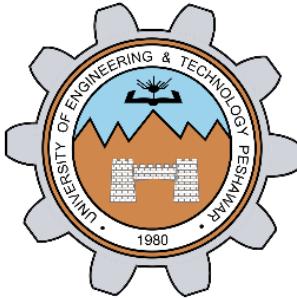


DBMS PROJECT REPORT
ONLINE INVENTORY MANAGEMENT SYSTEM

Key Milestone 04



CSE403L Database Management System Lab

Group members

Arsalan khan (22PWCSE2110)

Waseem (22PWCSE2179)

Adnan Zeb (22PWCSE2191)

Class Section: A

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”

Student Signature:

Submitted to:
Engr.Sumayyea Salahuddin
(JULY 08, 2025)

Department of Computer Systems Engineering
University of Engineering and Technology, Peshawar

Project Title

Online Inventory Management System using Laravel 12 and MySQL

Abstract

This project is a web-based inventory and order management system built with Laravel 12 (PHP framework) and MySQL. It allows users to manage products, suppliers, customers, purchases, and sales orders with real-time stock updates and user-based access. The system helps small and medium businesses maintain clear records, generate reports, and ensure data integrity.

Introduction

Managing inventory and sales manually can be time-consuming and error-prone. This system aims to automate those tasks using a modern PHP framework (Laravel) and a relational database (MySQL). With responsive design and user-friendly interfaces, the application supports streamlined business operations.

Objectives

- To automate inventory tracking
- To manage suppliers and customers efficiently
- To enable real-time stock updates
- To generate purchase and order reports
- To deploy a secure, scalable web system

Modules

- User Authentication (Login/Register)
- Supplier Management
- Customer Management
- Product Management (with Categories & Units)
- Purchases and Purchase Details
- Orders and Order Details
- Stock Updates
- Payment Tracking (paid, due)
- Admin Dashboard with Charts

Tools and Technologies

- Laravel 12 – PHP web framework
- PHP 8.3 – Server-side language (used on InfinityFree)
- MySQL – Relational database
- XAMPP / Laragon – Local development server
- GitHub – Version control
- InfinityFree – Free hosting with PHP 8.3 support

Project Benefits

- Centralized inventory and sales tracking
- Automatic stock updates on purchases and sales
- Reduced risk of overstocking or running out of stock
- Faster customer order processing
- Transparent reporting and accountability
- Easy to maintain and extend

Security Features

- Laravel's built-in CSRF protection
- Hashed passwords using Bcrypt
- Authenticated routes
- Form validation for all input fields

Challenges Faced

- Handling many-to-many relationships with pivot tables (e.g., order_details)
- Real-time stock updates during simultaneous orders
- Deployment limitations with shared/free hosting (e.g., InfinityFree's file limits)

Real-World Use Case

Consider a retail store with multiple products and frequent purchase and sales transactions. Using this system, the store owner can easily:

- Add or update product details
- Track which supplier delivered which product and when
- Generate invoices and receipts for customer orders
- Monitor daily stock movement without relying on manual registers or Excel

System Architecture

The application follows the MVC (Model-View-Controller) architecture:

- Model: Handles data logic and communicates with the MySQL database
- View: Blade templates render dynamic HTML pages
- Controller: Manages user requests and responses

Scalability and Maintenance

Laravel's modular structure and use of Composer packages make the system highly scalable. As the business grows, features like multi-user roles, warehouse management, or product barcodes can be added with minimal code changes.

Data Accuracy

One of the key goals of this system is to ensure that the inventory is always accurate. All stock increases or decreases are logged through purchases and orders, minimizing manual errors and providing a clear audit trail.

Who Can Use This System

- Small businesses (retailers, wholesalers)
- Inventory-based startups
- Schools/colleges managing store inventory
- Office supply departments
- Freelance developers as a base project

System Modules

- User Authentication
- Product Management
- Supplier and Customer Management
- Purchase Module
- Sales and Orders Module
- Stock Control
- Reports and Analytics Dashboard

Entity Analysis Table

Entity Name	Purpose	Key Attributes	Relationships
Users	Manages the system	id, name, email, username, password	One-to-many with Purchases (created_by, updated_by)
Suppliers	Source of purchased products	id, name, email, phone, shopname, bank details	One-to-many with Purchases
Customers	Places orders	id, name, email, phone, address	One-to-many with Orders
Categories	Groups products	id, category_name	One-to-many with Products
Units	Defines measurement units	id, unit_name	One-to-many with Products
Products	Inventory items	id, name, code, buying_price, selling_price, stock	Belongs to Category & Unit; Many-to-many with Orders & Purchases
Purchases	Records supplier purchases	id, purchase_no, purchase_date, supplier_id	Many Purchase_Details; Linked to Supplier & User
Purchase_Details	Line items for purchases	purchase_id, product_id, quantity, unitcost, total	Links Purchases and Products (composite key)
Orders	Records customer orders	id, customer_id, order_date, total, payment_type	Many Order_Details; Linked to Customers
Order_Details	Line items for orders	order_id, product_id, quantity, unitcost, total	Links Orders and Products (composite key)

Entity Explanation with Attributes and Relationships

1. Users

- Purpose:** System administrators or staff who log in and manage the system.
- Attributes:** id, name, email, username, password, photo
- Relationships:** One-to-many with purchases (created_by, updated_by)

2. Suppliers

- Purpose:** External vendors from whom products are purchased.
- Attributes:** id, name, email, phone, address, shopname, bank details

- **Relationships:** One-to-many with purchases

3. Customers

- **Purpose:** Buyers who place orders in the system.
- **Attributes:** id, name, email, phone, address, payment details
- **Relationships:** One-to-many with orders

