Ramen Stadium

La idea

El plan es crear una aplicación web enfocada al estudio básico del ramen, con información como preparaciones de los aderezos, preparaciones de los caldos básicos etc.

La idea es atraer a los usuarios de la página creando una plataforma donde ellos mismos pueden crear su propio ramen, escogiendo desde el tipo de caldo hasta el último de los aderezos. Cada uno de los componentes que elijan tendrán su propia receta incluida, de forma que haga agradable y sencilla su preparación en casa.

A diferencia

Toda la información irá basada en un libro en formato cómic con diferentes recetas de los componentes de un plato de ramen.

Casos de usos dentro de la página

Estos serán los usuarios de los que se compone mi aplicación:

- **Administrador**
  - Control total sobre los usuarios registrados
  - Visualizar una gráfica con datos de cuantas personas hay registradas y cuantas subscritas al premium
  - Acceso de edición de algunos de los tutoriales
- **Usuario anónimo**
  - Posibilidad de registrarse o loguearse
  - Podrá ver el home pero para empezar tendrá que estar registrado
  - Acceso a la historia sobre el ramen
- **Usuario registrado (Plan gratuito)**
  - Podrá crear ramen pero de forma limitada y no tendrá todos los tutoriales disponibles
  - Podrá postear comentarios (No ofensivos) de su experiencia
  - Todas las funcionalidades del usuario anónimo
  - Podrá ver y postear las combinaciones que más le han gustado y a su vez valorar la de los demás
  - Tiene anuncios.
- **Usuario registrado (Con subscripción)**
  - Podrá ver al completo todos los tutoriales además de que contará con combinaciones de ramen prediseñadas por expertos
  - Todas las funcionalidades del usuario registrado
  - No tiene anuncios
  - Tendra acceso a tutoriales de acompañamiento

Tecnologías pensadas para el proyecto

El lenguaje a usar serán JavaScript para ello tengo pensado usar el stack mern que se compone de un conjunto de tecnologías que son:

- MongoDB:
    - Mongo Atlas
- ExpressJS
- ReactJs
    - Material UI (Librería de estilos)
    - Uso de Hooks
    - React Bootstrap
    - Natural Slider
- NodeJs:
    - AUTH0 (Registro y Login con google)
    - Bycript para encriptar contraseñas

La elección del stack MERN para desarrollar mi proyecto se basa en varias razones fundamentales. Este conjunto de tecnologías proporciona una solución completa y altamente eficiente para construir aplicaciones web modernas y escalables. A continuación, detallo las principales justificaciones:

Flexibilidad y eficiencia: El stack MERN combina tecnologías populares y ampliamente adoptadas en el desarrollo web, lo que brinda una amplia gama de recursos y bibliotecas para acelerar el desarrollo. Cada componente del stack (MongoDB, Express, React y Node.js) está diseñado para ser flexible y escalable, lo que facilita la construcción de aplicaciones adaptables a medida que crecen y evolucionan.

Base de datos robusta: MongoDB, como base de datos NoSQL, es una elección sólida dentro del stack MERN. Proporciona una forma eficiente y flexible de almacenar y consultar datos, especialmente cuando se trata de

estructuras de datos no relacionales o semi-estructuradas. Además, MongoDB es conocida por su escalabilidad horizontal, lo que permite manejar grandes volúmenes de información de manera efectiva.

Backend consistente: Express y Node.js juntos forman un entorno de backend sólido y coherente. Node.js, como entorno de tiempo de ejecución del lado del servidor, es altamente eficiente y permite un manejo asincrónico de solicitudes, lo que garantiza una alta capacidad de respuesta y un rendimiento óptimo. Express, como marco de aplicaciones web, ofrece una estructura flexible y minimalista para el desarrollo del backend, lo que facilita la creación de API robustas y mantenibles.

Experiencia de usuario dinámica: React, como biblioteca de JavaScript para construir interfaces de usuario, es altamente efectiva para crear experiencias de usuario interactivas y en tiempo real. Su enfoque basado en componentes permite una estructura modular y reutilizable, lo que agiliza el desarrollo y el mantenimiento de la interfaz de usuario. Además, React tiene una amplia comunidad de desarrolladores y una gran cantidad de bibliotecas complementarias disponibles, lo que facilita la personalización y la implementación de características avanzadas.

Arquitectura de aplicaciones de página única (SPA): El stack MERN es especialmente adecuado para la construcción de aplicaciones de página única. Al utilizar React para el frontend y Node.js para el backend, podemos crear una aplicación altamente interactiva y fluida, donde las actualizaciones y cambios se reflejan de manera instantánea en la interfaz de usuario sin tener que recargar la página completa. Esto mejora significativamente la experiencia del

usuario, brindando una sensación nativa de aplicación de escritorio.

En resumen, el stack MERN ofrece una combinación poderosa de tecnologías que proporcionan flexibilidad, eficiencia y escalabilidad tanto en el frontend como en el backend. La elección de estas tecnologías se basa en su amplia adopción, su soporte comunitario activo y su capacidad para satisfacer los requisitos de desarrollo modernos. Al utilizar el stack MERN, puedo crear una aplicación web robusta, rápida y altamente interactiva que se adapte a las necesidades de mi proyecto.

