A MINI PROJECT

REPORT

MEDICAL SUPERVISION PORTAL

submitted by

SOLLETI GOWTHAM KUMAR
1NH18CS747
5D

In partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

NEW HORIZON
COLLEGE OF ENGINEERING

Certificate

This is to certify that the mini project work titled

MEDICAL SUPERVISION PORTAL

Submitted in partial fulfillment of the degree of Bachelor of Engineering in Computer Science and Engineering

Submitted by

SOLLETI GOWTHAM KUMAR

1NH18CS747

DURING

ODD SEMESTER 2020-2021

For 20CSE59

Signature of Reviewer Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner	*Signature with date*
1.____________________________	____________________________
2.____________________________	____________________________

ABSTRACT

The aim of this project is to build a software which is very useful for the hospital digitalization, which is main used to maintain the every record safe and secure.

The Medical Supervision Portal is a software product suite designed to improve the quality and management of hospital management in the areas of clinical process analysis and activity-based costing. Hospital Management System enables you to develop your organization and improve its effectiveness and quality of work

The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically, here in this medical supervision portal will digitalize every record regarding to patients and staff, there is no more use of storing the paper records. The Medical Supervision Portal can be entered using a username and password. It is accessible either by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. User can search availability of a doctor and the details of a patient using the id we can do anything in this portal. User can also add his or her appointment and also delete his appointment, if any one join as inpatient in hospital then with of help of user's unique ID, all information regarding to the patient will be stored in the database. Admin can retrieve the any patient data when ever they want. The Medical Supervision Portal is designed for multispecialty hospitals, to cover a wide range of hospital administration processes. It is an integrated end-to-end Hospital Management System that provides relevant information across the hospital to support effective decision making for patient care, hospital administration and critical financial accounting, in a seamless flow.

This is main theme of the project.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal NHCE, for his constant support and encouragement.

I would also like to thank Dr**. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Ms. Yogitha,** Assistant Professor, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

SOLLETI GOWTHAM KUMAR (1NH18CS747)

TABLE OF CONTENTS

CHAPTERS

LIST OF FIGURES

LIST OF SCREENSHOTS

CHAPTER 1

INTRODUCTION

In this project we have programmed the working and functioning of using Python and also include Dbms.

1.1 Medical Supervision Portal Classification:

The Main theme of this project Medical Supervision Portal is it includes registration of patients, storing their details into the system, and also computerized billing in the pharmacy, and labs. The software has the facility to give a unique id for every patient and stores the details of every patient and the staff automatically. It includes a search facility to know the current status of each room. User can search availability of a doctor and the details of a patient using the id we can do anything in this portal.

1.2 Objectives:

The Medical Supervision Portal can be entered using a username and password. It is accessible either by an administrator or receptionist. Only they can add data into the database. The data can be retrieved easily. The interface is very user-friendly. The data are well protected for personal use and makes the data processing very fast. This Medical Supervision Portal is powerful, flexible, and easy to use and is designed and developed to deliver real conceivable benefits to hospitals

The Medical Supervision Portal is designed for multispeciality hospitals, to cover a wide range of hospital administration processes. It is an integrated end-to-end Hospital Management System that provides relevant information across the hospital to support effective decision making for patient care, hospital administration and critical financial accounting, in a seamless flow.

The Medical Supervision Portal is a software product suite designed to improve the quality and management of hospital management in the areas of clinical process analysis and activity-based costing. Hospital Management System enables you to develop your organization and improve its effectiveness and quality of work. Managing the key processes efficiently is critical to the success of the hospital helps you manage your processes.

1.3 Problem Definition

Lack of immediate information storage: -

The information generated by various transactions takes time and efforts to be stored at right place.

Lack of prompt updating: -

Various changes to information like patient details or details of child are difficult to make as paper work is involved.

Error in manual calculation: -

Manual calculations are error prone and take a lot of time this may result in incorrect information. For example calculation of patient's bill based on various treatments.

Preparation of accurate and prompt reports: -

This becomes a difficult task as information is difficult to collect from various register.

1.4 Expected outcomes

The entire project mainly consists of 7 modules, which are:

- ❖ Admin module
- ❖ User module (patient)
- ❖ Department module
- ❖ Doctor module
- ❖ Pharmacist module
- ❖ Accountant module.

CHAPTER 2

REQUIREMENT AND SPECIFICATIONS

2.1 **Hardware and Software requirements**

Hardware Requirements:

- Processor: Any processor above 500MHz.
- RAM: 4 GB+ RAM or more.
- Space: 500 GB

Software Requirements:

- Operation system: Windows 7 / Windows 8 / Windows 10/Mac os X
- User interface: Python
- Language: Python

CHAPTER 3

DATA MODELS AND ER DIAGRAM

3.1 Data Models:

• A data model—a collection of concepts that can be used to describe the structure of a database—provides the necessary means to achieve the abstraction.

• Most data models also include a set of basic operations for specifying retrievals and updates on the database. Categories of Data Models: ¬ High-level or conceptual data models ¬ Low-level or physical data models ¬ Representational (or implementation) data model

i) **High-level or conceptual data models:** Conceptual data models use concepts such as entities, attributes, and relationships. ¬ An entity represents a real-world object or concept, such as an employee or a project from the mini world that is described in the database. ¬ An attribute represents some property of interest that further describes an entity, such as the employee’s name or salary. ¬ A relationship among two or more entities represents an association among the entities, for example, a works-on relationship between an employee and a project. ¬ The entity–relationship model—a popular high-level conceptual data model.

ii) **Low-level or physical data models:** ¬ Physical data models describe how data is stored as files in the computer by representing information such as record formats, record orderings, and access paths. ¬ An access path is a search structure that makes the search for particular database records efficient, such as indexing or hashing.

iii) **Representational (or implementation) data model:** It is used most frequently in traditional commercial DBMSs. These include the widely used relational data model, as well as the so-called legacy data models—the network and hierarchical models.

• Representational data models represent data by using record structures and hence are sometimes called record-based data models.

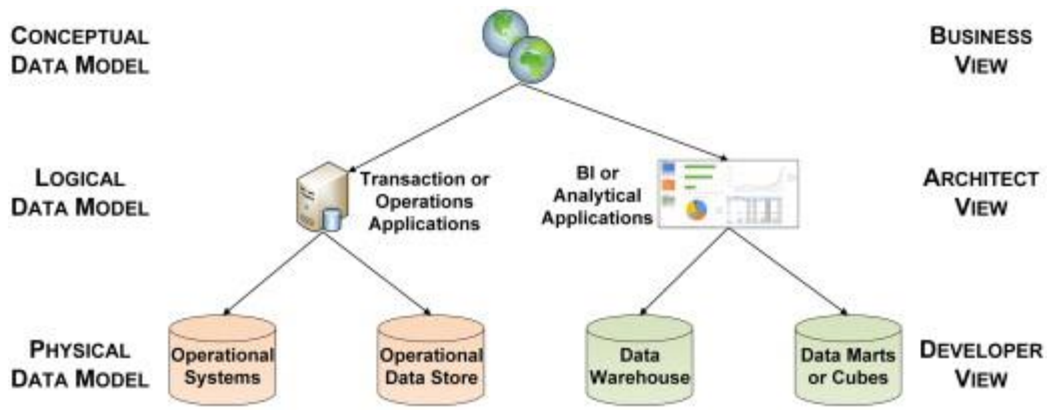


Fig-1

3.1.1 Entities and Attributes:

Entities:

- The basic concept that the ER model represents is an entity, which is a thing or object in the real world with an independent existence.
- An entity may be an object with a physical existence (for example, a particular person, car, house, or employee) or it may be an object with a conceptual existence (for instance, a company, a job, or a university course).

Attributes:

- Entities are represented by means of their **properties**, called attributes.

  Types Of Attributes:

  1.Key: Key attributes are those attributes which can identify an entity uniquely in an entity set.

  2.Composite: Composite attributes are those attributes which are composed of many other simple attributes.

  3.Multivalued: Multi valued attributes are those attributes which can take more than one value for a given entity from an entity set

  4.Derived: Derived attributes are those attributes which can be derived from other attribute(s)

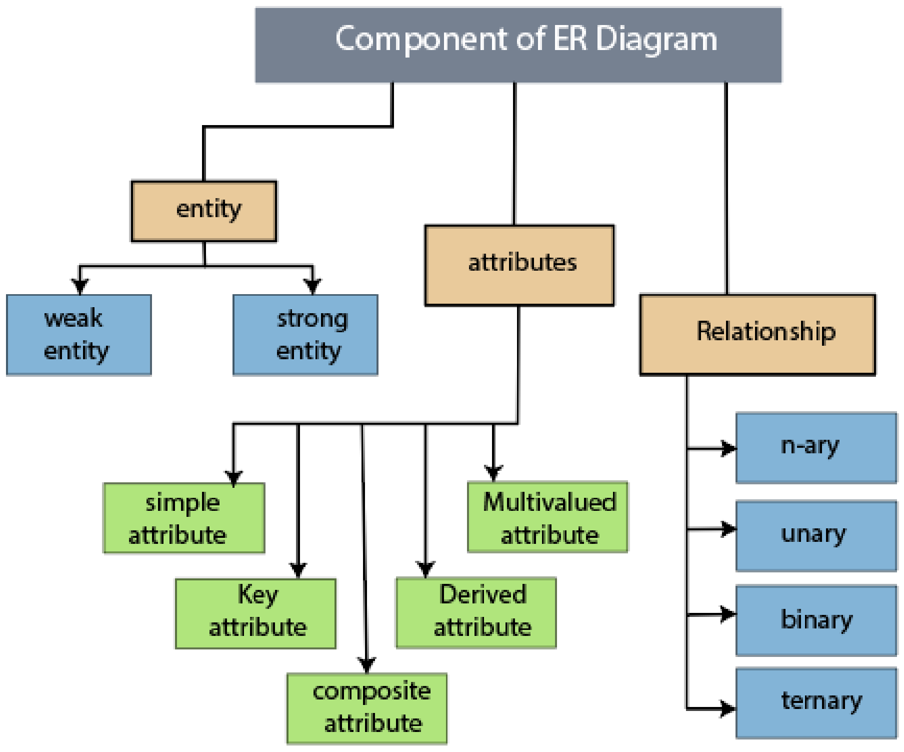


Fig 2

3.1.2 Keys

- Keys play an important role in the relational database.
- It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between tables.
- Types of keys:

1.Primary Key :

Primary key is the one which helps in unique identification of each data record present in the tables. It can be a single attribute of the table or combination of attribute.
Primary key can also be called as a key which is most suitably chosen from set of candidate keys.

