Faculty of Engineering Ain Shams University
AIN SHAMS UNIVERSITY
CHEP
Credit Hours Engineering Programs

Database I

PROJECT REPORT

To Be Presented To:
Dr. Gamal Ibrahim Abdel Elshafy
Eng. Sally Edward
|
By:

Youssef Assem Mohammed	16P6064
Karim Walid Elhammady	16P3090
Ahmed Sameh Shahin	16P6063
Moataz Khalid Zakaria	16P8244
Mohammed Ehab Elsaeed	16P8160
Eslam Ahmed Genedy	17P6043

Abstract

Database is absolutely an integral part of software system. The main aim of this project is providing information about the functioning of databases in a traditional banking system. Those databases should be able keep the day by day tally record as a complete banking. It can keep the information regarding account type, account opening form, deposits, and withdrawals. As well as keeping tabs on transactions, transaction reports, individual accounts opening forms, and group account.

TABLE OF CONTENTS

Contents

TABLE OF FIGURES

ER Diagram

Entity Relationship Diagram, also known as ERD, ER Diagram or ER model, is a type of structural diagram for use in database design. An ERD contains different symbols and connectors that visualize two important information: The major entities within the system scope, and the inter-relationships among these entities.

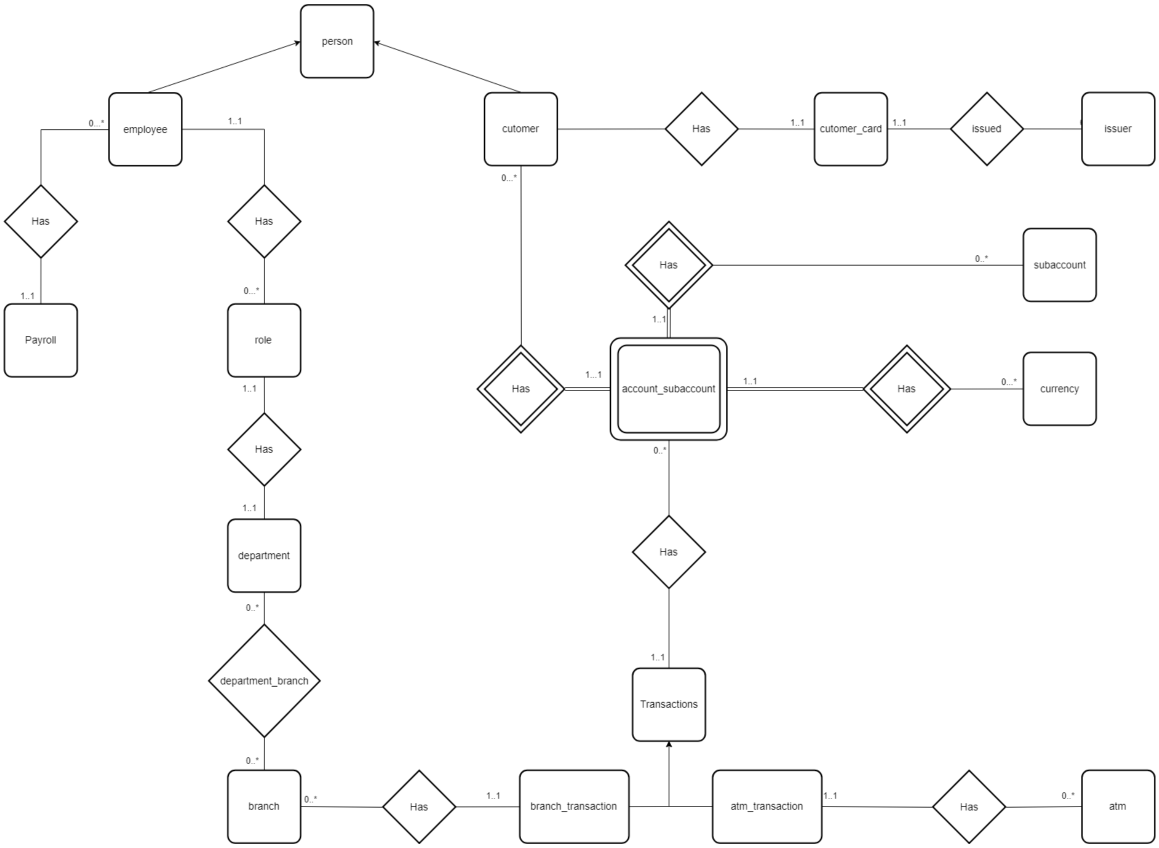


Figure 1 ER diagram

Fully utilizing an ER Diagram in database engineering guarantees you to produce high quality database design to use in database creation, management and maintenance. An ER model also provides a means for communication.

Physical Schema

Physical schema is a term used in data management to describe how data is to be represented and stored (files, indices, et al.) in secondary storage using a particular database management system (DBMS) (e.g., Oracle RDBMS, Sybase SQL Server, etc.).

