# Plano de Migração Multi-Tenant para AWS

## Sistema de Gestão Mercado Livre - Transformação SaaS

---

**Documento**: Plano de Migração Multi-Tenant
**Versão**: 1.0
**Data**: Novembro 2024
**Status**: Planejamento

---

## Sumário Executivo

### Objetivo

Transformar o sistema atual de **single-tenant** para uma plataforma **SaaS multi-tenant** escalável, capaz de atender **1000+ clientes pagantes** na AWS.

### Situação Atual

- Sistema preparado apenas para uso próprio
- Sem sistema de autenticação de usuários finais
- Sem isolamento de dados por tenant
- Estado armazenado em memória (não escalável)
- Banco de dados monolítico sem replicação

### Resultado Esperado

- Plataforma SaaS multi-tenant completa
- Suporte para 1000+ clientes simultâneos
- Infraestrutura AWS escalável e resiliente
- Isolamento total de dados entre tenants
- Sistema de billing e planos
- Observabilidade e monitoramento completos

### Prazo Total

**15 semanas** (3,5 meses) para implementação completa

### Investimento

- **Desenvolvimento**: ~$75,000 USD
- **Infraestrutura AWS**: ~$1,620 USD/mês
- **Total primeiro ano**: ~$94,440 USD

---

## Análise da Situação Atual

### Problemas Críticos

**1. Ausência de Isolamento Multi-Tenant**

- Não há modelo de User ou Tenant
- Sem sistema de autenticação de usuários finais
- Sem controle de acesso baseado em roles
- Impossível vender como SaaS

**2. Banco de Dados Monolítico**

- PostgreSQL local sem replicação
- Sem connection pooling
- Sem backups automáticos
- Não suporta 1000+ clientes

**3. Estado em Memória**

- OAuth state em Map local
- Sync status em memória
- Não funciona em ambiente distribuído

**4. Ausência de Cache Distribuído**

- Sem Redis
- Sem rate limiting distribuído
- Performance degradada

**5. Falta de Observabilidade**

- Sem APM
- Sem distributed tracing
- Sem alertas
- Impossível diagnosticar problemas

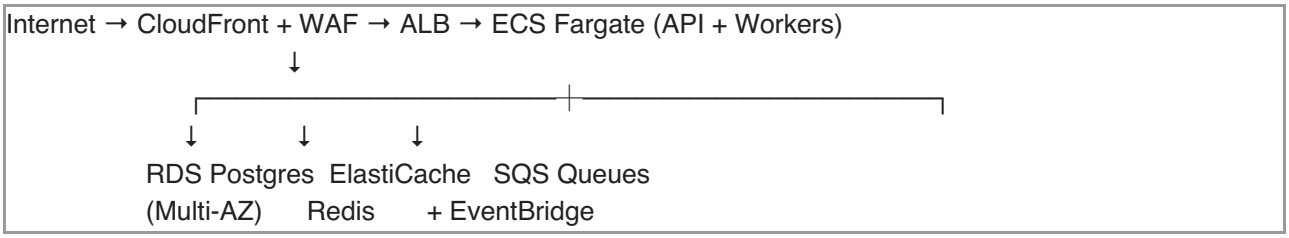
### Pontos Positivos

1. Tokens criptografados (AES-256-GCM)
2. Estrutura modular (NestJS)
3. ORM Prisma
4. Webhooks com dedupe
5. Docker Compose
6. Funcionalidades core funcionais

---

## Arquitetura Proposta AWS

### Diagrama de Componentes




## Componentes Principais

**Camada de Aplicação**

- CloudFront: CDN + DDoS protection
- ALB: Load balancing + SSL termination
- ECS Fargate: API (2-20 tasks) + Workers (1-10 tasks)

**Camada de Dados**

- RDS PostgreSQL Multi-AZ: db.r6g.xlarge + 2 replicas
- ElastiCache Redis: cache.r6g.large (2 nodes)
- S3: Assets + backups

