**Module 2: Lab 2 Basic Full Stack Labs**

Sushmita Prafull Halasawade

IFT 544: Middleware Prog & Database Sec

Dinesh Sthapit

10 Sep 2023

**Exercise 1: Variables:**

**Output:**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Exercise 2: Arrays and Objects**

**Output:**

**A screenshot of a computer

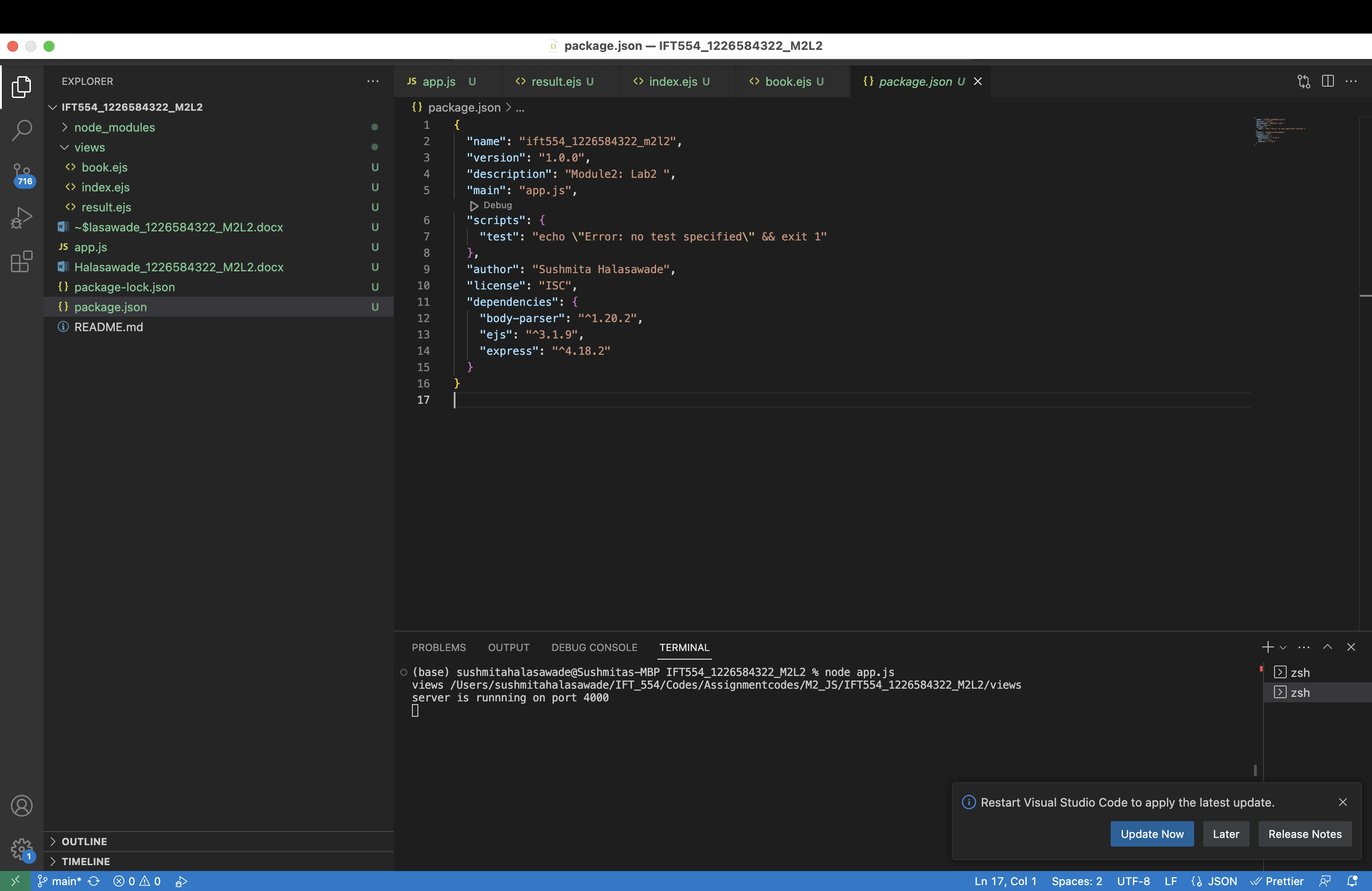
Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Step1: Setup:**

