A Project Report

on

TALLY

For AISSCE 2021 Examination

[As a part of the cOMPUTER sciENce Course (083)]

SUBMITTED BY: -

NAME: NIPUN GUPTA

CBSE rOLL nO.:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Under the Guidance of: Ms.Ritu Jain

PGT (Comp.Sc)

**CERTIFICATE**

This is to certify that the Project entitled TALLY is a bonafide work done by NIPUN GUPTA of class XII Session 2020-21 in partial fulfillment of CBSE’s AISSCE Examination 2021 and has been carried out under my direct supervision and guidance. This report or a similar report on the topic has not been submitted for any other examination and does not form a part of any other course undergone by the candidate.

## ………………………… ……………………………..

## Signature of Signature of

## External Examiner Internal Examiner

**Name: Name: Ms**. **Ritu** **Jain**

**ACKNOWLEDGEMENT**

I

, undertook this Project work, as the part of my XII-Computer Science course(083). I had tried to apply my best of knowledge and experience, gained during the study and class work experience. However, developing software system is generally a quite complex and time-consuming process. It requires a systematic study, insight vision and professional approach during the design and development. Moreover, the developer always feels the need, the help and good wishes of the people near you, who have considerable experience and idea.

I would like to extend my sincere thanks and gratitude to my teacher

**Ms. Ritu Jain,** for giving valuable time and moral support to develop this software.

I also feel indebted to my friends for the valuable suggestions during the project work.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Class XII**

**C O N T E N T S**

1**. Introduction----** ------------5

**2. Objective & Scope of the Project** 7

**3. System Implementation** 9

3.1 The Hardware used: 9

3.2 The Softwares used: 9

**4. Theoretical Background**---------------------------------10

4.1 What is Python? ------ ---------------------------------------------10

4.2 What is File Handling --------------------------------------------12

4.3 What is Database? -----------------------------------------------13

4.4 What is MySQL?--------------------------------------------------14

**5. System Design & Development** ------18

5.1 Database Design: 18

5.2 Event Coding: 20

**6. Output----------------------------------------------------------------------46**

**7. User Manual** 53

7.1 How to install: 53

7.2 Working with Software: 54

**8. References**  55

**1. Introduction**

This software project is developed to automate the functionalities of a **TALLY APLICATION.**

The purpose of the software project is to develop the Management Information System (MIS) and to automate the record keeping of accounts, debiting, crediting and other details of the account 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 like reference of acc\_id in all tables , 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(Python) 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 TALLY APPLICATION.

During coding and design of the software Project, Python IDLE, as a powerful front-end tool is used for getting Graphical User Interface (GUI) based integrated platform and coding simplicity. As a back-end a powerful, open source RDBMS, MySQL is used as per requirement of the CBSE curriculum of Computer Science Course(083).

**2. Objective & Scope of the Project**

T

he objective of the software project is to develop a computerized MIS and to automate the functions of a **TALLY APPLICATIONS** used by CHARTERED ACCOUNTANTS. This software project is also aimed to enhance the current record keeping system, which will help CA to retrieve the up-to-date information at right time in right shape.

The proposed software system is expected to do the following functionality-

* To provide a user friendly, Graphical User Interface (GUI) based integrated and centralized environment for MIS activities.
* The proposed system should maintain all the records and transactions, and should generate the required reports and information when required.
* To provide graphical and 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  **TALLY APPLICATION** 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 has only limited functions compared to a REAL TALLY APPLICATION.
2. The program takes a lot of time to load.

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. System Implementation**

## 

## 3.1 The Hardware used:

While developing the system, the used hardware are:

PC with Intel Core i3-7th Gen processor having 4.00 GB RAM, 64-bit Operating System , SVGA and other required devices.

## 

## 3.2 The Softwares used:

* Microsoft Windows® 10 Pro as Operating System.
* Python 3.7.2 as Front-end Development environment.
* MySQL as Back-end Sever with Database for Testing.
* PyMySql to connect Python module with database.
* SMTP, Email to send Emails to Me.
* Pillow to import Images and Logos.
* MS-Word 2010 for documentation.

**4. Theoretical Background**

4.1 **What is Python ?**

Python is an open source , object oriented high level programming language developed by Guido Van Rossum in 1991 at the National Research Institute for Mathematics,Netherlands.

Features of Python:

* It is an interactive ,interpreted language.
* It is a loosely typed object –oriented language.
* It is a free open –source and portable language,
* It takes less time to develop programs.
* It is extensible / extendable and highly efficient .
* It supports GUI.
* It can be easily compatible with other languages like C , C++ etc.
* It is used for both scientific and non-scientific programming.

**Installing Python:**

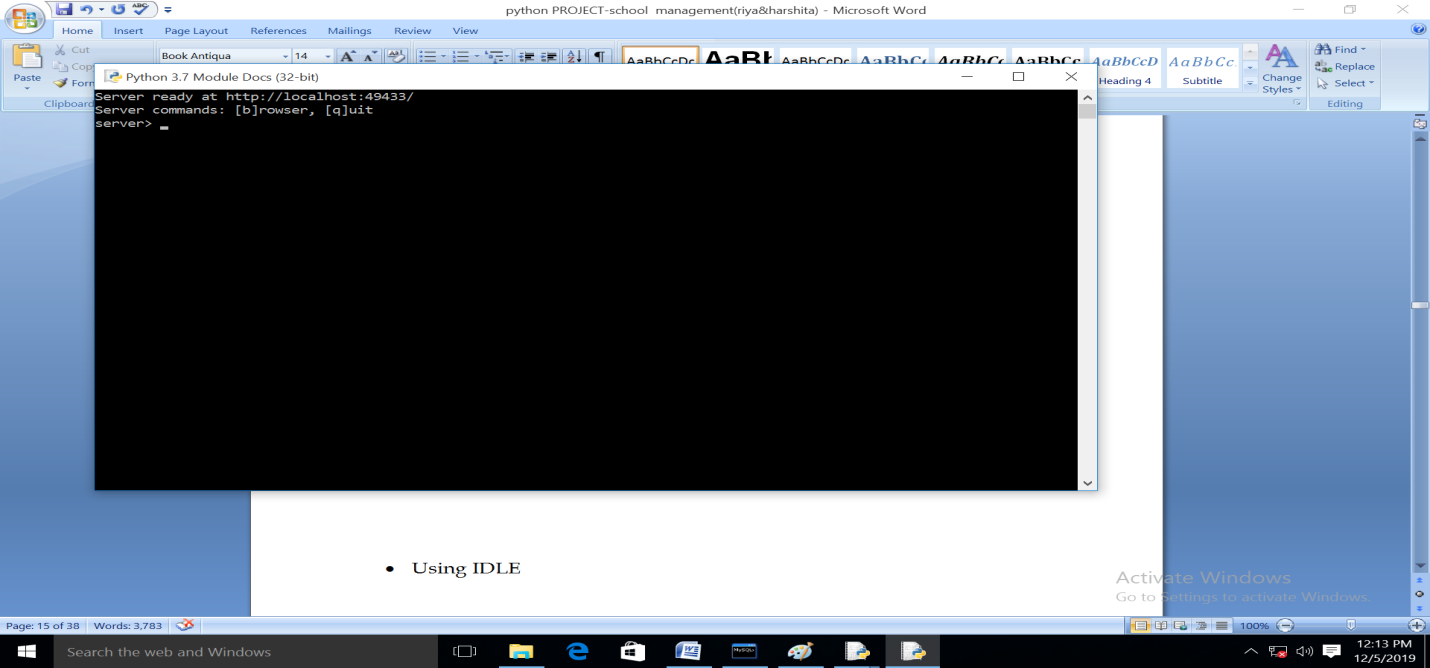
It can be installed by using website :

