Parshvanath Charitable Trust's

A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)

(Religious Jain Minority)

A MINI PROJECT REPORT

On

Profile Management Application

Submitted in partial fulfillment of the requirement of
University of Mumbai for the Course

In

Computer Engineering (IV SEM)

Submitted By
Om Bhoir
Aarya Bivalkar
Karan Fursule

Subject Incharge
Prof. Harshali Bhuwad

Parshvanath Charitable Trust's

A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)

(Religious Jain Minority)

CERTIFICATE

This is to certify that the requirements for the project report entitled ‘**Profile Management App**’ have been successfully completed by the following students:

Name	**Moodle Id**
Om Bhoir	23102010
Aarya Bivalkar	23102008
Karan Fursule	23102111

In partial fulfilment of the course Skill Base Lab Course: Python Programming (CSL405) in Sem: IV of Mumbai University in the Department of Computer Engineering during academic year 2024-2025.

Sub-in-Charge

Parshvanath Charitable Trust's

A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)

(Religious Jain Minority)

PROJECT APPROVAL

The project entitled '**Profile Management App**' by **Om Bhoir, Aarya Bivalkar and Karan Fursule** are approved for the course of Skill Base Lab Course: Python Programming (CSL405) in Sem: IV of Mumbai University in the Department of Computer Engineering.

Subject-in-Charge

Date:

Place: Thane

Parshvanath Charitable Trust's

A. P. SHAH INSTITUTE OF TECHNOLOGY

(Approved by AICTE New Delhi & Govt. of Maharashtra, Affiliated to University of Mumbai)
(Religious Jain Minority)

Table of Contents

Abstract

The Profile Management App is a Python-based application designed to streamline the creation, modification, and management of user profiles in a secure and user-friendly environment. This mini project aims to demonstrate the core concepts of object-oriented programming, file handling, and user interface design using Python. The app allows users to create new profiles, update existing information, view profile details, and delete profiles as needed. Data is stored either in local files (e.g., JSON, CSV, or SQLite) to ensure persistence between sessions. The system includes basic input validation and user authentication features to maintain data integrity and security. This project provides a practical understanding of how profile management systems operate and serves as a foundational step for more advanced applications involving databases, web development, or mobile platforms.

The app allows users to create, view, update, and delete personal profile information with ease. Built using core Python concepts such as object-oriented programming and file handling, it ensures data persistence and basic user interaction through a command-line or graphical interface. The project demonstrates practical implementation of CRUD operations and can serve as a foundational model for more advanced systems involving databases or web integration.

Problem Definition

In the digital era, efficient data collection and management are essential for every organization, especially those involved in training, recruitment, or educational activities. Collecting and organizing user profile information such as name, phone number, gender, and programming language proficiency is a recurring requirement. However, when this information is gathered and stored manually, it can lead to several problems such as data redundancy, errors in entry, difficulty in storage and retrieval, and overall inefficiency in management.
For small to mid-scale organizations and training centers that may not have access to complex software systems, a simple and user-friendly solution is required to automate the process of profile data collection. A lightweight desktop application with a graphical user interface (GUI) offers an accessible and practical way to streamline the process. The system should be capable of accepting user inputs through clearly labeled fields, validating the inputs to ensure data correctness, and storing them in a structured database format for future reference and analysis.

This project aims to develop a **Profile Management System** using **Python** as the programming language. The **Tkinter** library is utilized to build the graphical user interface, making it intuitive for users without any technical background. The backend of the application uses **SQLite**, a lightweight and serverless relational database system, to store and manage user records efficiently. Users can input their name, phone number, select their gender using radio buttons, and indicate their programming language skills using checkboxes.

To ensure reliability and data integrity, basic input validation is implemented. For instance, if the name or phone number is left blank, the application alerts the user with an appropriate error message and prevents incorrect submission. Once all validations are passed, the data is stored in the database, and the form is reset for the next entry. This cycle ensures both ease of use and robust data handling.

