



# Plano Estratégico de Preparação

Pessoa Desenvolvedora Android Sênior · GOK | Inovação Digital

## Visão Geral da Estratégia

O perfil técnico de Jean é genuinamente forte para esta vaga. **O desafio não é competência — é apresentação.**

A preparação deve atuar em três frentes simultâneas:

1. **Fechar lacunas técnicas críticas** exigidas explicitamente pela vaga: Clean Architecture, MVI, StateFlow e modularização
2. **Defender o histórico existente com profundidade arquitetural** — especialmente Monnos, Leega e eclipseworks
3. **Construir narrativas comportamentais estruturadas** ancoradas nas experiências reais em fintech, com foco em ownership, decisão e responsabilidade

**Atenção:** O maior risco não é Jean não saber — é chegar à entrevista sem conseguir articular *por que* tomou as decisões que tomou.

## Leitura do Contexto da Vaga

A GOK não busca um executor Android. Busca uma **referência técnica** capaz de tomar decisões arquiteturais com autonomia e se responsabilizar pelo ciclo completo — da concepção à sustentação em produção.

O ambiente é consultivo, com projetos de alta criticidade transacional em fintech, banking e varejo digital. Isso implica adaptação a contextos distintos, lidar com código legado e entregar sob pressão sem abrir mão da qualidade.

### O que isso significa na prática:

- Perguntas sobre trade-offs arquiteturais serão frequentes e aprofundadas
- Cenários de falha em produção serão usados para testar raciocínio real
- Não basta saber a resposta certa — é preciso explicar o raciocínio por trás dela
- O peso comportamental equivale ao técnico

## Aderência Perfil x Vaga

**Score geral: 79/100 — Alta aderência, com lacunas de apresentação mais do que de experiência real.**

Jean já possui o que a vaga mais valoriza: experiência em sistemas financeiros críticos, liderança técnica Android, stack moderna e segurança mobile avançada. As lacunas existem, mas são manejáveis.

| Situação                              | Competências  |
|---------------------------------------|---|
| ☑ Forte e bem demonstrado             | Kotlin, MVVM, Jetpack Compose, Hilt, Coroutines, Room, Retrofit, segurança mobile, Play Store                   |
| ⚠ Forte, mas mal demonstrado          | Ownership técnico, decisões arquiteturais, liderança — presentes na trajetória, mas pouco visíveis no currículo |
| ☐ Lacuna real — estudo necessário     | Clean Architecture (formal), MVI, StateFlow/Flow, modularização   |
| ☐ Lacuna real — prioridade secundária | CI/CD mobile, observabilidade, feature flags  |
| ☐ Risco de rejeição                   | Descrição fraca da Monnos, ausência de evidências concretas de testes, ATS score baixo                          |

**Dica estratégica:** As lacunas mais críticas — Clean Architecture, MVI e StateFlow — têm alta chance de já existirem de forma implícita na trajetória de Jean. O que falta é o vocabulário correto e a capacidade de articular com profundidade na entrevista.

---

# Tópicos Prioritários de Estudo

## 1 • Clean Architecture e Separação de Camadas

**Por que importa para esta vaga:** A descrição da vaga cita Clean Architecture como requisito explícito. Em ambientes críticos, quem não domina esse padrão não conduz refatorações com segurança.

**Profundidade esperada:** Saber implementar *e* defender. O entrevistador vai perguntar *por que* você separou assim, *como* testou cada camada e *o que acontece* quando o backend muda.

**Status no currículo:** Não mencionado — ausência que pode gerar descarte automático no ATS.

**Erro comum:** Confundir Clean Architecture com MVVM. São padrões complementares, não equivalentes. Saber essa distinção é o mínimo esperado de um sênior.

Outros erros frequentes:

- Não saber explicar o papel de um Use Case
- Não conseguir justificar trade-offs da separação de camadas em projetos menores
- Usar os termos corretos sem conseguir aplicar em código real

---

## 2 • MVI e Gerenciamento de Estado

**Por que importa para esta vaga:** MVI é o padrão mais adequado para estados complexos em alta volumetria transacional — exatamente o contexto da GOK.

