🚀 Guía de Despliegue - EscalaFin

📋 Opciones de Despliegue

Esta guía cubre el despliegue de EscalaFin en diferentes plataformas:

1. Easypanel (Recomendado)
2. Vercel
3. Docker
4. Manual/VPS

🎯 Easypanel (Recomendado)

Pre-requisitos

- Cuenta en Easypanel
- Repositorio GitHub del proyecto
- Base de datos PostgreSQL configurada

Paso 1: Preparar el Repositorio

1.1 Crear Dockerfile

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

# Install dependencies only when needed
FROM base AS deps
RUN apk add --no-cache libc6-compat
WORKDIR /app

# Install dependencies based on the preferred package manager
COPY package.json yarn.lock* package-lock.json* pnpm-lock.yaml* ./
RUN \
  if [ -f yarn.lock ]; then yarn --frozen-lockfile; \
  elif [ -f package-lock.json ]; then npm ci; \
  elif [ -f pnpm-lock.yaml ]; then yarn global add pnpm && pnpm i --frozen-lockfile; \
  else echo "Lockfile not found." && exit 1; \
  fi

# Rebuild the source code only when needed
FROM base AS builder
WORKDIR /app
COPY --from=deps /app/node_modules ./node_modules
COPY . .

# Generate Prisma client
RUN npx prisma generate

# Build application
RUN yarn build

# Production image, copy all the files and run next
FROM base AS runner
WORKDIR /app

ENV NODE_ENV production

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

COPY --from=builder /app/public ./public
COPY --from=builder /app/.next/standalone ./
COPY --from=builder /app/.next/static ./.next/static
COPY --from=builder /app/node_modules/.prisma ./node_modules/.prisma

USER nextjs

EXPOSE 3000

ENV PORT 3000
ENV HOSTNAME "0.0.0.0"

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

1.2 Configurar next.config.js

```
/** @type {import('next').NextConfig} */
const nextConfig = {
  output: 'standalone',
  experimental: {
    outputFileTracingRoot: "/opt/app",
  },
  images: {
    remotePatterns: [
      {
        protocol: 'https',
        hostname: '**',
      },
    ],
  },
  webpack: (config) => {
    config.externals.push('@node-rs/argon2', '@node-rs/bcrypt');
    return config;
  },
};

module.exports = nextConfig;
```

1.3 Crear .dockerignore

```
node_modules
.next
.git
.env.local
.env.*.local
README.md
Dockerfile
docker-compose.yml
.dockerignore
```

Paso 2: Configurar en Easypanel

2.1 Crear Nueva Aplicación

1. Login en Easypanel
2. Crear nuevo proyecto
3. Seleccionar “GitHub Repository”
4. Conectar repositorio EscalaFin

2.2 Configurar Variables de Entorno

```
NODE_ENV=production
NEXTAUTH_SECRET=tu_secret_super_seguro_de_32_caracteres
NEXTAUTH_URL=https://tu-dominio.tu-panel.app

# Base de Datos (PostgreSQL de Easypanel)
DATABASE_URL=postgresql://user:password@db:5432/escalafin

# Openpay (Producción)
OPENPAY_MERCHANT_ID=tu_merchant_id_produccion
OPENPAY_PRIVATE_KEY=tu_private_key_produccion
OPENPAY_PUBLIC_KEY=tu_public_key_produccion
OPENPAY_BASE_URL=https://api.openpay.mx/v1

# AWS S3
AWS_ACCESS_KEY_ID=tu_access_key
AWS_SECRET_ACCESS_KEY=tu_secret_key
AWS_BUCKET_NAME=escalafin-prod
AWS_REGION=us-east-1
AWS_FOLDER_PREFIX=production/

# WhatsApp EvolutionAPI
EVOLUTION_API_URL=https://tu-evolution-api.com
EVOLUTION_API_TOKEN=tu_token_produccion
EVOLUTION_INSTANCE_NAME=escalafin-prod
```

2.3 Configurar Base de Datos

1. Crear servicio PostgreSQL en Easypanel
2. Configurar nombre: `escalafin-db`
3. Usuario: `escalafin_user`
4. Password: generar password seguro
5. Base de datos: `escalafin`

2.4 Configurar Build

```
# easypanel.yml (si es necesario)
build:
  dockerfile: Dockerfile
  context: .
```

Paso 3: Desplegar

3.1 Primera Implementación

1. Push código a GitHub
2. Easypanel detectará cambios automáticamente
3. Iniciará build automático
4. Despliegue automático

3.2 Configurar Dominio

1. En Easypanel → Aplicación → Domains
2. Agregar dominio personalizado
3. SSL automático se configurará

3.3 Inicializar Base de Datos

Ejecutar una sola vez después del primer despliegue:

```
# Conectar a la aplicación via SSH o ejecutar comando
npx prisma db push
npx prisma db seed
```

⚡ Vercel (Alternativa)

