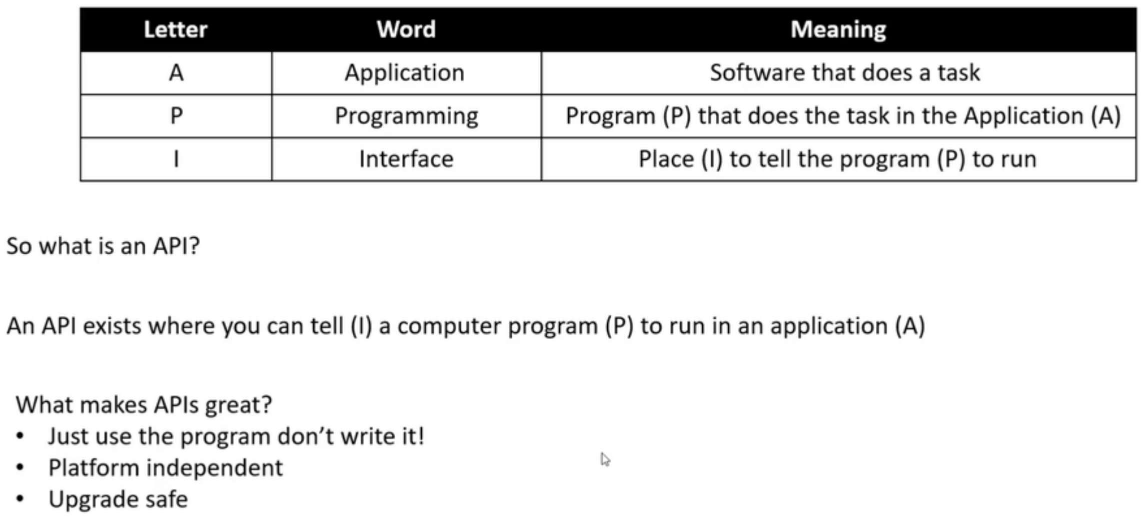
API and Web Service Introduction

This is a course taken on udemy API and Web Service Introduction , the following are my learnings.



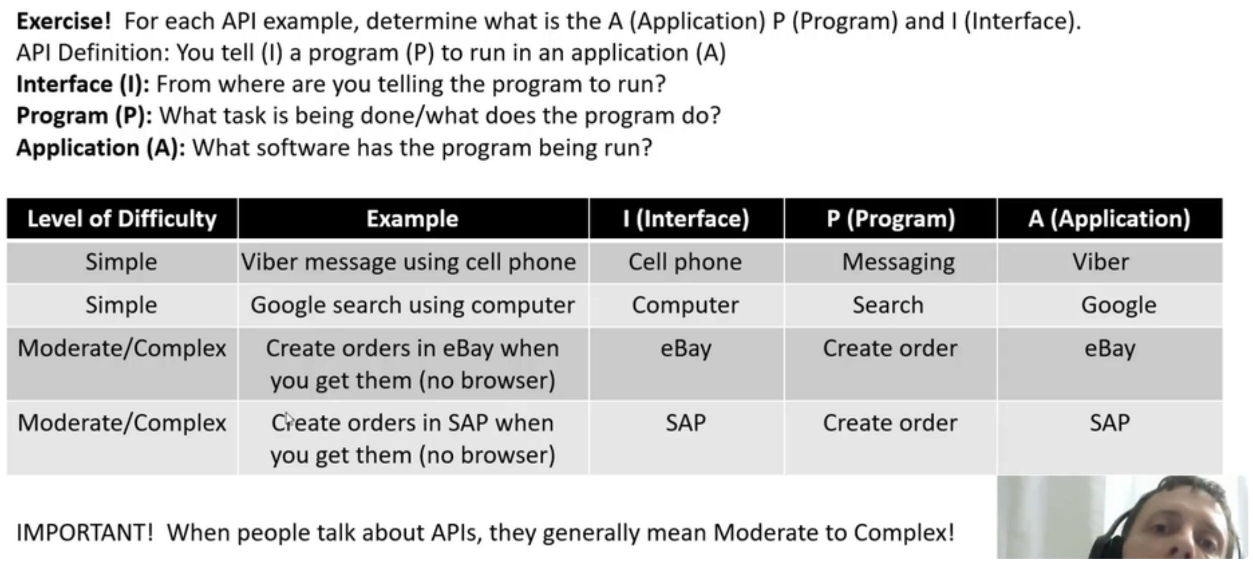
Letter	Word	Meaning
A	Application	Software that does a task
P	Programming	Program (P) that does the task in the Application (A)
I	Interface	Place (I) to tell the program (P) to run

So what is an API?

