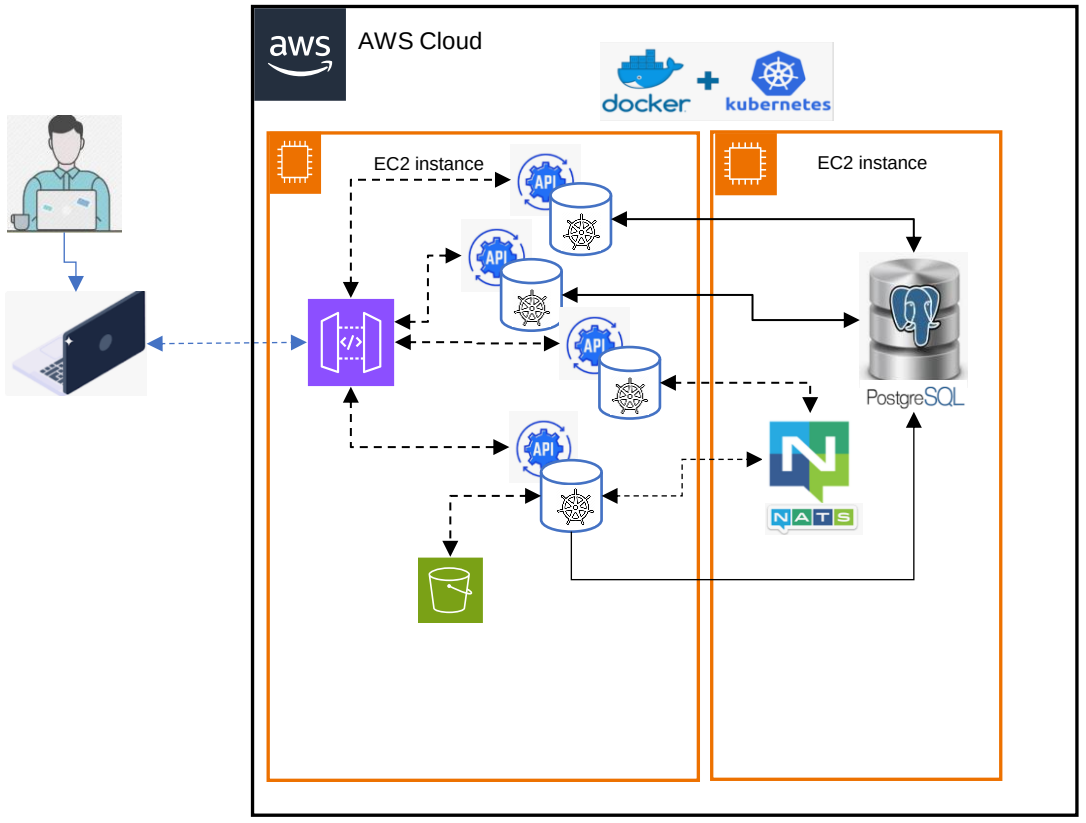
Arquitetura de solução para fluxo de caixa

Janeiro de 2025

Arquitetura para Fluxo de Caixa – Contextualização

Arquitetura Alvo



O desafio

O desafio escolhido está contido no arquivo "***desafio-arquiteto-solução-jan25.pdf***"
Toda documentação gerada nesta solução encontra-se no repositório público do Github ***https://github.com/srj-figueiredo/desafiogft.git***

