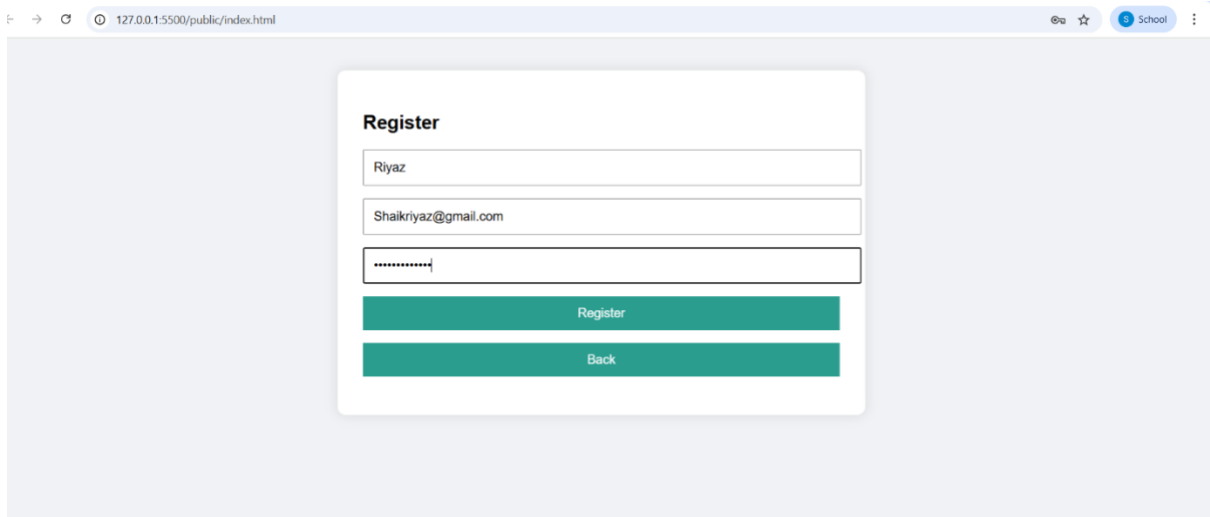
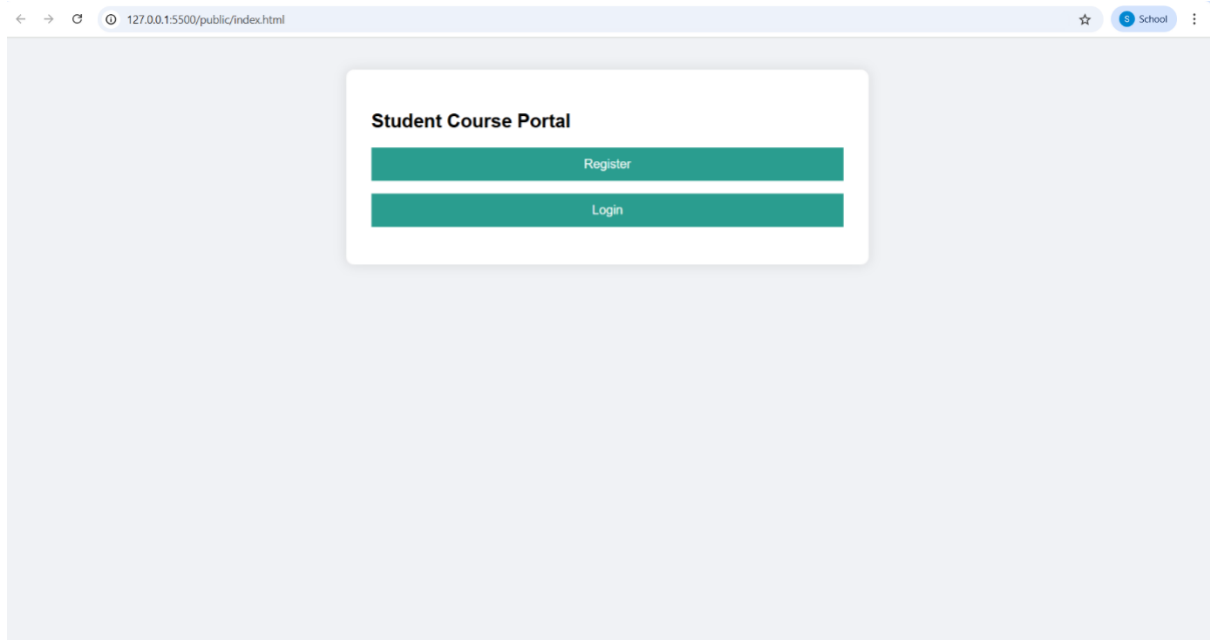
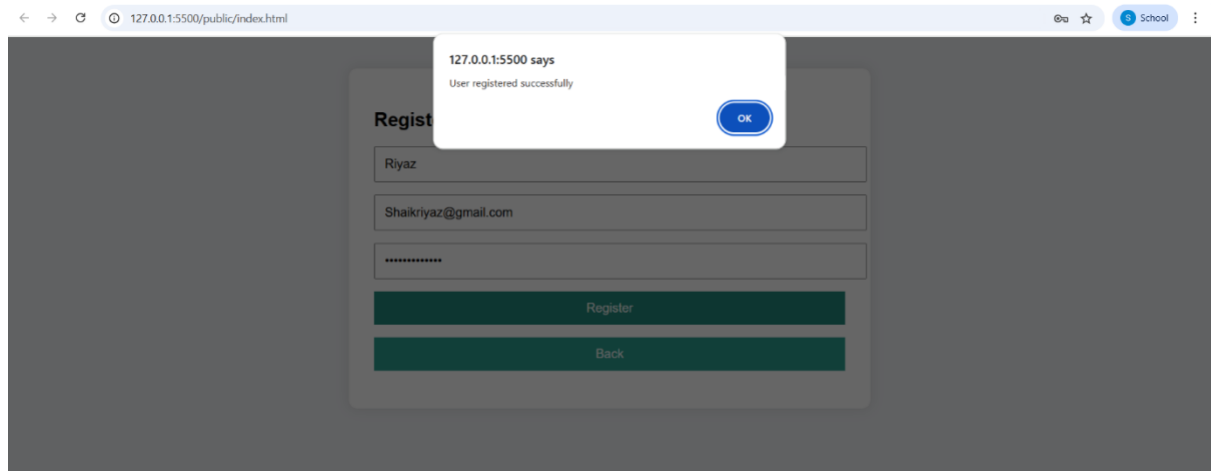


