A Database for the Management of a Restaurant System

Project for the class Database Systems II
in the Summer Semester 2021

Contributed by

(Signature)
Enea Risto
(02rien1mst@hft-stuttgart.de)

(Signature)
Philipp Kurrle
(11kuph1mst@hft-stuttgart.de)

(Signature)
Valliammai Radhakrishnan
(11rava1mst@hft-stuttgart.de)

1. Abstract

This documentation outlines the creation process of a database maintenance system that supports several functionalities that tackle real-world problems. This is realized by the implementation of triggers and stored procedures. Due to time restrictions, only basic features have been implemented. The motivation is explored, alongside functionalities and structure.

2. Introduction

Ever since the commercial database integration by Oracle in the late 1970s, it has become increasingly difficult to stumble upon a larger program that does not make use out of the functionality the database maintenance system provides. Nowadays, database integration is a minimum requirement in the vast majority of projects. Though born and raised within the confinement of the IT industry, the usefulness of the data management structures made its way into other aspects of life. One of which is gastronomy. It is a common misperception that it does not take a substantial amount of skill to work at a restaurant, something that people who have or are working in the industry will tell you the contrary. Some say that the stress they experience in their shift during work hours, cannot be compared to anything else. Considering the above, it is no surprise that the functionality of database systems would come in handy.

3. Motivation

As mentioned above, the motivation comes from the need to deal with the complexity caused by a large influx of customers during a short period of time. The focal points can be summarized in the following manner:

1. Managing orders

A lot of things happen parallelly behind the scenes at a restaurant. Once the customers are seated, they are expected to receive timely attendance from a server who will take their order. These orders typically consist of an appetizer, the main course and the dessert. It is in the best interest of the business, that their orders do not only arrive in a short time, but that they are also reasonably spaced out from one another. This is why it makes the most sense to automate the process of order scheduling. Instead of a piece of paper, the servers would punch in the order into the database management system, where it would be accompanied with a timestamp.

2. Managing seating

One of the most important aspects of your experience at a restaurant is also the ambiente. Though very hard to manage during the busiest hours, a good host will attempt to give all the customers enough personal space so that they can feel like they can converse freely. This

is often overlooked, but a skilled manager will keep this in the back of their head. Obviously, depending on the scale of the restaurant, the difficulty of this task can vary. While a small restaurant can do just fine with a simple look at the open tables, bigger places find it very difficult to keep up. This is why it is a great aid to have the seated tables recorded into the database system in order to help decide where and if the next guests should be seated.

3. Managing takeaways

Most restaurants offer the option of delivery so that they can maximize their income, as businesses usually aim to do. Of course, there is a clear trade-off in the implementation that comes in the form of complexity. Taking into account the two management aspects explored above and adding delivery on top, the difficulty of avoiding mistakes is put to perspective. You definitely do not want to get the order wrong, deliver it too late or even deliver it to the wrong address. This is why it is important to document everything.
These are only a few examples to illustrate how a database system can be useful in the field of gastronomy.

