Visão Arquitetônica Inicial: Agendador Inteligente de Atividades

Introdução

Este documento apresenta a visão arquitetônica inicial do sistema "Agendador Inteligente de Atividades", conforme sua concepção atual, antes da aplicação de qualquer metodologia formal de modelagem de ameaças. O objetivo é descrever os componentes principais do sistema, suas interações e os fluxos de dados, fornecendo uma base para a posterior identificação e mitigação de riscos de segurança. O sistema é projetado como uma arquitetura de microserviços, orquestrada por uma API principal e composta por agentes de Inteligência Artificial especializados.

Componentes do Sistema

O sistema é composto pelos seguintes microserviços, cada um encapsulado em um contêiner Docker, conforme definido no arquivo `docker-compose.yml`:

1. API Principal (Orquestrador)

- **Função:** Atua como o ponto de entrada central para o sistema. Recebe requisições do frontend (ou de outros clientes), orquestra a comunicação entre os agentes de IA e retorna as respostas consolidadas. É responsável por rotear as requisições para os agentes apropriados e agregar seus resultados.
- **Tecnologia:** Baseado em Node.js (conforme `src/api/index.js` e `Dockerfile`).
- **Porta Exposta:** 3000.
- **Dependências:** Depende do `agent1-clima` e `agent2-atividades` para operar.

2. Agente 1: Previsor de Horários Climáticos Otimizados (`agent1-clima`)

- **Função:** Responsável por interagir com APIs externas de clima (como OpenWeather) para obter dados meteorológicos. Processa esses dados para prever os horários mais adequados para atividades ao ar livre, considerando fatores como temperatura, umidade e radiação solar.
- **Tecnologia:** Baseado em Node.js (conforme `src/agents/agent1_clima/index.js` e `Dockerfile`).
- **Porta Exposta:** 3001.
- **Dependências:** Requer uma chave de API para o serviço OpenWeather (`OPENWEATHER_API_KEY`).

3. Agente 2: Classificador de Atividades Recomendadas (`agent2-atividades`)

- **Função:** Utiliza um modelo de IA (como Gemini) para classificar e recomendar atividades com base nas condições climáticas fornecidas pelo `agent1-clima` e nas preferências do usuário (implícito na descrição do problema). Pode sugerir atividades alternativas ou adaptar sugestões existentes.
- **Tecnologia:** Baseado em Node.js (conforme `src/agents/agent2_atividades/index.js` e `Dockerfile`).
- **Porta Exposta:** 3002.
- **Dependências:** Requer uma chave de API para o serviço Gemini (`GEMINI_API_KEY`).

4. Frontend

- **Função:** Interface de usuário que permite aos usuários interagir com o sistema. Envia requisições para a API Principal e exibe os resultados (horários otimizados e atividades recomendadas).
- **Tecnologia:** HTML estático (conforme `frontend/index.html`). Presume-se que haverá lógica JavaScript para interagir com a API Principal.

Fluxo de Dados e Interações

O fluxo de dados no sistema pode ser visualizado da seguinte forma:

1. **Usuário para Frontend:** O usuário interage com a interface web (Frontend) para solicitar o agendamento de uma atividade, possivelmente fornecendo detalhes como tipo de atividade, localização e data/hora desejada.
2. **Frontend para API Principal:** O Frontend envia uma requisição HTTP para a API Principal (Orquestrador) na porta 3000. Esta requisição contém os parâmetros da solicitação do usuário.
3. **API Principal para Agente 1 (Clima):** A API Principal, ao receber a requisição, encaminha uma parte dela (e.g., localização, data) para o `agent1-clima` na porta 3001. O `agent1-clima` então consulta a API externa de clima (OpenWeather) usando sua chave de API.
4. **Agente 1 (Clima) para API Externa (OpenWeather):** O `agent1-clima` faz uma requisição para a API OpenWeather, recebendo dados climáticos em tempo real ou previsões.
5. **API Externa (OpenWeather) para Agente 1 (Clima):** A API OpenWeather retorna os dados climáticos para o `agent1-clima`.
6. **Agente 1 (Clima) para API Principal:** O `agent1-clima` processa os dados climáticos e retorna à API Principal os horários otimizados para a atividade.
7. **API Principal para Agente 2 (Atividades):** Simultaneamente ou sequencialmente, a API Principal envia informações (e.g., tipo de atividade, horários otimizados, talvez dados climáticos brutos) para o `agent2-atividades` na porta 3002. O `agent2-atividades` utiliza sua chave de API Gemini para interagir com o modelo de IA.
8. **Agente 2 (Atividades) para API Externa (Gemini):** O `agent2-atividades` faz uma requisição para a API Gemini, enviando os dados para classificação e recomendação.
9. **API Externa (Gemini) para Agente 2 (Atividades):** A API Gemini retorna as recomendações de atividades para o `agent2-atividades`.
10. **Agente 2 (Atividades) para API Principal:** O `agent2-atividades` retorna as atividades classificadas e recomendadas para a API Principal.

11. **API Principal para Frontend:** A API Principal consolida os resultados de ambos os agentes (horários otimizados e atividades recomendadas) e os envia de volta para o Frontend.
12. **Frontend para Usuário:** O Frontend exibe os resultados finais ao usuário.

Limites de Confiança

Com base nesta arquitetura inicial, os seguintes limites de confiança podem ser identificados:

- **Entre o Usuário e o Frontend:** O Frontend é o primeiro ponto de interação e deve ser considerado um limite de confiança. A entrada do usuário deve ser validada e sanitizada.
- **Entre o Frontend e a API Principal:** A comunicação entre o Frontend e a API Principal ocorre via HTTP. Este é um limite de confiança crítico, pois a API Principal expõe a lógica de negócios e orquestração.
- **Entre a API Principal e os Agentes (agent1-clima, agent2-atividades):** A comunicação interna entre a API Principal e os agentes ocorre via HTTP dentro da rede Docker. Embora seja uma rede interna, ainda representa um limite de confiança, pois a integridade e a confidencialidade dos dados podem ser comprometidas se um contêiner for invadido.
- **Entre os Agentes e as APIs Externas (OpenWeather, Gemini):** Os agentes se comunicam com APIs externas. Este é um limite de confiança externo, onde a segurança depende da comunicação segura (HTTPS) e da proteção das chaves de API.

Considerações Iniciais de Segurança

Nesta fase inicial, algumas considerações de segurança preliminares incluem:

- **Proteção de Chaves de API:** As chaves de API (OpenWeather, Gemini) são sensíveis e devem ser protegidas adequadamente, não sendo expostas diretamente no código-fonte ou em repositórios públicos.
- **Comunicação Segura:** A comunicação entre os componentes, especialmente entre o Frontend e a API Principal, e entre os agentes e as APIs externas, deve ser

realizada via HTTPS para garantir a confidencialidade e integridade dos dados.

- **Validação de Entrada:** Todas as entradas de usuário e dados recebidos de outros serviços devem ser rigorosamente validadas para prevenir ataques como injeção de código ou dados maliciosos.
- **Controle de Acesso:** Embora não detalhado, é importante considerar como o acesso aos serviços será controlado, especialmente para a API Principal.

Esta visão inicial servirá como ponto de partida para a aplicação da metodologia STRIDE e a identificação sistemática de ameaças, que serão abordadas na próxima fase de modelagem de ameaças.