Assignment 3

1. Use Cases and Use Case Diagram

1. Use Case: Create or Edit Customer's Account

Allows customer or employee to add information about a new customer account to the database or to edit existing account and update customer information.

- Customer or employee types in the customer information.
- A new customer account is created if customer ID does not exist
- If customer ID exists, account information with given ID is updated.

2. Use Case: Track package

- Customer types tracking number of package to be tracked.
- System prints out the inventory status of a package based on tracking number.

3. Use Case: Send Message

Allows customer to send a message to the shipping store with any possible questions.

- Customer types in the information about themselves and the message.
- A new message is placed in the new messages queue.

4. Use Case: Reply to Customer's Message

- The next unanswered message is pulled from the new messages queue and displayed to the employee.
- Employee types in the response.
- Response, along with the original customer message, is stored into the answered messages pool.
- An e-mail is generated and sent to the customer.

5. Use Case: Create Shipping Transaction

Allows customer or employee to create a new shipping transaction.

- Customer or employee types in the package information (type, specification, etc.)
- A new shipping transaction record is created and stored in the system.

- New accounting record is created and added to the database (Use Case: **Update Accounting**).
- *(Optional)* When a shipping transaction is created, customer or employee can optionally print a shipping label of the package (Use Case: Print **Shipping Label**).
- *(Optional)* When a shipping transaction is created, customer or employee can optionally print a receipt of the transaction (Use Case: **Print Receipt**).

6. Use Case: Print Shipping Label

- System prints out shipping label of a package.

7. Use Case: Print Receipt

- System prints out receipt of a transaction.
- (Optional) System prints out all transactions

8. Use Case: Update Inventory

System updates inventory information.

- A new package is added or an existing package is removed depending on if an employee collected or delivered a package.

9. Use Case: Collect Package

Allows employee to update package shipping information when package is collected.

- Employee types in the tracking number of package.
- System reads current date and time.
- Time information is updated to the inventory record of given package ID as Shipping Date (Use Case: **Update Inventory**).

10. Use Case: Deliver Package

Allows employee to update package shipping information as delivered.

- Employee types in the tracking number of package.
- System reads current date and time.
- Time information is updated to the inventory record of given package ID as Delivery Date (Use Case: **Update Inventory**).

11. Use Case: Update Accounting

The status of the accounting is updated when a new income or expense occurs. Shipping transactions results in new income, whereas payments result in new expenses.

- *(Optional)* When the accounting information is updated, administrator can optionally print a form to have a hard copy of the change (Use Case: **Print Financial Document**).

12. Use Case: Pay Expenses

Allows Administrator to pay expenses to employees.

- A new accounting record is added to the database (Use Case: **Update Accounting**).

13. Use Case: Print Financial Document

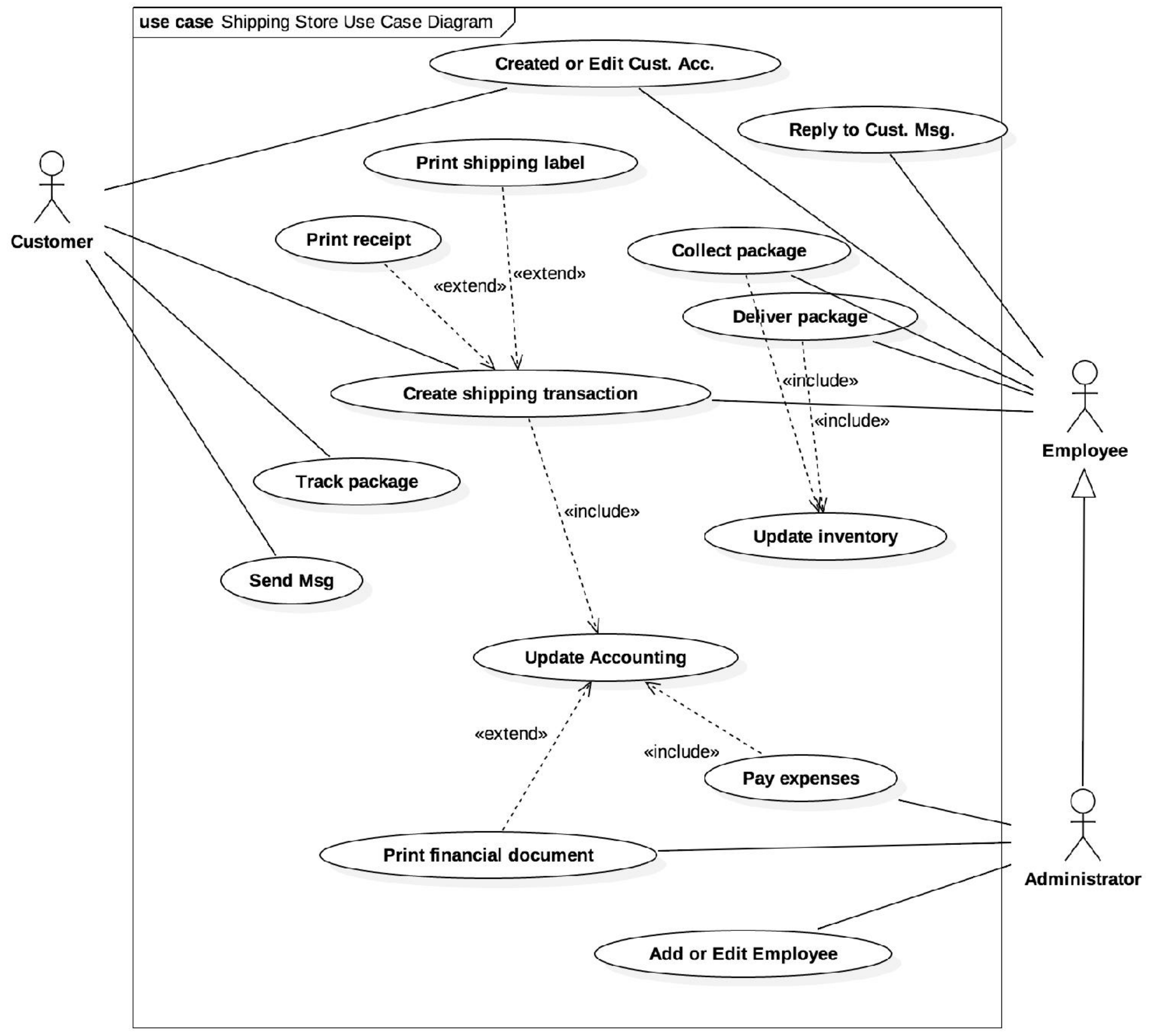
Allows Administrator to print accounting documents (Balance Sheets, Cash Flow, etc.)

