



**UNIVERSITI TEKNOLOGI MARA**  
**COLLEGE OF COMPUTING, INFORMATICS AND MATHEMATIC**  
INFORMATION SCIENCE STUDIES

IMS560: ADVANCED DATABASE MANAGEMENT SYSTEM

**GROUP ASSIGNMENT:**

DATABASE DESIGN DOCUMENTATION AND DEVELOPMENT

**PREPARED BY :**

AHMAD NAQIB BIN SOFIAN (2023214202)  
AIMAN MUSTAQIM AMILIN BIN ZAINUDDIN (2023406952)  
MARK SCHMEICHEL ANAK JUAN (2023285846)

**CLASS :**

D1CDIM262/4A

**PREPARED FOR:**

TS. DR. MOHAMAD RAHIMI BIN MOHAMAD ROSMAN

**DATE OF SUBMISSION :**

17 JULY 2025

### **ACKNOWLEDGEMENT**

First, I want to thank Ts. Dr. Mohamad Rahimi bin Mohamad Rosman, my lecturer for Advanced Database Management System (IMS560), for his un interrupting guidance, support, and motivation during the session of this course. His valuable feedback and comments have helped us a lot in accomplishing of this project.

And I want to thank my classmates and group members who work to get the best of every meeting we've had and without them, the learning wouldn't be as fun and meaningful as it was for me.

Finally, I would like to thank Universiti Teknologi MARA (UiTM) for the facilities and space to grow academically and develop the project. Thank you.

## **ABSTRACT**

This project is engineered to develop an online mushroom booking system for Mushromiee to assist in digital sales and operational workflow management. The system combines its major functions of user registration, product fires, the administration of order, answer for payment and also a feedback of buyer. With this, Mushromiee becomes more effective in customer service and up-to-date in its business transactions. The current implementation fulfils the basic needs of an e-commerce system, but needs to be extended to comply with the developing industry standards and the business scales. With strategic enhancements, Mushromiee is poised to become a sophisticated, user-friendly platform, a viable and competitive digital solution that promotes sustainability and customer satisfaction in agricultural marketplace for years to come.

**TABLE OF CONTENT**

ACKNOWLEDGEMENT.....	i
ABSTRACT.....	ii
TABLE OF CONTENT.....	iii
1.0 INTRODUCTION.....	1
1.1 Background of the Enterprise.....	1
1.2 Organizational Setting.....	1
1.3 Digital Transformation in Agricultural Business.....	2
1.4 The UiTM Machang Partnership.....	2
1.5 Vision and Mission.....	3
1.6 Academic and Commercial Significance.....	3
2.0 BUSINESS PRODUCT/SERVICES/GOOD.....	4
2.1 Overview of Mushroom Offerings.....	4
2.2 Categorization and Inventory Structure.....	4
2.3 Booking and Ordering Services.....	5
2.4 Review and Payment Features.....	5
3.0 BUSINESS ACTIVITIES/PROCESS.....	6
3.1 Operational Flow.....	6
3.2 Stakeholder Roles.....	6
3.3 Technology Integration.....	6
4.0 PROBLEM STATEMENT.....	7
5.0 OBJECTIVES.....	8
5.1 Operational Efficiency.....	8
5.2 Real-Time Inventory Management.....	8
5.3 Improved Customer Experience.....	8
5.4 Improved Customer Experience.....	8
5.5 Scalability and Flexibility.....	8
5.6 Institutional Collaboration.....	8
6.0 BUSINESS RULES.....	10
7.0 ENTITY RELATIONSHIP DIAGRAM (ERD).....	12
8.0 CONCEPTUAL ERD.....	14

# DATABASE DESIGN DOCUMENTATION AND DEVELOPMENT

9.0 NORMALIZATION.....	15
9.1 Unnormalized Form (UNF).....	15
9.2 First Normal Form (1NF).....	16
9.3 Second Normal Form (2NF).....	17
9.4 Third Normal Form (3NF).....	18
10.0 RELATIONAL DIAGRAM.....	19
11.0 DATA DICTIONARY.....	20
12.0 LIMITATION & FUTURE ENHANCEMENT.....	22
13.0 CONCLUSION.....	23
14.0 REFERENCES.....	24
15.0 APPENDICIES.....	25

## 1.0 INTRODUCTION TO BUSINESS

### 1.1 Background of the Enterprise

The agricultural sector has for decades served as a significant contributor to Malaysia's economy, especially in its rural and semi-urban areas. In recent years, mushroom cultivation has found a foothold in this sector as high-potential market segment for reasons of short growth cycles, little space needs, and growing interest of health-aware consumers. In response to this trend, some enterprising staff members started Mushromiee, a mushroom booking company which works directly with Universiti Teknologi MARA (UiTM) Machang. Unlike the student-led or faculty-administered projects, Mushromiee is a fully-fledged UiTM staff owned and managed project, with access to university infrastructure and agricultural space as a formalised partnership model.

Established in 2022, Mushromiee started as an experimental project to commercialize organic mushroom farming, as well as to explore digital agri-business. The partnership with UiTM Machang allows the company to cultivate mushrooms within the plantation site of the university. This special relationship lets Mushromiee draw on the power of academia conceived of ideal ownership and the control rights. It also demonstrates a model of "progressive" engagement between the public and private sectors in Malaysia's agri-entrepreneurship ecosystem.

### 1.2 Organizational Setting

Mushromiee sits at the crossroads of enterprise innovation and agriculture. It uses the mushroom cultivation facilities available at UiTM Machang campus; i.e. consist of grow houses, storage ampoules, and environmental controlled incubation room. Access to these facilities is governed by a formal access agreement with the university in order to formalize the interest of both parties and to maintain a separation of institutional roles and private business.

Organisationally Mushromiee is a micro-enterprise with defined operational roles as a Production Team deals with the cultivation, harvesting and quality control of mushrooms, a Sales & Logistics Team, which administers the booking of deliveries and handles customer service and a Digital Systems Team, responsible for building and maintaining the online booking platform, inventory databases, and admin dashboards.

This slim model enables Mushromiee to join hands with agricultural experience and take care of its digital system management, harmoniously connecting cultivation cycles with the client-fulfillment while keeping inventory in control.

