##### **Grocery Store Management**

##### **Database Systems**

##### **CSCI-6622**

#### Team Data Dive



##### **University of New Haven**

##### **TAGLIATELA COLLEGE OF ENGINEERING,**

##### **West Haven, CT**

##### 

##### **Submitted To:**

##### **Prof Reza Sadeghi**

**Spring 2021**

**Table of Contents**

1. Team ………………………………………………………………………………....... 4

1.1 Roles of Team members……………………………………………………………. 4

1.2 Objective…………………………………………………………………………... 5

2. ER Diagram…………………………………………………………………………..... 5

2.1 List of Tables…………………………………………………………………….... 6

3. GUI…………………………………………………………………………………….. 8

3.1 Database connection to GUI……………………………………………………….. 8

* 1. Description of GUI………………………………………………………………… 8
  2. Explanation of GUI……………………………………………………………….... 9
     1. Main.py……………………………………………………………………… 9
     2. Employees.py………………………………………………………………. 11
     3. Manager.py…………………………………………………………………. 16

4. SQL……………………………………………………………………………………. 17

5. Git repository link …………………………………………………………………….. 19

6. References……………………………………………………………………………… 19

**Table of Figures**

Fig 1: ER Diagram ---------------------------------------------------------------------------------- 5

Fig 2: Describing the Python code for SQL connection and login screen-------------------- 9

Fig 3: Login Screen---------------------------------------------------------------------------------- 9

Fig 4: Python code for User validation----------------------------------------------------------- 10

Fig 5: User Validation ------------------------------------------------------------------------------ 10

Fig 6: Python code for Invalid password -------------------------------------------------------- 10

Fig 7: Invalid password ---------------------------------------------------------------------------- 11

Fig 8: Main Screen – Upon successful login ---------------------------------------------------- 11

Fig 9: Python Code to navigate to Employees Information Screen--------------------------- 12

Fig 10: Employees Information Screen----------------------------------------------------------- 12

Fig 11: Python code for Inserting data into MySQL database -------------------------------- 13

Fig 12: Error when user didn’t fill all the fields ------------------------------------------------- 13

Fig 13: Python code for search screen ------------------------------------------------------------ 14

Fig 14: earch screen----------------------------------------------------------------------------------- 14

Fig 15: Data Displayed on Search ------------------------------------------------------------------ 15

Fig 16: In case no data found ------------------------------------------------------------------------ 15

Fig 17: Select to edit the data – Options to Update and Delete ---------------------------------16

Fig 18 : Python code for manager screen ----------------------------------------------------------17

Fig 19 GUI for Managaer Information --------------------------------------------------------------17

Fig 20 : MySQL Queries for database creation ----------------------------------------------------18

**Final Project Report of Data Dive**

**1 Team Data Dive**

**Team Members**

|  |  |
| --- | --- |
| Name | Email |
| Madhavi Kuchipudi | mkuch2@unh.newhaven.edu |
| Charitha Sree Jayaramireddy | cjaya1@unh.newhaven.edu |
| Ganesh Putta | gputt1@unh.newhaven.edu |
| Pranitha Challa | pchal4@unh.newhaven.edu |
| Monika Veerati | mveer3@unh.newhaven.edu |

* 1. **Roles of Team Members**

All the team members have been contributed equally for the project. We have divided the project into following five modules:

Research, Design, Coding, Testing, Documentation.

In each phase of the project team members have been actively participated and taken inputs from each other and implemented together to a successful outcome of the project. In each phase, team came across few challenges while working with Python code and solved them by interacting in group calls outside the class time. Each and every team member had their 100 % contribution in our project to deliver a successful outcome of out project Grocery Management System.

* 1. **Objective**

The goal of our project is to design a GUI for Grocery store Management System which incorporates details of the Employees, Manager, Designation of the employees, categories of the products, details of the Customer, list of available commodities and location information of the grocery stores. Suppliers and details of commodities which shows which items are going to be out of stock for the store which has various branches situated at various areas with different Managers taking care of that data set.

This database is efficacious in running the grocery stores. The users of the database will be the store managers.

