**Project Report**

R&J Burgers

Restaurant Management System

## **Overview**

The goal of the project is to create an application for a restaurant based on a given business rules as well as with some assumptions necessary for the better implementation. The application consists of a database and a client application.

## **Results**

The Database was built for a restaurant management system.

* Tables were normalized to BCNF.
* Table indexes were created to increase queries efficiency.
* The cost of the various queries was calculated.

UI was built to perform various operations for the restaurant management system. The application has the below features.

* Manage Employees - Add/Modify/Delete
* Manage Customers - Add/ Modify/Delete
* Manage Orders - Add/Modify/Delete
* Manage Delivery Areas - Add/Modify/Delete

## **Business rules**

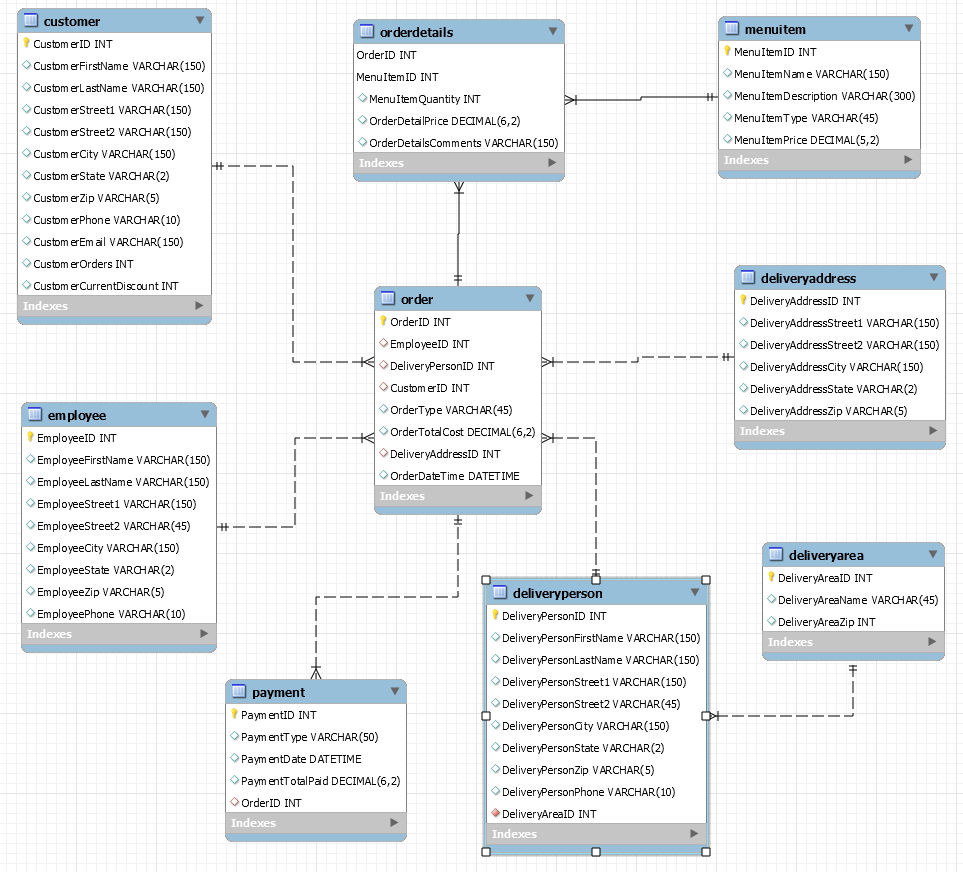
1. Fast-food restaurant “R&J Burgers” is based in Plano, serves burgers, sides, and non-alcoholic beverages
2. Apart from providing food facility at their own premise, the restaurant takes orders online through its site. Phone orders are also entertained.
3. No servers on the premises, guests serve themselves.
4. Each employee of restaurant trained to perform several duties:
   1. takes orders (on premises, online, on phone),
   2. accepts payments,
   3. assembles orders,
   4. gives orders to customers or assigns deliveries to delivery personnel
5. To deliver the orders, the restaurant has delivery personnel. Each delivery person is assigned to a specific area code. The delivery person can deliver only to the assigned area. Delivery is possible in the area within 15 miles radius.
6. An order can either be DINE-IN or TAKE-OUT order.
7. A customer record is maintained so that premium customers can be awarded discounts. There are 3 types of discounts:
8. for customers who made 10 – 30 orders – 3%
9. for customers who made 31 – 50 orders – 5%
10. for customers who made 51 – and more – 7%

