॥वसुधैव कुटुम्बकम्॥

SYMBIOSIS

INSTITUTE OF TECHNOLOGY, PUNE

A DBMS PROJECT ON
PHARMACY MANAGEMENT SYSTEM

SUBMITTED BY:

PRANAV MAHAJAN (22070122153)

PRIYANKA GUPTA (22070122157)

SAMIDHA MANJREKAR (22070122176)

SAMRUDDHI BORHADE (22070122177)

(GROUP NO. B8 , CS-B)

Under the Guidance of

Prof. Pooja Bagane

Department of Computer Science

SYMBIOSIS INSTITUTE OF TECHNOLOGY, PUNE

INDEX:

INTRODUCTION:-

The need for effective pharmacy management has reached a new high in the fast-paced world of medicine. Recognizing the challenges that chemists experience in maintaining correct stock records, monitoring stock levels, and handling pricing details, we present an innovative Pharmacy Management System (PMS). This comprehensive solution serves as a powerful ally for pharmacists, offering a user-friendly platform that streamlines inventory management while facilitating real-time tracking of medicine stock. Furthermore, the system includes dynamic pricing information to keep up with market swings, giving chemists accurate and up-to-date information for educated decision-making.

Our Pharmacy Management System goes beyond inventory control, extending its capabilities to the heart of customer interactions. Pharmacists can effortlessly generate detailed bills for customers, ensuring accuracy and transparency in every transaction by seamlessly integrating technology into daily pharmacy operations. This innovative approach is poised to redefine pharmacy management, offering a holistic solution that empowers pharmacists and improves the overall quality of service within the pharmaceutical industry.

The Pharmacy Management System aspires to simplify the acquisition and management of pharmaceuticals for the pharmacy. In essence, this system endeavors to elevate the overall experience within the pharmaceutical industry, ensuring enhanced accessibility and availability of vital healthcare products.

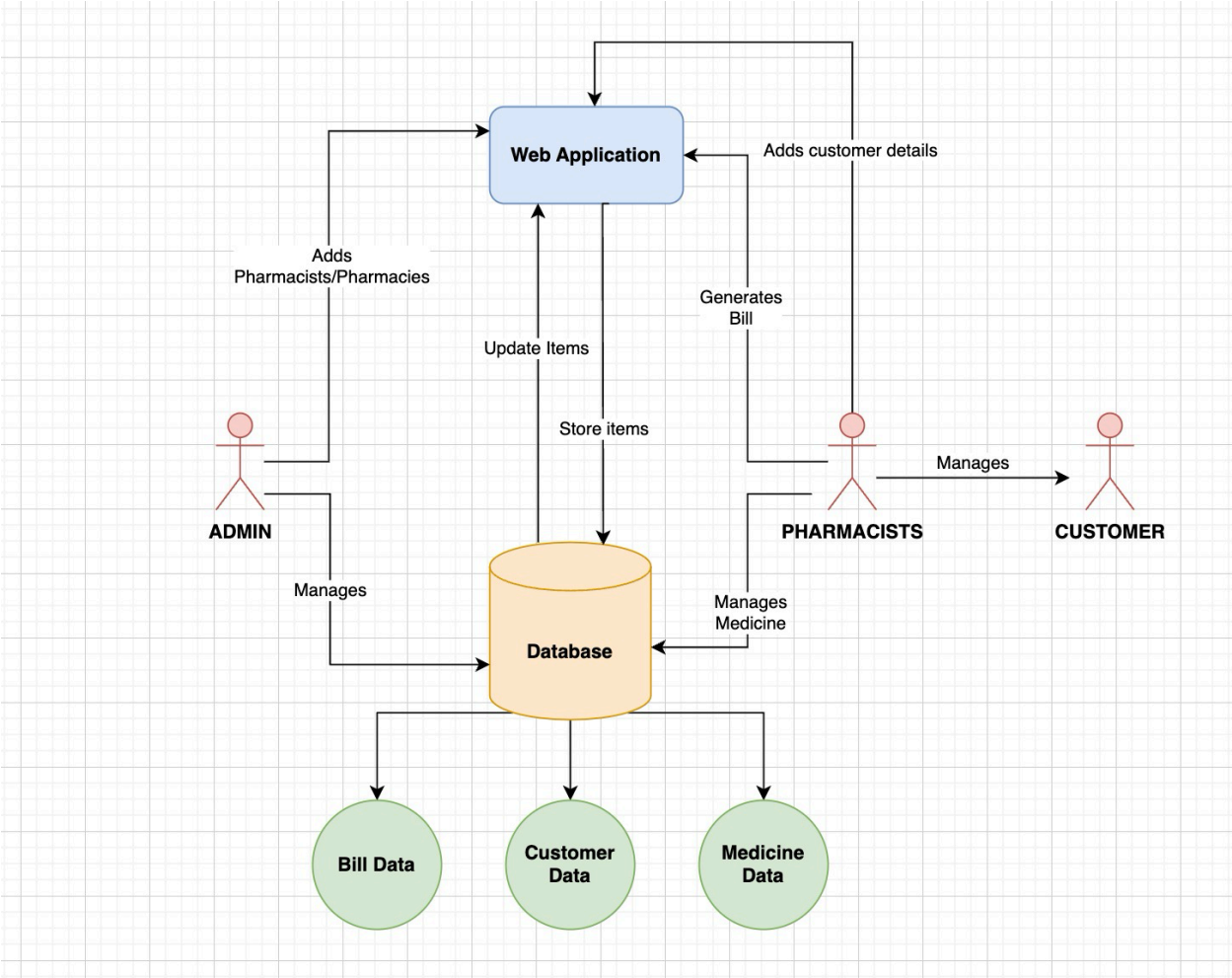
PROBLEM STATEMENT:

The existing landscape of pharmacy administration is characterized by operational inefficiencies caused by manual operations. Pharmacists frequently face the challenges of keeping accurate stock records and checking stock levels, which can lead to discrepancies and errors. Pricing details, critical for financial transparency, are frequently difficult to handle in real time, leaving chemists a time-consuming chore. These operational inefficiencies reduce overall pharmacy efficiency and increase the risk of mistakes in stock levels and pricing, compromising the quality of service delivered to clients.

Pharmacies work in a dynamic market where medicine pricing and availability can fluctuate frequently. Pharmacists who do not have a comprehensive Pharmacy Management System struggle to adjust effectively to market conditions. The inability to update pricing details in real time makes it challenging to remain competitive and respond quickly to market changes. This lack of adaptability can lead to financial losses, an imbalance in stock levels, and a decreased ability to meet the market's and customers' changing needs. Addressing these problems is critical for pharmacies to succeed in a continuously changing pharmaceutical industry.

By implementing this Pharmacy Management System, we aspire to simplify the complexities associated with acquiring and managing pharmaceuticals. The system aims to create a more accessible and transparent marketplace where customers can easily find the medications they need, and sellers can connect with a broader customer base. Our project aims to enhance the overall efficiency and experience within the pharmaceutical industry.

SYSTEM ARCHITECTURE :



Pharmacy Management System

MODULES:

I. Pharmacist Module

- This module allows pharmacists to add, update, delete, and view medication to the inventory.
- Pharmacists get notified by the app if the stock of a medicine is below the specified quantity. The notification will contain the details of the manufacturer.
- Pharmacists have the ability to generate bills for the customers.

II. Admin Module

- This module is essential for administrators to manage the pharmacy system, pharmacist accounts, and the pharmacy database.
- The administrator will be able to manage the manufacturer details.
- This module would allow admins to manage system security, such as user authentication and data encryption.

III. Website Module

- This module allows users to log in as admins or pharmacists.
- Pharmacists will be able to navigate through the medicine database and generate bills for the customer.

IV. Bill Module

- The bill will contain the medications, quantity, cost, mode of payment, and bill date.
- Integration with the inventory system to track medication availability and update stock levels based on supplies received.
- The pharmacist can see all the bills generated for that pharmacy.
- Communication features for pharmacies to place orders, receive invoices, and maintain a record of transactions with suppliers.