**Camada de Mensageria**

- SQS: Webhooks + Jobs + DLQ
- EventBridge: Cron jobs

**Observabilidade**

- CloudWatch: Logs + Metrics + Alarms
- X-Ray: Distributed tracing
- CloudTrail: Audit logs

---

# Fases do Projeto

## FASE 1: Fundação Multi-Tenant (3 semanas)

**1.1 Modelagem de Dados**

**Novos Modelos**:

- Tenant (clientes/empresas)
- User (usuários de cada tenant)
- Session (autenticação)
- Subscription (billing)
- ApiKey (integrações)
- AuditLog (auditoria)

**Modificações**:

- Adicionar tenantId em todos os modelos existentes
- Criar índices compostos
- Migration de dados existentes

### 1.2 Autenticação e Autorização

**Implementar**:

- JWT strategy (access + refresh tokens)
- Guards: Auth, Roles, Tenant
- Decorators: @Auth(), @CurrentUser(), @CurrentTenant()
- Endpoints: login, register, refresh, logout

### 1.3 Middleware de Tenant Isolation

**Criar**:

- Interceptor global que injeta tenantId
- Validação de acesso cross-tenant
- Audit log de acessos

### 1.4 Refatoração de Services

**Atualizar todos os services**:

- AccountsService, ItemsService, OrdersService
- ShipmentsService, QuestionsService, BillingService
- Adicionar tenantId em todas as queries

---

## FASE 2: Frontend Multi-Tenant (2 semanas)

### 2.1 Sistema de Autenticação

**Páginas**:

- /login - Login de usuários
- /register - Registro de tenants
- /forgot-password - Recuperação
- /reset-password - Reset

**Features**:

- AuthContext com React
- Auto-refresh de tokens
- Logout automático em 401

### 2.2 Dashboard Multi-Tenant

**Páginas de Gestão**:

- /settings/profile - Perfil
- /settings/team - Usuários
- /settings/ml-accounts - Contas ML
- /settings/subscription - Plano

**Features**:

- Seletor de tenant

- Gestão de usuários e roles
- Limite de contas baseado no plano

#### 2.3 Billing e Planos

**Planos Sugeridos**:

- Free: $0 - 1 conta ML
- Basic: $29 - 3 contas ML
- Pro: $99 - 10 contas ML
- Enterprise: Custom - Ilimitado

**Integração Stripe**:

- Checkout embarcado
- Webhook de pagamentos
- Portal do cliente

---

## FASE 3: Infraestrutura AWS (3 semanas)

#### 3.1 Banco de Dados

**RDS PostgreSQL**:

- db.r6g.xlarge Multi-AZ
- 500GB GP3 storage
- 2 read replicas
- RDS Proxy (connection pooling)
- Automated backups (7 dias)

#### 3.2 Cache Distribuído

**ElastiCache Redis**:

- cache.r6g.large (2 nodes)
- Cluster mode enabled
- Encryption in-transit + at-rest

**Migrar para Redis**:

- OAuth state
- Sync status
- Rate limiting
- Session storage
- Cache de dados

#### 3.3 Mensageria

**SQS Queues**:

- painelml-webhooks (Standard)
- painelml-jobs (Standard)
- painelml-webhooks-dlq (DLQ)

**EventBridge**:

- Refresh tokens (1h)
- Auto-sync (30min)
- Cleanup sessions (diário)

#### 3.4 Containerização

**ECS Fargate**:

- API: 2 vCPU, 4GB (2-20 tasks)
- Workers: 1 vCPU, 2GB (1-10 tasks)
- Auto-scaling: CPU > 70%

