Lab Report

Only for course Teacher						
		Needs Improvement	Developing	Sufficient	Above Average	Total Mark
Allocate mark & Percentage		**25%**	**50%**	**75%**	**100%**	**5**
Clarity	**1**					
Content Quality	**2**					
Spelling & Grammar	**1**					
Organization and Formatting	**1**					
					Total obtained mark	
Comments						

Semester: Spring - 2024

Student Information:

- Name: Md. Rasel Hossain ID Number: 0242220005341013
- Name: Md. Ahanaf Mohosen ID Number: 0242220005341072
- Name: Md Sajidur Rahman Shimul ID Number: 0242220005341051
- Name: Kayes Ahmed Himu ID Number: 0242220005341035

Batch: 39th **Section:** "A"
Course Code: SE 224 **Course Name:** Database System Lab
Course Teacher Name: Mr. Fazla Rabby Raihan
Designation: Lecturer, Department of Software Engineering, DIU

Submission Date: 11 June 2024

INTRODUCTION

In this slide, Database and its types, advantages, components etc. are discussed.

DBMS:

A Database Management System (DBMS) is a software tool designed to create, manage, retrieve, update, and organize data within a database. It plays a crucial role in the expanding use of computers, significantly impacting various fields such as business, electronic commerce, engineering, medicine, genetics, law, education, and library science. Essentially, databases are integral to almost all areas where computers are utilized.

The primary goal of a DBMS is to provide a convenient and efficient method for storing and retrieving database information. Data refers to recorded facts with embedded meaning. Common software used for storing data in databases includes DBASE IV or V, Microsoft Access, and Excel. A single unit of data is called a datum, and when meaningful data are combined, they form information—interpreted data enriched with semantics. Microsoft Access is a widely recognized example of database management software.

Database Management Systems have become indispensable for information management. Consequently, courses on the principles and practice of database systems are now a fundamental part of computer science curricula. This book delves into the basics of modern database management systems, focusing particularly on relational database systems.

1.1 Causes to use DBMS

- To develop software applications in less time.
- Data independence and efficient use of data.
- For uniform data administration.
- For data integrity and security.
- For concurrent access to data, and data recovery from crashes.
- To use user-friendly declarative query language.

1.2 Advantage of DBMS

A DBMS manage data and has many advantages. These are:

- **Data Independence:** Application programs should be as free or independent as possible from details of data representation and storage. DBMS can supply an abstract view of the data for insulating application code from such facts.
- **Efficient Data Access:** DBMS utilizes a mixture of sophisticated concepts and techniques for storing and retrieving data competently, and this feature becomes important in cases where the data is stored on external storage devices.
- **Data Integrity and Security:** If data is accessed through the DBMS, the DBMS can enforce integrity constraints on the data.

- **Data Administration:** When several users share the data, integrating the administration of data can offer major improvements. Experienced professionals understand the nature of the data being managed and can be responsible for organizing the data representation to reduce redundancy and make the data to retrieve efficiently.

1.3 Components of DBMS

- **Users:** Users may be of any kind such as DB administrator, System developer or database users.
- **Database application:** Database application may be Departmental, Personal, organization's and / or Internal.
- **DBMS:** Software that allows users to create and manipulate database access.
- **Database**: Collection of logical data as a single unit.

