**CrowdAcademy**

Add collection description…﻿

**POST http://127.0.0.1:5000/login**

http://127.0.0.1:5000/login

This endpoint accepts data posting and generates a JWT token for authentication form terminals and client applications.

It expects two parameters; identifier and password the identifier can be an email address or a username which the user used to register thier account.

﻿  
**Body** raw (json):

{

"identifier": "zulan",

"password": "Password"

}

**POST http://127.0.0.1:5000/logout**

http://127.0.0.1:5000/logout

This endpoint accepts data posting and generates a JWT token for authentication form terminals and client applications.

It expects two parameters; identifier and password the identifier can be an email address or a username which the user used to register thier account.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{

"identifier": "cameron",

"password": "Password"

}

**GET http://127.0.0.1:5000/account**

http://127.0.0.1:5000/account

This endpoint returns the account details of the authenticated user if the Authorization header has been set with a Bearer token.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**POST http://127.0.0.1:5000/account/create**

http://127.0.0.1:5000/account/create

This platform is used to register users, it takes three parameters, username, email and password and returns a success mesage and user\_id or an error message.

﻿

**Body** raw (json):

{

"username": "cameron",

"email": "cameron@crowdacademy.com",

"password": "Password",

"roles": ["student"]

}

**PUT http://127.0.0.1:5000/account/update**

http://127.0.0.1:5000/account/update

This endpoint returns the account details of the authenticated user if the Authorization header has been set with a Bearer token.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{ "permissions": ["create\_lesson"] }

**DELETE http://127.0.0.1:5000/account/delete**

http://127.0.0.1:5000/account/delete

Add request description…﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**POST http://127.0.0.1:5000/users/add**

http://127.0.0.1:5000/users/add

This platform is used to register users, it takes three parameters, username, email and password and returns a success mesage and user\_id or an error message.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{

"username": "amadou",

"email": "amadou@crowdacademy.com",

"password": "Password",

"roles": ["student"]

}

**GET http://127.0.0.1:5000/users**

http://127.0.0.1:5000/users

This endpoint requires the user to be authenticated and an admin user. It returns an array of objects representing registered users on the platform.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**GET http://127.0.0.1:5000/users/id/<string:user\_id>**

http://127.0.0.1:5000/users/id/662de66e58caca203a82026b

Add request description…﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**GET http://127.0.0.1:5000/users/email/<string: email>**

http://127.0.0.1:5000/users/email/cameron@crowdacademy.com

This platform is used to register users, it takes three parameters, username, email and password and returns a success mesage and user\_id or an error message.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{

"username": "ethan",

"email": "ethan@floids.co",

"password": "Password"

}

**GET http://127.0.0.1:5000/users/username/<string: username>**

http://127.0.0.1:5000/users/username/cameron

This platform is used to register users, it takes three parameters, username, email and password and returns a success mesage and user\_id or an error message.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{

"username": "ethan",

"email": "ethan@floids.co",

"password": "Password"

}

**PUT http://127.0.0.1:5000/users/update/<string: user\_id>**

http://127.0.0.1:5000/users/update/662ddcf3eee4cb7e027d44ff

This endpoint returns the account details of the authenticated user if the Authorization header has been set with a Bearer token.

﻿  
**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{

"password": "Malakai"

}

**DELETE http://127.0.0.1:5000/user/delete/<string: user\_id>**

http://127.0.0.1:5000/users/delete/662ddc8aeee4cb7e027d44fe

Add request description…﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**POST http://127.0.0.1:5000/create-super-admin**

http://127.0.0.1:5000/create-super-admin

Add request description…﻿

**GET http://127.0.0.1:5000/challenges**

http://127.0.0.1:5000/challenges

This endpoint requires the user to be authenticated and an admin user. It returns an array of objects representing registered users on the platform.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**GET http://127.0.0.1:5000/challenges/<challenge\_id: string>**

http://127.0.0.1:5000/challenges/66406141dac875efc2c9d008

This endpoint requires the user to be authenticated and an admin user. It returns an array of objects representing registered users on the platform.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**POST http://127.0.0.1:5000/challenges/create**

http://127.0.0.1:5000/challenges/create

This endpoint requires the user to be authenticated and an admin user. It returns an array of objects representing registered users on the platform.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json)

{

"title": "Help! I'm Stuck on Quadratic Equations",

"description": "Hey everyone, I'm having trouble with quadratic equations! Can anyone help me solve this one: ax^2 + bx + c = 0?",

"tags": ["math", "algebra", "quadratic-equations"]

}

**PUT http://127.0.0.1:5000/challenges/update/<challenge\_id: string>**

http://127.0.0.1:5000/challenges/update/66449205651487803d87f965

This endpoint requires the user to be authenticated and an admin user. It returns an array of objects representing registered users on the platform.

