Setup Script for Project

Overview

This script automates the setup process for the server environment of the project. It checks for necessary dependencies, sets up the environment configuration, installs required packages, and prepares the database.

Prerequisites

Before running the script, ensure you have the following installed on your system:

- **Node.js**: You can download it from nodejs.org.
- **npm**: npm comes bundled with Node.js, but you can also install it separately if needed.

Usage

To use the setup script, follow these steps:

1. **Run the Setup Script**:

   Execute the script using the following command:

   ```
   ./setup.sh

   # or with database reset

   RESET_DB=true ./setup.sh
   ```

   This script performs the following actions:

   - Checks for the existence of Node.js and npm, and prompts for installation if they are missing.
   - Checks for Prisma installation and installs it if not present.
   - Resets the database if the `RESET_DB` environment variable is set to `true`.
   - Installs the required npm packages.
   - Runs Prisma migrations to ensure the database schema is up to date.
   - Creates an admin user using the `createAdmin.js` script.

2. **Start the Server**:

   After running the setup script, you can start your server with:

   ```
   ./run.sh
   ```

Test Admin Account

- **Email**: admin@example.com
- **Password**: LeFlop23

API Documentation

The API documentation is created with the OpenAPI schema with the version 3.0.0, a PDF version is attached below and the interactive version can be found at http://localhost:3000/api-docs

Running the Server on CS Teaching Labs

- Start the server on the CS Teaching Labs
- Run the following command on your local machine

```
ssh -L 3000:localhost:3000 username@teach.cs.utoronto.ca
```

- Access the server at http://localhost:3000

Database Schema

1. **User**
    - **Fields**:
        - `id` (Int, Primary Key, Auto-increment): Unique identifier for each user.
        - `firstName` (String): First name of the user.
        - `lastName` (String): Last name of the user.
        - `email` (String, Unique): User's email address.
        - `password` (String): Password for user authentication.
        - `role` (String, Default: "user"): Role assigned to the user (e.g., "user" or "admin").
        - `avatar` (String, Nullable): URL to the user's avatar image.
        - `phoneNumber` (String, Nullable): Contact phone number of the user.
        - `createdAt` (DateTime, Default: Current timestamp): When the user account was created.
        - `updatedAt` (DateTime, Default: Auto-update on change): Timestamp of the last update.
    - **Relationships**:
        - **One-to-Many** with `Template`, `BlogPost`, `Comment`, `Report`, `Rating`, and `CodeExecution`.

---

2. **Template**
    - **Fields**:
        - `id` (Int, Primary Key, Auto-increment): Unique identifier for each template.
        - `title` (String): Title of the template.
        - `description` (String): Detailed description of the template.
        - `code` (String): The actual code in the template.

- `stdin` (String, Nullable): Standard input for code execution, if applicable.
- `language` (String): Programming language of the template.
- `isForked` (Boolean): Flag indicating if the template is a fork of another.
- `forkedFromId` (Int, Nullable): ID of the parent template if this is a fork.
- `userId` (Int): ID of the user who created the template.
- `createdAt` (DateTime, Default: Current timestamp): Creation time of the template.
- `updatedAt` (DateTime, Default: Auto-update on change): Timestamp of the last update.

- **Relationships**:

  - **Many-to-One** with `User` (template creator).
  - **Self-Referencing One-to-Many** for forking relationships.
  - **Many-to-Many** with `TemplateTag` (tags on templates).

---

3. **TemplateTag**

   - **Fields**:

     - `id` (Int, Primary Key, Auto-increment): Unique identifier for each tag.
     - `tag` (String, Unique): Tag name associated with templates.

   - **Relationships**:

     - **Many-to-Many** with `Template`.

---

4. **BlogPost**

   - **Fields**:

     - `id` (Int, Primary Key, Auto-increment): Unique identifier for each blog post.
     - `title` (String): Title of the blog post.
     - `description` (String): Content or description of the blog post.
     - `userId` (Int): ID of the user who created the blog post.
     - `isVisible` (Boolean, Default: true): Flag indicating if the blog post is publicly visible.
     - `createdAt` (DateTime, Default: Current timestamp): Creation time of the blog post.
     - `updatedAt` (DateTime, Default: Auto-update on change): Timestamp of the last update.

   - **Relationships**:

     - **Many-to-One** with `User`.
     - **Many-to-Many** with `BlogPostTag`.
     - **One-to-Many** with `Comment`, `Report`, and `Rating`.

---

5. **BlogPostTag**

   - **Fields**:

     - `id` (Int, Primary Key, Auto-increment): Unique identifier for each blog post tag.
     - `tag` (String, Unique): Tag name for categorizing blog posts.

- **Relationships**:
  - **Many-to-Many** with `BlogPost`.

---

6. **Comment**
   - **Fields**:
     - `id` (Int, Primary Key, Auto-increment): Unique identifier for each comment.
     - `content` (String): Text content of the comment.
     - `userId` (Int): ID of the user who made the comment.
     - `blogPostId` (Int): ID of the blog post this comment is associated with.
     - `parentId` (Int, Nullable): ID of the parent comment for nested replies.
     - `isVisible` (Boolean, Default: true): Visibility flag for the comment.
     - `createdAt` (DateTime, Default: Current timestamp): Creation time of the comment.
     - `updatedAt` (DateTime, Default: Auto-update on change): Timestamp of the last update.
   - **Relationships**:
     - **Many-to-One** with `User` and `BlogPost`.
     - **Self-Referencing One-to-Many** for replies within comments.
     - **One-to-Many** with `Rating` and `Report`.

---

7. **Report**
   - **Fields**:
     - `id` (Int, Primary Key, Auto-increment): Unique identifier for each report.
     - `explanation` (String): Explanation or reason for reporting the content.
     - `userId` (Int): ID of the user who submitted the report.
     - `blogPostId` (Int, Nullable): ID of the blog post being reported (if applicable).
     - `commentId` (Int, Nullable): ID of the comment being reported (if applicable).
     - `status` (String, Default: "open"): Status of the report.
   - **Relationships**:
     - **Many-to-One** with `User`, `BlogPost`, and `Comment`.

