ANTEPROYECTO

Proyecto de final de curso

Jaime Martínez Rincón

Desarrollo de Aplicaciones Multiplataforma

Curso 2019 – 2020 | IES Francisco de Goya

Índice

1. Explicación de la idea

Mi idea de proyecto es de una plataforma web de monitorización del tiempo usado al trabajar. Esta idea ha surgido a partir del aumento de trabajadores en remoto debido a la crisis del COVID-19.

En mi opinión, a los empleadores les asusta el trabajo remoto por muchas razones, una de ellas es la falta de confianza en la productividad o falta de atención al trabajar debido a las distracciones que pueden ocurrir en tu casa.

Mi plataforma web intenta solucionar esto dando al empleador y empleado una plataforma en la que se puede consultar el tiempo consumido en la jornada laboral, esta información se categoriza por cada aplicación usada.

Además, se proveen estadísticas relacionadas (inactividad, atención, productividad) y pruebas (ver que ventanas se han estado utilizando)

Mi proyecto no solo está enfocado para empresas con trabajadores en remoto, también quiero que se pueda usar por “freelancers”, los cuales necesitan herramientas robustas y seguras para facturar y controlar el tiempo usado en sus proyectos.

Otra oportunidad de uso es para personas que se distraen fácilmente (como yo) y quieren mejorar su eficiencia al saber que es lo hacen exactamente y por cuanto tiempo.

En el momento de la creación de este documento, no existe ninguna alternativa comparable con los requisitos de mi proyecto, que son los siguientes:

- Tiene que ser totalmente gratuita para uso personal
- Tiene que ser compatible con Windows y Linux
- Tiene que monitorizar el tiempo usado por ventana
- Tiene que soportar proyectos con múltiples usuarios (ejemplo: jefe y trabajador)

Para poder monitorizar las ventanas activas y el tiempo activo, se necesita un cliente de escritorio el cual tiene que enviar esta información a la plataforma web.

La aplicación está formada por usuarios, los cuales pueden crear proyectos. Estos proyectos pueden tener mas miembros, a los cuales se les pueden asignar roles, como “Manager” o “Trabajador”. En cada proyecto, se pueden ver sesiones, iniciadas por el cliente.

La aplicación cliente tiene que permitir iniciar sesión, elegir un proyecto (creado previamente en la plataforma web), iniciar una sesión de trabajo (empezar a contar el tiempo usado), y obviamente, parar una sesión de trabajo y cerrar sesión.

La plataforma web es la que tiene que soportar iniciar sesión, crear una cuenta, recuperar cuenta, crear y borrar proyectos, invitar usuarios a un proyecto, ver las sesiones por cada proyecto ordenadas cronológicamente y ver las estadísticas de uso por cada sesión.

2. Tecnologías pensadas

2.1 Plataforma web

Para la realización de la parte de la plataforma web he pensado en utilizar las siguientes tecnologías:

React

https://es.reactjs.org/

Es una librería de JavaScript para construir interfaces de usuario mediante componentes, con muy buen rendimiento, con manejo de estado y desarrollo declarativo (los elementos se subscriben al cambio del estado, en vez de cambiar los elementos cuando cambiamos el estado). Es comparable con otros proyectos como Angular, Vue o Elm pero siendo mucho más ligero y modular.

Tiene una gran comunidad detrás y usado por grandes empresas como Facebook (sus creadores), Amazon, Microsoft, Twitter, Uber, Netflix, Airbnb, etc.

TypeORM

https://typeorm.io/

Es un ORM del mundo de NodeJS, su desarrollo ha sido influenciado por Hibernate, Doctrine y Entity de Spring.

Soporta los sistemas de base de datos más populares, modelos, asociaciones, migraciones, herencia, entre muchas otras.

He elegido esta librería porque parece ser el ORM mas popular en NodeJS y por todas las funcionalidades que soporta.

Express

https://expressjs.com/es/

Es la librería más popular del mundo de NodeJS para crear servidores web. Es muy fácil de utilizar, muy rápido, con una gran comunidad detrás y muy completo.

Es el encargado de aceptar peticiones del “frontend” y de la aplicación de escritorio, realizar los cambios en la base de datos a través del ORM.

PostgreSQL

https://www.postgresql.org/

Es un sistema de base de datos de código abierto y totalmente libre, estable, rápido y sencillo de aprender. Lo que me ha hecho interesarme en el es que soporta mas funcionalidades que MySQL y que soporta herencia de tablas y tipos de objetos que MySQL no soporta, como BOOLEAN, JSON y XML.

TailwindCSS

https://tailwindcss.com/

Es una framework de estilos de CSS minimalista, se podría comprar con Bootstrap, pero TailwindCSS no te da elementos por defecto, solo te da las clases CSS para crear tus componentes sin tener que crear mucho CSS.

Se podría decir que en general, nuestro frontend estaría formado por React y TailwindCSS, y que el backend estaría formado por Express, PostgreSQL y TypeORM.

Un repunte adicional es que para el frontend voy a utilizar JavaScript ES6 (ECMAScript 6) pero para el backend voy a utilizar TypeScript, que es básicamente JavaScript pero con tipos, y al compilarlo se genera un JavaScript minificado.

Para desplegar la plataforma podemos desplegar el frontend separado del backend o hacer que el backend sirva el frontend.

Creo que para empezar sería mejor tenerlos separados y en el futuro unirlos. En este caso el frontend sería desplegado en Netlify (un servicio gratuito para servir páginas web estáticas) y el backend sería desplegado en AWS EC2 (gratis gracias a los créditos para estudiante).

