

C O N T E N T S

S.N	Topics
1	Introduction
2	Objective & Scope of the Project
3	Theoretical Background
4	Problem Definition & Analysis
5	System Implementation
	5.1 The Hardware used:
	5.2 The Softwares used:
6	System Design & Development
	6.1 Database Design
	6.2 Table Design:
	6.3 I/O Forms Design & Event Coding
7	User Manual
	7.1 How to install
	7.2 Working with Software:
8	References

1. Introduction

This software project is developed to automate the functionalities of a Medical Shop. The purpose of the software project is to develop the Management Information System (MIS) to automate the record keeping of medicines, Manufacturing Co., Transactions(Bill/Invoice) with a view to enhance the decision making of the functionaries.

A MIS mainly consists of a computerized database, a collection of inter-related tables for a particular subject or purpose, capable to produce different reports relevant to the user. An application program is tied with the database for easy access and interface to the database. Using Application program or front-end, we can store, retrieve and manage all information in proper way.

This software, being simple in design and working, does not require much of training to users, and can be used as a powerful tool for automating a **MEDICAL SHOP SYSTEM.**

During coding and design of the software Project, Python IDLE, a powerful front-end tool is used for coding simplicity. As a back-end a powerful, open source RDBMS, My SQL is used as per requirement of the CBSE curriculum of In Computer Science Course.

2. Objective & Scope of the Project

The objective of the software project is to develop a computerized MIS to automate the functions of a **MEDICAL SHOP SYSTEM**. This software project is also aimed to enhance the current record keeping system, which will help managers to retrieve the up-to-date information at right time in right shape. The proposed software system is expected to do the following functionality-

- ✓ The proposed system should maintain all the records and transactions, and should generate the required reports and information when required.
- ✓ To provide user-friendly interface to interact with a centralized database based on client-server architecture.
- ✓ To identify the critical operation procedure and possibilities of simplification using modern IT tools and practices.

In its current scope, the software enables user to retrieve and update the information from centralized database designed with MySQL. This software does not require much training time of the users due to limited functionality and simplicity.

During the development of **MEDICAL SHOP SYSTEM** project, Python IDLE, a powerful, open source event-driven form-based development environment is used for modular design and future expandability of the system.

Despite of the best effort of the developer, the following limitations and functional boundaries are visible, which limits the scope of this application software.

1. This software can store records and produce invoices in pre-designed format in soft copy. There is no facility yet to produce customized reports. Only specified invoices are covered.

2. There is no provision to produce invoices of previously purchased medicines; however it can be developed easily with the help of adding modules.
3. Some application area like accounting of sold medicines etc. are not implemented in the project. It facilitates manager to record, delete, update and view record.

So far as future scope of the project is concerned, firstly it is open to any modular expansion i.e. other modules or functions can be designed and embedded to handle the user need in future. Any part of the software and reports can be modified independently without much effort.

3. Theoretical Background

3.1 What is Database?

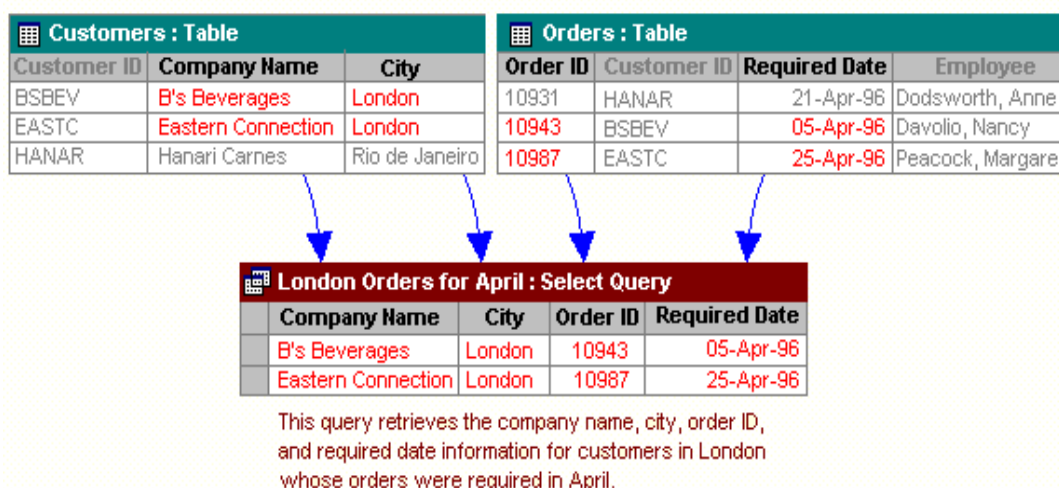
Introduction and Concepts:

A database is a collection of information related to a particular subject or purpose, such as tracking customer orders or maintaining a music collection. Using any RDBMS application software like MS SQL Server, MySQL, Oracle, Sybase etc., you can manage all your information from a single database file. Within the file, divide your data into separate storage containers called tables. You may insert and retrieve the data using queries.

A table is a collection of data about a specific topic, such as products or suppliers. Using a separate table for each topic means you can store that data only once, which makes your database more efficient and reduces data-entry errors. Table organises data into columns (called fields) and rows (called records).

A Primary key is one or more fields whose value or values uniquely identify each record in a table. In a relationship, a primary key is used to refer to specific record in one table from another table. A primary key is called foreign key when it is referred to from another table.

To find and retrieve just the data that meets conditions you specify, including data from multiple tables, create a query. A query can also update or delete multiple records at the same time, and perform built-in or custom calculations on your data.



Role of RDBMS Application Program:

A computer database works as a electronic filing system, which has a large number of ways of cross-referencing, and this allows the user many different ways in which to re-organize and retrieve data. A database can handle business inventory, accounting and filing and use the information in its files to prepare summaries, estimates and other reports. The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available DBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase. A database management system, therefore, is a combination of hardware and software that can be used to set up and monitor a database, and can manage the updating and retrieval of database that has been stored in it. Most of the database management systems have the following capabilities:

- ◆ Creating of a table, addition, deletion, modification of records.
- ◆ Retrieving data collectively or selectively.
- ◆ The data stored can be sorted or indexed at the user's discretion and direction.
- ◆ Various reports can be produced from the system. These may be either standardized report or that may be specifically generated according to specific user definition.
- ◆ Mathematical functions can be performed and the data stored in the database can be manipulated with these functions to perform the desired calculations.
- ◆ To maintain data integrity and database use.

The DBMS interprets and processes users' requests to retrieve information from a database. In most cases, a query request will have to penetrate several layers of software in the DBMS and operating system before the physical database can be accessed. The DBMS responds to a query by invoking the appropriate subprograms, each of which performs its special function to interpret the query, or to locate the desired data in the database and present it in the desired order.



3.2 What is My SQL ?

The management of data in a database system is done by means of a general-purpose software package called a Database Management System (DBMS). Some commercially available RDBMS are MS SQL Server, MS ACCESS, INGRES, ORACLE, and Sybase.

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation. MySQL is named after co-founder Monty Widenius's daughter, My. The name of the MySQL Dolphin (or logo) is "Sakila".

- **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL is based on SQL.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. This adds speed and flexibility. The SQL part of "MySQL" stands for "Structured Query Language." SQL is the most common standardized language used to access databases and is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, "SQL-92" refers to the standard released in 1992, "SQL:1999" refers to the standard released in 1999, and "SQL:2003" refers to the current version of the standard.

- **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License).

- **The MySQL Database Server is very fast, reliable, and easy to use.**

If that is what you are looking for, you should give it a try. MySQL Server also has a practical set of features developed in close cooperation with our users. You can find a performance comparison of MySQL Server with other database managers on our benchmark page. MySQL Server was

originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems.**

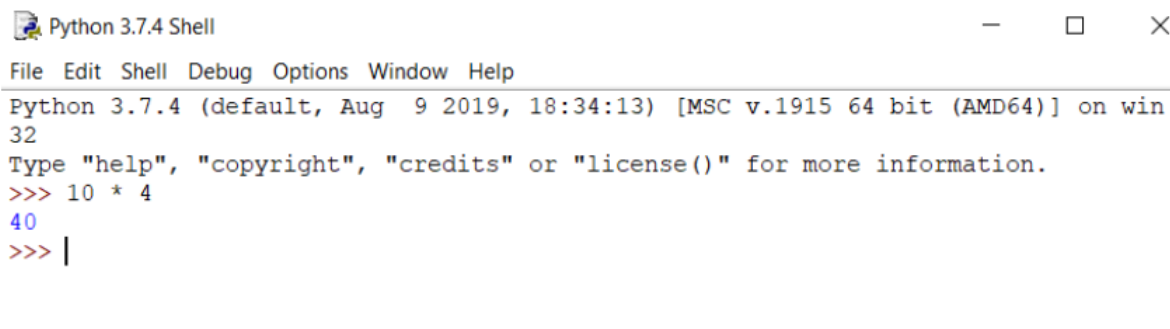
The MySQL Database Software is a client/server system that consists of a multi-threaded SQL server that supports different backend, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

The Main Features of MySQL

- Written in C and C++.
- Works on many different platforms.
- Uses multi-layered server design with independent modules.
- Provides transactional and non-transactional storage engines.
- Designed to make it relatively easy to add other storage engines. This is useful if you want to provide an SQL interface for an in-house database.
- Uses a very fast thread-based memory allocation system.
- Executes very fast joins using an optimized nested-loop join.
- Implements SQL functions using a highly optimized class library that should be as fast as possible. Usually there is no memory allocation at all after query initialization.
- Provides the server as a separate program for use in a client/server networked environment, and as a library that can be embedded (linked) into standalone applications. Such applications can be used in isolation or in environments where no network is available.
- Password security by encryption of all password traffic when you connect to a server.
- Support for large databases. We use MySQL Server with databases that contain 50 million records. We also know of users who use MySQL Server with 200,000 tables and about 5,000,000,000 rows.
- MySQL client programs can be written in many languages. A client library written in C is available for clients written in C or C++, or for any language that provides C bindings.
- APIs for C, C++, Eiffel, Java, Perl, PHP, Python, Ruby, and Tcl are available, enabling MySQL clients to be written in many languages.

