**Four Weeks Industrial Training Project Report on**  
**BANK MANAGEMENT SYSTEM**

Submitted in the partial fulfilment of the requirement for the award of degree of  
**Bachelor of Technology**  
in  
**Computer Science and Engineering  
Batch (2025-26)**



Submitted to: Submitted by:

**Dr. Ridhi Kapoor** **Ranjana**

**(12300283)**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING  
DAV UNIVERSITY**

**JALANDHAR- PUNJAB 144012**

**1. Certificate of Training**

****

****

**2. ABSTRACT**

This project focuses on developing an efficient **Bank Management System** aimed at streamlining core banking operations and enhancing customer service using modern software development techniques. The objective is to build a secure, reliable, and user-friendly system capable of managing essential banking functionalities such as customer account creation, transaction processing, balance inquiries, and record maintenance.

The system underwent a structured design and development process, including requirement analysis, database design, interface creation, and implementation of key modules to ensure accuracy, data integrity, and smooth workflow. A robust backend database was integrated to store and manage customer information securely, while validation and error-handling mechanisms were incorporated to prevent inconsistencies and unauthorized access.

Various features—such as deposit and withdrawal management, transaction history tracking, account updates, and automated calculations—were implemented to support efficient banking operations. The system was thoroughly tested to evaluate its performance, reliability, and responsiveness across different scenarios.

The results demonstrate how a well-designed Bank Management System can significantly improve operational efficiency, reduce manual workload, and enhance decision-making within the banking sector. This project reflects the practical application of software engineering principles learned during the training at **Qspider Pune, Deccan Gymkhana**, and highlights their relevance in solving real-world challenges related to digital banking and financial management.

**3. ACKNOWLEDGEMENT**

I would like to extend my heartfelt gratitude to everyone who supported us during the various stages of this project’s development. I am deeply thankful to **Mr. Rahul Hans (Coordinator), DAV University**, for granting me the opportunity to undertake the 45-day summer training at **Qspider Pune, Deccan Gymkhana.**

I am also grateful to all the faculty members of the **Department of Computer Science and Engineering** for their guidance, encouragement, and assistance, which proved invaluable in the preparation of the final report and presentation.

Finally, I express my sincere thanks to the entire team at **Qspider Pune, Deccan Gymkhana**, especially **Mr. Prabhu Gouda and Mr. Nitin Singh, our guiders**, for their constant support, mentorship, and for making this training experience meaningful, productive, and truly beneficial.

**4. DECLARATION**

I, **Ranjana (Reg. No.: 12300283)**, hereby declare that the work presented in this project/training titled **“Bank Management System”** is an authentic record of my own work carried out under the guidance of **Mr. Prabhu Gouda and Mr. Nitin Singh (Qspider Pune, Deccan Gymkhana)**. This project is submitted in partial fulfilment of the requirements for the award of the **Bachelor of Technology (B.Tech) Degree in Computer Science and Engineering**.

To the best of my knowledge, the content of this report has not been submitted to any other University or Institute for the award of any degree or diploma.

**Student Name:** Ranjana  
**Reg. No.:** 12300283  
**Signature:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
**Date:** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Table of Contents**

Certificate of Training .................................................... 2

Abstract ............................................................................ 3

Acknowledgement ............................................................ 4

Declaration ....................................................................... 5

Introduction ....................................................................... 7

Title of the Project .......................................................... 8

Objectives ........................................................................... 9

Steps to Achieve Objectives ......................................... 10-11

Python and MySQL: Bank Management System ............ 12

 9.1 Introduction to Python ................................................ 12

 9.2 Introduction to MySQL ................................................. 13

 9.3 Introduction to Banking Management System .......... 14

 9.4 System Feature Overview ............................................ 14-15

Project Summary .......................................................... 16-17

Appendices ...................................................................... 18

 Appendix A: Source Code .................................................. 18-23

 Appendix B: Tools, Technologies, and Platforms ............ 24

References ....................................................................... 24

**5. INTRODUCTION**

This report presents the work completed during my 45-day summer training at Qspider Pune, Deccan Gymkhana, undertaken as part of the academic requirements of the Department of Computer Science and Engineering, DAV University. During this training period, I worked on the project titled **“Bank Management System.”**

The primary objective of this project was to design and develop a secure, efficient, and user-friendly system capable of managing essential banking operations. In modern financial institutions, manual processes can lead to errors, delays, and data management challenges. A digital banking system provides an effective solution by automating key tasks such as customer account creation, transaction handling, balance tracking and record maintenance.

The development workflow involved several major steps, including requirement analysis, database design, interface development, module implementation, and system testing. Core functionalities such as deposit and withdrawal processing, account updates, transaction history management, and customer information storage were integrated into the system. Special emphasis was placed on accuracy, data security, and consistency to ensure smooth and reliable operations.

This project offered valuable hands-on experience in software development practices and highlighted the importance of structured system design in solving real-world problems. The training experience not only strengthened my technical skills but also enhanced my understanding of how digital systems support efficient and error-free management within the banking sector.

