

# **PHP Simple To-Do List**

## **Cover Page**

- **Title:** PHP Simple To-Do List
- **Subtitle:** Using PHP and HTML5
- **Name:** Chaitanya Singh Negi
- **Date:** 8th Aug 2024

## **Table of Contents**

1. Abstract
2. Objective
3. Introduction
4. Methodology
  - Setting Up the Environment
  - Creating the HTML Frontend
  - Developing the PHP Backend
  - CSS Styling
  - Testing
5. Code
  - HTML Code
  - PHP Code
  - CSS Code
6. Screenshots of Output
7. Conclusion

## **Abstract**

This report describes the development of a simple to-do list application using PHP and HTML5. The application allows users to manage a list of tasks through a web interface, with tasks stored in a session rather than a file. This approach simplifies the application by avoiding file handling and focusing on session management. The project demonstrates how to use PHP for server-side scripting and HTML5 for building a dynamic, interactive web interface. Additionally, it highlights the practical use of PHP sessions for maintaining user data across page reloads, making it suitable for understanding basic state management in web applications.

## Objective

The main objectives of this project are to:

1. Develop a functional to-do list application using PHP and HTML5.
2. Implement functionality to add and delete tasks.
3. Use PHP sessions for data storage to simplify file management.
4. Create a clean and user-friendly interface using HTML5 and CSS.
5. Demonstrate basic session handling in PHP for managing state without relying on databases.
6. Ensure the application is responsive and accessible on different devices by using HTML5 and CSS best practices.
7. Provide a practical example of integrating server-side logic with client-side design to build a complete web application.

## Introduction

To-do list applications are pivotal in helping individuals and teams organize their tasks, manage their time effectively, and improve productivity. These applications are often used in personal and professional settings to track tasks, set deadlines, and monitor progress. This project aims to develop a basic to-do list application using PHP and HTML5, showcasing a practical example of dynamic web development.

The choice to use PHP sessions for task management offers a simplified approach compared to database storage, making the application lightweight and easy to understand. PHP sessions allow the application to maintain user-specific task lists across page reloads, providing a seamless user experience without relying on complex file or database management. This method demonstrates how server-side scripting can handle state management efficiently.