- The Connector/ODBC (MyODBC) interface provides MySQL support for client programs that use ODBC (Open Database Connectivity) connections.
- The Connector/J interface provides MySQL support for Java client programs that use JDBC connections. Clients can be run on Windows or Unix. Connector/J source is available.

3.3 Python IDLE



Python IDLE

Introduction

IDLE stands for Integrated Development and Learning Environment. The story behind the name IDLE is similar to Python. Guido Van Rossum named Python after the British comedy group Monty Python while the name IDLE was chosen to pay tribute to Eric Idle, who was one of the Monty Python's founding members. IDLE comes bundled with the default implementation of the Python language since the 01.5.2b1 release. It is packaged as an optional part of the Python packaging with many Linux, Windows, and Mac distributions.

IDLE, as shown above, is a very simple and sophisticated IDE developed primarily for beginners, and because of its simplicity, it is highly considered and recommended for educational purposes. It offers a variety of features that you will look in detail along with examples later in this tutorial.

Some of the key features it offers are:

- Python shell with syntax highlighting,
- Multi-window text editor,
- Code auto completion,
- Intelligent indenting,
- Program animation and stepping which allows one line of code to run at a time helpful for debugging,
- Persistent breakpoints,
- Finally, Call stack visibility.

How to Install IDLE?

- One way to install IDLE is to install Python from the official Python website, as shown below.

You can download the latest Python versions from this website for different operating systems like windows, Linux, and mac. It also provides you the docker images for Python, which you can directly use if you have a docker installed on your system.



Let's install Python for the Windows operating system.

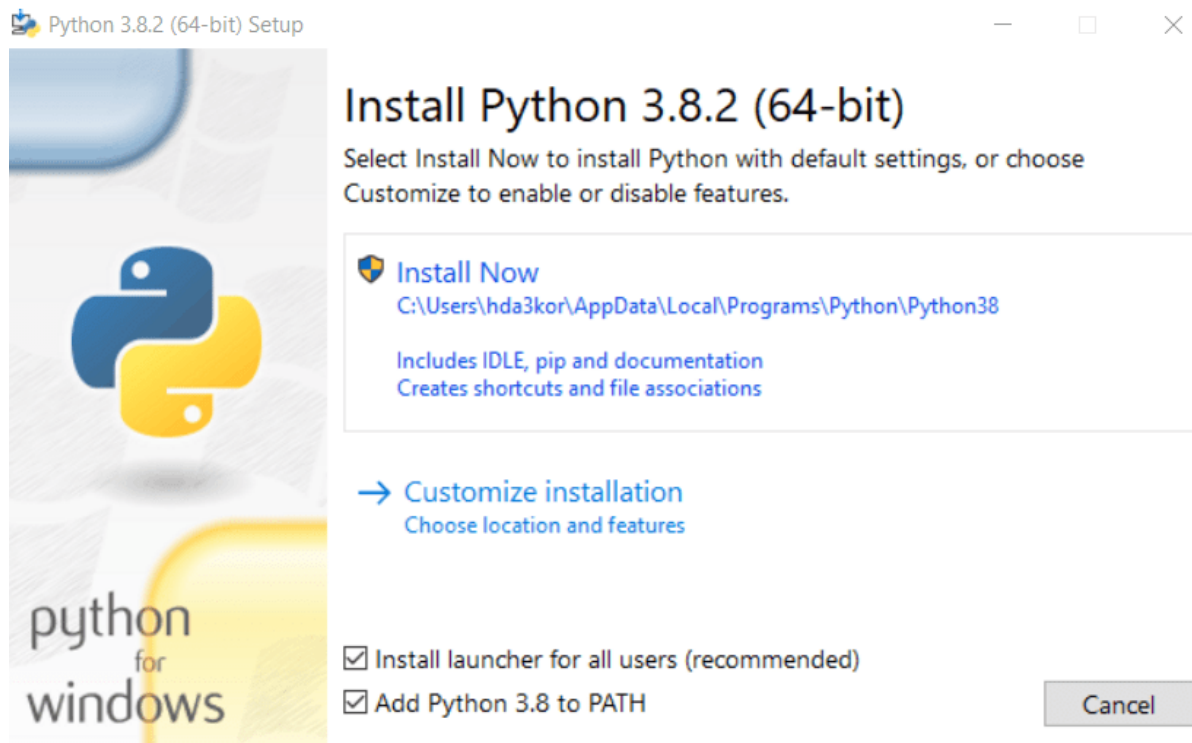
Installing Python on Windows OS

- Since Python2 development has been stopped, let's install the latest Python version, i.e., 3.8.2.

Python Releases for Windows

- [Latest Python 3 Release - Python 3.8.2](#)
- [Latest Python 2 Release - Python 2.7.18](#)

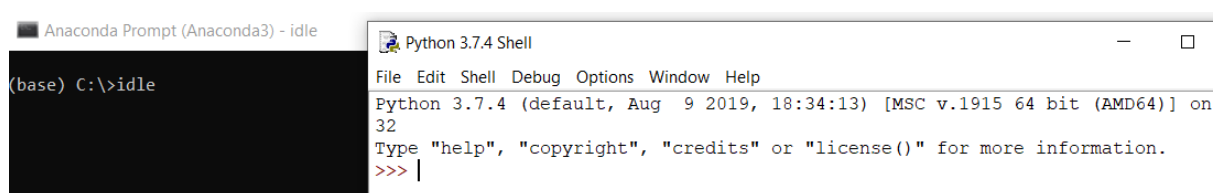
- Install the Windows x86 executable installer file depending on whether you have a 32-bit or 64-bit OS and run that file. Once you run the file, a window will open, as shown below. Make sure you select the install launcher for all users and add Python3.8 to Path along with the recommended installation.



- Once the installation is complete, go to the start menu and type idle and click on the same. You should now be seeing the IDLE software on your system.

IDLE with Anaconda

IDLE can also be used through Anaconda. If you already have Anaconda installed on your system, just open the anaconda prompt from the start menu and type idle in the anaconda terminal, as shown below.



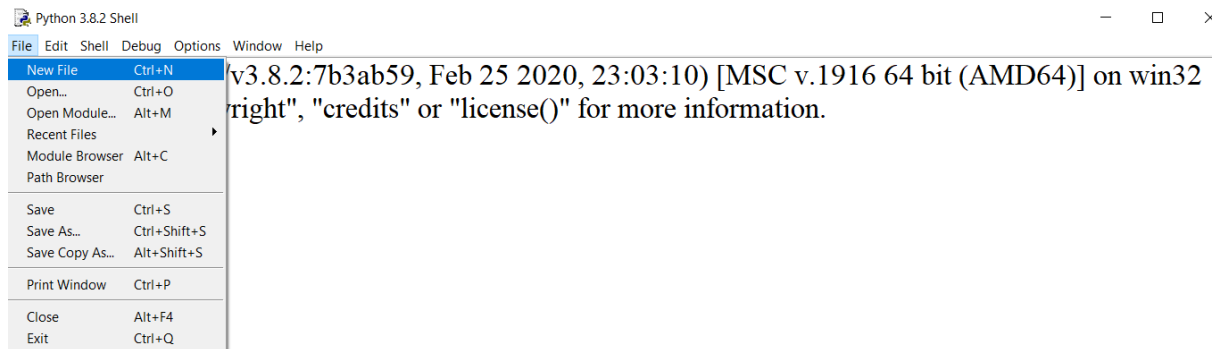
Since IDLE comes packaged with Anaconda, you will need to download Anaconda from <https://www.anaconda.com/>.

To learn how to install Anaconda, check out the documentation.

Well, now that you have the IDLE software installed on your respective systems, let's understand its features in detail.

IDLE Software Features

- Remember that it is not advisable to write multiple lines of code that have functions/classes in the IDLE shell. In such cases, you can go to the File option of IDLE and click on New File.

