

CS6004ES Application Development Individual Coursework

ARGO POS System

|  |  |
| --- | --- |
| **Student Id** | **Name** |
| COL/E-005143 | W.G. Thilina Thushantha |

# 

# Acknowledgement

I would like to thank all those who helped and supported me throughout the completion of this project. First of all, I would like to thank E-Soft Management and Miss. Mahesha Thejani (Program Manager) for rending their valuable support for the fulfillment of my project.

Secondly, I would like to thank my module lecturer, Mr. Thamara. It would have not been possible for me to complete this project without his valuable support and frequent guidance.

**Introduction**

This Documentation is prepared to address the ARGO POS System which is a point of sales system.

The purpose of the inventory management is to save the owner time and encourage more organized and efficient business processes. This will allow the owner to know what stocks are presently available enabling him to manage and organize his business more effectively. The software package should be general purpose for users who have similar inadequacies with their existing inventory control system, allowing the user to manage stocks more efficiently.

The application contains features such as Access restricted logins for Admin and Cashier, creating an invoice for a customer, printing receipt, registering loyalty customer, adding loyalty points to loyalty customer, add/update/delete stocks, View sales, View stocks, view registered customers, Generate reports.

## **Software Development Task**

“ARGO POS” is a point of sales software that can manage day to day sales in a sales outlet.

1. Only administrator can add, update, search and delete item categories and items to item categories.
2. Administrator should update GRN (Goods Received Note) to maintain stock up-to-date.
3. Cashier can register a loyalty customer and update the customer details. All the customers will be getting unique IDs
4. Cashier can manage sales and system should be able to print a receipt. (1% of loyalty points will be added for each loyalty customer)
5. Administrator can search the customer purchase history.
6. System has the ability to display the total stock, item category stock and item stock and calculate the daily and monthly income of the outlet (only administrator can view income details)
7. System should be able to generate reports like customer details, daily transaction summary, customer transaction summary, etc.
8. System can sort customers according to their loyalty.

# **Why do we use C#**

As far as programming languages go, C# is fairly simple to digest. It's a high-level language, and that means it is somewhat similar to English. It's also designed with ease of use as a priority, and it abstracts away most of the complex tasks like memory management and exception handling, enabling code to learn it without frying their brains.

# **Why did we build Visual Studio Code?**

Visual Studio Code combines the simplicity of a source code editor with powerful developer tooling, like IntelliSense code completion and debugging.

Visual Studio Code supports macOS, Linux, and Windows - so you can hit the ground running, no matter the platform.

# Requirement Specification

## Functional Requirements

* User credentials for Admin and cashier with access restrictions
* Billing the customer purchase
* Invoice calculations (Calculations of total, Quantity \* price)
* Add loyalty points
* Redeem loyalty points
* Register loyalty customer
* Insert, Update and Delete stocks
* Stock Management
* View stocks
* View customer details
* Generate Report
* Read product with QR code reader

## Non-Functional Requirements

* Performance requirement - The system should perform at its best without lagging and less waiting
* Security requirements - All data must be stored safely
* Availability – The system will be available 24/7 as long as the system is running
* Other System Attributes
  + Correctness – should meet the end user(s) objectives
  + Efficiency – achieve the task with minimum number of resources and also with a productive output.
  + Flexibility – System will always welcome new features
  + Integrity – avoid data loses and corruptions.
  + Usability – Provided a user manual to teach users how to use this
  + Maintainability – The system should be maintainable without spending lot of money.

**Software Development Task**

**Why 3-Tier Architecture?**

The solution is tired architecture. If we separate our code by layers then changes in one layer will not affect another very much. Tomorrow if the business demands, we can change the UI very quickly by using existing code.

