Project Overview

DCIT308 Group 7

Version 1.0.0

# PROJECT OBJECTIVES

* Design a user interface
* Implement data structures such as stacks, queues, list
* Design a backend architecture
* Implement the backend architecture

# Approach for Managing the Project

We designed a UI interface for a desktop application.

We implemented the UI design using JavaFX

SQL database to design a backend interface

. We created it locally.

INTRODUCTION

This project is an inventory system that is backed by a database. The inventory system contains products, their categories and prices. It also has vendors and the items they sell. The inventory system is used to issue goods to customers.

The focus of the system is to apply knowledge of data structures and algorithms in java to a real world application, an inventory system.

With the inventory system, one can view goods, view bills, view vendors, issue and view the issued goods. The admin can add categories and products to the categories. There were different data structure that were implemented for the inventory system. Stacks queues, lists were used in adding and removing of items in the various categories. Maps were used to store information about vendors. The different data structures were implemented per the category of the product. For products between category one to four, stacks were implemented. For products between category five to eight, queues were implemented and for products between nine to eleven, lists were implemented in the addition and removing of products.

IMPLEMENTATION

Data Structures used:

1. Stacks
2. Queues
3. Lists
4. HashMaps
5. Maps
6. Iterators

IDE USED: IntelliJ

Database: MySQL(PhpMyAdmin)

IMPLEMENTATION OF DATA STRUCTURES

package sample;  
  
import classes.Product;  
import classes.Sale;  
import classes.Vendor;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
  
import java.util.\*;  
  
public class Management {  
 // data structures  
 public Stack<Product> products\_stack = new Stack<Product>();  
 public Queue<Product> products\_queue = new LinkedList<Product>();  
 public List<Product> products\_list = new ArrayList<Product>();  
 public Map<Integer, String> productSales = new HashMap<Integer, String>() ; // <Sale ID, Product Code>  
 public HashMap<String, String> vendors = new HashMap<String, String>(); // <Vendor Name, Vendor Category>  
  
  
 // add a product  
 public void addProduct(Product product){  
 System.*out*.println(product);  
  
 // category of product  
 int cat\_id = product.category.getValue();  
  
  
 if (cat\_id >= 1 & cat\_id <= 4) {  
 System.*out*.println("=====================Pushing to stack============================");  
 products\_stack.push(product);  
 } else if (cat\_id >= 5 & cat\_id <= 7) {  
 products\_queue.add(product);  
 } else if (cat\_id >= 8 & cat\_id <= 11) {  
 products\_list.add(product);  
 }  
 }  
  
  
 // delete product  
 public void deleteProduct(int cat\_id){  
  
 if (cat\_id >= 1 & cat\_id <= 4) {  
 products\_stack.pop();  
 System.*out*.println("Popping");  
 } else if (cat\_id >= 5 & cat\_id <= 7) {  
 products\_queue.remove();  
 } else if (cat\_id >= 8 & cat\_id <= 11) {  
 products\_list.remove(0);  
 }  
 }  
  
  
 // add a product sale  
 public void addProductSale(Sale sale){  
 int saleID = sale.id;  
 String productCode = sale.product\_code;  
  
 // add to Map data structure  
 productSales.put(saleID, productCode);  
  
 }  
  
 // add a product sale  
 public void addVendor(Vendor vendor){  
 String vendor\_name = vendor.getName();  
 String vendor\_category = vendor.getCategory();  
  
 // add vendor information to HashMap data structure  
 vendors.put(vendor\_name, vendor\_category);  
 }  
  
 // method for javafx collections  
 public ObservableList<Product> dsToObservableList(Integer category){  
 ObservableList<Product> products = FXCollections.*observableArrayList*();  
  
 if (category == 0){  
 // for the stacked products  
 Iterator<Product> iter = products\_stack.iterator();  
 while (iter.hasNext()){  
 products.add(iter.next());  
 }  
  
 // for the stacked products  
 Iterator<Product> iter2 = products\_queue.iterator();  
 while (iter2.hasNext()){  
 products.add(iter2.next());  
 }  
  
 // for the stacked products  
 Iterator<Product> iter3 = products\_list.iterator();  
 while (iter3.hasNext()){  
 products.add(iter3.next());  
 }  
  
