CHAPTER 1

INTRODUCTION

Python uses Tkinter (Tk=Tool kit; inter=interface) to create GUI (Graphical User Interface). GUI helps to create a display interface in windows which is being used to take input from users and displaying output. Inserting and accessing data is happening with the help of backend through SQLite3 connection.

The create table command has been implemented in Python from which the database has been invoked to create the tables as well as the insert queries inserts the values into that table. Applications such as adding member, client and project, showing details of all the members, projects and clients, sending and receiving messages, showing bar chart , etc. have been done in Python as a frontend which is processed in backend with the help of SQL in SQLite database.

The project contains entire information about the members, projects, clients for each teams. In the SQLite seven tables (Teams, Members, Clients, Role, Project, Client\_mobile, Workon) have been created in the backend. All table has its own attributes and are related to each other through various participation and relationship which is described briefly below in coming chapters.

* 1. Course Objectives:
* To learn how to design and program Python applications.
* To learn how to build and package Python modules for reusability.
* To learn how to design Object oriented programs.
* To learn how to use class inheritance in Python for reusability.
* To learn how to use exception handling in Python applications for error handling.
* To make better understanding of lists, tuples, dictionaries in Python programs.
* To understand why Python is a useful scripting language for developers.
  1. Problem Definition:
* Project management has become a major issue in this era.
* It has become a tedious task for a team captain for managing multiple projects at a time.
* Copping up with the detailed report about the progress of the work and team members.
* No any data visualization rather same old traditional regular paper work for each progress.
* Members had to reach office for each work review.
* Updating due date issues, using cellular network for communications and so on.
  1. Outcomes of Project Work:
* Traditional paper work replaced by digital report.
* Easy for Team captain to manage and look after various projects.
* Regular update of all the team members and progress of the work involved in particular project.
* One tap review: No need of rushing to office for regular review.
* Management of multiple projects.
* Easy to use interface (user friendly interface).
* Data visualization using Bar Chart for reports which increases the understandability.

CHAPTER 2

REQUIREMENTS AND DESIGN

* 1. Hardware Requirements:
* Processor: Intel core i3
* Ram: at least 1gb
* Hard Disk: 10gb or higher
* Mouse & Keyboard
  1. Software Requirements:
* Operating System: Windows 7 or later
* Platform: Python IDE or Jupyter Notebook or Pycharm
* Database Software: SQLite3

CHAPTER 3

Data Models and ER Models

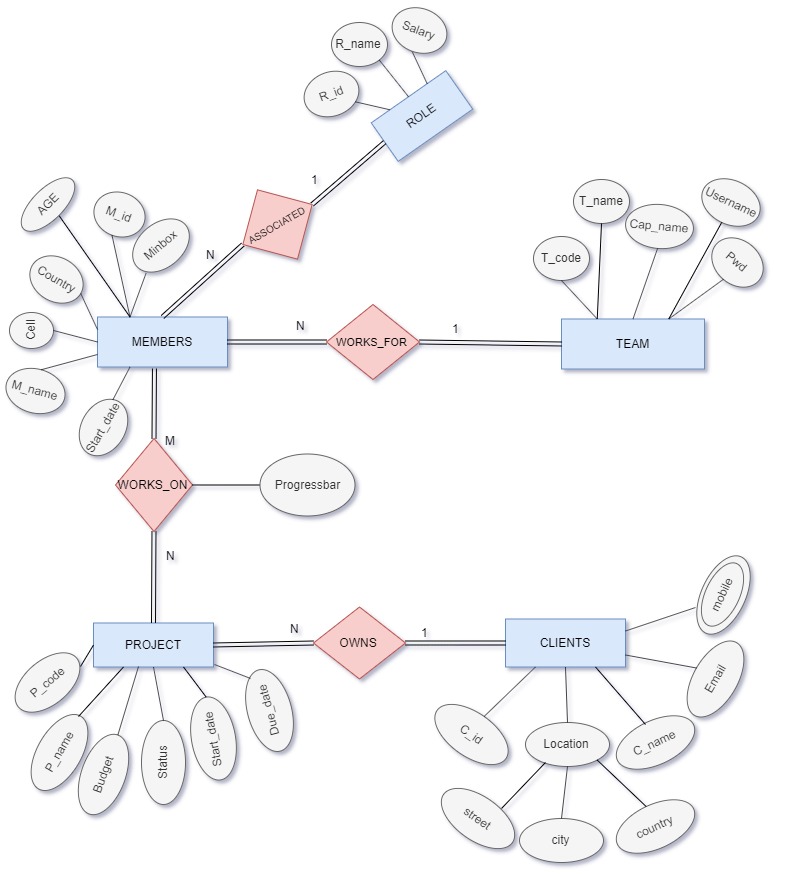


Fig 3 a ER diagram of Gaffer

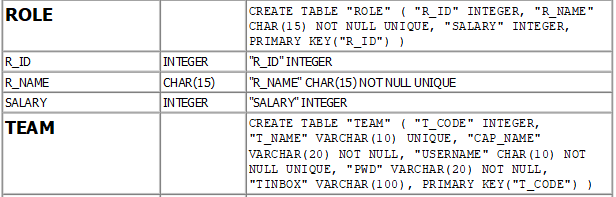


Fig 3 b Role table Schema

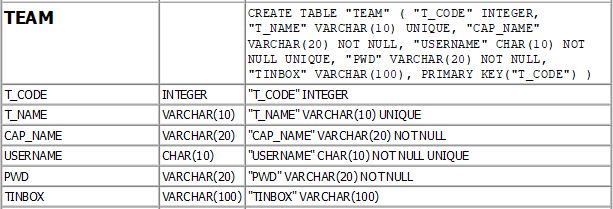


Fig 3 c Team table Schema

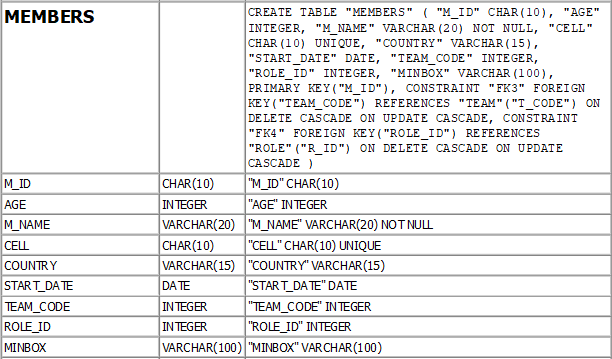


Fig 3 d Members table Schema

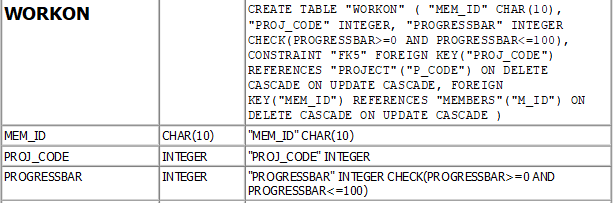
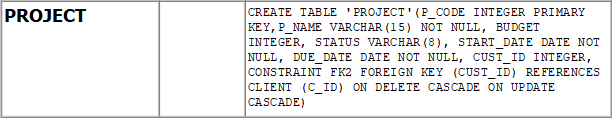


Fig 3 e Work On table schema



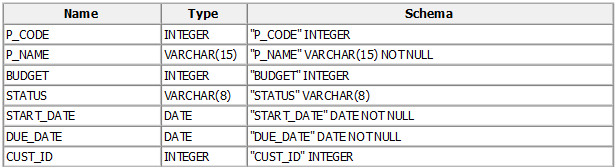


Fig 3 f Project table schema

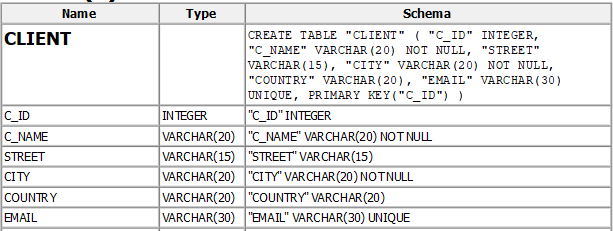


Fig 3 g Client table schema

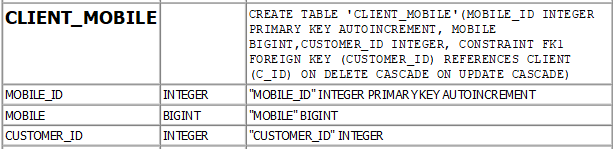


Fig 3 h Client\_mobile table schema

* 1. Entity and Attributes:

Entity type is collection of entities which consists of various attributes which may be of various data types and can have various constraints. While selecting an entity we must always care of two things, does that selected entity have enough members and also enough attributes. If the entity satisfies both the conditions then such entities are called good entity but if the entity fails to satisfy one of these conditions then such entities are considered to be bad entities. Attributes for each entity should have a proper data type assigned to it and may or may not have constraints.

All the entities of Gaffer are good entities because all of them has enough members and enough attributes. The entities used in Gaffer and corresponding attributes are listed below:

* TEAM (T\_code, T\_name, Cap\_name, Username, Pwd)
* MEMBERS (M\_id, M\_name, Age, Cell, Country, Star\_date, Minbox)
* ROLE (R\_id, R\_name, Salary)
* PROJECT (P\_code, P\_name, Budget, Status, Start\_date, Due\_date)
* CLIENTS (C\_id, C\_name, Email, Street, City, Country, Mobile)

* 1. Keys:

Keys is a constraint used while defining attributes in a table. Keys is used to identify a row in a table. Keys plays a vital role in finding the relation between two tables. It helps you uniquely identify a row in a table by combination of one or more columns in that table. There are various keys which has various properties, one of them which is widely used is Primary key. Primary key is a unique identification of a table that is used while combining tables and it can never be NULL. In a single table more than one column can be primary key. When this primary key column is used in other tables then that becomes Foreign key and its values can be NULL. Both primary and foreign key plays important role in determining relation between two tables. Unique key is also one of them which ensures that the particular column has unique values.

* 1. Relationship and Participation:

If one table has a foreign key that references the primary key of another table then there exists relation between two relational databases. This means one or the other way a entity is related with the other like member entity type is related with project entity with the relation “works\_on”. Participation may be 1: 1, 1: N or M: N. One entity can relate with another table and can have partial or full participation. Here partial participation means the members of one entity may or may not be associated with the other entity whereas the full participation means that at least one members of on entity is associated with the other entity.

