1.1.1. Ejercicios

1. ¿Qué estamos realizando en la línea 6 ? concretamente, ¿para qué se usa el método fs.mkdirSync? Y ¿ el parámetro recursive:true que le hemos pasado ?

-Creamos un directorio. Se usa para crear un directorio de manera síncrona. Para que se cree de manera recursiva.

2. En el módulo fs, se ofrecen métodos para la gestión de ficheros. Entender lo que hacen algunos es sencillo al leer su nombre. Pero hay otros, que no son tan sencillos de comprender, por ejemplo, ¿qué hace el método fs.createWriteStream (línea 17)?

-Crea un archivo, en nuestro caso “png”, en el directorio ”json/leagues/” con el contenido recibido del fetch, la imagen de la liga para ser preciso.

3. Preguntamos sobre la variable res.status que aparece en la línea 16. El código 200 HTTP nos indica que todo ha ido correctamente. Otro cógido famoso es 404 HTTP (404: page not found). Pero, ¿para qué se utilizan estos otros códigos: 302,401,429, 500? (recuerda el código 429, aparecerá más veces)

-El código 302 significa que el recurso de la URI solicitada ha sido cambiado temporalmente.

-El código 401 significa que se necesita autenticar para obtener la respuesta solicitada

-El código 429 significa que el usuario ha enviado demasiadas solicitudes en un periodo de tiempo dado.

-El código 500 significa que el servidor ha encontrado una situación que no sabe cómo manejarla.

4. (Adelantado) ¿Qué hace el método res.body.pipe de la línea 17? (Aclaración: streams, pipe, nodejs)

-El método sirve para juntar un flujo de escritura a uno de lectura con lo que podemos pasar los datos del flujo de lectura al de escritura.

1.3. Ejercicios

```
import fs from 'fs';
import fetch from 'node-fetch'

const writepath = 'flags/'

fs.mkdirSync(writepath, { recursive: true })

try {
    // read leagues file into an array of lines
    let data = fs.readFileSync('nationalities.txt',
'utf8').split(/\r?\n/);
    data.forEach((elem, idx) => {
        let elem1 = elem.split(" ")
        elem1 = elem1.join("%20")
        console.log(elem1)
        const url =
`https://playfootball.games/who-are-ya/media/nations/${elem1}.svg`
        fetch(url)
            .then(res => {
                // check status
                if (res.status == 200) {

res.body.pipe(fs.createWriteStream(`${writepath}${elem}.svg`))
                } else {
                    console.log(`status: ${res.status} line: ${idx}
elem:${elem} not found`)
                }
            })
            .catch(err => console.log(err))
    })
} catch (err) {
    console.error(err);
}
```

1.5. Obten los escudos de los equipos

\json> jq 'sort_by(.teamId) | unique_by(.teamId) | map(.teamId) | .[]' fullplayers.json > teamIds.txt

```
import fs from 'fs';
import fetch from 'node-fetch'

const writepath = 'teamIcons/'

fs.mkdirSync(writepath, { recursive: true })

try {
    // read leagues file into an array of lines
    let data = fs.readFileSync('teamIds.txt', 'utf8').split(/\r?\n/);
    data.forEach((elem, idx) => {
        let id32 = elem % 32
        const url =
`https://cdn.sportmonks.com/images/soccer/teams/${id32}/${elem}.png`
        fetch(url)
            .then(res => {
                // check status
                if (res.status == 200) {

res.body.pipe(fs.createWriteStream(`${writepath}${elem}.png`))
                } else {
                    console.log(`status: ${res.status} line: ${idx}
elem:${elem} not found`)
                }
            })
            .catch(err => console.log(err))
    })
} catch (err) {
    console.error(err);
}
```

1.7. Ejercicios

```
let writepath3 = 'playerIcons/'

fs.mkdirSync(writepath3, { recursive: true })

function getPlayerIcons(data) {
    let elem = data.shift()
    if (elem === undefined){
        clearInterval(idTemp)
    }
    else{
        let e32 = elem % 32
        const url =
`https://playfootball.games/media/players/${e32}/${elem}.png`
        fetch(url)
            .then(res => {
                // check status
                if (res.status == 200) {

res.body.pipe(fs.createWriteStream(`${writepath3}${elem}.png`))
                } else {
                    console.log(`status: ${res.status} elem:${elem} not
found`)
                }
            })
            .catch(err => console.log(err))
    }
}

let idTemp
function temporizador(data){
    idTemp = setInterval(getPlayerIcons, 100, data)
}

try {
    let data = fs.readFileSync('playerIcons.txt',
'utf8').split(/\r?\n/);
    temporizador(data)
} catch (err) {
    console.error(err);
}
```