THE CODE:

```
drop database if exists BANK;

create database BANK;

use BANK;

CREATE TABLE person (

   national_id INTEGER AUTO_INCREMENT,

   person_name VARCHAR(50),

   birth_date DATE,

   address VARCHAR(250),

   city VARCHAR(50),

   phone_number VARCHAR(15),

   email VARCHAR(25),

   PRIMARY KEY (national_id)

)  ENGINE=INNODB;

CREATE TABLE sub_accounts (

   id INTEGER PRIMARY KEY AUTO_INCREMENT,

   acc_name VARCHAR(50),

   description TINYTEXT,

   interest_rate DOUBLE,

   interest_interval INTEGER

)  ENGINE=INNODB;

CREATE TABLE currencies (

   id INTEGER PRIMARY KEY AUTO_INCREMENT,
    triple_code CHAR(3),
    exchange_rate DOUBLE
)  ENGINE=INNODB;
CREATE TABLE issuers (
    id INTEGER AUTO_INCREMENT,
    issuerName VARCHAR(25),
    PRIMARY KEY (id)
)  ENGINE=INNODB;
CREATE TABLE ATMs (
    id INTEGER PRIMARY KEY AUTO_INCREMENT,
    address VARCHAR(50),
    balance DOUBLE
)  ENGINE=INNODB;
CREATE TABLE branches (
    SWIFT VARCHAR(11) PRIMARY KEY,
    b_name VARCHAR(50),
    phone_number VARCHAR(15),
    postal_code VARCHAR(5),
    address VARCHAR(50),
    city VARCHAR(10),
    opening DATE
)  ENGINE=INNODB;
CREATE TABLE departments (
    id INTEGER PRIMARY KEY AUTO_INCREMENT,
    d_name VARCHAR(50)
)  ENGINE=INNODB;
```

```
CREATE TABLE department_branch (
   branch_swift VARCHAR(11),
   dept_id INTEGER,
   CONSTRAINT FOREIGN KEY (dept_id)
      REFERENCES departments (id)
      ON UPDATE CASCADE ON DELETE CASCADE,
   CONSTRAINT FOREIGN KEY (branch_SWIFT)
      REFERENCES branches (SWIFT)
      ON UPDATE CASCADE ON DELETE CASCADE,
      PRIMARY KEY(branch_swift,dept_id)
)  ENGINE=INNODB;
CREATE TABLE customers (
          id int,
   account_id INTEGER PRIMARY KEY AUTO_INCREMENT,
   CONSTRAINT FOREIGN KEY (id)
      REFERENCES person (national_id)
      ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
CREATE TABLE roles (
   id INTEGER PRIMARY KEY AUTO_INCREMENT,
   r_name VARCHAR(50),
   dept_id INTEGER,
   CONSTRAINT FOREIGN KEY (dept_id)
      REFERENCES departments (id)
      ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
```

```
CREATE TABLE employees (
    id INTEGER PRIMARY KEY,
    salary DOUBLE,
    role_id INTEGER,
    username varchar(45) UNIQUE,
            user_password varchar(45),
    CONSTRAINT FOREIGN KEY (id)
        REFERENCES person (national_id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (role_id)
        REFERENCES roles (id)
        ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
CREATE TABLE accounts_sub_accounts (
    account_id INTEGER,
    subaccount_id INTEGER AUTO_INCREMENT,
    currency_id INTEGER,
    balance DOUBLE,
    PRIMARY KEY (account_id , subaccount_id , currency_id),
    CONSTRAINT FOREIGN KEY (account_id)
        REFERENCES customers (account_id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (subaccount_id)
        REFERENCES sub_accounts (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (currency_id)
```

```
        REFERENCES currencies (id)
        ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
CREATE TABLE payroll (
    id INTEGER PRIMARY KEY AUTO_INCREMENT,
    employee_id INTEGER,
    payment_date DATE,
    bonuses DOUBLE,
    penalties DOUBLE,
    CONSTRAINT FOREIGN KEY (employee_id)
        REFERENCES employees (id)
        ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
create TABLE transactions (
    id INTEGER PRIMARY KEY AUTO_INCREMENT,
    from_account INTEGER,
    from_subAccount INTEGER,
    to_account INTEGER,
    to_subAccount INTEGER,
    amount DOUBLE,
    t_time DATETIME,
    CONSTRAINT FOREIGN KEY (from_account,from_subaccount)
        REFERENCES accounts_sub_accounts (account_id,subaccount_id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (to_account,to_subAccount)
        REFERENCES accounts_sub_accounts (account_id,subaccount_id)
        ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
CREATE TABLE atm_transaction (
    atm_id INT,
    transsaction_id INT,
    CONSTRAINT FOREIGN KEY (atm_id)
        REFERENCES atms (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (transaction_id)
        REFERENCES transactions (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
        PRIMARY KEY(atm_id,transaction_id)
)  ENGINE=INNODB;
CREATE TABLE branch_transaction (
    branch_swift VARCHAR(11),
    transaction_id INT,
    CONSTRAINT FOREIGN KEY (branch_swift)
        REFERENCES branches (swift)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (transaction_id)
        REFERENCES transactions (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
        PRIMARY KEY(branch_swift,transaction_id)
)  ENGINE=INNODB;
CREATE TABLE customer_cards (
    card_number INTEGER PRIMARY KEY AUTO_INCREMENT,
    account_id INTEGER,
    issuer_id INT,
    issue_date DATE,
    expirey_date DATE,
    daily_limit DOUBLE,
    card_type ENUM('debit', 'credit'),
    CVV VARCHAR(3),
    CONSTRAINT FOREIGN KEY (issuer_id)
        REFERENCES issuers (id)
        ON UPDATE CASCADE ON DELETE CASCADE,
    CONSTRAINT FOREIGN KEY (account_id)
        REFERENCES customers (account_id)
        ON UPDATE CASCADE ON DELETE CASCADE
)  ENGINE=INNODB;
```

The Logical Schema

A logical schema is a design-centric database structure built to meet your business requirements. It is a model that exists on a white board or in a diagraming tool. It is like the architect's drawings of your database.

Legend:

- Bold Underlined => Primary Key (may compose of foreign keys)
- Only Bold => Foreign Key

person (**national_id**, person_name, birth_date, address, city, phone_number, email)

sub_accounts (**id**, acc_name, description, interest_rate, interest_interval)

currencies (**id**, triple_code, exchange_rate)

issuers (**id**, issuerName)

ATMs (**id**, address, balance)

branches (**SWIFT**, b_name, phone_number, postal_code, address, city, opening)

departments (**id**, d_name)

department_branch (**branch_swift, dept_id**)

customers (**id**, account_id)

roles (**id**, r_name, **dept_id**)

employees (**id**, salary, **role_id**, username, user_password)

accounts_sub_accounts (**account_id, subaccount_id, currency_id,** balance)

payroll (**id, employee_id,** payment_date, bonuses, penalties)

transactions (**id**, **from_account, from_subAccount, to_account, to_subAccount**, amount, t_time)

atm_transaction (atm_id, **transaction_id**)

branch_transaction (branch_swift, **transaction_id**)

customer_cards **(card_number, account_id, issuer_id,** issue_date, expirey_date, daily_limit, card_type, CVV)

User Guide

Log In

The first screen all users will see is the login screen.

Employees can register or login to their accounts using this screen.

