Introducción a las preguntas teóricas:

Como sabemos, en la actualidad existen grandes modelos de inteligencia artificial con la capacidad de responder la mayoría de preguntas que podamos realizar como reclutadores. Si bien las respuestas van a ser evaluadas, la finalidad de la parte teórica es ser una guía para que el postulante pueda entender la orientación de la posición y afianzarse con los conocimientos necesarios para su desarrollo laboral.

Posteriormente a la evaluación del trabajo práctico, podrá existir una instancia de conversación donde validemos los conocimientos y el entendimiento de los conceptos.

Teoría:

Preguntas generales sobre HTTP/HTTPS:

¿Qué es HTTP y cuál es su función principal?
Es un protocolo orientado a transferencias y su función es que los clientes y servidores se comuniquen en base a este protocolo.

¿Cuál es la diferencia entre HTTP y HTTPS?
La diferencia es que en el HTTPS se cifra el intercambio de datos entre el cliente-servidor, volviéndolo más seguro a diferencia del HTTP

¿Cómo funciona el proceso de cifrado en HTTPS?
-Cuando alguien visita una página web, el navegador verifica la autenticidad del sitio pidiendo el certificado SSL/TLS
-El servidor responde con el certificado con una clave pública
-Con esto el sitio demuestra su autenticidad y el navegador utiliza la clave pública para cifrar y envía un mensaje con una clave de sesión secreta
-El sitio web usa la clave privada para descifrar el mensaje y así recuperar la sesión secreta
-Por último, el sitio web cifra la clave de sesión y envía un mensaje de confirmación al navegador para que así el navegador y el sitio web intercambien mensajes usando la misma clave de sesión.

¿Qué es un certificado SSL/TLS y cuál es su importancia en HTTPS?
Es un archivo digital que verifica la identidad entre los sistemas para luego establecer una conexión cifrada. Y la importancia es que brindan más seguridad al usuario verificando la identidad del sitio

¿Qué es un método HTTP? ¿Podrías enumerar algunos de los más utilizados?
Es la acción de la solicitud HTTP que espera del servidor
Los más utilizados son 4. GET, POST, PUT y DELETE
Explica las diferencias entre los métodos HTTP GET y POST.
El GET envía los parámetros por la URL y a diferencia del POST que lo envía por el body, es decir, en uno tiene visible los datos y en otro no.
¿Qué es un código de estado HTTP? ¿Podrías mencionar algunos de los más comunes y lo que significan?
Son grupos de códigos que responden a la solicitud HTTP dependiendo si la solicitud fue cumplida o no. Alguno son :

- 200 indica que la respuesta fue satisfactoria
- De los 400 son respuesta a que no cumplen con alguna validación. El 404 indica que no encontró los que busca.
- 500 que hay un error interno en el server.

¿Qué es una cabecera HTTP? Da ejemplos de cabeceras comunes.
Es una sección de la solicitud o respuesta HTTP donde se puede agregar más información.
En esta sección se usa mucho para la autenticación, almacenar caché o cookies

¿En qué consiste el concepto de "idempotencia" en los métodos HTTP? ¿Qué métodos cumplen con esta característica?
Consiste en que si van a hacer múltiples peticiones al mismo recurso con los mismos parámetros, el estado del recurso no cambiará.
Los métodos que cumplen con esto son los GET, PUT y DELETE
Cabe aclarar que se debe configurar este comportamiento

¿Qué es un redirect (redirección) HTTP y cuándo es utilizado?
Es la petición indicando que la URL va a cambiar y se utiliza cuando una página cambia de dominio o para corregir la URL si el cliente le faltó completar

Preguntas técnicas y de seguridad en HTTP/HTTPS:

¿Cómo se asegura la integridad de los datos en una conexión HTTPS?
Se asegura gracias al protocolo de cifrado SSL/TLS

¿Qué diferencia hay entre un ataque de "man-in-the-middle" y un ataque de "replay" en un contexto HTTPs?
-

Explica el concepto de "handshake" en HTTPS.
Es el principio cuando un cliente y servidor establecen una conexión segura involucrando un sistema de cifrado acordado. Después de esto el servidor envía el certificado al navegador para comprobar su autenticidad.

