Subject No. 12

An online supermarket management system enables management staff, warehouse staff, saler and customer to use:

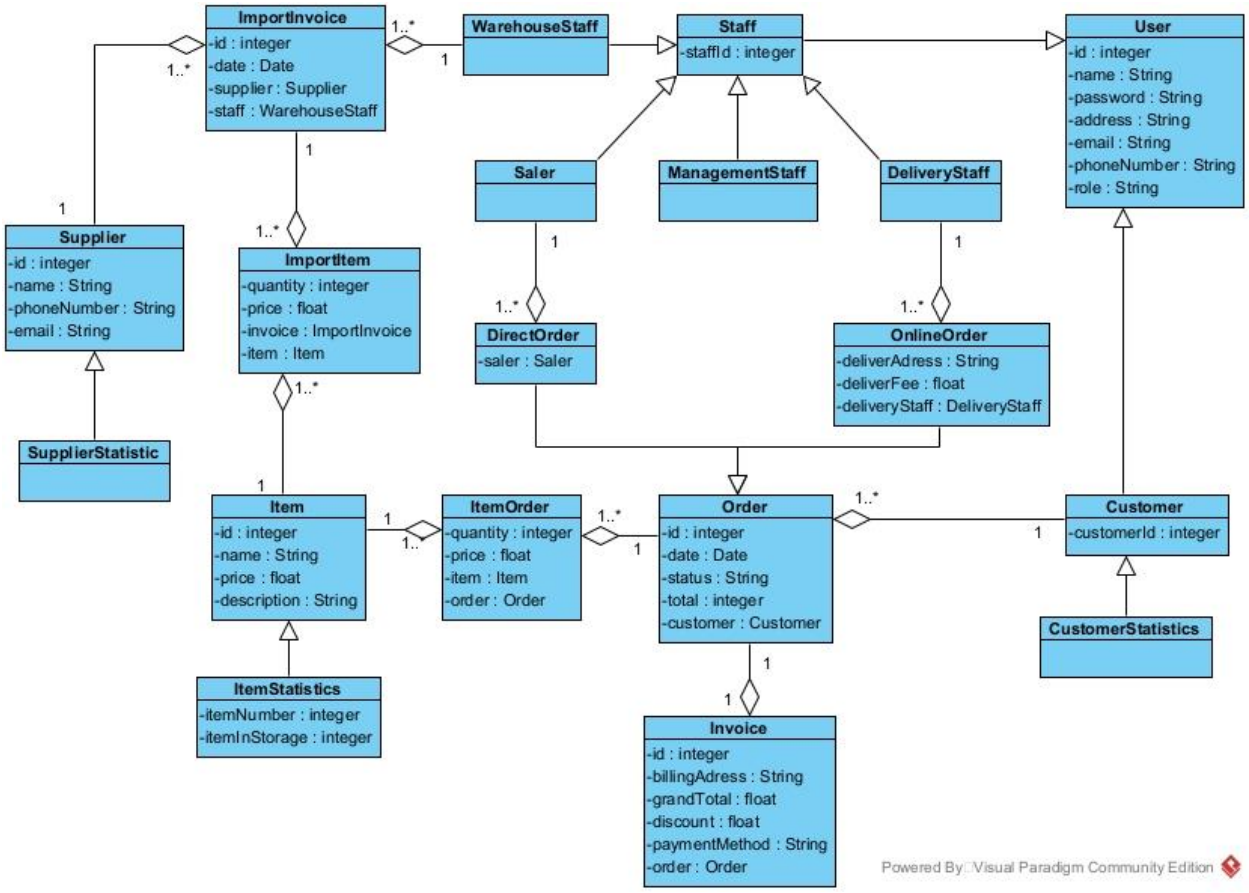
- Management staff: view the types of statistics: items, suppliers and revenue.
- Warehouse staff: import goods from suppliers, manage (add, delete, change) item information and supplier information, browse online orders and export to delivery staff.
- Saler: selling goods at the counter to customers
- Customer: register as a member, search, order online, buy directly at the counter

Considering two modules and answering questions:

- ***Customer registers as a member:*** selects the member registration menu → enters customer information, clicks add → the system saves to the database.
- ***Management staff views customer statistics***: selects the menu to view statistical reports → chooses to view customer statistics by revenue → selects the start and end date → Views statistics of customers → clicks on a customer to view details → views list of related transactions.

Entity class diagram

- Step 1: Add id attribute to all class that is not inherit from other class
- Step 2: Add attribute type for all attribute. Attribute type must compatible with the chosen programming language
- Step 3: Change all association relationship to aggregation/composition
    - ImportInvoice associate Item create ImportItem: Change to ImportItem include ImportInvoice and Item
    - Item associate Order create ItemOrder: Change to ItemOrder include Item and Order
- Step 4: Add class atribute to entity class
    - ImportInvoice has Supplier, WarehouseStaff
    - ImportItem has ImportInvoice, Item
    - DirectOrder has Saler
    - OnlineOrder has DeliveryStaff
    - ItemOrder has Item, Order
    - Order has Customer
    - Invoice has Order



