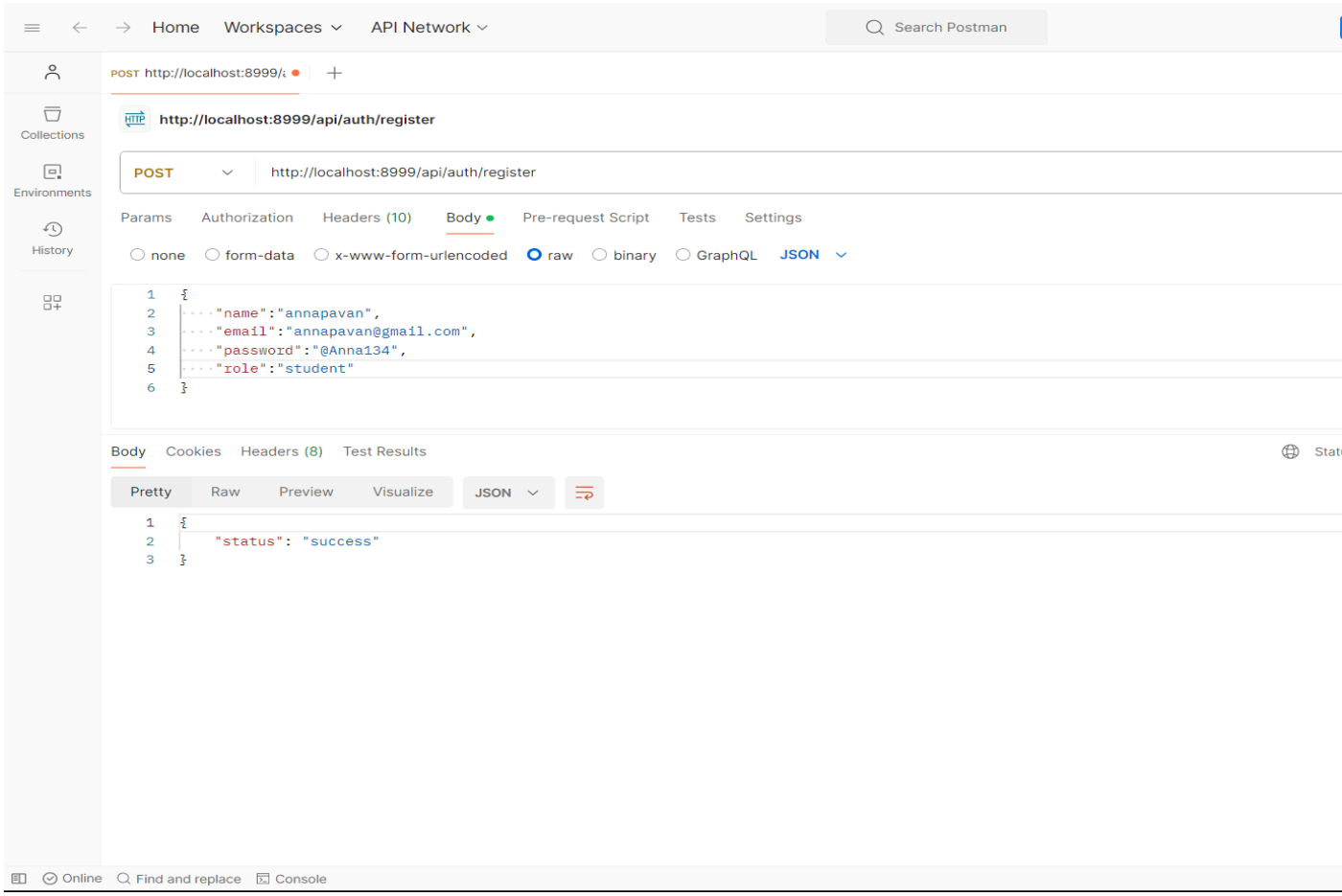
Student Assignments API Documentation

1. Register a New User

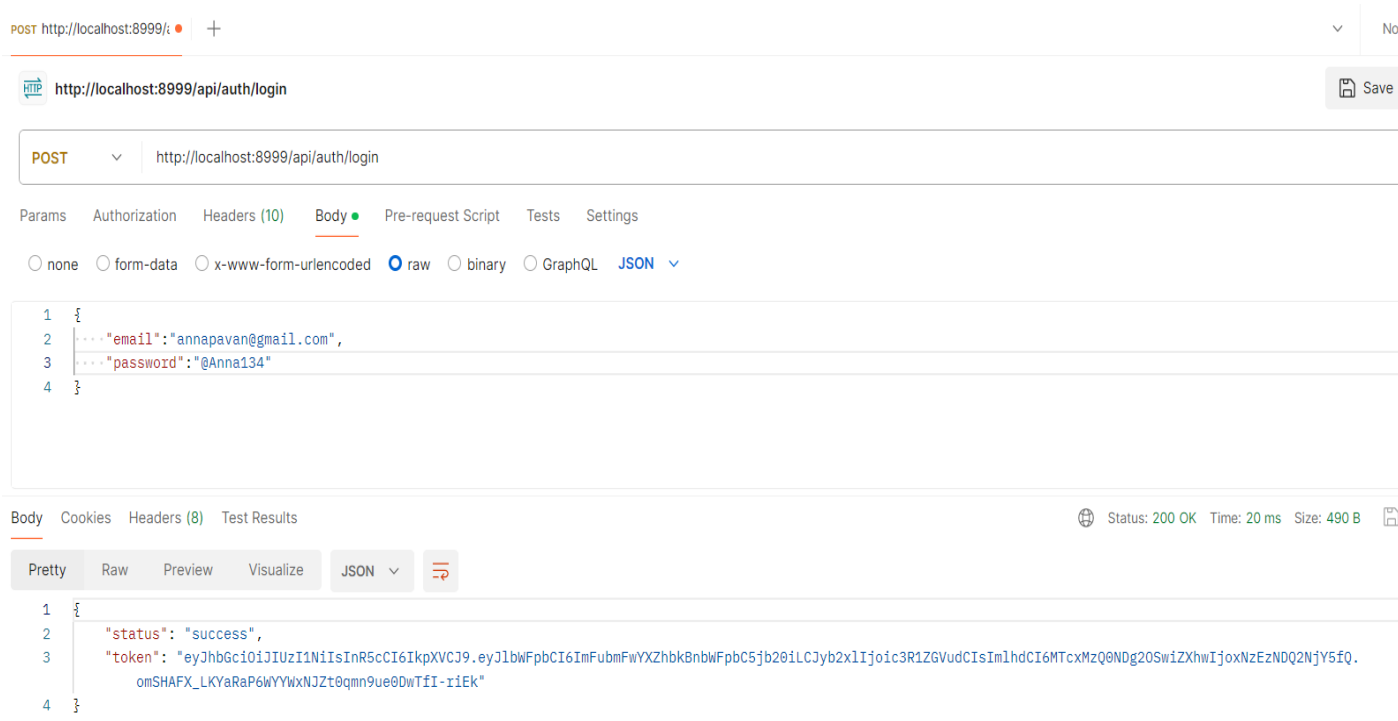
- Endpoint: /api/auth/register
- Method: POST



