



**NEW HORIZON
COLLEGE OF ENGINEERING**

Autonomous College, Affiliated to VTU | Approved by AICTE New Delhi & UGC
Accredited by NAAC with 'A' Grade & Accredited by NBA



**A MINI PROJECT
REPORT**

for

Mini Project in PYTHON (20CSE59)

VILLAGE PROFILE

Submitted by

RANGANATH.K

1NH19CS739

5th SEM/E SEC

In partial fulfilment for the award of

the degree of

Bachelor of Engineering

in

COMPUTER SCIENCE AND ENGINEERING



**NEW HORIZON
COLLEGE OF ENGINEERING**

Autonomous College, Affiliated to VTU | Approved by AICTE New Delhi & UGC
Accredited by NAAC with 'A' Grade & Accredited by NBA



Certificate

This is to certify that the mini project work titled

VILLAGE PROFILE

*Submitted in partial fulfilment of the degree of
Bachelor of Engineering in
Computer Science and Engineering by*

RANGANATH.K

USN: 1NH19CS739

DURING

EVEN SEMESTER 2021-2022

for

COURSE CODE: 20CSE59

Signature of Reviewer

Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner

Signature with date

1. _____

2. _____

PLAGARISM CERTIFICATE

1NH19CS739

ORIGINALITY REPORT

2%

SIMILARITY INDEX

%

INTERNET SOURCES

2%

PUBLICATIONS

%

STUDENT PAPERS

PRIMARY SOURCES

1 B. N. Madhukar, Sanjay Jain. "A duality theorem for the discrete sine transform (DST)", 2015 International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), 2015 1%

Publication

2 Chet Hosmer. "Defending IoT Infrastructures with the Raspberry Pi", Springer Science and Business Media LLC, 2018 <1%

Publication

3 Aswath Suresh, Debrup Laha, Dhruv Gaba, Siddhant Bhambri. "Chapter 19 Design and Development of Innovative Pet Feeding Robot", Springer Science and Business Media LLC, 2019 <1%

Publication

4 W. David Ashley. "Foundation Db2 and Python", Springer Science and Business Media LLC, 2021 <1%

Publication

5 Daniel Duffy. "Financial Instrument Pricing Using C++ 2e + Website", Wiley, 2018 <1%

Publication

Exclude quotes Off

Exclude bibliography On

Exclude matches Off

ABSTRACT

VILLAGE PROFILE which will be very helpful project to collect the census of the village people's and it's going to collect information of every house in the village like name, age, gender and contact details etc. of the people and it also shows job offers and government schemes available for village people based on age and qualification and it will collect and store data in database and it will be implemented in Python with database and GUI like tkinter so based on these we can also collect the people are vaccinated or not. So using these we can collect, modify, store, delete and displaying the census we collected

This project will be the very helpful topic to collect a village census and in this project it will collect the information about the village based on each house, in that particular house how many people are there and their ages, contact details etc. same way it will collect and store.

SCOPE: Our project includes

- keeps the village data
- Add new house details
- Modify the house details
- Display and search the selected house details
- Government schemes and jobs

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha**, Principal, New Horizon College of Engineering, for his constant support and encouragement.

I am grateful to **Dr. Amarjeet Singh**, Dean - Academics, for his unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank **Dr. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I also express my gratitude to **Dr. Pamela Vinitha Eric**, professor, Department of Computer Science and Engineering, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

RANGANATH.K
USN: 1NH19CS739

INDEX

Sl.NO.	TITLE	PAGE NO
---------------	--------------	----------------

	ABSTRACT	I
	ACKNOWLEDGEMENT	II
	LIST OF FIGURES	VI

1. INTRODUCTION

1.1.	PROBLEM DEFINITION	1
1.2.	OBJECTIVES	1
1.3.	METHODOLOGY TO BE FOLLOWED	2
1.4.	EXPECTED OUTCOMES	2
1.5.	HARDWARE AND SOFTWARE REQUIREMENTS	3

2. FUNDAMENTALS OF PYTHON

2.1.	INTRODUCTION TO PYTHON	4
2.2.	ADVANTAGES OF PYTHON	4
2.3.	DATA TYPES	5
2.4.	PYTHON STRINGS	5
2.5.	PYTHON LISTS	5
2.6.	PYTHON TUPLES	5-6
2.7.	PYTHON SETS	6
2.8.	PYTHON DICTIONARIES	6
2.9.	FUNCTIONS IN PYTHON	6

3. FUNDAMENTALS OF TKINTER

3.1.	INTRODUCTION	7
3.2.	WIDGETS	7
3.3.	GEOMETRY MANAGERS	8

3.4. LABELS	8
3.5. BUTTONS	8

4. FUNDAMENTALS OF DBMS

4.1. INTRODUCTION	9
4.2. CHARACTERISTICS OF A DBMS	9
4.3. DATA MODEL	9
4.4. THREE - SCHEMA ARCHITECTURE	10
4.5. DBMS COMPONENT MODULES	10
4.6. ENTITY-RELATIONSHIP (ER) MODEL	11
4.7. RELATIONAL SCHEMA	11

5. FUNDAMENTALS OF SQL

5.1. INTRODUCTION	12
5.2. SQL COMMANDS	12
5.3. DATA DEFINITION LANGUAGE	12-13
5.4. DATA MANIPULATION LANGUAGE	13
5.5. DATA CONTROL LANGUAGE	13
5.6. TRANSACTION CONTROL LANGUAGE	14
5.7. DATA QUERY LANGUAGE	14

6. DESIGN

6.1. DESIGN GOALS	15
6.2. DATABASE STRUCTURE	16
6.3. GUI STRUCTURE	17

7. IMPLEMENTATION

7.1. MODULE 1 FUNCTIONALITY	18
7.2. MODULE 2 FUNCTIONALITY	19
7.3. MODULE 3 FUNCTIONALITY	20
7.4. MODULE 4 FUNCTIONALITY	21

7.5.	MODULE 5 FUNCTIONALITY	22
7.6.	MODULE 6 FUNCTIONALITY	23-24

8. RESULTS

8.1.	MAIN SCREEN	25
8.2.	USER LOGIN WINDOW	25
8.3.	ADMIN WINDOW	26-27
8.4.	SEARCHING OF DETAILS	27
8.5.	DISPLAYING DATA	28
8.6.	JOBS SCREEN	28
8.7.	SCHEMES SCREEN	29

9. CONCLUSION

REFERENCES	31
-------------------	-----------

LIST OF FIGURES

<u>Figure No</u>	<u>Figure Description</u>	<u>Page No</u>
2.3.1	Python Data types	5
3.2.1	Widgets classes	7
4.4.1	Three-Schema Architecture	10
4.5.1	DBMS Component Modules	10
4.5.1	ER Diagram	11
5.2.1	SQL Commands	12
6.1.1	Designing goals of project	15
8.1.1	Main screen	25
8.2.1	Login window	25
8.3.1	Insertion of data	26
8.3.2	Modification of data	26
8.3.3	Deletion of data	27
8.4.1	Searching of data	27
8.5.1	Displaying data	28
8.6.1	Jobs screen	28
8.7.1	Schemes screen	29

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

Develop a village profile project to collect the census information in the village level and needs to display the menu for the user and the menu contains create, delete, modify, search, display and exit options and in this program admin need to login to access the create, modify, delete options so data have secured and a display, search options are made it available publicly so anyone can access this two options.

This project will be the very helpful topic to collect a village census and in this project it will collect the information about the village based on each house, in that house how many peoples are there and their ages, contact details etc. same way it will collect and store.

Here this project will collect and stores data in database and it will be implemented in Python with database and GUI like tkinter so using these concepts we can easily store and access the data of a village profile.

1.2 OBJECTIVES

- The main objective of developing this project is get know about Python and their concepts like strings, sets, tuples, dictionaries, functions and oops concepts in python etc.
- By developing mini projects students can be capable of solving a real time problems related to computer science field.
- This projects helps to get more knowledge about Python and database management system by implementing it in own way.
- The objective of developing such a computerized program reduces the paper work and it's safe.

1.3 METHODOLOGY TO BE FOLLOWED

- In this project we need follow some methodology while dealing with this project.
- In the main screen you can select only three options for operations and exit option is made for completely closing the output screen.
- After selecting the login option you must remember the admin id and password, so after entering the admin password only it will allow doing the further process.
- In deletion process you need to enter only Name and House number and in modification you need enter old Name also
- This project is developed using the visual studio code so its show better performance in this software.

1.4 EXPECTED OUTCOMES

- In the main screen it is going to display welcome to VILLAGE PROFILE and the buttons for user selection and the menu contain the following list.
 1. Admin
 2. Display –it displays all data stored in the database.
 3. Search –display only searched house details.
 4. Government schemes-list the schemes of government
 5. Government jobs-list available jobs
 6. Exit -exit from the screen.
- In the admin option, it will ask user to enter the admin id and password
- After entering admin id and password, if it matches then it will show a further menu to admin in new window.
 1. Create new –entering the new details
 2. Modify details –modify the existing details
 3. Delete details –deleting the details from the mini census
 4. Logout –it will get back into the main screen.