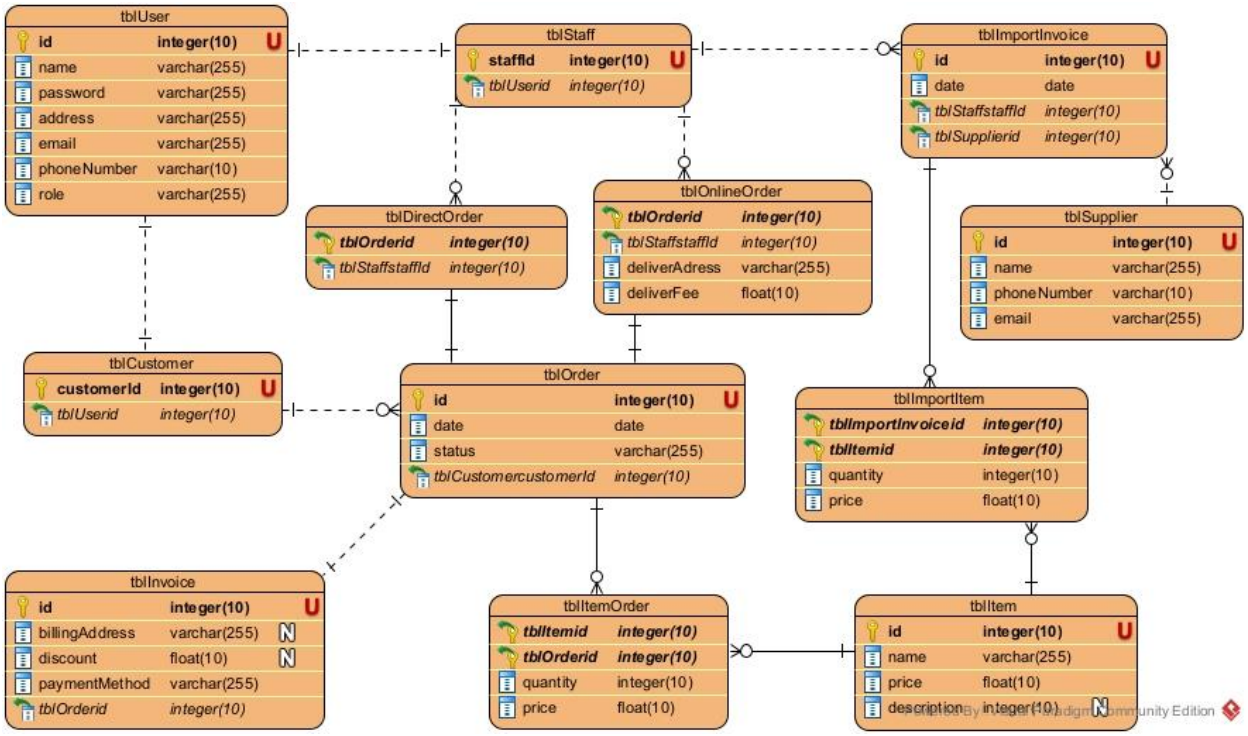
Database

- Step 1: For each class entity, create a table
    - User: tblUser
    - Staff: tblStaff
    - Customer: tblCustomer
    - Order: tblOrder
    - DirectOrder: tblDirectOrder
    - OnlineOrder: tblOnlineOrder
    - ImportInvoice: tblImportInvoice
    - Invoice: tblInvoice
    - Item: tblItem
    - Supplier: tblSupplier
    - ImportItem: tblImportItem
- Step 2: For each entity class, take all attribute type except entity attribute
    - tblUser:
        - id: integer(10)
        - name: varchar(255)
        - password: varchar(255)

        - address: varchar(255)
        - email: varchar(255)
        - phoneNumber: varchar(255)
        - role: varchar(255)
    - tblStaff:
        - staffId: integer(10)
        - tblUserid: integer(10)
    - tblCustomer:
        - customerId: integer(10)
        - tblUserid: integer(10)
    - tblOrder:
        - id: integer(10)
        - date: date
        - status: varchar(255)
        - total: float(10)
        - tblCustomercustomerId: integer(10)
    - tblInvoice:
        - id: integer(10)
        - billingAddress: varchar(255)
        - discount: float(10)
        - grandTotal: float(10)
        - paymentMethod: varchar(255)
        - tblOrderid: integer(10)
    - tblItemOrder
        - quantity: integer(10)
        - price: float(10)
        - tblItemid: integer(10)
        - tblOrderid: integer(10)
    - tblItem
        - id: integer(10)
        - name: varchar(255)
        - price: float(10)
        - description: varchar(255)
    - tblDirectOrder
        - tblStaffstaffId: integer(10)
        - tblOrderid: integer(10)
    - tblOnlineOrder
        - tblStaffstaffId: integer(10)
        - tblOrderid: integer(10)
        - deliverAdress: varchar(255)
        - deliverFee: integer(10)
    - tblImportInvoice
        - id: integer(10)
        - date: date

