

# GatorShare API

## Overview

Backend APIs for the GatorShare app

## Version information

*Version* : 1.0

## URI scheme

*Host* : localhost:8080

*BasePath* : /api/v1

## Paths

### Creates a new comment on a post

POST /comment/{postId}

### Description

This API allows a logged-in user to add a new comment to a specific post.

### Parameters

Type	Name	Description	Schema
Path	postId <i>required</i>	Post ID	string
Body	comment <i>required</i>	New comment data	<a href="#">models.Comment</a>

### Responses

HTTP Code	Description	Schema
200	Created comment details	<a href="#">models.Comment</a>
400	Bad Request	string
401	Unauthorized	string

HTTP Code	Description	Schema
404	Post not found	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `comment`

# Retrieves all comments for a specific post

```
GET /comment/{postId}/
```

## Description

This API fetches all comments associated with a specific post, ordered by creation date (latest first).

## Parameters

Type	Name	Description	Schema
Path	<code>postId</code> <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	List of comments	< <a href="#">models.Comment</a> > array
400	Bad Request	string
404	Post not found	string

## Consumes

¥ `application/json`

## Produces

¥ application/json

## Tags

¥ comment

# Retrieves a specific comment by its ID

```
GET /comment/{postId}/{commentId}
```

## Description

This API fetches a comment associated with a specific post by its unique comment ID.

## Parameters

Type	Name	Description	Schema
Path	commentId <i>required</i>	Comment ID	string
Path	postId <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	Comment details	<a href="#">models.Comment</a>
400	Bad Request	string
404	Comment not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ comment

# Updates an existing comment

```
PUT /comment/{postId}/{commentId}
```

## Description

This API allows the author of a comment to update its content.

## Parameters

Type	Name	Description	Schema
Path	commentId <i>required</i>	Comment ID	string
Path	postId <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	Updated comment details	<a href="#">models.Comment</a>
400	Bad Request or Empty Content	string
401	Unauthorized	string
403	Forbidden - Only the author can update their own comments	string
404	Comment not found	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `comment`

# Deletes an existing comment

```
DELETE /comment/{postId}/{commentId}
```

## Description

This API allows the author of a comment to delete it from a specific post.

## Parameters

Type	Name	Description	Schema
Path	commentId <i>required</i>	Comment ID	string
Path	postId <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	Comment deleted successfully	string
400	Bad Request	string
401	Unauthorized	string
403	Forbidden - Only the author can delete their own comments	string
404	Comment not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ comment

# Like or dislike action on a comment

```
PUT /comment/{postId}/{commentId}/like-dislike
```

## Description

This API allows a logged-in user to like or dislike a specific comment. The action is specified in the request body as either "like" or "dislike".

## Parameters

Type	Name	Description	Schema
Path	commentId <i>required</i>	Comment ID	string
Path	postId <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	Action applied successfully with updated like/dislike counts	object
400	Invalid action	string
401	Unauthorized	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ comment

# Gets the current logged-in member

GET /current-user

## Description

This API returns the username of the currently logged-in Member

## Responses

HTTP Code	Description	Schema
200	Success	string
401	Unauthorized	string
403	Forbidden	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Logs in an existing member

POST /login

## Description

This API is used to login a member by using the stored credentials in the database

## Parameters

Type	Name	Description	Schema
Body	member <i>required</i>	Member username and password	<a href="#">models.Member</a>

## Responses

HTTP Code	Description	Schema
200	Success	string
400	Bad Request	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Logs out a currently logged in member

POST /logout

## Description

This API clears the tokens in the cookies and database for the logged in Member

## Parameters

Type	Name	Description	Schema
Body	<i>member required</i>	Member username	<a href="#">models.Member</a>

## Responses

HTTP Code	Description	Schema
200	Success	string
401	Unauthorized	string
404	Not Found	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ member

# Gets a list of members

GET /member

## Description

Gets a slice of members using the limit and offset parameters, sorts based on the column and order (desc or asc) parameters, and filters based off the search\_key parameter