2.Candidate Key:

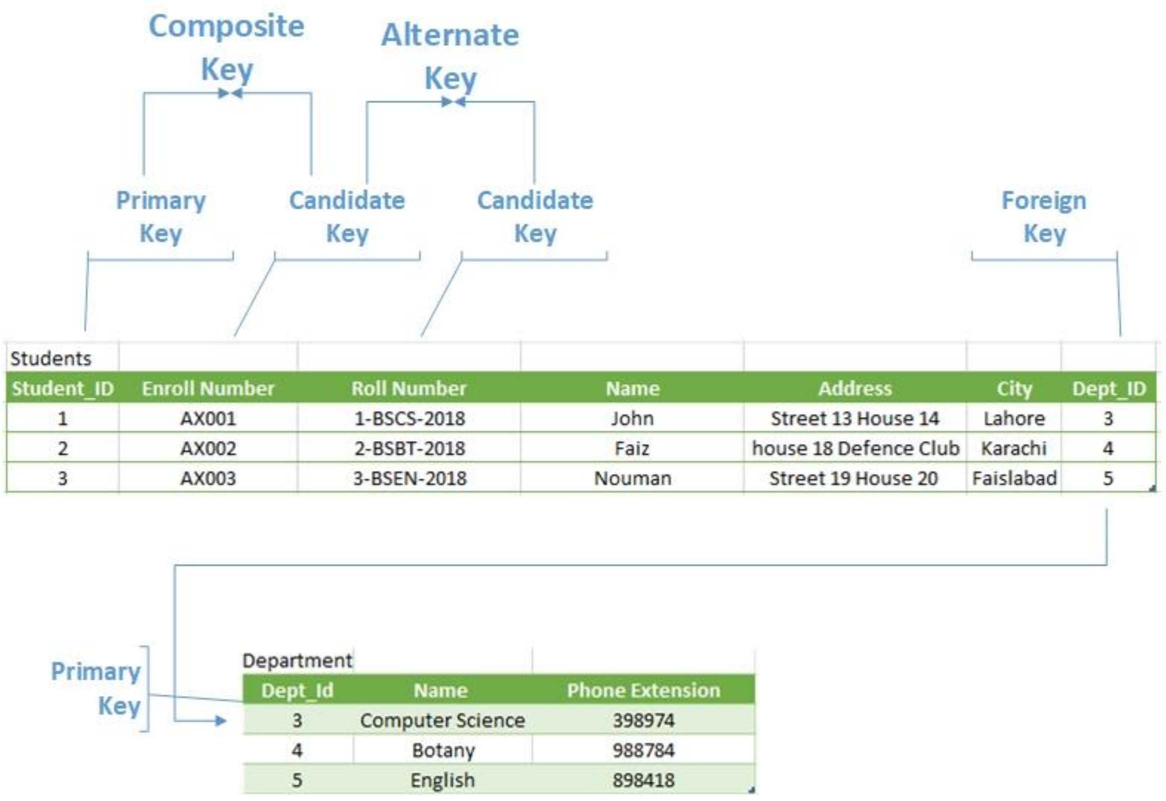
Candidate keys are similar to primary key. The only difference is, primary key for a table can only be one but candidate key can be more than one.

3.Super Key:

Super key is the one which is able to determine any tuples/data/entity from record table. Super key can be a single attribute or a set of attributes.

4.Foriegn Key:

A foreign key is nothing but an attribute that is commonly linked between two relation using that same attribute. Both the relations/tables must contain the same attribute.



Students

Student_ID	Enroll Number	Roll Number	Name	Address	City	Dept_ID
1	AX001	1-BSCS-2018	John	Street 13 House 14	Lahore	3
2	AX002	2-BSBT-2018	Faiz	house 18 Defence Club	Karachi	4
3	AX003	3-BSEN-2018	Nouman	Street 19 House 20	Faislabad	5

Department

Dept_Id	Name	Phone Extension
3	Computer Science	398974
4	Botany	988784
5	English	898418

Fig 3

. 3.1.3 Relation and Participation

Relationships are represented by diamond-shaped box. Name of the relationship is written inside the diamond-box. All the entities (rectangles) participating in a relationship, are connected to it by a line.

1.One to one

2.One to many

3.Many to one

4.Many to many

1.One to one: When only one instance of an entity is associated with the relationship, it is marked as '1:1'

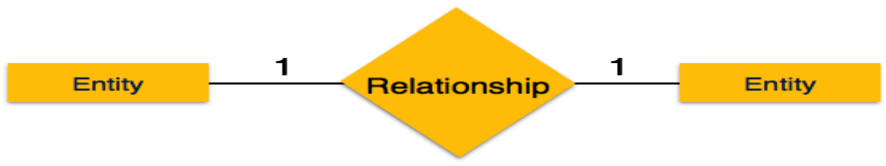


Fig 4

2. **One-to-many** – When more than one instance of an entity is associated with a relationship, it is marked as '1:N'. The following image reflects that only one instance of entity on the left and more than one instance of an entity on the right can be associated with the relationship. It depicts one-to-many relationship

3. **Many-to-one** – When more than one instance of entity is associated with the relationship, it is marked as 'N:1'. The following image reflects that more than one instance of an entity on the left and only one instance of an entity on the right can be associated with the relationship. It depicts many-to-one relationship.

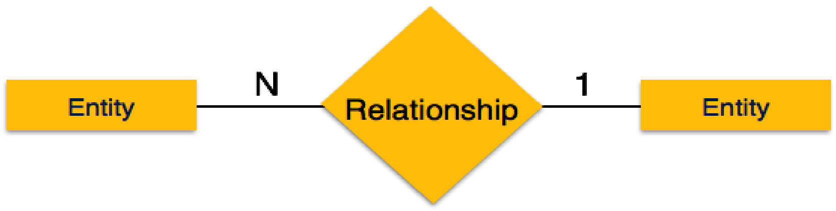


Fig 5

4. **Many-to-many** – The following image reflects that more than one instance of an entity on the left and more than one instance of an entity on the right can be associated with the relations hip. It depicts many-to-many relationship.

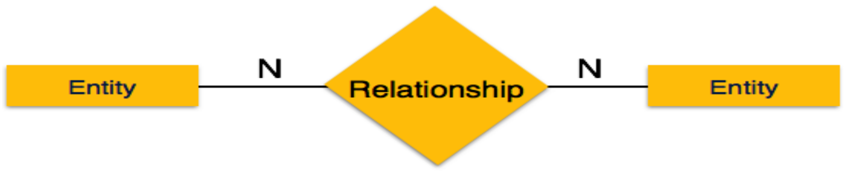


Fig 6

Participation Constraints:

- **Total Participation** – Each entity is involved in the relationship. Total participation is represented by double lines.
- **Partial participation** – Not all entities are involved in the relationship. Partial participation is represented by single lines.

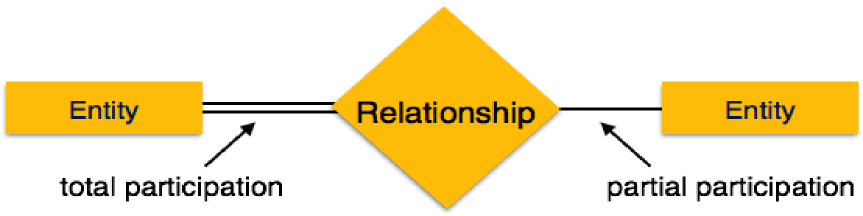


Fig 7

3.2 DBMS FUNDAMENTALS:

SQL:

- SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.
- SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

RDBMS

- RDBMS stands for Relational Database Management System.
- RDBMS is the basis for SQL, and for all modern database systems such as MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.
- The data in RDBMS is stored in database objects called tables. A table is a collection of related data entries and it consists of columns and rows.
- Every table is broken up into smaller entities called fields. The fields in the Customers table consist of CustomerID, CustomerName, ContactName, Address, City, PostalCode and Country. A field is a column in a table that is designed to maintain specific information about every record in the table.
- A record, also called a row, is each individual entry that exists in a table. For example, there are 91 records in the above Customers table. A record is a horizontal entity in a table.
- A column is a vertical entity in a table that contains all information associated with a specific field in a table.

