We will create a simple API that returns a greeting message, HelloWorld, and access it using GraphiQL.

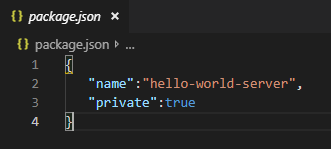
This example is based on NodeJS, Express and Apollo server.

## Step 1 − Setting up Express

ExpressJS is a web application framework that helps to build websites and web applications.

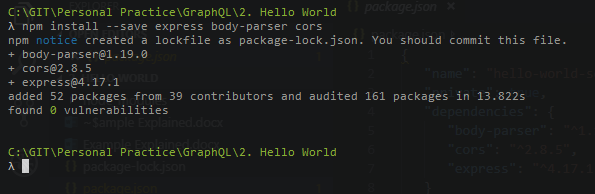
In this example, we will build a GraphQL API on top of the Express framework.

Add package.json



Since this package is only used internally, we can declare it private.

Install the dependencies for Express server



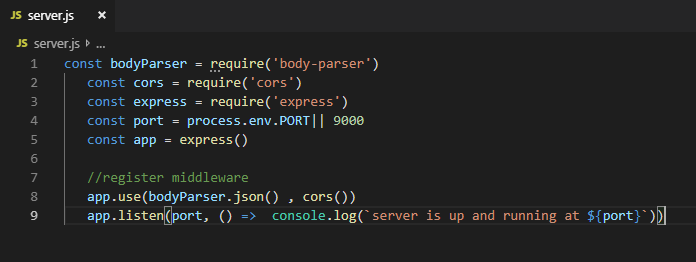
Updated package.json looks like



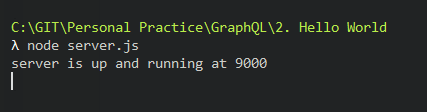
*body-parser* is a middleware package which helps Express to handle HTTP Post requests efficiently.

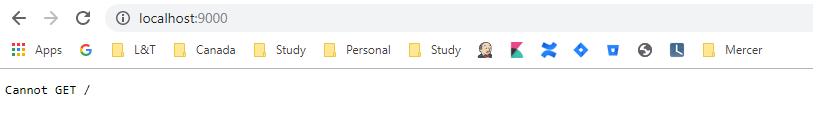
*cors* is another middleware package that handles cross-origin resource sharing.

Create a **server.js** file



To verify if the Express server is up and running, execute the following code in the terminal window



