

# Manual de instalación y despliegue en Railway

**Proyecto (ejemplo):** Sistema de Seguridad (MySQL + Backend Node.js/Express)

## Objetivo:

1. Montar una base de datos MySQL en Railway y migrar una BD local (dump → nube).
2. Desplegar un backend Node.js/Express en Railway y conectarlo a la BD MySQL.

**Audiencia:** equipos de desarrollo/DevOps.

## Pre-requisitos

- Cuenta en Railway.
- Repositorio GitHub con el backend.
- Tener **la base de datos ya creada y funcionando en local** (MySQL) para poder exportarla.
  - Conocer:
    - nombre de BD local (ejemplo: `app_db_local`)
    - usuario local (ejemplo: `root`)
    - host/puerto local (ejemplo: `127.0.0.1:3306`)
- Contar con un **archivo dump** (o la posibilidad de generarlo) y espacio en disco para guardarlo.
- Tener permisos para conectarte a la BD local (credenciales válidas).
- Windows (PowerShell/CMD) con:
  - `mysql.exe` y `mysqldump.exe` instalados (MySQL Server/Client o Workbench + MySQL binaries).
- Conexión estable a internet.

### Nota sobre seguridad:

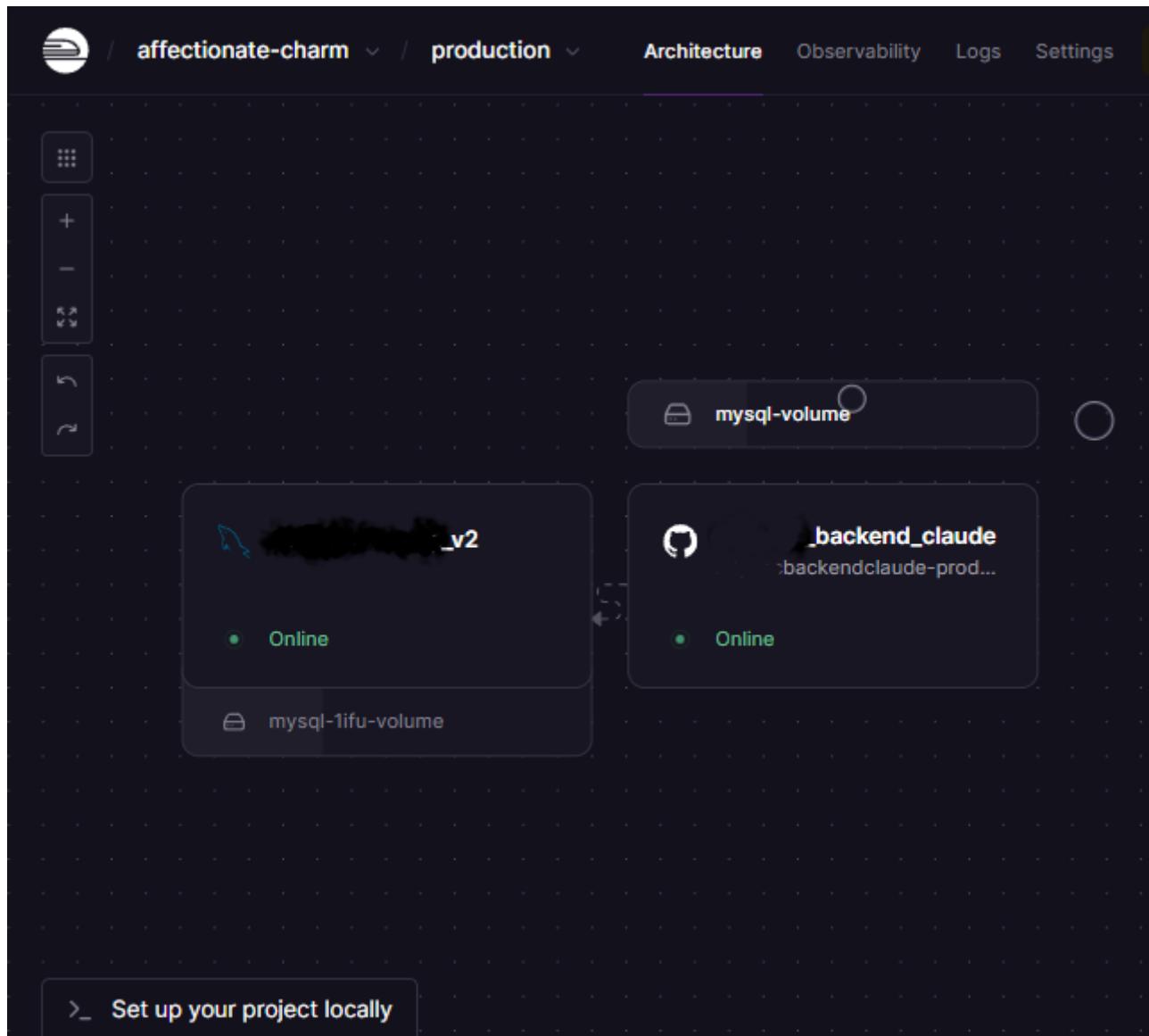
- NO compartas públicamente URLs tipo `mysql://user:password@host:port/db`.
- Si alguna credencial se expone, **rota/regenera** el password en Railway inmediatamente.

