Roadmap de Faseamento Estratégico da Doutrina Arquitetural

Fase 0: Fundação Imediata (O Agora - Triage | Próximas 2-4 semanas)

Foco: "Estancar o sangramento". Mitigar riscos críticos imediatos. Estabelecer visibilidade mínima, segurança fundamental e migrar para uma infraestrutura controlada e viável.

I. Fundamentos Estratégicos e Requisitos

- **Ponto 6 - Definição dos Limites do Sistema (Scope):**
  - *Justificativa:* Documentar o escopo atual (80% feito) imediatamente é vital para focar na estabilização do MVP e prevenir *scope creep* descontrolado.

  Linha de Raciocínio: O rigor no escopo do MVP é vital. O MVP deve focar na validação das hipóteses mais arriscadas (Riskiest Assumptions). Precisamos de um processo rigoroso e formal para gerenciar mudanças de escopo.

  Subtópicos Obrigatórios:

  Definição clara do MVP, "In-Scope" e "Out-of-Scope".

  Roadmap de alto nível (Pós-MVP).

Processo de Gerenciamento de Mudanças de Escopo (Scope Change Management) Formal.

Mapeamento das Premissas Mais Arriscadas (Riskiest Assumptions Mapping) a serem validadas pelo MVP.

- **Ponto 7 - Requisitos Arquiteturalmente Significativos (RAS):**
  - *Justificativa:* Identificar os NFRs mais críticos (ex: segurança financeira, disponibilidade) agora para guiar as decisões de infraestrutura e estabilização imediatas.

Linha de Raciocínio: Os NFRs direcionam todos os trade-offs e são conflitantes. Priorização rigorosa e quantificada (SLOs). Usar Orçamentos de Erro (Error Budgets). Como o sistema se comportará sob estresse extremo, além dos SLOs normais (Ponto de Saturação)?

Subtópicos Obrigatórios:

Priorização dos NFRs (Matriz de Priorização) e Quantificação (SLOs).

Cenários de Qualidade (Quality Attribute Scenarios) documentados.

Definição do Orçamento de Erro (Error Budget).

Análise de Conflitos e Matriz de Interdependência de NFRs.

Requisitos de Comportamento sob Estresse Extremo e Ponto de Saturação (System Saturation Point).

- **Ponto 8 - Restrições (Constraints):**
  - *Justificativa:* Mapear as limitações atuais (PostgreSQL obrigatório, Replit, integrações legadas) é essencial para criar um plano de migração realista.

Linha de Raciocínio: Arquitetura pragmática dentro das limitações. Identificar restrições "duras" e "suaves". O plano de mitigação para restrições críticas deve ser parte integral da arquitetura. Analisar o impacto de cada restrição.

Subtópicos Obrigatórios:

Restrições documentadas (Técnicas, Orçamentárias, Prazo, Legais).

Análise das competências da equipe (Skills Gap Analysis).

Restrições de Integração com Sistemas Legados.

Plano de Ação para Mitigação de Restrições Críticas.

Classificação das Restrições (Duras vs. Suaves).

Análise de Impacto das Restrições na Arquitetura (Constraint Impact Analysis).

II. Macro-arquitetura e Padrões de Alto Nível

- **Ponto 18 - Diagramas de Arquitetura (Visão Macro):**
  - *Justificativa:* Criar diagramas C4 Nível 1 (Contexto) e de Deployment *As-Is* imediatamente. Precisamos entender o que existe hoje para migrar e proteger.

Linha de Raciocínio: A documentação deve ser viva ("Diagrams as Code"). Precisamos de múltiplas vistas (Modelo 4+1) além do C4 (ex: Vista de Segurança, Vista de Dados, Vista de Operações).

Subtópicos Obrigatórios:

Diagrama de Contexto (C4 Nível 1) e Contêineres (C4 Nível 2).

Diagrama de Deployment (Visão Física/Infraestrutura).

Adoção de Ferramentas para "Diagrams as Code" (ex: Structurizr).

Vistas Arquiteturais Adicionais (Modelo 4+1, ex: Vista de Segurança, Vista de Operações, Vista de Dados).

V. Arquitetura de Dados

- **Ponto 41 - Estratégia de Persistência (Gestão de Schema):**
  - *Justificativa:* Implementar uma ferramenta de migração de banco de dados (ex: Flyway/Liquibase) imediatamente para controlar as mudanças no schema de forma versionada.

Linha de Raciocínio: Isolamento rigoroso (Database per Service). O desafio é gerenciar consultas cross-service e a evolução do esquema sem downtime (Zero Downtime Migration usando padrões como Expand/Contract).

Subtópicos Obrigatórios:

Definição do padrão de isolamento e Regras de acesso (apenas via API/Eventos).

Estratégia para consultas multi-serviço (API Composition, CQRS).

Estratégia de Gerenciamento de Schema e Migrações (ex: Flyway, Liquibase).

Padrões para Evolução de Schema sem Downtime (Zero Downtime Schema Migration) (ex: Expand/Contract).

- **Ponto 45 - Classificação de Dados:**
  - *Justificativa:* Identificar imediatamente onde estão os dados sensíveis (PII, financeiros) é a base para aplicar controles de segurança urgentes.

  Linha de Raciocínio: A base da segurança de dados. A classificação deve ser automatizada (Descoberta Automática) e integrada com ferramentas de DLP (Data Loss Prevention).

  Subtópicos Obrigatórios:

  Níveis de sensibilidade definidos e Mapeamento de PII/PHI.

  Tagging automatizado de recursos.

  Ferramentas de Descoberta Automática de Dados Sensíveis.

  Estratégia de Prevenção de Perda de Dados (DLP - Data Loss Prevention) integrada.

VII. Infraestrutura e Deployment (DevOps/Cloud/SRE)

- **Ponto 62 - Estratégia de Nuvem:**
  - *Justificativa:* Decisão urgente: sair do Replit. Selecionar o provedor de nuvem primário e definir a estrutura de contas básica (Landing Zone).

  Linha de Raciocínio: A governança da estrutura de contas (Landing Zone) é fundamental. A portabilidade deve ser garantida por abstrações (K8s), e a Estratégia de Saída deve ser viável. Construir uma Plataforma de Desenvolvedor Interna (IDP) para padronização e eficiência.

  Subtópicos Obrigatórios:

  Seleção do Provedor de Nuvem Primário e Estratégia Multi-Cloud/Hybrid.

  Desenvolvimento de uma Estratégia de Saída (Exit Strategy).

  Definição da Estrutura de Contas/Organizações na Nuvem (Landing Zone).

  Modelo de Governança de Nuvem (Cloud Governance Model).

  Estratégia de Plataforma de Desenvolvedor Interna (Internal Developer Platform - IDP).

- **Ponto 67 - Estratégia de Ambientes:**

- *Justificativa:* Definir e provisionar ambientes separados (Staging, Produção) na nova plataforma de nuvem para garantir isolamento.

Linha de Raciocínio: Garantir paridade via Infraestrutura Imutável. Ambientes efêmeros para velocidade. Controle de custos e segurança dos dados (higienização).

Subtópicos Obrigatórios:

Definição do propósito e políticas de acesso.

Estratégia para Ambientes Efêmeros.

Política de Higienização de Dados e Controle de Custo.

Estratégia de Infraestrutura Imutável (Immutable Infrastructure Strategy).

- **Ponto 71 - Gerenciamento de Configuração:**
  - *Justificativa:* Externalizar configurações (12-Factor App) é uma ação de baixo esforço e alto impacto imediato para segurança e preparo para múltiplos ambientes.

Linha de Raciocínio: Externalizar configurações (12-Factor App). Feature Flags para Progressive Delivery. Gerenciar ativamente o ciclo de vida dos Feature Flags (dívida técnica).

Subtópicos Obrigatórios:

Seleção da ferramenta para armazenamento de configurações externas.

Implementação de plataforma de Feature Flags/Toggles.

Estratégia de atualização dinâmica de configuração.

Auditoria e Versionamento de Configurações.

Governança e Ciclo de Vida dos Feature Flags (Technical Debt Management for Flags).

- **Ponto 72 - Pipelines de CI/CD:**
  - *Justificativa:* Estabelecer um pipeline mínimo automatizado (CI/CD) para mover o deployment do manual/Replit para um processo controlado e seguro.

Linha de Raciocínio: O pipeline é a cadeia de suprimentos de software. Otimizar para velocidade (DORA) e segurança (DevSecOps). A segurança

da própria cadeia (SLSA, SBOM) é crítica. O pipeline em si deve ser seguro (Pipeline Security).

Subtópicos Obrigatórios:

Seleção da ferramenta de CI/CD e Definição dos estágios.

Métricas DORA a serem monitoradas.

Implementação de Segurança da Cadeia de Suprimentos (SLSA framework, SBOM).

Estratégia de Gerenciamento de Artefatos.

Hardening do Sistema de CI/CD e Plano de Segurança do Pipeline (Pipeline Security Plan).

- **Ponto 76 - Estratégia de Backup e Restore:**
  - *Justificativa:* Configurar backups automáticos do PostgreSQL imediatamente. Operar sem backup em um sistema financeiro é inaceitável.

  Linha de Raciocínio: Defesa contra Ransomware exige backups imutáveis e isolados (Cross-Account). O tempo de restauração é a métrica chave e deve ser testado automaticamente e regularmente. Definir SLAs de Restauração.

  Subtópicos Obrigatórios:

  Frequência de backup e política de retenção.

  Estratégia de Imutabilidade de Backups.

  Testes de Restauração Automatizados e Regulares.

  Estratégia de Backup Cross-Region/Cross-Account (Isolamento).

  Definição do SLA de Tempo de Restauração (Restore Time SLA).

VIII. Qualidades Sistêmicas e Cross-Cutting Concerns (NFRs)

- **Ponto 82 - Gestão de Chaves e Segredos:**
  - *Justificativa:* Risco crítico. Remover imediatamente todos os segredos hardcoded e migrá-los para um gerenciador de segredos seguro (ex: Vault, Secrets Manager).

Linha de Raciocínio: Armazenamento seguro (HSM/KMS) e rotação automatizada. Acesso dinâmico e auditado. Criptografia de Envelope é mandatória. Plano de DR para o gerenciador de segredos.

Subtópicos Obrigatórios:

Ferramenta de Gerenciamento de Segredos (ex: Vault) e Políticas de Rotação.

Integração no ambiente de execução (Dynamic Secrets).

Estratégia de Criptografia de Envelope (Envelope Encryption) usando KMS/HSM.

Plano de Recuperação de Desastres para o Gerenciador de Segredos (Secrets Manager DR Plan).

- **Ponto 92 - Excelência Operacional - Observabilidade (o11y):**
  - *Justificativa:* Implementar logging estruturado básico e métricas essenciais (APM) agora. Sem visibilidade mínima, estamos voando às cegas.

  Linha de Raciocínio: Objetivo: reduzir MTTD/MTTR. Desafios em escala: gerenciar custos, cardinalidade de métricas e amostragem de traces. Continuous Profiling é essencial. Precisamos de uma Revisão de Prontidão Operacional (ORR) formal.

