RoomEdit

Javier Coronel Ortiz

2022-2023

2º de Desarrollo de Aplicaciones Web

Índice:

INTRODUCCIÓN

El proyecto trata sobre una página web llamada RoomEdit en el que cada usuario puede crear una habitación o sala, las salas son imágenes creadas por los usuarios juntando varias imágenes las cuales están guardadas en un servidor, estas salas pueden ser valoradas y comentadas por otros usuarios, habrá moderadores y administradores los cuales pueden activar y desactivar cuentas y eliminar comentarios y salas, además, los administradores podrán añadir más imágenes al servidor con las que decorar las salas.

IDENTIFICACIÓN DE LAS NECESIDADES DEL PROYECTO

Estas son las diferentes acciones que pueden hacer los usuarios en la aplicación:

- Un usuario puede registrarse.
- Un usuario puede iniciar sesión.
- Un usuario puede cerrar sesión.
- Un usuario puede resetear su contraseña.
- Un usuario no registrado puede ver las salas de otros usuarios y los comentarios de estas pero no puede crear una sala propia.
- Un usuario registrado puede añadir imágenes a su sala.
- Un usuario registrado puede copiar la sala de otra persona y modificarla como la suya propia.
- Un usuario puede comentar salas de otros usuarios.
- Un usuario puede valorar salas de otros usuarios.
- Un usuario puede buscar a otro usuario.
- Un usuario puede reportar a otro usuario.
- Un usuario puede ver los comentarios que otros usuarios han hecho en otras salas.
- Un usuario y un usuario no registrado pueden buscar salas y filtrar por valoración, si la sala ha sido modificada hace poco o por la cantidad de comentarios que tiene la sala.
- Un administrador puede publicar nuevas imágenes con las que los usuarios pueden decorar las salas.
- Un moderador o un administrador puede eliminar un comentario o sala.
- Un moderador o un administrador puede activar o desactivar la cuenta de un usuario.

COMPARATIVA CON LAS ALTERNATIVAS DEL MERCADO

Esta aplicación no tiene alternativas en el mercado como tal, pero existen algunas aplicaciones que tienen alguna característica que se asemeja a la base del proyecto, aunque este parecido es que se pueden poner imágenes en documentos dentro de la página, pero estas imágenes son dadas por el usuario y no por la aplicación.

JUSTIFICACIÓN DEL PROYECTO

Lo que me ha llevado a pensar en esta idea de proyecto es que este proyecto es una parte de una idea de un juego que tengo el cual sería complicado de crear, pero se podría hacer esta parte en concreto del juego para la parte principal de esta aplicación.