**6. TITLE OF THE PROJECT**

**Bank Management System**

**7.OBJECTIVES**

**• To understand the** **core concepts of** **Python and MySQL** and learn how these technologies can be used to manage and organize data in banking-related operations.  
  
**• To successfully complete the 45-day summer training progra**m at **Qspider Pune, Deccan Gymkhana**, gaining practical exposure to handling databases, writing Python scripts, and performing backend operations.  
  
• To work on the project **“Bank Management System,”** aimed at simulating basic banking processes such as account creation, transactions, and record management using Python and MySQL.  
  
• **To perform essential tasks** involved in the project workflow, including:  
 o Understanding banking data requirements  
 o Creating and managing MySQL databases  
 o Writing Python code to interact with the database  
 o Executing queries for storing, updating, and retrieving data  
 o Implementing logic for deposits, withdrawals, and balance checks  
 o Testing the correctness of results  
  
• To develop a clear understanding of **how data flows within a banking system and how Python–MySQL** integration helps in managing customer and transaction records.  
  
**• To gain hands-on experience in writing structured Python programs and SQL** queries for handling real-time banking operations.  
  
• **To understand the importance of** data accuracy, consistency, and security while working with financial information.  
  
• **To enhance problem-solving ability by** designing logical workflows for banking functions such as account handling, transaction tracking, and record maintenance.  
  
• **To apply theoretical knowledge** from academic coursework in a practical environment and learn how Python and MySQL can be used together to build data-driven systems.

**8. STEPS TO ACHIEVE OBJECTIVES**

**1. Understanding the Project Domain**

a. Studied the basics of banking operations such as customer management, account handling, and transaction processing.  
b. Understood how a Bank Management System helps in maintaining customer records and managing financial transactions.  
c. Discussed the project objectives and functional requirements with the mentor.

**2. Analysing the Project Requirements**

a. Understood the expectations of developing a functional bank management system using Python and MySQL.  
b. Identified the required technologies: Python (IDLE), MySQL database, and MySQL Connector for Python.  
c. Designed a project workflow covering database setup, Python scripting, and testing.

**3. Studying the Problem Statement**

a. Analysed the goal: to create a system that performs essential banking operations digitally.  
b. Identified the main functionalities—account creation, deposit, withdrawal, balance check, and viewing transaction history.  
c. Understood system limitations and security considerations (input validation, safe queries, etc.).

**4. Database Design and Preparation (MySQL)**

a. Created a MySQL database specifically for the Bank Management System.  
b. Designed tables such as **customers**, **accounts**, and **transactions**.  
c. Defined primary keys, foreign keys, and constraints to maintain data accuracy.

**5. Backend Logic Development (Python in IDLE)**

a. Connected Python to the MySQL database using mysql.connector.  
b. Wrote Python functions for each operation such as:

* Adding new customers
* Creating accounts
* Deposits and withdrawals
* Checking balance
* Recording transactions  
  c. Added exception handling and input validation to ensure error-free execution.

**6. Implementing System Features**

a. Implemented CRUD operations for customer and account details.  
b. Developed menu-driven interaction so users can select options like deposit, withdraw, or view information.  
c. Ensured automatic transaction entries after each financial operation.

**7. Testing and Debugging**

a. Tested each feature by performing sample deposits, withdrawals, and account creations.  
b. Checked whether MySQL tables were updated correctly after each operation.  
c. Fixed errors related to database connectivity, incorrect queries, and invalid user inputs.

**8. Result Verification**

a. Verified all operations by comparing the Python output with stored values in the MySQL database.  
b. Ensured accurate balance updates and correct recording of all transactions.  
c. Confirmed that the system works smoothly without data loss or misuse.

**9. Skill Enhancement and Learning**

a. Strengthened understanding of Python programming using IDLE.  
b. Improved knowledge of MySQL database design, SQL queries, and database relationships.  
c. Learned how to integrate Python with MySQL to build real-world applications.

**10. Documentation and Report Preparation**

a. Documented the database schema, Python scripts, and system working in detail.  
b. Included screenshots of table structures, code execution, and sample outputs.  
c. Prepared a structured final report covering objectives, methodology, results, and learning outcomes.

**9. Python and MySQL:**

**Bank Management System**

**9.1. Introduction to Python**

Python is a high-level, general-purpose programming language widely known for its simplicity, readability, and powerful capabilities. Created by Guido van Rossum in 1991, Python has grown into one of the most popular languages used across industries, academia, and research fields. Its clean and intuitive syntax makes it an excellent choice for beginners, while its extensive libraries and frameworks make it equally valuable for advanced programmers and large-scale application development.

One of Python’s key strengths lies in its versatility. It supports multiple programming paradigms, including procedural, object-oriented, and functional programming. This flexibility allows developers to choose the most suitable approach depending on the problem they are solving. Python’s dynamic typing, automatic memory management, and interactive environment make development faster and more efficient.

