FINAL PRODUCT BACKLOG
BUSINESS INTELLIGENCE SYSTEM

JUNGMIN HA (S3718887)
KHOA VU DUY ANH (S3678490)
NICHOLAS OLIVER (S3752041)
SHREY PAREKH (S3710669)
31ST MAY 2019

Table of Contents

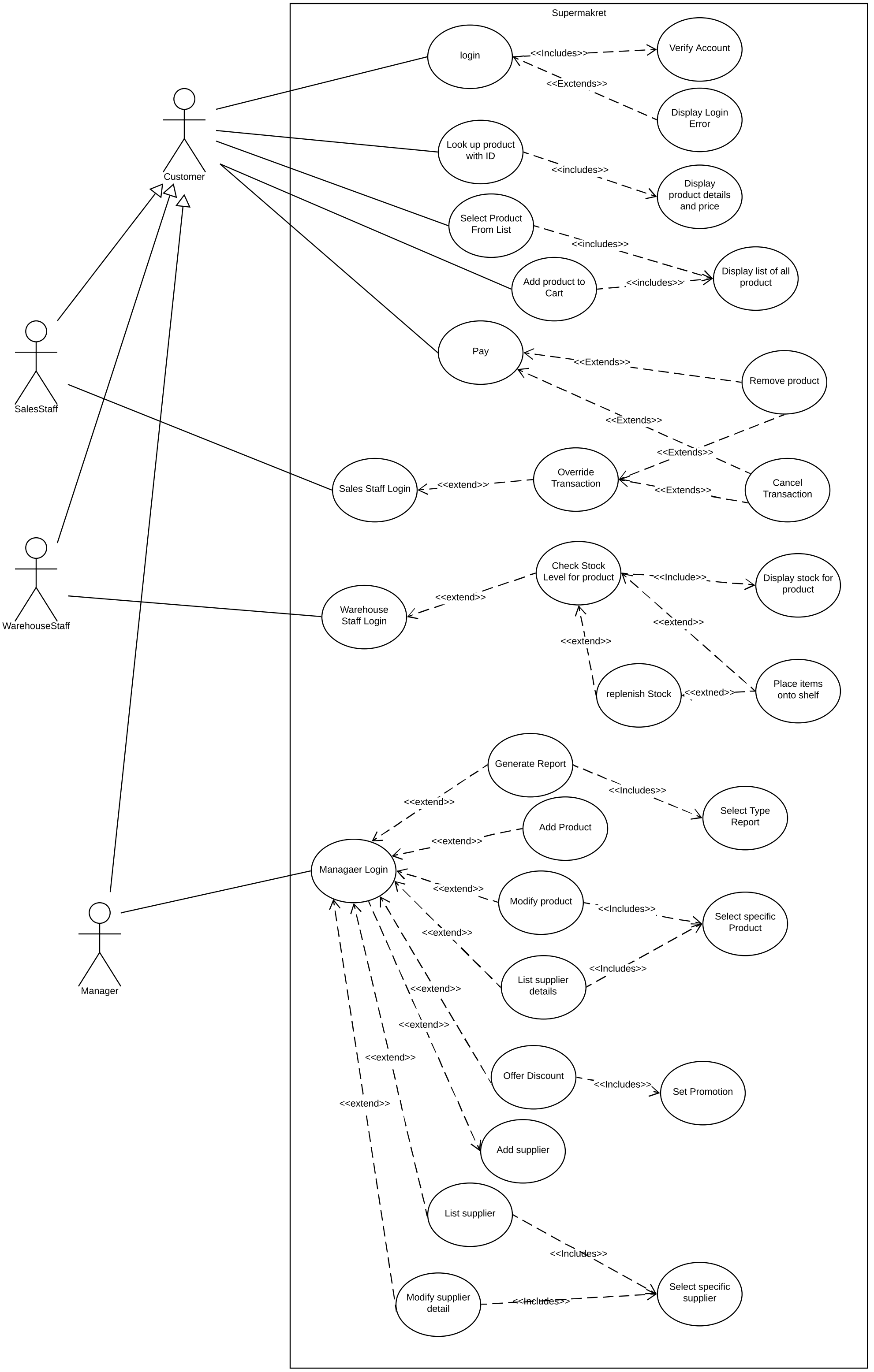
SOURCE CODE

The following section contains the source code for the latest version of the supermarket system.

GROUP DIAGRAMS

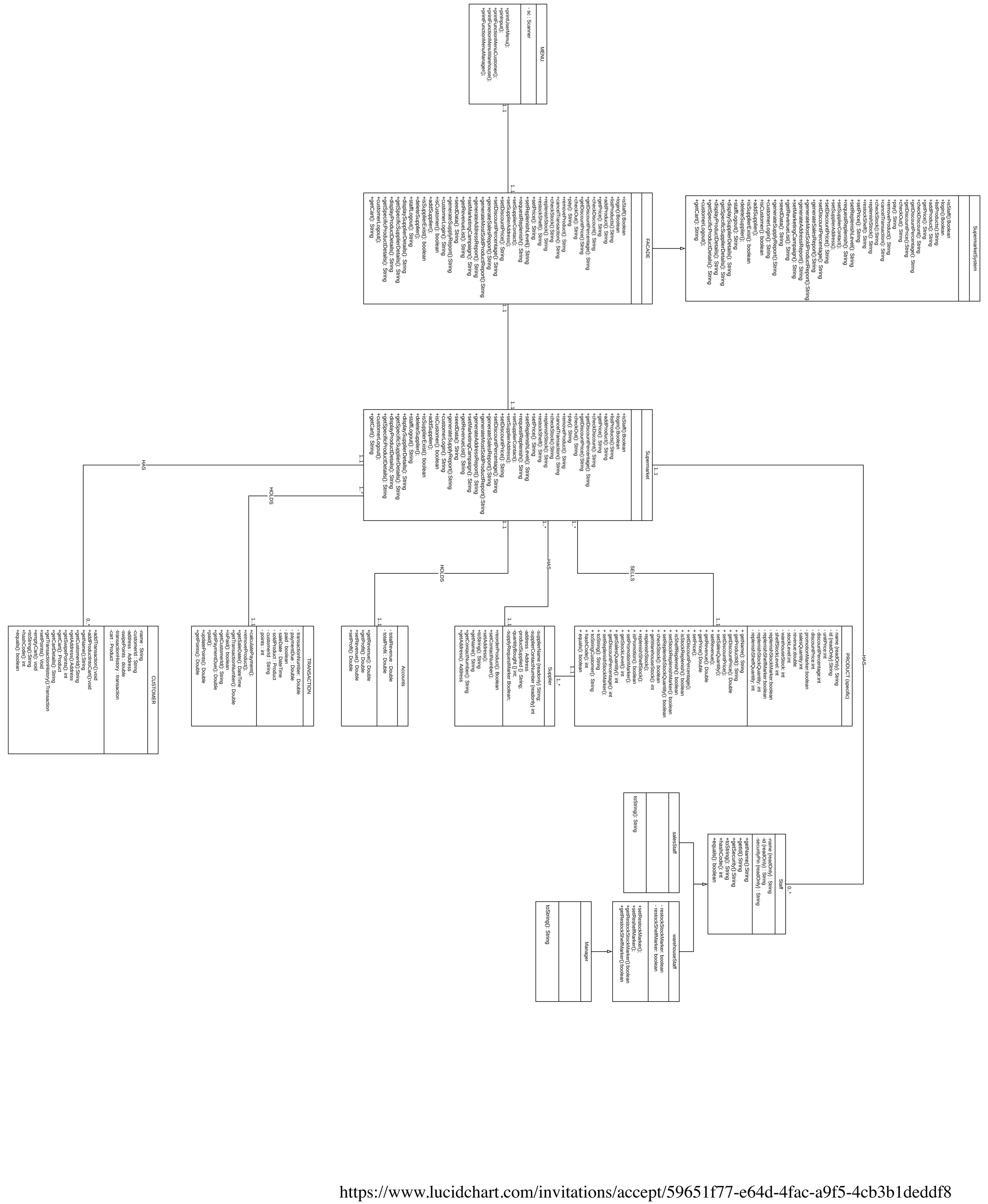
The following section contains the two group diagrams created; The Use-Case Diagram and The Final Class Diagram, respectively.

FINAL USE-CASE DIAGRAM



https://www.lucidchart.com/invitations/accept/f76d56fb-2540-441a-9127-69cb1eef4213

FINAL CLASS DIAGRAM



https://www.lucidchart.com/invitations/accept/59651f77-e64d-4fac-a9f5-4cb3b1deddf8

GROUP EVALUATION

The following section contains group evaluation for the project.

Group Contribution Percentages:

Jungmin Ha:	25%
Khoa Vu Duy Anh:	25%
Nicholas Oliver:	25%
Shrey Parekh:	25%

Building of the Business Intelligence System has been an extensive 12 weeks long process. Starting from forming our groups, creating the first diagram, leading to our first milestone and up to the final day of completion our team has encountered with challenges along the way. The first one of many was learning and implementing new diagrams to a personalized design was huge difficulty. Along with multi- tasking, time-management and prioritization of this project, as a group, over other commitments was difficult. There was a constant issue of language barrier as well. But we sailed through.

We have completed quantifiably 95% of the code and have tried to design each diagram with utmost precision. This could have only been accomplished if all of us had worked together as a unit rather than individual members in disarray of each other. Upon this realization is where we started jumping over our problems and started coming up with solutions leading to us actually learning about software engineering and project management. The question remains what are we taking away from this experience?

All of us have one common answer; we learnt how crucial team work and team management is to be successful in the industry. This project was a micro-organism of a real-life scenario and if this gave us challenges, we have been enabled to get insight into what we should expect put tenfold. We learnt how to effectively and clearly demonstrate our ideas to our team in order to have productive meetings and utilize our time to the maximum. From here we got the idea of micro-planning and working in a group full of different streams of knowledge, strengths as well as weaknesses. Making sure that the right mind was put to the right job and also shared equal load of other tasks was a significant lesson project management taught us. Having been able to apply software engineering theory to a practical opportunity has not made us complete software engineers yet, but it has helped us implant the correct initial footsteps.

As we mentioned that 95% of the code is complete, it is time to address the aspects of our application. 95% because, we have not been able to successfully store data that is created at runtime into a database, text file or serializable hashes, that is we don't have automatic restore either. We still are having to seed in sample data in order to check if our code is functioning or not. This acts as a downside on our reusability because there is never really a record of what happens in our application, it is only valid till the application is running and then it disappears. Moreover, the code does have some redundancies and bugs that need to be fixed and refactored for later releases.

On the other hand, the design our application in terms of its classes has been structured quite well. With the use of a façade that implements an interface, we are able to really confine the system to act according to how the client would want it. The application will not be able to run functions that are not existent in our application. Additionally, having a main supermarket class that holds all the functional methods of the software allows for easy code maintainability considering that we won't have to search all our classes to remove an error or improve the code. We would just have open that one class and look for the method that addresses our issue. Adding to these benefits is that the supermarket class in a sense indirectly implements the interface as it contains all the methods in the interface. Meaning that the application is quite extensible in terms of adding a GUI to the application, considering that the supermarket class can work as our model in MVC approach.