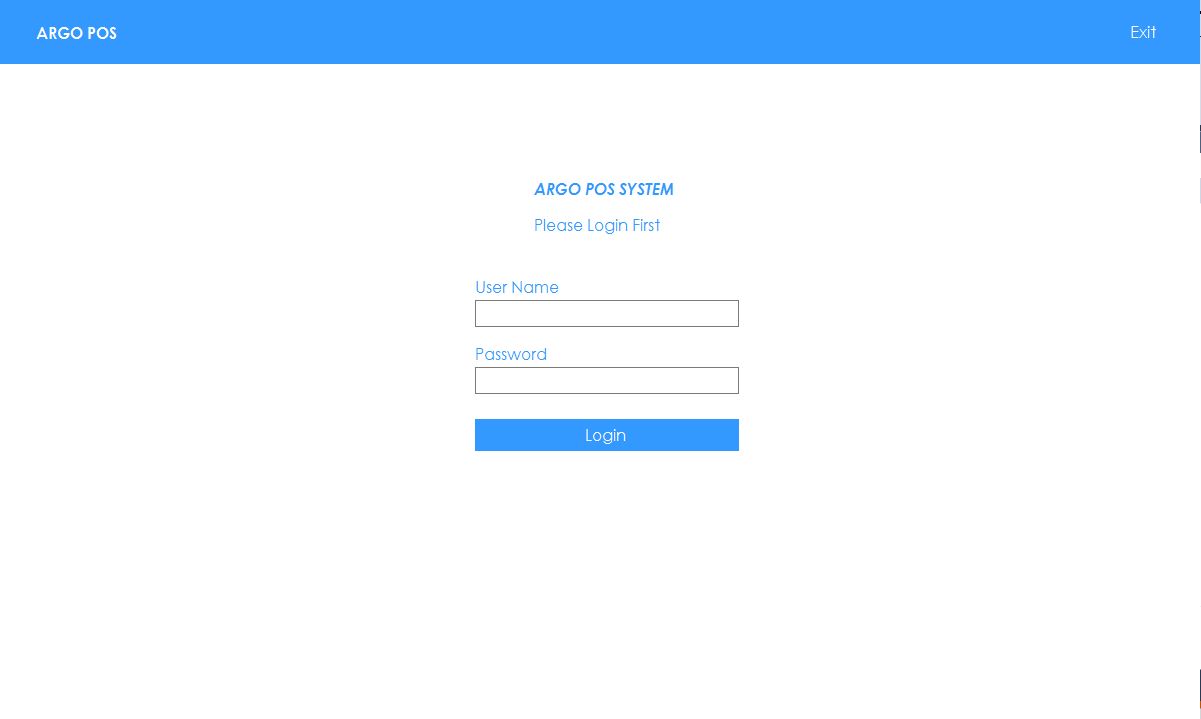
**Presentation Layer/ UI Layer**

This is the top-most layer of the application where the user performs their activity. Let's take the example of any application where the user needs to fill up a form. This form is nothing but the Presentation Layer. In Windows applications Windows Forms are the Presentation Layer belongs to the Presentation Layer. Basically, the user's input validation and rule processing is done in this layer.

**Business Layer**  
This is on top of the Presentation Layer. As the name suggests, most of the business operations are performed here. For example, after collecting form data we want to validate them with our custom business rule. Basically, we define classes and business entities in this layer.

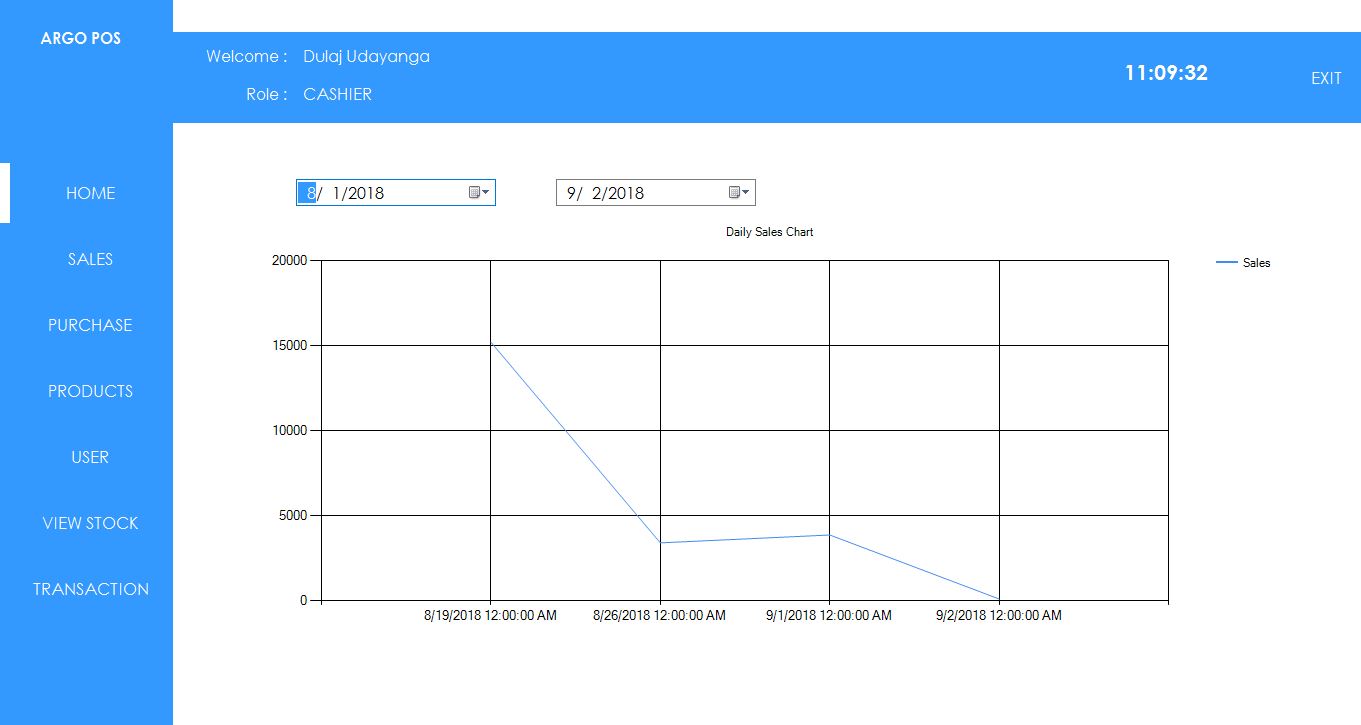
**Data Access Layer**  
On top of the Business Logic Layer is the Data Access Layer. It contains methods that help the business layer to connect with the database and perform CRUD operations. Generally, all database related code and stuff belongs to the Data Access Layer. Sometimes people use a platform-independent Data Access Layer to fetch data from various database vendors.