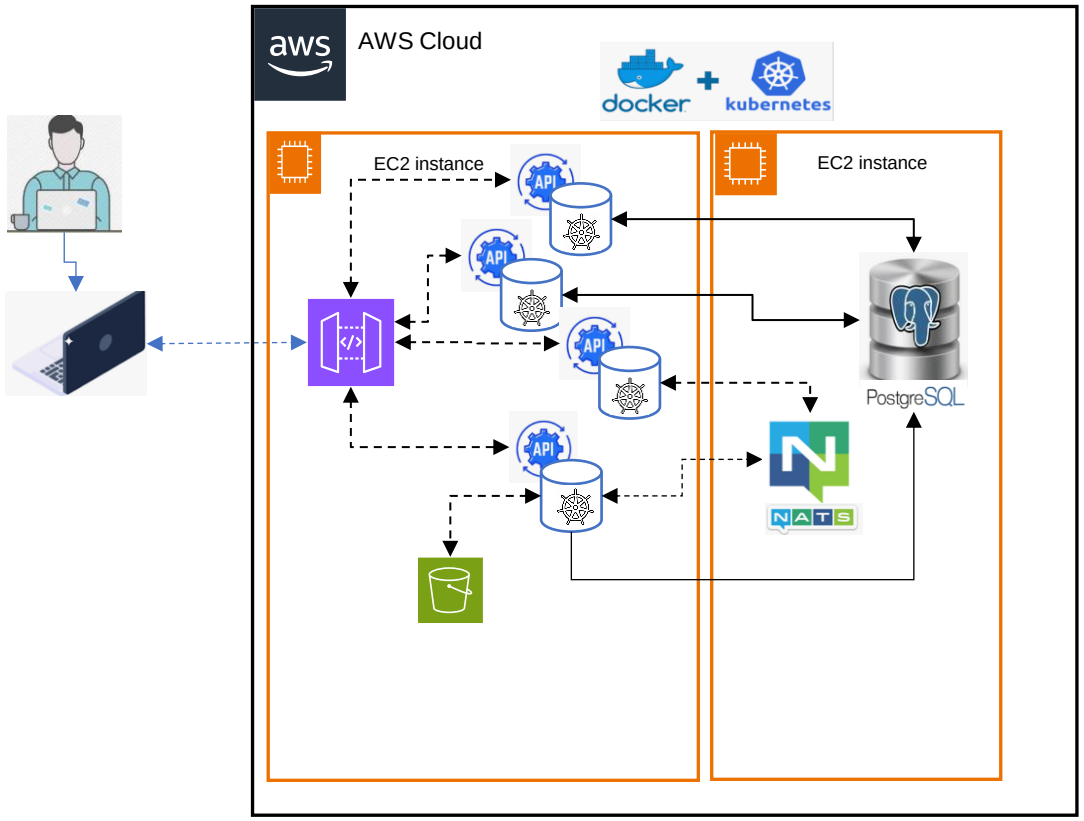
Objetivo do Desafio

Desenvolver uma arquitetura que integre processos e sistemas de forma eficiente, garantindo a entrega de valor para a organização. Isso inclui a definição de contextos, capacidades de negócio e domínios funcionais, escalabilidade das soluções para garantir alta disponibilidade, segurança e desempenho, a comunicação eficaz entre áreas e serviços, a seleção adequada de padrões arquiteturais, integração de tecnologias e frameworks, além de otimização de requisitos não-funcionais. Deve abranger a capacidade analítica, a visão sistêmica e a habilidade de criar soluções flexíveis e reutilizáveis.

Um comerciante precisa controlar o seu fluxo de caixa diário com os lançamentos (débitos e créditos), também precisa de um relatório que disponibilize o saldo diário consolidado.

Arquitetura para Fluxo de Caixa – domínios funcionais e capacidades de negócio

Arquitetura Alvo



Bounded Contexts

Contexto Fluxo de Caixa – Transações

- Registro de transações (débitos, créditos, cancelamentos)
- Atualização do saldo em tempo real
- Categorias das transações

Contexto Fluxo de relatórios Financeiros

- Geração do relatório consolidado com filtros
- Repositório de relatórios

Capacidades de negócio

Contexto Fluxo de Caixa – Transações

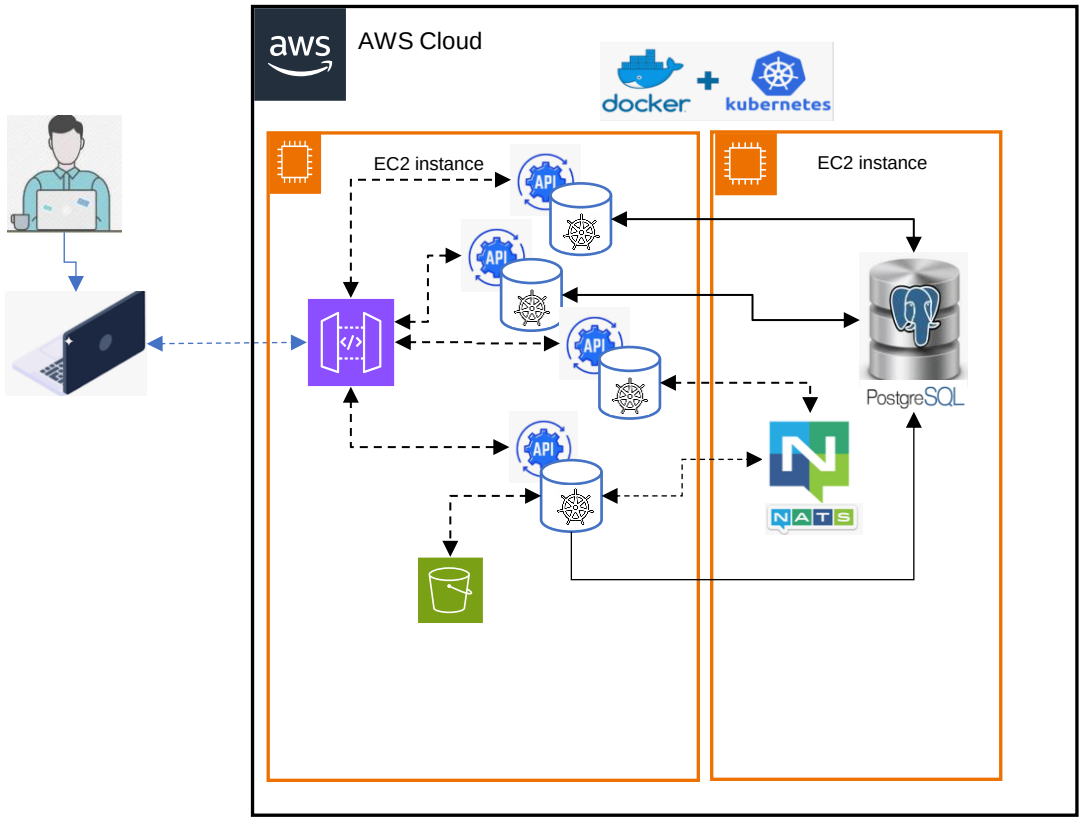
- Registrar uma transação (debito, crédito)
- Cancelar uma transação
- Atualizar o saldo em tempo real
- Categorizar transações

Contexto Fluxo de relatórios Financeiros

- Gerar relatório consolidado por período, tipo de lançamento e categoria
- Manter relatórios gerados por tempo determinado

Arquitetura para Fluxo de Caixa – Requisitos funcionais refinados

Arquitetura Alvo



Registrar transações

- Registrar transações financeiras de débito e crédito com validação de data e categorias no momento da solicitação
- Cancelar uma transação
- Atualização automática do saldo em tempo real no momento do registro da transação
- Cadastrar categorias de lançamentos

Gerar relatório financeiro

- Gerar relatório financeiro consolidado. A consolidação pode ser diária, semanal, mensal ou por um intervalo de datas específico
- O relatório pode ser personalizado por filtro de categorias de transação ou débito e/ou crédito
- O relatório poderá ser exportado em formato PDF ou CSV

