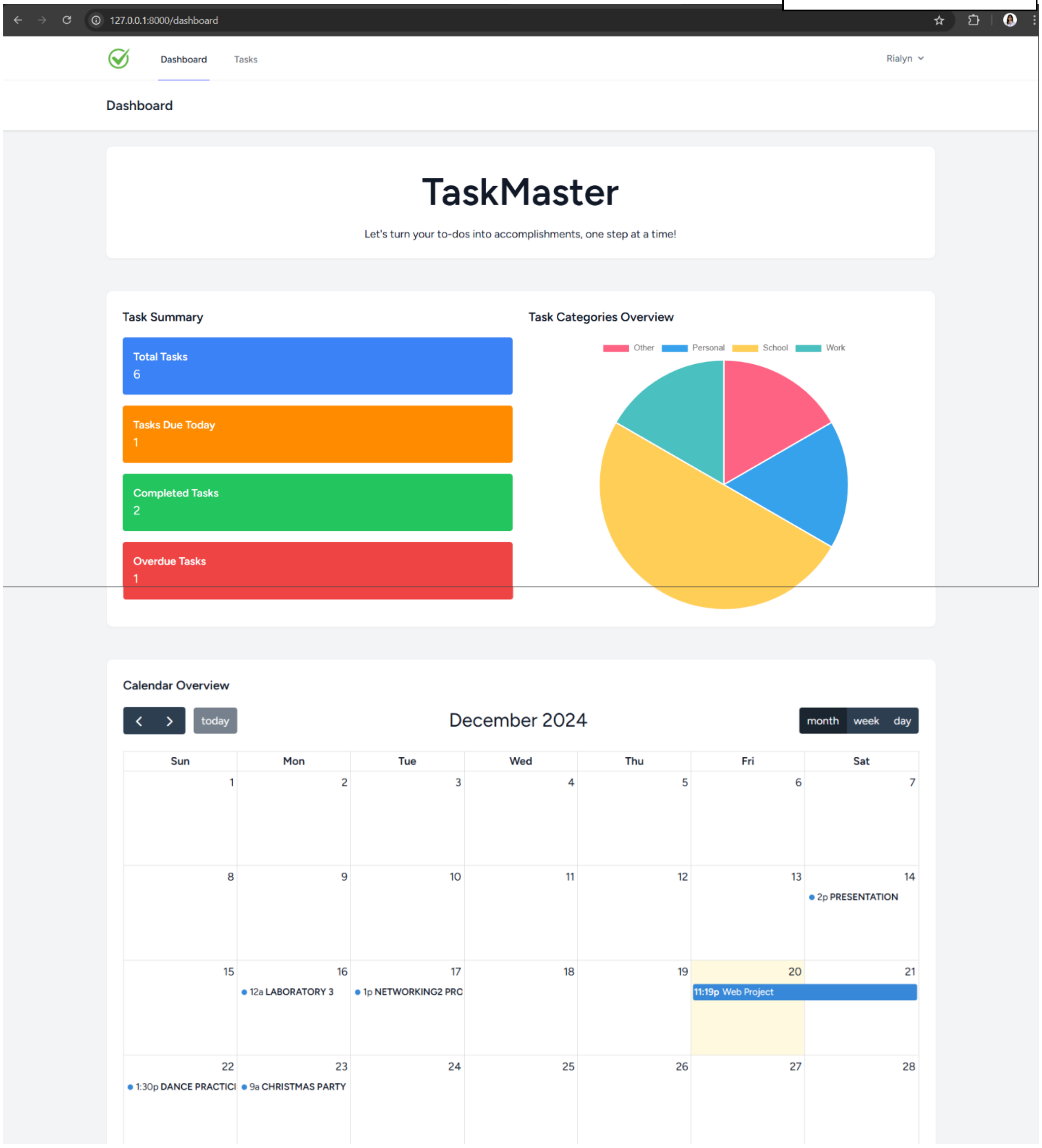
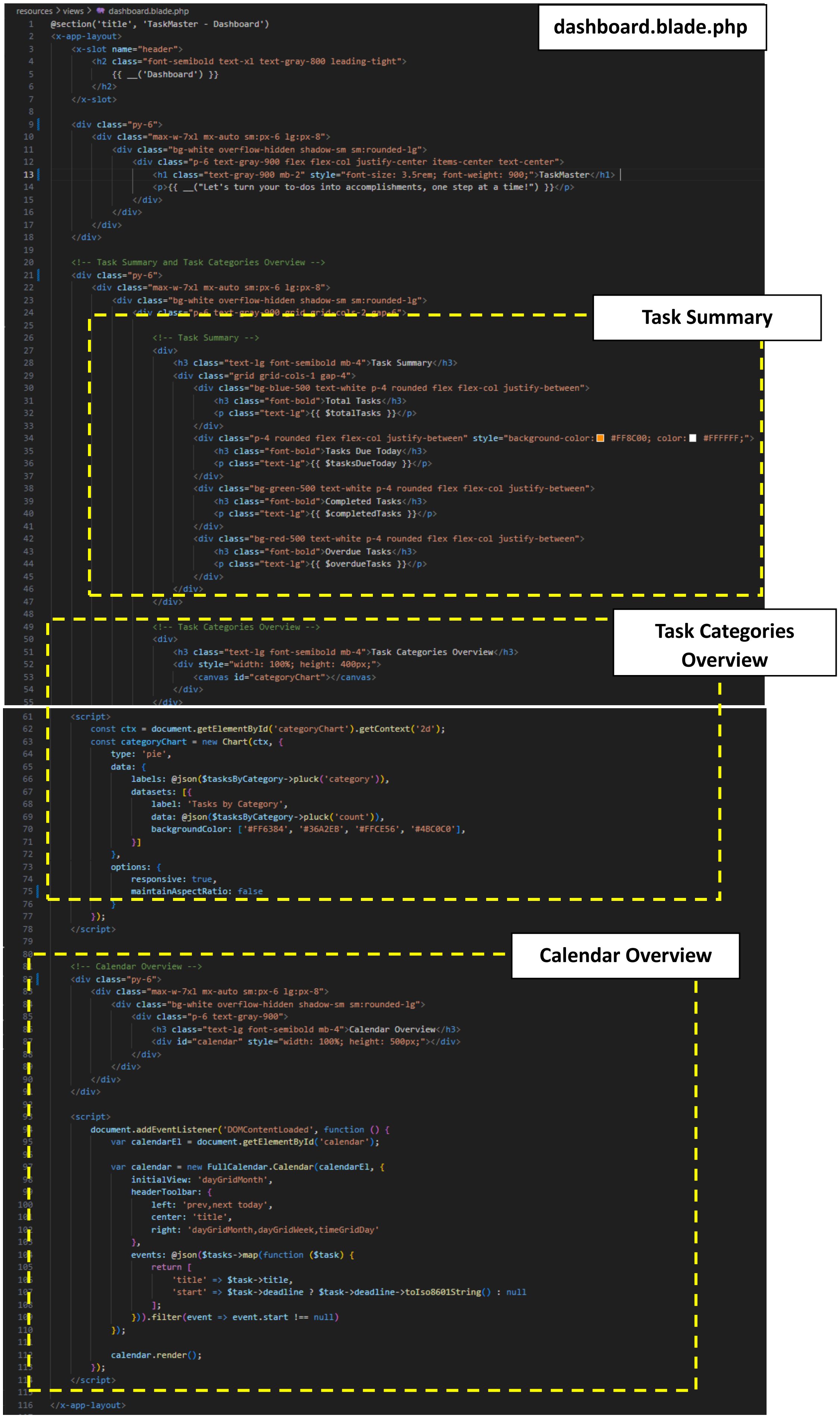
RIALYN CAMPOSANO **BSIT-3C** **WEB DEVELOPMENT**

