

<b>Course:</b>	<b>INFO6144</b>
<b>Professor:</b>	<b>Osam Ali / Jim Cooper</b>
<b>Project:</b>	<b>Project #2 – Customer Maintenance Page – Node.js</b>
<b>Due Date:</b>	<b>Friday, August 16, 2024</b>
<b>Submitting:</b>	<b>Please see the last page for instructions.</b>

## How will my project be marked?

---

- This project counts for 15% of your final grade and will be evaluated using the following grid:

<b>Marks Available</b>	<b>What are the Marks Awarded For?</b>	<b>Mark Assigned</b>
2	Good coding style including proper indentation and use of variable and object naming conventions and suitable comments	
3	Display index.html page correctly and with good styling	
3	Index.html handles all client-side requirements correctly	
5	Node.js server code handles all server-side requirements correctly	
2	Proper submission	
15	Total	

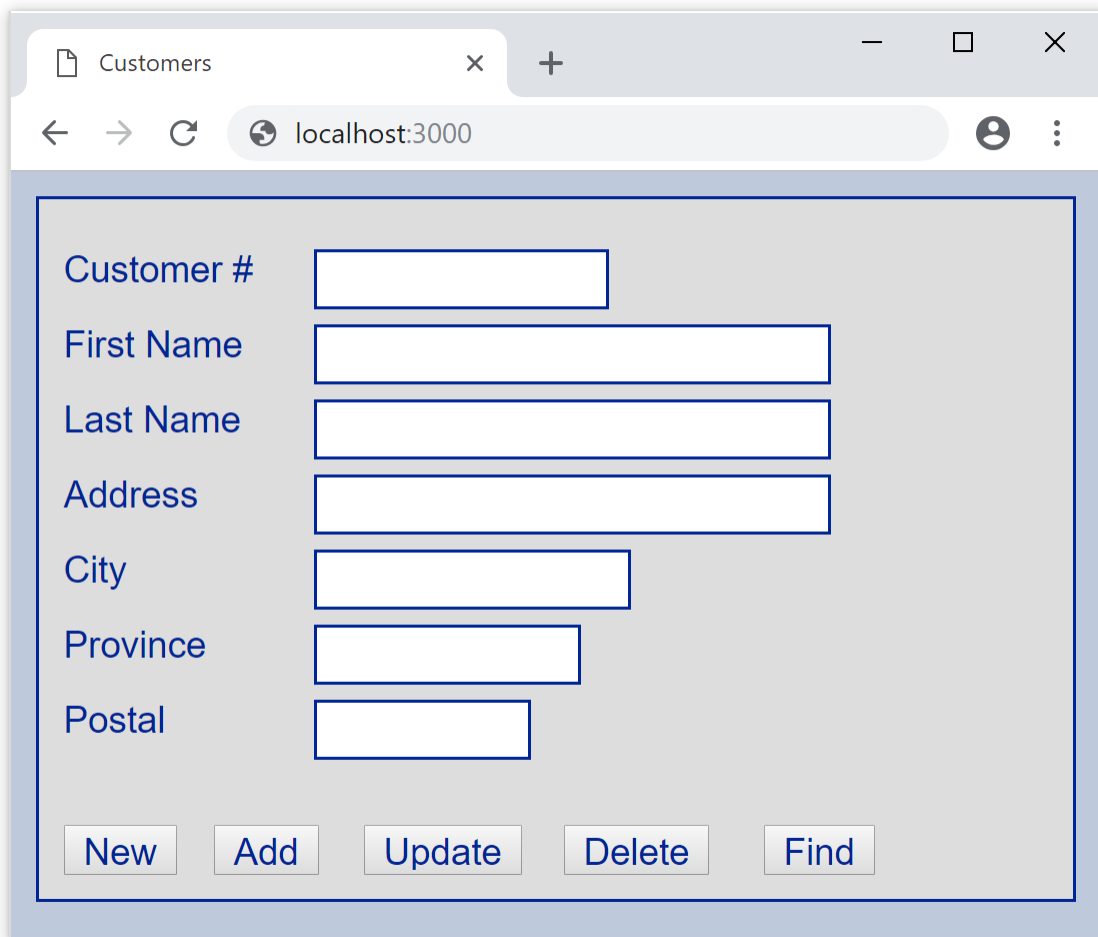
## Project Description

---

Use Visual Studio 2022 to create a Node.js server application called Customers with a server page named server.js and a client page named index.html. The purpose of this kind of application would be to:

1. Allow users to maintain customer data using a web page where maintenance means the ability to add new customers, update existing customer data, delete customers, and find a specific customer by customer number. In the database world, this is called CRUD (Create, Read, Update and Delete).
2. You'll need to use AJAX to issue post requests (with responses) between the html page and the Node.js server.
3. For help with this, see the NodeAJAX example.
4. While it's not a requirement for this project, it is recommended that you use jQuery to manage the client-side post requests and response call backs. The NodeAJAX solution uses jQuery AJAX and provides a good example of utilizing round-trip request/response strategies with Node.js.

5. The UI for your index.html page should be designed to have the following functions although you can use different styling.



The screenshot shows a web browser window with a single tab titled 'Customers'. The address bar displays 'localhost:3000'. The main content area contains a form with the following fields and labels: 'Customer #' (a short text box), 'First Name' (a medium text box), 'Last Name' (a medium text box), 'Address' (a long text box), 'City' (a medium text box), 'Province' (a medium text box), and 'Postal' (a short text box). Below these fields is a row of five buttons: 'New', 'Add', 'Update', 'Delete', and 'Find'. The buttons are light blue with dark blue text.

#### UI Requirements:

1. The **New** button must clear all the text boxes.
2. The **Add** button verifies that all fields have some content and if so, will issue a POST request to the server to attempt to add a new customer record. The POST request will consist of a JSON string containing all the field data. If subsequently, the customer # is found to be not unique, the server will not add the new record and will return a suitable error message. If the customer # is unique, a new file will be created using the customer # + ".txt" as the file name and the JSON string received from the browser as the content – for example "123.txt".
3. The **Update** button verifies that all fields have some content and if so, will issue a POST request to the server to attempt to update the file for the

current customer #. If the file for the customer # does not exist, return an error message otherwise overwrite the file with the new JSON data.

4. The **Delete** button will first use the JavaScript “confirm” function to have the user verify the decision to delete the customer. If the deletion is confirmed, take the current customer # and issue a POST request to the server to delete the relevant file. When the response comes back to the browser, clear all fields.
5. The **Find** button will take the current customer # and issue a POST request to the server to read the JSON string from the file for the current customer and return that string as the response to the browser. The client-side JavaScript will then take the returned JSON, convert it to a JavaScript object and populate all the text boxes.
6. All files discussed above must be stored in a folder on the server called “data”.

#### **Notes:**

Since all of our Node.js examples have been developed using Visual Studio 2022, it's recommended that you use this development framework to complete the project.

Please note that using Visual Studio is not a project requirement, i.e., you can use a different development platform if you like, however you should package up your solution such that I won't need to install a new development environment to test your submission.

**Also keep in mind that all grading of projects will be done on a Windows 10/11 platform and as such, your submission must be testable as a Node.js application on a Windows machine without the need to install any new software or make any changes to your submission.**

## **How should I submit my project?**

---

### **Electronic Submission:**

Submit your program files to the *Info6144“Project 2”* electronic submission folder in *FanshaweOnline*. These files should be submitted as a single “zip” file containing your web application's complete website.

I strongly recommend that you test your own submission to ensure that nothing has been missed.

### **Submit your project on time!**

Project submissions must be made on time! Late projects will be subject to divisional policy on missed test and late projects. In accordance with this policy, no late projects will be accepted without prior notification being received by the instructor from the student.

### **Submit your own work!**

It is considered cheating to submit work done by another student or from another source. Helping another student cheat by sharing your work with them is also not tolerated. Students are encouraged to share ideas and to work together on practice exercises, but any code or documentation prepared for a project must be done by the individual student. Penalties for cheating or helping another student cheat may include being assigned zero on the project with even more severe penalties if you are caught cheating more than once. Just submit your own work and benefit from having made the effort on your own.