



## SPRINT 2: Desarrollo del Backend

Identificación Proyecto	
Nombre Proyecto:	ScoreBoardPlay
Número Equipo:	1
Integrantes del equipo	
Rol (Líder-Desarrollador – Cliente)	Nombre
Scrum master	Luisa Juliana Carrillo Cacua
Product owner	Brayan Duvan Bernal Sarmiento
development team	Lady Viviana Fandiño Angel
development team	Daniel Francisco Basto Arenas
development team	José Luis Castillo Camacho

## Evidencia construcción del Backend

Como evidencia de la construcción del Backend, se debe presentar capturas de pantalla donde se visualice el proceso de construcción del Backend, como la creación en Spring Boot, modelo, controlador, etc.

```

1  const express = require('express');
2  const server = express();
3  const puerto=3000;
4  const eqRuta = require('./routers/equiposRouter.js')
5  const prRuta = require('./routers/partidosRouter.js')
6  const ctRuta = require('./routers/categoriasRouter.js')
7
8
9  //Crear servidor con "Express"
10 server.get('/',function (req,res){
11   res.send('Pagina Principal');
12   res.writeHead(200);
13 })
14 > /* // Aquí nos llevamos esto a las rutas (otros archivos)...
15
16
17
18
19
20
21
22
23
24 //Rutas para partidos, equipos, categorias, etc., desde el index
25 server.use(express.json())
26 server.use('/equipos', eqRuta)
27 server.use('/partidos', prRuta)
28 server.use('/categorias', ctRuta)
29
30
31 server.listen(puerto, function(){
32   console.log('Servidor Activo.');

```

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node index.js`
Servidor Activo.
  
```


```



```
Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda categoriasController.js - ScorePlay - Visual Studio Code

EXPLORADOR ... JS index.js JS categoriasController.js JS categoriasRouter.js

SCOREPLAY
  controllers
    JS categoriasContr...
    JS equiposContr...
    JS partidosContr...
  node_modules
  routers
    JS categoriasRoute...
    JS equiposRouter.js
    JS partidosRouter.js
    JS index.js
    package-lock.json
    package.json

ESQUEMA
  MySQL

1 const categoriasListar=()=>{
2   //Crear listado clave valor
3   listado = {
4     '1': 'Futbol',
5     '2': 'Basketball',
6     '3': 'Boleiboll',
7     '4': 'Natación',
8     '5': 'Ciclismo',
9     '6': 'Patinaje'
10  }
11  return {listado};
12 }
13
14 const categoriasGuardar = async (req,res)=>{
15   console.log(req.body);
16   const nombre = req.body.nombre//Extraer el nombre de la consulta y guardarlo en variable nombre
17   let respuesta={
18     if(nombre==''){
19       respuesta = {'msj':'Categoria Vacía, el nombre es requerido'}
20       res.status(400).json(respuesta)
21     }else{
22       respuesta = {'msj':'Categoria almacenada con éxito'}
23       res.status(200).json(respuesta)
24     }
25   }
26
27   const categoriasObtener= async (req, res)=>{
28     const id=req.params.id
29     console.log(id)
30     let c
31     if(id == 1)
32       c = {'1': 'Futbol'}
33     if(id == 2)
34       c = {'2': 'Basketball'}
35     res.status(200).json(c)
36   }
37 }
```

```
Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda categoriasRouter.js - ScorePlay - Visual Studio Code

EXPLORADOR ... JS index.js JS categoriasController.js JS categoriasRouter.js X

SCOREPLAY
  controllers
    JS categoriasContr...
    JS equiposContr...
    JS partidosContr...
  node_modules
  routers
    JS categoriasRoute...
    JS equiposRouter.js
    JS partidosRouter.js
    JS index.js
    package-lock.json
    package.json

1 const express = require('express');
2 const router = express.Router();
3 const categoriasControllers=require('../controllers/categoriasController.js');
4
5
6
7 router.get("/",function (req,res){
8   res.send(categoriasControllers.categoriasListar());
9 })
10
11 router.get("/:id", categoriasControllers.categoriasObtener);
12
13 router.post("/", categoriasControllers.categoriasGuardar);
14
15 router.put("/", categoriasControllers.categoriasActualizar);
16
17 router.delete("/:id", categoriasControllers.categoriasEliminar);
18
19 module.exports=router;
```



