# Attendance System API Documentation

## Base URL

For deployment: https://edutrack-backend-orms.onrender.com

## API Path Structure

The API uses a double /api prefix in the URL path. For example:

* Correct format: https://edutrack-backend-orms.onrender.com/api/api/courses
* Not: https://edutrack-backend-orms.onrender.com/api/courses

## Authentication

The API uses JWT (JSON Web Token) based authentication.

### Register a New User

POST /api/api/auth/register

**Request Body:**

{  
 "username": "john.doe",  
 "password": "password123",  
 "fullName": "John Doe",  
 "email": "john.doe@example.com"  
}

**Response:**

{  
 "success": true,  
 "message": "User registered successfully. Please check your email for verification code.",  
 "data": {  
 "email": "john.doe@example.com",  
 "verificationCode": "123456", // For testing only  
 "message": "A verification code has been sent to john.doe@example.com. If you don't receive it, check logs or contact support."  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Verify Email

POST /api/api/auth/verify-email

**Request Body:**

{  
 "email": "john.doe@example.com",  
 "code": "123456"  
}

**Response:**

{  
 "success": true,  
 "message": "Email verified successfully",  
 "data": {  
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",  
 "username": "john.doe",  
 "fullName": "John Doe",  
 "email": "john.doe@example.com",  
 "role": "ROLE\_STUDENT"  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Login

POST /api/api/auth/login

**Request Body:**

{  
 "username": "john.doe",  
 "password": "password123"  
}

**Response:**

{  
 "success": true,  
 "message": "Login successful",  
 "data": {  
 "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",  
 "username": "john.doe",  
 "fullName": "John Doe",  
 "email": "john.doe@example.com",  
 "role": "ROLE\_STUDENT"  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

## Courses

### Get All Courses

GET /api/api/courses  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Operation successful",  
 "data": [  
 {  
 "id": 1,  
 "courseCode": "CS101",  
 "courseName": "Introduction to Computer Science",  
 "description": "Fundamental concepts of programming",  
 "startTime": "09:00:00", // Note: Schedule info is now informational only for attendance  
 "endTime": "10:30:00", // Note: Schedule info is now informational only for attendance  
 "days": ["MONDAY", "WEDNESDAY"] // Note: Schedule info is now informational only for attendance  
 }  
 ],  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Get Current Active Courses (Deprecated - Schedule No Longer Used for Attendance)

GET /api/api/courses/current  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Note:** This endpoint’s logic based on current time might be misleading as schedule is no longer enforced for attendance recording. Consider removing or adjusting its purpose.

**Response:** Similar to Get All Courses, but filtered based on the current day/time matching the course schedule.

### Get Enrolled Courses

GET /api/api/courses/enrolled  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response:** Returns courses the current user is enrolled in.

### Create a New Course (Admin Only)

POST /api/api/courses  
Authorization: Bearer ADMIN\_TOKEN\_HERE

**Request Body:**

{  
 "courseCode": "CS501",  
 "courseName": "Advanced Algorithms",  
 "description": "Study of advanced algorithmic techniques",  
 "startTime": "14:00:00",  
 "endTime": "15:30:00",  
 "days": ["MONDAY", "WEDNESDAY"]  
}

**Response:**

{  
 "success": true,  
 "message": "Course created successfully",  
 "data": {  
 "id": 6,  
 "courseCode": "CS501",  
 "courseName": "Advanced Algorithms",  
 "description": "Study of advanced algorithmic techniques",  
 "startTime": "14:00:00",  
 "endTime": "15:30:00",  
 "days": ["MONDAY", "WEDNESDAY"]  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Update a Course (Admin Only)

PUT /api/api/courses/{courseId}  
Authorization: Bearer ADMIN\_TOKEN\_HERE

**Request Body:** Same as Create Course

**Response:** Similar to Create Course response

### Delete a Course (Admin Only)

DELETE /api/api/courses/{courseId}  
Authorization: Bearer ADMIN\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Course deleted successfully",  
 "data": null,  
 "timestamp": "2023-03-23T12:34:56.789"  
}

