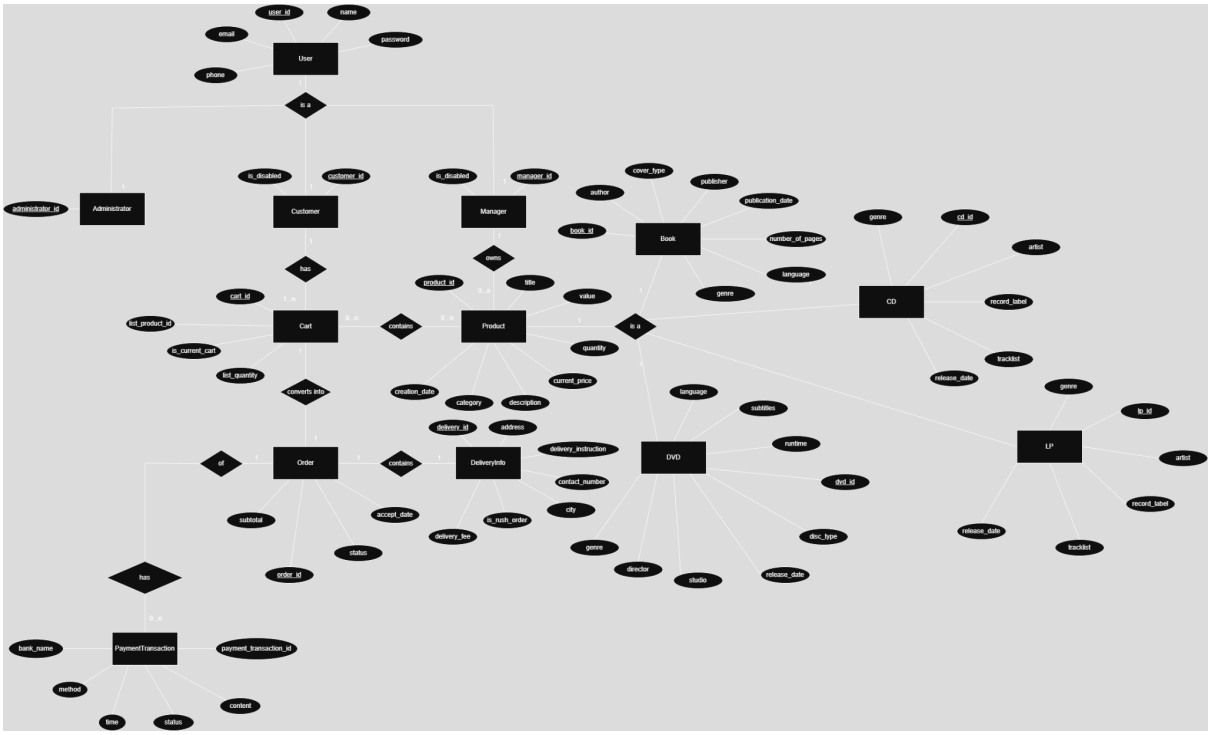
I. Conceptual Data Model



Conceptual Data Model Explanation (ERD):

- User Entity: in AIMS software requirements, each user has 1 role: Administrator, Customer and Manager. So the relation “is a” specify them clearly. “1-1” is used here since one User Record is exactly one Admin/ Customer/ Manager record.

  After a Customer is created, a Cart ID will be automatically linked with that Customer ID.

- Product Entity: the design pattern of this entity is the same as the User entity. Additionally, the 0…n - 1 relationship between it and the Manager entity means 1 manager can have 0 to many products in store. Cart and Product should be many to many since products appear in many carts and many cats can have 1 product.

- Relation between Customer, Cart, Order, Product, Delivery Info, PaymentTransaction is designed base on:

  - “Place Order” UC: in the requirement document, each customer has 1 Cart but it doesn't limit CartID related to 1 Customer. At the same time, only 1 cart is used for “Add to Cart” operation, the attribute “is_current”

will handle it. Each time a user makes an Order, current Cart data will be used for further UC after this UC. After finishing payment for that Order , a new Cart is created and linked to the Customer. The reason why the 1-1 relationship will be explained below.

- Rush Order is specified by attribute “is_rush_order”, if this attribute is true, a new shipping fee algorithm will be applied for this order.

  We also save an attribute named “subtotal” to show the total price of this order before adding VAT. The reason it is added is because we want to reduce the time if User wants to search for an order in the price range. If subtotal does not exist, each time a filter queries happens, AIMS software needs to recalculate the order value, this could take more time so creating attributes although requires more memory but can save time.

- “Pay Order” UC: After clicking on Pay Order on UI, this UC is called. Order entity linked to Cart entity with cart_id to receive information, status attribute is used to track it. Order ‘s status depends on Payment Transaction status and the choice of Product Manager. In this UC, only Payment Transaction status matter

  When customers pay for an order, they must enter Delivery Info for that Order. After filling in, AIMS software creates a request to pay to VN_Pay and get back the result of the payment session. Each payment session creates a PaymentTranscation record with status successful/ failed.

