Inserting data from Node to the database

Welcome to this lab activity

In this lab activity, you will explore how to pass different variables to your application backend and database.

For example, how about adding a new book to your database? The data can be entered into a form, passed to the backend and inserted into the database.

Task 1: Make a copy of the previous lab code

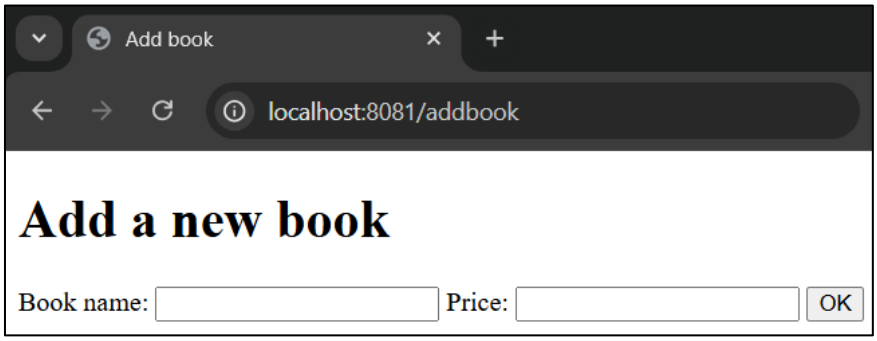
You will start with your previous Node.js lab code.

1. Make a copy of the folder `16_retrieving_data_from_database` from your previous lab and call it `17_inserting_data_to_database`.
2. Open the new folder in Visual Studio Code.
3. Run the code and confirm that the list page still works (refer to the previous lab if you need a reminder of how to do this).

Task 2: Create an Add book form

You will now create a new web page, showing a form where users can enter the details of a new book.

4. Create a new Add book page that displays a form like this:



The form should use the POST method with an action '`/bookadded`'.

Hint:

- Ask yourself which page of your web application has similar functionalities to the Add book page. Can you re-use parts of the code for this new page?
- Update your `main.js` file adding the new route `/addbook`.
- Create a new EJS file called `addbook.ejs` containing the input form.

Task 3: Handle the form POST method

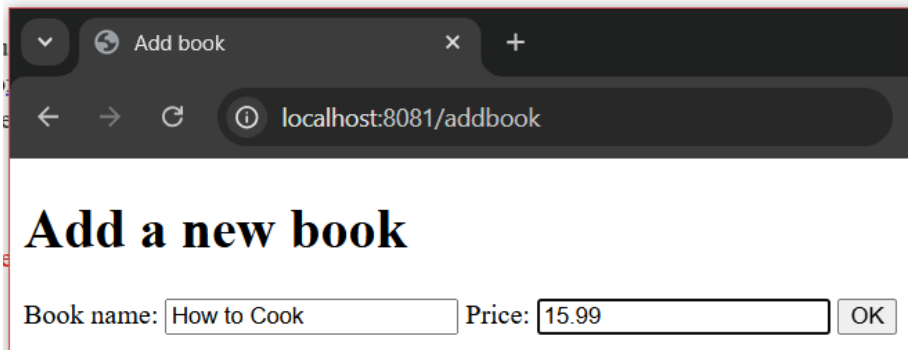
Your form should send a POST request to the `/bookadded` route when submitted. So, you will need to add a handler for this post method. This handler should extract the contents of the form and insert them into the database.

5. Add the following POST route handler to `main.js`:

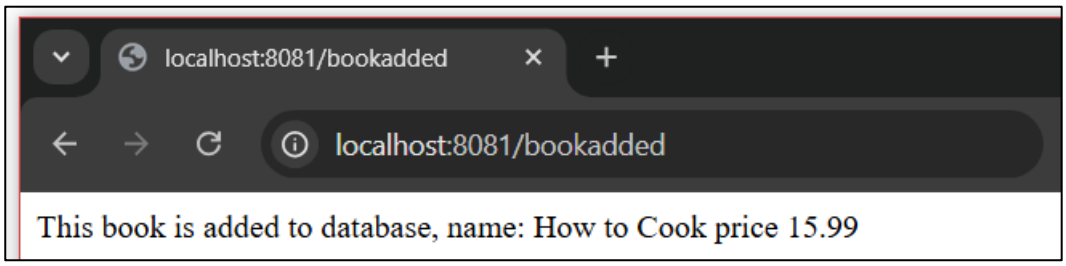
```
router.post("/bookadded", function (req, res, next) {
    // Create query
    let sqlquery = "INSERT INTO books (name, price) VALUES (?,?)";

    // Execute sql query
    let newrecord = [req.body.name, req.body.price];
    db.query(sqlquery, newrecord, (err, result) => {
        if (err) {
            next(err);
        } else {
            res.send(" This book is added to database, name: " +
req.body.name + " price " +       req.body.price);
        }
    });
});
```

Task 4: Test your app

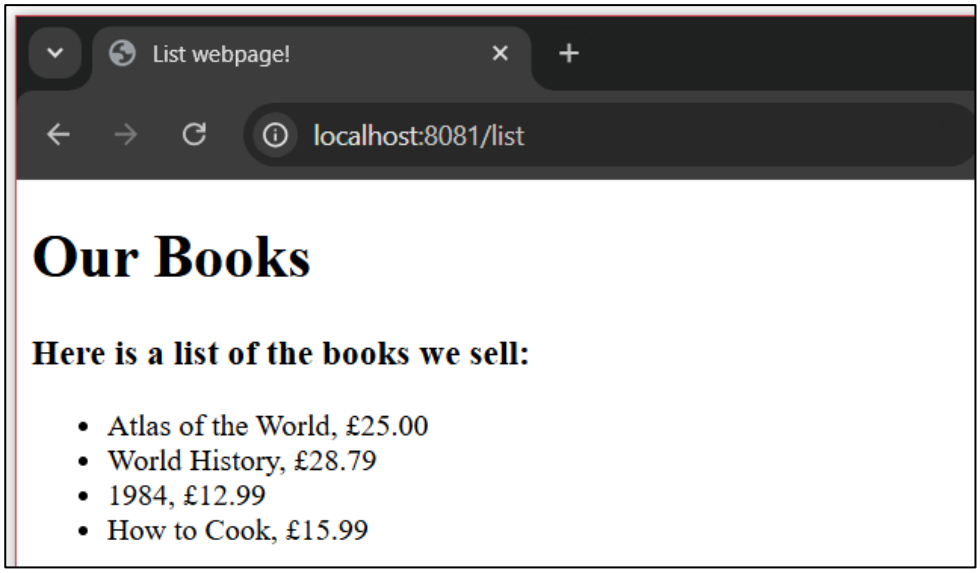
6. Run the app and browse to the `/addbook` route. Enter a new book, for example:



7. Press ‘OK’ on the form and check that the confirmation message is shown:



8. Check that the book has been added to the database by browsing to the `/list` route:



Task 5: Explore further

When tackling these lab activities, it’s always good to stretch yourself by doing some research and attempting some changes on your own.

Once the book is added, it would be nice to display the list of books. Can you add a link to the `/bookadded` page with a link to the `/list` page?

End of lab

Congratulations on completing this lab.

In the next lab activity, you will learn how to use the ‘search’ query in your database to search a particular item.