POSTS AND TELECOMMUNICATIONS
INSTITUTE OF TECHNOLOGY

FACULTY OF INFORMATION TECHNOLOGY

REPORT

Introduction to Software Engineering

Class: E22CQCN03-B

Project group: 2

Project name: Store Management

Group members: Lê Tiến Đạt - B22DCAT072
Nguyễn Thế Khiêm - B22DCDT171
Nguyễn Xuân Kiên - B22DCKH062
Đặng Tuấn Phong - B22DCCN612

Module name: Statistic of sub-agent

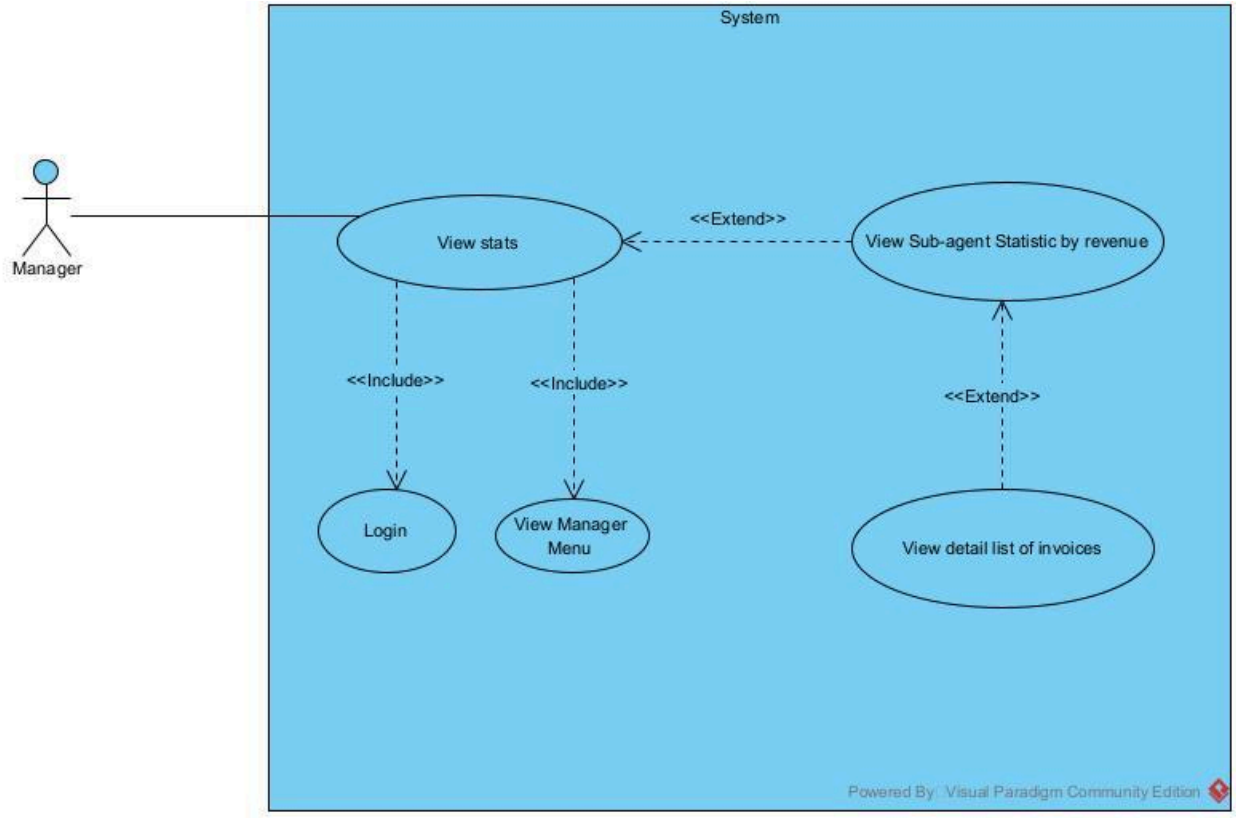
Final Report

Introduction to Software Engineering term project

Topic 4: Store Management

1. UC Diagram

*** Use case Statistic of Sub-agent:**



Describe use case:

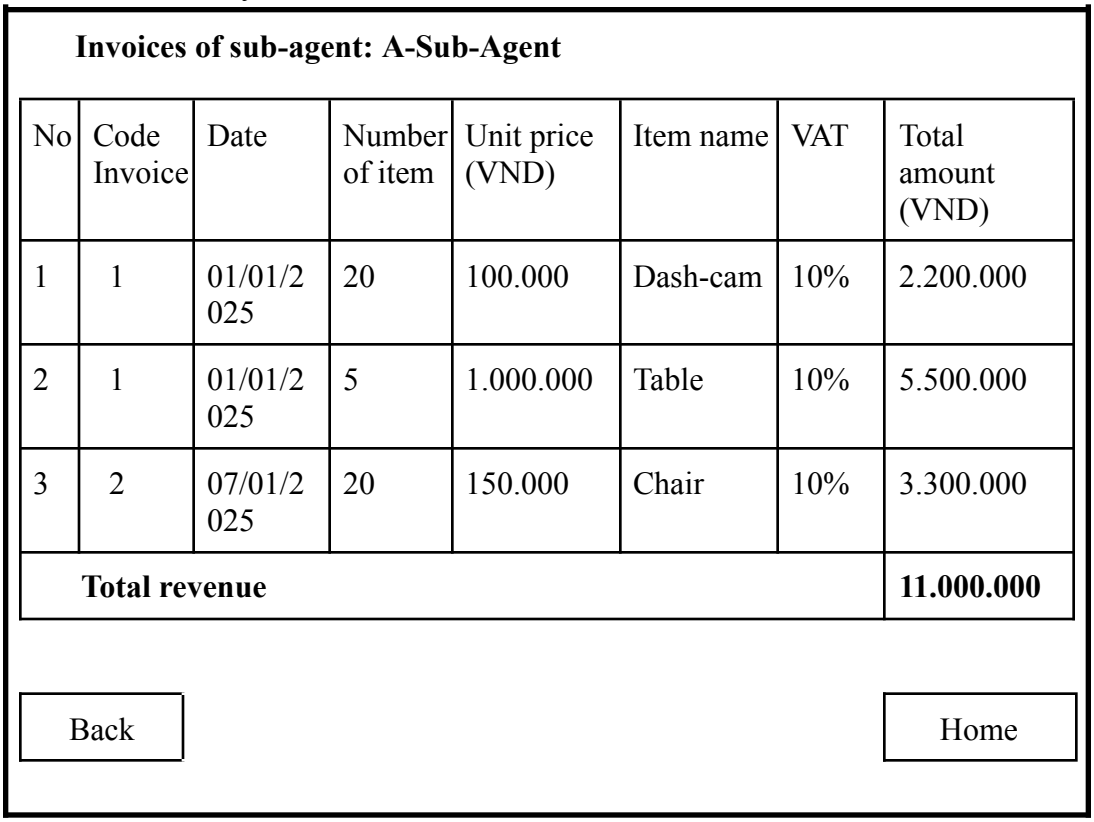
- Login: Allow manager login the system
- View Manager Menu: Allow manager to choose function that manager authorized
- View Statistic menu: Allow manager navigation toView Sub-agent Statistic function
- View Sub-agent Statistic by revenue: Allow manager to view a list of sub-agence in order of the most total revenue to the least in the selected time period
- View details list of invoices: Allow manager to view a detailed list of invoices of each time the sub-agence has imported item

2. Standard and exception scenario

Scenario	**Statistic of Sub-agent by revenue**
Actor	Manager
Pre-conditio	Manager have Manager account

