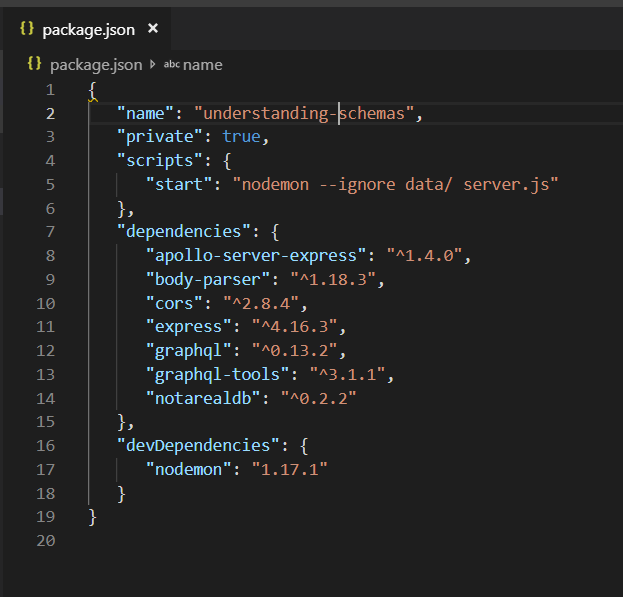
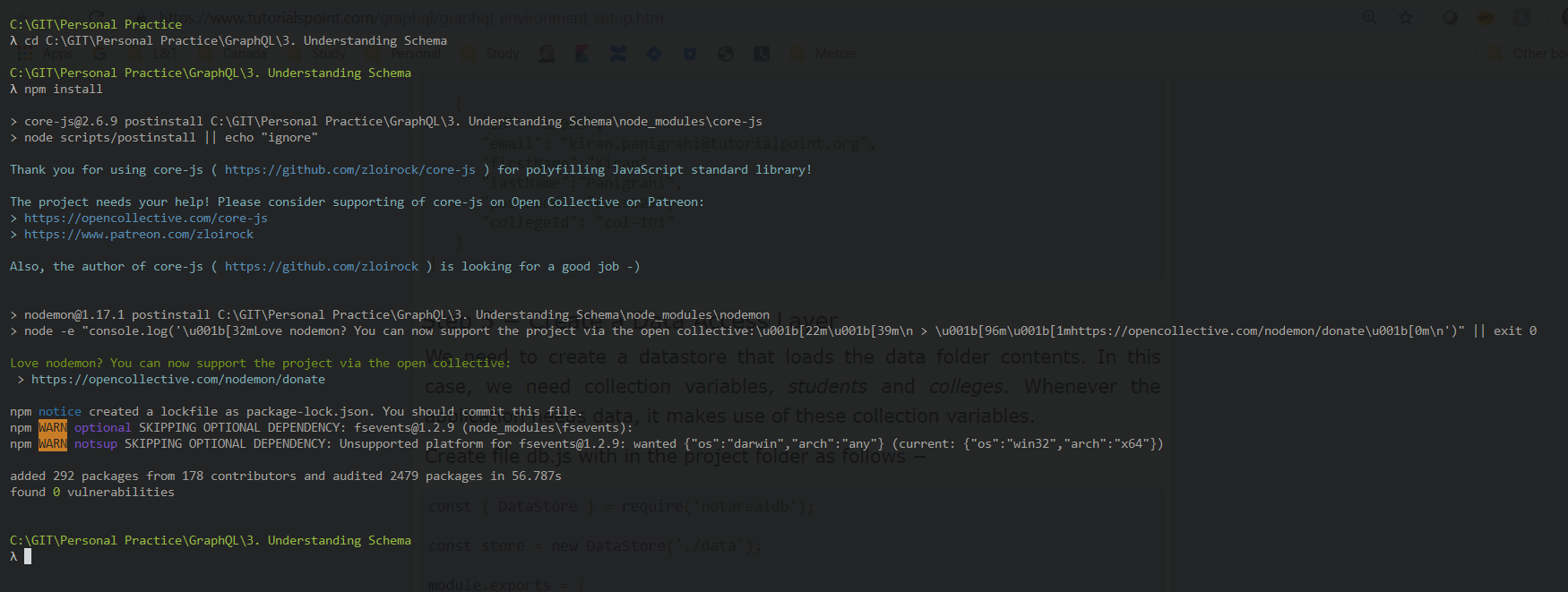
It is best practice to return an object in mutation.