### **1.3 Digital Transformation in Agricultural Business**

At the core of Mushromiee's business operations is its proprietary online mushroom booking system. Written in HTML, PHP and MySQL, it functions as a client facing front end, as well as an internal management system. You can look up what kinds of mushrooms are available, book for a future quantity of something, get it delivered or arrange for a drop-off all on a neat, user-friendly website. Pamme offers a backend dashboard for admin to make bookings, approve orders, adjust available stock and generate reports in real-time.

This system not only adds on more convenience to buyer but also eliminates human error on manual operation, improves inventory control, and build up reliable & stable database for long term business analysis. In this sense, Mushromiee is positioned within a wider movement toward digital agriculture, where information technology is used to improve the efficiency of production, marketing, and distribution. It is as well in line with Malaysia national aspiration for modernization of small and medium agri-businesses through technology exploitation.

### **1.4 The UiTM Machang Partnership**

The partnership between Mushromiee and UiTM Machang is a synergistic partnership. For a university, hosting private ventures on the campus offers students and researchers with living examples of sustainable agricultural business models, in addition to agricultural planning, to the integration of technology and so on. The collaboration offers Mushromiee access to expertly-managed cultivation facilities, technical agricultural advice and an enhanced reputation in the community.

It should be noted Mushromiee is not a university-owned or faculty-directed business and remains completely independent in its management, staff, financial, and business practice decisions. But in terms of its physical and institutional closeness to UiTM Machang, there are possibilities for real cooperation and real impact on the community and mutual learning. It's a model that's an entirely new

form of of public-private symbiosis that serves education, entrepreneurship and the local food system.

### **1.5 Vision and Mission**

Mushromiee aspires to be the key community-embedded agri-tech company in East Coast of the Peninsula Malaysia. The future outlook of the company is to widen business territory outside UiTM Machang and to be a name known by the regional market, as well as the enhancement of the company image for good quality, freshness and responsible production for the environment. The company is dedicated to utilizing data-driven decision making in their cultivation & booking systems to aid in company growth and customer satisfaction.

The mission includes to bring organically grown, superior-quality mushrooms to the people with an easy online btn system, to infuse technology into every part of the mushroom supply chain — from seeding to sales, to foster sustainable agriculture hand in hand with education institutions such as UiTM and as a pilot, to demonstrate for universities' staff and local farming community, the prototype example of digital-enabled agri-entrepreneurship.

### **1.6 Academic and Commercial Significance**

From a wider angle, Mushromiee is also a great reference when we talk of information systems in agrobased micro enterprises. But its creation covers such topics as relational database design, ER modeling, normalization, data integrity, and UI development, all with the scope of an actual, working business. That's part of what makes it such a fun case study for academic examination, whether you're looking at it in database management or entrepreneurial studies or into system development life cycles (SDLC).

And the company provides valuable lessons in innovation, scaling and adoptability. Its combination of physical agricultural production/processing, digital system design and local distribution network is a powerful model for how even small scale enterprises can embrace digital competitiveness. This is especially true in a post-pandemic world where digital inclusion and food security are paramount concerns on a global scale.



## 2.0 BUSINESS PRODUCT/GOOD/SERVICES

### 2.1 Overview of Mushroom Offerings

Mushroomiee's goal is to provide a wide range of high quality mushrooms that have been organically cultivated for different customer base. Edible mushrooms Its products are high-quality edible mushrooms, all grown under the best available control to guarantee : - Freshness, - Safe of usage, - Pure taste arena. Some of the prominent varieties of mushrooms that are offered, through a booking platform, by Mushroomiee include:

- i. Oyster Mushrooms (*Pleurotus ostreatus*) – These light-textured, mild-flavored mushrooms are easy to prepare and are widely used by home cooks and professional chefs.
- ii. Shiitake Mushrooms (*Lentinula edodes*) – valued for their strong umami taste and health benefits, shiitake mushrooms are frequently used in Asian foods.
- iii. White Button Mushrooms (*Agaricus bisporus*) – The world's most frequently eaten mushroom, it works for most recipes.
- iv. Enoki Mushrooms (*Flammulina velutipes*) – Known for their slim stalk and crunchy texture, these mushrooms are popular in soups and salads.
- v. Lion's Mane Mushrooms (*Hericium erinaceus*) – These mushrooms are coveted by health-conscious consumers and the functional food market for their neurological benefits.

These mushroom products are offered through both pre-harvest and post-harvest bookings, enabling buyers to book their desired quantity and variety in advance.

### 2.2 Categorization and Inventory Structure

When you are being motivated to do something, don't you also feel great? Mushroomiee has a variety of categories of products, categorising everything you might see to enhance convenience and provide a more efficient way for your inventory planning. The product entries in the database include names and text descriptions along with a price and image to aid users in visually recognizing

mushrooms while browsing. Products are distinguished by a product ID and administered by products table.

From the backend of the system administrators can real-time updates on product availability, stock levels and seat bookings and the customers are able to view this information before booking.

### **2.3 Booking and Ordering Services**

Consumers can view products and order on digital. When an order is created, it gets saved to the orders table and is associated to a user by the user\_id. There can be many items in an order, which are stored in orders\_items table. Each item of the order references a product, a quantity and a price. This results in a versatile and easy to grow transaction system.

### **2.4 Review and Payment Features**

Mushromiee's system also supports:

User Reviews - In the reviews table, so users can provide feedback on an individual mushroom product. Reviews are associated with users and products.

Payment (Digital) Tracked in payments table this will hold the order\_id, mode(cash, etc), amount, status of the payment and timestamps for monitoring.

This organized fusion of reviews along with payment systems is beneficial to the customers as well as to owners.

### 3.0 BUSINESS ACTIVITIES/PROCESS

#### 3.1 Operational Flow

Ordering and inventory of mushroom supplies is coordinated with Mushromiee's web and app-based services. The chain of events in the business process can be described as follows:

- i. User Registration and Login: Users register and log in with the users table.
- ii. Browsing: Mushrooms available can be accessed from the products table.
- iii. Order Generation: Customers choose the product, quantity, and so on. These are saved in order\_items table referencing order entry.
- iv. Order Processing : Adms can view and manage the order from orders table, they can updated the order status as 'Pending' or 'Success'.
- v. Updating Payment: Once the payment will be processed, Insert it in the payment table.
- vi. Customer Ratings: Customers can rate products after they have been shipped to them.

#### 3.2 Stakeholder Roles

The system is for customers (saved in users) to order and track. Admins process orders, add new listings, and verify payments. Review Contributors send feedback through the reviews entity.

