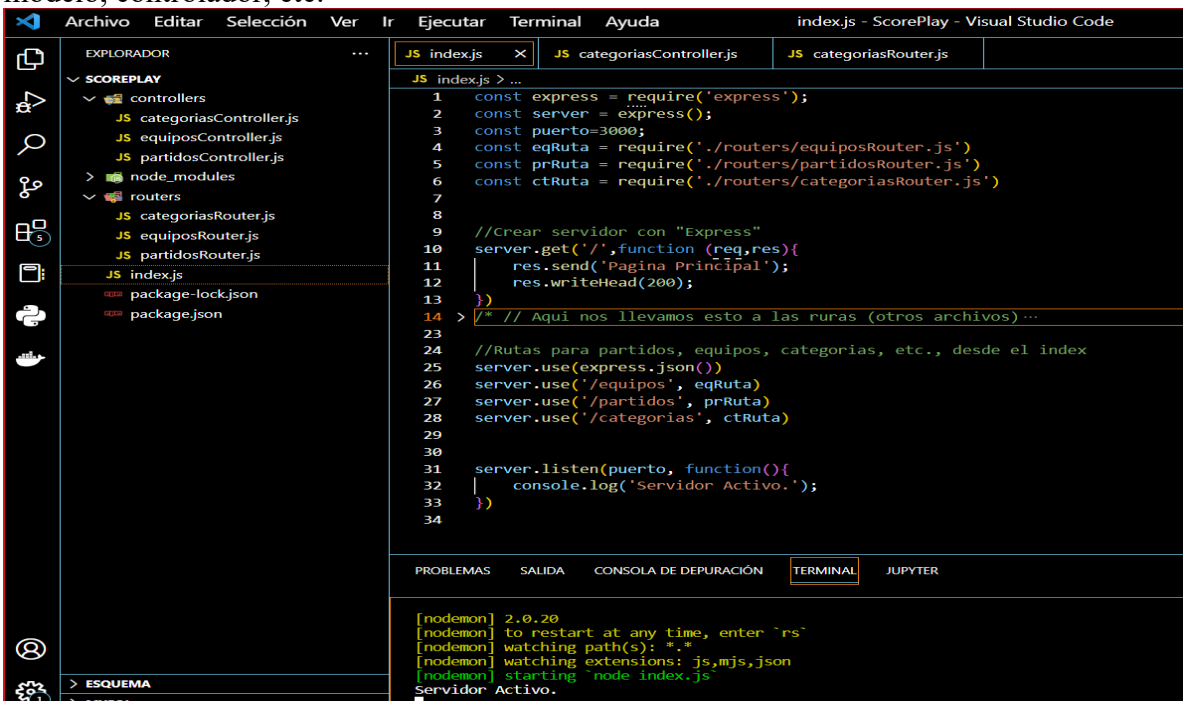
SPRINT 2: Desarrollo del Backend

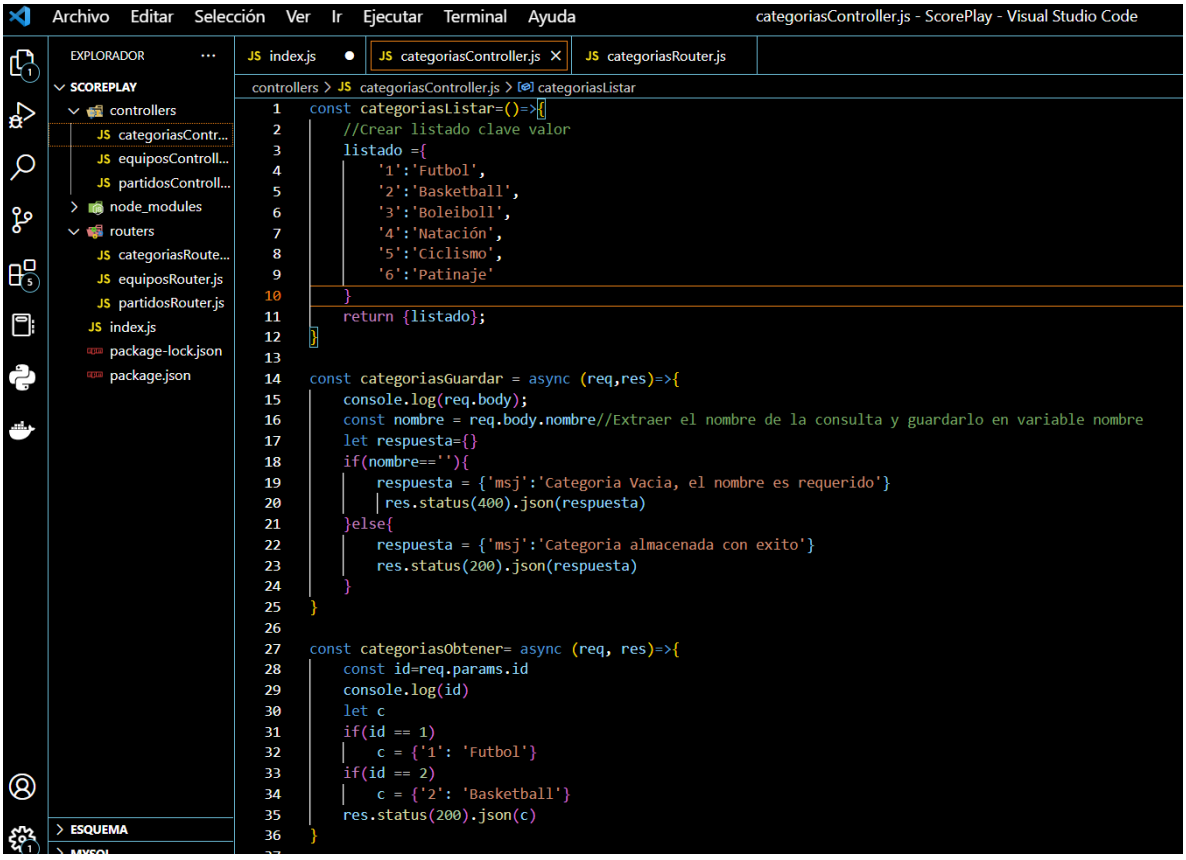
Identificación Proyecto	
Nombre Proyecto:	ScoreBoardPlay
Número Equipo:	1
Integrantes del equipo	
Rol (Líder-Desarrollador – Cliente)	Nombre
Scrum master	Luisa Juliana Carrillo Cacua
Product owner	Brayan Duvan Bernal Sarmiento
development team	Lady Viviana Fandiño Angel
development team	Daniel Francisco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

