VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELAGAVI – 590018

A MINI-PROJECT REPORT ON

"BANK ACCOUNTS MANAGEMENT SYSTEM"

Submitted in partial fulfillment of the requirements of the award of degree of

BACHELOR OF ENGINEERING
IN
INFORMATION SCIENCE & ENGINEERING

Submitted By:

UVAIS MON V V N	**[4VV18IS112]**
VIGNESH V	**[4VV18IS116]**
ZABIULLA SHERIFF	**[4VV18IS119]**

UNDER THE GUIDANCE OF

Mr. Yashpal Gupta S
Asst Prof
Dept of ISE
VVCE, Mysuru

Ms. Kavyashree S
Asst Prof
Dept of ISE
VVCE, Mysuru

2020-2021

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
VIDYAVARDHAKA COLLEGE OF ENGINEERING
MYSURU-570002

Vidyavardhaka College of Engineering
Gokulam 3rd Stage,
Mysuru-570002
Department of Information Science and Engineering,

CERTIFICATE

This is to certify that the mini-project report entitled **"BANK ACCOUNTS MANAGEMENT SYSTEM"** is a bona-fide work carried out by **UVAIS MON V V N** (4VV18IS112), **VIGNESH V** (4VV18IS116) and **ZABIULLA SHERIFF** (4VV18IS119) students of 6th semester Information Science and Engineering, **Vidyavardhaka College of Engineering, Mysuru** in partial fulfillment for the award of the degree of **Bachelor of Engineering** in **Information Science & Engineering** of the **Visvesvaraya Technological University, Belagavi**, during the academic year **2020-2021**. It is certified that all the suggestions and corrections indicated for the internal assessment have been incorporated in the report deposited in the department library. The report has been approved as it satisfies the requirements in respect of mini-project work prescribed for the said degree.

Signature of the Guide	**Signature of the Guide**	**Signature of the HOD**
(Mr. Yashpal Gupta S)	**(Ms. Kavyashree S)**	**(Dr. A B Rajendra)**

Name of the Examiners **Signature with Date**

1)

2)

Acknowledgement

The Mini-project would not have been possible without the guidance, assistance and suggestions of many individuals. I would like to express our deep sense of gratitude and indebtedness to each and every one who has helped me to make this project a success.

We heartily thank our beloved Principal, **Dr. B Sadashive Gowda** for his whole-hearted support and for his kind permission to undergo the mini-project.

We wish to express our deepest gratitude to **Dr. A B Rajendra,** Head of Department, Information Science and Engineering, VVCE, for his constant encouragement and inspiration in taking up this mini-project.

We gracefully thank our mini-project guides, **Mr. Yashpal Gupta S** and **Ms. Kavyashree S**, Assistant Professors, Dept. of Information Science and Engineering for their encouragement and advice throughout the course of the mini-project work.

In the end, we are anxious to offer our sincere thanks to our family members and friends for their valuable suggestions and encouragement.

Uvais Mon V V N (4VV18IS112)
Vignesh V (4VV18IS116)
Zabiulla Sheriff (4VV18IS119)

Abstract

The Bank Accounts Management System is a software to maintain customers and admins accounts details and track all bank transactions. In this project we will show the working of a bank accounting system, cover the basic functionality of a bank and provide fast access to bank data as and when required by the customers and admins.

Both customers and admins will have separate module to interact with the system. Once logged in, they can perform a variety of operations. The operations differ for a customer account and an admin account.

The app is developed in Python, all user interactions occur through Tkinter GUI elements. Users interact with the app modules through these Tkinter windows.

TABLE OF CONTENTS

LIST OF TABLES AND FIGURES

Chapter 1

Introduction

1.1 OVERVIEW:

A bank is an institution that provides financial services such as issuing of money, receiving money in deposits and processing transactions. It is a service which is used by almost every individual frequently, this leads to storing of huge number of records of transactions.

In such cases efficiency of handling files plays an important role. Our project has features which help in retrieving records at a faster rate, and it also has features that reduce the risk in transactions by providing necessary security to its users.

