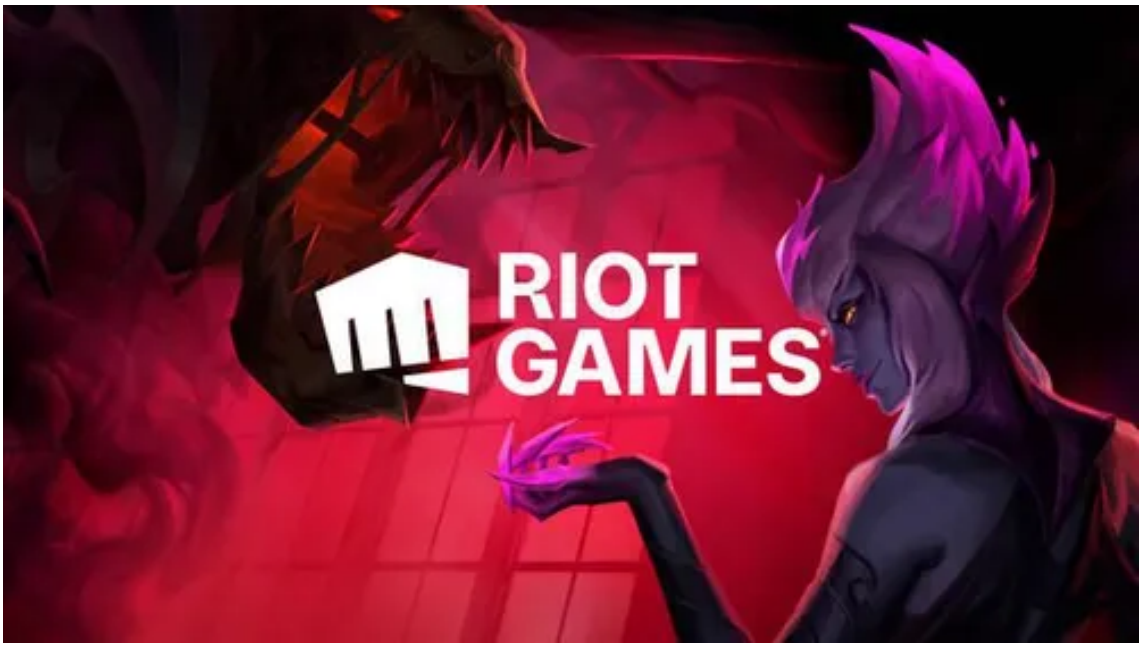
Riot Games



Introducción

Mi proyecto se basa en la recreación de las interfaces cliente de la compañía Riot Games.

¿Qué es Riot Games?

Riot Games, Inc. es una empresa estadounidense dedicada al desarrollo y publicación de videojuegos, así como a la organización de torneos de deportes electrónicos.

Fundada en septiembre de 2006 por Brandon Beck y Marc Merrill, tiene su sede en Los Ángeles, California. La compañía es reconocida principalmente por la creación de "League of Legends", un videojuego multijugador de arena de batalla en línea (MOBA) que ha alcanzado gran popularidad a nivel mundial.

Además de "League of Legends", Riot Games ha desarrollado otros títulos como "Valorant", un shooter táctico en primera persona, y "Teamfight Tactics", un juego de estrategia automática.

La empresa también organiza y gestiona competiciones de deportes electrónicos relacionadas con sus juegos, consolidándose como un actor destacado en la industria del gaming y los esports.

Arquitectura de los Directorios

La organización de los directorios dentro de un proyecto es muy importante para la organización y comprensión del mismo.

He optado por una arquitectura de manera lógica y modular, basada en la funcionalidad de cada archivo.

Primero he creado la carpeta raíz (interfaz-riot-games), la cual representa el proyecto completo y en la cual se sitúan las subcarpetas y archivos necesarios para la implementación y organización del proyecto.

Dentro de la carpeta raíz he creado tres directorios principales.

El directorio Public

Esta carpeta almacena todos los archivos accesibles de manera pública en la aplicación.

