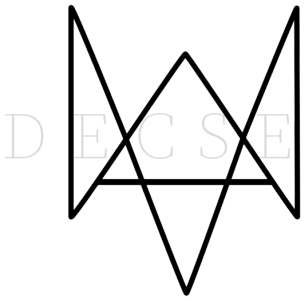
DECSEC

Alberto Hidalgo Olivera

Proyecto Final DAW Curso 2023-2024

Índice:



1. **Identificación de las necesidades del proyecto.**
2. **Breve análisis/comparativa con las alternativas del mercado**
3. **Justificación del proyecto**
4. **Stack tecnológico.**
5. **Esquema de la BD**
6. **Prototipo en Figma o similar**
7. **Definición API REST publicación servicios**
8. **Manual de despliegue**
9. **Conclusiones y Postmortem**

Identificación de las necesidades del proyecto

Esta aplicación como cualquier red social busca la interacción significativa entre los usuarios, priorizando un diseño intuitivo y garantizando altos estándares de privacidad y protección de datos. Esta red social responderá a la creciente demanda de experiencias digitales seguras y adaptadas, promoviendo un espacio inclusivo donde los usuarios puedan compartir contenido, interactuar, y crear conexiones dentro de un entorno optimizado para sus necesidades específicas.

Breve análisis/comparativa con las alternativas del mercado

Como explico en el siguiente punto de la documentación este proyecto no busca ser una aplicación diferente a las que existen actualmente en el mercado. Simplemente me pareció un buen concepto para poner a prueba lo aprendido durante el curso y abarcar lo máximo posible.

Si pudiese diferenciar mi aplicación de otras sería la abusiva cantidad de información que nos proporcionan las redes sociales hoy en día, mostrándonos contenido que o no nos interesa o contenido promocionado. En esta aplicación solo veremos las publicaciones de los usuarios que deseemos tener agregados.

Justificación del proyecto

El concepto del proyecto es el mismo que realicé el año pasado para el anterior grado, pero teniendo en cuenta las cosas aprendidas en este curso quise repetir el proyecto para poder implementarlas y hacer un mejor proyecto con ideas y tecnologías que no añadí en su día. Las principales motivaciones son aplicar lo aprendido, aumentar el alcance de la anterior aplicación que desarrolle y aprender en el proceso.

Stack tecnológico.

- MySql (BBDD):
    - Mi decisión al usar sql se basa en que permite manejar grandes volúmenes de datos de manera eficiente y la capacidad de almacenar imágenes, esto es prioritario en un sistema en el que se necesite manejar una gran cantidad de datos, en nuestro caso una gran cantidad de publicaciones con imágenes.
- Spring Boot (Backend):
    - Mi decisión al usar Spring Boot se basa en la integración sencilla con MySql, las características de seguridad gracias a la configuración de web security basada en tokens y el uso de Java por su programación orientada a objetos.

- Angular (Frontend):
    - Mi decisión al usar Angular se basa en la capacidad de usar componentes para poder reutilizar código de forma eficiente y tiene la ventaja sobre otros modelos basados en JavaScript, ya que al usar TypeScript permite el tipado de los objetos y permite un entendimiento más rápido de los componentes generados previamente.
- PrimeNG (Frontend UI):
    - PrimeNG es una biblioteca de componentes de interfaz de usuario (UI) desarrollada para Angular. Ofrece una amplia colección de componentes reutilizables, como tablas, formularios, menús, gráficos y calendarios, con soporte para personalización mediante temas predefinidos y estilos CSS.

Definición API REST publicación servicios.

`/api/v1/auth/signup`	**POST**	Crear usuario
`/api/v1/auth/signin`	**POST**	Iniciar Sesión
`/api/v1/auth/validar-email`	**GET**	Valida el email el formulario de creación de usuario
`/api/v1/comentarios`	**GET**	Lista todos los comentarios
`/api/v1/comentarios/{id}`	**GET**	Lista los comentarios de una publicación
`/api/v1/comentarios/{id}`	**POST**	Crear Comentario
`/api/v1/comentarios/{id}`	**PATCH**	Actualizar Comentario
`/api/v1/comentarios/{id}`	**DELETE**	Borrar Comentario
`/api/v1/peticiones`	**GET**	Devuelve una petición específica o todas las peticiones del usuario dependiendo el parámetros ‘email’.
`/api/v1/peticiones/amigos`	**GET**	Lista los amigos del usuario que realiza la petición
`/api/v1/peticiones`	**POST**	Crea una petición
`/api/v1/peticiones/cambiarEstado`	**PUT**	Cambia el estado de una petición
`/api/v1/publicaciones`	**GET**	Mapea una solicitud GET para listar publicaciones
`/api/v1/publicaciones/publicacionesFeed`	**GET**	Mapea una solicitud GET para el feed de publicaciones
`/api/v1/publicaciones/publicacionesMeGusta`	**GET**	Mapea una solicitud GET para las publicaciones con "me gusta"

