**Relatório de Arquitetura de Software**

**1. Introdução e Objetivos do Sistema**

Este relatório apresenta a proposta arquitetural para um sistema de e-commerce especializado em produtos sustentáveis. A arquitetura foi projetada para atender às necessidades de uma startup em rápido crescimento, garantindo escalabilidade, segurança e flexibilidade.

**Objetivos do Sistema**

* **Oferecer uma plataforma confiável** para a venda de produtos ecologicamente corretos.
* **Atender ao crescimento rápido da empresa**, garantindo a capacidade de escalar o sistema.
* **Implementar boas práticas de segurança**, garantindo conformidade com a LGPD e GDPR.
* **Facilitar a entrega contínua de funcionalidades**, com uma arquitetura modular e bem documentada.

**2. Escolha da Arquitetura**

**Arquitetura Proposta**

A arquitetura escolhida é baseada em **Microservices**, que divide o sistema em serviços independentes e especializados.

**Justificativa**

* **Escalabilidade Horizontal**: Cada serviço pode ser escalado independentemente.
* **Modularidade**: Facilita a manutenção e a integração de novos desenvolvedores.
* **Resiliência**: O isolamento de falhas entre os serviços reduz o impacto de problemas em um módulo específico.
* **Compatibilidade com Infraestrutura Cloud**: O uso de containers Docker e orquestração com Kubernetes é ideal para ambientes na nuvem.

**3. Diagramas UML**

Os diagramas UML a seguir ilustram a arquitetura do sistema:

* **Diagrama de Componentes**: Disponível no arquivo Componentes.png.
* **Diagrama de Sequência**: Disponível no arquivo Sequencia.png.
* **Diagrama de Classes**: Disponível no arquivo Classes.png.

**4. Decisões de Design**

**Infraestrutura de Cloud Computing**

* **Containers e Orquestração**:
  + Utilização de Docker para empacotar os serviços em containers.
  + Kubernetes para gerenciamento e escalabilidade horizontal.

**Segurança**

* **TLS/SSL** para proteger a comunicação entre cliente e servidor.
* **Autenticação baseada em JWT**, garantindo acesso seguro às APIs.
* **Conformidade com LGPD/GDPR**:
  + Consentimento explícito para coleta de dados.
  + Implementação de exclusão de dados sob solicitação.

**Escalabilidade**

* **Banco de Dados**:
  + Sharding para dividir os dados em fragmentos menores.
  + Replicação para alta disponibilidade.
* **Cache**:
  + Redis para acelerar consultas frequentes e reduzir a carga no banco.

**Impactos**

* **Facilidade de manutenção**: A modularidade permite que a equipe foque em serviços específicos.
* **Desempenho aprimorado**: Estratégias de cache e escalabilidade garantem resposta rápida mesmo sob alta carga.

**5. Estratégia de Escalabilidade**

A estratégia para garantir o crescimento contínuo do sistema está descrita a seguir:

**Infraestrutura**

* Containers Docker para empacotamento dos serviços.
* Kubernetes para orquestração, balanceamento de carga e escalabilidade horizontal.

**Banco de Dados**

* **Sharding**: Distribuição dos dados entre múltiplos servidores para melhorar o desempenho.
* **Replicação**: Configuração de réplicas para garantir alta disponibilidade.
* **Cache**: Redis como camada de cache para consultas frequentes.

**Serviços**

* **Fila de Mensagens (RabbitMQ)**: Para comunicação assíncrona entre microservices.
* **API Gateway**: Gerencia solicitações externas e distribui entre os serviços.

**6. Pseudocódigo das Classes e Funções**

**Classe Usuario**

* **Atributos**: id, nome, email, senha.
* **Método**: autenticar(email, senha).

**Classe Carrinho**

* **Atributos**: itens.
* **Métodos**:
  + adicionarItem(produto, quantidade)
  + removerItem(produto)
  + calcularTotal().

**Classe Item**

* **Atributos**: produto, quantidade.
* **Método**: calcularSubtotal().

**Classe Pagamento**

* **Atributos**: metodoPagamento, valor.
* **Método**: processarPagamento().

**Nota**

Os pseudocódigos completos das classes estão disponíveis na pasta Pseudocodigo(Portugol).

**7. Conclusão**

A proposta arquitetural desenvolvida atende plenamente ao cenário descrito, proporcionando uma plataforma escalável, segura e modular. A documentação detalhada e os pseudocódigos fornecem um guia completo para implementação, enquanto a infraestrutura baseada em microservices garante a capacidade de atender a um crescimento exponencial da base de usuários.