<https://www.python.org/downloads/>

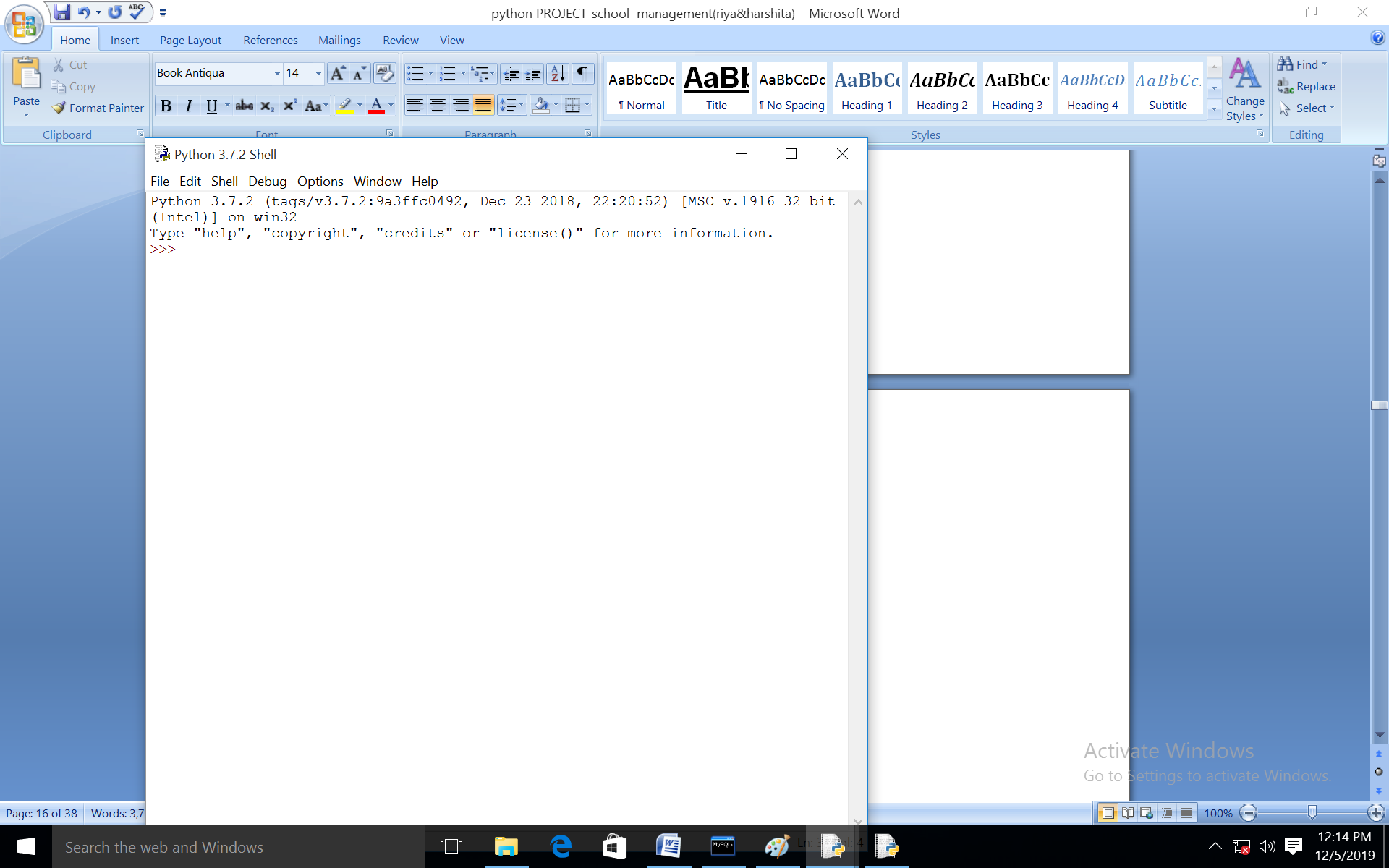
**Interacting with Python:**

Python programs can be run in two ways:

* Using Command line window



* Using IDLE



## 4.2 What is File Handling?

A file in itself is a bunch of bytes stored on some storage devices like hard-disk, thumb-drive etc.

The data files can be stored in two ways:

**Text files**

A text file stores information in ASCII or Unicode characters, where each line of text is terminated, (delimited) with a sepcial character known as EOL (End of Line) character. In text files some internal translations take place when this EOL character is read or written.

A text file can be opened in these file modes: 'r', 'w', 'a', 'r+', 'w+', 'a+'

**Binary files**

A binary file is just a file that contains information in the same format in which the information is held in memory, i.e., the file content that is returned to you is raw (with no translation or no specific encoding). The open() function is used to open a data file in a program through a file-object (or a file-handle). A file-mode governs the type of operations (e.g., read/ write/ append) possible in the opened file i.e., it refers to how the file will be used once it's opened.

A binary file can be opened in these file modes: 'rb', 'wb', 'ab', 'r+b'('rb+'), 'w+b'('wb+'); a+b'('ab+').

**Read and Write Functions:**

The three file reading functions of Python are: read(), readline(), readlines() While read() reads some bytes from the file and returns it as a string, readline() reads a line at a time and readlines() reads all the lines from the file and returns it in the form of a list.

The two writing functions for Python data files are write() and writelines(). While write() writes a string in file, writelines() writes a list in a file.

The input and output devices are implemented as files, also called standard streams. There are three standard streams: stdin (standard input), stdout (standard output) and stderr (standard error)

**Absolute and Relative Path**

The absolute paths are from the topmost level of the directory structure. The relative paths are relative to current working directory denoted as a dot(.) while its parent directory is denoted with two dots(..). The full name of a file or a directory is called pathname.

**Steps to Process a file:**

There are four steps to use files in the python program.

1. Determine the type of file usage Under this step, you need to determine whether you need to open the file for reading purpose (input type of usage) or writing purpose (output type of usage).
2. Open the file and assign its reference to a file-object or file-handle Next, you need to open the file using open() and assign it to a file-handle on which all the file-operations will be performed. Just remember to open the file in the filemode that you decided in step 1.
3. Now process as required as per the situation, you need to write instructions to process the file as desired. For example, you might need to open the file and then read it one line at a time while making some computation, and so on.
4. Close the file. This is very important step especially if you have opened the file in write mode. This is because, sometimes the last lap of data remains in buffer and is not pushed on to disk until a close() operation is performed.

## 4.3 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 and retrieve the data using queries.

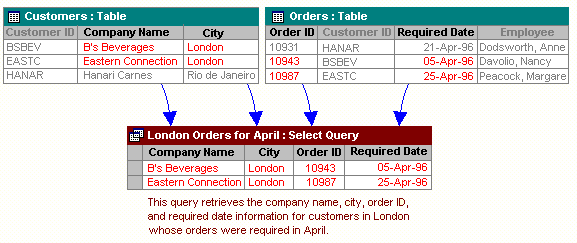
A table is a collection of data about a specific topic, such as products or suppliers. Using a separate table for each topic means you can store that data only once, which makes your database more efficient and reduces data-entry errors. Table organises data into columns (called fields) and rows (called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

## 4.4 What is My SQL ? logo-mysql

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 (our 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 backends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

**The Main Features of MySQL are :**

* Written in C and C++.
* Works on many different platforms.
* Uses multi-layered server design with independent modules.
* Provides transactional and nontransactional 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.

5**. System Design & Development**

## 5.1 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 contraints 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 class12 which contains the following tables:

### Table Design:

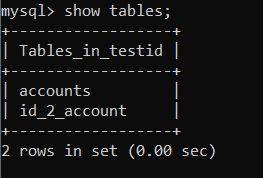
The database of **TALLY PROJECT** contains 1 main table of that particular user to store the account details of his client and tables with the clients debit and credit transaction in a database with the user name and database changes if logged in with a different account. The tables are normalized to minimize the redundancies of data and enforcing the validation rules of the organization. Most of the tables are designed using python’s inbuilt code and do not need to be crated separately. The tables and their structure are given below:

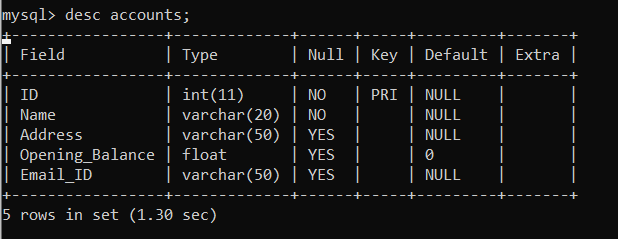
**DBMS: MySQL**

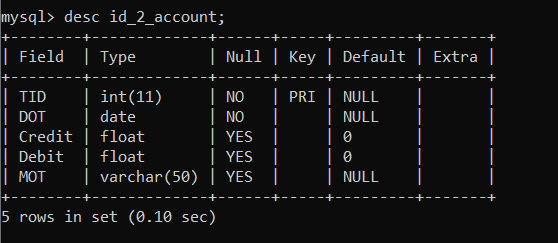
**Host: localhost , User: root , Pass: nipun , DataBase: nipun**

