Privacy Control API

Author: Kibuti, Backend Developer
Last Updated: 2025-12-11
Version: v1.0

Base URL: `https://api.nexgate.com/api/v1`

Short Description: The Privacy Control API allows users to block and mute other users to control their social interactions. Blocking removes all follow relationships and prevents future interactions, while muting hides content without breaking the follow relationship. These features give users granular control over their social experience.

Hints:

- All endpoints require authentication via Bearer token
- Blocking a user automatically unfollows both users from each other
- Muting only affects content visibility and does not break follow relationships
- Users cannot block or mute themselves
- Blocked and muted lists are retrieved in reverse chronological order (newest first)

Standard Response Format

All API responses follow a consistent structure using our Globe Response Builder pattern:

Success Response Structure

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Operation completed successfully",
  "action_time": "2025-09-23T10:30:45",
  "data": {
    // Actual response data goes here
  }
}
```

Error Response Structure

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Error description",
  "action_time": "2025-09-23T10:30:45",
  "data": "Error description"
}
```

Standard Response Fields

Field	Type	Description
`success`	boolean	Always `true` for successful operations, `false` for errors
`httpStatus`	string	HTTP status name (OK, BAD_REQUEST, NOT_FOUND, etc.)
`message`	string	Human-readable message describing the operation result
`action_time`	string	ISO 8601 timestamp of when the response was generated
`data`	object/string	Response payload for success, error details for failures

---

HTTP Method Badge Standards

For better visual clarity, all endpoints use colored badges for HTTP methods with the following standard colors:

- **GET** - GET - Green (Safe, read-only operations)
- **POST** - POST - Blue (Create new resources)
- **DELETE** - DELETE - Red (Remove resources)

Endpoints

1. Block User

Purpose: Block a user to prevent all interactions and remove existing follow relationships

Endpoint: `POST` `{base_url}/e-social/privacy-control/block/{userId}`

Access Level: □□ Protected (Requires Bearer Token Authentication)

Authentication: Bearer Token

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer token for authentication (format: `Bearer <token>`)
Content-Type	string	Yes	Must be `application/json`

Path Parameters:

Parameter	Type	Required	Description	Validation
userId	string	Yes	UUID of the user to block	Must be valid UUID format

Success Response JSON Sample:


```
{
  "success": true,
  "httpStatus": "OK",
  "message": "User blocked successfully",
  "action_time": "2025-12-11T10:30:45",
  "data": {
    "id": "880e8400-e29b-41d4-a716-446655440033",
    "user": {
      "id": "777e7777-e77b-77d7-a777-777777777777",
      "userName": "spam_account",
      "firstName": "Spam",
      "lastName": "Account",
      "profilePictureUrls": ["https://cdn.nexgate.com/profiles/spam_account.jpg"],
      "isVerified": false
    },
    "createdAt": "2025-12-11T10:30:45"
```

```
  }
}
```

Success Response Fields:

Field	Description
id	Unique identifier for the block record (UUID format)
user	Object containing information about the blocked user
user.id	UUID of the blocked user
user.userName	Username of the blocked user
user.firstName	First name of the blocked user
user.lastName	Last name of the blocked user
user.profilePictureUrls	Array of profile picture URLs for the blocked user
user.isVerified	Boolean indicating if the blocked account is verified
createdAt	Timestamp when the block was created (ISO 8601 format)

Error Response JSON Sample:

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Cannot block yourself",
  "action_time": "2025-12-11T10:30:45",
  "data": "Cannot block yourself"
}
```

Standard Error Types:

Application-Level Exceptions (400-499)

- `400 BAD_REQUEST`: Cannot block yourself or user already blocked
- `401 UNAUTHORIZED`: Authentication issues (empty, invalid, expired, or malformed tokens)
- `404 NOT_FOUND`: User to block not found
- `500 INTERNAL_SERVER_ERROR`: Unexpected server errors

Error Response Examples:

Bad Request - Self Block (400):

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Cannot block yourself",
  "action_time": "2025-12-11T10:30:45",
  "data": "Cannot block yourself"
}
```


Bad Request - Already Blocked (400):


```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "User already blocked",
  "action_time": "2025-12-11T10:30:45",
  "data": "User already blocked"
}
```