## Responses

HTTP Code	Description	Schema
200	Success	string
400	Bad Request	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `member`

# Updates a members information

PUT `/member`

## Description

This API updates the field values for the logged-in Member

## Parameters

Type	Name	Description	Schema
Body	<code>member</code> <i>required</i>	Updated member info	<a href="#">models.Member</a>

## Responses

HTTP Code	Description	Schema
200	Success	string
400	Bad Request	string
401	Unauthorized	string
404	Not Found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Deletes a member from the system

```
DELETE /member
```

## Description

This API removes the Member entity from the database for the logged-in Member

## Parameters

Type	Name	Description	Schema
Body	member <i>required</i>	Member username	<a href="#">models.Member</a>

## Responses

HTTP Code	Description	Schema
200	Success	string
401	Unauthorized	string
404	Not Found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Handles OPTIONS requests

OPTIONS /member

## Description

This API handles OPTIONS requests for CORS preflight

## Responses

HTTP Code	Description	Schema
200	Success	string

## Consumes

application/json

## Produces

application/json

## Tags

member

# Gets a member's info by their username

GET /member/{username}

## Description

This API fetches a Member entity by their unique username

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username	string

## Responses

HTTP Code	Description	Schema
200	Success	string
400	Bad Request	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Retrieves comments disliked by a specific user

```
GET /member/{username}/disliked-comments
```

## Description

This API fetches all comments that have been disliked by a specific user, identified by their username.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member	string

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">models.Comment</a> > array
400	Bad Request	string
401	Unauthorized	string
404	User not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Retrieves posts disliked by a specific user

```
GET /member/{username}/disliked-posts
```

## Description

This API fetches all posts that have been disliked by a specific user, identified by their username.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member	string

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">models.Post</a> > array
400	Bad Request	string
404	User not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Follow a member

POST /member/{username}/follow

## Description

Allows the logged-in user to follow another member by username.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member to follow	string

## Responses

HTTP Code	Description	Schema
200	Followed successfully	string
400	Cannot follow yourself or bad request	string
401	Unauthorized	string
404	User not found	string
500	Failed to follow	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Unfollow a member

DELETE /member/{username}/follow

## Description

Allows the logged-in user to unfollow another member by username.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member to unfollow	string

## Responses

HTTP Code	Description	Schema
200	Unfollowed successfully	string
400	Cannot unfollow yourself or bad request	string
401	Unauthorized	string
404	User not found	string
500	Failed to unfollow	string

## Consumes

application/json

## Produces

application/json

## Tags

member

## Get a member's followers

GET /member/{username}/followers

## Description

Retrieves a list of members who follow the specified user.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member	string

## Responses

HTTP Code	Description	Schema
200	List of followers	object
404	User not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Get members a user is following

```
GET /member/{username}/following
```

## Description

Retrieves a list of members that the specified user is following.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member	string

## Responses

HTTP Code	Description	Schema
200	List of following	object



HTTP Code	Description	Schema
404	User not found	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `member`

# Retrieves comments liked by a specific user

```
GET /member/{username}/liked-comments
```

## Description

This API fetches all comments that have been liked by a specific user, identified by their username.

## Parameters

Type	Name	Description	Schema
Path	<code>username</code> <i>required</i>	Username of the member	string

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">models.Comment</a> > array
400	Bad Request	string
401	Unauthorized	string
404	User not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

# Retrieves posts liked by a specific user

```
GET /member/{username}/liked-posts
```

## Description

This API fetches all posts that have been liked by a specific user, identified by their username.