4. Categories

- **Purpose:** Logical grouping of products.
- **Attributes:** id, category_name
- **Relationships:** One-to-many with products

5. Units

- **Purpose:** Measurement units for products (e.g., kg, pcs).
- **Attributes:** id, unit_name
- **Relationships:** One-to-many with products

6. Products

- **Purpose:** Items available for purchase and sale.
- **Attributes:** id, name, code, buying price, selling price, stock, image
- **Relationships:**
 - Belongs to a category
 - Belongs to a unit
 - Many-to-many with purchases (via purchase_details)
 - Many-to-many with orders (via order_details)

7. Purchases

- **Purpose:** Records of products bought from suppliers.
- **Attributes:** id, purchase_no, purchase_date, status, supplier_id
- **Relationships:**
 - Many purchase_details
 - Linked to suppliers and users

8. Purchase_Details

- **Purpose:** Line items for each purchase.
- **Attributes:** purchase_id, product_id, quantity, unitcost, total

- **Relationships:**
 - Composite key
 - Connects purchases and products

9. Orders

- **Purpose:** Sales made to customers.
- **Attributes:** id, order_date, total, status, customer_id, invoice_no, payment_type
- **Relationships:**
 - Many order_details
 - Linked to customers

10. Order_Details

- **Purpose:** Line items for each order.
- **Attributes:** order_id, product_id, quantity, unitcost, total
- **Relationships:**
 - Composite key
 - Connects orders and products

Entity Relationship Diagram (ERD)

1. users

- id (PK)
- name
- email
- username
- photo
- password

2. suppliers

- id (PK)
- name
- email
- phone
- address
- shopname
- type

- bank_name
- account_holder
- account_number
- photo

3. customers

- id (PK)
- name
- email
- phone
- address
- type
- bank_name
- account_holder
- account_number
- photo

4. categories

- id (PK)
- category_name

5. units

- id (PK)
- unit_name

6. products

- id (PK)
- product_name
- product_code
- buying_price
- selling_price
- stock
- product_image
- category_id (FK) → categories.id
- unit_id (FK) → units.id

7. purchases

- id (PK)
- purchase_date

- purchase_no
- purchase_status
- supplier_id (FK) → suppliers.id
- created_by (FK) → users.id
- updated_by (FK) → users.id

8. purchase_details

- purchase_id (FK, PK) → purchases.id
- product_id (FK, PK) → products.id
- quantity
- unitcost
- total

9. orders

- id (PK)
- order_date
- order_status
- total_products
- sub_total
- vat
- total
- invoice_no
- payment_type
- pay_int
- due_int
- customer_id (FK) → customers.id

10. order_details

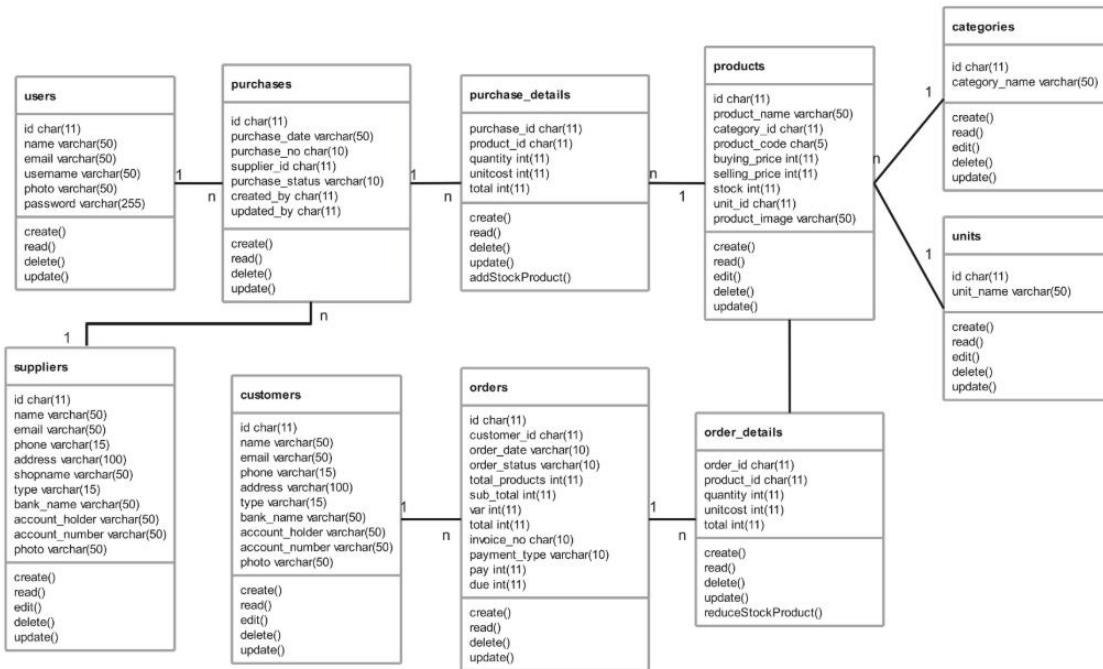
- order_id (FK, PK) → orders.id
- product_id (FK, PK) → products.id
- quantity
- unitcost
- total

Relationships

- **One-to-Many:**

- users → purchases
- suppliers → purchases
- customers → orders

- orders → order_details
- purchases → purchase_details
- products → purchase_details / order_details
- categories → products
- units → products

