

C237: Software Application Development

LESSON 06B: FORM PROCESSING IN NODE.JS WITH EXPRESS AND EJS

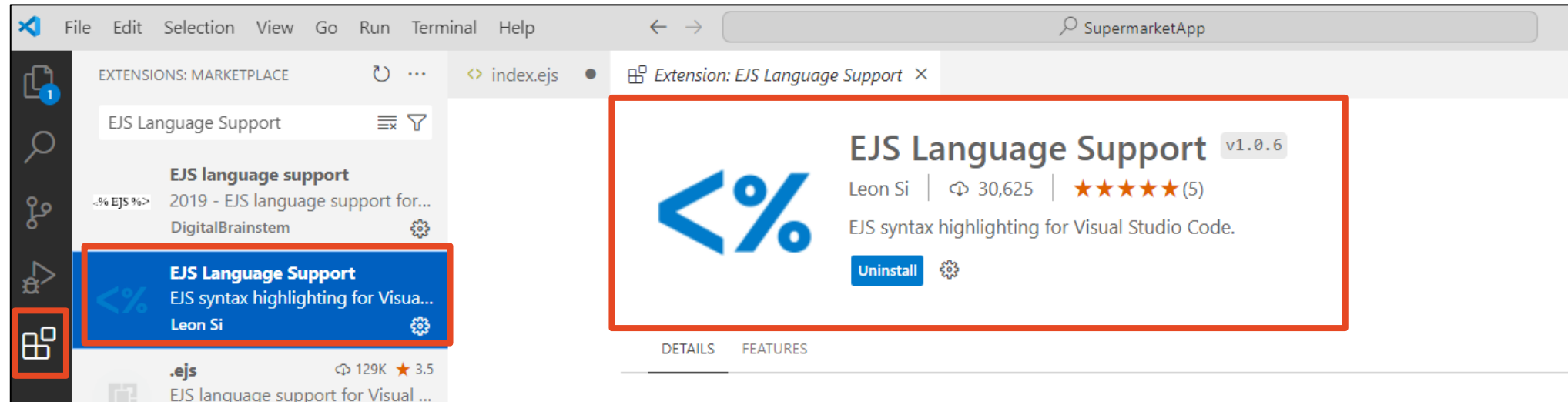
1.45PM – 4.00PM

EJS Templates

PASSING DATA TO EJS TEMPLATES

Before we start

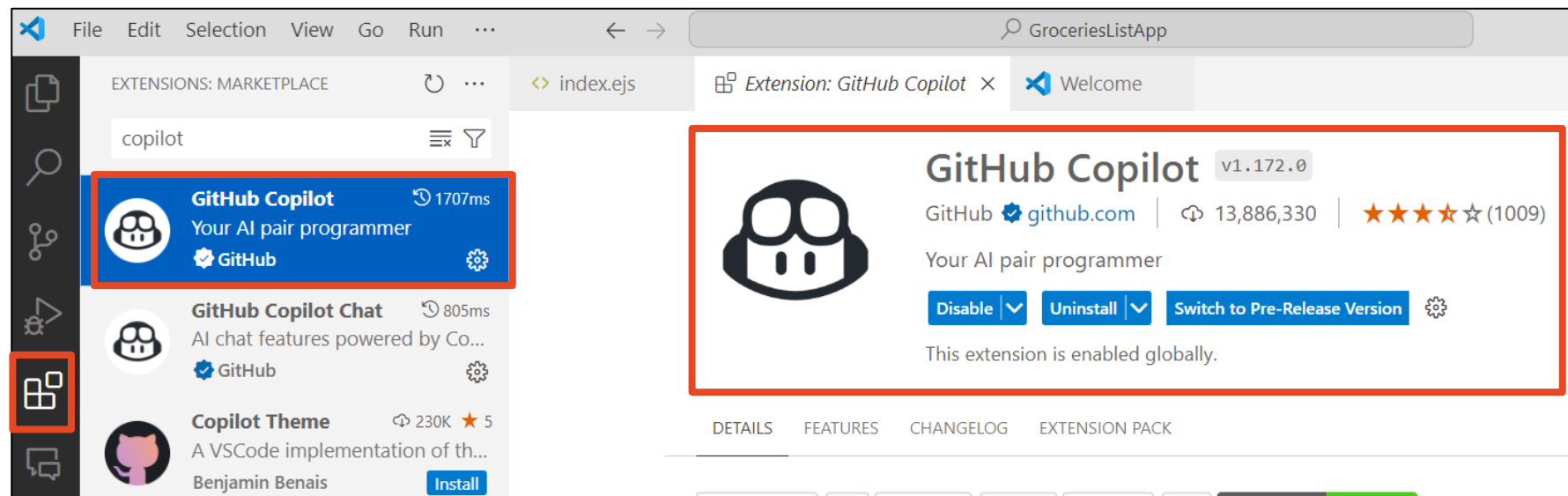
- Open Visual Studio Code and install the EJS Language Support extension.