The ultimate goal of this application is to provide a reliable and efficient solution for managing user profile data in a digital format, minimizing human error, and reducing manual workload. This project demonstrates the effective integration of GUI programming with database management and serves as a foundational example for more advanced systems that involve data collection and user management.

Introduction

In recent years, the need for digitizing basic data entry and management tasks has increased significantly across various sectors, including education, training, and small businesses. One such essential task is the creation and maintenance of individual user profiles. Whether it's for enrolling students in a course, registering participants for workshops, or gathering employee information, the process typically involves collecting standard personal data like name, contact number, gender, and skill sets. Performing this process manually is not only tedious but also leads to various issues like errors in data entry, misplacement of records, and difficulty in accessing and organizing information.
This project aims to develop a **Profile Management System**, a user-friendly desktop application using **Python** as the programming language. The application uses the **Tkinter** library to create a graphical user interface (GUI) and **SQLite** as the backend database for storing user information. Tkinter allows for the rapid development of visually interactive applications, while SQLite offers a lightweight, serverless database that is ideal for small-scale local data storage.

The core functionality of the application includes accepting user input for basic details such as name and phone number, allowing gender selection through radio buttons, and enabling users to select known programming languages through checkboxes. Basic input validation ensures that mandatory fields are not left empty and that data integrity is maintained before storing it into the database. After a successful submission, the application resets the form to accept new entries, providing a seamless experience for repeated usage.

By implementing this project, students gain practical exposure to developing desktop-based applications, designing user interfaces, handling input validation, and performing database operations. It also serves as a foundational project for understanding how to integrate frontend components with backend data storage systems. This system can be further enhanced in the future to include more functionalities such as editing and deleting records, data export, and even cloud integration.

Description of the modules used

The Profile Management System is developed using **Python**, a powerful high-level programming language, along with **Tkinter** for the graphical user interface and **SQLite** for database management. The application is modular in nature and comprises the following key components or modules:

1. Tkinter Module (GUI Module)

Tkinter is the standard GUI library for Python. It allows the creation of window-based applications using widgets such as labels, entry boxes, radio buttons, checkboxes, and buttons. In this project, Tkinter is used to design the main interface of the Profile App. The GUI provides an interactive form where users can input their data. The `Label`, `Entry`, `Radiobutton`, `Checkbutton`, and `Button` widgets are used extensively to create a user-friendly form layout.

2. tkinter.messagebox Module

This module is used to display popup dialogs like warnings, information messages, and error alerts. It enhances user experience by guiding them with appropriate messages during data entry. For instance, if the user attempts to submit the form without filling in mandatory fields like name or phone number, the `showerror()` method is triggered to prompt the user.

3. sqlite3 Module (Database Module)

SQLite is a lightweight, disk-based database that doesn't require a separate server process. Python's `sqlite3` module provides an easy interface to connect and interact with SQLite databases. In the Profile Management System, SQLite is used to create and maintain a table named `candidates` where each user's data is stored. The application connects to the database, executes an SQL `INSERT` command to store the data, and handles transactions and exceptions accordingly.

4. Validation and Logic Module

This internal logic module is implemented within the `save()` function. It checks whether required fields are filled and ensures valid data is submitted to the database. It also manages form resetting after successful submission and handles error conditions using exception handling (`try-except` block). Additionally, it sets default values for form controls and manages user interaction smoothly.

These combined modules work together to create a robust and functional application that can be used to manage and store user profiles efficiently. Each module has been selected and implemented to keep the application lightweight, responsive, and simple for end users.

Implementation details with screen-shots

The implementation of the Profile Management System was carried out in a structured and modular manner. Each feature was developed incrementally, ensuring proper functionality and seamless integration. The following are the step-by-step details of the project implementation:

Step 1: Designing the GUI Layout using Tkinter

The graphical user interface was designed using the `Tkinter` module. A main window was created using the `Tk()` class, and its title, size, and position were set. Labels and Entry widgets were placed for user input fields like name and phone number. A header label was also added for better visual appeal.