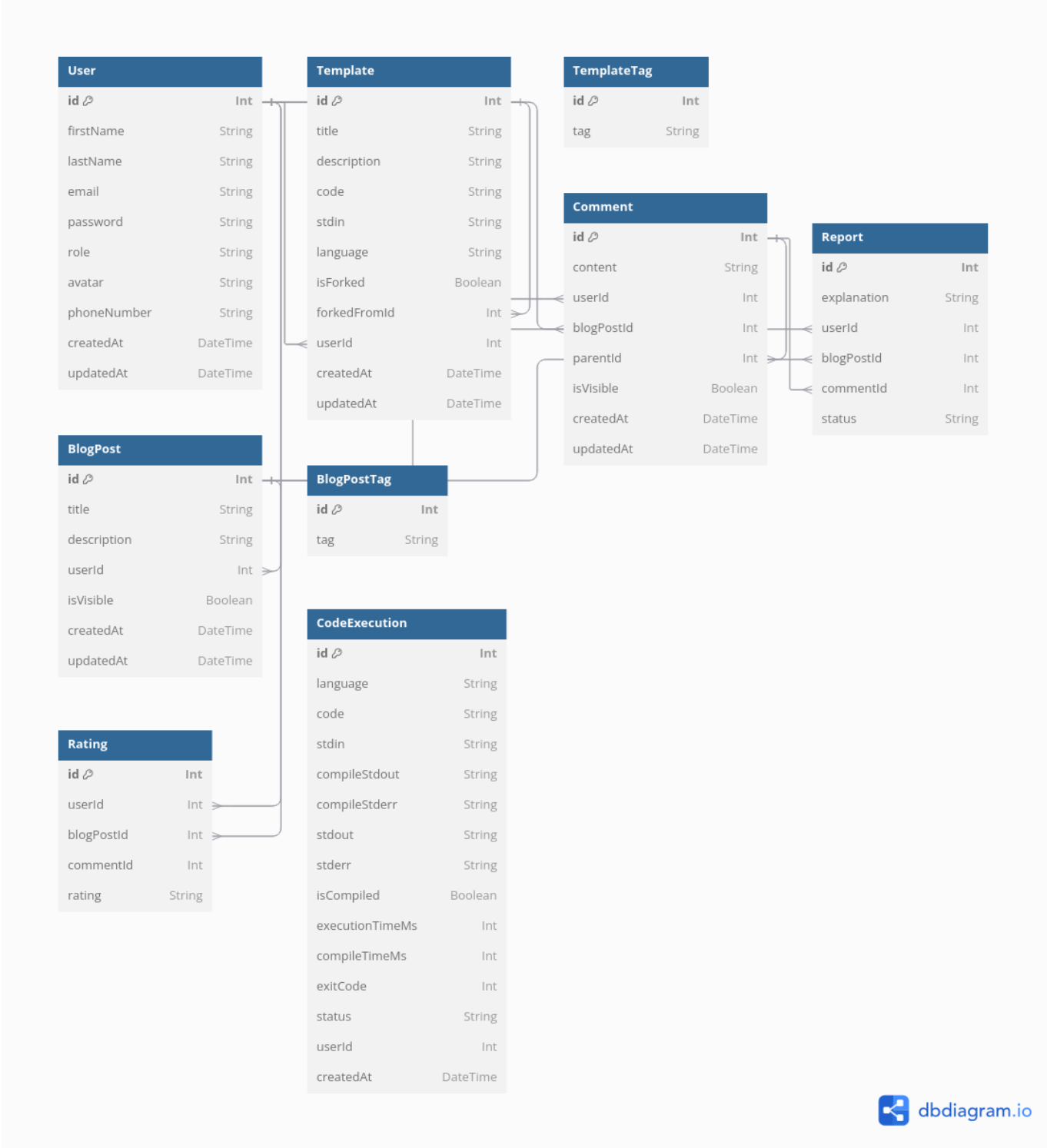
---

8. **Rating**
   - **Fields**:
     - `id` (Int, Primary Key, Auto-increment): Unique identifier for each rating.
     - `userId` (Int): ID of the user who gave the rating.
     - `blogPostId` (Int, Nullable): ID of the rated blog post (if applicable).
     - `commentId` (Int, Nullable): ID of the rated comment (if applicable).
     - `rating` (String): Type of rating, such as "+1" or "-1".
   - **Relationships**:

- **Many-to-One** with `User`, `BlogPost`, and `Comment`.

---

9. **CodeExecution**
    - **Fields**:
        - `id` (Int, Primary Key, Auto-increment): Unique identifier for each code execution.
        - `language` (String): Programming language of the code.
        - `code` (String): Code being executed.
        - `stdin` (String, Nullable): Input to the code, if applicable.
        - `isCompiled` (Boolean): Flag indicating if the code is compiled.
        - `compileStdout` (String, Nullable): Output of the compilation process.
        - `compileStderr` (String, Nullable): Error output from compilation.
        - `stdout` (String, Nullable): Standard output from code execution.
        - `stderr` (String, Nullable): Error output from code execution.
        - `executionTimeMs` (Int, Nullable): Time taken for execution in milliseconds.
        - `compileTimeMs` (Int, Nullable): Time taken for compilation in milliseconds.
        - `exitCode` (Int, Nullable): Exit code from the code execution.
        - `status` (String, Default: "pending"): Current status of the code execution.
        - `userId` (Int, Nullable): ID of the user who executed the code.
        - `createdAt` (DateTime, Default: Current timestamp): Creation time of the code execution.
    - **Relationships**:
        - **Many-to-One** with `User`.


User
id Int
firstName String
lastName String
email String
password String
role String
avatar String
phoneNumber String
createdAt DateTime
updatedAt DateTime
Template
id Int
title String
description String
code String
stdin String
language String
isForked Boolean
forkedFromId Int
userId Int
createdAt DateTime
updatedAt DateTime
TemplateTag
id Int
tag String
Comment
id Int
content String
userId Int
blogPostId Int
parentId Int
isVisible Boolean
createdAt DateTime
updatedAt DateTime
Report
id Int
explanation String
userId Int
blogPostId Int
commentId Int
status String
BlogPost
id Int
title String
description String
userId Int
isVisible Boolean
createdAt DateTime
updatedAt DateTime
BlogPostTag
id Int
tag String
CodeExecution
id Int
language String
code String
stdin String
compileStdout String
compileStderr String
stdout String
stderr String
isCompiled Boolean
executionTimeMs Int
compileTimeMs Int
exitCode Int
status String
userId Int
createdAt DateTime
Rating
id Int
userId Int
blogPostId Int
commentId Int
rating String
dbdiagram.io

Scriptorium API

No description provided (generated by Swagger Codegen https://github.com/swagger-api/swagger-codegen)
More information: https://helloreverb.com
Contact Info: hello@helloreverb.com
Version: 1.0.0

http://apache.org/licenses/LICENSE-2.0.html

Access

1.

