node --version

npm --version

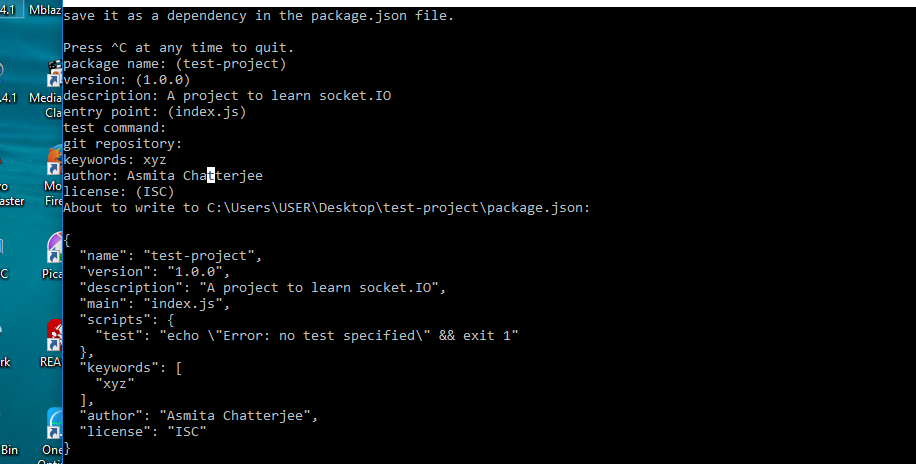
---Open your terminal and enter the following in your terminal to create a new folder and enter the following commands −

mkdir test-project

cd test-project

npm init

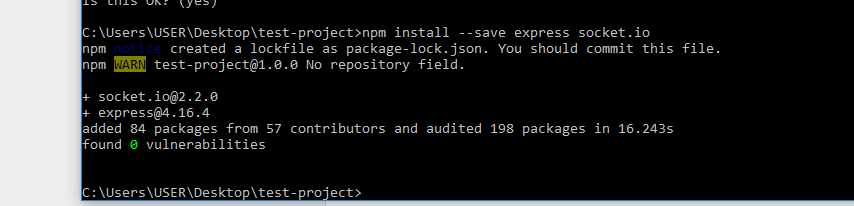
It will ask you some questions; answer them in the following way −



This will create a **‘package.json node.js’** configuration file. Now we need to install **Express** and **Socket.IO**.

To install these and save them to **package.json** file, enter the following command in your terminal, into the project directory.

npm install --save express socket.io

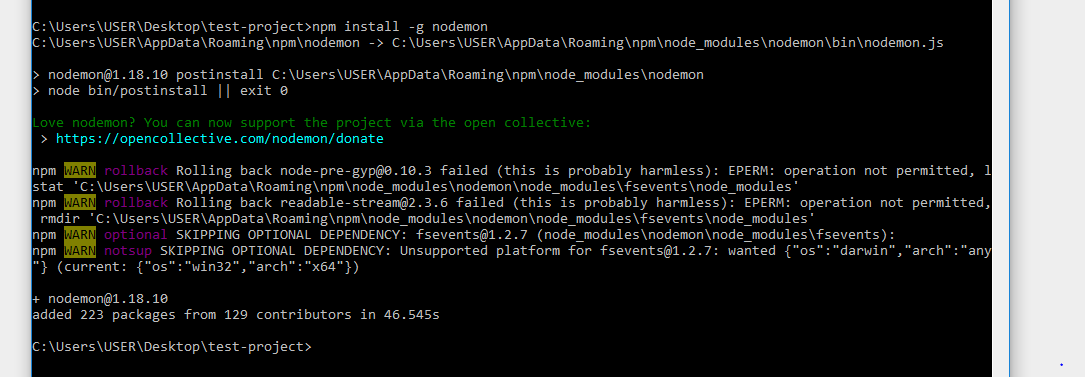


One final thing is that we should keep restarting the server. When we make changes, we will need a tool called nodemon. To install **nodemon**, open your terminal and enter the following command −

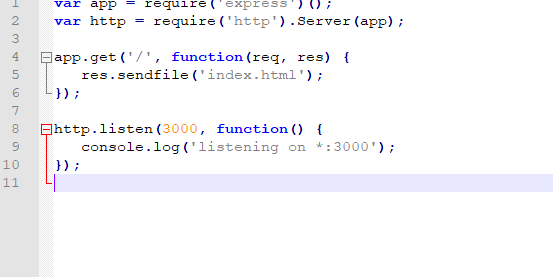
npm install -g nodemon

Whenever you need to start the server, instead of using the **node app.js** use, **nodemon app.js**. This will ensure that you do not need to restart the server whenever you change a file. It speeds up the development process.

Now, we have our development environment set up. Let us now get started with developing real-time applications with Socket.IO.