## **Implementation Technologies**

|  |  |
| --- | --- |
| **Purpose** | **Technology** |
| Database | MySQL 8.0.2 (Community Edition) |
| Database GUI Tool | MySQL Workbench 8.0.21 |
| Backend | Java/J2EE 1.8, Servlet, JDBC, Maven |
| User Interface | HTML 5, JSP, JavaScript, CSS |
| Web Server | Apache Tomcat 8.5.59 |

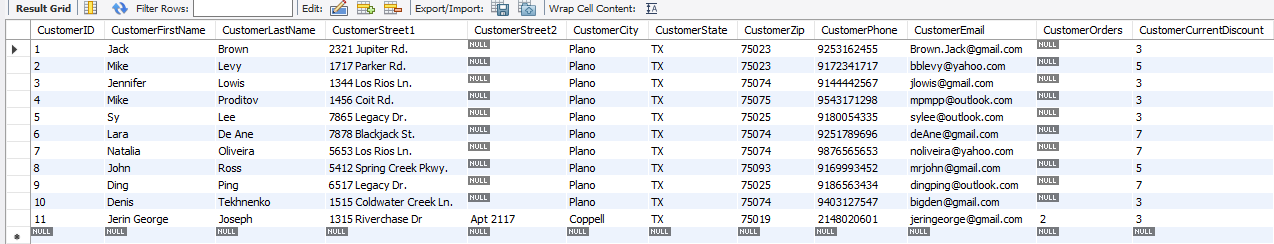
**Database Design**

**Schema**

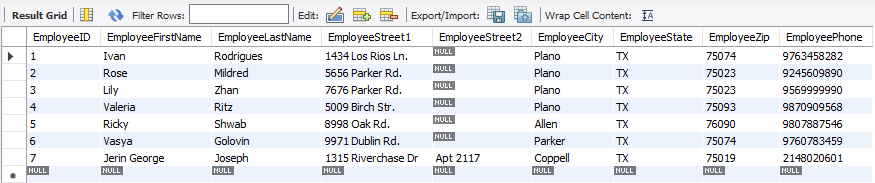
****

The r\_j\_burgers database includes 9 relations which show only “order - delivery” activities of a restaurant’s business:

1. Customer

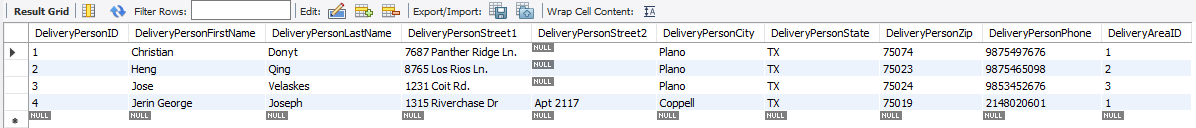


1. Employee

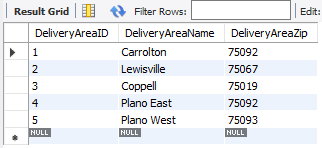


1. DeliveryPerson

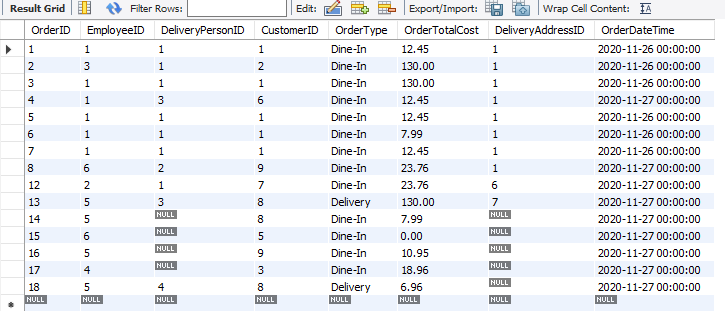
The delivery person details are maintained independently from the Employees because they participate in relationships that are unique. Every delivery person is assigned to a delivery area, and to only one delivery area.



1. DeliveryArea

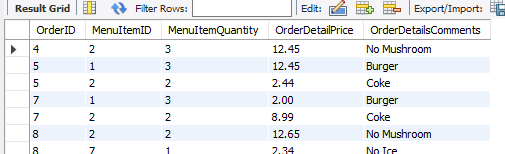


1. Orders



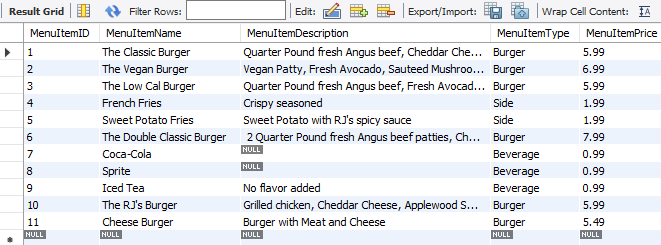
1. OrderDetails

The table was introduced to avoid a many-to-many relationship between the Orders and MenuItem tables.