- This extension will allow VS Code to do syntax highlighting for EJS. The EJS tags will be highlighted with a different color to stand out for easier viewing.

Another useful extension

- Install the GitHub Copilot extension.



- GitHub Copilot is an AI-powered code completion tool developed by GitHub in collaboration with OpenAI. It is designed to assist developers by providing context-aware code suggestions and completions directly within their Integrated Development Environment (IDE).

Passing Data to EJS Templates <%= %>

- EJS can pass data to templates and use that data to generate dynamic HTML.
- You can pass data to a template by providing an object as the second argument to the render function.
- For instance, you want to pass a variable called name to your index.ejs template:

```
app.get('/', (req, res) => {  
  |   res.render('index', { name: 'Peter' });  
  |  
  |});
```

- You can use the variable in your index.ejs template in this way.

```
<body>  
|   <h1>Hello <%= name %>!</h1>  
|  
</body>
```

- When the template is rendered, that is, when the script is executed, the name variable will be replaced with the value(Peter) passed to the render function.

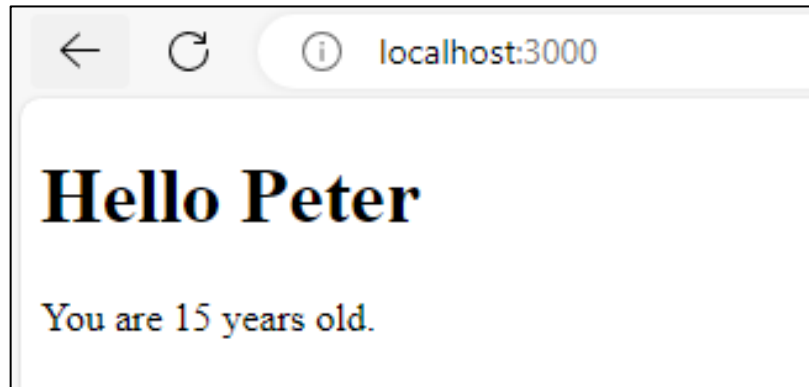
Activity 1

- Copy the “EJSTemplateExercise” folder located in the provided **L06b Resources** folder to the L06 folder you created earlier.
- In Visual Studio Code, open the folder “EJSTemplateExercise” found in your L06 folder
- Edit app.js to include another variable “age” to be passed to the index.ejs page

```
//Define a route to render the index page
app.get('/', (req, res) => {
  res.render('index',
    {
      name : 'Peter',
      age: 15
    });
});
```

Activity 1

- Edit index.ejs to display “You are **age** years old” where age is the number passed in from app.js.
- Sample output:



Conditional Statements in EJS

- You can also use JavaScript conditional statements to make your express app more dynamic.
- With conditional statements like **if..else statements**, You can choose to display some part of your app or not.

- For Example:

```
<body>
  <% if (numberOfNames >= 5) {%>
    <p> There are at least 5 people on the list </p>
  <% } else { %>
    <p> There are less than 5 people on the list</p>
  <% } %>
</body>
```

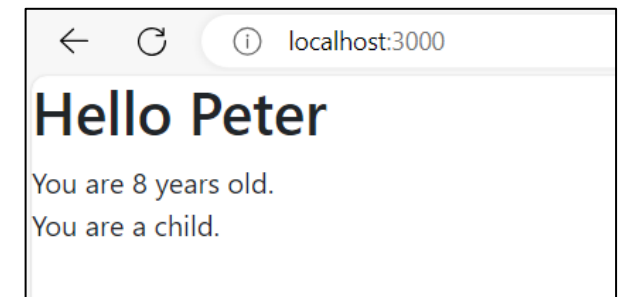
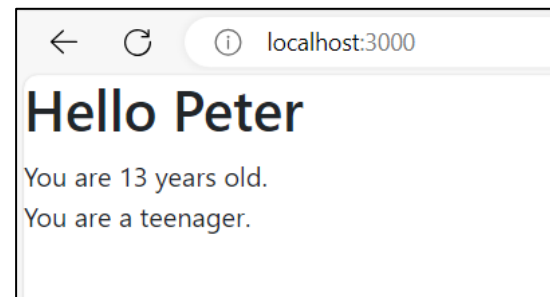
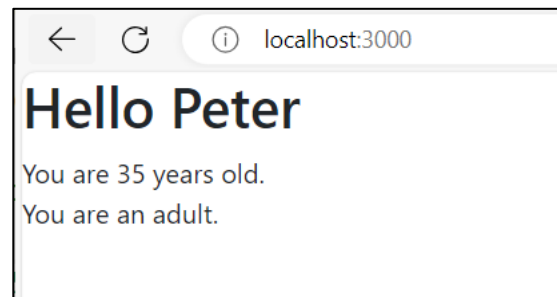
- The variable **numberOfNames** created below the for loop saves the number of elements in the data array. Then, update your index.ejs file to perform an action based on some condition about the numberOfNames variable.