Contiene tres subcarpetas:

- css/
  - Contiene hojas de estilo para el diseño visual del sitio.
    - `bootstrap-theme.css` y `bootstrap.css`: Archivos del framework Bootstrap que permiten crear diseños responsivos y predefinidos.
    - `styles.css`: Archivo CSS personalizado para los estilos específicos del proyecto.
- img/
  - Carpeta dedicada a las imágenes utilizadas en la aplicación, como banners, logotipos y elementos gráficos.
    - Ejemplo:
      - `banner-lol.jpg` y `promocion1.jpg`: Posiblemente se usan en las vistas como elementos visuales.
      - Archivos como `logo-lol.png`, `logo-valorant.png`, y otros logos se usan probablemente en menús o en secciones específicas del sitio para diferenciar cada juego.
- js/
  - Contiene scripts JavaScript utilizados para añadir funcionalidad interactiva al sitio.
    - `bootstrap.js`: Script del framework Bootstrap.
    - `login.js` y `register.js`: Scripts dedicados para manejar funcionalidades de login y registro.

- `script.js`: Archivo general para otras funcionalidades de la página web.

El directorio SRC

Esta carpeta contiene el código fuente principal del proyecto. Aquí se definen las funcionalidades del lado del servidor y otros recursos esenciales.

Contiene dos archivos:

- login.php
  - Archivo PHP para manejar el backend del sistema de autenticación de usuarios.
- usuarios.json
  - Archivo JSON para almacenar información de usuarios (temporalmente). Puede ser usado para validar datos en el login o para realizar pruebas locales.

El directorio Templates

Contiene plantillas HTML que estructuran las vistas del sitio web. Se organiza en dos categorías principales:

a) partials/
   i) **Propósito:** Contiene partes reutilizables de las vistas (por ejemplo, la navegación o elementos comunes).
      1) `_navigation.html`: Archivo para el menú o barra de navegación reutilizable en varias páginas.

b) Vistas principales
   i) Archivos como `home.html`, `register.html`, `valorant.html`, etc., representan las diferentes páginas que el usuario puede visitar.
   ii) Cada archivo corresponde a una sección específica:
      1) `leagueoflegends.html`, `runeterra.html`, y `valorant.html`: Páginas dedicadas a cada juego.
      2) `misjuegos.html`: Posiblemente muestra los juegos asociados al usuario.
      3) `register.html`: Página para registrar nuevos usuarios.

Conclusión

Esta estructura sigue un enfoque modular y claro:

1. **Separación de responsabilidades:** Cada carpeta agrupa archivos según su función (estilos, imágenes, scripts, plantillas, backend, etc.).
2. **Mantenibilidad:** Facilita futuras modificaciones o expansiones del proyecto.
3. **Reutilización:** Los archivos en `partials/` permiten reutilizar componentes comunes, ahorrando tiempo y evitando redundancia.
4. **Accesibilidad:** La carpeta `public/` asegura que los recursos estáticos sean fácilmente accesibles desde el navegador.

Explicación de los archivos JS

En este proyecto, he implementado scripts para manejar la funcionalidad del formulario de inicio de sesión, el registro de usuarios, y funcionalidades generales de la interfaz. Aquí explico cada uno de los archivos, sus funcionalidades principales, su ubicación en el proyecto y el código implementado.

1. login.js

¿Qué hace?

- Este archivo gestiona la funcionalidad del formulario de inicio de sesión, incluyendo validación básica, visibilidad de contraseñas y redirección a la página principal tras un inicio exitoso.

Funciones principales:

- **Evento `DOMContentLoaded`:**
  - Se ejecuta cuando el DOM está completamente cargado.
  - Asigna funcionalidad al botón de inicio de sesión y a la visibilidad de la contraseña.
- **Manejo del botón "Login":**
  - Previene la recarga de la página al presionarlo.
  - Obtiene los valores de los campos `username` y `password`.
  - Usa la función `verificarCampos` para validar que los campos no estén vacíos. Si están vacíos, muestra una alerta y no continúa.
  - Envía una solicitud HTTP GET al archivo `home.html` utilizando `XMLHttpRequest`. Si la solicitud es exitosa, redirige al usuario; de lo contrario, muestra un mensaje de error.
- **Mostrar/Ocultar contraseña:**
  - Cambia dinámicamente el tipo del campo entre `text` y `password`.
  - Alterna los íconos de "ojo abierto" y "ojo cerrado".
- **Función `verificarCampos`:**
  - Valida que los campos `username` y `password` no estén vacíos. Si alguno lo está, muestra una alerta y detiene el envío del formulario.