SQL DML, DDL, DCL, and TCL Commands:

What are SQL DML, DDL, DCL, and TCL Commands, and their abbreviations are:

SQL DDL Commands
In SQL, DDL means Data Definition Language. The Sql Server DDL commands are used to create and modify the structure of a database and database objects.
Examples of Sql Server DDL commands are
CREATE – Create an object. I mean, create a database, table, triggers, index, functions, stored procedures, etc.
DROP – This SQL DDL command helps to delete objects. For example, delete tables, delete a database, etc.
ALTER – Used to alter the existing database or its object structures.

TRUNCATE – This SQL DDL command removes records from tables
RENAME – Renaming the database objects

SQL DML Commands

DML means Data Manipulation Language in Sql Server. As its name suggests, these Sql Server DML commands will perform data manipulation manipulate data presented in the server).
Examples of DML commands in SQL Server are
SELECT – This SQL DML command select records or data from a table
INSERT – Insert data into a database table.
UPDATE – This SQL DML command will update existing records within a table
DELETE – Delete unwanted records from a table

SQL DCL Commands

The Sql Server DCL means Data Control Language. These DCL commands in SQL Server will control the Data access permission.
Sql Server DCL commands Examples are
GRANT – It permits users to access the database.
REVOKE – This SQL DCL command withdraws the permission given by GRANT to access the database.

SQL TCL Commands

TCL means Transaction Control Language. These Sql Server TCL commands will control the Transactions. Please refer SQL interviews.
Examples of TCL commands in SQL Server are
COMMIT – This SQL TCL command will commit the running transaction
ROLLBACK – Rollback the current transaction
SAVEPOINT – You can set a save point so that, next time it will start from here
SET TRANSACTION – Specify the characteristics of the transactions .

3.3 PYTHON FEATURES

Python provides many useful features which make it popular and valuable from the other programming languages. It supports object-oriented programming, procedural programming approaches and provides dynamic memory allocation. We have listed below a few essential features.

1) Easy to Learn and Use

Python is easy to learn as compared to other programming languages. Its syntax is straightforward and much the same as the English language. There is no use of the semicolon or curly-bracket, the indentation defines the code block. It is the recommended programming language for beginners.

2) Expressive Language

Python can perform complex tasks using a few lines of code. A simple example, the hello world program you simply type **print("Hello World")**. It will take only one line to execute, while Java or C takes multiple lines.

3) Interpreted Language

Python is an interpreted language; it means the Python program is executed one line at a time. The advantage of being interpreted language, it makes debugging easy and portable.

4) Cross-platform Language

Python can run equally on different platforms such as Windows, Linux, UNIX, and Macintosh, etc. So, we can say that Python is a portable language. It enables programmers to develop the software for several competing platforms by writing a program only once.

5) Free and Open Source

Python is freely available for everyone. It is freely available on its official website www.python.org. It has a large community across the world that is dedicatedly working towards make new python modules and functions. Anyone can contribute to the Python community. The open-source means, "Anyone can download its source code without paying any penny."

6) Object-Oriented Language

Python supports object-oriented language and concepts of classes and objects come into existence. It supports inheritance, polymorphism, and encapsulation, etc. The object-oriented procedure helps to programmer to write reusable code and develop applications in less code.

7) Extensible

It implies that other languages such as C/C++ can be used to compile the code and thus it can be used further in our Python code. It converts the program into byte code, and any platform can use that byte code.

8) Large Standard Library

It provides a vast range of libraries for the various fields such as machine learning, web developer, and also for the scripting. There are various machine learning libraries, such as Tensor flow, Pandas, Numpy, Keras, and Pytorch, etc. Django, flask, pyramids are the popular framework for Python web development.

9) GUI Programming Support

Graphical User Interface is used for the developing Desktop application. PyQT5, Tkinter, Kivy are the libraries which are used for developing the web application.

10) Integrated

It can be easily integrated with languages like C, C++, and JAVA, etc. Python runs code line by line like C,C++ Java. It makes easy to debug the code.

11. Embeddable

The code of the other programming language can use in the Python source code. We can use Python source code in another programming language as well. It can embed other language into our code.

12. Dynamic Memory Allocation

In Python, we don't need to specify the data-type of the variable. When we assign some value to the variable, it automatically allocates the memory to the variable at run time. Suppose we are assigned integer value 15 to **x,** then we don't need to write **int x = 15.** Just write x = 15

NUMBERS:

Type			Format			Description
int			a = 10			Signed Integer
long			a = 345L			(L) Long integers, they can also be represented in octal and hexadecimal
float			a = 45.67			(.) Floating point real values
complex			a = 3.14J			(J) Contains integer in the range 0 to 255.

Most of the time Python will do variable conversion automatically. You can also use Python conversion functions (int(), long(), float(), complex()) to convert data from one type to another. In addition, the `type` function returns information about how your data is stored within a variable.

3.2.1 String

- Create string variables by enclosing characters in quotes. Python uses single quotes `'` double quotes `"` and triple quotes `"""` to denote literal strings. Only the triple quoted strings `"""` also will automatically continue across the end of line statement.
- Strings can be accessed as a whole string, or a substring of the complete variable using brackets '[]'. Here are a couple examples:
- Python can use a special syntax to format multiple strings and numbers. The string formatter is quickly covered here because it is seen often and it is important to recognize the syntax.

3.2.2 List

- Lists are a very useful variable type in Python. A list can contain a series of values. List variables are declared by using brackets `[ ]` following the variable name.
- Lists aren't limited to a single dimension. Although most people can't comprehend more than three or four dimensions. You can declare multiple dimensions by separating an with commas. In the following example, the MyTable variable is a two-dimensional array

3.2.3Tuple

- Tuples are a group of values like a list and are manipulated in similar ways. But, tuples are fixed in size once they are assigned.
- In Python the fixed size is considered immutable as compared to a list that is dynamic and
- mutable. Tuples are defined by parenthesis ().

3.2.4 Set

- Sets are used to store multiple items in a single variable.
- Set is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Tuple, and Dictionary, all with different qualities and usage.
- A set is a collection which is both *unordered* and *unindexed*.
- Sets are written with curly brackets

3.2.5Dictionary

- Dictionaries in Python are lists of `Key`:`Value` pairs. This is a very powerful datatype to hold a lot of related information that can be associated through `keys`. The main operation of a dictionary is to extract a value based on the `key` name. Unlike lists, where index numbers are used, dictionaries allow the use of a `key` to access its members. Dictionaries can also be used to sort, iterate and compare data.
- Dictionaries are created by using braces ({}) with pairs separated by a comma (,) and the key values associated with a colon(:). In Dictionaries the `Key` must be unique. Here is a quick example on how dictionaries might be used.

3.3 TKINTER WIDGET

Python provides various options for developing graphical user interfaces (GUIs). Most important are listed below.

- **Tkinter** − Tkinter is the Python interface to the Tk GUI toolkit shipped with Python. We would look this option in this chapter.
- **wxPython** − This is an open-source Python interface for wxWindows http://wxpython.org.
- **JPython** − JPython is a Python port for Java which gives Python scripts seamless access to Java class libraries on the local machine http://www.jython.org.

There are many other interfaces available, which you can find them on the net.

Tkinter Programming

Tkinter is the standard GUI library for Python. Python when combined with Tkinter provides a fast and easy way to create GUI applications. Tkinter provides a powerful object-oriented interface to the Tk GUI toolkit.

Creating a GUI application using Tkinter is an easy task. All you need to do is perform the following steps −

- Import the *Tkinter* module.
- Create the GUI application main window.
- Add one or more of the above-mentioned widgets to the GUI application.
- Enter the main event loop to take action against each event triggered by the user.

Example

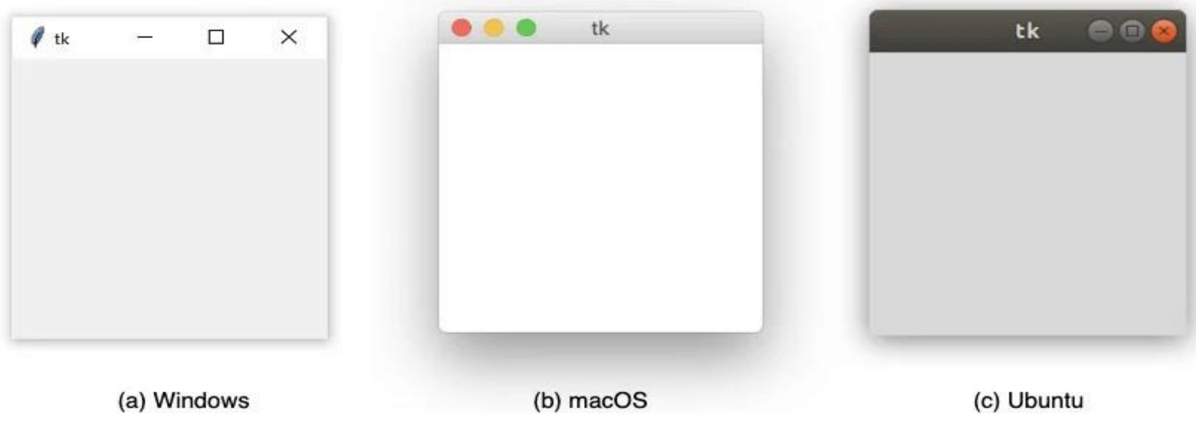


Fig 8

```
#!/usr/bin/python

import Tkinter
top = Tkinter.Tk()
# Code to add widgets will go here...
top.mainloop()
```

This would create a following window −

Tkinter Widgets

Tkinter provides various controls, such as buttons, labels and text boxes used in a GUI application. These controls are commonly called widgets.

