API Documentation   1.0.0   OAS 3.0

Your API description here

Servers

| http://localhost:3000 - Local server |

Authorize 🔓

Auth   Authentication related endpoints   ⌃

POST	/auth/register	Register a new user	🔓 ⌄
POST	/auth/login	Log in a user	🔓 ⌄
POST	/auth/logout	Log out a user	🔓 ⌄

Assignments   ⌃

POST	/assignments	Create a new assignment	🔓 ⌄
GET	/assignments	Retrieve a list of all assignments	🔓 ⌄
GET	/assignments/module/{ModuleID}	Get all assignments for a specific module	🔓 ⌄
GET	/assignments/{id}	Get details of a specific assignment	🔓 ⌄
PUT	/assignments/{id}	Update assignment information	🔓 ⌄
DELETE	/assignments/{id}	Delete an assignment	🔓 ⌄

Courses   ⌃

GET   `/courses`   Retrieve a list of all courses   🔓 ⌄

POST   `/courses`   Create a new course   🔓 ⌄

GET   `/courses/{id}`   Get details of a specific course   🔓 ⌄

PUT   `/courses/{id}`   Update a course   🔓 ⌄

DELETE   `/courses/{id}`   Delete a course   🔓 ⌄

Enrollments ⌃

POST   `/enrollments`   Enroll a student in a module   🔓 ⌄

GET   `/enrollments`   Retrieve a list of all enrollments   🔓 ⌄

DELETE   `/enrollments/{id}`   Remove a student from a module   🔓 ⌄

Feedback ⌃

POST   `/feedbacks`   Provide feedback on a submission   🔓 ⌄

GET   `/feedbacks`   Retrieve a list of all feedback   🔓 ⌄

GET   `/feedbacks/{id}`   Get details of specific feedback   🔓 ⌄

PUT   `/feedbacks/{id}`   Update specific feedback   🔓 ⌄

DELETE   `/feedbacks/{id}`   Delete specific feedback   🔓 ⌄

GET   `/feedbacks/download/csv`   Download feedback data as CSV   🔓 ⌄

Modules on Courses ⌃

POST   `/module-on-course`   Add a module to a course   🔓 ⌄

DELETE   `/module-on-course/{id}`   Remove a module from a course   🔓 ⌄

Modules ⌃

POST	**/modules** Create a new module	🔓 ⌄

GET	**/modules** Retrieve a list of all modules	🔓 ⌄

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

PUT	**/modules/{id}** Update module information	🔓 ⌄

DELETE	**/modules/{id}** Delete a module	🔓 ⌄

Submissions ⌃

POST	**/submissions** Submit an assignment	🔓 ⌄

GET	**/submissions** Retrieve a list of all submissions	🔓 ⌄

GET	**/submissions/{id}** Get details of a specific submission	🔓 ⌄

PUT	**/submissions/{id}** Update submission status	🔓 ⌄

DELETE	**/submissions/{id}** Delete a submission	🔓 ⌄

Users ⌃

GET	**/users** Get a list of all users (Admins only)	🔓 ⌄

GET	**/users/{id}** Get details of a specific user (Admins only)	🔓 ⌄

PUT	**/users/{id}** Update user details (Admins only)	🔓 ⌄

DELETE	**/users/{id}** Delete a user (Admins only)	🔓 ⌄

Utility ⌃

GET	**/api/files/download/{id}** Download a video file using its ID	🔓 ⌄

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

Videos   ⌃

POST	/videos	Upload a video	🔓 ⌄

GET	/videos	Retrieve all videos	🔓 ⌄

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

DELETE	/videos/{id}	Delete a video	🔓 ⌄

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
                              [...]
}

Video {

   VideoID
                                                 [...]
   VideoTitle
                                                 [...]
   VideoURL
                                                 [...]
}
example: OrderedMap { "VideoID": 1, "VideoTitle": "Introduction to AI", "VideoURL":
"https://hmsvideostorage.blob.core.windows.net/videos/intro_to_ai.mp4" }