

CPEN321: Software Engineering

# **API Endpoints Documentation**

Group: Cyann

Chen Chen (3281 6143)  
Howard Zhou (4898 6137)  
Justin Toh (2954 9136)  
Luvian Wang (1693 9134)  
Xi Chu (1735 3137)  
Yufei Qiao (4885 7130)

# Table of Content

|   |   |
|---|---|
| 1. <u>User authentication APIs</u>          | 3 |
| 1.1. User login                             | 3 |
| 2. <u>User information retrieve APIs</u>    | 3 |
| 2.1. Find all users                         | 3 |
| 2.2. Find a user                            | 4 |
| 2.3. Find all posts of a user               | 4 |
| 2.4. Find all comments of a user            | 4 |
| 2.5. Get all courses added by a user        | 4 |
| 3. <u>Courses APIs</u>                      | 5 |
| 3.1. Create a course                        | 5 |
| 3.2. Get all courses                        | 5 |
| 3.3. Get a course                           | 5 |
| 3.4. Get all users in a course              | 5 |
| 3.5. Add instructors and/or TAs in a course | 6 |
| 3.6. Add a user to a course                 | 6 |
| 3.7. Remove a user from a course            | 6 |
| 4. <u>Posts APIs</u>                        | 7 |
| 4.1. Create a post                          | 7 |
| 4.2. Find all posts in a course             | 7 |
| 4.3. Find a post in a course                | 7 |
| 4.4. Edit a post                            | 8 |
| 4.5. Delete a post                          | 8 |
| 4.6. Delete all posts in a course           | 8 |

|  |           |
|--|-----------|
| <b>5. <u>Comments APIs</u></b>               | <b>9</b>  |
| 5.1. Create a comment on a post              | 9         |
| 5.2. Get all comments on a post              | 9         |
| 5.3. Get a comment on a post                 | 9         |
| 5.4. Edit a comment                          | 10        |
| 5.5. Set a comment as the answer to a post   | 10        |
| 5.6. Unset a comment as the answer to a post | 10        |
| 5.7. Upvote <sup>1</sup> for a comment       | 11        |
| 5.8. Reset the upvote for a comment          | 11        |
| 5.9. Delete a comment                        | 11        |
| <b>6. <u>Honor system APIs</u></b>           | <b>12</b> |
| 6.1. Get honor points of a user              | 12        |
| <b>7. <u>Search APIs</u></b>                 | <b>12</b> |
| 7.1. Search for posts in a course            | 12        |
| <b>8. <u>File Management APIs</u></b>        | <b>12</b> |
| 8.1. Upload a file                           | 12        |
| 8.2. Get a list of all files in a course     | 13        |
| 8.3. Download a file                         | 13        |
| 8.4. Delete a file                           | 13        |

---

<sup>1</sup> [Definition from [Oxford Online Dictionary](#)]: In an online context, register approval of or agreement with a post or poster by means of a particular icon.

## 1. User authentication APIs

### 1.1. User login

|                   |   |
|-------------------|---|
| Description       | Allow new and returning users to log in with their Facebook account. If log in is successful, a unique Json Web Token (JWT), together with this user's user type and ID will be returned.   |
| Route             | /api/users/register   |
| HTTP Method       | POST  |
| Parameters        | None.   |
| Data Parameters   | <b>email</b> [ <i>required</i> ]: a user's email address.<br><b>profileImg</b> [ <i>required</i> ]: a profile image of the user.<br><b>socialToken</b> [ <i>required</i> ]: a valid social token from Facebook.   |
| Access Permission | All users.  |
| Notes             | <p>Every time when a user attempts to log in, our system will ask for his/her permission to access his/her Facebook account. Once he/she agrees, our system will receive a social token from Facebook that uniquely identifies this user. Our system will then create a user object for our app which is stored in our database.</p> <p>At the same time, a time invariant JWT that contains information including this user's Facebook ID, user ID of our app and the user type will be generated. User information of the current user will be stored in req.user.</p> <p>All API calls except for this one will require a header that includes a JWT, which will be checked for validity. Only JWTs that contain information about existing users will be considered as valid. An invalid JWT will result in an unsuccessful API call.</p> |

## 2. User information retrieve APIs

### 2.1. Find all users

|                   |   |
|-------------------|---|
| Description       | Return a list of all users in the database. |
| Route             | /api/users                                  |
| HTTP Method       | GET   |
| Parameters        | None.                                       |
| Data Parameters   | None.                                       |
| Access Permission | All users.                                  |