## PARTE 1 — MySQL en Railway (migración desde local)

### 1. Crear el servicio MySQL en Railway

1. En Railway → tu **Project** → **Create**.
2. Elige **Database** → **MySQL**.
3. Espera a que el servicio quede en estado **Online**.

Imagen 1 — Arquitectura del proyecto en Railway



## 2. Identificar variables de conexión en Railway

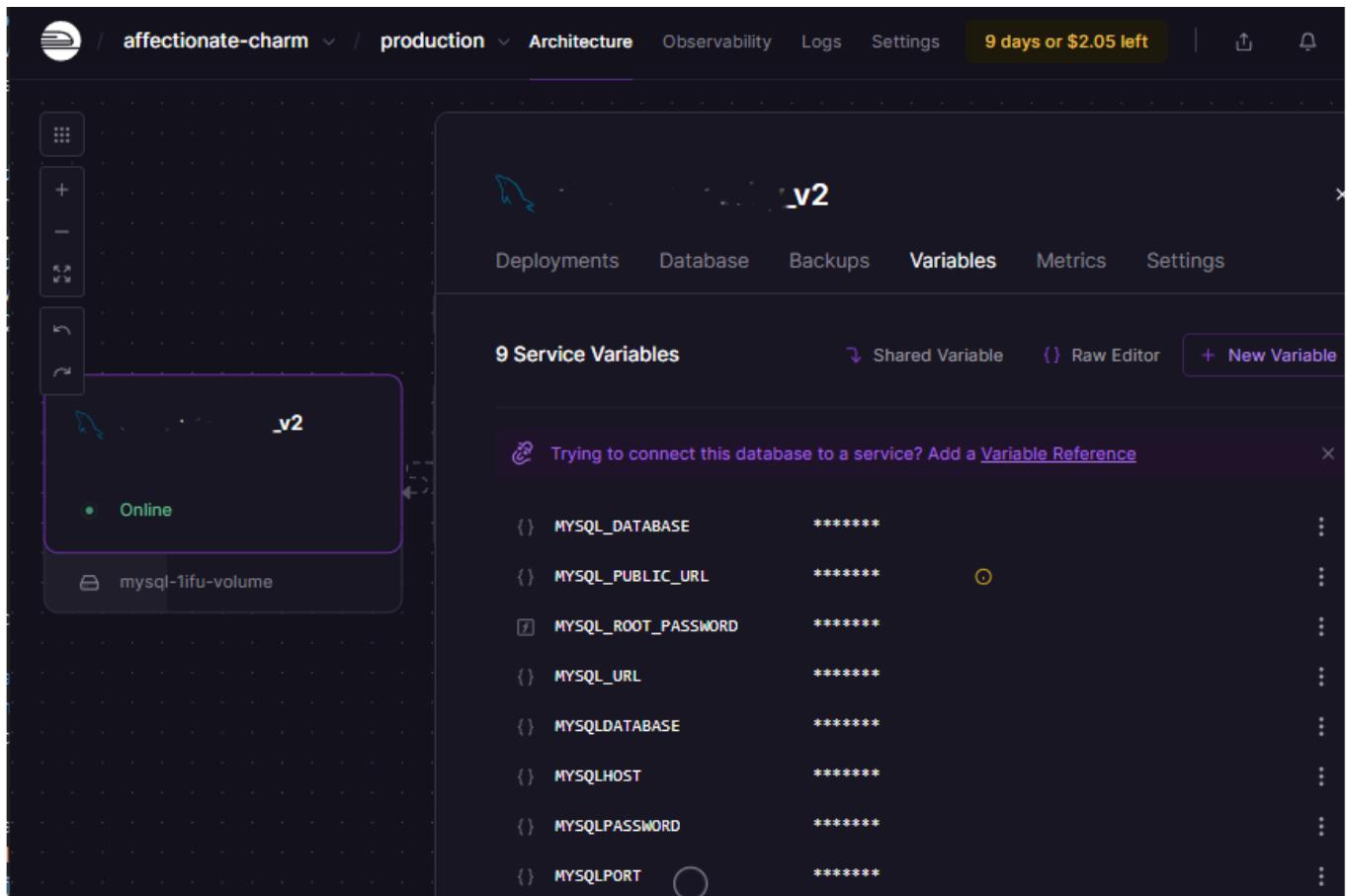
En el servicio **MySQL** → pestaña **Variables** encontrarás (nombres pueden variar ligeramente):