The Bank Accounts Management System is a software to maintain customers and admins accounts details and track all bank transactions. In this project we will show the working of a bank accounting system, cover the basic functionality of a bank and provide fast access to bank data as and when required by the customers and admins.

1.4 OBJECTIVE:

The main objective of our project is to perform transactions, retrieve records at a faster rate, and to reduce the risk in transactions by providing necessary security to its users. The app maintains the balance between retrieval rate and disk space utilization. It also makes sure that no sensitive user data is accessed without proper authentication.

Chapter 2

HARDWARE AND SOFTWARE REQUIREMENTS

2.1 HARDWARE REQUIREMENTS:

Processor: Any 2.2 GHz or higher Intel or AMD processor.

RAM: 4 GB or higher

Storage capacity: 20 GB or higher

2.2 SOFTWARE REQUIREMENTS:

Operating system: Windows 7 or higher, Mac OS X, Linux.

Python 3.7 or higher interpreter.

Pycharm

Chapter 3

DESIGN AND IMPLEMENTATION

3.1 Design

This app consists of several modules that interact with each other to manipulate the data stored in the text file. All the user data and other data related to the app are stored in text files.

The starting point of the app is customer log in window. From there either the user can log in with his credentials or open admin window to log in with an admin account. Once logged in as a customer, the user can perform transactions, view his transaction details and perform account level operations like changing password or closing account.

The admin user can create new account, fetch customer transaction details, deposit or withdraw money to or from customer account and change their password.

All these operations by users internally calls different modules in the app to achieve the required result. All the data generated are written to the data files and index entries are created for every record. As the number of data files increases, level of indexing also increases.

3.2 File structure

This app will maintain a root folder named appdata. This folder consists of 4 sub folders named admin_data, customer_data, transaction_data and meta.

Meta folder contains files that stores universal data. Universal data includes number of entries in a file, name of the latest data file and next free account number that could be assigned to a new user.

Customer_data folder and admin_data folder contains customer and admin account details respectively. It contains 2 sub folders, data and index. Data folder contains records containing account details. Index folder contains indexing to data folder. Here, we have implemented multilevel indexing. Each level index file is stored at different folders. It also contains a meta file. This meta file stores meta data related to index levels.

Transaction_data folder contains data folder and index folder to store transaction data and its multilevel index. It also contains a secondary index folder that is used to index transactions based on transaction date grouped by account number.

Index files stores the primary key of the data record, name of the data file, and the offset of the respective record in that data file. Higher level index files stores starting key of an immediate lower level index file and its filename.

3.3 Implementation

This app implements multilevel and secondary indexing. There is a universal constant called block size that defines how many records should be stored in a single file. Insertion of data creates a respective index record in the index file. If the data file or index file exceeds the maximum number of records, an overflow occurs. Overflow event creates a new text file and writes data to the new file. Meta files are updated respectively to reflect the creation of a new file.

When an index file overflows, it also creates respective index at higher level index files to point to the new index file. This process occurs recursively as long as the index file overflows at higher levels.

Secondary index is created for transaction date. When a transaction occurs between two customers, say A and B on 2021/07/31, then two secondary index record is created, one under sec_index folder of A and another under sec_index folder of B. Secondary index record stores transaction type (credit or debit) and the transaction ID. The name of the file is same as the transaction date. So all transaction occurring on the same date relating to same account holder is saved in one file. This gives direct access to transaction data based on transaction date.

To access a file, the highest level of index is obtained from the meta file. Index file at the highest level is accessed to obtain the lower level index file name, this process continues recursively till level 1 index file is accessed. Then level 1 index data is used to access the record in data file directly.

3.3 MODULES AND THEIR DESCRIPTIONS:

3.3.1 Account manager module:

Account manager module manages both admin and customer accounts. This module is used to perform creation of new accounts, validation checks, account modification and account deletion.