Methods

[Jump to Models]

Table of Contents

- POST /api/v1/templates
- GET /api/v1/templates/{templateId}/blog-posts
- DELETE /api/v1/templates/{templateId}
- POST /api/v1/templates/{templateId}/execute
- POST /api/v1/templates/{templateId}/fork
- GET /api/v1/templates/{templateId}
- PATCH /api/v1/templates/{templateId}

UserProfile

- PUT /api/v1/me/avatar
- GET /api/v1/me
- PATCH /api/v1/me
- GET /api/v1/me/templates

Admin

GET /api/v1/admin/blog-posts/{blogPostId}/comments

Up

Retrieve comments for a blog post for admin, which enables sort by reports.
(**apiV1AdminBlogPostsBlogPostIdCommentsGet**)

Fetches a paginated list of comments for a specific blog post with optional sorting.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to retrieve comments for.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

sortBy (optional)
Query Parameter — Field to sort by (e.g., 'createdAt').

sortDirection (optional)
Query Parameter — Sort direction, either ascending (asc) or descending (desc).

Return type
inline_response_200

Example data
Content-Type: application/json


```
{
  "data" : [ {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }, {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  } ],
  "_metadata" : {
    "pageCount" : 5,
```

```
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```

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
Comments retrieved successfully. inline_response_200
400
Bad request, possibly due to invalid query parameters.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
500
Internal server error.

GET /api/v1/admin/blog-posts

Up

Retrieve a list of blog posts for admin, which enables sort by reports. (**apiV1AdminBlogPostsGet**)

Fetches a paginated list of blog posts with optional filters.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

title (optional)
Query Parameter — Filter blog posts by title.

description (optional)
Query Parameter — Filter blog posts by description.

templateContent (optional)
Query Parameter — Filter blog posts by template content.

sortBy (optional)
Query Parameter — Field to sort by (e.g., 'createdAt').

sortDirection (optional)
Query Parameter — Sort direction, either ascending (asc) or descending (desc).

tags (optional)
Query Parameter — Comma-separated list of tags to filter by.

Return type
inline_response_200_1

Example data
Content-Type: application/json

```
{
  "data" : [ {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
    }, {
```

```
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  }, {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
    }, {
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  } ],
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```


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
Blog posts retrieved successfully. inline_response_200_1
400
Bad request, possibly due to invalid query parameters.
401
Unauthorized, possibly due to missing or invalid authentication token.
500
Internal server error.

GET /api/v1/admin/reports

Up

Retrieve a list of reports (**apiV1AdminReportsGet**)

Allows an admin to retrieve a paginated list of reports with optional filters by status.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

status (optional)
Query Parameter — Filter reports by their status.

Return type
inline_response_200_3

Example data
Content-Type: application/json


```
{
  "data" : [ {
    "createdAt" : "2024-10-31T12:00:00Z",
    "id" : 101,
    "explanation" : "This post contains inappropriate content.",
    "status" : "pending",
    "updatedAt" : "2024-10-31T14:00:00Z"
  }, {
    "createdAt" : "2024-10-31T12:00:00Z",
    "id" : 101,
    "explanation" : "This post contains inappropriate content.",
    "status" : "pending",
    "updatedAt" : "2024-10-31T14:00:00Z"
  } ],
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```


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
Reports retrieved successfully. inline_response_200_3
400
Bad request, possibly due to invalid query parameters.
401
Unauthorized, possibly due to missing or invalid authentication token.
403
Forbidden, user lacks the necessary permissions to access this resource.
500
Internal server error.

POST /api/v1/admin/reports/{reportId}/approve

Up

Approve a report (**apiV1AdminReportsReportIdApprovePost**)

Allows an admin to approve a specific report which hides the post or comment.

Path parameters

reportId (required)
Path Parameter — The unique ID of the report to approve.

Return type
inline_response_200_2

Example data
Content-Type: application/json

```
{
  "message" : "Report successfully approved."
}
```

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
Report approved successfully. inline_response_200_2
400
Bad request, possibly due to invalid report ID.
401
Unauthorized, possibly due to missing or invalid authentication token.
403
Forbidden, user lacks the necessary permissions to approve reports.
404
Report not found.
500
Internal server error.

Auth

`POST /api/v1/auth/login`

Up

Log in a user (**apiV1AuthLoginPost**)

Authenticates a user by validating the credentials and returns access and refresh tokens.

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

- `application/json`

Request body

body auth_login_body (required)
Body Parameter —

Return type
inline_response_200_4

Example data
Content-Type: application/json

```
{
  "data" : {
    "accessToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
    "refreshToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
  }
}
```

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 logged in successfully, returning tokens. inline_response_200_4
400
Bad request, possibly due to invalid input format.
401

Unauthorized access due to incorrect credentials.
422
Unprocessable entity, possibly due to invalid or missing input values.
500
Internal server error.

POST /api/v1/auth/refresh

Up

Refresh access token (**apiV1AuthRefreshPost**)

Uses a valid refresh token to issue a new access token.

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

- `application/json`

Request body

body auth_refresh_body (required)
Body Parameter —

Return type
inline_response_200_5

Example data
Content-Type: application/json

```
{
  "data" : {
    "accessToken" : "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
  }
}
```

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
Access token refreshed successfully. inline_response_200_5
400
Bad request, possibly due to invalid input format.
401
Unauthorized access due to incorrect credentials.
422
Unprocessable entity, possibly due to invalid or missing input values.
500
Internal server error.

POST /api/v1/auth/register

Up

Register a new user (**apiV1AuthRegisterPost**)

Creates a new user account with the provided registration details.

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

- `application/json`

Request body

body auth_register_body (required)

Body Parameter —

Return type
inline_response_201

Example data
Content-Type: application/json

```
{
  "data" : {
    "firstName" : "John",
    "lastName" : "Doe",
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "role" : "user",
    "phoneNumber" : "+1234567890",
    "id" : 1,
    "avatar" : "https://example.com/avatar.jpg",
    "email" : "user@example.com",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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
201
User registered successfully. inline_response_201
400
Bad request, possibly due to invalid input format.
422
Unprocessable entity, possibly due to invalid or missing input values (e.g., non-matching passwords).
500
Internal server error.

POST /api/v1/auth/reset-password

Up

Reset the user's password (**apiV1AuthResetPasswordPost**)

Allows an authenticated user to reset their password by providing the current password and a new password.

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

- `application/json`

Request body

body auth_resetpassword_body (required)
Body Parameter —

Return type
inline_response_200_6

Example data
Content-Type: application/json

```
{
  "message" : "Password reset successfully"
}
```

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
Password reset successfully. inline_response_200_6
400
Bad request, possibly due to invalid input format.
401
Unauthorized, possibly due to an invalid or missing authentication token.
422
Unprocessable entity, possibly due to incorrect current password or password policy violation.
500
Internal server error.

