Backend

Note Taking App CRUD API

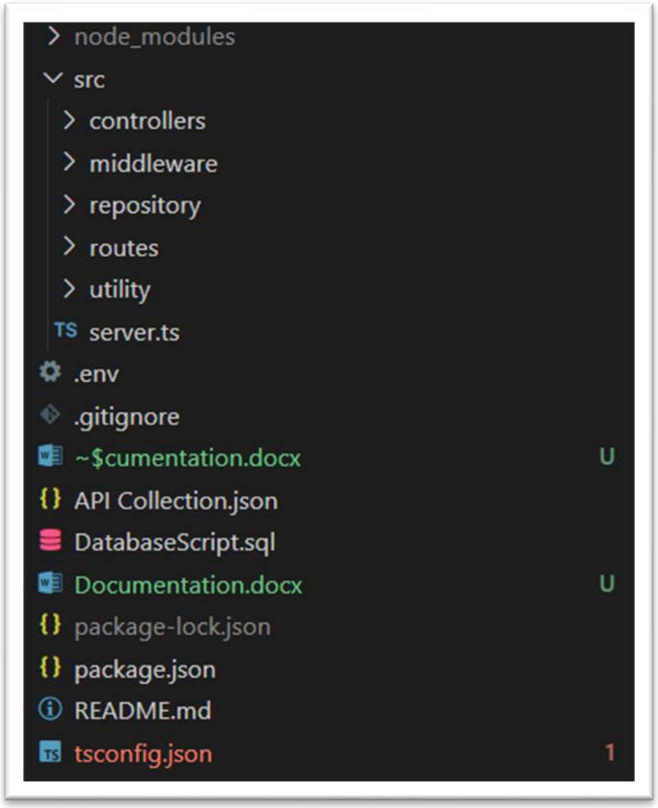
Parthraj Panchal

DATE :21st March,2024

Index

Project Structure

For developing backend API project, I have followed a particular folder structure to keep it clean and organized.



```
> node_modules
v src
  > controllers
  > middleware
  > repository
  > routes
  > utility
  TS server.ts
.env
.gitignore
~$cumentation.docx                      U
API Collection.json
DatabaseScript.sql
Documentation.docx                      U
package-lock.json
package.json
README.md
tsconfig.json                           1
```

Figure 1: Project folder structure

1) Controllers

Controllers in Express.js projects are modules containing encapsulating request processing logic for better code organization and maintainability.

In this project, we only have 'NoteController.ts' which handles logic for note's CRUD operations.

2) Middleware

Middleware in Express.js intercepts and modifies incoming requests or outgoing responses, enabling functionalities like logging, authentication, and error handling, enhancing the flexibility and extensibility of web applications.

In this project, we have implemented 'RequestLoggerMiddleware.ts' which logs all incoming requests with data.

3) Repository

In Express.js projects, repositories typically encapsulate data access logic, providing an abstraction layer for interacting with databases or other data sources, promoting separation of concerns and maintainability in applications.

In this project, we have created 'NoteRepository.ts' which consist helper methods for to interact with database.

4) Routes

In an Express.js project, routes define the URL endpoints that clients can request, and they map these endpoints to specific controller functions, facilitating the organization and handling of incoming requests in the application.