DOCUMENTATION: Laboratory 3: Populating From a Database

1. Rendered Pages and Accompanying Code

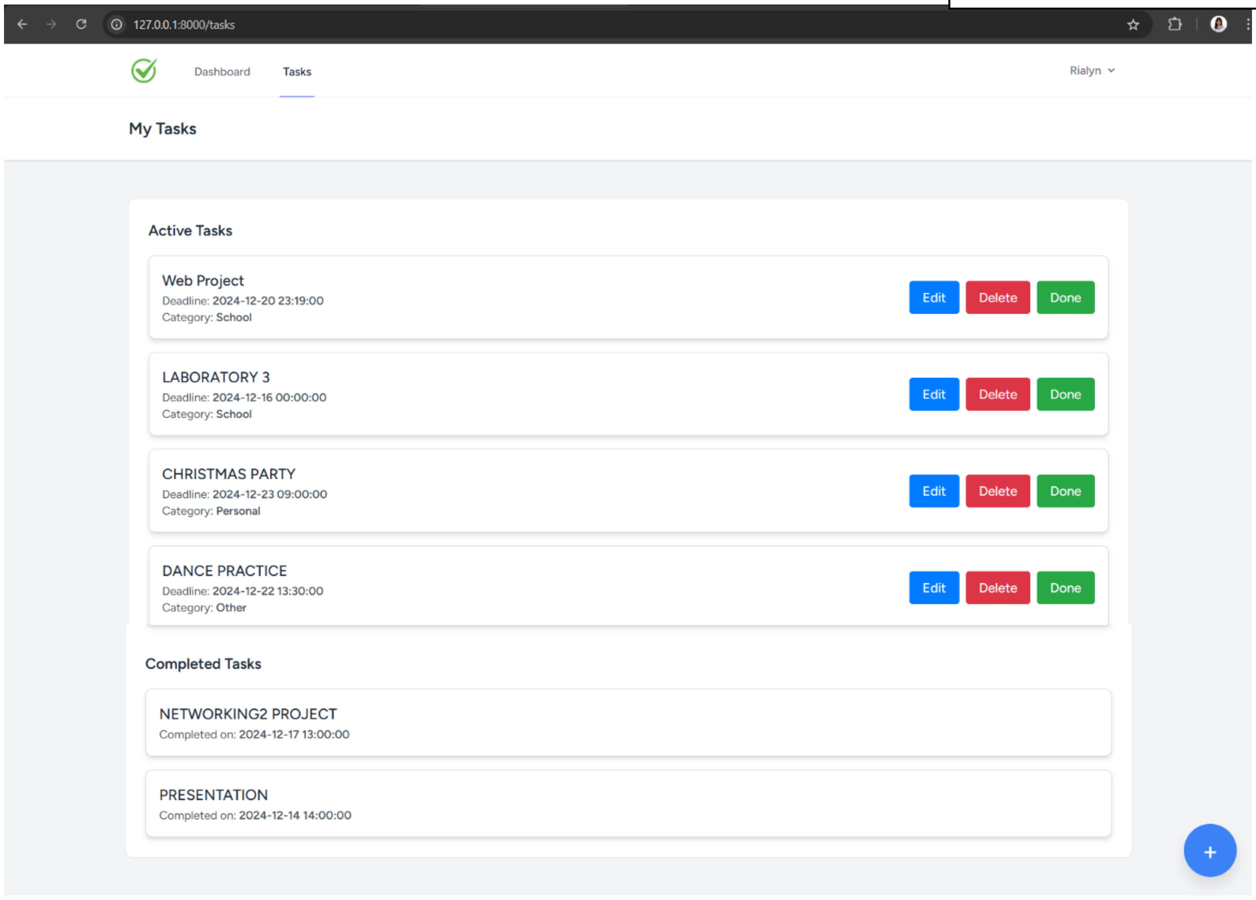
dashboard.blade.php





```
@section('title', 'TaskMaster - Dashboard')
<x-app-layout>
    <x-slot name="header">
        <h2 class="font-semibold text-xl text-gray-800 leading-tight">
            {{ __('Dashboard') }}
        </h2>
    </x-slot>

    <div class="py-6">
        <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
            <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6 text-gray-900 flex flex-col justify-center items-center text-center">
                    <h1 class="text-gray-900 mb-2" style="font-size: 3.5rem; font-weight: 900;">TaskMaster</h1>
                    <p>{{ __("Let's turn your to-dos into accomplishments, one step at a time!") }}</p>
                </div>
            </div>
        </div>
    </div>

    <!-- Task Summary and Task Categories Overview -->
    <div class="py-6">
        <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
            <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6 text-gray-900 grid grid-cols-2 gap-6">

                    <!-- Task Summary -->
                    <div>
                        <h3 class="text-lg font-semibold mb-4">Task Summary</h3>
                        <div class="grid grid-cols-1 gap-4">
                            <div class="bg-blue-500 text-white p-4 rounded flex flex-col justify-between">
                                <h3 class="font-bold">Total Tasks</h3>
                                <p class="text-lg">{{ $totalTasks }}</p>
                            </div>
                            <div class="p-4 rounded flex flex-col justify-between" style="background-color: #FF8C00; color: #FFFFFF;">
                                <h3 class="font-bold">Tasks Due Today</h3>
                                <p class="text-lg">{{ $tasksDueToday }}</p>
                            </div>
                            <div class="bg-green-500 text-white p-4 rounded flex flex-col justify-between">
                                <h3 class="font-bold">Completed Tasks</h3>
                                <p class="text-lg">{{ $completedTasks }}</p>
                            </div>
                            <div class="bg-red-500 text-white p-4 rounded flex flex-col justify-between">
                                <h3 class="font-bold">Overdue Tasks</h3>
                                <p class="text-lg">{{ $overdueTasks }}</p>
                            </div>
                        </div>
                    </div>

                    <!-- Task Categories Overview -->
                    <div>
                        <h3 class="text-lg font-semibold mb-4">Task Categories Overview</h3>
                        <div style="width: 100%; height: 400px;">
                            <canvas id="categoryChart"></canvas>
                        </div>
                    </div>
    <script>
        const ctx = document.getElementById('categoryChart').getContext('2d');
        const categoryChart = new Chart(ctx, {
            type: 'pie',
            data: {
                labels: @json($tasksByCategory->pluck('category')),
                datasets: [{
                    label: 'Tasks by Category',
                    data: @json($tasksByCategory->pluck('count')),
                    backgroundColor: ['#FF6384', '#36A2EB', '#FFCE56', '#4BC0C0'],
                }]
            },
            options: {
                responsive: true,
                maintainAspectRatio: false
            }
        });
    </script>


    <!-- Calendar Overview -->
    <div class="py-6">
        <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
            <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6 text-gray-900">
                    <h3 class="text-lg font-semibold mb-4">Calendar Overview</h3>
                    <div id="calendar" style="width: 100%; height: 500px;"></div>
                </div>
            </div>
        </div>
    </div>

    <script>
        document.addEventListener('DOMContentLoaded', function () {
            var calendarEl = document.getElementById('calendar');

            var calendar = new FullCalendar.Calendar(calendarEl, {
                initialView: 'dayGridMonth',
                headerToolbar: {
                    left: 'prev,next today',
                    center: 'title',
                    right: 'dayGridMonth,dayGridWeek,timeGridDay'
                },
                events: @json($tasks->map(function ($task) {
                    return [
                        'title' => $task->title,
                        'start' => $task->deadline ? $task->deadline->toIso8601String() : null
                    ];
                })).filter(event => event.start !== null)
            });

            calendar.render();
        });
    </script>

</x-app-layout>
```