Python has a rich ecosystem of libraries and modules that greatly extend its functionality:

* Standard Library: Includes modules for file handling, math operations, networking, date/time utilities, and more, reducing the need for writing code from scratch.
* Data Science Libraries: Tools like *NumPy*, *Pandas*, and *Matplotlib* enable users to analyze data, perform statistical computations, manipulate datasets, and create visualizations.
* Machine Learning & AI Libraries: Frameworks such as *Scikit-Learn*, *TensorFlow*, and *PyTorch* help build predictive models, neural networks, and advanced AI applications.
* Web Development Frameworks: Django and Flask simplify the creation of scalable and secure web applications.
* Database Connectivity: Python provides connectors such as MySQL. Connector and sqlite3 to interact seamlessly with databases, making it ideal for building systems like a Bank Management System.

Python’s simplicity and power have made it a preferred language in many industries. In software development, Python is used for automation, scripting, and application development. In scientific research, it supports simulations, modelling, and data analysis. In finance, Python helps with risk modelling, algorithmic trading, and fraud detection. It also plays a significant role in education due to its beginner-friendly nature.

For example, in a Bank Management System project, Python can be used to create a user-friendly, menu-driven program that interacts with a MySQL database. Through Python’s database connectivity modules, developers can fetch, store, and update customer and transaction records efficiently. This demonstrates how Python acts as a bridge between user input and backend database operations, making it ideal for real-world applications.

The continuous growth of Python is supported by its active community, open-source nature, and ever-expanding ecosystem. With emerging technologies such as artificial intelligence, data science, and automation becoming more prominent, Python’s role is becoming increasingly significant. As a result, Python continues to be a powerful, adaptable, and accessible language that meets the demands of modern computing and future innovation.

**9.2. Introduction to MySQL**

MySQL is an open-source, relational database management system (RDBMS) widely recognized for its reliability, speed, and ease of use. Originally developed by MySQL AB and first released in 1995, MySQL has grown into one of the most popular database systems used across web applications, enterprise solutions, and data-driven platforms. Its structured design and SQL-based query language make it a powerful tool for both beginners and experienced developers.

One of MySQL’s greatest strengths is its robust relational model, which organizes data into tables consisting of rows and columns. This structure ensures data consistency, integrity, and easy access. By using SQL (Structured Query Language), MySQL allows users to efficiently create, read, update, and delete data—operations foundational to nearly all database systems. Its support for transactions, indexing, and secure access control makes it suitable for mission-critical applications.

MySQL provides a wide range of features that enhance its flexibility and performance:

**Data Storage & Management**

MySQL offers multiple storage engines such as InnoDB and MyISAM, each optimized for different use cases. InnoDB supports transactions and foreign keys, which are essential for maintaining data integrity in complex applications.

**Scalability & Performance**

Known for handling large datasets and high-traffic environments, MySQL is used by many large-scale systems. Features like query optimization, indexing, caching, and replication allow developers to build scalable and high-performance solutions.

**Security Features**

MySQL ensures strong data protection through user authentication, role-based privileges, encryption, and secure access protocols. These features make it trusted for applications dealing with sensitive information, such as financial or customer records.

**Integration & Compatibility**

MySQL integrates seamlessly with numerous programming languages—including Python, Java, PHP, and C++—making it highly versatile. Its compatibility with modern platforms and cloud services allows developers to deploy databases efficiently across various environments.

**Use in Applications**

MySQL plays a central role in many types of applications:

* **Web Development:** Popular platforms like WordPress, Facebook, and YouTube have historically relied on MySQL due to its stability and speed.
* **Enterprise Systems:** Organizations use MySQL for CRM systems, inventory management, and financial applications.
* **Education & Research:** Its open-source nature makes it accessible for academic projects and experimentation.

For example, in a **Bank Management System**, MySQL can be used to store and manage customer records, transaction histories, account details, and authentication information. When combined with a programming language like Python, MySQL acts as the backend database where all structured data is stored. Through SQL queries, developers can efficiently insert, update, and retrieve banking data, ensuring accuracy and security in operations.

The continuous development of MySQL, backed by its large community and support from Oracle Corporation, ensures that it remains a reliable and evolving database technology. Its balance of simplicity, performance, and advanced features makes MySQL a preferred choice for modern applications requiring secure and efficient data management.

**9.3. Introduction to Banking Management System**

A Banking Management System is a software application designed to simplify, automate, and secure various banking operations. Modern banks deal with large volumes of customer data, financial transactions, and account-related activities. Managing all these tasks manually is time-consuming, error-prone, and inefficient. A computerized banking system ensures accuracy, speed, and reliability in handling everyday banking processes.

The primary purpose of a Banking Management System is to provide a centralized platform where all essential banking functions—such as account creation, deposit, withdrawal, fund transfer, balance enquiry, and transaction history—can be performed efficiently. By storing all customer and account information in a structured database, the system ensures data integrity and easy retrieval whenever required.

