

# Web API Design



**[https://github.com/up1/  
course-web-api](https://github.com/up1/course-web-api)**



# Agenda day 1

- Understanding APIs
- Principles of modern Web APIs
- Make a great APIs
- Modeling APIs
- From modelling to API design
- Workshop



# Agenda day 2

- Working with REST
- Documenting your APIs
- Secure your APIs
- Testing your APIs
- Workshop

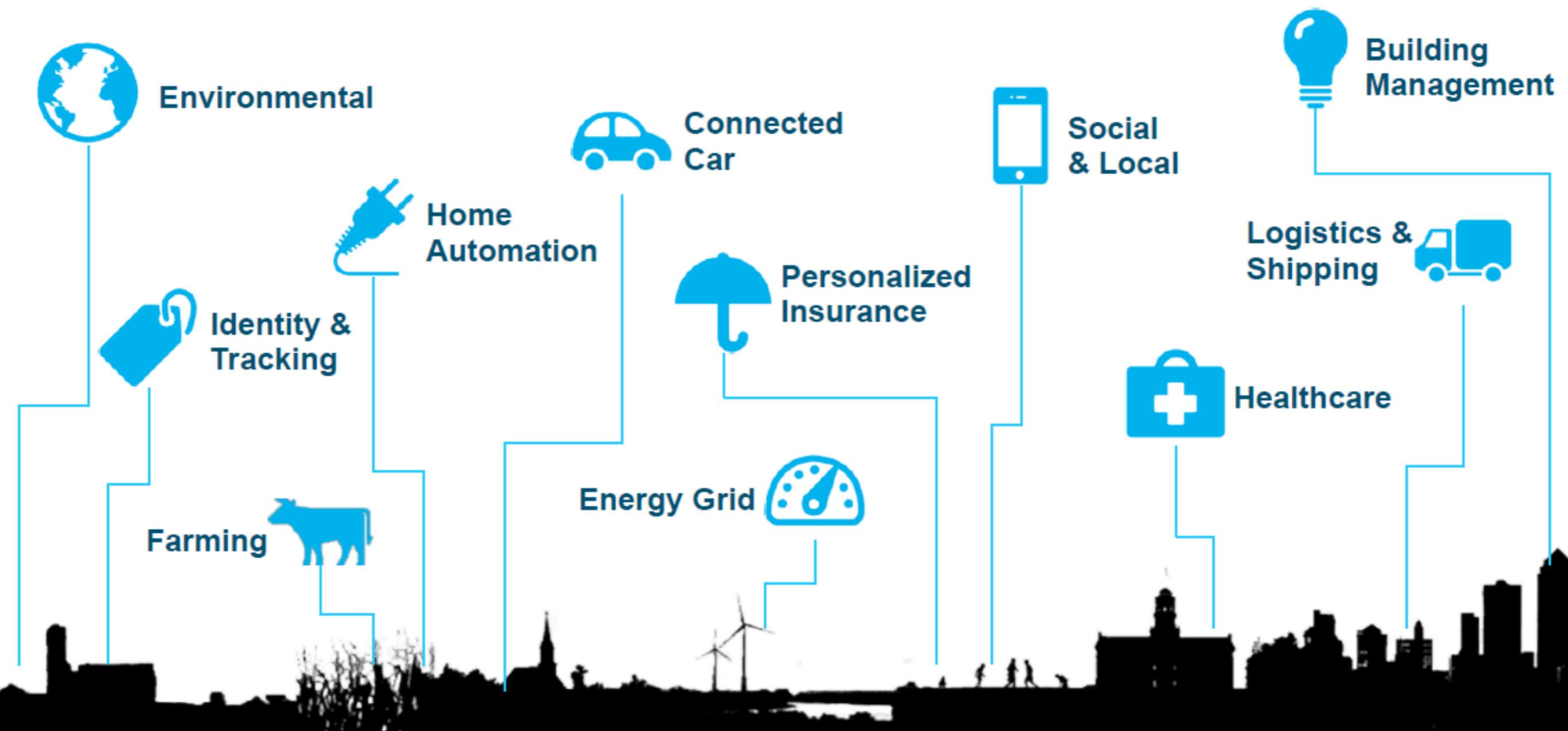


# Understanding APIs





# The world of APIs



# What is an APIs ?

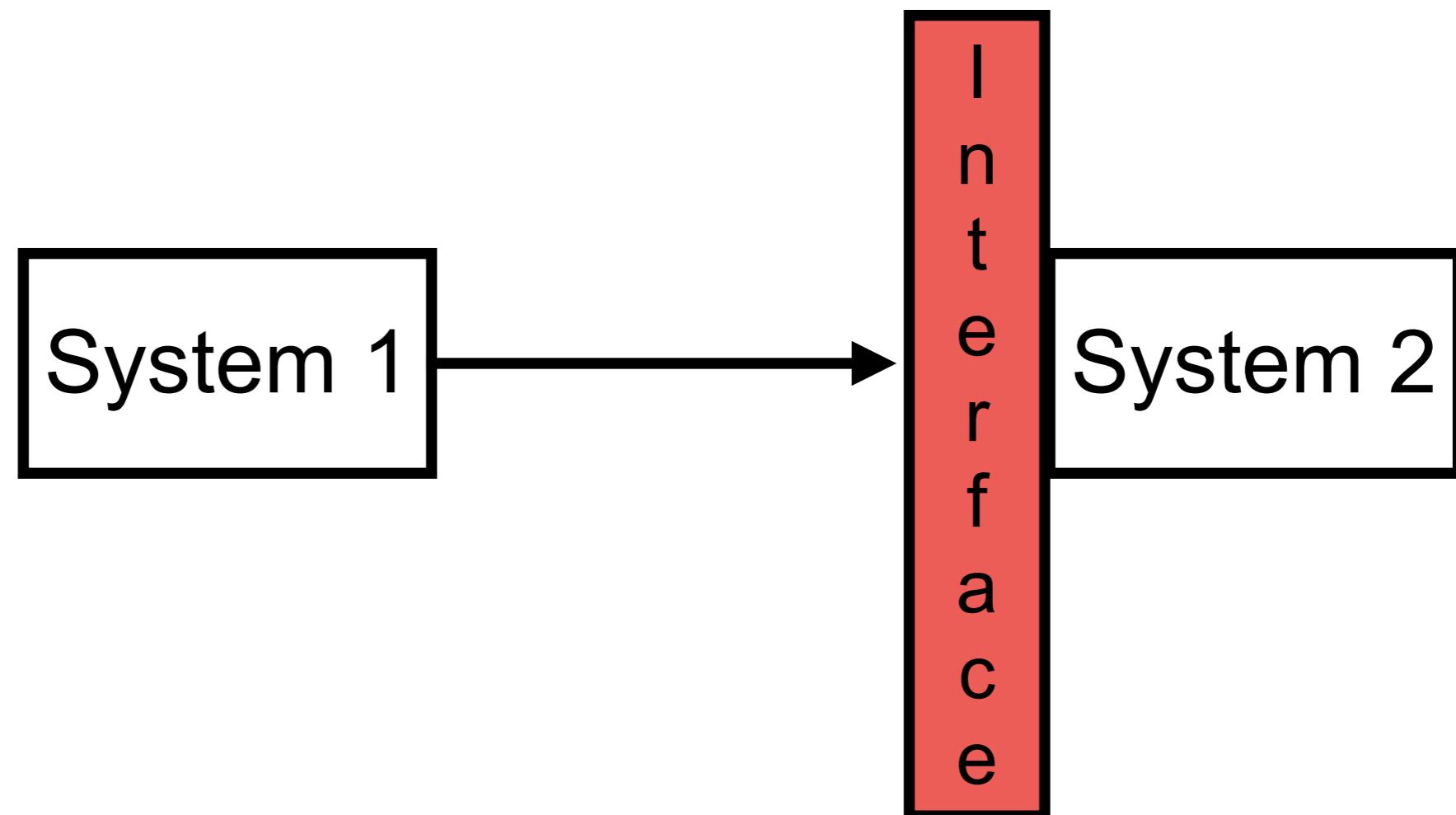


# Application Programming Interfaces



# What is an APIs ?

Software uses **interfaces** to communicate



# User Interface

# facebook

Facebook ช่วยคุณเชื่อมต่อและ  
แชร์กับผู้คนมากมายรอบตัวคุณ

อีเมลหรือหมายเลขโทรศัพท์มือถือ

รหัสผ่าน

เข้าสู่ระบบ

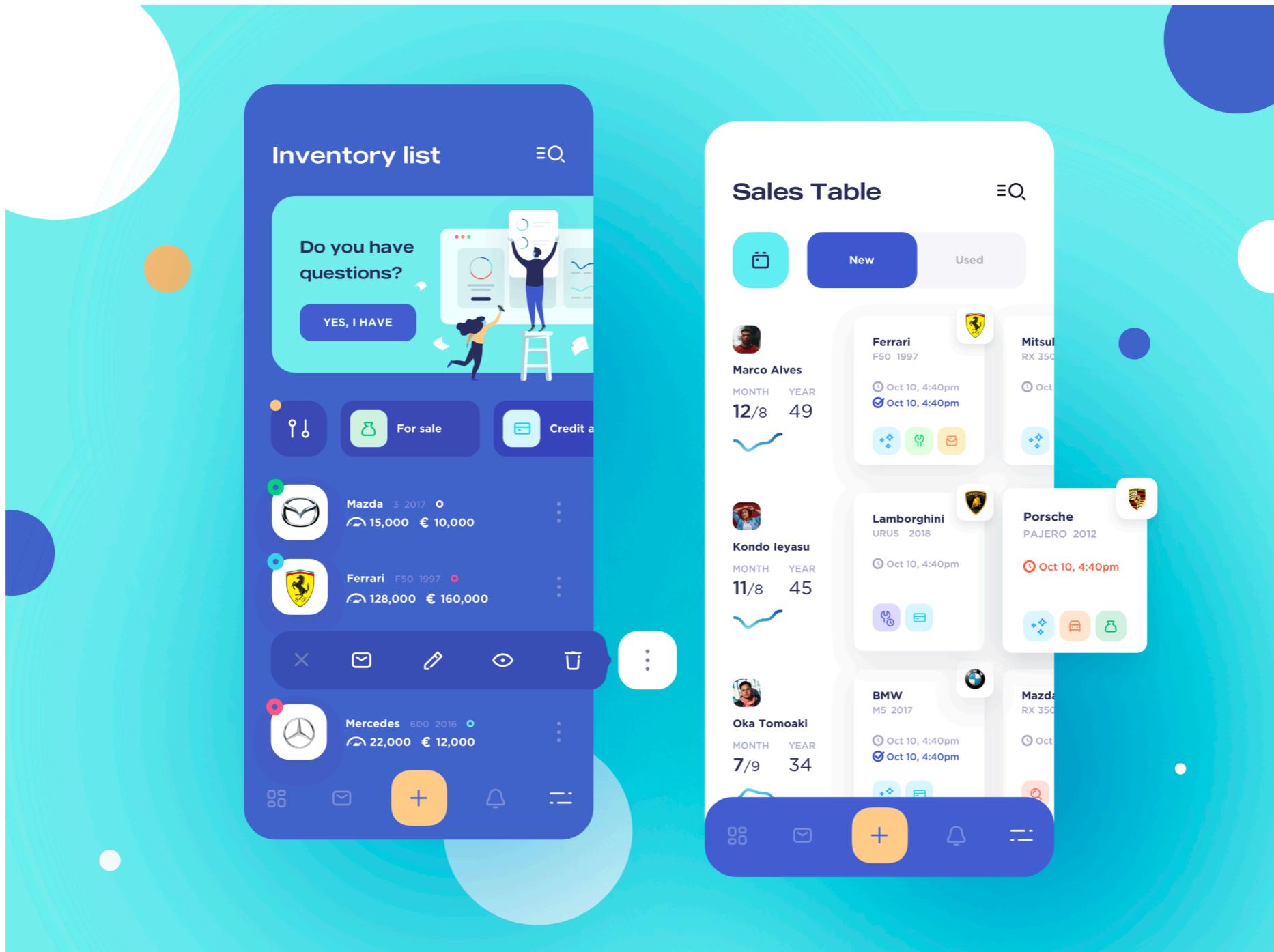
ลืมรหัสผ่าน ใช่หรือไม่

สร้างบัญชีใหม่

สร้างเพจ สำหรับบุคคลมีชื่อเสียง วงดนตรี หรือธุรกิจ



# User Interface



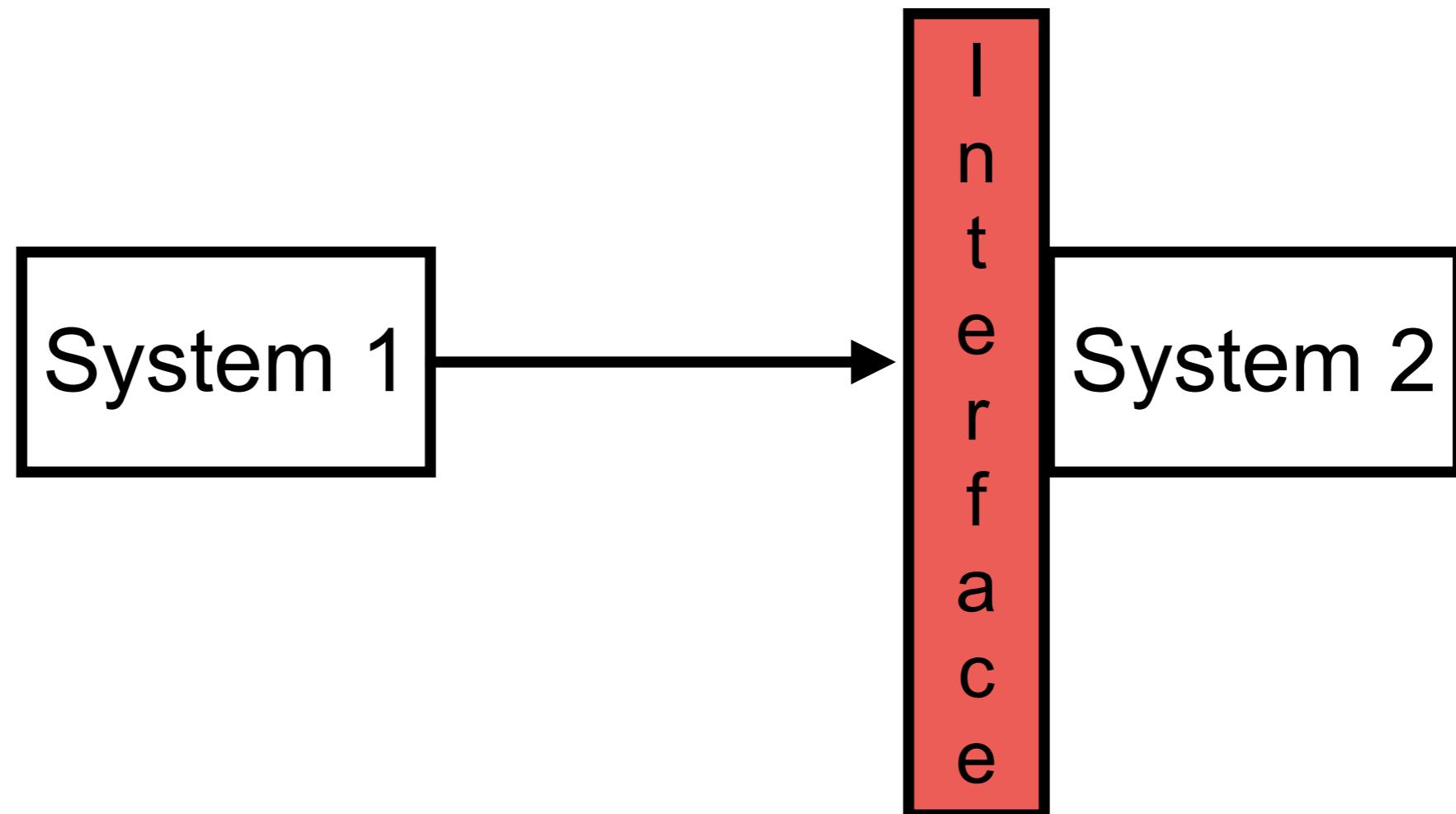
# Interface

```
{  
    age: "69",  
    age_display: "69 year",  
    + aliases: [...],  
    @context: "/terms/",  
    + lab: {...},  
    + biosample_ontology: {...},  
    references: [ ],  
    status: "released",  
    + dbxrefs: [...],  
    internal_tags: [ ],  
    description: "mammary gland, adenocarcinoma",  
    @id: "/biosamples/ENCBS000AAA/",  
    age_units: "year",  
    uuid: "56e94f2b-25ac-4c58-9828-f63b66220999",  
    parent_of: [ ],  
    + submitted_by: {...},  
    + documents: [...],  
    genetic_modifications: [ ],  
    + organism: {...},  
    health_status: "breast cancer (adenocarcinoma)",  
    + @type: [...],  
    alternate_acccessions: [ ],  
    url: "http://www.atcc.org/Products/All/HTB-22.aspx",
```



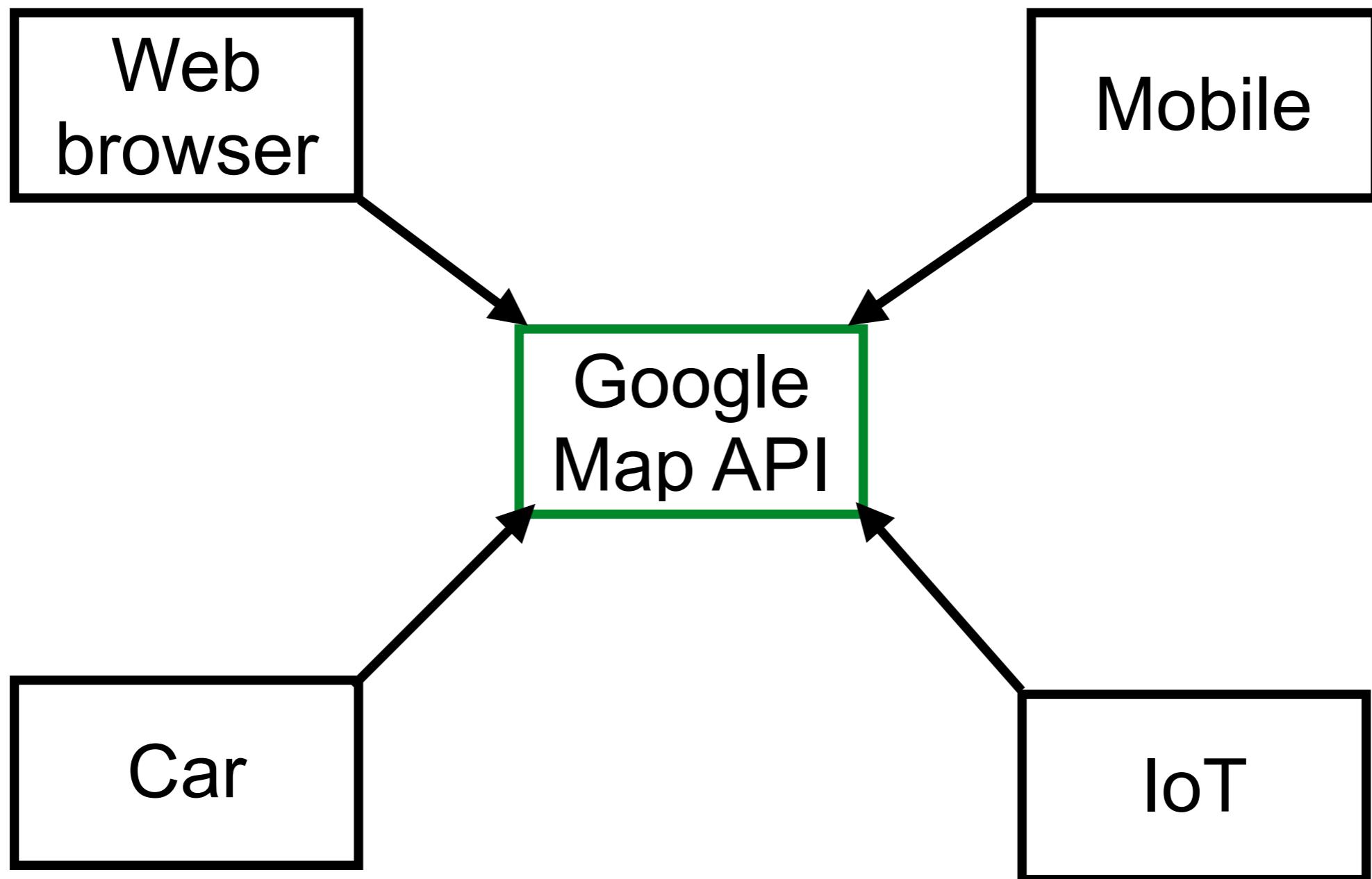
# What is an APIs ?

