VIETNAM NATIONAL UNIVERSITY - HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

DATABASE SYSTEMS

Assignment 2

CC02 - Group 03

Instructors/Lecturers: Nguyen Thi Ai Thao

Authors: Pham Anh Quan - 2053376
Nguyen Manh Dan - 2052932
Truong Nguyen Hung Thinh - 2053463
Nguyen Hai Dang - 2052444

HO CHI MINH CITY, 12/2022

Contents

1 Evaluation

No.	Full Name	Student ID	Work in the assignment	Evaluation
1	Nguyen Manh Dan	2052932	Create table write procedures and report	25%
2	Nguyen Hai Dang	2052444	Insert data and make presentation slides	25%
3	Pham Anh Quan	2053376	Design the app and connect it to the database	25%
4	Truong Nguyen Hung Thinh	2053463	Bugfix and write regex to check data input	25%

2 Requirements

2.1 Design the database

To begin, we create some tables in our database and insert data into them. Here are 3 out of 20 tables we have in our database.

```
CREATE TABLE USER (
    `ID` varchar(200) NOT NULL,
    `Name` varchar(200) NOT NULL,
    `Sex` varchar(200) NOT NULL,
    `Birthdate` datetime NOT NULL,
    `Phone` varchar(200) NOT NULL,
    `Email` varchar(200) NOT NULL,
    `Address` varchar(200) NOT NULL,
    PRIMARY KEY(`ID`)
);
```

```
CREATE TABLE SELLER (
    `ID` varchar(200) NOT NULL,
    `StoreName` varchar(200) NOT NULL,
    `StoreAddress` varchar(200) NOT NULL,
    `Logo` varchar(200) NOT NULL,
    `NationalID` varchar(200) NOT NULL,
    `TotalRevenue` double DEFAULT 0,
    PRIMARY KEY(`ID`),
    FOREIGN KEY (ID) REFERENCES USER(ID)
);
```

```
CREATE TABLE BUYER (
    `ID` varchar(200) NOT NULL,
    `AccountType` varchar(200) NOT NULL,
    `TotalMoneySpent` double DEFAULT 0,
    PRIMARY KEY(`ID`),
    FOREIGN KEY (ID) REFERENCES USER(ID)
);
```

Create tables

```
INSERT INTO `USER` (`ID`, `Name`, `Sex`, `Birthdate`, `Phone`, `Email`, `Address`) VALUES
                  ('000000', 'Pham Anh Quan', 'M', '1990-12-09', '0909xxxxx2', 'email0@gmail.com', '65 Ham Nghi, P.Nguyen Thai Binh, Q.1, TPH
                  ('000001', 'Nguyen Hai Dang', 'M', '2005-10-07', '0909xxxxx6', 'email1@gmail.com', '109 Tran Duy Hung, P.Trung Hoa, Q.Cau G
                  ('000002', 'Nguyen Manh Dan', 'M', '1995-06-17', '0909xxxxx8', 'email2@gmail.com', '1006 3 Thang 2, P.14, Q.10, TPHCM'),
                  ('000003', 'Truong Nguyen Hung Thinh', 'M', '1998-11-29', '0903xxxxx6', 'email3@gmail.com', '84 Nui Thanh, P.Hoa Cuong Bac,
                  ('000004', 'Dao Anh Tuan', 'M', '2001-03-23', '0903xxxxx1', 'email4@gmail.com', '241 Dao Duy Tu, P.7, Q.10, TPHCM'),
                  ('000005', 'Bui Quoc Minh Quan', 'M', '1987-01-21', '0902xxxxx7', 'email5@gmail.com', '491 Tran Nguyen Han, P.Ho Nam, Q.Le
                  ('000006', 'Nguyen Ha Trong Hieu', 'M', '1988-12-13', '0903xxxxx5', 'email6@gmail.com', '178 Le Thanh Nghi, P.Dong Tam, Q.H
                  ('000007', 'Chu Gia Vu Khanh', 'M', '2008-05-04', '0902xxxxx4', 'email7@gmail.com', '315 Vo Van Ngan, P.Linh Chieu, Q.Thu D
```

```
INSERT INTO `SELLER` (`ID`,`StoreName`, `StoreAddress`, `Logo`, `NationalID`) VALUES
                ('000000','BigAbs Shop', '558 Tran Hung Dao, P.14, Q.5, TPHCM', 'logo0.jpg', '0972xxxxxxx2'),
                ('000002','Dai Ca Laptop Shop', '165 3 Thang 2, P.11, Q.10, TPHCM', 'logo2.jpg', '0911xxxxxxx5'),
                ('000003','LadyMassage Store', '428 Truong Chinh, P.Hoa An, Q.Cam Le, Da Nang', 'logo3.jpg', '0872xxxxxxx1'),
                ('000004','Rau Xanh Groceries', '241 Dao Duy Tu, P.7, Q.10, TPHCM', 'logo4.jpg', '0199xxxxxxx8'),
                ('000005','Chopin Musical', '182 Nguyen Binh Khiem, P.Dang Giang, Q.Ngo Quyen, Hai Phong', 'logo5.jpg', '0595xxxxxxx6'),
                ('000007','Blackwheel Motors', '291 Pham Van Dong, P.1, Q.Go Vap, TP.HCM', 'logo7.jpg', '0447xxxxxxx3');
```

```
INSERT INTO `BUYER` (`ID`, `AccountType`) VALUES
                    ('000000', 'Silver'),
                    ('000001', 'Gold'),
                    ('000002', 'Gold'),
                    ('000003', 'Diamond'),
                    ('000004', 'Bronze'),
                    ('000005', 'Silver'),
                    ('000006', 'Bronze'),
                    ('000007', 'Diamond');
```