Having said that, it is clear that as of the moment the system lacks a GUI, meaning that the application is completely text based. Even though the text and the menus are usable and clear having long prints to console draws away from the useability of the application. Unfortunately, neither have we been able to extend a barcode scanning system which would bring a real-life aspect to our program. Still, we believe that for a text-based application, the system adheres to good clarity and intuition in how the system should be run. Therefore, at this stage the program is relatively in good condition for later versions and releases.

INDIVIDUAL CONTRIBUTIONS

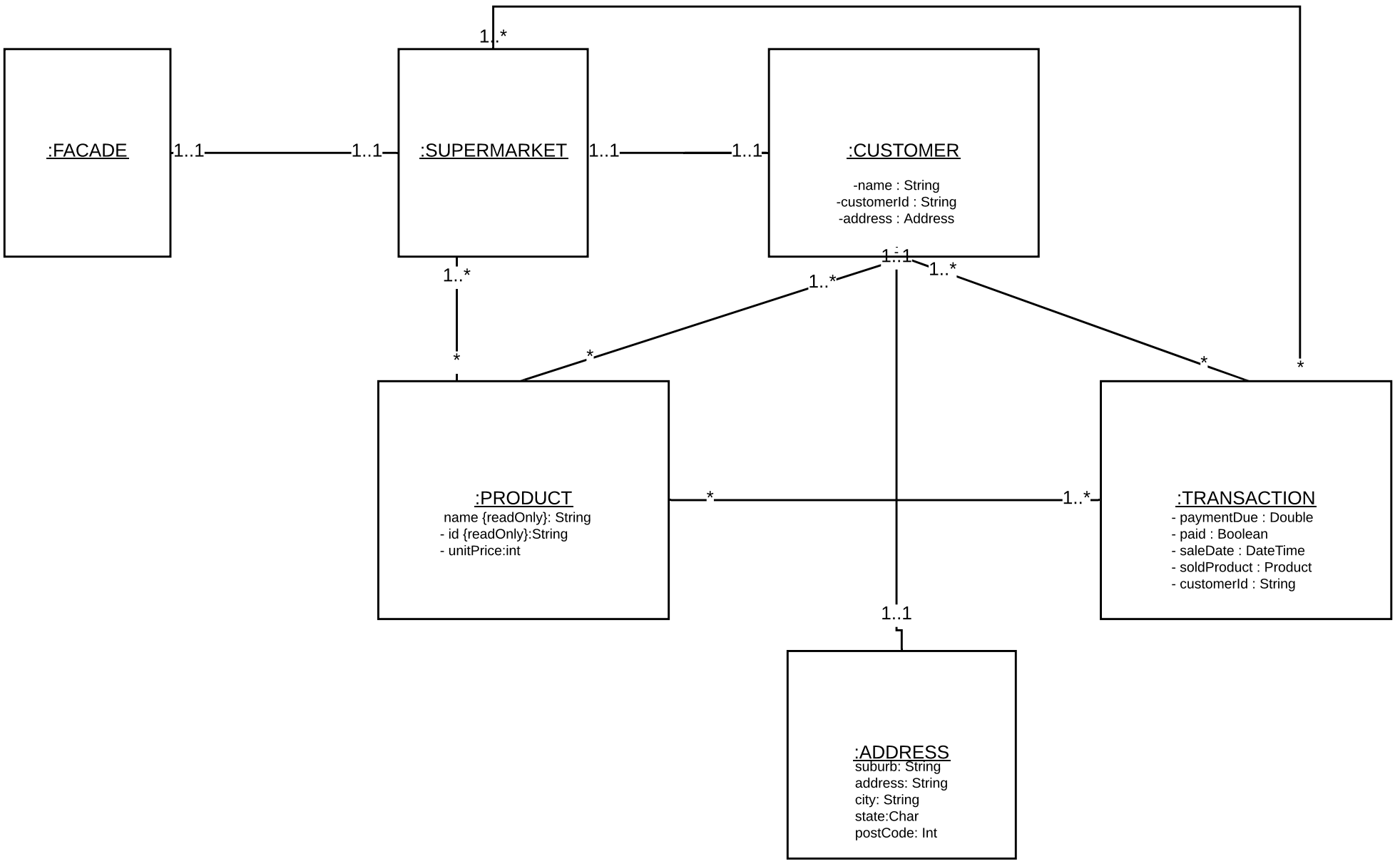
The following section contains the individual diagrams created by each of the group members. The section also contains each member's textual description to the group Use-Case Diagram.

JUNGMIN HA (S3718887)

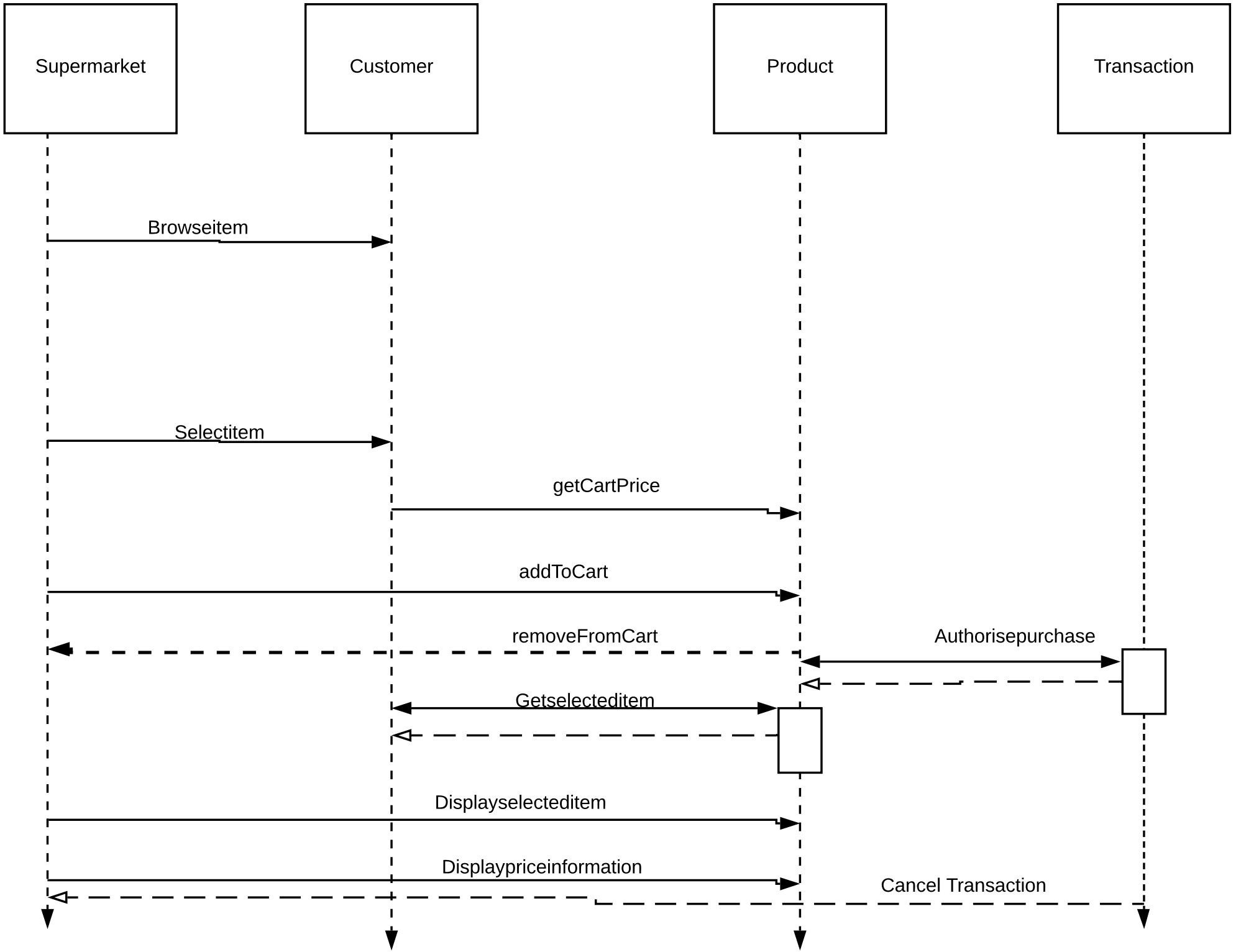
This subsection contains Jungmin Ha's individual contributions to this project. In terms of coding Jungmin was responsible for creating the address utility class and created half of the staff hierarchy. There was also help provided on editing and adding methods to the menu, supermarket and façade classes. In the following pages the following will be shown respectively: textual description of the use case diagram, object diagram, sequence diagram, activity diagram, state diagram and a mini use-case diagram created for milestone 1. Jungmin was also the product owner of this project.

User Case Name	Customer, Sales Staff, Manager Actions
Version	Latest
Goal:	Manager: To check customers order, maintain proper level of stock and generate report Customer: To search product and all related pricings including discounts Sales Staff: To override customers order and check/replenish stock.
Summary:	Customer: Searches product by either its name or ID and checks discount relevant to the items. Customer then checks out. Sales Staff: Staff can override customers order by removing or cancelling products as well as check stock to replenish it if necessary. Manager: Can override customers order like sales staff, maintain proper stock level for the store and generate reports relevant to the store operations.
Actors	Customer, Sales Staff, Manager
Preconditions	All the actors need to be logged in to their respective account.
Triggers	Customer: when customer has logged in and wish to do shopping. Sales Staff: When customers want to remove product or cancel transaction. Also, when stock levels are low. Manager: When customers want to modify or cancel their cart, when stock levels are low and when time comes to check store operations matter.
Basic course of event	Customer: Customer chooses product then checks pricing for discount or promotion and checks out Sales Staff: Staff checks customers order removes a product or cancel transaction in whole. Staff checks low stock and then replenishes it. Manager: Checks customers cart and modifies it, generates store relevant reports and maintains stock level by checking quantities.
Alternative paths	n/a
Post Condition:	Customer finishes transaction Staff modifies the customers cart and replenishes stock Manager generates all report, modifies order and maintains stock levels in store.
Business rules	
Notes	Added too many actors in my use case diagram
Author and date	Jungmin Ha 31/05/2019