1.5 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

- A laptop/Desktop
- Windows 7/8/10,linux or any operating system
- 500 MB or higher RAM
- Minimum 1 GB ROM.
- Keyboard , mouse and monitor

SOFTWARE REQUIREMENTS

- Pre-installed libraries of python
- Visual studio code software
- Notepad and Python IDLE.
- Mysql workbench software

CHAPTER 2

FUNDAMENTALS OF PYTHON

2.1 INTRODUCTION

Python is a high-level scripting language which will be used for a good sort of text processing, system administration and internet-related tasks. Unlike many similar languages, its core language is very small and easy to master, while allowing the addition of modules to perform a virtually limitless variety of tasks. Python was true object-oriented language, and is out there on a good sort of platforms. There's even a python interpreter written entirely in Java, further enhancing python's position as a superb solution for internet-based problems.

Python was developed within the early 1990's by Guido van Rossum, then at CWI in Amsterdam, and currently at CNRI in Virginia. In some ways, python grew out of a project to style a computer-oriented language which might be easy for beginners to find out , yet would be powerful enough for even advanced users. This heritage is reflected in python's small, clean syntax and therefore the thoroughness of the implementation of ideas like object-oriented programming, without eliminating the power to program during a more traditional style. So python is a superb choice as a primary programming language without sacrificing the facility and advanced capabilities that users will eventually need.

2.2 ADVANTAGES OF PYTHON

- Python is a high level language. it's a free and open language
- It is an interpreted language, as Python programs are executed by an interpreter.
- Python programs are easy to know as they need a clearly defined syntax and comparatively simple structure
- Python is case-sensitive. for instance , NUMBER and number aren't same in Python.
- Python is portable and platform independent, means it can run on various operating systems and hardware platforms.
- Python features a rich library of predefined functions.
- Python is additionally helpful in web development. Many popular web services and applications are built using Python.
- Python uses indentation for blocks and nested blocks.

2.3 DATA TYPES

Every value belongs to a selected data type in Python. Data type identifies the type of data values a variable can hold and the operations that can be performed on that data.

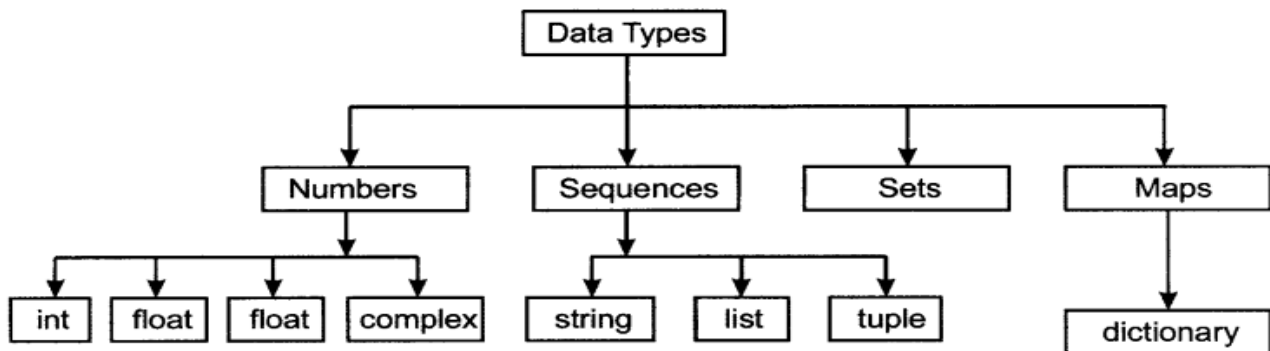


Fig 2.3.1 Python Data types

2.4 PYTHON STRINGS

String is a group of characters. These characters could also be alphabets, digits or special characters including spaces. String values are enclosed either in single quotation marks or in double quotation marks. We cannot perform numerical operations on strings, even when the string contains a numeric value, as in str2.

For example, `>>> str1 = 'Hello Friend'`

`>>> str2 = "452"`

2.5 PYTHON LISTS

List is a sequence of items separated by commas and the items are enclosed in square brackets [].

Example. #To create a list

```
>>> list1 = [5, 3.4, "New Delhi", "20C", 45]
```

```
#print the elements of the list list1
```

```
>>> print(list1) [5, 3.4, 'New Delhi', '20C', 45]
```

2.6 PYTHON TUPLES

Tuple is a sequence of items separated by commas and items are enclosed in parenthesis ().

This is unlike list, where values are enclosed in brackets []. Once created, we cannot change the tuple.

Example. #create a tuple tuple1

```
>>> tuple1 = (10, 20, "Apple", 3.4, 'a')
# print the elements of the tuple tuple1
>>> print(tuple1) (10, 20, "Apple", 3.4, 'a')
```

2.7 PYTHON SETS

Set is an unordered collection of items separated by commas and the items are enclosed in curly brackets { }. A set is similar to list, except that it cannot have duplicate entries. Once created, elements of a set cannot be changed.

Example. #create a set

```
>>> set1 = {10,20,3.14,"New Delhi"}
>>> print(set1) {10, 20, 3.14, "New Delhi"}
```

2.8 PYTHON DICTIONARIES

Dictionary in Python holds data items in key-value pairs. Items in an exceedingly dictionary are enclosed in curly brackets . Dictionaries permit faster access to data. Every secret is separated from its value employing a colon (:) sign. The key : value pairs of a dictionary may be accessed using the key. The keys are usually strings and their values are often any data type.

Example. #create a dictionary

```
>>> dict1 = {'Fruit':'Apple', 'Climate':'Cold', 'Price(kg)':120}
>>> print(dict1) {'Fruit': 'Apple', 'Climate': 'Cold', 'Price(kg)': 120}
```

2.9 FUNCTIONS IN PYTHON

Functions are a convenient way to divide your code into useful blocks, allowing us to order our code, make it more readable, reuse it and save some time. Also functions are a key way to define interfaces so programmers can share their code.

Syntax: def function_name(parameters):

```
    """docstring"""
    statement(s)
    function_name()
```

CHAPTER 3

FUNDAMENTALS OF TKINTER

3.1 INTRODUCTION

The tkinter package could be a thin object-oriented layer on top of Tcl/Tk. To use tkinter, you don't have to write Tcl code, but you'll must consult the Tk documentation, and sometimes the Tcl documentation. tkinter could be a set of wrappers that implement the Tk widgets as Python classes.

The primary GUI toolkit we are going to be using is Tk, Python's default GUI. We'll access Tk from its Python interface called Tkinter (short for "Tk interface").

Tk isn't the newest and greatest, nor does it have the foremost robust set of GUI building blocks, but it's fairly simple to use, and with it, you'll be able to build GUIs that run on most platforms

3.2 WIDGETS

Widget is an element of Graphical User Interface (GUI) that displays/illustrates information or gives a way for the user to interact with the OS. In Tkinter , Widgets are objects instances of classes that represent buttons, frames, and so on.

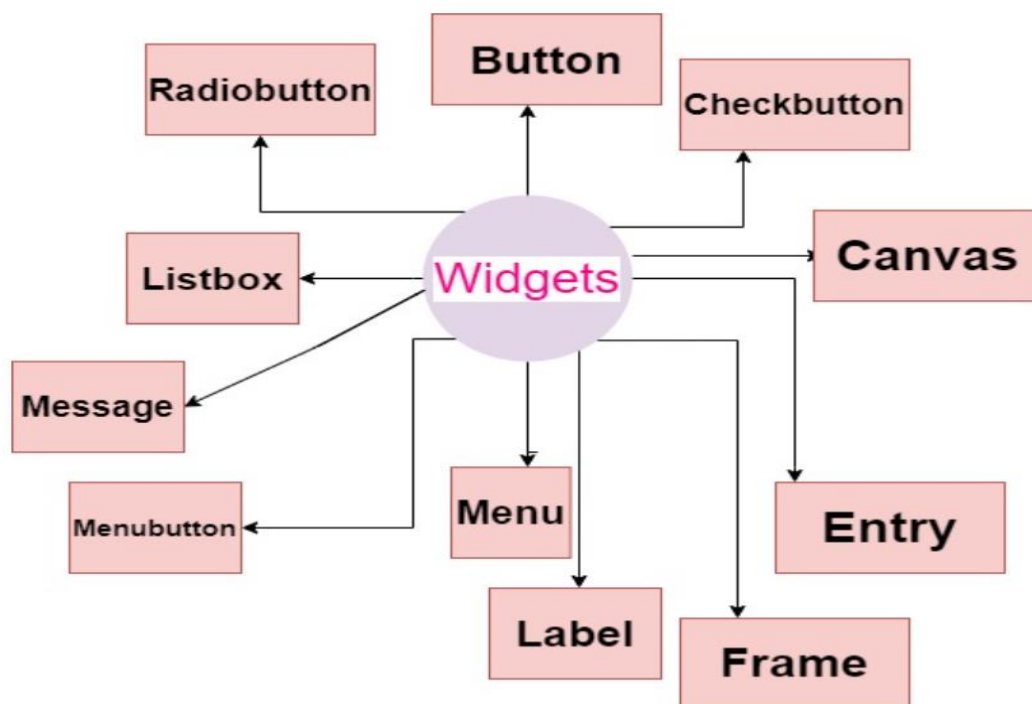


Fig 3.2.1 Widgets classes.

