

# Primer Entregable

## PROYECTO COVIDBUS

Javier Roldán Marín

Iván Romero Pastor

- **Introducción:**

Hemos decidido realizar el proyecto de una compañía de autobuses con el nombre de CovidBus, para recalcar la comprometida situación en la que nos encontramos actualmente y el aprovechamiento de esto para poder introducir nuevas mejoras en estos, para aportar mayor seguridad y confianza a los consumidores, efectuando así un mayor y mejor uso de los transportes públicos para las personas.

Por esto hemos desarrollado un dispositivo que vaya conectado al autobús que nos indique al usuario la proximidad del autobús a nosotros, con esto estimaremos el tiempo de espera en la parada. Podríamos consultar la temperatura dentro del autobús, así como de la humedad que hay en él. Todo ello con el fin de proporcionar al usuario información acerca del autobús, y así poder decidir si es más seguro coger un autobús u otro en estos tiempos que corren.

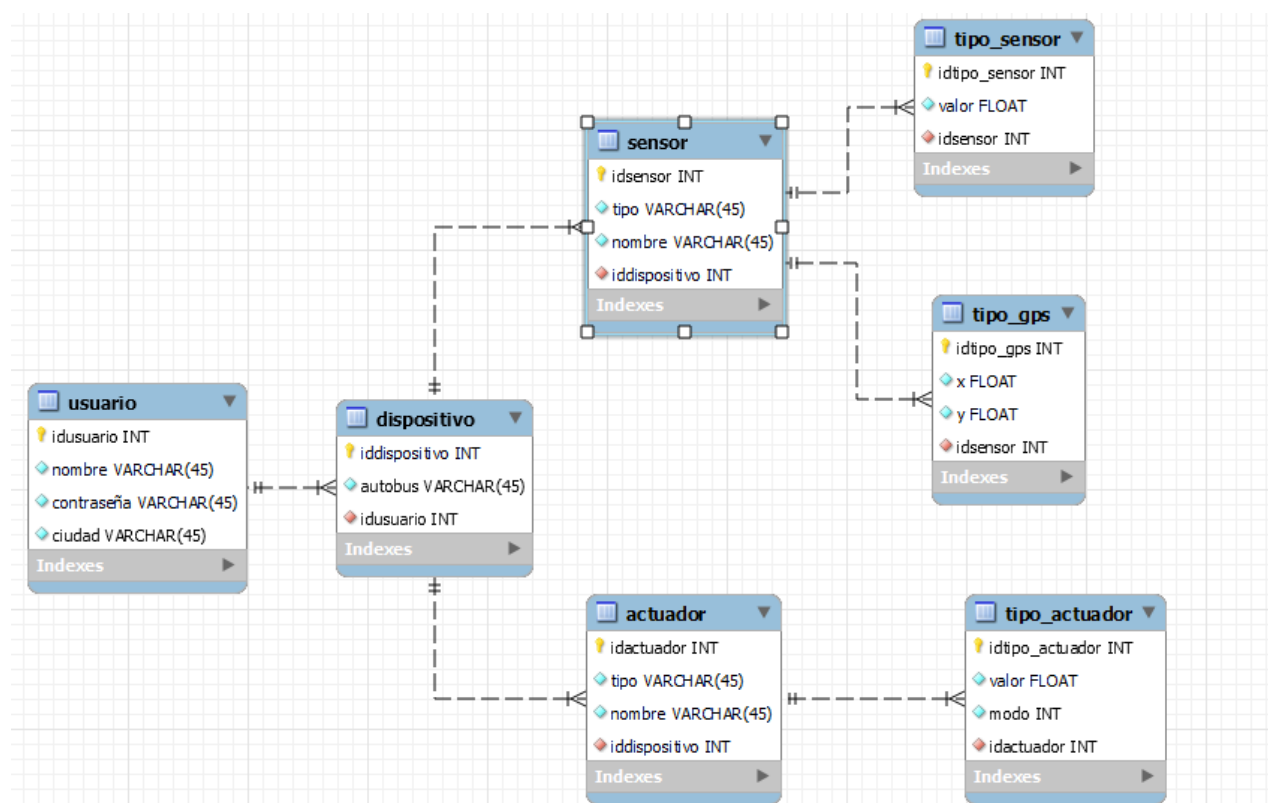
- **Base de datos:**

Nuestra base de datos, se trata de una base de datos simplificada con el fin de que no sea muy laborioso trabajar con ella. Por lo que, hemos creado las siguientes tablas:

- Usuario: Donde almacenaremos la información correspondiente de cada usuario de la aplicación.

- Dispositivo: Este irá colocado en cada autobús.
- Sensor: Donde se fijará el tipo de sensor que es y a que dispositivo está conectado.
- Tipo\_Sensor y Tipo GPS: Almacenan la información extraída de los sensores(En este caso sensore tipo gps, humedad, temperatura..).
- Actuador: Donde se fijará el tipo de actuador que es y a que dispositivo está conectado.
- Tipo\_Actuador: Almacenan la información extraída de los actuadores(Como actuadores tipo led, sonido).

