**CHAT ONLINE CON** **CLIENTE WEB**

Autor: Dylan Hurtado López

**Proyecto de desarrollo de aplicaciones multiplataforma**

**CHAT ONLINE CON CLIENTE WEB**

CICLO FORMATIVO DE GRADO SUPERIOR

**Desarrollo de Aplicaciones Multiplataforma (IFCS02)**

**Curso 2021-22**



Autor/a/es:

**Dylan Hurtado López**

Tutor/a:

**Jose Luis González Sánchez**

Departamento de Informática y Comunicaciones

**I.E.S. Luis Vives**

# INTRODUCCIÓN

Spock se aburre mucho y se montado un gallinero para fastidiar a los alumnos de DAM, un momento esto no era, perdón que se me reviven los traumas.

Spock siempre con su rollo de *“Larga vida y prosperidad”.*

**

Bueno a lo que iba…

Hace mucho pero mucho tiempo donde vosotros, profesores, erais asquerosamente jóvenes como muchos de nosotros ahora. Por aquel tiempo probablemente ni Instagram era conocido. Por eso la manera de socializar a parte de Tuenti, donde la gente solía subir las fotos de las que actualmente sienten vergüenza ajena.

Tenías Windows Live Messenger una aplicación de mensajería que recuerdo ligeramente usar a lo largo de mi infancia. Seguro que alguno de vosotros conoció con esto a su exnovia o algo así. Yo en cambio lo usaba para hablar después de clase con mis amigos del colegio.

Bueno en fin, todo eso se acabó de un día para otro, esta aplicación se esfumo y renació en forma de Skype. Recuerdo con nostalgia aquella época y es por ello que me quiero redimir y hacer mi propio chat en honor al Messenger.

La aplicación trata de imitar lo que conocemos actualmente como aplicación de chat. Una lista de contactos, con los que puedes intercambiar mensajes a tiempo real. Con su parte cliente y su parte servidor, una forma de acceder a los datos ya sea API Rest o GraphQL. Me voy a basar en aspectos de: Discord, WhatsApp y Telegram.

## OBJETIVO

Con este proyecto se pretende tanto aprender nuevas tecnologías orientadas al desarrollo web como desenvolverse en un territorio desconocido y salir victorioso.

Asentar los conocimientos aprendidos para un cercano futuro laboral.

Tener un proyecto completo para que vean que me todo mi trabajo en serio.

Y por encima de todo esto demostrarme que puedo hacerlo.

## ALCANCE

Establecer un almacenamiento solido en la base de datos.

Comunicación sincronizada entre cliente y servidor para que la aplicación funcione.

Envió de mensajes, estado del usuario(offline/online), si esta escribiendo el usuario, si hace focus en el input y demás eventos de chat con ayuda de los websockets.

Portear el chat web a un cliente de escritorio gracias a la librería electronjs

Parte servidor: Escucha las peticiones de clientes y las almacenará en una base de datos.

Parte cliente: Web con funcionalidad de registrarse y logarse; enviar y recibir mensajes de un chat.

Queda fuera del alcance: Video-llamadas, envió de audios/imágenes/videos, Grupos de varios contactos. Y cliente móvil.

## JUSTIFICACIÓN

El porque de los objetivos que he propuesto viene delimitado de los aspectos vistos en los puntos anteriores. Teniendo en cuenta que yo cuando entre al segundo curso del grado no tenia claro que rama del desarrollo me gustaba mas. Este año me he encontrado con muchos aspectos de la programación que jamás pensé que iba a conocer, pero que al fin de al cabo me ha gustado conocer. A lo largo del año descubrí que me llamaba más la atención la parte de la interfaz, que se conoce como el front-end. Pero tampoco me disgusta la parte de atrás (back-end).

Me fui metiendo mas en ese mundo y descubrí que esta reinado por la web, JavaScript había conseguido derrumbar a todos sus oponentes y proclamarse campeón en el mundo del front-end. Se han implantado tres frameworks que actúan por encima de este: Angular, React y Vue.js.

Llegado a este punto empece a aprender javascript y reorientarme hasta esta rama del desarrollo. Por otro lado hemos desarrollado aplicaciones de escritorio con Java FX y la verdad lo he disfrutado mucho. Las aplicaciones de escritorio es algo que me llama también bastante la atención.