**Profundidade esperada:** Implementar com StateFlow/SharedFlow, entender o fluxo unidirecional de dados (UDF), saber quando MVI supera o MVVM e vice-versa, e comparar os dois com argumentos reais de projeto.

**Status no currículo:** Não mencionado. O uso de LiveData sinaliza desatualização para o nível exigido.

**Erro comum:** Reduzir MVI a “usar Intent como sealed class.” O entrevistador espera que você entenda UiState como snapshot imutável, side effects via SharedFlow e as implicações de ciclo de vida do StateFlow versus LiveData.

### 3 · Coroutines, Flow e Concorrência Kotlin

**Por que importa para esta vaga:** Em ambientes transacionais, concorrência mal gerenciada causa crashes, race conditions e inconsistências de estado — os problemas que a GOK precisa garantir que não aconteçam.

**Profundidade esperada:** Domínio real. Diferenciar `launch` de `async`, entender structured concurrency, cancelar coroutines corretamente, explicar dispatchers, diferenciar `Flow`, `StateFlow` e `SharedFlow`, e tratar exceções com `CoroutineExceptionHandler` e `supervisorScope`.

**Status no currículo:** Jean menciona Coroutines em Leega — ponto forte. O risco está na profundidade: LiveData ainda aparece como padrão e StateFlow/Flow estão ausentes, o que pode sugerir uso superficial.

**Erro comum:** Confundir Cold Flow com Hot Flow (StateFlow/SharedFlow), ou não saber lidar com exceções em `async / await` — que exigem `try/catch` explícito, ao contrário de `launch`.

### 4 · Resiliência no Cliente Android e Tratamento de Erros

**Por que importa para esta vaga:** A vaga cita explicitamente tratamento de erros, resiliência e integração com BFF/API Gateway. Em operações financeiras, falha de rede mal tratada pode causar double charges, inconsistência de dados e experiência degradada.

**Profundidade esperada:** Retry com backoff exponencial, sealed classes para modelar estados de erro, fallback para dados locais via Room, tratamento de timeouts e respostas parciais. Saber como o app deve se comportar quando o backend está degradado.

**Status no currículo:** Parcialmente demonstrado — o SDK do Banco Inter e o SDK de Segurança do Banco Master implicam robustez no tratamento de erros, mas nenhuma

experiência verbaliza essa estratégia.

**Dica estratégica:** Jean tem a experiência — o que falta é a narrativa. A pergunta “como você implementa resiliência no cliente quando o backend falha?” já tem resposta na trajetória. Precisa apenas ser articulada.

## 5 • Decisões Arquiteturais e Ownership Técnico

**Por que importa para esta vaga:** A GOK quer alguém que assume ownership — não um executor. O entrevistador técnico vai testar diretamente a capacidade de justificar decisões, lidar com ambiguidade e influenciar o time.

**Profundidade esperada:** Narrativas STAR concretas. Não *“participei da arquitetura”* — mas *“identifiquei o problema X, propus a solução Y pelos trade-offs Z, conduzi a implementação e o resultado foi W.”*

**Status no currículo:** Monnos, Singu e Cedro têm potencial enorme, mas as descrições são genéricas demais para convencer um entrevistador de nível sênior.

**Erro comum:** Falar do que o time fez em vez do que Jean fez especificamente. Ou mencionar o *o quê* sem explicar o *porquê* da decisão.