Diseño

Ajustaré el diseño a lo que son las imágenes del libro que voy a coger de referencia.

También usaré varios factores visuales a la hora de navegar dentro de la aplicación como pueden ser Barras de carga, spinners de carga, paginación con animación de transición, ventanas emergentes con consejos etc.

Alcance

Tras analizar y haber pensado en todas las funcionalidades que se me han podido ocurrir para la aplicación, en un principio el mínimo que me impongo es que un usuario pueda crear un plato y visualizar los platos que el va creando, y como extra para un mayor alcance sería implementar una audio guía, teniendo que usar otras librerias para leer archivos en formato imagen y finalmente convertirlo en pdf

para que pueda ser extraído la información en forma de objeto.

Esquema base de datos no relacional

Usuarios

Atributos: ID de usuario (clave primaria), nombre de usuario, dirección de correo electrónico, contraseña, otros atributos específicos de los clientes premium (por ejemplo, nivel de membresía, beneficios adicionales, etc.)

Admin

Atributos: ID de admin (clave primaria), nombre de admin, dirección de correo electrónico, contraseña, permisos (un array de permisos)

Tutoriales

Atributos: ID de tutorial (clave primaria), título del tutorial, descripción del tutorial, pasos o instrucciones para hacer el ramen(una serie de imagenes que funcionarán como el tutorial) , ID de usuario que creó el tutorial (una referencia al ID de usuario), nivel de dificultad, tiempo de preparación, puntuación media de los otros usuario (calculada con la puntuación de los comentarios)

Platos

Atributos: ID de plato (clave primaria), nombre del plato, partes del plato (un array de id para relacionarlos con los

tutoriales), ID de usuario que creó el plato (una referencia al ID de usuario), nivel de picante

TABLA DE INFORMACIÓN DE LA API REST

Servicio	Descripción	Método HTTP	Ruta
Registro de un usuario	Registra un nuevo usuario en el sistema	POST	/register
Crear un tutorial	Crea un nuevo tutorial	POST	/login
Obtener todos los tutoriales	Obtiene todos los tutoriales disponibles	GET	/tutorial
Obtener un tutorial por su ID	Obtiene un tutorial específico según su ID	GET	/admin/tutorial:{id}
Actualizar un tutorial	Actualiza la información de un tutorial existente	PUT	/usuario:{id}
Eliminar un tutorial	Elimina un tutorial según su ID	DELETE	/platos:{id}/comentarios
Obtener un usuario por su ID	Obtiene un usuario según su ID	GET	/platos:{id}/comentarios
Eliminar un usuario por su ID	Elimina un usuario según su ID	DELETE	/usuario:{id}/platos
Actualizar un usuario por su	Actualiza la información de un	PUT	/platos

ID	usuario existente		
Obtener usuario por su correo electrónico	Obtiene un usuario según su correo electrónico	GET	/platos:{id}
Añadir un plato al array de platos del usuario	Añade un plato al array de platos del usuario	POST	/usuario/platos:{id}
Obtener platos de un usuario	Obtiene los platos asociados a un usuario específico	GET	/admin/usuario:{id}

Manual de despliegue

Despliegue local

Para desplegarlo de forma local basta con bajarse el proyecto de mi GitHub: https://github.com/Mcanteros2021/Ramen_Stadium_Cantero_Shimizu_Mario

Tras descargarlo hacemos NPI tanto en la carpeta backend y ramen_stadium.

Primero iniciamos el back con node app.js dentro de la carpeta backend

Y luego escribimos npm start para empezar a ver la pagina en react dentro de la carpeta ramen_stadium y podremos visualizarla en localhost:1234

Despliegue con Docker

1. Para desplegarlo con docker primero deberemos tener docker instalado.

2. Tras esto solo tienes que bajarte el repositorio de github: https://github.com/Mcanteros2021/Ramen_Stadium_Cantero_Shimizu_Mario que contiene un archivo dockerfile para montar la imagen

3. Tras esto montamos la imagen navegamos hasta el repositorio con la terminal por ejemplo y ejecutamos,
   `docker build -t myapp .`

4. Donde pone myapp puedes simplemente poner el nombre de tu aplicación. Tras eso esperamos a que se monte la imagen.

5. Una vez que la imagen esta montada verificamos que esta poniendo en la terminal **docker images**

6. Por último hacemos docker run -p 3001:3001 myapp
   Y podremos ver nuestra aplicación corriendo en ese puerto.

Conclusiones Postmortem

Ha sido un proyecto muy duro de llevar por una sola persona y pienso que cada pequeño detalle o error que me ha dado durante mi proyecto ha sido al final una gran carga en conjunto.

He aprendido mucho pero a la vez he sufrido debido a que no solamente he tenido que lidiar con el proyecto sino que tambien he tenido que ir a trabajar. A pesar de ello estoy orgulloso de lo que he hecho a pesar de que me ha faltado implementar muchas cosas más.