## Parameters

Type	Name	Description	Schema
Path	username <i>required</i>	Username of the member	string

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">models.Post</a> > array
400	Bad Request	string
404	User not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ member

## Retrieves posts by a specific user

```
GET /member/{username}/posts
```

### Description

This API fetches all posts created by a specific member

### Parameters

Type	Name	Description	Schema
Body	member <i>required</i>	Member username	<a href="#">models.Member</a>

### Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">models.Post</a> > array
500	Internal Server Error	string

### Consumes

¥ [application/json](#)

### Produces

¥ [application/json](#)

## Tags

¥ post

## Sends a notification to a user

```
POST /notification
```

## Description

Sends a notification with a title and content to a specified user.

## Parameters

Type	Name	Description	Schema
Body	notification <i>required</i>	Notification data	<a href="#">models.Notification</a>

## Responses

HTTP Code	Description	Schema
201	Notification sent	string
400	Notification title, content, and username (recipient) required	string
401	Unauthorized	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ notification

# Gets notifications for the current user

GET /notification

## Description

Fetches a slice of notifications for the logged-in user. Supports optional query parameters for sorting, limit, and offset.

## Responses

HTTP Code	Description	Schema
200	Notifications list and count	object

HTTP Code	Description	Schema
400	Bad Request	string
401	Unauthorized	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ notification

# Bulk update notifications' read status

PUT /notification

## Description

Sets all notifications for the logged-in user to read/unread.

## Responses

HTTP Code	Description	Schema
200	Notifications updated successfully	string
400	Bad Request	string
401	Unauthorized	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ notification

## Gets a notification by its ID

```
GET /notification/{id}
```

### Description

Fetches a single notification by its unique ID.

### Parameters

Type	Name	Description	Schema
Path	id <i>required</i>	Notification ID	string

### Responses

HTTP Code	Description	Schema
200	OK	<a href="#">models.Notification</a>
400	No Records Found	string
401	Unauthorized	string

### Consumes

🔗 [application/json](#)

### Produces

🔗 [application/json](#)

### Tags

🔗 [notification](#)

## Updates a notification's read status

```
PUT /notification/{id}
```

### Description

Updates the read status of a notification by its unique ID. Only the recipient can update their own notifications.

## Parameters

Type	Name	Description	Schema
Path	id <i>required</i>	Notification ID	string

## Responses

HTTP Code	Description	Schema
200	Notification updated successfully	string
400	You can only update your own notifications	string
401	Unauthorized	string
404	Notification not found	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ notification

# Deletes a notification

```
DELETE /notification/{id}
```

## Description

Deletes a notification by its unique ID. Only the recipient can delete their own notifications.

## Parameters

Type	Name	Description	Schema
Path	id <i>required</i>	Notification ID	string

## Responses

HTTP Code	Description	Schema
200	Notification deleted successfully	string
400	Notification not found or not owned by user	string
401	Unauthorized	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `notification`

# Creates a new post

POST `/post`

## Description

This API creates a new post for the logged-in member

## Parameters

Type	Name	Description	Schema
Body	<code>post</code> <i>required</i>	New post	<a href="#">models.Post</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">models.Post</a>
400	Bad Request	string
401	Unauthorized	string



## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ post

## Retrieves posts

GET /post

## Description

Gets a slice of posts using the limit and offset parameters, sorts based on the column and order (desc or asc) parameters, and filters based off the search\_key parameter

## Responses

HTTP Code	Description	Schema
200	OK	< <a href="#">models.Post</a> > array
400	Bad Request	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ post

## Retrieves a specific post by ID

GET /post/{postId}

## Description

This API fetches a post and its comments by the post ID

## Parameters

Type	Name	Description	Schema
Path	postId <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">models.Post</a>
400	Bad Request	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `post`

# Updates a post

```
PUT /post/{postId}
```

## Description

This API updates a post's content if the logged-in member is the author

## Parameters

Type	Name	Description	Schema
Path	postId <i>required</i>	Post ID	string
Body	post <i>required</i>	Updated post	<a href="#">models.Post</a>

## Responses

HTTP Code	Description	Schema
200	OK	<a href="#">models.Post</a>
400	Bad Request	string
401	Unauthorized	string
403	Forbidden	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ `post`