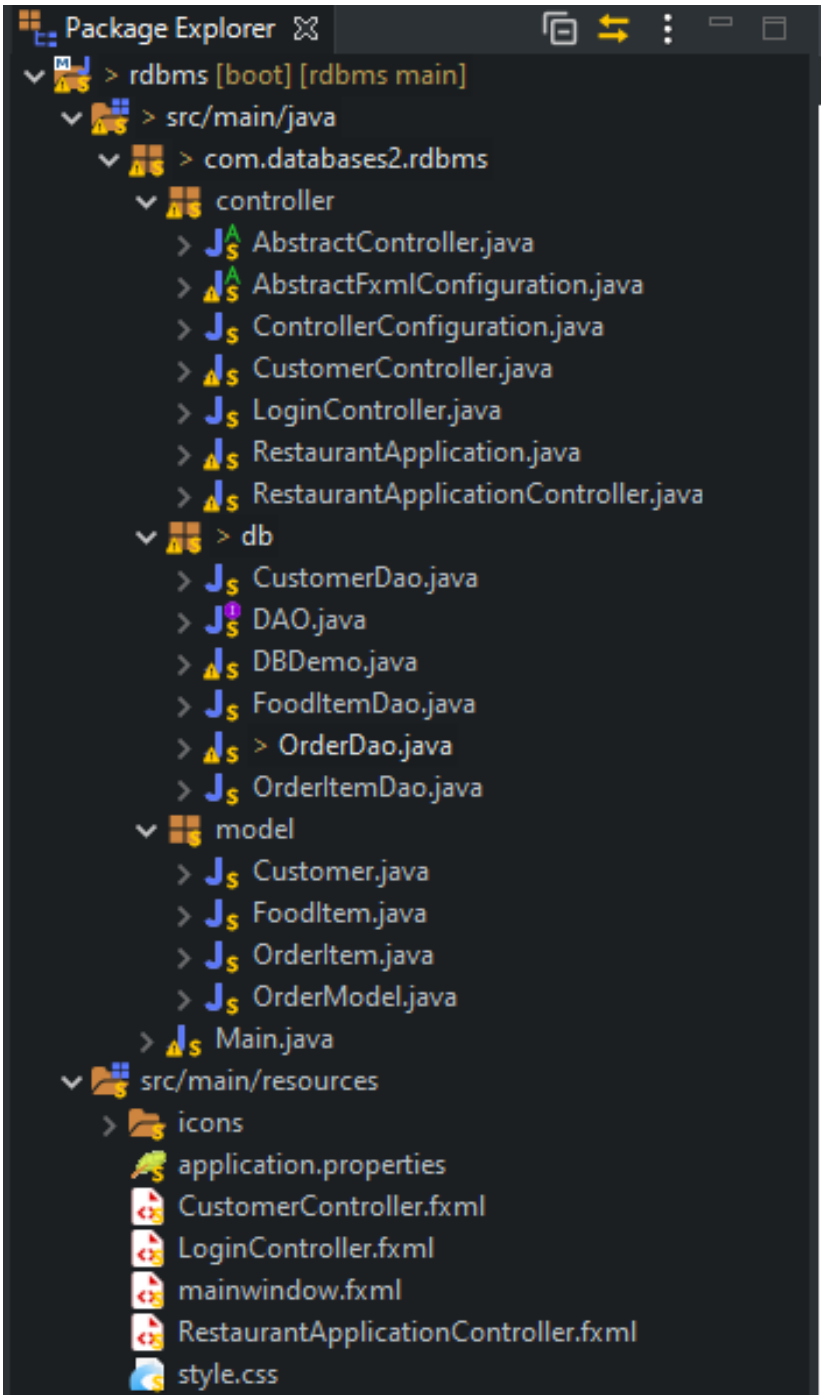
4. Methodology - System Platform

For our Project we decided to use technologies we already worked in the past with. A programming language all group members were familiar with is Java. Therefore we chose Java in combination with JavaFX as UI Framework and SpringBoot as application Framework. To run the source code eclipse with the SpringBoot Plugin is needed. Run the Main class as “SpringBoot Project.” It is a maven project and therefore all dependencies configured in the pom.xml will be pulled automatically. For the back-end mysql needs to be installed and the tables need to be created. The corresponding SQL statements can be found in chapter 4.

Software	Version	Use
SpringBoot Starter	-	Basic Spring Boot Project
Spring Boot Framework	2.4.5	The Application Framework we used
JFoenix	8.0.10	Material UI Library
MySql	5.1.6	Database

5. Structure - Data

We used the Model-View-Controller Pattern to structure the project. The views are written in the user interface markup language FXML. All models can be found in the models package and all the controllers can be found in the controller package. Every View has its own models and controllers. All classes for the Backend are in the db package. Every model has a corresponding DAO class. The DBDemo and abstract DAO class handle the connection to the database.