partidosSchema.js - ScoreBoardPlay - Visual Studio Code

EXPLORADOR

- SCOREBOARDPLAY
  - config
    - database.js
  - controllers
    - categoriasController.js
    - equiposController.js
    - partidosController.js
  - models
    - categoriasSchema.js
    - equiposSchema.js
    - partidosSchema.js
  - node\_modules
  - routers
    - categoriasRouter.js
    - equiposRouter.js
    - partidosRouter.js
  - app.js
  - estilo\_hamburguerMenu.css
  - estiloRegistro.css
  - index.html
  - index.js
  - package-lock.json
  - package.json
  - registro.html

partidosSchema.js

```
1 const mongoose = require("mongoose");
2
3 //Establecer Schema (esquema)
4 const partidosSchema = mongoose.Schema({
5   partido: {
6     type: String,
7     required: true,
8     trim: true,
9   },
10
11   actualizado_en: {
12     (property) default: () => number
13     default: Date.now
14   },
15 });
16
17 module.exports = mongoose.model("partidos", partidosSchema);
18
19
```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL JUPYTER

```
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
Servidor Activo y escuchando.
Conexion exitosa
{ partido: 'Pereira Vs Junior' }
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
Servidor Activo y escuchando.
Conexion exitosa
{ partido: 'America Vs Medellin' }
{ partido: 'Pasto Vs Aguilas Doradas' }
```

MongoDB Compass - localhost:27017/scoreboardplay.partidos

Connect View Collection Help

localhost:27017

Documents

scoreboardplay.pa...

### scoreboardplay.partidos

4 DOCUMENTS 1 INDEXES

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' } OPTIONS FIND RESET ...

ADD DATA VIEW ...

Displaying documents 1 - 4 of 4 REFRESH

```
{
  "_id": ObjectId('63743ec8991173aa04e1ab1b'),
  "partido": "Pereira Vs Junior",
  "actualizado_en": 2022-11-16T01:37:12.273+00:00,
  "_v": 0
}

{
  "_id": ObjectId('63743f3a3715f51ec90d6b10'),
  "partido": "America Vs Medellin",
  "actualizado_en": 2022-11-16T01:39:06.557+00:00,
  "_v": 0
}

{
  "_id": ObjectId('63743f473715f51ec90d6b12'),
  "partido": "Pasto Vs Aguilas Doradas",
  "actualizado_en": 2022-11-16T01:39:19.583+00:00,
  "_v": 0
}
```



## Evidencias de los “endpoint” con el consumo de recursos del API REST

Como evidencia de los “endpoint” donde se visualice el consumo de recursos del API REST.

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** http://localhost:3000/equipos
- Body Type:** JSON
- Response Body (Pretty):**

```
1 {
2   "grupoA": {
3     "1": "Millonarios",
4     "2": "Pereira",
5     "3": "SantaFe",
6     "4": "Junior"
7   },
8   "grupoB": {
9     "1": "Aguilas Doradas",
10    "2": "Medellin",
11    "3": "Pasto",
12    "4": "America"
13  }
```



GET

▼

http://localhost:3000/partidos

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

● none

● form-data

● x-www-form-urlencoded

● raw

● binary

● GraphQL

JSON

▼

1

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

Visualize

JSON

▼

≡

1

2

3

4

5

6

7

8

9

10

11

12

13

"fecha1": {

"Grupo A": [

{

"1": "Millonarios vs Santafe",

"2": "Pereira vs Junior"

