Universidad ORT Uruguay

Facultad de Ingeniería

Bernard Wand Polak

Diseño de Aplicaciones 2

Obligatorio 1

2-Especificación de la API

Santiago Castro – 168347

https://github.com/ORT-DA2/168347

Grupo M6B

Tabla de contenido

1. Criterios REST

REST es un estilo arquitectónico que describe buenas prácticas de implementación de APIs. Algunos de los criterios que se tomaron en cuenta son:

URL simple e intuitiva: Existen como máximo dos URLs base por recurso, por ejemplo, /requests y /requests/{id}

URL base no contiene verbos: Las URL base son sustantivos como /Requests, /Areas, /TopicTypes, /Users, etc. No se usan verbos.

Uso de verbos HTTP: Se utilizaron los verbos POST, GET, PUT, DELETE para operar con los datos, de esta forma se abarcan todas las posibles funcionalidades del sistema.

Resource	POST create	GET read	PUT update	DELETE delete
/dogs	Create a new dog	List dogs	Bulk update dogs	Delete all dogs
/dogs/1234	Error	Show Bo	If exists update Bo If not error	Delete Bo

Manejo de errores: Si ocurre algún error al momento de procesar una petición se muestra al usuario información útil para poder saber qué sucedió.

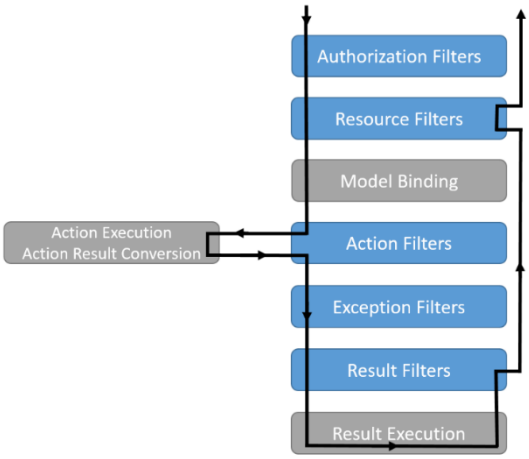
Response body

```
{
  "errorCode": "ERR_AUTH_MISSING",
  "status": 400,
  "details": "Auth header missing. Need to specify a session token in the Auth header"
}
```

2. Autenticación

Algunas funcionalidades requieren que el usuario que las usa debe ser un usuario administrador, por ejemplo, al momento de crear o borrar tipos, crear usuarios, modificar estado de solicitudes, etc. Para esto se utiliza un mecanismo de autenticación a través de filtros. Los filtros permiten ejecutar código antes o después de procesar solicitudes HTTP, de esta forma si una solicitud requiere estar autenticado se puede verificar antes de procesar la solicitud.

Se creó una clase llamada *AuthFilter* la cual implementa la interfaz *IAuthorizationFilter* con el método OnAuthorization. Este método toma el valor del header "Auth" del contexto y llama a la lógica para buscar el token en la base de datos con el método IsLogged. Si se encuentra el token, se procede y si no, se retorna un error con status 401.



Como se muestra en la imagen el filtro de Authorization es el primero que se ejecuta al momento de realizar una petición HTTP, lo cual no nos permite dar uso del filtro de excepciones en caso de que se quiera retornar un error.

Si el usuario quiere iniciar sesión este envía una petición POST /Sessions con los datos de email y password en formato JSON. La API valida los datos con el método Login de la clase SessionLogic que retorna un nuevo token si los datos son válidos o retorna una excepción la cual va a ser procesada por el filtro de excepciones. Este token es almacenado en la base de datos junto al ID del usuario y la fecha de creación del token. Finalmente se retorna el token en el cuerpo de la respuesta HTTP.

Si el usuario desea cerrar la sesión este envía una petición DELETE /Sessions con el token en el header con nombre Auth. Pimero se ejecuta el filtro de autenticación para verificar si el token es válido y existe en el sistema, luego se procede a ejecutar el endpoint el cual llama al método Logout de la clase SessionLogic el cual elimina el token de la base de datos. Finalmente se retorna el status 200.