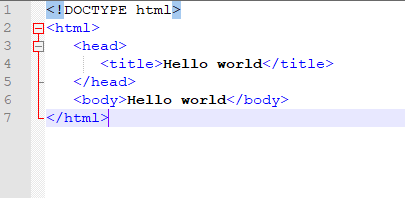


Now , Create a file called **app.js** and enter the following code to set up an express application –



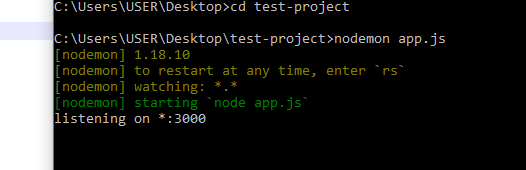


We will need an **index.html** file to serve, create a new file called index.html and enter the following code in it –



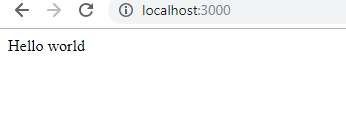


To test if this works, go to the terminal and run this app using the following command –



Now in browser , check th local host :

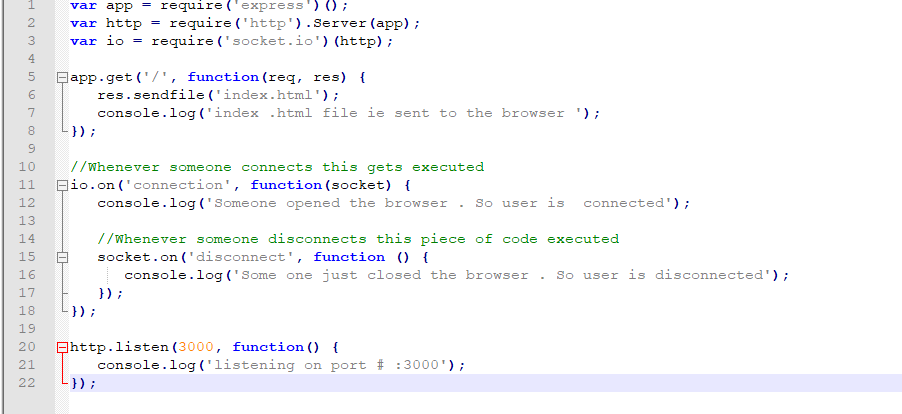
<http://localhost:3000>



This sets up our express application and is now serving a HTML file on the root route.

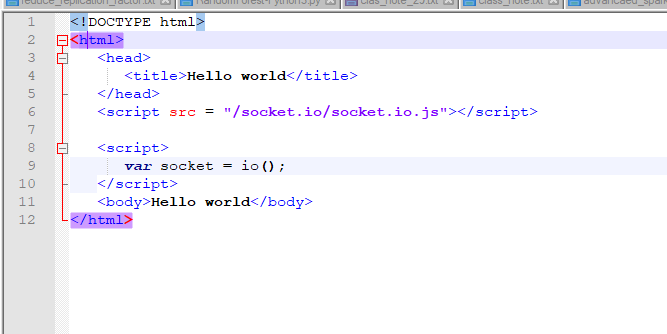
Now we will require Socket.IO and will log "A user connected", every time a user goes to this page and "A user disconnected", every time someone navigates away/closes this page.

Modify the app.js as below :

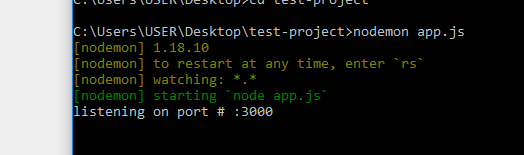


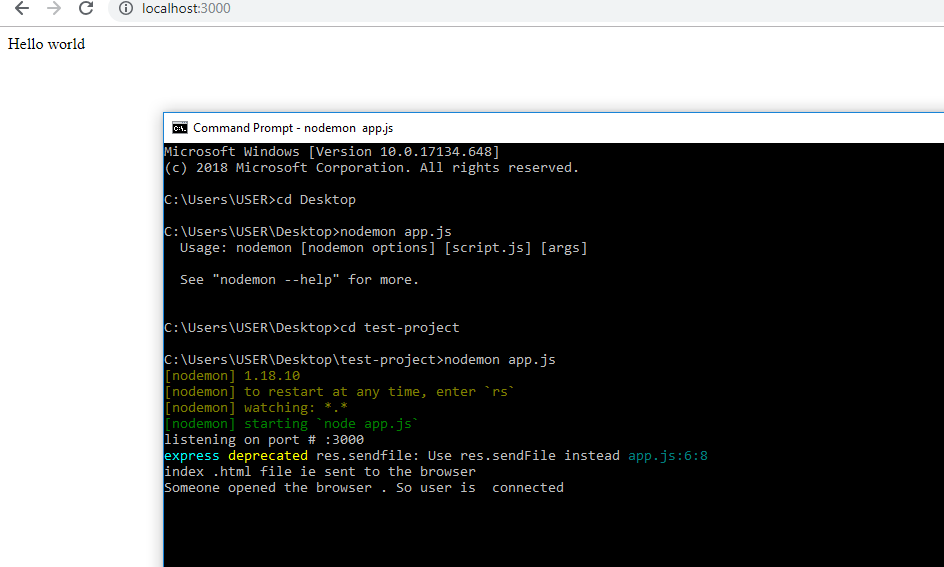
The require('socket.io')(http) creates a new socket.io instance attached to the http server. The **io.on event handler** handles connection, disconnection, etc., events in it, using the socket object.

