6. Use Node.js, Express.js, Mongoose and other necessary JS APIs to implement a RESTful Application which manages students and their courses.
   When fetch students, the students array returned like this. When you design your application, you can choose to use 1 model or 2 models. Below is an example:

```json
[
    {
        "studentId": "610001",
        "firstname": "John",
        "lastname": "Smith",
        "courses": [
            {
                "courseId": "CS303",
                "coursename": "Web Application Programming in JavaScript",
                "semester": "2021-Spring",
                "grade": 99
            },
            {
                "courseId": "CS445",
                "coursename": "Modern Asynchronous Programming",
                "semester": "2021-Spring",
                "grade": 95
            }
        ]
    },
    {
        "studentId": "610002",
        "firstname": "Edward",
        "lastname": "Jack",
        "courses": [
            {
                "courseId": "CS472",
                "coursename": "Web Application Programming",
                "semester": "2021-Spring",
                "grade": 82
            },
            {
                "courseId": "CS544",
                "coursename": "Web Application Architecture",
                "semester": "2021-Spring",
                "grade": 88
            }
        ]
    }
]
```

You need to implement 5 APIs for this programming question:

1) **[4 points]** GET http://localhost:3000/students/{studentId} – this API returns a single student based on student Id.
   Example:
   Request: GET http://localhost:3000/students/610001
   Response JSON:

```
{
    "studentId": "610001",
    "firstname": "John",
    "lastname": "Smith",
    "courses": [{
            "courseId": "CS303",
            "coursename": "Web Application Programming in JavaScript",
            "semester": "2021-Spring",
            "grade": 99
        },
        {
            "courseId": "CS445",
            "coursename": "Modern Asynchronous Programming",
            "semester": "2021-Spring",
            "grade": 95
        }
    ]
}
```

2) **[5 points]** POST http://localhost:3000/students - this API saves a single student into database. The student id is manually assigned in Request body, NOT automatically generated on the server side. When save a student, need to check if the student id exists in database. If doesn't exists, save the student directly into database. If already exists in database, replace the existing one with the student sent via request body. In a nutshell, only 1 student exists with a specific student Id in the database, student Id is unique for a student.
Example:
Request: POST http://localhost:3000/students
Request Body:

```
{
    "studentId": "610001",
    "firstname": "John",
    "lastname": "Smith",
    "courses": [
        {
            "courseId": "CS303",
            "coursename": "Web Application Programming in JavaScript",
            "semester": "2021-Spring",
            "grade": 99
        }
    ]
}
```

Response JSON:

```
{
    "studentId": "610001",
    "firstname": "John",
    "lastname": "Smith",
    "courses": [
        {
            "courseId": "CS303",
            "coursename": "Web Application Programming in JavaScript",
            "semester": "2021-Spring",
            "grade": 99
        }
    ]
}
```

3) **[5 points]** GET http://localhost:3000/students/{studentId}/courses/{coureseId} – this API returns a single course of a particular student based on student Id and course Id.
Example: Suppose currently the database looks like below:

```
[
    {
        "studentId": "610001",
        "firstname": "John",
        "lastname": "Smith",
        "courses": [
            {
                "courseId": "CS303",
                "coursename": "Web Application Programming in JavaScript",
                "semester": "2021-Spring",
                "grade": 99
            }
        ]
    },
    {
        "studentId": "610002",
        "firstname": "Edward",
        "lastname": "Jack",
        "courses": [
            {
                "courseId": "CS303",
                "coursename": "Web Application Programming in JavaScript",
                "semester": "2021-Spring",
                "grade": 82
            }
        ]
    }
]
```

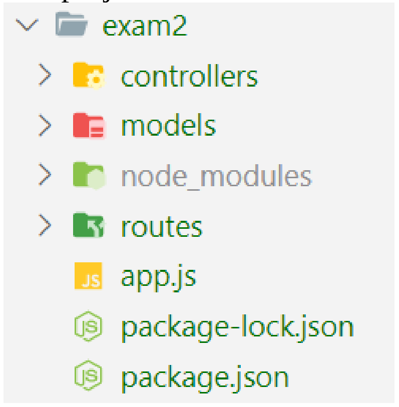
Request: GET http://localhost:3000/students/610001/courses/CS303 – based on database above, it returns a single course with course Id CS303 which belongs to a student with student Id: 610001

Response JSON:

```
{
    "courseId": "CS303",
    "coursename": "Web Application Programming in JavaScript",
    "semester": "2021-Spring",
    "grade": 99
}
```

NOTE for entire programming question:

I. You need to design the domain model by yourself. It's your own choice to have a single domain model or multiple domain models.
II. Fill in the necessary code below. You also need to name the js files in each folder and connect/import modules by yourself.
III. The project folder structure is:



```
exam2
  controllers
  models
  node_modules
  routes
  app.js
  package-lock.json
  package.json
```

app.js

```
const express = require('express');
const app = express();
app.use(express.json());

//place your code below









app.listen(3000, () => console.log('listening to 3000...'));
```

routes/______________________________.js

```
const express = require('express');
const router = express.Router();
//place your code below
















module.exports = router;
```

controllers/________________________________.js

```
//place your code below
```

models/___________________________________.js

```
//place your code below
```