Write a results document, test script, java doc, at least 2 sequence diagrams, a usecase diagram, how requirement is implemented, all of the questions in the grade sheet need to be answered.

More test scripts if time permits and more scenarios if time permits

Document the changes made to SMS in design doc. Document adding move method to robots to move them and ability to open/close turnstiles

Do the add to basket and update inventory orchestration from SCS rather than SMS as in assignment 2.

Calculate inventory count based on every item removed and added to basket both in the general inventory map and in the shelf. Prod10 and others are in multiple shelves so account for that. Also add the product to a customer’s basket

Turnstiles don’t remain open and they are closed after customer leaves. Make sure turnstiles are closed after emergency is resolved

Need to modify appliance command scripts

Result doc - When a product is dropped, update the inventory count since it is no longer for sale

When customer’s evacuated, think about how to update their location to out of store and time last seen

Result doc – I am making assumption that customers put products back to shelves where they found them. Removes from shelf also means add to basket according to my assumption

Review assignment 1 and 2 solutions comments

Is the Robot going to take a product from store room and restock it on the floor?

Logging events and actions is mandatory

Add dynamic location for a robot and assign a task to the closest robust, don't fixate them to one aisle

Make printing more descriptive

Result doc – I left appliance and sensor events as they are and assumed that SCS is not interested in listening to those events

Attend Tuesday section and ask questions

Have a linkedlist of observers to optimize adding/removing efficiency

Design doc risk - Cascade of events looping and persistence

Talk about requirements in implementation details and document the choices you made

Think about commands and events from SMS

Use correct arrows for relationship between subject and observer

Document how to run the scripts both in design doc and results doc

Design doc – SCS has dependency on appliances through SMS so it is not a hard dependency

Class dictionary – list out what the execute method does for each command

Commands and observers must be colored differently. Write notes to indicate the use of design patterns in the class diagram

Partially show Store Model Service to show subject and some of the classes in the class diagram

Show emergency and check out and maybe one more in a sequence diagram

It maybe more than one command for an event. CommandFactory can return a list of commands

When you have multiple instances in a sequence diagram, have a note that says for each

Find the closest distance when sending robots A1 and A2 are consecutive. B1, B2 and B3 are also consecutive

Stop directly updating customer’s location as in assignment 2

Result doc - When customer requests something, force adding to basket

Result doc – I am assuming that customer picks up one product at a time so I didn’t include counts but instead events will be generated for each product picked up

Handle SMS exceptions in SCS

Exception classes should be with customer properties

APIs must be modified with auth\_token – check instructor’s response on Piazza

1 usecase diagram, 2 sequence diagrams and a class diagram indicating command and observer patterns

Results document should be turned in separately and should include the peer review documents including explanations how the feedback helped me

Does the design address the requirements should be answered in results doc. Cite the design choices corresponding to requirements

Attach the output from test

Don’t forget java doc for all 3 services

Result doc – I added some convenience getters

Design doc – concrete commands are immutable once created

Sequence diagram

*\* This extends the reuirement to make it more realistic. The steps are  
\* 1 - customer requests an item  
\* 2 - The nearest robot gets assigned to find item for customer  
\* 3 - Robot moves to the aisle where the product is located  
\* 4 - Robot moves back to the customer and hands in the product of however many was requested  
\* 5 - Customer's basket gets updated with the product and the corresponding count  
\* 6 - Inventory count gets updated to reflect the item the customer picked up*

Checkout will also be good for sequence diagram and also basket changed command

Result doc – I am assuming there will only be one person with the same name at a given time when finding a missing person. If there is more than one person and the customer looking for a missing person is not satisified, another command shall be sent again to find the missing person

Design doc - The speaker’s echo important announcements only and the rest is logged for information

Customer removes one item at a time and the robot restocks right away

I am using a queuing mechanism to avoid chattiness

May need to add a logger jar

Include peer review comments

Review grade sheet before submitting

Write how it follows design patterns

Mention queuing mechanism for extra credit

Result doc - I have avoided circular dependencies

Design choices must be in design doc not results doc

Stop clearing basket association

Rename folder to mussa\_tofik\_hw3, delete .class files, out folder and iml junk. Zip it

Generate toString()

*\* It is assumed that the customer removes one item at a time. The inventory gets updated to reflect for the count that  
\* got put back and then a robot takes the item from floor to store room because it is assumed that the shelf was  
\* stocked already*

*==========*

*An extension from the requirements is when a product is dropped, inventory count gets updated since  
\* that particular item will no longer be for sale*

*I kept the events from assignment but SCS is not interested in those. Only the specified ones SCS is listening to*