Configuración para Vercel

vercel.json

```
{
  "env": {
    "NEXTAUTH_SECRET": "@nextauth-secret",
    "DATABASE_URL": "@database-url",
    "OPENPAY_MERCHANT_ID": "@openpay-merchant-id"
  },
  "build": {
    "env": {
      "DATABASE_URL": "@database-url"
    }
  },
  "functions": {
    "app/api/**/*.ts": {
      "maxDuration": 30
    }
  }
}
```

Pasos para Vercel

1. Conectar repositorio GitHub
2. Configurar variables de entorno
3. Conectar base de datos externa (Supabase, Railway, etc.)
4. Deploy automático

🐳 Docker (Local/VPS)

docker-compose.yml

```yaml
version: '3.8'

services:
  app:
    build: .
    ports:
      - "3000:3000"
    environment:
      - NODE_ENV=production
      - DATABASE_URL=postgresql://escalafin_user:escalafin_pass@db:5432/escalafin
    env_file:
      - .env.production
    depends_on:
      - db
    volumes:
      - ./uploads:/app/uploads

  db:
    image: postgres:14
    environment:
      POSTGRES_DB: escalafin
      POSTGRES_USER: escalafin_user
      POSTGRES_PASSWORD: escalafin_pass
    ports:
      - "5432:5432"
    volumes:
      - postgres_data:/var/lib/postgresql/data
      - ./backups:/backups

  nginx:
    image: nginx:alpine
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf
      - ./ssl:/etc/nginx/ssl
    depends_on:
      - app

volumes:
  postgres_data:
```

Comandos Docker

```
# Build y ejecutar
docker-compose up -d

# Ver logs
docker-compose logs -f app

# Ejecutar migraciones
docker-compose exec app npx prisma db push

# Backup de base de datos
docker-compose exec db pg_dump -U escalafin_user escalafin > backup.sql
```

🖥️ Manual/VPS

Para Ubuntu/Debian

1. Preparar Servidor

```
# Actualizar sistema
sudo apt update && sudo apt upgrade -y

# Instalar Node.js
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs

# Instalar PM2
sudo npm install pm2@latest -g

# Instalar PostgreSQL
sudo apt install postgresql postgresql-contrib
```

2. Configurar Base de Datos

```
sudo -u postgres psql
CREATE DATABASE escalafin;
CREATE USER escalafin_user WITH PASSWORD 'tu_password_seguro';
GRANT ALL PRIVILEGES ON DATABASE escalafin TO escalafin_user;
\q
```

3. Configurar Aplicación

```
# Clonar repositorio
git clone <tu-repo-url> /var/www/escalafin
cd /var/www/escalafin

# Instalar dependencias
npm install

# Configurar variables de entorno
cp .env.example .env.production
# Editar .env.production con valores correctos

# Build aplicación
npm run build

# Configurar PM2
pm2 start ecosystem.config.js
pm2 save
pm2 startup
```

4. Configurar Nginx

```
# /etc/nginx/sites-available/escalafin
server {
    listen 80;
    server_name tu-dominio.com;

    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_cache_bypass $http_upgrade;
    }
}
```

```
# Activar sitio
sudo ln -s /etc/nginx/sites-available/escalafin /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl restart nginx

# SSL con Certbot
sudo apt install certbot python3-certbot-nginx
sudo certbot --nginx -d tu-dominio.com
```

✅ Checklist Post-Despliegue

Verificaciones Técnicas

- [ ] Aplicación accesible en dominio

- [ ] SSL funcionando correctamente
- [ ] Base de datos conectada
- [ ] Migraciones ejecutadas
- [ ] Datos de prueba cargados
- [ ] APIs respondiendo correctamente
- [ ] Webhooks configurados
- [ ] Archivos S3 funcionando
- [ ] Notificaciones WhatsApp operativas

Verificaciones Funcionales

- [ ] Login con usuarios de prueba
- [ ] Crear nuevo cliente
- [ ] Procesar préstamo de prueba
- [ ] Realizar pago de prueba
- [ ] Generar reporte
- [ ] Enviar notificación WhatsApp
- [ ] Upload/download de archivo

Configuraciones de Producción

- [ ] Cambiar usuarios y passwords por defecto
- [ ] Configurar respaldos automáticos
- [ ] Configurar monitoreo
- [ ] Configurar alertas de error
- [ ] Documentar credenciales de producción

🔧 Solución de Problemas

Build Failures

```
# Limpiar caché
rm -rf .next node_modules
npm install
npm run build
```

Database Issues

```
# Reset database (¡CUIDADO!)
npx prisma db push --force-reset
npx prisma db seed
```

Memory Issues

```
# Aumentar memoria Node.js
NODE_OPTIONS="--max-old-space-size=4096" npm run build
```

SSL Issues

```
# Verificar certificados
sudo certbot certificates
sudo certbot renew --dry-run
```

📞 Soporte de Despliegue

Para problemas específicos de despliegue:
1. Revisar logs de la aplicación
2. Verificar variables de entorno
3. Comprobar conectividad de base de datos
4. Validar configuración de DNS

¡Tu aplicación EscalaFin está lista para producción! 🚀