Tasks/index.blade.php



My Tasks

Active Tasks

Web Project
Deadline: 2024-12-20 23:19:00
Category: School

LABORATORY 3
Deadline: 2024-12-16 00:00:00
Category: School

CHRISTMAS PARTY
Deadline: 2024-12-23 09:00:00
Category: Personal

DANCE PRACTICE
Deadline: 2024-12-22 13:30:00
Category: Other

Completed Tasks

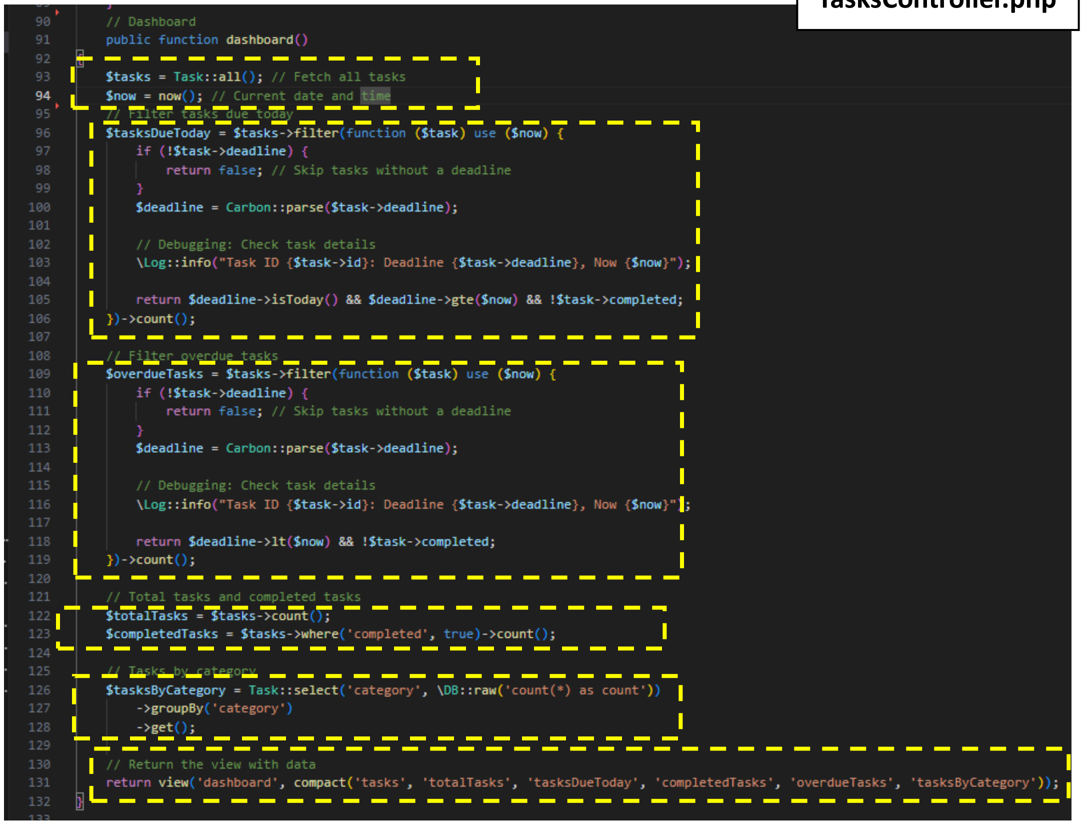
NETWORKING2 PROJECT
Completed on: 2024-12-17 13:00:00