#### 3.3 Technology Integration

Mushromiee system adopts a relational database model as follows:

- users, products, orders, order\_items, payment and reviews are independent tables but related to each other.
- Data integrity is being taken care of by MySQL.
- PHP for the user interaction and the CRUD operations

### 4.0 PROBLEM STATEMENT

Although there has been an upward trend in demand for locally produced fresh mushrooms in the region, Mushromiee had faced some operational issues initially as a result of the absence of a centralised and automated management system. Prior to the introduction of its online booking system, the company took orders from its customers informally, over the phone, WhatsApp, and the manual notebooks. These previous methods have caused several problems:

- i. Data Discrepancies: Many orders were not in the system, there were also duplicates and errors in order logging through a manual process.
- ii. Mismatched inventory: Stock was mistakenly claimed during ordering process, resulting in delays or cancelled orders.
- iii. Communication: People couldn't get confirmations updated information on deliveries.
- iv. Limited Visibility into Customers: In the absence of a database, customer patterns could not be followed or their purchase history reviewed to determine feedback.
- v. Not Easily Scalable Business was not scalable, the business cannot scale to meet increasing demand, or process bulk order though administrative needs to be done by admin.

These pain points underscored that a comprehensive, integrated information system was needed which would improve operational efficiencies, enable scalability and enhance service quality. Migration onto a database-driven platform was therefore necessary for the sustainability and expansion of Mushromiee.

### 5.0 OBJECTIVE

The main goal of Mushromiee's system development endeavor is to implement a single, digital system that can streamline and improve all operations within the business - from production tracking to customer management. The specific aims are:

#### 5.1 Operational Efficiency

Automate the process of booking manually, eliminating errors, reducing process time and improving resource utilization.

#### 5.2 Real-Time Inventory Management

Now to develop a real-time system, we can monitor the amount of stock we have by a pair of tweezers literally and not over book.

#### 5.3 Improved Customer Experience

To provide a user-friendly online interface for the convenience of customers who might like to browse items, order them, and also get updated.

#### 5.4 Data Centralization

To consolidate all business data — while keeping all customer profiles, booking history, and product inventory in one safe, searchable data source that is also optimized for analytics and reporting.

#### 5.5 Scalability and Flexibility

To ensure the system architecture can scale as the business grows, when new product lines launch, when you add more user roles, or when you start to integrate real-time shipping or payment gateway modules.

#### 5.6 Institutional Collaboration

To be in line with the larger preventive innovation ecosystem of UiTM Machang by illustrating an exemplary case of staff-driven digital entrepreneurship in agriculture.

In partnership, all of these objectives change Mushromiee into the kind of data-efficient and modern business that can meet market needs and deliver business with operational excellence.

### 6.0 BUSINESS RULES

Business rule is the important thing when we want to design a good database system. It is like the rule that control how the data being store, use and manage in the system. Every business have their own rule that follow the real activity of their work. For example, if customer must register first before make order, that one is already a business rule. So when we create a system like mushroom stall ordering app, we need to write down all the rule first so we can create the table, relationship, and function that follow the process. Business rules help to make sure that all the data in the system is correct and not against the process. It also help to make the database more clean and easy to understand. Without clear business rules, the system might have wrong data or cannot function properly. That why, before do anything, we need to list all the rule that happen in the real business process. Then we use that rule to build our ERD, design the table, and write the PHP code correctly. In this mushroom business, the rule include things like how user make order, when payment happen, how admin update status and how review is save. All this are important to make the system work smooth and correct.

1. **One user can make many orders, but each order belong to one user only.**

This relationship is shown in ERD as one-to-many between user and order. System must keep track which user make the order and not allow order without user.

2. **Each order can have many products, but one product can be in many orders.**

The order\_items table in ERD show many-to-many relationship between orders and products. This allow one order to include multiple items and one product to appear in many orders.

3. **Each payment is connected to one order only.**

Payment entity have foreign key to order. This mean every payment must belong to one order and system cannot have same payment used for more than one order.

4. **One product can have many review, but review must belong to one product.**

In ERD, product and review is one-to-many. So, user can give many review for different product, but every review must be linked to exact one product.

5. **A user can give many review, but review must come from registered user.**

The ERD show that review table have user\_id as foreign key. This mean system not allow anonymous review. Every review must be link to user account.

6. **Admin have access to view and manage all entity in the system.**

From ERD, role attribute in user table is used to identify admin. Admin can access product, orders, users and payment record. Normal user cannot do this action.

**7. Product cannot be ordered if it not exist in product table.**

order\_items must use existing product\_id. So if a product is deleted, the system must remove or stop order from using that ID. This prevent broken foreign key.

**8. Payment status can only be 'paid' or 'unpaid'.**

Based on ERD, payment table use enum. So, payment must follow this status. System not allow other value like "processing" or "waiting".

**9. User can only place order if they already login.**

ERD show that order table is connected to user table. So, user\_id must be stored for each order. This mean only logged-in user can make order. Guest is not allowed.

**10. Each order have one or more order item, and order item must belong to one order.**

Order and order\_items have one-to-many relationship. This rule make sure every order have detail product list, and every item must be inside one valid order.



