# WEB CONCEPTS

uniform Resource Identifier / niform Resource locator

**HTTP PROTOCOL** – hyper text transfer protocol  
Communication between web server and client

REQUEST/RESPONSE CYCLE

HTTPS is the same but the data is encrypted by SSL/TLS

HTTP METODE:

GET: Retrieves data from the server

POST: submit data to the server

PUT: update data already on the server

DELETE: deletes data from the server  
  
HTTP STATUS CODES

1xx: Informational

Request received/processing

2xx: Success

Successfully Received, understood and accepted

3xx: Redirect

Further action must be taken/redirect

4xx: Client error  
Request does not have what it needs

5xx: Server error  
Server failed to fulfil an apparent valid request

**IMPORTANT CODES:  
200 – OK  
201 – OK CREATED**

301 – Moved to new URL

304 – Not modified (cached version)

400 – Bad request

401 – Unauthorized

404 – Not found

500 – Internal server error

HTTP HEADERS:  
GENERAL (request url, request method, status code, IP address, etc)

RESPONSE (content type, text/html, charset, date, setcookie on the client for example)

REQUEST (authority (root domain, such as facebook.com, method (GET, etc), path (URL), scheme (protocol – https for example), user-agent (browser type, OS, etc)

HTTP/2 – HTTP but better, more efficient, but does not change how everything works, its just optimized (under the hood)

DNS – Domain Name System

DNS resolves names to numbers (domain names to IP addresses)

Google.com > browser check its cache memory and OS cache memory, if not > resolver (ISP), if not > ROOT server > TLD (top level domain server, .com, .org, etc) > name server (they know everything about the domain, inc IP address) > give IP to resolver > resolver to my browser

**NGINX**

Can act as a:

**Reverse Proxy**

Browser > NGINX (web server) > AWS for example

**Load Balancer – good for scaling**

Receives a request and then forwards it to an available server out of list of servers.

**Encryption**

Instead of encrypting every server, we can just encrypt NGINX

Events {

Worker\_connections 1024;  
}  
Worker connection is a key and is known as a directive.

Event/http/wahtever are known as context

Maybe learn WEBSOCKET?

**AWS SERVERS:**

 **AWS IAM (Identity and Access Management)**: AWS IAM allows you to securely control access to AWS services and resources for your users. It enables you to create and manage AWS users and groups, assign permissions, and generate security credentials, ensuring only authorized entities can interact with your AWS environment.

 **AWS EC2 (Elastic Compute Cloud)**: AWS EC2 provides resizable compute capacity in the cloud, allowing you to quickly scale and deploy virtual servers as needed. It offers a wide selection of instance types optimized for various workloads, giving you flexibility and control over your computing resources.

 **AWS Lambda**: AWS Lambda is a serverless compute service that lets you run code without provisioning or managing servers. You can upload your code and Lambda automatically scales to handle incoming requests or events. It's ideal for building event-driven applications and microservices, offering cost efficiency and operational simplicity.

 **Security Groups**: Security groups act as virtual firewalls for your AWS instances, controlling inbound and outbound traffic. You can define rules to allow specific types of traffic to reach your instances based on protocols, ports, and IP addresses. Security groups provide an additional layer of security to protect your AWS resources from unauthorized access.

 **ACLs (Access Control Lists)**: ACLs are used to control network traffic at the subnet level in AWS. They provide a way to create rules that allow or deny traffic based on IP addresses or CIDR blocks. ACLs offer granular control over network access within your VPC (Virtual Private Cloud), enhancing security and isolation for your resources.

 **S3 (Simple Storage Service)**: AWS S3 is a scalable object storage service designed to store and retrieve any amount of data from anywhere on the web. It offers high durability, availability, and security features such as encryption and access control. S3 is commonly used for data backup, archival, content distribution, and application data storage.

 **ECS (Elastic Container Service)**: AWS ECS is a fully managed container orchestration service that supports Docker containers. It allows you to easily run, scale, and manage containerized applications using familiar AWS tools and APIs. ECS simplifies the deployment and management of containerized workloads, enabling you to focus on building your applications.

 **EKS (Elastic Kubernetes Service)**: AWS EKS is a managed Kubernetes service that makes it easy to run Kubernetes on AWS without needing to install, operate, and maintain your own Kubernetes clusters. It provides a highly available and secure platform for deploying, managing, and scaling containerized applications using Kubernetes.

 **AWS RDS (Relational Database Service)**: AWS RDS is a managed relational database service that makes it easy to set up, operate, and scale relational databases in the cloud. It supports multiple database engines such as MySQL, PostgreSQL, SQL Server, Oracle, and MariaDB, handling routine database tasks like patching, backups, and replication, allowing you to focus on your applications.