**Xavier Ledesma Pons**

**Ejercicio 1**

La **PWA** que he diseñado, permite introducir un nombre de usuario de **Github** y muestra un listado con todos los repositorios públicos de ese usuario. Si entramos en el detalle de un repositorio, podemos ver información sobre la rama principal, la url o la imagen de perfil del usuario.

Una vez ejecutado **Lighthouse**, he obtenido el siguiente resultado. Cabe destacar que inicialmente he tenido algunos puntos que faltaba alguna cosa, pero los he modifico para obtener una correcta puntuación.

En el caso del **SEO** en el fichero **index.html** he añadido el **meta tag description** ya que faltaba y eso hace que la indexación en internet sea mejor. En el apartado de **PWA**, me faltaba añadir en **index.html** un enlace para **apple-touch-icon** (icono cuando se instala la app en ios). También he incluido una batería de iconos para los distintos dispositivos.

**Captura de pantalla de un celular

Descripción generada automáticamente**

Ilustración 1: Resultado de Lighthouse

En cuanto al service worker, he creado el **dataGroup** **github-api**, que almacena en cache las urls [https://api.github.com/\*\*](https://api.github.com/**) y [https://avatars.githubusercontent.com/u/\*\*](https://avatars.githubusercontent.com/u/**) que se usan para obtener la información de los repositorios y las imágenes de perfil.

La configuración de caché utilizada ha sido la misma que la que se presenta en el apartado de teoría de esta PEC.

**Ejercicio 2**

|  |  |
| --- | --- |
| **Fichero** | **Descripción** |
| **server.ts** | Incluye el código del servidor **express** encargado de servir los ficheros generados. |
| **app.server.module.ts** | Modulo del servidor web generado con **express** donde se inyectan las dependencias. |
| **main.server.ts** | **Main** encargado de inicializar el entorno antes de arrancar el servidor **express**. |
| **tsconfig.server.ts** | Fichero de configuración de la aplicación del servidor donde se especifican los ficheros y las configuraciones principales de **typescript** para el proyecto. |

**Ejercicio 3**

En este apartado se ha desplegado la aplicación en **Netlify**, obteniendo un enlace de acceso público.

**Netlify URL**: <https://friendly-benz-28a30a.netlify.app>

Como podemos ver, cuando accedemos en un dispositivo móvil Android, el navegador detecta que la web es una **PWA** y nos muestra un mensaje para añadirla a la pantalla principal. El resultado se puede observar en las siguientes imágenes:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamenteImagen de la pantalla de un celular

Descripción generada automáticamente

Ilustración 2: Instalar PWA en nuestro dispositivo móvil

En versión escritorio, Chrome también nos ofrece la opción de instalar la aplicación en su deposito de aplicaciones.