Description: When registering, users can choose between two roles: teacher or student. Select your desired role during the registration process.

2. Login

- Endpoint: /api/auth/login
- Method: POST
- Description: Authenticates users and returns a JWT token.



Teacher Assignments Endpoints

Teacher should login before performing these tasks.

These endpoints requires token verification in the header to authenticate the user before performing the task. Please ensure to include the token in the header when making the request.

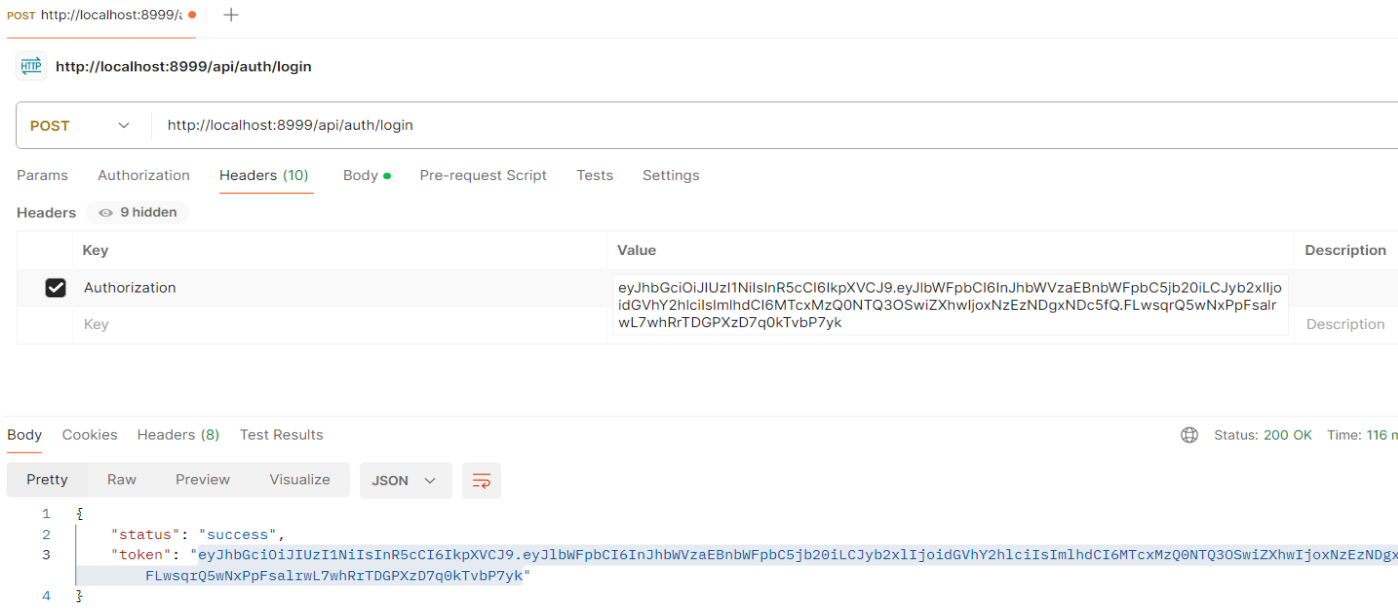
1. Create Assignment

EndPoint : /api/teach/addtask

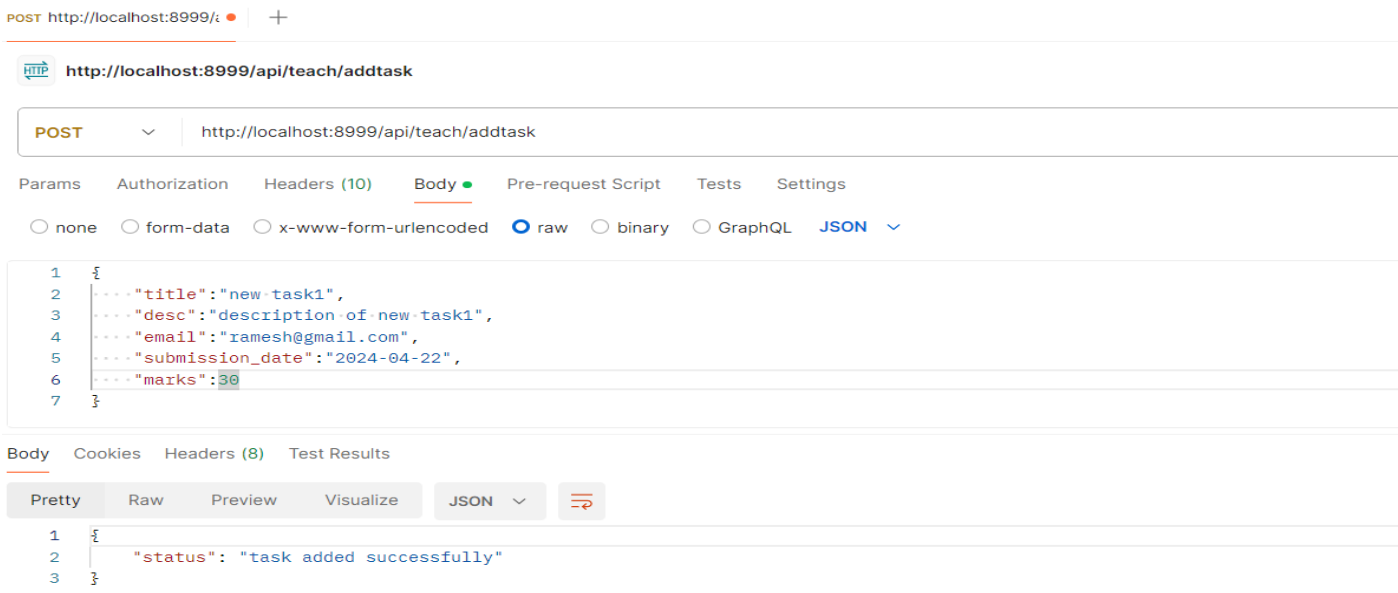
Method : GET

Description : Creates a new Assignment

Providing a token in the header part before performing the add task.



Adding a new Assignment



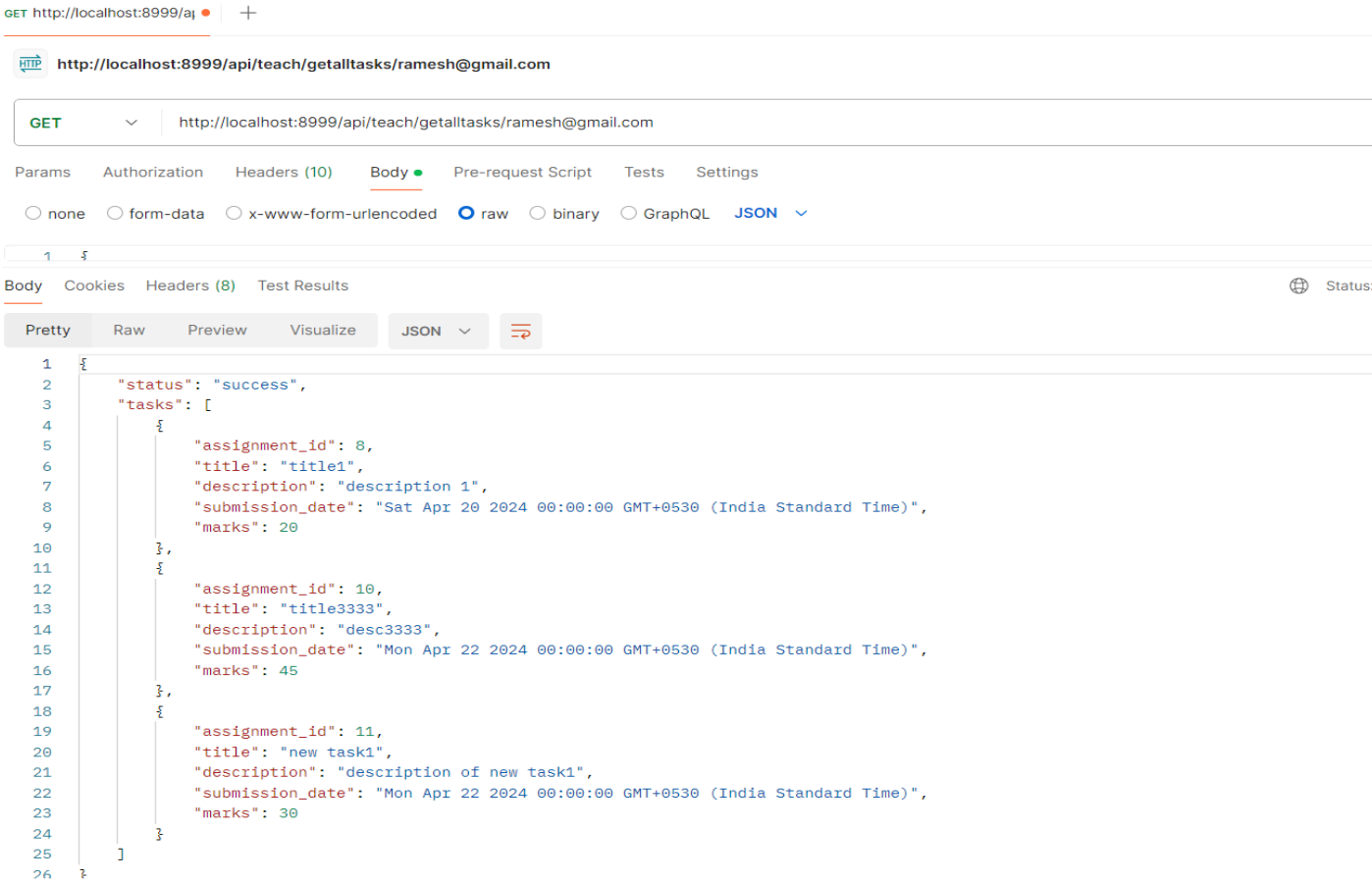
Assignment table before and after adding new task in MYSQL database.

```
mysql> select * from assignments;
+---------------+----------+--------------+-----------------+---------------------+---------------------+-----------------+-------+
| assignment_id | title    | description  | user_email      | created_at          | updated_at          | submission_date | marks |
+---------------+----------+--------------+-----------------+---------------------+---------------------+-----------------+-------+
