Web applications **send and retrieve data asynchronously** (in the background).

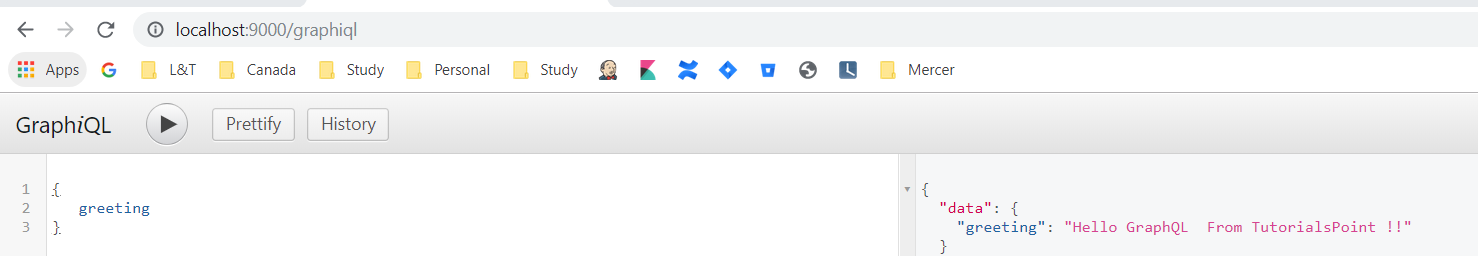
**AJAX** allows websites to load content onto the screen **without refreshing the page**.

**jQuery** provides several **methods for AJAX functionality** thus making it easier to use AJAX.

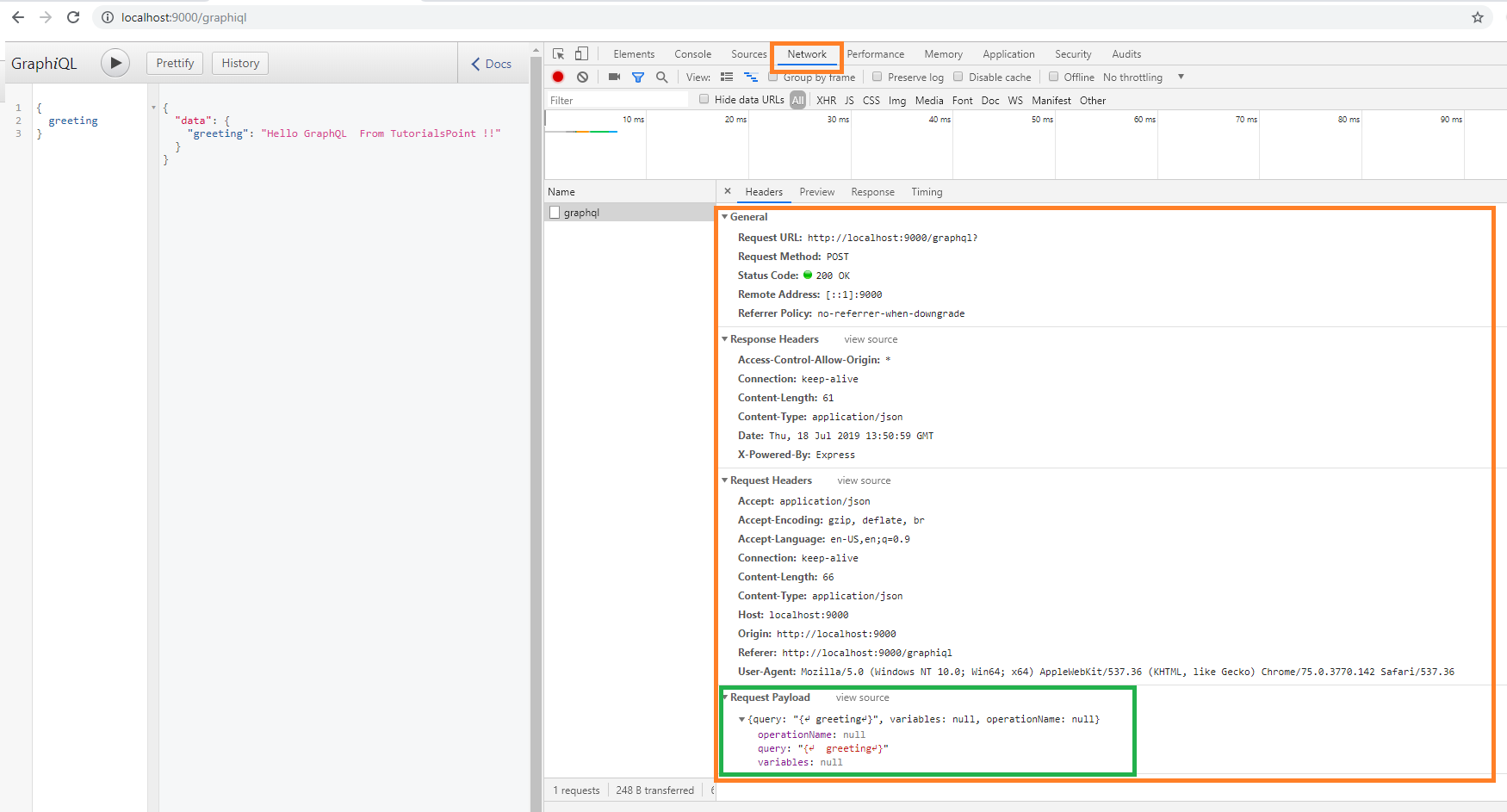
Below we will learn how we can integrate GraphQL with jQuery.

