Cinthya Daniela Lugo Novoa A01332942
Alejandra Cuéllar González A01333324

Third-party APIs: Facebook, Spotify

Facebook API

El API permite a las aplicaciones utilizar las conexiones de la red social e información de perfil para para poder hacer que las aplicaciones se involucren más, todo esto sujeto a las configuraciones de privacidad de los usuarios. El API utiliza un protocolo RESTful y los datos de respuesta están en formato JSON.

En este caso se hará uso de Graph API versión 2.10, la cual es la principal forma de introducir datos y extraer datos; es una API basada en HTTP, por lo tanto puede ser compatible con cualquier lenguaje que tenga una biblioteca de este tipo. La información disponible, gracias a Graph API, está comprendida en tres conceptos principales: (1) Nodos, que representan los objetos, (2) Perímetros, conexiones entre los objetos y (3) Campos, información sobre los objetos.

Facebook te permite crear una aplicación en la que te dará la información personalizada, con el id de la aplicación, para implementarla, por ejemplo al usar Facebook SDK for Javascript al hacer un login.

La mayoría de las solicitudes son pasadas a la API en graph.facebook.com, realizando solicitudes con HTTP GET, HTTP POST y eliminar elementos con HTTP DELETE. Por ejemplo.

GET graph.facebook.com POST graph.facebook.com
/{node-id}/{edge-name} /{node-id}/{edge-name}

Los usuarios deberán iniciar sesión en Facebook para poder acceder tener acceso temporal y seguro a las APIs de Facebook, y hacer uso de ciertas propiedades. Al iniciar sesión se obtendrá un identificador de acceso, que identifica a un usuario, aplicación o página, con el cual se hacen llamadas a Graph API. El Identificador de Usuario es el tipo de identificador más usado, el cual también será utilizado en nuestra aplicación web y será obtenido mediante un cuadro de diálogo de inicio de sesión y requieren que la persona conceda permiso a la aplicación, ya que se cuenta con el identificador se puede utilizar para realizar llamadas. En web, utilizando JavaScript, se vería como lo siguiente:

```
FB.getLoginStatus(function (response)){
if(response.status == ‘connected’){
var accessToken = response.authResponse.accessToken;
}
} );
```

Para agregar el SDK se utiliza un fragmento de código especificado en la siguiente liga: https://developers.facebook.com/docs/javascript/quickstart , este se debe insertar después de la etiqueta <body> en el HTML. Posteriormente se debe comprobar el estado de inicio de sesión:

```
FB.getLoginStatus(function (reponse)){
```

statusChangeCallback(response);
});
Posteriormente, invitar a las personas a hacer login por medio de un botón. En el apartado scope se especifican los permisos que quieren ser solicitados al usuario.
<fb: login-button scope="public_profile, email" onlogin="checkLoginState();">
</fb:login-button>
Para leer la lista de amigos del usuario se necesita del permiso user_friends, el cual viene por default mostrando los amigos que también utilizan la aplicación en cuestión, en el caso de Facebook SDK for Javascript cualquier otro permiso debe ser especificado en el apartado scope. Teniendo el permiso se utiliza: /{user-id}/friends, el resultado es arrojado en formato JSON, con el campo nombre y id.
En el caso de los eventos, en la versión 2.10 de Graph API se permite leer los eventos del usuario por medio de: /{even-id}, para esto se necesita el permiso user_events para ver los eventos visibles para el usuario.

Spotify API
Spotify Web API permite a las aplicaciones extraer datos del catálogo de música de Spotify y administrar las playlists y música guardada de los usuarios, si este así lo permite; el Web API endpoint regresa metadatos en formato JSON. Está basada en principios REST, lo que quiere decir que los recursos de datos son accesados en formato de solicitudes HTTPS en formato UTF-8; en algunas acciones utiliza los términos usados en HTTP como GET, POST, PUT y DELETE.
Cabe mencionar que para hacer uso de Spotify Web API se tiene que tener una cuenta de usuario en Spotify.
Todas las solicitudes que se hagan al API requieren autorización y el encabezado de la solicitud enviada por la aplicación debe incluir un token de acceso válido. Para tener acceso a la información de los usuarios primero se tiene que registrar la aplicación en My Applications, para que nos genere un Client ID y Secret Key, las cuales se usarán para hacer solicitudes desde la aplicación.
Spotify Web API soporta 3 flows de autorización:
● Authorization Code: obtiene un código para luego cambiarlo por un access token y un refresh token.
● Client Credentials: obtiene un access token proporcionando tu Client ID y Secret Key, es usado en autenticación server-to-server. Solo pueden ser accesados los endpoints que no accedan a información del usuario.
● Implicit Grant: se da desde el lado del cliente y no involucra secret keys.
En este caso utilizaremos Authorization Code flow por las ventajas como extender la validez del access token.
Para pedirle al usuario autorización se utiliza el endpoint:

GET
https://accounts.spotify.com/authorize/?cliend_id={client_id}&response_type=code&r

edirect_uri={url}&scope={permisos}
El parámetro scope es opcional, si no se especifican permisos, la autorización será solo para la información pública.
Una vez que el usuario aceptó es redirigido a la URL especificada en el parámetro redirect_uri junto con el código que puede ser intercambiado por un access token.
Para obtener el access token se debe hacer un solicitud POST:
POST
https://accounts.spotify.com/api/token?grant_type=”authorization_code”&code={cod
e}
Una vez el intercambio este hecho y tengamos el access token, se usa para acceder a Spotify Web API: “-H Authorization: Bearer {access_token}”
https://api.spotify.com/v1/me