Autenticação e permissões do usuário

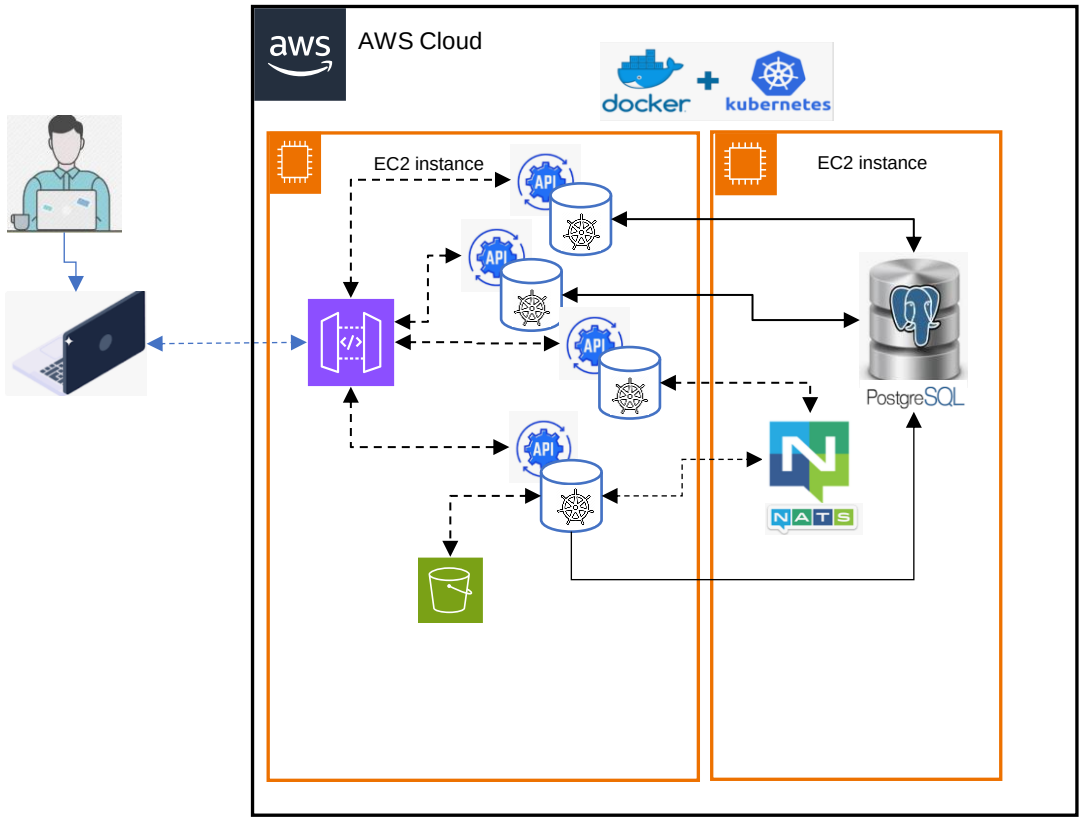
- O usuário deverá ser autenticado no momento de login do sistema através de uma senha
- O usuário deverá ter acesso ao sistema de acordo com o perfil definido:
  - Operador – Pode apenas gerar relatórios
  - Vendedor - pode fazer lançamentos de débito r crédito e gerar relatórios
  - Supervisor – pode fazer lançamentos de débito e crédito, pode gerar relatórios e cancelar lançamentos

Interface com o usuário

- A interface deve ser amigável, responsiva, adaptada para dispositivos móveis e desktops

Arquitetura para Fluxo de Caixa – Requisitos não funcionais refinados

Arquitetura Alvo



Performance
- Tempo de resposta para registro de transações <= 2s
- Geração de relatórios consolidados até 5s em dias de pico

Escalabilidade
- Suporte ai crescimento do numero de usuários e transações sem perda de desempenho
- Infraestrutura escalável horizontalmente

Disponibilidade
- Sistema disponível a 99,0% do tempo
- Janelas de manutenção limitadas a 1 hora por mês

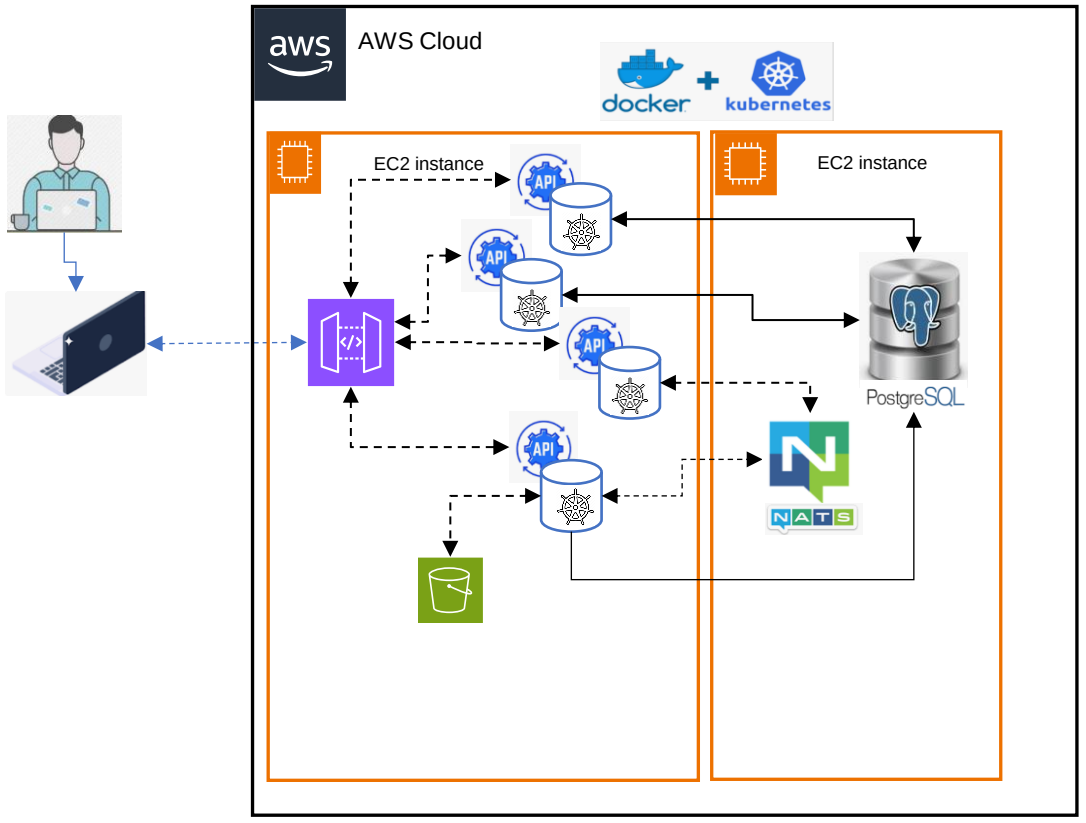
Segurança
- Criptografia de dados em trânsito (SSL/TLS)
- Criptografia de dados em repouso (AES-256)
- Proteção contra fraudes e validação de transações inconsistentes
- Armazenamento seguro de senhas com hashing

Usabilidade
- Navegação intuitiva para usuários não técnicos
- Compatibilidade total cm navegadores modernos e dispositivos móveis

Manutenibilidade
- A interface deve ser amigável, responsiva, adaptada para dispositivos móveis e desktops

Arquitetura para Fluxo de Caixa – Escolha da arquitetura e seus componentes

Arquitetura Alvo



A arquitetura baseada em AWS com PostgreSQL, NATS, Docker e Kubernetes é ideal para o sistema de fluxo de caixa, garantindo desempenho, segurança, escalabilidade e facilidade de manutenção. Esses componentes trabalham de forma sinérgica para oferecer uma solução confiável e preparada para o crescimento do negócio do comerciante

Itens considerados para tomada de decisão:

Escalabilidade e Alto desempenho
- O PostgreSQL oferece uma base de dados confiável e escalável para dados estruturados.
- O NATS reduz a carga no banco de dados, acelerando consultas frequentes.
- Docker e Kubernetes garantem que o sistema possa escalar horizontalmente sob demanda
- RabitMQ faz o gerenciamento de filas evitando sobrecarga e concorrência indevida no banco

Manutenibilidade
- A modularidade obtida com contêineres e orquestração facilita a manutenção e implantação de novas funcionalidades

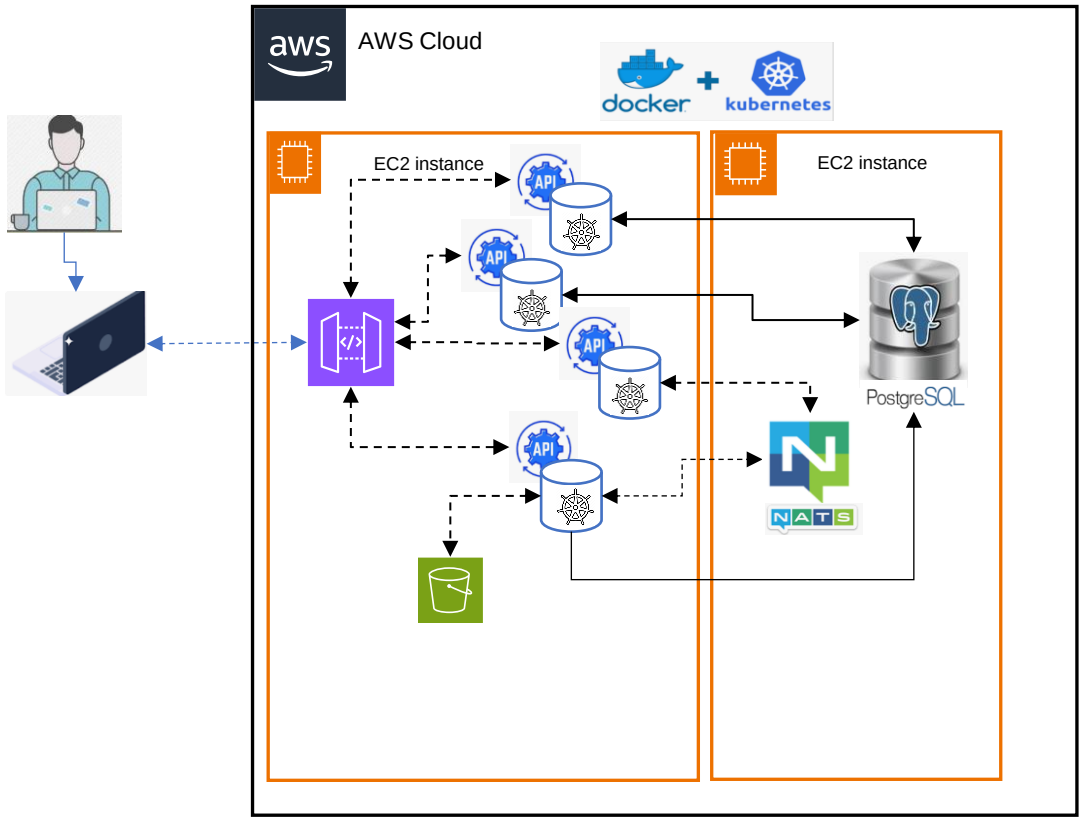
Alta disponibilidade e Disaster recovery
- A AWS oferece backups automáticos e failover integrado, garantindo disponibilidade contínua e recuperação de desastres.

Segurança
- A arquitetura aproveita serviços gerenciados da AWS para proteger dados em trânsito e em repouso.

Custos otimizados
- Os serviços gerenciados e a escalabilidade automática permitem que os custos sejam proporcionais à demanda real do cliente.

Arquitetura para Fluxo de Caixa – Padrões arquiteturais

Arquitetura Alvo



A solução adota uma **arquitetura baseada em microsserviços**, Hospeda em uma **nuvem AWS** segue princípios modernos de desenvolvimento e implantação de sistemas distribuídos. Este padrão arquitetural foi escolhido por sua capacidade de oferecer escalabilidade, modularidade e independência entre os componentes, além de facilitar a manutenção e a implantação contínua.

