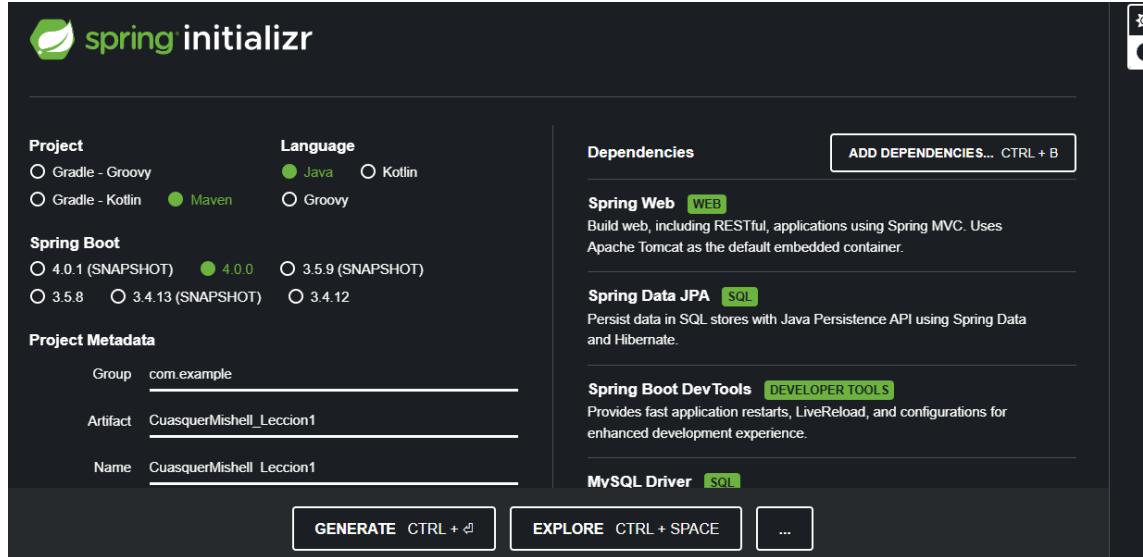


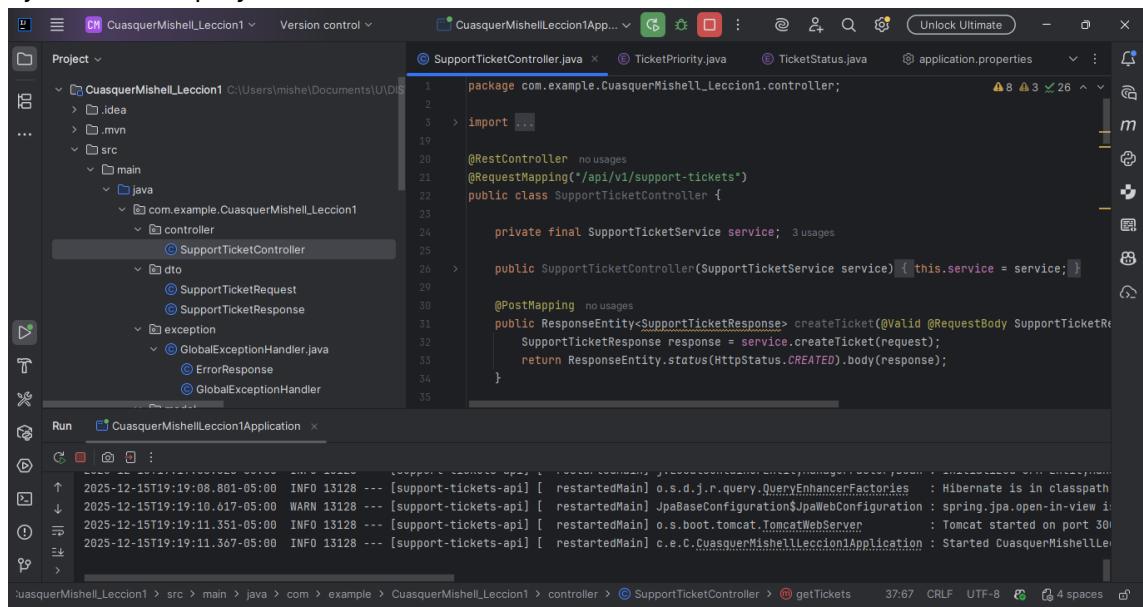
Nombre: Mishell Cuasquer

### Taller JPA- Diseño e Implementación de una API RESTful con Filtros

#### 1. Creamos nuestra plantilla



#### 2. Ejecutamos el proyecto



#### 3. Verificamos la conexión a la bd

Nombre: Mishell Cuasquer

### Taller JPA- Diseño e Implementación de una API RESTful con Filtros

The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the SCHEMAS section with various databases like 'ecuador', 'espe\_extraccion', 'estudiante', 'ferreteria\_inventario', 'medicamentos\_system', 'pharmacy\_system', 'prueba\_extraccion', 'sisdb2026', 'sisdb2027', and the 'support\_tickets\_db' database, which contains a single table named 'support\_tickets'. The main area shows a query editor with the SQL command: 'SELECT \* FROM support\_tickets\_db.support\_tickets;'. Below it is a Result Grid table with one row of data:

id	category	created_at	currency	due_date	estimated_cost	priority	requester_name	status	ticketNumber
1	Soporte Técnico	2025-12-15 20:16:45.818174	USD	2025-12-20	150.50	HIGH	Juan Pérez	OPEN	ST-2025-637057

Verificamos y realizamos las pruebas en el postman

#### 1) Crear ticket

POST /api/v1/support-tickets

Registra un nuevo ticket.

createdAt debe ser generado por el sistema.

The screenshot shows the Postman application interface. The request URL is 'http://localhost:3001/api/v1/support-tickets'. The request method is 'POST'. The 'Body' tab is selected, showing a raw JSON payload:

```

1 {
2   "requesterName": "Carlos Andrade",
3   "status": "CLOSED",
4   "priority": "LOW",
5   "category": "Consulta General",
6   "estimatedCost": 0.00,
7   "currency": "USD"
8 }

```

The response status is '201 Created' with a response time of '780 ms' and a response size of '401 B'. The response body is identical to the request body.

#### 2) Listar tickets con filtros

GET /api/v1/support-tickets

Debe soportar filtros opcionales con query params.