- **MYSQLHOST** (host interno, típico: `mysql.railway.internal`)
- **MYSQLPORT** (puerto interno, típico: `3306`)
- **MYSQLUSER**
- **MYSQLPASSWORD**
- **MYSQLDATABASE** (ejemplo: `app_db_cloud`)

Y variables de acceso público (desde tu PC):

- **MYSQL\_PUBLIC\_URL / URL PÚBLICA DE MYSQL** (incluye usuario y password)

Imagen 2 — Variables del servicio MySQL



### 3. Conectar desde tu PC (MySQL Workbench o CLI)

#### 3.1 Conexión recomendada desde tu PC

Usa **la URL pública** del MySQL (proxy) solo para administración desde tu PC:

- Host similar a \*.proxy.rlwy.net
- Puerto dinámico

**Importante:** El host mysql.railway.internal **NO funciona** desde tu PC. Solo funciona entre servicios dentro de Railway.

#### 3.2 Probar conectividad por CLI (Windows)

1. Ubica **mysql.exe** si Windows no lo reconoce:

```
where /r "C:\Program Files" mysql.exe
```

2. Prueba conexión con host/puerto públicos (ejemplo):

```
"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql.exe" --protocol=TCP -h <HOST_PUBLICO> -P <PUERTO_PUBLICO> -u <USUARIO> -p -e "SELECT 1;"
```

**Resultado esperado:** devuelve 1.

## 4. Exportar (dump) desde tu MySQL local

### 4.1 Dump completo (recomendado)

```
mysqldump -h <LOCAL_HOST> -P <LOCAL_PORT> -u <LOCAL_USER> -p <LOCAL_DB_NAME> >  
C:\RUTA\dump_v2.sql
```

### 4.2 Si el dump falla por triggers (recomendación práctica)

Cuando un dump incluye triggers que referencian columnas inexistentes o cambiaste nombres de columnas, puede fallar al importar.

Para exportar **sin triggers** (y luego recrearlos manualmente):

```
mysqldump -h <LOCAL_HOST> -P <LOCAL_PORT> -u <LOCAL_USER> -p --databases  
<LOCAL_DB_NAME> --routines --events --skip-triggers >  
C:\RUTA\dump_v2_NOTRIGGERS.sql
```

## 5. Importar el dump en Railway

### 5.1 (Opcional) Hacer backup del estado actual de Railway

```
"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqldump.exe" --protocol=TCP -h  
<HOST_PUBLICO> -P <PUERTO_PUBLICO> -u <USUARIO> -p <MYSQLDATABASE> >  
C:\RUTA\railway_backup.sql
```

### 5.2 Limpiar la base de datos en Railway (si hubo imports fallidos)

```
"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql.exe" --protocol=TCP -h  
<HOST_PUBLICO> -P <PUERTO_PUBLICO> -u <USUARIO> -p -e "DROP DATABASE  
<MYSQLDATABASE>; CREATE DATABASE <MYSQLDATABASE>;"
```

Si no tienes permisos para **DROP DATABASE**, entonces se debe dropear tablas una por una (alternativa avanzada).