BlogPostComments

`DELETE /api/v1/blog-posts/{blogPostId}/comments/{commentId}`

Up

Delete a comment on a blog post (**apiV1BlogPostsBlogPostIdCommentsCommentIdDelete**)

Allows an authenticated user to delete a specific comment they made on a blog post.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post.

commentId (required)
Path Parameter — The unique ID of the comment to delete.

Return type
inline_response_200_10

Example data
Content-Type: application/json

```
{
  "message" : "Comment deleted successfully"
}
```

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
Comment deleted successfully. inline_response_200_10
400
Bad request, possibly due to an invalid comment ID.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Comment not found.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

`GET /api/v1/blog-posts/{blogPostId}/comments/{commentId}`

Up

Retrieve a specific comment (**apiV1BlogPostsBlogPostIdCommentsCommentIdGet**)

Fetches a specific comment by its ID, with optional authentication.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to retrieve comments for.

commentId (required)
Path Parameter — The unique ID of the comment to retrieve.

Return type
inline_response_200_9

Example data
Content-Type: application/json

```
{
  "data" : {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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
Comment retrieved successfully. inline_response_200_9
400
Bad request, possibly due to an invalid comment ID.
404
Comment not found.
500
Internal server error.

POST /api/v1/blog-posts/{blogPostId}/comments/{commentId}/rate

Up

Rate a comment on a blog post (**apiV1BlogPostsBlogPostIdCommentsCommentIdRatePost**)

Allows an authenticated user to rate a specific comment on a blog post.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post.

commentId (required)
Path Parameter — The unique ID of the comment to rate.

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

- `application/json`

Request body

body commentId_rate_body (required)
Body Parameter —

Return type
inline_response_200_7

Example data

Content-Type: application/json

```
{
  "message" : "Comment rated successfully"
}
```

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
Comment rated successfully. inline_response_200_7
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Comment not found.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

GET /api/v1/blog-posts/{blogPostId}/comments/{commentId}/replies

Up

Retrieve replies to a specific comment (**apiV1BlogPostsBlogPostIdCommentsCommentIdRepliesGet**)

Fetches a paginated list of replies for a specific comment on a blog post, with optional authentication to view hidden comments.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post the comment belongs to.

commentId (required)
Path Parameter — The unique ID of the comment to retrieve replies for.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

Return type
inline_response_200_8

Example data
Content-Type: application/json

```
{
  "data" : [ {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }, {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
```

```
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  } ],
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```


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
Replies retrieved successfully. inline_response_200_8
400
Bad request, possibly due to invalid query parameters.
404
Comment not found or has no replies.
500
Internal server error.

POST /api/v1/blog-posts/{blogPostId}/comments/{commentId}/report

Up

Report a comment (**apiV1BlogPostsBlogPostIdCommentsCommentIdReportPost**)

Allows an authenticated user to report a specific blog post with an explanation.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post.

commentId (required)
Path Parameter — The unique ID of the comment to report.

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

- `application/json`

Request body

body commentId_report_body (required)
Body Parameter —

Return type
inline_response_201_1

Example data
Content-Type: application/json


```
{
  "createdAt" : "2024-10-31T12:00:00Z",
  "reportId" : 101,
  "commentId" : 456,
  "blogPostId" : 456,
  "explanation" : "This comment contains inappropriate content.",
  "userId" : 123
}
```


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
201
Report submitted successfully. inline_response_201_1
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Comment not found.
409
Conflict due to duplicate data.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

GET /api/v1/blog-posts/{blogPostId}/comments

Up

Retrieve comments for a blog post (**apiV1BlogPostsBlogPostIdCommentsGet**)

Fetches a paginated list of comments for a specific blog post with optional sorting.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to retrieve comments for.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

sortBy (optional)
Query Parameter — Field to sort by (e.g., 'createdAt').

sortDirection (optional)
Query Parameter — Sort direction, either ascending (asc) or descending (desc).

Return type
inline_response_200

Example data
Content-Type: application/json


```
{
  "data" : [ {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }, {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  } ],
```

```
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```

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
Comments retrieved successfully. inline_response_200
400
Bad request, possibly due to invalid query parameters.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
500
Internal server error.

POST /api/v1/blog-posts/{blogPostId}/comments

Up

Add a comment to a blog post (**apiV1BlogPostsBlogPostIdCommentsPost**)

Allows an authenticated user to add a comment to a specified blog post, optionally replying to another comment. Note that you can only reply to a top-level comment.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to add a comment to.

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

- `application/json`

Request body

body blogPostId_comments_body (required)
Body Parameter —

Return type
inline_response_200_9

Example data
Content-Type: application/json

```
{
  "data" : {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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
201
Comment added successfully. inline_response_200_9
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

BlogPosts

DELETE /api/v1/blog-posts/{blogPostId}

Up

Delete a blog post (**apiV1BlogPostsBlogPostIdDelete**)

Allows an authenticated user to delete a specific blog post they own.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to delete.

Return type
inline_response_200_13

Example data
Content-Type: application/json

```
{
  "message" : "Blog post deleted successfully"
}
```

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
Blog post deleted successfully. inline_response_200_13
400
Bad request, possibly due to an invalid blog post ID.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
500
Internal server error.

GET /api/v1/blog-posts/{blogPostId}

Up

Retrieve a specific blog post (**apiV1BlogPostsBlogPostIdGet**)

Fetches a specific blog post by its ID.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to retrieve.

Return type
inline_response_200_12

Example data
Content-Type: application/json


```
{
  "data" : {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
    }, {
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  }
}
```


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
Blog post retrieved successfully. inline_response_200_12
400
Bad request, possibly due to an invalid blog post ID.
404
Blog post not found.
500
Internal server error.

`PATCH /api/v1/blog-posts/{blogPostId}`

Up

Update a blog post (**apiV1BlogPostsBlogPostIdPatch**)

Allows an authenticated user to update specified fields of a blog post they own.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to update.

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

- `application/json`

Request body

body **blogposts_blogPostId_body** **(required)**
Body Parameter —

Return type
Blog

Example data
Content-Type: application/json

```
{
  "netRating" : 5,
  "createdAt" : "2000-01-23T04:56:07.000+00:00",
  "description" : "Blog Content",
  "links" : [ {
    "id" : 0
  }, {
    "id" : 0
  } ],
  "id" : 1,
  "isVisible" : true,
  "title" : "Blog Title",
  "userId" : 1,
  "updatedAt" : "2000-01-23T04:56:07.000+00:00",
  "tags" : [ {
    "id" : 1,
    "tag" : "tutorial"
  } ]
}
```

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
Blog post updated successfully. Blog
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
500
Internal server error.