1. **ER Diagram**

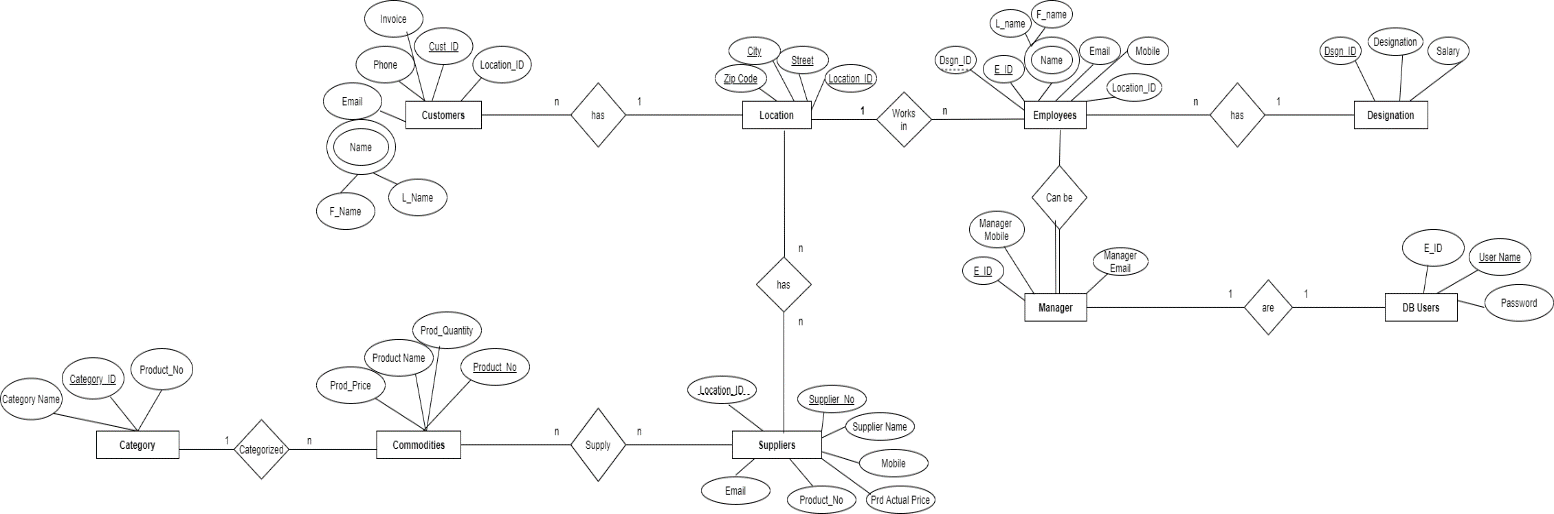
****

Fig 1: ER Diagram



* 1. **List of tables**

In Grocery store management, we have the following tables which will store the corresponding data-

1. **Location**

The Location table has records about the location information of the grocery stores. For now, we have defined grocery stores in 20 locations.

A unique Location\_ID is defined for all the locations and this acts as primary key for this table. Other information includes City, Street, Zip Code of the particular location. Only the city column can have null values stored in them and we have every other column defined as not null (A value must exist for these columns).

1. **Employees**

Employees table has records of the Employees working in the grocery store. As we have defined our model to have managers for each store and they exclusively have the access to the data base of the grocery store (DB Users), for now, we have defined all the managers for all the grocery stores listed in the Location table.

This table gets information of Location\_Id from Location table and Dsgn\_Id from Designation table. Employees table has information about First Name, Last Name, Unique Employee Id, Employee Mobile number, Employee email address and the designation Id of the employee.

E\_Id (Employee\_Id) is defined as the primary key of this table. We have also defined Location\_Id and Dsgn\_Id as foreign keys with references from Location and Designation tables respectively. Every column in Employees table is defined as not null, meaning, every column must have some value, if defined.

1. **Designation**

The Designation table has various records of different designations applicable/available in the grocery store. We have defined some designations like Level\* and Staff\* which we are not using now, but might be using in the future.

Dsgn\_Id and Designation are defined as not null, as they are the primary information in this table. Storing salary for any designation is optional. Dsgn\_Id is defined as the primary key.

1. **Customers**

The Customers table holds the details of the customer such as the Cust\_Id which is the Primary key and other details such as Fname, Lname and Phone. This table provides details of the Customers with reference to the Location\_Id from Location table. Hence, it is considered as the foreign key.

1. **Manager**

The Manager table holds the details the employee such as the E\_Id which is the primary key, mobile and Email. Managers are the DB users in our store. As the E\_Id is taken reference from Employee table it is considered as the foreign key.

1. **DB Users**

DB Users are the Managers and they need the E\_Id, Username and Password to login to Database. Username and E\_Id can be referred from the Manager table. E\_Id is considered as the foreign key.

1. **Commodities**

The Commodities table holds the details of various products such as the Product\_No, Product\_Name, Prod\_Quantity and Prod\_Price. Product\_No is considered as the Primary key of Commodities table.

1. **Suppliers**

The Suppliers table holds the details of the product suppliers such as the Supplier\_No, Supplier\_Name, Mobile, Location\_Id, Email, Product\_No and Prd\_Actual\_Price. Supplier\_No is considered as the Primary key where as Product\_No is the foreign key which is referred from the Commodities table and Location\_Id is the foreign key which is referred from the Location table.

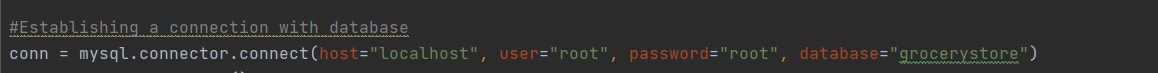
1. **Categories**

The Categories table holds the details of the Product categories such as the Category\_Id, Category\_Name and Product\_No which is referred from the Commodities table is considered as the foreign key.

1. **GUI**

The GUI for our project, Grocery Store Management system is built using Python Tkinter. We have created GUI for all our tables, where we can perform operations, such as INSERT, UPDATE, DELETE, SEARCH on all the tables of our Grocery Store database. Elements such as Window, Labels, Buttons, Treeview, Dropdown, Images are used in building the GUI.

**3.1 Database connection to GUI**

In order to establish connection to database and GUI the following syntax needs to be used. 

Here, I have used the credentials of my localhost database connection, however, one has to replace it with their own credentials in their system. By using this, the user can connect to database and a GUI will be displayed accordingly.

**3.2 Description of GUI**

The main.py file has to be run, in which all the other modules are imported. Each module is for each table in our database. Main.py holds everything and it directs us to different modules with the help of buttons. When we run “main.py”, a login window, where we have to enter the credentials will be displayed and will only take you inside, if you enter valid credentials. If the credentials are correct, then a window is displayed with image buttons for all our tables.

When we click on each button, it will be redirected to the respective module and connection is established with the database. In order to establish a successful connection with the database “grocerystore”, it should be available in our system.

**3.4 Explanation of GUI**

**3.3.1** **Main.py**

Main.py file is executed. It has all modules imported into it. It is configured in such a way that only managers can access the database and credentials are provided only to them.

