**Tarea 1.1 – Análisis de arquitecturas web**

**1. Describe la arquitectura en 3 capas de un sistema web de información. Añade un dibujo o**

**diagrama de la misma.**

Una arquitectura en 3 capas de un sistema web se basa en lo siguiente:

1.-Cliente (Navegador) envia las peticiones al servidor y una vez son aceptadas, el servidor le manda los datos

2.-Servidor (Lógica de negocio) recibe las peticiones del cliente y envia las peticiones necesarias a la base de datos para conseguir la información y enviársela al cliente

3.-Base de datos (Almacena los datos) recibe las peticiones del servidor y se las envia al servidor

**2. Qué ventajas aporta una arquitectura de un SWI en 3 capas respecto al modelo cliente-servidor**

**anterior?**

En caso de que haya un problema con el servidor central no se paraliza el proceso, ya que puede haber otros servidores donde puedas conectarte como en el caso de **SAS**, si el servidor de málaga se cae, seguiremos pudiendo acceder al de Sevilla y viceversa.

**3. ¿Para qué se emplea una arquitectura de más de 3 capas?**

Las arquitecturas de más de 3 capas se utilizan para cuando queremos usar APIS (es decir, datos de terceros) estos se envían mediante un JSON al servidor y el servidor lo envía al cliente

**4. ¿Qué es una API REST? ¿Cómo se llama al formato de mensajes que se transfieren en una API**

**REST? Pon un ejemplo del mismo.**

Una API REST es una aplicación/base de datos de terceros la cual guarda información para que nosotros la podamos utilizar, el formato de mensaje que transfiere es **JSON u otros tipos**, como ejemplo de un api podríamos ver la API de Pokémon (la cual es publica) o la API de league of legends la cual necesita un token para poder utilizar.

Por ejemplo, esto es una parte del JSON del api de Pokémon:

"abilities": [

{

"ability": {

"name": "steadfast",

"url": "https://pokeapi.co/api/v2/ability/80/"

},

"is\_hidden": false,

"slot": 1

},

**5. La arquitectura en 3 capas de un sistema web de información, como sabemos, no es suficiente,**

**pues nos encontramos problemas como:**

**• Necesidad de escalabilidad horizontal –** Añadir servidores, para que esto funcione existe la necesidad de un balanceador de carga entre los servidores y el CDN, el cual va poniendo o quitando server según su necesidad

**• Necesidad de balancear la carga –** Necesitamos un balanceador de carga

**• Necesidad de distribuir los contenidos a distintos países (CDN) –** Necesitamos un Content Delivery Network

**• Certificados de SSL/TLS –** Necesitamos un certificado de la red

**• Dominios –** Necesitamos un DNS

**• etc.**

**Realiza un dibujo en el que muestres cada uno de estos componentes.**

**6. Como vimos en clase, hay distintas formas de implementar un despliegue: como un modelo on premise o mediante alquiler. Busca por internet servicios para realizar un despliegue. Necesitaremos**

**desplegar:**

**• Una base de datos MongoDB** – Podemos alojarlo en [Cloudways](https://www.cloudways.com/es/)

**• La aplicación web, hecha en nodejs**  - Podemos alojarlo en [Back4apps](https://www.back4app.com), [Heroku](https://www.heroku.com) Vercel

**• Un CDN** – [Google cloud CDN](https://cloud.google.com/cdn?hl=es), Cloudflare, Cloudfront

**• Un proveedor de dominio** - [Domain](https://www.domain.com), Ionos

**• Un proveedor de certificado SSL/TLS** - [Digicert](https://www.digicert.com/es/tls-ssl/basic-tls-ssl-certificates), Lets encrypt

**7.-EXTRA! ¿Qué es Mongo DB? ¿Cuál es su diferencia con MySQL?**

MongoDB es un sistema de base de datos NoSQL, orientado a documentos y de código abierto. En lugar de guardar los datos en tablas, tal y como se hace en las bases de datos relacionales, MongoDB guarda estructuras de datos BSON (una especificación similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.

La principal diferencia es que mientras MySQL guarda los datos en tablas, MongoDB lo hace en una estructura de datos de BSON, además de no permitir síntesis avanzada, es decir, no puedes usar “Join” como se puede hacer en MySQL

Protocolos http – 80 (puerto base) https – 443

Diferencia entre http y html

HTML es el lenguaje en el que se hacen las páginas webs

HTTP es el protocolo de las páginas webs

<http://marca.com:8080/liga/primera-division/39>

(el protocolo va a la derecha del dominio, antes de iniciar la ruta)

CDN Content Delivery Network – Su funcion es distrubuir el servidor a lo largo de los países, las peticiones van primero al CDN y después al servidor web, hay un CDN por cada pais

La escalabilidad vertical es aumentar la capacidad del servidor

La escalabilidad horizontal es aumentar el número de servidores