¿Qué es HSTS (HTTP Strict Transport Security) y cómo mejora la seguridad de una aplicación web?

-

¿Qué es un ataque "downgrade" y cómo HTTPS lo previene?
Es una técnica la cual obliga a usar una versión antigua como HTTP en la conexión de dos sistemas para así dejar vulnerable los datos. HTTPS lo previene deshabilitando versiones antiguas.

¿Qué es el CORS (Cross-Origin Resource Sharing) y cómo se implementa en una aplicación web?
Es un mecanismo para integrar otras aplicaciones así pueden interactuar con los recursos de un dominio distinto
Se implementa indicando el dominio, los métodos y otros datos en una clase privada.

¿Qué diferencia hay entre una cabecera Authorization y una cabecera Cookie?
La diferencia es su propósito en la cual Authorization envía un token jwt con información para autenticarlo en cambio la Cookie envía información del navegador según preferencias del usuario.

¿Qué son las cabeceras de seguridad como Content-Security-Policy o X-Frame-Options?

-

¿Cómo ayudan a mitigar ataques comunes?

-

¿Cuáles son las diferencias entre HTTP/1.1, HTTP/2 y HTTP/3?
En el 1.1 al ser de las primeras versiones estableció los conceptos básicos como los grupos de código 200, 300, 400 o 500
En el 2 permite la realización de múltiples solicitudes y respuestas en una misma conexión y en paralelo
En el 3 mejoró el sistema en cuanto a velocidad y seguridad funcionando en UDP

¿Qué es un "keep-alive" en HTTP y cómo mejora el rendimiento de las aplicaciones?
Es la comunicación tradicional pero siguiendo un esquema de solicitud-respuesta y mejora el rendimiento porque permite mantener la conexión abierta entre el cliente y el servidor para así realizar varias solicitudes sin tener que cerrar y reabrir la conexión.

Preguntas de implementación práctica:

¿Cómo manejarías la autenticación en una API basada en HTTP/HTTPS? ¿Qué métodos conoces (Basic, OAuth, JWT, etc.)?
Manejaría la autenticación generando un token JWT firmado, con la información del usuario y un tiempo de vida para que el front me lo devuelva en el Header de la petición HTTPS

¿Qué es un proxy inverso (reverse proxy) y cómo se utiliza en entornos HTTP/HTTPS?
Es un servidor que está entre el cliente y los servidores. Hace de intermediario para procesar las solicitudes y redirigirlas al servidor que corresponda.

¿Cómo implementarías una redirección automática de HTTP a HTTPS en un servidor?
Para implementarlo es necesario configurar la redirección en el servidor para que detecte las solicitudes de un puerto 80 osea HTTP y los redirija al puerto 443 osea HTTPS

¿Cómo mitigarías un ataque de denegación de servicio (DDoS) en un servidor HTTP?
Primero intentaría identificar el ataque y el tráfico legítimo, luego cambiaría las entradas DNS para redirigir el ataque a un sistema que pueda manejarlo y por último filtraría y bloquearía las peticiones maliciosas.

¿Qué problemas podrías enfrentar al trabajar con APIs que dependen de HTTP, y cómo los resolverías?
Al ser HTTP me enfrentaría a problemas de seguridad y confiabilidad, para resolverlos debería implementar un cifrado de datos en el cliente antes de enviarlos a la API.

¿Qué es un cliente HTTP? ¿Mencionar la diferencia entre los clientes POSTMAN y CURL?
Es una herramienta que permite enviar solicitudes y recibir respuestas HTTP como si fuera un navegador.
POSTMAN es una aplicación con una interfaz intuitiva, dándote la posibilidad de construir o guardar peticiones HTTP y verlos en diferentes formatos
CURL es una biblioteca que para enviar solicitudes se debe hacer mediante líneas de comandos.

Preguntas de GIT

¿Qué es GIT y para qué se utiliza en desarrollo de software?
GIT es un sistema de control de versiones y se utiliza para la gestión del código fuente ,y para trabajar en conjunto en un mismo repositorio.

¿Cuál es la diferencia entre un repositorio local y un repositorio remoto en GIT?
La diferencia en un repositorio local es que se trabaja desde una máquina local en la ubicación de una carpeta, en cambio en un repositorio remoto está en la nube.

