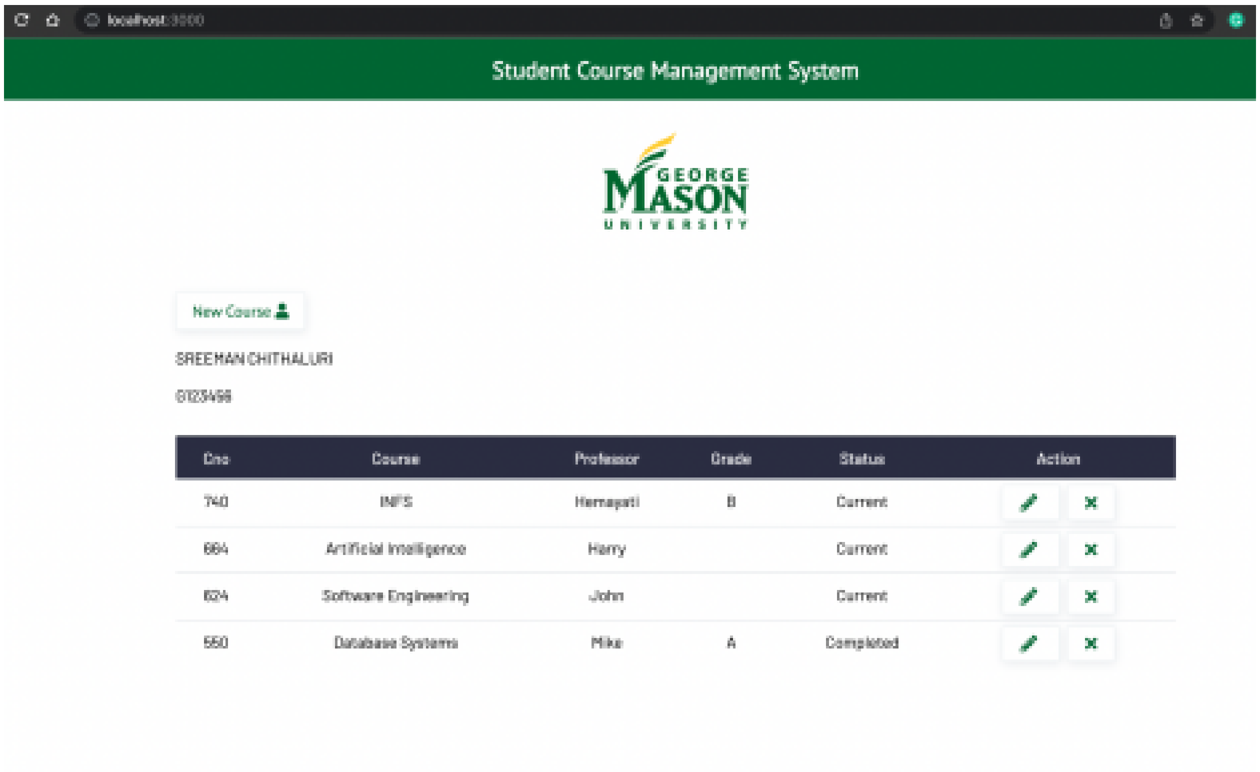
George Mason University

STUDENT COURSE MANAGEMENT SYSTEM

Description

The project I developed is a student course management system where it is useful for an individual student. This website is from the point of view of an individual college student who can manage his courses that he is taking throughout his college. In this project, I have used Node js for backend, Ejs for front end and Mongodb for database and mongodb has 3 collections(Userdb, Student, Faculty). This is a simple website with all 3 functionality like Insert, update and delete. I have used MVC Architecture.



