

Ebenezer Tax Services CRM

Manual de Despliegue e IT

Versión: 1.2 Fecha: Febrero 2026 Audiencia: Personal de IT / DevOps

Tabla de Contenidos

- [1. Requisitos del Sistema](#)
- [2. Arquitectura](#)
- [3. Instalación con Docker](#)
- [4. Instalación Manual](#)
- [5. Configuración](#)
- [6. Base de Datos](#)
- [7. Seguridad](#)
- [8. Monitoreo](#)
- [9. Respaldos](#)
- [10. Solución de Problemas](#)

Requisitos del Sistema

Hardware Mínimo (Producción)

Recurso	Especificación
CPU	4 cores
RAM	8 GB
Disco	100 GB SSD
OS	Ubuntu 22.04 LTS / Windows Server 2022

Hardware Recomendado

Recurso	Especificación
CPU	8+ cores
RAM	16+ GB
Disco	250+ GB SSD
OS	Ubuntu 22.04 LTS

Software Requerido

Python 3.11+
Node.js 18+
PostgreSQL 16+
Redis 7+
Nginx 1.24+
Docker 24+ (opcional pero recomendado)

Arquitectura

Diagrama de Componentes

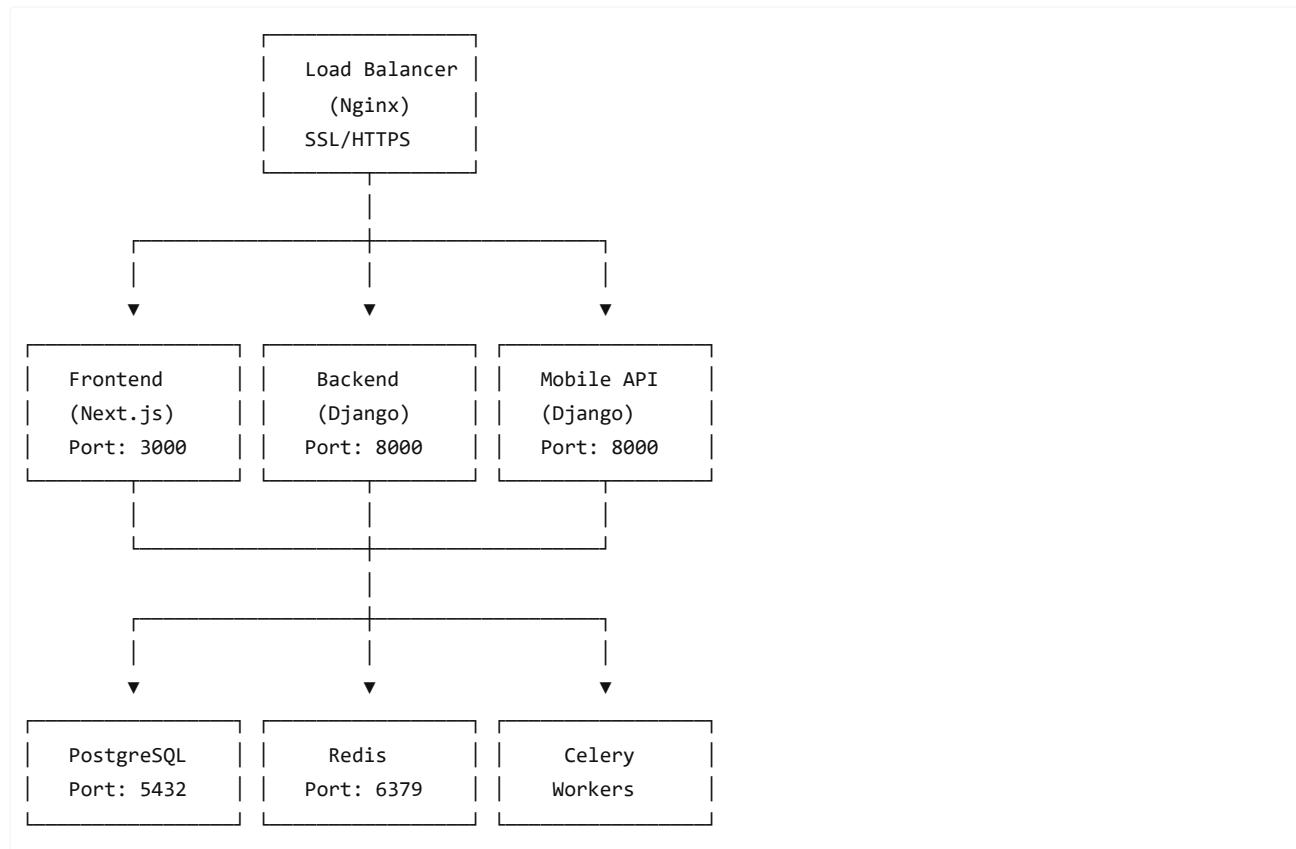


Figura 3: Dashboard Principal

Este dashboard principal muestra las siguientes métricas:

- Casos Activos: 24 (↑ 12% vs mes anterior)
- Citas de Hoy: 5 (Próxima: 2:30 PM)
- Tareas Pendientes: 8 (2 vencidas)
- Mensajes: 12 (3 sin leer)

En la sección "Notas Rápidas" se presentan las siguientes tareas:

- Llamar a María García para confirmar documentos del caso 2024 (Hace 2 horas)
- Revisar declaración de ABC Corp antes del viernes (Ayer)
- Enviar cotización a nuevo cliente potencial (15 Feb)

+ Nueva Nota

Figura 3: Dashboard principal con métricas y notas adhesivas

Figura 1: Arquitectura del sistema

Componentes

Componente	Tecnología	Puerto	Descripción
Backend API	Django 5.1 + DRF	8000	API REST principal
Frontend	Next.js 15	3000	Interfaz web
Base de Datos	PostgreSQL 16	5432	Almacenamiento persistente
Cache	Redis 7	6379	Cache y broker
Task Queue	Celery	-	Tareas asíncronas
Proxy	Nginx	80/443	Reverse proxy + SSL

Instalación con Docker (Recomendado)

Paso 1: Clonar Repositorio

```
git clone https://github.com/empresa/ebenezer-crm.git  
cd ebenezer-crm
```

Paso 2: Configurar Variables de Entorno

```
cp .env.example .env  
# Editar .env con valores de producción
```

Figura 5: Formulario de Contacto con Múltiples Corporaciones

Editar Contacto

Nombre	Apellido
Carlos	Rodríguez
Email	Teléfono
carlos.r@email.com	(555) 345-6789

Corporaciones

Corporación Primaria *

ABC Corp

La corporación principal asociada a este contacto

Corporaciones Adicionales

XYZ Inc X Tech Solutions LLC X

Seleccionar corporación... + Agregar

Otras corporaciones donde este contacto tiene participación

Cancelar Guardar Cambios

Figura 5: Formulario de contacto con selector de múltiples corporaciones

Figura 2: Configuración de variables de entorno

Paso 3: Iniciar Servicios

```
# Desarrollo  
docker-compose up -d  
  
# Producción (con Nginx)  
docker-compose --profile production up -d
```

Paso 4: Ejecutar Migraciones

```
docker-compose exec backend python manage.py migrate
```

Paso 5: Crear Superusuario

```
docker-compose exec backend python manage.py createsuperuser
```

Verificar Instalación

```
# Verificar que todos los contenedores estén corriendo  
docker-compose ps
```

```
# Ver logs
docker-compose logs -f backend
```

Figura 4: Lista de Contactos						
Contactos						+ Nuevo Contacto
Buscar contactos...			Estado: Todos	Corporación: Todas		
NOMBRE	EMAIL	TELÉFONO	CORPORACIÓN	ESTADO	ACCIONES	
MG María García Cliente desde 2022	maria.garcia@email.com	(555) 123-4567	ABC Corp	Activo	...	
JP Juan Pérez Cliente desde 2023	juan.perez@email.com	(555) 234-5678	XYZ Inc	Activo	...	
CR Carlos Rodríguez Cliente desde 2021	carlos.r@email.com	(555) 345-6789	ABC Corp +2	Activo	...	

Figura 4: Lista de contactos con filtros y acciones

Figura 3: Servicios Docker corriendo

Instalación Manual

Backend (Django)

```
cd "CRM Back end"

# Crear entorno virtual
python -m venv venv
source venv/bin/activate # Linux
.\venv\Scripts\activate # Windows

# Instalar dependencias
pip install -r requirements/production.txt

# Configurar variables
export DJANGO_SETTINGS_MODULE=config.settings.production

# Migraciones
python manage.py migrate

# Archivos estáticos
python manage.py collectstatic --noinput

# Iniciar con Gunicorn
gunicorn config.wsgi:application --bind 0.0.0.0:8000 --workers 4
```