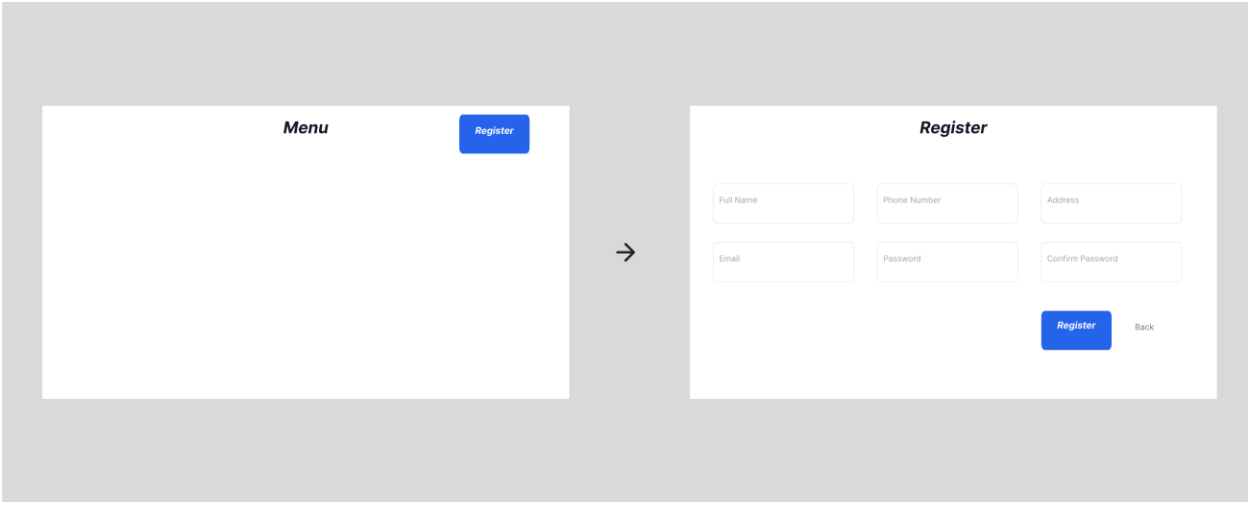
        - tblStaffstaffId: integer(10)
        - tblSupplierid: integer(10)
    - tblImportItem
        - quantity: integer(10)
        - price: float (10)
        - tblItemid: integer(10)
        - tblImportInvoiceid: integer(10)
    - tblSupplier
        - id: integer(10)
        - name: varchar(255)
        - phoneNumber: varchar(10)
        - email: varchar(255)
- Step 3: Consider quantity relationship between classes:
    - tblUser – tblStaff: 1 – 1
    - tblUser – tblCustomer: 1 – 1
    - tblStaff – tblDirectOrder: 1 – n
    - tblStaff – tblOnlineOrder: 1 – n
    - tblDirectOrder – tblOrder: 1 – 1
    - tblOnlineOrder – tblOrder: 1 – 1
    - tblCustomer – tblOrder: 1 – n
    - tblOrder – tblInvoice: 1 – 1
    - tblOrder – tblItemOrder: 1 – n
    - tblItem – tblItemOrder: 1 – n
    - tblStaff – tblImportInvoice: 1 – n
    - tblSupplier – tblImportInvoice: 1 – n
    - tblImportInvoice – tblImportItem: 1 – n
    - tblItem – tblImportItem: 1 -n
- Step 4: Add primary key and foreign key for all table
    - tblUser:
        - Primary Key: id
    - tblStaff:
        - Primary Key: staffId
        - Foreign Key: tblUserid → tblUser.id
    - tblCustomer:
        - Primary Key: customerId
        - Foreign Key: tblUserid → tblUser.id
    - tblOrder:
        - Primary Key: id
        - Foreign Key: tblCustomercustomerId → tblCustomer.customerId
    - tblInvoice:
        - Primary Key: id

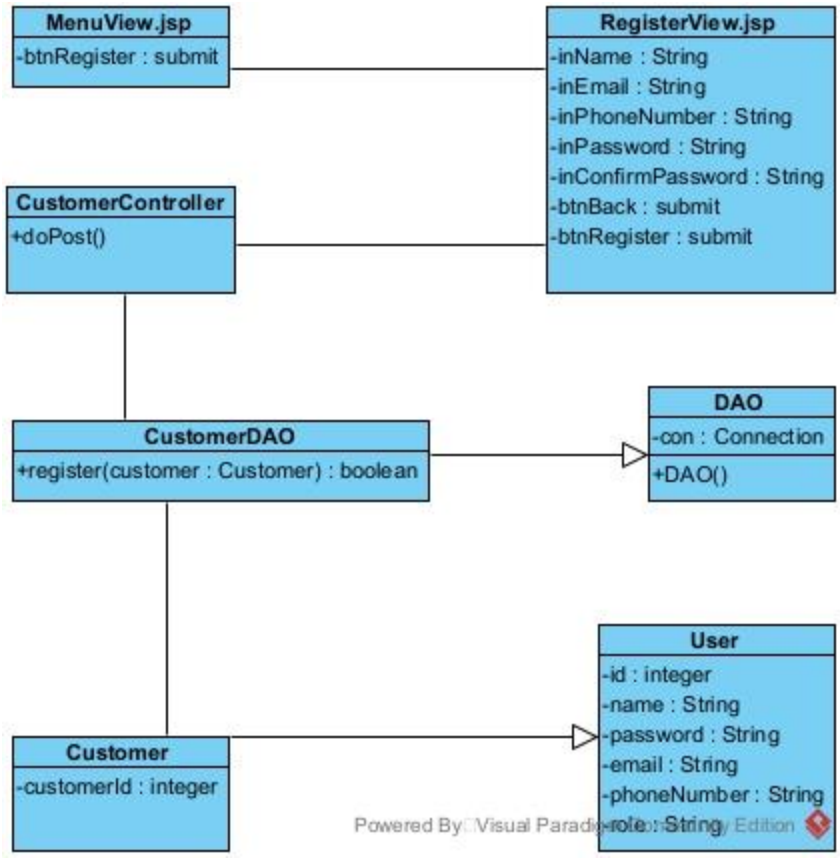
        - Foreign Key: tblOrderid → tblOrder.id
    - tblItemOrder:
        - Primary Key: tblItemid, tblOrderid
            - Foreign Keys: tblItemid → tblItem.id,t blOrderid → tblOrder.id
    - tblItem:
        - Primary Key: id
    - tblDirectOrder:
        - Primary Key: tblOrderid
        - Foreign Keys: tblStaffstaffId → tblStaff.staffId, tblOrderid → tblOrder.id
    - tblOnlineOrder:
        - Primary Key: tblOrderid
        - Foreign Keys: tblStaffstaffId → tblStaff.staffId, tblOrderid → tblOrder.id
    - tblImportInvoice:
        - Primary Key: id
        - Foreign Keys: tblStaffstaffId → tblStaff.staffId, tblSupplierid → tblSupplier.id
    - tblImportItem:
        - Primary Key: tblItemid, tblImportInvoiceid
        - Foreign Keys: tblItemid → tblItem.id, tblImportInvoiceid → tblImportInvoice.id
    - tblSupplier
        - Primary Key: id
- Step 5: Remove abundant attribute
    - Attribute total in order
    - Attribute grandTotal in invoice



