Name : Prathamesh Mandave

Task : Both (Frontend and Backend)

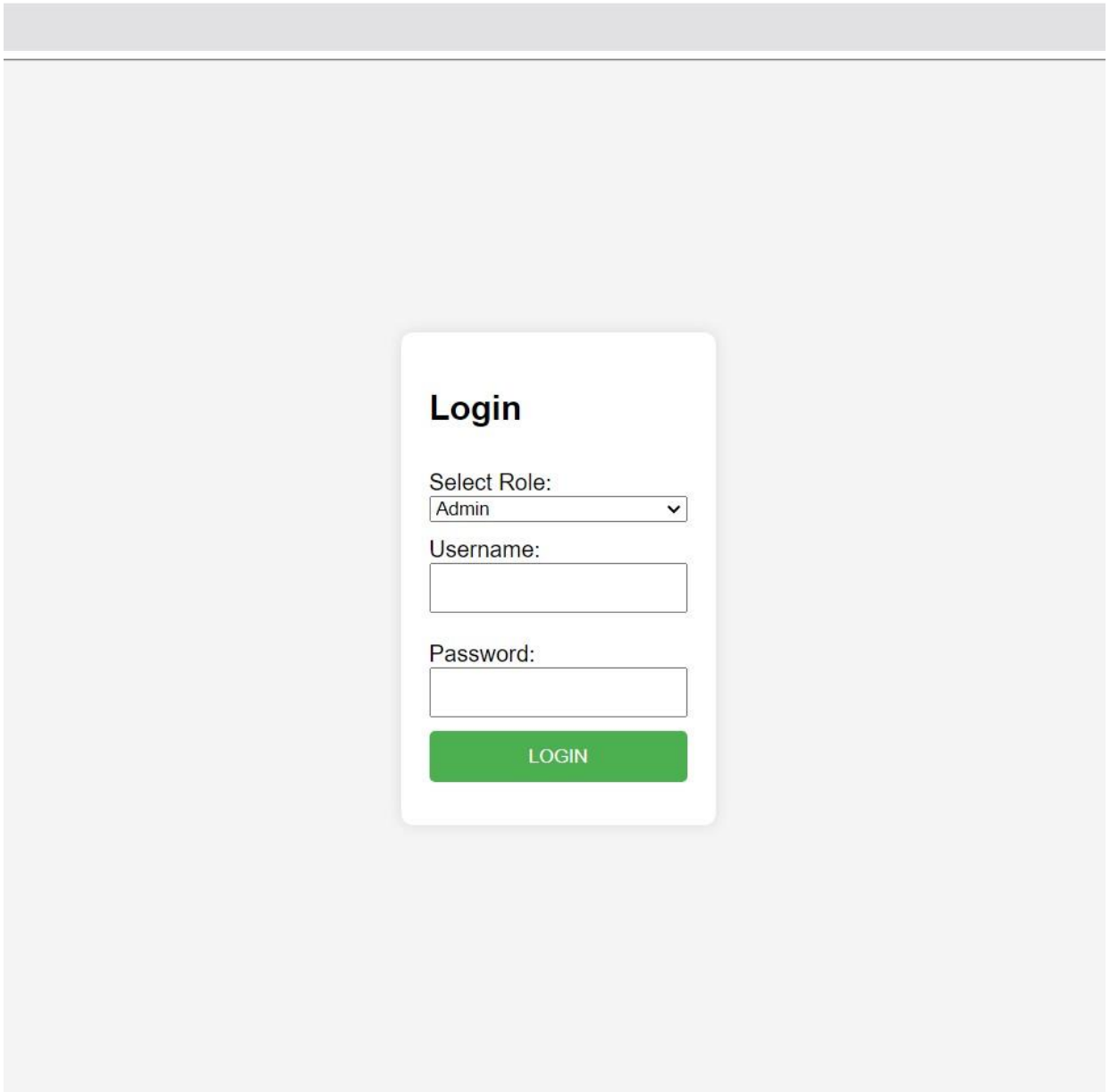
Choosed Task : Second(BackEnd Task)

Technologies used:

Frontend: HTML, CSS, Javascript.

