Microsoft Learn
Student Ambassadors

Workshop on Web Development with .NET & React

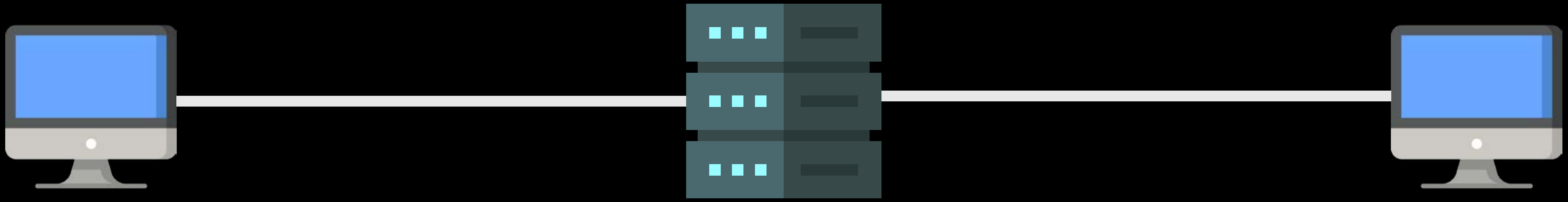
Hosted by: Ali Ahnaf

The Internet

- A long piece of wire connecting our computers across the world

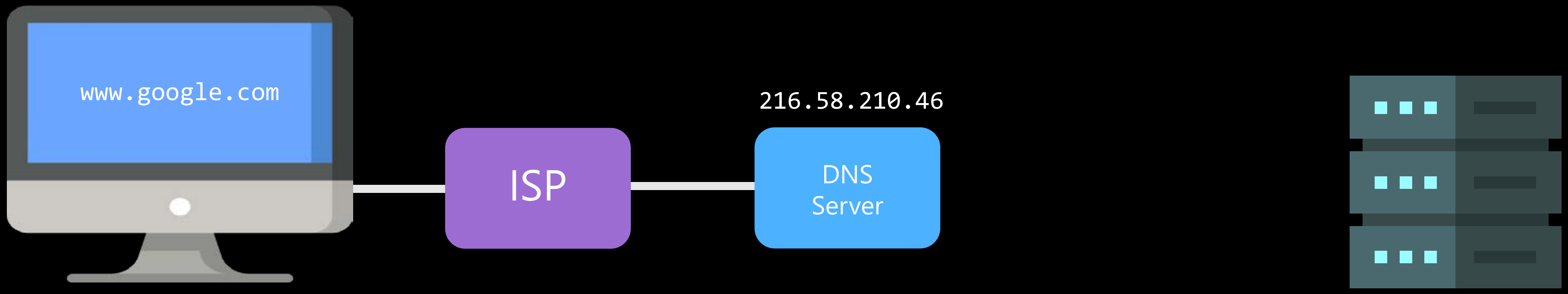
The Internet

- Some of these computers are online 24/7.
- They are called **servers**
- Servers serves files/contents to computers