Para acceder a las canciones de una playlist de un usuario de Spotify se utiliza el siguiente endpoint:
GET “https://api.spotify.com/v1/users/{user-id}/playlists/{playlist_id}/tracks” -H
Authorization: Bearer {access token}
Si es satisfactorio se tendrá como resultado un arreglo con las canciones en formato JSON y un status code de 200.
Por otro lado, para obtener una playlist se utiliza el siguiente endpoint:
GET “https://api.spotify.com/v1/users/{user-id}/playlists/{playlist_id}” -H
Authorization: Bearer {access token}
Para poder buscar un artista, álbum, canción o playlist que corresponda a cierta palabra clave, se utiliza:
GET “ https://api.spotify/v1/search?q=musica20%disco& type=album” -H
Authorization: Bearer {access token}
Para borrar canciones de una playlist se usa lo siguiente:
DELETE https://api.spotify.com/v1/users/{user_id}/playlists/{playlist_id}/tracks

Research API REST
REST es un enfoque de desarrollo de proyectos y servicios web, definido por Roy Fielding en su disertación “Architectural Styles and the Design of Network-based Software Architecture”. REST, Representional State Transfer (Transferencia de Estado Representacional) es el líder en el desarrollo de servicios de aplicaciones, ya que es el estándar más lógico, eficiente y habitual en la creación de APIs para servicios de internet.

Con REST se refiere a cualquier interfaz entre sistemas que use HTTP para obtener datos en muchos formatos posibles.
Algunas de las características destacables de REST son: protocolo cliente/servidor sin estado, lo que quiere decir que cada petición HTTP que se hace contiene la información necesaria para ejecutarla, para que el cliente o servidor no necesiten recordar estados previos. Los sistemas REST tienen cinco operaciones importantes: POST, GET, PUT,

DELETE y PATCH, además de HEAD y OPTIONS. Los objetos se manipulan a partir de la URI, identificador único de cada recurso. REST cuenta con una arquitectura jerárquica entre los componentes. Las ventajas de REST son: separación entre cliente y servidor, visibilidad, fiabilidad y escalabilidad y la API REST es independiente del tipo de plataformas y/o lenguajes.
POST http://www.example.com/customers/12345/orders
GET http://www.example.com/customers/12345/orders
PUT http://www.example.com/customers/12345
En un API REST es necesario contar con el principio Hypermedia As The Engine Of Application State, el cual permite definir que cada vez que se hace una solicitud al servidor, este regrese una respuesta con los hipervínculos de navegación asociados a otros recursos del cliente.
Las buenas prácticas que se mencionan en una API REST, es usar identificadores en los URLs en lugar de query-string cuando se traten nombres de recursos; además de usar plural en los segmentos de URIs, por ejemplo: /customers/33279 en lugar de /customer/33279.

How will you define your API

Cliente
/home
Descripción: Log in
GET /reservaciones/{id}
Descripción: Mostrar todas las reservaciones del usuario
GET /alimentos
Descripción: Mostrar el menú
GET /canciones
Descripción: Mostrar las canciones en la lista
POST /canciones
Descripción: Agregar canción a la lista de reproducción

Admin
/login
Descripción:Log in
GET /reservaciones
Descripción: Mostrar todas las reservaciones
GET /reservaciones/{date}
Descripción: Mostrar todas las reservaciones de una fecha
DELETE /reservaciones/{id}
Descripción: Eliminar una reservación por id
GET /cuentas/{time-frame}
Descripción: Mostrar las cuentas de un lapso de tiempo
GET /estadisticas
Descripción: Mostrar la información histórica
GET /canciones
Descripción: Mostrar las canciones en la lista
GET /alimentos
Descripción: Mostrar el menú
POST /alimentos
Descripción: Agregar item al menú

Document to define data transformation
Los endpoint que utilizaremos son los siguientes: {user_id} para Facebook, {event_id} para Facebook, {user_id} para Spotify, {playlist_id} para Spotify, {track_id} para Spotify. {user_id} para Facebook se utilizará para que el usuario inicie sesión y de autorización para acceder a su información como sus amigos y eventos. Con {user_id} identificaremos las acciones que realice el cliente dentro de la aplicación, como ordenar comida.

{even_id} para Facebook se utilizará para mostrar los eventos que el cliente tiene agendados en su cuenta de Facebook, así como para agregar invitados a la lista de acuerdo a la elección del cliente.
{user_id} para Spotify se utilizará para que el cliente pueda acceder a su cuenta y a sus playlist y canciones guardadas. Con el se mantendrá el registro de las canciones seleccionadas por cliente y turnos para cantar.
{playlist_id} para Spotify se utilizará para mostrar la playlist seleccionada por el usuario.
{track_id} para Spotify se utilizará para mostrar las canciones de una playlist, así como para determinar la canción que corresponde a cada usuario, reproducir y eliminar, en caso de desearlo.

Referencias

https://developer.spotify.com/web-api/search-item/
https://developer.spotify.com/web-api/playlist-endpoints/
https://developer.spotify.com/web-api/authorization-guide/
https://developers.facebook.com/docs/graph-api/using-graph-api/
https://developers.facebook.com/docs/graph-api/overview
https://developers.facebook.com/docs/graph-api/reference/user/friends/
https://developers.facebook.com/docs/facebook-login/access-tokens/
https://developers.facebook.com/docs/facebook-login/
https://developers.facebook.com/docs/graph-api/reference/v2.10/event
https://www.youtube.com/watch?time_continue=774&v=pVAMOielOJQ