14. Use Case: Add or Edit Employee’s Account

Allows administrator to add information about a new employee account to the database or to edit existing employee account information.

- Administrator types in the employee information
- A new employee account is created if employee ID does not exist, or account information with given ID is updated.

Use Case Diagram:



2. CRC Card and Class Diagram

Create CRC cards from the well-defined elements, e.g. three different types of users:

Customer	
- Manage information specific to customer	

Employee	
- Manage information specific to employee	

Administrator	
- Manage information specific to administrator	

Several information fields (e.g. user id, name) are the same for all users: create a superclass named “User” to manage that information.

User	
- Manage information common to all user types	

User records are managed by a separate class

UserRecords	
- Allow customer to create/update his own account. - Allow employees to add/update any customer’s account. - Allow administrators to edit employee’s account.	User

Same logic can be applied to types of packages

Envelope	
- Manage information specific to envelop	

Box	
- Manage information specific to box	

Crate	
- Manage information specific to crate	

Drum	
- Manage information specific to drum	

Where a a superclass named “Package” manages information common for all types

Package	
- Manage information common to all package types	

Inventory class is the class for maintaining all packages and delivery records. Also, since we need to print receipts, this class will also associated to transaction records which we will add later.

Inventory	
- Maintain a list of all packages in the inventory. - Allow customer or employee to add a new package record. - Allow employee to update existing package records.	- Envelope - Box - Create - Drum - Package

Transaction records imply a class for package delivery transactions

ShippingTransaction	
- Holds information about one package shipping transaction	

TransactionRecords class maintains an active list of transactions

TransactionRecords	
- Maintain a list of transactions	- ShippingTransaction

Accounting manager class handles various financial records, including all income from transactions and expense from salary.

Accounting	
- Keep financial information	- Expense
- Print forms	- TransactionRecords

Expense	
- Keep financial information about an expense	

Web interface allows customers to leave a message: “Message” class

Message	
- Manage message content	

MessagePool class handles storing and accessing messages

MessagePool	
- Manage messages - Allow customers to send message to the store - Allow employees to reply unanswered messages.	- Message

ShippingStore: main controller of all separate inventories and functions. Shipping Store is in charge of creating and maintaining instances of the above classes.

ShippingStore	
- Instantiates the different parts of the system.	- UserRecords - Inventory - MessagePool - Accounting - TransactionRecords

Customers can perform operations through a web interface: CustomerInterface

CustomerInterface	
- Send a Message - Create Shipping Transaction - Track pagckage	

Employees have their own interface which perform different functionalities compared to the that of customers.

EmployeeInterface	
- Reply to a customer Message - Create Shipping Transaction - Add/Edit customer info - Collect package	

EmployeeInterface	
- Deliver package	

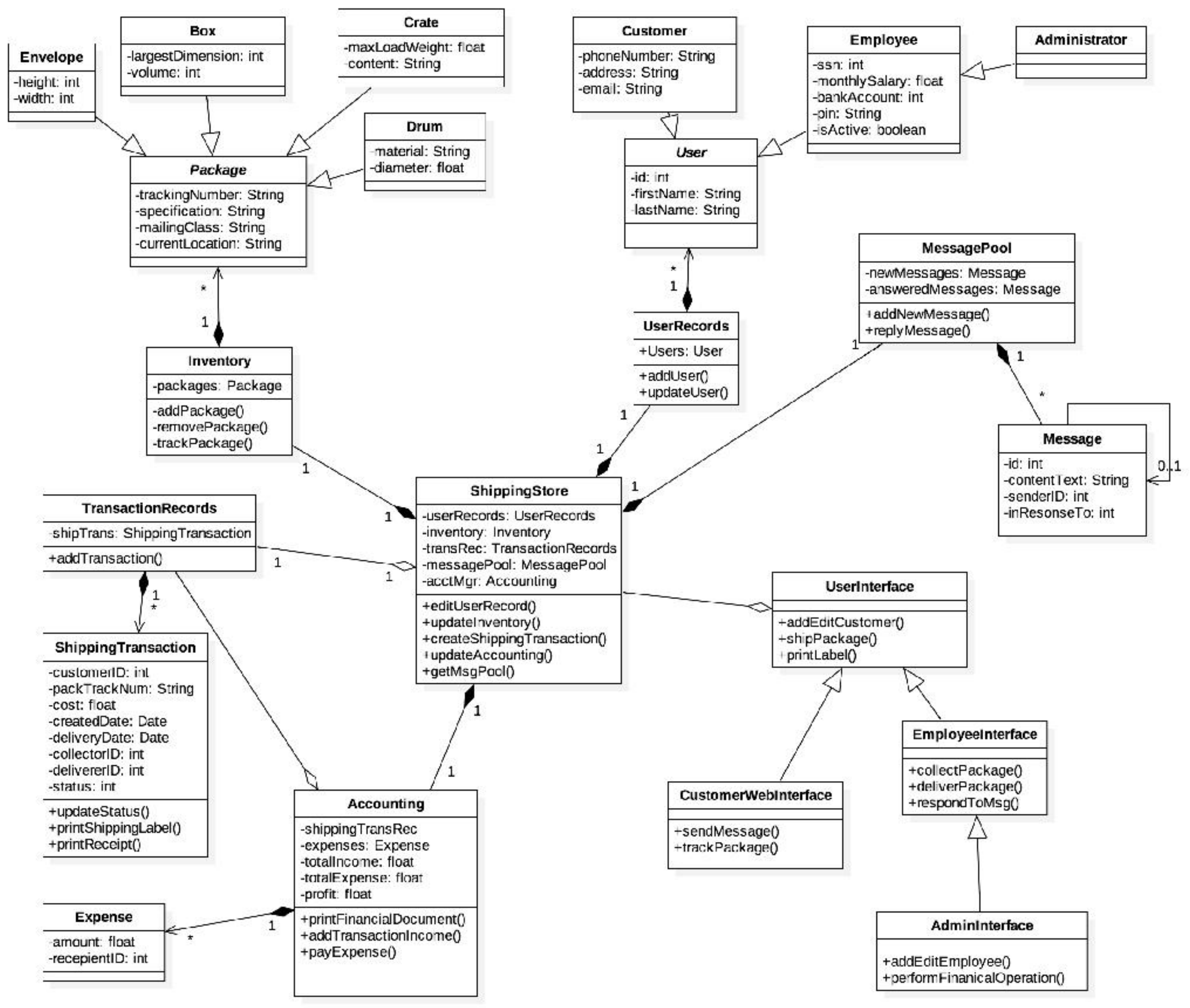
Administrators have their own interface which have some extra functionality compared to simple employees.

AdminInterface	
- Perform all functions of simple employees - Add/Edit employees - Perform Financial operations	

Some functionality is common among all users. Thus, we can create a superclass to avoid replicating the functionality. The common functionality can be added here and be removed from the subclasses.