}

],

"Grupo B": [

{

"1": "Aguilas Doradas vs America",

"2": "Medellin vs Pasto"

}

]

}

GET

▼

http://localhost:3000/partidos/3

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

Visualize

JSON

▼

≡

1

2

3

"3": "GrupoB: Aguilas Doradas vs America"

Status: 200 OK Time: 9 ms Size: 277 B



POST http://localhost:3000/categorias

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```

1  {
2    "7": "Bolos",
3    "8": "Boxeo",
4    "9": "Karate"
5  }
6

```

Body Cookies Headers (7) Test Results Status: 200 OK Time: 52 ms Size: 275 B

Pretty Raw Preview Visualize JSON

```

1  {
2    "msj": "Categoria almacenada con exito"
3  }

```

EXPLORADOR

- SCOREPLAY
  - controllers
    - JS categoriasController.js
    - JS equiposController.js
    - JS partidosController.js
  - node\_modules
  - routers
    - JS categoriasRouter.js
    - JS equiposRouter.js
    - JS partidosRouter.js
  - index.js
  - package-lock.json
  - package.json

JS index.js JS categoriasController.js JS equiposController.js JS categoriasRouter.js

controllers > JS categoriasController.js > categoriasActualizar

```

1  const categoriasListar=()=>{
2    //Crear listado clave valor
3    listado ={}
4    '1': 'Futbol',
5    '2': 'Basketball',
6    '3': 'Boleibol',
7    '4': 'Natación',
8    '5': 'Ciclismo',
9    '6': 'Patinaje'
10   }
11   return {listado};
12 }
13
14 const categoriasGuardar = async (req,res)=>{
15   console.log(req.body);
16   const nombre = req.body.nombre//Extraer el nombre de la consulta y guardarlo en variable nombre
17   let respuesta={}
18   if(nombre==''){
19     respuesta = {'msj': 'Categoria Vacía, el nombre es requerido'}
20     res.status(400).json(respuesta)
21   }else{
22     respuesta = {'msj': 'Categoria almacenada con exito'}
23     res.status(200).json(respuesta)
24   }
25 }
26
27 const categoriasObtener= async (req, res)=>{
28   const id=req.params.id

```

PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN **TERMINAL** JUPYTER

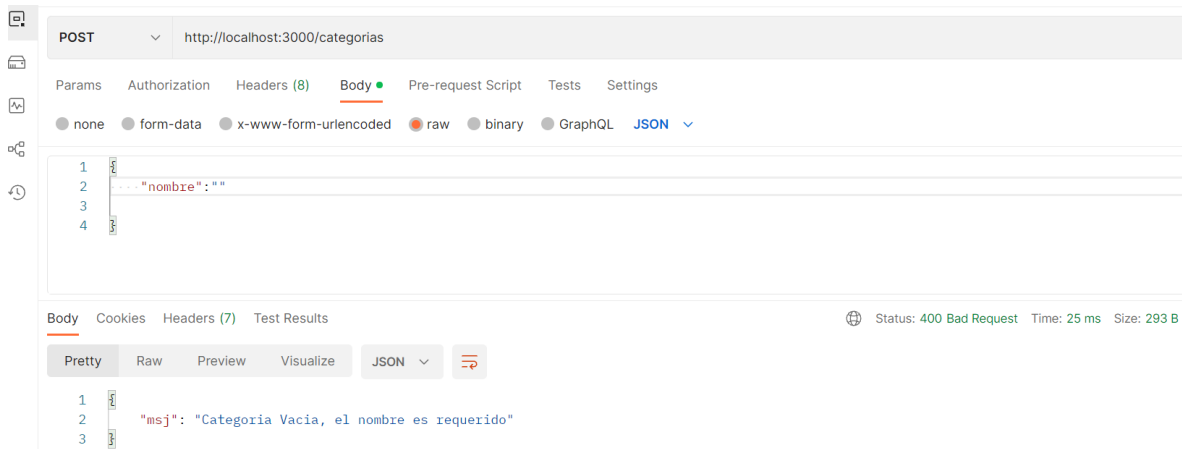
```

