Stock and Billing Management with Multiple Logins using Hashing

# Abstract

# **TITLE:** Stock and Billing Management with Multiple Logins using Hashing

## **AIM of Project:**

To create an application that has the capability of displaying the Stock of the items, add, update and delete the items. Admin can add user accounts and can provide the access to his staff. Admin can also update and delete the user accounts he created. He can even make another account as admin.

We have also provided a feature to calculate his daily income based on the items he sold on that day. So that he can aware of business going into profit or loss. The user can refresh the stock every time if necessary.

We implemented the features like printing a bill, saving the bill and adding items to the bill in the billing section. This makes the user provide the bills to the customers and can also save a bill for his future use. User can also check expiry of an item by clicking on the Expiry option of the Check Expiry Menu on the Menu bar.

We have implemented the Industry level standards in saving the passwords by using the Hashing techniques using ‘Salt’ mechanism of the Cryptography. Salt is a random data used as an additional input to a 1-way hashing technique. We have used “Bcrypt” module for hashing the passwords and storing them in the database. As a result the passwords are highly secured. We used Mysql database as it is the fastest database that also allows the user to create multiple databases and can handle huge data without losing the performance.

Whenever the user adds, updates or deletes his account an email is sent in the background to the user’s email id. We used SMTPLIB module for the sending of mails in the HTML format. Admin can change some of the parameters regarding the project from the Config file provided, like, database connection IpAddress , port number etc.

We have used the latest trending programming language, python for creating the entire project. Tkinter and PILLOW for the Graphical User Interface. PyMysqldb module for communicating with the database. Bcrypt for hashing the passwords. We also used the SQL schema file for ‘Database Scripting’. This allows the user to avoid the usage of command line shells for interacting with database directly.

All the modules used are open source and can be found in the python installation of packages (PIP). The entire project follows the Industry standards and is developed in “Pycharm” IDE.

## **Future Scope:**

We can improve the security by adding ‘Role-based-permission’ model to the project which helps the admin in providing his own permissions for his staff. He can create, update and delete the permissions and can assign them to other accounts. We can optimize the code for making the GUI respond faster and smoother. We can also use threading concepts for background execution of tasks.

**Technologies used:** Python.

**Front end technology:** Tkinter

**Modules Used:** Pillow, PyMysqldb, bcrypt, smtplib, configparser

**Database used:** MySQL.

# Why Python?

Python is one among the most popular dynamic programming languages that is being used today. Python is an open-source and object-oriented programming language developed by Dutchman Guido van Possum in 1980s. It’s the fourth most popular language according to an [IEEE survey](http://relus.com/top-10-programming-languages-to-learn-in-2016/), behind old classics Java, C, and C++. This language can be utilized for a wide range of applications like scripting, developing and testing.  Due to its elegance and simplicity, top technology organizations like Dropbox, Google, Quora, Mozilla, Hewlett-Packard, Qualcomm, IBM, and Cisco have implemented Python.

Several websites state that Python is one among the most famous programming language of 2016. Because of its implementation and syntax, it pressures more on code readability. When compared to other programming languages like C++ and Java, it requires the programmer to develop lesser codes. It offers automatic memory management and several standard libraries for the programmer.

The following are some of the useful features of this language:

* It uses the **elegant syntax**; hence, the programs are easier to read.
* It is a **simple** to access language, which makes it easy to achieve the program working. This feature makes Pythons as the ideal programming language for prototype development as well as other ad-hoc programming functions. There is no need to compromise maintainability.
* The **large standard library** included in this language supports several programming tasks like searching text along with systematic expressions, linking to Web servers, modifying and reading files.
* The **interactive mode** of Python makes its simple to test code’s short snippets. It also comprises the bundles development environment known as IDLE.
* It is also simple to extend the code by appending new modules that are implemented in other compiled language like C++ or C.
* An expressive language which is possible to **embed** into applications to offer a programmable interface.
* Allows developer to run the code anywhere, including Windows, Mac OS X, UNIX, and Linux.
* It is free software in a couple of categories. It does not cost anything to use or download Pythons or to add it in the application. It can also be redistributable and modifiable without any cost; since, the coding language is copyrighted and it is available under the open source license.

## 1. Big Name Companies use Python

Python is already used by some of the biggest names in tech, along with some less likely but equally-impressive users.

Uber, PayPal, Google, Facebook, Instagram, Netflix, Dropbox, and Reddit all use Python in their development and testing. Moreover, Python is also used extensively in robotics and embedded systems (it can even be [used to control Arduinos](https://www.makeuseof.com/tag/program-control-arduino-python/)). Even legacy systems written in C and C++ are easy to interface with Python.

Anywhere that data analysis is required, Python and its assorted libraries shine. Goldman Sachs is one of several large financial institutions using Python to express the massive amounts of data they generate.

## 2. Machine Learning With Python

Machine learning as a skill is in greater demand every day. A good grasp of the Python programming language puts you a step ahead of others learning it from scratch.

[Google’s TensorFlow](https://www.makeuseof.com/tag/google-tensorflow-examples/) works primarily with Python. Almost every course on neural networks uses Python. The data analysis and parsing required for machine learning go well with Python, and its libraries.

## 3. Allow Developers to build applications in

Of course, Python was inspired from traditional programming languages like Java and C++. Hence, there is no wonder in several similar features present in the Python as that in Java or C++. Python includes the capability to set up any kind of application as like Java. That means, the Python developer can able to build a web application, desktop software, hardware program and smartphone games. It enables its programmers not to bind their skills in any particular domain. The Python programs can build any kind of applications irrespective of device, domain, and platform.

## 4. Fully Cross Platform

The [Dropbox desktop client](https://talkpython.fm/episodes/transcript/30/python-community-and-python-at-dropbox) is written entirely in Python, which speaks to its cross-platform compatibility. Dropbox has about [400 million](http://techcrunch.com/2015/06/24/dropbox-hits-400-million-registered-users/) users and considering it isn’t bundled with any operating system distribution, that’s a lot of users downloading and installing Dropbox. In addition to their desktop client, Dropbox’s server-side code is in Python as well, making it the majority language used at the company.

# Unified Modeling Language (UML)

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

UML is a pictorial language used to make software blueprints.

UML can be described as a general purpose visual modeling language to visualize, specify, construct, and document software system.

Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design.

## UML Relationships

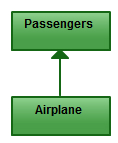
Classes are interrelated to each other in specific ways. In particular, relationships in class diagrams include different types of logical connections. The following are such types of logical connections that are possible in UML:

* [Association](https://creately.com/blog/diagrams/class-diagram-relationships/#Association)
* [Multiplicity](https://creately.com/blog/diagrams/class-diagram-relationships/#Multiplicity)
* [Aggregation](https://creately.com/blog/diagrams/class-diagram-relationships/#Aggregation)
* [Composition](https://creately.com/blog/diagrams/class-diagram-relationships/#Composition)
* [Inheritance/Generalization](https://creately.com/blog/diagrams/class-diagram-relationships/#Inheritance)
* [Realization](https://creately.com/blog/diagrams/class-diagram-relationships/#Realization)

### Association

Association is a broad term that encompasses just about any logical connection or relationship between classes. For example, passenger and airline may be linked as above:

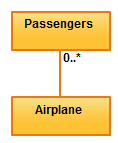
#### Directed Association

[](https://d3n817fwly711g.cloudfront.net/blog/wp-content/uploads/2012/03/Directed-Association-Relationship.jpeg)

*Directed Association*

Directed Association refers to a directional relationship represented by a line with an arrowhead. The arrowhead depicts a container-contained directional flow.