Gaffer associates all relation with full participation which is reflected by the ER diagram above i.e. Fig 3.1. It means one or the other way at least one of the members of each entity is associated with other which are in relation. A sample snippet for relationship and participation from this project is as given below:

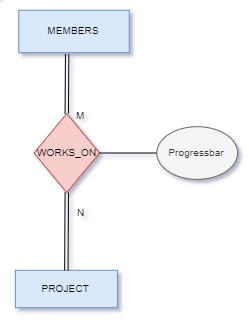


Fig 3.3.1 Members-Project relation and participation

* 1. Problem Statement (assumptions):
* Members are represented by its unique key Member id (automatic numeric values), name, cell number, country name, minbox and project started date of the project.
* Every team has a captain, username (no two team captain are allowed to have same username, alphanumeric), password and has a unique team code and name.
* Every project has its unique code, name, budget of project, active status, date of starting and date of completion.
* A Client is represented by unique client id (auto numeric values), Location, name, email and mobile.
* Every role has its unique Role id (auto numeric values), name and salary for each role.
* A team can have many members but a member can work for only one team.
* One project can be associated with many members but one member does can work on many projects. ( Member with no project is possible)
* A client can have many projects working for it and a project belongs to only one client.
* Each member must be designated with one role and many members can have similar role.

CHAPTER 4

Relational Model concepts and Schema

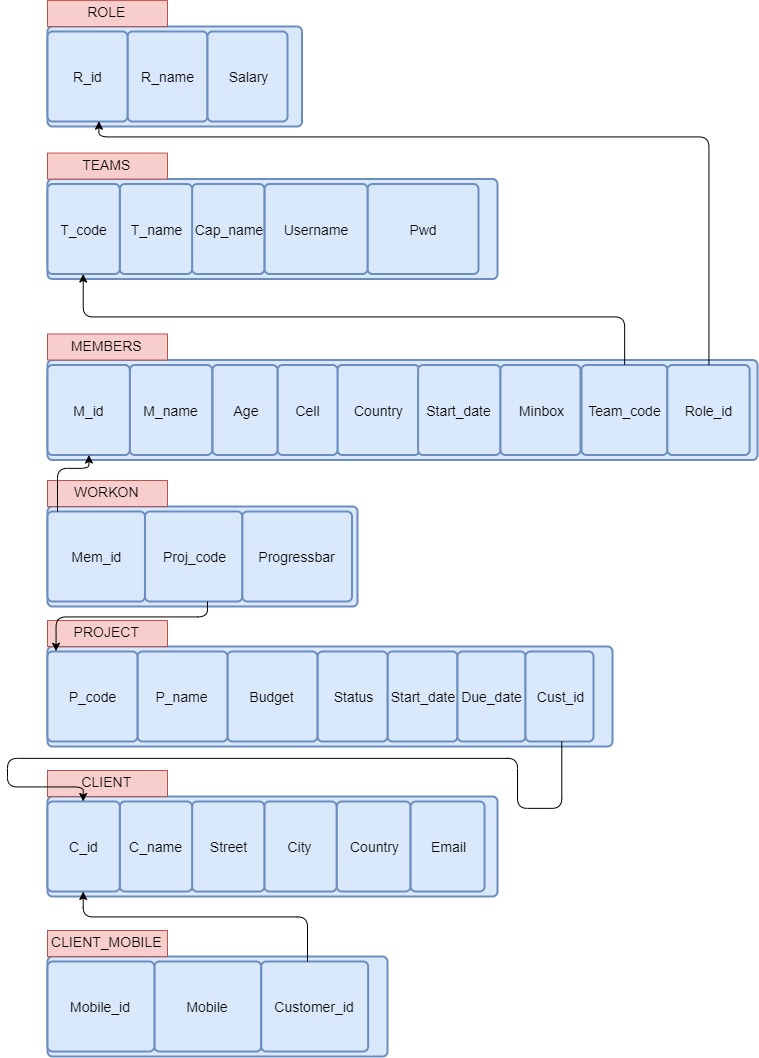


Fig 4.1 Relational Schema

CHAPTER 5

SQL

SQL stands for Structured Query Language. It’s a standard computer language for managing the relational database and manipulation of data. It is used to do all the operation in database like creation of schema, table, inserting, updating, deleting and retrieving rows. SQL is used by various database management system like: MySQL, oracle, vertica, Sybase, etc.

There are various SQL commands which are listed below:

* 1. DDL commands:

DDL stands for Data Definition Language. These are the commands used for schema or the table definition manipulation but not for data. There are various DDL commands which are listed below:

* + 1. CREATE:

This command is used to create a table and objects like view, triggers, assertions, etc. in database.

Syntax: CREATE TABLE <table\_name> VALUES (column1, column2,….);

e.g.: CREATE TABLE IF NOT EXISTS 'TEAM'(T\_CODE INTEGER PRIMARY KEY, T\_NAME VARCHAR(10), CAP\_NAME VARCHAR(20) NOT NULL, USERNAME CHAR(10) NOT NULL UNIQUE,PWD VARCHAR(20) NOT NULL,TINBOX VARCHAR(100));

* + 1. DROP:

This command is used to delete table or the objects from the database.