Os componentes são **divididos em camadas** bem definidas para front, aplicação e banco

Padrões arquiteturais adotados:

- **API Gateway** centralizando solicitações externas, gerencia autentica e roteia.
- **Circuit breaker** garantindo resiliência em caso de falha de um microsserviço os outros não são afetados

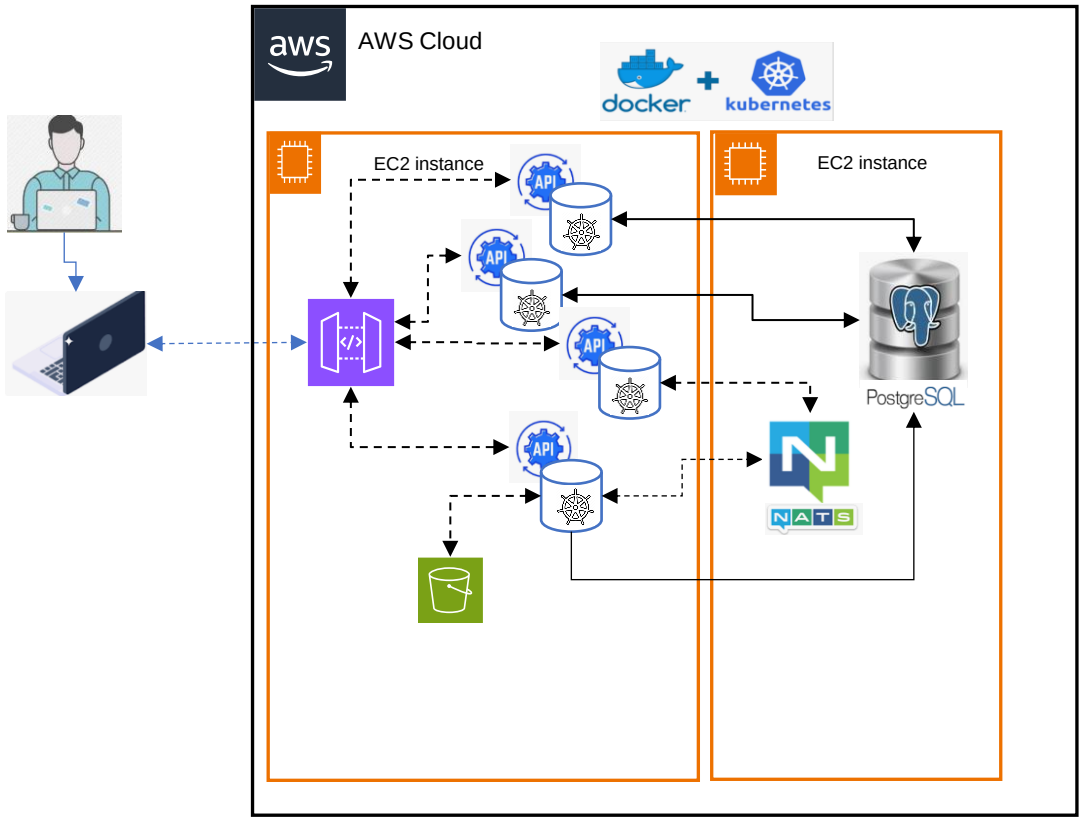
Outra característica desta arquitetura é **deploy continuo (CI/CD)** e **infraestrutura como código.** Também sustentada por tecnologias modernas como cloud(Aws), Kubernetes, Docker, PostgreSQL, NATS

APIs Criadas para esta solução

- Registrar Transação
- Consultar lançamentos
- Gerar relatório
- Consultar caracteristicas