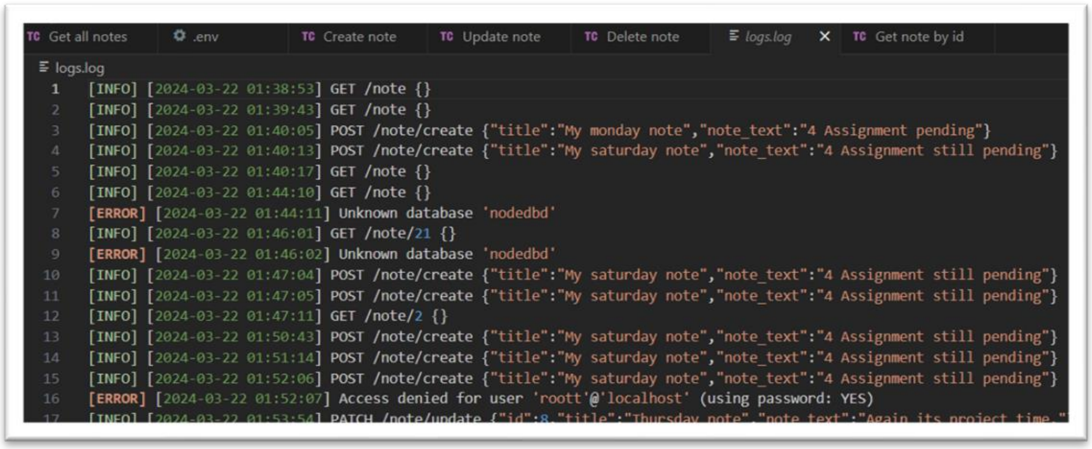
In this project, we have used 'NoteRoute.ts' which handles and redirect the incoming HTTP requests to specific controller method.

5) Utility

This folder contains specific utility items such as

- **ResponseModel (for ensuring specific response format to user)**
  This model contains three fields
  - **isSuccess -> shows status of operation**
  - **message -> specific message from backend**
  - **responseData-> data of interest**

- **Logger (for logging important logs to file and console)**
  - **Example:**



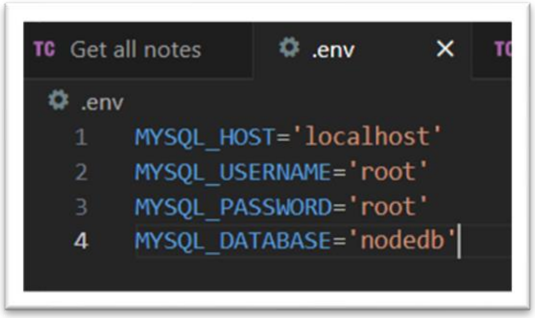
```
[INFO] [2024-03-22 01:38:53] GET /note {}
[INFO] [2024-03-22 01:39:43] GET /note {}
[INFO] [2024-03-22 01:40:05] POST /note/create {"title":"My monday note","note_text":"4 Assignment pending"}
[INFO] [2024-03-22 01:40:13] POST /note/create {"title":"My saturday note","note_text":"4 Assignment still pending"}
[INFO] [2024-03-22 01:40:17] GET /note {}
[INFO] [2024-03-22 01:44:10] GET /note {}
[ERROR] [2024-03-22 01:44:11] Unknown database 'nodedbd'
[INFO] [2024-03-22 01:46:01] GET /note/21 {}
[ERROR] [2024-03-22 01:46:02] Unknown database 'nodedbd'
[INFO] [2024-03-22 01:47:04] POST /note/create {"title":"My saturday note","note_text":"4 Assignment still pending"}
[INFO] [2024-03-22 01:47:05] POST /note/create {"title":"My saturday note","note_text":"4 Assignment still pending"}
[INFO] [2024-03-22 01:47:11] GET /note/2 {}
[INFO] [2024-03-22 01:50:43] POST /note/create {"title":"My saturday note","note_text":"4 Assignment still pending"}
[INFO] [2024-03-22 01:51:14] POST /note/create {"title":"My saturday note","note_text":"4 Assignment still pending"}
[INFO] [2024-03-22 01:52:06] POST /note/create {"title":"My saturday note","note_text":"4 Assignment still pending"}
[ERROR] [2024-03-22 01:52:07] Access denied for user 'roott'@'localhost' (using password: YES)
[INFO] [2024-03-22 01:53:54] PATCH /note/update {"id":8,"title":"Thursday note","note_text":"Again its project time."
```

Figure 2: Application logs

- **DatabaseConstants (for storing all the SQL query string)**
  - **Query to select all notes**
  - **Query to select note based on note id**
  - **Query to create new note**
  - **Query to update existing note**
  - **Query to delete note**

6) .env

This file contains the configurations elements of project. In our project it is specifically