We have set up our server to log messages on connections and disconnections. We now have to include the client script and initialize the socket object there, so that clients can establish connections when required. The script is served by our **io server** at **'/socket.io/socket.io.js'**

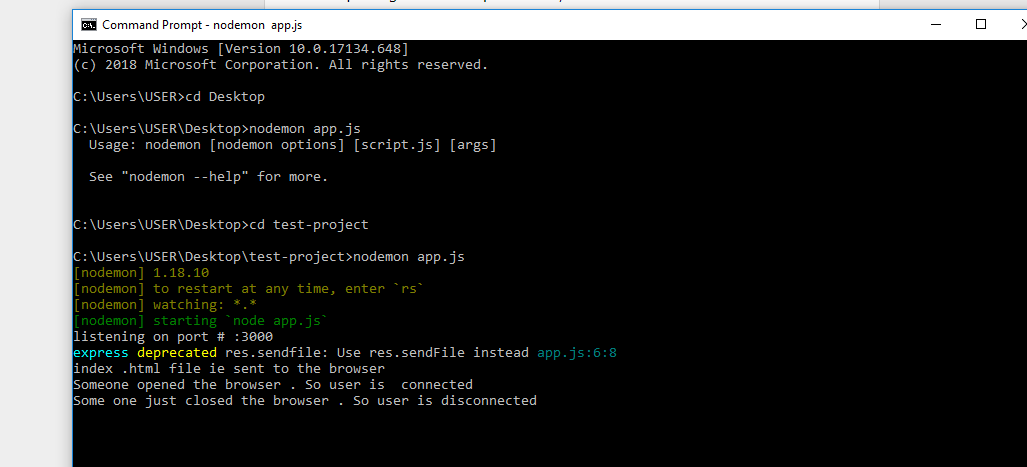


If you go to localhost:3000 now (make sure your server is running), you will get **Hello World** printed in your browser. Now check your server console logs, it will show the following message –

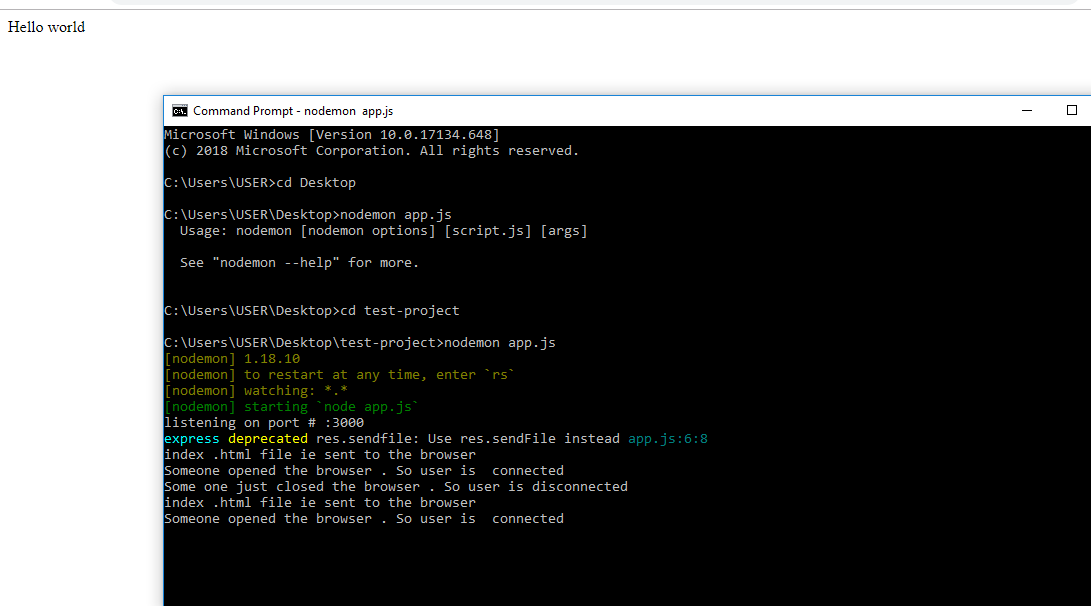




Now closing the browser



Again opening the browser



We now have socket connections working. This is how easy it is to set up connections in Socket.IO.