CES
Juan Pablo II
MADRID

MÓDULO PROYECTO

Rick y Morty

Darío Sánchez Iturralde
Borja Martín Herrera

ÍNDICE

1. Contenido

1.1. Resumen

Resumen en Español:

La página web desarrollada sobre Rick y Morty se centra en proporcionar una experiencia completa para los fanáticos de la serie. Utilizando React, se logró una interfaz de usuario atractiva, permitiendo a los usuarios explorar información detallada sobre los personajes a través de la integración de una API de Rick y Morty. El registro en Firebase ofrece a los usuarios la posibilidad de recibir noticias y ofertas personalizadas. La inclusión de una tienda afiliada de Amazon no solo enriquece la experiencia, sino que también presenta una oportunidad de magnetización. Un test estilo Kahoot agrega un toque educativo y lúdico al sitio, permitiendo a los usuarios evaluar sus conocimientos sobre la serie. Páginas adicionales, como la de error y la de descarga de contenido multimedia, mejoran la navegación y la diversidad de la plataforma. La aplicación de inteligencia artificial de Microsoft para la generación de logos aporta innovación al diseño. El uso de Bootstrap garantiza una interfaz responsive y visualmente agradable. En conclusión, el proyecto logra integrar de manera efectiva aspectos informativos, interactivos y comerciales, proporcionando una experiencia integral para los entusiastas de Rick y Morty.

Summary in English:

The developed Rick and Morty website focuses on providing a comprehensive experience for fans of the series. Using React, an attractive user interface was achieved, allowing users to explore detailed information about the characters through the integration of the Rick and Morty API. Firebase registration offers users the opportunity to receive personalized news and offers. The inclusion of an Amazon affiliate store not only enriches the experience but also presents a monetization opportunity. A Kahoot-style quiz adds an educational and playful touch to the site, allowing users to assess their knowledge of the series. Additional pages, such as the error page and the multimedia content download page, enhance navigation and platform diversity. The application of Microsoft's

artificial intelligence for logo generation brings innovation to the design. The use of Bootstrap ensures a responsive and visually appealing interface. In conclusion, the project effectively integrates informative, interactive, and commercial aspects, providing a comprehensive experience for Rick and Morty enthusiasts.

1.2. Justificación

El desarrollo de la página web dedicada a Rick y Morty surge como respuesta a diversas necesidades identificadas en el mercado y entre los seguidores apasionados de la serie. La motivación principal radica en abordar estas necesidades específicas de manera integral y proporcionar una experiencia completa y enriquecedora para los usuarios.

La demanda de información detallada sobre los personajes de Rick y Morty ha sido una constante. Los fanáticos buscan una plataforma que les permita explorar de manera exhaustiva la rica narrativa de la serie, profundizando en los detalles de sus personajes favoritos. El proyecto satisface esta necesidad al utilizar React y una API de Rick y Morty para crear una interfaz de usuario atractiva y funcional que presenta información detallada de manera clara y accesible.

La interactividad siempre ha sido un aspecto esencial en la comunidad de seguidores. Para abordar esta demanda, se desarrolló un test interactivo al estilo Kahoot. Este elemento educativo y lúdico permite a los usuarios evaluar sus conocimientos sobre la serie de manera divertida y atractiva.

La integración de una tienda afiliada de Amazon agrega una dimensión comercial al proyecto. No solo proporciona a los usuarios la oportunidad de adquirir productos relacionados con Rick y Morty de manera conveniente, sino que también presenta una oportunidad de monetización a través de comisiones por ventas.

En un esfuerzo por diversificar el contenido y ofrecer más opciones a los usuarios, se crearon páginas específicas para la descarga de contenido multimedia. Esto responde a la necesidad de los seguidores de acceder a

material adicional relacionado con la serie.

La implementación de inteligencia artificial de Microsoft para generar logos representa un enfoque innovador en el diseño. Esta característica no solo añade un toque único a la identidad visual del sitio, sino que también demuestra un compromiso con la innovación tecnológica.

La opción de registro en Firebase se integra para facilitar a los usuarios el acceso a ofertas y noticias personalizadas. Esta función contribuye a la creación de una experiencia más envolvente y personalizada para los usuarios.

En el proceso de desarrollo, se llevó a cabo un estudio de mercado exhaustivo para entender la competencia y las ofertas existentes en el ámbito de plataformas relacionadas con Rick y Morty. La mayoría de las aplicaciones existentes carecen de la combinación integral de características ofrecidas por este proyecto, destacando así la oportunidad de llenar un vacío en el mercado.

