BYDLAC_API

This API allows frontend side to communicate with database.
More information: https://helloreverb.com
Contact Info: hello@helloreverb.com
Version: 1.0
BasePath:/api/
BSD 2-Clause "Simplified" License
http://apache.org/licenses/LICENSE-2.0.html

Access

Methods

[Jump to Models]

Table of Contents

Default

GET /groups/{groupId}/members/{userId}

Up

Add New User to a Group (**addGroupsGroupIdMembersUserId**)

Adds user with id equal to user_id to the group with id equal to group_id. Only moderators can add users to private groups. Any authenticated user can join non-private group, permisson: Authenticated

Path parameters

groupId (required)
Path Parameter —

userId (required)
Path Parameter —

Responses

200
OK
403
Forbidden
404
Not Found

DELETE /events/{eventId}

Up

Delete an Event (**deleteEventsEventId**)

Deletes an event, permisson: Authenticated, EventHost

Path parameters

eventId (required)
Path Parameter —

Responses
204
No Content
403
Forbidden
404
Not Found

DELETE /groups/{groupId}

Up

Delete Group (**deleteGroupsGroupId**)

Deletes group, permisson: Authenticated, GroupHost

Path parameters

groupId (required)
Path Parameter —

Responses
204
No Content
403
Forbidden
404
Not Found

DELETE /groups/{group_id}/messages/{messageId}

Up

Delete a Message (**deleteGroupsGroupIdMessagesMessageId**)

Deletes message with given message_id, permisson: Authenticated, (Author | Moderator)

Path parameters

group_id (required)
Path Parameter —

messageId (required)
Path Parameter —

Responses
200
OK

DELETE /groups/{groupId}/members/{userId}

Up

Delete User from a Group (**deleteGroupsGroupIdUserId**)

Removes user with id equal to user_id to the group with id equal to group_id. User can remove themself form the group. Only group host can remove moderators from the group. Host cannot be removed from the group., permisson: Authenticated, (Self | Moderator)

Path parameters

groupId (required)
Path Parameter —

userId (required)
Path Parameter —

Responses
204
No Content
403
Forbidden
404
Not Found

DELETE /users/{userId}

Up

Delete User (**deleteUsersUserId**)

Deletes user, permisson: AdminUser.

Path parameters

userId (required)
Path Parameter — Id of an existing user.

Responses
204
No Content
403
Forbidden
404
Not Found

GET /events

Up

Get All Events (**getEvents**)

Gets list of all events, permisson: Authenticated

Responses
200
OK
403
Forbidden

GET /events/{eventId}

Up

Get Event Info by Event ID (**getEventsEventId**)

Returns event with given id, permisson: Authenticated

Path parameters

eventId (required)
Path Parameter —

Return type
Event

Example data
Content-Type: application/json


```
{
  "expires" : "2000-01-23T04:56:07.000+00:00",
  "created" : "2000-01-23T04:56:07.000+00:00",
  "members" : { },
  "host" : 6,
  "name" : "name",
  "description" : "description",
  "location" : "http://example.com/aeiou",
  "id" : 0,
  "group" : 1,
  "maxParticipants" : 5
}
```


Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK Event
404
Not Found

GET /events/{eventId}/group

Up

Get Group Data by Event ID (**getEventsEventIdGroup**)

Retrieves data of a group of the event, to retreive this data user have to prticipate in the event, permisson: Authenticated, Participant

Path parameters

eventId (required)
Path Parameter —

Return type
ConversationGroup

Example data
Content-Type: application/json

```
{
  "is_private" : true,
  "created" : "created",
  "host" : 6,
  "name" : "name",
  "description" : "description",
  "id" : 0,
  "updated" : "updated"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK ConversationGroup
403
Forbidden
404
Not Found

GET /events/{eventId}/join

Up

Join an Event (**getEventsEventIdJoin**)

User sending request joins event, creates a link between the user and the event, permisson: Authenticated

Path parameters

eventId (required)
Path Parameter —

Responses
200
OK
404
Not Found

GET /events/{eventId}/leave

Up

Leave an Event (**getEventsEventIdLeave**)

User sending request leaves event, deletes a link between the user and the event, permisson: Authenticated

Path parameters

eventId (required)
Path Parameter —

Responses
200
OK
404
Not Found

GET /groups

Up

Get All Non-Private Groups (**getGroups**)

Returns list of non-private groups, permisson: Authenticated.

Responses
200
OK
400
Bad Request
403
Forbidden
404
Not Found

GET /groups/all

Up

Get All Groups (**getGroupsAll**)

Returns list of all groups, permisson: AdminUser

Responses

GET /groups/{groupId}

Up

Get Group Info by Group ID (**getGroupsGroupId**)

Retreives information of group with given id, permisson: Authenticated, Member.

Path parameters

groupId (required)
Path Parameter —

Return type
ConversationGroup

Example data
Content-Type: application/json

```
{
  "is_private" : true,
  "created" : "created",
  "host" : 6,
  "name" : "name",
  "description" : "description",
  "id" : 0,
  "updated" : "updated"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK ConversationGroup
Example data
Content-Type: Example 1

```
{"id":1,"host":3,"name":"Group 1","description":"","is_private":true,"updated":"2022-11-12T21:07:28.282995Z","created":"
```