### 5.3 Importar

```
"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql.exe" --protocol=TCP -h  
<HOST_PUBLICO> -P <PUERTO_PUBLICO> -u <USUARIO> -p <MYSQLDATABASE> <
```

```
C:\RUTA\dump_v2.sql
```

## 5.4 Validar tablas

```
"C:\Program Files\MySQL\MySQL Server 8.0\bin\mysql.exe" --protocol=TCP -h <HOST_PUBLICO> -P <PUERTO_PUBLICO> -u <USUARIO> -p <MYSQLDATABASE> -e "SHOW TABLES;"
```

Imagen 3 — Validación visual (tab Variables del Backend)

Variable	Value
API_VERSION	*****
BCRYPT_ROUNDS	*****
CORS_ORIGIN	*****
DB_DIALECT	*****
DB_HOST	*****
DB_NAME	*****
DB_PASSWORD	*****
DB_PORT	*****

## 6. Problemas comunes y solución

### 6.1 mysql no se reconoce en CMD

- El cliente no está en el PATH.
- Solución: usar ruta completa a `mysql.exe` o agregar `...\\MySQL Server 8.0\\bin` al PATH.

### 6.2 ERROR 2013 ... reading initial communication packet

- Host/puerto incorrecto (probable), servicio recreado y cambió el proxy.
- Solución: copiar nuevamente `MYSQL_PUBLIC_URL` y extraer host/puerto correctos.

### 6.3 Unknown column 'X' in 'NEW' durante import

- Trigger en el dump referencia una columna inexistente.
- Solución recomendada:
  - Regenerar dump con `--skip-triggers`.
  - Importar.
  - Crear triggers corregidos manualmente.

## 6.4 Acceso denegado al generar dump con >

- Windows no permite escribir en `C:\` sin permisos.
- Solución: exportar a una carpeta de usuario, por ejemplo:

```
mysqldump -u root -p <DB> > "%USERPROFILE%\Desktop\dump.sql"
```

---

# PARTE 2 — Backend Node.js/Express en Railway

---

## 1. Preparar el repo

Verifica en `package.json`:

- `"start": "node src/app.js"`
- `engines.node` compatible (ideal  $\geq 18$ )

Verifica que tu server use:

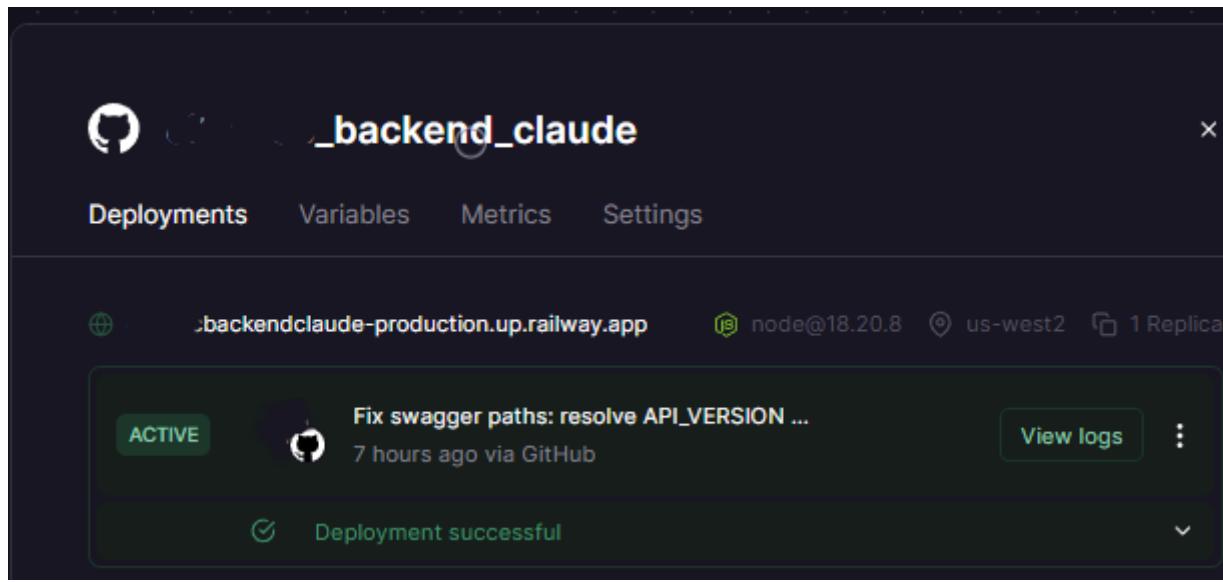
- `const PORT = process.env.PORT || 3000;`
- `app.listen(PORT, ...)`

---

## 2. Crear el servicio Backend desde GitHub

1. En Railway → Project → **Create**.
2. Selecciona **GitHub Repo**.
3. Elige tu repo del backend.
4. Railway detectará Node e intentará build/deploy.

