CSC309 Data Model

Kyle

October 2024

Schema

- User($\underline{\text{uID}}$, username, password, name, email, avatar, phoneNumber, role)
- CodeTemplate($\underline{\text{ctID}}$, uID, title, explanation, code, forkID, private) The forkID is the ctID of the original template, null if the template is not forked. **Integrity Constraints:**
  - CodeTemplate[uID] $\subseteq$ User[uID]
  - CodeTemplate[forkID] $\subseteq$ CodeTemplate[ctID]
- Blog($\underline{\text{bID}}$, uID, title, content, up, down, flags, difference, absDifference, hide) **Integrity Constraints:**
  - Blog[uID] $\subseteq$ User[uID]
- BlogRating($\underline{\text{brID}}$, $\underline{\text{uID, bID}}$, upvote, downvote) **Integrity Constraints:**
  - BlogRating[uID] $\subseteq$ User[uID]
  - BlogRating[bID] $\subseteq$ Blog[bID]
- CommentRating($\underline{\text{crID}}$, $\underline{\text{uID, bID}}$, upvote, downvote) **Integrity Constraints:**
  - CommentRating[uID] $\subseteq$ User[uID]
  - CommentRating[bID] $\subseteq$ Blog[bID]
- ContentReport($\underline{\text{contentRID}}$, uID, bID, cID, reason) **Integrity Constraints:**
  - contentRID[uID] $\subseteq$ User[uID]
  - contentRID[bID] $\subseteq$ Blog[bID]
  - contentRID[cID] $\subseteq$ Comment[cID]

- Comment(cID, bID, uID, up, down, content, difference, absDifference, hide, parentCommentID, childComments) **Integrity Constraints:**
  - Comment[uID] $\subseteq$ User[uID]
  - Comment[bID] $\subseteq$ Blog[bID]
  - Comment[parentCommentID] $\subseteq$ Comment[cID]
  - Comment[childComments] $\subseteq$ Comment[cID]
- Tag(tID, bID, ctID, tag) **Integrity Constraints:**
  - Tag[ctID] $\subseteq$ CodeTemplate[ctID]
  - Tag[bID] $\subseteq$ Blog[bID]

User/Admin APIs

Register: Register the user onto the database (USER only)

```
POST    ~/api/user/register
RETURN  (200 OK) (405 Not Allowed) (401 Invalid Params)

JSON PARAMS (ASSUME REQUIRED UNLESS OTHERWISE STATED)

username (Unique): string of the username

password: password with a required strength (at least 9 long, 1 Number, 1 Capital)

name: name of the user

email (Unique): email of the form foo@domin

avatar: integer from set (1: pizza, 2: corndog)

phoneNumber (Not Required): phone number in the form XXX XXXX XXXX, must be 11
digits, whitespaces allowed

_ADMIN: ADMIN is only available by changing the role to "ADMIN" from a USER
directly form the database (for security).
An admin is created for you through the startup.sh file use (admin1, Admin1psss)
```

Login: Login the user/admin onto the database

```
POST    ~/api/user/login
RETURN  (200 Access/Refresh Tokens) (405 Not Allowed) (400 Invalid Params) (401
Unauthorized)

JSON PARAMS (ASSUME REQUIRED UNLESS OTHERWISE STATED)

username: username

password: password for the username
```

Refresh: Refresh access token

```
POST    ~/api/user/refresh
RETURN  (200 Access Token) (405 Not Allowed) (400 Invalid Params) (401
Unauthorized)

JSON PARAMS (ASSUME REQUIRED UNLESS OTHERWISE STATED)
```

```
refreshToken: string of the refresh token given from login
```

Get Info: Get user info

```
POST    ~/api/user/get_info
RETURN  (200 user info) (405 Not Allowed) (400 user not found)

JSON PARAMS (ASSUME REQUIRED UNLESS OTHERWISE STATED)

username: username of the user
```

Edit: Edit the user onto the database

```
POST    ~/api/user/edit
RETURN  (200 Access/Refresh Tokens) (405 Not Allowed) (401 Invalid Params) (403 Forbidden)

HEADERS: (Authorization, Bearer <<replace with access token>>)
JSON PARAMS (All not required)

username (Unique): new username

password: new password with a required strength (at least 9 long, 1 Number, 1 Capital)

name: new name of the user

email (Unique): new email of the form foo@domin

avatar: integer from set (1: pizza, 2: corndog)

phoneNumber: new phone number in the form XXX XXXX XXXX, must be 11 digits, whitespaces allowed
```