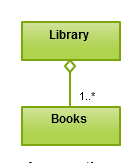
#### Multiplicity

[](https://d3n817fwly711g.cloudfront.net/blog/wp-content/uploads/2012/03/Multiplicity-Relationship.jpeg)

*Multiplicity*

Multiplicity is the active logical association when the cardinality of a class in relation to another is being depicted. For example, one fleet may include multiple airplanes, while one commercial airplane may contain zero to many passengers. The notation 0..\* in the diagram means “zero to many”.

#### Aggregation

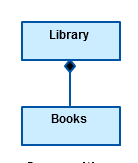
[](https://d3n817fwly711g.cloudfront.net/blog/wp-content/uploads/2012/03/Aggregation-Relationship.png)

*Aggregation*

Aggregation refers to the formation of a particular class as a result of one class being aggregated or built as a collection. For example, the class “library” is made up of one or more books, among other materials. In aggregation, the contained classes are not strongly dependent on the lifecycle of the container. In the same example, books will remain so even when the library is dissolved. To show aggregation in a diagram, draw a line from the parent class to the child class with a diamond shape near the parent class.

To show aggregation in a diagram, draw a line from the parent class to the child class with a diamond shape near the parent class.

#### Composition

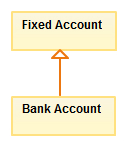
[](https://d3n817fwly711g.cloudfront.net/blog/wp-content/uploads/2012/03/Composition-Relationship-UML.png)

*Composition*

The composition relationship is very similar to the aggregation relationship. With the only difference being its key purpose of emphasizing the dependence of the contained class to the life cycle of the container class. That is, the contained class will be obliterated when the container class is destroyed. For example, a shoulder bag’s side pocket will also cease to exist once the shoulder bag is destroyed.

To show a composition relationship in a UML diagram, use a directional line connecting the two classes, with a filled diamond shape adjacent to the container class and the directional arrow to the contained class.

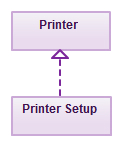
### Inheritance/Generalization

[](https://d3n817fwly711g.cloudfront.net/blog/wp-content/uploads/2012/03/Inheritance-Relationship.jpeg)

*Inheritance*

Inheritance refers to a type of relationship wherein one associated class is a child of another by virtue of assuming the same functionalities of the parent class. In other words, the child class is a specific type of the parent class. To show inheritance in a UML diagram, a solid line from the child class to the parent class is drawn using an unfilled arrowhead.

#### Realization

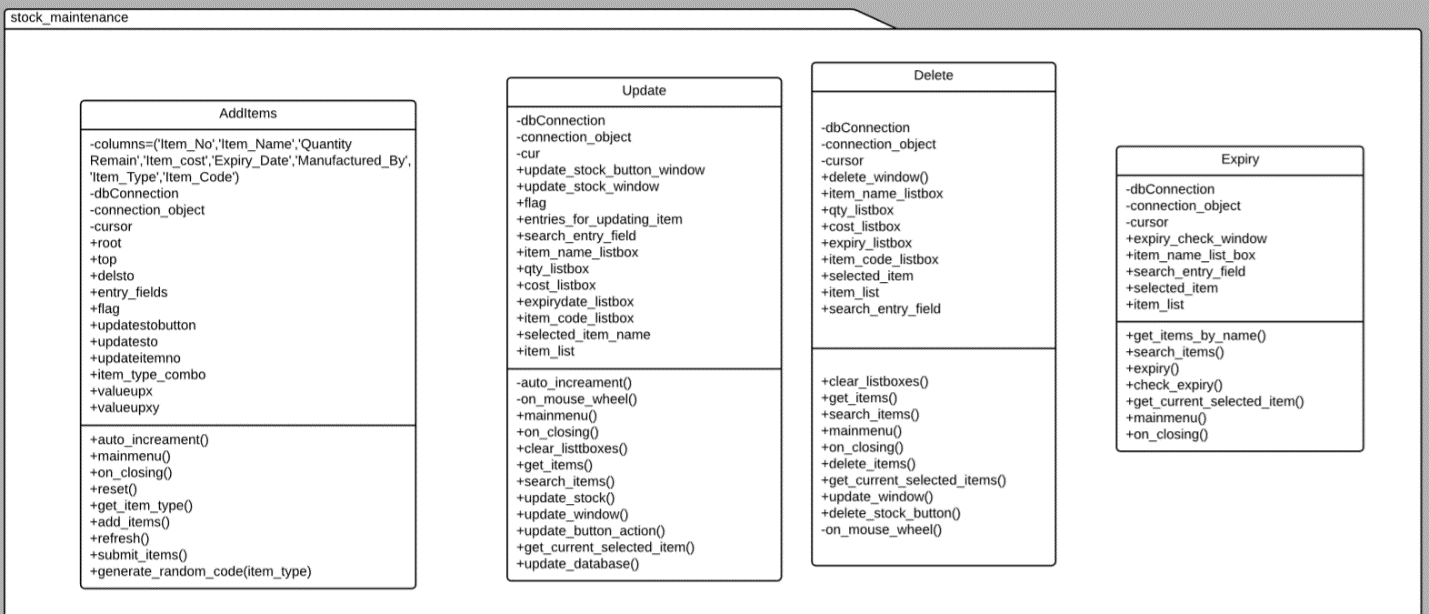
[](https://d3n817fwly711g.cloudfront.net/blog/wp-content/uploads/2012/03/Realization-Relationship.jpeg)

*Realization*

Realization denotes the implementation of the functionality defined in one class by another class. To show the relationship in UML, a broken line with an unfilled solid arrowhead is drawn from the class that defines the functionality of the class that implements the function. In the example, the printing preferences that are set using the printer setup interface are being implemented by the printer.

## UML Diagrams in the Project

Please go through the “UML.pdf” document for getting the full relationship diagram of the project.



Stock Maintenance is a package that contains the classes “AddItems”, ”Update”, “Delete”, “Expiry”.