<table>
<tr><td>n</td><td></td></tr>
<tr><td>Post-condition</td><td>- System display table statistic of sub-agent by revenue, each row is corresponds to 1 agent with information: code, name, total revenue
- When manager click into 1 row, system display details list of invoices sorted by date</td></tr>
<tr><td>Main events</td><td>1. A manager starts the application to view details list of invoice of each time the A-sub-agence has imported item between 01/01/2025 and 31/01/2025.
2. The login interface appears with: an input text for username, an input text for password, a button to login.
3. The manager enters username as “manager”, password as “******” and then, clicks on the login button.
4. The main manager menu appears with the option is View statistic
5. The manager chooses to view the statistics.
6. The view statistic UI appears with two options: view statistic of item and view statistic of sub-agent.
7. The manager chooses to view statistic of sub-agent.
8. The view statistic of sub-agent UI appears with: two input text for statistical period(start - end) and a button to search.
9. The manager enters a keyword as “01/01/2025” to the start date input and “31/01/2025” to the end date input, then clicks on the search button.
10. The list of sub-agents in order of the most total revenue to the least in the selected time period and a “Back” button appears as follows.
List of Sub Agent Statistic between:
01/01/2025 31/01/2025 Search
<table><tr><th>Code</th><th>Name</th><th>Address</th><th>Phone number</th><th>Total quantity</th><th>Total revenue (VND)</th></tr><tr><td>01</td><td>A-sub-agent</td><td>Ha Dong, Ha Noi</td><td>0111111111</td><td>45</td><td>11.000.000</td></tr><tr><td>02</td><td>B-sub-agent</td><td>Cau Giay, Ha Noi</td><td>022222222</td><td>50</td><td>4.000.000</td></tr><tr><td>03</td><td>C-sub-agent</td><td>Dong Da, Ha Noi</td><td>03333333</td><td>0</td><td>0</td></tr></table></td></tr>
</table>

<table>
<tr>
<td></td>
<td>


11. The manager clicks into the information row of the sub-agent that has the 01 code.
12. System display the detail list of invoices about items from sub-agent has 01 code sort by time and a “Home” button:


13. Manager staff click “Home” button
14. System comeback to the main menu.</td>
</tr>
<tr>
<td>Exceptions</td>
<td>4. The error alert appears to show that login failed and a “Try again” button.
4.1 Manager staff click to the button
4.2 System comeback to the UI login.
4.3 Manager staff enter the right username and password
4.4 System displays the home menu.

10. System shows the error alert that no invoice of sub-agent found at period and a “OK” button.
10.1 Manager click to the “OK” button</td>
</tr>
</table>

	10.2 System display the sub-agent statistic UI 10.3 manager staff choose again the period that have sub-agent import item. 10.4 System displays a list of sub-agent.

3. Entity class diagram_ Analysis

Step 1: Describe the system functions in a short paragraph

Review the scenario.

Step 2+3: Extract nouns and classify them

- System: abstract noun → reject
- Manager: a type of user → part of User class
- UI/Interface: abstract concept → reject
- Button/Menu: interface elements → reject
- Statistics: data representation → potential class
- Sub-agent: needs to be managed → a class: SubAgent
- Search bar: interface element → reject
- Start date: time parameter → reject
- End date: time parameter → reject
- Period: can be represented by start date and end date → reject
- Error alert: interface element → reject
- List: data structure → reject
- Sub-agent code: identifier → attribute of SubAgent
- Sub-agent name: identifier → attribute of SubAgent
- Total revenue: financial metric → attribute of SubAgentStat
- Invoice/Bill: transaction record → a class: Invoice
- Item: need to be managed → a class: Item
- Bill date: time information → attribute of Invoice
- Total number of items: quantity metric → attribute of Invoice
- Total bill amount: financial metric → attribute of Invoice

From this analysis, we identify these key classes:

- User (with Manager role): fullname, username, password, role
- SubAgent: name, address, phone number, taxNumber
- SubAgentStat: totalRevenue, totalQuantity
- Invoice: invoiceId, date, totalItems, totalAmount
- Item: name, description

Step 4: Consider the quantitative relationship among entity classes

- A User can manage many Invoice, each Invoice can be created by one Managers

=>User - Invoice is a 1:n relationship

- A SubAgent can have multiple statistic records for different time periods, each SubAgentStat belongs to exactly one SubAgent

=>SubAgent - SubAgentStat a 1:n relationship

- A SubAgent can have multiple export bills over time, each Invoice belongs to exactly one SubAgent

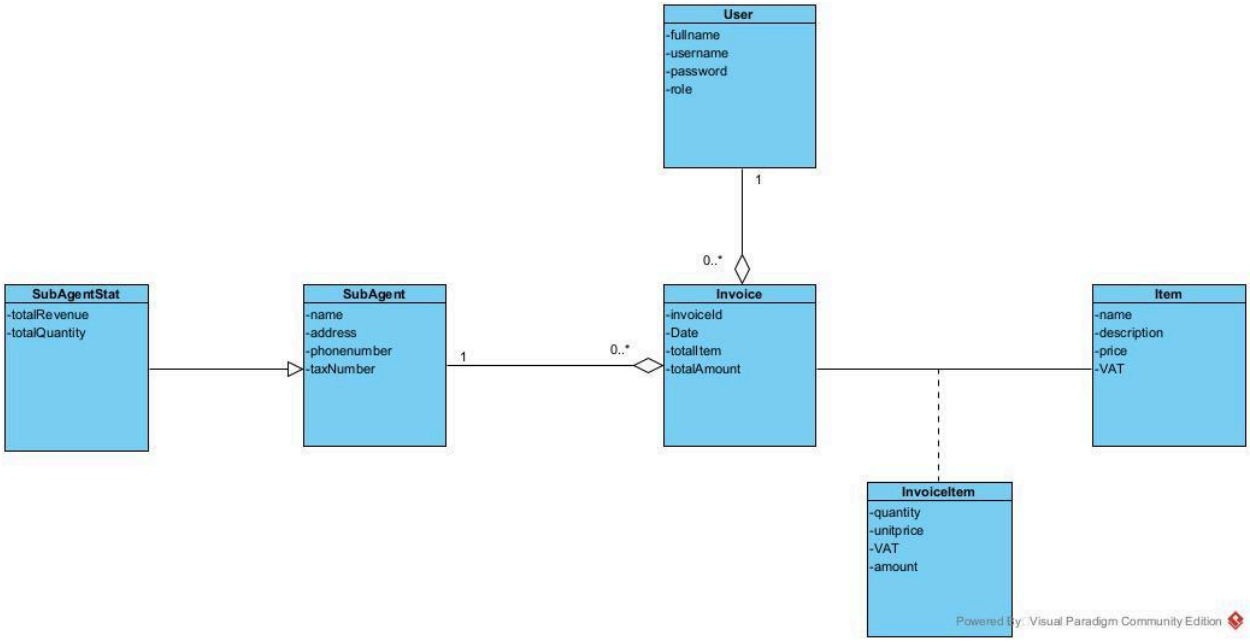
=>SubAgent - Invoice is a 1:n relationship

- Each invoice can contain many items, each item can be in many invoices

=>Invoice - Item is a n:n relationship

=> So we could propose a class between them: InvoiceItem with some information: Date, quantity, unit price

Step 5: Determine the object relationship among entity classes



4. Class diagram_Analysis

- Enter the system -> The login interface is appeared

=> need a class: LoginView

- input for username -> inUsername
- input for password -> inPassword

- a submit to login -> subLogin

- Enter the username/password -> the system must check if the login is correct -> need a method:
- name: checkLogin()
- input: username password (of the class User)
- output: true if user and password correct, else false
- Owner class: User.

