

# Guía de Despliegue en EasyPanel - EscalaFin MVP

## Descripción General

Esta guía detalla el proceso completo para desplegar EscalaFin MVP en EasyPanel, incluyendo configuración de base de datos, variables de entorno, y optimizaciones específicas para producción.

## Prerrequisitos

### 1. Cuenta EasyPanel

- ✓ Cuenta activa en EasyPanel
- ✓ Servidor configurado (mínimo 2GB RAM)
- ✓ Dominio personalizado (opcional)

### 2. Servicios Externos

- ✓ Base de datos PostgreSQL
- ✓ Credenciales AWS S3 (si usa almacenamiento remoto)
- ✓ API Keys de Openpay
- ✓ Configuración EvolutionAPI (opcional)

## Preparación del Proyecto

### 1. Archivos Requeridos

Asegúrese de que su proyecto incluya:

```
escalafin_mvp/  
├── app/                                # Aplicación Next.js  
│   ├── package.json  
│   ├── next.config.js  
│   ├── prisma/  
│   └── ...  
├── Dockerfile                          # Para containerización  
├── docker-compose.yml                  # Para desarrollo local  
└── README.md
```

## 2. Dockerfile Optimizado

```
# Dockerfile para EasyPanel
FROM node:18-alpine AS base

# Dependencias
FROM base AS deps
RUN apk add --no-cache libc6-compat
WORKDIR /app
COPY app/package.json app/yarn.lock* ./
RUN yarn --frozen-lockfile

# Builder
FROM base AS builder
WORKDIR /app
COPY --from=deps /app/node_modules ./node_modules
COPY app/ .

# Variables de entorno para build
ENV NEXT_TELEMETRY_DISABLED 1
ENV NODE_ENV production

# Generar Prisma Client
RUN npx prisma generate

# Build aplicación
RUN yarn build

# Runner
FROM base AS runner
WORKDIR /app

ENV NODE_ENV production
ENV NEXT_TELEMETRY_DISABLED 1

RUN addgroup --system --gid 1001 nodejs
RUN adduser --system --uid 1001 nextjs

# Copiar archivos públicos
COPY --from=builder /app/public ./public

# Copiar build
COPY --from=builder --chown=nextjs:nodejs /app/.next/standalone ./
COPY --from=builder --chown=nextjs:nodejs /app/.next/static ./next/static

USER nextjs

EXPOSE 3000
ENV PORT 3000
ENV HOSTNAME "0.0.0.0"

CMD ["node", "server.js"]
```

### 3. next.config.js Para Producción

```
/** @type {import('next').NextConfig} */
const nextConfig = {
  // Configuración para EasyPanel
  output: 'standalone',

  // Optimizaciones
  compress: true,
  poweredByHeader: false,

  // Variables de entorno públicas
  env: {
    CUSTOM_KEY: process.env.CUSTOM_KEY,
  },

  // Headers de seguridad
  async headers() {
    return [
      {
        source: '/(.*)',
        headers: [
          {
            key: 'X-Frame-Options',
            value: 'DENY',
          },
          {
            key: 'X-Content-Type-Options',
            value: 'nosniff',
          },
          {
            key: 'Referrer-Policy',
            value: 'origin-when-cross-origin',
          },
        ],
      },
    ],
  },

  // Redirects para SEO
  async redirects() {
    return [
      {
        source: '/login',
        destination: '/auth/login',
        permanent: true,
      },
    ],
  },

  // Configuración de imágenes
  images: {
    domains: ['escalafin-uploads.s3.amazonaws.com'],
    formats: ['image/webp', 'image/avif'],
  },
}

module.exports = nextConfig
```

# Configuración en EasyPanel

## 1. Crear Nueva Aplicación

```
# 1. Conectar repositorio Git
# 2. Seleccionar rama main/master
# 3. Configurar build settings
```