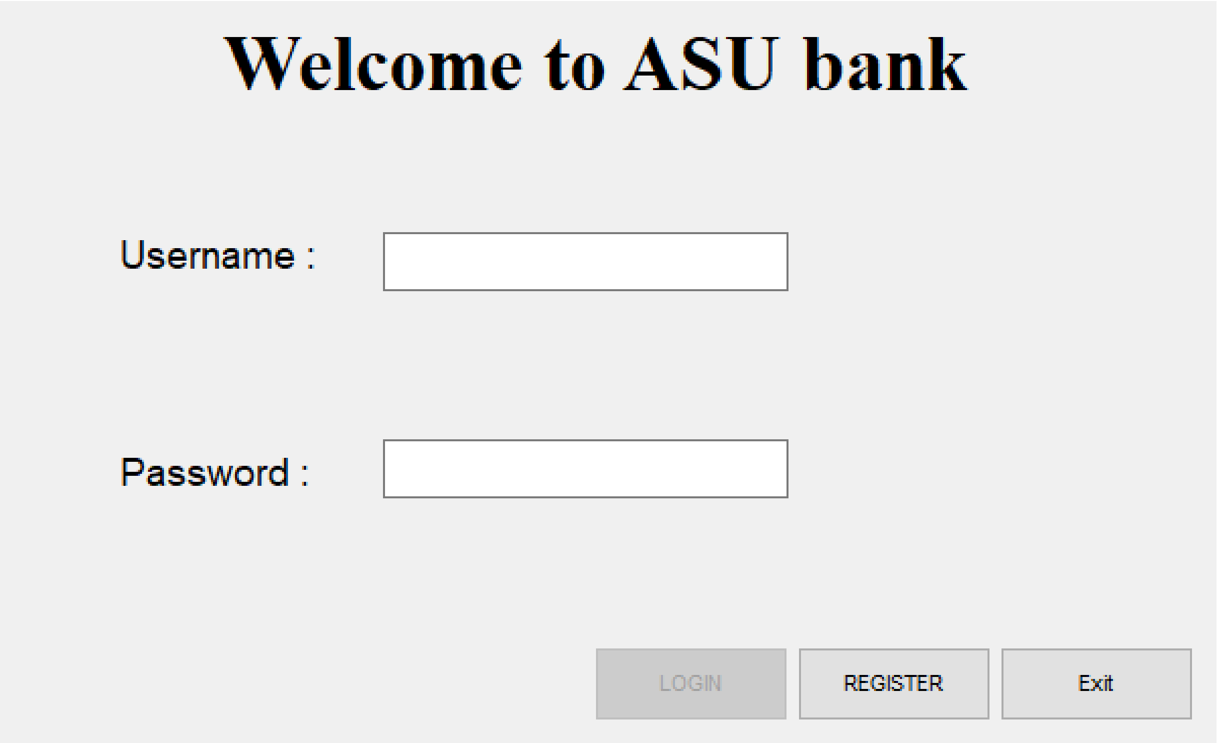
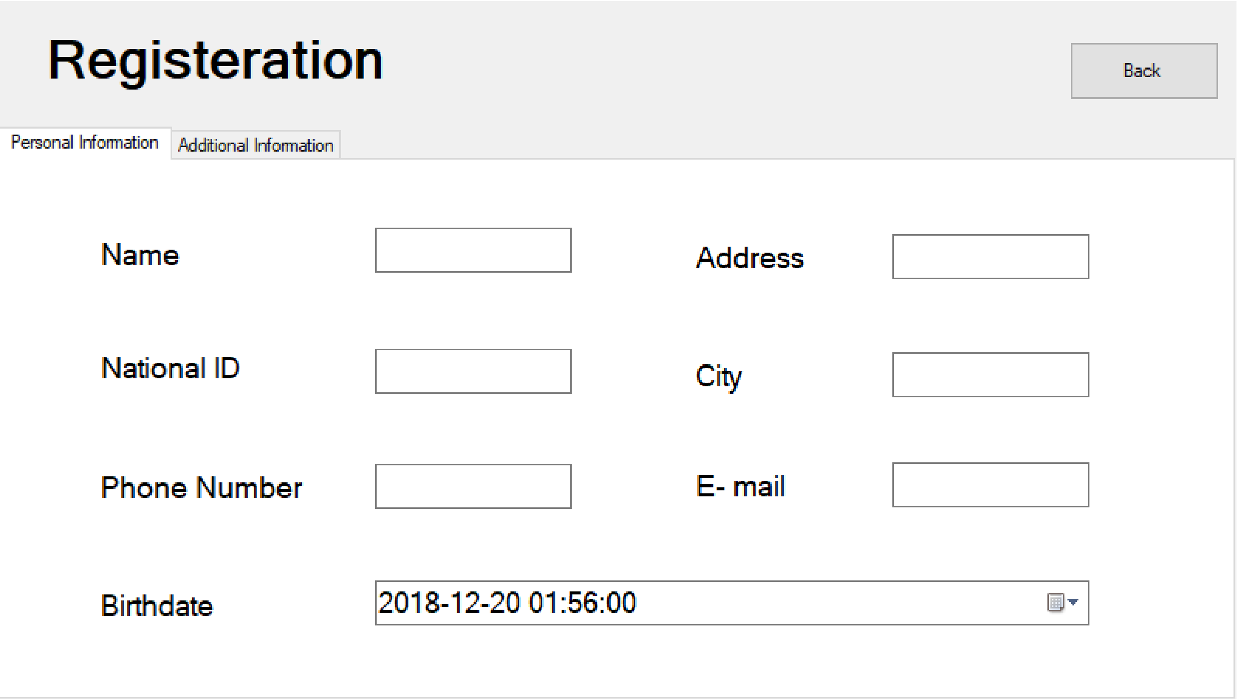


Figure 2 Log-in Screen

Registration

The Registration form allows for both employee and customer registration.

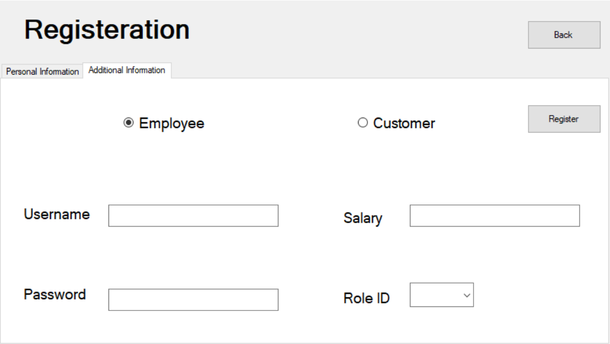
The first tab takes personal information about the person registering in the bank.



Birthdate: 2018-12-20 01:56:00

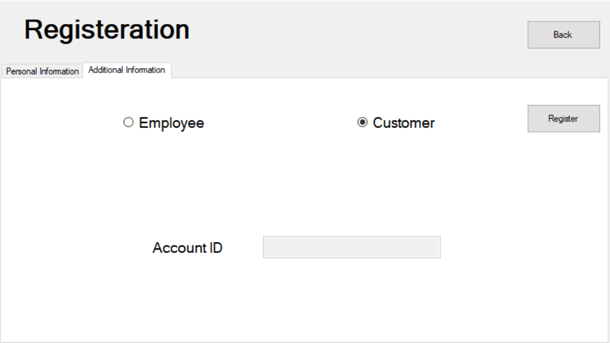
Figure 3 Registration Screen

The second tab gives the option either to register as an employee or a customer.



● Employee ○ Customer

Figure 5 Registration Screen 2



○ Employee ● Customer

Figure 4 Registration Screen 3

IN EMPLOYEE PANEL:

- Username will be the employee's username to log into system.
- Password will be the employee's password to log into system.
- Salary is the employee's salary.
- Role ID is his role in the bank (Manager, Teller, Bookkeeper, financial manager).

IN CUSTOMER PANEL:

- Account ID text field will show the new customer's account ID (his account number) after the new creation.

Services

After Login, all employees will go to the service page. However, according to each employee's role, the corresponding services will be available.

- Financial managers: can create sub account for customers and issue them card.
- Teller: allow customers to withdraw and deposit funds from and to their accounts.
- Bookkeeper: manages the transfer of funds between accounts.
- Manager: allow employees to adjust their personal information as well as their account username and password.