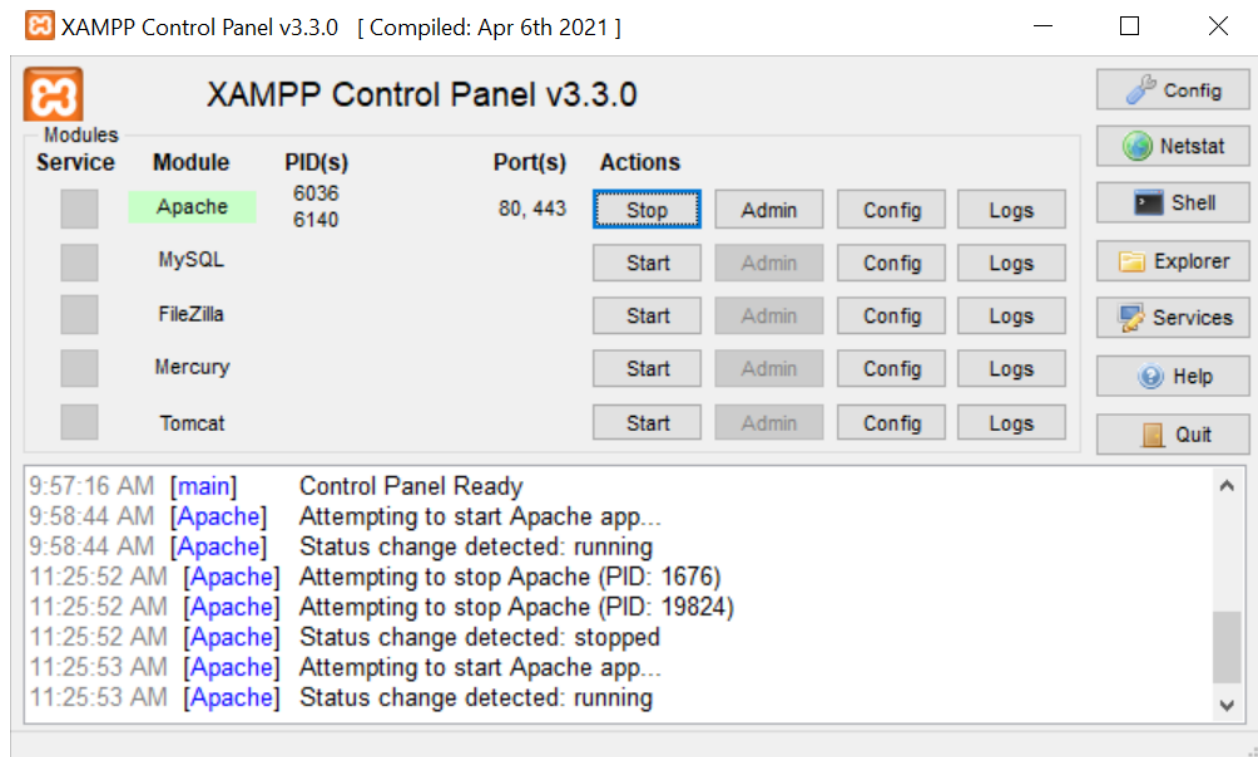
HTML5 is employed to create a modern, accessible user interface that is compatible with various devices and browsers. The use of CSS ensures that the application has a clean, user-friendly design, enhancing usability. By focusing on these technologies, the project illustrates fundamental concepts in web development, including session management, dynamic content rendering, and responsive design. This application not only fulfills the basic requirements of a to-do list but also serves as an educational tool for understanding the integration of server-side and client-side technologies.

## Methodology

### Setting Up the Environment

#### 1. Install XAMPP:

- **Download and Install XAMPP:** Obtain the XAMPP control panel from the XAMPP official website. XAMPP provides an easy-to-install package that includes Apache, PHP, and MySQL. Follow the installation instructions for your operating system.
- **Start XAMPP:** Once installed, open the XAMPP control panel and start the Apache server. This action will initiate both the web server and PHP interpreter, making them ready for use.



#### 2. Set Up Project Directory:

- **Create Project Directory:** Navigate to the htdocs directory within the XAMPP installation directory (typically found in 'C:\xampp\htdocs' on Windows). Create a new directory named '1Stop.ai\_To\_Do\_List' to hold your project files.
- **Configure Permissions:** Ensure that the project directory has the appropriate permissions so that Apache can read and write files as needed. This step might not

be necessary on all operating systems, but it's important to check for any permission issues.

### 3. Install a Code Editor:

- **Choose a Code Editor:** Select a code editor suitable for PHP and HTML development, such as Visual Studio Code. Install and configure the editor to enhance your development workflow.
- **Install Extensions:** Consider installing relevant extensions or plugins for PHP and HTML development to assist with coding, debugging, and managing your project.