## Attendance & Sessions

### Enroll in a Course

POST /api/api/attendance/enroll/{courseId}  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Successfully enrolled in Introduction to Computer Science",  
 "data": "You are now enrolled in CS101",  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Create Attendance Session (Professor Only)

**Requires the authenticated professor to be enrolled in the specified course.**

POST /api/api/attendance/sessions/create  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Request Body:**

{  
 "courseId": 1,  
 "expiryMinutes": 5 // Duration for which the code will be valid  
}

**Response (Success - 201 Created):**

{  
 "success": true,  
 "message": "Attendance session created successfully",  
 "data": {  
 "sessionId": 123,  
 "courseId": 1,  
 "courseCode": "CS101",  
 "courseName": "Introduction to Computer Science",  
 "verificationCode": "A8F3K9", // The generated code for students  
 "createdAt": "2023-04-11T10:30:00.123",  
 "expiresAt": "2023-04-11T10:35:00.123",  
 "active": true  
 },  
 "timestamp": "2023-04-11T10:30:00.456"  
}

**Response (Error - 400 Bad Request / 403 Forbidden / 404 Not Found):**

{  
 "success": false,  
 "message": "Error message (e.g., Expiry time must be positive, Professor not associated with this course, Course not found)",  
 "data": null,  
 "timestamp": "2023-04-11T10:30:00.456"  
}

### Record Attendance (Student Only)

POST /api/api/attendance/record  
Authorization: Bearer STUDENT\_TOKEN\_HERE

**Request Body:**

{  
 "courseId": 1,  
 "verificationCode": "A8F3K9" // Code provided by the professor for an active session  
}

**Response (Success - 200 OK):**

{  
 "success": true,  
 "message": "Attendance recorded successfully",  
 "data": {  
 "id": 1,  
 "studentName": "John Doe",  
 "studentId": "S54321",  
 "courseCode": "CS101",  
 "courseName": "Introduction to Computer Science",  
 "timestamp": "2023-04-11T10:32:15.987",  
 "verified": true // Indicates the code was valid  
 // verificationMethod removed  
 },  
 "timestamp": "2023-04-11T10:32:15.999"  
}

**Response (Error - 400 Bad Request):**

{  
 "success": false,  
 "message": "Error message (e.g., User is not enrolled in this course, Attendance already recorded for this course today, Invalid or expired verification code for this course.)",  
 "data": null,  
 "timestamp": "2023-04-11T10:32:15.999"  
}

### Get Active Sessions for Professor (Professor Only)

GET /api/api/attendance/sessions/active  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Response (Success - 200 OK):**

{  
 "success": true,  
 "message": "Active sessions retrieved successfully",  
 "data": [  
 {  
 "sessionId": 123,  
 "courseId": 1,  
 "courseCode": "CS101",  
 "courseName": "Introduction to Computer Science",  
 "verificationCode": "A8F3K9",  
 "createdAt": "2023-04-11T10:30:00.123",  
 "expiresAt": "2023-04-11T10:35:00.123",  
 "active": true  
 }  
 // ... other active sessions for this professor  
 ],  
 "timestamp": "2023-04-11T10:40:00.111"  
}

### Get Session Attendees (Professor Only)

GET /api/api/attendance/sessions/{sessionId}/attendees  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Response (Success - 200 OK):**

{  
 "success": true,  
 "message": "Session attendees retrieved successfully",  
 "data": [  
 {  
 "id": 10,  
 "username": "john.doe",  
 "fullName": "John Doe",  
 "email": "john.doe@example.com",  
 "studentId": "S54321",  
 "role": "STUDENT"  
 },  
 {  
 "id": 15,  
 "username": "jane.smith",  
 "fullName": "Jane Smith",  
 "email": "jane.smith@example.com",  
 "studentId": "S67890",  
 "role": "STUDENT"  
 }  
 // ... other students who attended using the code during the session timeframe  
 ],  
 "timestamp": "2023-04-11T10:45:00.222"  
}

**Response (Error - 403 Forbidden / 404 Not Found):**

{  
 "success": false,  
 "message": "Error message (e.g., Professor does not have permission to view attendees for this session, Attendance session not found with ID: 123)",  
 "data": null,  
 "timestamp": "2023-04-11T10:45:00.222"  
}

### Get Daily Attendees for Course (Professor Only)

GET /api/api/attendance/course/{courseId}/date/{date}/attendees  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

* {date} should be in YYYY-MM-DD format.