Once login is successful -> the main interface of the manager is appeared
=> need a class: ManagerHomeView which has at least:

- An option to choose to view statistics -> subStatisticView.
- An button “Logout” -> subLogout

Choose to view statistic menu -> Display an UI to choose statistic function
=> need a class : StatisticView which have:

- Option to view sub-agent by revenue -> subSaStatView
- Button back to home page -> subBackHome

Choose to view sub-agent statistic by revenue -> appears sub-agent statistic by revenue interface
=> need class: SubAgentStatView that has:

- Button back to view statistic menu : subBackStat
- input text for start date: inStartDate
- input text for end date: inEndDate
- Search button to find the list sub-agent statistic: subSearch
- a result list of sub-agent statistic sort by revenue that can be choose to view detail: outsubSaStat

Enter the start date and end date then click to search button -> System has to search all sub-agent statistic contains in period -> need a method:

- name: searchSaByRev()
- input: start date, end date -> inStartDate, inEndDate
- output: list of sub-agent
- assign to the entity class: SubAgentStat

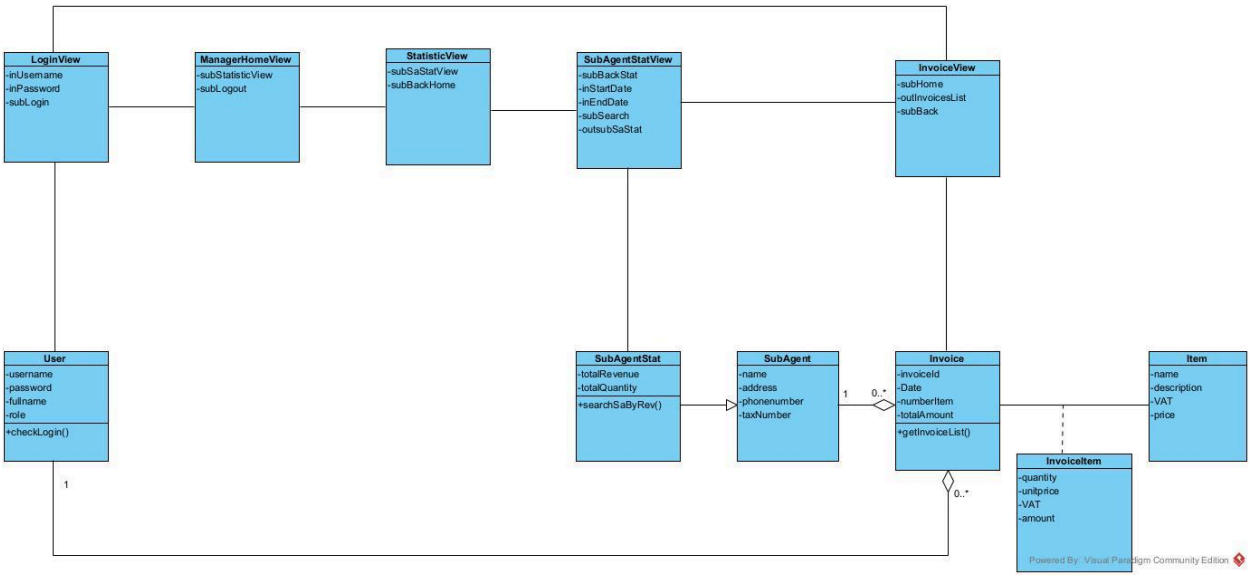
The list of founded sub-agent will be listed in the SubAgentStatView.
Click on a sub-agent to view invoices -> System has to search all invoice of sub-agent in period -> need a method:

- name: getInvoiceList()
- input: SubAgentStat
- output: list of invoice
- assign to the entity class: Invoice

The invoices of sub-agent interface appears
=> Need class: InvoiceView that has:

- Button go to home: subHome
- Button to go back: subBack

- A list of invoices: -> outInvoiceList

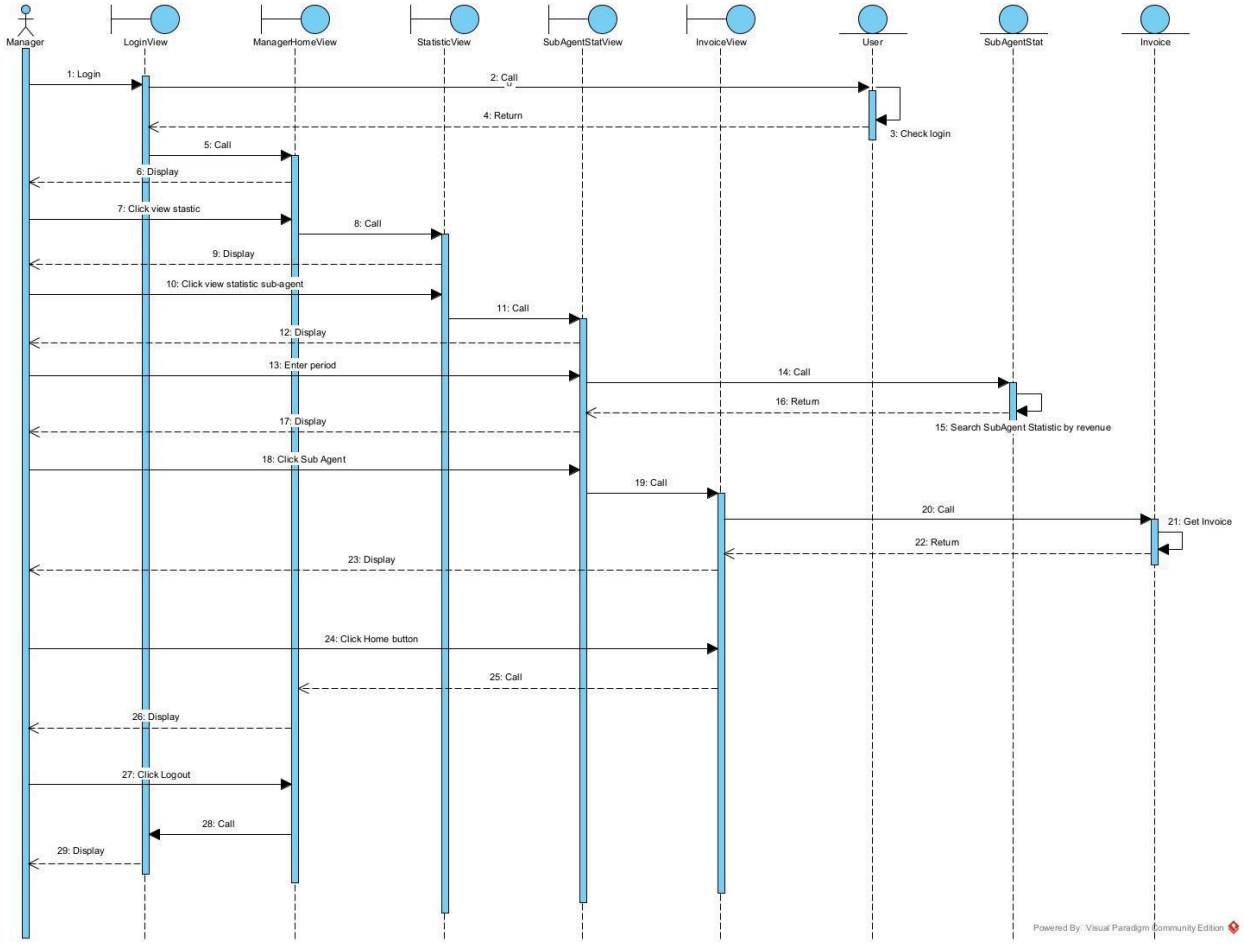


5. Sequence diagram_Analysis

Scenario version2:

1. The manager enters username/password and then clicks on the Login button.
2. The class LoginView calls the class User to process.
3. The class User calls the method checkLogin(). The login is successful.
4. The class User returns the results to the class LoginView.
5. The class LoginView calls the class ManagerHomeView.
6. The class ManagerHomeView displays itself to the manager.
7. The manager chooses the option of view statistics.
8. The class ManagerHomeView calls the class StatisticView.
9. The class StatisticView displays itself to the manager.
10. The manager chooses the option of view statistic of sub-agent.
11. The class StatisticView calls the class SubAgentStatView.
12. The class SubAgentStatView displays itself to the manager.
13. The manager enters a start date, end date and clicks on the search button.
14. The class SubAgentStatView calls the class SubAngetStat to process.
15. The class SubAngetStat calls the method searchSaByRev().
16. The class SubAngetStat returns the results to the class SubAgentStatView.
17. The class SubAgentStatView displays the results to the manager.
18. The manager chooses a sub-agent to view the list of invoices.