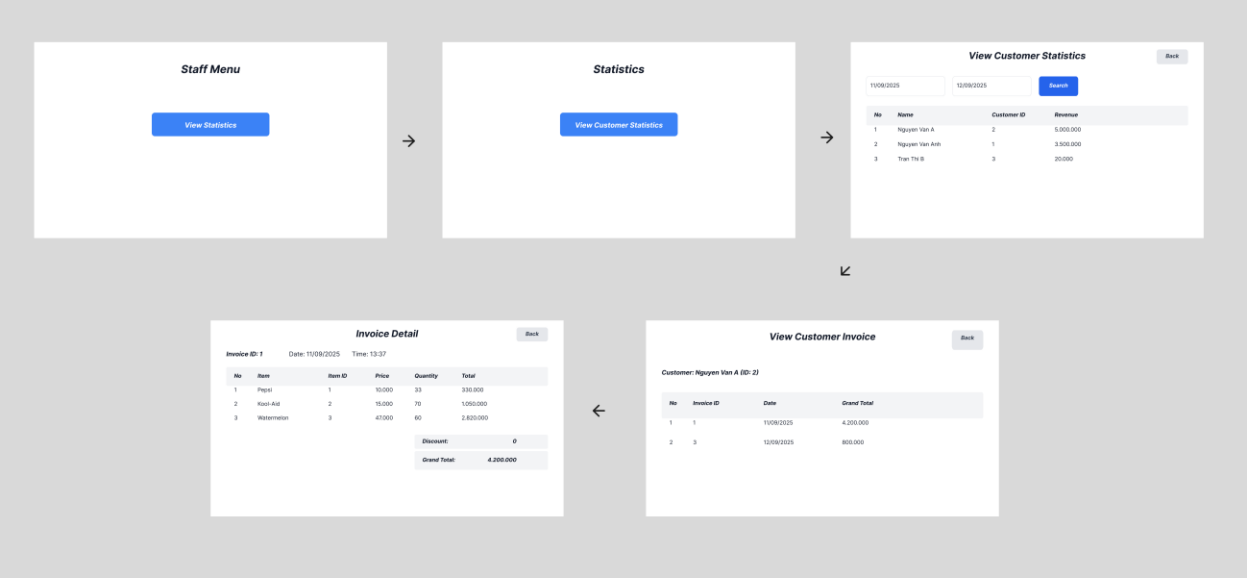
Class diagram

Module register as a member



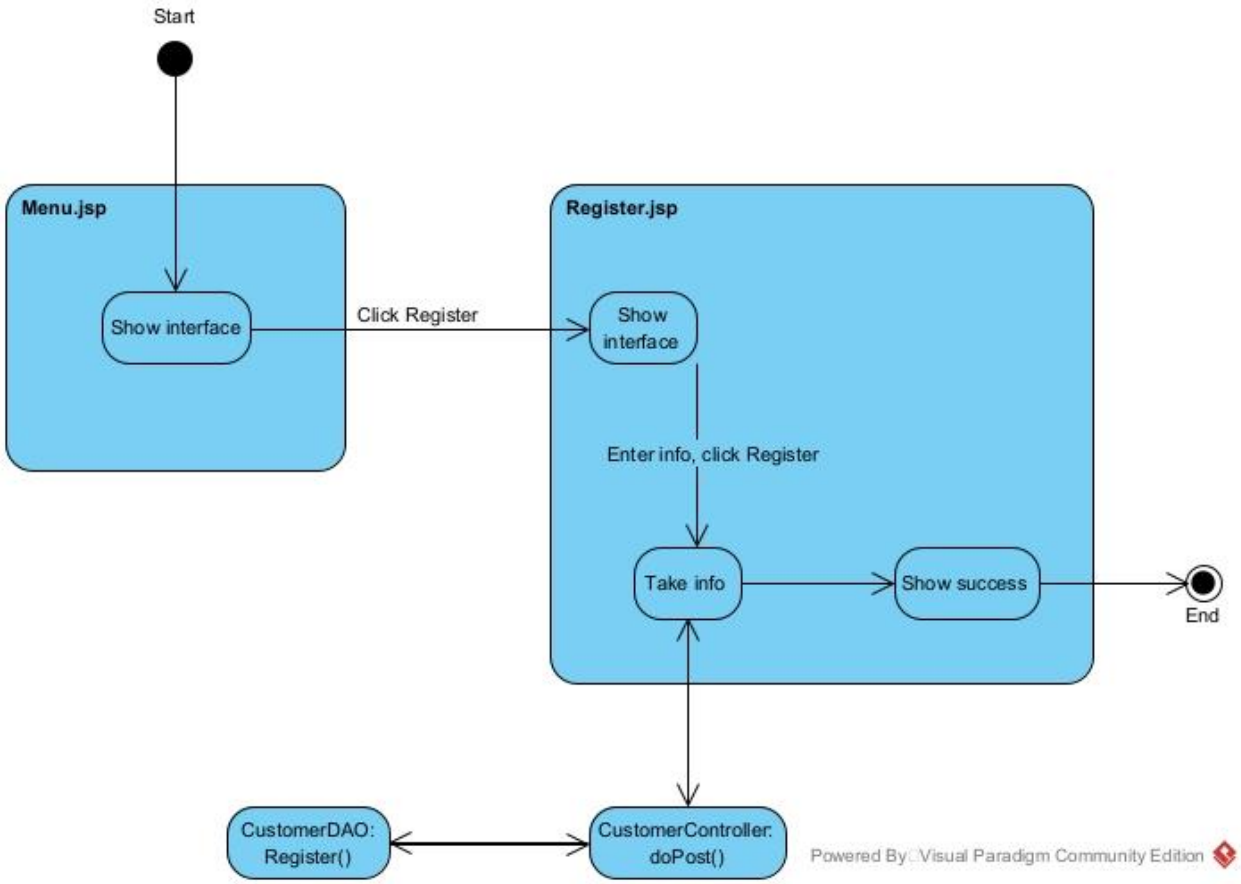


Module view customer statistics by revenue

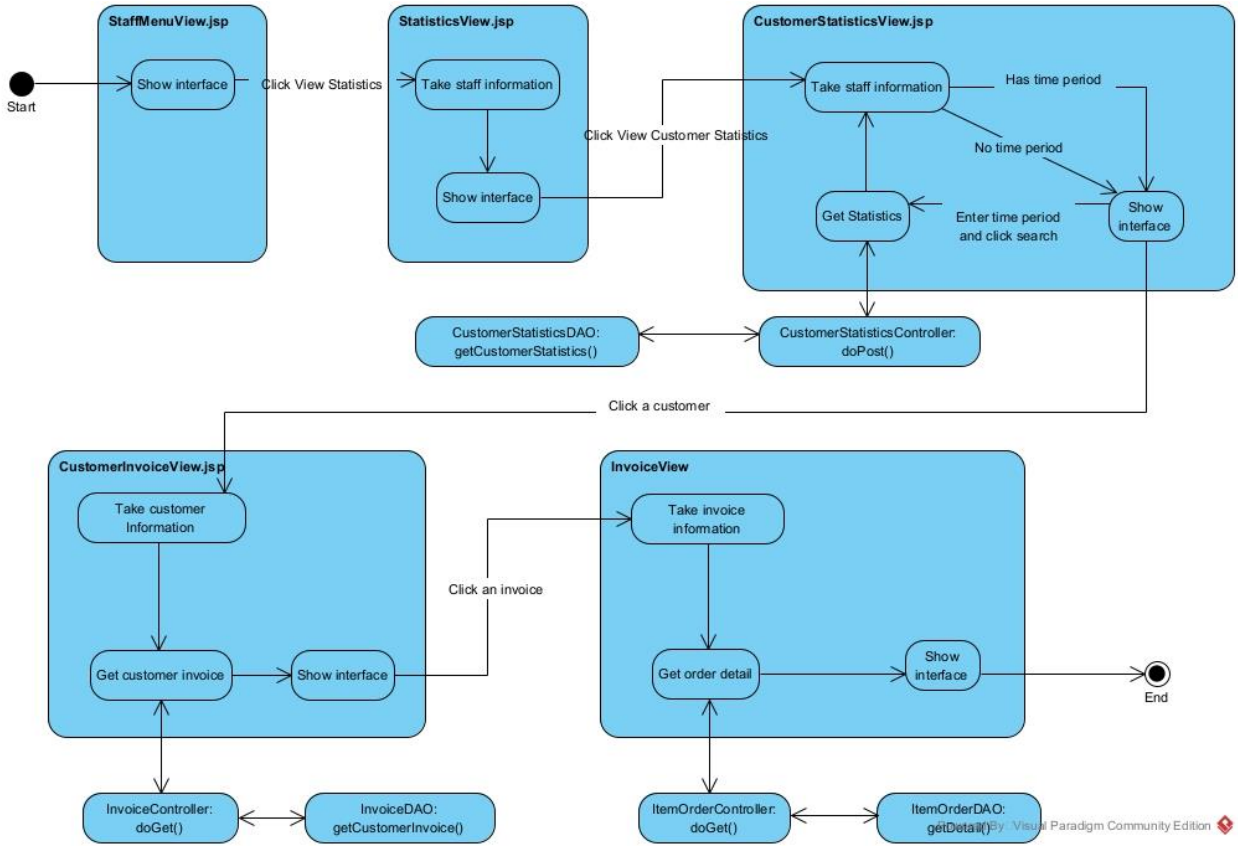


Activity diagram

Module register as a member



Module view customer statistics by revenue



Scenario 3.0

Module register as customer

1. In main menu interface, customer click register