Debe ser paginado y ordenable (page, size, sort).

Nombre: Mishell Cuasquer

### Taller JPA- Diseño e Implementación de una API RESTful con Filtros

The screenshot shows a Postman interface with the following details:

- Method: GET
- URL: <http://localhost:3001/api/v1/support-tickets>
- Body tab is selected.
- Response status: 200 OK
- Response time: 417 ms
- Response size: 1.32 KB
- Response content (JSON):

```
2 |   "content": [
3 |     {
4 |       "id": 3,
5 |       "ticketNumber": "ST-2025-907553",
6 |       "requesterName": "Juan Pérez",
7 |       "status": "OPEN",
8 |       "priority": "HIGH",
9 |       "category": "Soporte Técnico",
10 |      "estimatedCost": 150.50,
11 |      "currency": "USD",
12 |      "createdAt": "2025-12-15T19:28:27.553573",
13 |      "dueDate": "2025-12-20"
14 |    },
15 |    {
16 |      "id": 2,
17 |      "ticketNumber": "ST-2025-637057",
18 |      "requesterName": "Carlos Andrade",
19 |      "status": "CLOSED"
}
```

#### Filtros obligatorios a implementar

##### 1) Búsqueda de texto (q)

Campos afectados: ticketNumber, requesterName

Búsqueda case-insensitive

Si q está vacío o no se envía → no filtra

Ejemplo:

GET /api/v1/support-tickets?q=chicaiza

Si envoi un dato sin la info del nombre no me permite registrar.

**Nombre:** Mishell Cuasquer

### Taller JPA- Diseño e Implementación de una API RESTful con Filtros

The screenshot shows a POST request to `http://localhost:3001/api/v1/support-tickets`. The request body is invalid JSON:

```

1
2 {
3     "requesterName": "",
4     "status": "OPEN",
5     "priority": "HIGH",
6     "category": "Soporte Técnico",
7     "estimatedCost": 150.50,
8     "currency": "USD",
9     "dueDate": "2025-12-20"
10 }

```

The response is a **400 Bad Request** with the following JSON error message:

```

1 {
2     "status": 400,
3     "message": "Error de validación",
4     "errors": {
5         "requesterName": "El nombre debe tener entre 2 y 100 caracteres"
6     },
7     "timestamp": "2025-12-15T19:37:36.5760269"
8 }