Como resultado de todas estas tablas, obtenemos el siguiente diagrama UML de la base de datos:



## • API Rest:

A continuación explicamos los diferentes métodos o servicios REST que tenemos añadidos en el proyecto por ahora, para compartir recursos e información entre los usuarios y el servidor:

- **GET** : Usados para obtener información de la base de datos, los métodos get no usan cuerpo, usan de la URL para solicitar la información deseada.

Hemos introducido este método para todas las tablas.

- `this::getUsuario:` mediante el identificador único de usuario, obtenemos la información de un usuario específico.

URL introducida: `"/api/usuario/:idusuario"`

The screenshot shows a REST client interface with a GET request to `localhost:8080/api/usuario/1`. The 'Params' tab is selected, showing a table for Query Params.

KEY	VALUE	DESCRIPTION
Key	Value	Description

The screenshot shows the 'Body' tab of the REST client, displaying the JSON response of the GET request. The status is 200 OK, time is 118 ms, and size is 177 B.

```
1 {
2   "idusuario": 1,
3   "nombre": "ivan_put",
4   "contraseña": "ivan",
5   "ciudad": "sevilla"
6 }
7
8
```

- **this::getDispositivo:** mediante el id de un dispositivo, obtenemos toda su información asociada.

URL introducida: ["/api/dispositivo/:iddispositivo"](#)

GET localhost:8080/api/dispositivo/2 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 21 ms Size: 145 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "iddispositivo": 2,
3   "autobus": "bus2",
4   "idusuario": 2
5 }
```

- **this::getDispositivosUsuarios:** este método nos devuelve todos los dispositivos registrados.

URL introducida: ["/api/dispositivosUsuarios/"](#)

GET localhost:8080/api/dispositivosUsuarios/ Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 11 ms Size: 299 B Save Response

Pretty Raw Preview Visualize JSON

```
2 {
3   "iddispositivo": 1,
4   "autobus": "bus1_put",
5   "idusuario": 1
6 },
7 {
8   "iddispositivo": 2,
9   "autobus": "bus2",
10  "idusuario": 2
11 },
12 {
13   "iddispositivo": 3,
14   "autobus": "bus2_put2",
15   "idusuario": 4
16 }
```

- **this::getDispositivoUsuario:** método que dado un id de usuario, te devuelve en que bús(dispositivo) está montado.

URL introducida: ["/api/dispositivosUsuarios/:idusuario"](/api/dispositivosUsuarios/:idusuario)

GET localhost:8080/api/dispositivosUsuarios/1

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 9 ms Size: 149 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "iddispositivo": 1,
3   "autobus": "bus1_put",
4   "idusuario": 1
5 }
```

- **this::getSensor:** este método te devuelve el tipo de sensor estamos dándole como parámetro.

URL introducida: ["/api/sensor/:idsensor"](/api/sensor/:idsensor)

GET localhost:8080/api/sensor/1

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

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

This request does not have a body

Body Cookies Headers (2) Test Results Status: 200 OK Time: 13 ms Size: 164 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "tipo": "temp",
4   "nombre": "temperatura",
5   "iddispositivo": 1
6 }
```

- **this::getTipoSensor:** dado un id de sensor (por ejemplo id\_humedad, id\_calor, id\_CO2...) devuelve su información actual asociada.

URL introducida: `/api/tipoSensor/:idtipo_sensor`

GET localhost:8080/api/tipoSensor/1 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 21 ms Size: 140 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "idtipo_sensor": 1,
3   "valor": 28.0,
4   "idsensor": 1
5 }
6
7
```

- **this::getSensorGPS** este método nos devolverá las coordenadas GPS actuales de nuestros dispositivos

URL introducida: `/api/tipoSensorGPS/:idtipo_gps`

GET localhost:8080/api/tipoSensorGPS/1 Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 14 ms Size: 146 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "idtipo_gps": 1,
3   "x": 5.0,
4   "y": 7.0,
5   "idsensor": 3
6 }
7
8
```

- `this::getActuador` información respecto a los diferentes actuadores que puedan haber en el proyecto.

URL introducida: `/api/actuador/:idactuador`

GET localhost:8080/api/actuador/1

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 7 ms Size: 159 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "tipo": "led",
4   "nombre": "ledbus1",
5   "iddispositivo": 1
6 }
```

- `this::getTipoActuador` información asociada a cada dispositivo actuador.