SREEMAN CHITHALURI

G123456

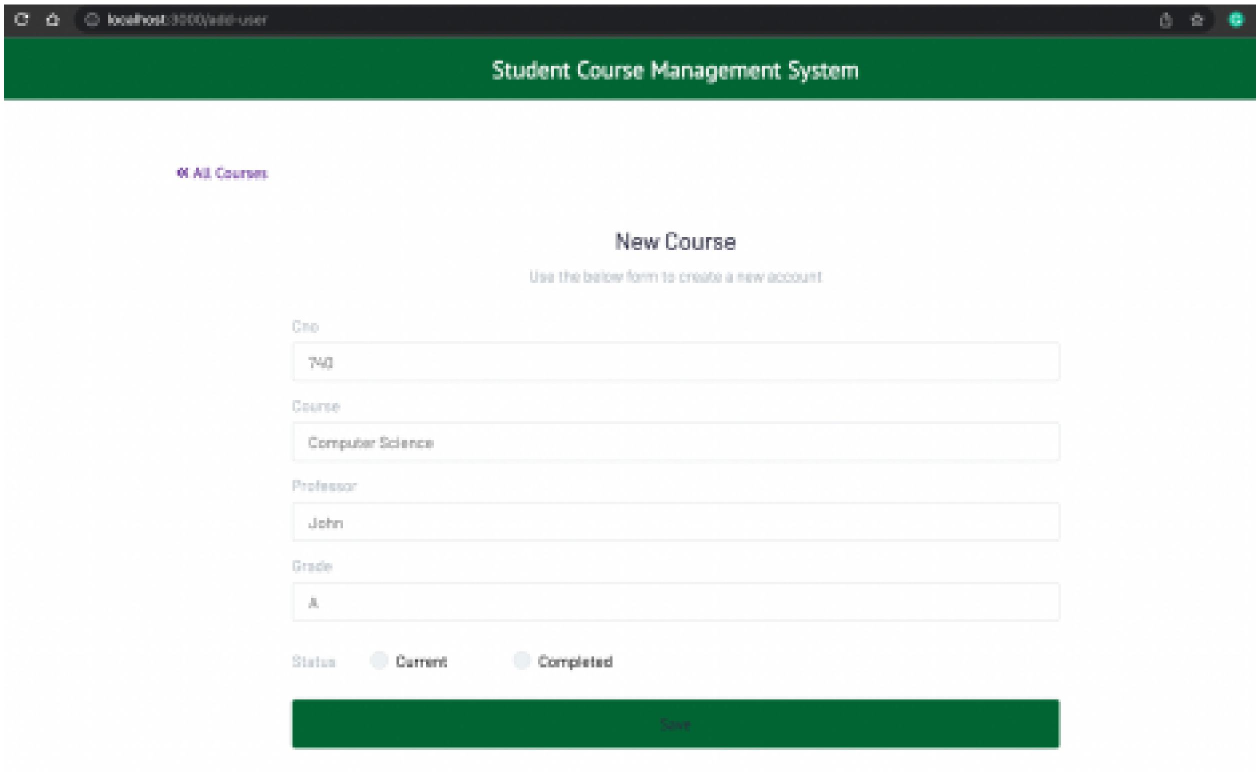
Cno	Course	Professor	Grade	Status	Action
740	INFS	Hemapati	B	Current	✎ ✕
664	Artificial Intelligence	Harry		Current	✎ ✕
624	Software Engineering	John		Current	✎ ✕
550	Database Systems	Mike	A	Completed	✎ ✕

The first page shows all the courses that an individual student has entered. Each course has attributes like Course Number, Course Name, Professor teaching that course, Grade that student received and status of each course like whether it is current course or completed course. For this website, I have used the theme of our college website.

Functionality

1 New Course Button:

In this project, I have developed a button named "New Course", whenever this button is clicked, a new form will be opened where a student can enter his course details. Whenever we hit the save button the course record will be stored into the mongodb database.



