de risco.

Funcionalidades Principais As funcionalidades principais do sistema incluem:

- Monitoramento de 16 regiões administrativas de Porto Alegre
- Simulação independente de dados de nível de água e volume de chuva
- Detecção automática de pessoas em risco usando uma Rede Neural Convolucional (CNN)
- Sistema de alertas em 3 níveis (Normal, Alerta, Crítico) visualizados com cores diferentes no mapa
- Envio de notificações automáticas via e-mail e WhatsApp
- Interface web com mapa interativo que exibe os dados em tempo real

2. ARQUITETURA TÉCNICA

Stack Tecnológico O sistema é construído com uma arquitetura robusta, utilizando as seguintes tecnologias:

Backend: Desenvolvido em Python com o framework Django, utiliza MariaDB para gerenciamento de banco de dados. Conta com PyTorch para a inteligência artificial na detecção de pessoas, Django REST Framework para a criação de APIs, e WebSockets (Django Channels) para comunicação em tempo real. As APIs de e-mail (SMTP) e WhatsApp Business são usadas para as notificações.

Frontend: Construído com HTML, CSS e JavaScript puros, utilizando Bootstrap para o design da interface do usuário. O mapa interativo é renderizado com Leaflet.js, e a comunicação com o backend é feita através de APIs padrão.

3. MODELO DE DADOS

Estrutura do Banco de Dados O banco de dados armazena informações essenciais para o monitoramento:

Dados de Dicionário (fixos): Incluem informações sobre as 16 regiões administrativas de Porto Alegre (incluindo sua geometria para exibição no mapa), os diferentes níveis de água (Normal, Alerta, Crítico), os volumes de chuva (Baixa, Moderada, Intensa) e os status de alerta (Verde, Amarelo, Vermelho).

Tabelas de Cadastro: Armazenam dados dos usuários que desejam receber notificações (nome, telefone, e-mail e a região de interesse).

Tabelas Transacionais: Registram as medições de sensores em tempo real (nível da água e volume de chuva por região), as análises de risco humano (detecção de pessoas via IA e confiança da IA), e o histórico básico de envios de alertas para os usuários.

4. LÓGICA DE ALERTAS

Critérios de Entrada A lógica de alertas baseia-se em três critérios principais:

- Presença Humana (via CNN): Detecta se há pessoas presentes na imagem
- Nível de Água (1-3): Classificado como 1=Normal, 2=Alerta ou 3=Crítico/Inundação
- Volume de Chuva (1-3): Classificado como 1=Baixa, 2=Moderada ou 3=Intensa

Níveis de Alerta O sistema define três níveis de alerta, indicados por cores no mapa:

NÍVEL 0 - NORMAL (Verde/Incolor):

- Condições: Nível de água e volume de chuva são baixos (nível 1). A detecção de pessoas não é ativada.
- Ação: Nenhuma ação necessária.
- Significado: Situação rotineira e segura.

NÍVEL 1 - ALERTA (Amarelo):

- Condições: Nível de água ou volume de chuva atingem o nível 2 ou superior, E nenhuma pessoa é detectada. A detecção de pessoas via CNN é ativada para monitoramento.
- Ação: Alerta preventivo.
- Significado: Inundação inferida, mas sem pessoas em risco detectadas.

NÍVEL 2 - CRÍTICO (Vermelho):

- Condições: Pessoas são confirmadas na imagem, E o nível de água ou volume de chuva atingem o nível 2 ou superior. A detecção de pessoas via CNN está ativada.
- Ação: Alerta máximo e disparo de notificações.
- Significado: Risco direto à vida confirmado.

Fluxo de Execução O fluxo de execução do sistema segue os seguintes passos:

1. O sistema simula dados de nível de água e volume de chuva a cada 30 segundos
2. Se o nível de água ou volume de chuva atingir o nível 2 ou superior, a detecção de pessoas via CNN é ativada
3. A CNN processa a imagem simulada e informa a presença humana
4. O sistema calcula o nível de alerta (0, 1 ou 2) com base em todos os critérios
5. Se houver mudança no nível de alerta, o sistema atualiza o status da região e registra no banco de dados
6. Em caso de alerta crítico (nível 2), as notificações são disparadas para os usuários cadastrados
7. A interface web recebe atualizações em tempo real via WebSocket e atualiza o mapa e o dashboard

5. APIS E ENDPOINTS

O sistema disponibiliza APIs básicas para permitir a consulta e manipulação de dados. Isso inclui APIs para listar e obter detalhes das regiões (incluindo suas geometrias para o mapa), consultar dados atuais dos sensores, gerenciar o histórico básico de alertas, e cadastrar usuários.

Além das APIs REST, um canal WebSocket é utilizado para enviar atualizações em tempo real para a interface web, garantindo que o mapa e os dados estejam sempre atualizados.

6. SIMULAÇÃO DE DADOS

Algoritmo de Simulação O sistema simula dados de nível de água e volume de chuva para cada região de forma independente. Cada região varia seus níveis aleatoriamente, sem correlações espaciais complexas entre regiões vizinhas.

Periodicidade Os dados são gerados a cada 30 segundos. A simulação é projetada para mudanças graduais e realistas nos níveis de cada região.

7. CNN PARA DETECÇÃO DE PESSOAS

Especificação Técnica A detecção de pessoas é realizada por uma Rede Neural Convolucional (CNN) desenvolvida com PyTorch. Ela recebe como entrada imagens simuladas de câmeras urbanas e classifica a presença humana como "PRESENÇA CONFIRMADA" (se detectar uma ou mais pessoas) ou "AUSENTE".

Ativação Condicional A CNN é ativada condicionalmente, ou seja, só executa a análise de imagem se o nível de água ou volume de chuva na região for igual ou superior ao nível de alerta (nível 2 ou 3). Caso contrário, a presença humana é considerada "AUSENTE" por padrão, sem processamento de imagem.

Métricas de Performance As métricas básicas para a CNN incluem um tempo de processamento inferior a 5 segundos e uma confiança mínima de 70% na detecção.

8. SISTEMA DE NOTIFICAÇÕES

Canais de Notificação O sistema envia notificações automáticas por e-mail e WhatsApp Business API.

- **E-mail:** Configuração básica com template simples
- **WhatsApp (Business API):** Integração oficial via API, envia mensagens de texto simples

Lógica de Disparo As notificações são disparadas quando uma análise de risco humano resulta em um nível crítico (Vermelho), alertando os usuários cadastrados da região afetada.

Templates de Mensagens As mensagens de alerta seguem modelos básicos:

- **Nível 1 (Amarelo):** "ATENÇÃO: Condições de risco detectadas na sua região. Mantenha-se alerta e evite áreas alagáveis."
- **Nível 2 (Vermelho):** "ALERTA CRÍTICO: Risco iminente de alagamento com pessoas em área de perigo. Procure local seguro imediatamente!"

9. INTERFACE WEB E ESTRATÉGIA DE FRONTEND

A interface web é a principal forma de interação com o sistema, apresentando um layout responsivo e dinâmico.

9.1. Fluxo de Dados para o Frontend O fluxo de dados que impacta diretamente o frontend é o seguinte:

1. Os dados de nível de água e chuva são gerados a partir das medições dos sensores por região

2. A análise de risco humano, que utiliza a CNN para detecção em imagens, é ativada somente se o nível de água ou chuva atingir o nível 'Alerta' (nível 2) ou superior
3. O status de alerta de cada região é então calculado combinando esses dados: nível de água, volume de chuva e a presença humana detectada
4. Notificações de alerta são disparadas exclusivamente para o nível 'Crítico' (vermelho)

9.2. Views Básicas Disponíveis para o Frontend O banco de dados possui views simples para facilitar a consulta de dados:

- **vw_situacao_atual:** fornece o status atual de cada região
- **vw_historico_alertas:** oferece o histórico básico de alertas para exibição em tabelas

9.3. Estrutura de Arquivos e Arquitetura Frontend A arquitetura do frontend será totalmente alinhada com o backend Django existente:

- Os templates HTML serão renderizados pelo backend Django
- Arquivos estáticos (CSS, JavaScript) serão organizados e servidos de acordo com a estrutura do Django
- A comunicação com o backend será feita através dos endpoints de APIs Django REST básicos
- A funcionalidade de tempo real será implementada via Django Channels (WebSocket)

9.4. Páginas Simplificadas As páginas da interface web foram simplificadas:

HOME - Dashboard Principal:

- Contadores que mostram o número de regiões em status Normal (Verde), Alerta (Amarelo) e Crítico (Vermelho)
- Um mapa interativo exibindo as 16 regiões de Porto Alegre, com suas geometrias carregadas do banco de dados
- A data e hora da última atualização dos dados

Histórico:

- Tabela básica com: Data do evento, Nome da região afetada, Tipo de alerta, Status de envio básico, Confiança da IA na detecção de pessoas

Cadastro:

- Formulário simples para cadastro de usuários: Nome, Telefone, Email, Região, Situação (ATIVO/INATIVO)

9.5. Integração com APIs Django A comunicação entre o frontend e o backend será feita por meio dos seguintes endpoints básicos da API Django REST:

- Para listar regiões com suas geometrias
- Para obter a situação atual das regiões (baseado na vw_situacao_atual)
- Para consultar o histórico de alertas (baseado na vw_historico_alertas)
- Para cadastrar usuários básicos

O canal WebSocket (alagamentos_updates) será utilizado para enviar atualizações em tempo real com informações essenciais: ID da região, nível e cor do status, timestamp da atualização.