Jungmin Ha - S3718887



Jungmin Ha - S3718887



Jungmin Ha - S3718887

Activity Diagram

Customer

Transaction

Manager

Login

Authentication Successful

Browse Products

Search Product ID

Search Product Name

Select Product

Choose Quantity

Remove Product

Add to Cart

Checkout

Add More Product

Cancel Transaction

Pay Now

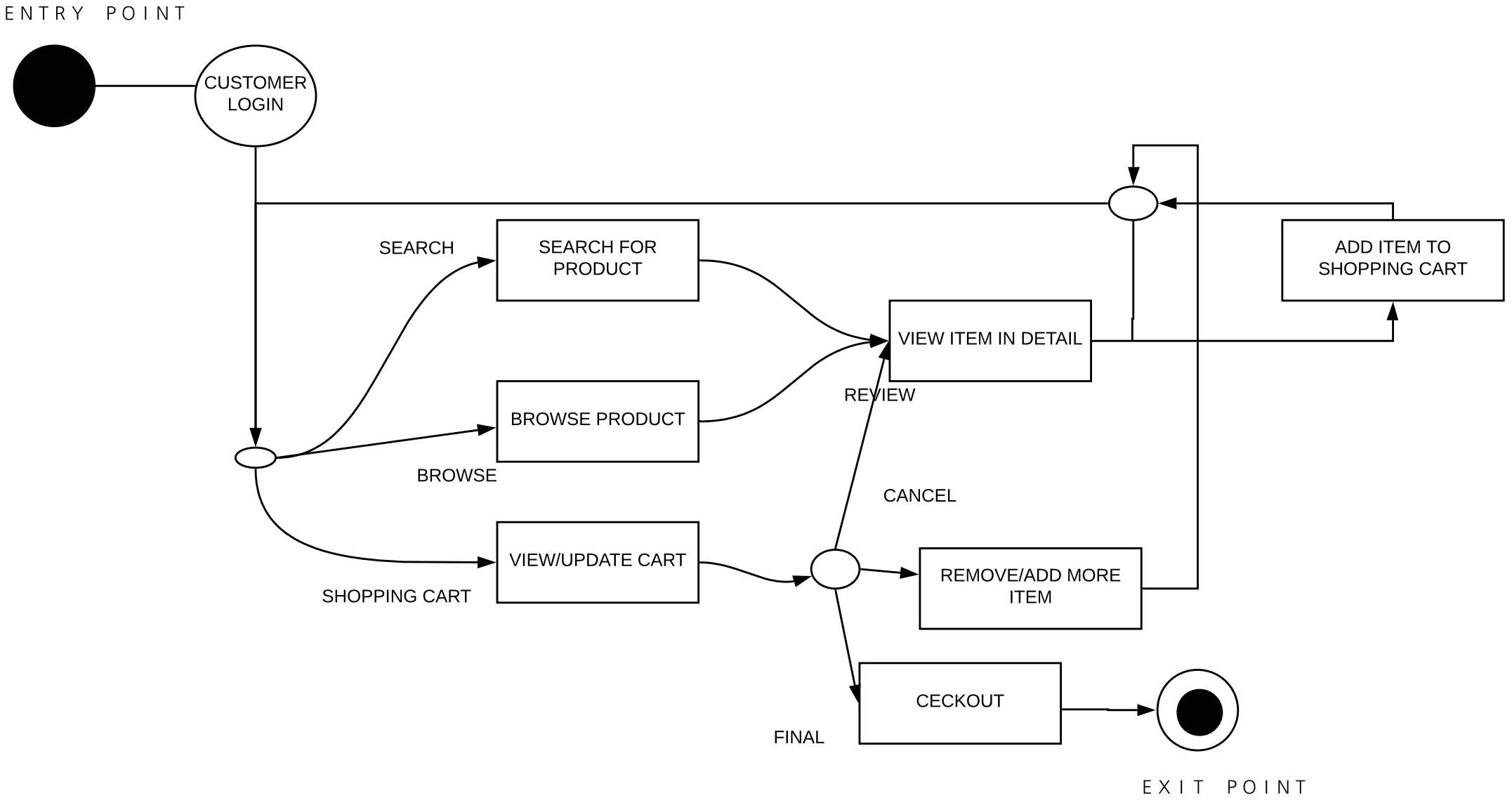
Confirm Purchase

Customers Information is Saved

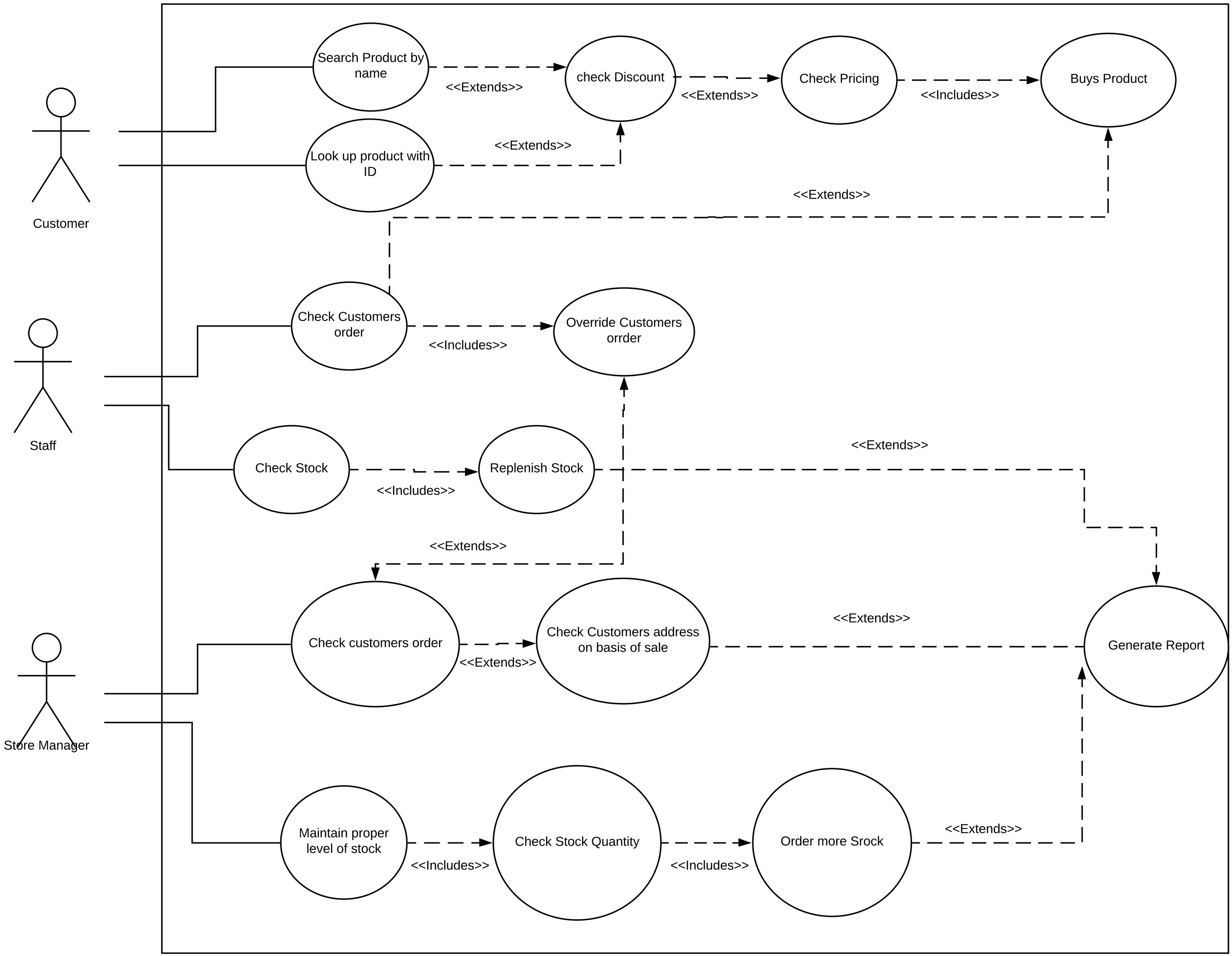
Report is generated regarding most revenue on address

Manager sets campaign and sends flyer to specifc locations

Jungmin Ha - S3718887



Jungmin Ha - S3718887



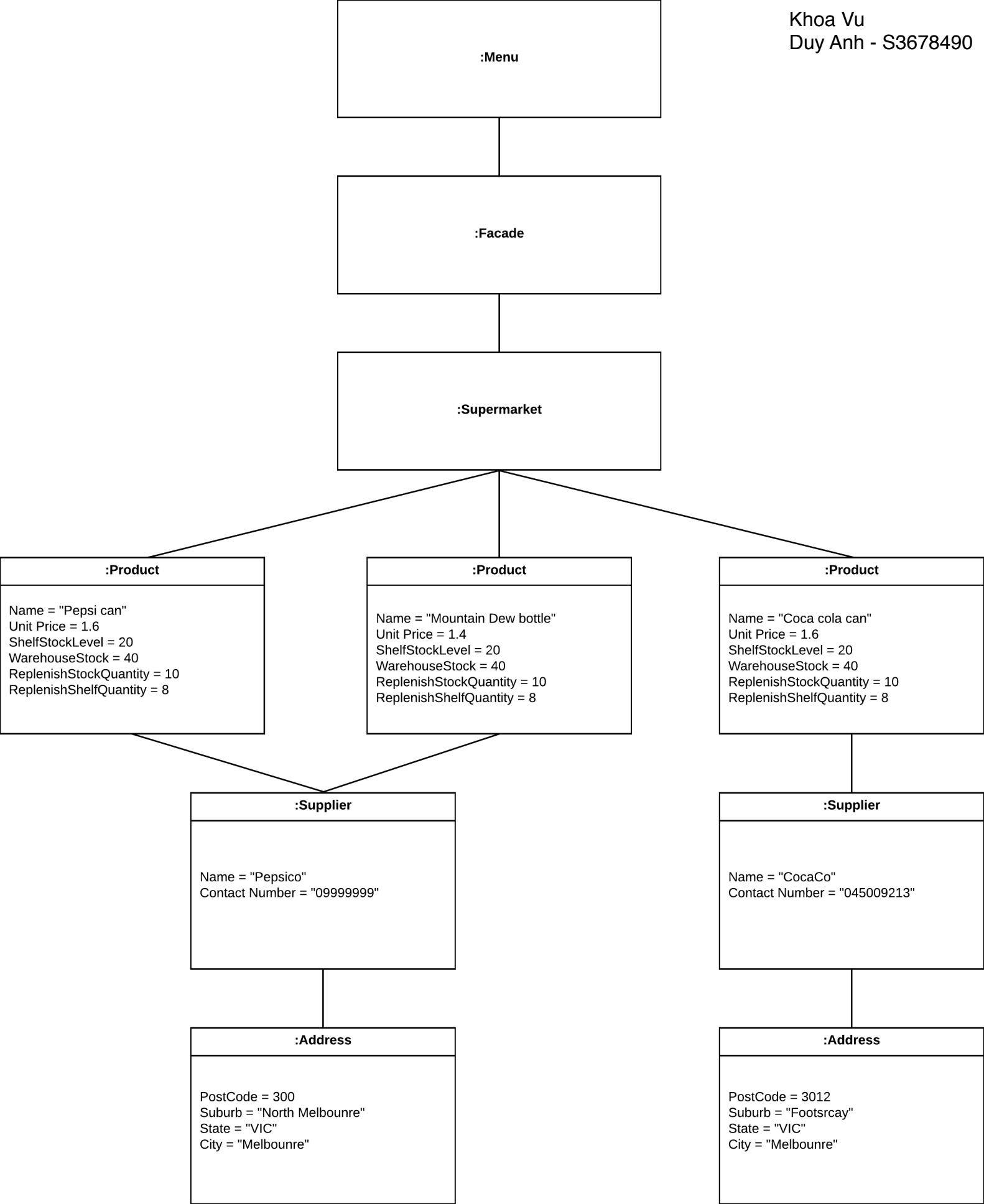
KHOA VU DUY ANH (S3718887)

This subsection contains Khoa Vu's individual contributions to this project. In terms of coding Khoa Vu was responsible for creating the menu class, product class and supplier, supermarket class. There was also help provided on editing and adding methods to the façade and the interface. In the following pages the following will be shown respectively: textual description of the use case diagram, object diagram, sequence diagram, activity diagram, state diagram and a mini use-case diagram created for milestone 1. Khoa Vu was also the technical head of this project.

