

## Introduction:

The **Contact Management System** is a Python-based application designed to manage and organize a list of contacts efficiently. This project helps users store, update, and delete contact details like name, address, gender, and phone number, all within a simple GUI created using the Tkinter module. The system's primary purpose is to allow users to store personal information and maintain an organized contact list, which can be modified as needed.

This project is a great way to demonstrate how Python can be used to build functional GUI applications. It's also an excellent learning tool for beginners to improve their skills in Python programming, especially in creating GUI-based applications using Tkinter.

### Features of the Contact Management System:

1. **View All Contacts:** On launching the application, the user can view all stored contacts.
2. **Add Contacts:** Users can add new contacts by filling out the form with relevant details.
3. **Update Contacts:** The system allows users to modify existing contact details by double-clicking on a particular entry.
4. **Delete Contacts:** Users can remove any contact from the list by selecting it and clicking the delete button.

## Required Modules or Packages:

The system was developed using **Tkinter**, which is a Python package used for building simple GUI applications. Below are the essential packages required for this project:

### Tkinter:

- 
- Used for building the graphical user interface (GUI).
- Provides various widgets such as buttons, text fields, and tables, which are essential for creating the contact management system.
- Installation:

```
pip install tk
```

### SQLite3:

- 
- Used for managing the database that stores all the contact details.
- SQLite is a lightweight, file-based database that requires minimal configuration and works well for small applications.
- No external installation is required for SQLite, as it comes bundled with Python.

### Dependencies:

- **Python 3.x:** Ensure Python is installed on your system.
- **Tkinter Library:** Tkinter is part of the Python Standard Library, but you might need to install it manually, depending on your OS

## How to Run the Code:

Follow the steps below to run the **Contact Management System** on your machine:

1. **Download Python:**
  - Make sure Python is installed. You can download it from [here](#).
2. **Extract the Project:**
  - Extract the downloaded ZIP file containing the project.
3. **Open the Project:**
  - Navigate to the project folder and open the `index.py` file using Python's IDLE.
4. **Run the Script:**
  - Press F5 or click on the **Run** menu, and select **Run Module** to execute the code. This will launch the contact management system.

## Code Explanation:

The following sections explain some important parts of the code used in the **Contact Management System**:

### Database Connection:

- 
- The project uses SQLite3 to handle database operations, ensuring efficient management of contact information.

```
import sqlite3

conn = sqlite3.connect('contact.db')
c = conn.cursor()
```

Here, a connection to the **contact.db** database is created, allowing the system to read and write contact data.

### Adding Contacts:

- A function is created to allow users to add a new contact by inputting their personal details.

```
def add_contact():
    first_name = entry_first_name.get()
    last_name = entry_last_name.get()
    ...
    c.execute('INSERT INTO contacts (first_name, last_name, ...) VALUES (?, ?, ...)', ...)
    conn.commit()
```

- This function retrieves user input and inserts it into the database using an SQL query.

### Updating Contacts:

- The system allows updating contact details by double-clicking on a contact and editing the data.

```
def update_contact():
    selected_contact = contact_list.selection()
    ...
    c.execute('UPDATE contacts SET first_name=?, last_name=? WHERE id=?',
    ...)
    conn.commit()
```

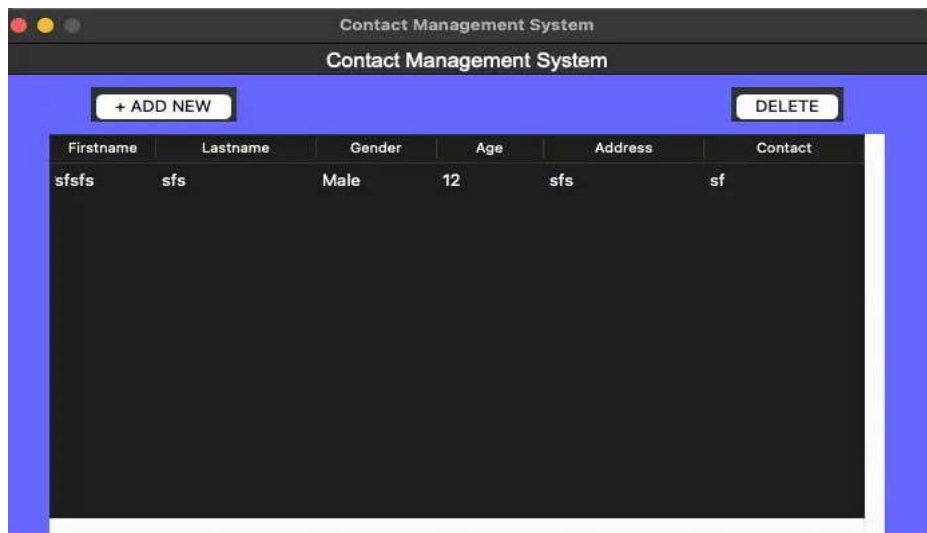
- The updated contact details are then saved back to the database.

### Deleting Contacts:

- A function to delete contacts from the database.

```
def delete_contact():
    selected_contact = contact_list.selection()
    ...
    c.execute('DELETE FROM contacts WHERE id=?', ...)
    conn.commit()
```

### Output:



Firstname	Lastname	Gender	Age	Address	Contact
sfsfs	sfs	Male	12	sfs	sf



Adding New Contacts

Firstname:

Lastname:

Gender: ☒ Male ☐ Female

Age:

Address:

Contact: