**VIGIA WEB**

**Documento de Arquitetura do Sistema**

**Jannderson Silva**

**Tayna Santana**

**1. Componentes Principais da Arquitetura**

1. **Front-end**:
   * Interface web para os usuários visualizarem o status de monitoramento, dados históricos e configurar alertas.
   * Deve ser responsivo e de fácil navegação, com gráficos e visualizações claras de tempo de atividade e histórico de anomalias.
   * **Tecnologias**:
     + **Frameworks**: React, Vue.js ou Angular.
     + **Bibliotecas de visualização**: Chart.js ou D3.js, e Grafana para dashboards em tempo real.
2. **Backend (API)**:
   * Serve como uma camada de lógica de negócio, processando requisições de frontend e comunicando-se com o banco de dados e componentes de monitoramento.
   * Expõe uma API RESTful ou GraphQL para o frontend, fornece endpoints para configuração de sites monitorados, exibição de logs e status de verificações.
   * **Tecnologias**:
     + **Frameworks**: Node.js (Express), Python (Django REST Framework ou Flask), ou Ruby on Rails.
     + **Autenticação**: JWT ou OAuth2 para autenticação e controle de acesso.
3. **Monitor de Disponibilidade (Verificador de Serviço)**:
   * Responsável por verificar periodicamente o status dos sites monitorados.
   * Pode ser um microserviço separado, que realiza requisições HTTP aos sites monitorados, registrando o status, o tempo de resposta, e outras métricas relevantes.
   * **Tecnologias**:
     + **Linguagens**: Python (solicitações), Go ou Node.js.
     + **Armazenamento de Logs**: RabbitMQ, Kafka (para streaming de logs) ou diretamente em um banco de dados.
4. **Banco de Dados**:
   * Armazena informações de usuários, sites monitorados, registros de tempo de atividade, e histórico de alertas.
   * Um banco de dados relacional (como PostgreSQL) pode ser usado para armazenar a configuração de sites e usuários, enquanto um banco de dados de séries temporais (como InfluxDB) é ideal para armazenar registros de monitoramento.
   * **Tecnologias**:
     + **Banco de dados relacionais**: PostgreSQL, MySQL.
     + **Banco de dados de séries temporais**: InfluxDB, TimescaleDB.
5. **Detecção de Anomalias**:
   * Esse módulo analisa os dados de resposta e identifica possíveis anomalias, como aumento de latência ou erros HTTP (500, 404, etc.).
   * Você pode utilizar um serviço de aprendizado de máquina básico ou regras estatísticas para definir padrões de resposta aceitáveis.
   * **Tecnologias**:
     + **Linguagens**: Python (com scikit-learn para machine learning) ou bibliotecas específicas para detecção de anomalias.
     + **Serviços em nuvem**: AWS SageMaker, Google AI para modelos de machine learning.
6. **Sistema de Alertas**:
   * Envie alertas para os canais ativados (e-mail, SMS, Slack, etc.) quando uma anomalia for detectada.
   * Você deve permitir a configuração dos tipos de alerta e canais de comunicação preferidos pelo usuário.
   * **Tecnologias**:
     + **Serviços**: Twilio (para SMS), Amazon SNS, Slack API, SendGrid (para e-mails).
     + **Frameworks de integração**: Celery (Python) para gerenciamento de tarefas assíncronas, permitindo escalabilidade e processamento de alertas em tempo real.
7. **Painel de Monitoramento**:
   * Interface visual para usuários acompanharem o ritmo de atividade e desempenho dos sites monitorados.
   * Exibe métricas em tempo real, histórico de resposta e alertas detectados.
   * **Tecnologias**:
     + Grafana (para gráficos e dashboards), ou uma implementação customizada com React/D3.js.

**2. Fluxo de Dados**

1. **Usuário** interage com o **Frontend** para configurar sites monitorados e definir parâmetros de alertas.
2. **Monitor de Disponibilidade** realiza verificações de cada site configurado, enviando os dados ao **Banco de Dados**.
3. Os dados de monitoramento são desenvolvidos pelo módulo de **Detecção de Anomalias**, que identifica e sinaliza eventos fora do padrão.
4. Quando uma anomalia é detectada, o **Sistema de Alertas** envia uma notificação conforme as preferências do usuário.
5. **Dashboard de Monitoramento** exibe todos os dados de monitoramento, permitindo que o usuário visualize e analise as informações em tempo real.

**3. Considerações Adicionais**

* **Escalabilidade**: Utilize serviços em contêineres (Docker e Kubernetes) para gerenciar os serviços de monitoramento e detecção de anomalias.
* **Alta Disponibilidade**: Usar balanceadores de carga e servidores redundantes.
* **Monitoramento e Logging**: Implemente registro de eventos e monitoramento de desempenho com ferramentas como Prometheus e Grafana.