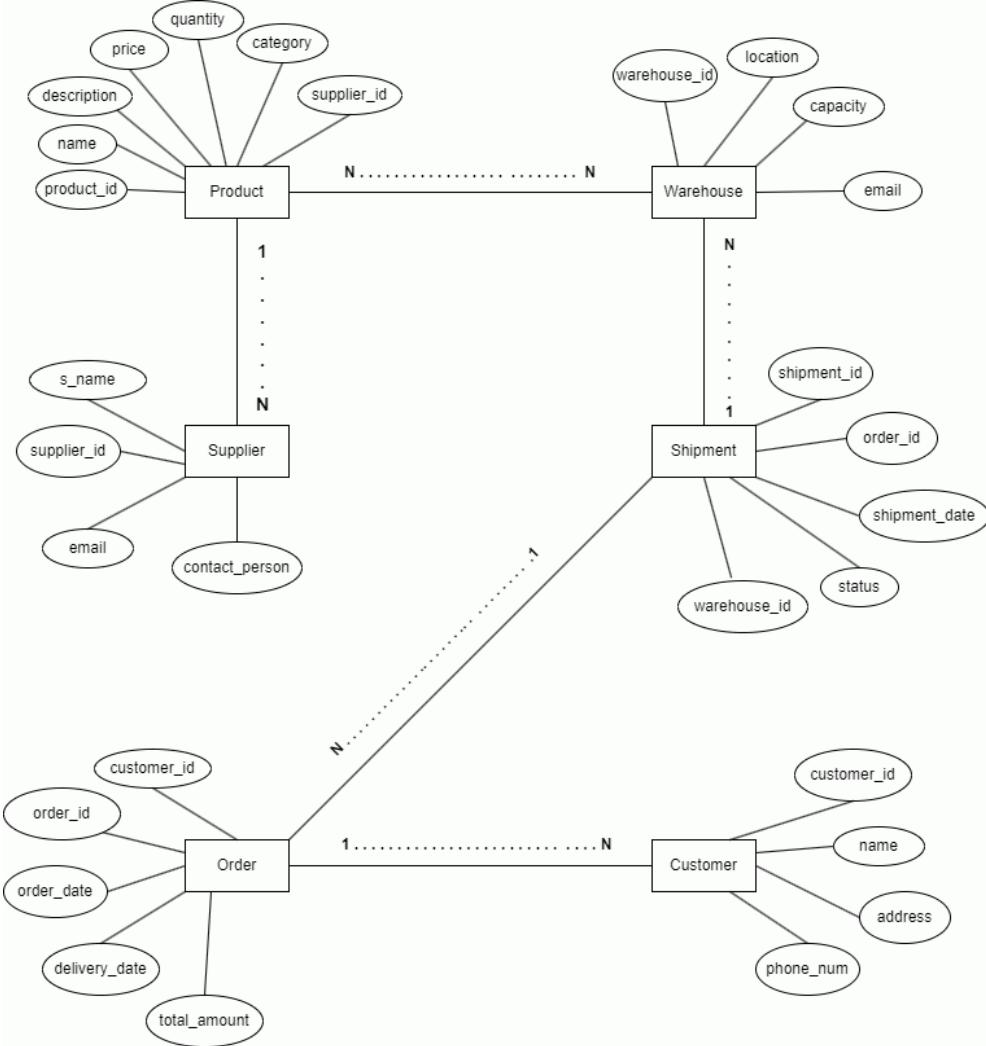


Enhanced Entity Relationship Diagram (EERD)

The Enhanced Entity Relationship Diagram (EERD) expands on the basic ERD by adding more advanced modeling features such as specialization, generalization, and more detailed relationships. It is used to better represent real-world scenarios in complex systems like inventory management.

In this project:

- **Customers** and **Suppliers** are generalized as **Persons**, since both share common attributes (e.g., name, email, phone).
- **Products** are associated with both **Categories** and **Units**, showing classification and measurement relationships.
- **Purchases** and **Orders** are transactional entities connecting **Users**, **Suppliers**, **Customers**, and **Products**.
- **Purchase_Details** and **Order_Details** are used to handle many-to-many relationships with quantities and prices.



Laravel Development Process

1: Setup Project

```
composer create-project laravel/laravel inventory-system
cd inventory-system
php artisan serve
```

2: Configure .env

```
DB_DATABASE=inventory_db
DB_USERNAME=root
DB_PASSWORD=
```

3: Create Models and Migrations

```
php artisan make:model Product -m  
php artisan make:model Supplier -m  
php artisan make:model Order -m
```

4: Run Migrations

```
php artisan migrate
```

5: Create Controllers

```
php artisan make:controller ProductController  
php artisan make:controller SupplierController
```

6: Blade Templates & Routes

- Views stored in `/resources/views`
- Routes defined in `routes/web.php`

7: Authentication

```
composer require laravel/ui  
php artisan ui bootstrap --auth  
npm install && npm run dev
```

Implementation Details

The system was built using **Laravel 8** and **MySQL**. Laravel's MVC structure was used to organize models, controllers, and views. All modules (products, orders, purchases, users) include full CRUD functionality using Eloquent ORM. Authentication is handled with Laravel UI.

The project was deployed on **InfinityFree** using **PHP 8.3**, and the database was imported via phpMyAdmin.