## Lacunas e Plano de Ação

### □ Alta Prioridade — Fechar antes da entrevista

| Lacuna             | Ação necessária   |
|--------------------|---|
| Clean Architecture | Estudar, implementar e explicar Use Cases, Repositories e separação de camadas. Verificar se projetos anteriores já tinham essa estrutura de forma implícita. |

| Lacuna                      | Ação necessária   |
|-----------------------------|---|
| <b>MVI + StateFlow</b>      | Implementar um exemplo com StateFlow para estado e SharedFlow para side effects. Preparar comparação argumentada com MVVM.  |
| <b>Narrativa da Monnos</b>  | Reconstruir a história dos 4 anos e 9 meses com decisões arquiteturais reais, escala do produto e resultados concretos. É a experiência mais poderosa do currículo — e está sendo desperdiçada. |
| <b>Testes automatizados</b> | Aprofundar JUnit, MockK e Espresso. Se necessário, criar um projeto de referência com testes escritos para ter exemplos concretos na entrevista.  |

## □ Média Prioridade — Preparar para não ser pego desprevenido

| Lacuna                                  | Ação necessária   |
|---|---|
| <b>Modularização</b>                    | Entender feature modules e módulos de dados e UI. Saber argumentar estratégia e benefícios, mesmo sem projeto real.       |
| <b>StateFlow vs LiveData</b>            | Preparar resposta clara sobre quando e por que migrar. Conhecer as diferenças técnicas reais.                             |
| <b>Performance e memória</b>            | Estudar Android Profiler, tipos de memory leaks (contexto vazado, listeners não removidos) e como investigar ANRs e jank. |
| <b>Integração com BFF e API Gateway</b> | Entender o padrão BFF e como o cliente deve se comportar nesse contexto. Conectar com a experiência do Banco Inter.       |

## □ Baixa Prioridade — Ter noção, não aprofundar agora

| Lacuna                               | Postura recomendada   |
|--------------------------------------|---|
| <b>CI/CD mobile</b>                  | <i>"Minha experiência foi mais no lado backend. Conheço Fastlane e GitHub Actions e quero evoluir nisso."</i> |
| <b>Observabilidade (Crashlytics)</b> | Saber o que é, como se configura e como analisar crashes. Mencionar se usou em algum projeto pessoal.         |

| Lacuna        | Postura recomendada  |
|---------------|--|
| Feature flags | Entender o conceito e Firebase Remote Config. Não é crítico para a entrevista. |

## Referências de Estudo

### Gratuitas

#### Clean Architecture

| Recurso  | Por que usar   |
|--|--|
| <a href="#">Guide to App Architecture — Android Developers</a> | Documentação oficial do Google. Cobre separação em camadas UI, Domain e Data com exemplos reais em Kotlin. Ponto de partida obrigatório. |
| <a href="#">Now in Android — Google (GitHub)</a>               | App de produção real com Clean Architecture, MVI, modularização, Hilt e Compose. Ler o código vale mais do que qualquer tutorial.        |

#### MVI e StateFlow

| Recurso   | Por que usar  |
|---|---|
| <a href="#">StateFlow and SharedFlow — Android Developers</a> | Documentação oficial. Casos de uso, diferenças com LiveData e exemplos de implementação. Leitura obrigatória. |
| <a href="#">Unidirectional Data Flow — Google I/O 2021</a>    | Talk oficial explicando UDF com StateFlow, Compose e MVVM/MVI. Direto ao ponto.                               |

#### Coroutines e Flow

| Recurso   | Por que usar   |
|---|--|
| <a href="#">Kotlin Coroutines — documentação oficial</a>                                      | Fonte primária. Structured concurrency, dispatchers, cancelamento, exceções e Flow.                |
| <a href="#">Coroutines &amp; Patterns for work that shouldn't be cancelled — Android Blog</a> | Padrões para operações não canceláveis — exatamente o cenário de ambientes transacionais críticos. |

## Resiliência e Tratamento de Erros

| Recurso   | Por que usar  |
|---|---|
| <a href="#">Android Vitals — Android Developers</a> | ANRs, crashes e como construir apps resilientes. Métricas que a GOK provavelmente usa internamente. |
| <a href="#">Sealed classes — Kotlin Docs</a>        | Fundamento técnico para modelar estados de erro com Result e Either de forma idiomática.            |

## Decisões Arquiteturais

| Recurso   | Por que usar  |
|---|---|
| <a href="#">Android Architecture Blueprints — Google (GitHub)</a> | Mesmo app em diferentes arquiteturas (MVVM, MVI). Ideal para construir argumentos de trade-off.                     |
| <a href="#">Engineering Practices — Google</a>                    | Guia de code review do Google. Prepara respostas sobre como Jean conduz revisões — tema provável na comportamental. |