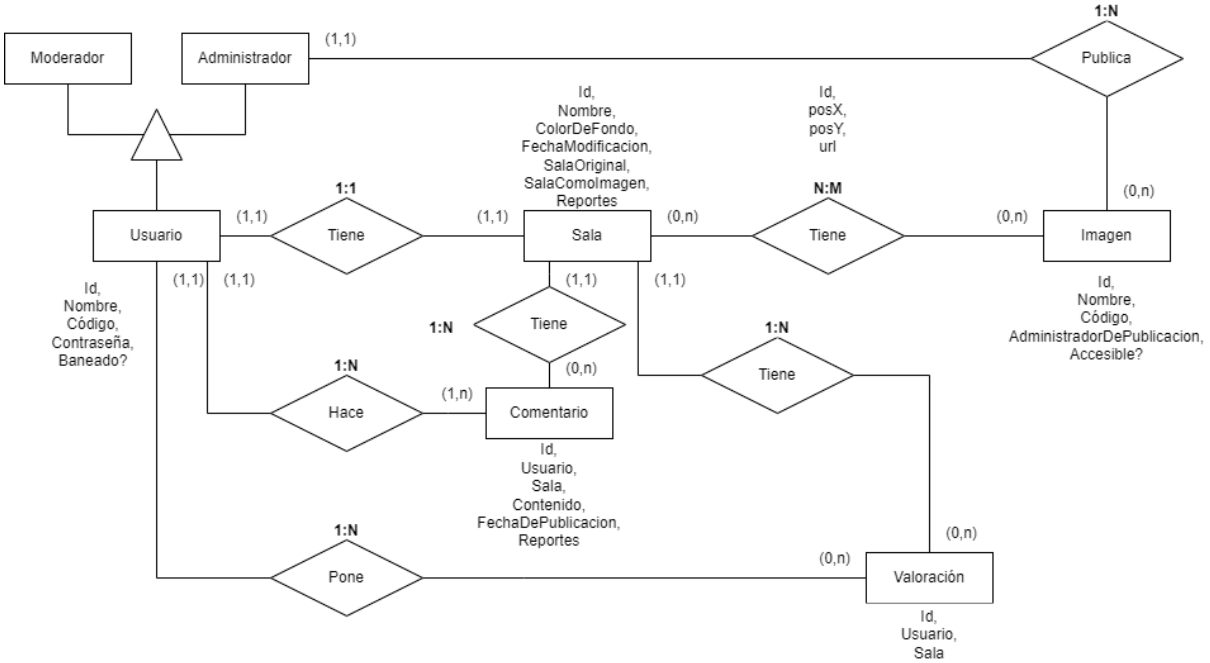
STACK TECNOLÓGICO

Para la creación de esta aplicación se han usado las siguientes tecnologías:

- Node.js versión 16.18.0: Utilizada para el back-end, junto con Express y Multer (para guardar diferentes imágenes).
- Angular versión 15.2.7: Utilizada para el front-end, junto con Bootstrap y Material.
- MongoDB versión 6.0.11: Es utilizada como base de datos.
- Unity versión 2019.4.20f1: Esta tecnología ha servido para poder mostrar e interactuar con las imágenes de las salas, la he usado porque ya había trabajado con Unity anteriormente.

MODELO DE DATOS

El diagrama de la base de datos es el siguiente:



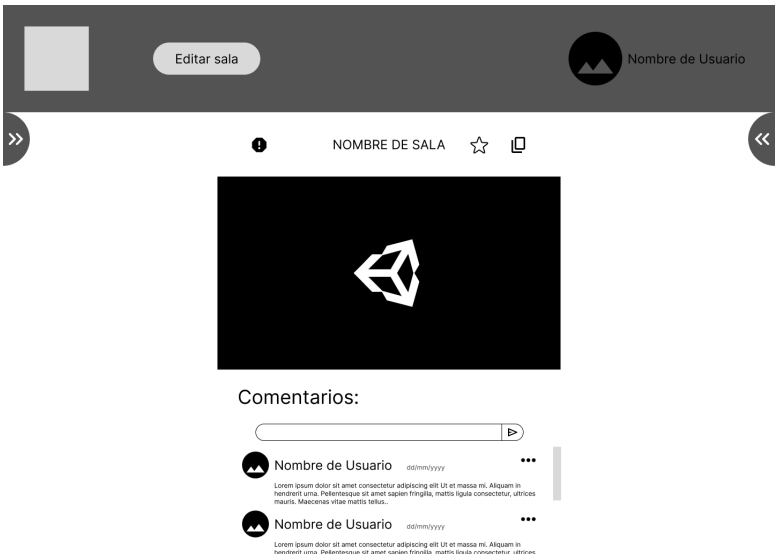
En la entidad de usuarios contiene un nombre de ese usuario, un código, una contraseña y un estado para comprobar si está baneado o no, un usuario puede ser un moderador un administrador o ninguna de estas dos cosas, todos los usuarios tienen una sala que tiene nombre, un color, la fecha en la que se modificó por última vez, la sala original en la que se basa, el nombre del archivo png en el que está guardada la sala, cuantas veces a sido la sala reportada y también tiene un conjunto de posiciones y imágenes, estas imágenes son obtenidas la entidad de imágenes la cual contiene un nombre, un código, un estado para comprobar si es accesible o no y el id de un administrador.

La entidad de comentarios tiene el id del usuario que lo ha puesto, el id de la sala en la que se pone, el contenido del comentario, la fecha en la que se publicó, y el número de reportes, la entidad de valoraciones también tiene el id de un usuario y de una sala, un usuario puede poner tantos comentarios y valoraciones como quiera y una sala puede tener cualquier cantidad de comentarios y valoraciones.