LARAVEL MVC IMPLEMENTATION

MODEL LAYER

The screenshot shows a code editor interface with the title bar "Models". The left sidebar displays the "EXPLORER" view, which lists various model files under the "MODELS" category, including Account.php, Category.php, Customer.php, DepositCategory.php, Order.php, OrderDetails.php, Product.php, Purchase.php, PurchaseDetails.php, Quotation.php, QuotationDetails.php, Supplier.php, Unit.php, and User.php. The main editor area shows the "Order.php" file:

```
<?php  
namespace App\Models;  
  
use App\Enums\OrderStatus;  
use Illuminate\Database\Eloquent\Model;  
use Illuminate\Database\Eloquent\Relations\HasMany;  
use Illuminate\Database\Eloquent\Relations\BelongsTo;  
use Illuminate\Database\Eloquent\Factories\HasFactory;  
  
class Order extends Model  
{  
    use HasFactory;  
  
    protected $guarded = [  
        'id',  
    ];  
  
    protected $fillable = [  
        'customer_id',  
        'order_date',  
        'order_status',  
        'total_products',  
        'sub_total',  
        'vat',  
        'total',  
        'invoice_no',  
        'payment_type',  
        'pay',  
        'due',  
    ].
```

A tooltip message "A git repository was found in the parent folders of the workspace or the open file(s). Would you like to open the repository?" is visible in the center-right area. At the bottom right, there are buttons for "Source: Git" with options "Yes", "Always", and "Never". The status bar at the bottom indicates "Ln 1, Col 1 Spaces: 4 UTF-8 LF {} PHP".

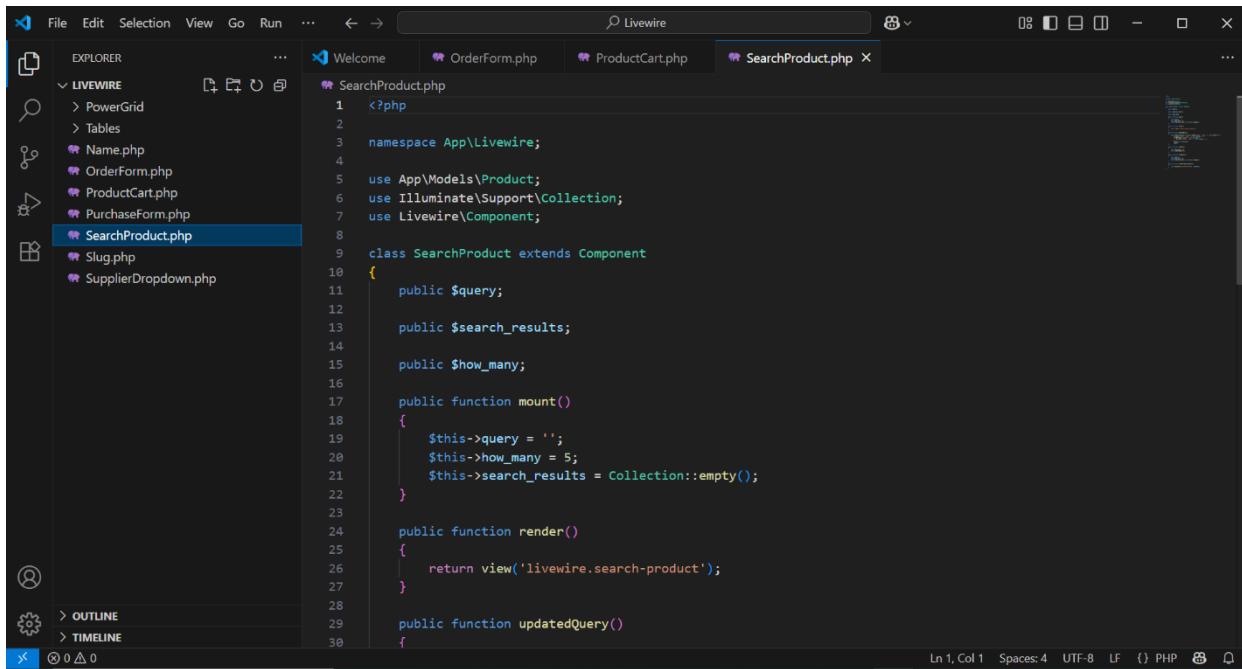
CONTROLLER LAYER

The screenshot shows a code editor interface with the title bar "Controllers". The left sidebar displays the "CONTROLLERS" view, listing controllers such as API, Auth, Dashboards, Order, Product, Purchase, Quotation, CartController.php, CategoryController.php, Controller.php, CustomerController.php, InvoiceController.php, ProfileController.php, SupplierController.php, UnitController.php, and UserController.php. The main editor area shows the "CustomerController.php" file:

```
<?php  
namespace App\Http\Controllers;  
  
use App\Models\Customer;  
use App\Http\Requests\Customer\StoreCustomerRequest;  
use App\Http\Requests\Customer\UpdateCustomerRequest;  
  
class CustomerController extends Controller  
{  
    public function index()  
    {  
        $customers = Customer::all();  
  
        return view('customers.index', [  
            'customers' => $customers  
        ]);  
    }  
  
    public function create()  
    {  
        return view('customers.create');  
    }  
  
    public function store(StoreCustomerRequest $request)  
    {  
        $customer = Customer::create($request->all());  
  
        /**
         * Handle upload an image
        */
```

The status bar at the bottom indicates "Ln 1, Col 1 Spaces: 4 UTF-8 LF {} PHP".

LIVEWIRE



The screenshot shows a code editor interface with a dark theme. The left sidebar is titled 'EXPLORER' and lists several files under a 'LIVEWIRE' folder, including PowerGrid, Tables, Name.php, OrderForm.php, ProductCart.php, PurchaseForm.php, SearchProduct.php (which is selected), Slug.php, and SupplierDropdown.php. The main area displays the contents of 'SearchProduct.php'. The code is written in PHP and defines a Livewire component:

```
<?php  
namespace App\LiveWire;  
use App\Models\Product;  
use Illuminate\Support\Collection;  
use Livewire\Component;  
  
class SearchProduct extends Component  
{  
    public $query;  
    public $search_results;  
    public $how_many;  
  
    public function mount()  
    {  
        $this->query = '';  
        $this->how_many = 5;  
        $this->search_results = Collection::empty();  
    }  
  
    public function render()  
    {  
        return view('livewire.search-product');  
    }  
  
    public function updatedQuery()  
    {  
        $this->search_results = Product::where('name', 'like', "%{$this->query}%")  
            ->take($this->how_many)  
            ->get();  
    }  
}
```

At the bottom right of the editor, there are status indicators: Line 1, Column 1, Spaces: 4, UTF-8, LF, {}, PHP, and a refresh icon.