Syntax: DROP TABLE <table\_name>;

e.g.: DROP TABLE TEAM;

* + 1. ALTER:

This command is used to alter the table structure like table definitions, constraints, delete columns, add columns, etc. It has various options like ADD, MODIFY, CHANGE, DELETEetc.

Syntax: ALTER TABLE <table\_name> <option> <commands>;

e.g.: ALTER TABLE TEAM DROP COLUMN PWD;

* + 1. TRUNCATE:

This command is used to delete the contents of the table including all the spaces.

Syntax: TRUNCATE TABLE <table\_name>;

e.g.: TRUNCATE TABLE TEAM;

* 1. DQL commands:

DQL stands for Data Query Language for performing database queries.

* + 1. SELECT:

This command is used to retrieve the data from the database. It is one of the widely used command and is complex.

Syntax: SELECT <column\_name> FROM<table\_name> WHERE <condition>;

e.g.: SELECT \* FROM TEAM WHERE T\_code=900;

* 1. DML commands:

DML stands for Data Manipulation Language and is used to manipulate the rows in the table. The rows in the table can be updated, deleted, inserted, etc.

* + 1. INSERT:

This command is used to insert the tuples into the table.

Syntax: INSERT INTO<table\_name> <column\_name> VALUES <column\_values>;

e.g.: INSERT INTO TEAM (T\_CODE, T\_NAME, CAP\_NAME, USERNAME, PWD) VALUES (900, ’Ranger’, ’Dipesh’, ‘iamdipesh’, ‘iamdipesh’);

* + 1. UPDATE:

This command is used to update the existing values of the tuple. Variable values can be updated using this command.

Syntax: UPDATE <table\_name> SET <column\_name>=<values> WHERE <condition>;

e.g.: UPDATE TEAM SET PWD= ‘iamdipeshstha’ WHERE T\_CODE=900;

* + 1. DELETE:

This command is used to delete the tuples from the tables.

Syntax: DELETE FROM <table\_name> WHERE <conditions>;

e.g.: DELETE FROM TEAM WHERE T\_CODE=900;

* 1. TCL command:

TCL stands for Transaction Control Language which deals with the transaction within the database.

* + 1. COMMIT:

This command is used to commit the transaction so that the previous transaction will be successfully saved into the database. Once commit is done it is not possible to rollback.

Syntax: COMMIT;

* + 1. ROLLBACK:

This command is used to rollback/ undo the transaction if any error occurs.

Syntax: ROLLBACK;

CHAPTER 6

Python Features

6.1 Broad Standard Library:

Python has huge collection of defined library which makes very easy to code in python. Its library is portable and compatible with all platforms like Macintosh, UNIX, and Windows. You don’t have to write your own code for each and every thing as it provides rich sets of modules and functions. It has various libraries for web browsing, regular expressions, etc.

* 1. Interpreted Language:

Python is one of the Interpreted Language as its code is executed line by line at a time. It is not required to compile our code like in other languages like java, c++, etc. which makes it easier to debug our code. The source code of python is converted into an immediate form called byte code.

* 1. Support for GUI Programming:

Python provides various modules like PyQt, Tkinter, wxPython through which user can created Graphical User Interface for mobile applications. The most popular for creating graphical apps using python is PyQt5. Tkinter also provides all of the required options to create a beautiful user interface even Gaffer is made importing Tkinter module. This user interface can be connected to the backend using any one of the DBMS also supported by python, makes it more beautiful.

* 1. Object Oriented Programming:

Python is an object oriented programming language which include the concept of class and object. It support all OOPs concepts like inheritance, data abstraction, polymorphism, encapsulation etc.

* 1. Scalable and Extendable:

Python provides a better structure and support for large programs than shell scripting. You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

CHAPTER 7

Tkinter Widgets

* 1. Frame:

A frame is a container which associates other widgets. It is mainly used for grouping and organizing widgets. A bunch of labels, entry, buttons, etc. can be added into the frame and moving frame alone moves other widgets too. Various options like: bg, bd, cursor, relief, width, etc. can be used to configure frame.

Syntax: Frame( window\_name, options)

Gaffer has made use of multiple frames with the help of which other contents can be raised above the previous contents using same window without creating an extra window creating an illusion of user interface like in real world applications.

* 1. Label:

A label is a widget which is used to display non-editable text. Label infact is also used to display images using PhotoImage module. The most commonly used label is with ‘text’ configuration option and can change this at any time. Label makes use of many options like fg, bg, font, width, height, etc.

Syntax: Label( window\_name, options)

We can add an image into label as below:

project\_img=PhotoImage(file="path with file name")

project\_img\_label=ttk.Label ( captainDashboard, image=project\_img)

* 1. Entry:

A entry is a single line text field user can use to type anything. It’s mostly used in log in form for retrieving username and password. It has a special property to hide/ encrypt the text typed by user by using “show=’\*’ ”option which replaces each and every letter with the specified symbol/letter (in this case every letter typed by user is encrypted with ‘\*’).

Syntax: Entry( window\_name, options)

* 1. Button:

Button is one of the widely used widget among all in GUI with Tkinter. It is a functional widget that is clickable and on click it performs some action defined in the command option. It is used for linking two functions. They can display text or images same as labels, but also have a whole range of new options used to control their behavior.

