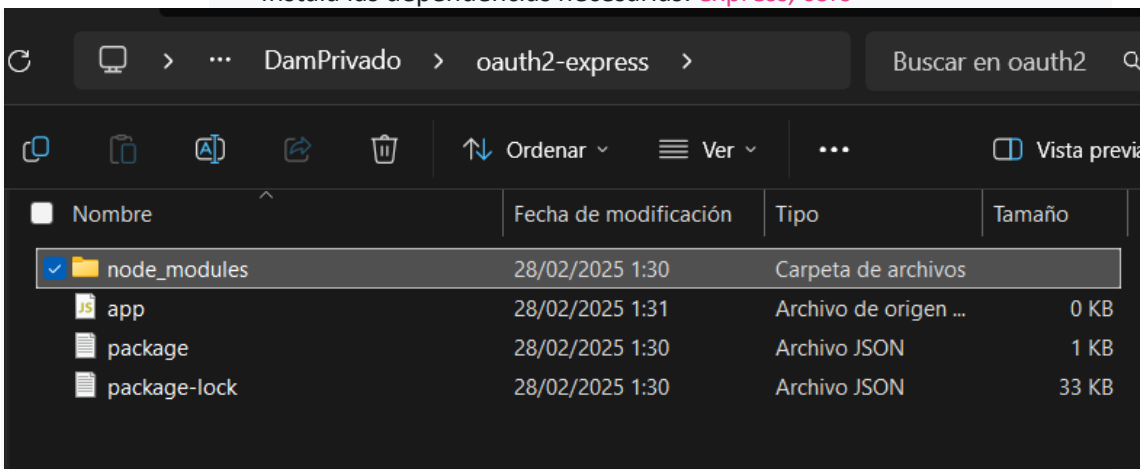
Descripción de la Tarea 27: Implementación de Autenticación OAuth 2.0 en Node.js con Express para comprender en profundidad los tokens de acceso y el funcionamiento del protocolo OAuth 2.0. Implementar un sistema de autenticación OAuth 2.0 en una aplicación Node.js utilizando el framework Express.

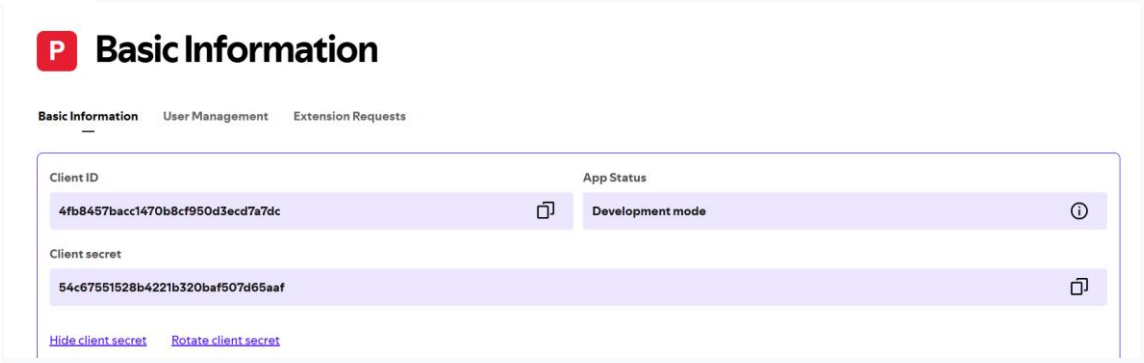
1. **Configuración del Entorno:**
    - Crea un nuevo proyecto Node.js.
    - Instala las dependencias necesarias: express, cors



2. **Implementación de OAuth 2.0:**Implementa las rutas necesarias para iniciar el flujo de autenticación (redirigir al usuario al servidor de autorización) y para manejar la respuesta del servidor de autorización (callback).
3. **Manejo de Tokens y Perfil de Usuario**

Entrega:

- Capturas de pantalla del código implementado (archivos app.js).




```
require("dotenv").config();
const express = require("express");
const axios = require("axios");
const querystring = require("querystring");
const cors = require("cors");

const app = express();
const PORT = process.env.PORT || 12345;
```

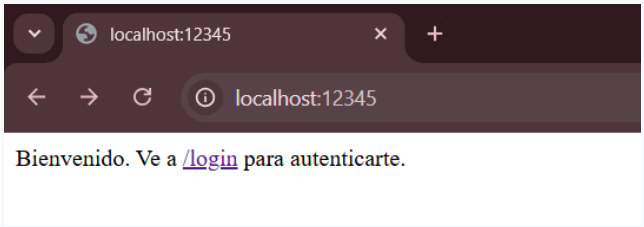
```
const client_id = process.env.SPOTIFY_CLIENT_ID;
const client_secret = process.env.SPOTIFY_CLIENT_SECRET;
const redirect_uri = process.env.REDIRECT_URI;

app.use(cors());
app.use(express.json());

app.get("/", (req, res) => {
  res.send(
    "Bienvenido. Ve a <a href='/login'>/login</a> para autenticarte con
Spotify."
  );
});

app.get("/login", (req, res) => {
  const scope = "user-read-private user-read-email";
  const authUrl =
    "https://accounts.spotify.com/authorize?" +
    querystring.stringify({
      response_type: "code",
      client_id: client_id,
      scope: scope,
      redirect_uri: redirect_uri,
    });
  res.redirect(authUrl);
});

app.get("/callback", async (req, res) => {
  const code = req.query.code || null;

  if (!code) {
    return res
      .status(400)
      .send("Error: No se recibió el código de autorización.");
  }

  try {
    const response = await axios.post(
      "https://accounts.spotify.com/api/token",
      querystring.stringify({
        grant_type: "authorization_code",
        code: code,
        redirect_uri: redirect_uri,
        client_id: client_id,
        client_secret: client_secret,
      }),
      {
        headers: { "Content-Type": "application/x-www-form-urlencoded" },
      }
```

```
    );

    const { access_token } = response.data;
    const userProfile = await axios.get("https://api.spotify.com/v1/me",
{
      headers: { Authorization: `Bearer ${access_token}` },
    });

    res.json({
      message: "Autenticación exitosa",
      user: userProfile.data,
    });
  } catch (error) {
    console.error(
      "Error obteniendo el token:",
      error.response?.data || error.message
    );
    res.status(500).send("Error al obtener el token de acceso.");
  }
});

app.listen(PORT, () => {
  console.log(`Servidor corriendo en http://localhost:${PORT}`);
});
```

```
Servidor corriendo en http://localhost:12345
PS C:\Users\heily\Desktop\PROYECTOS\Dam-Privado\DamPrivado\oauth2-express> node app.js
Servidor corriendo en http://localhost:12345
```

- Capturas de pantalla del funcionamiento de OAuth 2.0:

Si introducimos el link nos lleva:



Y si damos click a login nos redirige a :

https://accounts.spotify.com/esES/login?continue=https%3A%2F%2Faccounts.spotify.com%2Fauthorize%3Fscope%3Duser-read-private%2Buser-read-email%26response_type%3Dcode%26redirect_uri%3Dhttp%253A%252F%252Flocalhost%253A12345%252Fcallback%26client_id%3D4fb8457bacc1470b8cf950d3ecd7a7dc