3. Códigos de error

La API retorna diferentes códigos Status luego de procesar las solicitudes los cuales informan al usuario el resultado de la misma. Los códigos utilizados fueron:

200 - Success: Cuando los siguientes endpoints se procesan correctamente:

- GET /Areas
- GET /Areas/{id}
- GET /Requests/{id}
- GET /Requests
- PUT /Requests/{id}
- DELETE /Sessions/{id}
- GET /Topics/{id}
- GET /TopicType/{id}
- DELETE /TopicTypes/{id}
- GET /Users/{id}
- GET /Users
- PUT /Users/{id}
- DELETE /Users/{id}

201 - Created: Cuando los siguientes endpoints se procesan correctamente:

- POST /Requests
- POST /Sessions
- POST /TopicTypes
- POST /Users

400 – Bad Request: Cuando la información dada por el usuario en la solicitud es incorrecta, errónea o faltante. Por ejemplo, si al momento de ingresar un nuevo usuario el email no cumple con el formato xxx@xxx.xxx la respuesta va a ser 400 y un error en el body.

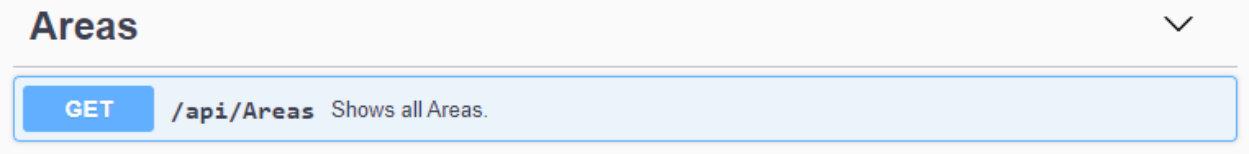
401 – Unauthorized: Cuando se requiere estar autenticado para utilizar el endpoint. Normalmente por no haber pasado el token por el header.

404 – Not Found: Cuando se hacen solicitudes del tipo GET y no se encontró ningún resultado. Por ejemplo GET /Requests/{id} y no se encuentra ninguna Request con esa id.

500 – Internal Server Error: Cuando ocurre algún error al procesar la solicitud, por ejemplo, algún error en la base de datos, tanto al momento de conectarse o utilizarla.

4. Recursos de la API

Utilizando la herramienta Swagger podemos ver información útil sobre la API. Para acceder a ella debemos acceder a la dirección /swagger una vez que la WEB API se encuentra en funcionamiento.



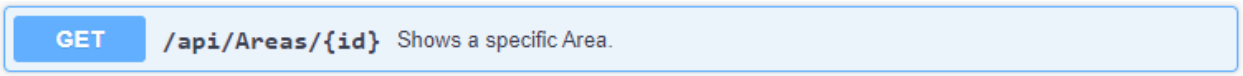
Descripción: Se utiliza para que el usuario al momento de crear una nueva solicitud pueda navegar entre las categorías. Este endpoint devuelve todas las áreas del sistema.

Parámetros: Ninguno.

Respuestas:

- 200 – Success: Devuelve una lista con las áreas en el sistema.

Headers: Ninguno.



Descripción: Al momento que el usuario selecciona un área, se envía una solicitud a esta dirección para obtener los temas asociados al área seleccionada.

Parámetros: {id} ID del área a buscar.

Respuestas:

- 200 – Success: Devuelve la información del área.
- 404 – Not Found: Si no se encuentra el área con la ID especificada.

Headers: Ninguno.

Requests

POST `/api/Requests` Creates a new Request.

Descripción: La utiliza el usuario para crear una nueva solicitud.

Parámetros: Ninguno.

Respuestas:

- 201 – Success: Solicitud creada, devuelve la solicitud junto con su ID.
- 400 - Bad Request: Cuando la información dada por el usuario en la solicitud es incorrecta, errónea o faltante.
- 404 – Not found: cuando el TopicType no se encuentra en el sistema

Headers: Ninguno.

```
{
  "details": "string",
  "applicant": {
    "name": "string",
    "email": "string",
    "phoneNumber": "string"
  },
  "topicType": "guid",
  "additionalFields": [
    {
      "additionalField": "guid",
      "data": "string"
    }
  ]
}
```

Body: Datos de la solicitud a ingresar

Donde:

- Details campo obligatorio no mayor a 2000 caracteres.
- Applicant.name obligatorio.
- Applicant.email obligatorio con formato email.
- topicType obligatorio y existente en el sistema
- additionalFields cumplen el formato especificado para el tipo seleccionado

GET `/api/Requests` Shows all Requests.