The interface that a software program presents to other programs, to humans

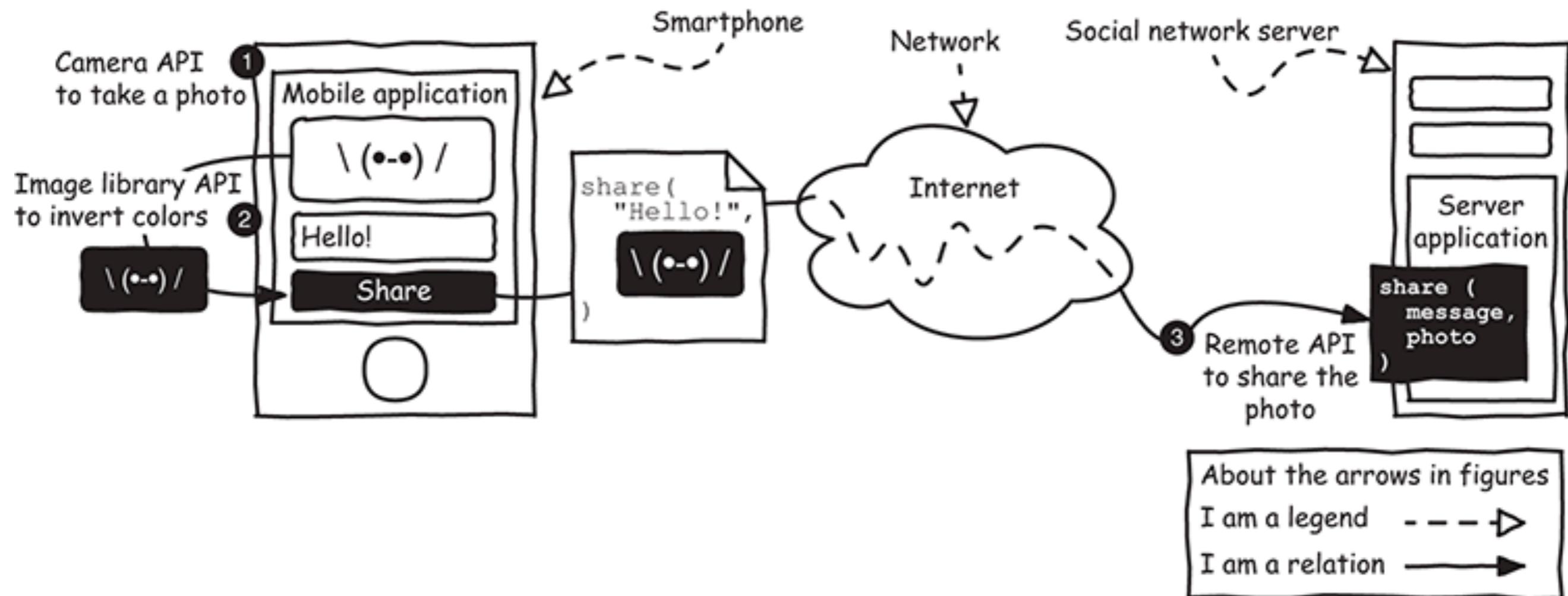


# Google Map

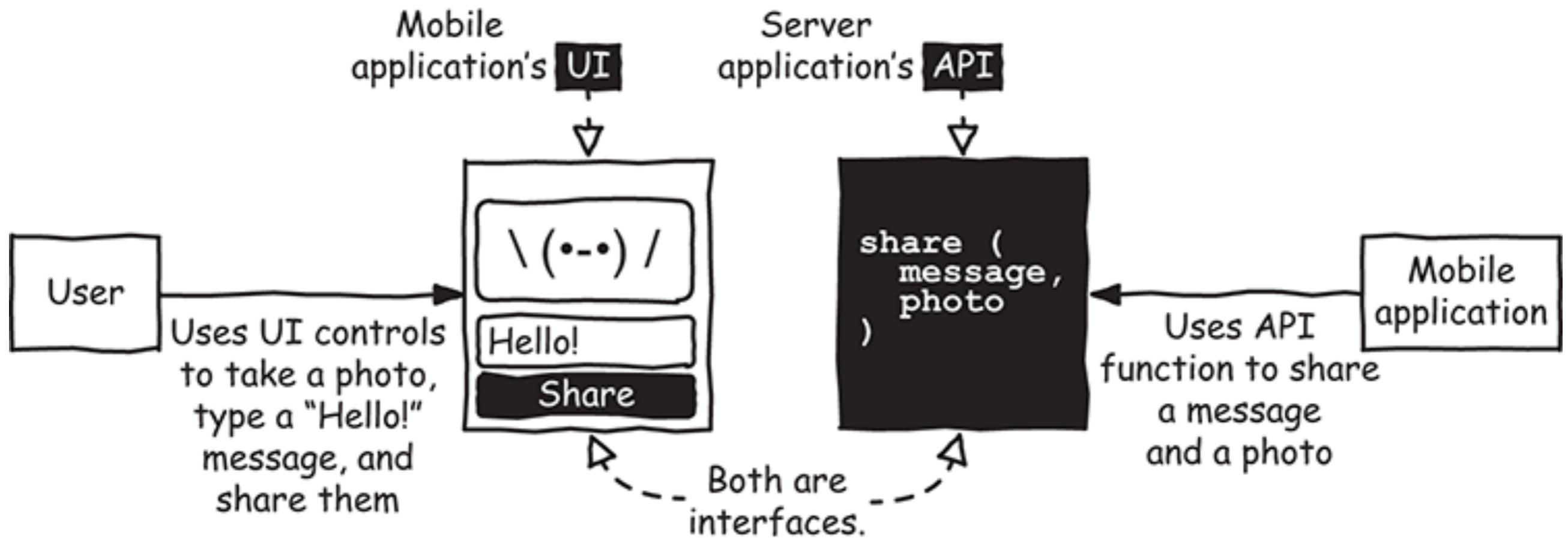




# Example of APIs with Mobile



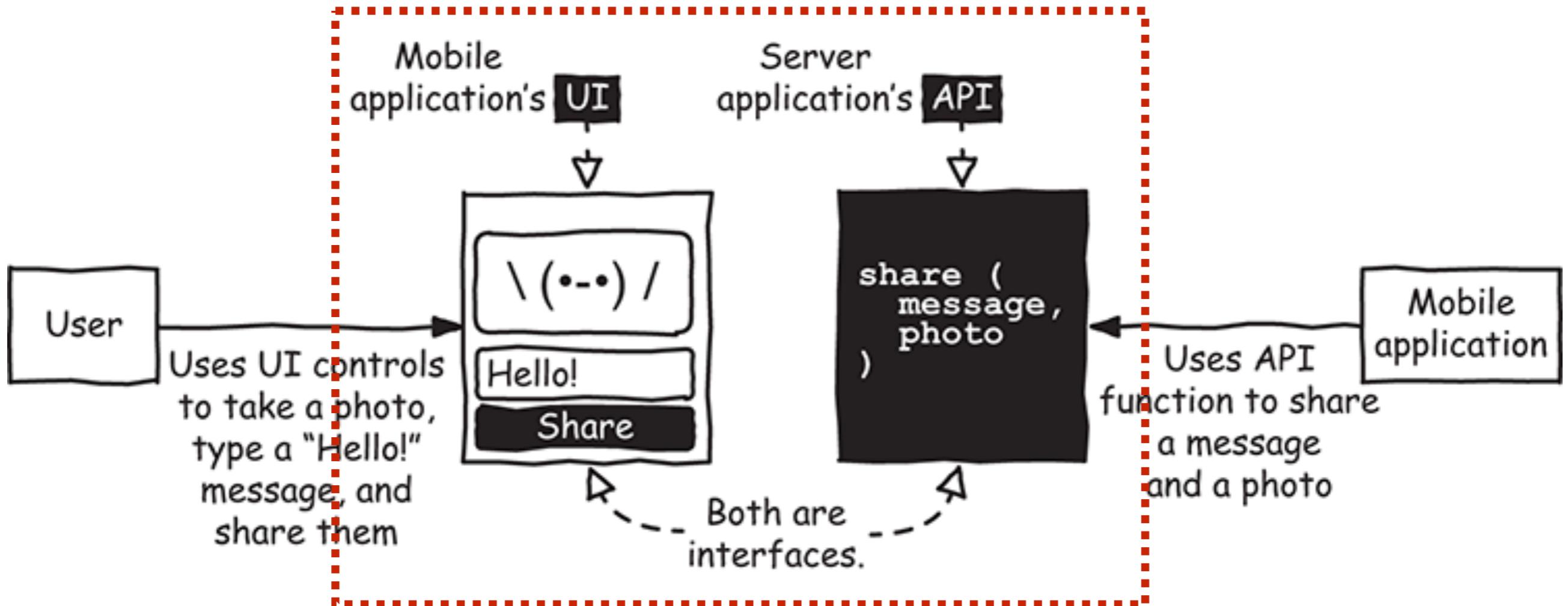
# UI to API

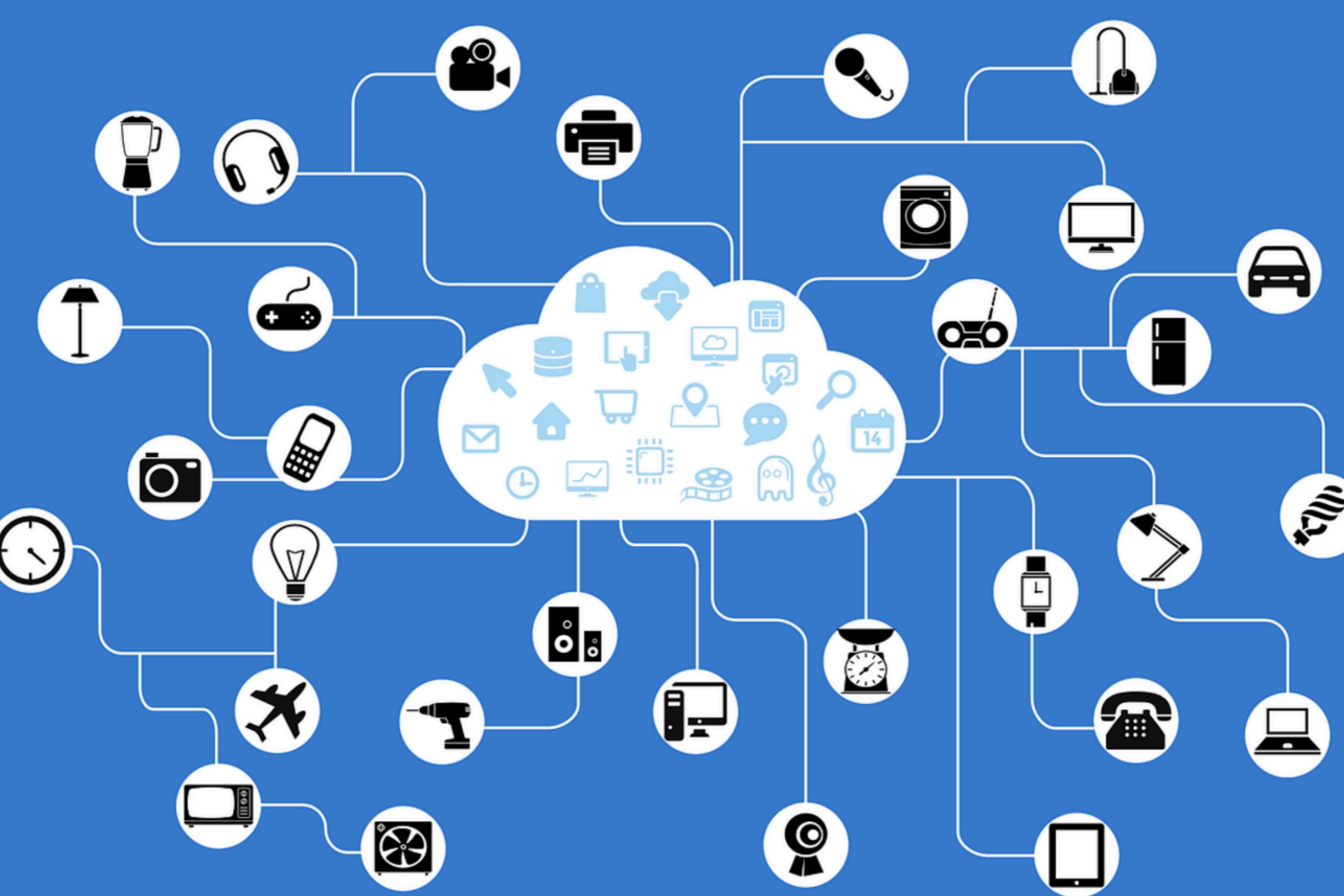


# Communication protocol ?

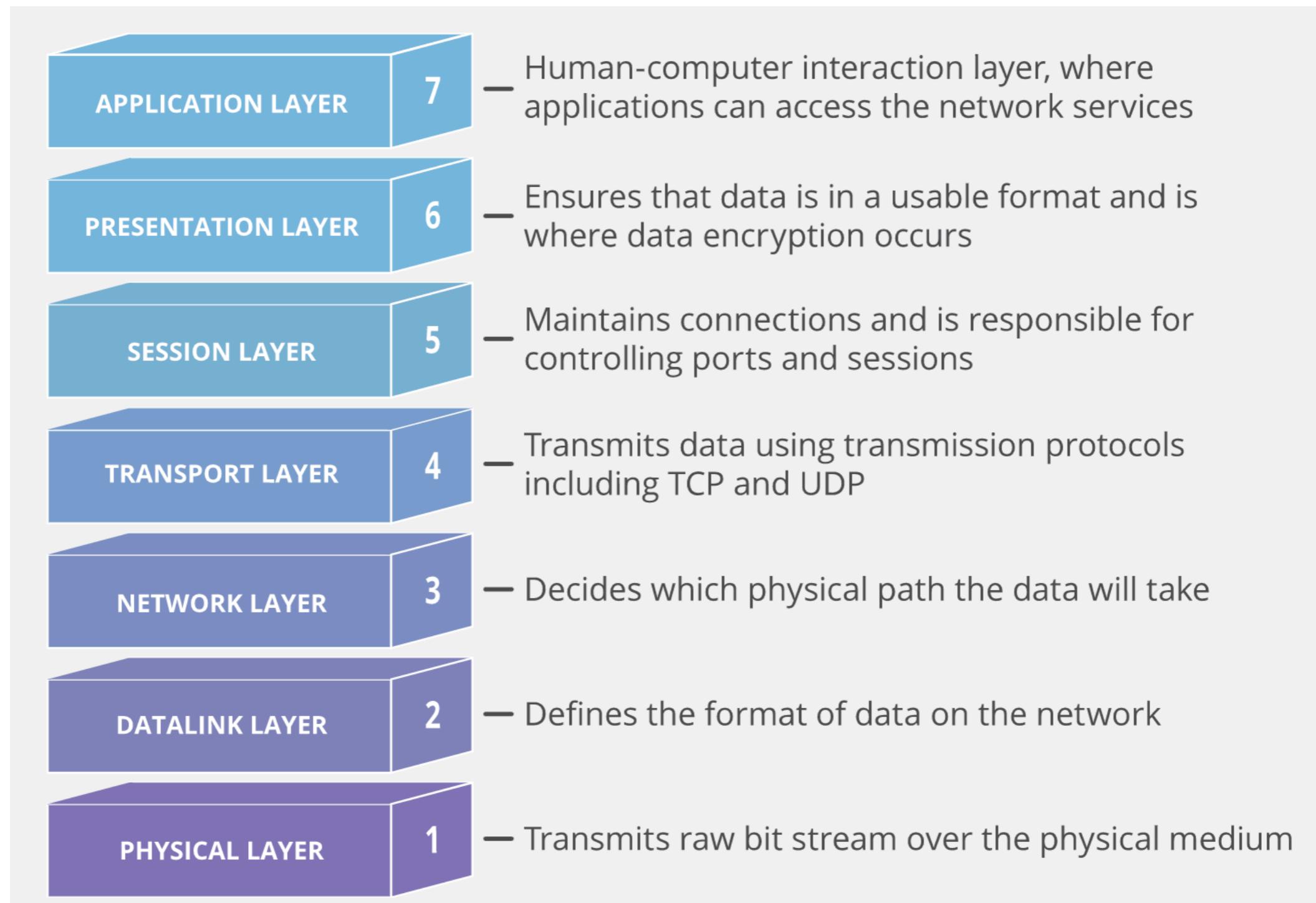


# Communication protocol ?





# Open Systems Interconnection (OSI)



<https://www.cloudflare.com/learning/network-layer/what-is-a-protocol/>



# Web APIs



# **HTTP**

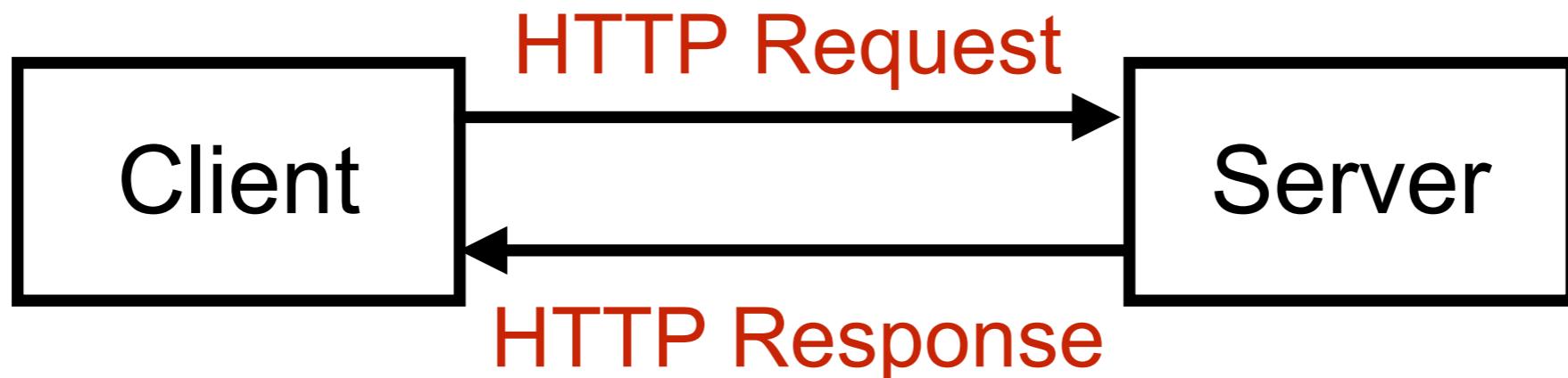
# **Hyper Text Transfer Protocol**



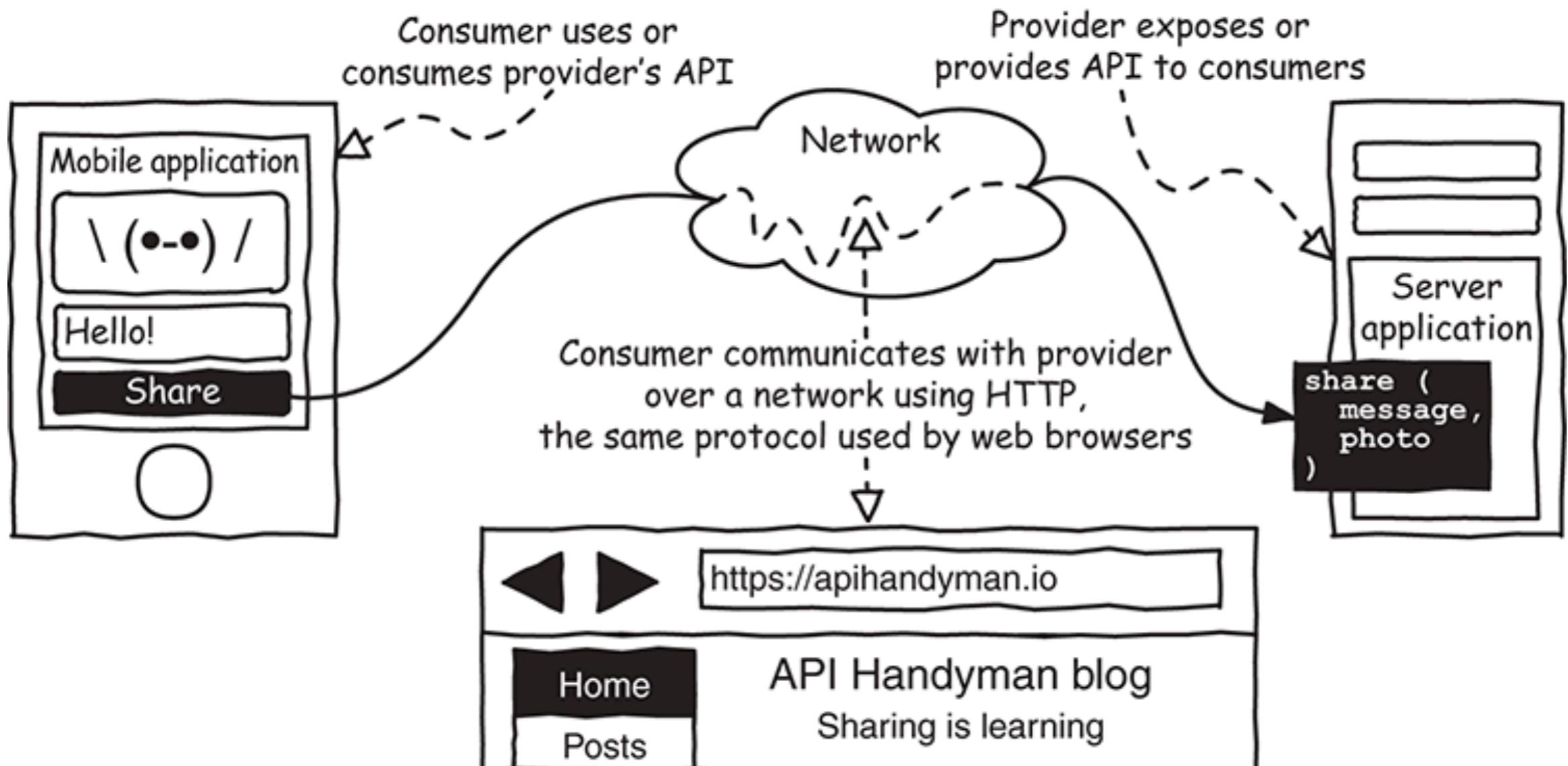
# HTTP ?

