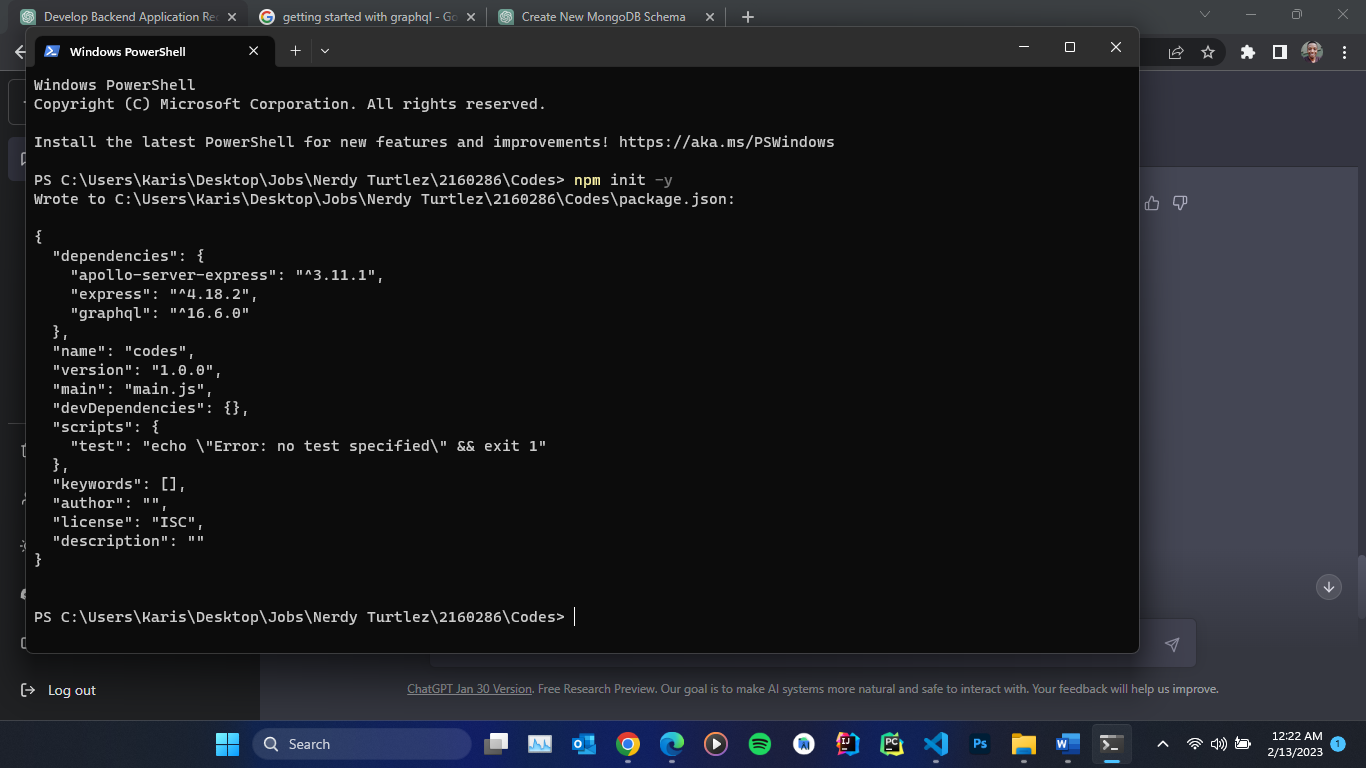
The implementation of the GraphQL API for a User and Employee database involves several steps that are essential for the proper functioning of the application. In this report, we outline the steps involved in setting up the project, connecting to the MongoDB database, creating the MongoDB schemas, defining the GraphQL schema, and defining the resolvers. The steps involved in the implementation of the GraphQL API are described below.

**Setup**

The project was set up by initializing a Node.js project and installing the required dependencies. The project structure was created by creating directories and files, including the schema directory, the resolvers directory, and the index.js file. The index.js file served as the entry point of the application.

**Initializing the Node.js project**

Initializing the Node.js project by running ‘npm init -y’ command



This creates a ‘package.json’ file, which contains information about the project and its dependencies.

**Installing dependencies**

We run the following commands to install the required dependencies:

*npm install apollo-server-express*

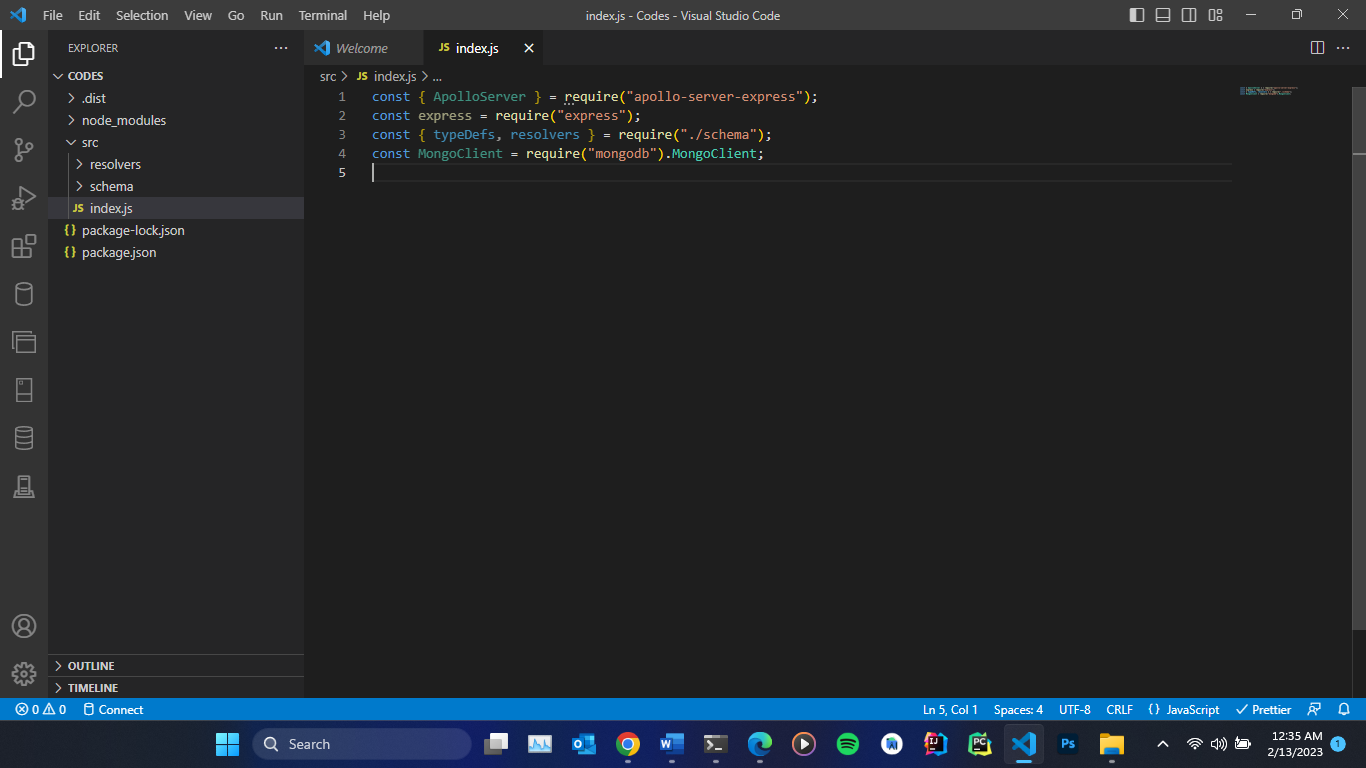
*npm install express*

*npm install graphql*

*npm install mongodb*

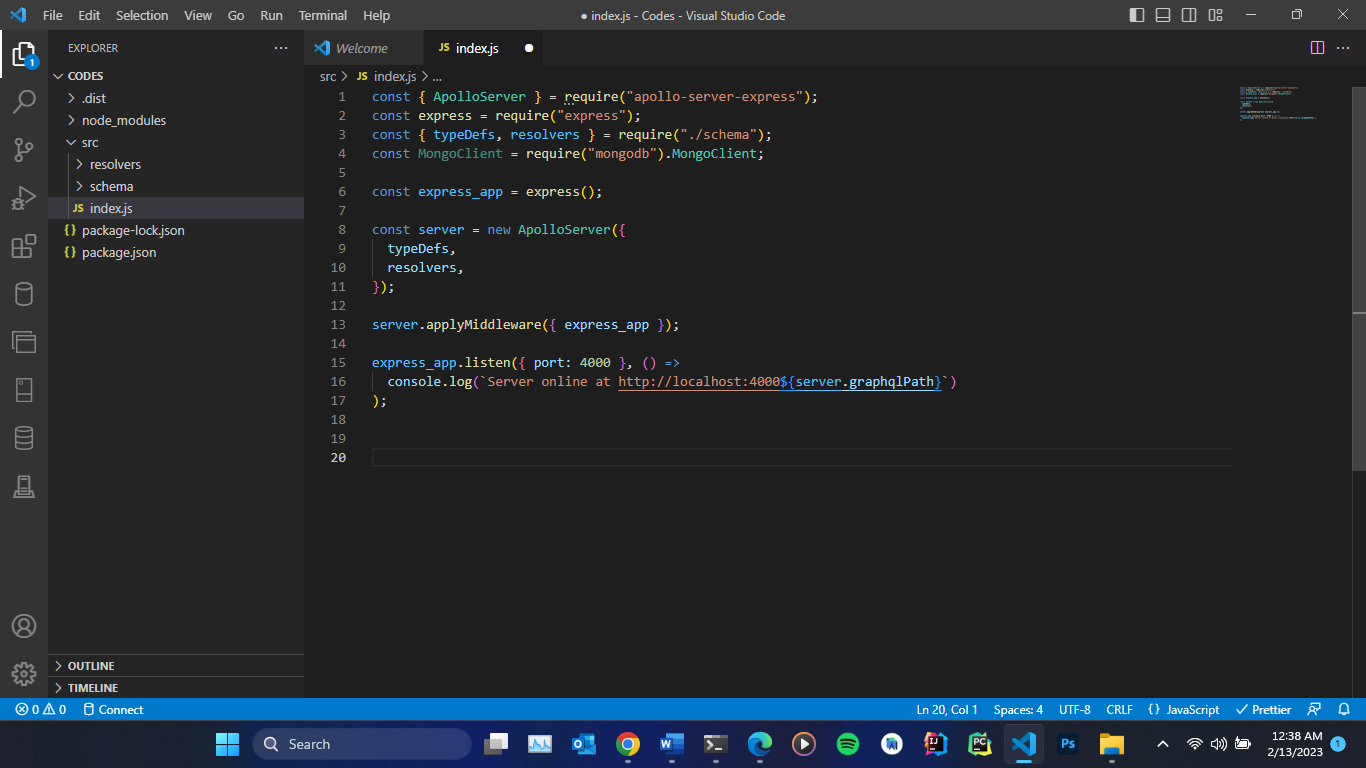
**Importing the required dependencies**

In the ‘index.js’ file, we import the dependencies previously installed



**Creating an Express application**

In the index.js file, we then create an Express application with the following code:



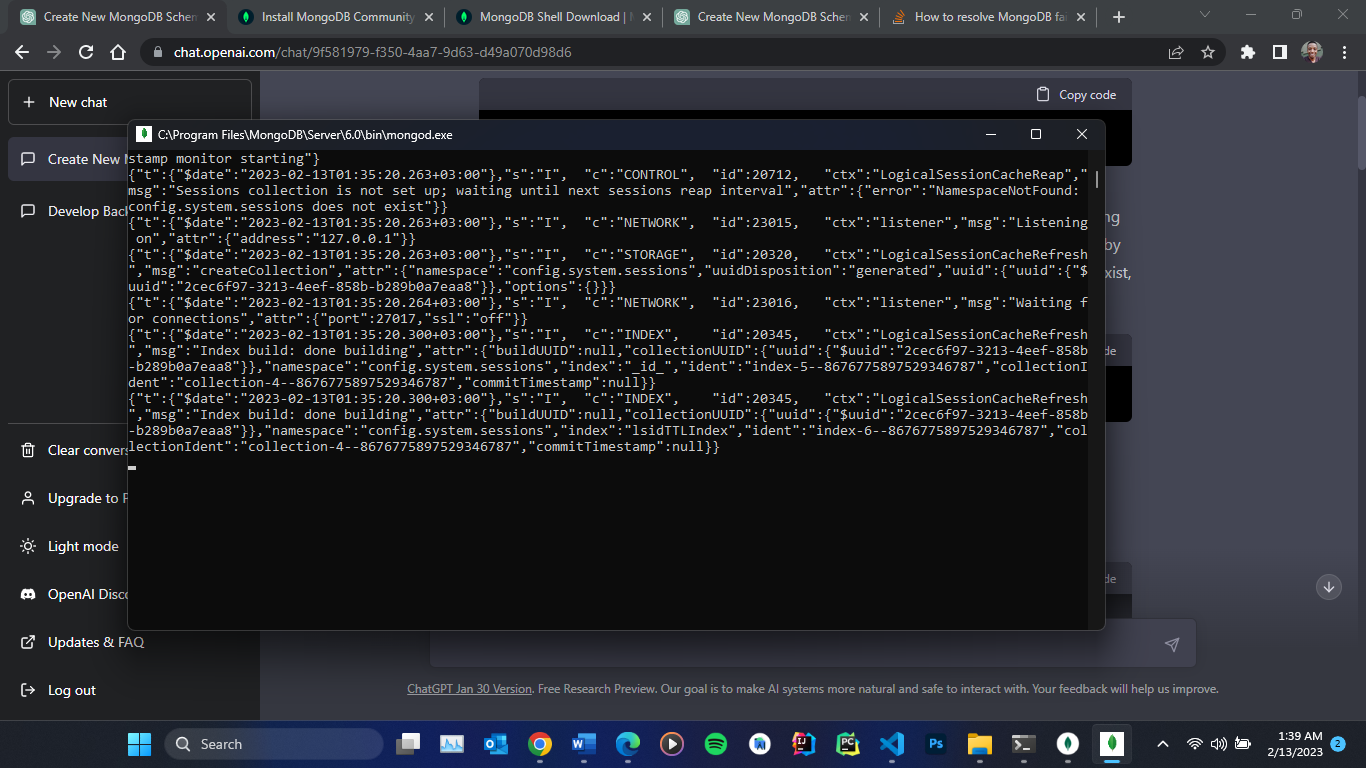
With these steps, we have successfully set up the project structure for the Node.js, Express, GraphQL, and MongoDB application.