﻿  
**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json):

{

"title": "Updated Challenge Title",

"description": "Updated Challenge Description",

"status": "active",

"resource\_ids": [],

"tags": ["tag1", "tag2"]

}

**DELETE http://127.0.0.1:5000/challenges/delete/<challenge\_id: string>**

http://127.0.0.1:5000/challenges/delete/66406141dac875efc2c9d008

This endpoint requires the user to be authenticated and an admin user. It returns an array of objects representing registered users on the platform.

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**GET http://127.0.0.1:5000/resources**

http://127.0.0.1:5000/resources

This endpoint allows the creation of new resources. The request should be sent as an HTTP POST to <http://127.0.0.1:5000/resources/create>. The request body should be of form-data type and include the following parameters:

* "file" (type: file): The file to be uploaded as the resource.
* "resource\_type" (type: text): The type of the resource. can be image/video/audio.
* "description" (type: text): A description for the resource.

Upon successful creation, the response will include the details of the newly created resource.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**GET http://127.0.0.1:5000/resources/<resource\_id: string>**

http://127.0.0.1:5000/resources/6643ee03f3d6a6745a2df077

This endpoint allows the creation of new resources. The request should be sent as an HTTP POST to <http://127.0.0.1:5000/resources/create>. The request body should be of form-data type and include the following parameters:

* "file" (type: file): The file to be uploaded as the resource.
* "resource\_type" (type: text): The type of the resource. can be image/video/audio.
* "description" (type: text): A description for the resource.

Upon successful creation, the response will include the details of the newly created resource.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**POST http://127.0.0.1:5000/resources/create [image]**

http://127.0.0.1:5000/resources/create

This endpoint allows the creation of new resources. The request should be sent as an HTTP POST to <http://127.0.0.1:5000/resources/create>. The request body should be of form-data type and include the following parameters:

* "file" (type: file): The file to be uploaded as the resource.
* "resource\_type" (type: text): The type of the resource. can be image/video/audio.
* "description" (type: text): A description for the resource.

Upon successful creation, the response will include the details of the newly created resource.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** form-data

**file:** postman-cloud:///1ef1244c-6c09-4280-8b86-c3ff5d69151c

**resource\_type:** image

**description:** A tutorial on machine learning algorithms

**POST http://127.0.0.1:5000/resources/create [video]**

http://127.0.0.1:5000/resources/create

Add request description…﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** form-data

**file:** postman-cloud:///1ef123e5-4f8d-4280-96e3-83ef80e6b66a

**resource\_type:** video

**description:** An introduction clip for a hypothetical YouTube chanel

**POST http://127.0.0.1:5000/resources/create [audio]4**

http://127.0.0.1:5000/resources/create

Add request description…﻿

**Request Headers**

**Authorization**

Bearer [Enter JWT Bearer Token]

**Body** form-data

**File:** postman-cloud:///1ef1244c-635d-4eb0-ad7d-540111fe9a12

**resource\_type:** audio

**description:** An upbeat and energetic audio track with catchy melodies and rhythmic beats

**PUT http://127.0.0.1:5000/resources/update/<resource\_id: string>**

http://127.0.0.1:5000/resources/update/6643ee03f3d6a6745a2df077

This endpoint allows the creation of new resources. The request should be sent as an HTTP POST to <http://127.0.0.1:5000/resources/create>. The request body should be of form-data type and include the following parameters:

* "file" (type: file): The file to be uploaded as the resource.
* "resource\_type" (type: text): The type of the resource. can be image/video/audio.
* "description" (type: text): A description for the resource.

Upon successful creation, the response will include the details of the newly created resource.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]

**Body** raw (json)

{

"description": "A brand new description for the resource with id 6643ee03f3d6a6745a2df077"

}

**DELETE http://127.0.0.1:5000/resources/delete/<resource\_id: string>**

http://127.0.0.1:5000/resources/delete/66446ec260ef3124a2f723e6

This endpoint allows the creation of new resources. The request should be sent as an HTTP POST to <http://127.0.0.1:5000/resources/create>. The request body should be of form-data type and include the following parameters:

* "file" (type: file): The file to be uploaded as the resource.
* "resource\_type" (type: text): The type of the resource. can be image/video/audio.
* "description" (type: text): A description for the resource.

Upon successful creation, the response will include the details of the newly created resource.

﻿

**Request Headers**

**Authorization:** Bearer [Enter JWT Bearer Token]