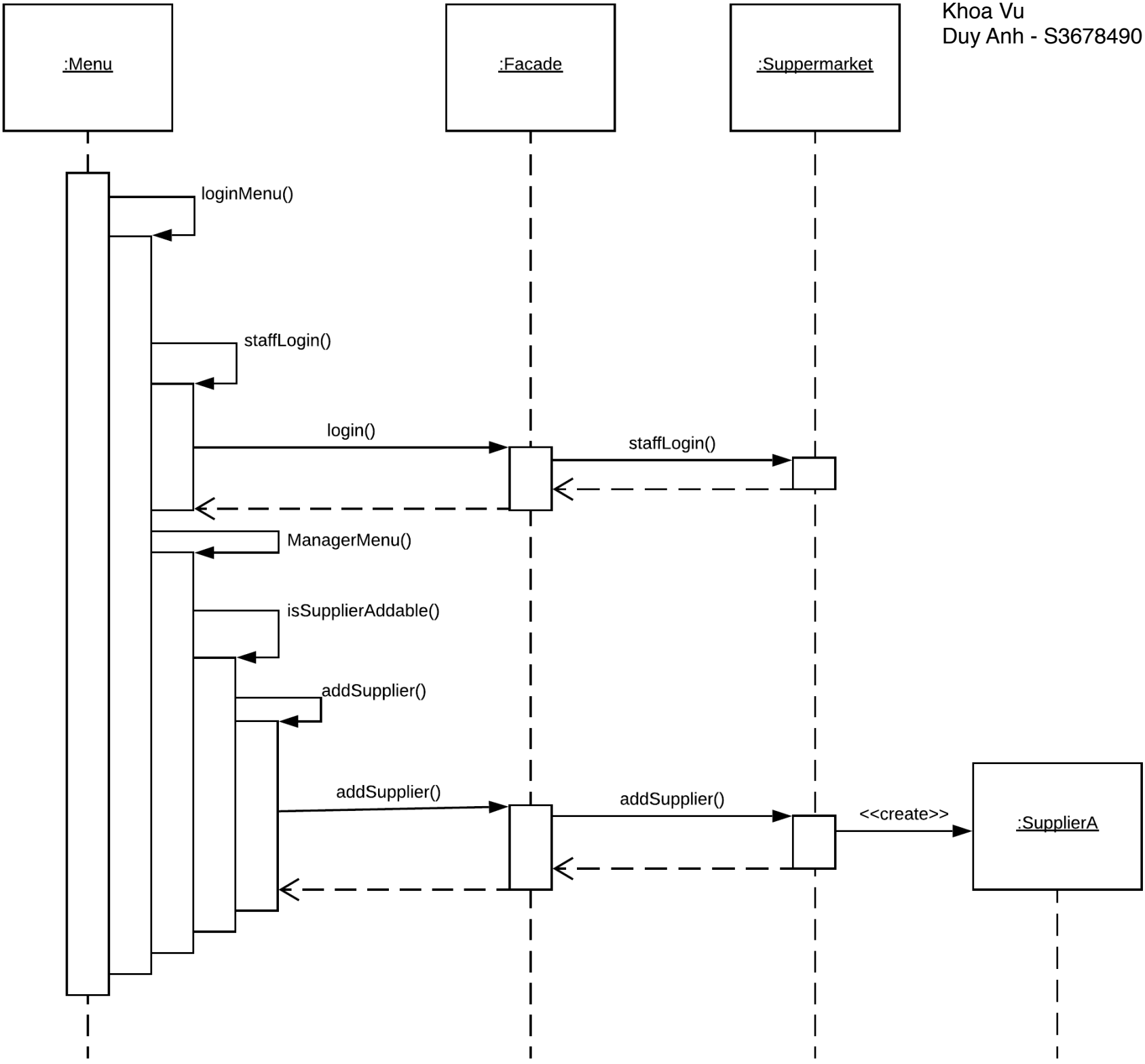
Textual Description

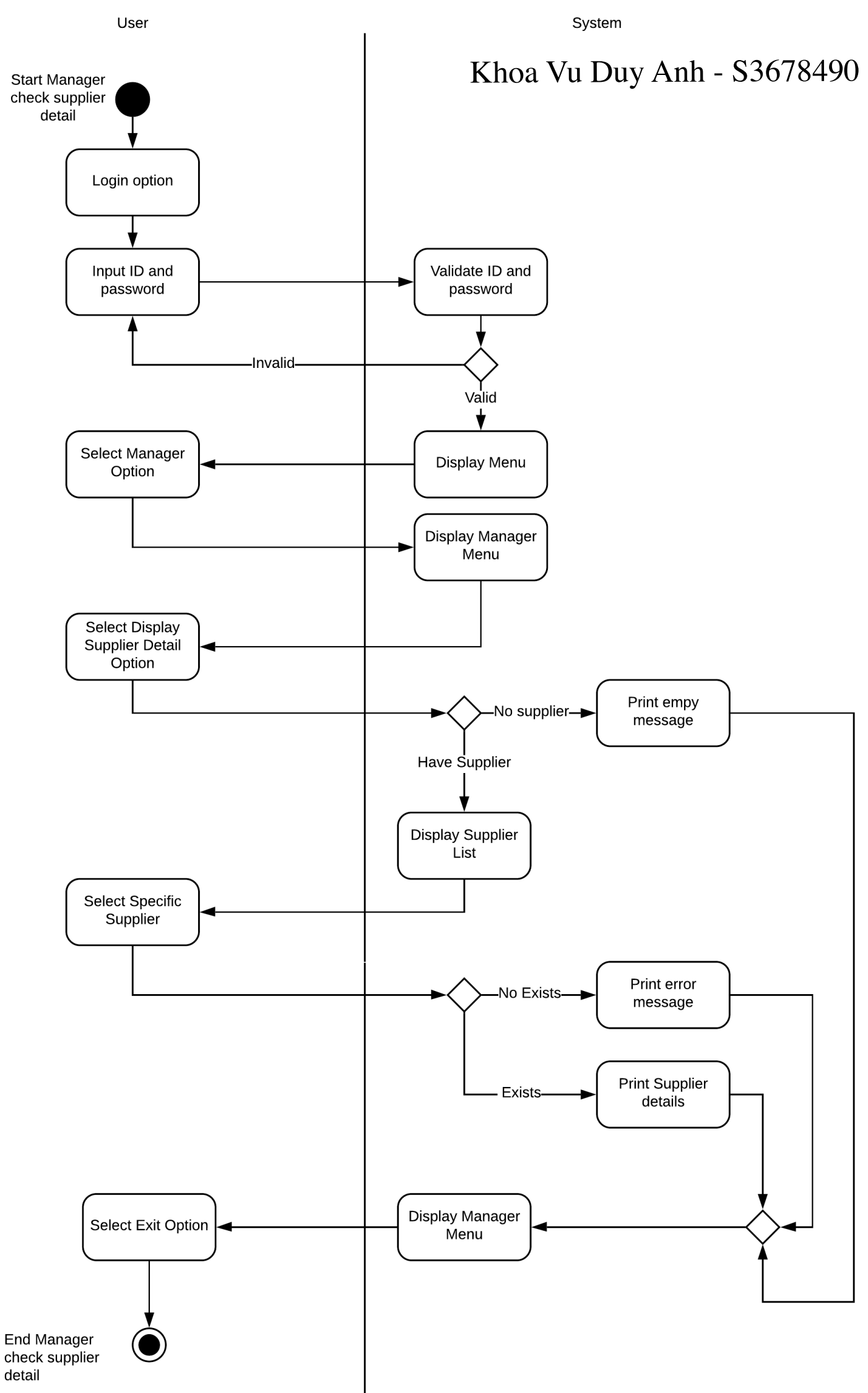
Use Case Name	Login
Version	1.0
Goal	To be able to access Manager menu
Summary	Manager type in Manager pass word and id to verify
Actors	Manager
Preconditions	None
Triggers	Select Login option
Basic Course of Events	Login successfully and able to select manager menu
Alternative paths	Login fail return to Main menu
Post-conditions	Logged in as Manager
Business Rules	None
Notes	Only one user can log in the system at the same time
Author and Date	Khoa Vu Duy Anh