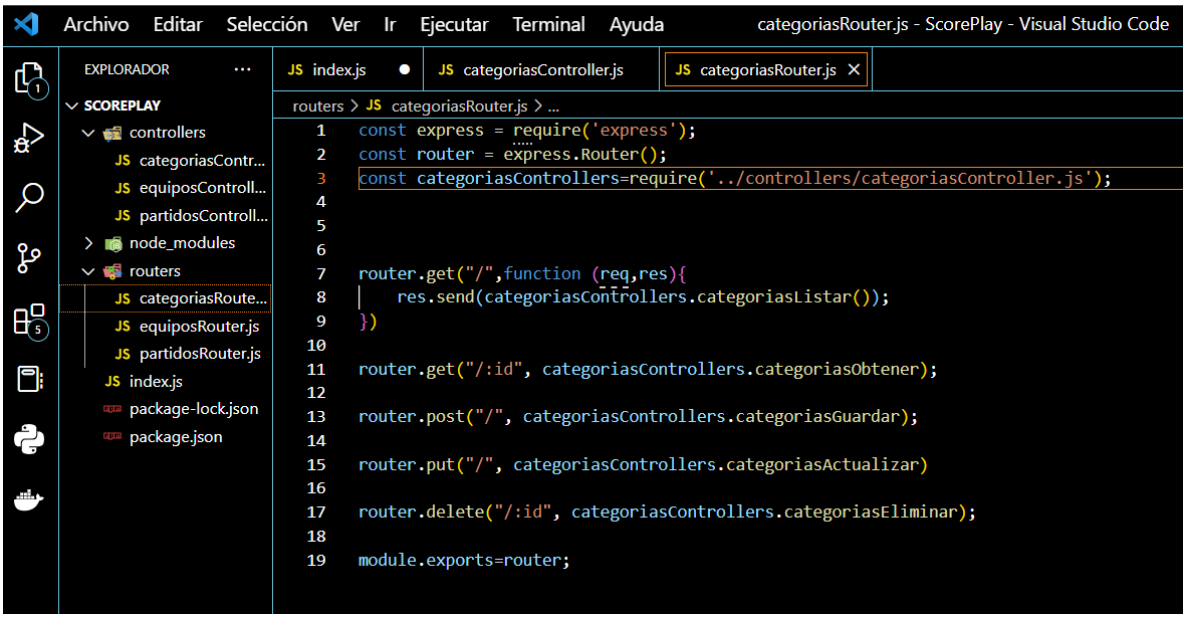
Evidencia construcción del Backend

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.