¿Cómo se crea un nuevo repositorio en GIT y cuál es el comando para inicializarlo? Explica la diferencia entre los comandos git commit y git push.
Un repositorio nuevo se puede crear desde una carpeta local e iniciar con git init en la ubicación. Después con git add agregar los archivos que se realizaron los cambios y con git commit se guardan los cambios con un mensaje y con git push se suben a la nube.

¿Qué es un "branch" en GIT y para qué se utilizan las ramas en el desarrollo de software?
Un branch es una versión del código y estas se utilizan para trabajar en diferentes enfoques a partir de un código fuente base

¿Qué significa hacer un "merge" en GIT y cuáles son los posibles conflictos que pueden surgir durante un merge?
Git merge significa combinar los cambios con otra rama en una sola rama.Los posibles conflictos son que en un archivos en la misma línea se haya cambiado algo y se deba decidir por cuál optar o dejar ambas. Luego se comitean los cambios del merge y se puede pushear.

Describe el concepto de "branching model" en GIT y menciona algunos modelos comunes (por ejemplo, Git Flow, GitHub Flow).
-

¿Cómo se deshace un cambio en GIT después de hacer un commit pero antes de hacer push?
Para deshacer un cambio se puede usar :

```
git revert <hashCommit>
```

Para obtener el hash del commit se usa git log . El cual nos da el historial de los commits realizados y así obtener el que se quiere revertir.

¿Qué es un "pull request" y cómo contribuye a la revisión de código en un equipo?
El pull request es una solicitud para que se pueda subir los cambios a una rama y acá se puede indicar al miembro para que lo revise y confirme el merge.

¿Cómo puedes clonar un repositorio de GIT y cuál es la diferencia entre git clone y git pull?
Se puede clonar un repositorio con el comando git clone <URLrepo>. La diferencia es que con clone se bajan los cambios de la rama principal y con el pull se bajan las actualizaciones de la rama.

Preguntas de GIT —----------

¿Qué es Node.js y por qué es una opción popular para el desarrollo backend?

Node.js es un entorno para ejecutar JavaScript fuera del navegador. Se volvió últimamente popular en el back porque es capaz de manejar múltiples conexiones en simultáneo y tiene la capacidad de gestionar grandes volúmenes de datos de manera rápida.

¿Cómo funciona el modelo de I/O no bloqueante en Node.js y cómo beneficia el rendimiento de una aplicación backend?

El modelo I/O no bloqueante maneja múltiples solicitudes en asincrónico osea no tiene que esperar a que se completen una por una para pasar a la siguiente lo que beneficia el rendimiento de las aplicaciones.

¿Qué es el Event Loop en Node.js y cuál es su papel en la ejecución de código asincrónico?

Event Loop es un modelo que permite manejar una gran cantidad de eventos de forma asíncrona y su papel es ejecutar de forma independiente y sin interferir unos en otros.

¿Cuál es la diferencia entre require() y import en Node.js?

Ambos se usan para incluir módulos la diferencia es que el requiere se puede poner cualquier parte del código y el import al principio del archivo y no permite importaciones condicionales.

¿Qué es npm y cuál es su función en el ecosistema de Node.js?

Npm es un gestor de paquetes y su función es poder instalar , actualizar y gestionar bibliotecas y dependencias para el proyecto.

¿Cómo se inicializa un proyecto de Node.js usando npm y cuál es el propósito del archivo package.json?

Se abre en la ubicación de la carpeta que se esté trabajando y se escribe por línea de comando “npm init”. Esto instalará el paquete y todas las dependencias. También se generará el package.json su propósito es dar información del proyecto y las dependencias que usa con sus nombres y versiones.

¿Qué son las dependencias en npm y cómo se instalan? Explica la diferencia entre dependencias y dependencias de desarrollo.

Las dependencias son bibliotecas que se pueden incorporar al proyecto y se instala con el comando npm install <nombre-paquete>. La principal diferencia entre dependencias y dependencias de desarrollo es que la de desarrollo son herramientas en el momento de la codificación del proyecto , como, por ejemplo, Jest que sirve para el desarrollo de pruebas

unitarias, y no son dependencias que el proyecto necesite para su funcionamiento, como en el caso de express.