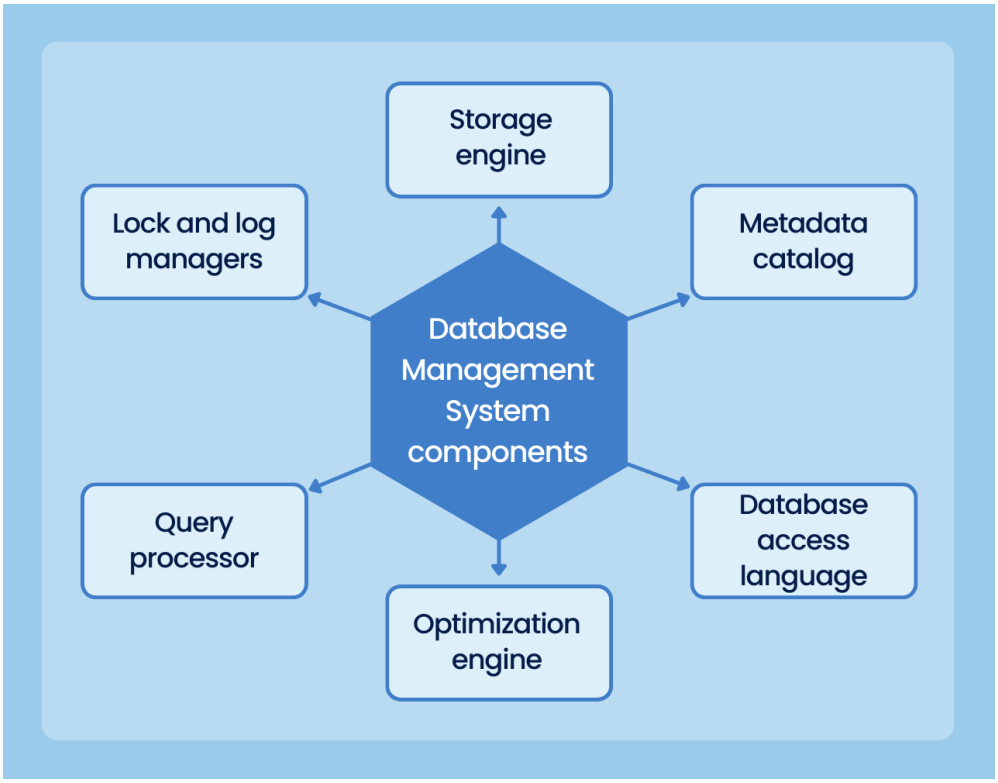

Fig 1.1: components of database management system

Types of DBMS

There are four main types of Database Management Systems (DBMS), classified based on their management of database structures. The type of DBMS depends on how the database is structured by that particular system.

2.1 Hierarchical DBMS

A DBMS is considered hierarchical if the relationships among data are organized in a hierarchy, resembling a tree structure. Each data item is a subordinate or subunit of another. This model, developed by IBM in 1968 and introduced in information management systems, structures records as nodes in a tree.

2.2 Network DBMS

A DBMS is categorized as a Network DBMS if the relationships among data are of the many-to-many type. This paradigm, one of the major Internet computing paradigms, allows multiple users to contribute and receive information, with data elements often interlinked across different websites.

2.3 Relational DBMS

A DBMS is referred to as a Relational DBMS (RDBMS) if it organizes database relationships in the form of tables. The three primary components of an RDBMS are relations, domains, and attributes.

2.4 Object-Oriented DBMS

Object-oriented DBMS can handle various new data types, including graphics, photographs, audio, and video, representing a significant advancement over other types. Unlike hierarchical and network databases, which are designed to handle structured data fitting into fields, rows, and columns, object-oriented databases manage more complex and unstructured data.

REQUIREMENTS SPECIFICATION

In this chapter, hardware requirements and software requirements have been discussed.

System Specification can be divided into two-
Hardware specifications
Software specifications

Table: Specifications

Hardware specifications	Software specifications
32/64-bit operating system	Operating System: Windows.
2 GB RAM or above	Languages: MySQL
40 GB hard disk or above	Database: MySQL
VGA COLOR Monitor	Server: XAMPP, MySQL
Keyboard	Browser: Chrome etc.
Mouse	Text editor: Sublime Text.

MODULE DESCRIPTION

In this chapter we describe the list of modules in "Online Medicine Buying App"

3.1. User Registration and Login

- **Description:** This module handles the registration of new users and the login process for existing users.
- **Tables Involved:** `Users`, `LoginCredentials`
- **Key Operations:**
  - ✓ Registration: Users provide their name, email, phone, and address to create an account.
  - ✓ Email and Phone Verification: Ensures authenticity.
  - ✓ Login: Users authenticate using a username and password.

3.2. Browsing and Searching for Medicines

- **Description:** Allows users to explore and search for medicines within different categories.
- **Tables Involved:** `Categories`, `Medicines`
- **Key Operations:**
  - ✓ Category Browsing: Users can view medicines by categories like Prescription Medicines, Over-the-Counter Medicines, and Health Supplements.
  - ✓ Search Functionality: Users can search for specific medications by name.
  - ✓ Detailed Views: Users can view detailed descriptions and read reviews of medicines.

3.3. Cart Management

- **Description:** Enables users to add, review, and modify items in their shopping cart.
- **Tables Involved:** `Cart`, `Users`, `Medicines`
- **Key Operations:**
  - ✓ Add to Cart: Users add medicines to their cart.
  - ✓ Review Cart: Users review the items and quantities in their cart.
  - ✓ Modify Cart: Users can update or remove items from the cart.