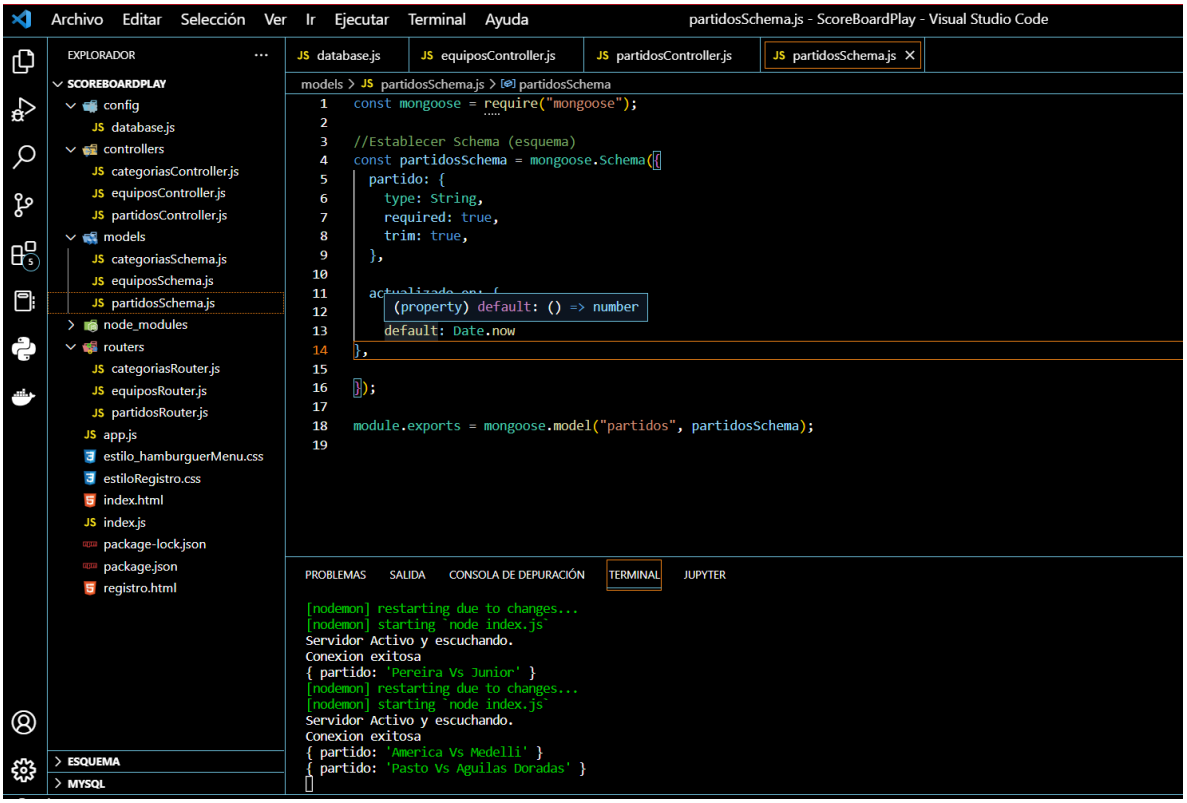


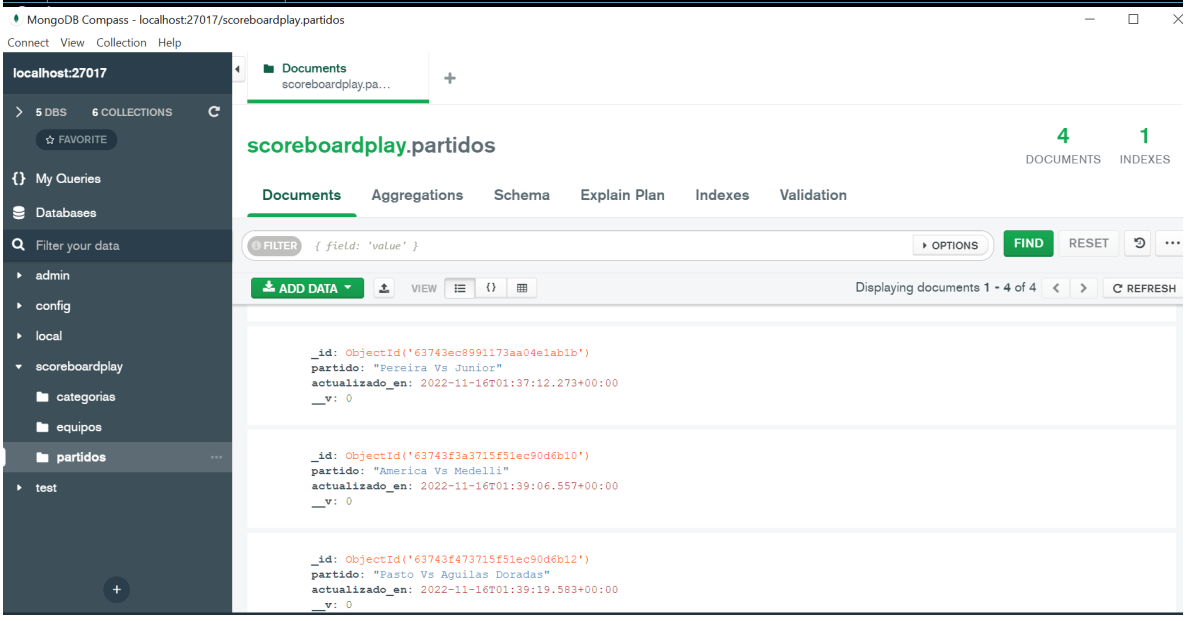

Mision
TIC2022
unab


Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
categoriasController.js - ScorePlay - Visual Studio Code
EXPLORADOR
SCOREPLAY
controllers
categoriasContr...
equiposControll...
partidosControll...
node_modules
routers
categoriasRoute...
equiposRouter.js
partidosRouter.js
index.js
package-lock.json
package.json
ESQUEMA
index.js
categoriasController.js
categoriasRouter.js
controllers > categoriasController.js > categoriasListar
const categoriasListar=()=>{
//Crear listado clave valor
listado ={
'1':'Futbol',
'2':'Basketball',
'3':'Boleiboll',
'4':'Natación',
'5':'Ciclismo',
'6':'Patinaje'
}
return {listado};
}
const categoriasGuardar = async (req,res)=>{
console.log(req.body);
const nombre = req.body.nombre//Extraer el nombre de la consulta y guardarlo en variable nombre
let respuesta={}
if(nombre==''){
respuesta = {'msj':'Categoria Vacia, el nombre es requerido'}
res.status(400).json(respuesta)
}else{
respuesta = {'msj':'Categoria almacenada con exito'}
res.status(200).json(respuesta)
}
}
const categoriasObtener= async (req, res)=>{
const id=req.params.id
console.log(id)
let c
if(id == 1)
c = {'1': 'Futbol'}
if(id == 2)
c = {'2': 'Basketball'}
res.status(200).json(c)
}


Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
categoriasRouter.js - ScorePlay - Visual Studio Code
EXPLORADOR
SCOREPLAY
controllers
categoriasContr...
equiposControll...
partidosControll...
node_modules
routers
categoriasRoute...
equiposRouter.js
partidosRouter.js
index.js
package-lock.json
package.json
index.js
categoriasController.js
categoriasRouter.js
routers > categoriasRouter.js > ...
const express = require('express');
const router = express.Router();
const categoriasControllers=require('../controllers/categoriasController.js');
router.get("/",function (req,res){
res.send(categoriasControllers.categoriasListar());
})
router.get("/:id", categoriasControllers.categoriasObtener);
router.post("/", categoriasControllers.categoriasGuardar);
router.put("/", categoriasControllers.categoriasActualizar)
router.delete("/:id", categoriasControllers.categoriasEliminar);
module.exports=router;

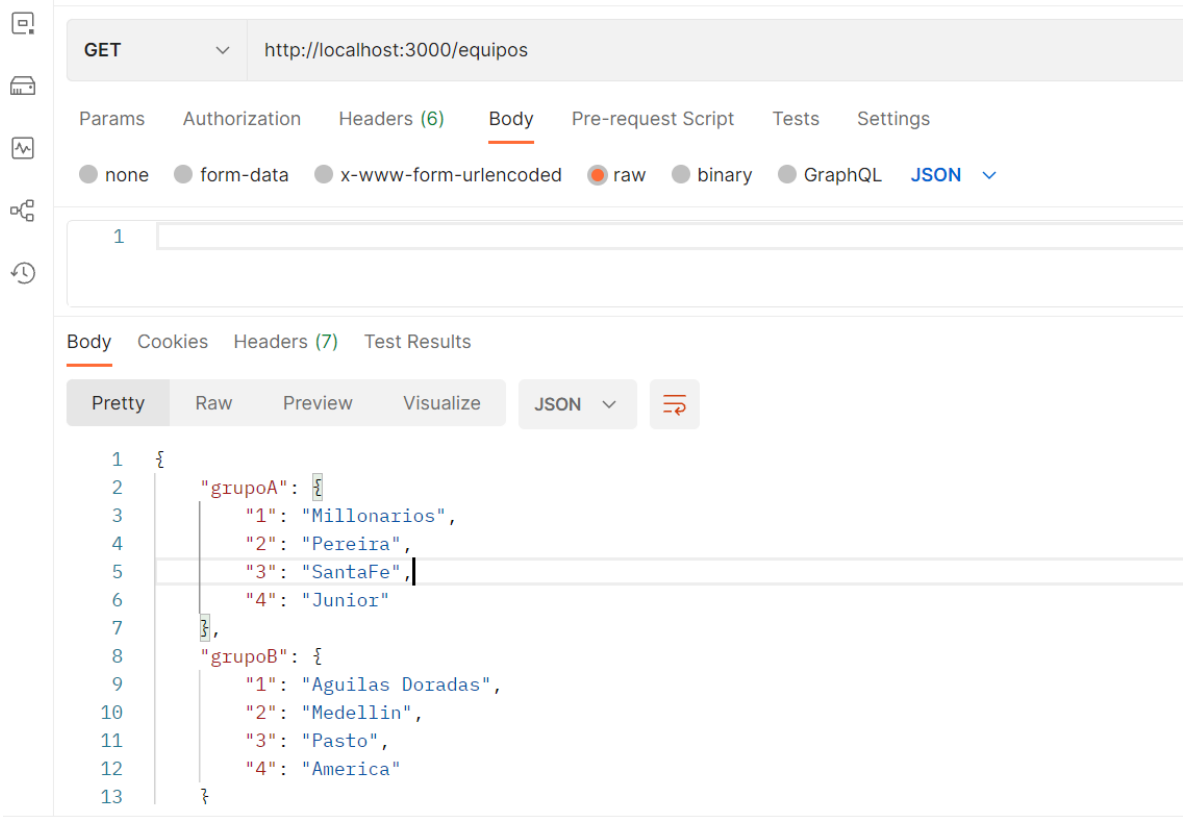

Mision
TIC 2022
unab


Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
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
ESQUEMA
MYSQL
database.js
equiposController.js
partidosController.js
partidosSchema.js
models > 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 (property) default: () => number
13 default: Date.now
14 },
15
16 });
17
18 module.exports = mongoose.model("partidos", partidosSchema);
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 Medelli' }
{ partido: 'Pasto Vs Aguilas Doradas' }


MongoDB Compass - localhost:27017/scoreboardplay.partidos
Connect View Collection Help
localhost:27017
5 DBS 6 COLLECTIONS
FAVORITE
My Queries
Databases
Filter your data
admin
config
local
scoreboardplay
categorias
equipos
partidos
test
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 Medelli"
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.



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


```
{
    "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 Status: 200 OK Time: 9 ms Size: 277 B

Pretty Raw Preview Visualize JSON


```
{
    "3": "GrupoB: Aguilas Doradas vs America"
}
```

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

```
{
    "7":"Bolos",
    "8":"Boxeo",
    "9":"Karate"
}
```

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

Pretty Raw Preview Visualize JSON

```
{
    "msj": "Categoria almacenada con exito"
}
```

EXPLORADOR

SCOREPLAY
- controllers
  - categoriasController.js
  - equiposController.js
  - partidosController.js
- node_modules
- routers
  - categoriasRouter.js
  - equiposRouter.js
  - partidosRouter.js
- index.js
- package-lock.json
- package.json

ESQUEMA
MYSQL

index.js | categoriasController.js | equiposController.js | categoriasRouter.js

controllers > categoriasController.js > categoriasActualizar

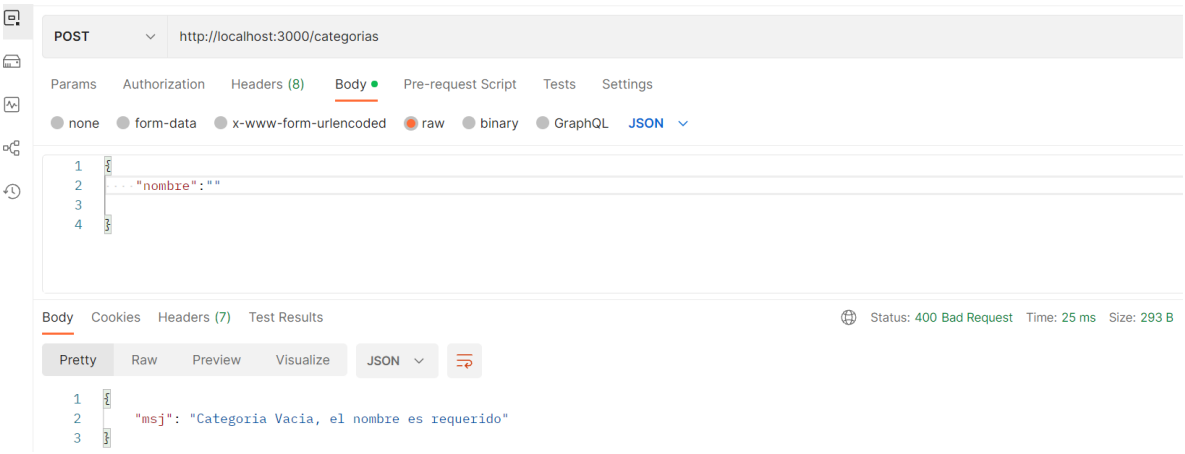
```
const categoriasListar=()=>{
    //Crear listado clave valor
    listado ={
        '1':'Futbol',
        '2':'Basketball',
        '3':'Boleiboll',
        '4':'Natación',
        '5':'Ciclismo',
        '6':'Patinaje'
    }
    return {listado};
}

const categoriasGuardar = async (req,res)=>{
    console.log(req.body);
    const nombre = req.body.nombre//Extraer el nombre de la consulta y guardarlo en variable nombre
    let respuesta={}
    if(nombre==''){
        respuesta = {'msj':'Categoria Vacia, el nombre es requerido'}
        res.status(400).json(respuesta)
    }else{
        respuesta = {'msj':'Categoria almacenada con exito'}
        res.status(200).json(respuesta)
    }
}

