



SPRINT 2: Desarrollo del Backend

Identificación Proyecto	
Nombre Proyecto:	Deportes
Número Equipo:	
Integrantes del equipo	
Rol (Líder-Desarrollador – Cliente)	Nombre
Líder	Cristhiam Felipe Mora
Desarrollador	Jorge Moreno
Desarrollador	Juan Álvarez
Desarrollador	Doris Aroca
Desarrollador	Felipe Álvarez
Desarrollador	Brandon Ortiz Herrera
Cliente	Alvaro Stid Bolaños Vásquez

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 mongoose=require('mongoose')
3 const bodyParser=require('body-parser')
4 const cors=require('cors')
5 const routes=require('./routes')
6
7 const app=express()
8
9 mongoose.Promise=global.Promise
10 mongoose.connect(
11   'mongodb://127.0.0.1/deportes',
12   {useNewUrlParser:true,}
13 )
14
15 //habilitar el bodyParser
16 app.use(bodyParser.json())
17 app.use(bodyParser.urlencoded({extended:true}))
18
19 app.use(cors())
20
21 app.use('/',routes())
22
23 app.listen(5000,()=>{
24   console.log('server listen in: 5000')
25 })
26

```

```

> backendMongoG15-G2 start
> nodemon index.js

[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
server listen in: 5000

```



Archivo Editar Selección Ver Ir Ejecutar Terminal Ayuda

EXPLORADOR ... JS index.js U JS deportesController.js U JS deportes.js U JS eventos.js U

EDITOR... JS index.js rou... U
X JS deportesCo... U
JS deportes.js... U
JS eventos.js... U
X JS usuarios.js... U
JS eventosCo... U

BACKEN... JS deportesContr... U
JS equiposControl... U
JS eventosControl... U
JS usuariosContro... U

models JS deportes.js U
JS equipos.js U
JS eventos.js U
JS usuarios.js U

```
models > JS usuarios.js > [?] <unknown>
1  const mongoose=require('mongoose')
2  const Schema=mongoose.Schema
3
4  const usuariosSchema=new Schema({
5    usu_email:{type:String,Trim:true,unique:true,lowercase:true},
6    usu_clave:{type:String,Trim:true},
7    usu_nombres:{type:String,Trim:true},
8    usu_apellidos:{type:String,Trim:true}
9  })
10
11  module.exports=mongoose.model('usuarios',usuariosSchema)
```



```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda

EXPLORADOR  ...  JS index.js U  JS deportesController.js U  JS deportes.js U  JS eventos.js U

EDITORES ABIERTOS
  JS index.js rou... U
  JS deportesCo... U
  JS deportes.js... U
  JS eventos.js... U
  X JS usuariosCo... U
  JS eventosCo... U

  BACKEN... [?] [U] [C] [D] ...
    controllers
      JS deportesContr... U
      JS equiposControl... U
      JS eventosControl... U
      JS usuariosContro... U
    models
      JS deportes.js U
      JS equipos.js U
      JS eventos.js U
      JS usuarios.js U
    > node_modules
    > routes
  JS index.js U
  {} package-lock.json U
  {} package.json U

controllers > JS usuariosController.js > update > update
1  const usuarios=require('../models/usuarios')
2
3  //primera accion listar todos
4  exports.list=async(req,res)=>{
5
6      try{
7
8          const colUsuarios=await usuarios.find({})
9          res.json(colUsuarios)
10         }catch(error){
11
12             console.log(error)
13             res.send(error)
14             next()
15         }
16     }
17
18     //primera accion ingresar todos
19     exports.add=async(req,res)=>{
20         const usuario=new usuarios(req.body)
21         try{
22
23             await usuario.save()
24             res.json({message:'nuevo usuario agregado'})
25         }catch(error){
26
27             console.log(error)
28             res.send(error)
29             next()
30         }
```

```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda

EXPLORADOR  ...  JS index.js U  JS deportesController.js U  JS deportes.js U  JS eventos.js U

EDITORES ABIERTOS
  JS index.js rou... U
  JS deportesCo... U
  JS deportes.js... U
  JS eventos.js... U
  X JS equipos.js... U
  JS eventosCo... U

  BACKEN... [?] [U] [C] [D] ...
    controllers
      JS deportesContr... U
      JS equiposControl... U
      JS eventosControl... U
      JS usuariosContro... U
    models
      JS deportes.js U
      JS equipos.js U

models > JS equipos.js > [?] <unknown>
1  //Este Schema ayuda que la información conserve esta estructura
2  const mongoose=require('mongoose')
3  const Schema=mongoose.Schema
4
5  const equiposSchema=new Schema({
6      equ_nombre:{type:String, Trim:true,unique:true},
7  })
8
9  //se utiliza porque va ser llamado desde otro modulo
10 module.exports=mongoose.model('equipos',equiposSchema)
```



```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda  ← →

EXPLORADOR  ...  JS index.js U  JS deportesController.js U  JS deportes.js U  JS even...