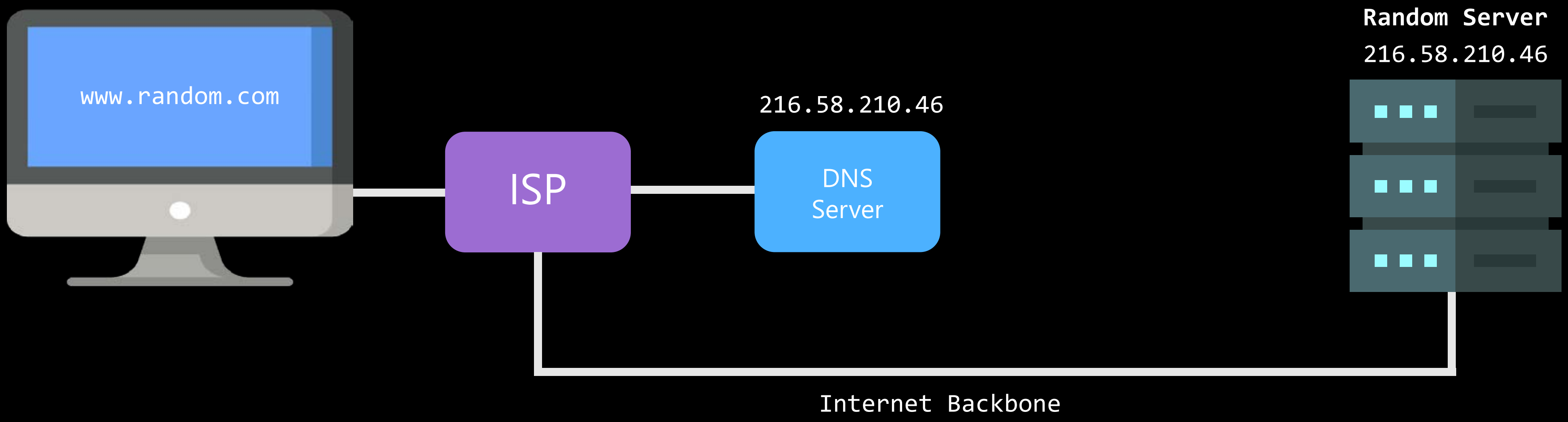
The Internet

- DNS Servers stores the IP address



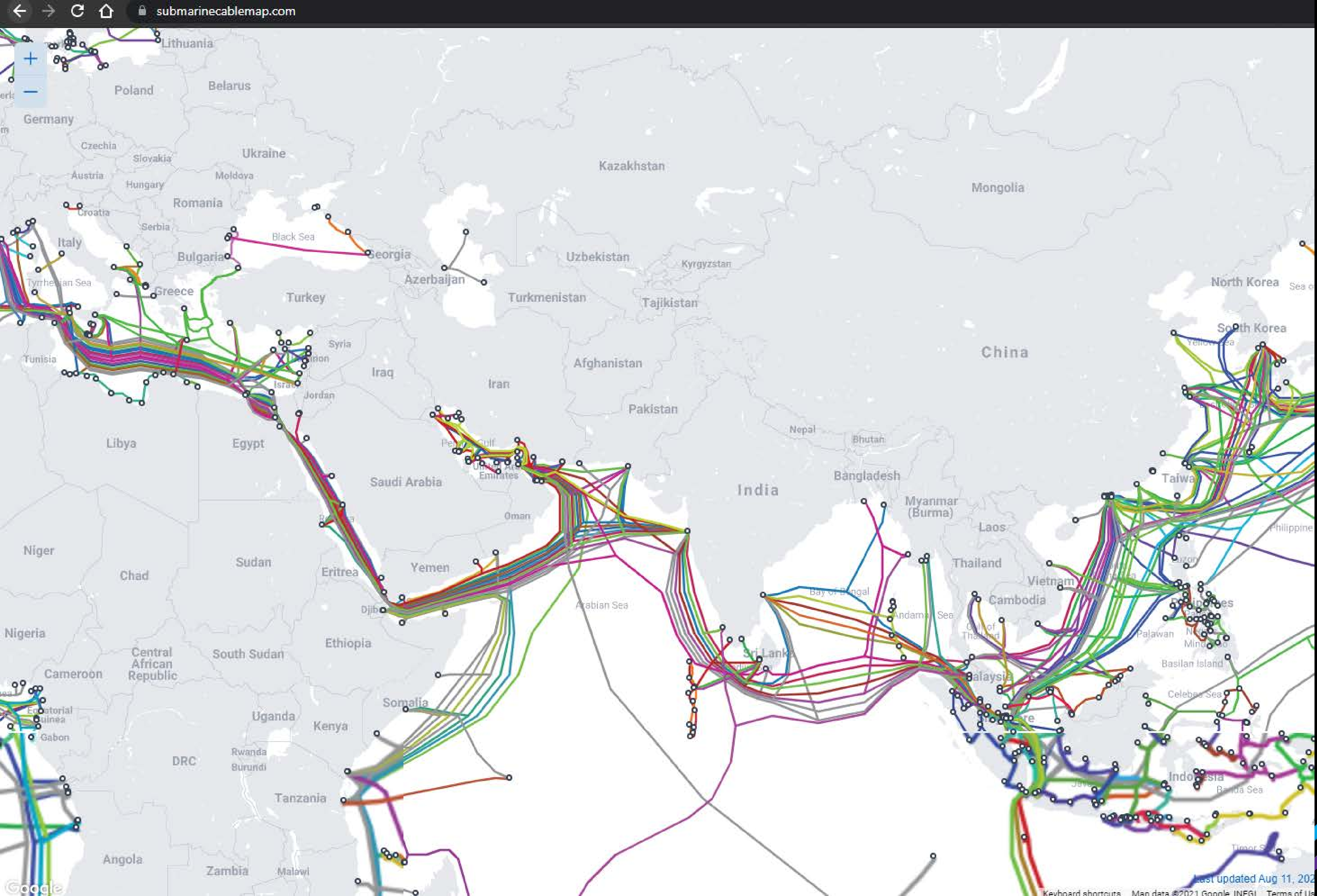
The Internet

- DNS Servers stores the IP address
- IP addresses are delivered to the Internet Backbone to point to the specified server