Khoa Vu
Duy Anh - S3678490

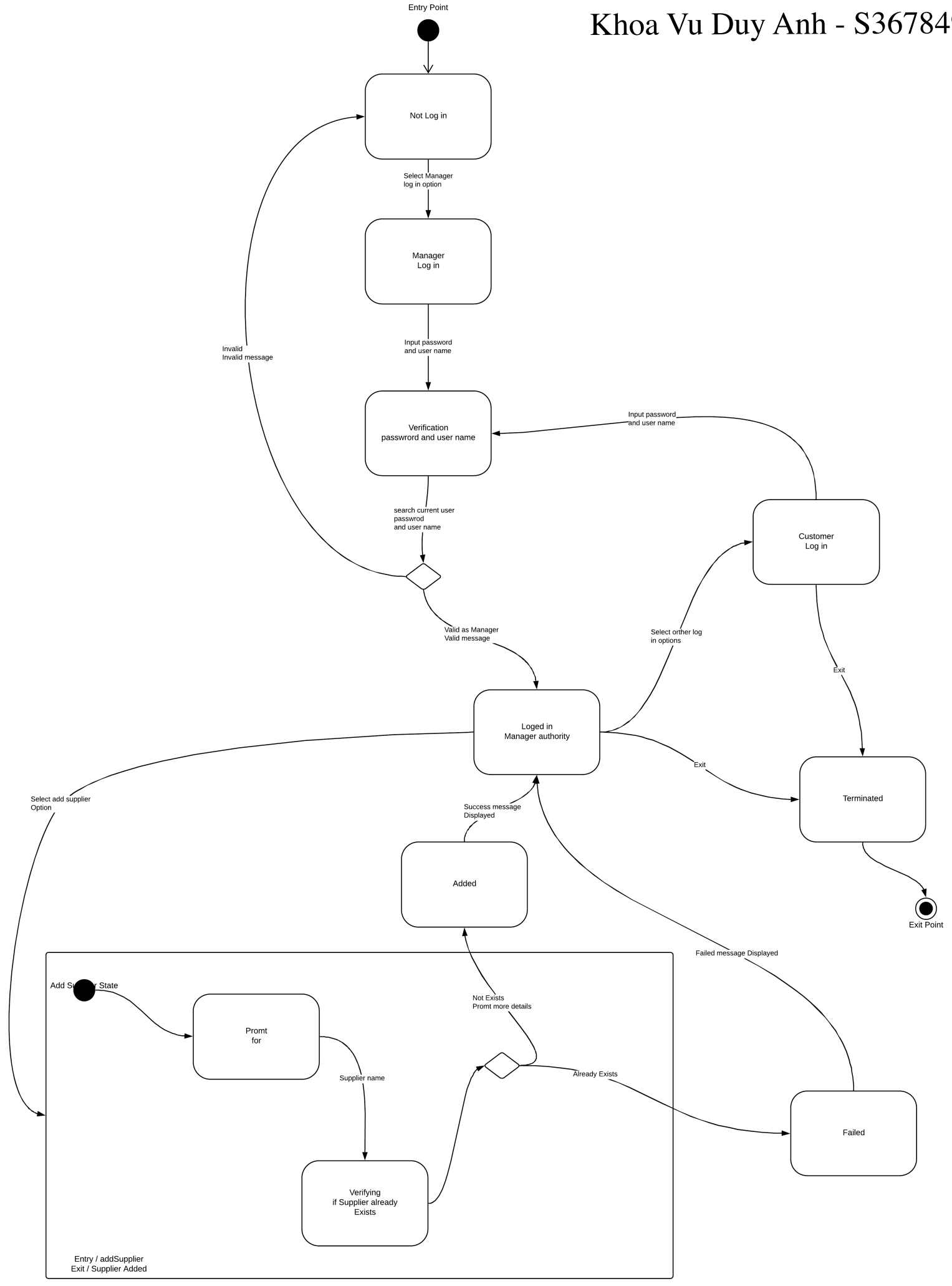


Khoa Vu
Duy Anh - S3678490

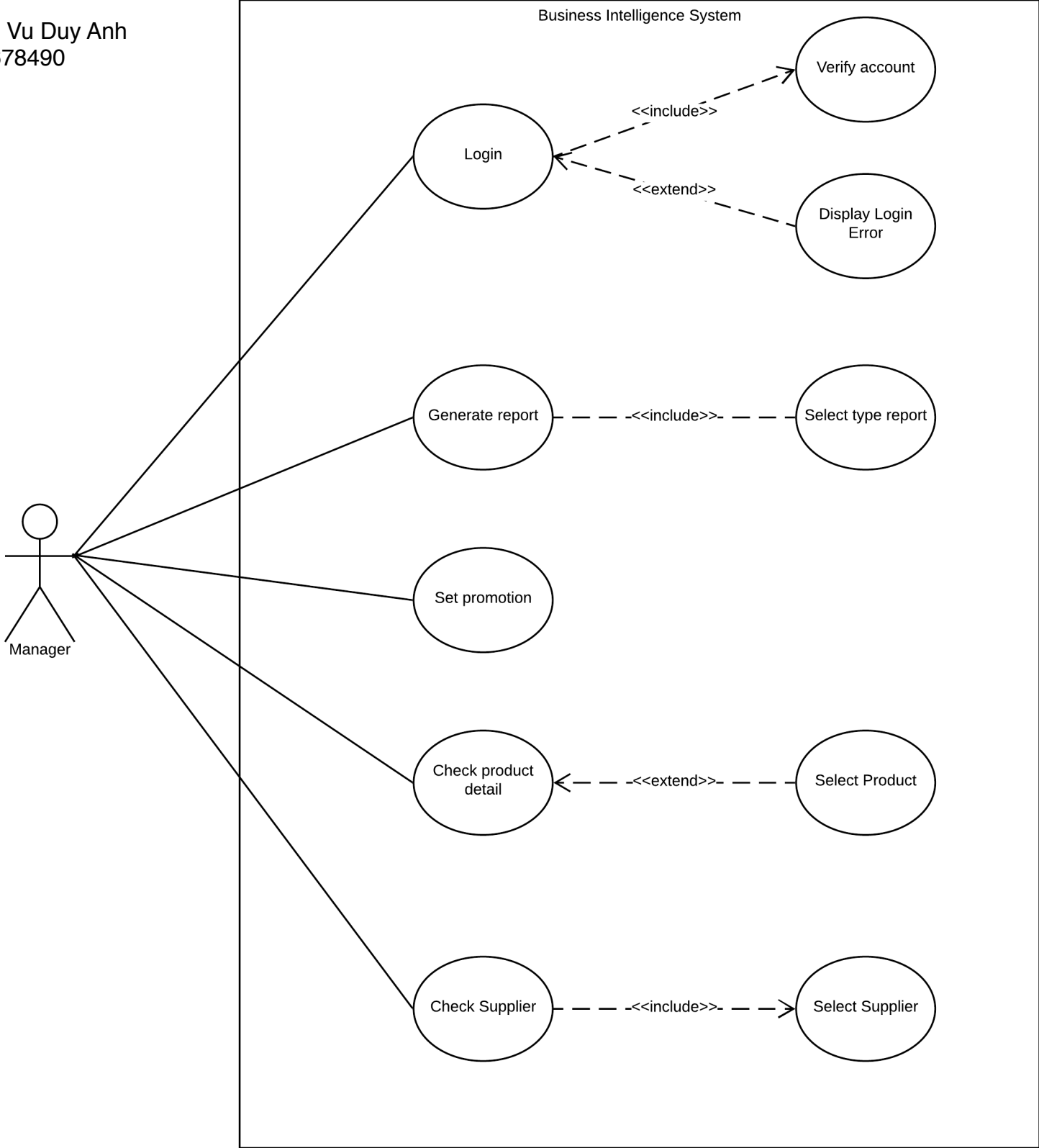



User
System
Khoa Vu Duy Anh - S3678490
Start Manager check supplier detail
Login option
Input ID and password
Validate ID and password
Invalid
Valid
Select Manager Option
Display Menu
Display Manager Menu
Select Display Supplier Detail Option
No supplier
Print empy message
Have Supplier
Display Supplier List
Select Specific Supplier
No Exists
Print error message
Exists
Print Supplier details
Select Exit Option
Display Manager Menu
End Manager check supplier detail

Khoa Vu Duy Anh - S3678490



Khoa Vu Duy Anh
- S3678490

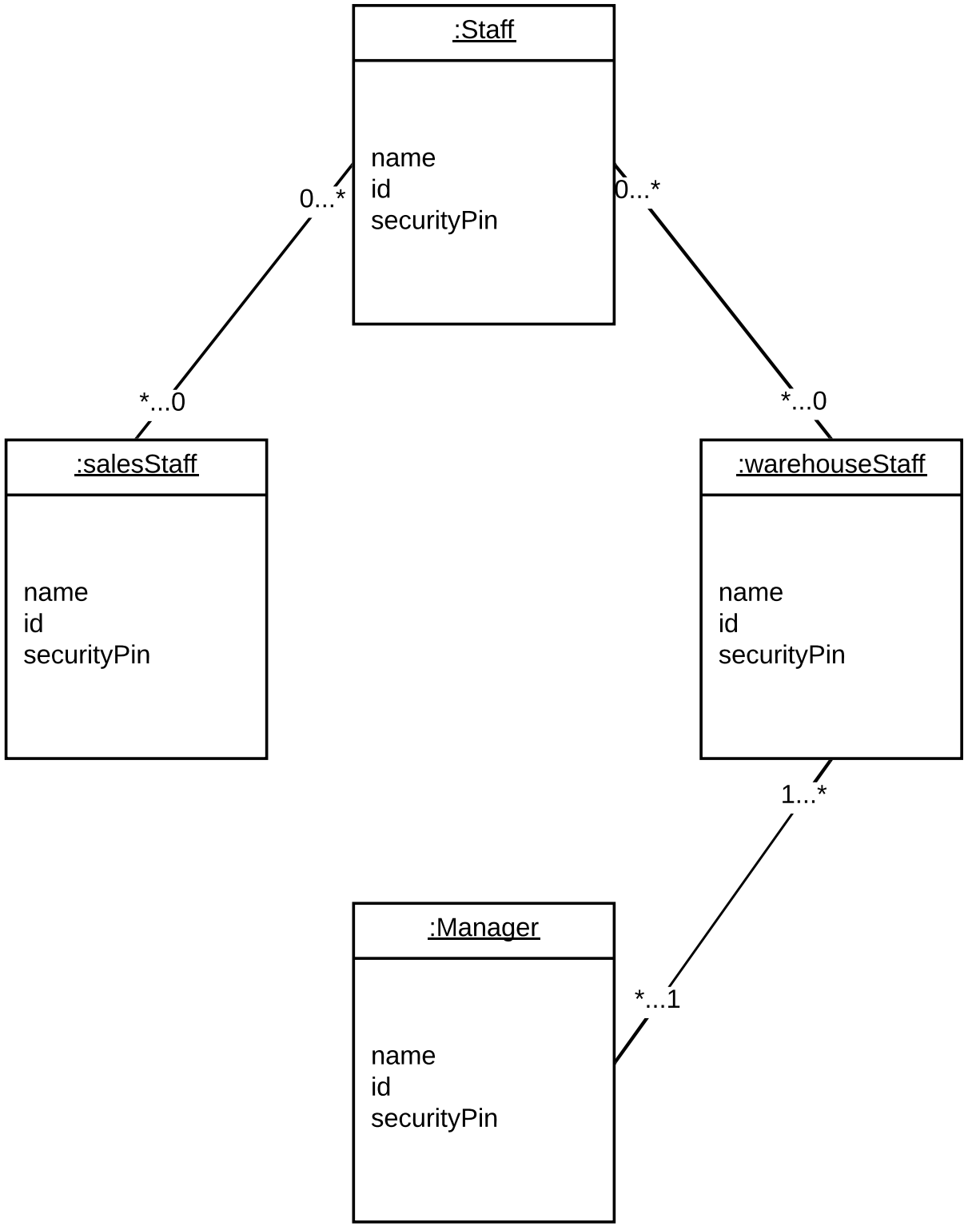


NICHOLAS OLIVER (S3752041)

This subsection contains Nicholas Oliver's individual contributions to this project. In terms of coding Nicholas was responsible for creating the account class and created the half of the staff hierarchy. There was also help provided on editing and adding methods to the menu, supermarket and façade classes. In the following pages the following will be shown respectively: textual description of the use case diagram, object diagram, sequence diagram, activity diagram, state diagram and a mini use-case diagram created for milestone 1.