403
Forbidden
404
Not Found

GET /groups/{groupId}/members

Up

Get Group Members by Group ID (**getGroupsGroupIdMembers**)

Returns list of group users that are members of the given group. If group is private, user must be a member of the group, permisson: Authenticated

Path parameters

groupId (required)
Path Parameter —

Responses
200
OK
403
Forbidden
404
Not Found

GET /groups/{groupId}/messages

Up

Get All Messages Sent to a Group (**getGroupsGroupIdMessages**)

Gets all messages sent to group with given group_id, permisson: Authenticated, Member

Path parameters

groupId (required)
Path Parameter —

Responses
200
OK
403
Forbidden
404
Not Found

GET /groups/{group_id}/messages/{messageId}

Up

Get a Message by Group ID (**getGroupsGroupIdMessagesMessageId**)

Gets message with given message id, user sending request must be a member of the group, permisson: Authenticated, Member

Path parameters

group_id (required)
Path Parameter —

messageId (required)
Path Parameter —

Return type
Message

Example data
Content-Type: application/json

```
{
  "author" : 6,
  "create" : "2000-01-23T04:56:07.000+00:00",
  "id" : 0,
  "body" : "body",
  "updated" : "2000-01-23T04:56:07.000+00:00",
  "group" : 1
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK Message
403
Forbidden
404
Not Found

GET /logout

Up

Logout User (**getLogout**)

Logs out user with data sent in get request, permisson: Any.

Responses
202
Accepted

GET /users

Up

Get All Users (**getUsers**)

Returns list of all registered users, permisson: AdminUser.

Responses
200
OK
403
Authentication credentials were not provided or user does not have permission to do this action.

GET /users/{userId}

Up

Get User Info by User ID (**getUsersUserId**)

Returns user with given id, permisson: Authenticated.

Path parameters

userId (required)
Path Parameter — Id of an existing user.

Return type
User

Example data
Content-Type: application/json

```
{
  "created" : "2000-01-23T04:56:07.000+00:00",
  "bio" : "bio",
  "id" : 0,
  "profileImage" : { },
  "email" : "",
  "username" : "username"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
User Found User
Example data
Content-Type: Get User AliceSmith

```
{"id":142,"username":"AliceSmith","email":"alice.smith@gmail.com","bio":"Hi i am Alice","created":"2022-11-09T17:50:05.9
```

404
User Not Found

`PATCH /events/{eventId}`

Up

Update an Event (**patchEventsEventId**)

Updates an event, permisson: Authenticated, EventHost

Path parameters

eventId (required)
Path Parameter —

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body object (optional)
Body Parameter — Patch data to update event
example: {

```
  "value" : {
    "description" : "New description",
    "expires" : "2019-08-24T14:15:22Z"
  }
}
```

Responses
200
OK
400
Bad Request
403
Forbidden
404
Not Found

`PATCH /groups/{groupId}`

Up

Update Group Information (**patchGroupsGroupId**)

Updates group's data, permisson: Authenticated, Moderator

Path parameters

groupId (required)
Path Parameter —

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body object (optional)
Body Parameter — Patch information to update group.
example: {
"value" : {
"name" : "New name"
}
}

Return type
ConversationGroup

Example data
Content-Type: application/json

```
{
  "is_private" : true,
  "created" : "created",
  "host" : 6,
  "name" : "name",
  "description" : "description",
  "id" : 0,
  "updated" : "updated"
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK ConversationGroup
400
Bad Request
403
Forbidden
404
Not Found

`PATCH /groups/{group_id}/messages/{messageId}`

Up

Update a Message (**patchGroupsGroupIdMessagesMessageId**)

Updates message with given message_id. Only author of the group can update message, permisson: Authenticated, Author

Path parameters

group_id (required)
Path Parameter —

messageId (required)
Path Parameter —

Return type
Message

Example data
Content-Type: application/json

```
{
  "author" : 6,
  "create" : "2000-01-23T04:56:07.000+00:00",
  "id" : 0,
  "body" : "body",
  "updated" : "2000-01-23T04:56:07.000+00:00",
  "group" : 1
```

```
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK Message
400
Bad Request
403
Forbidden
404
Not Found

PATCH /groups/{groupId}/members/{userId}

Up

Change User Role in a Group (**patchGroupsGroupIdUserId**)

Changes moderator status of user with id equal to user_id to the group with id equal to group_id. Only group host can change moderator status of other moderators. Group host's moderator status cannot be changed. permisson: Authenticated

Path parameters

groupId (required)
Path Parameter —

userId (required)
Path Parameter —

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body object (optional)
Body Parameter — Data about user state in group.
example:
```
{
  "value" : {
    "is_moderator" : false
  }
}
```