Arquitetura para Fluxo de Caixa – Componentes AWS que compõem a solução

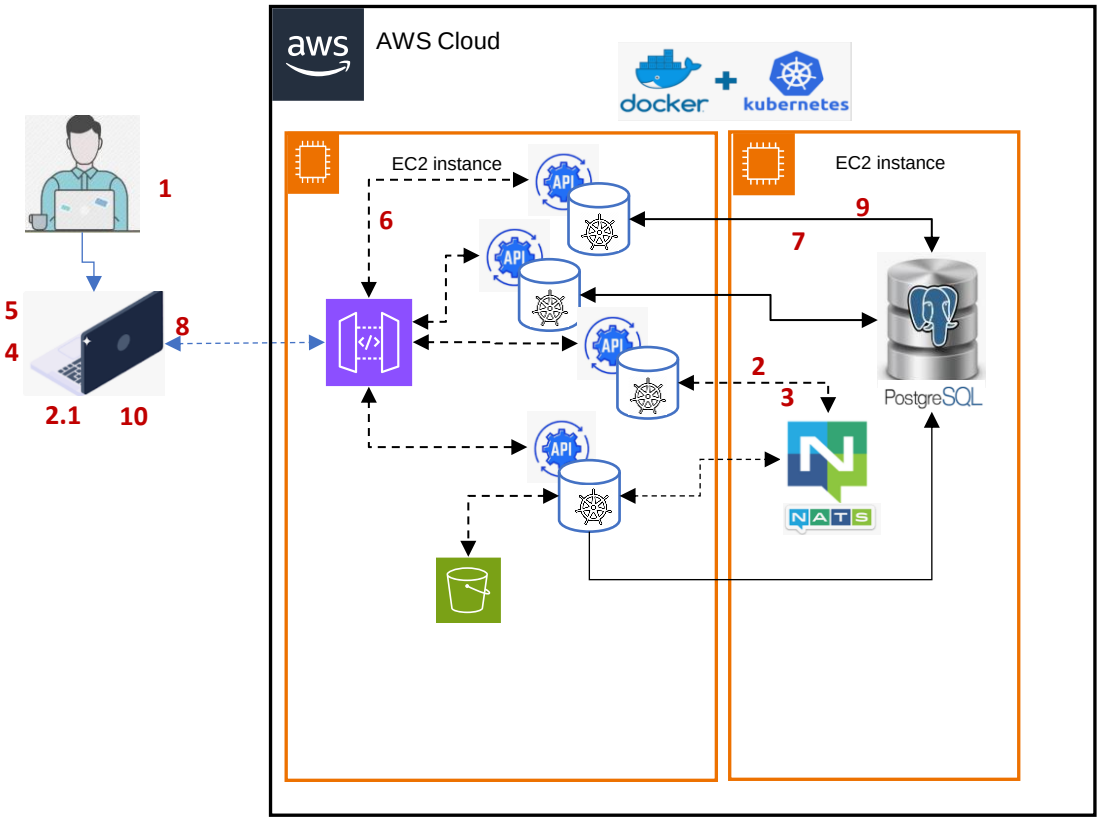
Arquitetura Alvo



Componente	Descrição
EC2	Hospedagem da aplicação e APIs.
RDS (PostgreSQL)	Armazenamento de dados transacionais (transações, saldos, cancelamentos).
S3	Armazenamento de relatórios gerados e logs.
API Gateway	Exposição das APIs para interação com o front-end.
Lambda	Processamento assíncrono de transações e geração de relatórios.
IAM	Controle de acesso seguro a recursos da AWS.
CloudFront	Distribuição rápida dos relatórios para os usuários.
Elasticache (opcional)	Cache de dados frequentemente acessados.
Amazon CloudWatch	Coleta e monitora métricas, logs e alarmes da infraestrutura e aplicação.
AWS X-Ray	Realiza rastreamento distribuído das requisições para monitorar latência e interações entre serviços.
Amazon RDS Performance Insights	Monitora o desempenho de bancos de dados RDS, identificando consultas lentas e gargalos.
AWS CloudTrail	Registra e audita todas as chamadas de API, permitindo rastrear ações e eventos na infraestrutura.
AWS Config	Monitora a configuração e o compliance dos recursos da AWS, garantindo a governança.
Amazon SNS	Envia notificações em tempo real sobre falhas ou eventos críticos na aplicação.
Amazon SQS	Garante o processamento assíncrono de eventos e monitora a integridade das filas de mensagens.

Arquitetura para Fluxo de Caixa – Desenho da solução – Fluxo de Registrar uma transação

