Task Management Application

A full-stack task management application built with Laravel (backend) and Vue.js 3 (frontend), using MySQL as the database. This project features RESTful APIs, authentication with Laravel Sanctum, and a responsive frontend with pagination, sorting, filtering, and robust error handling.

Table of Contents

Setup Instructions

Backend (Laravel)

Frontend (Vue.js)

API Documentation

Authentication Endpoints

Task Endpoints

User Endpoint

Additional Decisions

Running the Application

Evaluation Criteria

Application Structure

Frontend (Vue.js)

Backend (Laravel)

Notes

Setup Instructions

Backend (Laravel)

Clone the Repository:

git clone https://github.com/kochieng50/task-management-backend.git

cd task-management-backend

Install Dependencies:

composer install

Configure Environment:Copy the .env.example file to .env:

cp .env.example .env

Update .env with your MySQL credentials and Sanctum settings:

APP\_URL=http://localhost:8000

DB\_CONNECTION=mysql

DB\_HOST=127.0.0.1

DB\_PORT=3306

DB\_DATABASE=task\_management

DB\_USERNAME=root

DB\_PASSWORD=

SANCTUM\_STATEFUL\_DOMAINS=localhost:5173

Generate Application Key:

php artisan key:generate

Run Migrations:

php artisan migrate

Start the Laravel Server:

php artisan serve

The backend will run at http://localhost:8000.

Frontend (Vue.js)

Clone the Repository:

git clone https://github.com/kochieng50/task-management.git

cd task-management

Install Dependencies:

npm install

Start the Vue Development Server:

npm run dev

The frontend will run at http://localhost:5173 (default Vite port).

API Documentation

All API endpoints are prefixed with /api and hosted at http://localhost:8000/api.

Authentication Endpoints

Register: POST /api/register

Description: Register a new user.

Body:{

"name": "John Doe",

"email": "john@example.com",

"password": "password123",

"password\_confirmation": "password123"

}

Response: User data and Sanctum token.

Login: POST /api/login

Description: Authenticate a user.

Body:{

"email": "john@example.com",

"password": "password123"

}

Response: User data and Sanctum token.

Logout: POST /api/logout

Description: Log out the authenticated user (requires authentication).

Headers: Authorization: Bearer <token>

Response: Success message.

Task Endpoints

List Tasks: GET /api/tasks

Description: Retrieve a paginated list of tasks with optional sorting and filtering (requires authentication).

Query Parameters:

page: Page number for pagination.

sort: Field to sort by (e.g., due\_date, title).

order: Sort order (asc or desc).

title: Filter by task title.

due\_date: Filter by due date.

priority: Filter by priority (low, medium, high).

status: Filter by status (pending, completed).

Example:GET /api/tasks?title=Meeting&status=pending&sort=due\_date&order=asc

Response: Paginated task list.

Create Task: POST /api/tasks

Description: Create a new task (requires authentication).

Body:{

"title": "Team Meeting",

"description": "Discuss project updates",

"due\_date": "2025-08-01",

"priority": "high",

"status": "pending"

}

Response: Created task data.

Show Task: GET /api/tasks/{id}

Description: Retrieve a specific task (requires authentication).

Response: Task data.

Update Task: PUT /api/tasks/{id}

Description: Update a task (requires authentication).

Body: Same as create task (partial updates allowed).

Response: Updated task data.

Delete Task: DELETE /api/tasks/{id}

Description: Delete a task (requires authentication).

Response: Success message.

User Endpoint

Get User: GET /api/user

Description: Retrieve authenticated user data (requires authentication).

Response:{

"id": 1,

"name": "John Doe",

"email": "john@example.com",

"avatar\_url": "https://example.com/avatar.jpg"

}

Additional Decisions

Caching: Utilized Laravel’s Cache facade to cache task listings, reducing database queries and improving performance.

Responsive Design: Implemented with Tailwind CSS (or inline styles in provided components) for a seamless UI across mobile, tablet, and desktop devices.

Error Handling: Comprehensive error handling on both backend and frontend with clear, user-friendly error messages for validation failures and API errors.

Authorization: Sanctum middleware ensures users can only access and modify their own tasks, enforced via user\_id in the tasks table.

Running the Application

Start the Laravel Backend:

cd task-management-backend

php artisan serve

Start the Vue Frontend:

cd task-management

npm run dev

Access the Application:Open your browser and navigate to http://localhost:5173.

Evaluation Criteria

Code Structure: Modular and maintainable with clear separation of concerns (controllers, models, components).

API Design: RESTful endpoints with proper validation, pagination, and error handling.