For example, the client application wants to fetch student and college details. In this case, rather than making two different requests, we can create a query that returns an object containing students and their college details.

## Step 1 − Download and Install Required Dependencies for the Project

Create new package.json file with below code and install all dependencies.

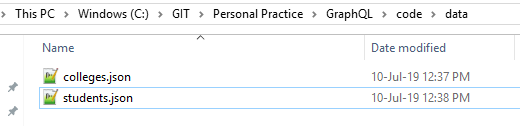


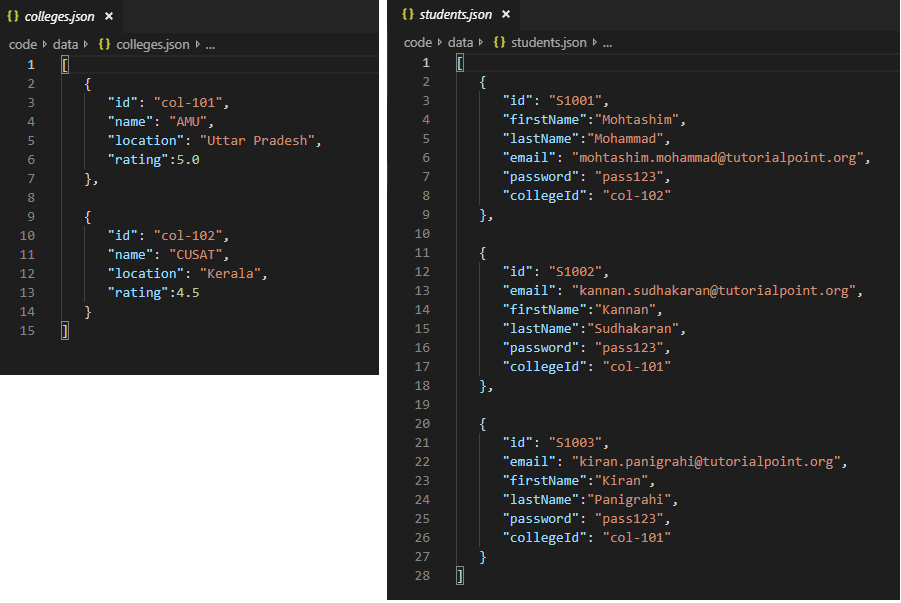


**Create Flat File Database in Data Folder**

we use flat files to store and retrieve data.

Create a folder data and add two files **students.json** and **colleges.json**.





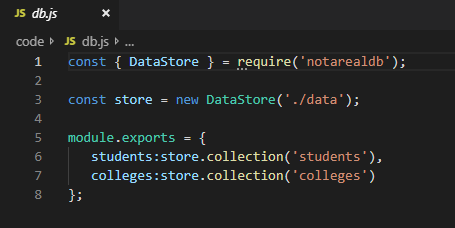
**Create a Data Access Layer**

We need to create a datastore that loads the data folder contents.

In this case, we need collection variables, *students* and *colleges*.

Whenever the application needs data, it makes use of these collection variables.

Create file db.js with in the project folder as follows



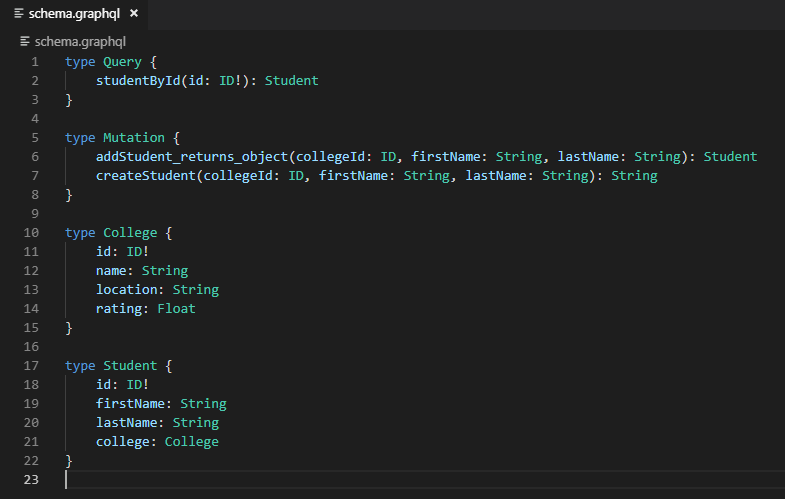
## Step 2 − Edit Schema

Add **schema.graphql** file with following code.

Add a new method named **addStudent** which returns object in mutation type of **schema.graphql**.