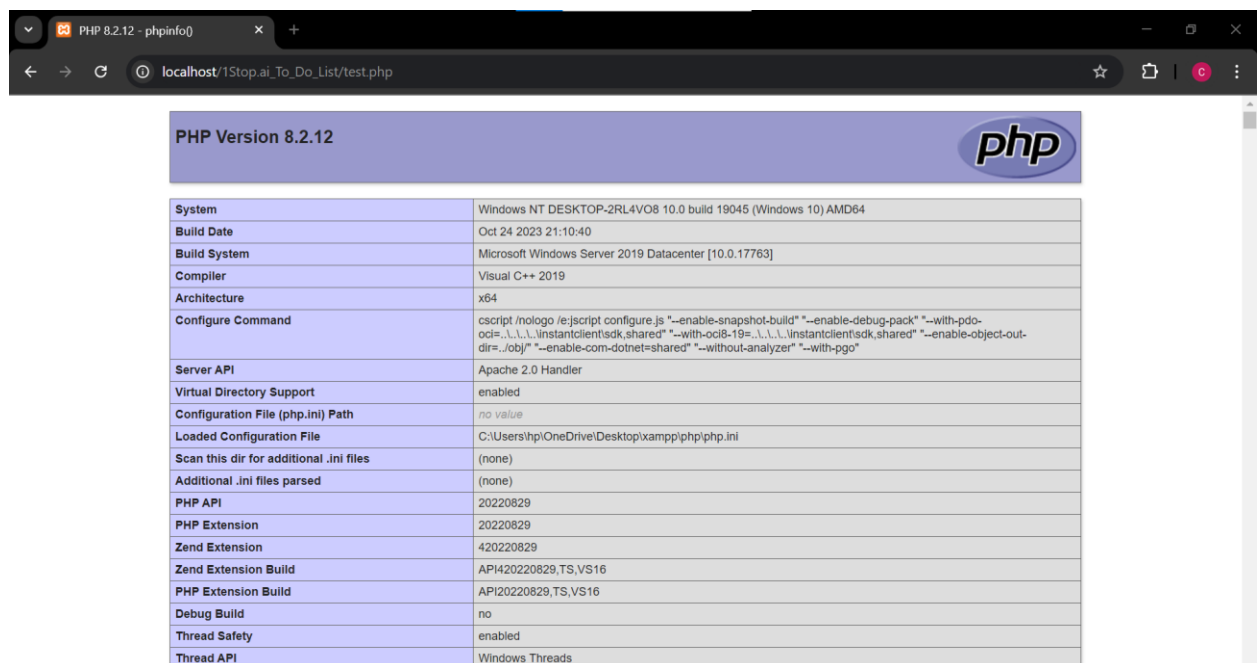
### 4. Test the Environment:

- **Create a Test PHP File:** Create a simple PHP file (e.g., test.php) in the project directory with the following content to ensure PHP is working correctly:

Code in a PHP file named test.php :

```
<?php
phpinfo();
?>
```

- **Access the Test File:** Open a web browser and navigate to 'http://localhost/1Stop.ai\_To\_Do\_List/test.php'. You should see the PHP information page, confirming that PHP is correctly installed and configured.



PHP Version 8.2.12

System	Windows NT DESKTOP-2RL4VO8 10.0 build 19045 (Windows 10) AMD64
Build Date	Oct 24 2023 21:10:40
Build System	Microsoft Windows Server 2019 Datacenter [10.0.17763]
Compiler	Visual C++ 2019
Architecture	x64
Configure Command	cmdscript /nologo /e:jscrip configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-odbc=.\..\..\..\instantclient\shared" "--with-oci8-19=.\..\..\..\instantclient\shared" "--enable-object-out-dir=.\obj" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	no value
Loaded Configuration File	C:\Users\hp\OneDrive\Desktop\xampp\php\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20220829
PHP Extension	20220829
Zend Extension	420220829
Zend Extension Build	API420220829,TS,VS16
PHP Extension Build	API20220829,TS,VS16
Debug Build	no
Thread Safety	enabled
Thread API	Windows Threads

## **Creating the HTML Frontend**

### **1. Design the Web Page:**

- Create an HTML page for handling task input and display.
- Link to an external CSS file for styling.

## **Developing the PHP Backend**

### **1. Session Management:**

- Use PHP sessions to manage and persist tasks across page reloads.

## **CSS Styling**

- Create a CSS file for styling the application.

## **Testing**

### **1. Run the Application:**

- Test the application to ensure it correctly handles task management using sessions.

### **2. Test Scenarios:**

- Verify that tasks can be added and deleted.
- Check for proper handling of empty task entries and session management.

### **3. Debugging:**

- Address any issues encountered during testing.

## **Code**

### **HTML Code (index.html)**

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple To-Do List</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container">
        <h1>Simple To-Do List</h1>
        <form method="post" action="tasks.php">
            <input type="text" name="task" placeholder="Enter new task" required>
            <button type="submit" name="add">Add Task</button>
        </form>
        <ul>
            <?php include 'tasks.php'; ?>
        </ul>
    </div>
</body>
</html>

```

### PHP Code (tasks.php)

```

<?php
session_start();

if (!isset($_SESSION['tasks'])) {
    $_SESSION['tasks'] = array();
}

if (isset($_POST['add'])) {
    $task = htmlspecialchars($_POST['task']);
    if (!empty($task)) {
        array_push($_SESSION['tasks'], $task);
    }
}

```

```

if (isset($_POST['delete'])) {
    $index = $_POST['index'];
    if (isset($_SESSION['tasks'][$index])) {
        array_splice($_SESSION['tasks'], $index, 1);
    }
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Simple To-Do List</title>
    <link rel="stylesheet" href="style.css">
</head>
<body>
    <div class="container">
        <h1>Simple To-Do List</h1>
        <form method="post" action="">
            <input type="text" name="task" placeholder="Enter new task" required>
            <button type="submit" name="add">Add Task</button>
        </form>
        <ul>
            <?php foreach ($_SESSION['tasks'] as $index => $task): ?>
                <li>
                    <?php echo htmlspecialchars($task); ?>
                    <form method="post" action="" style="display:inline;">
                        <input type="hidden" name="index" value="<?php echo
$index; ?>">
                        <button type="submit" name="delete">Delete</button>
                    </form>
                </li>
            <?php endforeach; ?>
        </ul>
    </div>
</body>
</html>

```

### CSS Code (style.css)

```

body {
    font-family: Arial, sans-serif;
}

```

```
background-color: #f4f4f4;
margin: 0;
padding: 0;
}

.container {
width: 50%;
margin: 50px auto;
background: #fff;
padding: 20px;
box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

h1 {
text-align: center;
color: #333;
}

form {
display: flex;
justify-content: space-between;
margin-bottom: 20px;
}

input[type="text"] {
width: 75%;
padding: 10px;
border: 1px solid #ddd;
}

button {
padding: 10px 20px;
border: none;
background: #333;
color: #fff;
cursor: pointer;
}