Cloud MySQL Database Deployment

Steps:

1. Sign in to Aiven.io and create a new MySQL service.
2. After successful creation, the following database credentials were provided:

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=inventory-management-system
DB_USERNAME=root
DB_PASSWORD=

3. These values should be added to Laravel's .env file to connect the application to the cloud MySQL database.

MySQL Database

```
MariaDB [(none)]> USE `inventory-management-system`;  
Database changed  
MariaDB [inventory-management-system]> SHOW TABLES;  
+-----+  
| Tables_in_inventory-management-system |  
+-----+  
accounts  
categories  
customers  
deposit_categories  
failed_jobs  
migrations  
notifications  
order_details  
orders  
password_reset_tokens  
personal_access_tokens  
products  
purchase_details  
purchases  
quotation_details  
quotations  
shoppingcart  
suppliers  
units  
users  
+-----+  
20 rows in set (0.003 sec)
```

localhost/phpmyadmin

The screenshot shows the phpMyAdmin interface for the 'inventory-management-system' database. The left sidebar lists tables: accounts, categories, customers, deposit_categories, failed_jobs, migrations, notifications, orders, order_details, password_reset_tokens, personal_access_tokens, products, purchases, purchase_details, quotations, quotation_details, and shoppingcart. The main area displays a table structure for the 'accounts' table, with columns: id (Primary Key), name, email, password, created_at, updated_at, and deleted_at. The table has 10 rows.

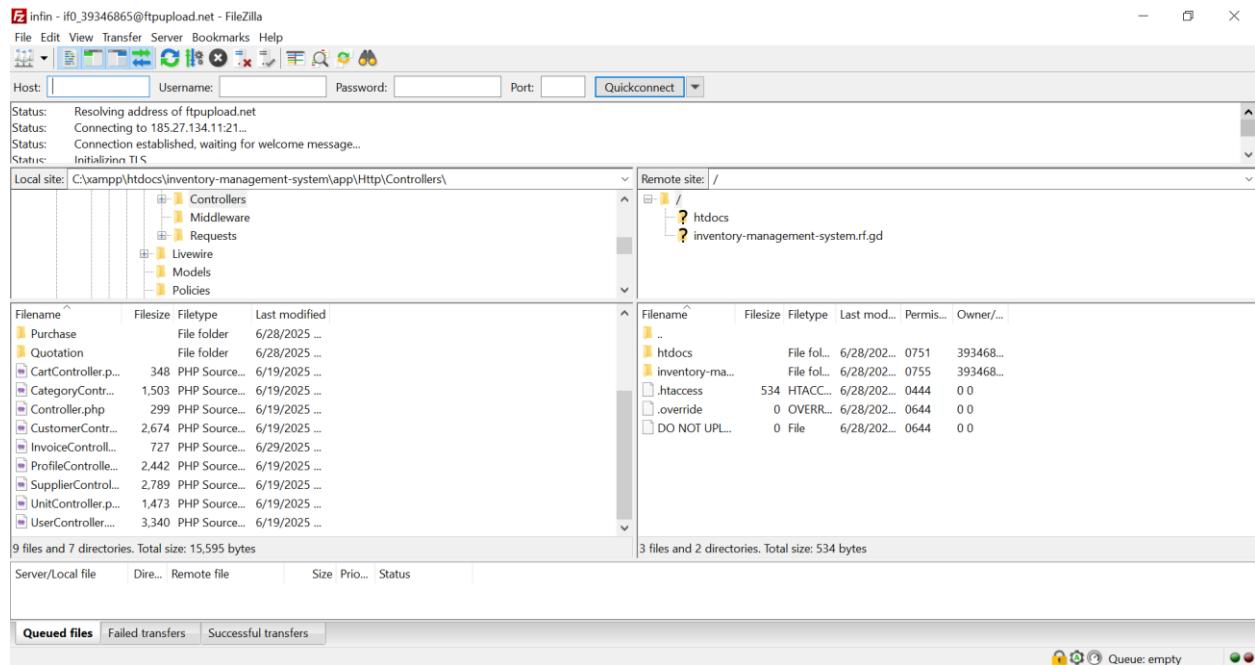
Table	Action	Rows	Type	Collation	Size	Overhead
accounts	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
categories	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	48.0 Kib	
customers	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci	48.0 Kib	
deposit_categories	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
failed_jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 Kib	
migrations	Browse Structure Search Insert Empty Drop	19	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
notifications	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 Kib	
orders	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	32.0 Kib	
order_details	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 Kib	
password_reset_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	
personal_access_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	48.0 Kib	
products	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci	64.0 Kib	
purchases	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	32.0 Kib	
Console	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci	16.0 Kib	

<input type="checkbox"/> purchase_details								0 InnoDB utf8mb4_unicode_ci 48.0 KiB
<input type="checkbox"/> quotations								0 InnoDB utf8mb4_unicode_ci 32.0 KiB
<input type="checkbox"/> quotation_details								0 InnoDB utf8mb4_unicode_ci 48.0 KiB
<input type="checkbox"/> shoppingcart								0 InnoDB utf8mb4_unicode_ci 16.0 KiB
<input type="checkbox"/> suppliers								1 InnoDB utf8mb4_unicode_ci 48.0 KiB
<input type="checkbox"/> units								2 InnoDB utf8mb4_unicode_ci 48.0 KiB
<input type="checkbox"/> users								2 InnoDB utf8mb4_unicode_ci 48.0 KiB
20 tables		Sum		51 InnoDB utf8mb4_general_ci				736.0 KiB
0 E								

Deployment:

1. Upload Project to FileZilla

Instead of GitHub, the Laravel project was uploaded using **FileZilla** — a file hosting platform that supports direct file uploads.