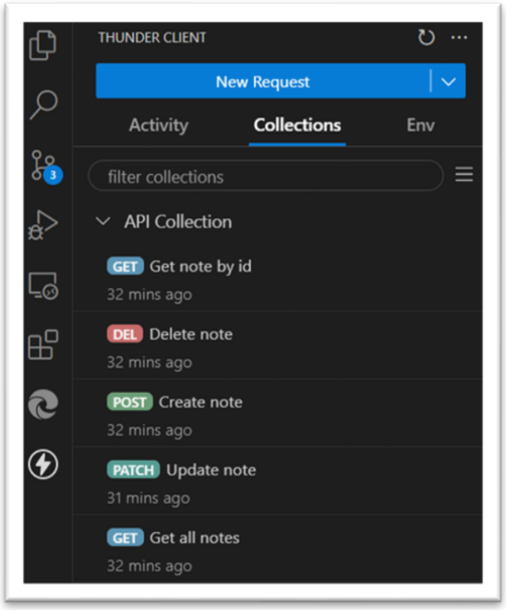
```
MYSQL_HOST='localhost'
MYSQL_USERNAME='root'
MYSQL_PASSWORD='root'
MYSQL_DATABASE='nodedb'
```

Figure 3: .env file content

used for database credentials and information.

7) API Collection.json

This file is an import file for API client such as postman, thunderclient, etc. It contains all the API request of this project to test.

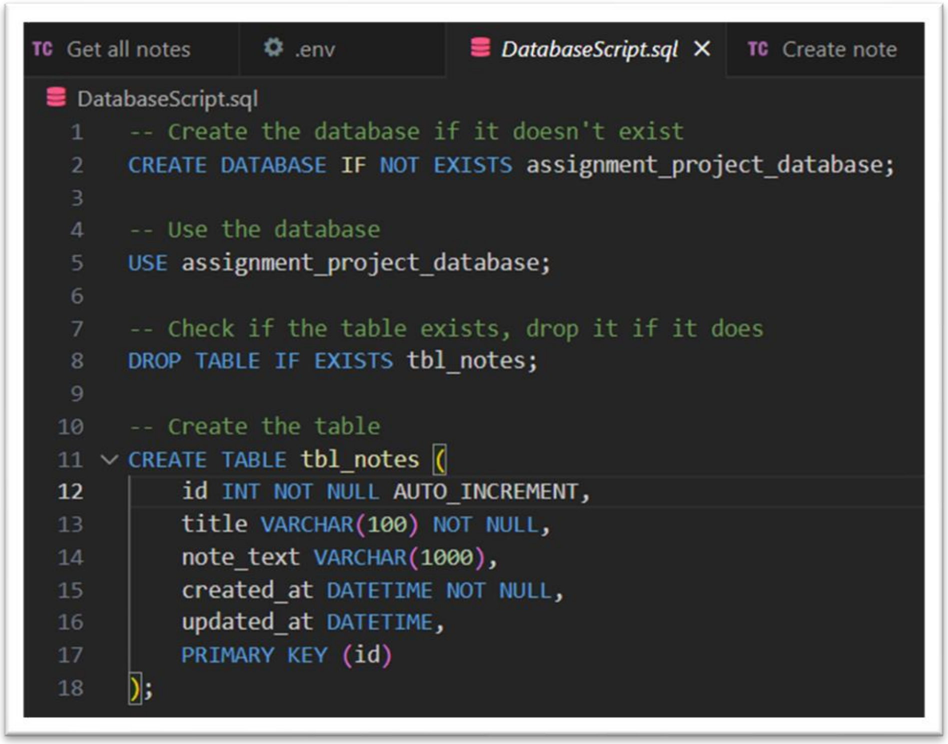


GET Get note by id
32 mins ago
DEL Delete note
32 mins ago
POST Create note
32 mins ago
PATCH Update note
31 mins ago
GET Get all notes
32 mins ago

Figure 4: API collection

8) DatabaseScript.sql

This SQL script file contains the SQL commands to setup database for this project. It contains database creation and table creation command with exist check.