Foundation of World Wide Web (www)  
Application layer protocol

Designed to transfer information networked  
devices

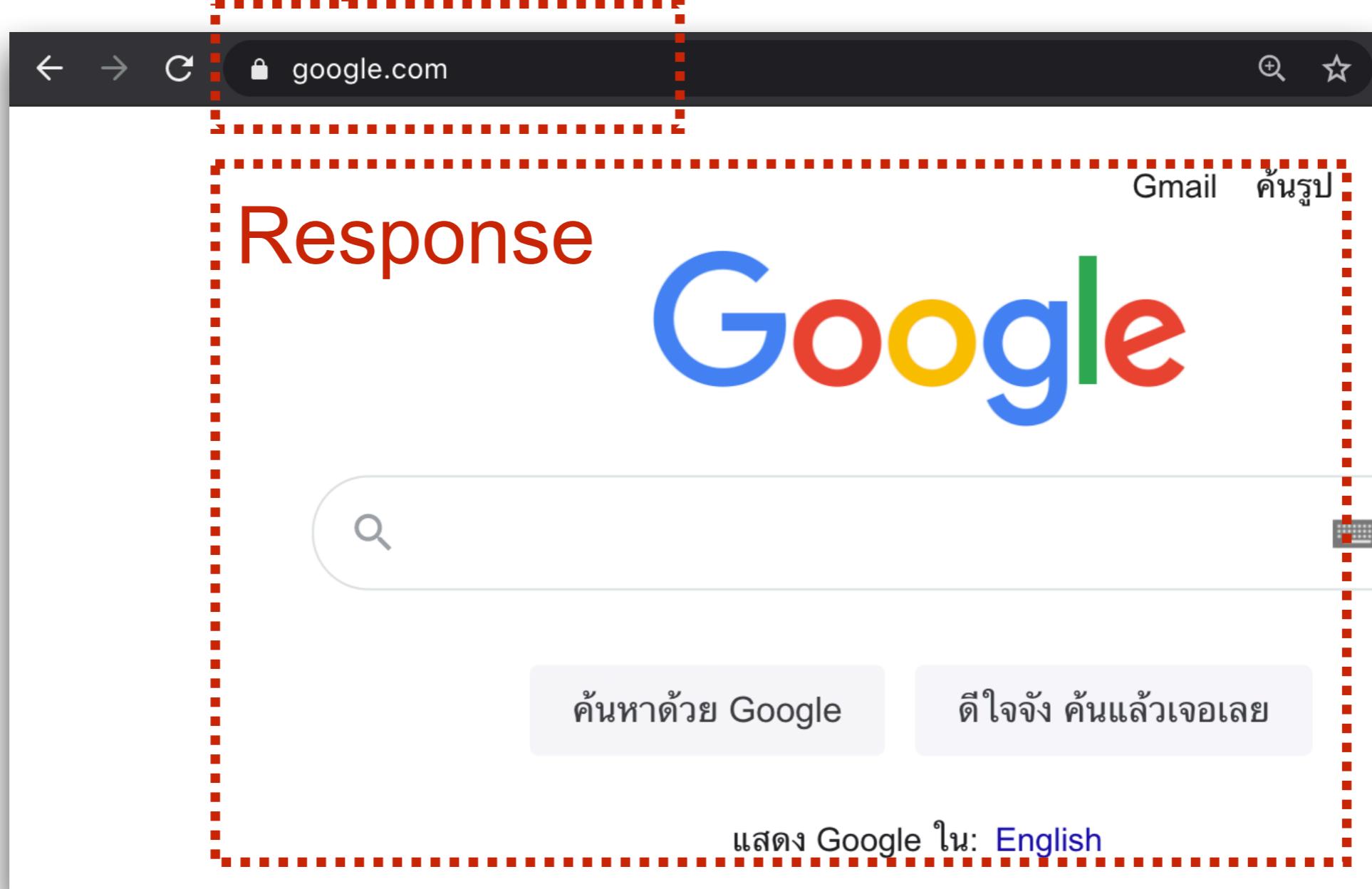


# HTTP



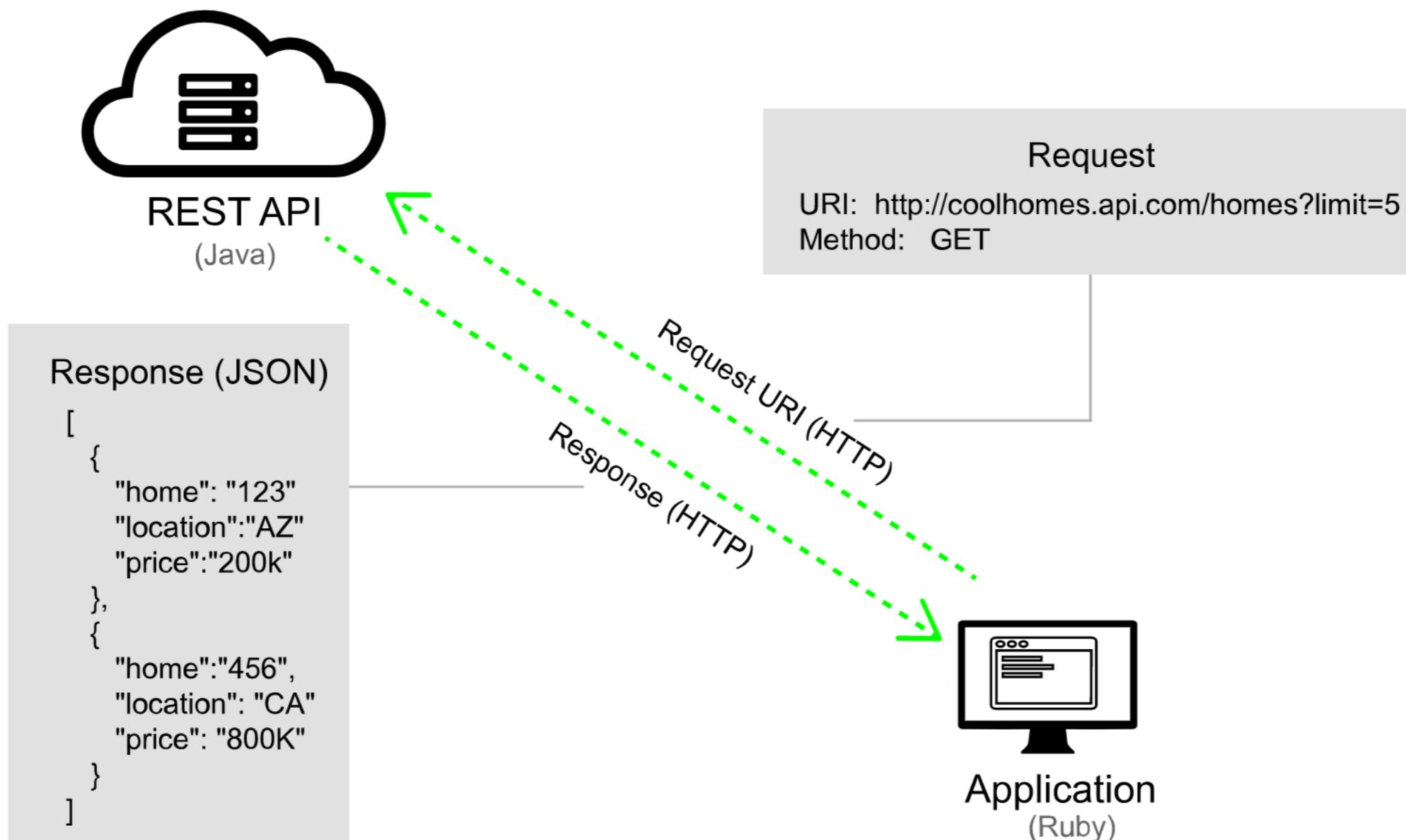
# Example of request and response

Request



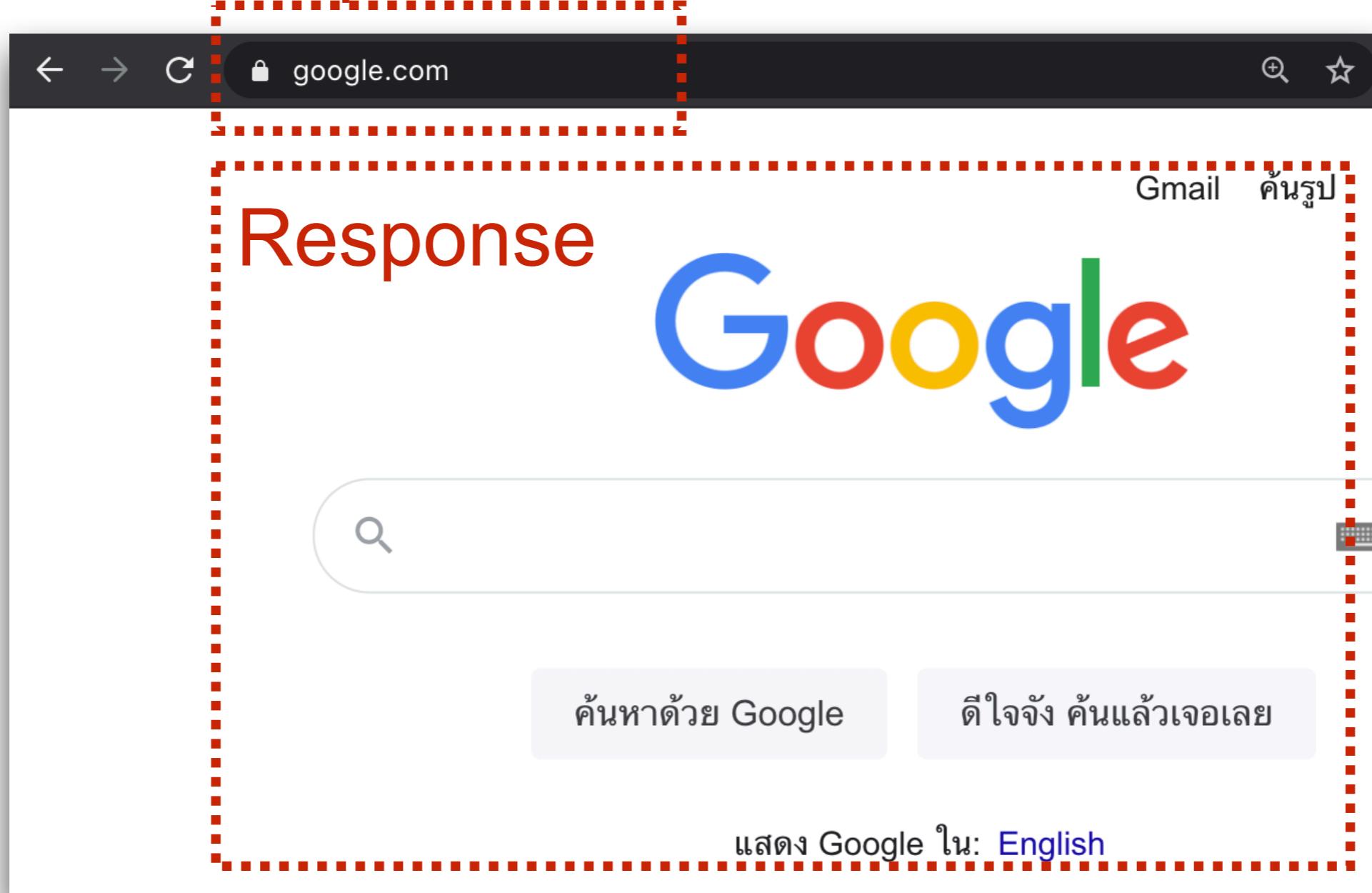
# Web APIs + HTTP

## Working with Request and Response



# Example

Request



Response



# Example !!

## ▼ General

**Request URL:** [https://www.google.com/complete/search?q&cp=0&c0jv0X9qyCv\\_az7sPufWhmAE.1607351250444&dpr=2.5](https://www.google.com/complete/search?q&cp=0&c0jv0X9qyCv_az7sPufWhmAE.1607351250444&dpr=2.5)

**Request Method:** GET

**Status Code:** 200

**Remote Address:** 216.58.221.196:443

**Referrer Policy:** origin

---

## ▼ Response Headers

**alt-svc:** h3-29=":443"; ma=2592000, h3-T051=":443"; ma=2592000, h3-43=":443"; ma=2592000, quic=":443"; ma=2592000; v="46,43"

**cache-control:** private, max-age=3600

**content-disposition:** attachment; filename="f.txt"

**content-encoding:** br



# Example !!

## ▼ Request Headers

**:authority:** www.google.com

**:method:** GET

**:path:** /complete/search?q&cp=0&client=psy-ab&xssi=t&gs\_ri  
51250444&dpr=2.5

**:scheme:** https

**accept:** \*/\*

**accept-encoding:** gzip, deflate, br

**accept-language:** en-US,en;q=0.9,th;q=0.8

**cookie:** CGIC=IocBdGV4dC9odG1sLGFwcGxpY2F0aW9uL3hodG1sK3h-



# Example !!

## ▼ Query String Parameters

[view source](#)[view URL encoded](#)**q:****cp: 0****client: psy-ab****xssi: t****gs\_ri: gws-wiz****hl: en-TH****authuser: 0****psi: 0jv0X9qyCv\_az7sPufWhmAE.1607351250444****dpr: 2.5**

# Let's Start with Web APIs



# APIs

**REST**

**RPC (Remote Procedure Call)**

**GraphQL**

**WebSocket**

**WebHook**

**etc...**



# Request and Response APIs

Using HTTP

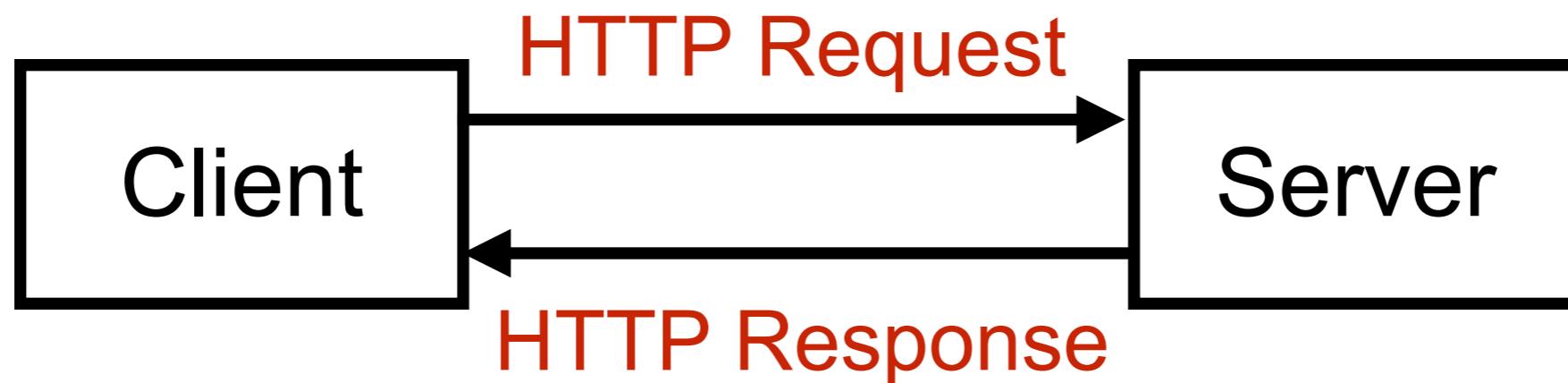
APIs define a set of **endpoints**  
**Client and Server**



# Working with HTTP

Client make HTTP requests for data to endpoint on server

Server returns responses



# HTTP Request



# HTTP Request ?

HTTP version

URL

HTTP method

HTTP request header

HTTP body (optional)

```
1 | GET / HTTP/1.1
2 | Host: developer.mozilla.org
3 | Accept-Language: fr
```



# HTTP Request ?

```
1 | GET / HTTP/1.1
2 | Host: developer.mozilla.org
3 | Accept-Language: fr
```

HTTP version = 1.1



# HTTP Request ?

```
1 | GET / HTTP/1.1
2 | Host: developer.mozilla.org
3 | Accept-Language: fr
```

HTTP method = GET



# HTTP Request ?

```
1 | GET / HTTP/1.1
2 | Host: developer.mozilla.org
3 | Accept-Language: fr
```

PATH = /



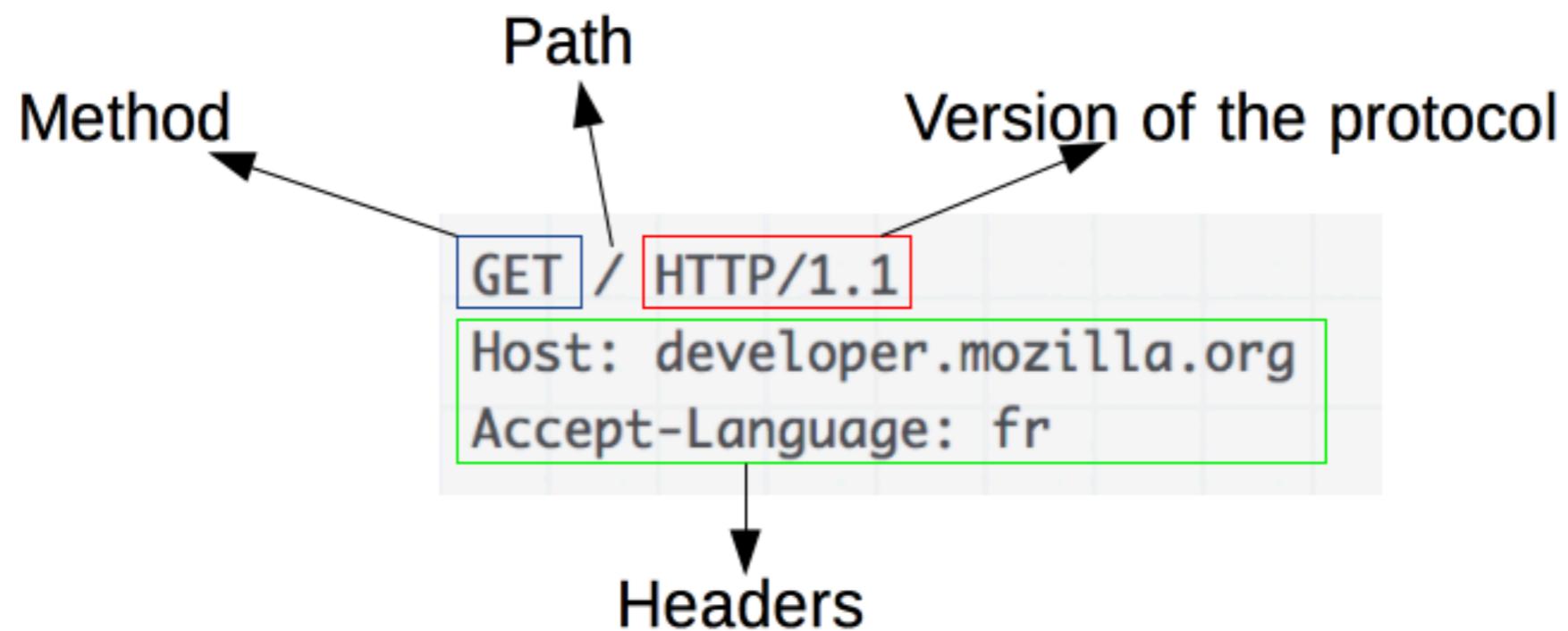
# HTTP Request ?

```
1 | GET / HTTP/1.1
2 | Host: developer.mozilla.org
3 | Accept-Language: fr
```

HTTP request header



# HTTP Request ?



# HTTP method ?

HTTP defines a set of **request methods** to indicate the desired action to be performed for a given **resource**.

Called “**HTTP Verbs**”



# HTTP method ?

HTTP method	Description
GET	Retrive data
POST	Create data
PUT	Update data
DELETE	Delete data

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods>



# Path ?

The path of the **resource** to fetch from URL

`http://developer.mozilla.org/`

**URL = Uniform Resource Locator**

`https://en.wikipedia.org/wiki/URL`



# Path ?

The path of the **resource** to fetch from URL

`http://developer.mozilla.org/`

Protocol = http, https

`https://en.wikipedia.org/wiki/URL`



# Path ?

The path of the **resource** to fetch from URL

http://developer.mozilla.org/

Domain name

https://en.wikipedia.org/wiki/Domain\_name



# Path ?

The path of the **resource** to fetch from URL

http://developer.mozilla.org:80/

TCP port = 80 (default)

https://en.wikipedia.org/wiki/Domain\_name



# Path ?

The path of the **resource** to fetch from URL

`http://developer.mozilla.org:80/`

Path = / = resource to access/use

`https://en.wikipedia.org/wiki/Domain_name`



# Resources ?

Employee  
Product  
Message  
Department



# Example

HTTP method	Path	Description
GET	/product	Retrieve all product
GET	/product/1	Retrieve product detail of 1
POST	/product	Create new product
PUT	/product/1	Update existing product 1
DELETE	/product/1	Delete product 1



# More complexity

**/product/1 or /product/?id=1**



Path variables

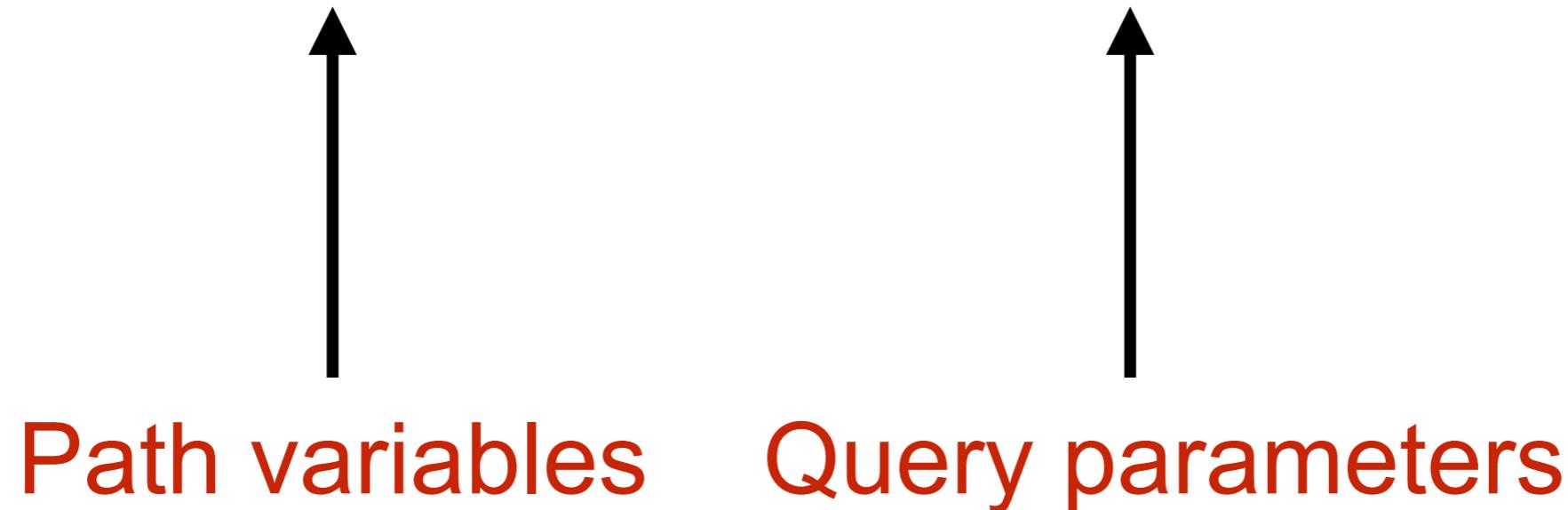


Query parameters



# More complexity

`/product/1` or `/product/?id=1`



# **Path variable vs Query parameter**



# Recommendation

## Path variables

Used to **identify** from resource directly

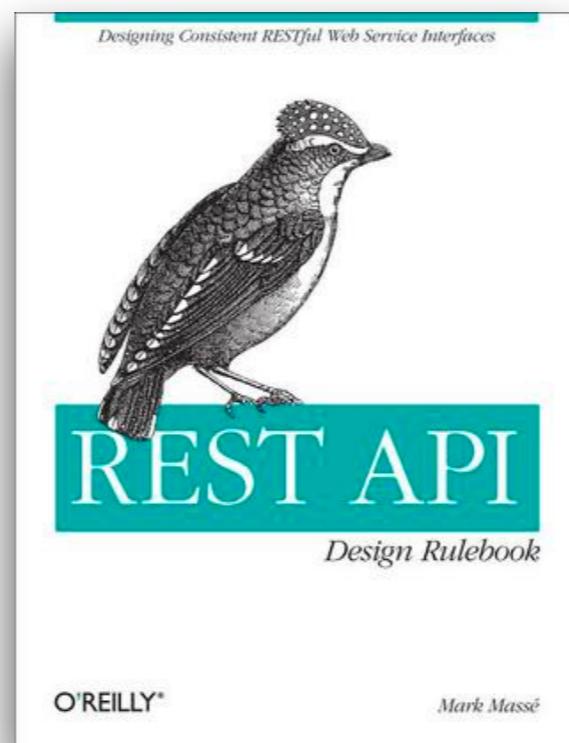
## Query parameters

Used to **sort** or **filter** from resource

<https://hackernoon.com/restful-api-designing-guidelines-the-best-practices-60e1d954e7c9>