This project demonstrates how programming and database technologies can be combined to create a functional banking system. Python is used to develop the application interface and handle user inputs, while MySQL serves as the backend database for storing and managing the data. Through this integration, the system allows real-time operations such as updating balances, recording transactions, and verifying customer details.

A well-designed Banking Management System offers several benefits:

* Automation of Tasks: Reduces manual work and speeds up operations.
* Improved Accuracy: Minimizes human errors in calculations and record-keeping.
* Data Security: Protects sensitive financial information through authentication and controlled access.
* Easy Data Retrieval: Allows quick access to customer records and transaction details.
* User-Friendly Interface: Makes banking processes simpler for both staff and customers.

This project highlights the importance of technology in modern banking and demonstrates how software solutions can help build a secure, efficient, and organized system for managing financial operations. It also serves as a practical example of how Python and MySQL can work together to create real-world applications.

**9.3.1. System Feature Overview**

The Banking Management System is designed to handle essential banking operations in an efficient, secure, and user-friendly manner. The system provides a menu-driven interface where users can perform various functions related to account management and financial transactions. Each feature has been developed to ensure accuracy, reliability, and ease of use. Below is an overview of the key features included in the system:

**1. Account Creation**

Allows new customers to open a bank account by entering personal and account-related details. The system automatically stores this information in the database and generates a unique account number.

**2. Deposit Money**

Enables users to deposit money into an existing account. The system updates the account balance and records the transaction in the database for future reference.

**3. Withdraw Money**

Allows customers to withdraw money from their accounts, with built-in checks to prevent overdrawing. The updated balance is saved in the database after each successful withdrawal.

**4. Balance Enquiry**

Provides customers with their current account balance. The system fetches the latest data from the database to ensure accuracy.

**5. Fund Transfer**

Facilitates transferring money between two accounts within the bank. Both accounts are updated simultaneously, and transaction details are recorded securely.

**6. View Transaction History**

Displays a detailed list of all transactions associated with an account, including deposits, withdrawals, and transfers. This feature helps maintain transparency and track financial activity.

**7. Update Customer Details**

Allows authorized users to modify customer information such as address, phone number, or account type. The changes are stored instantly in the database.

**8. Delete Account**

Provides the option to close an existing account. All related records are removed from the database after confirmation, ensuring proper data management.

**9. User-Friendly Interface**

The system includes a simple and intuitive menu, making it easy for users to navigate between different options. Clear prompts and messages guide users through each operation.

**PROJECT SUMMARY — BANK MANAGEMENT SYSTEM (Python + MySQL)**

This project is a **console-based Bank Management System** built using **Python**, **MySQL**, and **NumPy**, designed to automate basic banking operations. The system allows users to create and manage bank accounts while storing all customer information securely in a MySQL database.

The database used is **BANK**, containing a single table **BANKING**, where customer records such as account number, name, account type, age, gender, mobile number, balance, and city are maintained. The Python program interacts with MySQL using the mysql.connector library.

The system is fully **menu-driven**, making it easy for users to navigate and perform banking operations. Each operation corresponds to a separate function in the program.

**⭐ Key Functionalities**

**✔ 1. Create New Account**

Collects customer details, generates an 8-digit random account number using NumPy, and inserts the record into the MySQL database. Optional initial deposit is supported.

**✔ 2. Modify Account**

Users can update their account type, age, mobile number, or city. SQL UPDATE queries ensure changes are saved in the database.

**✔ 3. Deposit Money**

Retrieves existing balance, adds the deposited amount, and updates the balance in MySQL.

**✔ 4. Withdraw Money**

Subtracts withdrawal amount from the customer's balance after validation and records the updated value.

**✔ 5. Transfer Money**

Transfers money from one account to another by updating both sender's and receiver's account balances.

**✔ 6. Check Current Balance**

Displays account number, name, and current balance using a SELECT query.

**✔ 7. Search Accounts**

Provides multiple search options:

* By name
* By account type
* By account number
* By city

**✔ 8. Delete Account**

Allows closing an account by removing it from the database using a DELETE query.

**⭐ Purpose of the System**

The purpose of this project is to demonstrate how Python can be used with MySQL to create a real-world banking system. It exhibits practical use of CRUD (Create, Read, Update, Delete) operations, data handling, and menu-driven application design.

**⭐ Benefits of the System**

* Simplifies banking operations
* Reduces manual errors
* Secure data storage in MySQL
* Fast and reliable
* Easy to modify or expand
* Demonstrates Python–SQL integration effectively