[nodeemon] 2.0.20
[nodeemon] to restart at any time, enter `rs`
[nodeemon] watching path(s): *,*
[nodeemon] watching extensions: js,mjs,json
[nodeemon] starting `node index.js`
Servidor Activo.
{ '7': 'Bolos', '8': 'Boxeo', '9': 'Karate' }

```

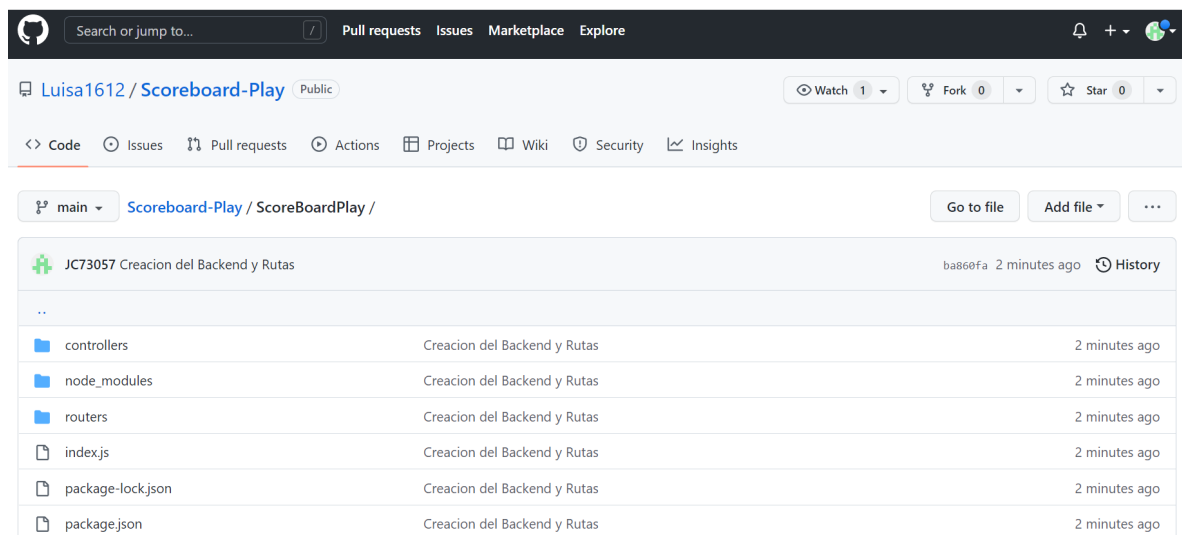
ESQUEMA

MYSQL



### Evidencia GitLab o GitHub

Evidencia de la realización de alguna actualización (commit), donde se visualice la actualización y el historial de actualizaciones (Versión)



### Evidencia JIRA (Seguimiento del proyecto)

Como evidencia del seguimiento del proyecto con la metodología ágil SCRUM, utilizando el software JIRA, se debe presentar capturas de pantalla donde se visualice la ejecución de los Sprint con las historias de usuario relacionadas con el desarrollo del Backend.



Jira Software Tu trabajo Proyectos Filtros Paneles Personas Aplicaciones Crear Q Buscar

Scoreboard Play Proyecto de software

PLANIFICACIÓN

- Hoja de ruta
- Backlog
- Tablero

DESARROLLO

- Código
- Páginas de proyect...
- Añadir acceso rápido
- Configuración del ...

Proyectos / Scoreboard Play

## Backlog

Q JC LC LA B DB Epic

Insights

▼ Sprint 2 ScoreBoardPlay 9 nov – 17 nov (5 incidencias) 0 0 0 Completar sprint

El objetivo es la creación, desarrollo y pruebas de backend (Servidor, Rutas, Controladores, prueba consumo de API's mediante Postman y Base de datos en MongoDB).

SP-16 Creación de Servidor	CONSTRUCCIÓN DEL BACKEND	TAREAS POR HACER	JC
SP-17 Creación de Rutas	CONSTRUCCIÓN DEL BACKEND	TAREAS POR HACER	B
SP-18 Creación de controladores	CONSTRUCCIÓN DEL BACKEND	TAREAS POR HACER	DB