Step 2: Adding Gender Selection using Radiobuttons

To allow users to select their gender, two radio buttons were created and grouped using an `IntVar()` variable. The default selection was set to "Male". The selection updates the internal variable that is later used during data storage.

Step 3: Adding Language Options using Checkbuttons

Three checkboxes were added for users to select their known programming languages: Python, Java, and JavaScript. These checkbuttons are linked to separate `IntVar()` variables which are checked while constructing the language string before storing the data.

Step 4: Input Validation

A `save()` function was defined to handle data validation and database interaction. This function first checks if the name and phone number fields are filled. If not, appropriate error messages are shown using the `messagebox.showerror()` function, and the input focus is directed to the missing field.

Step 5: Database Integration using SQLite

The SQLite database named `profile.db` was created. The application connects to this database using the `sqlite3` module. Inside the `save()` function, an SQL `INSERT` query is constructed to add the validated data into a table named `candidates`. The values inserted include the name, phone number, selected gender, and programming languages.

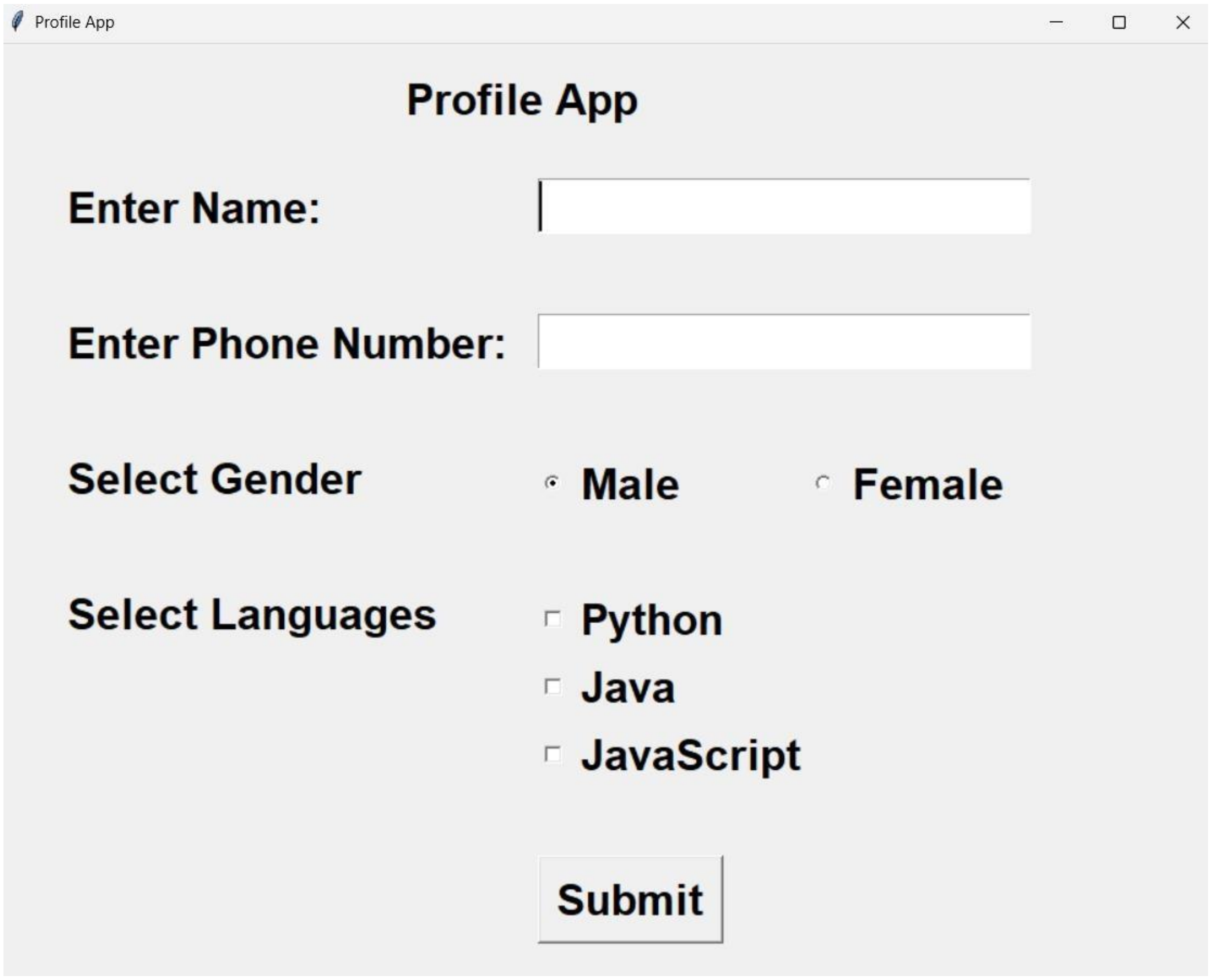
Step 6: Handling Successful Submission

Once the data is successfully inserted into the database, a success message is displayed using `showinfo()`. The form is then reset — all input fields are cleared, checkboxes and radio buttons are reset to default states, and the cursor is returned to the name entry field.

Step 7: Exception Handling and Cleanup

Proper exception handling is implemented using `try-except-finally` blocks to manage database errors and ensure the connection is closed regardless of the outcome. If any error occurs during data insertion, the changes are rolled back, and the user is notified.

This modular and structured approach ensures that the application runs efficiently and remains maintainable. The implementation highlights the integration of GUI design, user interaction, validation, and database management in Python.


Profile App

Profile App

Enter Name:

Enter Phone Number:

Select Gender [✓ Male] Female

Select Languages
Python
Java
JavaScript

Conclusion and Future Scope

The Profile Management System was successfully designed and implemented using Python's Tkinter and SQLite libraries. The primary objective of developing a user-friendly desktop application for storing and managing profile data was achieved. The system allows users to input their name, contact number, select gender, and choose known programming languages, all through a clean graphical interface. The use of input validation ensures that only correct and complete data is stored in the backend SQLite database.
