Flow Wise

Uma solução estratégica e moderna para o fluxo de caixa.

Sumário Executivo

Esta seção oferece uma visão concisa e de alto nível do Projeto Flow Wise, destacando seus objetivos estratégicos, o valor de negócio que ele entrega e os principais elementos da solução.

É projetada para líderes e executivos que precisam de uma compreensão rápida e impactante do projeto

Propósito e Visão Geral

O Projeto Flow Wise surge como uma solução estratégica para modernizar e otimizar o controle de fluxo de caixa diário da organização.

Atualmente, a gestão financeira enfrenta desafios de visibilidade, precisão e eficiência devido à fragmentação de dados e processos manuais.

O Flow Wise visa centralizar, automatizar e prover insights acionáveis sobre o fluxo de caixa, permitindo decisões financeiras mais ágeis e assertivas, além de preparar a organização para futuras capacidades de análise preditiva através de Inteligência Artificial.

Valor de Negócio e Impacto Estratégico

A implementação do Flow Wise trará impactos significativos para a organização, alinhando-se diretamente aos nossos objetivos estratégicos de eficiência e inovação:

- **Melhora na Tomada de Decisão:** Visão consolidada e em tempo real do fluxo de caixa, permitindo planejamento financeiro proativo e redução de riscos.
- **Aumento da Eficiência Operacional:** Automação de registros e relatórios, reduzindo o esforço manual, minimizando erros e liberando a equipe financeira para análises mais estratégicas.
- **Redução de Riscos:** Melhor controle e rastreabilidade dos lançamentos, mitigando fraudes, inconsistências e dependência de planilhas e processos manuais.
- **Preparação para o Futuro:** Arquitetura flexível e modular que facilitará futuras integrações e a incorporação de funcionalidades avançadas, como projeções de fluxo de caixa e análise de anomalias baseadas em Inteligência Artificial, posicionando a empresa na vanguarda da gestão financeira.

Escopo de Alto Nível

O escopo inicial do Flow Wise abrange a gestão completa de lançamentos de débito e crédito, a consolidação diária do saldo financeiro e a geração de relatórios de saldo consolidado.

O sistema será projetado para ser a base de futuras expansões, incluindo integrações com sistemas externos e funcionalidades de análise preditiva.

Principais Tecnologias e Abordagem Arquitetural

O Flow Wise será construído sobre uma **arquitetura de Microsserviços**, utilizando as mais modernas tecnologias **C#/.NET 8+**.

Esta escolha estratégica, alinhada com o *know-how existente da nossa equipe de engenharia*, fundamenta-se na necessidade de alta **escalabilidade**, **resiliência** e **independência** no desenvolvimento e implantação.

Isso garante que o sistema possa evoluir rapidamente e operar com máxima disponibilidade, especialmente para serviços críticos como a consolidação diária, **otimizando o tempo de desenvolvimento e a curva de aprendizado**.

Serão aplicados padrões de arquitetura como **Domain-Driven Design (DDD)**, **CQRS (Command Query Responsibility Segregation)** e, para dados sensíveis, **Event Sourcing**, promovendo um design limpo e manutenível.

A segurança será um pilar fundamental, com autenticação robusta e proteção de dados.

Próximos Passos

As próximas fases do projeto envolve o detalhamento da arquitetura de software (Nível Contêineres e Componentes do C4 Model), o design aprofundado dos microsserviços, a validação contínua dos requisitos com as áreas de negócio e o planejamento detalhado das atividades de desenvolvimento e implantação.

Introdução ao Projeto

Cenário Atual e Desafios

Atualmente, a gestão de fluxo de caixa diário na organização é majoritariamente realizada através de planilhas eletrônicas e processos manuais.

Esta abordagem, embora funcional em menor escala, apresenta diversos desafios significativos que impactam a eficiência e a capacidade de tomada de decisão:

- **Dados Fragmentados e Descentralizados:** As informações financeiras estão dispersas em múltiplas planilhas e sistemas legados, dificultando a consolidação de uma visão única e precisa do fluxo de caixa.
- **Susceptibilidade a Erros Manuais:** A dependência de entrada e manipulação manual de dados aumenta o risco de erros de digitação, cálculos incorretos e inconsistências, comprometendo a integridade financeira.
- **Falta de Visibilidade em Tempo Real:** A consolidação dos dados é um processo demorado e manual, resultando em uma visão tardia do fluxo de caixa e impossibilitando a tomada de decisões ágeis e proativas.
- **Dificuldade de Auditoria e Rastreabilidade:** Acompanhar a origem e as alterações de lançamentos financeiros torna-se complexo, dificultando auditorias internas e externas e a identificação de fraudes ou desvios.
- **Dependência de Pessoas Chave:** O conhecimento e a operação do fluxo de caixa estão concentrados em indivíduos específicos, criando um ponto único de falha e dificultando a continuidade operacional em caso de ausência.
- **Barreiras à Escalabilidade:** O modelo atual não suporta o volume crescente de transações e a demanda por análises mais sofisticadas, limitando o crescimento da organização.

Visão do Produto Flow Wise

O Flow Wise será a plataforma centralizada, inteligente e resiliente para a gestão financeira diária da organização.

Ele proverá uma visão clara, precisa e preditiva do fluxo de caixa, capacitando as equipes financeiras a operar com máxima eficiência, a identificar oportunidades e riscos financeiros antecipadamente, e a tomar decisões estratégicas embasadas em dados.

Nosso objetivo é transformar a gestão de fluxo de caixa de um processo reativo e manual para uma operação proativa, automatizada e com valor estratégico para o negócio.

Objetivos do Projeto (SMART)

Os objetivos do Projeto Flow Wise foram definidos seguindo a metodologia SMART (Specific, Measurable, Achievable, Relevant, Time-bound), garantindo clareza, direcionamento e a capacidade de medir o sucesso da iniciativa.

Objetivos Funcionais

Nesta seção, será abordado “O QUE o sistema deve fazer”.

O-FUNC-001: Automação do Registro de Lançamentos:

- **Objetivo:** Permitir o registro manual e eficiente de todos os lançamentos de débito e crédito diários.
- **Medida de Sucesso:** 100% dos lançamentos manuais realizados através do sistema Flow Wise, eliminando a dependência de planilhas, até o final do Q4/2025.

O-FUNC-002: Consolidação e Visualização de Saldo Diário:

- **Objetivo:** Fornecer uma visão consolidada e precisa do saldo de caixa diário (D-1).
- **Medida de Sucesso:** Disponibilização de um relatório de saldo diário consolidado para o dia anterior (D-1) com 100% de precisão e atualizado até 8h da manhã de cada dia útil, a partir do primeiro mês de operação.

O-FUNC-003: Geração de Relatórios Financeiros:

- **Objetivo:** Possibilitar a geração de relatórios de fluxo de caixa por período, com filtros e agrupamentos básicos.
- **Medida de Sucesso:** Capacidade de gerar relatórios de fluxo de caixa para qualquer período selecionado em até 5 segundos, a partir do primeiro mês de operação.

Objetivos Não-Funcionais

Nesta seção, será abordado “COMO o sistema deve se comportar”.

O-NFR-001: Desempenho do Serviço de Consolidação:

- **Objetivo:** Assegurar que o serviço de consolidação diária suporte alta demanda.
- **Medida de Sucesso:** O serviço de consolidação diária deve sustentar **50 requisições por segundo**, com uma taxa de sucesso de no mínimo **95%** (ou seja, no máximo 5% de perda/erros) em dias de pico, a partir da fase de homologação.

O-NFR-002: Disponibilidade do Serviço de Lançamentos:

- **Objetivo:** Garantir a máxima disponibilidade para o registro de operações financeiras.
- **Medida de Sucesso:** O microsserviço de gestão de lançamentos deve atingir uma disponibilidade de **99.99%** em produção (equivalente a no máximo ~5 minutos de inatividade anual), a partir do go-live.

O-NFR-003: Resiliência e Isolamento de Falhas:

- **Objetivo:** Prover que a indisponibilidade de um serviço não afete a operação crítica de outro.

- **Medida de Sucesso:** O registro de novos lançamentos **não deve ser afetado** pela indisponibilidade ou lentidão do serviço de consolidação de relatórios, validado por testes de resiliência antes do go-live.

O-NFR-004: Segurança e Acesso Controlado:

- **Objetivo:** Proteger os dados financeiros e o acesso ao sistema.
- **Medida de Sucesso:** Implementar autenticação via SSO corporativo (Okta/Azure AD) e autorização granular (RBAC) com zero vulnerabilidades críticas detectadas em testes de segurança (SAST/DAST/Pentest) antes do go-live.

Escopo Detalhado

Esta seção detalha as funcionalidades que serão desenvolvidas no Flow Wise, dividindo-as entre o escopo inicial (MVP) e as funcionalidades futuras previstas no roadmap (do ponto de vista técnico/funcional).

Funcionalidades Iniciais (MVP)

As funcionalidades que compõem o Produto Mínimo Viável (MVP) do Flow Wise são focadas em entregar valor rapidamente, automatizando os processos mais críticos de controle de fluxo de caixa.

Registro manual de lançamentos de débito e crédito

Permite que os usuários autorizados (Analistas Financeiros) insiram manualmente os detalhes de cada movimentação financeira (entrada ou saída) no sistema.

Este processo substituirá as planilhas manuais existentes.

Usuários Principais: Principalmente **Analistas Financeiros**, mas também poderá ser acessado por **Gerentes Financeiros** para consulta ou pequenos ajustes (se possuir acesso ao serviço).

Responsabilidade: O **Microsserviço de Lançamentos** será o responsável por receber, validar, persistir e gerenciar esses registros.

Interface de Usuário (UI): Criação de uma tela intuitiva para inserção de dados (Web Application).

Persistência: Armazenamento seguro dos dados do lançamento em um banco de dados transacional. (ex: PostgreSQL).

Auditoria: Registro automático de quem criou/alterou o lançamento e quando.

APIs: Exposição de endpoint POST para criação de lançamentos.

Campos Requeridos:

- **Valor:** Numérico, obrigatório, positivo.
- **Tipo de Lançamento:** (Débito/Crédito) - Seleção de lista.
- **Data do Lançamento:** Data (obrigatório, não pode ser futura).
- **Descrição:** Texto livre (obrigatório, máximo 255 caracteres).
- **Categoria:** Seleção de lista predefinida (ex: 'Salários', 'Aluguel', 'Vendas', 'Serviços', 'Fornecedores'). A gestão de categorias será uma funcionalidade separada em roadmap ou via integração com um sistema de cadastro existente.
- **Observações:** Texto livre opcional.

Validações:

- Validação de campos obrigatórios.
- Validação de formato e tipo de dados (ex: valor numérico, data).
- Validação de regras de negócio (ex: data não pode ser futura).

Consulta e Edição de Lançamentos

Permite que os usuários consultem a lista de lançamentos registrados, aplicando filtros (por data, tipo, categoria) e editem ou excluam lançamentos dentro das regras de negócio estabelecidas.

Usuários Principais: Analistas Financeiros, Gerentes Financeiros.

Responsabilidade: O **Microsserviço de Lançamentos** será o responsável por fornecer os dados para consulta e por processar as requisições de edição/exclusão, com validações.

Regras de Negócio para Edição/Exclusão: Lançamentos só podem ser editados/excluídos dentro de 24h após a criação ou até o fechamento diário (RN-001).

APIs: Endpoints para consulta GET, edição PUT e exclusão DELETE.

Geração de Relatórios de Saldo Diário Consolidado

Fornece uma visão consolidada do saldo total de caixa (Créditos - Débitos) para um dia específico, com dados atualizados até o dia anterior (D-1).

Usuários Principais: Gerentes Financeiros, Analistas Financeiros.

Responsabilidade: O **Microsserviço de Consolidados** será o responsável por processar os dados dos lançamentos e gerar o saldo consolidado. Este serviço será independente do serviço de lançamentos.

Persistência da Consolidado: Armazenamento do saldo consolidado em um banco de dados otimizado para leitura (Query Model, CQRS).

Regra de Negócio: O consolidado é sempre de D-1 (dia anterior).

APIs: Endpoint para consulta do saldo consolidado GET.

Funcionalidades Futuras (Roadmap)

O roadmap do Flow Wise prevê expansões significativas que aumentarão ainda mais seu valor estratégico:

- **Integração com Sistemas Bancários:** Importação automatizada de extratos bancários para preenchimento de lançamentos.
- **Integração com ERP:** Sincronização de dados financeiros com o sistema ERP da organização (SAP/Oracle).
- **Módulos de Projeção de Fluxo de Caixa (IA):** Utilização de algoritmos de Inteligência Artificial para prever o fluxo de caixa futuro.
- **Dashboard Interativo e Personalizável:** Criação de um painel de controle dinâmico para visualização de métricas e tendências financeiras.
- **Alertas e Notificações Personalizadas:** Configuração de alertas para eventos financeiros específicos (ex: saldo abaixo de um limite, grandes lançamentos).
- **Gestão de Lançamentos Recorrentes:** Funcionalidade para cadastrar lançamentos que se repetem periodicamente.
- **Auditoria:** Registro de todas as edições e exclusões.
- **UI de Consulta:** Tela para listar lançamentos, com filtros e paginação.
- **UI de Edição:** Formulário pré-preenchido para edição de lançamentos existentes.
- **UI de Relatório:** Tela simples para seleção de data e visualização do saldo consolidado.
- **Reprocessamento de Consolidação:** Mecanismo que quando acionado, consolida os lançamentos de uma data alvo.

Glossário de Termos

Para garantir o alinhamento de todos os envolvidos no projeto, segue o glossário dos termos técnicos e de negócio utilizados no Projeto Flow Wise:

- **Lançamento:** Registro individual de uma movimentação financeira, podendo ser de débito (saída de caixa) ou crédito (entrada de caixa).
- **Saldo Diário Consolidado:** O valor líquido do fluxo de caixa (créditos menos débitos) para um dia específico, gerado após o fechamento do dia anterior.
- **MVP (Minimum Viable Product):** Produto Mínimo Viável. A versão inicial do sistema Flow Wise com as funcionalidades essenciais para entregar valor de negócio.
- **NFR (Non-Functional Requirement):** Requisito Não-Funcional. Descreve como o sistema deve se comportar (desempenho, segurança, disponibilidade, etc.).
- **Microsserviço:** Abordagem arquitetural onde uma aplicação é construída como um conjunto de pequenos serviços independentes e fracamente acoplados, cada um executando um processo único e se comunicando através de interfaces bem definidas.
- **DDD (Domain-Driven Design):** Abordagem de desenvolvimento de software que foca na modelagem de um domínio de negócio complexo, com a colaboração de especialistas de domínio e desenvolvedores.

- **CQRS (Command Query Responsibility Segregation):** Padrão que separa as operações de leitura (Queries) e escrita (Commands) de um sistema, permitindo que cada lado seja otimizado independentemente.
- **Event Sourcing:** Padrão onde todas as mudanças no estado da aplicação são armazenadas como uma sequência de eventos. Permite a reconstrução do estado a qualquer momento e facilita auditoria e projeções.
- **RTO (Recovery Time Objective):** Tempo Máximo de Recuperação. O tempo máximo aceitável para que um sistema ou serviço seja restaurado após uma interrupção.
- **RPO (Recovery Point Objective):** Ponto Máximo de Recuperação. A quantidade máxima de dados que pode ser perdida após uma interrupção.
- **SSO (Single Sign-On):** Autenticação única que permite que um usuário acesse múltiplos sistemas com um único conjunto de credenciais.

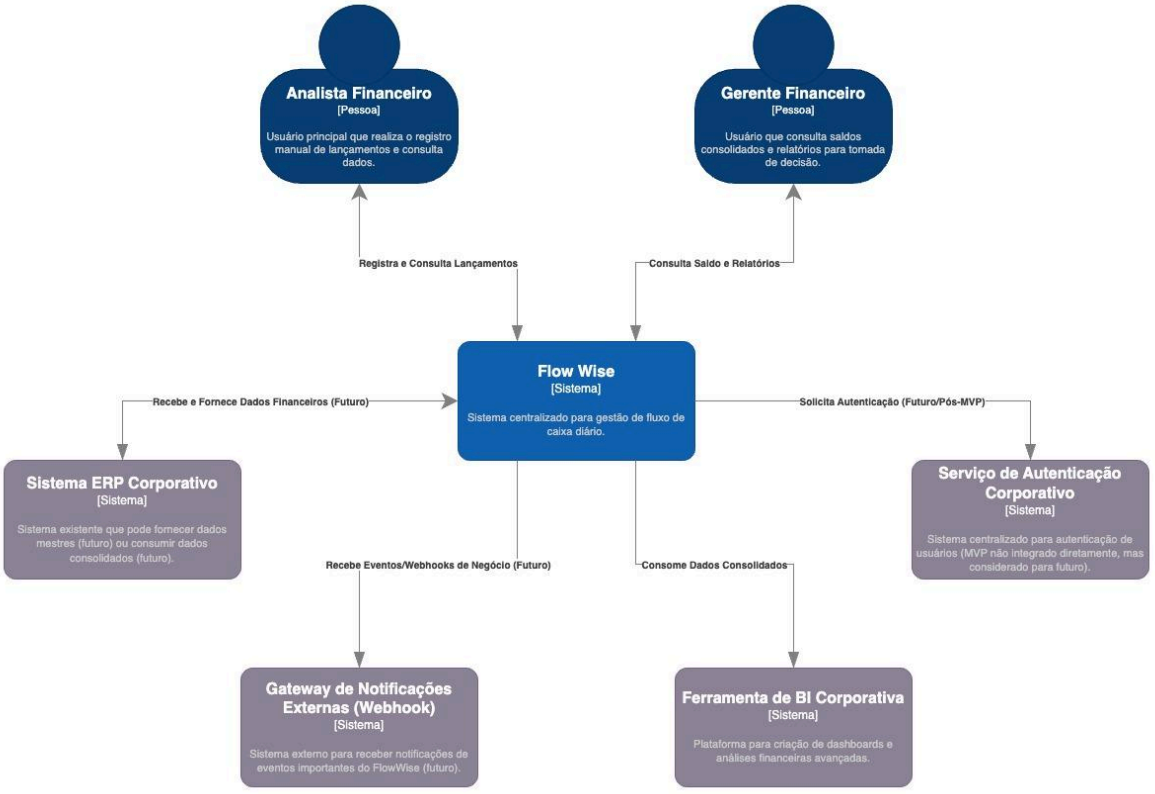
Arquitetura da Solução - Nível Contexto

Esta seção apresenta uma visão de alto nível do sistema Flow Wise, descrevendo como ele se encaixa no ecossistema de TI da organização e suas principais interações.

Visão Geral do Sistema Flow Wise

O diagrama de contexto abaixo ilustra o sistema Flow Wise como um todo, como ele se relaciona com os usuários e outros sistemas da organização, e os fluxos de informação de alto nível. Este nível do C4 Model foca nas interações externas e no papel do Flow Wise dentro do ecossistema de TI da empresa.

Diagrama de Contexto (C4 Model)



A figura 1.0 apresenta o diagrama de contexto do Projeto Flow Wise.

Descrição dos Elementos do Diagrama

Detalhes sobre cada elemento representado no Diagrama de Contexto:

- **Sistema Flow Wise:** A aplicação de software a ser desenvolvida, responsável por centralizar e gerenciar o fluxo de caixa diário da organização.
- **Analista Financeiro:** Pessoa que utilizará o Flow Wise para registrar e consultar lançamentos financeiros de débito e crédito.
- **Gerente Financeiro:** Pessoa que utilizará o Flow Wise para obter uma visão consolidada do fluxo de caixa e acessar relatórios estratégicos.

