Table of Contents

Meeting Notes

Meeting date	*Attendees*	*Meeting Type*	*Meeting notes*
9/08/22 4:10PM - 5:15PM	Catherine Argueta paul Leah Granado Christian Avila	Team Meeting	• Went over the document that includes the discussion of requirements • Created brief summary detailing requirements
9/22/22 4:30PM - 5:10PM	Xavier Howell Catherine Argueta Leah Granado paul	Team Meeting	• Assigned Project Roles • Discussed project design • Shared Tool Collaborations
10/6/22 4:30PM - 5:15PM	Xavier Howell	Class Breakout Room	• R1 Release
10/8/22 9:00AM - 9:45AM	Catherine Argueta paul Leah Granado Christian Avila	Team Meeting	• Discussed presentation ideas • Agreed to share competes on Jira • Create Google Slides • Development planning • Use cases • Discussed presentation requirements • Assigned each product design subsystem among team

10/26/22
 	
6:30PM - 7:30PM
	Xavier Howell
	
	Catherine Argueta
	
	Leah Granado
	
	paul
		Team Meeting	• Started Slides
			• Started Code
			• Discussed:
			  ○ Progress
			  ○ Presentation requirements
			  ○ Test cases

10/27/2022
 	
4:00PM - 5:15PM
	Catherine Argueta
	
	paul
	
	Leah Granado
		Class Breakout Room	• Discussed:
			  ○ the group's progress with professor
			  ○ GUI for management, customer, and stocker
			• Created emphasis on diagrams

10/29/22 9:00AM - 10:00AM	paul Xavier Howell Catherine Argueta	Team Meeting	• Updated our progress • Went over presentation details
11/7/2022 5:00PM - 6:00PM	Catherine Argueta paul Leah Granado Xavier Howell Christian Avila	Team Meeting	• Shared progress • Put together completed work • Discussed missing content • Discussed presentation

Project Plan

Risk/Issues

- Team members have conflicting time schedules
- Christian Avila was out sick for a bit earlier in development and has power and WiFi problems the week the Team delivers R1 deliverables

Brief problem statement

Some of the problems of the vending machine include expired items, recalled items, remove items that aren't selling, and adding items to the right spot. The documents are in paper form and the re-stocker has to manually count the items.

Team Members

Replace this text and the instructions below with your statement in black.
(List each team member and their role on the project.)

Xavier Howell - xavierhowell@csus.edu (Team Coordinator)

Christian Avila - christianavila@csus.edu (Development Coordinator/Release)

Paul Tung - paultung@csus.edu (Requiurment Cordinator/Release)

Leah Granado - leahgranado@csus.edu (Test Coordinator)

Catherine Argueta- cargueta2@csus.edu (Configuration/QA Coordinator)

Team Communication

The team will meet Saturdays at 9:00AM through Discord. We will use Discord to discuss the project.

Development Environment

- Jira
- Lucidchart
- GitHub
- Eclipse

Milestone Schedule

No	Tasks	Status	Assigned to	Due Date	Release	Deliverable	Dependency
1	*Project Plan*	Done	Xavier Howell	W2	R1	Yes	None
2	User Stories	Done	Catherine Argueta	W2	R1	Yes	None
3	Use Cases	Done	Catherine Argueta	W4	R1	Yes	Task 2
4	*Use Case Diagram*	Done	Catherine Argueta	W4	R1	Yes	Task 3
5	R1 Presentation Planning	In Progress	@everyone	W11	R1	No	None
6	*Detail Components and Functions*	Done	@everyone	W9	R1	No	Tasks 1-3
7	Start Code and Test Cases for R1	Done	Catherine Argueta paul	W11	R1	Yes	Task 6
8	UML Diagrams	In Progress	Catherine Argueta Leah Granado Xavier Howell Christian Avila	W10	R1	Yes	Tasks 1-3
9	GUI Mock up of Interfaces	Done	paul	W11	R1	Yes	Task 1
10	Compile Meeting Notes up to Wk11	Done	Leah Granado	W11			
11	R1 Presentation Slides	In Progress	@everyone	W11	R1	Yes	Task 5

12	Review Project Design for R2	To Do	@everyone	W12	R2	No	None
13	Complete coding features for R2	To Do	everyone	W13	R2	Yes	Task 11
14	Implement Test Cases for R2	To Do	everyone	W13	R2	Yes	Task 12
15	Complete coding for GUI	To Do	everyone	W14	R2	Yes	Task 11
16	R2 Presentation Planning	To Do	Everyone	W13	R2	No	None
17	R2 Presentation	To Do	Everyone	W15	R2	Yes	Task 15

Issue Log

[C1-1] Select Roles Created: 21/Sep/22 Updated: 22/Sep/22 Resolved: 22/Sep/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Xavier Howell	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0
Labels:	None		

[C1-5] Fill Out Project Plan Sheet Created: 21/Sep/22 Updated: 08/Nov/22 Resolved: 08/Nov/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Xavier Howell	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0

[C1-7] Contacts Created: 22/Sep/22 Updated: 22/Sep/22 Resolved: 22/Sep/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Xavier Howell	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0

[C1-8] Project Requirements Created: 22/Sep/22 Updated: 08/Nov/22 Due: 01/Oct/22	
Status:	Done
Project:	CSC 131

Type:	Epic	**Priority:**	Medium
Reporter:	Xavier Howell	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0

Time Spent:	2 Weeks
Start date:	16/Sept/22

[C1-10] Project Requirements - Draw UML use case context diagram Created: 01/Oct/22 Updated: 08/Nov/22 Resolved: 07/Nov/22

Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Catherine Argueta
Resolution:	Done	**Votes:**	0

[C1-11] Project Requirements - Fill out Use Case Description Table Created: 02/Oct/22 Updated: 22/Oct/22 Resolved: 22/Oct/22

Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Catherine Argueta
Resolution:	Done	**Votes:**	0

[C1-12] Product Requirements - User Stories Created: 04/Oct/22 Updated: 04/Oct/22 Resolved: 04/Oct/22

Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0

[C1-13] Discuss Product Design Created: 04/Oct/22 Updated: 08/Nov/22

Status:	In Progress
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Unassigned
Resolution:	Unresolved	**Votes:**	0

[C1-15] Product Design - Components & Functions(Vending Machine)

Created: 04/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22

Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Xavier Howell
Resolution:	Done	**Votes:**	0

[C1-16] Product Design - Components & Functions(Customer)

Created: 04/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22

Status:	Done
Project:	CSC 131
Components:	Customer

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Leah Granado
Resolution:	Done	**Votes:**	0

[C1-17] Product Design - Components & Functions(Restocker)

Created: 04/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22

Status:	Done
Project:	CSC 131
Components:	Restocker

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	paul
Resolution:	Done	**Votes:**	0

[C1-18] Product Design - Components & Functions(Management)

Created: 04/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22

Status:	Done
Project:	CSC 131
Components:	Management

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Catherine Argueta

Resolution:	Done	**Votes:**	0

[C1-19] Activity Tracker/Meetings Created: 07/Oct/22 Updated: 08/Nov/22	
Status:	In Progress
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Xavier Howell	**Assignee:**	Xavier Howell
Resolution:	Unresolved	**Votes:**	0

[C1-20] Team Evaluation Created: 07/Oct/22 Updated: 07/Oct/22	
Status:	To Do
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Xavier Howell	**Assignee:**	Unassigned
Resolution:	Unresolved	**Votes:**	0

[C1-21] Sequence Diagrams Created: 27/Oct/22 Updated: 08/Nov/22 Resolved: 27/Oct/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Catherine Argueta
Resolution:	Done	**Votes:**	0

[C1-22] Activity Diagrams Created: 27/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Leah Granado
Resolution:	Done	**Votes:**	0
Original estimate:	Not Specified		

[C1-23] State Chart Diagram Created: 27/Oct/22 Updated: 08/Nov/22	
Status:	In Progress
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Xavier Howell
Resolution:	Unresolved	**Votes:**	0

[C1-24] Component Diagram Created: 27/Oct/22 Updated: 08/Nov/22 Resolved: 07/Nov/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Catherine Argueta
Resolution:	Done	**Votes:**	0

[C1-25] Deployment Diagram Created: 27/Oct/22 Updated: 08/Nov/22	
Status:	In Progress
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Christian Avila
Resolution:	Unresolved	**Votes:**	0

[C1-26] Class Diagram Created: 27/Oct/22 Updated: 08/Nov/22	
Status:	In Progress
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Christian Avila
Resolution:	Unresolved	**Votes:**	0
Original estimate:	Not Specified		

[C1-27] Customer GUI Mock up For Presentation Created: 27/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	paul
Resolution:	Done	**Votes:**	0

[C1-28] Restocker GUI Mock up For Presentation Created: 27/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	paul
Resolution:	Done	**Votes:**	0

[C1-29] Management GUI Mock up For Presentation Created: 27/Oct/22 Updated: 08/Nov/22 Resolved: 08/Nov/22	
Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	paul
Resolution:	Done	**Votes:**	0

[C1-30] Project Design Created: 08/Nov/22 Updated: 08/Nov/22 Due: 26/Oct/22	
Status:	To Do
Project:	CSC 131

Type:	Epic	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0

Time Spent:	3 Weeks
Start date:	04/Oct/22

[C1-31] Code and Test Cases For R1 Features

Created: 08/Nov/22 Updated: 08/Nov/22 Resolved: 08/Nov/22

Status:	Done
Project:	CSC 131

Type:	Task	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Unassigned
Resolution:	Done	**Votes:**	0

[C1-32] Project R1 Features and Deliverables

Created: 08/Nov/22 Updated: 08/Nov/22 Due: 08/Nov/22

Status:	To Do
Project:	CSC 131
Components:	None
Affects versions:	None
Fix versions:	None

Type:	Epic	**Priority:**	Medium
Reporter:	Catherine Argueta	**Assignee:**	Unassigned
Resolution:	Unresolved	**Votes:**	0
Time Spent:	2 Weeks		
Start date:	22/Oct/22		

Project Requirements

Brief problem statement

The clients vending machines only have basic functions. They don't have the capacity to be able to report back to corporate on things like its current inventory and purchase history. The restockers who service the vending machines also do more work than they should. They are the ones who check for expired items, have to manually count things and decide where items should be placed which should be the job of corporate. The restockers also have to report back to corporate, but because its paper based, it is expensive and slow which leads to big lapses in time between reports about individual vending machines.

Stakeholders

- Dan(Owner)
- Restocker
- Corporate
- Team Ravenclaw

Users profile

The three users who will using the system are: customer, restocker, and management. The customer will be able to interact with the system through a touch screen, the restocker will be able to interact with the system through a mobile app on a smartphone or tablet, and management will be able to interact with the system through a PC application on a corporate computer.