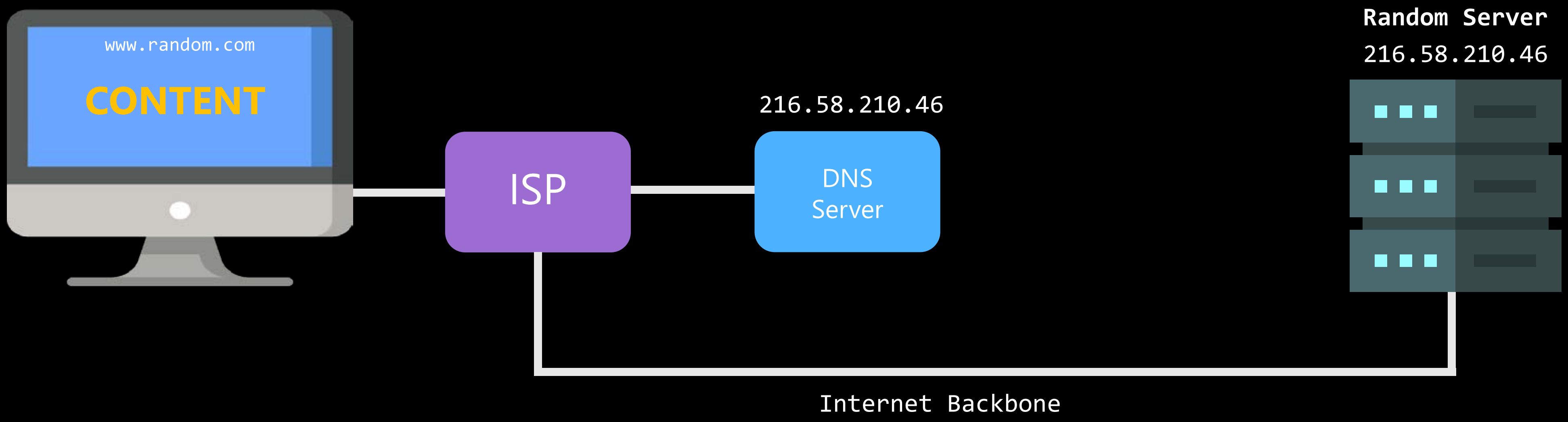
The Internet Backbone

Contains a set of submarine cables



The Internet

- The server will serve the client with the following content: HTML, CSS and Javascript



Front End

The front end contains the visual content that the user sees in a website.
They are made using HTML, CSS and Javascript.

Frameworks

Frameworks are libraries that make life easier.

Bootstrap

React

Angular

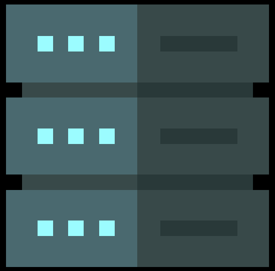
Vue

Back End

The back end contains the logic behind all the user requests.

For example:

- It serves the HTML, CSS, or Javascript contents
- It stores user’s information in a database
- Manages user accounts