SP-19 Prueba Consumo APIS (Postaman)	CONSTRUCCIÓN DEL BACKEND	TAREAS POR HACER	LC
SP-20 Creación de Base Datos (MongoDB)	CONSTRUCCIÓN DEL BACKEND	TAREAS POR HACER	LA

+ Crear incidencia

Jira Software Tu trabajo Proyectos Filtros Paneles Personas Aplicaciones Crear Q Buscar

Scoreboard Play Proyecto de software

PLANIFICACIÓN

- Hoja de ruta
- Backlog
- Tablero

DESARROLLO

- Código
- Páginas de proyect...
- Añadir acceso rápido
- Configuración del ...

Proyectos / Scoreboard Play

## Backlog

Q JC LC LA B DB Epic

Insights

▼ Sprint 2 ScoreBoardPlay 9 nov – 17 nov (5 incidencias) 0 1 0 Completar sprint

El objetivo es la creación, desarrollo y pruebas de backend (Servidor, Rutas, Controladores, prueba consumo de API's mediante Postman y Base de datos en MongoDB).

SP-16 Creación de Servidor	CONSTRUCCIÓN DEL BACKEND	FINALIZADA	JC
SP-17 Creación de Rutas	CONSTRUCCIÓN DEL BACKEND	FINALIZADA	B
SP-18 Creación de controladores	CONSTRUCCIÓN DEL BACKEND	FINALIZADA	DB
SP-19 Prueba Consumo APIS (Postaman)	CONSTRUCCIÓN DEL BACKEND	FINALIZADA	LC
SP-20 Creación de Base Datos (MongoDB)	CONSTRUCCIÓN DEL BACKEND	FINALIZADA	LA

+ Crear incidencia

Estás en un proyecto gestionado por el equipo

## Evidencias de las Reuniones de Equipo

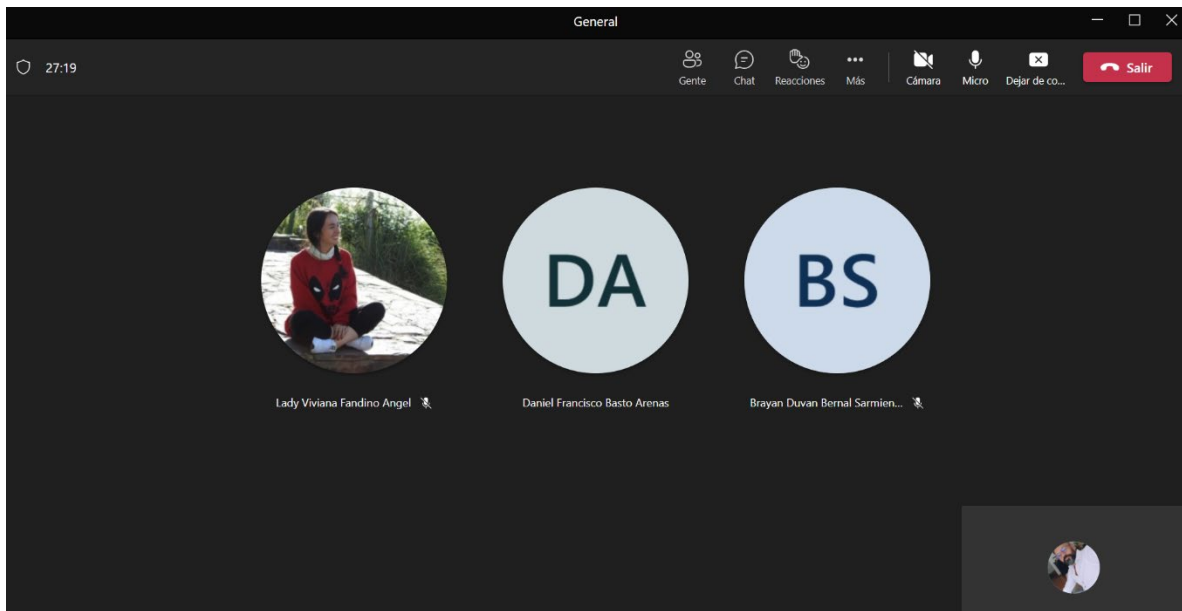
Como evidencia de las reuniones que efectúa el equipo del proyecto, presentar capturas de pantalla de las reuniones efectuadas y si lo consideran pertinente algunas actas de las reuniones.





partidosController.js - Sin título (área de trabajo) - Visual Studio Code