3.3.2 File handler module:

This module is used for primary indexing and to do read write operations directly from the disk. This module has two classes i.e., Indexer and ReadWrite. Indexer class performs primary indexing of files and ReadWrite class performs reading and writing data from the disk.

3.3.3 Transactions module:

This module is used for managing transactions and to create secondary indexing of files. Transaction management include registering new transactions to the files, fetching transactions based on account number, transaction date and type of transaction. Secondary index is created for date grouped by account number to map transaction ID.

3.3.4 Meta module:

This module is used to handle meta data of the whole project. Meta data tracks global variables like next account number, current file name, number of record entries and index levels. It also stores meta data related to individual index levels.

3.3.5 U I module:

This module is used to perform the front-end designs. In this module there are two classes i.e., Admin and Customer. Admin class is used for rendering admin account interface and Customer module is used for rendering customer account interface.

Chapter 4

RESULT AND CONCLUSION

4.1 SNAPSHOTS:

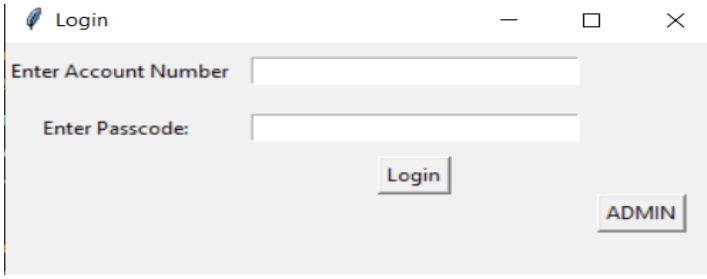

Fig 4.1.1

This window is the login page, a customer can log in using this window. An admin can move to admin window by clicking the admin button.

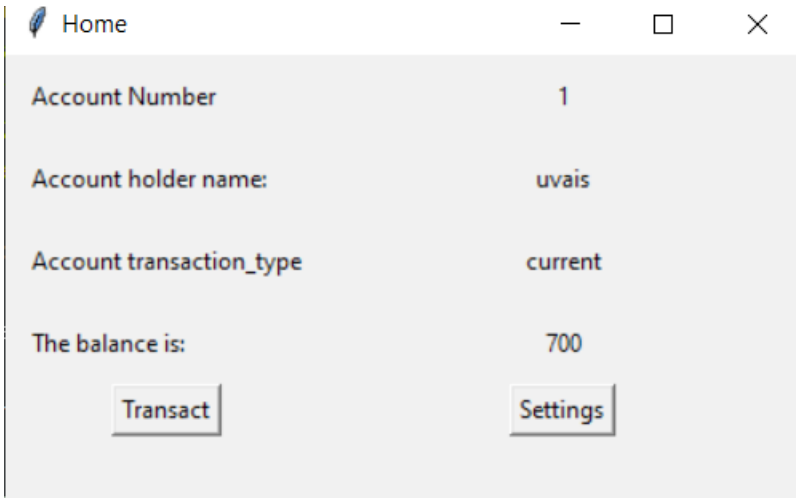

Fig 4.1.2

In this window several details of a customer are displayed. User can also access transaction and settings options from this window.

Fig 4.1.3

This is the transaction window. This window allows the user to check account statement, search transactions and send money.

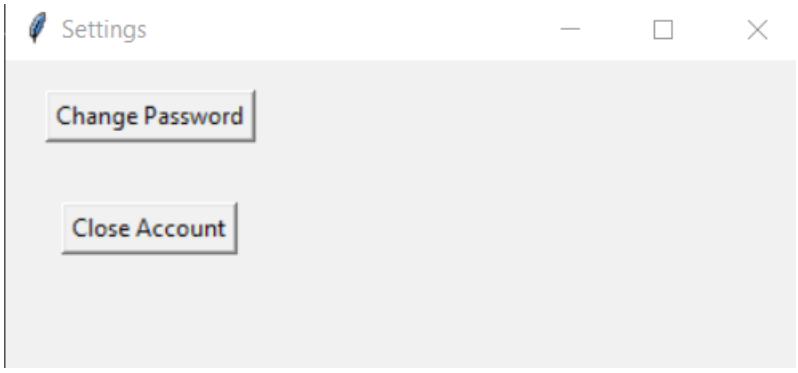

Fig 4.1.4

This is the settings window. It allows the user to change password and close account.