Frontend UX/UI: Intuitive navigation, responsive design, and user feedback (e.g., toasts, loading states).

Authentication: Secure token-based authentication with Sanctum and protected routes in Vue.js.

Performance: Optimized database queries and caching for efficient data retrieval.

Application Structure

Frontend (Vue.js)

task-management/

├── public/

│ ├── index.html # Main HTML file for Vue.js SPA

│ └── favicon.ico # Favicon

├── src/

│ ├── assets/ # Static assets

│ │ ├── logo.png # Application logo

│ │ └── styles.css # Global CSS (if not using inline styles)

│ ├── components/ # Reusable components

│ │ ├── Navbar.vue # Top navigation with user profile and logout

│ │ └── Sidebar.vue # Sidebar with Dashboard/Tasks links

│ ├── views/ # Page-level components

│ │ ├── Dashboard.vue # Task statistics and filters

│ │ ├── Tasks.vue # Task list with filtering and actions

│ │ ├── TaskCreate.vue # Form to create a task

│ │ ├── TaskEdit.vue # Form to edit a task

│ │ ├── Login.vue # Login page

│ │ ├── Register.vue # Registration page

│ │ └── ForgotPassword.vue # Forgot password page

│ ├── router/

│ │ └── index.js # Vue Router configuration

│ ├── services/

│ │ └── api.js # Axios setup for API calls

│ ├── App.vue # Root Vue component

│ ├── main.js # Entry point for Vue.js app

│ └── store/ # Vuex/Pinia store (optional)

│ └── index.js # State management

├── .gitignore # Ignores node\_modules/, dist/, .env

├── package.json # Node.js dependencies and scripts

├── vite.config.js # Vite configuration

└── README.md # Frontend documentation

Backend (Laravel)

task-management-backend/

├── app/

│ ├── Http/

│ │ ├── Controllers/

│ │ │ ├── Auth/

│ │ │ │ ├── LoginController.php # POST /login

│ │ │ │ ├── RegisterController.php # POST /register

│ │ │ │ └── ForgotPasswordController.php # POST /forgot-password

│ │ │ ├── TaskController.php # /tasks CRUD

│ │ │ └── UserController.php # /user endpoint

│ │ ├── Middleware/

│ │ │ └── Authenticate.php # Sanctum middleware

│ │ └── Requests/

│ │ ├── TaskRequest.php # Task validation

│ │ ├── LoginRequest.php # Login validation

│ │ └── RegisterRequest.php # Registration validation

│ ├── Models/

│ │ ├── User.php # User model

│ │ └── Task.php # Task model

│ └── Providers/

│ └── AuthServiceProvider.php # Authentication policies

├── config/

│ ├── app.php # App configuration

│ ├── auth.php # Sanctum configuration

│ └── sanctum.php # Sanctum settings

├── database/

│ ├── migrations/

│ │ ├── 2023\_01\_01\_000001\_create\_users\_table.php # Users table

│ │ └── 2023\_01\_01\_000002\_create\_tasks\_table.php # Tasks table

│ ├── seeders/

│ │ └── DatabaseSeeder.php # Seed data

│ └── factories/

│ ├── UserFactory.php # User factory

│ └── TaskFactory.php # Task factory

├── routes/

│ ├── api.php # API routes

│ └── web.php # Optional web routes

├── .env # Environment variables

├── .gitignore # Ignores vendor/, .env

├── composer.json # PHP dependencies

├── artisan # Laravel CLI

└── README.md # Backend documentation

Notes

Repository Structure: This README assumes separate repositories for frontend (task-management) and backend (task-management-backend). For a monorepo, combine them:task-management/

├── frontend/ # Vue.js files

├── backend/ # Laravel files

├── .gitignore

└── README.md

CORS Configuration: Ensure the backend allows requests from http://localhost:5173. Update config/cors.php:'paths' => ['api/\*'],

'allowed\_origins' => ['http://localhost:5173'],

Git Setup: To push changes (resolving previous errors):cd task-management

git init

git add .

git commit -m "Initial frontend commit"

git branch -M main

git remote add origin https://github.com/kochieng50/task-management.git

git push -u origin main

Repeat for task-management-backend with its repository URL.

Tailwind CSS: The project mentions Tailwind CSS, but provided components use inline styles. To use Tailwind, add it to package.json and configure in vite.config.js.

Database Schema:

Users Table: id, name, email, password, avatar\_url, created\_at, updated\_at.

Tasks Table: id, user\_id, title, description, due\_date, priority (low, medium, high), status (pending, completed), created\_at, updated\_at.

For additional setup help, specific file contents, or monorepo configuration, please reach out!