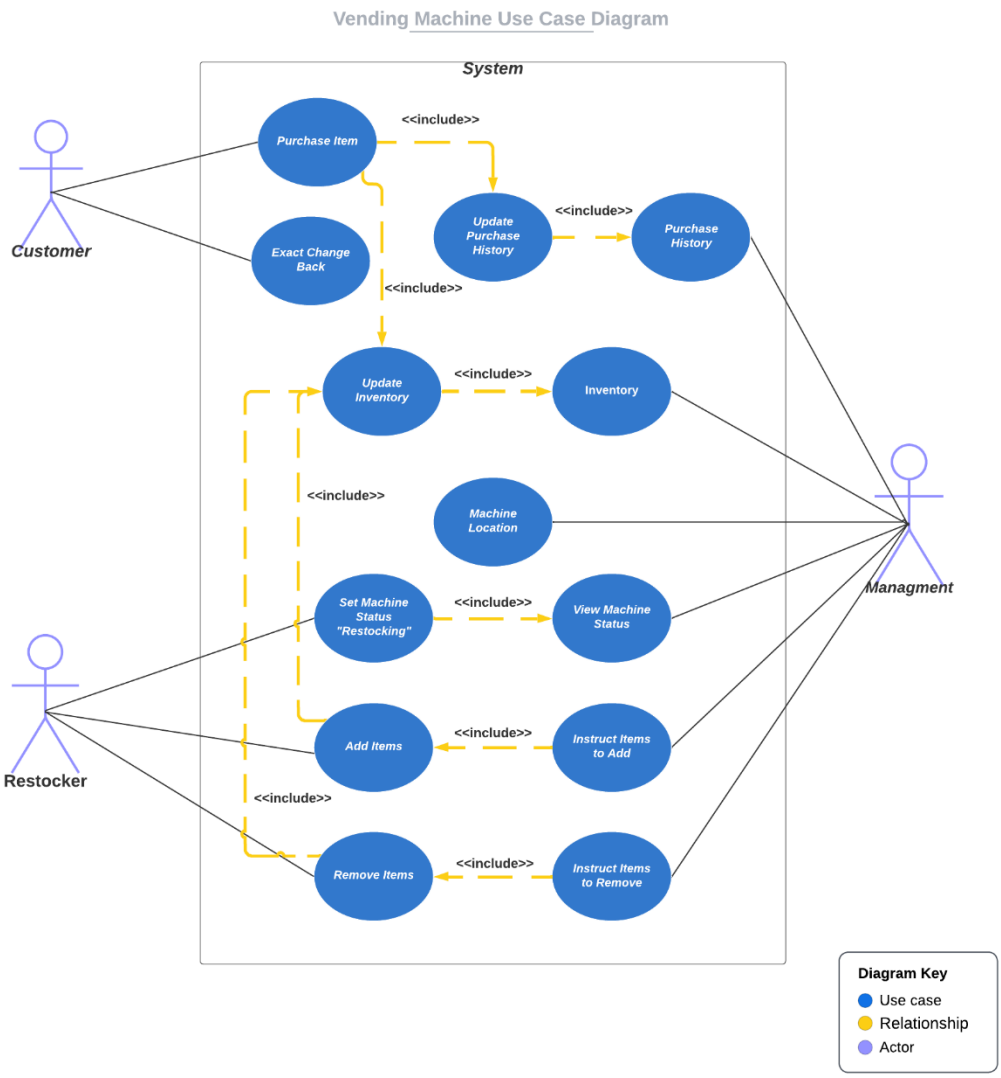
System requirements

For our solution, the language we will be using is Java.

Feature requirements (user stories)

No.	User Story Name	Description	Release
1	Purchase Item	As a customer, I want to be able to purchase my item	R1
2	Receive Change Back	As a customer, I want to be able to receive the correct change back after receiving the item I purchased	R1
3	Remove Items	As a restocker, I want to be told what needs to be removed	R2
4	Add Items	As a restocker, I want to be able to be told what items to add and where in the machine	R2
5	See Inventory	As management, I want to be able to see the inventory of each vending machine	R1
6	See Machine Location	As management, I want to be able to see where the vending machine is located	R2
7	See Machine Status	As management, I want to be able to see the status of the machine. Whether its online, offline, or being currently restocked	R1
8	Set Items to Remove	As management, I want to be able to dictate what items will be removed	R2
9	Set Items to Add	As management, I want to be able to dictate what items will be added and where in the machine	R2
10	Purchase History	As management, I want to be able to see the purchase history of each machine	R2
11	See Machine Analytics	As management, I want to be able to see the which items are more popular in each machine	R2

Use case diagram



Use case description

Use Case Number:	**UC-01**
Use Case Name:	**Purchase Item**
Overview:	**This use case starts when the customer wants to purchase an item from the vending machine. It ends after the customer receives their item.**
Actor(s):	· **Customer**
Pre condition(s):	**Vending machine must be online(not offline or being restocked)**
Scenario Flow:	**Main (success) Flow:** 1. Customer puts money into the vending machine 2. The vending machine prompts the customer to choose an item 3. The customer chooses an item 4. The vending machine verifies the item is in stock 5. The vending machine verifies that customer inserted enough money 6. The vending machine gives the customer the item 7. Customer takes the item
Post Condition:	**Customer receives correct item they purchased**
	Alternate Flow #1: 1. Customer puts in the item they want to purchase 2. The vending machine informs customer the item is not in stock 3. The vending machine cancels the order 4. The customer can then choose another item
Post Condition:	**Customer can choose another item**

	Alternate Flow #2 1. The customer chooses an item 2. The vending machine verifies the item is in stock 3. The vending machine verifies the customer didn't insert enough money 4. The vending machine informs the customer they didn't insert enough money 5. The vending machine prompts the customer to add more money
Post Condition:	**Customer can add more money**

Use Case Number:	**UC-02**
Use Case Name:	**Receive Exact Change Back After Purchase**
Overview:	**This use case starts when the customer receives item purchased. It ends after receiving exact change back.**
Actor(s):	**Customer**
Pre condition(s):	**The customer must have already purchased an item**
Scenario Flow:	**Main (success) Flow:** 1. The customer receives item purchased 2. The vending machine verifies the change it needs to give the customer 3. The vending machine gives exact change back to customer 4. Customer receives exact change back
Post Condition:	**Customer receives exact change back**

Use Case Number:	**UC-03**
Use Case Name:	**Set Status of Vending Machine to "Restocking"**

Overview:	**This use case starts when the restocker wants to restock the vending machine. It ends when the restocker sets the status of the vending machine to "Restocking".**
Actor(s):	· **Restocker** · **Customer**
Pre condition(s):	**The restocker must have been instructed to restock the vending machine**
Scenario Flow:	**Main (success) Flow:** 1. The restocker verifies they have been instructed to restock the vending machine 2. They set the status of the vending machine to "Restocking" 3. The vending can not be used by customers
Post Condition:	**The status of the vending machine is set to "Restocking". The customer can't use the vending machine.**

Use Case Number:	**UC-04**
Use Case Name:	**Remove Items From Vending Machine**
Overview:	**This use case starts when the restocker receives instructions from management on what items need to be removed. It ends after the restocker completes the instructions.**
Actor(s):	· **Restocker** · **Management**
Pre condition(s):	· **The restocker must receive instructions from management to restock the vending machine** · **The restocker must set the status of the vending machine to "Restocking"**

Scenario Flow:	**Main (success) Flow:** 1. The restocker verifies the items that must be removed from vending machine 2. The restocker then proceeds to remove items from vending machine 3. The restocker than confirms with management that items have been removed 4. The restocker than proceeds to verify instructions for what items to added
Post Condition:	**The items the restocker was instructed to remove by management have been removed**
	Alternate Flows: 1. The restocker sees no items must be removed from the vending machine 2. The restocker then proceeds to verify instruction for what items to add
Post Condition:	**The restocker has verified no items must be removed**

Use Case Number:	**UC-05**
Use Case Name:	**Add Items to Vending Machine**
Overview:	**This use case starts after the restockers has finished completing the instruction given by management on what items to remove if any. It ends when the restocker completes the instructions.**
Actor(s):	· **Restocker** · **Management**
Pre condition(s):	· **The status of the vending machine is "Restocking"** · **The restocker must have completed instruction of what items to remove**