Not Found (404):


```
{
  "success": false,
  "httpStatus": "NOT_FOUND",
  "message": "User not found",
  "action_time": "2025-12-11T10:30:45",
  "data": "User not found"
}
```


---

2. Unblock User

Purpose: Remove a block on a previously blocked user

Endpoint: DELETE `{base_url}/e-social/privacy-control/unblock/{userId}`

Access Level: □□ Protected (Requires Bearer Token Authentication)

Authentication: Bearer Token

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer token for authentication (format: `Bearer <token>`)
Content-Type	string	Yes	Must be `application/json`

Path Parameters:

Parameter	Type	Required	Description	Validation
userId	string	Yes	UUID of the user to unblock	Must be valid UUID format

Success Response JSON Sample:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "User unblocked successfully",
  "action_time": "2025-12-11T10:30:45",
  "data": null
}
```

Success Response Fields:

Field	Description
data	Always null for this endpoint

Error Response JSON Sample:

```
{
  "success": false,
  "httpStatus": "NOT_FOUND",
  "message": "User not found",
  "action_time": "2025-12-11T10:30:45",
  "data": "User not found"
}
```

Standard Error Types:

Application-Level Exceptions (400-499)

- `401 UNAUTHORIZED`: Authentication issues (empty, invalid, expired, or malformed tokens)

- `404 NOT_FOUND`: User to unblock not found
- `500 INTERNAL_SERVER_ERROR`: Unexpected server errors

Error Response Examples:

Not Found (404):

```
{
  "success": false,
  "httpStatus": "NOT_FOUND",
  "message": "User not found",
  "action_time": "2025-12-11T10:30:45",
  "data": "User not found"
}
```

---

3. Mute User

Purpose: Mute a user to hide their content without breaking the follow relationship

Endpoint: `POST` `{base_url}/e-social/privacy-control/mute/{userId}`

Access Level: □□ Protected (Requires Bearer Token Authentication)

Authentication: Bearer Token

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer token for authentication (format: `Bearer <token>`)
Content-Type	string	Yes	Must be `application/json`

Path Parameters:

Parameter	Type	Required	Description	Validation
userId	string	Yes	UUID of the user to mute	Must be valid UUID format

Success Response JSON Sample:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "User muted successfully",
  "action_time": "2025-12-11T10:30:45",
  "data": {
    "id": "990e8400-e29b-41d4-a716-446655440044",
    "user": {
      "id": "888e8888-e88b-88d8-a888-888888888888",
      "userName": "noisy_user",
      "firstName": "Noisy",
      "lastName": "User",
      "profilePictureUrls": ["https://cdn.nexgate.com/profiles/noisy_user.jpg"],
      "isVerified": true
    },
    "createdAt": "2025-12-11T10:30:45"
  }
}
```


Success Response Fields:

Field	Description
id	Unique identifier for the mute record (UUID format)
user	Object containing information about the muted user
user.id	UUID of the muted user
user.userName	Username of the muted user
user.firstName	First name of the muted user
user.lastName	Last name of the muted user
user.profilePictureUrls	Array of profile picture URLs for the muted user
user.isVerified	Boolean indicating if the muted account is verified
createdAt	Timestamp when the mute was created (ISO 8601 format)

Error Response JSON Sample:


```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Cannot mute yourself",
```

```
  "action_time": "2025-12-11T10:30:45",
  "data": "Cannot mute yourself"
}
```

Standard Error Types:

Application-Level Exceptions (400-499)

- `400 BAD_REQUEST`: Cannot mute yourself or user already muted
- `401 UNAUTHORIZED`: Authentication issues (empty, invalid, expired, or malformed tokens)
- `404 NOT_FOUND`: User to mute not found
- `500 INTERNAL_SERVER_ERROR`: Unexpected server errors

Error Response Examples:

Bad Request - Self Mute (400):

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "Cannot mute yourself",
  "action_time": "2025-12-11T10:30:45",
  "data": "Cannot mute yourself"
}
```

Bad Request - Already Muted (400):

```
{
  "success": false,
  "httpStatus": "BAD_REQUEST",
  "message": "User already muted",
  "action_time": "2025-12-11T10:30:45",
  "data": "User already muted"
}
```

Not Found (404):