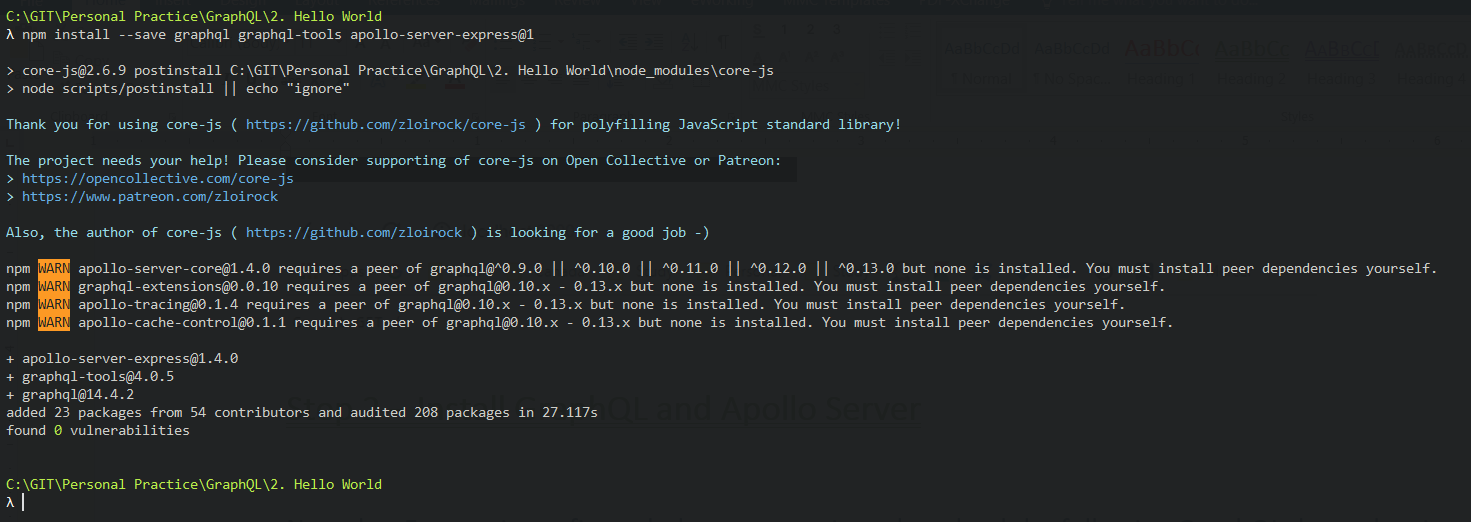


## Step 2 − Install GraphQL and Apollo Server

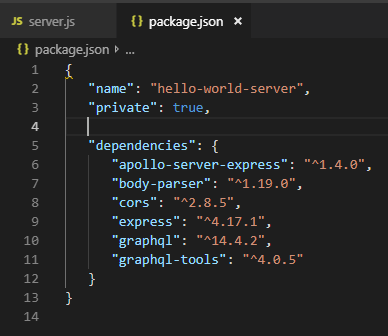
Now that Express is configured, the next step is to download the following GraphQL dependencies −

* graphql
* graphql-tools
* apollo-server-express@1

We shall use Apollo server v1.0 as it is a stable release.



Updated package.json



## Step 3 − Define the Schema

A GraphQL schema defines **what kind of object can be fetched from a service, and what fields it has**.

The schema can be defined using **GraphQL Schema Definition Language**.

Now, add the following code snippet in the **server.js** file



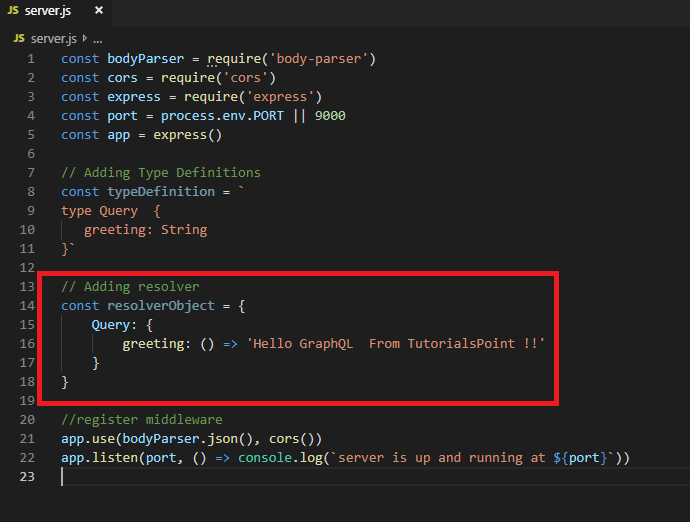
## Step 4 − Create a Resolver

The first step in creating a resolver is to add some code to process the request for greeting field.

This is specified in a **resolver**.

The structure of the resolver function must match the schema.

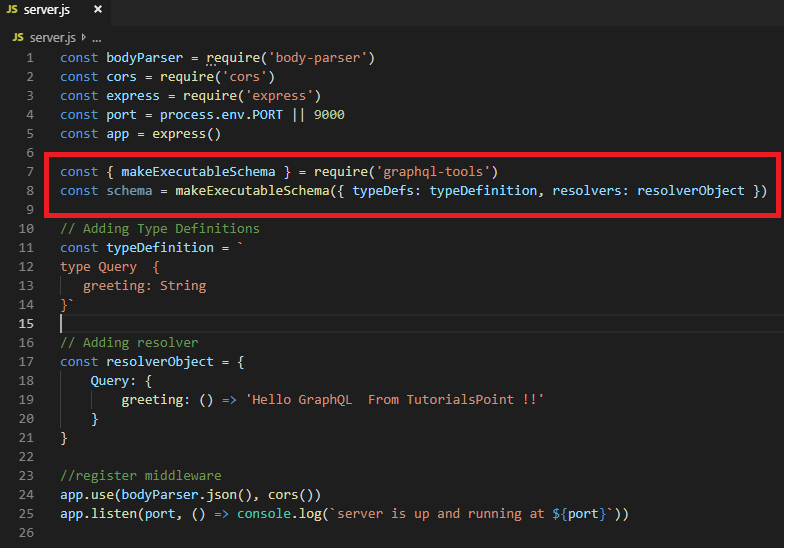
Add the following code snippet in the **server.js** file.



The second step is to bind the schema and resolver using **makeExecutableSchema**.

