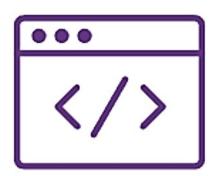


# API Documentation Group 15



# **JAVA CAN'T SEE SHARP**

Front to Back, We're on Track

Deliverable Two: Back-End

(CMPG323)

#### **Group Members:**

41425626 - Z. Hansa - (Group Leader)

37833707 - S. Ndobela

42304954 - S. Nhlapo

35928654 - T Gumbi

Submission date: 27/09/2024

# **Table of Contents**

API Documentation	3
Auth	3
Assignments	4
Courses	5
Enrollments	7
Feedback	8
Modules on Courses	9
Modules	10
Submissions	11
Users	
Utility	14
Videos	15

#### << NOTE: SWAGGER SNAPSHOTS INCLUDED AT THE END OF THIS DOCUMENT>>

#### **API LINK ACCESS**

https://nwu-hms-g6fjckard7fggqar.southafricanorth-01.azurewebsites.net/api-docs/



#### **API** Documentation

a description of each endpoint, including parameters, request/response examples, and possible error responses.

#### Auth

Authentication related endpoints

POST/auth/register Register a new user

Description:	Registers a new user to the system, providing basic account information.
Role based access:	Admins only
Parameters:	"UserNumber": 41425626, "Password": "securepassword123", "FirstName": "Barry", "LastName": "Leew", "Email": "41425626@mynwu.ac.za", "CourseID": 1, "UserRole": "Student"
Request/response examples:	Code 201 - User successfully registered
Possible error responses:	Code 400 - Invalid UserNumber Code 500 - Error registering user

# POST/auth/login Log in a user

Description:	Authenticates a user by checking their UserNumber (student number) and password, issuing an access token if successful.
Role based access:	No restrictions
Parameters:	"UserNumber": 41425626, "Password": "0123456789"
Request/response examples:	Code 201 - Access token successfully issued
Possible error responses:	Code 401 - Invalid credentials Code 500 - Error logging in

# POST / auth/logout Log out a user

Description:	Logs the user out of the system, invalidating the current session.
Role based access:	No restrictions
Parameters:	No Parameters
Request/response examples:	Code 401 - User successfully logged out



Possible error	Code 500
responses:	

#### Assignments

# POST / assignments Create a new assignment

Description:	Creates a new assignment for a specific module with title, instructions, and due date.
Role based access:	Lecture and Admin
Parameters:	"title": "string", "instructions": "string", "dueDate": "2024-09-27T14:37:52.552Z", "ModuleID": 0
Request/response examples:	Code 201 - Assignment created successfully
Possible error responses:	Code 400 - Bad request, invalid data Code 500 - Error creating assignment

#### GET/assignments Retrieve a list of all assignments

Description:	Retrieves a list of all assignments created in the system.
Role based access:	Admin
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of assignments
Possible error responses:	500 Error retrieving assignments

## GET/assignments/module/{ModuleID} Get all assignments for a specific module

Description:	Retrieves a list of assignments for a specific module by its ID.
Role based access:	Admin, Lecturer and Student
Parameters:	ModuleID
Request/response examples:	Code 200 - A list of assignments for a specific module
Possible error responses:	Code 404 - No assignments found for the given module Code 500 - Error retrieving assignments

GET/assignments/{id}Get details of a specific assignment



Description:	Fetches the details of a specific assignment using its ID.
Role based access:	Admin, Lecturer and Student
Parameters:	AssignmentID
Request/response examples:	Code 200 - Assignment details
Possible error responses:	Code 404 - Assignment not found Code 500 - Error retrieving assignment

# PUT/assignments/{id}Update assignment information

Description:	Updates the details of an assignment, such as title or due date, by its ID.
Role based access:	Admin Only
Parameters:	AssignmentID
Request/response examples:	Code 200 - Assignment updated successfully
Possible error responses:	Code 400 - Bad request, invalid data Code 500 - Error updating assignment