Descripción: Muestra todas las solicitudes del sistema, debe estar autenticado como administrador.

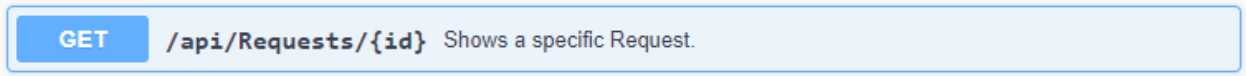
Parámetros: Ninguno.

Respuestas:

- 200 – Success: Devuelve todas las solicitudes
- 401 – Unauthorized: Si el token es incorrecto o inválido.

Headers:

- Auth: Guid = token de autenticación.



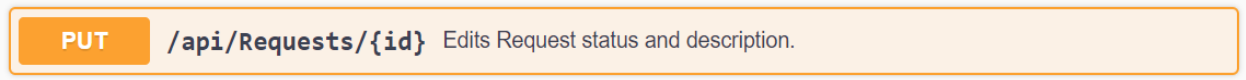
Descripción: Este endpoint es utilizado para obtener el estado y su descripción de una solicitud.

Parámetros: {id} ID de la solicitud a buscar.

Respuestas:

- 200 – Success: Devuelve la información de la solicitud.
- 404 – Not Found: Si no se encuentra la solicitud con la ID especificada.

Headers: Ninguno.



Descripción: Este endpoint es utilizado por los administradores del sistema para modificar el estado de alguna solicitud.

Parámetros: {id} ID de la solicitud a modificar.

Respuestas:

- 200 – Success: Solicitud modificada correctamente.
- 400 - Bad Request: Cuando la información dada por el usuario en la solicitud es incorrecta, errónea o faltante. Posiblemente el nuevo estado es inválido.
- 401 – Unauthorized: Si el token es incorrecto o inválido.
- 404 – Not Found: Si no se encuentra la solicitud con la ID especificada.

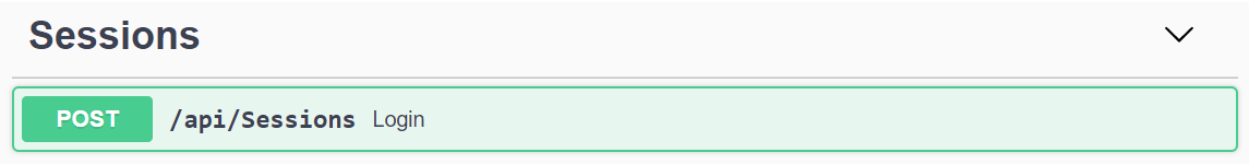
Headers:

- Auth: Guid = token de autenticación.

Body: Nuevo estado y descripción

```
{
  "status": "string",
  "statusDescription": "string"
}
```

Status: debe cumplir la precedencia: **Created <-> In review <-> Accepted <-> Denied <-> Closed**



Descripción: Se utiliza para crear una sesión en el sistema.

Parámetros: Ninguno.

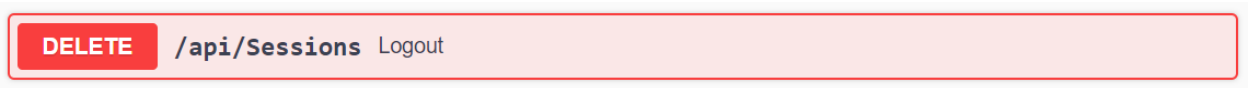
Respuestas:

- 201 – Success: Sesión creada correctamente.
- 400 - Bad Request: Cuando la información del body es incorrecta, errónea o faltante.

Headers: Ninguno.