`/api/v1/publicaciones`	**POST**	Mapea una solicitud POST para crear una publicación
`/api/v1/publicaciones/{id}`	**PATCH**	Mapea una solicitud PATCH para actualizar parcialmente una publicación
`/api/v1/publicaciones/{id}`	**DELETE**	Mapea una solicitud DELETE para borrar una publicación
`/api/v1/publicaciones/darmegusta/{id}`	**POST**	Mapea una solicitud POST para dar "me gusta" a una publicación
`/api/v1/publicaciones/quitarmegusta/{id}`	**POST**	Mapea una solicitud POST para quitar "me gusta" a una publicación
`/api/v1/users`	**GET**	Mapea una solicitud GET para listar los usuarios
`/api/v1/users/search`	**GET**	Mapea una solicitud GET para búsqueda
`/api/v1/users/{id}`	**GET**	Mapea una solicitud GET para un usuario específico
`/api/v1/users/token`	**GET**	Mapea una solicitud GET para obtener un usuario por token
`/api/v1/users`	**DELETE**	Mapea una solicitud DELETE para borrar el usuario que realiza la petición
`/api/v1/users/{email}`	**DELETE**	Mapea una solicitud DELETE para borrar un usuario por email
`/api/v1/users/{id}`	**PATCH**	Mapea una solicitud PATCH para actualizar un usuario específico
`/api/v1/users`	**PATCH**	Mapea una solicitud PATCH para actualizar el usuario que realiza la petición
`/api/v1/users/actualizarMedia`	**PATCH**	Mapea una solicitud PATCH para actualizar la foto y el banner del usuario que realiza la petición
`/api/v1/users/nick/{nick}`	**GET**	Mapea una solicitud GET para obtener un usuario por nick

Despliegue

Para desplegar esta aplicación usaremos docker, un paso previo antes de poder usar el docker-compose es compilar la api.

Una forma sencilla de hacerlo es abriendo una consola en la carpeta raíz del proyecto y ejecutando estos comandos:

- cdsrc-api
- mvnclean
- mvninstall

Tras compilar la api en un jar volveremos a la carpeta raíz y ejecutamos estos comandos:

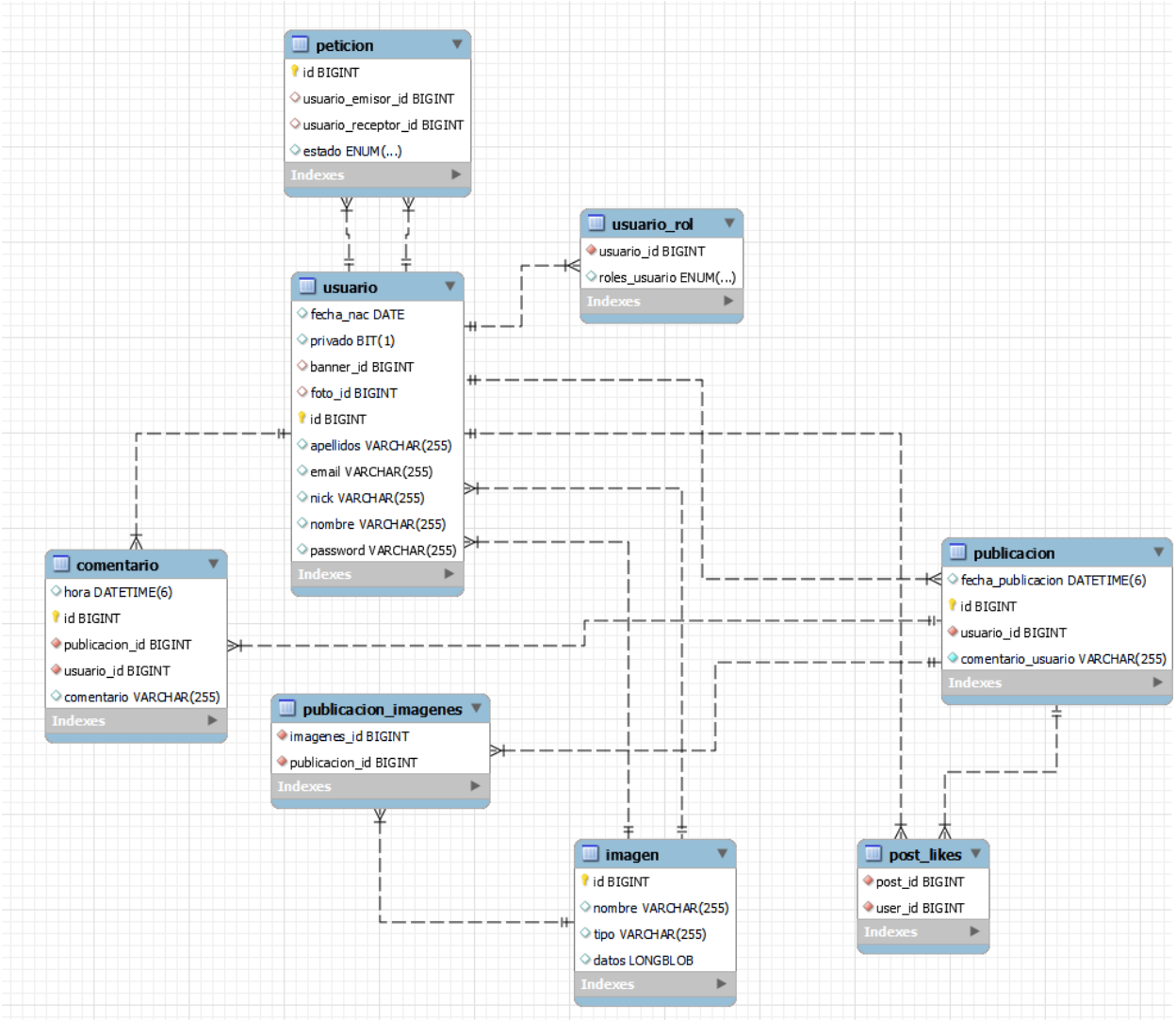
- docker-compose build
- docker-compose up-d

Con esto tendremos desplegada nuestra aplicación en el puerto 4201

Code Blame 40 lines (36 loc) · 673 Bytes

```yaml
version: '3.8'

services:
  sql:
    image: mysql:latest
    ports:
      - "3306:3306"
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: decsec
    networks:
      - local-network

  api:
    build:
      context: ./src/src-api/decsecBackendDaw
      dockerfile: Dockerfile
    ports:
      - "8081:8081"
    depends_on:
      - sql
    networks:
      - local-network

  frontend:
    build:
      context: ./src/src-frontend/DecsecFrontend
      dockerfile: Dockerfile
    ports:
      - "4200:4200"
    volumes:
      - ./:/angular-app
    depends_on:
      - api
    networks:
      - local-network

networks:
  local-network:
    driver: bridge
```

Esquema de la BD



Prototipo en Figma o similar

Dev Mode

https://www.figma.com/design/66cYnXl6OV0fW0svmwgJ2c/TFG?node-id=2019-305&m=dev&t=xR0PMFdXxPyTWGvH-1

Prototipo

https://www.figma.com/proto/66cYnXl6OV0fW0svmwgJ2c/TFG?node-id=2019-305&t=xR0PMFdXxPyTWGvH-1

Conclusiones y Postmortem

A la hora de desarrollar el proyecto he encontrado muchas dificultades a la hora de implementar el sistema de websockets para el chat en tiempo real, eso y la falta de tiempo para hacer más hincapié en ello me ha llevado a la decisión de dejarlo fuera e incluirlo en actualizaciones futuras. También me hubiese gustado darle más protagonismo al administrador creando una serie de reportes que este pudiera ver y considerar si es necesario la eliminación del usuario, publicación o comentario.

En cuanto a las dificultades que me he encontrado la mayoría ha sido con el diseño del Front. Aunque primeng pone a disposición multitud de componentes para el desarrollo del mismo, es verdad que su edición es limitada y tediosa lo que lleva a que no siempre quede un elemento como te gustaria.

Como conclusión final la aplicación tiene mucho margen de mejora y funciones a implementar que por falta de tiempo no he podido incluir pero harían de este proyecto una experiencia más profesional y completa.