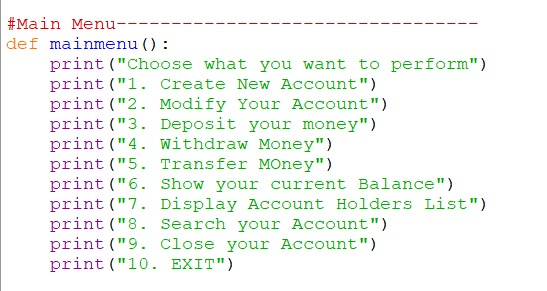
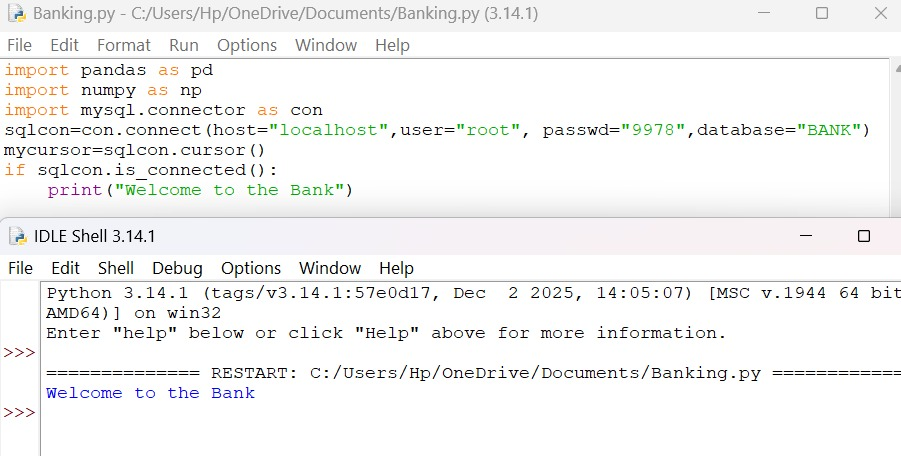
**⭐ Conclusion**

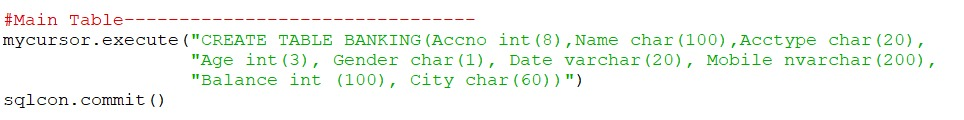
The Bank Management System successfully automates core banking operations such as account creation, money transactions, and record maintenance. With a structured database and efficient Python code, it offers an accurate and user-friendly banking solution. The project is an excellent example of integrating Python programming with database management for real-world applications.

