Universidad ORT Uruguay

Diseño de Aplicaciones 2

Obligatorio 1

Link al repositorio:
https://github.com/IngSoft-DA2/278311_227286_266889.git

Docentes:
Nicolás Fierro
Alexander Wieler
Máximo Halty

Alumnos:
Nicolás Russo 227286
Matias Hirschfeld 266889
Victoria Chappuis 278311

DOCUMENTO: Evidencia de la ejecución de las pruebas de la API con Postman

LINK AL VIDEO:

https://youtu.be/xwQRLR6skBg?si=uU3ILk825eRzcToj

Casos de prueba postman

Asociar dispositivos al hogar:

Asociar un dispositivo a un hogar exitosamente:

Esta prueba fue realizada para verificar qué se pueda asociar un dispositivo a un hogar exitosamente.
Método HTTP: POST
URI: /api/homes/:homeId/devices
En el qué homeId representa el id de la home a la qué se está intentando agregar.
Body:

```
{
   "deviceId": 1
}
```

Headers:
Authorization: Bearer {token}

Respuesta esperada:
200(OK)

Autenticación fallida

En esta prueba se verifica qué ocurre cuando se envía la request pero sin el token de autenticación válido.
Método HTTP: POST
URI: /api/homes/:homeId/devices
Headers:
(Sin el token de autenticación)
Body:

```
{
   "deviceId": 1
}
```

Respuesta esperada:
401(Unauthorized access)

Asociar un device a una home no existente

Esta prueba fue realizada para verificar qué no se pueda agregar un device a una home no existente
Método HTTP: POST
URI: /api/homes/:homeId/devices
En el qué homeId representa el id de la home no existente
Body:

```
{
```

"deviceId": 1
}
Headers:
Authorization: Bearer {token}

Respuesta esperada:
400(BadRequest)
“Home not found”

Asociar un device a una home la cual no se pertenece el usuario no forma parte

Esta prueba fue realizada para verificar qué los usuarios qué pueden agregar devices a homes únicamente puedan realizarlo a homes de las qué sean miembros.
Método HTTP: POST
URI: /api/homes/:homeId/devices
Body:
{
"deviceId": 1
}
Headers:
Authorization: Bearer {token} (Dicho usuario no pertenece a la home seleccionada)

Respuesta esperada:
401(Unauthorized)
“You must be a member of the home to add a device”

Asociar un device no existente a una home

Esta prueba fue realizada para verificar qué no se pueda agregar un device no existente a una home
Método HTTP: POST
URI: /api/homes/:homeId/devices
Body:
{
deviceId: 9999 (en este caso dicho device no existe)
}
Headers:
Authorization: Bearer {token}

Respuesta esperada:
400(BadRequest)
“Device not found”

Asociar un device a una home sin body

Esta prueba fue realizada para verificar qué no se pueda agregar un device si no se tiene un body
Método HTTP: POST
URI: /api/homes/:homeId/devices
Body:

(sin body)
Headers:
Authorization: Bearer {token}

Respuesta esperada:
415(UnsupportedMediaType)

Asociar un device a una home con datos inválidos

Esta prueba fue realizada para evitar qué un usuario envíe datos invalidos
Método HTTP: POST
URI: /api/homes/:homeId/devices
Body:
```
{
  devId: 1
}
```
Headers:
Authorization: Bearer {token}

Respuesta esperada:
400(BadRequest)

Creación de una empresa

Crear una empresa exitosamente

Esta prueba fue realizada para verificar qué un user pueda crear una company correctamente
Método HTTP: POST
URI: /api/companies
En el qué homeId representa el id de la home a la qué se está intentando agregar.
Body:
```
{
  "name": "Apple",
  "logotypeUrl": “apple.jpg",
  "rut": "255567"
}
```
Headers:
Authorization: Bearer {token}
Respuesta esperada:
200(Ok)

Un usuario qué no sea CompanyOwner intenta crear una empresa

Esta prueba fue realizada para verificar qué un usuario registrado como Administrador no pueda crear una company