3.4. Order Placement and Checkout

- **Description:** Manages the process of finalizing orders and processing payments.
- **Tables Involved:** `Orders`, `Users`
- **Key Operations:**
  - ✓ Checkout: Users review their cart and proceed to checkout.
  - ✓ Payment: Users select a payment method and provide payment details.
  - ✓ Order Confirmation: Confirms the order and provides an order summary.

3.5. Prescription Upload and Verification

- **Description:** Handles the uploading and verification of prescriptions for prescription medications.
- **Tables Involved:** `PrescriptionUploads`, `Users`
- **Key Operations:**
  - ✓ Upload Prescription: Users take and upload a photo of their prescription.
  - ✓ Verification: The system verifies the authenticity and validity of the prescription.

3.6. Order Tracking and Delivery

- **Description:** Allows users to track the status of their orders in real-time.
- **Tables Involved:** `OrderTracking`, `Orders`, `OrderStatus`
- **Key Operations:**
  - ✓ Order Status Updates: Provides real-time updates on order status.
  - ✓ Tracking Information: Displays tracking information including dispatch and delivery estimates.

3.7. Medicine Refills and Reminder Alerts

- **Description:** Provides features for managing medicine refills and setting reminders.
- **Tables Involved:** `Refills`, `ReminderAlerts`, `Users`, `Medicines`
- **Key Operations:**
  - ✓ Refills: Users can reorder regular medications.
  - ✓ Reminders: Users can set alerts for when to take medications or refill prescriptions.

3.8. Customer Support and Feedback

- **Description:** Facilitates customer support interactions and collects feedback.
- **Tables Involved:** `CustomerSupportChat`, `Feedback`, `Users`, `Orders`
- **Key Operations:**
  - ✓ Support Chat: Users can communicate with support representatives.
  - ✓ Feedback: Users can rate and provide comments on their orders and shopping experience.

3.9. Promotions and Discounts

- Description: Manages promotional offers and discounts available to users.
- Tables Involved: `Promotions`, `MedicinePromotions`
- Key Operations:
  - ✓ Promotion Notifications: Users receive notifications about ongoing promotions.
  - ✓ Apply Discounts: Discounts are applied to eligible medicines during checkout.

10. User Account Management

- **Description:** Allows users to manage their account settings and view their order history.
- **Tables Involved:** `UserAccountSettings`, `Users`, `OrderHistory`

- **Key Operations:**
  - ✓ Account Settings: Users can update personal information and account preferences.
  - ✓ Order History: Users can view past orders and their details.

Summary

The above module descriptions provide a comprehensive overview of the functionalities provided by the online medicine buying app. Each module interacts with specific tables in the database to perform its operations, ensuring a seamless and user-friendly experience for managing medication purchases and health needs.

ER-Diagram:

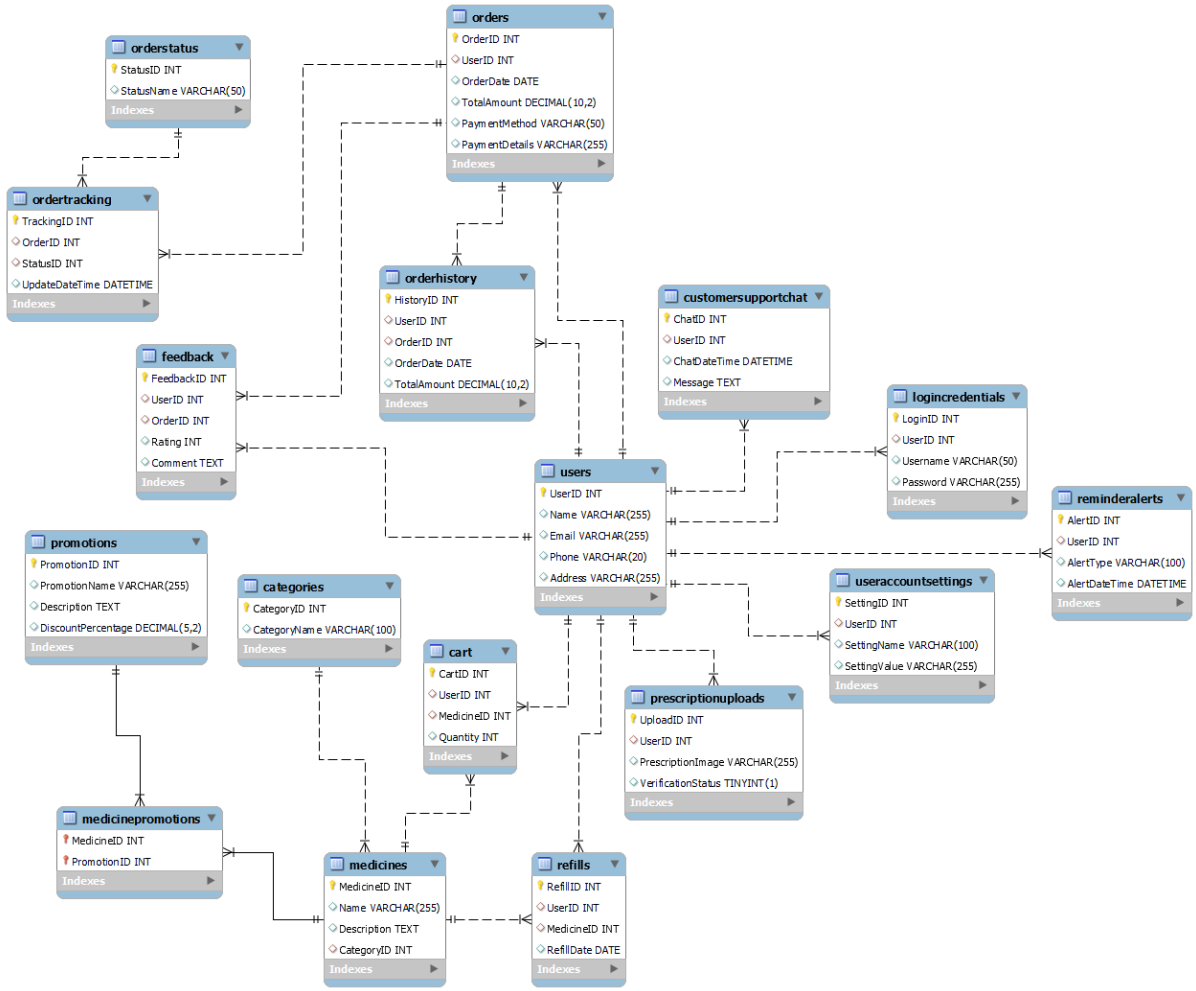

Fig. E-R diagram of Online Medicine Buying App

DATABASE TABLES:

The database consists of one or more tables. Each table is made of rows and column. Each row in a relational uniquely by primary key. This can be by one or more sets of columns value in most of scenarios it is a single column.

TABLE 01: Users Table

Field Name	Data Type	Constraints
UserID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
Name	VARCHAR(255)	NOT NULL
Email	VARCHAR(255)	NOT NULL
Phone	VARCHAR(20)	NOT NULL
Address	VARCHAR(255)	NOT NULL

TABLE 02: LoginCredentials Table

Field Name	Data Type	Constraints
LoginID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
Username	VARCHAR(50)	NOT NULL
Password	VARCHAR(255)	NOT NULL

TABLE 03: Categories Table

Field Name	Data Type	Constraints
CategoryID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
CategoryName	VARCHAR(100)	NOT NULL

TABLE 04: Medicines Table

Field Name	Data Type	Constraints
MedicineID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
Name	VARCHAR(255)	NOT NULL
Description	TEXT	NOT NULL
CategoryID	INT	FOREIGN KEY, NOT NULL

TABLE 05: Cart Table

Field Name	Data Type	Constraints
CartID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
MedicineID	INT	FOREIGN KEY, NOT NULL
Quantity	INT	NOT NULL

TABLE 06: Orders Table

Field Name	Data Type	Constraints
OrderID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
OrderDate	DATE	NOT NULL
TotalAmount	DECIMAL(10, 2)	NOT NULL
PaymentMethod	VARCHAR(50)	NOT NULL
PaymentDetails	VARCHAR(255)	NOT NULL

TABLE 07: PrescriptionUploads Table

Field Name	Data Type	Constraints
UploadID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
PrescriptionImage	VARCHAR(255)	NOT NULL
VerificationStatus	BOOLEAN	NOT NULL

TABLE 08: OrderStatus Table

Field Name	Data Type	Constraints
StatusID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
StatusName	VARCHAR(50)	NOT NULL

TABLE 09: OrderTracking Table

Field Name	Data Type	Constraints
TrackingID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
OrderID	INT	FOREIGN KEY, NOT NULL
StatusID	INT	FOREIGN KEY, NOT NULL
UpdateDateTime	DATETIME	NOT NULL

