Preguntas generales sobre HTTP/HTTPS:

-¿Qué es HTTP y cuál es su función principal?
HTTP es un protocolo de comunicación que se usa para transmitir información entre el navegador web y un servidor.

-¿Cuál es la diferencia entre HTTP y HTTPS?
HTTPS es un HTTP pero con una mayor seguridad encriptando las solicitudes y respuestas HTTP.

-¿Cómo funciona el proceso de cifrado en HTTPS?
El navegador web cifra los datos que se envían al servidor web utilizando la clave pública del certificado TLS/SSL del sitio. Luego este cifrado lo convierte en un texto que solo es legible para las partes autorizadas. Por útlimo el sitio web recibo los datos, y con una clave privada correcta puede descifrarlo.

-¿Qué es un certificado SSL/TLS y cuál es su importancia en HTTPS?
Este certificado es una tecnología que permite cifrar los datos entre el navegador y el servidor. Es de suma importancia ya que impide que un hacker pueda acceder a esos datos y termina solo viendo una cadena de caracteres aleatoria.

-¿Qué es un método HTTP? ¿Podrías enumerar algunos de los más utilizados?
Un método HTTP es una serie de acciones que se le indica hacía un recurso que se tenga.
Dentro de los más conocidos están el GET, POST, PUT, DELETE, entro otros.

-Explica las diferencias entre los métodos HTTP GET y POST.
HTTP GET espera que se le devuelva una informacíon, por lo general suele ser una página web. En cambio HTTP POST el usuario es el que le da información al servidor.

-¿Qué es un código de estado HTTP? ¿Podrías mencionar algunos de los más comunes y lo que significan?

Los códigos de estado HTTP son unos códigos de 3 dígitos que se utilizan para indicar si una solicitud HTTP se pudo completar correctamente. Alguno de los más comunes son: El informativo (1xx), éxito (2xx), errores de cliente (4xx). Las xx previamente significan números entre el 0 y el 99.

-¿Qué es una cabecera HTTP? Da ejemplos de cabeceras comunes.
Además de intercambiarse datos en los métodos, hay “información meta” o extra. Esta información se utiliza para coordinar al cliente con el servidor. Por ej:

Request header que es la cabecera de solicitud.

Response header que es la cabecera de respuesta.

-¿En qué consiste el concepto de "idempotencia" en los métodos HTTP? ¿Qué métodos cumplen con esta característica?
Significa que el servidor calcula el estado final del recurso solo a partir de la solicitud. Si se intentara aplicar nuevamente se obtendría lo mismo. Algunos de los métodos que cumplen son el PUT y el DELETE

-¿Qué es un redirect (redirección) HTTP y cuándo es utilizado?
Es una instrucción que se envía a los navegadores para informa que una URL se cambio a otra. Se usa cuando se cambia, elimina o modifica su contenido, o cuando cambia el dominio.

Preguntas técnicas y de seguridad en HTTP/HTTPS:

-¿Cómo se asegura la integridad de los datos en una conexión HTTPS?
A través del certificado TLS/SSL se puede verificar si cierta página es realmente la verdadera. El sitio web le manda el certificado SSL que tiene la clave pública para poder iniciar sesión de forma segura.

-¿Qué diferencia hay entre un ataque de "man-in-the-middle" y un ataque de "replay" en un contexto HTTPs?
El ataque de man in the middle ocurre cuando alguien intercepta la comunicación con el servidor y hace de puente, de esta forma obtiene los datos como si fuese el

servidor. En cambio en el ataque de replay intercepta la comunicación exitosa de un usuario y la réplica.

-Explica el concepto de "handshake" en HTTPS.
Es un práctica en la que dos dispositivos se ponen de acuerdo en la forma de comunicarse. Como la velocidad, el formato que se tiene que utilizar, entro otros.

-¿Qué es HSTS (HTTP Strict Transport Security) y cómo mejora la seguridad de una aplicación web?
HSTS solo permite que se conecte al servidor si la página es https, si no lo es lo rechaza directamente. Mejora la seguridad ya que de esta manera siempre se trabaja con datos cifrados y no hay una redirección.

-¿Qué es un ataque "downgrade" y cómo HTTPS lo previene?
Este ataque lo que busca es volver a una versión anterior y por lo tanto menos segura. Https ayuda a prevenirlo mediante la actualización de sus certificados.

-¿Qué es el CORS (Cross-Origin Resource Sharing) y cómo se implementa en una aplicación web?
Es un mecanismo con el que se busca integrar las aplicaciones web, es una forma en la que un cliente con un dominio web pueda interactuar con otra página de otro dominio.

-¿Qué diferencia hay entre una cabecera Authorization y una cabecera Cookie?
En la cabecera authorization contiene las credenciales para autorizar al usuario, en cambio en la cookie está incluido en la petición http del cliente solo si ya se posee una cookie para esa página.

-¿Qué son las cabeceras de seguridad como Content-Security-Policy o X-Frame-Options? ¿Cómo ayudan a mitigar ataques comunes?
Son cabeceras que ayudar a reducir los riesgos de ciertos ataques como clickjacking.

-¿Cuáles son las diferencias entre HTTP/1.1, HTTP/2 y HTTP/3?

Http/1.1 fue una mejora del http original, en la cual se agregaron temas como múltiples solicitudes o pipelining, el http/2 fue un gran avance en la velocidad, manteniendo la compatibilidad, y http/3 no solo se centra en mejorar el rendimiento sino que también la seguridad.

-¿Qué es un "keep-alive" en HTTP y cómo mejora el rendimiento de las aplicaciones?
Permite que haya muchas solicitudes http respuestas en una sola conexión, en vez de que haya una por cada solicitud.

