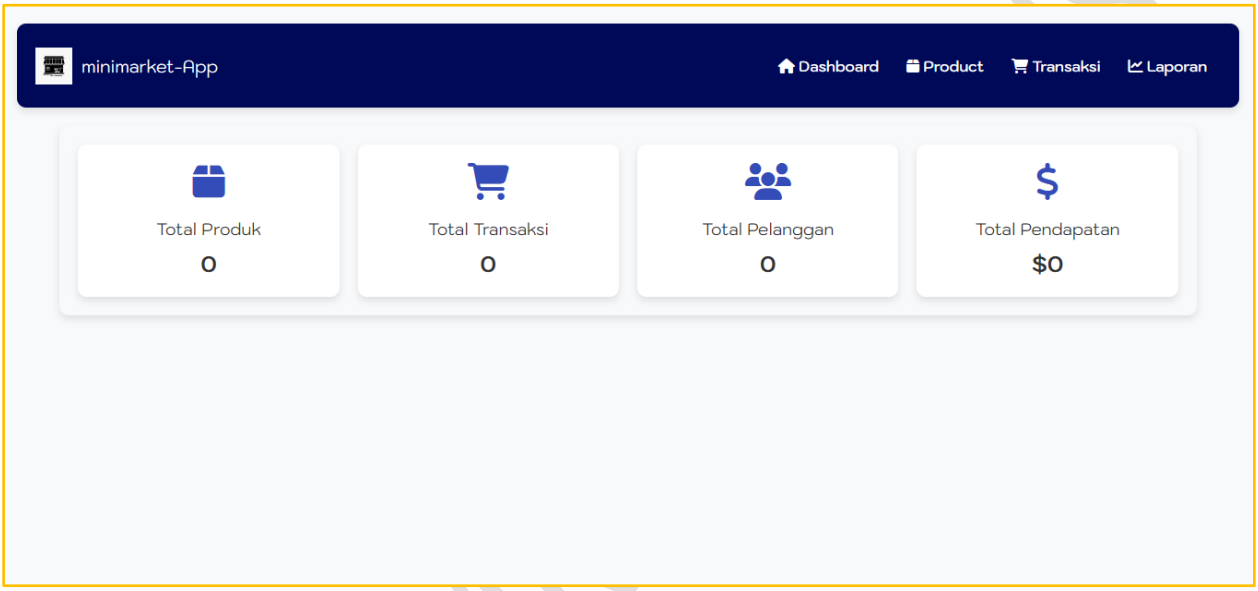
PEMBAHASAN UKK PAKET 4

APLIKASI MINIMARKET DENGAN DISKON



Rekayasa Perangkat Lunak - SMK Negeri 1 Kuwus

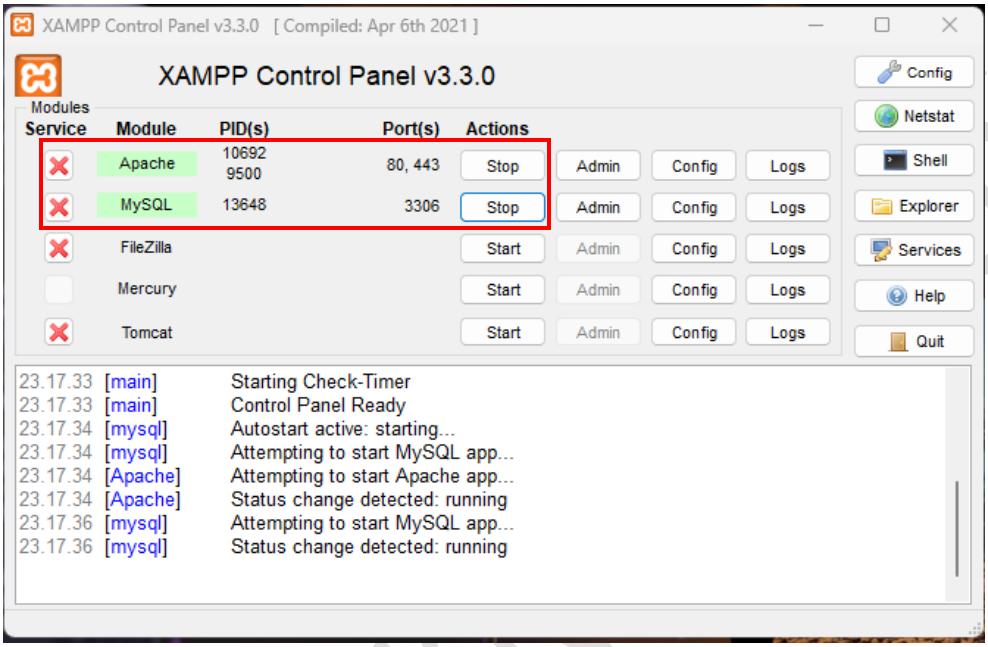
Manggarai Barat

Nusa Tenggara Timur

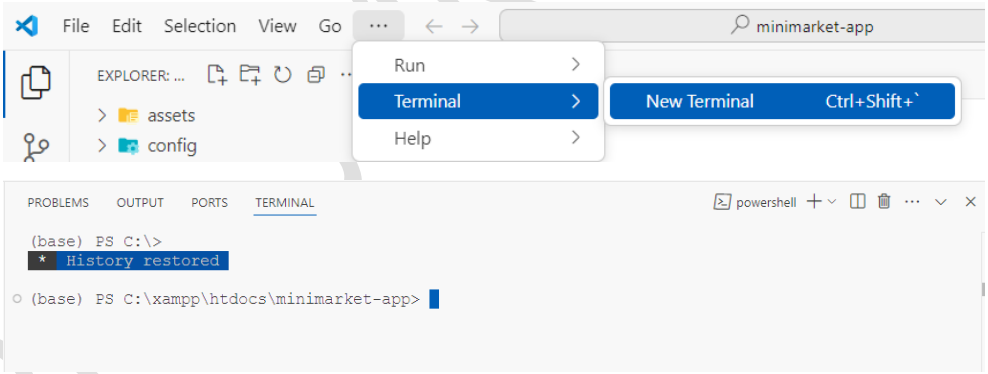
2024/2025

Menjalankan WebServer XAMPP dan Koneksi menggunakan Terminal Visual Studio Code

1. Jalankan **XAMPP** dengan mengaktifkan **Service Apcahe** dan **MSQL**



2. Buka **Visual Studio** dan Jalankan Menu **Terminal** dan Pilih **New Terminal**



3. Masuk ke direktori folder **C:\xampp\mysql\bin**

   cd ://xampp/mysql/bin

   ./mysql -u root -p

   (Muncul Password: -> **Tekan Enter**)

```
PROBLEMS  OUTPUT  PORTS  TERMINAL                                  powershell

(base) PS C:\>
*  History restored

(base) PS C:\xampp\htdocs\minimarket-app> cd C://
```

```
PROBLEMS  OUTPUT  PORTS  TERMINAL                                  mysql

PS C:\> cd xampp/mysql/bin
PS C:\xampp\mysql\bin> ./mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 9
Server version: 10.4.32-MariaDB mariadb.org binary distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

4. Tampilkan Basis Data (***Show Databases;***)

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| phpmyadmin         |
| test               |
+--------------------+
5 rows in set (0.029 sec)

MariaDB [(none)]>
```

CRUD (Create, Read, Update dan Delete) SQL (Structured Query Language) MYSQL