# REST API Design rules



# Use / for hierarchy data

Country => province

**http://api/countries/thailand/provinces**



# Naming

Using hyphen (-) over underscore (\_)

Using lowercase

**<http://api.com/blog/hello-my-world>**



# Workshop



# HTTP Response



# HTTP Response ?

HTTP version  
HTTP status code  
HTTP response header  
HTTP body (optional)

```
1 | HTTP/1.1 200 OK
2 | Date: Sat, 09 Oct 2010 14:28:02 GMT
3 | Server: Apache
4 | Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
5 | ETag: "51142bc1-7449-479b075b2891b"
6 | Accept-Ranges: bytes
7 | Content-Length: 29769
8 | Content-Type: text/html
9 |
10| <!DOCTYPE html... (here comes the 29769 bytes of the requ
```



# HTTP Response ?

```
1 | HTTP/1.1 200 OK
2 | Date: Sat, 09 Oct 2010 14:28:02 GMT
3 | Server: Apache
4 | Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
5 | ETag: "51142bc1-7449-479b075b2891b"
6 | Accept-Ranges: bytes
7 | Content-Length: 29769
8 | Content-Type: text/html
9 |
10| <!DOCTYPE html... (here comes the 29769 bytes of the requ
```

HTTP version = 1.1



# HTTP Response ?

```
1 | HTTP/1.1 200 OK
2 | Date: Sat, 09 Oct 2010 14:28:02 GMT
3 | Server: Apache
4 | Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
5 | ETag: "51142bc1-7449-479b075b2891b"
6 | Accept-Ranges: bytes
7 | Content-Length: 29769
8 | Content-Type: text/html
9 |
10| <!DOCTYPE html... (here comes the 29769 bytes of the requ
```

HTTP status code = 200



# HTTP Response ?

```
1 | HTTP/1.1 200 OK
2 | Date: Sat, 09 Oct 2010 14:28:02 GMT
3 | Server: Apache
4 | Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
5 | ETag: "51142bc1-7449-479b075b2891b"
6 | Accept-Ranges: bytes
7 | Content-Length: 29769
8 | Content-Type: text/html
9 |
10| <!DOCTYPE html... (here comes the 29769 bytes of the requ
```

HTTP response header



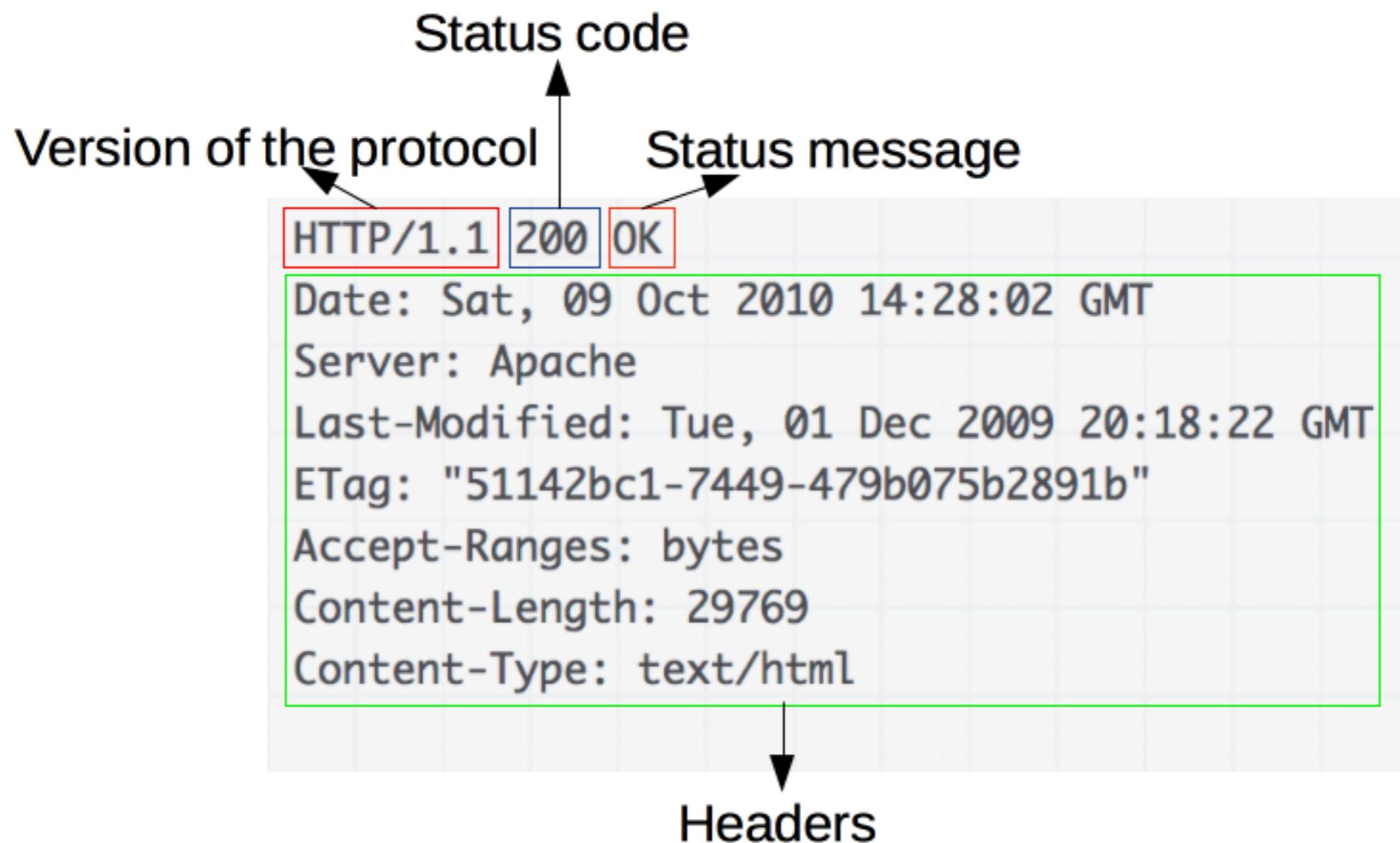
# HTTP Response ?

```
1 | HTTP/1.1 200 OK
2 | Date: Sat, 09 Oct 2010 14:28:02 GMT
3 | Server: Apache
4 | Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
5 | ETag: "51142bc1-7449-479b075b2891b"
6 | Accept-Ranges: bytes
7 | Content-Length: 29769
8 | Content-Type: text/html
9 |
10| <!DOCTYPE html... (here comes the 29769 bytes of the requ
```

HTTP body



# HTTP Response ?



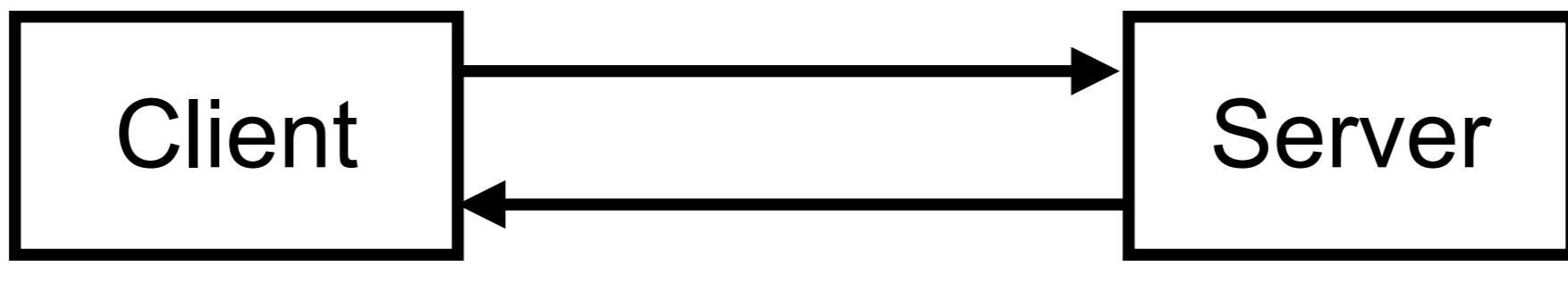
# HTTP Status Code

<https://developer.mozilla.org/en-US/docs/Web/HTTP>Status>



# HTTP Status Code

HTTP response status codes indicate whether a specific HTTP request has been successfully completed.



# 5 groups of status codes

Code	Name
100-199	Information responses
200-299	Successful responses
300-399	Redirects
400-499	Client errors
500-599	Server error



# Successful Responses

Code	Name
200	OK
201	Created
202	Accepted



# Redirection message

Code	Name
301	Move permanently
302	Found, Move temporarily



# Client Errors Responses

Code	Name
400	Bad request
401	Bad authorized
403	Forbidden
404	Not found
405	Method not allowed



# Server Errors Responses

Code	Name
500	Internal server error
502	Bad gateway
503	Service unavailable
504	Gateway timeout

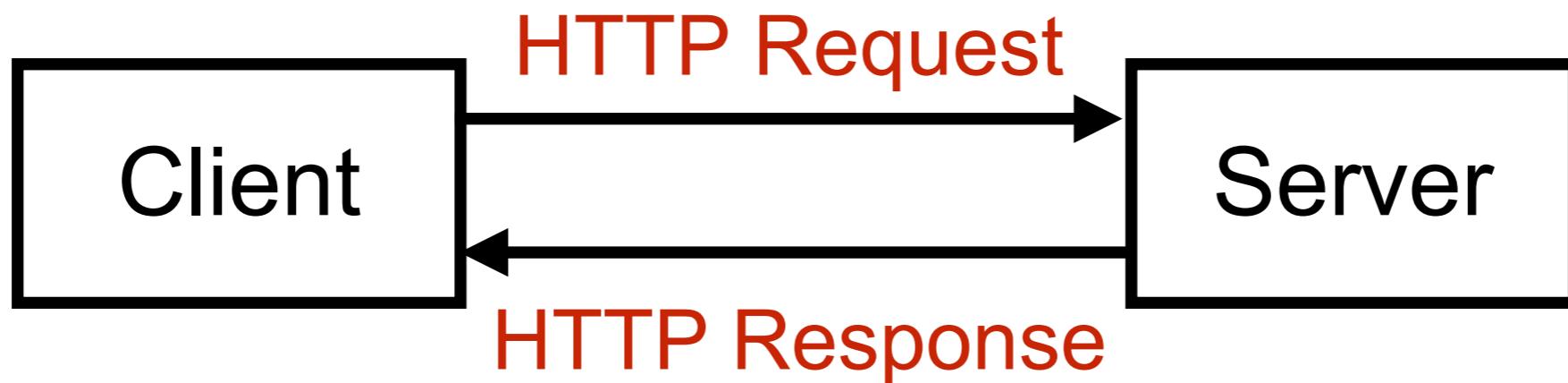


# HTTP Body



# HTTP Body

Information's format of request and response body  
Plain text, XML, JSON, Binary ... etc



# What is JSON ?

**JavaScript Object Notation**

**Lightweight** format to store and transport data

**Self-descriptive** and easy to understand

<https://www.json.org/json-en.html>



# Example of JSON message

## List of products

```
{  
  "products": [  
    { "id": 1, "name": "name 1", "price": 1.00},  
    { "id": 2, "name": "name 2", "price": 2.00},  
    { "id": 3, "name": "name 3", "price": 3.00},  
    { "id": 4, "name": "name 4", "price": 4.00},  
    { "id": 5, "name": "name 5", "price": 5.00},  
  ]  
}
```



# Example of JSON message

```
{  
    "products": [  
        { "id": 1, "name": "name 1", "price": 1.00},  
        { "id": 2, "name": "name 2", "price": 2.00},  
        { "id": 3, "name": "name 3", "price": 3.00},  
        { "id": 4, "name": "name 4", "price": 4.00},  
        { "id": 5, "name": "name 5", "price": 5.00},  
    ]  
}
```

{ } = Object



# Example of JSON message

```
{  
    "products": [  
        { "id": 1, "name": "name 1", "price": 1.00},  
        { "id": 2, "name": "name 2", "price": 2.00},  
        { "id": 3, "name": "name 3", "price": 3.00},  
        { "id": 4, "name": "name 4", "price": 4.00},  
        { "id": 5, "name": "name 5", "price": 5.00},  
    ]  
}
```

[ ] = List or Array



# Example of JSON message

```
{  
    Key  
    "products": [  
        { "id": 1, "name": "name 1", "price": 1.00},  
        { "id": 2, "name": "name 2", "price": 2.00},  
        { "id": 3, "name": "name 3", "price": 3.00},  
        { "id": 4, "name": "name 4", "price": 4.00},  
        { "id": 5, "name": "name 5", "price": 5.00},  
    ]  
}
```



# Example of JSON message

```
{  
    "Key": ["products": [  
        {"id": 1, "name": "name 1", "price": 1.00},  
        {"id": 2, "name": "name 2", "price": 2.00},  
        {"id": 3, "name": "name 3", "price": 3.00},  
        {"id": 4, "name": "name 4", "price": 4.00},  
        {"id": 5, "name": "name 5", "price": 5.00},  
    ]]  
}
```



# JSON syntax rules

Data is in name/value pairs

Data is separated by comma (,)