Name	UC: Remove Product
Goal	To remove a product from the cart
Actor	Sales Staff
Description	A customer can only remove a product from the cart specifically using the help from a Sales Staff
Pre-conditions	A customer has to be logged in, at least one item needs to be in the cart
Trigger	Upon payment, the customer selects "No" when prompted to pay
Basic Course of Events	1. Menu is printed by the System 2. Customer logs in 3. Customer adds an item to the cart 4. Customer checks out to pay 5. Customer declines to pay 6. Customer selects "Remove Product" from a different menu 7. Sales Staff login is prompted 8. Sales Staff logs in 9. System asks for product Code for product to be removed 10. Code is entered, product is removed
Alternative Course of Events	1. Menu is printed by the System 2. Customer logs in 3. Customer adds an item to the cart 4. Customer checks out to pay 5. Customer declines to pay 6. Customer selects "Cancel Transaction" from a different menu 7. Sales Staff login is prompted 8. Sales Staff logs in 9. Entire transaction is cancelled and all products are removed
Post Conditions	The Item with the product code typed in can no longer be in the cart

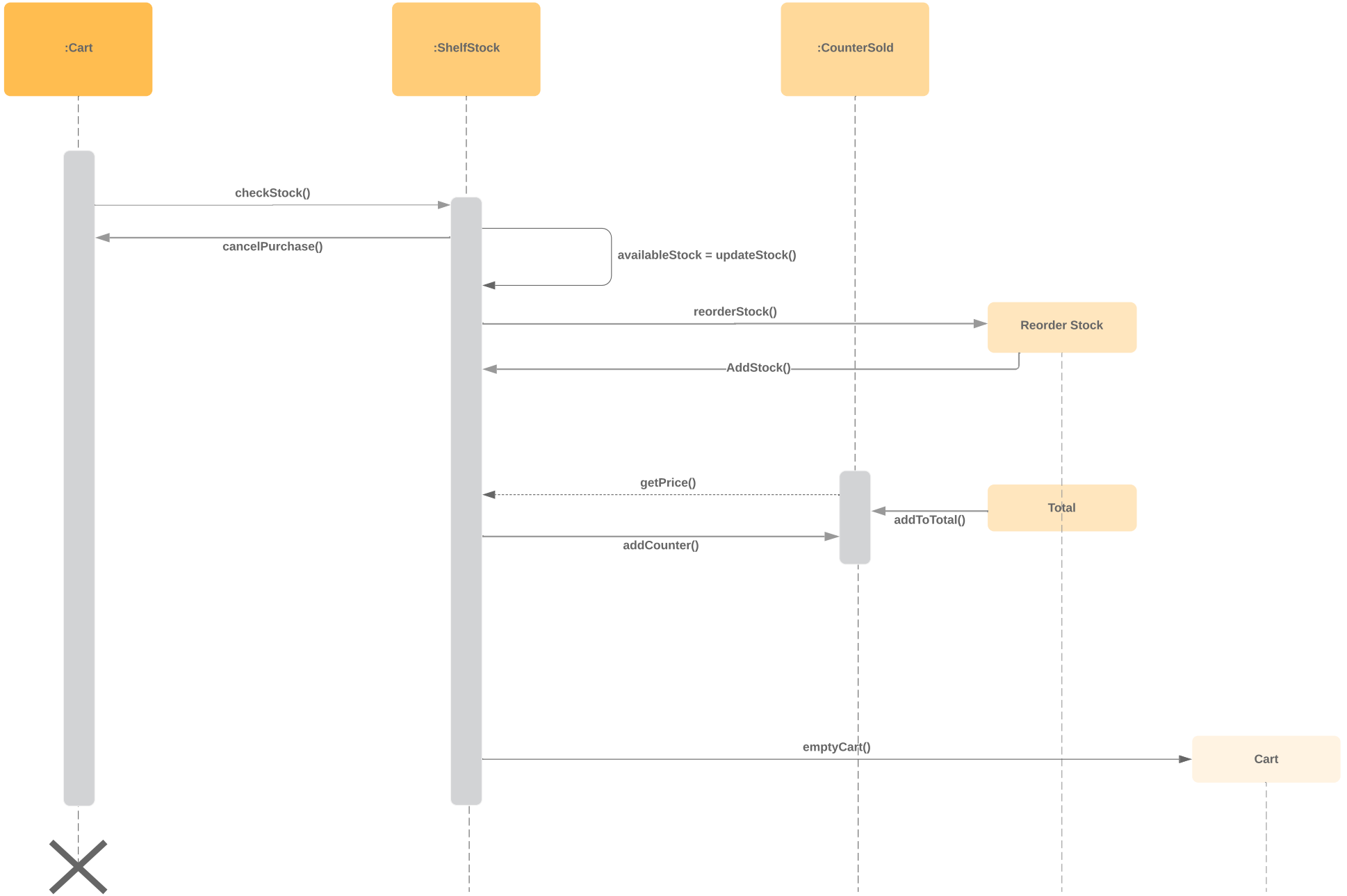
Nicholas Oliver - S3752041



Nicholas Oliver - S3752041

Buying an Item

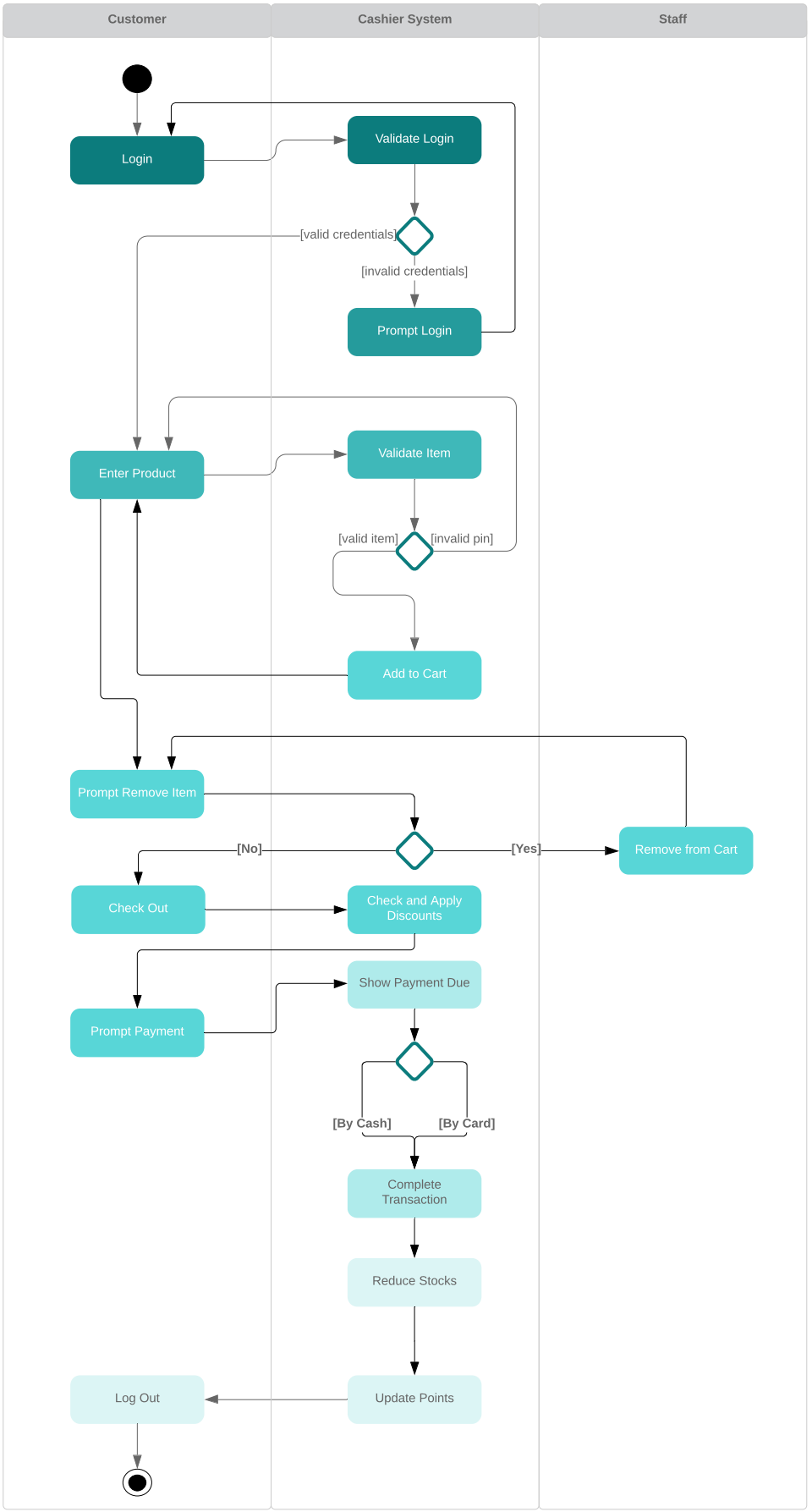
s3752041 | May 29, 2019



Nicholas Oliver - S3752041

Supermarket Activity Diagram

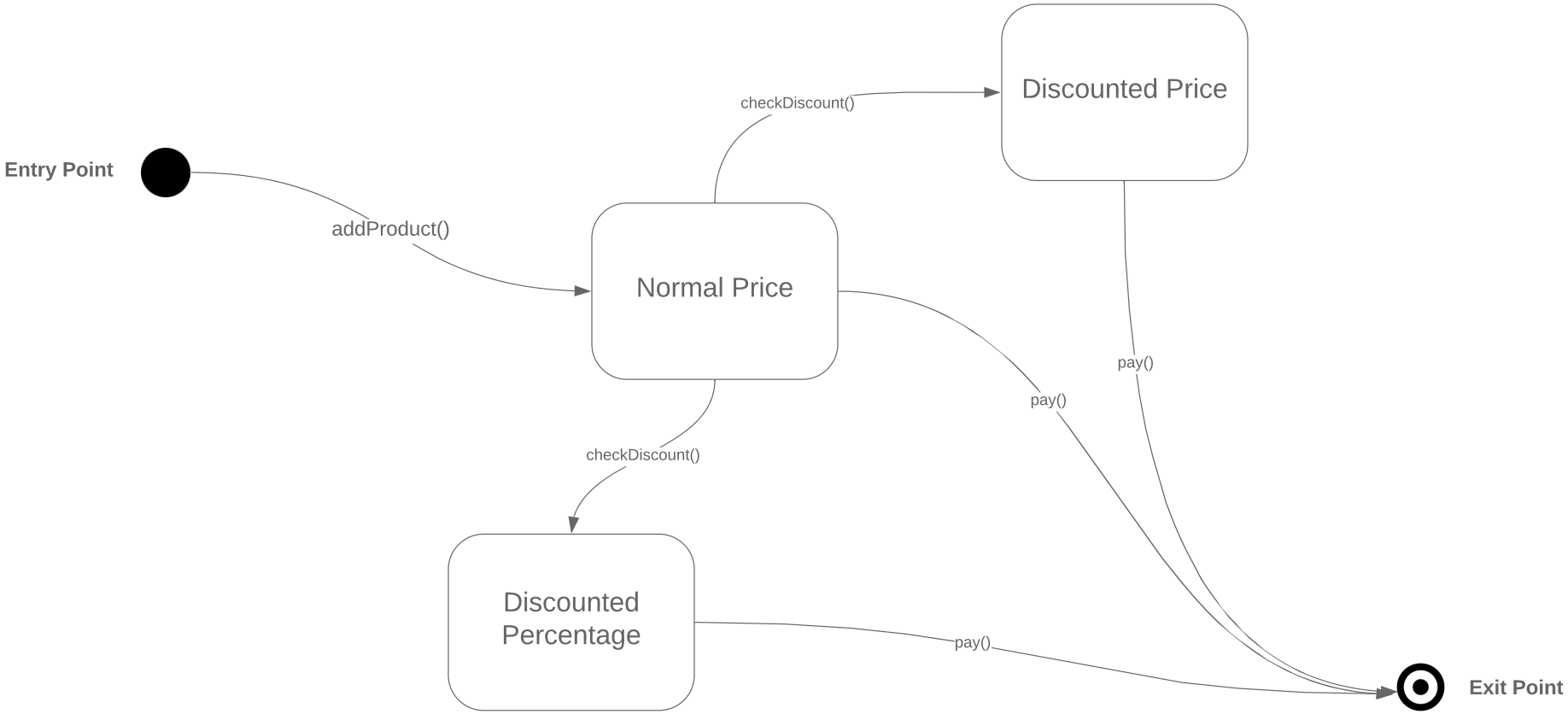
s3752041 | May 29, 2019



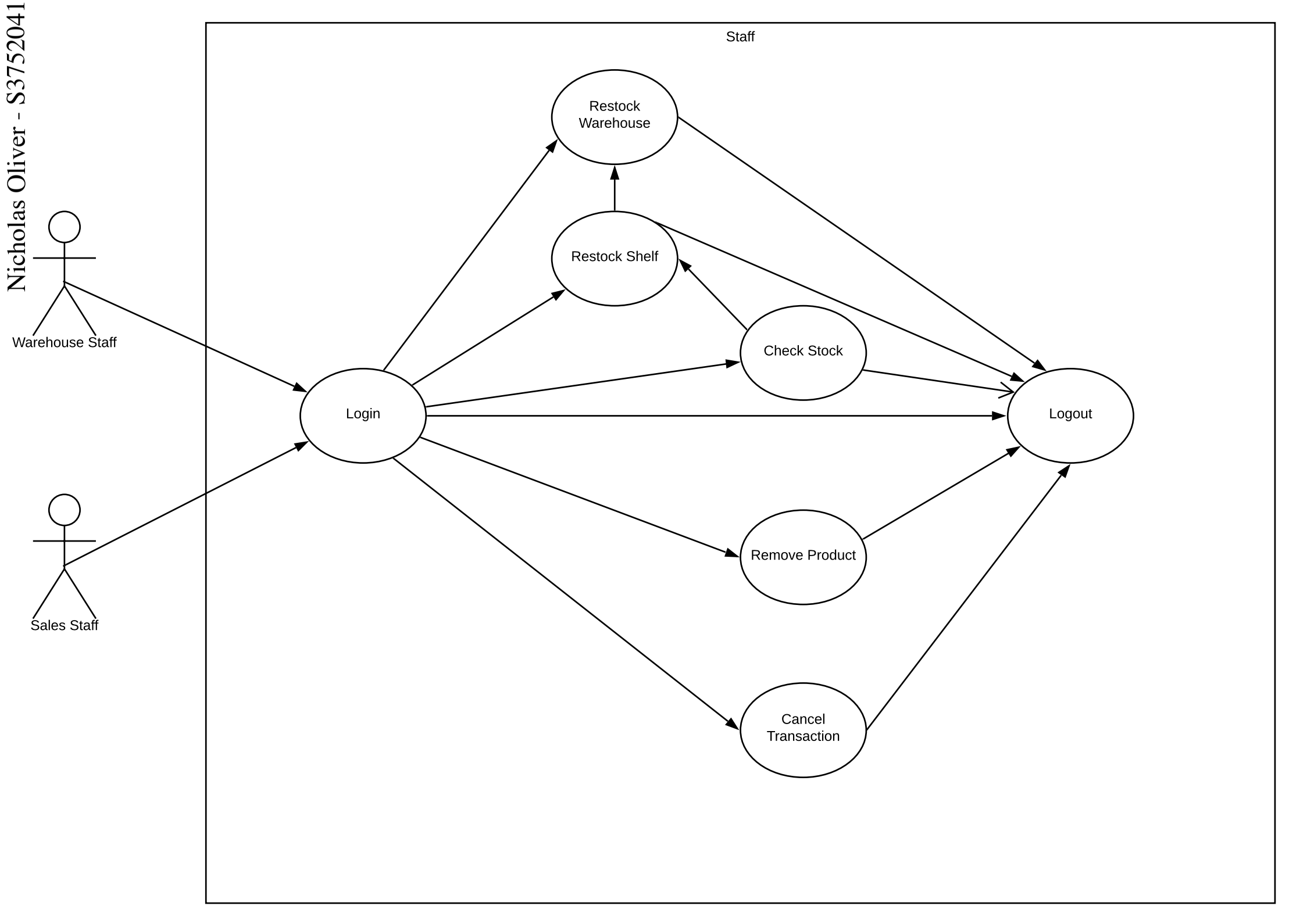
Nicholas Oliver - S3752041

State Diagram

s3752041 | May 29, 2019



Nicholas Oliver - S3752041



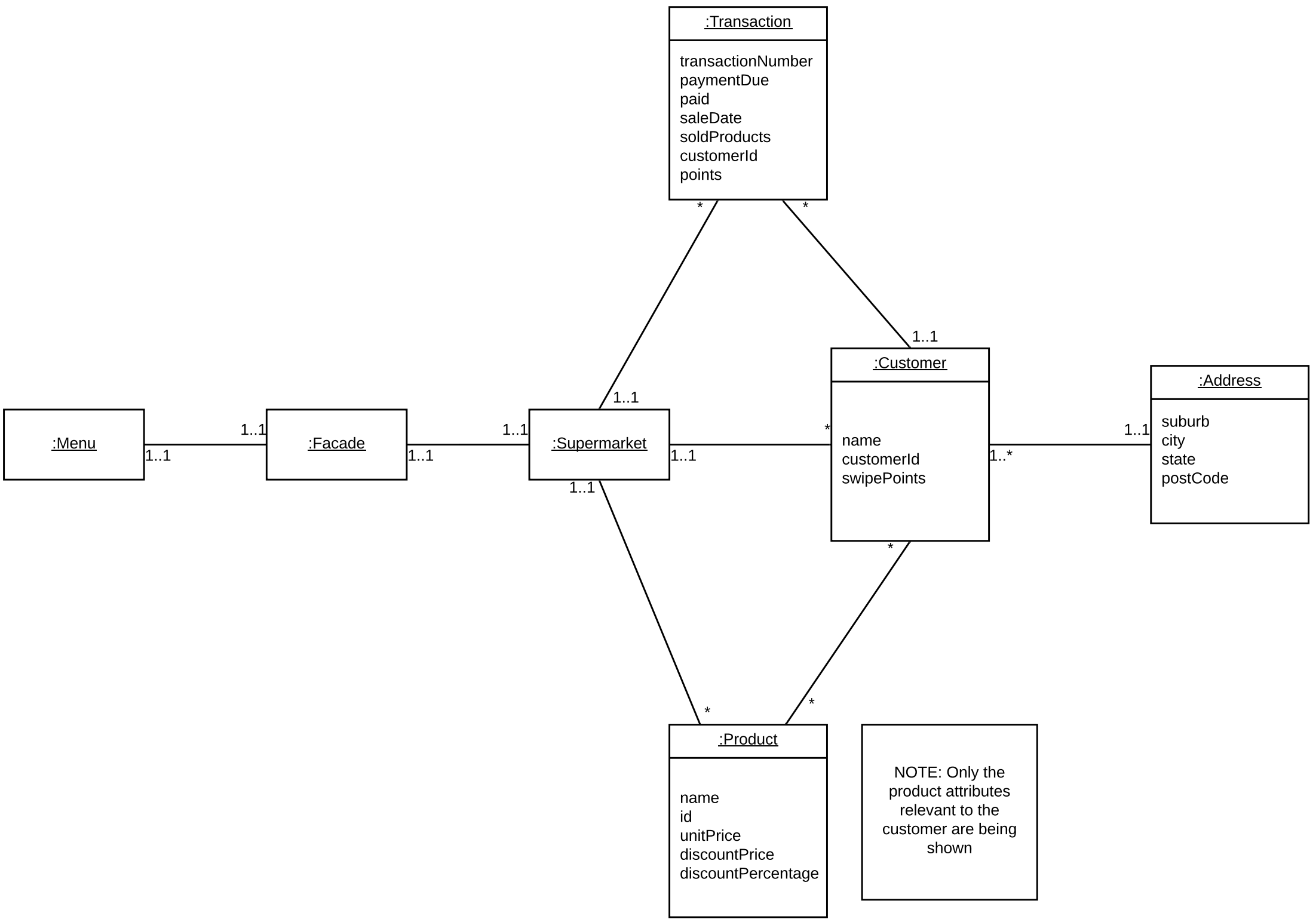
SHREY PAREKH (S3710669)

This subsection contains Shrey Parekh's individual contributions to this project. In terms of coding Shrey was responsible for creating the interface, façade, customer and transaction classes. There was also help provided on editing and adding methods to the menu, supermarket and façade classes. In the following pages the following will be shown respectively: textual description of the use case diagram, object diagram, sequence diagram, activity diagram, state diagram and a mini use-case diagram created for milestone 1. Shrey was the scrum master for this project.