Desde una perspectiva económica, la integración de estrategias de magnetización, como la tienda afiliada de Amazon, respalda la viabilidad financiera del proyecto. La diversificación de contenido y la implementación técnica sólida también contribuyen a la sostenibilidad y éxito a largo plazo del proyecto.

En conclusión, la justificación del proyecto se basa en la identificación y respuesta a necesidades específicas en el mercado, la creación de una experiencia completa para los seguidores de Rick y Morty, y la viabilidad económica a través de estrategias de magnetización y análisis de competencia.

1.3.Objetivos

- **Desarrollo de Interfaz Atractiva:**
  Crear una interfaz de usuario visualmente atractiva y funcional utilizando React y Bootstrap.

- **Visualización de Personajes:**
  Implementar la visualización de personajes utilizando una API de Rick y Morty, mostrando detalles como imágenes y nombres.

- **Registro en Firebase:**
  Integrar un sistema de registro en Firebase para que los usuarios puedan introducir su correo electrónico y recibir ofertas o noticias.

- **Integración de Tienda Afiliada de Amazon:**
  Incorporar una tienda afiliada de Amazon para ofrecer productos relacionados con Rick y Morty.

- **Desarrollo de Test Estilo Kahoot:**
  Crear un test interactivo al estilo Kahoot para evaluar el conocimiento de los usuarios sobre la serie.

- **Gestión de Páginas Especiales:**
  Diseñar y gestionar páginas especiales como la de inicio, página de error y página de descarga de contenido multimedia.

- **Optimización de Consumo de API:**
  Utilizar el hook `useEffect` de React para consumir eficientemente la API de Rick y Morty.

- **Integración de Inteligencia Artificial de Microsoft:**
  Implementar la generación de logos mediante inteligencia artificial de Microsoft para ciertos elementos visuales.

- **Asegurar Privacidad y Seguridad:**
  Garantizar la privacidad y seguridad de los datos del usuario, especialmente en el proceso de registro en Firebase.

- **Documentación del Código:**
  Documentar el código de manera clara, explicando decisiones de diseño y facilitando el mantenimiento y colaboración futura.

2. Desarrollo.

2.1. Implementación o Estudio.

Desarrollo de Interfaz Atractiva:
Se aplicó el diseño centrado en el usuario, utilizando React para la construcción modular de componentes y Bootstrap para la maquetación responsiva. Iteraciones basadas en prototipos permitieron ajustes continuos.

Visualización de Personajes:
La metodología ágil facilitó la implementación rápida de la visualización de personajes, utilizando React para componentes dinámicos y el hook `useEffect` para gestionar eficientemente las llamadas a la API de Rick y Morty.

Registro en Firebase:
Se implementó la autenticación de Firebase para el registro de usuarios, aplicando principios de seguridad web. Firebase Firestore gestionó eficazmente la información de usuario, garantizando la privacidad de los datos.

Integración de Tienda Afiliada de Amazon:
La técnica de afiliación generó oportunidades de monetización a través de comisiones por ventas.

Desarrollo de Test Estilo Kahoot:
Se aplicó la metodología ágil para el desarrollo iterativo del test, utilizando React para la interfaz y un backend para gestionar preguntas y respuestas. Pruebas de usuario continuas ajustaron el contenido.

Gestión de Páginas Especiales:
React Router facilitó la creación de páginas específicas como inicio, error y descarga. La agilidad en el desarrollo permitió adaptar las páginas según las necesidades y feedback de los usuarios.

Optimización de Consumo de API:
Se aplicó el hook `useEffect` de React con cuidado para optimizar las llamadas a la API. Técnicas de caché minimizaron la carga de datos, asegurando una experiencia fluida y eficiente.

Integración de Inteligencia Artificial de Microsoft:
Se experimentó con la generación de logos mediante inteligencia artificial de Microsoft, aportando innovación visual al proyecto creativo y personalizando elementos visuales.

Asegurar Privacidad y Seguridad:
Prácticas de seguridad web y Firebase aseguraron la privacidad de los usuarios. Se realizaron pruebas de penetración para identificar y abordar posibles vulnerabilidades.

Optimización para Dispositivos Móviles:
Bootstrap posibilitó la creación de una interfaz responsive, garantizando una experiencia consistente en dispositivos móviles. Pruebas exhaustivas validaron la adaptabilidad en diversos dispositivos.

Documentación del Código en GitHub:

Se documentó el código utilizando JSDoc y se centralizó en un repositorio en GitHub. READMEs detallados proporcionaron orientaciones claras para la instalación y contribuciones futuras, fomentando la colaboración y transparencia.