`POST /api/v1/blog-posts/{blogPostId}/rate`

Up

Rate a blog post (**apiV1BlogPostsBlogPostIdRatePost**)

Allows an authenticated user to rate a specific blog post. New ratings will overwrite existing ratings.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to rate.

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

- `application/json`

Request body

body **blogPostId_rate_body** **(required)**

Body Parameter —

Return type
inline_response_200_11

Example data
Content-Type: application/json

```
{
  "netRating" : 5,
  "message" : "Blog post rated successfully"
}
```

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
Blog post rated successfully. inline_response_200_11
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

POST /api/v1/blog-posts/{blogPostId}/report

Up

Report a blog post (**apiV1BlogPostsBlogPostIdReportPost**)

Allows an authenticated user to report a specific blog post with an explanation.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to report.

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

- `application/json`

Request body

body blogPostId_report_body (required)
Body Parameter —

Return type
inline_response_201_2

Example data
Content-Type: application/json

```
{
  "createdAt" : "2024-10-31T12:00:00Z",
  "reportId" : 101,
  "blogPostId" : 456,
  "explanation" : "This post contains inappropriate content.",
```

```
  "userId" : 123
}
```

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
201
Report submitted successfully. inline_response_201_2
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Blog post not found.
409
Conflict due to duplicate data.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

GET /api/v1/blog-posts

Up

Retrieve a list of blog posts (**apiV1BlogPostsGet**)

Fetches a paginated list of blog posts with optional filters and sorting.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

title (optional)
Query Parameter — Filter blog posts by title.

description (optional)
Query Parameter — Filter blog posts by description.

templateContent (optional)
Query Parameter — Filter blog posts by template content.

sortBy (optional)
Query Parameter — Field to sort by (e.g., 'createdAt'). Options are 'createdAt', 'rating', and 'controversial', 'controversial' is determined by the total number of ratings.

sortDirection (optional)
Query Parameter — Sort direction, either ascending (asc) or descending (desc).

tags (optional)
Query Parameter — Comma-separated list of tags to filter by.

Return type
inline_response_200_1

Example data
Content-Type: application/json

```
{
  "data" : [ {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
```

```
    }, {
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  }, {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
    }, {
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  } ],
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```


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
Blog posts retrieved successfully. inline_response_200_1
400
Bad request, possibly due to invalid query parameters.
500
Internal server error.

POST /api/v1/blog-posts

Up

Create a new blog post (**apiV1BlogPostsPost**)

Allows an authenticated user to create a new blog post.

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

- `application/json`

Request body

body v1_blogposts_body (required)

Body Parameter —

Return type
Blog

Example data
Content-Type: application/json

```
{
  "netRating" : 5,
  "createdAt" : "2000-01-23T04:56:07.000+00:00",
  "description" : "Blog Content",
  "links" : [ {
    "id" : 0
  }, {
    "id" : 0
  } ],
  "id" : 1,
  "isVisible" : true,
  "title" : "Blog Title",
  "userId" : 1,
  "updatedAt" : "2000-01-23T04:56:07.000+00:00",
  "tags" : [ {
    "id" : 1,
    "tag" : "tutorial"
  } ]
}
```

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
201
Blog post created successfully. Blog
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

CodeExecution

Up

`POST /api/v1/execute`

Execute code (**apiV1ExecutePost**)

Execute code.

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

- `application/json`

Request body

body **v1_execute_body** **(required)**
Body Parameter —

Return type

inline_response_200_14

Example data
Content-Type: application/json


```
{
  "data" : {
    "isCompiled" : true,
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "stdout" : "Hello, World!",
    "executionTimeMs" : 50,
    "language" : "javascript",
    "stderr" : "stderr",
    "userId" : 123,
    "compileTimeMs" : 100,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "compileStdout" : "compileStdout",
    "compileStderr" : "compileStderr",
    "exitCode" : 0,
    "id" : 1,
    "status" : "completed"
  }
}
```


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
Code executed successfully. inline_response_200_14
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

Comments

`PATCH /api/v1/blog-posts/{blogPostId}/comments/{commentId}`

Up

Update a specific comment (**apiV1BlogPostsBlogPostIdCommentsCommentIdPatch**)

Allows an authenticated user to update their comment by its ID.

Path parameters

blogPostId (required)
Path Parameter — The unique ID of the blog post to update a comment on.

commentId (required)
Path Parameter — The unique ID of the comment to update.

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

- `application/json`

Request body

body comments_commentId_body (required)
Body Parameter —

Return type

inline_response_200_9

Example data
Content-Type: application/json

```
{
  "data" : {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "id" : 1,
    "blogPostId" : 1,
    "userId" : 123,
    "content" : "This is a great post! Thanks for sharing.",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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
Comment updated successfully. inline_response_200_9
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
403
Forbidden, possibly due to insufficient permissions.
404
Comment not found.
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

Templates

GET /api/v1/templates

Up

Retrieve a list of templates (**apiV1TemplatesGet**)

Fetches a paginated list of templates with optional filters.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

title (optional)
Query Parameter — Filter templates by title.

description (optional)
Query Parameter — Filter templates by description.

userId (optional)
Query Parameter — Filter templates by user ID.

tags (optional)
Query Parameter — Comma-separated list of tags to filter by.

languages (optional)
Query Parameter — Comma-separated list of programming languages to filter by.

Return type

inline_response_200_19

Example data
Content-Type: application/json


```
{
  "data" : [ {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }, {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  } ],
  "_metadata" : {
    "pageCount" : 1,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 2
  }
}
```


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
Templates retrieved successfully. inline_response_200_19
400
Bad request, possibly due to invalid query parameters.
500
Internal server error.

POST /api/v1/templates

Up

Create a new template (**apiV1TemplatesPost**)

Allows an authenticated user to create a new template with specified details.