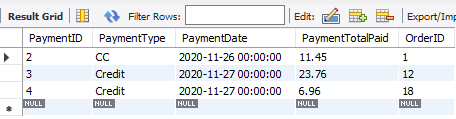


1. MenuItem

The list of items in the menu is stored in the MenuItem table. The price of each item is stored in this table. When the menu items are added to the order, the application uses this price to calculate the total.



1. Payment

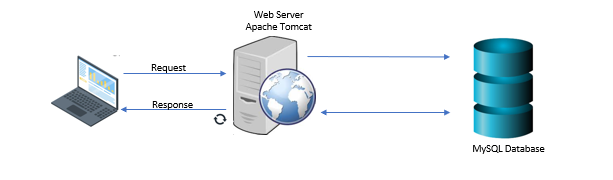


1. DeliveryAddress

The table enhances flexibility of the Database since the customer address could differ from the delivery address.

**Application Design**

The application follows the MVC (Model - View - Controller) Architecture, where the model, controller and view is separated into distinct layers.

****

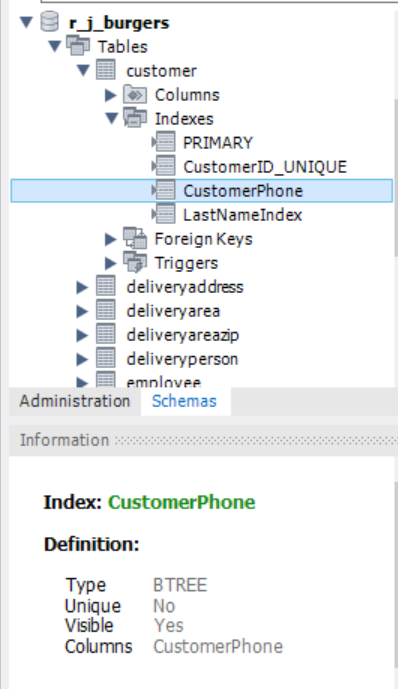
**Analysis**

1. **in terms of speed:**

Index on phone number column (customer table) was used to demonstrate how speed of queries depends on indexes. Results are in the below table:

|  |  |  |
| --- | --- | --- |
| Query | SELECT \*  FROM\_j\_burgers.customer  WHERE r\_j\_burgers.customer.CustomerPhone =  "9172341717"; | SELECT \*  FROM\_j\_burgers.customer ignore index  (CustomerPhone)  WHERE r\_j\_burgers.customer.CustomerPhone =  "9172341717"; |
| Results (seconds) | starting 0.000173  checking permissions 0.000018  Opening tables 0.000053  init 0.000094  System lock 0.000030  optimizing 0.000031  statistics 0.000249  preparing 0.000032  executing 0.000013  Sending data 0.000142  end 0.000011  query end 0.000018  closing tables 0.000017  freeing items 0.000120  cleaning up 0.000027 | starting 0.000132  checking permissions 0.000014  Opening tables 0.000033  init 0.000058  System lock 0.000017  optimizing 0.000019  statistics 0.000040  preparing 0.000023  executing 0.000009  Sending data 0.023235  end 0.000018  query end 0.000018  closing tables 0.000017  freeing items 0.000176  cleaning up 0.000027 |
| Query Cost |  |  |

Type of Index is BTREE by default in MySQL:



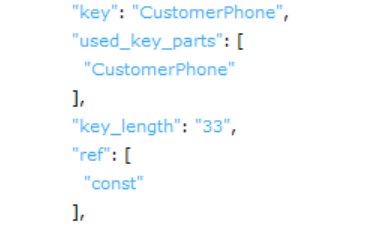
We created two indexes in the Customer table - for the CustomerPhone and CustomerLastName attributes. To run the next query the CustomerPhone index was used as it is more efficient:

SELECT \*

FROM r\_j\_burgers.customer

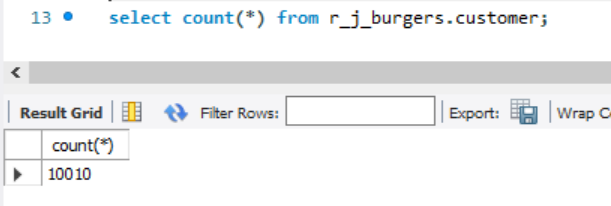
WHERE r\_j\_burgers.customer.CustomerPhone = "9172341717" AND

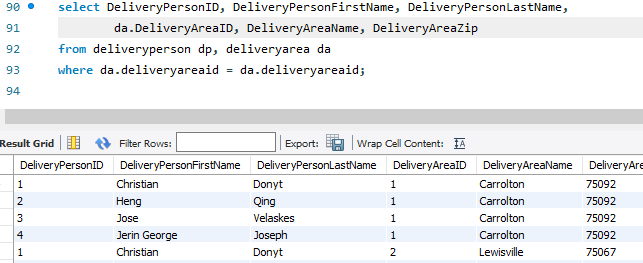
r\_j\_burgers.customer.CustomerLastName = "Levy";



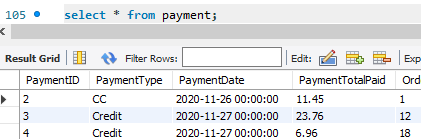
1. **in terms of accuracy:**

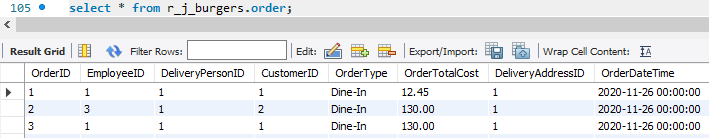
The results of the following queries prove accuracy in the database design and implementation.





As shown below, we maintained the same date format across the application to maintain accuracy





1. **What challenges did you face in the development process?**

* Lack of raw data - To test the performance of the indexes and the query plans, we did not have enough data to perform the test.
* Some technical issues during the design process:
* Database schema must have unique names for constraints
* The initial name for the ORDERS table was ORDER which is a keyword.
* Faced some access issues because the user id used did not have the grants to create tables.

1. **How did you overcome those?**

* Lack of raw data - We created an utility program to insert 10,000 rows into the customer table to setup the data for testing.
* To avoid the ORDER Table name clash with the keyword, we used the table name along with the schema as “r\_j\_burgers.ORDERS.
* Used documentations to resolve technical issues.

1. **What are the limitations of the project?**

* The application does not have an authentication mechanism, it can be included.
* The application does not allow the tip feature which is very important for American restaurants.
* The application is currently hosted locally on our computer, the application can be deployed in the cloud and be open to the internet, so that multiple users can use it.