2.2.Resultados

Interfaz Atractiva y Funcional:

Se logró una interfaz visualmente atractiva y funcional, con una combinación efectiva de React y Bootstrap, proporcionando una experiencia de usuario agradable.

Visualización de Personajes:

La implementación de la visualización de personajes a través de la API de Rick y Morty fue exitosa, permitiendo a los usuarios explorar detalles de manera eficiente.

Registro en Firebase:

El sistema de registro en Firebase se implementó con éxito, ofreciendo a los usuarios la posibilidad de recibir ofertas o noticias de manera segura.

Integración de Tienda Afiliada de Amazon:

La tienda afiliada de Amazon se integró satisfactoriamente, proporcionando a los usuarios una opción conveniente para adquirir productos relacionados.

Test Estilo Kahoot:

El test interactivo al estilo Kahoot se desarrolló con éxito, brindando a los usuarios una manera lúdica de evaluar y ampliar sus conocimientos sobre la serie.

Gestión de Páginas Especiales: Las páginas especiales, como la de inicio y la de descarga, se gestionaron de manera efectiva, mejorando la navegación y la experiencia del usuario.

Optimización de Consumo de API:

Se logró una optimización eficiente del consumo de la API de Rick y Morty, garantizando una carga de datos suave y rápida.

Integración de Inteligencia Artificial de Microsoft:

La integración de inteligencia artificial de Microsoft para generar logos fue exitosa, aportando un elemento innovador al diseño visual.

Asegurar Privacidad y Seguridad:

Las medidas de seguridad implementadas aseguraron la privacidad de los datos del usuario, brindando una experiencia segura en el registro y almacenamiento de información.

Optimización para Dispositivos Móviles:

Bootstrap facilitó la optimización para dispositivos móviles, garantizando una experiencia consistente y atractiva en diversas plataformas.

Documentación del Código en GitHub:

La documentación detallada en GitHub proporciona recursos claros para futuros desarrolladores, facilitando la comprensión y contribución al proyecto.

En conjunto, estos resultados reflejan el éxito en la implementación de los objetivos planteados, proporcionando una plataforma completa y funcional centrada en la experiencia del usuario.

3.Conclusiones

Análisis de Resultados y Futuras Implementaciones:

Los resultados obtenidos reflejan un éxito en la implementación de la página web de Rick y Morty, pero también señalan áreas de mejora y oportunidades para futuras implementaciones:

Interfaz y Experiencia del Usuario:

Mejoras*:* Se podría realizar un análisis de feedback de usuarios para identificar áreas específicas de mejora en la interfaz. La implementación de animaciones o micro interacciones podría agregar un toque adicional a la experiencia del usuario.

Funcionalidad de la Tienda Afiliada:

Mejoras*:* Explorar la posibilidad de ampliar la selección de productos y categorías en la tienda afiliada. Implementar funciones de búsqueda avanzada y recomendaciones personalizadas podría mejorar la experiencia de compra.

Registro de usuarios:

Mejoras: realizar un login en el que el usuario registrado tenga acceso a contenido que una persona no registrada no tenga.

Test Interactivo y Contenido Adicional:

Mejoras*:* Continuar actualizando y ampliando el test interactivo con nuevas preguntas y desafíos. Explorar la posibilidad de agregar contenido multimedia exclusivo para usuarios registrados podría aumentar la participación.

Optimización y Rendimiento:

Mejoras*:* Realizar auditorías de rendimiento periódicas para identificar y abordar posibles cuellos de botella. Considerar estrategias de carga progresiva para optimizar el tiempo de carga de la página.

Aumento de ingresos:

Mejoras: valorar la posibilidad de añadir Google Ads para monetizar las visitas mediante anu8ncios valorando también que no perjudique la experiencia de usuario

En conclusión, la página actual proporciona una base sólida, pero la evolución constante es esencial. Explorar las tendencias tecnológicas, recopilar feedback de usuarios y priorizar mejoras incrementales garantizarán una experiencia continua y relevante para los seguidores de Rick y Morty.

