Question 1 Notes

# Use Case Process

1. User selects “ProcessShoppingListDelivery” option from menu
2. Systems prompts user to specify id of shopping list
3. User enters id
4. System connects to cloud storage
5. System locates shopping list from id given
6. System prompts user to scan items
7. For each scanned delivered item:
   1. User scans barcode of delivered items
   2. System stores barcode in session.barcode
   3. System sets session.matchingFound to false
   4. For each item held in memory of fridge:
      1. System places current item in session.item
      2. System compares barcode of item (session.item) with barcode held in session.barcode
      3. If the item is found:
         1. System prompts user to type in quantity of goods held in delivered package
         2. User types in quantity
         3. System updates quantity of session.item item with quantity that user provided
         4. System sets session.matchingFound = true
      4. If no item with matching barcode found:
         1. System submits a query to cloud storage
         2. Query asks for item location with barcode matching session.barcode
         3. Cloud locates the item and returns the details for that item
         4. System creates new item using returned details
         5. Attribute “qty” of newly created item is set to 0
         6. System stores reference to newly created item into session.item
         7. System asks user to input quantity
         8. User types in amount
         9. System updates quantity of item (session.item)
8. Systems creates copy of selected ShoppingList object on cloud
9. For each item of the processed ShoppingList:
   1. Systems sends copy of item to cloud storage to be added to the copy of ShoppingList
10. System deletes processed ShoppingList from fridge memory
11. System disconnects from cloud storage
12. System deletes session.item, session.barcode and session.matchingFound
13. System informs user that processing is complete

# Examples of function definitions from use cases

Function promptUser(message, function()){

Output message;

Return function()

}

Function connectToCloud()

Function findShoppingList(ShoppingListID)

Function scanDeliveredItems(){

For each item:

scanBarcode()

storeBarcode()

session.matchingFound = false

for each item in fridge memory:

session.item = current item

if(session.item.barcode == session.barcode){

session.item.qty = promptUser(“Enter quantity: “, getQuantity())

session.matchingFound = true

}else{

Item new\_item = new Item(getItem(“barcode”, session.barcode))

New\_item.qty = 0

Session.item = new\_item

Session.item.qty = promptUser(“Enter quantity: “, getQuantity())

}

}

Function copyShoppingList()

Function updateShoppingList(items){

copyItem(item)

sendItemtoCloud(item)

}

Function deleteProcessedShoppingList()

Function disconnectFromCloud()

Function resetSessionVariables(){

Session.item = null

Session.barcode = null

Session.matchingFound = null

}

Function informUser()

# Classes and their definitions

**Supermarket**

setFavourite

**PaymentAccess**

makePayment

**AutoPreferences**

addSupermarket

**ShoppingList**

addItem

removeItem

retrieveData

setSubmitted

setDelivered

setArchiving

**Item**

No Operations

**User**

retrieveDetails

**SensorControls**

adjustAirFlow

activateAlarm

is90secDoorOpen

isDoorOpen

**BarcodeReader**

Scan

**CloudGateway**

storeData

retrieveData

connect

disconnect

**Timer**

setWhen

**Controller**

scanBarcode

addItem

withdrawItem

adjustTemp

sentOrder

editOrder

setTreashold

addNotification

findShoppingList

scanDeliveredItems

copyShoppingList

updateShoppingList

deleteProcessedShoppingList

processShoppingListDelivery

**FridgeItem**

adjustQty

adjustThreshold

getQuantity

**Engineer**

retrieveData

**GUIGateway**

addItem

withdrawItem

adjustTemp

sendOrder

editOrder

setThreshold

addUser

removeUser

addEngineer

removeEngineer

retrieveNotifications

promptUser

**LCDGateway**

turnOn

turnoff

**MobileAppGateway**

Connect

**ItemsInFridge**

Generate

**Report**

Generate

View

Store

Email

**ItemConsumed**

Generate

**AllItemsConsumed**

generate