2. The interface MenuView.jsp call the interface RegisterView.jsp
3. The interface RegisterView.jsp display itself to the customer
4. The customer input information and click register
5. The interface RegisterView.jsp call the class CustomerController
6. The class CustomerController execute doPost()
7. The class CustomerController call the class Customer
8. The class Customer pack the information to a Customer object
9. The class Customer return the Customer object to the class CustomerController
10. The class CustomerController call the class CustomerDAO
11. The class CustomerDAO execute register()
12. The class CustomerDAO return the result to the class CustomerController
13. The class CustomerController return the result to the interface RegisterView.jsp
14. Class RegisterView.jsp display success message

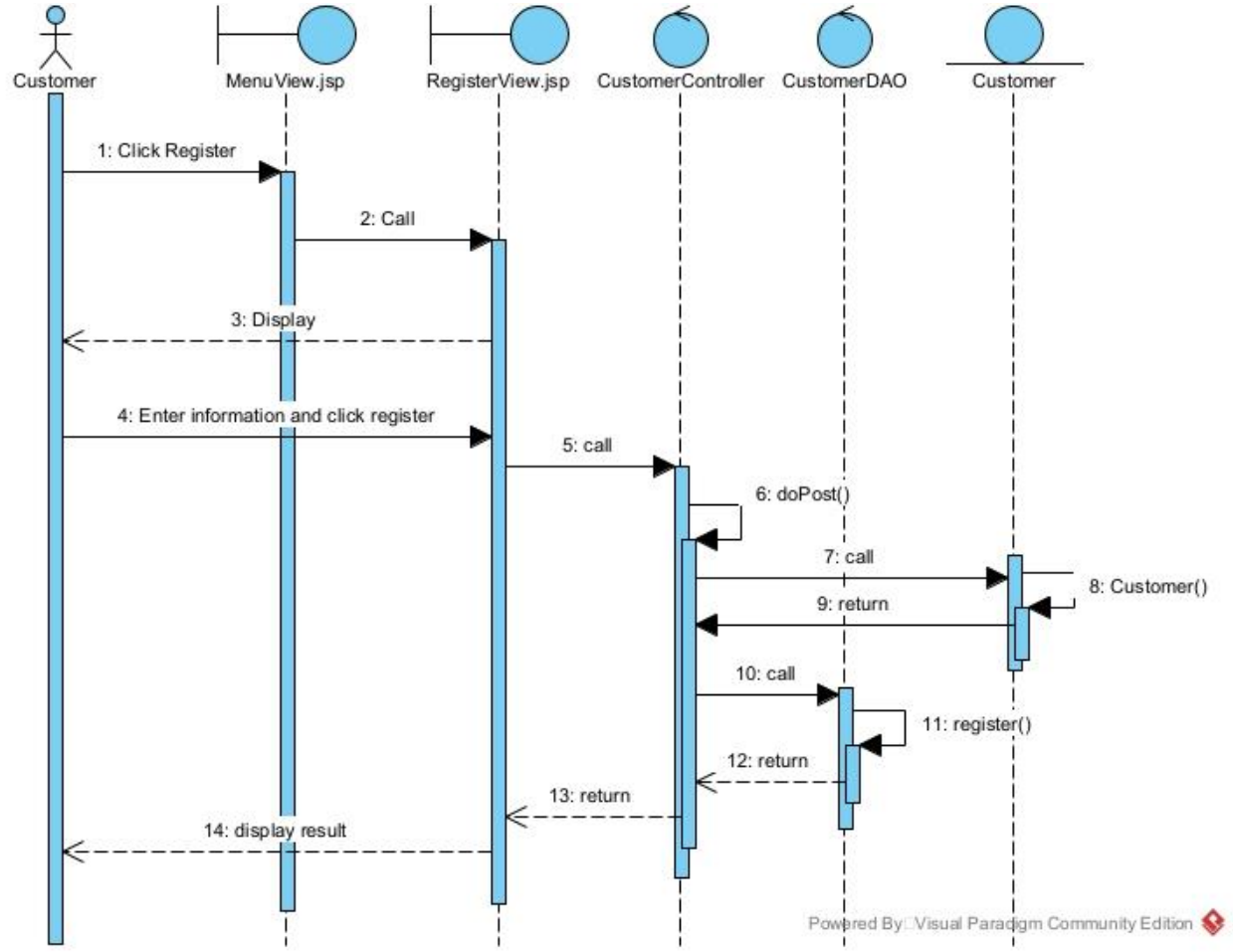
Module view customer statistics by revenue