Activity 2

- With the folder “EJSTemplateExercise” still open in Visual Studio Code
- Edit index.ejs to display the last statement base on the conditions in the table below:

Age	Statement to display
18 and above	“You are an adult.”
Between 13 to 17 (inclusive)	“You are a teenager.”
12 and below	“You are a child.”

- Sample output:



Form processing with EJS

- EJS allows us to embed JavaScript code directly into HTML templates, making it easy to generate dynamic content.

- For Example:

```
<body>
  <h1>Submit Form</h1>
  <form action="/submit" method="post">
    <label for="name">Name:</label>
    <input type="text" id="name" name="name" required>
    <br>
    <br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email" required>
    <br>
    <br>
    <button type="submit">Submit</button>
  </form>
</body>
```

form.ejs

```
<body>
  <h1>Form Submitted Successfully</h1>
  <p>Name: <%= name %></p>
  <p>Email: <%= email %></p>
</body>
```

submitted.ejs

```
// Define a route to handle form submission
app.post('/submit', (req, res) => {
  const { name, email } = req.body;
  res.render('submitted', { name, email });
});
```

app.js

- Information entered in a form can be displayed on another page after the form have been submitted.

Form processing with EJS

➤ **req.body** in Express.js is a property of the request object (req) that contains the parsed request body of the HTTP POST or PUT request sent by the client.

➤ **Parsing the Request Body:**

➤ When a client sends a POST or PUT request with data (like form submissions), Express does not automatically parse the data into a usable format.

```
// Import required modules
const express = require('express');
const bodyParser = require('body-parser');

// Create an Express application
const app = express();
// Middleware to parse request bodies
app.use(bodyParser.urlencoded({ extended: true }));
```

➤ With this middleware added to your Express application, incoming request bodies are parsed and made available under **req.body**.

Form processing with EJS

➤ Accessing Form Data:

- After parsing, **req.body** will contain an object with key-value pairs representing the form fields sent by the client.
- For example, if you have a form field with the name **name**, you can access its value using **req.body.name**.

```
app.post('/submit', (req, res) => {  
  const name = req.body.name;  
  const email = req.body.email;  
  // Access other form fields as needed  
  
  res.render('submitted', {name, email})  
});
```

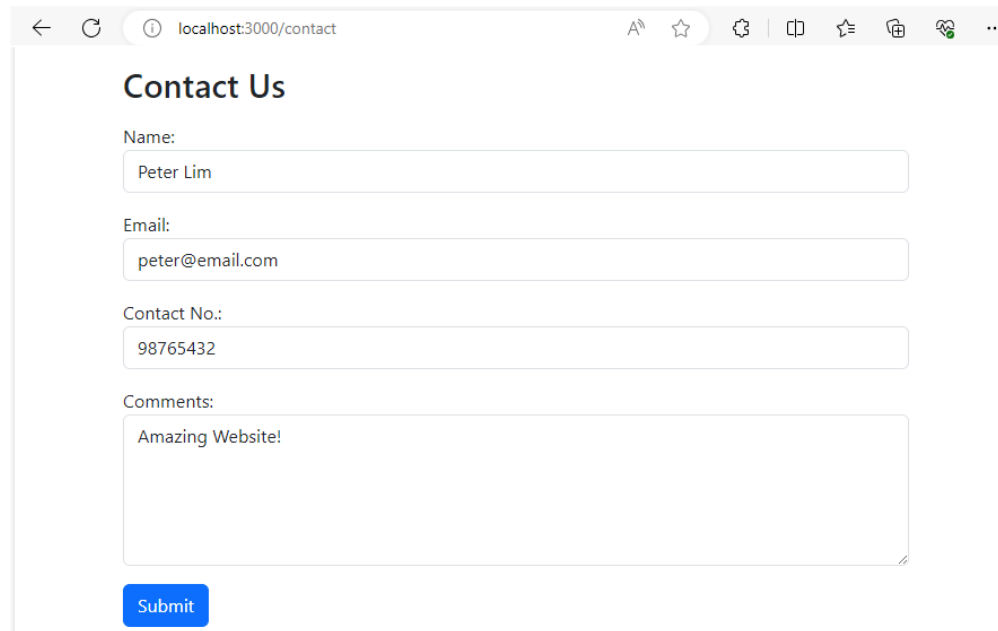
OR

```
app.post('/submit', (req, res) => {  
  const { name, email } = req.body;  
  // Access other form fields as needed  
  
  res.render('submitted', {name, email})  
});
```

- By using **req.body**, you can access the data sent by the client and use it in your Express routes to process and respond accordingly. It's an essential part of handling form submissions and API requests in an Express.js application.

