

# TASK MANAGER

---

BY- Anoop Kumar Yadav

## ABSTRACT

The Task Manager project is an innovative web-based application crafted using Laravel, PHP, and MySQL. It's designed to streamline task management for users, offering a user-friendly platform to efficiently organize their tasks. Thanks to Laravel's MVC architecture, the project maintains a clear separation of concerns, ensuring ease of maintenance and scalability as the application evolves.

By integrating essential services such as Postman for API testing and Mailtrap for email notifications, the project delivers a comprehensive solution for managing tasks effectively. This strategic integration enhances the project's functionality, providing users with a seamless experience from task creation to completion.

In essence, the Task Manager project represents a modern approach to task management, combining robust technology with user-centric design principles to empower individuals and teams to achieve their goals with greater efficiency and clarity.

## **OBJECTIVE**

The Task Manager project's main goal is to simplify task management by offering users a comprehensive application. It strives to achieve this by implementing CRUD operations (Create, Read, Update, Delete) for tasks and integrating real-time email notifications through Mailtrap.

Using Laravel's powerful features, the project aims to provide a dependable, scalable, and intuitive task management system that enhances user productivity and organization.

# INTRODUCTION

Amidst the fast-paced world of project management, keeping track of tasks efficiently is key to success. That's where our Task Manager comes in. It's like your personal helper for managing tasks, but in a digital form!

Our Task Manager is built using a special tool called Laravel PHP, which helps make things run smoothly behind the scenes. It's like the engine of a car, making sure everything works as it should. We also store all your task information in a safe place called MySQL, so you can access it whenever you need to.

We've made sure our Task Manager is super easy to use. You can create new tasks, update existing ones, and keep an eye on their progress, all from a simple and easy-to-understand website. Think of it like writing notes on a digital sticky pad!

But that's not all! We've also added some cool features like sending you emails when there's an update on your tasks. It's like having a little bird whispering in your ear whenever something important happens.

So, join us on this journey of making task management a breeze. With our Task Manager, you'll be organized, efficient, and always on top of your game!

# METHODOLOGY

## TASK CREATION:

- **User Input:** Users input task details including description, owner name, owner email, and estimated time of completion (ETA) through the frontend interface.
- **Request Handling:** The frontend sends a POST request to the `/api/task` endpoint, which is routed to the `create` method in the `APITasksController`.
- **Data Validation:** The controller validates the incoming request data to ensure correctness and completeness.
- **Database Interaction:** Upon successful validation, a new task record is created in the MySQL database using the `TaskManager` model.
- **Email Notification:** A background job (`SendEmailJob`) is dispatched to send a notification email to the task owner's email address using Mailtrap.
- **Response:** The controller sends a success message or error response back to the frontend based on the outcome of the task creation process.

# METHODOLOGY

## TASK UPDATE :

- **User Request:** Users initiate task updates by editing task details and submitting changes through the frontend interface.
- **Request Routing:** The frontend sends a PUT request to the `/api/task/{id}` endpoint, where `{id}` represents the unique identifier of the task to be updated.
- **Update Handling:** The request is routed to the `update` method in the `APITasksController`, which fetches the corresponding task record from the database.
- **Data Modification:** The controller updates the task attributes with the new values provided by the user and saves the changes to the database.
- **Response:** Upon successful update, the controller sends a success message to the frontend; otherwise, it returns an error response.

# METHODOLOGY

## TASK DELETION:

- **User Trigger:** Users initiate task deletion by selecting the delete option for a specific task on the frontend interface.
- **Request Processing:** The frontend sends a DELETE request to the `‘/api/task/{id}’` endpoint, where `‘{id}’` identifies the task to be deleted..
- **Deletion Logic:** The request is handled by the `‘delete’` method in the `‘APITasksController’`, which retrieves the task record associated with the provided `‘{id}’` and deletes it from the database
- **Data Modification :** The controller updates the task attributes with the new values provided by the user and saves the changes to the database.
- **Response:** Upon successful deletion, the controller sends a success message back to the frontend; otherwise, it returns an error response..

# METHODOLOGY

## REAL-TIME NOTIFICATIONS:

- **Event Trigger:** Task creation and completion events trigger real-time email notifications to the task owner.
- **Queueing:** When a task is created or marked as done, a background job '[SendEmailJob](#)' is queued to handle email notification tasks asynchronously.
- **Email Construction:** The job constructs an email message using the '[TaskMail](#)' Mailable, passing relevant task details to personalize the notification.
- **Dispatching:** The job dispatches the email to the task owner's email address using Mailtrap's SMTP server.
- **Delivery Confirmation:** Upon successful email dispatch, the job completes execution, providing feedback to the system.



# METHODOLOGY

## API TESTING:

- **Postman Usage:** Postman is utilized to test the functionality and reliability of the API endpoints..
- **Test Cases:** Test cases are designed to cover CRUD operations, input validation, error handling, and response formats.
- **Endpoint Verification:** Each endpoint is tested individually to ensure it behaves as expected under various scenarios.
- **Dispatching:** The job dispatches the email to the task owner's email address using Mailtrap's SMTP server.
- **Feedback Analysis:** Test results and feedback are analyzed to identify and address any issues or improvements needed in the API implementation..