4. Revisión Bibliográfica

- YouTube creador: MonckeyWit: “Cómo consumir una API con React | Rick and Morty App”
  Link:https://www.youtube.com/watch?v=BTJtTkoyprc&ab_channel=MonkeyWit

- YouTube creador: Uzoanya Dominic: “React Tutorial - Build a Quiz Application ”
  Link:https://www.youtube.com/watch?v=DJxM76m4eP0&list=PLEVTJcDnFDm9lpEEHTftRa9JSRV4jY_p9&ab_channel=UzoanyaDominic

- Página web: ChatGPT
  Link: https://chat.openai.com/

- Página web: Rick y Morty: “Documentación API Rick y Morty.”
  Link: https://rickandmortyapi.com/documentation/

- Página web: Bootstrap
  Link: https://getbootstrap.com/

5. Anexos

5.1. Manual técnico

He usado varias tecnologías para crear la pagina web entre ellas Firebase, Bootstrap, Router, todo esto usando react de base, para instalar estas herramientas deberemos usar **npm install **Nombre de la herramienta**** ej: **npm install Firebase.**

Lo primero que se ha hecho ha sido crear la pagina de los personajes en la que consultamos esta API **https://rickandmortyapi.com/api/**, después crearemos un componente **useEffect** para manejar la lógica de la API, utilizamos el método **fetch** para realizar la solicitud HTTP a la API .

```
const Personajes = () => {
  const [paginaActual, setPaginaActual] = useState(1);
  const [datos, setDatos] = useState(null);

  const apiUrl = `https://rickandmortyapi.com/api/character/?page=${paginaActual}`;

  useEffect(() => {
    fetch(apiUrl)
      .then(response => response.json())
      .then(data => setDatos(data))
      .catch(error => console.error('Error al obtener datos desde la API:', error));
  }, [apiUrl]);

  const avanzarPagina = () => {
    setPaginaActual(paginaActual + 1);
  };

  const retrocederPagina = () => {
    if (paginaActual > 1) {
      setPaginaActual(paginaActual - 1);
    }
  };
```

Por ultimo toca renderizar los datos obtenidos en tu componente React.

```
{datos && (
  <div className="row text-bg-dark">
    {/* Renderiza los datos obtenidos de la API */}
    {datos.results.map(character => (
      <div className="col">
      <div>
        <div
          key={character.id}
          className="card border-info text-bg-dark"
          style={{
            width: "18rem",
            marginBottom: "20px",
            marginTop: "20px",
          }}
        >
          <img
            src={character.image}
            className="card-img-top text-bg-dark"
            alt=""
            style={{
              width: "200px",
              alignSelf: "center",
              margin: "20px",
              height: "200px",
            }}
          />
          <div className="card text-center text-bg-dark">
            <div className="card-body">
              <h5 className="card-title">Nombre: {character.name}</h5>
              <p className="card-text">Especie: {character.species}</p>
              <p className="card-text">Genero: {character.gender}</p>
            </div>
          </div>
        </div>
      </div>
```

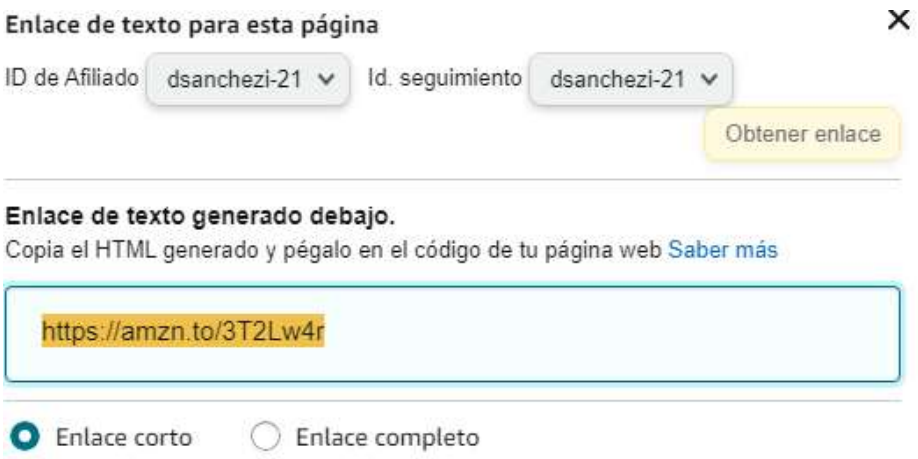
Además hemos añadido la funcionalidad de movernos entre las paginas de la api mediante dos botones uno de avance y otro de retroceso, para ello creamos dos constantes una que haga +1 al numero de la pagina y otra que haga -1, después de ello el useEffect se ejecuta cada vez que ser cambia de pagina y la URL de la API se ajusta para incluir el parámetro de página actual.

```
const avanzarPagina = () => {
  setPaginaActual(paginaActual + 1);
};

const retrocederPagina = () => {
  if (paginaActual > 1) {
    setPaginaActual(paginaActual - 1);
  }
};
```