Return type
GroupMember

Example data
Content-Type: application/json

```
{
  "id" : 0,
  "is_moderator" : false,
  "user" : 6,
  "group" : 1
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK GroupMember
400
Bad Request
403
Forbidden
404
Not Found

PATCH /users/{userId}

Up

Update User Information (**patchUsersUserId**)

Updates user bio and profile_image (Note: email and username cannot be changed once set), permisson: Authenticated, Self.

Path parameters

userId (required)
Path Parameter — Id of an existing user.

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body users_userId_body (optional)
Body Parameter — Patch user properties to update.
example:
```
{
  "value" : {
    "bio" : "I am Rebecca"
  }
}
```

Produces
This API call produces the following media types according to the Accept request header; the media type will be conveyed by the Content-Type response header.

- `application/json`

Responses
200
OK
400
Bad Request
403
User Updated User
Example data
Content-Type: Update User AliceSmith


```
{"id":142,"username":"AliceSmith","email":"alice.smith@gmail.com","bio":"Hi i am Alice","created":"2022-11-09T17:50:05.9
```


404
User Not Found

POST /events

Up

Create an Event (**postEvents**)

Creates an event, permisson: Authenticated

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body object (optional)
Body Parameter — Post data to create event.
example:
```
{
  "value" : {
    "name" : "Socializing game",
    "description" : "Very good game",
    "max_participants" : 5,
    "location" : "http://example.com",
    "expires" : "2019-08-24T14:15:22Z"
  }
}
```

Responses
201
Created
400
Bad Request
403
Forbidden

POST /events/{eventId}/group

Up

Create Group of an Event (**postEventsEventIdGroup**)

Creates a group for the event, permisson: Authenticated, EventHost

Path parameters

eventId (required)
Path Parameter —

Responses
200
OK

POST /groups

Up

Create New Group (**postGroups**)

Creates new group with data sent in post request, permisson: Authenticated

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body object (optional)
Body Parameter — Post data to create group.
example: {
"value" : {
"name" : "Group1",
"description" : "Very good group",
"is_private" : false
}
}

Responses
201
Created
400
Bad Request
403
Forbidden

POST /groups/{groupId}/messages

Up

Send a Message to a Group (**postGroupsGroupIdMessages**)

Sends message to the group with id equal to group_id, permisson: Authenticated, Member

Path parameters

groupId (required)
Path Parameter —

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body object (optional)
Body Parameter — Post data to create message.
example: {
"value" : {
"body" : "New message",
"author" : 65
}
}

Responses
201
Created
400
Bad Request
403
Forbidden
404
Not Found

POST /login

Up

Login User (**postLogin**)

Logs in user with data sent in post request, permisson: Any.

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body **login_body** **(optional)**
Body Parameter — Post necessary for a user to be logged in. Both fields need to be filled.
example:
```
{
  "value" : {
    "email" : "user@example.com",
    "password" : "pa$$word"
  }
}
```

Responses
202
Accepted
400
Bad Request

POST /register

Up

Register User (**postRegister**)

Registers user with data sent in post request, permisson: Any.

Consumes
This API call consumes the following media types via the Content-Type request header:

- `application/json`

Request body

body **register_body** **(optional)**
Body Parameter — Post data to register user.
example:
```
{
  "value" : {
    "username" : "user",
    "password" : "pa$$word",
    "password2" : "pa$$word",
    "email" : "user@example.com"
  }
}
```

Responses
201
Created
400
Data provided were insufficient to create a user or invalid.

Models

[Jump to Methods]

Table of Contents

`ConversationGroup` - ConversationGroup

Up

Model of conversation group keeps data about conversations.

id
Integer Unique identifier for the conversation group.

host
Integer Id of the group founder.

name
String

description (optional)
String

is_private
Boolean If group is private users have to be added to the group by its moderator.

updated
String Last time message was sent to the group or its properties changed.

created
String The date that the conversation group was created.

`Event` - Event

Up

Model of the events keeps data of the events.

id
Integer Unique indentifier of the event.

host
Integer Id of the user that founded the group.

group (optional)
Integer Id of the group linked to the given event. Note: there might not be any group linked.

members (optional)
Object Many to many field conected with user model. Describes which users participate in the given event.

name
String

description (optional)
String

maxParticipants (optional)
Integer

location (optional)
String Link to location from Google Maps API. format: uri

expires
Date Event's expiration time and date. format: date-time

created
Date Time and date when event was created. format: date-time

`GroupMember` - GroupMember

Up

Model keeps information about users memberships in groups.

id
Integer Unique identifier.

user
Integer Id of user that is member.

group
Integer Id of group of which user is member.

is_moderator (optional)
Boolean Is user group moderator.

`Message` - Message

Up

Message model keeps

id
Integer Unique identifier for the given message.

author
Integer Id of user who created given message.

group
Integer Id of group to which the given message was sent.

body
String Body of the message.

updated
Date The date that the message was created. format: date-time

create
Date The date that the message was updated last time. format: date-time

`User` - User

Up

id
Integer Unique identifier for the given user.

email

String format: email

username
String

profileImage (optional)
Object Profile image stored locally at server.

bio (optional)
String User description.

created
Date The date that the user was created. format: date-time

`login_body` Up

email
String format: email

password
String format: password

`register_body` Up

username
String

password
String format: password

password2
String format: password

email
String format: email

`users_userId_body` Up

profileImage (optional)
Object

bio (optional)
String