# **Introduction**

This file covers development and features of an inventory system created as an assignment. Built using Laravel 11, this web application is hosted locally on a XAMPP server. The system is therefore used to manage product data proficiently in simulating the real world scenario of inventory management for small scale operations. The user gets to see a user friendly interface and functionality that can enable a user to add, view, edit, or delete products in an inventory.

This system uses CRUD functionalities to operate, employing best practices in coding for web development and will work on using the Laravel framework that uses an architecture of MVC, maintains well-organized code and ease of maintenance and expansion. This project considers data management, user experience, and security attributes vital to any inventory management solution.

# **Features and Functionality**

The main functions of this application are based on CRUD operations:

1. **Create:** The system will let the user to add a new product in inventory. A form is displayed where users can enter details such as name of product, product ID, and price. This will enforce data integrity, for instance, the price field should be a positive numeric number.
2. **Read:** The home page displays the list of all the products in an ordered table format. This contains entry IDs, names, product IDs, price, and date of creation. This view helps to easily view inventory at a glance, which is helpful in times of quick reference or check up on inventory.
3. **Update**: There is an "Edit" button on the list of products, which can update information about the products. When the "Edit" button is clicked, a form opens wherein the already current product details will be filled in for easy modification by the user. When the form is submitted, the changes are saved directly into the database.
4. **Delete:** Every product can be deleted using a "Delete" button. And to prevent accidental deletions, a warning prompt is displayed, and thus the user must approve the deletion before it actually occurs.
5. **Upload Image:** You can upload picture of product while adding that product in inventory.

*Other features that the application contains and raise its functionality and usability:*

1. **Validation:** The validation mechanisms implemented in Laravel would verify that the products to be stored in the database are correct and valid. Consider for example the price, which must contain only positive numbers, and the product ID upon which unique constraints can be applied.
2. **Responsiveness:** The design of the interface of the application would adapt to various devices including tablets and mobile phones so that it will look great on any device used.
3. **Confirmation Prompts:** Before an actual delete action is performed, a confirmation prompt is displayed to avoid accidental data deletion.

# **Technical Overview**

## **Application Development in Laravel and MVC Architecture**

Because of this, in Laravel's MVC, business logic will be separated from the user interface code. The **Model** layer will interact with a database. The **View** layer will be used for presentation, and the **Controller** will be a bridge connecting them all. This will make your code modular, organized, and debuggable.

## **Blade Templates**

The application uses the Blade engine of Laravel to create reusable components and UI layouts. For example, the entire product list row is created using a single Blade component. This makes it easy to apply consistent styling and to make layout changes when the need arises.

## **Database Management**

One of the greatest ways that Laravel's Eloquent ORM simplifies database operations is by the ease with which one can define a model for each table in a database. For this project, a model `Product` was built to represent every product entry made in the database with attributes such as name, product ID, price, and date of creation. Eloquent also makes it easy to perform CRUD operations without writing any SQL queries.

## **XAMPP and Local Hosting**

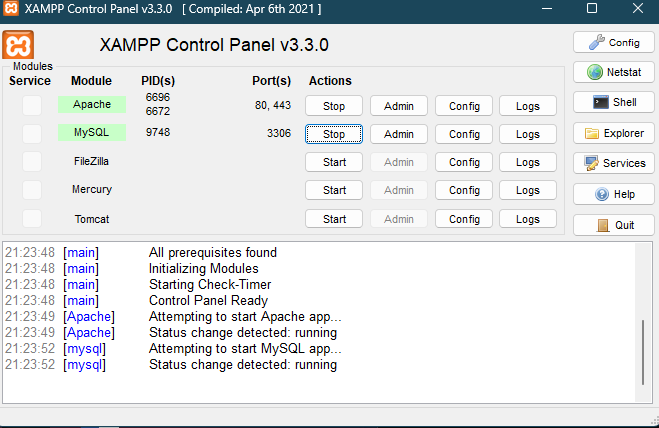
The application is running locally on XAMPP, that's a very popular development environment which includes Apache and MySQL. XAMPP makes it really easy to test and develop code without the need for a live server, hence great for local deployments, and debugging.

# **Good Practices and Other Features**

* Some good practices have been charted during the development of this application. Such practice insures code quality and maintenance.
* Components are created using Blade templates which reduce duplication and even ensure UI consistency.
* **Validation and Error Handling**: Inputs from the user are validated to make sure no error can occur and edge cases are handled. For example, only numeric values can be accepted in the price field and an alert is shown if any value is missing or not put correctly.
* **User Feedback:** That success messages and alerts will furnish the user with feedback about his actions, such as positive confirmations for correct edits and deletions performed.
* **Modularity:** The code is modular so that each function and part of your code has but one duty and is therefore easy to be maintained and extended.