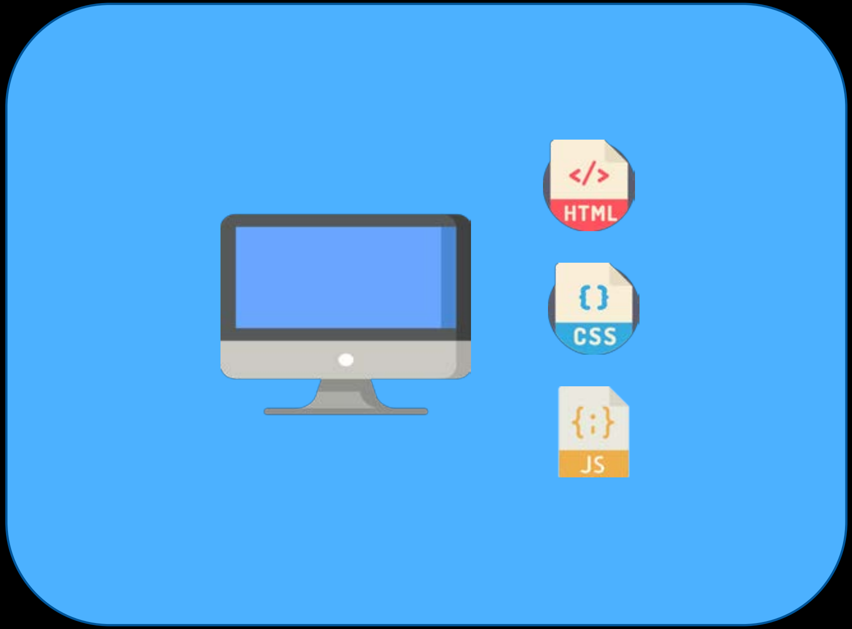
Back End

Some of the popular backend frameworks are:

- .NET
- Django
- Express.js
- Laravel
- Spring Boot

Web Development

The front end and the back end together, makes a website.

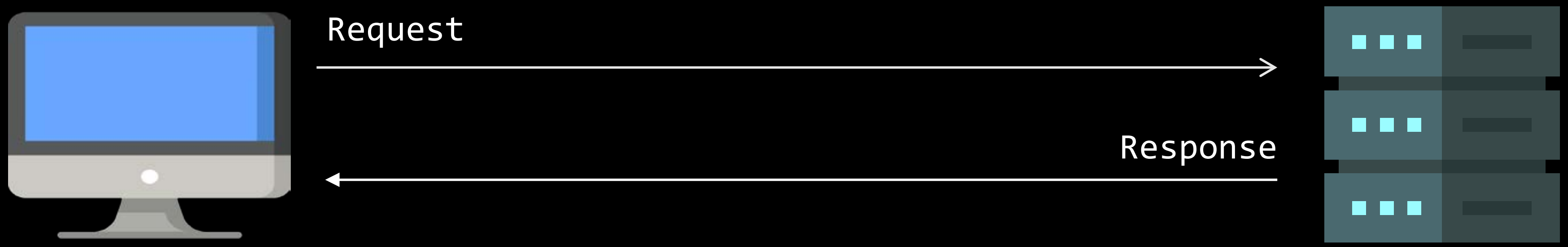


Front end

Back end

APIs

- Stands for **Application Programming Interface**
- It contains a set of rules that delivers your request to the server and gets back the response from the server back to you



API Architectures

- There are several agreed upon rules for designing APIs
- Some of them are:
  - REST
  - SOAP
  - RPC
  - GraphQL

REST API

- REST stands for “Representational State Transfer”
- Exposes data as resources where each resource is an entity
- Uses **HTTP methods** to represent CRUD (Create, Read, Update, Delete) operations, such as GET, POST, PUT or DELETE
- Provides Status Codes within a response to identify whether the operation succeeded or not

Status code	**Meaning**
2XX	Success
3XX	A resource has been moved
4XX	Error from the client side
5XX	Error from the server side

Demonstrating the React Application

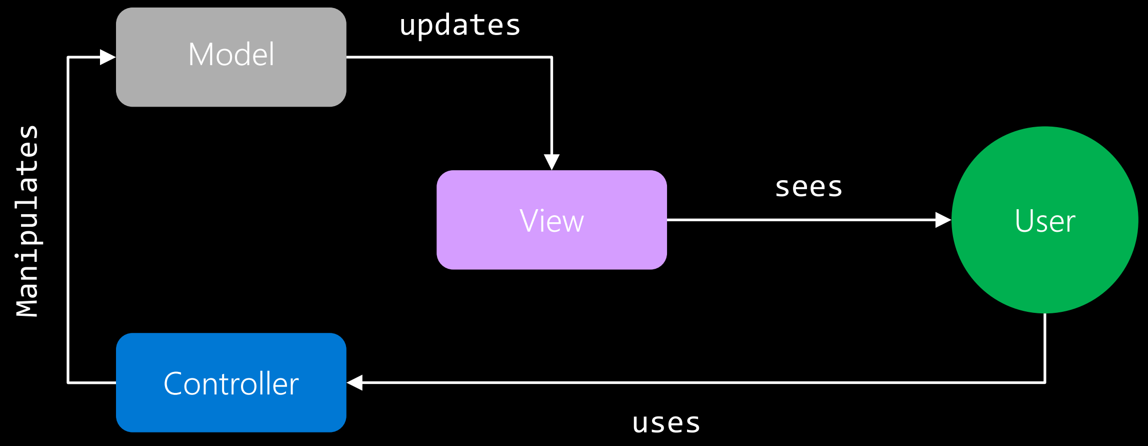
We're going to use a simple Blogging Website as a demonstration

