WEB DEVELOPMENT BACK-END :

PROJECT 1- PHP SIMPLE TO-DO LIST

Abstract

This project presents a Simple To-Do List Application built using PHP, HTML5, and Bootstrap, designed to help users efficiently manage their daily tasks. The application allows users to add, view, update, and delete tasks while maintaining an intuitive and responsive interface. Bootstrap has been used to enhance the design, ensuring a clean and mobile-friendly user experience. The front end leverages HTML5 and Bootstrap to create an organized and visually appealing layout with responsive tables, modals, and form controls for seamless task management.

PHP handles the backend functionality by interacting with a MySQL database to store tasks persistently. CRUD (Create, Read, Update, Delete) operations are implemented to allow users to manage their tasks efficiently. Form validation and error handling ensure data integrity and prevent errors during task entry and modification. The application also uses session management to provide a secure and personalized experience.

With features like task prioritization and easy task modification, this application simplifies task tracking, making it ideal for personal and professional use. The use of Bootstrap not only enhances the visual appeal but also ensures that the application is fully responsive and adapts to different screen sizes. This project provides a practical demonstration of PHP and MySQL integration while showcasing front-end development with Bootstrap and HTML5.

Objective

The main objective of this Simple To-Do List Application is to create an intuitive and user-friendly platform that helps users organize and manage their tasks effectively. Built using PHP, HTML5, and Bootstrap, this project aims to simplify task management by allowing users to add, edit, delete, and view tasks effortlessly. The application provides a responsive and interactive interface, ensuring that users can manage their daily activities from any device, whether it's a desktop, tablet, or smartphone.

A key goal of this project is to implement CRUD (Create, Read, Update, Delete) operations with PHP and MySQL. This ensures that tasks can be stored, modified, and retrieved efficiently. By using a MySQL database as the backend, task information is persistently stored, preventing any data loss and providing seamless task management. The application allows users to add new tasks, update existing tasks, mark tasks as completed, and delete unnecessary tasks—all while ensuring that data integrity is maintained.

Another important objective is to ensure secure and error-free task operations by implementing input validation and error handling techniques. This helps protect the application from potential security threats such as SQL injection and improper data inputs. The use of Bootstrap ensures that the interface is not only visually appealing but also responsive, providing a smooth user experience regardless of the device being used.

This project also aims to enhance the overall user experience by incorporating a clean and minimalist design. The use of Bootstrap modals and well-structured tables ensures that task information is presented clearly, making it easy for users to interact with the application. Features such as task prioritization and dynamic task listing further enhance the usability of the system.

From a learning perspective, this project serves as an excellent opportunity for aspiring developers to gain hands-on experience with PHP and MySQL for backend functionality and HTML5 and Bootstrap for front-end development. It demonstrates how to build a complete, fully-functional web application by integrating backend logic with an interactive and responsive front-end interface.

Key Objectives:

- To understand and implement CRUD operations using PHP and HTML5.
- To create a user-friendly interface for managing tasks.
- To demonstrate the practical application of PHP in web development.
- To enhance knowledge of server-side scripting and database management.
- To implement a structured approach to web development by following a systematic methodology.
- To ensure data persistence through database connectivity using MySQL.
- To enhance skills in debugging and troubleshooting web applications.
- To gain a practical understanding of integrating PHP with HTML5 for creating dynamic web pages.
- To develop a project that can be expanded upon with more advanced features in the future.

By achieving these objectives, this project provides a robust solution for task management while serving as a valuable learning tool for developers.

Introduction

In today's fast-paced world, staying organized and keeping track of tasks is essential for managing personal and professional responsibilities effectively. A To-Do List Application serves as a simple yet powerful tool that allows users to create, manage, and track their tasks in an organized manner. This project, built using PHP, HTML5, and Bootstrap, focuses on providing a dynamic and responsive web-based task management system that helps users stay on top of their daily goals.

The application uses PHP as the backend scripting language, responsible for handling user requests, processing data, and interacting with a MySQL database to ensure data persistence. CRUD (Create, Read, Update, Delete) operations have been implemented to allow users to manage tasks efficiently. Users can easily add new tasks, mark them as complete, update task details, and delete unnecessary tasks with just a few clicks. Task information is stored securely in the MySQL database, ensuring that all data remains intact even after the application is closed or refreshed.