1. **Path to future work?**

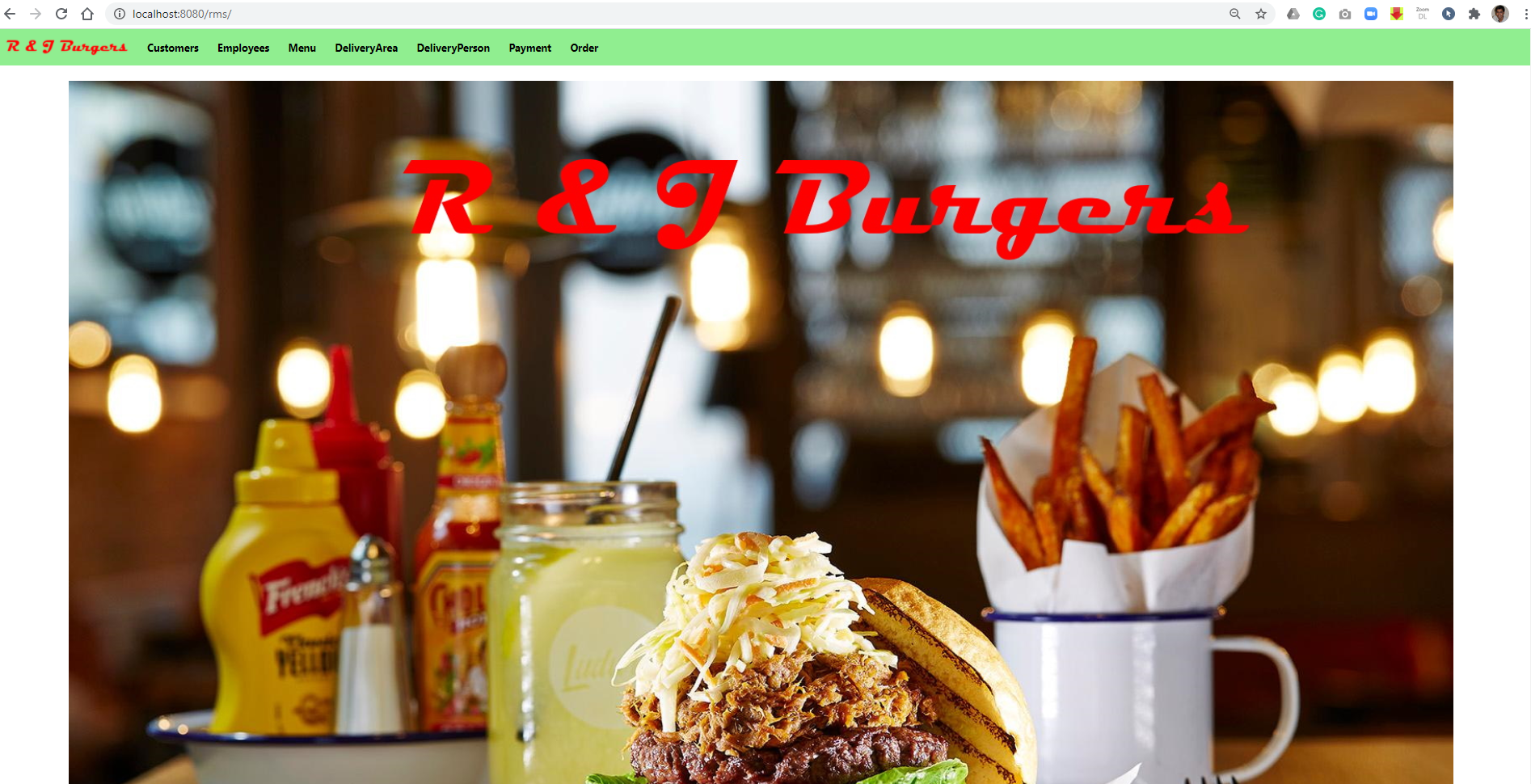
* Extend database with integrations to payment processors
* More customer data, more orders’ records and hence optimize the application to handle higher volume.

**References**

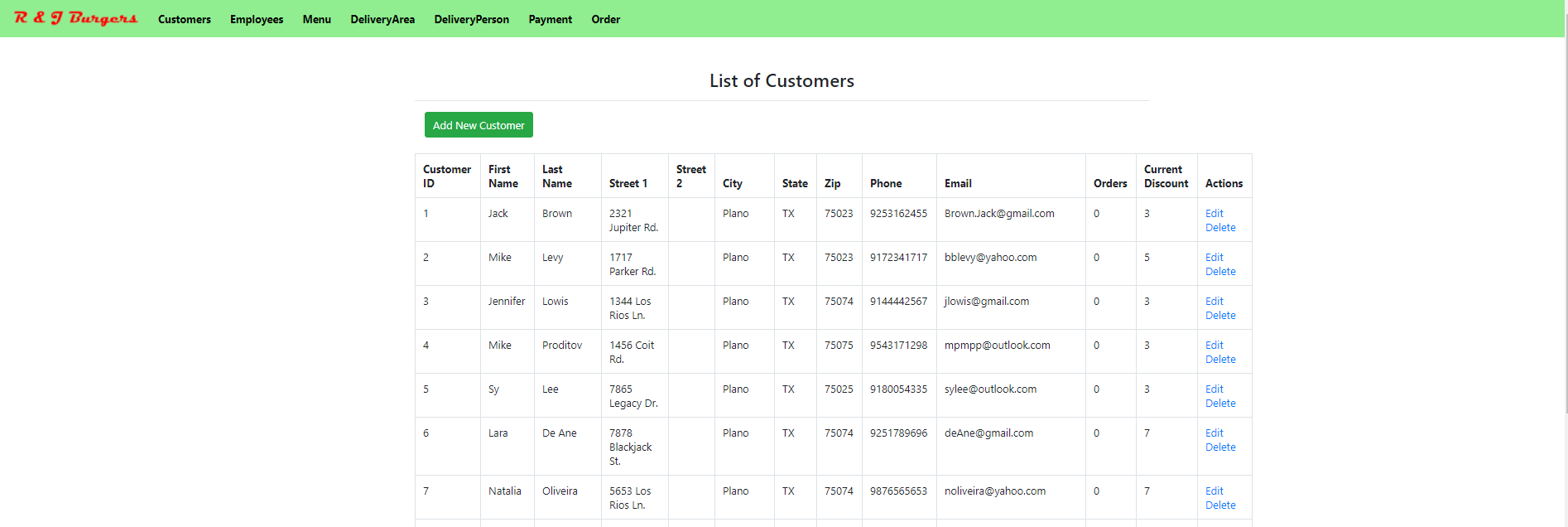
* Database System Concepts (Seventh Edition) by [Avi Silberschatz](http://www.cs.yale.edu/homes/avi), [Henry F. Korth](http://www.cse.lehigh.edu/~korth), [S. Sudarshan](http://www.cse.iitb.ac.in/~sudarsha);McGraw-Hill
* Database Systems: Design, Implementation, & Management (13th edition) by Carlos Coronel, Steven Morris
* MySQL Workbench - <https://dev.mysql.com/doc/workbench/en/>
* MySQL Indexes - <https://dev.mysql.com/doc/refman/8.0/en/mysql-indexes.html>