6. Database Construction

This section documents the queries used to create the database management system.

```
CREATE TABLE CUSTOMER (CUSTOMERID INTEGER NOT NULL PRIMARY KEY,NAME
VARCHAR(255) NOT NULL, PHONE VARCHAR(20),FEEDBACK VARCHAR(255));

INSERT INTO CUSTOMER VALUES (101,'RICKY',+4926105678762,'Great experience');
INSERT INTO CUSTOMER VALUES (102,'OLIVIA',+4917620456195,'Pleasant interiors
and good service');
INSERT INTO CUSTOMER VALUES (103,'TONY',+4920187789124,'Food was tasty, nice
staff, & awesome people');
INSERT INTO CUSTOMER VALUES (104,'BOB',+4976546286209,'Pricy!');
INSERT INTO CUSTOMER VALUES (105,'WILLIUM',+4976542789137,'Very good pizzas!
Good service');
INSERT INTO CUSTOMER VALUES (106,'ANTONY',+4917628977436,'Good Italian food,
friendly staff, good price for quality');

-------------------------------------------------------------
CREATE TABLE FACULTY (ID INTEGER NOT NULL PRIMARY KEY,NAME VARCHAR(255) NOT
NULL,POSITION VARCHAR(255),YEAROFJOINING INT,SALARY FLOAT,BONUS FLOAT);

INSERT INTO FACULTY VALUES (10,'ROMA
SOUZA','RECEPTIONIST',2015,1200.00,0.00);
INSERT INTO FACULTY VALUES (11,'LEO ANDERSON','CASHIER',2010,1600.00,0);
INSERT INTO FACULTY VALUES (13,'MAYA
NICOLAUS','WAITRESS',2017,990.00,100.00);
INSERT INTO FACULTY VALUES (12,'MIRELA HANS','WAITRESS',2014,1055.00,150.00);
INSERT INTO FACULTY VALUES (17,'JULIA PETER','WAITRESS',2017,990.00,100.00);
INSERT INTO FACULTY VALUES (16,'EMILY MARTIN','DISHWASHER',2018,990.00,0.0);
INSERT INTO FACULTY VALUES (14,'LUCY THOMAS','DISHWASHER',2016,990.00,0.0);
INSERT INTO FACULTY VALUES (15,'GEORGE ENN','COOK',2015,1800.00,200.0);
INSERT INTO FACULTY VALUES (18,'BONITA JAMES','COOK',2017,1800.00,150.0);
INSERT INTO FACULTY VALUES (19,'AKSA KAMAL','WAITRESS',2020,1055.00,0.00);
INSERT INTO FACULTY VALUES (20,'VANESSA
BENJAMIN','RECEPTIONIST',2020,990.00,0.0);

-------------------------------------------------------------
CREATE TABLE FOOD_MENU (ITEM_NAME VARCHAR(255),ITEM_ID INTEGER NOT NULL
PRIMARY KEY,PRICE FLOAT);

INSERT INTO FOOD_MENU VALUES ('MARGHERITA PIZZA',111,10.30);
INSERT INTO FOOD_MENU VALUES ('BOLOGNA PIZZA',112,12.45);
INSERT INTO FOOD_MENU VALUES ('VEGETARIAN PIZZA',113,10.30);
INSERT INTO FOOD_MENU VALUES ('SICILIAN PIZZA',114,13.50);
INSERT INTO FOOD_MENU VALUES ('QUATRO FORMAGGI PIZZA',115,09.70);
INSERT INTO FOOD_MENU VALUES ('LASAGNE',116,08.15);
INSERT INTO FOOD_MENU VALUES ('RICOTTA CANNELLONI',117,09.30);
INSERT INTO FOOD_MENU VALUES ('CHICKEN PARMIGIANA',118,12.15);
INSERT INTO FOOD_MENU VALUES ('MOZZARELLA STICKS',119,05.78);
INSERT INTO FOOD_MENU VALUES ('GARLIC BREAD WITH CHEESE',120,03.50);
INSERT INTO FOOD_MENU VALUES ('BRUSCHETTA',121,05.95);
INSERT INTO FOOD_MENU VALUES ('FRIES',123,02.30);
INSERT INTO FOOD_MENU VALUES ('DRINKS',124,02.50);
INSERT INTO FOOD_MENU VALUES ('ADD-ON SAUCES',125,02.50);
```

```
-------------------------------------------------------------
CREATE TABLE BILLS (ORDER_NO INTEGER NOT NULL PRIMARY KEY,PAYMENT_METHOD
VARCHAR(255),ORDER_TOTAL FLOAT,ADD_CHARGES FLOAT,TAX FLOAT);

INSERT INTO BILLS VALUES (151,'CASH',46.78,5.89,2.00);
INSERT INTO BILLS VALUES (172,'CREDIT CARD',29.28,2.89,2.00)
INSERT INTO BILLS VALUES (139,'DEBIT CARD',08.15,0.00,1.89);
INSERT INTO BILLS VALUES (123,'PAYPAL',68.85,1.90,2.00);
INSERT INTO BILLS VALUES (105,'PAYPAL',22.85,0.00,1.54);
INSERT INTO BILLS VALUES (178,'CREDIT CARD',33.05,0.00,1.54);
INSERT INTO BILLS VALUES (116,'DEBIT CARD',78.66,2.50,2.10);
-------------------------------------------------------------

CREATE TABLE ORDER_DETAILS (ORDER_NO INTEGER NOT NULL PRIMARY KEY, ORDER_TYPE
VARCHAR(255),ORDER_DATE VARCHAR(20),NO_OF_PERSON INT,ORDER_FILLEDBY
VARCHAR(255));

INSERT INTO ORDER_DETAILS VALUES (151,'DINE-IN','08/06/2021',3,'VANESSA
BENJAMIN');
INSERT INTO ORDER_DETAILS VALUES (172,'TAKEAWAY','02/06/2021',2,'VANESSA
BENJAMIN');
INSERT INTO ORDER_DETAILS VALUES (139,'DINE-IN','05/06/2021',1,'ROMA SOUZA');
INSERT INTO ORDER_DETAILS VALUES (123,'TAKEAWAY','27/05/2021',1,'ROMA
SOUZA');
INSERT INTO ORDER_DETAILS VALUES (105,'TAKEAWAY','30/05/2021',1,'ROMA
SOUZA');
INSERT INTO ORDER_DETAILS VALUES (178,'DINE-IN','01/06/2021',2,'VANESSA
BENJAMIN');
INSERT INTO ORDER_DETAILS VALUES (116,'DINE-IN','03/06/2021',4,'VANESSA
BENJAMIN');
-------------------------------------------------------------
CREATE TABLE STOCK_DETAILS (ITEM_NAME VARCHAR(255),ITEM_NO
VARCHAR(255),QUANTITY VARCHAR(255),RECEIVED_DATE VARCHAR(255),PRICE FLOAT);

INSERT INTO STOCK_DETAILS VALUES ('MOZZARELLA CHEESE','MC01','5KGS',
'06/06/2021',110.00);
INSERT INTO STOCK_DETAILS VALUES ('SAN MARZANO TOMATOSAUCE','S01','10KGS',
'06/06/2021',78.59);
INSERT INTO STOCK_DETAILS VALUES ('BOLOGNESE SAUCE','S02','10KGS',
'06/06/2021',140.25);
INSERT INTO STOCK_DETAILS VALUES ('RICOTTA','CH01','5KGS',
'05/06/2021',88.95);
INSERT INTO STOCK_DETAILS VALUES ('PARMESAN','CH02','3KGS',
'04/06/2021',185.36);
INSERT INTO STOCK_DETAILS VALUES ('OLIVES','V01','2KGS', '07/06/2021',56.69);
INSERT INTO STOCK_DETAILS VALUES ('MUSHROOMS','V02','3KGS',
'07/06/2021',15.75);
INSERT INTO STOCK_DETAILS VALUES ('ONIONS','V03','5KGS', '07/06/2021',13.27);
INSERT INTO STOCK_DETAILS VALUES ('SPINACH','V04','3KGS',
'07/06/2021',37.29);
INSERT INTO STOCK_DETAILS VALUES ('FRESH CREAM','F03','5CANS',
'07/06/2021',89.00);
INSERT INTO STOCK_DETAILS VALUES ('CHICKEN BREAST','F01','10KGS',
'07/06/2021',119.00);
INSERT INTO STOCK_DETAILS VALUES ('SWEET ITALIAN SAUSAGE','F02','10KGS',
'06/06/2021',170.25);

INSERT INTO STOCK_DETAILS VALUES ('SALAMI','F02','2KKS', '07/06/2021',36.75);
INSERT INTO STOCK_DETAILS VALUES ('EGGS','F04','50NOS', '08/06/2021',45.29);
INSERT INTO STOCK_DETAILS VALUES ('TOMATOES','V05','10KGS',
'06/06/2021',26.25);
```

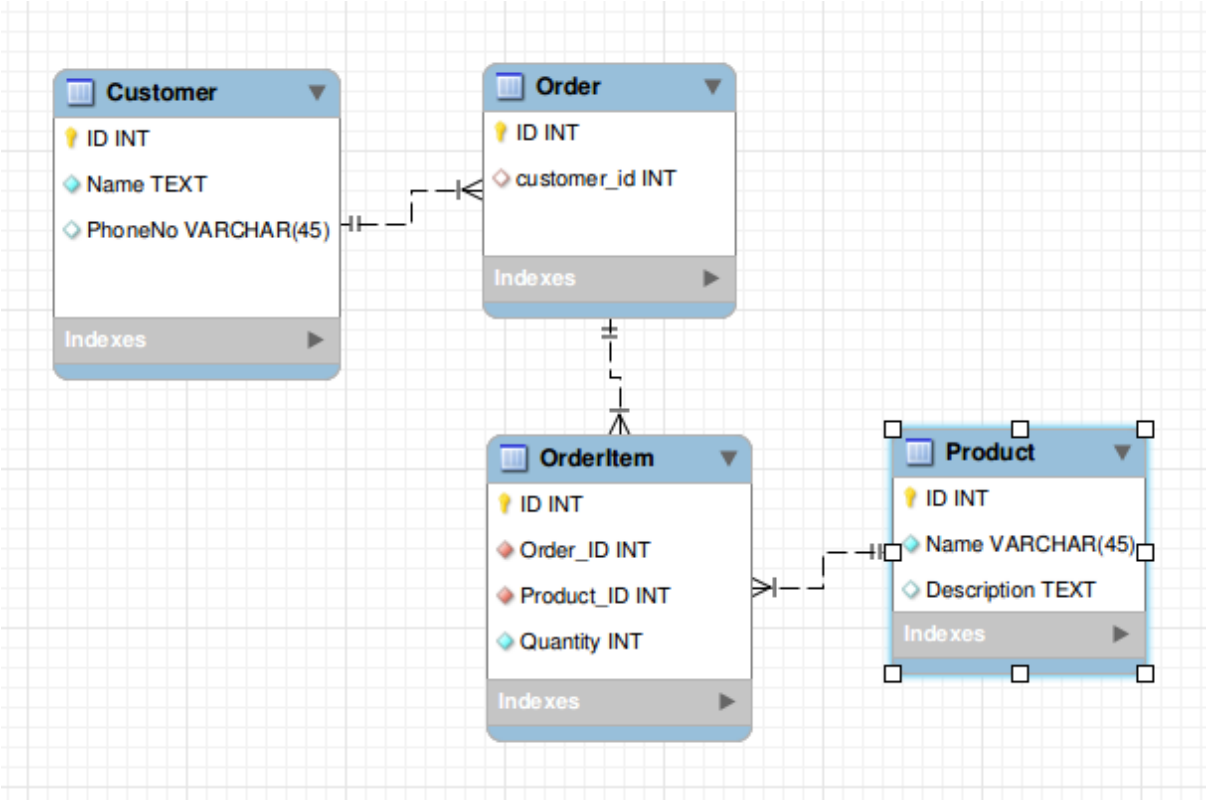
```
INSERT INTO STOCK_DETAILS VALUES ('POTATOES','V06','40KGS',
'06/06/2021',240.58);
```

```
mysql> mysql> select * from CUSTOMER;
+------------+---------+---------------+-----------------------------------------------------------------+
| CUSTOMERID | NAME    | PHONE         | FEEDBACK                                                        |
+------------+---------+---------------+-----------------------------------------------------------------+
|        101 | RICKY   | 4926105678762 | Great experience                                                |
|        102 | OLIVIA  | 4917620456195 | Pleasant interiors and good service                             |
|        103 | TONY    | 4920187789124 | Food was tasty, nice staff, & awesome people                    |
|        104 | BOB     | 4976546286209 | Pricy!                                                          |
|        105 | WILLIUM | 4976542789137 | Very good pizzas! Good service                                  |
|        106 | ANTONY  | 4917628977436 | Good Italian food, friendly staff, good price for quality       |
+------------+---------+---------------+-----------------------------------------------------------------+
6 rows in set (0.00 sec)

mysql> select * from FACULTY;
+----+-----------------+--------------+---------------+--------+-------+
| ID | NAME            | POSITION     | YEAROFJOINING | SALARY | BONUS |
+----+-----------------+--------------+---------------+--------+-------+
| 10 | ROMA SOUZA      | RECEPTIONIST |          2015 |   1200 |     0 |
| 11 | LEO ANDERSON    | CASHIER      |          2010 |   1600 |     0 |
| 12 | MIRELA HANS     | WAITRESS     |          2014 |   1055 |   150 |
| 13 | MAYA NICOLAUS   | WAITRESS     |          2017 |    990 |   100 |
| 14 | LUCY THOMAS     | DISHWASHER   |          2016 |    990 |     0 |
| 15 | GEORGE ENN      | COOK         |          2015 |   1800 |   200 |
| 16 | EMILY MARTIN    | DISHWASHER   |          2018 |    990 |     0 |
| 17 | JULIA PETER     | WAITRESS     |          2017 |    990 |   100 |
| 18 | BONITA JAMES    | COOK         |          2017 |   1800 |   150 |
| 19 | AKSA KAMAL      | WAITRESS     |          2020 |   1055 |     0 |
| 20 | VANESSA BENJAMIN | RECEPTIONIST |          2020 |    990 |     0 |
+----+-----------------+--------------+---------------+--------+-------+
11 rows in set (0.00 sec)

mysql> select * from FOOD_MENU;
+--------------------------+---------+-------+
| ITEM_NAME                | ITEM_ID | PRICE |
+--------------------------+---------+-------+
| MARGHERITA PIZZA         |     111 |  10.3 |
| BOLOGNA PIZZA            |     112 | 12.45 |
| VEGETARIAN PIZZA         |     113 |  10.3 |
| SICILIAN PIZZA           |     114 |  13.5 |
| QUATRO FORMAGGI PIZZA    |     115 |   9.7 |
| LASAGNE                  |     116 |  8.15 |
| RICOTTA CANNELLONI       |     117 |   9.3 |
| CHICKEN PARMIGIANA       |     118 | 12.15 |
| MOZZARELLA STICKS        |     119 |  5.78 |
| GARLIC BREAD WITH CHEESE |     120 |   3.5 |
| BRUSCHETTA               |     121 |  5.95 |
| FRIES                    |     123 |   2.3 |
| DRINKS                   |     124 |   2.5 |
| ADD-ON SAUCES            |     125 |   2.5 |
+--------------------------+---------+-------+
14 rows in set (0.00 sec)
```

