1.Refer FB + LinkedIn API's ::

🡺Which API referred

FB🡺Graph API

Marketing API

Messenger API

Login Api

Linkedin🡺Marketing Developer Platform

OAuth 2.0 API

Share API

API Endpoint 🡺This API use to specifies the location where you can send your requests.

APIs support different request methods such as GET, POST, PUT, DELETE.

🡺API’s

I understand using APIs developers may include features into their applications, including the ability to retrieve user data, submit content, control advertising and more. Each API has a unique function and offers multiple states endpoints and functions to communicate with the relevant platforms.

🡺Parameter endpoints used Linkedin & FB

Facebook Graph API include access token for authentication, fields to specify the data fields to retrieve, limit to restrict the amount of results, since and till for time-based filtering, among others. Depending on the endpoint and the data you wish to retrieve or interact with, different parameters will be used.

Facebook Marketing API: Access token for authentication, campaign id or adset id to define the campaign or ad set, fields to describe the data fields to receive, time\_range for filtering data within a certain time range, and more are all parameters for the Marketing API. Depending on the exact endpoint and the action you wish to take, the parameters change.

Parameters for the LinkedIn Marketing Developer Platform : The LinkedIn Marketing API has a number of parameters, such as the oauth2\_access\_token for authentication, q for search queries, start and count for pagination, fields to indicate the data fields to obtain, and more. The settings change depending on the endpoint in question and the data you want to access or work with.

Share API for LinkedIn:Oauth2\_access\_token is used for authentication. Other parameters for the sharing API include remark, which specifies the text content of the sharing, content, which provides the URL or other information about the shared resource, and more. Specific to the share endpoint and the content you want to share are the parameters.

4.DB Schema --> Structure of the tables :

A database schema explains the organization and structure of a database like a design or plan. It acts as a manual on how information is kept, arranged, and connected within the database. Consider it a road plan that details the database's tables, fields, data types, constraints, and connections between various components.

The tables' structure, which can be viewed as data storage containers, is determined by the schema. The various sorts of information that can be kept are defined by the columns (also known as fields) that make up each table.

An important component of a database system is the database schema. It provides a framework for consistently and effectively storing and accessing information by defining the structure, organization, and relationships of the data.

6. Queue endpoint + topic endpoint

**Queue endpoints** are locations where messages are temporarily kept in a messaging system until they are consumed by the intended recipient or consumer application. A message is added to a queue when it is sent to a queue endpoint, and it stays there until it is retrieved by a consumer. Following processing, the message can be taken out of the queue by the consumer.

**Topic endpoint** is the destination or address where messages are sent and received in a Publish-subscribe messaging pattern,. Topics are used to enable a single producer or publisher to send messages to multiple consumers. When a message is sent to a topic endpoint, it is distributed to all the subscribed consumers who have expressed interest in receiving messages on that topic.

The main difference lies in how the messages are handled and consumed. Queue endpoints store messages until they are consumed by a specific recipient, while topic endpoints distribute messages to multiple consumers interested in a particular topic.

7.What is High Availability Architecture? what happens in High Availability Architecture

A system's architectural strategy known as high availability architecture tries to achieve maximum uptime and minimal service interruptions. It entails the use of redundant methods and components to get rid of single points of failure and maintain service availability.

Multiple instances of crucial components, like servers, databases, or network devices, are deployed simultaneously in a high availability architecture.

8.NodeJS calculator

// Importing required modules

const express = require('express');

const xml2js = require('xml2js');

const fs = require('fs');

// Creating an Express app

const app = express();

const port = 3000;

// Calculator endpoints

app.get('/add/:num1/:num2', (req, res) => {

const num1 = parseInt(req.params.num1);

const num2 = parseInt(req.params.num2);

const result = num1 + num2;

res.send(`Addition: ${num1} + ${num2} = ${result}`);

});

app.get('/subtract/:num1/:num2', (req, res) => {

const num1 = parseInt(req.params.num1);

const num2 = parseInt(req.params.num2);

const result = num1 - num2;

res.send(`Subtraction: ${num1} - ${num2} = ${result}`);

});

app.get('/divide/:num1/:num2', (req, res) => {

const num1 = parseInt(req.params.num1);

const num2 = parseInt(req.params.num2);

if (num2 === 0) {

res.send('Error: Division by zero is not allowed');

} else {

const result = num1 / num2;

res.send(`Division: ${num1} / ${num2} = ${result}`);

}

});

// XML parsing endpoint

app.get('/parse-xml', (req, res) => {

// Read the sample XML file

const xmlData = fs.readFileSync('sample.xml', 'utf-8');

// Parse the XML data

xml2js.parseString(xmlData, (err, result) => {

if (err) {

res.send('Error parsing XML');

} else {

res.json(result);

}

});

});

// Start the server

app.listen(port, () => {

console.log(`Server is running on http://localhost:${port}`);

});