- **Sistema ERP Corporativo (Ex: SAP/Oracle):** Sistema transacional da empresa que pode, no futuro, interagir com o Flow Wise para troca de dados financeiros consolidados ou de cadastro.
- **Ferramenta de BI Corporativa (Ex: Power BI/Tableau):** Plataforma analítica utilizada para visualização e análise de dados, que consumirá informações consolidadas do Flow Wise para paineis de controle e relatórios gerenciais.
- **Serviço de Autenticação Corporativo (Ex: Okta/Azure AD):** Sistema responsável por gerenciar as identidades e o acesso dos usuários na organização, com o qual o Flow Wise se integrará para autenticação centralizada (pós-MVP).
- **Gateway de Notificações Externas (Webhook):** Um ponto de integração para parceiros externos ou outros sistemas que poderão receber notificações de eventos de negócio do Flow Wise de forma assíncrona (pós-MVP).

Arquitetura de Software - Nível Contêineres

Esta seção aprofunda a arquitetura do Flow Wise, detalhando os principais contêineres que compõem o sistema.

Um contêiner representa uma aplicação executável ou um armazenamento de dados que hospeda código ou dados, e esta visão é crucial para entender como os diferentes serviços interagem para entregar as funcionalidades.

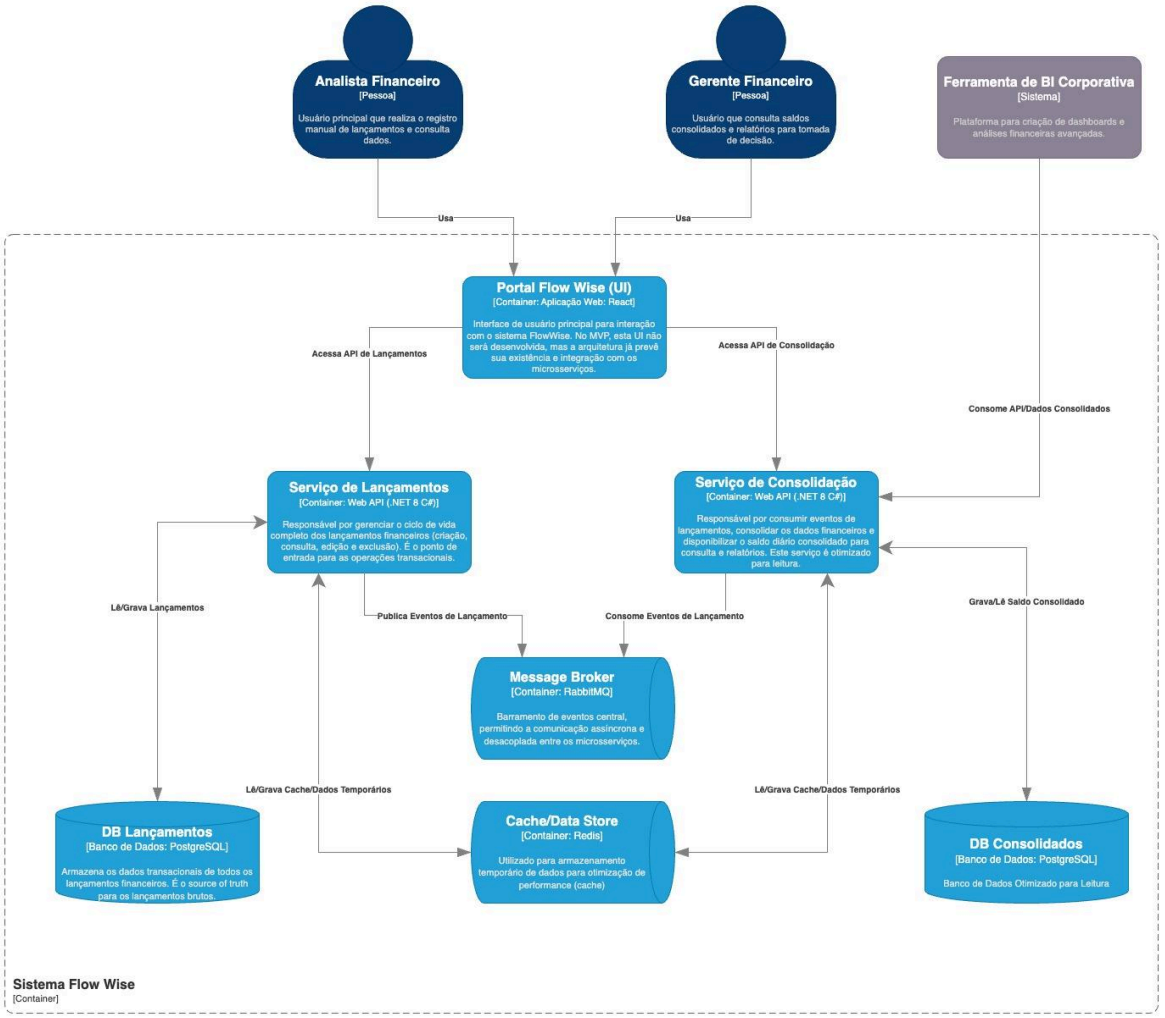
Visão Geral dos Contêineres

O sistema Flow Wise é composto por um conjunto de microsserviços e componentes de infraestrutura que trabalham em conjunto para gerenciar o fluxo de caixa.

A escolha por microsserviços permite o desenvolvimento, implantação e escalabilidade independentes de cada componente, promovendo resiliência e agilidade.

Para o MVP, esses serviços serão agrupados logicamente em um único repositório (*monorepo*), facilitando a gestão inicial, mas projetados para serem independentemente implantáveis e escaláveis.

Diagrama de Contêineres (C4 Model)



A Figura 2.0 apresenta o diagrama de contêineres do Projeto Flow Wise, detalhando a arquitetura interna da solução e como seus principais componentes interagem.

Descrição dos Elementos do Diagrama

Detalhes sobre cada contêiner representado no Diagrama de Contêineres:

Aplicação Web Flow Wise (UI):

- **Tecnologia:** Aplicação Web com React.
- **Responsabilidade:** Será a interface de usuário principal do sistema Flow Wise, permitindo que Analistas e Gerentes Financeiros interajam com as funcionalidades de registro, consulta e relatórios. **Para o MVP, esta aplicação não será desenvolvida**, e a interação será via API (ex: Swagger/Postman), mas sua existência é prevista e arquitetada para futuras fases.

Serviço de Lançamentos:

- **Tecnologia:** C#/.NET 8+ (API Web).

- **Responsabilidade:** Único responsável pela criação, atualização, exclusão e consulta de lançamentos financeiros individuais. Implementa o "lado de comando" do CQRS para as operações de escrita transacionais.

Serviço de Consolidação:

- **Tecnologia:** C#/.NET 8+ (Aplicação de Processamento de Eventos e API Web).
- **Responsabilidade:** Consome os eventos de lançamentos publicados pelo Serviço de Lançamentos. Processa esses eventos de forma assíncrona para construir e manter uma projeção otimizada para leitura (Query Model) do saldo de caixa diário. Expõe uma API para consulta do saldo consolidado e para a geração de relatórios. Implementa o "lado de query" do CQRS.

RabbitMQ:

- **Tecnologia:** Message Broker (Servidor de Mensagens).
- **Responsabilidade:** Facilita a comunicação assíncrona entre os microsserviços Flow Wise. Garante que eventos de negócio (como a criação de um lançamento) sejam entregues de forma confiável e desacoplada, mesmo se o serviço consumidor estiver temporariamente indisponível. Fundamental para a resiliência e independência entre os serviços.

Redis:

- **Tecnologia:** Banco de Dados em Memória (Key-Value Store).
- **Responsabilidade:** Utilizado como uma camada de cache para dados frequentemente acessados pelos serviços de Lançamentos e Consolidação, melhorando a performance e reduzindo a carga sobre os bancos de dados primários. Também pode ser utilizado para mecanismos de *rate limiting* e armazenamento temporário de estados distribuídos.

DB Lançamentos:

- **Tecnologia:** Banco de Dados PostgreSQL (Relacional, com potencial para uso como banco de dados não relacional).
- **Responsabilidade:** Armazenar de forma durável e transacional todos os dados brutos dos lançamentos financeiros. É o *source of truth* para as operações de débito e crédito.

DB Consolidados (Query Model):

- **Tecnologia:** Banco de Dados PostgreSQL Otimizado para Leitura (Relacional, com potencial para uso como banco de dados não relacional).
- **Responsabilidade:** Armazenar o saldo de caixa diário e outras projeções de dados financeiros em um formato otimizado para consultas rápidas e geração de relatórios, desacoplado do banco de dados transacional de lançamentos.

Analista Financeiro, Gerente Financeiro, Ferramenta de BI Corporativa:

- Conforme descrito no Diagrama de Contexto, estes são os usuários e sistemas externos que interagem com os contêineres do Flow Wise, primariamente através da **Aplicação Web Flow Wise (UI)** para usuários humanos (no pós-MVP) e diretamente via APIs para sistemas externos como a Ferramenta de BI.

Como os Contêineres Interagem

As interações entre os contêineres do Flow Wise são projetadas para garantir a resiliência, escalabilidade e manutenibilidade da solução:

Comunicação de Lançamentos:

1. O **Analista Financeiro** interage com a **Aplicação Web Flow Wise (UI)** (no pós-MVP; no MVP, a interação será direta com a API do Serviço de Lançamentos, ex: via Swagger/Postman) para registrar, consultar, editar e excluir lançamentos.
2. A **Aplicação Web Flow Wise (UI)** acessa a API do **Serviço de Lançamentos**.
3. O **Serviço de Lançamentos** persiste os dados no **DB Lançamentos**.
4. Após a persistência bem-sucedida, o **Serviço de Lançamentos publica eventos** de domínio (ex: `LancamentoRegistradoEvent`, `LancamentoAtualizadoEvent`, `LancamentoExcluidoEvent`) no **RabbitMQ**.

Comunicação de Consolidação:

1. O **Serviço de Consolidação consome os eventos** publicados no **RabbitMQ**.
2. Ao consumir um evento, o **Serviço de Consolidação** processa o evento e atualiza seu **DB Consolidados**, mantendo a projeção do saldo diário atualizada.
3. O **Gerente Financeiro** (via a **Aplicação Web Flow Wise (UI)** no pós-MVP) ou a **Ferramenta de BI Corporativa** (diretamente via API) consultam o **Serviço de Consolidação** para obter o saldo diário ou gerar relatórios, que leem diretamente do **DB Consolidados**.