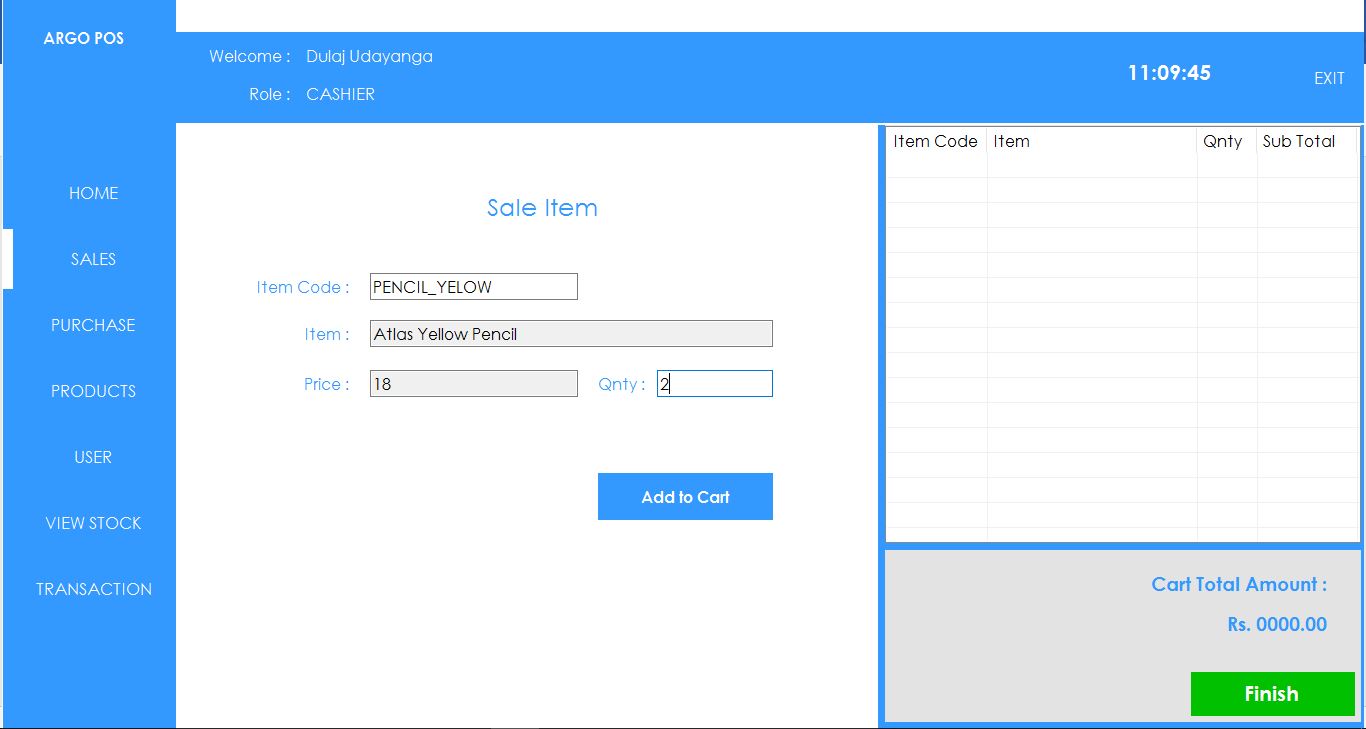
**INTERFACES**

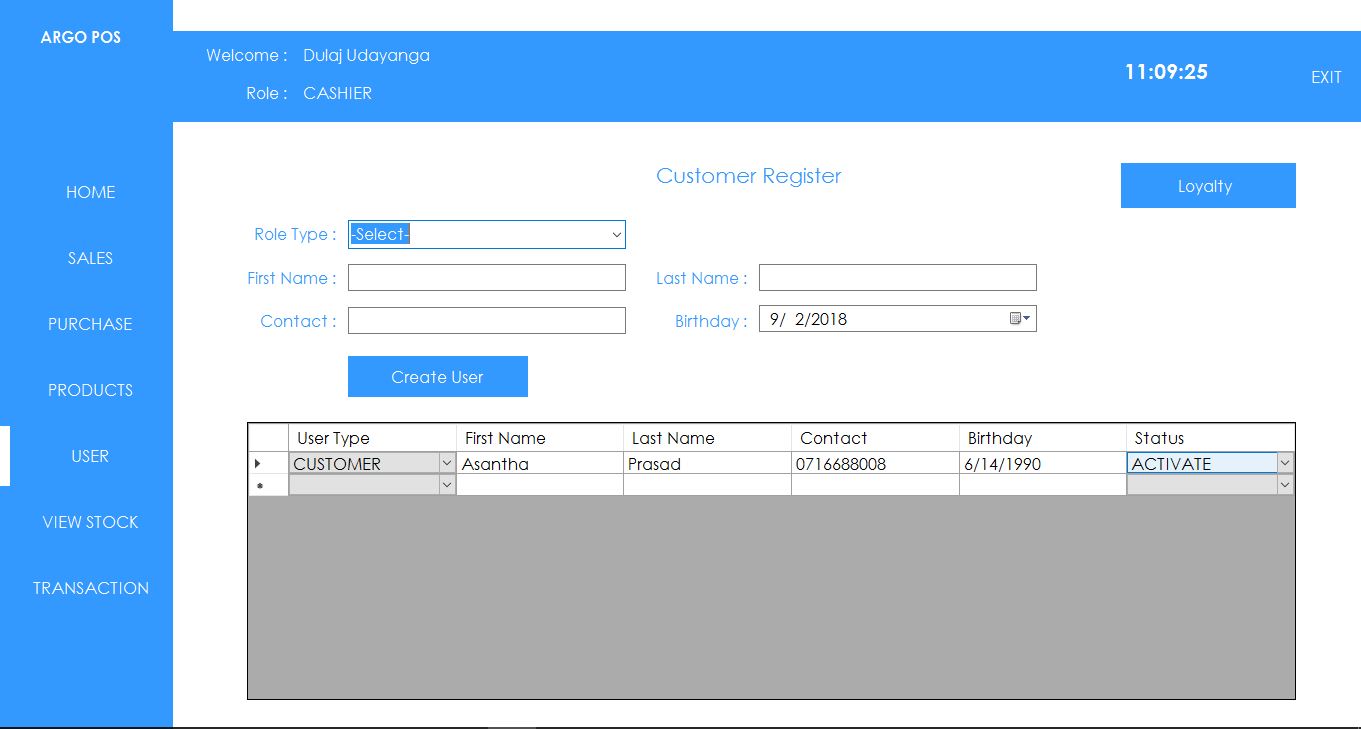
**LOGIN VIEW**

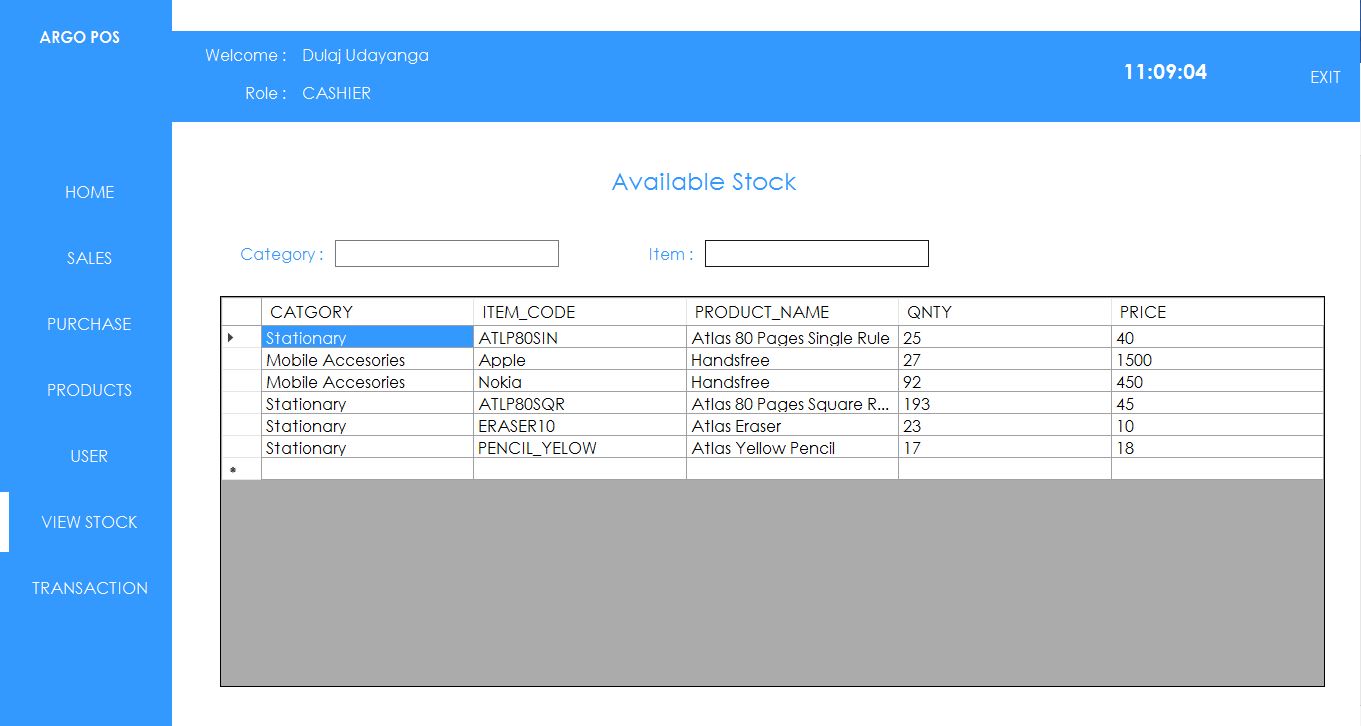