TABLE 10: Refills Table

Field Name	Data Type	Constraints
RefillID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
MedicineID	INT	FOREIGN KEY, NOT NULL
RefillDate	DATE	NOT NULL

TABLE 11: ReminderAlerts Table

Field Name	Data Type	Constraints
AlertID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
AlertType	VARCHAR(100)	NOT NULL
AlertDateTime	DATETIME	NOT NULL

TABLE 12: CustomerSupportChat Table

Field Name	Data Type	Constraints
ChatID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
ChatDateTime	DATETIME	NOT NULL
Message	TEXT	NOT NULL

TABLE 13: Feedback Table

Field Name	Data Type	Constraints
FeedbackID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
OrderID	INT	FOREIGN KEY, NOT NULL
Rating	INT	NOT NULL
Comment	TEXT	NOT NULL

TABLE 14: Promotions Table

Field Name	Data Type	Constraints
PromotionID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
PromotionName	VARCHAR(255)	NOT NULL
Description	TEXT	NOT NULL
DiscountPercentage	DECIMAL(5, 2)	NOT NULL

TABLE 15: MedicinePromotions Table

Field Name	Data Type	Constraints
MedicineID	INT	FOREIGN KEY, NOT NULL
PromotionID	INT	FOREIGN KEY, NOT NULL
		PRIMARY KEY(MedicineID, PromotionID)

TABLE 16: UserAccountSettings Table

Field Name	Data Type	Constraints
SettingID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
SettingName	VARCHAR(100)	NOT NULL
SettingValue	VARCHAR(255)	NOT NULL

TABLE 17: OrderHistory Table

Field Name	Data Type	Constraints
HistoryID	INT	PRIMARY KEY, AUTO_INCREMENT, NOT NULL
UserID	INT	FOREIGN KEY, NOT NULL
OrderID	INT	FOREIGN KEY, NOT NULL
OrderDate	DATE	NOT NULL
TotalAmount	DECIMAL(10, 2)	NOT NULL

BASIC DATA RETRIEVAL

Query 1: Select all data from a table of your choice.

```
3   -- Query 1: Select all data from a table of your choice.
4   USE medicine_buying_app;
5   SELECT * FROM Users;
```

UserID	Name	Email	Phone	Address
1	John Smith	john.smith@example.com	1234567890	123 Main St
2	Jane Doe	jane.doe@example.com	0987654321	456 Elm St
3	Michael Johnson	michael.johnson@example.com	5551234567	789 Oak St
4	Emily Davis	emily.davis@example.com	9876543210	321 Pine St
5	William Brown	william.brown@example.com	5559876543	654 Maple St
6	Olivia Wilson	olivia.wilson@example.com	1239876543	987 Cedar St
7	James Taylor	james.taylor@example.com	5555551234	654 Birch St
8	Emma Martinez	emma.martinez@example.com	1235554321	321 Oak St
9	Alexander Anderson	alexander.anderson@example.com	5555559876	456 Elm St
10	Sophia Thomas	sophia.thomas@example.com	5551239876	789 Maple St

Query 2: Select specific columns from a table. (Students should replace 'column_names' and 'table_name' with actual column and table names.

```
6
7   /* Query 2: Select specific columns from a table.
8      (Students should replace 'column_names' and 'table_name'
9       with actual column and table names.) */
10
11      SELECT Name, Email FROM Users;
12
13
14
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Email
John Smith	john.smith@example.com
Jane Doe	jane.doe@example.com
Michael Johnson	michael.johnson@example.com
Emily Davis	emily.davis@example.com
William Brown	william.brown@example.com
Olivia Wilson	olivia.wilson@example.com
James Taylor	james.taylor@example.com
Emma Martinez	emma.martinez@example.com
Alexander Anderson	alexander.anderson@example.com
Sophia Thomas	sophia.thomas@example.com
Daniel Jackson	daniel.jackson@example.com

Conditional Retrieval

Query 3: Retrieve data from a table where a particular column meets a certain condition. (Students should define the column and the condition.)

