Building a web server (basics)

Saba Sahban

What is an API?

API (Application Programming Interface) acts as an intermediary that allows different software applications to communicate with each other.

It defines the methods and protocols that developers can use to request and exchange data or functionality between different systems, enabling interaction between various software components over the web.

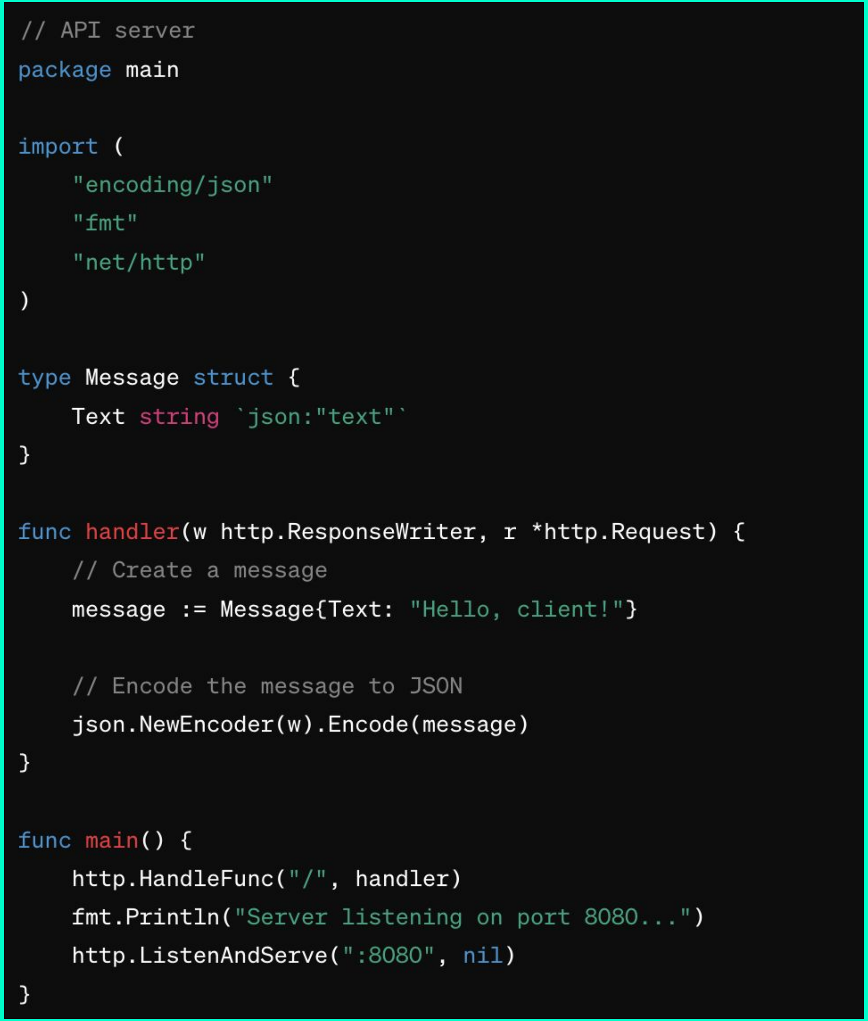
How to Use an API: Just the Basics

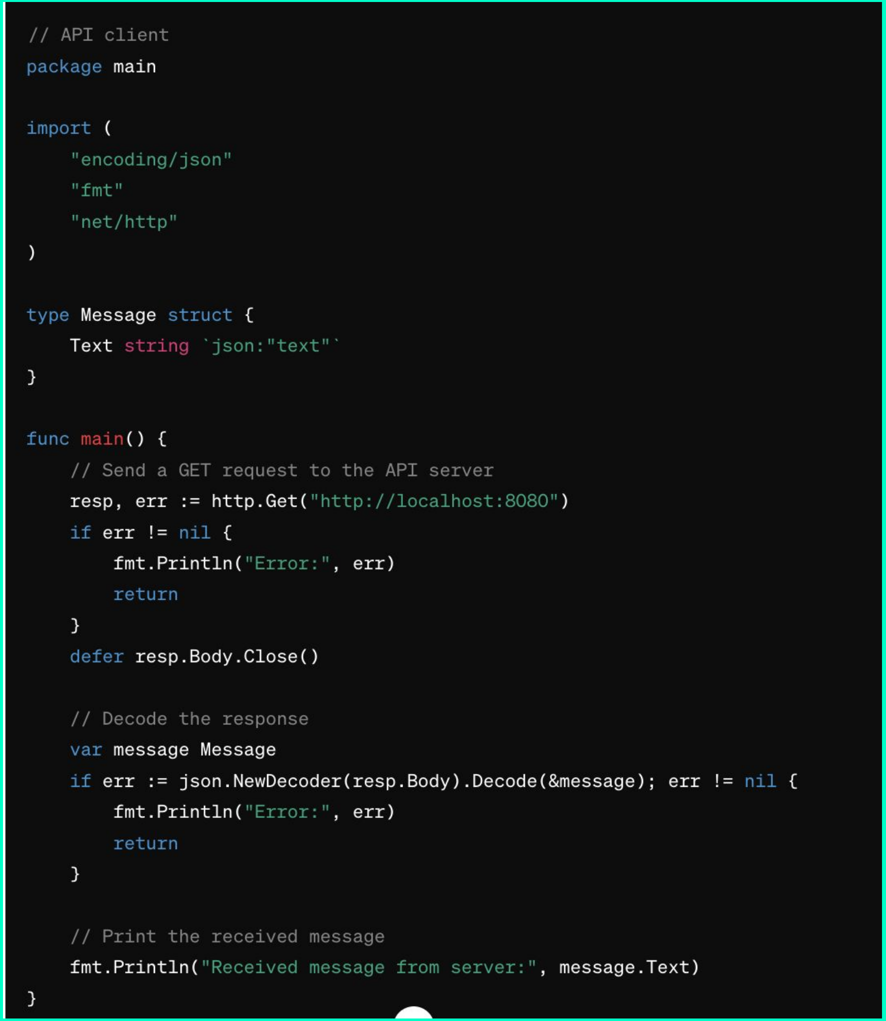
API SERVER AND CLIENT

API client is a software application that consumes or uses the functionality provided by an API. It sends requests to the API server and processes the responses received.

An API server, on the other hand, is a software application that exposes certain functionalities or data through an API. It receives requests from API clients, processes them, and sends back responses.

What’s the difference: Server-side API vs. client-side API | Sendbird


```go
// API server
package main

import (
    "encoding/json"
    "fmt"
    "net/http"
)