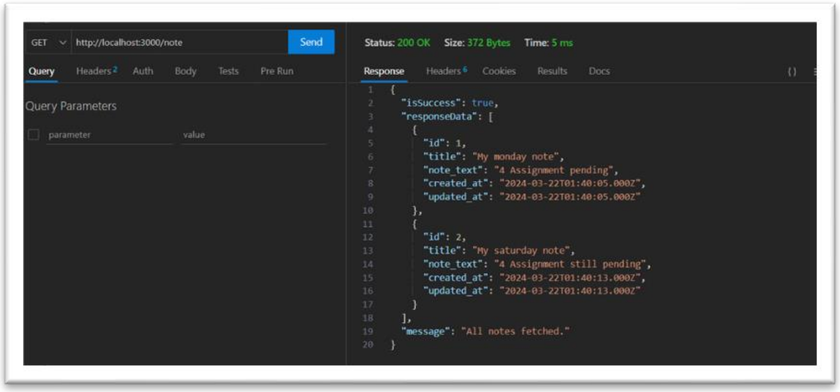
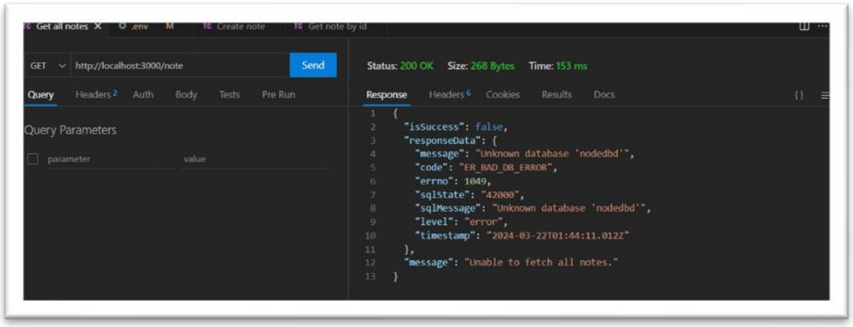


```sql
-- Create the database if it doesn't exist
CREATE DATABASE IF NOT EXISTS assignment_project_database;

-- Use the database
USE assignment_project_database;

-- Check if the table exists, drop it if it does
DROP TABLE IF EXISTS tbl_notes;

-- Create the table
CREATE TABLE tbl_notes (
    id INT NOT NULL AUTO_INCREMENT,
    title VARCHAR(100) NOT NULL,
    note_text VARCHAR(1000),
    created_at DATETIME NOT NULL,
    updated_at DATETIME,
    PRIMARY KEY (id)
);
```

Figure 5: Database script

APIs

1) Get all notes

Get all notes	
URL	/note
Request parameter	-
Request method	GET
Request body	-
Response body	`{ "isSuccess": true, "responseData": [ { "id": 1, "title": "My monday note", "note_text": "4 Assignment pending", "created_at": "2024-03-22T01:40:05.000Z", "updated_at": "2024-03-22T01:40:05.000Z" } ], "message": "All notes fetched." }`
Success test	
Fail test	

```
{
  "isSuccess": true,
  "responseData": [
    {
      "id": 1,
      "title": "My monday note",
      "note_text": "4 Assignment pending",
      "created_at": "2024-03-22T01:40:05.000Z",
      "updated_at": "2024-03-22T01:40:05.000Z"
    }
  ],
  "message": "All notes fetched."
}
```

Success test:

GET http://localhost:3000/note

Status: 200 OK Size: 372 Bytes Time: 5 ms

```
{
  "isSuccess": true,
  "responseData": [
    {
      "id": 1,
      "title": "My monday note",
      "note_text": "4 Assignment pending",
      "created_at": "2024-03-22T01:40:05.000Z",
      "updated_at": "2024-03-22T01:40:05.000Z"
    },
    {
      "id": 2,
      "title": "My saturday note",
      "note_text": "4 Assignment still pending",
      "created_at": "2024-03-22T01:40:13.000Z",
      "updated_at": "2024-03-22T01:40:13.000Z"
    }
  ],
  "message": "All notes fetched."
}
```

Fail test:

GET http://localhost:3000/note

Status: 200 OK Size: 268 Bytes Time: 153 ms

```
{
  "isSuccess": false,
  "responseData": {
    "message": "Unknown database 'nodedbd'",
    "code": "ER_BAD_DB_ERROR",
    "errno": 1049,
    "sqlState": "42000",
    "sqlMessage": "Unknown database 'nodedbd'",
    "level": "error",
    "timestamp": "2024-03-22T01:44:11.012Z"
  },
  "message": "Unable to fetch all notes."
}
```

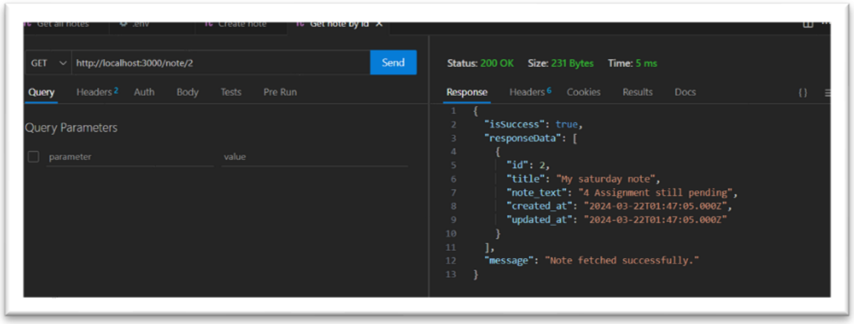
2) Get note by id

Get note by id	
URL	/note/21
Request parameter	Note id
Request method	GET
Request body	-

Response body

```
{
  "isSuccess": true,
  "responseData": [
    {
      "id": 2,
      "title": "My saturday note",
      "note_text": "4 Assignment still pending",
      "created_at": "2024-03-22T01:47:05.000Z",
      "updated_at": "2024-03-22T01:47:05.000Z"
    }
  ],
  "message": "Note fetched successfully."
}
```

Success test

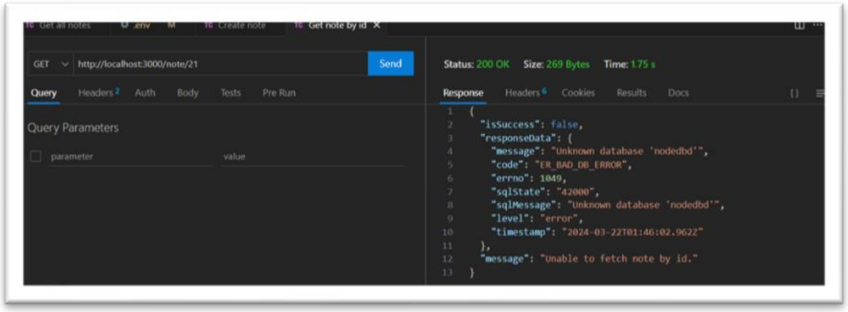


GET http://localhost:3000/note/2

Status: 200 OK Size: 231 Bytes Time: 5 ms

```
{
  "isSuccess": true,
  "responseData": [
    {
      "id": 2,
      "title": "My saturday note",
      "note_text": "4 Assignment still pending",
      "created_at": "2024-03-22T01:47:05.000Z",
      "updated_at": "2024-03-22T01:47:05.000Z"
    }
  ],
  "message": "Note fetched successfully."
}
```