# SCREENSHOTS

Add Tasks

Serial No.	Task Description	Task Owner	Task ETA	Action
------------	------------------	------------	----------	--------

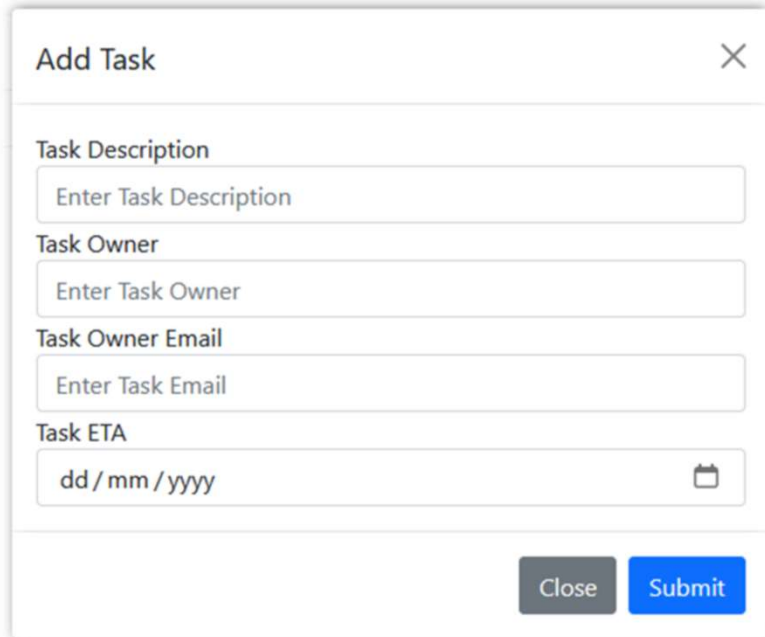
TODO LIST APPLICATION ( empty )

Add Tasks

Serial No.	Task Description	Task Owner	Task ETA	Action
1	UX Patch 1	John	2024-03-25	  
2	UX Patch 2	Sweta	2024-03-26	  
3	UX Patch 3	Rohan	2024-03-26	  

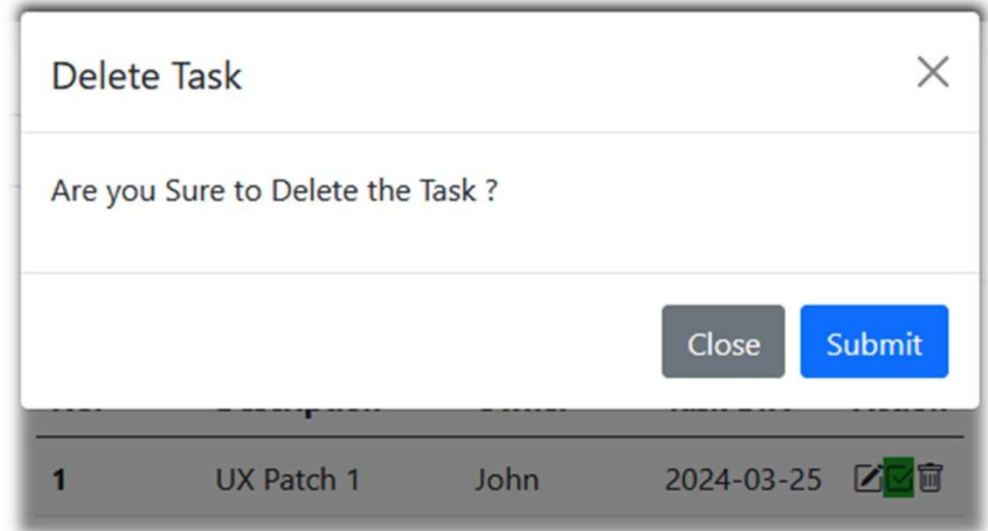
TODO LIST APPLICATION  
( have element )

# SCREENSHOTS





A screenshot of a 'Add Task' pop-up form. The form has a title bar with 'Add Task' and a close button (X). It contains four input fields: 'Task Description' with placeholder text 'Enter Task Description', 'Task Owner' with placeholder text 'Enter Task Owner', 'Task Owner Email' with placeholder text 'Enter Task Email', and 'Task ETA' with placeholder text 'dd / mm / yyyy' and a calendar icon. At the bottom right, there are two buttons: 'Close' (grey) and 'Submit' (blue).

