



MANUAL DE INSTALACIÓN COMPLETO - VIRA

Esta guía te llevará paso a paso para instalar y configurar VIRA desde cero en cualquier entorno.



REQUISITOS DEL SISTEMA

Mínimos

- **Node.js:** v18.17.0 o superior
- **Memoria RAM:** 4GB mínimo
- **Espacio en disco:** 2GB libres
- **Sistema operativo:** Windows 10+, macOS 10.15+, Linux Ubuntu 20.04+

Recomendados para Producción

- **Node.js:** v20.x LTS
- **Memoria RAM:** 8GB o más
- **CPU:** 4 cores mínimo
- **Almacenamiento:** SSD con 10GB libres
- **Base de datos:** PostgreSQL 14+



PASO 1: PREPARACIÓN DEL ENTORNO

1.1 Instalar Node.js y Yarn

En Windows:

```
# Descargar e instalar Node.js desde https://nodejs.org
# Luego instalar Yarn
npm install -g yarn
```

En macOS:

```
# Usando Homebrew
brew install node yarn
```

En Linux (Ubuntu):

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

# Instalar Yarn
npm install -g yarn
```

1.2 Verificar Instalación

```
node --version # Debe mostrar v18+ o v20+
yarn --version # Debe mostrar 1.22+ o 4.0+
```

PASO 2: CONFIGURACIÓN DEL PROYECTO

2.1 Descomprimir y Preparar Archivos

```
# Si tienes un ZIP, descomprime
unzip VIRA_BACKUP_COMPLETO.zip
cd VIRA_BACKUP_COMPLETO

# O si clonaste desde Git
git clone <tu-repositorio> vira
cd vira
```

2.2 Instalar Dependencias

```
# Instalar todas las dependencias
yarn install

# Esto puede tomar 2-3 minutos
```

PASO 3: CONFIGURACIÓN DE BASE DE DATOS

Opción A: Supabase (Recomendado)

3A.1 Crear Proyecto en Supabase

1. Ve a supabase.com (<https://supabase.com>) y regístrate
2. Crea un nuevo proyecto:
 - **Nombre:** "VIRA Production"
 - **Región:** São Paulo (más cercana a Chile)
 - **Contraseña:** Genera una segura y guárdala

3A.2 Configurar Schema

1. En Supabase Dashboard, ve a **SQL Editor**
2. Abre el archivo `DATABASE/supabase_schema.sql`
3. Copia TODO el contenido y pégalo en SQL Editor
4. Ejecuta la query (toma 1-2 minutos)

3A.3 Obtener Credenciales

En **Settings > API** copia:

- Project URL
- Anon Public Key
- Service Role Key (secret)

Opción B: PostgreSQL Local

3B.1 Instalar PostgreSQL

En Windows:

```
# Descargar desde https://www.postgresql.org/download/windows/
# Instalar con contraseña memorable
```

En macOS:

```
brew install postgresql
brew services start postgresql
createdb vira_development
```

En Linux:

```
sudo apt update
sudo apt install postgresql postgresql-contrib
sudo -u postgres createdb vira_development
```

3B.2 Configurar Base de Datos

```
-- Conectar como usuario postgres
psql -U postgres

-- Crear base de datos y usuario
CREATE DATABASE vira_production;
CREATE USER vira_user WITH ENCRYPTED PASSWORD 'tu_password_segura';
GRANT ALL PRIVILEGES ON DATABASE vira_production TO vira_user;
\q
```



PASO 4: CONFIGURACIÓN DE VARIABLES DE ENTORNO

4.1 Crear Archivo .env

```
# Copiar template
cp .env.example .env

# O crear manualmente
touch .env
```

4.2 Configurar Variables Esenciales

Edita el archivo `.env` con tu editor favorito:

```
# === BASE DE DATOS ===
# Para Supabase:
NEXT_PUBLIC_SUPABASE_URL=https://tu-proyecto-id.supabase.co
NEXT_PUBLIC_SUPABASE_ANON_KEY=tu_anon_key_aquí
SUPABASE_SERVICE_ROLE_KEY=tu_service_role_key_aquí

# Para PostgreSQL local:
DATABASE_URL="postgresql://vira_user:tu_password@localhost:5432/vira_production"

# === AUTENTICACIÓN ===
NEXTAUTH_SECRET="tu_secreto_super_seguro_aquí"
NEXTAUTH_URL=http://localhost:3000 # Cambia en producción