Fail test



GET http://localhost:3000/note/21

Status: 200 OK Size: 269 Bytes Time: 1.75 s

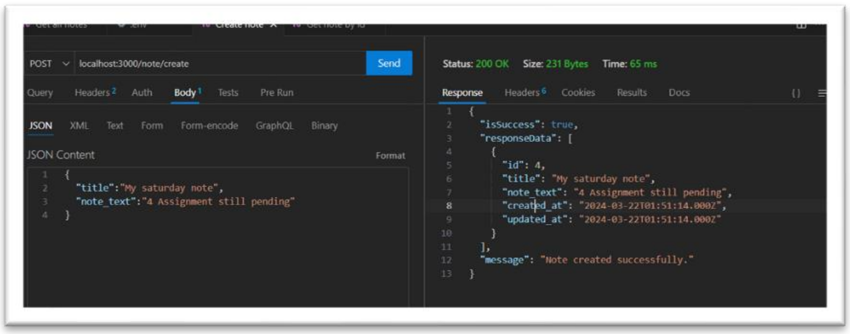
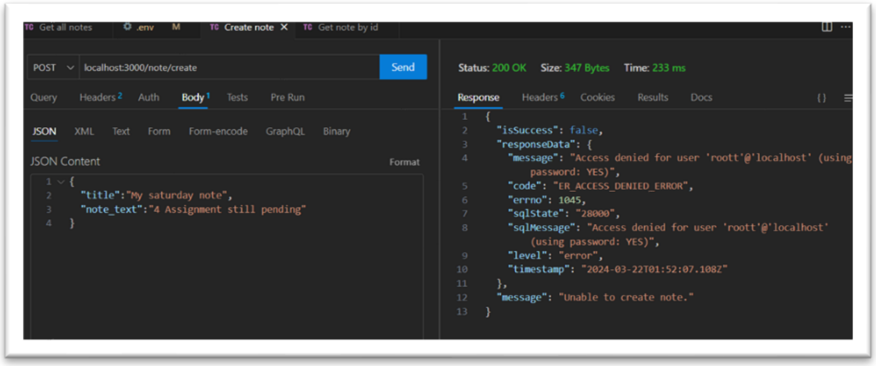
```
{
  "isSuccess": false,
  "responseData": {
    "message": "Unknown database 'nodedbd'",
    "code": "ER_BAD_DB_ERROR",
    "errno": 1049,
    "sqlState": "42000",
    "sqlMessage": "Unknown database 'nodedbd'",
    "level": "error",
    "timestamp": "2024-03-22T01:46:02.962Z"
  },
  "message": "Unable to fetch note by id."
}
```

3) Create Note

Create note	
URL	/note/create
Request parameter	-
Request method	POST

Request body	`{`
	`  "title":"My saturday note",`
	`  "note_text":"4 Assignment still pending"`
	`}`

Response body	`{`
	`  "isSuccess": true,`
	`  "responseData": [`
	`    {`
	`      "id": 4,`
	`      "title": "My saturday note",`
	`      "note_text": "4 Assignment still pending",`
	`      "created_at": "2024-03-22T01:51:14.000Z",`
	`      "updated_at": "2024-03-22T01:51:14.000Z"`
	`    }`
	`  ],`
	`  "message": "Note created successfully."`
	`}`

Success test	
Fail test	

Success test:

```
POST  localhost:3000/note/create
{
  "title":"My saturday note",
  "note_text":"4 Assignment still pending"
}
```

```
Status: 200 OK   Size: 231 Bytes   Time: 65 ms
{
  "isSuccess": true,
  "responseData": [
    {
      "id": 4,
      "title": "My saturday note",
      "note_text": "4 Assignment still pending",
      "created_at": "2024-03-22T01:51:14.000Z",
      "updated_at": "2024-03-22T01:51:14.000Z"
    }
  ],
  "message": "Note created successfully."
}
```

Fail test:

```
POST  localhost:3000/note/create
{
  "title":"My saturday note",
  "note_text":"4 Assignment still pending"
}
```