**Response (Success - 200 OK):**

{  
 "success": true,  
 "message": "Attendees retrieved successfully",  
 "data": [  
 {  
 "id": 10,  
 "username": "john.doe",  
 "fullName": "John Doe",  
 "email": "john.doe@example.com",  
 "studentId": "S54321",  
 "role": "STUDENT"  
 }  
 // ... other students with verified attendance for this course on this date  
 ],  
 "timestamp": "2023-04-11T11:00:00.333"  
}

### Download Course Attendance Spreadsheet (Professor Only)

GET /api/api/attendance/course/{courseId}/spreadsheet  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE  
Accept: text/plain

**Response (Success - 200 OK):**

* **Content-Type:** text/plain
* **Content-Disposition:** attachment; filename="attendance-course-1-2023-04-11.csv"
* **Body:** Raw CSV data as text:
* Student Name,Student ID,Date,Time,Verified  
  John Doe,S54321,2023-04-10,09:05:12,Yes  
  Jane Smith,S67890,2023-04-10,09:06:30,Yes  
  John Doe,S54321,2023-04-11,10:32:15,Yes  
  ...

**Response (No Content - 204 No Content):**

* Returned if no attendance records exist for the course.

**Response (Error - 404 Not Found):**

* Returned if the course ID does not exist.

### Get User Attendance for Course

GET /api/api/attendance/user/{userId}/course/{courseId}  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response:** (Structure remains the same, but verificationMethod is removed)

{  
 "success": true,  
 "message": "Operation successful",  
 "data": [  
 {  
 "id": 1,  
 "studentName": "John Doe",  
 "studentId": "S54321",  
 "courseCode": "CS101",  
 "courseName": "Introduction to Computer Science",  
 "timestamp": "2023-04-11T10:32:15.987",  
 "verified": true  
 }  
 ],  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Get Total Classes (Number of Held Classes for a Course)

GET /api/api/attendance/sessions/class-days-count/{courseId}  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response (Success - 200 OK):**

{  
 "success": true,  
 "message": "Class days count retrieved successfully",  
 "data": 12, // Number of unique days attendance sessions were created for this course  
 "timestamp": "2025-04-15T12:00:00"  
}

* This endpoint returns the number of unique days on which attendance sessions were created for the course (i.e., the number of classes held).
* Professors can reset this count (e.g., at the start of a new semester) using the endpoint below.
* **Note:** This endpoint is accessible to both professors and students enrolled in the course.

### Reset Total Classes Count (Professor Only)

POST /api/api/attendance/sessions/class-days-count/{courseId}/reset  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Response (Success - 200 OK):**

{  
 "success": true,  
 "message": "Class days count reset successfully",  
 "data": null,  
 "timestamp": "2025-04-15T12:00:00"  
}

* This endpoint resets the class days count for the course for the requesting professor (sets the reset timestamp to now).
* Only professors can call this endpoint.

## Professor Requests

### Submit Professor Request (No Authentication Required)

POST /api/api/professor-requests

**Request Body:**

{  
 "fullName": "Professor Smith",  
 "email": "prof.smith@university.edu",  
 "idImageUrl": "/uploads/id-12345.jpg",  
 "department": "Computer Science",  
 "additionalInfo": "Specialized in Artificial Intelligence"  
}

**Response:**

{  
 "success": true,  
 "message": "Request submitted successfully",  
 "data": {  
 "id": 1,  
 "fullName": "Professor Smith",  
 "email": "prof.smith@university.edu",  
 "idImageUrl": "/uploads/id-12345.jpg",  
 "department": "Computer Science",  
 "additionalInfo": "Specialized in Artificial Intelligence",  
 "status": "PENDING",  
 "requestDate": "2023-03-23T12:34:56.789"  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