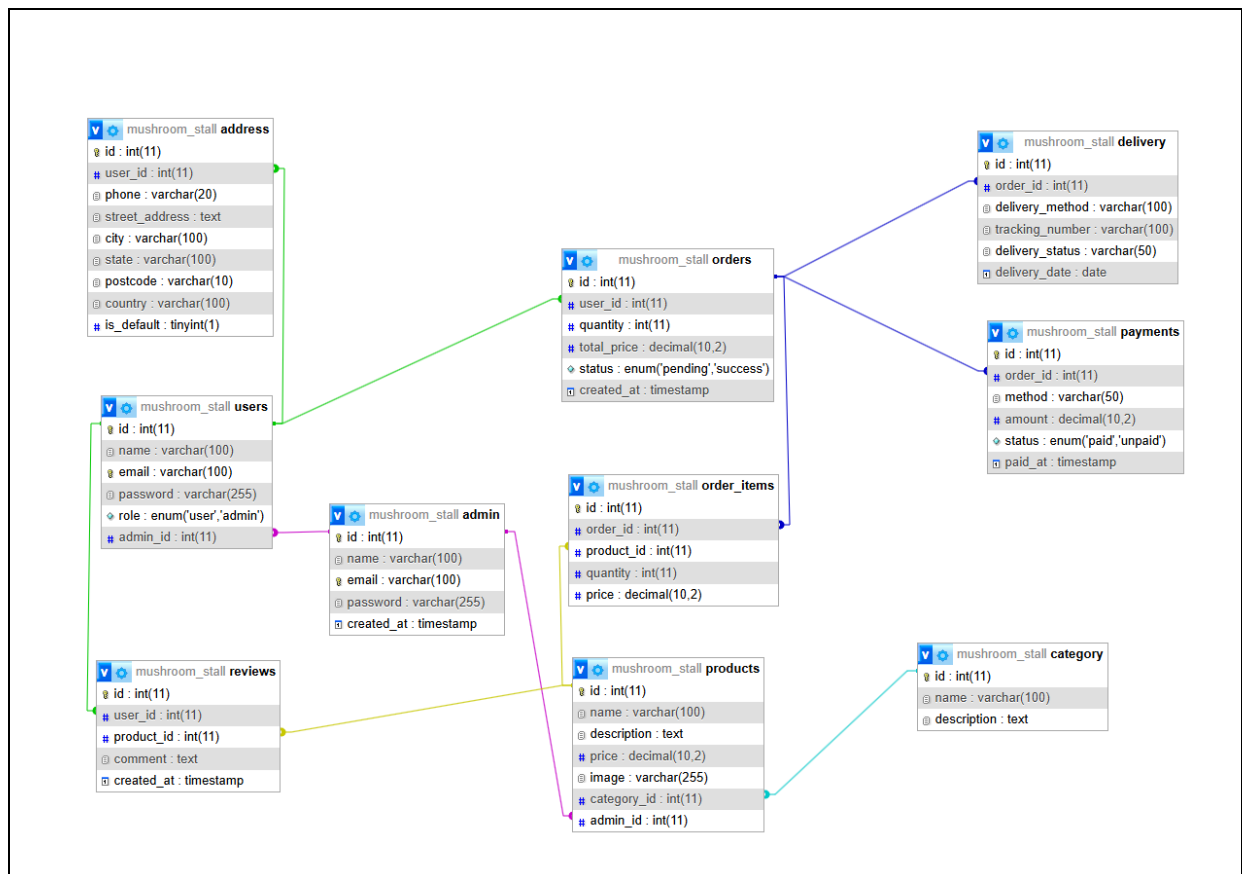
## 7.0 ENTITY RELATIONSHIP DIAGRAM (ERD)

Entity Relationship Diagram or we call ERD is one of the important thing when we want to build database system. It show how the data is connected to each other. ERD use symbol like box for entity, diamond for relationship and line to connect between them. Entity is like table in database, example of entity is user, product, or order. Inside the entity, we have attribute like user name, user email, or order date. Relationship show how two entity link together, like one user can make many order. This is very helpful because it help us understand how the data flow and how we want to design the table later. Before we create real database in phpmyadmin, we need to draw ERD first to make sure we understand the structure. In our mushroom stall system, we can see user make order, order have item, user also can give review and admin will manage the product. All of this we can show in ERD. If we do not use ERD, maybe we will forget the relationship or make wrong table. That's why ERD is very useful and important to create system with database.

Entity Name	Attributes	Relationship	Explanation
Users	user_id, name, email, password, role	One-to-Many with orders, reviews	One user can make many orders and write many reviews. Each order and review must belong to one user.
Products	product_id, name, description, price, image	One-to-Many with order_items, reviews	A product can appear in many orders and can have many reviews from users.
Orders	order_id, user_id, total_price, status, created_at	Many-to-One with users and One-to-Many with order_items, payments	Each order is made by one user. One order can have many items and one payment.
Order_items	id, order_id, product_id, quantity, price	Many-to-One with orders and products	This table is for the items inside the order. It connects each order to the product with quantity and price.