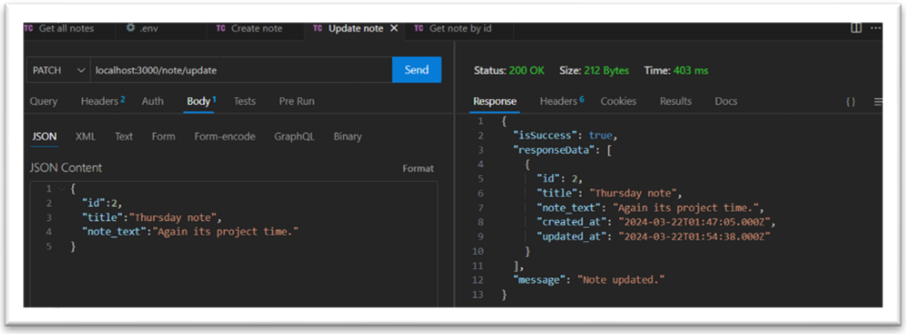
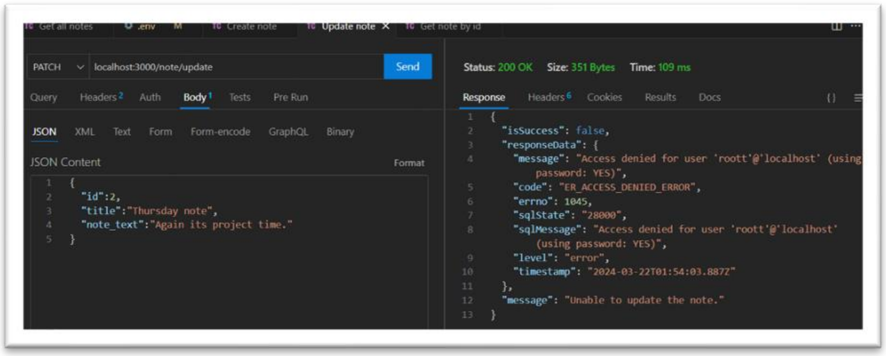
```
Status: 200 OK   Size: 347 Bytes   Time: 233 ms
{
  "isSuccess": false,
  "responseData": {
    "message": "Access denied for user 'roott'@'localhost' (using
        password: YES)",
    "code": "ER_ACCESS_DENIED_ERROR",
    "errno": 1045,
    "sqlState": "28000",
    "sqlMessage": "Access denied for user 'roott'@'localhost'
        (using password: YES)",
    "level": "error",
    "timestamp": "2024-03-22T01:52:07.108Z"
  },
  "message": "Unable to create note."
}
```

4) Update note

Update note	
URL	/note/update
Request parameter	-
Request method	PATCH

Request body	`{`
	`  "id":8,`
	`  "title":"title 1 update 4",`
	`  "note_text":"note text it is updated"`
	`}`

Response body	`{`
	`  "isSuccess": true,`
	`  "responseData": [`
	`    {`
	`      "id": 4,`
	`      "title": "My saturday note",`
	`      "note_text": "4 Assignment still pending",`
	`      "created_at": "2024-03-22T01:51:14.000Z",`
	`      "updated_at": "2024-03-22T01:51:14.000Z"`
	`    }`
	`  ],`
	`  "message": "Note created successfully."`
	`}`

Success test	
Fail test	

Success test:

PATCH localhost:3000/note/update

Status: 200 OK Size: 212 Bytes Time: 403 ms

```
{
  "id":2,
  "title":"Thursday note",
  "note_text":"Again its project time."
}
```

```
{
  "isSuccess": true,
  "responseData": [
    {
      "id": 2,
      "title": "Thursday note",
      "note_text": "Again its project time.",
      "created_at": "2024-03-22T01:47:05.000Z",
      "updated_at": "2024-03-22T01:54:38.000Z"
    }
  ],
  "message": "Note updated."
}
```

Fail test:

PATCH localhost:3000/note/update

Status: 200 OK Size: 351 Bytes Time: 109 ms

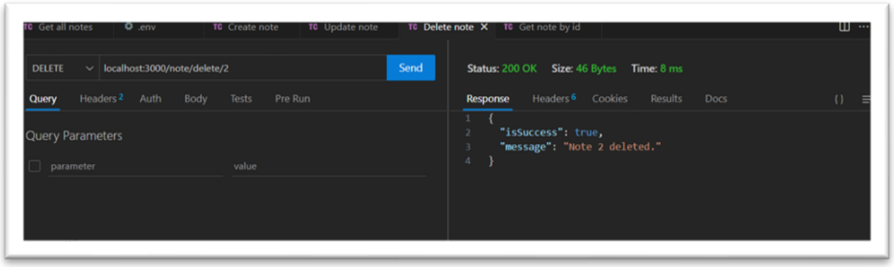
```
{
  "id":2,
  "title":"Thursday note",
  "note_text":"Again its project time."
}
```