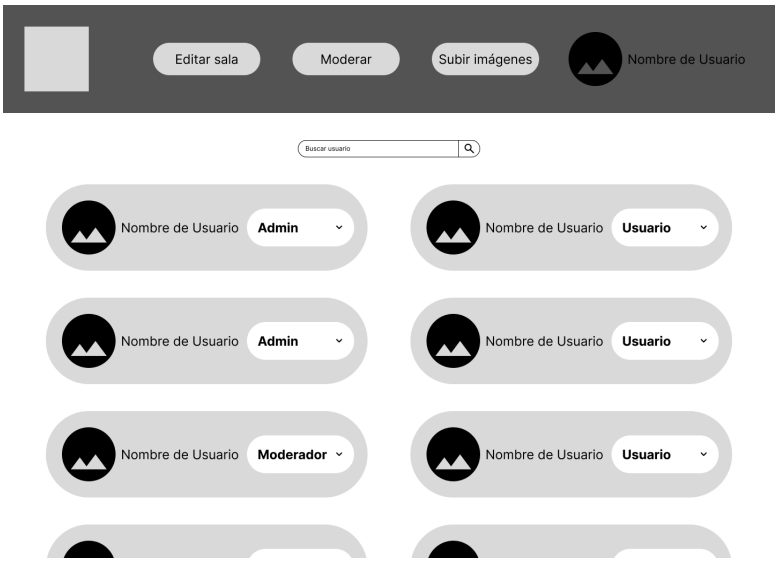
PROTOTIPO DE LA APLICACIÓN WEB

Estas son algunas de las páginas de la aplicación, se puede encontrar el proyecto de figma aquí o aquí:

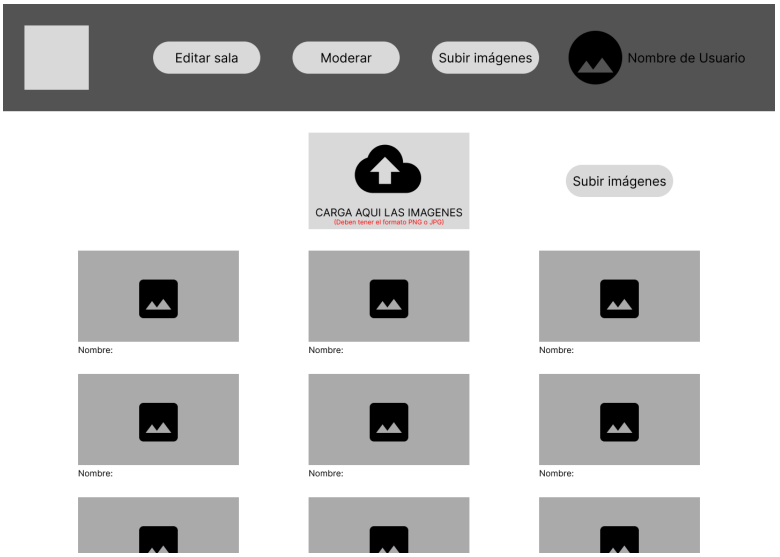
Inicio de la aplicación:



Menú del administrador:



Menú para subir imagenes:



Menú de moderación:

DEFINICIÓN API REST

Estas son las rutas de la aplicación, también se puede encontrar aquí:

Users			
Ruta	**Método**	**Descripción**	**Comentarios**
/users	POST	Inserta un nuevo usuario en la base de datos.	Además, inserta una nueva sala para ese usuario.
/users/signin	POST	Comprueba el nombre y contraseña introducidos en la base de datos.	
/users	GET	Devuelve a todos los usuarios.	
/users/searchByName/:name	GET	Busca a todos los usuarios que tengan en el nombre el dato pasado.	
/users/searchByCode/:code	GET	Busca a un usuario por su código.	
/users/changeTypeOfUser	PUT	Actualiza el tipo de usuario de un usuario.	
/users/changeBanningOfUser	PUT	Actualiza el estado de baneo de un usuario.	
/users/changePassword	PUT	Actualiza la contraseña de un usuario.	

Rooms			
Ruta	**Método**	**Descripción**	**Comentarios**
/rooms	GET	Devuelve todas las salas por fecha de creación.	
/rooms/getAllDataByUser/:name	GET	Devuelve una sala a partir del nombre de un usuario.	
/rooms/searchByUser/:name	GET	Devuelve el id de la sala a partir del nombre del usuario.	
/rooms/userRoom/:id	GET	Devuelve las imagenes y el color de una sala.	
/rooms/sortByValorations	GET	Devuelve todas las salas por su cantidad de valoraciones.	
/rooms/sortByComments	GET	Devuelve todas las salas por su cantidad de comentarios.	
/rooms/reportedRooms	GET	Devuelve todas las salas reportadas.	
/rooms/updateRoom	POST	Actualiza una sala.	
/rooms/copyRoom	PUT	Copia la sala de un usuario en la sala de otro usuario.	
/rooms/renameRoom	PUT	Cambia el nombre de una sala.	
/rooms/reportRoom	PUT	Reporta una sala.	
/rooms/unReportRoom	PUT	Elimina los reportes de una sala.	
/rooms	DELETE	Elimina datos de una sala de la base de datos.	No elimina la sala como tal, solo cambia datos de esta.