## 2.2. Find a user

|                   |  |
|-------------------|--|
| Description       | Return the user information of the current user. |
| Route             | /api/users/my                                    |
| HTTP Method       | GET  |
| Parameters        | None.  |
| Data Parameters   | None.  |
| Access Permission | All users.                                       |

## 2.3. Find all posts of a user

|                   |   |
|-------------------|---|
| Description       | Return a list of all posts created by the current user. |
| Route             | /api/users/my/posts                                     |
| HTTP Method       | GET   |
| Parameters        | None.   |
| Data Parameters   | None.   |
| Access Permission | All users.  |

## 2.4. Find all comments of a user

|                   |  |
|-------------------|--|
| Description       | Return a list of all comments written by the current user. |
| Route             | /api/users/my/comments                                     |
| HTTP Method       | GET  |
| Parameters        | None.  |
| Data Parameters   | None.  |
| Access Permission | All users.   |

## 2.5. Get all courses added by a user

|                   |   |
|-------------------|---|
| Description       | Returns a list of courses and their detailed information that the current user is in. |
| Route             | /api/users/my/courseData  |
| HTTP Method       | GET   |
| Parameters        | None.   |
| Data Parameters   | None.   |
| Access Permission | All users.  |

### 3. Courses APIs

#### 3.1. Create a course

|                   |   |
|-------------------|---|
| Description       | Create an empty course object and stores in the database.   |
| Route             | /api/courses  |
| HTTP Method       | POST  |
| Parameters        | None.   |
| Data Parameters   | <b>courseName</b> [ <i>required</i> ]: name of this course.<br><b>Instructor</b> [ <i>optional</i> ]: the user ID of the instructor of this course. It can be added later (see section 3.5).<br><b>TAs</b> [ <i>optional</i> ]: the user ID of the teaching assistant(s) of this course. It can be added later (see section 3.5). |
| Access Permission | Instructor only.  |

#### 3.2. Get all courses

|                   |   |
|-------------------|---|
| Description       | Return a list of all courses and their detailed information in the database |
| Route             | /api/courses  |
| HTTP Method       | GET   |
| Parameters        | None.   |
| Data Parameters   | None.   |
| Access Permission | All users.  |

#### 3.3. Get a course

|                   |   |
|-------------------|---|
| Description       | Return the detailed information of a specific course.   |
| Route             | /api/courses/:courseId  |
| HTTP Method       | GET   |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to get information about. |
| Data Parameters   | None.   |
| Access Permission | All users.  |

#### 3.4. Get all users in a course

|                   |  |
|-------------------|--|
| Description       | Returns a list of all users and their detailed information who have added themselves in this course.         |
| Route             | /api/courses/users/:courseId   |
| HTTP Method       | GET  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to get users information from. |
| Data Parameters   | None.  |
| Access Permission | All users.   |

### 3.5. Add instructors and/or TAs in a course

|                   |  |
|-------------------|--|
| Description       | Add instructor(s) and teaching assistant(s) to a particular course   |
| Route             | /api/courses/:courseId   |
| HTTP Method       | PUT  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to add instructors and TAs to.   |
| Data Parameters   | Instructors [ <i>optional</i> ]: the user ID of the instructor(s) whom we want to add to the course.<br>TAs [ <i>optional</i> ]: the user ID of the TA(s) whom we want to add to the course. |
| Access Permission | Instructors only.  |

### 3.6. Add a user to a course

|                   |   |
|-------------------|---|
| Description       | Add the current user into a particular course that he/she has not been added into.                          |
| Route             | /api/courses/addUser/:courseId  |
| HTTP Method       | PUT   |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to add the current user into. |
| Data Parameters   | None.   |
| Access Permission | All users.  |

### 3.7. Remove a user from a course

|                   |  |
|-------------------|--|
| Description       | Remove the current user from a particular course that he/she has been added into                               |
| Route             | /api/courses/removeUser/:courseId  |
| HTTP Method       | PUT  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to remove the current user from. |
| Data Parameters   | None.  |
| Access Permission | All users.   |

## 4. Posts APIs

### 4.1. Create a post

|                   |   |
|-------------------|---|
| Description       | Create a new post in a particular course.   |
| Route             | /api/courses/:courseId/posts  |
| HTTP Method       | POST  |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that we want to create a new post in.   |
| Data Parameters   | <b>title</b> [required]: the title of the new post.<br><b>content</b> [required]: the detail that users want to include in this new post. |
| Access Permission | All users.  |

