

LAPORAN PROJECT UTS
PEMBUATAN SISTEM KASIR COFFEE SHOP



**UNIVERSITAS
SAINS DAN TEKNOLOGI
INDONESIA**

DISUSUN OLEH :

Defry Maulana

2110031802050

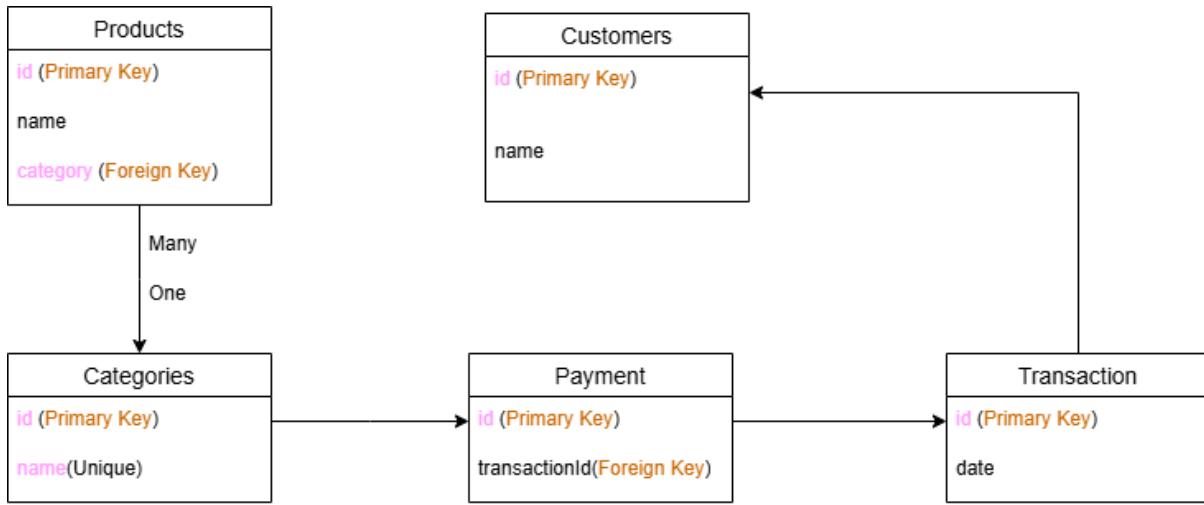
DOSEN PENGAMPU :

Dosen Pengampu :

Torkis Nasution, M.Kom

PROGRAM STUDI TEKNIK INFORMATIKA
UNIVERSITAS SAINS DAN TEKNOLOGI INDONESIA
2024

1. Entity Relational Diagram



2. Penjelasan Entity Relational Diagram

Entities :

Products:

- `id` (PK): Primary Key, unik untuk setiap produk.
- `name`: Nama produk.
- `category` (FK): Foreign Key yang merujuk ke `Categories`, menunjukkan kategori produk tersebut.

Categories:

- `id` (PK): Primary Key, unik untuk setiap kategori.
- `name` (Unique): Nama kategori yang harus unik.

Transactions:

- `id` (PK): Primary Key, unik untuk setiap transaksi.
- `date`: Tanggal transaksi dilakukan.

Customers:

- `id` (PK): Primary Key, unik untuk setiap pelanggan.
- `name`: Nama pelanggan.

Payments:

- `id` (PK): Primary Key, unik untuk setiap pembayaran.
- `transactionId` (FK): Foreign Key yang merujuk ke `Transactions`, menunjukkan transaksi yang terkait dengan pembayaran ini.

Hubungan

Products ke Categories:

- Setiap produk dapat dimiliki oleh satu kategori (One), tetapi satu kategori dapat memiliki banyak produk (Many). Ini adalah hubungan Many-to-One.

Payments ke Transactions:

- Setiap pembayaran terkait dengan satu transaksi (One), tetapi satu transaksi dapat memiliki banyak pembayaran (Many). Ini juga merupakan hubungan Many-to-One.