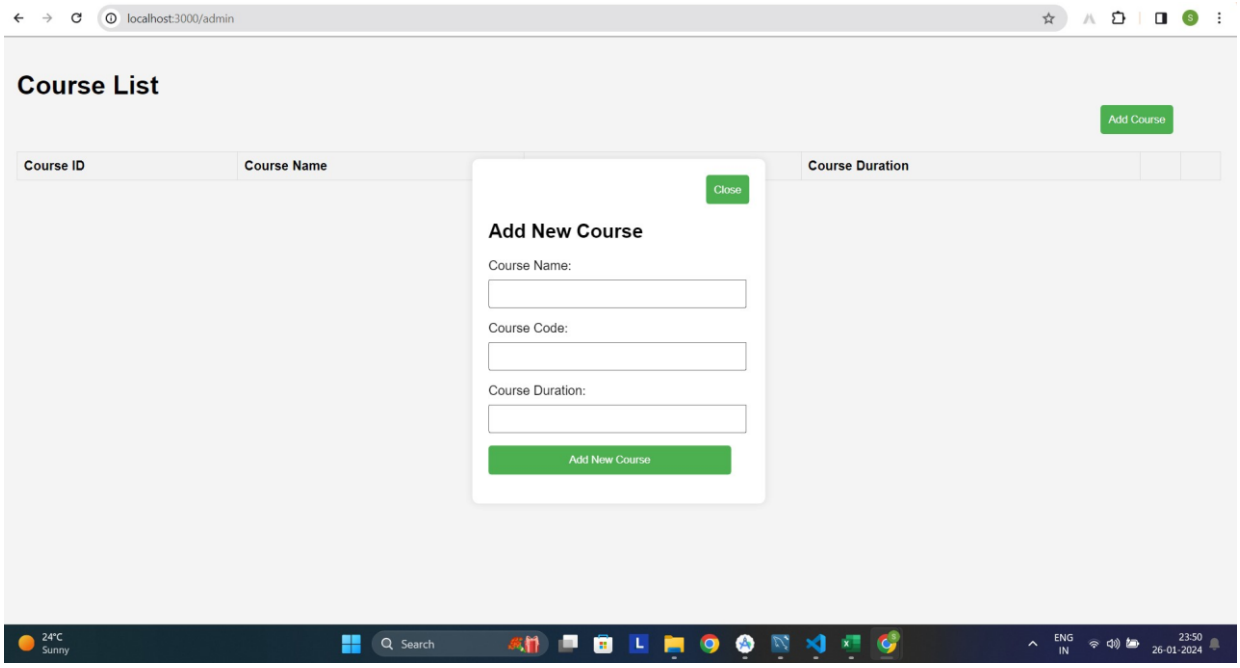
Backend: NodeJs, ExpressJs and MySQL

Admin Portal:

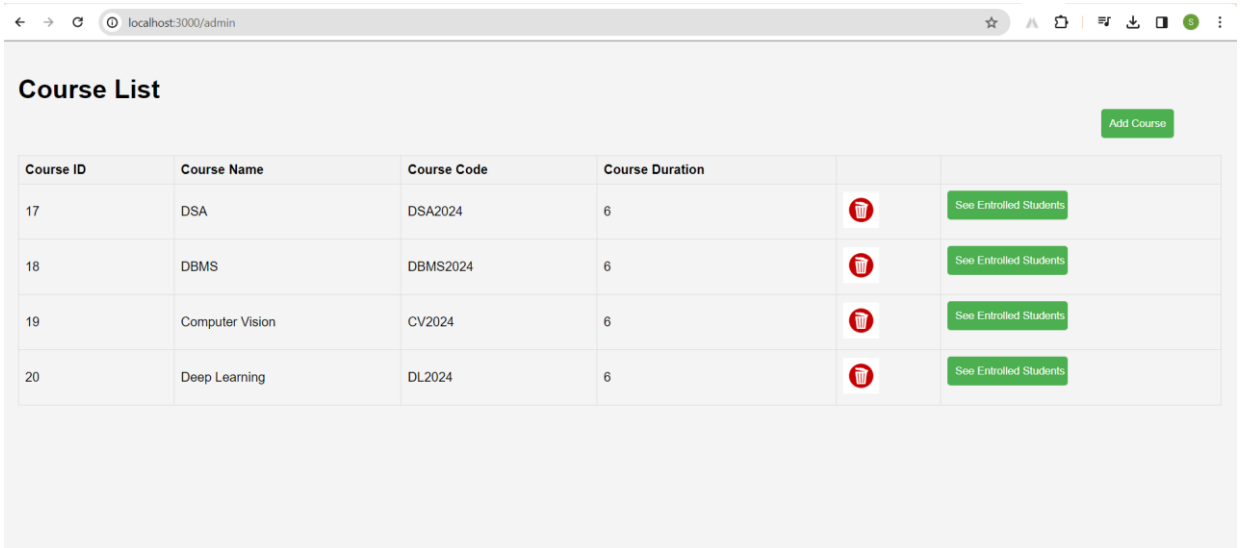


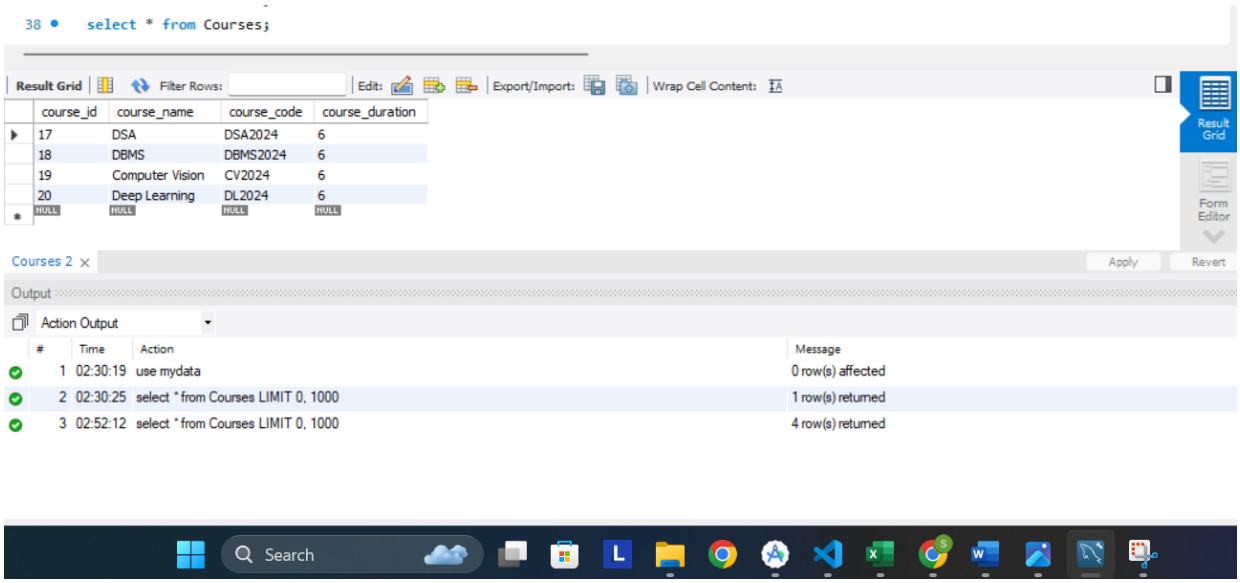
Admin Dashboard:

1.Admin can able to add courses from admin portal and added courses are saved to database as well as displayed on portal.