Let us learn how to access the college details through student details.

Add college type in the schema file.



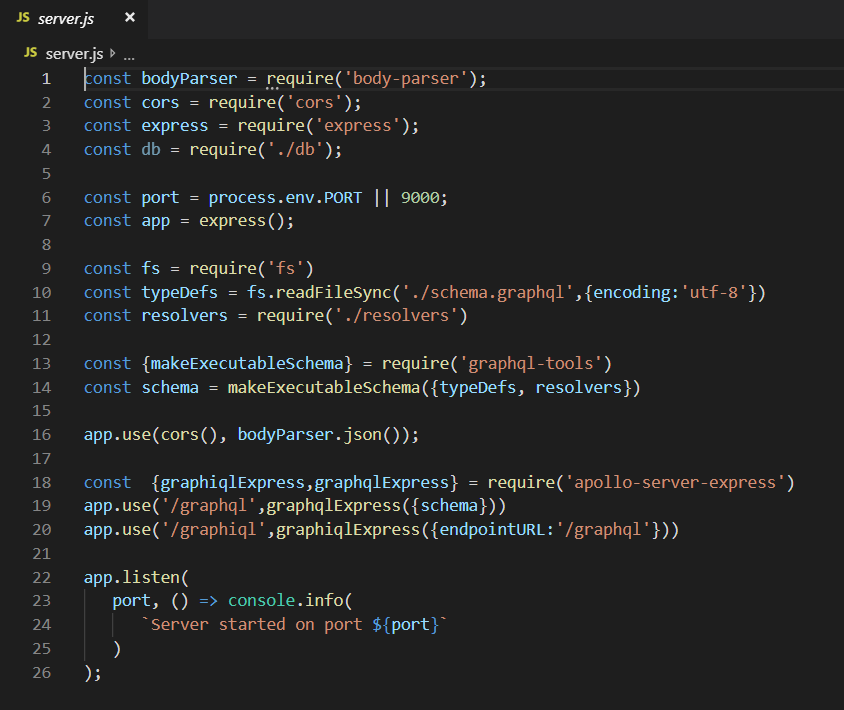
## Step 3 − Update Resolver

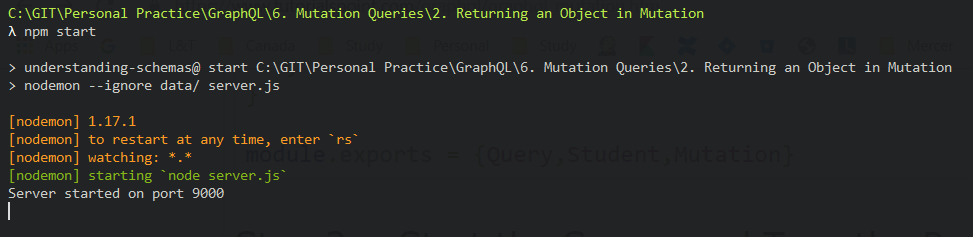
Create a file **resolvers.js** with following code.



## Step 4 − Run the Application

Create a server.js file.





Next step is to open browser and type the URL [**http://localhost:9000/graphiql**](http://localhost:9000/graphiql).

## Testing

Type the following query in the editor –

mutation {

addStudent\_returns\_object(collegeId:"col-101",firstName:"Susan",lastName:"George") {

id

firstName

college{

id

name

}

}

}

The above query adds a new student and retrieves the student object along with college object. This saves round trips to the server.

The response is as given below

{

"data": {

"addStudent\_returns\_object": {

"id": "rJUCCt9ZH",

"firstName": "Susan",

"college": {

"id": "col-101",

"name": "AMU"