1. In staff menu interface, management staff click view statistics
2. The interface StaffMenuView.jsp call the interface StatisticsView.jsp
3. The interface StatisticsView.jsp display to the management staff
4. The staff click view customer statistics by revenue
5. The interface StatisticsView.jsp call the interface CustomerStatisticsView.jsp
6. The interface CustomerStatisticsView.jsp display to the management staff
7. The staff enter date and click search
8. The interface CustomerStatisticsView.jsp call the class CustomerStatisticsController
9. The class CustomerStatisticsController execute doPost()
10. The class CustomerStatisticsController call the class CustomerStatisticsDAO
11. The class CustomerStatisticsDAO execute getCustomerStatistics()
12. The class CustomerStatisticsDAO call the class CustomerStatistics
13. The class CustomerStatistics pack the information into a CustomerStatistics object
14. The class CustomerStatistics return the object to the class CustomerStatisticsDAO
15. The class CustomerStatisticsDAO return the result to the class CustomerStatisticsController
16. The class CustomerStatisticsController return the result to the interface CustomerStatisticsView.jsp
17. The interface CustomerStatisticsView.jsp display the result to the management staff
18. The management staff click a customer
19. The interface CustomerStatisticsView.jsp call the interface CustomerInvoiceView.jsp
20. The interface CustomerInvoiceView.jsp call the class InvoiceController
21. The class InvoiceController execute doGet()
22. The class InvoiceController call the class InvoiceDAO
23. The class InvoiceDAO execute getCustomerInvoice()
24. The class InvoiceDAO call the class Invoice
25. The class Invoice pack the information into an Invoice object
26. The class Invoice return the result to the class InvoiceDAO
27. The class InvoiceDAO return the result to the class InvoiceController
28. The class InvoiceController return the result to the interface CustomerInvoiceView.jsp
29. The interface CustomerInvoiceView.jsp display the result
30. The management staff click an invoice
31. The interface CustomerInvoiceView.jsp call the interface InvoiceView.jsp
32. The interface InvoiceView.jsp call the class ItemOrderController
33. The class ItemOrderController execute deGet()
34. The class ItemOrderController call the class ItemOrderDAO
35. The class ItemOrder execute the function getOrderDetail()
36. The class ItemOrder call the class Item
37. The class Item pack the information into an Item object
38. The class Item return the object to the class ItemOrderDAO
39. The class ItemOrderDAO call the class ItemOrder
40. The class ItemOrder pack the information into an ItemOrder object
41. The class ItemOrder return the result to the class ItemOrderDAO
42. The class ItemOrderDAO return the result to the class ItemOrderController

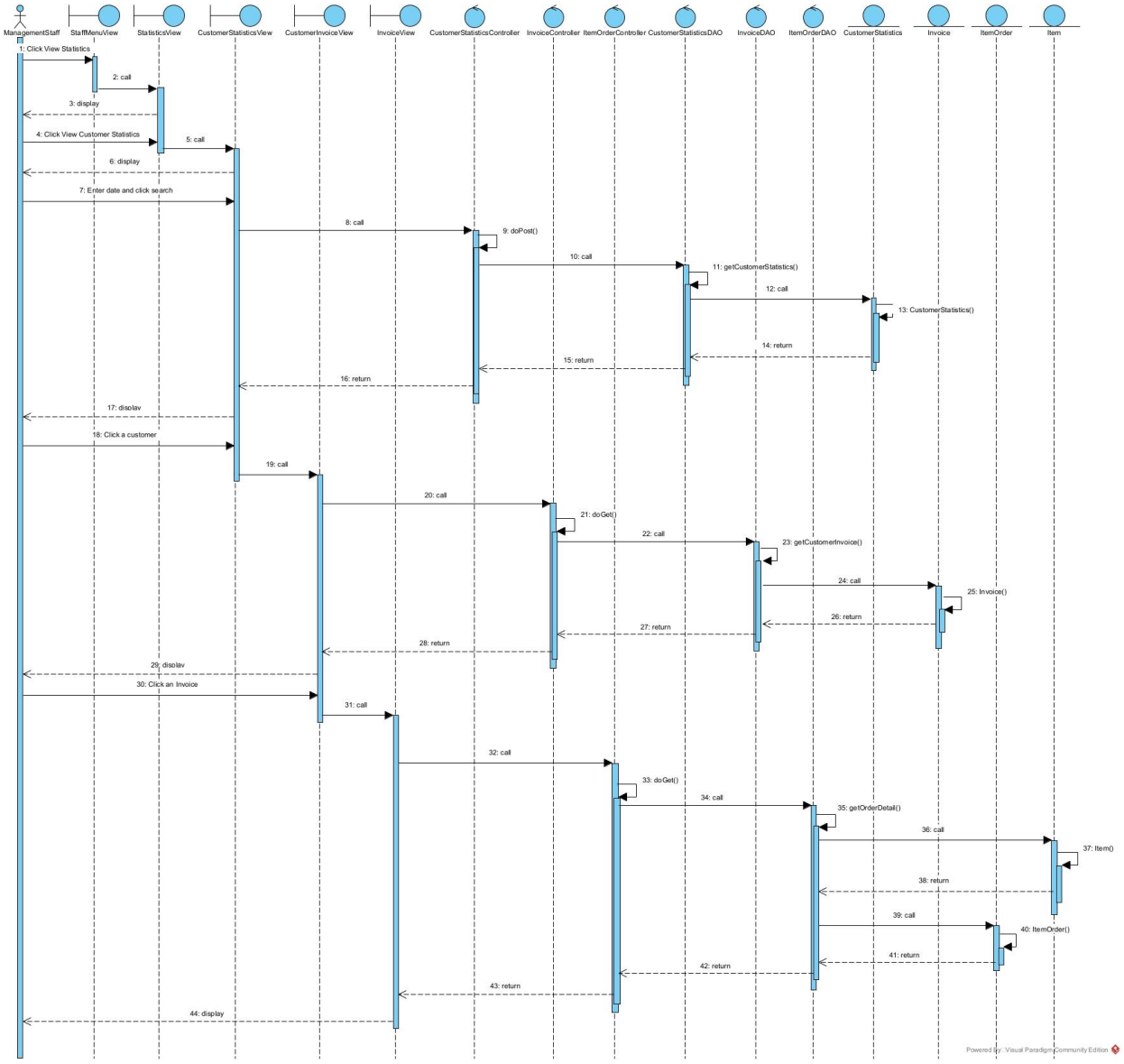
43. The class ItemOrderController return the result to the interface InvoiceView.jsp
44. The interface InvoiceView.jsp display the result to the management staff