button:hover {
background: #555;
}

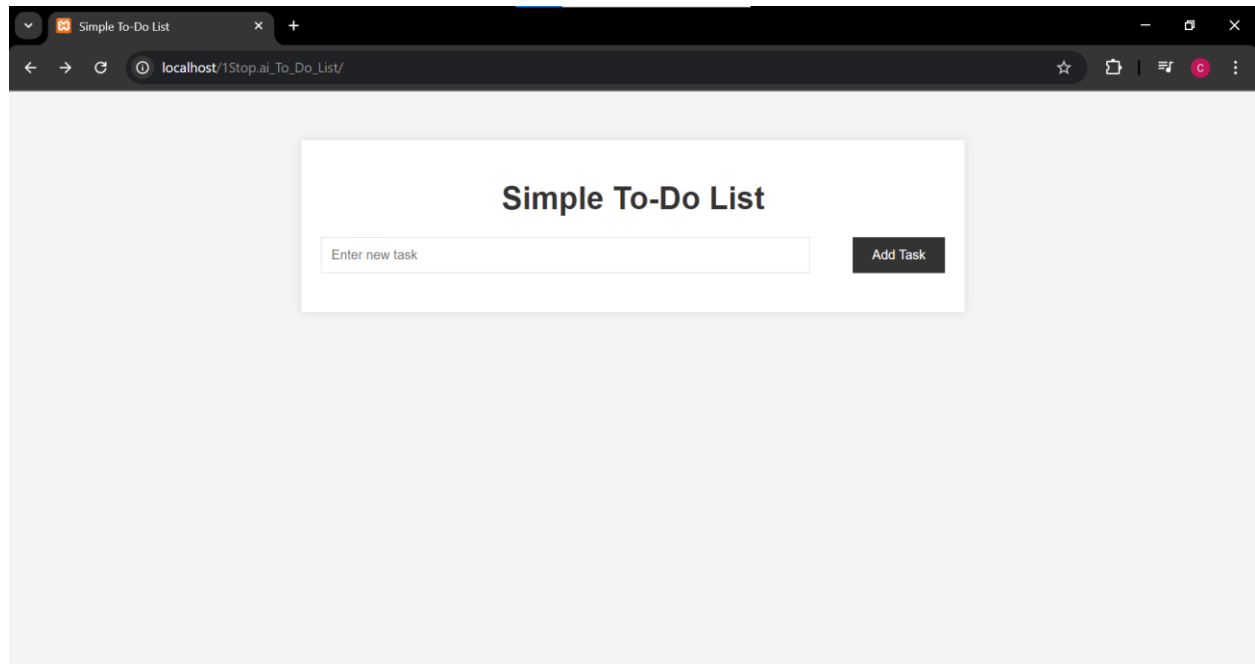
ul {
list-style: none;
padding: 0;
```



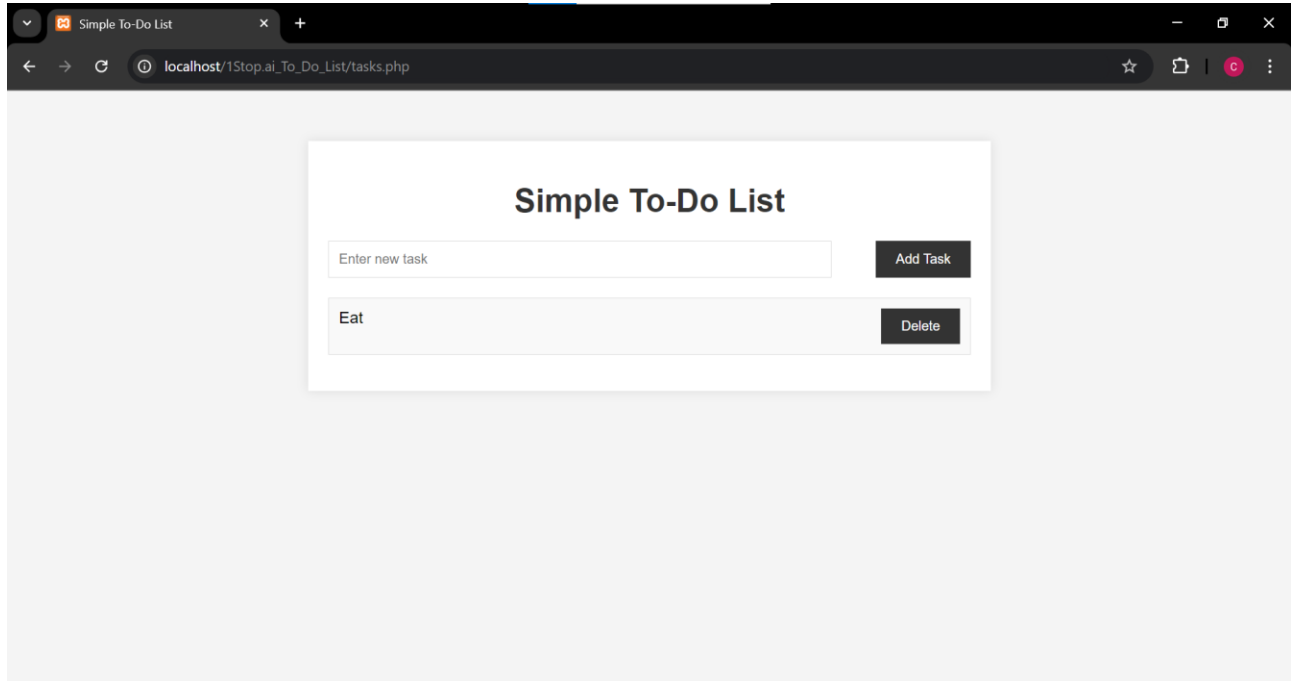
```
}  
  