Uso do Cache (Redis):

- Ambos os serviços (Lançamentos e Consolidação) e a futura **Aplicação Web Flow Wise (UI)** podem utilizar o **Redis** para armazenar dados temporários ou cache de informações frequentemente acessadas, melhorando a performance e reduzindo a carga sobre os bancos de dados primários.

Esboço de Implantação em Cluster

Para garantir a escalabilidade, resiliência e portabilidade, os contêineres do Flow Wise serão implantados em um ambiente orquestrado por Kubernetes.

- **Conteinerização:** Cada microsserviço (Aplicação Web, Serviço de Lançamentos, Serviço de Consolidação) será empacotado como uma **imagem Docker** e executado em um **Pod Kubernetes**.
- **Replicação e Balanceamento de Carga:** Os serviços de Lançamentos e Consolidação serão configurados com múltiplos *pods* (réplicas) e serviços de *Load*

Balancer para distribuir a carga e garantir alta disponibilidade. Em caso de falha de um *pod*, o Kubernetes automaticamente provisionará um novo.

- **Armazenamento Persistente:** Os bancos de dados (DB Lançamentos, DB Consolidados) serão provisionados como serviços gerenciados na nuvem ou como *Persistent Volumes* no cluster Kubernetes, garantindo a durabilidade dos dados.
- **Componentes de Mensageria e Cache:** RabbitMQ e Redis serão implantados como serviços gerenciados na nuvem ou como *clusters* dedicados dentro do ambiente Kubernetes, com alta disponibilidade e replicação para garantir a resiliência do barramento de eventos e da camada de cache.
- **Exposição de Serviços:** Os endpoints de API dos microsserviços serão expostos através de **Ingress Controllers** ou **API Gateways** no Kubernetes, controlando o acesso externo e roteando as requisições para os *pods* corretos.
- **Preparação para Multi-Cloud:** O uso de Kubernetes e Terraform (para IaC) garante que a solução seja intrinsecamente preparada para ser implantada em diferentes provedores de nuvem (Azure, AWS, GCP) ou ambientes *on-premise* com clusters Kubernetes, sem a necessidade de grandes refatorações no código da aplicação.

Decisões Arquiteturais de Alto Nível

A arquitetura do Flow Wise será baseada em **Microsserviços**, uma escolha estratégica fundamental para atender aos objetivos de negócio e aos rigorosos Requisitos Não-Funcionais (NFRs) do projeto, mesmo em sua fase de MVP.

Esta abordagem visa não apenas a entrega do escopo inicial, mas também a **construção de uma base sólida e preparada para o futuro crescimento e evolução da solução**, alinhada com as capacidades e estratégias tecnológicas da organização.

Justificativa para a Arquitetura de Microsserviços

A adoção de microsserviços para o Flow Wise é suportada por múltiplos fatores, com foco em resiliência, escalabilidade e otimização de recursos:

- **Escalabilidade Independente e Otimização de Recursos:** Permite que componentes críticos, como o **Serviço de Consolidação** (com NFR de 50 requisições/segundo), sejam escalados horizontalmente de forma autônoma. Isso otimiza o uso de recursos de infraestrutura e garante que o sistema possa absorver picos de demanda sem impactar outras funcionalidades.
- **Resiliência e Isolamento de Falhas:** Uma falha ou indisponibilidade em um microsserviço (ex: Serviço de Consolidação) não compromete a funcionalidade de outros serviços essenciais (ex: Serviço de Lançamentos), garantindo a continuidade das operações mais críticas de registro de dados financeiros. Este isolamento é fundamental para o **NFR de resiliência**, onde o serviço de lançamentos não deve ser afetado pela indisponibilidade do consolidado.
- **Independência de Desenvolvimento e Implantação (CI/CD):** A modularidade intrínseca dos microsserviços permite que a **célula estratégica do POC** e, futuramente, equipes independentes trabalhem em serviços distintos de forma

autônoma. Isso acelera o ciclo de desenvolvimento, facilita a integração contínua (CI) e a entrega contínua (CD), otimizando o *time-to-market* para novas funcionalidades.
- **Reaproveitamento de Conhecimento e Capacitação Existente:** A escolha por **C#/.NET 8+** é estratégica e alinha-se perfeitamente com a expertise já estabelecida e **capacidade instalada da nossa equipe de engenharia na organização**. Isso minimiza a curva de aprendizado para os desenvolvedores, acelera o processo de desenvolvimento, reduz riscos e otimiza o investimento em treinamento, garantindo uma execução mais eficiente e eficaz do projeto desde o MVP.

Padrões de Arquitetura e Design

O Flow Wise será construído com uma base arquitetural robusta, incorporando os seguintes padrões para garantir qualidade, manutenibilidade e preparação para o futuro:

- **Domain-Driven Design (DDD):** Será a base para modelar os domínios de negócio de Lançamentos e Consolidados, garantindo que o software reflita a complexidade e a linguagem do negócio, facilitando a comunicação entre a célula estratégica e as áreas de negócio.
- **CQRS (Command Query Responsibility Segregation):** Essencial para a separação entre as operações de escrita (Commands, como o registro de lançamentos) e leitura (Queries, como a consulta de saldos consolidados). Isso permitirá otimizar o desempenho do lado de leitura (Serviço de Consolidação para os 50 req/s) e garantir que as operações de escrita sejam altamente disponíveis e resilientes.
- **Event Sourcing (para dados sensíveis e auditoria):** Todas as mudanças de estado de um lançamento financeiro serão armazenadas como uma sequência imutável de eventos. Isso proporciona um log de auditoria completo, crucial para conformidade financeira, e permite a reconstrução do estado em qualquer ponto no tempo. Embora a re-hidratação de agregados possa não ser um foco primário do MVP, a base será estabelecida.
- **Padrão Saga:** Para gerenciar a coordenação de transações distribuídas entre os microsserviços, garantindo a consistência eventual e a capacidade de *rollback* de operações complexas, crucial para a tolerância a falhas.
- **Correlation ID:** Implementação mandatório do Correlation ID em todos os fluxos (APIs, Mensageria, Eventos, Webhooks), garantindo rastreabilidade fim-a-fim de uma requisição. Isso será vital para monitoramento, depuração e auditoria, mesmo no MVP.
- **Princípios SOLID:** Aplicação rigorosa dos princípios de design de software (Single Responsibility, Open/Closed, Liskov Substitution, Interface Segregation, Dependency Inversion) para promover um código limpo, modular e fácil de manter e estender.

Estratégias de Resiliência e Performance

A célula estratégica garantirá que a solução seja intrinsecamente resiliente e performática:

- **Polly (Circuit Breaker, Retry, Fallback):** Utilização da biblioteca Polly para implementar padrões de resiliência como *Circuit Breaker* (para isolar falhas em

dependências externas), *Retry* (para lidar com falhas transitórias) e *Fallback* (para prover respostas alternativas em caso de falha), garantindo a robustez das interações entre serviços.
- **Redis para Cache e Resiliência:** O Redis será empregado como camada de cache para dados frequentemente acessados, melhorando a performance. Além disso, pode ser utilizado para mecanismos de *rate limiting*, controle de estados distribuídos, e como um *message broker* leve para cenários específicos, contribuindo para a resiliência geral.
- **RabbitMQ como Message Broker:** Utilização do RabbitMQ para comunicação assíncrona entre microsserviços através de mensagens e eventos. Isso garante o desacoplamento, a durabilidade das mensagens e a capacidade de processamento em segundo plano, crucial para a independência entre os serviços de Lançamentos e Consolidados.

Preparação para Cloud, Containers e Infraestrutura como Código (IaC)

O MVP, embora focado na funcionalidade, será projetado com uma visão de longo prazo para implantação em ambientes de produção modernos:

- **Containers (Docker) e Orquestração (Kubernetes):** Toda a solução será conteinerizada usando Docker. Será preparada para ser implantada em clusters Kubernetes, permitindo escalabilidade elástica e gerenciamento automatizado, seja localmente (para desenvolvimento), em *clusters on-premise*, ou em ambientes *multi-cloud*.
- **Preparação para Multi-Cloud:** Embora não haja um foco imediato na implantação multi-cloud para o MVP, a arquitetura e as diretrizes de IaC (Infraestrutura como Código) com Terraform serão estabelecidas para permitir essa flexibilidade no futuro. Isso incluirá a colaboração próxima com o **Time de Infraestrutura** para desenvolver scripts Terraform que suportem essa capacidade, sem acoplar a solução a um único provedor de nuvem.
- **Swagger (OpenAPI):** Será utilizado internamente para documentar e expor os endpoints das APIs dos microsserviços, facilitando o desenvolvimento e a integração interna entre os serviços da célula estratégica e, futuramente, outros times.

Estratégia de Segurança e Observabilidade (Preparação para Futuro)

Reconhecendo a criticidade desses pilares, o MVP estabelecerá as bases, mas com priorização adequada para a fase inicial:

- **Segurança:**
    - **Foco no Secure by Design:** As práticas de desenvolvimento seguro serão aplicadas desde o início para mitigar vulnerabilidades comuns (OWASP Top 10).
    - **Integração (Pós-MVP):** A integração com o **Serviço de Autenticação Corporativo (SSO)** via Okta/Azure AD e a implementação de **autorização granular (RBAC)** serão prioridades após o MVP.
    - **Parceria com o Time de Segurança:** O **time de Blue e Red Team** da organização terá um papel fundamental na análise e aprovação das decisões

de segurança. No MVP, o foco será na preparação, e uma estratégia de **Bug Bounty** será considerada para futuras fases, potencialmente com parceiros externos como a **Tempest Security**, para uma validação de segurança robusta (pentests caixa preta/branca).