Activity 3

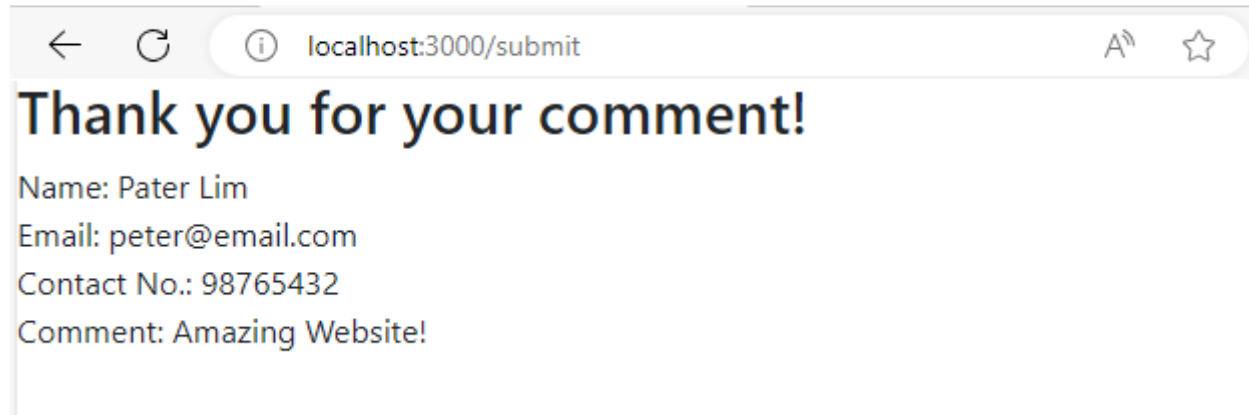
- With the folder “EJSTemplateExercise” still open in Visual Studio Code
- Edit contact.ejs to include more form fields – Contact No. and Comments.
- Sample output:



A screenshot of a web browser window displaying a contact form. The browser's address bar shows 'localhost:3000/contact'. The form is titled 'Contact Us' and contains four input fields: 'Name' (filled with 'Peter Lim'), 'Email' (filled with 'peter@email.com'), 'Contact No.' (filled with '98765432'), and 'Comments' (filled with 'Amazing Website!'). A blue 'Submit' button is located at the bottom left of the form.

Activity 3

- With the folder “EJSTemplateExercise” still open in Visual Studio Code
- Update **app.js** to include key-value pairs representing the form fields sent by the client from the form page.
- Edit **submitted.ejs** to display the information entered by user in the contact page:
- Sample output:



Using EJS to Include Reusable Template Components <%- %>

- EJS also allows you to combine different templates for different sections of your code, like your head, header, and footer sections.
- You can add .ejs files to a current template using the <%- include ('file.ejs') %> statement.
- For example, if you have a template called header.ejs that you want to include in your index.ejs template, you can use the following code in your index.ejs template:

```
<header>
  <nav>
    <ul>
      <li><a href="/">Home</a></li>
      <li><a href="/about">About</a></li>
      <li><a href="/contact">Contact</a></li>
    </ul>
  </nav>
  <h1>My Website</h1>
</header>
```

header.ejs

```
<body>
  <%- include ('header.ejs') %>
  <h1>Hello <%= name %>!</h1>
</body>
```