**Appendix**

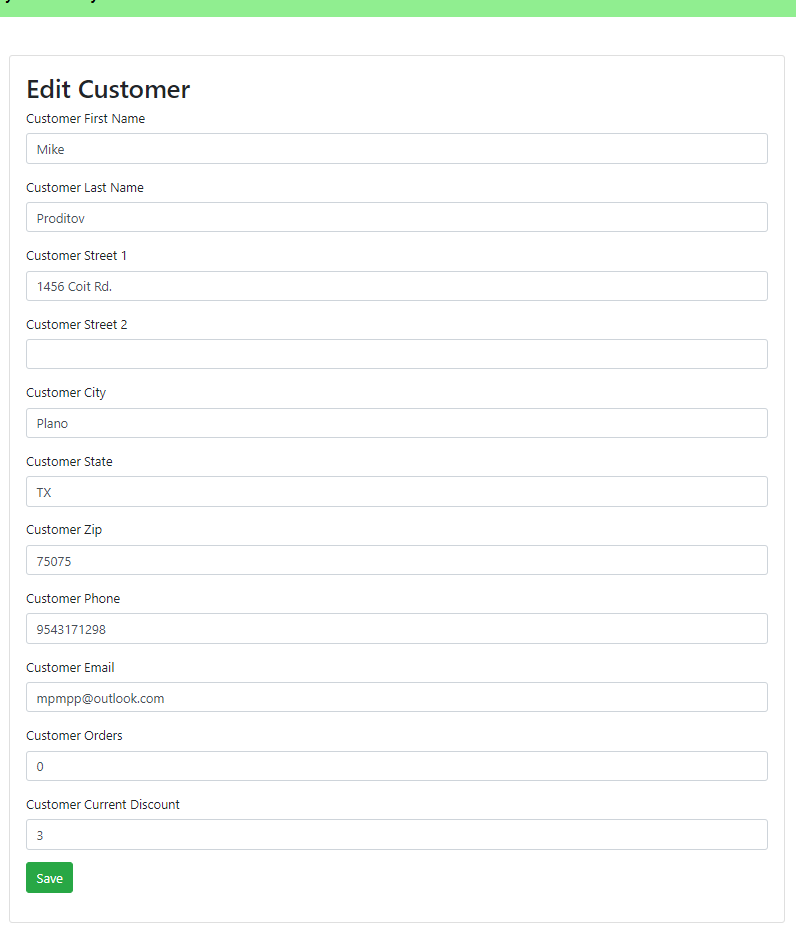
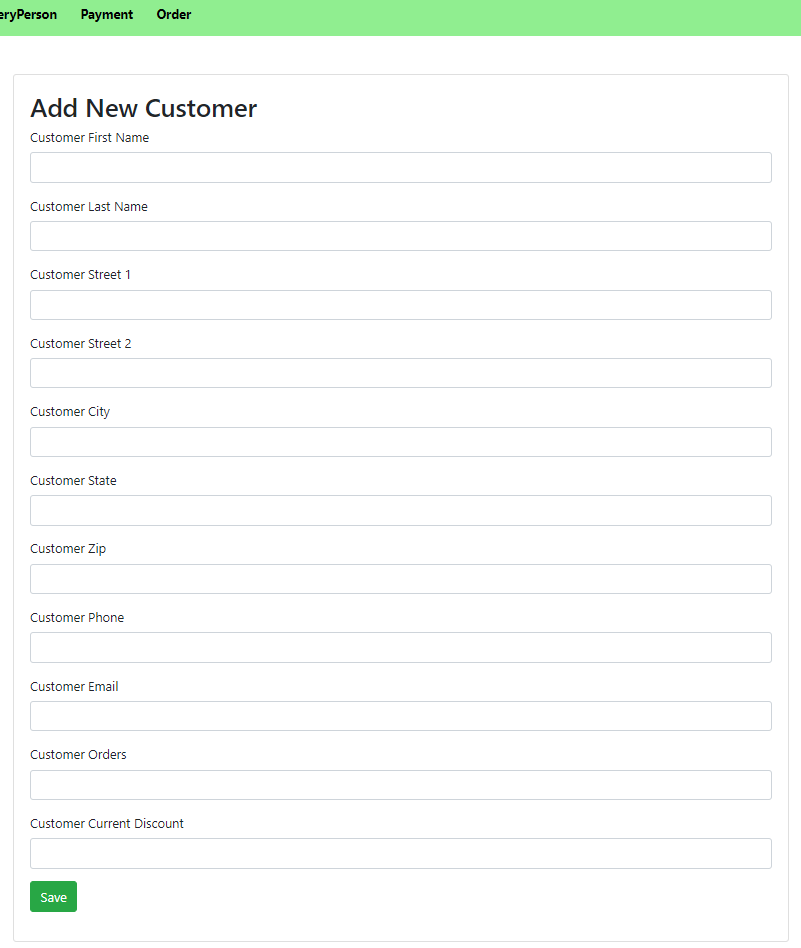
Appendix A - UI Screenshots - Homepage