Insert data into tables

Next, to calculate derived attributes such as TotalCost or TotalMoneySpent then we will perform the following operations.

```
-- CALCULATE TOTAL COST OF EACH ORDER
CREATE TABLE TOTALCOST
    AS SELECT OrderID, USER.Name, sum(PRODUCT_ORDER.Quantity * Price) as TotalCost
        FROM PRODUCT_ORDER
            INNER JOIN PRODUCT ON ProductID = PRODUCT.ID
            INNER JOIN SHOP_ORDER ON OrderID = SHOP_ORDER.ID
            INNER JOIN USER ON BuyerID = USER.ID
                GROUP BY OrderID
                    ORDER BY OrderID ASC;
ALTER TABLE TOTALCOST
    ADD PRIMARY KEY (OrderID);
UPDATE SHOP_ORDER
    INNER JOIN TOTALCOST
        ON SHOP_ORDER.ID = TOTALCOST.OrderID
            SET SHOP_ORDER.TotalCost = TOTALCOST.TotalCost;
```

```
-- CALCULATE TOTAL MONEY SPENT BY EACH BUYER
CREATE TABLE TOTALMONEYSPENT
    AS SELECT USER.ID, USER.Name, sum(PRODUCT_ORDER.Quantity * Price) as TotalMoneySpent
        FROM PRODUCT_ORDER
            INNER JOIN PRODUCT ON ProductID = PRODUCT.ID
            INNER JOIN SHOP_ORDER ON OrderID = SHOP_ORDER.ID
            INNER JOIN USER ON BuyerID = USER.ID
                GROUP BY USER.Name
                    ORDER BY USER.ID ASC;
ALTER TABLE TOTALMONEYSPENT
    ADD PRIMARY KEY (ID);
UPDATE BUYER
    INNER JOIN TOTALMONEYSPENT
        ON BUYER.ID = TOTALMONEYSPENT.ID
            SET BUYER.TotalMoneySpent = TOTALMONEYSPENT.TotalMoneySpent;
```

Calculating derived attributes

Next, to SELECT, INSERT, DELETE, and UPDATE data in tables, we will put the queries into procedures and call the procedures to perform those operations. Each procedure has different inputs, for example:

- INSERT and UPDATE take in 8 inputs of 1 table name and 7 attribute values as our tables have at most 7 attributes. For tables that have all 7 attributes like USER, we will perform INSERT and UPDATE on those tables using all input attributes. Whereas for other tables, the excessive input attributes will be of value empty string.
- DELETE and SELECT take in 4 inputs of 1 table name and 3 primary key values as our tables have at most 3 primary keys. For tables that have 1 primary key like USER, we will compare only the first attribute with the primary key of that table. Similarly, for tables that have 2 or more primary keys like ROUTE then we will compare the first, second, and third attributes with the primary keys of that table.

```
DELIMITER $
CREATE PROCEDURE ADDVALUE (
    IN tablename varchar(200),
    IN att1 varchar(200),
    IN att2 varchar(200),
    IN att3 varchar(200),
    IN att4 varchar(200),
    IN att5 varchar(200),
    IN att6 varchar(200),
    IN att7 varchar(200)
)
BEGIN
    IF tablename = 'USER' THEN
    INSERT INTO USER VALUES (att1, att2, att3, str_to_date(att4, '%Y-%m-%d'), att5, att6, att7);
    ELSEIF tablename = 'SELLER' THEN INSERT INTO SELLER VALUES (att1, att2, att3, att4, att5, att6);
    ELSEIF tablename = 'PERSONAL_ACCOUNT' THEN INSERT INTO PERSONAL_ACCOUNT VALUES (att1, att2);
    ELSEIF tablename = 'BUSINESS_ACCOUNT' THEN INSERT INTO BUSINESS_ACCOUNT VALUES (att1, att2);
    ELSEIF tablename = 'BRAND_ACCOUNT' THEN INSERT INTO BRAND_ACCOUNT VALUES (att1, att2, att3);
```

```
DELIMITER $
CREATE PROCEDURE DELETEVALUE(
    IN tablename varchar(200),
    IN att1 varchar(200),
    IN att2 varchar(200),
    IN att3 varchar(200)
)
BEGIN
    IF tablename = 'USER' THEN DELETE FROM USER WHERE att1 = USER.ID;
    ELSEIF tablename = 'SELLER' THEN DELETE FROM SELLER WHERE att1 = SELLER.ID;
    ELSEIF tablename = 'PERSONAL_ACCOUNT' THEN DELETE FROM PERSONAL_ACCOUNT WHERE att1 = PERSONAL_ACCOUNT.ID;
    ELSEIF tablename = 'BUSINESS_ACCOUNT' THEN DELETE FROM BUSINESS_ACCOUNT WHERE att1 = BUSINESS_ACCOUNT.ID;
    ELSEIF tablename = 'BRAND_ACCOUNT' THEN DELETE FROM BRAND_ACCOUNT WHERE att1 = BRAND_ACCOUNT.ID;
```

```
DELIMITER $
CREATE PROCEDURE SEARCHVALUE(
    IN tablename varchar(200),
    IN att1 varchar(200),
    IN att2 varchar(200),
    IN att3 varchar(200)
)
BEGIN
    IF tablename = 'USER' THEN SELECT * FROM USER WHERE att1 = USER.ID;
    ELSEIF tablename = 'SELLER' THEN SELECT * FROM SELLER  WHERE att1 = SELLER.ID;
    ELSEIF tablename = 'PERSONAL_ACCOUNT' THEN SELECT * FROM PERSONAL_ACCOUNT WHERE att1 = PERSONAL_ACCOUNT.ID;
    ELSEIF tablename = 'BUSINESS_ACCOUNT' THEN SELECT * FROM BUSINESS_ACCOUNT  WHERE att1 = BUSINESS_ACCOUNT.ID;
    ELSEIF tablename = 'BRAND_ACCOUNT' THEN SELECT * FROM BRAND_ACCOUNT WHERE att1 = BRAND_ACCOUNT.ID;
```

```
DELIMITER $
CREATE PROCEDURE UPDATEVALUE(
    IN tablename varchar(200),
    IN att1 varchar(200),
    IN att2 varchar(200),
    IN att3 varchar(200),
    IN att4 varchar(200),
    IN att5 varchar(200),
    IN att6 varchar(200),
    IN att7 varchar(200)
)
BEGIN
    IF tablename = 'USER' THEN UPDATE USER SET
        USER.ID = att1,
        USER.Name = att2,
        USER.Sex = att3,
        USER.Birthdate = str_to_date(att4, '%Y-%m-%d'),
        USER.Phone = att5,
        USER.Email = att6,
        USER.Address = att7
            WHERE att1 = USER.ID;
    ELSEIF tablename = 'SELLER' THEN UPDATE SELLER SET
        SELLER.ID = att1,
        SELLER.StoreName = att2,
        SELLER.StoreAddress = att3,
        SELLER.Logo = att4,
        SELLER.NationalID = att5,
        SELLER.TotalRevenue = att6
             WHERE att1 = SELLER.ID;
```

Procedure

2.2 Design the application

We implement this assignment using Python and MySQL, with the GUI of the app made with Tkinter. The source code is attached to this report.