Estuve mirando como estaban hechas algunas de estas aplicaciones y me lleve unas cuantas sorpresas porque muchas de ellas como Visual Studio Code, Discord, Telegram, etc.

Combinaban los dos mundos que a mi me gustaban que son el desarrollo web y las aplicaciones de escritorio. Tras esta reflexión ya sabia que quería hacer mi proyecto con un framework de JavaScript y electron. Electron es un framework de código abierto conocido porque permite el desarrollo de aplicaciones graficas de escritorio usando componentes del lado cliente y del servidor originalmente desarrolladas para aplicaciones web. Hemos hablado de cliente y de servidor, pero a mi no es hasta este ultimo curso que me ha quedado claro estos conceptos. Un día de estos en clase para explicarlo Jose Luis, mi profesor, también conocido por algunos como la pesadilla de todas las noches. Hizo una especie de chat con java y swing. Y como estábamos en una red LAN pudimos conectarnos todos y poner unos cuantos mensajes :) . De verdad que estoy muy agradecido por lo vivido en este ciclo.

Todo esto fue muy determinante para decidirme por este proyecto.

# IMPLEMENTACIÓN

## ANÁLISIS DE LA APLICACIÓN

En este análisis veremos las tecnologías que escogí y las que no para hacer este proyecto. Y el porque de todo esto.

Empecemos por la base de datos. Hay de dos tipos:

**Base de datos relacionales / SQL:**

Las bases de datos SQL son un tipo de base de datos que almacena y proporciona acceso a puntos de datos relacionados entre sí. Estas como su nombre indica, se basan en un modelo relacional.

Este modelo es una forma intuitiva y directa de representar datos en tablas.

Esta estandarizado como lenguaje de consulta SQL.

Algunos ejemplos serían: mySQL,PostgreSQL y MariaDB entre otros.

**Base de datos no Relaciones/ noSQL:.**

as bases de datos noSQL(a veces llamada “Bases de datos no solo SQL”) es una amplia clase de sistemas de gestión de bases de datos que difieren del modelo clásico SGBDR(Sistema de Gestión de Bases de Datos Relaciones), en aspectos importantes siendo mas destacados porque no usan SQL como lenguaje principal de consultas. Este tipo almacena información como un documento, generalmente utilizando para ello una estructura simple como son JSON o XML y donde se utiliza una clave única para cada registro.

Algunos ejemplos serian: MongoDB, Cassandra, Neo4J, Redis, HyperTable, Riak entre otras.

En este caso me decidí por MongoDB porque ademas de haberlo trabajado más en clase. En el desarrollo web es muy popular y para embeber los mensajes en la entidad usuario me venia de perlas. Creo que habría sido un error no escoger esta base de datos pues trabajando con JavaScript / TypeScript el formato del documento coincide con el de las clases. En mi opinión pienso que hice lo correcto.

¿ Por que no elegí una de las bases de datos relacionales ?

Bueno aquí hay dos partes por las que no, una personal y otra de lógica. La personal es que me notaba muy apegado al modelo relacional y me notaba miedo.

Entonces decidí de romper con este modelo y enfrentarme al por venir de algo diferente. La parte lógica es que tras una investigación las aplicaciones sociales suelen usar las base de datos no Relacionales / noSQL.

Antes hemos hablado de la palabra framework pero no hemos dicho que es.

Bueno un framework no es mas que un esquema o marco de trabajo que ofrece una estructura base para elaborar un proyecto con objetivos específicos, una especie de plantilla que sirve como punto de partida para la organización y desarrollo de software.

Respecto a la parte del back-end habia muchísimas opciones, las que yo tenia en mente eran:

SpringBoot (Back-end hecho con Java):

Es un framework para el desarrollo de aplicaciones y contenedor de inversión de control, de código de abierto para Java.

Esta es la opción mas conservadora pues es el framework con el que hemos trabajado en clase y me sentía ya bastante cómodo con java.

Node.js (Back-end hecho con JavaScript/TypeScript):

Es un entorno en tiempo de ejcución multiplataforma, de código abierto, mediante el cual el lenguaje de programación JavaScript dio el salto fuera de los navegadores para adentrarse en la capa del servidor(pero no limitandose solo a ello). Este fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables, como por ejemplo, servidores web.