Package.json and installation of express,ejs and body



**Step2: .ejs Code Analysis:**

1. Review the .ejs file line by line and identify sections that contain JavaScript code within <% %> tags:

In the book.ejs file, JavaScript code is embedded within <% %> tags. These tags are used to execute JavaScript code within the HTML template. Here are the parts of .ejs file that contain JavaScript code:

book.ejs:

<ul>

<%books.forEach(book=>{%>

<li><%= book.title %> by <%= book.author%> (Published: <%= book.publicationYear%>)</li>

<% }); %>

</ul>

Here, <% books.forEach(book => { %> indicates the beginning of a JavaScript loop. It iterates over the books array and generates list items for each book's title, author, and publication year.

2. Analyze and understand the JavaScript code embedded in the .ejs file, including control structures, variable usage, and EJS-specific syntax:

* Control Structures:The forEach loop is used to iterate through each book in the books array.
* Variable Usage:book.title, book.author, and book.publicationYear are variables representing properties of each book object.
* EJS-specific Syntax:<%= ... %> is used to output the value of a JavaScript variable or expression to the HTML.

3. Take note of any notable features or functionalities implemented in the .ejs file.:

dynamic generation of the list items: HTML is generated based on the data in the books array.

EJS Templating: The file demonstrates the use of EJS as a templating engine for generating dynamic HTML content. EJS tags (<%= ... %>) are employed to insert JavaScript variables and expressions into the HTML structure.