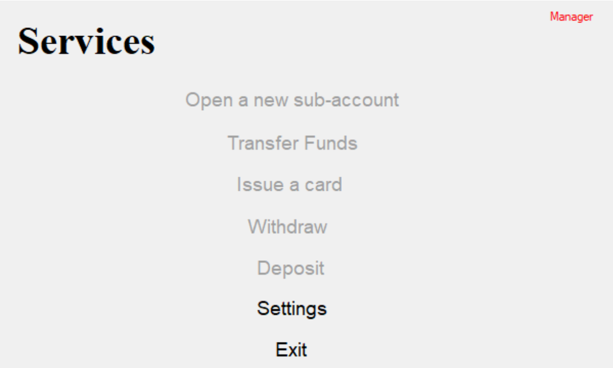


Figure 7 Services Screen 1

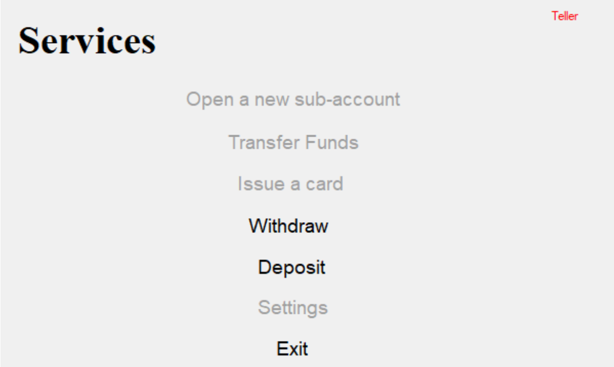


Figure 6 Services Screen 2

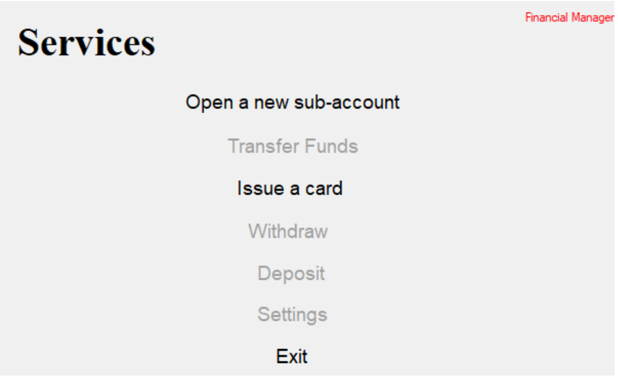


Figure 9 Services Screen 3

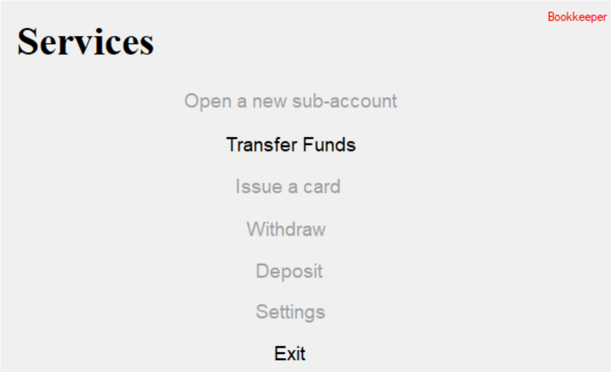


Figure 8 Services Screen 4

New Subaccount Form

In this form the financial manager can create a new sub account for a customer.

The customer must tell the financial manager his account ID and what type of account he wants (checking, saving, dividend and current account) and which currency he wants his account in.

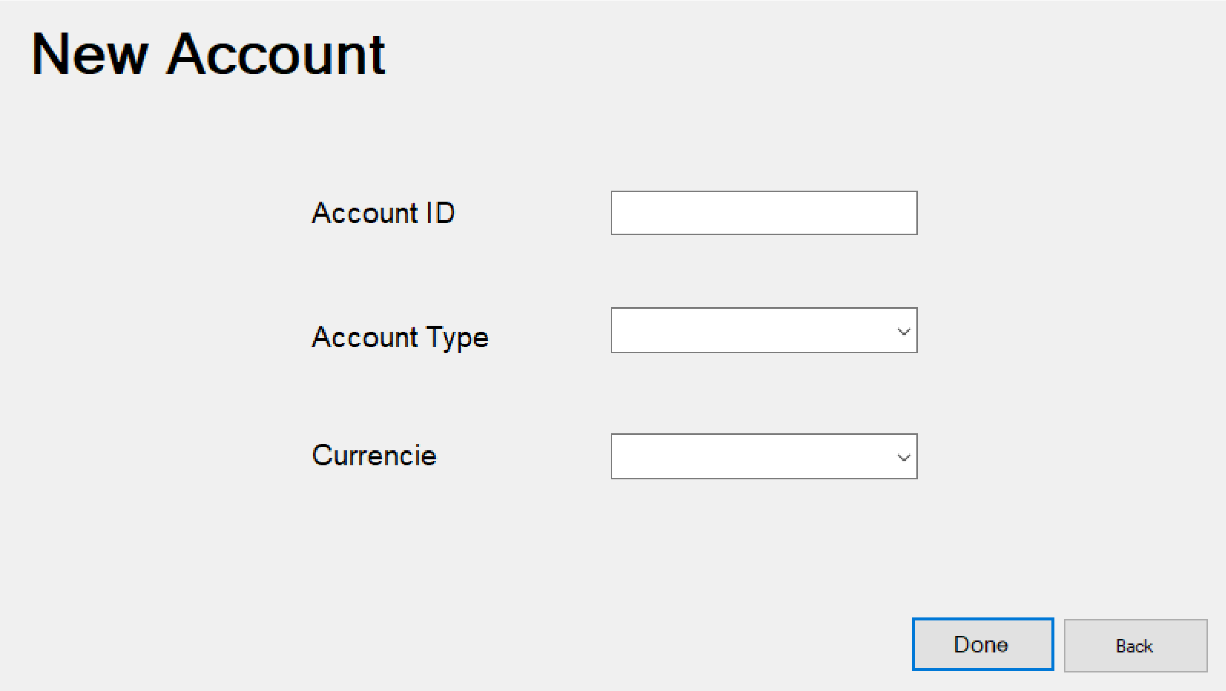
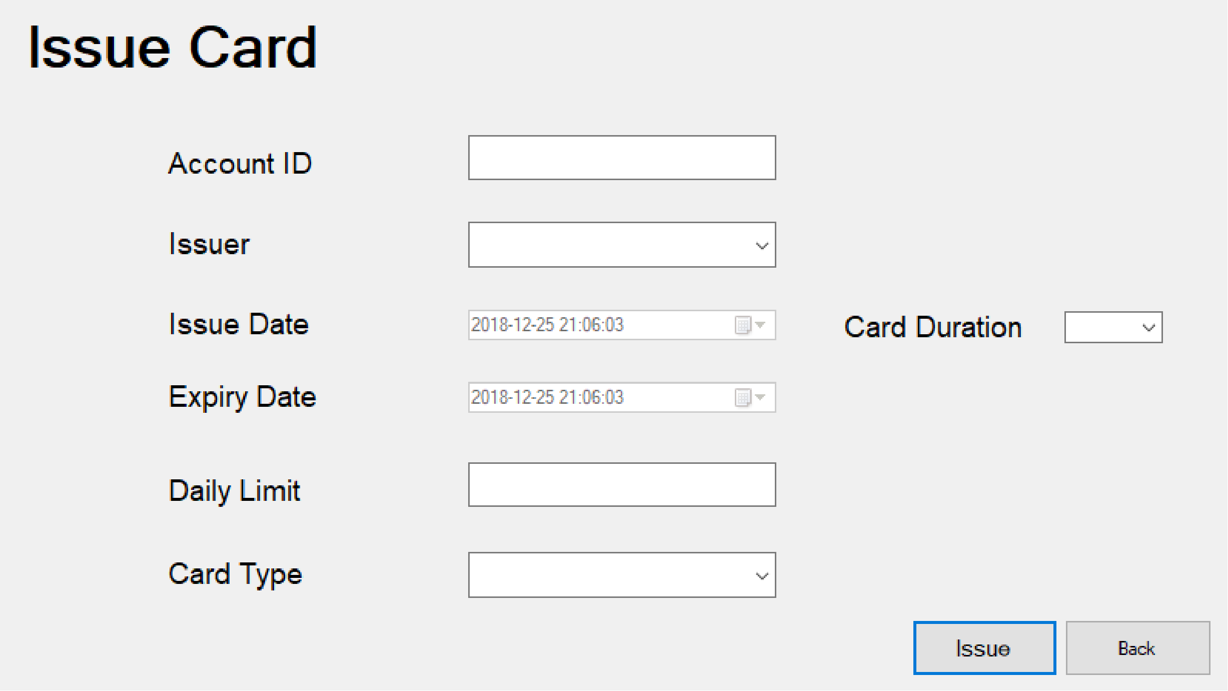