const categoriasObtener= async (req, res)=>{
    const id=req.params.id
```

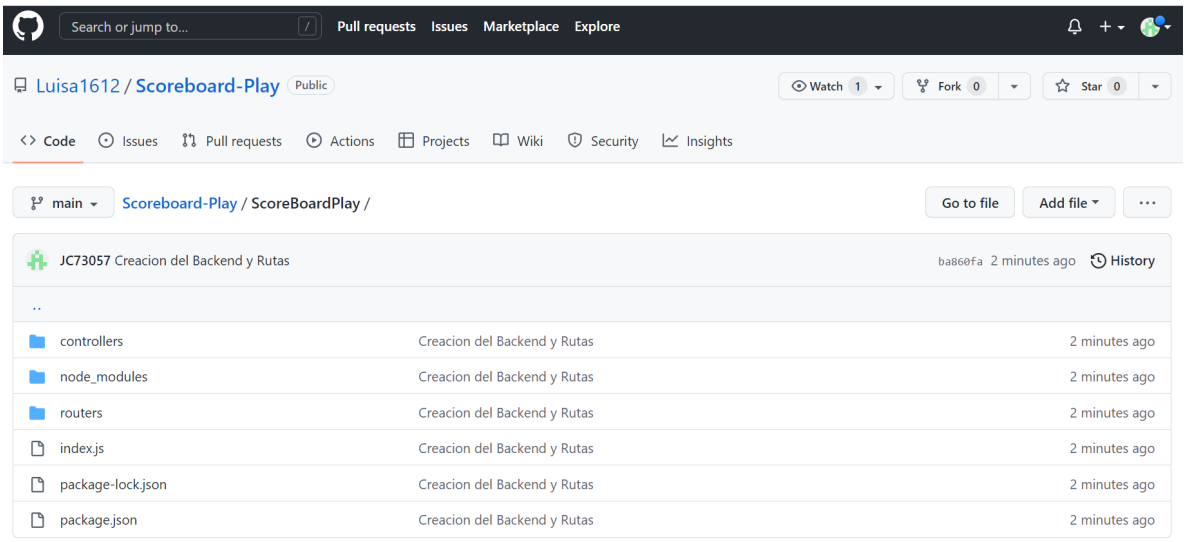
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL JUPYTER