2.4. Ejercicios

/app.js

```
app.post('/insertar',

  body("id").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("id").isInt().withMessage("Must be an integer"),

  body("name").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("name").isAlpha().withMessage("Must be a string"),

  body("birthdate").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("birthdate").isISO8601().withMessage("Must be a date"),

  body("nationality").isLength({min: 1}).withMessage("Must be at least
1 characters long"),
  body("nationality").isAlpha().withMessage("Must be a string"),

  body("teamId").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("teamId").isInt().withMessage("Must be an integer"),

  body("position").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("position").isIn(['Goalkeeper', 'Defender', 'Midfielder',
'Forward']).withMessage("Must be a valid position"),

  body("number").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("number").isInt().withMessage("Must be an integer"),

  body("leagueId").isLength({min: 1}).withMessage("Must be at least 1
characters long"),
  body("leagueId").isInt().withMessage("Must be an integer"),

  (req, res) => {
  var errors = validationResult(req);
  if (!errors.isEmpty()) {
    console.log(errors);
  } else{
```

```
    console.log(req.body);
    res.send('Recibido');
  }

});
```

/views/index.ejs

```
<body>
    <h1>Insertar jugador</h1>
    <form method="POST" action="/insertar" id="form">
      <label>Id</label>
      <input type="text" name="id">
      <br>
      <label>Name</label>
      <input type="text" name="name">
      <br>
      <label>Birthdate</label>
      <input type="text" name="birthdate">
      <br>
      <label>Nationality</label>
      <input type="text" name="nationality">
      <br>
      <label>TeamId</label>
      <input type="text" name="teamId">
      <br>
      <label>Position</label>
      <input type="text" name="position">
      <br>
      <label>Number</label>
      <input type="text" name="number">
      <br>
      <label>LeagueId</label>
      <input type="text" name="leagueId">
      <br>
      <input type="submit" value="Insertar" name="submit">
    </form>
  </body>
```

4.1. Ejercicios

Hemos utilizado este mismo código para sacar los equipos de todas las ligas, simplemente cambiando la url.

```
import fullLaLiga from './fullLaLiga.json' assert { type: 'json' }
import fullBundesliga from './fullBundesliga.json' assert { type:
'json' }
import fullLigue1 from './fullLigue1.json'  assert { type: 'json' }
import fullPremiere from './fullPremiere.json' assert { type: 'json' }
import fullSerieA from './fullSerieA.json' assert { type: 'json' }
import request from 'request';
import { assert } from 'console';

import mongojs from 'mongojs'
const mongodb = mongojs('mongodb://127.0.0.1:27017/footballdata',
['teams'])

try {
    var data = fullLaLiga.map(elem => elem.newId)
    data.forEach((elem, idx) => {
        var options = {
            'method': 'GET',
            'url': "https://v3.football.api-sports.io/teams?id=" +
elem,
            'headers': {
                'x-rapidapi-host': 'v3.football.api-sports.io',
                'x-rapidapi-key': 'c9be68632b6864fd94ecb8cb6db567b7'
            }
        }
        request(options, function (error, response) {
            if (error) throw new Error(error)
            console.log(JSON.parse(response.body).response[0])

mongodb.teams.insertOne(JSON.parse(response.body).response[0])
        })
    })
} catch (err) {
    console.error(err);
}
```

5.1. Ejercicios

/players.js

```
var express = require('express');
var router = express.Router();

const mongojs = require('mongojs')
const db = mongojs('mongodb://127.0.0.1:27017/footballdata',
['players'])

router.get('/api/v1/players', (req, res) => {
  db.players.find((err, docs) => {
      if (err) {
          res.send(err);
      } else {
          res.render('players', {elements: docs})
      }
  })
})

module.exports = router;
```

/players.ejs

```
<!DOCTYPE html>
<html>
<head>
    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.m
in.css" rel="stylesheet"

integrity="sha384-Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3
URy9Bv1WTRi" crossorigin="anonymous">
</head>
</html>

<button type="button" class="btn btn-primary"
onclick="window.location.href='/api/v1/players/add'">Add
Player</button>
<button type="button" class="btn btn-primary"
onclick="window.location.href='/api/v1/login'">Logout</button>
<table class="table table-striped">
  <thead class="thead-dark">
  <tr>
```

```
      <th>Id</th>
      <th>Name</th>
      <th>Birthdate</th>
      <th>Nationality</th>
      <th>TeamId</th>
      <th>Position</th>
      <th>Number</th>
      <th>LeagueId</th>
  </tr>
  </thead>
  <% elements.forEach( element => { %>
      <tr>
          <td> <%= element.id %></td>
          <td> <%= element.name %></td>
          <td> <%= element.birthdate %></td>
          <td> <%= element.nationality %></td>
          <td> <%= element.teamId %></td>
          <td> <%= element.position %></td>
          <td> <%= element.number %></td>
          <td> <%= element.leagueId %></td>
          <td> <a href="/api/v1/players/edit/<%= element._id
%>">Edit</a></td>
          <td> <a href="/api/v1/players/remove/<%= element._id
%>">Remove</a></td>
      </tr>
  <% }) %>
</table>
```