Below are the codes written in python to call the procedures in the database.

```python
def add(self, child):
    attribute1 = str(child.entry1.get())
    attribute2 = str(child.entry2.get())
    attribute3 = str(child.entry3.get())
    attribute4 = str(child.entry4.get())
    attribute5 = str(child.entry5.get())
    attribute6 = str(child.entry6.get())
    attribute7 = str(child.entry7.get())

    attributes = list(chain(attribute1, attribute2, attribute3, attribute4, attribute5, attribute6, attribute7))

    for num in range(0, child.NUMBER_OF_ATTRIBUTE):
        if (len(attributes[num]) == 0 or attributes[num].isspace() == 1):
            messagebox.showinfo("Error!", "Please fill up the blank entry.")
            return
    else:
        try:
            dbConnection = self.connection()
            cursor = dbConnection.cursor()
            cursor.callproc('ADDVALUE', (child.TABLE, attribute1, attribute2, attribute3, attribute4, attribute5, attribute6, attribute7,))
            dbConnection.commit()
            dbConnection.close()
        except:
            messagebox.showinfo("Error!", "Value in one of the fields already exists, is in wrong format or can not be fixed.")
            return

    self.refreshTable(child)
```

```python
def delete(self, child):
    decision = messagebox.askquestion("Warning!", "This will delete the data directly from the table. Do you want to continue?")
    if decision != "yes":
        return
    else:
        if not child.tree.selection():
            messagebox.showinfo("Error!", "You have not chosen the data from the table.")
            return
        else:
            selectedItem = child.tree.selection()[0]
            deleteData = str(selectedItem.split( )[0])
            try:
                dbConnection = self.connection()
                cursor = dbConnection.cursor()
                cursor.callproc('DELETEVALUE', (child.TABLE, deleteData, "", "",))
                dbConnection.commit()
                dbConnection.close()
            except:
                messagebox.showinfo("Error!", "Sorry an error occurred!")
                return

            self.refreshTable(child)
```

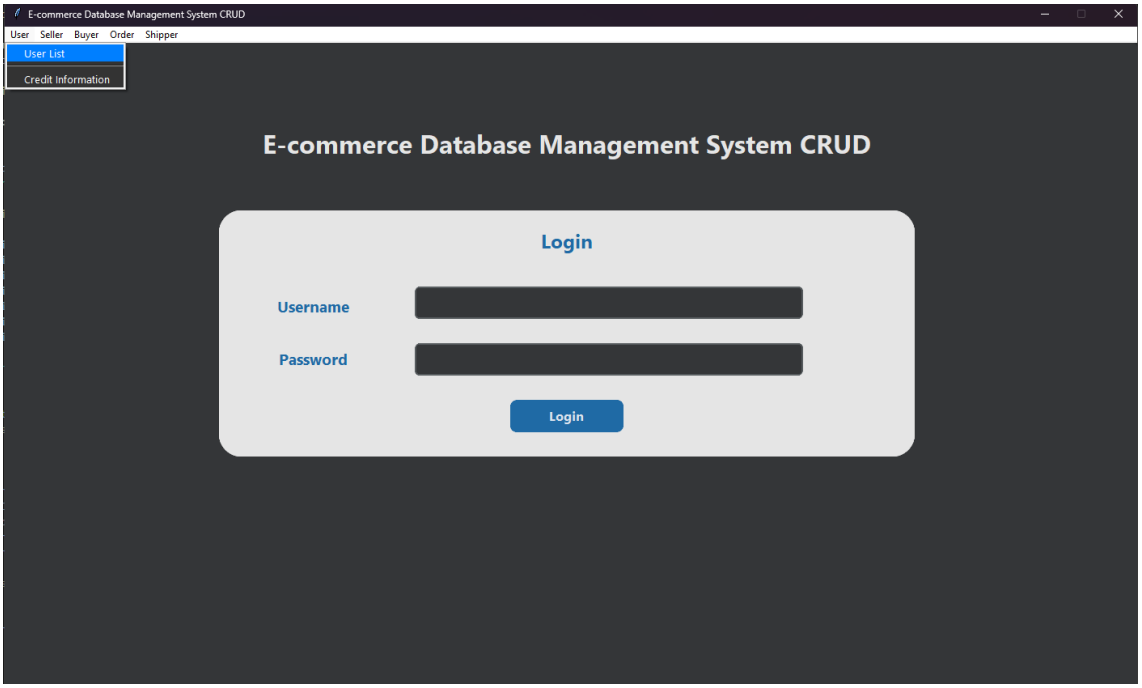
```
def search(self, child):
    attribute1 = str(child.entry1.get())
    attribute2 = str(child.entry2.get())
    attribute3 = str(child.entry3.get())

    dbConnection = self.connection()
    cursor = dbConnection.cursor()
    cursor.callproc('SEARCHVALUE', (child.TABLE, attribute1, attribute2, attribute3,))
    try:
        result = cursor.fetchall()

        for num in range(0, child.NUMBER_OF_ATTRIBUTE):
            self.setPlaceHolder(child, result[0][num], (num + 1))

        dbConnection.commit()
        dbConnection.close()
    except:
        messagebox.showinfo("No data found.")
```

```python
def update(self, child):
    selectedId = ""
    try:
        selectedItem = child.tree.selection()[0]
        selectedId = str(selectedItem.split( )[0])
    except:
        messagebox.showinfo("Error!", "Please select a data row.")

    attribute1 = str(child.entry1.get())
    attribute2 = str(child.entry2.get())
    attribute3 = str(child.entry3.get())
    attribute4 = str(child.entry4.get())
    attribute5 = str(child.entry5.get())
    attribute6 = str(child.entry6.get())
    attribute7 = str(child.entry7.get())

    attributes = list(chain(attribute1, attribute2, attribute3, attribute4, attribute5, attribute6, attribute7))

    for num in range(0, child.NUMBER_OF_ATTRIBUTE):
        if (len(attributes[num]) == 0 or attributes[num].isspace() == 1):
            messagebox.showinfo("Error!", "Please fill up the blank entry.")
            return
    else:
        try:
            dbConnection = self.connection()
            cursor = dbConnection.cursor()
            cursor.callproc('UPDATEVALUE', (child.TABLE, attribute1, attribute2, attribute3, attribute4, attribute5, attribute6, attribute7,))
            dbConnection.commit()
            dbConnection.close()
        except:
            messagebox.showinfo("Error!", "Value in one of the fields already exists, is in wrong format or can not be fixed.")
            return

    self.refreshTable(child)
```