Preguntas de implementación práctica:

-¿Cómo manejarías la autenticación en una API basada en HTTP/HTTPS? ¿Qué métodos conoces (Basic, OAuth, JWT, etc.)?

-¿Qué es un proxy inverso (reverse proxy) y cómo se utiliza en entornos HTTP/HTTPS?

-¿Cómo implementarías una redirección automática de HTTP a HTTPS en un servidor?

-¿Cómo mitigarías un ataque de denegación de servicio (DDoS) en un servidor HTTP?

-¿Qué problemas podrías enfrentar al trabajar con APIs que dependen de HTTP, y cómo los resolverías?

-¿Qué es un cliente HTTP? ¿Mencionar la diferencia entre los clientes POSTMAN y CURL?

Preguntas de GIT

-¿Qué es GIT y para qué se utiliza en desarrollo de software?
Git es un sistema de control de versiones y en el desarrolo de software se utiliza para gestionar el código de un proyecto pudiendo varios desarroladores puedan trabajar a la vez.

-¿Cuál es la diferencia entre un repositorio local y un repositorio remoto en GIT?
El repositorio local sirve un equipo local (pc), para un individuo. En cambio un repositorio remoto se aloja en la nube, como por ejemplo github, y sirve para que varias personas puedan acceder a él y trabajar en el proyecto.

-¿Cómo se crea un nuevo repositorio en GIT y cuál es el comando para inicializarlo?
Además de previamente instalarlo se tiene que posicionar en la carpeta que quiera crear el repositorio, y a través de la consola usar el comando git init

-Explica la diferencia entre los comandos git commit y git push.
El git commit funciona a nivel local, guarda los cambios hechos. Git push los actualiza en el repositorio remoto que se esté usando, sube los cambios a la nube.

-¿Qué es un "branch" en GIT y para qué se utilizan las ramas en el desarrollo de software?
Es una rama de desarrollo. Sirven para dividir el proyecto en sub grupos para que sea más ordenado y que se lleve un mejor control de este mismo.

-¿Qué significa hacer un "merge" en GIT y cuáles son los posibles conflictos que pueden surgir durante un merge?
Significa unir distintas ramas en una sola. Puede llegar a ocurrir errores cuando en dos ramas se modifican las mismas lineas de código o cuando se eliminan archivos que otra rama este usando.

-Describe el concepto de "branching model" en GIT y menciona algunos modelos comunes (por ejemplo, Git Flow, GitHub Flow).
Es la forma de abarcar el tema de las ramas en un equipo de desarrollo. Como por ejemplo gitlab flow o trunk based development.

-¿Cómo se deshace un cambio en GIT después de hacer un commit pero antes de hacer push?
Se utiliza el comando git reset, este se usa en el ambiente local.

-¿Qué es un "pull request" y cómo contribuye a la revisión de código en un equipo?
Una vez hecho el push, el pull request es una petición al que gestiona el repositorio con tal de que este apruebe los cambios y los incorpore a la rama principal. De esta forma se tiene un mejor control de que se añade.

-¿Cómo puedes clonar un repositorio de GIT y cuál es la diferencia entre git clone y git pull?
Desde la consola se usa el comando git clone y se coloca la URL de donde está alojado el repositorio. En cambio el git pull actualiza los cambios que se hayan hecho en el repositorio remoto con el fin de que estes trabajando con la última versión.

Preguntas de Node

-¿Qué es Node.js y por qué es una opción popular para el desarrollo backend?
Es un entorno que ejecuta scripts de java y es popular debido a que se puede usar js para desarrollar desde el lado del servidor, ejecuta de forma asincrónica lo que le permite manejar muchas solicitudes a la vez, entre otros beneficios.

-¿Cómo funciona el modelo de I/O no bloqueante en Node.js y cómo beneficia el rendimiento de una aplicación backend?
Mientras se espera que se termine una tarea node permite atender otras solicitudes lo que incrementa su rendimiento.

-¿Qué es el Event Loop en Node.js y cuál es su papel en la ejecución de código asincrónico? ¿Cuál es la diferencia entre require() y import en Node.js?
Es un bucle interno de js que se encarga de manejar la pila de tareas y poder ejecutarlas de forma secuencial. La principal diferencia es que require es sincrónico

e import es asincrónico, y además import es una versión más moderna de incluir módulos.

-¿Qué es npm y cuál es su función en el ecosistema de Node.js?
Es un gestor de paquetes y la función es aprovechar código que subieron otros desarrolladores con el fin de ahorrar tiempo en ciertos aspectos.

-¿Cómo se inicializa un proyecto de Node.js usando npm y cuál es el propósito del archivo package.json?
Se inicializa con el comando npm init -y en la carpeta que queramos. Y el archivo package.json se usa para guardar los datos importantes de la aplicación.

-¿Qué son las dependencias en npm y cómo se instalan? Explica la diferencia entre dependencias y dependencias de desarrollo.
Las dependencias son los paquetes que vamos a necesitar en el proyecto. Las dependencias de desarrollo son aquellas que necesitamos mientras se desarrolla pero luego de terminar ya no son necesarias.

-¿Cómo puedes gestionar versiones específicas de paquetes en npm y para qué sirve el archivo package-lock.json?
Se gestiona con el comando npm install “el paquete”@”la versión” y el archivo package-lock.json sirve para que la persona que instale nuestro paquete tenga la versión que nosotros instalamos. En vez de el package.json que puede variar entre la versión minor o fix.

-¿Qué es nest.js cómo se usa en Node.js para construir aplicaciones backend?
Nest.js es un framework de desarrollo web basado en node.js.

-¿Cómo se manejan errores en Node.js y cuál es la diferencia entre callbacks, promesas y async/await para manejar código asincrónico?