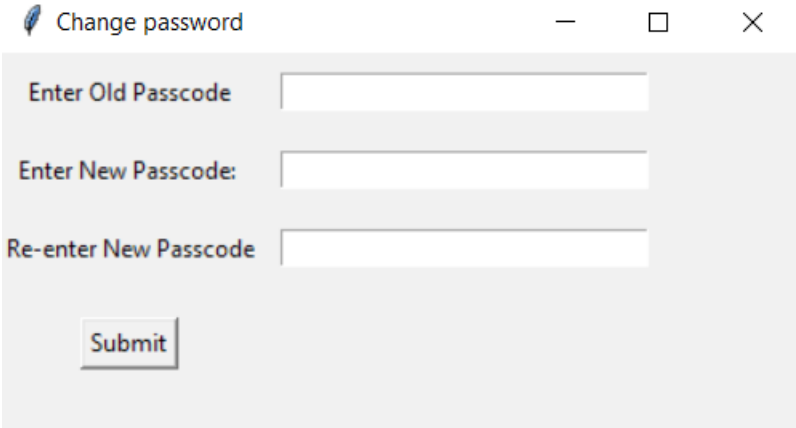

Fig 4.1.5

This is the window which is used to change password.

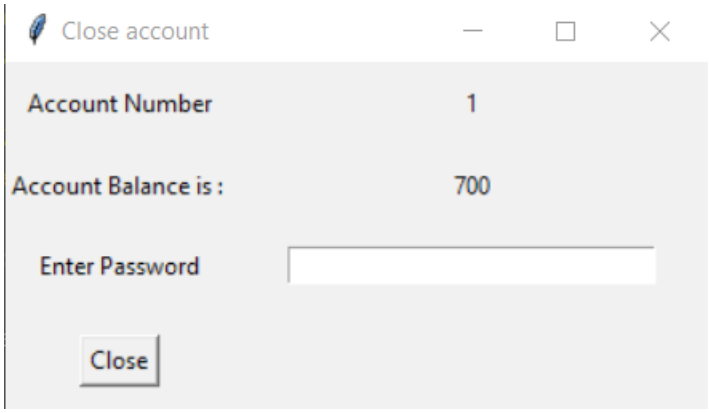

Fig 4.1.6

This is the close account window. The user can close his account through this window.