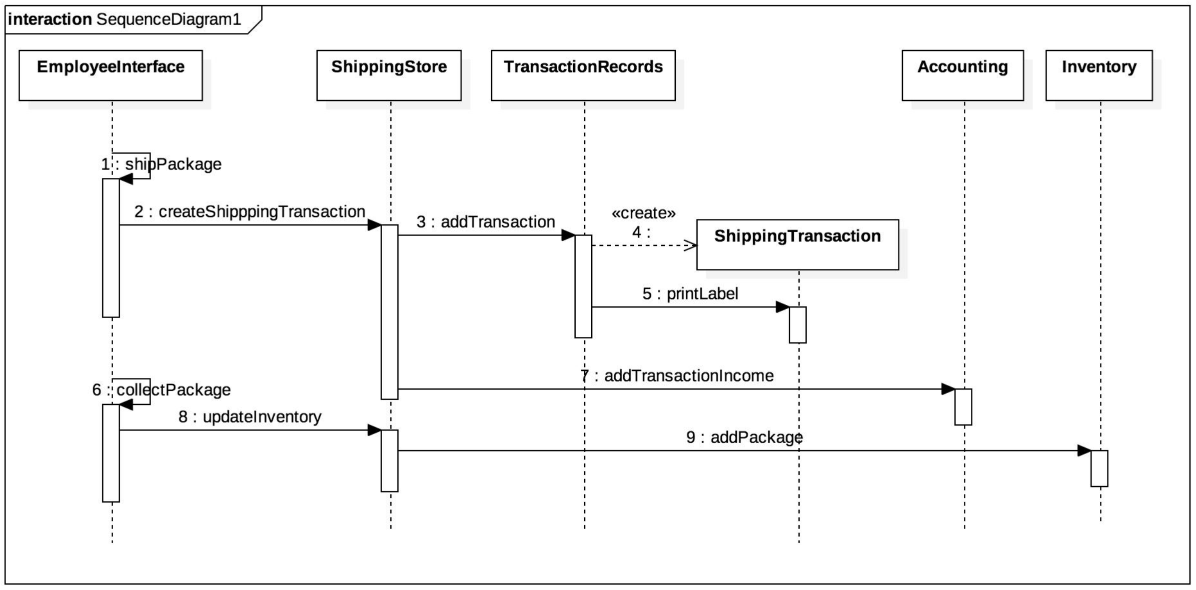
UserInterface	
- Create Shipping Transaction - Add/Edit customer info	- ShippingStore

Class Diagram:



3. Sequence Diagram of “Employee Creates Transaction and Collects Package”

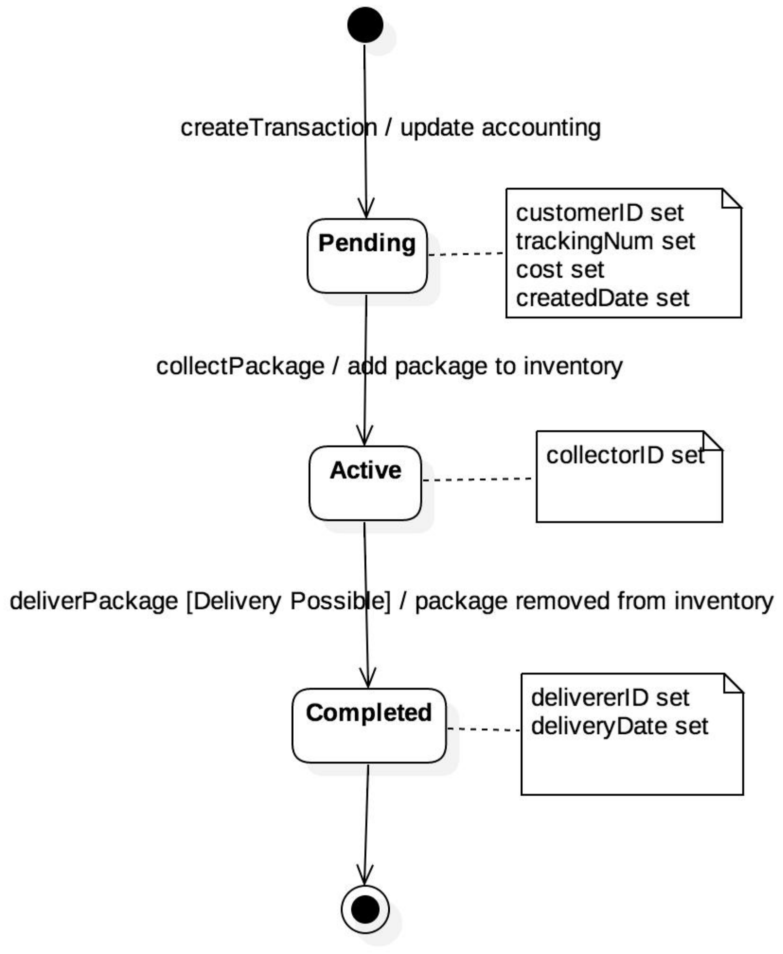
The method names used in the sequence diagram come from the class diagram.



4. Transaction State Machine Diagram

The state machine uses the “status” of the transaction as its main states. Notes associated with each state indicate the status of the other variables of the object.

The transitions indicate the event that caused the transition, possible guard associated with the transition and the actions that take affect when that transition happens, following the notation: **event [guard] / action**


createTransaction / update accounting
Pending
customerID set
trackingNum set
cost set
createdDate set
collectPackage / add package to inventory
Active
collectorID set
deliverPackage [Delivery Possible] / package removed from inventory
Completed
delivererID set
deliveryDate set