- “Approve/Reject” UC: as we mentioned before, status of Order after successful pay is “successful payment”. The Product Manager then can choose to approve or reject that Order.

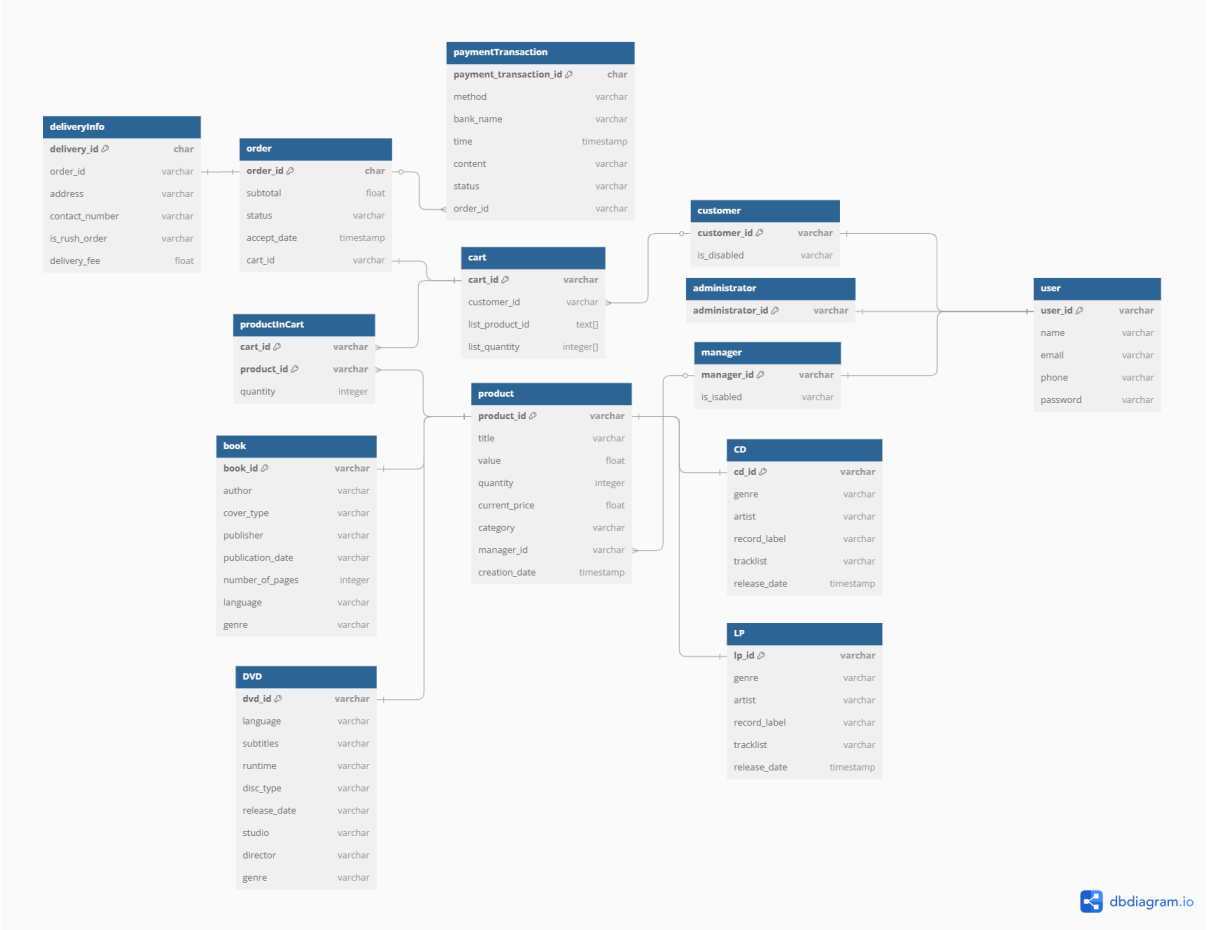
II. Database Design

1. Logical Data Model

Based on the conceptual data model (i.e., ER diagram) presented in the previous section, we can derive the logical data model tailored to the selected DBMS, in this case, SQLite. Below is the logical data model corresponding to our ER diagram.

2. Physical Data Model

In this part, we need to provide a detailed design of each element in the database (DB) diagram. For instance, in a relational DBMS, we present a detailed design for each table and its constraints, as illustrated in the table below (PK: Primary Key, FK: Foreign Key).