Add Task Pop-Up  
(on click Add Task Button)

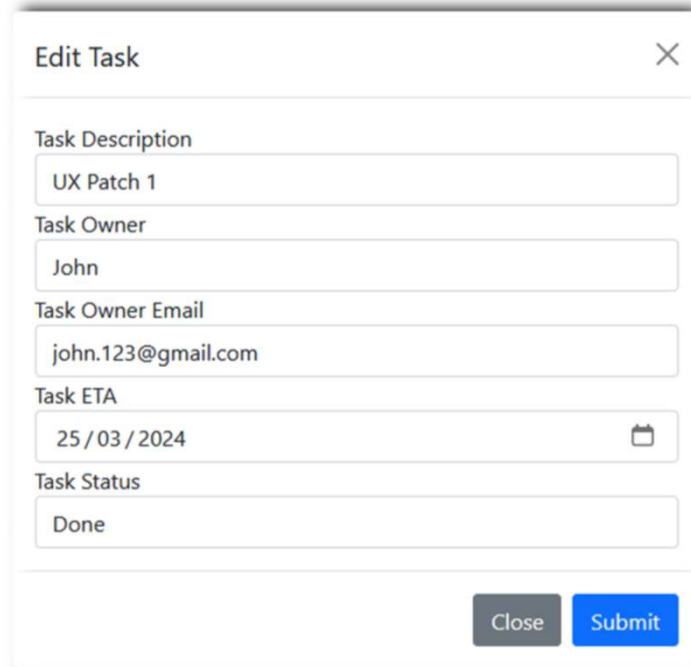


A screenshot of a 'Delete Task' confirmation pop-up. The form has a title bar with 'Delete Task' and a close button (X). The main text asks 'Are you Sure to Delete the Task ?'. At the bottom right, there are two buttons: 'Close' (grey) and 'Submit' (blue). Below the pop-up, a table is visible with the following data:

1	UX Patch 1	John	2024-03-25	 
---	------------	------	------------	---

Delete Confirmation Pop-Up  
(on click delete icon)

# SCREENSHOTS

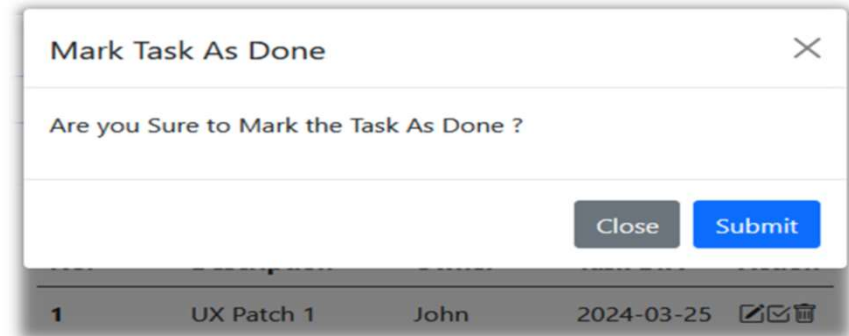


The 'Edit Task' pop-up form contains the following fields:

- Task Description:** UX Patch 1
- Task Owner:** John
- Task Owner Email:** john.123@gmail.com
- Task ETA:** 25/03/2024
- Task Status:** Done

Buttons: Close, Submit

Edit Task Pop-Up  
(on click Edit icon)



The 'Mark Task As Done' pop-up dialog displays the confirmation message: "Are you Sure to Mark the Task As Done ?". It includes "Close" and "Submit" buttons.

Serial No.	Task Description	Task Owner	Task ETA	Action
1	UX Patch 1	John	2024-03-25	  
2	UX Patch 2	Sweta	2024-03-26	  
3	UX Patch 3	Rohan	2024-03-26	  

Mark Task Done Pop-Up with Result  
(Green Color Indicates Done)

# SCREENSHOTS

Add Task

Task Description

UX Patch 4

Task Owner

Kashish

Task Owner Email

kashish123@gmail.com

Task ETA

27 / 03 / 2024

Close

Submit

Task Creation Notification	
to: <kashish123@gmail.com>	a minute ago
Task Creation Notification	
to: <rohan123@gmail.com>	8 minutes ago
Task Creation Notification	
to: <john.123@gmail.com>	8 minutes ago
Task Creation Notification	
to: <rohan123@gmail.com>	9 minutes ago
Task Creation Notification	
to: <sweta123@gmail.com>	10 minutes ago
Task Creation Notification	
to: <john.123@gmail.com>	10 minutes ago

From: Laravel <hello@example.com>  
To: <kashish123@gmail.com>

[Show Headers](#)

**HTML**

HTML Source

Text

Raw

Dear Kashish,

The Task "UX Patch 4" has been added for you  
Kindly complete it by 2024-03-27.

Thank You

Illustration Of Task Assignment On Email Notification  
(Mailtrap is used )

## CONCLUSION:

The Task Manager project showcases the effective utilization of Laravel, PHP, and MySQL to develop a robust task management system. By adhering to best practices in software development and leveraging external services like Mailtrap and Postman, the project delivers a comprehensive solution for users to organize, track, and manage their tasks efficiently.

With its scalable architecture, intuitive interface, and real-time notification capabilities, the Task Manager project sets a new standard in task management applications, catering to the needs of individuals and organizations alike.