**MongoDB Schemas**

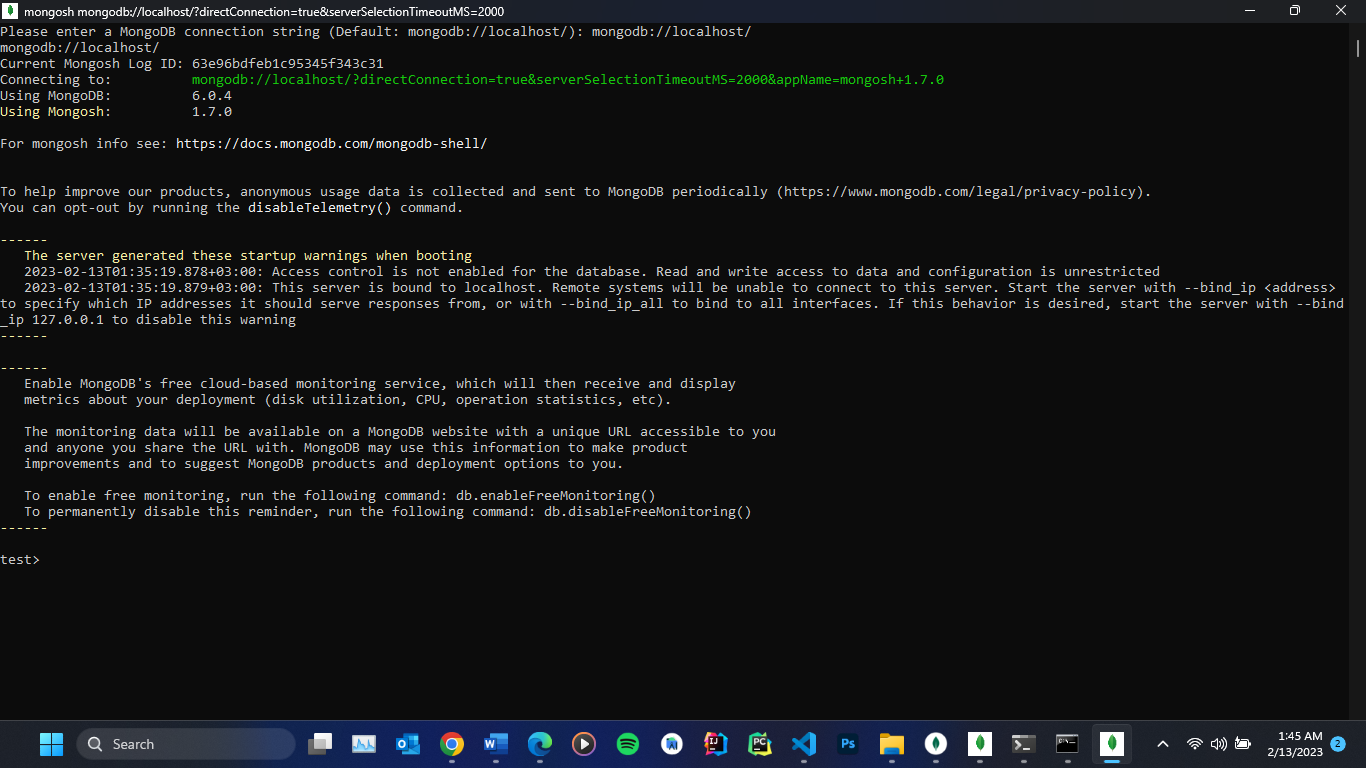
The MongoDB database was connected to and the User and Employee collections were created. The collections were verified to ensure that they had been created successfully.

**Connecting to MongoDB**

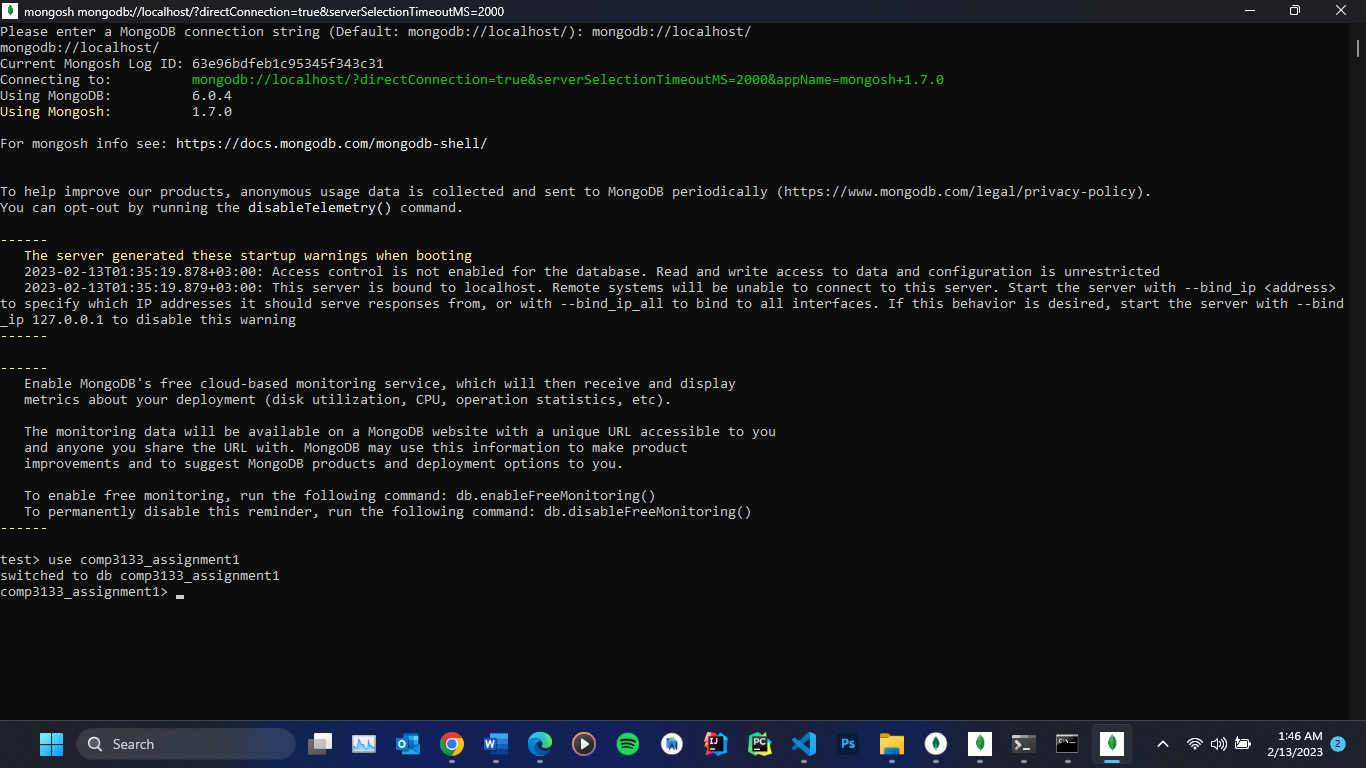
Starting the server. We run the ‘mongod’ command