|             8 | title1   | description 1| ramesh@gmail.com| 2024-04-17 21:41:40 | 2024-04-17 21:41:40 | 2024-04-20      | 20.00 |
|            10 | title3333| desc3333     | ramesh@gmail.com| 2024-04-18 00:05:35 | 2024-04-18 00:10:41 | 2024-04-22      | 45.00 |
+---------------+----------+--------------+-----------------+---------------------+---------------------+-----------------+-------+
2 rows in set (0.00 sec)

mysql> select * from assignments;
+---------------+----------+-------------------------+-----------------+---------------------+---------------------+-----------------+-------+
| assignment_id | title    | description             | user_email      | created_at          | updated_at          | submission_date | marks |
+---------------+----------+-------------------------+-----------------+---------------------+---------------------+-----------------+-------+
|             8 | title1   | description 1           | ramesh@gmail.com| 2024-04-17 21:41:40 | 2024-04-17 21:41:40 | 2024-04-20      | 20.00 |
|            10 | title3333| desc3333                | ramesh@gmail.com| 2024-04-18 00:05:35 | 2024-04-18 00:10:41 | 2024-04-22      | 45.00 |
|            11 | new task1| description of new task1| ramesh@gmail.com| 2024-04-18 18:38:43 | 2024-04-18 18:38:43 | 2024-04-22      | 30.00 |
+---------------+----------+-------------------------+-----------------+---------------------+---------------------+-----------------+-------+
3 rows in set (0.00 sec)
```