3.3.1 Button

The Button widget is used to add buttons in a Python application. These buttons can display text or images that convey the purpose of the buttons. You can attach a function or a method to a button which is called automatically when you click the button.

Syntax

Here is the simple syntax to create this widget −

```
w = Button ( master, option=value, ... )
```

Parameters

- **master** − This represents the parent window.

- **options** − Here is the list of most commonly used options for this widget. These options can be used as key-value pairs separated by commas.

Sr.No.	Option & Description
1	**activebackground** Background color when the button is under the cursor.
2	**activeforeground** Foreground color when the button is under the cursor.
3	**bd** Border width in pixels. Default is 2.
4	**bg** Normal background color.
5	**command** Function or method to be called when the button is clicked.

3.3.2 Label

This widget implements a display box where you can place text or images. The text displayed by this widget can be updated at any time you want.

It is also possible to underline part of the text (like to identify a keyboard shortcut) and span the text across multiple lines.

Syntax

Here is the simple syntax to create this widget −

```
w = Label ( master, option, ... )
```

Parameters

- **master** − This represents the parent window.
- **options** − Here is the list of most commonly used options for this widget. These options can be used as key-value pairs separated by commas.

Sr.No.	Option & Description
1	**anchor** This options controls where the text is positioned if the widget has more space than the text needs. The default is anchor=CENTER, which centers the text in the available space.
2	**bg** The normal background color displayed behind the label and indicator.

3.3.3 Entry

The Entry widget is used to accept single-line text strings from a user.

- If you want to display multiple lines of text that can be edited, then you should use the *Text* widget.
- If you want to display one or more lines of text that cannot be modified by the user, then you should use the *Label* widget.

Syntax

Here is the simple syntax to create this widget −

w = Entry(master, option, ...)

Parameters

- **master** − This represents the parent window.
- **options** − Here is the list of most commonly used options for this widget. These options can be used as key-value pairs separated by commas.

Sr.No.	Option & Description
1	**bg** The normal background color displayed behind the label and indicator.
2	**bd** The size of the border around the indicator. Default is 2 pixels.

3	**command** A procedure to be called every time the user changes the state of this checkbutton.
4	**cursor** If you set this option to a cursor name (*arrow, dot etc.*), the mouse cursor will change to that pattern when it is over the checkbutton.

3.3.4 Tree View

The TreeView widget is very useful if you want to display a hierarchy of items, with all attributes listed side by side.

For example, if you want to construct an application which looks like the Windows File Explorer, we can do this using Tkinter’s TreeView widget.

Treeview widget is used to choose a numeric value through sliders. The syntax for treeview widget is shown below.

treeview treeviewName options
Options

Sr.No.	Syntax & Description
1	**-columns columnNames** An array of column names for widget.
2	**-displaycolumns columns** An array of column names or indices specifying columns to be displayed. Use #all for all.
3	**-height number** Height for widget.

4	**-selectmode mode** Selection mode which can be extended, browse, or none.

The options available for the treeview widget are listed below in table.

CHAPTER 4

DATABASE/ALGORITHM DESIGN

The relational model represents the database as a collection of relations. Informally, each relation resembles a table of values or, to some extent, a flat file of records. It is called a flat file because each record has a simple linear or flat structure.

Tables

–In relational data model, relations are saved in the format of Tables. This format stores the relation among entities. A table has rows and columns, where rows represents records and columns represent the attributes.

Tuple

–A single row of a table, which contains a single record for that relation is called a tuple.

Relation instance

– A finite set of tuples in the relational database system represents relation instance. Relation instances do not have duplicate tuples.

Relation schema

A relation schema describes the relation name (table name), attributes, and their names.

Relation key

− Each row has one or more attributes, known as relation key, which can identify the row in the relation (table) uniquely.

Attribute domain

− Every attribute has some pre-defined value scope, known as attribute domain.

Constraints:

Every relation has some conditions that must hold for it to be a valid relation. These conditions are called **Relational Integrity Constraints**. There are three main integrity constraints −

- Key constraints
- Domain constraints
- Referential integrity constraints

Key constraints force that −

In a relation with a key attribute, no two tuples can have identical values for key attributes. A key attribute can not have NULL values.

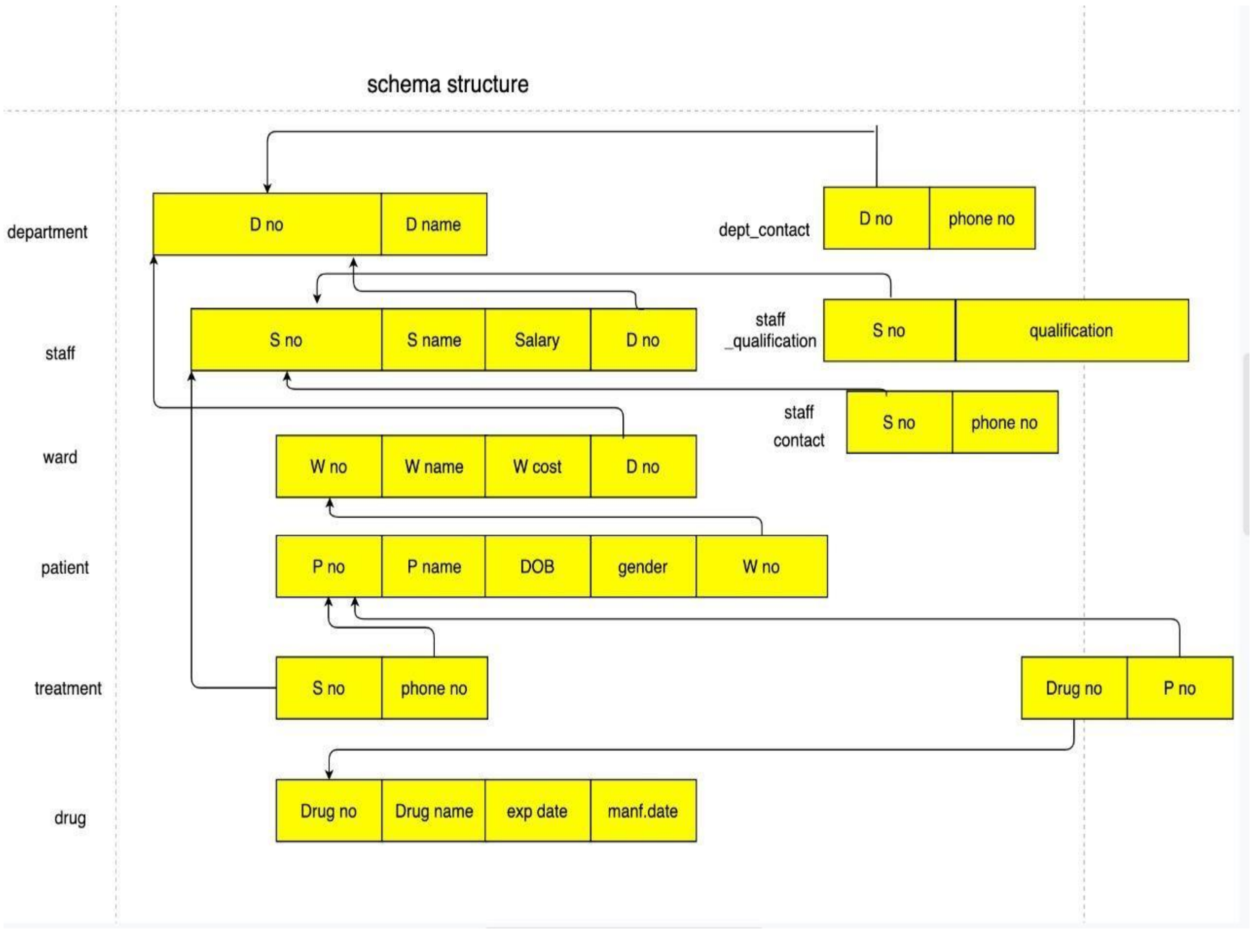
Key constraints are also referred to as Entity Constraints.

Domain Constraints:
Attributes have specific values in real-world scenario. For example, age can only be a positive integer.
The same constraints have been tried to employ on the attributes of a relation.
Referential integrity Constraints:
Referential integrity constraints work on the concept of Foreign Keys. A foreign key is a key attribute of a relation that can be referred in other relation.
Referential integrity constraint states that if a relation refers to a key attribute of a different or same relation, then that key element must exist.

