

JAVA SWINGS BASED- DRINKSBUCKS STORE - SQL CONNECTIVITY USING JDBC

A

Report

*Submitted in partial fulfillment of the
Requirements for the award of the Degree of*

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

By

Chemudu Sreeya <1602-19-737-110 >

Under the guidance of Ms B. Leelavathy



Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

BONAFIDE CERTIFICATE

This is to certify that this project report titled '**DRINKSBUCKS STORE**' is a project work of Ms. Chemudu Sreeya bearing roll no. 1602-19-737-110 who carried out the project under my supervision in the IV semester for the academic year 2020- 2021.

Signature

Internal Examiner

Signature

External Examiner

ABSTRACT

Drinksbucks is a store which is designed to ease customers life. sometimes you feel like having some liquids or doesn't feel like to make your own drink or coffee or juice. So, drinksbucks is here to help you. In this drinksbucks, customers need to select their drink from the menu and then make an order. employees in the store will provide the order which the customer have ordered and make the bill payments accordingly. Drinks can be made within the store or raw materials needed can be supplied by a supplier. Here, every person involving has their separate id's and information which is stored separately in the database named drinksbucks.

INTRODUCTION

REQUIREMENTS

<u>Table name</u>	<u>Attributes</u>
Employee	employee_id number empname varchar2(25) age number contact number
Customer	customer_id number customer_name varchar2(25) gender varchar2(10) contactno number
Supplier	supplier_id number supplier_name varchar2(20) contactno number mailaddress varchar2(20)
Drinks	drinks_id number drinks_name varchar2(25) quantity number price number
Torder	order_id number customer_id number drinks_id number qty number price_perqty number
Payments	payment_id number order_id number customer_id number ttlquantity number ttlprice number

Transaction	transaction_id number customer_id number employee_id number order_id number drinks_id number payment_id number
-------------	---

AIM AND PRIORITY OF THE PROJECT

To create a Java GUI based drinksbucks store which takes the values like: customer id, drinks id etc. from the customer who orders their choice from the given drinks, it also stores information about the employees, suppliers, payments and orders. These values are to be updated in the database using JDBC connectivity .

ARCHITECTURE AND TECHNOLOGY

Software used: Java Eclipse, Oracle 11g Database, Java SE version 13, SQL*Plus.

Java AWT:

Java AWT

SWING is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) –an API for providing a graphical user interface (GUI) for Java programs.

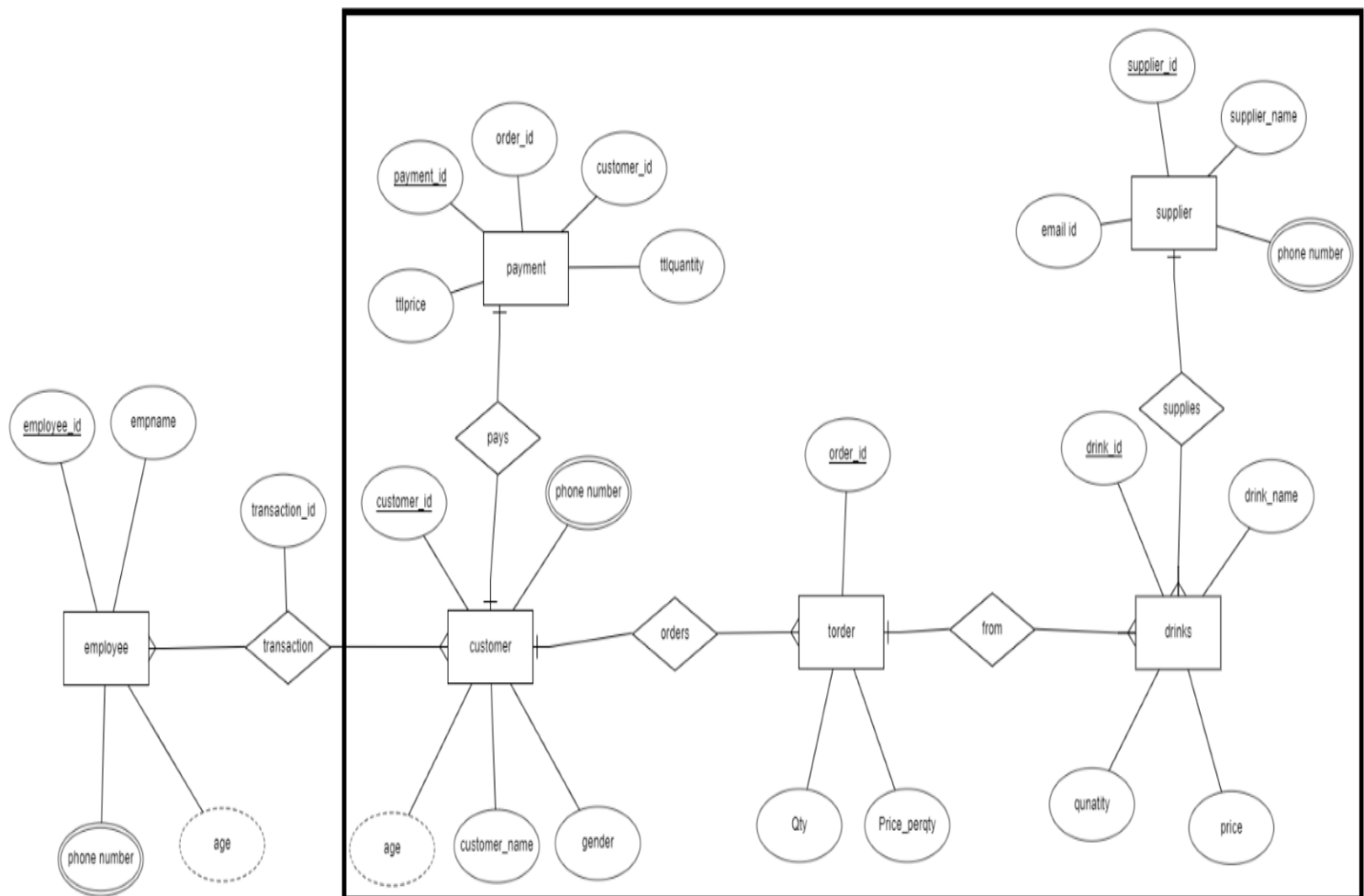
Swing was developed to provide a more sophisticated set of GUI components than the earlier AWT. Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

SQL:

Structure Query Language (SQL) is a database query language used for storing and managing data in Relational DBMS. SQL was the first commercial language introduced for E.F Codd's **Relational** model of database. Today almost all RDBMS (MySQL, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language. SQL is used to perform all types of data operations in RDBMS.

DESIGN

Entity relationship diagram



Database Design:

DDL Operations

Employee table:

```
SQL> create table employee(  
  2  employee_id number,  
  3  empname varchar2(25) NOT NULL,  
  4  age number NOT NULL,  
  5  contact number NOT NULL);  
  
Table created.
```

```
SQL> alter table employee add constraint p_employee_id primary key(employee_id);  
  
Table altered.
```

```
SQL> desc employee;  
+-----+-----+-----+  
Name                               Null?   Type  
+-----+-----+-----+  
EMPLOYEE_ID                       NOT NULL NUMBER  
EMPNAME                           NOT NULL VARCHAR2(25)  
AGE                               NOT NULL NUMBER  
CONTACT                           NOT NULL NUMBER  
  
SQL>
```

Customer table:

```
SQL> create table customer(  
  2  customer_id number,  
  3  customer_name varchar2(25),  
  4  gender varchar2(25),  
  5  contactno number NOT NULL);  
  
Table created.
```

```
SQL> alter table customer add constraint p_customer_id primary key(customer_id);  
Table altered.
```

```
SQL> desc customer;
```

Name	Null?	Type
CUSTOMER_ID	NOT NULL	NUMBER
CUSTOMER_NAME		VARCHAR2(25)
GENDER		VARCHAR2(25)
CONTACTNO	NOT NULL	NUMBER

Supplier table:

```
SQL> create table supplier(  
  2  supplier_id number,  
  3  supplier_name varchar2(25),  
  4  contact_no number NOT NULL,  
  5  mailaddress varchar2(20));
```

```
Table created.
```

```
Table created.
```

```
SQL> alter table supplier add constraint p_supplier_id primary key(supplier_id);
```

```
Table altered.
```

```
SQL> desc supplier;
```

Name	Null?	Type
SUPPLIER_ID	NOT NULL	NUMBER
SUPPLIER_NAME		VARCHAR2(25)
CONTACT_NO	NOT NULL	NUMBER
MAILADDRESS		VARCHAR2(20)

```
SQL>
```

Drinks table:

```
SQL> create table drinks(  
  2  drinks_id number,  
  3  drinks_name varchar2(25),  
  4  quantity number,  
  5  price number NOT NULL);
```

Table created.

```
SQL> alter table drinks add constraint p_drinks_id primary key(drinks_id);
```

Table altered.

```
SQL> desc drinks
```

Name	Null?	Type
DRINKS_ID	NOT NULL	NUMBER
DRINKS_NAME		VARCHAR2(25)
QUANTITY		NUMBER
PRICE	NOT NULL	NUMBER

```
SQL>
```

Order table:

```
SQL> create table torder(  
  2  order_id number,  
  3  customer_id number,  
  4  drinks_id number,  
  5  qty number,  
  6  price_perqty number);
```

Table created.

```
SQL> alter table torder add constraint p_order_id primary key(order_id);
Table altered.

SQL> alter table torder add foreign key(customer_id) references customer;
Table altered.

SQL> alter table torder add foreign key(drinks_id) references drinks;
Table altered.
```

```
SQL> desc torder;
Name                                         Null?    Type
-----
ORDER_ID                                    NOT NULL NUMBER
CUSTOMER_ID                                NUMBER
DRINKS_ID                                  NUMBER
QTY                                         NUMBER
PRICE_PERQTY                              NUMBER

SQL>
```

Payment table:

```
SQL> create table payments(
  2  payment_id number,
  3  order_id number,
  4  customer_id number,
  5  ttlquantity number,
  6  ttlprice number);
Table created.
```

```
SQL> alter table payments add constraint p_payment_id primary key(payment_id);
Table altered.

SQL> alter table payments add foreign key(order_id) references torder;
Table altered.

SQL> alter table payments add foreign key(customer_id) references customer;
Table altered.
```

```
SQL> desc payments;
Name                               Null?    Type
-----
PAYMENT_ID                         NOT NULL NUMBER
ORDER_ID                           NUMBER
CUSTOMER_ID                        NUMBER
TTLQUANTITY                        NUMBER
TTLPRICE                           NUMBER

SQL>
```

Transaction table:

```
SQL> create table transaction(
  2 transaction_id number,
  3 customer_id number,
  4 employee_id number,
  5 order_id number,
  6 drinks_id number,
  7 payment_id number);
Table created.

SQL> alter table transaction add(supplier_id number);
Table altered.
```

```
SQL> alter table transaction add constraint p_transaction_id primary key(transaction_id);
Table altered.
```

```
SQL> alter table transaction add foreign key(employee_id) references employee;
Table altered.
```

```
SQL> alter table transaction add foreign key(customer_id) references customer;
Table altered.
```

```
SQL> alter table transaction add foreign key(order_id) references torder;
Table altered.
```

```
SQL> alter table transaction add foreign key(drinks_id) references drinks;
Table altered.
```

```
SQL> alter table transaction add foreign key(payment_id) references payments;
Table altered.
```

```
SQL> alter table transaction add foreign key(supplier_id) references supplier;
Table altered.
```

```
SQL>
```

DML Operations

Employee table:

```
SQL> insert into employee values(&employee_id,&empname,&age,&contact);
Enter value for employee_id: 73701
Enter value for empname: garg
Enter value for age: 35
Enter value for contact: 89765432
old 1: insert into employee values(&employee_id,&empname,&age,&contact)
new 1: insert into employee values(73701,'garg',35,89765432)

1 row created.
```

```
SQL> insert into employee values(&employee_id,&empname,&age,&contact);
Enter value for employee_id: 73703
Enter value for empname: bhuvneshwar
Enter value for age: 31
Enter value for contact: 56789053
old 1: insert into employee values(&employee_id,&empname,&age,&contact)
new 1: insert into employee values(73703,'bhuvneshwar',31,56789053)

1 row created.
```

```
SQL> insert into employee values(&employee_id,&empname,&age,&contact);
Enter value for employee_id: 73704
Enter value for empname: warner
Enter value for age: 56
Enter value for contact: 12345678
old 1: insert into employee values(&employee_id,&empname,&age,&contact)
new 1: insert into employee values(73704,'warner',56,12345678)

1 row created.
```

```
SQL> insert into employee values(&employee_id,&empname,&age,&contact);
Enter value for employee_id: 73705
Enter value for empname: kane
Enter value for age: 28
Enter value for contact: 56431096
old 1: insert into employee values(&employee_id,&empname,&age,&contact)
new 1: insert into employee values(73705,'kane',28,56431096)

1 row created.
```

```
SQL> select * from employee;
```

EMPLOYEE_ID	EMPNAME	AGE	CONTACT
73701	garg	35	89765432
73702	priyam	42	89907654
73703	bhuvneshwar	31	56789053
73704	warner	56	12345678
73705	kane	28	56431096

Customer table:

```
SQL> insert into customer values(&customer_id,&customer_name,&gender,&contactno);
```

```
Enter value for customer_id: 1901
```

```
Enter value for customer_name: shub
```

```
Enter value for gender: male
```

```
Enter value for contactno: 89780098
```

```
old 1: insert into customer values(&customer_id,&customer_name,&gender,&contactno)
```

```
new 1: insert into customer values(1901,'shub','male',89780098)
```

```
1 row created.
```

```
SQL> insert into customer values(&customer_id,&customer_name,&gender,&contactno);
```

```
Enter value for customer_id: 1902
```

```
Enter value for customer_name: goyal
```

```
Enter value for gender: female
```

```
Enter value for contactno: 123908
```

```
old 1: insert into customer values(&customer_id,&customer_name,&gender,&contactno)
```

```
new 1: insert into customer values(1902,'goyal','female',123908)
```

```
1 row created.
```

```
SQL> insert into customer values(&customer_id,&customer_name,&gender,&contactno);
```

```
Enter value for customer_id: 1903
```

```
Enter value for customer_name: priya
```

```
Enter value for gender: female
```

```
Enter value for contactno: 675489
```

```
old 1: insert into customer values(&customer_id,&customer_name,&gender,&contactno)
```

```
new 1: insert into customer values(1903,'priya','female',675489)
```

```
1 row created.
```



```

1 row created.

SQL> insert into customer values(&customer_id,&customer_name,&gender,&contactno);
Enter value for customer_id: 1904
Enter value for customer_name: devilliars
Enter value for gender: male
Enter value for contactno: 908765
old 1: insert into customer values(&customer_id,&customer_name,&gender,&contactno)
new 1: insert into customer values(1904,'devilliars','male',908765)

1 row created.

SQL> insert into customer values(&customer_id,&customer_name,&gender,&contactno);
Enter value for customer_id: 1905
Enter value for customer_name: sita
Enter value for gender: female
Enter value for contactno: 456789
old 1: insert into customer values(&customer_id,&customer_name,&gender,&contactno)
new 1: insert into customer values(1905,'sita','female',456789)

1 row created.

```

```

SQL> select * from customer;

```

CUSTOMER_ID	CUSTOMER_NAME	GENDER	CONTACTNO
1901	shub	male	89780098
1902	goyal	female	123908
1903	priya	female	675489
1904	devilliars	male	908765
1905	sita	female	456789

Supplier table:

```

SQL> insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress');
Enter value for supplier_id: 160201
Enter value for supplier_name: raghu
Enter value for contact_no: 908765
Enter value for mailaddress: raghu@gmail.com
old 1: insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress')
new 1: insert into supplier values(160201,'raghu',908765,'raghu@gmail.com')

1 row created.

