

Online Pizza Order System

Aim

- To develop a console based application for a pizza restaurant.
- Customer can view menu and order one or more pizzas; while admin (shopkeeper) can keep track of orders.

Technologies to use

- Core Java
 - JDBC
- Database:
 - MySQL

Requirements

- This console application comprises two modules i.e. customer module and admin module.

Customers Module

- Customer module provides following functionalities (to the customers):
 - To Register a New User (along with contact details and delivery address)
 - To Sign In (with email ID and password)
 - To View Pizza Restaurant Menu which is categorized as Veg and NonVeg. There are multiple subcategories like:
 - Veg: Simply Veg, Veg Treat, Veg Special, Veg Feast, Sides
 - NonVeg: Simply NonVeg, NonVeg Treat, NonVeg Special, NonVeg Feast, Sides
 - To select any pizza from above sub-categories and view its details.
 - To select one or more sizes (Regular, Medium or Large) of selected pizza and add to cart. Obviously customer can go back to the main menu to choose more pizzas.
 - To view the cart and purchase selected items. Cart is to be managed as a collection in memory, but Order is to be saved in Database table.


- Note that purchase facility is not to be implemented (just to be emulated) i.e. customer order will be added into database, so that it can be viewed by the restaurant manager. Scope of this application does not include accounting or business analysis of the restaurant.

Admin Module

- Admin module provides following functionalities (to the restaurant manager):
 - To view the all orders in descending order by date.
 - Note that pizza delivery (i.e. status = dispatched or delivered) is not included in the scope of this application.

Design

Database design

- To simplify development database is not fully normalized.
- There are five tables:
 - PIZZA_CUSTOMERS, PIZZA_ITEMS, PIZZA_PRICING, PIZZA_ORDERS & PIZZA_ORDERDETAILS.
- Each table has primary key which is auto generated using MySQL Auto Increment feature.
- Tables have One to Many & Many to One relationships as shown in ER diagram.  ER Diagram
- The .sql files are attached to create database structure and insert basic data quickly.

DAO design

- There should be POJO (entity class) corresponding to each table in database.
 - Customer, Item, ItemPrice, Order, OrderDetails.
- All POJOs (entity classes) follow annotation based configuration.
- For all POJOs (entity classes) IDs are auto-generated.
- All relations are bi-directional (i.e. One-To-Many & Many-To-One) except last one:
 - Customer 1 -> * Order
 - Item 1 -> * Pricing
 - Order 1 -> * OrderDetails
 - OrderDetails * -> 1 Pricing (Uni-directional)
- Implement DAO classes with given functionalities as follows:
- MenuDao:

- Fetch Types i.e. Veg & NonVeg
 - Fetch Sub-Categories (see requirements)
 - Fetch Items (Pizzas) of given Type & Sub-Categories (see requirements)
 - Fetch Item of given Id
 - Fetch ItemPrice of given Id
- OrderDao:
 - Insert given order along with order details.
 - Fetch all orders in descending order of time.
 - Fetch order of given id.
- LoginDao:
 - Insert new customer.
 - Fetch customer by email.

UI design

- Top Level menus
 1. Sign In
 2. Sign Up
 3. Exit
- Customer menu
 1. Show Veg Items
 2. Show Non-Veg Items
 3. Show available sizes (for given Item id)
 4. Add to cart (for given price id)
 5. Show Cart (Pizzas with Size & Price Details)
 6. Place Order (Save Order in Database for current customer)
 7. Sign Out
- Admin menu
 1. Show all orders
 2. Show order details (for given order id show Pizza & Customer details)
 3. Sign Out

Implementation Plan/Steps

1. Database: execute given .sql files (on appropriate database).
2. Console App: Implement nested menus for application.
3. Console App: Implement all POJOs/Entities and Daos.
4. Console App: Sign In & Sign Out, Display Hello message for authenticated user and store its reference (in some static field).
5. Console App: Registration
6. Console App: Pizza Menu Display (Veg/Non-Veg Items)
7. Console App: Pizza Item Details (Size selection) & add to customer cart. Cart may be implemented as `List<ItemPrice>`.
8. Console App: Show cart contents with total bill.
9. Console App: Purchase i.e. add order along with its details into database.
10. Console App: Display all orders list for admin login.
11. Console App: Display Order details for given order id.