Required : Valid User Name and Password

HOME VIEW

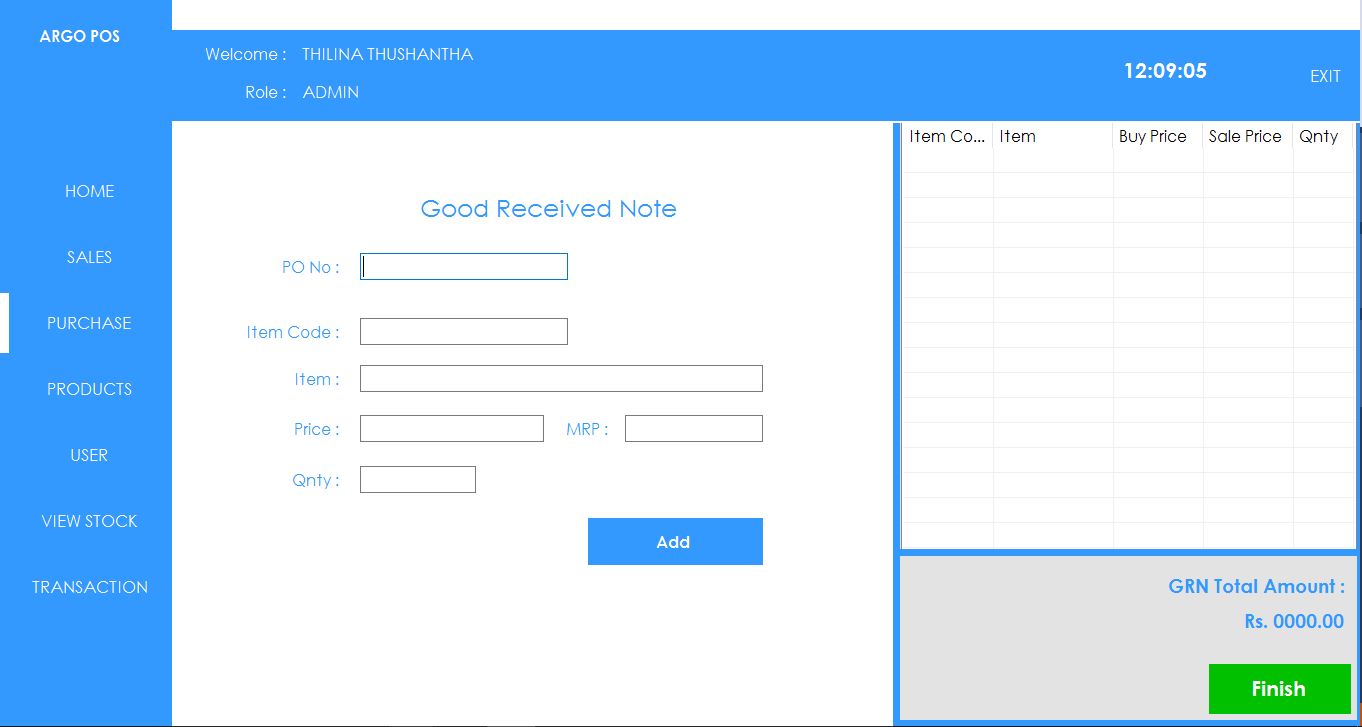


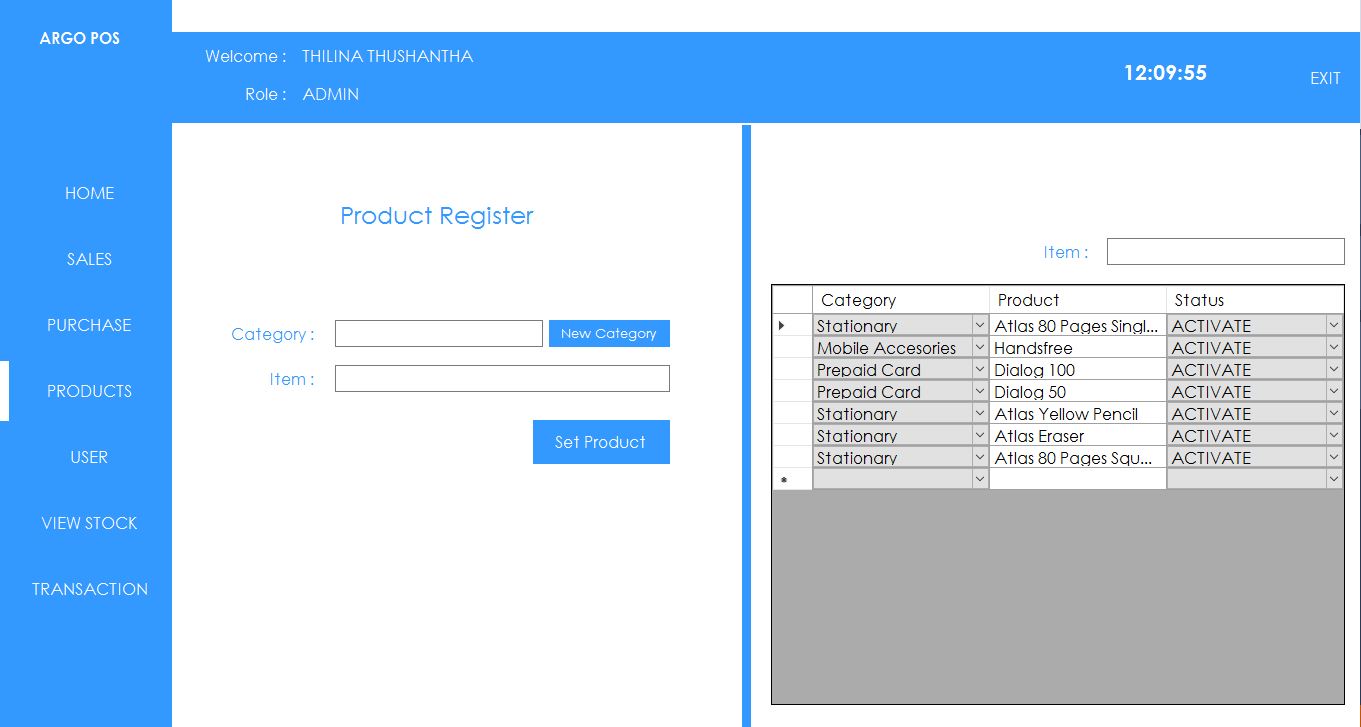
**SALES VIEW**

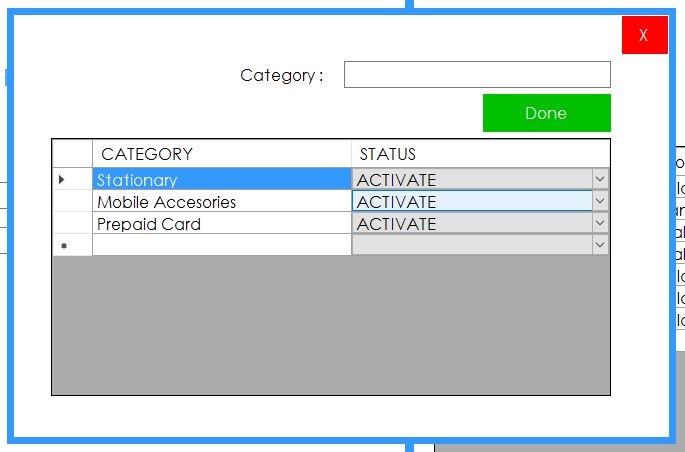