# DELETE/assignments/{id} Delete an assignment

Description:	Deletes an assignment from the system using its ID.
Role based access:	Admin Only
Parameters:	AssignmentID
Request/response examples:	Code 200 - Assignment deleted successfully
Possible error responses:	Code 500 - Error deleting assignment

#### Courses

# GET / courses Retrieve a list of all courses

Description:	Retrieves a list of all courses available in the system.
Role based access:	Admin
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of courses



responses:		Code 500 - Error retrieving courses
------------	--	-------------------------------------

# POST / courses Create a new course

Description:	Creates a new course with details such as course code, name, and duration.
Role based access:	Admin Only
Parameters:	"courseCode": "string", "courseName": "string", "duration": 3
Request/response examples:	Code 201 - Course created successfully
Possible error responses:	Code 500 - Error creating course

# GET/courses/{id}Get details of a specific course

Description:	Retrieves details of a specific course by its ID.
Role based access:	Admin
Parameters:	CourseID
Request/response examples:	Code 200 - Course details
Possible error responses:	Code 404 - Course not found Code 500 - Error retrieving course

# PUT/courses/{id}Update a course

Description:	Updates the information of a specific course, such as course name or duration.
Role based access:	Admin Only
Parameters:	CourseID
Request/response examples:	Code 200 - Course updated successfully
Possible error responses:	Code 500 - Error updating course

DELETE/courses/{id}Delete a course



Description:	Deletes a course from the system by its ID.
Role based access:	Admin Only
Parameters:	CourseID
Request/response examples:	Code 200 - Course deleted successfully
Possible error responses:	Code 500 - Error deleting course

#### **Enrollments**

# POST / enrollments Enrol a student in a module

Description:	Enrols a student in a specific module by providing student and module IDs.
Role based access:	Admin Only
Parameters:	"studentId": 0, "moduleId": 0
Request/response examples:	Code 201 - Student enrolled successfully
Possible error responses:	Code 500 - Error enrolling student

# GET/enrollments Retrieve a list of all enrollments

Description:	Retrieves a list of all enrollments in the system.
Role based access:	Admin
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of enrollments
Possible error responses:	Code 500 - Error retrieving enrollments

# DELETE / enrollments / { id} Remove a student from a module

Description:	Removes a student's enrollment from a module using the enrollment ID.
Role based access:	Admin Only
Parameters:	EnrollmentID
Request/response examples:	Code 200 - Enrollment removed successfully



Possible error	Code 500 - Error removing enrollment
responses:	

#### Feedback

# POST / feedbacks Provide feedback on a submission

Description:	Provides feedback on a student's submission by specifying the submission ID, feedback text, and mark.
Role based access:	Admin and Lecturer
Parameters:	"submissionId": 0, "lectureId": 0, "feedbackText": "string", "mark": 0
Request/response examples:	Code 201 - Feedback provided successfully
Possible error responses:	Code 500 - Error providing feedback

#### GET / feedbacks Retrieve a list of all feedback

Description:	Retrieves a list of all feedback entries.
Role based access:	Admins
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of feedback
Possible error responses:	Code 500 - Error retrieving feedback

# GET/feedbacks/{id} Get details of specific feedback

Description:	Fetches details of a specific feedback entry by its ID.
Role based access:	Admin, Lecturer and Student
Parameters:	FeedbackID
Request/response examples:	Code 200 - Feedback details
Possible error responses:	Code 404 - Feedback not found Code 500 - Error retrieving feedback

PUT/feedbacks/{id} Update specific feedback



Description:	Updates the details of a specific feedback, such as feedback text or mark.
Role based access:	Admin and Lecturer
Parameters:	FeedbackID
Request/response examples:	Code 200 - Feedback updated successfully
Possible error responses:	Code 404 - Feedback not found Code 500 - Error updating feedback

# DELETE / feedbacks / {id} Delete specific feedback

Description:	Deletes a specific feedback entry by its ID.
Role based access:	Admin and Lecturer
Parameters:	FeedbackID
Request/response examples:	Code 200 - Feedback deleted successfully
Possible error responses:	Code 404 - Feedback not found Code 500 - Error deleting feedback