The front end of the application is designed using HTML5 and Bootstrap, providing a visually appealing and responsive interface. Bootstrap enhances the design by offering a clean and consistent layout that adapts to different screen sizes, ensuring that the application works seamlessly on desktops, tablets, and mobile devices. The use of Bootstrap modals and form controls makes task management more intuitive, allowing users to interact with the application easily.

A notable feature of the application is its ability to prioritize tasks by marking them as complete, helping users focus on pending tasks while maintaining a clear overview of completed ones. Additionally, error handling and input validation have been integrated to prevent incorrect or malicious inputs, ensuring a secure and smooth user experience.

In summary, this To-Do List Application provides a practical solution for task management while demonstrating the seamless integration of PHP and MySQL with a responsive front-end interface built using HTML5 and Bootstrap.

Methodology

The methodology followed for developing the Simple To-Do List Application involves a systematic and structured approach to ensure that all aspects of the project are covered effectively. The process consists of multiple phases, including requirement analysis, design, development, testing, deployment, and future maintenance. Each phase was carried out with a focus on delivering a high-quality, functional, and user-friendly application that meets the desired objectives.

1. Requirement Analysis

The initial phase involved analyzing the project requirements and defining the core functionalities that needed to be implemented. The objective was to build a simple, dynamic, and interactive task management system where users could:

- Add new tasks to the list.
- View and manage existing tasks.
- Mark tasks as completed.
- Delete tasks that are no longer needed.

During this phase, the scope and limitations of the project were clearly defined. Since task editing functionality was not part of the initial requirements, it was excluded from the project. Instead, the application focuses on allowing users to remove tasks and re-add them if necessary.

2. Designing

The design phase involved creating a clean and intuitive user interface (UI) using HTML5 and Bootstrap. The goal was to create a responsive and visually appealing layout that provides easy access to all functionalities. Key design considerations included:

- Designing a simple, user-friendly interface that allows users to manage tasks effortlessly.
- Structuring the application layout to enhance accessibility and ensure smooth navigation.
- Ensuring that the interface adapts to various devices and screen sizes by incorporating Bootstrap's responsive design features.

- Using Bootstrap's pre-built UI components such as buttons, forms, and modals to maintain consistency and improve the overall look and feel of the application.

3. Development

The development phase involved implementing the server-side logic using PHP to handle user requests and manage task data. MySQL was used as the backend database to store task information securely and persistently. CRUD (Create, Read, Update, Delete) operations were implemented to allow seamless task management. Key development activities included:

- Setting up a MySQL database to store task data with relevant fields such as task name, status, and task ID.
- Writing PHP code to handle task creation, retrieval, marking as done, and deletion.
- Developing a secure and efficient connection between the application and the database to ensure data persistence.
- Implementing dynamic task listing to fetch and display tasks in real time.
- Ensuring that the application responds accurately to user actions, such as adding and deleting tasks.

4. Testing

The testing phase was crucial to ensure that all functionalities were working as expected. Comprehensive testing was conducted to identify and resolve potential issues. The testing process included:

- Functional Testing: Verifying that all task-related operations (adding, viewing, marking as done, and deleting tasks) performed correctly.
- Database Testing: Ensuring that data was stored, retrieved, and deleted accurately from the MySQL database.
- Cross-Browser Compatibility Testing: Testing the application on different browsers such as Chrome, Firefox, and Edge to ensure consistent behavior.
- Responsiveness Testing: Checking the application's performance and layout on different devices to ensure mobile-friendliness and responsiveness.

- Error Handling and Security Testing: Ensuring that the application could handle unexpected inputs and errors gracefully without affecting the overall functionality.

5. Deployment

Once the development and testing phases were complete, the application was deployed using XAMPP, a local server environment. This enabled smooth execution of PHP scripts and database connectivity, providing a reliable demonstration of the application's functionalities. Key deployment activities included:

- Configuring the local server environment to host the application files.
- Ensuring that the database was correctly connected and accessible.
- Validating that all features worked correctly in the deployed environment.