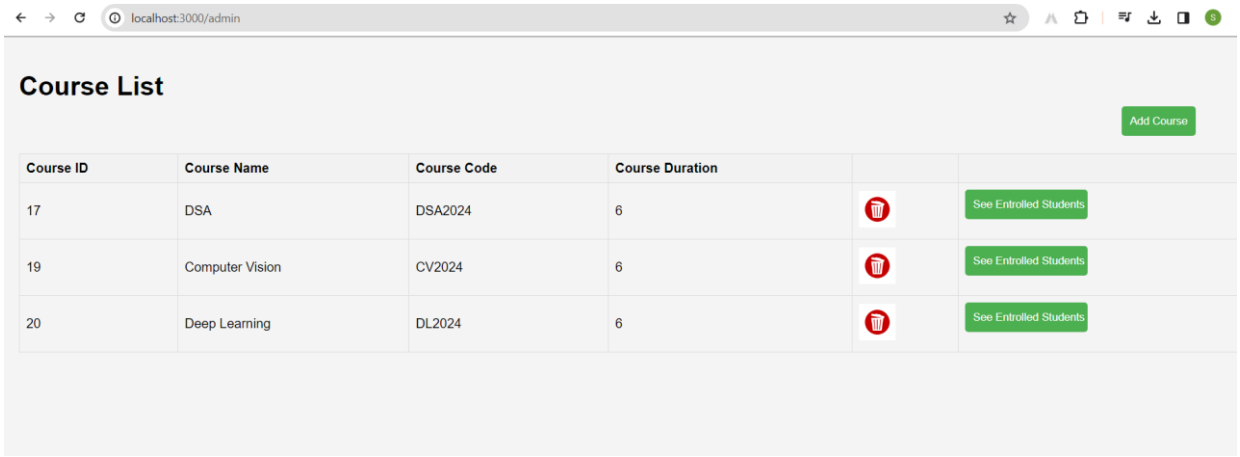


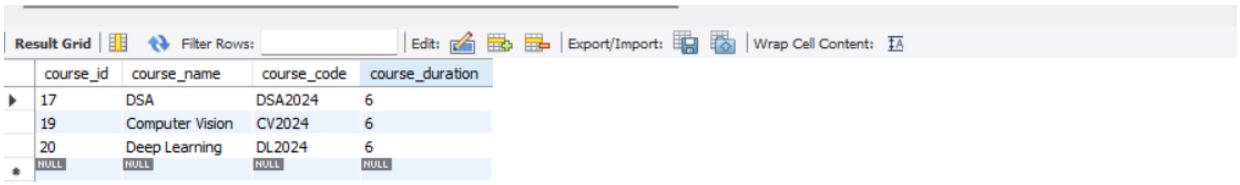
2.Added courses are displayed on portal as well as saved in Courses Table in Mysql Database.

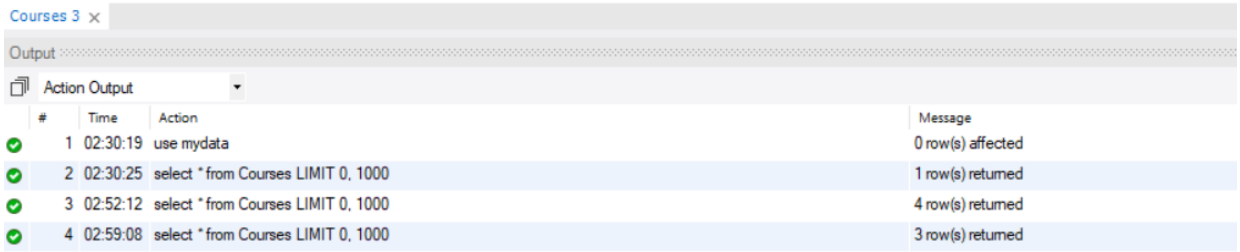




3.Admin can delete any course from Admin portal and according to that same delete operations reflect into Mysql database also ON DELETE CASCADE works properly.For eg now admin delete course with couse id 18 by clicking on delete icon.





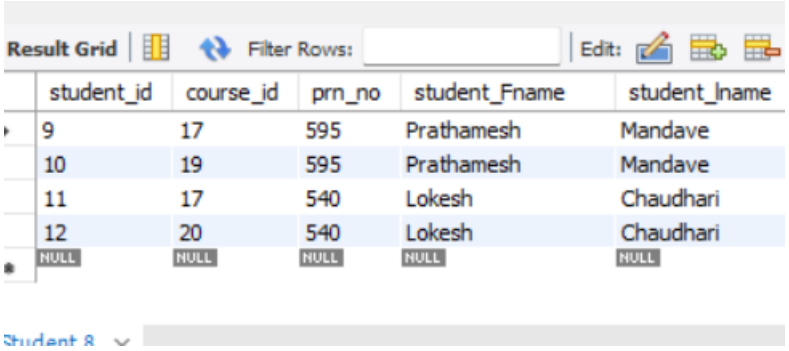


4.Admin can see all students who enrolled in specific course by clicking on ‘see enrolled students’ button which is infront of delete icon.

In below screen shot two user enroll in a course with course id=17 Given result is when admin click on button associated with course id=17.