¿Dónde se encuentra?

- En `/public/js/login.js`.
- Está referenciado en `login.html` dentro de la etiqueta `<script src="/public/js/login.js"></script>`.

¿Es propio o de un tercero?

- Es un script propio desarrollado específicamente para este proyecto.

Código:

```
document.addEventListener("DOMContentLoaded",function () {

      // Comprobar el From y redireccionar al home.html. Hay que hacer cuando le des
enter que funcione también (Hecho por mi)

     const loginButton = document.getElementById("login-button");

     const homeHtml = 'home.html';

     const xhr = new XMLHttpRequest();

     loginButton.addEventListener("click", function(event){

         event.preventDefault();



         //Coger los datos de los campos del formulario

         var userName = document.getElementById("username").value.trim();

         var password = document.getElementById("password").value.trim();



         // Llamar a la funcion de verificación

           if (!verificarCampos(userName, password)) {

             return; // Si la validación falla, detener el envío

         }

         // Falta verificar bien los datos.

         xhr.open('GET', homeHtml, true);

         xhr.onreadystatechange = function () {

             if (xhr.readyState === 4 && xhr.status === 200) {

                 // Si la autenticación es exitosa, redirigir a home.html

                 window.location.href = "home.html";

                 } else {

                     // Manejar el error de autenticación (opcional)

                     alert("Error de autenticación. Por favor verifica tus datos.");

                 }
```

```
        }

        xhr.send();

    });



     // Script para mostrar u ocultar la contraseña en el login.
    var passwordInput = document.getElementById("password");
    var togglePassword = document.getElementById("toggle-password");
    togglePassword.addEventListener("click", function() {
        // Alternar el tipo de entrada
        const type = passwordInput.type === "password" ? "text" : "password";
        passwordInput.type = type;

        // Cambiar el ícono entre el ojo abierto y cerrado
        this.classList.toggle("fa-eye");
        this.classList.toggle("fa-eye-slash");
    });
});

// Funcion para verificar si los campos están vacios
function verificarCampos(userName,password) {

    if (userName === "") {
        alert("El campo 'Nombre' es obligatorio.");
        return false; // Detener el envío si el campo está vacío
    }

    if (password === "") {
```

```
        alert("El campo 'Password' es obligatorio.");

        return false; // Detener el envío si el campo está vacío

    }

    return true; // Si los campos no están vacíos, enviar el formulario

}
```

---

2. register.js

¿Qué hace?

- Este script gestiona el formulario de registro. Valida los datos ingresados por el usuario, permite alternar la visibilidad de la contraseña y redirige al usuario a la página de inicio de sesión tras completar el registro.

Funciones principales:

- **Evento `DOMContentLoaded`:**
  - Ejecuta el código una vez que el DOM está completamente cargado.
  - Asigna funcionalidad al botón de registro y a la visibilidad de la contraseña.
- **Manejo del botón "Register":**
  - Previene la acción por defecto del formulario.
  - Obtiene los valores de los campos `username` y `password`.
  - Usa la función `verificarCampos` para validar los campos antes de continuar.
  - Envía una solicitud HTTP GET al archivo `login.html`. Si tiene éxito, redirige al usuario a la página de inicio de sesión.
- **Mostrar/Ocultar contraseña:**
  - Igual que en `login.js`, alterna entre los tipos `text` y `password` en el campo de contraseña, y actualiza los íconos.
- **Función `verificarCampos`:**
  - Similar a la de `login.js`, asegura que los campos no estén vacíos y muestra alertas en caso de que lo estén.

¿Dónde se encuentra?