****

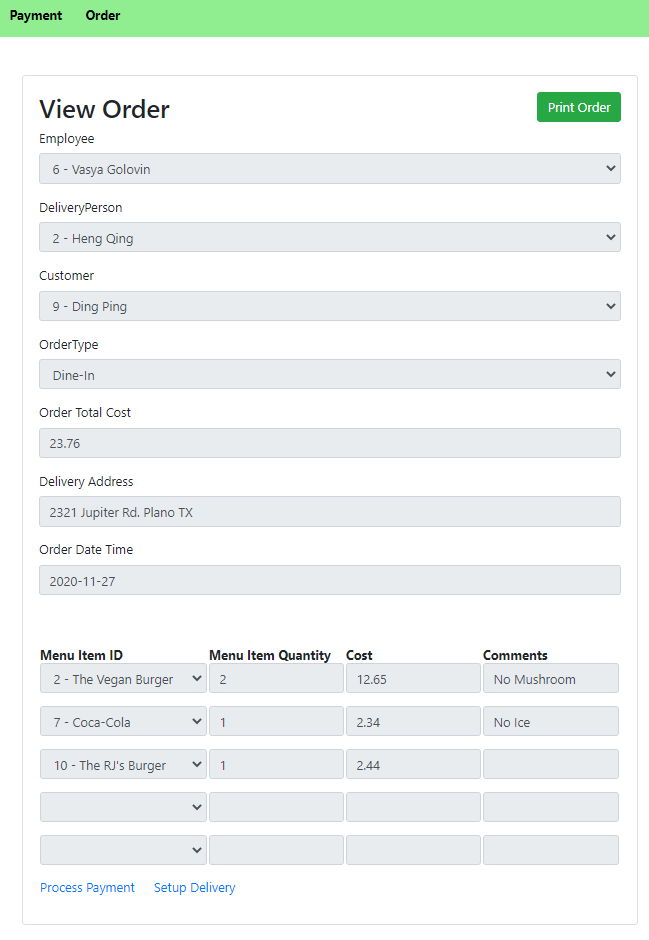
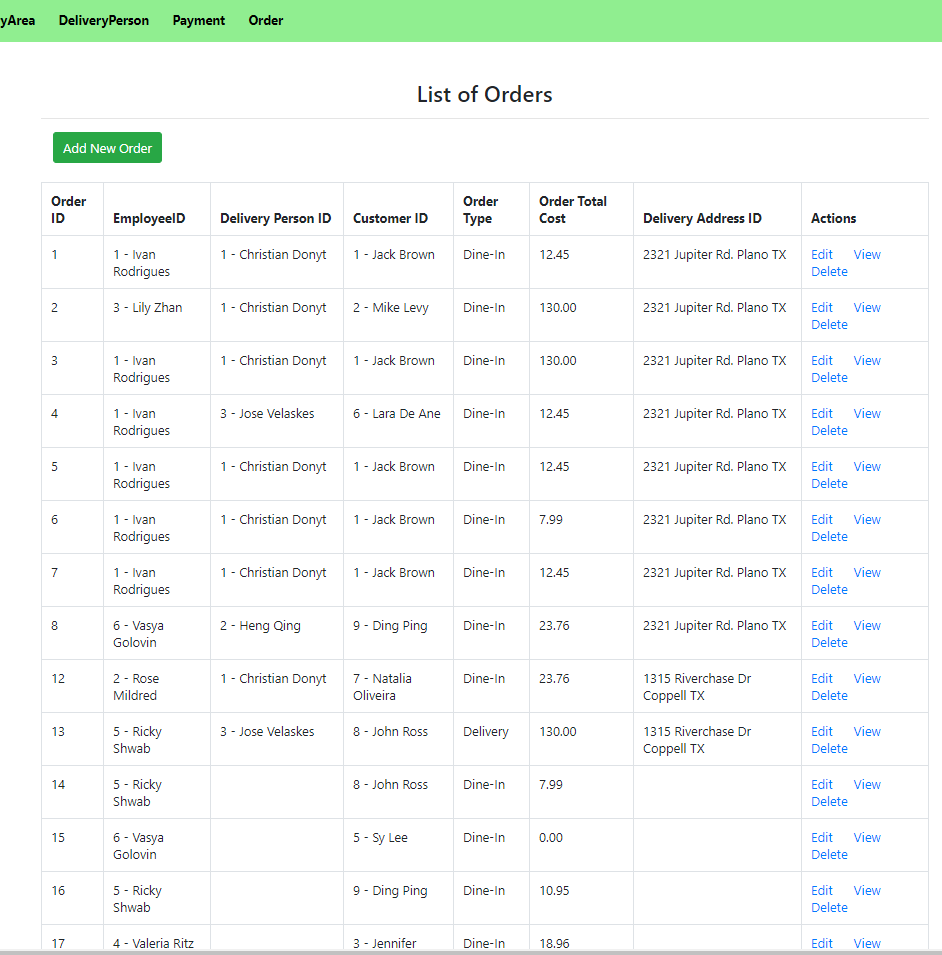
Appendix B - List of Customers Screen