  Subtópicos Obrigatórios:

  Seleção do Stack de Ferramentas (APM) e Logging Estruturado.

  Implementação de Tracing Distribuído (OpenTelemetry) e Correlation IDs obrigatórios.

  Estratégia de Amostragem (Sampling) para Tracing.

  Estratégia de Gerenciamento de Cardinalidade de Métricas (Metric Cardinality Management).

  Implementação de Continuous Profiling em Produção.

  Gestão de Custos de Observabilidade (Observability Cost Management).

  Checklist de Revisão de Prontidão Operacional (Operational Readiness Review - ORR).

- **Ponto 93 - Gestão de Incidentes:**

- *Justificativa:* Definir um processo mínimo de resposta a incidentes (Quem chamar? Onde comunicar?) antes que ocorra a primeira grande falha.

Linha de Raciocínio: Otimizar MTTR. Alertas acionáveis baseados em SLOs. Adotar um Sistema de Comando de Incidentes (ICS) formal é crucial para coordenação em larga escala. Treinamento e simulação (Drills) são necessários.

Subtópicos Obrigatórios:

Planejamento de Resposta a Incidentes (IRP) e Severidade (SEV).

Criação de Runbooks e Playbooks.

Estratégia de Suporte e Escalonamento (On-Call Rotation).

Processo de Post-Mortem (Blameless) e RCA.

Definição de Alertas Acionáveis baseados em SLOs.

Definição do Sistema de Comando de Incidentes (ICS - Incident Command System).

Treinamento e Simulação de Resposta a Incidentes (Incident Response Drills).

IX. Governança, Stacks e Documentação

- **Ponto 96 - Stack Tecnológica Completa:**
  - *Justificativa:* Fazer um inventário da stack atual para entender a superfície tecnológica e identificar riscos imediatos.

Linha de Raciocínio: Equilibrar padronização com inovação (Tech Radar). Definir Princípios Arquiteturais claros para guiar as decisões descentralizadas. Gerenciar ativamente a obsolescência tecnológica (Lifecycle Management).

Subtópicos Obrigatórios:

Lista detalhada de tecnologias aprovadas e Tech Radar.

Processo de Governança de Stack.

Estratégia de Depreciação/Obsolescência Tecnológica (Technology Lifecycle Management).

Definição dos Princípios Arquiteturais (Architectural Principles).

- **Ponto 100 - Estratégia de Branching e Release Management:**

- *Justificativa:* Definir o modelo de branching (ex: Trunk-Based simplificado) imediatamente para organizar o fluxo de trabalho e controlar releases.

Linha de Raciocínio: Otimizar o fluxo de entrega. Trunk-Based Development favorece velocidade. Definir processo claro de Hotfix.

Subtópicos Obrigatórios:

Escolha do modelo de branching (Trunk-Based preferível).

Processo de gerenciamento de releases e Versionamento Semântico (SemVer).

Definição do Processo de Hotfix.

Fase 1: Desenvolvimento Contínuo (O Durante | 2-6 meses)

Foco: Integrar melhores práticas ao fluxo de desenvolvimento diário sem grande disrupção. Melhorias incrementais e refatorações de oportunidade.

I. Fundamentos Estratégicos e Requisitos

- **Ponto 1 - Objetivos de Negócio e Drivers:**
  - *Justificativa:* Garantir que todo novo desenvolvimento esteja explicitamente alinhado aos OKRs e KPIs definidos, movendo da execução tática para a estratégica.

Linha de Raciocínio: Qual é a visão de longo prazo (5-10 anos) e como a arquitetura habilita a Opcionalidade Estratégica (capacidade de pivotar, M&A)? Qual a tolerância a risco do negócio (Risk Appetite)? Como a arquitetura se alinha ao sistema socio-técnico (Alinhamento Socio-Técnico)? Como ela suporta o modelo de monetização e a vantagem competitiva sustentável? Analisar o impacto geopolítico e de mercado.

Subtópicos Obrigatórios:

Definição dos OKRs e KPIs quantificáveis.

Personas de Usuários e Jobs To Be Done (JTBD).

Análise do Cenário Competitivo e Definição da Vantagem Competitiva Sustentável.

Mapa de Stakeholders e Matriz RACI.

Mapeamento do Fluxo de Valor (Value Stream Mapping).

Vida útil esperada e Critérios de Sucesso/Saída (Exit Criteria).

Análise da Volatilidade do Domínio (Taxa de mudança esperada).

Estratégias de Pivô Arquitetural (Plan B) e Análise de Flexibilidade Estratégica.

Perfil de Tolerância a Risco do Negócio (Risk Appetite Statement).

Análise de Impacto Socio-Técnico (Socio-Technical Impact Analysis).

Análise de Fatores PESTEL (Político, Econômico, Social, Tecnológico, Ambiental, Legal) com impacto arquitetural.

entre outros...

- **Ponto 9 - Modelagem de Domínio (DDD):**
    - *Justificativa:* Introduzir a Linguagem Ubíqua e identificar Bounded Contexts. Aplicar princípios de DDD em novas funcionalidades e refatorações incrementais.

  Linha de Raciocínio: A decisão mais crítica para o desacoplamento e alinhamento socio-técnico. Precisamos de mecanismos arquiteturais para impor os limites dos Bounded Contexts (Enforcement Automatizado), caso contrário, a modularidade se degradará.

  Subtópicos Obrigatórios:

  Linguagem Ubíqua e Identificação dos Domínios (Core/Suporte/Genéricos).

  Artefatos do Event Storming e Identificação dos Bounded Contexts.

  Criação do Mapa de Contextos (Context Map) e definição dos Padrões Estratégicos (ACL, OHS/PL).

  Definição rigorosa das Invariantes de Domínio (Domain Invariants).

  Estratégia para Enforcement Automatizado dos Limites de Contexto (ex: ArchUnit) integrada ao CI.

  Análise de Alinhamento Socio-Técnico (Socio-Technical Alignment Analysis).

II. Macro-arquitetura e Padrões de Alto Nível

- **Ponto 12 - Estilo Arquitetural Principal:**
    - *Justificativa:* Definir formalmente o estilo arquitetural alvo (ex: Monolito Modular bem estruturado) para guiar a evolução.

Linha de Raciocínio: Escolha pragmática. Quantificar o custo da complexidade distribuída. Definir critérios de gatilho objetivos para evolução e usar Fitness Functions automatizadas para validação contínua.

Subtópicos Obrigatórios:

Análise comparativa detalhada (Trade-off Analysis Matrix).

Plano de Evolução Controlada e Roadmap Arquitetural.

ADR (Architecture Decision Record) detalhado.

Definição dos Critérios de Gatilho (Trigger Criteria) para evolução.

Definição das Fitness Functions iniciais (Automatizadas).

Análise Quantitativa do Custo da Complexidade Distribuída (Distributed Complexity Cost Analysis).

- **Ponto 19 - Padrões de Integração e Comunicação:**
  - *Justificativa:* Definir critérios claros para comunicação síncrona vs. assíncrona e aplicá-los a todas as novas integrações.

  Linha de Raciocínio: "Assíncrono por padrão, Síncrono quando necessário" para minimizar o acoplamento temporal. Evitar APIs "tagarelas" (Chatty APIs).

  Subtópicos Obrigatórios:

  Critérios para uso de Comunicação Síncrona e Assíncrona.

  Definição da granularidade da comunicação (evitar Chatty APIs).

  Análise de Acoplamento Temporal (Temporal Coupling Analysis).

III. Micro-arquitetura e Design de Componentes (Backend)

- **Ponto 20 - Design Interno dos Componentes:**
  - *Justificativa:* Adotar um padrão interno (ex: Hexagonal/Clean) para todo novo código e iniciar o uso de validação automatizada (ArchUnit).

  Linha de Raciocínio: Isolar o Core de negócio (Hexagonal/Clean). O Modelo de Concorrência deve ser explicitamente definido (Threads, Reactive). Usar automação (ex: ArchUnit) integrada ao CI para impor a modularidade. Gerenciamento rigoroso de recursos (Pools).

  Subtópicos Obrigatórios:

  Seleção do padrão arquitetural interno e Regras de dependência (DIP).

Template padronizado para novos serviços.

Definição do Modelo de Concorrência interno (Threads, Actors, Coroutines, Reactive Streams).

Ferramentas de Validação de Dependência Automatizada (ex: ArchUnit) integradas ao CI.

Estratégia de Gerenciamento de Recursos (Resource Management) (Thread Pools, Connection Pools).

- **Ponto 21 - Lógica de Negócio e Fluxos de Trabalho:**
  - *Justificativa:* Focar na proteção de invariantes e no design de Agregados (DDD) para novas funcionalidades, garantindo consistência.

Linha de Raciocínio: Proteger as Invariantes. O design dos Agregados (DDD) define os limites transacionais exatos. Ciclos de vida complexos exigem Máquinas de Estado explícitas para gerenciar a complexidade do estado. Análise de Complexidade Ciclomática.

Subtópicos Obrigatórios:

Identificação das invariantes de negócio críticas.

Design dos Agregados (Aggregates - DDD) e Modelagem de Consistência.

Estratégia para Validação de Regras de Negócio.

Definição de Máquinas de Estado (State Machines) para ciclos de vida complexos.

Análise de Complexidade Ciclomática e Estratégia de Refatoração.

- **Ponto 25 - Padrões de Design (Design Patterns):**
  - *Justificativa:* Incentivar o uso de padrões estabelecidos (ex: Repository, DI) no novo código durante o code review.

Linha de Raciocínio: Foco em padrões que gerenciem complexidade, concorrência (Locking) e tratamento de erros de forma robusta. A Injeção de Dependência (DI) é fundamental para a testabilidade.

Subtópicos Obrigatórios:

Padrões GoF relevantes e Padrões de persistência (Repository, Unit of Work).

Padrões para Gerenciamento de Concorrência (Locking Otimista/Pessimista).

Padrões de Tratamento de Erros robustos.

Padrões de Injeção de Dependência (DI) e Inversão de Controle (IoC).

- **Ponto 28 - Diagramas de Componentes (C4 Model - Nível 3):**
  - *Justificativa:* Detalhar os componentes internos (Nível 3) conforme eles são construídos ou modificados.

  Linha de Raciocínio: Detalhar apenas os serviços críticos. O foco deve ser nas interfaces (Portas) e nos adaptadores, validando a adesão ao padrão interno (ex: Hexagonal).

  Subtópicos Obrigatórios:

  Mapeamento dos componentes internos e interações.

  Identificação das interfaces (Portas de Entrada/Saída) e Adaptadores.

- **Ponto 29 - Diagramas de Sequência/Fluxo:**
  - *Justificativa:* Exigir diagramas de sequência para novas funcionalidades complexas para validar interações.

  Linha de Raciocínio: Essencial para analisar interações distribuídas. Focar nos cenários de falha (Unhappy Paths) para validar a resiliência e identificar o caminho crítico (Critical Path) para otimização de latência. Analisar pontos de falha distribuídos.

  Subtópicos Obrigatórios:

  Modelagem dos fluxos de autenticação/autorização e transações complexas.

  Modelagem detalhada dos fluxos de erro e recuperação (Unhappy Path).

  Análise de Latência Preditiva.

  Identificação de Chamadas Síncronas Críticas (Critical Path Analysis).

  Análise de Pontos de Falha Distribuídos (Distributed Failure Point Analysis).

IV. Design de APIs, Interfaces e Comunicação

- **Ponto 30 - Protocolos de Comunicação:**
  - *Justificativa:* Padronizar os protocolos e formatos de serialização utilizados no desenvolvimento contínuo.

  Linha de Raciocínio: Escolher o protocolo certo (REST/gRPC/GraphQL). Analisar a eficiência do protocolo (overhead) é crucial em escala.

Padronizar a segurança (mTLS) para comunicação interna.

Subtópicos Obrigatórios:

Critérios definidos para REST vs. gRPC vs. GraphQL.

Seleção do formato de serialização e Estratégia de Compressão.

Padrões de Comunicação Cross-Origin (CORS).

Estratégia de mTLS (Mutual TLS) mandatória para comunicação interna.

Análise de Overhead de Protocolo (Protocol Overhead Analysis).

- **Ponto 33 - Contrato da API (API Contract):**
  - *Justificativa:* Adotar abordagem "Design-First" com OpenAPI V3 para todas as novas APIs.

Linha de Raciocínio: Design-First é mandatório. Os Testes de Contrato (Contract Testing - CDC) são essenciais para evolução independente e detectar quebras de compatibilidade no CI.

Subtópicos Obrigatórios:

Adoção do OpenAPI V3 / AsyncAPI.

Processo de Governança (Design-First e Revisão).

Estratégia de Geração Automática de Código (SDKs/Stubs).

Estratégia de Testes de Contrato (Contract Testing) (ex: Pact, Consumer-Driven Contracts).

Validação de Compatibilidade Retroativa (Backward Compatibility Validation) automatizada no CI.

- **Ponto 34 - Design de APIs RESTful (Padrões de Interface):**
  - *Justificativa:* Aplicar o Guia de Estilo de API rigorosamente a novos endpoints, incluindo versionamento e idempotência.

Linha de Raciocínio: Consistência rigorosa (API Style Guide imposto por Linting). A idempotência (Idempotency-Key) é mandatória para resiliência. A cacheabilidade (HTTP Caching) deve ser explicitamente projetada.

Subtópicos Obrigatórios:

Estratégia de Versionamento Mandatória.

Uso Correto e Semântico de Métodos HTTP e Recursos.

Padronização de Cabeçalhos (Correlation-ID).

Garantias de Idempotência (Idempotency-Key obrigatório) e estratégia de armazenamento de chaves.

Estratégia de Cacheabilidade (HTTP Caching: ETag, Cache-Control).

Definição do Guia de Estilo de APIs (API Style Guide) detalhado e imposto por Linting.

- **Ponto 35 - Contrato de Dados (Payloads):**
  - *Justificativa:* Implementar validação rigorosa e sanitização de entrada na borda para todas as novas APIs.

  Linha de Raciocínio: Estrutura rigorosa e validação na borda (Fail Fast/Sanitização de Entrada). A política de evolução do schema deve ser clara. Gerenciar PII nos payloads (Criptografia de Campo/Tokenização).

  Subtópicos Obrigatórios:

  Padrões de nomenclatura e formatos de dados (ISO 8601).

  Repositório centralizado de Schemas (JSON Schema).

  Estratégia de Validação de Payloads na borda (Sanitização de Entrada).

  Estratégia para lidar com campos sensíveis (PII) nos payloads (Criptografia de Campo/Tokenização).

  Política de Evolução de Schema e Compatibilidade (Schema Evolution Policy).

- **Ponto 36 - Comunicação de Resultados e Erros:**
  - *Justificativa:* Adotar RFC 7807/9457 e Correlation IDs em todo novo código para melhorar a depuração e a DX das APIs.

  Linha de Raciocínio: Erros são a interface de depuração. RFC 7807 (Problem Details) é mandatório. Trace IDs são essenciais. Precisamos de estratégias para erros em lote.

  Subtópicos Obrigatórios:

  Mapeamento completo e semântico dos Códigos de Status HTTP.

  Implementação mandatória do padrão RFC 7807/9457.

  Catálogo de erros de negócio padronizado.

  Inclusão de IDs de Correlação (Trace IDs) em todas as respostas de erro.

  Estratégia para tratamento de erros em lote (Batch Error Handling).

- **Ponto 37 - Interação com Coleções:**
    - *Justificativa:* Padronizar a paginação (preferencialmente Cursor-based) e a filtragem em todos os novos endpoints de coleção.

  Linha de Raciocínio: Paginação eficiente (Cursor-based) é mandatória para escala. Limites de tamanho de página são necessários para proteção contra DoS.

  Subtópicos Obrigatórios:

  Estratégia de paginação padrão (Cursor-based preferível).

  Sintaxe padrão para filtragem e ordenação.

  Estratégia para Sparse Fieldsets.

  Limites de Tamanho de Página (Page Size Limits) obrigatórios e não negociáveis.

V. Arquitetura de Dados

- **Ponto 39 - Modelagem de Dados:**
    - *Justificativa:* Garantir que novas modelagens sejam revisadas com foco nos padrões de acesso e estratégia de indexação no PostgreSQL.

  Linha de Raciocínio: Modelagem otimizada para Padrões de Acesso (Query-First Design). Estimativas de volumetria e estratégia de evolução do esquema. Considerar modelagem de dados temporais (Bi-temporalidade) para auditoria e análise histórica rigorosa.

  Subtópicos Obrigatórios:

  Modelo Conceitual, Lógico e Físico.

  Análise dos Padrões de Acesso a Dados (Data Access Patterns Analysis).

  Estratégia de Indexação detalhada e Justificativa para Normalização/Desnormalização.

  Estimativas de Volumetria de Dados (Data Volumetry Estimates).

  Estratégia de Evolução do Schema (Schema Evolution Strategy).

  Modelagem de Dados Temporais (Temporal/Bi-temporal Data Modeling) (se aplicável).

- **Ponto 51 - Gestão de Transações:**

- *Justificativa:* Garantir que o escopo das transações ACID esteja correto e implementar idempotência para operações críticas.

Linha de Raciocínio: Padrão Saga para transações distribuídas. Exige design rigoroso das transações de compensação (rollback semântico) e idempotência absoluta. Análise detalhada das falhas da Saga e pontos de não retorno.

Subtópicos Obrigatórios:

Escopo das transações ACID locais (Agregados).

Design detalhado das Sagas e Transações de Compensação.

Requisitos de Idempotência para todas as etapas da Saga.

Monitoramento e Alertas para Falhas em Sagas.

Análise Detalhada de Falhas da Saga e Pontos de Não Retorno (Point of No Return Analysis).

VI. Design de Frontend e Experiência do Usuário (UX/UI)

- **Ponto 56 - Arquitetura do Frontend Completa:**
  - *Justificativa:* Definir a arquitetura de frontend alvo e estabelecer orçamentos de performance iniciais.

Linha de Raciocínio: Otimizar para Core Web Vitals com Orçamentos de Performance rigorosos. Microfrontends exigem governança rigorosa de composição e dependências. Monitorar a performance real do usuário (RUM) continuamente. Otimizar o Caminho Crítico de Renderização.

Subtópicos Obrigatórios:

Seleção do Framework e Estratégia de Renderização (CSR, SSR, SSG, ISR).

Estratégia Mobile (Nativo, Híbrido, PWA).

Decisão sobre Microfrontends, modelo de composição e Estratégia de Orquestração/Governança.

Definição do Orçamento de Performance (Performance Budgeting).

Estratégia de Gerenciamento de Dependências e Monorepo.

Estratégia de Monitoramento de Performance em Produção (RUM - Real User Monitoring).

Otimização do Caminho Crítico de Renderização (Critical Rendering Path Optimization).

- **Ponto 59 - Gerenciamento de Estado no Cliente:**
  - *Justificativa:* Padronizar a arquitetura de estado e estratégias de caching/sincronização no frontend.

  Linha de Raciocínio: Distinguir Estado de UI e Estado de Servidor Cacheado. O desafio é a sincronização, caching e invalidação eficientes (ex: React Query).

  Subtópicos Obrigatórios:

  Seleção da biblioteca e Definição da arquitetura de estado.

  Estratégia de Caching, Sincronização e Invalidação de Estado do Servidor.

  Estratégia de Persistência de Estado no Cliente (LocalStorage).

- **Ponto 60 - Comunicação Frontend-Backend:**
  - *Justificativa:* Implementar medidas básicas de segurança no frontend (XSS/CSRF mitigation) e avaliar a necessidade de um BFF.

  Subtópicos Obrigatórios:

  Definição da necessidade de um BFF e Avaliação de GraphQL vs. REST.

  Padrões de Resiliência no Frontend.

  Estratégia Offline-First (se aplicável) e sincronização de dados.

  Implementação Rigorosa de Políticas de Segurança HTTP (CSP, HSTS, Feature Policy).

  Estratégia de Segurança do Frontend (XSS, CSRF Mitigation) detalhada.

VII. Infraestrutura e Deployment (DevOps/Cloud/SRE)

- **Ponto 63 - Estratégia de Migração de Plataforma:**
  - *Justificativa:* Executar a migração completa do Replit para a nova infraestrutura de nuvem definida na Fase 0.

  Linha de Raciocínio: Minimizar risco e downtime. Um plano de rollback detalhado e testado é essencial.

  Subtópicos Obrigatórios:

  Escolha da Estratégia de Migração (6 R's) e Análise de dependências.

  Planejamento das fases de migração e cutover.

  Plano de Contingência e Rollback da Migração detalhado e testado.

- **Ponto 69 - Infrastructure as Code (IaC):**
    - *Justificativa:* Garantir que toda a nova infraestrutura seja provisionada via IaC (ex: Terraform), garantindo reprodutibilidade.

  Linha de Raciocínio: Tudo como Código. GitOps é o modelo operacional alvo. Precisamos de testes de IaC rigorosos, Policy as Code (OPA) para governança automatizada e detecção de desvios (Drift Detection) com remediação automática.

  Subtópicos Obrigatórios:

  Seleção da ferramenta de IaC (ex: Terraform) e Estrutura de repositórios.

  Adoção de práticas de GitOps (ex: ArgoCD, Flux).

  Estratégia de Testes de Infraestrutura (IaC Testing).

  Estratégia de Detecção de Drift (Drift Detection) e Remediação Automática.

  Implementação de Policy as Code (ex: OPA, Kyverno) para governança de infraestrutura.

- **Ponto 74 - Estratégias de Rollback:**
    - *Justificativa:* Exigir que as migrações de DB sejam retrocompatíveis (Expand/Contract) e que haja procedimentos de rollback de aplicação.