- **User**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		user_id	VARCHAR	Yes	User ID
2			name	VARCHAR	Yes	Username
3			email	VARCHAR	Yes	Email Address
4			phone	VARCHAR	Yes	Phone Number
5			password	VARCHAR	Yes	User password

- **Administrator**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		administrator_id	VARCHAR	Yes	Administrator ID

- **Manager**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		manager_id	VARCHAR	Yes	Manager ID
2			is_disabled	VARCHAR	Yes	Status of account (enabled or disabled)

- **Customer**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		customer_id	VARCHAR	Yes	Customer ID
2			disabled	VARCHAR	Yes	Status of account (enabled or disabled)

- **Product**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		product_id	VARCHAR	Yes	Product ID
2			title	VARCHAR	Yes	Product title

3			value	FLOAT	Yes	Original value
4			quantity	INTEGER	Yes	Stock quantity
5			current_price	FLOAT	Yes	Current price
6			category	VARCHAR	Yes	Product category
7		x	manager_id	VARCHAR	Yes	Manager responsible for the product
8			creation_date	TIMESTAMP	Yes	Date product was created

- **Book**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		book_id	VARCHAR	Yes	Book ID
2			author	VARCHAR	Yes	Book author
3			cover_type	VARCHAR	Yes	Type of book cover
4			publisher	VARCHAR	Yes	Publisher name
5			publication_date	VARCHAR	Yes	Date of publication
6			number_of_pages	INTEGER	Yes	Number of pages
7			language	VARCHAR	Yes	Language of the book

8			genre	VARCHAR	Yes	Genre of the book

- **CD**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		cd_id	VARCHAR	Yes	CD ID
2			genre	VARCHAR	Yes	Genre of the CD
3			artist	VARCHAR	Yes	Artist name
4			record_label	VARCHAR	Yes	Record label
5			tracklist	VARCHAR	Yes	Tracklist of the CD
6			release_date	TIMESTAMP	Yes	Release date

- **DVD**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		dvd_id	VARCHAR	Yes	DVD ID
2			language	VARCHAR	Yes	Language
3			subtitles	VARCHAR	Yes	Available subtitles
4			runtime	VARCHAR	Yes	Runtime duration
5			disc_type	VARCHAR	Yes	Type of disc

6			release_date	VARCHAR	Yes	Release date
7			studio	VARCHAR	Yes	Studio name
8			director	VARCHAR	Yes	Director name
9			genre	VARCHAR	Yes	Genre of the DVD

- **LP**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		lp_id	VARCHAR	Yes	LP ID
2			genre	VARCHAR	Yes	Genre of the LP
3			artist	VARCHAR	Yes	Artist name
4			record_label	VARCHAR	Yes	Record label
5			tracklist	VARCHAR	Yes	Tracklist of the LP
6			release_date	TIMESTAMP	Yes	Release date

- **Cart**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		cart_id	VARCHAR	Yes	Cart ID
2		x	customer_id	VARCHAR	Yes	Customer who owns cart

3			list_productID	TEXT	Yes	List of product IDs
4			list_quantity	INTEGER	Yes	Quantity of each product

- **Order**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		order_id	VARCHAR	Yes	Order ID
2			subtotal	FLOAT	Yes	Total before taxes/fees
3			status	VARCHAR	Yes	Order status
4			accept_date	TIMESTAMP	Yes	When order was accepted
5		x	cart_id	VARCHAR	Yes	Associated cart

- **DeliveryInfo**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		delivery_id	VARCHAR	Yes	Delivery ID
2		x	order_id	VARCHAR	Yes	Related order
3			address	VARCHAR	Yes	Delivery address
4			contact_number	VARCHAR	Yes	Contact number

5			is_rush_order	VARCHAR	Yes	Rush delivery? (Yes/No)
6			delivery_fee	FLOAT	Yes	Delivery fee

- **ProductInCart**

#	PK	FK	Column Name	Data type	Mandatory	Description
1		x	cart_id	VARCHAR	Yes	Referenced Cart ID
2		x	product_id	VARCHAR	Yes	Referenced Product ID
3			quantity	INTEGER	Yes	Quantity of the product

- **PaymentTransaction**

#	PK	FK	Column Name	Data type	Mandatory	Description
1	x		payment_transaction_id	VARCHAR	Yes	Transaction ID
2			method	VARCHAR	Yes	Payment method
3			bank_name	VARCHAR	Yes	Bank involved
4			time	TIMESTAMP	Yes	Time of transaction
5			content	VARCHAR	Yes	Transaction content/details
6			status	VARCHAR	Yes	Status of transaction
7		x	order_id	VARCHAR	Yes	Related order

Physical Data Model Explanation:

This class diagram is designed based on ERD. The “productInCart” table is created as a junction table to represent this many-to-many relationship. This ensures:

- You can't have the same product twice in the same cart (unless you update the quantity).
- Efficiency for enforcing uniqueness in the context of a cart-product combo.

We also create an index for improving performance. By default, all id is index, further reports would create more depending on the project.