It displays the login screen as shown below.



Fig 2: Describing the Python code for SQL connection and login screen

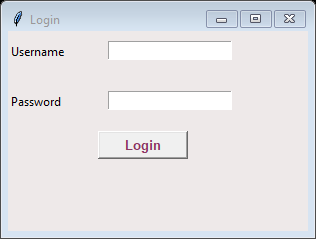


Fig 3 : Login Screen

In case of invalid username, an error will be popped up.

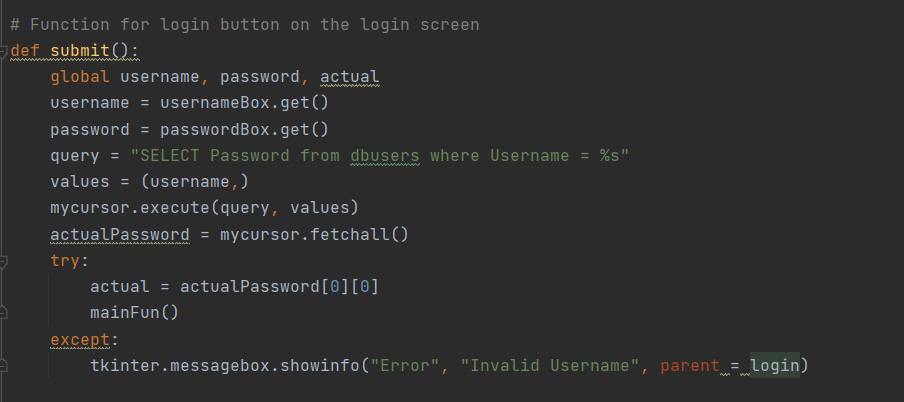


Fig 4 : Python code for User validation

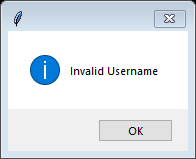


Fig 5 : Invalid Username

In case of wrong password, an error like below will be popped up.

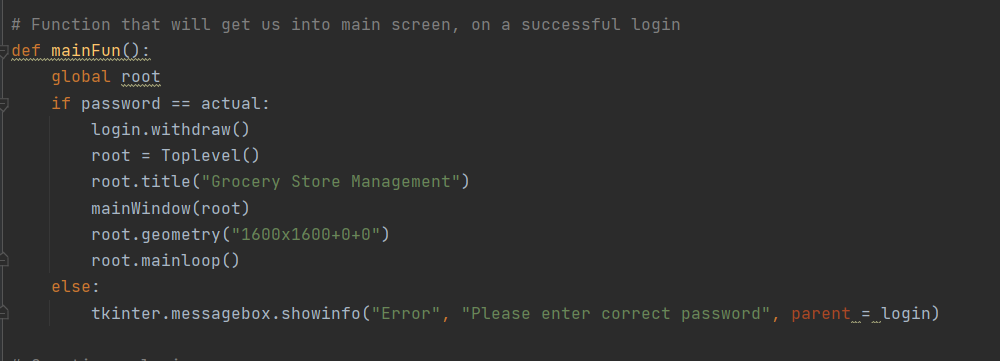


Fig 6 : Python code for Invalid Password

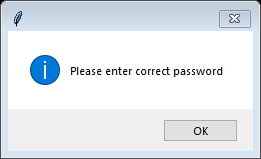


Fig 7 : Invalid Password

If valid credentials are entered, the main screen with access to all our database will be displayed.



Fig 8 : Main Screen – Upon successful login

The images displayed on the screen are buttons and are clickable. Corresponding text is displayed on each button.

Every subpage has the same view, but with the different content. Let us check one sub module, to know how exactly does our GUI looks and how we can perform operations such as INSERT, DELETE, UPDATE and SEARCH.

We have also added menu to the screen – About Us, Contact Us, Help/Exit which has options to close the screen, and our emails to reach out to us.

* + 1. **Employees.py**

Click on Employees Button. It will be redirected to the employees information page in the same window.

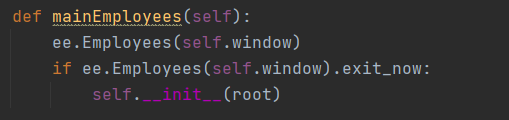


Fig 9 : Python Code to navigate to Employees Information Screen



Fig 10 : Employees Information Screen

Text is displayed using Labels and to input data, textboxes are used.

The above window has four options in it –

1. **Add Employee:**

By using add Employee we can insert data into our table. If no information is filled and clicked on add employee which is shown in below screenshot, a message box will be displayed as shown below asking to fill all the fields. A function is written in a way that it inserts data into our table taking all the information from the GUI.