Figure 10 New Accounts Screen

Issue Card Form

When a customer needs to create a new card, he has to go to the financial manager who will ask him for his Account ID and which issuer he want to create the card with (visa, Mastercard, American Express, chase, discover, Citibank, capital one) , ,the card duration (3 , 5 or 7 years),Daily limit for card and finally the card type(debit or credit).



Issue Card

Account ID

Issuer

Issue Date 2018-12-25 21:06:03

Card Duration

Expiry Date 2018-12-25 21:06:03

Daily Limit

Card Type

Figure 11 Issue Card Screen

Withdraw Form

The teller is responsible for depositing and withdrawing money from customers' accounts.

In this Form, the customer tells the teller his account number and which account he wants to withdraw form and the amount to withdraw.

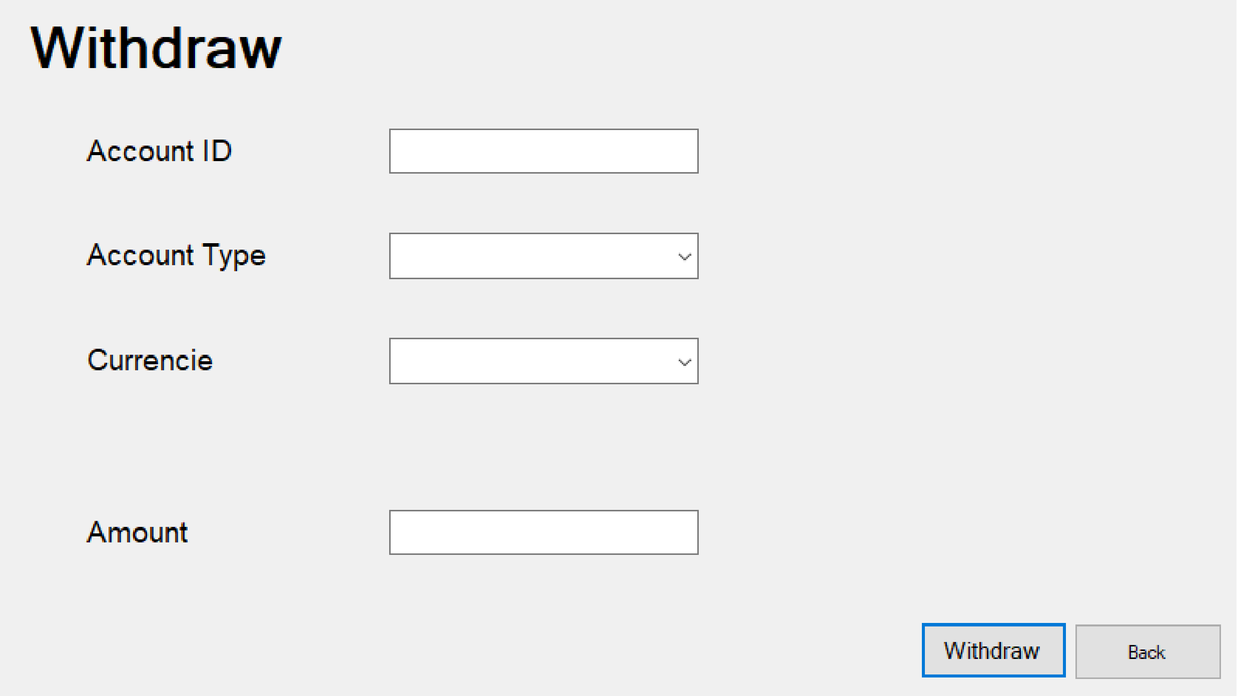


Figure 12 Withdraw Screen

Deposit Form

In this Form, the customer tells the teller his account number and which account he wants to deposit in and the amount to deposit.