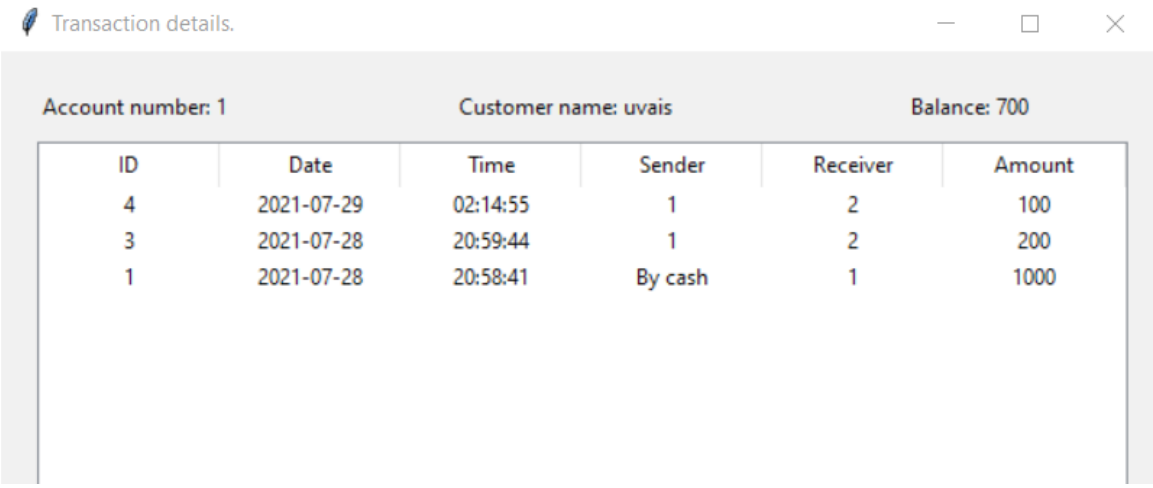

Fig 4.1.7

This window is the transaction details window, here the user can view the transaction history of his account.

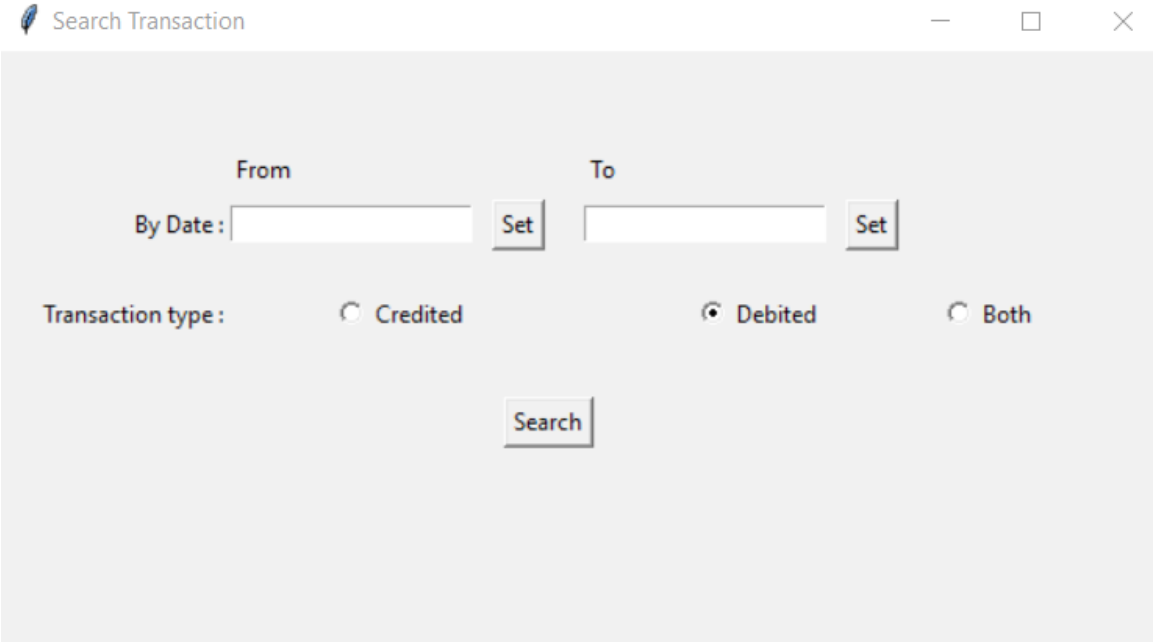

Fig 4.1.8

This is the search transaction window, in this window the user can search transactions by setting the date and account type.

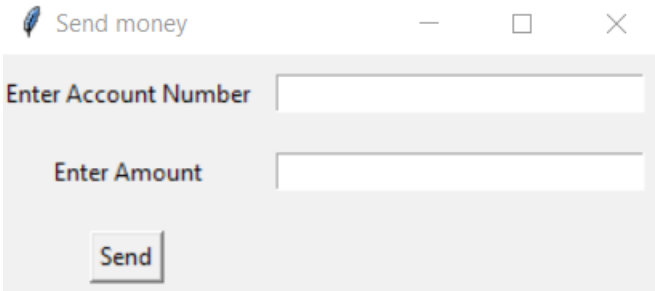

Fig 4.1.9

This is the send money window. Using this window, the user can send money to other accounts.