URL introducida: `/api/tipoActuador/:idtipo_actuador`

GET localhost:8080/api/tipoActuador/2

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (2) Test Results Status: 200 OK Time: 10 ms Size: 159 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "idtipo_actuador": 2,
3   "valor": 30.2,
4   "modo": 0,
5   "idactuador": 2
6 }
```

- **POST** : Usados para incluir nuevas entidades a la base de datos.

Tenemos varios métodos POST, uno para incluir a los usuarios, otro para incluir a los dispositivos (un dispositivo por autobús) , y los tres últimos para incluir tanto un tipo actuador como de los diferentes tipos de sensores.

- `this::postUsuario`  
URL introducida: `/api/PostUsuario/`

POST localhost:8080/api/PostUsuario/ Send

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

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

```
1 {
2   "idusuario": "7",
3   "nombre": "german_post",
4   "contraseña": "germina",
5   "ciudad": "berlin"
6 }
```

Body Cookies Headers (2) Test Results Status: 200 OK Time: 143 ms Size: 89 B Save Response

Pretty Raw Preview Visualize JSON

```
1 Usuario registrado
```

- `this::postDispositivo`  
URL introducida: `/api/PostDispositivo/`

POST localhost:8080/api/PostDispositivo/ Send

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

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

```
1 {
2   "iddispositivo": "6",
3   "autobus": "bus5_post",
4   "idusuario": "2"
5 }
```

Body Cookies Headers (2) Test Results Status: 200 OK Time: 20 ms Size: 146 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "iddispositivo": 6,
3   "autobus": "bus5_post",
4   "idusuario": 2
5 }
```



- `this::postTipo_GPS`  
URL introducida: `/api/PostGPS/`

POST localhost:8080/api/PostGPS/ Send

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

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

```
1 {
2   ... "idtipo_gps": "3",
3   ... "x": "681.0",
4   ... "y": "-359.6",
5   ... "idsensor": "6"
6 }
```

Body Cookies Headers (2) Test Results Status: 200 OK Time: 42 ms Size: 147 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "idtipo_gps": 3,
3   "x": 681.0,
4   "y": -359.6,
5   "idsensor": 6
6 }
```

- `this::postTipo_Sensor`  
URL introducida: `/api/PostSensor/`

POST localhost:8080/api/PostSensor/ Send

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

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

```
1 {
2   ... "idtipo_sensor": "3",
3   ... "valor": "61.0",
4   ... "idsensor": "4"
5 }
```

Body Cookies Headers (2) Test Results Status: 200 OK Time: 22 ms Size: 136 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "idtipo_sensor": 3,
3   "valor": 61.0,
4   "idsensor": 4
5 }
```

- `this::postTipo_Actuador`  
URL introducida: `/api/PostActuador/`

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

- Method:** POST
- URL:** localhost:8080/api/PostActuador/
- Body:** A JSON object: 

```
{  "idtipo_actuador": 3,  "valor": 47.8,  "modo": 1,  "idactuador": 3}
```
- Status:** 200 OK
- Time:** 24 ms
- Size:** 155 B
- Response:** A JSON object: 

```
{  "idtipo_actuador": 3,  "valor": 47.8,  "modo": 1,  "idactuador": 3}
```

- **PUT** : Método usado para actualizar un recurso en el servidor, de igual manera al POST, únicamente los usuarios y los dispositivos requerirán de esta opción.

- `this::PutUsuario`  
URL introducida: `/api/PutUsuario/:idusuario`

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

- Method:** PUT
- URL:** localhost:8080/api/PutUsuario/1
- Body:** A JSON object: 

```
{  "idusuario": 1,  "nombre": "ivan_put",  "contraseña": "ivan",  "ciudad": "sevilla"}
```
- Status:** 200 OK
- Time:** 149 ms
- Size:** 90 B
- Response:** Usuario actualizado

- **this::PutDispositivo**

URL introducida: ["/api/PutDispositivo/:iddispositivo"](/api/PutDispositivo/:iddispositivo)

PUT localhost:8080/api/PutDispositivo/1

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

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

```
1 {
2   "iddispositivo": "1",
3   "autobus": "bus1_put",
4   "idusuario": "1"
5 }
```

Body Cookies Headers (2) Test Results Status: 200 OK Time: 13 ms Size: 145 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "iddispositivo": 1,
3   "autobus": "bus1_put",
4   "idusuario": 1
5 }
```

- **DELETE** : Método DELETE para eliminar a una entidad o recurso, de igual forma, hemos intrucido este método para los usuarios y los dispositivos.

- **this::DeleteUsuario**

URL introducida: ["/api/EliminarUsuario/:idusuario"](/api/EliminarUsuario/:idusuario)

DELETE localhost:8080/api/EliminarUsuario/6

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (2) Test Results Status: 200 OK Time: 10 ms Size: 100 B Save Response

Pretty Raw Preview Visualize JSON

```
1 Usuario borrado correctamente
```

- **this::DeleteDispositivo**  
URL introducida: `"/api/PutDispositivo/:iddispositivo"`

DELETE

localhost:8080/api/EliminarDispositivo/6

Send

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (2)

Test Results

Status: 200 OK

Time: 15 ms

Size: 104 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

1 Dispositivo borrado correctamente