li {  
  background: #f9f9f9;  
  margin: 5px 0;  
  padding: 10px;  
  border: 1px solid #ddd;  
  display: flex;  
  justify-content: space-between;  
}  
  
li form {  
  margin: 0;  
}
```

## Screenshots of Output

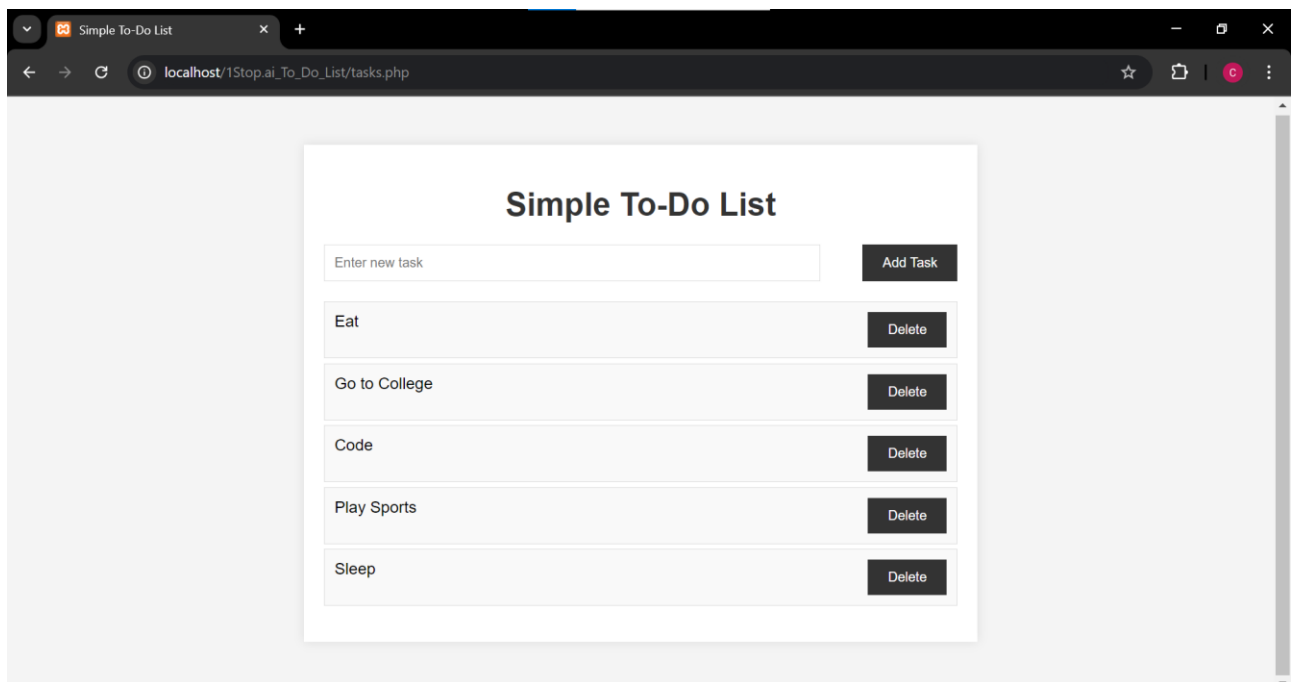
### To-Do List without adding tasks



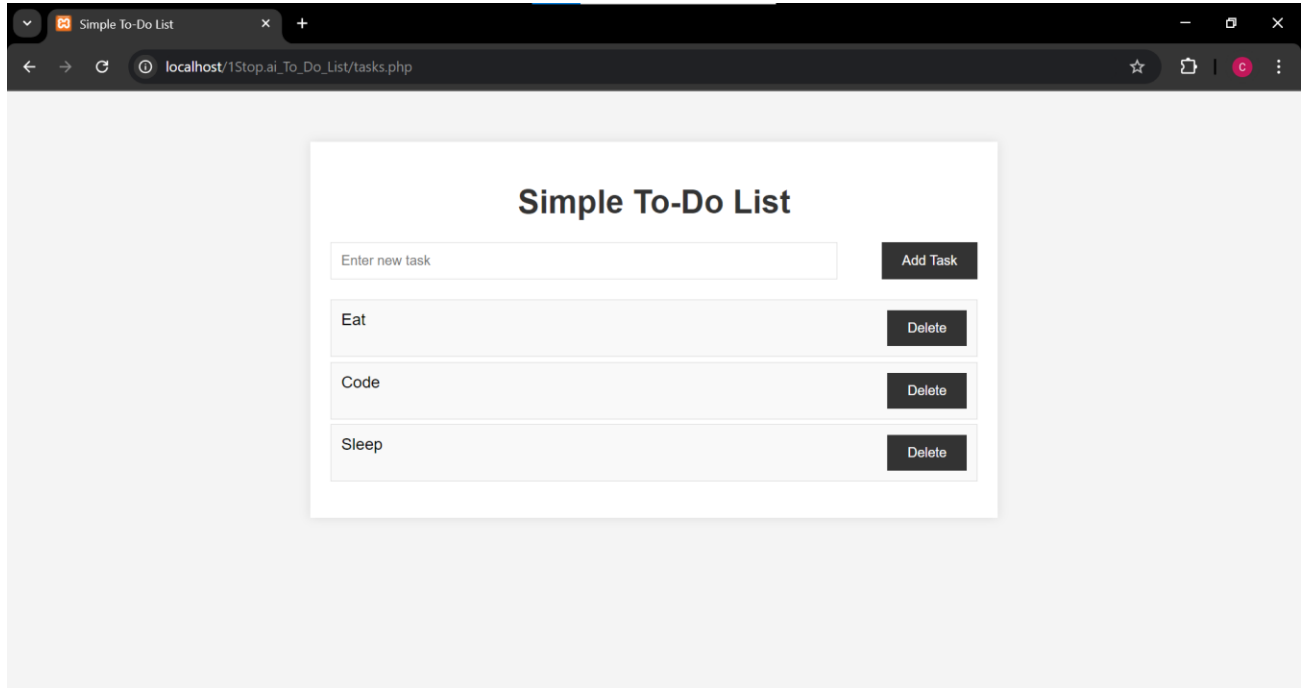
## Adding a single task



## Adding multiple tasks



## Deleting some tasks



## Conclusion

The development of the PHP Simple To-Do List application demonstrates a practical approach to building dynamic web applications using PHP and HTML5. By utilizing PHP sessions for task management, the project simplifies data handling and avoids the complexity of file or database management. This approach not only streamlines the development process but also provides a clear example of how server-side scripting can manage state and user data effectively.

The application's user interface, designed with HTML5 and styled with CSS, ensures a clean and intuitive experience for users. The integration of PHP for server-side logic with HTML5 for the front-end highlights the power of combining these technologies to create functional and interactive web applications. This project provides a solid foundation for understanding core web development concepts, including session management, dynamic content generation, and responsive design.

Additionally, the simplicity of the to-do list application makes it an excellent starting point for beginners learning PHP and web development. It offers hands-on experience with key technologies and concepts, including form handling, session management, and basic user

interface design. Future enhancements could involve integrating database storage for persistent data management or expanding the application with additional features such as user authentication or advanced task categorization.

Overall, this project not only fulfills its primary objective of demonstrating PHP and HTML5 capabilities but also serves as a valuable learning tool for developing more complex web applications. The successful implementation of the to-do list application underscores the effectiveness of using PHP sessions for managing user data and provides a practical example of how to build a dynamic and responsive web application.