**USER VIEW**

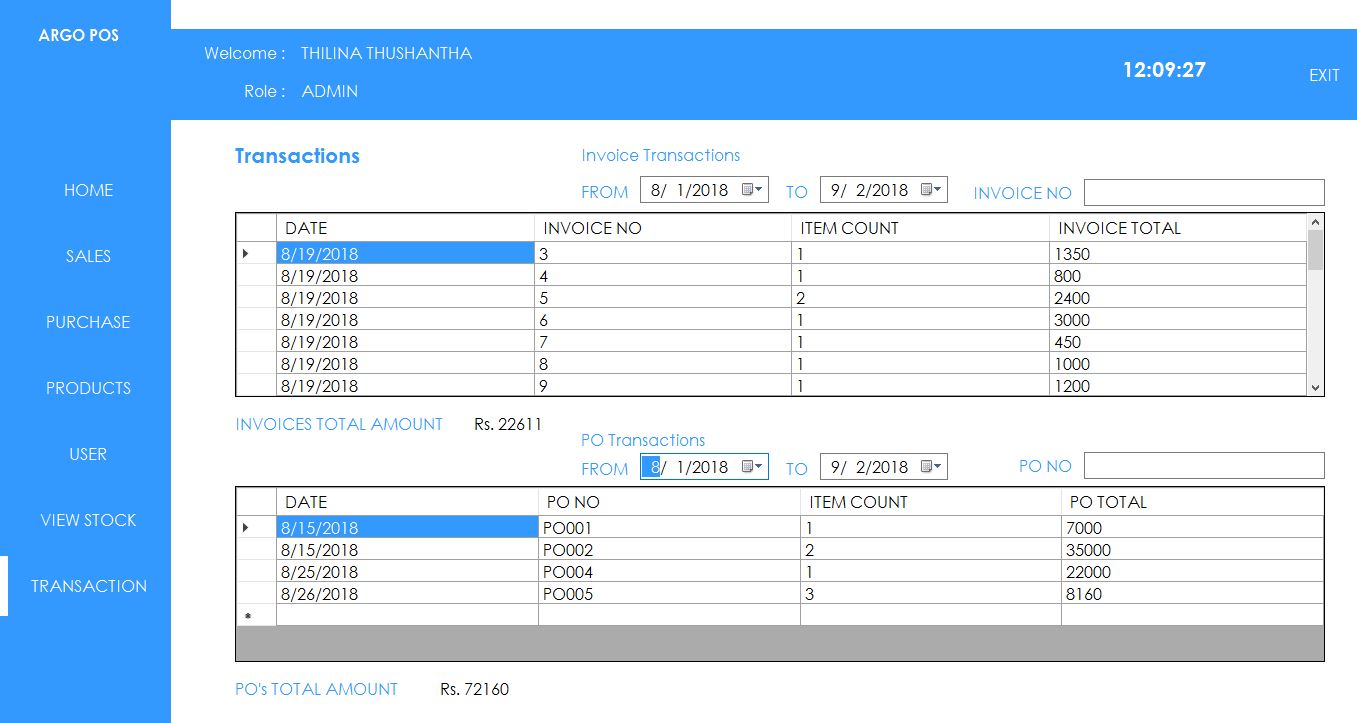
**VIEW STOCK VIEW**

**ADMIN ACCESS VIEWS**

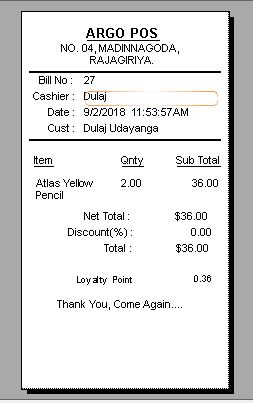