Download the files from
https://github.com/Propo41/msa-workshop.git

MVC Pattern

It is a software design pattern that divides a program into 3 elements:

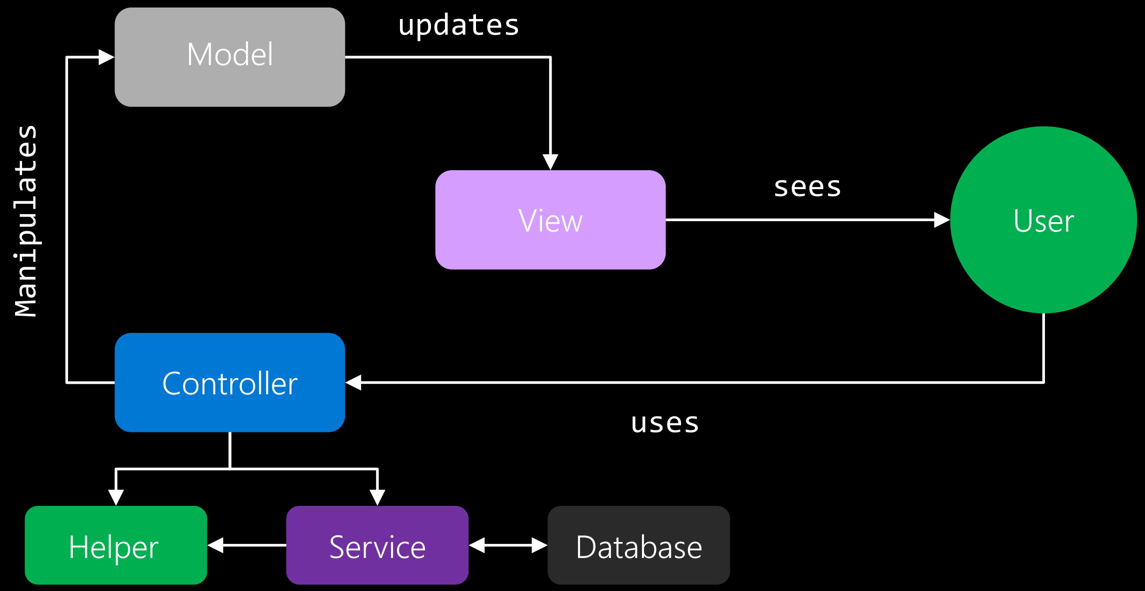
1. **Models**: holds the data
2. **Views**: renders the visual elements
3. **Controllers**: contains the business logic



Helpers & Services

Controllers may interact with additional elements, such as:

1. **Helpers**: used for keeping boiler plate code, such as converting time from 24hrs to 12hrs
2. **Services**: keeps the code for client functionality. For example, a Restaurant Service might contain service methods such as **orderFood** or **payCash**



Creating a .NET project from template

```
$ dotnet new react
```

Anatomy of a Request

A request is made up of 4 things:

1. The endpoint
2. The method
3. The headers
4. The data (or body)

Consider the following URL:

`https://api.github.com/users/propo41/repos`

Here,

`https://api.github.com` -> this is the root endpoint

`/users/propo41/repos` -> this is the path

Anatomy of a Request

The Method

It is the type of request you send to the server. Some of them are:

- GET: used to get a resource from a server
- POST: used to create a new resource
- PUT: used to update a resource
- DELETE: used to delete a resource from a server

Headers

Headers are used to provide information to both the client and server, such as for authorizing a request to a server, or providing information for the body content.

Headers are property-value pairs that are separated by a colon, for example: `"Content-Type: application/json"`

Anatomy of a Request

The Body

The body contains the information you want to send to the server.

This is only used with **POST**, **PUT**, **PATCH** or **DELETE** requests.

Databases

Database is data stored in a computer, typically in a hosted server.