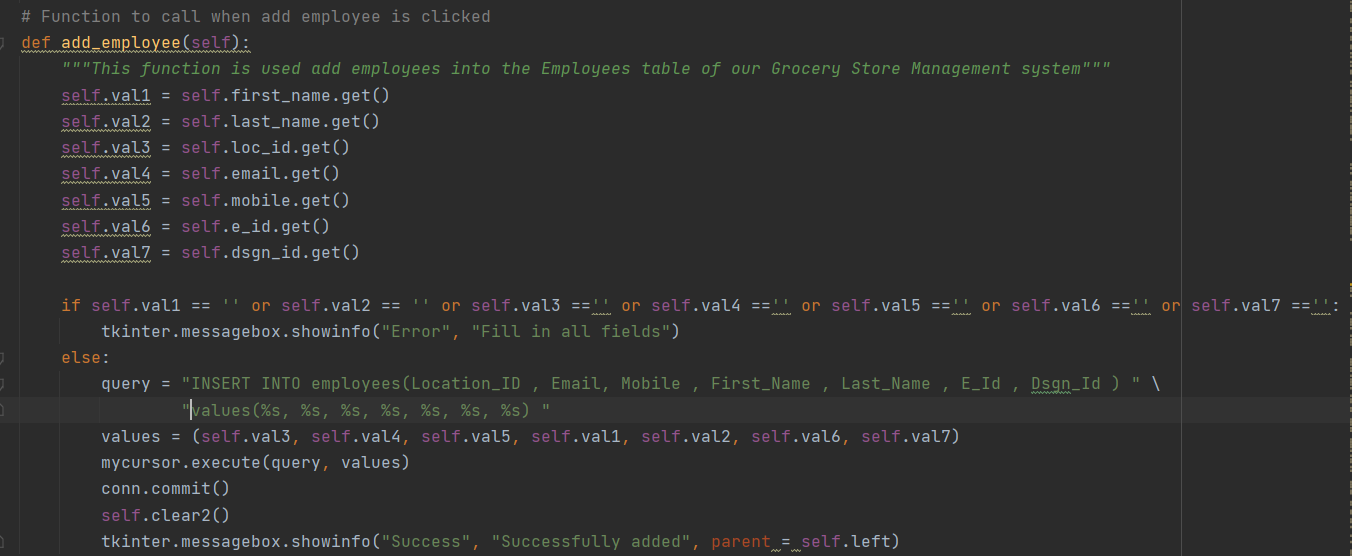


Fig 11 : Python code for Inserting data into MySQL database

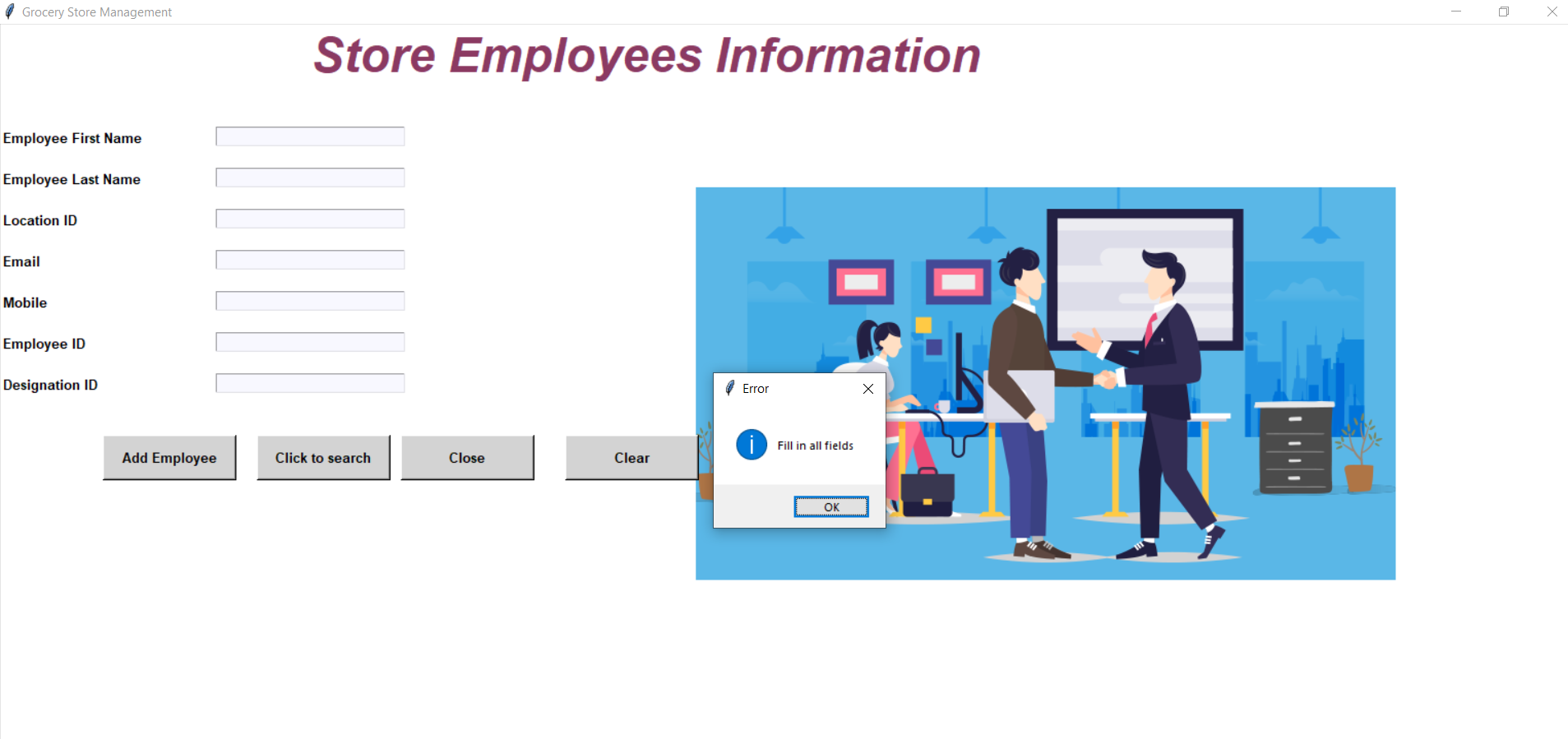


Fig12 : Error when user didn’t fill in all fields

1. **Close:**

This will navigate us to initial main window.

