Universidad Nacional de Colombia - sede Bogotá
Facultad de Ingeniería
Departamento de Sistemas e Industrial
Curso: Ingeniería de Software 1 (2016701)

Entrega 6 Parte 2 - Grupo 7 UNEmployed

Nuestro Linter

Debido a que nuestro proyecto está siendo desarrollado en TypeScript, el cual a su vez podemos ver como una extensión de JavaScript, decidimos utilizar ESLint como nuestro linter.

Cabe destacar que nuestro proyecto no usa una estructura monorepo per se; a pesar de que tanto el BackEnd como el FrontEnd están en un directorio padre, cada parte tiene configuraciones totalmente independientes en lo que a su `package.json` respecta. Por esto precisamente en vez de instalar el Linter en la raíz del proyecto, lo instalamos en ambas carpetas.

Por defecto, ESLint es usado para JavaScript, por tanto, tuvimos que añadir los siguientes paquetes como dependencias de desarrollo para que el linter pudiera identificar la sintaxis propia de TypeScript: `@typescript-eslint/eslint-plugin`, `@typescript-eslint/parser`.

La configuración del Linter

En cada directorio raíz, se ubicó un fichero llamado `eslint.config.js`, el cual define la configuración que va a utilizar nuestro Linter al revisar nuestro proyecto. Por defecto cargamos las reglas recomendadas para JavaScript y las sobreescribimos parcialmente con las reglas recomendadas para TypeScript. Finalmente, en el objeto que cargamos se puede ver la propiedad `rules`, esta propiedad contiene una configuración personalizada para una cantidad significativa de reglas para las que ESLint da soporte. Varias de ellas lo que hacen es sobreescribir algunas configuraciones por defecto que ofrecen las plantillas que usamos, y otras varias son personalizaciones que hemos traído de proyectos en los que hemos trabajado usando JavaScript.

Nota: En las reglas definidas se puede ver que la regla `'arraybracket-spacing'` se encuentra comentada. Esto lo hicimos debido a que la versión que instalamos del linter deprecó el soporte para esta regla y para migrar a la nueva versión de esta requería instalar un plugin adicional entero para solo esta regla.

Reporte del Linter

Una vez logramos correr el linter por primera vez en ambos directorios se obtuvieron los siguientes resultados:

None

```
@Yareaj ➜ /workspaces/ingesoft-i/Proyecto/Backend
(feat/linter-setup) $ npx eslint

/workspaces/ingesoft-i/Proyecto/Backend/eslint.config.js
  7:1  error  Expected indentation of 1 tab but found 4 spaces
indent
...
...
...
/workspaces/ingesoft-i/Proyecto/Backend/index.ts
  1:27  error  Trailing spaces not allowed  no-trailing-spaces
✖ 162 problems (161 errors, 1 warning)
  161 errors and 0 warnings potentially fixable with the `--fix`
option.
```

None

```
@Yareaj ➜ /workspaces/ingesoft-i/Proyecto/Frontend
(feat/linter-setup) $ npx eslint

/workspaces/ingesoft-i/Proyecto/Frontend/App.tsx
  6:1  error  Expected indentation of 1 tab but found 2 spaces
indent

/workspaces/ingesoft-i/Proyecto/Frontend/src/screens/LoginScreen.
tsx
   10:1   error    Expected indentation of 1 tab but found 2
spaces    indent
   11:1   error    Expected indentation of 1 tab but found 2
spaces    indent
...
...
...

✖ 347 problems (314 errors, 33 warnings)
  313 errors and 33 warnings potentially fixable with the `--fix`
option.
```

Aplicando la flag de `--fix` pudimos resolver la mayoría de problemas y solo quedarnos con un warning de un builder vacío que se está usando para autenticar usuarios, este fue el output generado por la terminal al aplicar las correcciones y posteriormente la verificación:

```
None
@Yareaj ➜ /workspaces/ingesoft-i/Proyecto/Frontend (main) $ npx eslint --stats
@Yareaj ➜ /workspaces/ingesoft-i/Proyecto/Frontend (main) $ cd ../Backend/
@Yareaj ➜ /workspaces/ingesoft-i/Proyecto/Backend (main) $ npx eslint --stats

/workspaces/ingesoft-i/Proyecto/Backend/src/db_connection/db/Database.ts
  8:24  warning  Unexpected empty constructor  no-empty-function

✖ 1 problem (0 errors, 1 warning)

@Yareaj ➜ /workspaces/ingesoft-i/Proyecto/Backend (main) $
```

Una nota importante a realizar y fue algo que comentamos durante la clase fue que el linter se corre por separado en ambos repositorios ya que no tenemos un `package.json` en el directorio raíz ni mucho menos los `node_modules,` y era necesario generarlos debido a los paquetes adicionales de configuraciones que utilizamos.