**Docker**:

- Multi-stage builds
- ECR registry
- Health checks

#### 3.5 Load Balancing

**ALB**:

- SSL/TLS termination (ACM)
- Health checks: /health
- Sticky sessions

**CloudFront**:

- CDN global
- WAF + DDoS protection
- Cache de assets

---

## FASE 4: Observabilidade (2 semanas)

#### 4.1 Logging

**CloudWatch Logs**:

- Structured JSON logging
- Log groups por serviço
- Retention: 30 dias

**Campos obrigatórios**:

- tenantId, userId, requestId
- timestamp, level, message

#### 4.2 Métricas

**CloudWatch Metrics**:

- Métricas de negócio:

    - Contas ML por tenant
    - Sync jobs executados
    - Webhooks processados
    - Erros de API ML

**Dashboards**:

- Saúde do sistema
- Métricas por tenant
- Billing e uso

#### 4.3 Tracing

**AWS X-Ray**:

- Instrumentação automática
- Trace end-to-end
- Service map

#### 4.4 Alertas

**CloudWatch Alarms**:

- CPU > 80% por 5min
- Memory > 85% por 5min
- Error rate > 5% por 2min
- Latência p99 > 2s
- RDS connections > 80%

**SNS**:

- Email para equipe
- Integração Slack

---

## FASE 5: Segurança (2 semanas)

#### 5.1 Network Security

**VPC**:

- Subnets públicas (ALB, NAT)
- Subnets privadas (ECS, RDS, Redis)
- Security Groups restritivos

**WAF**:

- Rate limiting (100 req/min por IP)
- SQL injection protection
- XSS protection

#### 5.2 Secrets Management

**AWS Secrets Manager**:

- Migrar secrets do .env
- Rotation automática de passwords
- IAM roles com least privilege

**5.3 Encryption**

**At Rest**:

- RDS: encryption enabled
- S3: SSE-S3
- Redis: encryption enabled

**In Transit**:

- TLS 1.3 everywhere
- ACM certificates

**5.4 Audit**

**CloudTrail**:

- Log de ações AWS
- Retention: 90 dias

**AuditLog**:

- Login/logout
- Mudança de roles
- Conexão de contas ML
- Alteração de planos

---

## FASE 6: Testes e QA (2 semanas)

**6.1 Testes Automatizados**

**Unit Tests**:

- Coverage mínimo: 70%
- Testes de isolamento de tenant
- Testes de autenticação

**Integration Tests**:

- API end-to-end
- Webhooks
- Sync

**Load Tests**:

- k6 ou Artillery
- 1000 tenants simultâneos
- 10k req/min

**6.2 Testes de Segurança**

**Penetration Testing**:

- OWASP Top 10
- SQL injection
- XSS, CSRF

**Tenant Isolation**:

- Validar que tenant A não acessa dados de B
- Tokens manipulados

---

## FASE 7: Migration e Go-Live (1 semana)

### 7.1 Preparação

**Backup**:

- Snapshot do banco
- Backup de .env
- Documentação de rollback

**Migration Scripts**:

- Criar tenant default
- Migrar dados
- Validar integridade

### 7.2 Execução

**Blue-Green Deployment**:

- Manter ambiente antigo (blue)
- Deploy novo (green)
- Validação
- Switch de DNS

**Smoke Tests**:

- Login
- OAuth ML
- Sync
- Webhooks

### 7.3 Pós-Deploy

**Monitoramento 24h**:

- Métricas
- Logs de erro
- Billing

**Rollback Plan**:

- Procedimento documentado

- Tempo: < 15min

---

# Estimativas de Custo

## Desenvolvimento (15 semanas)

| Fase | Duração | Custo |
|---|---|---|
| Fase 1: Multi-Tenant | 3 semanas | $15,000 |
| Fase 2: Frontend | 2 semanas | $10,000 |
| Fase 3: AWS | 3 semanas | $15,000 |
| Fase 4: Observabilidade | 2 semanas | $10,000 |
| Fase 5: Segurança | 2 semanas | $10,000 |
| Fase 6: Testes | 2 semanas | $10,000 |
| Fase 7: Migration | 1 semana | $5,000 |
| **TOTAL** | **15 semanas** | **$75,000** |