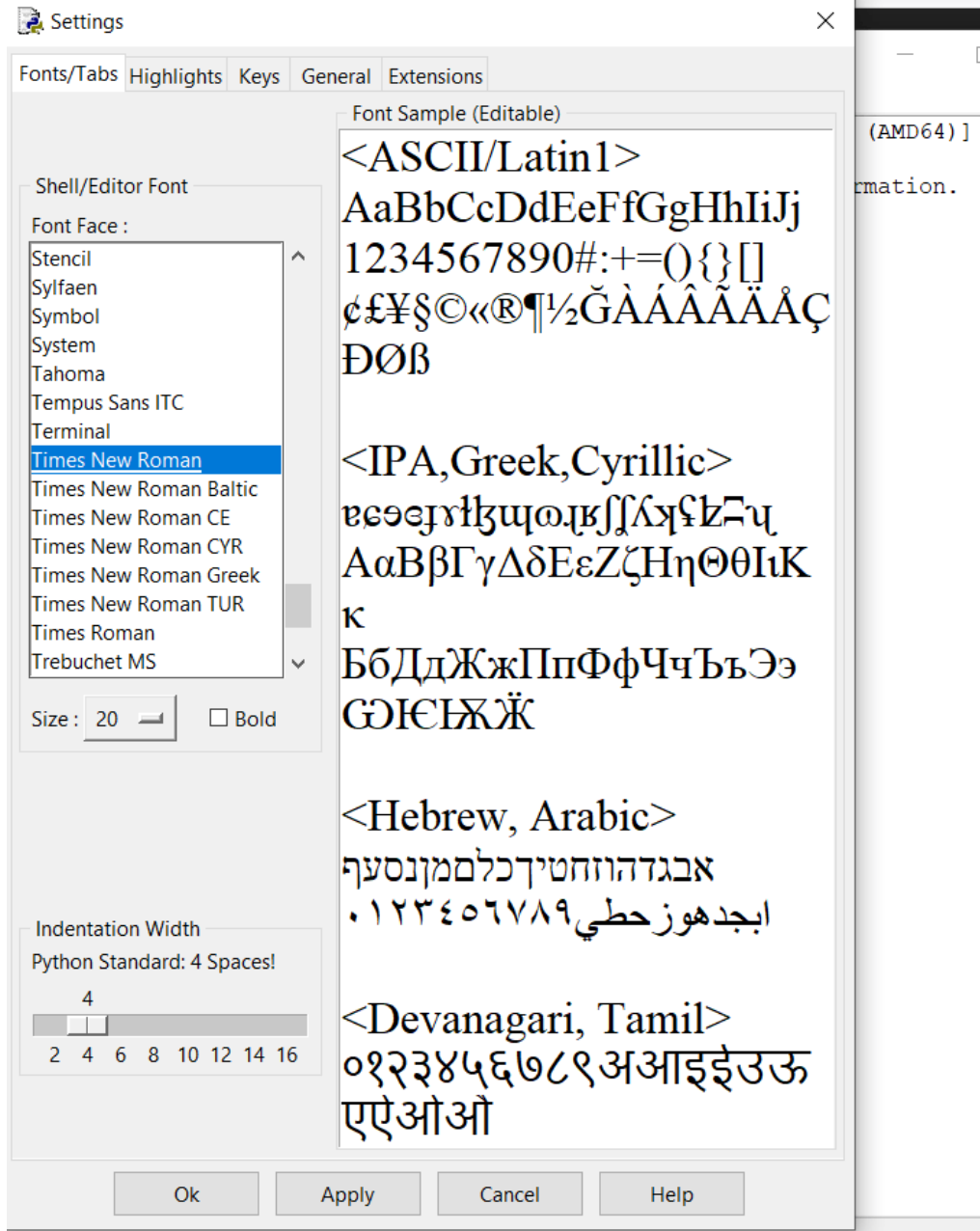


- IDLE can be customized using the options present in the Format, Edit, and Options menu.
 - Modifying the font-size: Go to the Options menu and click configure idle.**

From this, you can choose any font face, font size, make the font bold and even calibrate the indentation width you like working with, as shown below.

IDLE provides you this feature so that each and every person can work with a font size he/she is comfortable working with. A lot of times, your monitor screen size is small, and as a result, you need an IDE that can allow you to increase the font size. That's where IDLE can come into play and give a sense of relief to your eyes.

Not just the font size and style, you can even adjust the indentation, as mentioned before, which is set to 4 spaces by default as per the PEP-8 coding guidelines.



- **Writing your first code in IDLE**

You will write a small program in which you will import the NumPy library, define two arrays, and sum them up. Finally, you will run the code and see the output in the IDLE shell.

To be able to import the NumPy library, make sure it is installed on your system. If not, just go to the command prompt and type `pip install NumPy`.

```
import numpy as np
```

```
arr1 = np.array([[19,12], [20,100]])
```

```
print(np.sum(arr1))
```

Call Stack Visibility

While writing the above code, you would notice a dialog box on the screen for both the print function and np.sum function. IDLE will give you the details as to what parameters can be passed to the np.sum function. This is one of the more important and less obvious features of IDLE, which shows the call stack in a verbose manner.

```
import numpy as np
```

```
arr1 = np.array([[19,12], [20,100]])
```

```
print(np.sum(arr1))
```

```
(a, axis=None, dtype=None, out=None, keepdims=<no value>, initial=<no value>,  
where=<no value>)
```

To run the above code, go to the run menu and click on the run module option or directly press F5. It should open the IDLE shell along with the output of your program.

```
import numpy as np
```

```
arr1 = np.array([[19,12], [20,100]])
```

```
print(np.sum(arr1))
```

Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] Type "help", "copyright", "credits" or "license()" for more information.

>>>

===== RESTART: C:\Users\hda3kor\Documents\Python_IDLE\first_idle.py ==
151

Edit Menu in IDLE

The edit menu in IDLE has a lot of general features like:

- undo,
- redo,
- find,
- find in files,
- replace,
- go to the line,
- show surrounding parenthesis

But one which makes it special is the show completions feature with which you can auto complete your code. Imagine you are working on a large and complicated project which involves a lot of coding. You might end up spending a lot of time typing the code.

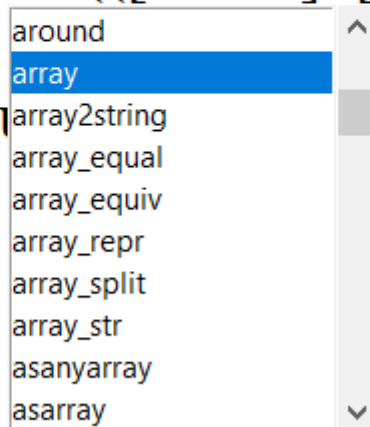
The show complete feature in IDLE helps you in saving typing time by trying to finish the code for you. The Python IDLE is capable of auto completing only functions and classes. The auto completion feature can be used by pressing either the tab key or ctrl+space, as shown below:

File Edit Format Run Options Window Help

import numpy as np

arr1 = np.array([19,12], [20,100])

