🐧 Guía Completa de Instalación en Servidor Ubuntu

Control de Facturas PWA - Instalación Independiente

Esta guía te permitirá instalar la aplicación **Control de Facturas** en tu propio servidor Ubuntu, completamente independiente de cualquier servicio externo.

📜 Índice

Requisitos del Sistema

Hardware Mínimo:

- **CPU:** 2 cores
- **RAM:** 2 GB
- **Disco:** 20 GB de espacio libre
- **Red:** Conexión a Internet

Hardware Recomendado:

- **CPU:** 4 cores
- **RAM:** 4 GB o más
- **Disco:** 50 GB SSD
- **Red:** Ancho de banda adecuado para tu tráfico esperado

Software:

- **Sistema Operativo:** Ubuntu 20.04 LTS o superior (22.04 LTS recomendado)
- **Acceso:** Acceso root o sudo
- **Dominio:** (Opcional) Un dominio apuntando a la IP del servidor

---

Método 1: Instalación Automática

Paso 1: Conectarse al Servidor

```
ssh usuario@tu-servidor.com
```

Paso 2: Descargar o Copiar el Proyecto

Opción A: Si tienes el proyecto en un repositorio Git:

```
cd /opt
sudo git clone https://tu-repositorio.git control_facturas
cd control_facturas
```

Opción B: Si tienes el proyecto en tu máquina local:

```
# En tu máquina local:
scp -r /ruta/al/proyecto usuario@tu-servidor.com:/opt/control_facturas

# Luego en el servidor:
cd /opt/control_facturas
```

Paso 3: Ejecutar el Script de Instalación

```
sudo chmod +x install.sh
sudo ./install.sh
```

El script instalará automáticamente:
- Node.js 20.x
- Yarn
- PostgreSQL
- PM2
- Nginx
- Configurará el firewall

Durante la instalación, se te pedirá:
- **Usuario de PostgreSQL** (default: facturas_user)
- **Contraseña de PostgreSQL**

¡Guarda estos datos! Los necesitarás para el archivo .env

Paso 4: Configurar Variables de Entorno

```
cd nextjs_space
cp .env.example .env
nano .env
```

Edita el archivo `.env` con tus datos:

```
DATABASE_URL="postgresql://facturas_user:tu_contraseña@localhost:5432/con-
trol_facturas"
NEXTAUTH_SECRET="$(openssl rand -base64 32)"
NEXTAUTH_URL="http://tu-dominio.com"
NODE_ENV="production"
PORT=3000
```

Generar NEXTAUTH_SECRET:

```
openssl rand -base64 32
```

Copia el resultado y pégalo en `NEXTAUTH_SECRET`.

Paso 5: Instalar Dependencias y Configurar Base de Datos

```
# Instalar dependencias de Node
yarn install

# Ejecutar migraciones de base de datos
yarn prisma migrate deploy

# Generar cliente Prisma
yarn prisma generate

# (Opcional) Cargar datos de prueba
yarn prisma db seed
```

Paso 6: Construir la Aplicación

```
yarn build
```

Paso 7: Crear Carpeta de Logs

```
mkdir -p logs
```

Paso 8: Iniciar con PM2

```
pm2 start ecosystem.config.js
pm2 save
pm2 startup
```

Paso 9: Verificar que Está Funcionando

```
pm2 status
pm2 logs control-facturas
```

Abre tu navegador y ve a: `http://tu-ip-del-servidor:3000`

Método 2: Instalación Manual

Paso 1: Actualizar Sistema

```
sudo apt update && sudo apt upgrade -y
```

Paso 2: Instalar Node.js 20.x

```
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt install -y nodejs
```

Verificar:

```
node -v  # Debe mostrar v20.x.x
npm -v
```

Paso 3: Instalar Yarn

```
sudo npm install -g yarn
yarn -v
```

Paso 4: Instalar PostgreSQL

```
sudo apt install -y postgresql postgresql-contrib
sudo systemctl start postgresql
sudo systemctl enable postgresql
```

Paso 5: Configurar PostgreSQL

```
sudo -u postgres psql
```

Dentro de psql:

```
-- Crear usuario
CREATE USER facturas_user WITH PASSWORD 'tu_contraseña_segura';

-- Crear base de datos
CREATE DATABASE control_facturas OWNER facturas_user;

-- Otorgar privilegios
GRANT ALL PRIVILEGES ON DATABASE control_facturas TO facturas_user;

-- Salir
\q
```

Paso 6: Instalar PM2

```
sudo npm install -g pm2
```

Paso 7: Instalar Nginx

```
sudo apt install -y nginx
sudo systemctl start nginx
sudo systemctl enable nginx
```

Paso 8: Configurar Firewall

```
sudo ufw allow 'Nginx Full'
sudo ufw allow OpenSSH
sudo ufw enable
```

Pasos 9-15: Igual que Método 1 (Pasos 2-8)

Sigue los pasos del Método 1 desde el Paso 2 hasta el Paso 8.