An API exists where you can tell (I) a computer program (P) to run in an application (A)

What makes APIs great?

- Just use the program don't write it!
- Platform independent
- Upgrade safe



Exercise! For each API example, determine what is the A (Application) P (Program) and I (Interface).
API Definition: You tell (I) a program (P) to run in an application (A)
Interface (I): From where are you telling the program to run?
Program (P): What task is being done/what does the program do?
Application (A): What software has the program being run?

Level of Difficulty	Example	I (Interface)	P (Program)	A (Application)
Simple	Viber message using cell phone	Cell phone	Messaging	Viber
Simple	Google search using computer	Computer	Search	Google
Moderate/Complex	Create orders in eBay when you get them (no browser)	eBay	Create order	eBay
Moderate/Complex	Create orders in SAP when you get them (no browser)	SAP	Create order	SAP

IMPORTANT! When people talk about APIs, they generally mean Moderate to Complex!

It is easier to understand what an API is by reversing the letters 'IPA' → An interface where we can run a program present in a software. Every App say GooglePay in our mobile is an API, the interface (I) being the mobile, Program (P) is sending money and Application (A) is GooglePay. But when people talk about API they don't mean this but rather a more complex thing that generally operates in a web.

Three things happen in every API transaction:

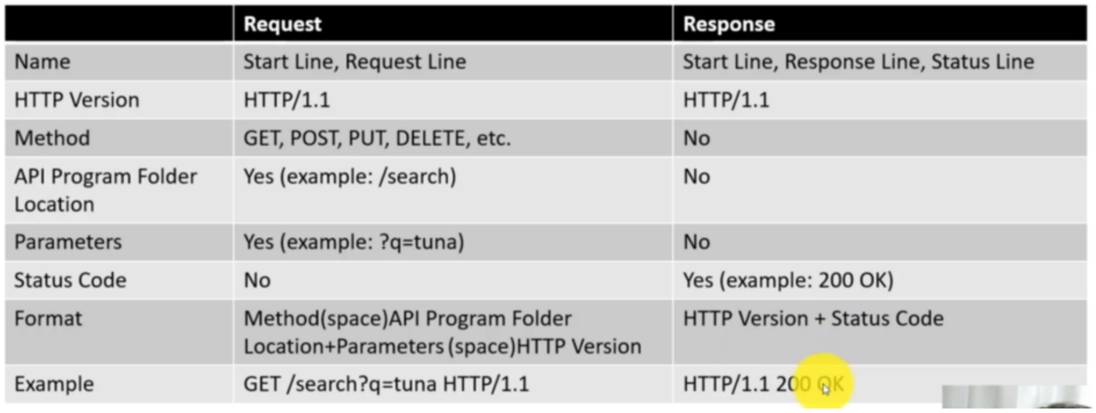
1. Request (For something to be done)
2. A program is run (To complete the request)
3. Program sends back a Response

Simple example is a google search:
`https://www.google.com/search?q=parrot` (Copy and paste this and check)

Makes a request to a google computer (we dont know where). The request is made to a program in google computer present in a folder named `search` with a parameter/query `parrot`.

A Webservice is an API that uses the internet. Web services use popularly XML or json to transfer data. The protocol (or rule) to transfer data is HTTP. HyperText meaning go to that URL (Hyper-Text). Each HTTP request or response has

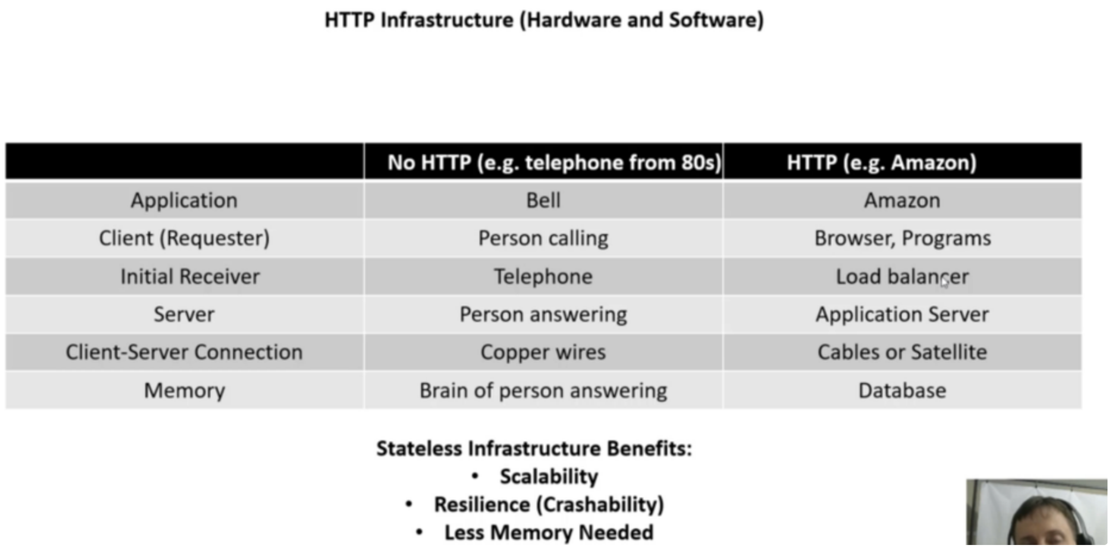
1. Start Line (Version + Method ; Version + Status Code
2. Header (Host URL + Token ; Cookie, specify what / how big they are sending
3. Blank Line
4. Body (Any input to the program such as json/xml file or image, text ; What we requested, HTML webpage in our case of parrot)