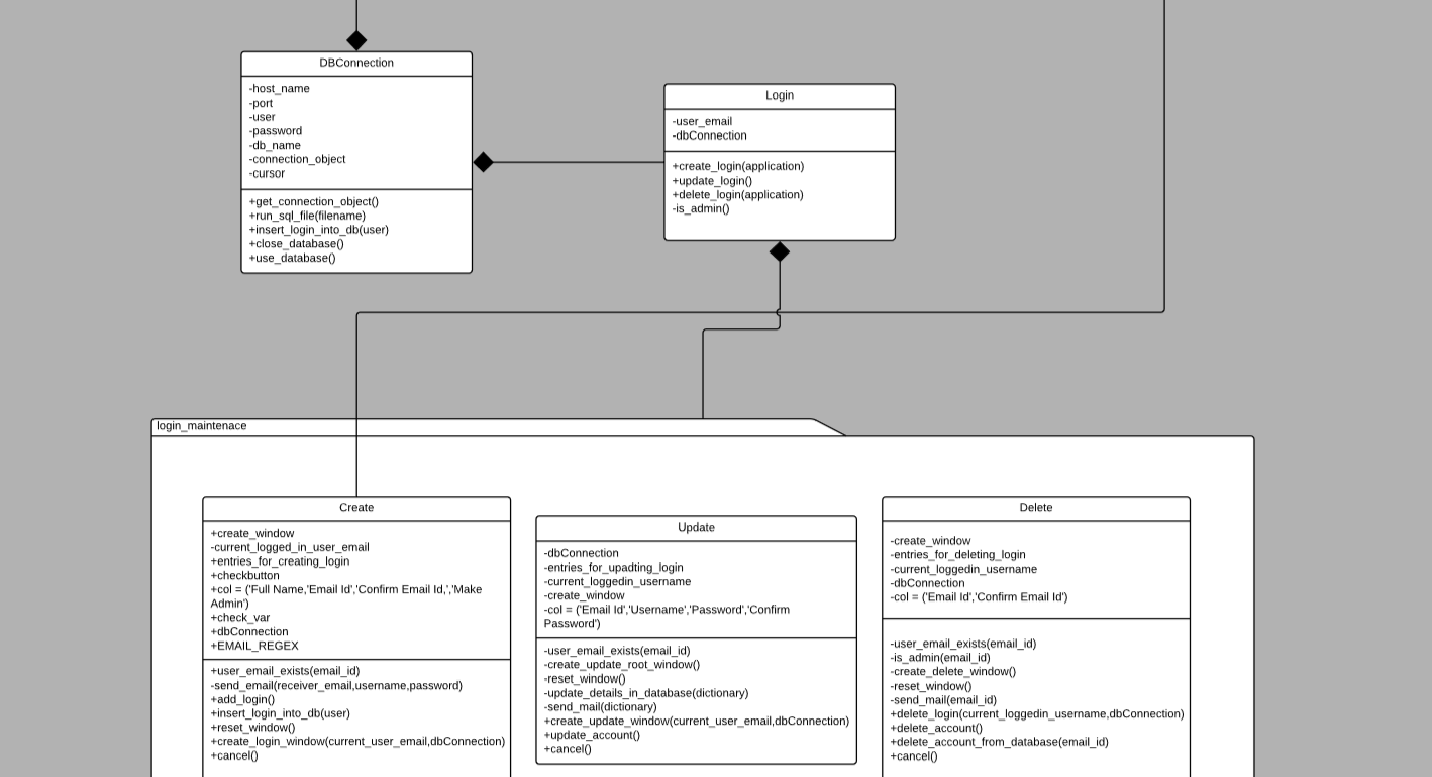
The respective class members and their methods are shown in the above diagram.

The next UML diagram is split from the whole relationship diagram. It consists of the classes “Main”, “Mail” and “User” classes. “Mail” class belongs to the module “mailing” and “User” class belongs to the “Common” module. It also consists of the package “common”. Common package consists of the classes that are used by almost every package or module in the project. It contains “Config”, “Hashing”, “User”, ”Values” classes.



The black arrow is the symbol for the strong composition between the packages or the classes.

“Main” class has a strong composition with the all the classes of the project. If the Main class is not there then the most of the important logic won’t work. Observe the diagram, it is in composition with the mailing and common modules.



“login\_maintenance” is the module that consists of all the classes required for creating, updating and deleting a login. Remember a super user/admin can only delete the logins. The respective logic is implemented in the project.

Login is the class that is in composition with the module “login\_maintenance”. That is, if the Login class doesn’t exist then the login\_maintenance package won’t exist. The “Create” class of the module “login\_maintenance” is also in composition with the “User” class of the module common. Because only a User can create a Login. So if a User doesn’t exist then the Login won’t exist.

Observe that we have a “DBConnection” class common and is strongly in composition with the “Login” class and the “Main” class.

“DBConnection” class consists of the properties and actions that are required for connecting to Database created for the project.

# Database and Its Tables

We are using MySQL as the database. Any application in the world needs faster performance with robust in nature. MySQL is the fastest database among all the databases and also free and open source.

## Database Connectivity

Installing database is nothing but installing a server. The server is known as database server. The database server when installed in localhost, stores all the information on the hard-disk.

For storing a client needs to first connect to the server and raise a request to the server. The client is nothing but a python program or application. For connecting the client to the server we use a connector. This connector connects to the database with the database server’s Ip Address and Port and then performs the client request.

Install the MySQL database server. The default port number to the MySQL database server is 3306. As the server is installed locally the Ip Address of the Server is 127.0.0.1.

We have a module that performs all these operations like connecting and disconnecting the database, CRUD operations etc. A module is a folder that contains some built-in classes and functions that performs a specific task(s).

We are using PyMySQl module for connecting the python program to the MySQl database.

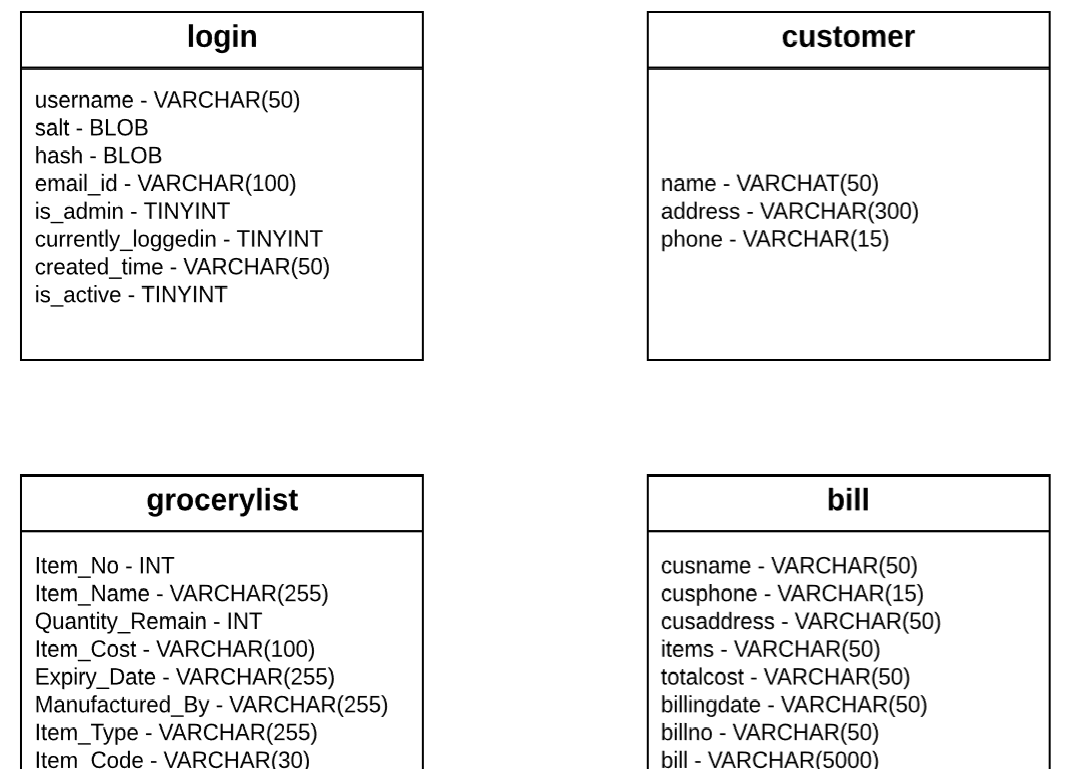
Install PyMySQL using PIP (Python Installation of Packages) using the following command:

***pip install PyMySQL***

## Database Tables

We have a bunch of tables used in the current project. We named the database as ‘grocery’. We can also change that in the ‘config.ini’ file and the ‘grocery.sql’ file. Following are the database tables used in the project:

* login – login details like username, type, admin, email id, etc
* customer – details of the customer are stored
* grocerylist – all the groceries are stored
* bill – for storing the bill items of the customer



The above picture shows the tables we are using in our project with their respective column names and type of the data they store.

# Scripting

Scripting can be defined in two ways:  
1) A piece of code of one language that is written in another code of different language that helps in performing a specific task.

Example: Javascript and PHP written in the HTML page that performs dynamic checking.

2) A piece of code/commands written in a language that can be executed by another language.

Example: Python script that executes another language code script.

Scripting helps in performing some task easily. The usage of Javascript in HTML is one of the revolutionary thing.

## Database Scripting

We used a piece of code that actually executes “.sql” schema file and creates the database schema on the MySQL database server. This process of executing a database commands/schema file that is of Structured Query Language by Python program to create the entire database schema is one of the examples of the Scripting. As we are using the Database commands we named it as Database Scripting.

The “grocery.sql” is given as follows:

CREATE DATABASE GROCERY;

USE GROCERY;

CREATE TABLE IF NOT EXISTS grocerylist (

Item\_No INT,

Item\_Name VARCHAR(255) DEFAULT NULL,

Quantity\_Remain INT DEFAULT NULL,

Item\_Cost VARCHAR(100) DEFAULT NULL,

Expiry\_Date VARCHAR(255),

Manufactured\_By VARCHAR(255) DEFAULT NULL,

Item\_Type VARCHAR(255) DEFAULT NULL,

Item\_Code VARCHAR(30) DEFAULT NULL

);

INSERT INTO grocerylist VALUES (1, 'Milk', 25, '23.00', '12/10/2018', 'Prairie', 'DAIRY PRODUCT', 'DAIRY-1234567890');

INSERT INTO grocerylist VALUES (2, 'Sandwich Wheat ', 93, '9.99', '08/31/2019', 'Essential Everyday', 'BREAD & BAKERY', 'BREAD-1478523690');

INSERT INTO grocerylist VALUES (4, 'Chocalate', 343, '15.33', '12/04/2019', 'Tempteys', 'DAIRY PRODUCT', 'DAIRY-3698521470');

INSERT INTO grocerylist VALUES (5, 'Cake', 113, '12.99', '12/12/2019', 'Hersheys', 'CAKES', 'CAKES-0123654987');

INSERT INTO grocerylist VALUES (6, 'Marie ', 122, '2.99', '12/01/2019', 'Britannia', 'BISCUITS', 'BISCU-7896543210');

INSERT INTO grocerylist VALUES (7, 'Biscuits', 258, '64.25', '25/12/2018', 'Karachi', 'BISCUITS', 'BISCU-2587533571');

INSERT INTO grocerylist VALUES (9, 'Perfume', 12, '152.32', '28/08/2019', 'Fogg', 'COSMETICS', 'COSME-1599514753');

INSERT INTO grocerylist VALUES (10, 'Kaju', 47, '850.85', '25/04/2020', 'Haldi', 'DRYFRUITS & NUTS', 'DRYFR-1236549875');

INSERT INTO grocerylist VALUES (11, 'Knifes', 15, '84.47', '21/04/2020', 'WeCool', 'KITCHENWARE', 'KITCH-0147852369');

INSERT INTO grocerylist VALUES (13, 'Apple', 25, '25.00', '12/11/2018', 'ITC', 'FRUITS', 'FRUIT-1587423240');

INSERT INTO grocerylist VALUES (14, 'Basmati', 12, '95.00', '25/09/2019', 'India Gate', 'RICEITEM', 'RICEI-8574698525');

CREATE TABLE IF NOT EXISTS customer (

name varchar(50),

address varchar(300),

phone varchar(15)

);

INSERT INTO customer VALUES('Hemanth','shardanagar','9125847120');

INSERT INTO customer VALUES('apurva','springfield,illinois','8745631278');

INSERT INTO customer VALUES('apurva','springfield, illinois','8845712369');

INSERT INTO customer VALUES('sumiit','srinagar','7548961234');

INSERT INTO customer VALUES('Rakshitha','shardanagar','7744112233');

INSERT INTO customer VALUES('Kumar','Bangalore','7894561230');

INSERT INTO customer VALUES('Teja','Karnataka','1478529630');

INSERT INTO customer VALUES('Shilpa','s','4567891236');

INSERT INTO customer VALUES('sowmya','illinois','7852369871');

INSERT INTO customer VALUES('sana','india','9100415587');

INSERT INTO customer VALUES('tarun','sprinfield','6975412770');

INSERT INTO customer VALUES('somya','Chicago','6975412445');

INSERT INTO customer VALUES('shreya','illinois','8423579612');

INSERT INTO customer VALUES('sana','hyd','8966554471');

INSERT INTO customer VALUES('rohit','asas','7814526987');

INSERT INTO customer VALUES('Kiran','Hyderabad','4569321247');

INSERT INTO customer VALUES('srinija','springfield,illinois','9875310252');

INSERT INTO customer VALUES('somya','springfield,illinois','9875310252');

CREATE TABLE IF NOT EXISTS bill (

cusname varchar(50),

cusphone varchar(15),

cusaddress varchar(50),

items varchar(50),

totalcost varchar(50),

billingdate varchar(50),

billno varchar(50),

bill varchar(5000)

);

COMMIT;

create TABLE IF NOT EXISTS login

(

username varchar(50) not null,

salt blob not null,

hash blob not null,

email\_id varchar(100) not null,

is\_admin tinyint(1) default '0' null,

currently\_loggedin tinyint(1) default '0' null,

created\_time varchar(50) not null,

is\_active tinyint(1) default '1' null,

constraint login\_email\_id\_uindex

unique (email\_id)

);

COMMIT;

INSERT INTO login VALUES ('admin', 0x243262243132244754586552304C76756946784B6B302F634762374C4F, 0x243262243132244754586552304C76756946784B6B302F634762374C4F666467667A62376856316F684F564C3472534C4E64344D33646E78342F464B, 'sowryaa.datatechs@gmail.com', 1, 0, '2018-09-17 18:14:31.052558');

COMMIT;

All the above commands perform a particular task on the database when executed by the Python program.