Sequence diagram

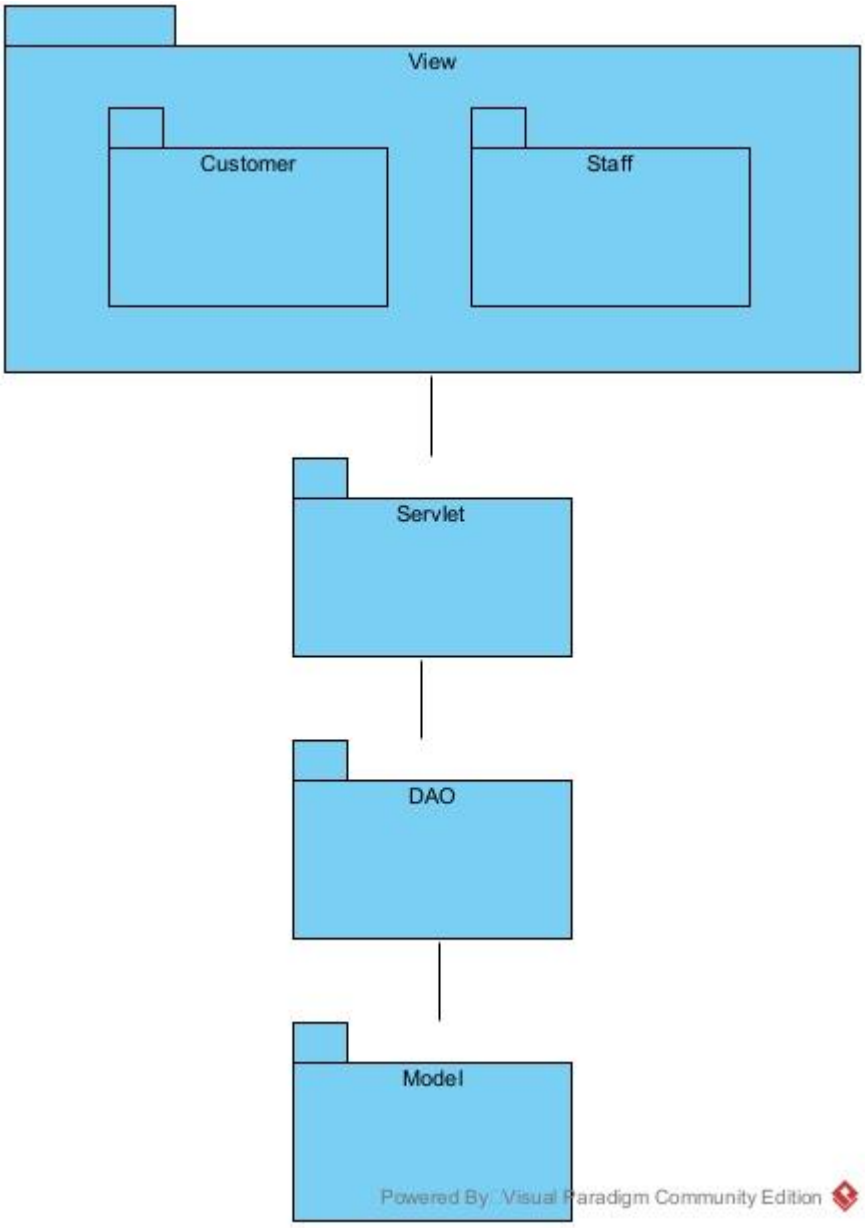
Module register as a member



Module view customer statistics by revenue



Package Diagram



Deployment Diagram