type Message struct {
    Text string `json:"text"`
}

func handler(w http.ResponseWriter, r *http.Request) {
    // Create a message
    message := Message{Text: "Hello, client!"}

    // Encode the message to JSON
    json.NewEncoder(w).Encode(message)
}

func main() {
    http.HandleFunc("/", handler)
    fmt.Println("Server listening on port 8080...")
    http.ListenAndServe(":8080", nil)
}
```

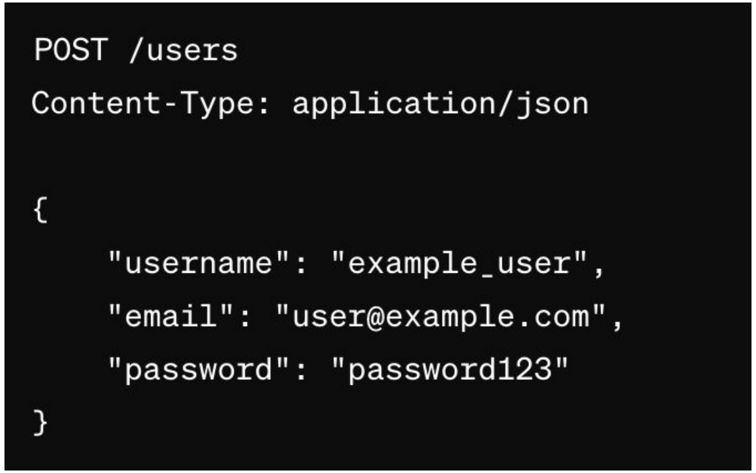

```go
// API client
package main

import (
    "encoding/json"
    "fmt"
    "net/http"
)

type Message struct {
    Text string `json:"text"`
}

func main() {
    // Send a GET request to the API server
    resp, err := http.Get("http://localhost:8080")
    if err != nil {
        fmt.Println("Error:", err)
        return
    }
    defer resp.Body.Close()

    // Decode the response
    var message Message
    if err := json.NewDecoder(resp.Body).Decode(&message); err != nil {
        fmt.Println("Error:", err)
        return
    }

    // Print the received message
    fmt.Println("Received message from server:", message.Text)
}
```

What is CRUD?