6. Maintenance and Future Enhancements

The final phase focuses on maintaining the application and identifying potential enhancements to improve functionality and user experience. Future enhancements that can be considered include:

- Task Prioritization: Adding the ability to prioritize tasks based on urgency or importance.
- Task Categorization: Allowing users to categorize tasks for better organization.
- Due Date Management: Enabling users to set deadlines and receive notifications for upcoming tasks.
- Improved UI/UX Design: Continuously refining the interface to improve ease of use and visual appeal.
- User Authentication: Implementing a multi-user environment with authentication and role-based task management for increased security and personalized task lists.

By following this structured methodology, the project successfully delivered a fully functional and responsive To-Do List application that meets user needs while providing a strong foundation for future enhancements.

Code Implementation

Directory structure: TODO/

```
├── Folder.png     # An image file
├── index.php      # The main PHP file for the to-do application
└── todo.sql       # SQL database file (contains table structure and data)
```

index.php code:

```
<?php
    $server = 'localhost';
    $username = 'root';
    $password = '';
    $database = 'todo_master';

    $conn = mysqli_connect($server,$username,$password,$database);

    if($conn->connect_errno)
    {
        die('Connection to MySQL failed : ' .$conn->connect_error);

    }

    //CREATING A TODO ITEM
    if(isset($_POST['add'])){
        $item = $_POST['item'];
```

```
    if(!empty($item)){

        $query = "INSERT INTO todo (name) VALUES ('$item')";

        if(mysqli_query($conn,$query)){
            echo '
            <center>
            <div class="alert alert-success" role="alert">
                Item Added Successfully!!
            </div>
            </center>
            ';
        }
        else{
            echo mysqli_error($conn);
        }
    }
}

//Checking if action parameter is present
if(isset($_GET['action'])){
    $itemId = $_GET['item'];
```

```
if($_GET['action']== 'done')
{
   $query = "UPDATE todo SET status = 1 WHERE id = '$itemId'";
   if(mysqli_query($conn,$query)){
      echo '
      <center>
      <div class="alert alert-info" role="alert">
         Item Marked as Done!
      </div>
      </center>
      ';
   }
   else
   {
      echo mysqli_error($conn);
   }
}
elseif($_GET['action'] == 'delete')
{
   $query = " DELETE FROM todo WHERE id= '$itemId'";
      if(mysqli_query($conn,$query)){
         echo '
         <center>
         <div class="alert alert-danger" role="alert">
```

```
                Item Deleted successfully!
                </div>
                </center>
                ';
            }
            else
            {
                echo mysqli_error($conn);
            }
        }
    }
?>

<!DOCTYPE html>
<html>
    <head>
        <title>Todo List Application</title>
        <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH" crossorigin="anonymous">
        <style>
            .done{
                text-decoration: line-through;
            }
```

```
    </style>


</head>
<body>
    <main>


        <div class="container pt-5">
            <div class="row">
            <div class="col-sm-12 col-md-3"></div>
            <div class="col-sm-12 col-md-6">
            <div class="card">
                <div class="card-holder">
                    <p> Todo List</p>
                </div>
                <div class="card-body">
                    <form method="post" action="<?=$_SERVER['PHP_SELF']?>">
                        <div class="mb-3">
                            <input type="text" class="form-control" name="item" placeholder="Add
a Todo Item">
                        </div>
                        <input type="submit" class="btn btn-dark" name="add" value="Add Item">
                        </form>
                         <div class="mt-5 mb-5">


                            <?php
```

```
$query = "SELECT * FROM todo";
$result =  mysqli_query($conn,$query);
if($result->num_rows>0)
{
   $i=1;
   while($row = $result->fetch_assoc()){
      $done = $row['status'] == 1 ? "done" : " ";

   echo '
   <div class="row mt-4">
   <div class="col-sm-12 col-md-1"><h5>'.$i.'</h5></div>
   <div class="col-sm-12 col-md-6"><h5
class="'.$done.'">'.$row['name'].'</h5></div>
   <div class="col-sm-12 col-md-5">
      <a href="?action=done&item='.$row['id'].'" class="btn btn-
outline-dark">Mark as Done</a>
      <a href="?action=delete&item='.$row['id'].'" class="btn btn-
outline-danger">Delete</a>
   </div>
</div>
';
   $i++;
}
}
else
```

```
                    {
                        echo '
                        <center>
                        <img src="Folder.png" width="50px" alt="Empty
list"><br><span>Your list is Empty</span>
                        </center>
                        ';
                    }
                ?>

                    </div>
                </div>
            </div>
        </div>
    </div>
</main>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.11.8/dist/umd/popper.min.js"
integrity="sha384-
I7E8VVD/ismYTF4hNIPjVp/Zjvgyol6VFvRkX/vR+Vc4jQkC+hVqc2pM8ODewa9r"
crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"
crossorigin="anonymous"></script>
```

```
    <script>
        $(document).ready(function(){
            $(".alert").fadeTo(5000,500).slideUp(500,function(){
                $('.alert').slideUp(500);
            });
        })
    </script>
</body>
</html>
```