2.2 Cliente multiplataforma

Para el cliente multiplataforma, he decidido usar Qt, un framework muy utilizado en el ámbito empresarial, gratuito para uso no comercial y de código abierto.

Para realizar aplicaciones en Qt, se necesita saber C++ para la lógica y QML para los elementos graficos, lo que me permite acercarme lo máximo posible a las APIs de bajo nivel de cada sistema operativo, en las cuales voy a tener que depender para realizar esta aplicación.

He decidido utilizar esta framework porque compila para todos los sistemas operativos actuales, incluso para dispositivos móviles, y además al usar C++ no debería tener mucho problema en acceder las APIs nativas.

3. Objetivos del proyecto

Como objetivos del proyecto, me he propuesto lo siguiente:

- Aprender la ultima versión de React, creando código reutilizable y conciso.
- Aprender a realizar autenticación mediante React y Express con sesiones o tokens.
- Aprender alguna funcionalidad relacionada con las bases de datos gracias a PostgreSQL y TypeORM.
- Poner en práctica mis conocimientos básicos de C y C++ para realizar la app de cliente.
- Mejorar mis habilidades de diseño web y de arquitectura de sistemas.
- Realizar una aplicación que yo pueda utilizar y que sea agradable e intuitiva al usar.
- Realizar todas las funcionalidades que me he propuesto, es un proyecto ambicioso sobre mi punto de vista, pero si tengo suerte tendré un resultado bastante bueno.
- Utilizar repositorios de Git para mi código y Notion para tener notas y organizar las tareas que tengo que realizar con un tablero Kanban.

4. Alcance y limitaciones

Aunque en mi opinión, mi proyecto es bastante ambicioso, quiero llegar a poder realizar todas las funcionalidades previstas y que funcione todo de forma correcta. Y, sobre todo, aprender mucho, es por lo que he intentado utilizar todo menos de lo ya sé.

Si tengo tiempo, se podría añadir las siguientes funcionalidades adicionales:

- Hacer capturas de pantalla automáticas
- Ver el tiempo inactivo (sin tocar el ratón o/y teclado)
- Soporte para dispositivos móviles, etc.
- Enviar correos electrónicos (confirmación alta, recordar contraseña, resumen mensual)

Aun así, hay que ser realistas y por eso tengo en cuenta que puedo encontrarme con varios problemas:

- Puede que aprender React sea más difícil de lo que parece y que tarde demasiado tiempo en ello, he hecho ya varias pruebas y parece viable, pero son solo pruebas, no proyectos completos.
- La autentificación puede ser difícil de implementar, ya sea para la relación backend <-> frontend o para backend <-> app cliente.
- Puede que el código crezca demasiado y que me empiece a perder, tendré que tener cuidado ya que esto puede dar lugar a problemas difíciles de arreglar.
- Puede que el realizar la interfaz web se complique demasiado al tener que diseñar yo mi propia interfaz, si ese es el caso lo haré con Bootstrap, que me da elementos ya diseñados que son bastante aceptables.

5. Procedimiento

Para realizar este proyecto voy a tener que verificar que es viable, antes de nada, para ello he realizado una prueba en cada plataforma a la que tengo acceso (Windows y Linux). Mi aplicación se centra en la conexión entre la aplicación cliente y la plataforma web, y uno no puede existir sin el otro, por lo que he tenido que comprobar que esta relación es posible.

Sobre todo, he tenido que probar las APIs de cada sistema operativo para ver si se podían acceder a los recursos requeridos sin comprometer a la seguridad del usuario (sin permisos de administrador, sin configuración adicional, sin drivers, etc.).

Esta prueba ha resultado favorable, por lo que he decidido los siguientes pasos a realizar:

- Diseñar una interfaz básica sin lógica por detrás.
- Crear los modelos de la base de datos.
- Crear la lógica del servidor, crear la API que realiza los cambios en la base de datos.
- Conectar el frontend y el backend a través de la API.
- Probar la autenticación con JWT (JSON Web Tokens) para la conexión entre el cliente y la plataforma web.
- Pruebas de seguridad (Inyección SQL, XSS, posibles rutas sin autenticación, etc.)
- Despliegue a las diferentes plataformas (Netlify y AWS)
- Documentación (memoria del proyecto, despliegue, etc.)

6. Fuentes

Para la mayoría de la información que he necesitado, estoy necesitando y voy a necesitar, he encontrado las paginas oficiales de las librerías/frameworks/etcétera. muy útiles.

React https://es.reactjs.org/ https://reactjs.org/tutorial/tutorial.html

Qt https://www.qt.io/

TypeORM http://typeorm.io/

ExpressJS https://expressjs.com/es/

NodeJS https://nodejs.org/es/

TypeScript https://www.typescriptlang.org/

C++ https://www.learncpp.com/

PostgreSQL https://www.postgresql.org/

AWS EC2 https://aws.amazon.com/es/ec2/

Netlify https://www.netlify.com/

TailwindCSS https://tailwindcss.com/

Notion https://www.notion.so/

GitHub Student Pack (estudiantes) https://education.github.com/pack

Ventana actual Linux https://github.com/UltimateHackingKeyboard/current-window-linux/blob/master/get-current-window.c

Ventana actual Windows https://docs.microsoft.com/en-us/windows/win32/api/winuser/nf-winuser-getforegroundwindow