19. The class SubAgentStatView calls the class InvoiceView.
20. The class InvoiceView calls the class Invoice to process.
21. The class Invoice calls the method getInvoice().
22. The class Invoice returns the results to the class InvoiceView
23. The class InvoiceView displays the results to the manager.
24. The manager clicks into the Home button.
25. The InvoiceView calls the class ManagerHomeView.
26. The class ManagerHomeView displays itself to the manager.
27. Manager clicks to logout.
28. The class ManagerHomeView calls LoginView
29. The class LoginView displays itself to the manager.



6. Entity class_Design

Step 1: Add the id attribute for the classes which don't inherit from other class: User, SubAgent, Invoice, InovoiceItem, Item

Step 2: Add the type of each attribute in all classes:

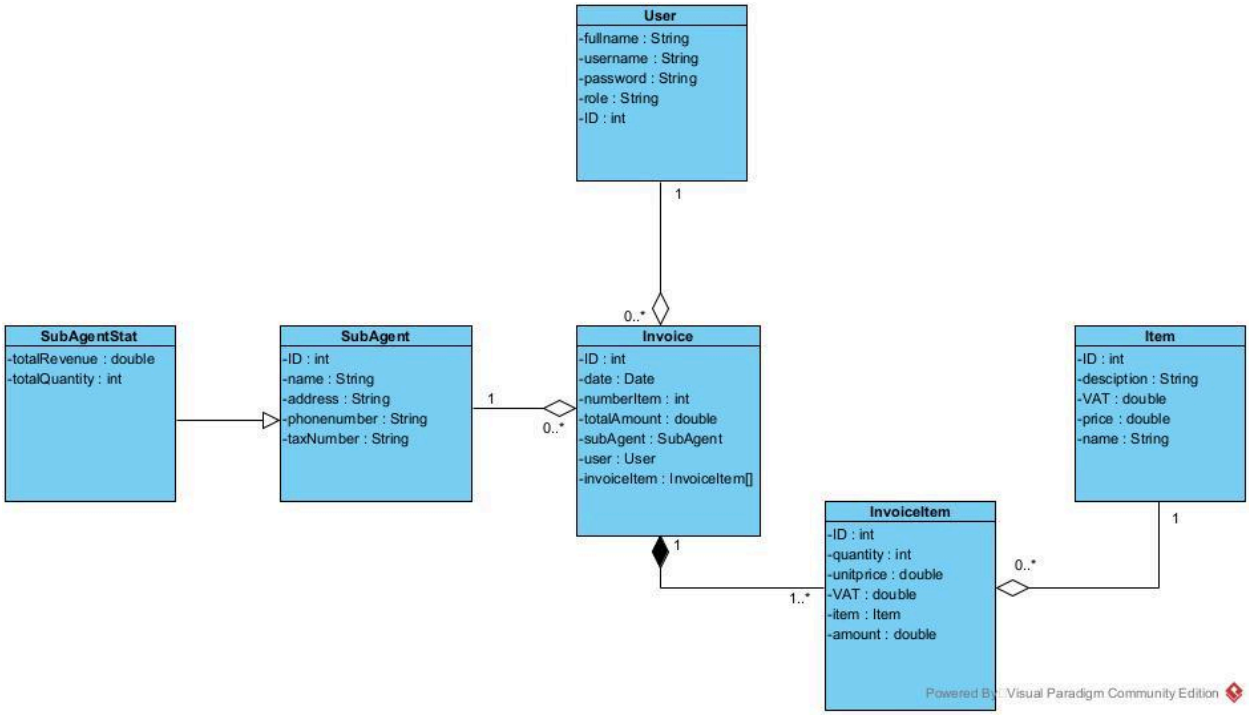
- User:
  - id: int
  - username: String
  - fullname: String
  - password: String
  - role: String
- SubAgent
  - id: int
  - name: String
  - address: String
  - phonenumber: String
  - taxNumber: String
- Invoice
  - id: int
  - Date: Date
  - NumberItem: int
  - TotalAmount: double
- InvoiceItem
  - id: int
  - quantity: int
  - unitprice: double
  - VAT: double
  - amount: double
- Item
  - id: int
  - description: String
  - VAT: double
  - price: double
  - name: String
- SubAgentStat
  - totalRevenue: double
  - totalQuantity: int

Step 3: Convert all association relationships to correspond aggregation/composition relationships:

- Item + Invoice -> InvoiceItem convert to: InvoiceItem is a component of Invoice, Item is a component of InvoiceItem

Step 4: Add the object attributes which correspond to the aggregation/composition relationships:

- InvoiceItem is a component of Invoice, of type n-1 -> Invoice has a list of InvoiceItem
- SubAgent is a component of Invoice , of type 1-n -> Invoice has a SubAgent
- User is a component of Invoice , of type 1-n -> Invoice has a User
- Item is a component of InvoiceItem, of type: 1-n -> InvoiceItem has an Item



7.Database class_Design

-Input: entity class diagram of design

Step 1: Each entity class -> create 1 table

- Class User -> tblUser
- Class SubAgent -> tblSubAgent
- Class Invoice -> tblInvoice
- Class InvoiceItem -> tblInvoiceItem
- Class Item -> tblItem
- Class SubAgentStat -> tblSubAgentStat

Step 2:For each entity class, transfer all non-object attributes to contribute as the columns of the corresponding table

- tblUser has attributes: id, fullname, username, password, role
- tblInvoice has attributes: id, date, numberItem, totalAmount, VAT
- tblInvoiceItem has attributes: id, date, quantity, unitprice, VAT, discount
- tblItem has attributes: id, description, VAT, discount
- tblSubAgent has attributes: id, name, address, phonenumber, taxNumber
- tblSubAgentStat has attributes: totalRevenue

Step 3: Consider the quantity relationships among entity classes

- 1 tblUser - n tblInvoice
- 1 tblSubAgent - n tblInvoice
- 1 tblInvoice - n tblInvoiceItem
- 1 tblItem - n tblInvoiceItem
- t tblSubAgent - 1 tblSubAgentStat

Step 4: Config the key columns for tables:

- tblUser: ID is primary key
- tblSubAgent: ID is primary key
- tblItem : ID is primary key
- tblInvoice: ID is primary key
    - n tblInvoice - 1 tblUser => tblInvoice has foreign key tblUserId
    - n tblInvoice - 1 tblSubAgent => tblInvoice has foreign key tblSubAgentId
- tblInvoiceItem: ID is primary key
    - n tblInvoiceItem - 1 tblInvoice => tblInvoiceItem has foreign key tblInvoiceId
    - n tblInvoiceItem - 1 tblItem => tblInvoiceItem has foreign key tblItemId

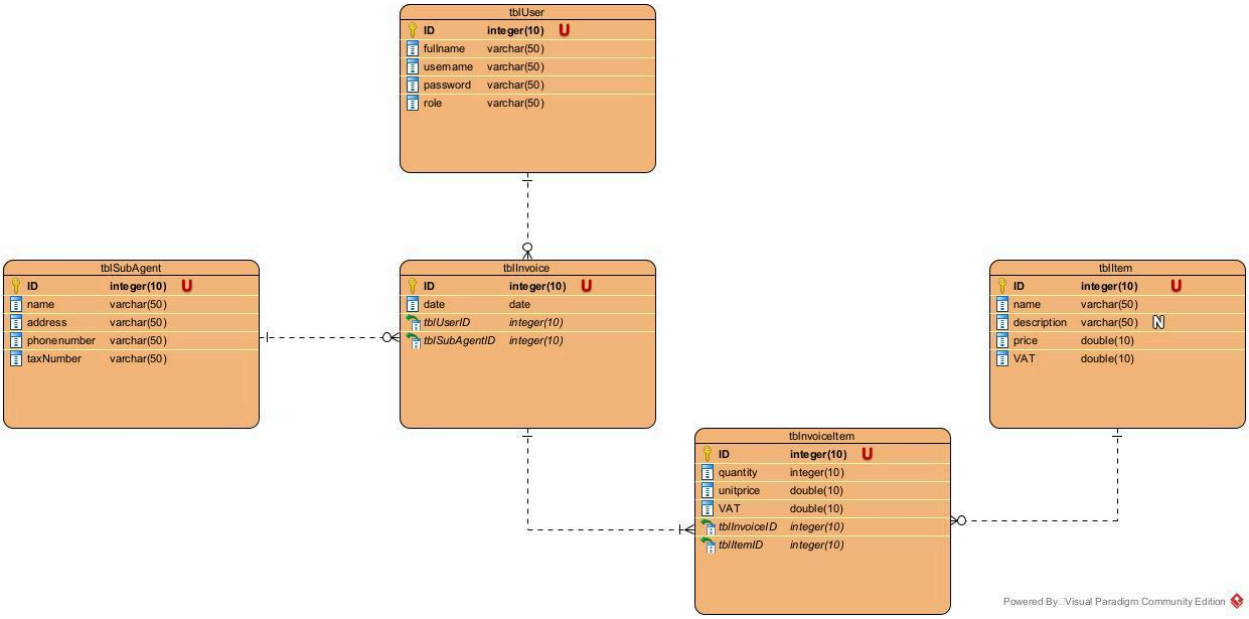
Step 5: Remove redundant attributes:

*Secondary:

- Remove all attributes of tblStatSubAgent -> Remove tblStatSubAgent
- Remove totalAmount, numberItem in Invoice
- Remove amount in InvoiceItem

*Duplicate:

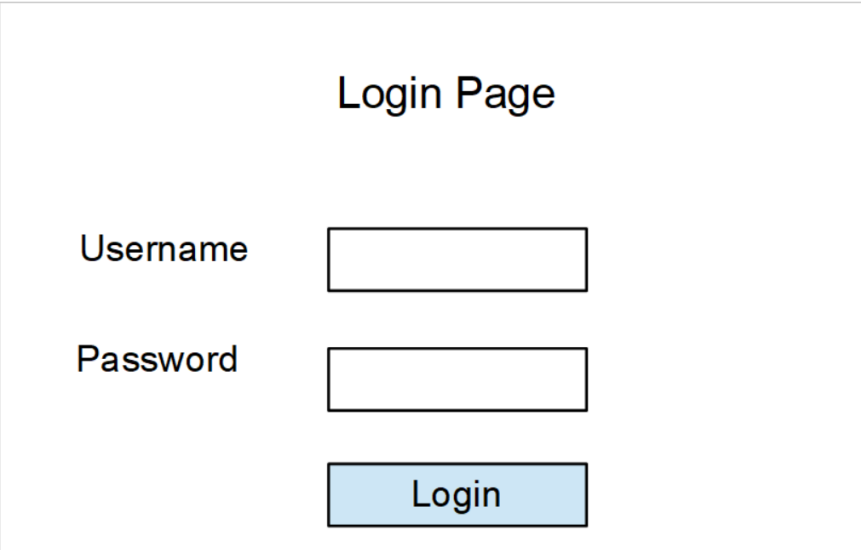
- Remove date in tblInvoiceItem


tblUser
ID integer(10) U
fullname varchar(50)
username varchar(50)
password varchar(50)
role varchar(50)
tblSubAgent
ID integer(10) U
name varchar(50)
address varchar(50)
phonenumber varchar(50)
taxNumber varchar(50)
tblInvoice
ID integer(10) U
date date
tblUserID integer(10)
tblSubAgentID integer(10)
tblItem
ID integer(10) U
name varchar(50)
description varchar(50) N
price double(10)
VAT double(10)
tblInvoiceItem
ID integer(10) U
quantity integer(10)
unitprice double(10)
VAT double(10)
tblInvoiceID integer(10)
tblItemID integer(10)
Powered By Visual Paradigm Community Edition

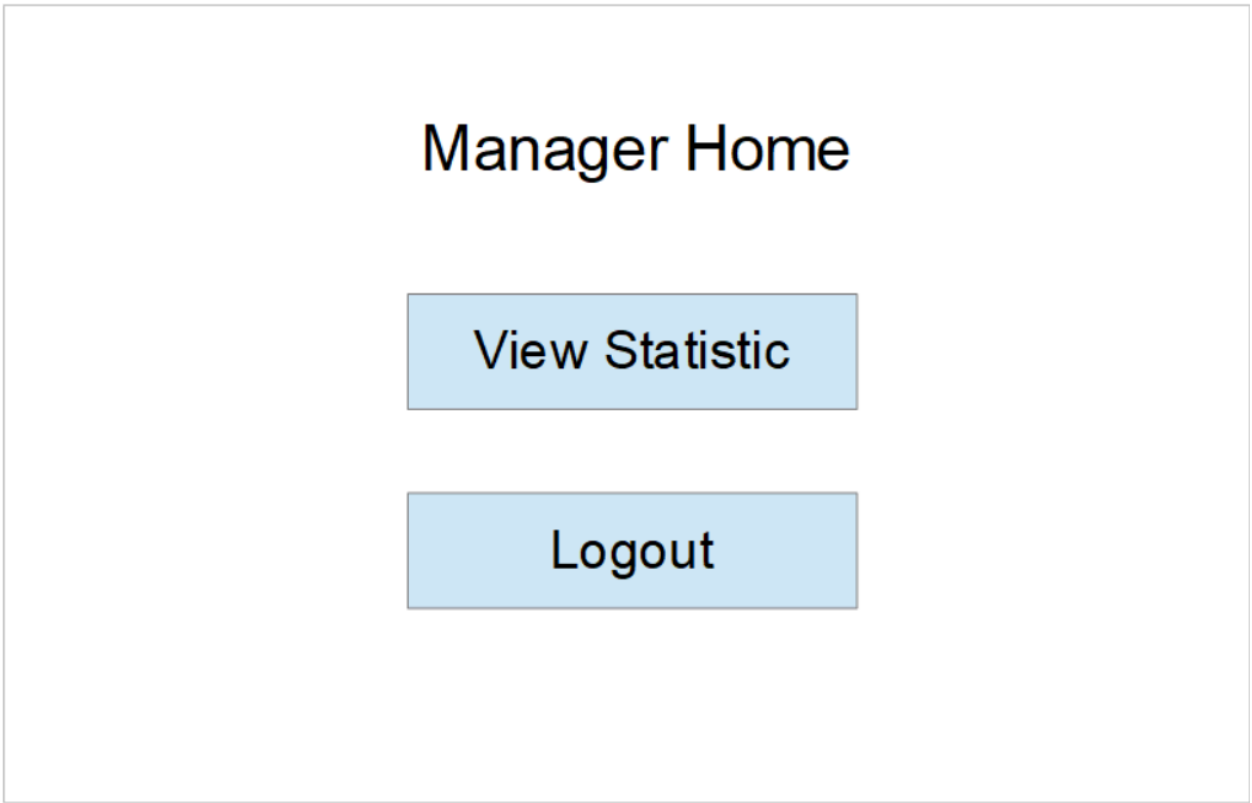
8. UI + Fully class diagram_Design

Interface:

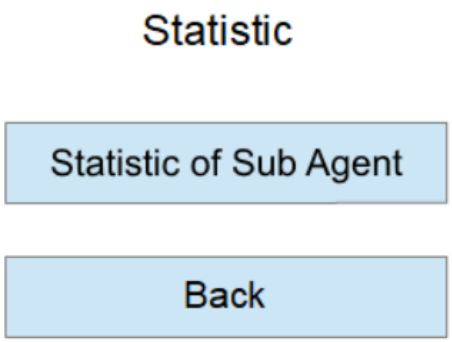
- LoginFrm is the interface to login.