EDITORES ABIERTOS
  JS index.js rou... U
  JS deportesCo... U
  JS deportes.js... U
  JS eventos.js... U
  X JS equiposCon... U
  JS eventosCo... U

BACKEN...  [?]  [?]  [?]  ...
  controllers
    JS deportesContr... U
    JS equiposControl... U
    JS eventosControl... U
    JS usuariosContro... U
  models
    JS deportes.js U
    JS equipos.js U
    JS eventos.js U
    JS usuarios.js U
  > node_modules
  > routes
  JS index.js U
  {} package-lock.json U
  {} package.json U

controllers > JS equiposController.js > list > list
4 //primera accion listar todos
5 exports.list=async(req,res)=>{
6
7   try{
8
9     const colequipos=await equipos.find({})
10    res.json(colequipos)
11  }catch(error){
12    console.log(error) //mostrar el error
13    res.send(error) //envia el error al navegador
14    next() //si hay un error que lo retorne
15  }
16 }
17
18 //primera accion ingresar todos
19 exports.add=async(req,res)=>{
20   const equipo=new equipos(req.body)
21   try{
22
23     await equipo.save()
24     res.json({message:'nuevo equipo agregado'})
25   }catch(error){
26     console.log(error) //mostrar el error
27     res.send(error) //envia el error al navegador
28     next() //si hay un error que lo retorne
29   }
30 }
31
32 //primera accion actualizar todos
33 exports.update=async(req,res, next)=>{
```



```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda

EXPLORADOR  ...  JS index.js U  JS deportesController.js U  JS deportes.js U X  JS eventos.js U

EDITORES ABIERTOS
  JS index.js rou... U
  JS deportesCo... U
  X JS deportes.js... U
  JS eventos.js... U
  JS equipos.js... U
  JS eventosCo... U

BACKENDMON...  ...
  controllers
    JS deportesContr... U
    JS equiposControl... U
    JS eventosControl... U
    JS usuariosContro... U
  models
    JS deportes.js U

models > JS deportes.js > [?] <unknown>
1  const mongoose=require('mongoose')
2  const Schema=mongoose.Schema
3
4  const deportesSchema=new Schema({
5    dep_nombre:{type:String, Trim:true,unique:true},
6  })
7
8  //se utiliza porque va ser llamado desde otro modulo
9  module.exports=mongoose.model('deportes',deportesSchema)
```

```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda

EXPLORADOR  ...  JS index.js U  JS deportesController.js U X  JS deportes.js U  JS eventos.js U

EDITORES ABIERTOS
  JS index.js routes U
  X JS deportesController... U
  JS deportes.js models U
  JS eventos.js models U
  JS equiposControllerj... U
  JS eventosController... U

BACKENDMON...  ...
  controllers
    JS deportesController.js U
    JS equiposController.js U
    JS eventosController.js U
    JS usuariosController.js U
  models
    JS deportes.js U
    JS equipos.js U
    JS eventos.js U
    JS usuarios.js U
  > node_modules
  > routes
  JS index.js U
  {} package-lock.json U
  {} package.json U

controllers > JS deportesController.js > add > add > [?] deporte
1  const deportes=require('../models/deportes')
2
3  //primera accion listar todos
4  exports.list=async(req,res)=>{
5
6    try{
7
8      const colDeportes=await deportes.find({})
9      res.json(colDeportes)
10    }catch(error){
11      console.log(error)
12      res.send(error)
13      next()
14    }
15  }
16
17  //primera accion ingresar todos
18  exports.add=async(req,res,next)=>{
19    const deporte=new deportes(req.body)
20    try{
21
22      await deporte.save()
23      res.json({message:'nuevo deporte agregado'})
24    }catch(error){
25      console.log(error)
26      res.send(error)
27      next()
28    }
29  }
30
```



```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda

EXPLORADOR  ...  JS index.js U  JS deportesController.js U  JS deportes.js U  JS eventos.js U

EDITORES ABIERTOS
  JS index.js rou... U
  JS deportesCo... U
  JS deportes.js... U
  X JS eventos.js... U
  JS equipos.js... U
  JS eventosCo... U

BACKENDMON...  ...
  controllers
    JS deportesContr... U
    JS equiposControl... U
    JS eventosControl... U
    JS usuariosContro... U
  models
    JS deportes.js U
    JS equipos.js U
    JS eventos.js U
    JS usuarios.js U

models > JS eventos.js > [?] <unknown>
1  const mongoose=require('mongoose')
2  const Schema=mongoose.Schema
3  const Deportes=mongoose.model('deportes')
4  const Equipos=mongoose.model('equipos')
5  const Usuarios=mongoose.model('usuarios')
6
7  const eventosSchema=new Schema({
8    eve_fecha:{type:Date,unique:true},
9    equ_equipo1:{type: Schema.ObjectId, ref: "equipos"},
10   equ_equipo2:{type: Schema.ObjectId, ref: "Equipos"},
11   eve_marca1:{type:Number},
12   eve_marca2:{type:Number},
13   dep_id:{type: Schema.ObjectId, ref: "Deportes"},
14   eve_descrip:{type:String,Trim:true},
15   usu_id:{type: Schema.ObjectId, ref: "Usuarios"}
16 })
17
18 module.exports=mongoose.model('eventos',eventosSchema)
```