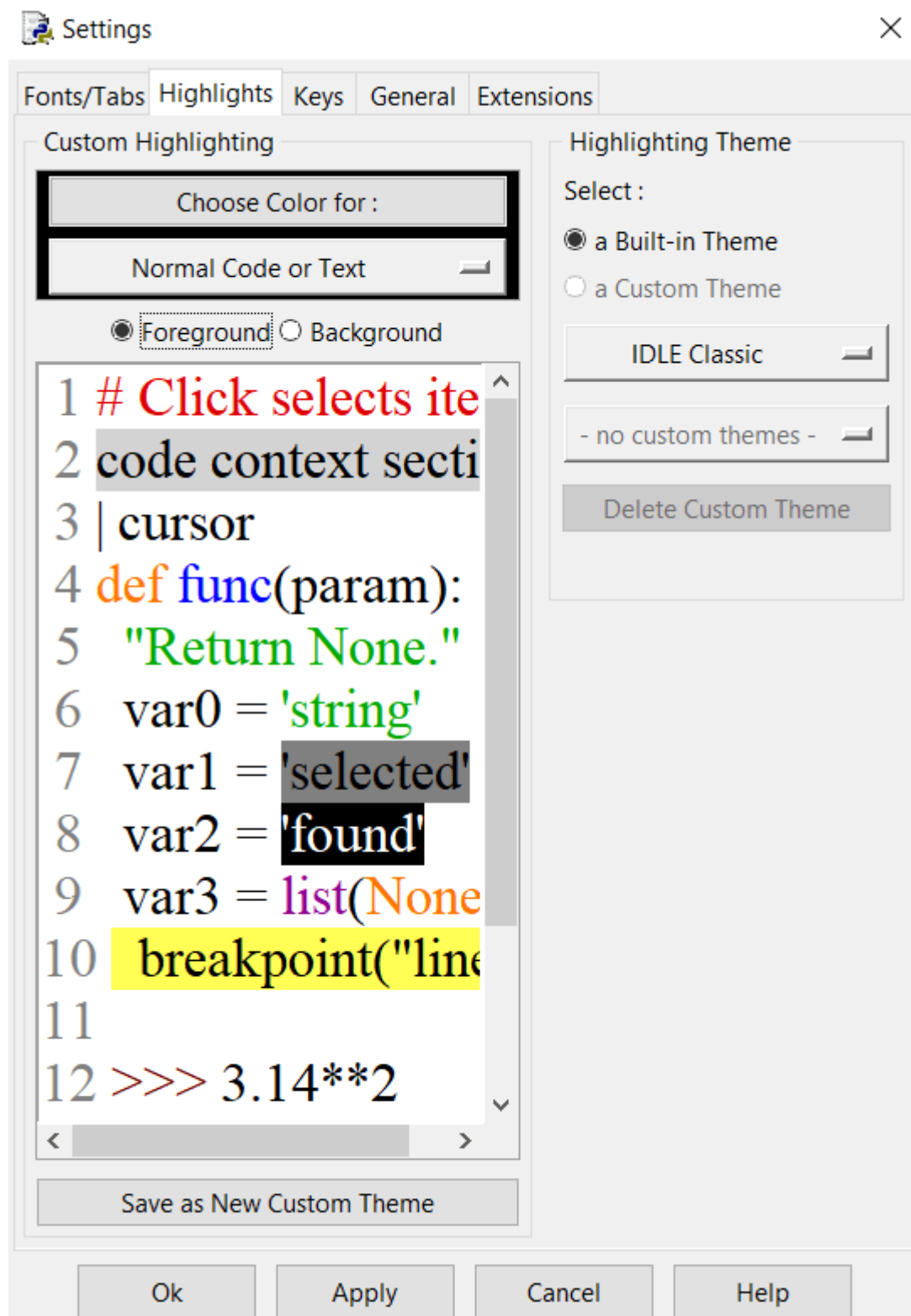
print(np.s



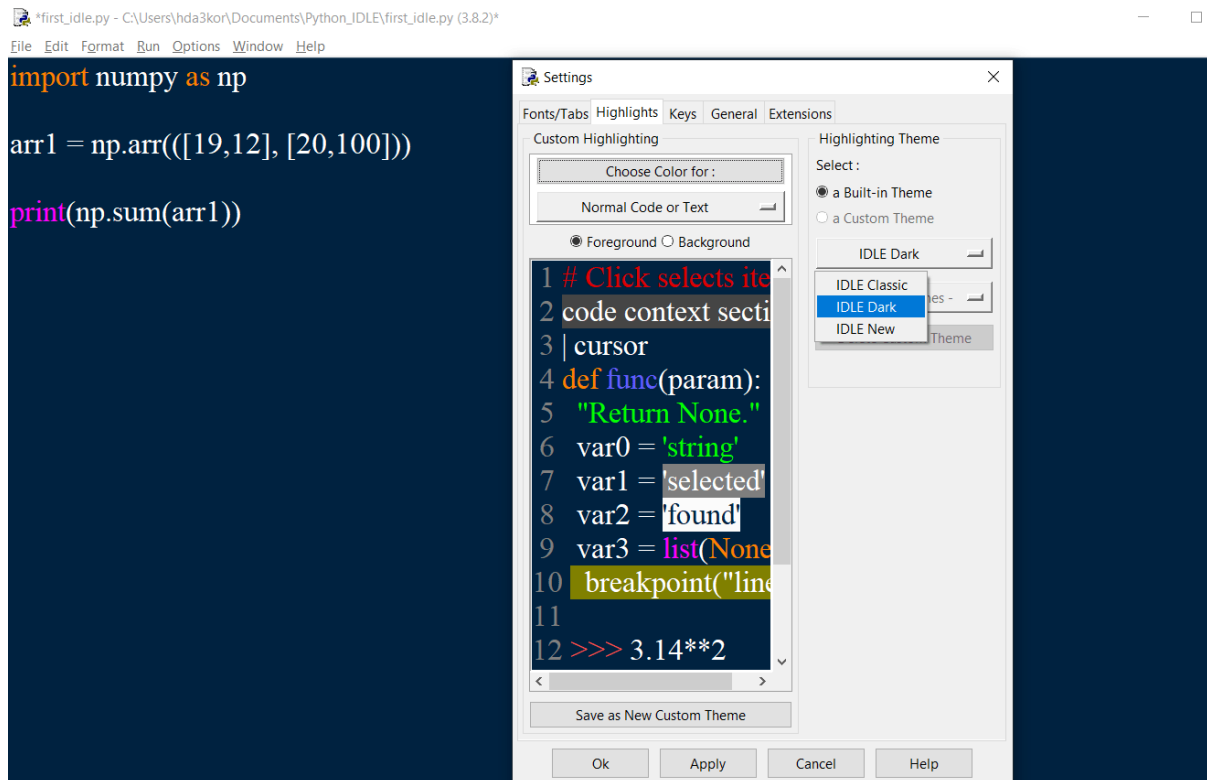
You already saw the Fonts/Tabs feature, now let's look at some more features in `configure_idle` under the options menu.

- Highlights: Here, you can change the color of the code syntax according to your preference and not just that. You can also configure idle to work in dark mode or theme, which is a great feature to have, especially when you want to code for longer duration and give some relief to your eyes.

By default, idle works in classic mode and can be tuned to work in a dark mode.

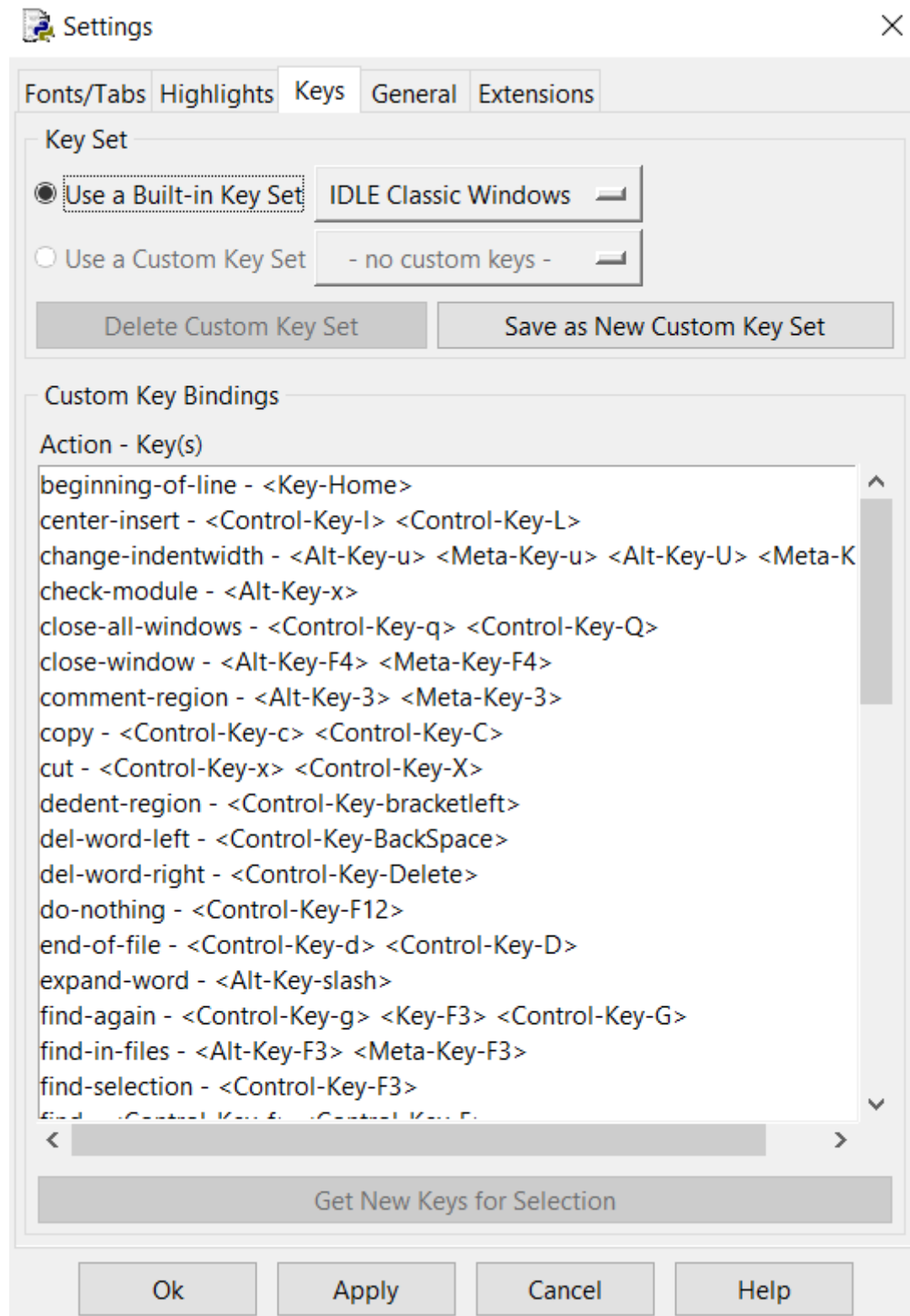


Idle in Dark Theme!