PRESENTATION
Completed on: 2024-12-14 14:00:00

2. Controller Logic

2.1 Dashboard Logic

TasksController.php



```php
    // Dashboard
    public function dashboard()
    {
        $tasks = Task::all(); // Fetch all tasks
        $now = now(); // Current date and time
        // Filter tasks due today
        $tasksDueToday = $tasks->filter(function ($task) use ($now) {
            if (!$task->deadline) {
                return false; // Skip tasks without a deadline
            }
            $deadline = Carbon::parse($task->deadline);

            // Debugging: Check task details
            \Log::info("Task ID {$task->id}: Deadline {$task->deadline}, Now {$now}");

            return $deadline->isToday() && $deadline->gte($now) && !$task->completed;
        })->count();

        // Filter overdue tasks
        $overdueTasks = $tasks->filter(function ($task) use ($now) {
            if (!$task->deadline) {
                return false; // Skip tasks without a deadline
            }
            $deadline = Carbon::parse($task->deadline);

            // Debugging: Check task details
            \Log::info("Task ID {$task->id}: Deadline {$task->deadline}, Now {$now}");

            return $deadline->lt($now) && !$task->completed;
        })->count();

        // Total tasks and completed tasks
        $totalTasks = $tasks->count();
        $completedTasks = $tasks->where('completed', true)->count();

        // Tasks by category
        $tasksByCategory = Task::select('category', \DB::raw('count(*) as count'))
            ->groupBy('category')
            ->get();

        // Return the view with data
        return view('dashboard', compact('tasks', 'totalTasks', 'tasksDueToday', 'completedTasks', 'overdueTasks', 'tasksByCategory'));
    }
```

$tasks = Task::all(); Retrieves all tasks from the database to analyze and display
$now = now(); Captures the current date and time for comparisons

Tasks Due Today

- Filters tasks with:
  - A deadline set for today.
  - A deadline time in the future.
  - A status of not completed.
- Uses Carbon::isToday() and gte() for accurate filtering.

Overdue Tasks

- Filters tasks with:
  - A past deadline.
  - A status of not completed.
- Uses **Carbon::lt()** to identify overdue tasks.

Total and Completed Tasks

- **$totalTasks:** Counts all tasks in the system.
- **$completedTasks:** Counts tasks marked as completed.

Tasks by Category

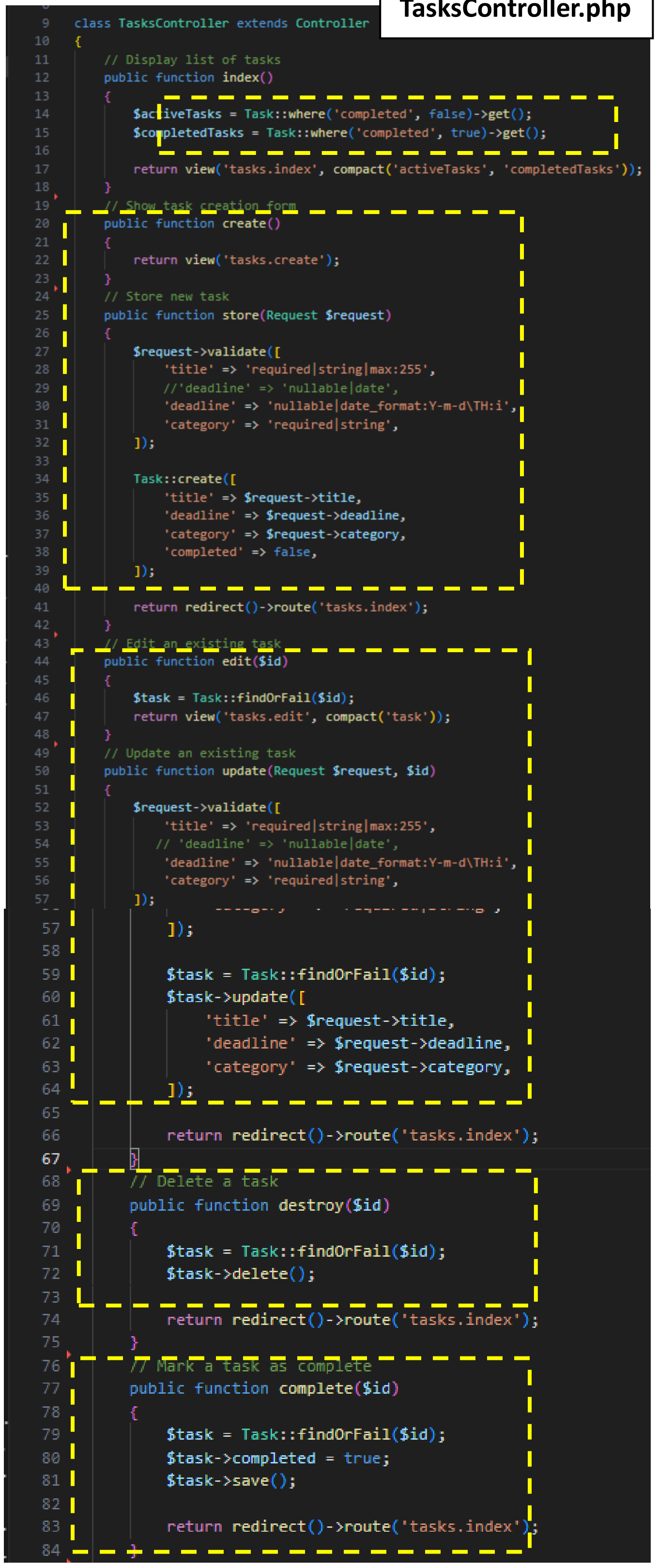
- Groups tasks by their category and counts each group.