1. **Clear:**

This will clear if any data filled in the text box and refreshes the window.

1. **Click to search:**

For searching employees, we need to select option in database. This will open a new window upon clicking on it and will provide various options, which will be discussed later.

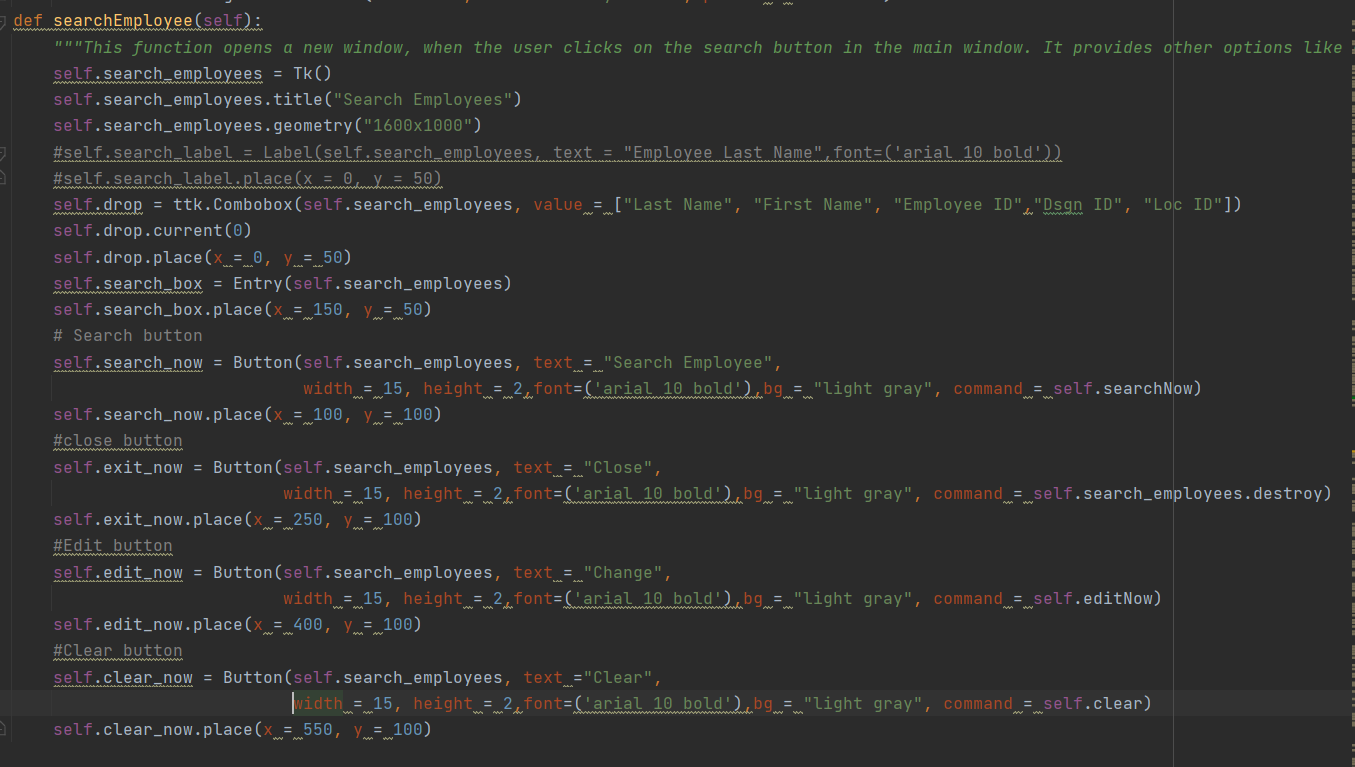


Fig 13 : Python code for search screen

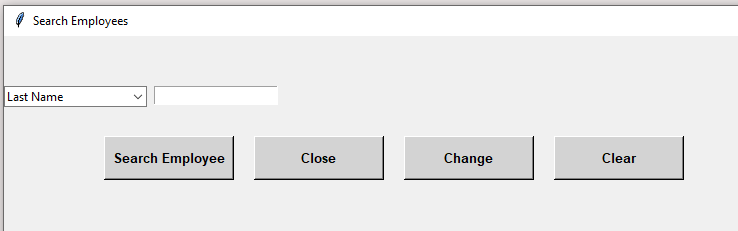


Fig 14 : Search Screen

There is dropdown with the column names present in tables of our database. We can search with any option available in the dropdown.

Options available in search window are as below:

1. **Search Employee:**

This is used to search for an employee.

