Rakamin Academy

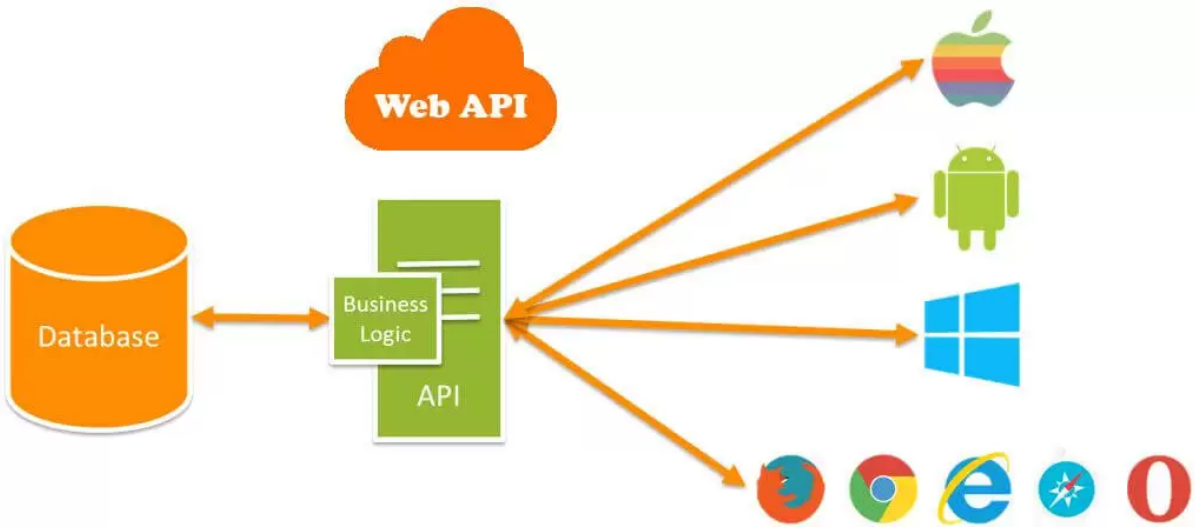
Reading Material

Melakukan API Testing - Pengenalan API

API

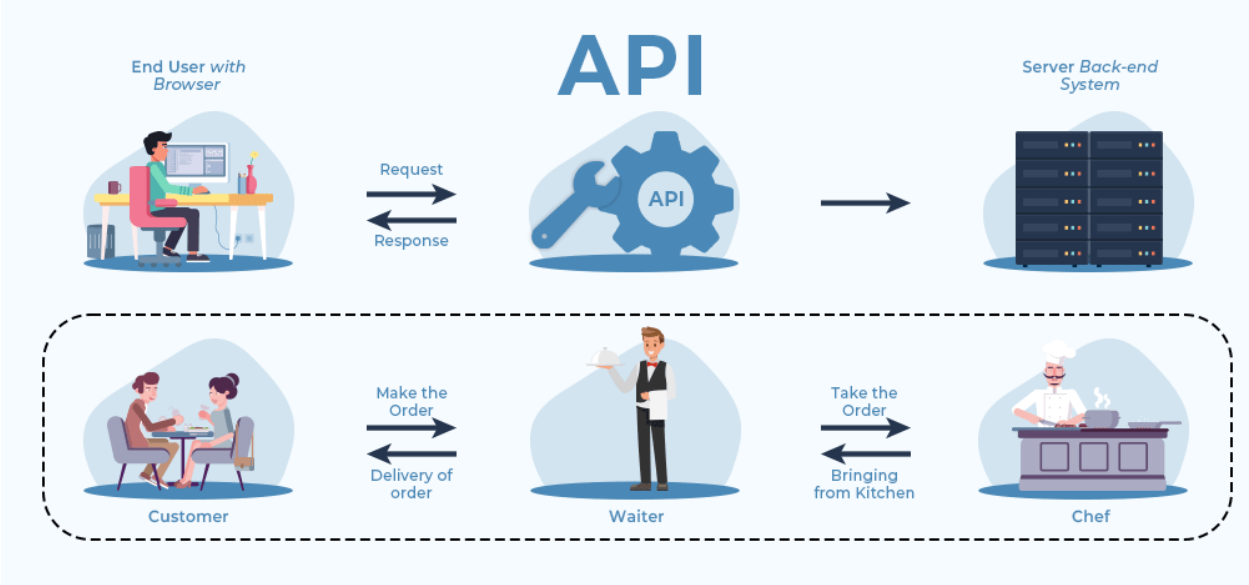
API adalah singkatan dari *Application Programming Interface* yang berarti antarmuka pemrograman aplikasi.