¿Cómo puedes gestionar versiones específicas de paquetes en npm y para qué sirve el archivo package-lock.json?
En el package.json se indica la versión específica o el rango de versiones al lado del paquete
El package-lock.json sirve para congelar las versiones que haya de cada paquete para que si alguien clona e instala todos los módulos, pueda tener la misma versión y así evitar incompatibilidades.

¿Qué es nest.js cómo se usa en Node.js para construir aplicaciones backend?
-

¿Cómo se manejan errores en Node.js y cuál es la diferencia entre callbacks, promesas y async/await para manejar código asincrónico?
Los callbacks son funciones que se pasan como argumento a otra función y se ejecutan cuando esta termina la tarea. Pero al momento de llamar a varias funciones se complejiza y se vuelven difíciles de leer y gestionar. El manejo de error es error-first en la función callback.

Las promesas pueden manejar varias operaciones asincrónicas de forma más ordenada con los métodos con el .then() y .catch() para su manejo de errores, me permite encadenar operaciones de manera más limpia pero cuando tiene varias promesas puede complicarse.

El async/await funciona como promesas pero permite escribir un código asincrónico como uno secuencial. El await espera a que una promesa se resuelva, y los errores se pueden manejar con try/catch.

Práctica

Para realizar los ejercicios prácticos deberás contar en tu ambiente de trabajo con las siguientes herramientas:

- Postman
- GIT
- Node.js instalado

La entrega deberá realizarse en un repositorio de código público que se deberá compartir al equipo de reclutamiento. El repositorio deberá contener tanto la parte teórica como la práctica

Actividad práctica número 1
Pasos:

1. Realizar una petición GET a la siguiente URL a través de Postman: https://reclutamiento-dev-procontacto-default-rtdb.firebaseio.com/reclutier.json

Ejemplo con CURL:

curl --location --request GET 'https://reclutamiento-dev-procontacto-default-rtdb.firebaseio.com/reclutier.json' \
--header 'Content-Type: application/json'

2. Realizar una petición POST a la siguiente URL a través de Postman: https://reclutamiento-dev-procontacto-default-rtdb.firebaseio.com/reclutier.json y con el siguiente body:

```
{
   "name":"TuNombre",
   "suraname":"TuApellido",
   "birthday":"1995/11/16/",
   "age":29,
   "documentType":"CUIT",
   "documentNumber":201234567
   81
}
```

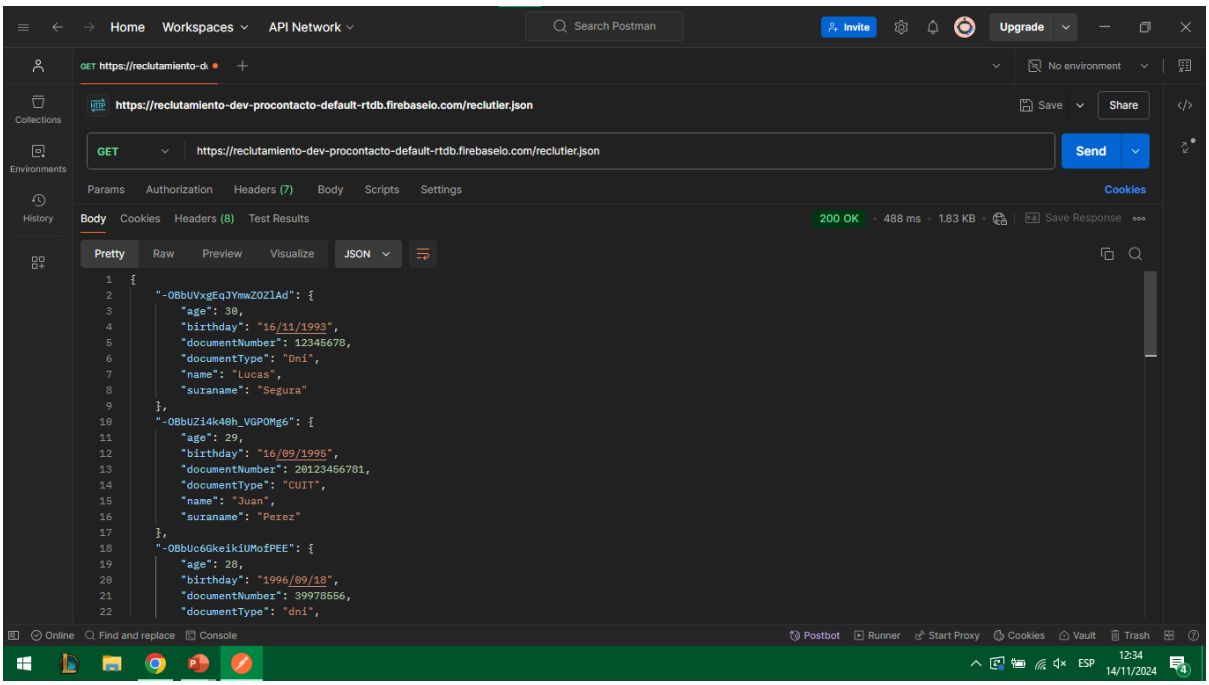
Reemplazar los campos por los valores personales tuyos.