* Consider an application using client server architecture.
* We can build a front end webpage that requests data from a GraphQL server.
* The webpage will make AJAX calls using jQuery to the GraphQL server.
* To integrate GraphQL with JQuery, let us inspect the GraphiQL request headers and understand the request parameters.

Refer **2. Hello World** example app. Type the graphql query {greeting} in the GraphiQL window.



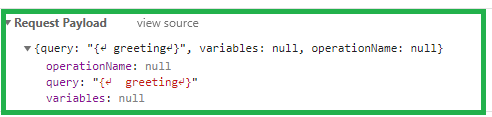
Right-click and inspect or press (ctrl + shift + I) on chrome to go to the network tab as shown below



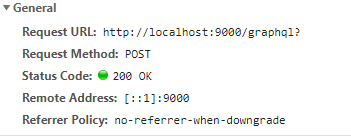
From the simple **hello-world** example, we can understand that **http method** used is **POST**.

Now in the browser, scroll down to the header section to view the *request payload*.

Once you click on **view code**, you will see the following in the request payload section of chrome.



Also note the request URL, **http://localhost:9000/graphql** that should be called from client application.

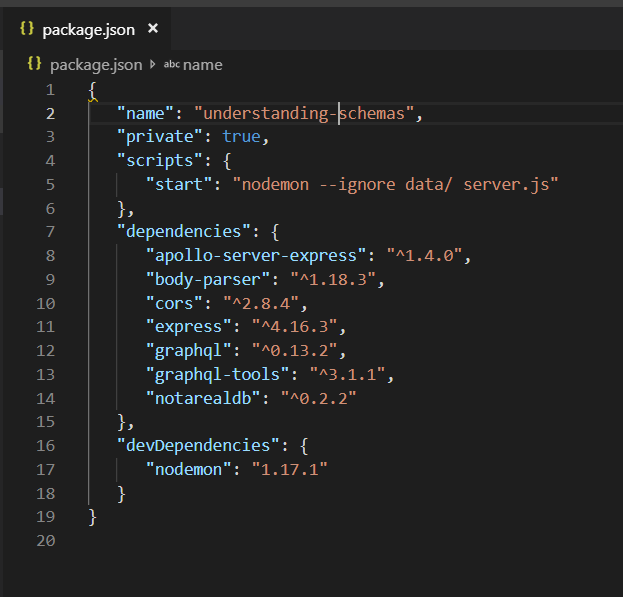


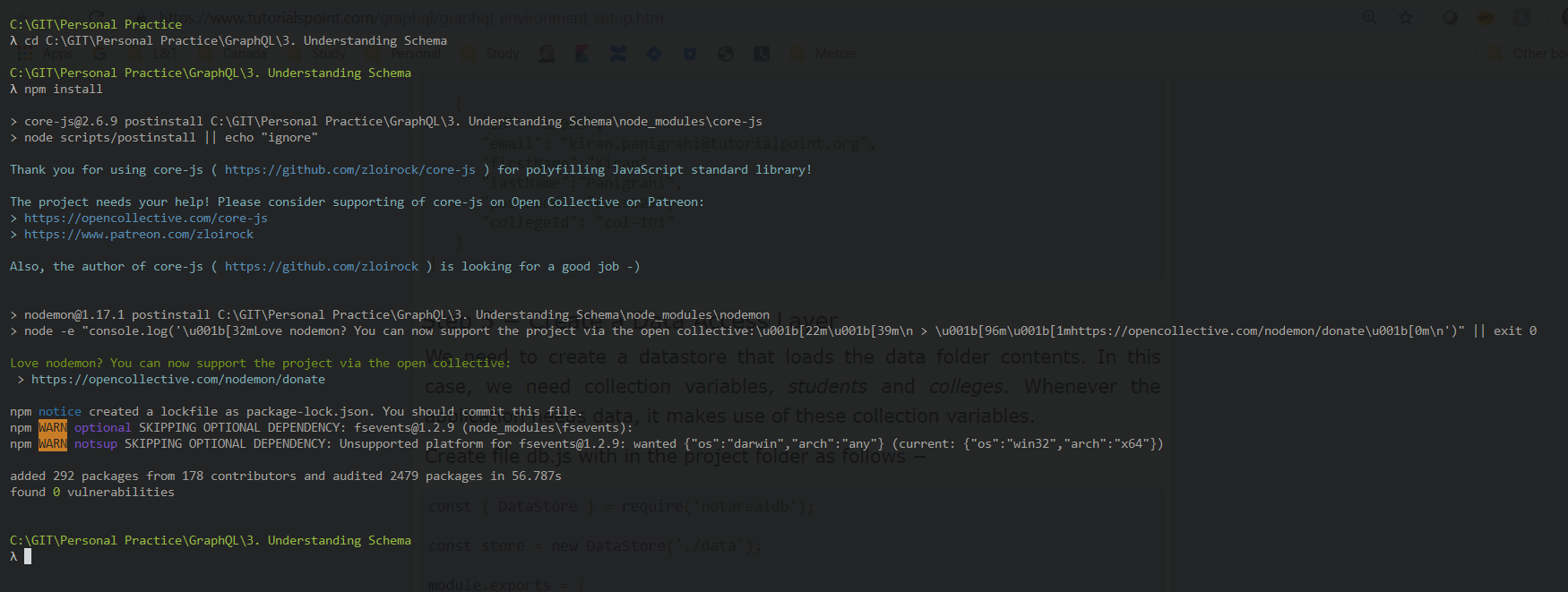
To understand how to integrate GraphQL with JQuery using a step-wise process.