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

- `application/json`

Request body

body **v1_templates_body** **(required)**
Body Parameter —

Return type
inline_response_200_17

Example data
Content-Type: application/json


```
{
  "data" : {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```


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
201
Template created successfully. inline_response_200_17
400
Bad request, possibly due to invalid input format.
401
Unauthorized, possibly due to missing or invalid authentication token.
422
Unprocessable entity, possibly due to invalid or missing input values.
500
Internal server error.

GET /api/v1/templates/{templateId}/blog-posts

Up

Retrieve blog posts by template ID (**apiV1TemplatesTemplateIdBlogPostsGet**)

Fetches a paginated list of blog posts associated with a specific template.

Path parameters

templateId (required)
Path Parameter — The unique ID of the template to retrieve blog posts for.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

Return type
inline_response_200_1

Example data
Content-Type: application/json


```
{
  "data" : [ {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
    }, {
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  }, {
    "netRating" : 5,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "description" : "Blog Content",
    "links" : [ {
      "id" : 0
    }, {
      "id" : 0
    } ],
    "id" : 1,
    "isVisible" : true,
    "title" : "Blog Title",
    "userId" : 1,
    "updatedAt" : "2000-01-23T04:56:07.000+00:00",
    "tags" : [ {
      "id" : 1,
      "tag" : "tutorial"
    } ]
  } ],
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
    "page" : 1,
    "totalCount" : 50
  }
}
```


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

Blog posts retrieved successfully. inline_response_200_1
400
Bad request, possibly due to invalid query parameters or template ID.
404
Template not found.
500
Internal server error.

DELETE /api/v1/templates/{templateId}

Up

Delete a template (**apiV1TemplatesTemplateIdDelete**)

Deletes a template owned by the authenticated user.

Path parameters

templateId (required)
Path Parameter — The unique ID of the template to delete.

Return type
inline_response_200_18

Example data
Content-Type: application/json

```
{
  "message" : "Template deleted successfully"
}
```

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
Template deleted successfully. inline_response_200_18
400
Bad request, possibly due to an invalid template ID.
401
Unauthorized, possibly due to missing or invalid authentication token.
404
Template not found.
500
Internal server error.

POST /api/v1/templates/{templateId}/execute

Up

Execute code. (**apiV1TemplatesTemplateIdExecutePost**)

Execute code.

Path parameters

templateId (required)
Path Parameter — The unique ID of the template to retrieve blog posts for.

Return type
inline_response_200_14

Example data
Content-Type: application/json

```
{
  "data" : {
```

```
    "isCompiled" : true,
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "stdout" : "Hello, World!",
    "executionTimeMs" : 50,
    "language" : "javascript",
    "stderr" : "stderr",
    "userId" : 123,
    "compileTimeMs" : 100,
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "compileStdout" : "compileStdout",
    "compileStderr" : "compileStderr",
    "exitCode" : 0,
    "id" : 1,
    "status" : "completed"
  }
}
```


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
Code executed successfully. inline_response_200_14
422
Unprocessable entity, possibly due to invalid input data.
500
Internal server error.

`POST /api/v1/templates/{templateId}/fork`

Up

Fork a template (**apiV1TemplatesTemplateIdForkPost**)

Creates a fork of an existing template for the authenticated user. Note that you cannot fork your own template.

Path parameters

templateId (required)
Path Parameter — The unique ID of the template to fork.

Return type
inline_response_201_3

Example data
Content-Type: application/json


```
{
  "message" : "Template forked successfully"
}
```


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
201
Template forked successfully. inline_response_201_3
400
Bad request, possibly due to an invalid template ID.
401
Unauthorized, possibly due to missing or invalid authentication token.
404

Template not found.
500
Internal server error.

GET /api/v1/templates/{templateId}

Up

Retrieve a template by ID (**apiV1TemplatesTemplateIdGet**)

Fetches a specific template by its unique ID.

Path parameters

templateId (required)
Path Parameter — The unique ID of the template to retrieve.

Return type
inline_response_200_17

Example data
Content-Type: application/json


```
{
  "data" : {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```


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
Template retrieved successfully. inline_response_200_17
400
Bad request, possibly due to an invalid template ID.
404
Template not found.
500
Internal server error.

PATCH /api/v1/templates/{templateId}

Up

Update a template (**apiV1TemplatesTemplateIdPatch**)

Updates the specified fields of a template owned by the authenticated user.

Path parameters

templateId (required)

Path Parameter — The unique ID of the template to update.

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

- `application/json`

Request body

body [templates_templateId_body](#) (required)
Body Parameter —

Return type
inline_response_200_17

Example data
Content-Type: application/json

```
{
  "data" : {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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
Template updated successfully. inline_response_200_17
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
403
Forbidden, possibly due to insufficient permissions.
404
Template not found.
500
Internal server error.

UserProfile

Up

`PUT /api/v1/me/avatar`

Upload or update user avatar (**apiV1MeAvatarPut**)

Allows an authenticated user to upload and update their avatar image.

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

- `multipart/form-data`

Form parameters

file (required)
Form Parameter — format: binary

Return type
inline_response_200_15

Example data
Content-Type: application/json

```
{
  "data" : {
    "filename" : "avatar12345.png"
  }
}
```

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
Avatar uploaded successfully. inline_response_200_15
400
Bad request, possibly due to invalid file format or missing file.
401
Unauthorized, possibly due to missing or invalid authentication token.
422
Unprocessable entity, possibly due to invalid file upload data.
500
Internal server error.

GET /api/v1/me

Up

Get user information (**apiV1MeGet**)

Retrieves the authenticated user's information by their user ID.

Return type
inline_response_201

Example data
Content-Type: application/json

```
{
  "data" : {
    "firstName" : "John",
    "lastName" : "Doe",
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "role" : "user",
    "phoneNumber" : "+1234567890",
    "id" : 1,
    "avatar" : "https://example.com/avatar.jpg",
    "email" : "user@example.com",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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 information retrieved successfully. inline_response_201
401
Unauthorized, possibly due to missing or invalid authentication token.
500
Internal server error.

PATCH /api/v1/me

Up

Update user information (**apiV1MePatch**)

Updates the authenticated user's information with the provided data.

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

- `application/json`

Request body

body v1_me_body (required)
Body Parameter —

Return type
inline_response_201

Example data
Content-Type: application/json

```
{
  "data" : {
    "firstName" : "John",
    "lastName" : "Doe",
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "role" : "user",
    "phoneNumber" : "+1234567890",
    "id" : 1,
    "avatar" : "https://example.com/avatar.jpg",
    "email" : "user@example.com",
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }
}
```

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 information updated successfully. inline_response_201
400
Bad request, possibly due to invalid input data.
401
Unauthorized, possibly due to missing or invalid authentication token.
422
Unprocessable entity, possibly due to invalid or missing input values.
500
Internal server error.

GET /api/v1/me/templates

Up

Retrieve a list of templates created by the authenticated user (**apiV1MeTemplatesGet**)

Fetches a paginated list of templates with optional filters for the authenticated user.

Query parameters

page (optional)
Query Parameter — Page number for pagination.

limit (optional)
Query Parameter — Number of items per page.

title (optional)
Query Parameter — Filter templates by title.

description (optional)
Query Parameter — Filter templates by description.

tags (optional)
Query Parameter — Comma-separated list of tags to filter by.

languages (optional)
Query Parameter — Comma-separated list of programming languages to filter by.