## Premium

| Recurso  | Por que vale o investimento  |
|--|--|
| 📖 <i>Clean Architecture</i> — Robert C. Martin | Fonte primária do conceito. Leia os capítulos sobre dependências e camadas (Parte V) — suficiente para a preparação. |



| Recurso  | Por que vale o investimento   |
|--|---|
| 📺 <b>Android Architecture Masterclass — Philipp Lackner (Udemy)</b>                  | Clean Architecture, MVI, Coroutines, Flow e Hilt em projeto real. Cobre exatamente o gap entre o currículo atual e o que a vaga exige.  |
| 📺 <b>Kotlin Coroutines: Deep Dive — Marcin Moskala</b>                               | Vai além do básico: structured concurrency, Flow avançado, testes e padrões de produção. Separa quem “usa” coroutines de quem as entende de verdade.                              |
| 📺 <b>Android Testing with JUnit, Mockito &amp; Espresso — Catalin Stefan (Udemy)</b> | Focado no trio que a vaga implica. Fornece a base prática para falar sobre testes com exemplos concretos.   |
| 📺 <b>The Staff Engineer's Path — Tanya Reilly</b>                                    | Descreve exatamente o perfil que a GOK busca. Os 3 primeiros capítulos ajudam a articular ownership e influência técnica com vocabulário que ressoa com entrevistadores sêniores. |

## Cronograma de Preparação

**Premissa:** Este plano assume de 2 a 3 semanas antes da entrevista técnica. **Se o prazo for menor, concentre-se exclusivamente na Fase 1 e nas narrativas comportamentais.**

### Fase 1 — Revisão Estratégica

**Duração recomendada:** 4 a 5 dias

**Objetivo:** Mapear o que Jean já domina, conectar esse conhecimento com a vaga e preencher lacunas conceituais com teoria direcionada. Foco principal nas três experiências mais relevantes para a GOK: **Monnos, Leega e eclipseworks**.

- [ ] Narrativa completa da Monnos: decisões arquiteturais tomadas, escala do produto, resultados e como conduzia o time

- [ ] Narrativa do SDK Banco Inter: desafios técnicos reais, tratamento de falhas e garantia de qualidade
  - [ ] Narrativa do SDK de Segurança Banco Master: justificativa para Rust, JNA e os riscos considerados
  - [ ] Leitura da documentação oficial: Guide to App Architecture + StateFlow e SharedFlow
  - [ ] Comparação escrita (para si mesmo): MVVM vs MVI — quando usar cada um, com argumentos reais
- 

## Fase 2 — Prática Técnica Direcionada

**Duração recomendada:** 7 a 10 dias

**Objetivo:** Implementar Clean Architecture e MVI em código real. O projeto pessoal

**Palpite do Dia** é o laboratório ideal — refatorar uma feature existente é mais valioso do que criar um projeto do zero.

- [ ] Feature implementada com Clean Architecture: Use Case, Repository e camada de dados separada
  - [ ] Feature implementada com MVI: UiState, Intent, StateFlow para estado e SharedFlow para side effects
  - [ ] Mínimo de 5 testes unitários com JUnit e MockK cobrindo um Use Case e um ViewModel
  - [ ] Leitura do repositório Now in Android (foco em modularização e arquitetura de camadas)
  - [ ] Revisão de Coroutines: structured concurrency, tratamento de exceções e dispatchers
- 

## Fase 3 — Refino de Comunicação e Posicionamento

**Duração recomendada:** 3 a 4 dias

**Objetivo:** Transformar domínio técnico em comunicação clara e confiante sob pressão.

**Praticar em voz alta é inegociável.**

- [ ] Respostas escritas para as 6 perguntas comportamentais mais prováveis (formato STAR)