Return Data to View

- **return view('dashboard', compact(...));**
  - Purpose: Passes all calculated data **($tasks, $totalTasks, $tasksDueToday, $completedTasks, $overdueTasks, $tasksByCategory)** to the dashboard view for rendering.

2.2 Task Management Logic

TasksController.php



```php
class TasksController extends Controller
{
    // Display list of tasks
    public function index()
    {
        $activeTasks = Task::where('completed', false)->get();
        $completedTasks = Task::where('completed', true)->get();

        return view('tasks.index', compact('activeTasks', 'completedTasks'));
    }
    // Show task creation form
    public function create()
    {
        return view('tasks.create');
    }
    // Store new task
    public function store(Request $request)
    {
        $request->validate([
            'title' => 'required|string|max:255',
            //'deadline' => 'nullable|date',
            'deadline' => 'nullable|date_format:Y-m-d\TH:i',
            'category' => 'required|string',
        ]);

        Task::create([
            'title' => $request->title,
            'deadline' => $request->deadline,
            'category' => $request->category,
            'completed' => false,
        ]);

        return redirect()->route('tasks.index');
    }
    // Edit an existing task
    public function edit($id)
    {
        $task = Task::findOrFail($id);
        return view('tasks.edit', compact('task'));
    }
    // Update an existing task
    public function update(Request $request, $id)
    {
        $request->validate([
            'title' => 'required|string|max:255',
          // 'deadline' => 'nullable|date',
            'deadline' => 'nullable|date_format:Y-m-d\TH:i',
            'category' => 'required|string',
        ]);
        ]);

        $task = Task::findOrFail($id);
        $task->update([
            'title' => $request->title,
            'deadline' => $request->deadline,
            'category' => $request->category,
        ]);

        return redirect()->route('tasks.index');
    }
    // Delete a task
    public function destroy($id)
    {
        $task = Task::findOrFail($id);
        $task->delete();

        return redirect()->route('tasks.index');
    }
    // Mark a task as complete
    public function complete($id)
    {
        $task = Task::findOrFail($id);
        $task->completed = true;
        $task->save();

        return redirect()->route('tasks.index');
    }
```

Display List of Tasks

- **$activeTasks = Task::where('completed', false)->get();**
    - Retrieves tasks not marked as completed for the active list.
- **$completedTasks = Task::where('completed',** true)->get();
    - Retrieves tasks marked as completed for display in the completed section.

Create New Task

- **$request->validate([...]);**
    - Validates user input, ensuring title, deadline, and category meet requirements.
    - Deadline includes both date and time validation with **date_format:Y-m-d\TH:i.**
- **Task::create([...]);**
    - Stores the new task in the database with default completed set to false.

Edit and Update Task

- **Task::findOrFail($id);**
    - Fetches the task by its ID; throws an error if it doesn't exist.
- **$task->update([...]);**
    - Updates the task's title, deadline, and category fields in the database.