index.ejs

Using EJS to Include Reusable Template Components

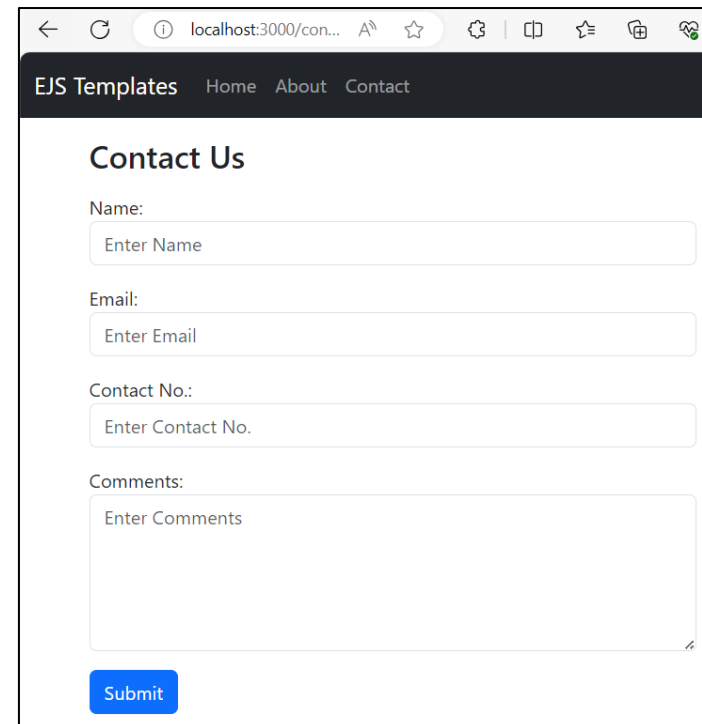
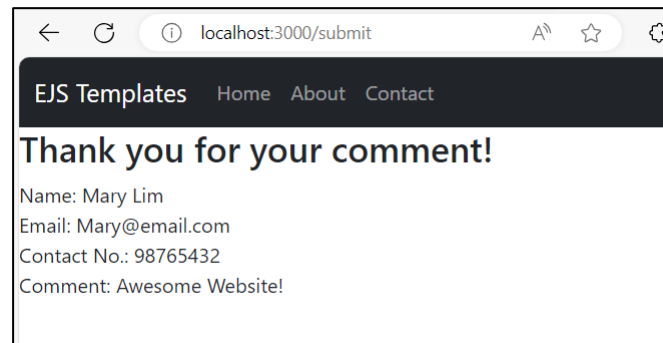
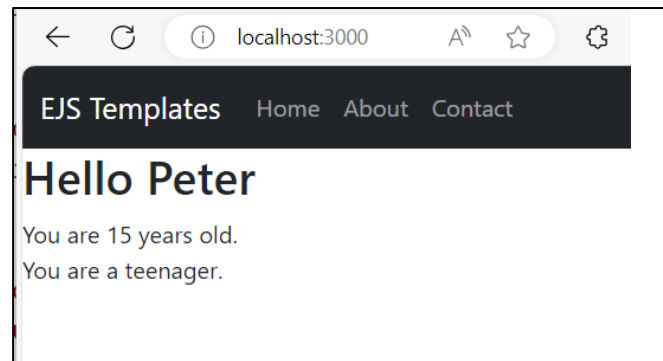
- This will include the contents of the header.ejs template to that point in the index.ejs template.
- It is common practice to save templates that you will want to reuse across multiple pages in a partials subdirectory in the views directory.
- To include the header in your code after doing this, attach the file path to the former syntax:

```
<%- include ('partials/header') %>
```

- The <%- include %> tag makes it simple to **reuse common HTML elements** across multiple pages or views, simplifies code maintenance, and preserves a consistent look across your application.

Activity 4

- With the folder “EJS_TemplateExercise” still open in Visual Studio Code
- Edit all ejs pages to include the header (found in the partials folder) to your page.
- Sample output:



Form Processing with EJS

L06B ACTIVITY

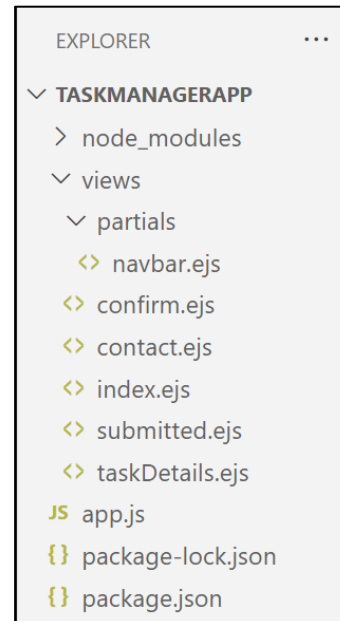
L06b Activity

- Create a **Task Manager web application** using Express.js and EJS.
- The application should have the following functionality:
 - Display a **form** with fields for **Task title, Description, Deadline, and priority (Low, Medium, High)**.
 - When the form is submitted, **redirect the user to a confirmation page** which displays a “Task Added” message and the details entered by the user.
 - Your page should include bootstrap navbar.