  Linha de Raciocínio: Otimizar para MTTR. O maior desafio é o banco de dados: migrações de schema DEVEM ser retrocompatíveis (Expand/Contract pattern) e testadas automaticamente.

  Subtópicos Obrigatórios:

  Procedimentos de rollback automatizados para aplicação.

  Estratégia mandatória para Migrações de Banco de Dados Compatíveis (Backward-Compatible Schema Changes).

  Testes Automatizados de Compatibilidade de Migração de DB e Rollback.

VIII. Qualidades Sistêmicas e Cross-Cutting Concerns (NFRs)

- **Ponto 80 - Segurança (Security by Design) e Privacidade:**
    - *Justificativa:* Introduzir checklists de segurança (OWASP) e modelagem de ameaças básica no ciclo de desenvolvimento para novas funcionalidades.

Linha de Raciocínio: Mentalidade Zero Trust. Modelagem de Ameaças contínua (Internas e Externas). A segurança da cadeia de suprimentos (SLSA) é crítica. Preparação para análise forense. Monitoramento contínuo da postura de segurança (CSPM). Preparação para Criptografia Pós-Quântica.

Subtópicos Obrigatórios:

Metodologia de Modelagem de Ameaças (STRIDE) e Checklist OWASP.

Modelo de Autorização detalhado (RBAC/ABAC/ReBAC) e Princípio do Menor Privilégio.

Estratégia de Criptografia.

Modelagem de Ameaças Internas (Insider Threat Modeling).

Prontidão para Análise Forense (Forensic Readiness).

Implementação do Framework SLSA (Supply-chain Levels for Software Artifacts).

Estratégia de Cloud Security Posture Management (CSPM).

Roadmap para Criptografia Pós-Quântica (Post-Quantum Cryptography Roadmap).

- **Ponto 81 - Estratégia de Identidade Federada e SSO:**
  - *Justificativa:* Padronizar os protocolos de autenticação (OIDC) e gerenciamento de sessão seguro.

  Linha de Raciocínio: Identidade é o novo perímetro. Autenticação forte (MFA/Passwordless). Gestão rigorosa de identidade M2M (mTLS/OAuth). Monitoramento de anomalias (Autenticação Adaptativa Baseada em Risco).

  Subtópicos Obrigatórios:

  Seleção do IdP e Protocolos (OIDC, SAML).

  Estratégia de MFA/Passwordless e Gerenciamento de Sessão Seguro.

  Estratégia de Autenticação Machine-to-Machine (M2M) (mTLS, OAuth Client Credentials).

  Políticas de Acesso Adaptativo Baseado em Risco (Risk-Based Adaptive Access Policies).

- **Ponto 88 - Confiabilidade e Resiliência (Reliability):**

- *Justificativa:* Implementar padrões de resiliência básicos (Retries com backoff, Timeouts) em todas as novas integrações externas.

Linha de Raciocínio: Design for Failure (SRE). Prevenir falhas em cascata, isolar falhas (Bulkheads) e garantir degradação graciosa. Usar Load Shedding em sobrecarga extrema. Foco em MTBF. Buscar Antifragilidade (aprender e melhorar com a falha).

Subtópicos Obrigatórios:

Implementação dos Padrões de Resiliência (Circuit Breaker, Retries, Bulkheads).

Design de Dead Letter Queues (DLQs) e Análise de SPOFs.

Estratégia de Load Shedding (Descarte de Carga).

Planos de Degradação Graciosa (Graceful Degradation Plans).

Métricas de Confiabilidade (MTBF - Mean Time Between Failures).

Estratégias de Antifragilidade (Antifragility Strategies).

IX. Governança, Stacks e Documentação

- **Ponto 97 - Estratégia de Ambiente de Desenvolvimento Local (DX):**
  - *Justificativa:* Padronizar o ambiente local (ex: Docker Compose/Dev Containers) para melhorar a produtividade e reduzir o "funciona na minha máquina".

Linha de Raciocínio: A produtividade do desenvolvedor (Inner Loop) impacta o Time-to-Market. Foco em Engenharia de Eficácia do Desenvolvedor (DEE) e medição da experiência (DevEx/SPACE metrics).

Subtópicos Obrigatórios:

Ferramentas padronizadas para o ambiente local (Dev Containers).

Estratégia para simular dependências externas localmente.

Documentação de Onboarding Técnico.

Estratégia de Engenharia de Eficácia do Desenvolvedor (DEE).

Métricas de Eficácia do Desenvolvedor (DevEx/SPACE Metrics).

- **Ponto 99 - Padrões de Codificação e Guias de Estilo:**
  - *Justificativa:* Configurar Linters/Formatters automatizados e bloqueantes no CI para todo novo código.

Linha de Raciocínio: Consistência reduz a carga cognitiva. Padronização 100% automatizada (Linters/Formatters) e bloqueante no CI. Definir Quality Gates automatizados.

Subtópicos Obrigatórios:

Definição das convenções.

Configuração de Linters e Formatters automatizados e bloqueantes no CI.

Métricas de Qualidade de Código Estático.

Definição de Quality Gates Automatizados (Automated Quality Gates).

- **Ponto 101 - Estratégia de Testes (Geral):**
  - *Justificativa:* Definir a Pirâmide de Testes e estabelecer metas mínimas de cobertura para todo novo código entregue.

  Linha de Raciocínio: Construir confiança na qualidade. Pirâmide de Testes. Testes de Contrato são essenciais. Testes de Mutação validam a eficácia dos testes unitários. Considerar Testes em Produção (Testing in Production).

  Subtópicos Obrigatórios:

  Definição da Pirâmide de Testes e Metas de cobertura.

  Estratégia de Testes de Contrato (ex: Pact).

  Estratégia de Testes de Mutação (Mutation Testing).

  Estratégia de Testes em Produção (Testing in Production Strategy) (ex: Shadowing, Traffic Mirroring).

- **Ponto 103 - Estratégia de Testes de Segurança:**
  - *Justificativa:* Integrar ferramentas de análise estática (SAST) e de dependências (SCA) no pipeline de CI.

  Linha de Raciocínio: DevSecOps integrado. Processo claro de triagem e remediação de vulnerabilidades (SLAs de Remediação). Treinamento contínuo (Security Champions).

  Subtópicos Obrigatórios:

  Integração de SAST, DAST e SCA no pipeline.

  Planejamento de Pentests regulares.

Processo de Triagem e Remediação de Vulnerabilidades (SLA de Remediação).

Treinamento de Segurança e Programa de Security Champions.

- **Ponto 108 - Governança, Documentação e Gestão de Mudanças (ADRs):**
  - *Justificativa:* Adotar o uso de Architecture Decision Records (ADRs) para documentar formalmente todas as novas decisões arquiteturais significativas.

  Linha de Raciocínio: Governança colaborativa e automatizada. Registro do "porquê" (ADRs). Documentação como código (Docs as Code). Gestão do conhecimento estratégica. Medir a adoção dos padrões.

  Subtópicos Obrigatórios:

  Definição do Processo de Governança Arquitetural (ARB, RFCs).

  Registro formal e imutável das decisões via Architecture Decision Records (ADRs).

  Manutenção dos Diagramas Arquiteturais (Diagrams as Code).

  Estratégia de Gestão de Mudanças Organizacionais.

  Estratégia de Gestão do Conhecimento (Knowledge Management Strategy).

  Métricas de Adoção dos Padrões Arquiteturais (Architectural Standards Adoption Metrics).

Fase 2: Consolidação e Endurecimento (O Pós-Lançamento/MVP | 6-12 meses)

Foco: Tornar o sistema robusto, escalável e confiável em produção real. Preparação para crescimento e garantia de operabilidade sob carga e cenários de falha.

I. Fundamentos Estratégicos e Requisitos

- **Ponto 5 - Modelagem de Processos de Negócio (BPM):**
  - *Justificativa:* Otimizar os processos existentes com base em dados reais, focando na automação de gargalos e no tratamento rigoroso de caminhos de exceção (Unhappy Paths).

Linha de Raciocínio: Otimizar antes de automatizar. Focar nos caminhos de exceção (Unhappy Paths) e na resiliência do processo. Como o sistema lida com falhas humanas (HCI)? A Simulação de Processos (Digital Twin/Monte Carlo) é vital para validar a eficácia e identificar gargalos sob carga.

Subtópicos Obrigatórios:

Diagramas "As-Is" e "To-Be" (BPMN) e Métricas de Eficiência.

Identificação de gargalos e automação.

Simulação de Processos (Digital Twin/Monte Carlo) para validação dos fluxos "To-Be" sob carga.

Modelagem detalhada dos Caminhos de Exceção e Compensação (Unhappy Paths).

Definição dos pontos de Handover e Intervenção Humana (HCI).

Análise de Resiliência do Processo de Negócio (Business Process Resilience Analysis).

II. Macro-arquitetura e Padrões de Alto Nível

- **Ponto 14 - Decomposição do Sistema:**
  - *Justificativa:* Definir a estratégia de decomposição alvo e começar a medir o acoplamento no código existente, refatorando para impor limites claros.

Linha de Raciocínio: Decomposição por Bounded Contexts e volatilidade. A granularidade é crítica. Monitorar acoplamento (Estrutural e Dinâmico) e coesão para evitar o "Monolito Distribuído". Ter estratégias claras para refatoração de limites (Re-Boundarying).

Subtópicos Obrigatórios:

Heurísticas de decomposição aplicadas e Lista de Serviços/Módulos (SRP).

Matriz de Interação/Dependência de Serviços.

Métricas de Acoplamento e Coesão (Automatizadas) a serem monitoradas.

Taxonomia de Serviços (Service Taxonomy).

Análise de Granularidade de Serviço e Estratégias de Consolidação/Divisão (Re-Boundarying).

Análise de Acoplamento Estrutural e Dinâmico.

- **Ponto 16 - Dependências de Serviços Externos e SLAs:**

- *Justificativa:* Implementar estratégias robustas de resiliência (Circuit Breakers, Fallback) e degradação graciosa para todas as integrações críticas (Banco Inter, ClickSign).

Linha de Raciocínio: Design for Failure. Nosso SLA composto é limitado pelo elo mais fraco. Precisamos de degradação graciosa, fallback e, em casos críticos, redundância de fornecedores com estratégias de roteamento dinâmico.

Subtópicos Obrigatórios:

Mapeamento de integrações externas e Cálculo do SLA Composto.

Design de ACLs e Estratégias de Resiliência/Failover.

Estratégia de Degradação Graciosa (Graceful Degradation) detalhada.

Scorecard de Risco de Dependência Externa.

Estratégia de Redundância de Fornecedores (Vendor Redundancy) e Roteamento Dinâmico.

III. Micro-arquitetura e Design de Componentes (Backend)

- **Ponto 22 - Estratégia de Workflow (Orquestração vs. Coreografia):**
  - *Justificativa:* Se houver transações distribuídas ou processos complexos, implementar o Padrão Saga com rigor em idempotência e monitoramento.