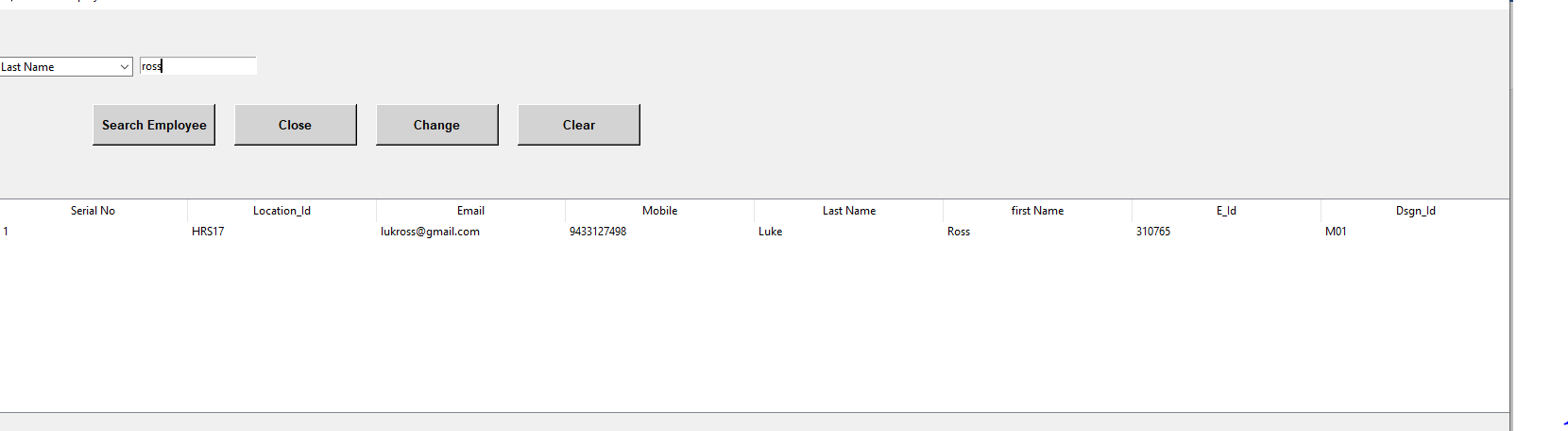


Fig 15 : Data Displayed on Search

Output is displayed with Treeview structure. If there is no EE, then a message will be displayed as “Employee Not Found” as shown below.

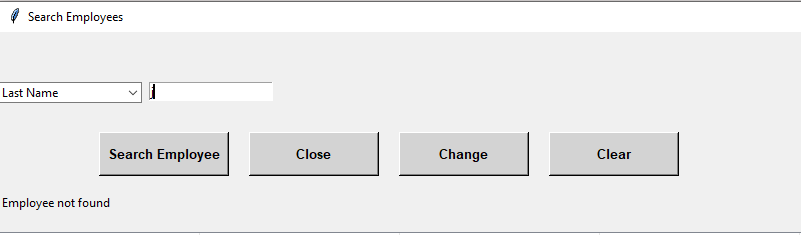


Fig 16 : In case no data found

1. **Close:**

This close is used to close the search window.

1. **Clear:**

This Clear option is used to refresh the window.

1. **Change:**

This change option is used to update or delete employee. When we click on change, the GUI asks us to select the employee first. When an employee is selected and clicked on the change, the options to update and delete will be visible as shown below.