```
[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.
{ '7': 'Bolos', '8': 'Boxeo', '9': 'Karate' }
```



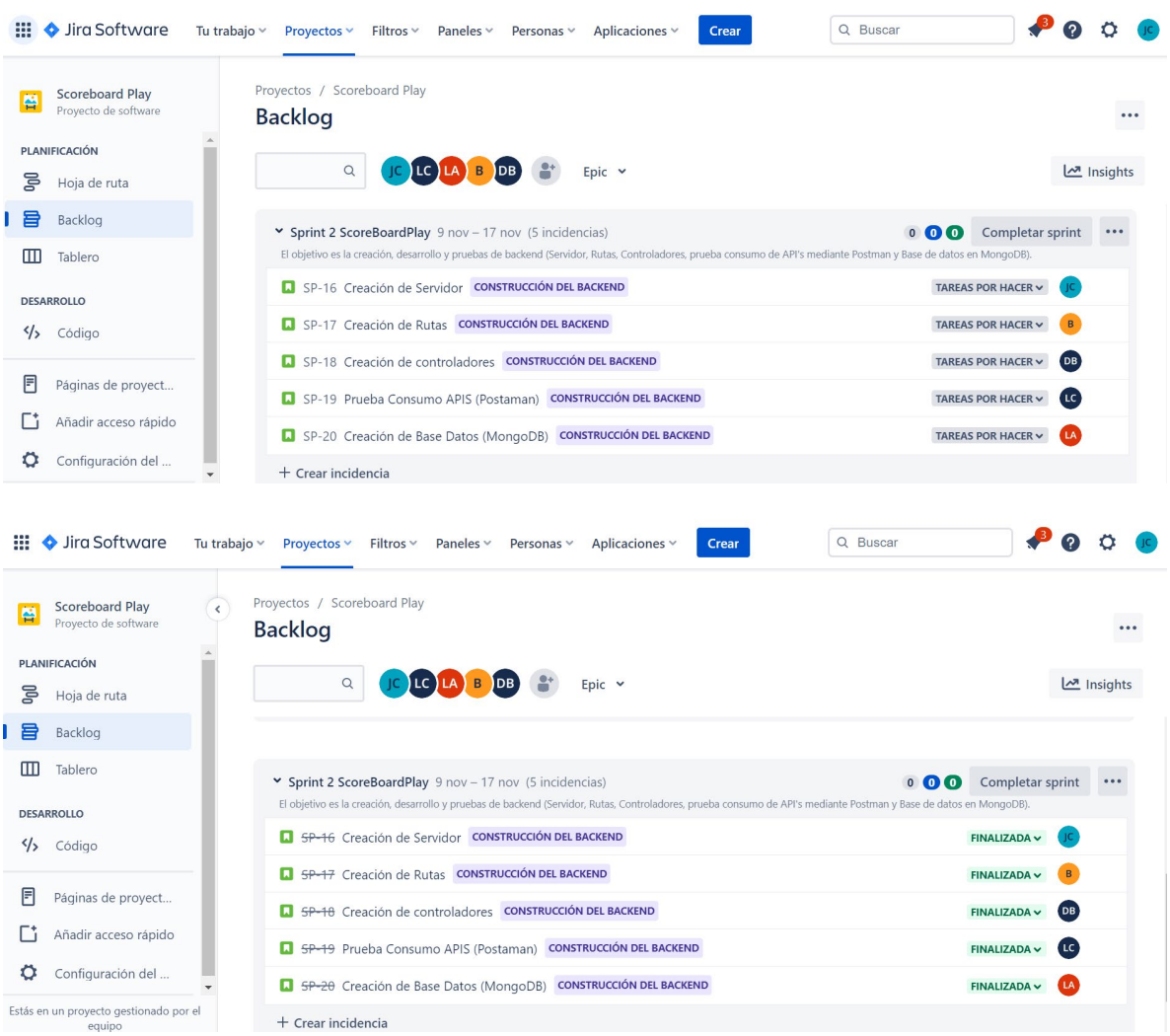
Evidencia GitLab o GitHub

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



Evidencia JIRA (Seguimiento del proyecto)

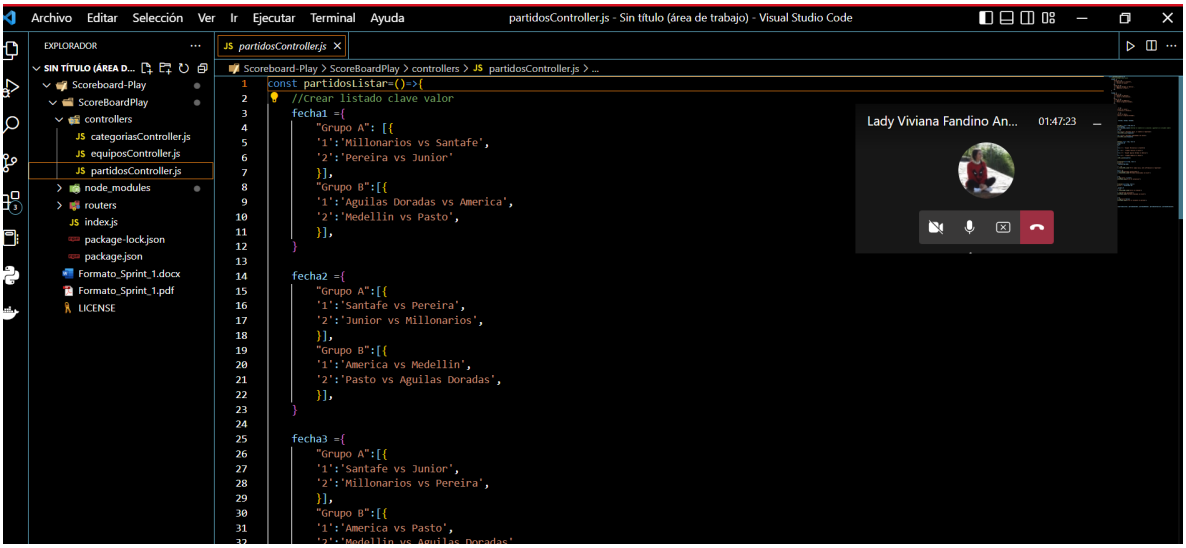
Como evidencia del seguimiento del proyecto con la metodología ágil SCRUM, utilizando el software JIRA, se debe presentar capturas de pantalla donde se visualice la ejecución de los Sprint con las historias de usuario relacionadas con el desarrollo del Backend.

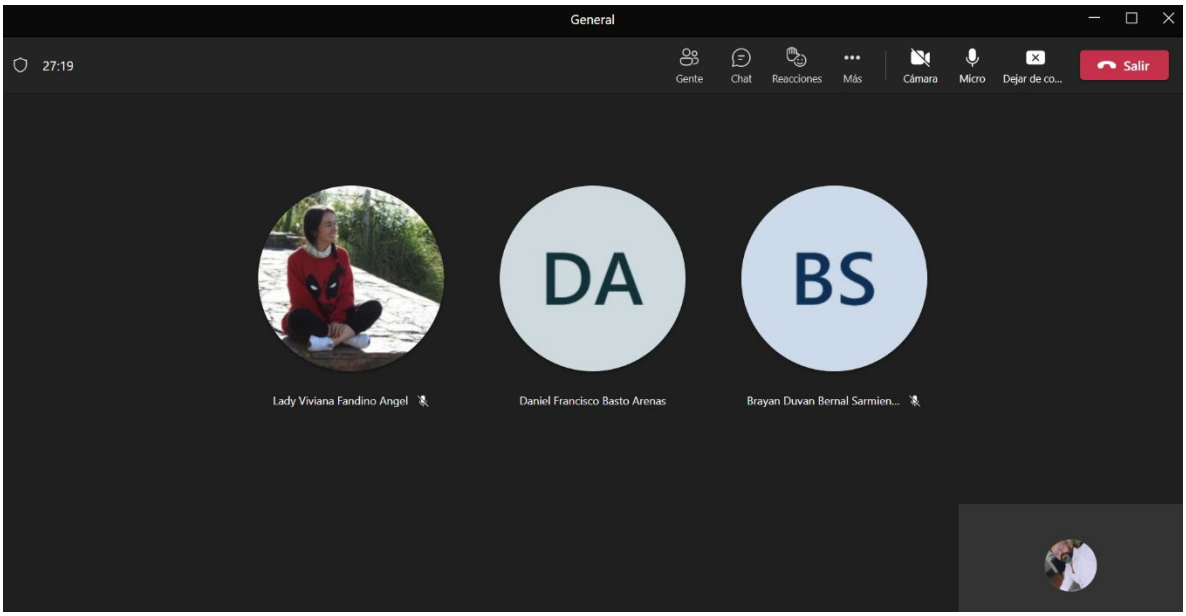


Evidencias de las Reuniones de Equipo

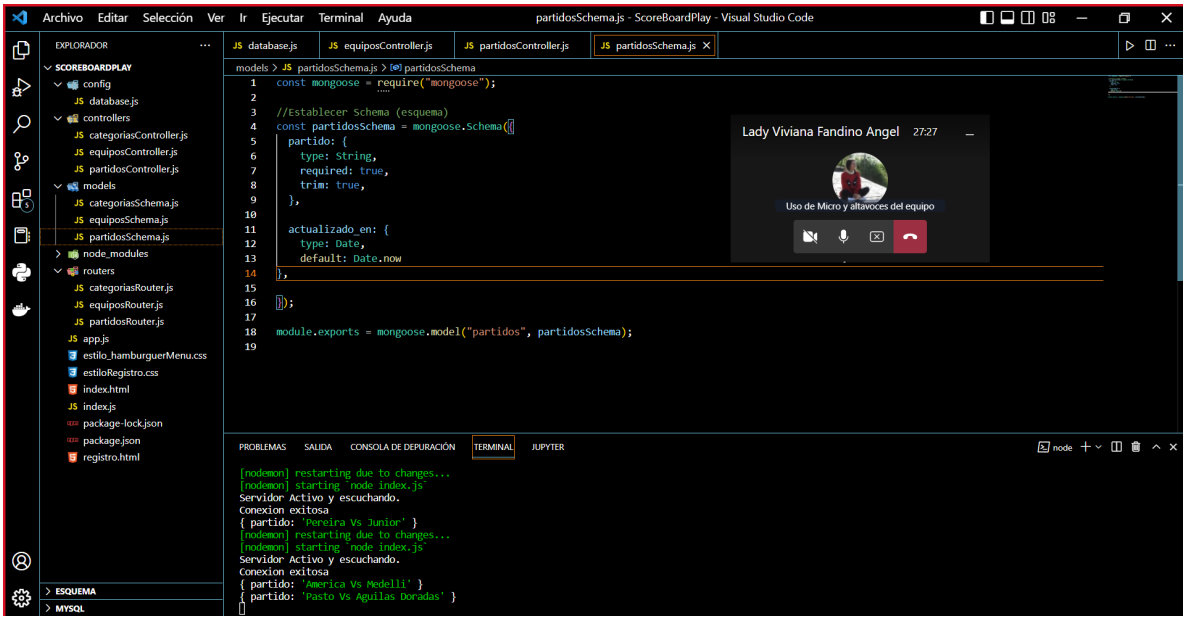
Como evidencia de las reuniones que efectúa el equipo del proyecto, presentar capturas de pantalla de las reuniones efectuadas y si lo consideran pertinente algunas actas de las reuniones.


Mision
TIC2022
unab


Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
partidosController.js - Sin título (área de trabajo) - Visual Studio Code
EXPLORADOR
partidosController.js
Scoreboard-Play > ScoreBoardPlay > controllers > partidosController.js > ...
Lady Viviana Fandino An... 01:47:23
const partidosListar=()=>{
//Crear listado clave valor
fecha1 ={
"Grupo A": [{
'1':'Millonarios vs Santafe',
'2':'Pereira vs Junior'
}],