3.3 GEOMETRY MANAGERS

In order to arrange or arrange or place all the widgets within the parent window, Tkinter provides us the geometric configuration of the widgets. The GUI Application Layout is principally controlled by Geometric Managers of Tkinter. It's important to notice here that every window and frame your application is allowed to use just one geometry manager. Also, different frames can use different geometry managers, whether or not they're already assigned to a frame or window using another geometry manager.

Method	Description
pack()	The Pack geometry manager packs widgets in rows or columns.
grid()	The Grid geometry manager puts the widgets in a 2-dimensional table. The master widget is split into a number of rows and columns, and each "cell" in the resulting table can hold a widget.
place()	The Place geometry manager is the simplest of the three general geometry managers provided in Tkinter. It allows you explicitly set the position and size of a window, either in absolute terms, or relative to another window.

Table 3.3.1 Geometry Managers.

3.4 LABELS

The label is used to display fields where you can place textual content or images. The textual content displayed by means of this widget can be updated at any time you want.

Syntax

```
w =Label ( a, option)
```

Parameters

- a – This represents the parent window of the computer.
- option – The list of most used options for this widget.

3.5 BUTTONS

Button widgets represent a clickable item in the applications. Typically, you use a text or an image to display the action that will be performed when clicked.

Buttons can display text in a single font. However, the text can span multiple lines. On top of that, you can make one of the characters underline to mark a keyboard shortcut.

CHAPTER 4

FUNDAMENTALS OF DBMS

4.1 INTRODUCTION

The developer builds an application or software, but Software needs Data to perform day to day operations and analytics over-processed data and data are some things that's driving the business nowadays to excel in their respective areas of operations. So as a developer, we'd like a management System (DBMS) where we will create, update, delete, administer and moreover to try to to an analysis of the information. A direction System (DBMS) may be a software package designed to define, manipulate, retrieve and manage data in a very database. allow us to examine this introduction to the DBMS article thoroughly.

4.2 CHARACTERISTICS OF A DBMS

- Provides security and removes redundancy
- Self-describing nature of a database system
- Insulation between programs and data abstraction
- Support of multiple views of the information
- Sharing of information and multiuser transaction processing
- Database Management Software allows entities and relations among them to create tables
- It follows the ACID concept (Atomicity, Consistency, Isolation, and Durability).
- DBMS supports multi-user environment that enables users to access and manipulate data in parallel.

4.3 DATA MODEL

A database model shows the logical structure of a database, including the relationships and constraints that determine how data are often stored and accessed. Most data models will be represented by an accompanying database diagram.

- Hierarchical database model
- Relational model
- Network model
- Object-oriented database model
- Entity-relationship model
- Entity-attribute-value model

4.4 THREE - SCHEMA ARCHITECTURE

The three schema architecture is additionally called ANSI/SPARC architecture or threelevel architecture. This framework is employed to explain the structure of a particular database system. The three schema architecture is additionally want to separate the user applications and physical database. The three schema architecture contains three-levels. It breaks the database down into three different categories.

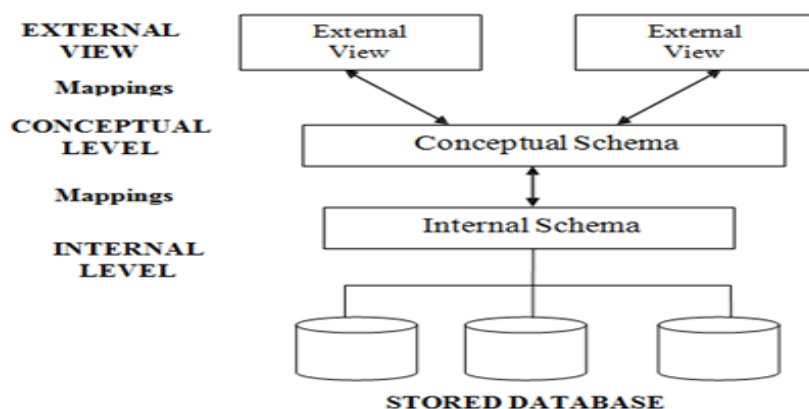


Fig 4.4.1 THREE - SCHEMA ARCHITECTURE.

4.5 DBMS COMPONENT MODULES

DBMS have several components, each performing very significant tasks in the database management system environment. Below is a list of components within the database and its environment.

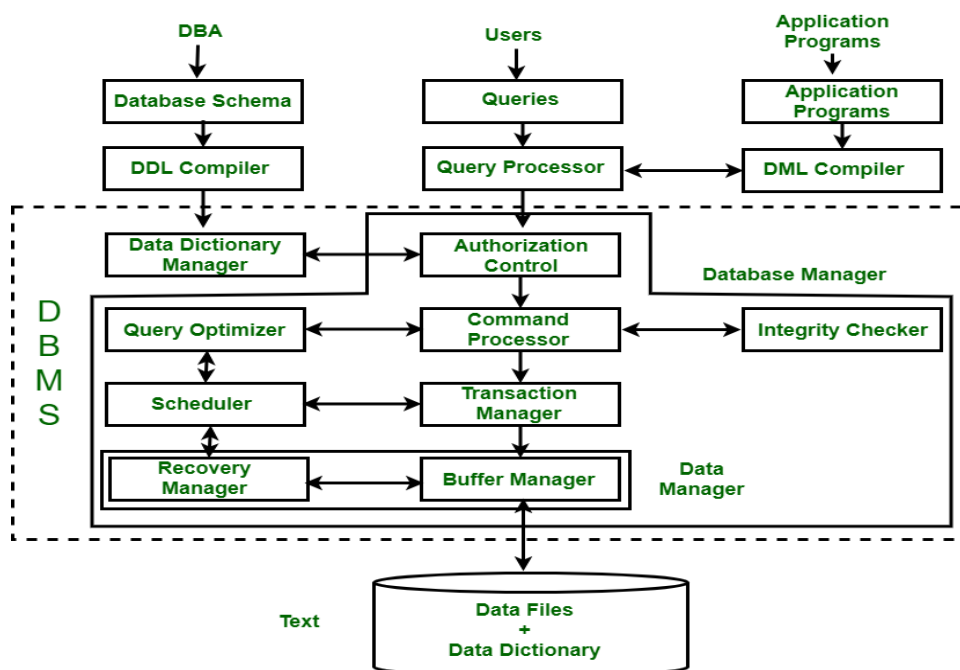


Fig 4.5.1 DBMS Component Modules.

4.6 ENTITY-RELATIONSHIP (ER) MODEL

An Entity-relationship model (ER model) describes the structure of a database with the assistance of a diagram, which is thought as Entity Relationship Diagram (ER Diagram). An ER model could be a design or blueprint of a database that may later be implemented as a database. the most components of E-R model are: entity set and relationship set

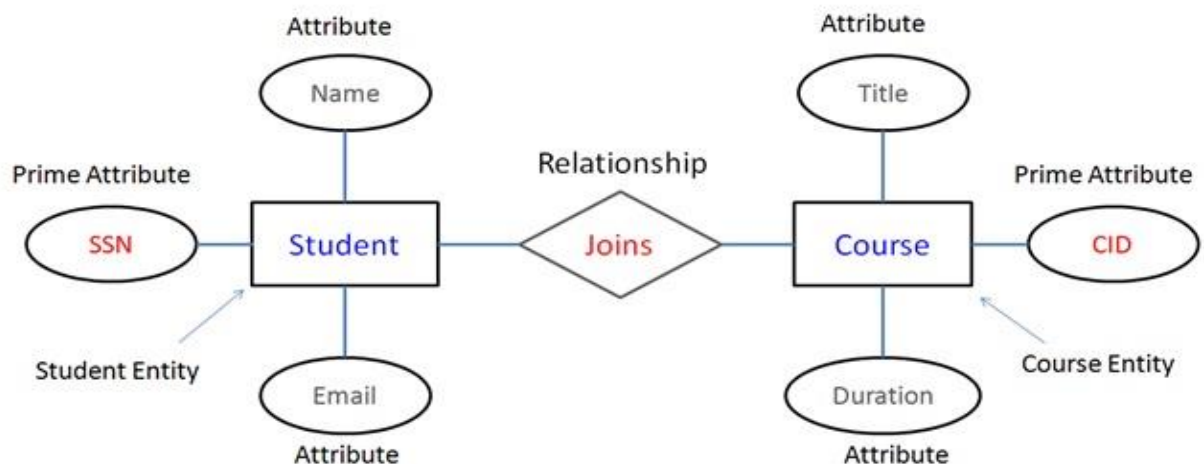


Fig 4.5.1 ER Diagram.