Curly braces ({} ) holds objects

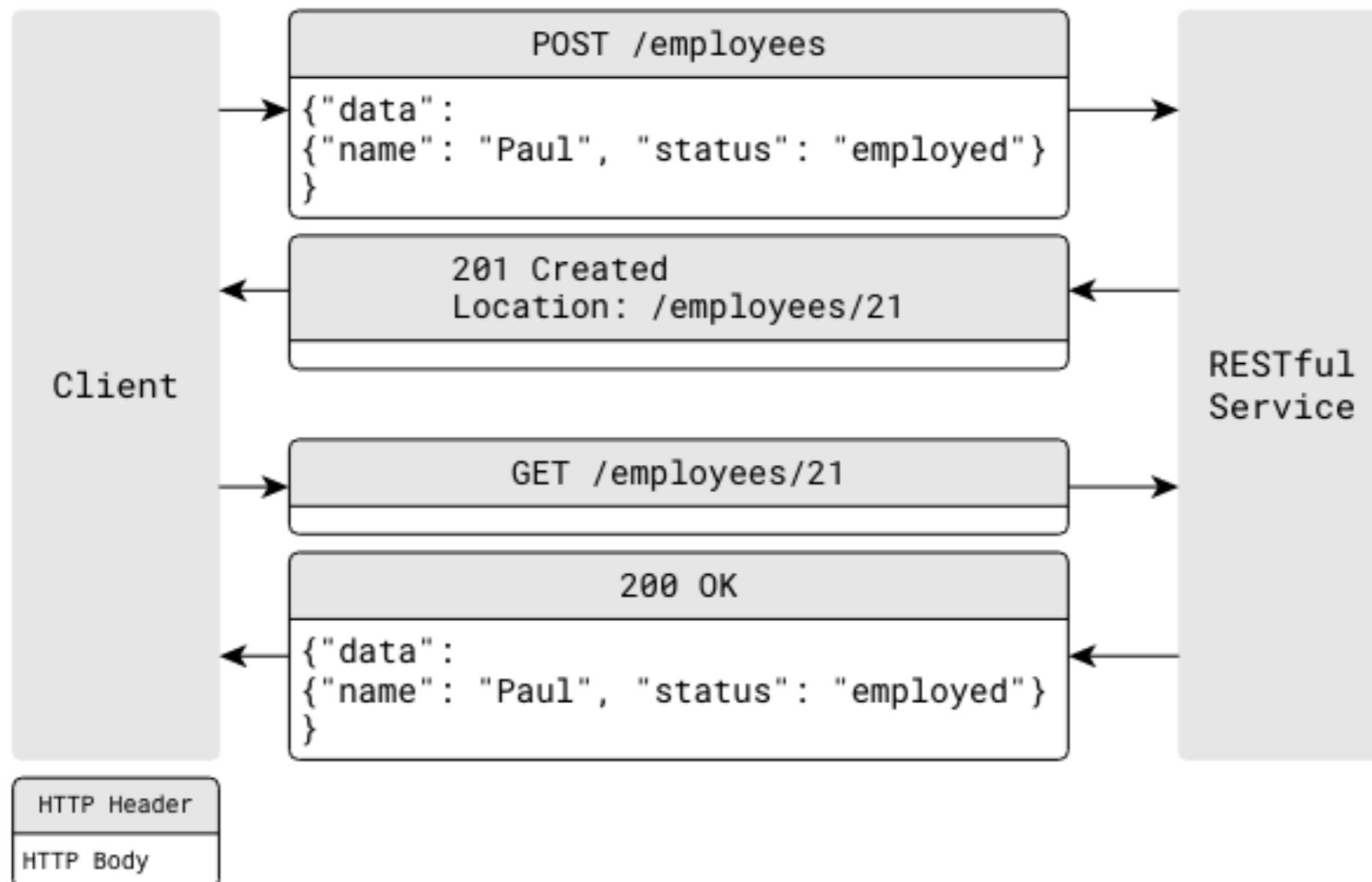
Square braces ([]) holds arrays/lists



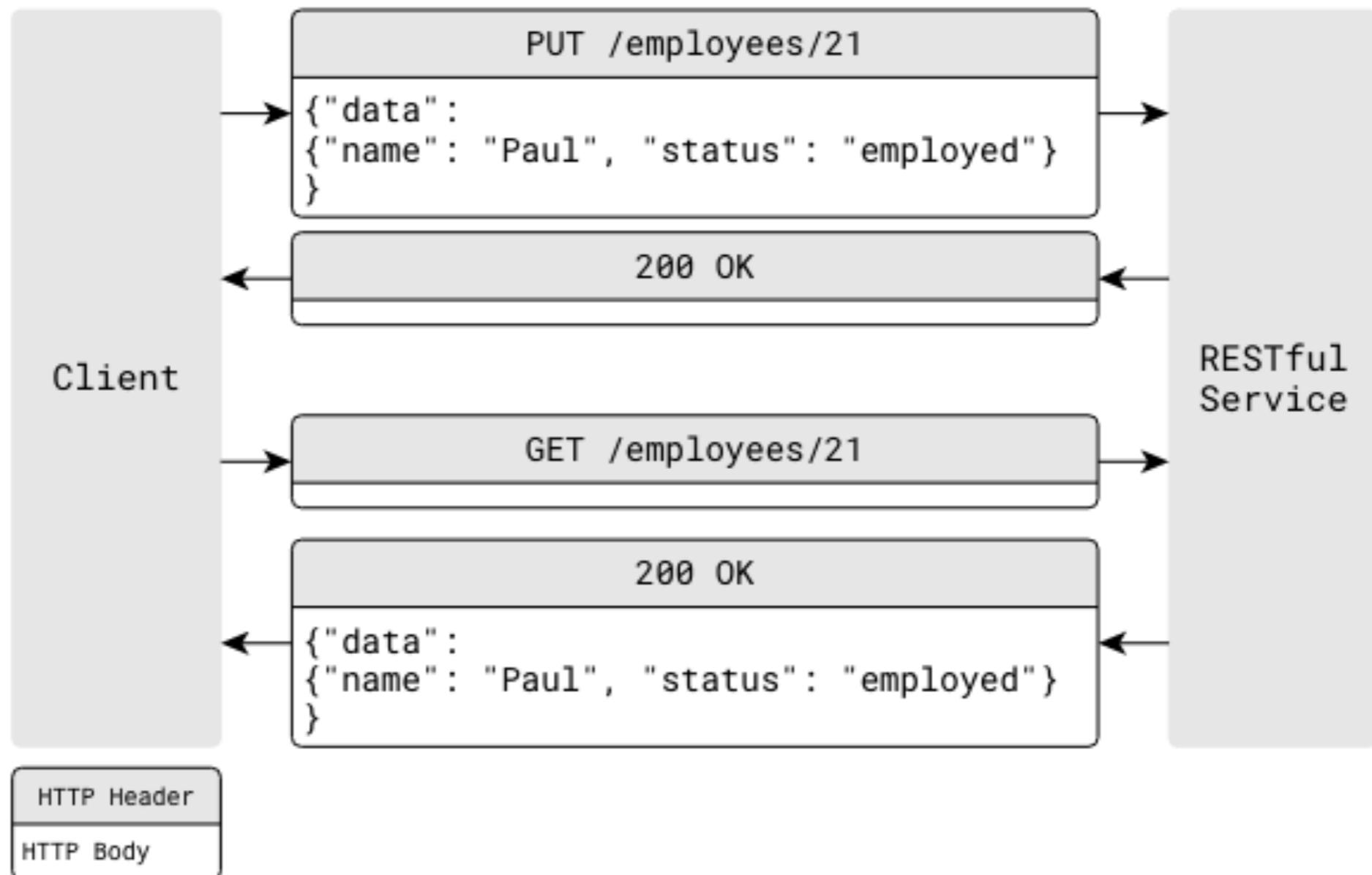
# Example



# Create new resource



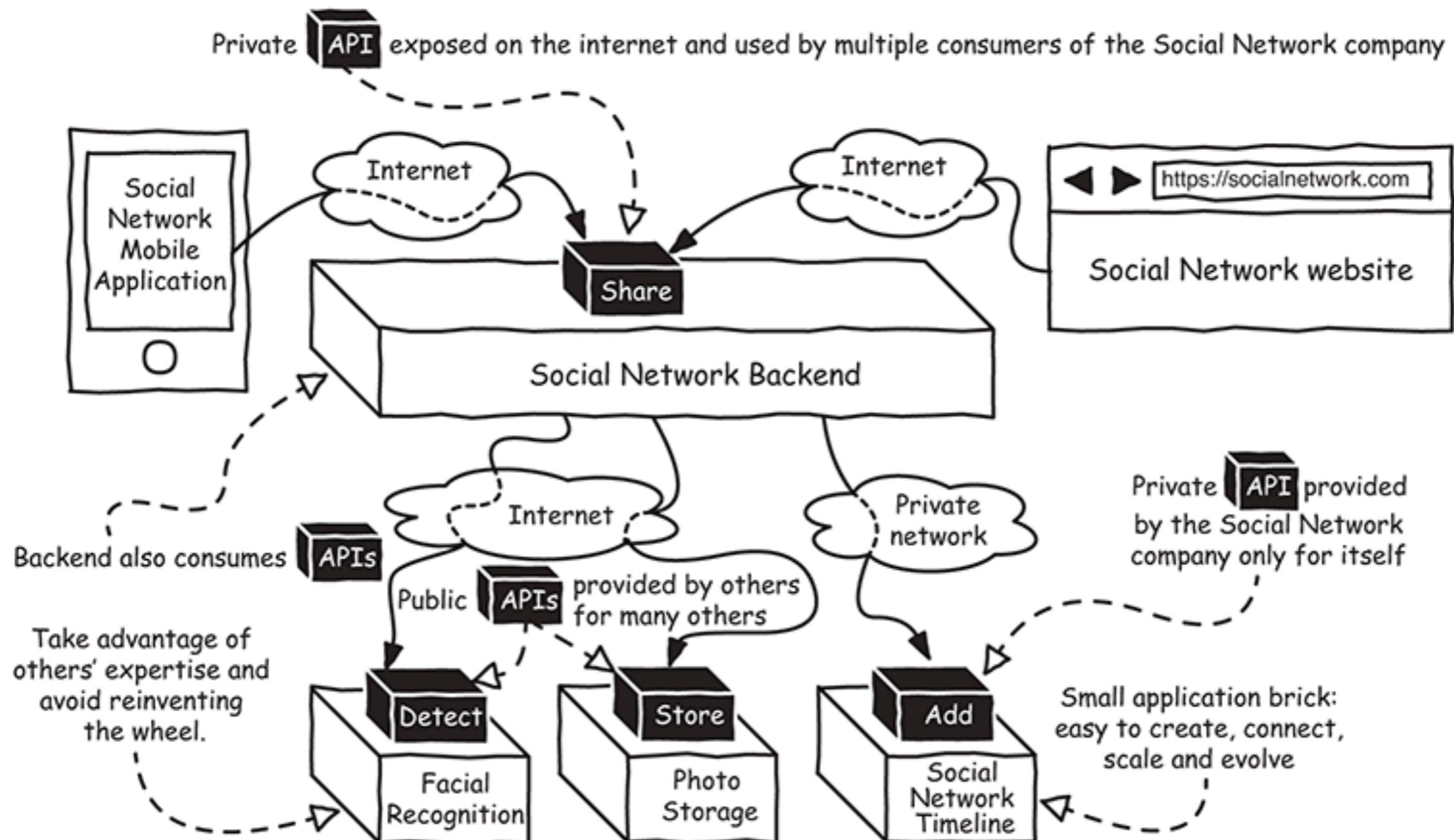
# Update a resource



# Workshop



# Complexity in real world



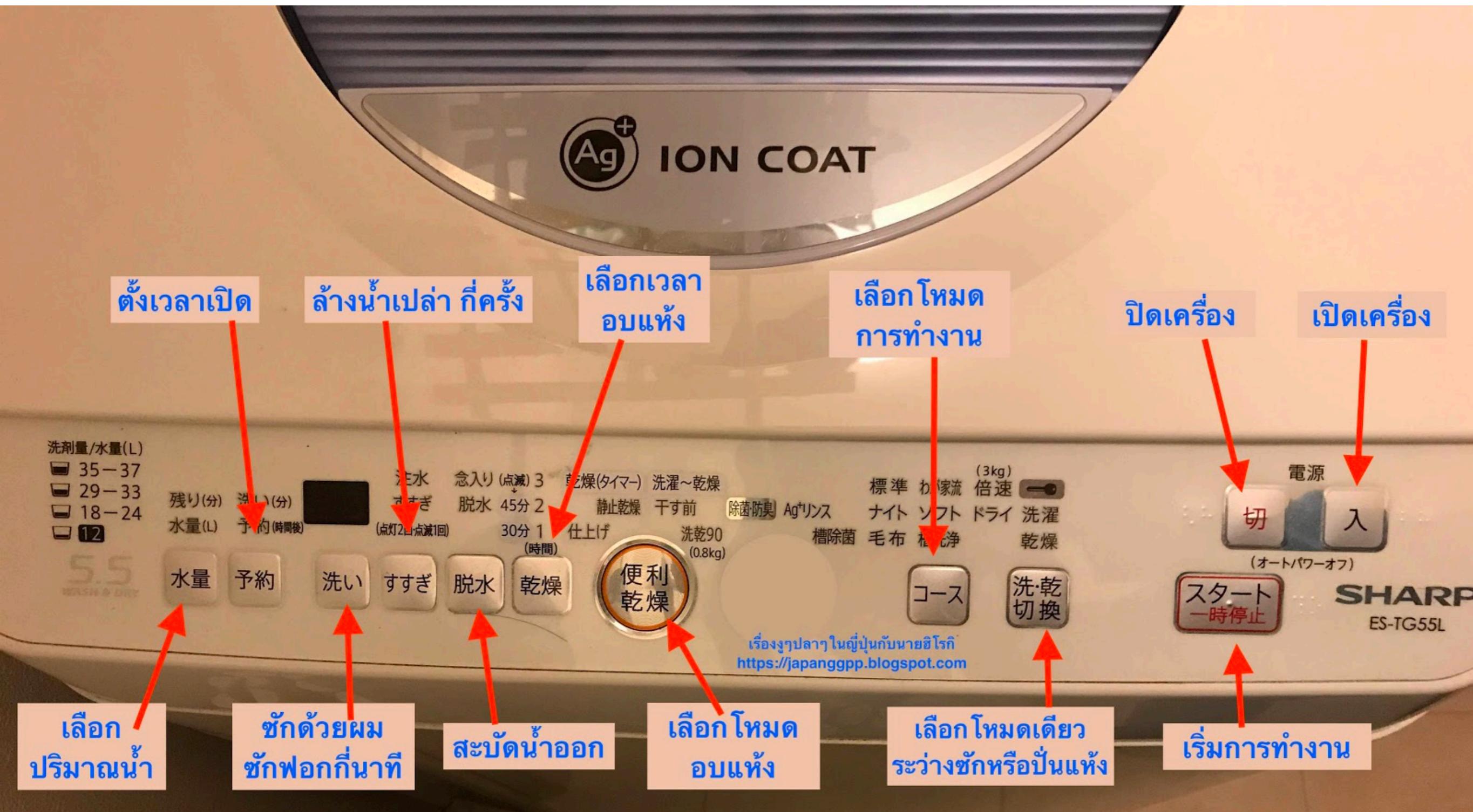
# Need API design



# Customer-First !!



# Customer-First !!



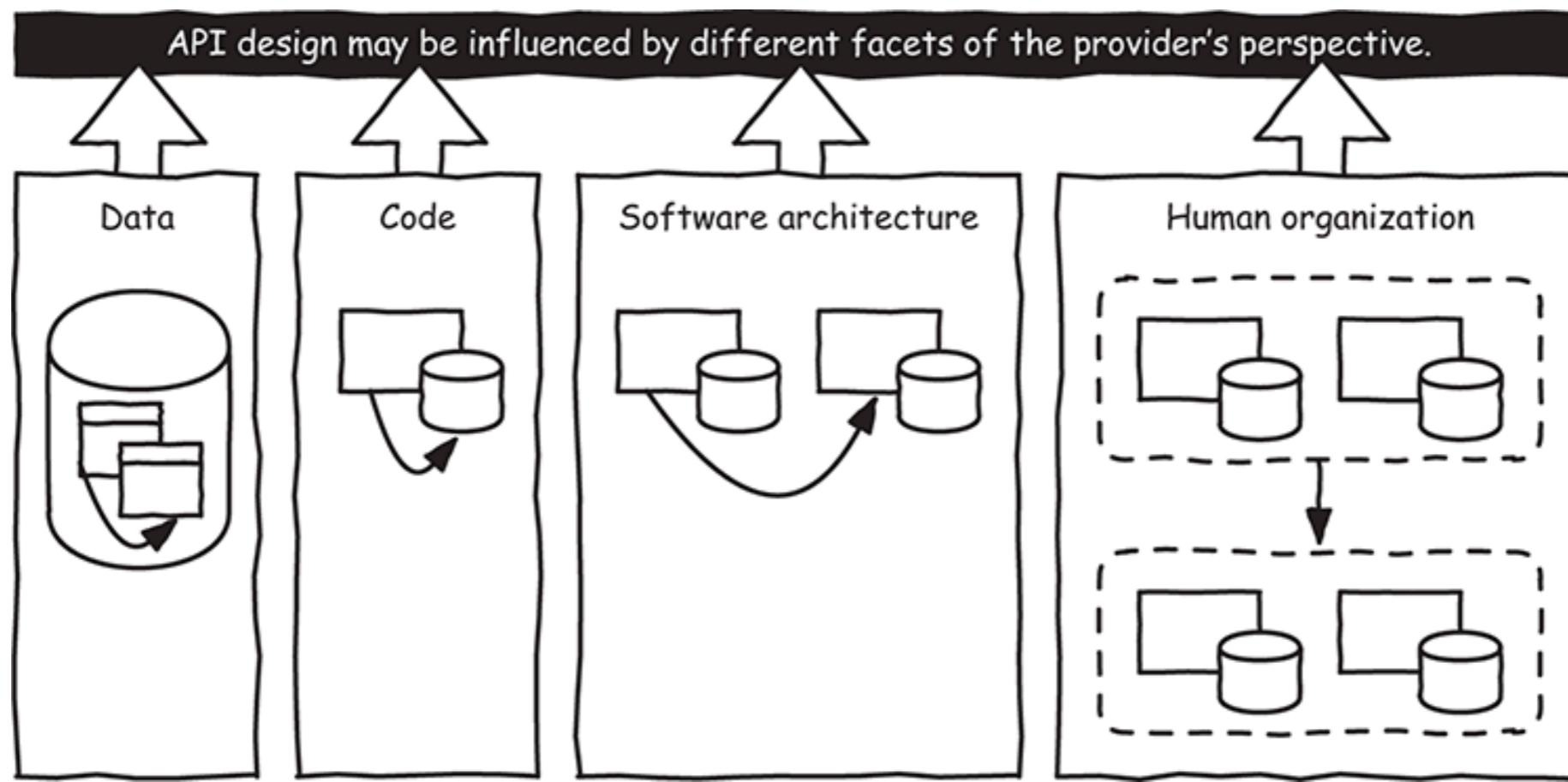
# Provider vs Consumer View



# How to design APIs ?

Identify API's goals

Avoid provider's view when designing



# Identify API's Goals



# Identify API's Goals

Who can use the API ?

What they can do ?

How they do it ?

What they need to do it ?

What they get in return ?



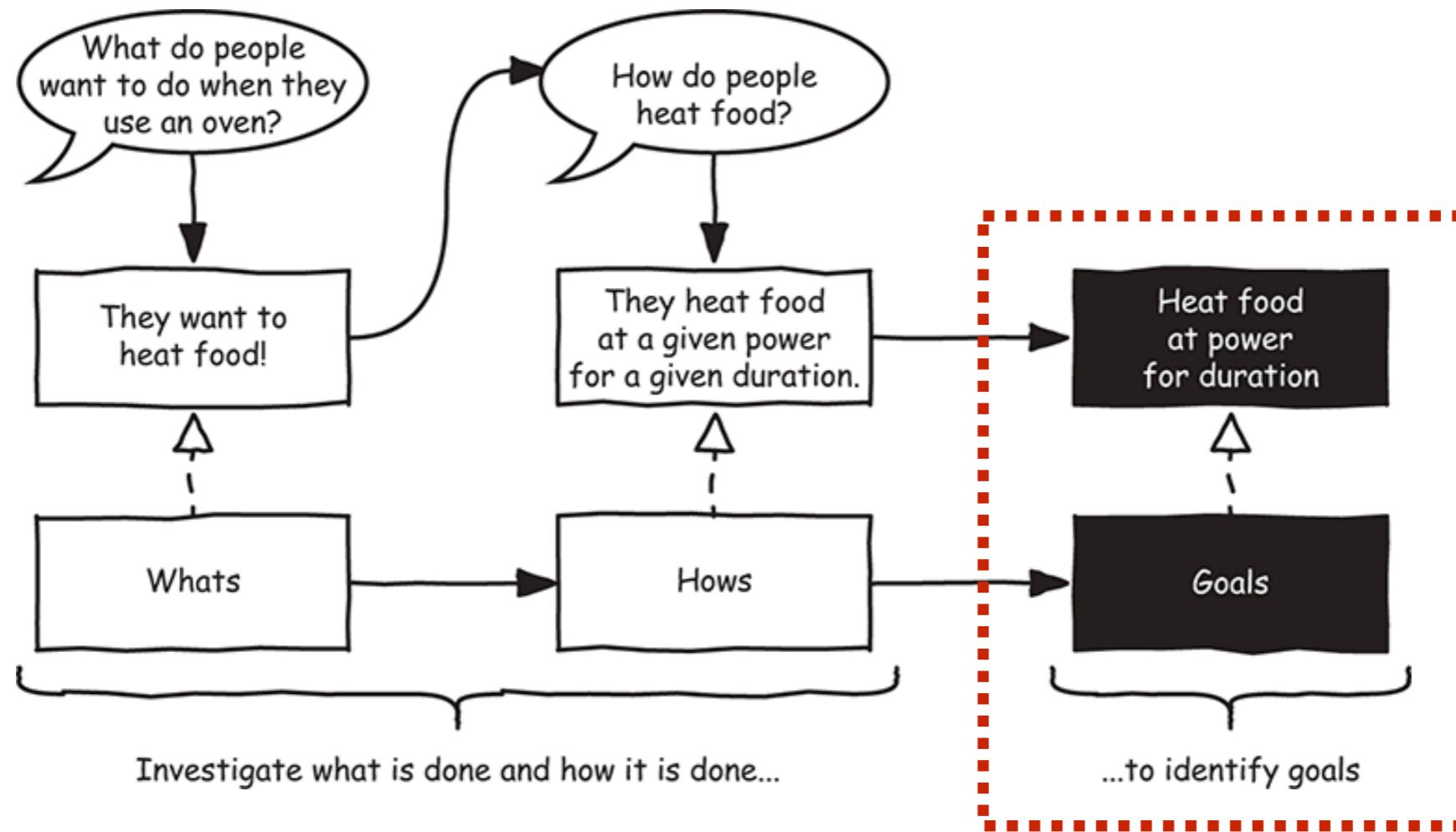
# Let's start



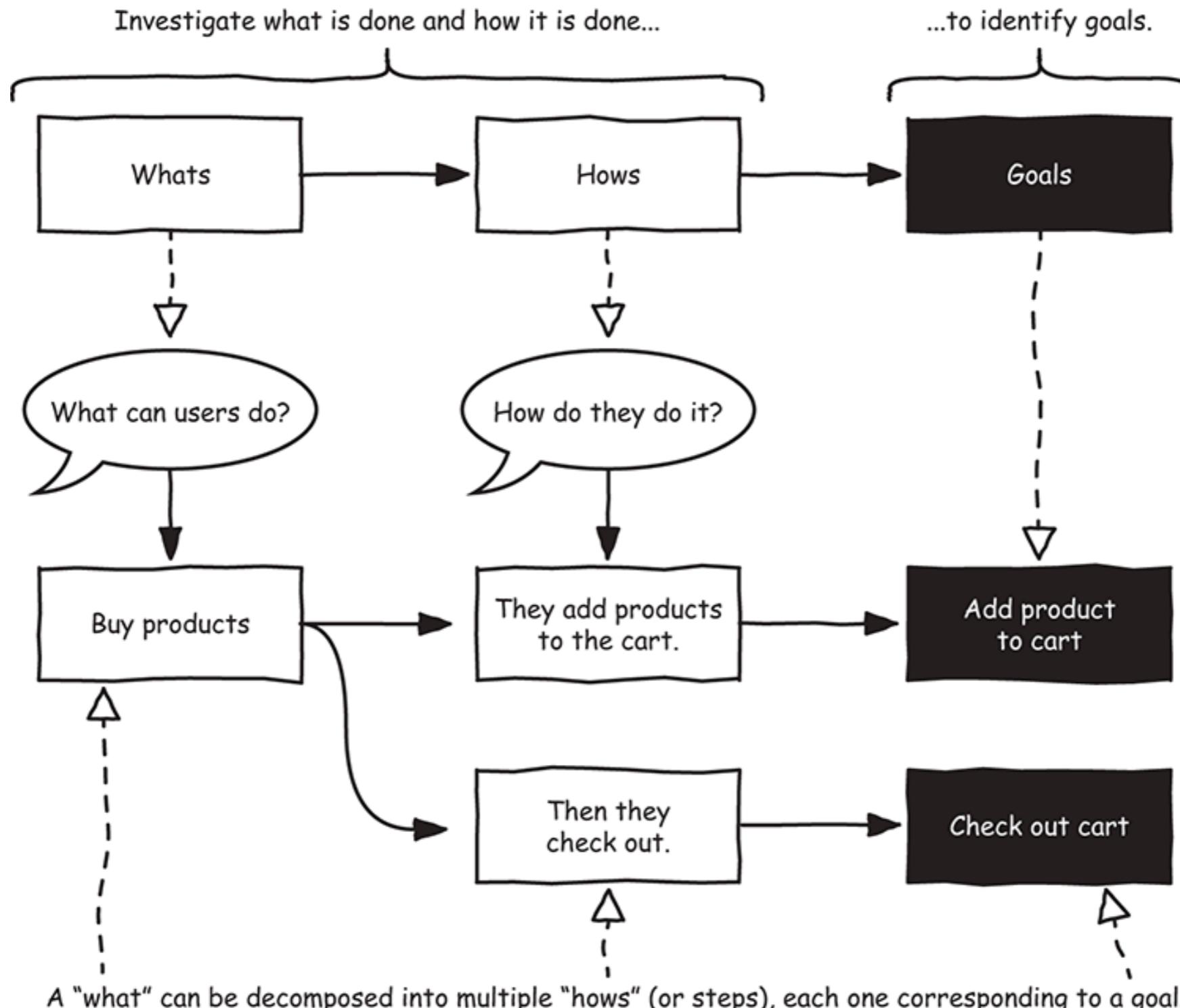
# 1. What and How ?

What can users do ?

How they do it ?



# User buy a product



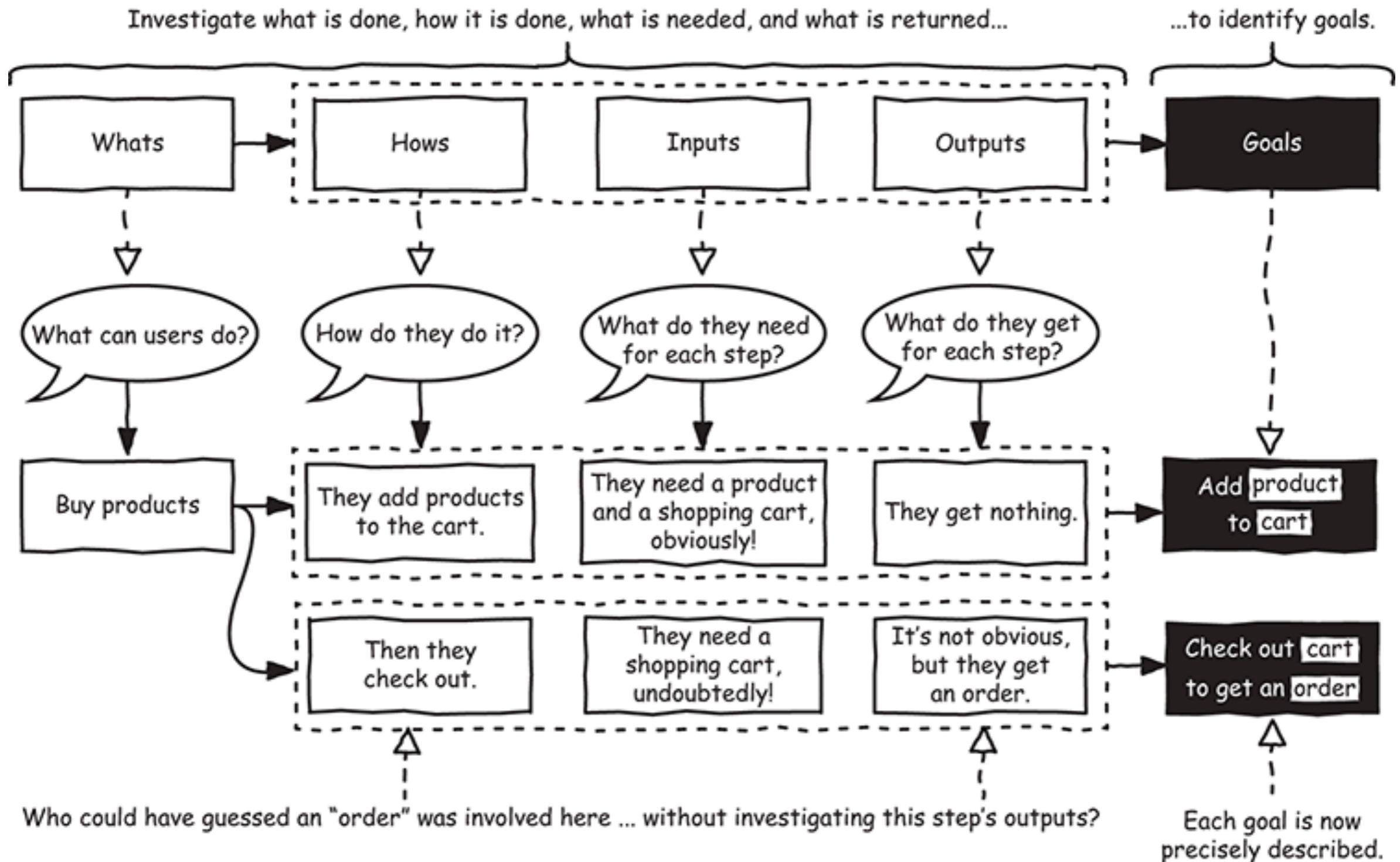
# 2. Input and Output ?

What they need to do it ?