Return type

inline_response_200_16

Example data

Content-Type: application/json


```
{
  "data" : [ {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  }, {
    "createdAt" : "2000-01-23T04:56:07.000+00:00",
    "stdin" : "input",
    "code" : "console.log('Hello, World!');",
    "forkedFromId" : 1,
    "isForked" : false,
    "description" : "Template Description",
    "language" : "javascript",
    "id" : 1,
    "title" : "Template Title",
    "userId" : 1,
    "tamplateTags" : [ {
      "id" : 1,
      "tag" : "python"
    } ],
    "updatedAt" : "2000-01-23T04:56:07.000+00:00"
  } ],
  "_metadata" : {
    "pageCount" : 5,
    "perPage" : 10,
```

```
    "page" : 1,
    "totalCount" : 50
  }
}
```

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
Templates retrieved successfully. inline_response_200_16
400
Bad request, possibly due to invalid query parameters.
401
Unauthorized, possibly due to missing or invalid authentication token.
500
Internal server error.

Models

[Jump to Methods]

Table of Contents

41. inline_response_200_4_data
42. inline_response_200_5
43. inline_response_200_5_data
44. inline_response_200_6
45. inline_response_200_7
46. inline_response_200_8
47. inline_response_200_8__metadata
48. inline_response_200_9
49. inline_response_200__metadata
50. inline_response_201
51. inline_response_201_1
52. inline_response_201_2
53. inline_response_201_3
54. me_avatar_body
55. templates_templateId_body
56. v1_blogposts_body
57. v1_execute_body
58. v1_me_body
59. v1_templates_body

Blog

Up

id (optional)
Integer
example: 1

title (optional)
String
example: Blog Title

description (optional)
String
example: Blog Content

userId (optional)
Integer
example: 1

isVisible (optional)
Boolean
example: true

netRating (optional)
Integer
example: 5

createdAt (optional)
Date format: date-time

updatedAt (optional)
Date format: date-time

tags (optional)
array[Blog_tags]
example: [{"id":1,"tag":"tutorial"}]

links (optional)
array[Blog_links]

BlogComment

Up

id (optional)
Integer
example: 1

userId (optional)
Integer
example: 123

blogPostId (optional)
Integer

example: 1

content (optional)
String
example: This is a great post! Thanks for sharing.

createdAt (optional)
Date format: date-time

updatedAt (optional)
Date format: date-time

Blog_links

Up

id (optional)
Integer

Blog_tags

Up

id (optional)
Integer

tag (optional)
String

CodeExecution

Up

id (optional)
Integer
example: 1

userId (optional)
Integer
example: 123

language (optional)
String
example: javascript

code (optional)
String
example: console.log('Hello, World!');

stdin (optional)
String
example: input

stdout (optional)
String
example: Hello, World!

stderr (optional)
String

compileStdout (optional)
String

compileStderr (optional)
String

compileTimeMs (optional)
Integer
example: 100

isCompiled (optional)
Boolean
example: true

executionTimeMs (optional)
Integer
example: 50

exitCode (optional)
Integer
example: 0

status (optional)
String
example: completed

createdAt (optional)
Date format: date-time

Template

[Up](#)

id (optional)
Integer
example: 1

title (optional)
String
example: Template Title

description (optional)
String
example: Template Description

code (optional)
String
example: console.log('Hello, World!');

stdin (optional)
String
example: input

isForked (optional)
Boolean
example: false

forkedFromId (optional)
Integer
example: 1

userId (optional)
Integer
example: 1

tamplateTags (optional)
array[Template_tamplateTags]
example: [{"id":1,"tag":"python"}]

language (optional)
String
example: javascript

createdAt (optional)
Date format: date-time

updatedAt (optional)
Date format: date-time

Template_tamplateTags

[Up](#)

id (optional)
Integer
example: 1

tag (optional)
String
example: python

User

[Up](#)

id (optional)
Integer
example: 1

email (optional)
String
example: user@example.com

firstName (optional)
String
example: John

lastName (optional)
String
example: Doe

role (optional)
String
example: user

avatar (optional)
String format: uri
example: https://example.com/avatar.jpg

phoneNumber (optional)
String
example: +1234567890

createdAt (optional)
Date format: date-time

updatedAt (optional)
Date format: date-time

`auth_login_body` Up

email
String The user's email address. format: email
example: user@example.com

password
String The user's password. format: password
example: Password123!

`auth_refresh_body` Up

refreshToken
String The refresh token provided during login.
example: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

`auth_register_body` Up

email
String The user's email address. format: email
example: newuser@example.com

password
String The user's password. Must be at least the minimum length required. format: password
example: SecurePassword123

confirmPassword
String Confirmation of the user's password. Must match the password field. format: password
example: SecurePassword123

firstName
String The user's first name.
example: John

lastName
String The user's last name.

example: Doe

phoneNumber (optional)
String Optional phone number of the user.
example: +1234567890

auth_resetpassword_body

Up

currentPassword
String The user's current password. format: password
example: OldPassword123

newPassword
String The user's new password. Must meet the required minimum length and complexity. format: password
example: NewSecurePassword456

blogPostId_comments_body

Up

content
String The content of the comment.
example: This is a great post! Thanks for sharing.

parentId (optional)
Integer The ID of the parent comment.

blogPostId_rate_body

Up

rating
String Rating given to the blog post (e.g., +1 for upvote, -1 for downvote).
Enum:
+1
-1
example: +1

blogPostId_report_body

Up

explanation
String Explanation of why the blog post is being reported.
example: This post contains inappropriate content.

blogposts_blogPostId_body

Up

title (optional)
String The updated title of the blog post.
example: Updated Blog Title

description (optional)
String The updated content of the blog post.
example: This blog post provides updated content on templates.

tags (optional)
array[String] Updated tags associated with the blog post.
example: ["tutorial","updated"]

links (optional)
array[BigDecimal]

commentId_rate_body

Up

rating
String Rating given to the blog post (e.g., +1 for upvote, -1 for downvote).
Enum:
+1

-1
example: +1

commentId_report_body [Up](#)

explanation
String Explanation of why the comment is being reported.
example: This comment contains inappropriate content.

comments_commentId_body [Up](#)

content
String The updated content of the comment.
example: This is an updated comment.

inline_response_200 [Up](#)

_metadata (optional)
inline_response_200__metadata

data (optional)
array[BlogComment]

inline_response_200_1 [Up](#)

