# Accessing the database from Node

## Welcome to this Lab activity

In this lab activity you will be exploring how to access databases from your Node web application. For the purpose of this lab activity you will be working on the same structure as the "Adding forms to your Express server" lab but under a new folder called *topic6*.

## Start the Lab environment application

It is simple to launch a lab exercise. You only need to click on the button "Start" below the activity title to enter a lab environment.

Let's explore this lab activity. Go ahead and click on the "Start" button!

Accessing the database from Node

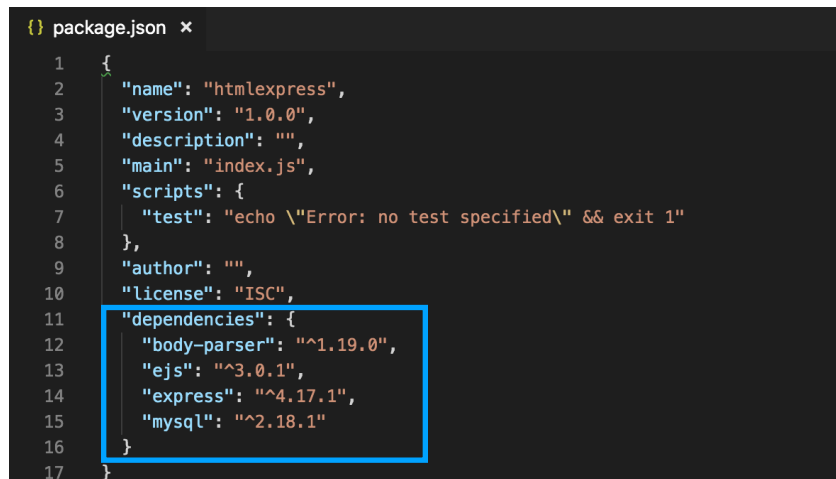⬆ **Start**
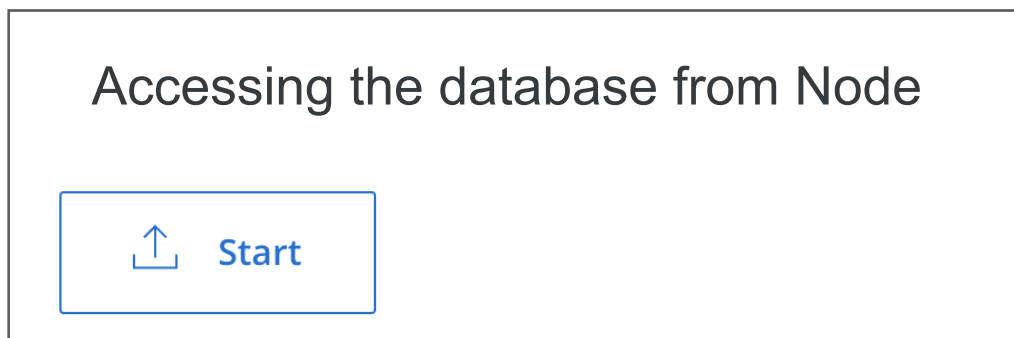
## Task 1: Query data in your Node.js file

The folder structure has already been partially constructed for you and organised into different topics. For the purpose of this lab, you will be making changes inside the *topic6* folder structure. Let's get started!

Please do not delete or move any existing folders/files inside the lab environment.

Use the Visual Studio Code Terminal and run the following commands:

- **cd topic6/htmlExpress**: type this command and press *Enter*. This command will change your working directory to be the *htmlExpress* folder.

- **npm init**: type this command and press *Enter*. This command will set up a new or existing npm package. You can skip all the npm initialisation questions by pressing *Enter*. Once the npm package file has been created, you can view it inside the *htmlExpress* directory (*package.json*). The *package.json* file contains a set of information regarding your current node project.

- **npm install express --save**: type this command and press *Enter*. This command will install the *Express* framework within your working directory so that it will be available for you to use.

- **npm install ejs --save**: type this command and press *Enter*. This command will install the *EJS* framework within your working directory so that it will be available for you to use.

- **npm install body-parser --save**: type this command and press *Enter*. This command will install the *body-parser* module within your working directory so that it will be available for you to use.