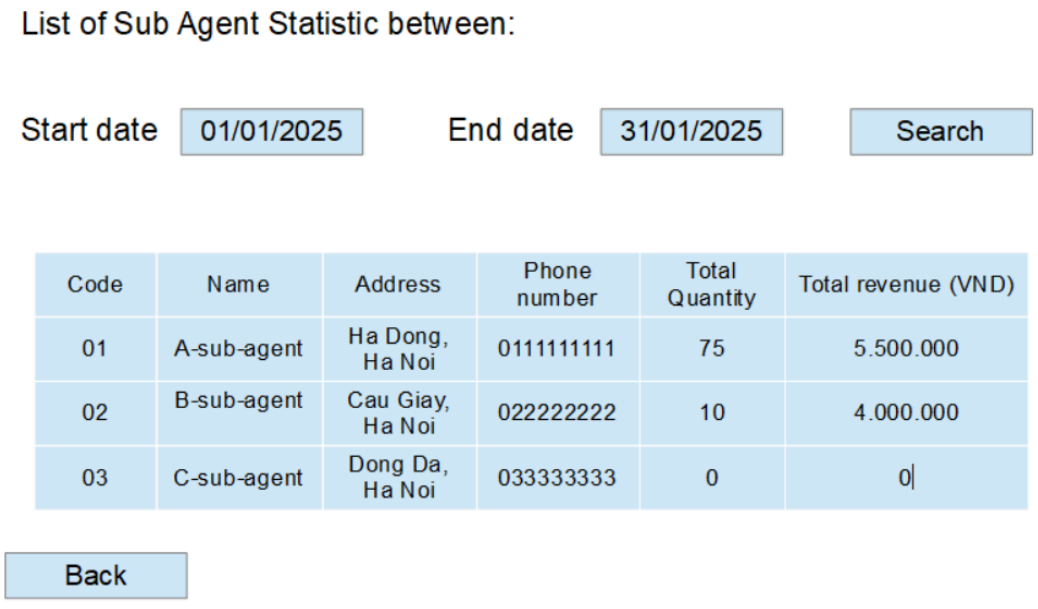
- ManagerHomeFrm is the home interface for the Staff.



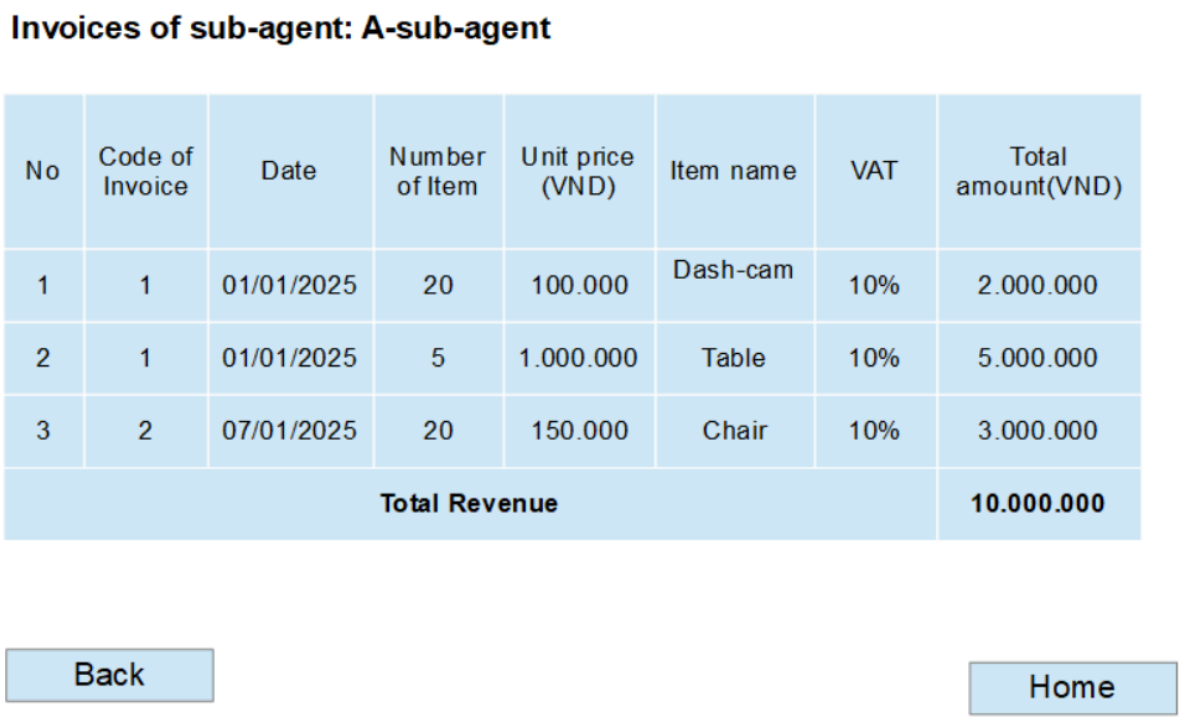
- StatisticFrm is the interface to view statistics.



- SubAgentStatFrm is the interface to view statistics of sub-agent.



- InvoiceFrm is the interface to view statistics list of invoices.



Design based on MVC model:

- Class JFrame
- Interface ActionListener:
  - actionPerformed(e : ActionEvent) : void()
- View classes: inherit JFrame and ActionListener
  - LoginFrm
    - Attributes
      - txtUsername: JTextField
      - txtPassword: JPasswordField
      - btnLogin: JButton
    - Operations

      +LoginFrm()

      +actionperformed(e : ActionEvent) : void
  - ManagerHomeFrm
    - Attributes
      - btnViewStat: JButton
      - btnLogout: JButton

        - lblUsername: JLabel
        - u: User
          *Input from checkLogin() is User so i take it and use it like attributes
    - Operations
          +ManagerHomeFrm(u: User)
          +actionperformed(e : ActionEvent) : void
- StatisticFrm
    - Attributes
        - btnSubAgentStat: JButton
        - btnBack: JButton
        - u: User
    - Operation
          +StatisticFrm(u: User)
          +actionperformed(e : ActionEvent) : void
- SubAgentStatFrm
    - Attributes
        - txtStartDate: JTextField
        - txtEndDate: JTextField
        - btnBack: JButton
        - btnSearch: JButton
        - btnSelect: JButton
        - tblResult: JTable
        - u: User
    - Operations
          + SubAgentStatFrm(u: User)
          +actionperformed(e : ActionEvent) : void
- InvoiceFrm
    - Attributes
        - btnHome: JButton
        - btnBack: JButton
        - lblSubAgent: JLabel
        - tblResult: JTable
        - u: User
        - ss: SubAgentStat
    - Operations
          +InvoiceFrm(u: User, ss: SubAgentStat, sd: Date, ed: Date)

+actionperformed(e : ActionEvent) : void

- Control(DAO – Data Access Object) classes:
  - Class DAO
    - Attributes
      - con: Connection
    - Operation

      +DAO()

  - UserDAO inherit DAO
    - Operations

      +UserDAO()

      +checkLogin(u: User): boolean

      *object as input since it contains username and password; returns boolean to simply indicate whether the credentials are valid—no need to return the whole user object.
  - SubAgentStatDAO inherit DAO
    - Operations