Arquitetura Alvo

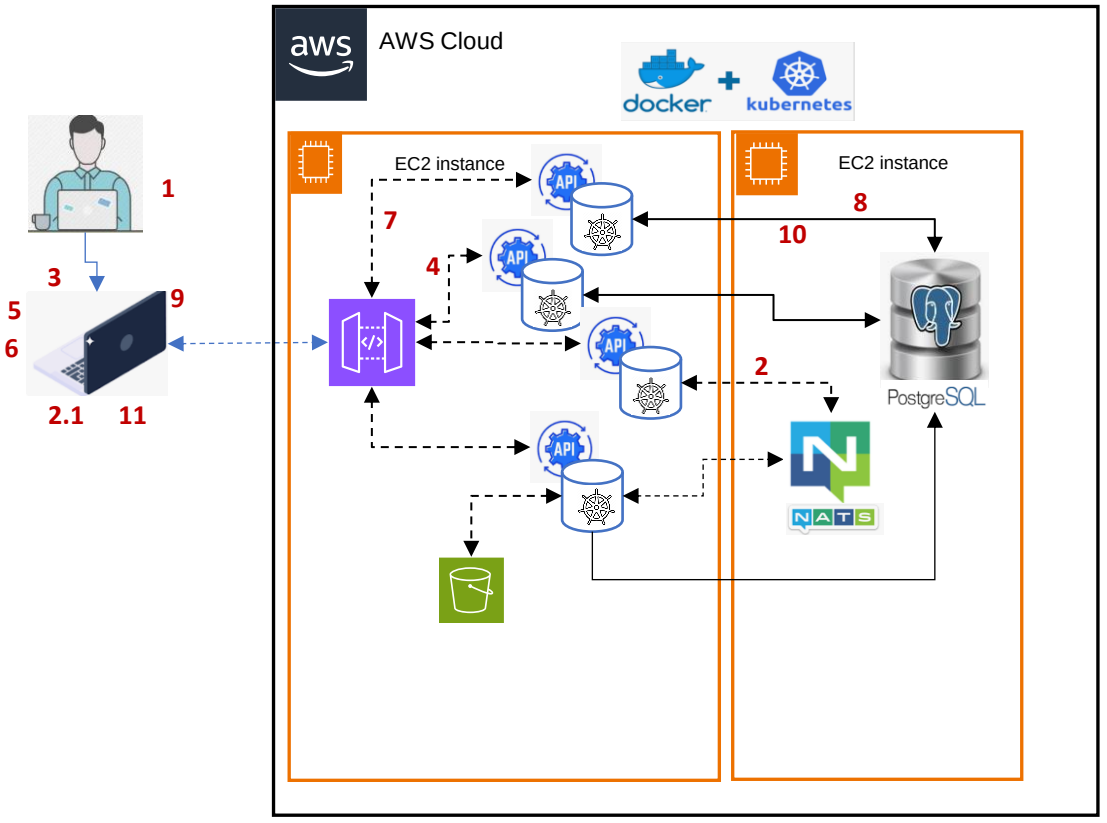


Fluxo da Solução registrar uma transação

1. O fluxo inicia com o usuário já logado no fron-end escolhendo na tela de lançamentos registrar uma transação
2. O sistema consulta a API **Consultar_Características** com parâmetro de “alçadas” para identificar o perfil do cliente
    1. Caso o perfil do cliente seja penas de consulta de relatórios, é enviado uma mensagem na tela informando que não existe perfil para esta ação e o fluxo se encerra mantendo desabilitadas funções, campos de texto e botões na tela de lançamentos.
3. O sistema consulta no cache da aplicação as categorias através da API **Consultar_características** com o parâmetro de “categorias” e exibe numa pick list as categorias do lançamento de acordo com a alçada do usuário
4. O sistema habilita os campos para o lançamento dos valores
5. O usuário insere os valores escolhendo a categoria correta e o tipo de transação (débito ou crédito) e submete para a aplicação
6. A aplicação chama a API **Registrar_Transação** e passa como parâmetros as informações do usuário
7. A API **Registrar_Transação** persiste no Banco PostergeSQL o lançamento
8. A API **Registrar_Transação** retorna para o front-end a mensagem de que a informação foi gravada
9. A API **Registrar_Transação** em segudo plano (backgorund) atualiza consulta o banco com o Saldo atual e faz o cálculo para ajustar o saldo e persiste novamente no banco a informação do saldo atualizado
10. O Fluxo se encerra com a tela habilitada na seção aberta aguardando novos registros

Arquitetura para Fluxo de Caixa – Desenho da solução – Fluxo de Cancelar uma transação

Arquitetura Alvo

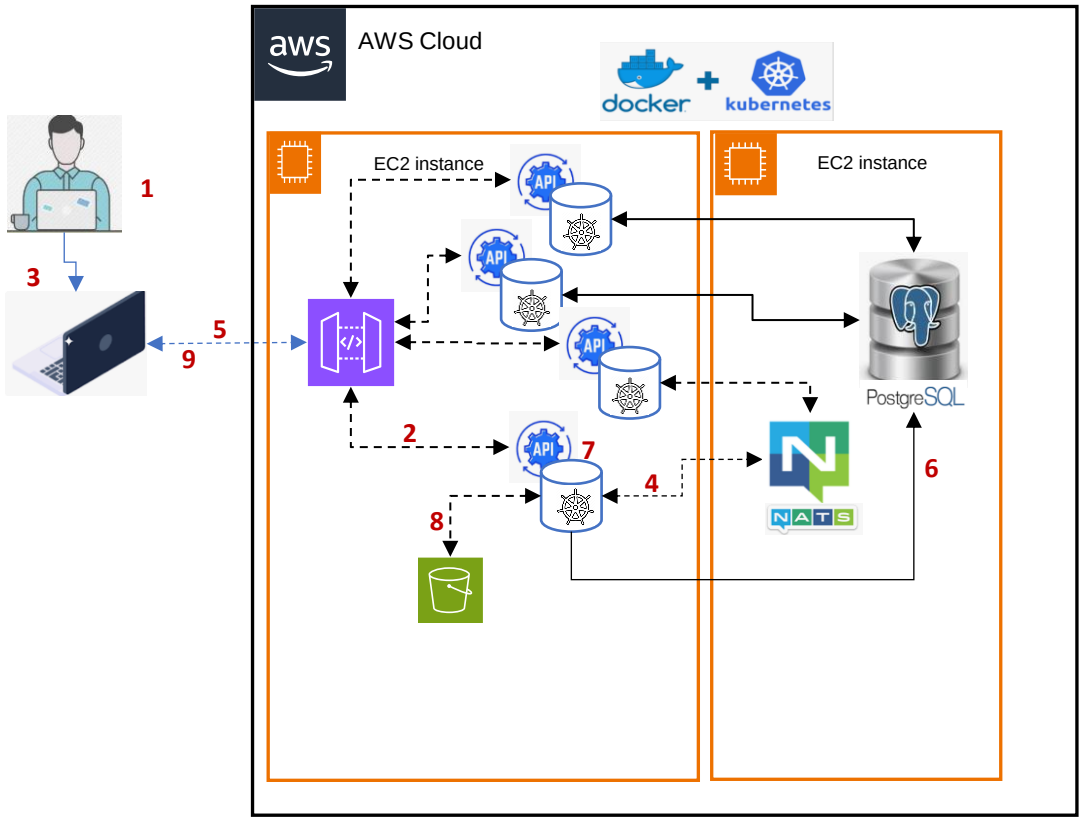


Fluxo da Solução cancelar uma transação

1. O fluxo inicia com o usuário já logado no fron-end escolhendo na tela de cancelamento de uma transação.
2. O sistema consulta a API **Consultar_Características** utilizando o parâmetro “Alçadas” para identificar o perfil do cliente
    1. Caso o perfil do cliente não seja de supervisor, é enviado uma mensagem na tela informando que não existe perfil para esta ação e o fluxo se encerra mantendo desabilitadas funções, campos de texto e botões na tela de lançamentos.
3. O sistema solicita ao usuário fazer um filtro por intervalo de datas, categoria e tipo de lançamento
4. O sistema consome a API **Consultar_Lançamentos** que retorna um range de lançamentos a partir de uma leitura no banco PostGreSQL
5. O sistema exibe na tela um grid de lançamentos possibilitando fazer múltiplas seleções e rolar paginando a informação
6. O usuário seleciona um ou mais lançamentos para serem cancelados e submete a transação
7. A aplicação chama a API **Registrar_Transação** e passa como parâmetro uma lista de lançamentos selecionados com a marcação “cancelamento”
8. A API **Registrar_Transação** recebe a informação e faz uma persistência massiva (bulk) no banco de dados com valores inversos ao original de cada lançamento todos marcados com a categoria “cancelamento”
9. A API **Registrar_Transação** retorna para o front-end a mensagem que os cancelamentos foram efetuados
10. A API **Registrar_Transação** calcula em segundo plano a atualização do saldo e persiste no banco o saldo atualizado.
11. O Fluxo se encerra com a tela habilitada na seção aberta aguardando novo filtro de registros