**Table Structure: (Images Below)**

Mysql commands :



****

****

**#THESE TABLES WERE CREATED AFTER I HAD EXECUTED MY PYTHON CODE AS DONE FROM THE OUTPUT SECTION.**

**PYTHON CODE :**

*from tkinter import \**

*from tkinter import messagebox*

*from PIL import ImageTk,Image #importing all modules needed for my program*

*import pickle*

*import pymysql*

*master = Tk() #main window*

*master.title("Tally Project")*

*icon\_photo = PhotoImage(file = "logo.png")*

*master.iconphoto(True,icon\_photo) #to set a photo as an icon*

*master.configure(bg="blue")*

*master.geometry('450x350') #change size and color*

*lsin\_frame = LabelFrame(master,padx=10, pady=10,relief= SUNKEN) #frame for log/sign in buttons*

*lsin\_frame.pack(padx=10,pady=10)*

*lsin\_frame.configure(bg = "aquamarine")*

*logo\_img=Image.open("logo.png")*

*logo\_img=logo\_img.resize((300, 250), Image.ANTIALIAS)*

*my\_logo\_img = ImageTk.PhotoImage(logo\_img)*

*logo\_label = Label(lsin\_frame,image=my\_logo\_img) #printing an image*

*logo\_label.grid(row=0,column=0, columnspan=2)*

*str\_criteria="" #will be used in change\_the\_value function*

*def sign():*

*global Id\_entry #globalling variables to use again in other functions*

*global pass\_entry*

*global con\_pass\_entry*

*global top1*

*top1 = Toplevel()*

*top1.title("Sign in")*

*lbl1=Label(top1,text="CREATE YOUR USER ID :").grid(row=0,column=0) #to show where to enter id*

*lbl2=Label(top1,text="CREATE YOUR PASSWORD :").grid(row=1,column=0)*

*lbl3=Label(top1,text="CONFIRM YOUR PASSWORD :").grid(row=3,column=0)*

*lbl4=Label(top1,text="minimum 8 characters",fg="red").grid(row=2,column=0)*

*Id\_entry=Entry(top1) #here user enter the values*

*Id\_entry.grid(row=0,column=1)*

*pass\_entry=Entry(top1,show="\*") #it shows what we enter as an asterisk*

*pass\_entry.grid(row=1,column=1)*

*con\_pass\_entry=Entry(top1,show="\*")*

*con\_pass\_entry.grid(row=3,column=1)*

*sign\_submit\_but=Button(top1,text="SUBMIT",bg="light blue",command=sign\_submit).grid(row=4,column=0)*

*#buttons can be given a command which performs a function*

*def sign\_submit(): #binary file for password is already created*

*Id=Id\_entry.get()*

*Pass=pass\_entry.get() #grt gives the value of what we enter*

*con\_pass=con\_pass\_entry.get()*

*if (len(Id)<=0):*

*messagebox.showwarning("Warning","Invalid ID") #ID parameter that it can't be null*

*elif(len(Pass)<8):*

*messagebox.showwarning("Warning","Password should be of atleast 8 characters") #Password can't nbe less than 8 char*

*elif(con\_pass!=Pass):*

*messagebox.showwarning("Warning","Invalid Password") #Pass has to be equal to confirm pass*

*else:*

*try:*

*g=open("Password",'rb')*

*f=open("Password",'ab') #storing in binary file to maintain the secrecy*

*found=0 # a counter variable to check if id exists or not*

*while True:*

*data=pickle.load(g)*

*d=data.keys()*

*for i in d:*

*if(i==Id.upper()):*

*messagebox.showwarning("Warning","Sorry!This ID is already there") # messagebox shows a messagebox with warning,info etc.*

*found=1*

*except EOFError:*

*g.close()*

*if found!=1:*

*z={Id.upper():Pass} #storing ID,Password as dictionary*

*pickle.dump(z,f) # dumping values in binary file*

*f.close()*

*top1.destroy()*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun')*

*cursor = myCon.cursor() #connecting to database*

*cursor.execute("CREATE DATABASE "+Id)#creaating ID as daabase*

*cursor.execute("use "+Id) #creating table to store data of different accounts(user created) in mysql*

*cursor.execute("CREATE TABLE IF NOT EXISTS Accounts(ID integer PRIMARY KEY,Name varchar(20) NOT NULL,Address varchar(50),Opening\_Balance float DEFAULT 0.00,Email\_ID varchar(50) )")*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.commit()*

*myCon.close()*

*def log(): #logging in the program to access its features*

*global user\_id*

*global user\_pass*

*global top2*

*top2 = Toplevel()*

*top2.title("Log in")*

*lbl1=Label(top2,text="ENTER YOUR USER ID :").grid(row=0,column=0)*

*lbl2=Label(top2,text="ENTER YOUR PASSWORD :").grid(row=1,column=0)*

*user\_id=Entry(top2)*

*user\_id.grid(row=0,column=1)*

*user\_pass=Entry(top2,show="\*")*

*user\_pass.grid(row=1,column=1)*

*log\_submit\_but=Button(top2,text="SUBMIT",bg="light blue",command=log\_submit).grid(row=3,column=0)*

*def log\_submit():*

*global l1 #globalling l1 as used in many places*

*l1=user\_id.get()*

*l2=user\_pass.get()*

*found=1 #again maintaining counter to check ID and Password*

*try:*

*g=open("Password",'rb')*

*while True:*

*data=pickle.load(g)*

*d=data.keys()*

*for i in d:*

*if i==l1.upper():*

*if data[i]==l2:*

*found=0*

*break*

*except EOFError:*

*g.close()*

*if found!=1:*

*top2.destroy()*

*lsin\_frame.destroy() #destroying frame to edit my main window*

*tally() #the main structure function call*

*else:*

*forgot\_pass\_but=Button(top2,text="Forgot Password click here!",command=forgot\_pass).grid(row=2,column=0)*

*messagebox.showwarning("Warning","Sorry!This ID or Password is incorrect!")*

*def forgot\_pass():*

*top2.destroy()*

*global p1*

*global p2*

*global top3*

*top3=Toplevel()*

*lbll=Label(top3,text="Type your ID :").grid(row=0,column=0)*

*lbl=Label(top3,text="Type your new password :").grid(row=1,column=0)*

*p1=Entry(top3)*

*p1.grid(row=0,column=1)*

*p2=Entry(top3)*

*p2.grid(row=1,column=1)*

*butttt=Button(top3,text="Enter",command=pass\_enter)*

*butttt.grid(row=2,column=0,columnspan=2)*

*def pass\_enter():*

*r=p1.get()*

*new\_pass=p2.get()*

*try:*

*found=1*

*h=open("Password",'rb+')*

*d=[]*

*while True:*

*d.append(pickle.load(h))*

*except EOFError:*

*for i in d:*

*for j in i.keys():*

*if j==r.upper():*

*if len(new\_pass) >=8:*

*i[j]=new\_pass*

*found=0*

*else:*

*found=2*

*break*

*if found==0:*

*h.seek(0)*

*for i in d:*

*pickle.dump(i,h)*

*h.close()*

*messagebox.showinfo("Succesful","Your password is changed successfully")*

*top3.destroy()*

*elif found==2:*

*messagebox.showwarning("Warning","Password should be of atleast 8 characters") #Password can't nbe less than 8 char*

*else:*

*h.close()*

*top3.destroy()*

*messagebox.showwarning("OOPS!!","ID does not exist")*

*def tally():*

*master.configure(bg='darkcyan')*

*master.geometry('850x500')*

*master.title("Tally Project") #changing parameters of the main window*

*welcome\_lbl=Label(master,text="WELCOME "+l1.upper()+" !!!",bg="lavender",fg="darkcyan",bd=4,relief=SUNKEN,pady=50)*

*welcome\_lbl.grid(row=0,column=0,columnspan=5,sticky=W+E) #length of label is from left to right*