#### GET/feedbacks/download/csv Download feedback data as CSV

Description:	Downloads feedback data in CSV format.
Role based access:	Admin and Lecturer
Parameters:	No Parameters
Request/response examples:	Code 200 - CSV file of feedback data
Possible error responses:	Code 404 - No feedback data found Code 500 - Error generating feedback CSV

#### Modules on Courses

#### POST / module - on - course Add a module to a course

Description:	Adds a specific module to a course.
Role based access:	Admin Only
Parameters:	"courseld": 0, "moduleld": 0
Request/response examples:	Code 201 - Module added to course successfully



Possible error	Code 500 - Error adding module to course
responses:	

## **DELETE**/module-on-course/{id} Remove a module from a course

Description:	Removes a module from a course by its relationship ID.
Role based access:	Admin Only
Parameters:	The ID of the module on course relationship
Request/response examples:	Code 200 - Module removed from course successfully
Possible error responses:	Code 404 - Module not found Code 500 - Error removing module from course

#### Modules

# POST / modules Create a new module

Description:	Creates a new module in the Human Movement Science Faculty.
Role based access:	Admins only
Parameters:	"moduleCode": "string", "moduleName": "string", "moduleDescription": "string", "lecturerId": 0
Request/response examples:	Code 201 - Module created successfully
Possible error responses:	Code 500 - Error creating module

#### GET/modules Retrieve a list of all modules

Description:	Retrieves a list of all modules in the system.
Role based access:	Admin Only
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of modules
Possible error responses:	Code 500 - Error retrieving modules

GET/modules/{id}Get details of a specific module



Description:	Fetches the details of a specific module by its ID.
Role based access:	Admin, Lecturer and Student
Parameters:	ModuleID
Request/response examples:	Code 200 - Module details
Possible error responses:	Code 404 - Module not found Code 500 - Error retrieving module

# PUT/modules/{id} Update module information

Description:	Updates the information of a specific module.
Role based access:	Admin only
Parameters:	ModuleID
Request/response examples:	Code 200 - Module updated successfully
Possible error responses:	Code 500 - Error updating module

# DELETE/modules/{id} Delete a module

Description:	Deletes a specific module by its ID.
Role based access:	Admin only
Parameters:	ModuleID
Request/response examples:	Code 200 - Module deleted successfully
Possible error responses:	Code 500 - Error deleting module

#### Submissions

# POST/submissions Submit an assignment

Description:	Submits an assignment with details such as submission text and status and a link (foreign key reference) to their respective video submission.
Role based access:	Student and Admin
Parameters:	"studentId": 0, "assignmentId": 0, "submissionText": "string", "videoId": 0, "status": "Submitted"



Request/response examples:	Code 201 - Assignment submitted successfully
Possible error responses:	Code 500 - Error submitting assignment

#### GET/submissions Retrieve a list of all submissions

Description:	Retrieves a list of all assignment submissions.
Role based access:	Lecturer and Admin
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of submissions
Possible error responses:	Code 500 - Error retrieving submissions

# GET/submissions/{id} Get details of a specific submission

Description:	Fetches the details of a specific submission by its ID.
Role based access:	Lecturer and Admin
Parameters:	SubmissionID
Request/response examples:	Code 200 - Submission details
Possible error responses:	Code 404 - Submission not found Code 500 - Error retrieving submissions

# PUT/submissions/{id} Update submission status

Description:	Updates the status of a specific submission.
Role based access:	Lecturer and Admin
Parameters:	SubmissionID "status": "string"
Request/response examples:	Code 200 - Submission status updated successfully
Possible error responses:	Code 500 - Error updating submission status

DELETE/submissions/{id}Delete a submission



Description:	Deletes a specific submission by its ID.
Role based access:	Student and Admin
Parameters:	SubmissionID
Request/response examples:	Code 200 - Submission deleted successfully
Possible error responses:	Code 500 - Error deleting submission