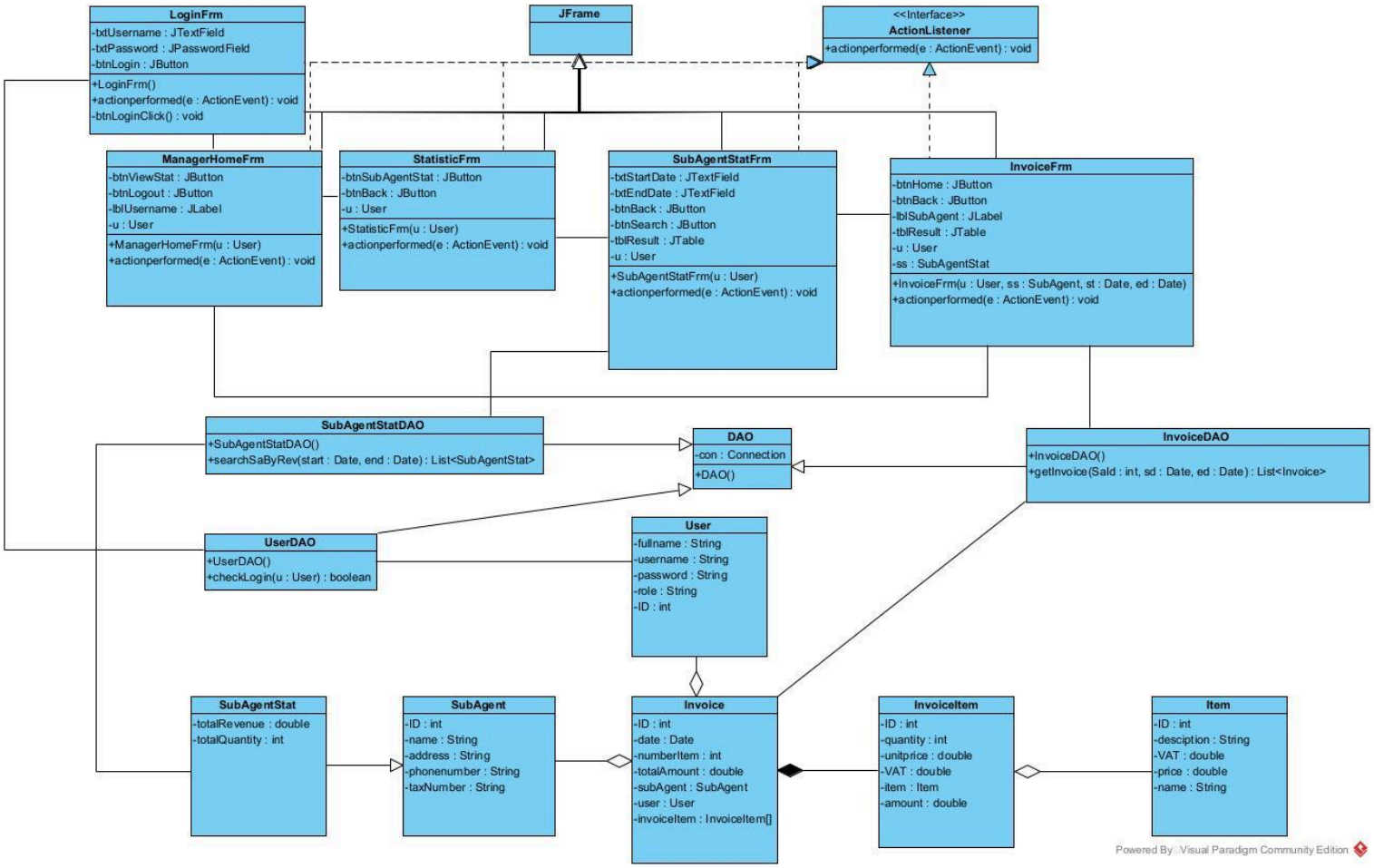
      +SubAgentStatDAO()

      +searchSaByRev(start: Date, end: Date): List<SubAgentStat>

      *because there may be multiple sub-agents operating within the given date range, and using a list allows dynamic sizing, ordering, and easy iteration over the results for further processing or display.
  - InvoiceDAO inherit DAO
    - Operations

      +InvoiceDAO()

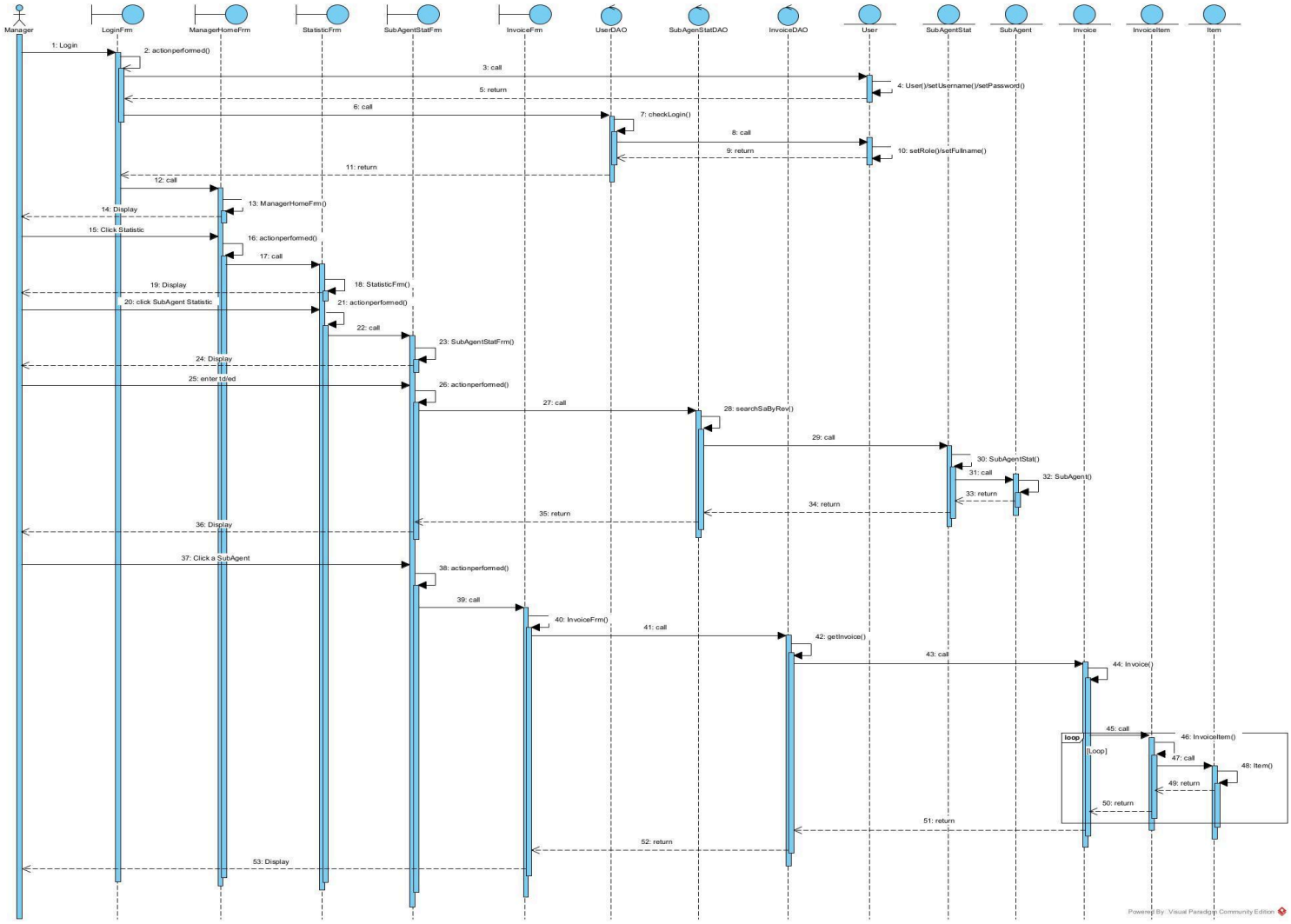
      +getInvoice(SaId: int, sd: Date, ed:Date): List<Invoice>


LoginFrm
-txtUsername : JTextField
-txtPassword : JPasswordField
-btnLogin : JButton
+LoginFrm()
+actionperformed(e : ActionEvent) : void
-btnLoginClick() : void
JFrame
<<Interface>>
ActionListener
+actionperformed(e : ActionEvent) : void
ManagerHomeFrm
-btnViewStat : JButton
-btnLogout : JButton
-lblUsername : JLabel
-u : User
+ManagerHomeFrm(u : User)
+actionperformed(e : ActionEvent) : void
StatisticFrm
-btnSubAgentStat : JButton
-btnBack : JButton
-u : User
+StatisticFrm(u : User)
+actionperformed(e : ActionEvent) : void
SubAgentStatFrm
-txtStartDate : JTextField
-txtEndDate : JTextField
-btnBack : JButton
-btnSearch : JButton
-tblResult : JTable
-u : User
+SubAgentStatFrm(u : User)
+actionperformed(e : ActionEvent) : void
InvoiceFrm
-btnHome : JButton
-btnBack : JButton
-lblSubAgent : JLabel
-tblResult : JTable
-u : User
-ss : SubAgentStat
+InvoiceFrm(u : User, ss : SubAgent, st : Date, ed : Date)
+actionperformed(e : ActionEvent) : void
SubAgentStatDAO
+SubAgentStatDAO()
+searchSaByRev(start : Date, end : Date) : List<SubAgentStat>
DAO
-con : Connection
+DAO()
InvoiceDAO
+InvoiceDAO()
+getInvoice(SaId : int, sd : Date, ed : Date) : List<Invoice>
UserDAO
+UserDAO()
+checkLogin(u : User) : boolean
User
-fullname : String
-username : String
-password : String
-role : String
-ID : int
SubAgentStat
-totalRevenue : double
-totalQuantity : int
SubAgent
-ID : int
-name : String
-address : String
-phonenumber : String
-taxNumber : String
Invoice
-ID : int
-date : Date
-numberItem : int
-totalAmount : double
-subAgent : SubAgent
-user : User
-invoiceItem : InvoiceItem[]
InvoiceItem
-ID : int
-quantity : int
-unitprice : double
-VAT : double
-item : Item
-amount : double
Item
-ID : int
-desciption : String
-VAT : double
-price : double
-name : String
Powered By Visual Paradigm Community Edition