1. Membuat Database

Membuat Databse dengan nama **db_minimarket**

Contoh sintaks SQL

```
CREATE DATABASE db_minimarket;
```

2. Menggunakan database

Contoh sintaks SQL

```
USE db_minimarket;
```

3. Membuat Tabel

Didalam database db_minimarket terdapat dua tabel yang dapat dijelaskan sebagai berikut:

Tabel barang

Atribut	Tipe Data	Size	Keterangan
id_barang	INT	4 Byte	Primary Key, Auto Increment
nama_barang	VARCHAR(100)	100 Karakter	Nama barang
kategori	VARCHAR(50)	50 Karakter	Kategori barang (boleh kosong)
harga	DECIMAL(10,2)	5-9 Byte	Harga barang dengan 2 desimal
stok	INT	4 Byte	Jumlah stok barang
diskon	DECIMAL(5,2)	3-5 Byte	Persentase diskon (default 0)
created_at	TIMESTAMP	4 Byte	Waktu pembuatan data

Contoh sintaks SQL:

```
CREATE TABLE barang (
    id_barang INT AUTO_INCREMENT PRIMARY KEY,
    nama_barang VARCHAR(100) NOT NULL,
    kategori VARCHAR(50),
    harga DECIMAL(10,2) NOT NULL,
    stok INT NOT NULL,
    diskon DECIMAL(5,2) DEFAULT 0, -- Persentase diskon
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

Tabel transaksi

Atribut	Tipe Data	Size	Keterangan
id_transaksi	INT	4 Byte	Primary Key, Auto Increment
id_barang	INT	4 Byte	Foreign Key ke `barang.id_barang`
jumlah	INT	4 Byte	Jumlah barang yang dibeli
total_harga	DECIMAL(10,2)	5-9 Byte	Total harga dengan 2 desimal
tanggal_transaksi	TIMESTAMP	4 Byte	Waktu terjadinya transaksi

Contoh sintaks SQL:

```
CREATE TABLE transaksi (
    id_transaksi INT AUTO_INCREMENT PRIMARY KEY,
    id_barang INT NOT NULL,
    jumlah INT NOT NULL,
    total_harga DECIMAL(10,2) NOT NULL,
    tanggal_transaksi TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (id_barang) REFERENCES barang(id_barang) ON DELETE CASCADE
);
```

ON DELETE CASCADE memastikan bahwa jika barang dihapus, transaksi yang berhubungan juga ikut terhapus.

4. Melakukan CRUD SQL

a. **CREATE** - Menambah Data ke Tabel **barang**

```
INSERT INTO barang (nama_barang, kategori, harga, stok, diskon)
VALUES ('Laptop Gaming', 'Elektronik', 15000000.00, 10, 10.00);
```

Penjelasan:

Menambah satu data barang dengan nama "Laptop Gaming" di kategori "Elektronik" dengan harga 15 juta, stok 10 unit, dan diskon 10%.

b. **READ** - Melihat Data darti Tabel **Barang**

- *Melihat Semua Data*

```
SELECT * FROM barang;
```

- *Melihat Semua Data Tertentu Berdasarkan Kondisi:*

```
SELECT nama_barang, harga, stok FROM barang
WHERE kategori = 'Elektronik';
```

Penjelasan:

Menampilkan nama barang, harga, dan stok untuk barang di kategori "Elektronik".

c. **UPDATE** – Mengubah Data di Tabel **Barang**

```
UPDATE barang
SET harga = 14000000.00, diskon = 15.00
WHERE nama_barang = 'Laptop Gaming';
```

Penjelasan:

Mengubah harga menjadi 14 juta dan diskon menjadi 15% untuk barang dengan nama "Laptop Gaming".

d. **DELETE** – Menghapus Data dari Tabel **Barang**

```
DELETE FROM barang
WHERE nama_barang = 'Laptop Gaming';
```

5. Tips Tambahan

- Gunakan SELECT * FROM barang; setelah operasi **CRUD** untuk melihat hasil perubahan.
- Gunakan LIMIT untuk membatasi jumlah data yang ditampilkan, misalnya:

```
SELECT * FROM barang LIMIT 5;
```

CRUD (Create, Read, Update dan Delete)

MYSQL DAN PHP (Hypertext Preprocessor)

1. db.php (folder config)

```php
<?php
$host = "localhost";
$dbname = "db_minimarket";
$username = "root";
$password = "";

// Buat koneksi
$conn = new mysqli($host, $username, $password, $dbname);

// Cek koneksi
if ($conn->connect_error) {
    die(json_encode(["error" => "Koneksi Gagal: " . $conn->connect_error]));
}
?>
```

Script di atas adalah kode PHP yang digunakan untuk melakukan koneksi ke database MySQL dengan menggunakan objek mysqli. Berikut penjelasan rinci setiap bagian dari kode tersebut:

a) Inisialisasi Variabel Koneksi

```php
$host = "localhost";
$dbname = "db_minimarket";
$username = "root";
$password = "";
```

- **$host = "localhost";**
  Menentukan host database. Dalam kasus ini, digunakan localhost, yang berarti database berada di server yang sama dengan tempat berjalannya script PHP.
- **$dbname = "db_minimarket";**
  Nama database yang ingin dihubungkan adalah db_minimarket.

- **$username = "root";**

Nama pengguna (username) untuk mengakses database. root adalah username default di MySQL.

- **$password = "";**

Password untuk pengguna root. Dalam contoh ini, password dikosongkan (default untuk sebagian besar instalasi MySQL lokal).

b) Membuat Koneksi ke Database

```
$conn = new mysqli($host, $username, $password, $dbname);
```

- **new mysqli()**

Digunakan untuk membuat objek koneksi baru ke database MySQL.

- **$conn**

Variabel yang menyimpan objek koneksi.

- Parameter yang diberikan ke new mysqli() adalah:
  - $host → Alamat server database (localhost).
  - $username → Nama pengguna database (root).
  - $password → Password pengguna database (kosong).
  - $dbname → Nama database (db_minimarket).

Jika koneksi berhasil, $conn akan menyimpan objek koneksi. Jika gagal, akan ada error pada tahap selanjutnya.

c) Cek Koneksi

```
if ($conn->connect_error) {
    die(json_encode(["error" => "Koneksi Gagal: " . $conn->connect_error]));
}
```

- **$conn->connect_error**

Properti yang mengecek apakah terjadi kesalahan saat melakukan koneksi.

- **if ($conn->connect_error)**

Jika terdapat error, maka blok ini akan dijalankan.

- **die()**

Fungsi ini digunakan untuk menghentikan eksekusi script dan menampilkan pesan error.

- **json_encode()**

Mengubah array PHP menjadi format JSON.

Contoh output jika koneksi gagal:

```
{"error": "Koneksi Gagal: (pesan error dari MySQL)"}
```

2. create.php (Folder php-barang)

```php
<?php
//print_r($_POST);

include "../../config/db.php";

if (isset($_POST['nama_barang'], $_POST['kategori'],
$_POST['harga'], $_POST['stok'], $_POST['diskon'])) {
    $nama_barang = $_POST['nama_barang'];
    $kategori = $_POST['kategori'];
    $harga = $_POST['harga'];
    $stok = $_POST['stok'];
    $diskon = $_POST['diskon'];

    $stmt = $conn->prepare("INSERT INTO barang (nama_barang,
kategori, harga, stok, diskon) VALUES (?, ?, ?, ?, ?)");
    $stmt->bind_param("ssdii", $nama_barang, $kategori, $harga,
$stok, $diskon);

    if ($stmt->execute()) {
        echo "success";
    } else {
        echo "error";
    }
}
?>
```

Script di atas adalah kode PHP yang digunakan untuk **menyimpan data barang** ke dalam tabel barang di database MySQL menggunakan **Prepared Statement** untuk mencegah **SQL Injection**. Berikut penjelasan rinci dari setiap bagian kode:

a) Include Koneksi Database

```php
include "../../config/db.php";
```

- Digunakan untuk **menghubungkan** script ini dengan file koneksi database (db.php) yang terletak di direktori **../../config/**.

- Dengan asumsi bahwa file db.php berisi kode koneksi ke database menggunakan objek **$conn**.

b) Pengecekan Data POST

```
if (isset($_POST['nama_barang'], $_POST['kategori'],
$_POST['harga'], $_POST['stok'], $_POST['diskon'])) {
```

- **isset()** digunakan untuk memeriksa apakah semua data yang diperlukan (nama_barang, kategori, harga, stok, dan diskon) **telah dikirimkan** melalui metode **POST**.
- Script ini hanya akan dilanjutkan jika **semua** data tersedia.

c) Mengambil Data dari Form

```
$nama_barang = $_POST['nama_barang'];
$kategori = $_POST['kategori'];
$harga = $_POST['harga'];
$stok = $_POST['stok'];
$diskon = $_POST['diskon'];
```

$_POST[] digunakan untuk mengambil data dari form HTML yang dikirim menggunakan metode POST.
Data yang diambil adalah:

- **$nama_barang** → Nama barang yang akan disimpan.
- **$kategori** → Kategori barang.
- **$harga** → Harga barang.
- **$stok** → Jumlah stok barang.
- **$diskon** → Diskon yang diberikan.

d) Mempersiapkan Query (Prepared Statement)

```
$stmt = $conn->prepare("INSERT INTO barang (nama_barang,
kategori, harga, stok, diskon) VALUES (?, ?, ?, ?, ?)");
```

- Menggunakan **Prepared Statement** untuk mencegah **SQL Injection**.

$conn->prepare() mempersiapkan query INSERT untuk menambah data ke tabel barang.

- ? digunakan sebagai **placeholder** untuk nilai yang akan dimasukkan.

e) Mengikat Parameter

```
$stmt->bind_param("ssdii", $nama_barang, $kategori, $harga,
$stok, $diskon);
```

bind_param() digunakan untuk mengikat data ke placeholder ? dalam query.
Parameter pertama adalah **tipe data** dari nilai yang akan diikat:

- **s** → String (nama_barang, kategori)
- **d** → Double/Desimal (untuk harga)
- **i** → Integer (untuk stok dan diskon)

Urutan tipe data harus sesuai dengan urutan nilai yang diikat.

f) Eksekusi dan Pengecekan Hasil

```php
if ($stmt->execute()) {
  echo "success";
} else {
  echo "error";
}
```

$stmt->execute() mengeksekusi pernyataan yang telah dipersiapkan.
if ($stmt->execute()) mengecek apakah eksekusi berhasil:

- **echo "success";** → Akan menampilkan success jika data berhasil disimpan ke database.
- **echo "error";** → Akan menampilkan error jika terjadi kesalahan saat menyimpan data.

3. delete.php (Folder php-barang)

```php
<?php
include "../../config/db.php";

if (isset($_POST['id_barang'])) {
    $id_barang = $_POST['id_barang'];

    $stmt = $conn->prepare("DELETE FROM barang WHERE
id_barang=?");
    $stmt->bind_param("i", $id_barang);

    if ($stmt->execute()) {
        echo "success";
    } else {
        echo "error";
    }
}
?>
```

Script di atas adalah kode PHP yang digunakan untuk **menghapus data barang** dari tabel barang di database MySQL berdasarkan **id_barang** yang

dikirim melalui metode **POST**. Kode ini menggunakan **Prepared Statement** untuk mencegah **SQL Injection**. Berikut penjelasan rinci dari setiap bagian kode:

1. Include Koneksi Database