*welcome\_lbl.config(font=("algerian",50))*

*create\_acc\_but=Button(master,text="CREATE ACCOUNT",padx=30,pady=10,command=create\_account).grid(row=1,column=0,pady=100,padx=25)*

*acc\_info\_but=Button(master,text="ACCOUNT INFO",padx=30,pady=10,command=account\_info).grid(row=1,column=1,padx=25)*

*acc\_ent\_but=Button(master,text="ACCOUNT ENTRIES",padx=30,pady=10,command=account\_entries\_fn).grid(row=1,column=2,padx=25)*

*bal\_sheet\_but=Button(master,text="BALANCE SHEET",padx=30,pady=10,command=balance\_sheet).grid(row=1,column=3,padx=25)*

*exit\_but=Button(master,text="EXIT",padx=30,pady=10,command=master.destroy).grid(row=2,column=1,padx=50,columnspan=2)*

*bfeedback\_but=Button(master,text="FEEDBACK",padx=30,pady=10,command=feedback).grid(row=2,column=0,padx=50,columnspan=2)*

*help\_but=Button(master,text="HELP",padx=30,pady=10,command=help\_tally).grid(row=2,column=2,padx=50,columnspan=2)*

*def create\_account(): #creating accounts with inputs like ID,Name,*

*global create\_ID #Opening Balance etc. and store in table in sql*

*global create\_name*

*global create\_address*

*global create\_openingbal*

*global create\_email*

*global top4*

*top4=Toplevel()*

*top4.title("Create Account")*

*lbl1=Label(top4,text="\*Enter ID(no.) :").grid(row=0,column=0,padx=10,pady=(10,0))*

*lbl2=Label(top4,text="\*Enter Name :").grid(row=1,column=0,padx=10)*

*lbl3=Label(top4,text="\*Enter Address :").grid(row=2,column=0,padx=10)*

*lbl4=Label(top4,text="\*Enter Opening Balance :").grid(row=3,column=0,padx=10)*

*lbl5=Label(top4,text="\*Enter Email ID :").grid(row=4,column=0,padx=10)*

*create\_ID=Entry(top4,width=60,fg="Blue")*

*create\_ID.grid(row=0,column=1,pady=(10,0))*

*create\_name=Entry(top4,width=60,fg="Blue")*

*create\_name.grid(row=1,column=1)*

*create\_address=Entry(top4,width=60,fg="Blue")*

*create\_address.grid(row=2,column=1)*

*create\_openingbal=Entry(top4,width=60,fg="Blue")*

*create\_openingbal.grid(row=3,column=1)*

*create\_email=Entry(top4,width=60,fg="Blue")*

*create\_email.grid(row=4,column=1)*

*create\_acc\_submit\_but=Button(top4,text="SUBMIT",command=create\_acc\_submit)*

*create\_acc\_submit\_but.grid(row=5,column=0,columnspan=2)*

*def create\_acc\_submit():*

*found=0*

*try:*

*l=create\_ID.get()*

*m=create\_name.get()*

*n=create\_address.get()*

*o=create\_openingbal.get() #taking value from the entry fields*

*p=create\_email.get()*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun')*

*cursor = myCon.cursor()*

*cursor.execute("use "+l1) #storing them in the table accounts*

*cursor.execute("INSERT into Accounts values("+str(l)+",\""+m+"\",\""+n+"\","+str(o)+",\""+p+"\")")*

*cursor.execute("CREATE TABLE id\_"+str(l)+"\_account(TID integer PRIMARY KEY,DOT date NOT NULL,Credit float DEFAULT 0.00,Debit float DEFAULT 0.00,MOT varchar(50) )")*

*myCon.commit() #creating table with account ID to store any entries of debit and credit*

*except Warning as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*top4.destroy()*

*def account\_info():*

*global top5*

*global acc\_id*

*top5=Toplevel()*

*top5.title("Account Infrormation")*

*search\_acc\_name=Label(top5,text="Enter the ID of the Account :").grid(row=0,column=0,pady=10)*

*acc\_id=Entry(top5,fg="Blue")*

*acc\_id.grid(row=0,column=1) # taking value of the Id to display and change account info*

*search\_acc\_but=Button(top5,text="Enter To Check",command=search\_account)*

*search\_acc\_but.grid(row=0,column=2,columnspan=2)*

*def search\_account():*

*global record\_val*

*acc\_id\_val\_inp=acc\_id.get()*

*if acc\_id\_val\_inp.isnumeric()== True: #checking if the Id entered is a number and not any other char*

*acc\_id\_val=acc\_id\_val\_inp*

*else:*

*messagebox.showwarning("Warning","Sorry!It has to be a number!")*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor() #cheking if the Id exists or not*

*check\_record="SELECT EXISTS(SELECT \* from Accounts WHERE ID="+str(acc\_id\_val)+")"*

*cursor.execute(check\_record)*

*check\_record\_tupval=cursor.fetchone()*

*check\_record\_numval=check\_record\_tupval[0]*

*if check\_record\_numval==1: #if Id exists then display info of the account*

*cursor.execute("SELECT \* from Accounts where ID="+str(acc\_id\_val))*

*record\_val=cursor.fetchone() #fetch the value from table accounts*

*acc\_id\_name\_1=Label(top5,text="Name : ").grid(row=1,column=0) #its in the form of a tuple*

*acc\_id\_name\_2=Label(top5,text=str(record\_val[1]),fg="Blue").grid(row=1,column=1,sticky=W)*

*acc\_id\_name\_change=Button(top5,text="Change Name",command= lambda: change\_the\_value("Name"))*

*acc\_id\_name\_change.grid(row=1,column=2,columnspan=2)*

*acc\_id\_address\_1=Label(top5,text="Address : ").grid(row=2,column=0)*

*acc\_id\_addess\_2=Label(top5,text=str(record\_val[2]),fg="Blue").grid(row=2,column=1,sticky=W)*

*acc\_id\_address\_change=Button(top5,text="Change Address",command= lambda: change\_the\_value("Address"))*

*acc\_id\_address\_change.grid(row=2,column=2,columnspan=2)*

*acc\_id\_opening\_1=Label(top5,text="Opening Balance : ").grid(row=3,column=0)*

*acc\_id\_opening\_2=Label(top5,text=str(record\_val[3]),fg="Blue").grid(row=3,column=1,sticky=W)*

*acc\_id\_opening\_change=Button(top5,text="Change Opening Balance",command= lambda: change\_the\_value("Opening\_Balance"))*

*acc\_id\_opening\_change.grid(row=3,column=2,columnspan=2)*

*acc\_id\_email\_1=Label(top5,text="Email ID : ").grid(row=4,column=0)*

*acc\_id\_email\_2=Label(top5,text=str(record\_val[4]),fg="Blue").grid(row=4,column=1,sticky=W)*

*acc\_id\_email\_change=Button(top5,text="Change Email ID",command= lambda: change\_the\_value("Email\_ID"))*

*acc\_id\_email\_change.grid(row=4,column=2,columnspan=2)*

*del\_acc\_but=Button(top5,text="Delete this account",command=delete\_account)*

*del\_acc\_but.grid(row=5,column=1,columnspan=2)*

*close\_top5=Button(top5,text="CLOSE",command=top5.destroy)*

*close\_top5.grid(row=5,column=0,pady=10) #button to close the window*

*else:*

*messagebox.showwarning("Warning","Sorry!This ID dos not exist!")*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*def change\_the\_value(str\_criteria):*

*global top6 #window where the new value for name,email,opening balance is entered*

*global change\_acc\_entry*

*top6=Toplevel()*

*top6.title("Change Value of "+str\_criteria)*

*change\_acc\_val=Label(top6,text="Change "+str\_criteria).grid(row=0,column=0)*

*change\_acc\_entry=Entry(top6,width=60,fg="Blue")*

*change\_acc\_entry.grid(row=0,column=1)*

*comm\_change\_sub=Button(top6,text="Submit",command= lambda: submit\_change\_value(str\_criteria))*

*comm\_change\_sub.grid(row=1,column=0,columnspan=2)*

*def submit\_change\_value(str\_criteria):*

*change\_acc\_inp\_val=change\_acc\_entry.get()*

*if str\_criteria=="Name": #giving # crit\_val a value acc to the str\_criteria to change accordingly*