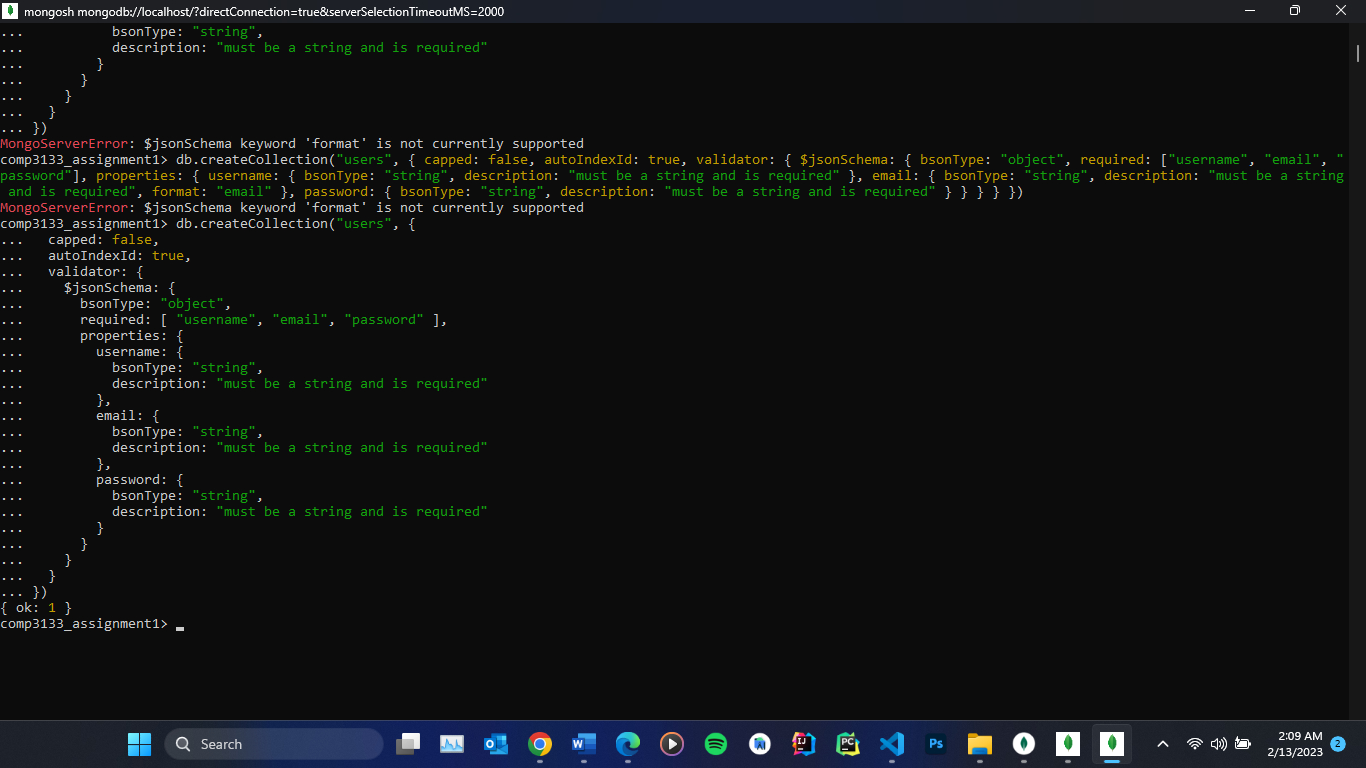
Connecting to a local instance of MongoDB