```

```
SQL> insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress');
Enter value for supplier_id: 160202
Enter value for supplier_name: ramu
Enter value for contact_no: 8911234
Enter value for mailaddress: ramu@gmail.com
old 1: insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress')
new 1: insert into supplier values(160202,'ramu',8911234,'ramu@gmail.com')

1 row created.
```

```
SQL> insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress');
Enter value for supplier_id: 160203
Enter value for supplier_name: janaki
Enter value for contact_no: 908744
Enter value for mailaddress: janaki@gmail.com
old 1: insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress')
new 1: insert into supplier values(160203,'janaki',908744,'janaki@gmail.com')

1 row created.
```

```
SQL> insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress');
Enter value for supplier_id: 160204
Enter value for supplier_name: sonakshibose
Enter value for contact_no: 897666
Enter value for mailaddress: sonakshi@gmail.com
old 1: insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress')
new 1: insert into supplier values(160204,'sonakshibose',897666,'sonakshi@gmail.com')

1 row created.
```

```
SQL> insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress');
Enter value for supplier_id: 160205
Enter value for supplier_name: devdixit
Enter value for contact_no: 667788
Enter value for mailaddress: devdixit@gmail.com
old 1: insert into supplier values(&supplier_id,&supplier_name,&contact_no,&mailaddress')
new 1: insert into supplier values(160205,'devdixit',667788,'devdixit@gmail.com')

1 row created.
```

```
SQL> select * from supplier;
```

SUPPLIER_ID	SUPPLIER_NAME	CONTACT_NO	MAILADDRESS
160201	raghu	908765	raghu@gmail.com
160202	ramu	8911234	ramu@gmail.com
160203	janaki	908744	janaki@gmail.com
160204	sonakshibose	897666	sonakshi@gmail.com
160205	devdixit	667788	devdixit@gmail.com

INFORMATION RETRIEVAL OF GOOGLE QUERIES OF POSITIVE AND NEGATIVE FEEDBACK

Drinks table:

```
SQL> insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price);
Enter value for drinks_id: 01
Enter value for drinks_name: capuccino
Enter value for quantity: 20
Enter value for price: 500
old 1: insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price)
new 1: insert into drinks values(01,'capuccino',20,500)

1 row created.
```

```
SQL> insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price);
Enter value for drinks_id: 02
Enter value for drinks_name: mangojuice
Enter value for quantity: 80
Enter value for price: 1000
old 1: insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price)
new 1: insert into drinks values(02,'mangojuice',80,1000)

1 row created.
```

```
SQL> insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price);
Enter value for drinks_id: 03
Enter value for drinks_name: chocolava milkshake
Enter value for quantity: 50
Enter value for price: 2000
old 1: insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price)
new 1: insert into drinks values(03,'chocolava milkshake',50,2000)

1 row created.
```

```
SQL> insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price);
Enter value for drinks_id: 04
Enter value for drinks_name: dietcoke
Enter value for quantity: 40
Enter value for price: 5000
old 1: insert into drinks values(&drinks_id,&'&drinks_name',&quantity,&price)
new 1: insert into drinks values(04,'dietcoke',40,5000)

1 row created.
```

```
SQL> insert into drinks values(&drinks_id,&drinks_name,&quantity,&price);
Enter value for drinks_id: 05
Enter value for drinks_name: sodalemon
Enter value for quantity: 60
Enter value for price: 200
old   1: insert into drinks values(&drinks_id,&drinks_name,&quantity,&price)
new   1: insert into drinks values(05,'sodalemon',60,200)

1 row created.
```

```
SQL> select * from drinks;
```

DRINKS_ID	DRINKS_NAME	QUANTITY	PRICE
1	capuccino	20	500
2	mangojuice	80	1000
3	chocolava milkshake	50	2000
4	dietcoke	40	5000
5	sodalemon	60	200

Order table:

```
SQL> insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty);
Enter value for order_id: 001
Enter value for customer_id: 1901
Enter value for drinks_id: 01
Enter value for qty: 2
Enter value for price_perqty: 50
old   1: insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty)
new   1: insert into torder values(001,1901,01,2,50)

1 row created.
```

```
SQL> insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty);
Enter value for order_id: 002
Enter value for customer_id: 1902
Enter value for drinks_id: 02
Enter value for qty: 3
Enter value for price_perqty: 800
old   1: insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty)
new   1: insert into torder values(002,1902,02,3,800)

1 row created.
```

```
SQL> insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty);
Enter value for order_id: 003
Enter value for customer_id: 1903
Enter value for drinks_id: 03
Enter value for qty: 1
Enter value for price_perqty: 40
old 1: insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty)
new 1: insert into torder values(003,1903,03,1,40)

1 row created.
```

```
SQL> insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty);
Enter value for order_id: 004
Enter value for customer_id: 1904
Enter value for drinks_id: 04
Enter value for qty: 5
Enter value for price_perqty: 625
old 1: insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty)
new 1: insert into torder values(004,1904,04,5,625)

1 row created.
```

```
SQL> insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty);
Enter value for order_id: 005
Enter value for customer_id: 1905
Enter value for drinks_id: 05
Enter value for qty: 2
Enter value for price_perqty: 40
old 1: insert into torder values(&order_id,&customer_id,&drinks_id,&qty,&price_perqty)
new 1: insert into torder values(005,1905,05,2,40)

1 row created.
```

```
SQL> select * from torder;
```

ORDER_ID	CUSTOMER_ID	DRINKS_ID	QTY	PRICE_PERQTY
1	1901	1	2	50
2	1902	2	3	800
3	1903	3	1	40
4	1904	4	5	625
5	1905	5	2	40

Payment table:

```
SQL> insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price);
Enter value for payment_id: 501
Enter value for order_id: 001
Enter value for customer_id: 1901
Enter value for t1_quantity: 2
Enter value for t1_price: 50
old 1: insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price)
new 1: insert into payments values(501,001,1901,2,50)

1 row created.
```

```
SQL> insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price);
Enter value for payment_id: 502
Enter value for order_id: 002
Enter value for customer_id: 1902
Enter value for t1_quantity: 3
Enter value for t1_price: 800
old 1: insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price)
new 1: insert into payments values(502,002,1902,3,800)

1 row created.
```

```
SQL> insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price);
Enter value for payment_id: 503
Enter value for order_id: 003
Enter value for customer_id: 1903
Enter value for t1_quantity: 1
Enter value for t1_price: 40
old 1: insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price)
new 1: insert into payments values(503,003,1903,1,40)

1 row created.
```

```
SQL> insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price);
Enter value for payment_id: 504
Enter value for order_id: 004
Enter value for customer_id: 1904
Enter value for t1_quantity: 5
Enter value for t1_price: 625
old 1: insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price)
new 1: insert into payments values(504,004,1904,5,625)

1 row created.
```

```
SQL> insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price);
Enter value for payment_id: 505
Enter value for order_id: 005
Enter value for customer_id: 1905
Enter value for t1_quantity: 2
Enter value for t1_price: 40
old 1: insert into payments values(&payment_id,&order_id,&customer_id,&t1_quantity,&t1_price)
new 1: insert into payments values(505,005,1905,2,40)

1 row created.
```

```
SQL> select * from payments;
```

PAYMENT_ID	ORDER_ID	CUSTOMER_ID	TTLQUANTITY	TTLPRICE
501	1	1901	2	50
502	2	1902	3	800
503	3	1903	1	40
504	4	1904	5	625
505	5	1905	2	40

Transaction table:

```
SQL> insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id);
Enter value for transaction_id: 202101
Enter value for customer_id: 1901
Enter value for employee_id: 73701
Enter value for order_id: 001
Enter value for drinks_id: 01
Enter value for payment_id: 501
Enter value for supplier_id: 160201
old 1: insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id)
new 1: insert into transaction values(202101,1901,73701,001,01,501,160201)

1 row created.
```

```
SQL> insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id);
Enter value for transaction_id: 202102
Enter value for customer_id: 1902
Enter value for employee_id: 73702
Enter value for order_id: 002
Enter value for drinks_id: 02
Enter value for payment_id: 502
Enter value for supplier_id: 160202
old 1: insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id)
new 1: insert into transaction values(202102,1902,73702,002,02,502,160202)

1 row created.
```

```
SQL> insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id);
Enter value for transaction_id: 202103
Enter value for customer_id: 1903
Enter value for employee_id: 73703
Enter value for order_id: 003
Enter value for drinks_id: 03
Enter value for payment_id: 503
Enter value for supplier_id: 160203
old 1: insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id)
new 1: insert into transaction values(202103,1903,73703,003,03,503,160203)

1 row created.
```



```
SQL> insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id);
Enter value for transaction_id: 202104
Enter value for customer_id: 1904
Enter value for employee_id: 73704
Enter value for order_id: 004
Enter value for drinks_id: 04
Enter value for payment_id: 504
Enter value for supplier_id: 160204
old 1: insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id)
new 1: insert into transaction values(202104,1904,73704,004,04,504,160204)
```

1 row created.

```
SQL> insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id);
Enter value for transaction_id: 202105
Enter value for customer_id: 1905
Enter value for employee_id: 73705
Enter value for order_id: 005
Enter value for drinks_id: 05
Enter value for payment_id: 505
Enter value for supplier_id: 160205
old 1: insert into transaction values(&transaction_id,&customer_id,&employee_id,&order_id,&drinks_id,&payment_id,&supplier_id)
new 1: insert into transaction values(202105,1905,73705,005,05,505,160205)
```

1 row created.

```
SQL> select * from transaction;
```

TRANSACTION_ID	CUSTOMER_ID	EMPLOYEE_ID	ORDER_ID	DRINKS_ID	PAYMENT_ID	SUPPLIER_ID
----------------	-------------	-------------	----------	-----------	------------	-------------

202101	1901	73701	1	1	501	160201
--------	------	-------	---	---	-----	--------

202102	1902	73702	2	2	502	160202
--------	------	-------	---	---	-----	--------

202103	1903	73703	3	3	503	160203
--------	------	-------	---	---	-----	--------

TRANSACTION_ID	CUSTOMER_ID	EMPLOYEE_ID	ORDER_ID	DRINKS_ID	PAYMENT_ID	SUPPLIER_ID
----------------	-------------	-------------	----------	-----------	------------	-------------

202104	1904	73704	4	4	504	160204
--------	------	-------	---	---	-----	--------

202105	1905	73705	5	5	505	160205
--------	------	-------	---	---	-----	--------

SQL>

IMPLEMENTATION

Front end programs and its connectivity

Java Database Connectivity (JDBC) is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

The connection to the database can be performed using Java programming (JDBC API) as:

```
public void connectToDB()
{
    try
    {
        connection =
        DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");
        statement = connection.createStatement();
    }
    catch (SQLException connectException)
    {
        System.out.println(connectException.getMessage())
        ;
        System.out.println(connectException.getSQLState()
        );
        System.out.println(connectException.getErrorCode());
        System.exit(1);
    }
}
```

Thus, the connection from Java to Oracle database is performed and therefore, can be used for updating tables in the database directly.

PROGRAMS:

Employee:

```
package DrinkB;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.*;
import java.util.StringTokenizer;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class Emp{
/**
 *
 */
private static final long serialVersionUID = 1L;
private JButton insertButton,deleteButton,updateButton,viewButton;
private JPanel p1,p2,p3,p;
private JLabel lblemployee_id,blbempname,lblage,lblcontact;
private JTextField txtemployee_id,txtempname,txtage,txtcontact;

private List MIDList;
Connection con;ResultSet rs;
Statement statement;
private JFrame frame;
private JMenuItem insert,delete,update,view;
public Emp(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem update,JMenuItem view)
{

    try
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch (Exception e)
    {
        System.err.println("Unable to find and load driver");
    }
}
```

Drinksbucks store

```
                System.exit(1);
            }
            connectToDB();

            this.frame=frame;
            this.insert=insert;
            this.delete=delete;
            this.update=update;
            this.view=view;

            lblemployee_id=new JLabel("employee_id");
            lblername=new JLabel("empname");
            lblage=new JLabel("age");
            lblcontact=new JLabel("contact");

            txtemployee_id=new JTextField(15);
            txtername=new JTextField(15);
            txtage=new JTextField(8);
            txtcontact=new JTextField(15);

            this.p=p;
        }

        public void connectToDB()
        {
            try {

                Connection con=DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");

                statement=con.createStatement();
                statement.executeUpdate("commit");

            }
            catch (SQLException connectException)
            {
                System.out.println(connectException.getMessage());
                System.out.println(connectException.getSQLState());
                System.out.println(connectException.getErrorCode());
                System.exit(1);
            }
        }
        private void displaySQLExceptions(SQLException e)
        {
            JOptionPane.showMessageDialog(p,"\nSQLException: " + e.getMessage() + "\n"+"SQLState: " +
            e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");
        }
    }
}
```

```

}
public void loademployees() {
    try {
        MIDList.removeAll();
        rs=statement.executeQuery("select * from employee");
        while(rs.next()) {
            MIDList.add(rs.getString("employee_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void buildGUI() {

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub
            insertButton=new JButton("submit");
            txtemployee_id.setText(null);
            txttempname.setText(null);
            txtage.setText(null);
            txtcontact.setText(null);

            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();

            p1=new JPanel();

            p1.setLayout(new GridLayout(4,2));
            p1.add(lblemployee_id);
            p1.add(txtemployee_id);
            p1.add(lblempname);
            p1.add(txttempname);
            p1.add(lblage);
            p1.add(txtage);
            p1.add(lblcontact);
            p1.add(txtcontact);

            p3=new JPanel(new FlowLayout());
            p3.add(insertButton);
            //p1.add(txtf1);
            p3.setBackground(Color.yellow);
            p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
            p1.setBackground(Color.pink);

```

```
p2 = new JPanel(new FlowLayout());

        MIDList=new List(10);
        loademployees();
        p2.add(MIDList);p2.setBackground(Color.cyan) ;

        p2.setBounds(450,150,350,180);


p. add(p1);p.add(p3);
p. add(p2);


p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

insertButton.addActionListener(new ActionListener() {
    @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
try {
        String          query="INSERT          INTO          employee
VALUES("+txtemployee_id.getText()+"','"+txttempname.getText()+"','"+txttag
e.getText()+"','"+txtcontact.getText()+"");

        int i=statement.executeUpdate(query);
        JOptionPane.showMessageDialog(p,"\ninserted          "+i+"          rows
succesfully");loademployees();

    }
    catch(SQLException insertException){
        displaySQLErrors(insertException);
    }

}

        });
});

delete.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
```

```

// TODO Auto-generated method stub
deleteButton=new JButton("delete");

txtemployee_id.setText(null);
txtempname.setText(null);
txtage.setText(null);
txtcontact.setText(null);


p.removeAll();
frame.invalidate();
frame.validate();
frame.repaint();


p1=new JPanel();

p1.setLayout(new GridLayout(4,2));
p1.add(lblemployee_id);
p1.add(txtemployee_id);
p1.add(lblempname);
p1.add(txtempname);
p1.add(lblage);
p1.add(txtage);
p1.add(lblcontact);
p1.add(txtcontact);


p3=new JPanel(new FlowLayout());
p3.add(deleteButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;


// p1.setBounds(100,100,500,300);

p2 = new JPanel(new FlowLayout());

MIDList=new List(10);
loademployees();
p2.add(MIDList);p2.setBackground(Color.cyan) ;

p2.setBounds(450,150,350,180);


p. add(p1);p.add(p3);
p. add(p2);
MIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {

```

```

        rs=statement.executeQuery("select      *      from
employee");
StringTokenizer      st=new
StringTokenizer(MIDList.getSelectedItemAt(">"));
String p=st.nextToken();
while (rs.next())
{
    if
    (rs.getString("employee_id").equals(p)
    )
        break;
}
if (!rs.isAfterLast())
{
    txtemployee_id.setText(rs.getString("employee_id"));

    txtempname.setText(rs.getString("emp
name"));
    txtage.setText(rs.getString("age"));
    txtcontact.setText(rs.getString("contact
"));
}
}
catch (SQLException selectException)
{
    displaySQLErrors(selectException);
}
});

p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

deleteButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
try {

    int a=JOptionPane.showConfirmDialog(p,"Are you sure want to delete:");
    if(a==JOptionPane.YES_OPTION){
        StringTokenizer      st=new
        StringTokenizer(MIDList.getSelectedItemAt(">"));
        String      query="DELETE      FROM      employee      WHERE
employee_id="+st.nextToken();

        int i=statement.executeUpdate(query);
        JOptionPane.showMessageDialog(p,"\nDeleted      "+i+"      rows
succesfully");loademployees();
    }
}

```

```

    }
    catch(SQLException deleteException){
        displaySQLErrors(deleteException);
    }
}

});
}
});

```

```

update.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        JButton updateButton = new JButton("modify");
        txtemployee_id.setText(null);
        txttempname.setText(null);
        txtage.setText(null);
        txtcontact.setText(null);

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        p1=new JPanel();

        p1.setLayout(new GridLayout(4,2));
        p1.add(lblemployee_id);
        p1.add(txtemployee_id);
        p1.add(lblempname);
        p1.add(txttempname);
        p1.add(lblage);
        p1.add(txtage);
        p1.add(lblcontact);
        p1.add(txtcontact);
        p3=new JPanel(new FlowLayout());
        p3.add(updateButton);
        //p1.add(txtf1);
        p3.setBackground(Color.yellow);
        p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
        p1.setBackground(Color.pink) ;
        p2 = new JPanel(new FlowLayout());
        MIDList=new List(10);
        loademployees();
        p2.add(MIDList);p2.setBackground(Color.cyan) ;
        p2.setBounds(450,150,350,180);
        p. add(p1);p.add(p3);
        p. add(p2);
        MIDList.addItemListener(new ItemListener()
        {

```



```

        public void itemStateChanged(ItemEvent e)
        {
            try
            {
                rs=statement.executeQuery("select      *      from
                employee");
                while (rs.next())
                {
                    if
                    (rs.getString("employee_id").equals(M
                    IDList.getSelectedItem()))
                        break;
                }
                if (!rs.isAfterLast())
                {
                    txtemployee_id.setText(rs.getString("employee_id"));

                    txtempname.setText(rs.getString("emp
                    name"));
                    txtage.setText(rs.getString("age"));
                    txtcontact.setText(rs.getString("contact
                    "));
                }
            }
            catch (SQLException selectException)
            {
                displaySQLErrors(selectException);
            }
        }
    });
    p.setLayout(new BorderLayout());
    frame.add(p);
    frame.setSize(800,800);
    frame.validate();
    updateButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            try {
                loademployees();
                String age=JOptionPane.showInputDialog(p,"enter
                the age");

                txtage.setText(age);
                //int  a=JOptionPane.showConfirmDialog(p,"Are
                you sure want to update:");
                //if(a==JOptionPane.YES_OPTION){
                String  query="update      employee      set
                age='"+age+"'where
                employee_id="+txtemployee_id.getText();

                //      int      a=JOptionPane.showConfirmDialog(p,"Are
                you sure want to update:");
                //      if(a==JOptionPane.YES_OPTION){

```

```

        //String query="update employee set
        empname='"+txtempname.getText()+"',age="+txta
        ge.getText()+"",contact='"+txtcontact.getText()+"
        WHERE
        employee_id="+MIDList.getSelectedItem();

        int i=statement.executeUpdate(query);
        JOptionPane.showMessageDialog(p,"\nupdated
        "+i+" rows succesfully");loademployees();
    }
    catch(SQLException deleteException){
        displaySQLErrors(deleteException);
    }
}

});

}

});

view.addActionListener(new ActionListener(){

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        Label view1=new Label("employees view");
        //view1.setAlignment(Label.CENTER);
        Font myFont = new Font("Serif",Font.BOLD,50);
        view1.setFont((myFont));
        viewButton=new JButton("View");
        p1=new JPanel();
        p2=new JPanel();
        p1.add(view1);
        p2.add(viewButton);p1.setBackground(Color.cyan) ;p2.setBackground(Color.cyan) ;
        p.add(p1);p.add(p2); p.setLayout(new FlowLayout());
        frame.add(p);
        frame.setSize(800,800);
        frame.validate();
        viewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                JFrame f;

                JTable j;

                f = new JFrame();

                f.setTitle("Medicine details");