- **Observabilidade:**
    - **Preparação para Plug-and-Play:** O MVP será projetado para ser **'observability-ready'**, com abstrações e padrões que permitam a fácil integração com diferentes ferramentas de monitoramento (ex: Datadog, Elastic, Dynatrace). Não haverá foco na implementação completa de uma ferramenta específica no MVP, mas a solução estará preparada para que, quando o **Time de Observabilidade** decidir qual ferramenta usar, seja um processo de 'plug-and-play' sem impacto no código do negócio. Isso inclui o registro de logs estruturados e a instrumentação para métricas e tracing distribuído.

Gerenciamento de Contratos e Evolução (Pós-MVP)

Com a visão de evolução contínua, o Flow Wise será preparado para lidar com mudanças de forma controlada:

- **Versionamento de Contratos de API/Mensagens:** Será definida uma estratégia clara para o versionamento de APIs e modelos de mensagens/eventos, garantindo a compatibilidade retroativa e a coexistência de diferentes versões durante transições de deploy.
- **Estratégias de Deployment (Canary Releases):** Para garantir a segurança nas implantações, o projeto considerará estratégias como *Canary Releases* (liberação gradual para um pequeno grupo de usuários) para validar novas versões em produção antes de um rollout completo, minimizando o risco de impacto em produção e garantindo que novas versões não quebrem a que está em execução.

Documentação e Governança (Pilar Fundamental)

A documentação será um artefato vivo e central para o sucesso do projeto:

- **Catálogo de Informações de Domínio:** Criação e manutenção de um catálogo detalhado de dados, fluxos, serviços, tipos de eventos e mensagens, garantindo uma compreensão clara e unificada do domínio.
- **Documentação Centralizada e Acessível:** Todos os documentos técnicos e de negócio (incluindo padrões organizacionais) serão mantidos de forma centralizada (Google Docs, GitHub) e facilmente acessíveis a todos os times.
- **Treinamentos e Comunicação:** Serão planejados treinamentos para as equipes para garantir que todos sigam os padrões definidos e haja uma comunicação forte e constante sobre o projeto e suas diretrizes.

Trade-offs e Desafios

- **Complexidade Operacional:** Microsserviços introduzem maior complexidade em termos de monitoramento, deployment e depuração. Isso será mitigado com ferramentas robustas de DevOps (CI/CD, observabilidade) e automação.
- **Gerenciamento de Dados Distribuídos:** A consistência de dados entre diferentes microsserviços requer padrões como Eventual Consistency e Sagas, que aumentam a complexidade de design.
- **Comunicação entre Serviços:** A comunicação (síncrona via HTTP/gRPC e assíncrona via *message brokers*) precisa ser bem gerenciada para evitar latência e problemas de acoplamento.

Requisitos de Negócio e Funcionais

Esta seção detalha o que o sistema Flow Wise deve fazer para atender às necessidades de negócio, descrevendo as funcionalidades através de Épicos, Histórias de Usuário e Regras de Negócio.

Os requisitos aqui definidos guiarão o desenvolvimento do MVP e servirão como base para o planejamento das futuras iterações.

Visão Geral dos Requisitos Funcionais

Os requisitos funcionais especificam as capacidades que o sistema Flow Wise deve possuir para permitir aos usuários realizar tarefas e alcançar os objetivos de negócio.

Eles são organizados em Épicos, que representam grandes blocos de funcionalidades, detalhados em Histórias de Usuário, que descrevem funcionalidades específicas da perspectiva do usuário, e complementados por Regras de Negócio que governam o comportamento do sistema.

Épicos e Histórias de Usuário

Os épicos e suas respectivas histórias de usuário fornecem uma estrutura hierárquica para o backlog do Flow Wise, focando nas entregas de valor para o usuário final.

Épico: Gestão de Lançamentos Financeiros

Este épico abrange todas as funcionalidades relacionadas à criação, consulta, edição e exclusão de lançamentos de débito e crédito no sistema Flow Wise e será a base para o controle diário das movimentações financeiras.

Histórias de Usuário:

HU-001: Registrar Lançamento de Débito Manualmente

- **Como** um **Analista Financeiro**,
- **Eu quero** registrar um lançamento de débito (saída de caixa) com seus detalhes,
- **Para que** eu possa ter controle preciso sobre as despesas e saídas de capital da organização.
- **Critérios de Aceitação:**
    - O sistema deve apresentar um formulário intuitivo para o registro de novos lançamentos de débito.
    - O usuário deve conseguir informar os seguintes dados obrigatórios:
        - **Valor:** Numérico (positivo), com duas casas decimais.
        - **Data do Lançamento:** Data (formato DD/MM/AAAA). Deve ser igual ou anterior ao dia atual (D-0). Não pode ser data futura.
        - **Descrição:** Texto livre, com no mínimo 5 e no máximo 255 caracteres.

        - **Categoria:** Seleção de uma lista pré-definida de categorias de despesa (ex: 'Aluguel', 'Salários', 'Fornecedores', 'Impostos', 'Outros').
    - O usuário deve poder informar um campo opcional:
        - **Observações:** Texto livre, máximo 500 caracteres, para detalhes adicionais.
    - Após o preenchimento e submissão, o sistema deve validar todos os campos.
    - Em caso de sucesso, o sistema deve salvar o lançamento e exibir uma mensagem de confirmação.
    - Em caso de falha de validação, o sistema deve apresentar mensagens de erro claras para cada campo inválido.
    - O lançamento registrado deve ser imediatamente visível na tela de consulta de lançamentos do dia correspondente.

HU-002: Registrar Lançamento de Crédito Manualmente

- **Como** um **Analista Financeiro**,
- **Eu quero** registrar um lançamento de crédito (entrada de caixa) com seus detalhes,
- **Para que** eu possa ter controle preciso sobre as receitas e entradas de capital da organização.
- **Critérios de Aceitação:**
    - O sistema deve apresentar um formulário intuitivo para o registro de novos lançamentos de crédito.
    - Os campos e validações devem ser os mesmos do registro de débito, mas o Tipo de Lançamento será 'Crédito'.
    - A categoria deve ser selecionada de uma lista pré-definida de categorias de receita (ex: 'Vendas de Produtos', 'Vendas de Serviços', 'Juros Recebidos', 'Outras Receitas').
    - Após o preenchimento e submissão, o sistema deve validar e salvar o lançamento, exibindo confirmação ou erros.
    - O lançamento registrado deve ser imediatamente visível na tela de consulta de lançamentos do dia correspondente.

HU-003: Consultar Lançamentos por Data

- **Como** um **Analista Financeiro** ou **Gerente Financeiro**,
- **Eu quero** consultar todos os lançamentos (débito e crédito) de um dia específico,
- **Para que** eu possa ter uma visão detalhada das movimentações financeiras daquele período.
- **Critérios de Aceitação:**
    - O sistema deve permitir a seleção de uma data para consulta.
    - O sistema deve exibir uma lista paginada e ordenada cronologicamente de todos os lançamentos (débito e crédito) para a data selecionada.
    - Para cada lançamento, devem ser exibidos: Valor, Tipo (Débito/Crédito), Data do Lançamento, Descrição, Categoria e Observações.
    - Deve haver uma funcionalidade de busca/filtro por descrição ou categoria.
    - A consulta deve ser performática (tempo de resposta conforme NFRs de performance).

HU-004: Editar Lançamento Existente

- **Como** um **Analista Financeiro**,
- **Eu quero** editar um lançamento financeiro já registrado,
- **Para que** eu possa corrigir eventuais erros ou atualizar informações.
- **Critérios de Aceitação:**
    - O sistema deve permitir que um lançamento seja editado apenas se a data do lançamento for igual ao dia atual (D-0). Lançamentos de dias anteriores não podem ser editados. (Esta regra será detalhada na seção de Regras de Negócio).
    - Ao selecionar um lançamento para edição, o sistema deve pré-preencher o formulário com os dados existentes.
    - Todos os campos do lançamento, exceto o **Tipo de Lançamento** (débito/crédito), devem ser editáveis.
    - As mesmas validações de campos obrigatórios e formatos aplicadas ao registro devem ser aplicadas na edição.
    - Após a edição, o sistema deve registrar a alteração (quem alterou, quando e quais campos foram alterados) para fins de auditoria.
    - O sistema deve exibir uma mensagem de sucesso ou erro após a tentativa de edição.

HU-005: Excluir Lançamento Existente

- **Como** um **Analista Financeiro**,
- **Eu quero** excluir um lançamento financeiro registrado por engano ou duplicidade,
- **Para que** os dados do fluxo de caixa permaneçam precisos.
- **Critérios de Aceitação:**
    - O sistema deve permitir que um lançamento seja excluído apenas se a data do lançamento for igual ao dia atual (D-0). Lançamentos de dias anteriores não podem ser excluídos. (Esta regra será detalhada na seção de Regras de Negócio).
    - Antes da exclusão, o sistema deve solicitar uma confirmação ao usuário para evitar exclusões acidentais.
    - Após a exclusão, o sistema deve registrar a operação (quem excluiu, quando e qual lançamento foi excluído) para fins de auditoria.
    - O lançamento excluído não deve mais aparecer nas consultas de lançamentos.
    - O sistema deve exibir uma mensagem de sucesso ou erro após a tentativa de exclusão.

Épico: Consolidado e Relatórios Financeiros

Este épico trata da consolidação dos lançamentos diários e da apresentação do saldo de caixa consolidado, fornecendo uma visão agregada para análise gerencial.

Histórias de Usuário:

HU-006: Visualizar Saldo Diário Consolidado (D-1)

- **Como** um **Gerente Financeiro** ou **Analista Financeiro**,
- **Eu quero** visualizar o saldo consolidado de caixa para um dia específico,
- **Para que** eu possa acompanhar a saúde financeira diária da organização e tomar decisões rápidas.
- **Critérios de Aceitação:**
    - O sistema deve permitir a seleção de uma data para consulta do saldo consolidado.
    - O saldo exibido deve ser calculado com base em **todos os lançamentos finalizados até o final do dia anterior (D-1)** à data selecionada.
    - O sistema deve exibir o **Saldo Total** (Créditos Totais - Débitos Totais) para a data consolidada.
    - A consulta deve ser performática e atender ao NFR de 50 requisições/segundo.
    - Os dados do saldo devem ser consistentes com os lançamentos auditados.

