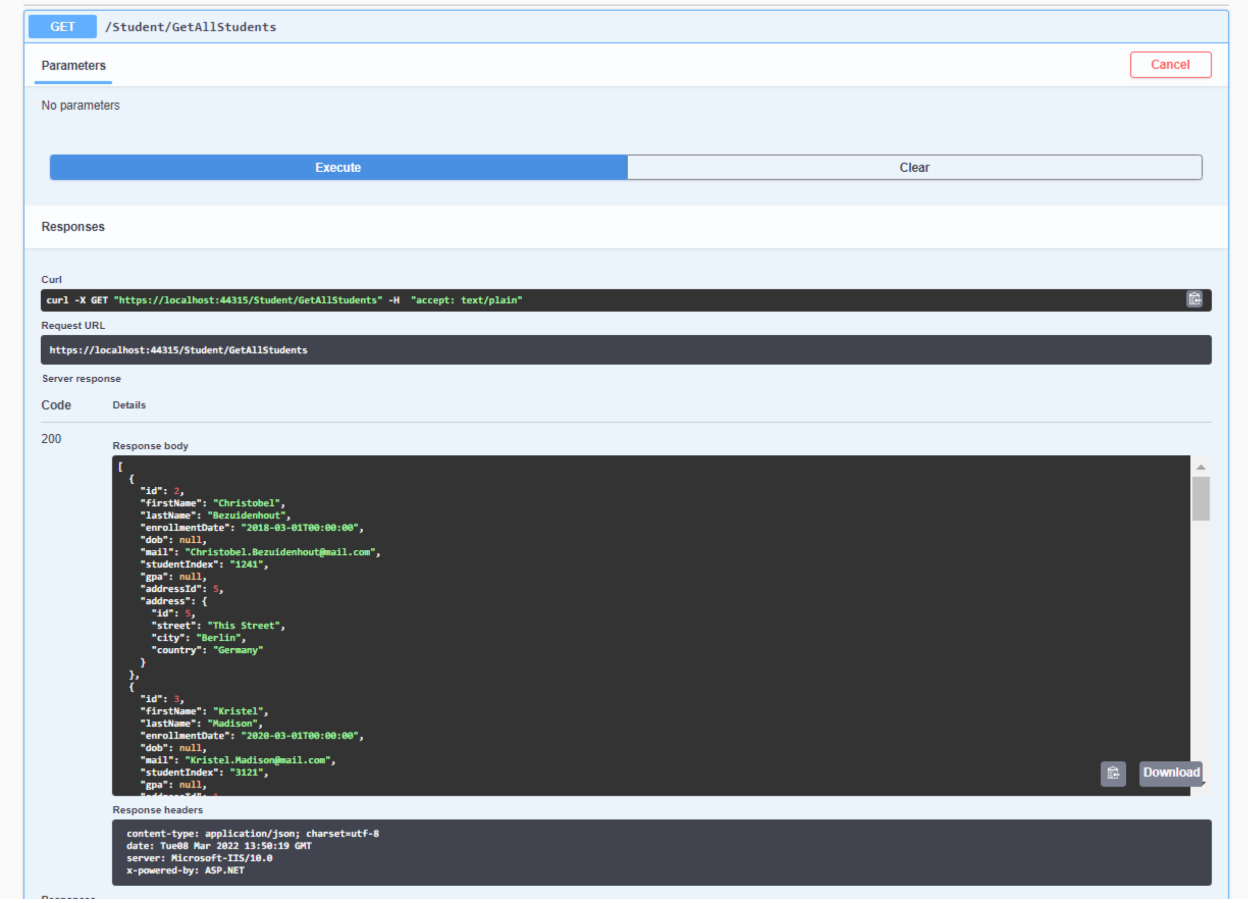
Internet Services Trial Final Exam

Instructions: Create a simple Hotel Room Management Application. Before you start, read the instructions to the end.

Tasks:

- Create a Web Api solution using .Net 8.
- All data in your solution should be persisted in a database. Use Entity Framework Core Code first approach.
- Create a layered architecture following the Service-Repository pattern and implement the requirements in the 3 layers appropriately: Data, Service and WebApi layers (feel free to use other naming convention for your layers as preferable).
- Create the necessary entities: Guest and Room.
- Guest should have the following properties: Autogenerated PK Id, First Name, Last Name, DOB, Address, Nationality, Check-in Date, Check-out Date an FK Room Id which will be the connection to the Rooms table.
- On the Guest entity set the following restrictions: Guest Id should be autogenerated, all other properties (First Name, Last Name, DOB, Address, Nationality, Check-in Date, Check-out Date) should be required, First Name should have max length of 200 chars and Last Name should have max length of 400 chars, Address should have max length of 600 chars.
- Room should have the following properties: Autogenerated PK Id, Number, Floor, Type.
- On the Room entity set the following restrictions: Room Id should be autogenerated, all other properties (Number, Floor, Type) should be required.
- Each Guest is allowed to book (stay) one Room at the same time only, while multiple Guests can be staying the same Room (one Room can relate to multiple Guests).
- Create the two appropriate interfaces and define methods for get all, get by Id, create, update and delete for Guest and Room.
- Create DTOs for the Guest and Room.
- Create two services and implement the interfaces for Guest and Room, so that they can be called as services.
- Create two API controllers for Guest and Room CRUD methods using the services and add appropriate implementation using DI and IOC.
- Implement the GuestRepository and RoomRepository.
- Implement the two interfaces IGuestRepository and IRoomRespository respectively in the correct layer.