CRUD stands for Create, Read, Update, and Delete. These are the four basic operations that can be performed on data.

Create (POST): Adding new data to a system.

Example: Creating a new user profile



```
POST /users
Content-Type: application/json

{
    "username": "example_user",
    "email": "user@example.com",
    "password": "password123"
}
```

What is CRUD? CRUD Operations in APIs | Alex Hyett

What is crud? (contd.)

Read (GET): Retrieving existing data from a system.
Example: Fetching user details

```
GET /users/123
```

Update (PUT/PATCH):
Modifying existing data in a system.
Example:
Updating a user's profile information

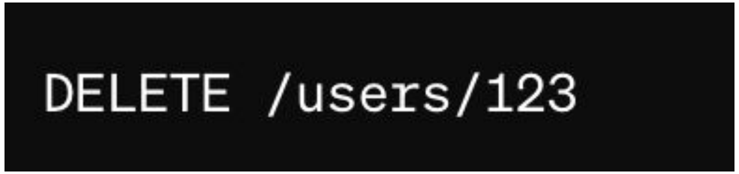
```
PUT /users/123
Content-Type: application/json

{
    "username": "updated_user",
    "email": "updated_email@example.com"
}
```

What is crud? (contd.)

Delete (DELETE):

Removing existing data from a system.

Example: Deleting a user account



```
DELETE /users/123
```

HTTP Methods

An HTTP method is a verb that defines the action the client wants to take on a resource located on a server. It specifies how the server should process the request. Common HTTP methods include GET, POST, PUT, PATCH, and DELETE.

- GET: Retrieve data (Read)
- POST: Create data (Create)
- PUT/PATCH: Update data (Update)
- DELETE: Remove data (Delete)

Learn HTTP basics:

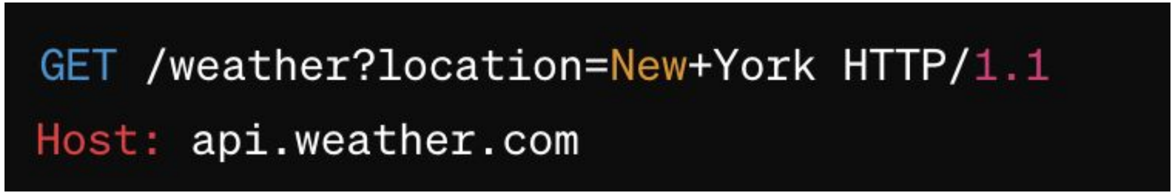
HTTP Fundamentals in 10 Minutes

What is HTTP? Protocol Overview for Beginners

Api request

An API request is a message sent from a client to a server, typically over HTTP or HTTPS, to request certain data or perform a specific action. It contains information such as the request method (e.g., GET, POST), headers, parameters, and sometimes a body with additional data.

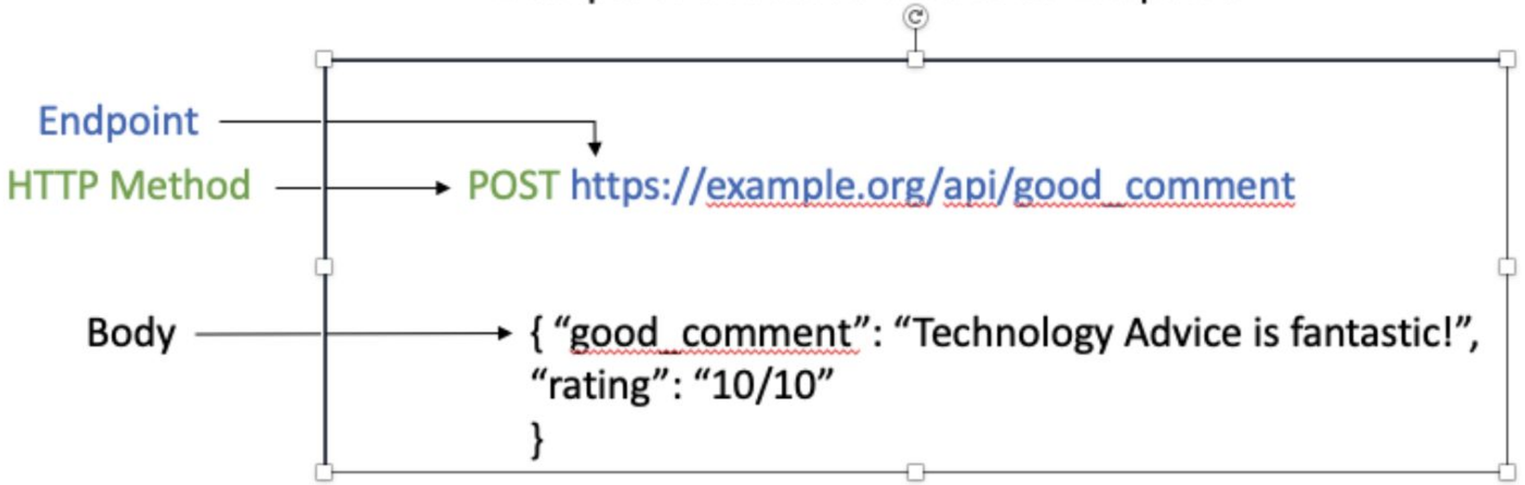
For example, in a weather API, a client might send a request to retrieve the current weather conditions for a specific location. The request could be structured as a GET request with parameters specifying the location, like this:



```
GET /weather?location=New+York HTTP/1.1
Host: api.weather.com
```

In this example, the client is requesting weather data for New York from the server hosting the weather API.

Example of a method used in an Endpoint:



```
POST https://example.org/api/good_comment

{ “good_comment”: “Technology Advice is fantastic!”,
“rating”: “10/10”
}
```

In the example, the good_comment phrase in the Body field will be posted as a new comment in the URL resource.

Building your web server:

20 Tutorials on How to Create Your Own API — [Sorted by Programming Language] | by Alex Walling | The Era of APIs | Medium

Golang(echo, gin, ...):

Creating Golang WebServer With Echo - Part 1: Project Setup and HelloWorld

Building Microservice using Golang Echo framework | by Suman Das | Crux Intelligence | Medium

Golang: Build a simple web server and interact with it | by icelandcheng | Nerd For Tech | Medium

Go Web Frameworks | Gin, Beego, Echo | Setup Simple Server (Part 1)

Build a Rest API with GoLang

Python (FastAPI):

Python FastAPI Tutorial: Build a REST API in 15 Minutes

Starting With Fast API to Create a Simple REST API

Build for the Web with FastAPI

Using FastAPI to Build Python Web APIs

PYTHON(FLASK):

https://m.youtube.com/watch?v=GMppyAPbLYk

https://m.youtube.com/watch?v=xpy-LXHkpk0

https://towardsdatascience.com/creating-restful-apis-using-flask-and-python-655bad51b24?gi=d700bcdfe3de

Node.js:

RESTful APIs in 100 Seconds // Build an API from Scratch with Node.js Express

How to build a REST API with Node js & Express

Creating a REST API with Node.js and Express | Postman Blog

How do I use Node.js to create a REST API? | Reintech media

JAVA(SPRING BOOT)

https://m.youtube.com/watch?v=q_RLfOB7axQ

https://m.youtube.com/watch?v=t608nsKSEh4

https://medium.com/javarevisited/building-restful-apis-in-java-a-step-by-step-tutorial-e1b9b2d3e6ab

Send your first API request with Postman

[Basics of API Testing Using Postman - GeeksforGeeks]

[Send your first API request | Postman Learning Center]