Scenario Flow:	**Main (success) Flow:** 1. The restocker verifies instructions given by management on what items need to be restocked 2. The restocker proceeds to restock items 3. The restocker verifies instructions given by management on what items to add. 4. The restocker verifies instruction given by management on where to add items 5. The restocker than proceeds to add items to their respective places 6. The restocker then confirms items with management that items have been added 7. The restocker sets the status of the vending machine back to "Online"
Post Condition:	**The vending machine's inventory has been restocked and the status is set back to "Online"**

Use Case Number:	**UC-06**
Use Case Name:	**See Inventory of Vending Machine**
Overview:	**This use case starts when management wants to check on the inventory of the vending machine.**
Actor(s):	· **Management**
Pre condition(s):	**The status of the machine must be "Online"**
Scenario Flow:	**Main (success) Flow:** 1. Management verifies the status of the vending machine to be "Online" 2. Management then requests current inventory of the vending machine 3. Management receives current inventory of vending machine
Post Condition:	**Management receives report on the current inventory of the vending machine.**

Use Case Number:	**UC-07**
Use Case Name:	**See Machine Location**
Overview:	**This use case starts when management wants to check the location of the vending machine.**
Actor(s):	· **Management**
Pre condition(s):	**The status of the vending machine must be "Online".**
Scenario Flow:	**Main (success) Flow:** 1. Management verifies the status of the vending machine is "Online" 2. Management requests location of the vending machine 3. Management receives location of the vending machine
Post Condition:	**Management receives the location of the vending machine**

Use Case Number:	**UC-08**
Use Case Name:	**See Machine Status**
Overview:	**This use case starts when management wants to check the status of the vending machine**
Actor(s):	· **Management**
Pre condition(s):	**None**
Scenario Flow:	**Main (success) Flow:** 1. Management requests status of the vending machine 2. Management verifies the status of the vending machine is "Online"
Post Condition:	**Management receives the status of the vending machine. The status of the vending machine is "Online". The vending machine is in working order**

	Alternate Flow #1: 1. Management requests status of the vending machine 2. Management verifies the status of the vending machine is "Restocking"
Post Condition:	**Management receives status of the vending machine. The status of the vending is "Restocking" meaning it is being currently restocked.**
	Alternate Flow #2: 1. Management requests status of the vending machine 2. Management verifies the status of the vending machine is "Offline"
Post Condition:	**Management receives status of the vending machine. The status of the vending is "Offline". The vending machine is currently out of order.**

Use Case Number:	**UC-09**
Use Case Name:	**Set Instructions for Items to Remove**
Overview:	**This use case starts when management wants to send instructions to the restocker on what items to remove from the vending machine. It ends after the restocker receives the instructions.**
Actor(s):	· **Management** · **Restocker**
Pre condition(s):	· **The status of the vending machine should be "Online"** · **The vending machine must be scheduled for restocking**

Scenario Flow:	**Main (success) Flow:** 1. Management verifies items that are expired 2. Management verifies items that are recalled 3. Management verifies items that are no longer saleable 4. Management creates a list of items that meet the previous criteria 5. Management then sends the list of items to the restocker with the order to remove these items from the vending machines 6. The restocker then verifies they have received the instructions
Post Condition:	**The restocker receives instructions from management on which items to remove from the vending machines inventory.**
	Alternate Flows: 1. Management verifies that there are no expired, recalled or unpopular items in the vending machines inventory 2. Management then sends instructions that no items will be removed 3. The restocker then verifies they have received the instructions
Post Condition:	**The restocker receives instructions from management that no items will be removed from the vending machines inventory**

Use Case Number:	**UC-10**
Use Case Name:	**Set Items to Add to Vending Machine Inventory**
Overview:	**This use case starts when management wants to send instructions to the restocker on what items to add into the vending machine. It ends when the restocker receives the instructions.**
Actor(s):	· **Management** · **Restocker**
Pre condition(s):	· **The status of the vending machine must be "Online"** · **The vending machine must be scheduled for restocking**

Scenario Flow:	**Main (success) Flow:** 1. Management verifies slots that require more items and how many 2. Management verifies slots where new items will be added are empty 3. Management then sets what items need to be added and where 4. Management sends instructions to restocker 5. The restocker verifies they have received the instructions.
Post Condition:	**The restocker receives set of instructions of items to add to vending machine inventory**

Use Case Number:	**UC-11**
Use Case Name:	**See Purchase History**
Overview:	**This use case starts when management wants to see the purchase history of the vending machine**
Actor(s):	• **Management**
Pre condition(s):	**None**
Scenario Flow:	**Main (success) Flow:** 1. Management requests an up-to-date purchase history of the vending machine 2. Management receives the up-to-date purchase history of the vending machine
Post Condition:	**Management receives an up-to-date report on the purchase history of the vending machine.**
	Alternate Flows: 1. Management requests an up-to-date purchase history of the vending machine 2. The vending machine verifies there are is no current purchase history 3. Management receives a message stating the is no current purchase history

Post Condition:	**Management receives a message there is no current purchase history of the vending machine.**

Use Case Number:	**UC-12**
Use Case Name:	**See Machine Analytics**
Overview:	**This use case starts when management was to view which items are popular and which items are not selling well**
Actor(s):	**Management**
Pre condition(s):	**There must be current purchase history available for the vending machine**
Scenario Flow:	**Main (success) Flow:** 1. Management request a report on item popularity 2. Management receives a list compiled from purchase history that contains most popular to least popular item based on purchase history.
Post Condition:	**Management receives a list of items sorted from most popular to least popular.**