3. Volver a realizar el GET del punto número 1.

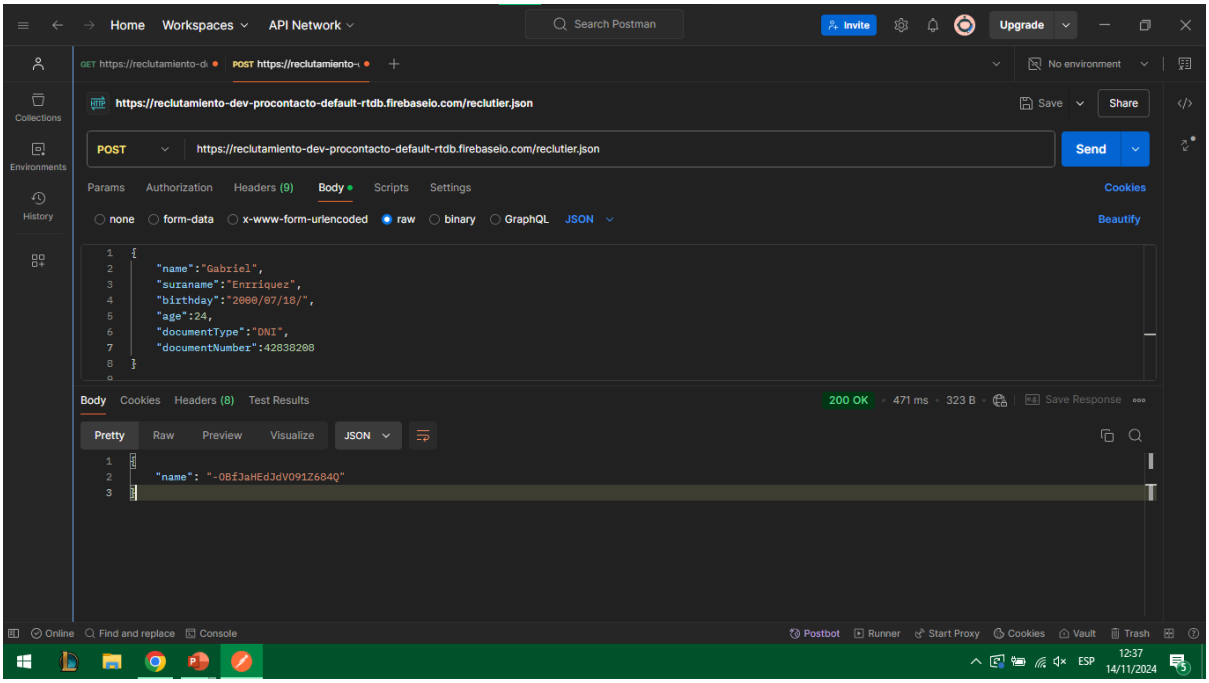
Preguntas/Ejercicios:

1. Adjuntar imagenes del response de un GET y de un POST de cada punto

GET—

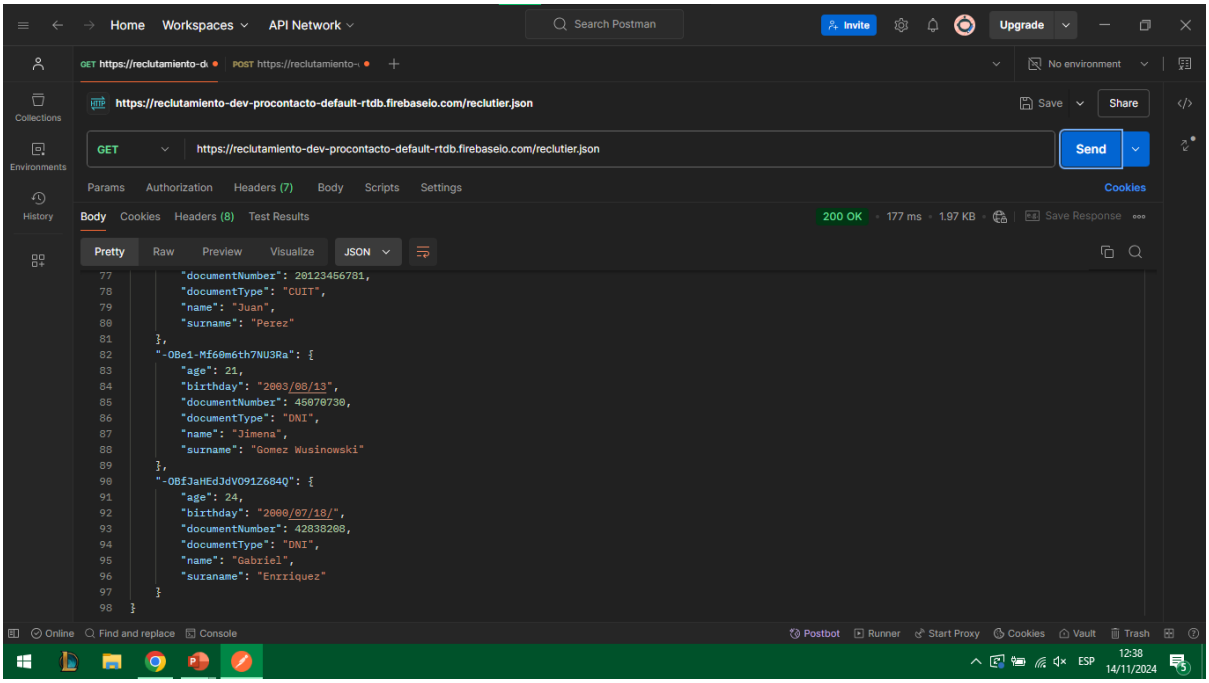


POST—



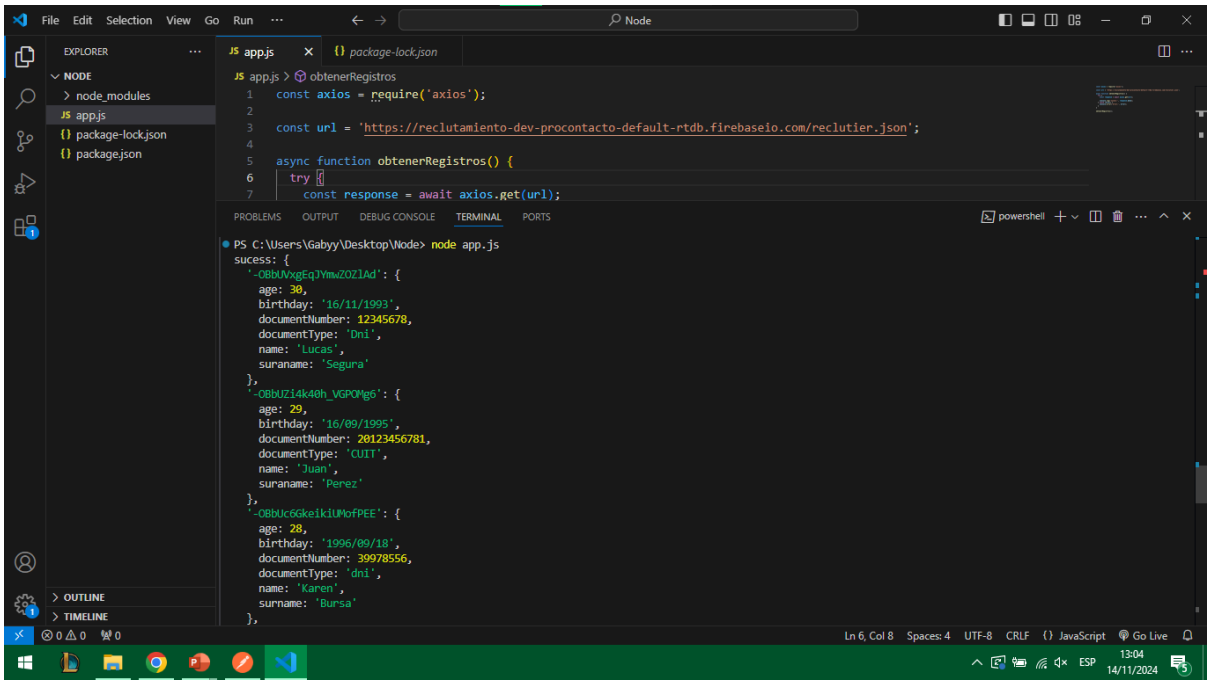
2. ¿Qué sucede cuando hacemos el GET por segunda vez, luego de haber ejecutado el POST?