Después creamos el resto de las paginas para ponernos con la gestión de rutas mediante el **Router,** básicamente a cada pagina le asignamos una ruta, y para la pagina de Error unicamente deberemos indicarle que todo lo que no este contemplado en las rutas nos mande automáticamente a **/Error** y se mostrara una pantalla de error.

```
function App() {
  return (
    <Router>
      <Nav />
      <Routes>
        <Route path="/" element={<Inicio />}></Route>
        <Route path="/Personajes" element={<Personajes />}></Route>
        <Route path="/Tienda" element={<Tienda />}></Route>
        <Route path="/Multimedia" element={<Multimedia />}></Route>
        <Route path="/Noticias" element={<Noticias />}></Route>
        <Route path="/Test" element={<Test />}></Route>
        <Route path="/Login" element={<Login />}></Route>
        <Route path="/Error" element={<Error />}></Route>
        <Route path="/*" element={<Navigate to='/Error' />}></Route>
      </Routes>
    </Router>
  );
}

export default App;
```

A continuación crearemos la tienda en la que deberemos primero crearnos una cuenta en Amazon afiliados, después de ello buscar un producto que se relacione con nuestro contenido y obtener una o varas fotos de el en nuestro caso con una basta, después deberemos copiar el enlace que nos proporciona Amazon afilados ya que es diferente a uno normal ya que lleva un identificativo nuestro para llevarnos una comisión de una posible venta.


Enlace de texto para esta página
ID de Afiliado
dsanchezi-21
Id. seguimiento
dsanchezi-21
Obtener enlace
Enlace de texto generado debajo.
Copia el HTML generado y pégalo en el código de tu página web Saber más
https://amzn.to/3T2Lw4r
Enlace corto
Enlace completo

Para crear el Navbar o menú deberemos crear un archivo Nav.js dentro de la carpeta Components para que mediante Link todas la paginas lo tengan.

```
import React from "react";
import { Link } from "react-router-dom";

const Nav = () => {
  return (
    <nav class="navbar navbar-dark bg-dark fixed-top">
  <div class="container-fluid">
    <a class="navbar-brand" href="#">Rick y Morty</a>
    <button class="navbar-toggler" type="button" data-bs-toggle="offcanvas" data-bs-target="#offcanvasDarkNa
      <span class="navbar-toggler-icon"></span>
    </button>
    <div class="offcanvas offcanvas-end text-bg-dark" tabindex="-1" id="offcanvasDarkNavbar" aria-labelledby
      <div class="offcanvas-header">
        <h5 class="offcanvas-title" id="offcanvasDarkNavbarLabel">Rick y Morty</h5>
        <button type="button" class="btn-close btn-close-white" data-bs-dismiss="offcanvas" aria-label="Clos
      </div>
      <div class="offcanvas-body">
        <ul class="navbar-nav justify-content-end flex-grow-1 pe-3">
          <li class="nav-item">
            <Link className="nav-link" to={"/"}>Inicio</Link>
          </li>
          <li class="nav-item">
            <Link className="nav-link" to={"/Personajes"}>Personajes</Link>
          </li>
          <li class="nav-item">
            <Link className="nav-link" to={"/Tienda"}>Tienda</Link>
          </li>
          <li class="nav-item">
            <Link className="nav-link" to={"/Test"}>¿Cuanto sabes de Rick y Morty?</Link>
          </li>
          <li class="nav-item">
            <Link className="nav-link" to={"/Login"}>Login</Link>
          </li>
          <li class="nav-item">
            <Link className="nav-link" to={"/Multimedia"}>Multimedia</Link>
          </li>
        </ul>
      </div>
    </div>
  </div>
</nav>
  );
};

export default Nav;
```

Después de esto creamos el registro en el que el usuario pone su correo y una contraseña, para poder recibir ofertas y noticias de la pagina web para el desarrollo de este login tuvimos que instalar firebase como indicamos anteriormente, seguidamente crearemos un Crea un archivo firebase.js y configura Firebase con tus credenciales y con la función de guardar los datos.

```
import { initializeApp } from "firebase/app";
import { getFirestore } from "firebase/firestore";
import { addDoc, collection } from "@firebase/firestore";

const getFirebaseFirestore = () => {
  const firebaseConfig = {
    apiKey: "AIzaSyBakn1OgCcC6-q0ECjJqnfz2MonBkWPOU8",
    authDomain: "rick-morty-tfg.firebaseapp.com",
    projectId: "rick-morty-tfg",
    storageBucket: "rick-morty-tfg.appspot.com",
    messagingSenderId: "975265130939",
    appId: "1:975265130939:web:3024b787fd3d254944fc9e"
  };

  const app = initializeApp(firebaseConfig);
  return getFirestore(app);
};

const save = (correo, password) => {
  const firestore = getFirebaseFirestore();
  const ref = collection(firestore, "login");

  try {
    addDoc(ref, { correo, password});
    console.log("usuario registrado");
  } catch (err) {
    console.log(err);
  }
};

export default save;
```

