CUENTY Backend - Guía de Instalación y Configuración

📋 Requisitos Previos

- Node.js 16+ instalado
- PostgreSQL 12+ instalado y corriendo
- Git (para control de versiones)

🚀 Instalación

1. Clonar o navegar al proyecto

```
cd /home/ubuntu/cuenty_mvp/backend
```

2. Instalar dependencias

```
npm install
```

3. Configurar variables de entorno

Crear archivo `.env` en la raíz del backend:

```
# Servidor
NODE_ENV=development
PORT=3000

# Base de datos (Opción 1: URL completa)
DATABASE_URL=postgresql://usuario:password@localhost:5432/cuenty_db?sslmode=disable

# O Base de datos (Opción 2: Variables individuales)
# DB_HOST=localhost
# DB_PORT=5432
# DB_NAME=cuenty_db
# DB_USER=postgres
# DB_PASSWORD=postgres

# JWT Secret (cambiar en producción)
JWT_SECRET=tu-secreto-super-seguro-cambiar-en-produccion-123456

# CORS
CORS_ORIGIN=*
```

4. Crear la base de datos

```
# Conectarse a PostgreSQL
psql -U postgres

# Crear base de datos
CREATE DATABASE cuenty_db;

# Salir de psql
\q
```

5. Ejecutar el esquema de base de datos

```
psql -U postgres -d cuenty_db -f ../database/schema.sql
```

Esto creará todas las tablas, índices y datos de ejemplo incluyendo:
- 5 servicios (Netflix, Disney+, HBO Max, Prime Video, Spotify)
- 20 planes (4 por servicio: 1, 3, 6, 12 meses)
- Usuario admin (username: admin, password: admin123)
- Instrucciones de pago por defecto

6. Verificar la instalación

```
# Iniciar el servidor
npm start

# O en modo desarrollo (con nodemon)
npm run dev
```

El servidor debería iniciar en `http://localhost:3000`

7. Probar el API

```
# Verificar salud del servidor
curl http://localhost:3000/health

# Ver información del API
curl http://localhost:3000/

# Login admin
curl -X POST http://localhost:3000/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{"username":"admin","password":"admin123"}'
```

🔧 Estructura del Proyecto

```
backend/
├── config/
│   └── database.js          # Configuración de PostgreSQL
├── controllers/
│   ├── authController.js    # Auth admin (legacy)
│   ├── authEnhancedController.js  # Auth usuarios con teléfono
│   ├── servicioController.js      # Gestión de servicios
│   ├── servicePlanController.js   # Gestión de planes
│   ├── cartController.js          # Carrito de compras
│   └── ordenEnhancedController.js # Órdenes mejoradas
├── middleware/
│   ├── auth.js              # Middleware de autenticación
│   └── validation.js        # Validaciones de entrada
├── models/
│   ├── Usuario.js           # Modelo de usuario
│   ├── PhoneVerification.js # Verificación telefónica
│   ├── Servicio.js          # Modelo de servicio
│   ├── ServicePlan.js       # Modelo de plan
│   ├── ShoppingCart.js      # Modelo de carrito
│   └── OrdenEnhanced.js     # Modelo de orden mejorado
├── routes/
│   ├── authRoutes.js        # Rutas auth admin
│   ├── authEnhancedRoutes.js  # Rutas auth usuarios
│   ├── servicioRoutes.js    # Rutas de servicios
│   ├── servicePlanRoutes.js # Rutas de planes
│   ├── cartRoutes.js        # Rutas de carrito
│   └── ordenEnhancedRoutes.js # Rutas de órdenes
├── server.js                # Punto de entrada
├── package.json
├── API_DOCUMENTATION.md     # Documentación completa
└── SETUP_GUIDE.md           # Esta guía
```

🔐 Seguridad

Cambiar contraseña de admin

```sql
-- Conectarse a la base de datos
psql -U postgres -d cuenty_db

-- Generar hash de nueva contraseña (usar bcryptjs)
-- Ejemplo: para "miNuevaPassword123"
UPDATE admins
SET password = '$2a$10$NuevoHashGeneradoAqui'
WHERE username = 'admin';
```

Cambiar JWT_SECRET

1. Generar un secreto aleatorio fuerte:

```
node -e "console.log(require('crypto').randomBytes(64).toString('hex'))"
```

1. Actualizar en `.env`:

```
JWT_SECRET=tu_nuevo_secreto_generado
```

📱 Configuración de SMS/WhatsApp (Producción)

Para enviar códigos de verificación por SMS o WhatsApp en producción:

Opción 1: Twilio

1. Crear cuenta en Twilio (https://www.twilio.com)
2. Obtener Account SID, Auth Token y número de teléfono
3. Instalar SDK: `npm install twilio`
4. Modificar `controllers/authEnhancedController.js`:

```
const twilio = require('twilio');
const client = twilio(process.env.TWILIO_ACCOUNT_SID, process.env.TWILIO_AUTH_TOKEN);

// En solicitarCodigo(), después de generar el código:
if (process.env.NODE_ENV === 'production') {
  await client.messages.create({
    body: `Tu código de verificación CUENTY es: ${verification.codigo}`,
    from: process.env.TWILIO_PHONE_NUMBER,
    to: `+52${celular}`
  });
}
```

Opción 2: WhatsApp API

1. Configurar WhatsApp Business API
2. Usar n8n (ya configurado en el proyecto) para enviar mensajes
3. Modificar el código para hacer POST al webhook de n8n

🗄️ Backup de Base de Datos

Crear backup

```
pg_dump -U postgres cuenty_db > backup_$(date +%Y%m%d).sql
```

Restaurar backup

```
psql -U postgres -d cuenty_db < backup_20241017.sql
```

🔄 Actualizaciones del Esquema

Si necesitas actualizar el esquema de la base de datos:

1. Crear archivo de migración en `database/migrations/`
2. Aplicar migración:

```
psql -U postgres -d cuenty_db -f database/migrations/001_nueva_feature.sql
```

📊 Monitoreo

Ver logs del servidor

```
# Si usas pm2
pm2 logs cuenty-backend

# O con nodemon en desarrollo
npm run dev
```

Verificar conexión a base de datos

```
psql -U postgres -d cuenty_db -c "SELECT COUNT(*) FROM servicios;"
```

🧪 Testing

Probar endpoints

Ver ejemplos completos en `API_DOCUMENTATION.md`

```
# Test de health
curl http://localhost:3000/health

# Test de servicios activos
curl http://localhost:3000/api/servicios/activos

# Test de planes
curl http://localhost:3000/api/planes/activos
```

🚀 Despliegue

Con PM2 (Producción)

```
# Instalar PM2 globalmente
npm install -g pm2

# Iniciar aplicación
pm2 start server.js --name cuenty-backend

# Configurar inicio automático
pm2 startup
pm2 save

# Monitorear
pm2 monit
```

Con Docker (Opcional)

```
# Crear Dockerfile en la raíz del backend
FROM node:16-alpine
WORKDIR /app
COPY package*.json ./
RUN npm install --production
COPY . .
EXPOSE 3000
CMD ["node", "server.js"]
```

```
# Construir y ejecutar
docker build -t cuenty-backend .
docker run -p 3000:3000 --env-file .env cuenty-backend
```

🐛 Troubleshooting

Error: “Cannot connect to database”

1. Verificar que PostgreSQL esté corriendo:

```
sudo systemctl status postgresql
```

1. Verificar credenciales en `.env`

2. Verificar que la base de datos existe:

```
psql -U postgres -l | grep cuenty_db
```

Error: “Port 3000 already in use”

```
# Encontrar proceso usando el puerto
lsof -i :3000

# Matar proceso
kill -9 <PID>

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

Error: “JWT Secret not configured”

Asegurarse de tener `JWT_SECRET` en `.env`

📞 Soporte

- Documentación API: `/backend/API_DOCUMENTATION.md`
- GitHub Issues: [Crear issue]
- Email: admin@cuenty.com

¡Listo para usar! 🎉