9. Sequence diagram_Desgin

1. A manager clicks the Login button on the interface LoginFrm after entering username and password.
2. The method actionPerformed() of the class LoginFrm is called.
3. The method actionPerformed() calls the constructor User(username, password).
4. The constructor User() packs the information into an User object.
5. The class User returns User object to the method actionPerformed().
6. The method actionPerformed() calls method checkLogin() of the class UserDAO.
7. The method checkLogin() checks the login information.
8. The method checkLogin() calls the class User set more two attributes name, position.
9. The class User calls its method setName(), setPosition().
10. The class User returns the User object to the method checkLogin().
11. The method checkLogin() returns the results to the actionPerformed().
12. The method actionPerformed() calls the class ManagerHomeFrm.
13. The constructor ManagerHomeFrm() is called
14. The interface ManagerHomeFrm is shown to the manager.
15. A manager clicks the Statistic button on the interface ManagerHomeFrm.
16. The method actionPerformed() of the class ManagerHomeFrm is called.
17. The method actionPerformed() calls the constructor StatisticFrm().
18. The constructor StatisticFrm() is called.
19. The interface StatisticFrm is displayed to the manager.
20. A manager clicks the SubAgent Statistic button on the interface StatisticFrm.
21. The method actionPerformed() of the class StatisticFrm is called.
22. The method actionPerformed() calls the constructor SubAgentStatFrm().
23. The constructor SubAgentStatFrm() is called.
24. The interface SubAgentStatFrm is displayed to the manager.
25. The manager enters the start date/end date on SubAgentStatFrm and clicks the Search button.
26. The method actionPerformed() of the class SubAgentStatFrm is called.
27. The method actionPerformed() calls the method searchSaByRev() of the class SubAgentStatDAO.
28. The method searchSaByRev() searches subagents by total revenue.
29. The method searchSaByRev() calls the constructor SubAgentStat() to pack statistical results.
30. The constructor SubAgentStat() is called.

31. The constructor SubAgentStat() calls the setter methods on the class SubAgent to pack attributes.
32. The setter methods return and the SubAgent object is packed.
33. The method searchSaByRev() returns the SubAgentStat object to the method actionPerformed().
34. The method actionPerformed() return to SubAgentStatDAO()
35. The SubAgentDAO returns the results to SubAgentStatFrm.
36. The constructor SubAgentStatFrm() display to the manager
37. A manager clicks on a Sub‐Agent record on the interface SubAgentStatFrm() to view invoices.
38. The method actionPerformed() of the class SubAgentStatFrm is called.
39. The method actionPerformed() calls the constructor InvoiceFrm().
40. The constructor InvoiceFrm() is called.
41. The constructor InvoiceFrm() calls the method getInvoice() of the class InvoiceDAO.
42. The method getInvoice() of the class InvoiceDAO get invoice list by subagentId and period.
43. The method getInvoice() calls the constructor Invoice().
44. The constructor Invoice() is called.
45. The constructor Invoice() calls the InvoiceItem() to pack its object attribute.
46. The constructor Invoice() packs its normal attributes.
47. InvoiceItem class calls Item class to packs.
48. The Item class packs its attributes.
49. The class Item returns the packed object to the class InvoiceItem.
50. The class InvoiceItem return the object to the class Invoice
51. The class Invoice returns the object to the method getInvoice()
52. The method getInvoice() return the results to the constructor InvoiceFrm()
53. The constructor InvoiceFrm() displays the detailed statistic of the invoice list on the interface InvoiceFrm to the manager.



10. Testplan + Testcase

Black-box testcase list:

No.	Module	Test case
1	View sub-agent statistic	There is sub-agent: start date <= date issued <= end

		date
2	View sub-agent statistic	There not invoice: start date <= date issued <= end date
3	View sub-agent statistic	No available period (start > end) no sub-agent
4	View sub-agent statistic	Many invoices from 1 sub-agent at 1 day
5	View sub-agent statistic	There is exactly invoice in start/end date
6	View sub-agent statistic	Sub-agent has many invoices but only appears invoices in perriod

Test case No.1

Database before testing:

tbl User:

ID	fullname	username	password	role
1	Manager	manager	manager	manager
2	Staff	staff	staff	staff

tblSubAgent:

ID	name	address	phonenumber	taxNumber
1	Nguyen Van A	Ha Noi	0999999999	111111111
2	Nguyen Thi B	Vung Tau	0888888888	222222222
3	Nguyen The C	Binh Duong	0777777777	333333333

tblItem:

ID	name	description	price	VAT
1	Dash-cam	dashcam	100.000	10%

2	Table	table	1.000.000	10%
3	Chair	chair	150.000	10%
4	Phone	iphone	5.000.000	10%

tblInvoice:

ID	UserID	SubAgentID	Date
1	2	1	01/01/2025
2	2	1	30/01/2025
3	2	1	30/01/2025
4	2	1	01/08/2025
5	2	2	31/1/2025
6	2	2	12/1/2025
7	2	3	01/02/2025

tblInvoiceItem:

ID	InvoiceID	ItemID	quantity	unit price	VAT
1	1	1	20	100.000	10%
2	1	2	5	1.000.000	10%
3	2	1	10	100.000	10%
4	2	2	20	1.000.000	10%
5	2	4	15	5.000.000	10%
6	6	4	5	5.000.000	10%
7	7	1	10	100.000	10%
8	4	1	10	100.000	10%

9	5	2	5	1.000.000	10%
10	3	1	10	100.000	10%

Testing scenario and expected results

Scenario	**Excepted results**
The manager Manager(id=1) run the program	Interface appears with: ● Text username ● Text password ● Button Login
Enter username: manager, password: manager and click Login	The main manager menu appears with the option is View statistic
Chooses to view the statistics.	The view statistic UI appears with two options: ● view statistic of item ● view statistic of sub-agent.
Chooses to view statistic of sub-agent.	The view statistic of sub-agent UI appears with: ● two input text for statistical period(start - end) ● a button to search.
Enter a keyword as “01/01/2025” to the start date input and “31/01/2025” to the end date input Click on the search button.	The list of sub-agents appears

List of Sub Agent Statistic between:

01/01/2025 31/01/2025 Search

ID	Name	Address	Phone number	Total quantity	Total revenue (VND)	Select
1	Nguyen Van A	Ha Noi	0999999999	80	114.400.000	v
2	Nguyen Thi B	Vung Tau	0888888888	10	33.000.000	
3	Nguyen The C	Binh Duong	0777777777	0	0	

Back

Clicks into the information row of the sub-agent that has the 01 code.

System display

Invoices of sub-agent: Nguyen Van A

No	Code Invoice	Date	Number of item	Unit price (VND)	Item name	VAT	Total amount (VND)
1	1	01/01/2025	20	100.000	Dash-cam	10%	2.200.000
2	1	01/01/2025	5	1.000.000	Table	10%	5.500.000
3	2	30/01/2025	10	100.000	Dash-cam	10%	1.100.000
4	2	30/01/2025	20	1.000.000	Table	10%	22.000.000

5	2	30/01/2025	15	5.000.000	Phone	10%	82.500.000
6	3	30/01/2025	10	100.000	Dash-cam	10%	1.100.000
Total revenue							**114.400.000**

Back Home

The database after testing: no change