3. Physical Data Model

- Tabel Products

Kolom	Tipe Data	Keterangan
Id	INT (Primary Key)	Auto Increment
Name	Varchar(255)	Nama Produk
Price	Decimal (10,2)	Harga Produk
categoryId	INT (foreign Key)	Mengacu ke categories.id

- Tabel Categories

Kolom	Tipe Data	Keterangan
Id	INT (Primary Key)	Auto Increment
Name	Varchar(255)	Nama Kategori (Unique)

- Tabel Customers

Kolom	Tipe Data	Keterangan
Id	INT (Primary Key)	Auto Increment
Name	Varchar(255)	Nama Pelanggan
Email	Varchar(255)	Email pelanggan

- Tabel Transactions

Kolom	Tipe Data	Keterangan
Id	INT (Primary Key)	Auto Increment
customerId	INT (Foreign Key)	Mengacu ke customers.id
productId	INT (Foreign Key)	Mengacu ke products.id
Quantity	INT	Jumlah Produk

- **Tabel Payments**

Kolom	Tipe data	Keterangan
Id	INT (Primary Key)	Auto Increment
transactionId	INT (Foreign Key)	Mengacu ke transactions.id
Amount	Decimal (10,2)	Total Pembayaran
Method	Varchar (255)	Metode Pembayaran

4. Penjelasan Physical Data Model

Tabel dan Perannya

1. Products:

- Menyimpan informasi produk yang dijual di Coffee Shop.
- Memiliki hubungan dengan tabel **Categories** untuk menentukan kategori dari setiap produk.

2. Categories:

- Menyimpan daftar kategori produk, seperti *beverages*, *snacks*, atau *merchandise*.
- Setiap kategori unik (tidak boleh ada nama kategori yang sama).

3. Customers:

- Menyimpan data pelanggan yang melakukan transaksi.
- Setiap pelanggan memiliki informasi unik, seperti nama dan email.

4. Transactions:

- Menyimpan informasi setiap transaksi pembelian, termasuk produk yang dibeli, pelanggan yang membeli, dan jumlah produk.
- Berfungsi sebagai penghubung antara pelanggan (**Customers**) dan produk (**Products**).

5. Payments:

- Menyimpan data pembayaran untuk transaksi tertentu.
- Setiap pembayaran terkait dengan satu transaksi saja (relasi *one-to-one*).

Hubungan Antar Tabel

1. Relasi Products ↔ Categories (Many-to-One):

- Setiap produk memiliki satu kategori, tetapi satu kategori dapat memiliki banyak produk.
- Foreign Key: Products.categoryId mengacu ke Categories.id.

2. Relasi Transactions ↔ Customers (Many-to-One):

- Setiap transaksi dilakukan oleh satu pelanggan, tetapi satu pelanggan dapat memiliki banyak transaksi.
- Foreign Key: Transactions.customerId mengacu ke Customers.id.

3. Relasi Transactions ↔ Products (Many-to-One):

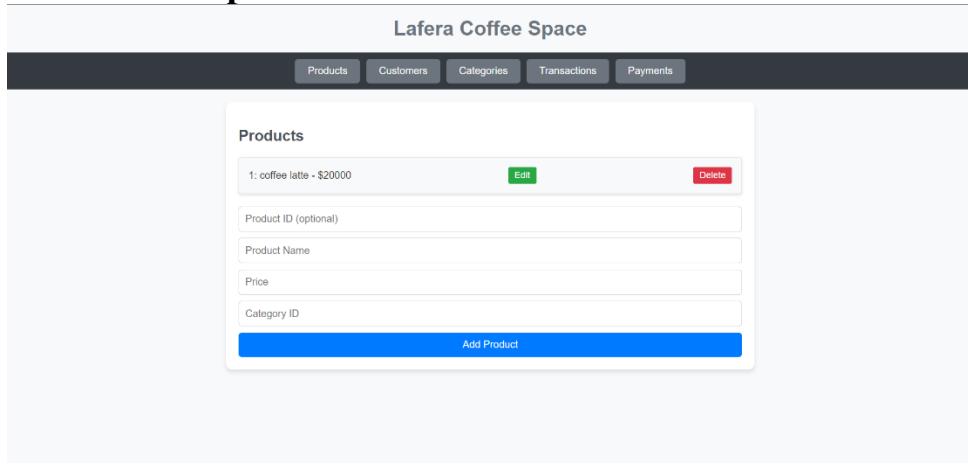
- Setiap transaksi mencatat produk tertentu yang dibeli, tetapi satu produk dapat terlibat dalam banyak transaksi.
- Foreign Key: Transactions.productId mengacu ke Products.id.