#### Users

#### GET/users Get a list of all users

Description:	Retrieves a list of all users in the system.
Role based access:	Admins only
Parameters:	No Parameters
Request/response examples:	Code 200 - A list of users
Possible error responses:	Code 500 - Error retrieving users Code 403 - Access denied

# GET/users/{id} Get details of a specific user

Description:	Fetches details of a specific user by their ID.
Role based access:	Admins only
Parameters:	UserID
Request/response examples:	Code 200 - User Details
Possible error responses:	Code 403 - Access denied Code 404 - User not found Code 500 - Error retrieving user

# PUT/users/{id} Update user details

Description:	Updates the details of a specific user, such as email or role.
Role based access:	Admins only
Parameters:	"Username": "string", "FirstName": "string", "LastName": "string", "Email": "string", "UserRole": "string",



	"CourseID": 0
Request/response examples:	Code 200 - User updated successfully
Possible error responses:	Code 403 - Access denied Code 500 - Error updating user

# DELETE / users / { id} Delete a user

Description:	Deletes a user from the system by their ID.
Role based access:	Admins only
Parameters:	UserID
Request/response examples:	Code 200 - User deleted successfully
Possible error responses:	Code 403 - Access denied Code 500 - Error deleting user

#### Utility

# GET/api/files/download/{id} Download a video file using its ID

Description:	Initiates a download for a video file using its ID.
Role based access:	Lecturer and Admin
Parameters:	VideoID
Request/response examples:	Code 200 - File download initiated
Possible error responses:	Code 404 - Video not found Code 500 - Error downloading video

# GET/api/files/stream/{id} Stream a video file using its ID

Description:	Streams a video file using its ID.
Role based access:	Lecturer and Admin
Parameters:	VideoID
Request/response examples:	Code 200 - File streaming initiated
Possible error responses:	Code 404 - Video not found Code 500 - Error streaming video



#### Videos

# POST / videos Upload a video

Description:	Uploads, converts and compresses a new video file to the database and file management system.
Role based access:	Admin and Student
Parameters:	Video File (binary) VideoTitle
Request/response examples:	Code 201 - Video uploaded successfully
Possible error responses:	Code 500 - Server error

#### GET/videos Retrieve all videos

Description:	Retrieves a list of all uploaded videos.
Role based access:	Admin and Lecturer
Parameters:	No Parameters
Request/response examples:	Code 200
Possible error responses:	Code 500 - Server error

# GET/videos/{id} Get details of a specific video

Description:	Fetches details of a specific video by its ID.
Role based access:	Admin , Student and Lecturer
Parameters:	VideoID
Request/response examples:	Code 200 - Video retrieved successfully
Possible error responses:	Code 404 - Video not found Code 500 - Server error

# DELETE/videos/{id} Delete a video

Description:	Deletes a video file from the database and from the file management system by its ID.
Role based access:	Admin and Student
Parameters:	VideoID