localhost:3000/add-user

Student Course Management System

« All Courses

New Course

Use the below form to create a new account

Cno
740

Course
Computer Science

Professor
John

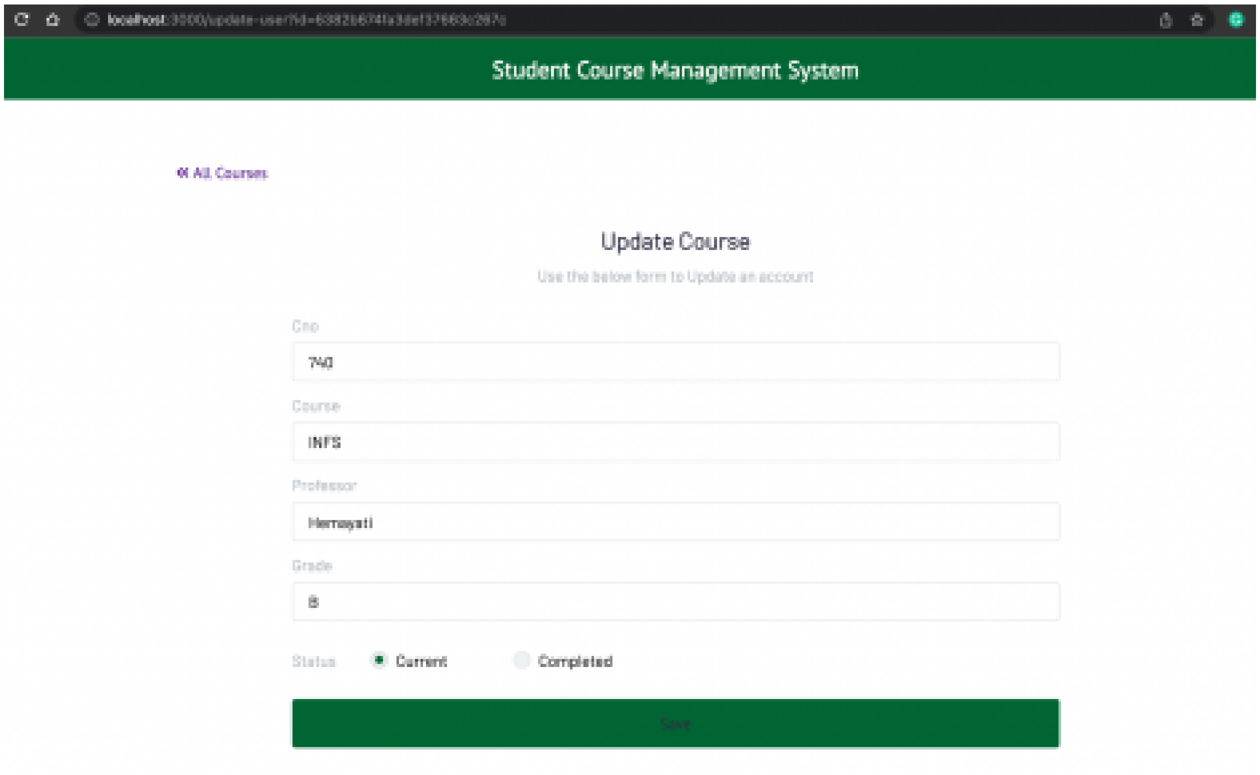
Grade
A

Status ○ Current ○ Completed

Save

2. All Courses Button:
After Data Insertion, Whenever we click the “All Courses” button, we are redirected to the home page where the inserted data can be viewed at this place.

3. Edit Button:
If a student wants to edit an existing record, after clicking the edit button, based on Id, the record will be searched in the mongodb database and can be updated.