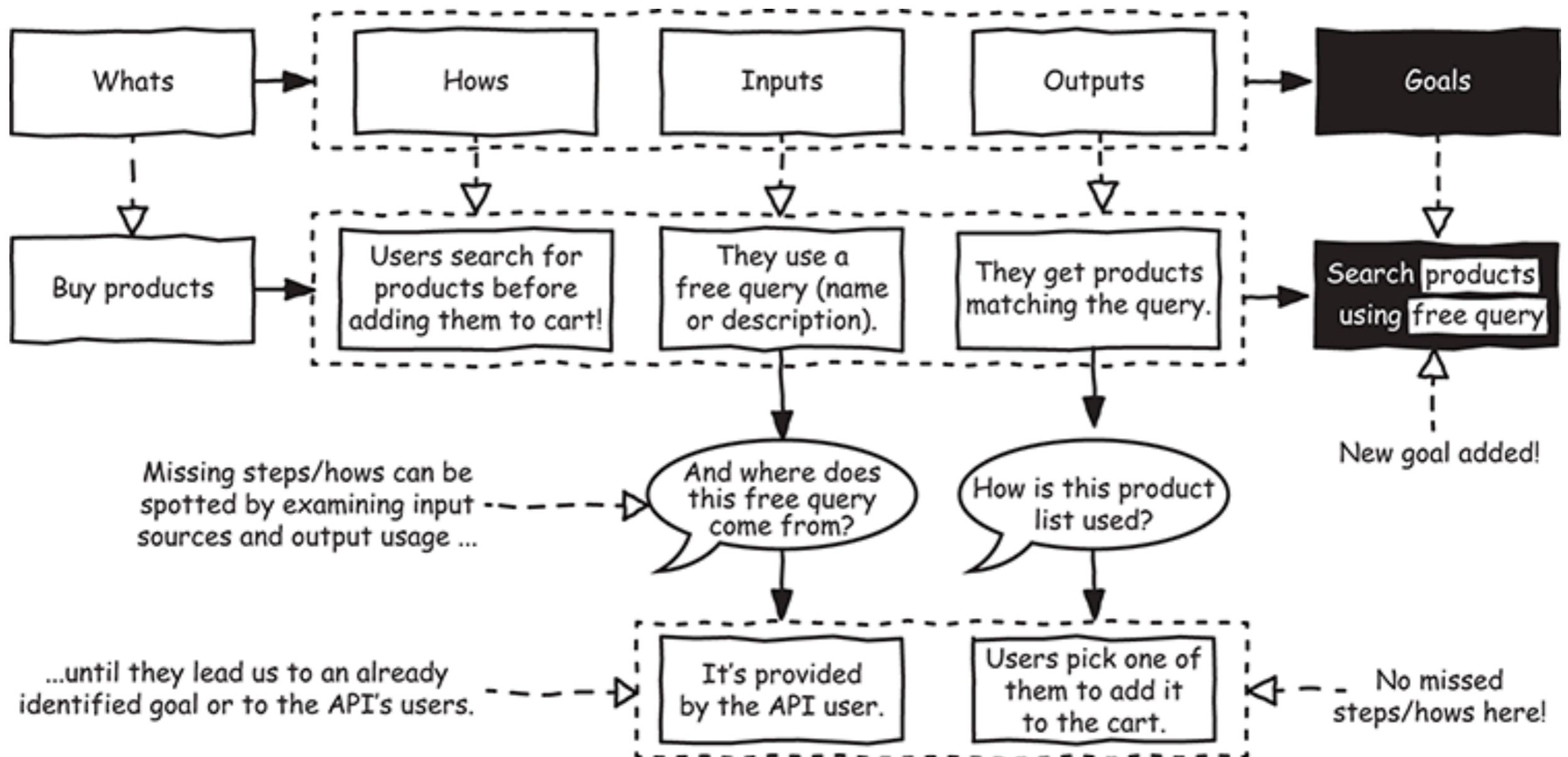
What they get in return ?



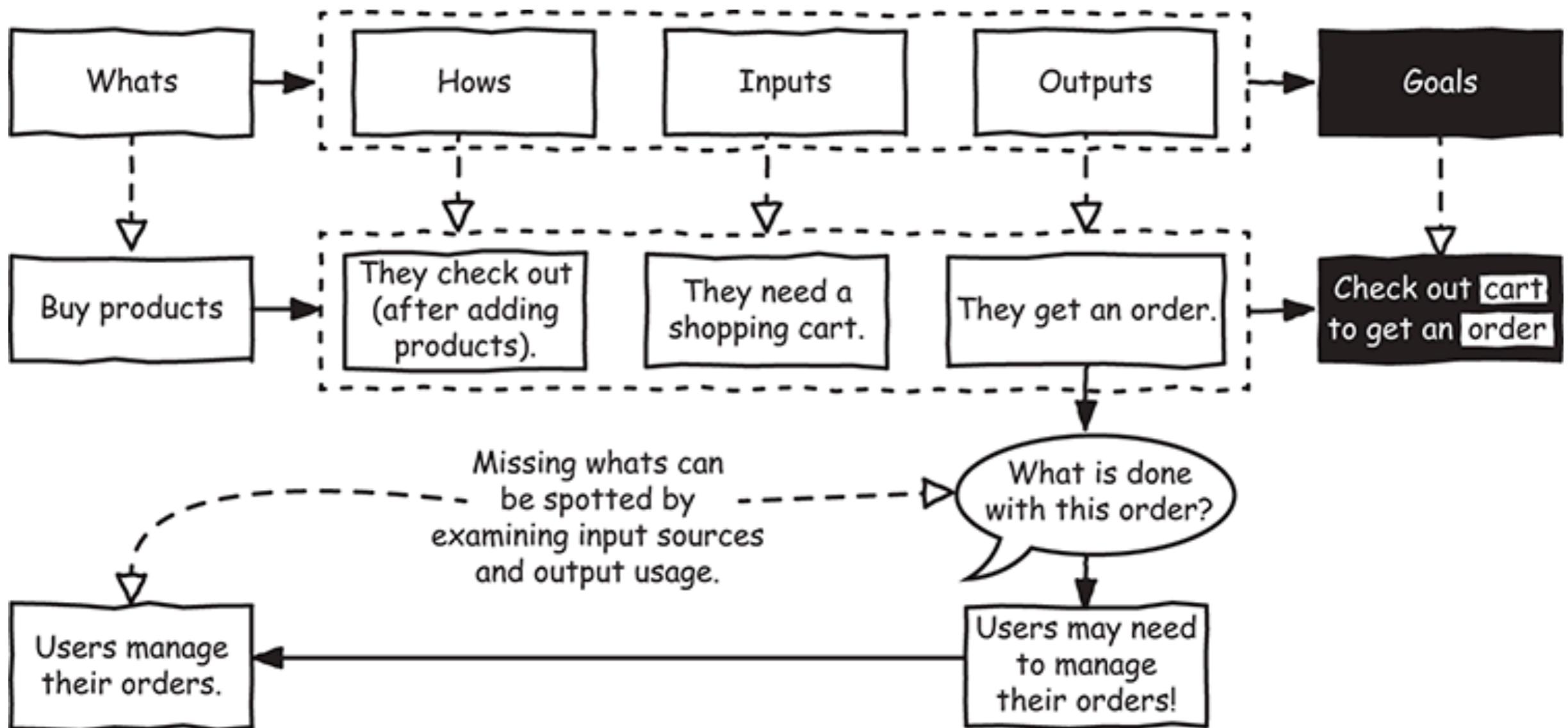
# 2. Input and Output ?



# 3. Find missing goals ?



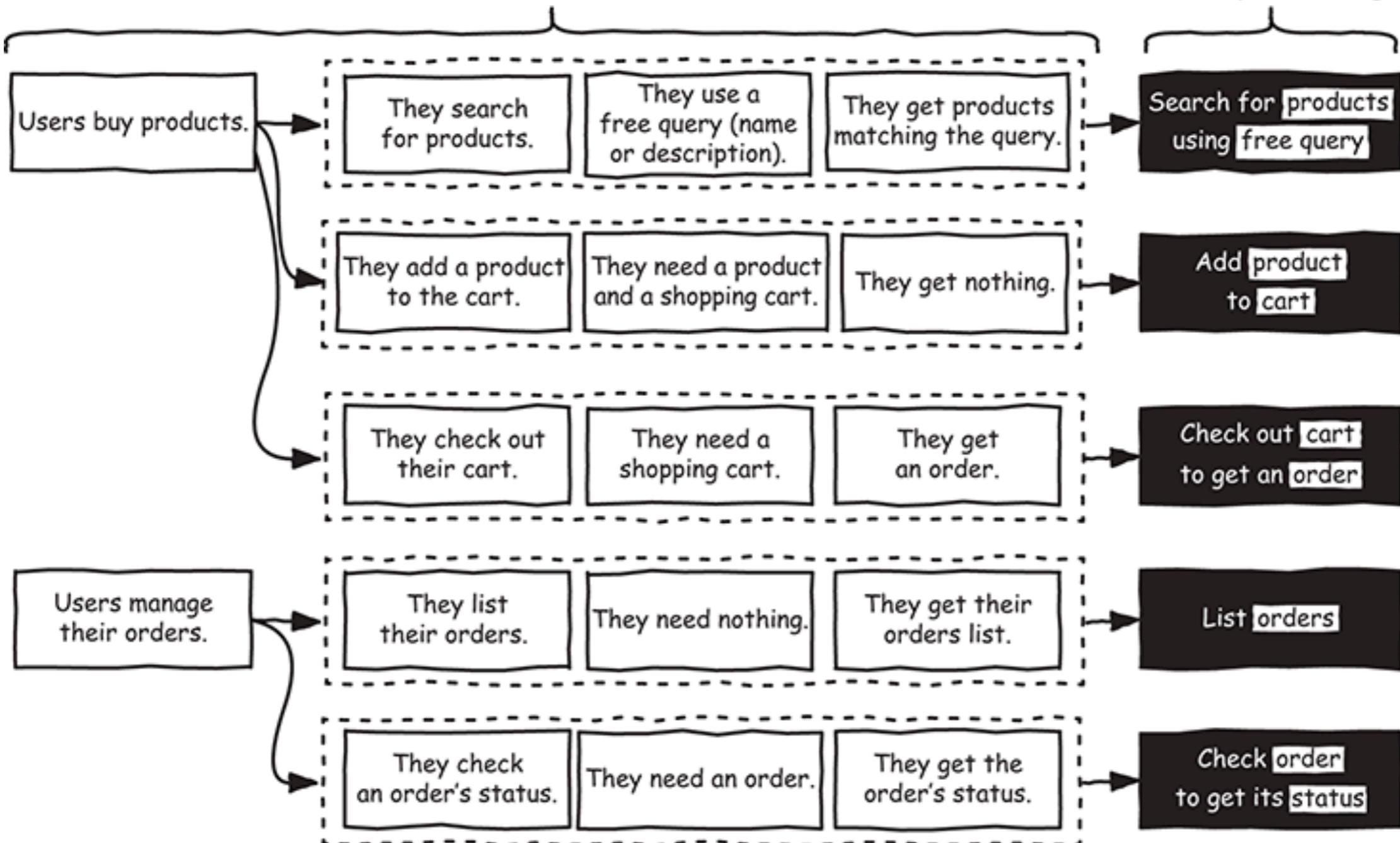
# 3. Find missing goals ?



# Add to Goals

Investigate what is done, how it is done, what is needed and where it comes from, what is returned and how it is used ...

...to identify accurately and exhaustively the API's goals.

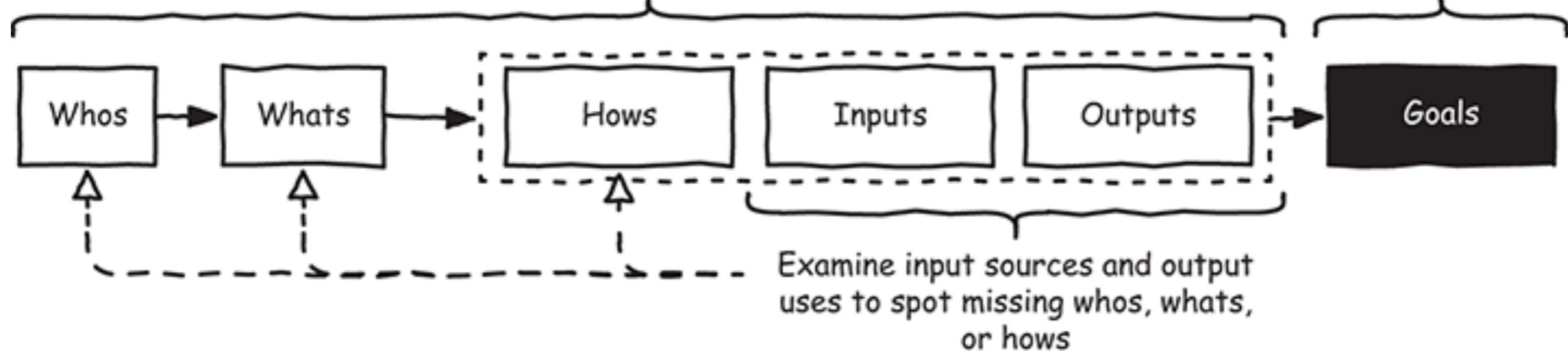


# 4. Identify all users (Who)

## Who are the users ?

Investigate who the users are, what they can do, how it is done, what is needed and where it comes from, what is returned and how it is used...

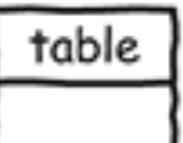
...to identify accurately and exhaustively the API's goals

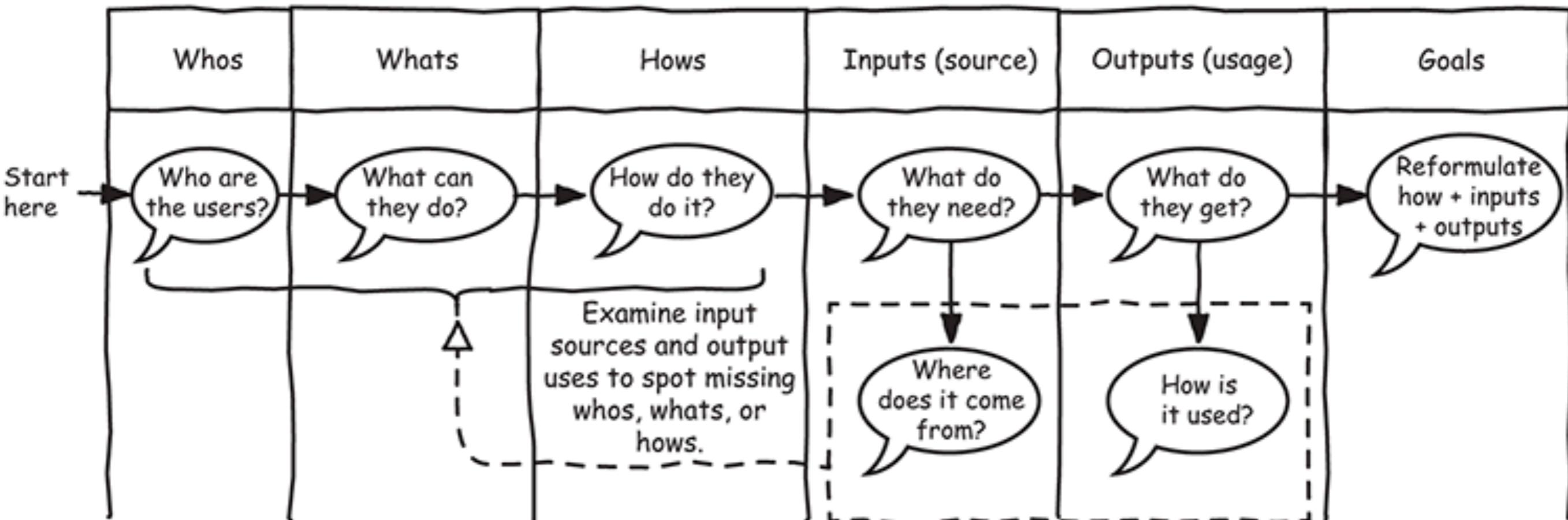


# API Goals Canvas



# API Goals Canvas

Draw this  on a whiteboard, flipchart, sheet of paper, or spreadsheet and start the questioning.



# Example

Whos	Whats	Hows	Inputs (source)	Outputs (usage)	Goals
Customers	Buy products	Search for products	Catalog (manage catalog), free query (provided by user)	Products (add product to cart)	Search for products in catalog using free query
		Add product to cart	Product (search for products), cart (owned by user)		Add product to cart
Admin	Manage catalog	Add product to catalog	Catalog (owned by user), product (provided by user)		Add product to catalog



# Workshop

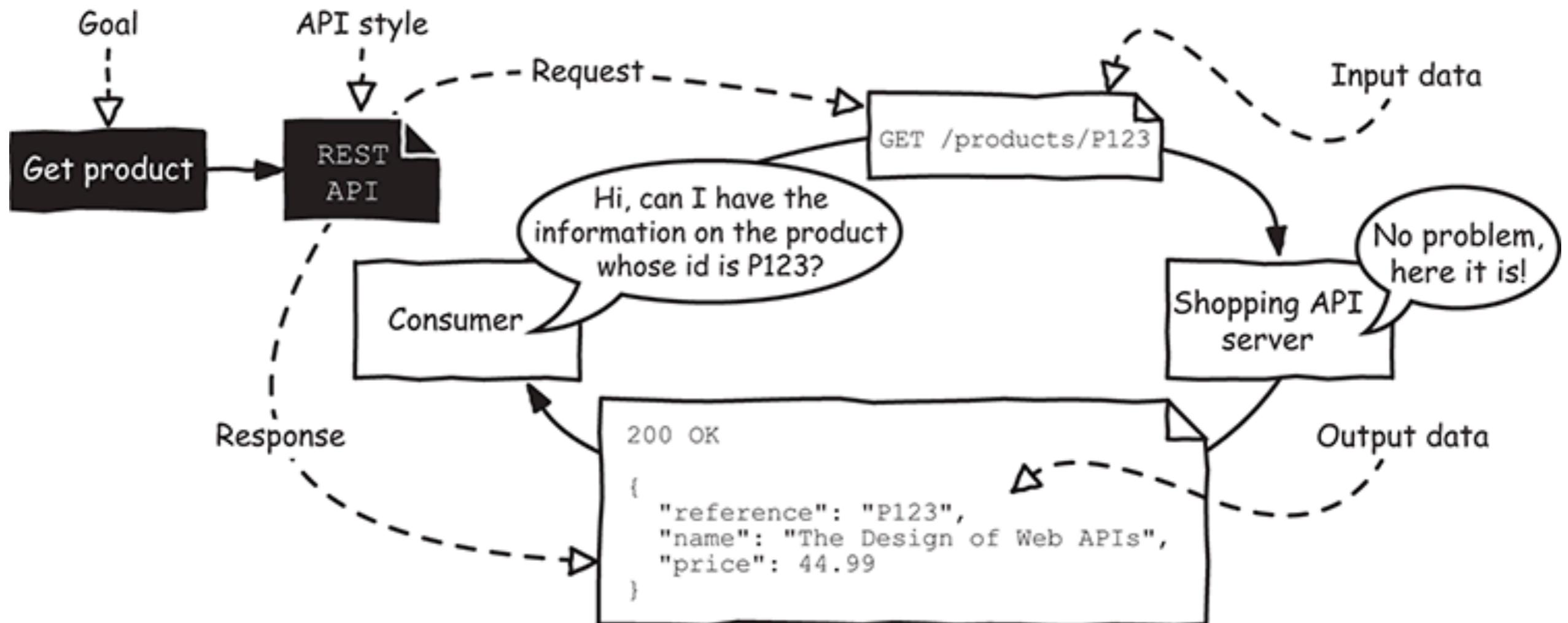


# **Next Step Design Programming Interface**



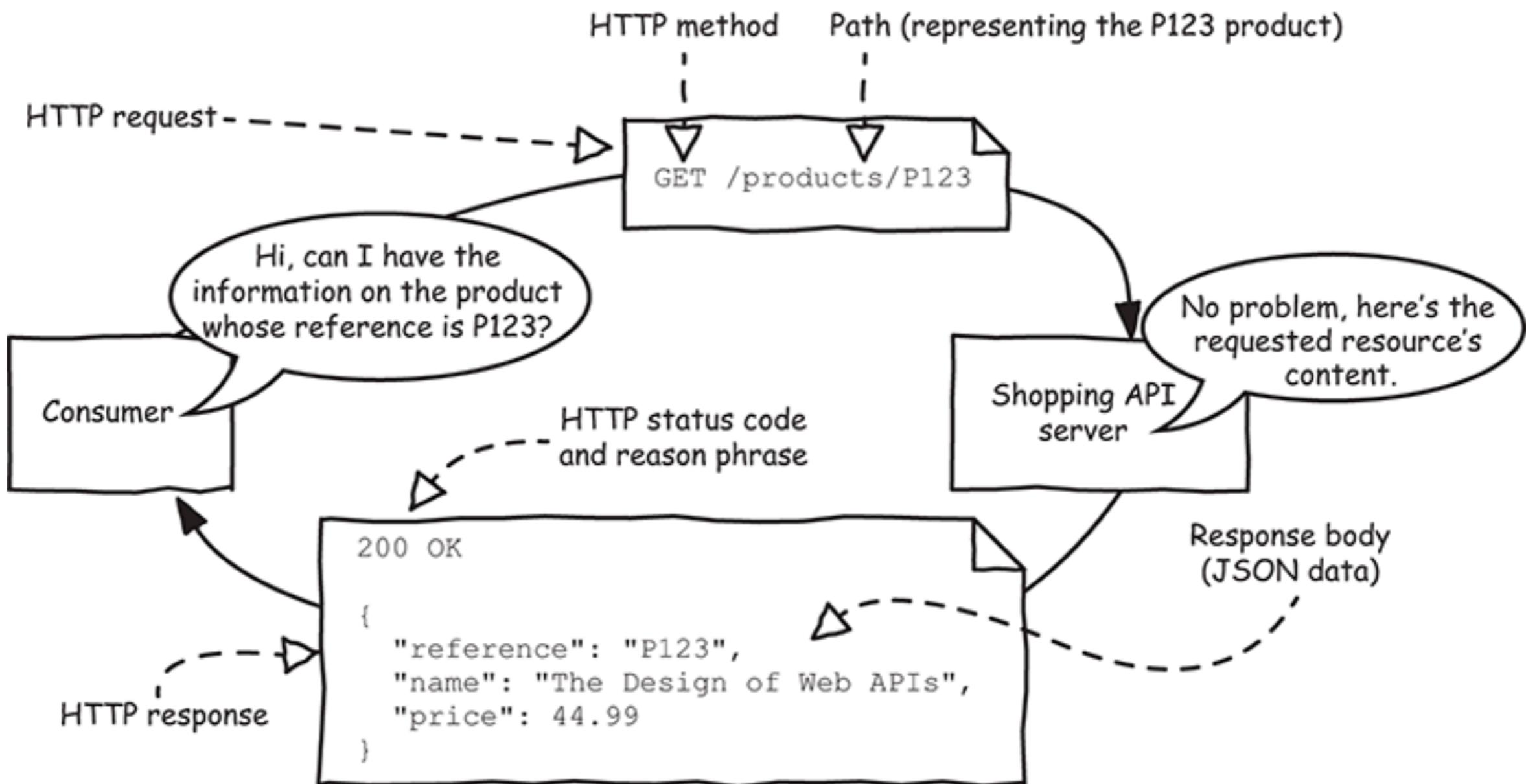
# Example

## Get product detail



# Example

## Get product detail



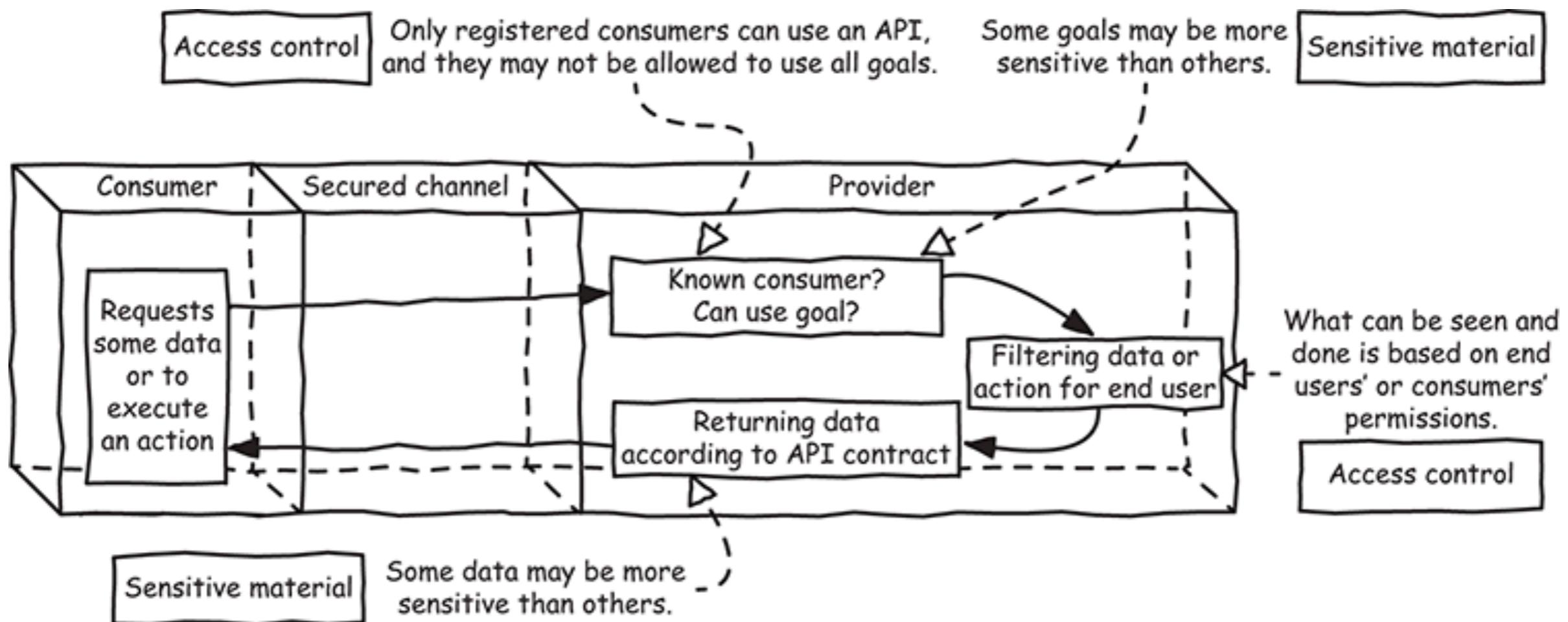
# API Security



# **Security is a critical element for any web application (APIs)**

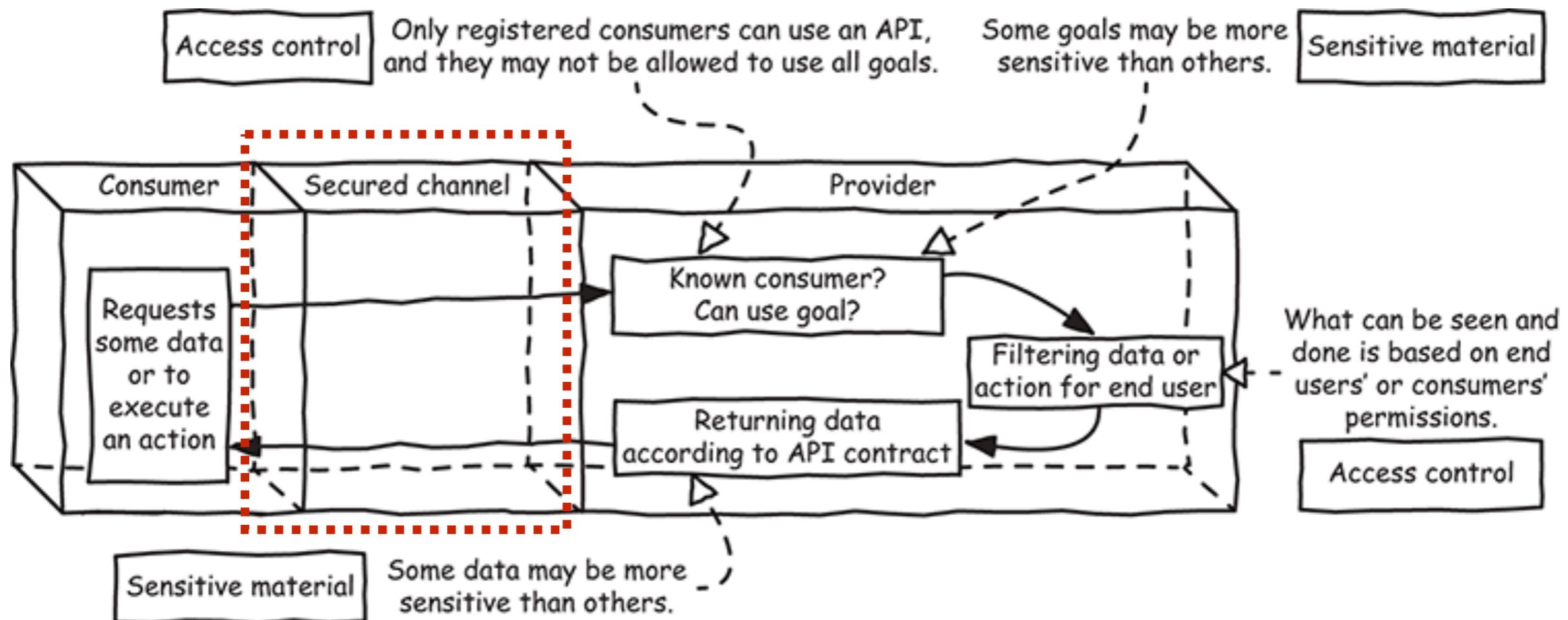


# API from security perspective



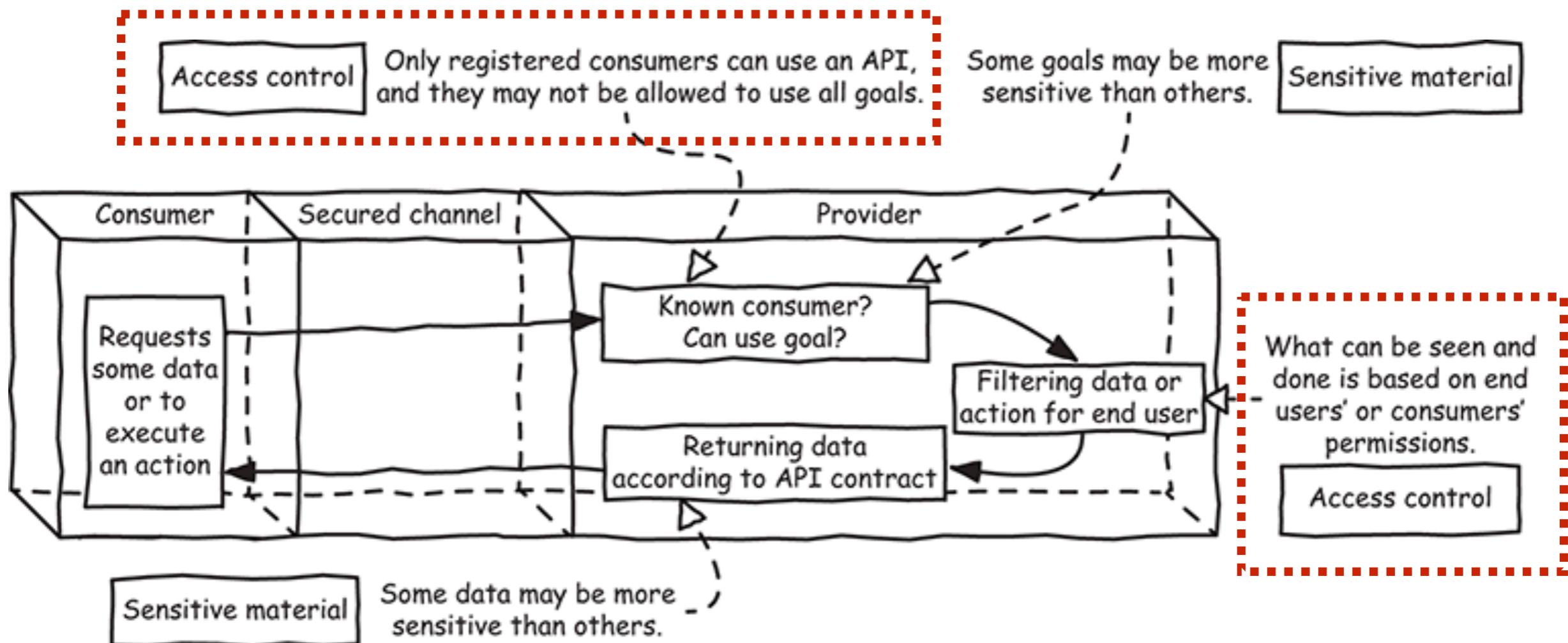
# API from security perspective

## Secure channel



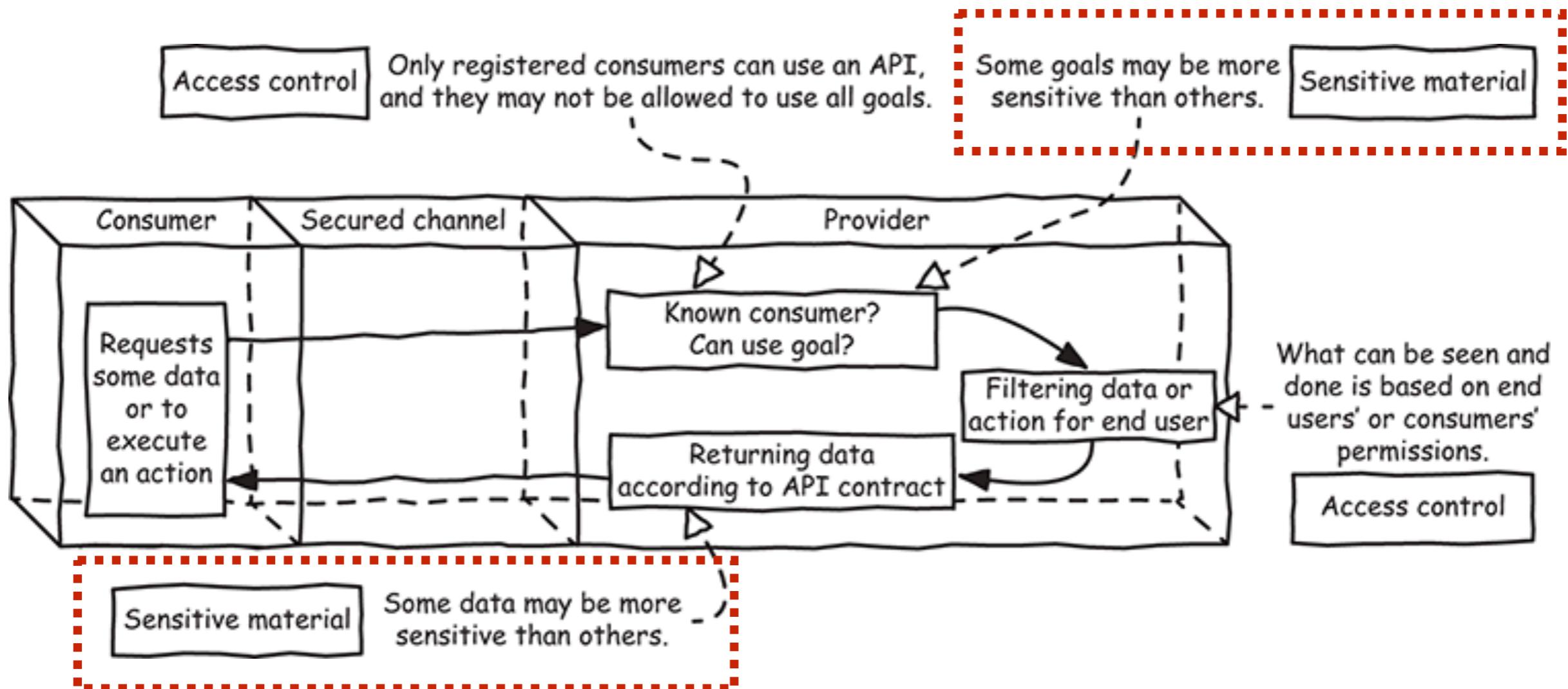
# API from security perspective

## Access control



# API from security perspective

## Sensitive data



# Security ?

Input and output validation

Using SSL (Secure Socket Layer)

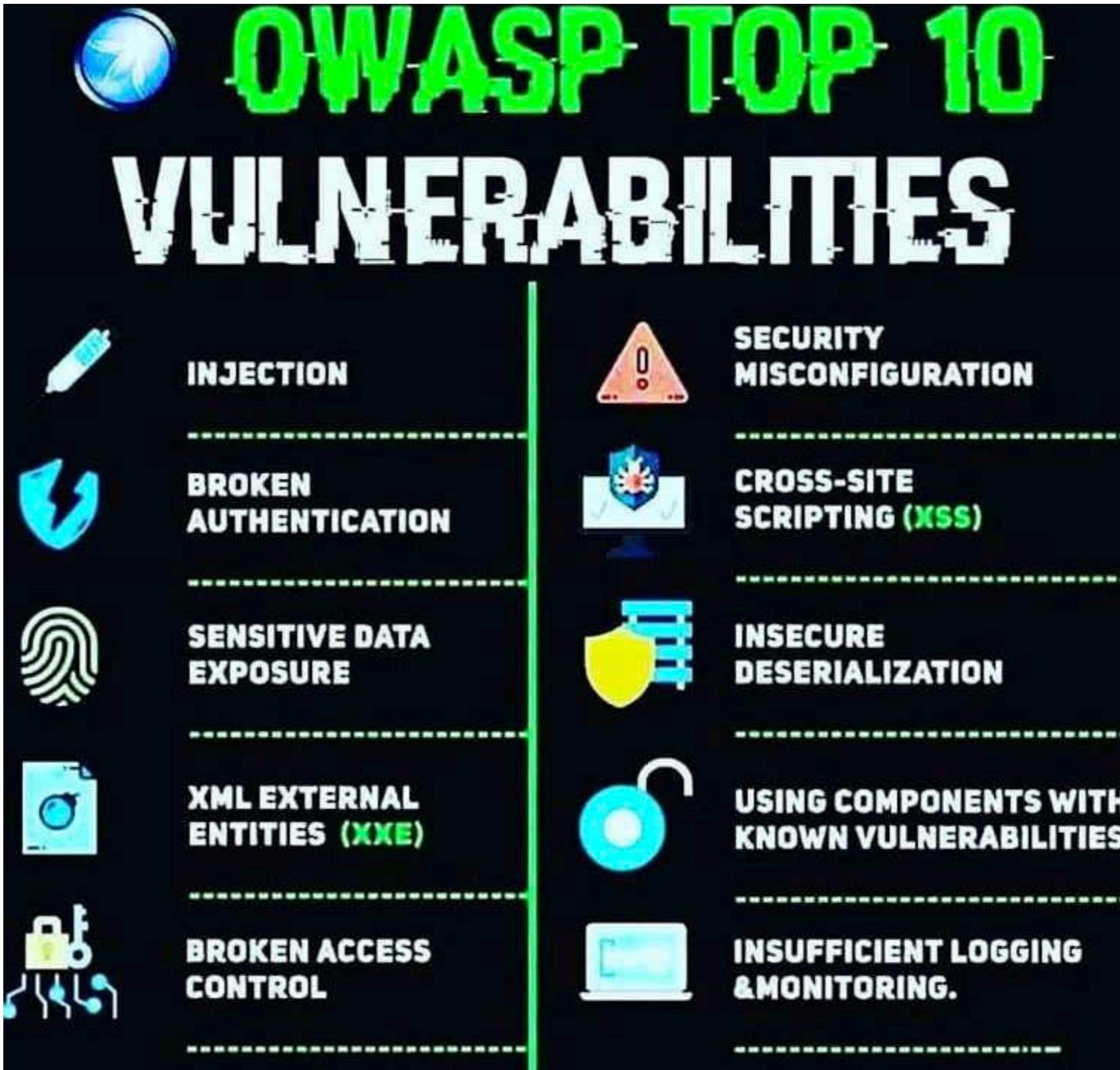
Validate content types

Maintain audit logs

Protect from CSRF (Cross-Site Request Forgery)

Protect from XSS (Cross-Site Scripting)





<https://owasp.org/www-project-top-ten/>



# Injection Game

Play SQL Injection

► Play SQL 1 / Basic SQL

Ø Play SQL 2 / Exploit Login

Ø Play SQL 3 / Blind Injection

Play Cross-Site Scripting

► Play XSS 1 / Basic XSS

Ø Play XSS 2 / Stored XSS

Ø Play XSS 3 / Invisible XSS

<https://injection.pythonanywhere.com/>



Web API Design

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

# XSS Game

**Warning: You are entering the XSS game area**

**Welcome, recruit!**

Cross-site scripting (XSS) bugs are one of the most common and dangerous types of vulnerabilities in Web applications. These nasty buggers can allow your enemies to steal or modify user data in your apps and you must learn to dispatch them, pronto!

At Google, we know very well how important these bugs are. In fact, Google is so serious about finding and fixing XSS issues that we are paying mercenaries up to \$7,500 for dangerous XSS bugs discovered in our most sensitive products.

In this training program, you will learn to find and exploit XSS bugs. You'll use this knowledge to confuse and infuriate your adversaries by preventing such bugs from happening in your applications.

There will be cake at the end of the test.

**Let me at 'em!**

?

<https://xss-game.appspot.com/>



# Authentication & Authorization



# Authentication patterns

Basic Auth  
Digest  
API Keys  
OAuth

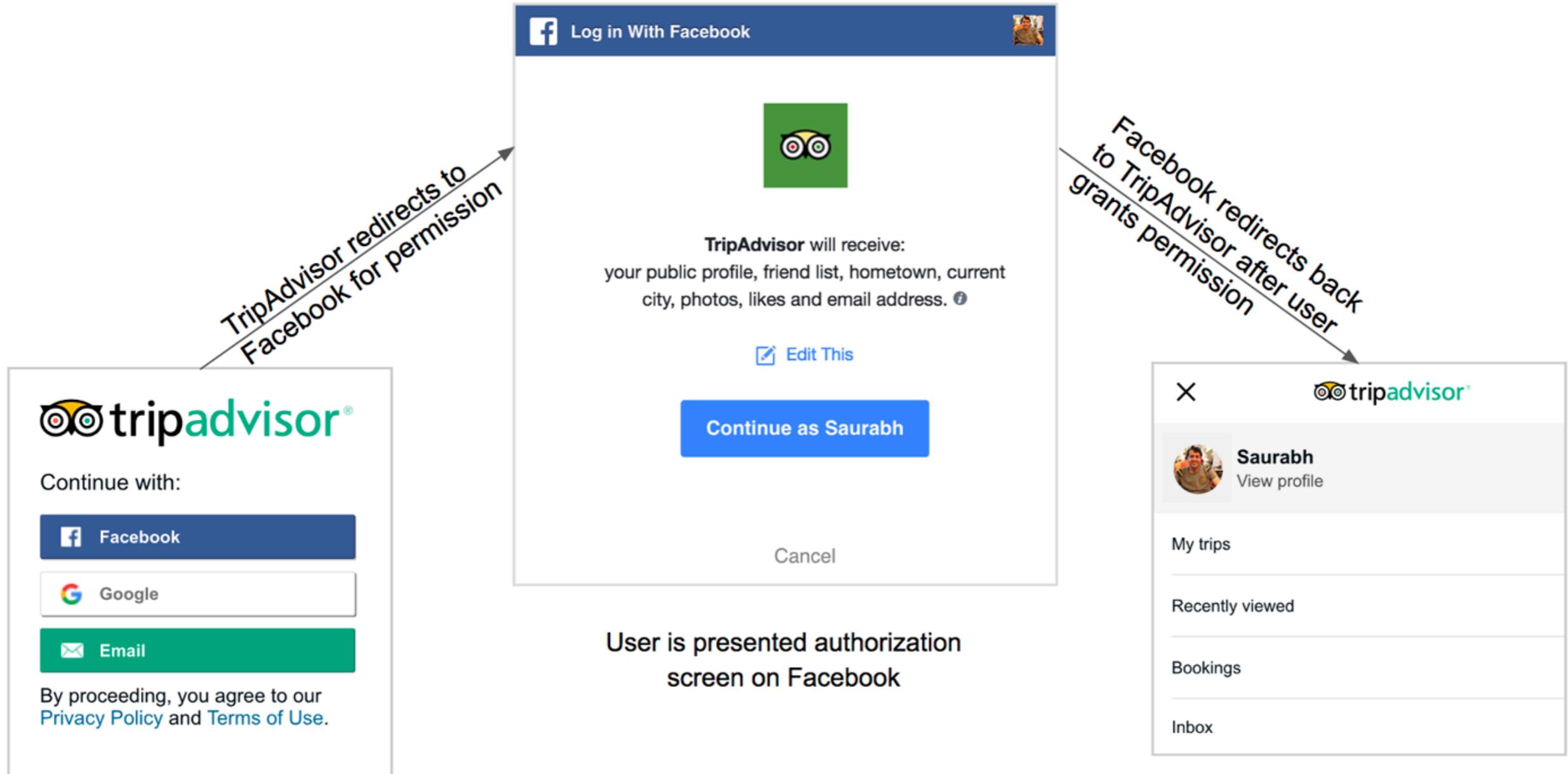


# OAuth

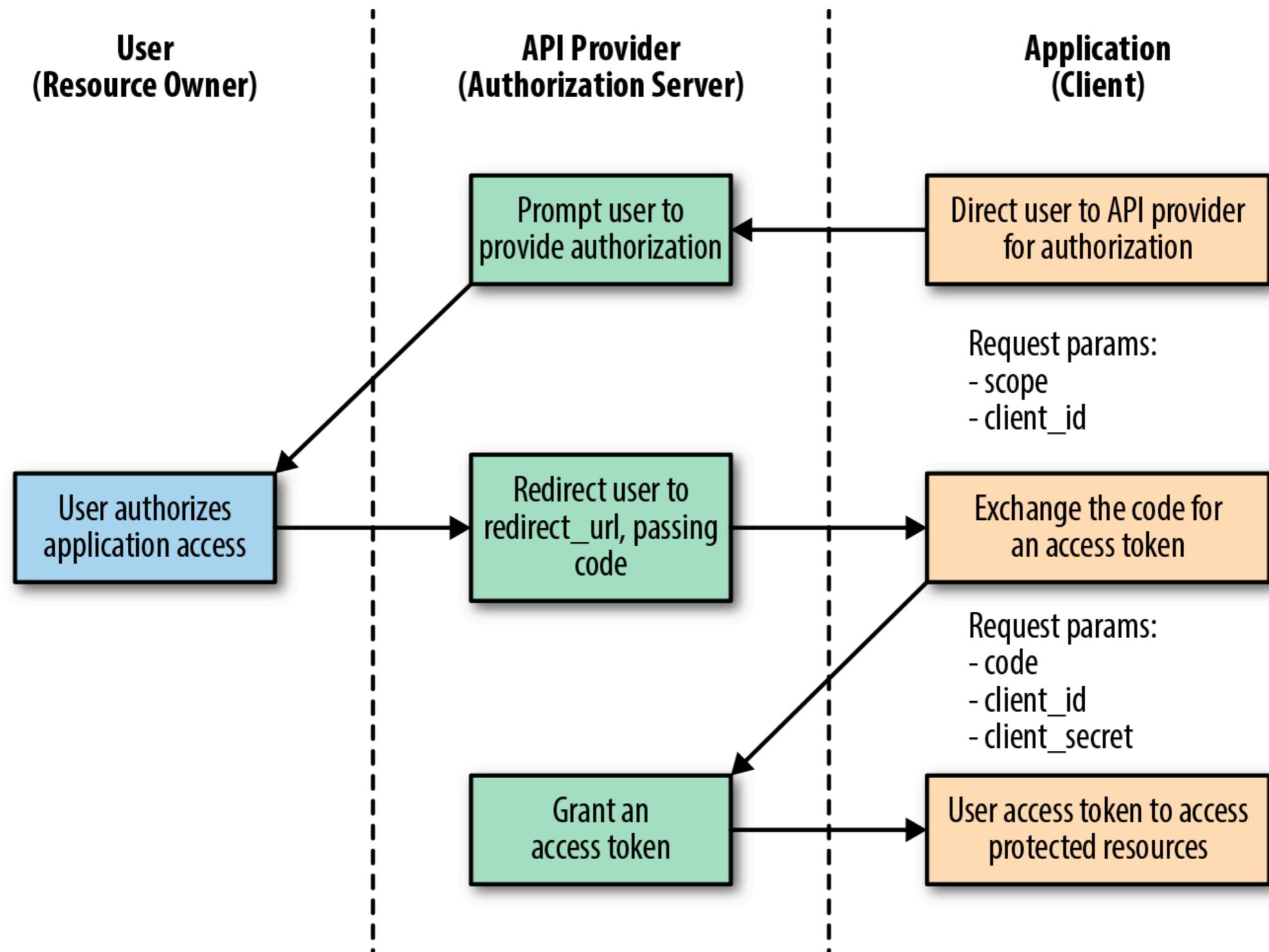
Open standard that allows users to grant access  
to applications without sharing passwords



# Example OAuth



# Flow OAuth



# Workshop



# Online Flight Booking (1)

The screenshot shows the AirAsia homepage with a search bar and various travel service links. The search bar displays the flight details: From Bangkok - Don Mueang (DMK) to Ubon Ratchathani (UBP), Depart date 15/12/2020, and Return date 18/12/2020. A red 'Search' button is visible. Below the search bar, there are filters for 1 Adult, Multi-city, and a Promo code input field. A checkbox for 'Redeem BIG Points' is also present. A promotional banner for Koh Samui travel is displayed at the bottom.