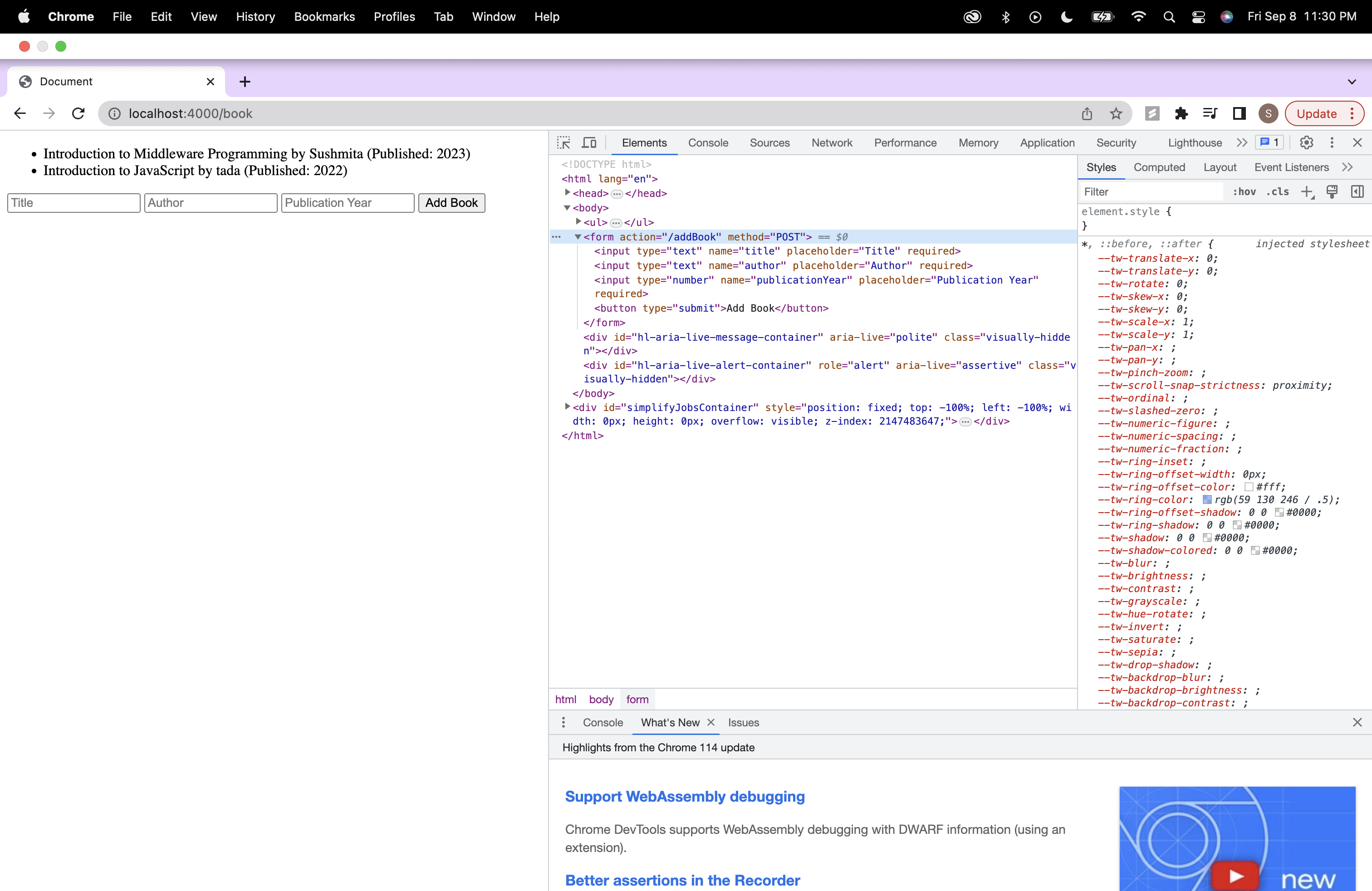
**Step3: HTML Code Analysis:**

1.Open the HTML file in a browser or a code editor:

A screenshot of a computer

Description automatically generated

2. Inspect the rendered HTML code in the browser or view the HTML file's source code:



3. Identify the parts of the HTML code that correspond to the JavaScript code in the .ejs file.:

JavaScript code is embedded within <% %> tags. JavaScript code iterates through the books array and generates list items (<li>) dynamically based on the books array. Here are the parts of the HTML code that correspond to JavaScript code:

<ul>

<%books.forEach(book=>{%>

<li><%= book.title %> by <%= book.author%> (Published: <%= book.publicationYear%>)</li>

<% }); %>

</ul>

4. Compare the rendered HTML code to the original .ejs code and note any differences or modifications.:

<body>

<ul>

<li>Introduction to Middleware Programming by Sushmita (Published: 2023)</li>

<li>Introduction to JavaScript by tada (Published: 2022)</li>

</ul>

<form action="/addBook" method="POST">

<input type="text" name="title" placeholder="Title" required="">

<input type="text" name="author" placeholder="Author" required="">

<input type="number" name="publicationYear" placeholder="Publication Year" required="">

<button type="submit">Add Book</button>

</form>

</body>

In the original book.ejs code, JavaScript expressions (<%= %>) are used to dynamically insert values (e.g., book.title, book.author, book.publicationYear) into the HTML elements. In the rendered HTML, these expressions are replaced with actual values.

The <% %> tags in the original book.ejs code are not present in the rendered HTML. They are used for embedding JavaScript logic and are not visible in the final output.

The form action attribute action="/addBook" in the HTML code corresponds to the route defined in the app.js file (app.post("/addBook", ...)) for handling form submissions.

**Step4: documentation:**

1.Analysis Summary:

In this code we are creating a simple web application using Node.js and Express framework. The application allows users to add books with their title, author, and publication year. It uses EJS as the templating engine for rendering dynamic content.

