Postman Document

1. Set Up Postman

1. Create a Collection:
   * Open Postman and create a new collection named Online Learning Platform.
2. Add Environment Variables (Optional):
   * Create an environment for your backend URL ( http://localhost:5000).
   * Add a variable named base\_url with the value http://localhost:5000.

2. Authentication

Register a New User

* Method: POST
* URL: http://localhost:5000/api/auth/register
* Headers:
  + Content-Type: application/json
* Body (JSON):

json

{

"name": "John Doe",

"email": "john@example.com",

"password": "password123",

"role": "student"

}

* Response:

json

{

"message": "User registered successfully"

}

Login a User

* Method: POST
* URL: http://localhost:5000/api/auth/login
* Headers:
  + Content-Type: application/json
* Body (JSON):

json

{

"email": "john@example.com",

"password": "password123"

}

* Response:

json

{

"token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."

}

* Save the Token: Copy the token from the response. You'll use it for authenticated requests.

3. Course Management

Add a New Course

* Method: POST
* URL: http://localhost:5000/api/courses
* Headers:
  + Content-Type: application/json
  + Authorization: Bearer <token> (replace <token> with the JWT token from the login response)
* Body (JSON):

json

{

"title": "Introduction to React",

"description": "Learn the basics of React.js"

}

* Response:

json

{

"message": "Course added successfully"

}

Get All Courses

* Method: GET
* URL: http://localhost:5000/api/courses
* Headers:
  + Authorization: Bearer <token>
* Response:

json

[

{

"id": 1,

"title": "Introduction to React",

"description": "Learn the basics of React.js",

"created\_at": "2023-10-01T12:00:00.000Z"

}

]

Update a Course

* Method: PUT
* URL: http://localhost:5000/api/courses/1 (replace 1 with the course ID)
* Headers:
  + Content-Type: application/json
  + Authorization: Bearer <token>
* Body (JSON):

json

{

"title": "Introduction to React (Updated)",

"description": "Learn the basics of React.js (Updated)"

}

* Response:

json

{

"message": "Course updated successfully"

}

Delete a Course

* Method: DELETE
* URL: http://localhost:5000/api/courses/1 (replace 1 with the course ID)
* Headers:
  + Authorization: Bearer <token>
* Response:

json

{

"message": "Course deleted successfully"

}

4. Student Management

Add a New Student

* Method: POST
* URL: http://localhost:5000/api/students
* Headers:
  + Content-Type: application/json
  + Authorization: Bearer <token>
* Body (JSON):

json

{

"user\_id": 1,

"name": "John Doe",

"email": "john@example.com"

}

* Response:

json

{

"message": "Student added successfully"

}

Get All Students

* Method: GET
* URL: http://localhost:5000/api/students
* Headers:
  + Authorization: Bearer <token>
* Response:

json

[

{

"id": 1,

"user\_id": 1,

"name": "John Doe",

"email": "john@example.com"

}

]

Update a Student

* Method: PUT
* URL: http://localhost:5000/api/students/1 (replace 1 with the student ID)
* Headers:
  + Content-Type: application/json
  + Authorization: Bearer <token>
* Body (JSON):

json

{

"name": "John Doe (Updated)",

"email": "john.updated@example.com"

}

* Response:

json

{

"message": "Student updated successfully"

}

Delete a Student

* Method: DELETE
* URL: http://localhost:5000/api/students/1 (replace 1 with the student ID)
* Headers:
  + Authorization: Bearer <token>
* Response:

json

{

"message": "Student deleted successfully"

}

5. Enrollment Management

Enroll a Student in a Course

* Method: POST
* URL: http://localhost:5000/api/enrollments
* Headers:
  + Content-Type: application/json
  + Authorization: Bearer <token>
* Body (JSON):

json

{

"student\_id": 1,

"course\_id": 1

}

* Response:

Json

{

"message": "Student enrolled successfully"

}

Get All Enrollments

* Method: GET
* URL: http://localhost:5000/api/enrollments
* Headers:
  + Authorization: Bearer <token>
* Response:

json

[

{

"id": 1,

"student\_id": 1,

"course\_id": 1,

"enrolled\_at": "2023-10-01T12:00:00.000Z"

}

]

6. Testing Workflow in Postman

1. Register a User:
   * Use the /api/auth/register endpoint to create a new user.
2. Login:
   * Use the /api/auth/login endpoint to get a JWT token.
3. Add a Course:
   * Use the /api/courses endpoint to add a new course.
4. Add a Student:
   * Use the /api/students endpoint to add a new student.
5. Enroll a Student:
   * Use the /api/enrollments endpoint to enroll a student in a course.
6. View Data:
   * Use the /api/courses, /api/students, and /api/enrollments endpoints to view the data.