Application Programming Interfaces

Zeham Management Technologies BootCamp
by SDAIA

October 2024

SDAIA
الهيئة السعودية للبيانات
والذكاء الاصطناعي
Saudi Data & AI Authority

Agenda

Introduction to APIs

APIs Communication

API Authentication and Security

Building and Consuming APIs

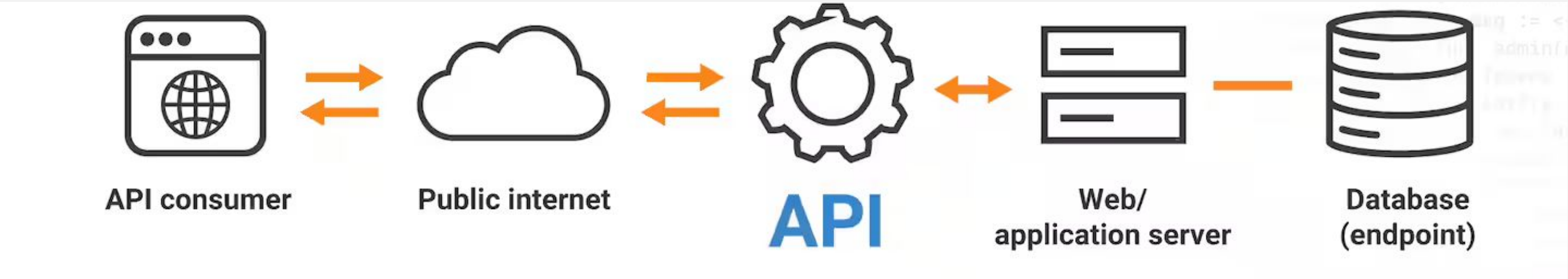
Practical Use Cases of APIs

Introduction to Application Programming Interface

Introduction to APIs

Definition and Purpose:

Definition: API stands for Application Programming Interface. In the context of APIs, the word Application refers to any software with a distinct function. Interface can be thought of as a contract of service between two applications. This contract defines how the two communicate with each other using requests and responses.



Introduction to APIs

Key Benefits:

- Reusability
- Ease of integration
- Abstraction of complex logic.

Real-World Examples:

- Social media login integrations (e.g., Facebook, Google login).

Introduction to APIs

Types of APIs:

- Open/Public APIs: Available for external users (e.g., Google Maps API).
- Private APIs: Internal company APIs for internal use.
- Partner APIs: APIs shared between business partners.
- Composite APIs: APIs that combine multiple API calls into one.

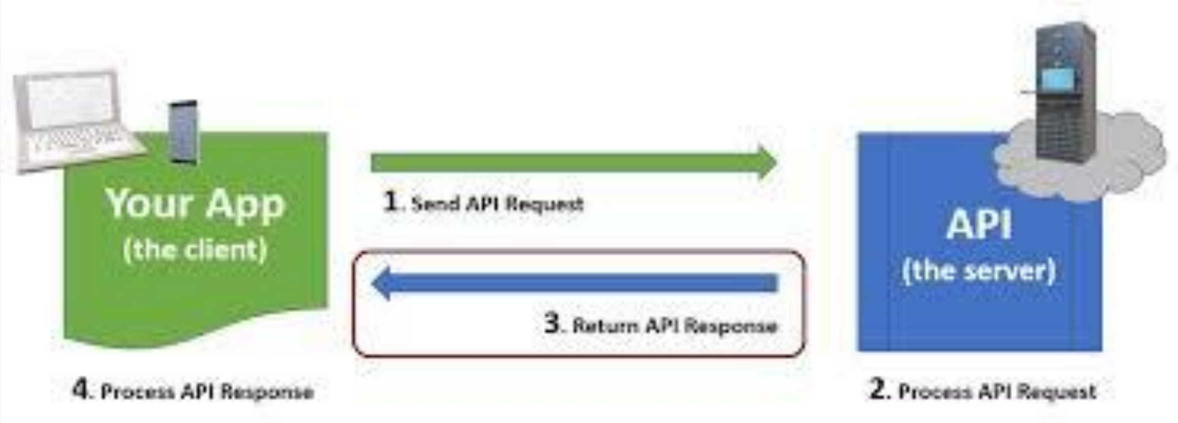
APIs Communication

APIs Communication

How APIs work:

Request/Response Cycle:

- Client sends a request to the server (API provider), and the server returns a response.



APIs Communication

How APIs work:

Data Formats:

- JSON: JavaScript Object Notation, lightweight, widely used in REST APIs.
- XML: Extensible Markup Language, more verbose, used in older SOAP APIs.

APIs Communication

HTTP Methods:

- GET: Retrieve data (e.g., get user profile).
- POST: Create new data (e.g., submit a form).
- PUT: Update existing data.
- DELETE: Remove data.