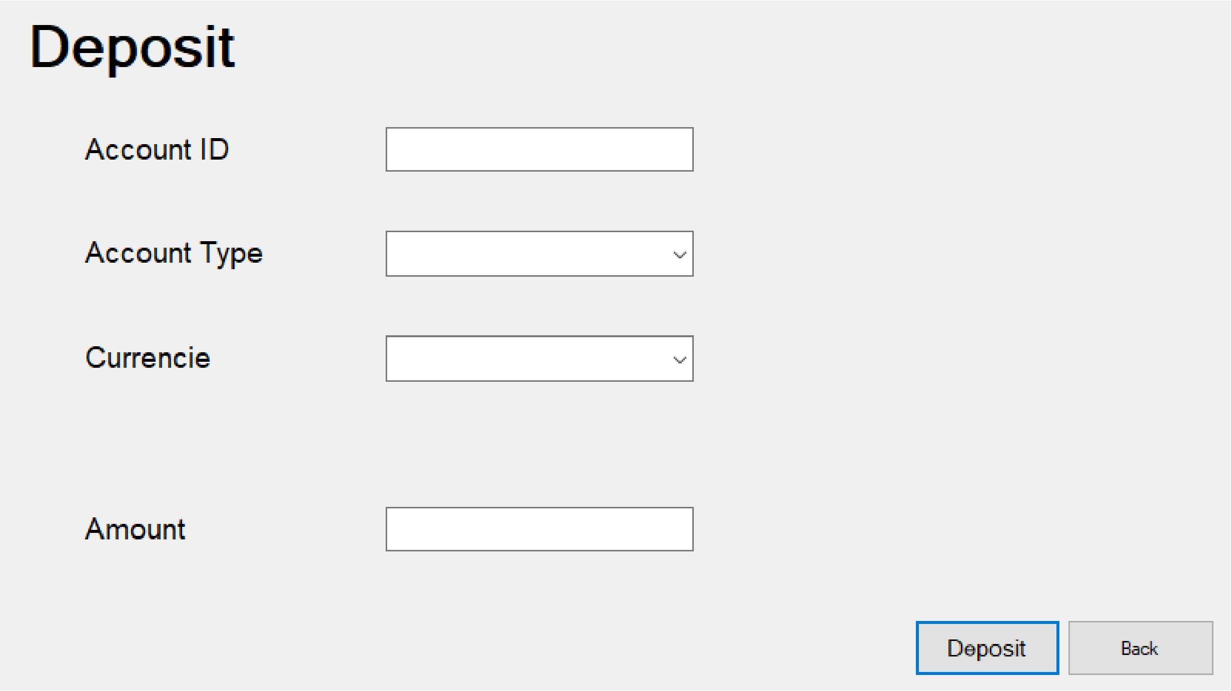
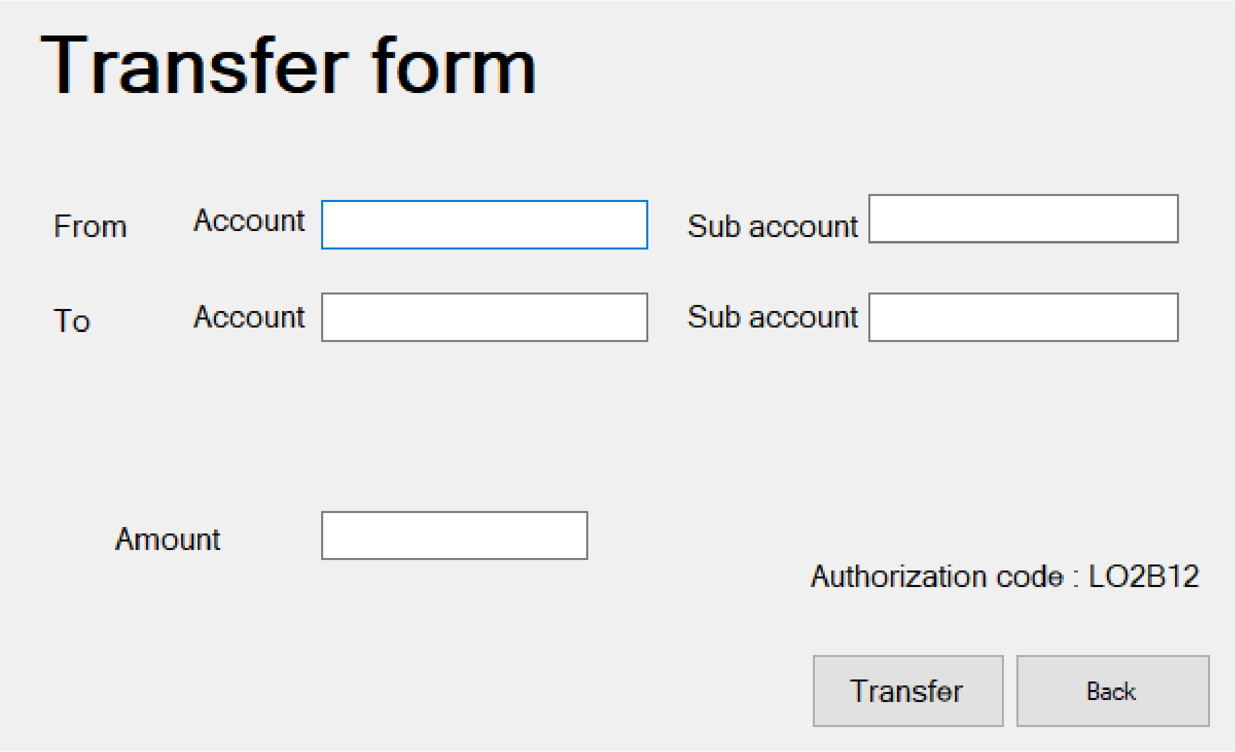


Figure 13 Deposit Screen

Transfer Fund Form

In this form, the bookkeeper can transfer fund from an account to another



Authorization code : LO2B12

Figure 14 Funds Screen

Settings Form

The manager is the only individual with access to the settings forms, he can change personnel information as well as username and password of employee account.

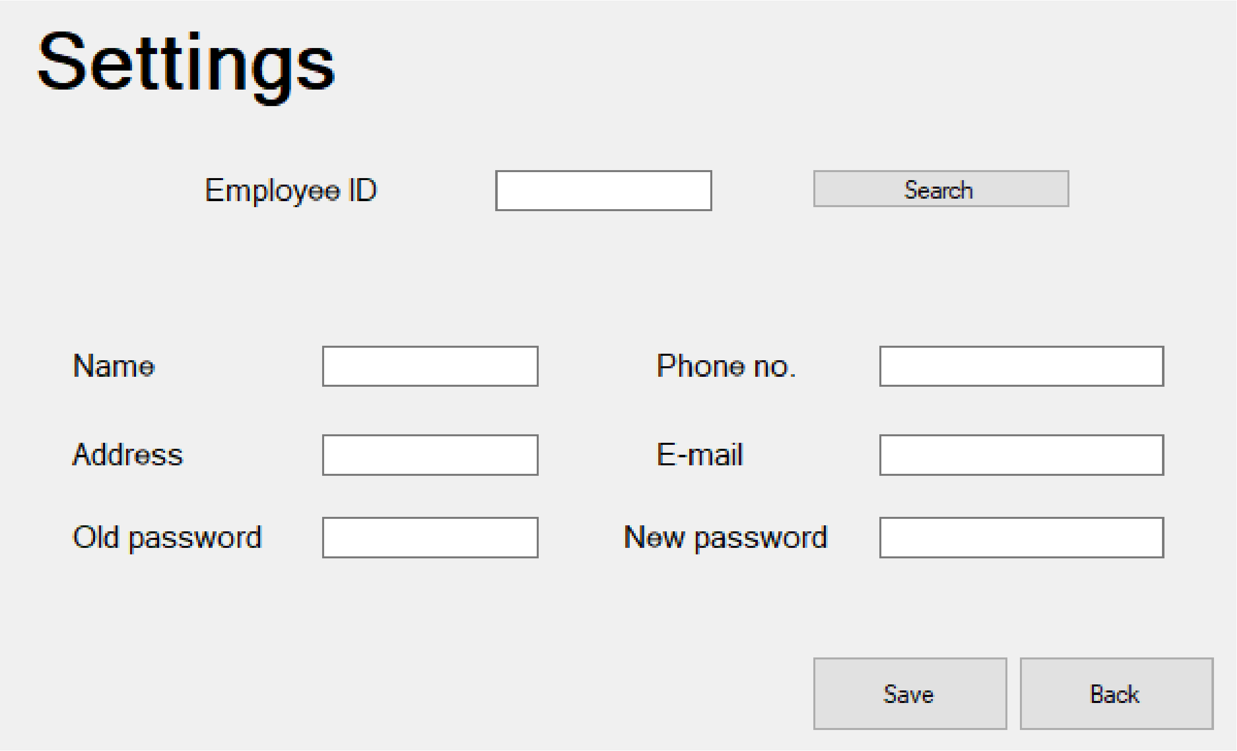


Figure 15 Settings Screen

View Transaction

In this form, the employee can view all transactions in a specific branch and filter them according to the amount.

He can also filter sort them by customer name or by the transaction amount and sort them accordingly.