**PURCHASING VIEW**

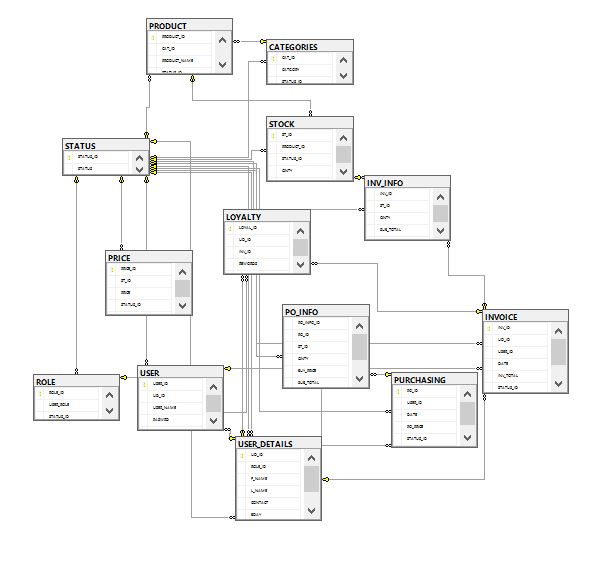
**PRODUCT VIEW**

**CATEGORIES VIEW**

**TRANSACTION VIEW**

**BILL PREVIEW**

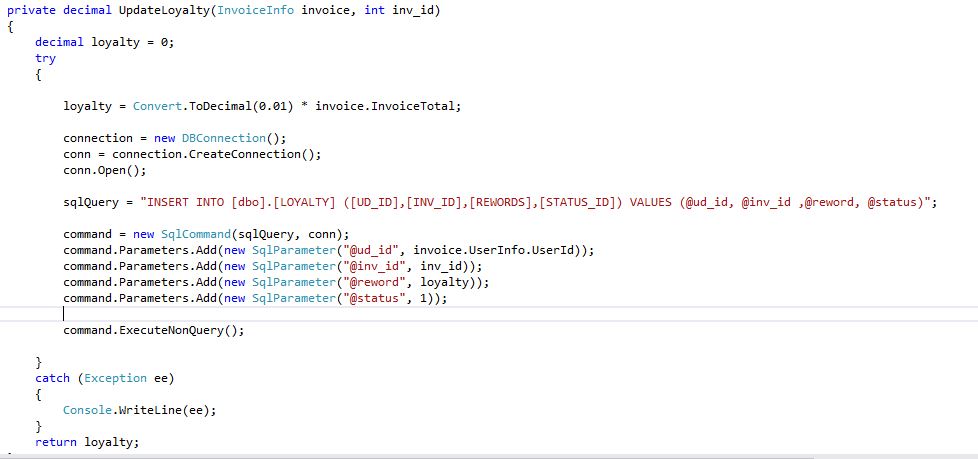


