**Formal Specifications**

**Final Application:**

**Employee** - A table with the personal information such as their name, id, address, their employee id#, their position at work, password for the login, phone number, and the amount of hours worked each day, and pay rate.

**Orders** - A table of orders uniquely represented by their order id#, a description of what the order was and how much it cost, and it’s payment type(credit,debit, cash, etc.). Also, it includes customerID# for the customer who made the order.

**IsTakenBy** - This represents the data of the employees who took each order by their respective id#s.

**Customer**- A table which has each customer uniquely represented by a customerID#, with each individual customer there will be the personal information such as the name, address, password for the login,and phone number and favorite menu item. It also includes employeeID of the employee who delivers food to the customer.

**Inventory** - There will be a table which stores, and updates the information of the inventory via sales and profit/loss and includes the employeeID who does the inventory.

**Items** - The information of each individual item will be stored in a table as well where it will have the item’s unique id#, how much of that item is left, the cost of the particular item, and the expiration date.

**Supplier** - information on the suppliers will also be handled, the data on the supplier will include the name, address, what kind of supplies do they bring, and their unique id#.

**menuItems** - This entity includes the menu items offered at the restaurant. It will store the name of each item, their price, which kind of item it is (food,beverage, desert, etc.). Each item will also have a unique id that can distinct each individual item, just in case there are two items with the same name.

**Contains**- A relation that includes the menuItemid, and order Id to check which menu item includes which order, or which order includes which menu items.

**consistOf** - contains the items used in making menu items or contains the menu items made of specific item. Includes menuItemsId, itemID

**delivers** (supplies changed to deliver)- this relation will be a connection between the supplier and the restaurant items which includes supplier id, item Id, item quantity and delivery date.

**Constraints**:

* All the id of each table will be used as primary keys and foreign keys.
* The hours worked and pay rate in employee table must be greater than zero. A check statment is used to satisfy this condition.
* The id’s must be unique and must not be empty. A unique statement, and not null is used to satisfy this condition.
* The name variable for every table must not be empty as well. Not null is used again to satisfy that property.
* Menu item id and employee id are used as foreign keys in customer table to satisfy the favorite item and delivery employee requirement.
* In order table, the customer id is used as foreign key to check which customer made the order. The amount variable is checked to be greater than zero.
* All the foreign keys are set null or empty if the information is deleted from the parent table.
* All the price or amount variable have numeric data type which allows 2 decimal places.
* Expiration date has the data type date.
* All many-to-many relations includes id’s of the table being used in relation. Id’s are used as foreign keys.
* Quantity variables must be greater than zero. Check statment is used in SQL to handle it.

**Administrator features** - In this case, the administrator would be the manager or supervisor and they would be able to login into an admin account where they could access the inventory data to view/modify the current inventory. They would also have permission to view/modify regular employee hours worked if need be. Basically admins would be able to view everything, and modify everything even orders.

**Employee user features** - employees who are not the manager or supervisor will also have login credentials where they can log in and view their hours worked, however, they will not be able to view other employee’s hours worked, and they can not check the inventory. That will also be handled in web.py where the if statement will check the position of the employee. If the employee is administrator, he can check anything, otherwise the access will be denied to other pages.

**Customer features-** It includes customer logging in and seeing menu item. Customer can select menu item and place the order. The cool feature includes customers having their favorite menu item in the their table or information.

**Order page**- includes all the orders that have been placed by the customers. Names of the employees who are working on the order etc.

**Labor Division:**

**Muhammad**- created tables and database. Will create the customer and employee login pages.

**Edem -** will do the registration and home pages for the employee and customer.

**Sharod** - will implement the parsing of sql queries in web.py.

**DDL Tables**

CREATE TABLE Employee(

empid varchar(8) not null,

epasscode varchar (120) not null,

UNIQUE(empid),

ename varchar(15) not null,

Address varchar(50),

ephonenum numeric(10),

hoursworked numeric(2) check (hoursworked > 0),

position varchar(10),

payrate numeric(4,2) check (payrate > 0),

primary key (empid)

);

CREATE TABLE menuitem(

menuitemid varchar(8) not null,

UNIQUE(menuitemid),

miname varchar(15) not null,

price numeric(4,2) check (price > 0),

mitemtype varchar(8),

primary key (menuitemid)

);

CREATE TABLE Customer(

cusid varchar(8) not null,

UNIQUE(cusid),

cusname varchar(15) not null,

cusphonenum numeric(10),

cpasscode varchar(120),

Address varchar(50),

empid varchar(8) not null,

menuitemid varchar(8) not null,

primary key (cusid),

foreign key (empid) references Employee

on delete set null,

foreign key (menuitemid) references menuitem

on delete set null

);

CREATE TABLE orders(

orderid varchar(8) not null,

UNIQUE(orderid),

amount numeric(6,2) check (amount > 0),

Instructions varchar(200),

Paymenttype varchar(10),

cusid varchar(8) not null,

primary key(orderid),

foreign key(cusid) references Customer

on delete set null

);

CREATE TABLE isTakenby(

orderid varchar(8) not null,

empid varchar(8) not null,

foreign key (orderid) references orders

on delete set null,

foreign key (empid) references Employee

on delete set null

);

CREATE TABLE inventory(

inventoryid varchar(8) not null,

UNIQUE(inventoryid),

Sales numeric(6,2) check (Sales > 0),

Profit numeric(6,2),

empid varchar(8) not null,

primary key (inventoryid),

foreign key (empid) references Employee

on delete set null

);

CREATE TABLE items(

Itemid varchar(8) not null,

UNIQUE(Itemid),

Itemname varchar(15) not null,

Itemquantity numeric(5) check (Itemquantity>0),

Cost numeric(5,2) check (Cost>=0),

Expirationdate date,

primary key(Itemid)

);

CREATE TABLE supplier(

suppliername varchar(15) not null,

supplierid varchar(8) not null,

UNIQUE(supplierid),

sphonenum numeric(10),

saddress varchar(50),

suppliertype varchar(10),

primary key(supplierid)

);

CREATE TABLE delivers(

supplierid varchar(8) not null,

Itemid varchar(8) not null,

Deliverydate date,

foreign key (Itemid) references items

on delete set null,

foreign key (supplierid) references supplier

on delete set null

);

CREATE TABLE consistof(

menuitemid varchar(8) not null,

Itemid varchar(8) not null,

foreign key (Itemid) references items

on delete set null,

foreign key (menuitemid) references menuitem

on delete set null

);

CREATE TABLE contain(

orderid varchar(8) not null,

menuitemid varchar(8) not null,

foreign key (orderid) references orders

on delete set null,

foreign key (menuitemid) references menuitem

on delete set null

);