HU-007: Gerar Relatório de Fluxo de Caixa por Período

- **Como** um **Gerente Financeiro**,
- **Eu quero** gerar um relatório consolidado de fluxo de caixa para um intervalo de datas,
- **Para que** eu possa analisar tendências de longo prazo e apoiar o planejamento financeiro estratégico.
- **Critérios de Aceitação:**
    - O sistema deve permitir a seleção de uma data de início e uma data de fim para o relatório (máximo 30 dias de período inicial).
    - O relatório deve apresentar: Saldo Inicial do período, Total de Débitos no período, Total de Créditos no período, e Saldo Final do período.
    - O relatório deve ser gerado em um formato visualmente claro e de fácil compreensão.
    - Deve haver uma opção para exportar o relatório para um formato comum (ex: CSV ou PDF).
    - A geração do relatório deve ser performática (tempo de resposta conforme NFRs de performance).

Regras de Negócio

As regras de negócio definem o comportamento específico e as restrições que o sistema Flow Wise deve aplicar para garantir a integridade e a conformidade das operações financeiras.

- **RN-001: Janela de Edição/Exclusão de Lançamentos:**
    - Um lançamento financeiro pode ser **editado** ou **excluído** apenas se a **data do lançamento for igual ao dia atual (D-0)**. Lançamentos de dias anteriores não podem ser modificados para garantir a integridade histórica dos dados após o fechamento do dia.

- **RN-002: Validação de Data de Lançamento:**
    - A data de um lançamento (débito ou crédito) deve ser igual ou anterior ao dia atual (D-0). Lançamentos com data futura não são permitidos.
- **RN-003: Cálculo do Saldo Consolidado Diário:**
    - O saldo diário consolidado para uma data X é calculado com base na soma de todos os lançamentos (créditos menos débitos) que foram processados e finalizados até o final do dia X−1 (dia anterior). A consolidação é um processo noturno/batch.
- **RN-004: Obrigatoriedade de Categoria:**
    - Todo lançamento (débito ou crédito) deve ser associado a uma categoria válida, selecionada de uma lista predefinida no sistema.
- **RN-005: Imutabilidade do Tipo de Lançamento:**
    - Uma vez registrado, o tipo de um lançamento (Débito ou Crédito) não pode ser alterado. Em caso de erro, o lançamento deve ser excluído e um novo lançado.
- **RN-006: Validação de Valor:**
    - O valor de um lançamento deve ser um número positivo e maior que zero. Não são permitidos lançamentos com valor zero ou negativo.
- **RN-007: Auditoria de Operações:**
    - Todas as operações de criação, edição e exclusão de lançamentos devem ser logadas, registrando o usuário responsável, a data/hora da operação e os detalhes da alteração (se aplicável), para fins de rastreabilidade e conformidade.

Requisitos Não-Funcionais (NFRs)

Esta seção define os critérios de qualidade e as restrições operacionais que o sistema Flow Wise deve atender, essenciais para sua robustez, segurança, usabilidade e capacidade de evolução.

Os NFRs são cruciais para o sucesso do projeto e para garantir que a solução atenda às expectativas de desempenho e disponibilidade da organização.

Visão Geral dos NFRs

Os Requisitos Não-Funcionais (NFRs) especificam como o sistema Flow Wise deve se comportar, focando em atributos de qualidade como desempenho, segurança, disponibilidade, resiliência, escalabilidade e manutenibilidade.

A abordagem de Microsserviços e a seleção tecnológica foram estrategicamente definidas para atender a estes requisitos desde a fase de MVP.

Desempenho (Performance)

Critérios relacionados à velocidade e eficiência do sistema, garantindo uma experiência de usuário fluida e processamento de dados eficaz.

NFR-PERF-001: Latência de Registro de Lançamentos

- **Requisito:** O tempo de resposta para o registro de um novo lançamento (débito ou crédito) não deve exceder **500 milissegundos (ms)** para **95%** das requisições sob carga normal.
- **Métrica:** Tempo médio de resposta da API de registro de lançamentos.

NFR-PERF-002: Vazão do Serviço de Consolidação Diária

- **Requisito:** O serviço de consolidação diária deve suportar **50 requisições por segundo (req/s)**, com uma taxa de sucesso de no mínimo **95%** (ou seja, no máximo 5% de perda ou erros) em dias de pico.
- **Métrica:** Requisições por segundo e taxa de erro do endpoint de consulta do saldo consolidado.

NFR-PERF-003: Latência de Consulta de Lançamentos

- **Requisito:** A consulta de lançamentos diários deve retornar os resultados em no máximo **1 segundo** para **99%** das requisições.
- **Métrica:** Tempo médio de resposta da API de consulta de lançamentos.

NFR-PERF-004: Tempo de Geração de Relatório de Fluxo de Caixa

- **Requisito:** A geração de relatório de fluxo de caixa por período (até 30 dias) deve ser concluída em no máximo **5 segundos**.
- **Métrica:** Tempo de processamento e entrega do relatório de fluxo de caixa.

Disponibilidade

Percentual de tempo em que o sistema e seus serviços críticos devem estar operacionais e acessíveis.

NFR-DISP-001: Disponibilidade do Serviço de Lançamentos

- **Requisito:** O microsserviço de gestão de lançamentos deve atingir uma disponibilidade de **99.99%** em ambiente de produção (equivalente a no máximo ~5 minutos de inatividade anual).
- **Métrica:** Uptime do microsserviço de lançamentos.

NFR-DISP-002: Disponibilidade do Serviço de Consolidação Diária

- **Requisito:** O microsserviço de consolidação e relatórios deve ter uma disponibilidade de **99.9%** em ambiente de produção (equivalente a no máximo ~8 horas e 45 minutos de inatividade anual).
- **Métrica:** Uptime do microsserviço de consolidação.

Resiliência e Tolerância a Falhas

Capacidade do sistema de lidar e se recuperar de falhas, minimizando o impacto nas operações e garantindo a continuidade do negócio. Esta é uma área de foco crucial para o Flow Wise.

NFR-RES-001: Isolamento de Falhas Críticas

- **Requisito:** A indisponibilidade ou lentidão do microsserviço de consolidação diária *não deve afetar* a capacidade de registro de novos lançamentos no microsserviço de lançamentos.
- **Métrica:** Sucesso nas operações de registro de lançamentos durante cenários de falha do serviço de consolidação.

NFR-RES-002: Recuperação Automática de Falhas

- **Requisito:** Em caso de falha de um componente ou serviço, o sistema deve ser capaz de se recuperar automaticamente (ex: *failover*, reinício de instâncias) em no máximo **5 minutos**, sem intervenção manual, para serviços críticos.
- **Métrica:** Recovery Time Objective (RTO) para serviços críticos.

NFR-RES-003: Tolerância a Falhas Transitórias

- **Requisito:** Mecanismos de *retry* (com *backoff* exponencial) e *circuit breaker* devem ser implementados para interações entre microsserviços e com dependências externas, minimizando o impacto de falhas transitórias e prevenindo a sobrecarga de serviços.
- **Métrica:** Redução da propagação de falhas e sucesso nas retentativas.

NFR-RES-004: Consistência de Dados em Transações Distribuídas

- **Requisito:** Operações que envolvem múltiplos microsserviços devem garantir consistência eventual dos dados, com mecanismos para gerenciar e compensar falhas de transações distribuídas (Padrão Saga).
- **Métrica:** Integridade dos dados após transações distribuídas, capacidade de *rollback* ou compensação.

Escalabilidade

Capacidade de aumentar a capacidade do sistema para lidar com o crescimento da demanda e do volume de dados.

NFR-ESC-001: Escalabilidade Horizontal dos Microsserviços

- **Requisito:** Todos os microsserviços da solução Flow Wise devem ser projetados para permitir escalabilidade horizontal, ou seja, a capacidade de aumentar a capacidade de processamento através da adição de novas instâncias.
- **Métrica:** Capacidade de processamento por instância adicionada.

NFR-ESC-002: Preparação para Elasticidade (Auto-scaling)

- **Requisito:** A arquitetura da solução deve estar preparada para a implementação de *auto-scaling*, permitindo que os recursos sejam ajustados dinamicamente em resposta a picos de demanda.
- **Métrica:** Capacidade do sistema de escalar automaticamente em testes de carga.

NFR-ESC-003: Suporte a Crescimento de Volume de Dados

- **Requisito:** O sistema Flow Wise deve ser capaz de gerenciar e processar um volume inicial de **50.000 lançamentos diários**, com capacidade de escalabilidade para **150.000 lançamentos diários** em um período de **3 anos**. Adicionalmente, o sistema deve ser capaz de consolidar e disponibilizar **30 relatórios de saldo diário** por mês para até **500 usuários ativos simultaneamente** no serviço de consolidação.
- **Métrica:** Capacidade de armazenamento de dados (volume e *throughput*) e desempenho do processamento de lançamentos e consolidação sob carga crescente.

Segurança

Medidas para proteger o sistema contra acessos não autorizados, vazamento de dados, ataques e garantir a integridade das informações financeiras.

NFR-SEG-001: Autenticação Centralizada (Pós-MVP)

- **Requisito:** No pós-MVP, todos os acessos ao sistema Flow Wise devem ser autenticados através do sistema de Single Sign-On (SSO) corporativo (ex: Okta/Azure AD), garantindo uma gestão de identidade centralizada e robusta.
- **Métrica:** Integração bem-sucedida com o provedor de identidade.

NFR-SEG-002: Autorização Baseada em Papeis (RBAC)

- **Requisito:** O sistema deve implementar controle de acesso baseado em papeis (RBAC), garantindo que apenas usuários com as permissões adequadas possam realizar operações específicas (ex: apenas Analistas Financeiros podem registrar lançamentos, Gerentes Financeiros podem acessar relatórios gerenciais).
- **Métrica:** Auditoria de acessos e teste de permissões.