```
14  /* Query 3: Retrieve data from a table where a particular column meets
15  a certain condition. (Students should define the column and the condition.)*/
16
17  SELECT * FROM Users WHERE Name = 'Jane Doe';
18
19
20
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content

UserID	Name	Email	Phone	Address
2	Jane Doe	jane.doe@example.com	0987654321	456 Elm St
NULL	NULL	NULL	NULL	NULL

Sorting and Ordering

Query 4: Retrieve data from a table and sort it in ascending/descending order based on a specific column.

```
20  /* Query 4: Retrieve data from a table and sort it in
21  ascending/descending order based on a specific column.*/
22
23  SELECT * FROM Users ORDER BY Name ASC;
24
25
26
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap

UserID	Name	Email	Phone	Address
35	Abigail Harris	abigail.harris@example.com	1237899876	789 Oak St
50	Abigail Lee	abigail.lee@example.com	5557895555	789 Oak St
42	Aiden Smith	aiden.smith@example.com	5555551234	654 Pine St
92	Aiden Walker	aiden.walker@example.com	5557895555	654 Pine St
9	Alexander Anderson	alexander.anderson@example.com	5555559876	456 Elm St
69	Alexander Lee	alexander.lee@example.com	1235559876	456 Birch St
46	Alexander Martin	alexander.martin@example.com	5557899876	987 Elm St
19	Alexander Rodriguez	alexander.rodriguez@example.com	1237895555	456 Birch St
61	Alexander Smith	alexander.smith@example.com	1237895555	987 Elm St
16	Amelia Garcia	amelia.garcia@example.com	5555553210	987 Maple St
65	Amelia Lee	amelia.lee@example.com	1237899876	789 Oak St
83	Amelia Rodriguez	amelia.rodriguez@example.com	1237899876	321 Cedar St
67	Avery Harris	avery.harris@example.com	1237895555	654 Pine St

Aggregation Functions

Query 5: Use an aggregation function (SUM, AVG, MIN, MAX, COUNT) on a column.

```
26  /* Query 5: Use an aggregation function
27  (SUM, AVG, MIN, MAX, COUNT) on a column.*/
28
29  SELECT COUNT(*) FROM Users;
30
31
32
```

Result Grid | Filter Rows: | Export:

COUNT(*)
100

Query 6: Group results by one or more columns and apply an aggregation function.

```
32  /* Query 6: Group results by one or more columns
33  and apply an aggregation function.*/
34
35  SELECT Address, COUNT(*) FROM Users GROUP BY Address;
36
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Address	COUNT(*)
123 Main St	1
456 Elm St	2
789 Oak St	18
321 Pine St	1
654 Maple St	2
987 Cedar St	1
654 Birch St	1
321 Oak St	1
789 Maple St	1

Joins

Query 7: Perform an INNER JOIN between two tables on a related column.

```
39  /* Query 7: Perform an INNER JOIN between two tables
40  on a related column.*/
41
42  SELECT Users.Name, LoginCredentials.Username
43  FROM Users
44  INNER JOIN LoginCredentials ON Users.UserID = LoginCredentials.UserID;
45
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Username
John Smith	john_doe
Jane Doe	emma_smith
Michael Johnson	amanda_jones
Emily Davis	michael_brown
William Brown	jessica_taylor
Olivia Wilson	david_miller
James Taylor	olivia_wilson
Emma Martinez	william_taylor
Alexander Anderson	sophia_johnson
Sophia Thomas	james_anderson
Daniel Jackson	emily_thomas
Mia White	matthew_jackson
Matthew Harris	emma_white

Query 8: Perform a LEFT JOIN or RIGHT JOIN between two tables on a related column.

```
49  /* Query 8: Perform a LEFT JOIN or RIGHT JOIN between
50  two tables on a related column.*/
51
52  SELECT Users.Name, LoginCredentials.Username
53  FROM Users
54  LEFT JOIN LoginCredentials ON Users.UserID = LoginCredentials.UserID;
55
56
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	Username
John Smith	john_doe
Jane Doe	emma_smith
Michael Johnson	amanda_jones
Emily Davis	michael_brown
William Brown	jessica_taylor
Olivia Wilson	david_miller
James Taylor	olivia_wilson
Emma Martinez	william_taylor
Alexander Anderson	sophia_johnson
Sophia Thomas	james_anderson
Daniel Jackson	emily_thomas