Web page Explanation

1. **Application Title:**

- The top section displays the title "**To-Do List**," indicating the purpose of the application.
- The title is prominently displayed in a centered position, giving users clarity about the functionality of the application.

2. **Input Field for Adding Tasks:**

- Below the title, there is a text input field where users can type the name of a new task they want to add to their list.
- This input field is accompanied by a button labeled **"Add Item"**, which submits the task to the database when clicked.

3. **Empty Task List Section:**

- Since no tasks have been added yet, the task list section is currently empty.
- A default image (a folder icon) with a message like **"Your list is Empty"** is displayed, indicating that no tasks are available.
- This visual cue provides clear feedback to users and encourages them to add their first task.

4. **Responsive Design:**

- The UI is built using Bootstrap, ensuring that the layout adjusts properly to different screen sizes.
- The central positioning of the task input field and the empty list message ensures that users' attention is directed towards adding new tasks.

5. **Bootstrap Styling and Alerts:**

- The button and input field use Bootstrap styling, giving them a modern and professional appearance.
- Alerts (though not visible yet) are programmed to display confirmation messages upon successfully adding or deleting tasks.

Screenshots:

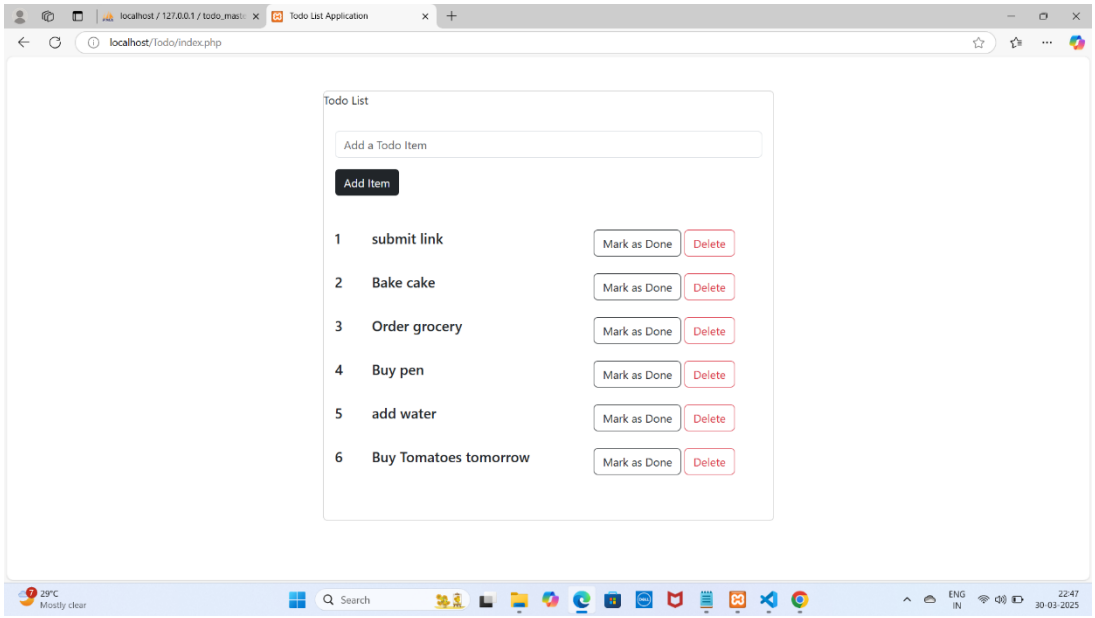

Figure 1 : PHP-Based To-Do List Web Application

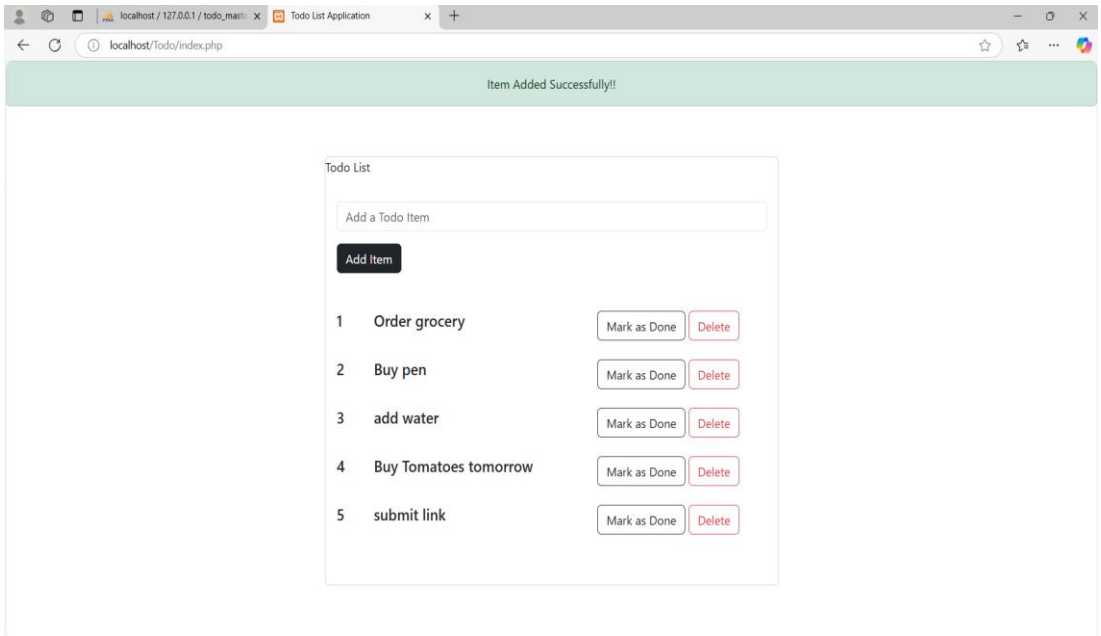

Figure 2 : submit link item added to the Todo list

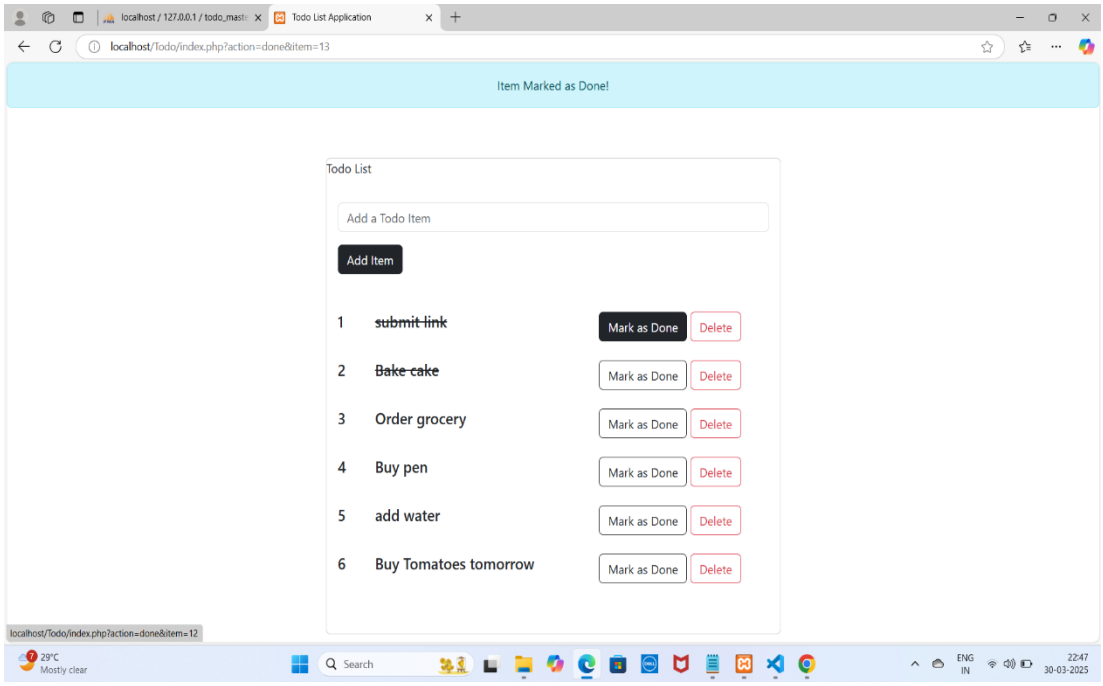

Figure 3 : submit link item is marked as Done

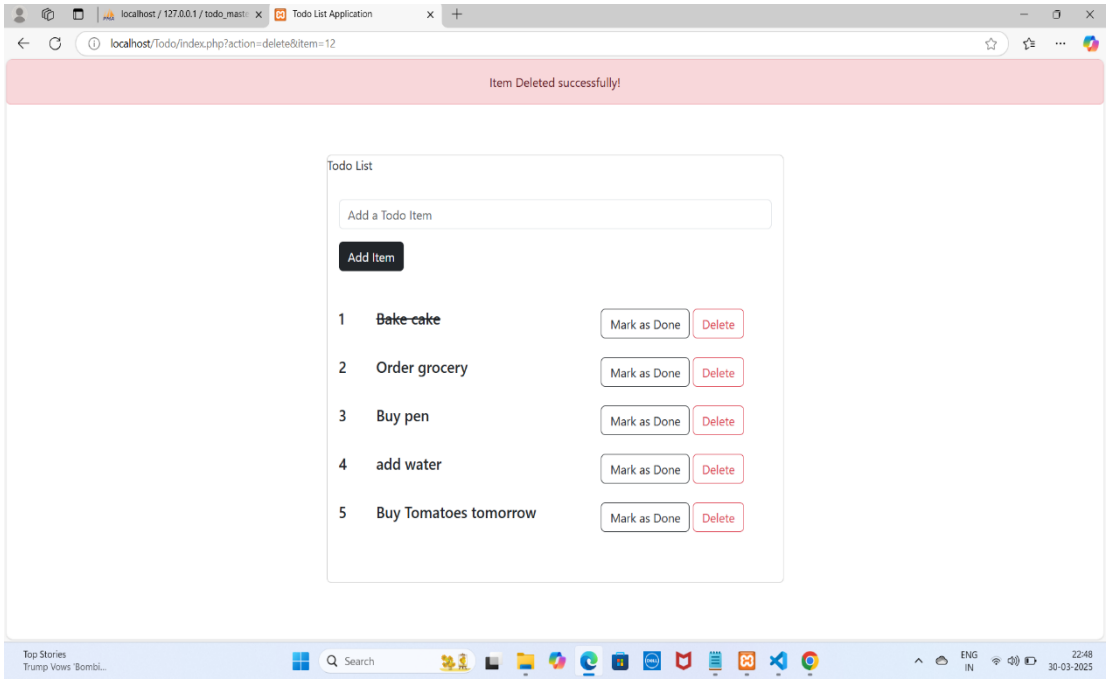

Figure 4 : Submit link item deleted from Todo list

Todo.sql:

```
-- phpMyAdmin SQL Dump

-- version 5.2.1

-- https://www.phpmyadmin.net/

--

-- Host: 127.0.0.1

-- Generation Time: Mar 30, 2025 at 06:18 PM

-- Server version: 10.4.32-MariaDB

-- PHP Version: 8.2.12


SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";

START TRANSACTION;

SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT
*/;

/*!40101 SET
@OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;

/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION
*/;

/*!40101 SET NAMES utf8mb4 */;


--

-- Database: `todo_master`

--
```

```
-- --------------------------------------------------------

--
-- Table structure for table `todo`