```

```
DefaultTableModel model = new DefaultTableModel();
j = new JTable(model);
model.addColumn("employee id");
model.addColumn("empname");
model.addColumn("age");
model.addColumn("contact");

try {

    rs=statement.executeQuery("select      *
    from employee");
    while(rs.next()) {
        model.addRow(new
        Object[]{rs.getString("employee_id"),
        rs.getString("empname"),rs.getString("
        age"),rs.getString("contact")});
    }
    catch(SQLException viewException) {
        displaySQLErrors(viewException);
    }
    j.setEnabled(false);
j.setBounds(30, 40, 300, 300);

JScrollPane sp = new JScrollPane(j);
f.add(sp);

f.setSize(800, 400);

f.setVisible(true);

}
```

```
});
```

```
}
```

```
});
```

```
}
```

```
}
```

Customer:

```
package DrinkB;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.*;
import java.util.StringTokenizer;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class Cust{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JButton insertButton,deleteButton,updateButton,viewButton;
    private JPanel p1,p2,p3,p;
    private JLabel lblcustomer_id,lbldcustomer_name,lbldgender,lbldcontactno;
    private JTextField txtcustomer_id,txtcustomer_name,txtgender,txtcontactno;
    private List OrderIDList;
    Connection con;ResultSet rs;
    Statement statement;
    private JFrame frame;
    private JMenuItem insert,delete,update,view;
    public Cust(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem update,JMenuItem
view)
    {

        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
            System.err.println("Unable to find and load driver");
            System.exit(1);
        }
    }
}
```

Drinksbucks store

```
connectToDB();

this.frame=frame;
this.insert=insert;
this.delete=delete;
this.update=update;
this.view=view;

lblcustomer_id=new JLabel("customer id");
lblcustomer_name=new JLabel("customer name");
lblgender=new JLabel("gender");
lblcontactno=new JLabel("contactno");

txtcustomer_id=new JTextField(15);
txtcustomer_name=new JTextField(15);
txtgender=new JTextField(8);
txtcontactno=new JTextField(8);
this.p=p;

}

public void connectToDB()
{
    try {

        Connection con=DriverManager.getConnection(
            "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");

        statement=con.createStatement();
        statement.executeUpdate("commit");

    }
    catch (SQLException connectException)
    {
        System.out.println(connectException.getMessage());
        System.out.println(connectException.getSQLState());
        System.out.println(connectException.getErrorCode());
        System.exit(1);
    }
}

private void displaySQLExceptions(SQLException e)
{
    JOptionPane.showMessageDialog(p,"\nSQLException: " + e.getMessage() + "\n"+"SQLState:
" + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

}

public void loadcustomers() {
```

```

try {
    OrderIDList.removeAll();
    rs=statement.executeQuery("select * from customer");
    while(rs.next()) {
        OrderIDList.add(rs.getString("customer_id"));
    }
}
catch(SQLException e) {
    displaySQLErrors(e);
}
}

```

```

public void buildGUI() {

```

```

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub
            insertButton=new JButton("submit");
            txtcustomer_id.setText(null);
            txtcustomer_name.setText(null);
            txtgender.setText(null);
            txtcontactno.setText(null);

            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();

            p1=new JPanel();

            p1.setLayout(new GridLayout(4,2));
            p1.add(lblcustomer_id);
            p1.add(txtcustomer_id);
            p1.add(lblcustomer_name);
            p1.add(txtcustomer_name);
            p1.add(lblgender);
            p1.add(txtgender);
            p1.add(lblcontactno);
            p1.add(txtcontactno);

            p3=new JPanel(new FlowLayout());
            p3.add(insertButton);
            //p1.add(txtf1);
            p3.setBackground(Color.yellow);
            p3.setBounds(200,350,75,35); p1.setBounds(115,80,300,250);
            p1.setBackground(Color.pink) ;

            p2 = new JPanel(new FlowLayout());

```

```
OrderIDList=new List(10);
loadcustomers();
p2.add(OrderIDList);p2.setBackground(Color.cyan) ;

        p2.setBounds(450,150,350,180);

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

insertButton.addActionListener(new ActionListener() {
    @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            try {
                String query="INSERT INTO customer
VALUES("+txtcustomer_id.getText()+",""+txtcustomer_name.getText()+",""+txtgender.getText()+",""+txtcontact
no.getText()+")";

                int i=statement.executeUpdate(query);
                JOptionPane.showMessageDialog(p,"\ninserted "+i+" rows
succesfully");

                loadcustomers();

            }
            catch(SQLException insertException){
                displaySQLErrors(insertException);
            }

        }

    });

delete.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        deleteButton=new JButton("delete");
```

```
txtcustomer_id.setText(null);
txtcustomer_name.setText(null);
txtgender.setText(null);
txtcontactno.setText(null);
```

```
p.removeAll();
frame.invalidate();
frame.validate();
frame.repaint();
```

```
p1=new JPanel();
```

```
p1.setLayout(new GridLayout(4,2));
```

```
p1.add(lblcustomer_id);
p1.add(txtcustomer_id);
p1.add(lblcustomer_name);
p1.add(txtcustomer_name);
p1.add(lblgender);
p1.add(txtgender);
p1.add(lblcontactno);
p1.add(txtcontactno);
```

```
p3=new JPanel(new FlowLayout());
p3.add(deleteButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p3.setBounds(200,350,75,35); p1.setBounds(115,80,300,250);
p1.setBackground(Color.pink);
```

```
// p1.setBounds(100,100,500,300);
```

```
p2 = new JPanel(new FlowLayout());
```

```
OrderIDList=new List(10);
loadcustomers();
p2.add(OrderIDList);p2.setBackground(Color.cyan);

p2.setBounds(450,150,350,180);
```

```
p. add(p1);p.add(p3);
p. add(p2);
```

```
OrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
```


Drinksbucks store

```

        {
            try
            {
                rs=statement.executeQuery("select * from
customer");

                StringTokenizer st=new
StringTokenizer(OrderIDList.getSelectedItem(),"->");

                String p=st.nextToken();
                while (rs.next())
                {
                    if
(rs.getString("customer_id").equals(p))

                    break;
                }
                if (!rs.isAfterLast())
                {

                    txtcustomer_id.setText(rs.getString("customer_id"));

                    txtcustomer_name.setText(rs.getString("customer_name"));

                    txtgender.setText(rs.getString("gender"));

                    txtcontactno.setText(rs.getString("contactno"));

                }
            }
            catch (SQLException selectException)
            {
                displaySQLErrors(selectException);
            }
        });
        p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

        deleteButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
            }
        });

        try {

            int a=JOptionPane.showConfirmDialog(p,"Are you sure want to
delete:");

            if(a==JOptionPane.YES_OPTION){

                StringTokenizer st=new
StringTokenizer(
OrderIDList.getSelectedItem(),"->");

                String query="DELETE FROM customer WHERE
customer_id="+st.nextToken();

                int i=statement.executeUpdate(query);
            }
        }
    }
}

```

```

        JOptionPane.showMessageDialog(p, "\nDeleted      "+i+"      rows
succesfully");loadcustomers();
    }
}
catch(SQLException deleteException){
    displaySQLErrors(deleteException);
}
}

});

}
});

update.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        JButton updateButton = new JButton("modify");
        txtcustomer_id.setText(null);
        txtcustomer_name.setText(null);
        txtgender.setText(null);
        txtcontactno.setText(null);

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        p1=new JPanel();

        p1.setLayout(new GridLayout(4,2));
        p1.add(lblcustomer_id);
        p1.add(txtcustomer_id);
        p1.add(lblcustomer_name);
        p1.add(txtcustomer_name);
        p1.add(lblgender);
        p1.add(txtgender);
        p1.add(lblcontactno);
        p1.add(txtcontactno);

        p3=new JPanel(new FlowLayout());
        p3.add(updateButton);
        //p1.add(txtf1);
        p3.setBackground(Color.yellow);
        p3.setBounds(200,350,75,35); p1.setBounds(115,80,300,250);
        p1.setBackground(Color.pink) ;

        p2 = new JPanel(new FlowLayout());

```

```
OrderIDList=new List(10);
loadcustomers();
p2.add(OrderIDList);p2.setBackground(Color.cyan) ;

p2.setBounds(450,150,350,180);

p. add(p1);p.add(p3);
p. add(p2);

OrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs=statement.executeQuery("select * from
customer");

            while (rs.next())
            {
                if
(rs.getString("customer_id").equals(OrderIDList.getSelectedItem()))
                break;
            }
            if (!rs.isAfterLast())
            {

                txtcustomer_id.setText(rs.getString("customer_id"));

                txtcustomer_name.setText(rs.getString("customer_name"));

                txtgender.setText(rs.getString("gender"));

                txtcontactno.setText(rs.getString("contactno"));

            }
        }
        catch (SQLException selectException)
        {
            displaySQLErrors(selectException);
        }
    }
});
p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();
```

```

        updateButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                try {
                    loadcustomers();
                    String
contactno=JOptionPane.showInputDialog(p,"enter the contactno");

                    txtcontactno.setText(contactno);
                    //int
a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
                    //if(a==JOptionPane.YES_OPTION){
                    String query="update customer set
contactno='"+contactno+"'where customer_id="+txtcustomer_id.getText();

                    int i=statement.executeUpdate(query);

                    JOptionPane.showMessageDialog(p,"\nupdated "+i+" rows succesfully");loadcustomers();
                }

                catch(SQLException deleteException){
                    displaySQLErrors(deleteException);
                }
            }
        });
    }
});

view.addActionListener(new ActionListener(){
    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        Label view1=new Label("customer view");
        //view1.setAlignment(Label.CENTER);
        Font myFont = new Font("Serif",Font.BOLD,50);
        view1.setFont((myFont));
        viewButton=new JButton("View");
        p1=new JPanel();
        p2=new JPanel();
        p1.add(view1);
        p2.add(viewButton);p1.setBackground(Color.cyan)
;p2.setBackground(Color.cyan) ;

        p.add(p1);p.add(p2);
        p.setLayout(new FlowLayout());

```

```
p.setBounds(500,800,300,300);
frame.add(p);
frame.setSize(800,800);
frame.validate();
viewButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        JFrame f;

        JTable j;

        f = new JFrame();

        f.setTitle("Customer Details ");

        DefaultTableModel model = new
DefaultTableModel();

        j = new JTable(model);
        model.addColumn("customer id");
        model.addColumn("customer name");
        model.addColumn("gender");
        model.addColumn("contactno");
        try {

            rs=statement.executeQuery("select * from customer");

            while(rs.next()) {
                model.addRow(new
Object[]{rs.getString("customer_id"),
rs.getString("customer_name"),rs.getString("gender"),rs.getString("contactno")});
            }
        } catch(SQLException viewException) {

            displaySQLErrors(viewException);

        }
        j.setEnabled(false);
        j.setBounds(30, 40, 300, 300);

        JScrollPane sp = new JScrollPane(j);
        f.add(sp);

        f.setSize(800, 400);

        f.setVisible(true);

    }
}
```

```

        });

    }

});

}

}

```

Supplier:

```

package DrinkB;
import java.awt.BorderLayout;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.*;
import java.util.StringTokenizer;
import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class Supp{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JButton insertButton,deleteButton,updateButton,viewButton;
    private JPanel p1,p2,p3,p;
    //private JLabel lblorder_id,bltot,blstatus;
    private JLabel lblsupplier_id,blsupplier_name,blcontact_no,blmailaddress;

```

Drinksbucks store

```
//private JTextField txtorder_id,txttot,txtstatus;
private JTextField txtsupplier_id,txtsupplier_name,txtcontact_no,txtmailaddress;
private List OrderIDList;
Connection con;ResultSet rs;
Statement statement;
private JFrame frame;
private JMenuItem insert,delete,update,view;
public Supp(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem update,JMenuItem
view)
{
    try
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch (Exception e)
    {
        System.err.println("Unable to find and load driver");
        System.exit(1);
    }
    connectToDB();

    this.frame=frame;
    this.insert=insert;
    this.delete=delete;
    this.update=update;
    this.view=view;

    lblsupplier_id=new JLabel("supplier id");
    lblsupplier_name=new JLabel("supplier name");
    lblcontact_no=new JLabel("contact no");
    lblmailaddress=new JLabel("mail address");

    txtsupplier_id=new JTextField(15);
    txtsupplier_name=new JTextField(15);
    txtcontact_no=new JTextField(8);
    txtmailaddress=new JTextField(8);

    this.p=p;

}

public void connectToDB()
{
    try {

        Connection con=DriverManager.getConnection(
            "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");
```

```

statement=con.createStatement();
statement.executeUpdate("commit");

    }
    catch (SQLException connectException)
    {
        System.out.println(connectException.getMessage());
        System.out.println(connectException.getSQLState());
        System.out.println(connectException.getErrorCode());
        System.exit(1);
    }
}

private void displaySQLErrors(SQLException e)
{
    JOptionPane.showMessageDialog(p, "\nSQLException: " + e.getMessage() + "\n" + "SQLState: " + e.getSQLState() + "\n" + "VendorError: " + e.getErrorCode() + "\n");

}

public void loadsuppliers() {
    try {
        OrderIDList.removeAll();
        rs=statement.executeQuery("select * from supplier");
        while(rs.next()) {
            OrderIDList.add(rs.getString("supplier_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void buildGUI() {

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub
            insertButton=new JButton("submit");
            txtsupplier_id.setText(null);
            txtsupplier_name.setText(null);
            txtcontact_no.setText(null);
            txtmailaddress.setText(null);

            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();
        }
    });
}

```



```
p1=new JPanel();

p1.setLayout(new GridLayout(4,2));
p1.add(lblsupplier_id);
p1.add(txtsupplier_id);
p1.add(lblsupplier_name);
p1.add(txtsupplier_name);
p1.add(lblcontact_no);
p1.add(txtcontact_no);
p1.add(lblmailaddress);
p1.add(txtmailaddress);


p3=new JPanel(new FlowLayout());
p3.add(insertButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p3.setBounds(200,350,75,35); p1.setBounds(115,80,300,250);
p1.setBackground(Color.pink) ;


p2 = new JPanel(new FlowLayout());


OrderIDList=new List(10);
loadsuppliers();
p2.add(OrderIDList);p2.setBackground(Color.cyan) ;

p2.setBounds(450,150,350,180);


p. add(p1);p.add(p3);
p. add(p2);


p.setLayout(new BorderLayout());


frame.add(p);
frame.setSize(800,800);
frame.validate();


insertButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
try {
        String query="INSERT INTO supplier
VALUES("+txtsupplier_id.getText()+",""+txtsupplier_name.getText()+",""+txtcontact_no.getText()+",""+txtmaila
ddress.getText()+""");
```

```

        int i=statement.executeUpdate(query);
        JOptionPane.showMessageDialog(p,"\ninserted      "+i+"      rows
succesfully");

        loadsuppliers();

    }
    catch(SQLException insertException){
        displaySQLErrors(insertException);
    }

}

});

}
});

delete.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        deleteButton=new JButton("delete");

        txtsupplier_id.setText(null);
        txtsupplier_name.setText(null);
        txtcontact_no.setText(null);
        txtmailaddress.setText(null);

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        p1=new JPanel();

        p1.setLayout(new GridLayout(4,2));

        p1.add(lblsupplier_id);
        p1.add(txtsupplier_id);
        p1.add(lblsupplier_name);
        p1.add(txtsupplier_name);
        p1.add(lblcontact_no);
        p1.add(txtcontact_no);
        p1.add(lblmailaddress);
        p1.add(txtmailaddress);

        p3=new JPanel(new FlowLayout());
        p3.add(deleteButton);
        //p1.add(txtf1);
        p3.setBackground(Color.yellow);
        p3.setBounds(200,350,75,35); p1.setBounds(115,80,300,250);
        p1.setBackground(Color.pink) ;

```

```
// p1.setBounds(100,100,500,300);

p2 = new JPanel(new FlowLayout());

OrderIDList=new List(10);
loadsuppliers();
p2.add(OrderIDList);p2.setBackground(Color.cyan) ;

p2.setBounds(450,150,350,180);


p. add(p1);p.add(p3);
p. add(p2);


OrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs=statement.executeQuery("select * from
supplier");
            StringTokenizer st=new
StringTokenizer(OrderIDList.getSelectedItem(),"->");
            String p=st.nextToken();
            while (rs.next())
            {
                if
(rs.getString("supplier_id").equals(p))
                break;
            }
            if (!rs.isAfterLast())
            {
                txtsupplier_id.setText(rs.getString("supplier_id"));
                txtsupplier_name.setText(rs.getString("supplier_name"));
                txtcontact_no.setText(rs.getString("contact_no"));
                txtmailaddress.setText(rs.getString("mailaddress"));
            }
        }
        catch (SQLException selectException)
        {
            displaySQLErrors(selectException);
        }
    }
}
```

```

        }
    });
    p.setLayout(new BorderLayout());

    frame.add(p);
    frame.setSize(800,800);
    frame.validate();

    deleteButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
try {

            int a=JOptionPane.showConfirmDialog(p,"Are you sure want to
delete:");

            if(a==JOptionPane.YES_OPTION){
                StringTokenizer st=new
StringTokenizer(OrderIDList.getSelectedItem(),"->");
                String query="DELETE FROM supplier WHERE
supplier_id="+st.nextToken();

                int i=statement.executeUpdate(query);
                JOptionPane.showMessageDialog(p,"\nDeleted "+i+" rows
succesfully");loadsuppliers();

            }

        }
        catch(SQLException deleteException){
            displaySQLErrors(deleteException);
        }

    }

    });

}
});

update.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        JButton updateButton = new JButton("modify");
        txtsupplier_id.setText(null);
        txtsupplier_name.setText(null);
        txtcontact_no.setText(null);
        txtmailaddress.setText(null);

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();
    }
});