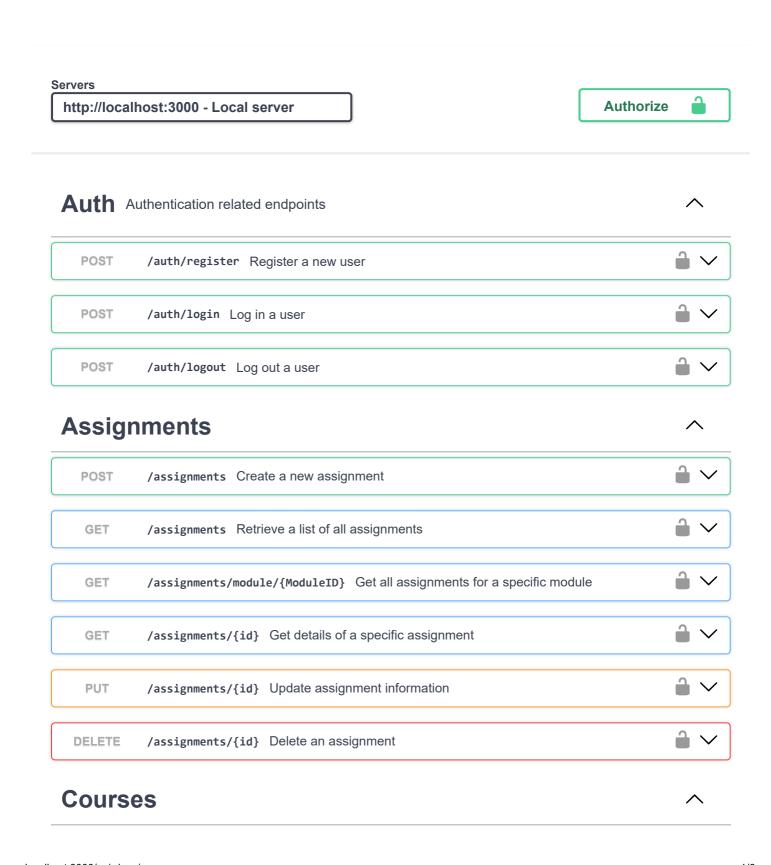


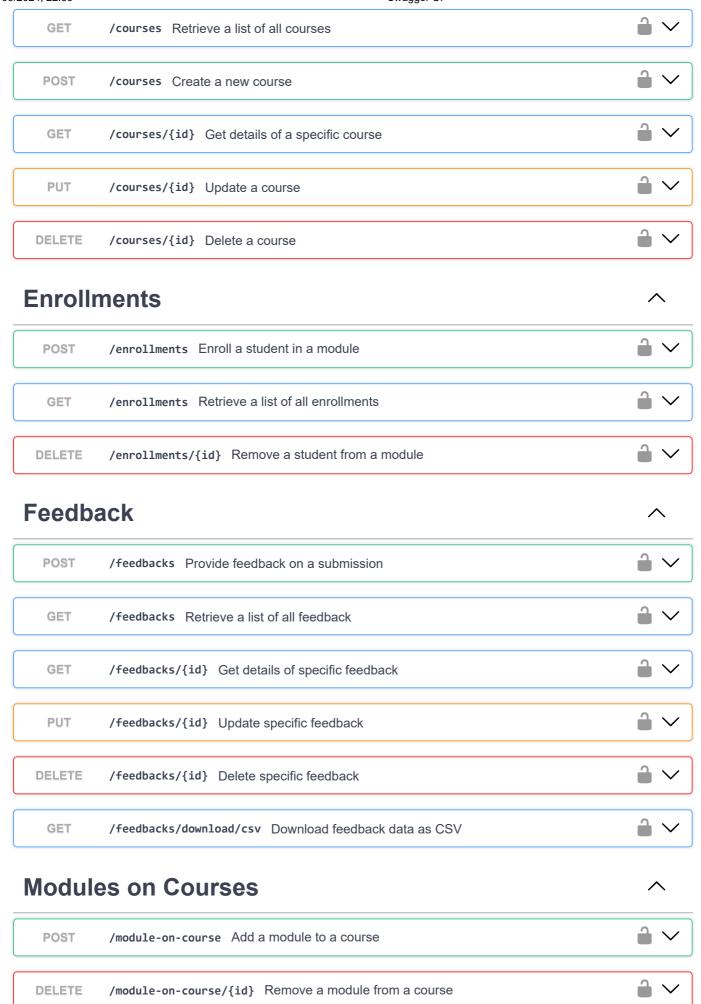
Request/response examples:	Code 200 - Video deleted successfully
Possible error responses:	Code 404 - Video not found Code 500 - Server error

