POST /v2/auth/login

request:

body: {

email: [xxx@xxx.com](mailto:xxx@xxx.com),

password: 123456

}

response:

200 body: {

user: {

id

email

name

lastname

}

token: jwt generado para la sesión

}

400 body: {

msg: ‘Usuario y/o contraseña incorrectos.’

|| … posibles errores de campos

}

500 body: {

msg: ‘Server error: contacte al administrador.’’

}

POST /v2/auth/renew

request:

header: {

token

}

response:

200 body: {

token

user: {

id

email

name

lastname

}

}

401 body: {

msg: ‘JWT no válido’

}

POST /v2/users/

request:

body: {

email

name

lastname

password

confirmPassword

}

response:

201 body: {

msg: ‘Registrado correctamente.’

}

400 body: {

msg: ‘Email ya registrado.’

|| msg: ’Las contraseñas no coinciden.’

|| msg: ‘Email no válido.’

… posibles errores de campos

}

GET /v2/users/:uid

*Si la request viene con token, y el token es del usuario solicitado, se devolverá información extra ya que se entiende que está entrando a su propio perfil, en caso de no coincidir el token, sólo se devolverá el email del usuario, por cuestiones de privacidad.*

request:

header: {

token

}

response:

200 body: {

user: {

email

name: opcional

lastname: opcional

}

}

404 body: {

msg: ‘Usuario no encontrado.’

}

POST /v2/events

*Creación de un nuevo evento*

request:

header: {

token

}

body: {

name

description

options: [{

start

end

}]

}

response:

201 body:{

msg: ‘Evento creado.’

id: *id del evento creado*

}

400 body: {

msg: …errores de validación

}

GET /v2/events/:id

*Devuelve la información del evento y un array con las optionId de los votos del usuario actual, en caso de ya haber votado.*

request:

header: {

token

}

response:

200 body: {

event: {

id

name

description

options:[{

id

start

end

votes: integer

}]

// participants: [ { id, email } ]

status

createdDate

owner: { id, email }

}

optionsVoted: [ optionId ]

}

404 body: { msg: ‘Evento no encontrado.’ }

POST /v2/events/:eventId/vote

*Se envía las ids de las opciones elegidas para efectuar los votos relacionados al user de la sesión actual.*

request:

header: {

token

}

body: {

[ optionId ]

}

response:

201 body: {

msg: ‘Votos registrados correctamente.’

}

400 body: {

msg: ‘Usted ya ha votado una opción para este evento.’

|| msg: ‘El dueño del evento cerró la votación.’

|| msg: ‘Error al registrar el voto.’

}

GET /v2/events?user=:uid

request:

header: {

token

}

response:

200 body: {

"myEvents": [

{

"id": "645ddade8ecbab481ccfbdb0",

"name": "Evento Aburrido",

"description": "Un evento que no es divertido",

"status": null,

"totalParticipants": 1

}

],

"participants": [

{

"id": "645ddade8ecbab481ccfbdb0",

"name": "Evento Aburrido",

"description": "Un evento que no es divertido",

"status": null,

"totalParticipants": 1

}

]

}

404 body: {

msg: ‘Usuario no encontrado’

}

PUT /v2/events/:id

*Utilizado para cambiar el status de un evento*

request:

header: {

token

}

body: {

status

}

response:

200 body: {

msg: ‘Evento actualizado’

}

400: {

msg: ‘Evento no encontrado.’

}

401: {

msg: ‘No posee autorización para editar este evento.’

}

GET /v2/monitoring

*Retorna un contador con la cantidad de eventos creados y horarios votados anotados en las últimas 2 horas.*

request:

header: {

token

}

response:

200 body: {

totalVotes

totalEvents

}