- En `/public/js/register.js`.
- Está referenciado en `register.html` dentro de la etiqueta `<script src="/public/js/register.js"></script>`.

¿Es propio o de un tercero?

- Es un script propio desarrollado para este proyecto.

Código:

```
document.addEventListener("DOMContentLoaded", function () {

     // Comprobar el From y redireccionar al home.html. Hay que hacer cuando le des
enter que funcione también (Hecho por mi)

    const registerButton = document.getElementById("register-button");

    const loginHtml = 'login.html';

    const xhr = new XMLHttpRequest();

    registerButton.addEventListener("click", function (event) {

        event.preventDefault();



        //Coger los datos de los campos del formulario

        var userName = document.getElementById("username").value.trim();

        var password = document.getElementById("password").value.trim();



        // Llamar a la funcion de verificación

        if (!verificarCampos(userName, password)) {

            return; // Si la validación falla, detener el envío

        }

        // Falta verificar bien los datos.

        xhr.open('GET', loginHtml, true);

        xhr.onreadystatechange = function () {

            if (xhr.readyState === 4 && xhr.status === 200) {

                // Si la autenticación es exitosa, redirigir a home.html

                window.location.href = "login.html";

            } else {

                // Manejar el error de autenticación (opcional)
```

```
                alert("Error de autenticación. Por favor verifica tus datos.");

            }

        }

        xhr.send();

    });




    // Script para mostrar u ocultar la contraseña en el login.

    var passwordInput = document.getElementById("password");

    var togglePassword = document.getElementById("toggle-password");

    togglePassword.addEventListener("click", function () {

        // Alternar el tipo de entrada

        const type = passwordInput.type === "password" ? "text" : "password";

        passwordInput.type = type;



        // Cambiar el ícono entre el ojo abierto y cerrado

        this.classList.toggle("fa-eye");

        this.classList.toggle("fa-eye-slash");

    });

});



// Funcion para verificar si los campos están vacios

function verificarCampos(userName, password) {



    if (userName === "") {

        alert("El campo 'Nombre' es obligatorio.");

        return false; // Detener el envío si el campo está vacío

    }
```

```
    if (password === "") {
        alert("El campo 'Password' es obligatorio.");
        return false; // Detener el envío si el campo está vacío
    }
    return true; // Si los campos no están vacíos, enviar el formulario
}
```

---

3. script.js

¿Qué hace?

- Este archivo añade funcionalidades generales a la interfaz, como la carga dinámica de la barra de navegación y el resaltado del enlace activo.

Funciones principales:

- **Cargar un parcial de navegación:**
  - Utiliza jQuery para cargar dinámicamente el archivo `_navigation.html` en el contenedor con el ID `#navbar-section`.
  - Resalta el enlace del menú correspondiente a la página actual, comparando las rutas.
- **Comentarios adicionales:**
  - Incluye código comentado que podría usarse para pausar videos en un carrusel de Bootstrap al cambiar de diapositiva.

¿Dónde se encuentra?

- En `/public/js/script.js`.
- Está referenciado en varios archivos HTML dentro de la etiqueta `<script src="/public/js/script.js"></script>`.

¿Es propio o de un tercero?

- Es un script propio desarrollado para este proyecto.

Código:

```
document.addEventListener("DOMContentLoaded",function () {



    //Cargar Partial de navegación

    $("#navbar-section").load("/templates/partials/_navigation.html", function () {

        //(Hecho por mi)

        document.querySelectorAll('.navbar a').forEach(link => {

            // Extrae solo el path de la URL del link y de la página actual

            const linkPath = new URL(link.href).pathname;

            const currentPath = window.location.pathname;



            if (linkPath === currentPath) {

                link.classList.add('active');

            }

        });

    }); // Carga el contenido del archivo



    /*

    // Pausar el video cuando se cambia de diapositiva (no hecho por mi)

    $('#myCarousel').on('slide.bs.carousel', function () {

        var activeSlide = $(this).find('.item.active iframe')[0];

        if (activeSlide) {


activeSlide.contentWindow.postMessage('{"event":"command","func":"pauseVideo","args":
""}', '*');

        }

    });

    */

});
```