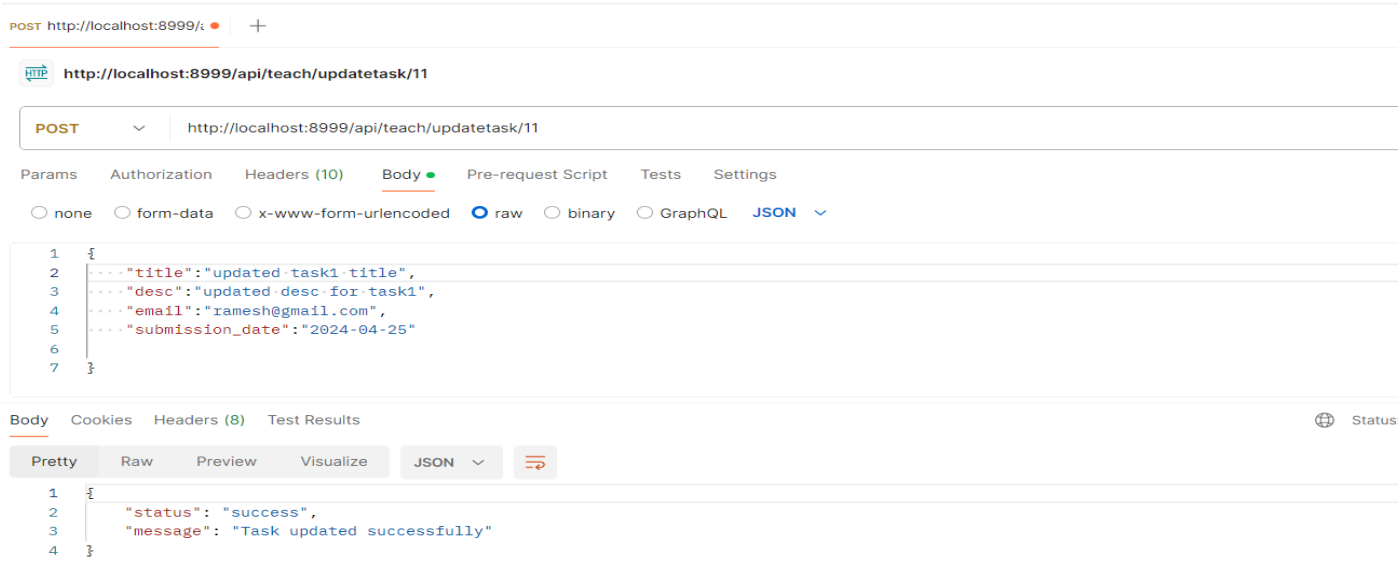
2. Get All Assignments

- Endpoint: /api/teach/getalltasks/{email_id}
- Method: GET
- Description: Retrieves all assignments



3. Update Assignment

- Endpoint: /api/teach/updatetask/{assignmentId}
- Method: PUT
- Description: Updates an existing assignment