--

CREATE TABLE `todo` (
  `id` int(11) NOT NULL,
  `name` varchar(255) NOT NULL,
  `status` int(11) NOT NULL DEFAULT 0,
  `created_at` timestamp NOT NULL DEFAULT current_timestamp(),
  `updated_at` timestamp NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;

--
-- Dumping data for table `todo`
--

INSERT INTO `todo` (`id`, `name`, `status`, `created_at`, `updated_at`) VALUES
(12, 'submit link', 0, '2025-03-30 16:16:23', '2025-03-30 16:16:23'),
(13, 'Bake cake', 0, '2025-03-30 16:16:30', '2025-03-30 16:16:30'),
(14, 'Order grocery', 0, '2025-03-30 16:16:44', '2025-03-30 16:16:44'),
(15, 'Buy pen', 0, '2025-03-30 16:16:47', '2025-03-30 16:16:47'),
```

```
(16, 'add water', 0, '2025-03-30 16:16:50', '2025-03-30 16:16:50'),
(17, 'Buy Tomatoes tomorrow', 0, '2025-03-30 16:16:54', '2025-03-30 16:16:54');

--
-- Indexes for dumped tables
--

--
-- Indexes for table `todo`
--
ALTER TABLE `todo`
  ADD PRIMARY KEY (`id`);

--
-- AUTO_INCREMENT for dumped tables
--

--
-- AUTO_INCREMENT for table `todo`
--
ALTER TABLE `todo`
  MODIFY `id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=18;
COMMIT;
```

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;

/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;