```

```
p1=new JPanel();

p1.setLayout(new GridLayout(4,2));
p1.add(lblsupplier_id);
p1.add(txtsupplier_id);
p1.add(lblsupplier_name);
p1.add(txtsupplier_name);
p1.add(lblcontact_no);
p1.add(txtcontact_no);
p1.add(lblmailaddress);
p1.add(txtmailaddress);

p3=new JPanel(new FlowLayout());
p3.add(updateButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p3.setBounds(200,350,75,35); p1.setBounds(115,80,300,250);
p1.setBackground(Color.pink) ;

p2 = new JPanel(new FlowLayout());

OrderIDList=new List(10);
loadsuppliers();
p2.add(OrderIDList);p2.setBackground(Color.cyan) ;

p2.setBounds(450,150,350,180);

p. add(p1);p.add(p3);
p. add(p2);
```

```
OrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs=statement.executeQuery("select * from
supplier");

            while (rs.next())
            {
                if
(rs.getString("supplier_id").equals(OrderIDList.getSelectedItem()))
                break;
            }
            if (!rs.isAfterLast())
            {
```

```

txtsupplier_id.setText(rs.getString("supplier_id"));

txtsupplier_name.setText(rs.getString("supplier_name"));

txtcontact_no.setText(rs.getString("contact_no"));

txtmailaddress.setText(rs.getString("mailaddress"));

        }
    }
    catch (SQLException selectException)
    {
        displaySQLErrors(selectException);
    }
}

});
p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

updateButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        try {
            loadsuppliers();
            String
contact_no=JOptionPane.showInputDialog(p,"enter the contactno");

            txtcontact_no.setText(contact_no);
            //int
a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
            //if(a==JOptionPane.YES_OPTION){
            String query="update supplier set
contact_no='"+contact_no+"'where supplier_id="+txtsupplier_id.getText();

            //int
a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
            //if(a==JOptionPane.YES_OPTION){
            //String query="update supplier set
supplier_name="+txtsupplier_name.getText()+",contact_no="+txtcontact_no.getText()+",mailaddress="+txtmail
address.getText()+" WHERE supplier_id="+OrderIDList.getSelectedItem();
            int i=statement.executeUpdate(query);

            JOptionPane.showMessageDialog(p,"\nupdated "+i+" rows succesfully");loadsuppliers();
        }

        catch(SQLException deleteException){
            displaySQLErrors(deleteException);
        }
    }
}

```

```
        }

        });

    }
});

view.addActionListener(new ActionListener(){

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        Label view1=new Label("suppliers view");
        //view1.setAlignment(Label.CENTER);
        Font myFont = new Font("Serif",Font.BOLD,50);
        view1.setFont(myFont);
        viewButton=new JButton("View");
        p1=new JPanel();
        p2=new JPanel();
        p1.add(view1);
        p2.add(viewButton);p1.setBackground(Color.cyan)
;p2.setBackground(Color.cyan) ;
        p.add(p1);p.add(p2);
        p.setLayout(new FlowLayout());

        p.setBounds(500,800,300,300);
        frame.add(p);
        frame.setSize(800,800);
        frame.validate();
        viewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub

                JFrame f;

                JTable j;

                f = new JFrame();

                f.setTitle("orderDetails ");

                DefaultTableModel model = new
DefaultTableModel();

                j = new JTable(model);
                model.addColumn("supplier id");
```

```

        model.addColumn("supplier name");
        model.addColumn("contact no");
        model.addColumn("mail address");

        try {

            rs=statement.executeQuery("select * from supplier");

            while(rs.next()) {
                model.addRow(new
Object[] {rs.getString("supplier_id"),
rs.getString("supplier_name"),rs.getString("contact_no"),rs.getString("mailaddress")});
            }
        } catch(SQLException viewException) {

            displaySQLErrors(viewException);

        }
        j.setEnabled(false);
        j.setBounds(30, 40, 300, 300);

        JScrollPane sp = new JScrollPane(j);
        f.add(sp);

        f.setSize(800, 400);

        f.setVisible(true);

    }

});

}

});

}

}

```

Drinks:

```

package DrinkB;
import java.awt.BorderLayout;
import java.awt.Color;

```


Drinksbucks store

```
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.*;
import java.util.StringTokenizer;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class Drinks{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JButton insertButton,deleteButton,updateButton,viewButton;
    private JPanel p1,p2,p3,p;
    //private JLabel lblmid,lblmname,lblprice,lblinstock;
    private JLabel lbl drinksid,lbl drinksname,lbl quantity,lbl price;
    //private JTextField txtmid,txtmname,txtprice,txtinstock;
    private JTextField txt drinksid,txt drinksname,txt quantity,txt price;

    private List MIDList;
    Connection con;ResultSet rs;
    Statement statement;
    private JFrame frame;
    private JMenuItem insert,delete,update,view;
    public Drinks(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem update,JMenuItem
view)
    {

        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
            System.err.println("Unable to find and load driver");
            System.exit(1);
        }
        connectToDB();
    }
}
```

```

        this.frame=frame;
        this.insert=insert;
        this.delete=delete;
        this.update=update;
        this.view=view;

        lbl drinksid=new JLabel("Drinks id");
        lbl drinksname=new JLabel("Drinks Name");
        lbl quantity=new JLabel("quantity");
        lbl price=new JLabel("price");

        txt drinksid=new JTextField(15);
        txt drinksname=new JTextField(15);
        txt quantity=new JTextField(15);
        txt price=new JTextField(8);

        this.p=p;

    }

    public void connectToDB()
    {
        try {

            Connection con=DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");

            statement=con.createStatement();
            statement.executeUpdate("commit");

        }
        catch (SQLException connectException)
        {
            System.out.println(connectException.getMessage());
            System.out.println(connectException.getSQLState());
            System.out.println(connectException.getErrorCode());
            System.exit(1);
        }
    }

    private void displaySQLExceptions(SQLException e)
    {
        JOptionPane.showMessageDialog(p,"\\nSQLException: " + e.getMessage() + "\\n"+"SQLState:
" + e.getSQLState() + "\\n"+"VendorError: " + e.getErrorCode() + "\\n");

    }

    public void loaddrinks() {

```

```
try {
    MIDList.removeAll();
    rs=statement.executeQuery("select * from drinks ");
    while(rs.next()) {
        MIDList.add(rs.getString("drinks_id"));
    }
}
catch(SQLException e) {
    displaySQLErrors(e);
}

}

public void buildGUI() {

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {

            // TODO Auto-generated method stub
            insertButton=new JButton("submit");
            txtdrinksid.setText(null);
            txtdrinksname.setText(null);
            txtquantity.setText(null);
            txtprice.setText(null);

            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();

            p1=new JPanel();

            p1.setLayout(new GridLayout(4,2));
            p1.add(lbldrinksid);
            p1.add(txtdrinksid);
            p1.add(lbldrinksname);
            p1.add(txtdrinksname);
            p1.add(lblquantity);
            p1.add(txtquantity);
            p1.add(lblprice);
            p1.add(txtprice);

            p3=new JPanel(new FlowLayout());
            p3.add(insertButton);
            //p1.add(txtf1);
            p3.setBackground(Color.yellow);
            p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
```

```

p1.setBackground(Color.pink) ;

p2 = new JPanel(new FlowLayout());

MIDList=new List(10);
loaddrinks();
p2.add(MIDList);p2.setBackground(Color.cyan) ;

p2.setBounds(450,150,350,180);

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

insertButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        try {
            String query="INSERT INTO drinks
VALUES("+txt drinksid.getText()+","+txt drinksname.getText()+","+txt quantity.getText()+","+txt price.getText(
)+")";

            int i=statement.executeUpdate(query);
            JOptionPane.showMessageDialog(p,"\ninserted "+i+" rows
succesfully");loaddrinks();

        }
        catch(SQLException insertException){
            displaySQLErrors(insertException);
        }

    }

});

}
});

delete.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {

```

```
// TODO Auto-generated method stub
deleteButton=new JButton("delete");

txtdrinksid.setText(null);
txtdrinksname.setText(null);
txtquantity.setText(null);
txtprice.setText(null);


p.removeAll();
frame.invalidate();
frame.validate();
frame.repaint();


p1=new JPanel();

p1.setLayout(new GridLayout(4,2));
p1.add(lbldrinksid);
p1.add(txtdrinksid);
p1.add(lbldrinksname);
p1.add(txtdrinksname);
p1.add(lblquantity);
p1.add(txtquantity);
p1.add(lblprice);
p1.add(txtprice);

p3=new JPanel(new FlowLayout());
p3.add(deleteButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;


// p1.setBounds(100,100,500,300);

p2 = new JPanel(new FlowLayout());

    MIDList=new List(10);
    loaddrinks();
    p2.add(MIDList);p2.setBackground(Color.cyan) ;

    p2.setBounds(450,150,350,180);

p. add(p1);p.add(p3);
p. add(p2);
MIDList.addItemListener(new ItemListener()
    {
        public void itemStateChanged(ItemEvent e)
        {
            try
```

```

        {
            rs=statement.executeQuery("select * from
drinks");

            StringTokenizer st=new
StringTokenizer(MIDList.getSelectedItem(),"->");

            String p=st.nextToken();
            while (rs.next())
            {
                if
                break;
            }
            if (!rs.isAfterLast())
            {

                txtdrinksid.setText(rs.getString("drinks_id"));

                txtdrinksname.setText(rs.getString("drinks_name"));

                txtquantity.setText(rs.getString("quantity"));

                txtprice.setText(rs.getString("price"));

            }
        }
        catch (SQLException selectException)
        {
            displaySQLErrors(selectException);
        }
    }
});

p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

deleteButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
    try {

        int a=JOptionPane.showConfirmDialog(p,"Are you sure want to
delete:");

        if(a==JOptionPane.YES_OPTION){
            StringTokenizer st=new
StringTokenizer(MIDList.getSelectedItem(),"->");
            String query="DELETE FROM DRINKS WHERE
drinks_id="+st.nextToken();

            int i=statement.executeUpdate(query);
            JOptionPane.showMessageDialog(p,"\nDeleted "+i+" rows
succesfully");loaddrinks();

        }
    }
}

```

```
        }
        catch(SQLException deleteException){
            displaySQLErrors(deleteException);
        }
    }

    });

}
});

update.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        JButton updateButton = new JButton("modify");
        txtdrinksid.setText(null);
        txtdrinksname.setText(null);
        txtquantity.setText(null);
        txtprice.setText(null);

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        p1=new JPanel();

        p1.setLayout(new GridLayout(4,2));
        p1.add(lbldrinksid);
        p1.add(txtdrinksid);
        p1.add(lbldrinksname);
        p1.add(txtdrinksname);
        p1.add(lblquantity);
        p1.add(txtquantity);
        p1.add(lblprice);
        p1.add(txtprice);
        p3=new JPanel(new FlowLayout());
        p3.add(updateButton);
        //p1.add(txtf1);
        p3.setBackground(Color.yellow);
        p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
        p1.setBackground(Color.pink) ;

        p2 = new JPanel(new FlowLayout());

        MIDList=new List(10);
        loaddrinks();
```

```

        p2.add(MIDList);p2.setBackground(Color.cyan) ;

        p2.setBounds(450,150,350,180);

        p. add(p1);p.add(p3);
        p. add(p2);
        MIDList.addItemListener(new ItemListener()
        {
            public void itemStateChanged(ItemEvent e)
            {
                try
                {
                    rs=statement.executeQuery("select * from
drinks");

                    while (rs.next())
                    {
                        if
(rs.getString("drinks_id").equals(MIDList.getSelectedItem()))

                        break;
                    }
                    if (!rs.isAfterLast())
                    {

                        txtdrinksid.setText(rs.getString("drinks_id"));

                        txtdrinksname.setText(rs.getString("drinks_name"));

                        txtquantity.setText(rs.getString("quantity"));

                        txtprice.setText(rs.getString("price"));

                    }
                }
                catch (SQLException selectException)
                {
                    displaySQLErrors(selectException);
                }
            }
        });

        p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

        updateButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                try {
                    loaddrinks();
                    String
price=JOptionPane.showInputDialog(p,"enter the new price");

                    txtprice.setText(price);

```


Drinksbucks store

```

//      int
a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
//      if(a==JOptionPane.YES_OPTION){
String      query="update      drinks      set
price="" +price+"where drinks_id="+txtdrinksid.getText();

int i=statement.executeUpdate(query);

JOptionPane.showMessageDialog(p,"\nupdated "+i+" rows succesfully");loaddrinks();
}

catch(SQLException deleteException){
    displaySQLErrors(deleteException);
}

}

});

}
});

view.addActionListener(new ActionListener(){

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        Label view1=new Label("Drinks view");
        //view1.setAlignment(Label.CENTER);
        Font myFont = new Font("Serif",Font.BOLD,50);
        view1.setFont((myFont));
        viewButton=new JButton("View");
        p1=new JPanel();
        p2=new JPanel();
        p1.add(view1);
        p2.add(viewButton);p1.setBackground(Color.cyan)
;p2.setBackground(Color.cyan) ;
        p.add(p1);p.add(p2); p.setLayout(new FlowLayout());
        frame.add(p);
        frame.setSize(800,800);
        frame.validate();
        viewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                JFrame f;
```

```

JTable j;

f = new JFrame();

f.setTitle("Drinks details");

DefaultTableModel model = new
DefaultTableModel();

j = new JTable(model);
model.addColumn("Drinks id");
model.addColumn("Drinks Name");
model.addColumn("quantity");
model.addColumn("price");

try {

    rs=statement.executeQuery("select * from drinks");

    while(rs.next()) {
        model.addRow(new
Object[]{rs.getString("drinks_id"), rs.getString("drinks_name"),rs.getString("quantity"),rs.getString("price")});
    }
    catch(SQLException viewException) {

        displaySQLErrors(viewException);

    }
    j.setEnabled(false);
    j.setBounds(30, 40, 300, 300);

    JScrollPane sp = new JScrollPane(j);
    f.add(sp);

    f.setSize(800, 400);

    f.setVisible(true);

}

});

}

});

```

```
    }  
  
}
```

Torder:

```
package DrinkB;  
import java.awt.BorderLayout;  
import java.awt.Choice;  
import java.awt.Color;  
import java.awt.FlowLayout;  
import java.awt.Font;  
import java.awt.GridLayout;  
import java.awt.Label;  
import java.awt.List;  
import java.awt.event.ActionEvent;  
import java.awt.event.ActionListener;  
import java.awt.event.ItemEvent;  
import java.awt.event.ItemListener;  
import java.sql.*;  
import java.util.StringTokenizer;  
  
import javax.swing.JButton;  
import javax.swing.JFrame;  
import javax.swing.JLabel;  
import javax.swing.JMenuItem;  
import javax.swing.JOptionPane;  
import javax.swing.JPanel;  
import javax.swing.JScrollPane;  
import javax.swing.JTable;  
import javax.swing.JTextField;  
import javax.swing.table.DefaultTableModel;  
  
public class Order{  
    /**  
     *  
     */  
    private static final long serialVersionUID = 1L;  
    private JButton insertButton,deleteButton,updateButton,viewButton;  
    private JPanel p1,p2,p3,p;  
    //private JLabel lbluser_id,lblmid,blborder_id,bllday;  
    private JLabel lblorder_id,blcustomer_id,blldrinks_id,blqty,blprice_perqty;  
    //private JTextField txtuser_id,txtmid,txtday,txtorder_id;  
    //private Choice user_id,order_id,mid;  
    private JTextField txtorder_id,txtcustomer_id,txtdrinks_id,txtqty,txtprice_perqty;  
    private Choice customer_id,drinks_id;  
    private List UserOrderIDList;  
    Connection con;ResultSet rs;  
    Statement statement;  
    private JFrame frame;
```

```

private JMenuItem insert,delete,update,view;
public Order(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem update,JMenuItem
view)
{

    try
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch (Exception e)
    {
        System.err.println("Unable to find and load driver");
        System.exit(1);
    }
    connectToDB();

    this.frame=frame;
    this.insert=insert;
    this.delete=delete;
    this.update=update;
    this.view=view;

    lblorder_id=new JLabel("Order ID");
    lblcustomer_id=new JLabel("Customer ID");
    lbldrinks_id=new JLabel("Drinks ID");
    lblqty=new JLabel("quantity");
    lblprice_perqty=new JLabel("priceperquantity");
    //order_id=new Choice();
    customer_id=new Choice();
    drinks_id=new Choice();
    txtorder_id=new JTextField(15);
    txtcustomer_id=new JTextField(15);
    txtdrinks_id=new JTextField(15);
    txtqty=new JTextField(8);
    txtprice_perqty=new JTextField(8);

    this.p=p;

}

public void connectToDB()
{
    try {

        Connection con=DriverManager.getConnection(
        "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");

        statement=con.createStatement();
        statement.executeUpdate("commit");
    }
}