NFR-SEG-003: Criptografia de Dados (Em Trânsito e Em Repouso)

- **Requisito:** Todos os dados sensíveis (ex: valores financeiros, informações de usuário) devem ser criptografados em trânsito (utilizando TLS 1.2+ para APIs e comunicação entre serviços) e em repouso (criptografia de banco de dados e armazenamento).
- **Métrica:** Conformidade com padrões de criptografia.

NFR-SEG-004: Proteção contra Vulnerabilidades Comuns (OWASP Top 10)

- **Requisito:** O desenvolvimento deve seguir as melhores práticas de segurança de código para mitigar os riscos do OWASP Top 10 (ex: injeção de SQL, XSS, quebra de controle de acesso), aplicando o conceito de *Secure by Design* desde o MVP.
- **Métrica:** Zero vulnerabilidades críticas/altas detectadas em análises SAST/DAST e revisões de código.

NFR-SEG-005: Auditoria Completa de Operações

- **Requisito:** Todas as operações críticas (criação, edição, exclusão de lançamentos, acessos privilegiados) devem ser auditáveis, registrando o usuário, data/hora e detalhes da ação. Os logs de auditoria devem ser imutáveis.
- **Métrica:** Rastreabilidade completa das operações financeiras.

NFR-SEG-006: Estratégia de Pentest e Bug Bounty (Pós-MVP)

- **Requisito:** Embora não seja foco do MVP, a célula estratégica trabalhará em conjunto com o **Time de Blue e Red Team** da organização para definir uma estratégia robusta de testes de segurança, incluindo *pentests* (caixa-preta, caixa-branca), e considerar a implementação de um programa de *Bug Bounty* em fases posteriores, possivelmente com parceiros especializados como a **Tempest Security**.
- **Métrica:** Plano de testes de segurança formalizado.

Manutenibilidade

Facilidade com que o sistema pode ser modificado, evoluído e corrigido ao longo do tempo, garantindo sua longevidade.

NFR-MAN-001: Modularidade e Baixo Acoplamento

- **Requisito:** O sistema deve ser construído com alta modularidade e baixo acoplamento entre os microsserviços e seus componentes internos, facilitando a substituição e evolução de partes da solução.
- **Métrica:** Baixa dependência entre módulos e serviços.

NFR-MAN-002: Testabilidade e Cobertura de Testes

- **Requisito:** O código-fonte deve ser amplamente testável, com cobertura robusta de testes unitários (mínimo de 60% para lógica de negócio crítica) e testes de integração, garantindo a qualidade e a segurança das alterações.
- **Métrica:** Percentual de cobertura de código por testes automatizados.

NFR-MAN-003: Documentação Abrangente e Atualizada

- **Requisito:** A documentação técnica (arquitetural, de design, APIs via Swagger/OpenAPI) e de negócio deve ser mantida atualizada e ser facilmente acessível, refletindo o estado atual da solução.
- **Métrica:** Disponibilidade e qualidade da documentação.

NFR-MAN-004: Versionamento de Contratos

- **Requisito:** Será definida uma estratégia clara para o versionamento de APIs e modelos de mensagens/eventos, garantindo compatibilidade retroativa e a coexistência de diferentes versões para facilitar a evolução sem quebras.
- **Métrica:** Conformidade com a política de versionamento.

Observabilidade

Capacidade de monitorar, coletar métricas e rastrear o comportamento do sistema em produção, essencial para identificar e diagnosticar problemas rapidamente.

NFR-OBS-001: Preparação para 'Plug-and-Play' de Ferramentas de Observabilidade

- **Requisito:** O MVP será projetado para ser **'observability-ready'**, com abstrações e padrões que permitam a fácil integração com diferentes ferramentas de monitoramento corporativas (ex: Datadog, Elastic, Dynatrace), sem impacto no código de negócio. A solução estará preparada para que a ferramenta seja 'plugada' quando o **Time de Observabilidade** decidir qual provedor utilizar.
- **Métrica:** Facilidade de integração com diferentes ferramentas (evidência de design).

NFR-OBS-002: Logging Estruturado e Centralizado

- **Requisito:** Todos os logs de aplicação devem ser estruturados (JSON/key-value) e centralizados em uma plataforma para fácil coleta, análise e busca, facilitando a depuração e o monitoramento proativo.
- **Métrica:** Formato e disponibilidade dos logs.

NFR-OBS-003: Exposição de Métricas de Negócio e Técnicas

- **Requisito:** O sistema deve expor métricas de desempenho (ex: tempo de resposta, erros) e métricas de negócio (ex: quantidade de lançamentos processados, saldo consolidado por dia) para monitoramento em tempo real através de dashboards.
- **Métrica:** Disponibilidade e granularidade das métricas.

NFR-OBS-004: Tracing Distribuído

- **Requisito:** A capacidade de rastrear requisições através de múltiplos microsserviços (tracing distribuído, utilizando o Correlation ID) deve ser implementada, permitindo a visibilidade completa do fluxo de uma transação.
- **Métrica:** Capacidade de rastrear fluxos de requisições.

Padrões Organizacionais e Diretrizes

Esta seção estabelece os padrões, diretrizes e boas práticas que guiarão o desenvolvimento, a operação e a manutenção do Projeto Flow Wise.

A adesão a esses padrões é fundamental para garantir a consistência, a qualidade, a segurança e a manutenibilidade da solução, alinhando-se às políticas e estratégias tecnológicas da organização.

Visão Geral dos Padrões

O Projeto Flow Wise será desenvolvido e mantido com base em um conjunto rigoroso de padrões e diretrizes organizacionais.

Isso assegura não apenas a excelência técnica, mas também a facilidade de integração com o ecossistema de TI existente e a capacidade de evolução sustentável. A governança será um pilar central, garantindo que as decisões sejam documentadas e que os times sigam as melhores práticas estabelecidas.

Ciclo de Vida do Desenvolvimento de Software (SDLC) e Metodologia Ágil

O desenvolvimento do Flow Wise seguirá uma abordagem ágil, com foco na entrega incremental de valor e na adaptação contínua.

Metodologia de Trabalho

- Será adotada uma metodologia ágil híbrida (Scrum/Kanban), com ciclos curtos de desenvolvimento (sprints) e entregas frequentes.
- O gerenciamento de tarefas e o progresso do projeto serão realizados utilizando a ferramenta Azure Boards ou Jira, garantindo transparência e visibilidade para todos os *stakeholders*.

Ambientes de Desenvolvimento e Implantação

- Serão estabelecidos ambientes dedicados para Desenvolvimento (Dev), Qualidade (QA), Homologação (UAT) e Produção (Prod), garantindo um fluxo de trabalho seguro e controlado.
- A promoção de código entre ambientes será automatizada via pipelines de CI/CD.

Padrões de Código e Qualidade de Software

A qualidade do código-fonte é prioritária para a manutenibilidade e escalabilidade do Flow Wise.

Boas Práticas de Codificação C#/.NET 8+

- **Princípios SOLID:** Aplicação mandatória dos princípios de Responsabilidade Única, Aberto/Fechado, Substituição de Liskov, Segregação de Interface e Inversão de Dependência para um design modular e testável.
- **Domain-Driven Design (DDD):** Modelagem do domínio de negócio com Agregados, Entidades e Value Objects, garantindo que o código reflita a complexidade do negócio.
- **CQRS e Event Sourcing:** Implementação rigorosa desses padrões para desacoplamento de responsabilidades e garantia de auditabilidade e resiliência, conforme detalhado na arquitetura.
- **Injeção de Dependência (DI):** Uso extensivo de DI para promover o baixo acoplamento e facilitar a testabilidade.
- **Tratamento de Exceções:** Implementação de uma estratégia centralizada e padronizada para tratamento de exceções, com registro adequado em logs.
- **Convenções de Nomenclatura:** Aderência às convenções de nomenclatura C# (PascalCase para tipos e membros públicos, camelCase para parâmetros e variáveis locais).
- **Uso de LINQ:** Preferência pelo uso de LINQ para operações de consulta, promovendo código mais legível e conciso.

Qualidade de Código e Revisão

- **Análise Estática de Código:** Utilização de ferramentas de análise estática (ex: SonarQube, Roslyn Analyzers) integradas ao pipeline de CI para identificar e corrigir débitos técnicos e vulnerabilidades.
- **Métricas de Cobertura de Testes:** Definição de um limiar mínimo de **60% de cobertura de código por testes unitários** para a lógica de negócio crítica, garantindo a validação da funcionalidade e a segurança em refatorações.
- **Revisão de Código (Pull Requests):** Todo código deve passar por revisão por pares através de Pull Requests (PRs) no GitHub, garantindo a qualidade, aderência aos padrões e compartilhamento de conhecimento.

Padrões de Teste e Qualidade Assegurada (QA)

A estratégia de testes visa garantir a qualidade, a funcionalidade correta e o atendimento aos NFRs do Flow Wise.

Estratégia de Testes Automatizados

- **Testes Unitários:** Foco em testes granulares para a lógica de negócio individual de classes e métodos.
- **Testes de Integração:** Verificação da comunicação correta entre os microsserviços e suas dependências (bancos de dados, *message brokers*, APIs de terceiros).
- **Testes de Aceitação (End-to-End):** Validação de fluxos de usuário completos, simulando cenários reais de uso do sistema.

- **Testes de Performance e Carga:** Realização de testes de carga regulares para validar o atendimento aos NFRs de performance, como o de 50 requisições/segundo para o serviço de consolidação.
- **Testes de Resiliência:** Simulação de falhas (ex: latência, indisponibilidade de dependências) para validar a eficácia dos *circuit breakers*, *retries* e mecanismos de *fallback*.

Qualidade Assegurada Contínua

- Integração de testes automatizados ao pipeline de CI/CD, permitindo antecipar novas falhas com as alterações.
- Criação de massa de dados de teste representativa e gerenciamento adequado dos dados de teste.

Padrões de Segurança no Desenvolvimento (Secure SDLC)

A segurança é um pilar fundamental do Flow Wise e será incorporada em todas as fases do SDLC (Secure by Design).