L06b Activity

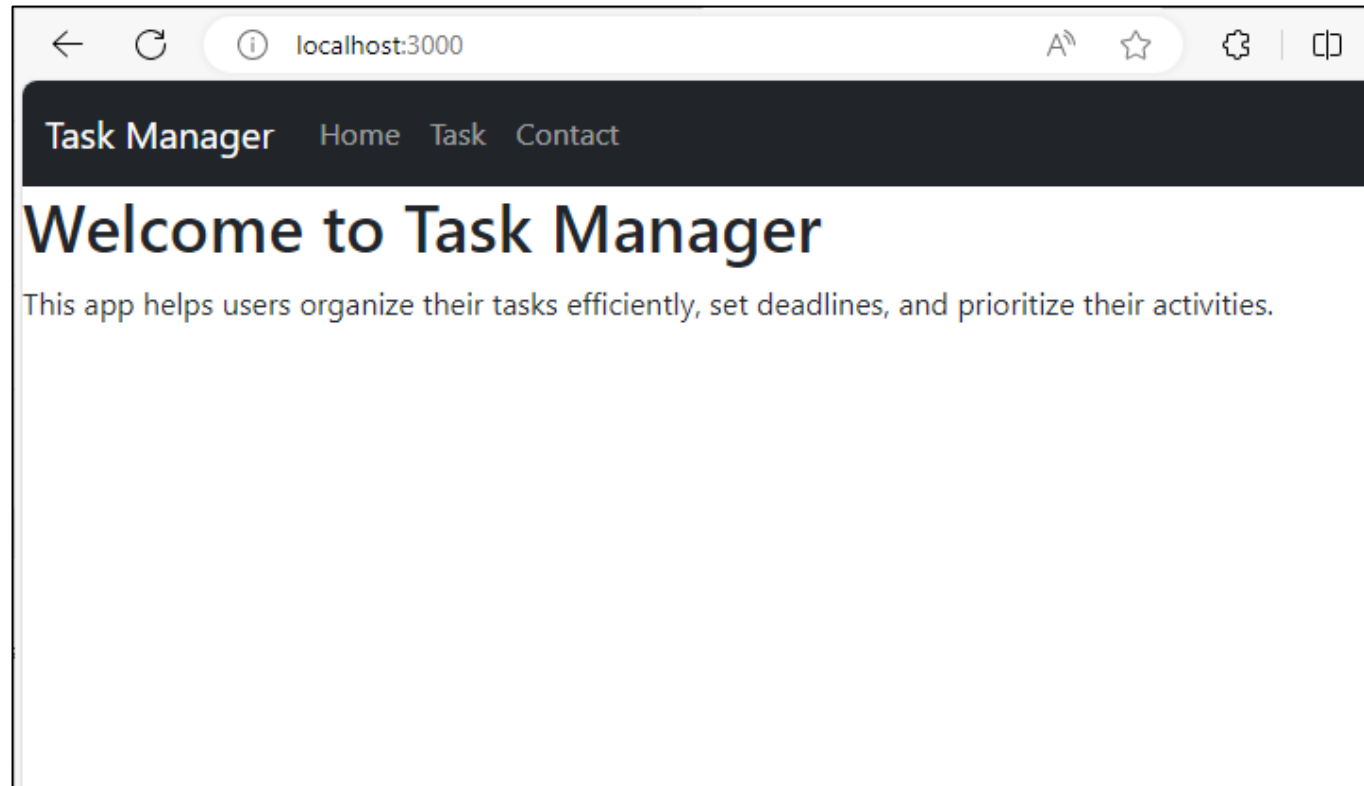
➤ Steps to complete the exercise:

1. Set up a new Express.js project
2. Create EJS templates for the registration form and any additional views
3. Implement routes and middleware for handling form submissions and rendering views
4. Basic Structure of your Express.js application:



L06b Activity

➤ Sample Screenshot (Home page – index.ejs):



L06b Activity

➤ Sample Screenshot (Task page – taskDetails.ejs):

The screenshot shows a web browser window with the address bar displaying `localhost:3000/task`. The browser's navigation bar includes back, forward, and refresh buttons, along with various extension icons. The application's navigation bar is dark and contains the text "Task Manager" followed by links for "Home", "Task", and "Contact".