Subqueries and Nested Queries

Query 9: Select data from a table where a column's value is in the result of another SELECT statement.

```
59  /* Query 9: Select data from a table where a column's value is
60  in the result of another SELECT statement.*/
61
62  SELECT * FROM Users WHERE UserID IN (SELECT UserID FROM LoginCredentials);
63
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell C

UserID	Name	Email	Phone	Address
1	John Smith	john.smith@example.com	1234567890	123 Main St
2	Jane Doe	jane.doe@example.com	0987654321	456 Elm St
3	Michael Johnson	michael.johnson@example.com	5551234567	789 Oak St
4	Emily Davis	emily.davis@example.com	9876543210	321 Pine St
5	William Brown	william.brown@example.com	5559876543	654 Maple St
6	Olivia Wilson	olivia.wilson@example.com	1239876543	987 Cedar St
7	James Taylor	james.taylor@example.com	5555551234	654 Birch St
8	Emma Martinez	emma.martinez@example.com	1235554321	321 Oak St
9	Alexander Anderson	alexander.anderson@example.com	5555559876	456 Elm St
10	Sophia Thomas	sophia.thomas@example.com	5551239876	789 Maple St
11	Daniel Jackson	daniel.jackson@example.com	1234569876	987 Pine St
12	Mia White	mia.white@example.com	5553219876	654 Cedar St
13	Matthew Harris	matthew.harris@example.com	1235557890	321 Birch St
14	Charlotte Martin	charlotte.martin@example.com	5559871234	456 Oak St
15	Ethan Thompson	ethan.thompson@example.com	1234565555	789 Elm St
16	Amelia Garcia	amelia.garcia@example.com	5555553210	987 Maple St
17	Benjamin Martinez	benjamin.martinez@example.com	1235551234	654 Pine St

Query 10: Select the top 3 unique values from a column of a table. (Use DISTINCT and LIMIT or TOP depending on the SQL dialect.)

```
66  /* Query 10: Select the top 3 unique values from a column of a table.
67  (Use DISTINCT and LIMIT or TOP depending on the SQL dialect.)*/
68
69  SELECT DISTINCT Address FROM Users LIMIT 3;
70
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

Address
123 Main St
456 Elm St
789 Oak St

Inserting Data

Query 11: Insert new data into a table. (Students should specify the table and the values for each column.)

```
73  /* Query 11: Insert new data into a table.
74  (Students should specify the table and the values for each column.)*/
75
76  INSERT INTO Users (Name, Email, Phone, Address)
77  VALUES ('kayes himu', 'kayes.himu@example.com', '123-456-9870', '153 Elm Street');
78
79
80
```

Output

Action Output

#	Time	Action
61	13:17:47	SELECT Users.Name, LoginCredentials.Username FROM Users LEFT JOIN LoginCredentials ON Users.UserID = L
62	13:17:47	SELECT * FROM Users WHERE UserID IN (SELECT UserID FROM LoginCredentials) LIMIT 0, 1000
63	13:17:47	SELECT DISTINCT Address FROM Users LIMIT 3
64	13:17:47	INSERT INTO Users (Name, Email, Phone, Address) VALUES ('kayes himu', 'kayes.himu@example.com', '123-456-
65	13:18:07	INSERT INTO Users (Name, Email, Phone, Address) VALUES ('kayes himu', 'kayes.himu@example.com', '123-456-

Updating Data

Query 12: Update existing data in a table. (Specify which rows to update based on a condition and which columns to change.)

```
80  /* Query 12: Update existing data in a table. (Specify which rows
81  to update based on a condition and which columns to change.)*/
82
83  UPDATE Users SET Email = 'new.email@example.com' WHERE UserID = 1;
84
85
86
87
```

Output

Action Output

#	Time	Action
75	13:19:10	SELECT * FROM Users WHERE UserID IN (SELECT UserID FROM LoginCredentials) LIMIT 0,
76	13:19:10	SELECT DISTINCT Address FROM Users LIMIT 3
77	13:19:10	INSERT INTO Users (Name, Email, Phone, Address) VALUES ('kayes himu', 'kayes.himu@exar
78	13:19:10	UPDATE Users SET Email = 'new.email@example.com' WHERE UserID = 1
79	13:19:21	UPDATE Users SET Email = 'new.email@example.com' WHERE UserID = 1

Deleting Data

Query 13: Delete data from a table based on a specific condition.

```
87  /* Query 13: Delete data from a table based on a specific condition.*/
88
89  DELETE FROM cart WHERE UserID = 7;
90
91
92
93
```

Output

Action Output

#	Time	Action
1	13:31:00	USE medicine_buying_app
2	13:31:07	DELETE FROM cart WHERE UserID = 7

Advanced Features

Query 14: Use a CASE statement within a SELECT query to create conditional logic in the output.

```
92  /* Query 14: Use a CASE statement within a SELECT query
93     to create conditional logic in the output.*/
94
95  SELECT Name,
96  CASE
97      WHEN Address LIKE '%Street%' THEN 'Street'
98      ELSE 'Other'
99  END AS AddressType
100 FROM Users;
101
102
103
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Name	AddressType
Ella Lee	Other
Jackson Young	Other
Charlotte Rodriguez	Other
Liam Martin	Other
Emma Harris	Other
Aiden Walker	Other
Avery Lee	Other
Harper Young	Other
Mia Rodriguez	Other
William Martin	Other
Evelyn Harris	Other
Olivia Walker	Other
Noah Lee	Other
Sophia Young	Other
kayes himu	Street
kayes himu	Street

Query 15: Implement a transaction that includes at least one INSERT, UPDATE, and DELETE operation, and uses COMMIT and ROLLBACK based on a condition.