Sebuah API adalah sekumpulan definisi dan protokol untuk membangun dan mengintegrasikan perangkat lunak aplikasi. API memungkinkan produk atau layanan dapat berkomunikasi dengan produk dan layanan lain tanpa harus tahu bagaimana penerapannya.



Gambar diatas menunjukkan gambaran tugas dari API dalam pengembangan aplikasi. API juga bisa digunakan untuk komunikasi dengan berbagai bahasa pemrograman yang berbeda. Hal ini tentu sangat memudahkan bagi developer karena tidak perlu menyediakan semua data sendiri, cukup mengambil data yang dibutuhkan dari platform lain melalui API.

Contoh implementasi API dalam kehidupan sehari hari



Perumpamaan yang bisa digunakan untuk menjelaskan API adalah **seorang pelayan di restoran**. Tugas pelayan tersebut adalah menghubungkan tamu restoran dengan juru masak.

Jadi, tamu cukup memesan makanan sesuai daftar menu yang ada dan pelayan memberitahukannya ke juru masak. Nantinya, pelayan akan kembali ke tamu tadi dengan masakan yang sudah siap sesuai pesanan.

Dalam kasus ini, tamu tidak perlu mengetahui siapa juru masak yang membuat pesanannya, bahan bahan apa saja yang diperlukan untuk membuat hidangannya. Begitupun sebaliknya, juru masak tidak perlu mengetahui siapa tamu yang memesan hidangan. Jadi mereka terhubung melalui pelayan restoran saja.

Kegunaan dan Manfaat API

API secara khusus sangat membantu bagi para developer. Sebab, API bisa meningkatkan efisiensi waktu, fleksibilitas, dan menghemat biaya. Selain itu, ada juga sejumlah kegunaan lainnya. Berikut adalah beberapa manfaat dari penggunaan API:

1. Mempermudah Pembuatan Aplikasi Fungsional

Penggunaan API sangat diperlukan untuk membuat aplikasi yang fungsional dan kompleks. Sebab, aplikasi yang dikembangkan berdasarkan API akan otomatis memiliki fitur yang diberikan oleh aplikasi tujuan, tanpa perlu menambahkan datanya secara manual.

Contoh: Pada aplikasi transportasi ojek online yang memberikan banyak layanan, pasti akan sangat membutuhkan fungsionalitas maps. Nah, di sini, developer aplikasi ojek online tidak perlu membuat aplikasi maps sendiri, cukup mengambil data API dari Google Maps.

2. Efisiensi Pengembangan Aplikasi

Baik developer maupun pengguna aplikasi pasti ingin bisa menggunakan berbagai platform dengan mudah. Kemudahan ini bisa dicapai dengan komunikasi yang baik antar aplikasi. Dengan menggunakan API, developer tidak perlu repot membangun komunikasi dengan aplikasi yang ingin dihubungkan.

Kemudian, API juga berguna jika developer ingin membangun aplikasi lintas platform dengan berbagai layanan sekaligus, seperti aplikasi/website pemesanan tiket online. Mereka cukup melakukan integrasi dengan masing-masing layanan, misalnya maskapai penerbangan. Jadi, developer tidak perlu komunikasi manual untuk menanyakan harga dan ketersediaan seat di pesawat.

3. Meringankan Beban Server

Semua data yang dibutuhkan di server tidak perlu disimpan secara keseluruhan karena sudah ada API. Cukup minta API untuk memperoleh data terbaru dari server aplikasi sumber. Kemudahan ini tentu akan sangat membantu karena server tidak akan terbebani dan meminimalkan downtime website.

Dampak positifnya juga tentu akan signifikan, karena server yang down bisa menurunkan performa website dan membuat pengunjung website menjadi kurang nyaman. Jadi, ringannya beban server ini juga akan menguntungkan bagi pemilik website, juga pengunjung website yang akan merasa lebih nyaman saat mengakses website.