### 4.2. Find all posts in a course

|                   |  |
|-------------------|--|
| Description       | Return a list of all posts and their detailed information in a particular course.      |
| Route             | /api/courses/:courseId/posts   |
| HTTP Method       | GET  |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that we want to find posts in. |
| Data Parameters   | None.  |
| Access Permission | All users.   |

### 4.3. Find a post in a course

|                   |  |
|-------------------|--|
| Description       | Return a specific post and its detailed information in a particular course.  |
| Route             | /api/courses/:courseId/posts/:postId   |
| HTTP Method       | GET  |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that we want to find the post in.<br><b>postId</b> [required]: the post ID of the post that we want to find. |
| Data Parameters   | None.  |
| Access Permission | All users.   |



#### 4.4. Edit a post

|                   |  |
|-------------------|--|
| Description       | Change the title and/or content of an existing post.   |
| Route             | api/courses/:courseId/posts/:postId  |
| HTTP Method       | PUT  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to edit the post in.<br><b>postId</b> [ <i>required</i> ]: the post ID of the post that we want to edit.   |
| Data Parameters   | <b>title</b> [ <i>optional</i> ]: the updated title of this existing post. If this is empty, the title will remain unchanged.<br><b>content</b> [ <i>optional</i> ]: the updated content of this existing post. If this is empty, the content will remain unchanged. |
| Access Permission | The author of the post only.   |

#### 4.5. Delete a post

|                   |  |
|-------------------|--|
| Description       | Delete a post from the database.   |
| Route             | /api/courses/:courseId/posts/:postId   |
| HTTP Method       | DELETE   |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to delete the post in.<br><b>postId</b> [ <i>required</i> ]: the post ID of the post that we want to delete. |
| Data Parameters   | None.  |
| Access Permission | The author of the post and instructors (in order to delete any inappropriate posts).   |

#### 4.6. Delete all posts in a course

|                   |   |
|-------------------|---|
| Description       | Delete all posts in a course from the database.   |
| Route             | /api/courses/:courseId/posts  |
| HTTP Method       | DELETE  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to delete posts in. |
| Data Parameters   | None.   |
| Access Permission | Instructors only.   |

## 5. Comments APIs

### 5.1. Create a comment on a post

|                   |   |
|-------------------|---|
| Description       | Create a new comment on an existing post.   |
| Route             | /api/courses/:courseId/posts/:postId/comments   |
| HTTP Method       | POST  |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to comment on. |
| Data Parameters   | <b>commentContent</b> [required]: the content of this new comment. A comment with empty content is not allowed.   |
| Access Permission | All users.  |

### 5.2. Get all comments to a post

|                   |   |
|-------------------|---|
| Description       | Return a list of all comments and their detailed information on a particular post.  |
| Route             | /api/courses/:courseId/posts/:postId/comments   |
| HTTP Method       | GET   |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to find comments on. |
| Data Parameters   | None.   |
| Access Permission | All users.  |

### 5.3. Get a comment to a post

|                   |   |
|-------------------|---|
| Description       | Return a specific comment and its detailed information on a particular post.  |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId  |
| HTTP Method       | GET   |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to find the comment on.<br><b>commentId</b> [required]: the comment ID of the comment we want to find. |
| Data Parameters   | None.   |
| Access Permission | All users.  |

#### 5.4. Edit a comment

|                   |   |
|-------------------|---|
| Description       | Change the content of an existing comment.  |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId  |
| HTTP Method       | GET   |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to edit the comment on.<br><b>commentId</b> [required]: the comment ID of the comment we want to edit. |
| Data Parameters   | <b>content</b> [required]: the updated content of the existing comment. An attempt to change the comment to an empty comment is not allowed.  |
| Access Permission | The author of this existing comment only.   |

#### 5.5. Set a comment as the answer to a post

|                   |   |
|-------------------|---|
| Description       | Set an existing comment as the answer to a particular post.   |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId/setAsAnswer  |
| HTTP Method       | PUT   |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to set the comment as the answer to.<br><b>commentId</b> [required]: the comment ID of the comment we want to set as the answer. |
| Data Parameters   | None.   |
| Access Permission | Instructors and TAs. Only comments from students can be set as answer to posts.   |

#### 5.6. Unset a comment as the answer to a post

|                   |   |
|-------------------|---|
| Description       | Unset an existing comment that is the answer to a particular post to a non-answer comment.  |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId/unsetAsAnswer  |
| HTTP Method       | PUT   |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to unset the comment on.<br><b>commentId</b> [required]: the comment ID of the comment we want to unset to non-answer. |
| Data Parameters   | None.   |
| Access Permission | Instructors and TAs. Only comments from students and currently set as answers can be unset.   |