- **npm install mysql --save**: type this command and press *Enter*. This command will install the mysql module within your working directory so that it will be available for you to use. It will allow you to run mysql commands directly in your javascript files.

Once you run the above Terminal commands, your *package.json* file located inside the *topic6/htmlExpress* folder should contain **Express, EJS, Body-parser** and **Mysql** modules:



The very next step is to add some code to the *index.js* file located inside the *topic6/htmlExpress* folder in order to create your web server.

Add the following code to the *index.js* file and remember to save it:

```javascript
const express = require ("express");
const bodyParser = require ("body-parser");
const app = express();
const port = 8087;

app.use(bodyParser.urlencoded({ extended: true }));

require("./routes/main")(app);

app.set("views",__dirname + "/views");
app.set("view engine", "ejs");
app.engine("html", require("ejs").renderFile);
app.listen(port, () => console.log(`Example app listening on port ${port}!`));
```

If you have correctly followed all the above steps, your environment should be similar to the one below:
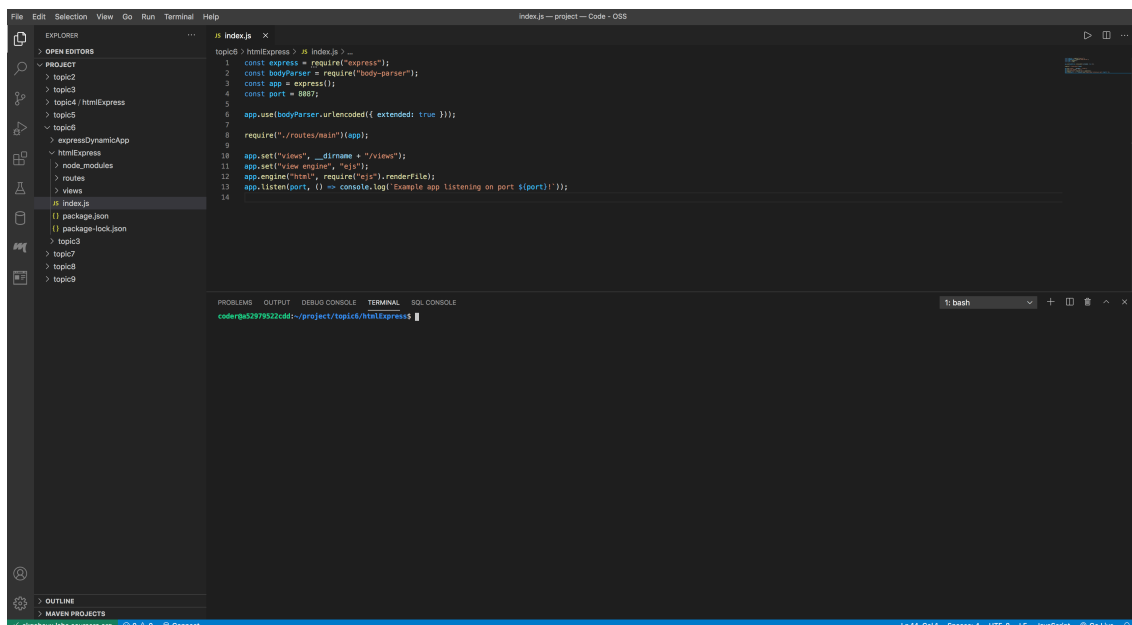


Now you have to copy your web application created inside the *topic4* folder to the *topic6* folder by following the steps below.

Create four html files inside the *topic6/htmlExpress/views* folder called *index.html about.html, search.html* and *register.html* as you did inside the *topic4/htmlExpress/views* folder. Copy the content of each html page from the *topic4/htmlExpress/views* folder to the ones inside the *topic6/htmlExpress/views* folder.

Finally copy the code from the *topic4/htmlExpress/routes/main.js* file inside the *topic6/htmlExpress/routes/main.js* file.

At this point you should have a copy of your Express application created in topic4 inside the *topic6/htmlExpress* folder.