- **Keys:** This feature lets you map different key presses to actions, also known as keyboard shortcuts. These are vital components to increase your overall productivity while using an IDE. You can either come up with your own keyboard shortcuts or use the default ones.

As can be seen from the below figure, the keys are represented in the Action - Key format, which shows you the action that is performed when a specific key or keys are used. Though rarely used by developers, the keys feature in IDLE makes your life easier, especially when you want to write a lot of code in a quick manner.

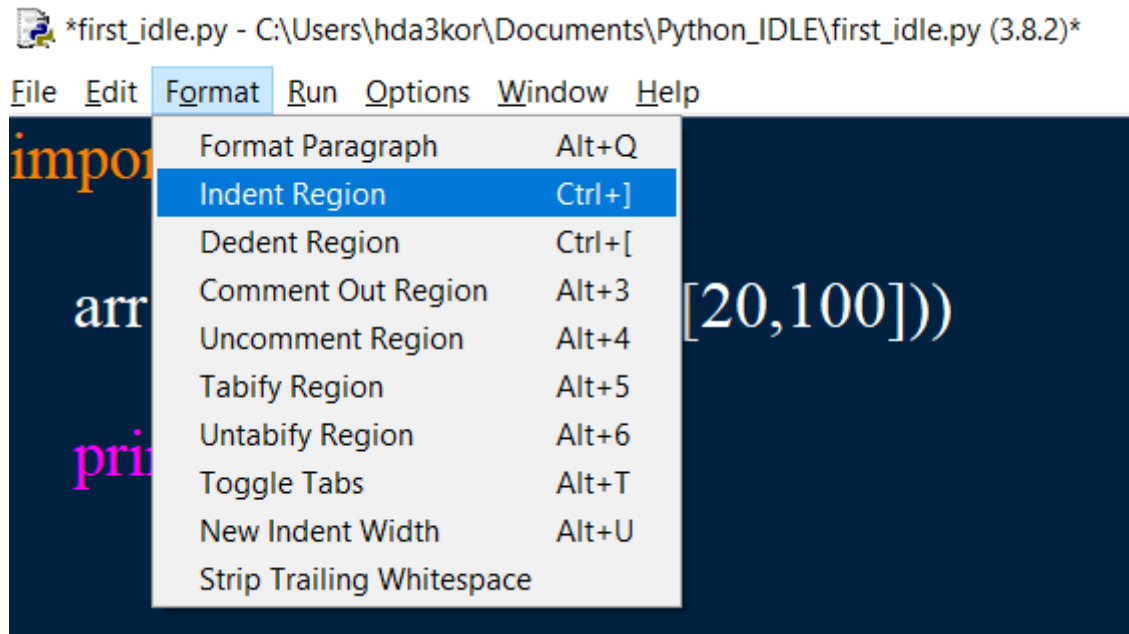


Format Menu

Like other menu's, Format also has few useful features like indent region, dedent region, comment out a region, etc.

Let's look at the indent region. The indentation in Python is used to segregate a block of code, let's say you have written a code but missed giving indentations to it, the indent region feature could come to your rescue.

The indent region tab added 4 spaces to the selected lines of code, as shown below:



Similarly, the dedent region will help you in removing any unnecessary spaces from your code.

Writing a Palindrome Code in IDLE

A palindrome is a word, phrase, or sequence of numbers that read the same backward as forward.

For example, sequence 121, 151, etc.

Let's quickly write the Palindrome code in IDLE and check the output.

```
forward = input("Enter the sequence:") #it will read a string from standard input
backward = forward[::-1] #store the reverse copy of forward in backward
if forward == backward: #check if forward is equal to backward
    print("The sequence is a palindrome")
else:
    print("Not a palindrome sequence")
```

Let's now save the above code and run it.

```
Python 3.8.2 Shell
File Edit Shell Debug Options Window Help
Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 23:03:10) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:\Users\hda3kor\Documents\Python_IDLE\first_idle.py =====
Enter the sequence:151
The sequence is a palindrome
>>>
===== RESTART: C:\Users\hda3kor\Documents\Python_IDLE\first_idle.py =====
Enter the sequence:123456
Not a palindrome sequence
>>>
===== RESTART: C:\Users\hda3kor\Documents\Python_IDLE\first_idle.py =====
Enter the sequence:DataCamp
Not a palindrome sequence
>>>
===== RESTART: C:\Users\hda3kor\Documents\Python_IDLE\first_idle.py =====
Enter the sequence:CIVIC
The sequence is a palindrome
```

From the above output, you can observe that in all four experiments, the program expected input from the user, and based on the input, the if-else conditions were executed.

4. Problem Definition & Analysis

The hardest part of building a software system is deciding precisely what to build. No other part of the conceptual work is so difficult as establishing the detailed technical requirement. Defining and applying good, complete requirements are hard to work, and success in this endeavor has eluded many of us. Yet, we continue to make progress.

Problem definition describes the *What* of a system, not *How* . The quality of a software product is only as good as the process that creates it. Problem definition is one of the most crucial steps in this creation process. Without defining a problem, developers do not know what to build, customers do not know what to expect, and there is no way to validate that the built system satisfies the requirement.

Problem definition and Analysis is the activity that encompasses learning about the problem to be solved, understanding the needs of customer and users, trying to find out who the user really is, and understanding all the constraints on the solution. It includes all activities related to the following:

- ✓ Identification and documentation of customer's or user's needs.
- ✓ Creation of a document that describes the external behavior and the association constraints that will satisfies those needs.
- ✓ Analysis and validation of the requirements documents to ensure consistency, completeness, and feasibility
- ✓ Evolution of needs.

After the analysis of the functioning of a **Medical Shop System**, the proposed System is expected to do the following: -

- ✓ To provide a user friendly based integrated and centralized environment for computerized **Medical Shop System**.
- ✓ The proposed system should maintain all the records and transactions, and should generate the required reports and information when required.
- ✓ To provide efficient and secured Information storage, flow and retrieval system, ensuring the integrity and validity of records.
- ✓ To provide user-friendly interface to interact with a centralized database based on client-server architecture.
- ✓ To identify the critical operation procedure and possibilities of simplification using modern IT tools and practices.

5. System Implementation

5.1 The Hardware used:

While developing the system, the used hardware are:

PC with Intel processor with (2.3 GHz) processor having 01 GB RAM, SVGA and other required devices.

5.2 The Softwares used:

- Microsoft Windows® 7 as Operating System.
- Python 3.7 as Front-end Development environment.
- MySQL as Back-end Sever with Database for Testing.
- MS-Word 2007 for documentation.
- MS-Excel for Viewing csv Files.

6. System Design & Development

6.1 Database Design:

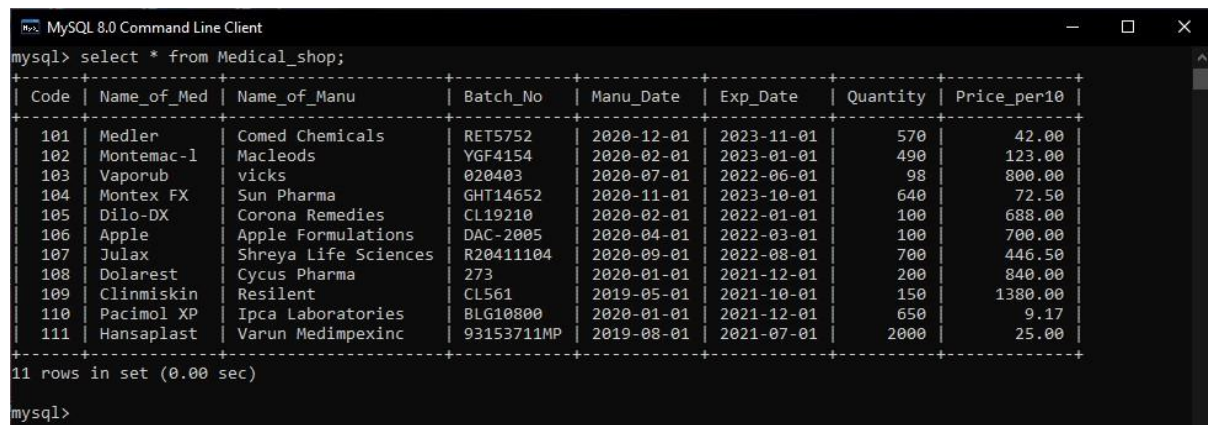
An important aspect of system design is the design of data storage structure. To begin with a logical model of data structure is developed first. A database is a container object which contains tables, queries, reports and data validation policies enforcement rules or constraints etc. A logical data often represented as a records are kept in different tables after reducing anomalies and redundancies. The goodness of data base design lies in the table structure and its relationship.

This software project maintains a database named **CS2020_21** which contains the following tables.

6.2 Table Design:

The database of **Medical Shop System** contains 2 tables. The tables are normalized to minimize the redundancies of data and enforcing the validation rules of the organization. Most of the tables are designed to store master records. The tables and their structure are given below.

