Background

Ufinity School of Technology (UST) is engaging you to design and develop the backend APIs of School Administration System (SAS), which will be used by school administrators and teachers to perform various administrative functions.

The following can be assumed about the teachers and students:

1. A teacher can teach in multiple classes.
2. A teacher can teach multiple subjects, regardless to the same or different class.
3. 2 different teachers can teach the same subject in the same class.
4. A student can be in multiple classes.

You are tasked to implement the following APIs, with more details found under *User Stories* section:

1. **Registration** API that **create/update** the required data.
2. **WorkloadReport** API that **returns the required data in JSON format**, so that it can be used to generate the report.

Your Task

1. Use one of two base codes (JavaScript or TypeScript) provided.
2. Extend the base code with a set of API endpoints, listed under *User Stories* section.
3. Your code must be hosted on GitHub, or any other similar publicly accessible code repository.
4. You should overwrite the default README.md to includes the following:
   a. The NodeJS version that you are using.
   b. Instructions for running the local instance of your API server as we need to be able to launch and test your solution locally.
5. You may add any new libraries needed, without replacing the following libraries:
   a. **Framework**: ExpressJS
   b. **ORM**: Sequelize
   c. **Database:** MySql 8.0
   d. **Logger:** WinstonJS
   e. **Test Runner:** Jest
6. Use the **async/await** syntax instead of Promise chaining (.then(), .catch()).
7. Include unit tests.
8. Your API will be subjected to automated test tools, so **please adhere closely to the given specifications** (according to ***User Stories*** section)
9. **Send us the URL of the code repository containing the completed assignment. Ensure both the URL and README.md do not have the word “Ufinity" in it.**
10. If you are selected for a face-to-face interview, you should be prepared to:
    a. Walk through your code to interviewers.
    b. Explain any design decisions you have made.
    c. Modify the API endpoints or implement more endpoints.

Assessment Criteria

1. Readability
2. Maintainability
3. Code cleanliness
4. Code structure/design, e.g. modularity, testability
5. Database design, e.g. normalized, correct keys and indices
6. Balance between performance and readability
7. Meaningful error responses
8. Appropriate logs
9. Appropriate comments
10. Appropriate typing, which facilitate code completion and type checking, if typescript is used

Queries

If you have any queries, contact the Ufinity person who is currently liaising with you.

User Stories

1. **As an Administrator, I want to register Teachers, Students, Classes and Subject in a single API, so that I can use the system for administrative purposes.**

   Description

   Registration API should be able to:

   - create new record(s)
   - update existing record(s)

   Requirements

   - **Registration** API should return 204.
   - Teachers and students can be **uniquely identified** by their email address.
   - Subject can be **uniquely identified** by subject code.
   - Class can be **uniquely identified** by class code.
   - **All** input fields are **mandatory**.

   Expected Request

   Method: POST
   Endpoint: /api/register
   Body:

```
{
  "teacher": {
    name: "Teacher 1",
    email: "teacher1@gmail.com"
  },
  "students": [{
    name: "Student 1",
    email: "student1@gmail.com"
  }, {
    name: "Student 2",
    email: "student2@gmail.com"
  }],
  "subject": {
    subjectCode: "ENG",
    name: "English"
  },
  "class": {
    classCode: "P1-1",
    name: "P1 Integrity"
  }
}
```

   Expected Response

   Success Code: 204
   Error Code: 400 or 500

2. **As an Administrator, I should be able to generate a report on a Teacher's workload, so that I use it for planning.**

Description

WorkloadReport API will return the required data in JSON format, so that the data can be used to generate a report in the frontend.

Requirements

- **WorkloadReport** API should return the required data in JSON format as below.
- The data required for each teacher are:
  a) Subject Code
  b) Subject Name
  c) How many classes is the Teacher teaching for the Subject (in a)

Expected Request

Method: GET
Endpoint: /api/reports/workload

Expected Response

Success HTTP Code: 200
Error HTTP Code: 500
Body:

```
{
  "dummy teacher name 1": [
    {
      "subjectCode": "ENG",
      "subjectName": "English",
      "numberOfClasses": 1,
    },
    {
      "subjectCode": "MATH",
      "subjectName": "Mathematics",
      "numberOfClasses": 3,
    }
  ],
  "dummy teacher name 2": [
    {
      "subjectCode": " ENG",
      "subjectName": " English",
      "numberOfClasses": 2,
    }
  ]
}
```