9.6. Componentes JavaScript Específicos Componentes JavaScript básicos serão desenvolvidos:

- **MapController.js:** Gerencia o mapa Leaflet, carrega as geometrias das regiões e atualiza dinamicamente as cores das regiões no mapa
- **WebSocketManager.js:** Gerencia a conexão WebSocket básica e aciona funções para atualizar o mapa e os contadores do dashboard

9.7. Cronograma Simplificado de Desenvolvimento Frontend (3 Semanas)

- **Semana 1:** Setup e Estrutura básica, integração das bibliotecas Bootstrap e Leaflet, implementação das APIs básicas Django REST
- **Semana 2:** Mapa Leaflet com dados reais e integração WebSocket para sistema de cores dinâmico
- **Semana 3:** Histórico básico e Cadastro simples, responsividade e testes básicos

10. CONFIGURAÇÃO E DEPLOY

Variáveis de Ambiente (.env) O sistema é configurável através de variáveis de ambiente básicas para conexão com o banco de dados, credenciais para envio de e-mails e chaves da API do WhatsApp Business.

Configurações do Django As configurações do Django definem os aplicativos básicos instalados no sistema e a configuração dos WebSockets (Channels) para comunicação em tempo real, utilizando uma camada de canal em memória para demonstração.

Dependências Python (requirements.txt) As dependências essenciais do projeto, incluindo Django, PyTorch, Django REST Framework, e bibliotecas para banco de dados e comunicação em tempo real.

11. TESTES E VALIDAÇÃO

Critérios de Funcionamento O sistema será considerado funcionando corretamente quando for capaz de:

1. **Visualização:** Mostrar um mapa de Porto Alegre com as 16 regiões coloridas conforme o status de alerta
2. **Simulação:** Gerar dados simulados de nível de água e chuva automaticamente a cada 30 segundos
3. **IA:** Ativar a CNN quando o nível de água ou chuva atingir o nível 2 ou superior
4. **Alertas:** Mudar a cor da região no mapa conforme o nível de alerta calculado
5. **Notificações:** Enviar e-mail e/ou WhatsApp quando um alerta crítico for disparado
6. **Histórico:** Mostrar um registro básico de todos os alertas no histórico

Casos de Teste Básicos

- **Teste 1 - Transição Normal → Alerta:** Entrada: Nível de água=1, volume de chuva=2. Resultado Esperado: CNN ativada, região exibida na cor amarela, sem notificações
- **Teste 2 - Transição Alerta → Crítico:** Entrada: Nível de água=2, volume de chuva=2, pessoas=CONFIRMADA. Resultado Esperado: Região exibida na cor vermelha, notificações disparadas

12. LIMITAÇÕES E CONSIDERAÇÕES

Limitações Conhecidas

1. **Dados Simulados:** Os dados utilizados não refletem condições reais de Porto Alegre, sendo apenas para fins de simulação
2. **CNN Genérica:** A CNN utilizada precisará de treinamento específico para detecção otimizada em cenários de alagamentos reais
3. **Escopo Acadêmico:** O sistema foi projetado especificamente para ambiente acadêmico e demonstração
4. **Conectividade:** Requer conexão com a internet para o envio de notificações via WhatsApp

Considerações de Performance

- **WebSocket:** Conexões limitadas para demonstração acadêmica
- **CNN:** Processamento sequencial adequado para o escopo do projeto
- **Notificações:** Envio básico sem controles complexos de volume

13. DOCUMENTAÇÃO DE DESENVOLVIMENTO

Padrões de Código O projeto segue padrões básicos de código para garantir consistência:

- Python: PEP 8
- JavaScript: ES6+
- CSS: Organização simples
- SQL: Convenção snake_case para nomes de tabelas e colunas

Ambiente de Desenvolvimento O ambiente de desenvolvimento pode ser ativado e gerenciado com comandos padrão do Django para executar o servidor de desenvolvimento e acessar o shell interativo.

Status do Projeto: Especificações simplificadas - Pronto para desenvolvimento acadêmico imediato.

Última Atualização: 02/06/2025