 } else if (category >= 1 & category <= 4) {  
 Iterator<Product> iter = products\_stack.iterator();  
 while (iter.hasNext()){  
 Product product = iter.next();  
 if (product.category.getValue() == category) products.add(product);  
 }  
 } else if (category >= 5 & category <= 7) {  
 Iterator<Product> iter2 = products\_queue.iterator();  
 while (iter2.hasNext()){  
 Product product = iter2.next();  
 if (product.category.getValue() == category) products.add(product);  
 }  
 } else if (category >= 8 & category <= 11) {  
 Iterator<Product> iter3 = products\_list.iterator();  
 while (iter3.hasNext()){  
 Product product = iter3.next();  
 if (product.category.getValue() == category) products.add(product);  
 }  
 }  
  
  
 return products;  
 }  
  
 // method for javafx collections  
 public ObservableList<Product> dsToObservableList(String type){  
 ObservableList<Product> products = FXCollections.*observableArrayList*();  
  
 if (type == "Stacked") {  
 Iterator<Product> iter = products\_stack.iterator();  
 while (iter.hasNext()){  
 Product product = iter.next();  
 products.add(product);  
 }  
 } else if (type=="Queued") {  
 Iterator<Product> iter2 = products\_queue.iterator();  
 while (iter2.hasNext()){  
 Product product = iter2.next();  
 products.add(product);  
 }  
 } else if (type == "List") {  
 Iterator<Product> iter3 = products\_list.iterator();  
 while (iter3.hasNext()){  
 Product product = iter3.next();  
 products.add(product);  
 }  
 }  
  
  
 return products;  
 }  
}

ADD and DELETE PRODUCTS

package classes;  
  
import javafx.beans.property.SimpleFloatProperty;  
import javafx.beans.property.SimpleIntegerProperty;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
import sample.Main;  
  
import java.sql.\*;  
  
public class Product {  
 public SimpleIntegerProperty id;  
 public SimpleStringProperty name;  
 public SimpleIntegerProperty quantity;  
 public SimpleFloatProperty cost\_price;  
 public SimpleFloatProperty selling\_price;  
 public SimpleFloatProperty gross\_price;  
 public SimpleIntegerProperty category;  
  
 public Product(int id, String name, int quantity, float cost\_price, float selling\_price, float gross\_price, int cat) {  
 this.id = new SimpleIntegerProperty(id);  
 this.name = new SimpleStringProperty(name);  
 this.quantity = new SimpleIntegerProperty(quantity);  
 this.cost\_price = new SimpleFloatProperty(cost\_price);  
 this.selling\_price = new SimpleFloatProperty(selling\_price);  
 this.gross\_price = new SimpleFloatProperty(gross\_price);  
 this.category = new SimpleIntegerProperty(cat);  
 }  
  
 public String getName() {  
 return name.get();  
 }  
  
 public int getId() {  
 return id.get();  
 }  
  
 public float getCost\_price() {  
 return cost\_price.get();  
 }  
  
 public float getSelling\_price() {  
 return selling\_price.get();  
 }  
  
 public float getGross\_price() {  
 return gross\_price.get();  
 }  
  
 public int getQuantity() {  
 return quantity.get();  
 }  
  
 public int getCategory() {  
 return category.get();  
 }  
  
 // get persistent data from database  
 public static int getProducts() {  
  
 // list of goods from db  
// ObservableList<Product> products = FXCollections.observableArrayList();  
  
 try {  
 // Get a connection to the database  
 Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dsainventory", "root", "prince");  
  
 // SQL statement  
 Statement stmt=con.createStatement();  
  
 ResultSet rs = stmt.executeQuery("select \* from product;");  
  
 // process the results  
 while(rs.next()) {  
 System.*out*.println(rs.getInt(1)+" "+rs.getString(2)+" "+rs.getString(3) +" "+rs.getString(4) +" "+rs.getString(6));  
  
 // create a product object  
 int cat\_id = rs.getInt(7);  
 Product prod = new Product(rs.getInt(1), rs.getString(2), rs.getInt(3), rs.getFloat(4), rs.getFloat(5), rs.getFloat(6), cat\_id);  
  
 // add to data structure  
 Main.*inventory*.addProduct(prod);  
 }  
  
 // close mysql db connection  
 con.close();  
  
 return 1;  
  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 return 0;  
 }  
  
  
 }  
  
  
 public static int addProduct(String name, int quantity, float cost\_price, float selling\_price, int category) {  
  
 // Get a connection to the database  
 try {  
 Connection conn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dsainventory", "root", "prince");  
  
 String sql = "INSERT INTO `product` (`name`, `quantity`, `cost\_price`, `selling\_price`, `category`) VALUES (?, ?, ?, ?, ?);";  
 // SQL statement  
// Statement stmt = con.createStatement();  
  