From Bangkok - Don Mueang (DMK) To Ubon Ratchathani (UBP)

Depart date 15/12/2020

Return date 18/12/2020

Search

1 Adult

Multi-city

Promo code

Redeem BIG Points

Koh Samui\*

\*via Surat Thani / Nakhon Si Thammarat

Book from 7 Dec 2020 - 12 Dec 2020

Travel from 14 Dec 2020 - 30 Apr 2021

Terms & Conditions apply.

<https://www.airasia.com/en/gb>



# Online Flight Booking (2)

The screenshot shows the AirAsia flight booking interface. At the top, there are five navigation steps: Search flights, Select flights, Add-ons, Guest details, and Payment. A yellow banner at the top states: "Please note that you are about to book a flight that operates in Don Mueang International Airport (DMK). Please ensure that your t... [Read More](#)". Below this, the departure location is set to "Bangkok - Don Mueang to Ubon Ratchathani". The date selector shows flights from Tuesday, 08 Dec (Unavailable) to Monday, 14 Dec. The fare for Wednesday, 09 Dec, is highlighted in red as "From 1,786.08 THB". The results table lists three flights:

Flight Details	Fare
08:30 -> 09:50, 1h 20m (i), Direct, Low Fare, 2,116.08 THB	2,116.08 THB
11:00 -> 12:15, 1h 15m (i), Direct, 1,786.08 THB (selected)	1,786.08 THB
17:25 -> 19:05, Low Fare	Low Fare

At the bottom, there are options to "Redeem BIG Points" or "Log in to redeem/earn points". The total amount is listed as "Total THB 3,357.16" with a shopping cart icon. A red "Continue" button is at the bottom right.

<https://www.airasia.com/en/gb>



# Example

**Name ::** Get product detail by id

**GET** products/<id>

**Request**

Body = JSON ?

**Response**

Code = ?

Body = JSON ?



# Using APIs

## Make requests with Postman



# Open Weather

## OpenWeather global services

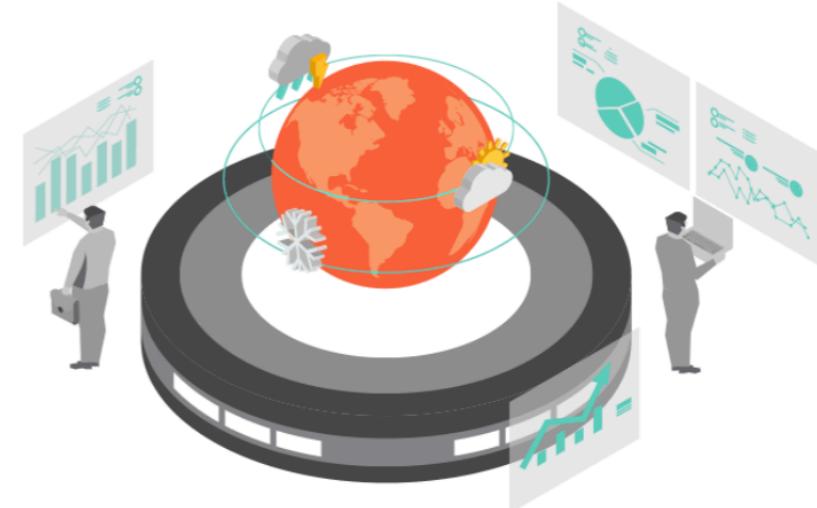
Weather forecasts, nowcasts and history in fast and elegant way

2 Billion Forecasts Per Day

2,500 new subscribers a day

2,600,000 customers

20+ weather APIs



Search city

Search



Different Weather?

Metric: °C, m/s   Imperial: °F, mph

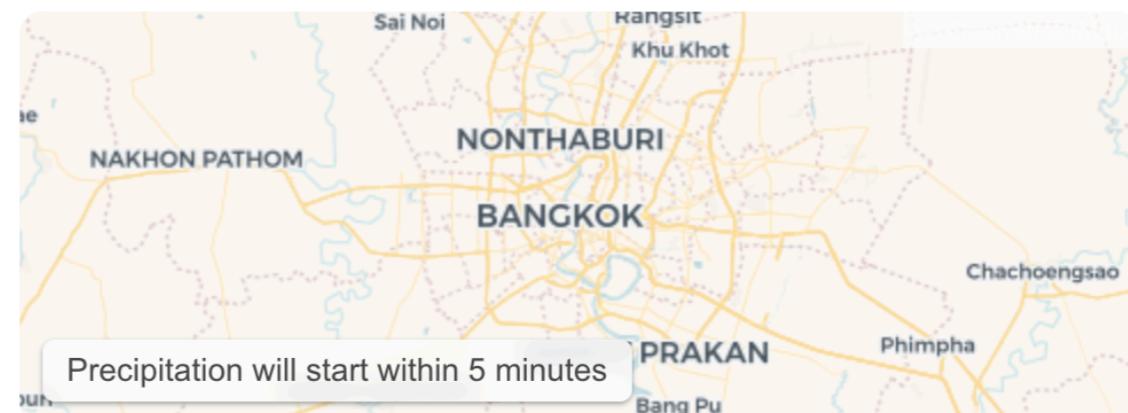
11:37pm, Dec 9  
**Bangkok, TH**

● 26°C

Feels like 27°C. Clear sky. Light breeze

▲ 3.1m/s ENE   ☘ 1014hPa   Humidity: 69%

Dew point: 20°C   Visibility: 10.0km



Minute forecast

<https://openweathermap.org>



Web API Design

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Open Weather API

## Weather API

[Home](#) / Weather API

Please [sign up](#) and use our fast and easy-to-work weather APIs for free. Look at our [monthly subscriptions](#) for more options rather than the Free account that we provide you. Read [How to start](#) first and enjoy using our powerful weather APIs.

## Current & Forecast weather data collection

### Current Weather Data

[API doc](#) [Subscribe](#)

- Access current weather data for any location including over 200,000 cities
- We collect and process weather data from different sources such as global and local weather models, satellites, radars and vast network of weather stations
- JSON, XML, and HTML formats
- Available for both Free and paid subscriptions

### Hourly Forecast 4 days

[API doc](#) [Subscribe](#)

- Hourly forecast is available for 4 days
- Forecast weather data for 96 timestamps
- Higher geographic accuracy
- JSON and XML formats
- Available for Developer, Professional and Enterprise accounts

### One Call API NEW

[API doc](#) [Subscribe](#)

- Make one API call and get current, forecast and historical weather data
- **Minute forecast** for 1 hour
- **Hourly forecast** for 48 hours
- **Daily forecast** for 7 days
- **Historical data** for 5 previous days
- **Government weather alerts**
- JSON format
- Available for both Free and paid subscriptions

<https://openweathermap.org>



Web API Design

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Sign up and Sign in



# Current Weather Data API

## Current weather data

[Home](#) / [API](#) / Current weather

Access current weather data for any location on Earth including over 200,000 cities! We collect and process weather data from different sources such as global and local weather models, satellites, radars and vast network of weather stations. Data is available in JSON, XML, or HTML format.

### Call current weather data for one location

#### By city name

You can call by city name or city name, state code and country code. Please note that searching by states available only for the USA locations.

##### API call

```
api.openweathermap.org/data/2.5/weather?q={city name}&appid=  
{API key}
```

- [Call current weather data for one location](#)
- [By city name](#)
- [By city ID](#)
- [By geographic coordinates](#)
- [By ZIP code](#)
- [Call current weather data for several cities](#)
- [Cities within a rectangle zone](#)
- [Cities in circle](#)
- [Call for several city IDs](#)
- [Bulk downloading](#)
- [Weather fields in API response](#)
- [JSON](#)
- [XML](#)
- [List of condition codes](#)
- [Min/max temperature in current weather](#)
- [API and forecast API](#)
- [Other features](#)
- [Format](#)
- [Units of measurement](#)
- [Multilingual support](#)
- [Call back function for JavaScript code](#)

<https://openweathermap.org/current>



# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Make requests with Postman

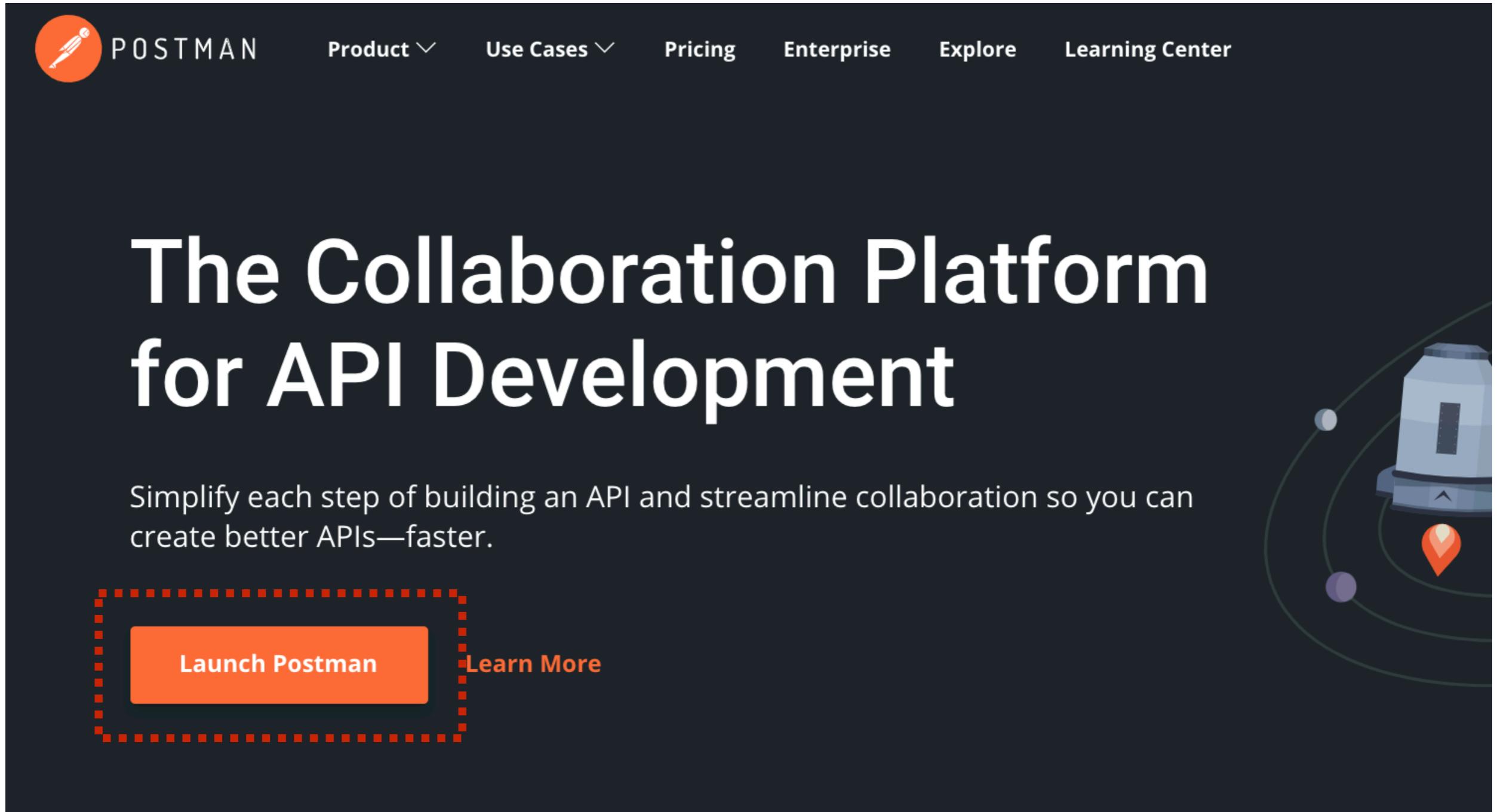
<https://www.postman.com/>



# Sign up and Sign in



# Postman

The screenshot shows the Postman homepage. At the top, there's a navigation bar with the Postman logo (an orange circle with a white pen icon), followed by the word "POSTMAN" and several menu items: "Product ▾", "Use Cases ▾", "Pricing", "Enterprise", "Explore", and "Learning Center". The main title "The Collaboration Platform for API Development" is prominently displayed in large white font. Below it, a subtitle reads: "Simplify each step of building an API and streamline collaboration so you can create better APIs—faster." To the right of the text, there's a graphic of a rocket launching from Earth, with orbital paths and a location pin. At the bottom left, there's a call-to-action button with the text "Launch Postman" and a "Learn More" link.

<https://www.postman.com/>



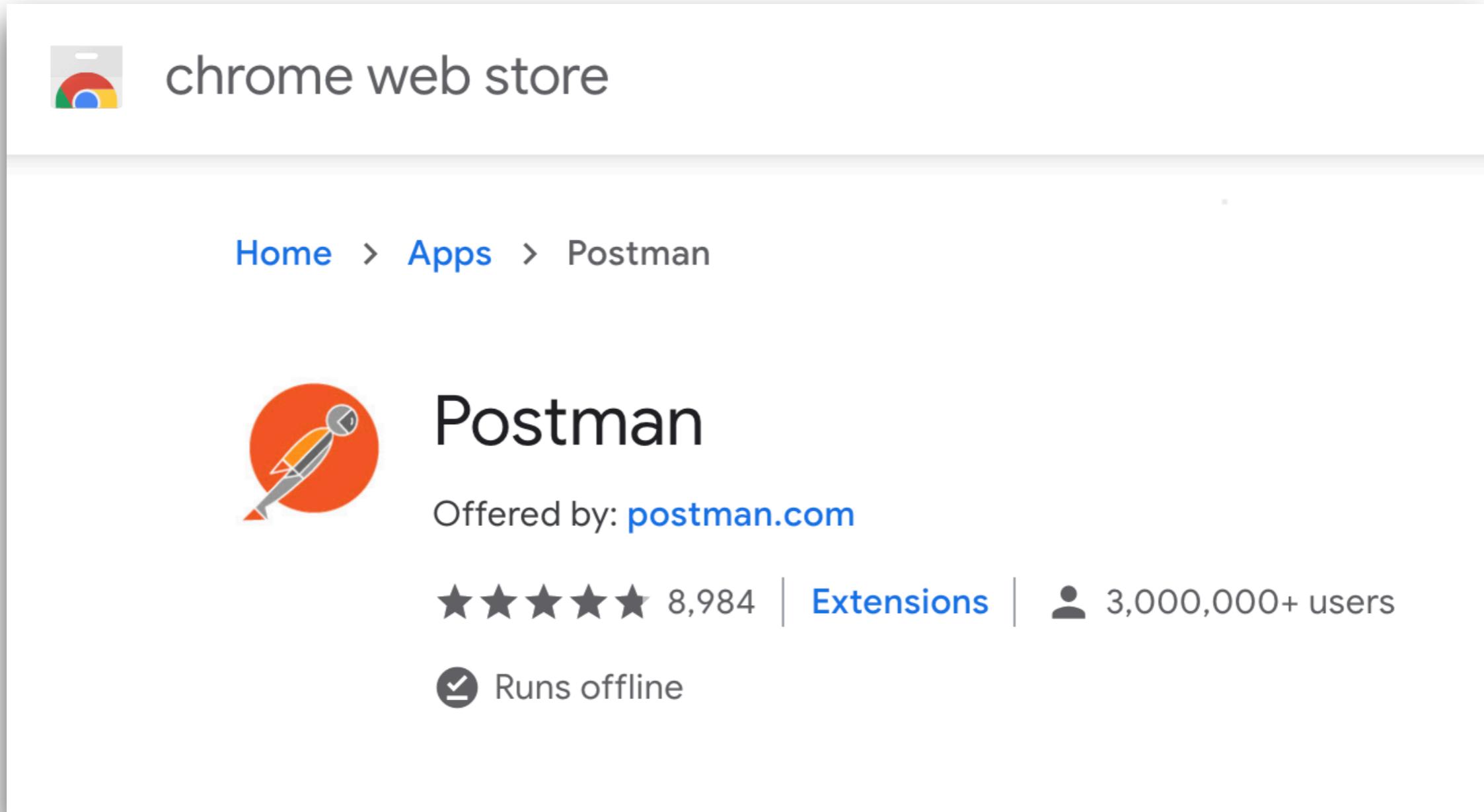
# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Postman on Chrome



The screenshot shows the Postman extension page on the Chrome Web Store. At the top left is the Chrome Web Store logo. Below it, the text "chrome web store" is displayed. A navigation bar shows the path: "Home > Apps > Postman". The main content features the Postman logo (an orange circle with a white stylized pen icon) and the word "Postman". Below this, it says "Offered by: [postman.com](#)". The extension has a rating of "★★★★★ 8,984" and links to "Extensions" and "3,000,000+ users". A note indicates that it "Runs offline".

<https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjb dggehcdcbncddomop?hl=en>



# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



Web API Design

© 2017 - 2018 Siam Chamnankit Company Limited. All rights reserved.

# Create new request to call API

GET Untitled Request x + ooo No Environment v

Untitled Request Save v edit comment

GET v Enter request URL Send v

Params Authorization Headers Body Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION	ooo	Bulk Edit
	Key	Value	Description		

Response



# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Ready to Go !!

The screenshot shows the Postman application interface. At the top, there's a header bar with the project name "api.openweathermap" and environment dropdowns. Below the header, a search bar contains the URL "api.openweathermap.org/data/2.5/weather?q=bangkok&appid=". The method is set to "GET". To the right of the URL are "Params" and a "Send" button. Below the search bar, tabs for "Authorization", "Headers", "Body", "Pre-request Script", and "Tests" are visible, with "Authorization" being the active tab. A dropdown menu under "Type" shows "No Auth". In the main body area, tabs for "Body", "Cookies", "Headers (9)", and "Test Results" are present, with "Body" being the active tab. The status is shown as "Status: 401 Unauthorized". Below the tabs, there are buttons for "Pretty", "Raw", "Preview", and "JSON". The JSON response is displayed in a code editor-like format:

```
1 {  
2   "cod": 401,  
3   "message": "Invalid API key. Please see http://openweathermap.org/faq#error401 for more info."  
4 }
```



# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Create new request to call API

## HTTP request

The screenshot shows the Postman application interface for creating a new HTTP request. The title bar indicates it's a "GET Untitled Request". The main area is titled "Untitled Request". A toolbar at the top right includes "Save", "Edit", and "Send" buttons. Below the toolbar, the method is set to "GET" and there is a field to "Enter request URL". A "Send" button is also present. The navigation tabs include "Params" (which is selected), "Authorization", "Headers", "Body", "Pre-request Script", "Tests", and "Settings". A "Cookies" tab is visible on the right. Under the "Params" tab, there is a "Query Params" section with a table:

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

The "Response" section is currently empty, showing a placeholder icon of a rocket and an astronaut.



# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Create new request to call API

GET Untitled Request + ⚙ No Environment ⌄

Untitled Request Save ⌄ Edit ⌄ Comment ⌄

GET ⌄ Enter request URL Send ⌄

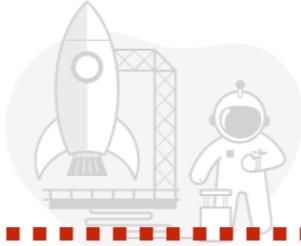
Params Authorization Headers Body Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION	⋮	Bulk Edit
	Key	Value	Description		

HTTP response

Response



# Open Weather API

Access current weather data for any location on  
Earth including over 200,000 cities

<https://openweathermap.org/current>



# Working with Weather API !!



# **Next Step**

# **Develop + Testing + Deploy**