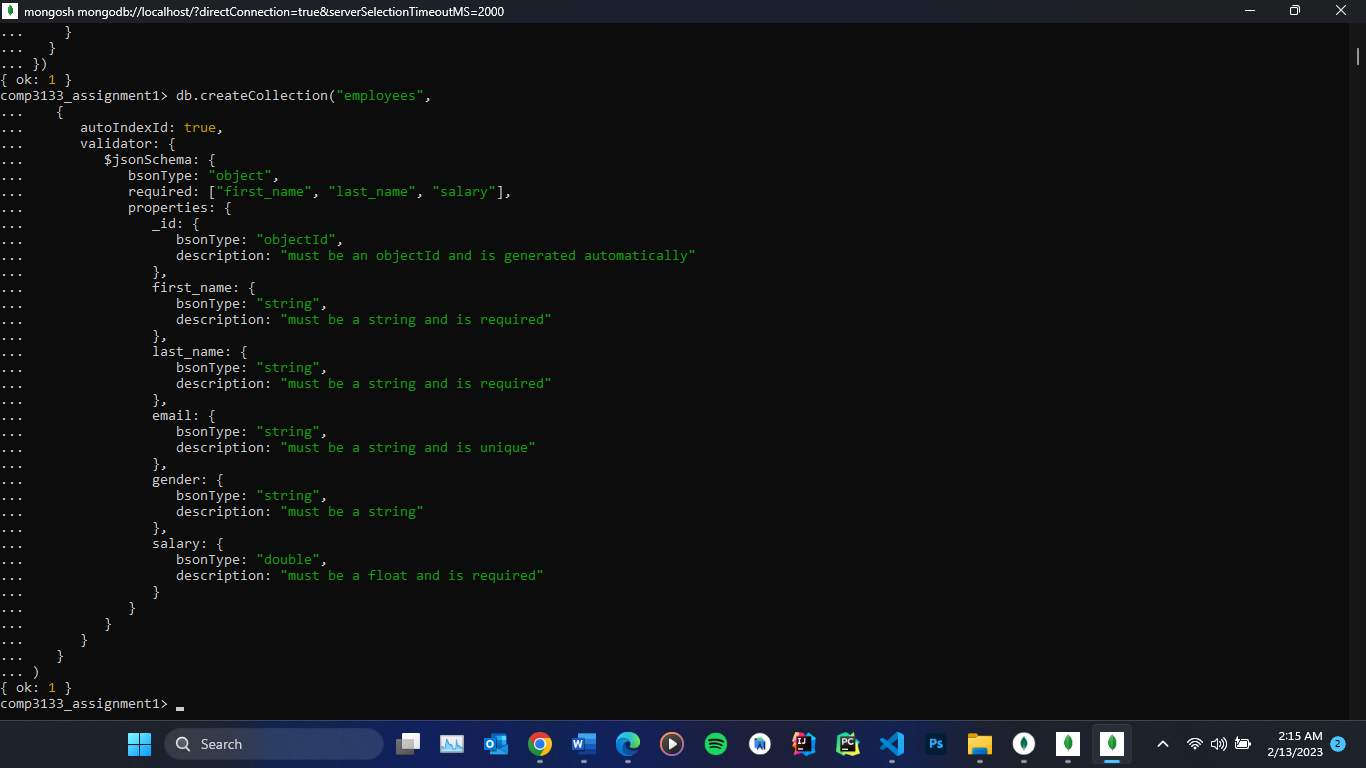
Creating a new database



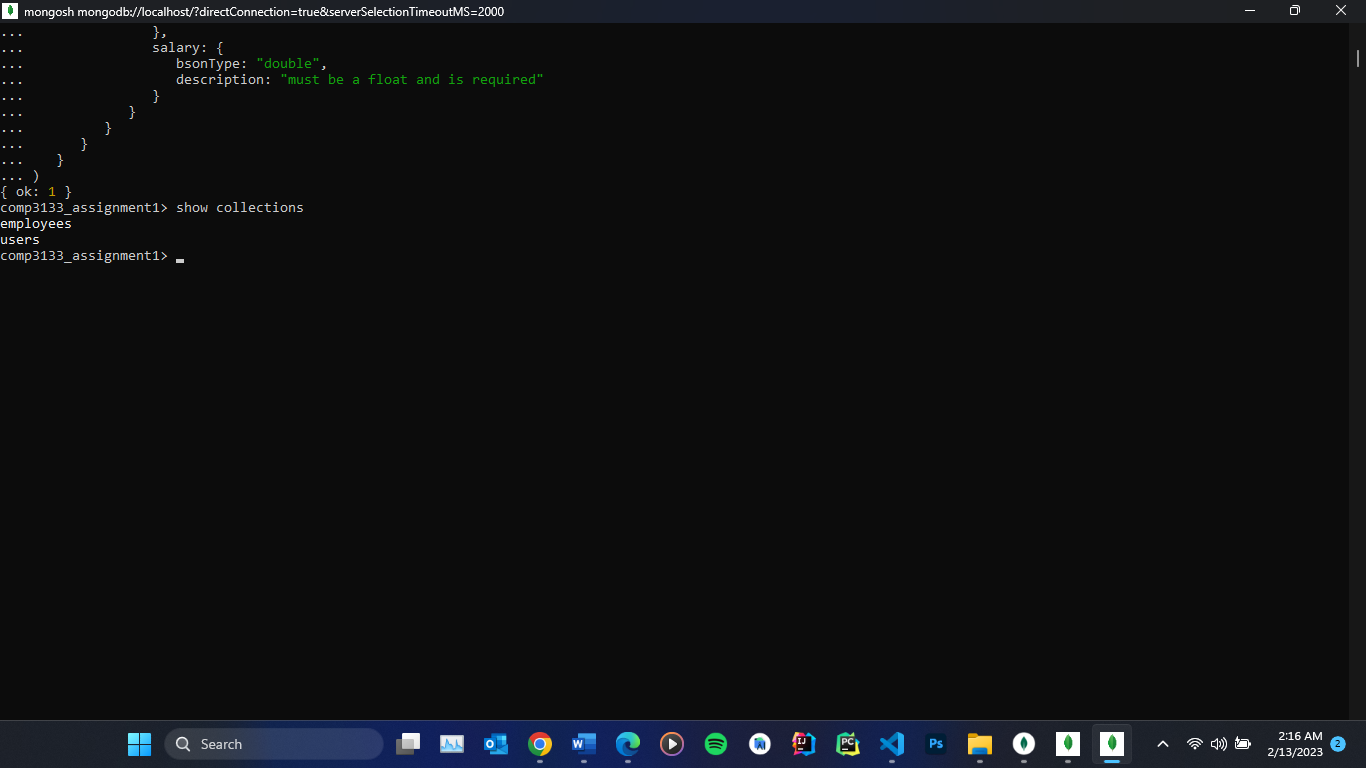
Creating the Users Collection: We use the db.createCollection() method to create the Users Collection with the specified fields and constraints



Creating the Employees Collection: We use the db.createCollection() method to create the Employees Collection with the specified fields and constraints



Verify the collections have been created



**GraphQL APIs**

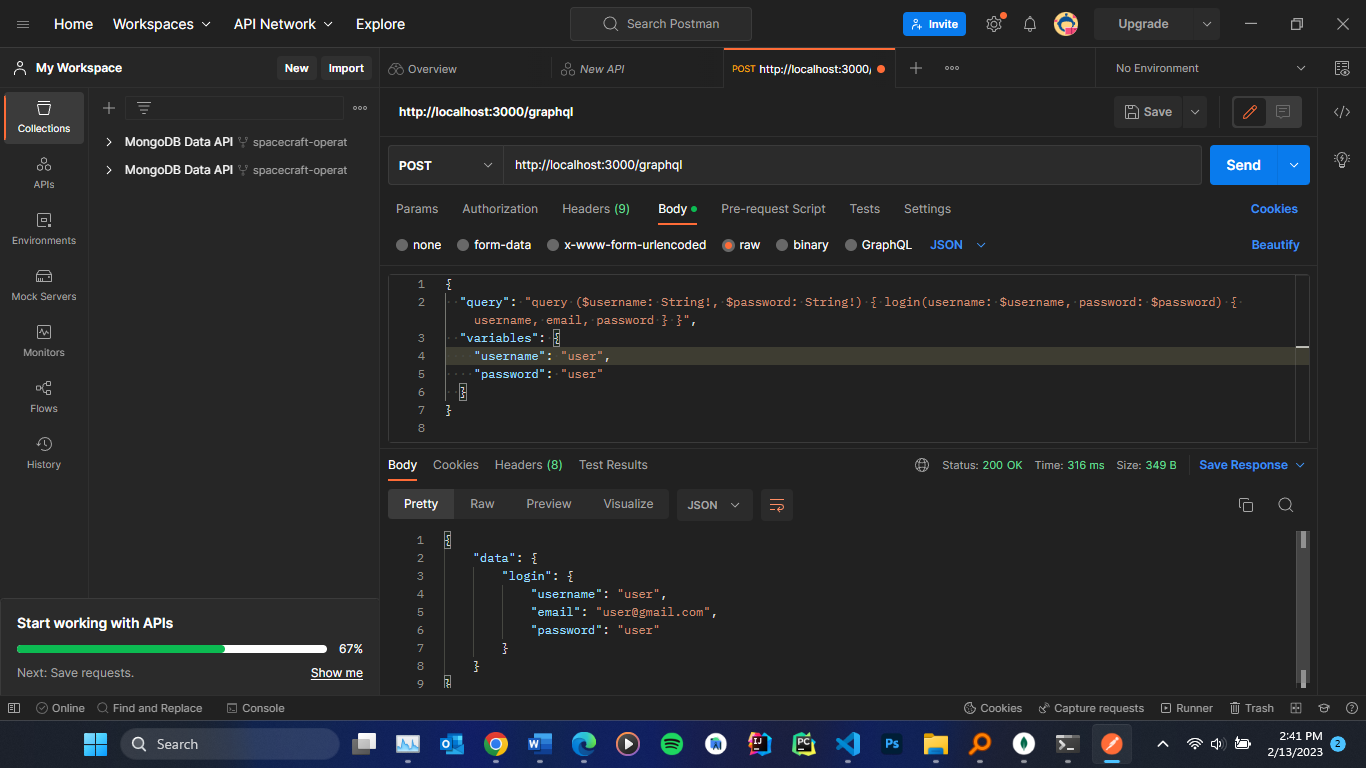
The implementation of the GraphQL API involved the creation of the User and Employee models using the mongoose.model() function. The GraphQL schema was defined using the GraphQL schema language, which included the definition of two types, User and Employee, as well as the Query and Mutation types. The resolvers for the fields in the GraphQL schema were defined, which included methods for performing the respective operations on the User and Employee models.

**Test the GraphQL APIs**

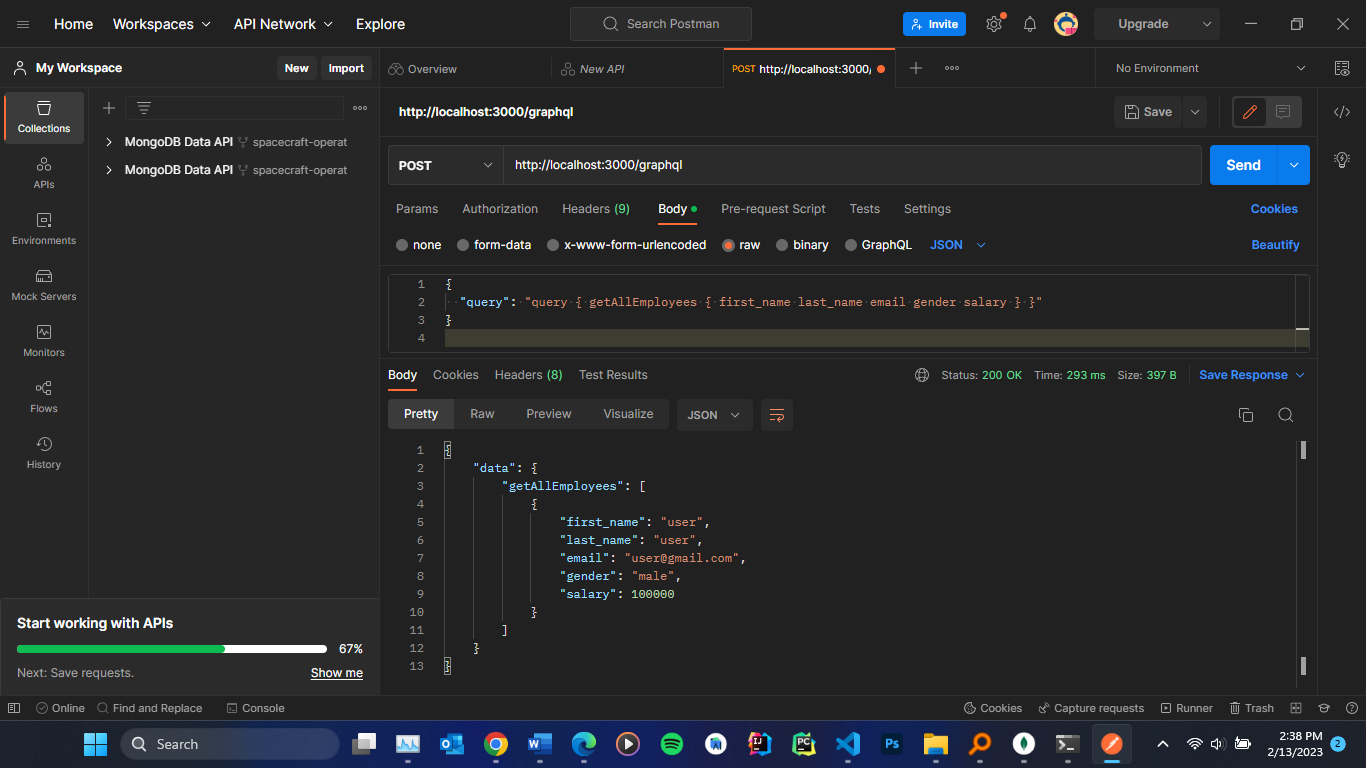
The GraphQL APIs were tested by performing queries and mutations, including logging in a user, retrieving all employees, searching for an employee by ID, adding a new user, adding a new employee, updating an employee by ID, and deleting an employee by ID. The results of these tests were used to verify that the GraphQL APIs were working as expected.

**Queries:**

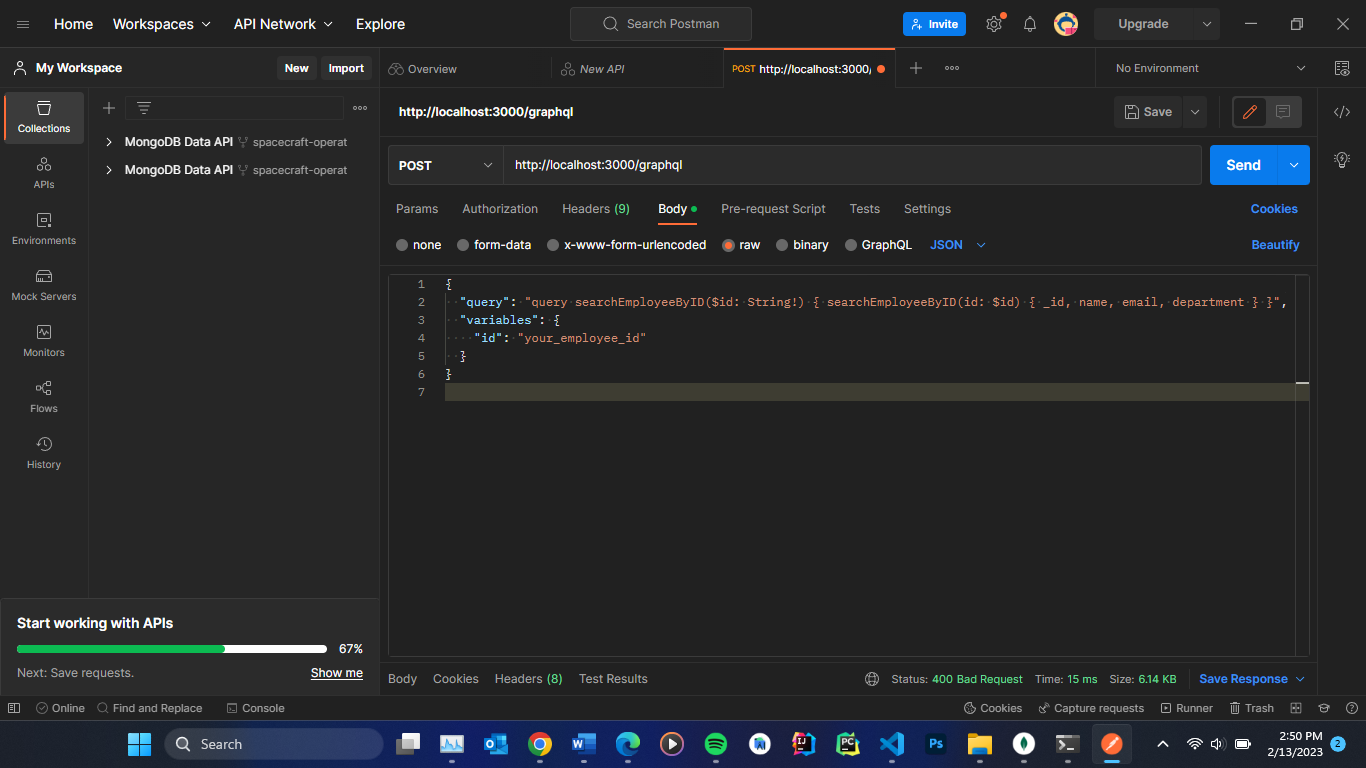
login: Returns a user matching the given username and password.



getAllEmployees: Returns all the employees stored in the database.

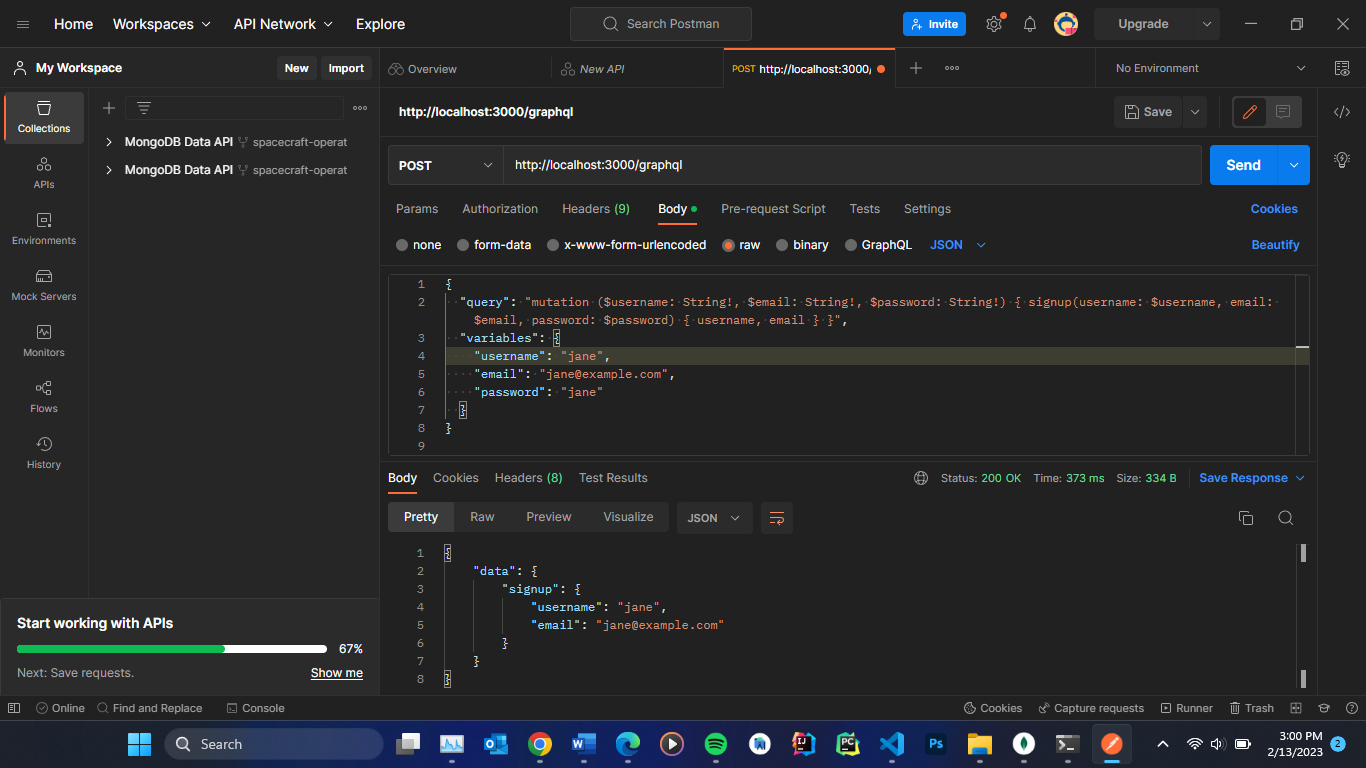


searchEmployeeByID: Returns an employee whose id matches the given id.