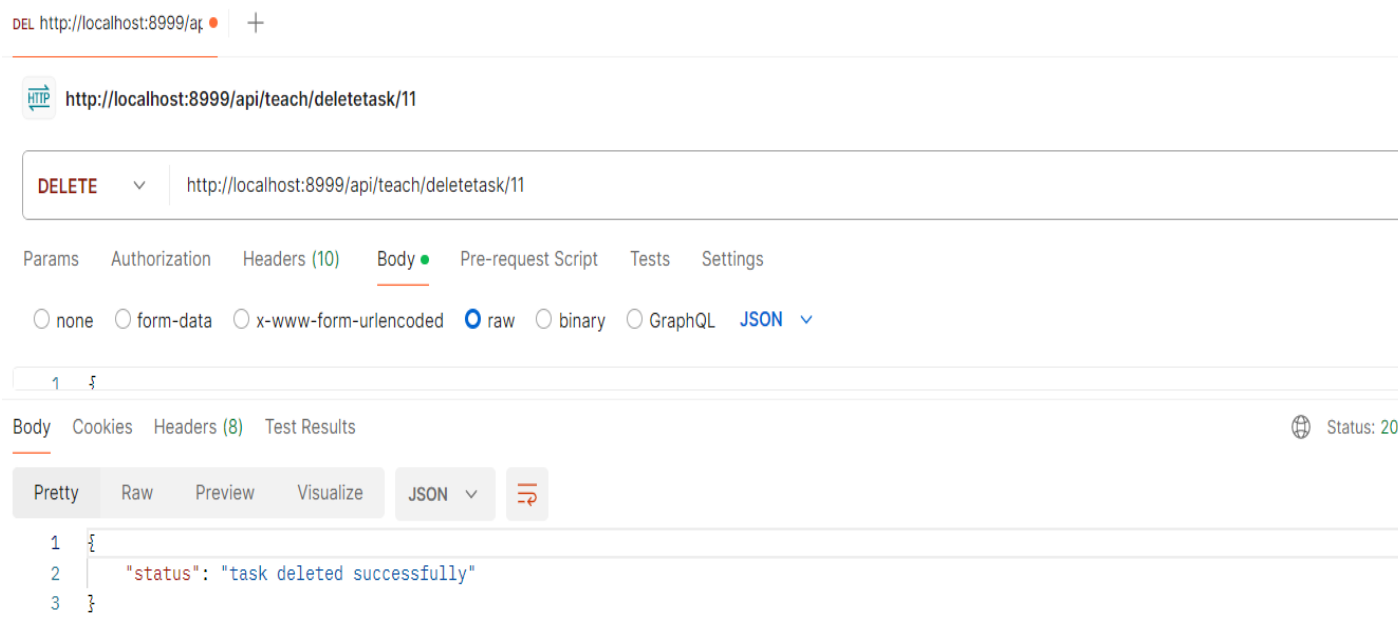
Assignment table before and after updating a task.

```
mysql> select * from assignments;
+---------------+-----------+--------------------------+-----------------+---------------------+---------------------+-----------------+-------+
| assignment_id | title     | description              | user_email      | created_at          | updated_at          | submission_date | marks |
+---------------+-----------+--------------------------+-----------------+---------------------+---------------------+-----------------+-------+
|             8 | title1    | description 1            | ramesh@gmail.com | 2024-04-17 21:41:40 | 2024-04-17 21:41:40 | 2024-04-20      | 20.00 |
|            10 | title3333 | desc3333                 | ramesh@gmail.com | 2024-04-18 00:05:35 | 2024-04-18 00:10:41 | 2024-04-22      | 45.00 |
|            11 | new task1 | description of new task1 | ramesh@gmail.com | 2024-04-18 18:38:43 | 2024-04-18 18:38:43 | 2024-04-22      | 30.00 |
+---------------+-----------+--------------------------+-----------------+---------------------+---------------------+-----------------+-------+
3 rows in set (0.00 sec)

mysql> select * from assignments;
+---------------+--------------------+----------------------+-----------------+---------------------+---------------------+-----------------+-------+
| assignment_id | title              | description          | user_email      | created_at          | updated_at          | submission_date | marks |
+---------------+--------------------+----------------------+-----------------+---------------------+---------------------+-----------------+-------+
|             8 | title1             | description 1        | ramesh@gmail.com | 2024-04-17 21:41:40 | 2024-04-17 21:41:40 | 2024-04-20      | 20.00 |
|            10 | title3333          | desc3333             | ramesh@gmail.com | 2024-04-18 00:05:35 | 2024-04-18 00:10:41 | 2024-04-22      | 45.00 |
|            11 | updated task1 title | updated desc for task1 | ramesh@gmail.com | 2024-04-18 18:38:43 | 2024-04-18 18:58:54 | 2024-04-25      | 30.00 |
+---------------+--------------------+----------------------+-----------------+---------------------+---------------------+-----------------+-------+
3 rows in set (0.00 sec)
```

4.Delete Assignment

- Endpoint: /api/teach/deletetask/{assignmentId}
- Method: DELETE
- Description: Deletes an assignment by ID.



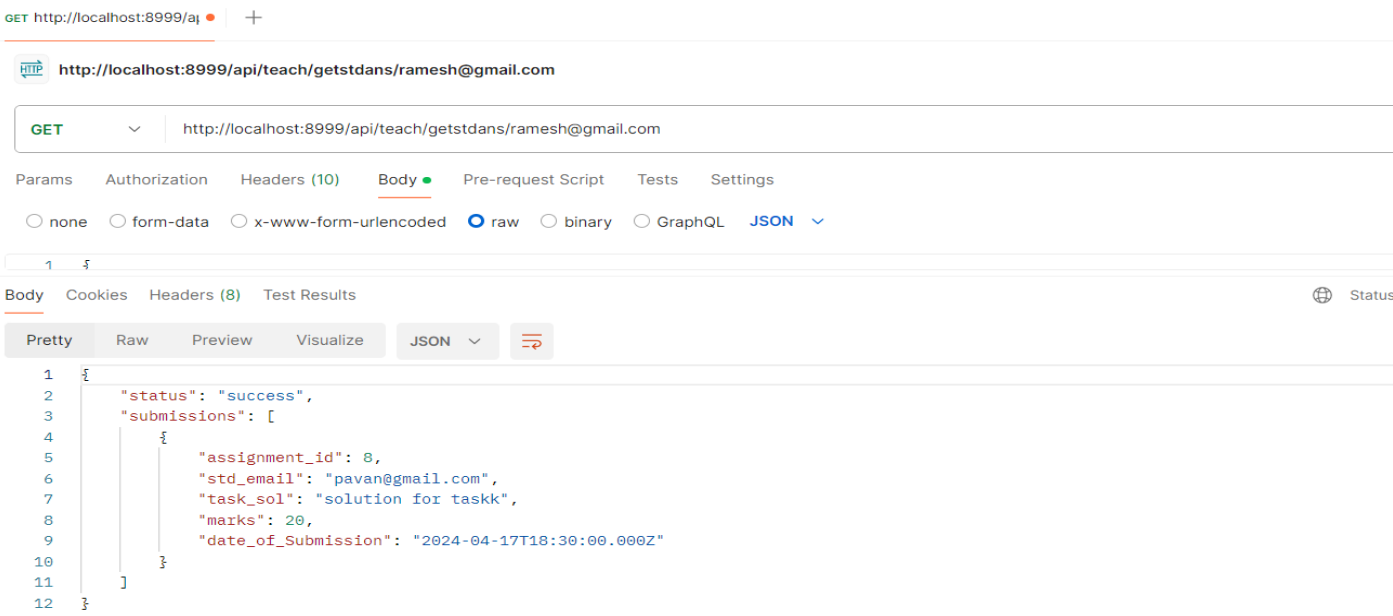
Assignments table before and after deleting a task.

```
mysql> select * from assignments;
+---------------+--------------------+-----------------------+-----------------+---------------------+---------------------+-----------------+-------+
| assignment_id | title              | description           | user_email      | created_at          | updated_at          | submission_date | marks |
+---------------+--------------------+-----------------------+-----------------+---------------------+---------------------+-----------------+-------+
|             8 | title1             | description 1         | ramesh@gmail.com | 2024-04-17 21:41:40 | 2024-04-17 21:41:40 | 2024-04-20      | 20.00 |
|            10 | title3333          | desc3333              | ramesh@gmail.com | 2024-04-18 00:05:35 | 2024-04-18 00:10:41 | 2024-04-22      | 45.00 |
|            11 | updated task1 title | updated desc for task1 | ramesh@gmail.com | 2024-04-18 18:38:43 | 2024-04-18 18:58:54 | 2024-04-25      | 30.00 |
+---------------+--------------------+-----------------------+-----------------+---------------------+---------------------+-----------------+-------+
3 rows in set (0.00 sec)

mysql> select * from assignments;
+---------------+-----------+---------------+-----------------+---------------------+---------------------+-----------------+-------+
| assignment_id | title     | description   | user_email      | created_at          | updated_at          | submission_date | marks |
+---------------+-----------+---------------+-----------------+---------------------+---------------------+-----------------+-------+
|             8 | title1    | description 1 | ramesh@gmail.com | 2024-04-17 21:41:40 | 2024-04-17 21:41:40 | 2024-04-20      | 20.00 |
|            10 | title3333 | desc3333      | ramesh@gmail.com | 2024-04-18 00:05:35 | 2024-04-18 00:10:41 | 2024-04-22      | 45.00 |
+---------------+-----------+---------------+-----------------+---------------------+---------------------+-----------------+-------+
2 rows in set (0.00 sec)
```