The application simplifies the process of profile data collection, reducing human error, and improving overall efficiency. It also provides immediate feedback through informative messages and maintains the entered records in a structured format. From a learning perspective, the project helped gain hands-on experience in GUI development, backend integration, and the use of database operations in Python.

Future Scope:

While the current version of the application meets the basic requirements of profile data entry and storage, there are several enhancements that can be added to extend its functionality:

1. **Edit and Delete Features:** The ability to update or remove records from the database can be introduced for better data management.

2. **View All Entries:** A new interface or window can be created to view all submitted profiles in a tabular format.

3. **Data Export:** The option to export data to CSV or Excel format can be useful for analysis and reporting.

4. **Search Functionality:** Implementing a search feature would allow users to retrieve specific records quickly.

5. **Cloud Integration:** Instead of storing data locally, integration with cloud-based databases can make the application scalable and accessible remotely.

6. **Authentication System:** Adding a login system would make the application more secure and restrict access to authorized users only.

Overall, this project provides a strong foundation for building more complex applications and can be further expanded into a complete data management tool with additional features.

References

1. https://docs.python.org/3/
2. https://tkdocs.com/
3. https://www.sqlite.org/index.html
4. https://www.geeksforgeeks.org/python-gui-tkinter/

Acknowledgement

I would like to express my sincere gratitude to all those who supported me throughout the development of this mini project, **Profile Management App**. First and foremost, I extend my heartfelt thanks to my project guide/mentor [**Harshali Bhuwad**], whose valuable guidance, constant support, and insightful feedback helped me in completing this project successfully. I am also thankful to my faculty members and classmates for their encouragement and constructive suggestions during the project. Lastly, I would like to acknowledge my family and friends for their continuous motivation and support, which played a crucial role in the successful completion of this project.