```
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda

EXPLORADOR  ...  JS index.js U  JS deportesController.js U  JS deportes.js U  JS eventos.js U

EDITORES ABIERTOS
  JS index.js routes U
  JS deportesControl... U
  JS deportes.js models U
  JS eventos.js models U
  JS equiposController.j... U
  X JS eventosController... U

BACKENDMON...  ...
  controllers
    JS deportesController.js U
    JS equiposController.js U
    JS eventosController.js U
    JS usuariosController.js U
  models
    JS deportes.js U
    JS equipos.js U
    JS eventos.js U
    JS usuarios.js U
  > node_modules
  > routes
  JS index.js U
  {} package-lock.json U
  {} package.json U

controllers > JS eventosController.js > ...
1  const eventos=require('../models/eventos')
2
3
4
5  //primera accion listar todos
6  exports.list=async(req,res)=>{
7
8    try{
9
10     const colEventos=await eventos.find({})
11     res.json(colEventos)
12   }catch(error){
13
14     console.log(error)
15     res.send(error)
16     next()
17   }
18 }
19
20
21 //primera accion ingresar todos
22 exports.add=async(req,res,next)=>{
23   const evento=new eventos(req.body)
24   try{
25
26     await evento.save()
27     res.json({message:'nuevo evento agregado'})
28   }catch(error){
29
30     console.log(error)
```



The screenshot shows a VS Code editor with a project structure on the left and the `routes/index.js` file open in the main editor. The project structure includes:

- EDITORES ABIERTOS**
 - `JS index.js routes`
 - `JS deportesControlle...`
 - `JS deportes.js models`
 - `JS eventos.js models`
 - `JS equiposControllerj...`
 - `JS eventosController...`
- BACKENDMON...**
 - controllers**
 - `JS deportesController.js`
 - `JS equiposController.js`
 - `JS eventosController.js`
 - `JS usuariosController.js`
 - models**
 - `JS deportes.js`
 - `JS equipos.js`
 - `JS eventos.js`
 - `JS usuarios.js`
 - node_modules**
 - routes**
 - `JS index.js` (selected)
 - `JS index.js`
 - `{ package-lock.json`
 - `{ package.json`

The `routes/index.js` file contains the following code:

```

1 const express=require('express')
2 const router=express.Router()
3 const usuariosController=require('../controllers/usuariosController')
4 const equiposController=require('../controllers/equiposController')
5 const deportesController=require('../controllers/deportesController')
6 const eventosController=require('../controllers/eventosController')
7
8 module.exports=()=>{
9   //llamado get de usuarios
10  router.get('/usuarios',usuariosController.list)
11
12  //llamado post de usuarios
13  router.post('/usuarios',usuariosController.add)
14
15  //llamado put de usuarios
16  router.put('/usuarios/:id',usuariosController.update)
17
18  //llamado delete de usuarios
19  router.delete('/usuarios/:id',usuariosController.delete)
20
21  //llamado get de equipos
22  router.get('/equipos',equiposController.list)
23
24  //llamado post de equipos
25  router.post('/equipos',equiposController.add)
26
27  //llamado put de equipos
28  router.put('/equipos/:id',equiposController.update)
29
30  //llamado delete de equipos

```

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 the Postman API client interface. The left sidebar displays a list of collections, including 'BackendMongo' and 'Rent_Maquinaria'. The main area shows a GET request to `http://localhost:5000/usuarios` with a status of 200 OK. The response body is displayed in JSON format:

```

1 {
2   "_id": "63894d9948de0a129252db1",
3   "usu_email": "ara@gmail.com",
4   "usu_clave": "1234",
5   "usu_nombre": "ara",
6   "usu_apellidos": "xxxx"
7 },
8 {
9   "_id": "63894f1348de0a129252db3",
10  "usu_email": "pepe@gmail.com",
11  "usu_clave": "1234",
12  "usu_nombre": "pepe",
13  "usu_apellidos": "eeee"
14 },
15 }

```



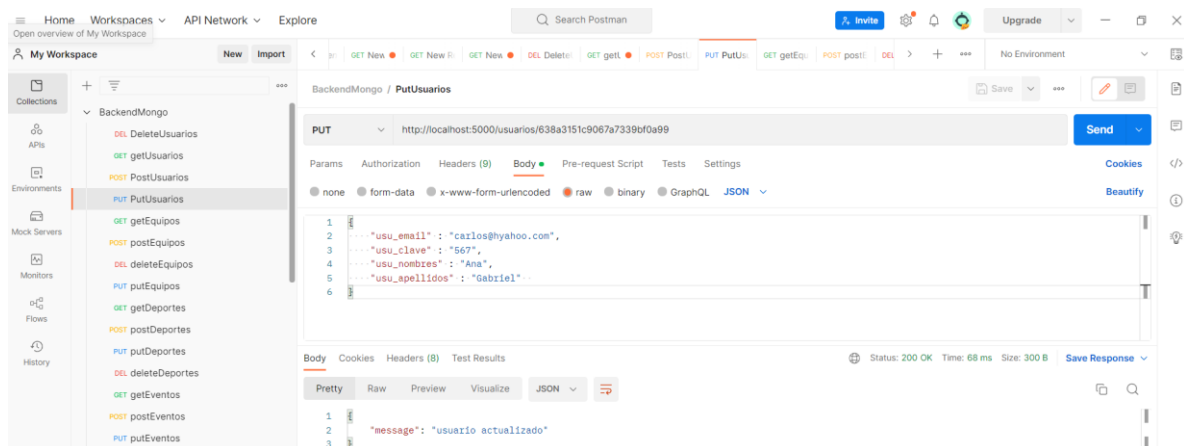
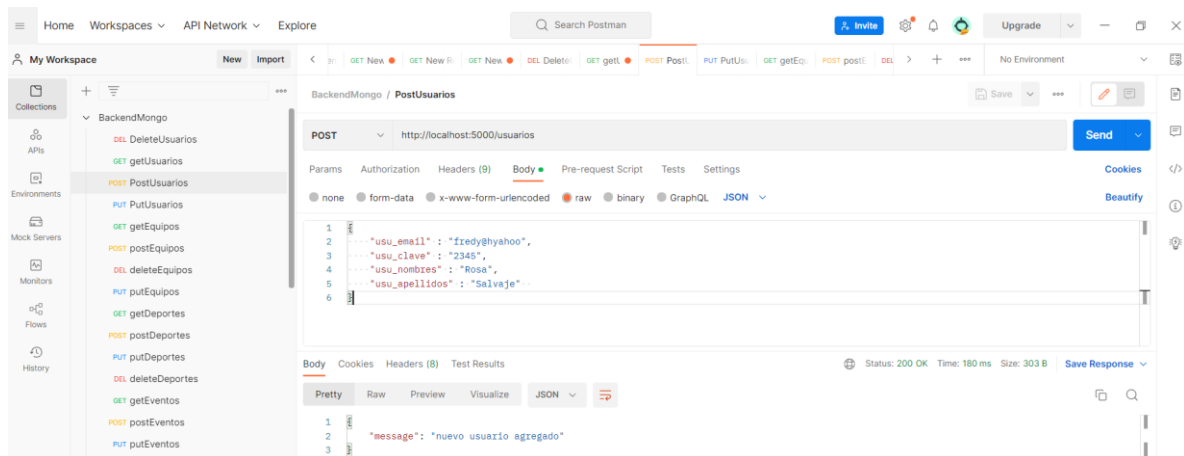
```
[
  {
    "_id": "63894d9048de0aa129252db1",
    "usu_email": "ara@gmail.com",
    "usu_clave": "123",
    "usu_nombre": "ara",
    "usu_apellidos": "xxx"
  },
  {
    "_id": "63894f1348de0aa129252db3",
    "usu_email": "pepe@gmail.com",
    "usu_clave": "1234",
    "usu_nombres": "pepe",
    "usu_apellidos": "eeee"
  },
  {
    "_id": "638a22fdc9067a7339bf0a96",
    "usu_email": "prueba@com",
    "usu_clave": "123456",
    "usu_nombres": "zeus",
    "usu_apellidos": "aro"
  },
  {
    "_id": "638a2560c9067a7339bf0a97",
    "usu_email": "ejemploOne1@com",
    "usu_clave": "1234567",
    "usu_nombres": "lana",
    "usu_apellidos": "hijo1"
  },
  {
    "_id": "638a3151c9067a7339bf0a98",
    "usu_email": "ejemploMany1@com",
    "usu_clave": "12",
    "usu_nombres": "lanaMany1",
    "usu_apellidos": "hijoMany1"
  },
  {
    "_id": "638a3151c9067a7339bf0a99",
    "usu_email": "carlos@hyahoo.com",
    "usu_clave": "567",
    "usu_nombres": "Ana",
    "usu_apellidos": "Gabriel"
  }
]
```




```

    },
    {
      "_id": "6396ad1a854b1f55789a83b2",
      "usu_email": "fredy@hyahoo",
      "usu_clave": "2345",
      "usu_nombres": "Rosa",
      "usu_apellidos": "Salvaje",
      "__v": 0
    }
  ]

```





Postman interface showing a DELETE request to `http://localhost:5000/usuarios/638fd50c143452b5e5c343c`. The response status is 200 OK, and the body contains:

```
1 {
2   "message": "usuario eliminado"
3 }
```

Postman interface showing a GET request to `http://localhost:5000/equipos`. The response status is 200 OK, and the body contains:

```
1 [
2   {
3     "_id": "6396cbddae2ab5fd42b6f715",
4     "equ_nombre": "Millonarios",
5     "__v": 0
6   },
7   {
8     "_id": "63980b23bef17843aa9037cf",
9     "equ_nombre": "Medellin",
10    "__v": 0
11  }
12 ]
```

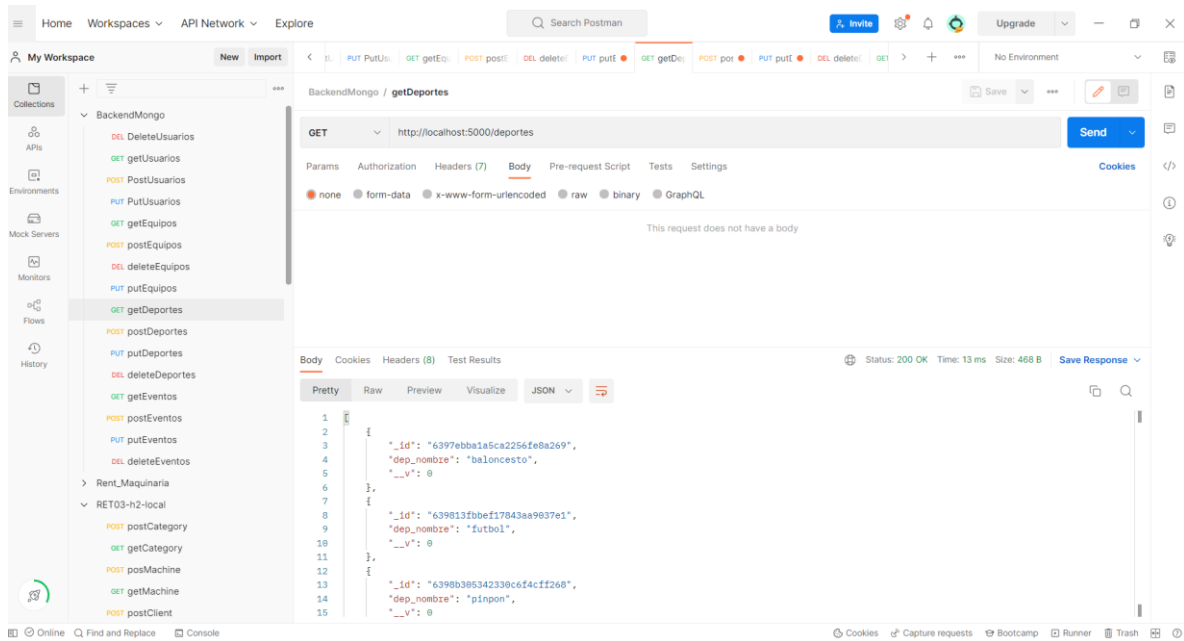
```
[
  {
    "_id": "6396cbddae2ab5fd42b6f715",
    "equ_nombre": "Millonarios",
    "__v": 0
  },
  {
    "_id": "63980b23bef17843aa9037cf",
    "equ_nombre": "Medellin",
    "__v": 0
  }
]
```



Postman interface showing a REST client request for **BackendMongo / postEquipos**. The request is a **POST** to `http://localhost:5000/equipos`. The body is a JSON object: `{ "equ_nombre": "Redellin" }`. The response status is **200 OK** with a message: `"message": "nuevo equipo agregado"`.

Postman interface showing a REST client request for **BackendMongo / putEquipos**. The request is a **PUT** to `http://localhost:5000/equipos/6396bbbf351e4201476663c`. The body is a JSON object: `{ "equ_nombre": "América" }`. The response status is **200 OK** with a message: `"message": "equipo actualizado"`.

Postman interface showing a REST client request for **BackendMongo / deleteEquipos**. The request is a **DELETE** to `http://localhost:5000/equipos/6396bbbf351e4201476663c`. The response status is **200 OK** with a message: `"message": "equipo eliminado"`.



```
[
  {
    "_id": "6397ebba1a5ca2256fe8a269",
    "dep_nombre": "baloncesto",
    "__v": 0
  },
  {
    "_id": "639813fbbef17843aa9037e1",
    "dep_nombre": "futbol",
    "__v": 0
  },
  {
    "_id": "6398b305342330c6f4cff268",
    "dep_nombre": "pinpon",
    "__v": 0
  }
]
```



Home Workspaces API Network Explore Search Postman

My Workspace

BackendMongo / postDeportes

POST http://localhost:5000/deportes

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Body

```
1 {
2   "dep_nombre": "pinpon"
3 }
```

Status: 200 OK Time: 13 ms Size: 303 B Save Response

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "nuevo deporte agregado"
3 }
```

Home Workspaces API Network Explore Search Postman

My Workspace

BackendMongo / putDeportes

PUT http://localhost:5000/deportes/63980ecabef17843aa903796

Params Authorization Headers (9) Body Pre-request Script Tests Settings

Body

```
1 {
2   "dep_nombre": "pinpon"
3 }
```

Status: 200 OK Time: 22 ms Size: 300 B Save Response

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "deporte actualizado"
3 }
```

Home Workspaces API Network Explore Search Postman

My Workspace

BackendMongo / deleteDeportes

DELETE http://localhost:5000/deportes/6397ac5ba8d459a73ccc288

Params Authorization Headers (7) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Status: 200 OK Time: 40 ms Size: 298 B Save Response

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "deporte eliminado"
3 }
```