Syntax: Button( window\_name, options)

Gaffer has multiple buttons with various functions like raising a frame over other, linking two functions, etc.

* 1. Listbox:

Listbox displays a list of contents which a user interact with and user can accept any number of times. It looks like a column of a tables that displays values in various rows. It provide option to browse, select multiple, select single through selectmode option. It also offers other variety of options like: bg, fg, font, height, width, highlightcolor, etc.

Syntax: Listbox( window\_name, options)

Gaffer makes use of listbox for listing members lists, projects lists, clients lists, etc. and uses the functionality of click and get values for further processing.

CHAPTER 8

Implementation

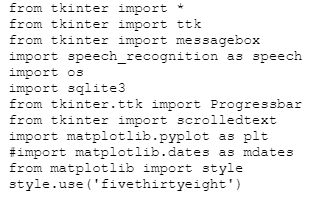


Fig 8.1 Importing required Python modules

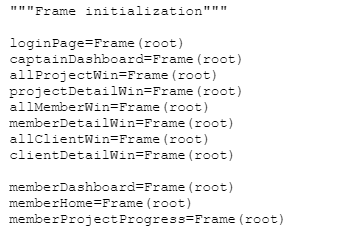


Fig 8.2 Frame initialization



Fig 8.3 Initializing backend (SQLite3)