# Setting up the Server

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

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

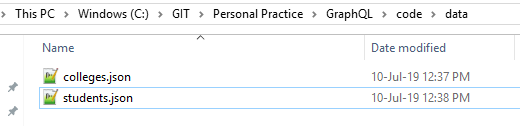


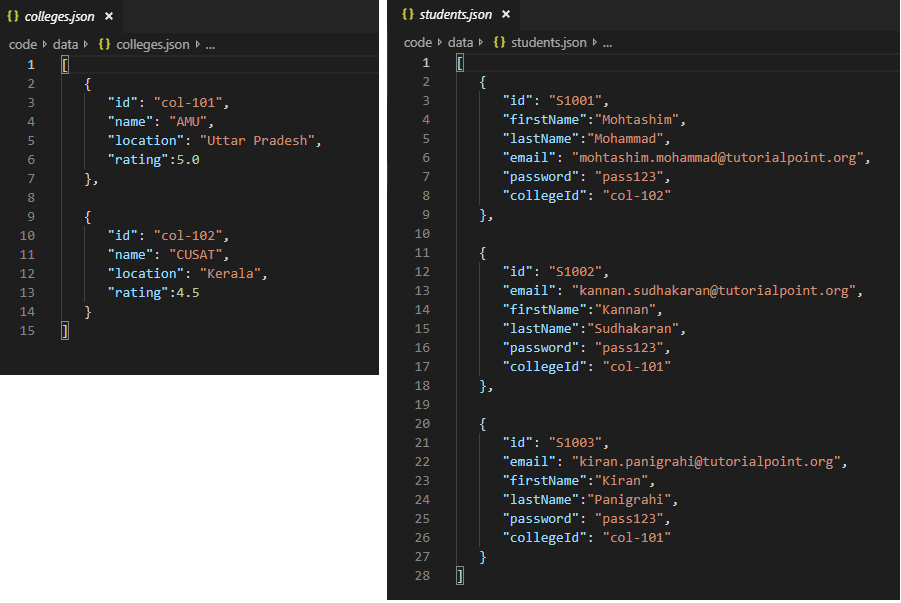


**Create Flat File Database in Data Folder**

we use flat files to store and retrieve data.

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





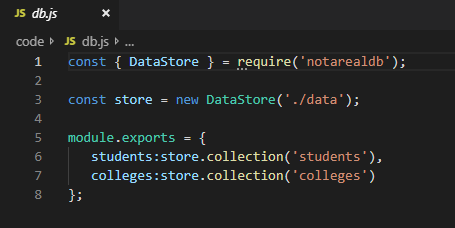
**Create a Data Access Layer**

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

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

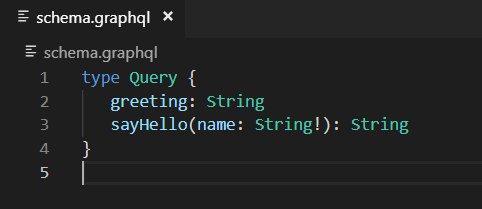
Whenever the application needs data, it makes use of these collection variables.

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



## Step 2 − Create a Schema

Add schema file at root level with below code

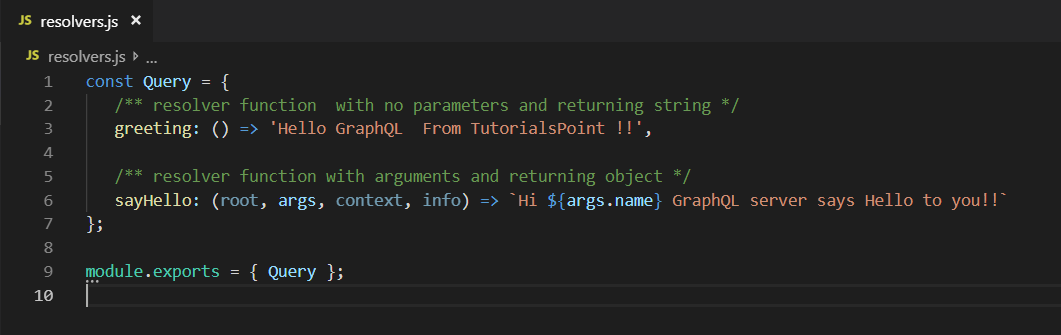


The file has defined two queries **greeting** and **sayHello**.

The sayHello query accepts a string parameter and returns another string. The parameter to the sayHello() function is not null.