Después unicamente deberemos crear las constantes correo y contraseña para pasárselas a la lógica que hemos visto en la pagina anterior, para esto deberemos crear una interfaz en la que el usuario pueda introducir sus credenciales.

```jsx
const Login = () => {
  const [correo, setCorreo] = useState("");
  const [password, setPassword] = useState("");

  return (
    <div style={{ marginTop:"-20px", marginBottom:"-20px" ,backgroundImage: `url(${Fondo})` }}>
    <div className="container text-center" style={{ marginTop: "75px" }}>
    <img src={Logo} alt="" style={{ width: "900px", height: "100%", marginTop: "100px", }} />
    <br/>
    <p style={{ textAlign: "center", marginTop: "20px", color:"white" }}>
        ¡Únete a nuestra comunidad de Rick y Morty! Regístrate ahora para recibir ofertas exclusivas y estar al tanto
    </p>
      <form>
        <div className="form-group">
          <input
            onKeyUp={(event) => setCorreo(event.target.value)}
            type="email"
            name="correo"
            className="form-control"
            placeholder="Email"
          />
        </div>
        <br />
        <div className="form-group">
          <input
            onKeyUp={(event) => setPassword(event.target.value)}
            type="password"
            name="password"
            className="form-control"
            placeholder="Contraseña"
```

Lo siguiente fue crear el juego de preguntas estilo Kahoot primero Creamos un array que contiene objetos de preguntas con sus respectivas opciones y respuestas correctas.

```jsx
const questionsData = [
  {
    id: 1,
    text: '¿Cuál es el nombre del abuelo en Rick and Morty?',
    options: ['Rick', 'Morty', 'Jerry', 'Birdperson'],
    correctAnswer: 'Rick',
  },
  // Pregunta 2
  {
    id: 2,
    text: '¿Qué especie es Birdperson?',
    options: ['Humano', 'Gaviota', 'Cyborg', 'Alien'],
    correctAnswer: 'Alien',
  },
  // Pregunta 3
  {
    id: 3,
    text: '¿Cómo se llama la dimensión hogar de Rick?',
    options: ['Dimensión C-137', 'Dimensión XYZ', 'Dimensión 123', 'Dimensión WXYZ'],
    correctAnswer: 'Dimensión C-137',
  },
  // Pregunta 4
  {
    id: 4,
    text: '¿Quién es el mejor amigo de Morty?',
    options: ['Birdperson', 'Jerry', 'Rick', 'Summer'],
    correctAnswer: 'Birdperson',
  },
  // Pregunta 5
```

utilizamos el hook useState para definir tres estados:currentQuestionIndex para rastrear la pregunta actual, selectedOption para almacenar la opción seleccionada por el usuario, y score para registrar la puntuación del usuario.
La función `handleAnswer` evalúa si la opción seleccionada por el usuario coincide con la respuesta correcta. En caso afirmativo, se incrementa la puntuación (`score`).Se avanza a la siguiente pregunta si no es la última.
La función principal App renderiza la interfaz del juego. Si hay más preguntas disponibles, se muestra la pregunta actual y las opciones. Al seleccionar una opción y hacer clic en "Enviar respuesta", se ejecuta la función `handleAnswer`.

```jsx
const App = () => {
  const [currentQuestionIndex, setCurrentQuestionIndex] = useState(0);
  const [selectedOption, setSelectedOption] = useState(null);
  const [score, setScore] = useState(0);

  const handleAnswer = () => {
    const currentAnswer = questionsData[currentQuestionIndex].correctAnswer;
    if (selectedOption === currentAnswer) {
      setScore(score + 1);
    }

    // Avanzar a la siguiente pregunta
    if (currentQuestionIndex < questionsData.length - 1) {
      setCurrentQuestionIndex(currentQuestionIndex + 1);
      setSelectedOption(null); // Reiniciar la opción seleccionada para la nueva pregunta
    }
  };
```

```jsx
return (
  <div className="game-container">
    {currentQuestionIndex < questionsData.length ? (
      // Mostrar preguntas mientras haya más preguntas disponibles
      <div>
        <h1 className="game-title">Rick and Morty Kahoot</h1>
        <div>
          <h2 className="question-title">{questionsData[currentQuestionIndex].text}</h2>
          <ul className="options-list">
            {questionsData[currentQuestionIndex].options.map((option, index) => (
              <li
                key={index}
                className={`option-item ${selectedOption === option ? 'selected' : ''}`}
                onClick={() => setSelectedOption(option)}
              >
                {option}
              </li>
            ))}
          </ul>
          <button className="answer-button" onClick={handleAnswer}>
            Enviar respuesta
          </button>
          {selectedOption && (
            <p className={`feedback ${selectedOption === questionsData[currentQuestionIndex].correctAnswer ? 'correct' : 'incorrect'}`}>
              {selectedOption === questionsData[currentQuestionIndex].correctAnswer ? '¡Correcto!' : 'Incorrecto. Elige otra respuesta.'}
            </p>
          )}
        </div>
      </div>
    ) : (
      // Mostrar resultados al finalizar todas las preguntas
      <div>
        <h1 className="game-title">¡Juego Terminado!</h1>
        <p>Tu conteo de aciertos es: {score} de {questionsData.length}</p>
      </div>
    )}
```