Table: Medical_Shop



```
mysql> select * from Medical_shop;
```

Code	Name_of_Med	Name_of_Manu	Batch_No	Manu_Date	Exp_Date	Quantity	Price_per10
101	Medler	Comed Chemicals	RET5752	2020-12-01	2023-11-01	570	42.00
102	Montemac-1	Macleods	YGF4154	2020-02-01	2023-01-01	490	123.00
103	Vaporub	vicks	020403	2020-07-01	2022-06-01	98	800.00
104	Montex FX	Sun Pharma	GHT14652	2020-11-01	2023-10-01	640	72.50
105	Dilo-DX	Corona Remedies	CL19210	2020-02-01	2022-01-01	100	688.00
106	Apple	Apple Formulations	DAC-2005	2020-04-01	2022-03-01	100	700.00
107	Julax	Shreya Life Sciences	R20411104	2020-09-01	2022-08-01	700	446.50
108	Dolarest	Cycus Pharma	273	2020-01-01	2021-12-01	200	840.00
109	Clinmiskin	Resilent	CL561	2019-05-01	2021-10-01	150	1380.00
110	Pacimol XP	Ipca Laboratories	BLG10800	2020-01-01	2021-12-01	650	9.17
111	Hansaplast	Varun Medimpexinc	93153711MP	2019-08-01	2021-07-01	2000	25.00

11 rows in set (0.00 sec)

```
mysql>
```

Table: Bill



```
mysql> select * from Bill;
```

Billno	Patient_Name	Date
1	Chetash Turkar	2021-02-03
2	Hemashree Turkar	2021-02-03
3	Apoorva Bagde	2021-02-03
4	Aman Rahangdale	2021-02-03

4 rows in set (0.00 sec)

```
mysql>
```

6.3 I/O FORMS DESIGN & EVENT CODING:

THE SOFTWARE PROJECT FOR **MEDICAL SHOP MANAGEMENT** CONTAINS VARIOUS FORMS ALONG WITH PROGRAMMING CODES. FORMS AND THEIR EVENT CODING ARE GIVEN BELOW.

```
# Python Project
# Program to manage medical shop using SQL, File

print("-----Medical Shop Management System-----\n-----USAVN Medical Shop-----\n")

#-----Imported Required Modules-----#

import time
from datetime import *
import csv
import os
import mysql.connector as sqltor # Connecting to Database

#-----Database Connection-----#

mycon = sqltor.connect(host = '127.0.0.1', user = 'root', passwd = 'root', database =
'CS2020_21') # Name of database is CS2020_21
cursor = mycon.cursor() # Creating Cursor

cursor.execute("CREATE TABLE IF NOT EXISTS Medical_Shop(Code INT PRIMARY KEY,
Name_Of_Med VARCHAR(40), Name_Of_Manu VARCHAR(40), Batch_No VARCHAR(15),
Manu_Date Date, Exp_Date Date, Quantity INT(4), Price_Per10 FLOAT(5,2))")
cursor.execute("CREATE TABLE IF NOT EXISTS Bill(Billno INT(5) PRIMARY KEY,Patient_Name
VARCHAR(40),Date Date)")

#-----Fetching Current date and time from database-----#

cursor.execute("select NOW()")    # getting current date with the help of my sql database
t = cursor.fetchone()
for row in t :
    Time = row
    #print("Time :", Time)

cursor.execute("select CURDATE()")    # getting current date with the help of my sql
database
d = cursor.fetchone()
for row in d :
    Date = row
    print("Date :", Date)

'''
We can also use
print(date.today())
'''
```



```

#-----function to reduce quantity of stock whenever a medicine is purchased-----#

def reduce():
    cursor = mycon.cursor() # creating cursor object

    quant = stockquant - quan

    queryup = 'update medical_shop set Quantity = {} where Code = {}'.format(quant, code) #
Updating the quantity
    cursor.execute(queryup)
    mycon.commit()

#-----Main Code-----#
#-----main menu-----#

answer1 = 'y'
while answer1.lower() == 'y' :

    print("\n-----")
    print("1. Update\View Stock On Database. \n2. Print Invoice/Bill for a Purchase.(Create a
file) \n3. Exit. ") # Choice
    ch = int(input("What do you want to do ? (1/2/3) : "))

#-----Adding Medicine To Database-----#
    if ch == 1 :
        if mycon.is_connected():
            print('Successfully Connected to MySQL Databases \n')

            print("-----")
            cursor = mycon.cursor() # creating cursor object

            print("-----Medicine Database-----\n\n")

            answer = 'y'
            while answer.lower() == 'y' :
                print("1. Add Medicine \n2. Remove Medicine \n3. Update Medicine Information \n4.
List Of Medicine \n5. Main Menu")
                ch1 = int(input("What do you want to do ? (1/2/3/4/5) : "))

                if ch1 == 1 : # Adding Medicine to stock
                    ans = 'y'
                    while ans.lower() == 'y':
                        print()
                        print("-----Adding Medicine to Stock-----\n")

                        code = int(input("Enter Code : "))
                        queryup = 'select * from medical_shop where Code = {}'.format(code)
                        cursor.execute(queryup)
                        result = cursor.fetchall()

```

```

if cursor.rowcount == 0 : # Checking for duplicate entry

    nmed = input("Enter Name of Medicine : ")
    nmanu = input("Enter Name of Manufacturer : ")
    batch = input("Enter Batch No. : ")
    mandate = input("Enter Date of Manufacturing (YYYY-MM-DD) : ")
    expdate = input("Enter Date of Expiry (YYYY-MM-DD) : ")

    a = date.fromisoformat(mandate)
    #print(a)
    b = date.fromisoformat(expdate)
    #print(b)

    if a < b :
        pass
    else:
        print('Invalid Input ! \nDate Expiry Date should be after Manufacturing date ')
        break

    quan = int(input("Enter Quantity : "))
    price = float(input("Enter Price Per 10 Tablets / 10 units : "))

    query = 'insert into medical_shop values({}, {}, {}, {}, {}, {}, {},'
    {}).format(code, nmed, nmanu, batch, mandate, expdate, quan, price)
    cursor.execute(query)
    mycon.commit()

    print('Data Saved Successfully')

    ans = input('Want to add more ?(y/n): ')

else:

    print('%10s'% 'Code', '%20s'% 'Name of Medicine', '%25s'% 'Name of
Manufacturer', '%15s'% 'Batch No.', '%25s'% 'Date of Manufacturer', '%15s'% 'Date of Expiry',
'%20s'% 'Quantity', '%18s'% 'Price')
    for row in result:
        print('%10s'% row[0], '%20s'% row[1], '%25s'% row[2], '%13s'% row[3],
'%18s'% row[4], '%20s'% row[5], '%20s'% row[6], '%21s'% row[7])
        print("Medicine with code", code, "Already Present !!!\n")
        break

#-----Deleting Medicine from Database-----#

elif ch1 == 2 :
    ans = 'y'
    while ans.lower() == 'y':
        print("-----Deleting Medicine from Stock-----\n")

```

```

nmed = input("Enter name of medicine to be Deleted: ")
query = 'select * from medical_shop where Name_of_Med = "{}".format(nmed)
cursor.execute(query)
result = cursor.fetchall()

if cursor.rowcount == 0 :
    tim.sleep(0.5)
    print("Sorry! medicine with ", nmed, "not found")

else:
    print('%10s'%Code, '%20s'%Name of Medicine, '%25s'%Name of
Manufacturer, '%15s'%Batch No.', '%25s'%Date of Manufacturer, '%15s'%Date of Expiry',
'%20s'%Quantity, '%18s'%Price')
    for row in result:
        print('%10s'%row[0], '%20s'%row[1], '%25s'%row[2], '%13s'%row[3],
'%18s'%row[4], '%20s'%row[5], '%20s'%row[6], '%21s'%row[7])
        code = row[0]
        su = input("\n\nWant to delete Sure(y/n): ")

        if su.lower() == 'y':
            queryde = 'delete from medical_shop where code = {}'.format(code)
            cursor.execute(queryde)
            mycon.commit()

            print('Record Deleted Successfully')

ans = input('Want to delete more?(y/n): ')

#-----Updating Medicine of previously entered Database-----#

elif ch1 == 3 : # Updating Medicine Information
    print("-----Updating Info of Medicine of Stock-----\n")
    ans = 'y'
    while ans.lower() == 'y':
        nmed = input("Enter name of medicine to be updated: ")
        queryup = 'select * from medical_shop where Name_of_Med =
"{}".format(nmed)
        cursor.execute(queryup)
        result = cursor.fetchall()

        if cursor.rowcount == 0 :

            print("Sorry! medicine ", nmed, "not found\n")

        else:
            print('%10s'%Code, '%20s'%Name of Medicine, '%25s'%Name of
Manufacturer, '%15s'%Batch No.', '%25s'%Date of Manufacturer, '%15s'%Date of Expiry',
'%20s'%Quantity, '%18s'%Price')
            for row in result:
                code = row[0]