Design Document

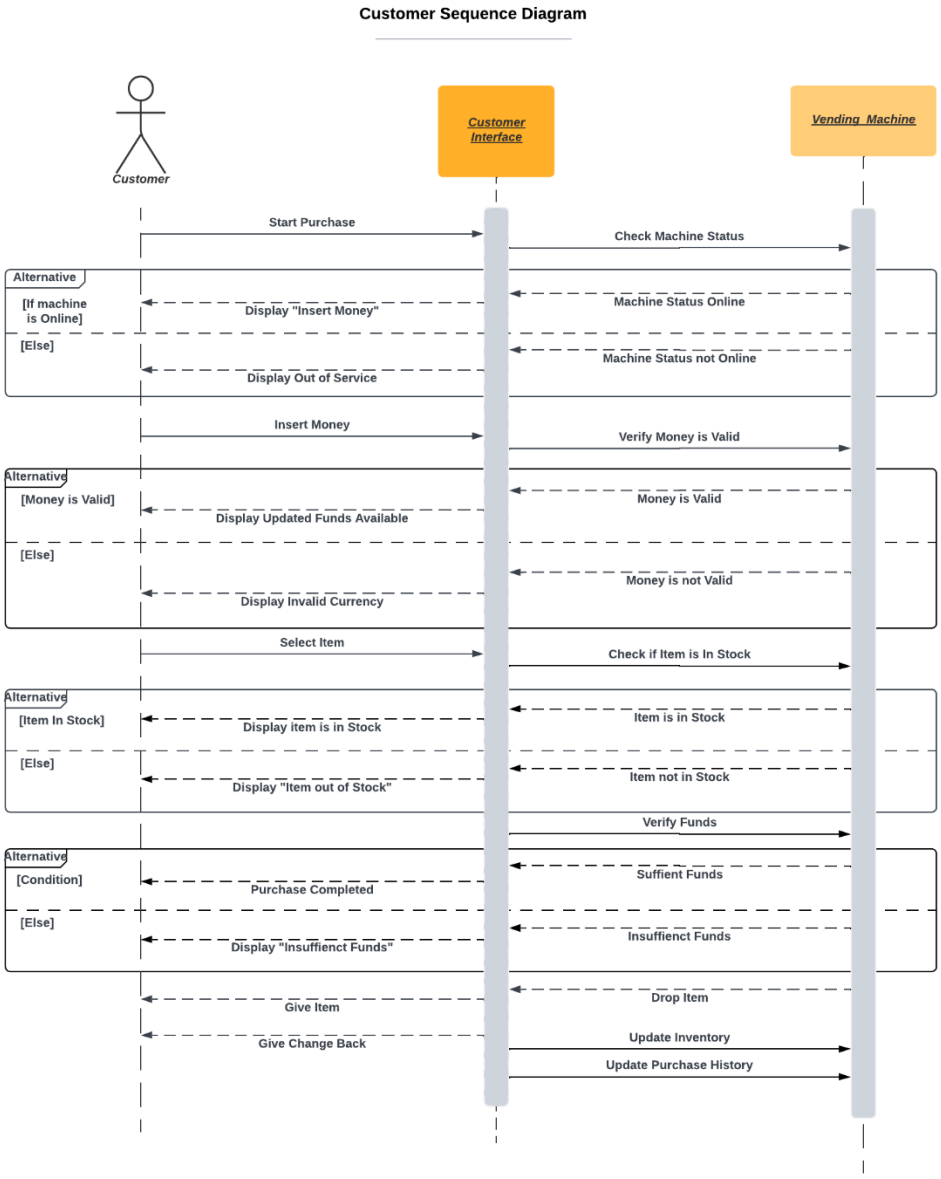
Components and Functions

No.	Component Name	Component Description
1	Vending Machine	Component State • This component is responsible for maintaining the information of the vending machines currently being managed • It will store information such as machine status, inventory, and purchase history Component Behavior • The Vending Machine component will provide the Customer and Management components information about a vending machines current inventory, and status. • It will also provide the Restocker component information about a vending machines status
2	Customer Interface	Component State • This component is responsible for receiving inputs from the customer as well as displaying information relevant to customer in order for them to purchase an item. Component Behavior • The customer interface will provide the Customer component which is responsible for processing the transaction the necessary information to do so. • This information includes the item chosen and the amount of money inserted into the vending machine
3	Customer	Component State • This component is responsible for processing the transaction made by the customer Component Behavior • The Customer component will provide the Customer Interface with information to display to the customer such as item name and item price as well as provide information on whether or not there are sufficient funds to buy that item and give change back when the purchase is complete.

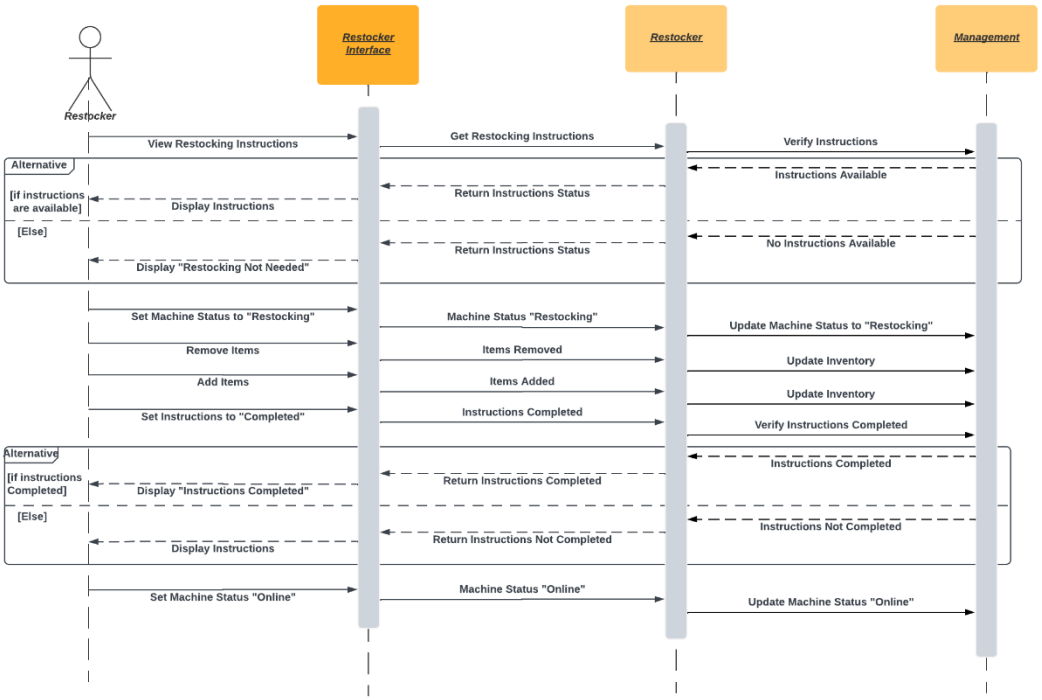
		• The Customer component after processing the purchase, will update the inventory of the vending machine as well as the purchase history which will allow the Management component to view the most current inventory and purchase history of that vending machine.
4	Restocker Interface	<u>Component State</u> • The Restocker Interface is responsible for displaying information relevant to the user: which vending machine needs to be service, what items to add, what items to remove. • It is also responsible for receiving inputs from the user <u>Component Behavior</u> • The Restocker Interface component will provide the Restocker component user inputs from the restocker
5	Restocker	<u>Component State</u> • The Restocker component is responsible for keeping track of service instructions set by the Management component for each vending machine • This component is also responsible for retrieving and setting a vending machine's status <u>Component Behavior</u> • The Restocker component will provide the Restocker Interface component the information requested by the user through the Restocker Interface • It will also set the status of a vending machine from "Online" to "Restocking' and vice versa when requested by the user. • It will also send Management a confirmation of when a vending machine is done being restocked and update its inventory.
6	Management Interface	<u>Component State</u> • The Management Interface is responsible for displaying information relevant to the user: vending machine status, current inventory, and purchase history. <u>Component Behavior</u> • The Management Interface component will provide the Management component user inputs from management.