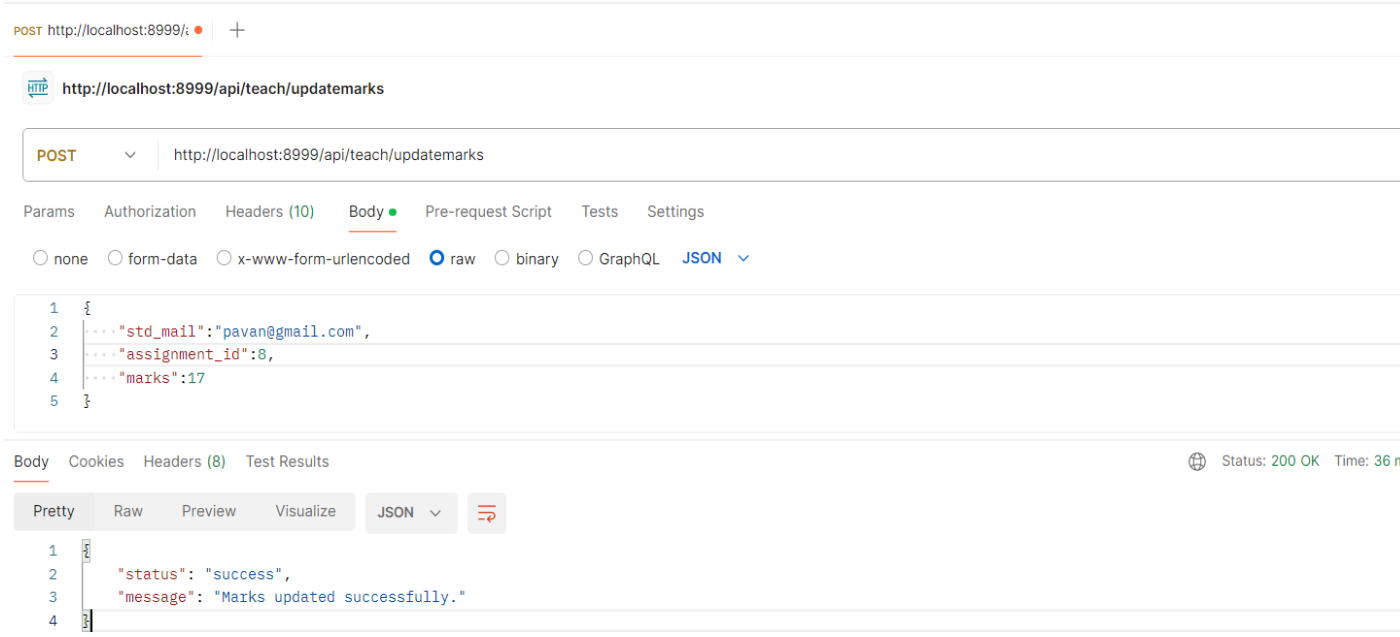
5.Getting Submissions from students

- Endpoint: /api/teach/getstdans/{email_Id}
- Method: GET
- Description: Get all the submissions.



6.Updating Marks

- Endpoint: /api/teach/updatemarks
- Method: POST
- Description: Update the marks after the task submission by students.



Submit table before and after updating the marks by respective teacher.

```
mysql> select * from submit;
+-----------+---------------+----------------+--------------------+---------------+-------------------+-------------+
| submit_id | assignment_id | std_email      | task_sol           | obtained_marks | date_of_Submission | total_marks |
+-----------+---------------+----------------+--------------------+---------------+-------------------+-------------+
|         2 |             8 | pavan@gmail.com | solution for taskk |          NULL | 2024-04-18        |       20.00 |
+-----------+---------------+----------------+--------------------+---------------+-------------------+-------------+
1 row in set (0.00 sec)

mysql> select * from submit;
+-----------+---------------+----------------+--------------------+---------------+-------------------+-------------+
| submit_id | assignment_id | std_email      | task_sol           | obtained_marks | date_of_Submission | total_marks |
+-----------+---------------+----------------+--------------------+---------------+-------------------+-------------+
|         2 |             8 | pavan@gmail.com | solution for taskk |         17.00 | 2024-04-18        |       20.00 |
+-----------+---------------+----------------+--------------------+---------------+-------------------+-------------+
1 row in set (0.00 sec)
```