```

```

        print('%10s'%row[0], '%20s'%row[1], '%25s'%row[2], '%13s'%row[3],
'%18s'%row[4], '%20s'%row[5], '%20s'%row[6], '%21s'%row[7])
        choice = input("Sure to Update(y/n) : ")

        if choice.lower() == 'y':
            print("---You can update only Batch No., Date of manufacturing and
expiry, Quantity and Price---")

            bno = input("Enter new batch No. (leave blank if you don't want to change
it): ")
            if bno == "":
                bno = row[3]

            Dman = input("Enter new Date of Manufacturing (leave blank if you don't
want to change it): ")
            if Dman == "":
                Dman = row[4]

            Dexp = input("Enter new Date of Expiry (leave blank if you don't want to
change it): ")
            if Dexp == "":
                Dexp = row[5]

            qua = input("Enter new Quantity(leave blank if you don't want to change
it): ")
            if qua == "":
                qua0 = row[6]
                qua = int(qua0)

            pri = input("Enter new Price (leave blank if you don't want to change it): ")
            if pri == "":
                pri0 = row[7]
                pri = int(pri0)

            queryup = 'update medical_shop set Batch_No = "{}", Manu_Date = "{}",
Exp_Date = "{}", Quantity = {}, Price_Per10 = {} where code = {}'.format(bno, Dman, Dexp,
qua, pri, code)
            cursor.execute(queryup)
            mycon.commit()

            print("Record updated succesfully\n")

            ans = input('Want to update more?(y/n): ')

#-----Getting Medicine(List) Details from Database-----#

elif ch1 == 4 :
    print("----- Medicines in Stock-----\n")
    query = 'select * from medical_shop'

```

```

        cursor.execute(query)
        result = cursor.fetchall()
        print('%10s'%Code, '%20s'%Name of Medicine', '%25s'%Name of Manufacturer',
'%15s'%Batch No.', '%25s'%Date of Manufacturer', '%15s'%Date of Expiry',
'%20s'%Quantity', '%18s'%Price')
        for row in result:
            print('%10s'%row[0], '%20s'%row[1], '%25s'%row[2], '%13s'%row[3],
'%18s'%row[4], '%20s'%row[5], '%20s'%row[6], '%21s'%row[7])

#-----If User want to go to main menu-----#

        elif ch1 == 5 :
            break

#-----In case invalid input is given-----#
        else :

            print("Invalid Input ! ")

#-----Shows option to choose menu and sub menu-----#

            print("-----\n")
            answer = input("For Database Menu Press y and Press Enter(return) for Main Menu :
")

#-----Code To Print Invoice-----#

        elif ch == 2 :

            print("Time :", Time)

#-----Getting Bill no. from database-----#

            query2 = 'select * from bill'
            cursor.execute(query2)
            result2 = cursor.fetchall()
            if result2 == [] :
                billno = 0
            else:
                for row in result2 :
                    billno = row[0]

#-----Invoice Interface-----#

            print("\n-----USAVN Medical Shop Tumsar-----
-----\n")
            billno = billno + 1 #int(input("Bill No. : "))
            print("Bill No.",billno)

            print("Date & Time :", Time)
            pname = input("Name of patient : ") # Patient name

```

```

query1 = 'select * from bill where Billno = {}'.format(billno)
cursor.execute(query1)
result1 = cursor.fetchall()

#-----inserting current billno. to database-----#

if result1 == [] :

    query0 = 'insert into bill values({}, {}, {})'.format(billno, pname, Date) # saving Bill
No to database
    cursor.execute(query0)
    mycon.commit()

#-----getting details about patient-----#

rdoc = input("Referred by Doctor : ")
add = input("Address of Patient : ")
n = int(input("Number of Medicines : ")) # getting the number of medicines prescribed
by doctor
fname = (pname+ str(billno)+".csv")

#-----Creating and opening a csv file using patient name and billno.-----#

with open(fname, mode = 'w', newline = '\n') as csvfile :

    date0 = Date.strftime("%A, %d %B %Y")
    time0 = str(Time)
    time1 = time0[11:19]
    #print(time1)

    filewriter = csv.writer(csvfile,delimiter = ',')

#-----Writing data onto csv file-----#

    filewriter.writerow(["-----USAVN Medical Shop
Tumsar-----"])
    filewriter.writerow(["Bill No. : "+ str(billno),",",",",",",", str(date0)])
    filewriter.writerow(["Patient's Name : " + pname ,",",",",",",", 'Issue Time : ' +
str(time1)])
    filewriter.writerow(["Address : " + add ])
    filewriter.writerow(["Referred by Doctor : " + rdoc ])
    filewriter.writerow([])
    filewriter.writerow(["Sr.No.", "Name of Medicine",",", " Manufacturing.Co.",",",
"Batch No.", "Man.Date", "Exp.Date", "Quantity", "Price", "Amount"])

    b = c =0
    total = amt = 0
    ans = 'y'
    for i in range(n):

```

```

print("\nSr. No.", i+1)

nmed0 = input("Enter name of medicine : ") # Using the name of medicine the
other details will be obtained from the database
nmed = nmed0.capitalize()
query1 = 'select * from medical_shop where Name_of_Med = "{}".format(nmed)
cursor.execute(query1)
result = cursor.fetchall()
for row in result:
    if row[6] <= 50 :
        print(row[1], 'is getting out of stock\n')
    else:
        pass

if result == []:
    print("Sorry! medicine with name ", nmed, "not found\n")
    b = 1
    c = 0
    csvfile.close()
    break

for row in result:    # Checking the existence of a medicine
    if cursor.rowcount != 0 :
        quan = int(input("Quantity : " ))
        code = row[0]    # Getting details according to name of medicine
        nmanu = row[2]    # Fetching data from database
        batchn = row[3]

        mandate0 = str(row[4])
        mandate = mandate0[0:7]

        expdate0 = str(row[5])
        expdate = expdate0[0:7]

        #print(mandate, expdate)
        stockquant = row[6]
        price = row[7]

        #print(price)
        oneprice = price / 10 # Calculating the amount of each tablet
        amt = quan * oneprice # Calculating the amount of One medicine

        text = [str(i+1), nmed,",", nmanu,",", str(batchn), str(mandate), str(expdate),
str(quan), str(price), str(amt)]
        filewriter.writerow(text)

    print()
    total = total + amt # Total Amount
    reduce() # Calling reduce function to reduce the purchased medicine from
database

```

```

        c = 1

    if c == 1:
        filewriter.writerow([])
        filewriter.writerow(["Total Amount: " + str(total)])
        filewriter.writerow(["Inclusive of all taxes. "])
        csvfile.close()

    print("\nInvoice Printed Successfully ")
    print("_____ \n")

#-----incase medicine not found-----#
    if b == 1:
        os.remove(fname)

        print("Invoice Printing Failed ! ")
        print("_____ ")

        queryde = 'delete from bill where billno = {}'.format(billno)
        cursor.execute(queryde)
        mycon.commit()
    else :

        print("Bill no.", billno, "present already\n")
        print('%10s'%Bill No.', '%20s'%Name of Patient', '%25s'%Date',)
        for row in result1 :
            print('%10s'%row[0], '%20s'%row[1], '%25s'%row[2])
        print()

#-----Closing connection with database when user quits-----#
    elif ch == 3 :
        break
        mycon.close() # closing connection with database

    else :
        print("Invalid Input")

```


7. User Manual

7.1 How to install Software:

Hardware Requirement-

- ◆ Intel Pentium/Celeron or similar processor based PC at Client/Server end.
- ◆ 128 MB RAM and 4GB HDD space (for Database) is desirable.
- ◆ Standard I/O devices like Keyboard and Mouse etc.
- ◆ Printer is needed for hard-copy reports.
- ◆ Local Area Network(LAN) is required for Client-Server Installation

Software Requirement-

- ◆ Windows 2000/XP/7 OS is desirable.
- ◆ Python 3.7 IDLE with required modules.
- ◆ MySQL Ver 5.1 with Library Database must be present at machine.

Database Installation

The software project is distributed with a backup copy of a Database named **CS2020_21** with required tables. Some dummy records are present in the tables for testing purposes, which can be deleted before inserting real data. The project is shipped with **Mohit.sql** file which installs a database and tables in the computer system.

Note: The PC must have MySQL server with user (**root**) and password (**root**) . If root password is any other password, it can be changed by running MySQL Server Instance Configure Wizard.

Start ► Program ► MySQL ► MySQL Server ► MySQL Server Instance Config Wizard

Provide current password of root and new password as “root” , this will change the root password.

To install a MySQL database from a dump file (**Mohit.sql**) , simply follow the following steps.

Step 1: Copy the **Mohit.sql** file in **C:\Program files\Mysql\MySql server 5.1\Bin** folder.

Step 2: Open MySQL and type the following command to create the database named **CS2020_21**.

```
mysql> create database CS2020_21;
```

Step 3: Open Command Window (Start ► Run ► cmd)

Step 4: Go to the following folder using CD command of DOS.

```
C:\Program files\Mysql\MySql server 5.1\Bin>
```

Step 5: type the following command on above prompt -

```
C:...\bin> mysql -u root -p Mohit CS2020_21 <Mohit.sql
```

This will create a **CS2020_21** database with required tables.