There are different types of databases, with the most common ones be:

1. SQL Databases
2. NoSQL Databases

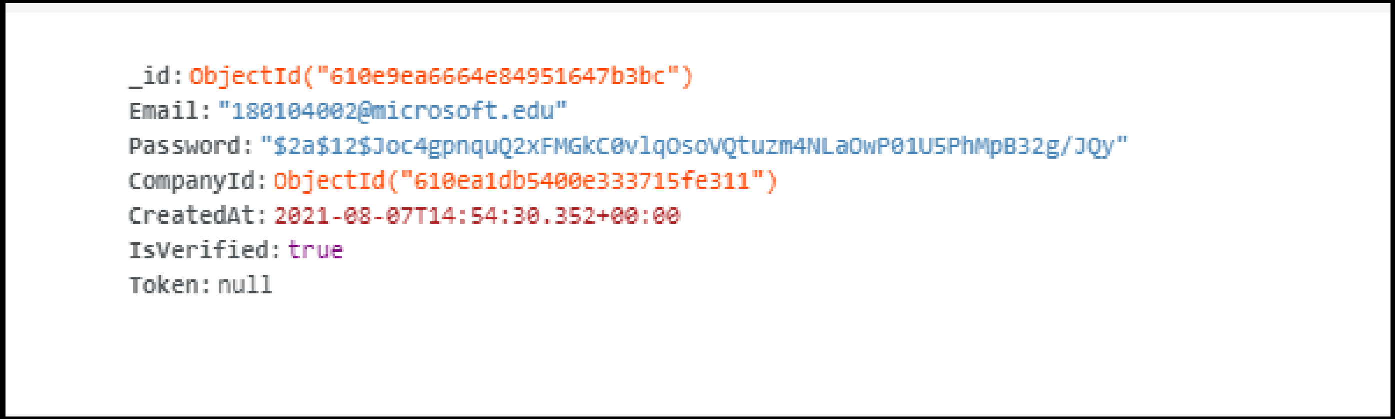
Some popular NoSQL database platforms are:

1. Firebase
2. MongoDB
3. DynamoDB

MongoDB

- Document oriented database with JSON based structure
- Free for limited amount of usage
- Easy to set up

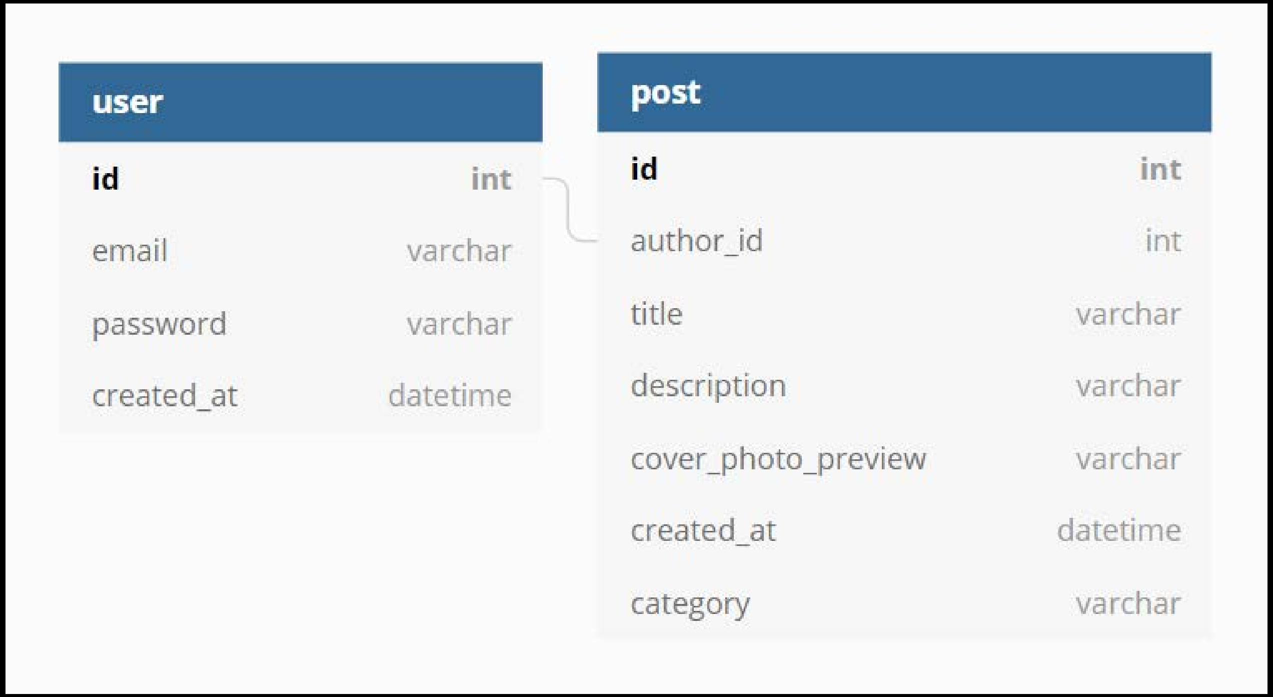
This is an example of how the data is stored in mongo DB



```
_id: ObjectId("610e9ea6664e84951647b3bc")
Email: "180104002@microsoft.edu"
Password: "$2a$12$Joc4gpnquQ2xFMGkC0vlqOsoVQtuzm4NLaOwP01U5PhMpB32g/JQy"
CompanyId: ObjectId("610ea1db5400e333715fe311")
CreatedAt: 2021-08-07T14:54:30.352+00:00
IsVerified: true
Token: null
```