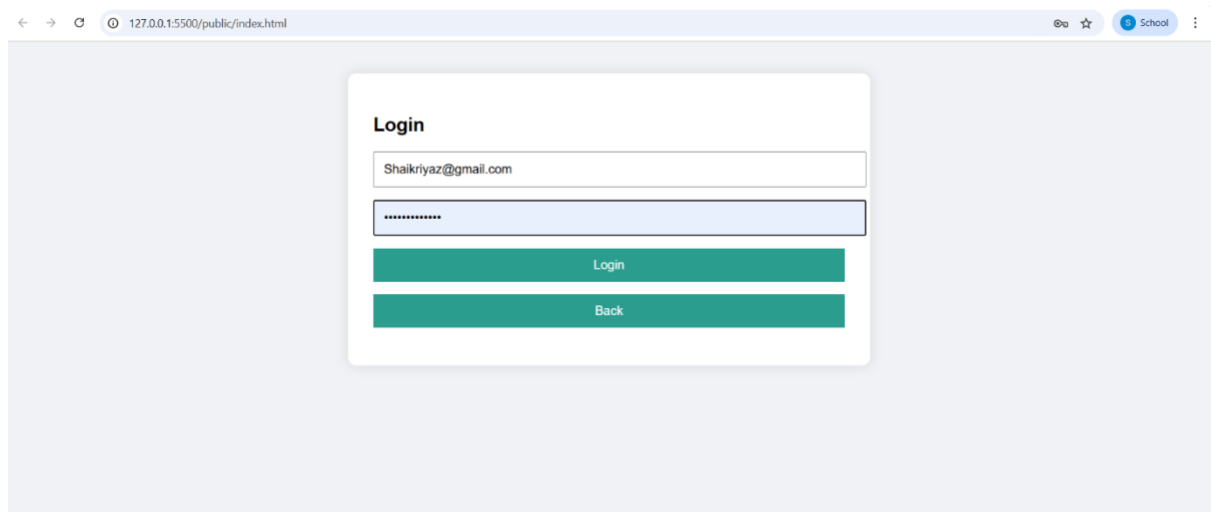
# FRONTEND LOGIN

## First Step is to Register



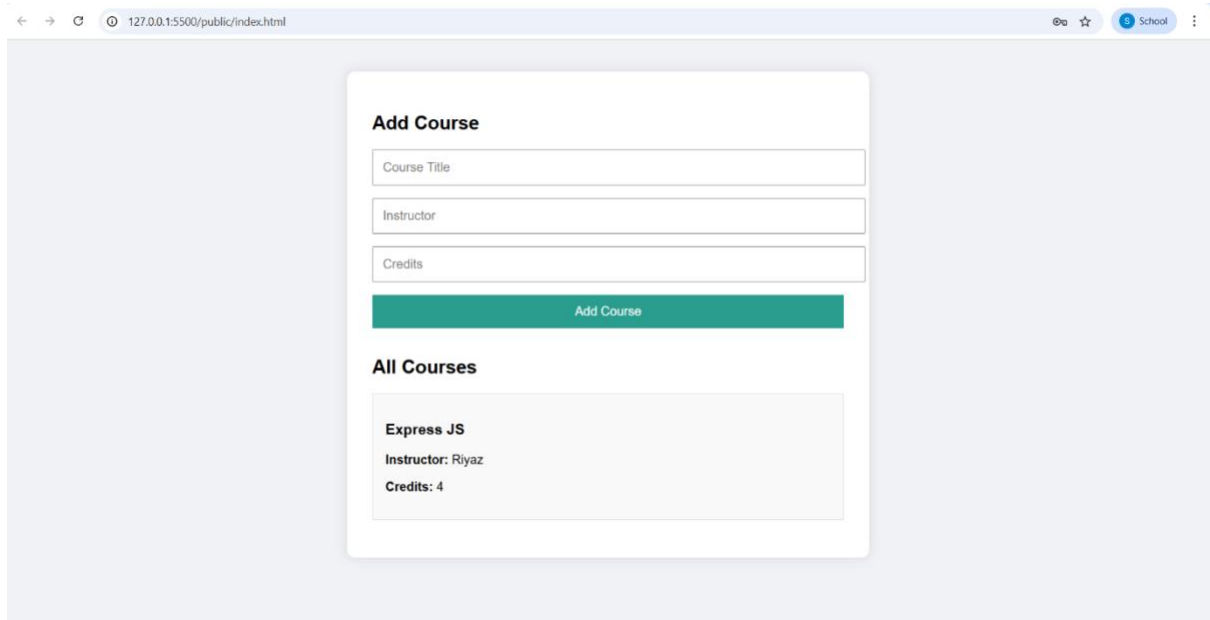


**Registration is successfully completed next step is to login with same information as registred to login in..**



**After Click on the Login button , We had logged in and Our Course Add Page will be openend**

## “Add Course” Section



A screenshot of a web browser showing a form titled "Add Course". The form has three input fields: "Course Title", "Instructor", and "Credits". Below these fields is a green button labeled "Add Course". Underneath the button, there is a section titled "All Courses" which displays a list of courses. The first course listed is "Express JS" with the instructor "Riyaz" and "Credits: 4".