4.7 RELATIONAL SCHEMA

A relational schema could be a blueprint employed in database design to represent the information to be entered into the database and describe how that data is structured in tables (called relations in relational schemas). The schema describes how those tables relate to every other.

INFORMAL DESIGN GUIDELINES FOR RELATIONAL SCHEMA

- Semantics of the Attributes
- Reducing the Redundant Value in Tuples.
- Reducing Null values in Tuples.
- Disallowing spurious Tuples.

CHAPTER 5

FUNDAMENTALS OF SQL

5.1 INTRODUCTION

SQL stands for Structured command language. SQL is employed to form, remove, alter the database and database objects in a very management system and to store, retrieve, update the info in a very database. SQL could be a standard language for creating, accessing, and manipulating direction system. SQL works for all modern electronic information service management systems, like SQL Server, Oracle, MySQL, etc.

5.2 SQL COMMANDS

SQL commands are instructions. it's accustomed communicate with the database. it's also accustomed perform specific tasks, functions, and queries of information. SQL can perform various tasks like create a table, add data to tables, drop the table, modify the table, set permission for users.

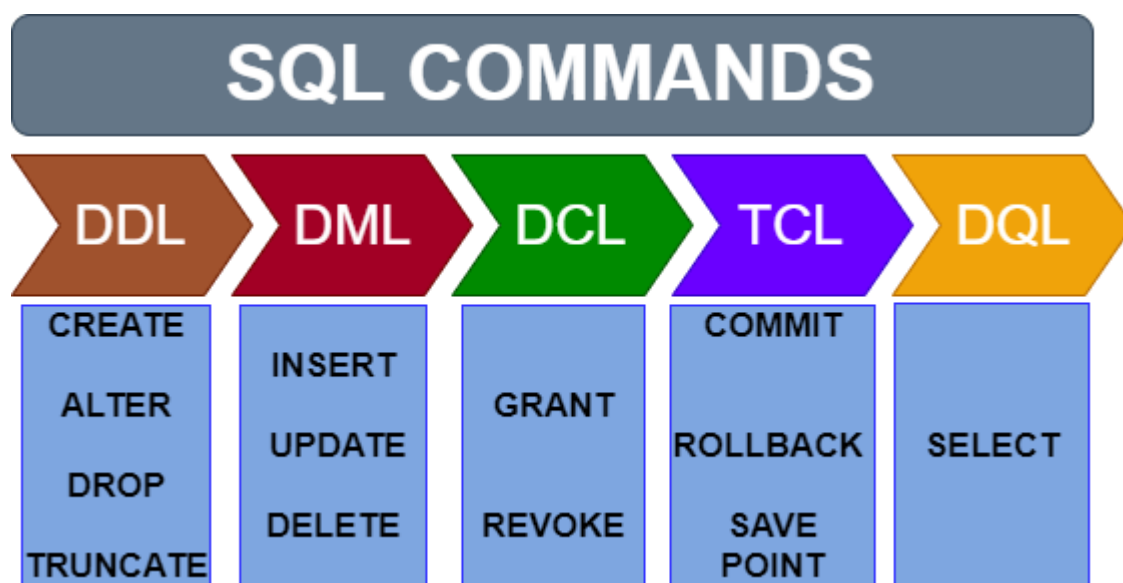


Fig 5.2.1 SQL Commands.

5.3 DATA DEFINITION LANGUAGE

DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc. All the command of DDL are auto-committed meaning it permanently save all the changes within the database.

a. CREATE: It is employed to form a brand new table within the database.

```
CREATE TABLE TABLE_NAME (COLUMN_NAME DATATYPES[,...]);
```

b. DROP: it's accustomed delete both the structure and record stored within the table.

```
DROP TABLE table_name;
```

c. ALTER: It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.

```
ALTER TABLE table_name ADD column_name COLUMN-definition;
```

d. TRUNCATE: It is used to delete all the rows from the table and free the space containing the table.

```
TRUNCATE TABLE table_name;
```

5.4 DATA MANIPULATION LANGUAGE

DML commands are used to modify the database. It is responsible for all form of changes in the database. The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

a. INSERT: The INSERT statement is a SQL query. It is used to insert data into the row of a table.

```
INSERT INTO TABLE_NAME  
(col1, col2, col3,... col N)  
VALUES (value1, value2, value3, .... valueN);
```

b. UPDATE: This command is used to update or modify the value of a column in the table.

```
UPDATE table_name SET [column_name1= value1,...column_nameN = valueN] [WHERE CONDITION]
```

c. DELETE: It is used to remove one or more row from a table.

```
DELETE FROM table_name [WHERE condition];
```

5.5 DATA CONTROL LANGUAGE

DCL commands are used to grant and take back authority from any database user.

a. Grant: It is used to give user access privileges to a database.

```
GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER, ANOTHER_USER;
```

b. Revoke: It is used to take back permissions from the user.

```
REVOKE SELECT, UPDATE ON MY_TABLE FROM USER1, USER2;
```

5.6 TRANSACTION CONTROL LANGUAGE

TCL commands can only use with DML commands like INSERT, DELETE and UPDATE only. These operations are automatically committed in the database that's why they cannot be used while creating tables or dropping them.

a. Commit: Commit command is used to save all the transactions to the database.

```
DELETE FROM CUSTOMERS  
WHERE AGE = 25;  
COMMIT;
```

b. Rollback: Rollback command is used to undo transactions that have not already been saved to the database.

```
DELETE FROM CUSTOMERS  
WHERE AGE = 25;  
ROLLBACK;
```

c. SAVEPOINT: It is used to roll the transaction back to a certain point without rolling back the entire transaction.

```
SAVEPOINT SAVEPOINT_NAME;
```

5.7 DATA QUERY LANGUAGE

DQL is used to fetch the data from the database.

a. SELECT: This is the same as the projection operation of relational algebra. It is used to select the attribute based on the condition described by WHERE clause.

```
SELECT column_list FROM table-name  
[WHERE clause]  
[GROUP BY clause]  
[HAVING clause]  
[ORDER BY clause];
```

CHAPTER 6

DESIGN

6.1 DESIGN GOALS

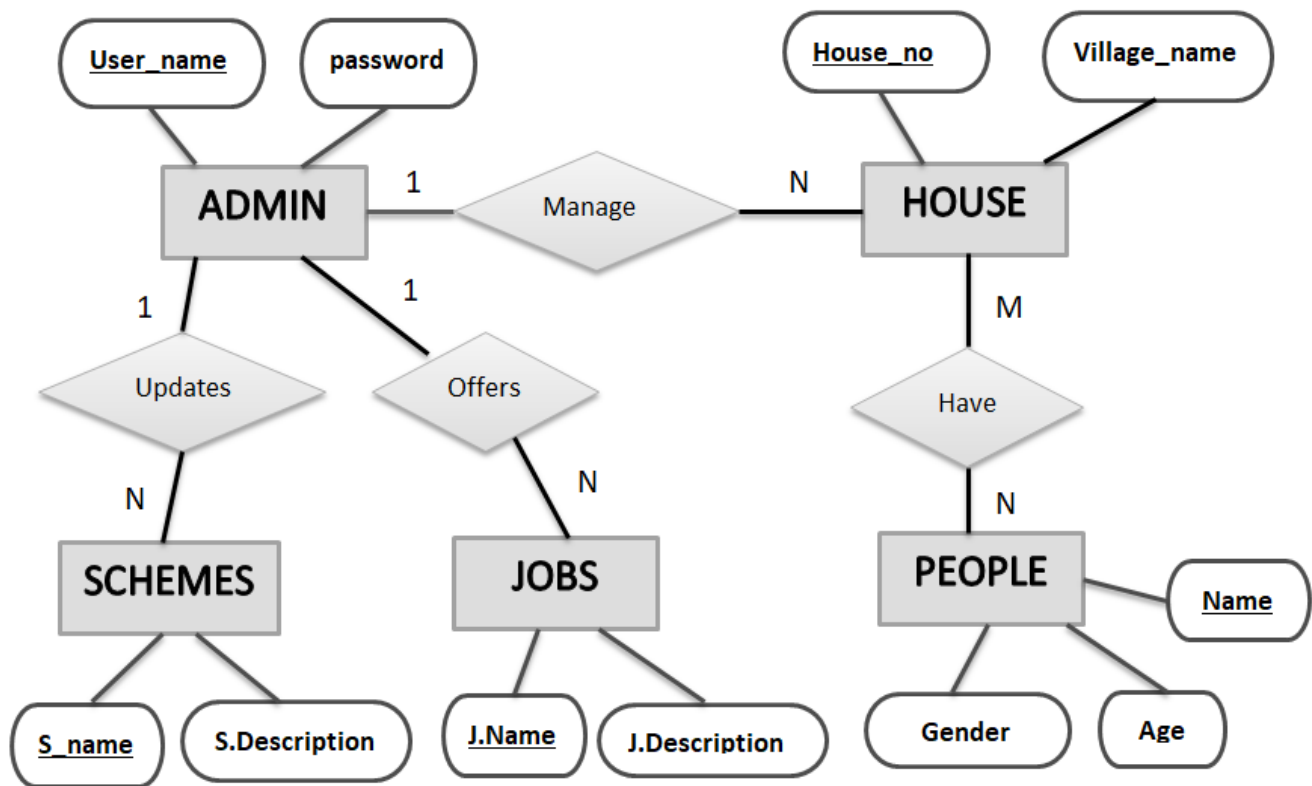
- Allow users to easily see and make it user friendly.
- Give users an easy way to back out (i.e. change & error recovery).
- Allow user to complete their task without being distracted by software or losing train of thought, which is while they are reading and typing.
- Everything will be made it by menu selection for easy understand.
- Give users access to information they need to complete their task.
- Wishing the user whenever they open the screen.
- Collects the details one by one without overloading.