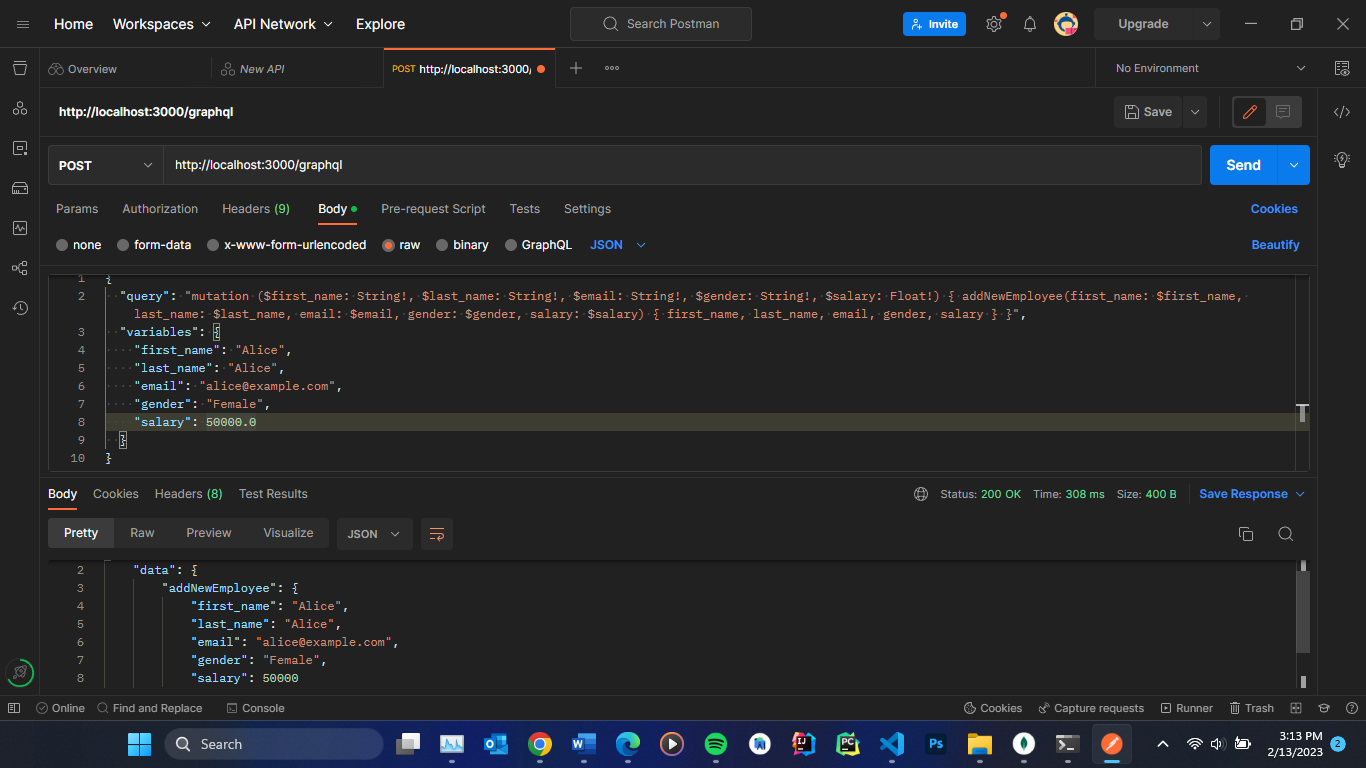


**Mutations:**

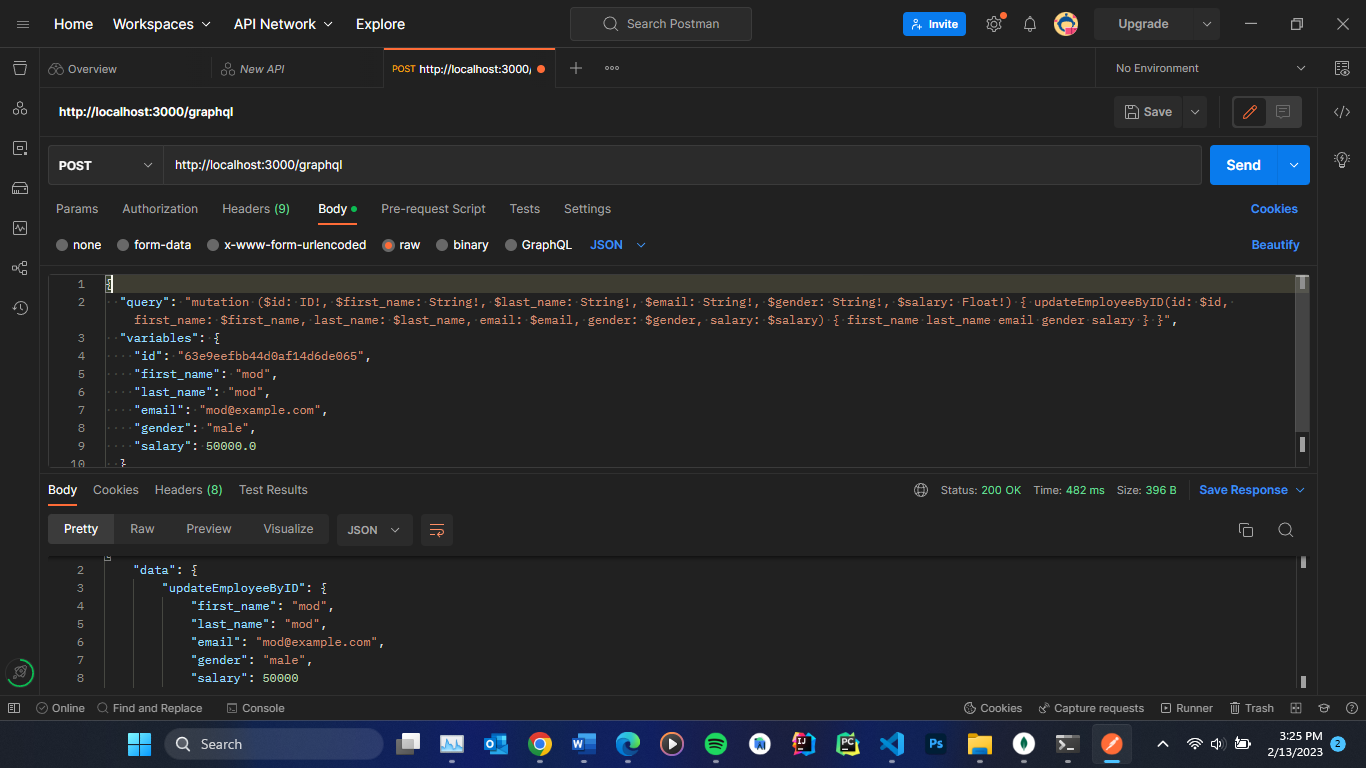
signup: Adds a new user to the database with the given username, email, and password.



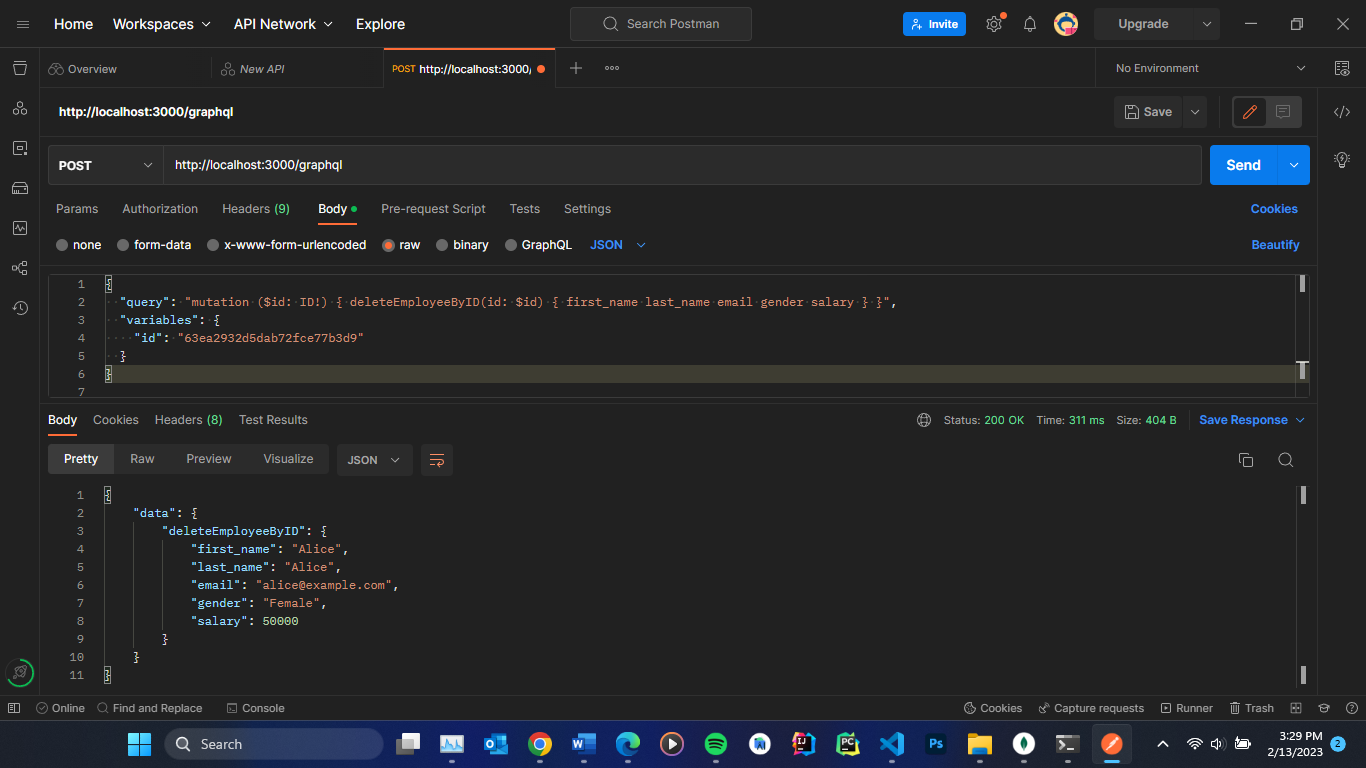
addNewEmployee: Adds a new employee to the database with the given first name, last name, email, gender, and salary.



updateEmployeeByID: Updates an employee whose id matches the given id with the new values provided for the first name, last name, email, gender, and salary.



deleteEmployeeByID: Deletes an employee whose id matches the given id.



The implementation of the GraphQL API for a User and Employee database involved several steps that were essential for the proper functioning of the application. The steps involved in the implementation of the GraphQL API, including setting up the project, connecting to the MongoDB database, creating the MongoDB schemas, defining the GraphQL schema, and defining the resolvers, were described in detail in this report.

**Sample User Details**

*Users*

**Username**: "user"

**Email**: "user@gmail.com"

**Password**: "user"

*Employees*

\_**id**: 63e9eefbb44d0af14d6de065

**first\_name**: "mod"

**last\_name**: "mod"

**email**: "mod@example.com"

**gender**: "male"

**salary**: 50000