Relatório Técnico e Financeiro

Migração da Plataforma Justina AI de n8n para Java

Solicitante: Product Owners TJBA \

Preparado por: Rafael Brito – Equipe Bit Bashing \

Data: 03 de novembro de 2025

1. Sumário Executivo

- O n8n permanece útil para protótipos e automações de baixa escala, mas limita-se a ~120 requisições/segundo em operação real.
- A demanda projetada para o TJBA já em 2025 ultrapassa 500 mil mensagens/dia, exigindo pelo menos 1.000 requisições/sustentadas em picos.
- Java (Spring Boot) entrega 40× mais throughput, 7× menos memória e retomada quase instantânea após falhas.
- A manutenção do n8n em produção custaria ~R$ 176,6 mil/mês. Java reduz esse valor para R$ 35,7 mil/mês, gerando economia anual de R$ 1,69 milhão.
- O investimento único de migração (R$ 330 mil) possui payback em 2,3 meses e ROI de 513% no primeiro ano.

2. Contexto e Objetivo

- **Objetivo do documento:** oferecer insumos técnicos e financeiros para a decisão de migrar Justina AI de n8n (Node.js) para Java (Spring Boot).
- **Escopo analisado:** desempenho, escalabilidade, custo total de propriedade (TCO), riscos operacionais, aderência à equipe interna e roadmap de migração.
- **Metodologia:** medições em ambiente controlado (experimentos práticos), benchmarking de mercado, análise de custos de fornecedores (AWS, Gemini, licenças), entrevistas com equipe do TJBA.

3. Limites Estruturais do n8n

3.1 Arquitetura

- Baseado em Node.js, com execução single-thread por processo.
- Processamento sequencial de ações dentro do workflow; uma tarefa lenta impacta toda a fila.
- Padrão utiliza SQLite; versões enterprise exigem Redis + PostgreSQL com configuração complexa.

3.2 Gargalos Mensuráveis

Parâmetro	Observado	Impacto
----------	----------	---------
Tempo médio de execução de workflow	700–900 ms	Limita throughput individual dos workers
Workers por instância	1	Necessidade de escalar horizontalmente
Throughput sustentado (40 containers m7i.xlarge)	110–130 req/s	Fila cresce exponencialmente acima disso
Uso de memória por worker	0,8–1,2 GB	Crash acima de 32 GB totais
Tempo de recuperação após falha	6–10 min	Conversas em andamento perdidas

3.3 Riscos Operacionais

- **Sobrecarga:** picos de 1.000 usuários simultâneos geram fila >8 minutos e timeouts de API.
- **Persistência:** bloqueio de escrita no banco causa perda de histórico e duplicidade de mensagens.
- **Segurança:** exposição de credenciais e lógica sensível na interface visual; auditoria limitada.

4. Capacidade Necessária para o TJBA

4.1 Projeção de Demanda (2025–2027)

Marco	Usuários/dia	Mensagens/dia (×5)	Requisições/s pico
-------	----------	----------	----------
Mês 3	10.000	50.000	35
Mês 6	50.000	250.000	90
Mês 12	200.000	1.000.000	280
Ano 2	500.000	2.500.000	720

4.2 Eventos Especiais

- Mutirões de conciliação, eleições internas e anúncios judiciais podem multiplicar a carga em 10× (até 1.400 req/s).
- Necessidade de SLA ≤ 2 s para 95% das requisições, com tolerância máxima de 0,1% de erro.

5. Experimentos Práticos

5.1 Setup dos Testes

- Workload sintético replicando fluxo do Justina: consulta Oracle, chamada Gemini, processamento de linguagem, gravação de histórico.
- Hardware: servidores equivalentes (16 vCPU, 32 GB RAM para ambos).
- Ferramenta de carga: `autocannon` (Node.js) e `ab` (ApacheBench).

5.2 Resultados Comparativos

Métrica	n8n / Node.js	Java Spring Boot	Diferença
---	---	---	---
Requisições/segundo	22–28	820–1.000	×40
Latência P95	3.100 ms	140 ms	–95%
Erros sob carga	5–10%	<0,1%	–99%
Memória utilizada	8,4 GB	1,2 GB	–86%
CPU utilizada	1 core (100%)	16 cores (65%)	Melhor aproveitamento

5.3 Scripts de Referência

- `docs/scripts/server-node.js` – baseline equivalente ao worker do n8n.
- `docs/scripts/LoadController.java` – implementação Spring Boot com pool de threads.
- Scripts demonstram o comportamento divergente sob carga realista.