Esto opción me llamo bastante cuando vimos un ejemplo en clase porque con 3 lineas de código montabas un servidor.

Django (Back-end hecho con Python):

Es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo-vista-controlador (MVC).

Fue desarrollado en origen para gestionar varias páginas orientas a noticias de la World Copony de Lawrence, Kansas, y fue liberada al publico bajo una licencia BSD.

De este framework me había hablado en clase y según he visto es bastante denso. Mi profesor Javier Palacios. Lo describía como el “SpringBoot de Python”.

Flask (Back-end hecho con Python):

Es un framework minimalista escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo de líneas de código. Esta bajo una licencia BSD

Al final me decidí por Node.js debido en parte a la influencia que me nacía de alguno de mis profesores y porque desarrolle un el un par de ejemplos antes de decidirme y me gustó bastante. Bueno y porque por entonces ya era mucho del team JavaScript / TypeScript.

¿ Porque no elegí otro back-end ?

Bueno a pesar de ser muy determinante con que tecnología trabajes la verdad que en el back-end estaba más abierto a cualquier cosa.

Pero pensándolo bien escogí Node.Js y no otra cosa por que si se me gustaba y le ponía ganas me volvería Full-Stack JavaScript. Esto ultimo me motivaba muchísimo y fue en definitiva porque descarte las otras opciones.

La parte del front-end en mi cabeza solo se repetían 3 opciones:

Angular:

Es un framework para aplicaciones web desarrollado en TypeScript, de código abierto,mantenido por Google, que utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en un navegador con capacidad de Modelo Vista Controlador(MVC),en un esfuerzo para hacer el desarrollo y las pruebas sean más fáciles.

De este framework se habla mucho ya que en España esta muy integrado en el mercado laboral. Como dice mi profesor Jose Luis Gonzalez: “Angular + SpringBoot es el pack español”.

React:

Tambien llamado(React.js) es una librería JavaScript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre. Tiene la caracteristica de tener lo que se llama un Virtual DOM que en lugar de confiar solamente en el DOM del navegador. Este Virtual DOM es comparado con el DOM que conocemos utilizando como resultado para determinar como actualizar eficientemente el DOM del navegador.

Este framework me atraía porque lo vi utlizar en el canal de midudev.

Por otro lado a mi profesor Javier Palacios no le hace mucha gracia ya que esta hecho por Facebook. Bueno y como Javier diga lo que de verdad piensa sobre la compañía esta noche le meten preso.

Vue.js:

Comúnmente conocido como Vue es un framework JavaScript de código abierto para la construccion de aplicaciones de una sola página. Es el framework mas joven respecto a los anteriores nombrados y como consecuencia tiene como dicen a los que les gusta este framework lo mejor de los dos mundos. Que decir mas sobre este framework, mi profesor Jose Luis lo tiene siempre en la boca, es su favorito y le reza todas las noches.

Posteriormente descubrí otro framework que no llegaba al nivel de estos tres pero que se estaba destacando:

Svelte:

es un compilador front-end gratuito y de código abierto, no se incluyen referencias al framework en su lugar la compilación de una aplicación Svelte genera código el cual modifica el DOM, esto hace que se reduzcan los tamaños de los archivos transferidos y así proporcionar un mejor arranque y rendimiento en tiempo de ejecución al cliente. Esta escrito en TypeScript y se convierte el código de la aplicación a JavaScript para el cliente.

Este framework me parece muy interesante, y le tengo fe pero aun no esta tan implantado como los anteriores.

En esta dura elección parecida a cuando te daban a elegir tu pokemon inicial.

A ver que elijo Charmander, Squirtle o Bulbasaur pues asi estaba yo con los frameworks que encima tienen los mismo colores que los pokemon.

Mi decisión vino muy de la mano de la noticia de las prácticas porque iba a trabajar con Angular.

Bueno pues llegados a este punto ya no tenía dudas ,el que mucho abarca poco aprieta, a fuego con Angular.

Llego la hora de testear un poco el back-end y no sabia que tecnología elegir así que inicie una investigación y recabe las siguientes:

Jest:

Es un framework JavaScript de testing mantenido por Meta (Facebook).

La mayor ventaja que tiene es que no necesitas de mucha configuracion para primeros usuarios usando frameworks de testing.

Mocha:

Es un framework JavaScript de testing bastante extendido a día de hoy.