_metadata (optional)
inline_response_200_1__metadata

data (optional)
array[Blog]

inline_response_200_10 [Up](#)

message (optional)
String
example: Comment deleted successfully

inline_response_200_11 [Up](#)

message (optional)
String
example: Blog post rated successfully

netRating (optional)
Integer Updated net rating of the blog post.
example: 5

inline_response_200_12 [Up](#)

data (optional)
Blog

inline_response_200_13 [Up](#)

message (optional)
String
example: Blog post deleted successfully

inline_response_200_14 [Up](#)

data (optional)
CodeExecution

inline_response_200_15 [Up]

data (optional)
inline_response_200_15_data

inline_response_200_15_data [Up]

filename (optional)
String The filename of the uploaded avatar.
example: avatar12345.png

inline_response_200_16 [Up]

_metadata (optional)
inline_response_200_16__metadata

data (optional)
array[Template]

inline_response_200_16__metadata [Up]

page (optional)
Integer
example: 1

perPage (optional)
Integer
example: 10

pageCount (optional)
Integer Total number of pages.
example: 5

totalCount (optional)
Integer Total number of templates.
example: 50

inline_response_200_17 [Up]

data (optional)
Template

inline_response_200_18 [Up]

message (optional)
String
example: Template deleted successfully

inline_response_200_19 [Up]

_metadata (optional)
inline_response_200_19__metadata

data (optional)
array[Template]

inline_response_200_19__metadata [Up]

page (optional)
Integer
example: 1

perPage (optional)
Integer
example: 10

pageCount (optional)
Integer Total number of pages.
example: 1

totalCount (optional)
Integer Total number of templates.
example: 2

inline_response_200_1__metadata

Up

page (optional)
Integer
example: 1

perPage (optional)
Integer
example: 10

pageCount (optional)
Integer Total number of pages.
example: 5

totalCount (optional)
Integer Total number of blog posts.
example: 50

inline_response_200_2

Up

message (optional)
String
example: Report successfully approved.

inline_response_200_3

Up

_metadata (optional)
inline_response_200_3__metadata

data (optional)
array[inline_response_200_3_data]

inline_response_200_3__metadata

Up

page (optional)
Integer
example: 1

perPage (optional)
Integer
example: 10

pageCount (optional)
Integer Total number of pages.
example: 5

totalCount (optional)
Integer Total number of reports.
example: 50

inline_response_200_3_data

Up

id (optional)
Integer Unique identifier for the report.
example: 101

status (optional)
String Status of the report.
example: pending

createdAt (optional)
Date Date and time when the report was created. format: date-time
example: 2024-10-31T12:00Z

updatedAt (optional)
Date Date and time when the report was last updated. format: date-time
example: 2024-10-31T14:00Z

explanation (optional)
String Explanation provided by the user for the report.
example: This post contains inappropriate content.

inline_response_200_4 Up

data (optional)
inline_response_200_4_data

inline_response_200_4_data Up

accessToken (optional)
String JWT access token.
example: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

refreshToken (optional)
String JWT refresh token.
example: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

inline_response_200_5 Up

data (optional)
inline_response_200_5_data

inline_response_200_5_data Up

accessToken (optional)
String JWT access token.
example: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...

inline_response_200_6 Up

message (optional)
String
example: Password reset successfully

inline_response_200_7 Up

message (optional)
String
example: Comment rated successfully

inline_response_200_8 Up

_metadata (optional)

inline_response_200_8__metadata

data (optional)
array[BlogComment]

inline_response_200_8__metadata

Up

page (optional)
Integer
example: 1

perPage (optional)
Integer
example: 10

pageCount (optional)
Integer Total number of pages.
example: 5

totalCount (optional)
Integer Total number of replies.
example: 50

inline_response_200_9

Up

data (optional)
BlogComment

inline_response_200__metadata

Up

page (optional)
Integer
example: 1

perPage (optional)
Integer
example: 10

pageCount (optional)
Integer Total number of pages.
example: 5

totalCount (optional)
Integer Total number of comments.
example: 50

inline_response_201

Up

data (optional)
User

inline_response_201_1

Up

reportId (optional)
Integer Unique identifier for the report.
example: 101

userId (optional)
Integer ID of the user who submitted the report.
example: 123

blogPostId (optional)
Integer ID of the reported blog post.
example: 456

commentId (optional)
Integer ID of the reported comment.

example: 456

explanation (optional)
String Explanation provided by the user.
example: This comment contains inappropriate content.

createdAt (optional)
Date Date and time when the report was created. format: date-time
example: 2024-10-31T12:00Z

inline_response_201_2 [Up]

reportId (optional)
Integer Unique identifier for the report.
example: 101

userId (optional)
Integer ID of the user who submitted the report.
example: 123

blogPostId (optional)
Integer ID of the reported blog post.
example: 456

explanation (optional)
String Explanation provided by the user.
example: This post contains inappropriate content.

createdAt (optional)
Date Date and time when the report was created. format: date-time
example: 2024-10-31T12:00Z

inline_response_201_3 [Up]

message (optional)
String
example: Template forked successfully

me_avatar_body [Up]

file (optional)
byte[] The avatar image file to be uploaded. format: binary

templates_templateId_body [Up]

title (optional)
String The title of the template.
example: My Updated Template

description (optional)
String The description of the template.
example: An updated description of the template.

tags (optional)
array[String]
example: ["tag1","tag2"]

language (optional)
String The language of the template.
example: javascript

stdin (optional)
String Optional standard input for code execution.
example: input data

code (optional)
String The code content of the template.
example: console.log('Hello, world!');

v1_blogposts_body Up

title
String Title of the blog post.
example: Getting Started with Templates

description
String Short description of the blog post.
example: An introductory guide on templates.

content
String Full content of the blog post.
example: Lorem ipsum dolor sit amet...

tags (optional)
array[String] Tags associated with the blog post.
example: ["guide","templates"]

links (optional)
array[BigDecimal]
example: [1,2]

v1_execute_body Up

code
String

stdin (optional)
String

language
String
Enum:
python
javascript
java
c
cpp
example: python

v1_me_body Up

email (optional)
String The user's email address. format: email
example: updateduser@example.com

firstName (optional)
String The user's first name.
example: Jane

lastName (optional)
String The user's last name.
example: Smith

avatar (optional)
String Optional URL for the user's avatar image. format: uri
example: https://example.com/avatar.jpg

phoneNumber (optional)
String Optional phone number of the user.
example: +9876543210

v1_templates_body Up

title
String The title of the template. Each user can have only one template with the same title.
example: Sample Template

description

String The description of the template.
example: A description of the sample template.

code
String The code content of the template.
example: console.log('Hello, world!');

stdin (optional)
String Optional standard input for code execution.
example: input data

language
String Programming language of the template.
example: javascript

tags (optional)
array[String] Tags associated with the template.
example: ["example","template"]