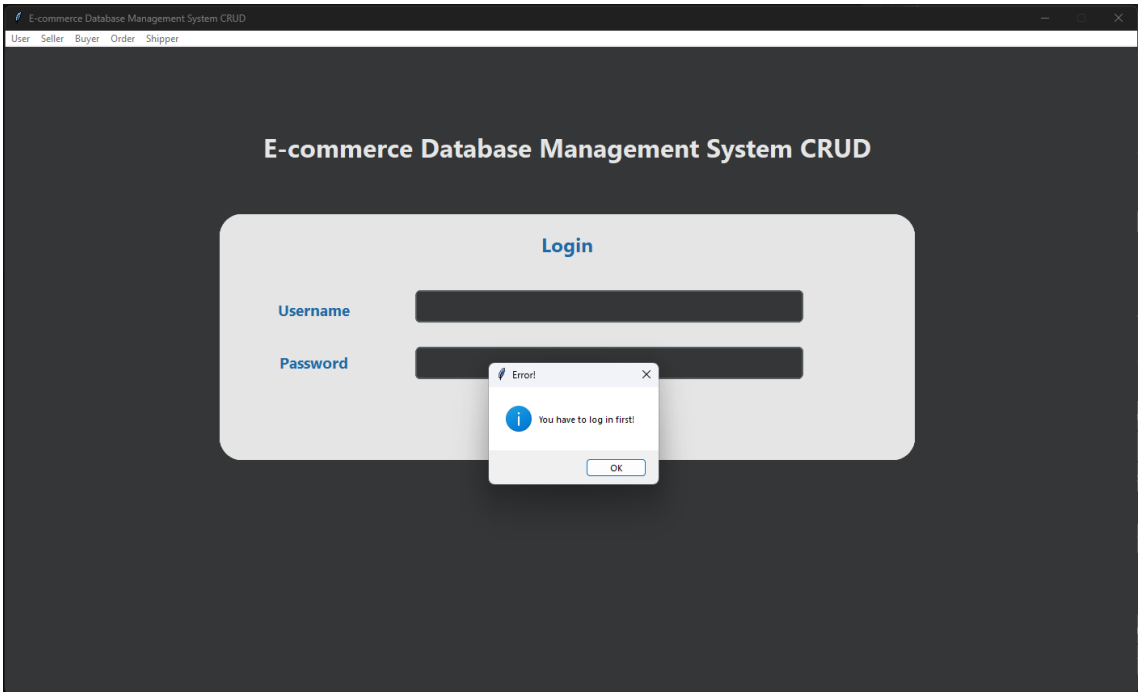
Python codes to call procedures

We also separate the user of this app into 3 classes:

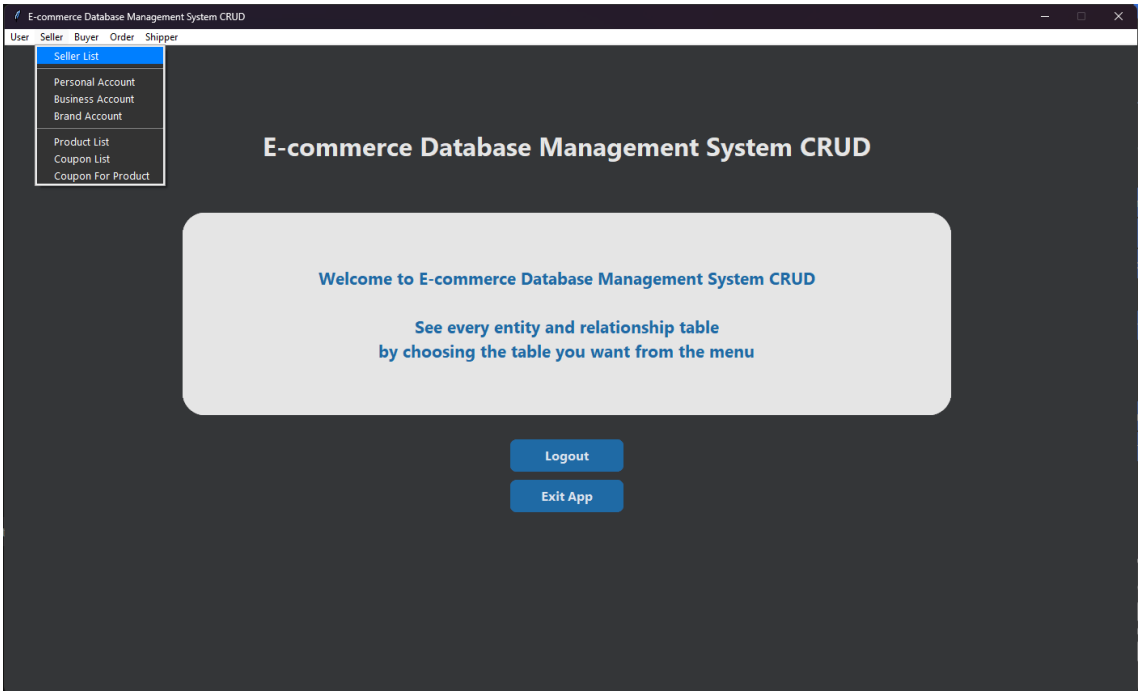
- Admin: Can view all tables.
- Seller: Can view all tables related to them, orders, and shippers.
- Buyer: Can view all tables related to them and orders.