Jenis-Jenis API

API dikelompokkan berdasarkan hak aksesnya. Seperti yang telah dijelaskan bahwa API adalah antarmuka yang berperan sebagai penghubung. Dalam hal ini API diberi hak khusus yang membatasi sejauh mana hubungan tersebut bisa dibuat. Berikut adalah jenis-jenis API:

1. Public API

Sesuai namanya, Public API adalah jenis API yang boleh digunakan oleh siapa saja di berbagai platform. Public API juga sering disebut sebagai Open API, dan merupakan yang paling sering digunakan.

Contoh Public API
https://developers.google.com/maps/documentation/javascript/get-api-key
https://newsapi.org/s/indonesia-news-api
https://github.com/dyazincahya/quran-api-with-php-codeigniter
https://rajaongkir.com/dokumentasi

2. Private API

Berkebalikan dengan Public API yang bisa digunakan oleh siapa saja, Private API tidak boleh digunakan secara umum. Jenis API ini biasanya digunakan untuk keperluan pribadi atau internal dalam pengembangan aplikasi tertentu.

Sebagai contoh yaitu di dalam sebuah organisasi, API dari backend yang digunakan untuk mengakses frontend suatu website.

3. Partner API

Partner API boleh digunakan secara umum, tapi hanya bagi pihak yang bekerja sama dan memiliki izin untuk menggunakannya.

Mirip seperti Public API, proses pendaftaran kepada penyedia API harus dilakukan lebih dulu. Kemudian, penggunaanya juga hanya diperbolehkan untuk aplikasi tertentu sesuai perjanjian.

4. Composite API

Composite API adalah jenis API yang menyimpan data dari banyak server atau hosting di satu tempat.

Tentu saja, jenis ini bisa membantu menghemat waktu dan sangat menguntungkan user karena benar-benar mempersingkat pekerjaan, yaitu cukup dengan satu kali akses untuk memperoleh sejumlah data dari sumber yang berbeda.

Arsitektur API

API bekerja dengan cara bertukar perintah dan data, dalam hal ini memerlukan protokol dan arsitektur yang jelas seperti aturan, struktur, dan batasan yang mengatur operasi API. Arsitektur API berkaitan pada bentuk data yang dikirim. Ada tiga arsitektur API yang sering digunakan oleh developer dalam pembangunan aplikasi. Arsitektur berkaitan pada bentuk data yang dikirim.

1. RPC

RPC (*Remote Procedure Call*) merupakan teknologi untuk membuat komunikasi antara client side dan server side bisa dilakukan dengan konsep sederhana.

RPC memiliki dua jenis, yaitu XML-RPC dan JSON-RPC. Sesuai namanya, XML-RPC menggunakan format XML sebagai media perpindahan data, sedangkan JSON-RPC menggunakan JSON untuk perpindahan data.

2. SOAP

SOAP (Simple Object Access Protocol). Arsitektur ini menggunakan XML (Extensible Markup Language) yang memungkinkan semua data disimpan dalam dokumen.

Dalam penerapannya, SOAP secara ketat mendefinisikan bagaimana pesan harus dikirim dan apa yang harus disertakan di dalamnya. Sehingga membuat arsitektur SOAP menjadi lebih aman.

3. REST

REST (Representational State Transfer) merupakan arsitektur API yang sangat populer karena kemudahan penggunaannya. REST menggunakan JSON sebagai bentuk datanya sehingga lebih ringan sehingga performa aplikasi pun menjadi lebih baik.

Developer API dapat merancang API menggunakan beberapa arsitektur yang berbeda. API yang mengikuti gaya arsitektur REST disebut sebagai API REST. Layanan web yang menerapkan arsitektur REST disebut sebagai layanan web RESTful. Istilah API RESTful umumnya merujuk pada API web RESTful.

API RESTful

Pengidentifikasi sumber daya

Pada layanan REST, server biasanya melakukan identifikasi sumber daya dengan menggunakan Uniform Resource Locator (URL). URL menentukan jalur ke sumber daya. URL mirip dengan alamat situs web yang digunakan untuk mengunjungi halaman web mana pun. URL juga disebut titik akhir (endpoint) permintaan dan dengan jelas menentukan server yang dibutuhkan klien.

Metode

Developer sering mengimplementasikan API RESTful dengan menggunakan Hypertext Transfer Protocol (HTTP). Metode HTTP memberi tahu server hal-hal yang perlu dilakukan terhadap sumber daya. Berikut ini adalah empat metode HTTP yang paling umum digunakan:

Metode	**Deskripsi**
GET	Mengambil informasi tentang *REST API resource*
POST	Membuat sebuah *REST API resource*
PUT	Mengupdate *REST API resource*
DELETE	Menghapus *REST API resource* atau Komponen terkait

Header HTTP

Header permintaan adalah pertukaran metadata antara klien dan server. Misalnya, header permintaan menunjukkan format permintaan dan respons, memberikan informasi tentang status permintaan, dan sebagainya.

