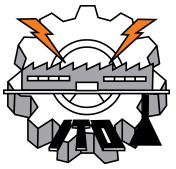

INSTITUTO TECNOLÓGICO DE OAXACA

Manual de APIs: **Replica** de la **Aplicación** de **YouTube** para **Dispositivos Android**

Integración De Procesos De Desarrollo De Software

Prof. Espinosa Pérez Jacob

Ángel Arturo Pérez García

04/11/2025

Introducción

Este proyecto Expo/React Native incluye un módulo backend (Express) con rutas, controladores, middlewares y servicios para autenticación, usuarios, historiales, referencias de video, carga de imágenes y consultas a la API de YouTube.

A su vez, el avance del proyecto esta reflejado en un pequeño dominio, el cual contiene todo lo antes mencionado.

Público

Desarrolladores móviles y backend que integran el cliente RN con las rutas Express.

Alcance

Lo principal de este manual son los servicios de Google como la API de YouTube y los Mapas, además de la autenticación (JWT + Google OAuth), y los demás módulos creados para un mejor manejo de la información de las APIS externas.

Requisitos

Node.js 18+ (para backend).

Dependencias backend: express, axios, jsonwebtoken, bcryptjs, multer, nodemailer, google-auth-library, etc. Para mejores referencias, consultar el manual técnico.

Variables de entorno:

- YOUTUBE_API_KEY (YouTube Data API v3)
- TOKEN_SECRET (JWT de sesiones propias)
- GOOGLE_CLIENT_ID (Google Sign-In)
- EMAIL_USER, EMAIL_PASS (Gmail para envío de códigos)
- NODE_ENV (prod/dev; cookies seguras)
- Almacenamiento de archivos (carpeta uploads).
- Base de datos para modelos: user.model, image.model, record.model, videoReference.model, code.model

Arquitectura de la API

Diagrama por módulos

Capas

Rutas (src/routes/*.route.js): exponen endpoints REST y delegan a controladores.

Controladores (src/controllers/*.controller.js): validan y llaman a servicios/modelos.

Middlewares (src/middlewares/*.middleware.js): validación de token JWT y verificación de correo/código.

Servicios (src/services/*.service.js): integración con YouTube, email, y consultas a modelos.

Patrones de respuesta

Estructura estándar: { success: boolean, message?: string, data?: any }.

Errores: 4xx/5xx con success: false y message.

Autenticaciones

- JWT propio
  Generación de token en POST /auth/login con TOKEN_SECRET.
  Middleware validarToken lee cookie o header y verifica.
- Google OAuth
- YouTube Data API v3
  Autenticación por API Key (YOUTUBE_API_KEY) vía query param key.

Descripción

Módulos principales:

- Usuarios: crear/obtener/actualizar/eliminar.
- Login/logout con JWT.
- Códigos de verificación por correo (envío/verificación).

- Historiales y referencias de videos.
- Imagenes: crear/obtener/eliminar.

URL base

Interno: http://localhost:3000/api

Públicos:

userRoute → /api/users

logRoute → /api/auth

codeRoute → /api/code

imageRoutes → /api/images

recordRoute → /api/records

videoReferenceRoute → /api/references

youtubeRoutes → /api/youtube

API externa YouTube: https://www.googleapis.com/youtube/v3

Estructura de las rutas

- **Autenticación** (src/routes/log.route.js)
    - POST /auth/login → Inicia sesión { correoElectronico, contraseña }, y retorna { token, usuario }.
    - POST /auth/logout → Limpia cookie token.
- **Usuarios** (src/routes/user.route.js)
    - POST /users/enviar → Envía código de verificación por correo (rate limit).
    - POST /users → Crea usuario (middleware verificarCodigo requerido).
    - GET /users → Lista usuarios.

    - GET /users/id → Devuelve id por correo (en body).
    - GET /users/:id → Detalle usuario.
    - PUT /users/:id → Actualiza campos (valida formatos y duplicados).
    - DELETE /users/:id → Elimina usuario.
- Códigos (src/routes/code.route.js)
    - POST /code/enviar → Envío de código.
    - POST /code/verificar → Verificación (marca como usado y valida expiración).
- Imágenes (src/routes/image.route.js)
    - POST /images/subir (multipart image, body idUsuario) → Sube/actualiza imagen de perfil.
    - GET /images/:idUsuario → Devuelve archivo binario.
    - DELETE /images/:idUsuario → Elimina imagen (archivo + registro).
- Historiales (src/routes/record.route.js)
    - POST /records → Crea historial con PK compuesta idUsuario_lat_long.
    - GET /records → Lista historiales.
    - GET /records/:idUsuario → Obtiene historial por PK (el nombre del parámetro sugiere idHistorial).
    - GET /records/:idHistorial/referencias → Referencias por historial.
- Referencias de video (src/routes/videoReference.route.js)
    - POST /references → Crea referencia asociada a historial.
    - GET /references → Lista referencias.
    - GET /references/:id → Detalle referencia.

- YouTube (src/routes/youtube.route.js con src/controllers/youtube.controller.js y src/services/youtube.service.js)
    - GET /youtube/buscar?q=:query → Busca videos por texto.
    - GET /youtube/:id → Obtiene info de video por ID.
    - GET /youtube/cercanos?q=:query&lat=:lat&lon=:lon → Busca por ubicación.
    - GET /youtube/populares?lat=:lat&lon=:lon → Populares cerca de coordenadas.

Entregas

Formatos de respuesta

- Éxito general: { success: true, data: ... } con mensajes dinámicos según el servicio.
- Error general: { success: false, message: string }.

APIs externas usadas en el servidor

YouTube Data API v3:

Base: https://www.googleapis.com/youtube/v3

Recursos: /search, /videos, /channels

Auth: API Key (key).

Endpoints principales

JWT

POST /auth/login body: { correoElectronico, contraseña } → { token, usuario }

POST /auth/logout → 200 sin cuerpo adicional

Usuarios

POST /users/enviar body: { correoElectronico } → envía código

POST /users body: { nombres, correoElectronico, contraseña } (requiere verificarCodigo) → crea usuario

Códigos

POST /code/enviar body: { correoElectronico }

POST /code/verificar body: { correoElectronico, codigo }

Imágenes

POST /images/subir form-data: image + idUsuario

GET /images/:idUsuario → archivo

DELETE /images/:idUsuario

Historiales y referencias

POST /records body: { idUsuario, latitud, longitud }

GET /records | GET /records/:idUsuario

GET /records/:idHistorial/referencias

POST /references body: { idHistorial, idVideo, descripcion? }

GET /references | GET /references/:id

YouTube

GET /youtube/buscar?q=texto&maxResults?=20

GET /youtube/:id

GET /youtube/cercanos?q?=texto&lat=..&lon=..&radio?=10km&maxResults?=20

GET /youtube/populares?lat=..&lon=..&radio?=10km&maxResults?=20