"Grupo B":[{
'1':'Aguilas Doradas vs America',
'2':'Medellin vs Pasto',
}],
}
fecha2 ={
"Grupo A":[{
'1':'Santafe vs Pereira',
'2':'Junior vs Millonarios',
}],
"Grupo B":[{
'1':'America vs Medellin',
'2':'Pasto vs Aguilas Doradas',
}],
}
fecha3 ={
"Grupo A":[{
'1':'Santafe vs Junior',
'2':'Millonarios vs Pereira',
}],
"Grupo B":[{
'1':'America vs Pasto',


General
27:19
Gente
Chat
Reacciones
Más
Cámara
Micro
Dejar de co...
Salir
DA
BS
Lady Viviana Fandino Angel
Daniel Francisco Basto Arenas
Brayan Duvan Bernal Sarmien...


Mision
TIC 2022
unab


Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda
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
ESQUEMA
MYSQL
database.js
equiposController.js
partidosController.js
partidosSchema.js
models > partidosSchema.js > partidosSchema
1 const mongoose = require("mongoose");
3 //Establecer Schema (esquema)
4 const partidosSchema = mongoose.Schema({
5 partido: {
6 type: String,
7 required: true,
8 trim: true,
9 },
11 actualizado_en: {
12 type: Date,
13 default: Date.now
14 },
16 });
18 module.exports = mongoose.model("partidos", partidosSchema);
Lady Viviana Fandino Angel 27:27
Uso de Micro y altavoces del equipo
PROBLEMAS SALIDA CONSOLA DE DEPURACIÓN TERMINAL JUPYTER
node
[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 Medelli' }
{ partido: 'Pasto Vs Aguilas Doradas' }