- Use the created repositories and appropriate interfaces to communicate and manipulate the data in the database using DI and IOC.
- Create a Test project and implement a few Unit tests.
- For the GuestService create at least 3 Unit tests.
- For the RoomService create at least 3 Unit tests.
- Run the Unit tests and make sure that they run successfully.
- Put the source code in a Zip folder named Name_Surname_Id.zip and upload on the provided submission link. If the solution is too big for the submission link, you can upload it on github or cloud and upload a .txt file containing a link to your online code repository on the provided submission link. **Make sure that your code repository is publicly accessible!**
- Together with the source code, provide screenshots of all the endpoints for both Guest and Room: Get All Guests, Get Guest, Post Guest, Put Guest, Delete Guest and Get All Rooms, Get Room, Post Room, Put Room, Delete Room. Screenshots could be from either Swagger or Postman. Screenshots should contain request and response. **Screenshots are mandatory! No screenshots = no points!** Example of how one screenshot should look can be found below:



/Student/GetAllStudents

No parameters

Curl

```
curl -X GET "https://localhost:44315/Student/GetAllStudents" -H  "accept: text/plain"
```

Request URL

```
https://localhost:44315/Student/GetAllStudents
```

Server response

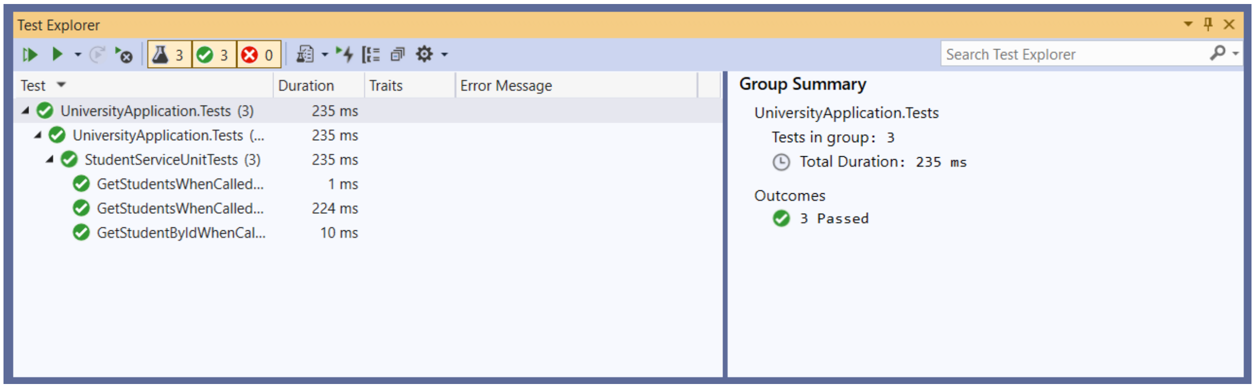
Code	Details
200	Response body

```
[
  {
    "id": 2,
    "firstName": "Christobel",
    "lastName": "Bezuidenhout",
    "enrollmentDate": "2018-03-01T00:00:00",
    "dob": null,
    "mail": "Christobel.Bezuidenhout@mail.com",
    "studentIndex": "1241",
    "gpa": null,
    "addressId": 5,
    "address": {
      "id": 5,
      "street": "This Street",
      "city": "Berlin",
      "country": "Germany"
    }
  },
  {
    "id": 3,
    "firstName": "Kristel",
    "lastName": "Madison",
    "enrollmentDate": "2020-03-01T00:00:00",
    "dob": null,
    "mail": "Kristel.Madison@mail.com",
    "studentIndex": "3121",
    "gpa": null,
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue08 Mar 2022 13:50:19 GMT
server: Microsoft-IIS/10.0
x-powered-by: ASP.NET
```

- Together with the source code, provide screenshots of the Test Run after you have ran them all. **Screenshots are mandatory! No screenshots = no points!** Example of how one screenshot from the test runs should look can be found below:



Test	Duration	Traits	Error Message
✓ UniversityApplication.Tests (3)	235 ms		
✓ UniversityApplication.Tests (...	235 ms		
✓ StudentServiceUnitTests (3)	235 ms		
✓ GetStudentsWhenCalled...	1 ms		
✓ GetStudentsWhenCalled...	224 ms		
✓ GetStudentByIdWhenCal...	10 ms		

Group Summary

UniversityApplication.Tests

Tests in group: 3

Total Duration: 235 ms

Outcomes

✓ 3 Passed

Requirements & Tips:

- You should use .Net 8 version.
- You can use Automapper for models mapping. Using Automapper is advisable, but you can also implement manual mapping for the entities and DTOs.
- For data layer you should use Entity Framework Core.
- You can add some test data to the database, but it is not mandatory.
- You can use either Swagger or Postman for testing your web api and for providing the test screenshots.
- Providing the backup from your database is not mandatory.