Método HTTP: POST
URI: /api/companies

Body:
{
  "name": "Apple",
  "logotypeUrl": “apple.jpg",
  "rut": "255567"
}
Headers:
Authorization: Bearer {token} (este token debera ser el de un user con un rol distinto a CompanyOwner)
Respuesta esperada:
400()

Usuario qué ya tiene una empresa intenta crear una empresa

Esta prueba fue realizada para verificar qué un usuario el cual ya registro su empresa no pueda hacerrlo dos veces

Método HTTP: POST
URI: /api/companies
Body:
{
  "name": "Apple",
  "logotypeUrl": “apple.jpg",
  "rut": "255567"
}
Headers:
Authorization: Bearer {token}
Respuesta esperada:
400(User already has a company)

Usuario no autenticado intenta crear una company

Esta prueba fue realizada para verificar qué los usuarios qué puedan crear una company estén registrados
Método HTTP: POST
URI:/api/companies
Body:
{
  "name": "Apple",
  "logotypeUrl": “apple.jpg",
  "rut": "255567"
}
Headers:
(Sin el token de autenticación)

Respuesta esperada:
401(Unauthorized)

Usuario intenta crear una company con datos inválidos

Esta prueba fue realizada para verificar la integridad de los datos en el momento de crear una company
Método HTTP: POST
URI: /api/companies
Body:
{
  "ne": "Apple",
  "logotypeUrl": “apple.jpg",
  "rut": "255567"
}
Headers:
Authorization: Bearer {token}
Respuesta esperada:
400()

Detección de movimiento

Crear correctamente una notificación cuando se detecta movimiento

Esta prueba fue realizada para verificar qué se pueda crear una notificación a partir de una security camera.

Método HTTP: POST
URI:api/homes/:homeId/devices/:deviceId/notifications
Body:
“Motion”
Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Respuesta esperada:
200(Ok)

Creación de una notificación sin body

Esta prueba fue realizada para verificar qué se ingresen datos a la hora de crear una notificación
Método HTTP: POST
URI:api/homes/:homeId/devices/:deviceId/notifications
Body:
“Motion”
Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Respuesta esperada:
400(Ok)

Mantenimiento de cuentas de administrador

Crear un usuario de administrador correctamente

Esta prueba fue realizada para verificar qué los únicos usuarios qué son capaces de crear otros administradores, tengan el rol de administrador.
Método HTTP: POST
URI: /api/users
Body:

```
{
  "firstName": "Trevor",
  "lastName": "Jones",
  "email": "tjones@gmail.com",
  "password": "TrevorIsTheBest!",
  "role": "CompanyOwner",
  "imageUrl": "https://placehold.co/128"
}
```

Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Respuesta esperada:
200(Ok)

Crear un usuario de administrador sin body

Esta prueba fue realizada para verificar qué se ingresen datos a la hora de crear un administrador
Método HTTP: POST
URI: /api/users
Body:

Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Respuesta esperada:
400()

Creacion de administrador con datos inválidos

Esta prueba fue realizada para verificar qué se ingresen datos correctos a la hora de crear un administrador
Método HTTP: POST
URI: /api/users
Body:

```
{
  "fie": "Trevor",
  "lastName": "Jones",
  "email": "tjones@gmail.com",
  "pard": "TrevorIsTheBest!",
  "role": "CompanyOwner",
  "imageUrl": "https://placehold.co/128"
```

}
Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Resultado esperado:
400

Eliminar a un administrador correctamente

Esta prueba fue realizada para verificar qué cuando se quiere eliminar a un administrador se haga correctamente.
Método HTTP: DELETE
URI: /api/users/:userId
Body: (Vacio)
Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Resultado esperado:
204(No content)

Eliminar a un usuario no existente

Esta prueba fue realizada para verificar qué cuando se intenta eliminar a un usuario no existente salte un error
Método HTTP: DELETE
URI: /api/users/:userId (siendo este el id del usuario no existente)
Body: (Vacio)
Headers:
Authorization: Bearer {token} (Con el rol de administrador)
Resultado esperado:
400(User not found)