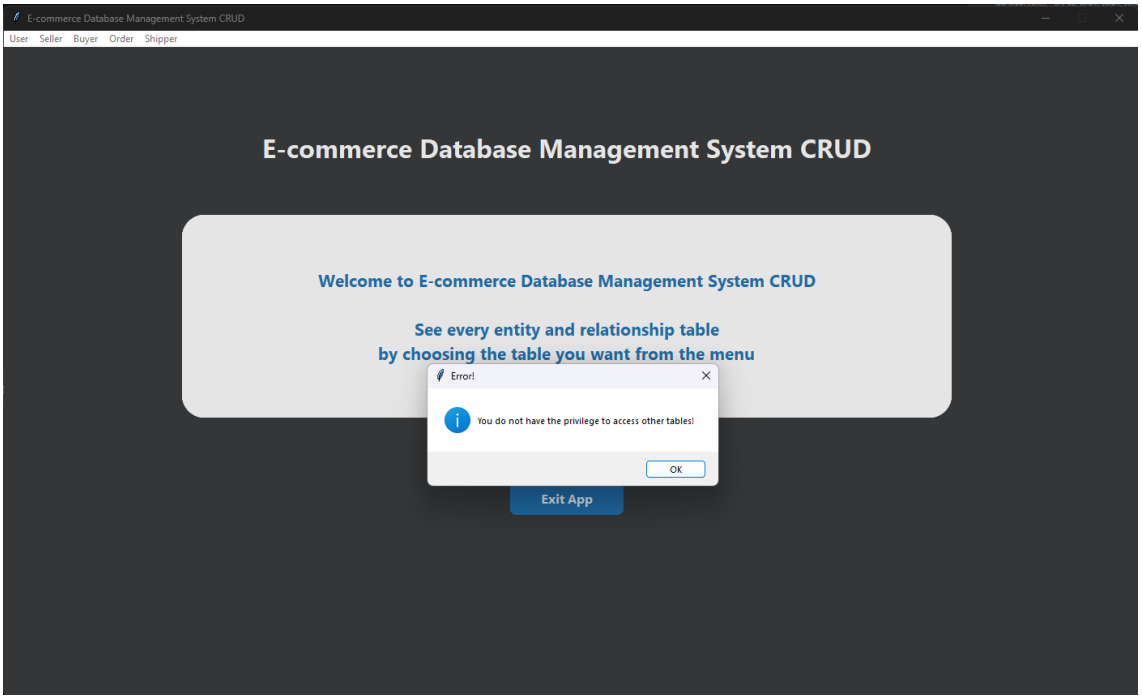
App GUI



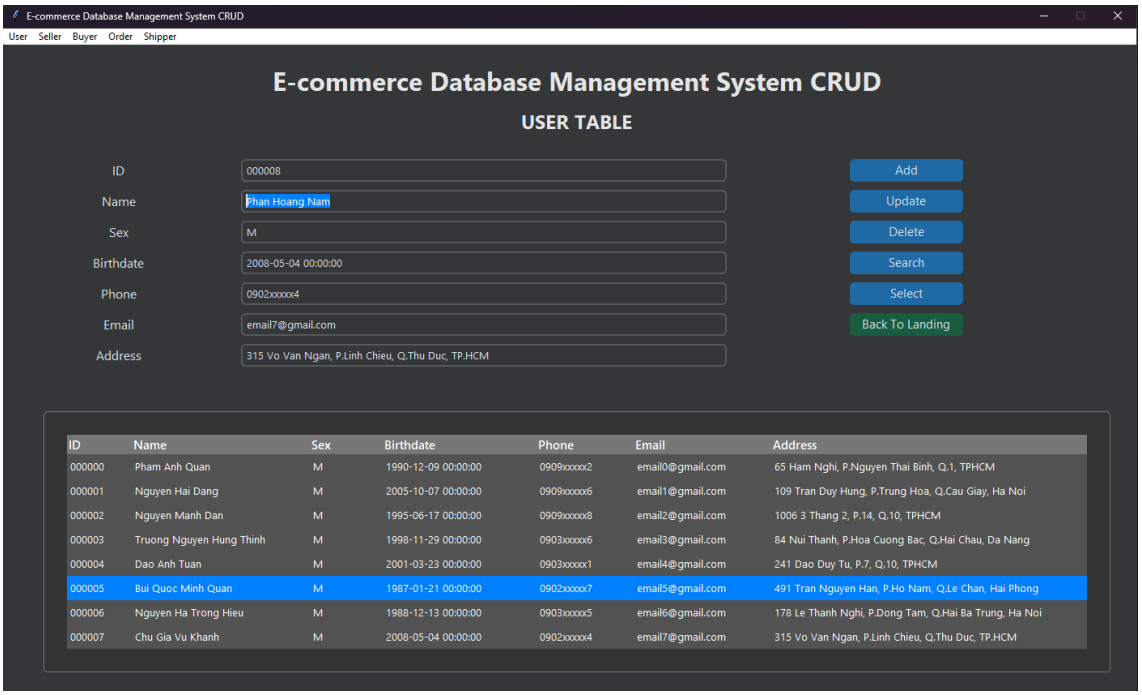
View table before logging in

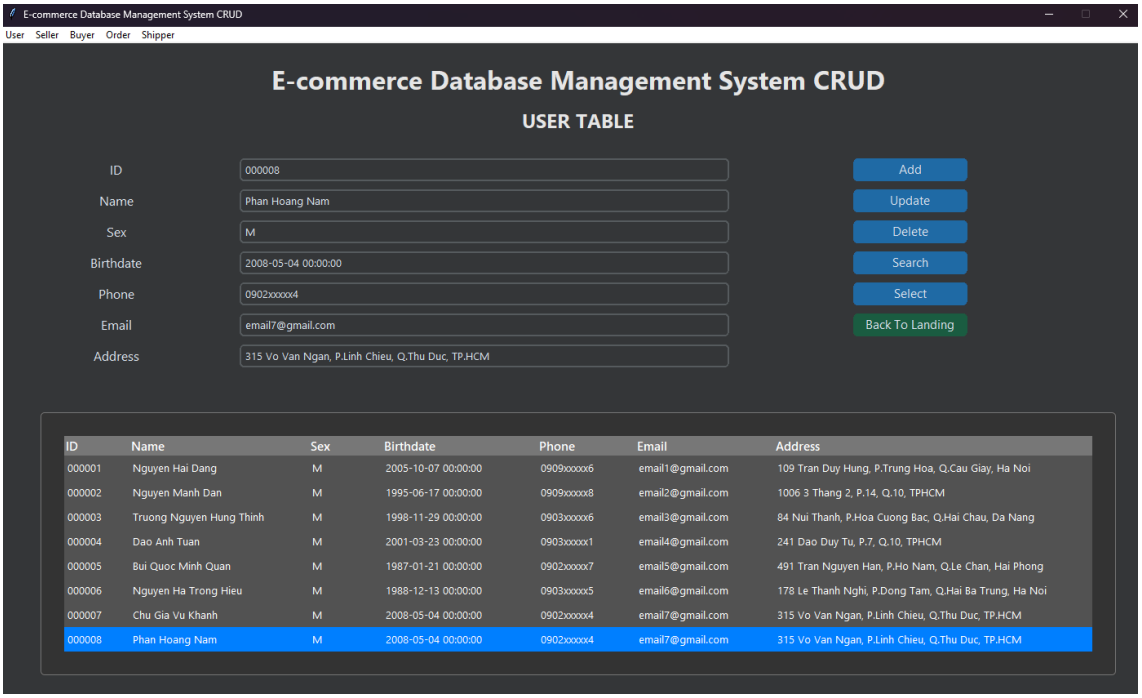


Logged in as a Buyer

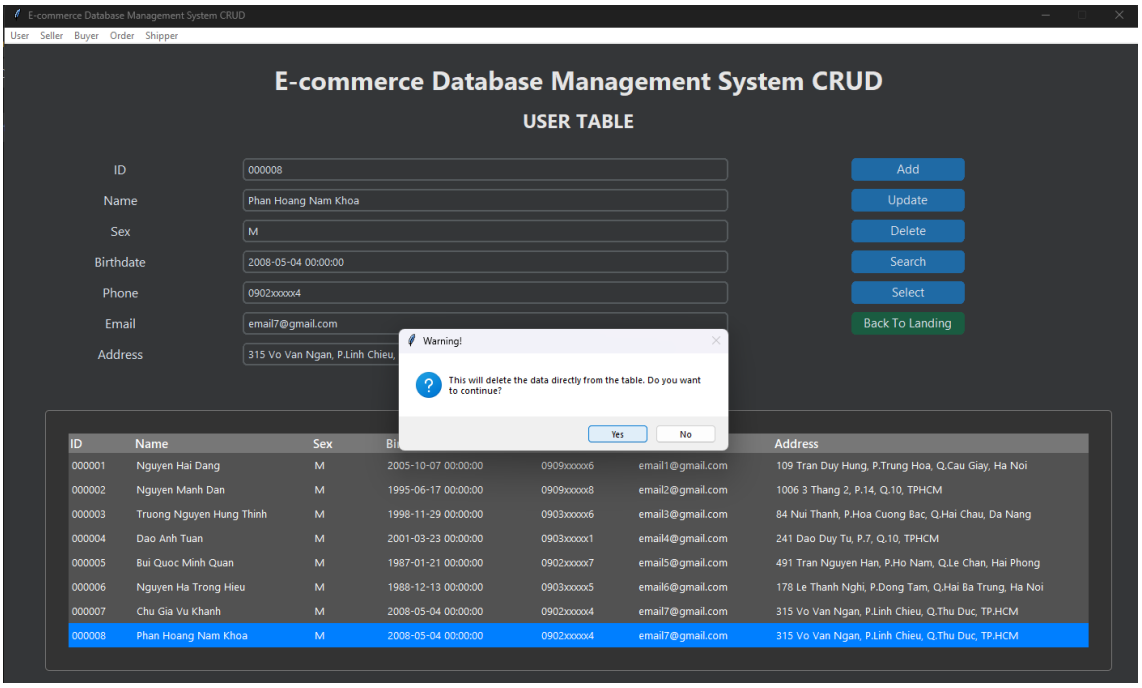


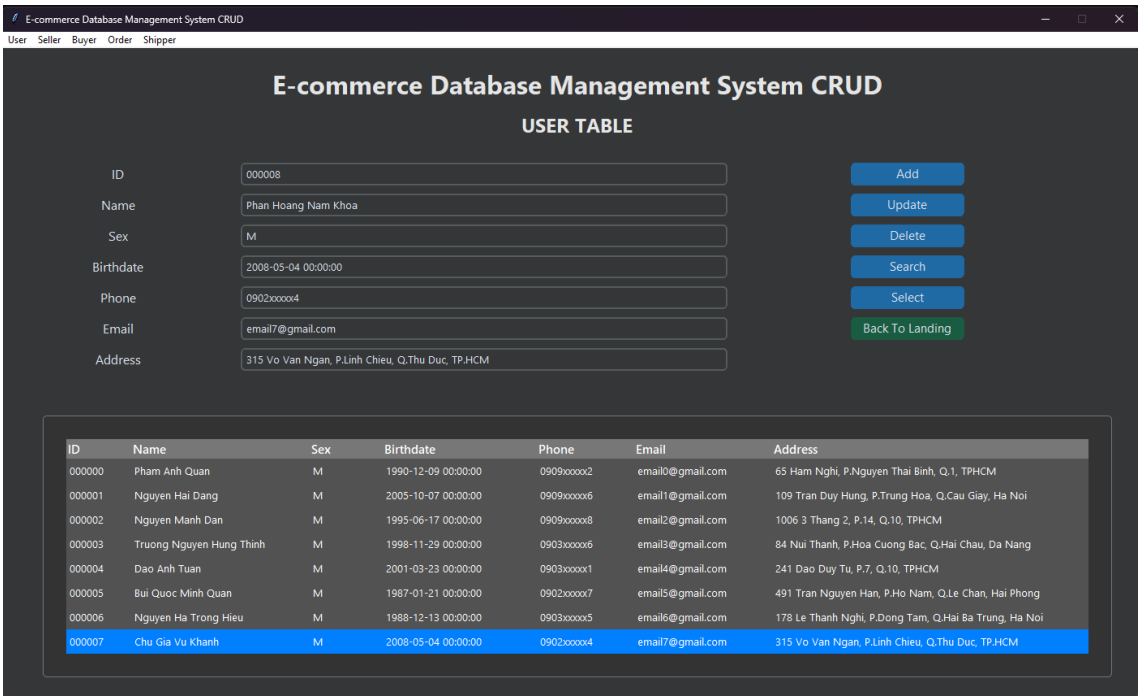
View Seller tables as a Buyer


E-commerce Database Management System CRUD
User Seller Buyer Order Shipper
E-commerce Database Management System CRUD
USER TABLE
ID 000008
Name Phan Hoang Nam
Sex M
Birthdate 2008-05-04 00:00:00
Phone 0902xxxxx4
Email email7@gmail.com
Address 315 Vo Van Ngan, P.Linh Chieu, Q.Thu Duc, TP.HCM
Add
Update
Delete
Search
Select
Back To Landing
ID Name Sex Birthdate Phone Email Address
000000 Pham Anh Quan M 1990-12-09 00:00:00 0909xxxxx2 email0@gmail.com 65 Ham Nghi, P.Nguyen Thai Binh, Q.1, TPHCM