localhost:3000/update-user?id=6382b674fa3def37663c287c

Student Course Management System

« All Courses

Update Course

Use the below form to Update an account

Cno
740

Course
INFS

Professor
Hemayati

Grade
B

Status ● Current ○ Completed

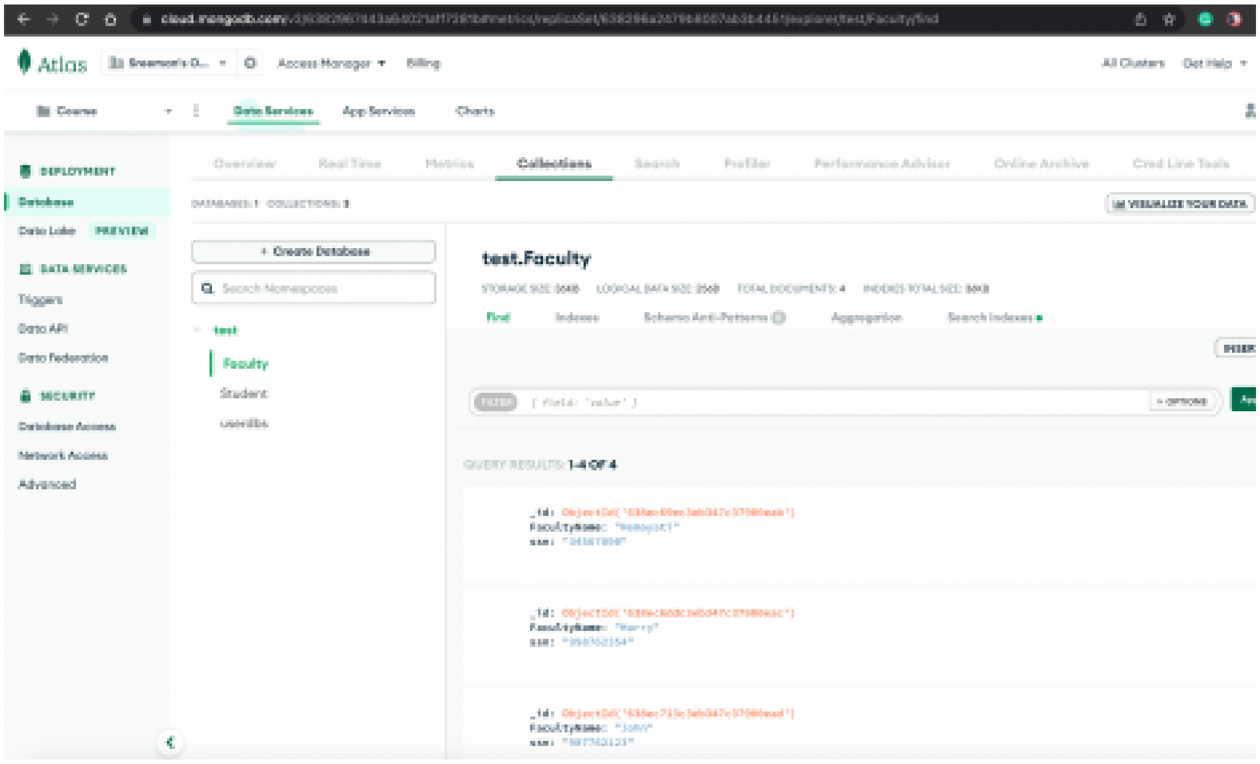
Save

4. Delete Button:
This delete button will pull up the data record from the database using ID and erase the

record.

Collections in Mongodb

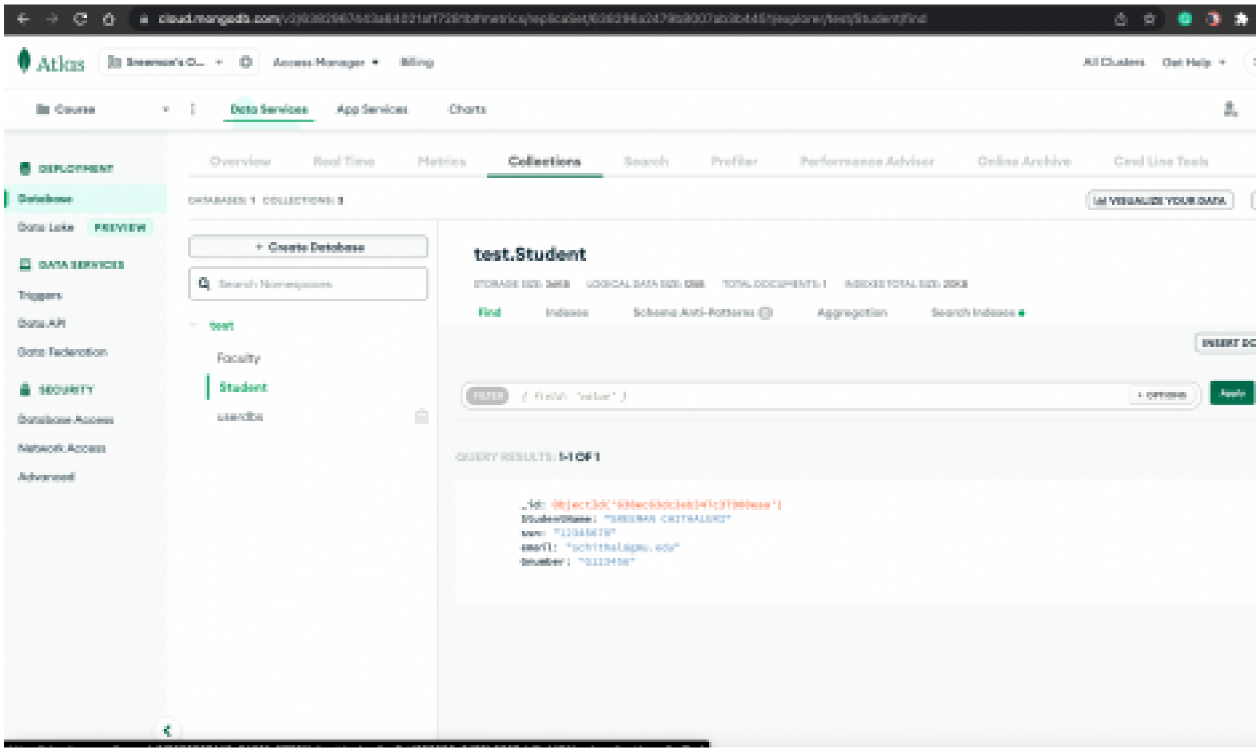
I have used 3 collections in the database: userdb(course), Faculty, Student.