# === INTELIGENCIA ARTIFICIAL ===
ABACUSAI_API_KEY="77af0fb805d34069b609ef8baea62041"

# === SÍNTESIS DE VOZ (Configurar al menos una) ===
# ElevenLabs (Recomendado para calidad)
ELEVENLABS_API_KEY=tu_elevenlabs_key_aquí

# Azure Speech (Voces chilenas)
AZURE_SPEECH_KEY=tu_azure_key_aquí
AZURE_SPEECH_REGION=eastus

# Amazon Polly (Económico)
AWS_ACCESS_KEY_ID=tu_aws_access_key
AWS_SECRET_ACCESS_KEY=tu_aws_secret_key
AWS_REGION=us-east-1

# === ALMACENAMIENTO ===
AWS_BUCKET_NAME=tu-bucket-name
AWS_FOLDER_PREFIX=development/
```

4.3 Configurar APIs Opcionales (Puedes configurar después)

```
# === PAGOS MERCADOPAGO ===
MERCADOPAGO_ACCESS_TOKEN=tu_access_token_aquí
MERCADOPAGO_PUBLIC_KEY=tu_public_key_aquí

# === SOCIAL MEDIA ===
TWITTER_BEARER_TOKEN=tu_twitter_token_aquí
FACEBOOK_APP_ID=tu_facebook_app_id

# === NOTIFICACIONES ===
SENDGRID_API_KEY=tu_sendgrid_key_aquí
SENDGRID_FROM_EMAIL=notificaciones@tu-dominio.com
```

PASO 5: CONFIGURACIÓN DE SERVICIOS IA

5.1 AbacusAI (Ya incluido)

- API Key ya está incluida en el backup
- No necesitas configurar nada adicional

5.2 ElevenLabs (Recomendado)

1. Ve a elevenlabs.io (https://elevenlabs.io)
2. Regístrate y ve a **Profile > API Keys**

3. Crea una nueva API key
4. Agrégala a `.env` como `ELEVENLABS_API_KEY`

5.3 Azure Speech (Voces Chilenas)

1. Ve a portal.azure.com (<https://portal.azure.com>)
2. Crea un recurso **Speech Services**
3. Obtén la **Key** y **Region**
4. Agrégalas a `.env`

5.4 AWS Polly (Económico)

1. Ve a aws.amazon.com (<https://aws.amazon.com>)
2. Crea una cuenta y ve a IAM
3. Crea un usuario con permisos para Polly y S3
4. Obtén **Access Key ID** y **Secret**
5. Agrégalos a `.env`



PASO 6: CONFIGURACIÓN DE ALMACENAMIENTO S3

6.1 Crear Bucket S3

```
# En AWS Console:
# 1. Ve a S3
# 2. Crear bucket con nombre único (ej: vira-audio-tu-nombre)
# 3. Configurar permisos públicos de lectura
# 4. Habilitar CORS
```

6.2 Configurar CORS en S3

```
[
  {
    "AllowedHeaders": ["*"],
    "AllowedMethods": ["GET", "PUT", "POST", "DELETE"],
    "AllowedOrigins": ["*"],
    "ExposeHeaders": ["ETag"]
  }
]
```

6.3 Política de Bucket

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::tu-bucket-name/*"
    }
  ]
}
```

PASO 7: INICIALIZACIÓN DE LA BASE DE DATOS

7.1 Para Supabase

```
# Las tablas ya fueron creadas en el paso 3A.2
# Solo verifica que todo esté funcionando:
yarn dev
# Ve a http://localhost:3000 y haz login
```

7.2 Para PostgreSQL Local con Prisma

```
# Generar cliente Prisma
yarn prisma generate

# Aplicar schema
yarn prisma db push

# Seed con datos iniciales (opcional)
yarn prisma db seed
```

PASO 8: PRIMERA EJECUCIÓN

8.1 Iniciar Servidor de Desarrollo

```
yarn dev
```

8.2 Verificar que Todo Funciona

1. Abre **http://localhost:3000**
2. ☒ La página principal debe cargar
3. ☒ Haz clic en “Iniciar Sesión” (debe funcionar)
4. ☒ Ve al Dashboard (debe mostrar métricas vacías)
5. ☒ Ve a “Crear Noticiero” (debe cargar el formulario)

8.3 Prueba Funcional Básica

```
# 1. Hacer login con Google/GitHub
# 2. Crear una plantilla básica
# 3. Generar un noticiero de prueba (5 minutos)
# 4. Verificar que se genere timeline
# 5. Revisar que se guarde en base de datos
```

PASO 9: VERIFICACIÓN DE FUNCIONALIDADES

9.1 Checklist de APIs

- ☐ ☒ Autenticación funciona
- ☐ ☒ Base de datos conecta
- ☐ ☒ AbacusAI responde (generación de texto)
- ☐ ☒ ElevenLabs/Azure (síntesis de voz)
- ☐ ☒ S3 (subida de archivos)

- [] ☒ Scraping de noticias

9.2 Verificar Logs

```
# En otra terminal, monitorear logs:
tail -f .next/server.log

# 0 para desarrollo:
yarn dev # Los logs aparecen en tiempo real
```

9.3 Test de APIs Individuales

```
# Probar endpoint de noticias
curl http://localhost:3000/api/news-sources

# Probar generación simple
curl -X POST http://localhost:3000/api/generate-newscast \
  -H "Content-Type: application/json" \
  -d '{"duration": 5, "region": "Santiago"}'
```



PASO 10: SOLUCIÓN DE PROBLEMAS COMUNES

10.1 Error: “Module not found”

```
# Limpiar cache y reinstalar
rm -rf node_modules yarn.lock
yarn install
```

10.2 Error: “Database connection failed”

```
# Verificar variables de entorno
cat .env | grep DATABASE

# Para Supabase, verificar URLs y keys
# Para PostgreSQL, verificar que el servicio esté corriendo
```

10.3 Error: “API key invalid”

```
# Verificar cada API key en .env
# Probar keys individualmente en Postman/curl
```

10.4 Error: “S3 bucket not accessible”

```
# Verificar permisos del bucket
# Confirmar CORS configurado
# Verificar credenciales AWS
```

10.5 Performance Lenta

```
# Verificar que estés usando yarn dev en desarrollo
# En producción, usar yarn build && yarn start
# Verificar conexión a internet para APIs externas
```

PASO 11: CONFIGURACIÓN PARA PRODUCCIÓN

11.1 Variables de Producción

```
# Actualizar .env para producción:
NEXTAUTH_URL=https://tu-dominio.com
NODE_ENV=production
NEXT_PUBLIC_APP_URL=https://tu-dominio.com

# Cambiar S3 folder a production/
AWS_FOLDER_PREFIX=production/
```

11.2 Build de Producción

```
# Construir aplicación optimizada
yarn build

# Probar build localmente
yarn start

# Debe estar disponible en http://localhost:3000
```










11.3 Deploy (Vercel/Netlify)

```
# Para Vercel
npm i -g vercel
vercel

# Para Netlify
npm i -g netlify-cli
netlify deploy --prod
```

CHECKLIST FINAL DE INSTALACIÓN

Antes de considerar la instalación completa, verifica:

- []  Node.js y Yarn instalados correctamente
- []  Proyecto descargado y dependencias instaladas
- []  Base de datos configurada (Supabase o PostgreSQL)
- []  Variables de entorno configuradas
- []  Al menos una API de IA configurada
- []  S3 configurado para almacenamiento
- []  Servidor de desarrollo funciona (`yarn dev`)
- []  Login/autenticación funciona
- []  Dashboard carga correctamente

- [] ☒ Formulario de creación funciona
- [] ☒ Generación básica de noticiero funciona
- [] ☒ Audio se genera y reproduce
- [] ☒ Timeline interactivo funciona
- [] ☒ Build de producción funciona (`yarn build`)

SOPORTE POST-INSTALACIÓN

Si encuentras problemas:

1. **Revisa los logs** en consola del navegador y terminal
2. **Verifica variables** de entorno una por una
3. **Prueba APIs** individualmente con curl/Postman
4. **Consulta documentación** en `DOCUMENTACION/MANUAL_TECNICO.md`
5. **Revisa FAQ** en `DOCUMENTACION/FAQ.md`

¡VIRA está listo para revolucionar tu producción de contenido! 🗣️✨

Tiempo estimado de instalación completa: 30-60 minutos

Dificultad: Intermedia (requiere conocimientos básicos de desarrollo web)