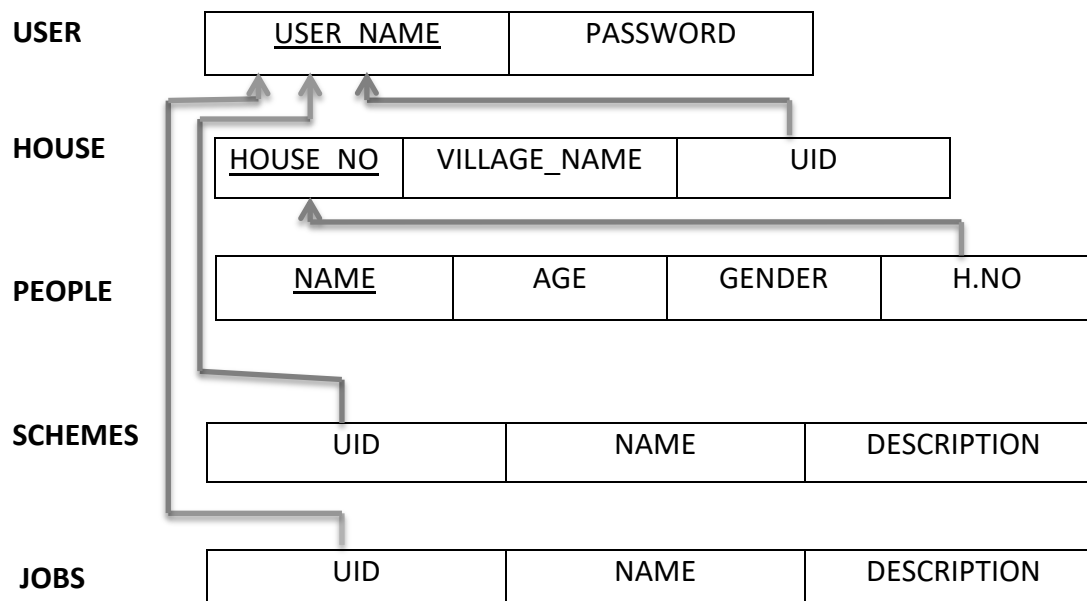
Fig 6.1.1 Designing goals of project.

6.2 DATABASE STRUCTURE

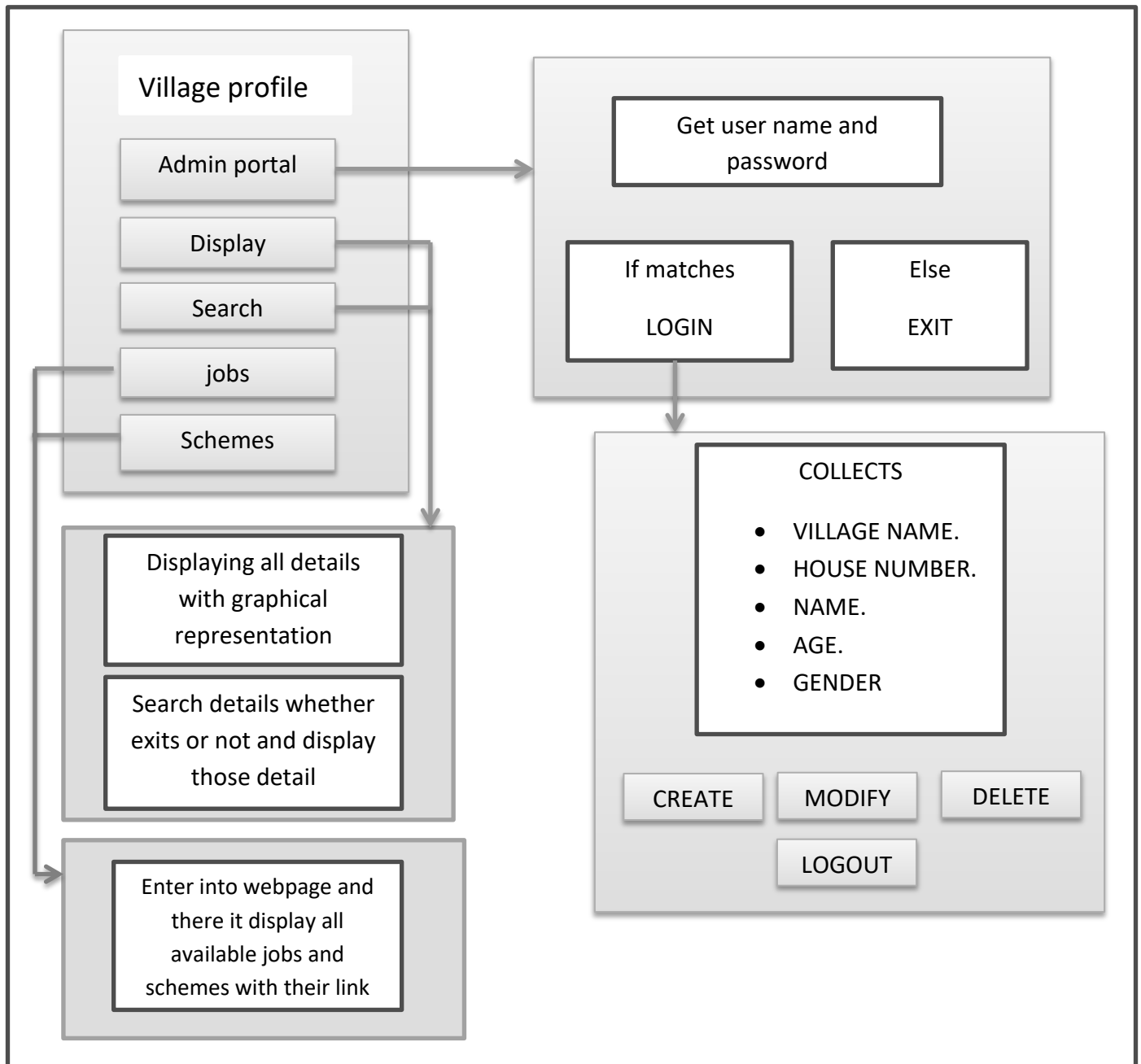
Entity Relationship Diagram (ER Diagram):



RELATIONAL SCHEMA:



6.3 GUI STRUCTURE



CHAPTER 7

IMPLEMENTATION

7.1 MODULE 1 FUNCTIONALITY

➤ Creating Main window.

```
def main():
    root = Tk()
    root.title("VILLAGE PROFILE")
    root.geometry("300x250")
    bg1= PhotoImage(file="12.png")
    background = Label(root, image=bg1)
    background.pack()

    Label(root, text="VILLAGE PROFILE",font=("Arial", 15, "bold"), bg="#00376b", fg="#FFFCF9").place(x=60, y=10)
    Button(root, text="Admin portal", command=login,bd=3, height = 1, width = 13).place(x=100, y=50)
    Button(root, text="Display", command=disp,bd=3, height = 1, width = 13).place(x=100, y=80)
    Button(root, text="Search", command=ser,bd=3, height = 1, width = 13).place(x=100, y=110)
    Button(root, text="Government schemes",bd=3, command=schem, height = 1, width = 20).place(x=75, y=140)
    Button(root, text="Government Jobs",bd=3, command=job, height = 1, width = 20).place(x=75, y=170)
    Button(root, text="EXIT", command=exit,bd=3, height = 1, width = 13).place(x=100, y=200)
    root.mainloop()
main()
```