```

Drinksbucks store

```
    }
    catch (SQLException connectException)
    {
        System.out.println(connectException.getMessage());
        System.out.println(connectException.getSQLState());
        System.out.println(connectException.getErrorCode());
        System.exit(1);
    }
}

private void displaySQLErrors(SQLException e)
{
    JOptionPane.showMessageDialog(p, "\nSQLException: " + e.getMessage() + "\n" + "SQLState: " + e.getSQLState() + "\n" + "VendorError: " + e.getErrorCode() + "\n");
}

public void loadorders() {
    try {
        UserOrderIDList.removeAll();
        rs=statement.executeQuery("select * from torder");
        while(rs.next()) {
            UserOrderIDList.add(rs.getString("order_id")+"->" +
rs.getString("customer_id")+"->" +rs.getString("drinks_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void loadcustomers() {
    try {
        customer_id.removeAll();
        rs=statement.executeQuery("select * from customer");
        while(rs.next()) {
            customer_id.add(rs.getString("customer_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void loaddrinks() {
    try {
        drinks_id.removeAll();
        rs=statement.executeQuery("select * from drinks");
        while(rs.next()) {
            drinks_id.add(rs.getString("drinks_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}
```

```

public void buildGUI() {

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub

            insertButton=new JButton("submit");
            loadcustomers();
            loaddrinks();
            txtorder_id.setText(null);
            //txtcustomer_id.setText(null);
            //txtdrinks_id.setText(null);
            txtqty.setText(null);
            txtprice_perqty.setText(null);
            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();
            // loadorders();
            // loadcustomers();
            // loaddrinks();

            p1=new JPanel();

            p1.setLayout(new GridLayout(5,1));

            p1.add(lblorder_id);
            p1.add(txtorder_id);
            p1.add(lblcustomer_id);
            p1.add(customer_id);
            p1.add(lbldrinks_id);
            p1.add(drinks_id);
            p1.add(lblqty);
            p1.add(txtqty);
            p1.add(lblprice_perqty);
            p1.add(txtprice_perqty);
            p3=new JPanel(new FlowLayout());
            p3.add(insertButton);
            //p1.add(txtf1);
            p3.setBackground(Color.yellow);
            p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
            p1.setBackground(Color.pink) ;

            p2 = new JPanel(new FlowLayout());

            UserOrderIDList=new List(10);
            loadorders();
            p2.add(UserOrderIDList);

```

```
p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan);

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

insertButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        try {
            String query="INSERT INTO TORDER
VALUES("+txtorder_id.getText()+","+customer_id.getSelectedItem()+","+drinks_id.getSelectedItem()+","+txtqty.getText()+","+txtprice_perqty.getText()+")";
            //System.out.println(query);
            int i=statement.executeUpdate(query);
            JOptionPane.showMessageDialog(p,"\ninserted "+i+" rows
succesfully");

            loadorders();
            //loadorders();
            loadcustomers();
            loaddrinks();

        }
        catch(SQLException insertException){
            displaySQLErrors(insertException);
        }

    }

});

delete.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        deleteButton=new JButton("delete");
        txtorder_id.setText(null);
```

```

txtcustomer_id.setText(null);
txtdrinks_id.setText(null);
txtqty.setText(null);
txtprice_perqty.setText(null);
p.removeAll();
frame.invalidate();
frame.validate();
frame.repaint();

p1=new JPanel();

p1.setLayout(new GridLayout(5,1));

p1.add(lblorder_id);
p1.add(txtorder_id);
p1.add(lblcustomer_id);
p1.add(txtcustomer_id);
p1.add(lbldrinks_id);
p1.add(txtdrinks_id);
p1.add(lblqty);
p1.add(txtqty);
p1.add(lblprice_perqty);
p1.add(txtprice_perqty);
p3=new JPanel(new FlowLayout());
p3.add(deleteButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;

// p1.setBounds(100,100,400,300);

p2 = new JPanel(new FlowLayout());

    UserOrderIDList=new List(10);
    loadorders();
    p2.add(UserOrderIDList);
    p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

    frame.add(p);
    frame.setSize(800,800);
    frame.validate();

UserOrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {

```


Drinksbucks store

```

        try
        {
            rs=statement.executeQuery("select * from
torder");

            StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

            String p=st.nextToken();
            String q=st.nextToken();
            String r=st.nextToken();
            while (rs.next())
            {
                if
(rs.getString("order_id").equals(p)      &&      rs.getString("customer_id").equals(q)      &&
rs.getString("drinks_id").equals(r))

                break;
            }
            if (!rs.isAfterLast())
            {

                txtorder_id.setText(rs.getString("order_id"));

                txtcustomer_id.setText(rs.getString("customer_id"));

                txtdrinks_id.setText(rs.getString("drinks_id"));

                txtqty.setText(rs.getString("qty"));

                txtprice_perqty.setText(rs.getString("price_perqty"));

            }
        }
        catch (SQLException selectException)
        {
            displaySQLErrors(selectException);
        }
    });
    p.setLayout(new BorderLayout());

    frame.add(p);
    frame.setSize(800,800);
    frame.validate();
    deleteButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            try {

                int a=JOptionPane.showConfirmDialog(p,"Are you sure want to
delete:");

                if(a==JOptionPane.YES_OPTION){
                    //StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

```

```

        String query="DELETE FROM TORDER WHERE
order_id="+txtorder_id.getText()+"";

        int i=statement.executeUpdate(query);
        JOptionPane.showMessageDialog(p,"\nDeleted "+i+" rows
succesfully");loadorders();
        //
        loadorders();
        loadcustomers();
        loaddrinks();
    }

    }
    catch(SQLException deleteException){
        displaySQLErrors(deleteException);
    }

    }

    });

}
});

update.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        JButton updateButton = new JButton("modify");
        txtorder_id.setText(null);
        txtcustomer_id.setText(null);
        txtdrinks_id.setText(null);
        txtqty.setText(null);
        txtprice_perqty.setText(null);
        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        p1=new JPanel();

        p1.setLayout(new GridLayout(5,1));

        p1.add(lblorder_id);
        p1.add(txtorder_id);
        p1.add(lblcustomer_id);
        p1.add(txtcustomer_id);
        p1.add(lbldrinks_id);
        p1.add(txtdrinks_id);
        p1.add(lblqty);
        p1.add(txtqty);
        p1.add(lblprice_perqty);
        p1.add(txtprice_perqty);
        p3=new JPanel(new FlowLayout());

```

Drinksbucks store

```
p3.add(updateButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;

p2 = new JPanel(new FlowLayout());

    UserOrderIDList=new List(10);
    loadorders();
    p2.add(UserOrderIDList);
    p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

    frame.add(p);
    frame.setSize(800,800);
    frame.validate();

UserOrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs=statement.executeQuery("select * from
torder");
            StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

            String p=st.nextToken();
            String q=st.nextToken();
            String r=st.nextToken();

            while (rs.next())
            {
                if
(rs.getString("order_id").equals(p)      &&      rs.getString("customer_id").equals(q)      &&
rs.getString("drinks_id").equals(r))

                    break;
            }
            if (!rs.isAfterLast())
            {

                txtorder_id.setText(rs.getString("order_id"));

                txtcustomer_id.setText(rs.getString("customer_id"));

                txtdrinks_id.setText(rs.getString("drinks_id"));
```

```

txtqty.setText(rs.getString("qty"));

txtprice_perqty.setText(rs.getString("price_perqty"));

        }
    }
    catch (SQLException selectException)
    {
        displaySQLErrors(selectException);
    }
}

});
p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

updateButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        try {

            loadorders();
            String

qty=JOptionPane.showInputDialog(p,"enter the qty");

            txtqty.setText(qty);
            //int

a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
            //if(a==JOptionPane.YES_OPTION){
                //StringTokenizer      st=new

StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

                //String  query="update  torder  set
order="+txttorder.getText()+"customer_id="+txtcustomer_id.getText()+"day="+txtday.getText()+" WHERE
ORDER_ID="+st.nextToken()+"and user_id="+st.nextToken();

                String  query="update  torder  set
qty="+qty+"where order_id="+txttorder_id.getText();

                int i=statement.executeUpdate(query);

JOptionPane.showMessageDialog(p,"\nupdated "+i+" rows succesfully");loadorders();
            //
                loadorders();
                loadcustomers();
                loaddrinks();
            }

            catch(SQLException deleteException){
                displaySQLErrors(deleteException);
            }

        }

    });

```

```
    }
    });
view.addActionListener(new ActionListener(){

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub

        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        Label view1=new Label("ORDER view");
        //view1.setAlignment(Label.CENTER);
        Font myFont = new Font("Serif",Font.BOLD,50);
        view1.setFont(myFont);
        viewButton=new JButton("View");
        p1=new JPanel();
        p2=new JPanel();
        p1.add(view1);
        p2.add(viewButton);p1.setBackground(Color.cyan)
;p2.setBackground(Color.cyan) ;
        p.add(p1);p.add(p2);
        p.setLayout(new FlowLayout());

        p.setBounds(500,800,300,300);
        frame.add(p);
        frame.setSize(800,800);
        frame.validate();
        viewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub

                JFrame f;

                JTable j;

                f = new JFrame();

                f.setTitle("order Details ");

                DefaultTableModel model = new
DefaultTableModel();

                j = new JTable(model);
                model.addColumn("Order ID");
                model.addColumn("Customer ID");
                model.addColumn("Drinks ID");

                model.addColumn("quantity");
                model.addColumn("priceperquantity");
```

```

        try {

            rs=statement.executeQuery("select * from torder");

            while(rs.next()) {
                model.addRow(new
Object[] {rs.getString("order_id"),
rs.getString("customer_id"),rs.getString("drinks_id"),rs.getString("qty"),rs.getString("price_perqty")});
            }
        } catch(SQLException viewException) {

            displaySQLErrors(viewException);

        }
        j.setEnabled(false);
        j.setBounds(30, 40, 300, 300);

        JScrollPane sp = new JScrollPane(j);
        f.add(sp);

        f.setSize(800, 400);

        f.setVisible(true);

    }

});

}

});

}

}

```

Payments:

```

package DrinkB;
import java.awt.BorderLayout;
import java.awt.Choice;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.List;

```

Drinksbucks store

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.*;
import java.util.StringTokenizer;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class pay{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JButton insertButton,deleteButton,updateButton,viewButton;
    private JPanel p1,p2,p3,p;
    private JLabel lblpayment_id,lbldorder_id,lbldcustomer_id,lbllttlquantity,lbllttlprice;
    private JTextField txtpayment_id,txtorder_id,txtcustomer_id,txtttlquantity,txtttlprice;
    private Choice order_id,customer_id;
    private List UserOrderIDList;
    Connection con;ResultSet rs;
    Statement statement;
    private JFrame frame;
    private JMenuItem insert,delete,update,view;
    public pay(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem update,JMenuItem
view)
    {

        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        }
        catch (Exception e)
        {
            System.err.println("Unable to find and load driver");
            System.exit(1);
        }
        connectToDB();

        this.frame=frame;
        this.insert=insert;
        this.delete=delete;
        this.update=update;
        this.view=view;
    }
}
```

```

        lblpayment_id=new JLabel("payment_id");
        lblorder_id=new JLabel("order_id");
        lblcustomer_id=new JLabel("customer_id");
        lblttlquantity=new JLabel("ttlquantity");
        lblttlprice=new JLabel("ttlprice");
        //payment_id=new Choice();
        order_id=new Choice();
        customer_id=new Choice();

        txtpayment_id=new JTextField(15);
        txtorder_id=new JTextField(15);
        txtcustomer_id=new JTextField(15);

        txttlquantity=new JTextField(8);
        txttlprice=new JTextField(8);
        this.p=p;

    }

    public void connectToDB()
    {
        try {

            Connection con=DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");

            statement=con.createStatement();
            statement.executeUpdate("commit");

        }
        catch (SQLException connectException)
        {
            System.out.println(connectException.getMessage());
            System.out.println(connectException.getSQLState());
            System.out.println(connectException.getErrorCode());
            System.exit(1);
        }
    }

    private void displaySQLExceptions(SQLException e)
    {
        JOptionPane.showMessageDialog(p,"\\nSQLException: " + e.getMessage() + "\\n"+"SQLState: " + e.getSQLState() + "\\n"+"VendorError: " + e.getErrorCode() + "\\n");
    }

    public void loadpayments() {
        try {
            UserOrderIDList.removeAll();
            rs=statement.executeQuery("select * from payments");
            while(rs.next()) {

```


Drinksbucks store

```

        UserOrderIDList.add(rs.getString("payment_id")+"-
>" +rs.getString("order_id")+"->" +rs.getString("customer_id"));
    }
}
catch(SQLException e) {
    displaySQLErrors(e);
}
}
public void loadorders() {
    try {
        order_id.removeAll();
        rs=statement.executeQuery("select * from torder");
        while(rs.next()) {
            order_id.add(rs.getString("order_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}
public void loadcustomers() {
    try {
        customer_id.removeAll();
        rs=statement.executeQuery("select * from customer");
        while(rs.next()) {
            customer_id.add(rs.getString("customer_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void buildGUI() {

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub
            insertButton=new JButton("submit");

            txtpayment_id.setText(null);
            // txtorder_id.setText(null);
            //txtcustomer_id.setText(null);
            txtttlquantity.setText(null);
            txtttlprice.setText(null);
            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();
            loadpayments();
        }
    });
}
```

```

loadorders();
loadcustomers();

p1=new JPanel();

p1.setLayout(new GridLayout(5,1));

p1.add(lblpayment_id);
p1.add(txtpayment_id);
p1.add(lblorder_id);
p1.add(order_id);
p1.add(lblcustomer_id);
p1.add(customer_id);
p1.add(lblttlquantity);
p1.add(txttlquantity);
p1.add(lblttlprice);
p1.add(txttlprice);
p3=new JPanel(new FlowLayout());
p3.add(insertButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;

p2 = new JPanel(new FlowLayout());

        UserOrderIDList=new List(10);
        loadpayments();
        p2.add(UserOrderIDList);
        p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

insertButton.addActionListener(new ActionListener() {
    @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
try {
            String query="INSERT INTO payments
VALUES("+txtpayment_id.getText()+","+order_id.getSelectedItem()+","+customer_id.getSelectedItem()+","+tx
ttlquantity.getText()+","+txttlprice.getText()+")";
            //System.out.println(query);
            int i=statement.executeUpdate(query);
            JOptionPane.showMessageDialog(p,"\ninserted "+i+" rows
succesfully");

            loadpayments();

```

```
        //loadpayment();
        loadorders();
        loadcustomers();

    }
    catch(SQLException insertException){
        displaySQLErrors(insertException);
    }

}

});

}
});

delete.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {
        // TODO Auto-generated method stub
        deleteButton=new JButton("delete");
        txtpayment_id.setText(null);
        txtorder_id.setText(null);
        txtcustomer_id.setText(null);
        txtttlquantity.setText(null);
        txtttlprice.setText(null);
        p.removeAll();
        frame.invalidate();
        frame.validate();
        frame.repaint();

        p1=new JPanel();

        p1.setLayout(new GridLayout(5,1));

        p1.add(lblpayment_id);
        p1.add(txtpayment_id);
        p1.add(lblorder_id);
        p1.add(txtorder_id);
        p1.add(lblcustomer_id);
        p1.add(txtcustomer_id);
        p1.add(lblttlquantity);
        p1.add(txtttlquantity);
        p1.add(lblttlprice);
        p1.add(txtttlprice);
        p3=new JPanel(new FlowLayout());
        p3.add(deleteButton);
        //p1.add(txtf1);
        p3.setBackground(Color.yellow);
        p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
```

```

p1.setBackground(Color.pink) ;

// p1.setBounds(100,100,400,300);

p2 = new JPanel(new FlowLayout());

    UserOrderIDList=new List(10);
    loadpayments();
    p2.add(UserOrderIDList);
    p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

    frame.add(p);
    frame.setSize(800,800);
    frame.validate();

UserOrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs=statement.executeQuery("select * from
payments");
            StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

            String p=st.nextToken();
            String q=st.nextToken();
            String r=st.nextToken();

            while (rs.next())
            {
                if
(rs.getString("payment_id").equals(p)      &&      rs.getString("order_id").equals(q)      &&
rs.getString("customer_id").equals(r))

                break;
            }
            if (!rs.isAfterLast())
            {

                txtpayment_id.setText(rs.getString("payment_id"));

                txtorder_id.setText(rs.getString("order_id"));

                txtcustomer_id.setText(rs.getString("customer_id"));

                txtttlquantity.setText(rs.getString("ttlquantity"));

                txtttlprice.setText(rs.getString("ttlprice"));

```

```

        }
    }
    catch (SQLException selectException)
    {
        displaySQLErrors(selectException);
    }
}

});
p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();
deleteButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
try {

        int a=JOptionPane.showConfirmDialog(p,"Are you sure want to
delete:");

        if(a==JOptionPane.YES_OPTION){
            //StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

            String query="DELETE FROM payments WHERE
payment_id="+txtpayment_id.getText()+"";

            int i=statement.executeUpdate(query);
            JOptionPane.showMessageDialog(p,"\nDeleted "+i+" rows
succesfully");

            loadpayments();
            // loadpayment();
            loadorders();
            loadcustomers();
        }

    }
    catch(SQLException deleteException){
        displaySQLErrors(deleteException);
    }

}

});

}
});

update.addActionListener(new ActionListener() {

    @Override
    public void actionPerformed(ActionEvent arg0) {

```

```

// TODO Auto-generated method stub
JButton updateButton = new JButton("modify");
txtpayment_id.setText(null);
txtorder_id.setText(null);
txtcustomer_id.setText(null);
txttlquantity.setText(null);
txttlprice.setText(null);
p.removeAll();
frame.invalidate();
frame.validate();
frame.repaint();

p1=new JPanel();

p1.setLayout(new GridLayout(5,1));

p1.add(lblpayment_id);
p1.add(txtpayment_id);
p1.add(lblorder_id);
p1.add(txtorder_id);
p1.add(lblcustomer_id);
p1.add(txtcustomer_id);
p1.add(lblttlquantity);
p1.add(txtttlquantity);
p1.add(lblttlprice);
p1.add(txtttlprice);

p3=new JPanel(new FlowLayout());
p3.add(updateButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;

p2 = new JPanel(new FlowLayout());

        UserOrderIDList=new List(10);
        loadpayments();
        p2.add(UserOrderIDList);
        p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

UserOrderIDList.addItemListener(new ItemListener()
{
        public void itemStateChanged(ItemEvent e)

```

Drinksbucks store

```
        {
            try
            {
                rs=statement.executeQuery("select * from
payments");

                StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

                String p=st.nextToken();
                String q=st.nextToken();
                String r=st.nextToken();
                while (rs.next())
                {
                    if
(rs.getString("payment_id").equals(p)      &&      rs.getString("order_id").equals(q)      &&
rs.getString("customer_id").equals(r))

                        break;
                }
                if (!rs.isAfterLast())
                {

                    txtpayment_id.setText(rs.getString("payment_id"));

                    txtorder_id.setText(rs.getString("order_id"));

                    txtcustomer_id.setText(rs.getString("customer_id"));

                    txtttlquantity.setText(rs.getString("ttlquantity"));

                    txtttlprice.setText(rs.getString("ttlprice"));

                }
            }
            catch (SQLException selectException)
            {
                displaySQLErrors(selectException);
            }
        }
    });
    p.setLayout(new BorderLayout());

    frame.add(p);
    frame.setSize(800,800);
    frame.validate();

    updateButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            try {

                loadpayments();
                String

                ttlquantity=JOptionPane.showInputDialog(p,"enter the qty");

                txtttlquantity.setText(ttlquantity);
```

```

//int
a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
//if(a==JOptionPane.YES_OPTION){
//StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");
String query="update payments set
ttlquantity='"+ttlquantity+"'where payment_id="+txtpayment_id.getText();
int i=statement.executeUpdate(query);

JOptionPane.showMessageDialog(p,"\nupdated "+i+" rows succesfully");loadpayments();
//loadpayment();
loadorders();
loadcustomers();
}

catch(SQLException deleteException){
displaySQLErrors(deleteException);
}

}

});

}
});
view.addActionListener(new ActionListener(){

@Override
public void actionPerformed(ActionEvent arg0) {
// TODO Auto-generated method stub

p.removeAll();
frame.invalidate();
frame.validate();
frame.repaint();

Label view1=new Label("payments view");
//view1.setAlignment(Label.CENTER);
Font myFont = new Font("Serif",Font.BOLD,50);
view1.setFont(myFont);
viewButton=new JButton("View");
p1=new JPanel();
p2=new JPanel();
p1.add(view1);
p2.add(viewButton);p1.setBackground(Color.cyan)
;p2.setBackground(Color.cyan) ;
p.add(p1);p.add(p2);
p.setLayout(new FlowLayout());

p.setBounds(500,800,300,300);
frame.add(p);
frame.setSize(800,800);
frame.validate();

```



```
viewButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
        JFrame f;

        JTable j;

        f = new JFrame();

        f.setTitle("payments Details ");

        DefaultTableModel model = new
DefaultTableModel();

        j = new JTable(model);
        model.addColumn("payment_id");
        model.addColumn("order_id");
        model.addColumn("customer_id");

        model.addColumn("ttlquantity");
        model.addColumn("ttlprice");

        try {

            rs=statement.executeQuery("select * from payments");

            while(rs.next()) {
                model.addRow(new
Object[]{rs.getString("payment_id"),
rs.getString("order_id"),rs.getString("customer_id"),rs.getString("ttlquantity"),rs.getString("ttlprice")});
            }
        } catch(SQLException viewException) {

            displaySQLErrors(viewException);

        }
        j.setEnabled(false);
        j.setBounds(30, 40, 180, 150);

        JScrollPane sp = new JScrollPane(j);
        f.add(sp);

        f.setSize(500, 400);

        f.setVisible(true);

    }
}
```

```

        });

    }

});

}

}

```

Transaction:

```

package DrinkB;
import java.awt.BorderLayout;
import java.awt.Choice;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.*;
import java.util.StringTokenizer;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class trans{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
    private JButton insertButton,deleteButton,updateButton,viewButton;
    private JPanel p1,p2,p3,p;
    private
    lbltransaction_id,lblcustomer_id,blmployee_id,blorder_id,bldrinks_id,blpayment_id,blsupplier_id;
}

```

Drinksbucks store

```
private                                                                    JTextField
txttransaction_id,txtpayment_id,txtemployee_id,txtcustomer_id,txtdrinks_id,txtorder_id,txtsupplier_id;
private Choice employee_id,drinks_id,payment_id,order_id,customer_id,supplier_id;
private List UserOrderIDList;
Connection con;ResultSet rs;
Statement statement;
private JFrame frame;
private JMenuItem insert,delete,view;
public trans(JPanel p,JFrame frame,JMenuItem insert,JMenuItem delete,JMenuItem view)
{

    try
    {
        Class.forName("oracle.jdbc.driver.OracleDriver");
    }
    catch (Exception e)
    {
        System.err.println("Unable to find and load driver");
        System.exit(1);
    }
    connectToDB();

    this.frame=frame;
    this.insert=insert;
    this.delete=delete;
    //this.update=update;
    this.view=view;

    lbltransaction_id=new JLabel("transaction_id");
    lblcustomer_id=new JLabel("customer_id");
    lblemployee_id=new JLabel("employee_id");
    lblorder_id=new JLabel("order_id");
    lbldrinks_id=new JLabel("drinks_id");
    lblpayment_id=new JLabel("payment_id");
    lblsupplier_id=new JLabel("supplier_id");
    //payment_id=new Choice();
    customer_id=new Choice();
    employee_id=new Choice();
    order_id=new Choice();
    drinks_id=new Choice();
    payment_id=new Choice();
    supplier_id=new Choice();
    txttransaction_id=new JTextField(15);
    txtcustomer_id=new JTextField(15);
    txtemployee_id=new JTextField(15);
    txtorder_id=new JTextField(8);
    txtdrinks_id=new JTextField(8);
    txtpayment_id=new JTextField(8);
    txtsupplier_id=new JTextField(8);
    this.p=p;

}
```

```

    public void connectToDB()
    {
        try {

            Connection con=DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:xe","it19737110","vasavi");

            statement=con.createStatement();
            statement.executeUpdate("commit");

        }
        catch (SQLException connectException)
        {
            System.out.println(connectException.getMessage());
            System.out.println(connectException.getSQLState());
            System.out.println(connectException.getErrorCode());
            System.exit(1);
        }
    }

    private void displaySQLErrors(SQLException e)
    {
        JOptionPane.showMessageDialog(p,"\nSQLException: " + e.getMessage() + "\n"+"SQLState:
" + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

    }

    public void loaduserOrderIDs() {
        try {
            UserOrderIDList.removeAll();
            rs=statement.executeQuery("select * from transaction");
            while(rs.next()) {
                UserOrderIDList.add(rs.getString("customer_id")+"-
>" +rs.getString("employee_id")+"->" +rs.getString("order_id")+"->" +rs.getString("drinks_id")+"-
>" +rs.getString("payment_id")+"->" +rs.getString("supplier_id"));
            }
        }
        catch(SQLException e) {
            displaySQLErrors(e);
        }
    }

    public void loadcustomers() {
        try {
            customer_id.removeAll();
            rs=statement.executeQuery("select * from customer");
            while(rs.next()) {
                customer_id.add(rs.getString("customer_id"));
            }
        }
        catch(SQLException e) {
            displaySQLErrors(e);
        }
    }
}

```

Drinksbucks store

```
public void loademployees() {
    try {
        employee_id.removeAll();
        rs=statement.executeQuery("select * from employee");
        while(rs.next()) {
            employee_id.add(rs.getString("employee_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void loadorders() {
    try {
        order_id.removeAll();
        rs=statement.executeQuery("select * from torder");
        while(rs.next()) {
            order_id.add(rs.getString("order_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void loaddrinks() {
    try {
        drinks_id.removeAll();
        rs=statement.executeQuery("select * from drinks");
        while(rs.next()) {
            drinks_id.add(rs.getString("drinks_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void loadpayments() {
    try {
        payment_id.removeAll();
        rs=statement.executeQuery("select * from payments");
        while(rs.next()) {
            payment_id.add(rs.getString("payment_id"));
        }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

public void loadsuppliers() {
    try {
        supplier_id.removeAll();
        rs=statement.executeQuery("select * from supplier");
        while(rs.next()) {
            supplier_id.add(rs.getString("supplier_id"));
        }
    }
}
```

```

    }
    }
    catch(SQLException e) {
        displaySQLErrors(e);
    }
}

```

```

public void buildGUI() {

```

```

    insert.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub
            insertButton=new JButton("submit");
            loadcustomers();
            loademployees();
            loadorders();
            loaddrinks();
            loadpayments();
            loadsuppliers();
            txttransaction_id.setText(null);
            // txtcustomer_id.setText(null);
            // txtemployee_id.setText(null);
            // txtorder_id.setText(null);
            // txtdrinks_id.setText(null);
            // txtpayment_id.setText(null);
            // txtsupplier_id.setText(null);
            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();
            //loadpayment();

            p1=new JPanel();

            p1.setLayout(new GridLayout(7,1));

            p1.add(lbltransaction_id);
            p1.add(txttransaction_id);
            p1.add(lblcustomer_id);
            p1.add(customer_id);
            p1.add(lblemployee_id);
            p1.add(employee_id);
            p1.add(lblorder_id);
            p1.add(order_id);
            p1.add(lbldrinks_id);
            p1.add(drinks_id);
            p1.add(lblpayment_id);
            p1.add(payment_id);
            p1.add(lblsupplier_id);
            p1.add(supplier_id);
            p3=new JPanel(new FlowLayout());

```

```
p3.add(insertButton);
//p1.add(txtf1);
p3.setBackground(Color.yellow);
p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
p1.setBackground(Color.pink) ;

p2 = new JPanel(new FlowLayout());

        UserOrderIDList=new List(10);
        loaduserOrderIDs();
        p2.add(UserOrderIDList);
        p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

insertButton.addActionListener(new ActionListener() {
    @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
try {
            String query="INSERT INTO transaction
VALUES("+txttransaction_id.getText()+","+customer_id.getSelectedItem()+","+employee_id.getSelectedItem()
+","+order_id.getSelectedItem()+","+drinks_id.getSelectedItem()+","+payment_id.getSelectedItem()+","+suppli
er_id.getSelectedItem()+")";

            //System.out.println(query);
            int i=statement.executeUpdate(query);
            JOptionPane.showMessageDialog(p,"\ninserted "+i+" rows
succesfully");

            loaduserOrderIDs();
            loadcustomers();
            loademployees();
            loadorders();
            loaddrinks();
            loadpayments();
            loadsuppliers();
        }
catch(SQLException insertException){
            displaySQLErrors(insertException);
        }

    }

});
```

```
}  
});
```

```
delete.addActionListener(new ActionListener() {  
  
    @Override  
    public void actionPerformed(ActionEvent arg0) {  
        // TODO Auto-generated method stub  
        deleteButton=new JButton("delete");  
        txttransaction_id.setText(null);  
        txtcustomer_id.setText(null);  
        txtemployee_id.setText(null);  
        txtorder_id.setText(null);  
        txtdrinks_id.setText(null);  
        txtpayment_id.setText(null);  
        txtsupplier_id.setText(null);  
        p.removeAll();  
        frame.invalidate();  
        frame.validate();  
        frame.repaint();  
  
        p1=new JPanel();  
  
        p1.setLayout(new GridLayout(7,1));  
  
        p1.add(lbltransaction_id);  
        p1.add(txttransaction_id);  
        p1.add(lblcustomer_id);  
        p1.add(txtcustomer_id);  
        p1.add(lblemployee_id);  
        p1.add(txtemployee_id);  
        p1.add(lblorder_id);  
        p1.add(txtorder_id);  
        p1.add(lbldrinks_id);  
        p1.add(txtdrinks_id);  
  
        p1.add(lblpayment_id);  
        p1.add(txtpayment_id);  
        p1.add(lblsupplier_id);  
        p1.add(txtsupplier_id);  
        p3=new JPanel(new FlowLayout());  
        p3.add(deleteButton);  
        //p1.add(txtf1);  
        p3.setBackground(Color.yellow);  
        p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);  
        p1.setBackground(Color.pink) ;  
  
        // p1.setBounds(100,100,400,300);  
  
        p2 = new JPanel(new FlowLayout());  
  
        UserOrderIDList=new List(10);
```


Drinksbucks store

```
loaduserOrderIDs();
p2.add(UserOrderIDList);
p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan);

p. add(p1);p.add(p3);
p. add(p2);

p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();

UserOrderIDList.addItemListener(new ItemListener()
{
    public void itemStateChanged(ItemEvent e)
    {
        try
        {
            rs=statement.executeQuery("select * from
transaction");
            StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

            String p=st.nextToken();
            String q=st.nextToken();
            String r=st.nextToken();
            String s=st.nextToken();
            String t=st.nextToken();
            String u=st.nextToken();
            while (rs.next())
            {
                if
(rs.getString("customer_id").equals(p)      &&      rs.getString("employee_id").equals(q)      &&
rs.getString("order_id").equals(r) && rs.getString("drinks_id").equals(s) && rs.getString("payment_id").equals(t)
&& rs.getString("supplier_id").equals(u))

                break;
            }
            if (!rs.isAfterLast())
            {

                txttransaction_id.setText(rs.getString("transaction_id"));

                txtcustomer_id.setText(rs.getString("customer_id"));

                txtemployee_id.setText(rs.getString("employee_id"));

                txtorder_id.setText(rs.getString("order_id"));

                txtdrinks_id.setText(rs.getString("drinks_id"));

                txtpayment_id.setText(rs.getString("payment_id"));
```

```

txtsupplier_id.setText(rs.getString("supplier_id"));
    }
}
catch (SQLException selectException)
{
    displaySQLErrors(selectException);
}
}
});
p.setLayout(new BorderLayout());

frame.add(p);
frame.setSize(800,800);
frame.validate();
deleteButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        // TODO Auto-generated method stub
try {

        int a=JOptionPane.showConfirmDialog(p,"Are you sure want to
delete:");

        if(a==JOptionPane.YES_OPTION){
            //StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

            String query="DELETE FROM transaction WHERE
transaction_id="+txttransaction_id.getText()+"";

            int i=statement.executeUpdate(query);
JOptionPane.showMessageDialog(p,"\nDeleted "+i+" rows
succesfully");loaduserOrderIDs();
            // loadpayment();
            loadcustomers();
            loademployees();
            loadorders();
            loaddrinks();
            loadpayments();
            loadsuppliers();
        }

    }
catch(SQLException deleteException){
    displaySQLErrors(deleteException);
}

}

});

}
});

/*update.addActionListener(new ActionListener() {

```

```
@Override
public void actionPerformed(ActionEvent arg0) {
    // TODO Auto-generated method stub
    JButton updateButton = new JButton("update");
    txttransaction_id.setText(null);
    txtcustomer_id.setText(null);
    txtemployee_id.setText(null);
    txtorder_id.setText(null);
    txtdrinks_id.setText(null);
    txtpayment_id.setText(null);
    p.removeAll();
    frame.invalidate();
    frame.validate();
    frame.repaint();

    p1=new JPanel();

    p1.setLayout(new GridLayout(5,1));

    p1.add(lbltransaction_id);
    p1.add(txttransaction_id);
    p1.add(lblcustomer_id);
    p1.add(txtcustomer_id);
    p1.add(lblemployee_id);
    p1.add(txtemployee_id);
    p1.add(lblorder_id);
    p1.add(txtorder_id);
    p1.add(lbldrinks_id);
    p1.add(txtdrinks_id);
    p1.add(lblpayment_id);
    p1.add(txtpayment_id);
    p3=new JPanel(new FlowLayout());
    p3.add(updateButton);
    //p1.add(txtf1);
    p3.setBackground(Color.yellow);
    p1.setBounds(115,80,300,250);p3.setBounds(200,350,75,35);
    p1.setBackground(Color.pink) ;

    p2 = new JPanel(new FlowLayout());

    UserOrderIDList=new List(10);
    loaduserOrderIDs();
    p2.add(UserOrderIDList);
    p2.setBounds(450,150,350,180); p2.setBackground(Color.cyan) ;

    p. add(p1);p.add(p3);
    p. add(p2);

    p.setLayout(new BorderLayout());
```

```

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

        UserOrderIDList.addItemListener(new ItemListener()
        {
            public void itemStateChanged(ItemEvent e)
            {
                try
                {
                    rs=statement.executeQuery("select * from
transaction");

                    StringTokenizer st=new
StringTokenizer(UserOrderIDList.getSelectedItem(),"->");

                    String p=st.nextToken();
                    String q=st.nextToken();
                    String r=st.nextToken();
                    String s=st.nextToken();
                    String t=st.nextToken();
                    while (rs.next())
                    {
                        if
(rs.getString("customer_id").equals(p)      &&      rs.getString("employee_id").equals(q)      &&
rs.getString("order_id").equals(r)          &&      rs.getString("drinks_id").equals(s)          &&
rs.getString("payment_id").equals(t))

                                break;
                    }
                    if (!rs.isAfterLast())
                    {

                        txttransaction_id.setText(rs.getString("transaction_id"));

                        txtcustomer_id.setText(rs.getString("customer_id"));

                        txtemployee_id.setText(rs.getString("employee_id"));

                        txtorder_id.setText(rs.getString("order_id"));

                        txtdrinks_id.setText(rs.getString("drinks_id"));

                        txtpayment_id.setText(rs.getString("payment_id"));

                    }
                }
                catch (SQLException selectException)
                {
                    displaySQLErrors(selectException);
                }
            }
        });
        p.setLayout(new BorderLayout());

        frame.add(p);
        frame.setSize(800,800);
        frame.validate();

        updateButton.addActionListener(new ActionListener() {

```

```

        @Override
        public void actionPerformed(ActionEvent e) {
            // TODO Auto-generated method stub
            try {

                loaduserOrderIDs();
                String
                ttlquantity=JOptionPane.showInputDialog(p,"enter the qty");

                txtttlquantity.setText(ttlquantity);

                //int
                a=JOptionPane.showConfirmDialog(p,"Are you sure want to update:");
                //if(a==JOptionPane.YES_OPTION){
                //StringTokenizer      st=new
                StringTokenizer(UserOrderIDList.getSelectedItem(),"->");
                String query="update payments set
                ttlquantity='"+ttlquantity+"'where payment_id="+txtpayment_id.getText();
                int i=statement.executeUpdate(query);

                JOptionPane.showMessageDialog(p,"\nupdated "+i+" rows succesfully");loaduserOrderIDs();
                //loadpayment();
                loadorders();
                loadcustomer();
                }

                catch(SQLException deleteException){
                    displaySQLErrors(deleteException);
                }