Imagen 5 — Deploy exitoso del backend



### 3. Configurar Variables del Backend (ENV)

#### 3.1 Variables mínimas recomendadas

- `NODE_ENV=production`
- `API_VERSION=v1`

#### **JWT (obligatorias para login)**

- `JWT_SECRET` (string largo aleatorio)
- `JWT_REFRESH_SECRET` (string largo aleatorio)

#### **DB (Sequelize con variables sueltas)**

Si tu `src/config/database.js` usa `DB_HOST/DB_PORT/...`, define:

- `DB_HOST`
- `DB_PORT`
- `DB_USER`
- `DB_PASSWORD`
- `DB_NAME`

#### 3.2 Conectar el backend al MySQL por **Variable Reference** (recomendado)

En el servicio backend → **Variables**:

- Usa **Add Variable Reference** y referencia las variables del servicio MySQL.

Mapeo típico:

- `DB_HOST` → `MYSQLHOST`
- `DB_PORT` → `MYSQLPORT`
- `DB_USER` → `MYSQLUSER`
- `DB_PASSWORD` → `MYSQLPASSWORD`

- `DB_NAME` → `MYSQLDATABASE`

**Importante:**

- No escribir `127.0.0.1` (eso rompe en Railway).
- No usar  `${MYSQLHOST}` como texto: debe ser referencia real (selector), no string literal.

**Seguridad:** si vas a compartir este manual o capturas, asegúrate de no exponer URLs con credenciales, contraseñas, ni nombres internos sensibles. Si alguna credencial se filtra, **rota/regenera** el password/secret en Railway.

---

## 4. Generar dominio público (HTTP)

En servicio backend → **Settings** → **Networking**:

- Click en **Generate Domain**.

El dominio puede quedar pendiente si el servicio está crasheando. Primero resuelve el deploy.

El dominio tiene esta estructura `https://<...>.up.railway.app`.

---

## 5. Healthcheck y verificación

Endpoints típicos:

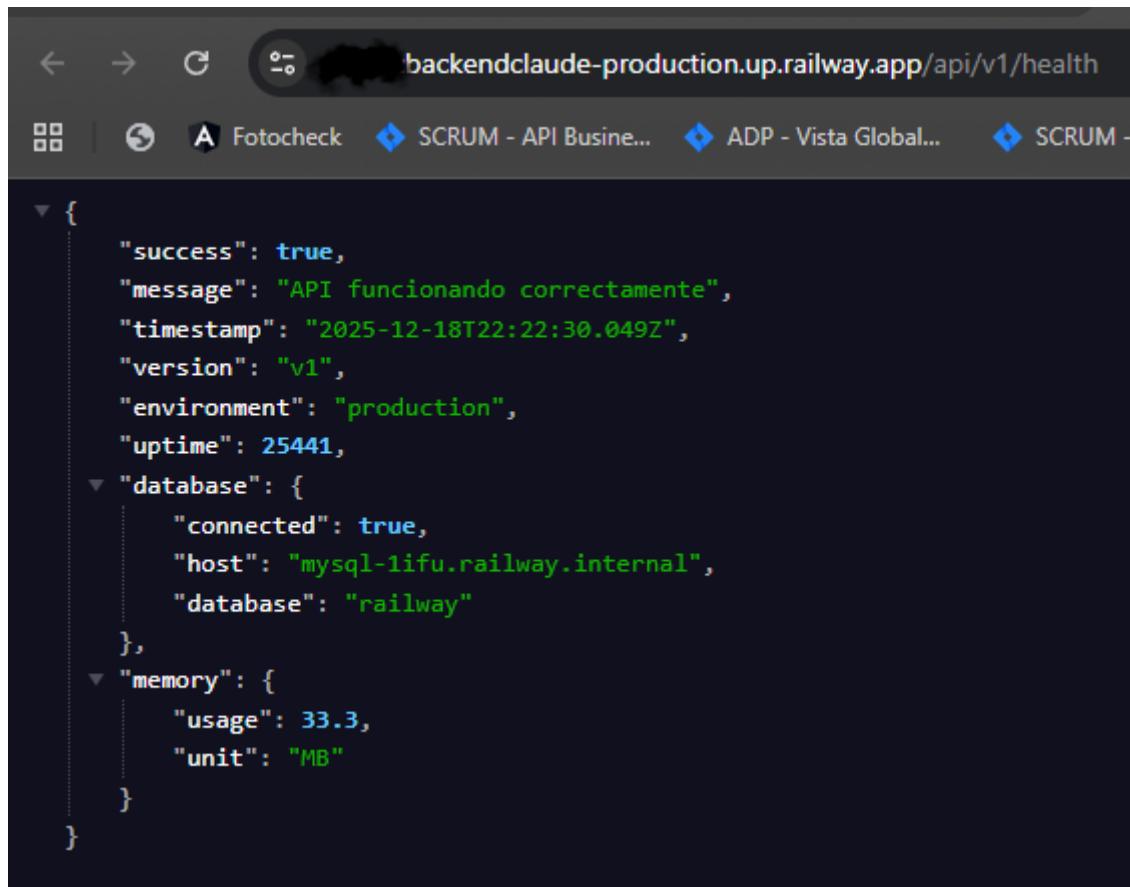
- `GET /health` (simple)
- `GET /api/v1/health` (incluye verificación de DB)