2. Deploy to InfinityFree

- Create a free hosting account on [InfinityFree.net](#).
- Upload the extracted public/ folder contents into the /htdocs directory via File Manager or FTP (e.g., FileZilla).
- Ensure the main Laravel index.php and .htaccess are inside /htdocs

A screenshot of the InfinityFree dashboard at dash.infinityfree.com/accounts. The top navigation bar includes links for Home, Profile, Accounts (which is selected), Free SSL Certificates, Website Builder, Domain Checker, Knowledge Base, and Community Forum. A yellow 'Go Premium' button and a user email '22pwcse2179@uetpeshawar.edu.pk' are also visible. The main content area is titled 'ACCOUNTS Hosting Accounts'. It features a section for 'Flutter Casual Games Toolkit' with a preview image and a 'Learn More' button. Below this is a list of 'Your Accounts' with one entry: 'if0_39346865 inventory-management-system.com'. A purple '+ Create Account' button is located to the right of this list. At the bottom left, it says 'Active Accounts: 1 / 3'.

The screenshot shows the InfinityFree dashboard interface, specifically the 'Hosting Accounts' section. It displays a single account entry for 'if0_39346865' which points to 'inventory-management-system.com'. There are buttons for creating a new account and navigating through the site.

3. Configure .env File

Update the database credentials in your .env file using Aiven.io's MySQL service:

```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:QZdeSoikDHSgZkiVGViVrlqaBVcSGN/1jXQOEHKUfcc=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=sql211.infinityfree.com
13 DB_PORT=3306
14 DB_DATABASE=if0_39346865_inventory_management_system
15 DB_USERNAME=if0_39346865
16 DB_PASSWORD=was744012345
17
18 BROADCAST_DRIVER=log
19 CACHE_DRIVER=file
20 FILESYSTEM_DISK=local
21 QUEUE_CONNECTION=sync
22 SESSION_DRIVER=file
23 SESSION_LIFETIME=120
24
25 MEMCACHED_HOST=127.0.0.1
```

Dashboard

The screenshot shows the dashboard of an inventory management system. At the top, there are two tabs: 'Inventory Project Report.pdf' and 'Laravel'. The main title is 'tabler'. Below the title, there's a navigation bar with links: Dashboard, Products, Orders, Purchases, Quotations, Pages, and Settings. A user profile 'waseem' is visible on the right.

The main area is titled 'OVERVIEW Dashboard'. It features four cards:

- 3 Products, 5 categories
- 11 Orders, 3 completed
- 1 Purchases, 0 today
- 0 Quotations, 0 today

Buttons for 'Add new Product' and 'Create new order' are located at the top right of this section.

Products Page

The screenshot shows the products page of the inventory management system. The title is 'tabler'. The navigation bar includes 'Products' which is currently selected. Other links are: Dashboard, Orders, Purchases, Quotations, Pages, and Settings.

The main content area is titled 'Products'. It includes a search bar with 'Search:' and a dropdown for 'Show' entries (set to 25). A table lists three products:

NO.	NAME	CODE	CATEGORY	QUANTITY	QUANTITY ALERT	ACTION
1	Stainless Steel Water Bottle (1L)	PC0006	Home Appliances	55	15	
2	Wireless Bluetooth Earbuds	PC0005	Sports & Outdoors	60	10	
3	Men's Cotton T-Shirt – Large	PC0004	Men's Fashion	60	15	

At the bottom, it says 'Showing 1 to 3 of 3 entries'.

Orders Page

The screenshot shows a web browser window titled "Inventory Project Report.pdf" with the URL "https://inventory-management-system.wuaze.com/orders". The page is titled "Orders" and displays a table of 4 entries. The columns are: NO., INVOICE NO., CUSTOMER, DATE, PAYMET, TOTAL, STATUS, and ACTION. The first three rows have a green "COMPLETE" status, while the fourth row has an orange "PENDING" status. Each row includes a "View" and a "Print" button in the ACTION column.

NO.	INVOICE NO.	CUSTOMER	DATE	PAYMET	TOTAL	STATUS	ACTION
1	INV-000011	Ayesha Siddiqui	04-07-2025	HandCash	€4,500.00	COMPLETE	
2	INV-000010	Usman Khan	03-07-2025	HandCash	€13,500.00	COMPLETE	
3	INV-000009	Sarah Ahmed	03-07-2025	HandCash	€4,500.00	PENDING	
4	INV-000008	Sarah Ahmed	03-07-2025	HandCash	€12,000.00	PENDING	

Purchases Page

The screenshot shows a web browser window titled "Inventory Project Report.pdf" with the URL "https://inventory-management-system.wuaze.com/purchases". The page is titled "Purchases" and displays a table of 1 entry. The columns are: NO., PURCHASE NO., SUPPLIER, DATE, TOTAL, STATUS, and ACTION. The entry has an "APPROVED" status. Each row includes a "View" and a "Edit" button in the ACTION column.