Images			
Ruta	**Método**	**Descripción**	**Comentarios**
/images	GET	Obtiene todas las imágenes de la base de datos.	
/images/names	GET	Obtiene los nombres de las imágenes de la base de datos.	
/images	POST	Inserta una imagen en la base de datos.	
/images/changeAccess	PUT	Actualiza el estado de acceso a una imagen.	

Valorations			
Ruta	**Método**	**Descripción**	**Comentarios**
/valorations	POST	Inserta una nueva valoración en la base de datos.	
/valorations/findValoration/:name/:room	GET	Obtiene una valoración.	
/valorations/findByUser/:user	GET	Obtiene las valoraciones de un usuario.	
/valorations/findByRoom/:room	GET	Obtiene las valoraciones de una sala.	
/valorations	DELETE	Elimina una valoración de la base de datos.	

Comments			
Ruta	**Método**	**Descripción**	**Comentarios**
/comments/roomid/:id	GET	Obtiene todos los comentarios de una sala.	
/comments/userid/:id	GET	Obtiene todos los comentarios de un usuario.	
/comments/roomofuser/:name	GET	Obtiene todos los comentarios de una sala.	
/comments/reportedComments	GET	Obtiene los comentarios que han sido reportados.	
/comments/reportComment	PUT	Reporta un comentario.	
/comments/unReportComment	PUT	Elimina los reportes de un comentario.	
/comments	POST	Inserta un nuevo comentario en la base de datos.	
/comments	DELETE	Elimina un comentario de la base de datos.	

MANUAL DE DESPLIEGUE

Antes de empezar el despliegue de la aplicación, es necesario tener instalado docker.

Tras instalar docker, se deberá de configurar la variable de entorno, para esto tendremos que crear el archivo .env dentro de la carpeta src-despliegue del proyecto, y dentro de este pondremos la variable `DB_URI`, que contendrá la url donde está la base de datos:

Tras esto ejecutaremos el siguiente comando desde la carpeta src-despliegue: `docker compose -f RoomEdit.yaml up`:

```
PS D:\plantilla_proyecto_iesalixar\src-despliegue> docker compose -f RoomEdit.yaml up
[+] Running 2/0
 - Container src-despliegue-api-1       Created                                                                    0.0s
 - Container src-despliegue-frontend-1  Created                                                                    0.0s
Attaching to src-despliegue-api-1, src-despliegue-frontend-1
src-despliegue-frontend-1  | /docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
src-despliegue-frontend-1  | /docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
src-despliegue-frontend-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
src-despliegue-frontend-1  | 10-listen-on-ipv6-by-default.sh: info: IPv6 listen already enabled
src-despliegue-frontend-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
src-despliegue-frontend-1  | /docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
src-despliegue-frontend-1  | /docker-entrypoint.sh: Configuration complete; ready for start up
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: using the "epoll" event method
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: nginx/1.23.3
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: built by gcc 12.2.1 20220924 (Alpine 12.2.1_git20220924-r4)
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: OS: Linux 5.10.16.3-microsoft-standard-WSL2
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker processes
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 22
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 23
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 24
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 25
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 26
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 27
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 28
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 29
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 30
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 31
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 32
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 33
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 34
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 35
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 36
src-despliegue-frontend-1  | 2023/12/03 19:51:57 [notice] 1#1: start worker process 37
src-despliegue-api-1       |
src-despliegue-api-1       | > src-api@0.0.0 start
src-despliegue-api-1       | > node ./bin/www
src-despliegue-api-1       |
src-despliegue-api-1       | connection successful
```

Y así estaría la aplicación ya desplegada, podremos acceder a un usuario con acceso de administrador con el nombre admin y la contraseña admin.

Algunas cosas a tener en cuenta: El puerto de la parte del backend es el 5000, el del frontend el 80, en el docker compose se crean dos volúmenes para almacenar las imágenes y las imágenes de las salas y el frontend se ejecuta en nginx.

CONCLUSIONES DEL PROYECTO

Haciendo este proyecto he tenido dificultades, he tenido que reducir parte de lo que quería hacer en un principio, he tenido complicaciones a la hora de guardar imágenes, cuando he tenido que mostrar las imágenes en unity, cuando he intentado mostrar la parte de unity en el frontend y más complicaciones, además de que el proyecto tiene muchos puntos en los que podría mejorar, cómo añadir una autenticación por correo electrónico, mejorar la interfaz dentro de unity y de la pagina para subir imágenes o añadir una interfaz para que los usuarios puedan editar sus datos y no sólo cambiar de contraseña.