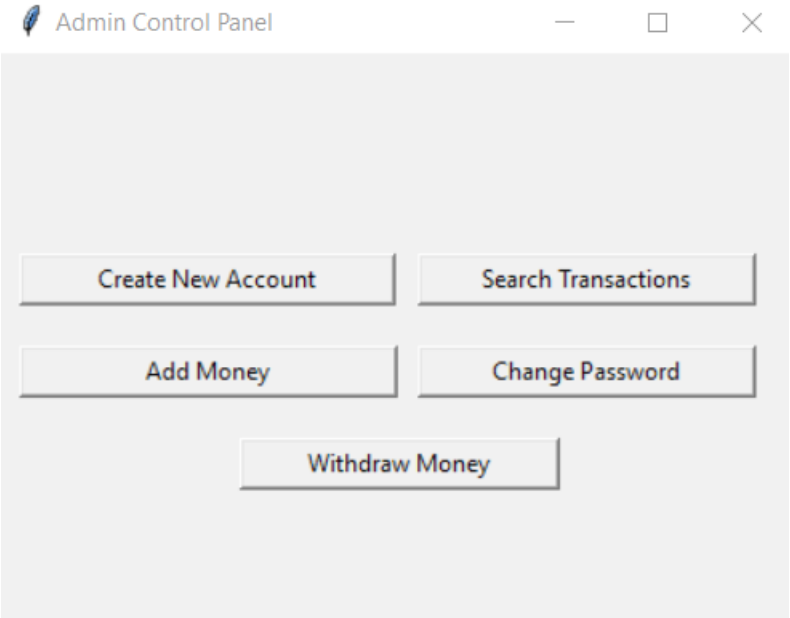

Fig 4.1.10

This window is the admin control panel window, through this window the admin can control various functions of user accounts.

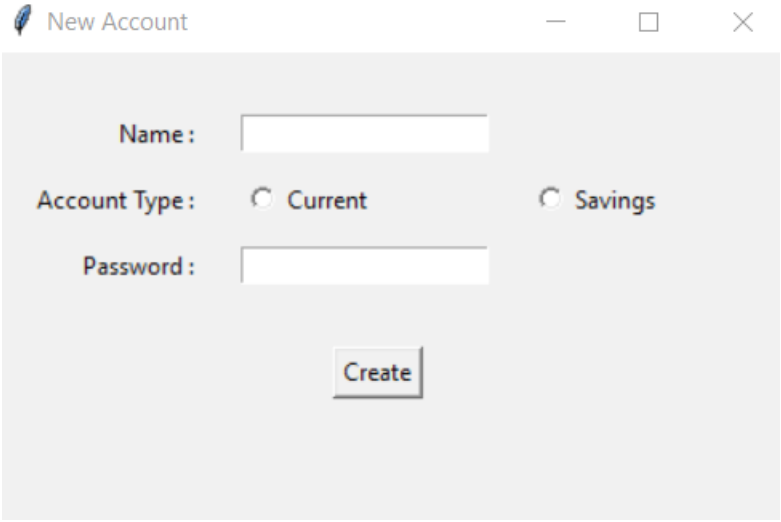

Fig 4.1.11

This window is used by admin to create a new customer account.