2.Comparison: .ejs vs. HTML:

book.ejs (.ejs Code):

<body>

<ul>

<%books.forEach(book=>{%>

<li><%= book.title %> by <%= book.author%> (Published: <%= book.publicationYear%>)</li>

<% }); %>

</ul>

<form action="/addBook" method = "POST">

<input type = "text" name = "title" placeholder="Title" required>

<input type = "text" name = "author" placeholder="Author" required>

<input type = "number" name = "publicationYear" placeholder="Publication Year" required>

<button type="submit">Add Book</button>

</form>

HTML Code Generated in the Browser:

<body>

<ul>

<li>Introduction to Middleware Programming by Sushmita (Published: 2023)</li>

<li>Introduction to JavaScript by tada (Published: 2022)</li>

</ul>

<form action="/addBook" method="POST">

<input type="text" name="title" placeholder="Title" required="">

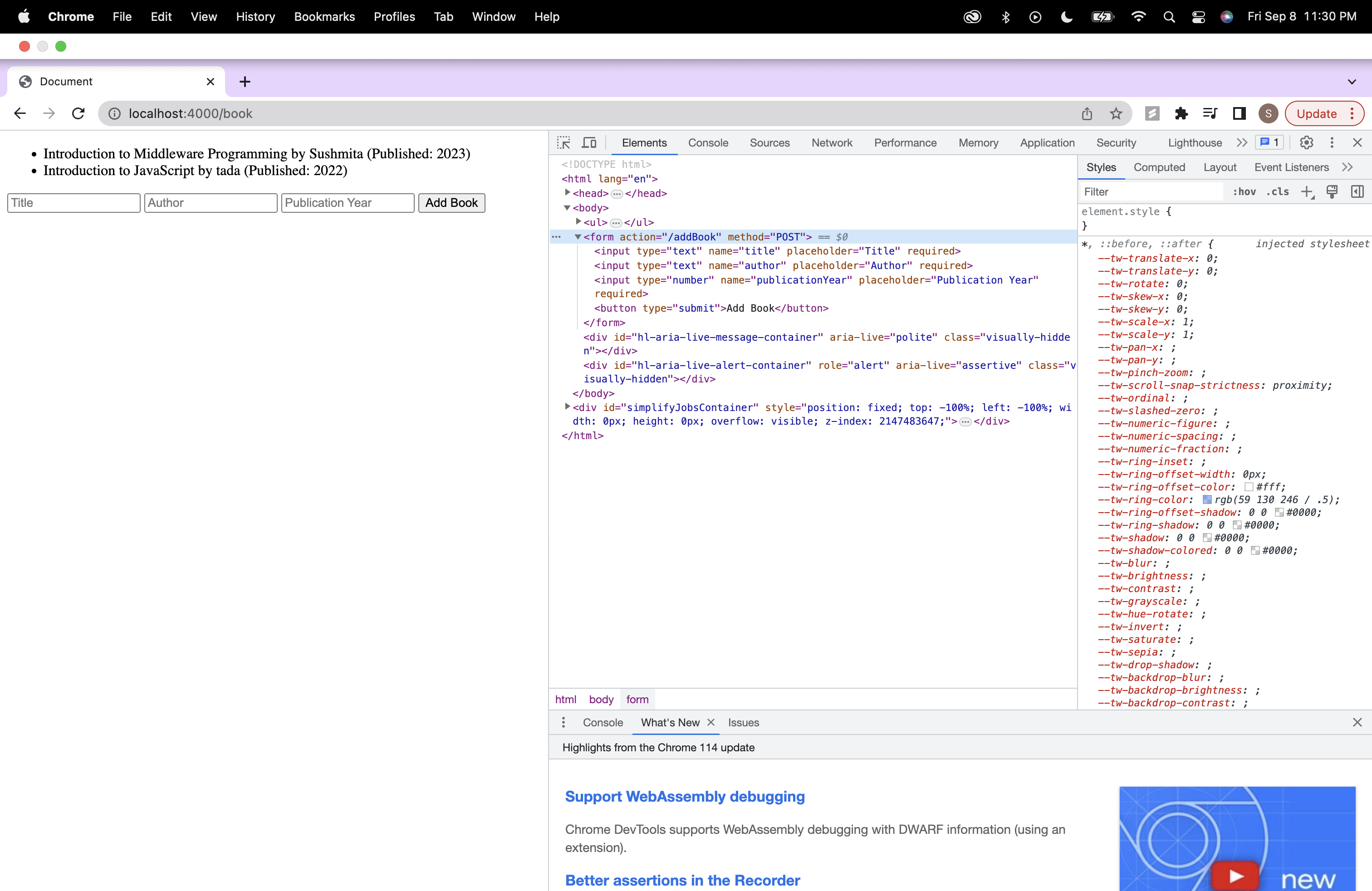
<input type="text" name="author" placeholder="Author" required="">