Y por ultimo creamos la pagina Multimedia que consiste en una pagina con el mismo estilo de la web en el que hay unas imágenes que se puede descargar el usuario.

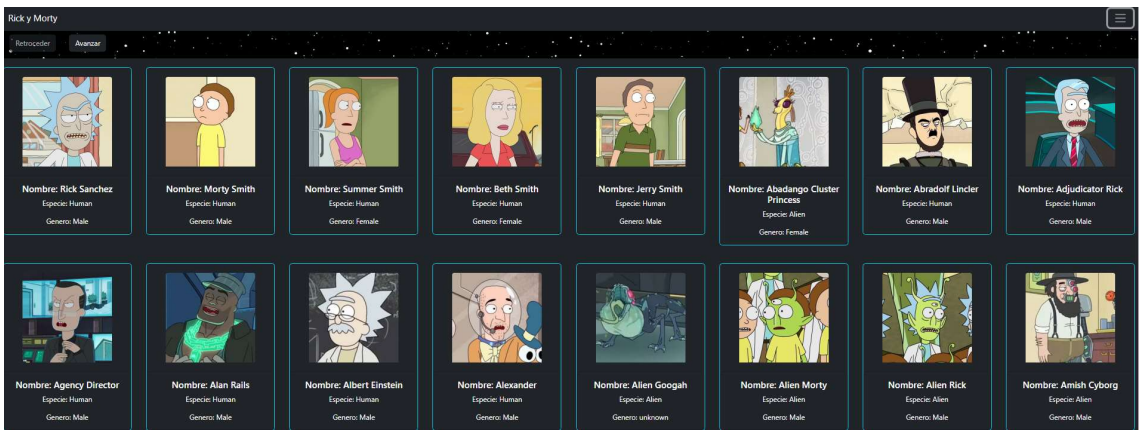
5.2. Manual de usuario

Para poder ver la pagina deberemos ir a https://github.com/DarioSanchez99/TFG allí descargaremos el proyecto en formato zip, después lo descomprimiremos en el local, al abrirlo con el Visual Studio Code deberemos abrir una terminal y e instalar las dependencias utilizando **npm install.**

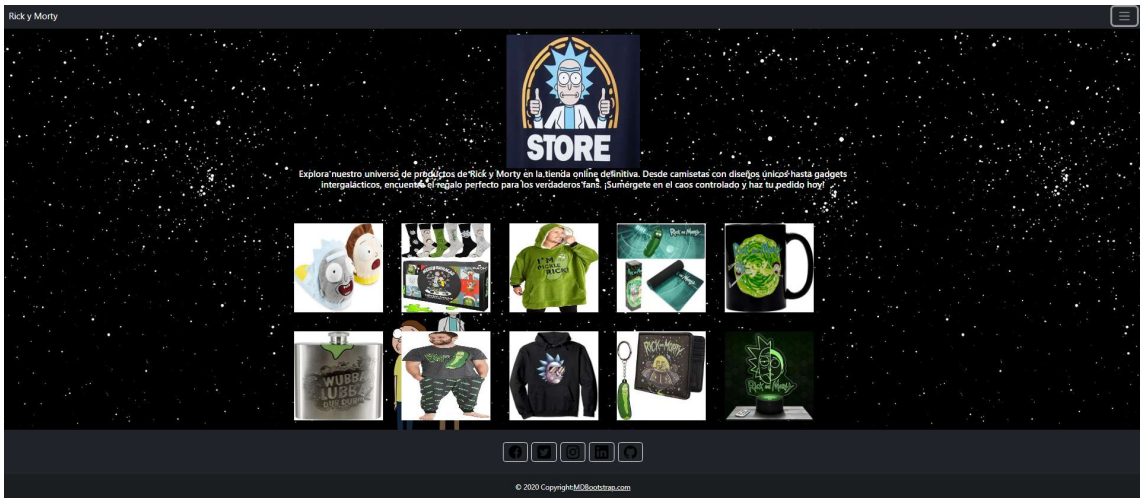
Después usaremos **npm start** para iniciar la aplicación web.