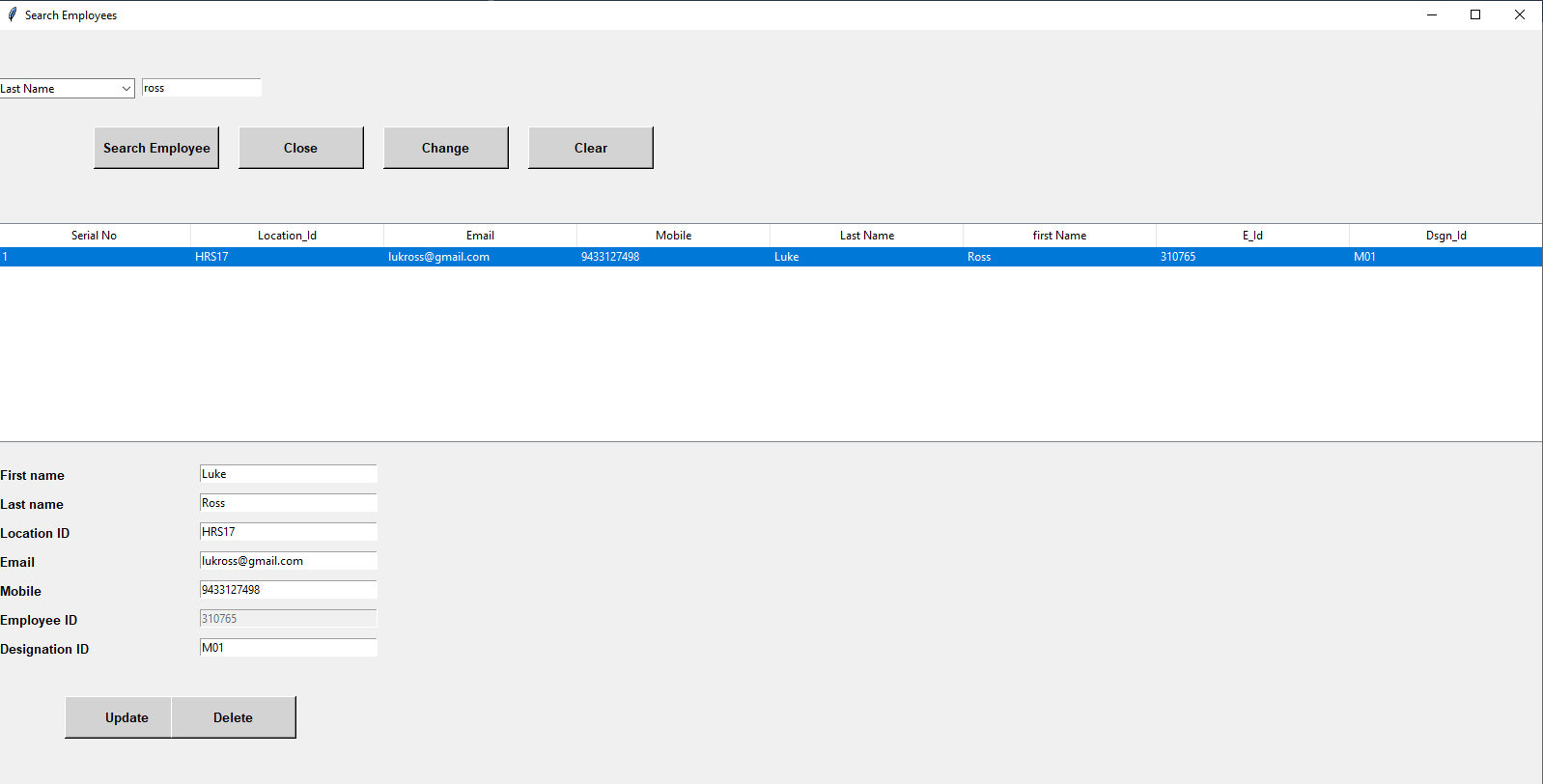


Fig 17 : Select to edit the data – Options to Update and Delete

As Employee Id is the primary key and it cannot be changed, EE ID is grayed out and all the information of the selected item will be populated in the respective fields. Now, there are two options, update and delete. To perform each operation, a separate function is created. If the operation is successfully performed, a pop up with message is displayed showing that the particular operation is successful.

E.g.: “Successfully Updated”.

**3.3.3 Manager.py:**

It has options only to search, clear and close. The button which says Branch – Manager gives us the view with details of manager and the concerned branch address.