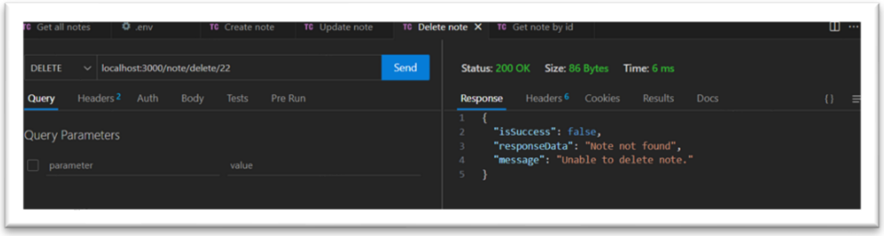
```
{
  "isSuccess": false,
  "responseData": {
    "message": "Access denied for user 'roott'@'localhost' (using
      password: YES)",
    "code": "ER_ACCESS_DENIED_ERROR",
    "errno": 1045,
    "sqlState": "28000",
    "sqlMessage": "Access denied for user 'roott'@'localhost'
      (using password: YES)",
    "level": "error",
    "timestamp": "2024-03-22T01:54:03.887Z"
  },
  "message": "Unable to update the note."
}
```

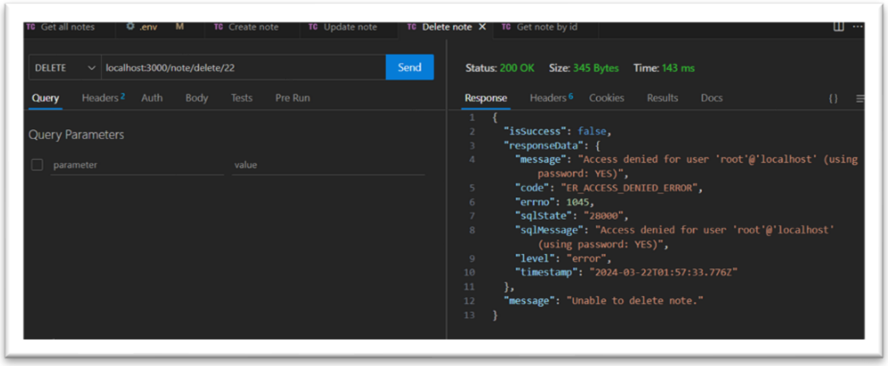
5) Delete note

Delete note	
URL	/note/delete/2
Request parameter	Note id
Request method	DELETE
Request body	-

Response body	`{`
	`  "isSuccess": true,`
	`  "message": "Note 3 deleted."`
	`}`

Success test	
	DELETE localhost:3000/note/delete/2 Send
	Status: 200 OK Size: 46 Bytes Time: 8 ms
	`{`
	`  "isSuccess": true,`
	`  "message": "Note 2 deleted."`
	`}`

Fail test 1	
	DELETE localhost:3000/note/delete/22 Send
	Status: 200 OK Size: 86 Bytes Time: 6 ms
	`{`
	`  "isSuccess": false,`
	`  "responseData": "Note not found",`
	`  "message": "Unable to delete note."`
	`}`

Fail test 2	
	DELETE localhost:3000/note/delete/22 Send
	Status: 200 OK Size: 345 Bytes Time: 143 ms
	`{`
	`  "isSuccess": false,`
	`  "responseData": {`
	`    "message": "Access denied for user 'root'@'localhost' (using password: YES)",`
	`    "code": "ER_ACCESS_DENIED_ERROR",`
	`    "errno": 1045,`
	`    "sqlState": "28000",`
	`    "sqlMessage": "Access denied for user 'root'@'localhost' (using password: YES)",`
	`    "level": "error",`
	`    "timestamp": "2024-03-22T01:57:33.776Z"`
	`  },`
	`  "message": "Unable to delete note."`
	`}`

Special note

I have 4.5 years of experience in software development in technology such as .NET, RabbitMQ, Spring boot (started in Dalhousie University course project). Though I did not touched Node.js in past, I tried my best to learn it and apply software development concept which I was aware of (for developing application using .Net and Spring boot). I can still develop the frontend part for this application, but due to time constraints and work load of assignment plus project (specially next week), I will not be able to do that.
Thank you SHIFTKEY !