Body: Nuevo estado y descripción

```
{
  "email": "admin@mail.com",
  "password": "password"
}
```



Descripción: Se utiliza para borrar la sesión con un token proporcionado.

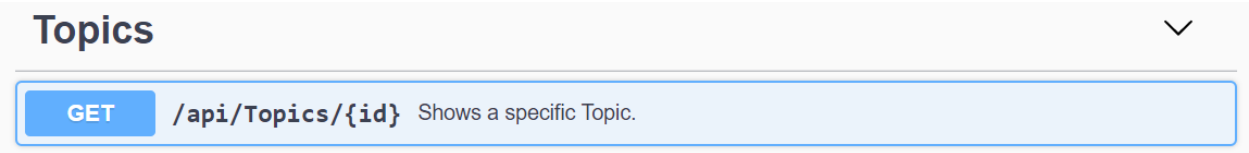
Parámetros: Ninguno.

Respuestas:

- 200 – Success: Sesión eliminada correctamente.
- 401 – Unauthorized: Si el token es incorrecto o inválido.

Headers:

- Auth: Guid = token de autenticación.



Descripción: Al momento que el usuario selecciona un tema, se envía una solicitud a esta dirección para obtener los tipos asociados al tema seleccionado.

Parámetros: {id} ID del tema a buscar.

Respuestas:

- 200 – Success: Devuelve la información del tema.
- 404 – Not Found: Si no se encuentra el tema con la ID especificada.

Headers: Ninguno.

TopicTypes

POST `/api/TopicTypes` Creates a new TopicType.

Descripción: La utiliza el usuario administrador para crear un nuevo tipo.

Parámetros: Ninguno.

Respuestas:

- 201 – Success: Tipo creado, devuelve la información del nuevo tipo.
- 400 - Bad Request: Cuando la información dada por el usuario en la solicitud es incorrecta, errónea o faltante.
- 404 – Not found: cuando el Tema no se encuentra en el sistema.

Headers:

- Auth: Guid = token de autenticación.

Body: Datos del tipo a ingresar

- topic obligatorio y existente en el sistema
- name campo obligatorio único para el tema.
- additionalField.name obligatorio único para el tipo.
- additionalField.fieldtype obligatorio con valor Text, Integer o Date.
- additionalField.possibleValues lista de valores posibles, en caso de ser del tipo Date o Integer, debe ser un rango entre dos valores.

```
{
  "topic": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "name": "string",
  "additionalFields": [
    {
      "name": "string",
      "fieldType": "string",
      "possibleValues": [
        "string"
      ]
    }
  ]
}
```

GET `/api/TopicTypes/{id}` Shows a specific TopicType.

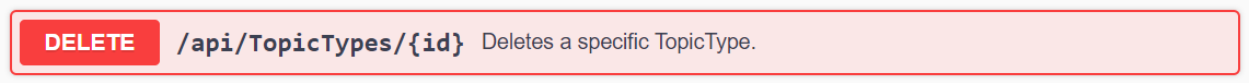
Descripción: Al momento que el usuario selecciona un tipo, se envía una solicitud a esta dirección para obtener los datos asociados al tipo seleccionado, incluyendo sus campos adicionales.

Parámetros: {id} ID del tipo a buscar.

Respuestas:

- 200 – Success: Devuelve la información del tipo.
- 404 – Not Found: Si no se encuentra el tipo con la ID especificada.

Headers: Ninguno.



Descripción: Se utiliza para borrar un tipo con una id específica.

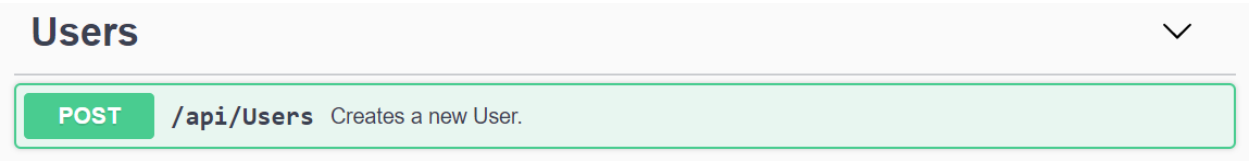
Parámetros: {id} ID del tipo a eliminar.