            }

        });
    }
    });*/
    view.addActionListener(new ActionListener(){

        @Override
        public void actionPerformed(ActionEvent arg0) {
            // TODO Auto-generated method stub

            p.removeAll();
            frame.invalidate();
            frame.validate();
            frame.repaint();

            Label view1=new Label("transaction view");
            //view1.setAlignment(Label.CENTER);
            Font myFont = new Font("Serif",Font.BOLD,50);
            view1.setFont((myFont));
            viewButton=new JButton("View");
            p1=new JPanel();

```

```

        p2=new JPanel();
        p1.add(view1);
        p2.add(viewButton);p1.setBackground(Color.cyan)
;p2.setBackground(Color.cyan) ;
        p.add(p1);p.add(p2);
        p.setLayout(new FlowLayout());

        p.setBounds(500,800,300,300);
        frame.add(p);
        frame.setSize(800,800);
        frame.validate();
        viewButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                JFrame f;

                JTable j;

                f = new JFrame();

                f.setTitle("transaction Details ");

                DefaultTableModel      model      =      new
DefaultTableModel();

                j = new JTable(model);
                model.addColumn("transaction_id");
                model.addColumn("customer_id");
                model.addColumn("employee_id");

                model.addColumn("order_id");
                model.addColumn("drinks_id");
                model.addColumn("payment_id");

                try {

                    rs=statement.executeQuery("select * from transaction");

                    while(rs.next()) {
                        model.addRow(new
Object[] {rs.getString("transaction_id"),
rs.getString("customer_id"),rs.getString("employee_id"),rs.getString("order_id"),rs.getS
tring("payment_id")});
                    }
                }
                catch(SQLException viewException) {

                    displaySQLErrors(viewException);

                }
                j.setEnabled(false);
                j.setBounds(30, 40, 180, 150);

```

```
        JScrollPane sp = new JScrollPane(j);
        f.add(sp);

        f.setSize(500, 400);

        f.setVisible(true);

    }

});

    }

});

    }

}
```

DBUI:

```
package DrinkB;
import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridLayout;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;
```

```
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
```

```
public class DBUI extends JFrame{
    /**
     *
     */
    private static final long serialVersionUID = 1L;
```

```

private JMenuBar mnu;

private JMenu mnuemployee;
private JMenu mnucustomer;
private JMenu mnusupplier;
private JMenu mnudrinks;
private JMenu mnutorder;
private JMenu mnupayments;
private JMenu mnutransaction;

private JMenuItem insert1,update1,delete1,view1;
private JMenuItem insert2,update2,delete2,view2;
private JMenuItem insert3,update3,delete3,view3;
private JMenuItem insert4,update4,delete4,view4;
private JMenuItem insert5,update5,delete5,view5;
private JMenuItem insert6,update6,delete6,view6;
private JMenuItem insert7,delete7,view7;

private JLabel labelName;

private static JPanel p0,p1;

void initialize() {
    mnu=new JMenuBar();

    mnuemployee= new JMenu("Employee");
    mnucustomer= new JMenu("Supplier");
    mnusupplier= new JMenu("Customer");
    mnudrinks= new JMenu("Drink");
    mnutorder= new JMenu("Orders");
    mnupayments= new JMenu("Payments");
    mnutransaction= new JMenu("Transactions");
    labelName=new JLabel("DRINK BUCKS");
    p1=new JPanel();
    p0=new JPanel();
    insert1=new JMenuItem("Insert");
    update1=new JMenuItem("Update");
    delete1=new JMenuItem("Delete");
    view1=new JMenuItem("View");
    insert2=new JMenuItem("Insert");
    update2=new JMenuItem("Update");
    delete2=new JMenuItem("Delete");
    view2=new JMenuItem("View");
    insert3=new JMenuItem("Insert");
    update3=new JMenuItem("Update");
    delete3=new JMenuItem("Delete");
    view3=new JMenuItem("View");
    insert4=new JMenuItem("Insert");
    update4=new JMenuItem("Update");
    delete4=new JMenuItem("Delete");
    view4=new JMenuItem("View");
    insert5=new JMenuItem("Insert");
    update5=new JMenuItem("Update");
    delete5=new JMenuItem("Delete");

```



```
view5=new JMenuItem("View");
insert6=new JMenuItem("Insert");
update6=new JMenuItem("Update");
delete6=new JMenuItem("Delete");
view6=new JMenuItem("View");
insert7=new JMenuItem("Insert");
// update7=new JMenuItem("Update");
delete7=new JMenuItem("Delete");
view7=new JMenuItem("View");
insert1=new JMenuItem("Insert");
update1=new JMenuItem("Update");
delete1=new JMenuItem("Delete");
view1=new JMenuItem("View");
}
void addComponentsToFrame() {
    mnuemployee.add(insert1);
    mnuemployee.add(delete1);
    mnuemployee.add(update1);
    mnuemployee.add(view1);
    mnucustomer.add(insert2);
    mnucustomer.add(delete2);
    mnucustomer.add(update2);
    mnucustomer.add(view2);
    mnusupplier.add(insert3);
    mnusupplier.add(delete3);
    mnusupplier.add(update3);
    mnusupplier.add(view3);
    mnudrinks.add(insert4);
    mnudrinks.add(delete4);
    mnudrinks.add(update4);
    mnudrinks.add(view4);
    mnutorder.add(insert5);
    mnutorder.add(delete5);
    mnutorder.add(update5);
    mnutorder.add(view5);
    mnupayments.add(insert6);
    mnupayments.add(delete6);
    mnupayments.add(update6);
    mnupayments.add(view6);
    mnutransaction.add(insert7);
    mnutransaction.add(delete7);
    // mnutransaction.add(update7);
    mnutransaction.add(view7);
    mnu.add(mnuemployee);
    mnu.add(mnucustomer);
    mnu.add(mnusupplier);
    mnu.add(mnudrinks);
    mnu.add(mnutorder);
    mnu.add(mnupayments);
    mnu.add(mnutransaction);
    setJMenuBar(mnu);
    p1.add(labelName);p1.setAlignmentY(CENTER_ALIGNMENT);
    p1.setBounds(500,500,800,100);
    p0.add(p1);
}
```

```

        p0.setBackground(Color.CYAN);
        add(p0);
    }
    void closeWindow(){
        try {
            int a=JOptionPane.showConfirmDialog(this,"Are you sure want to Quit DRINK
BUCKS:");
            if(a==JOptionPane.YES_OPTION){
                JOptionPane.showMessageDialog(this,
                    "Thank you!\nExiting DRINK BUCKS","Quit",
                    JOptionPane.WARNING_MESSAGE);
                System.exit(0);
            }
            else if (a== JOptionPane.NO_OPTION) {
                setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
            }
            else if (a== JOptionPane.CANCEL_OPTION) {
                setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE);
            }
        }
        catch(Exception e) {
            System.out.println(e);
        }
    }
    void register() {
        Emp log=new Emp(p0,DBUI.this,insert1,delete1,update1,view1);
        log.buildGUI();
        Cust user=new Cust(p0,DBUI.this,insert3,delete3,update3,view3);
        user.buildGUI();
        Supp user1=new Supp(p0,DBUI.this,insert2,delete2,update2,view2);
        user1.buildGUI();
        Drinks d=new Drinks(p0,DBUI.this,insert4,delete4,update4,view4);
        d.buildGUI();
        Order u=new Order(p0,DBUI.this,insert5,delete5,update5,view5);
        u.buildGUI();
        pay p=new pay(p0,DBUI.this,insert6,delete6,update6,view6);
        p.buildGUI();
        trans u_orders=new trans(p0,DBUI.this,insert7,delete7,view7);
        u_orders.buildGUI();
        addWindowListener(new WindowAdapter(){
            public void windowClosing(WindowEvent we)
            {
                closeWindow();
            }
        });
    }
}
public DBUI() {
    initialize();
    addComponentsToFrame();
    register();
    // setBackground(Color.RED);
    setTitle("DRINKSBUCKS");
    setSize(800,800);
    setVisible(true);
}

```

Drinksbucks store

```
}
```

MAIN:

```
package DrinkB;  
public class main {  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        new DBUI();  
    }  
  