<input type="number" name="publicationYear" placeholder="Publication Year" required="">

<button type="submit">Add Book</button>

</form>

</body>



3.Observations and Differences:

EJS templates allow you to inject dynamic data (e.g., book titles, authors) directly into your HTML structure, making it easier to create dynamic web pages.

The <%books.forEach(book=>{%> loop in the EJS template enables you to iterate through the books array and generate list items for each book.

4.Benefits of Using .ejs Templates:

Dynamic Content: .ejs templates allow us to inject dynamic data into your HTML, making it easier to display results or user-specific information.

Reusability: .ejs templates can be reused across different pages, reducing code duplication.

Maintainability: It separates code logic (in JavaScript) from the presentation (in .ejs files), making it easier to maintain and update.

**Step5: Optional Enhancement:**

Modified result.ejs:

<ul>

<%books.forEach(book=>{%>

<li><%= book.title %> by <%= book.author%> (Published: <%= book.publicationYear%>

<% if (book.publicationYear >= 2020) { %>

<span style="color: green;">(New)</span>

<% } else { %>

<span style="color: red;">(Old)</span>

<% } %>)</li>

<% }); %>

</ul>

I've added a conditional statement (if-else) that checks if the publicationYear of a book is greater than or equal to 2020. If it is, it adds a green "(New)" label next to the book title; otherwise, it adds a red "(Old)" label.

A screenshot of a computer

Description automatically generated

2. Analyze the corresponding changes in the rendered HTML code and document your findings.

Rendered HTML:

<ul>

<li>Introduction to Middleware Programming by Sushmita (Published: 2023

<span style="color: green;">(New)</span>

)</li>

<li>Introduction to Middleware Programming by tada (Published: 2002

<span style="color: red;">(Old)</span>

)</li>

</ul>

In this example, 1st book is marked as "(New)" because its publication year is 2023(2022 or later). While 2nd book is marked as "(Old)" because its publication year is assumed to be before 2020.

**Exercise 3: Constructors and DE structuring**

**Output:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Step2: .ejs Code Analysis:**

1. Review the .ejs file line by line and identify sections that contain JavaScript code within <% %> tags:

In the userInfo.ejs file, JavaScript code is embedded within <% %> tags. Here are the parts of .ejs file that contain JavaScript code:

userInfo.ejs:

<p>Name: <%=userName %></p>

<p>Age: <%=userAge %></p>

<p>Email: <%=userEmail %></p>

2. Analyze and understand the JavaScript code embedded in the .ejs file, including control structures, variable usage, and EJS-specific syntax:

In the .ejs file, JavaScript code within <% %> tags is used to embed variables (userName, userAge, and userEmail) into the HTML content. The variables are enclosed within <%= %> tags to indicate that their values should be displayed in the rendered HTML.

3. Take note of any notable features or functionalities implemented in the .ejs file.:

The .ejs file is a template for rendering user information, and it includes an HTML form for creating a user. When a user submits the form, the data is sent to the server for processing.