000001 Nguyen Hai Dang M 2005-10-07 00:00:00 0909xxxxx6 email1@gmail.com 109 Tran Duy Hung, P.Trung Hoa, Q.Cau Giay, Ha Noi
000002 Nguyen Manh Dan M 1995-06-17 00:00:00 0909xxxxx8 email2@gmail.com 1006 3 Thang 2, P.14, Q.10, TPHCM
000003 Truong Nguyen Hung Thinh M 1998-11-29 00:00:00 0903xxxxx6 email3@gmail.com 84 Nui Thanh, P.Hoa Cuong Bac, Q.Hai Chau, Da Nang
000004 Dao Anh Tuan M 2001-03-23 00:00:00 0903xxxxx1 email4@gmail.com 241 Dao Duy Tu, P.7, Q.10, TPHCM
000005 Bui Quoc Minh Quan M 1987-01-21 00:00:00 0902xxxxx7 email5@gmail.com 491 Tran Nguyen Han, P.Ho Nam, Q.Le Chan, Hai Phong
000006 Nguyen Ha Trong Hieu M 1988-12-13 00:00:00 0903xxxxx5 email6@gmail.com 178 Le Thanh Nghi, P.Dong Tam, Q.Hai Ba Trung, Ha Noi
000007 Chu Gia Vu Khanh M 2008-05-04 00:00:00 0902xxxxx4 email7@gmail.com 315 Vo Van Ngan, P.Linh Chieu, Q.Thu Duc, TP.HCM