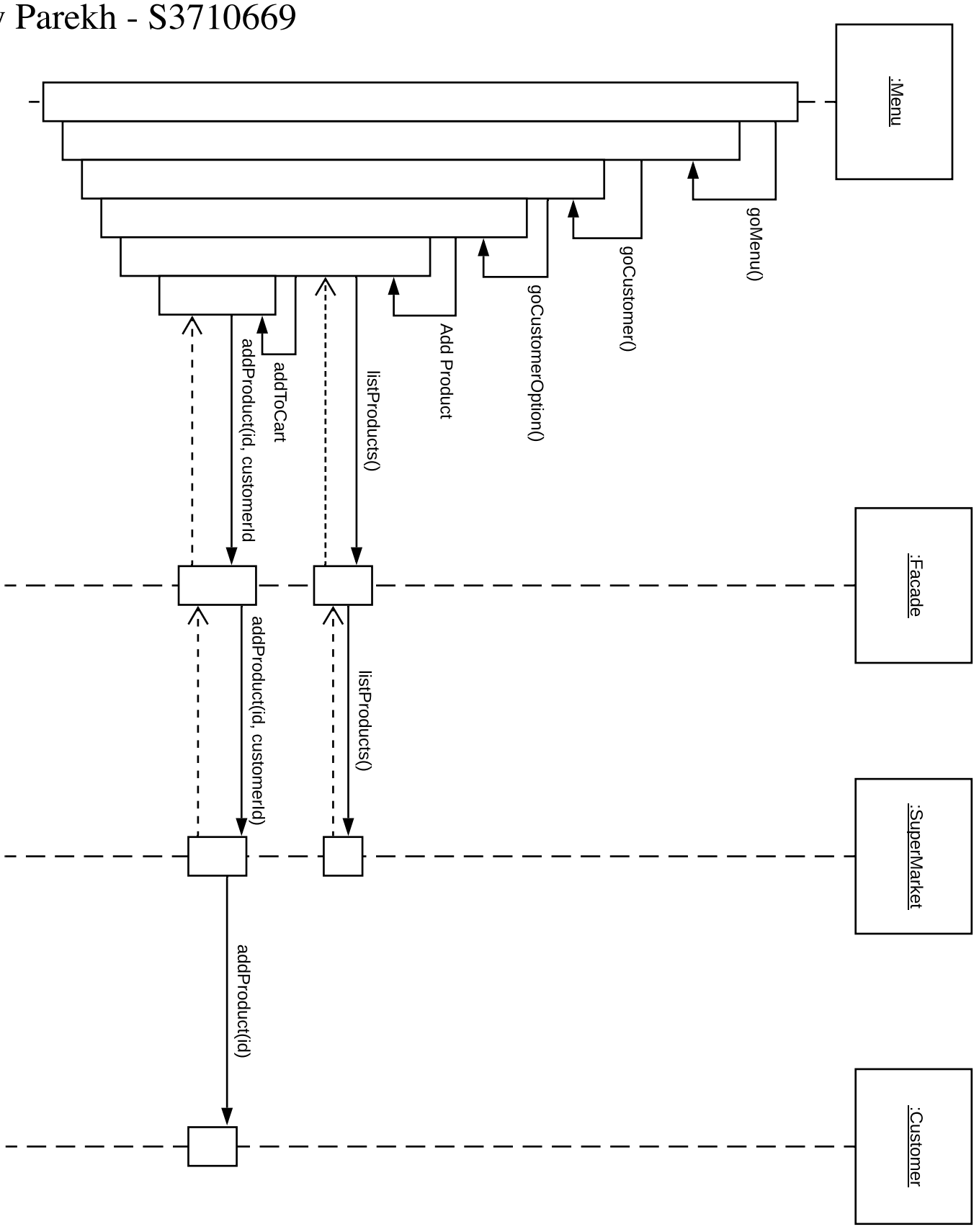
Textual Description for Use – Case Diagram

By: Shrey Parekh – S3710669

Name:	UC1: Adding Product to Cart
Goal:	A customer wants to add a product to his/her cart.
Actor:	Customer
Description:	After the program is started, the customer will select customer login in order for the system to access his/her account. Following the login, the customer will be provided with options to choose from. In order to add a product to the cart the respective option needs to be chosen.
Pre-Conditions	Customer must exist in the supermarket database. Customer must be logged in to use the supermarket database. (Assume authentication is done by fish level use case)
Trigger	Customer chooses the "add product to cart" option from the customer menu
Basic Course of Events	1. Customer chooses to view the customer options 2. System displays customer options 3. Customer selects add to cart option 4. System displays all products available in the supermarket 5. Customer inputs product ID of the product wanted 6. System validates id inputted 7. System adds the product to his/her cart 8. End of Use Case
Alternative Course of Events	2a.Returning to Main Menu 1. Customer chooses to view the customer options 2. System displays customer options 3. Customer selects to sign out 4. End of Use Case 3a.Product id entered wrong 1. Customer enters incorrect product id 2. System displays error message 3. Customer is brought back to customer options 4a. There is not enough stock of the product 1. The product chosen does not have enough stock 2. System displays error message 3. Customer is brought back to the customer options
Post-Conditions	Customer's cart holds all the products the customer desired and now he/she can proceed to checkout and pay.

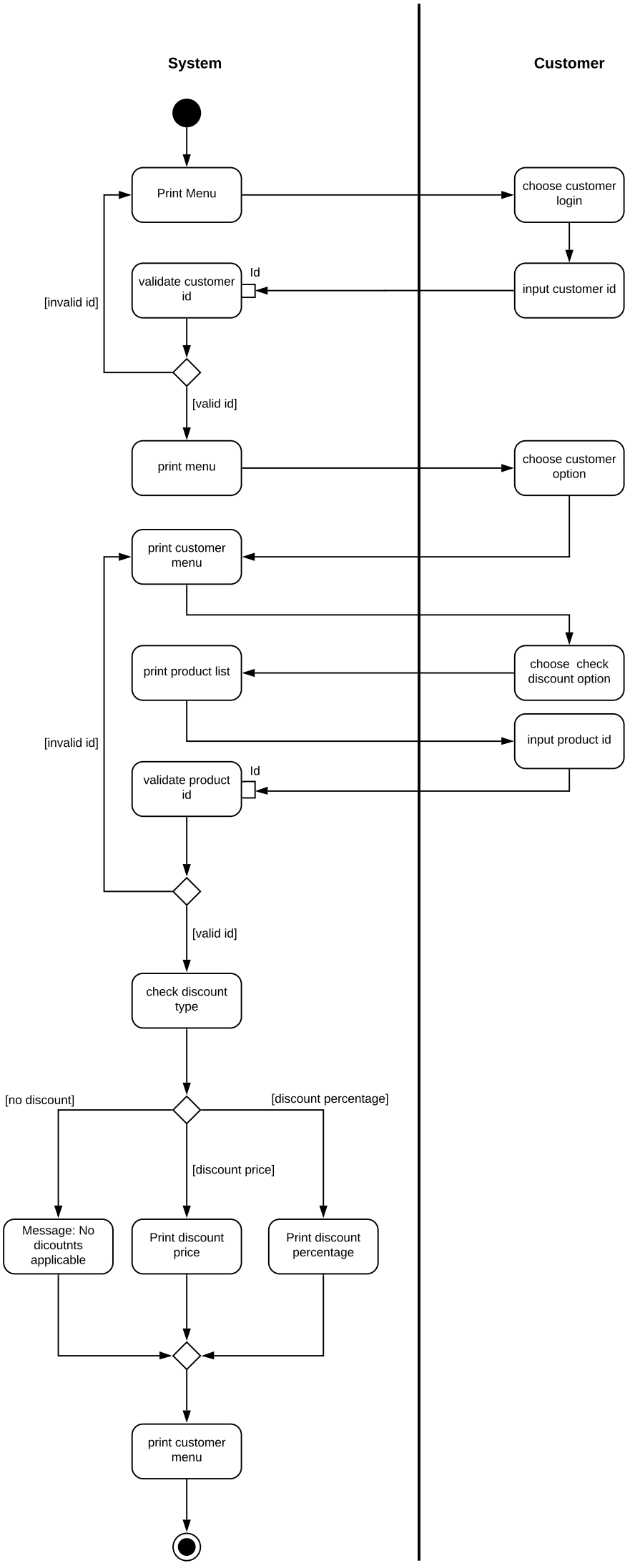

:Transaction
transactionNumber
paymentDue
paid
saleDate
soldProducts
customerId
points
:Menu
:Facade
:Supermarket
:Customer
name
customerId
swipePoints
:Address
suburb
city
state
postCode
:Product
name
id
unitPrice
discountPrice
discountPercentage
NOTE: Only the product attributes relevant to the customer are being shown
1..1
*

Shrey Parekh - S3710669

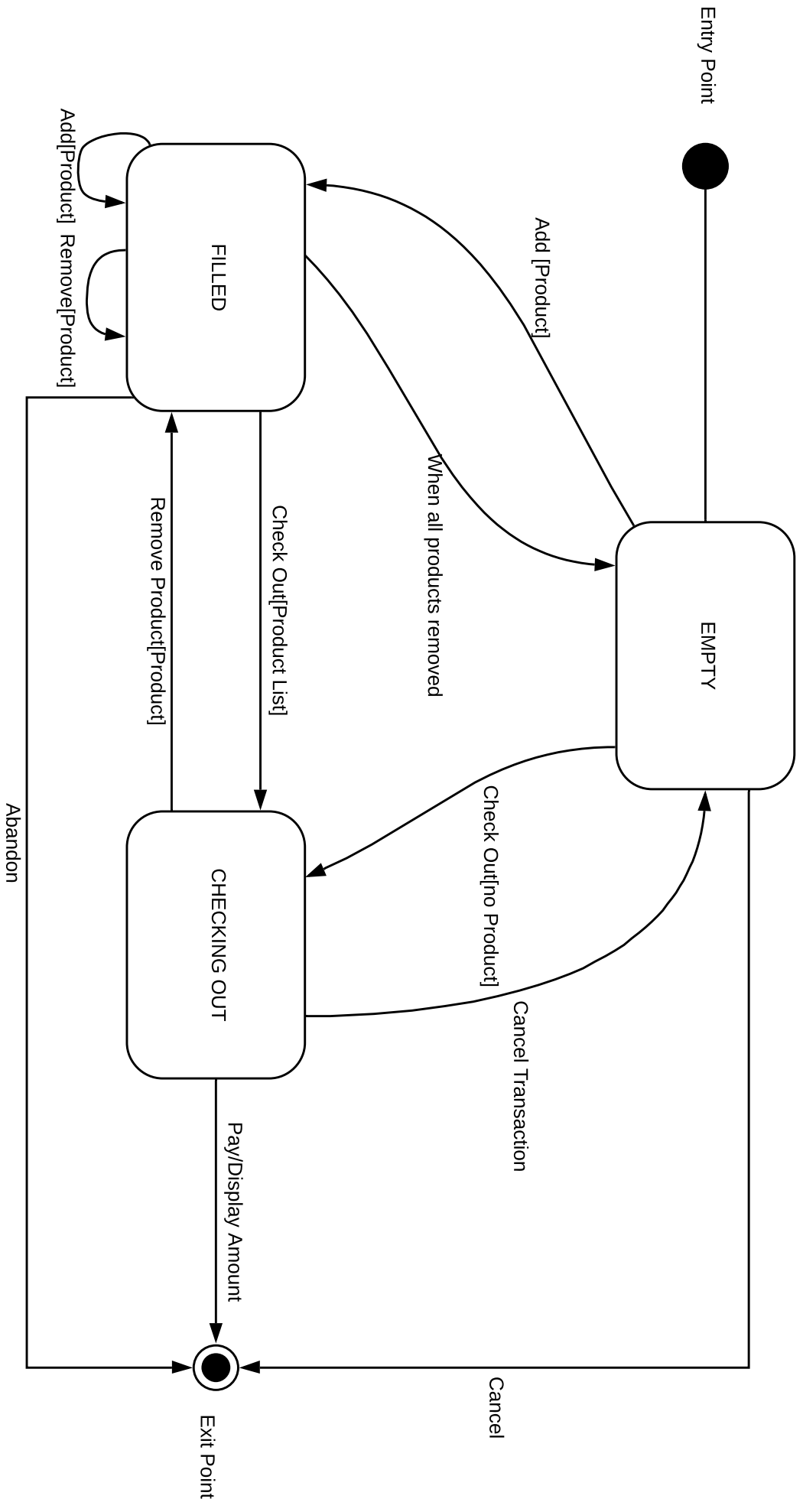


This diagram would not be a part of other diagrams. However, this diagram would be used after diagrams involving cutomers checking for product price and discounts on products. This is since our program only enables the user to do one function from the menu at a time, that is once the price or discounts are checked the user would need to select add to cart seperately.

Shrey Parekh - S3710669



Shrey Parekh - S3710669



Shrey Parekh - S3710669