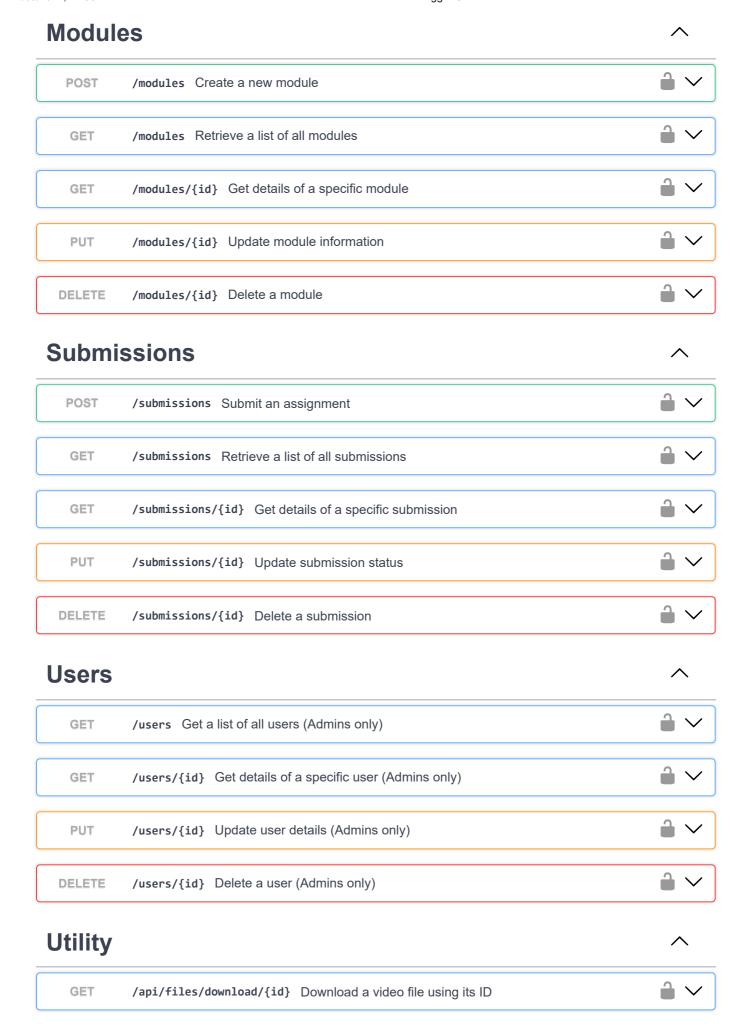


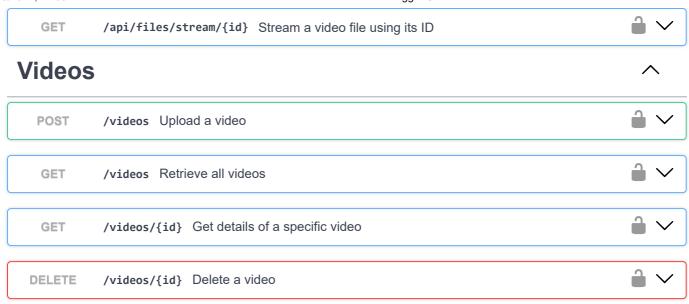
# **API Documentation** 1.0.0 OAS 3.0

Your API description here









```
Schemas
    Assignment
        title*
                                   [...]
        instructions
                                   [...]
        createdAt
                                   [...]
        dueDate*
                                   [...]
        ModuleID*
                                   [...]
    }
    Course
                {
        courseCode
                                   [...]
        courseName
                                   [...]
        duration
                                   [...]
        Year
                                   [...]
    }
    Enrollment
                     {
        StudentID
                                   [...]
        ModuleID
                                   [...]
    }
```

```
EnrollRequest
   studentId*
                              [...]
   moduleId*
                              [...]
}
Feedback
   mark
                              [...]
   submissionId
                              [...]
   lectureId
                              [...]
   feedbackText
                              [...]
   courseId
                              [...]
   moduleId
                              [...]
}
Module
   moduleCode
                              [...]
   moduleName
                              [...]
   moduleDescription
                              [...]
   lecturerId
                              [...]
}
Submission
                 {
   videoId
                              [...]
   studentId
                              [...]
   assignmentId
                              [...]
   submissionText
                              [...]
   status
                              [...]
}
User
   Username
                              [...]
   FirstName
                              [...]
   LastName
                              [...]
   Email
                              [...]
   UserRole
                              [...]
   CourseID
                              [...]
}
```