```
include "../../config/db.php";
```

- Menghubungkan script ini dengan file koneksi database (db.php) yang terletak di direktori ../../config/.
- Dengan asumsi bahwa file db.php berisi kode koneksi ke database menggunakan objek $conn.

2. Pengecekan Data POST

```
if (isset($_POST['id_barang'])) {
```

- isset(), digunakan untuk memeriksa apakah data id_barang telah dikirimkan melalui metode POST.
- Script ini hanya akan dilanjutkan jika id_barang tersedia.

3. Mengambil Data dari Form

```
$id_barang = $_POST['id_barang'];
```

- $_POST['id_barang'], digunakan untuk mengambil data id_barang dari form HTML yang dikirim menggunakan metode POST.
- $id_barang menyimpan ID barang yang ingin dihapus dari tabel barang.

4. Mempersiapkan Query (Prepared Statement)

```
$stmt = $conn->prepare("DELETE FROM barang WHERE id_barang=?");
```

- Menggunakan Prepared Statement untuk mencegah SQL Injection.
- $conn->prepare(), mempersiapkan query DELETE untuk menghapus data dari tabel barang berdasarkan id_barang.
- ? digunakan sebagai placeholder untuk nilai yang akan dimasukkan.

5. Mengikat Parameter

```
$stmt->bind_param("i", $id_barang);
```

- bind_param(), digunakan untuk mengikat data ke placeholder ? dalam query.
- Parameter pertama adalah tipe data dari nilai yang akan diikat:
- i → Integer (karena id_barang umumnya berupa angka).
- Parameter kedua adalah variabel yang diikat, yaitu $id_barang.

6. Eksekusi dan Pengecekan Hasil

```
if ($stmt->execute()) {
    echo "success";
} else {
    echo "error";
}
```

- `$stmt->execute()` mengeksekusi pernyataan yang telah dipersiapkan.
- `if ($stmt->execute())` mengecek apakah eksekusi berhasil:
  1. `echo "success";` → Akan menampilkan success jika data berhasil dihapus dari database.
  2. `echo "error";` → Akan menampilkan error jika terjadi kesalahan saat menghapus data.

4. read.php (Folder php-barang)

```
<?php
include "../../config/db.php";

$result = $conn->query("SELECT * FROM barang ORDER BY id_barang
DESC");

$data = [];
while ($row = $result->fetch_assoc()) {
    $data[] = $row;
}

echo json_encode($data);
?>
```

Script di atas adalah kode PHP yang digunakan untuk **mengambil data barang** dari tabel barang di database MySQL dan menampilkannya dalam format **JSON**. Kode ini cocok digunakan untuk keperluan **API** atau **AJAX** yang membutuhkan data dalam bentuk JSON. Berikut penjelasan rinci dari setiap bagian kode:

1. Include Koneksi Database

`include "../../config/db.php";`

Digunakan untuk menghubungkan script ini dengan file koneksi database (`db.php`) yang terletak di direktori `../../config/`.

Dengan asumsi bahwa file `db.php` berisi kode koneksi ke database menggunakan objek `$conn`.

2. Query untuk Mengambil Data

```
$result = $conn->query("SELECT * FROM barang ORDER BY id_barang DESC");
```

Menggunakan `$conn->query()` untuk menjalankan query SQL.
`SELECT * FROM barang` → Digunakan untuk mengambil semua data dari tabel barang.
`ORDER BY id_barang DESC` → Mengurutkan hasil berdasarkan id_barang dari yang terbaru (DESC) ke yang terlama.
Hasil query disimpan dalam variabel `$result` sebagai objek `mysqli_result`.

3. Mengubah Hasil Query menjadi Array Asosiatif

```
$data = [];
while ($row = $result->fetch_assoc()) {
    $data[] = $row;
}
```

`$data = [];` → Inisialisasi variabel $data sebagai array kosong.
`$result->fetch_assoc()` → Digunakan untuk mengambil baris data dalam bentuk array asosiatif, dimana nama kolom menjadi key dan nilainya menjadi value.
`while ($row = $result->fetch_assoc())` → Melakukan perulangan untuk mengambil setiap baris data dan menyimpannya dalam array $data.
`$data[] = $row;` → Menambah setiap baris data ke dalam array $data.
Hasil akhirnya, $data akan berisi semua data barang dalam bentuk array asosiatif.

4. Mengubah Array Menjadi JSON

```
echo json_encode($data);
```

`json_encode()` digunakan untuk mengubah array PHP ($data) menjadi format JSON.
echo digunakan untuk menampilkan JSON sebagai output.
Output ini biasanya digunakan untuk keperluan API atau AJAX di frontend.

Contoh Output JSON
Misalkan tabel barang memiliki data sebagai berikut:

id_barang	nama_barang	kategori	harga	stok	diskon
3	Sabun	Kebersihan	5000.00	100	10
2	Sikat Gigi	Perawatan	7000.00	50	5
1	Shampo	Kebersihan	15000.00	30	0

Maka output JSON yang dihasilkan adalah:


```
[
    {
        "id_barang": "3",
        "nama_barang": "Sabun",
        "kategori": "Kebersihan",
        "harga": "5000.00",
        "stok": "100",
        "diskon": "10"
    },
    {
        "id_barang": "2",
        "nama_barang": "Sikat Gigi",
        "kategori": "Perawatan",
        "harga": "7000.00",
        "stok": "50",
        "diskon": "5"
    },
    {
        "id_barang": "1",
        "nama_barang": "Shampo",
        "kategori": "Kebersihan",
        "harga": "15000.00",
        "stok": "30",
        "diskon": "0"
    }
]
```


Data diurutkan berdasarkan id_barang secara menurun (DESC) sehingga barang dengan id_barang terbesar muncul di awal.

5. update.php (Folder php-barang)

```php
<?php
include "../../config/db.php";
//print_r($_POST);

if (isset($_POST['id_barang'], $_POST['nama_barang'],
$_POST['kategori'], $_POST['harga'], $_POST['stok'],
$_POST['diskon'])) {
    $id_barang = $_POST['id_barang'];
    $nama_barang = $_POST['nama_barang'];
    $kategori = $_POST['kategori'];
    $harga = $_POST['harga'];
    $stok = $_POST['stok'];
    $diskon = $_POST['diskon'];

        $stmt = $conn->prepare("UPDATE barang SET nama_barang=?,
kategori=?, harga=?, stok=?, diskon=? WHERE id_barang=?");
        $stmt->bind_param("ssdiii", $nama_barang, $kategori,
$harga, $stok, $diskon, $id_barang);

        if ($stmt->execute()) {
            echo "success";
        } else {
            echo "error";
        }
}
?>
```

Script di atas adalah kode PHP yang digunakan untuk **mengubah data barang** di tabel barang pada database MySQL berdasarkan **id_barang** yang dikirim melalui **POST**. Kode ini menggunakan **Prepared Statement** untuk mencegah **SQL Injection**. Berikut penjelasan rinci dari setiap bagian kode:

1. Include Koneksi Database

```php
include "../../config/db.php";
```

- Menghubungkan script ini dengan file koneksi database (db.php) yang terletak di direktori ../../config/.
- Dengan asumsi bahwa file db.php berisi kode koneksi ke database menggunakan objek $conn.

2. Pengecekan Data POST

```
if (isset($_POST['id_barang'], $_POST['nama_barang'],
$_POST['kategori'], $_POST['harga'], $_POST['stok'],
$_POST['diskon'])) {
```

- isset() digunakan untuk memeriksa apakah semua data yang diperlukan telah dikirim melalui POST:
  1. id_barang
  2. nama_barang
  3. kategori
  4. harga
  5. stok
  6. diskon
- Script ini hanya akan dilanjutkan jika semua data tersedia.

3. Mengambil Data dari Form

```
$id_barang = $_POST['id_barang'];
$nama_barang = $_POST['nama_barang'];
$kategori = $_POST['kategori'];
$harga = $_POST['harga'];
$stok = $_POST['stok'];
$diskon = $_POST['diskon'];
```

- Data diambil dari form HTML yang dikirim menggunakan metode POST dan disimpan dalam variabel PHP:
  1. $id_barang → ID barang yang ingin diubah.
  2. $nama_barang → Nama baru untuk barang.
  3. $kategori → Kategori baru untuk barang.
  4. $harga → Harga baru barang.
  5. $stok → Jumlah stok yang diperbarui.
  6. $diskon → Persentase diskon yang diperbarui.

4. Mempersiapkan Query (Prepared Statement)

```
$stmt = $conn->prepare("UPDATE barang SET nama_barang=?,
kategori=?, harga=?, stok=?, diskon=? WHERE id_barang=?");
```

- Menggunakan Prepared Statement untuk mencegah SQL Injection.
- $conn->prepare() mempersiapkan query UPDATE untuk mengubah data di tabel barang.
- ? digunakan sebagai placeholder untuk nilai yang akan dimasukkan.
- Query akan memperbarui data pada kolom berikut:
  1. nama_barang
  2. kategori
  3. harga

4. stok
5. diskon

- Data akan diperbarui pada baris yang memiliki id_barang sesuai dengan nilai yang dikirimkan.

5. Mengikat Parameter