APIs Communication

REST and SOAP:

Types of web service architectures, also known as API protocols. They define the rules and standards for how applications communicate with each other over a network.

REST and SOAP are two types of web service protocols that allow two systems to exchange data over the web.

APIs Communication

REST and SOAP:

They are frameworks used to build APIs

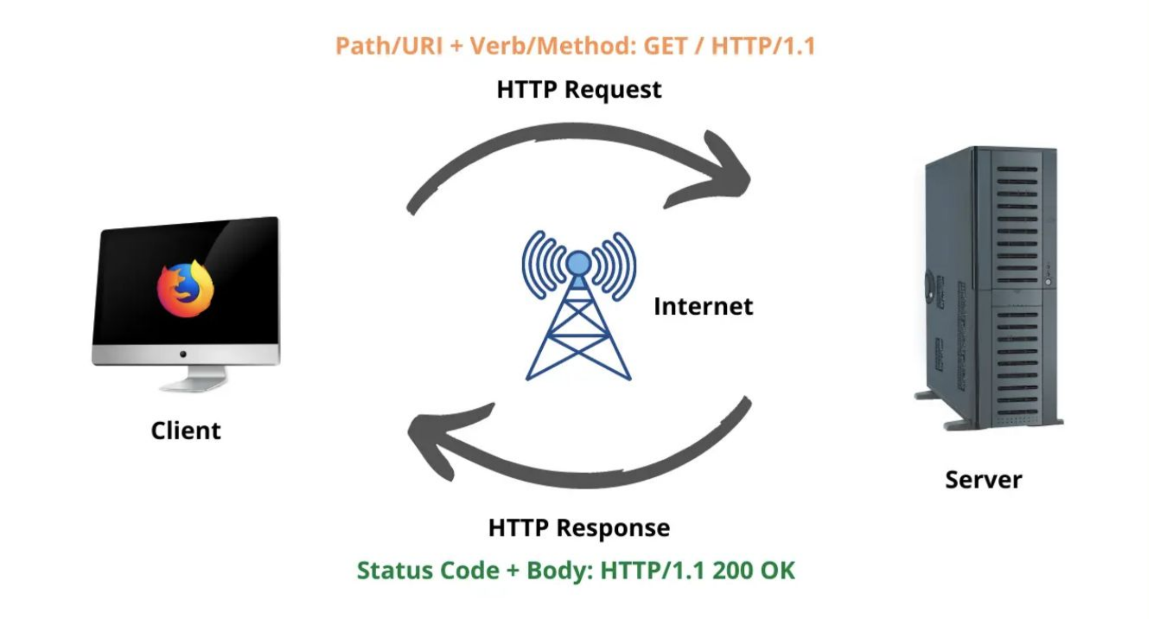
	Type	Main Idea	Data Format	Use Case
REST	It is an architectural style used to create APIs.	RESTful APIs use HTTP requests to perform actions (like getting data or updating data) in a stateless manner.	Typically uses JSON or XML.	Web and mobile apps use REST to communicate with servers (e.g., when you refresh your social media feed, a REST API is usually fetching that data).
SOAP	It is a protocol (a strict set of rules) for sending and receiving messages.	SOAP APIs communicate using XML and enforce strict standards for message structure.	Always uses XML.	SOAP is often used in enterprise applications, like banking or healthcare, where security and reliability are critical.

APIs Communication

Example:

Request: GET /api/users/123

Response: { "id": 123, "name": "John Doe" }


Path/URI + Verb/Method: GET / HTTP/1.1
HTTP Request
Internet
Client
Server
HTTP Response
Status Code + Body: HTTP/1.1 200 OK

API Authentication and Security

API Authentication and Security

Common Authentication Methods:

- API Keys: Simple token-based authentication.
- OAuth: Secure delegated access, widely used in web services (e.g., Google, Facebook).
- JWT (JSON Web Tokens): Tokens that ensure secure API communication.

API Authentication and Security

Example of API Keys:

1. Request: API key included in the query parameters or headers.
2. Server: Validates the API key.
3. Response: Returns the requested resource if valid.

API Authentication and Security

Example of OAuth 2.0:

1. Request (Step 1): Redirect user to authorization server (e.g., Google).
2. Request (Step 2): Exchange authorization code for access token.
3. Request (Step 3): Use access token for subsequent API requests.
4. Server: Validates the token.
5. Response: Returns the requested resource if valid.

API Authentication and Security

Example of JWT:

1. Request (Step 1): User logs in, and server returns a JWT.
2. Request (Step 2): Use JWT in Authorization header for API requests.
3. Server: Decodes and validates the JWT.
4. Response: Returns the requested resource if valid.