The following “run\_sql\_file()” method of the “DBConnection.py” class in the Project source code is responsible for execution process.

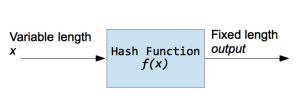
**def** run\_sql\_file(self,filename):  
 *'''  
 The function takes a filename and a connection as input  
 and will run the SQL query on the given connection  
 '''* file = open(filename, **'r'**)  
 sql = file.read().replace(**'\n'**, **''**).split(**';'**)[:-1]  
  
 **for** sql\_command **in** sql:  
 **try**:  
 self.\_cursor.execute(sql\_command)  
 **except** ProgrammingError **as** exec:  
 error = str(exec)  
 **if "database exists" in** error:  
 self.use\_database()  
 print(**"Database Already Exists"**)  
 **break**;  
 **else**:  
 messagebox.showerror(**"Error"**,error)  
 sys.exit(0)  
 **else**:  
 print(**"Executing : \n {}"**.format(sql\_command,))  
 time.sleep(1000)  
  
 self.\_connection\_object.commit()

# Cryptography Hashing

Hashing is mainly used in security applications for securing the information. Hashing is the process of converting variable length of input bytes to a fixed length sequence of bytes in one way. That means we change an input X (say) to an output Y (say) that cannot be again changed into X. Please observe that hashing is done on bytes.

## Hash Function

A hash function is a function that takes input of a variable length sequence of bytes and converts it to a fixed length sequence. It is a one way function. This means if f is the hashing function, calculating f(x) is pretty fast and simple, but trying to obtain x again will take years. The value returned by a hash function is often called a hash, message digest, hash value, or checksum.



* Hash functions are used inside some cryptographic algorithms, in digital signatures, message authentication codes, manipulation detection, fingerprints, checksums (message integrity check), hash tables, password storage and much more.
* In general, the hash is much smaller than the input data, hence hash functions are sometimes called **compression functions**.
* Since a hash is a smaller representation of a larger data, it is also referred to as a **digest**.
* Hash function with n bit output is referred to as an **n-bit hash function**. Popular hash functions generate values between 160 and 512 bits.

Some of the popular Hash function algorithms are:

* 1. MD5
  2. SHA
  3. WHIRLPOOL
  4. BCRYPT

## Problems with Cryptographic Hash Algorithms

Brute Force attack: Hashes can’t be reversed, so instead of reversing the hash of the password, an attacker can simply keep trying different inputs until he does not find the right now that generates the same hash value, called brute force attack.

Hash Collision attack: Hash functions have infinite input length and a predefined output length, so there is inevitably going to be the possibility of two different inputs that produce the same output hash. MD5, SHA1, SHA2 are vulnerable to Hash Collision Attack

To overcome such issues, we need algorithms which can make the brute force attacks slower and minimize the impact. Such algorithms are PBKDF2 and [BCrypt](https://en.wikipedia.org/wiki/Bcrypt), both of these algorithms use a technique called [Key Stretching](https://en.wikipedia.org/wiki/Key_stretching).

### Salting

In Software Industry we just don’t directly hash the passwords and store it in the database. Because the Hash Collision attack or dictionary attack contains a very long list of commonly used password hashes. So a hacker can check the password with his dictionary and there can be a chance of occurrence. So, for eliminating this we use a random code with the password and then hashed. This reduces the security breach problem. This random piece of code or string combined with the password is called Salt.

After generating the hash, we store both the hash and the salt in the database. When a user enters his username and password, we retrieve its respective hash and salt from the database and combine the salt with the entered password and generate the hash. This newly created hash is then compared with the hash that is return from the database. If both the hashes match then the username and the password are correct.

### BCrypt

Bcrypt is a hashing algorithm which is scalable with hardware (via a configurable number of rounds). Its slowness and multiple rounds ensures that an attacker must deploy massive funds and hardware to be able to crack your passwords. Add to that per-password [salts](https://en.wikipedia.org/wiki/Salt_(cryptography)) (bcrypt REQUIRES salts) and you can be sure that an attack is virtually unfeasible without either ludicrous amount of funds or hardware.

In our project we used the Bcrypt algorithm for hashing the passwords. We generate a hash of a string along with a salt and then store both the salt and hash in the database. Later we compare the hashed password with newly entered password after generating the hash.

Refer to the Hashing.py class of the common module for checking the class works.

# Graphical User Interface

We are using “tkinter” module for the GUI. “tkinter” is a python module through which we can design the frontend applications. “tkinter” is directly available when we install Python. The following are the widgets used in the project:

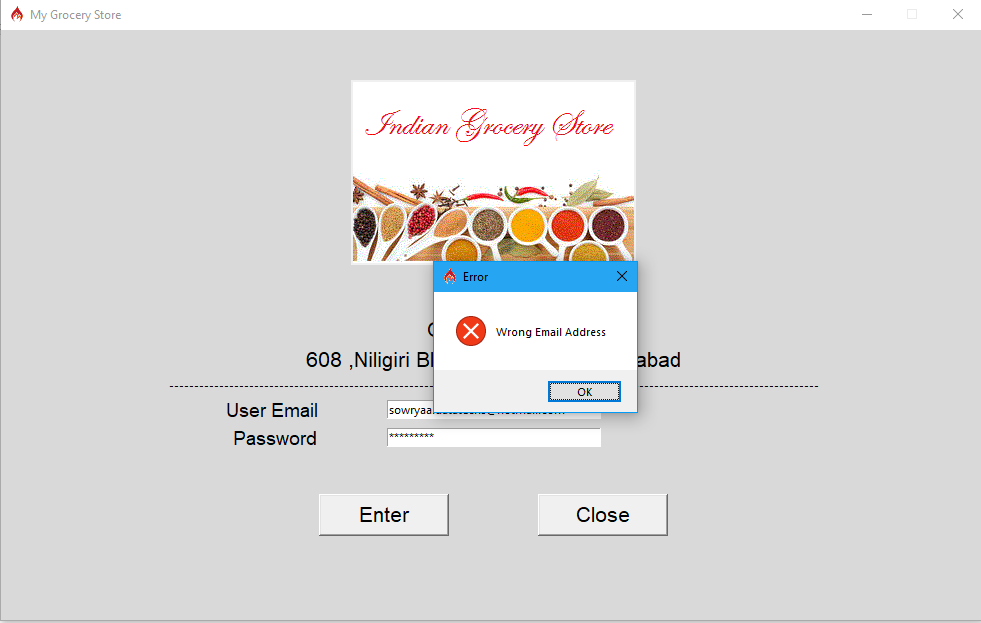
1. List boxes
2. Buttons
3. Scroll bars
4. Entry field
5. Label
6. Messagebox
7. Menu
8. Menu bar
9. Check box
10. Drop down
11. Text Field

## Login Screen

We have two Entry fields for entering username and password of the user. Please remember we create a login at that time of creating the database. Only an admin can create a user account. When the user account is created the login details are sent to the logged in user’s email address.