```

#### 2) Estado (status)

Valores permitidos: OPEN, IN\_PROGRESS, RESOLVED, CLOSED, CANCELLED

Si no se envía → no filtra

Valor inválido → 400 Bad Request

Ejemplo:

GET /api/v1/support-tickets?status=OPEN

The screenshot shows a GET request to `http://localhost:3001/api/v1/support-tickets?status=OPEN`. The query parameters are:

Key	Value	Description
status	OPEN	

The response is a **200 OK** with the following JSON data:

```

1 {
2     "content": [
3         {
4             "id": 3,
5             "ticketNumber": "ST-2025-907553",
6             "requesterName": "Juan Pérez",
7             "status": "OPEN",
8             "priority": "HIGH",
9             "category": "Soporte Técnico",
10            "estimatedCost": 150.50,
11            "currency": "USD",
12            "createdAt": "2025-12-15T19:28:27.553573",
13            "dueDate": "2025-12-20"
14        },
15        {
16        }
17    ]
18 }

```

#### 3) Moneda (currency)

Valores permitidos: USD, EUR

Coincidencia exacta

Valor inválido → 400 Bad Request

**Nombre:** Mishell Cuasquer

### Taller JPA- Diseño e Implementación de una API RESTful con Filtros

Ejemplo:

GET /api/v1/support-tickets?currency=USD

The screenshot shows a Postman interface with a GET request to `http://localhost:3001/api/v1/support-tickets?currency=USD`. The response is successful (200 OK) with a response time of 28 ms and a size of 1.32 KB. The response body is a JSON array containing two ticket objects. The first ticket has an ID of 3, a requester name of "Juan Pérez", and a due date of "2025-12-20". The second ticket has an ID of 2.

```

2   "content": [
3     {
4       "id": 3,
5       "ticketNumber": "ST-2025-907553",
6       "requesterName": "Juan Pérez",
7       "status": "OPEN",
8       "priority": "HIGH",
9       "category": "Soporte Técnico",
10      "estimatedCost": 150.50,
11      "currency": "USD",
12      "createdAt": "2025-12-15T19:28:27.553573",
13      "dueDate": "2025-12-20"
14    },
15    {
16      "id": 2,
17      ...
18    }
19  ]

```

Si coloco otro valor en monedas me sale lo siguiente

The screenshot shows a Postman interface with a POST request to `http://localhost:3001/api/v1/support-tickets`. The request body is a JSON object with a requester name of "Juan Chicaiza", a status of "OPEN", a priority of "HIGH", a category of "Soporte Técnico", an estimated cost of 150.50, a currency of "DOL", and a due date of "2025-12-20". The response is a 500 Internal Server Error with a message: "Error interno del servidor".

```

1 {
2   "requesterName": "Juan Chicaiza",
3   "status": "OPEN",
4   "priority": "HIGH",
5   "category": "Soporte Técnico",
6   "estimatedCost": 150.50,
7   "currency": "DOL",
8   "dueDate": "2025-12-20"
9 }

```

```

1 {
2   "status": 500,
3   "message": "Error interno del servidor",
4   "errors": {
5     "error": "Ha ocurrido un error inesperado"
6   },
7   "timestamp": "2025-12-15T19:36:49.5731875"
8 }

```

#### 4) Monto mínimo (minCost)

Campo afectado: `estimatedCost`

Regla: `estimatedCost >= minCost`

Validación: `minCost >= 0`

Si no se envía → no filtra

Ejemplo:

Nombre: Mishell Cuasquer

**Taller JPA- Diseño e Implementación de una API RESTful con Filtros**  
GET /api/v1/support-tickets?minCost=50

The screenshot shows the Postman interface with the following details:

- Method:** GET
- URL:** http://localhost:3001/api/v1/support-tickets?minCost=50
- Params:** minCost (Value: 50)
- Response Status:** 200 OK
- Response Body (JSON):**

```
1  {
2   "content": [
3     {
4       "id": 3,
5       "ticketNumber": "ST-2025-907553",
6       "requesterName": "Juan Pérez",
7       "status": "OPEN",
8       "priority": "HIGH",
9       "category": "Soporte Técnico",
10      "estimatedCost": 150.50,
11      "currency": "USD",
12      "createdAt": "2025-12-15T19:28:27.553573",
13      "dueDate": "2025-12-20"
14    }
15  ]
```

**5) Monto máximo (maxCost)**

Campo afectado: estimatedCost

Regla: estimatedCost <= maxCost

Validación: maxCost >= 0

Si no se envía → no filtra

Ejemplo:

GET /api/v1/support-tickets?maxCost=300

**Nombre:** Mishell Cuasquer

### Taller JPA- Diseño e Implementación de una API RESTful con Filtros

The screenshot shows a Postman request for a GET endpoint. The URL is `http://localhost:3001/api/v1/support-tickets?maxCost=300`. The 'Params' tab is selected, showing a query parameter `maxCost` with the value `300`. The 'Body' tab shows the JSON response, which is a list of support tickets. The first ticket has an ID of 2, ticket number ST-2025-637057, requester name Carlos Andrade, status CLOSED, priority LOW, category Consulta General, estimated cost 0.00, currency USD, created at 2025-12-15T19:23:57.057664, and due date null. The second ticket has an ID of 1, ticket number ST-2025-805819, requester name Juan Pérez, and status null.

```

16   "id": 2,
17   "ticketNumber": "ST-2025-637057",
18   "requesterName": "Carlos Andrade",
19   "status": "CLOSED",
20   "priority": "LOW",
21   "category": "Consulta General",
22   "estimatedCost": 0.00,
23   "currency": "USD",
24   "createdAt": "2025-12-15T19:23:57.057664",
25   "dueDate": null
26 },
27 [
28   {
29     "id": 1,
30     "ticketNumber": "ST-2025-805819",
31     "requesterName": "Juan Pérez",
32   }
33 ]
  
```

#### 6) Regla combinada de fechas (from y to)

Campo afectado: `createdAt`

Formato ISO-8601: yyyy-MM-dd'T'HH:mm:ss

Regla: si ambos se envían → from <= to

Si from > to → 400 Bad Request

Ejemplo:

GET /api/v1/support-tickets?from=2025-01-01T00:00:00&to=2025-06-30T23:59:59

The screenshot shows a Postman request for a GET endpoint. The URL is `http://localhost:3001/api/v1/support-tickets?from=2025-01-01T00:00:00&to=2025-06-30T23:59:59`. The 'Params' tab is selected, showing two parameters: `from` with the value `2025-01-01T00:00:00` and `to` with the value `2025-06-30T23:59:59`. The 'Body' tab shows the JSON response, which is a paginated object. The `pageable` field contains a `pageSize` of 10, indicating 10 items per page.

```

1  {
2   "content": [],
3   "empty": true,
4   "first": true,
5   "last": true,
6   "number": 0,
7   "numberOfElements": 0,
8   "pageable": {
9     "offset": 0,
10    "pageNumber": 0,
11    "pageSize": 10,
12    "paged": true,
13    "sort": [
14      {
15        "ascending": false,
16      }
17    ]
18  }
19 }
  
```

# Universidad de las Fuerzas Armadas Espe

Nombre: Mishell Cuasquer

## Taller JPA- Diseño e Implementación de una API RESTful con Filtros

The screenshot shows a POSTMAN interface with a GET request to `http://localhost:3001/api/v1/support-tickets?from=2025-12-31T23:59:59&to=2025-01-01T00:00:00`. The response is a 400 Bad Request with the following JSON body:

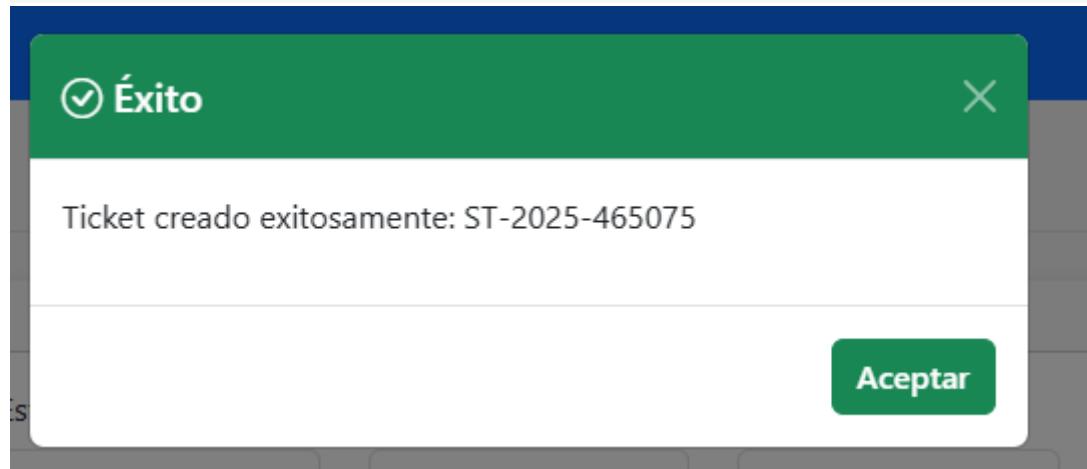
```
1 {
2     "status": 400,
3     "message": "Parámetro inválido",
4     "errors": [
5         "error": "La fecha 'from' debe ser anterior o igual a la fecha 'to'"
6     ],
7     "timestamp": "2025-12-15T19:42:11.042420Z"
8 }
```

Front

The screenshot shows a ticket creation form with the following fields and their values:

- Nombre del Solicitante \*: Veronica (green checkmark)
- Categoría \*: SOPORTE
- Estado \*: En Progreso (green checkmark)
- Prioridad \*: Baja (green checkmark)
- Moneda: USD
- Costo Estimado: \$ 120
- Fecha Límite: 25/12/2025

**Crear Ticket** button is present at the bottom.



Universidad de las Fuerzas Armadas Espe

**Nombre:** Mishell Cuasquer

Taller JPA- Diseño e Implementación de una API RESTful con Filtros

127.0.0.1:5500

## Sistema de Tickets

API En línea

+ Crear Ticket   Listar Tickets

### + Nuevo Ticket de Soporte

Nombre del Solicitante *	Categoría *	
<input type="text"/>	<input type="text"/>	
Estado *	Prioridad *	Moneda
<input type="button" value="Seleccionar..."/>	<input type="button" value="Seleccionar..."/>	<input type="button" value="Seleccionar..."/>
Costo Estimado	Fecha Límite	
<input type="text"/> \$	<input type="text"/> dd/mm/aaaa <input type="button" value=""/>	
<input type="button" value="Crear Ticket"/>		

Tickets de Soporte									
ID	Categoría	Created At	Moneda	Due Date	Costo Estimado	Prioridad	Requester Name	Estado	
5	Soporte Técnico	2025-12-16 00:51:50.489330	USD	2025-12-20	150.50	HIGH	Juan Pérez	ABIERTO	
6	Facturación	2025-12-16 00:52:38.343517	EUR	2025-12-25	200.00	MEDIUM	Maria Garcia	BAJA	
7	SOPORTE	2025-12-16 00:59:04.6552201	USD	2025-12-19	10.01	LOW	MISHELL	OPE	
8	SOPORTE	2025-12-16 01:27:45.074530	EUR	2025-12-25	120.00	HIGH	Estefania	RESOL	
9	SOPORTE	2025-12-16 01:28:23.367058	USD	2025-12-25	120.00	LOW	Veronica	IN_F	
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	cuasquermishel -			1.12%	756.39MB / 16.3						
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	mysql-suppor	76c4ba0a8449	mysql:8.0	3308:3306		0.91%	405.1MB / 5.45G				
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	cuasquer-mis	cc75edcb371d	cuasquer-	8082:3001		0.21%	340.8MB / 5.45G				
<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	frontend	511d2b8ab5f0	cuasquer-	5500:80		0%	10.49MB / 5.45G				

**Nombre:** Mishell Cuasquer

**Taller JPA- Diseño e Implementación de una API RESTful con Filtros**

Repositorios					
Todos los repositorios dentro del espacio de nombres mishellcuasquer .					
Nombre	Último empu...	Contiene	Visibilidad	Explorar	
mishellcuasquer / cuasquer-mishell-backend	Hace 1 minuto	IMAGEN	Público	Inactivo	
mishellcuasquer / cuasquer-mishell-frontend	Hace 2 minutos	IMAGEN	Público	Inactivo	
mishellcuasquer / API de tickets de soporte	hace unas 5 horas	IMAGEN	Público	Inactivo	
mishellcuasquer / imágenes	Hace 24 días	IMAGEN	Público	Inactivo	

**Link de Github.**

<https://github.com/MishellCuasquer/ExamenP2Cuasquer.git>

**Link´s de Dockerhub.**

General: <https://hub.docker.com/repositories/mishellcuasquer>

Backend: <https://hub.docker.com/repository/docker/mishellcuasquer/cuasquer-mishell-backend/general>

Front: <https://hub.docker.com/repository/docker/mishellcuasquer/cuasquer-mishell-frontend/general>