Home Workspaces API Network Explore Search Postman

My Workspace New Import

BackendMongo / getEventos

GET http://localhost:5000/eventos

Params Authorization Headers (7) Body Pre-request Script Tests Settings

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

This request does not have a body

Status: 200 OK Time: 37 ms Size: 566 B Save Response

Body Cookies Headers (8) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "_id": "639836e1342330c6f4cfff263",
4     "eve_fecha": "2021-11-16T13:30:00.000Z",
5     "equipo1": "63988b23bef17843aa9837cf",
6     "equipo2": "6396cbdae2ab5f42b6f715",
7     "eve_marca1": "7",
8     "eve_marca2": "5",
9     "dep_id": "639813fbef17843aa9837e1",
10    "eve_descrip": "COPA COLOMBIA",
11    "usu_id": "6396ad1a854b1f55789a83b2",
12    "...v": 0
13  }
14 ]
```

Online Find and Replace Console Cookies Capture requests Bootcamp Runner Trash

Home Workspaces API Network Explore Search Postman

My Workspace New Import

BackendMongo / postEventos

POST http://localhost:5000/eventos

Params Authorization Headers (9) Body Pre-request Script Tests Settings

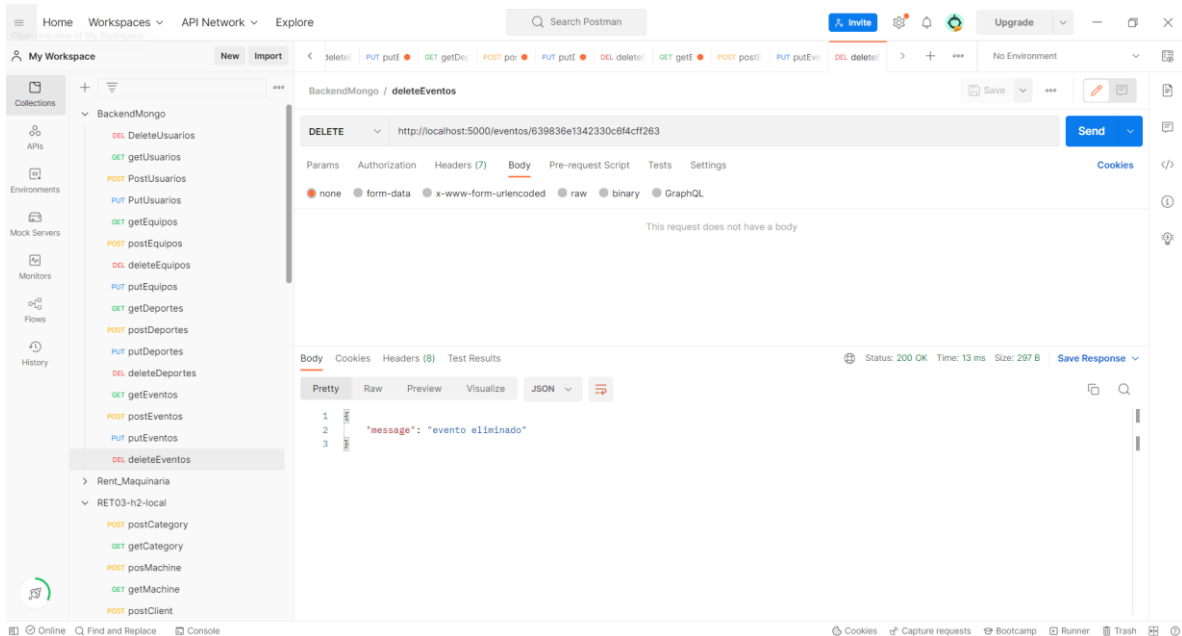
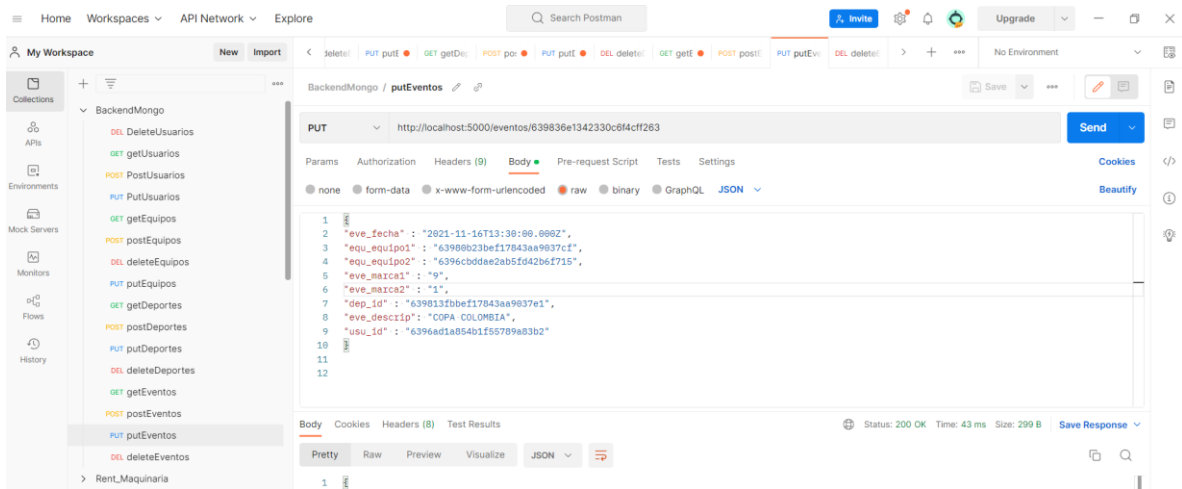
none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "eve_fecha": "2021-11-16T13:30:00.000Z",
3   "equipo1": "63988b23bef17843aa9837cf",
4   "equipo2": "6396cbdae2ab5f42b6f715",
5   "eve_marca1": "7",
6   "eve_marca2": "5",
7   "dep_id": "639813fbef17843aa9837e1",
8   "eve_descrip": "COPA COLOMBIA",
9   "usu_id": "6396ad1a854b1f55789a83b2"
10 }
```