FUNCTIONAL REQUIREMENTS:

- User authentication is a must for every Member. The member needs to provide their personal details (Name, Phone no., Email, etc.), username and password.
- The admin will be able to add, remove, or edit the details of the pharmacies and pharmacists for the corresponding pharmacy.
- Admin can also solve the query
- Inventory would be managed by pharmacists. Every pharmacist would be given a unique ID and official email ID.
- When a bill is created, the stock is updated accordingly.
- Pharmacists have the ability to create bills for customers.
- Pharmacists can view, update, insert, and delete medicines in the inventory.
- If the stock is below the specified quantity, the pharmacist is notified.
- The bill will display the quantity and price of each medicine, the discount, and the total amount.
- The notification will contain Manufacturer details like name, contact number and email.
- Customers will have to provide a name, phone, and email.
- Medicine will be divided into various types.
- The inventory will store medicine name, price, stock, type, and ID.
- The pharmacist can give discounts.
- The pharmacist will be able to view the previous customers and their bills.

ENTITIES, RELATIONSHIPS AND ATTRIBUTES:

Entity	Relationship	Entity
Pharmacy	One pharmacy can hire many pharmacists	Pharmacist
Pharmacy	One pharmacy has many Items	Item
Manufacturer	A manufacturer can manufacture many medicines.	Item
Bill	One bill can contain multiple medicines.	Item
Customer	One customer can have multiple bills.	Bill
Customer	One pharmacy can have many customers.	Pharmacy
Medicine	Is an Item	Item
Medical Instrument	Is an Item	Item

Attributes:

1. **Pharmacy**
    - pharmacyID
    - Name
    - Address
    - Phone

2. **Pharmacist**
    - pharmacistID
    - Name
    - Email
    - Phone

3. **Customer**
    - customerID
    - Name
    - Phone
    - Email

4. **Bill**
    - billID
    - Discount
    - Total
    - Mode of payment
    - Date

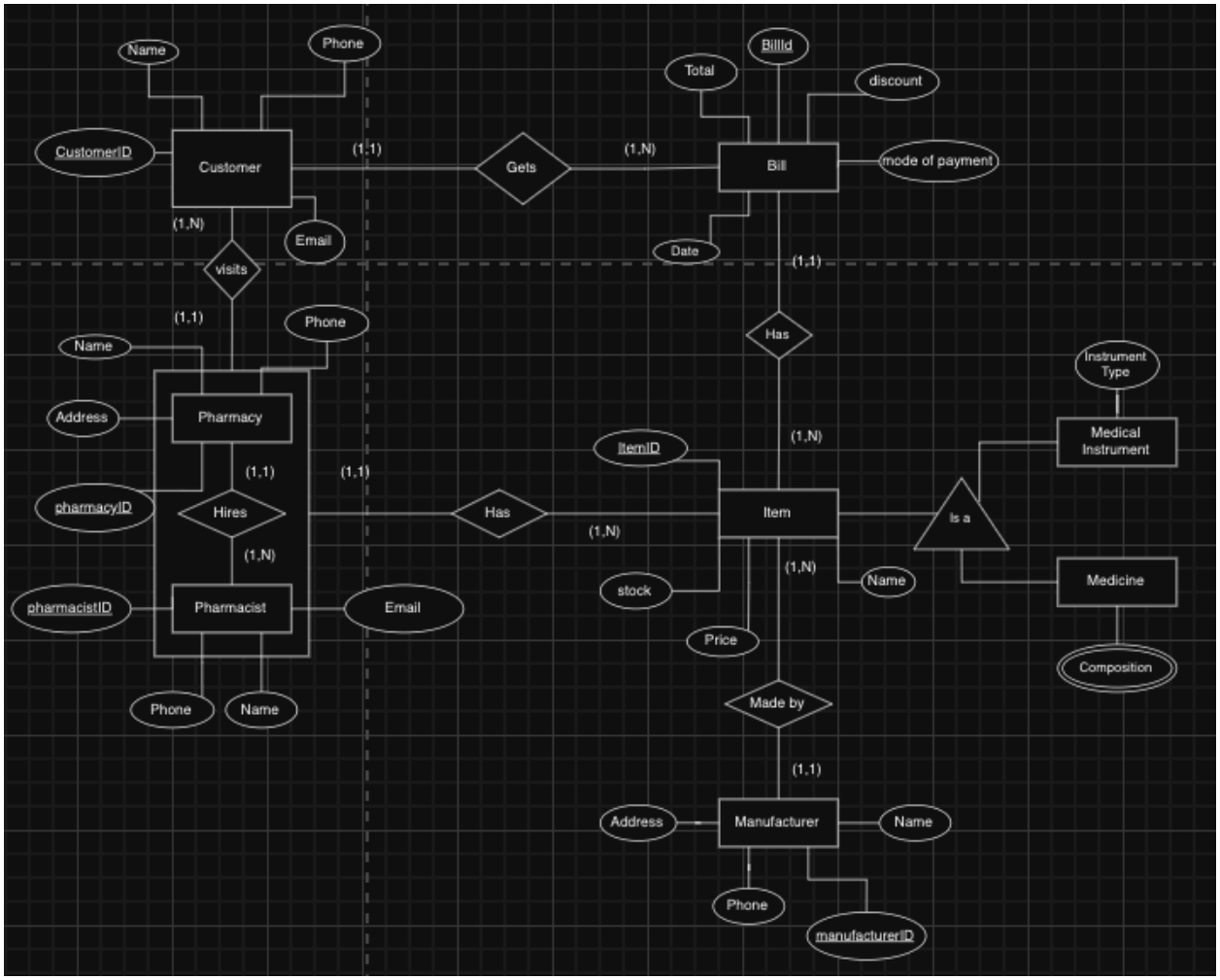
5. **Item**
    - itemID
    - Stock
    - Price
    - Name