Arquitetura para Fluxo de Caixa – Desenho da solução – Fluxo de Gerar Relatório

Arquitetura Alvo

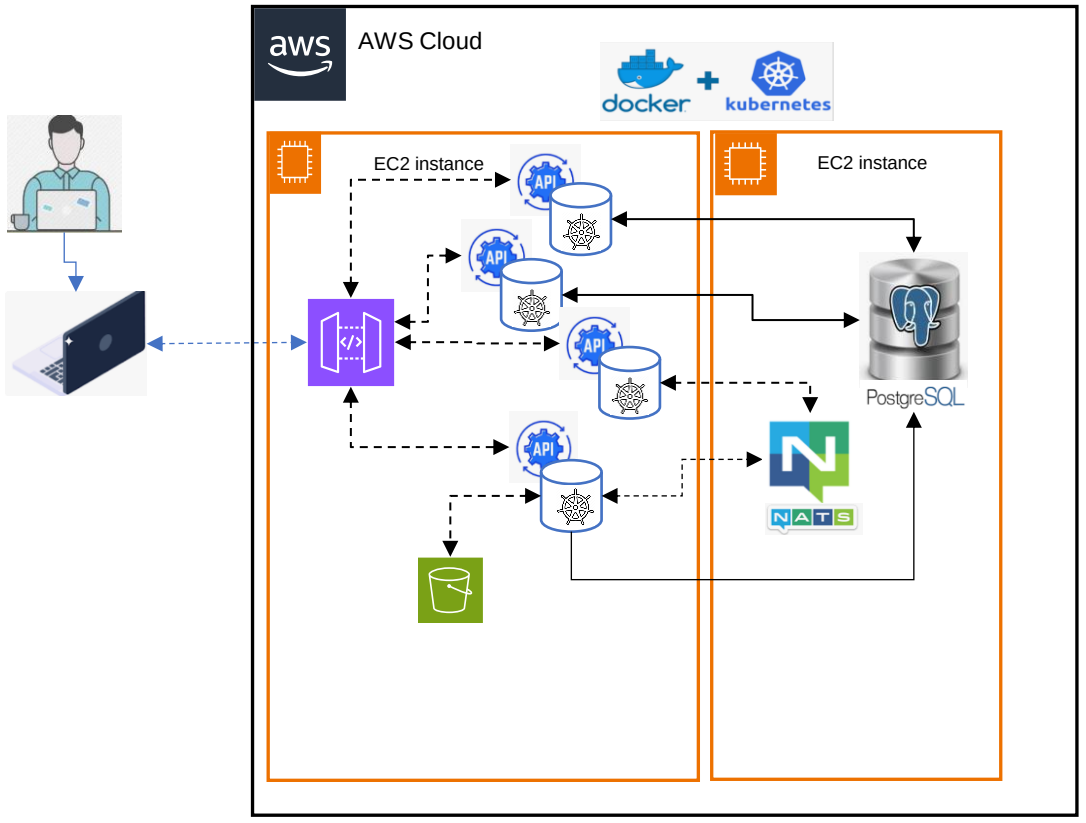


Fluxo da Solução gerar relatório

1. O fluxo inicia com o usuário já logado no fron-end escolhendo na tela de relatórios o período que deseja consultar o saldo consolidado.
2. O sistema consulta a API **Gerar_Relatório** passando os parâmetros informados pelo usuário
3. A API **Gerar_Relatório** verifica o cache da aplicação (NATS) se o link do relatório está disponível
4. Caso esteja disponível A API **Gerar_Relatório** retorna para a aplicação o link do relatório para download e atualiza o status para concluído
5. Caso o link do relatório não esteja disponível no NATS, a API **Gerar_Relatório** atualiza o status no front-end informando que o relatório está em construção.
6. A API **Gerar_Relatório** consulta os dados do relatório no banco PostgreSQL
7. a API **Gerar_Relatório** consolida a informação do relatório, gera o arquivo no formato solicitado com a nomenclatura sugerida <*YYYYMMDD_INI_YYYYMMDD_FIM_Fluxo_caixa*> e atualiza o bucket S3
8. Após atualizar o bucket S3 a API **Gerar_Relatório** atualiza o cache NATS
9. A API **Gerar_Relatório** retorna para a aplicação o link do relatório para download e atualiza o status para concluído.

Arquitetura para Fluxo de Caixa – Estimativa de custo

Arquitetura Alvo



Categoria	Descrição	Custo Estimado (R$)
Desenvolvimento e Implementação	Custo com desenvolvedores para implementar as funcionalidades do projeto (registro de transações, cancelamento, relatórios, etc.)	24.000,00
Infraestrutura	Custo de servidores para rodar o sistema (AWS, Google Cloud, ou servidores dedicados)	6.000,00
Banco de Dados (PostgreSQL)	Custo do banco de dados PostgreSQL hospedado (dependendo do provedor de nuvem, como AWS RDS, DigitalOcean, etc.)	3.000,00
Armazenamento na Nuvem (S3)	Custo de armazenamento dos relatórios gerados em S3 (Amazon Web Services)	600
Licenças e Ferramentas	Licenciamento de ferramentas utilizadas, como ferramentas de CI/CD ou outras dependências comerciais	1.800,00
Manutenção e Suporte	Custo estimado de manutenção mensal após a entrega (suporte técnico, correção de bugs, atualizações, etc.)	24.000,00
Testes e Validação	Custos com testes manuais e automáticos (incluindo horas de desenvolvedores e QA)	10.500,00
Treinamento e Documentação	Custos com elaboração da documentação e treinamento para usuários e equipe técnica	6.000,00
Total Estimado	Total de todos os custos estimados para o projeto	75.900,00

Racional utilizado:
Comparativo encontrado em pesquisas na internet e informações públicas nos sites dos fornecedores

Sites consultados:
Classdoor, vagas.com, AWS Pricing, AWS RDS pricing, Google cloud SQL pricing, Google Cloud Pricing, DigitalOcean pricing, Amazon S3 pricing, Github Actions Pricong, Jetkins pricing, QA & Test Automation services

OBRIGADO!