Protected_Test: Test API for user authentication of tokens.

```
This is not an actual API for production; it is a test for a internal function for token Auth

POST    ~/api/user/protected_test
RETURN  (200 User Data) (403 Forbidden)

HEADERS: (Authorization, Bearer <<replace with access token>>)
```

Protected_Test_ADMIN: Test API for admin authentication of tokens.

```
This is not an actual API for production; it is a test for a internal function for
token Auth

POST    ~/api/user/protected_test
RETURN  (200 Admin Data) (403 Forbidden)

HEADERS: (Authorization, Bearer <<replace with access token>>)
```

Run APIs

Run: Register the user onto the database (USER only)

```
GET    ~/api/user/run
RETURN  (200 stdout/stderr) (500 Server side error)

JSON PARAMS:

stdin (not required): standard input that users can use for their executable.
For example: ./a.out 1 2 3 => stdin: "1 2 3"

language (required): one of the language from c, c++, python, java, javascript

code (required): the actual code content

className (required when language=java): the class name of the main class,
only required when writing with java.
```

Code Template APIs

Save(Create) Code Template (USER)

```
POST    ~/api/user/code_templates
RETURN  (201 Code Template) (403 Forbidden) (503 error)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

Required parameters:

title: title of code template

explanation: explanation of code template

code: code of code template

tags: list of tags, and tags are strings
```

View and Search Code Template (USER/VISITOR)

```
GET    ~/api/user/code_templates
RETURN  (200 Code Template) (503 error)

HEADERS (OPTIONAL): (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (all optional)

page: number of page want to display (not providing this sets page to 1)

title: title of code template

code: code of code template

tags: list of tags, and tags are strings

NOTE:
Here are the example usages: ~/api/user/code_templates?page=1&title=computer,
~/api/user/code_templates?tags=Python, ~/api/user/code_templates?
explanation=I+am+happy
"I am happy" should be I+am+happy
```

Edit Code Template (USER)

```
PUT    ~/api/user/code_templates?id=
RETURN  (200 Code Template) (403 Forbidden) (503 error)

HEADERS: (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (Required)

id: id of code template, for example: ?id=2

JSON PARAMS (all optional)

title: title of code template

explanation: explanation of code template

code: code of code template

tags: list of tags, and tags are strings
```

Delete Code Template (USER)

```
DELETE    ~/api/user/code_templates?id=
RETURN  (200 message) (403 Forbidden)

HEADERS: (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (Required)

id: id of code template, for example: ?id=2
```

Fork Code Template (USER/VISITOR)

```
POST    ~/api/user/code_templates?fork=true&id=
RETURN  (200 Code Template) (403 Forbidden) (503 error)

HEADERS (optional): (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (Required)

id: id of code template, for example: ?id=2

JSON PARAMS (all optional)
```

```
title: title of code template

explanation: explanation of code template

code: code of code template

tags: list of tags, and tags are strings

```

Blog APIs

Create Blog Post (USER)

```
POST    ~/api/user/blogs
RETURN  (201 Created) (401 Unauthorized) (403 Forbidden)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

Required parameters:

title: title of blog post

content: content of blog post

Optional parameters:

tags: list of tags, and tags are strings

templates: list of code templates, and code templates represented by their ids
(integer)
```

Edit Blog Post (USER)

```
PUT    ~/api/user/blogs/[blogID]
RETURN  (200 OK) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS (all optional)

title: title of blog post

content: content of blog post

tags: list of tags, and tags are strings

templates: list of code templates, and code templates represented by their ids
(integer)

NOTE:
The blog post to be edited is specified by its id, [blogID].
The blog must be authored by the user in order for them to edit it.
If the blog has been hidden by the administrator, the user cannot edit it.
```

Delete Blog Post (USER)

```
PUT    ~/api/user/blogs/[blogID]
RETURN  (200 OK) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)

NOTE:
The blog post to be deleted is specified by its id, [blogID].
The blog must be authored by the user in order for them to delete it.
If the blog has been hidden by the administrator, the user cannot delete it.
```