Linha de Raciocínio: Gerenciamento de processos distribuídos (Sagas). A idempotência absoluta em todos os passos é mandatória. A visibilidade operacional do estado da Saga e o tratamento rigoroso de falhas/timeouts são cruciais para a resiliência. Análise detalhada de falhas.

Subtópicos Obrigatórios:

Critérios de escolha (Orquestração/Coreografia) e Ferramentas.

Design do monitoramento e tratamento de erros.

Detalhes da Implementação do Padrão Saga (Timeouts, Compensações).

Estratégia de Idempotência mandatória para todos os passos.

Mecanismos de Visibilidade do Estado da Saga (Saga State Visibility) e Dashboards Operacionais.

Análise de Falhas de Workflow e Estratégias de Recuperação (Workflow Failure Analysis).

- **Ponto 23 - Estratégia de Processamento em Lote (Batch Processing):**
  - *Justificativa:* Otimizar e robustecer os processos em lote, implementando Checkpointing, Bulkheading e Backpressure para lidar com o aumento de volume.

  Linha de Raciocínio: Robustez e eficiência. Idempotência, reinicialização (Checkpointing) e otimização de performance (Chunking). Isolamento de recursos (Bulkheading) e Backpressure são necessários para proteger o sistema OLTP. Métricas de eficiência.

  Subtópicos Obrigatórios:

  Definição dos SLAs (Janelas de Processamento) e Tecnologia.

  Estratégia de Agendamento e Gerenciamento de Erros.

  Estratégia de Checkpointing e Reinicialização.

  Otimização de Performance (Chunking, Paralelismo).

  Implementação de Backpressure e Isolamento de Recursos (Bulkheading).

  Métricas de Eficiência do Processamento em Lote.

- **Ponto 24 - Arquitetura de Busca (Search Architecture):**
  - *Justificativa:* Implementar uma solução de busca dedicada (ex: OpenSearch) se os requisitos de performance superarem as capacidades do PostgreSQL.

  Linha de Raciocínio: O desafio crítico é a sincronização confiável (CDC é preferível) e a reindexação sem downtime (Alias Switching). O design do índice e a estratégia de relevância (Ranking) são fundamentais. Monitorar o lag de indexação e definir SLOs.

  Subtópicos Obrigatórios:

  Seleção da tecnologia de busca e Design dos pipelines de indexação (CDC).

  Estratégia de Reindexação sem downtime (Alias Switching).

  Design do Modelo de Dados de Busca (Index Design) e Estratégia de Relevância (Ranking).

  Monitoramento do Lag de Indexação (Indexing Lag Monitoring) e SLOs.

IV. Design de APIs, Interfaces e Comunicação

- **Ponto 31 - Estratégia de Comunicação em Tempo Real:**
    - *Justificativa:* Escalar a arquitetura de notificações (WebSockets/SSE) com estratégias de Backplane (ex: Redis PubSub) e garantir entrega confiável.

  Linha de Raciocínio: Conexões persistentes são difíceis de escalar. Precisamos de autenticação robusta, garantias de entrega e estratégia de escala horizontal (Backplane). Lidar com reconexões e estado de sessão.

  Subtópicos Obrigatórios:

  Seleção da tecnologia (WebSockets, SSE) e Arquitetura de conexão (Gateway).

  Estratégia de Fallback (Long Polling).

  Mecanismos de Autenticação/Autorização para conexões persistentes.

  Estratégia de Escala Horizontal (Backplane) (ex: Redis PubSub).

  Garantias de Entrega de Mensagens em Tempo Real.

  Estratégia de Gerenciamento de Reconexão e Estado de Sessão (Reconnection and Session State Management).

- **Ponto 32 - Estratégia de Gerenciamento de APIs:**
    - *Justificativa:* Implementar governança automatizada (API Linting), Portal do Desenvolvedor e políticas de depreciação claras.

  Linha de Raciocínio: A governança em escala deve ser automatizada (API Linting). Precisamos gerenciar o ciclo de vida completo e medir a adoção (API Analytics).

  Subtópicos Obrigatórios:

  Design do Portal do Desenvolvedor e Política de Depreciação.

  Estratégia de Monetização (se aplicável).

  Automação da Governança de API (API Linting) integrada ao CI.

  Métricas de Adoção e Uso da API (API Analytics).

  Definição do Ciclo de Vida da API (API Lifecycle Management).

- **Ponto 38 - Comunicação Assíncrona (EDA):**
    - *Justificativa:* Se EDA for adotada, implementar governança rigorosa (Schema Registry), DLQs e tratamento de Poison Pills.

Linha de Raciocínio: Eventos são contratos. Governança rigorosa (Schema Registry). Definir garantias de entrega e ordenação. Idempotência no consumidor é mandatória. Tratamento de mensagens inválidas (Poison Pills), DLQs e Backpressure são cruciais para resiliência operacional.

Subtópicos Obrigatórios:

Design da taxonomia e nomenclatura dos Eventos.

Schema Registry e Estratégia de Evolução de Schema.

Definição dos padrões de Evento (ex: CloudEvents).

Garantias de Entrega (At-Least-Once/Exactly-Once) e Ordenação (Message Keys).

Requisitos de Idempotência para Consumidores.

Estratégia de Dead Letter Queue (DLQ) e Reprocessamento.

Estratégia de Tratamento de Mensagens Inválidas (Poison Pill Handling).

Estratégia de Backpressure em EDA (Consumer Backpressure Strategy)

V. Arquitetura de Dados

- **Ponto 40 - Diagramas de Fluxo de Dados (DFDs):**
  - *Justificativa:* Essencial para compliance (LGPD) e segurança. Mapear o fluxo de PII detalhadamente e identificar Limites de Confiança.

Linha de Raciocínio: Essencial para Modelagem de Ameaças e Compliance (LGPD). Rastrear dados sensíveis e identificar onde cruzam os Limites de Confiança. Integrar DFD com a Modelagem de Ameaças.

Subtópicos Obrigatórios:

Mapeamento das fontes e consumidores.

Identificação dos Limites de Confiança (Trust Boundaries).

Mapeamento do fluxo de dados sensíveis (PII).

Identificação dos pontos de Criptografia e Mascaramento.

Integração do DFD com a Modelagem de Ameaças (Threat Modeling Integration).

- **Ponto 42 - Tecnologias de Banco de Dados Operacionais (OLTP):**
  - *Justificativa:* Implementar padrões de segurança (Hardening) e operação detalhados para o PostgreSQL.

Linha de Raciocínio: Poliglotismo governado. Cada escolha justificada (ADR) e acompanhada de padrões operacionais e de segurança (Hardening). Avaliar o overhead operacional e o custo de manutenção de cada nova tecnologia.

Subtópicos Obrigatórios:

Análise dos padrões de acesso e Seleção justificada (ADR para cada DB).

Padrões de Configuração, Segurança (Hardening) e Operação para cada DB.

Análise de Overhead Operacional e Custo de Manutenção de cada tecnologia introduzida.

- **Ponto 44 - Governança de Dados:**
  - *Justificativa:* Estabelecer papéis (Owners), implementar Catálogo de Dados e monitorar métricas de qualidade de dados.

Linha de Raciocínio: Governança proativa e automatizada. Propriedade clara, qualidade mensurável (Data Quality Metrics), linhagem detalhada e gestão rigorosa de metadados. MDM para dados mestres.

Subtópicos Obrigatórios:

Definição de papéis (Data Owners, Stewards).

Implementação de Catálogo de Dados e Linhagem de Dados Detalhada (Detailed Data Lineage).

Estratégia de Master Data Management (MDM).

Métricas de Qualidade de Dados (Data Quality Metrics) e Monitoramento Contínuo.

Estratégia de Gerenciamento de Metadados (Metadata Management Strategy).

- **Ponto 46 - Estratégia de Mascaramento e Anonimização de Dados:**
  - *Justificativa:* Implementar mascaramento dinâmico/estático para proteger PII em logs e ambientes não produtivos.

Linha de Raciocínio: Privacidade por Design. Validar rigorosamente a eficácia da anonimização contra riscos de reidentificação. Considerar Privacidade Diferencial (Differential Privacy) para análises estatísticas seguras.

Subtópicos Obrigatórios:

Técnicas de mascaramento (Dinâmico/Estático) e Técnicas de anonimização.

Validação da eficácia da anonimização (Risco de Reidentificação).

Implementação de Privacidade Diferencial (Differential Privacy) (se aplicável).

- **Ponto 47 - Gerenciamento de Dados de Teste (TDM):**
  - *Justificativa:* Automatizar a geração de dados de teste seguros e representativos.

Linha de Raciocínio: Provisionamento rápido, seguro (sem PII real) e automatizado. O objetivo é uma plataforma de TDM Self-Service. O desafio é manter os dados de teste atualizados e representativos.

Subtópicos Obrigatórios:

Estratégia para geração de dados sintéticos ou subsets higienizados.

Provisionamento automatizado e sob demanda (Self-Service TDM Platform).

Estratégia de atualização (refresh) dos dados de teste.

- **Ponto 48 - Estratégia de Integridade de Dados:**
  - *Justificativa:* Implementar validação de qualidade de dados e garantir o uso correto de constraints de banco de dados no desenvolvimento. Considerar Ledger Imutável para auditoria crítica.

Linha de Raciocínio: Validação em todas as camadas (Integridade End-to-End). O monitoramento da consistência em sistemas distribuídos (Reconciliação Contínua) é crucial. Uso de tecnologias de Ledger Imutável (ex: QLDB) para dados de missão crítica.

Subtópicos Obrigatórios:

Implementação de checksums/assinaturas digitais (Integridade End-to-End).

Pipelines de validação de qualidade de dados e Constraints de DB.

Monitoramento de Consistência de Dados Distribuídos (Reconciliação Contínua).

Uso de Tecnologias de Ledger Imutável (Immutable Ledger Technologies) para dados críticos (ex: QLDB).

- **Ponto 52 - Estratégias de Escalabilidade de Dados:**
  - *Justificativa:* Implementar Read Replicas, otimizar Connection Pooling e planejar a estratégia de Sharding/Particionamento do PostgreSQL.

Linha de Raciocínio: Planejar o Sharding desde o início. A escolha da Chave de Partição é crítica e quase irreversível. Precisamos de estratégias para mitigação de Hotspots e um plano para o Re-sharding futuro. Gerenciamento de conexões (Connection Pooling) é vital em escala.

Subtópicos Obrigatórios:

Design da estratégia de Sharding e Configuração de Read Replicas.

Análise Detalhada da Seleção da Chave de Partição (Shard Key Analysis).

Mecanismos para lidar com Hotspots de dados (ex: Caching, Consistent Hashing).

Estratégia de Re-sharding/Rebalanceamento Futuro (Future Re-sharding Plan).

Estratégia de Connection Pooling detalhada.

- **Ponto 53 - Estratégia de Armazenamento de Dados (Hot/Warm/Cold):**
  - *Justificativa:* Implementar políticas de ciclo de vida de dados automatizadas para otimizar custos de armazenamento conforme o volume cresce.

  Linha de Raciocínio: Otimização de custos automatizada. A camada de acesso aos dados deve abstrair a localização física. Analisar a performance de recuperação de dados por tier.