**Add Course**

Course Title

Instructor

Credits

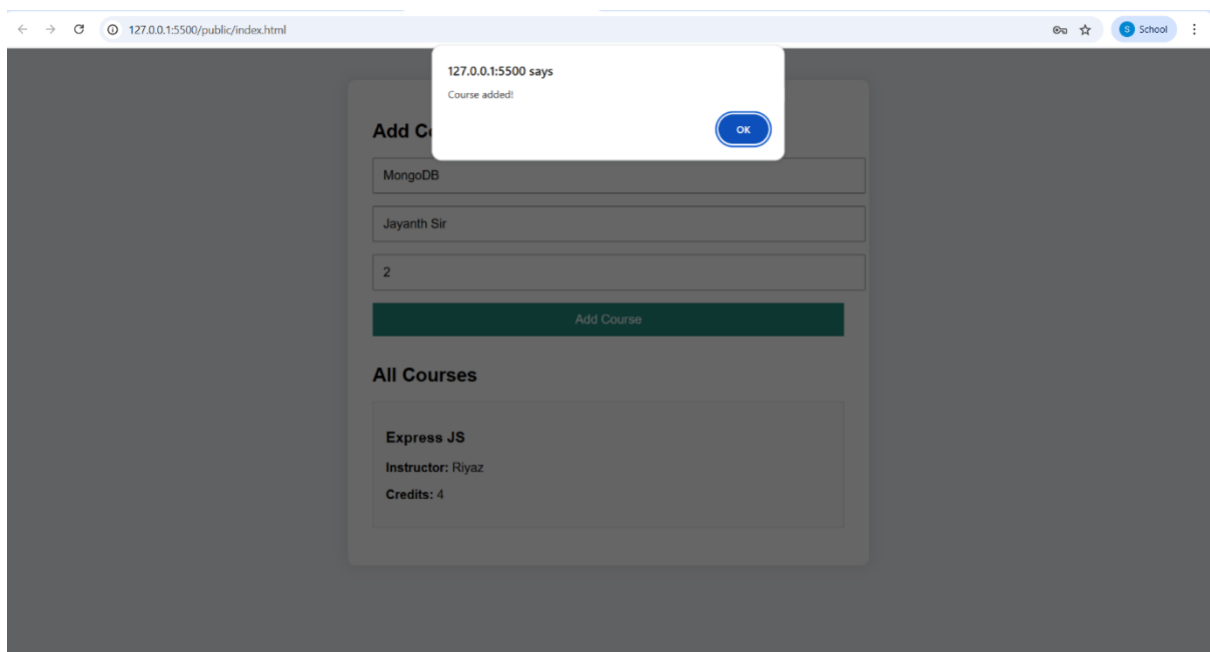
Add Course

**All Courses**

Express JS

Instructor: Riyaz

Credits: 4



A screenshot of the same web browser showing the "Add Course" form. A white modal box is overlaid on top of the form, displaying a success message: "127.0.0.1:5500 says Course added!". The modal box has an "OK" button. The form itself is dimmed in the background. The form fields now contain "MongoDB", "Jayanth Sir", and "2". The green "Add Course" button is still visible below the fields. The "All Courses" section is also visible below the button.

127.0.0.1:5500 says  
Course added!

OK

**Add Course**

MongoDB

Jayanth Sir

2

Add Course

**All Courses**

Express JS

Instructor: Riyaz

Credits: 4

**After Adding The course it shows the list of courses.**

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5500/public/index.html". The page has a light blue header with a "School" button. The main content area is divided into two sections: "Add Course" and "All Courses".

**Add Course**

This section contains a form with three input fields: "Course Title", "Instructor", and "Credits". Below the form is a green button labeled "Add Course".

**All Courses**

This section displays a list of courses. The first course is "Express JS" with an instructor of "Riyaz" and 4 credits. The second course is "MongoDB" with an instructor of "Jayanth Sir" and 2 credits.

Course Title	Instructor	Credits
Express JS	Riyaz	4
MongoDB	Jayanth Sir	2

**LOGIN & REGISTRATION IS DONE IN FRONTEND .**

## BACKEND LOGIN AND REGISTER THROUGH POSTMAN.

### REGISTER

The screenshot displays the Postman interface for a POST request to `http://localhost:5000/api/auth/register`. The request body is a JSON object with the following details:

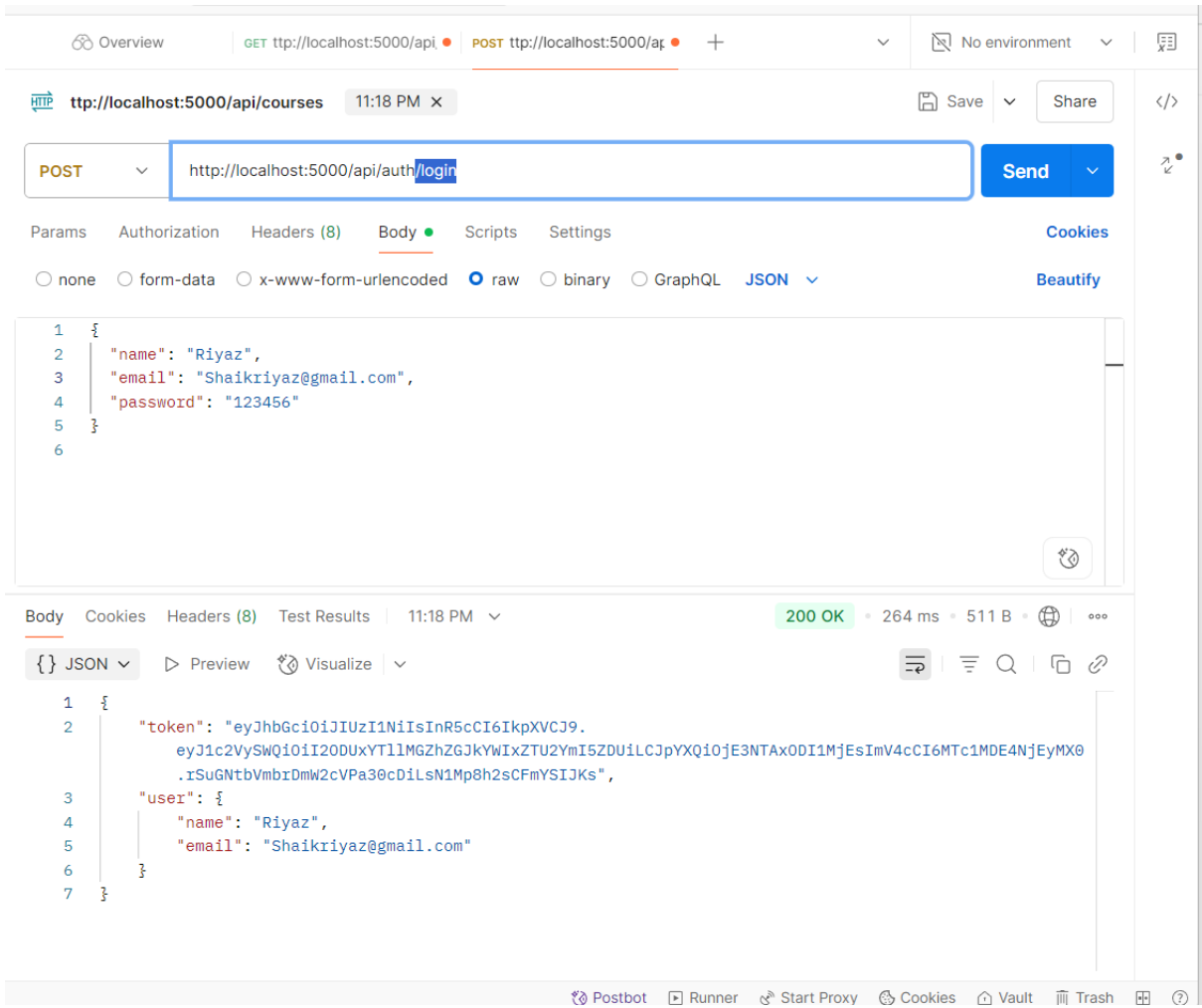
```
1 {
2   "name": "Riyaz",
3   "email": "Shaikriyaz@gmail.com",
4   "password": "123456"
5 }
6
```

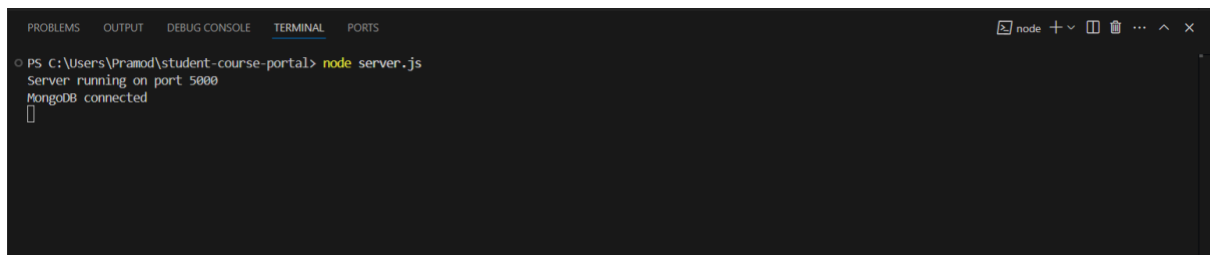
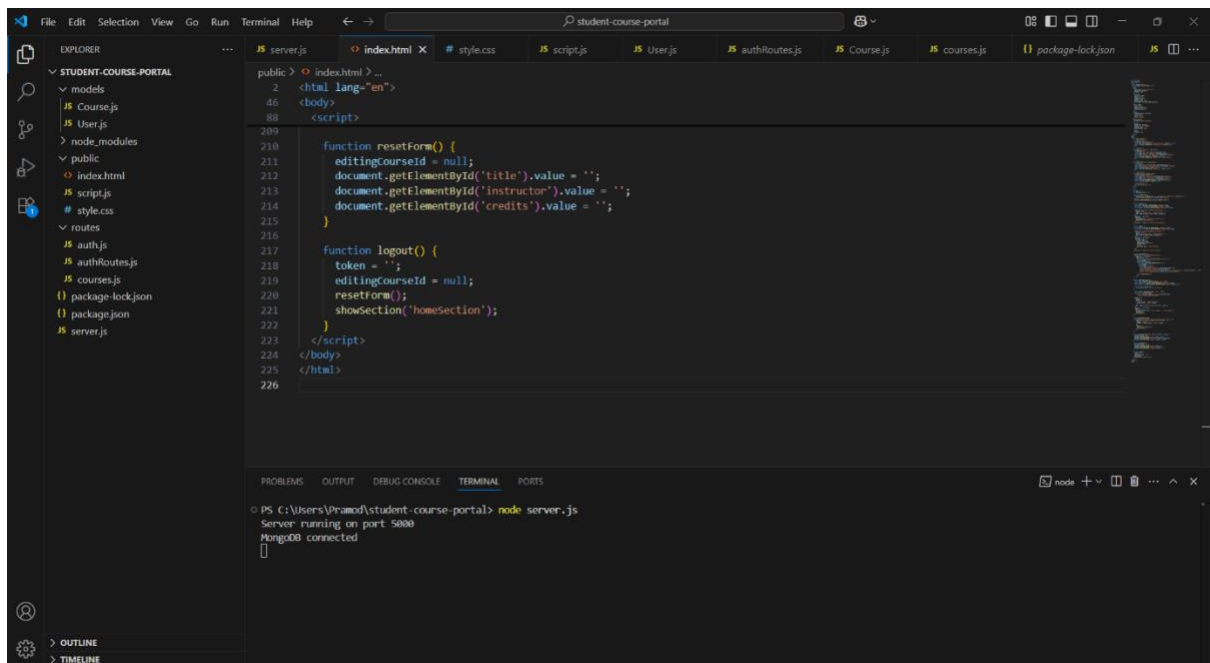
The response is a 201 Created status, indicating a successful registration. The response body is a JSON object:

```
1 {
2   "msg": "User registered successfully"
3 }
```