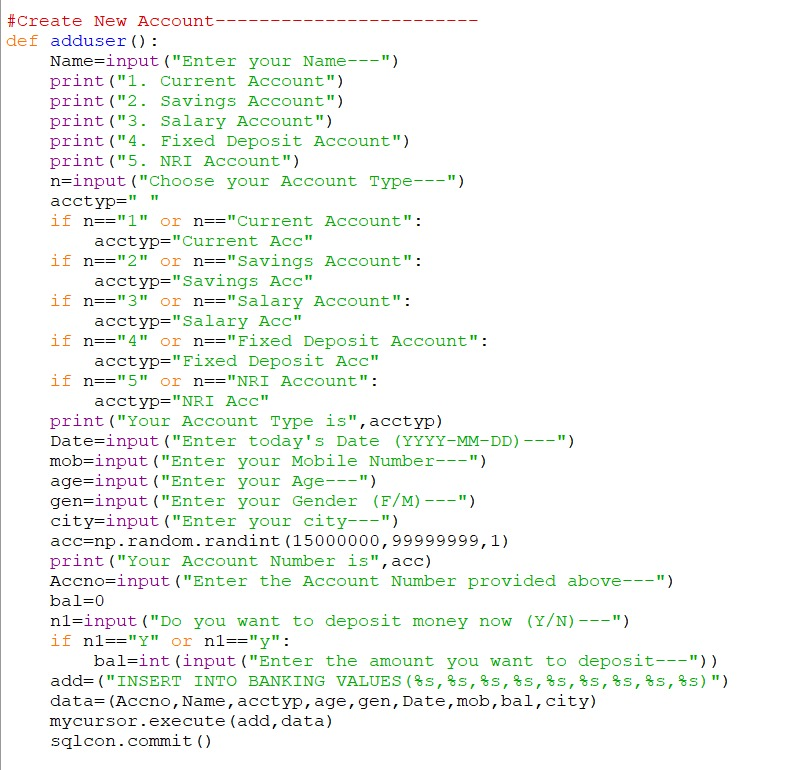
**20.Appendices**

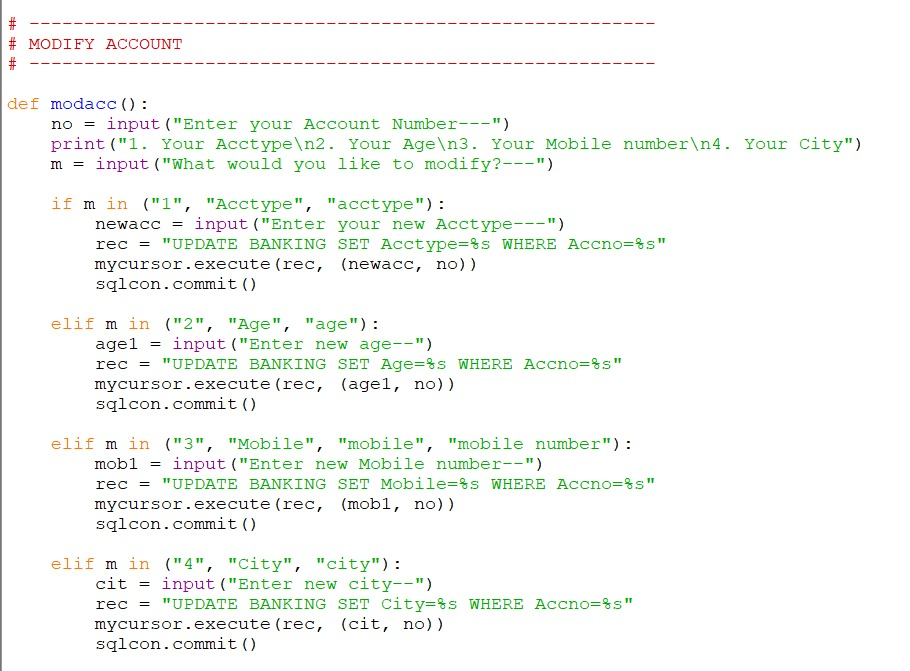
**Appendix – A: Source Code**

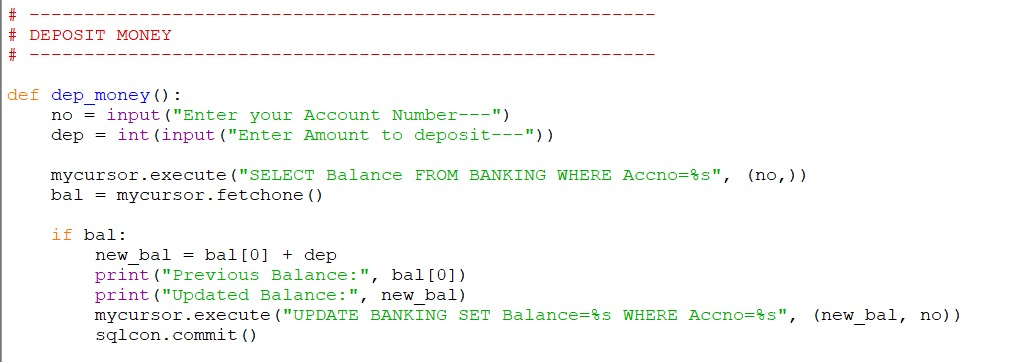
**This appendix contains the complete Python source code for the Bank Management System.  
The code includes modules for account creation, modification, deposit, withdrawal, balance inquiry, transfer, search, and deletion, along with MySQL database connectivity.**

