Step 1: Install Node.js and npm

Option A: Official Website (Recommended for All Platforms)

1. **Visit the official Node.js website**: https://nodejs.org/
2. **Download the LTS version** (Long Term Support) - click the green button
3. **Run the installer**:
   - On Windows: Run the `.msi` file and follow the setup wizard
   - On macOS: Run the `.pkg` file and follow the installer
   - On Linux: Extract and follow the installation instructions
4. **During installation**: Make sure to check "Automatically install the necessary tools"

Option B: Package Managers (Alternative)

Windows (using winget):

Shell
```
winget install OpenJS.NodeJS
```

macOS (using Homebrew):

Shell
```
brew install node
```

Linux (Ubuntu/Debian):

Shell
```
sudo apt update
sudo apt install nodejs npm
```

Step 2: Verify Installation

After installation, **restart your terminal/command prompt** and run:

```Shell
node --version
npm --version
```

Expected output (versions may differ):

```None
v20.10.0
10.2.3
```

Troubleshooting Installation Issues

Windows PowerShell Execution Policy Error

If you get an error like "cannot be loaded because running scripts is disabled":

1. **Open PowerShell as Administrator**
2. **Run this command**:

```None
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope
CurrentUser
```

3. **Type `Y`** when prompted

Alternative: Use Command Prompt (cmd)

If PowerShell continues to have issues, use Command Prompt instead:

- Press `Win + R`, type `cmd`, press Enter
- All npm commands work the same in cmd

Step 3: Create Your Project

1. **Create a new folder** for your project:

Shell
```
mkdir my-express-project
cd my-express-project
```

2. **Initialize a new npm project**:

Shell
```
npm init -y
```

3. This creates a `package.json` file with default settings.

Step 4: Install Express.js

Install Express and EJS template engine:

Shell
```
npm install express ejs
```

Verify installation - your `package.json` should now include:

JSON
```
{
  "dependencies": {
    "express": "^4.18.x",
    "ejs": "^3.1.x"
  }
}
```

Step 5: Install and Configure ESLint

Install ESLint

```shell
npm install --save-dev eslint
```

Initialize ESLint Configuration

```shell
npx eslint --init
```

When prompted, select these options:

1. **How would you like to use ESLint?**
   → `To check syntax and find problems`
2. **What type of modules does your project use?**
   → `CommonJS (require/exports)`
3. **Which framework does your project use?**
   → `None of these`
4. **Does your project use TypeScript?**
   → `No`
5. **Where does your code run?**
   → `Node` (use spacebar to select, Enter to confirm)
6. **What format do you want your config file to be in?**
   → `JavaScript`
7. **Install additional dependencies?**
   → `Yes` (if prompted)

Step 6: Update package.json Scripts

Edit your `package.json` file and add these scripts:

JSON
```json
{
  "scripts": {
    "start": "node index.js",
    "lint": "eslint .",
    "lint:fix": "eslint . --fix",
    "test": "echo \"No tests yet\" && exit 0"
  }
}
```

Step 7: Create Project Structure

Create the following folders and files:

None
```
my-express-project/
├── index.js                    (your main server file)
├── package.json               (created by npm init)
├── .eslintrc.js               (created by ESLint init)
├── views/                     (folder for EJS templates)
│   ├── home.ejs
│   ├── create.ejs
│   ├── update.ejs
│   └── delete.ejs
└── public/                    (folder for static files)
    └── css/
        └── style.css
```

Create folders:

Shell
```shell
mkdir views
mkdir public
mkdir public/css
```

Step 8: Test Your Setup

Create a Simple Test File

Create `index.js` with this basic Express server:

```JavaScript
const express = require('express');
const app = express();
const PORT = 8080;

app.get('/', (req, res) => {
    res.send('<h1>Hello Express!</h1><p>Setup successful!</p>');
});

app.listen(PORT, () => {
    console.log(`Server running at http://localhost:${PORT}`);
});
```

Run Your Server

```Shell
npm start
```

Expected output:

```None
Server running at http://localhost:8080
```

Test in Browser

Open your browser and go to: `http://localhost:8080`
You should see "Hello Express!" message.

Test ESLint

```shell
npm run lint
```

If there are no errors, you'll see something like: ✨ `Done in 0.xx s`

Step 9: Common Commands Reference

Command	Description
`npm start`	Start your Express server
`npm run lint`	Check code for ESLint errors
`npm run lint:fix`	Auto-fix ESLint errors where possible
`npm install <package>`	Install a new package
`node index.js`	Run your server directly
`Ctrl + C`	Stop the running server

Troubleshooting Common Issues

1. "Module not found" errors

Solution: Run `npm install` to install all dependencies

2. "Port already in use" error

Solutions:

- Change the PORT number in your `index.js`
- Or kill the process using that port:

Shell
```
# Windowsnetstat -ano | findstr :8080taskkill /PID <PID_NUMBER>
/F# macOS/Linuxlsof -ti:8080 | xargs kill
```

3. ESLint not working

Solution: Make sure ESLint is installed locally in your project:

Shell
```
npm install --save-dev eslint
```

4. Cannot access localhost

Check:

- Server is running (you should see the console message)
- Correct URL: `http://localhost:8080` (not https)
- No firewall blocking the port

Getting Help

If you encounter issues:

1. **Check the error message** carefully - it usually tells you what's wrong
2. **Ask classmates** - they might have faced the same issue
3. **Search online** - most Node.js/Express errors have been solved before
4. **Ask your instructor** during office hours

Important Notes

- **Always restart your server** (`Ctrl+C` then `npm start`) after making changes to `index.js`
- **ESLint errors won't stop your server** - they're just warnings to help write better code
- **Keep your terminal open** while developing to see error messages
- **Save all files** before testing changes