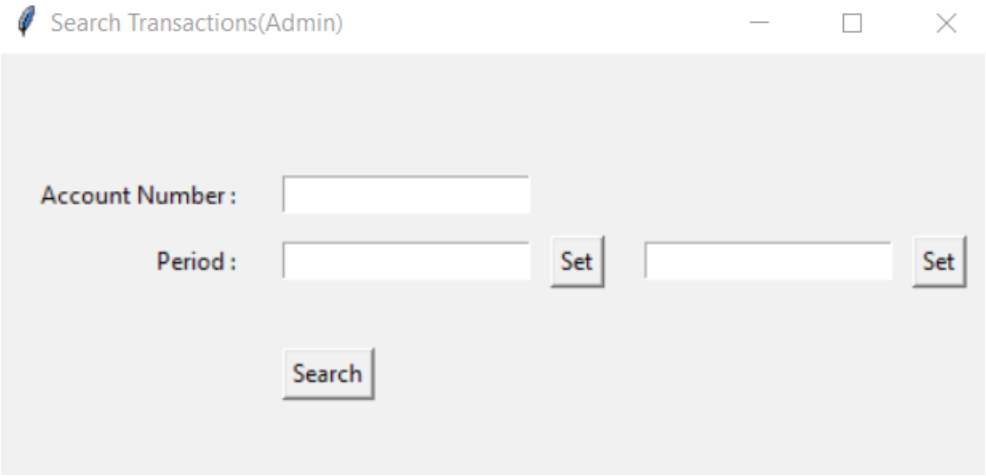

Fig 4.1.12

This window helps the admin to search transactions of the user.

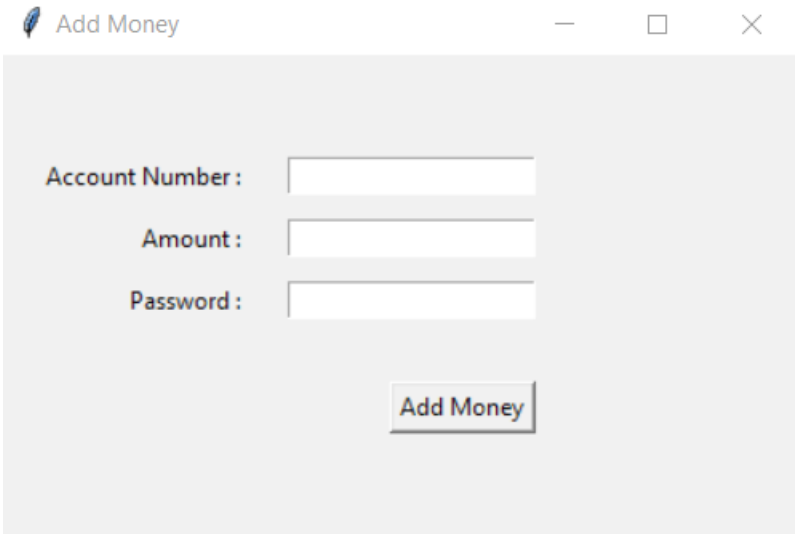

Fig 4.1.13

This window helps the admin to add money into the user account.

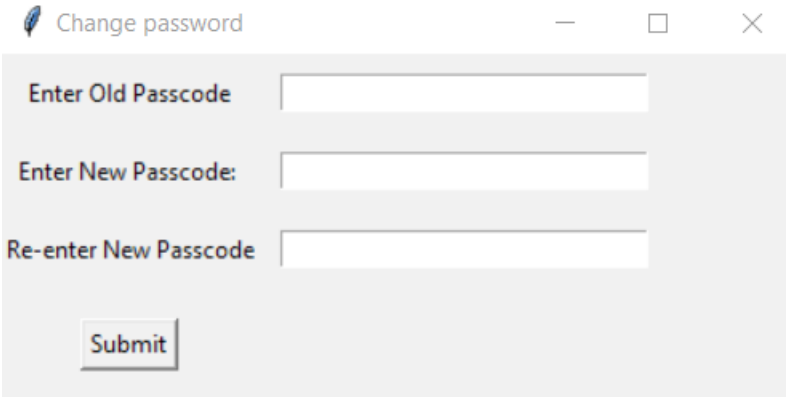

Fig 4.1.14

This window helps the admin to change his password.