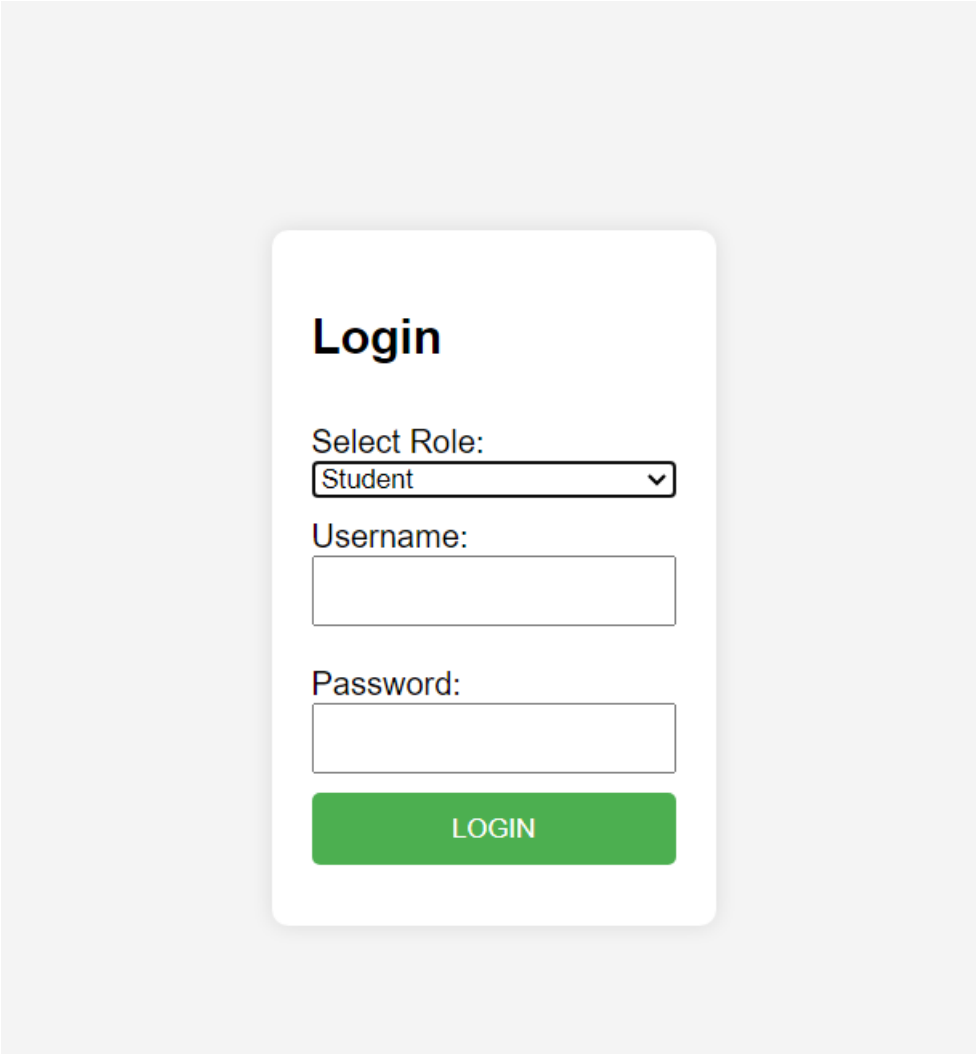
First Name	Last Name	PRN No	Course Id
Prathamesh	Mandave	595	17
Lokesh	Chaudhari	540	17

Enrolled students data:



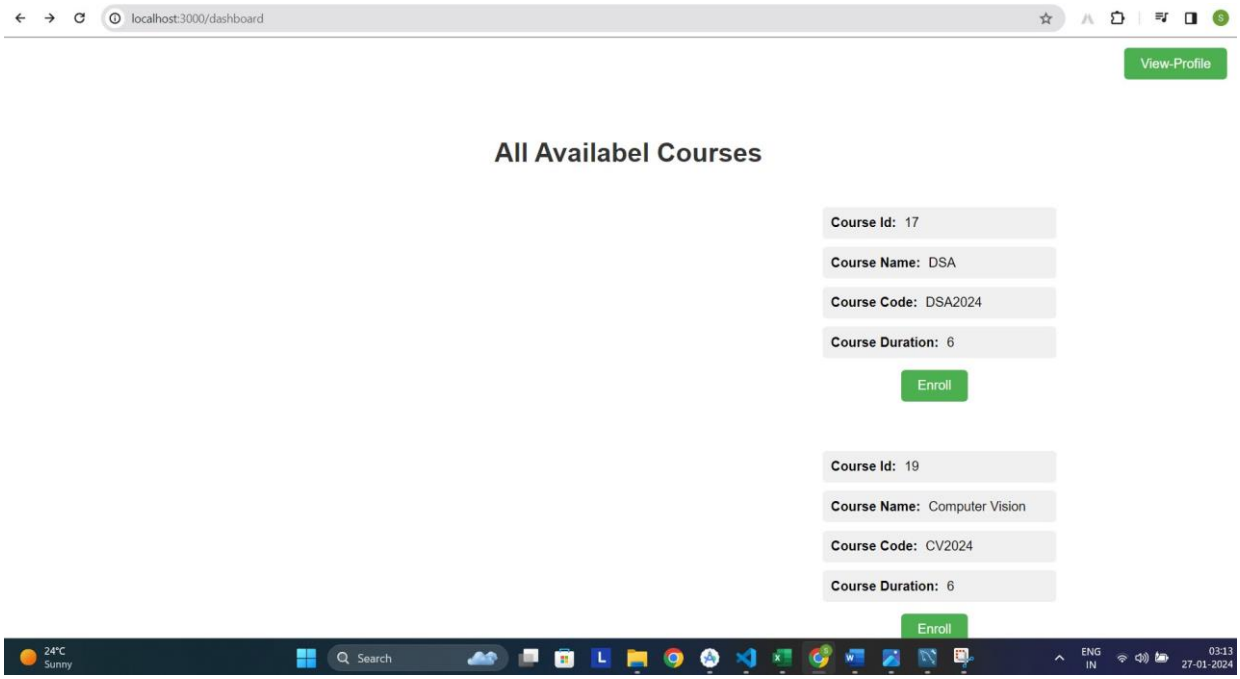
student_id	course_id	prn_no	student_Fname	student_lname
9	17	595	Prathamesh	Mandave
10	19	595	Prathamesh	Mandave
11	17	540	Lokesh	Chaudhari
12	20	540	Lokesh	Chaudhari
NULL	NULL	NULL	NULL	NULL