➤ Connecting with Database and website

```
import mysql.connector
from tkinter import *
from tkinter import ttk
import tkinter as tk
from tkinter import messagebox
import webbrowser

mysqlldb = mysql.connector.connect(host="localhost", user="root", password="ranga143", database="village")
mycursor = mysqlldb.cursor()

def schem():
    webbrowser.open("https://laxeg57714.wixsite.com/my-site-1")

def job():
    webbrowser.open("https://laxeg57714.wixsite.com/my-site")
```

7.2 MODULE 2 FUNCTIONALITY

➤ Login window

```
def login():
    global r1
    root = Toplevel()
    r1=root
    root.title("VILLAGE PROFILE")
    root.geometry("300x200")
    bg1= PhotoImage(file="14.png")
    background = Label(root, image=bg1)
    background.pack()
    global e1
    global e2

    Label(root, text="VILLAGE PROFILE",font=("Arial", 15, "bold"), bg="#00376b", fg="#FFFCF9").place(x=60, y=10)
    Label(root, text="UserName").place(x=30, y=60)
    Label(root, text="Password").place(x=30, y=90)

    e1 = Entry(root,bd=4)
    e1.place(x=100, y=60)

    e2 = Entry(root,bd=4)
    e2.place(x=100, y=90)
    e2.config(show="*")
    Button(root, text="Login", command=Ok,height = 2, width = 13,bg='#4a7abc',fg='yellow',
        | activebackground='green',activeforeground='white').place(x=120, y=120)

    root.mainloop()
```

➤ Password matching.

```
def Ok():
    global uname
    uname = e1.get()
    password = e2.get()
    sql = "select * from user where uname = %s and password = %s"
    mycursor.execute(sql, [(uname), (password)])
    results = mycursor.fetchall()

    if results:
        r1.destroy()
        messagebox.showinfo("", "Login Successfull")
        Admin()
    else :
        r1.destroy()
        messagebox.showinfo("", "Incorrent Username and Password")
```

7.3 MODULE 3 FUNCTIONALITY

➤ Admin window designing.

```
def Admin():
    root = Toplevel()
    root.title("VILLAGE PROFILE")
    root.geometry("300x330")
    bg1= PhotoImage(file="12.png")
    background = Label(root, image=bg1)
    background.pack()

    Label(root, text="VILLAGE PROFILE",font=("Arial", 15, "bold"), bg="#00376b", fg="#FFFCF9").place(x=60, y=10)
    Label(root, text="VILLAGE NAME").place(x=10, y=60)
    Label(root, text="HOUSE NUMBER").place(x=10, y=83)
    Label(root, text="NAME").place(x=10, y=106)
    Label(root, text="AGE").place(x=10, y=129)
    Label(root, text="GENDER").place(x=10, y=152)
    Label(root, text="OLD NAME(ONLY FOR MODIFY)").place(x=10, y=175)
    Label(root, text="NOTE:For deletion enter only Name and House number").place(x=0, y=260)
    Label(root, text="**For modiication of village name enter only").place(x=21, y=280)
    Label(root, text="village name and House number").place(x=35, y=298)

global vn,hn,na,ag,gen,on
vn = Entry(root,bd=2)
vn.place(x=110, y=60)
hn = Entry(root,bd=2)
hn.place(x=110, y=83)
na = Entry(root,bd=2)
na.place(x=110, y=106)
ag = Entry(root,bd=2)
ag.place(x=110, y=129)
gen = Entry(root,bd=2)
gen.place(x=110, y=152)
on = Entry(root,bd=2)
on.place(x=30, y=199)

def clr():
    vn.delete(0,END)
    hn.delete(0,END)
    na.delete(0,END)
    ag.delete(0,END)
    gen.delete(0,END)
    on.delete(0,END)
    Button(root, text="Create", command=lambda:[create(),clr()],height = 1, bd=3,width = 8,bg='sky blue',relief='raised').place(x=10, y=225)
    Button(root, text="Modify", command=lambda:[modify(),clr()], height = 1,bd=3, width = 8,relief='raised',bg="pink").place(x=80, y=225)
    Button(root, text="Delete", command=lambda:[delete(),clr()], height = 1,bd=3, width = 8,relief='raised',bg="wheat2").place(x=150, y=225)
    Button(root, text="Logout", command=root.destroy, height = 1, width = 8,bd=3,bg='red2',relief='raised').place(x=220, y=225)

root.mainloop()
```

7.4 MODULE 4 FUNCTIONALITY

➤ Data insertion into database

```
def create():
    vname=vn.get()
    hono=hn.get()
    nam=na.get()
    ages=ag.get()
    gend=gen.get()

    try:
        sql = "select houseno from house where houseno=%s"
        mycursor.execute(sql,[(hono)])
        results = mycursor.fetchall()

        if results:
            print()
        else:
            sql1 = "INSERT INTO house (houseno,vilname,uid) VALUES (%s, %s,%s)"
            val1 = (hono,vname,uname)
            mycursor.execute(sql1, val1)
            mysqldb.commit()

            sql2 = "INSERT IGNORE INTO people (name,age,gender,hno) VALUES (%s, %s,%s,%s)"
            val2=(nam,ages,gend,hono)
            mycursor.execute(sql2, val2)
            mysqldb.commit()
    except mysql.connector.Error as err:
        print("Something went wrong: {}".format(err))
    messagebox.showinfo("", "Inserted Successfully")
```

➤ Modifying data in database

```
def modify():
    vname=vn.get()
    hono=hn.get()
    nam=na.get()
    ages=ag.get()
    gend=gen.get()
    ona=on.get()
    sql1="UPDATE house SET vilname= %s WHERE houseno=%s"
    mycursor.execute(sql1,[(vname),(hono)])
    mysqldb.commit()

    sql2="UPDATE people SET name=%s,age=%s,gender=%s WHERE hno=%s AND name=%s"
    mycursor.execute(sql2,[(nam),(ages),(gend),(hono),(ona)])
    mysqldb.commit()
    messagebox.showinfo("", "Updated Successfully")
```

7.5 MODULE 5 FUNCTIONALITY

➤ Searching window

```
def ser():
    root = Toplevel()
    root.title("VILLAGE PROFILE")
    root.geometry("500x300")
    Label(root, text="VILLAGE PROFILE",font=("Arial", 15, "bold"), bg="#00376b", fg="#FFCF9").pack(pady=8)

    Label(root, text="ENTER HOUSE NUMBER").pack()
    hn1 = Entry(root,bd=2)
    hn1.pack()

    def View():
        hn2=hn1.get()
        sql = "SELECT vilname,houseno,name ,age,gender FROM village.house, village.people where houseno= hno and hno=%s"
        mycursor.execute(sql,[hn2])
        rows = mycursor.fetchall()
        for row in rows:
            tree.insert("", tk.END, values=row)

    style=ttk.Style(root)
    style.theme_use("clam")
    style.configure("Treeview",background="thistle1",foreground="black")
    style.map('Treeview',background=[("selected","green")])

    style=ttk.Style(root)
    style.theme_use("clam")
    style.configure("Treeview",background="thistle1",foreground="black")
    style.map('Treeview',background=[("selected","green")])

    tree =ttk.Treeview(root, column=("c1", "c2", "c3","c4","c5"), show='headings',height=5)
    tree.column("#1", anchor=tk.CENTER, width=100)
    tree.heading("#1", text="Village name")
    tree.column("#2", anchor=tk.CENTER, width=100)
    tree.heading("#2", text="House.no")
    tree.column("#3", anchor=tk.CENTER, width=100)
    tree.heading("#3", text="Name")
    tree.column("#4", anchor=tk.CENTER, width=100)
    tree.heading("#4", text="Age")
    tree.column("#5", anchor=tk.CENTER, width=100)
    tree.heading("#5", text="Gender")
    tree.pack(pady=12)

    Button(root, text="Search", command=View,bd=3, height = 1, width = 13).pack(pady=12)
    root.mainloop()
```

➤ Deleting data in database

```
def delete():
    hono=hn.get()
    nam=na.get()
    sql1="DELETE FROM people WHERE name=%s AND hno=%s"
    mycursor.execute(sql1,[(nam),(hono)])
    mysqldb.commit()
    messagebox.showinfo("", "Deleted Successfully")
```

7.6 MODULE 6 FUNCTIONALITY

- Display window design.

```
def disp():
    root = Tk()
    root.title("VILLAGE PROFILE")
    root.geometry("520x370")
    Label(root, text="VILLAGE PROFILE",font=("Arial", 15, "bold"), bg="#00376b", fg="#FFFCF9").pack(pady=6)

    style=ttk.Style(root)
    style.theme_use("clam")
    style.configure("Treeview",background="cyan2",foreground="black",rowheight=25,filebackground="silver")
    style.map('Treeview',background=[("selected","green")])

    tree =ttk.Treeview(root, column=("c1", "c2", "c3","c4","c5"), show='headings',height='7')
    tree.column("#1", anchor=tk.CENTER, width=100)
    tree.heading("#1", text="Village name")
    tree.column("#2", anchor=tk.CENTER, width=100)
    tree.heading("#2", text="House.no")
    tree.column("#3", anchor=tk.CENTER, width=100)
    tree.heading("#3", text="Name")
    tree.column("#4", anchor=tk.CENTER, width=100)
    tree.heading("#4", text="Age")
    tree.column("#5", anchor=tk.CENTER, width=100)
    tree.heading("#5", text="Gender")
    tree.pack()
    tree.place(x=7,y=45)
```