Status: 200 OK Time: 62 ms Size: 302 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "message": "nuevo evento agregado"
3 }
```



Evidencia GitLab o GitHub

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

url del proyecto Github:

<https://github.com/Daroca-2022/deportes.git>



github.com/Daroca-2022/deportes/tree/doris

Search or jump to... Pull requests Issues Codespaces Marketplace Explore

Daroca-2022 / deportes Public

Pin Unwatch 2 Fork 1 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

doris had recent pushes 3 minutes ago [Compare & pull request](#)

doris 5 branches 0 tags [Go to file](#) [Add file](#) [Code](#)

This branch is 2 commits ahead of main. [Contribute](#)

Daroca-2022 backendMongoG15-G2 Sprint2		89a4a96 6 minutes ago 3 commits
backendMongoG15-G2	backendMongoG15-G2 Sprint2	6 minutes ago
.gitignore	backendMongoG15-G2 Sprint2	6 minutes ago
README.md	Initial commit	8 days ago
Sprint1_Diligenciado_C4-G15-DESAP...	documento sprin1	7 days ago

README.md

About

No description, website, or topics provided.

Readme

0 stars

2 watching

1 fork

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

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.

mingetele.atlassian.net/jira/software/projects/AMD/boards/3/backlog

Jira Software Tu trabajo Proyectos Filtros Paneles Personas Aplicaciones Crear

Proyectos / APLICACION MARCADOR DEPORTES

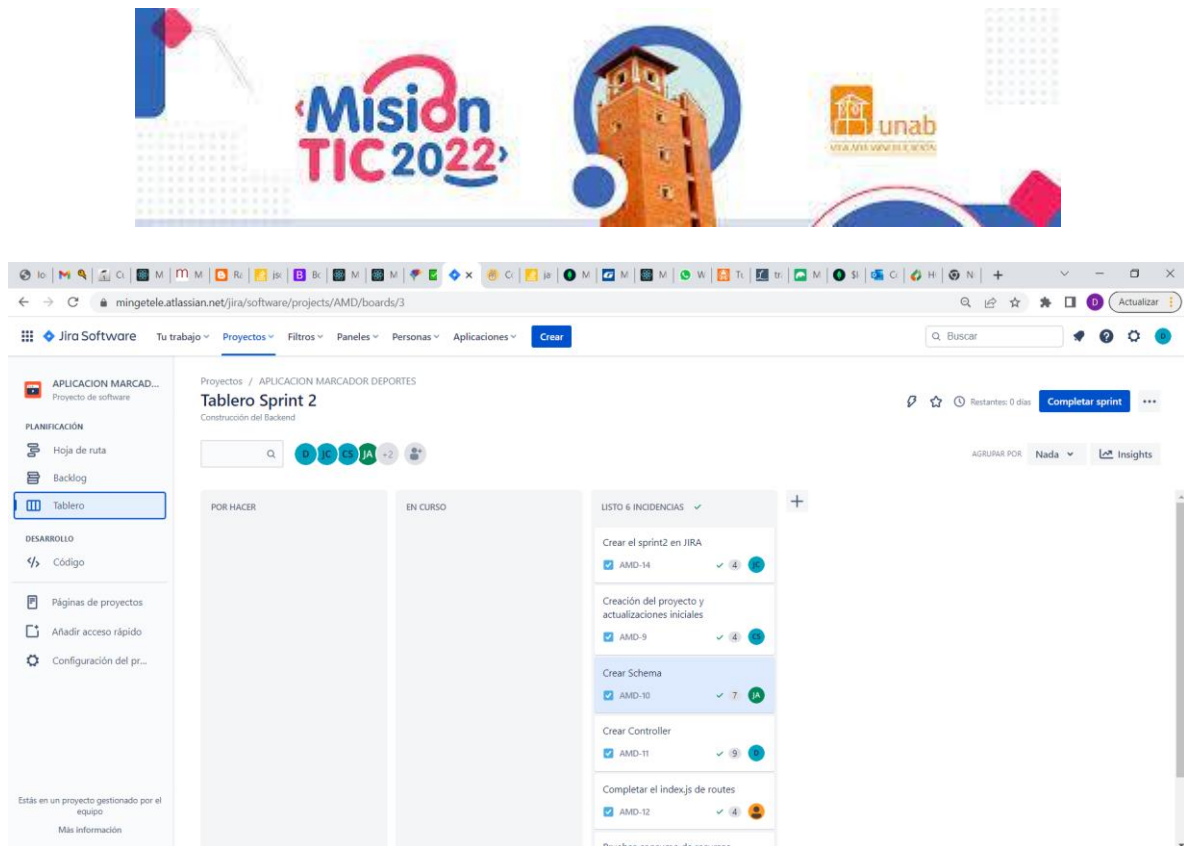
Backlog

Tablero Sprint 2 10 dic - 13 dic (6 incidencias) 0 0 24 Completar sprint

Construcción del Backend

<input checked="" type="checkbox"/>	AMD-14 Crear el sprint2 en JIRA	4 FINALIZADA