7	Management	Component State • The Management component is responsible for maintaining the service instructions for each vending machine. • It is also responsible for maintaining information about the current inventory such as what items are expired and have been recalled as well as analytics. Component Behavior • The Management component will provide the Restocker component instructions on what vending machine needs to be serviced and which items to add and remove. • It will also provide the Management Interface component the information request by the user: machine status, current inventory, current purchase history, status of items, and analytics

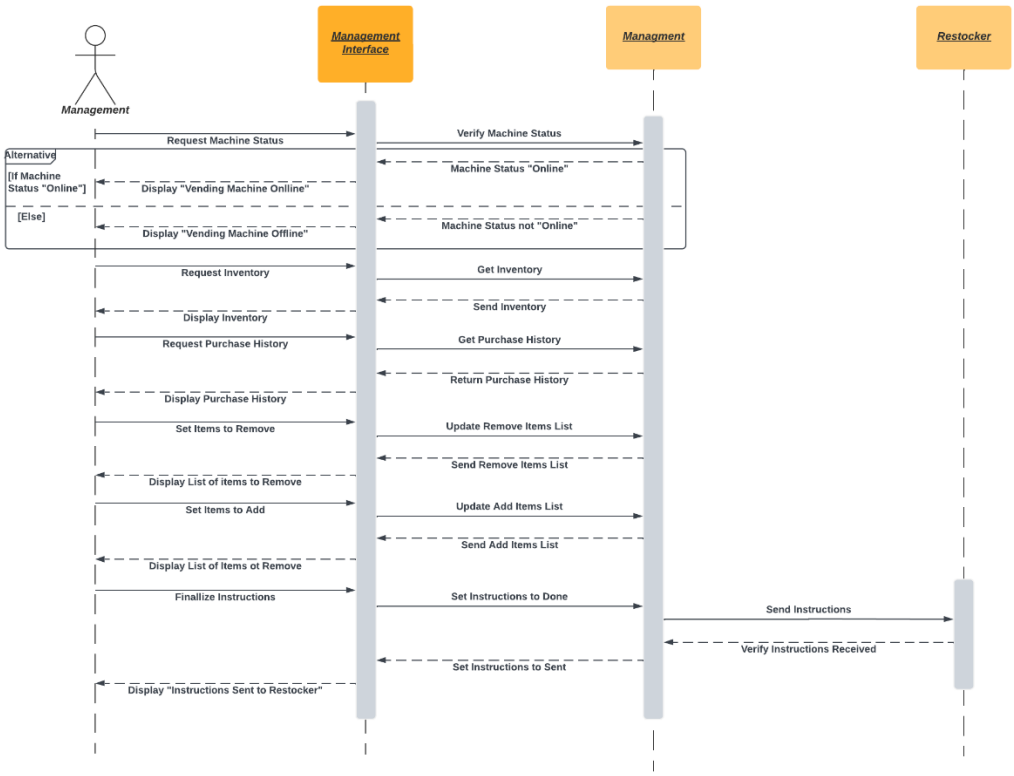
Sequence Diagrams



Restocker Sequence Diagram

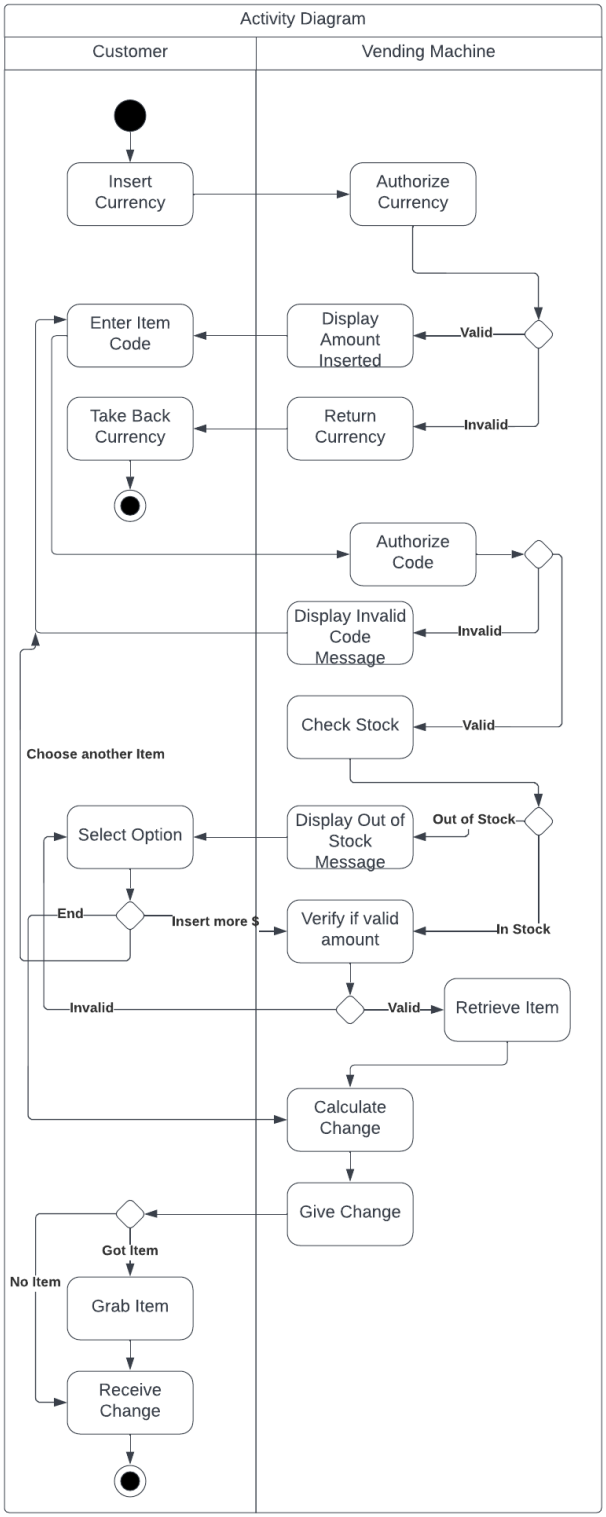


Management Sequence Diagram

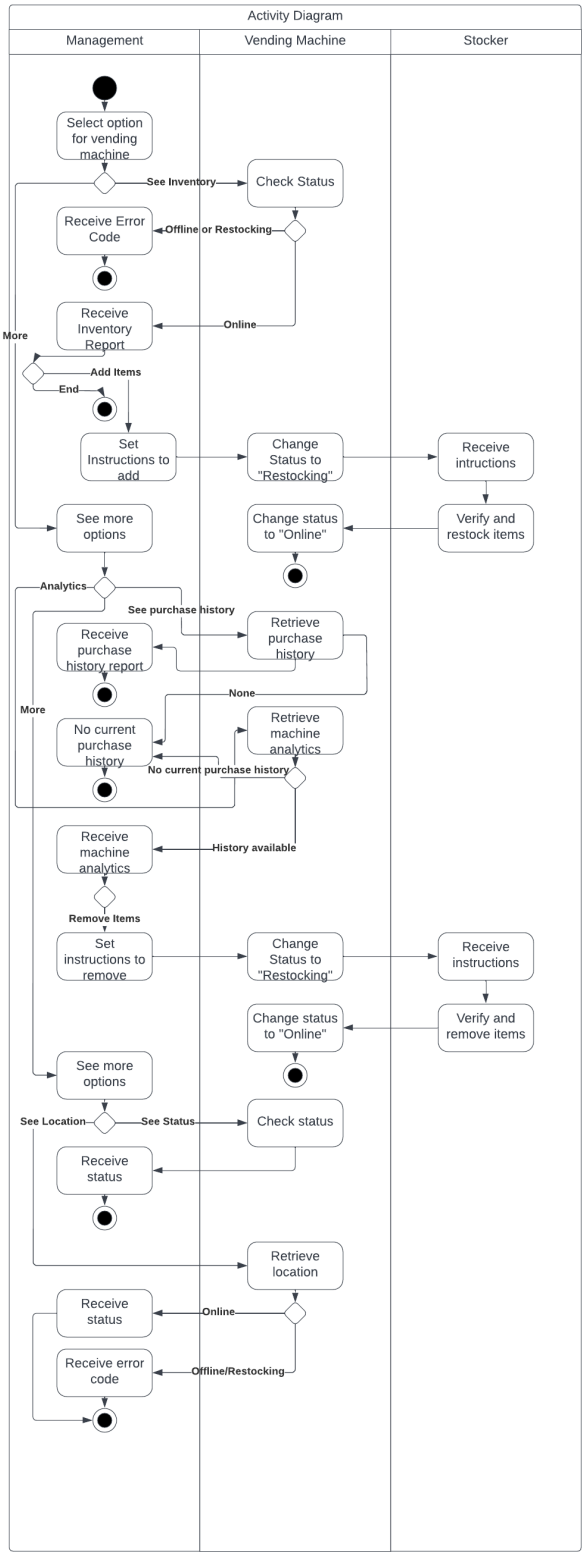


Activity Diagrams

- Customer Activity Diagram

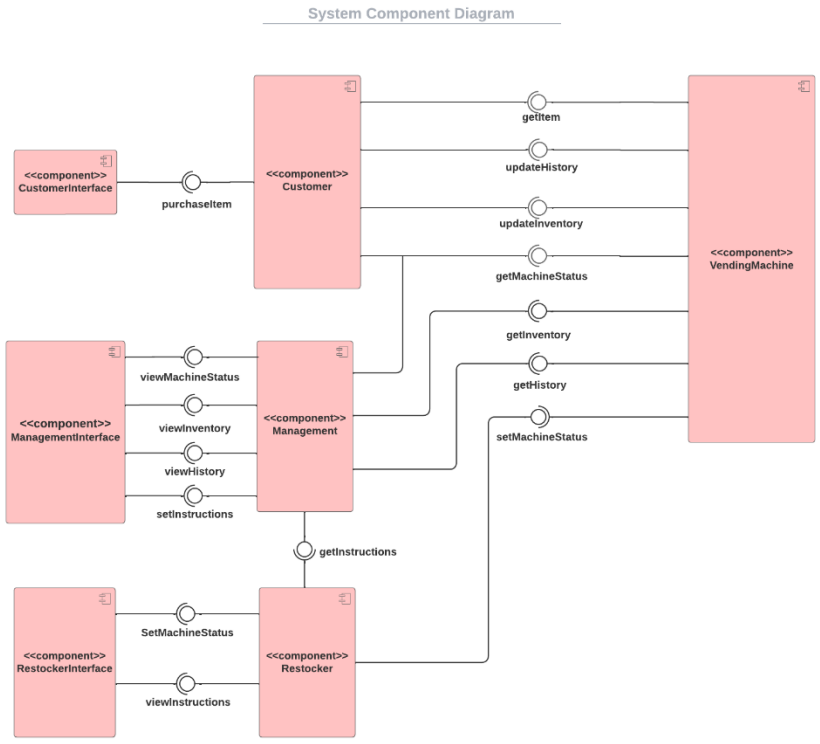


- Restocker and Management Activity Diagram



State chart Diagram

Component Diagram



Design Rationale

No.	Design Ideas

1	• We will implement three components: Customer, Restocker, and Management.
	• Customer
	  ○ Will process any transactions made by an individual using the vending machine
	• Restocker
	  ○ Will give access to the user through the app to view instructions for each vending machines
	  ○ Will allow user to set the status of machine from “Online” to “Restocking” to prevent the machine from being used
	• Management
	  ○ Will allow the user to view information on the vending machine such as inventory, purchase history and machine status
	  ○ Will allow the user to set instructions for the restockers on each vending machine

2	• Only accounted for the customer, restocker, and management component. When coding for the R1 features, we ran into a problem of which component should be in charge of the actual vending machine functions. • We decided to include a vending machine component that will be in charge of that

Development work

Test Cases for R1 Features

- Insert Money - Purchase Item

```java
@Test
public void testpaymentValidity1() {
    String money = "10.0";

    boolean valid = Customer.paymentValidity(money);

    assertEquals(true,valid);
}

@Test
public void testpaymentValidity2() {
    String money = "20.00";

    boolean valid = Customer.paymentValidity(money);

    assertEquals(false,valid);
}
```