}
```

GITHUB LINK:

<https://github.com/sreeya-c/DRINKS-BUCKS>

FOLDER STRUCTURE:

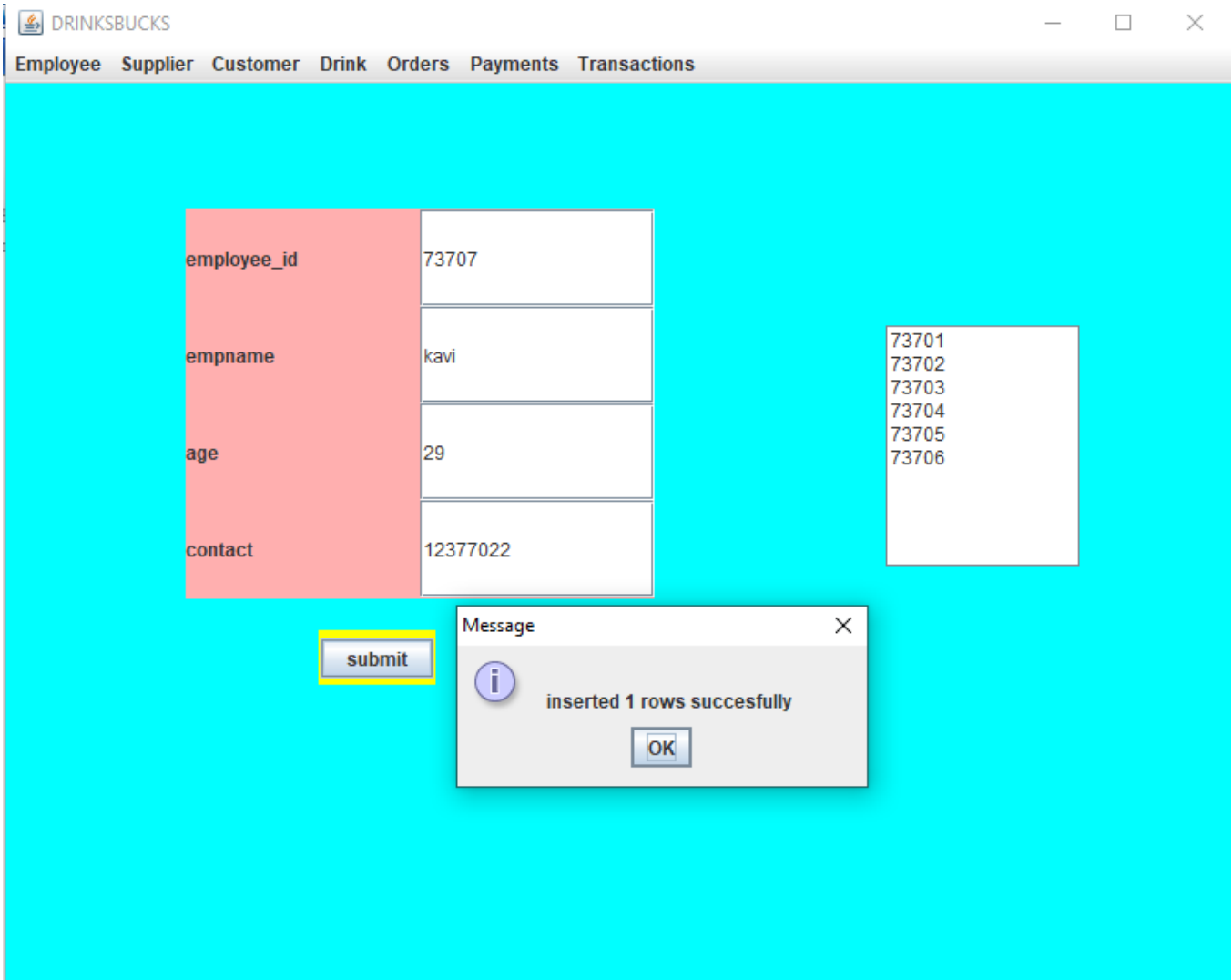
MY pc > Documents > JAVADBMS > drinksbuck				
	Name	Date modified	Type	Size
	.settings	13-06-2021 19:46	File folder	
	bin	19-06-2021 12:26	File folder	
	src	14-06-2021 14:51	File folder	
	.classpath	13-06-2021 19:47	CLASSPATH File	1 KB
	.project	13-06-2021 19:46	PROJECT File	1 KB

MY pc > Documents > JAVADBMS > drinksbuck > src				
	Name	Date modified	Type	Size
	DrinkB	15-06-2021 19:47	File folder	

MY pc > Documents > JAVADBMS > drinksbuck > src > DrinkB				
	Name	Date modified	Type	Size
	Cust	17-06-2021 17:19	JAVA File	14 KB
	DBUI	17-06-2021 15:40	JAVA File	6 KB
	Drinks	17-06-2021 19:29	JAVA File	14 KB
	Emp	17-06-2021 17:03	JAVA File	14 KB
	main	14-06-2021 17:16	JAVA File	1 KB
	Order	17-06-2021 16:29	JAVA File	16 KB
	pay	17-06-2021 20:32	JAVA File	16 KB
	Supp	17-06-2021 17:03	JAVA File	15 KB
	trans	17-06-2021 13:50	JAVA File	19 KB

TESTING

OUTPUT SCREENSHOTS:



employee_id	73704
empname	warner
age	56
contact	12345678

73701
73702
73703
73704
73705
73706
73707

delete

Message



SQLException: ORA-02292: integrity constraint (IT19737110.SYS_C007004) violated - child record found

SQLState: 23000

VendorError: 2292

OK

DRINKSBUCKS

Employee Supplier Customer Drink Orders Payments Transactions

employee_id

73707

empname

kavi

age

29

contact

12377022

73701

73702

73703

73704

73705

73706

73707

modify

Input

?

enter the age

19

OK

Cancel

DRINKSBUCKS

Employee

Supplier

Customer

Drink

Orders

Payments

Transactions

employee_id	73707
empname	kavi
age	19
contact	12377022

73701

73702

73703

73704

73705

73706

73707

modify

Message

i

updated 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

employees view

View

Medicine details

employee id	empname	age	contact
73701	garg	35	89765432
73702	priyam	42	89907654
73703	bhuvneshwar	31	56789053
73704	warner	56	12345678
73705	kane	28	56431096
73706	sangavi	23	56312
73707	kavi	19	12377022

DRINKSBUCKS

Employee

Supplier

Customer

Drink

Orders

Payments

Transactions

supplier id	160207
supplier name	shreyas
contact no	0405321
mail address	shreyas@gmail.com

submit

Message

i

inserted 1 rows succesfully

OK

160201

160202

160203

160204

160205

1236

160206

DRINKSBUCKS

Employee

Supplier

Customer

Drink

Orders

Payments

Transactions

supplier id	160201
supplier name	raghu
contact no	908764
mail address	raghu@gmail.com

160201

160202

160203

160204

160205

1236

160207

160206

delete

Select an Option

?

Are you sure want to delete:

Yes

No

Cancel

Employee Supplier Customer Drink Orders Payments Transactions

supplier id	160201
supplier name	raghu
contact no	908764
mail address	raghu@gmail.com

160201

160202

160203

160204

160205

1236

160207

160206

delete

Message

×



SQLException: ORA-02292: integrity constraint (IT19737110.SYS_C007009) violated - child record found

SQLState: 23000

VendorError: 2292

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

supplier id

160206

supplier name

venkatesh

contact no

23566

mail address

venkatesh@gmail.com

160201

160202

160203

160204

160205

1236

160207

160206

modify

Input

?

enter the contactno

OK


Cancel

supplier id	160206
supplier name	venkatesh
contact no	23577
mail address	venkatesh@gmail.com

160201
160202
160203
160204
160205
1236
160207
160206

modify

Message

 updated 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

suppliers view

View

orderDetails

supplier id	supplier name	contact no	mail address
160201	raghu	908764	raghu@gmail.com
160202	ramu	8911234	ramu@gmail.com
160203	janaki	908744	janaki@gmail.com
160204	sonakshibose	897666	sonakshi@gmail.com
160205	devdixit	667788	devdixit@gmail.com
1236	mila	16789	mila@gmail.com
160207	shreyas	405321	shreyas@gmail.com
160206	venkatesh	23577	venkatesh@gmai.com

DRINKSBUCKS

Employee

Supplier

Customer

Drink

Orders

Payments

Transactions

customer id

1907

customer name

camille

gender

female

contactno

01932456

submit

1901

1902

1903

1904

1905

1893

1906

512

Message

i

inserted 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

customer id

1907

customer name

camille

gender

female

contactno

1932456

1901

1902

1903

1904

1905

1893

1906

512

1907

modify

Input

?

enter the contactno

1932457

OK

Cancel

DRINKSBUCKS

Employee

Supplier

Customer

Drink

Orders

Payments

Transactions

customer id	1907
customer name	camille
gender	female
contactno	1932457

1901

1902

1903

1904

1905

1893

1906

512

1907

modify

Message

i

updated 1 rows succesfully

OK

DRINKSBUCKS

Employee Supplier Customer Drink Orders Payments Transactions

customer id

1904

customer name

devilliars

gender

male

contactno

908765

1901

1902

1903

1904

1905

1893

1906

512

1907

delete

Message

i

SQLException: ORA-02292: integrity constraint (IT19737110.SYS_C006998) violated - child record found

SQLState: 23000

VendorError: 2292

OK

customer view

[View](#)

Customer Details

customer id	customer name	gender	contactno
1901	shub	male	89780098
1902	goyal	female	123908
1903	priya	female	675489
1904	devilliars	male	908765
1905	sita	female	456789
1893	kia	female	123456
1906	manu	male	6770227100
512	priyamk	male	3421098
1907	camille	female	1932457

Drinksbucks store

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

Drinks id

7

Drinks Name

coffeechoco

quantity

15

price

20

submit

1

2

3

4

5

1782

34

Message

i

inserted 1 rows succesfully

OK

Drinks id	7
Drinks Name	coffeechoco
quantity	15
price	20

[modify](#)

1
2
3
4
5
1782
7
34

Input



enter the new price

OK

Cancel

DRINKSBUCKS

Employee Supplier Customer Drink Orders Payments Transactions

Drinks id

7

Drinks Name

coffeechoco

quantity

15

price

25

1

2

3

4

5

1782

7

34

modify

Message

i


updated 1 rows succesfully

OK

Employee Supplier Customer Drink Orders Payments Transactions

Drinks id	1
Drinks Name	capuccino
quantity	20
price	500

1
2
3
4
5
1782
7
34

Message		×
	SQLException: ORA-02292: integrity constraint (IT19737110.SYS_C006999) violated - child record found	
SQLState: 23000		
VendorError: 2292		
<input type="button" value="OK"/>		

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

Drinks view

View

Drinks details

Drinks id	Drinks Name	quantity	price
1	capuccino	20	500
2	mangojuice	80	1000
3	chocolava milkshake	50	2000
4	dietcoke	40	5000
5	sodalemon	60	200
1782	frootijuice	12	12
7	coffeechoco	15	25
34	parle	20	90

DRINKSBUCKS

Employee

Supplier

Customer

Drink

Orders

Payments

Transactions

Order ID	7
Customer ID	1905
Drinks ID	5
quantity	12
priceperquantity	20

submit

Message

i

inserted 1 rows succesfully

OK

1->1901->1

2->1902->2

3->1903->3

4->1904->4

5->1905->5

1533->512->34

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

Order ID	7
Customer ID	1905
Drinks ID	5
quantity	15
priceperquantity	20

modify

Message

i

updated 1 rows succesfully

OK

1->1901->1
2->1902->2
3->1903->3
4->1904->4
5->1905->5
1533->512->34
7->1905->5

Order ID	1
Customer ID	1901
Drinks ID	1
quantity	2
priceperquantity	50

1->1901->1
2->1902->2
3->1903->3
4->1904->4
5->1905->5
1533->512->34
7->1905->5

Message



SQLException: ORA-02292: integrity constraint (IT19737110.SYS_C007001) violated - child record found

SQLState: 23000

VendorError: 2292

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

ORDER view

View

order Details

Order ID	Customer ID	Drinks ID	quantity	priceperquantity
1	1901	1	2	50
2	1902	2	3	800
3	1903	3	1	40
4	1904	4	5	625
5	1905	5	2	40
1533	512	34	13	13
7	1905	5	15	20

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

payment_id

506

order_id

7

▼

customer_id

1905

▼

ttlquantity

200

ttlprice

500

submit

501->1->1901

502->2->1902

503->3->1903

504->4->1904

505->5->1905

1863->1533->512

Message

✕

i

inserted 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

payment_id

1863

order_id

1533

customer_id

512

ttlquantity

12

ttlprice

12

501->1->1901

502->2->1902

503->3->1903

504->4->1904

505->5->1905

506->7->1905

1863->1533->512

modify

Input

?

enter the qty

20

OK

Cancel

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

payment_id	1863
order_id	1533
customer_id	512
ttlquantity	20
ttlprice	12

501->1->1901
502->2->1902
503->3->1903
504->4->1904
505->5->1905
506->7->1905
1863->1533->512

modify

Message

i

updated 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

payment_id	502
order_id	2
customer_id	1902
ttlquantity	3
ttlprice	800

delete

Select an Option

?

Are you sure want to delete:

Yes

No

Cancel

501->1->1901

502->2->1902

503->3->1903

504->4->1904

505->5->1905

506->7->1905

1863->1533->512

payment_id	502
order_id	2
customer_id	1902
ttlquantity	3
ttlprice	800

501->1->1901
502->2->1902
503->3->1903
504->4->1904
505->5->1905
506->7->1905
1863->1533->512

delete

Message



SQLException: ORA-02292: integrity constraint (IT19737110.SYS_C007008) violated - child record found

SQLState: 23000

VendorError: 2292

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

payments view

View

payments Details

payment_id	order_id	customer_id	ttlquantity	ttlprice
501	1	1901	2	50
502	2	1902	3	800
503	3	1903	1	40
504	4	1904	5	625
505	5	1905	2	40
506	7	1905	200	500
1863	1533	512	20	12

DRINKSBUCKS

ployee Supplier Customer Drink Orders Payments Transactions

transaction_id	1252
customer_id	1905
employee_id	73707
order_id	7
drinks_id	5
payment_id	506
supplier_id	160206

submit

Message

i

inserted 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

transaction_id	1252
customer_id	1905
employee_id	73707
order_id	7
drinks_id	5
payment_id	506
supplier_id	160206

delete

Select an Option

?

Are you sure want to delete:

Yes

No

Cancel

1901->73701->1->1->501->160201

1902->73702->2->2->502->160202

1903->73703->3->3->503->160203

1904->73704->4->4->504->160204

1905->73705->5->5->505->160205

1905->73707->7->5->506->160206

transaction_id	1252
customer_id	1905
employee_id	73707
order_id	7
drinks_id	5
payment_id	506
supplier_id	160206

1901->73701->1->1->501->160201
1902->73702->2->2->502->160202
1903->73703->3->3->503->160203
1904->73704->4->4->504->160204
1905->73705->5->5->505->160205
1905->73707->7->5->506->160206

Message



Deleted 1 rows succesfully

OK

DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

transaction view


View

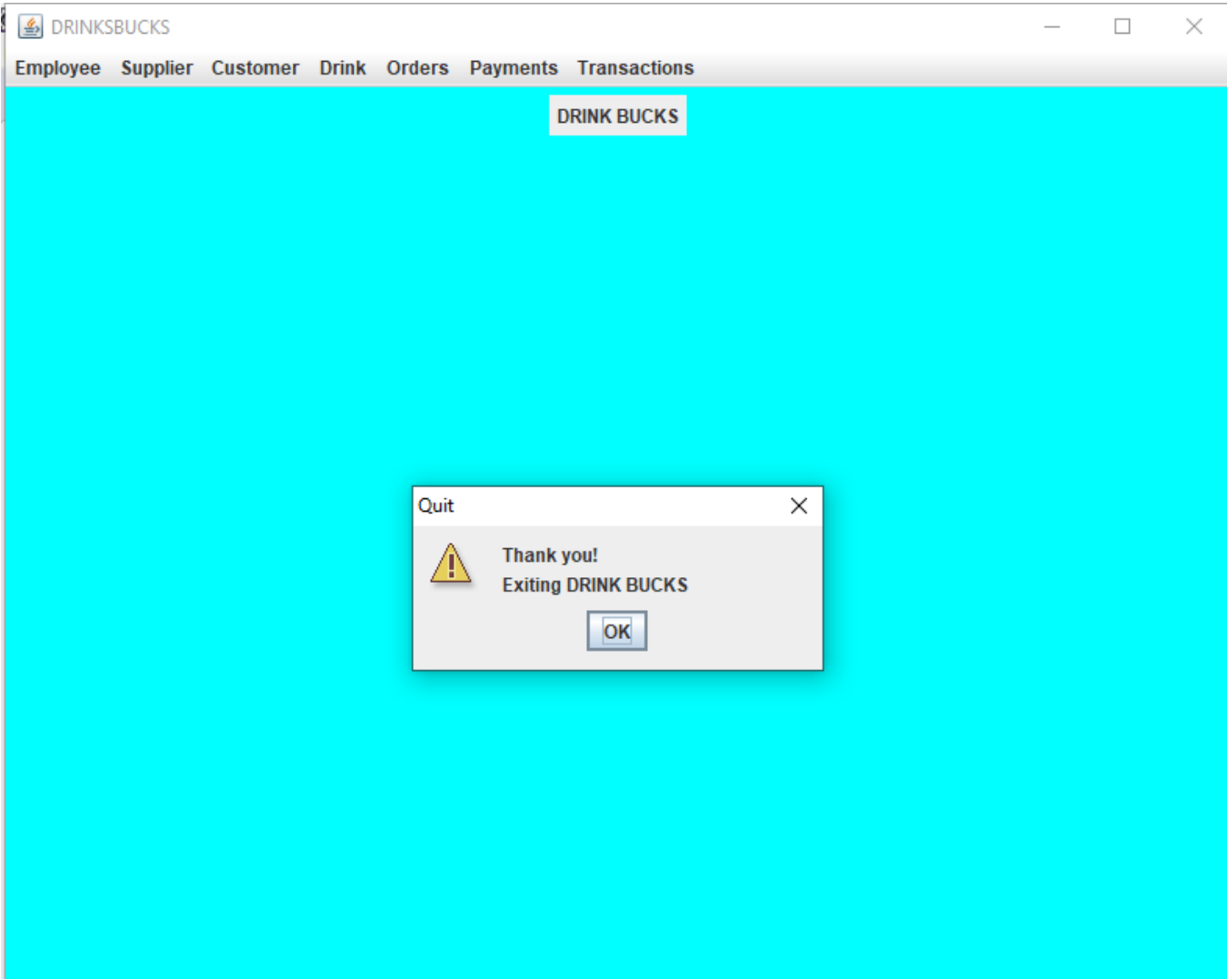
transaction Details

transaction_id	customer_id	employee_id	order_id	drinks_id	payment_id
202101	1901	73701	1	1	501
202102	1902	73702	2	2	502
202103	1903	73703	3	3	503
202104	1904	73704	4	4	504
202105	1905	73705	5	5	505

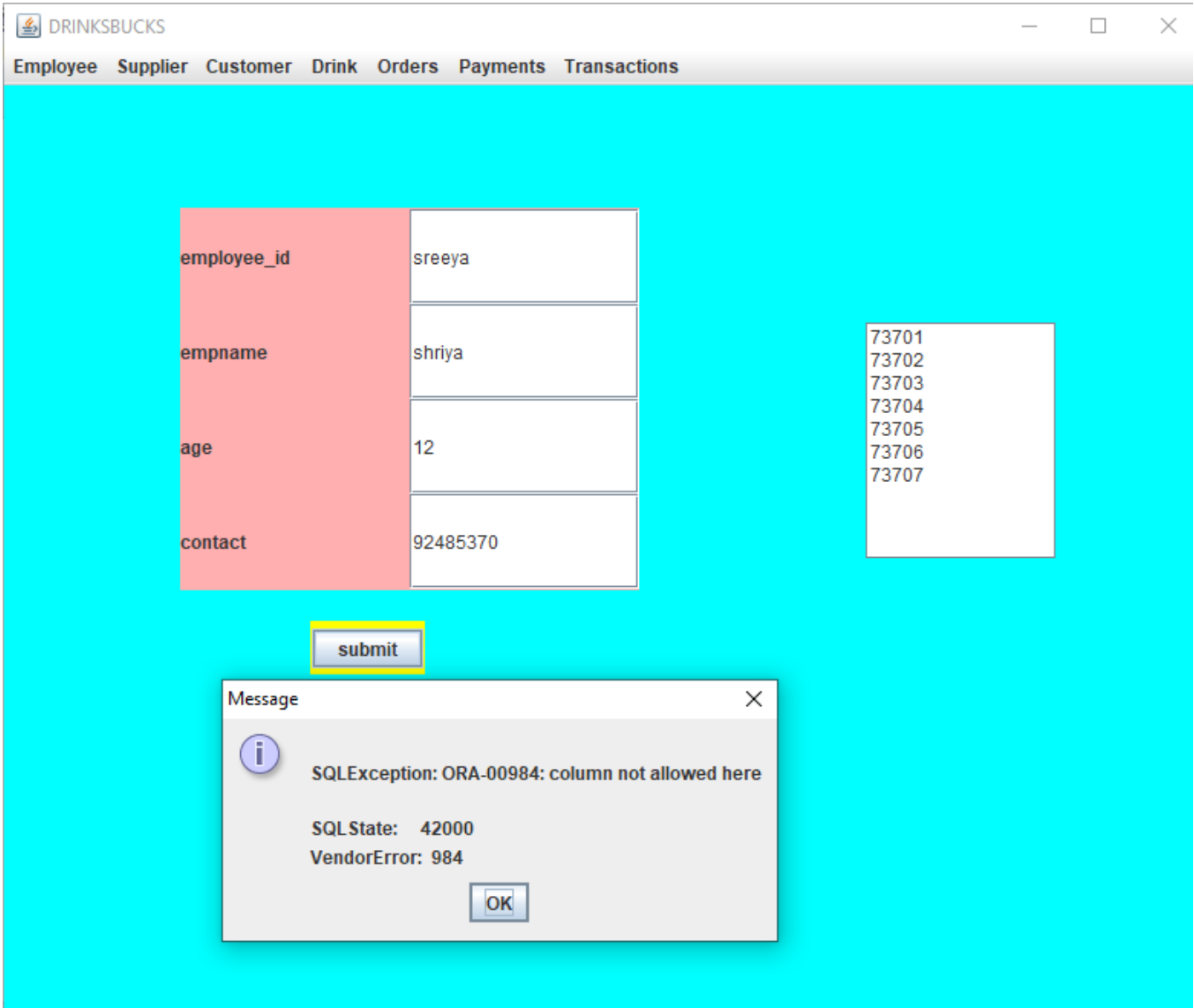
DRINK BUCKS

Select an Option ×

 Are you sure want to Quit DRINK BUCKS:



ADDITIONAL TESTCASES:



DRINKSBUCKS

EmployeeSupplierCustomerDrinkOrdersPaymentsTransactions

supplier id

abhinaya

supplier name

ashritha

contact no

04028971

mail address

chemudud@gmail.com

160201

160202

160203

160204

160205

1236

160207

160206

submit

Message

i

SQLException: ORA-00984: column not allowed here

SQLState: 42000

VendorError: 984

OK

The data entered in the above form is updated in the “employee” table of theOracle database 11g as:

```
SQL> select * from employee;
```

EMPLOYEE_ID	EMPNAME	AGE	CONTACT
73701	gang	35	89765432
73702	priyam	42	89907654
73703	bhuvneshwar	31	56789053
73704	warner	56	12345678
73705	kane	28	56431096
73706	sangavi	23	56312
73707	kavi	19	12377022

7 rows selected.

The data entered in the above form is updated in the “supplier” table of theOracle database 11g as:

```
SQL> select * from supplier;
```

SUPPLIER_ID	SUPPLIER_NAME	CONTACT_NO	MAILADDRESS
160201	raghu	908764	raghu@gmail.com
160202	ramu	8911234	ramu@gmail.com
160203	janaki	908744	janaki@gmail.com
160204	sonakshibose	897666	sonakshi@gmail.com
160205	devdixit	667788	devdixit@gmail.com
1236	mila	16789	mila@gmail.com
160207	shreyas	405321	shreyas@gmail.com
160206	venkatesh	23577	venkatesh@gmai.com

8 rows selected.

The data entered in the above form is updated in the “customer” table of theOracle database 11g as:

```
SQL> select * from customer;
```

CUSTOMER_ID	CUSTOMER_NAME	GENDER	CONTACTNO
1901	shub	male	89780098
1902	goyal	female	123908
1903	priya	female	675489
1904	devilliars	male	908765
1905	sita	female	456789
1893	kia	female	123456
1906	manu	male	6770227100
512	priyamk	male	3421098
1907	camille	female	1932457

```
9 rows selected.
```

The data entered in the above form is updated in the “drinks” table of the Oracle database 11g as:

```
SQL> select * from drinks;
```

DRINKS_ID	DRINKS_NAME	QUANTITY	PRICE
1	capuccino	20	500
2	mangojuice	80	1000
3	chocolava milkshake	50	2000
4	dietcoke	40	5000
5	sodalemon	60	200
1782	frootijuice	12	12
7	coffeechoco	15	25
34	parle	20	90

```
8 rows selected.
```

The data entered in the above form is updated in the “torder” table of the Oracle database 11g as:

```
SQL> select * from torder;
```

ORDER_ID	CUSTOMER_ID	DRINKS_ID	QTY	PRICE_PERQTY
1	1901	1	2	50
2	1902	2	3	800
3	1903	3	1	40
4	1904	4	5	625
5	1905	5	2	40
1533	512	34	13	13
7	1905	5	15	20

```
7 rows selected.
```

The data entered in the above form is updated in the “payments” table of theOracle database 11g as:

```
SQL> select * from payments;
```

PAYMENT_ID	ORDER_ID	CUSTOMER_ID	TTLQUANTITY	TTLPRICE
501	1	1901	2	50
502	2	1902	3	800
503	3	1903	1	40
504	4	1904	5	625
505	5	1905	2	40
506	7	1905	200	500
1863	1533	512	20	12

```
7 rows selected.
```

The data entered in the above form is updated in the “transaction” table of theOracle database 11g as:

```
SQL> select * from transaction;
```

TRANSACTION_ID	CUSTOMER_ID	EMPLOYEE_ID	ORDER_ID	DRINKS_ID	PAYMENT_ID
202101	1901	73701	1	1	501
160201					
202102	1902	73702	2	2	502
160202					
202103	1903	73703	3	3	503
160203					

TRANSACTION_ID	CUSTOMER_ID	EMPLOYEE_ID	ORDER_ID	DRINKS_ID	PAYMENT_ID
202104	1904	73704	4	4	504
160204					
202105	1905	73705	5	5	505
160205					

```
SQL>
```

RESULTS:

I had successfully completed PROJECT on “DRINKSBUCKS”.

DISCUSSION AND FUTURE WORK

So far this project has helped us in easing the customers life. The Customers order eventually is the crucial aspect here as it is the main parameter for which the project is designed for. It helps you to select your own drink without wasting your precious time and involves less human intervention.

In future the most probable aspects could be improvising payment options like adding paypal, paytm, Gift Cards etc and provide the customer with a visual graphical order status bar. Also it can be refined by adding a feature of sending an order ready notification to customer. Lastly, there could be one more additional feature added like providing deals and promotional offer details to home page.

CONCLUSION:

Thus, a Java SWING based DRINKSBUCKS is created which is connected to the Oracle 11g database. Therefore, all the entries and details are directly updated on their respective tables created in the database.

REFERENCES:

- <https://docs.oracle.com/javase/7/docs/api/>
- <https://www.javatpoint.com/dbms-tutorial>
- https://en.wikipedia.org/wiki/Online_pharmacy