Frontend (Next.js)

```
cd "CRM Front end"

# Instalar dependencias
```

```
npm install

# Build de producción
npm run build

# Iniciar servidor
npm start
```

Celery Workers

```
# Worker principal
celery -A config worker --loglevel=info

# Scheduler (Celery Beat)
celery -A config beat --loglevel=info \
--scheduler django_celery_beat.schedulers:DatabaseScheduler
```

Configuración

Variables de Entorno Críticas

```
# ===== DJANGO =====
DEBUG=False
SECRET_KEY=<clave-segura-64-caracteres>
ALLOWED_HOSTS=dominio.com,api.dominio.com

# ===== DATABASE =====
DATABASE_URL=postgres://usuario:password@host:5432/ebenezer_crm

# ===== REDIS =====
REDIS_URL=redis://localhost:6379/0
CELERY_BROKER_URL=redis://localhost:6379/1

# ===== JWT / SEGURIDAD =====
JWT_SIGNING_KEY=<clave-32-bytes>
PORTAL_JWT_SIGNING_KEY=<clave-diferente-32-bytes>
FIELD_ENCRYPTION_KEY=<fernet-key-base64>

# ===== CORS =====
CORS_ALLOWED_ORIGINS=https://dominio.com

# ===== EMAIL =====
EMAIL_HOST=smtp.gmail.com
EMAIL_PORT=587
EMAIL_HOST_USER=correo@dominio.com
EMAIL_HOST_PASSWORD=<app-password>
```

Generar Claves Seguras

```
# SECRET_KEY (Django)
python -c "from django.core.management.utils import get_random_secret_key;
print(get_random_secret_key())"
```

```
# JWT_SIGNING_KEY
python -c "import secrets; print(secrets.token_urlsafe(32))"

# FIELD_ENCRYPTION_KEY (Fernet)
python -c "from cryptography.fernet import Fernet; print(Fernet.generate_key().decode())"
```

Base de Datos

Estructura Principal

```
ebenezer_crm
├── auth_* (Django auth)
├── django_* (Django admin, sessions)
├── crm_users (Usuarios del sistema)
├── crm_departments (Departamentos)
├── crm_contacts (Contactos)
├── crm_contact_corporations (M2M: Contactos-Corporaciones)
├── crm_corporations (Corporaciones)
├── crm_corporation_related (M2M: Corporaciones relacionadas)
├── crm_tax_cases (Casos de impuestos)
├── crm_documents (Documentos)
├── crm_department_client_folders (Carpetas por departamento)
├── crm_document_access_logs (Logs de acceso)
├── crm_appointments (Citas)
├── crm_audit_logs (Auditoría)
└── celery_* (Tareas programadas)
```

Modelo Multi-Corporación

```
-- Contactos con múltiples corporaciones
crm_contacts
├── primary_corporation_id (FK) -- Corporación principal
└── corporations (M2M) -- Todas las asociaciones

-- Tabla intermedia M2M
crm_contact_corporations
├── contact_id (FK)
└── corporation_id (FK)
```

Modelo de Departamentos

```
-- Departamentos
CREATE TABLE crm_departments (
    id UUID PRIMARY KEY,
    name VARCHAR(100) UNIQUE,
    code VARCHAR(20) UNIQUE,
    color VARCHAR(7),
    icon VARCHAR(50),
    is_active BOOLEAN DEFAULT TRUE,
    "order" INTEGER,
    created_at TIMESTAMP,
    updated_at TIMESTAMP
```

```

);

-- Carpetas por departamento/cliente
CREATE TABLE crm_department_client_folders (
    id UUID PRIMARY KEY,
    name VARCHAR(255),
    department_id UUID REFERENCES crm_departments(id),
    contact_id UUID REFERENCES crm_contacts(id),
    corporation_id UUID REFERENCES crm_corporations(id),
    parent_id UUID REFERENCES crm_department_client_folders(id),
    is_default BOOLEAN DEFAULT FALSE,
    created_by_id UUID REFERENCES crm_users(id),
    created_at TIMESTAMP,
    updated_at TIMESTAMP,

    CONSTRAINT folder_linked_to_one_client_type
        CHECK (contact_id IS NOT NULL OR corporation_id IS NOT NULL)
);

```

Comandos de Base de Datos

```

# Conexión
psql -h localhost -U ebenezer -d ebenezer_crm

# Respaldo
pg_dump -h localhost -U ebenezer ebenezer_crm > backup.sql

# Restaurar
psql -h localhost -U ebenezer ebenezer_crm < backup.sql

# Migraciones Django
python manage.py showmigrations
python manage.py migrate