## 2. Variables de Entorno

Configure las siguientes variables en EasyPanel:

```
# Base de Datos
DATABASE_URL=postgresql://usuario:password@host:5432/escalafin_prod

# NextAuth
NEXTAUTH_SECRET=tu_secreto_super_seguro_aqui
NEXTAUTH_URL=https://tu-dominio.com

# Entorno
NODE_ENV=production
NEXT_TELEMETRY_DISABLED=1

# Openpay (Producción)
OPENPAY_MERCHANT_ID=tu_merchant_id_prod
OPENPAY_PRIVATE_KEY=tu_private_key_prod
OPENPAY_BASE_URL=https://api.openpay.mx/v1

# Almacenamiento (S3 Recomendado para Producción)
STORAGE_TYPE=s3
AWS_BUCKET_NAME=escalafin-prod-uploads
AWS_REGION=us-east-1
AWS_FOLDER_PREFIX=production/
S3_MAX_FILE_SIZE=25
AWS_ACCESS_KEY_ID=tu_access_key
AWS_SECRET_ACCESS_KEY=tu_secret_key

# EvolutionAPI (Opcional)
EVOLUTION_API_BASE_URL=https://tu-evolution-api.com
EVOLUTION_API_KEY=tu_api_key
EVOLUTION_INSTANCE_NAME=escalafin_prod

# Configuración Adicional
MAX_UPLOAD_SIZE=26214400 # 25MB en bytes
DEFAULT_PAGINATION_LIMIT=20
SESSION_MAX_AGE=86400 # 24 horas
```

## 3. Build Commands

```
# Build command
cd app && npm install && npx prisma generate && npm run build

# Start command
cd app && npm start

# Puerto
3000
```

## Configuración de Base de Datos

---

### 1. PostgreSQL en EasyPanel

```
-- Crear base de datos
CREATE DATABASE escalafin_prod;

-- Crear usuario
CREATE USER escalafin_user WITH ENCRYPTED PASSWORD 'password_seguro';

-- Otorgar permisos
GRANT ALL PRIVILEGES ON DATABASE escalafin_prod TO escalafin_user;
```

### 2. Migración de Datos

```
# En tu máquina local
cd app

# Aplicar migraciones
DATABASE_URL="postgresql://user:pass@host:5432/escalafin_prod" \
npx prisma db push

# Poblar datos iniciales (opcional)
DATABASE_URL="postgresql://user:pass@host:5432/escalafin_prod" \
npx prisma db seed
```

### **3. Script de Seed Para Producción**

```

// app/scripts/seed-production.js
const { Prisma } = require('@prisma/client')
const bcrypt = require('bcryptjs')

const prisma = new PrismaClient()

async function main() {
  // Crear usuario administrador
  const hashedPassword = await bcrypt.hash('admin123', 12)

  await prisma.user.upsert({
    where: { email: 'admin@escalafin.com' },
    update: {},
    create: {
      email: 'admin@escalafin.com',
      firstName: 'Administrador',
      lastName: 'Sistema',
      password: hashedPassword,
      role: 'ADMIN',
      status: 'ACTIVE',
    },
  })

  console.log('✅ Usuario administrador creado')

  // Configuraciones del sistema
  const systemConfigs = [
    {
      key: 'system_initialized',
      value: 'true',
      description: 'Sistema inicializado',
      category: 'system',
    },
    {
      key: 'default_interest_rate',
      value: '0.024', // 2.4% mensual
      description: 'Tasa de interés por defecto',
      category: 'lending',
    },
    {
      key: 'max_loan_amount',
      value: '100000',
      description: 'Monto máximo de préstamo',
      category: 'lending',
    },
  ]

  for (const config of systemConfigs) {
    await prisma.systemConfig.upsert({
      where: { key: config.key },
      update: {},
      create: config,
    })
  }

  console.log('✅ Configuraciones del sistema creadas')
}

main()
.catch((e) => {
  console.error(e)
  process.exit(1)
})

```

```

})
  .finally(async () => {
    await prisma.$disconnect()
  })

```

## Despliegue Paso a Paso

### 1. Configuración Inicial

```

# 1. Fork el repositorio a tu cuenta GitHub
# 2. Clone localmente para hacer ajustes
git clone https://github.com/tu-usuario/escalafin-mvp.git
cd escalafin-mvp

# 3. Ajustar configuración para producción
# Editar app/next.config.js
# Verificar app/package.json

```

### 2. Configurar EasyPanel

#### 1. Crear Aplicación:

- Tipo: Node.js Application
- Repository: tu-usuario/escalafin-mvp
- Branch: main

#### 2. Build Settings:

Build Command: `cd app && yarn install && npx prisma generate && yarn build`

Start Command: `cd app && yarn start`

Port: 3000

#### 3. Configurar Variables de Entorno:

- Copiar todas las variables listadas anteriormente
- Usar valores de producción (no development)

### 3. Configurar Base de Datos

#### 1. Crear PostgreSQL Instance en EasyPanel

#### 2. Configurar Variables:

env

DATABASE\_URL=postgresql://user:pass@postgres:5432/escalafin

#### 3. Aplicar Migraciones:

bash

`npx prisma db push`

`npx prisma db seed`



## 4. Configurar Almacenamiento S3

```
# Crear bucket S3
aws s3 mb s3://escalafin-prod-uploads

# Configurar CORS
aws s3api put-bucket-cors --bucket escalafin-prod-uploads --cors-configuration file://cors.json

# cors.json
{
  "CORSRules": [
    {
      "AllowedOrigins": ["https://tu-dominio.com"],
      "AllowedHeaders": ["*"],
      "AllowedMethods": ["GET", "PUT", "POST", "DELETE"],
      "MaxAgeSeconds": 3000
    }
  ]
}
```

## 5. Configurar Dominio

### 1. DNS Records:

Type: CNAME

Name: escalafin (o @)

Value: tu-app.easypanel.host

### 2. SSL Certificate:

- EasyPanel configura automáticamente Let's Encrypt
- Verificar en Settings > Domain

## 6. Health Checks

```
// app/pages/api/health.js
export default function handler(req, res) {
  res.status(200).json({
    status: 'healthy',
    timestamp: new Date().toISOString(),
    version: process.env.npm_package_version,
    environment: process.env.NODE_ENV,
  })
}
```



## Monitoreo y Mantenimiento

### 1. Logs de Aplicación

```
# Ver logs en tiempo real
tail -f /var/log/app.log

# Logs de error específicos
grep "ERROR" /var/log/app.log | tail -20

# Monitoreo de performance
grep "slow query" /var/log/app.log
```

## 2. Métricas Importantes

- **Response Time:** < 500ms promedio
- **Error Rate:** < 1%
- **Uptime:** > 99.9%
- **Database Connections:** Monitorear pool
- **Memory Usage:** < 80%
- **CPU Usage:** < 70% promedio

## 3. Backups Automáticos

```
# Script de backup diario
#!/bin/bash
# backup.sh

DATE=$(date +%Y%m%d_%H%M%S)
BACKUP_DIR="/backups"
DB_NAME="escalafin_prod"

# Backup base de datos
pg_dump $DATABASE_URL > "$BACKUP_DIR/db_backup_$DATE.sql"

# Backup archivos (si usa local storage)
tar -czf "$BACKUP_DIR/files_backup_$DATE.tar.gz" /app/uploads/

# Subir a S3
aws s3 cp "$BACKUP_DIR/" s3://escalafin-backups/ --recursive

# Limpiar backups antiguos (mantener 30 días)
find $BACKUP_DIR -type f -mtime +30 -delete

echo "✅ Backup completado: $DATE"
```

## 4. Configurar Monitoreo

```
# docker-compose.monitoring.yml (opcional)
version: '3.8'
services:
  prometheus:
    image: prom/prometheus
    ports:
      - "9090:9090"
    volumes:
      - ./prometheus.yml:/etc/prometheus/prometheus.yml

  grafana:
    image: grafana/grafana
    ports:
      - "3001:3000"
    environment:
      - GF_SECURITY_ADMIN_PASSWORD=admin123
```