****

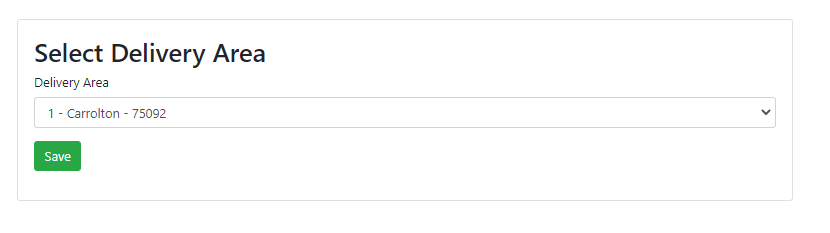
Appendix C - Add & Edit Customer

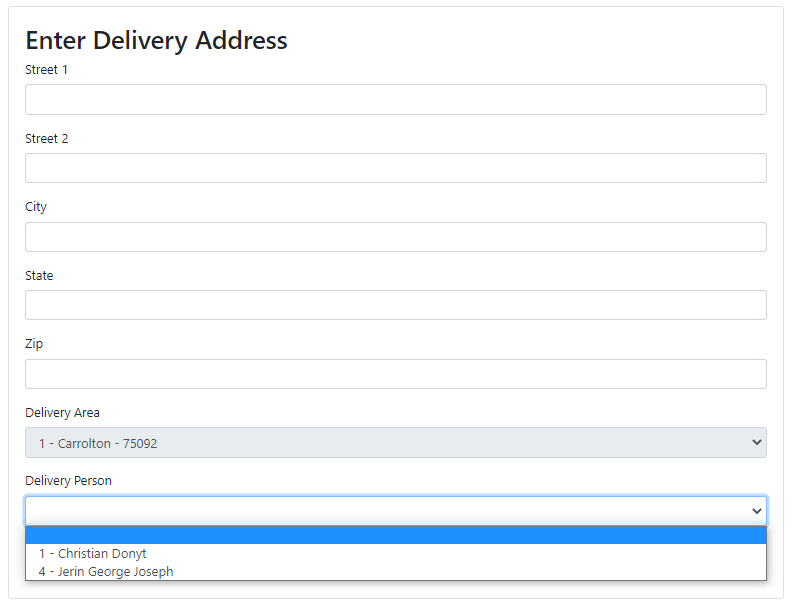
****

Appendix D - List of Orders & View Order

****

Appendix E - Delivery Area and Payment

****

****

****