 // Execute SQL query  
// ResultSet rs = stmt.executeQuery("INSERT INTO `category` (`name`, `description`) VALUES (?, ?);\n");  
 PreparedStatement preparedStmt = conn.prepareStatement(sql);  
 preparedStmt.setString (1, name);  
 preparedStmt.setInt (2, quantity);  
 preparedStmt.setFloat (3, cost\_price);  
 preparedStmt.setFloat (4, selling\_price);  
 preparedStmt.setInt (5, category);  
 preparedStmt.execute();  
  
  
 // close mysql db connection  
 conn.close();  
  
 return 1;  
  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 return 0;  
  
 }  
 }  
  
  
 public static int deleteProduct(String name) {  
  
 // Get a connection to the database  
 try {  
 Connection conn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dsainventory", "root", "prince");  
  
 String sql = "DELETE FROM `product` WHERE (`name` = ?);";  
 // SQL statement  
// Statement stmt = con.createStatement();  
  
 // Execute SQL query  
// ResultSet rs = stmt.executeQuery("INSERT INTO `category` (`name`, `description`) VALUES (?, ?);\n");  
 PreparedStatement preparedStmt = conn.prepareStatement(sql);  
 preparedStmt.setString (1, name);  
  
 // close mysql db connection  
 conn.close();  
  
 return 1;  
  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 return 0;  
  
 }  
 }  
}

PRODUCT CATEGORIES

package classes;  
  
import javafx.beans.property.SimpleIntegerProperty;  
import javafx.beans.property.SimpleStringProperty;  
import javafx.collections.FXCollections;  
import javafx.collections.ObservableList;  
  
import java.sql.\*;  
  
public class Category {  
 private SimpleIntegerProperty id;  
 private SimpleStringProperty name;  
 private SimpleStringProperty description;  
  
 public Category(int id, String name, String description) {  
 this.id = new SimpleIntegerProperty(id);  
 this.name = new SimpleStringProperty(name);  
 this.description = new SimpleStringProperty(description);  
 }  
  
 // Getters  
 public int getId() {  
 return id.get();  
 }  
  
 public String getName() {  
 return name.get();  
 }  
  
 public String getDescription() {  
 return description.get();  
 }  
  
 // database  
 public static ObservableList<Category> getCategories() {  
  
 // list of categories  
 ObservableList<Category> categories = FXCollections.*observableArrayList*();  
  
 // Get a connection to the database  
 try {  
 Connection con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dsainventory", "root", "prince");  
  
 // SQL statement  
 Statement stmt = con.createStatement();  
  
 // Execute SQL query  
 ResultSet rs = stmt.executeQuery("select \* from category");  
  
  
 // process the results  
 while (rs.next()) {  
 System.*out*.println("Adding");  
 categories.add(new Category(rs.getInt(1), rs.getString(2), rs.getString(3)));  
 System.*out*.println("Added");  
 }  
  
 // close mysql db connection  
 con.close();  
  
 return categories;  
  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 return categories;  
  
 }  
 }  
  
 public static int addCategory(String name, String description) {  
  
 // Get a connection to the database  
 try {  
 Connection conn = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/dsainventory", "root", "prince");  
  
 String sql = "INSERT INTO `category` (`name`, `description`) VALUES (?, ?);";  
 // SQL statement  
// Statement stmt = con.createStatement();  
  
 // Execute SQL query  
// ResultSet rs = stmt.executeQuery("INSERT INTO `category` (`name`, `description`) VALUES (?, ?);\n");  
 PreparedStatement preparedStmt = conn.prepareStatement(sql);  
 preparedStmt.setString (1, name);  
 preparedStmt.setString (2, description);  
 preparedStmt.execute();  
  
  
 // close mysql db connection  
 conn.close();  
  
 return 1;  
  
 } catch (Exception e) {  
 System.*out*.println(e.getMessage());  
 return 0;  
  
 }  
 }  
}

|  |  |  |  |
| --- | --- | --- | --- |
| Name | ID | Activities | %Contribution |
| Philemon Frimpong Antwi | 10841449 | Group Coordinator | 12.5 |
| Prince Kyeremanteng Samuel | 10841385 | UI Designer | 12.5 |
| Gerald De-Graft Quainoo | 10852326 | Backend Developer | 12.5 |
| Henrietta Armah | 10828643 | Database Developer | 12.5 |
| Richmond Martey | 10826913 | Backend Developer | 12.5 |
| Prince Dimafo Gyan | 10839289 | UI/Backend Developer | 12.5 |
| Sarpong Rebecca Agyapongmaa | 10839868 | Database Designer | 12.5 |
| Bernard Tetteh Djangbah | 10677436 | Backend Developer | 12.5 |

TEAM MEMBERS