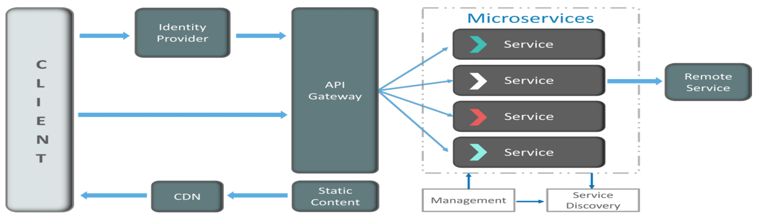
WebService	a service that a client (a computer) can call remotely over the internet, via web protocols like HTTP.		**REST API Rules:** (7 + 5)		
SOAP: Simple Object Access Protocol	XML based msg format. Doesn't specify how msg gets from the clnt to the web service, although most common scenario is via HTTP.	Envelope ● Header ● Body ○ Message Data ○ Fault (optional)	Rule # 1 Rule # 2 Rule # 3 Rule # 4 Rule # 5 Rule # 6 Rule # 7	A trailing forward slash (/) should not be included in U (/) must be used to indicate a hierarchical relationshi (-) should be used to improve the readability of URIs. Underscores (_) should not be used in URIs. Lowercase letters should be preferred in URI paths. File extensions should not be included in URIs. Should the end point be singular or plural?	
REST: REpresentational State Transfer	A REST request is a simple HTTP request just like a regular browser would send to a web server.	A REST response is typically an XML/JSON document sent back in a regular HTTP response, just as if a browser had requested it.	Rule # 8 [The choice of HTTP verbs matters.]	1. GET — For returning resources 2. POST — For creating a new resource 3. PUT — For updating a resource 4. PATCH — For updating a resource 5. DELETE — For deleting a resource	

WebService Interface
 	
`<description>`
`<types>`
`</types>`
`<interface>`
`</interface>`
`<binding>`
`</binding>`
`<service>`
`</service>`
`</description>`
	like an interface in Java. The only extra info needed is where the service is located (IP address), and the message format used by the service. Here is what a service description should contain:
	● Interface Name
	● Operation Name(s) (if the service has more than one operation).
	● Operation Input Parameters
	● Operation Return Values
	● Service Message Format
	● Service Location (IP Address / URL)
	
	SOAP WS has a standardized interface description - the Web Service Description Language (**WSDL**)
			Rule # 9
			Rule # 10
			Rule # 11
			Rule # 12
				Design API endpoints 2 be docs, don't reflect DB mod
				Provide extra data.
				Avoid nested variables when possible.
				Do not fake RPC calls through URI.	

			Idempotency	an idempotent operation is the one which does not change the target state of the server. -> OPTIONS, GET, HEAD, are safe idempotent operatio -> PUT and DELETE are non safe idempotent methods	
SOAP HEADER	The Header child elements has a list of standard attributes you can use inside them: ● mustUnderstand ● encodingStyle ● role ● relay				
WSDL	**description**	This elem is the root elem of the WSDL 2.0 file. All oth WSDL elem r nested inside this elem.			
	types	This elem contains a spec of the data types exchanged b/w the clnt and the web service. By default th data types are described using XML Schema.			
	interface	This elem describes what operations the web service has, and what messages are exchanged for each operation (input / output). It also describes possible fault messages.			
	binding	This elem describes **how** the web service is accessed over the network. Typically the binding element binds the web service to the HTTP protocol.			
	service	This elem describes **where** the web service can be accessed on the network. Typically the service elem contains a URL to the service.			
	documentation	This element is optional and may contain a humanly readable description of the web service.			
	import	This element is optional and may be used to import XML Schemas or other WSDL files.			
Differences between **HTTP** and **HTTPS**	In HTTP, URL begins with "http://" whereas URL starts with "https://" HTTP uses port number 80 for communication and HTTPS uses 443 HTTP is considered to be unsecure and HTTPS is secure HTTP Works at Application Layer and HTTPS works at Transport Layer In HTTP, Encryption is absent and Encryption is present in HTTPS as discussed above HTTP does not require any certificates and HTTPS needs SSL Certificates				
HTTP Session	HTTP is a stateless protocol. A stateless protocol does		**MicroService**	is an architectural style that	Independent DEV

	not require the server to retain information or status about each user for the duration of multiple requests. A common solution is the use of HTTP cookies. Other methods include server side sessions, hidden variables in a form, and URL-rewriting.
 C L I E N T Identity Provider API Gateway Microservices Service Service Service Service Remote Service CDN Static Content Management Service Discovery	
Clients	Different users from various devices send requests.
IDP	Auth.. user/clients, identities & issues security tokens.
API Gateway	Handles client requests.
Static Content	Houses all the content of the system.
Management	Balances services on nodes and identifies failures.
Service Discovery	A guide to find the route of comm b/w microservices.
CDN	Distributed n/w of proxy servers & their data centers.
Remote Service	Enables the remote access information that resides on a network of IT devices.
Bounded Context	DDD deals with large models by dividing them into different Bounded Contexts and being explicit about their inter-relationships.

& Advantages	structures an application as a collection of small autonomous services, modeled around a business domain.	Independent Deploym Fault Isolation Mixed Tech stack Granular scaling
Features	1 . Decoupling 2 . Componentization 3 . Business capabilities 4 . Autonomy	5 . Cont.. delivery 6 . Responsibility 7 . Decentralized gover 8 . Agility
Best practices	1 . Separate DB/Service 2 . Separate build/service	3 . Deploy into Contair 4 . Servers as stateless
Disadvantag e	1. Increa.. Troubleshooting 2. Inc.. delay due remotecall 3. Inc. effort of config/opera	5. Diff to maintain trar 6. Tough 2 track data 7. Diff 2 code b/w serv
Cohesion	The degree to which the elements inside a module belong together is said to be cohesion.	
Coupling	The measure of the strength of the dependencies between components is said to be coupling.	
Spring Boot	Using spring boot you can avoid all the boilerplate coc and configurations.	
Actuator	Spring Boot actuator provides restful web services to access the current state of running an application in tl production environment. With the help of actuator, y can check various metrics and monitor your applicatic	
Spring Cloud	Spring Cloud provides tools for developers to quickly build some of the common patterns in distributed systems (e.g. configuration management, service discovery, circuit breakers, intelligent routing, leaders election, distributed sessions, cluster state).	
Prob. Solved	● complexity associated with distributed syster ● Ability to handle Service Discovery ● redundancy issues ● Load balancing ● Reduces performance issues	

SOA	**MicroServices**
● Share as much as possible ● Business func reuse ● Common governance and standards ● ESB for communication ● Multi msg protocols ● Maximize application reuse ● CI/CD in popular, but not mainstream	● Share as little as possible ● Bounded context ● People collaboration ● Simple messaging system ● Light weight protocols like HTTP/REST ● Focus on decoupling ● Strong focus on CI/CD

Certificates	A type of digital certificate that is used by client systems to make authenticated requests to a remote server is known as the client certificate.
OAuth [open authorization protocol]	This allows accessing the resources of the resource owner by enabling the client applications on HTTP services such as third-party providers.

Containers:	Containers are a good way to manage microservice ba application to develop and deploy them individually. Y can encapsulate your microservice in a container imag along with its dependencies, which then can be used roll on-demand instances of microservice without any additional efforts required.	
DRY: Don't Repeat Yourself		

Note: Information gathered in this document has been collected from various sources on the Internet.

github.com/uvkrishnasai