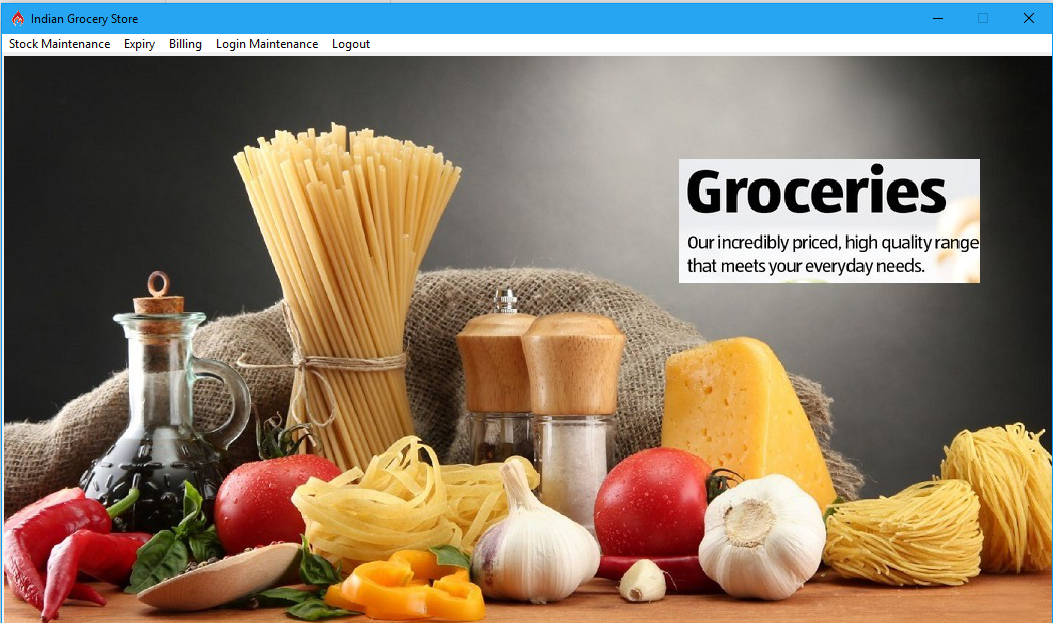
After entering the username and password the details are validated in the background. If valid then we prompt the user to the Main window screen. If the details are wrong then we show a message “Username not valid” if username is wrong and “Password is wrong” if password is wrong. When the user enters the username and password and clicks on the button “Enter” first we get the records from the database having entered username. We generate a hash for the entered password and compare that with the stored hash. If both hashes are same then the login details are valid and will show the Main screen.



The above image shows the message “Wrong Email Address” in a message box. If the user email entered is wrong we show this message.

## Main Window Screen

When the entered credentials are valid we show the next screen which is the main window screen. Below is the image of the Main window.



After logging in we have a menu bar with the following menu items:

1. Stock Maintenance
2. Expiry
3. Billing
4. Login Maintenance
5. Logout

### Stock Maintenance

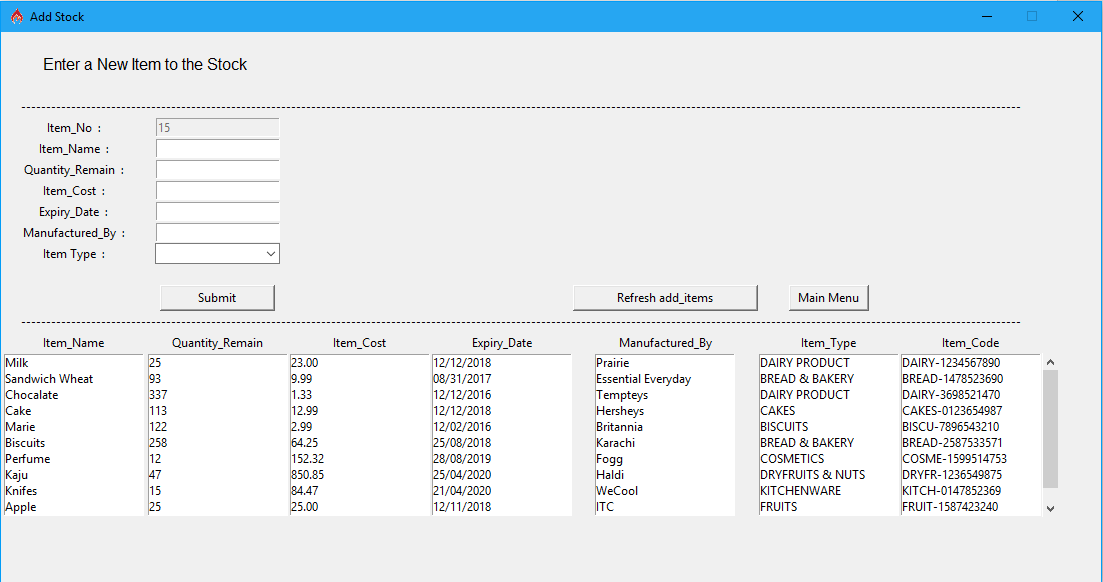
Stock Maintenance meu item again contains a menu list of actions. Those are,

1. Add Items
2. Update Items
3. Delete Items

#### Add Items

In this section we can add a new item to the database. We have seven list boxes showing already available items from the database. The list boxes are:

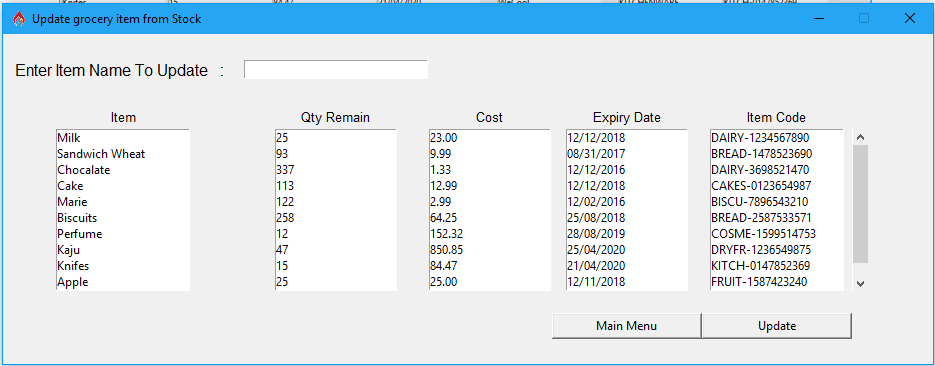
1. Item\_Name - Displays names of the Items that are present in the database
2. Quantity Remaining – Quantity of the respective Item
3. Item\_Cost – Cost of the Item
4. Expiry\_Date – Expiry of the Item
5. Manufactured\_By – Name of the Manufacturer
6. Item\_Type – Type of the Item like Vegetable, Milk product, Bakery etc
7. Item\_Code – Unique code that is generated internally for every Item.