## Deletes a post

```
DELETE /post/{postId}
```

## Description

This API deletes a post by its ID if the logged-in member is the author

## Parameters

Type	Name	Description	Schema
Path	<code>postId</code> <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	OK	string
400	Bad Request	string
401	Unauthorized	string

HTTP Code	Description	Schema
403	Forbidden	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ post

# Increments the view count of a post

```
PUT /post/{postId}/increment-views
```

## Description

This API increments the view count of a specific post by its ID

## Parameters

Type	Name	Description	Schema
Path	postId <i>required</i>	Post ID	string

## Responses

HTTP Code	Description	Schema
200	OK	string
400	Bad Request	string

## Consumes

¥ application/json

## Produces

¥ application/json

## Tags

¥ post

## Likes or dislikes a post

```
PUT /post/{postId}/like-dislike
```

### Description

This API allows a logged-in user to like or dislike a specific post. The action is specified in the request body as either "like" or "dislike".

### Parameters

Type	Name	Description	Schema
Path	postId <i>required</i>	Post ID	string

### Responses

HTTP Code	Description	Schema
200	Action applied successfully with updated like/dislike counts	object
400	Bad Request or Invalid Action	string
401	Unauthorized	string
404	Post not found	string

### Consumes

¥ application/json

### Produces

¥ application/json

## Tags

¥ post

## Registers a new member

```
POST /register
```

## Description

This API is used to add a Member entity to the database

## Parameters

Type	Name	Description	Schema
Body	member <i>required</i>	New member	<a href="#">models.Member</a>

## Responses

HTTP Code	Description	Schema
201	Created	string
400	Bad Request	string

## Consumes

¥ `application/json`

## Produces

¥ `application/json`

## Tags

¥ member

# Definitions

## models.Comment

Name	Description	Schema
author <i>optional</i>		string
comment_id <i>optional</i>		string
content <i>optional</i>		string
createdAt <i>optional</i>		string

Name	Description	Schema
disliked_comments <i>optional</i>		< <a href="#">models.Member</a> > array
dislikes <i>optional</i>		integer
liked_comments <i>optional</i>	Relationships	< <a href="#">models.Member</a> > array
likes <i>optional</i>		integer
post_id <i>optional</i>		string

## models.Member

Name	Description	Schema
bio <i>optional</i>		string
createdAt <i>optional</i>		string
csrf_token <i>optional</i>		string
disliked_comments <i>optional</i>		< <a href="#">models.Comment</a> > array
disliked_posts <i>optional</i>		< <a href="#">models.Post</a> > array
email <i>optional</i>		string
followers <i>optional</i>		< <a href="#">models.Member</a> > array
following <i>optional</i>		< <a href="#">models.Member</a> > array
liked_comments <i>optional</i>		< <a href="#">models.Comment</a> > array
liked_posts <i>optional</i>	Relationships	< <a href="#">models.Post</a> > array

Name	Description	Schema
password <i>optional</i>		string
session_token <i>optional</i>		string
updatedAt <i>optional</i>		string
username <i>optional</i>		string

## models.Notification

Name	Schema
content <i>optional</i>	string
createdAt <i>optional</i>	string
id <i>optional</i>	string
read <i>optional</i>	boolean
title <i>optional</i>	string
username <i>optional</i>	string

## models.Post

Name	Description	Schema
author <i>optional</i>		string
comments <i>optional</i>		< <a href="#">models.Comment</a> > array
content <i>optional</i>		string
createdAt <i>optional</i>		string
disliked_by_members <i>optional</i>		< <a href="#">models.Member</a> > array



Name	Description	Schema
dislikes <i>optional</i>		integer
images <i>optional</i>		< string > array
liked_by_members <i>optional</i>	Relationships	< <a href="#">models.Member</a> > array
likes <i>optional</i>		integer
post_id <i>optional</i>		string
title <i>optional</i>		string
views <i>optional</i>		integer