**WP Lab Exp. 9**

**Form Handling using Express and Nodejs**

1. **Steps to install nodejs in Windows**
2. **Download Node.js Installer:**
   * Go to the official Node.js website.
   * Link - https://nodejs.org/en/download/source-code
   * Download the latest stable version for Windows (use the LTS version if you're unsure).
3. **Run the Installer:**
   * Once the download is complete, run the installer.
   * Follow the prompts in the installer (accept the license agreement, choose the installation directory, etc.).
   * Make sure to check the box that says **"Automatically install the necessary tools"** (includes npm).
4. **Check Installation:**
   * Open **Command Prompt** or **PowerShell**.

Run the following command to check if Node.js and npm were installed successfully:  
bash  
Copy code  
node -v

npm -v

These commands should display the installed versions of Node.js and npm.

### **B. Steps to create the project:**

**Initialize Node.js Project:** Run these commands in your terminal:  
  
mkdir form-handling-example

cd form-handling-example

npm init -y

**Install Express:**  
npm install express

**Project Structure:** Create the following structure:  
  
form-handling-example/

├── public/

│ └── style.css

├── views/

│ └── index.html

├── app.js

└── package.json

### **1. app.js (Express Application)**

This is your main Node.js application:

const express = require('express');

const path = require('path');

const bodyParser = require('body-parser');

const app = express();

const port = 3000;

// Middleware to parse the form data

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

// Serve static files (for serving CSS)

app.use(express.static(path.join(\_\_dirname, 'public')));

// Route to display the form

app.get('/', (req, res) => {

res.sendFile(path.join(\_\_dirname, 'views', 'index.html'));

});

// Route to handle form submission

app.post('/submit-form', (req, res) => {

const { name, email } = req.body;

res.send(`<h1>Form Submitted</h1><p>Name: ${name}</p><p>Email: ${email}</p>`);

});

// Start the server

app.listen(port, () => {

console.log(`Server running at http://localhost:${port}`);

});

### **2. index.html (Form Template)**

This is the form that users will interact with:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Form Handling with Express</title>

<link rel="stylesheet" href="/style.css">

</head>

<body>

<div class="form-container">

<h1>Form Handling</h1>

<form action="/submit-form" method="POST">

<label for="name">Name:</label>

<input type="text" id="name" name="name" required>

<br>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<br>

<button type="submit">Submit</button>

</form>

</div>

</body>

</html>

### **3. style.css (External Stylesheet)**

This is the CSS file to style the form and set the background color:

body {

background-color: #f0f8ff;

font-family: Arial, sans-serif;

}

.form-container {

width: 300px;

margin: 50px auto;

padding: 20px;

background-color: white;

box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

h1 {

text-align: center;

}

label {

display: block;

margin: 10px 0 5px;

}

input {

width: 100%;

padding: 8px;

margin-bottom: 10px;

}

button {

width: 100%;

padding: 10px;

background-color: #007bff;

color: white;

border: none;

cursor: pointer;

}

button:hover {

background-color: #0056b3;

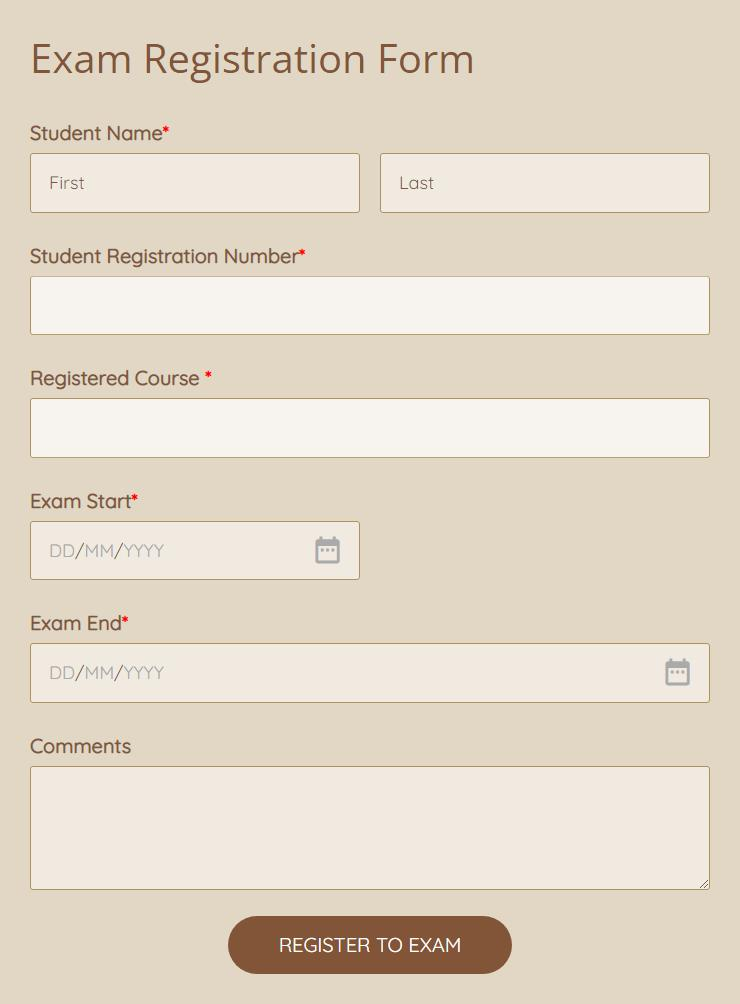
}

### **4. Run the Application:**

1. Save the files.

Start the server by running:  
  
node app.js

1. Open your browser and navigate to http://localhost:3000. You will see a form with a background color applied from the external CSS.
2. Submit the form, and you should see a confirmation page displaying the form data.

**Form :** 

**Fields to have in Form :**

* **Student Name**
* **Email**
* **Contact**
* **Class**
* **Branch**
* **Semester**
* **Student Hallticket Number**
* **Submit Button**

**\*Instructions :**

1. **Make use of box-sizing**
2. **box-shadow**
3. **Color for button, etc**