### 5.7. Upvote for a comment

|                   |   |
|-------------------|---|
| Description       | Upvote for a particular comment on a post.  |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId/upvote   |
| HTTP Method       | PUT   |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to upvote for the comment on.<br><b>commentId</b> [required]: the comment ID of the comment we want to upvote for. |
| Data Parameters   | None.   |
| Access Permission | Non-author of the comment only.   |

### 5.8. Reset the upvote for a comment

|                   |  |
|-------------------|--|
| Description       | Reset the upvote for a particular comment on a post that the current user has upvoted for.   |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId/upvote  |
| HTTP Method       | PUT  |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to reset the upvote for the comment on.<br><b>commentId</b> [required]: the comment ID of the comment we want to unset to non-answer. |
| Data Parameters   | None.  |
| Access Permission | Non-author of the comment only.  |

### 5.9. Delete a comment

|                   |   |
|-------------------|---|
| Description       |   |
| Route             | /api/courses/:courseId/posts/:postId/comments/:commentId  |
| HTTP Method       | DELETE  |
| Parameters        | <b>courseId</b> [required]: the course ID of the course that the target post is located.<br><b>postId</b> [required]: the post ID of the post that we want to delete the comment on.<br><b>commentId</b> [required]: the comment ID of the comment we want to delete. |
| Data Parameters   | None.   |
| Access Permission | Author of this comment and instructors (in order to delete any inappropriate comments).   |

## 6. Honor system APIs

### 6.1. Get honor points of a user

|                   |   |
|-------------------|---|
| Description       | Get the honor points of a particular user.  |
| Route             | /api/honor/:userId  |
| HTTP Method       | GET   |
| Parameters        | <b>userId</b> [ <i>required</i> ]: the user ID of the user whom we want to get the honor points of. |
| Data Parameters   | None.   |
| Access Permission | All users except for instructors.   |

## 7. Search APIs

### 7.1. Search for posts in a course

|                   |  |
|-------------------|--|
| Description       | Search for posts in a particular course that matches with keywords and/or filters.   |
| Route             | /api/courses/:courseId/search  |
| HTTP Method       | GET  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to search for posts in.  |
| Data Parameters   | None.  |
| Query             | <b>keyword</b> [ <i>optional</i> ]: the keyword that we want the title or the content of posts to contain.<br><b>type</b> [ <i>optional</i> ]: the user type of the author of the posts we search for.<br><b>weeksAgo</b> [ <i>optional</i> ]: the number of weeks that the posts we search for were created within. |
| Access Permission | All users.   |

## 8. File Management APIs

### 8.1. Upload a file

|                   |  |
|-------------------|--|
| Description       | Upload a file into a particular course.  |
| Route             | /api/:courseId/files/:type/upload  |
| HTTP Method       | POST   |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to upload a file into. |
| Data Parameters   | None.  |
| Access Permission | Instructors only   |

## 8.2. Get a list of all files in a course

|                   |   |
|-------------------|---|
| Description       | Return a list of files of a specific type in a particular course  |
| Route             | /api/:courseId/files/:type  |
| HTTP Method       | GET   |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to get a list of files in.<br><b>type</b> [ <i>required</i> ]: the type of files that we want to get a list of. |
| Data Parameters   | None.   |
| Access Permission | All users.  |

## 8.3. Download a file

|                   |   |
|-------------------|---|
| Description       | Download a particular file in a course  |
| Route             | /api/:courseId/files/:type/download/:fileName   |
| HTTP Method       | GET   |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to download the file from.<br><b>type</b> [ <i>required</i> ]: the type of file that we want to download.<br><b>fileName</b> [ <i>required</i> ]: the name of the file we want to download. |
| Data Parameters   | None.   |
| Access Permission | All users.  |

## 8.4. Delete a file

|                   |   |
|-------------------|---|
| Description       | Delete an existing file from a particular course.   |
| Route             | /api/:courseId/files/:type/:fileName  |
| HTTP Method       | DELETE  |
| Parameters        | <b>courseId</b> [ <i>required</i> ]: the course ID of the course that we want to delete the file from.<br><b>type</b> [ <i>required</i> ]: the type of file that we want to delete.<br><b>fileName</b> [ <i>required</i> ]: the name of the file we want to delete. |
| Data Parameters   | None.   |
| Access Permission | Instructors only.   |