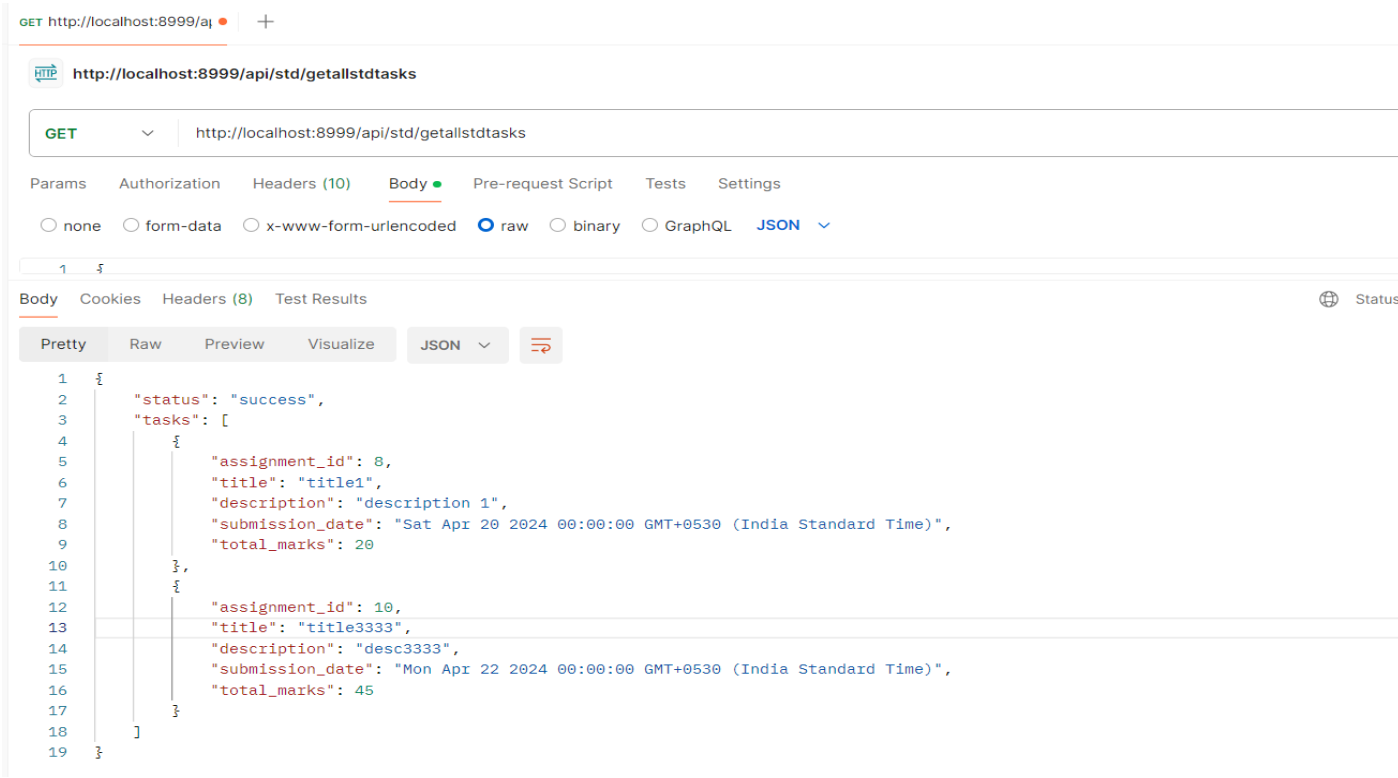
Student Assignments Endpoints

Student should login before performing these tasks.

These endpoints require token verification in the header to authenticate the user before performing the task. Please ensure to include the token in the header when making the request.

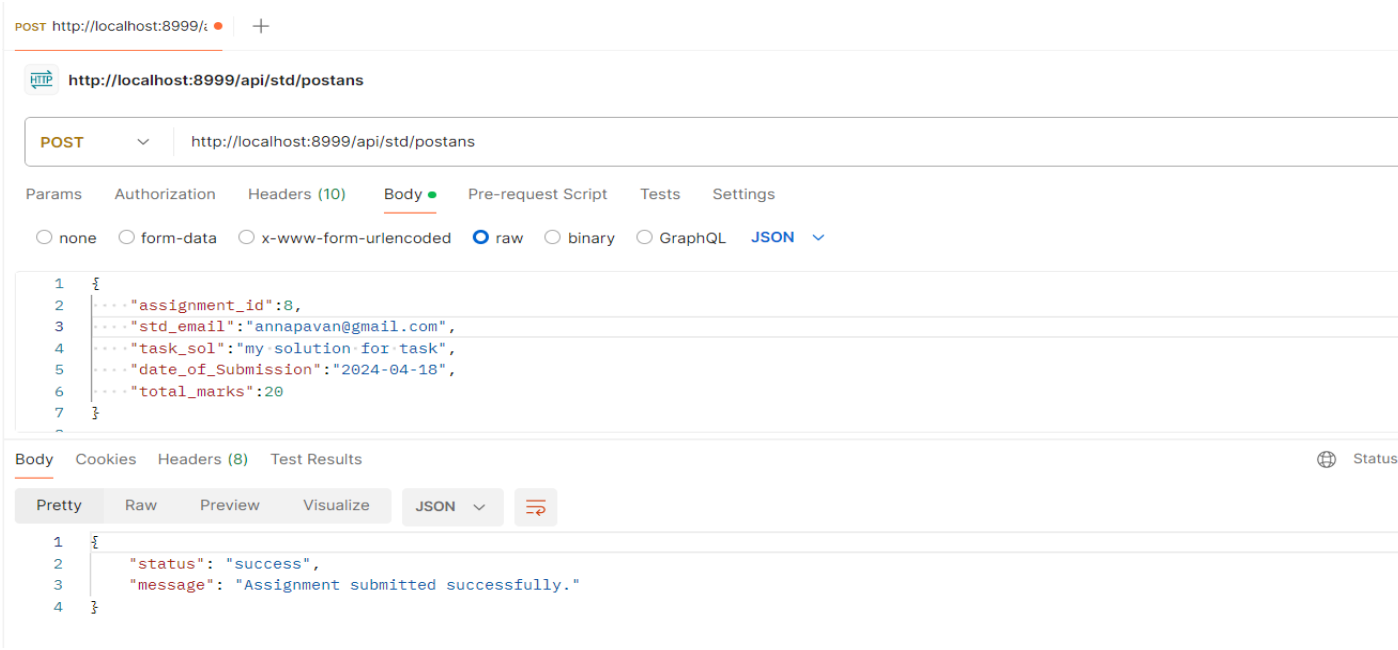
1.Getting all tasks

- Endpoint: /api/std/getallstdtasks
- Method: GET
- Description: Get all the tasks assigned by the teacher.