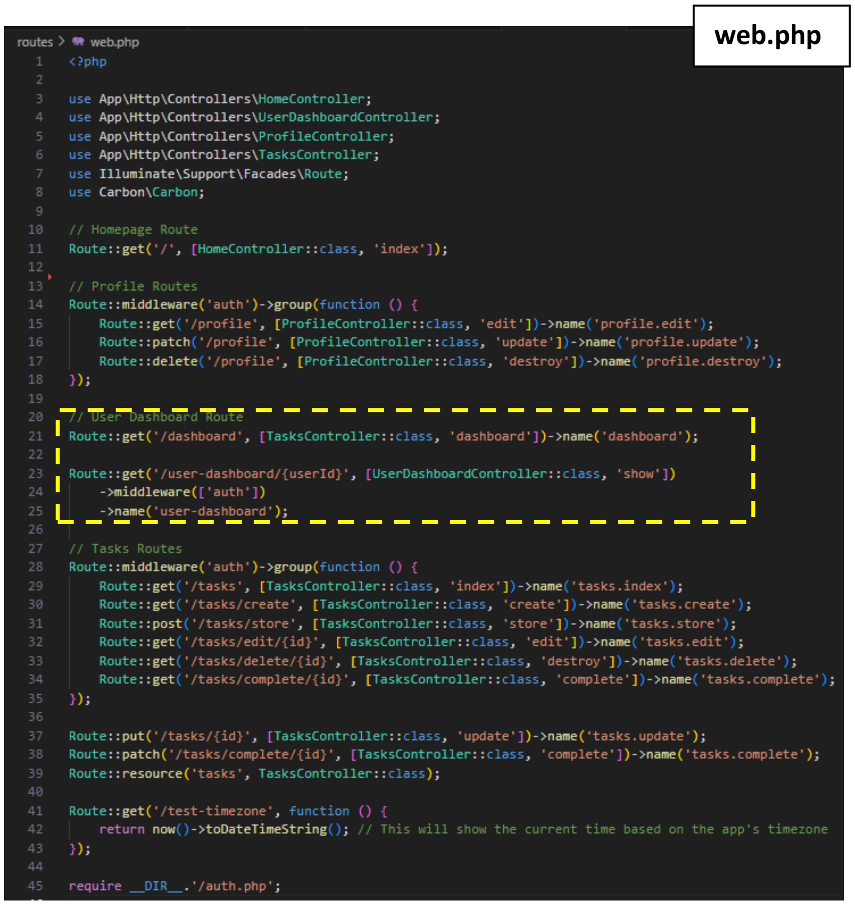
Delete Task

- **public function destroy($id)**
    - Deletes the task identified by its ID from the database.

Mark Task as Complete

- **$task->completed = true;**
    - Sets the completed status of the task to true.
    - Saves the change to the database.

3. Route Assignments



web.php

routes > web.php

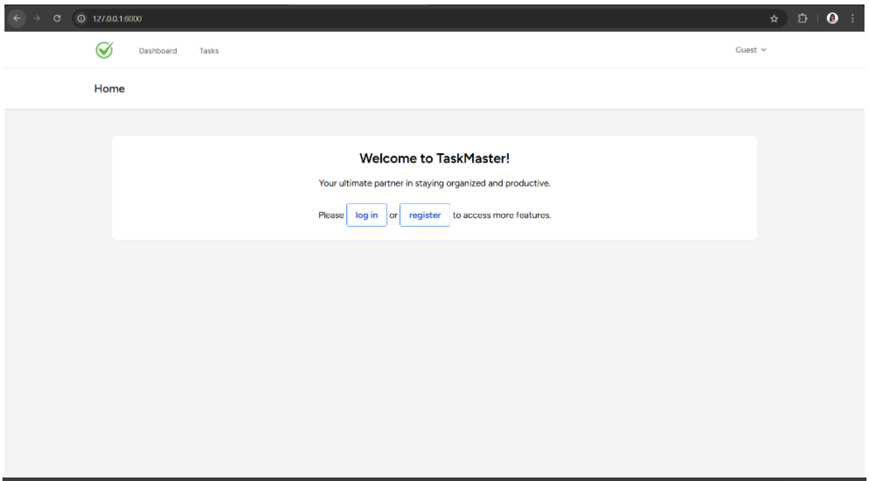
```php
<?php

use App\Http\Controllers\HomeController;
use App\Http\Controllers\UserDashboardController;
use App\Http\Controllers\ProfileController;
use App\Http\Controllers\TasksController;
use Illuminate\Support\Facades\Route;
use Carbon\Carbon;

// Homepage Route
Route::get('/', [HomeController::class, 'index']);

// Profile Routes
Route::middleware('auth')->group(function () {
    Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
    Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
    Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');
});

// User Dashboard Route
Route::get('/dashboard', [TasksController::class, 'dashboard'])->name('dashboard');

Route::get('/user-dashboard/{userId}', [UserDashboardController::class, 'show'])
    ->middleware(['auth'])
    ->name('user-dashboard');

// Tasks Routes
Route::middleware('auth')->group(function () {
    Route::get('/tasks', [TasksController::class, 'index'])->name('tasks.index');
    Route::get('/tasks/create', [TasksController::class, 'create'])->name('tasks.create');
    Route::post('/tasks/store', [TasksController::class, 'store'])->name('tasks.store');
    Route::get('/tasks/edit/{id}', [TasksController::class, 'edit'])->name('tasks.edit');
    Route::get('/tasks/delete/{id}', [TasksController::class, 'destroy'])->name('tasks.delete');
    Route::get('/tasks/complete/{id}', [TasksController::class, 'complete'])->name('tasks.complete');
});

Route::put('/tasks/{id}', [TasksController::class, 'update'])->name('tasks.update');
Route::patch('/tasks/complete/{id}', [TasksController::class, 'complete'])->name('tasks.complete');
Route::resource('tasks', TasksController::class);

Route::get('/test-timezone', function () {
    return now()->toDateTimeString(); // This will show the current time based on the app's timezone
});

require __DIR__.'/auth.php';
```

User Dashboard Route
This route listens for a GET request to **/user-dashboard/{userId}** where **{userId}** is a dynamic parameter.

This route points to the **dashboard()** method in the **TasksController**, which is responsible for handling logic related to displaying tasks or any other relevant data on the dashboard page.

The **show()** method in the **UserDashboardController** is responsible for handling the logic related to displaying the user dashboard for a specific user (identified by **userId**).