Student Portal:

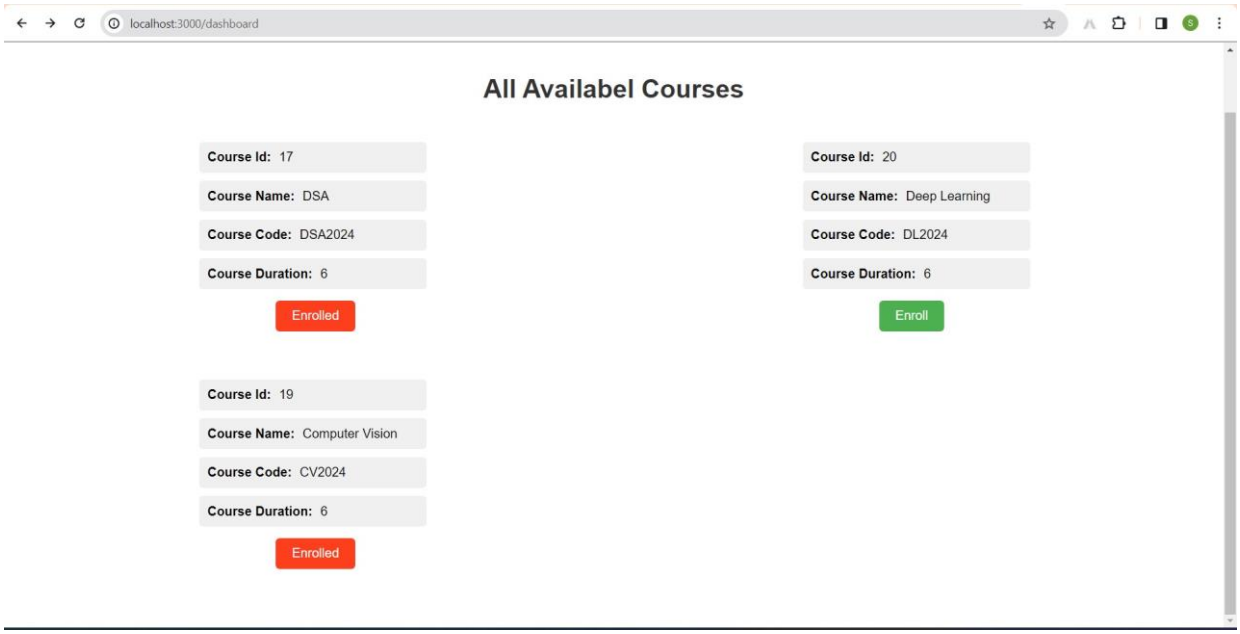


Student Dashboard:

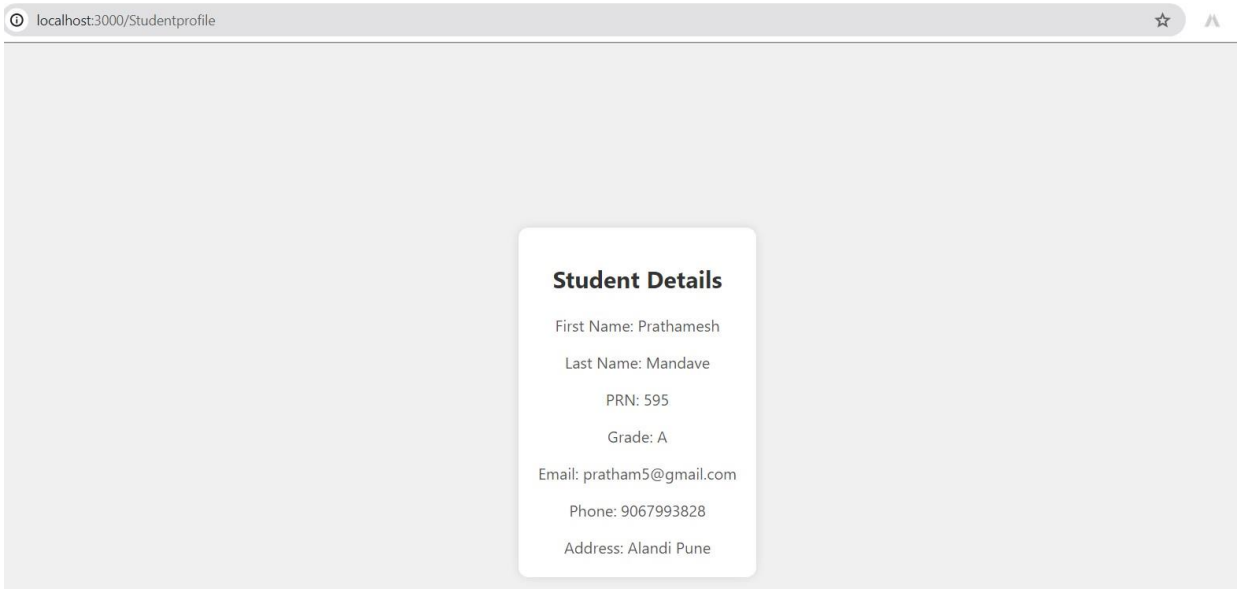
1.All available courses are displayed on portal from database before user's enrollment



2.All enrolled and unenrolled courses are displayed on portal and changes are reflect into database according to student's enrollment



3.Student can view their details by clicking on ‘view profile’ button and and all information fetched from database and displayed on page.