```
$stmt->bind_param("ssdiii", $nama_barang, $kategori, $harga, $stok, $diskon, $id_barang);
```

- bind_param() digunakan untuk mengikat data ke placeholder ? dalam query.
- Parameter pertama adalah tipe data dari nilai yang akan diikat:
  1. s → String (nama_barang, kategori)
  2. d → Double/Decimal (harga)
  3. i → Integer (stok, diskon, id_barang)
- Parameter kedua dan seterusnya adalah variabel yang diikat sesuai urutan:
  1. $nama_barang
  2. $kategori
  3. $harga
  4. $stok
  5. $diskon
  6. $id_barang

6. Eksekusi dan Pengecekan Hasil

```
if ($stmt->execute()) {
    echo "success";
} else {
    echo "error";
}
```

- $stmt->execute() mengeksekusi pernyataan yang telah dipersiapkan.
- if ($stmt->execute()) mengecek apakah eksekusi berhasil:
- echo "success"; → Akan menampilkan success jika data berhasil diupdate di database.
- echo "error"; → Akan menampilkan error jika terjadi kesalahan saat mengupdate data.

6. Create_transaksi.php (Folder php-transaksi)

```php
<?php
include "../../config/db.php";

if (isset($_POST['id_barang'], $_POST['jumlah'])) {
    $id_barang = $_POST['id_barang'];
    $jumlah = $_POST['jumlah'];

    // Ambil harga barang dari tabel barang
    $query = $conn->prepare("SELECT harga, diskon FROM barang
WHERE id_barang = ?");
    $query->bind_param("i", $id_barang);
    $query->execute();
    $result = $query->get_result();

    if ($result->num_rows > 0) {
        $row = $result->fetch_assoc();
        $harga = $row['harga'];
        $diskon = $row['diskon'];

        // Hitung harga setelah diskon
        $harga_diskon = $harga - ($harga * $diskon / 100);
        $total_harga = $harga_diskon * $jumlah;

        // Insert ke tabel transaksi
        $stmt = $conn->prepare("INSERT INTO transaksi (id_barang,
jumlah, total_harga) VALUES (?, ?, ?)");
        $stmt->bind_param("iid", $id_barang, $jumlah,
$total_harga);

        if ($stmt->execute()) {
            echo "success";
        } else {
            echo "error";
        }
    } else {
        echo "barang_not_found";
    }
}
?>
```

7. Delete_transaksi.php (Folder php-transaksi)

```php
<?php
include "../../config/db.php";

if (isset($_POST['id_transaksi'])) {
    $id_transaksi = $_POST['id_transaksi'];

    $stmt = $conn->prepare("DELETE FROM transaksi WHERE
id_transaksi = ?");
    $stmt->bind_param("i", $id_transaksi);

    if ($stmt->execute()) {
        echo "success";
    } else {
        echo "error";
    }
}
?>
```

8. Read_transaksi.php (Folder php-transaksi)

```php
<?php
include "../../config/db.php";

$result = $conn->query("
    SELECT t.id_transaksi, b.nama_barang, t.jumlah,
t.total_harga, t.tanggal_transaksi
    FROM transaksi t
    JOIN barang b ON t.id_barang = b.id_barang
    ORDER BY t.id_transaksi DESC
");

$data = [];
while ($row = $result->fetch_assoc()) {
    $data[] = $row;
}

echo json_encode($data);
?>
```

GET, POST, DELET, UPDATE

AJAX Javascript

1. index.js (folder js)

```
// Active menu Responsive
document.querySelector('.nav-toggle').addEventListener('click',
function () {
    document.querySelector('.menu').classList.toggle('active');
});

// Fetch dashboard data from API
async function fetchDashboardData() {
    try {
        const response = await
fetch('http://localhost/minimarket/dashboard_api.php');
        const data = await response.json();

        document.getElementById('totalProducts').textContent =
data.total_products;
        document.getElementById('totalTransactions').textContent =
data.total_transactions;
        document.getElementById('totalCustomers').textContent =
data.total_customers;
        document.getElementById('totalRevenue').textContent = '$' +
data.total_revenue;
    } catch (error) {
        console.error('Error fetching dashboard data:', error);
    }
}

// Load dashboard data when page loads
document.addEventListener('DOMContentLoaded', function () {
fetchDashboardData();
});
```

2. barang.js (folder js-barang)

```
const myTable = document.getElementById("display-barang");
const myEdit = document.getElementById("form-barang");
const myJudul = document.getElementById("judul");

// Active menu Responsive
document.querySelector('.nav-toggle').addEventListener('click',
function () {
    document.querySelector('.menu').classList.toggle('active');
});

$(document).ready(function() {
    loadData();

    $("#addBarang").click(function() {
        myTable.style.display = "none";
        myEdit.style.display = "block";
        myJudul.innerHTML ="Tambah Barang" ;
    });