## Quizzes

### Create Quiz (Professor Only)

POST /api/quizzes  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Request Body:**

{  
 "title": "Midterm Exam",  
 "description": "Covers chapters 1-5",  
 "courseId": 1,  
 "startDate": "2023-04-01T09:00:00",  
 "endDate": "2023-04-01T23:59:59",  
 "durationMinutes": 60,  
 "questions": [  
 {  
 "text": "What is the capital of France?",  
 "type": "MULTIPLE\_CHOICE",  
 "points": 2,  
 "options": [  
 {  
 "text": "London",  
 "correct": false  
 },  
 {  
 "text": "Paris",  
 "correct": true  
 },  
 {  
 "text": "Rome",  
 "correct": false  
 }  
 ]  
 },  
 {  
 "text": "Define polymorphism",  
 "type": "TEXT\_ANSWER",  
 "points": 5,  
 "correctAnswer": "Polymorphism is the ability of an object to take many forms"  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Quiz created successfully",  
 "data": {  
 "id": 1,  
 "title": "Midterm Exam",  
 "description": "Covers chapters 1-5",  
 "startDate": "2023-04-01T09:00:00",  
 "endDate": "2023-04-01T23:59:59",  
 "durationMinutes": 60,  
 "questions": [...]  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Get Available Quizzes

GET /api/quizzes/available?courseId=1  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Operation successful",  
 "data": [  
 {  
 "id": 1,  
 "title": "Midterm Exam",  
 "description": "Covers chapters 1-5",  
 "startDate": "2023-04-01T09:00:00",  
 "endDate": "2023-04-01T23:59:59",  
 "durationMinutes": 60  
 },  
 ...  
 ],  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Start Quiz (Student Only)

POST /api/quizzes/{quizId}/start  
Authorization: Bearer STUDENT\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Quiz started successfully",  
 "data": {  
 "id": 1,  
 "quiz": {...},  
 "student": {...},  
 "startTime": "2023-04-01T10:15:30",  
 "completed": false  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Submit Quiz (Student Only)

POST /api/quizzes/{quizId}/submit  
Authorization: Bearer STUDENT\_TOKEN\_HERE

**Request Body:**

{  
 "quizId": 1,  
 "answers": [  
 {  
 "questionId": 1,  
 "selectedOptionId": 2  
 },  
 {  
 "questionId": 2,  
 "textAnswer": "Polymorphism is the ability of an object to take many forms"  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Quiz submitted successfully",  
 "data": {  
 "id": 1,  
 "quiz": {...},  
 "student": {...},  
 "startTime": "2023-04-01T10:15:30",  
 "endTime": "2023-04-01T11:10:45",  
 "completed": true,  
 "score": 7,  
 "maxScore": 7  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Get Quiz Submissions (Professor Only)

GET /api/quizzes/{quizId}/submissions  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Operation successful",  
 "data": [  
 {  
 "id": 1,  
 "studentId": 123,  
 "studentName": "John Doe",  
 "startTime": "2023-04-01T10:15:30",  
 "endTime": "2023-04-01T11:10:45",  
 "score": 85,  
 "maxScore": 100  
 }  
 ],  
 "timestamp": "2023-04-01T12:00:00"  
}

### Get Submission Details (Professor Only)

GET /api/quizzes/{quizId}/submissions/{submissionId}  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Operation successful",  
 "data": {  
 "id": 1,  
 "student": {  
 "id": 123,  
 "fullName": "John Doe",  
 "email": "john.doe@example.com"  
 },  
 "startTime": "2023-04-01T10:15:30",  
 "endTime": "2023-04-01T11:10:45",  
 "score": 85,  
 "maxScore": 100,  
 "answers": [  
 {  
 "questionId": 1,  
 "questionText": "What is the capital of France?",  
 "questionType": "MULTIPLE\_CHOICE",  
 "points": 2,  
 "pointsAwarded": 2,  
 "selectedOption": {  
 "id": 2,  
 "text": "Paris",  
 "correct": true  
 },  
 "correctOption": {  
 "id": 2,  
 "text": "Paris",  
 "correct": true  
 }  
 },  
 {  
 "questionId": 2,  
 "questionText": "Define polymorphism",  
 "questionType": "TEXT\_ANSWER",  
 "points": 5,  
 "pointsAwarded": 5,  
 "studentAnswer": "Polymorphism is the ability of an object to take many forms",  
 "correctAnswer": "Polymorphism is the ability of an object to take many forms"  
 }  
 ]  
 },  
 "timestamp": "2023-04-01T12:00:00"  
}