6. **Manufacturer**
    - manufacturerID
    - Name
    - Address
    - Phone

7. **Medicine**
    - Type

8. **Medical Instrument**
    - Composition

EER DIAGRAM:


Name
Phone
BillId
Total
discount
CustomerID
Customer
(1,1)
Gets
(1,N)
Bill
mode of payment
(1,N)
Email
Date
(1,1)
visits
(1,1)
Phone
Has
Name
Instrument Type
Address
Pharmacy
ItemID
(1,N)
Medical Instrument
(1,1)
(1,1)
pharmacyID
Hires
Has
Item
Is a
(1,N)
(1,N)
(1,N)
Name
Medicine
pharmacistID
Pharmacist
Email
stock
Price
Composition
Made by
Phone
Name
(1,1)
Address
Manufacturer
Name
Phone
manufacturerID

RELATIONAL SCHEMA:

1. **Pharmacy**

pharmacyID	Name	Address	Phone

- **Primary Key:** pharmacyID
- **Foreign Key:** NULL
- **Candidates Keys:** pharmacyID, Phone
- **Alternate Keys:** name

2. **Pharmacist**

pharmacistID	Phone	Email	Name

- **Primary Key:** pharmacistID
- **Foreign Key:** NULL
- **Candidates Keys:** pharmacistID, Phone
- **Alternate Keys:** Name

3. **Item**

itemID	*pharmacyID*	*manfacturerID*	Stock	Price	Name

- **Primary Key:** medicineID
- **Foreign Key:** pharmacyID, manfacturerID
- **Candidates Keys:** medicineID
- **Alternate Keys:** Name

4. **Manufacturer**

manufacturerID	Name	Phone	Address

- **Primary Key:** manufacturerID
- **Foreign Key:** NULL

- **Candidates Keys:** manufacturerID, Phone
- **Alternate Keys:** Name

5. **Customer**

customerID	Name	Phone	Email

- **Primary Key:** customerID
- **Foreign Key:** NULL
- **Candidates Keys:** customerID, Phone
- **Alternate Keys:** Name

6. **Bill**

billID	*customerID*	Mode_of_payment	Date	Discount	Total

- **Primary Key:** billID
- **Foreign Key:** customerID
- **Candidates Keys:** billID
- **Alternate Keys:** NULL

7.MedicalInstrument

itemID	InstrumentType

- **Primary Key:** itemID
- **Foreign Key:** NULL
- **Candidates Keys:** itemID
- **Alternate Keys:** NULL

8.Medicine

itemID

- **Primary Key:** itemID
- **Foreign Key:** NULL
- **Candidates Keys:** itemID
- **Alternate Keys:** NULL