test.Faculty

QUERY RESULTS: 1-4 OF 4

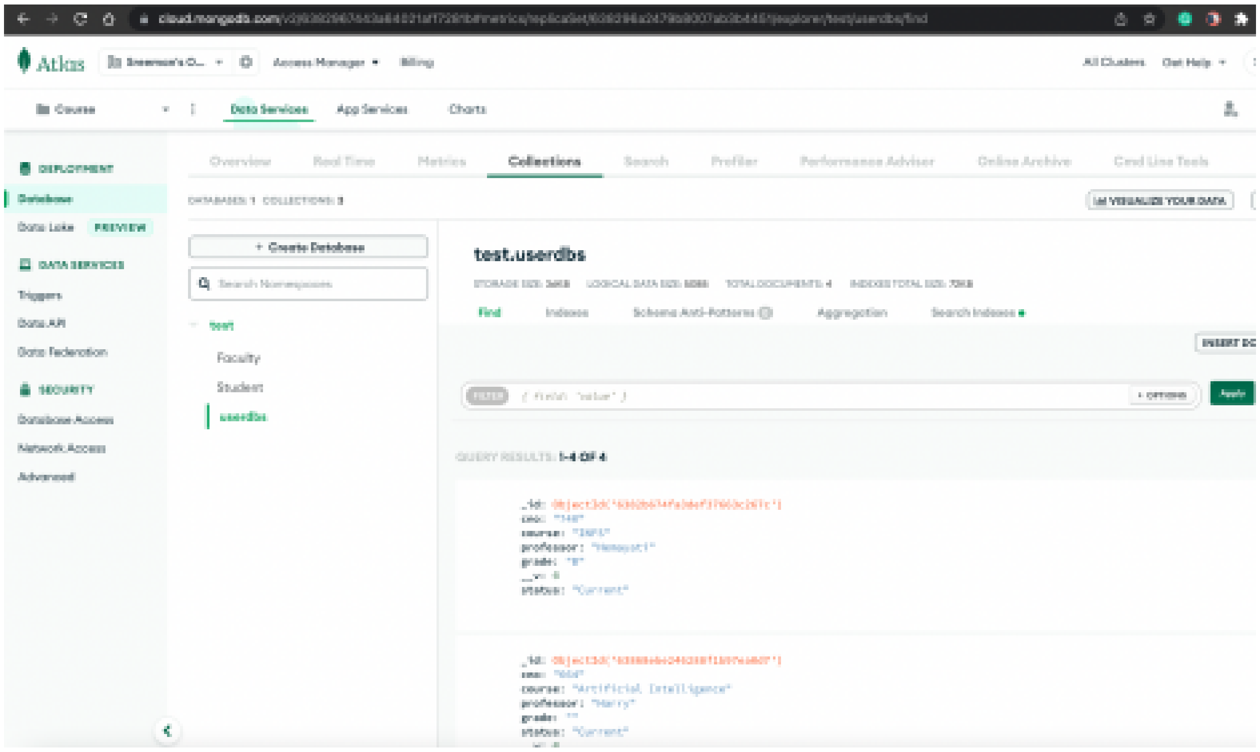
```
_id: ObjectId('[illegible]')
FacultyName: "[illegible]"
ssn: "[illegible]"

_id: ObjectId('[illegible]')
FacultyName: "Harry"
ssn: "[illegible]"

_id: ObjectId('[illegible]')
FacultyName: "John"
ssn: "[illegible]"
```