## Infraestrutura AWS (mensal)

| Serviço | Configuração | Custo/mês |
|---|---|---|
| RDS PostgreSQL | db.r6g.xlarge Multi-AZ | $500 |
| ElastiCache Redis | cache.r6g.large (2 nodes) | $300 |
| ECS Fargate | 10 tasks (2 vCPU, 4GB) | $400 |
| ALB | 1 ALB + data transfer | $50 |
| S3 + CloudFront | 500GB + CDN | $100 |
| SQS + EventBridge | 100M requests | $50 |
| CloudWatch | Logs + Metrics | $100 |
| Secrets Manager | 10 secrets | $20 |
| X-Ray | Tracing | $50 |
| WAF | Rules + requests | $50 |
| **TOTAL** | | **$1,620** |

## Custo Total Primeiro Ano

- Desenvolvimento: $75,000
- AWS (12 meses): $19,440
- **TOTAL**: **$94,440**

## Break-even

Com plano Basic ($29/mês):

- 56 clientes pagantes = break-even mensal AWS
- 270 clientes pagantes = break-even total ano 1

---

# Riscos e Mitigações

| Risco | Probabilidade | Impacto | Mitigação |
|---|---|---|---|

| | | | |
|---|---|---|---|
| Data loss na migração | Baixa | Crítico | Backups + dry-run + validação |
| Downtime prolongado | Média | Alto | Blue-green deployment |
| Custos AWS acima do esperado | Alta | Médio | Monitoramento + alertas de budget |
| Bugs de isolamento de tenant | Média | Crítico | Testes extensivos + code review |
| Performance degradada | Média | Alto | Load tests + auto-scaling |
| Problemas de segurança | Baixa | Crítico | Penetration testing + audit |
| Atraso no cronograma | Alta | Médio | Buffer de 20% no prazo |

---

# Métricas de Sucesso

## Técnicas

- Latência p99 < 500ms
- Uptime > 99.9%
- Zero data leaks entre tenants
- Auto-scaling funcional
- Recovery time < 15min
- Test coverage > 70%

## Negócio

- Suportar 1000+ tenants
- Onboarding < 5min
- Churn rate < 5%
- NPS > 50
- Break-even em 12 meses

---

# Próximos Passos

## Decisões Necessárias

Antes de começar, você precisa decidir:

1. **Qual opção de implementação?**
    - A: Completa (15 semanas)
    - B: MVP Multi-Tenant (4 semanas)
    - C: Incremental (20 semanas)
2. **Modelo de billing?**
    - Stripe (recomendado)
    - Paddle
    - Manual
3. **Planos e preços?**
    - Free, Basic, Pro, Enterprise
    - Definir limites e features
4. **Região AWS?**

    - us-east-1 (N. Virginia)
    - sa-east-1 (São Paulo)

5. **Manter dados atuais?**
    - Migrar para tenant "default"
    - Começar do zero

6. **Prioridade de features?**
    - Must-have vs nice-to-have

### Recomendação

**Opção A (Completa)** é recomendada para:

- Sistema enterprise-ready
- Suporte real a 1000+ clientes
- Segurança e compliance
- Observabilidade completa

**Cronograma Sugerido**:

- Semana 1-3: Fase 1 (Multi-Tenant)
- Semana 4-5: Fase 2 (Frontend)
- Semana 6-8: Fase 3 (AWS)
- Semana 9-10: Fase 4 (Observabilidade)
- Semana 11-12: Fase 5 (Segurança)
- Semana 13-14: Fase 6 (Testes)
- Semana 15: Fase 7 (Go-Live)

---

## Contato e Suporte

Para dúvidas ou ajustes no plano:

- Revisar prioridades
- Ajustar cronograma
- Redefinir escopo
- Validar custos

---

**Documento preparado em**: Novembro 2024
**Próxima revisão**: Após aprovação das decisões
**Status**: Aguardando aprovação para início