*crit\_val=1*

*elif str\_criteria=="Address":*

*crit\_val=2*

*elif str\_criteria=="Opening\_Balance":*

*crit\_val=3*

*elif str\_criteria=="Email\_ID":*

*crit\_val=4*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*change\_val\_query="UPDATE Accounts SET "+str\_criteria+" = \""+change\_acc\_inp\_val+"\" where "+str\_criteria+" = \""+str(record\_val[crit\_val])+"\""*

*cursor.execute(change\_val\_query) #updating value in accounts according str\_criteria*

*myCon.commit() #commiting the changes*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*top6.destroy()*

*end\_top5=Label(top5,text="Please close the window to review changes!!",fg="Red",font=("Algerian",16)).grid(row=5,column=0,columnspan=3)*

*def delete\_account():*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*del\_acc\_id=acc\_id.get()*

*cursor.execute("DELETE from Accounts where ID = "+del\_acc\_id)*

*cursor.execute("Drop Table id\_"+str(del\_acc\_id)+"\_account")*

*myCon.commit()*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*top5.destroy()*

*def account\_entries\_fn():*

*global top7*

*global acc\_find\_entry*

*top7=Toplevel() #entering the values for credit and debit into one account ID*

*top7.title("Account Entries")*

*acc\_find=Label(top7,text="Enter the ID of the Account :").grid(row=0,column=0)*

*acc\_find\_entry=Entry(top7,fg="Blue")*

*acc\_find\_entry.grid(row=0,column=1)*

*acc\_find\_button=Button(top7,text="Enter to Search",command=find\_account)*

*acc\_find\_button.grid(row=0,column=2)*

*def find\_account():*

*global acc\_find\_val*

*acc\_find\_val\_inp=acc\_find\_entry.get()*

*if acc\_find\_val\_inp.isnumeric()== True: #checking if the ID entered is a number or not*

*acc\_find\_val=acc\_find\_val\_inp*

*else:*

*messagebox.showwarning("Warning","Sorry!It has to be a number!")*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*find\_record="SELECT EXISTS(SELECT \* from Accounts WHERE ID="+str(acc\_find\_val)+")"*

*cursor.execute(find\_record) #checking if the account exists*

*find\_record\_tupval=cursor.fetchone()*

*find\_record\_numval=find\_record\_tupval[0]*

*if find\_record\_numval==1: #if the account exists only then the next statements will pass*

*credit\_but=Button(top7,text="CREDIT",command=credit\_but\_fn,width=10)*

*credit\_but.grid(row=1,column=0,padx=10,pady=10,columnspan=2) #giving option to credit or debit*

*debit\_but=Button(top7,text="DEBIT",command=debit\_but\_fn,width=10)*

*debit\_but.grid(row=1,column=1,pady=10,columnspan=2)*

*else:*

*messagebox.showwarning("Warning","Sorry!This ID dos not exist!")*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*def credit\_but\_fn(): #enter the value in the credit field*

*global create\_TID*

*global create\_DOT*

*global create\_Credit*

*global create\_MOT*

*credit\_frame=LabelFrame(top7,padx=10,pady=10,relief= SUNKEN)*

*credit\_frame.grid(row=2,column=0,columnspan=3)*

*lbl1=Label(credit\_frame,text="Enter Transaction ID :").grid(row=0,column=0,padx=10,pady=(10,0))*

*lbl2=Label(credit\_frame,text="Enter Date of Transaction(yyyy-mm-dd) :").grid(row=1,column=0,padx=10)*

*lbl3=Label(credit\_frame,text="Enter Credit Amount :").grid(row=2,column=0,padx=10)*

*lbl4=Label(credit\_frame,text="Enter Any Note(related to the transaction) :").grid(row=3,column=0,padx=10)*

*create\_TID=Entry(credit\_frame,width=60,fg="Blue")*

*create\_TID.grid(row=0,column=1,pady=(10,0))*

*create\_DOT=Entry(credit\_frame,width=60,fg="Blue")*

*create\_DOT.grid(row=1,column=1)*

*create\_Credit=Entry(credit\_frame,width=60,fg="Blue")*

*create\_Credit.grid(row=2,column=1)*

*create\_MOT=Entry(credit\_frame,width=60,fg="Blue")*

*create\_MOT.grid(row=3,column=1)*

*credit\_submit\_but=Button(credit\_frame,text="Submit",command=credit\_submit)*

*credit\_submit\_but.grid(row=4,column=0,columnspan=2)*

*def credit\_submit():*

*l=create\_TID.get()*

*m=create\_DOT.get()*

*n=create\_Credit.get() #insert the values of credit into table specific to ID of the account*

*o=create\_MOT.get()*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*cursor.execute("INSERT into id\_"+str(acc\_find\_val)+"\_account values("+str(l)+",\""+str(m)+"\","+str(n)+",0,\""+str(o)+"\")")*

*myCon.commit()*

*lbl5=Label(top7,text="YOUR CHANGES HAVE BEEN REGISTERED!!",fg="Green",font=("Algerian",16)).grid(row=3,column=0,columnspan=3)*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*def debit\_but\_fn(): #enter the value in the debit field*

*global create\_TID*

*global create\_DOT*

*global create\_Debit*

*global create\_MOT*

*debit\_frame=LabelFrame(top7,padx=10,pady=10,relief= SUNKEN)*

*debit\_frame.grid(row=2,column=0,columnspan=3)*

*lbl1=Label(debit\_frame,text="Enter Transaction ID :").grid(row=0,column=0,padx=10,pady=(10,0))*

*lbl2=Label(debit\_frame,text="Enter Date of Transaction(yyyy-mm-dd) :").grid(row=1,column=0,padx=10)*

*lbl3=Label(debit\_frame,text="Enter Debit Amount :").grid(row=2,column=0,padx=10)*

*lbl4=Label(debit\_frame,text="Enter Any Note(related to the transaction) :").grid(row=3,column=0,padx=10)*

*create\_TID=Entry(debit\_frame,width=60,fg="Blue")*

*create\_TID.grid(row=0,column=1,pady=(10,0))*

*create\_DOT=Entry(debit\_frame,width=60,fg="Blue")*

*create\_DOT.grid(row=1,column=1)*

*create\_Debit=Entry(debit\_frame,width=60,fg="Blue")*

*create\_Debit.grid(row=2,column=1)*

*create\_MOT=Entry(debit\_frame,width=60,fg="Blue")*

*create\_MOT.grid(row=3,column=1)*

*debit\_submit\_but=Button(debit\_frame,text="Submit",command=debit\_submit)*

*debit\_submit\_but.grid(row=4,column=0,columnspan=2)*

*def debit\_submit():*

*l=create\_TID.get()*

*m=create\_DOT.get()*

*n=create\_Debit.get() #insert the values of debit into table specific to ID of the account*

*o=create\_MOT.get()*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*cursor.execute("INSERT into id\_"+str(acc\_find\_val)+"\_account values("+str(l)+",\""+str(m)+"\", 0 ,"+str(n)+",\""+str(o)+"\")")*

*lbl5=Label(top7,text="YOUR CHANGES HAVE BEEN REGISTERED!!",fg="Green",font=("Algerian",16)).grid(row=3,column=0,columnspan=3)*

*myCon.commit()*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*def balance\_sheet(): #dispaying all the credit and debit records of a specific ID*

*global top8*

*global bal\_find\_entry*

*top8=Toplevel()*

*top8.title("Balance Sheet")*

*bal\_find=Label(top8,text="Enter the ID of the Account :").grid(row=0,column=0,columnspan=2)*

*bal\_find\_entry=Entry(top8,fg="Blue")*

*bal\_find\_entry.grid(row=0,column=2)*

*bal\_find\_button=Button(top8,text="Enter to Search",command=bal\_find\_account)*

*bal\_find\_button.grid(row=0,column=3,columnspan=2)*

*def bal\_find\_account():*

*global bal\_record\_numval*

*global row\_count*

*global bal\_find\_val*

*global bals\_closing\_bal*

*global bals\_opening\_bal*

*bal\_find\_val\_inp=bal\_find\_entry.get()*

*if bal\_find\_val\_inp.isnumeric()== True: #check whether the input is number or not*