**Table Diagram**

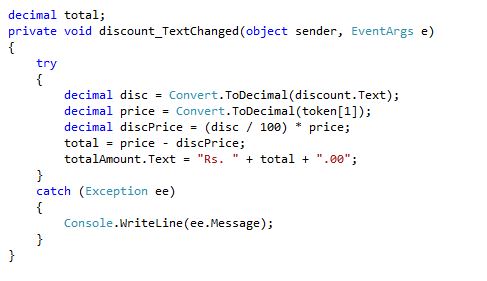
**Algorithm to sort the Customers According to Their Loyalty**

****

**Loyalty Calculation**



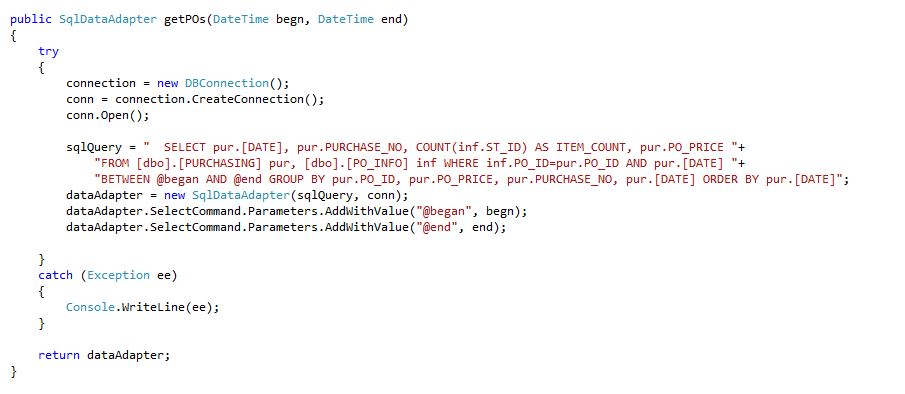
**Discount Calculate Algorithm**

****

**Transaction Calculate Algorithm**

**Invoice Transaction**

**PO Transaction**

****