The interface also shows the request method as POST, the URL as `http://localhost:5000/api/auth/register`, and the response status as 201 Created with a 208 ms response time and 310 B of data. The response is displayed in the JSON format.

**LOGIN**





# CRUD OPERATIONS

## CREATE

The screenshot shows the Postman interface for a POST request to `http://localhost:5000/api/courses`. The request body is a JSON object representing a course. The response is a 201 Created status with a JSON object containing the created course details, including a new ID and version number.

**Request:**

```
POST http://localhost:5000/api/courses
```

**Body (raw):**

```
1 {
2   "title": "MongoDB",
3   "instructor": "Jayanth Sir",
4   "credits": 2
5 }
6
```

**Response:**

```
1 {
2   "title": "MongoDB",
3   "instructor": "Jayanth Sir",
4   "credits": 2,
5   "_id": "6851ab1efadb1e56bb9dd",
6   "__v": 0
7 }
```

**Status:** 201 Created • 16 ms • 371 B



## 2.READ

The screenshot shows the Postman interface with a GET request to `http://localhost:5000/api/courses` at 11:22 PM. The request is in the "Body" tab, and the response is a 200 OK status with a 17 ms response time and 567 B of data. The response body is displayed in JSON format, showing a list of two courses.

**Request:**

- Method: GET
- URL: `http://localhost:5000/api/courses`
- Body: `Ctrl+Alt+P for Postbot`

**Response:**

```
{
  "courses": [
    {
      "_id": "684f10c44cd8d5dd6ab4966c",
      "title": "Express JS",
      "instructor": "Riyaz",
      "credits": 4,
      "__v": 0
    },
    {
      "_id": "6851a87bfadbdab1e56bb9d0",
      "title": "MongoDB",
      "instructor": "Jayanth Sir",
      "credits": 2
    }
  ]
}
```

The bottom of the interface shows a toolbar with icons for Postbot, Runner, Start Proxy, Cookies, Vault, Trash, and a help icon.

### 3.DELETE

Overview

DEL http://localhost:5000/api/courses<684f10c44cd8d5dd6ab4966c>

No environment

HTTP

http://localhost:5000/api/courses<684f10c44cd8d5dd6ab4966c>

Save

Share

</>

DELETE

http://localhost:5000/api/courses<684f10c44cd8d5dd6ab4966c>

Send

Params

Authorization

Headers (8)

Body

Scripts

Settings

Cookies

Query Params

	Key	Value	Description	...	Bulk Edit
	Key	Value	Description		

Body

Cookies

Headers (8)