NO.	PURCHASE NO.	SUPPLIER	DATE	TOTAL	STATUS	ACTION
1	PRS-000001	Muhammad Ali	01-07-2025	€0.00	APPROVED	

Showing 1 to 1 of 1 entries

Suppliers

The screenshot shows a browser window for the 'tabler' application at the URL <https://inventory-management-system.wuaze.com/suppliers>. The page title is 'Suppliers'. The interface includes a search bar and a table with the following data:

NO.	NAME	EMAIL ADDRESS	SHOP NAME	TYPE	ACTION
1	Shahid Mehmood	shahid.mehmood@hotmail.com	SM Electronics	DISTRIBUTOR	
2	Muhammad Ali	ali.supplies@gmail.com	Ali Traders & Sons	WHOLESALER	

Showing 1 to 2 of 2 entries

Customers

The screenshot shows a browser window for the 'tabler' application at the URL <https://inventory-management-system.wuaze.com/customers>. The page title is 'Customers'. The interface includes a search bar and a table with the following data:

NO.	NAME	EMAIL	ACTION
1	Usman Khan	usman.khan84@yahoo.com	
2	Sarah Ahmed	sarah.ahmed92@gmail.com	
3	Ayesha Siddiqui	ayesha.siddiqui01@hotmail.com	

Showing 1 to 3 of 3 entries

Categories

NO.	NAME	SLUG	ACTION
1	Sports & Outdoors	sports-outdoors	
2	Men's Fashion	mens-fashion	
3	kids accessories	kids-accessories	
4	Home Appliances	home-appliances	

Units

NO.	NAME	SLUG	ACTION
1	Sports & Outdoors	sports-outdoors	
2	Men's Fashion	mens-fashion	
3	kids accessories	kids-accessories	
4	Home Appliances	home-appliances	

4. Import the Database

- Go to **phpMyAdmin** in InfinityFree control panel.
- Create a new database (e.g., inventory-management-system).
- Import your **.sql** file using the Import tab.

MySQL Connection Details

The screenshot shows the 'Manage if0_39346865' interface. On the left, a sidebar lists 'Account Options' such as Home, Upgrade to Premium, Statistics, FTP Details, MySQL Databases (which is selected), Deactivation History, and Account Settings. The main area is titled 'MySQL Connection Details' and contains fields for MySQL USERNAME (if0_39346865), MySQL PASSWORD (redacted), MySQL DATABASE NAME (if0_39346865_XXX), MySQL HOSTNAME (sql211.infinityfree.com), MySQL PORT (OPTIONAL) (3306), and a 'List of MySQL Databases' table. The table has columns for DATABASE NAME (if0_39346865_inventory_management_system), ACTIONS (phpMyAdmin and Delete buttons), and a 'Create Database' button.

The screenshot shows the 'Structure' tab of the phpMyAdmin interface for the 'if0_39346865_inventory_management_system' database. It displays a list of tables with their respective structures. The tables listed include accounts, categories, customers, deposit_categories, failed_jobs, migrations, notifications, orders, order_details, password_reset_tokens, personal_access_tokens, products, purchases, purchase_details, quotations, quotation_details, shoppingcart, and suppliers. Each table row includes links for Browse, Structure, Search, Insert, Empty, and Drop operations.

Table	Action	Rows	Type	Collation
accounts	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_unicode_ci
categories	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_unicode_ci
customers	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_unicode_ci
deposit_categories	Browse Structure Search Insert Empty Drop	10	InnoDB	utf8mb4_unicode_ci
failed_jobs	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
migrations	Browse Structure Search Insert Empty Drop	19	InnoDB	utf8mb4_unicode_ci
notifications	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
orders	Browse Structure Search Insert Empty Drop	11	InnoDB	utf8mb4_unicode_ci
order_details	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8mb4_unicode_ci
password_reset_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
personal_access_tokens	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
products	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_unicode_ci
purchases	Browse Structure Search Insert Empty Drop	1	InnoDB	utf8mb4_unicode_ci
purchase_details	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
quotations	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
quotation_details	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
shoppingcart	Browse Structure Search Insert Empty Drop	0	InnoDB	utf8mb4_unicode_ci
suppliers	Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_unicode_ci

<input type="checkbox"/> suppliers								2 InnoDB	utf8mb4_unicode_ci
<input type="checkbox"/> units								3 InnoDB	utf8mb4_unicode_ci
<input type="checkbox"/> users								7 InnoDB	utf8mb4_unicode_ci
20 tables		Sum					81 MyISAM latin1_swedish_ci		
<input type="checkbox"/>	<input type="checkbox"/> Check all					With selected:			

Products

php-myadmin.net/sql.php?db=if0_39346865_inventory_management_system&table=products&pos=0

Server: sq211.infinityfree.com » Database: if0_39346865_inventory_management_system » Table: products

[Browse](#) [Structure](#) [SQL](#) [Search](#) [Insert](#) [Export](#) [Import](#) [Operations](#)

Showing rows 0 - 2 (3 total, Query took 0.0008 seconds.)

```
SELECT * FROM `products`
```

Profiling [Edit inline](#) [Edit](#) [Explain SQL](#) [Create PHP code](#) [Refresh](#)

<input type="checkbox"/> Show all	Number of rows:	25	Filter rows:		Search this table		Sort by key:		None									
+ Options																		
<input type="checkbox"/>	Edit	Copy	Delete	5	Men's Cotton T-Shirt – Large	mens-cotton-t-shirt-large	PC0004	60	450	699	15	NULL	0	NULL	NULL	4	5	2023-01-15 10:45:23
<input type="checkbox"/>	Edit	Copy	Delete	6	Wireless Bluetooth Earbuds	wireless-bluetooth-earbuds	PC0005	60	1200	1750	10	NULL	0	NULL	NULL	6	5	2023-01-15 10:45:23
<input type="checkbox"/>	Edit	Copy	Delete	7	Stainless Steel Water Bottle (1L)	stainless-steel-water-bottle-1l	PC0006	55	450	699	15	NULL	0	NULL	NULL	3	5	2023-01-15 10:45:23

[Check all](#) [With selected:](#) [Edit](#) [Copy](#) [Delete](#) [Export](#)

Categories

← → ⚙ php-myadmin.net/sql.php?db=i0_39346865_inventory_management_system&table=categories ⭐ 🌐 School

Server: sq211.infinityfree.com » Database: i0_39346865_inventory_management_system » Table: categories

Browse Structure SQL Search Insert Export Import Operations

Showing rows 0 - 4 (5 total, Query took 0.00005 seconds.)