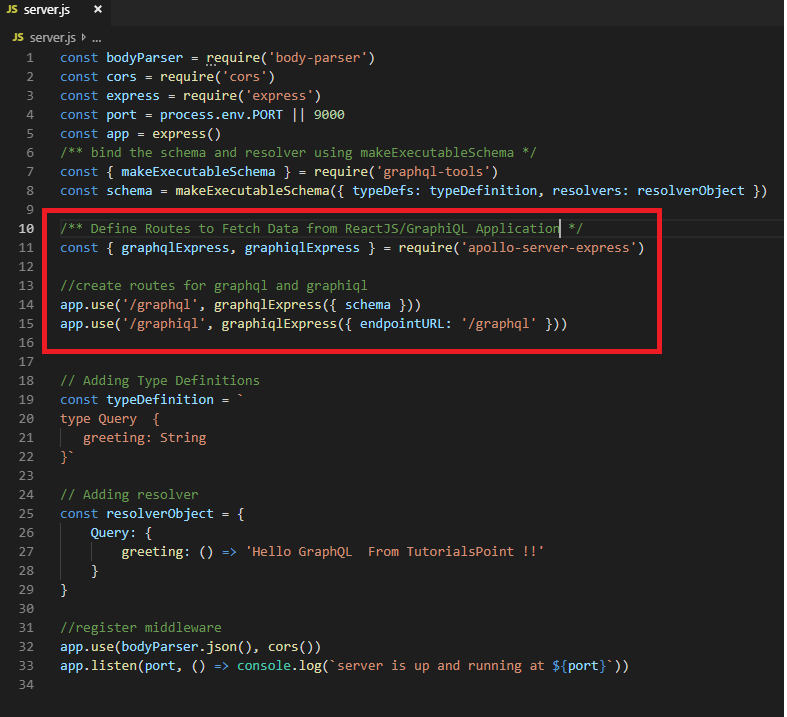
This function is pre-defined in the graphql-tools module.

Add the following code snippet in the **server.js** file.



## Step 5 − Define Routes to Fetch Data from ReactJS/GraphiQL Application

Add the following code snippet in the **server.js** file



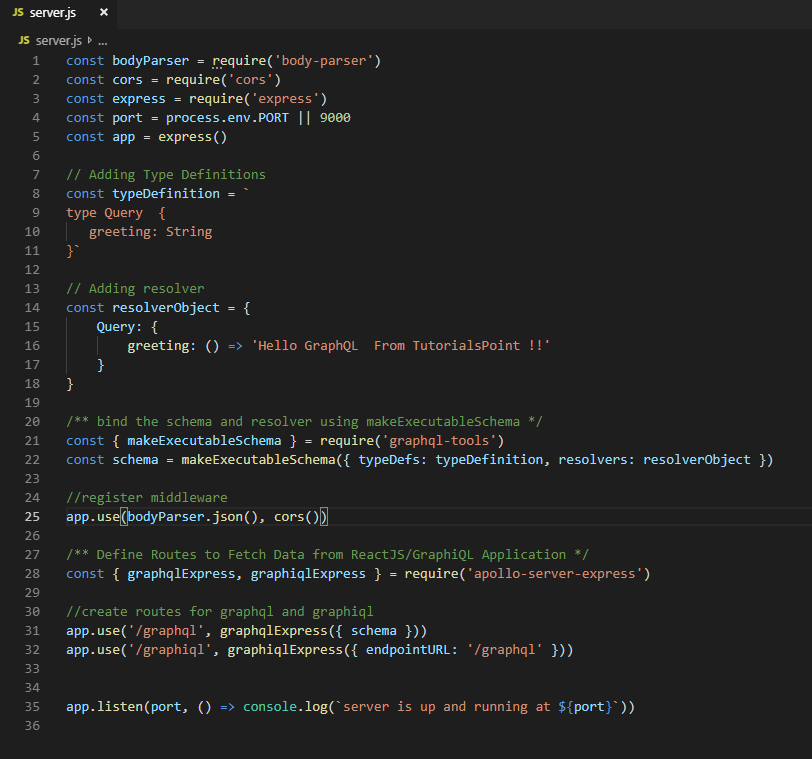
The *graphqlExpress* function helps to register the route **http://localhost:9000/graphql**.

The ReactJS application can use this endpoint to query data.

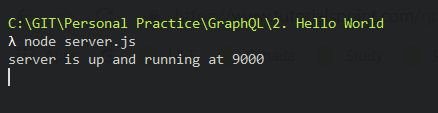
Similarly, the *graphqliExpress* function helps to register the route **http://localhost:9000/graphiql**. T

his will be used by the GraphiQL browser client to test the API.

Final **server.js**



## Step 6 − Start the Application



## Step 7 − Test the GraphQL API