```

Seguridad

Medidas Implementadas

Característica	Implementación
Autenticación	JWT access/refresh tokens
2FA	TOTP (Google Authenticator)
Encriptación PII	Fernet (AES-128)
Password Hashing	Argon2
Rate Limiting	200/min autenticado, 20/min anónimo
CORS	Whitelist de dominios
Session Timeout	30 minutos
Audit Logging	Todas las acciones

Permisos Departamento

Basado en departamento del usuario

Configuración SSL/HTTPS

```
# nginx.conf
server {
    listen 443 ssl http2;
    server_name api.dominio.com;

    ssl_certificate /etc/nginx/ssl/cert.pem;
    ssl_certificate_key /etc/nginx/ssl/key.pem;
    ssl_protocols TLSv1.2 TLSv1.3;

    location / {
        proxy_pass http://backend:8000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
    }
}
```

Rate Limits

Endpoint	Límite
Anónimo	20/minuto
Autenticado	200/minuto
Login	5/minuto
Password Reset	3/hora

Monitoreo

Logs de Aplicación

```
# Django
tail -f /var/log/ebenezer/django.log

# Celery
tail -f /var/log/ebenezer/celery.log

# Nginx
tail -f /var/log/nginx/access.log
```

Verificar Estado

```
# Estado de servicios
sudo systemctl status nginx postgresql redis celery

# Uso de recursos
htop
df -h
```

```
free -m  
  
# Conexiones de red  
netstat -tlnp
```

Métricas a Monitorear

- Tiempo de respuesta API (< 200ms)
- Uso de CPU (< 70%)
- Uso de RAM (< 80%)
- Conexiones de BD activas
- Cola de tareas Celery
- Errores 5xx (< 0.1%)

Respaldos

Estrategia

Tipo	Frecuencia	Retención
BD completa	Diario	30 días
BD incremental	Cada hora	7 días
Documentos	Semanal	90 días
Configuración	Por cambio	Indefinido

Script de Respaldo

```
#!/bin/bash  
# /opt/scripts/backup.sh  
  
DATE=$(date +%Y%m%d_%H%M)  
BACKUP_DIR=/backups/ebenezer  
  
# Database  
pg_dump -h localhost -U ebenezer_crm | gzip > $BACKUP_DIR/db_$DATE.sql.gz  
  
# Media files  
tar -czf $BACKUP_DIR/media_$DATE.tar.gz /var/www/ebenezer/media/  
  
# Limpiar backups antiguos (30 días)  
find $BACKUP_DIR -name "*.gz" -mtime +30 -delete  
  
# Subir a S3 (opcional)  
aws s3 sync $BACKUP_DIR s3://ebenezer-backups/
```

Cron Job

```
# crontab -e  
0 2 * * * /opt/scripts/backup.sh >> /var/log/backup.log 2>&1
```

Solución de Problemas

Error 500 en API

```
# Ver logs  
tail -f /var/log/ebenezer/django.log  
  
# Verificar configuración  
python manage.py check --deploy  
  
# Verificar conexión DB  
python manage.py dbshell
```

Celery no procesa tareas

```
# Estado de workers  
celery -A config inspect active  
  
# Verificar Redis  
redis-cli ping  
  
# Reiniciar  
sudo systemctl restart celery
```

Problemas de conexión

1. Verificar firewall (puertos 80, 443, 5432, 6379)
2. Verificar CORS en configuración
3. Verificar ALLOWED_HOSTS
4. Verificar SSL/certificados

Comandos de Diagnóstico

```
# Verificar Django  
python manage.py check  
  
# Verificar migraciones pendientes  
python manage.py showmigrations  
  
# Shell interactivo  
python manage.py shell_plus  
  
# Limpiar sesiones  
python manage.py clearsessions
```

Checklist de Producción

- Variables de entorno configuradas
- SECRET_KEY único y seguro
- DEBUG=False
- ALLOWED_HOSTS configurado
- HTTPS habilitado

- Certificados SSL válidos
 - Backups automáticos configurados
 - Monitoreo activo
 - Logs configurados
 - Rate limiting activo
 - CORS restringido
 - Firewall configurado
-

Contacto de Soporte

- **Email:** it@ebenezer-crm.com
 - **Documentación:** </docs/>
 - **API Docs:** </api/docs/>
-

© 2026 Ebenezer Tax Services. Documento Confidencial.