- SQL queries for display window.

```
sql = "SELECT vilname,houseno,name ,age,gender FROM village.house, village.people where houseno= hno order by houseno asc"
mycursor.execute(sql)
rows = mycursor.fetchall()
for row in rows:
    tree.insert("", tk.END, values=row)

T = Text(root, height = 6, width = 33,font =("Times", 11),bg ="azure")
T.place(x=150,y=255)
sql = "SELECT count(*)FROM village.house;"
mycursor.execute(sql)
r1 = mycursor.fetchone()[0]
T.insert(tk.END,"*Total Number of Houses:")
T.insert(tk.END,r1)
```



```
sql = "SELECT count(*) FROM village.people"
mycursor.execute(sql)
r2 = mycursor.fetchone()[0]
T.insert(tk.END, "\n*Total Number of peoples:")
T.insert(tk.END, r2)

sql = "SELECT count(*)FROM village.people where gender like 'male%'"
mycursor.execute(sql)
r3 = mycursor.fetchone()[0]
T.insert(tk.END, "\n*Number of Males:")
T.insert(tk.END, r3)

sql = "SELECT count(*)FROM village.people where gender like 'female%'"
mycursor.execute(sql)
r4 = mycursor.fetchone()[0]
T.insert(tk.END, "\n*Number of Females:")
T.insert(tk.END, r4)

sql = "SELECT count(*) FROM village.people where age<=18"
mycursor.execute(sql)
r5 = mycursor.fetchone()[0]
T.insert(tk.END, "\n*Number of Below 18 aged people:")
T.insert(tk.END, r5)

sql = "SELECT count(*) FROM village.people where age>18"
mycursor.execute(sql)
r6 = mycursor.fetchone()[0]
T.insert(tk.END, "\n*Number of Above 18 aged people:")
T.insert(tk.END, r6)
root.mainloop()
```

CHAPTER 8

RESULTS

8.1 Main screen



Fig 8.1.1 Main screen

8.2 User Login window

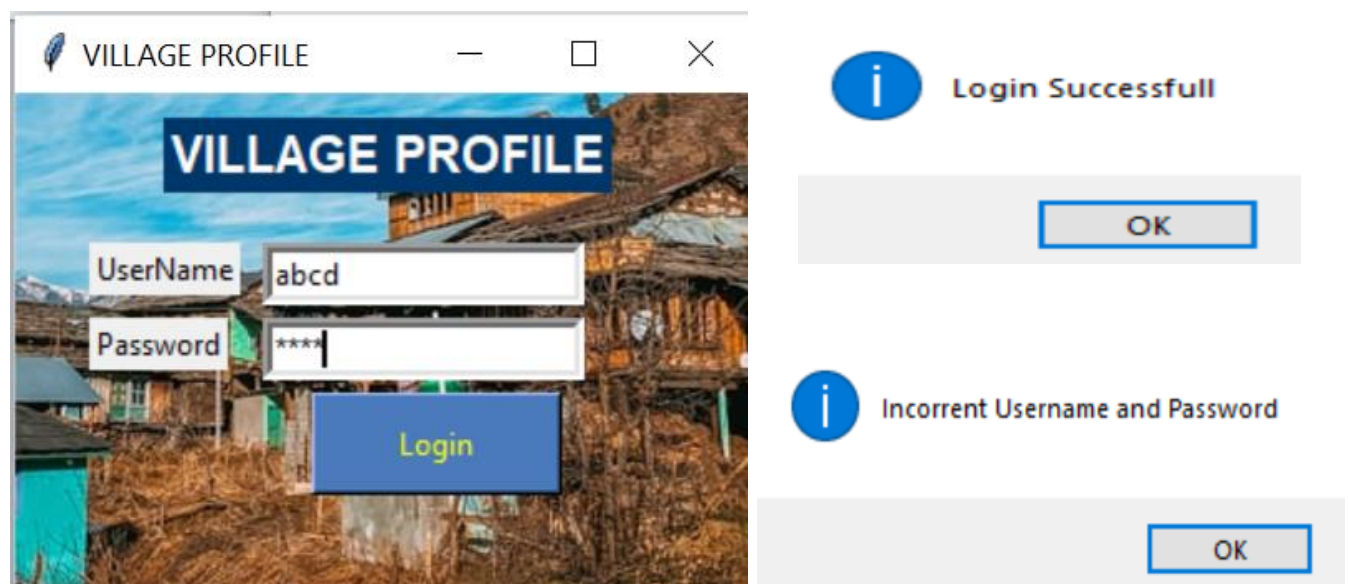


Fig 8.2.1 Login window

8.3 Admin window

➤ Creation of data



The screenshot shows a window titled "VILLAGE PROFILE" with a background image of a village. The form contains the following fields and values:

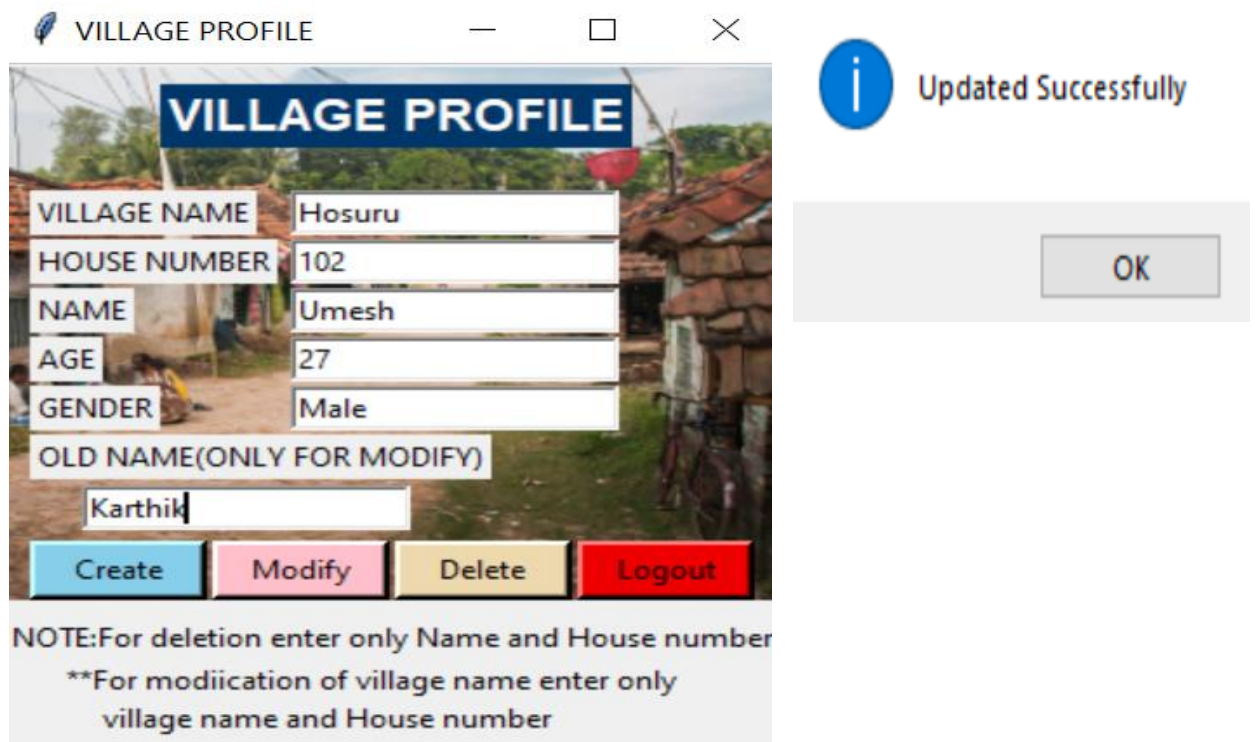
Field	Value
VILLAGE NAME	Hosuru
HOUSE NUMBER	102
NAME	Kiran
AGE	26
GENDER	Male
OLD NAME(ONLY FOR MODIFY)	

Below the form are four buttons: "Create" (blue), "Modify" (pink), "Delete" (yellow), and "Logout" (red). At the bottom, there is a note:

NOTE:For deletion enter only Name and House number
**For modiication of village name enter only village name and House number

Fig 8.3.1 Insertion of data

➤ Modification of data



The screenshot shows the "VILLAGE PROFILE" window with the "Modify" form. The fields and values are:

Field	Value
VILLAGE NAME	Hosuru
HOUSE NUMBER	102
NAME	Umesh
AGE	27
GENDER	Male
OLD NAME(ONLY FOR MODIFY)	Karthik

The buttons "Create", "Modify", "Delete", and "Logout" are visible. A success message "Updated Successfully" with an information icon is displayed on the right, and an "OK" button is below it. The note at the bottom is the same as in Fig 8.3.1.