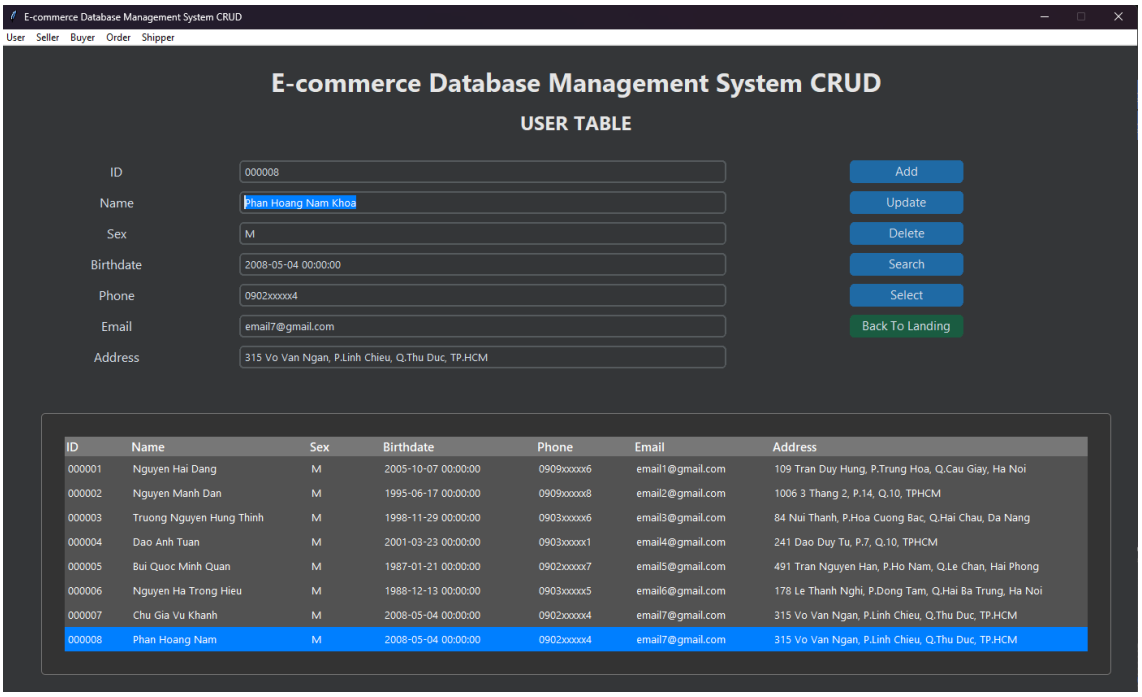

E-commerce Database Management System CRUD
User Seller Buyer Order Shipper
E-commerce Database Management System CRUD
USER TABLE
ID 000008
Name Phan Hoang Nam
Sex M
Birthdate 2008-05-04 00:00:00
Phone 0902xxxxx4
Email email7@gmail.com
Address 315 Vo Van Ngan, P.Linh Chieu, Q.Thu Duc, TP.HCM
Add
Update
Delete
Search
Select
Back To Landing
ID Name Sex Birthdate Phone Email Address
000001 Nguyen Hai Dang M 2005-10-07 00:00:00 0909xxxxx6 email1@gmail.com 109 Tran Duy Hung, P.Trung Hoa, Q.Cau Giay, Ha Noi
000002 Nguyen Manh Dan M 1995-06-17 00:00:00 0909xxxxx8 email2@gmail.com 1006 3 Thang 2, P.14, Q.10, TPHCM
000003 Truong Nguyen Hung Thinh M 1998-11-29 00:00:00 0903xxxxx6 email3@gmail.com 84 Nui Thanh, P.Hoa Cuong Bac, Q.Hai Chau, Da Nang
000004 Dao Anh Tuan M 2001-03-23 00:00:00 0903xxxxx1 email4@gmail.com 241 Dao Duy Tu, P.7, Q.10, TPHCM
000005 Bui Quoc Minh Quan M 1987-01-21 00:00:00 0902xxxxx7 email5@gmail.com 491 Tran Nguyen Han, P.Ho Nam, Q.Le Chan, Hai Phong
000006 Nguyen Ha Trong Hieu M 1988-12-13 00:00:00 0903xxxxx5 email6@gmail.com 178 Le Thanh Nghi, P.Dong Tam, Q.Hai Ba Trung, Ha Noi
000007 Chu Gia Vu Khanh M 2008-05-04 00:00:00 0902xxxxx4 email7@gmail.com 315 Vo Van Ngan, P.Linh Chieu, Q.Thu Duc, TP.HCM
000008 Phan Hoang Nam M 2008-05-04 00:00:00 0902xxxxx4 email7@gmail.com 315 Vo Van Ngan, P.Linh Chieu, Q.Thu Duc, TP.HCM