Demonstrate API designing

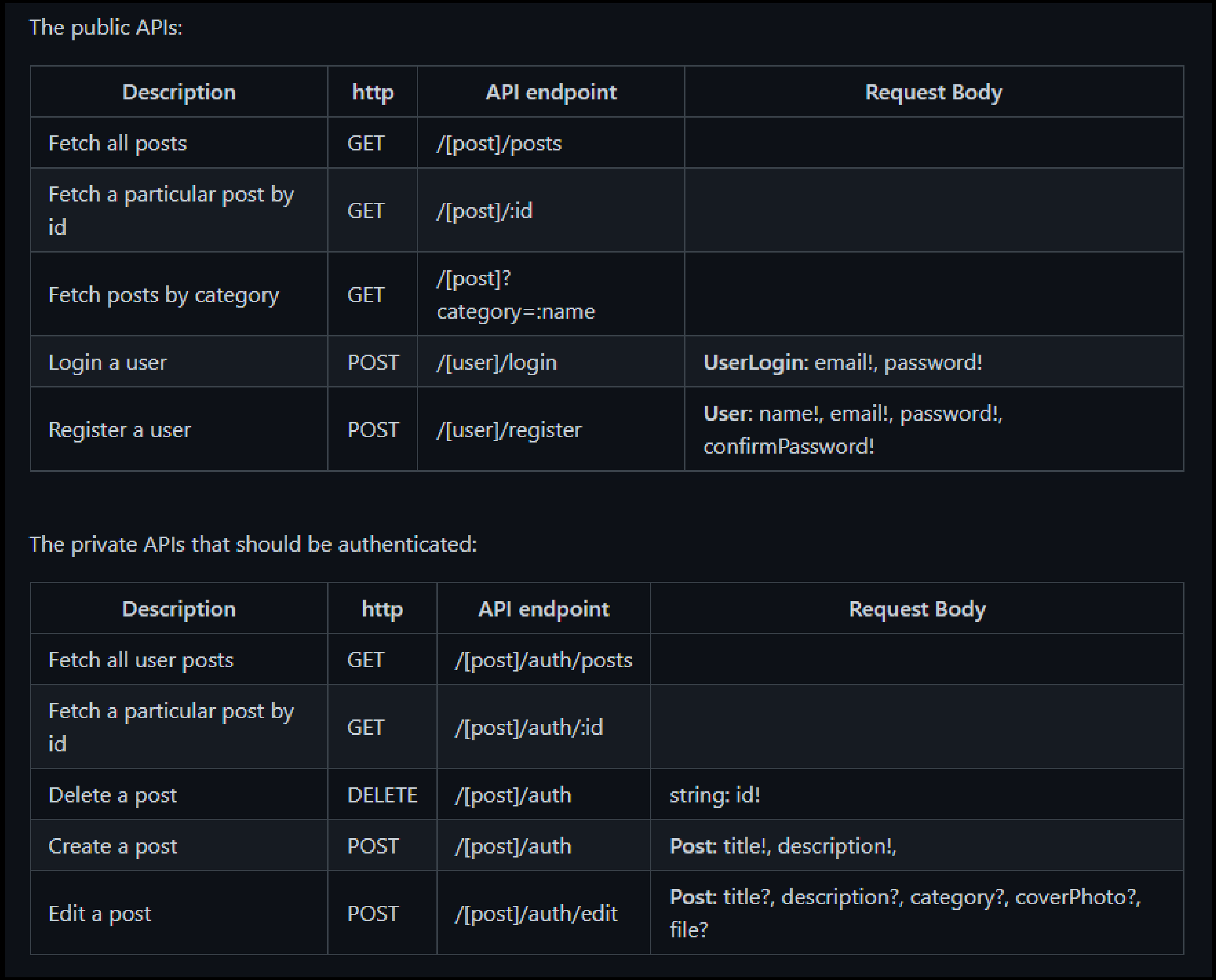
Database design



user	
id	**int**
email	varchar
password	varchar
created_at	datetime

post	
id	**int**
author_id	int
title	varchar
description	varchar
cover_photo_preview	varchar
created_at	datetime
category	varchar