We also have the Entry fields for adding the Item details. After entering the Item details we can click on Submit button and the details are saved into the database and is reflected back in the list boxes.

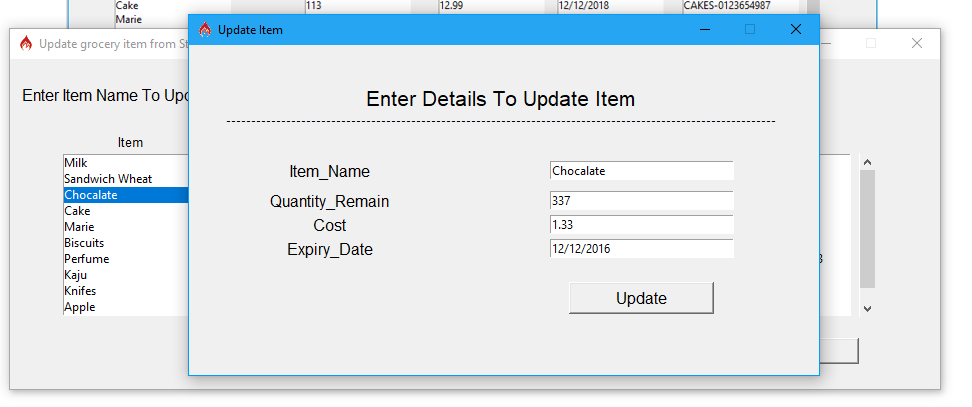
#### Update Items

We can also update the already available Items like quantity of the item, expiry of the item, cost of the item etc.



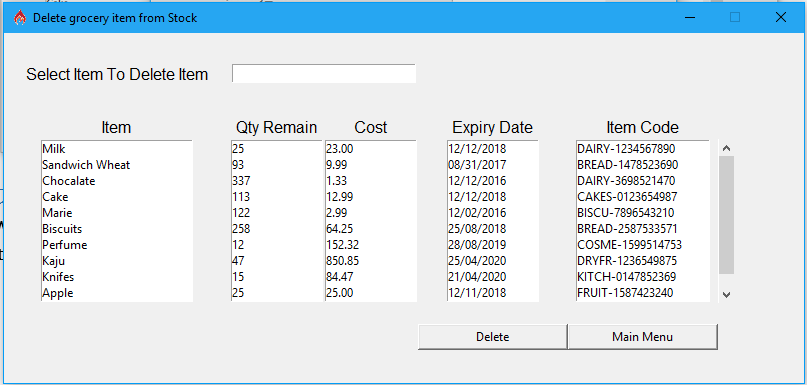
We are showing the Item Name, Qty\_Remaining of the Item, Cost of the Item, Expiry Date and Item Code of the already available Items in the database. We have an Entry field for searching the Item Name.

After selecting the Item from the Item\_Name list box click on “Update” button to update the details of the Item.



#### Deleting the Items

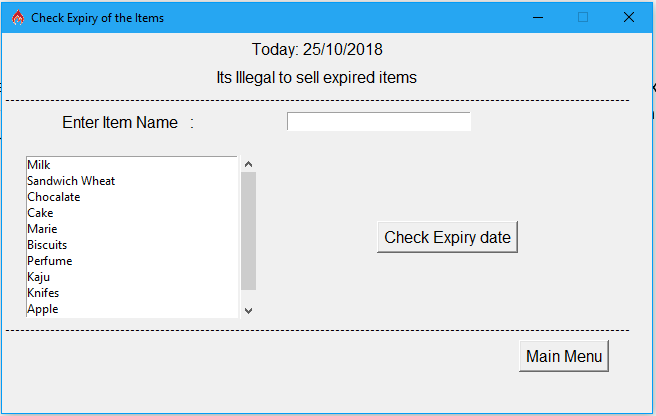
We can also delete an item from the stock. Click on the Delete Items from the Stock Maintenance menu item on the Menu bar for navigating to the Delete items window.



The delete window is similar to the Update window. Search for an Item by entering in the Entry field and click on the Delete Button. When you click on Delete a messagebox appears to confirm the action. Clicking on MainMenu button returns to the Main window screen.

## Expiry

We can check expiry date of an Item that is already present in the database from the Expiry menu. Navigate to the Expiry menu and click on “Check Expiry” menu item on the menu bar in the main window screen.



We have an Entry field to search for an Item. All the available item names are displayed in a list box. Click on an Item and click on the “Check Expiry Date” button to see the expiry of the item.

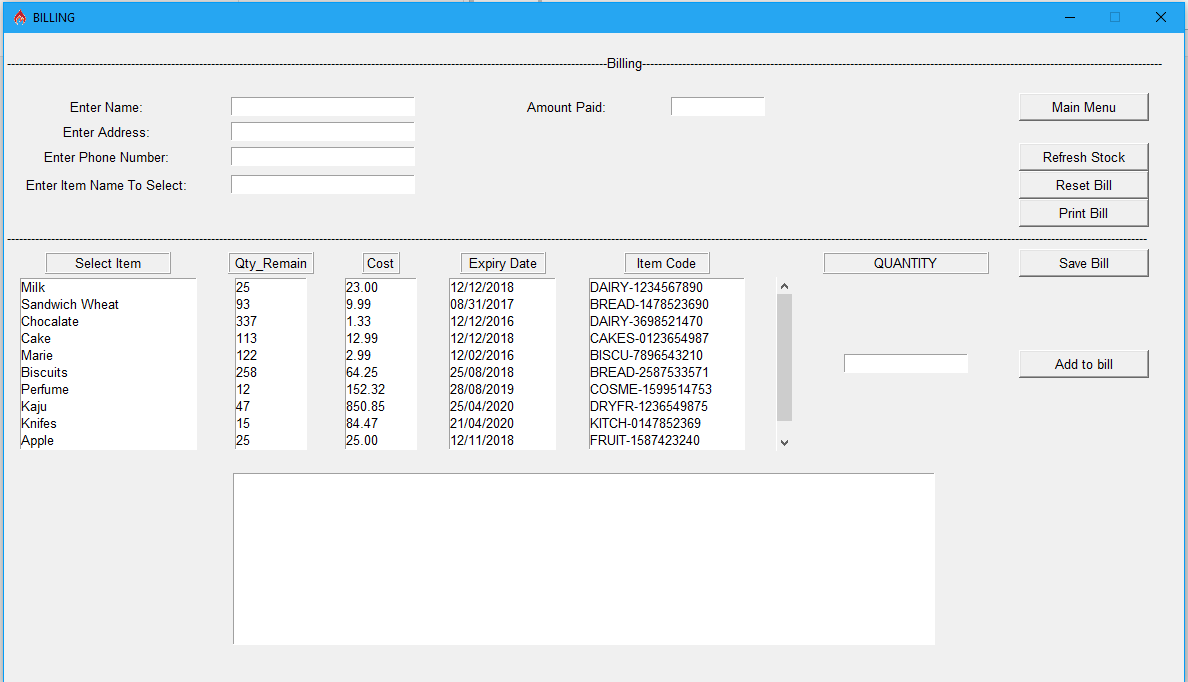
Click on the Main Menu button to return to the Main Window Screen.