<input checked="" type="checkbox"/>	AMD-14 Crear Controller	9 FINALIZADA
<input checked="" type="checkbox"/>	AMD-9 Creación del proyecto y actualizaciones iniciales	4 FINALIZADA
<input checked="" type="checkbox"/>	AMD-10 Crear Schema	7 FINALIZADA
<input checked="" type="checkbox"/>	AMD-12 Completar el index.js de routes	4 FINALIZADA
<input checked="" type="checkbox"/>	AMD-13 Pruebas consumo de recursos del API REST	6 FINALIZADA

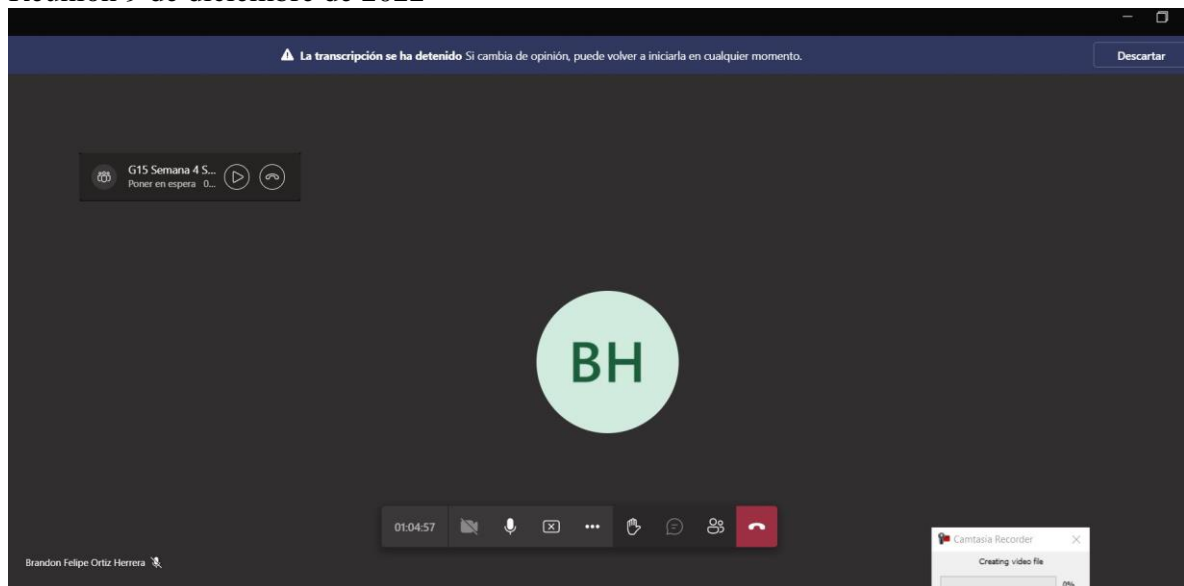
+ Crear incidencia



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.

Reunión 9 de diciembre de 2022





FECHA: 09 DE DICIEMBRE DE 2022

HORA: 10:22 P.M. A 11:26 P.M.

ASUNTO: Revisar el formato del Sprint 1 diligenciado para entrega

PARTICIPANTES:

- Juan Carlos Álvarez
- Doris Aroca
- Felipe Álvarez
- Brandon Ortiz Herrera

TEMAS TRATADOS:

- Se acuerda desarrollar el Backend en Mongo
 - Nombre del proyecto: PROYECTO G15-G2
 - Se definieron las tareas a ingresar para el sprint 2 en la aplicación JIRA:
 - 1- Creación del proyecto – subir servidor y dependencia
 - 2- Crear los esquemas: Juan Carlos Álvarez puntaje 7
 - usuarios.js
 - eventos.js
 - deportes.js
 - equipos.js – Crear controlares – Doris Aroca Tique puntaje 9
 - usuariosControllers.js
 - eventosControllers.js
 - deportesControllers.js
 - equiposControllers.js
 - 3- Completar el index.js de routes – Andres Felipe Álvarez puntaje 4.5
 - 4- Pruebas consumo de recursos del API REST - Brandon Felipe Ortiz - 6
 - 5- Crear el sprint en JIRA – Jorge Moreno puntaje 4

Lanzamiento del Sprint2 sábado 10 de 2022 y termina lunes 12 de diciembre de 2022
 - Evidencias Github: todos realizamos el proyecto completo y actualizados Github
-