Finally, we need a database script. With the help of specialized database development tools and plugins, we can generate the database script directly from the logical data model.

```
CREATE TABLE User (
    user_id VARCHAR PRIMARY KEY,
    name VARCHAR NOT NULL,
    email VARCHAR NOT NULL,
    phone VARCHAR NOT NULL,
    password VARCHAR NOT NULL
);

CREATE TABLE Customer (
    customer_id VARCHAR PRIMARY KEY,
    disabled VARCHAR NOT NULL
);

CREATE TABLE Manager (
    manager_id VARCHAR PRIMARY KEY,
    disabled VARCHAR NOT NULL
```

```
);


CREATE TABLE Administrator (
    administrator_id VARCHAR PRIMARY KEY
);


CREATE TABLE Product (
    product_id VARCHAR PRIMARY KEY,
    title VARCHAR NOT NULL,
    value FLOAT NOT NULL,
    quantity INTEGER NOT NULL,
    current_price FLOAT NOT NULL,
    category VARCHAR NOT NULL,
    manager_id VARCHAR NOT NULL,
    creation_date TIMESTAMP NOT NULL,
    FOREIGN KEY (manager_id) REFERENCES Manager(manager_id)
);


CREATE TABLE Cart (
    cart_id VARCHAR PRIMARY KEY,
    customer_id VARCHAR NOT NULL,
    list_productID TEXT[] NOT NULL,
    list_quantity INTEGER[] NOT NULL,
    FOREIGN KEY (customer_id) REFERENCES Customer(customer_id)
);
```

```sql
CREATE TABLE Order (
    order_id VARCHAR PRIMARY KEY,
    subtotal FLOAT NOT NULL,
    status VARCHAR NOT NULL,
    accept_date TIMESTAMP NOT NULL,
    cart_id VARCHAR NOT NULL,
    FOREIGN KEY (cart_id) REFERENCES Cart(cart_id)
);

CREATE TABLE PaymentTransaction (
    payment_transaction_id VARCHAR PRIMARY KEY,
    method VARCHAR NOT NULL,
    bank_name VARCHAR NOT NULL,
    time TIMESTAMP NOT NULL,
    content VARCHAR NOT NULL,
    status VARCHAR NOT NULL,
    order_id VARCHAR NOT NULL,
    FOREIGN KEY (order_id) REFERENCES "Order"(order_id)
);

CREATE TABLE DeliveryInfo (
    delivery_id VARCHAR PRIMARY KEY,
    order_id VARCHAR NOT NULL,
    address VARCHAR NOT NULL,
```

```
    contact_number VARCHAR NOT NULL,
    is_rush_order VARCHAR NOT NULL,
    delivery_fee FLOAT NOT NULL,
    FOREIGN KEY (order_id) REFERENCES "Order"(order_id)
);

CREATE TABLE Book (
    book_id VARCHAR PRIMARY KEY,
    author VARCHAR NOT NULL,
    cover_type VARCHAR NOT NULL,
    publisher VARCHAR NOT NULL,
    publication_date VARCHAR NOT NULL,
    number_of_pages INTEGER NOT NULL,
    language VARCHAR NOT NULL,
    genre VARCHAR NOT NULL
);

CREATE TABLE CD (
    cd_id VARCHAR PRIMARY KEY,
    genre VARCHAR NOT NULL,
    artist VARCHAR NOT NULL,
    record_label VARCHAR NOT NULL,
    tracklist VARCHAR NOT NULL,
    release_date TIMESTAMP NOT NULL
);
```

```
CREATE TABLE LP (
    lp_id VARCHAR PRIMARY KEY,
    genre VARCHAR NOT NULL,
    artist VARCHAR NOT NULL,
    record_label VARCHAR NOT NULL,
    tracklist VARCHAR NOT NULL,
    release_date TIMESTAMP NOT NULL
);

CREATE TABLE DVD (
    dvd_id VARCHAR PRIMARY KEY,
    language VARCHAR NOT NULL,
    subtitles VARCHAR NOT NULL,
    runtime VARCHAR NOT NULL,
    disc_type VARCHAR NOT NULL,
    release_date VARCHAR NOT NULL,
    studio VARCHAR NOT NULL,
    director VARCHAR NOT NULL,
    genre VARCHAR NOT NULL
);

CREATE TABLE Product_in_Cart (
    cart_id VARCHAR NOT NULL,
    product_id VARCHAR NOT NULL,
```

```
    quantity INTEGER NOT NULL,

    PRIMARY KEY (cart_id, product_id),

    FOREIGN KEY (cart_id) REFERENCES Cart(cart_id),

    FOREIGN KEY (product_id) REFERENCES Product(product_id)

);
```

3. Data Modelling for UC “Place Rush Order”

In this project, if a place rush order is created, only the shipping fee is changed so we don't need to add more entities or tables for it.