Account registration and login page

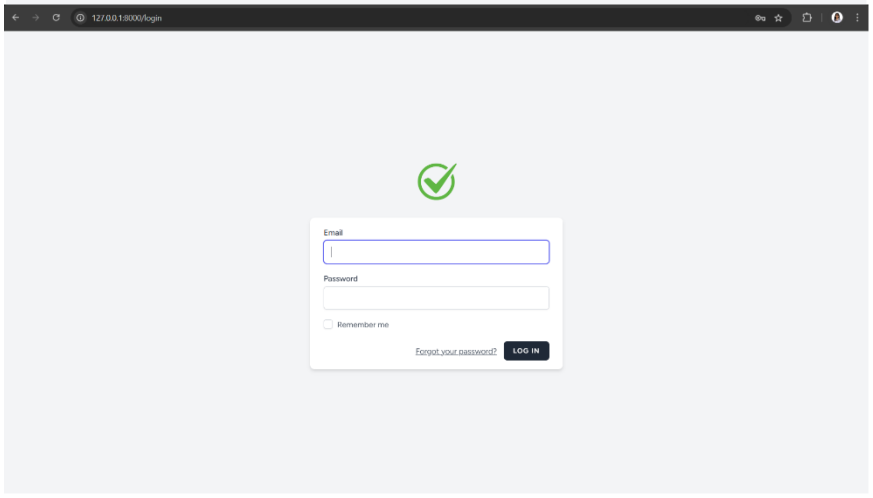


Home

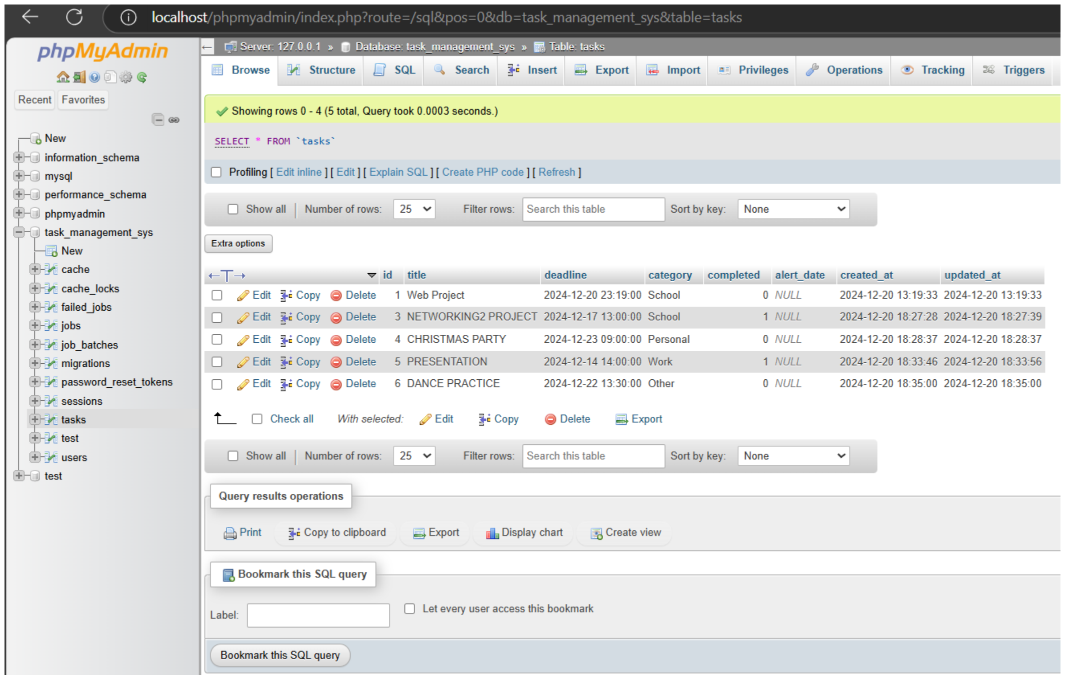
Welcome to TaskMaster!

Your ultimate partner in staying organized and productive.

Please log in or register to access more features.



Database Overview



Showing rows 0 - 4 (5 total, Query took 0.0003 seconds.)

```
SELECT * FROM `tasks`
```

	id	title	deadline	category	completed	alert_date	created_at	updated_at
Edit Copy Delete	1	Web Project	2024-12-20 23:19:00	School	0	NULL	2024-12-20 13:19:33	2024-12-20 13:19:33
Edit Copy Delete	3	NETWORKING2 PROJECT	2024-12-17 13:00:00	School	1	NULL	2024-12-20 18:27:28	2024-12-20 18:27:39
Edit Copy Delete	4	CHRISTMAS PARTY	2024-12-23 09:00:00	Personal	0	NULL	2024-12-20 18:28:37	2024-12-20 18:28:37
Edit Copy Delete	5	PRESENTATION	2024-12-14 14:00:00	Work	1	NULL	2024-12-20 18:33:46	2024-12-20 18:33:56
Edit Copy Delete	6	DANCE PRACTICE	2024-12-22 13:30:00	Other	0	NULL	2024-12-20 18:35:00	2024-12-20 18:35:00