## Step 3 − Create Resolvers

Create a file resolvers.js in the project folder and add the following code

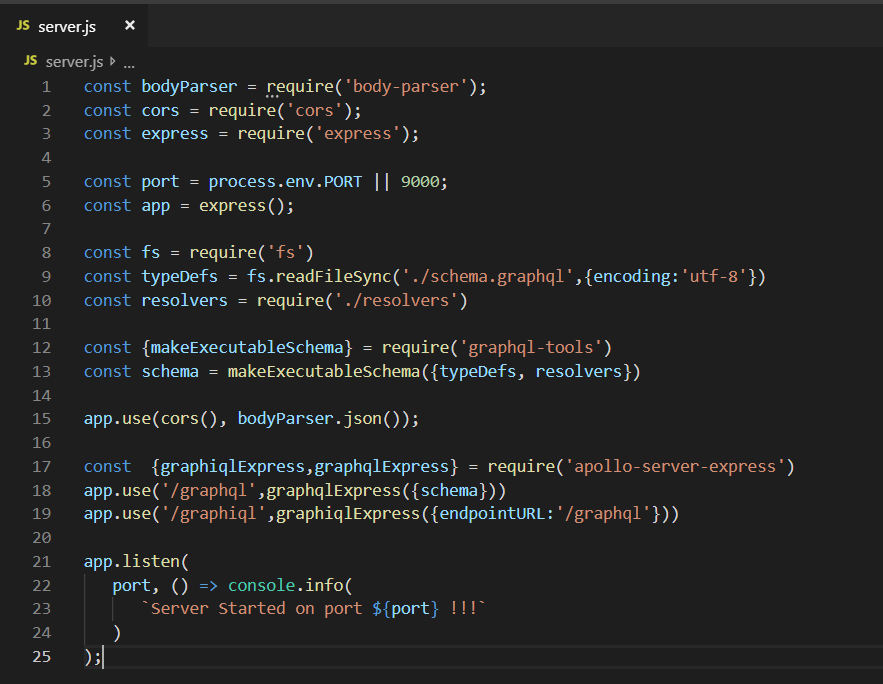


Here, **greeting** and **sayHello** are two resolvers.

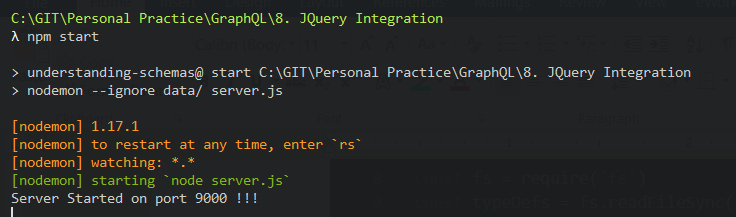
In sayHello resolver, the value passed to the name parameter can be accessed