Building and Consuming APIs

Building and Consuming APIs

Building a Simple REST API:

Framework: Flask (Python) or FastAPI.

Steps:

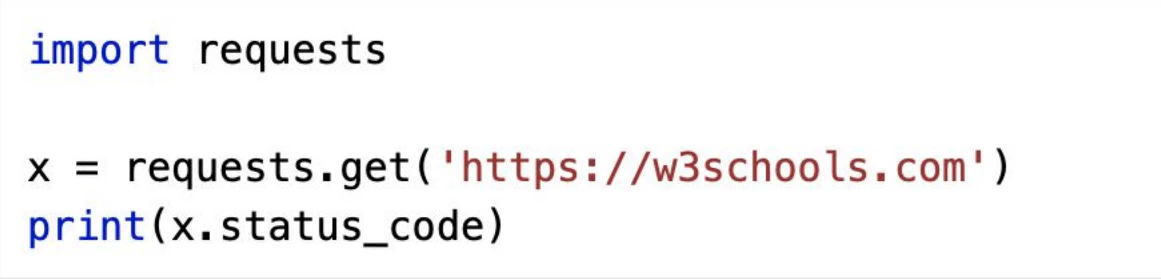
1. Create an API endpoint (e.g., /api/products).
2. Set up CRUD operations (GET, POST, PUT, DELETE).
3. Test your API using tools like Postman or cURL.

Building and Consuming APIs

Consuming Public APIs:

1. Choose an API (e.g., W3Scools API).
2. Write Python code to send a request and handle the response.

Example code:



```
import requests

x = requests.get('https://w3schools.com')
print(x.status_code)
```

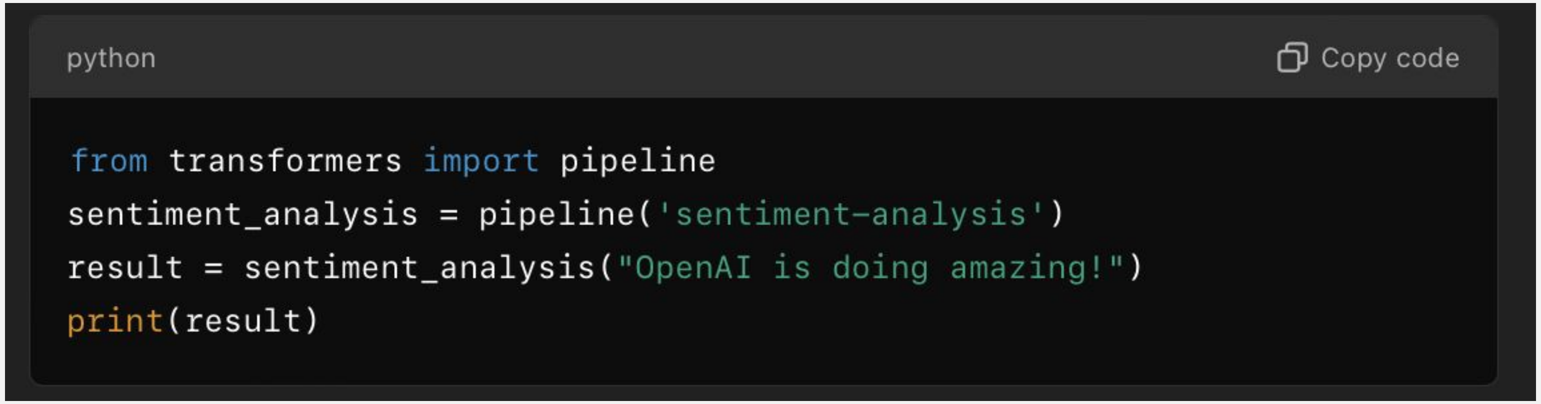
:Practical Use Cases of APIs

Practical Use Cases of APIs:

AI & ML API Integration:

Use Hugging Face APIs to call pre-trained machine learning models (e.g., sentiment analysis).

Example code:



```python
from transformers import pipeline
sentiment_analysis = pipeline('sentiment-analysis')
result = sentiment_analysis("OpenAI is doing amazing!")
print(result)
```

Practical Use Cases of APIs:

Healthcare APIs:

1. Access patient records, lab results via REST APIs.
2. Example: FHIR (Fast Healthcare Interoperability Resources).

E-commerce APIs:

1. Integrating payment gateways (Stripe, PayPal) and fetching product data from e-commerce platforms.

Conclusion

- APIs are essential for enabling the integration of systems, allowing developers to build scalable, modular, and secure applications.
- Understanding APIs is fundamental for building modern web, mobile, and AI-powered services.

Thank you!

SDAIA

الهيئة السعودية للبيانات
والذكاء الاصطناعي
Saudi Data & AI Authority