Jasmine:

Es un framework de testing open-source para JavaScript. Que se puede utilizar en cualquier plataforma permita JavaScript. Tiene una sintaxis fácil de leer. Y es una fuerte influencia en otros framework de pruebas unitarias como ScrewUnit, JSSpec, Jspec y Rspec.

Al final elegí Jest porque quería algo muy simple y rápido para testear. Quizá si tuviera otro proyecto me anime a probar Jasmine porque me pareció el mejor de los tres.

Mientras desarrollaba el fron-end en Angular me di cuenta que necesitaba una de esas librerías que ofrecen componentes porque me iba a hacer la vida mucho más fácil.

Tras mi búsqueda estos son los que más me convencían:

Angular Material:

Es una colección de componentes basados en el diseño Material Design que esta muy de moda basado en Google y esta librería lo pone a nuestro alcance para usar con Angular.

Tiene un conjunto amplio de componentes listos para usarse

Ionic:

Aunque no sea una librería de Angular como tal, como su sintaxis en TypeScript se complementa a la perfección. Es perfecta para crear aplicaciones hibridas. Esto es una ventaja ya que el desarrollo es mas rapido. En definitiva desarrollas una vez para varios dispositivos. Es el sueño de cualquier desarrollador.

PrimeNG:

Colección de muchos componentes preparados también para móvil. Tiene ciertos temas que puedes escoger para personalizar la apariencia de todos los componentes. Tiene desde componente de visualización de datos(tablas,graficas,estadísticas,etc) a muchos otros tipos de paneles,botones,menús e incluso un componente drag and drop.

Al final elegí Angular Material porque me gustaba más el estilo de los componentes.

¿ Por qué no elegiste Ionic, PrimeNG u otro ?

La verdad que ya estaba cansado de tanto buscar y quería empezar a desarrollar de una vez los componentes. En las practicas utilizaban Angular Material y ya me sabía la API y todos los componentes.

Podía haber cambiado no es muy dificil aprender si esta bien documentado pero como tenia ganas de desarrollar pues me case con Angular Material.

Además para mi los componentes para movilés eran secundarios porque desarrollaría el cliente móvil si me sobraba tiempo y me aumentaban las ganas de abrir el Android Studio y sufrir la compilacion de Gradle, etc.

PD:No pasó, ninguna de las condiciones se cumplieron.

## DISEÑO

…

## IMPLEMENTACIÓN

…

## IMPLANTACIÓN

## EMPAQUETADO

He realizado un empaquetado con Docker para poder despegarlo en cualquier parte.

Docker es un proyecto de código abierto que automatiza el despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y automatización de visualización de aplicaciones en múltiples sistemas operativos.

Gracias a Docker cree una imagen tanto de mi back-end hecho en Node.js, como de mi front-end en Angular, la cual la cargo en un contenedor software.

## DESPLEGACIÓN

## DOCUMENTACIÓN

… APIDOC

# RESULTADOS Y DISCUSIÓN

…

# TRABAJO FUTURO (Opcional)

Llegamos a este apartado donde repasamos los conceptos que no me ha dado tiempo a desarrollar. Los cuento para tener una lista ordenada yo o como dijeron mis profesores tirar la pelota al siguiente quien sabe quizá salgo Cosmos 2 algún día.

Aqui dejo la lista:

* Adjuntar y enviar archivos / videos / imágenes
* Usar para lo anterior la tenologia conocida como drag&drop (arrastrar y soltar).
* Envio de audios
* Videollamadas / Llamadas
* Crud completo tanto de los chat como mensajes
* Rediseñar la aplicación para incluir grupos en el chat
* Busqueda entre los mensajes
* Cifrado de mensajes de extremo a extremo
* Cliente movil (en condiciones optimas nativo con Jetpack Compose y sino modificar el cliente web y poner un webview, quizá también con los componentes de Ionic)
* Añadir internalización de idomas.
* Que puedas logearte con facebook o con google.
* Notificaciones
* Sonidos

Si estas pensando en continuar mi proyecto contactame a traves de correo.

Lo puedes encontrar en mi GitHub.

GitHub: <https://github.com/DyLaNHurtado>

# CONCLUSIONES

…

# BIBLIOGRAFÍA

…

# ANEXOS

…

## ANEXO 1

## ANEXO 2…