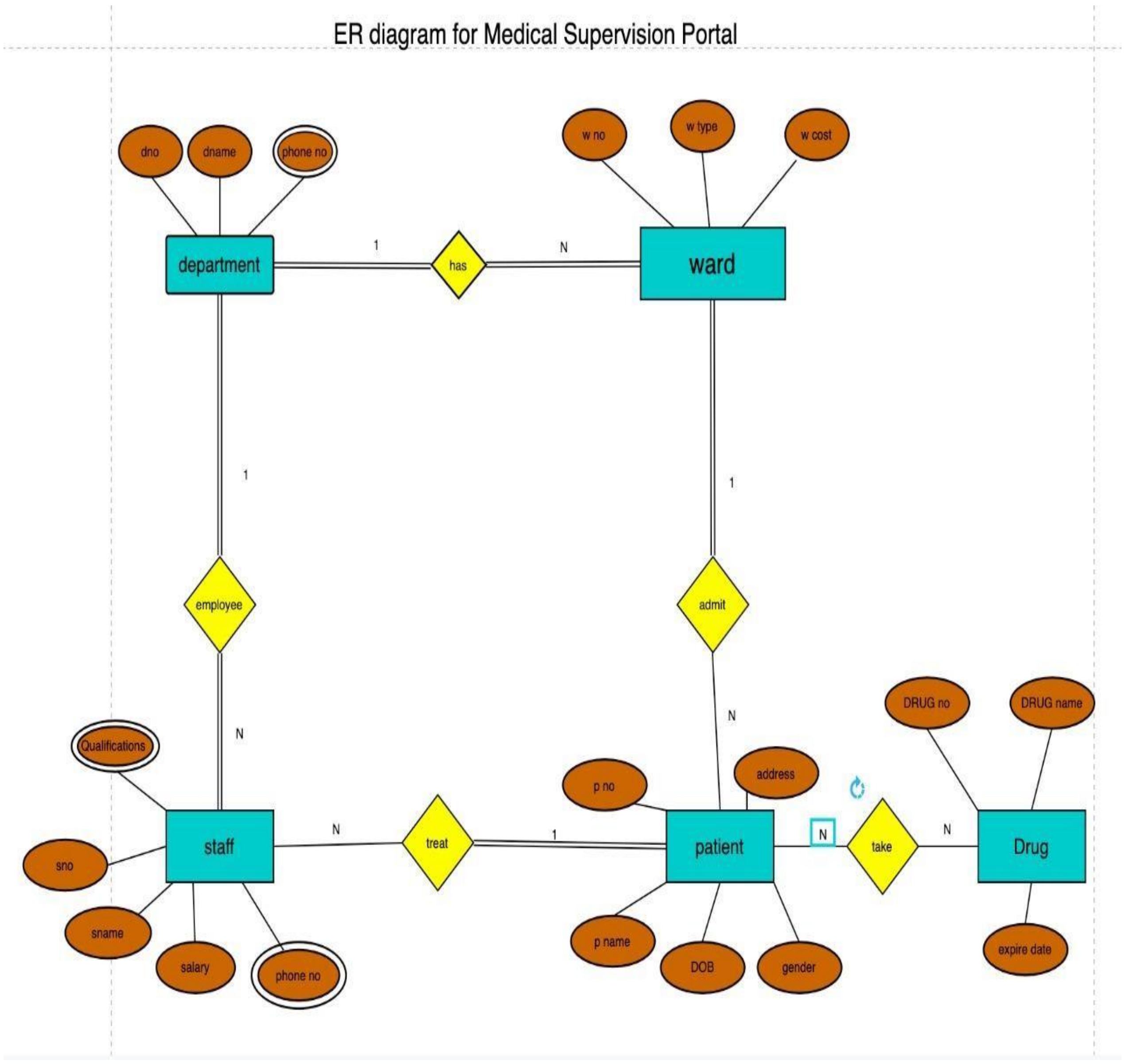
• Connecting code to database

```
conn = sqlite3.connect("C:\Pharma Lifeline\database\store.db")

c = conn.cursor()


result = c.execute("SELECT Max(id) from inventory")

for r in result:

   id = r[0]
```

SCHEMA STRUCTURE



ER Diagram:


ER diagram for Medical Supervision Portal
dno
dname
phone no
department
1
has
N
ward
w no
w type
w cost
1
employee
N
1
admit
N
Qualifications
staff
sno
sname
salary
phone no
N
treat
1
patient
p no
address
p name
DOB
gender
N
take
N
Drug
DRUG no
DRUG name
expire date

CHAPTER 5

IMPLEMENTATION

Displays the Menu Card in which it has options

- Patient registration
- Room allocation
- Employee registration
- Book appointment
- Patient bill
- exit

APPOINTMENT CREATION:

```
def set():
    global e3,e1,e2,e4,e5,e6,conn
    p1=e1.get()
    p2=e2.get()
    p3=e3.get(tkinter.ACTIVE)
    p4=e4.get()
    p5=e5.get()
    p6=e6.get(1.0,tkinter.END)
    conn = sqlite3.connect("MSP.db")
    conn.execute("Insert into appointment values(?,?,?,?,?,?)",(p1,p2,p3,p4,p5,p6,))
    conn.commit()
    tkinter.messagebox.showinfo("MEDICAL DATABASE SYSTEM", "APPOINTMENT SET
SUCCSESSFULLY")
```

APPOINTMENT DELETE:

```
def remove():
    global e7,edd
    edd=str(e7.get())
    v=list(conn.execute("select * from appointment where AP_NO=?", (edd,)))
    if (len(v)==0):
        errorD = tkinter.Label(rootAA, text="PATIENT APPOINTMENT NOT FIXED",fg="red")
        errorD.place(x=20,y=420)
    else:
        conn.execute('DELETE FROM PATIENT where PATIENT_ID=?',(edd,))
        disd1=tkinter.Label(rootAA,text="PATIENT APPOINTMENT DELETED",fg='green')
        disd1.place(x=20,y=420)
        conn.commit()
```

APPOINTMENT SEARCH:

```
def viewappointment():
    global e8
    ap=str(e8.get())
    vv = list(conn.execute("select * from appointment where AP_DATE=?", (ap,)))
    if (len(vv) == 0):
        errorD = tkinter.Label(rootAA, text="NO APPOINTMENT FOR TODAY", fg="red")
        errorD.place(x=20, y=420)
    else:
        s=conn.execute("Select PATIENT_ID,NAME,AP_NO,EMP_NAME,AP_DATE,AP_TIME from 
PATIENT NATURAL JOIN employee NATURAL JOIN appointment where AP_DATE=?",(ap,))
        for i in s:
            s1=tkinter.Label(rootAP,text=i,fg='green')
            s1.place(x=10,y=85)
```

- Here the given information for the appointment will be stored in the database then this data is used to retrive the data to search and delete appointment.
- In delete definition if we want to delete the appointment then click on the delete button then if appointment is already stored then it will delete otherwise it shows appointment not found.
- In search def if any appointment is registered it is easy to find the appointment all the records.

BILLING PAGE:

```
 def date_up():
    global b1,b2
    b1 = P_id.get()
    b2 = dd.get()
    conn.execute("UPDATE ROOM SET DATE_DISCHARGED=? where PATIENT_ID=?", (b2, b1,))
    conn.commit()
    tkinter.messagebox.showinfo("MEDICAL DATABASE SYSTEM", "DISCHARGE DATE UPDATED")

def up():
    global c1, b1, P_id, b3, b4, b5, b6, dd, treat_1, treat_2, cost_t, b7, b8, med, med_q, price, u
    conn = sqlite3.connect("MSP.db")
    c1 = conn.cursor()
    b1 = P_id.get()
    b3 = treat_1.get(tkinter.ACTIVE)
    b4 = treat_2.get(tkinter.ACTIVE)
    b5 = cost_t.get()
    b6 = med.get(tkinter.ACTIVE)
    b7 = med_q.get(tkinter.ACTIVE)
    b8 = price.get()
    conn.execute("INSERT INTO TREATMENT VALUES(?,?,?,?)", (b1, b3, b4, b5,))
    conn.execute("INSERT INTO MEDICINE VALUES(?,?,?,?)", (b1, b6, b7, b8,))
    conn.commit()
    tkinter.messagebox.showinfo("MEDICAL DATABASE SYSTEM", "BILLING DATA SAVED")
```

- Whenever the patient discharges it updates the date of discharge which is used to make whole bill of the patient.
- This page calculates the medical bill, doctor bill, room bill, and whole sums and shows the full amount of the patient

INPUT PATIENT FORM:

```
def IN_PAT():
    global pp1, pp2, pp3, pp4, pp5, pp6, pp7, pp8, pp9, pp10,ce1,conn
    conn=sqlite3.connect("MSP.db")
    conn.cursor()
    pp1=pat_ID.get()
    pp2=pat_name.get()
    pp3=pat_sex.get()
    pp4=pat_BG.get()
    pp5=pat_dob.get()
    pp6=pat_contact.get()
    pp7=pat_contactalt.get()
    pp8=pat_address.get()
    pp9=pat_CT.get()
    pp10=pat_email.get()
    conn.execute('INSERT INTO PATIENT 
VALUES(?,?,?,?,?,?,?,?)',(pp1,pp2,pp3,pp4,pp5,pp8,pp9,pp10,))
    conn.execute('INSERT INTO CONTACT_NO VALUES (?,?,?)',(pp1,pp6,pp7,))
    tkinter.messagebox.showinfo("MEDICAL DATABSE SYSTEM","DETAILS INSERTED INTO 
DATABASE")
    conn.commit()
```

- This form is used to store all the information of inpatient.