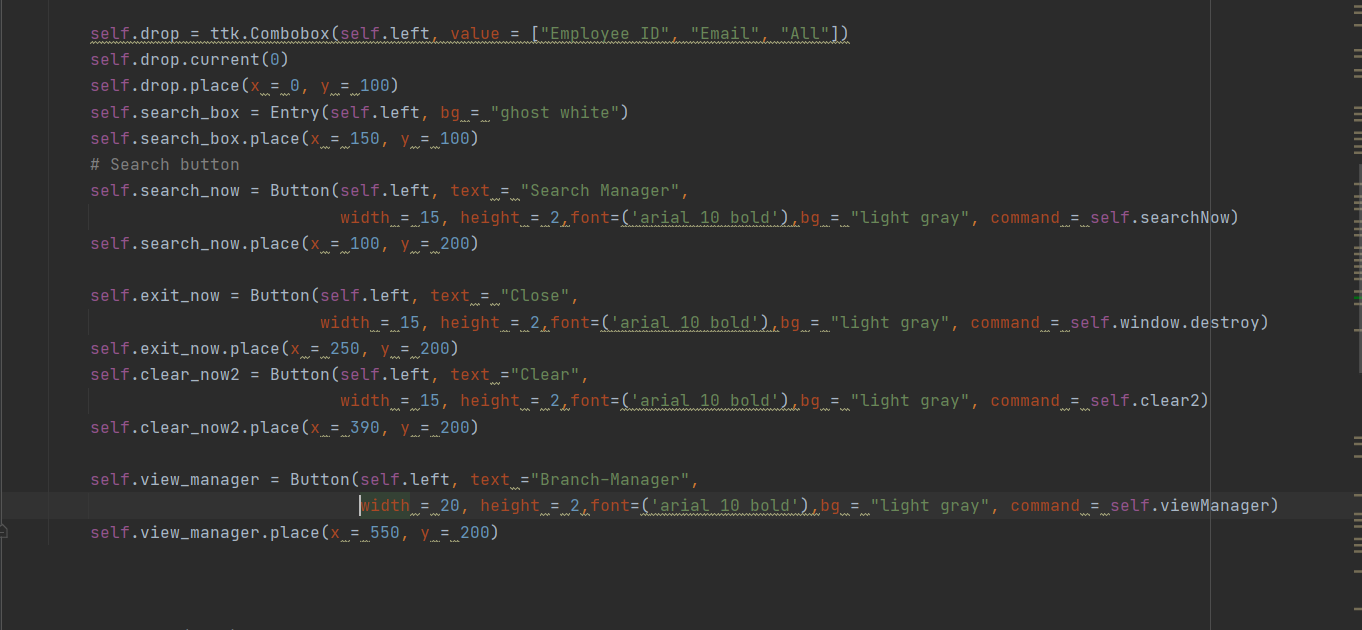


Fig 18 : Python code for manager screen

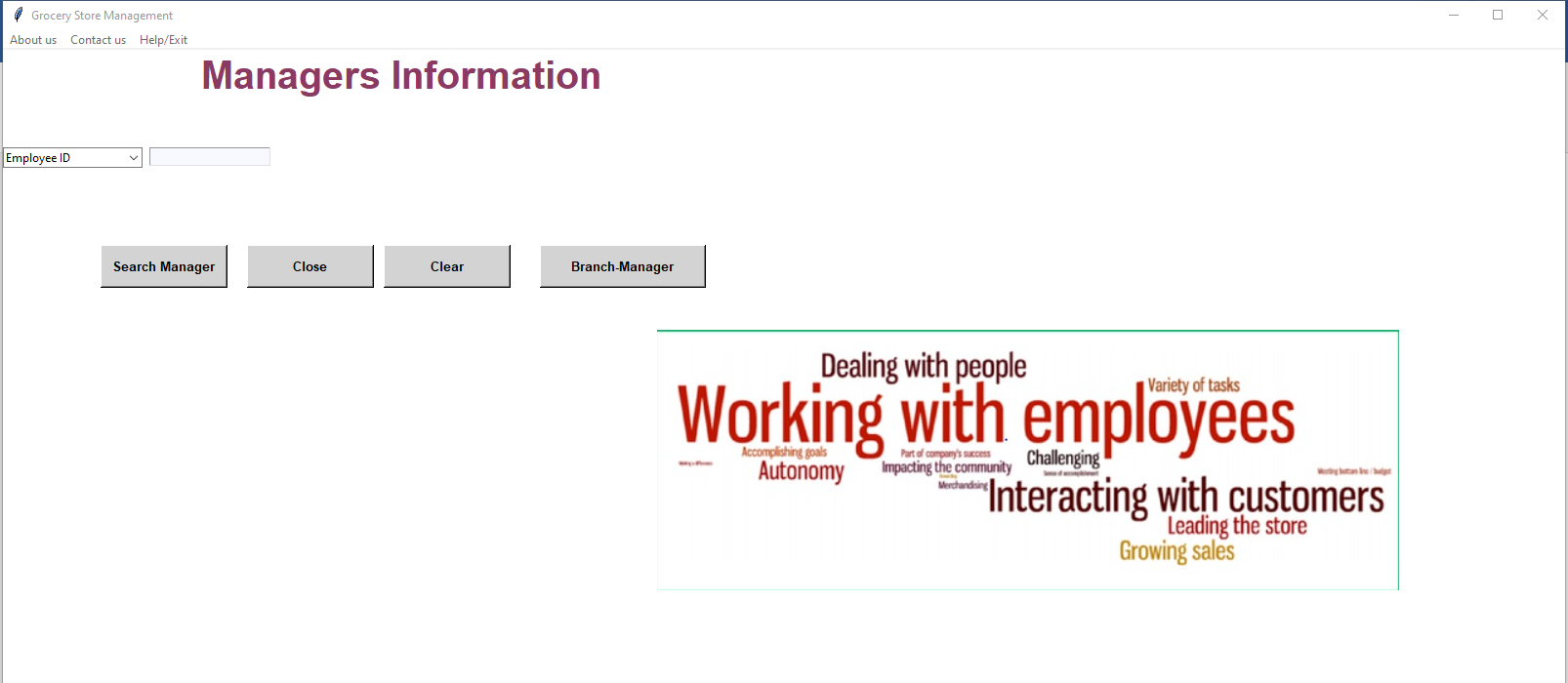
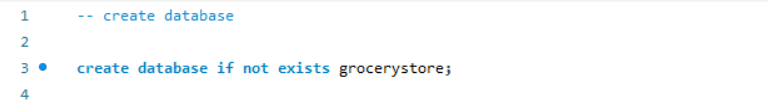


Fig 19 : GUI for Managaer Information

The GUI is same for all other tables too, with appropriate information.

**4. SQL**

We have used the database queries effectively and carefully to implement the insert, update, delete and search. We have also used a view to join out tables and view the records. Database with the name grocerystore has been created.



By using the database, we have created the following tables:

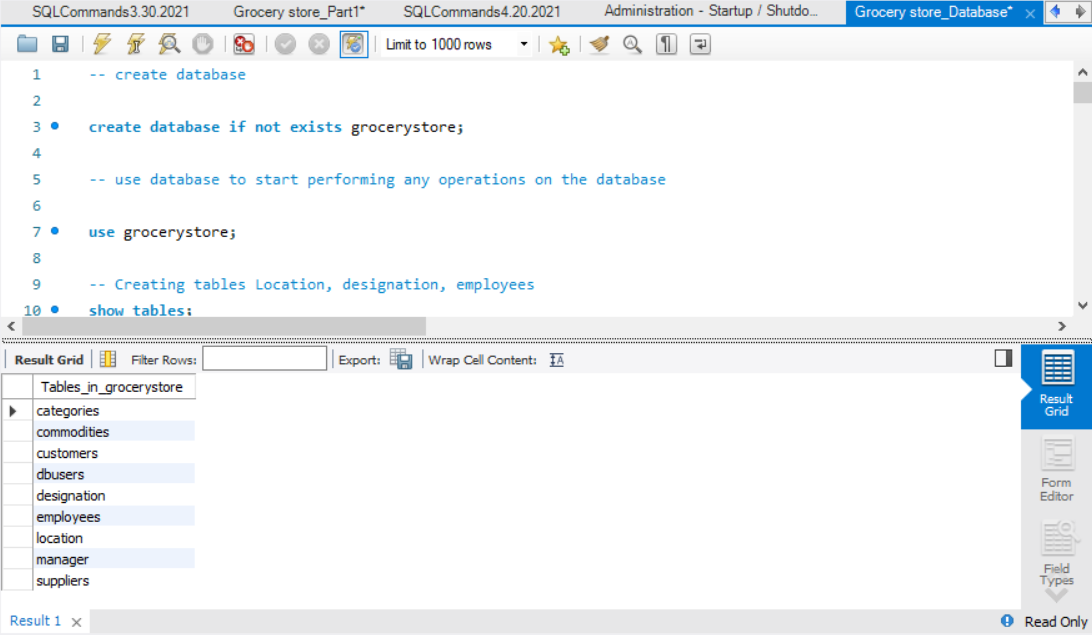


Fig 20 : MySQL Queries for database creation

Data has been inserted into the tables and modified accordingly.

1. **Git Repository**

Once we are done with the project, we pushed the code to Github repository and the link is been provided to the team member and the professor to access the code files and work on it if any further changes are required.

**Git Repository Link:**

You can access the code files in the below link.

<https://github.com/Charitha13/Grocery_Management_System>

1. **References**

<https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/ttk-Treeview.html>

<https://www.tutorialspoint.com/python/python_gui_programming.htm>

<https://www.w3schools.com/sql/default.asp>

<https://codemy.com/intro-tkinter-python-gui-apps/>