through args. To access resolver functions outside the module, Query object has to be exported using **module.exports**.

## Step 4 − Run the Application

Add below server.js



Now run server





# Setting up the Client

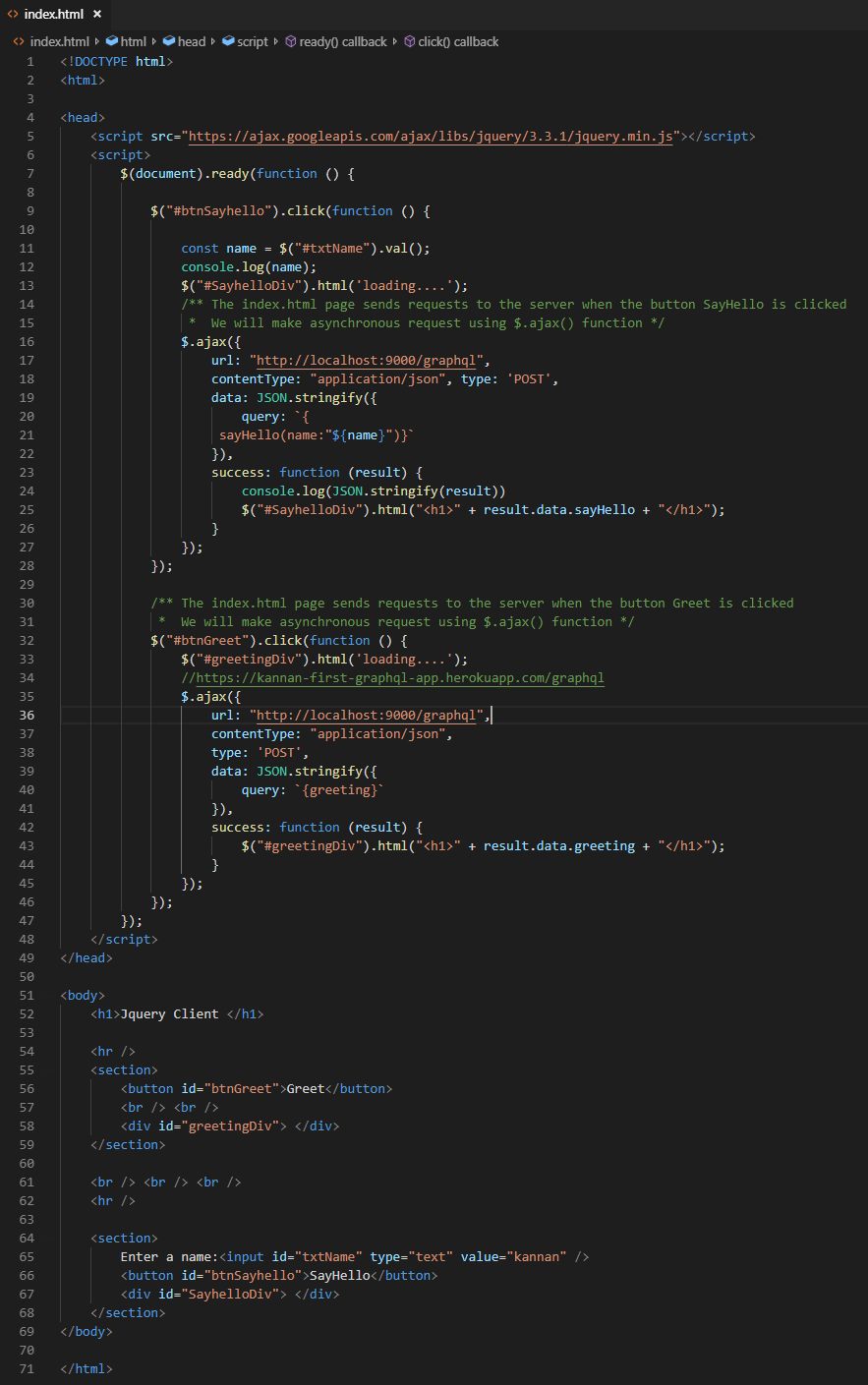
## Step 1 − Create a HTML Page index.html for jQuery Integration