2.Submission of tasks

- Endpoint: /api/teach/postans
- Method: POST
- Description: Student can post the solution for assigned tasks.



Submit table before and after posting the solution by student

```
mysql> select * from submit;
+-----------+---------------+----------------+--------------------+----------------+--------------------+-------------+
| submit_id | assignment_id | std_email      | task_sol           | obtained_marks | date_of_Submission | total_marks |
+-----------+---------------+----------------+--------------------+----------------+--------------------+-------------+
|         2 |             8 | pavan@gmail.com | solution for taskk |          17.00 | 2024-04-18         |       20.00 |
+-----------+---------------+----------------+--------------------+----------------+--------------------+-------------+
1 row in set (0.00 sec)

mysql> select * from submit;
+-----------+---------------+--------------------+--------------------+----------------+--------------------+-------------+
| submit_id | assignment_id | std_email          | task_sol           | obtained_marks | date_of_Submission | total_marks |
+-----------+---------------+--------------------+--------------------+----------------+--------------------+-------------+
|         2 |             8 | pavan@gmail.com    | solution for taskk |          17.00 | 2024-04-18         |       20.00 |
|         3 |             8 | annapavan@gmail.com | my solution for task |           NULL | 2024-04-18         |       20.00 |
+-----------+---------------+--------------------+--------------------+----------------+--------------------+-------------+
2 rows in set (0.00 sec)
```

Documentation for Dockerizing Nodejs Application

1. **Clone the Repository:**

   Clone the repository containing your Node.js application to your local machine.

2. **Build Docker Image:**

   Navigate to the root directory of your Node.js application in your terminal.

   Run the following command to build a Docker image:

   ```
   docker build -t annapavan/hey-bn:0.0.1.RELEASE .
   ```

   This command will create a Docker image with the specified tag.

3.Run Docker Container:

After building the Docker image, you can run a Docker container using the following command:

```
docker container run -d -p 8999:8999 annapavan/hey-bn:0.0.1.RELEASE
```

This command will start a Docker container in detached mode (-d) and expose port 8999 of the container to port 8999 of the host machine (-p 8999:8999).

4.Verify Container Status:

You can verify that the Docker container is running by executing the following command:

```
docker container ls
```

5. Stop Docker Container:

If you want to stop the Docker container, you can use the following command:

```
docker container stop <CONTAINER_ID>
```

Replace <CONTAINER_ID> with the actual ID of the Docker container.

6.Push Docker Image to Docker Hub:

If you want to push the Docker image to Docker Hub, you can execute the following command:

```
docker push annapavan/hey-nodejs:0.0.1.RELEASE
```

This command will push the Docker image to the specified repository on Docker Hub.

7.Access the Application:

Once the Docker container is running, you can access your Node.js application by navigating to http://localhost:8999 in your web browser**.**

8.Additional Notes:

Ensure that your Node.js application is configured to listen on port 8999 or the port that you have exposed in your Docker container.

Modify the Dockerfile and Docker image tag as needed to match your application's specific requirements.

Database Schema

Users Table (users):

Field	Type	Null	Key	Default	Extra
name	varchar(100)	NO		NULL	
email	varchar(50)	NO	PRI	NULL	
password	varchar(1000)	NO		NULL	
role	varchar(50)	NO		NULL	

Assignments Table (assignments):

Field	Type	Null	Key	Default	Extra
assignment_id	int	NO	PRI	NULL	auto_increment
title	varchar(255)	NO		NULL	
description	text	NO		NULL	
user_email	varchar(255)	NO	MUL	NULL	
created_at	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
updated_at	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED on update CURRENT_TIMESTAMP
submission_date	date	NO		NULL	
marks	decimal(5,2)	NO		NULL	

Submit Table (submit):

Field	Type	Null	Key	Default	Extra
submit_id	int	NO	PRI	NULL	auto_increment
assignment_id	int	NO	MUL	NULL	
std_email	varchar(255)	NO		NULL	
task_sol	text	NO		NULL	
obtained_marks	decimal(5,2)	NO		NULL	
date_of_Submission	date	NO		NULL	
total_marks	decimal(5,2)	NO		NULL	

README Instructions for Nodejs , Express

1. **Download the ZIP File**:
    - Extract it to your desired location.
2. **Install dependencies**:
    - Navigate to the project directory.
    - Run **npm install** to install all the required dependencies.
3. **Database setup**:
    - Set up your database and configure the connection details in either **config.js** or **.env** file.
    - Ensure that the database credentials and connection details are correctly configured to establish a successful connection.
4. **Run the application**:
    - Start the application by running **node Server.js** in your terminal.
    - Make sure the application is running without any errors.
5. **Access API endpoints**:
    - Use the provided API documentation to access and interact with the available endpoints.
    - The API endpoints are described in detail in the documentation.

Note : Above mentioned (screenshots) postman endpoints are being done using Express , Nodejs backend

README Instructions for SpringBoot

1. **Download the ZIP File :**
    - Extract it to your desired location.
2. **Install dependencies:**
    - Navigate to the project directory.
    - As we are using Maven we don't need to manually install dependencies, as Maven will handle it for you.
3. **Database setup:**
    - Set up your database and configure the connection details in the **application.properties** file located in the **src/main/resources** directory.
    - Ensure that the database credentials and connection details are correctly configured to establish a successful connection.
4. **Run the application:**
    - Start the Spring Boot application by running the main class, typically annotated with **@SpringBootApplication**, from your IDE or using the **mvn spring-boot:run** command .
    - Make sure the application starts up without any errors.
5. **Access API endpoints:**
    - Use the provided API documentation or explore the controller classes to understand and interact with the available endpoints.
    - The API endpoints are typically defined in controller classes annotated with **@RestController** and mapped to specific URL paths using **@RequestMapping** annotations.