# **Installation**

## **Start Xampp**

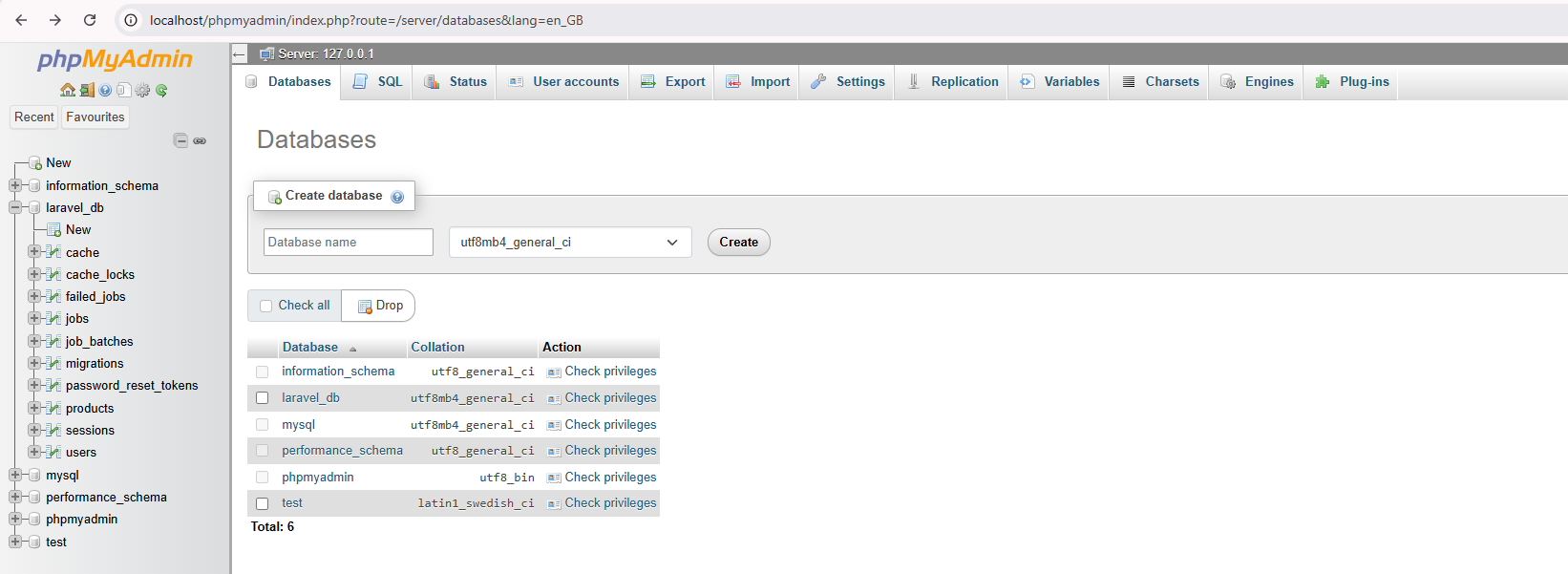
Open Xammp and start Apache and Mysql.

## **Copying the file**

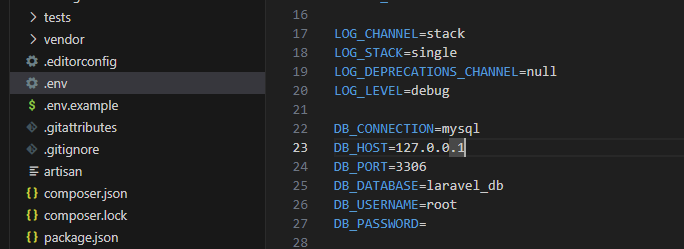
Extract the project folder name “laravel ” to C directory, then Xampp folder and in the xampp folder open htdocs and extract that folder there.

## **Create database**

Click on admin button in front of MySql after opening xampp. Then select database option and crraete a database . I have created with the name “laravel\_db”.

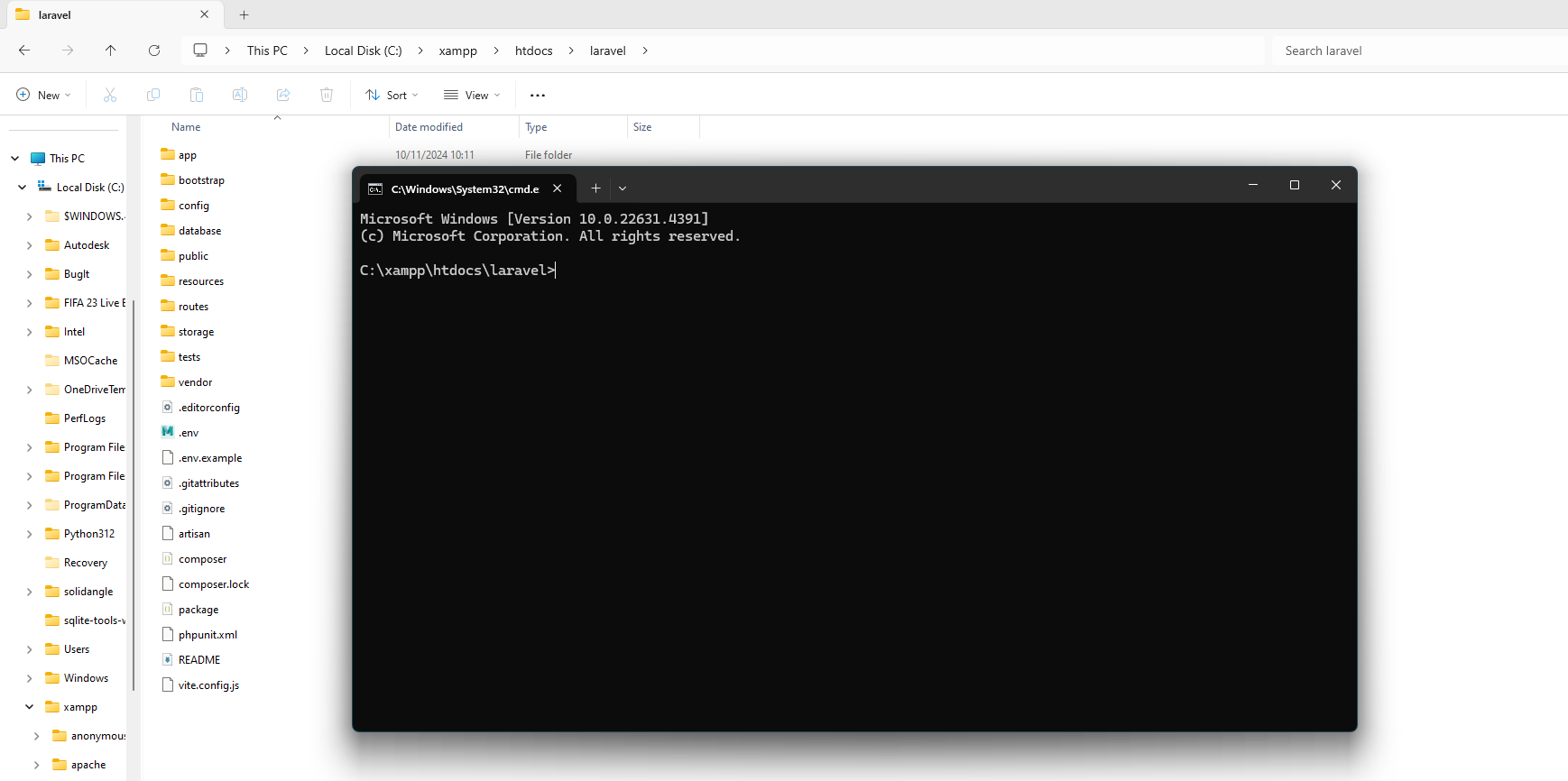


## **Configure .env file**

Open .env file do the following changes. Connection is “mysql” and database is “laravel\_db”.

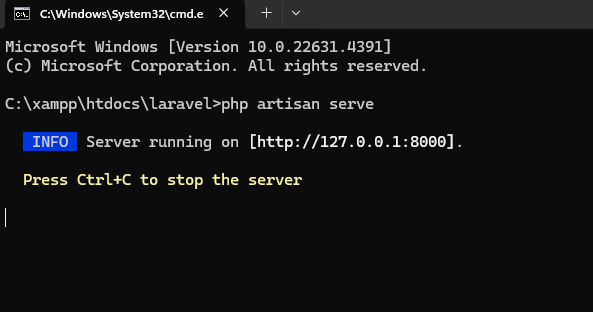
## **Navigate the Directory**

Navigate to the directory and open cmd.



## **Run server**

By using commad “ php artisan serve” run the server.



# **Conclusion**

This product inventory system using Laravel, it would be a proficient way of following up all product data. Its natural design and functionality mean that this tool can be easily used by anyone. Best practices for application mean that it's maintainable and scalable. The example of this project is the effective use of Laravel when developing a dynamic web application with further possibilities for extensions, such as adding user roles, search function, and exporting inventory data.