Me aparece la lista y un objeto con mis datos que agregue en el post



Actividad práctica número 2

Realizar un script en Node.JS que realice un GET a la URL: https://reclutamiento-dev-procontacto-default-rtdb.firebaseio.com/reclutier.json y muestre por pantalla todos los registros.




PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
surname: 'Bursa'
},
'-OBbUfgFq6NO4c-pNgX_': {
age: 21,
birthday: ' 2003/02/27',
documentNumber: 45009507,
documentType: 'DNI',
name: 'Zoe Abigail',
suraname: 'Perez'
},
'-OBbUixCZXHS4HG1J4tE': {
age: 27,
birthday: '1997/01/05',
documentNumber: 40137650,
documentType: 'DNI',
name: 'Mauro',
surname: 'Corrales'
},
'-OBbUlNT165geAS7dmtm': {
age: 22,
birthday: '18/12/2001/',
documentNumber: 43863355,
documentType: 'DNI',
name: 'Ivone',
suraname: 'Corleto'
},


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
suraname: 'Corleto'
},
'-OBbaDK_MPrkwt9JL0cF': {
age: 22,
birthday: '2001/12/18',
documentNumber: 43863355,
documentType: 'DNI',
name: 'Ivone',
surname: 'Corleto'
},
'-OBc1lfarf9LnRSnOXo4': {
age: 22,
birthday: '2002/06/30/',
documentNumber: 44318250,
documentType: 'DNI',
name: 'Brian',
suraname: 'Hidalgo'
},
'-OBdTtd0Fc1DlBzk1WIT': {
age: 21,
birthday: '2003/08/13/',
documentNumber: 45070730,
documentType: 'DNI',
name: 'Jimena',
suraname: 'Gomez Wusinowski'
},

Actividad práctica número 3:

Realizar un Servicio Web en nestjs que permita recibir una petición post con el formato:

```
{
   "name":"TuNombre",
   "suraname":"TuApellido",
   "birthday":"1995/11/16/",
   "age":29,
   "documentType":"CUIT",
   "documentNumber":201234567
   81
}
```

una vez recibida la petición deberá ejecutar otra petición hacia el servicio web de reclutamiento:
https://reclutamiento-dev-procontacto-default-rtdb.firebaseio.com/reclutier.json
Adicionalmente el servicio web de nestjs deberá tener las siguientes características:

1. El campo Name y Surename deberán empezar siempre con la primer letra de cada palabra en mayúscula, si se recibe una petición con otro formato deberá normalizarse al formato esperado
2. Validar que el campo birthday tenga un formato válido en el formato YYYY/MM/DD. La fecha proporcionada no podrá ser posterior al día de hoy ni anterior al 1900/01/01. En caso que no se cumpla alguna petición se deberá rechazar la petición
3. El campo age deberá ser un número entero. En caso que no se cumpla alguna petición se deberá rechazar la petición
4. Los valores posibles de documentType son: “CUIT” o “DNI” si se envía otros valores se deberá rechazar la petición