Search and Sort Blog Posts (VISITOR/USER)

```
GET    ~/api/user/blogs
RETURN  (200 OK) (403 Forbidden)

HEADERS (OPTIONAL): (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (all optional)

title: title of blog post (partial match)

content: content of blog post (partial match)

tags: a tag string

templates: a template id (number)

sort: sort method - can be valued, controversial, or recent

page: page number, default is 1

NOTE:
Each tag and template is specified separately (e.g. ~/api/user/blogs?
tags=javascript&tags=react&tags=hello).
Blogs that have been hidden by the administrator will not appear, except to the
author and administrator when they are logged in.
```

Get Individual Blog Post (VISITOR/USER)

```
GET    ~/api/user/blogs/[blogID]
RETURN  (200 OK) (403 Forbidden) (404 Not Found)
```

```
HEADERS (OPTIONAL): (Authorization, Bearer <<replace with access token>>)

NOTE:
A blog that has been hidden by the administrator will not appear, except to the
author and administrator when they are logged in.
```

Add Comment or Reply to Existing Comment (USER)

```
POST    ~/api/user/blogs/[blogID]/comments
RETURN  (201 Created) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

Required parameters:

content: content of comment

Optional parameters:

parentCommentID: ID of existing comment to respond to. This comment must also be
in the blog specified by [blogID].
```

Get and Sort Comments of Blog (VISITOR/USER)

```
GET    ~/api/user/blogs/[blogID]/comments
RETURN  (200 OK) (403 Forbidden)

HEADERS (OPTIONAL): (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (optional)

sort: sort method - can be valued, controversial, or recent

page: page number, default is 1

NOTE:
Comments that have been hidden by the administrator will not appear, except to the
author and administrator when they are logged in.
```

Upvote/Downvote Blog Post (USER)

```
PUT    ~/api/user/blogs/[blogID]
RETURN  (200 OK) (201 Created) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

upvote: true or false
downvote: true or false

NOTE:
A blog that has been hidden by the administrator cannot be upvoted or downvoted.
```

Upvote/Downvote Comment (USER)

```
PUT    ~/api/user/blogs/[blogID]/comments/[commentID]
RETURN  (200 OK) (201 Created) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

upvote: true or false
downvote: true or false

NOTE:
The comment specified by [commentID] must be associated with the blog specified by
[blogID].
A comment that has been hidden by the administrator cannot be upvoted or
downvoted.
```

Content Report APIs

Report Blog Post (USER)

```
POST    ~/api/user/blog_report
RETURN  (200 reason) (403 Forbidden) (503 errors)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

Required parameter:

blogID: id of blog post

Optional parameter:

reason: reason for reporting blog post
```

Report Comment (USER)

```
POST    ~/api/user/comment_report
RETURN  (200 reason) (403 Forbidden) (503 errors)

HEADERS: (Authorization, Bearer <<replace with access token>>)

JSON PARAMS

Required parameter:

commentID: id of comment

Optional parameter:

reason: reason for reporting comment
```

Sort Blog Posts Based On Number Of Reports In Descending Order (ADMIN)

```
GET    ~/api/admin/sort_blogs
RETURN  (201 blogs) (403 Forbidden) (503 errors)

HEADERS: (Authorization, Bearer <<replace with access token>>)
```

```
QUERY PARAMS (optional)

page: number of page want to display (not providing this sets page to 1)

NOTE:
Here is an example:  ~/api/admin/sort_blogs?page=1
```

Sort Comments Based On Number Of Reports In Descending Order (ADMIN)

```
GET    ~/api/admin/sort_comments
RETURN  (201 comments) (403 Forbidden) (503 errors)

HEADERS: (Authorization, Bearer <<replace with access token>>)

QUERY PARAMS (optional)

page: number of page want to display (not providing this sets page to 1)

NOTE:
Here is an example:  ~/api/admin/sort_comments?page=1
```

Hide Blog Post (ADMIN)

```
POST    ~/api/admin/blogs/[id]
RETURN  (200 OK) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)
```

Hide Comment (ADMIN)

```
POST    ~/api/admin/comments/[id]
RETURN  (200 OK) (401 Unauthorized) (403 Forbidden) (404 Not Found)

HEADERS: (Authorization, Bearer <<replace with access token>>)
```