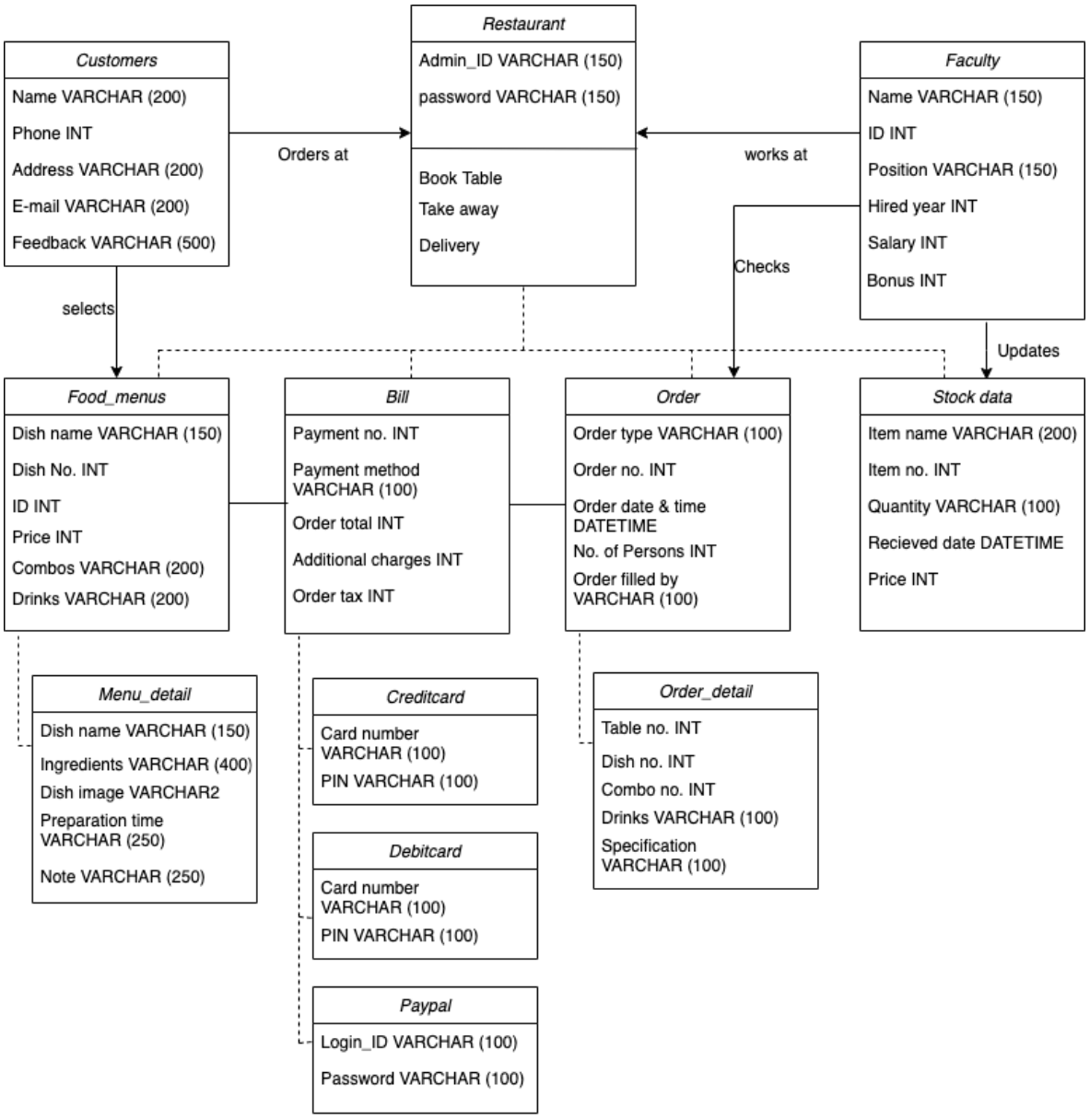
```
mysql> select * from BILLS;
+----------+----------------+-------------+-------------+------+
| ORDER_NO | PAYMENT_METHOD | ORDER_TOTAL | ADD_CHARGES | TAX  |
+----------+----------------+-------------+-------------+------+
|      105 | PAYPAL         |       22.85 |           0 | 1.54 |
|      116 | DEBIT CARD     |       78.66 |         2.5 |  2.1 |
|      123 | PAYPAL         |       68.85 |         1.9 |    2 |
|      139 | DEBIT CARD     |        8.15 |           0 | 1.89 |
|      151 | CASH           |       46.78 |        5.89 |    2 |
|      172 | CREDIT CARD    |       29.28 |        2.89 |    2 |
|      178 | CREDIT CARD    |       33.05 |           0 | 1.54 |
+----------+----------------+-------------+-------------+------+
7 rows in set (0.00 sec)

mysql>
mysql> select * from ORDER_DETAILS;
+----------+------------+------------+--------------+------------------+
| ORDER_NO | ORDER_TYPE | ORDER_DATE | NO_OF_PERSON | ORDER_FILLEDBY   |
+----------+------------+------------+--------------+------------------+
|      105 | TAKEAWAY   | 30/05/2021 |            1 | ROMA SOUZA       |
|      116 | DINE-IN    | 03/06/2021 |            4 | VANESSA BENJAMIN |
|      123 | TAKEAWAY   | 27/05/2021 |            1 | ROMA SOUZA       |
|      139 | DINE-IN    | 05/06/2021 |            1 | ROMA SOUZA       |
|      151 | DINE-IN    | 08/06/2021 |            3 | VANESSA BENJAMIN |
|      172 | TAKEAWAY   | 02/06/2021 |            2 | VANESSA BENJAMIN |
|      178 | DINE-IN    | 01/06/2021 |            2 | VANESSA BENJAMIN |
+----------+------------+------------+--------------+------------------+
7 rows in set (0.00 sec)

mysql> select * from STOCK_DETAILS;
+-------------------------+---------+----------+---------------+--------+
| ITEM_NAME               | ITEM_NO | QUANTITY | RECEIVED_DATE | PRICE  |
+-------------------------+---------+----------+---------------+--------+
| MOZZARELLA CHEESE       | MC01    | 5KGS     | 06/06/2021    |    110 |
| SAN MARZANO TOMATOSAUCE | S01     | 10KGS    | 06/06/2021    |  78.59 |
| BOLOGNESE SAUCE         | S02     | 10KGS    | 06/06/2021    | 140.25 |
| RICOTTA                 | CH01    | 5KGS     | 05/06/2021    |  88.95 |
| PARMESAN                | CH02    | 3KGS     | 04/06/2021    | 185.36 |
| OLIVES                  | V01     | 2KGS     | 07/06/2021    |  56.69 |
| MUSHROOMS               | V02     | 3KGS     | 07/06/2021    |  15.75 |
| ONIONS                  | V03     | 5KGS     | 07/06/2021    |  13.27 |
| SPINACH                 | V04     | 3KGS     | 07/06/2021    |  37.29 |
| FRESH CREAM             | F03     | 5CANS    | 07/06/2021    |     89 |
| CHICKEN BREAST          | F01     | 10KGS    | 07/06/2021    |    119 |
| SWEET ITALIAN SAUSAGE   | F02     | 10KGS    | 06/06/2021    | 170.25 |
| SALAMI                  | F02     | 2KKS     | 07/06/2021    |  36.75 |
| EGGS                    | F04     | 50NOS    | 08/06/2021    |  45.29 |
| TOMATOES                | V05     | 10KGS    | 06/06/2021    |  26.25 |
| POTATOES                | V06     | 40KGS    | 06/06/2021    | 240.58 |
+-------------------------+---------+----------+---------------+--------+
16 rows in set (0.00 sec)

mysql>
```