- [ ] Pitch de 3 minutos sobre a trajetória em fintech — para a pergunta “me conta sobre você”
  - [ ] Lista de 5 perguntas inteligentes para fazer ao entrevistador (projetos atuais, stack real, desafios)
  - [ ] Simulação de entrevista técnica com alguém de confiança ou gravação própria
  - [ ] Revisão do currículo com foco nas palavras-chave da descrição da vaga
- 

## Estratégia de Posicionamento

### Pontos Fortes que Jean Deve Enfatizar

**1. Fintech e ambientes transacionais críticos** Banco Inter e Banco Master são nomes que validam experiência exatamente no tipo de ambiente que a GOK atende. Jean deve mencionar isso nos primeiros 60 segundos da apresentação.

*“Nos últimos dois anos trabalhei em SDKs financeiros críticos — um de pagamentos para o Banco Inter e um de segurança mobile para o Banco Master. Ambos em ambientes onde falha não é opção.”*

**2. Segurança mobile avançada como diferencial raro** Anti-FRIDA, Anti-Root, AES-256 e Rust/NDK são competências que pouquíssimos desenvolvedores Android possuem. Em uma consultoria que atende ambientes transacionais, esse é um argumento forte de diferenciação.

**3. Liderança técnica real em fintech** Quatro anos e nove meses como Head Android no Monnos Crypto Bank é uma história poderosa — quando contada com profundidade e exemplos concretos de decisões arquiteturais.

**4. Visão full stack como facilitador de colaboração** O background em Node.js, TypeScript e PostgreSQL não é irrelevante. É um argumento direto para a comunicação horizontal que a GOK valoriza.

*“Consigo conversar com o backend sobre trade-offs de API porque já implementei APIs em produção.”*

**5. Ownership de ponta a ponta** Projetos pessoais publicados na Play Store com backend próprio demonstram que Jean assume o ciclo completo — da arquitetura ao deploy. É exatamente o que a vaga descreve como *atuação end-to-end*.

---

## Como Contornar Lacunas com Transparência

### Clean Architecture e MVI:

*"Trabalhei com MVVM com separação clara de responsabilidades — Use Cases isolados, Repositories como abstração da fonte de dados. Recentemente me aprofundei em Clean Architecture formal e MVI. Estou aplicando nos meus projetos pessoais e é o padrão que quero consolidar."*

### Testes automatizados:

*"Em Leega implementei testes unitários e de integração, mas reconheço que minha prática de TDD poderia ser mais sistemática. Estou usando MockK e JUnit com mais disciplina nos projetos recentes."*

### CI/CD mobile:

*"Minha experiência com CI/CD foi mais no lado backend. Tenho acompanhado práticas com Fastlane e GitHub Actions para mobile — é uma área que quero desenvolver. Se houver pipeline estabelecido, me adapto rápido."*

**Dica estratégica:** Transparência sobre lacunas, acompanhada de evidência de aprendizado proativo, demonstra autoconsciência — uma das características que a GOK mais valoriza em um perfil sênior.

---

## Conectando Experiências com a Vaga

| Experiência de Jean                  | Conexão com a GOK  |
|--------------------------------------|--|
| SDK Banco Inter (Pix, NFC, SmartPos) | Ambientes transacionais de alta criticidade — exatamente o que a GOK atende        |
| SDK Segurança Banco Master           | Segurança mobile em projetos financeiros — requisito implícito e diferencial       |
| Monnos Head Android (4a9m)           | Autonomia técnica, decisões arquiteturais, ownership — o que a GOK busca no sênior |
| Background full stack                | Comunicação horizontal com backend, produto e design sem atrito                    |
| Projetos pessoais publicados         | Ciclo completo: concepção, arquitetura, desenvolvimento, deploy e sustentação      |

## Palavras-chave para Usar na Entrevista

Clean Architecture · MVI · StateFlow · Ownership técnico · Decisões arquiteturais · Ambientes transacionais críticos · Resiliência no cliente · Fintech · SDK financeiro · Segurança mobile · Atuação end-to-end · Código em produção · Refatoração sem interrupção · Autonomia técnica