}

}

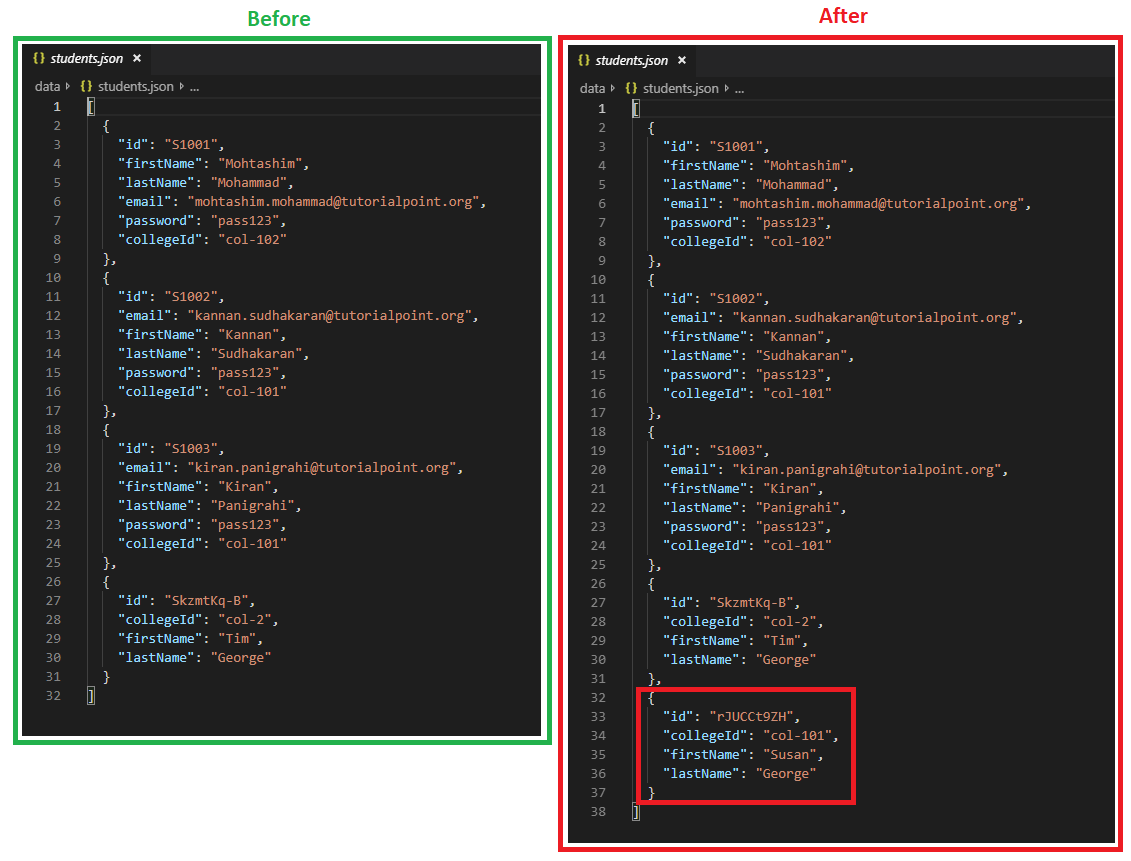
}

}

## On Browser



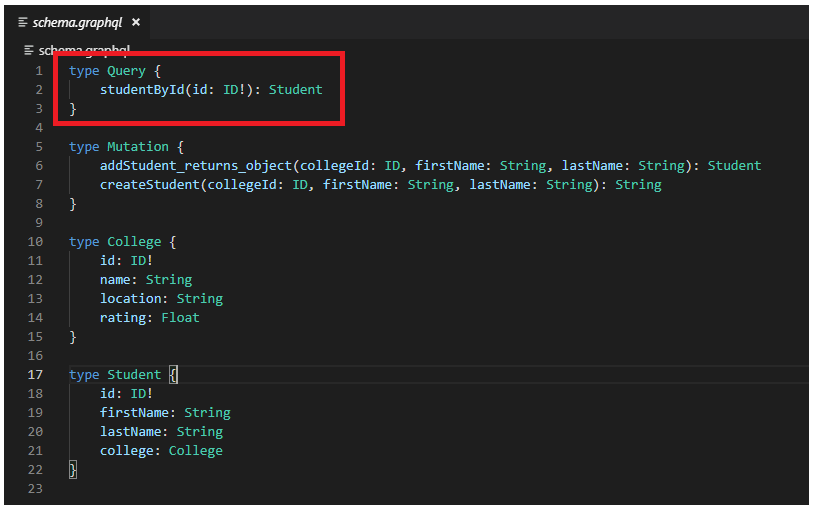
## Updation on Student.json



# verify student object

To verify if the student object is created, we can use the studentById query.

To use studentById query, edit the **schema.graphql** as given below

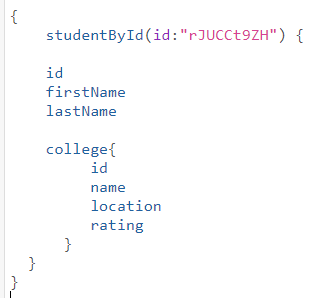


Edit the **resolver.js** file as given below



## Testing

Given below is the query to get student by unique id returned from the mutation query



The response from the server is as follows



## On Browser