```
{
  "success": false,
  "httpStatus": "NOT_FOUND",
  "message": "User not found",
  "action_time": "2025-12-11T10:30:45",
```

```
  "data": "User not found"
}
```

---

4. Unmute User

Purpose: Remove a mute on a previously muted user

Endpoint: DELETE `{base_url}/e-social/privacy-control/unmute/{userId}`

Access Level: □□ Protected (Requires Bearer Token Authentication)

Authentication: Bearer Token

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer token for authentication (format: `Bearer <token>`)
Content-Type	string	Yes	Must be `application/json`

Path Parameters:

Parameter	Type	Required	Description	Validation
userId	string	Yes	UUID of the user to unmute	Must be valid UUID format

Success Response JSON Sample:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "User unmuted successfully",
  "action_time": "2025-12-11T10:30:45",
  "data": null
}
```

Success Response Fields:

Field	Description
data	Always null for this endpoint

Error Response JSON Sample:

```
{
  "success": false,
  "httpStatus": "NOT_FOUND",
  "message": "User not found",
  "action_time": "2025-12-11T10:30:45",
  "data": "User not found"
}
```

Standard Error Types:

Application-Level Exceptions (400-499)

- `401 UNAUTHORIZED`: Authentication issues (empty, invalid, expired, or malformed tokens)
- `404 NOT_FOUND`: User to unmute not found
- `500 INTERNAL_SERVER_ERROR`: Unexpected server errors

Error Response Examples:

Not Found (404):

```
{
  "success": false,
  "httpStatus": "NOT_FOUND",
  "message": "User not found",
  "action_time": "2025-12-11T10:30:45",
  "data": "User not found"
}
```

---

5. Get Blocked Users

Purpose: Retrieve the complete list of users blocked by the authenticated user

Endpoint: `GET` `{base_url}/e-social/privacy-control/blocked`

Access Level: □□ Protected (Requires Bearer Token Authentication)

Authentication: Bearer Token

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer token for authentication (format: `Bearer <token>`)
Content-Type	string	Yes	Must be `application/json`

Success Response JSON Sample:


```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Blocked users retrieved successfully",
  "action_time": "2025-12-11T10:30:45",
  "data": [
    {
      "id": "880e8400-e29b-41d4-a716-446655440033",
      "user": {
        "id": "777e7777-e77b-77d7-a777-777777777777",
        "userName": "spam_account",
        "firstName": "Spam",
        "lastName": "Account",
        "profilePictureUrls": ["https://cdn.nexgate.com/profiles/spam_account.jpg"],
        "isVerified": false
      },
      "createdAt": "2025-12-11T10:30:45"
    },
    {
      "id": "991e8400-e29b-41d4-a716-446655440044",
      "user": {
        "id": "999e9999-e99b-99d9-a999-999999999999",
        "userName": "troll_user",
        "firstName": "Troll",
        "lastName": "User",
        "profilePictureUrls": ["https://cdn.nexgate.com/profiles/troll_user.jpg"],
        "isVerified": false
      },
      "createdAt": "2025-12-10T15:20:00"
    }
  ]
}
```

Success Response Fields:

Field	Description
data	Array of blocked user objects, sorted by createdAt in descending order (newest first)
data[].id	Unique identifier for the block record (UUID format)
data[].user	Object containing information about the blocked user
data[].user.id	UUID of the blocked user
data[].user.userName	Username of the blocked user
data[].user.firstName	First name of the blocked user
data[].user.lastName	Last name of the blocked user
data[].user.profilePictureUrls	Array of profile picture URLs for the blocked user
data[].user.isVerified	Boolean indicating if the blocked account is verified
data[].createdAt	Timestamp when the block was created (ISO 8601 format)

Error Response JSON Sample:

```
{
  "success": false,
  "httpStatus": "UNAUTHORIZED",
  "message": "Token has expired",
  "action_time": "2025-12-11T10:30:45",
  "data": "Token has expired"
}
```

Standard Error Types:

Application-Level Exceptions (400-499)

- `401 UNAUTHORIZED`: Authentication issues (empty, invalid, expired, or malformed tokens)
- `500 INTERNAL_SERVER_ERROR`: Unexpected server errors

Error Response Examples:

Unauthorized - Token Issues (401):

```
{
  "success": false,
  "httpStatus": "UNAUTHORIZED",
  "message": "Token has expired",
```

```
  "action_time": "2025-12-11T10:30:45",
  "data": "Token has expired"
}
```

---

6. Get Muted Users

Purpose: Retrieve the complete list of users muted by the authenticated user

Endpoint: `GET` `{base_url}/e-social/privacy-control/muted`

Access Level: □□ Protected (Requires Bearer Token Authentication)

Authentication: Bearer Token

Request Headers:

Header	Type	Required	Description
Authorization	string	Yes	Bearer token for authentication (format: `Bearer <token>`)
Content-Type	string	Yes	Must be `application/json`

Success Response JSON Sample:

```
{
  "success": true,
  "httpStatus": "OK",
  "message": "Muted users retrieved successfully",
  "action_time": "2025-12-11T10:30:45",
  "data": [
    {
      "id": "990e8400-e29b-41d4-a716-446655440044",
      "user": {
        "id": "888e8888-e88b-88d8-a888-888888888888",
        "userName": "noisy_user",
        "firstName": "Noisy",
        "lastName": "User",
        "profilePictureUrls": ["https://cdn.nexgate.com/profiles/noisy_user.jpg"],
        "isVerified": true
      },
```

```
      "createdAt": "2025-12-11T10:30:45"
    },
    {
      "id": "101e8400-e29b-41d4-a716-446655440055",
      "user": {
        "id": "100e1000-e10b-10d1-a100-100000000000",
        "userName": "oversharer",
        "firstName": "Over",
        "lastName": "Sharer",
        "profilePictureUrls": ["https://cdn.nexgate.com/profiles/oversharer.jpg"],
        "isVerified": false
      },
      "createdAt": "2025-12-09T08:15:30"
    }
  ]
}
```


Success Response Fields:

Field	Description
data	Array of muted user objects, sorted by createdAt in descending order (newest first)
data[].id	Unique identifier for the mute record (UUID format)
data[].user	Object containing information about the muted user
data[].user.id	UUID of the muted user
data[].user.userName	Username of the muted user
data[].user.firstName	First name of the muted user
data[].user.lastName	Last name of the muted user
data[].user.profilePictureUrls	Array of profile picture URLs for the muted user
data[].user.isVerified	Boolean indicating if the muted account is verified
data[].createdAt	Timestamp when the mute was created (ISO 8601 format)

Error Response JSON Sample:


```
{
  "success": false,
  "httpStatus": "UNAUTHORIZED",
  "message": "Token has expired",
  "action_time": "2025-12-11T10:30:45",
```

```
  "data": "Token has expired"
}
```

Standard Error Types:

Application-Level Exceptions (400-499)

- `401 UNAUTHORIZED`: Authentication issues (empty, invalid, expired, or malformed tokens)
- `500 INTERNAL_SERVER_ERROR`: Unexpected server errors

Error Response Examples:

Unauthorized - Token Issues (401):

```
{
  "success": false,
  "httpStatus": "UNAUTHORIZED",
  "message": "Token has expired",
  "action_time": "2025-12-11T10:30:45",
  "data": "Token has expired"
}
```

Quick Reference Guide

Common HTTP Status Codes

- `200 OK`: Successful GET/POST/DELETE request
- `400 Bad Request`: Invalid request data or business rule violation
- `401 Unauthorized`: Authentication required/failed
- `404 Not Found`: Resource not found
- `500 Internal Server Error`: Server error

Authentication

- **Bearer Token**: Include `Authorization: Bearer your_token` in headers

Data Format Standards

- **Dates**: Use ISO 8601 format (2025-12-11T14:30:00Z)

- **IDs**: UUID format (e.g., 550e8400-e29b-41d4-a716-446655440000)

Key Differences Between Block and Mute

Feature	Block	Mute
Removes follow relationships	Yes	No
Hides content	Yes	Yes
User knows they're blocked/muted	Possibly (can't follow)	No
Can send messages	No	Yes (if messaging exists)
Can view profile	Limited	Yes

Revision #1
Created 11 December 2025 09:07:51 by Admin Qbit
Updated 11 December 2025 09:08:18 by Admin Qbit