Data

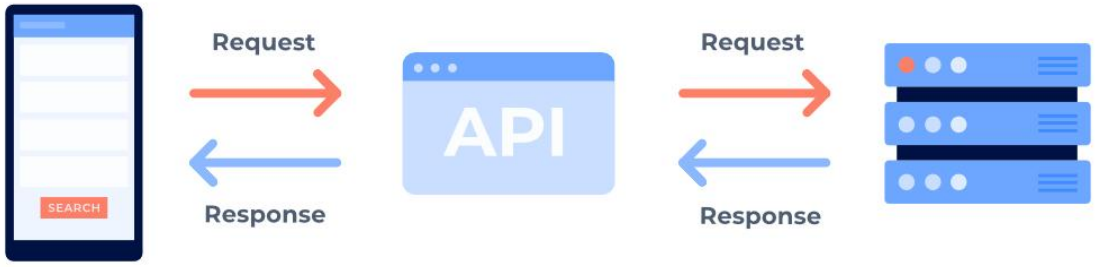
Permintaan API REST mungkin menyertakan data untuk metode POST, PUT, dan metode HTTP lainnya agar permintaan berhasil.

Parameter

Permintaan API RESTful dapat termasuk parameter yang memberi server lebih banyak detail tentang hal yang perlu dilakukan.

Cara Kerja API

Setelah mengetahui pengertian, jenis dan arsitektur API, kita akan mempelajari cara kerja dari API itu sendiri. Coba perhatikan ilustrasi cara kerja API sebagai berikut:



1. Aplikasi Mengakses API

Bagian pertama dari cara kerja API adalah ketika pengguna mengakses sebuah aplikasi.

Sebagai contoh pada pemesanan tiket pesawat melalui salah satu aplikasi online booking. Ketika ingin memesan tiket pesawat untuk tujuan tertentu, Aplikasi akan mengakses API maskapai penerbangan yang sudah dihubungkan.

2. API Melakukan Request ke Server

Setelah aplikasi berhasil mengakses alamat API, permintaan tersebut akan diteruskan ke server maskapai penerbangan. Jadi, API akan memberitahukan bahwa Aplikasi membutuhkan data penerbangan untuk tanggal dan tujuan yang telah disebutkan.

3. Server Memberi Respon ke API

Ketika menemukan data yang sesuai permintaan, server kembali menghubungi API. Data tersebut berupa informasi seperti ketersediaan tempat duduk, jam keberangkatan dan lainnya.

4. API Menyampaikan Respon ke Aplikasi

Selanjutnya, API meneruskan informasi dari server ke aplikasi. Dalam contoh ini, Aplikasi akan mendapatkan informasi yang didapatkan dari maskapai penerbangan yang dihubungi.

Proses ini berlangsung bersama dengan permintaan ke maskapai penerbangan lain. Oleh karena itu, dalam satu pencarian bisa menampilkan jadwal penerbangan dari berbagai maskapai sekaligus.

JSON Introduction

JSON (JavaScript Object Notation) adalah format file berbasis teks yang umumnya digunakan dalam proses pertukaran data antara server dan klien. File JSON memiliki ekstensi **.json** serta menggunakan teks yang sama-sama bisa dibaca oleh manusia dan dipahami oleh komputer.

JSON merupakan format yang banyak disukai karena mudah dipahami, ringan, ringkas, dan menunjukkan data terstruktur berdasarkan syntax objek JavaScript. Karena inilah program JavaScript bisa mengubah data JSON menjadi objek native JavaScript tanpa harus melakukan parsing.

Sintaks JSON

Contoh sederhana penggunaan sintaks JSON

{“kota”:”Jakarta Pusat”, “kode_pos”: 10110, “negara”:”Indonesia”}

JSON selalu dibuka dan ditutup dengan tanda {} atau kurung kurawal. Sintaksnya terdiri dari dua elemen, yaitu **key** dan **value.** Keduanya dipisahkan oleh tanda titik dua.

Contoh **key** adalah “kota”, sedangkan “Jakarta” adalah contoh **value**. Keduanya dibuka dan ditutup dengan tanda kutip dua.