	Request	Response
Name	Start Line, Request Line	Start Line, Response Line, Status Line
HTTP Version	HTTP/1.1	HTTP/1.1
Method	GET, POST, PUT, DELETE, etc.	No
API Program Folder Location	Yes (example: /search)	No
Parameters	Yes (example: ?q=tuna)	No
Status Code	No	Yes (example: 200 OK)
Format	Method(space)API Program Folder Location+Parameters (space)HTTP Version	HTTP Version + Status Code
Example	GET /search?q=tuna HTTP/1.1	HTTP/1.1 200 OK

HTTP start line comparision

GET (Read), POST (Create), PUT (Update) and DELETE (DELETE) are the common methods (like CRUD operations).



HTTP Infrastructure (Hardware and Software)

	No HTTP (e.g. telephone from 80s)	HTTP (e.g. Amazon)
Application	Bell	Amazon
Client (Requester)	Person calling	Browser, Programs
Initial Receiver	Telephone	Load balancer
Server	Person answering	Application Server
Client-Server Connection	Copper wires	Cables or Satellite
Memory	Brain of person answering	Database

Stateless Infrastructure Benefits:

- **Scalability**
- **Resilience (Crashability)**
- **Less Memory Needed**

Understand the above analogy, it is self explanatory. It explains what happens when an API is triggered.

1. Request goes to Load balancer
2. Load balancer checks for an empty server (referred as application server above)
3. If load balancer cannot find any server it crashes (THIS IS MY ASSUMPTION)

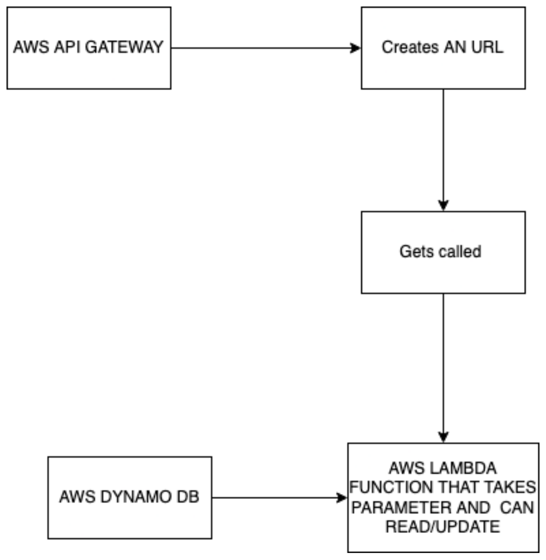
This is a HTTP infrastructure, i.e the infrastructure that is used to run our APIs. (REST API is nothing but this infrastructure of Sending requests and receiving it back - I’m sure this is what REST is in basic level, I might be slightly wrong). REST API also checks cache instead of repeatedly asking the server.
REST → Representational State Transfer must have a State

Postman is a Tool that can call an API, check the API. There was one project that was done in course where an API was created to read or update a Non relational database. This was done using

1. AWS Dynamo DB (to create DB)
2. AWS API Gateway (to create the API url like google.com/search. Note that this url is also called endpoint and it is common to say to create an endpoint)
3. AWS Lambda (Where code is written that will be executed when API is called)
4. AWS IAM roles to manage authorisation of Lambda, API gateway and Dynamo DB.

We will be implementing this in our project so was very helpful!

I have skipped the API access success that contained information about OAuth and other authorisation types which we can go through later if required.


AWS API GATEWAY
Creates AN URL
Gets called
AWS DYNAMO DB
AWS LAMBDA FUNCTION THAT TAKES PARAMETER AND CAN READ/UPDATE