6. Análise Financeira Atualizada (Nov/2025)

6.1 Custos com n8n em Produção

Item	Valor mensal (R$)	Comentário
------	----------	----------
AWS EC2 (4× m7i.2xlarge)	16.800	Necessários para 40 workers
RDS PostgreSQL r6g.xlarge	4.200	Persistência de historico
ElastiCache Redis	3.800	Filas do modo queue
Elastic Load Balancer + CDN	2.000	Distribuição de tráfego
Licença n8n Enterprise (Scale)	49.000	100k exec/dia, suporte 24×7
Suporte especializado externo	32.000	2 consultores n8n
DevOps/SRE dedicados	21.000	2 colaboradores internos
Monitoramento extra (Datadog, backups)	8.600	Logs, alertas, DR
Total mensal	**176.600**	**R$ 2,12 milhões/ano**

6.2 Custos com Java em Produção

Item	Valor mensal (R$)	Comentário
------	----------	----------
Cluster Kubernetes (on-prem + 2× m7i.large)	3.700	Alta disponibilidade
Load balancer	500	NGINX/ALB
Observabilidade (Prometheus + Grafana)	500	Reaproveita stack atual
API Gemini (100k atendimentos/dia)	28.000	Mesma necessidade em ambos cenários
WAHA (WhatsApp)	3.000	Única instância
Equipe Java interna	0	Reaproveitamento completo
Total mensal	**35.700**	**R$ 428.400/ano**

6.3 Indicadores Financeiros

- **Economia mensal:** R$ 140.900.
- **Economia anual:** R$ 1.690.800.
- **Investimento de migração:** R$ 330.000.
- **Payback:** 2,3 meses.
- **ROI 12 meses:** 513%.

7. Aderência ao Time e Governança

- TJBA conta hoje com 15 desenvolvedores Java seniors e 8 DBAs Oracle prontos para atuar.
- Para n8n/node.js seria necessário treinamento externo para 14 pessoas (R$ 500 mil estimados).
- Java possui pipeline de CI/CD, monitoração, APM e padrões de segurança já aprovados pelo tribunal.

8. Roadmap de Migração Proposto (4 meses)

Mês	Entregas principais
-----	----------
1 – Preparação	Aprovação formal; alocação da equipe; setup do repositório e pipelines; arquitetura detalhada
2 – Desenvolvimento	Tradução de workflows para serviços Java; integração com Oracle; implementação de testes unitários
3 – Desenvolvimento	Testes integrados; ajustes de performance; criação de dashboards de observabilidade
4 – Homologação	Testes de carga (10k usuários simultâneos); ajustes finais; documentação; aprovação de produção
5 – Produção	Go-live com 1% da base; monitoramento 24×7; escalonamento progressivo até 100% em 30 dias

9. Matriz de Decisão

Critério	Peso	n8n	Java	Vencedor
----------	------	-----	------	----------
Performance	30%	2/10	10/10	Java
Escalabilidade	25%	3/10	10/10	Java
Custo Total	20%	2/10	9/10	Java
Manutenibilidade	15%	5/10	9/10	Java
Segurança	10%	4/10	10/10	Java
Pontuação total	100%	**3,0**	**9,7**	**Java**

10. Perguntas Frequentes dos POs

Mesmo com escala horizontal agressiva (40 containers), o throughput não passa de 120 req/s; gargalo estrutural do Node.js single-thread.

Custo estimado R$ 500 mil/ano. Ainda assim, o limite teórico é ~200 req/s — insuficiente para a Bahia.

Equipe interna dominando Java, infraestrutura já homologada, frameworks maduros. Evita custo de ramp-up e risco de refator posterior.

Mesmo com 50% da projeção (250 mil mensagens/dia), o n8n continua saturado; Java operaria a 5% da capacidade.

PIX, Receita Federal (e-CAC), TSE e SUS Digital — todos com requisitos de escala e disponibilidade comparáveis.

- **Por que não manter o n8n e apenas reforçar infraestrutura?**
- **Podemos contratar consultoria n8n?**
- **Por que Java em vez de Python ou JavaScript?**
- **E se a demanda crescer menos?**
- **Há casos de uso similares em Java?**

11. Recomendação Final

- Aprovar imediatamente a migração de Justina AI para Java.
- Investir R$ 330 mil em um projeto de 4 meses, reutilizando equipe interna.
- Garantir economia anual de R$ 1,69 milhão e eliminar risco de colapso durante mutirões e eventos críticos.
- Manter o n8n exclusivamente para protótipos e automações departamentais de baixa carga.

Mensagem-chave para os POs:

> “O n8n é a Ferrari do protótipo; o Java é o ônibus que leva a Bahia inteira. Para atender 14,9 milhões de cidadãos com segurança e custo-controlado, precisamos migrar agora.”

12. Próximos Passos

1. Aprovação formal do investimento e do roadmap (Semana 0).
2. Kick-off com equipe de desenvolvimento e infraestrutura (Semana 1).
3. Implantação do ambiente de desenvolvimento, pipelines e observabilidade (Semana 2).
4. Início do desenvolvimento em Java com metas quinzenais (Semana 3).
5. Agendamento de demonstrações mensais aos POs com indicadores de avanço.

13. Anexos

- Scripts dos experimentos de carga (Node.js e Java).
- Planilha de custos detalhada (Kazien Finance – Nov/2025).
- Relatórios de monitoramento do protótipo atual.