Ejemplo:

- `https://TU-DOMINIO/api/v1/health`

**Resultado esperado:** JSON con `success: true` y `database.connected: true`.

Imagen 6 — Verificación del endpoint `/api/v1/health`



The screenshot shows a browser window with the URL `backendclaude-production.up.railway.app/api/v1/health`. The page displays a JSON response with the following structure:

```
{  
  "success": true,  
  "message": "API funcionando correctamente",  
  "timestamp": "2025-12-18T22:22:30.049Z",  
  "version": "v1",  
  "environment": "production",  
  "uptime": 25441,  
  "database": {  
    "connected": true,  
    "host": "mysql-1ifu.railway.internal",  
    "database": "railway"  
  },  
  "memory": {  
    "usage": 33.3,  
    "unit": "MB"  
  }  
}
```

---

## 6. Swagger (opcional, recomendado)

Si expones Swagger:

- `GET /api/v1/docs`
- `GET /api/v1/docs.json`
- `GET /api/v1/docs.yaml`

Para que Swagger use el dominio correcto:

- Setear `SWAGGER_SERVER_URL=https://TU-DOMINIO` en variables del backend.

URL de ejemplo para abrir Swagger UI (reemplaza `TU-DOMINIO`):

- `https://TU-DOMINIO/api/v1/docs`

Imagen 4 — Swagger UI (documentación del backend)

The screenshot shows the API documentation for the 'System' API. At the top, it says 'API Documentation for System'. Below that, it says 'API Backend para Sistema de Seguridad Ciudadana con autenticación JWT y RBAC'. The 'Servers' dropdown shows 'https://sude-production.up.railway.app - production'. There is a 'Authorize' button. The main area lists endpoints under 'default':

- GET /api/v1/docs.json**
- GET /api/v1/docs.yaml**
- GET /health**
- GET /api/v1**
- GET /api/v1/health**
- GET /api/v1/catalogos/tipos-novedad**
- POST /api/v1/catalogos/tipos-novedad**
- GET /api/v1/catalogos/subtipos-novedad**
- POST /api/v1/catalogos/subtipos-novedad**
- GET /api/v1/catalogos/estados-novedad**
- POST /api/v1/catalogos/estados-novedad**
- GET /api/v1/catalogos/tipos-vehiculo**
- POST /api/v1/catalogos/tipos-vehiculo**

## 7. Problemas comunes y solución

### 7.1 No genera dominio ("Public domain will be generated")

- Casi siempre el servicio no tiene deploy saludable.
- Revisar **Deployments** y **Logs**.

### 7.2 ECONNREFUSED 127.0.0.1:3306

- **DB\_HOST** quedó como **127.0.0.1**.
- Solución: usar **mysql.railway.internal** vía reference a **MYSQLHOST**.

### 7.3 getaddrinfo ENOTFOUND \${MYSQLHOST}

- **DB\_HOST** quedó como texto literal **\${MYSQLHOST}**.
- Solución: usar **Variable Reference** real.

### 7.4 unknown timed out

- Error de conectividad interna (mapeo incorrecto de variables / servicio MySQL distinto).

- Solución: re-hacer referencias **DB\_\*** apuntando al servicio MySQL correcto.
- 

## Checklist final

- MySQL en Railway con tablas cargadas.
- Backend en Railway con **DB\_\*** por variable reference.
- Dominio público generado.
- **GET /api/v1/health** responde OK.
- (Opcional) Swagger en **/api/v1/docs**.