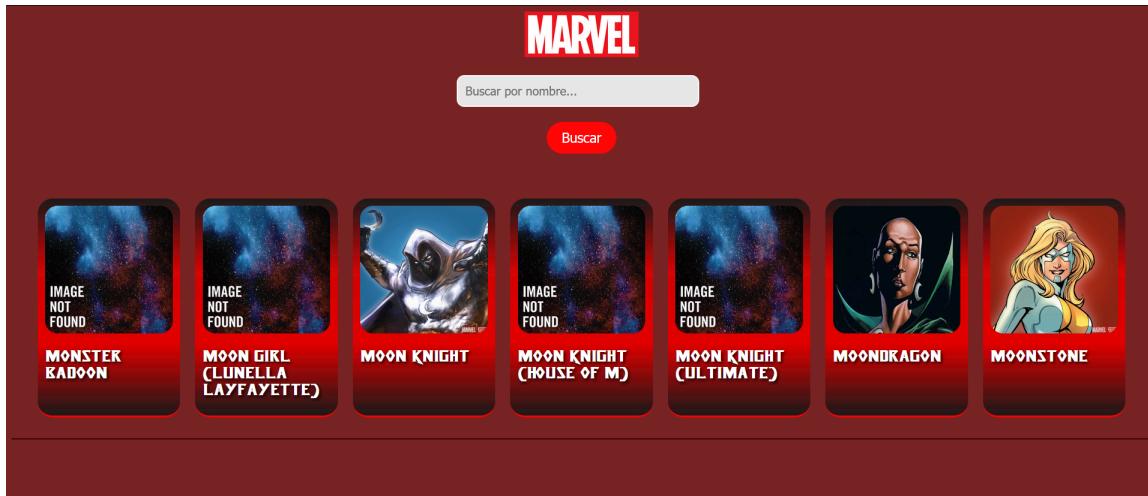
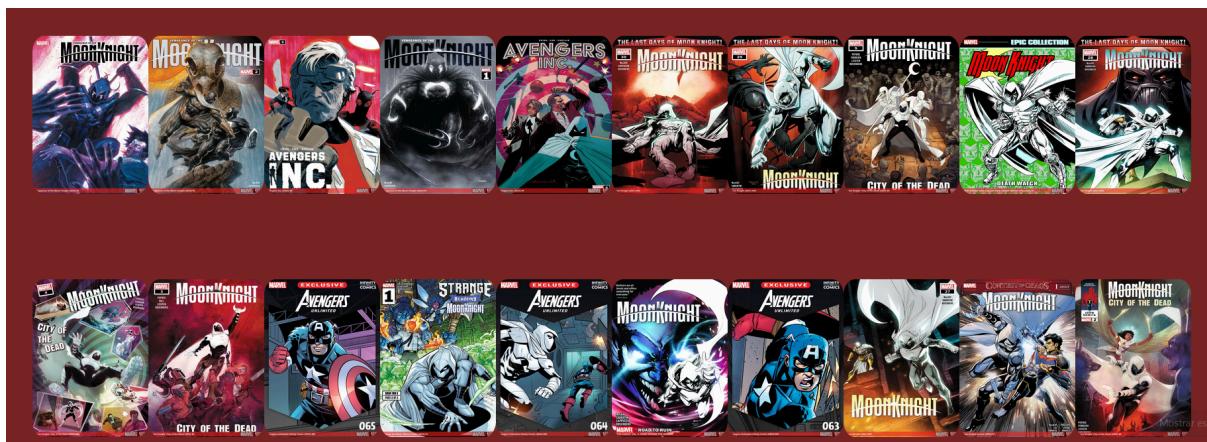