    $("#barangForm").submit(function(e) {
        e.preventDefault();
        let id_barang = $("#id_barang").val();
        let nama_barang = $("#nama_barang").val();
        let kategori = $("#kategori").val();
        let harga = $("#harga").val();
        let stok = $("#stok").val();
        let diskon = $("#diskon").val();

        let url = id_barang ? "php/barang/update.php" :
"php/barang/create.php";

        $.post(url, { id_barang, nama_barang, kategori, harga,
stok, diskon }, function(response) {
            if (response === "success") {
                alert("Data tersimpan!")
                $("#barangForm")[0].reset();
                $("#id_barang").val("");
                loadData();
            } else {
                alert("Gagal menyimpan data");
```

```
            }
        });
    });

    function loadData() {
        myTable.style.display = "block";
        myEdit.style.display = "none";
        $.getJSON("php/barang/read.php", function(data) {
            console.log(data);
            let rows = "";
            data.forEach(item => {
                rows += `<tr>
                    <td>${item.nama_barang}</td>
                    <td>${item.kategori}</td>
                    <td>${item.harga}</td>
                    <td>${item.stok}</td>
                    <td>${item.diskon}</td>
                    <td>
                        <button
                            title="Edit"
                            class="btn btn-edit"
                            onclick='editData(${JSON.stringify(item
)})'>
                            <i class="fas fa-edit"></i>
                        </button>
                        <button
                            title="Delete"
                            class="btn btn-delete"
                            onclick='deleteData(${item.id_barang})'
>
                            <i class="fas fa-trash"> </i>
                        </button>
                    </td>
                </tr>`;
            });
            $("#barangTable").html(rows);
        });
    }

    window.editData = function(item) {
        myTable.style.display = "none";
```

```
        myEdit.style.display = "block";
        myJudul.innerHTML ="Update Barang" ;

        $("#id_barang").val(item.id_barang);
        $("#nama_barang").val(item.nama_barang);
        $("#kategori").val(item.kategori);
        $("#harga").val(item.harga);
        $("#stok").val(item.stok);
        $("#diskon").val(item.diskon);
    };

    window.deleteData = function(id_barang) {
        if (confirm("Yakin ingin menghapus?")) {
            $.post("php/barang/delete.php", { id_barang },
function(response) {
                if (response === "success") {
                    loadData();
                } else {
                    alert("Gagal menghapus data");
                }
            });
        }
    };
});
```

3. transaksi.js (folder js-transaksi)

```
const myTable = document.getElementById("display-transaksi");
const myForm = document.getElementById("form-transaksi");

// Active menu Responsive
document.querySelector('.nav-toggle').addEventListener('click',
function () {
    document.querySelector('.menu').classList.toggle('active');
});

$(document).ready(function() {
    loadBarang();
    loadTransaksi();

    $("#addTransaksi").click(function() {
```

```
        myTable.style.display = "none";
        myForm.style.display = "block";
    });

    $("#transaksiForm").submit(function(e) {
        e.preventDefault();
        let id_barang = $("#id_barang").val();
        let jumlah = $("#jumlah").val();

        $.post("php/transaksi/create_transaksi.php", { id_barang,
jumlah }, function(response) {
            if (response === "success") {
                $("#transaksiForm")[0].reset();
                loadTransaksi();
            } else if (response === "barang_not_found") {
                alert("Barang tidak ditemukan!");
            } else {
                alert("Gagal menambahkan transaksi!");
            }
        });
    });

    function loadBarang() {
        $.getJSON("php/barang/read.php", function(data) {
            let options = '<option value="">Pilih Barang</option>';
            data.forEach(item => {
                options += `<option
value="${item.id_barang}">${item.nama_barang}</option>`;
            });
            $("#id_barang").html(options);
        });
    }

    function loadTransaksi() {
        myTable.style.display = "block";
        myForm.style.display = "none";
        $.getJSON("php/transaksi/read_transaksi.php",
function(data) {
            let rows = "";
            data.forEach(item => {
                rows += `<tr>
```

```
                    <td>${item.nama_barang}</td>
                    <td>${item.jumlah}</td>
                    <td>${item.total_harga}</td>
                    <td>${item.tanggal_transaksi}</td>
                    <td>
                        <button
                            class="btn btn-delete"
                            onclick="deleteTransaksi(${item.id_tran
saksi})">
                            <i class="fas fa-trash"> </i>
                        </button>
                    </td>
                </tr>`;
            });
            $("#transaksiTable").html(rows);
        });
    }

    window.deleteTransaksi = function(id_transaksi) {
        if (confirm("Yakin ingin menghapus transaksi?")) {
            $.post("php/transaksi/delete_transaksi.php", {
id_transaksi }, function(response) {
                if (response === "success") {
                    loadTransaksi();
                } else {
                    alert("Gagal menghapus transaksi!");
                }
            });
        }
    };
});
```