Diretrizes de Desenvolvimento Seguro

- **Mitigação do OWASP Top 10:** Aplicação de práticas de codificação segura para prevenir as vulnerabilidades mais comuns (injeção, quebra de controle de acesso, etc.).
- **Validação de Entradas:** Rigorosa validação de todas as entradas de usuário e dados recebidos de sistemas externos para prevenir ataques.
- **Gerenciamento de Segredos (Secrets Management):** Credenciais e chaves sensíveis não serão hardcoded, mas gerenciadas através de soluções de *secrets management* (ex: Azure Key Vault, AWS Secrets Manager, HashiCorp Vault), com acesso restrito e auditável.
- **Criptografia:** Garantia da criptografia de dados em trânsito (TLS 1.2+) e em repouso (criptografia de banco de dados/armazenamento).

Ferramentas e Processos de Segurança

- **Análise de Segurança de Código (SAST/DAST):** Integração de ferramentas de SAST (Static Application Security Testing) e DAST (Dynamic Application Security Testing) aos pipelines de CI/CD para detecção proativa de vulnerabilidades.
- **Análise de Composição de Software (SCA):** Verificação de dependências de terceiros para vulnerabilidades conhecidas (ex: ferramentas como Snyk ou WhiteSource).
- **Treinamento Contínuo:** Capacitação e conscientização contínua dos desenvolvedores sobre as melhores práticas de segurança.
- **Colaboração com Times de Segurança:** Colaboração próxima com o **Time de Blue e Red Team** da organização para revisão de arquitetura de segurança,

validação de controles e definição de estratégias avançadas de pentest e *bug bounty* (pós-MVP).

Padrões de Documentação e Conhecimento

A documentação é um artefato vivo e central para o sucesso do Flow Wise, garantindo que o conhecimento seja compartilhado e acessível a todos.

Abordagem C4 Model para Diagramas Arquiteturais

- Utilização padronizada do **C4 Model** para representação da arquitetura em múltiplos níveis de abstração:
    - **Contexto (Nível 1):** Onde o sistema se encaixa no panorama geral da organização.
    - **Contêineres (Nível 2):** Os principais componentes executáveis (microsserviços, bancos de dados, message brokers) do sistema.
    - **Componentes (Nível 3):** Detalhes internos de um contêiner (lógica de negócio, interfaces).
    - **Código (Nível 4):** Diagramas de classes ou fluxo de código específicos (quando necessário).
- **Ferramenta:** Todos os diagramas serão criados e mantidos no **Draw.io**, com os arquivos-fonte (.xml) versionados no repositório GitHub.

Documentação de Requisitos e Decisões

- **Requisitos Funcionais e Não-Funcionais:** Detalhamento completo e atualização contínua no Google Docs, conforme esta estrutura.
- **Decisões Arquiteturais (ADRs - Architectural Decision Records):** Registro formal de decisões arquiteturais importantes no repositório GitHub (`/docs/02-arquitetura-software/decisoes-arquiteturais-principais.md`), incluindo o contexto, a decisão, as alternativas consideradas, os *trade-offs* e suas consequências.

Documentação de Código e Repositórios

- **READMEs:** Cada repositório de microsserviço no GitHub (`/src/`) conterá um `README.md` abrangente, incluindo visão geral, como configurar e rodar o projeto localmente, estrutura do código, dependências e links para documentação mais detalhada.
- **Swagger/OpenAPI:** Será utilizado para documentar automaticamente as APIs RESTful dos microsserviços, facilitando o consumo e a integração entre os serviços internos e futuros consumidores.

Catálogo de Informações de Domínio

- Criação e manutenção de um documento ou seção dedicada para detalhar o domínio de negócio, incluindo glossário de termos, modelos de dados, fluxos de eventos, tipos de mensagens e contratos de comunicação, garantindo uma compreensão unificada do negócio e da solução.

Controle de Versão e Colaboração (Git/GitHub)

O GitHub será a plataforma central para o controle de versão do código-fonte e a colaboração da equipe.

Fluxo de Trabalho (Workflow)

- Será adotado o GitHub Flow ou Git Flow simplificado, com branches de *feature* para o desenvolvimento de novas funcionalidades e *branches* de *release* para lançamentos, garantindo um processo de desenvolvimento e implantação claro.
- A branch *main* será sempre mantida em um estado "implantável".

Processo de Pull Request (PR)

- Todos os PRs exigirão pelo menos um revisor aprovador, além de *checks* de CI/CD (testes automatizados, análise estática de código) passando antes do *merge*.
- As mensagens de commit e os títulos dos PRs devem ser claros e seguir um padrão predefinido para facilitar a rastreabilidade.

Diretrizes de Contribuição

Um documento *`CONTRIBUTING.md`* será disponibilizado no repositório do GitHub, detalhando como outros times e desenvolvedores podem contribuir com o projeto. Este documento incluirá:

- **Setup do Ambiente:** Instruções para configurar o ambiente de desenvolvimento local.
- **Execução de Testes:** Guia para rodar os testes automatizados.
- **Padrões de Commits Semânticos:** Diretrizes claras para a criação de mensagens de commit para garantir a clareza do histórico de versão e a automação de *release notes*.
- **Diretrizes para Abertura de Pull Requests (PRs):** Processo de criação e revisão de PRs, critérios de merge e responsabilidades.

Diretrizes de DevOps e CI/CD

A automação é fundamental para a entrega contínua de valor e a operação eficiente do Flow Wise.

Pipelines de Integração e Entrega Contínua (CI/CD)

Serão implementados pipelines de CI/CD automatizados (ex: utilizando GitHub Actions, Azure DevOps Pipelines) para cada microsserviço, que incluirão:

- Build do código.
- Execução de testes unitários e de integração.
- Análise estática de código e segurança.
- Geração de artefatos (imagens Docker).
- Deployment automatizado para ambientes de desenvolvimento e QA.

Estratégias de Deployment: Para produção, o projeto se preparará para estratégias de *deployment* avançadas como *Canary Releases* (liberação gradual para um pequeno grupo de usuários) ou *Blue/Green Deployments*, para garantir a segurança nas implantações, minimizar o risco de impacto em produção e garantir que novas versões não quebrem a que está em execução. Esta será a abordagem para **garantir que testes de regressão funcionem e impedir a quebra do projeto com novas mudanças ou novos *deploys***.

Infraestrutura como Código (IaC) e Multi-Cloud Ready

- Toda a infraestrutura e recursos de nuvem (ex: Kubernetes, RabbitMQ, Redis, bancos de dados) serão gerenciados via **Terraform**. Os scripts Terraform serão preparados para serem **multi-cloud ready**, permitindo a implantação em diferentes provedores de nuvem no futuro, caso seja uma decisão estratégica.
- O **Time de Infraestrutura** terá um papel fundamental no apoio à criação e manutenção desses scripts IaC.
- **Conteinerização:** A solução será totalmente conteinerizada com Docker, permitindo a execução consistente em ambientes locais (para desenvolvimento), *clusters* Kubernetes *on-premise* ou em qualquer provedor de nuvem.

Monitoramento e Observabilidade

- Conforme NFRs de Observabilidade, o MVP já incorporará a instrumentação necessária para logs estruturados, métricas e *tracing* distribuído (via *Correlation ID*), permitindo uma integração plug-and-play com ferramentas de observabilidade corporativas (Datadog, Elastic, Dynatrace) em fases futuras. O foco será na preparação, não na implementação completa no MVP.
- Será feita uma documentação de fluxo do Correlation ID, detalhando como ele será propagado em todos os pontos de interação (APIs, mensageria, eventos de negócio, webhooks), sendo crucial para a tolerância a falhas, *rollback* de operações e monitoramento.

Equipe do Projeto e Papeis

Esta seção define a estrutura e as responsabilidades da equipe multifuncional designada para o Projeto Flow Wise.

A formação de uma célula estratégica dedicada garante a agilidade e o foco necessários para o sucesso do Produto Mínimo Viável (MVP) e estabelece a base para a futura evolução e crescimento do projeto.

Estrutura da Célula Estratégica Flow Wise

A célula estratégica do Flow Wise é composta por profissionais-chave, operando de forma integrada para cobrir todas as frentes do desenvolvimento e gestão do produto.

A célula estratégica do Projeto Flow Wise é composta pelos seguintes papeis, fundamentais para a execução e o sucesso desta iniciativa:

- **Líder do Projeto / Arquiteto Principal**
    - **Responsabilidades:** Definição e evolução da visão estratégica e arquitetural do projeto, garantia do alinhamento técnico com os objetivos de negócio, governança sobre as decisões de design, liderança técnica da equipe, e comunicação com a diretoria e *stakeholders* de alto nível.
- **Product Owner (PO)**
    - **Responsabilidades:** Representar a voz do cliente e do negócio, definir e priorizar o backlog do produto, maximizar o valor entregue, e garantir a clareza dos requisitos funcionais e não-funcionais.
- **Scrum Master (SM)**
    - **Responsabilidades:** Facilitar o processo ágil de desenvolvimento, remover impedimentos que afetam a equipe, garantir a adesão às práticas ágeis e cerimônias, e otimizar a eficiência e a colaboração da célula.
- **Arquiteto de Solução**
    - **Responsabilidades:** Detalhar os designs dos microsserviços e componentes, assegurar a aplicação dos padrões de arquitetura (DDD, CQRS, Event Sourcing), realizar revisões de código e design, e garantir a consistência técnica da solução.
- **Desenvolvedores Backend (C#)**
    - **Responsabilidades:** Implementar a lógica de negócio, APIs e integração dos microsserviços utilizando C#/.NET 8+, garantir a qualidade do código através de testes unitários e de integração, e colaborar no design técnico das funcionalidades.
- **Engenheiros de QA/Testes**
    - **Responsabilidades:** Projetar e executar planos de teste (unitários, integração, aceitação, performance), identificar e documentar defeitos, garantir a cobertura de testes e validar que a solução atende aos requisitos de qualidade e NFRs.
- **Engenheiros DevOps**

- **Responsabilidades:** Desenvolver e manter os pipelines de CI/CD, implementar a infraestrutura como código (IaC) com Terraform, configurar monitoramento e observabilidade, e garantir a operação, escalabilidade e resiliência dos microsserviços em ambiente de nuvem.