Fig 8.3.2 Modification of data

➤ Deletion of data



VILLAGE PROFILE

VILLAGE NAME

HOUSE NUMBER

NAME

AGE

GENDER

OLD NAME(ONLY FOR MODIFY)

Create Modify Delete Logout

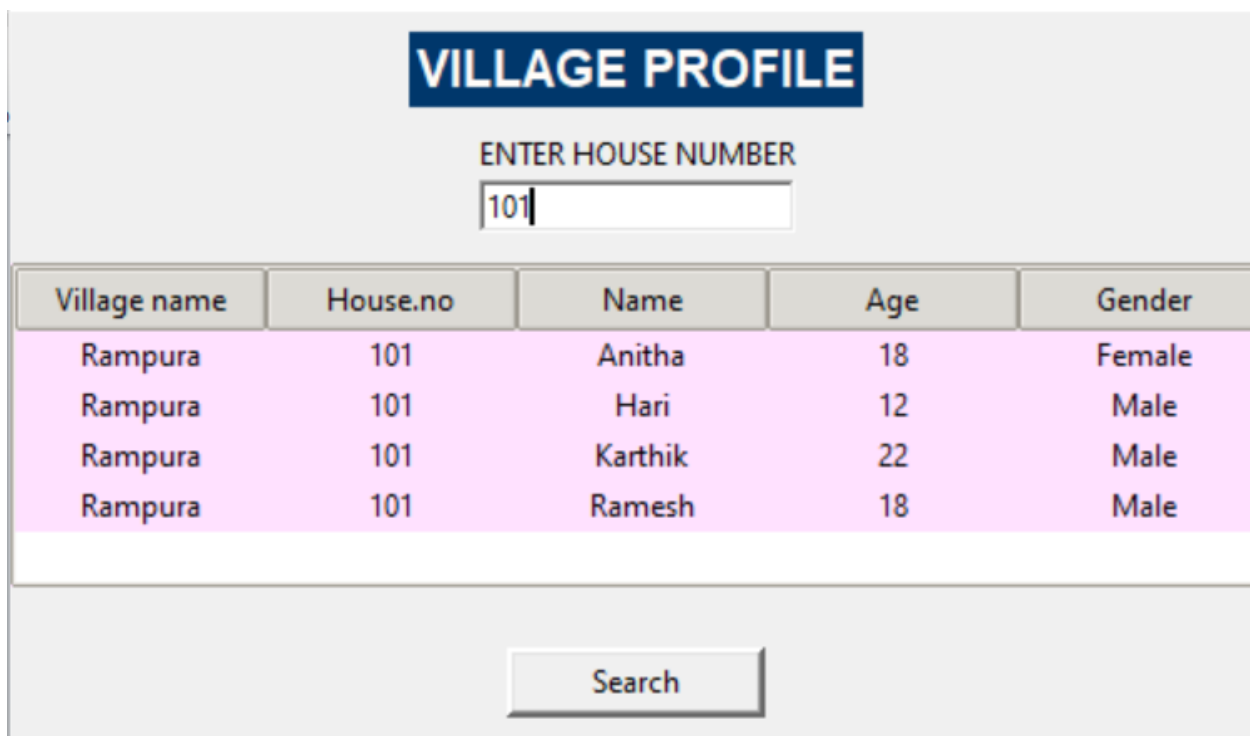
NOTE:For deletion enter only Name and House number
**For modification of village name enter only village name and House number

Deleted Successfully

OK

Fig 8.3.3 Deletion of data

8.4 Searching of details



VILLAGE PROFILE

ENTER HOUSE NUMBER

Village name	House.no	Name	Age	Gender
Rampura	101	Anitha	18	Female
Rampura	101	Hari	12	Male
Rampura	101	Karthik	22	Male
Rampura	101	Ramesh	18	Male

Search

Fig 8.4.1 Deletion of data

8.5 Displaying data

VILLAGE PROFILE				
Village name	House.no	Name	Age	Gender
Rampura	101	Anitha	18	Female
Rampura	101	Hari	12	Male
Rampura	101	Karthik	22	Male
Rampura	101	Ramesh	18	Male
Hosuru	102	Gururaj	45	Male
Hosuru	102	harish	48	Male
Hosuru	102	Ragu	25	Male

*Total Number of Houses:7
*Total Number of peoples:15
*Number of Males:12
*Number of Females:3
*Number of Below 18 aged people:3
*Number of Above 18 aged people:12

Fig 8.5.1 Displaying data

8.6 Jobs screen

VILLAGE PROFILE					Home	Gallery
JOBS						
Search						
Organization	Job Details	Education	Last Date	Link to Apply		
BFUHS	Staff Nurse, ECG Technician , Medical Physicist - 36 Posts	B.sc Nursing , Diploma , GNM , Intermediate (10+2) , M.Sc , Post Graduation	11 - Jan - 2022	https://allgovernmentjobs.in/bfuhs-recruitment		
E-courts	Assistant, Stenographer, Peon - 67 Posts	8th , Any Degree	31 - Jan - 2022	https://allgovernmentjobs.in/e-courts-recruitment		
SCTIMST	Research Nurse - 03 Posts	B.sc Nursing , GNM	06 - Jan - 2022	https://allgovernmentjobs.in/sctimst-recruitment		
BHEL	Engineer, Supervisor - 36 Posts	Engineer, Supervisor - 36 Posts	11 - Jan - 2022	https://allgovernmentjobs.in/bhel-recruitment		
Directorate of Education	Jr. Engineer, AE, Executive Engineer - 38 Posts	Civil Engineering , Diploma , Engineering	10 - Jan - 2022	https://allgovernmentjobs.in/directorate-of-education-recruitment		

Fig 8.6.1 Jobs screen

8.7 Schemes screen




VILLAGE PROFILE			Home	Gallery
<u>SCHEMES</u>				
			Search	
S No. ↑↓	Scheme Name ↑↓	Application / Registration Portal ↑↓		
1	Agri Clinics and Agri Business Centres ACABC	http://acabcmis.gov.in/ApplicantReg.aspx	  	
2	APEDA	https://apeda.gov.in/apedawebsite/		
3	Fertilizer Subsidy Scheme	https://fert.nic.in/dbt		
4	Integrated scheme on agriculture cooperation	https://ncui.coop/registration-form/		
5	Livestock Health and Diseases Control	https://kisansuvidha.gov.in/animal-husbandry		
6	PM KISAN	https://pmkisan.gov.in/RegistrationForm.aspx#		
7	Pradhan Mantri Fasal Bima Yojna	https://pmfby.gov.in/farmerRegistrationForm		
8	Pradhan Mantri Krishi Sinchai Yojana	https://pmksy.nic.in/Ksuvidha/Beneficiary_Entry.aspx		
9	Sub Mission on Agriculture Mechanization-Central Sector	https://agrimachinery.nic.in/Farmer/Management/Index		
10	Sub Mission on Agriculture Mechanization-Centrally Sponsored	https://agrimachinery.nic.in/Farmer/Management/Index		
11	Sub-Mission on Seeds and Planting Material	https://seednet.gov.in/		
12	National Fellowship and Scholarship for Higher education for ST students (for FELLOWSHIP)	https://fellowship.tribal.gov.in/StudentsRegistrationForm.aspx		
13	National Fellowship and Scholarship for higher education for ST students (for SCHOLARSHIP)	https://scholarships.gov.in/fresh/newstdRegfrmInstruction		
14	National Fellowship for Students with Disabilities	https://www.ugc.ac.in/nfpwd/		
15	National Renewable Energy Fellowship Programme and National Solar Science Fellowship Programme	https://hrd.mnre.gov.in/register		
16	P.G. Scholarship for University Rank Holders	https://scholarships.gov.in/fresh/newstdRegfrmInstruction		

Fig 8.4.1 Schemes screen

CHAPTER 9

CONCLUSION

This program makes the user simpler to collect the census of village. The entire house details stored under single house number, this is one benefit to user to simply search and locates his required house details. This program deals with mainly four operations of create details, deleting them, modifying, and searching according the user's choice. Each operation is made as an individual function and so control enters to different structures and all the data added, modified or deleted is going to be store and there is a government scheme, government jobs options also available so village peoples can get know about all the schemes and jobs of government by selecting the options.

REFERENCES

- [1] <https://www.programiz.com/>
- [2] <https://www.geeksforgeeks.org/>
- [3] <https://www.javatpoint.com/>
- [4] <https://anzeljg.github.io/>
- [5] <https://www.tutorialspoint.com/>
- [6] <https://docs.python.org/>
- [7] <https://realpython.com/>
- [8] <https://www.edureka.co/>
- [9] <https://pythonprogramming.net/>
- [10] <https://www.tutorialsteacher.com/>