*bal\_find\_val=bal\_find\_val\_inp*

*else:*

*messagebox.showwarning("Warning","Sorry!It has to be a number!")*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor() #checking the existence of account*

*bal\_find\_record="SELECT EXISTS(SELECT \* from Accounts WHERE ID="+str(bal\_find\_val)+")"*

*cursor.execute(bal\_find\_record)*

*bal\_record\_tupval=cursor.fetchone()*

*bal\_record\_numval=bal\_record\_tupval[0]*

*if bal\_record\_numval==1: #if the account exists only then the next statements will pass*

*cursor.execute("SELECT \* from Accounts where ID="+str(bal\_find\_val))*

*bal\_record\_val=cursor.fetchone()*

*bals\_opening\_bal=bal\_record\_val[3]*

*cursor.execute("SELECT SUM(Credit) from id\_"+str(bal\_find\_val)+"\_account")*

*sum\_credit\_record=cursor.fetchone()*

*bals\_credit\_sum=sum\_credit\_record[0]*

*cursor.execute("SELECT SUM(Debit) from id\_"+str(bal\_find\_val)+"\_account")*

*sum\_debit\_record=cursor.fetchone()*

*bals\_debit\_sum=sum\_debit\_record[0]*

*cursor.execute("SELECT \* from id\_"+str(bal\_find\_val)+"\_account ORDER BY TID")*

*all\_bal\_val=cursor.fetchall()*

*row\_count=2*

*if len(all\_bal\_val)!=0: #check whether ifthe tupls has any value or not*

*bals\_closing\_bal=bals\_opening\_bal+bals\_debit\_sum-bals\_credit\_sum*

*lbl1=Label(top8,text="Transaction ID").grid(row=1,column=0,pady=10)*

*lbl2=Label(top8,text="Date OF Transaction").grid(row=1,column=1,padx=5,pady=10)*

*lbl3=Label(top8,text="Credit").grid(row=1,column=2,pady=10)*

*lbl4=Label(top8,text="Debit").grid(row=1,column=3,padx=5,pady=10)*

*lbl5=Label(top8,text="Note").grid(row=1,column=4,pady=10)*

*for row\_bal\_val in all\_bal\_val: #it is the single tuple field inside the tuple*

*lbl6=Label(top8,text=str(row\_bal\_val[0]),fg="Blue").grid(row=row\_count,column=0,pady=10)*

*lbl7=Label(top8,text=str(row\_bal\_val[1]),fg="Blue").grid(row=row\_count,column=1,padx=5,pady=10)*

*lbl8=Label(top8,text=str(row\_bal\_val[2]),fg="Brown",font=('calibri',10,'bold')).grid(row=row\_count,column=2,pady=10)*

*lbl9=Label(top8,text=str(row\_bal\_val[3]),fg="Darkolivegreen",font=('calibri',10,'bold')).grid(row=row\_count,column=3,padx=5,pady=10)*

*lbl10=Label(top8,text=str(row\_bal\_val[4]),fg="Blue").grid(row=row\_count,column=4,pady=10)*

*row\_count+=1*

*lbl11=Label(top8,text="OPENING BALANCE : "+str(bals\_opening\_bal),fg="Darkorange").grid(row=row\_count,column=0,columnspan=2)*

*lbl12=Label(top8,text="CLOSING BALANCE : "+str(bals\_closing\_bal),fg="Darkred").grid(row=row\_count,column=2,columnspan=3)*

*delete\_bal\_row\_but=Button(top8,text="Delete any Row!!",font=("Algerian",12),fg="Blue",command=delete\_bal\_row)*

*delete\_bal\_row\_but.grid(row=(row\_count+1),column=0,columnspan=2)*

*send\_user\_but=Button(top8,text="Email Account Holder the Receipt",font=("Algerian",12),fg="Blue",command=send\_user)*

*send\_user\_but.grid(row=(row\_count+1),column=2,columnspan=3)*

*else:*

*bals\_closing\_bal=bals\_opening\_bal*

*lbl14=Label(top8,text="IT HAS NO RECORD!!",fg="Green").grid(row=row\_count+2,column=0,columnspan=5)*

*lbl11=Label(top8,text="OPENING BALANCE : "+str(bals\_opening\_bal),fg="Darkorange").grid(row=row\_count,column=0,columnspan=2)*

*lbl12=Label(top8,text="CLOSING BALANCE : "+str(bals\_closing\_bal),fg="Darkred").grid(row=row\_count,column=2,columnspan=3)*

*send\_user\_but=Button(top8,text="Email Account Holder the Receipt",font=("Algerian",12),fg="Blue",command=send\_user)*

*send\_user\_but.grid(row=(row\_count+1),column=0,columnspan=5)*

*else:*

*messagebox.showwarning("Warning","Sorry!This ID dos not exist!")*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*def delete\_bal\_row(): #delete any row*

*global top9*

*global row\_tid\_no*

*top9=Toplevel()*

*top9.title("Delete Any Row")*

*lbl1=Label(top9,text="Enter the Transaction ID :").grid(row=0,column=0)*

*row\_tid\_no=Entry(top9,fg="Blue")*

*row\_tid\_no.grid(row=0,column=1)*

*delete\_row\_sub\_but=Button(top9,text="Submit",command=delete\_row\_sub)*

*delete\_row\_sub\_but.grid(row=1,column=0,columnspan=2)*

*def delete\_row\_sub():*

*row\_tid\_val=row\_tid\_no.get()*

*if row\_tid\_val.isnumeric()== True: #check if the Transaction ID is number*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*cursor.execute("Select \* from id\_"+str(bal\_find\_val)+"\_account where TID = "+str(row\_tid\_val))*

*del\_row\_val=cursor.fetchone()*

*if del\_row\_val is not None: #to check if te row exists or not*

*cursor.execute("DELETE from id\_"+str(bal\_find\_val)+"\_account where TID = "+str(row\_tid\_val))*

*myCon.commit()*

*lbl13=Label(top8,text="Please close the window to review changes!!",fg="Red",font=("Algerian",16)).grid(row=(row\_count+2),column=0,columnspan=5)*

*top9.destroy()*

*else:*

*messagebox.showwarning("Warning","Sorry!This Transaction ID does not exist!")*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*else:*

*messagebox.showwarning("Warning","Sorry!It has to be a number!")*

*def send\_user():*

*import os*

*import smtplib # imorting tp get values to my email*

*from email.message import EmailMessage*

*my\_email\_id=os.environ.get('MY\_EMAIL\_ID') #storing my personal data else where to encrypt my data*

*my\_email\_pass=os.environ.get('MY\_EMAIL\_PASSWORD')*

*try:*

*myCon = pymysql.connect( host='localhost', user='root', passwd='nipun',database=l1)*

*cursor = myCon.cursor()*

*cursor.execute("SELECT \* from Accounts where ID="+str(bal\_find\_val))*

*send\_email\_val=cursor.fetchone()*

*user\_acc\_id=send\_email\_val[0]*

*user\_acc\_name=send\_email\_val[1]*

*user\_acc\_email=send\_email\_val[4]*

*except Exception as e:*

*print("Exception : {}".format(e))*

*finally:*

*myCon.close()*

*user\_email\_msg=EmailMessage()*

*user\_email\_msg['Subject']='From the Accountant '+l1*

*user\_email\_msg['To']=user\_acc\_email*

*user\_email\_msg['From']=my\_email\_id*

*user\_email\_msg.set\_content("Account ID : "+str(user\_acc\_id)+"\n"+"Account Name : "+user\_acc\_name+"\n"+"Account Opening Balance : "+str(bals\_opening\_bal)+"\n"+"Account Closing Balance : "+str(bals\_closing\_bal))*

*with smtplib.SMTP\_SSL('smtp.gmail.com',465) as smtp:*

*smtp.login(my\_email\_id,my\_email\_pass) #login to my id*

*smtp.send\_message(user\_email\_msg)*

*messagebox.showinfo("Receipt","Your Receipt has been sent successfully!")*

*def help\_tally(): #help people understand the functions*

*top10=Toplevel()*

*top10.title("HELP")*

*help\_str='''1. Create Account button allows you to create an account for which*