Server Programming in NodeJS :

```
const express = require('express');
const path = require('path');
const app = express();
const port = 3000; // Set your desired port number
const mysql = require('mysql');

const bodyParser = require("body-parser");
const encoder = bodyParser.urlencoded({ extended: true });

app.use(express.json());
app.use("/Assets",express.static("Assets"));

// Set EJS as the view engine
app.set('view engine', 'ejs');

const connection = mysql.createConnection({ host: 'localhost', // host for
connection
port: 3306, // default port for mysql is 3306
database: 'mydata', // database from which we want to connect out node
application
user: 'root', // username of the mysql connection
password: 'Pratham@123' });

// Connect to the database
connection.connect((err) => {
  if (err) {
      console.error('Error connecting to MySQL:', err);
  } else {
      console.log('Connected to MySQL successfully');
  }
});

app.get('/', (req, res) => {
  res.sendFile(path.join(__dirname, 'login.html'));
});

app.post('/',encoder, (req, res) => {


  const role = req.body.role;

    // Check the value of the "role" parameter
    if (role === 'admin') {
```

```
        // Handle logic for admin
        res.redirect("/admin");
    } else if (role === 'student') {
        // Handle logic for student
        res.redirect("/dashboard");
    } else {
        // Handle other cases or provide an error response
        res.send('Invalid role');
    }
});

app.get('/admin', (req, res) => {

       const selectQuery = 'SELECT * FROM Courses';

       connection.query(selectQuery, function (err, results, fields) {
           if (err) {
               console.error('Error occurred while retrieving data:', err);
           } else {
               // Send the results to the frontend (assuming this is an API
endpoint)
               // You can choose the appropriate way to send data to your
frontend, such as using Express to send JSON response
               console.log('Data retrieved successfully!');
               console.log('Results:', results);

               // Example of sending JSON response using Express (you might
need to install 'express' package)
               //res.json(results);
               //res.sendFile(path.join(__dirname, 'Admin.html'));

               res.render('index', { courses: results });

               // If you're not using Express
    }
  });
});

app.get('/dashboard', function (req, res) {
  const prn="595";
  const selectQuery = 'SELECT * FROM Courses WHERE course_id NOT IN (SELECT
course_id FROM Student WHERE prn_no = ?)';
      connection.query(selectQuery, [prn],function (err, unenrollCourses,
fields) {
        if (err) {
            console.error('Error occurred while retrieving data:', err);
        } else {
            // Send the results to the frontend (assuming this is an API
endpoint)
            // You can choose the appropriate way to send data to your
frontend, such as using Express to send JSON response
            console.log('Data retrieved successfully!');
```

```
            console.log('Results:', unenrollCourses);

            const selectUserCoursesQuery1 = 'SELECT Courses.* ' +
    'FROM Student ' +
    'INNER JOIN Courses ON Student.course_id = Courses.course_id ' +
    'WHERE Student.prn_no = ?';
            connection.query(selectUserCoursesQuery1, [prn], function (err,
userCourses, fields) {
                if (err) {
                    console.error('Error occurred while retrieving user
courses:', err);
                    res.status(500).send('Internal Server Error');
                } else {
                    // Send the courses and user's enrolled courses to the
frontend
                    res.render('dashboard', { unenrollCourses,userCourses });
                }
            });

 }
});

});

app.post('/dashboard', function (req, res) {
    res.redirect("/Studentprofile");
});

app.post('/admin', encoder ,(req, res) => {

     const action = req.body.action;

    if (action === 'seeEnrolledStudents') {
        // Handle the "See Enrolled Students" action
        // Access additional data using req.body.additionalData

        const courseId=req.body.additionalData;
        const query='select * from Student where course_id=?';
        connection.query(query, courseId, function (err, results, fields) {
          if (err) {
            console.error('Error occurred while inserting data:', err);
          } else {
            console.log('Data fetch successfully!');
            res.render('enrollment',{data: results});
          }});
    } else if (action === 'addNewCourse') {

      const courseName=req.body.courseName;
  const courseCode=req.body.courseCode;
  const courseDuration=req.body.courseDuration;

  console.log(courseName, courseCode, courseDuration);
```

```
  // Insert data into a table
   const insertQuery = 'INSERT INTO Courses (course_name, course_code,
course_duration) VALUES (?, ?, ?)';
   const values = [courseName, courseCode, courseDuration];

   connection.query(insertQuery, values, function (err, results, fields) {
     if (err) {
       console.error('Error occurred while inserting data:', err);
     } else {
       console.log('Data inserted successfully!');
       res.redirect('/admin');
     }});

    } else {
        // Handle other cases or provide an error response
    }

});

// Handle delete request
app.post('/deleteCourse', encoder,(req, res) => {
  const courseId = parseInt(req.body.courseId, 10);
  const deleteQuery = 'DELETE FROM Courses WHERE course_id = ?';

    connection.query(deleteQuery, [courseId], (err, results) => {
        if (err) {
            console.error('Error executing DELETE query:', err);
            res.status(500).json({ message: 'Internal Server Error' });
        } else {
            console.log('Delete successful:', results);
            res.status(200).json({ message: 'Delete successful' });
        }
    });
  console.log(courseId);
});

app.post('/enrollCourse', encoder,(req, res) => {
  const courseId = req.body.courseId;
  const PRN="595";
  const Fname="Prathamesh";
  const Lname="Mandave";
  const values = [courseId, PRN, Fname,Lname];
  const insertQuery = 'INSERT INTO
Student(course_id,prn_no,student_Fname,student_lname) VALUES(?,?,?,?)';

    connection.query(insertQuery, values, (err, results) => {
        if (err) {
            console.error('Error executing insert query:', err);
            res.status(500).json({ message: 'Internal Server Error' });
        } else {
            console.log('insert successful:', results);
```

```
            res.status(200).json({ message: 'Insert successful' });
        }
    });
  console.log(courseId);
});

app.get('/studentprofile', (req, res) => {

    const selectQuery='select * from Studentdata where email =?';
    const email="pratham5@gmail.com";
    connection.query(selectQuery, email, (err, results) => {
        if(err){
          console.error('Error executing select query:', err);
          res.status(500).json({ message: 'Internal Server Error' });
        }
        else{
          console.log('select successful:', results);
          //res.status(200).json({ message: 'select successfully' });
          res.render('Studentprofile',{data:results});
        }
    });

});

app.listen(port, () => {
  console.log(`Server is running on http://localhost:${port}`);
});
```