- Select Item - Purchase Item

```java
@Test
public void testgetItem() throws FileNotFoundException {
    VendingMachine v = new VendingMachine();

    String error = "";

    Items item = Customer.getItem(v.inventory, "A1");

    if (!(item.itemLocation.equals("A1"))) {
        error = error + "Item location: Not equal, ";
    } else if (!(item.itemName.equals("Dr.Pepper"))) {
        error = error + "Item name: Not equal, ";
    } else if (item.itemPrice != 1.5) {
        error = error + "Item price: Not equal, ";
    } else if ((item.itemCount != 10)) {
        error = error + "Item count: Not equal, ";
    } else if (!(item.expDate.equals("10/23/2023"))) {
        error = error + "Item expDate: Not equal";
    }

    assertEquals("", error);

}
```

- Give Change Back

```
@Test
public void testgiveChangeBack1() {
    double change = 2.5;

    int coins[] = Customer.giveChangeBack(change);

    int expected[] = {10,0,0,0};

    assertArrayEquals(expected,coins);
}
```

- See Inventory

```
@Test
public void testGetInventory() throws FileNotFoundException {
    VendingMachine v = new VendingMachine();

    ArrayList<Items> expected = v.inventory;

    assertEquals(expected, Management.getInventory(v));
}
```

- See Machine Status

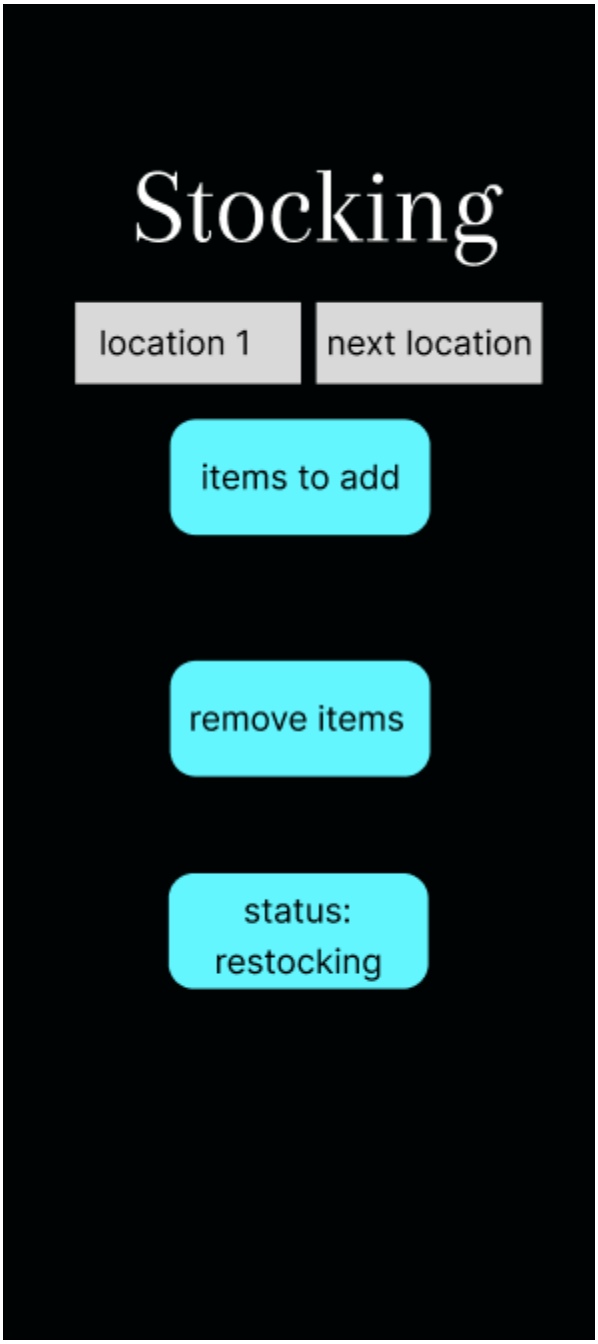
```
@Test
public void testGetMachineStatus() throws FileNotFoundException {
    VendingMachine v = new VendingMachine();

    String expected = v.machineStatus;

    assertEquals(expected, Management.getMachineStatus(v));
}
```

Mock Up of GUI for Customer Interface

Mock up GUI for Restocker Interface



Mock up GUI for Management Interface

Testing Plan and Tracker

Test No. ID	Related Systems / User Story	Pre-conditions	Test Description (steps)	Expected Outcome	R1 Outcome	R2 Outcome
1	Insert Money - Purchase Item	Vending Machine Status is "Online"	Customer inserts valid currency	Money should be accepted and funds should be updated	Passed	
2	Insert Money - Purchase Item	Vending Machine Status is "Online"	Customer inserts invalid currency	Money should not be accepted	Passed	
3	Selecting Item - Purchase Item	Vending Machine Status is "Online"	Customer inputs item location for "Dr. Pepper"	Item location given returns item information for "Dr. Pepper"	Passed	
4	Give Change Back	Vending Machine Status is "Online"	Pass in funds left after the transaction is complete	The change back and exact change are returned	Passed	
5	Set Machine Status	Machine status must not be "Offline"	Restocker sets the status of vending machine to "Restocking"	The status of the vending machine is set to "Restocking"	Not Implemented	
6	Remove Items	Must have received instructions from management	Restocker requests the list of items to remove from vending machine	List of items to remove is returned	Not Implemented	
7	Add Items	Must have received instructions from management	Restocker requests the list of items to add from vending machine	List of items to add and there location is returned	Not Implemented	

8	See Inventory	Vending Machine Status must "Online"	Management request to see the most current inventory of a vending machine	Returns the current inventory	Passed	
9	See Purchase History	Vending Machine Status "Online"	Management requests to see the most current purchase history	Returns the current purchase history	Not Implemented	
10	See Machine Location	Vending Machine Status is "Online"	Management requests location information of a vending machine	Returns machine location	Not Implemented	
11	See Machine Status	None	Management requests the status of a vending machine.	Returns machine status	Passed	
12	Set Items to Remove	Vending Machine status is "Online"	Management adds items that need to be removed from the vending machine to a list	Items that need to be removed are in the list	Not Implemented	
13	Set Items to Add	Vending Machine status is "Online"	Management adds items and where to add them in the vending machine to a list	Items that need to be added are in the list as well as their location	Not Implemented	
14	See Machine Analytics	Vending Machine status is "Online"	Management request to a list of items from most popular to least popular according to the purchase history	Returns a list of items from most popular to least popular	Not Implemented	

Release Roadmap for R1