*you have to make credit and debit entries to.*

*2. Acount Info gives you the information of the neccessary account*

*and allows you to make any changes.*

*3. Account Entries allow you to enter the desired values to credit*

*from or debit into account.*

*4. Balance sheet allows you to view entries made in the desired*

*acount and delete any rows.*

*5. Feedback allows you to give your insightful opinion to change and*

*make the interface better.*

*6. Exit allows you to quit the program'''*

*lbl1=Label(top10,text=help\_str,fg="Green",font=('Comic Sans MS',14)).grid(row=0,column=0,sticky=W)*

*lbl2=Label(top10,text="NOTE: Click any button only once to make the program work efficiently!!",fg="Red",font=('Algerian',15)).grid(row=1,column=0)*

*lbl3=Label(top10,text="THANK YOU!!",fg="Gold",font=('Algerian',16)).grid(row=2,column=0)*

*def feedback():*

*global top11*

*global rate\_val*

*global fdbck\_entry*

*top11=Toplevel()*

*top11.title("FEEDBACK")*

*exp\_lbl=Label(top11,text="HOW IS YOUR EXPERIENCE?").pack(anchor=W)*

*rate\_list=[("BAD","Bad"),("GOOD","good"),("VERY GOOD","Very Good")] #don't have to call radio button again*

*rate\_val = StringVar()*

*rate\_val.set("")*

*for rate,rate\_inp in rate\_list:*

*Radiobutton(top11, text=rate, variable=rate\_val, value=rate\_inp, fg="Blue").pack(anchor=W)*

*fdbck\_entry=Entry(top11,fg="Blue",width=200)*

*fdbck\_entry.pack(pady=10)*

*fdbck\_entry.insert(0,"PLEASE GIVE YOUR VALUABLE SUGGESTION")*

*fdbck\_submit\_but=Button(top11,text="SUBMIT",command=fdbck\_submit)*

*fdbck\_submit\_but.pack(padx=150,anchor=W)*

*def fdbck\_submit():*

*import os*

*import smtplib # imorting tp get values to my email*

*from email.message import EmailMessage*

*fdbck\_msg\_val=fdbck\_entry.get()*

*my\_email\_id=os.environ.get('MY\_EMAIL\_ID') #storing my personal data else where to encrypt my data*

*my\_email\_pass=os.environ.get('MY\_EMAIL\_PASSWORD')*

*fdbck\_msg=EmailMessage()*

*fdbck\_msg['Subject'] = "From the user "+l1 #content of the email*

*fdbck\_msg['From']=my\_email\_id*

*fdbck\_msg['To']=my\_email\_id*

*fdbck\_msg.set\_content("Rating App got was : "+rate\_val.get()+"\n"+"The Feedback was : "+fdbck\_msg\_val)*

*with smtplib.SMTP\_SSL('smtp.gmail.com',465) as smtp:*

*smtp.login(my\_email\_id,my\_email\_pass) #login to my id*

*smtp.send\_message(fdbck\_msg)*

*top11.destroy()*

*log\_in = Button(lsin\_frame, text="LOG IN",padx=52,pady=10,bg = "light blue",command=log)*

*log\_in.grid(row=1,column=0) #log in button in main window*

*sign\_in = Button(lsin\_frame, text="SIGN IN",padx=52,pady=10,bg = "light blue",command=sign)*

*sign\_in.grid(row=1,column=1) #sign in button in the main window*

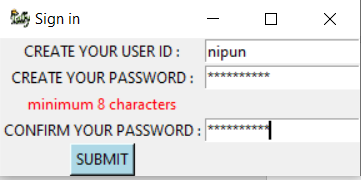
*master.mainloop() #so that loop runs infinitely and changes are read automatically*