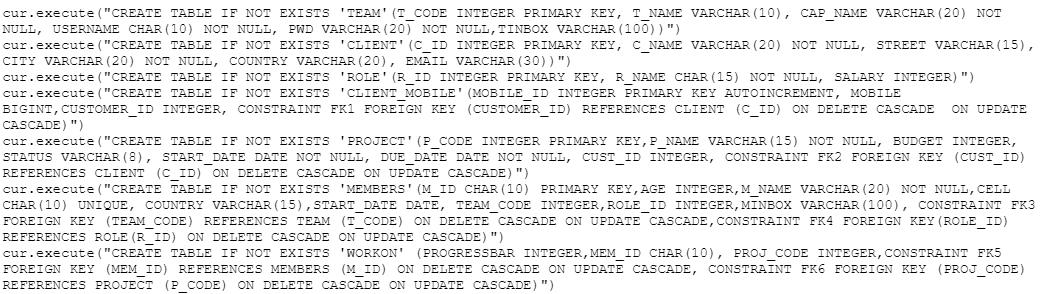


Fig 8.4 Creating tables query

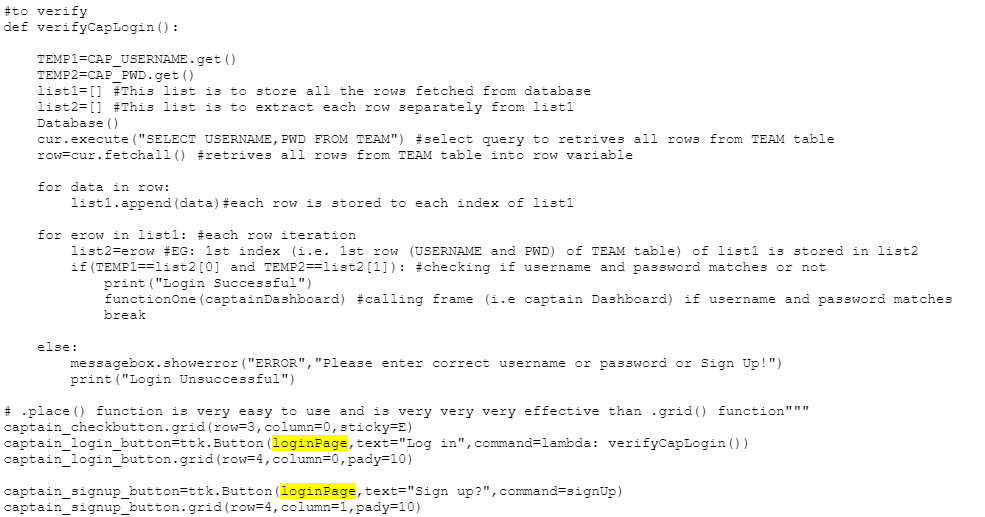


Fig 8.5 Verifying Captain login

Fig 8.6 Verifying Member login

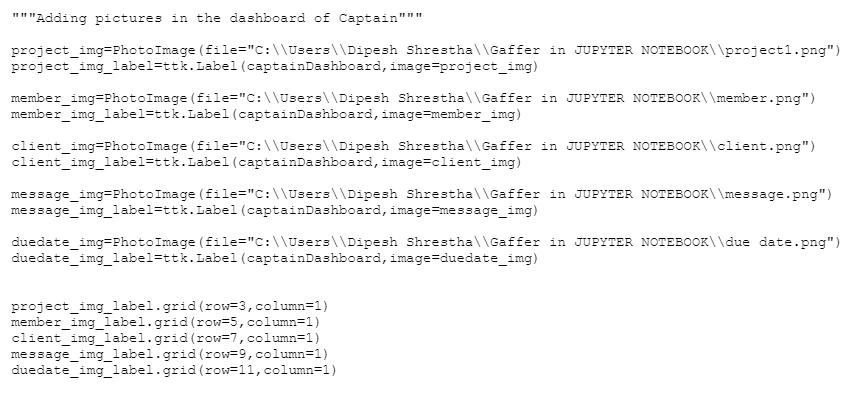
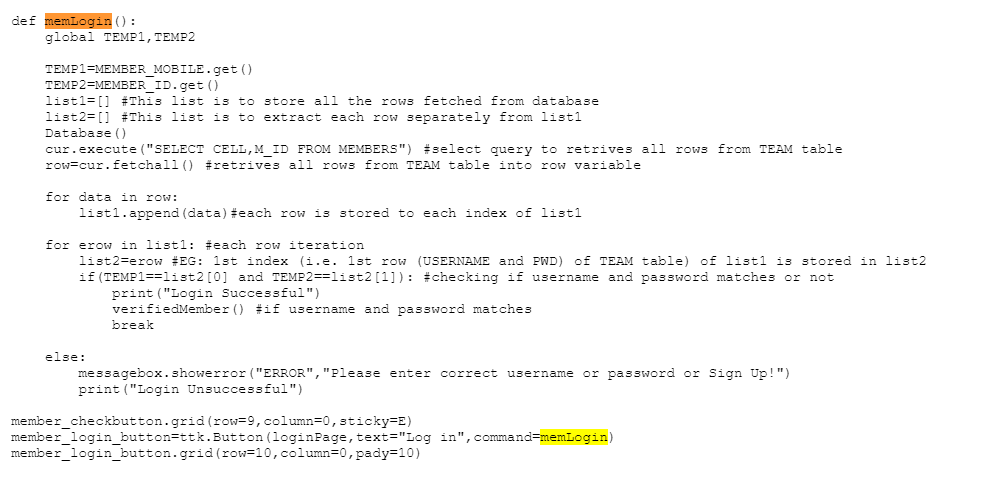