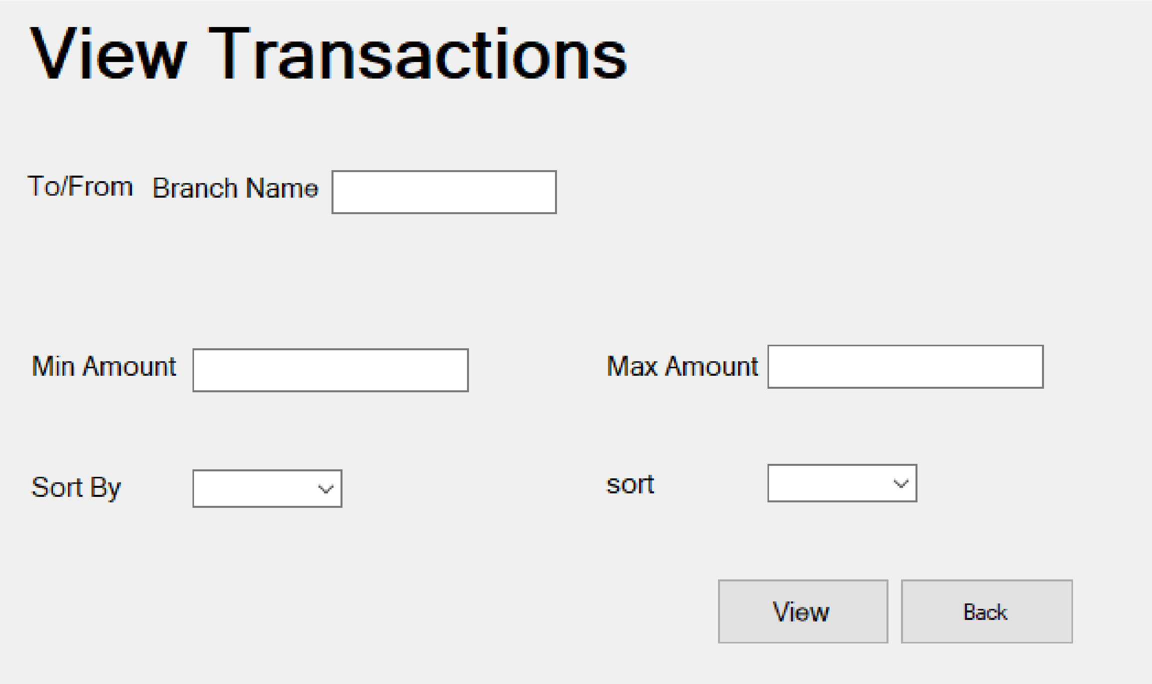


Figure 16 Transactions Screen

Statistical Form

This form provides statistical information, it computes the average of the balances in a specific city.

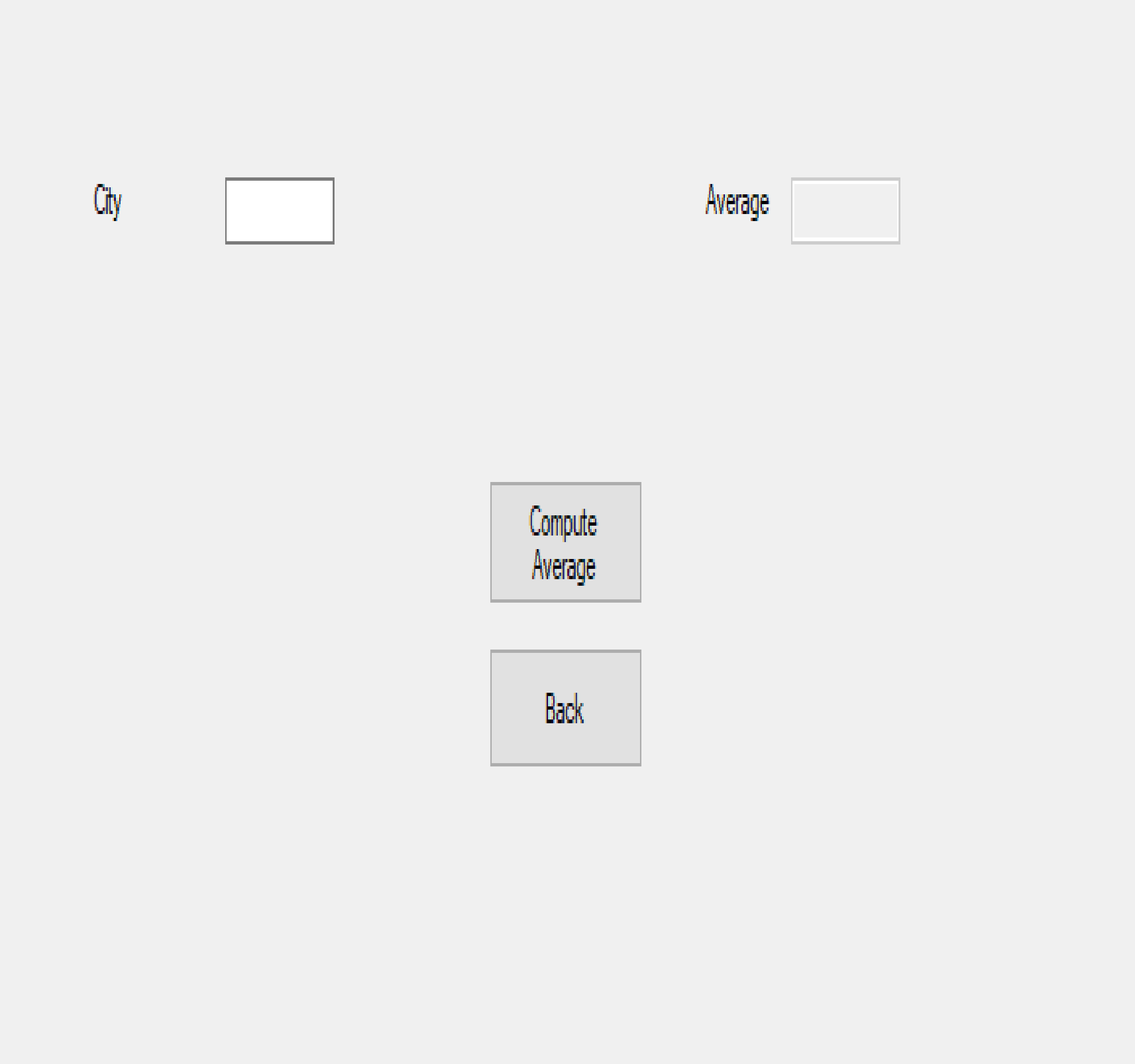


Figure 17 Statistical Screen

Issuers for Customers form

In this form the employee can view all customers in a specific city holding a card (debit or credit) form a specific issuer.