## Billing

Billing is the most and complicated thing. The code implemented for billing is huge as there are so many features included. We can also check the income of the day based on the Items sold on that day.

Click on the Billing Menu and navigate to the “Billing” section.

When a customer purchases a list or a single item we have to produce a bill to the customer. This bill details and the customer details have to be stored into the database. We should also print the bill. We implemented code for printing a generated bill by connecting to the “default printer”. We can also save the bill in a text document and also into the database.



We have Entry fields for entering the Customer details like Name, Address, and Phone Number. We also have an Entry field for searching the item from the list and add that to the bill. Select an item and enter the quantity customer purchased and click on “Add to bill” button to add the item to the bill. When an item is added to the bill the Item details, its quantity bought, cost of the item and total price of the item is added onto a text field.

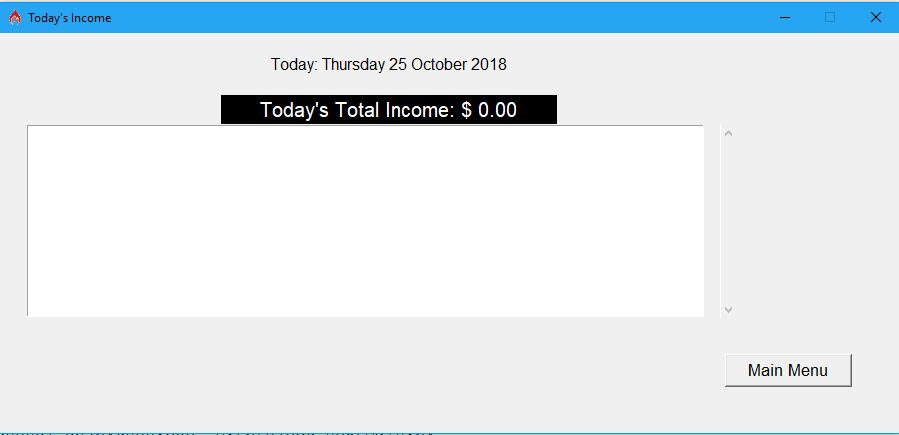
After adding an item or list of items we can save the bill or print the bill. We can also reset the bill by clicking on the “Reset bill” button. This clears all the Entry fields and text fields.

There is also an Entry field “Amount Paid” that accepts the input of how much customer paid to the items.

### Check Today’s Income

Whenever a bill is done the details of the bill are saved into the database automatically. Based upon these bills the billing manager can check for the Income gained on that particular day. Income is calculated for a day buy summing up all the bill values purchased by the customer for that day.

We can find all the bill details including the bill number and the amount of the bill purchased by the customer in the text area.



## Login Maintenance

An admin user can add an account, update his account and also can delete an account. Whenever an application is created we must have to create an account for security. We implemented a very decent login creation in the project.

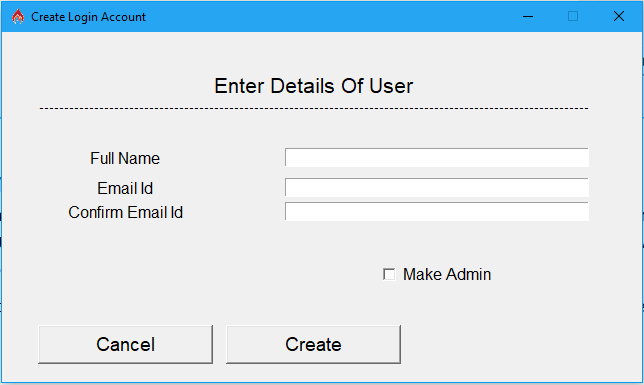
Navigate to the “Login Maintenance” menu in the menu bar. We can find the following menu items:

1. Create Login
2. Update Login
3. Delete Login

Please remember any user can update his account. If the user is an admin then he can create an account and can also delete an account. A user account is first created by the admin and is given to the user.

#### Create Login

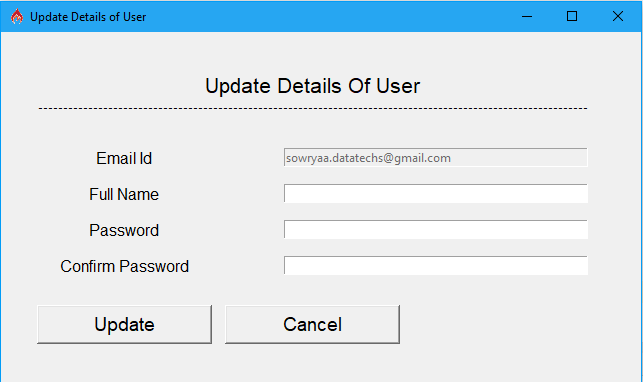
As said when the admin creates a user, he can make an account as admin. And so the created account also acts as an admin. When the logged in user clicks on the “Create Login” section we first check whether the logged user has admin privileges. If yes, then we allow the user to the next screen. If not, we show an error message “No admin rights to add an account”.



Full Name, Email Id and Confirm Email Id fields are mandatory fields for creating the user account. When all the details are entered and the email address matches with the Confirm Email Id then we create password for the user with the Email Id as the user name. The password is random and the length of the password can be configured in the “Values.py” of the common module by modifying the value of the variable “LENGTH\_OF\_THE\_RANDOM\_PASSWORD”. Currently we are using 10 characters. After creating the user account the user name and the password are shared to the current logged in user’s email address. We used SMTP lib module for sending the mails.

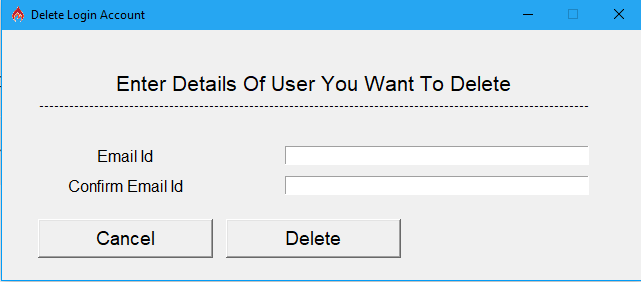
### Update Login

The current logged in user can update his details like full name and his password. Whenever he updates his details we send a mail in the background to his email id. Please remember the email id he entered during the creation of his account cannot be changed as it is unique.



#### Delete Login

An admin can only delete a user account. In fact we generally won’t delete the account from the database. We just make the account disable for future purpose.



## Logout

Click on the Logout in the Menu bar for logging out from the application. We show a message box for confirming from the user side whether to logout and if yes then the database is committed with all the changes and the application main window is destroyed and the login screen is shown back. Now the user can close the application or can re-login back to the application.

# References

https://www.google.co.in/

<https://creately.com/>

https://www.tutorialspoint.com/