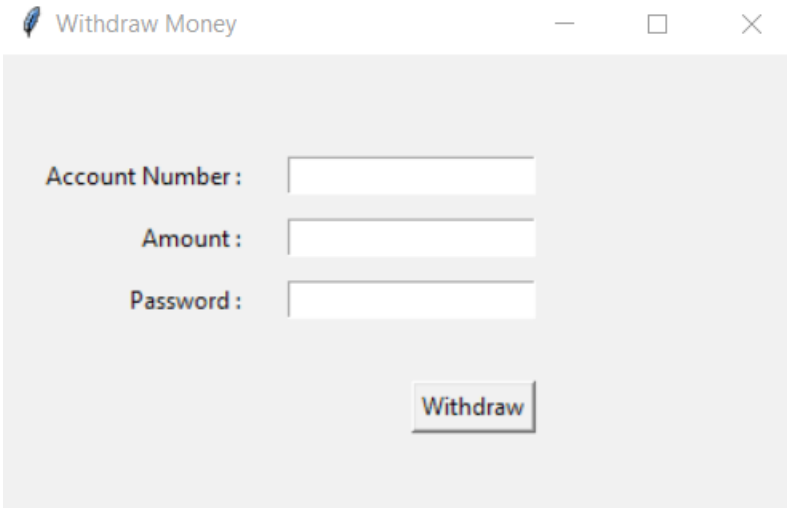

Fig 4.1.15

This window helps the admin to withdraw money from the user account for cash withdrawal by users.

```
1|Tester1@  |Tester1|savings|2021-07-25 16:56:36.889453|000119250
2|Tester2@  |Tester2|savings|2021-07-25 16:56:47.412669|000238350
3|Tester3@  |Tester3|current|2021-07-25 16:57:04.681515|000000830
```

Fig 4.1.16

This is how customer data is stored in files. Each field is separated by pipeline. Each record is separated by new line character.

```
1|Uvais0#   |Uvais|2021-07-25 16:54:31.090621
2|Admin2@   |Admin2|2021-07-25 16:54:51.005896
3|Admin3@   |Admin3|2021-07-25 16:55:13.040110
```

Fig 4.1.17

This is how admin data is stored in files.

```
1|2021-07-25|16:59:47|By cash|1|1000
2|2021-07-25|17:00:16|By cash|2|500000
3|2021-07-25|17:00:44|By cash|5|10000
```

Fig 4.1.18

Transaction data file contents. It stores transaction ID, transaction date and time stamp, receiver, sender, and amount.

```
1|1.txt|0
2|1.txt|47
3|1.txt|95
```

Fig 4.1.19

This is how level 1 index data is stored. It contains primary key, data file name and offset.

```
10|4.txt
13|5.txt
16|6.txt
```

Fig 4.1.20

This is how higher level index data are stored. It contains primary key and file name of lower level index.

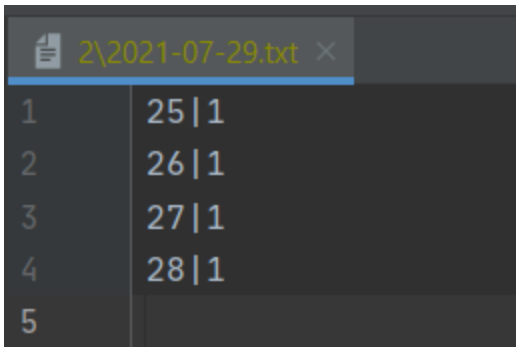

Fig 4.1.21

This is the secondary index file indexing all user transaction of account number 2, thus stored under folder 2. The file name is same as the transaction date. It contains transaction ID and type of transaction (0 = credit, 1 = debit).

4.2 Conclusion:

After completion of the project, the app can be run using a python interpreter. First, installer file is run to set up the entire app environment. Then an admin account is created. Using that account the admin can log in to admin panel and start using the app.

This app makes storing bank account details more efficient. Fetching of data from a large number of data file is made easier using multilevel indexing. It also provides all necessary security features through authentication.

Bibliography

1. https://docs.python.org/3/c-api/file.html
2. https://docs.python.org/3/library/filesys.html
3. https://docs.python.org/3/library/tk.html
4. https://www.cs.uct.ac.za/mit_notes/database/htmls/chp11.html#multilevel-indexes
5. https://www.cs.uct.ac.za/mit_notes/database/htmls/chp11.html#secondary-indexes