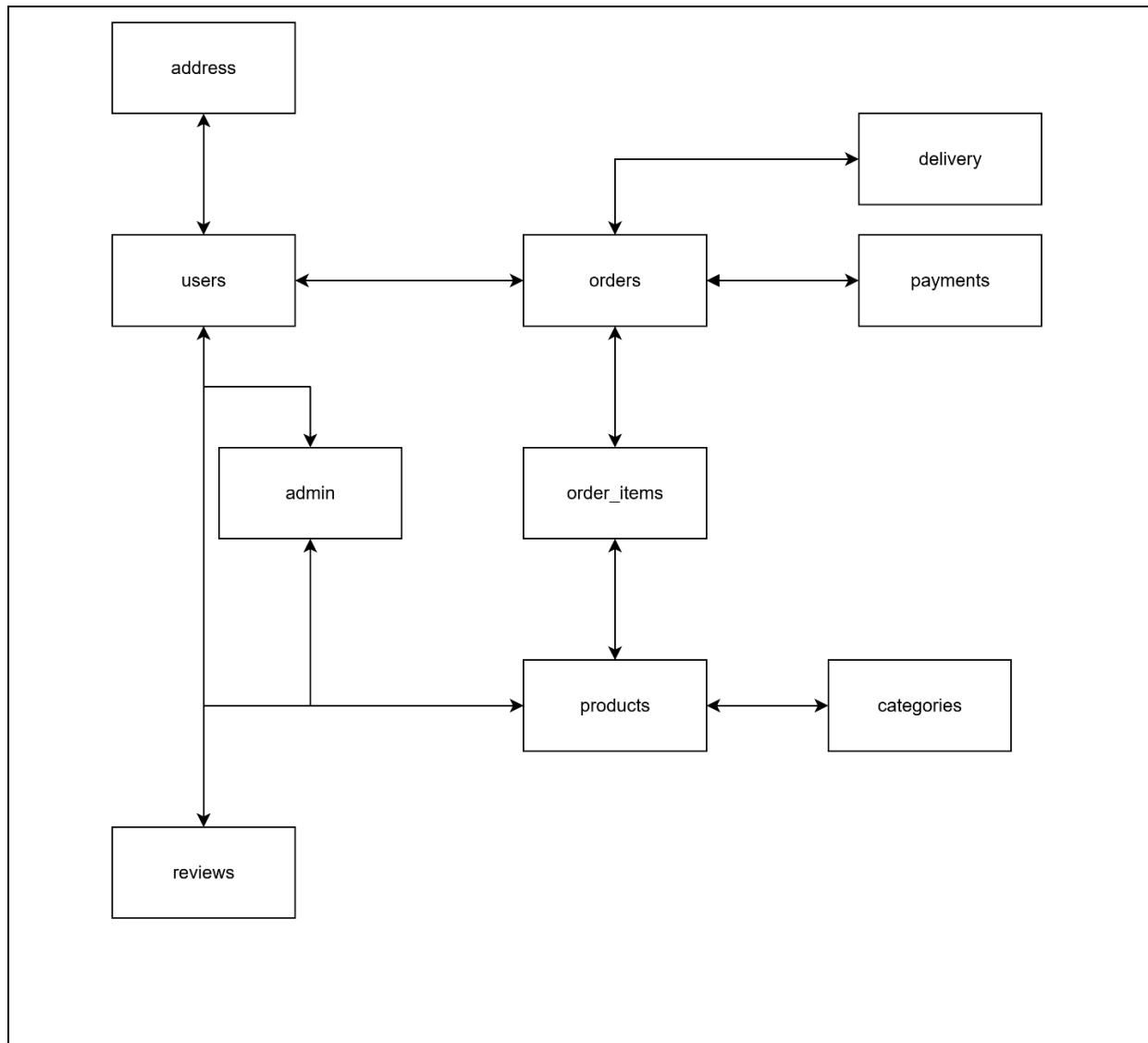
# DATABASE DESIGN DOCUMENTATION AND DEVELOPMENT

Payments	id, order_id, method, amount, status, paid_at	One-to-One with orders	One payment only for one order. Cannot share between orders.
Reviews	id, user_id, product_id, comment, created_at	Many-to-One with users and products	One user can write many reviews. One product can receive many reviews. Each review must link to one user and one product.



## 8.0 CONCEPTUAL ENTITY RELATIONSHIP DIAGRAM

The conceptual entities relationship diagram is one of the important part in database design. It show the connection between entities in a simple and clear way. This diagram help to understand how the data is related to each other before create the actual database. For example, it can show that one customer can make many orders, or one product can belong to one category. This is useful for planning and make sure the system follow the right structure. Even it just a early design, it already give the idea how the system work.



## 9.0 NORMALIZATION

In database, normalization is a process to make sure that the data in the table is organised well and no repeating or duplicate data is store. It help us to avoid problem like redundancy and make the data easier to maintain. Normalization also help to save storage and improve the performance of database. There are several stages of normalization, usually start from unnormalized form (UNF), then continue to 1NF, 2NF and 3NF. Each of this level have their own purpose and function to clean the data. In our mushroom stall app, normalization is important to manage order, product, user and review data so it can be store correctly. If the database not normalized, it may cause confusing data and mistake when admin or user try to access or update the information.

### 9.1 Unnormalized Form (UNF)

In unnormalized form or UNF, the data is stored in one big table that contain everything together. It not follow any rule of database normalization. This table maybe have multiple values in one column, repeating groups, and a lot of duplicate data. For example, in my mushroom stall project, the unnormalized table is like this.

order\_id →

user\_id, username, user\_email,  
product\_id, product\_name, product\_desc, product\_price, product\_image,  
quantity, total\_price,  
payment\_method, payment\_amount, payment\_status, paid\_at,  
order\_status, created\_at,  
review\_comment, review\_date

This table have many problem. First, if one order have two product, then product column need to be repeated, and so the user info and order info also repeated again. Second, if one product like "Enoki Mushroom" appear in 10 orders, the name, description, and price is repeated 10 times. It make the table bigger and harder to update. If the product price change, we must update in every row. Also, some value is not atomic. For example, maybe in review\_comment, user write multiple comment in same field. This make it not follow database rules.

UNF is common mistake when people first design database without thinking about normalization. It is fast to build, but slow to manage later. If something need to change, we might forget where all the data is repeated. That why we need to start

normalization process from here. In next step, we go to First Normal Form to fix this problem. We will break repeating groups and make sure all the data are in correct format.

### 9.2 First Normal Form (1NF)

First Normal Form or 1NF is the first rule we follow when fixing unnormalized data. In 1NF, we make sure that each column have only atomic values. That means no list or group of data inside one field. Also, each record must be unique and every field should have same type of value. For example, we should not have a column that store multiple product\_id or multiple review in one row. Every product in an order must have its own row.

In our mushroom stall project, we break down the UNF table into smaller pieces. We create separate tables like users, products, orders, order\_items, payments, and reviews. In users, we only store user\_id, name, email, and password. All the user info store only once. Same with products, we keep name, description, price, and image one time. Then, we separate the items ordered using order\_items table where we store order\_id, product\_id, and quantity. This allow one order to have many products without repeating the entire order or user info.

By doing this, we remove repeating groups and make sure every data is stored properly. Now, if we want to add a new product to an order, we just add one row in order\_items, not the whole order again. Also, if user submit multiple reviews, each one have their own row. 1NF help make data easier to manage, and it is very important first step before going to next level which is 2NF. It solve many problem, but still some issue exist like partial dependency which we fix in 2NF.

Example:

users

user\_id → username, user\_email

products

product\_id → product\_name, product\_desc, product\_price, product\_image

orders

order\_id → user\_id, total\_price, order\_status, created\_at

order\_items

$\text{order\_item\_id} \rightarrow \text{order\_id}, \text{product\_id}, \text{quantity}, \text{product\_price}$

payments

$\text{payment\_id} \rightarrow \text{order\_id}, \text{payment\_method}, \text{payment\_amount}, \text{payment\_status}, \text{paid\_at}$

reviews

$\text{review\_id} \rightarrow \text{user\_id}, \text{product\_id}, \text{review\_comment}, \text{review\_date}$

### 9.3 Second Normal Form (2NF)

Second Normal Form (2NF) improve on 1NF by removing partial dependency. Partial dependency happen when a table have a composite key (like `order_id` and `product_id` together), and some column only depend on part of the key. In 2NF, we want every non-key column to depend on the whole primary key, not just a part.

Let say we have `order_items` table with this structure:

$\text{order\_id} + \text{product\_id} \rightarrow \text{quantity}, \text{product\_name}, \text{product\_price}, \text{product\_image}$

Here, `product_name`, `product_price`, and `product_image` only depend on `product_id`, not on `order_id`. So, they are partial dependent. If we keep them in `order_items`, we break the 2NF rule. The correct way is to move all product info into the `products` table, and only store `product_id` in `order_items`. Now the table become:

$\text{order\_item\_id} \rightarrow \text{order\_id}, \text{product\_id}, \text{quantity}, \text{price}$

Same thing happen with user data. If `orders` table store `username`, `user_email`, but these only depend on `user_id`, then it's also a partial dependency. We remove them and put into `users` table. `orders` only keep `user_id`, and we link using foreign key.