  Subtópicos Obrigatórios:

  Definição dos critérios para classificação de dados e Políticas de Ciclo de Vida.

  Design da camada de acesso aos dados (Abstração de Localização).

  Análise de Performance de Recuperação de Dados por Tier (Data Retrieval Performance Analysis).

- **Ponto 55 - Ciclo de Vida dos Dados:**

- *Justificativa:* Implementar políticas de retenção e exclusão automatizada (TTL) para compliance e otimização de custos.

Linha de Raciocínio: Exclusão como requisito legal (LGPD) e de custo. O "Direito ao Esquecimento" deve ser auditável e propagado por todo o ecossistema (incluindo backups e logs). Gerenciar Legal Holds.

Subtópicos Obrigatórios:

Definição das políticas de retenção e Implementação de exclusão automatizada/TTL.

Processo para lidar com "Direito ao Esquecimento".

Trilhas de Auditoria para Exclusão de Dados.

Estratégia de Exclusão de Dados em Backups e Logs.

Estratégia de Legal Hold (Retenção Legal).

VI. Design de Frontend e Experiência do Usuário (UX/UI)

- **Ponto 58 - Design System e Componentização da UI:**
  - *Justificativa:* Iniciar a construção incremental do Design System para garantir consistência e reuso.

Linha de Raciocínio: O Design System é um produto interno. Requer governança contínua, versionamento rigoroso, documentação detalhada e métricas de adoção.

Subtópicos Obrigatórios:

Criação da biblioteca de componentes (ex: Storybook) e Tokens de design.

Processo de Governança e Versionamento do Design System.

Documentação de Uso e Diretrizes Detalhadas.

Métricas de Adoção e Eficácia do Design System.

- **Ponto 61 - Acessibilidade (A11y), i18n e L10n:**
  - *Justificativa:* Implementar e testar requisitos de acessibilidade (WCAG) e preparar para i18n se aplicável.

Linha de Raciocínio: Inclusão e prontidão global. Exige testes automatizados de acessibilidade e uma estratégia clara de gerenciamento de conteúdo multilíngue (Integração com CMS).

Subtópicos Obrigatórios:

Definição das diretrizes de acessibilidade (WCAG) e Estratégia de Testes Automatizados.

Seleção da biblioteca de i18n e processo de L10n.

Arquitetura para lidar com formatos e direções de escrita (RTL/LTR).

Estratégia de Gerenciamento de Conteúdo Multilíngue (CMS Integration).

VII. Infraestrutura e Deployment (DevOps/Cloud/SRE)

- **Ponto 64 - Design da Infraestrutura e Rede (Networking):**
  - *Justificativa:* Refinar o design da rede (VPCs, Subnets), implementar controle de Egress rigoroso e estratégia de CDN/Balanceamento de Carga.

  Linha de Raciocínio: Rede Zero Trust. Planejamento de IP (IPAM) rigoroso. Controle total sobre Ingress e Egress. Preparação mandatória para IPv6 em sistemas de larga escala.

  Subtópicos Obrigatórios:

  Design detalhado de VPCs, Subnets e Roteamento.

  Planejamento de IPAM/CIDR blocks e Conectividade Híbrida.

  Estratégia de CDN e Balanceamento de Carga (L4/L7).

  Design de Egress Control (Controle de Saída) rigoroso.

  Estratégia de Adoção de IPv6 (Dual Stack) mandatória.

- **Ponto 65 - Estratégia de Segmentação de Rede:**
  - *Justificativa:* Implementar segmentação rigorosa (Grupos de Segurança) e automação de regras (Security Groups as Code).

  Linha de Raciocínio: Pilar do Zero Trust. A micro-segmentação deve ser na camada de aplicação (Service Mesh, mTLS, Network Policies). A gestão de regras deve ser automatizada (Policy as Code). mTLS mandatória.

  Subtópicos Obrigatórios:

  Definição de Grupos de Segurança e NACLs.

  Implementação de Micro-segmentação.

  Design dos Pontos de Inspeção de Tráfego (Firewalls, IDS/IPS).

  Automação da Gestão de Regras de Firewall (Security Groups as Code).

Implementação de mTLS (Mutual TLS) mandatória para comunicação service-to-service.

- **Ponto 68 - Computação:**
  - *Justificativa:* Implementar Hardening da plataforma (CIS Benchmarks) e estratégias de Auto-Scaling avançadas.

  Linha de Raciocínio: Segurança da plataforma (Hardening) é crítica (CIS Benchmarks). Gestão segura de imagens (Golden Images, Assinatura). Alocação eficiente de recursos (Requests/Limits) e Auto-Scaling avançado (KEDA).

  Subtópicos Obrigatórios:

  Critérios de seleção (VMs, Containers, FaaS) e Configuração K8s.

  Estratégia de Gerenciamento de Imagens (Golden Images, Assinatura).

  Hardening da plataforma de computação (ex: CIS Benchmarks).

  Estratégia de Alocação de Recursos (Requests e Limits).

  Estratégia de Auto-Scaling Avançado (HPA, VPA, KEDA/Cluster Autoscaler).

- **Ponto 73 - Estratégias de Deployment:**
  - *Justificativa:* Evoluir o pipeline para suportar estratégias Zero Downtime (Blue/Green ou Canary).

Linha de Raciocínio: Zero downtime. Progressive Delivery (Canary) com análise automatizada baseada em métricas reais (SLOs) para rollback automático.

Subtópicos Obrigatórios:

Critérios para uso de Blue/Green e Canary.

Implementação de Progressive Delivery.

Automação da Análise de Canary (Automated Canary Analysis) (ex: Kayenta).

Estratégia de Rollback Automático baseado em Métricas (SLO-based Automatic Rollback).

- **Ponto 77 - Plano de Recuperação de Desastres (DR):**
  - *Justificativa:* Desenvolver o plano de DR detalhado e o Manual de Operações de Emergência.

  Linha de Raciocínio: O DR não é apenas técnico; cobre pessoas, processos e comunicação. Precisamos de um Manual de Operações de Emergência

claro.

Subtópicos Obrigatórios:

Identificação dos cenários de desastre.

Plano de ação técnico e de comunicação para DR.

Definição dos Papéis e Responsabilidades durante um DR (War Room/ICS).

Manual de Operações de Emergência (Emergency Operations Manual).

- **Ponto 78 - Estratégias de DR:**
  - *Justificativa:* Implementar a topologia de DR (ex: Pilot Light, Warm Standby) com base nos RPO/RTO definidos.

  Linha de Raciocínio: RPO e RTO definem a estratégia e o custo. Precisamos considerar a Recuperação Cibernética (Cyber Recovery), que exige Vaults Isolados (Air-Gapped) e processos de "Clean Room" para recuperação segura após um ataque de ransomware.

  Subtópicos Obrigatórios:

  Definição do RPO e RTO por serviço/criticidade.

  Seleção da topologia de DR (Pilot Light, Warm Standby, Active-Active).

  Estratégia de Recuperação Cibernética (Cyber Recovery Strategy) (Clean Room, Isolated/Air-Gapped Vault).

  Monitoramento do Lag de Replicação (Replication Lag Monitoring).

VIII. Qualidades Sistêmicas e Cross-Cutting Concerns (NFRs)

- **Ponto 83 - Estratégias de Mitigação de DDoS e Proteção contra Bots:**
  - *Justificativa:* Implementar WAF e proteção contra DDoS/Bots antes de escalar o tráfego significativamente.

  Linha de Raciocínio: Proteção multi-camada. Foco na proteção contra abuso da lógica de negócio (Business Logic Abuse) por bots sofisticados, além de ataques volumétricos.

  Subtópicos Obrigatórios:

  Implementação de WAF e Proteção contra DDoS.

  Estratégia de Detecção e Gerenciamento de Bots.

  Mecanismos de Proteção contra Abuso da Lógica de Negócio (Business Logic Abuse Prevention).

- **Ponto 85 - Conformidade Regulatória (Compliance):**
  - *Justificativa:* Automatizar a validação de conformidade (Policy as Code) e garantir a imutabilidade das trilhas de auditoria.

  Linha de Raciocínio: Conformidade contínua e automatizada (Compliance as Code / Policy as Code). A arquitetura deve garantir a imutabilidade das trilhas de auditoria.

  Subtópicos Obrigatórios:

  Mapeamento dos requisitos regulatórios aplicáveis.

  Design dos controles arquiteturais.

  Implementação de Policy as Code (ex: OPA) para validação contínua.

  Automação da Coleta de Evidências de Compliance.

  Design de Trilhas de Auditoria Imutáveis (Immutable Audit Trails).
- **Ponto 87 - Performance e Escalabilidade:**
  - *Justificativa:* Implementar estratégias de Caching detalhadas, Rate Limiting/Throttling e definir Orçamentos de Erro (Error Budgets).

  Linha de Raciocínio: Gerenciamento ativo de Orçamentos de Erro. Planejamento de capacidade quantitativo (Modelagem de Performance). Como o sistema lida com a contrapressão (Backpressure) e a latência de cauda longa (Tail Latency)?

  Subtópicos Obrigatórios:

  Definição de SLIs, SLOs e SLAs e Estimativas de Capacidade.

  Estratégias de Caching detalhadas e Rate Limiting/Throttling.

  Definição de Orçamentos de Erro (Error Budgets).

  Estratégia de Backpressure (Contrapressão).

  Modelagem de Performance (ex: Teoria das Filas).

  Estratégia de Otimização de Latência de Cauda Longa (Tail Latency Optimization).
- **Ponto 90 - Estratégia de Failover e Failback:**
  - *Justificativa:* Automatizar e testar os procedimentos de Failover/Failback, incluindo ressincronização de dados.

Linha de Raciocínio: O Failback e a ressincronização de dados são os desafios reais. Precisamos de mecanismos rigorosos para prevenir cenários de "Split-Brain" (ex: Fencing, Quorum).

Subtópicos Obrigatórios:

Mecanismos de detecção de falha e acionamento do Failover.

Procedimentos documentados e testados para o Failback.

Estratégia de Ressincronização de Dados Pós-Failback.

Mecanismos de Prevenção de Split-Brain (Split-Brain Prevention Mechanisms) (ex: Quorum, Fencing).

IX. Governança, Stacks e Documentação

- **Ponto 98 - Estratégia de Licenciamento:**
  - *Justificativa:* Integrar verificação de conformidade de licenças (SCA) e geração de SBOM no pipeline de CI/CD.

  Linha de Raciocínio: Risco legal e de segurança da cadeia de suprimentos. Conformidade automatizada (SCA) e geração de SBOM mandatória. Processo claro para avaliação (Vetting) de novas dependências e resposta rápida a vulnerabilidades.

  Subtópicos Obrigatórios:

  Políticas de Conformidade de Open Source.

  Ferramentas de verificação (SCA) integradas ao CI.

  Processo de Geração de SBOM (Software Bill of Materials).

  Processo de Vetting (Avaliação) de Novas Dependências e Resposta a Vulnerabilidades.