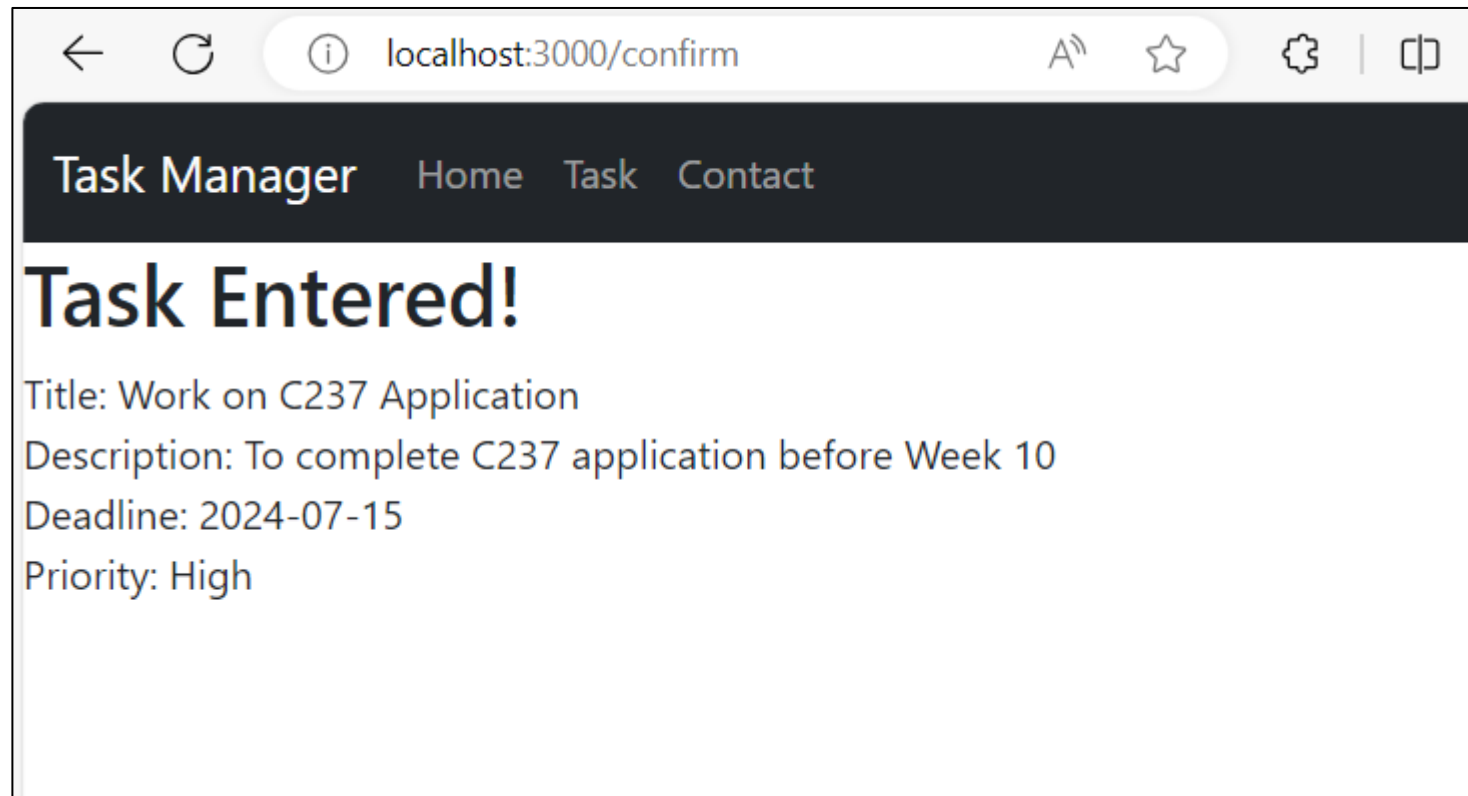
The main content area is titled "Task Details" and contains the following form fields:

- Title:** A text input field with the placeholder text "Enter task title".
- Description:** A text input field with the placeholder text "Enter task description".
- Deadline:** A text input field with the placeholder text "dd/mm/yyyy" and a calendar icon on the right.
- Priority:** A text input field with the placeholder text "Low".

A blue "Submit" button is located at the bottom of the form.

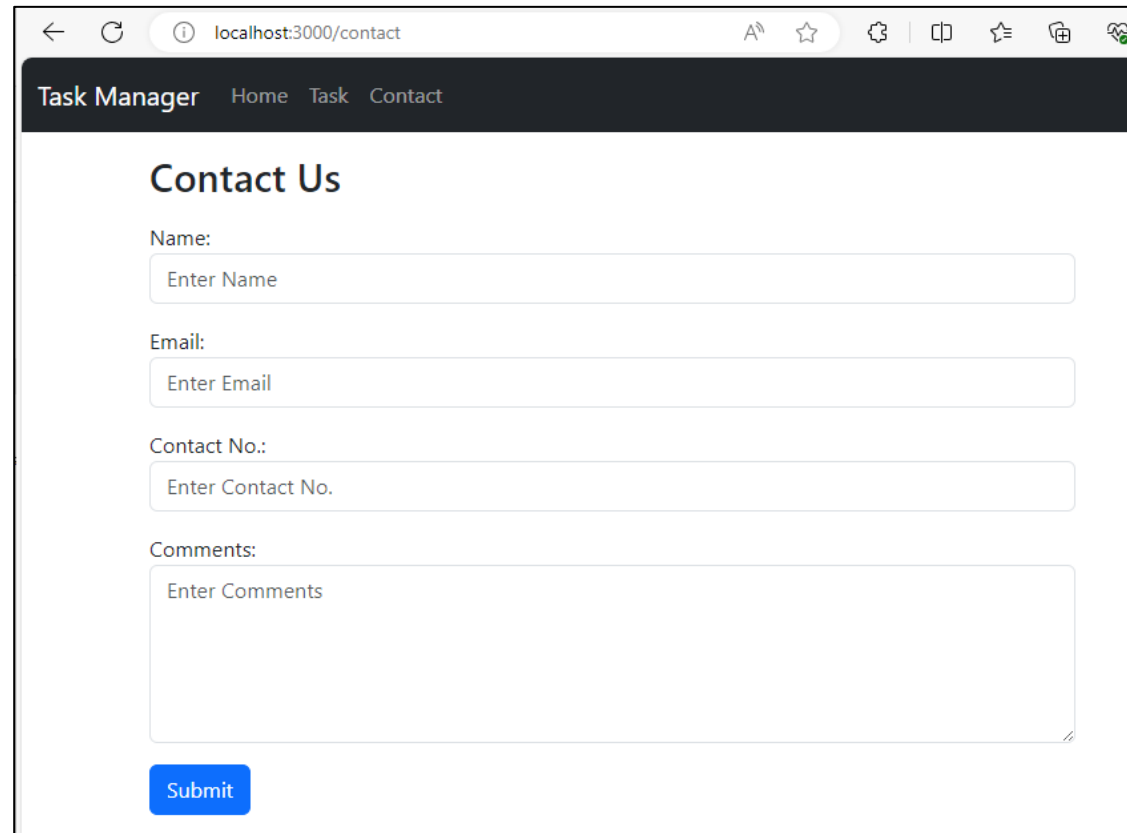
L06b Activity

➤ Sample Screenshot (Confirmation page – confirm.ejs):