## Proyecto Marvel



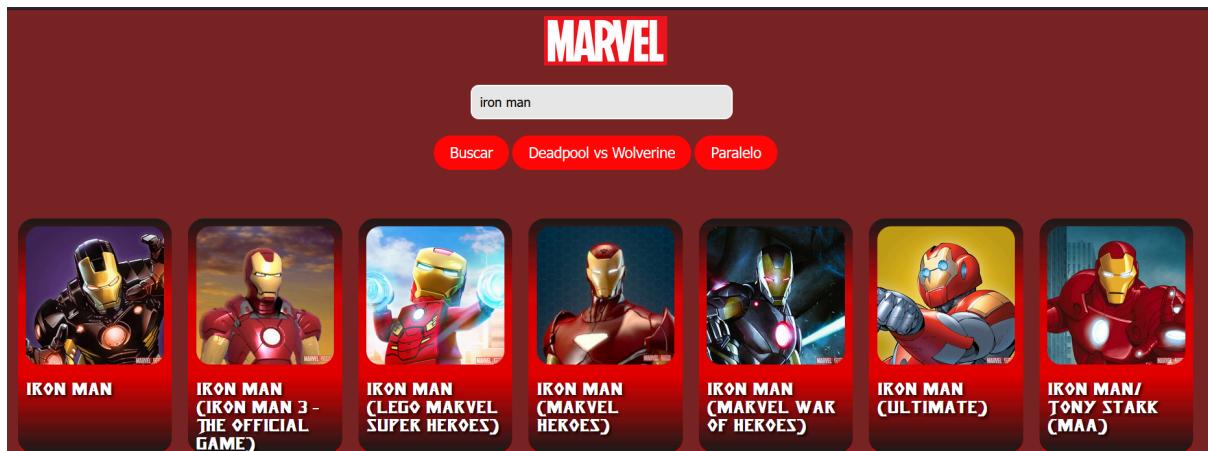
Salen 7 personajes random donde cuando clicas a uno de ellos donde salen 20 comics de ese personaje ya que al clicar coge la id del ese personaje y muestra los comics donde sale



después le das click a uno de los comics y sale un popup con el título de ese comic y la información

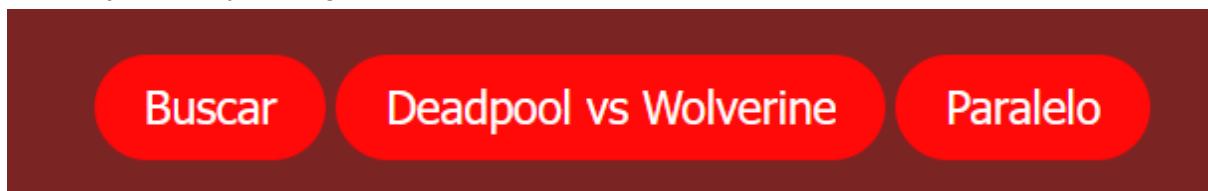


y por último una funcionalidad de búsqueda por ejemplo buscas un personaje y te salen los como máximo 7 con ese nombre



Fetch API

en fetch hay 2 botones extra Deadpool vs Wolverine es promesa tipo race, simplemente sale el primero que llegue y Paralelo que es la promesa tipo paralel, salen cuando todos los personajes se hayan cargado sino no sale



XHR2

esta imagen se pilla por blob binario mediante este código:



```

        xhr.onload = function (e) {
            if (this.status == 200) {
                var blob = this.response;
                var output = document.getElementById('imagenBlob');
                var URL = window.URL || window.webkitURL;
                output.src = URL.createObjectURL(blob);
            }
        };
        xhr.send();
    }

    window.addEventListener("load", inici, true);
}

if (ruta == '/imatge') {
    fs.readFile('./public/img/Marvel_Logo.svg.png', function(err, sortida) {
        if (err) {
            res.writeHead(404, { 'Content-Type': 'text/html' });
            res.write("404 Not Found");
            res.end();
        } else {
            res.writeHead(200, { 'Content-Type': 'image/png' });
            res.write(sortida);
            res.end();
        }
    });
}

```

## WebSockets

Utilizando la librería Socket.io para hacer las llamadas a la api y la función realizarSolicitud

```

io.on("connection", (socket) => {
    console.log("Nueva conexión Socket.IO");

    socket.on("message", (message) => {
        console.log("hola", message);
    });

    const randomOffset = `offset=${Math.round(Math.random() * 1473)}`;
    const url = `http://gateway.marvel.com/v1/public/characters?${limit}&${randomOffset}&ts=${ts}&apikey=${publickey}&hash=${hash}`;
    realizarSolicitud(url, (data) => {
        socket.emit("RespuestaImagenesAleatorias", data);
    });
}

```

```

function realizarSolicitud(url, callback) {
    let protocolo = url.startsWith("https") ? https : http;

    protocolo.get(url, (res) => {
        let data = '';
        res.on('data', (chunk) => {
            data += chunk;
        });
        res.on('end', () => {
            try {
                let jsonData = JSON.parse(data);
                callback(jsonData);
            } catch (error) {
                console.error("Error al analizar JSON:", error);
            }
        });
    }).on('error', (error) => {
        console.error("Error al realizar la solicitud:", error);
    });
}

```