Test Results

200 OK

73 ms

567 B

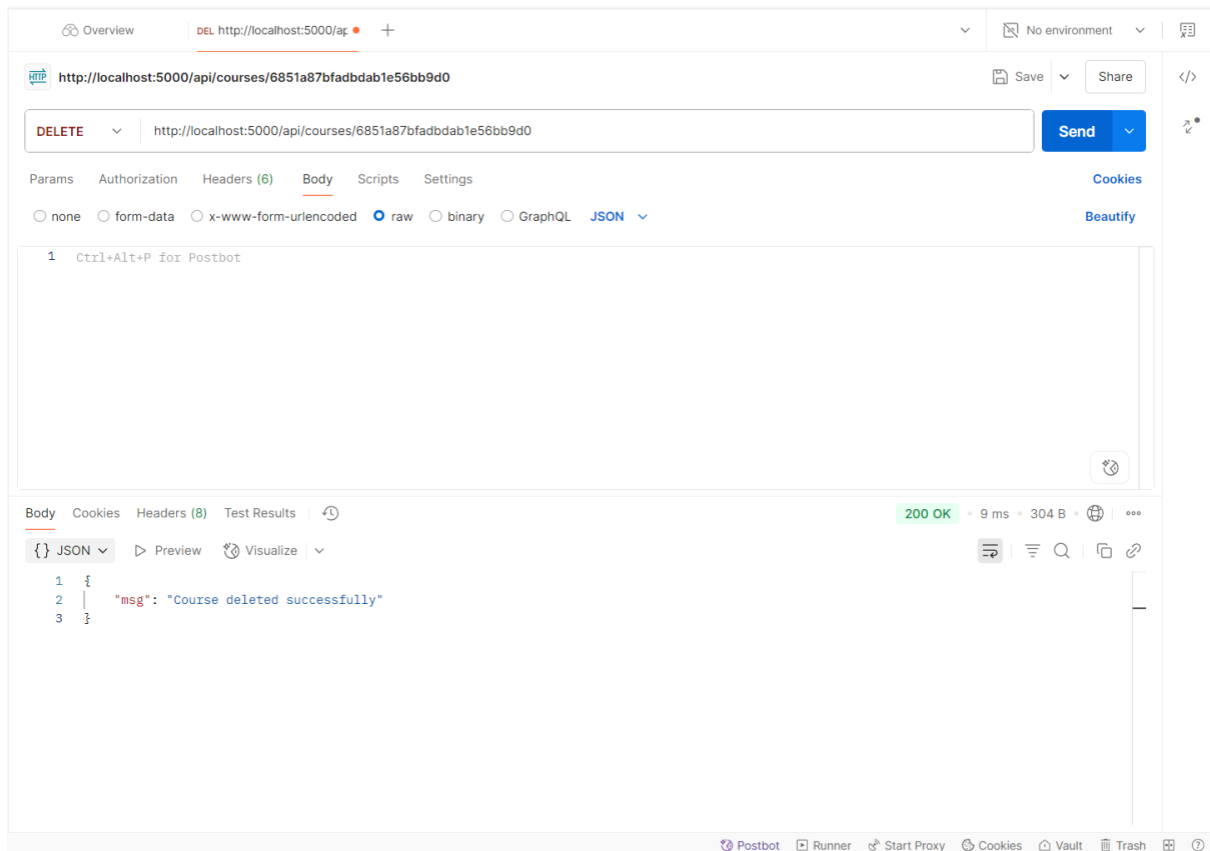
...