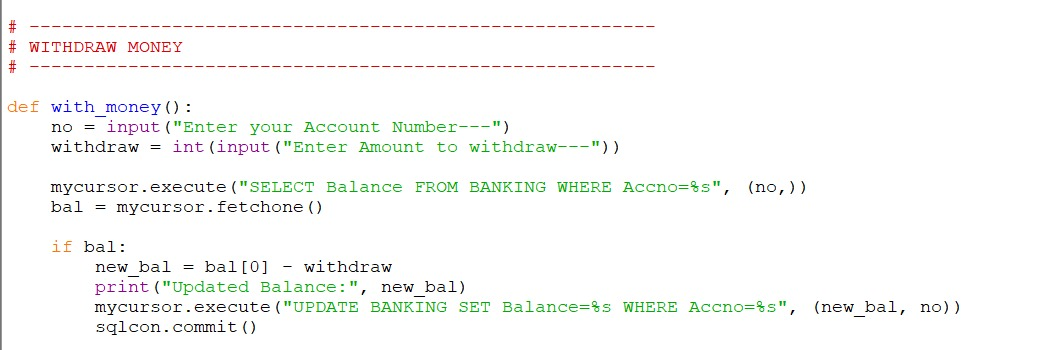
****

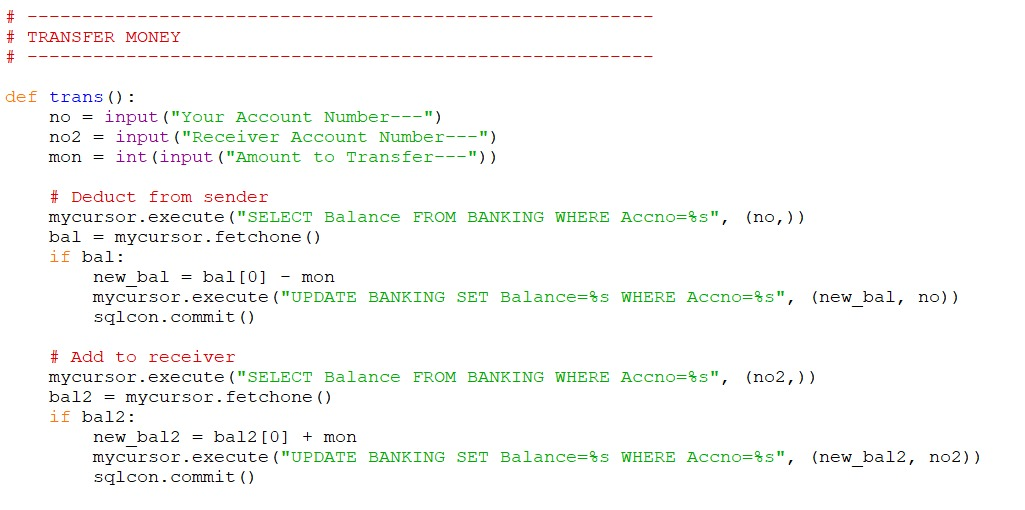
****

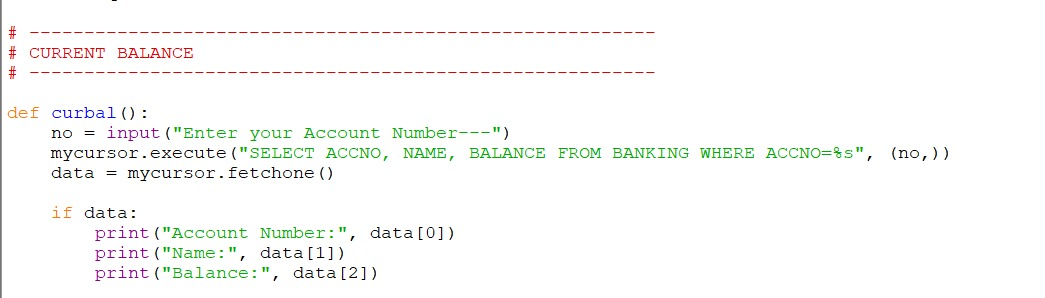
****

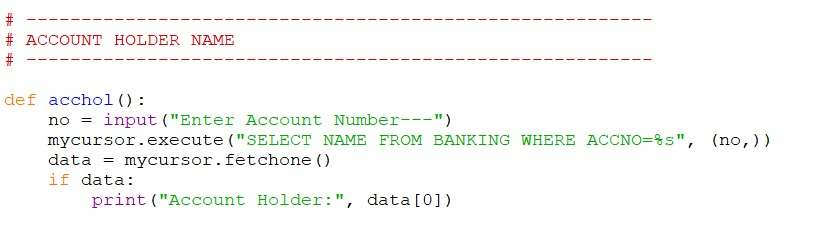
****

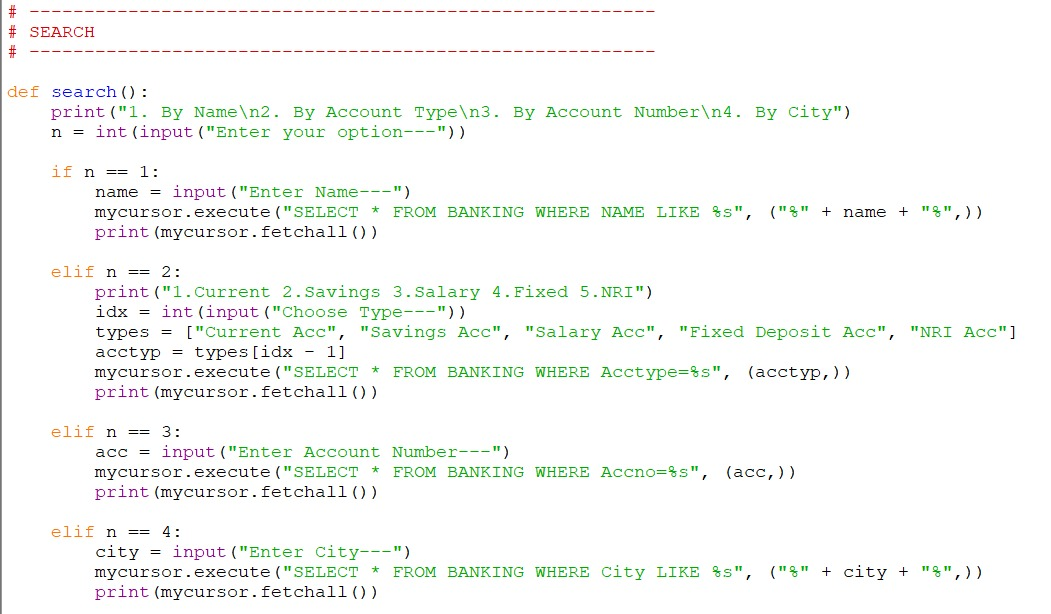
****

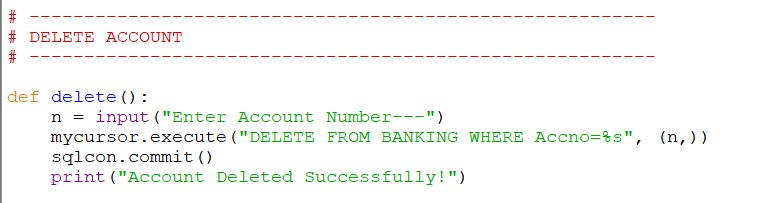


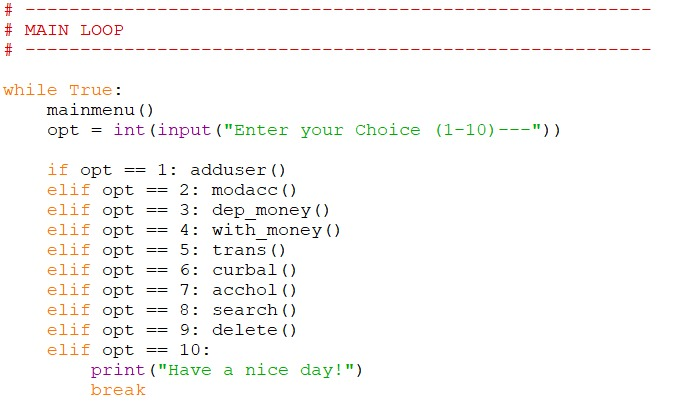
****

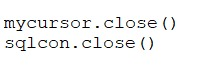
****

****

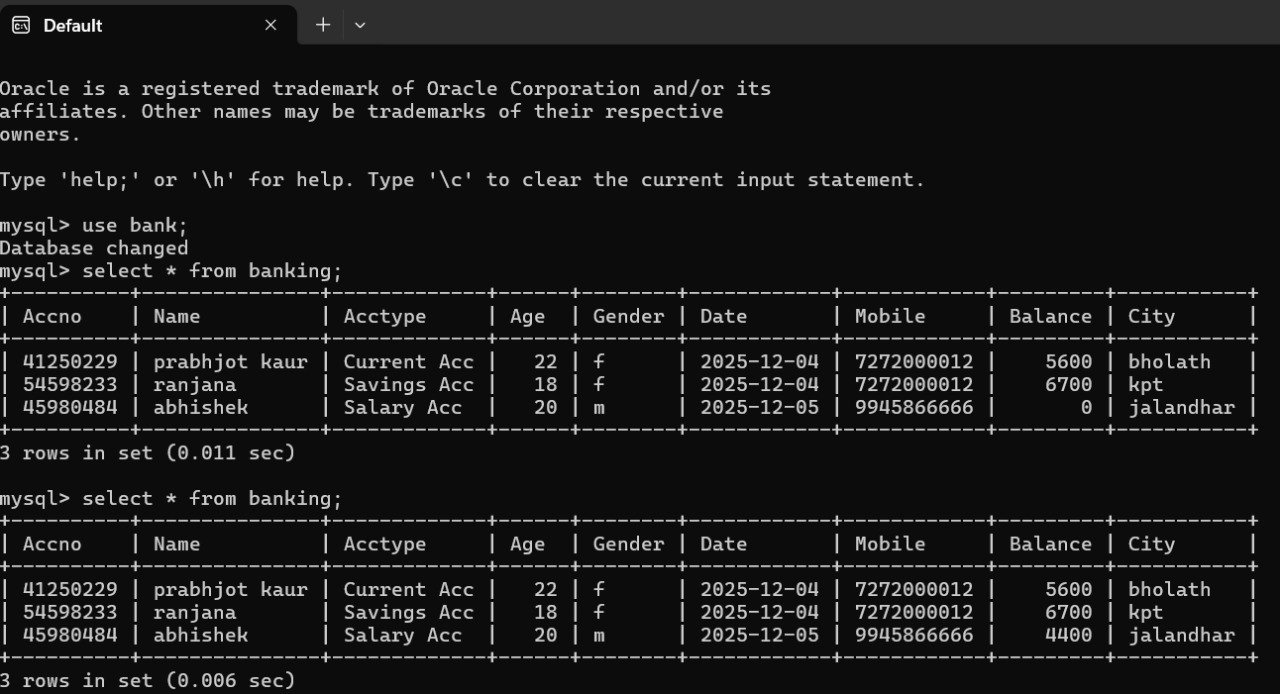
****

****

****

****

**OUTPUT:**

****

**Appendix B: Tools, Technologies, and Platforms**

**Programming Language**

* **Python 3**

**Development Environment(s)**

* **IDLE / VS Code / PyCharm** (Any Python IDE)
* **MySQL Workbench / phpMyAdmin** (For database management)

**Libraries Used**

* **mysql.connector** – To connect Python with MySQL
* **NumPy** – Used for generating random account numbers
* **Standard Python Libraries** – Input/output, loops, conditions

**Database Used**

* **MySQL Server**
* Database name: **BANK**
* Table name: **BANKING**

**Operating System**

* Windows / Linux (any compatible OS)

**References**

1. Python Software Foundation. *Python Documentation*.
2. MySQL Documentation. *SQL Commands and Database Management*.
3. NumPy Developers. *NumPy Library Documentation*.
4. TutorialsPoint. *Python MySQL Connector Usage*.
5. GeeksforGeeks. *Python Database Connectivity Tutorials*.