Respuestas:

- 200 – Success: Tipo eliminado correctamente.
- 404 – Not found: cuando el Tipo no se encuentra en el sistema.
- 401 – Unauthorized: Si el token es incorrecto o inválido.

Headers:

- Auth: Guid = token de autenticación.



Descripción: Se utiliza para que un usuario administrador pueda crear otros administradores.

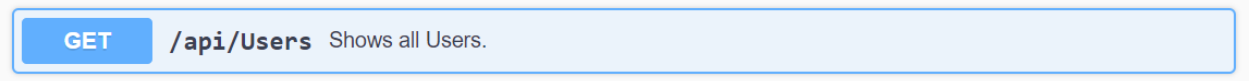
Parámetros: Ninguno.

Respuestas:

- 201 – Success: Usuario creado, devuelve la información del nuevo usuario.
- 400 - Bad Request: Cuando la información del nuevo usuario es incorrecta, errónea o faltante.
- 401 – Unauthorized: Si el token es incorrecto o inválido.

Headers:

- Auth: Guid = token de autenticación.



Descripción: Muestra todos los usuarios del sistema, debe estar autenticado como administrador.

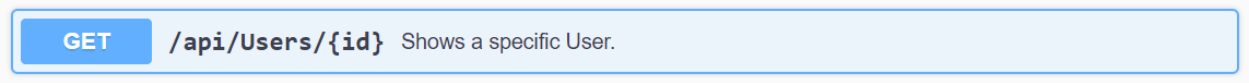
Parámetros: Ninguno.

Respuestas:

- 200 – Success: Devuelve todos los usuarios.
- 401 – Unauthorized: Si el token es incorrecto o inválido.

Headers:

- Auth: Guid = token de autenticación.



Descripción: Muestra la información de un usuario específico. Debe estar autenticado como administrador.

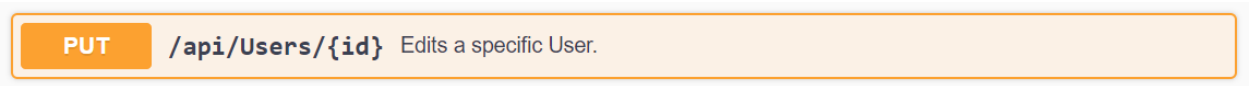
Parámetros: {id} ID del usuario a buscar.

Respuestas:

- 200 – Success: Devuelve la información del usuario.
- 401 – Unauthorized: Si el token es incorrecto o inválido.
- 404 – Not Found: Si no se encuentra el tema con la ID especificada.

Headers:

- Auth: Guid = token de autenticación.



Descripción: Este endpoint es utilizado por los administradores del sistema para modificar información de algún usuario.

Parámetros: {id} ID del usuario a modificar.

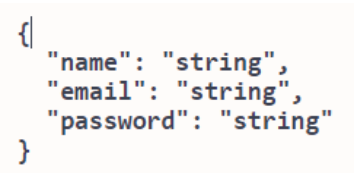
Respuestas:

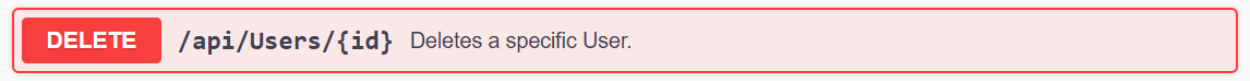
- 200 – Success: Uusario modificado correctamente.
- 400 - Bad Request: Cuando la información dada por el usuario es incorrecta, errónea o faltante.
- 401 – Unauthorized: Si el token es incorrecto o inválido.
- 404 – Not Found: Si no se encuentra el usuario con la ID especificada.

Headers:

- Auth: Guid = token de autenticación.

Body:





Descripción: Se utiliza para borrar un usuario con una id específica.

Parámetros: {id} ID del usuario a eliminar.

Respuestas:

- 200 – Success: Usuario eliminado correctamente.
- 404 – Not found: cuando el Usuario no se encuentra en el sistema.
- 401 – Unauthorized: Si el token es incorrecto o inválido.

Headers:

- Auth: Guid = token de autenticación.