/add.js

```
var express = require('express');
var router = express.Router();
const { body, validationResult } = require('express-validator');

const mongojs = require('mongojs')
const db = mongojs('mongodb://127.0.0.1:27017/footballdata',
['players'])

router.get('/api/v1/players/add', (req, res) => {
  res.render('add')
})

router.post('/api/v1/players/add',
```

```
  body("id").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
  body("id").isInt().withMessage("Must be an integer"),

  body("name").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
  body("name").isAlpha().withMessage("Must be a string"),

  body("birthdate").isLength({ min: 1 }).withMessage("Must be at least
1 characters long"),
  body("birthdate").isISO8601().withMessage("Must be a date"),

  body("nationality").isLength({ min: 1 }).withMessage("Must be at
least 1 characters long"),
  body("nationality").isAlpha().withMessage("Must be a string"),

  body("teamId").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
  body("teamId").isInt().withMessage("Must be an integer"),

  body("position").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
  body("position").isIn(['GK', 'DF', 'MF', 'FW']).withMessage("Must be
a valid position"),

  body("number").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
  body("number").isInt().withMessage("Must be an integer"),

  body("leagueId").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
  body("leagueId").isInt().withMessage("Must be an integer"),

  (req, res) => {
    var errors = validationResult(req);
    if (!errors.isEmpty()) {
      console.log(errors);
    } else {
      req.body.id = parseInt(req.body.id);
      req.body.teamId = parseInt(req.body.teamId);
      req.body.number = parseInt(req.body.number);
      req.body.leagueId = parseInt(req.body.leagueId);
      db.players.insertOne(req.body, (err, result) => {
```

```
        if (err) {
          res.send(err)
        }
        res.redirect('/api/v1/players');
      })
    }
  })

module.exports = router;
```

/add.ejs

```
<!DOCTYPE html>
<html>
<head>
    <title>Insertar jugador</title>
</head>
<body>
    <h1>Insertar jugador</h1>
    <form method="POST" action="./add" id="form">
        <label>Id</label>
        <input type="text" name="id">
        <br>
        <label>Name</label>
        <input type="text" name="name">
        <br>
        <label>Birthdate</label>
        <input type="text" name="birthdate">
        <br>
        <label>Nationality</label>
        <input type="text" name="nationality">
        <br>
        <label>TeamId</label>
        <input type="text" name="teamId">
        <br>
        <label>Position</label>
        <input type="text" name="position">
        <br>
        <label>Number</label>
        <input type="text" name="number">
        <br>
        <label>LeagueId</label>
        <input type="text" name="leagueId">
```

```
        <br>
        <input type="submit" value="Insertar" name="submit">
    </form>
</body>
</html>
```

/edit.js

```
var express = require('express');
var router = express.Router();
const { body, validationResult } = require('express-validator');

const mongojs = require('mongojs')
const db = mongojs('mongodb://127.0.0.1:27017/footballdata',
['players'])

router.get('/api/v1/players/edit/:id', (req, res) => {
    db.players.findOne({ _id: mongojs.ObjectId(req.params.id) }, (err,
doc) => {
        if (err) {
            res.send(err);
        } else {
            console.log(doc)
            res.render('edit', { element: doc })
        }
    })
})

router.post('/api/v1/players/edit/:id',
    body("id").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
    body("id").isInt().withMessage("Must be an integer"),

    body("name").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
    body("name").isAlpha().withMessage("Must be a string"),

    body("birthdate").isLength({ min: 1 }).withMessage("Must be at
least 1 characters long"),
    body("birthdate").isISO8601().withMessage("Must be a date"),

    body("nationality").isLength({ min: 1 }).withMessage("Must be at
least 1 characters long"),
```

```
    body("nationality").isAlpha().withMessage("Must be a string"),

    body("teamId").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
    body("teamId").isInt().withMessage("Must be an integer"),

    body("position").isLength({ min: 1 }).withMessage("Must be at least
1 characters long"),
    body("position").isIn(['GK', 'DF', 'MF', 'FW']).withMessage("Must
be a valid position"),

    body("number").isLength({ min: 1 }).withMessage("Must be at least 1
characters long"),
    body("number").isInt().withMessage("Must be an integer"),

    body("leagueId").isLength({ min: 1 }).withMessage("Must be at least
1 characters long"),
    body("leagueId").isInt().withMessage("Must be an integer"),

    (req, res) => {
        var errors = validationResult(req);
        if (!errors.isEmpty()) {
            console.log(errors);
        } else {
            db.inventory.findAndModify({
                query: { _id: mongojs.ObjectId(req.params.id) },
                update: { $set: req.body }
            },
                (err, result) => {
                    if (err) {
                        res.send(err);
                    }
                    res.redirect('/api/v1/players');
                })
        }
    })

module.exports = router;
```