API List



The public APIs:

Description	http	API endpoint	Request Body
Fetch all posts	GET	/[post]/posts	
Fetch a particular post by id	GET	/[post]/:id	
Fetch posts by category	GET	/[post]?category=:name	
Login a user	POST	/[user]/login	**UserLogin**: email!, password!
Register a user	POST	/[user]/register	**User**: name!, email!, password!, confirmPassword!

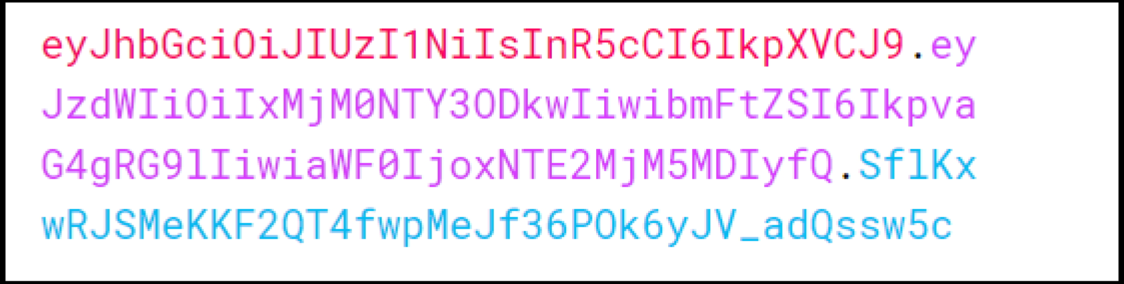
The private APIs that should be authenticated:

Description	http	API endpoint	Request Body
Fetch all user posts	GET	/[post]/auth/posts	
Fetch a particular post by id	GET	/[post]/auth/:id	
Delete a post	DELETE	/[post]/auth	string: id!
Create a post	POST	/[post]/auth	**Post:** title!, description!,
Edit a post	POST	/[post]/auth/edit	**Post:** title?, description?, category?, coverPhoto?, file?

Demonstrating how to create a login endpoint

JWT Token

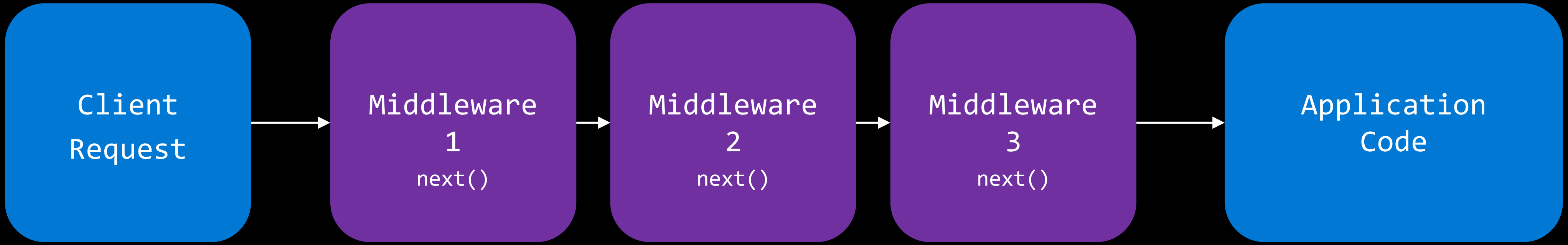
- Stands for **JSON Web Token**.
- It is a security validation mechanism used widely
- It’s a string of random alphanumeric characters.
- Has **3 parts** separated by dots:
  - Header
  - Payload
  - Signature



```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.ey
JzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6Ikpva
G4gRG9lIiwiaWF0IjoxNTE2MjM5MDIyfQ.SflKx
wRJSMeKKF2QT4fwpMeJf36POk6yJV_adQssw5c
```

Middleware

- A middleware is a function which is executed on every request
- A web application consists of several built-in middleware functions.
- Middleware functions can make changes to the request and response objects.
- next() is called to proceed executing the next middleware in the queue



Thank you

Microsoft
Learn
STUDENT AMBASSADOR

References

- https://www.smashingmagazine.com/2018/01/understanding-using-rest-api/
- https://www.tutorialsteacher.com/core/aspnet-core-middleware