/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

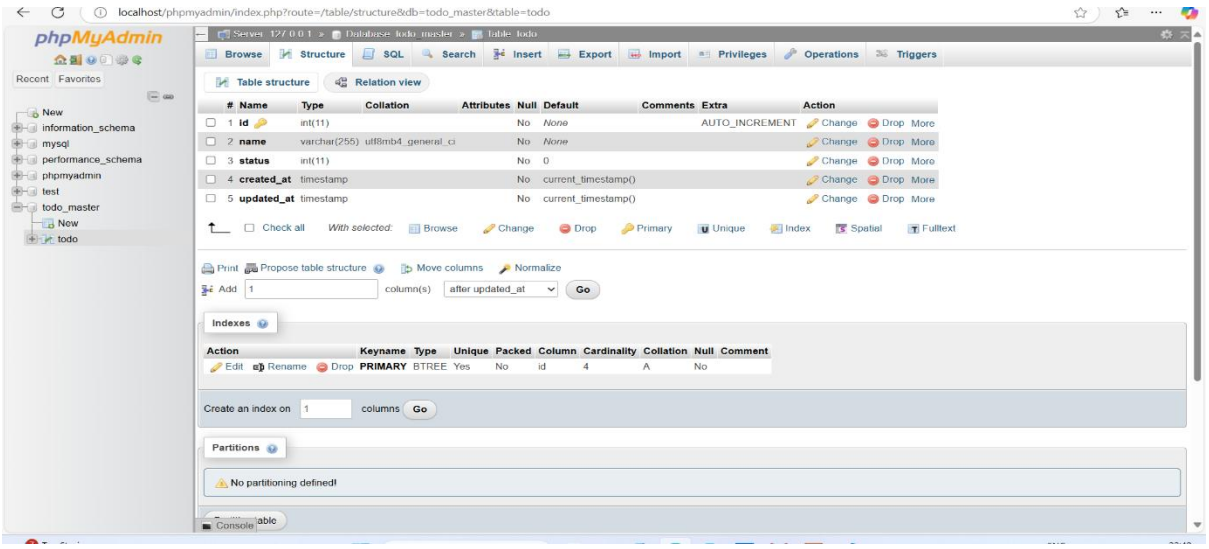

Figure 5 : Table structure of Todo

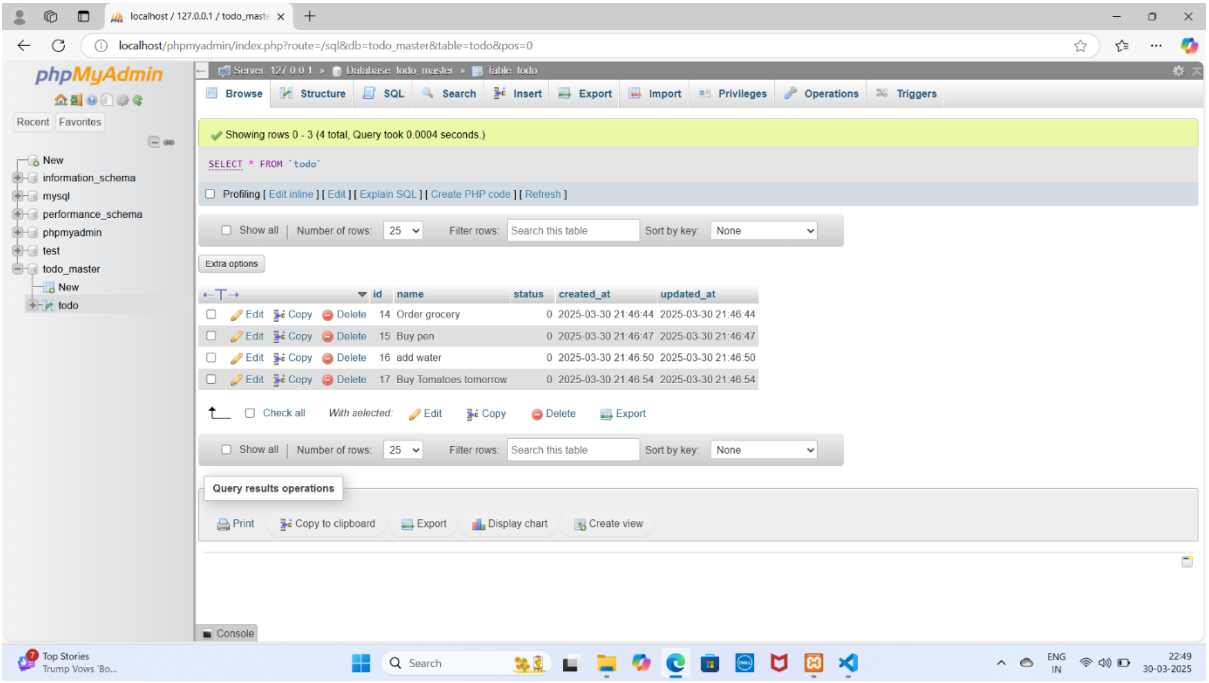

Figure 6 : Items list in the database server

Conclusion

The development of the Simple To-Do List Application using PHP, HTML5, Bootstrap, and MySQL successfully demonstrated the creation of a dynamic and interactive web application that helps users manage their daily tasks efficiently. The project provided an excellent opportunity to implement core web development concepts, including server-side scripting with PHP, database management with MySQL, and front-end design using HTML5 and Bootstrap. By focusing on essential task management functionalities such as adding, viewing, marking tasks as done, and deleting tasks, the application effectively addressed the primary requirements of a task management system.

One of the key accomplishments of this project was the successful implementation of CRUD (Create, Read, Update, Delete) operations using PHP and MySQL. This ensured that tasks were managed securely and persistently, allowing users to maintain their task lists without any data loss. The integration of Bootstrap enhanced the application's user interface by providing a clean, responsive, and visually appealing layout. This ensured that the application worked seamlessly across various devices, including desktops, tablets, and smartphones.

The project also emphasized the importance of form validation and error handling to prevent incorrect data inputs and improve the overall security of the application. By validating user inputs and sanitizing data before interacting with the database, potential vulnerabilities such as SQL injection and form manipulation were mitigated, ensuring a safe and reliable task management experience for the users.

During the testing phase, the application was rigorously tested to identify and resolve any issues. The application was tested for functional correctness, database integrity, responsiveness, and cross-browser compatibility to ensure a consistent and smooth user experience across different platforms. These tests validated that all features were working as intended, and data was being processed and stored accurately.

While the application meets its core objectives, it also lays the groundwork for future enhancements. Potential improvements include task prioritization, task categorization, due date management, and user authentication to make the application more robust and feature-rich. Additionally, enhancing the UI/UX design further can contribute to a more engaging and intuitive user experience.

From a learning perspective, this project provided invaluable hands-on experience in building a complete web application by integrating backend functionality with a dynamic and responsive front-end interface. It reinforced a solid understanding of PHP for server-side scripting, MySQL for database management, and Bootstrap for enhancing UI components.

In conclusion, this project successfully achieved its goal of creating a fully functional and user-friendly To-Do List application. It not only fulfilled the intended task management requirements but also provided a strong foundation for future growth and improvements. This project serves as a stepping stone for exploring web development concepts and further enhancing the application to meet the evolving needs of users.