Fig 8.7 Image label

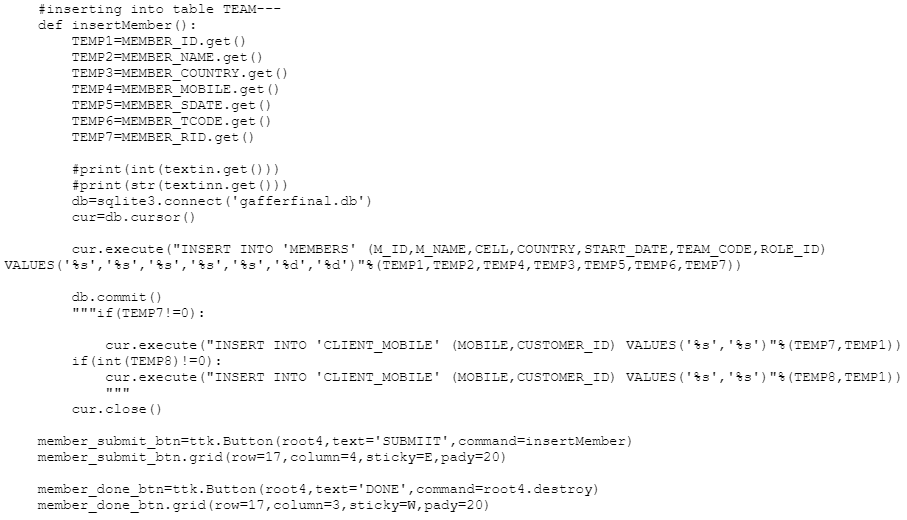


Fig 8.8 Insert query for members

CHAPTER 9

Output Snapshots

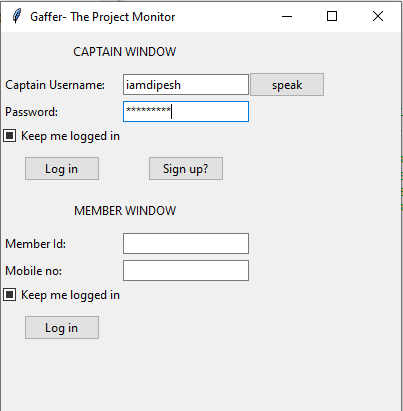


Fig 9.1 Login Page

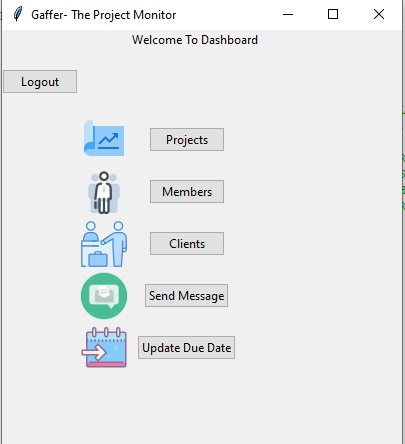


Fig 9.2 Captain Dashboard

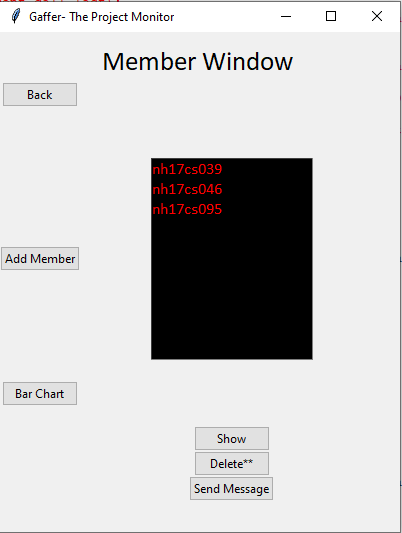


Fig 9.3 Member Window

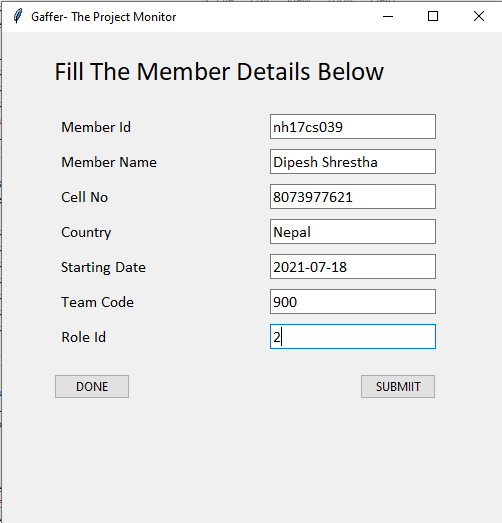


Fig 9.4 Adding Member

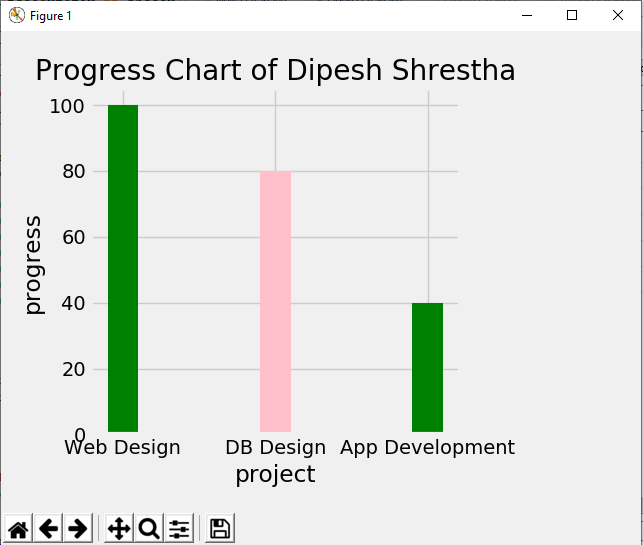


Fig 9.5 Member Progress Chart

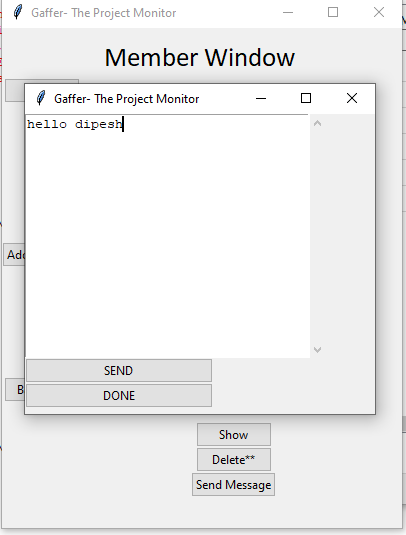


Fig 9.6 Captain Sending Message

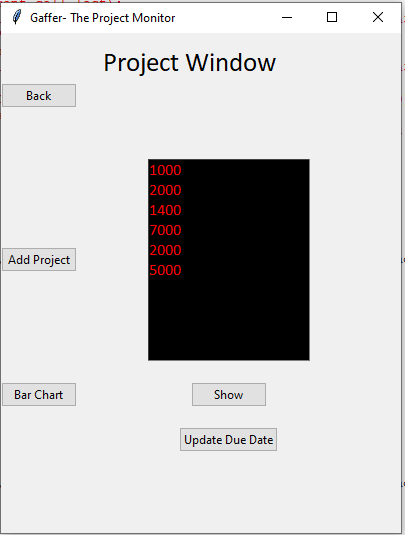


Fig 9.7 Project Window

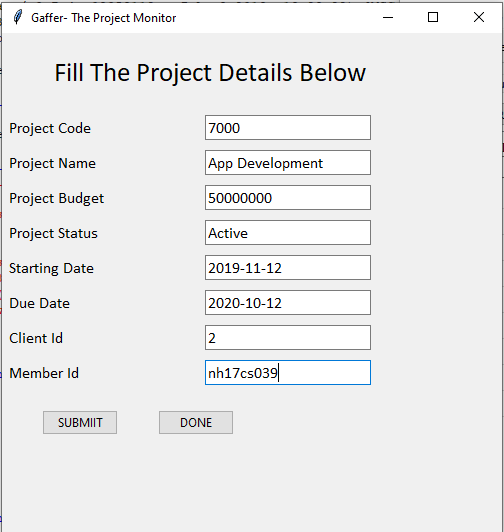


Fig 9.8 Adding Project

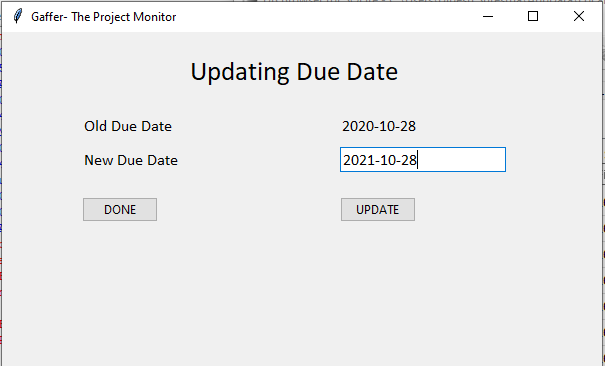


Fig 9.9 Updating Due Date

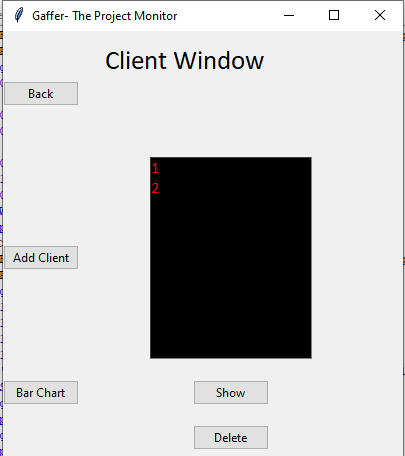


Fig 9.10 Client Window

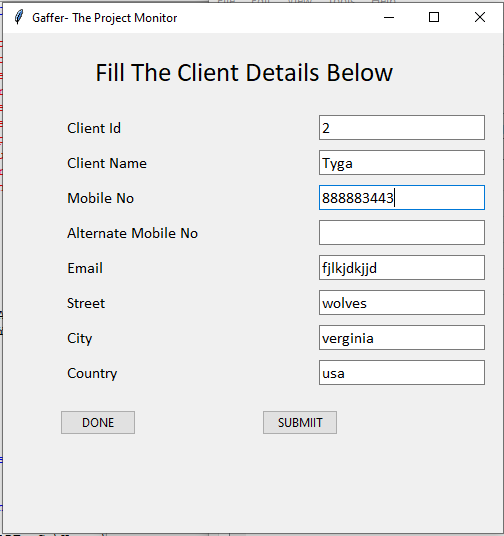


Fig 9.11 Add Client

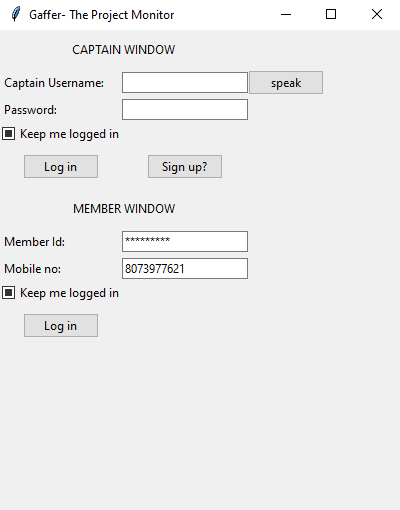


Fig 9.12 Member Login

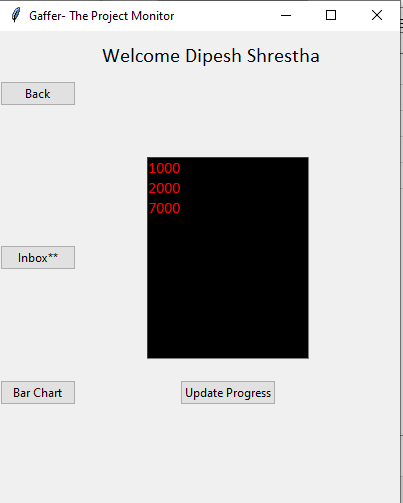


Fig 9.13 Member Dashboard

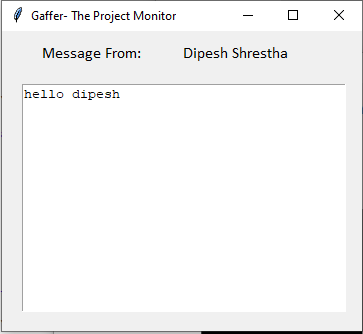


Fig 9.14 Member Inbox

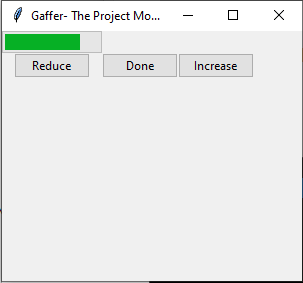


Fig 9.15 Progress of Member

CHAPTER 10

Conclusion

The main objective behind creating Gaffer application was to bring out changes in the corporate working environment. Its aim is to eradicate the ongoing traditional way of keeping track of projects, members, and clients. Keeping the record of each projects, members and clients in database is more efficient then in books physically which consumes a lot of time. Realizing all facts like digital recording of data, keeping track on members, data visualization, sending alert to members, checking progress on each projects, updating progress through any place would makes this beautiful application very useful in corporate world.

Further adding new features to it will continue in future and is currently working on automatic project assignment among the best members whenever a new project is added using algorithm.

References

1. Shipman, John W., Tkinter reference: a GUI for Python, New Mexico Tech Computer Center, 2012
2. <https://www.geeksforgeeks.org/python-gui-tkinter/>
3. <https://en.wikipedia.org/wiki/Tkinter>