/edit.ejs

```
<!DOCTYPE html>
<html>
<head>
    <title>Insertar jugador</title>
</head>
<body>
    <h1>Editar jugador</h1>
    <form method="POST" action="./edit/<%=element._id%>" id="form">
        <label>Id</label>
        <input type="text" name="id" value="<%=element.id%>">
        <br>
        <label>Name</label>
        <input type="text" name="name" value="<%=element.name%>">
        <br>
        <label>Birthdate</label>
        <input type="text" name="birthdate"
value="<%=element.birthdate%>">
        <br>
        <label>Nationality</label>
        <input type="text" name="nationality"
value="<%=element.nationality%>">
        <br>
        <label>TeamId</label>
        <input type="text" name="teamId" value="<%=element.teamId%>">
        <br>
        <label>Position</label>
        <input type="text" name="position"
value="<%=element.position%>">
        <br>
        <label>Number</label>
        <input type="text" name="number" value="<%=element.number%>">
        <br>
        <label>LeagueId</label>
        <input type="text" name="leagueId"
value="<%=element.leagueId%>">
        <br>
        <input type="submit" class="form-submit" value="Insertar"
name="submit">
    </form>
</body>
</html>
```

/remove.js

```
var express = require('express');
var router = express.Router();

const mongojs = require('mongojs')
const db = mongojs('mongodb://127.0.0.1:27017/footballdata',
['players'])

let remove = function(res, id){
    db.players.remove({_id: mongojs.ObjectId(id)}, (err, result) => {
        if (err) {
            res.send(err);
        } else {
            res.redirect("/api/v1/players/");
        }
    })
}

router.get('/api/v1/players/remove/:id', (req, res) => {
    remove(res, req.params.id)
})

module.exports = router;
```

6.1. Ejercicios

login.js

```
var express = require('express');
var router = express.Router();

const mongojs = require('mongojs')
const db = mongojs('mongodb://127.0.0.1:27017/footballdata',
['players'])

router.get('/api/v1/login', (req, res) => {
    res.render('index')
})

router.post('/api/v1/login', (req, res) => {
    db.users.findOne({ name: req.body.name, password: req.body.password
}, (err, result) => {
            if (err) {
                res.send(err);
            } else {
                if (!result) {
                    res.redirect('/api/v1/login')
                } else {
                    if (result.type === 'admin') {
                        res.redirect('/api/v1/players')
                    } else {

res.redirect('/static/milestone1_egina-Milestone5')
                    }
                }
            }
        })
})

module.exports = router;
```

login.ejs

```
<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
</head>
```

```
<body>
    <h1>Login</h1>
    <form method="POST" action="./login" id="form">
        <label>Usuario</label>
        <input type="text" name="name">
        <br>
        <label>Contraseña</label>
        <input type="password" name="password" >
        <br>
        <input type="submit" value="Login">
        <button type="button" class="btn btn-primary"
onclick="window.location.href = '/api/v1/register'">Register</button>
    </form>
</body>
</html>
```

register.js

```
var express = require('express');
var router = express.Router();

const mongojs = require('mongojs')
const db = mongojs('mongodb://127.0.0.1:27017/footballdata', ['users'])

router.get('/api/v1/register', (req, res) => {
    res.render('register')
})

router.post('/api/v1/register', (req, res) => {
    req.body.type = "user"
    db.users.insertOne(req.body, (err, result) => {
        if (err) {
            res.send(err);
        } else {
            res.redirect('/api/v1/login')
        }
    })
})

module.exports = router;
```

register.ejs

```
<!DOCTYPE html>
<html>
<head>
    <title>Register</title>
</head>
<body>
    <h1>Register</h1>
    <form method="POST" action="./register" id="form">
        <label>Nombre</label>
        <input type="text" name="name">
        <br>
        <label>Contraseña</label>
        <input type="text" name="password">
        <br>
        <label>Mail</label>
        <input type="text" name="mail">
        <br>
        <input type="submit" value="Register">
    </form>
</body>
</html>
```