{ } JSON

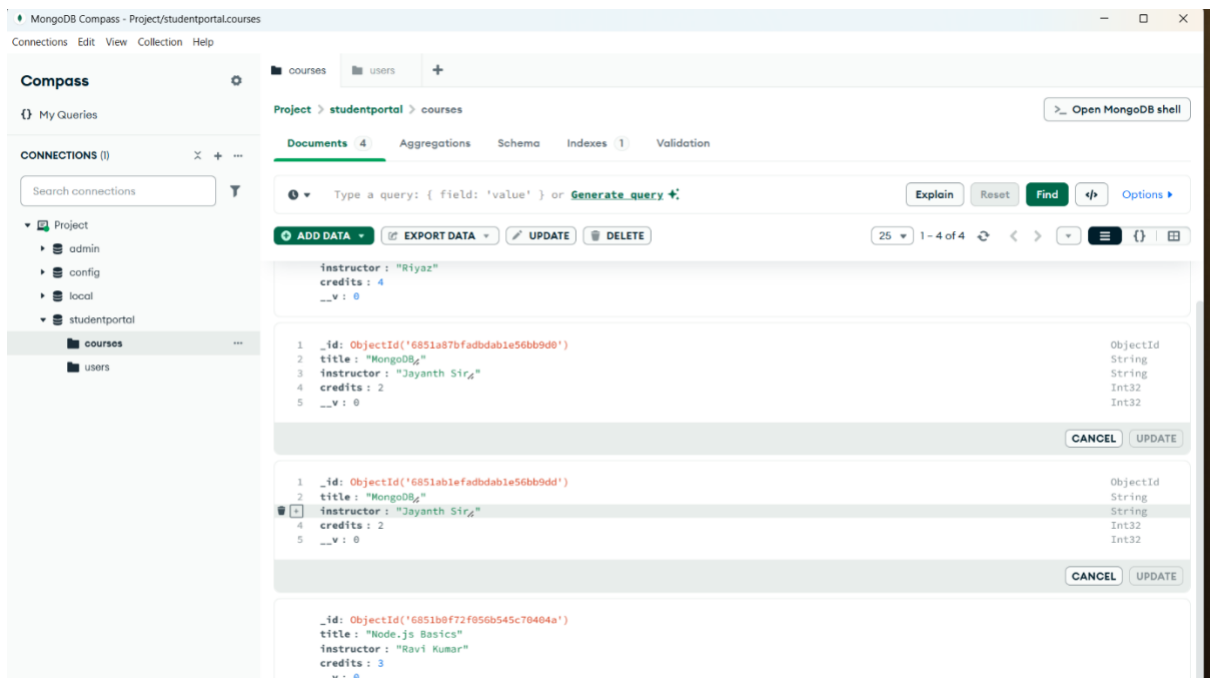
Preview

Visualize

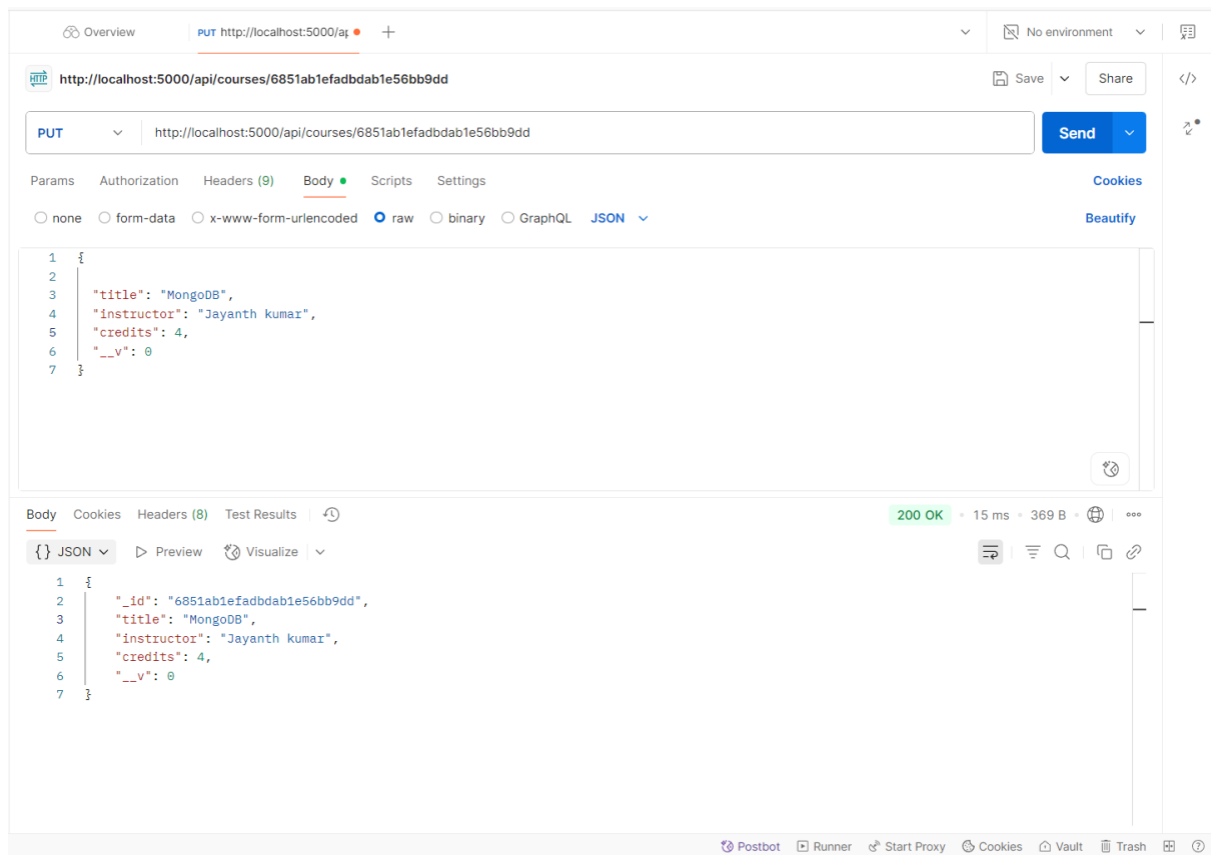
```
1  [
2    {
3      "_id": "684f10c44cd8d5dd6ab4966c",
4      "title": "Express JS",
5      "instructor": "Riyaz",
6      "credits": 4,
7      "__v": 0
8    },
9    {
10     "_id": "6851a87bfadbdab1e56bb9d0",
11     "title": "MongoDB",
12     "instructor": "Javanth Sir".
```



## Title:- Express JS is deleted



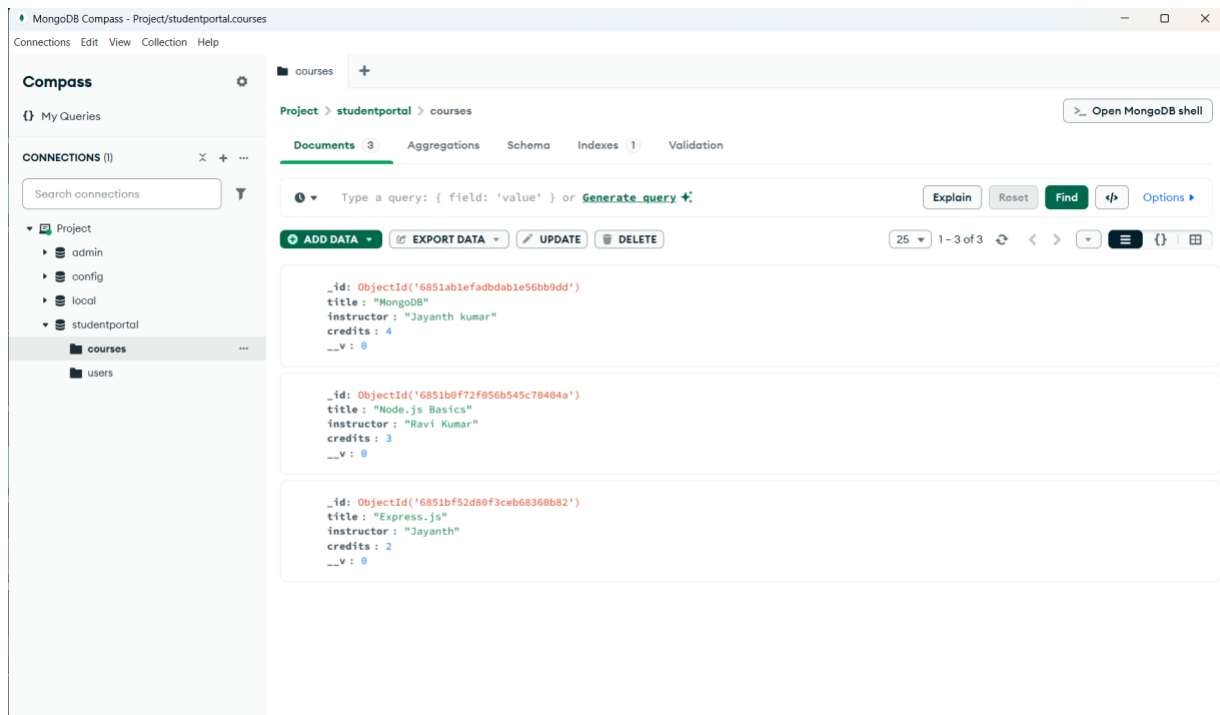
## UPDATE



HERE UPDATE OPERATION IS USED.