7. Data Operation

Due to timely restrictions, the actual infrastructure of the database looks akin to this:



With that said, taking into account future implementation would upgrade the systems state:



8. Triggers and Stored Procedures

We implemented one trigger and two stored procedures:

```
delimiter //
CREATE TRIGGER totalOrdersOfCustomer
AFTER INSERT ON customer_order FOR EACH ROW
PRECEDES checkSusCustomer
BEGIN
DECLARE updatecount INT;
    set updatecount = (SELECT COUNT(*) FROM customer_order WHERE CustomerId = NEW.CustomerId);
    IF EXISTS (SELECT * FROM order_management WHERE CustomerId = NEW.CustomerId) THEN
        DELETE FROM order_management WHERE CustomerId = NEW.CustomerId;
    END IF;
    INSERT INTO order_management (CustomerId, Orders)
    VALUES (NEW.CustomerId, updatecount);
END //
```

The trigger stores in a second table *order_management* how many orders there are for each customer. Every time a new order is inserted into the *customer_order* tabe this trigger is executed.

```
CREATE PROCEDURE GetBiggestOrders(IN number INT)
SELECT OrderId, COUNT(OrderId)
AS items_ordered
FROM order_item
GROUP BY OrderId
ORDER BY items_ordered DESC
LIMIT number;
```

```
CREATE PROCEDURE findSusCustomers()
SELECT * FROM order_management WHERE Orders > 3;
```

We defined two stored procedures. The first one gets the biggest order taken by a single customer. The procedure takes one parameter which specifies how many results it should return. The use-case of the second procedure is to find suspicious customers. This procedure returns all customers which executed more than 3 orders.