- **Ponto 102 - Estratégia de Testes de Performance:**
  - *Justificativa:* Implementar testes de carga/estresse contínuos no CI para validar SLOs e estabelecer baselines.

  Linha de Raciocínio: Validar os SLOs continuamente (Shift-Left). Testes realistas integrados ao CI em ambiente dedicado. Estabelecer baselines e alertas de regressão.

  Subtópicos Obrigatórios:

  Definição dos cenários de teste (Carga, Estresse, Picos, Endurance).

Seleção das ferramentas (ex: k6) e integração no CI.

Estabelecimento de Baselines de Performance e Alertas de Regressão.

Ambiente Dedicado e Isolado para Testes de Performance.

Fase 3: Rearquitetura Estratégica (O Horizonte de 1-2 anos)

Foco: Mudanças estruturais profundas e de alto custo/esforço. Projetos dedicados para evolução arquitetural significativa (ex: migração para microsserviços, MLOps, API Pública).

- **Ponto 3 - Estratégia de Plataforma:** (Se evoluir para PaaS/SaaS).

Linha de Raciocínio: Plataformas são ecossistemas. Exigem governança rigorosa, DX superior e isolamento robusto de tenants (Multi-tenancy). Como protegemos contra abusos e comportamentos inesperados (Noisy Neighbor)? Como incentivamos os efeitos de rede? Como gerenciamos a governança de dados na plataforma?

Subtópicos Obrigatórios:

Definição do modelo e Capacidades "Core".

Design inicial da DX da plataforma (Onboarding, Sandbox).

Modelo de Governança da Plataforma e Modelo de Extensibilidade.

Estratégia de Isolamento de Tenants Detalhada (Lógico vs. Físico, mitigação de Noisy Neighbor).

Mecanismos de Proteção da Plataforma (Throttling granular, Quotas).

Estratégia de Ecossistema, Efeitos de Rede e Modelo de Incentivo.

Modelo de Governança de Dados da Plataforma (Platform Data Governance Model).

- **Ponto 13 - Padrões Arquiteturais Especializados:** (Big Data/IoT).

Linha de Raciocínio: Arquiteturas especializadas (IoT, Big Data) devem ser integradas sem criar silos operacionais ou de segurança. O desafio é a governança e o modelo de segurança unificados.

Subtópicos Obrigatórios:

(IoT) Arquitetura detalhada de Edge e Ingestão. (Big Data) Arquitetura Lambda/Kappa.

Definição das interfaces de integração com o sistema Core.

Estratégia Operacional e de Governança Unificada.

Modelo de Segurança Unificado (Unified Security Model) para arquiteturas heterogêneas.

- **Ponto 15 - Padrões de Interoperabilidade:** (Padrões complexos como Open Finance).

Linha de Raciocínio: Isolar a conformidade em Adaptadores/Gateways. O desafio é o Mapeamento Semântico complexo e a validação contínua e automatizada da conformidade.

Subtópicos Obrigatórios:

Identificação dos padrões aplicáveis (ex: FHIR, PSD2).

Design dos adaptadores e gateways.

Estratégia de Testes de Conformidade.

Mapeamento Semântico Detalhado (Semantic Mapping).

- **Ponto 17 - Estratégia de Distribuição Geográfica:** (Multi-Região Ativo-Ativo).

Linha de Raciocínio: Multi-região é extremamente complexo (Teorema CAP). A modelagem de consistência de dados geo-distribuídos é o desafio central (CRDTs podem ser necessários). Edge Computing pode ser uma alternativa. Análise detalhada do impacto da latência da rede global.

Subtópicos Obrigatórios:

Estratégia de replicação de dados (Ativo-Ativo vs. Ativo-Passivo).

Tecnologias de roteamento geográfico (Geo-DNS) e GSLB.

Estratégia de CDN e caching de borda.

Modelagem de Consistência de Dados Geo-distribuídos (ex: CRDTs - Conflict-free Replicated Data Types).

Estratégia de Edge Computing (se aplicável).

Análise de Impacto da Latência da Rede Global (Global Network Latency Impact Analysis).

- **Ponto 26 - Padrões Arquiteturais Avançados (ES/CQRS):** (Alta complexidade, uso cirúrgico).

Linha de Raciocínio: Padrões de alta complexidade, uso cirúrgico. Os desafios técnicos são a evolução de eventos (Event Versioning/Upcasting), a gestão da

consistência eventual e a estratégia de reconstrução de projeções (Replay).

Subtópicos Obrigatórios:

Identificação justificada dos Bounded Contexts.

Design do Event Store e estratégia de Snapshots.

Design dos Modelos e estratégia de projeção.

Estratégia de Versionamento de Eventos (Event Versioning/Upcasting) detalhada.

Mecanismos para lidar com a Consistência Eventual nas Projeções.

Estratégia de Replay de Eventos e Reconstrução de Projeções (Projection Rebuilding Strategy).

- **Ponto 27 - Arquitetura para IA e Machine Learning (MLOps):** (Plataforma de ML para Scoring).

Linha de Raciocínio: Operacionalizar ML com rigor de engenharia. Foco em reprodutibilidade, monitoramento de drift e governança. O Feature Store é central. A segurança contra Ataques Adversariais (Adversarial ML) e o Feedback Loop são cruciais para sistemas de missão crítica.

Subtópicos Obrigatórios:

Design do pipeline de dados e Feature Store.

Estratégia de deployment de modelos (Shadow Mode, A/B Testing).

Monitoramento de Drift (Concept e Data Drift).

Estratégia de Versionamento de Modelos e Dados.

Governança do Modelo (Model Governance) e Auditoria.

Estratégia de Segurança contra Ataques Adversariais (Adversarial ML Defense).

Mecanismos de Feedback Loop para retreinamento contínuo.

- **Ponto 43 - Arquitetura de Dados Analíticos (OLAP):** (Data Mesh/Lakehouse).

Linha de Raciocínio: Separar OLTP de OLAP. A arquitetura (Warehouse, Lake, Mesh) deve garantir qualidade. A introdução de "Contratos de Dados" (Data Contracts) é essencial para confiabilidade e governança em escala (Data Mesh).

Subtópicos Obrigatórios:

Seleção da plataforma (DW/Lake/Mesh) e Design dos pipelines (ETL/ELT).

Estratégia de Modelagem Analítica e Arquitetura para Streaming Analytics.

Framework de Qualidade de Dados na Ingestão.

(Data Mesh) Definição de Produtos de Dados e Governança Federada.

Implementação de Contratos de Dados (Data Contracts) como mecanismo de governança.

- **Ponto 49 - Residência de Dados e Soberania:** (Expansão global com requisitos legais complexos).

Linha de Raciocínio: Requisitos legais ditam a localização física. Impacta a arquitetura global. Precisamos de controles de acesso baseados em localização (Geo-fencing) e estratégias para Soberania Digital (ex: gestão de chaves externa - External Key Management).

Subtópicos Obrigatórios:

Mapeamento dos requisitos legais por região e Arquitetura multi-regional.

Políticas de Roteamento de Requisições baseadas na origem.

Controles de Acesso Baseados em Localização (Geo-fencing).

Estratégia de Soberania Digital (Digital Sovereignty Strategy) (ex: External Key Management).

- **Ponto 50 - Modelos de Consistência (Avançados):** (ex: CRDTs para Ativo-Ativo).

Linha de Raciocínio: O trade-off fundamental (Teorema CAP). A complexidade da Consistência Eventual exige mecanismos de mitigação (Read-Your-Writes). Considerar modelos avançados como CRDTs para cenários Ativo-Ativo e analisar a convergência de dados.

Subtópicos Obrigatórios:

Análise dos requisitos de negócio por domínio e Justificativa (ADR).

Mecanismos para lidar com a Consistência Eventual (Read-Your-Writes, Session Consistency).

Análise de Convergência de Dados (Data Convergence Analysis).

Avaliação de Modelos de Consistência Avançados (ex: CRDTs).

- **Ponto 54 - Arquitetura de Arquivamento de Longo Prazo:** (WORM para retenção regulatória).

Linha de Raciocínio: Retenção regulatória exige durabilidade e imutabilidade (WORM). Garantir a legibilidade futura (formato) e a integridade dos dados ao longo de décadas (Verificação Periódica).

Subtópicos Obrigatórios:

Seleção de tecnologias de armazenamento e Processo de recuperação.

Estratégias para garantir a imutabilidade (WORM).

Seleção de Formatos de Dados de Longa Duração (ex: Parquet, XML).

Verificação Periódica da Integridade dos Dados Arquivados (Periodic Integrity Checks).

- **Ponto 70 - Estratégia de Descoberta de Serviços:** (Se migrar para Microsserviços).

Linha de Raciocínio: Descoberta dinâmica integrada aos Health Checks. Considerar balanceamento de carga baseado em localidade para otimizar latência e custo de tráfego.

Subtópicos Obrigatórios:

Escolha entre Descoberta Client-side e Server-side.

Integração com a plataforma ou Service Mesh e Health Checks.

Políticas de Balanceamento de Carga Local (Locality Load Balancing).

- **Ponto 75 - Componentes de Infraestrutura Distribuída (Service Mesh):** (Alta complexidade operacional).

Linha de Raciocínio: API Gateway (Norte-Sul). Service Mesh (Leste-Oeste) para mTLS e resiliência, mas sua complexidade operacional deve ser cuidadosamente avaliada e justificada.

Subtópicos Obrigatórios:

Configuração do API Gateway.

Implementação do Service Mesh (Istio, Linkerd) e Avaliação da Complexidade Operacional vs. Benefício.

Configuração do Message Broker (Alta Disponibilidade e Durabilidade).

- **Ponto 86 - Considerações Éticas e IA Responsável:** (XAI e Auditoria de Algoritmos para ML).

Linha de Raciocínio: Risco reputacional e legal. Garantir transparência (XAI), justiça (Fairness) e responsabilidade. Estabelecer um processo de revisão ética formal e auditoria de algoritmos. Considerar o impacto social do sistema.

Subtópicos Obrigatórios:

Implementação de mecanismos de explicabilidade (XAI).

Estratégias para detecção e mitigação de vieses algorítmicos (Fairness).

Mecanismos de Feedback e Contestação do Usuário.

Processo de Revisão Ética Formal e Auditoria de Algoritmos.

Análise de Impacto Social do Sistema (Social Impact Analysis).

- **Ponto 91 - Arquitetura para Experimentação (A/B Testing):** (Plataforma de experimentação robusta).

Linha de Raciocínio: Habilitar inovação baseada em dados (Testes A/B). O desafio é garantir o isolamento dos experimentos (prevenção de interferência) e a validade estatística dos resultados. Governança rigorosa dos experimentos.

Subtópicos Obrigatórios:

Seleção da plataforma de Testes A/B e Roteamento dinâmico.

Mecanismos de Isolamento de Experimentos e Prevenção de Interferência.

Pipeline de Análise Estatística dos Experimentos.

Framework de Governança de Experimentos.

- **Ponto 104 - Estratégia de Testes de Resiliência (Chaos Engineering):** (Implementação formal e contínua).