**Error Responses:**

* 400 Bad Request: If quiz hasn’t ended yet or professor doesn’t own the quiz
* 404 Not Found: If quiz or submission not found

### Get Professor’s Quizzes

GET /api/quizzes/my-quizzes  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": null,  
 "data": [  
 {  
 "id": 1,  
 "title": "Midterm Quiz",  
 "description": "Chapter 1-5 material",  
 "startDate": "2023-04-15T10:00:00",  
 "endDate": "2023-04-15T12:00:00",  
 "durationMinutes": 120,  
 "courseId": 1,  
 "creatorId": 2  
 }  
 ],  
 "timestamp": "2023-04-15T12:00:00"  
}

### Download Quiz Submissions

GET /api/quizzes/{quizId}/submissions/download  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

Downloads a CSV file containing all submissions for the quiz. File includes:

* Student ID
* Student Name
* Email
* Submission Date
* Score
* Max Score
* Percentage

Response is a CSV file download.

**Error Responses:**

* 400 Bad Request: If professor doesn’t own the quiz
* 404 Not Found: If quiz not found

### Update Quiz (Professor Only)

PUT /api/quizzes/{quizId}  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Request Body:**

{  
 "title": "Updated Midterm Quiz",  
 "description": "Updated quiz covering chapters 1-5",  
 "courseId": 1,  
 "startDate": "2023-04-15T10:00:00",  
 "endDate": "2023-04-15T12:00:00",  
 "durationMinutes": 120,  
 "questions": [  
 {  
 "text": "What is inheritance in OOP?",  
 "type": "MULTIPLE\_CHOICE",  
 "points": 5,  
 "options": [  
 {  
 "text": "A mechanism where one class acquires properties of another class",  
 "correct": true  
 },  
 {  
 "text": "A way to hide implementation details",  
 "correct": false  
 }  
 ]  
 },  
 {  
 "text": "Explain polymorphism",  
 "type": "TEXT\_ANSWER",  
 "points": 10,  
 "correctAnswer": "Polymorphism is the ability of different objects to respond to the same message in different ways"  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Quiz updated successfully",  
 "data": {  
 "id": 1,  
 "title": "Updated Midterm Quiz",  
 "description": "Updated quiz covering chapters 1-5",  
 "course": {...},  
 "creator": {...},  
 "startDate": "2023-04-15T10:00:00",  
 "endDate": "2023-04-15T12:00:00",  
 "durationMinutes": 120,  
 "questions": [...]  
 },  
 "timestamp": "2023-04-10T14:25:30"  
}

**Notes:**

* This endpoint is only available to the professor who created the quiz
* If the quiz already has attempts, only the end date can be modified (to extend the deadline)
* If there are no attempts yet, all quiz properties including questions can be modified

## Assignments

### Create Assignment (Professor Only)

POST /api/assignments  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Request Body:**

{  
 "title": "Final Project",  
 "description": "Implement a web application using Spring Boot",  
 "courseId": 1,  
 "dueDate": "2023-05-15T23:59:59",  
 "maxPoints": 100,  
 "files": [  
 {  
 "fileName": "project\_requirements.pdf",  
 "fileUrl": "/uploads/project\_requirements.pdf",  
 "contentType": "application/pdf",  
 "fileSize": 245789  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Assignment created successfully",  
 "data": {  
 "id": 1,  
 "title": "Final Project",  
 "description": "Implement a web application using Spring Boot",  
 "course": {...},  
 "creator": {...},  
 "dueDate": "2023-05-15T23:59:59",  
 "createdAt": "2023-03-23T12:34:56.789",  
 "maxPoints": 100,  
 "files": [...]  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Get Active Assignments