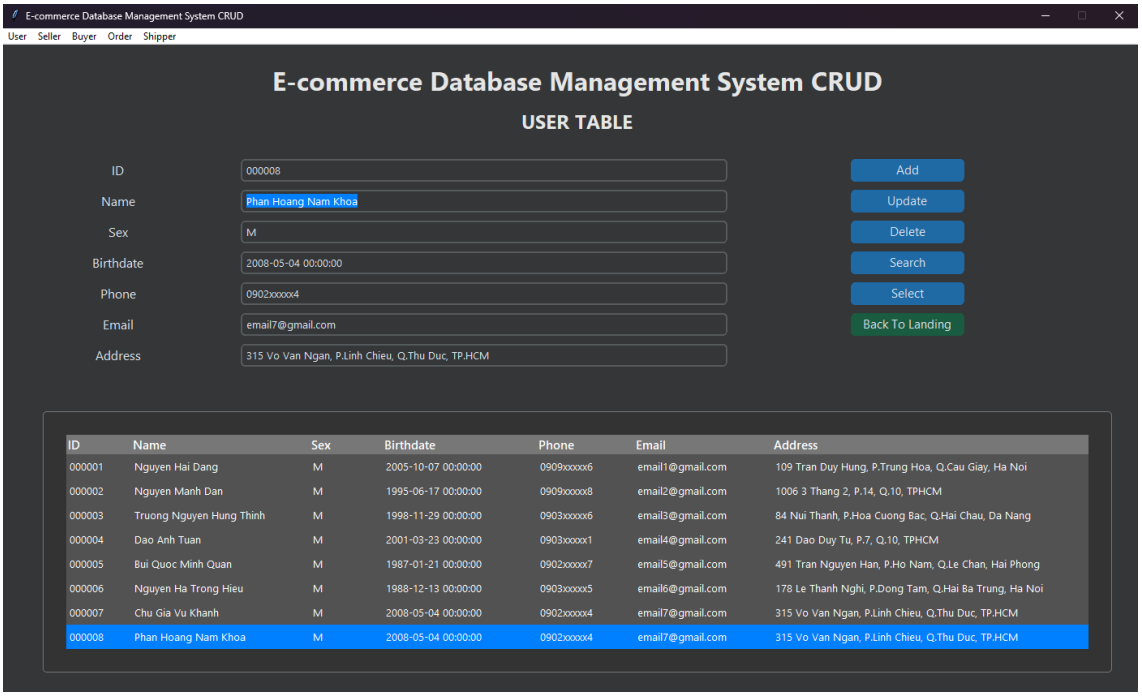
Add a user



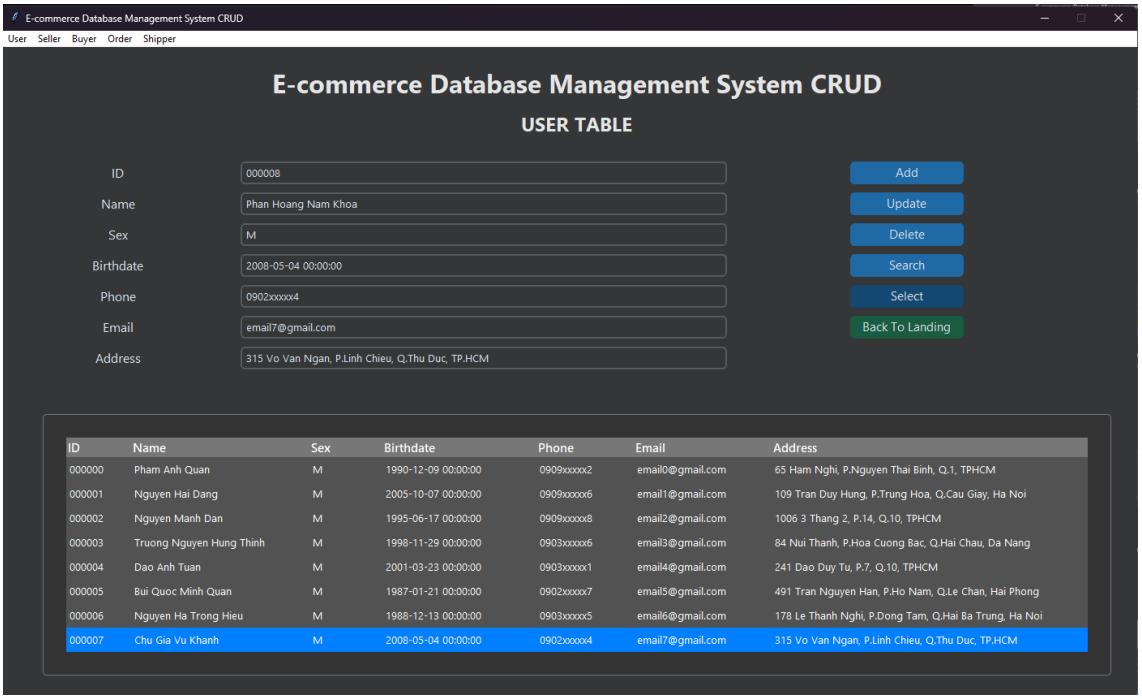


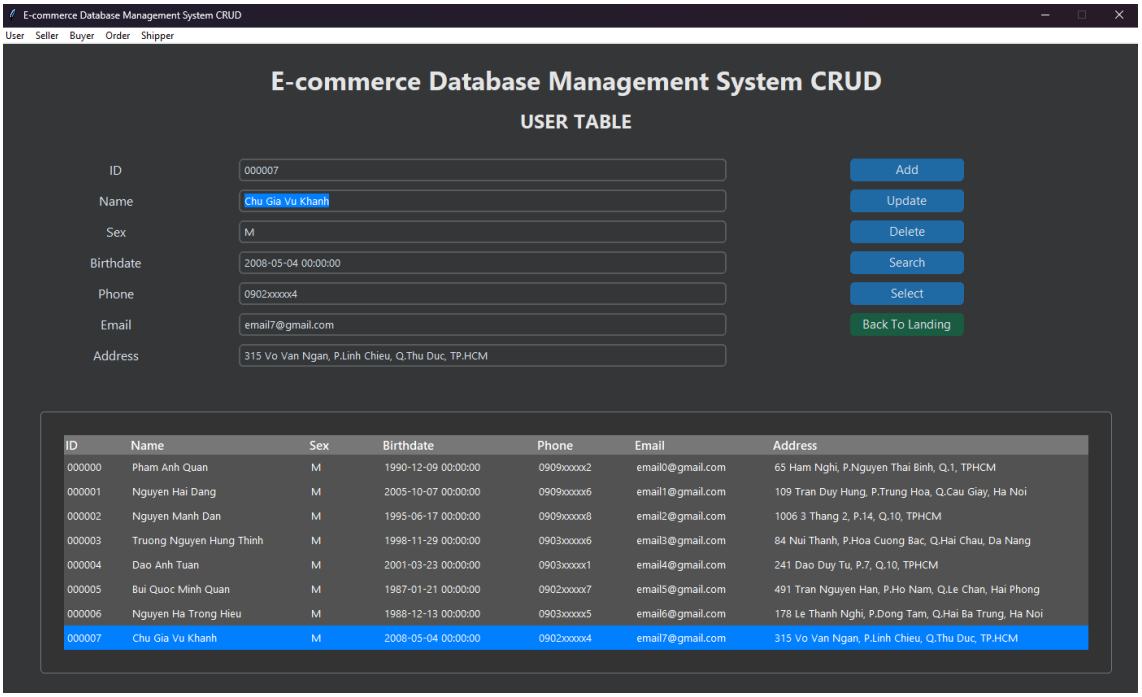
Delete a new user





Update a user's data





Select a user