```
1 const partidosListar=()=>{
2   //Crear listado clave valor
3   fecha1 =({
4     "Grupo A": [{
5       '1':'Millonarios vs Santafe',
6       '2':'Pereira vs Junior'
7     }],
8     "Grupo B": [{
9       '1':'Aguilas Doradas vs America',
10      '2':'Medellin vs Pasto',
11    }],
12  })
13
14  fecha2 =({
15    "Grupo A": [{
16      '1':'Santafe vs Pereira',
17      '2':'Junior vs Millonarios',
18    }],
19    "Grupo B": [{
20      '1':'America vs Medellin',
21      '2':'Pasto vs Aguilas Doradas',
22    }],
23  })
24
25  fecha3 =({
26    "Grupo A": [{
27      '1':'Santafe vs Junior',
28      '2':'Millonarios vs Pereira',
29    }],
30    "Grupo B": [{
31      '1':'America vs Pasto',
32      '2':'Medellin vs Aguilas Doradas'
```





Visual Studio Code interface showing the development of a Node.js application for a scoreboard.

**EXPLORADOR (Left Panel):**

- SCOREBOARDPLAY
  - config
  - JS database.js
  - controllers
    - JS categoriasController.js
    - JS equiposController.js
    - JS partidosController.js
  - models
    - JS categoriasSchema.js
    - JS equiposSchema.js
    - JS partidosSchema.js
  - node\_modules
  - routers
    - JS categoriasRouter.js
    - JS equiposRouter.js
    - JS partidosRouter.js
  - app.js
  - estilo\_hamburgerMenu.css
  - estiloRegistro.css
  - index.html
  - index.js
  - package-lock.json
  - package.json
  - registro.html

**partidosSchema.js - ScoreBoardPlay - Visual Studio Code**

```
models > JS partidosSchema.js > @partidosSchema
1 const mongoose = require("mongoose");
2 ....
3 //Establecer Schema (esquema)
4 const partidosSchema = mongoose.Schema(
5   partido: {
6     type: String,
7     required: true,
8     trim: true,
9   },
10
11   actualizado_en: {
12     type: Date,
13     default: Date.now
14   },
15 );
16
17 module.exports = mongoose.model("partidos", partidosSchema);
```

**TERMINAL (Bottom Panel):**

```
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
Servidor Activo y escuchando.
Conexion exitosa
{ partido: 'Pereira Vs Junior' }
[nodemon] restarting due to changes...
[nodemon] starting 'node index.js'
Servidor Activo y escuchando.
Conexion exitosa
{ partido: 'America Vs Medellin' }
{ partido: 'Pasto Vs Aguilas Doradas' }
```

**Video Call Overlay:** Lady Viviana Fandino Angel 2727. Uso de Micro y altavoces del equipo.