We will create a client application in jquery and invoke both the methods.

Following is the code for **index.html** file.

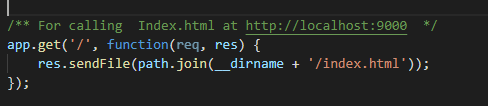
The **index.html** page sends requests to the server when the buttons – **Greet** and **SayHello** are clicked.

We will make asynchronous request using $.ajax() function.

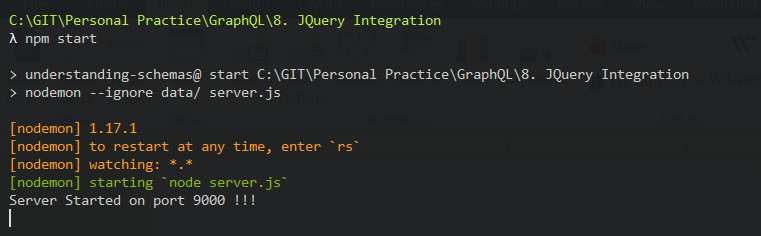


## Step 2 – Update required for Server.js

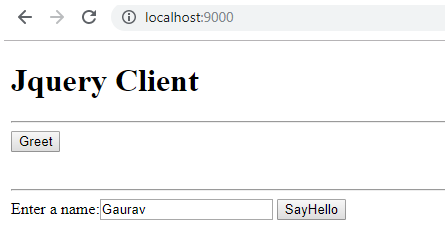
In order to invoke index.html via express, we need to add below code in server.js



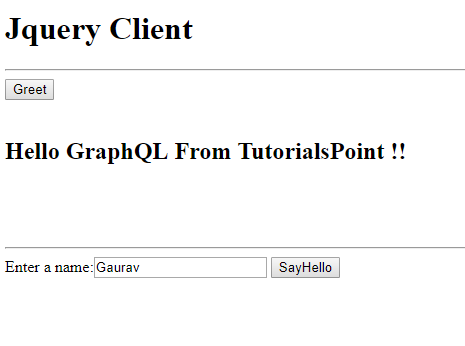
## Step 3 − Run the Application



On Page Load



On Clicking Greet button



On Clicking SayHello



<https://www.tutorialspoint.com/graphql/graphql_jquery_integration.htm>