GET /api/assignments/active?courseId=1  
Authorization: Bearer YOUR\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Operation successful",  
 "data": [  
 {  
 "id": 1,  
 "title": "Final Project",  
 "description": "Implement a web application using Spring Boot",  
 "dueDate": "2023-05-15T23:59:59",  
 "maxPoints": 100  
 },  
 ...  
 ],  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Submit Assignment (Student Only)

POST /api/assignments/{assignmentId}/submit  
Authorization: Bearer STUDENT\_TOKEN\_HERE

**Request Body:**

{  
 "assignmentId": 1,  
 "notes": "My final project submission",  
 "files": [  
 {  
 "fileName": "FinalProject.zip",  
 "fileUrl": "/uploads/FinalProject.zip",  
 "contentType": "application/zip",  
 "fileSize": 3456789  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Assignment submitted successfully",  
 "data": {  
 "id": 1,  
 "assignment": {...},  
 "student": {...},  
 "notes": "My final project submission",  
 "submissionDate": "2023-05-14T20:45:12",  
 "graded": false,  
 "late": false,  
 "files": [...]  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Edit Assignment Submission (Student Only)

PUT /api/assignments/submissions/{submissionId}  
Authorization: Bearer STUDENT\_TOKEN\_HERE

**Request Body:**

{  
 "notes": "Updated submission with corrections",  
 "files": [  
 {  
 "fileName": "FinalProject-Updated.zip",  
 "fileUrl": "/uploads/FinalProject-Updated.zip",  
 "contentType": "application/zip",  
 "fileSize": 3789012  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Assignment submission updated successfully",  
 "data": {  
 "id": 1,  
 "assignment": {...},  
 "student": {...},  
 "notes": "Updated submission with corrections",  
 "submissionDate": "2023-05-15T11:20:30",  
 "gradedDate": "2023-05-14T15:30:00", // If previously graded  
 "score": 85, // If previously graded  
 "feedback": "Good work, but missing proper documentation", // If previously graded  
 "graded": true, // If previously graded  
 "late": false,  
 "files": [...]  
 },  
 "timestamp": "2023-05-15T11:20:30"  
}

**Notes:**

* This endpoint is only available to the student who made the submission
* Students can only edit submissions before the assignment due date
* If the submission was previously graded, editing will not change the graded status but will update the submission date, which can be compared with gradedDate to detect edits after grading

### Update Assignment (Professor Only)

PUT /api/assignments/{assignmentId}  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Request Body:**

{  
 "title": "Updated Final Project",  
 "description": "Implement a web application using Spring Boot with updated requirements",  
 "dueDate": "2023-05-20T23:59:59",  
 "maxPoints": 100,  
 "files": [  
 {  
 "fileName": "updated\_project\_requirements.pdf",  
 "fileUrl": "/uploads/updated\_project\_requirements.pdf",  
 "contentType": "application/pdf",  
 "fileSize": 267890  
 }  
 ]  
}

**Response:**

{  
 "success": true,  
 "message": "Assignment updated successfully",  
 "data": {  
 "id": 1,  
 "title": "Updated Final Project",  
 "description": "Implement a web application using Spring Boot with updated requirements",  
 "course": {...},  
 "creator": {...},  
 "dueDate": "2023-05-20T23:59:59",  
 "createdAt": "2023-03-23T12:34:56.789",  
 "maxPoints": 100,  
 "files": [...]  
 },  
 "timestamp": "2023-05-10T14:25:30"  
}

**Notes:**

* This endpoint is only available to the professor who created the assignment
* Updating the assignment will not affect existing submissions

### Grade Assignment Submission (Professor Only)

POST /api/assignments/submissions/{submissionId}/grade  
Authorization: Bearer PROFESSOR\_TOKEN\_HERE

**Request Body:**

{  
 "score": 85,  
 "feedback": "Good work, but missing proper documentation"  
}