L06b Activity

➤ Sample Screenshot (Contact page – contact.ejs):



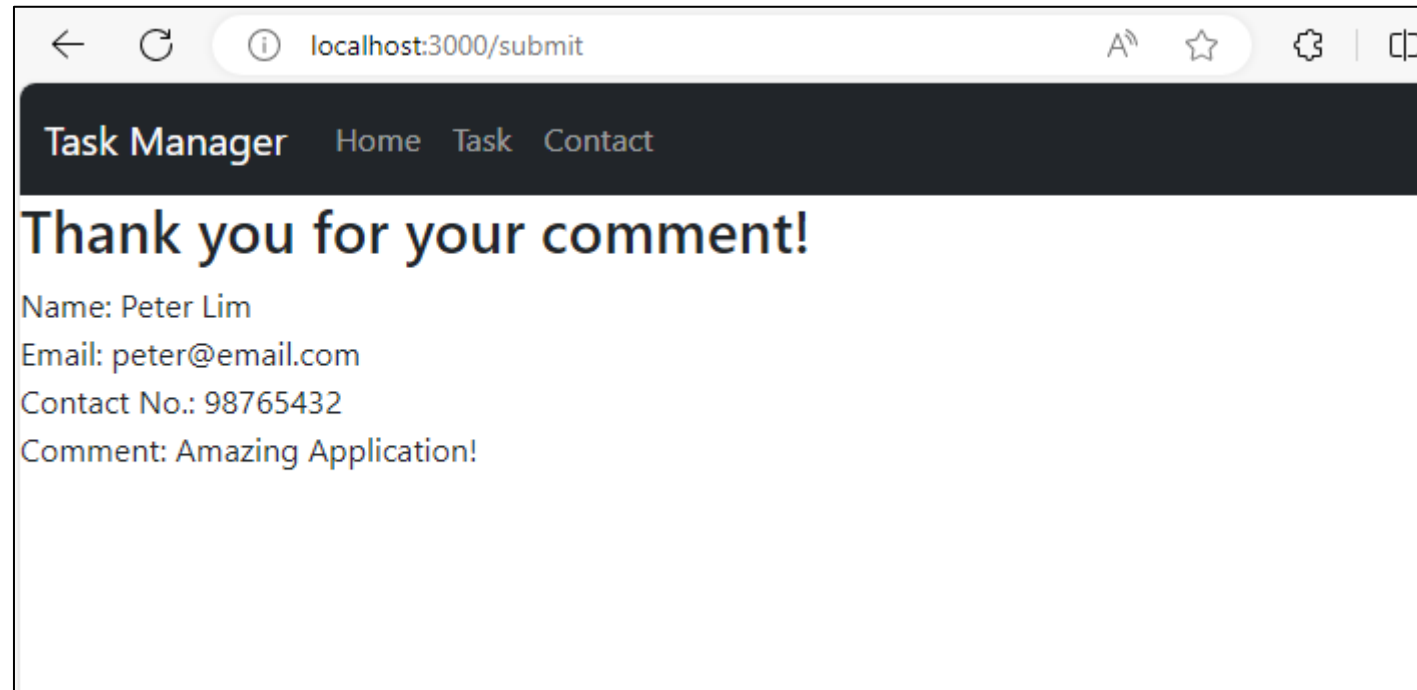
The screenshot shows a web browser window with the address bar displaying `localhost:3000/contact`. The browser's navigation bar includes a dark header with the text "Task Manager" and a navigation menu with links for "Home", "Task", and "Contact". The main content area is titled "Contact Us" and contains a form with the following fields:

- Name:** A text input field with the placeholder text "Enter Name".
- Email:** A text input field with the placeholder text "Enter Email".
- Contact No.:** A text input field with the placeholder text "Enter Contact No.".
- Comments:** A larger text area with the placeholder text "Enter Comments".

At the bottom of the form is a blue "Submit" button.

L06b Activity

➤ Sample Screenshot (Thank You page – submitted.ejs):



Submissions (Before 2359hrs **TODAY**)

- L06a Activity
- L06b Activity

What have you learnt?

- EJS Templates
 - Passing Data to EJS templates
 - Using EJS to Include Reusable Template Components
 - Conditional Statements in EJS
 - Form processing with EJS