This step reduce the size of data and make it easier to update. For example, if product price change, we only update in `products` table, not in 10 order rows. 2NF make sure all the non-key data is stored in the right place and depend fully on the main key. After this, we still have to fix transitive dependency, which is the focus of 3NF.

### 9.4 Third Normal Form (3NF)

Third Normal Form or 3NF is the final step of basic normalization. It remove transitive dependency. This is when a non-key column depends on another non-key column. In 3NF, every non-key column must only depend on the primary key. No other connection should exist inside the table.

For example, if in the orders table we store user\_id, user\_email, and user\_name, the user\_email and user\_name depend on user\_id, not directly on order\_id. This is called transitive dependency. So, we remove them and store only user\_id in the orders table. Same thing with product\_name and product\_image in order\_items. They depend on product\_id, not on the whole order\_item\_id. So we remove them from there too. After we finish fixing all the tables, now the database become clean and easy to understand.

Example:

user\_id → name, email, password, role

product\_id → name, description, price, image

order\_id → user\_id, total\_price, status, created\_at, status\_id

order\_item\_id → order\_id, product\_id, quantity, price

payment\_id → order\_id, method, amount, status, paid\_at

review\_id → user\_id, product\_id, comment, created\_at

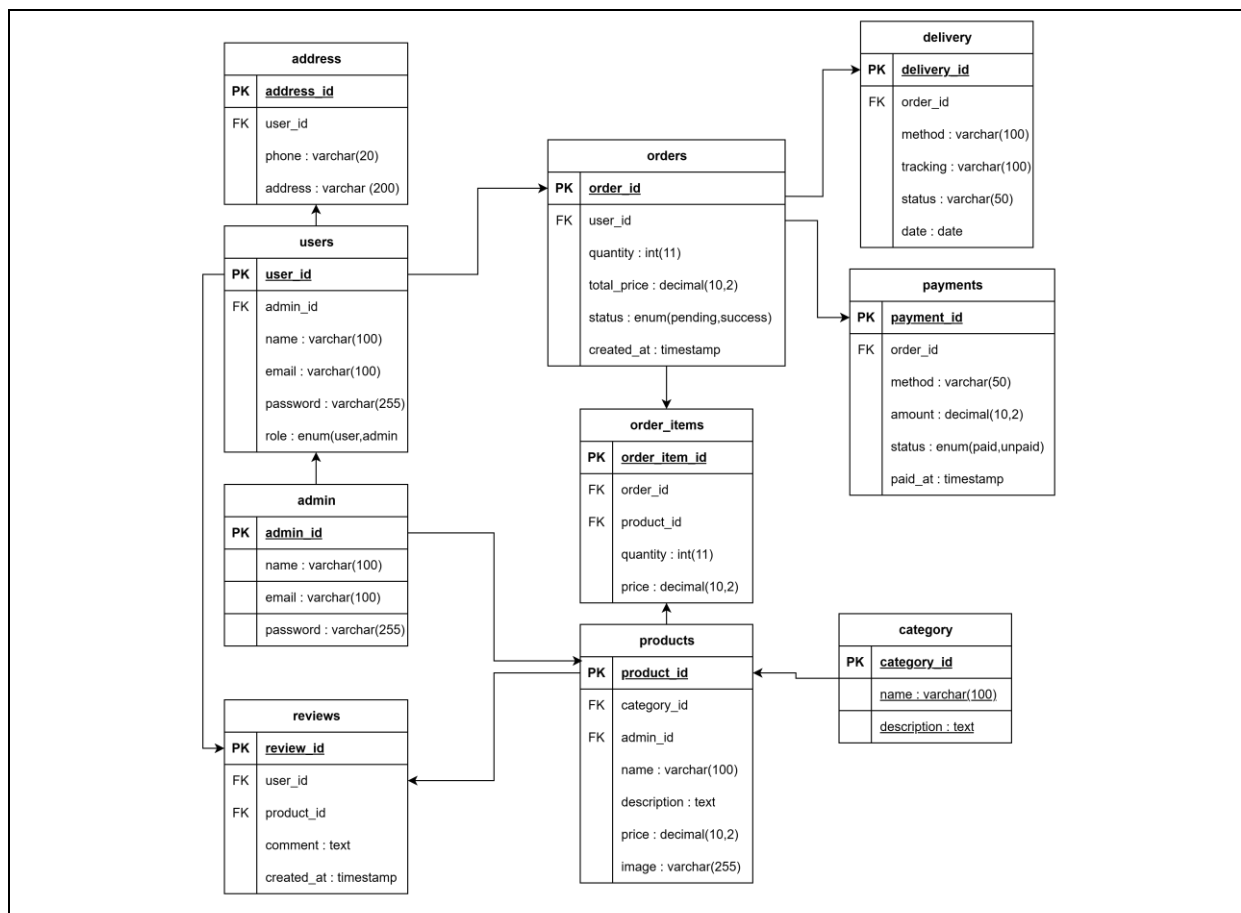
Each table store only the data it should store. All data is connected using foreign key. No more repeating values or transitive dependency. If we want to update user info, we just go to users table. If we want to see product detail, we check products table. 3NF help to keep the data integrity and make system more efficient. It is very important step for making a real-world database like this mushroom stall system. Now the data is safe, consistent, and easy to update.

## 10.0 RELATIONAL DIAGRAM

Relational diagram is one of the important part in designing database because it show how all the tables in system are connect together. It help us to understand the relationship between tables like which table have foreign key and which table is the parent table. Each table in the diagram have their own fields, and it show the primary key (PK) and also the foreign key (FK) clearly. This diagram not only help developer, but also help other team member to see how data move in the system.

In this mushroom stall database, relational diagram show 10 table including users, products, orders, reviews, payments, and more. For example, one user can make many orders, so the users table have one-to-many relationship with the orders table. Also, each order can have many items, so order\_items table is connect to both orders and products. These kind of connection is show in the diagram by line or arrow, and we can see which field is link.








Relational diagram is usually create after we design the ERD. It more focus on the real database implementation. It show what column is used to connect table, what is data type, and what table depend on another. This is very useful when we want to write SQL code, do database testing or debug some error in the system.