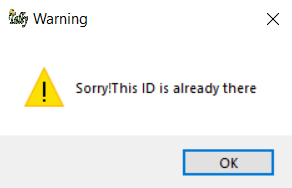
**6. OUTPUT**

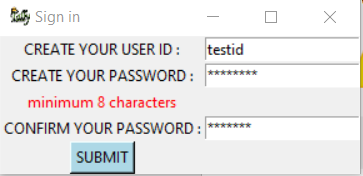
**CHOICE 1: RUNNING THE PROGRAM**

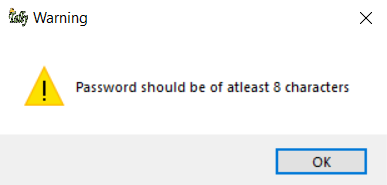
****

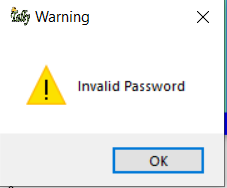
**CHOICE 2: SIGNING IN**

****

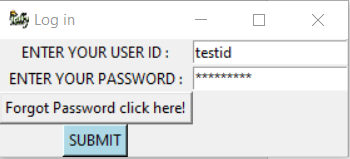


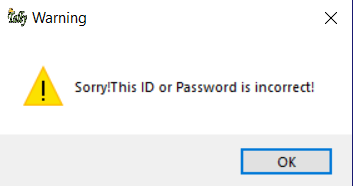


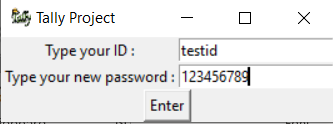


 #confirm and crate password are not equal

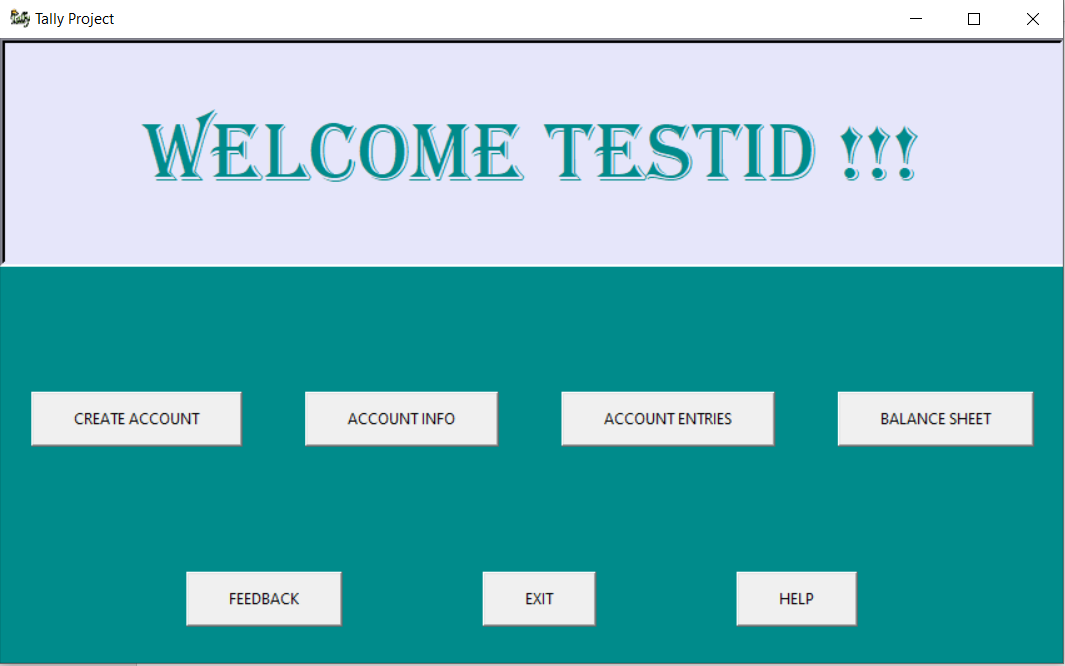
**CHOICE 3: LOGGING IN**

****

 # ID is correct but password is wrong

 #Changing the password from 12345678 to 123456789

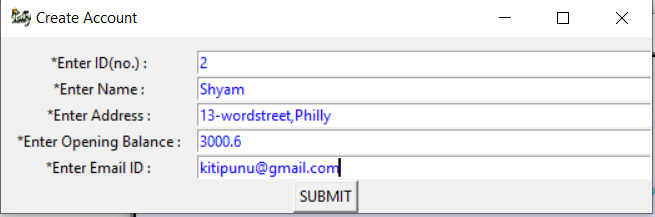
**CHOICE 4: SUCCESSFULLY LOGGING IN**



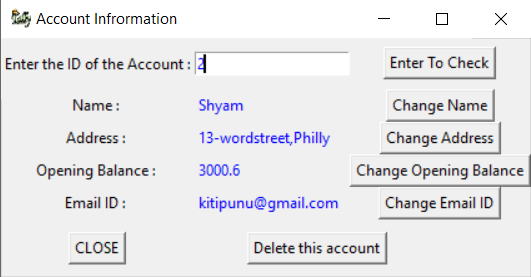
**CHOICE 5: HELP**

****

**CHOICE 6: CREATING CLIENT ACCOUNT**

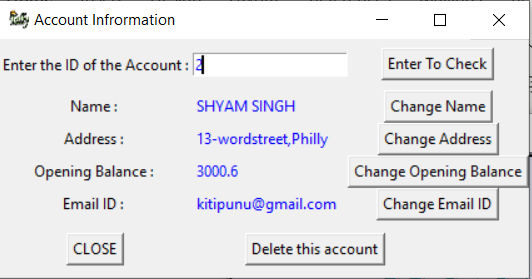
**** #my email id

**CHOICE 7: GETTING ACCOUNT INFORMATION**

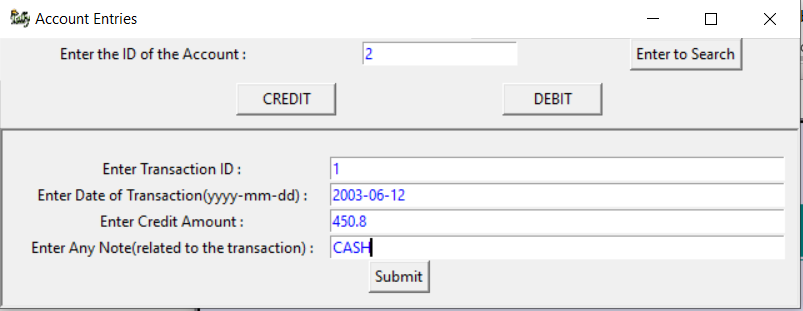
****

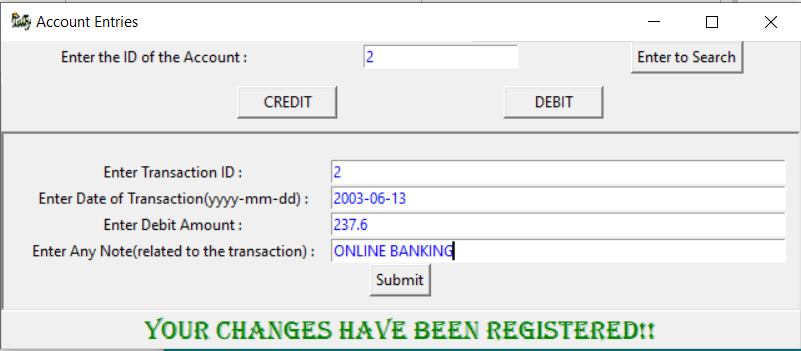
**CHOICE 8: CHANGE NAME**

****

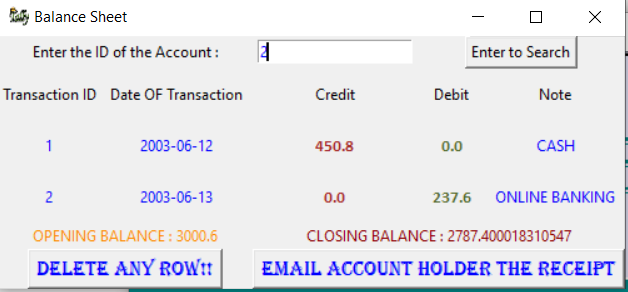


**CHOICE 9: MAKING TRANSACTIONS**

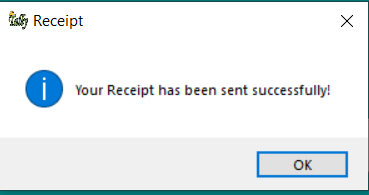
****

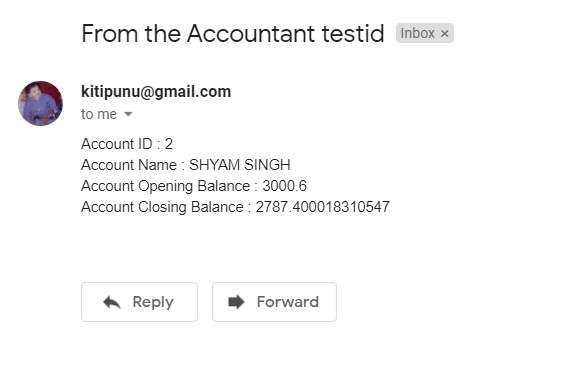


**CHOICE 10: REVIEWING THE BALACE SHEET**

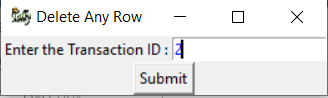
****

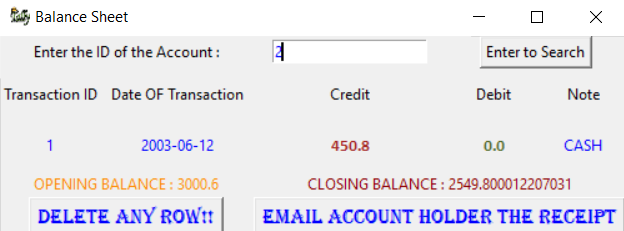
**CHOICE 11: EMAIL**

****

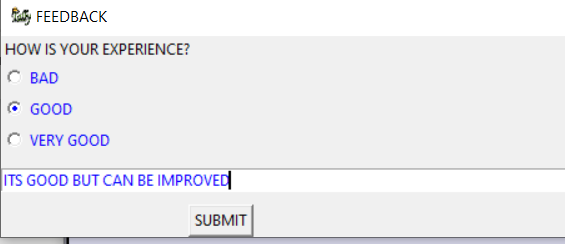
****#that is why I entered my email id

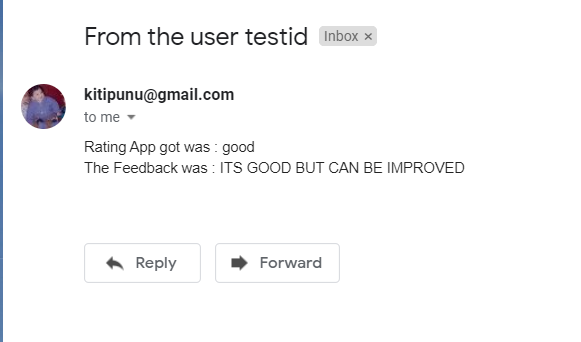
**CHOICE 12: DELETE ROW**

****

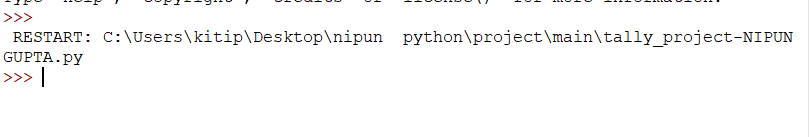


**CHOICE 13: FEEDBACK FROM THE USER**

****

****

**CHOICE 14: EXIT**

****#when program ends

**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 OS is desirable.
* NetBeans Ver 5.1 or higher should be installed with JDK and JVM.
* MySQL Ver 6.1 with Library Database must be present at machine.

### Database Installation-

The software project is distributed with a backup copy of a Database named class12 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 manav**.SQL** file which installs a database and tables in the computer system.

Note: The PC must have MySQL server with user (***root***) and password (h) . 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 “h” , this will change the root password.

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

**Step 1:** Copy the manav.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 class12.

mysql> create database class12;

**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*** -khn class12 <ais.sql

This will create a class12 database with required tables.

**8.References**

In order to work on this project titled – ***HOTEL MANAGEMENT SYSTEM,*** the following books and literature are referred by me during the various phases of development of the project:

(1) The Complete Reference Python 3.7

-by Shildit

1. MySQL, Black Book

-by Steven Holzner

(2) Understanding SQL

– Gruber

(3) <http://www.mysql.org/>

(4) <http://www.python.org>/

(5) On-line Help of Python ®

(6) Computer Science for class XII -by Sumita Arora

(7) Together with Computer Science

(8) Various Websites of Discussion Forum and software development activities.

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