Linha de Raciocínio: Validar a resiliência na prática (Game Days). Definir Hipóteses de Estado Estável e controlar o raio de explosão (Blast Radius). Automatizar experimentos contínuos.

Subtópicos Obrigatórios:

Seleção da plataforma de Chaos Engineering.

Definição dos experimentos controlados e do raio de explosão (Blast Radius).

Hipóteses de Estado Estável (Steady State Hypotheses) a serem validadas.

Roadmap de Engenharia de Caos e Processo de Game Days.

Automação de Experimentos de Caos Contínuos (Continuous Chaos Testing).

- **Ponto 105 - Estratégia de Migração de Dados:** (Migrações complexas associadas à rearquitetura).

Linha de Raciocínio: O passo mais arriscado. Zero perda de dados, mínimo downtime. Reconciliação detalhada é mandatória. Estratégias incrementais (Parallel Run/CDC) são preferíveis ao Big Bang. Plano de Rollback testado rigorosamente.

Subtópicos Obrigatórios:

Design do pipeline de migração (ETL).

Estratégia de validação e reconciliação de dados.

Plano de Cutover de dados (Big Bang vs. Parallel Run/Trickle Migration).

Plano de Rollback da Migração de Dados detalhado e testado rigorosamente.

- **Ponto 106 - Estratégia de Integração e Descomissionamento de Legados:** (Padrão Strangler Fig).

Linha de Raciocínio: Modernização incremental (Strangler Fig Pattern). ACLs rigorosas para isolamento. O descomissionamento deve ser planejado ativamente com critérios de sucesso claros e métricas de progresso.

Subtópicos Obrigatórios:

Implementação do padrão Strangler Fig e definição das fachadas/ACLs.

Plano de descomissionamento (Sunset Plan).

Métricas para acompanhar o progresso e Critérios de Sucesso

Fase 4: Evolução Contínua (O Sempre)

Foco: Processos e práticas contínuas que devem se tornar parte da cultura de engenharia. Não têm fim definido.

- **Ponto 2 - Modelagem Financeira e TCO:** (Análise contínua de custos e ROI).

Linha de Raciocínio: A arquitetura define a estrutura de custos futura. Rigor na "Unit Economics" (custo marginal). Como os custos escalam em cenários extremos? Qual é o Custo do Atraso (Cost of Delay)? Podemos usar a Análise de Opções Reais (Real Options Analysis) para fasear investimentos arquiteturais de alto risco/alto retorno? Qual o custo da complexidade técnica?

Subtópicos Obrigatórios:

Detalhamento dos custos (3-5 anos) e Análise de ROI/break-even point.

Modelagem de cenários de custo (FinOps inicial).

Definição rigorosa da Economia Unitária (Unit Economics).

Análise de Sensibilidade de Custos e Modelagem de Custo sob Cargas de Pico (Spike Loads).

Análise do Custo do Atraso (Cost of Delay).

Modelo de Atribuição de Custos (Cost Attribution Model) detalhado.

Análise de Opções Reais (Real Options Analysis) para investimentos faseados de alto risco.

Quantificação do Custo da Complexidade Técnica (Cost of Technical Complexity).

- **Ponto 4 - Critérios de Seleção de Tecnologia (Build vs. Buy):** (Avaliação contínua de tecnologias e fornecedores).

Linha de Raciocínio: Focar no Core, comprar o Context. Avaliar o risco de vendor lock-in e o Custo de Substituição Futura. A estratégia de saída deve ser definida antes da entrada. A segurança da cadeia de suprimentos (Supply Chain Security) é um risco estratégico primário. Due Diligence rigorosa do fornecedor.

Subtópicos Obrigatórios:

Mapeamento de Domínios Core vs. Context.

Matriz de decisão ponderada e Processo de PoC.

Análise de Risco de Fornecedor e Estratégia de Saída (Exit Strategy).

Análise do Custo de Substituição Futura.

Análise de Risco da Cadeia de Suprimentos (Supply Chain Risk Analysis).

Due Diligence Técnica, de Segurança e Financeira do Fornecedor (Vendor Due Diligence).

- **Ponto 10 - Análise de Trade-offs e Riscos:** (Gestão de riscos contínua - FMEA, SPOFs).

Linha de Raciocínio: Gestão de riscos rigorosa. Usar FMEA para análise proativa de falhas. Identificar decisões irreversíveis (Tipo 1). Como o sistema lida com eventos imprevisíveis de alto impacto (Black Swan events) – buscar Antifragilidade (ganhar com a desordem).

Subtópicos Obrigatórios:

Matriz de Risco e Estratégias de mitigação (PoCs, Spikes).

Análise de Pontos Únicos de Falha (SPOFs) (Técnico e Humano).

Documentação dos trade-offs aceitos.

Análise de Modos de Falha e Efeitos (FMEA) arquitetural detalhada.

Análise de Latência de Decisão (Decisões Tipo 1 vs. Tipo 2).

Análise de Risco de Eventos "Black Swan" (Cisne Negro) e estratégias de Antifragilidade.

- **Ponto 11 - Estrutura Organizacional (Lei de Conway):** (Monitoramento contínuo do alinhamento socio-técnico).

Linha de Raciocínio: Projetar a organização intencionalmente (Inverse Conway Maneuver) para suportar a arquitetura. O foco deve ser otimizar o fluxo de entrega e gerenciar ativamente a Carga Cognitiva da equipe. Planejar a evolução organizacional.

Subtópicos Obrigatórios:

Definição da topologia das equipes (Team Topologies) e Modos de Interação.

Mapeamento de Propriedade (Ownership Mapping) dos Bounded Contexts.

Avaliação e Plano de Gestão Ativa da Carga Cognitiva da Equipe (Team Cognitive Load Management).

Plano de Evolução Organizacional (Organizational Evolution Plan) para suportar a arquitetura futura.

- **Ponto 57 - Fluxos de Telas e Wireframes:** (Otimização contínua da jornada do usuário).

Linha de Raciocínio: Otimizar a performance percebida. Estratégias de carregamento de dados (Prefetching) e UI Otimista devem ser alinhadas à jornada do usuário. Design de estados de erro e carregamento.

Subtópicos Obrigatórios:

Diagramas de Jornada do Usuário e Protótipos validados.

Design da Arquitetura de Informação (IA) e Navegação.

Estratégias de Carregamento de Dados (Prefetching, Lazy Loading).

Estratégia de UI Otimista (Optimistic UI) e Tratamento de Erros.

Design de Estados de Carregamento (Skeleton Screens) e Erro (Error States).

- **Ponto 66 - Estratégia de DNS e Gerenciamento de Domínios:** (Gestão operacional contínua).

Linha de Raciocínio: DNS é crítico. Automação total de certificados. A segurança do DNS (DNSSEC) é mandatória para sistemas críticos.

Subtópicos Obrigatórios:

Configuração de zonas de DNS (Split-Horizon DNS).

Estratégia de gerenciamento e renovação automatizada de certificados.

Definição da convenção de nomenclatura de domínios.

Implementação de DNS Security (DNSSEC).

- **Ponto 79 - Estratégia de Testes de Recuperação de Desastres (DR Testing):** (Game Days regulares).

Linha de Raciocínio: Um plano não testado é inútil. Os testes (Game Days) devem ser realistas e regulares, incluindo cenários de recuperação de ataques cibernéticos.

Subtópicos Obrigatórios:

Planejamento de simulações de desastre regulares (Game Days).

Automação dos testes de DR.

Análise pós-teste e plano de melhoria contínua do DR.

Definição dos Cenários de Teste de DR (incluindo Cyber Attack Recovery Testing).

- **Ponto 84 - Operações de Segurança (SecOps):** (Monitoramento, resposta e Threat Hunting contínuos).

Linha de Raciocínio: Visibilidade centralizada (SIEM) e resposta automatizada (SOAR). Integrar inteligência de ameaças (Threat Intelligence) e realizar Threat Hunting proativo.

Subtópicos Obrigatórios:

Arquitetura para Coleta e Análise de Logs de Segurança e Auditoria.

Integração com SIEM e SOAR.

Definição de Casos de Uso de Monitoramento e Playbooks de Resposta.

Integração de Inteligência de Ameaças (Threat Intelligence Integration).

Estratégia de Threat Hunting (Caça a Ameaças) Proativa.

- **Ponto 89 - Plano de Continuidade de Negócios (BCP):** (Testes e revisões regulares - Tabletop Exercises).

Linha de Raciocínio: Foco na resiliência dos processos de negócio. O BCP deve ser testado regularmente com simulações de crise (Tabletop Exercises) envolvendo stakeholders de negócio.

Subtópicos Obrigatórios:

Análise de Impacto no Negócio (BIA).

Definição arquitetural de modos de operação degradados ou alternativos.

Plano de Comunicação de Crise (Interno e Externo).

Planejamento de Simulações de Continuidade de Negócios (BCP Simulations / Tabletop Exercises).

- **Ponto 94 - Otimização de Custos (FinOps):** (Prática cultural contínua de otimização de custos).

Linha de Raciocínio: FinOps como prática contínua. Visibilidade (Tagging rigoroso) e responsabilidade (Showback/Chargeback). Otimização automatizada (Right-Sizing). Detecção de Anomalias de Custo.

Subtópicos Obrigatórios:

Estratégia de tagging de recursos (Cost Allocation Tags).

Monitoramento de custos e alertas de orçamento.

Políticas de Right-Sizing de recursos.

Dashboards de Custo por Serviço/Produto.

Estratégia de Alocação de Custos e Showback/Chargeback.

Detecção de Anomalias de Custo (Cost Anomaly Detection).

- **Ponto 95 - Sustentabilidade (GreenOps):** (Otimização contínua da eficiência energética).

Linha de Raciocínio: Eficiência energética como objetivo de design. Adotar princípios de Design de Software Consciente de Carbono (Carbon-Aware Software Design).

Subtópicos Obrigatórios:

Escolha de regiões de nuvem com menor pegada de carbono.

Otimização da utilização de recursos (maximização da densidade).

Adoção de arquiteturas eficientes (Serverless, ARM).

Métricas de Pegada de Carbono da Infraestrutura.

Adoção de Princípios de Design de Software Consciente de Carbono (Carbon-Aware Software Design).

- **Ponto 107 - Estratégia de Evolução Arquitetural e Gestão de Dívida Técnica:** (Gestão proativa da dívida e Fitness Functions).

Linha de Raciocínio: Combater a entropia. Monitoramento objetivo (Fitness Functions). Gestão proativa da dívida técnica com orçamento dedicado. Automação da governança arquitetural. Quantificar a dívida técnica e seu ROI de pagamento.

Subtópicos Obrigatórios:

Implementação de Fitness Functions (Monitoramento Contínuo).

Processo formal para Gestão de Dívida Técnica.

Automação da Governança Arquitetural (ex: ArchUnit, Quality Gates).

Registro de Dívida Técnica (Technical Debt Registry) e Orçamento.

Quantificação da Dívida Técnica e Análise de ROI para Pagamento