# Running the index.js file

Run the *index.js* file with the following Terminal command:

- **node index.js**: type this command and press Enter. The node command, followed by the file name, tells Node.js to execute the content of the file.

The above command will start a web server running on port 8087.

Use the "Browser Preview" plugin to check that the application is running correctly.

If you do not remember how to use the "Browser Preview" plugin, please refer to the "Creating my first Node.js web server" lab instructions.

The 'root' route of your application should render the following:

localhost:8087/
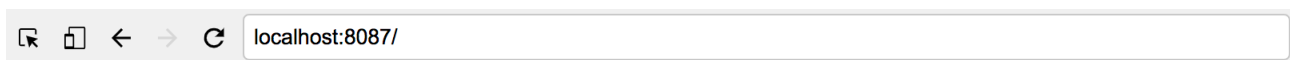
# This is home page

Homepage This is an example paragraph. Anything in the **body** tag will appear on the page, just like this **p** tag and its contents.

It is now time to integrate the "myBookshop" database with your web application.
You can open multiple Terminal panels by clicking on the + icon as shown below:
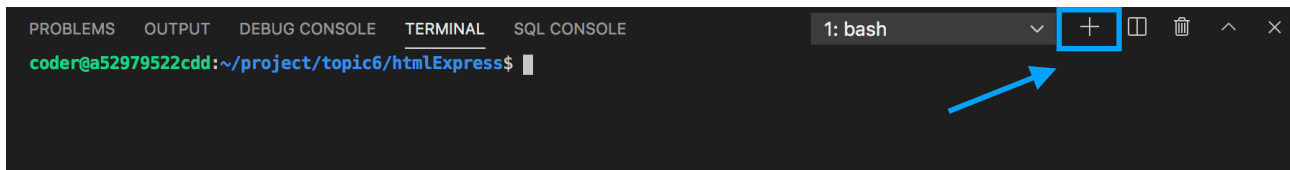


First of all, make sure that you can access the mysql shell:

- Login to your virtual server by running the **mysql** Terminal command

Next make sure that the "myBookshop" database is still working correctly:

- Make sure that "myBookshop" is among your databases (SHOW DATABASES;)
- Switch to "myBookshop" database (USE myBookshop;)
- See what fields have been defined within the "books" table (DESCRIBE books;)
- See what data is in the "books" table(SELECT * FROM books;)

If your "myBookshop" database is working correctly you should see data similar but not equal to the picture below:



Feel free to remove, update and add more books to your collection if you wish.

Now that you have successfully verified that your "myBookshop" database is working correctly, it is time to integrate it with your web application.

The very next step is to add some code to the *index.js* file located inside the *topic6/htmlExpress* folder in order to create your web server.

Add the following code to the *index.js* before the ***const port = 8087;*** line:

```
const mysql = require ("mysql");
```

Add the following code to the *index.js* after the ***const port = 8087;*** line:

```
const db = mysql.createConnection ({
  host: "localhost",
  user: "root",
  password: "",
  database: "myBookshop"
});
// connect to database
db.connect((err) => {
  if (err) {
    throw err;
  }
  console.log("Connected to database");
});
global.db = db;
```

Next you need to add the following route to the *topic6/htmlExpress/routes/main.js* file:

```
app.get("/list", function(req, res) {
 // query database to get all the books
 let sqlquery = "SELECT * FROM books";
 // execute sql query
 db.query(sqlquery, (err, result) => {
 if (err) {
   res.redirect("/");
 }
 res.send(result)
 });
});
```

Note the redirect to the home page in case of any database query errors.

# Task 2: Access your server via HTTP

Now that your code is running, serving your application on port 8087, you can access your web page from a browser!

Use the "Browser Preview" plugin to visualise your web application.

If you do not remember how to use the "Browser Preview" plugin, please refer to the "Creating my first Node.js web server" lab instructions.

Type *localhost:8087/list* on the "Browser Preview" tab and press *Enter* on your keyboard to visualise your web application:



# End of Section

Congratulations for completing this section. As long as you have saved your work, your files will remain when you close this lab activity so do not worry about losing your data. You have successfully created a web application that connects to a database.

In the next lab activity you will use the EJS templating engine to make your application dynamic.