test.Student

QUERY RESULTS: 1-1 OF 1

```
_id: ObjectId('[illegible]')
StudentName: "[illegible]"
ssn: "[illegible]"
email: "[illegible]"
Gnumber: "[illegible]"
```


test.userdbs

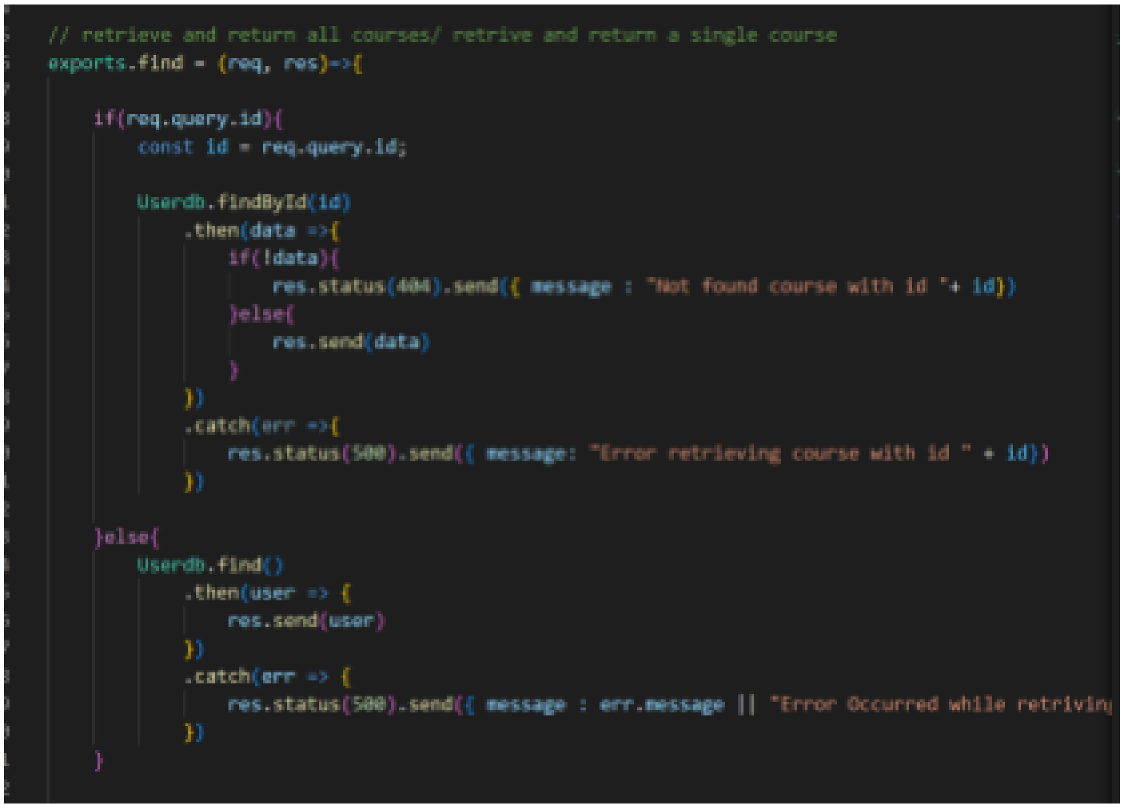
QUERY RESULTS: 1-4 OF 4

```
_id: ObjectId('[illegible]')
cno: "140"
course: "INFS"
professor: "Hemayati"
grade: "B"
__v: 0
status: "Current"
```

```
_id: ObjectId('[illegible]')
cno: "604"
course: "Artificial Intelligence"
professor: "Harry"
grade: ""
status: "Current"
```

Search Functionality in code:

Based on course ID and find function, I have searched for a particular record in Mongdb, retrieved and updated the same record after modifying.



```
// retrieve and return all courses/ retrive and return a single course
exports.find = (req, res)=>{

    if(req.query.id){
        const id = req.query.id;

        Userdb.findById(id)
            .then(data =>{
                if(!data){
                    res.status(404).send({ message : "Not found course with id "+ id})
                }else{
                    res.send(data)
                }
            })
            .catch(err =>{
                res.status(500).send({ message: "Error retrieving course with id " + id})
            })

    }else{
        Userdb.find()
            .then(user => {
                res.send(user)
            })
            .catch(err => {
                res.status(500).send({ message : err.message || "Error Occurred while retrivin
            })
    }
```