9. Functionalities

The database system currently supports these functionalities:

1. The listing of all-time customers

If a customer has ever ordered something off the restaurant, their data will be stored into the database.

2. Listing of feedback

Customers can leave feedback which will be recorded into the database and made accessible for the responsible person when needed.

3. Listing of contact information

Alongside the customers details, their contact information is also stored, making it possible to judge in the future if a past customer is currently calling or not.

4. Listing of the menu

The menu items are available in the database, alongside their price.

5. Listing of orders

Pending orders are listed sorted by the time of creation. Using this, orders can be handled in a cohesive manner.

6. Trigger: Customer Order

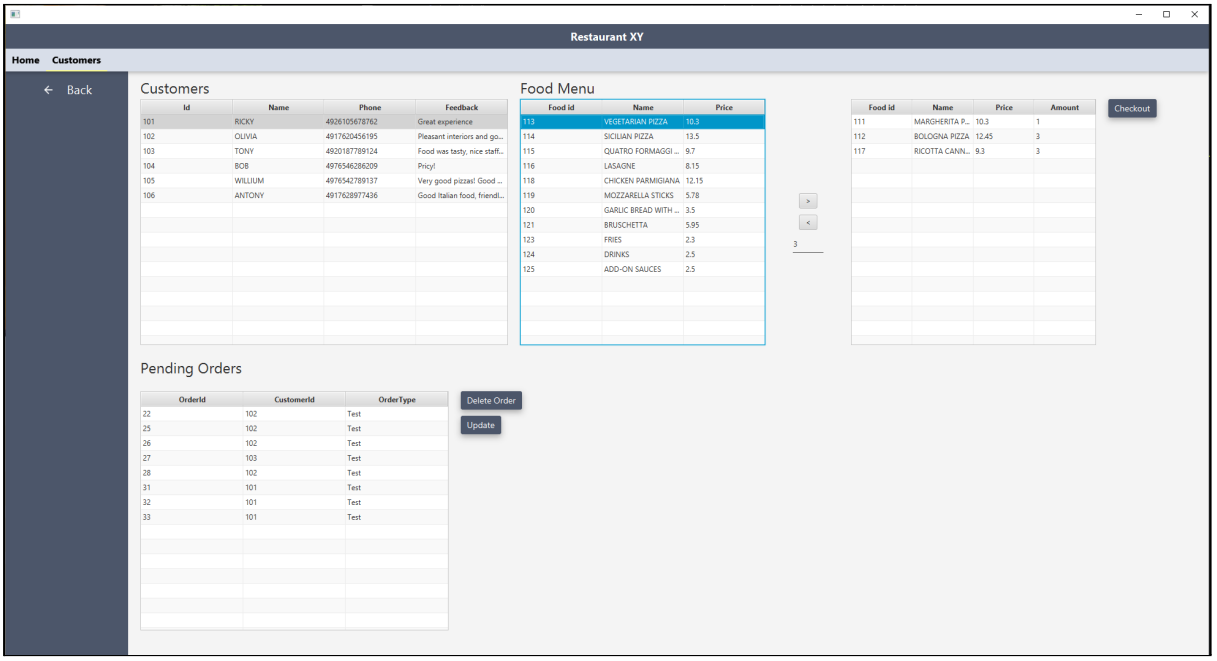
Once a customer places an order, a trigger runs in the background keeping track of the amount of times this particular customer has decided to purchase something off of the restaurant.

7. Stored Procedures: Customer Order

Using the integer gathered from the trigger above, the restaurant can enquire about the most loyal customer at any given time. This result can also return more than one customer.

10. Conclusion

To conclude, it can be said that the implementation of the database management system in the field of gastronomy, in particular restaurant management is a great success. Using not only the storage offered, but also the functionality, a great amount of responsibility, stress and effort can be relieved off the responsible employees. Although the delivery of this documentation is only a prototype, its usefulness can already be put into context.



11. Source Code

To get a structured view of the code please refer to: https://github.com/Philipp0205/rdbms

12. References

Hochschule für Technik Stuttgart - Software Technology, Summer Semester 2021, Database Systems II.