```
/* Query 15: Implement a transaction that includes
at least one INSERT, UPDATE, and DELETE operation,
and uses COMMIT and ROLLBACK based on a condition. */


START TRANSACTION;
INSERT INTO Users (Name, Email, Phone, Address)
VALUES ('John Smith', 'john.smith@example.com', '123-456-7890', '123 Elm St');
UPDATE Users
SET Phone = '555-555-5555'
WHERE Name = 'John Smith';
DELETE FROM Users
WHERE Name = 'John Smith' AND Email = 'john.smith@example.com';
IF ROW_COUNT() = 1 THEN
    COMMIT;
ELSE
    ROLLBACK;
END IF;
```

Output

Action Output

#	Time	Action
1	14:29:47	USE medicine_buying_app
2	14:29:57	START TRANSACTION

CONCLUSION AND FUTURE SCOPE OF THE PROJECT

Conclusion

The development of the online medicine buying app has provided a comprehensive solution for Sarah and other users who need a reliable and convenient way to purchase medications. The app ensures a seamless experience from registration to order delivery, incorporating features such as:

1. User Registration and Login: Secure and straightforward user registration with email and phone verification to ensure authenticity.
2. Browsing and Searching for Medicines: A user-friendly interface that allows easy navigation and searching for medicines, complete with detailed descriptions and reviews.
3. Cart and Checkout: An efficient cart and checkout system that supports various payment methods and secure payment details.
4. Prescription Upload and Verification: Automated prescription verification to ensure authenticity and compliance with regulations.
5. Order Tracking and Delivery: Real-time order tracking and delivery notifications to keep users informed of their order status.
6. Medicine Refills and Reminder Alerts: Convenient refill options and reminder alerts to help users manage their medication schedules.
7. Customer Support and Feedback: Accessible customer support and feedback mechanisms to address user issues and enhance the service.
8. Promotions and Discounts: Regular updates on promotions and discounts to help users save on their purchases.
9. User Account Management: Comprehensive account management features, including order history and secure storage of payment information.

Future Scope

The future scope of this project includes several enhancements and additional features to further improve the user experience and expand the app's capabilities:

1. Integration with Healthcare Providers: Partnering with healthcare providers and pharmacies to offer direct prescription services and professional health advice within the app.
2. Telemedicine Services: Incorporating telemedicine features to allow users to consult with doctors online, receive prescriptions, and order medications in one seamless process.

3. Enhanced Security Measures: Implementing advanced security measures, such as biometric authentication and end-to-end encryption, to ensure user data protection and privacy.
4. Personalized Medicine Recommendations: Using AI and machine learning algorithms to provide personalized medicine recommendations based on user health data and purchase history.
5. Expanded Delivery Options: Offering more flexible delivery options, including same-day delivery, pick-up points, and collaboration with local pharmacies for quicker access.
6. Health and Wellness Content: Providing users with access to a library of health and wellness content, including articles, videos, and tips from healthcare professionals.
7. International Expansion: Expanding the app's availability to international markets, ensuring compliance with local regulations and offering multi-language support.
8. Improved User Analytics: Leveraging user analytics to gain insights into user behaviours and preferences, which can help in improving the app's features and user interface.
9. Loyalty Programs: Introducing loyalty programs and rewards for frequent users to enhance user retention and satisfaction.
10. Integration with Wearable Devices: Syncing with wearable health devices to track health metrics and provide insights into medication adherence and overall health.

By continuously evolving and incorporating user feedback, the online medicine buying app can become an essential tool in managing health and wellness, offering users a reliable, secure, and convenient platform for all their medication needs.

REFERENCES

- https://www.geeksforgeeks.org/
- https://www.tutorialspoint.com/index.htm
- https://www.blackbox.ai/
- https://www.w3schools.com
- https://chatgpt.com/

CODE LINK

- **https://github.com/ahanaf-mohosen57/Database-SQL-Project**