# Seguridad en Producción

## 1. Headers de Seguridad

```
// app/middleware.js
import { NextResponse } from 'next/server'

export function middleware(request) {
  const response = NextResponse.next()

  // Headers de seguridad
  response.headers.set('X-DNS-Prefetch-Control', 'on')
  response.headers.set('Strict-Transport-Security', 'max-age=63072000')
  response.headers.set('X-XSS-Protection', '1; mode=block')
  response.headers.set('X-Content-Type-Options', 'nosniff')
  response.headers.set('Referrer-Policy', 'origin-when-cross-origin')

  return response
}
```

## 2. Rate Limiting

```
// app/lib/rate-limit.js
import { Redis } from 'ioredis'

const redis = new Redis(process.env.REDIS_URL)

export async function rateLimit(identifier, limit = 100, window = 3600) {
  const key = `rate_limit:${identifier}`
  const current = await redis.incr(key)

  if (current === 1) {
    await redis.expire(key, window)
  }

  return {
    count: current,
    remaining: Math.max(0, limit - current),
    reset: new Date(Date.now() + window * 1000),
    exceeded: current > limit,
  }
}
```

## 3. Input Validation

```
// app/lib/validation.js
import { z } from 'zod'

export const fileUploadSchema = z.object({
  file: z.custom((file) => {
    if (!(file instanceof File)) return false
    if (file.size > 25 * 1024 * 1024) return false // 25MB max
    return true
  }),
  category: z.enum(['identification', 'income_proof', 'contracts']),
  description: z.string().optional(),
})
```

## Solución de Problemas

---

### 1. Errores Comunes

#### Build Failures

```
# Limpiar cache
rm -rf app/.next app/node_modules
cd app && yarn install

# Verificar variables de entorno
echo $DATABASE_URL
```

#### Database Connection

```
# Test conexión
npx prisma db push --preview-feature

# Verificar SSL
DATABASE_URL="postgresql://user:pass@host:5432/db?sslmode=require"
```

#### File Upload Issues

```
# Verificar permisos S3
aws s3api get-bucket-acl --bucket escalafin-uploads

# Test upload
curl -X POST -F "file=@test.jpg" https://tu-dominio.com/api/files/upload
```

## 2. Performance Optimization

```
// app/next.config.js
const nextConfig = {
  // Compresión
  compress: true,

  // Optimización de imágenes
  images: {
    formats: ['image/webp', 'image/avif'],
    minimumCacheTTL: 60 * 60 * 24 * 7, // 1 semana
  },

  // Headers de cache
  async headers() {
    return [
      {
        source: '/api/(.*)',
        headers: [
          {
            key: 'Cache-Control',
            value: 'public, max-age=300, s-maxage=300',
          },
        ],
      },
    ],
  },
}
```




## Soporte Post-Despliegue

### 1. Checklist Post-Despliegue




- ☐ ☒ Aplicación accesible via HTTPS
- ☐ ☒ Base de datos conectada correctamente
- ☐ ☒ Login de administrador funciona
- ☐ ☒ Upload de archivos funciona
- ☐ ☒ Emails de notificación funcionan (si configurado)
- ☐ ☒ Integración de pagos funciona
- ☐ ☒ WhatsApp notifications funcionan (si configurado)
- ☐ ☒ Backups configurados
- ☐ ☒ Monitoreo activo
- ☐ ☒ Logs funcionando

### 2. Contacto Técnico

Para problemas específicos de EasyPanel:

-  **Soporte:** [support@easypanel.io](mailto:support@easypanel.io)
-  **Documentación:** <https://docs.easypanel.io>
-  **Discord:** EasyPanel Community

Para problemas específicos de EscalaFin:

-  **Documentación:** Revisar README.md
-  **Logs:** Verificar en EasyPanel dashboard
-  **Testing:** Usar endpoints de health check

---

**Versión:** 2.1.0

**Compatibilidad:** EasyPanel 1.x, Node.js 18+

**Última actualización:** Septiembre 2025