



# Instrucciones de Inicio - Aurum Control Center

---

## Inicio Rápido (5 minutos)

---

### Paso 1: Levantar Servicios Docker

```
cd /home/ubuntu/aurum-control-center  
docker-compose up -d
```

Esto iniciará:

- ☒ PostgreSQL en puerto 5432
- ☒ PgBouncer en puerto 6432
- ☒ Redis en puerto 6379

**Verificar que estén corriendo:**

```
docker-compose ps
```

### Paso 2: Aplicar Schema a Base de Datos

```
npm run db:push
```

Esto creará todas las tablas necesarias.

### Paso 3: Poblar con Datos Iniciales

```
npm run db:seed
```

Esto creará:

- 11 satélites
- 3 usuarios (CEO, Manager, Employee)
- Tareas de ejemplo

### Paso 4: Iniciar Aplicación

```
npm run dev
```

La aplicación estará disponible en: **http://localhost:3000**

### Paso 5: Iniciar Sesión

Abrir navegador en <http://localhost:3000>

**Credenciales CEO:**

- Email: [edwin@aurumcapital.com](mailto:edwin@aurumcapital.com)
- Password: AurumCEO2025!

---

## Comandos Útiles

---

### Desarrollo

```
npm run dev      # Iniciar en desarrollo
npm run build    # Construir para producción
npm run start    # Iniciar en producción
```

### Base de Datos

```
npm run db:studio  # Abrir Prisma Studio (GUI)
npm run db:migrate # Crear migración
npm run db:reset   # Resetear BD (⚠️ elimina datos)
```

### Docker

```
docker-compose up -d  # Iniciar servicios
docker-compose down    # Detener servicios
docker-compose logs -f # Ver logs
docker-compose restart # Reiniciar servicios
```

---

## Verificación de Instalación

---

### 1. Verificar Servicios Docker

```
docker-compose ps
```

Debe mostrar 3 servicios corriendo:

- qhosting\_aurum-control-center-db (PostgreSQL)
- aurum-pgbouncer (PgBouncer)
- qhosting\_aurum-control-center-redis (Redis)

### 2. Verificar Conexión a PostgreSQL

```
./scripts/database/diagnose-db.sh
```

### 3. Verificar Aplicación

Abrir <http://localhost:3000>

Debe mostrar la página de login.

---

## Solución de Problemas

---

### Error: “Cannot connect to PostgreSQL”

```
# Verificar que PostgreSQL esté corriendo
docker-compose ps

# Ver logs
docker-compose logs postgres

# Reiniciar PostgreSQL
docker-compose restart postgres
```

### Error: “Cannot connect to Redis”

```
# Verificar que Redis esté corriendo
docker-compose ps

# Ver logs
docker-compose logs redis

# Reiniciar Redis
docker-compose restart redis
```

### Error: “Port 3000 already in use”

```
# Matar proceso en puerto 3000
lsof -ti:3000 | xargs kill -9

# O cambiar puerto en .env
PORT=3001
```

### Resetear Todo el Sistema

```
# Detener servicios
docker-compose down

# Eliminar volúmenes (⚠ elimina datos)
docker-compose down -v

# Levantar servicios
docker-compose up -d

# Aplicar schema
npm run db:push

# Poblar datos
npm run db:seed

# Iniciar app
npm run dev
```

---

## Explorando el Sistema

---

### Dashboard Principal

- URL: <http://localhost:3000/dashboard>
- Muestra estadísticas y vista general

### Gestión de Satélites

- URL: <http://localhost:3000/dashboard/satellites>
- CRUD completo de satélites
- Solo accesible para CEO y MANAGER

### Prisma Studio (GUI de BD)

```
npm run db:studio
```

- URL: <http://localhost:5555>
- Interfaz gráfica para explorar BD

---

## Próximos Módulos a Implementar

---

- [ ] Gestión de Tareas con Kanban
- [ ] Sistema de Tickets de Soporte
- [ ] Módulo de Finanzas
- [ ] Dashboard Analítico
- [ ] Notificaciones en Tiempo Real

---

**¡Listo! El sistema está funcionando correctamente.**

Para más información, consultar el README.md principal.