## Onde Investir Energia — e Onde Não Investir

### Ações de Maior Impacto

| Ação   | Impacto esperado  |
|--|---|
| Narrativa forte da Monnos com decisões arquiteturais reais | <b>Alto</b> — demonstra o ownership que a GOK mais valoriza |

| Ação   | Impacto esperado   |
|--|--|
| Explicar Clean Architecture com código de referência próprio           | <b>Alto</b> — requisito explícito na descrição da vaga             |
| Comparação argumentada MVVM vs MVI com casos de uso reais              | <b>Alto</b> — pergunta quase certa na entrevista técnica           |
| Conectar SDK do Banco Inter com resiliência e tratamento de erros      | <b>Médio-alto</b> — linguagem técnica alinhada ao contexto da vaga |
| Fazer perguntas inteligentes sobre os projetos atuais da GOK           | <b>Médio-alto</b> — sinaliza ownership e postura de sênior         |
| Mencionar segurança mobile avançada (Anti-FRIDA, AES-256) naturalmente | <b>Médio</b> — diferencial raro que poucos concorrentes têm        |

## O Que Não Deve Consumir Tempo Agora

- **Feature flags e Firebase Remote Config** — “*Conheço o conceito, nunca implementei em produção*” é suficiente
- **Detalhes de CI/CD** — ter um discurso honesto sobre a lacuna é mais eficaz do que fingir domínio
- **Rust e NDK em profundidade** — Jean já tem essa competência; basta saber defendê-la bem
- **Certificações** — não vão mudar o resultado da entrevista neste momento
- **Jetpack Compose além do que já domina** — apenas revisar padrões de estado
- **Room além do uso atual** — já bem demonstrado no currículo

## Erros a Evitar

### Na entrevista técnica

- ☐ Usar termos corretos (Clean Architecture, MVI) sem conseguir implementar ou detalhar
- ☐ Não ter um exemplo de código próprio para referenciar ao falar de arquitetura
- ☐ Falar “a gente fazia assim” sem especificar o que Jean fez diretamente

- ❑ Dizer que nunca lidou com problemas em produção — a trajetória em ambientes críticos diz o contrário
- ❑ Ser vago sobre a Monnos: *“trabalhei no app de crypto”* não é resposta de Head Android de quase 5 anos

## Na entrevista comportamental

- ❑ Respostas sem estrutura — descrever situações sem deixar claro qual foi a ação e o resultado
- ❑ Minimizar a liderança técnica — Jean liderou times em Monnos, Singu e Cedro; deve afirmar isso com clareza
- ❑ Não ter perguntas para o entrevistador — ausência de perguntas sinaliza falta de interesse ou ownership
- ❑ Postura defensiva ao ser questionado sobre lacunas — reconhecer com naturalidade e mostrar o que está sendo feito para fechar

## Na preparação

- ❑ Estudar tudo superficialmente em vez de dominar os tópicos críticos com profundidade
- ❑ Memorizar respostas prontas em vez de entender os conceitos para raciocinar ao vivo
- ❑ Ignorar a Monnos por *“já saber o que fez lá”* — o desafio é articular, não recordar
- ❑ Não praticar em voz alta — saber tecnicamente e conseguir explicar sob pressão são habilidades distintas

---

**Lembrete final:** Jean não precisa ser o candidato perfeito — precisa ser o candidato mais honesto, mais articulado e mais consciente do seu próprio valor. A combinação de fintech crítica, segurança mobile avançada, liderança técnica e stack moderna é genuinamente difícil de encontrar no mercado. Esta preparação existe para garantir que esse valor apareça com clareza onde mais importa: na entrevista.

## Career Agent **PRO**

A preparação perfeita para cada vaga de emprego. Inteligência aplicada ao seu contexto e à empresa que você quer entrar.

 [www.career-agent-pro.com](http://www.career-agent-pro.com)

 [support@main.career-agent-pro.com](mailto:support@main.career-agent-pro.com)

---

© 2026 Career Agent PRO. Todos os direitos reservados.

PREPARAÇÃO · INTELIGÊNCIA · RESULTADO