1 Introdução

O objetivo deste documento é detalhar os requisitos técnicos para a implementação da solução orientada por APIs, que utiliza o PostgreSQL como banco de dados subjacente. A solução visa otimizar o controle de fluxo de caixa e processar relatórios de consolidação diária. O documento abrange todos os aspectos técnicos críticos para garantir uma entrega robusta, segura e evolutiva desse sistema rodando na AWS.

Para assegurar que a solução seja não apenas funcional mas também eficiente em termos de organização e segurança, será adotado o uso de um API Gateway na implementação da arquitetura. A adoção de um API Gateway na arquitetura centraliza o gerenciamento de requisições, fortalecendo a segurança e simplificando a complexidade da rede. Atuando como um ponto de acesso único, ele gerencia o tráfego e roteamento dentro do cluster Kubernetes, permitindo a aplicação de políticas universais e simplificando a administração. Integrando-se com ferramentas de autenticação como o Amazon Cognito, o API Gateway melhora a postura de segurança, aplicando rate limiting e proteção contra ataques, além de abstrair a complexidade de roteamento e distribuição de recursos, assegurando uma fundação sólida para futura expansão e escalabilidade eficiente.

2 Ambiente de Desenvolvimento

- **Linguagens e Frameworks:**
    - **C# .NET:** Utilizado para o desenvolvimento das APIs, proporcionando uma estrutura robusta e eficiente.
    - **Entity Framework Core:** Empregado para interação com o PostgreSQL, facilitando o mapeamento objeto-relacional e acesso a dados de forma eficiente.

- **Ferramentas de Desenvolvimento:**
    - **Visual Code:** IDE principal para desenvolvimento, oferecendo suporte abrangente a C# e .NET.
    - **Git:** Usado para controle de versão, garantindo gestão eficiente e rastreio de mudanças no código.
    - **Docker:** Crucial para a conteinerização de aplicativos, assegurando consistência e portabilidade nos ambientes de teste, desenvolvimento e produção.

- **Automação e Integração Contínua:**
    - **Jenkins ou GitLab CI:** Ferramentas para configuração de pipelines de integração e entrega contínuas, acelerando o ciclo de desenvolvimento e melhoria contínua.

- **Configuração do API Gateway:**
    - **Infraestrutura como Código (IaC):** Administração das configurações do API Gateway através de AWS CloudFormation facilitando a replicação e versionamento das configurações, assegurando consistência e controle.

- **Integração no Ciclo de Desenvolvimento:**
    - **Versionamento e Documentação das APIs:** Implementação de especificações OpenAPI (Swagger) para definir, documentar e negociar contratos dos endpoints geridos pelo API Gateway, promovendo clareza entre os times de desenvolvimento e outras partes interessadas.

- **Monitoramento e Debugging:**
    - **Ferramentas de Monitoramento:** Configuração de métricas com AWS CloudWatch para um monitoramento abrangente do desempenho do API Gateway, assegurando resposta eficiente a alertas de latência ou erros, e facilitando o debugging.

3 Plataforma de Implantação

- **Cloud Provider: Amazon Web Services (AWS)**
  - **Amazon Elastic Kubernetes Service (EKS):** Utilização para executar e gerenciar instâncias Kubernetes, aproveitando ao máximo o escalonamento e balanceamento dinâmico proporcionados pela nuvem.

- **Banco de Dados: Amazon RDS para PostgreSQL**
  - **Alta Disponibilidade:** Configurações otimizadas para garantir alta disponibilidade dos dados e suporte a múltiplas réplicas de leitura em várias zonas de disponibilidade (AZs), melhorando resiliência e performance.

- **Contêineres**
  - **Gerenciamento com Kubernetes:** Utilizada para orquestração de contêineres, facilitando deploys automatizados e escalabilidade flexível do ambiente.

- **API Gateway:**
  - **Amazon API Gateway:** Implementação para gerenciar as requisições de entrada, oferecendo uma interface unificada para exposição de APIs. Ele facilita o encaminhamento de tráfego para serviços adequados dentro do cluster Kubernetes.
  - **Segurança e Controle de Acesso:** Integra usuário com autenticação pelo Amazon Cognito, garantindo que apenas requisições autorizadas passem. Inclui medidas de segurança adicional, como rate limiting e proteção avançada com AWS Shield e AWS WAF.
  - **Observabilidade e Configuração:** Utilização do AWS CloudFormation para gerenciar a configuração do API Gateway como código, assegurando que toda configuração seja replicável e versionada.

4 Requisitos de Desempenho

- **Escalabilidade Horizontal:**
  - **Pods no Kubernetes:** Gerenciamento de cargas com escalabilidade horizontal para lidar eficientemente com picos de tráfego, utilizando múltiplos pods no Kubernetes que podem ser dinamicamente ajustados conforme a demanda.

- **Tempo de Resposta:**
  - **Respostas em <200ms:** As APIs devem ter a capacidade de responder em menos de 200ms sob condições normais para garantir transações rápidas e eficientes.

- **Carga Suportada:**
  - **Capacidade para 50 Requisições por Segundo:** A arquitetura deve suportar até 50 requisições por segundo, garantindo robustez por meio de buffering utilizando AWS ElastiCache para transações complexas quando necessário.