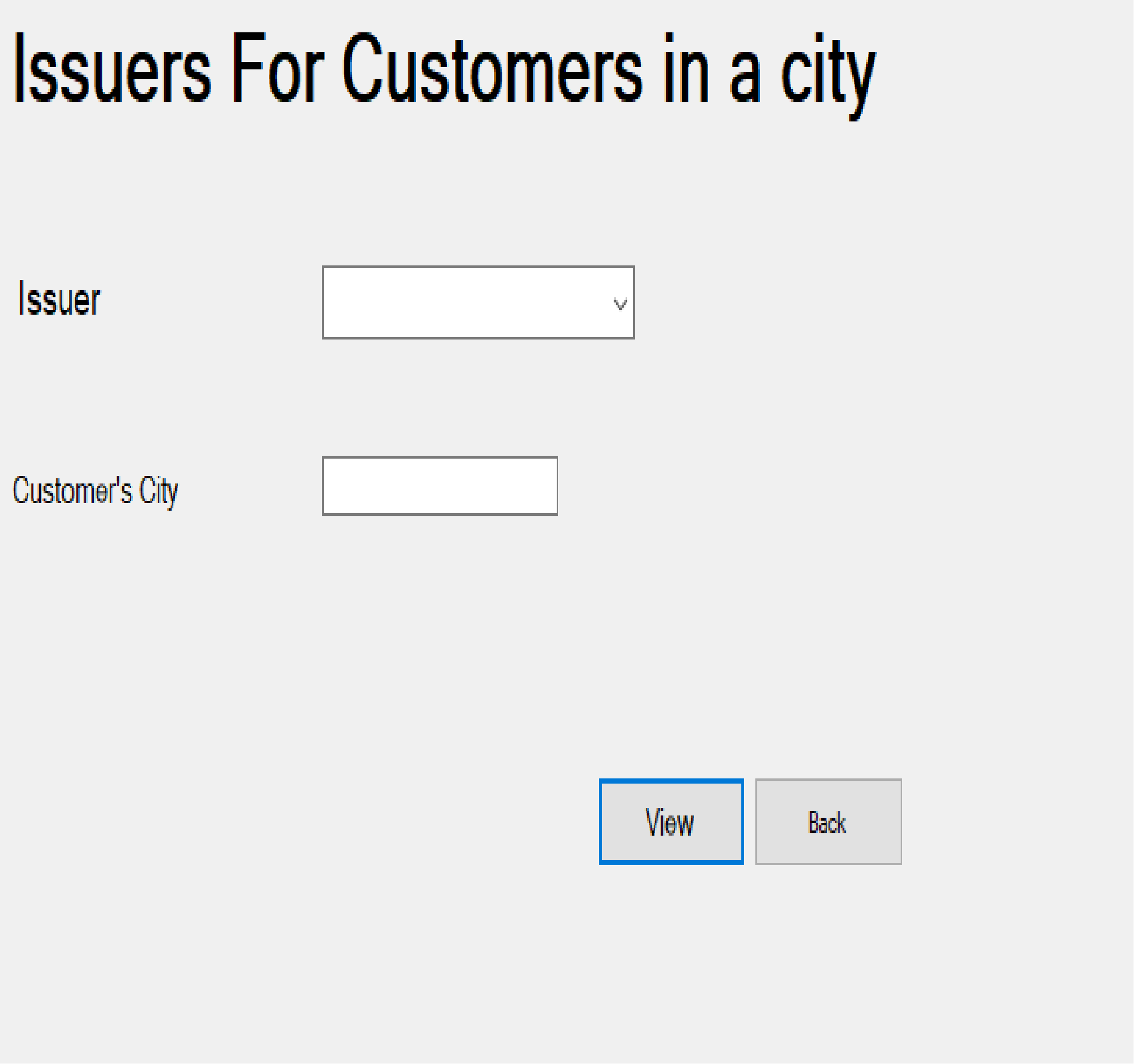


Figure 18 Issuers Screen

Complex Queries

Query 1:

Gets national ID, name of customers living in a specified city with cards issued by a specified issuer

```
SELECT
  national_id, person_name
FROM
  person
WHERE
  national_id IN (SELECT
      id
    FROM
      customers
    WHERE
      account_id IN (SELECT
          account_id
        FROM
          customer_cards
        WHERE
          issuer_id IN (SELECT
              id
            FROM
              issuers
            WHERE
```

```
              issuername LIKE 'Visa')))
    AND city LIKE 'Giza';
```

Query 2:

Find average of balances for all customers living in a specified city.

```
SELECT
  AVG(balance) AS Average
FROM
  accounts_sub_accounts
WHERE
  account_id IN (SELECT
      account_id
    FROM
      customers
    WHERE
      id IN (SELECT
          national_id
        FROM
          person
        WHERE
          city LIKE 'Aswan'));
```

Query 3:

Gets national ID, name, "from" account, "to" account, amount of transactions done at a specified branch with options to specify either or both minimum/maximum of amount and the option to order the result ascendingly or descendingly by person_name or amount (same applies to queries 4 to 7).

```
SELECT
  national_id, person_name, from_account, to_account, amount
```

```
FROM
    (person
    INNER JOIN customers ON person.national_id = customers.id
    INNER JOIN (SELECT
        *
    FROM
        transactions AS Tfrom
    WHERE
        Tfrom.id IN (SELECT
                transaction_id
            FROM
                branch_transaction
            WHERE
                branch_transaction.transaction_id IN (SELECT
                        SWIFT
                    FROM
                        branches
                    WHERE
                        b_name LIKE 'AlexPlaza'))) AS T ON T.from_account = customers.account_id)
WHERE
    amount >= 500 AND amount <= 5000
```

Query 4:

```
SELECT
    national_id, person_name, from_account, to_account, amount
FROM
```

```
    (person
    INNER JOIN customers ON person.national_id = customers.id
    INNER JOIN (SELECT
        *
    FROM
        transactions AS Tfrom
    WHERE
        Tfrom.id IN (SELECT
                transaction_id
            FROM
                branch_transaction
            WHERE
                branch_transaction.transaction_id IN (SELECT
                        SWIFT
                    FROM
                        branches
                    WHERE
                        b_name LIKE 'AlexPlaza'))) AS T ON T.from_account = customers.account_id)
WHERE
    amount >= 500 AND amount <= 5000
ORDER BY person_name ASC
```

Query 5:

```
SELECT
    national_id, person_name, from_account, to_account, amount
FROM
```

```
    (person
    INNER JOIN customers ON person.national_id = customers.id
    INNER JOIN (SELECT
        *
    FROM
        transactions AS Tfrom
    WHERE
        Tfrom.id IN (SELECT
                transaction_id
            FROM
                branch_transaction
            WHERE
                branch_transaction.transaction_id IN (SELECT
                        SWIFT
                    FROM
                        branches
                    WHERE
                        b_name LIKE 'AlexPlaza'))) AS T ON T.from_account = customers.account_id)
```

Query 6:

```
SELECT
    national_id, person_name, from_account, to_account, amount
FROM
    (person
    INNER JOIN customers ON person.national_id = customers.id
    INNER JOIN (SELECT
        *
    FROM
        transactions AS Tfrom
    WHERE
        Tfrom.id IN (SELECT
                transaction_id
            FROM
                branch_transaction
            WHERE
                branch_transaction.transaction_id IN (SELECT
                        SWIFT
                    FROM
                        branches
                    WHERE
                        b_name LIKE 'AlexPlaza'))) AS T ON T.from_account =
customers.account_id)
WHERE
    amount >= 500
```

Query 7:

```
SELECT
    national_id, person_name, from_account, to_account, amount
FROM
    (person
    INNER JOIN customers ON person.national_id = customers.id
    INNER JOIN (SELECT
        *
```

```
  FROM
    transactions AS Tfrom
  WHERE
    Tfrom.id IN (SELECT
        transaction_id
      FROM
        branch_transaction
      WHERE
        branch_transaction.transaction_id IN (SELECT
            SWIFT
          FROM
            branches
          WHERE
            b_name LIKE 'AlexPlaza'))) AS T ON T.from_account =
customers.account_id)
WHERE
  amount <= 5000
ORDER BY person_name ASC
```