SELECT * FROM `categories`

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 ▾ Filter rows: Search this table Sort by key: None

+ Options

	id	name	slug	created_at	updated_at
<input type="checkbox"/>	1	kids accessories	kids-accessories	2025-06-19 10:54:15	2025-06-19 10:55:36
<input type="checkbox"/>	3	Home Appliances	home-appliances	2025-07-01 18:23:07	2025-07-01 18:23:07
<input type="checkbox"/>	4	Men's Fashion	mens-fashion	2025-07-01 18:23:23	2025-07-01 18:23:23
<input type="checkbox"/>	5	Beauty & Health	beauty-health	2025-07-01 18:23:42	2025-07-01 18:23:42
<input type="checkbox"/>	6	Sports & Outdoors	sports-outdoors	2025-07-01 18:24:00	2025-07-01 18:24:00

Check all With selected: Edit Copy Delete Export

Show all | Number of rows: 25 ▾ Filter rows: Search this table Sort by key: None

Purchases

The screenshot shows the MySQL Workbench interface with the following details:

- Server:** sql211.infinityfree.com
- Database:** if0_39346865_inventory_management_system
- Table:** purchases
- SQL Query:** SELECT * FROM `purchases`
- Table Headers:** id, supplier_id, date, purchase_no, status, total_amount, created_by, updated_by, created_at, updated_at
- Data Row:** id: 1, supplier_id: 2, date: 2025-07-01 00:00:00, purchase_no: PRS-000001, status: 1 (Pending, 1=Approved), total_amount: 0, created_by: 2, updated_by: 2, created_at: 2025-07-01 18:47:55, updated_at: 2025-07-01 19:01:08
- Operations:** Edit, Copy, Delete, Export
- Query results operations:** Print, Copy to clipboard, Export, Display chart, Create view

Units

The screenshot shows the MySQL Workbench interface with the following details:

- Server:** sql211.infinityfree.com
- Database:** if0_39346865_inventory_management_system
- Table:** purchases
- SQL Query:** SELECT * FROM `purchases`
- Table Headers:** id, supplier_id, date, purchase_no, status, total_amount, created_by, updated_by, created_at, updated_at
- Data Row:** id: 1, supplier_id: 2, date: 2025-07-01 00:00:00, purchase_no: PRS-000001, status: 1 (Pending, 1=Approved), total_amount: 0, created_by: 2, updated_by: 2, created_at: 2025-07-01 18:47:55, updated_at: 2025-07-01 19:01:08
- Operations:** Edit, Copy, Delete, Export
- Query results operations:** Print, Copy to clipboard, Export, Display chart, Create view

Deployment Links:

Website Link: <https://inventory-management-system.wuaze.com/dashboard>

MySQL Database: https://php-myadmin.net/db_structure.php?db=if0_39346865_inventory_management_system

Admin Panel: https://dash.infinityfree.com/accounts/if0_39346865

Admin Credentials: USERNAME if0_39346865 | Password was744012345

Github.com

<https://github.com/waseem44049/inventory-management-system>

The screenshot shows the GitHub repository page for 'waseem44049/inventory-management-system'. The repository is public and contains 5 commits. The code structure includes 'app', 'bootstrap', 'config', 'database', 'lang/vendor/filament-shield', 'public', 'resources', 'routes', 'storage', 'tests', '.editorconfig', '.env.example', '.gitattributes', '.gitignore', 'CODE_OF_CONDUCT.md', 'LICENSE', 'README.md', 'artisan', 'composer.json', 'composer.lock', 'package-lock.json', 'package.json', 'phpunit.xml', and 'vite.config.js'. The repository has 0 stars, 0 forks, and 0 watching. It features no releases or packages. The languages used are CSS (38.5%), JavaScript (25.4%), PHP (18.2%), and Blade (17.9%). Suggested workflows include Datadog Synthetics, Gulp, and SLSA Generic generator.

Conclusion

The Inventory Management System was successfully designed and implemented using Laravel 12 and MySQL. It streamlines key business operations such as product management, purchases, sales, and reporting. By using modern web technologies and best practices, the system provides an efficient, user-friendly, and scalable solution for small to medium-sized businesses.

Deployment on **InfinityFree** with a cloud-hosted MySQL database demonstrates the system's flexibility for real-world usage. Overall, this project improves inventory visibility, reduces manual effort, and ensures data accuracy — helping businesses manage their stock and orders effectively.

REFERENCES:

- GITHUB: <https://github.com/fajarghifar/inventory-management-system>
- YOUTUBE: https://www.youtube.com/watch?v=jk8L4_Wx40U
- DEEPSEEK: <https://chat.deepseek.com/a/chat/s/a015c34d-5878-4c89-9cb1-ecd7c1ff4e56>
- Chatgpt: <https://chatgpt.com/c/686caf0d-040c-8003-a9fc-7925c299c3fc>