- **API Gateway:**
  - **Gerenciamento de Tráfego de Entrada:** O Amazon API Gateway será utilizado para gerenciar o tráfego de entrada de forma eficiente, distribuindo requisições de maneira otimizada através dos serviços no cluster Kubernetes.
  - **Otimização de Roteamento e Desempenho:** Implementação de cache no nível do API Gateway para otimizar o desempenho e reduzir a carga redundante nos serviços de back-end, contribuindo para tempos de resposta mais rápidos.
  - **Gatilhos Automáticos de Escalonamento:** Configuração para colaborar com o auto-scaling do Kubernetes, reagindo a alterações nas métricas de carga para escalar os recursos dinamicamente e atender aos picos sem comprometer a performance.

5 Segurança

- **Autenticação e Autorização:**
    - **Uso do OAuth 2.0 via Amazon Cognito:** Integração com o API Gateway para gerenciar autenticação de forma centralizada, garantindo que as requisições sejam autenticadas e autorizadas antes de alcançarem os microsserviços. O API Gateway se integra diretamente ao Amazon Cognito para implementar OAuth 2.0, assegurando acesso protegido e gerenciamento robusto de identidade.

- **Gerenciamento de Segredos:**
    - **AWS Secrets Manager:** Utilizado para gerenciamento seguro de credenciais e informações sensíveis. O API Gateway não armazena diretamente segredos, mas facilita a passagem segura de dados entre cliente e backend, assegurando que apenas requisições autorizadas possam acessar o AWS Secrets Manager.

- **Proteção de API:**
    - **Configuração de AWS WAF no API Gateway:** Ajuda a bloquear ameaças comuns listadas na OWASP Top Ten, com regras configuradas para mitigar riscos de segurança em tempo real. O API Gateway atuará como a primeira linha de defesa, se comunicando diretamente com o AWS WAF para aplicar políticas de segurança granularmente.
    - **Rate Limiting e Proteção contra DDoS:** Implementação de controles no API Gateway para limitar requisições e evitar tentativas de abuso ou ataques de negação de serviço, bem como integração com AWS Shield para proteção avançada.

6 Escalabilidade

- **Auto Scaling:**
  - **Configuração de Regras de Escalonamento no EKS e RDS:** Utilização de escalonamento automático para ajustar recursos de forma dinâmica conforme variações de tráfego e consumo. O Amazon API Gateway também acompanhará essas variações, escalando automaticamente sua capacidade para garantir respostas rápidas e eficientes.

- **Balanceamento de Carga:**
  - **Implementação de Elastic Load Balancing (ELB):** Distribuição uniforme do tráfego entre os pods Kubernetes e diferentes zonas de disponibilidade. Complementarmente, o API Gateway atua como intermediário, gerenciando e otimizando o roteamento de requisições de entrada antes de serem repassadas ao ELB, assegurando que o tráfego seja distribuído de forma eficiente e gerenciado de acordo com a demanda de serviços back-end.

- **Escalabilidade Transparente do API Gateway:**
  - **Gerenciamento de Capacidade Dinâmica:** O API Gateway é projetado para escalar de forma automática e transparente, gerindo milhões de requisições com alta disponibilidade, sem necessidade de gerenciamento manual de servidores ou infraestrutura adicional, permitindo que os serviços se adaptem aos picos de demanda de maneira eficiente.

7 Monitoramento e Log

- **Logs:**
  - **Integração com AWS CloudWatch Logs:** Configurado para coletar logs de todas as operações do sistema, inclusive do API Gateway, centralizando

a coleta de dados de desempenho e requisições para análise e auditoria contínuas.

- **Utilização de AWS X-Ray:** Análise detalhada de transações complexas e a rastreamento de requisições de ponta a ponta através do API Gateway até os microsserviços de backend, facilitando a identificação de gargalos e problemas de performance.

- **Monitoramento:**
  - **Configuração de Métricas de Saúde e Performance com AWS CloudWatch:** Estabelecimento de métricas específicas para monitorar o desempenho do API Gateway, incluindo taxas de sucesso, latências, e contagem de erros, além de configurar alertas para condições anômalas, assegurando que a equipe de operações possa responder proativamente a quaisquer incidentes.
  - **Visibilidade Aprimorada com API Gateway Analytics:** Utilização de funcionalidades integradas no API Gateway para analisar o tráfego de API, gerar relatórios sobre uso, identificar padrões de acesso e ajustar as configurações de segurança e capacidade conforme necessário.

8 Plano de Backup e Recuperação

- **Backup Automático:**
  - **Configuração de Snapshots Automáticos Diários no Amazon RDS para PostgreSQL:** Garantindo a proteção contínua de dados com um processo automatizado de snapshots para minimizar o potencial de perda de dados.

- **Retenção:**
  - **Políticas de Retenção de Backups:** Configurações para manter os backups por 30 dias, com a opção de arquivar dados no S3 para armazenamento duradouro e recuperação conforme necessário, ajustado para atender aos requisitos de compliance e segurança.

- **Recuperação:**
    - **Procedimentos Escritos para Operações de Recuperação Rápida:** Implementação de planos detalhados para recuperação, avaliando continuamente o RPO (Recovery Point Objective) e RTO (Recovery Time Objective) para se assegurar que a recuperação das operações seja eficiente e eficaz em caso de falhas significativas.

- **Disponibilidade e Continuidade do API Gateway:**
    - **Estratégia de Alta Disponibilidade:** Uso do API Gateway em múltiplas regiões e zonas de disponibilidade para garantir continuidade do serviço e direcionamento automático de tráfego em caso de falhas regionais.
    - **Failover e Rerouting Automático:** Configuração no API Gateway para redirecionar automaticamente o tráfego para instâncias em operação em casos de falhas, garantindo que os serviços permanecem acessíveis durante processos de recuperação e manutenção.