Configuración de la Aplicación

Archivo .env Completo

El archivo `.env` debe contener:

```
# Base de Datos PostgreSQL
DATABASE_URL="postgresql://facturas_user:tu_contraseña@localhost:5432/con-
trol_facturas"

# NextAuth Secret (genera uno nuevo con: openssl rand -base64 32)
NEXTAUTH_SECRET="tu_secret_generado_con_openssl"

# URL de tu aplicación
# Desarrollo: http://localhost:3000
# Producción con dominio: https://tu-dominio.com
# Producción con IP: http://tu-ip-publica
NEXTAUTH_URL="https://tu-dominio.com"

# Entorno
NODE_ENV="production"

# Puerto (opcional)
PORT=3000
```

Permisos de Archivos

```
cd /opt/control_facturas/nextjs_space
sudo chown -R $USER:$USER .
chmod 600 .env  # Solo el propietario puede leer/escribir
```

Configuración de Nginx

Paso 1: Configurar Nginx como Proxy Inverso

```
sudo nano /etc/nginx/sites-available/control-facturas
```

Pega la configuración del archivo `nginx-config.conf` (incluido en el proyecto) y **reemplaza `tu-dominio.com` con tu dominio real**.

Paso 2: Habilitar el Sitio

```
sudo ln -s /etc/nginx/sites-available/control-facturas /etc/nginx/sites-enabled/
```

Paso 3: Deshabilitar Sitio por Defecto (Opcional)

```
sudo rm /etc/nginx/sites-enabled/default
```

Paso 4: Verificar Configuración

```
sudo nginx -t
```

Debe mostrar:

```
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Paso 5: Reiniciar Nginx

```
sudo systemctl restart nginx
```

Paso 6: Verificar

Abre tu navegador y ve a: `http://tu-dominio.com` o `http://tu-ip`

Configuración SSL con Let’s Encrypt

Requisitos:

- Un dominio apuntando a la IP de tu servidor
- Puertos 80 y 443 abiertos

Paso 1: Instalar Certbot

```
sudo apt install -y certbot python3-certbot-nginx
```

Paso 2: Obtener Certificado SSL

```
sudo certbot --nginx -d tu-dominio.com -d www.tu-dominio.com
```

Sigue las instrucciones:
- Ingresa tu email
- Acepta los términos
- Elige si quieres redirigir HTTP a HTTPS (recomendado: Sí)

Paso 3: Verificar Renovación Automática

```
sudo certbot renew --dry-run
```

Let’s Encrypt configura automáticamente un cron job para renovar certificados.

Paso 4: Actualizar NEXTAUTH_URL

```
cd /opt/control_facturas/nextjs_space
nano .env
```

Cambia:

```
NEXTAUTH_URL="https://tu-dominio.com"
```

Reinicia la aplicación:

```
pm2 restart control-facturas
```

Gestión de la Aplicación con PM2

Comandos Básicos:

```
# Ver estado
pm2 status

# Ver logs en tiempo real
pm2 logs control-facturas

# Ver logs específicos
pm2 logs control-facturas --lines 100

# Reiniciar aplicación
pm2 restart control-facturas

# Detener aplicación
pm2 stop control-facturas

# Iniciar aplicación
pm2 start control-facturas

# Eliminar de PM2
pm2 delete control-facturas

# Ver información detallada
pm2 show control-facturas

# Monitoreo en tiempo real
pm2 monit
```

Guardar Configuración de PM2:

```
pm2 save
```

Configurar PM2 para Iniciar al Arranque:

```
pm2 startup systemd
# Ejecuta el comando que PM2 te muestra
```

Mantenimiento y Actualizaciones

Actualizar la Aplicación

```
cd /opt/control_facturas

# Detener la aplicación
pm2 stop control-facturas

# Hacer backup de la base de datos (recomendado)
sudo -u postgres pg_dump control_facturas > backup_$(date +%Y%m%d_%H%M%S).sql

# Obtener actualizaciones (si usas Git)
git pull origin main

# O copiar archivos actualizados manualmente

cd nextjs_space

# Instalar nuevas dependencias
yarn install

# Ejecutar migraciones (si hay)
yarn prisma migrate deploy
yarn prisma generate

# Reconstruir
yarn build

# Reiniciar
pm2 restart control-facturas
```

Actualizar Node.js

```
# Instalar nueva versión de Node
curl -fsSL https://deb.nodesource.com/setup_20.x | sudo -E bash -
sudo apt update
sudo apt install -y nodejs

# Verificar
node -v

# Reinstalar dependencias globales
sudo npm install -g yarn pm2

# Reconstruir la aplicación
cd /opt/control_facturas/nextjs_space
yarn install
yarn build
pm2 restart control-facturas
```

Backup y Restauración

Backup de la Base de Datos

Crear Backup Manual:

```
sudo -u postgres pg_dump control_facturas > backup_facturas_$(date +%Y%m%d_%H%M%S).sql
```

Crear Backup Automático (Cron Job):

```
crontab -e
```

Agregar:

```
# Backup diario a las 2 AM
0 2 * * * sudo -u postgres pg_dump control_facturas > /backups/facturas_$(date +\%Y\
%m\%d).sql

# Eliminar backups mayores a 30 días
0 3 * * * find /backups -name "facturas_*.sql" -mtime +30 -delete
```

Crear carpeta de backups:

```
sudo mkdir -p /backups
sudo chown postgres:postgres /backups
```

Restaurar Base de Datos

```
# Detener la aplicación
pm2 stop control-facturas

# Restaurar desde backup
sudo -u postgres psql control_facturas < backup_archivo.sql

# Reiniciar aplicación
pm2 start control-facturas
```

Backup Completo del Proyecto

```
# Backup del proyecto completo
tar -czf backup_completo_$(date +%Y%m%d).tar.gz /opt/control_facturas
```

Solución de Problemas

La aplicación no inicia

Verificar logs:

```
pm2 logs control-facturas --lines 50
```

Verificar puerto:

```
sudo lsof -i :3000
```

Verificar variables de entorno:

```
cat /opt/control_facturas/nextjs_space/.env
```

Error de conexión a base de datos

Verificar que PostgreSQL esté corriendo:

```
sudo systemctl status postgresql
```

Probar conexión manual:

```
psql -U facturas_user -d control_facturas -h localhost
```

Verificar DATABASE_URL en .env:

```
cat .env | grep DATABASE_URL
```

Nginx muestra error 502 Bad Gateway

Verificar que la aplicación esté corriendo:

```
pm2 status
curl http://localhost:3000
```

Verificar logs de Nginx:

```
sudo tail -f /var/log/nginx/error.log
```

Verificar configuración de Nginx:

```
sudo nginx -t
```

Error de autenticación (login no funciona)

Verificar NEXTAUTH_SECRET:

```
cat .env | grep NEXTAUTH_SECRET
```

Verificar NEXTAUTH_URL:

```
cat .env | grep NEXTAUTH_URL
```

Debe coincidir con la URL que usas en el navegador.

La aplicación consume mucha memoria

Ver uso de memoria:

```
pm2 monit
```

Ajustar límite de memoria en ecosystem.config.js:

```
max_memory_restart: '512M',  // Ajustar según disponibilidad
```

Reiniciar:

```
pm2 restart control-facturas
```

Certificado SSL no se renueva automáticamente

Verificar servicio de renovación:

```
sudo systemctl status certbot.timer
```

Renovar manualmente:

```
sudo certbot renew
```

Ver logs de Certbot:

```
sudo cat /var/log/letsencrypt/letsencrypt.log
```

Limpiar logs de PM2

```
pm2 flush control-facturas
```

Reiniciar todo el stack

```
# Detener aplicación
pm2 stop control-facturas

# Reiniciar PostgreSQL
sudo systemctl restart postgresql

# Reiniciar Nginx
sudo systemctl restart nginx

# Iniciar aplicación
pm2 start control-facturas
```

📞 Soporte y Recursos

Logs Importantes:

- **Aplicación:** `pm2 logs control-facturas`
- **Nginx Access:** `/var/log/nginx/control-facturas-access.log`
- **Nginx Error:** `/var/log/nginx/control-facturas-error.log`
- **PostgreSQL:** `/var/log/postgresql/postgresql-*.log`
- **Sistema:** `journalctl -xe`

Comandos Útiles de Diagnóstico:

```
# Ver puertos en uso
sudo netstat -tulpn | grep LISTEN

# Ver procesos de Node
ps aux | grep node

# Ver uso de disco
df -h

# Ver uso de memoria
free -h

# Ver carga del sistema
htop
```

🎉 ¡Instalación Completa!

Tu aplicación **Control de Facturas** ahora está corriendo en tu propio servidor Ubuntu de forma completamente independiente.

Acceso:

- **URL:** https://tu-dominio.com (o http://tu-ip:3000)
- **Credenciales de prueba:** `john@doe.com` / `johndoe123`

Próximos Pasos:

1. ✅ Crear tu propia cuenta de usuario
2. ✅ Eliminar o cambiar credenciales de prueba
3. ✅ Configurar backups automáticos
4. ✅ Configurar monitoreo (opcional)
5. ✅ Ajustar configuración según tus necesidades

¡Disfruta de tu aplicación! 🚀