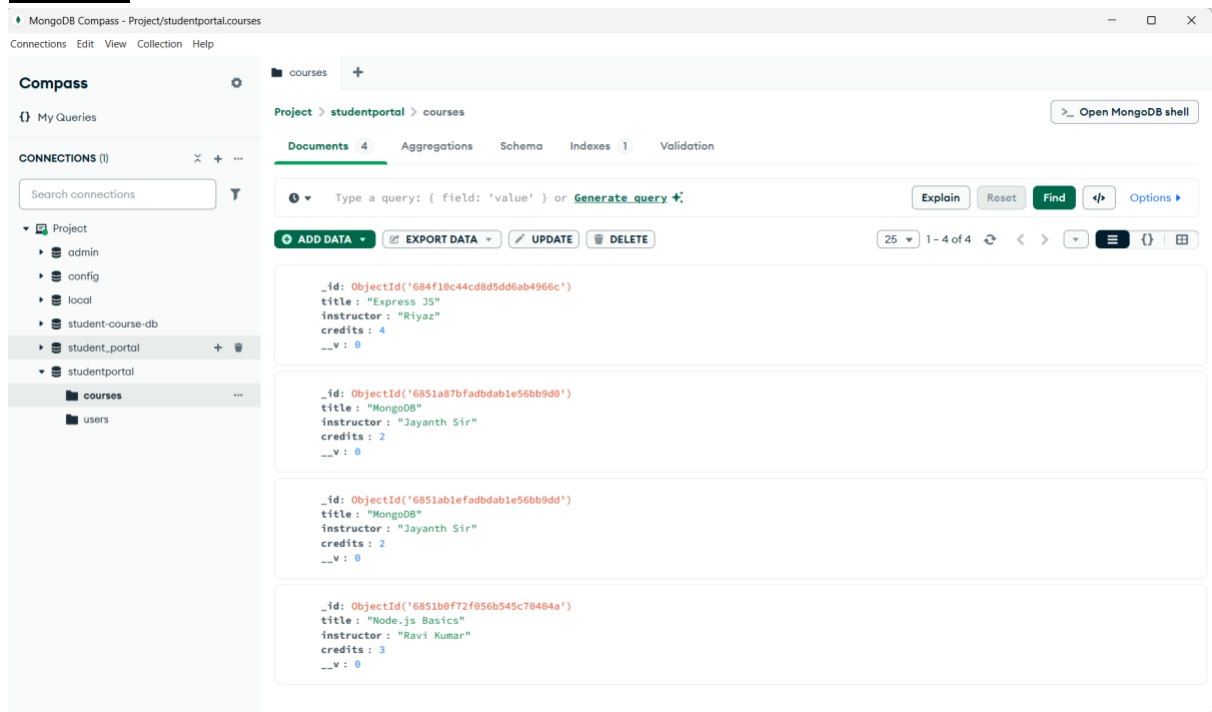
Instructor name is updated.

## After Updating And Deleting Operations



## BEFORE AND AFTER CRUD OPERATIONS DATA IN MONGODB

### BEFORE



MongoDB Compass - Project/studentportal.courses

Connections Edit View Collection Help

### Compass

{ My Queries

CONNECTIONS (1)

Search connections

- Project
  - admin
  - config
  - local
  - studentportal
    - courses**
    - users

Project > studentportal > courses

Documents 4 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1 - 4 of 4

```
{
  instructor: "Riyaz"
  credits: 4
  __v: 0
}
```

1 \_id: ObjectId('6851a87bfadbdb1e56bb9d0') ObjectId  
2 title: "MongoDB" String  
3 instructor: "Jayanth Sir" String  
4 credits: 2 Int32  
5 \_\_v: 0 Int32

CANCEL UPDATE

```
{
  _id: ObjectId('6851a87bfadbdb1e56bb9d0')
  title: "MongoDB"
  instructor: "Jayanth Sir"
  credits: 2
  __v: 0
}
```

CANCEL UPDATE

```
{
  _id: ObjectId('6851b0f72f056b545c70404a')
  title: "Node.js Basics"
  instructor: "Ravi Kumar"
  credits: 3
  __v: 0
}
```

MongoDB Compass - Project/studentportal.courses

Connections Edit View Collection Help

### Compass

{ My Queries

CONNECTIONS (1)

Search connections

- Project
  - admin
  - config
  - local
  - studentportal
    - courses**
    - users

Project > studentportal > courses

Documents 4 Aggregations Schema Indexes 1 Validation

Type a query: { field: 'value' } or [Generate query](#)

Explain Reset Find Options

ADD DATA EXPORT DATA UPDATE DELETE

25 1 - 4 of 4

```
{
  _id: ObjectId('684f10c44cd8d5dd6ab4966c')
  title: "Express JS"
  instructor: "Riyaz"
  credits: 4
  __v: 0
}
```

1 \_id: ObjectId('6851a87bfadbdb1e56bb9d0') ObjectId  
2 title: "MongoDB" String  
3 instructor: "Jayanth Sir" String  
4 credits: 2 Int32  
5 \_\_v: 0 Int32

CANCEL UPDATE

```
{
  _id: ObjectId('6851a87bfadbdb1e56bb9d0')
  title: "MongoDB"
  instructor: "Jayanth Sir"
  credits: 2
  __v: 0
}
```

CANCEL UPDATE

```
{
  _id: ObjectId('6851b0f72f056b545c70404a')
  title: "Node.js Basics"
}
```

## AFTER CRUD OPERATIONS