Se nos abrirá el buscador y estaremos en la pagina de inicio, en la que podremos desplegar el menú para ver las diferentes páginas.

La primera opción que nos aparece es el inicio, en la que estamos ahora, la siguiente es la de personajes, en la que podremos navegar mediante los dos botones entre diferentes paginas visualizando los diferentes personajes.



La siente pagina es la tienda de afiliados en la que podremos ver diferentes productos relacionados con la serie.



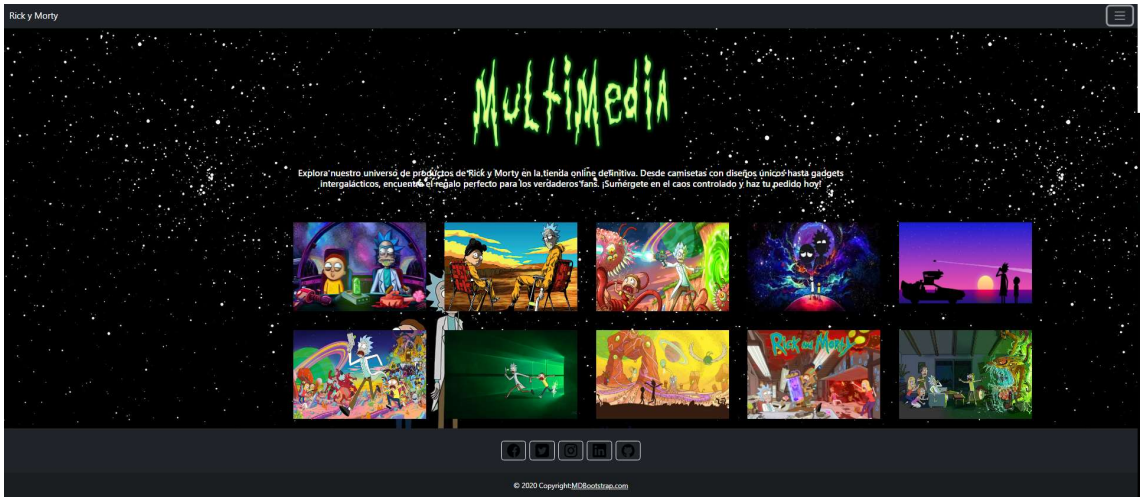
A continuación nos dirigiremos al test estilo kahoot sobre la serie, en la que contestaremos preguntas e iremos viendo el resultado de ellas.

Para poder asociar vuestro firebase deberéis cambiar el archivo acciones que esta en la carpeta service lo siguiente marcado por los datos correspondientes de tu cuenta de firebase.

```
import { initializeApp } from "firebase/app";
import { getFirestore } from "firebase/firestore";
import { addDoc, collection } from "@firebase/firestore";

const getFirebaseFirestore = () => {
  const firebaseConfig = {
    apiKey: "AIzaSyBakn1OgCcC6-q0ECjJqnfz2MonBkWPOU8",
    authDomain: "rick-morty-tfg.firebaseapp.com",
    projectId: "rick-morty-tfg",
    storageBucket: "rick-morty-tfg.appspot.com",
    messagingSenderId: "975265130939",
    appId: "1:975265130939:web:3024b787fd3d254944fc9e"
  };

  const app = initializeApp(firebaseConfig);
  return getFirestore(app);
};

const save = (correo, password) => {
  const firestore = getFirebaseFirestore();
  const ref = collection(firestore, "login");

  try {
    addDoc(ref, { correo, password});
    console.log("usuario registrado");
  } catch (err) {
    console.log(err);
  }
};

export default save;
```

El login sería este.

Y por ultimo esta la pagina de multimedia en la que serán unas imágenes en las que el usuario podrá descargarlas para su uso.


Rick y Morty
MuLtiMedia
Explora nuestro universo de productos de Rick y Morty en la tienda online definitiva. Desde camisetas con diseños únicos hasta gadgets intergalácticos, encuentra el regalo perfecto para los verdaderos fans. ¡Sumérgete en el caos controlado y haz tu pedido hoy!
© 2020 Copyright:MDBootstrap.com