Obligatorio Programación Web y Mobile

27 de diciembre de 2023

Avelino S., Conde L., Dibueno C., Lopez D., Odera J.

Contenido

1. Descripción del problema

Se debe desarrollar un software para la empresa *Juegos para Todos* que permita proponer juegos/chistes/imágenes para reuniones sociales o fiestas (actividades) en el cual jugadores utilizan sus dispositivos como controladores.

2. Implementación de la solución

2.1. Tecnologías involucradas

La parte web de la aplicación está desarrollada en *Angular* y esta consume una *Application Programming Interface* (*API*) generada en *Node JS*. Para la persistencia de los datos del programa se utiliza la tecnología *MongoDB*.

2.2. Planificación del Frontend

Siguiendo los lineamientos de la consigna, se concibió en un principio un boceto de cómo lucirían las pantallas principales de la aplicación (Ilustración 1).

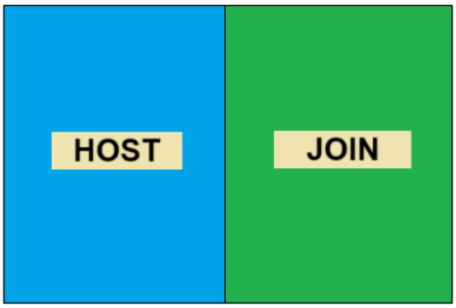

Ilustración 1 - Pantalla principal

El usuario aterrizaría en una pantalla que le permitiría seleccionar si desea crear una sala (*Host*) o ingresar a una (*Join*).

Al cliquear las opciones, estas desplegarían las opciones para realizar mencionadas opciones (Ilustración 2).

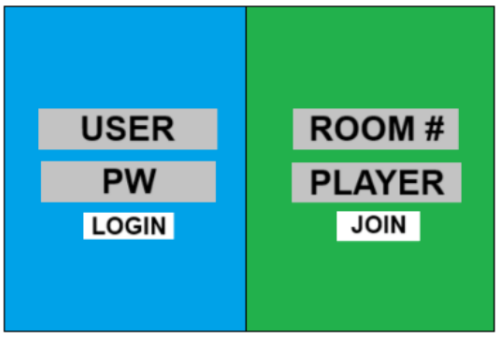

Ilustración 2 - Pantalla principal expandida

Al seleccionar la opción para crear una sala, el usuario ingresaría a una sala de espera en la cual podría visualizar las opciones para crear la sala, los usuarios que se conectan junto a la identificación de la sala y el botón para iniciar la partida (Ilustración 3).

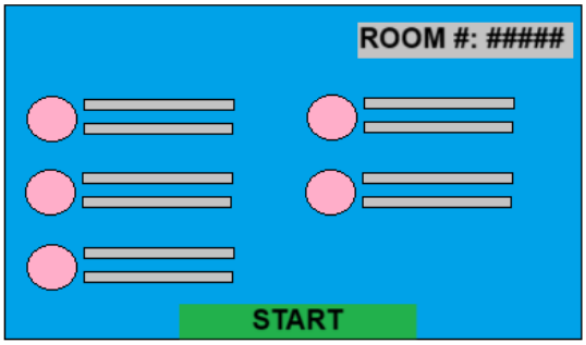

Ilustración 3 – Lobby administrador

Si se selecciona la opción de *Join*, el usuario sería dirigido a una pantalla tipo *Lobby* en la cual podría visualizar a los demás usuarios mientras espera que se inicie la partida (Ilustración 4).

Ilustración 4 - Lobby jugador

2.3. Desarrollo del Backend

Al comenzar a materializar la solución, la *API* de la aplicación se proyectó como tipo *REST*, permitiendo consumir datos desde el lado del cliente mediante consultas estructuradas con formato *JSON*.

Esta modalidad permitía, en las etapas iniciales, enviar al cliente la información que solicitaba.

En los tramos avanzados del proyecto se transformó la solución para que permitiera utilizar la tecnología de *websockets*, habilitando así un canal de comunicación bidireccional (Amazon, 2023).

La implementación de esta tecnología permitió generar en la solución la capacidad de actualizar recursos consumidos por los usuarios utilizando una funcionalidad similar al patrón *Observer* (Refactoring Guru, s.f.).

Para la seguridad de la aplicación se llevó a la práctica los *JSON Web Tokens (JWT)*, abriendo un canal seguro para transmitir las credenciales del usuario y de esta manera autenticar las solicitudes enviadas desde el cliente (JWT, 2023).

A su vez, al almacenarse en el navegador del usuario, se aplicó una mecánica para interceptar las solicitudes al servidor e inyectarlas con el *token*. Esto facilita la verificación del remitente por parte del servidor.

2.4. Base de datos

Hacer persistentes los datos es una necesidad impresa en la consigna y debido a la naturaleza de la información a almacenar se orientó el proyecto a la utilización de *MongoDB*.

La base de datos se encuentra alojada en un servicio gratuito de MongoDB Atlas en la nube.

Esto posibilitó guardar la información de manera flexible, empleando su estructura de documentos que se adecuó a los objetos que se manejan en la aplicación.

2.5. Diagrama de la solución

https://drive.google.com/file/d/1qsGtBsWKWnp6aHauhOt91yqtOxVZoqGC/view

3. Evaluación de la solución

El desarrollo del software mediante la utilización de las tecnologías descriptas anteriormente se realizó con éxito.

Durante el acondicionamiento del proyecto se encontraron diversos inconvenientes al momento de transformar el modelo de *API REST* al de *Websocket* debido a la dificultad que planteó la utilización de las *rooms* (salas) del segundo modelo.

En conclusión, se logró entregar un producto funcional, implementando las necesidades requeridas dentro del plazo necesario.

Referencias

Amazon. (2023). *Websocket API*. Obtenido de Amazon AWS: https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-websocket-api-overview.html?pg=wianapi&cta=websocketapi

JWT. (2023). *Introduction*. Obtenido de https://jwt.io/introduction

Refactoring Guru. (s.f.). *Observer pattern*. Obtenido de https://refactoring.guru/design-patterns/observer