**Step3: HTML Code Analysis:**

1.Open the HTML file in a browser or a code editor:

A screenshot of a computer

Description automatically generated

2. Inspect the rendered HTML code in the browser or view the HTML file's source code:

A screenshot of a computer

Description automatically generated

3. Identify the parts of the HTML code that correspond to the JavaScript code in the .ejs file.:

The JavaScript code within <% %> tags is used to display user information within the HTML document.:

<p>Name: <%=userName %></p>

<p>Age: <%=userAge %></p>

<p>Email: <%=userEmail %></p>

4. Compare the rendered HTML code to the original .ejs code and note any differences or modifications.:

<body>

<h1>User Information:</h1>

<form action="/createUser" method="POST">

<input type="text" name="name" placeholder="Name" required="">

<input type="number" name="age" placeholder="Age" required="">

<input type="text" name="email" placeholder="Email" required="">

<button type="submit">Add User</button>

</form>

<p>Name: Sushmita Prafull Halasawade</p>

<p>Age: 25</p>

<p>Email: sushmitahalasawade@gmail.com</p> </body>

The rendered HTML code will include the user's name, age, and email provided through post response in place of <%=userName %>, <%=userAge %>, and <%=userEmail %>.

**Step4: documentation:**

1.Analysis Summary:

The .ejs file serves as a template for rendering user information. It uses EJS tags to embed JavaScript code for displaying user data within the HTML content.

2.Comparison: .ejs vs. HTML:

userInfo.ejs (.ejs Code):

<body>

<h1>User Information:</h1>

<form action="/createUser" method = "POST">

<input type = "text" name = "name" placeholder="Name" required>

<input type = "number" name = "age" placeholder="Age" required>

<input type = "text" name = "email" placeholder="Email" required>

<button type="submit">Add User</button>

</form>

<p>Name: <%=userName %></p>

<p>Age: <%=userAge %></p>

<p>Email: <%=userEmail %></p>

</body>

HTML Code Generated in the Browser:

<body>

<h1>User Information:</h1>

<form action="/createUser" method="POST">

<input type="text" name="name" placeholder="Name" required="">

<input type="number" name="age" placeholder="Age" required="">

<input type="text" name="email" placeholder="Email" required="">

<button type="submit">Add User</button>

</form>

<p>Name: Sushmita Prafull Halasawade</p>

<p>Age: 25</p>

<p>Email: sushmitahalasawade@gmail.com</p> </body>

A screenshot of a computer

Description automatically generated

3.Observations and Differences:

The HTML content within the .ejs file contains placeholders (<%=userName %>, <%=userAge %>, <%=userEmail %>) for user data, which are filled in with actual user information when the template is rendered by the server. This approach allows the template to display user data dynamically without the need for static HTML content for each user.

**Step5: Optional Enhancement:**

Modified result.ejs:

<ul>

<%books.forEach(book=>{%>

<li><%= book.title %> by <%= book.author%> (Published: <%= book.publicationYear%>

<% if (book.publicationYear >= 2020) { %>

<span style="color: green;">(New)</span>

<% } else { %>

<span style="color: red;">(Old)</span>

<% } %>)</li>

<% }); %>

</ul>

In .ejs file, I've added an <ul> element to display a list of users. We use EJS templating to iterate over an array called users. For each user in the array, we display their name, age, and email.

A screenshot of a computer

Description automatically generated

2. Analyze the corresponding changes in the rendered HTML code and document your findings.

Rendered HTML:

<body>

<h1>User List:</h1>

<form action="/createUser" method="POST">

<input type="text" name="name" placeholder="Name" required="">

<input type="number" name="age" placeholder="Age" required="">

<input type="text" name="email" placeholder="Email" required="">

<button type="submit">Add User</button>

</form>

<ul>

<li>

<p>Name: Sushmita Prafull Halasawade</p>

<p>Age: 25</p>

<p>Email: sushmitahalasawade@gmail.com</p>

</li>

<li>

<p>Name: Aishwarya Prafull Halasawade</p>

<p>Age: 26</p>

<p>Email: aishhalasawade@gmail.com</p>

</li>

</ul>

The HTML page will now display a list of users with their information in an unordered list (<ul>). The user information is obtained from the user’s array, which is iterated using EJS.