**.API Endpoints**

**User Authentication**

1. **Sign Up**
   * **Endpoint**: POST /api/auth/signup
   * **Description**: Create a new user account.
   * **Payload**:

{

"firstName": "John",

"lastName": "Doe",

"email": "john@example.com",

"password": "securepassword"

}

* + **Response**:

{

"message": "User created successfully."

}

1. **Log In**
   * **Endpoint**: POST /api/auth/login
   * **Description**: Log in a user and return a JWT token.
   * **Payload**:

{

"email": "john@example.com",

"password": "securepassword"

}

* + **Response**:

{

"token": "JWT\_TOKEN"

}

1. **Log Out**
   * **Endpoint**: POST /api/auth/logout
   * **Description**: Log out a user.
   * **Response**:

{

"message": "Logged out successfully."

}

**Blog Posts**

1. **Create Blog Post**
   * **Endpoint**: POST /api/posts
   * **Description**: Create a new blog post.
   * **Payload**:

{

"title": "My First Post",

"content": "This is the content of my first post.",

"tags": ["introduction", "first"]

}

* + **Response**:

{

"message": "Post created successfully."

}

1. **Edit Blog Post**
   * **Endpoint**: PUT /api/posts/:id
   * **Description**: Update an existing blog post.
   * **Payload**:

{

"title": "Updated Title",

"content": "Updated content.",

"tags": ["updated"]

}

* + **Response**:

{

"message": "Post updated successfully."

}

1. **Delete Blog Post**
   * **Endpoint**: DELETE /api/posts/:id
   * **Description**: Delete a blog post.
   * **Response**:

{

"message": "Post deleted successfully."

}

1. **Get All Blog Posts**
   * **Endpoint**: GET /api/posts
   * **Description**: Retrieve a paginated list of all blog posts.
   * **Query Parameters**: page, limit
   * **Response**:

{

"totalPosts": 50,

"totalPages": 10,

"currentPage": 1,

"posts": [ /\* Array of posts \*/ ]

}

1. **Get Single Blog Post**
   * **Endpoint**: GET /api/posts/:id
   * **Description**: Retrieve a specific blog post by ID.
   * **Response**:

{

"id": 1,

"title": "My First Post",

"content": "This is the content of my first post.",

"tags": ["introduction", "first"]

}

**Comments**

1. **Create Comment**
   * **Endpoint**: POST /api/posts/:postId/comments
   * **Description**: Add a comment to a blog post.
   * **Payload**:

{

"content": "This is a comment."

}

* + **Response**:

{

"message": "Comment created successfully."

}

1. **Reply to Comment**
   * **Endpoint**: POST /api/comments/:commentId/replies
   * **Description**: Reply to a specific comment.
   * **Payload**:

{

"content": "This is a reply."

}

* + **Response**:

{

"message": "Reply created successfully."

}

1. **Edit Comment**
   * **Endpoint**: PUT /api/comments/:id
   * **Description**: Edit an existing comment.
   * **Payload**:

{

"content": "Updated comment content."

}

* + **Response**:

{

"message": "Comment updated successfully."

}

1. **Delete Comment**
   * **Endpoint**: DELETE /api/comments/:id
   * **Description**: Delete a comment.
   * **Response**:

{

"message": "Comment deleted successfully."

}

1. **Get Comments for a Post**
   * **Endpoint**: GET /api/posts/:postId/comments
   * **Description**: Retrieve a paginated list of comments for a specific post.
   * **Query Parameters**: page, limit
   * **Response**:

{

"totalComments": 20,

"totalPages": 4,

"currentPage": 1,

"comments": [ /\* Array of comments \*/ ]

}

**Ratings**

1. **Rate Blog Post**
   * **Endpoint**: POST /api/posts/:postId/rate
   * **Description**: Rate a blog post.
   * **Payload**:

{

"rating": 1 // or -1 for downvote

}

* + **Response**:

{

"message": "Post rated successfully."

}

1. **Rate Comment**
   * **Endpoint**: POST /api/comments/:commentId/rate
   * **Description**: Rate a comment.
   * **Payload**:

{

"rating": 1 // or -1 for downvote

}

* + **Response**:

{

"message": "Comment rated successfully."

}  
  
Execute.js: API endpoint used for executing code with optional inputs via stdin, with either output or error message being shown to the user. Only POST method is allowed. The required fields in the payload are language and code, with input being optional. eg.

request: { "language": "Python", "code": "print('hello')" }

response: { "output": "hello\n" }