ROOM BUTTON:

```
def room_button():
    global P_id,r1,r2,room_t,da,dd,rate,room_no,r3,r4,r5,r6,conn
    conn = sqlite3.connect("MSP.db")
    r1=P_id.get()
    r2=room_t.get(tkinter.ACTIVE)
    r3=room_no.get(tkinter.ACTIVE)
    r4=rate.get()
    r5=da.get()
    r6=dd.get()
    conn.execute('INSERT INTO ROOM VALUES(?,?,?,?,?,?)',(r1,r3, r2, r4, r5, r6,))
    tkinter.messagebox.showinfo("MEDICAL DATABSE SYSTEM", "ROOM ALLOCATED")
    conn.commit()
    conn.close()

def update_button():
    global P_id,r1,r2,room_t,da,dd,rate,room_no,r3,r4,r5,r6,conn
    r1=P_id.get()
    r2=room_t.get(tkinter.ACTIVE)
    r3=room_no.get(tkinter.ACTIVE)
    r4=rate.get()
    r5=da.get()
    r6=dd.get()
    p = list(conn.execute("Select * from ROOM where PATIENT_ID=?", (r1,)))
    if len(p) != 0:
        conn.execute('UPDATE ROOM SET
ROOM_NO=?,ROOM_TYPE=?,RATE=?,DATE_ADMITTED=?,DATE_DISCHARGED=? where
PATIENT_ID=?',(r3, r2, r4, r5, r6,r1,))
        tkinter.messagebox.showinfo("MEDICAL DATABSE SYSTEM", "ROOM DETAILS UPDATED")
        conn.commit()
    else:
        tkinter.messagebox.showinfo("MEDICAL DATABSE SYSTEM", "PATIENT IS NOT ALLOCATED A
ROOM")
##ROOT FOR DISPLAY ROOM INFO
rootRD=None

##EXIT FOR ROOM_PAGE
def EXITT():
    global rootR
    rootR.destroy()
```

- In this room allotment def if the patient wants to join in the hospital then the admin use to store all the records of patient, and all the medications which is used by the patient.
- We can update the patient when doctor said stay for two more days the we have to update discharge date. This is used for that purpose.

```
variables for update:
pat_ID=None
pat_name=None
pat_dob=None
pat_address=None
pat_sex=None
pat_BG=None
pat_email=None
pat_contact=None
pat_contactalt=None
pat_CT=None

def up1():
  global u1, u2, u3, u4, u5, u6, u7, u8, u9, u10, ue1, conn
  conn.cursor()
  u1 = pat_ID.get()
  u2 = pat_name.get()
  u3 = pat_sex.get()
  u4 = pat_dob.get()
  u5 = pat_BG.get()
  u6 = pat_contact.get()
  u7 = pat_contactalt.get()
  u8 = pat_email.get()
  u9 = pat_CT.get()
  u10 = pat_address.get()
  conn = sqlite3.connect("MSP.db")
  p = list(conn.execute("Select * from PATIENT where PATIENT_ID=?", (u1,)))
  if len(p) != 0:
     conn.execute('UPDATE PATIENT SET
NAME=?,SEX=?,DOB=?,BLOOD_GROUP=?,ADDRESS=?,CONSULT_TEAM=?,EMAIL=? where
PATIENT_ID=?', ( u2, u3, u4, u5, u10, u9, u8,u1,))
     conn.execute('UPDATE CONTACT_NO set CONTACTNO=?,ALT_CONTACT=? WHERE
PATIENT_ID=?', ( u6, u7,u1,))
     tkinter.messagebox.showinfo("MEDANTA DATABSE SYSTEM", "DETAILS UPDATED INTO
DATABASE")
     conn.commit()

  else:
     tkinter.messagebox.showinfo("MEDICAL DATABSE SYSTEM", "PATIENT IS NOT REGISTERED")
labelu=None
bu1=None
```

- This is used to update the details of patient personal details when patient wants a update in their details.

EMPLOYEE REGISTRATION:

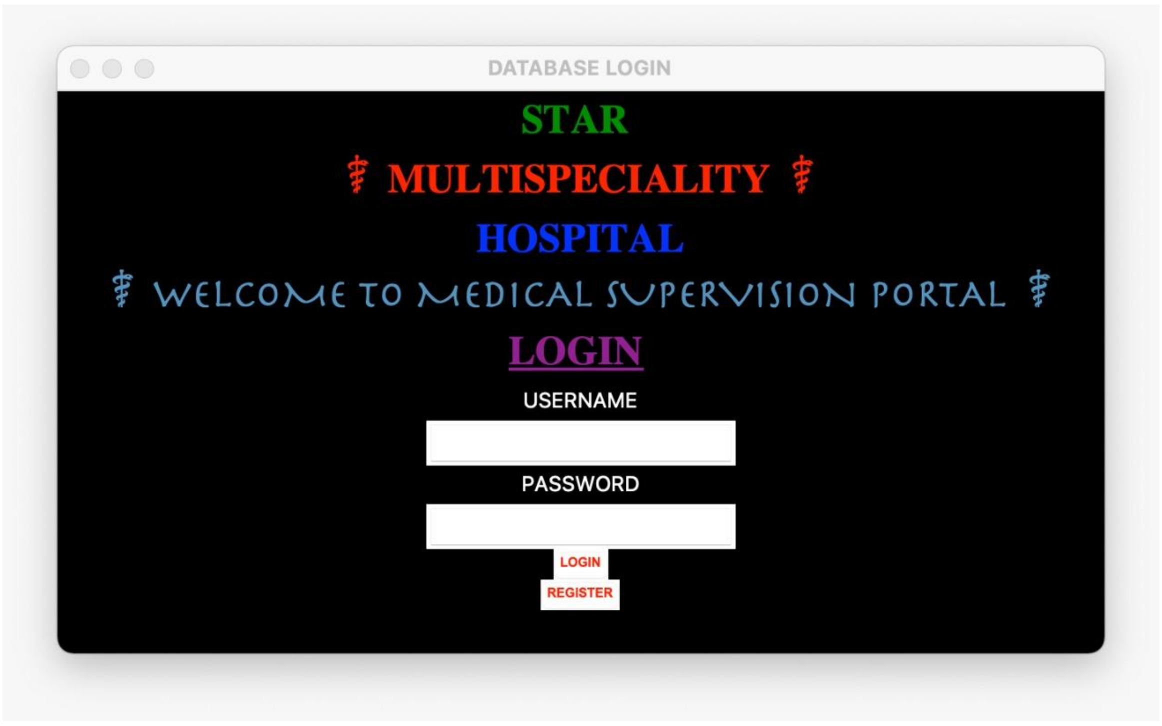
```
def inp():
  global e1,e2,e3,e4,e5,e6,e7,e8,e9,var
  e1=t1.get()
  e2=t2.get()
  e3=str(var.get())
  e4=t3.get()
  e5=lb.get(tkinter.ACTIVE)
  e6=t4.get()
  e7=t5.get()
  e8=t6.get()
  e9=t7.get()
  conn = sqlite3.connect("MSP.db")
  conn.execute("INSERT INTO employee VALUES(?,?,?,?,?,?,?,?,?)",(e1,e2,e3,e4,e5,e6,e7,e8,e9,))
  conn.commit()
  tkinter.messagebox.showinfo("MEDICAL DATABASE SYSTEM", "EMPLOYEE DATA ADDED")

def delling():
  global d1,de
  de=str(d1.get())
  conn = sqlite3.connect("MSP.db")
  p = list(conn.execute("select * from employee where EMP_ID=?", (de,)))
  if (len(p) != 0):
    conn.execute("DELETE from employee where EMP_ID=?", (de,))
    dme = tkinter.Label(rootDE, text="EMPLOYEE DELETED FROM DATABASE", fg="green")
    dme.place(x=20, y=100)
    conn.commit()
  else:
    error = tkinter.Label(rootDE, text="EMPLOYEE DOESN'T EXIST", fg="Red")
    error.place(x=20, y=100)
```

- In this the employees like doctor ,nurse. those who joins newly in the hospital this is used to store the data of the employee in the date and when the employee leaves the date then we can delete the data which is stored in the database.

CHAPTER 6

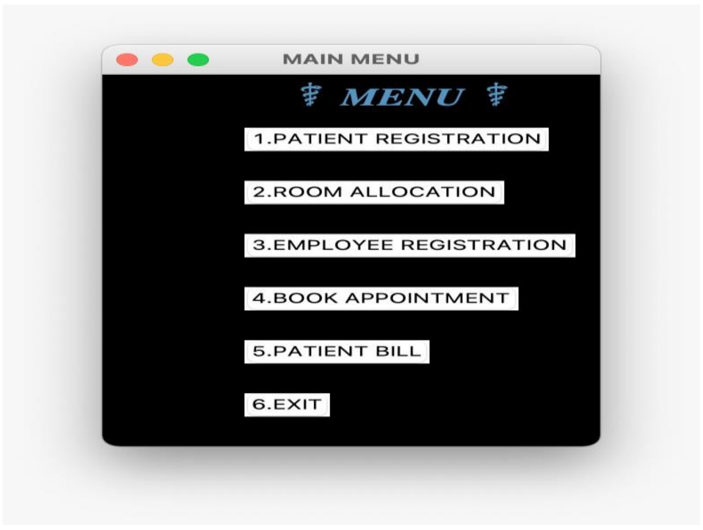
OUTPUT

Login page:

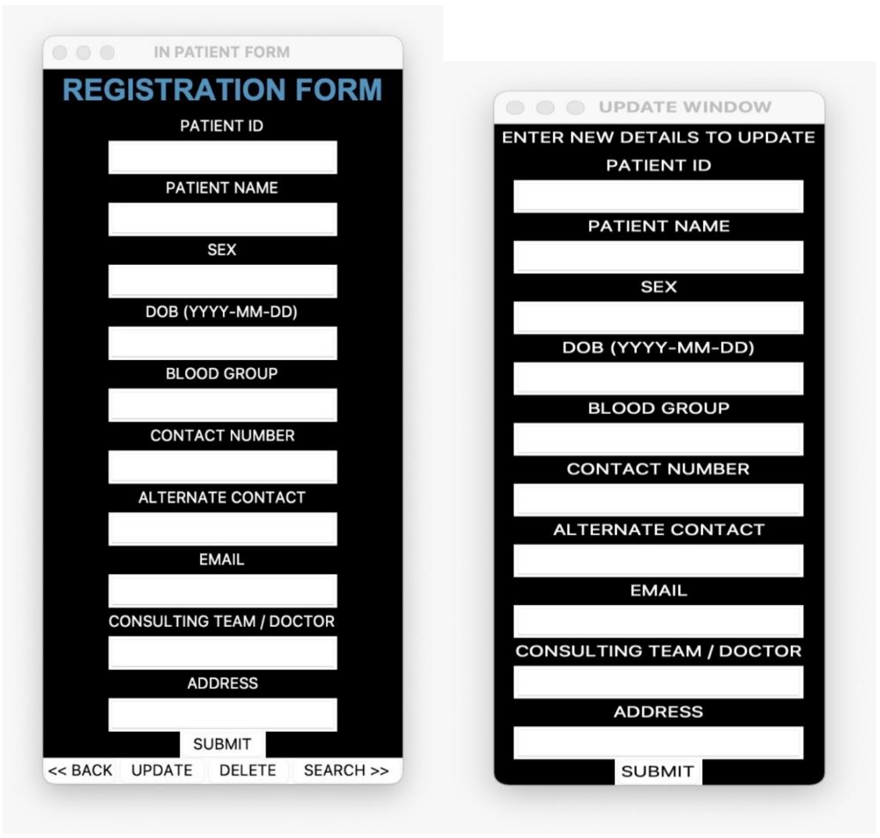


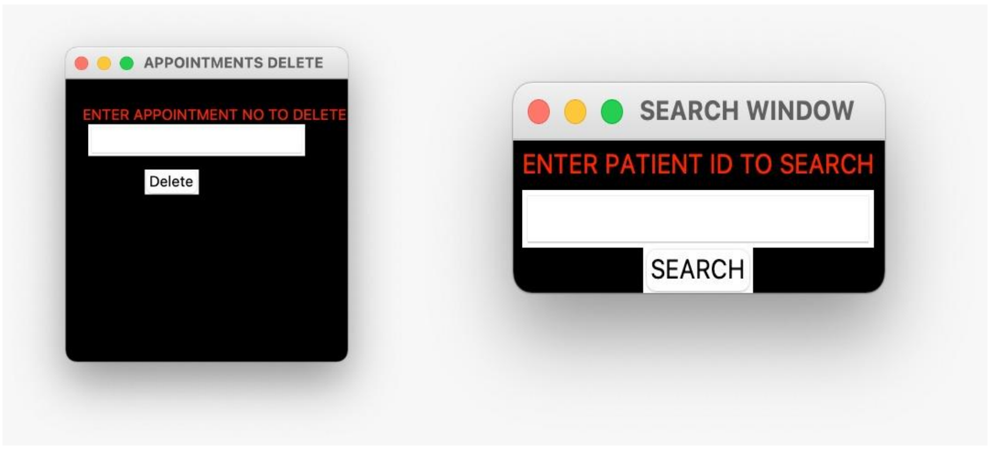
- This is the page of login where the admin login to MSP.

MENU: It will show all the option of medical supervision portal.

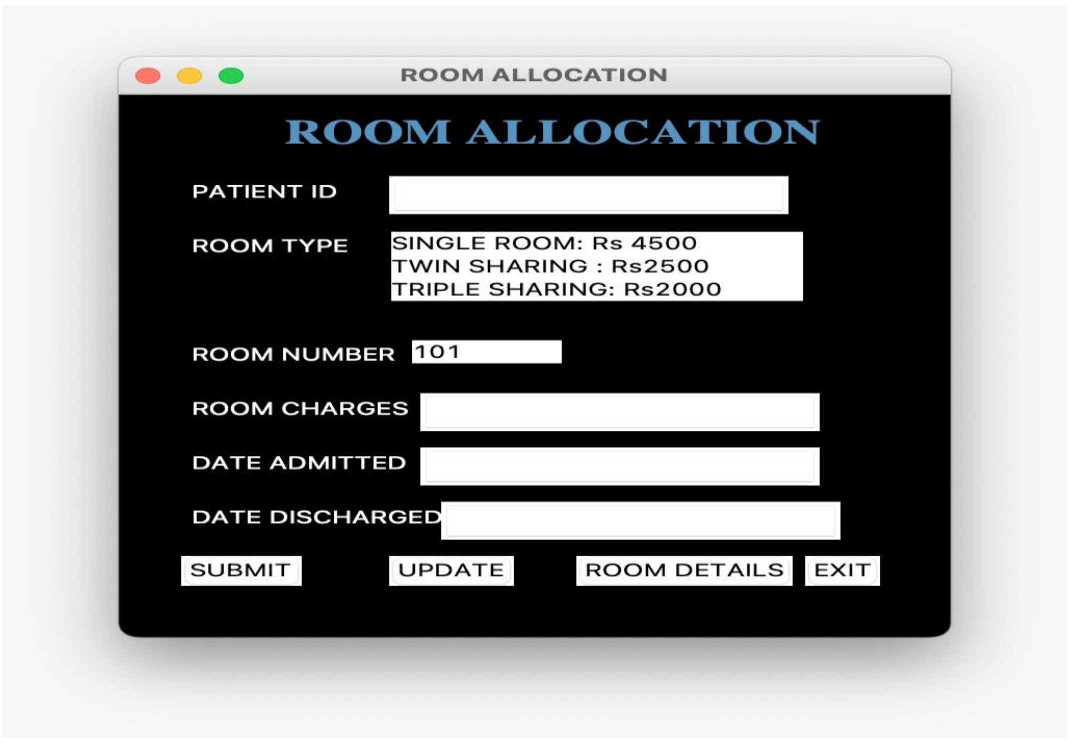


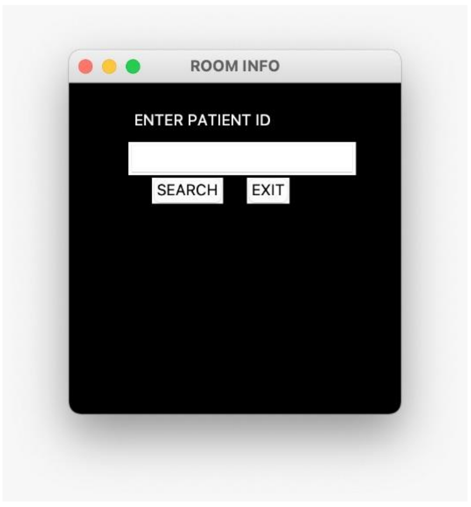
REGISTRATION PAGE: It will show all fields to fill the details of patient ,and update,search,delete.





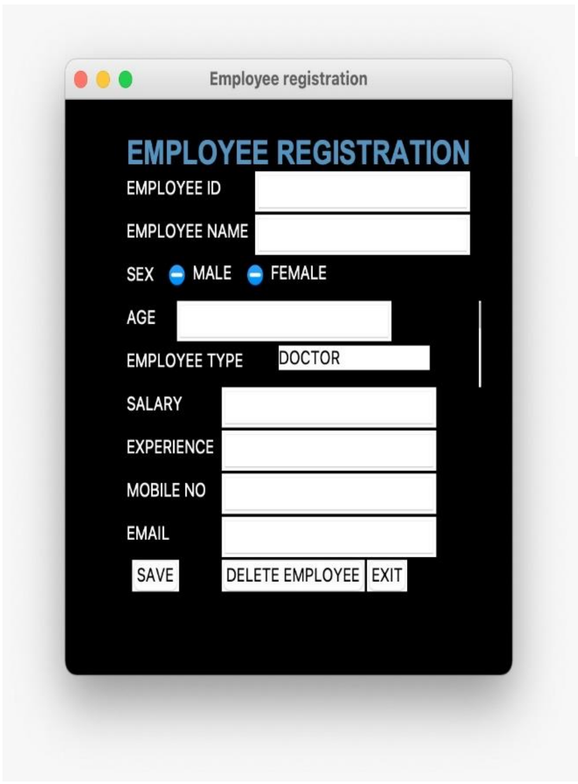
ROOM ALLOCATION: Here we can given the details to allot a room for patient.

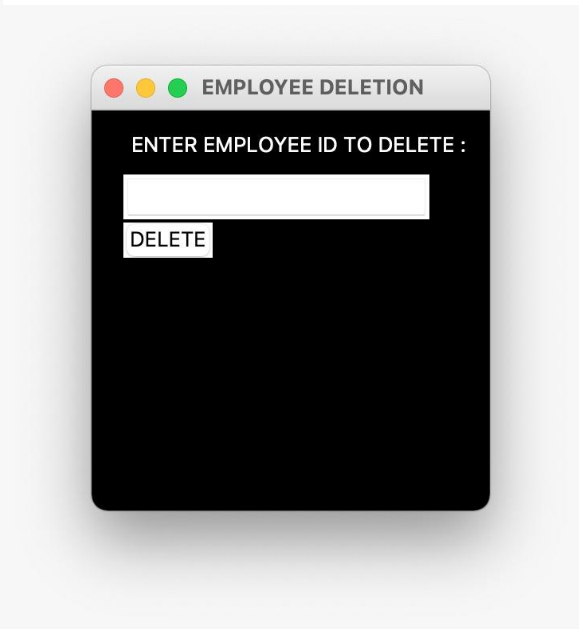




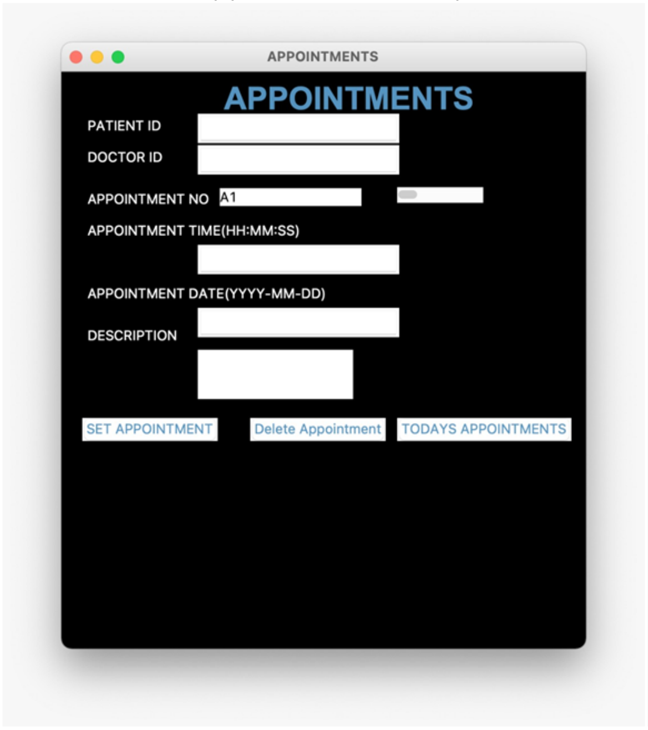
-> this is used to search the room details using patient id.

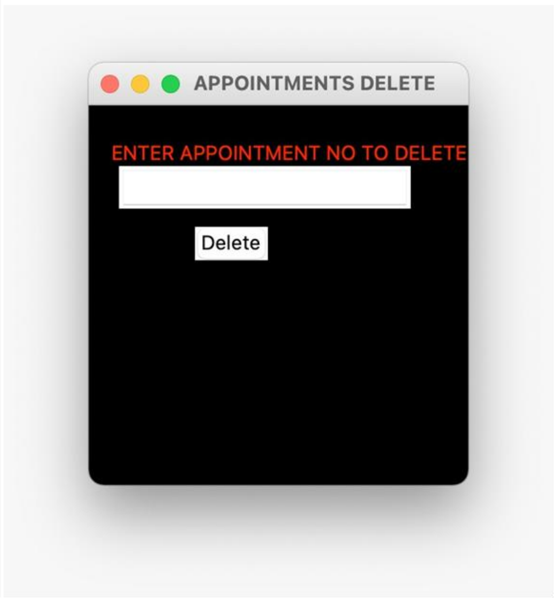
<u>EMPLOYEE REGISTRATION:</u> Here we can give the details of the new employee of hospital and we can delete the employee.

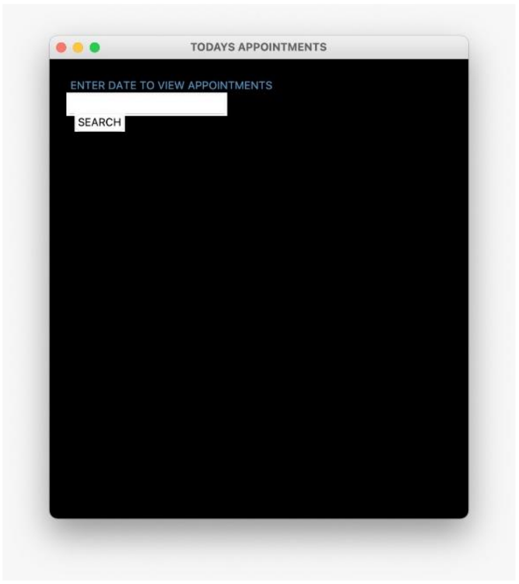




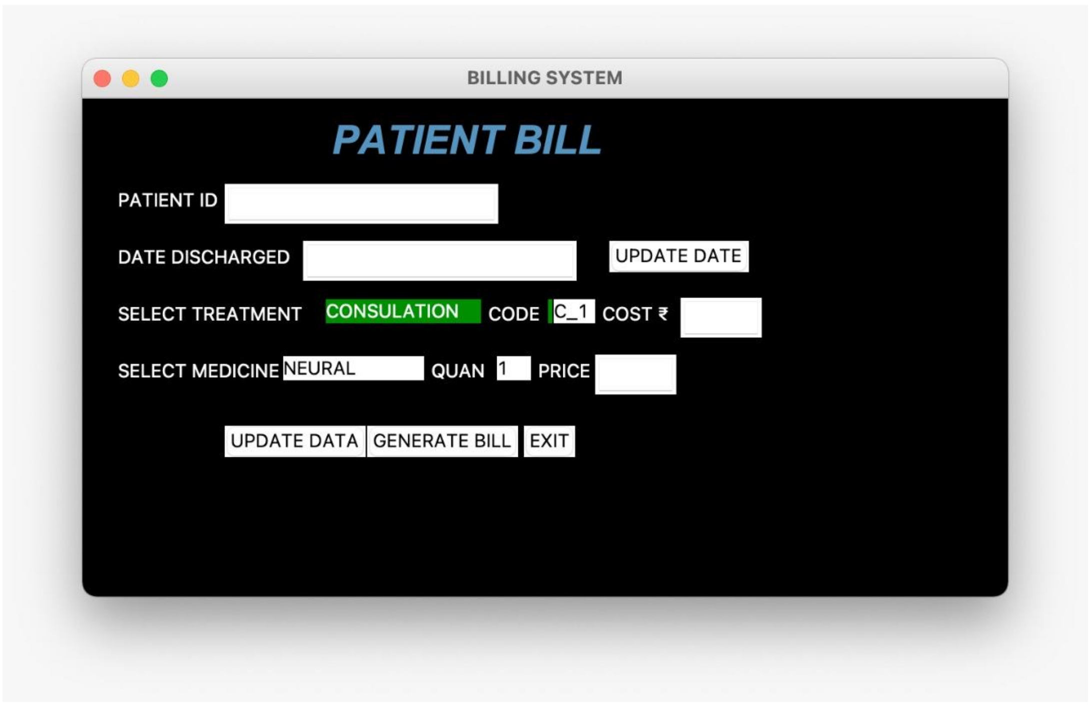
<u>APPONTMENTS</u>: Here is the gui where we can book a appointment to meet the doctor, and we can delete the appointment when patient wants to delete , we can search everyday appointments







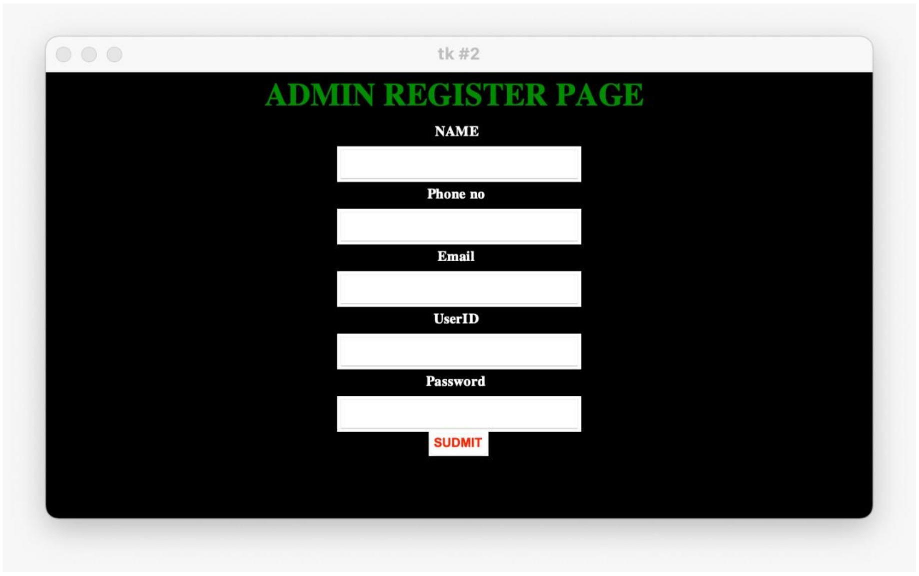
BILLING SYSTEM: Here we can calculate whole bill for required patient using patient id.



SELECT TREATMENT CONSULATION CODE C_1

SELECT MEDICINE NEURAL QUAN 1

ADMIN PAGE: Here we can register the new admin of hospital by providing details



CHAPTER 7

CONCLUSION

As per my medical supervision portal, we can make the conclusion that the medical supervision is user friendly and it has many options to maintain the patients and doctors database and retrive the data very easy and every one will easily understand. It remotes numerous daily operations and enables smooth interactions of the users and admins. Developing the hospital supervision portal software is to create the distinct, efficient and fast delivering healthcare model. Implementation of hospital supervision portal helps to store all the kinds of records, provide coordination and user communication, improve day-to-day operations, manage financial and human resources, and market hospital services. This MSP decision covers the needs of the patients, staff and hospital authorities and simplifies their interactions. It has become the usual approach to manage the hospital. Many clinics have already experienced its advantages and continue developing new hospital management system project modules. we need to improve management in every hospital or plan to innovate healthcare with this medical supervision portal. The MSP software is a solution for any medical institution, and w have to improve this supervision portal day-by-day and implement advance learning to become more and more user-friendly

CHAPTER 8

REFERENCES

- W3schools.com
- Javatutorials.com
- Tkdocs.com
- Geeks for Geeks