7.2 Screen Shots working with Software Project:

1. Adding medicine (inserting record into table) into database.

```
*Python 3.8.5 Shell*
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on w
in32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Desktop\CS XII\CS Project Mohit Gajbhiye\CS project Medical shop
(alter with name csv file) V2.8.py
-----Medical Shop Management System-----
-----USAVN Medical Shop-----

Date : 2021-02-03

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 1
Successfully Connected to MySQL Databases

-----
-----Medicine Database-----

1. Add Medicine
2. Remove Medicine
3. Update Medicine Information
4. List Of Medicine
5. Main Menu
What do you want to do ? (1/2/3/4/5) : 1

-----Adding Medicine to Stock-----

Enter Code : 112
Enter Name of Medicine : Asthalin DX
Enter Name of Manufacturer : Cipla
Enter Batch No. : CECK1932
Enter Date of Manufacturing (YYYY-MM-DD) : 2020-05-01
Enter Date of Expiry (YYYY-MM-DD) : 2022-04-01
Enter Quantity : 100
Enter Price Per 10 Tablets / 10 units : 726
Data Saved Successfully
Want to add more ?(y/n): n

-----
For Database Menu Press y and Press Enter(return) for Main Menu :

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 3
>>>
```

Ln: 49 Col: 4

2. Removing medicine (deleting record from table) from database.

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Desktop\CS XII\CS Project Mohit Gajbhiye\CS project Medical shop (alter with name csv file) V2.8.py
-----Medical Shop Management System-----
-----USAVN Medical Shop-----

Date : 2021-02-03

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 1
Successfully Connected to MySQL Databases

-----
-----Medicine Database-----

1. Add Medicine
2. Remove Medicine
3. Update Medicine Information
4. List Of Medicine
5. Main Menu
What do you want to do ? (1/2/3/4/5) : 2
-----Deleting Medicine from Stock-----

Enter name of medicine to be Deleted: Hansaplast
Code Name of Medicine      Name of Man.  Batch No.  Man. Date  Exp.Date  Quantity  Price
111      Hansaplast      Varun Medics  L201245  2020-01-01  2022-12-01  2000      25.0

Want to delete Sure(y/n): y
Record Deleted Successfully
Want to delete more ?(y/n): n
-----

For Database Menu Press y and Press Enter(return) for Main Menu :

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 3
>>> |
```

Ln: 46 Col: 4

3. Updating medicine (updating record of table) of database.

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Desktop\CS XII\CS Project Mohit Gajbhiye\CS project Medical shop (alt
er with name csv file) V2.8.py
-----Medical Shop Management System-----
-----USAVN Medical Shop-----

Date : 2021-02-03

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 1
Successfully Connected to MySQL Databases

-----
-----Medicine Database-----

1. Add Medicine
2. Remove Medicine
3. Update Medicine Information
4. List Of Medicine
5. Main Menu
What do you want to do ? (1/2/3/4/5) : 3
-----Updating Info of Medicine of Stock-----

Enter name of medicine to be updated: Montemac-L
Code Name of Medicine      Name of Man.  Batch No.  Man. Date  Exp.Date  Quantity  Price
102      Montemac-l      Macleods      YGF4154  2020-02-01  2023-01-01      490      123.0
Sure to Update(y/n) : y
---You can update only Batch No., Date of manufacturing and expiry, Quantity and Price---
Enter new batch No. (leave blank if you don't want to change it):
Enter new Date of Manufacturing (leave blank if you don't want to change it):
Enter new Date of Expiry (leave blank if you don't want to change it):
Enter new Quantity(leave blank if you don'twant to change it): 690
Enter new Price (leave blank if you don't want to change it): 123.75
Record updated succesfully

Want to update more ?(y/n): n

-----
For Database Menu Press y and Press Enter(return) for Main Menu :

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 3
>>> |
```

Ln: 51 Col: 4

4. Showing list of medicine (display record of table) in database.

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Desktop\CS XII\CS Project Mohit Gajbhiye\CS project Medical shop (alter with name csv file) V2.8.py
-----Medical Shop Management System-----
-----USAVN Medical Shop-----

Date : 2021-02-03

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 1
Successfully Connected to MySQL Databases

-----
-----Medicine Database-----

1. Add Medicine
2. Remove Medicine
3. Update Medicine Information
4. List Of Medicine
5. Main Menu
What do you want to do ? (1/2/3/4/5) : 4
----- Medicines in Stock-----

Code Name of Medicine      Name of Man.  Batch No.  Man. Date  Exp.Date  Quantity  Price
101      Medler      Comed Chemicals  RET5752  2020-12-01  2023-11-01      560      42.0
102      Montemac-1      Macleods      YGF4154  2020-02-01  2023-01-01      690      123.75
103      Vaporub      vicks      020403  2020-07-01  2022-06-01      97      800.0
104      Montex FX      Sun Pharma      GHT14652  2020-11-01  2023-10-01      640      72.5
105      Dilo-DX      Corona Remedies  CL19210  2020-02-01  2022-01-01      99      688.0
106      Apple      Apple Formulations  DAC-2005  2020-04-01  2022-03-01      99      700.0
107      Julax Shreya Life Sciences  R20411104  2020-09-01  2022-08-01      700      446.5
108      Dolarest      Cycus Pharma      273  2020-01-01  2021-12-01      199      840.0
109      Clinmiskin      Resilent      CL561  2019-05-01  2021-10-01      149      1380.0
110      Pacimol XP      Ipca Laboratories  BLG10800  2020-01-01  2021-12-01      642      9.17
112      Asthalin DX      Cipla      CECK1932  2020-05-01  2022-04-01      100      726.0

-----
For Database Menu Press y and Press Enter(return) for Main Menu : y
1. Add Medicine
2. Remove Medicine
3. Update Medicine Information
4. List Of Medicine
5. Main Menu
What do you want to do ? (1/2/3/4/5) : 5

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 3
>>>
```


5. Printing invoice (making csv file) by fetching data from database.

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on w
in32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\User\Desktop\CS XII\CS Project Mohit Gajbhiye\CS project Medical shop
(alter with name csv file) V2.8.py
-----Medical Shop Management System-----
-----USAVN Medical Shop-----

Date : 2021-02-03

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 2
Time : 2021-02-03 19:56:43

-----USAVN Medical Shop Tumsar-----
-----

Bill No. 5
Date & Time : 2021-02-03 19:56:43
Name of patient : Shushant Shrestha
Referred by Doctor : J.B. Laddha
Address of Patient : Tumsar
Number of Medicines : 3

Sr. No. 1
Enter name of medicine : Medler
Quantity : 15

Sr. No. 2
Enter name of medicine : Dilo-DX
Quantity : 1

Sr. No. 3
Enter name of medicine : Pacimol xp
Quantity : 10

Invoice Printed Successfully

-----
1. Update\View Stock On Database.
2. Print Invoice/Bill for a Purchase.(Create a file)
3. Exit.
What do you want to do ? (1/2/3) : 3
>>>
```

7.2.1 Screenshot of csv file

Shushant Shrestha5 - Excel

FILE HOME INSERT PAGE LAYOUT FORMULAS DATA REVIEW VIEW

Clipboard Font Alignment Number Conditional Formatting Styles Insert Delete Format Sort & Find & Filter Select Editing

A1 : -----USAVN Medical Shop Tumsar-----

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	-----USAVN Medical Shop Tumsar-----																	
2	Bill No. : 5										Wednesday, 03 February 2021							
3	Patient's Name : Shushant Shrestha										Issue Time :19:56:43							
4	Address : Tumsar																	
5	Referred by Doctor : J.B. Laddha																	
6																		
7	Sr.No.	Name of Medicine	Manufacturing.Co.	Batch No.	Man.Date	Exp.Date	Quantity	Price	Amount									
8	1	Medler	Comed Chemicals	RET5752	2020-12	2023-11	15	42	63									
9	2	Dilo-dx	Corona Remedies	CL19210	2020-02	2022-01	1	688	68.8									
10	3	Pacimol xp	Ipca Laboratories	BLG10800	2020-01	2021-12	10	9.17	9.17									
11																		
12	Total Amount: 140.97																	
13	Inclusive of all taxes.																	
14																		
15																		
16																		
17																		
18																		

Shushant Shrestha5

READY 100%

8. References

In order to work on this project titled - *Medical Shop Management System*, the following books and literature are referred by me during the various phases of development of the project.

1. MySQL, Black Book -by Steven Holzner
2. Understanding SQL – by Gruber
3. Head First Python -- Head First Series
4. <http://www.mysql.org/>
5. <http://www.python.org/>
6. On-line Help of Python IDLE ®
7. Computer Science for class XII -by Sumita Arora
8. Python Docs
9. Various Websites of Discussion Forum and software development activities.

Other than the above-mentioned books, the suggestions and supervision of my teacher and my class experience also helped me to develop this software project.