9. Visits

customerID	pharmacyID	pharmacistID

- **Primary Key:** customerID, pharmacyID, pharmacistID
- **Foreign Key:** NULL
- **Candidates Keys:** customerID, pharmacyID, pharmacistID
- **Alternate Keys:** NULL

10. Has

itemID	pharmacyID	pharmacistID

- **Primary Key:** itemID, pharmacyID, pharmacistID
- **Foreign Key:** NULL
- **Candidates Keys:** itemID, pharmacyID, pharmacistID
- **Alternate Keys:** NULL

11.Hires

pharmacyID	pharmacistID

- **Primary Key:** pharmacyID,
- **Foreign Key:** NULL
- **Candidates Keys:** NULL
- **Alternate Keys:** NULL

12.Composition

compositionID	*itemID*	composition

- **Primary Key:** compositionID
- **Foreign Key:** *itemID*
- **Candidates Keys:** compositionID
- **Alternate Keys:** NULL

CODDS RULES:-

The rules satisfied by our relational model are:

1. **Information Rule:** Data stored in the Relational model must be a value of some cell of a table.

    - This rule is followed in our database as in MySQL, the database is implemented in the form of tables only.

2. **Guaranteed Access Rule**: Every data element must be accessible by the table name, its primary key, and the attribute name whose value will be determined.

    - This rule is followed in our database as every data value can be accessed logically using queries in MySQL.

3. **Systematic Treatment of NULL values:** NULL values in the database only correspond to missing, unknown, or not applicable values.

    - This rule is followed in our database. The NULL value in the database represents missing data in our MySQL database.

4. **Active Online Catalog:** The structure of the database must be stored in an online catalog, which can be queried by authorized users.

    - This rule is followed in our database as we are using MySQL query language to check all the constraints and data in a database.

5. **Comprehensive Data Sublanguage Rule:** A database can only be accessed using a language having a linear syntax that supports data definition, data manipulation, and transaction management operations. This language can be used directly or by means of some application. If the database allows access to data without any help of this language, then it is considered a violation

- This rule is followed in our database as MySQL software is used to implement this database in SQL.

6. **View Updating Rule:** Different views created for various purposes should be automatically updated by the system.

    - This rule is not followed in our database as such a rule is hard to implement . Although updating the view will update the table used for creating it, it is not recommended by most of the database. Hence this rule is not used in most of the databases.

7. **High-level insert, update, and delete rule:** The Relational Model should support insert, delete, update, etc. operations at each level of relations. Also, set operations like Union, Intersection, and minus should be supported.

    - This rule is followed in our database. Our database supports high-level operations, and a single query may be used to change several entries.

8. **Physical data independence:** Any modification in the physical location of a table should not enforce modification at the application level.

    - This rule holds true in our database as changes are recognized through the relational link between the conceptual and internal levels of MySQL.

9. **Logical data independence:** Any modification in the logical or conceptual schema of a table should not enforce modification at the application level. For example, merging two tables into one should not affect the application accessing it, which is difficult to achieve.

- Our database doesn't adhere to this rule. However, achieving this in an ideal scenario is challenging because the logical and user views would be so closely interconnected that they would be nearly identical.

10. **Integrity Independence:** A database must be independent of its application. All its integrity constraints can be independently modified without the need of any change in the application. This rule makes a database independent of the front-end application and its interface.

    - Our database doesn't follow this rule. For example, when we make changes like setting minimum stock value for items or adjusting data ranges, it's crucial for the application to show the right messages or errors. Still, we do stick to fundamental rules like primary key and foreign key constraints as required for our project.

11. **Distribution Independence:** The distribution of data over various locations is not visible to end-users.

    - This rule is not followed in our database. Since we are using the MySQL community edition, this condition is violated. For a distributed database, we will have to use MySQL cluster.

12. **Non-Subversion Rule:** If a system has an interface that provides access to low-level records, then the interface must not be able to subvert the system and bypass security and integrity constraints.

    - Our database adheres to this rule. Users are restricted to accessing data solely through SQL in MySQL, without the option of using any lower-level language. Consequently, queries composed by users within the database are converted into the same low-level syntax, ensuring data integrity is maintained and preventing alterations to the language.