4. Relasi Payments ↔ Transactions (One-to-One):

- Setiap pembayaran hanya terkait dengan satu transaksi, dan setiap transaksi hanya memiliki satu pembayaran.
- Foreign Key: Payments.transactionId mengacu ke Transactions.id.

5. Program Menampilkan Data

- Gambar Tampilan Hasil



- **Penjelasan**

Pada program ini, sistem dapat menginput data dan menampilkan daftar produk secara langsung.

- **Source Code**

```
function getProducts() {
    const transaction = db.transaction("products", "readonly");
    const store = transaction.objectStore("products");
    const request = store.getAll();

    request.onsuccess = function(event) {
        const productList = document.getElementById('product-list');
        productList.innerHTML = '';
        event.target.result.forEach(product => {
            const li = document.createElement('li');
            li.textContent = `${product.id}: ${product.name} - ${product.price}`;
            // Create edit button
            const editButton = document.createElement('button');
            editButton.textContent = 'Edit';
            editButton.onclick = () => editProduct(product.id);
            li.appendChild(editButton);

            // Create delete button
            const deleteButton = document.createElement('button');
            deleteButton.textContent = 'Delete';
            deleteButton.onclick = () => deleteProduct(product.id);
            li.appendChild(deleteButton);
            productList.appendChild(li);
        });
    };
}
```

6. Program Entri Data

- **Gambar Entri Data**

The screenshot shows a web application interface for 'Lafera Coffee Space'. At the top, there is a navigation bar with tabs: 'Products' (which is selected), 'Customers', 'Categories', 'Transactions', and 'Payments'. Below the navigation bar is a sub-navigation bar for 'Products' with tabs: 'Products', 'Customers', 'Categories', 'Transactions', and 'Payments'. The main content area is titled 'Products' and contains four input fields: 'Product ID (optional)', 'Product Name', 'Price', and 'Category ID'. Below these fields is a blue button labeled 'Add Product'.

- **Penjelasan**

Pada setiap tabel sistem ini,dapat melakukan entry data

- **Source Code**

```
243 document.getElementById('product-form').onsubmit = function(event) {
244   event.preventDefault();
245   const id = document.getElementById('product-id').value || Date.now();
246   const name = document.getElementById('product-name').value;
247   const price = parseFloat(document.getElementById('product-price').value);
248   const categoryId = document.getElementById('product-categoryId').value;
249
250   const transaction = db.transaction("products", "readwrite");
251   const store = transaction.objectStore("products");
252   store.put({ id, name, price, categoryId });
253
254   transaction.oncomplete = function() {
255     getProducts();
256     document.getElementById('product-form').reset();
257   };
258 }
259
260 function editProduct(productId) {
261   const transaction = db.transaction("products", "readonly");
262   const store = transaction.objectStore("products");
263   const request = store.get(productId);
264
265   request.onsuccess = function(event) {
266     const product = event.target.result;
267     if (product) {
268       document.getElementById('product-id').value = product.id;
269       document.getElementById('product-name').value = product.name;
270       document.getElementById('product-price').value = product.price;
271       document.getElementById('product-categoryId').value = product.categoryId;
272     } else {
273       console.error("Product not found");
274     }
275   };
276 }
```

```

277 |     request.onerror = handleError;
278 |
279 }
280 document.getElementById('product-form').onsubmit = function(event) {
281     event.preventDefault();
282     const id = document.getElementById('product-id').value;
283     const name = document.getElementById('product-name').value;
284     const price = parseFloat(document.getElementById('product-price').value);
285     const categoryId = document.getElementById('product-categoryId').value;
286
287     const transaction = db.transaction(["products", "readwrite"]);
288     const store = transaction.objectStore("products");
289     const request = store.put({ id, name, price, categoryId });
290
291     request.onsuccess = function() {
292         getProducts();
293         document.getElementById('product-form').reset();
294     };
295
296     request.onerror = handleError;
297 };
298
299 function deleteProduct(productId) {
300     const transaction = db.transaction(["products", "readwrite"]);
301     const store = transaction.objectStore("products");
302     const request = store.delete(productId);
303
304     request.onsuccess = function() {
305         getProducts(); // Refresh the product list
306     };
307
308     request.onerror = handleError;
309 }

```

7. Program Update Data

- Gambar Update Data

The screenshot shows the 'Products' section of the application. The table displays a single product entry: '1: coffee latte - \$20000'. The 'Edit' button is highlighted in green, while the 'Delete' button is red. Below the table is a form with four input fields corresponding to the product details: ID (1), Name (coffee latte), Price (20000), and Category ID (1). A large blue 'Add Product' button is at the bottom of the form.