## 11.0 DATA DICTIONARY

Data dictionary is one of the important part in database system because it help to explain all the tables and field inside the database. It give clear meaning about what is the name of the field, what type of data is use, and what is the function of that field. For example, in mushroom stall system, we have many table like users, orders, products and more. Each table have their own column that store different type of data. With data dictionary, we can know easily which one is primary key (PK), which one is foreign key (FK), and what each column is doing in the system. This also help developer, user and admin to understand the system better without looking into the code. Sometimes, when many people are working on same project, data dictionary make it more easy to communicate and avoid mistake. It also important when we do documentation or presentation about our database. In this assignment, we will show a data dictionary for mushroom stall system that include all the important table with their field name, data type, description and key.

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(100)	utf8mb4_general_ci		No	None		
3	email 	varchar(100)	utf8mb4_general_ci		No	None		
4	password	varchar(255)	utf8mb4_general_ci		No	None		
5	role	enum('user', 'admin')	utf8mb4_general_ci		Yes	user		
users								
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	user_id 	int(11)			Yes	NULL		
3	quantity	int(11)			Yes	NULL		
4	total_price	decimal(10,2)			No	None		
5	status	enum('pending', 'success')	utf8mb4_general_ci		Yes	pending		
6	created_at	timestamp			No	current_timestamp()		
7	status_id	int(11)			Yes	NULL		
orders								
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	user_id 	int(11)			Yes	NULL		
3	product_id 	int(11)			Yes	NULL		
4	comment	text	utf8mb4_general_ci		Yes	NULL		
5	created_at	timestamp			No	current_timestamp()		
reviews								

## DATABASE DESIGN DOCUMENTATION AND DEVELOPMENT

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	name	varchar(100)	utf8mb4_general_ci		No	None		
3	description	text	utf8mb4_general_ci		No	None		
4	price	decimal(10,2)			No	None		
5	image	varchar(255)	utf8mb4_general_ci		No	None		
Products								
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	order_id 	int(11)			No	None		
3	method	varchar(50)	utf8mb4_general_ci		No	None		
4	amount	decimal(10,2)			No	None		
5	status	enum('paid', 'unpaid')	utf8mb4_general_ci		Yes	unpaid		
6	paid_at	timestamp			No	current_timestamp()		ON UPDATE CURRENT_TIMESTAMP()
payments								
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	order_id 	int(11)			No	None		
3	product_id 	int(11)			No	None		
4	quantity	int(11)			No	None		
5	price	decimal(10,2)			No	None		
Order_items								

### 12.0 LIMITATION & FUTURE ENHANCEMENT

Even though the website system developed by Mushromiee already help improve many business process, there still some limitation that need to be consider for future improvement. First, the system only focus on basic function such as order, payment and review. It does not have feature like real-time stock update or auto-alert when stock finish. Admin need to manually check and update, which may cause mistake if they forget or busy. Besides that, the review system also not complete because user cannot edit or delete the review once submitted.

Also, the admin dashboard still can be enhance. For example, it can show sales statistic, popular product chart or customer feedback summary to help admin make better decision. The design of the website is simple and use Bootstrap, but still not fully responsive on some mobile phone. Some button or text may not display properly in small screen. Security aspect also limited. There is no email verification, OTP login, or password reset link, which are important for protecting user account.

For future enhancement, this website can improve by adding better user interface that is mobile friendly. It also can integrate with email system for send receipt or order status. Other suggestion is to add delivery tracking page, so user can know where their order is. Also, live chat or support form will make customer service more good. For admin, export report to PDF or Excel and monthly sales dashboard will make their job easier. Finally, add real payment gateway like FPX or e-wallet can make system more professional and ready for big scale.

### **13.0 CONCLUSION**

In conclusion, this system will be specifically designed to support Mashromie in supporting online sales and related management. It has successfully performed and established essential features such as user registration, product listing, order processing, payment and customer feedback. However, to keep up with industry standards and support future business expansion, the system needs to undergo some major improvements. With appropriate enhancements, Mashromie will grow and become a professional, user-friendly and competitive e-commerce solution that supports growth and customer satisfaction in the digital marketplace.

## 14.0 REFERENCES

Amran, N., Bahry, F. D. S., Razak, A. H. A., Muksin, M. S., & Rahaman, N. A. A. (2022). Database Conceptual Diagram Using ERD and Data Dictionary Metadata for SME Construction Business. *International Journal of Academic Research in Business and Social Sciences*, 12(10), 2985-2999.

Bootstrap. (2022). Bootstrap. Getbootstrap.com. <https://getbootstrap.com/>

Cabello-Solorzano, K., Ortigosa de Araujo, I., Peña, M., Correia, L., & J. Tallón-Ballesteros, A. (2023). The impact of data normalization on the accuracy of machine learning algorithms: A comparative analysis. *International conference on soft computing models in industrial and environmental applications*,

LucidChart. (2024) "What Is an Entity Relationship Diagram (ERD)?" Lucidchart, [www.lucidchart.com/pages/er-diagrams](https://www.lucidchart.com/pages/er-diagrams).

Rashkovits, R., & Lavy, I. (2021). Mapping common errors in entity relationship diagram design of novice designers. *International Journal of Database Management Systems*, 13(1), 1-19.

15.0 APPENDICES

User Interface:

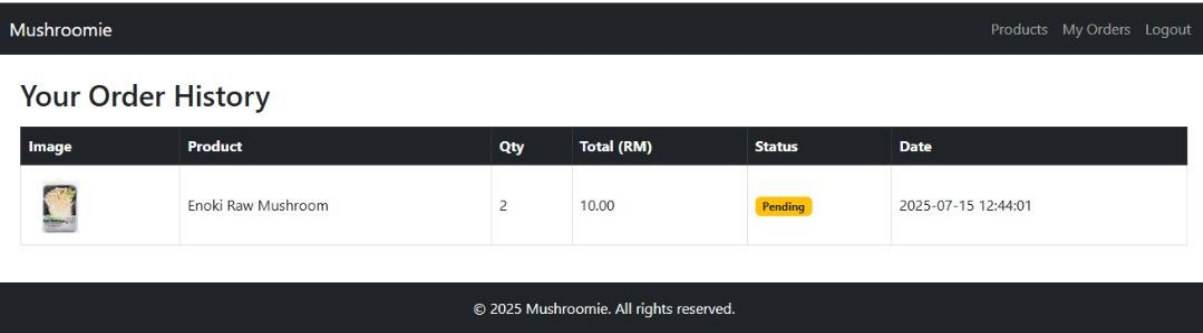


Figure 1: Order History

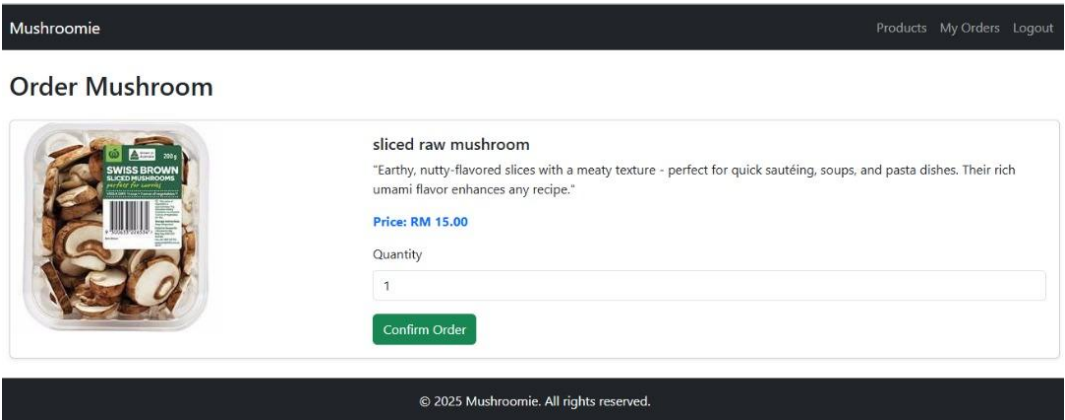


Figure 2: Order Item

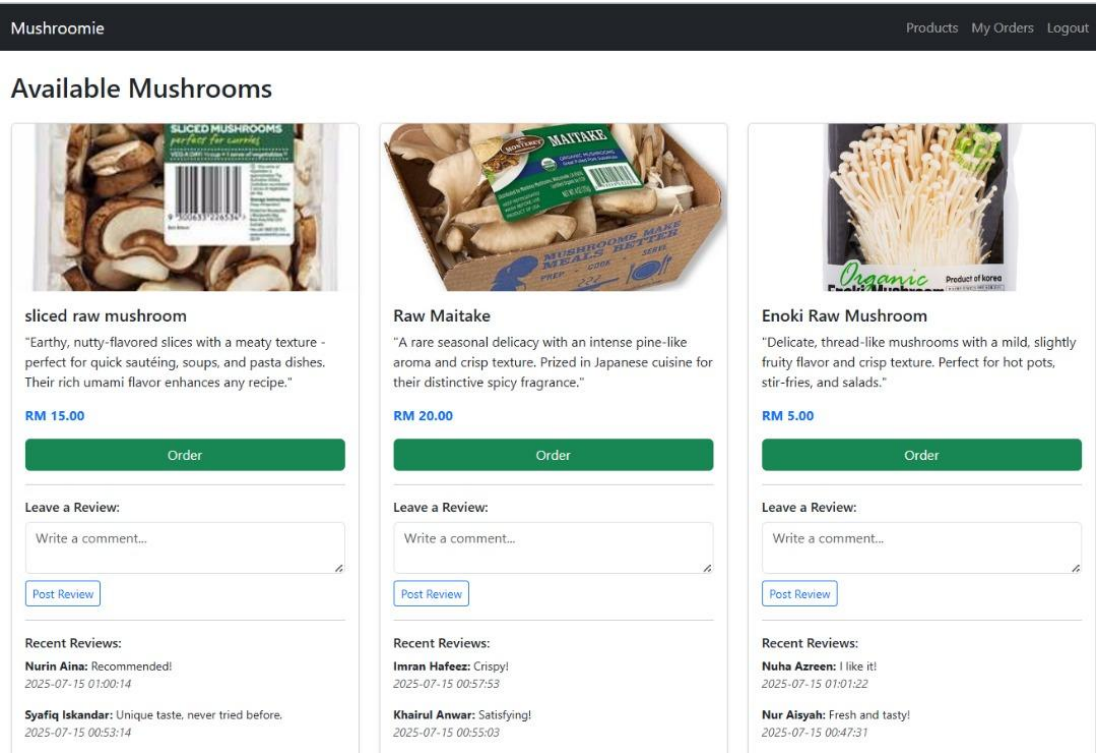


Figure 3: Menu Availabe

Admin Interface:

Mushroomie

DashboardLogout

User List

#	Name	Email	Actions
1	aiman	aiman@gmail.com	<div>EditDelete</div>
2	Ahmad Zulkifli	zulkifli@gmail.com	<div>EditDelete</div>
3	Nur Aisyah	Aisyah@gmail.com	<div>EditDelete</div>

Figure 4: User List

Mushroomie

DashboardLogout

Add Product

Name

Description

Price

Choose File

No file chosen

Add

Product List


#	Image	Name	Description	Price	Action
1		sliced raw mushroom	"Earthy, nutty-flavored slices with a meaty texture - perfect for quick sautéing, soups, and pasta dishes. Their rich umami flavor enhances any recipe."	RM 15.00	<div>EditDelete</div>

Figure 5: Product was Add

Mushroomie

DashboardLogout

Manage Orders

#	User	Items	Total Qty	Total Price	Status	Actions
1	Aisyah Humaira	Enoki Raw Mushroom (x2)	2	RM 10.00	Pending	<div>Pending</div> <div>Update</div>
2	naqib	sliced raw mushroom (x1)	1	RM 15.00	Pending	<div>Pending</div> <div>Update</div>
3	Hafiz Rahman	Fried Enoki (x1) Raw Maitake (x1) Enoki Beef Rolls (x1)	3	RM 65.00	Success	<div>Success</div> <div>Update</div>
4	Danish Haikal	Enoki Raw Mushroom (x1) sliced raw mushroom (x1)	2	RM 25.00	Pending	<div>Pending</div> <div>Update</div>
5	Zarina Bakar	Enoki Raw Mushroom (x1) Enoki Beef Rolls (x1)	2	RM 29.00	Success	<div>Success</div> <div>Update</div>
6	Alya Sofea	Mushroom Soup (x1) Enoki Beef Rolls (x1) Fried Enoki (x1)	3	RM 33.00	Pending	<div>Pending</div> <div>Update</div>

Figure 6: Admin Manage Ordersk