**Response:**

{  
 "success": true,  
 "message": "Submission graded successfully",  
 "data": {  
 "id": 1,  
 "assignment": {...},  
 "student": {...},  
 "notes": "My final project submission",  
 "submissionDate": "2023-05-14T20:45:12",  
 "gradedDate": "2023-05-16T14:30:45",  
 "score": 85,  
 "feedback": "Good work, but missing proper documentation",  
 "graded": true,  
 "late": false,  
 "files": [...]  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Upload ID Image for Professor Request (No Authentication Required)

POST /api/api/upload/public  
Content-Type: multipart/form-data

**Form Data:**

* file: The image file to upload

**Response:**

{  
 "success": true,  
 "message": "File uploaded successfully",  
 "data": {  
 "fileName": "id-12345.jpg",  
 "fileUrl": "/uploads/id-12345.jpg",  
 "contentType": "image/jpeg",  
 "fileSize": 253421,  
 "uploadedAt": "2023-03-23T12:34:56.789"  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Get Pending Professor Requests (Admin Only)

GET /api/api/professor-requests  
Authorization: Bearer ADMIN\_TOKEN\_HERE

**Response:**

{  
 "success": true,  
 "message": "Operation successful",  
 "data": [  
 {  
 "id": 1,  
 "fullName": "Professor Smith",  
 "email": "prof.smith@university.edu",  
 "idImageUrl": "/uploads/id-12345.jpg",  
 "department": "Computer Science",  
 "additionalInfo": "Specialized in Artificial Intelligence",  
 "status": "PENDING",  
 "requestDate": "2023-03-23T12:34:56.789"  
 }  
 ],  
 "timestamp": "2023-03-23T12:34:56.789"  
}

### Review Professor Request (Admin Only)

PUT /api/api/professor-requests/{requestId}/review  
Authorization: Bearer ADMIN\_TOKEN\_HERE

**Request Body:**

{  
 "approved": true,  
 "rejectionReason": null,  
 "reviewedBy": "admin\_edutrack"  
}

or

{  
 "approved": false,  
 "rejectionReason": "Insufficient credentials",  
 "reviewedBy": "admin\_edutrack"  
}

**Response:**

{  
 "success": true,  
 "message": "Request approved successfully",  
 "data": {  
 "id": 1,  
 "fullName": "Professor Smith",  
 "email": "prof.smith@university.edu",  
 "idImageUrl": "/uploads/id-12345.jpg",  
 "department": "Computer Science",  
 "additionalInfo": "Specialized in Artificial Intelligence",  
 "status": "APPROVED",  
 "requestDate": "2023-03-23T12:34:56.789",  
 "reviewDate": "2023-03-24T10:15:30.123",  
 "reviewedBy": "admin\_edutrack"  
 },  
 "timestamp": "2023-03-24T10:15:30.456"  
}

## User Roles and Permissions

The system supports three types of users, each with different permissions:

### Admin

* Credentials: Username: admin\_edutrack, Password: A9$k2pL8#xB7!fR3
* Permissions:
  + View, create, update, and delete courses
  + Review professor requests
  + Access admin-specific endpoints

### Professor

* Created when an admin approves a professor request
* Permissions:
  + Create quizzes and assignments
  + Create and manage attendance sessions for their courses
  + View attendees for their sessions/courses
  + Download attendance spreadsheets for their courses
  + Access professor-specific endpoints
  + View enrolled students
  + Creating an attendance session implicitly records that the class took place.

### Student

* Default role for registered users
* Permissions:
  + Enroll in courses
  + Record attendance using verification codes
  + View available courses and attendance history

## File Upload

### Upload File (Authenticated)

POST /api/api/upload  
Authorization: Bearer YOUR\_TOKEN\_HERE  
Content-Type: multipart/form-data

**Form Data:**

* file: The file to upload

**Response:**

{  
 "success": true,  
 "message": "File uploaded successfully",  
 "data": {  
 "fileName": "document.pdf",  
 "fileUrl": "/uploads/b4f9c8d7-e6f5-4a3b-b2d1-c0d9e8f7a6b5.pdf",  
 "contentType": "application/pdf",  
 "fileSize": 253421,  
 "uploadedAt": "2023-03-23T12:34:56.789"  
 },  
 "timestamp": "2023-03-23T12:34:56.789"  
}