- Penjelasan

Pada program update ini,sistem dapat melakukan edit pada setiap field

- Source Code

```

260     function editProduct(productId) {
261         const transaction = db.transaction(["products", "readonly"]);
262         const store = transaction.objectStore("products");
263         const request = store.get(productId);
264
265         request.onsuccess = function(event) {
266             const product = event.target.result;
267             if (product) {
268                 document.getElementById('product-id').value = product.id;
269                 document.getElementById('product-name').value = product.name;
270                 document.getElementById('product-price').value = product.price;
271                 document.getElementById('product-categoryId').value = product.categoryId;
272             } else {
273                 console.error("Product not found");
274             }
275         };
276
277         request.onerror = handleError;
278     }

```

8. Program Hapus Data

- Gambar Tampilan Hapus Data

The image contains two vertically stacked screenshots of a web application named "Lafera Coffee Space". Both screenshots show the "Products" section of the application.

Screenshot 1 (Top): This screenshot shows a product entry for "coffee latte" with a price of "\$20000". The entry is listed in a table with four columns: ID, Name, Price, and Quantity. A green "Edit" button and a red "Delete" button are visible at the top right of the table. Below the table is a blue "Add Product" button.

1	coffee latte	\$20000	1
---	--------------	---------	---

Screenshot 2 (Bottom): This screenshot shows the same product entry after it has been deleted. The table now only contains three rows, corresponding to the other products in the database. The "Add Product" button is also present at the bottom.

1	coffee latte	\$20000	1
---	--------------	---------	---

- **Penjelasan**

Gambar 1. Menunjukkan bahwa masih ada daftar produk

Gambar 2. Menunjukkan bahwa daftar produk telah dihapus

- **Source Code**

```
299 function deleteProduct(productId) {
300     const transaction = db.transaction("products", "readwrite");
301     const store = transaction.objectStore("products");
302     const request = store.delete(productId);
303
304     request.onsuccess = function() {
305         getProducts(); // Refresh the product list
306     };
307
308     request.onerror = handleError;
309 }
310
```

9. Penutup

- Kesimpulan**

Proyek pembuatan sistem kasir untuk Coffee Shop ini telah dirancang dan diimplementasikan dengan pendekatan yang sistematis. Sistem ini mampu menangani proses entri data, pembaruan, penghapusan, serta pengelolaan transaksi secara efisien. Dengan desain database yang baik, sistem ini dapat mendukung operasional Coffee Shop dan memberikan kemudahan dalam pengelolaan data produk, pelanggan, transaksi, serta pembayaran. Sistem yang dikembangkan diharapkan mampu meningkatkan produktivitas, mengurangi kesalahan manusia, dan memberikan pengalaman yang lebih baik kepada pelanggan.

- Saran**

- 1. Pengembangan Fitur Tambahan**

Untuk meningkatkan kapabilitas sistem, dapat ditambahkan fitur seperti laporan penjualan otomatis, integrasi dengan pembayaran digital, atau manajemen stok secara real-time.

- 2. Peningkatan Keamanan Sistem**

Mengimplementasikan enkripsi pada data pelanggan dan transaksi untuk memastikan keamanan informasi.

- 3. Uji Coba pada Skala Nyata**

Sebaiknya sistem diuji pada lingkungan operasional Coffee Shop untuk mengidentifikasi potensi perbaikan sebelum diterapkan sepenuhnya.

- 4. Pelatihan Pengguna**

Menyediakan pelatihan kepada staf Coffee Shop agar dapat menggunakan sistem ini secara efektif.

Dengan adanya saran ini, diharapkan sistem dapat terus dikembangkan dan memberikan manfaat yang optimal bagi penggunanya.

