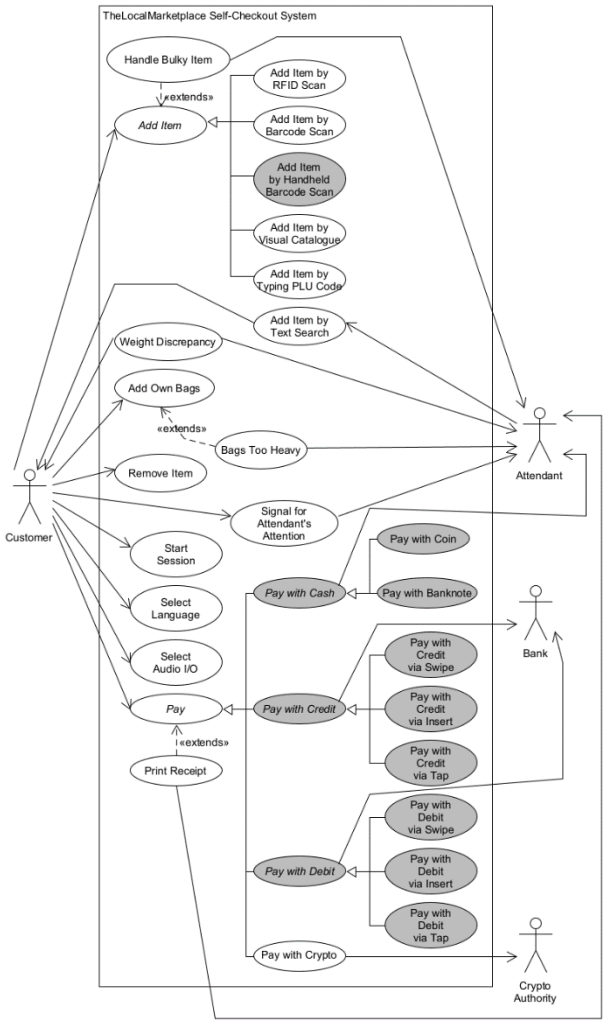
TheLocalMarketplace Self-Checkout System Use Cases (v2.0)



Some use cases have been altered or added as discussions with the client have proceeded and details have been refined. The diagram to the left shows the changed/added use cases with grey backgrounds.

Do Not Place Item in Bagging Area has been renamed *Handle Bulky Item* to improve clarity.

Use Case Descriptions

Use case:	*Add Own Bags*
Primary actor:	Customer.
Goal in context:	To allow the customer to add their own bags to the bagging area without causing a weight discrepancy.
Preconditions:	The system is ready to detect weight discrepancies.
Trigger:	The customer decides to make use of their own bags, activating an appropriate control on their station.
Scenario: 1. Customer: Signals the desire to add their own bags. 2. System: Indicates that the customer should add their own bags now. 3. Customer: Signals that the bags have been added. 4. System: Detects the weight change. 5. <**Bags Too Heavy extension point**> 6. System: Signals to the Customer that they may now continue.	
Exceptions: 1. The System is not ready to note weight discrepancies.	
Priority:	Low, can be skipped without major impact on system operation.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired connection.
Secondary actors:	None.
Channels to secondary actors: N/A	
Open issues: It is not clear why the System might not be ready to note weight discrepancies.	

Use case:	*Bags Too Heavy*
Primary actor:	Attendant.
Goal in context:	To detect the addition of items to the bagging area that are above the allowed threshold.
Preconditions:	The system is ready to detect weight discrepancies.
Trigger:	The system detects a weight change outside the allowable range.
Scenario: 1. System: Indicates that the weight in the bagging area differs from the allowable range. 2. System: Blocks further use of the customer station. 3. Attendant: Checks on the nature of the problem. 4. Attendant: Approves the weight discrepancy or corrects the problem. 5. System: Unblocks the customer station. 6. System: Signals to the Customer that they may now continue.	
Exceptions: 1. The System is not ready to note weight discrepancies.	
Priority:	Medium, can be skipped with some impact on system operation.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Hardwired connection.

Secondary actors:	None.
Channels to secondary actors: N/A	
Open issues: It is not clear why the System might not be ready to note weight discrepancies.	

Use case:	*Remove Item*
Primary actor:	Customer.
Goal in context:	To remove an item from the order, also allowing its removal from the bagging area without causing a weight discrepancy.
Preconditions:	The system is ready to detect weight discrepancies.
Trigger:	The customer wishes to eliminate an item from their order.
Scenario: 1. Customer: Signals that a specific item is to be removed from the order. 2. System: Blocks the self-checkout station from further customer actions. 3. System: Removes the item from the customer's order, reducing the expected weight in the bagging area. 4. System: Signals to the Customer that the order removal was successful and that the item should be removed from the bagging area or from their shopping cart. 5. System: Unblocks the self-checkout station.	
Exceptions: 1. The System is not ready to note weight discrepancies. 2. The item is not removed from the bagging area; see **Weight Discrepancy**.	
Priority:	Medium, necessary for good customer service and to avoid having to cancel transactions entirely.
When available:	Third iteration
Frequency of use:	A few times per day.
Channel to actor:	Graphical user interface; bagging area.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: If the item is not in the bagging area, how can the shopping market know that the item is not in the order? Should the item have to be returned to the Attendant, with them recording this fact?	

Use case:	*Add Item*
Primary actor:	Customer
Goal in context:	To allow the customer to add an item to their order for purchase.
Preconditions:	The system is ready to detect weight discrepancies and to take customer input.
Trigger:	The customer wishes to add an item to their order.
Scenario: 1. (Abstract use case) Details about the item to add must be provided to the System. 2. System: Blocks the self-checkout station from further customer interaction. 3. System: The expected weight in the bagging area is updated. 4. System: Signals to the Customer to place the item in the bagging area.	

5. System: Detects the weight change from the added item. 6. System: Unblocks the station.	
Exceptions: 1. The customer fails to place the item in the bagging area; see **Weight Discrepancy**.	
Priority:	High, the system cannot function without this.
When available:	First iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Customer station; bagging area.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: The manner in which the customer can indicate what is to be added is unclear. It seems reasonable that barcode scanning would be the default and so the system needs to be ready to scan each item unless an alternative means for providing the information is explicitly selected.	

Use case:	*Add Item by RFID Scan*
Primary actor:	Customer
Goal in context:	To permit the customer to scan an item via an RFID tag to add the item to their order.
Preconditions:	The system is ready to accept customer input.
Trigger:	The customer wishes to scan an item that possesses an RFID tag.
Scenario: 1. System: Detects an RFID tag. 2. System: Blocks the self-checkout station from further customer interaction. 3. System: Determines the characteristics (i.e., weight and cost) of the product associated with the barcode. 4. System: Updates the expected weight from the bagging area. 5. System: Signals to the Customer to place the scanned item in the bagging area. 6. Customer: Places the item in the bagging area. 7. System: Detects the weight change. 8. System: Unblocks the station.	
Exceptions: 1. The weight in the bagging area does not correspond to expectations; see **Weight Discrepancy**. 2. An item is scanned when a customer session is not in progress. The scanned information shall simply be ignored. 3. An item does not possess an RFID tag that is readable. The system will not be able to react to this item.	
Priority:	Medium, should be implemented for flexibility in handling inventory.
When available:	Second iteration.
Frequency of use:	Many times per day customer transaction.
Channel to actor:	Graphical user interface; RFID scanner.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues:	

None.

Use case:	*Add Item by Barcode Scan*
Primary actor:	Customer
Goal in context:	To permit the customer to scan a barcode of an item to add it to their order.
Preconditions:	The system is ready to accept customer input.
Trigger:	The customer wishes to scan a barcode on an item that possesses one.
Scenario: 1. System: Detects a barcode. 2. System: Blocks the self-checkout station from further customer interaction. 3. System: Determines the characteristics (weight and cost) of the product associated with the barcode. 4. System: Updates the expected weight from the bagging area. 5. System: Signals to the Customer to place the scanned item in the bagging area. 6. System: Signals to the System that the weight has changed. 7. System: Unblocks the station.	
Exceptions: 1. The weight in the bagging area does not correspond to expectations; see **Weight Discrepancy**. 2. An item is scanned when a customer session is not in progress. The scanned information shall simply be ignored.	
Priority:	High, must be implemented.
When available:	First iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Hardwired connection.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: None.	

Use case:	*Add Item by Handheld Barcode Scan*
Primary actor:	Customer
Goal in context:	To permit the customer to scan a barcode of an item to add it to their order by using the handheld scanner.
Preconditions:	The system is ready to accept customer input.
Trigger:	The customer wishes to scan a barcode on an item that possesses one.
Scenario: 1. System: Detects a barcode from the handheld scanner. 2. System: Blocks the self-checkout station from further customer interaction. 3. System: Determines the characteristics (weight and cost) of the product associated with the barcode. 4. System: Updates the expected weight from the bagging area. 5. System: Signals to the Customer to place the scanned item in the bagging area. 6. System: Signals to the System that the weight has changed. 7. System: Unblocks the station.	

Exceptions: 1. The weight in the bagging area does not correspond to expectations; see **Weight Discrepancy**. 2. An item is scanned when a customer session is not in progress. The scanned information shall simply be ignored.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Hardwired connection.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: None.	

Use case:	*Add Item by Text Search*
Primary actor:	Attendant.
Goal in context:	To permit the attendant to add an item to the customer's order for purchase, knowing details of the item that are not available to the customer (such as the name used by the organization to describe the item's product). This is important if a barcode or PLU code is absent or not recorded within the system.
Preconditions:	The system is ready for attendant input and a customer session is in progress at a connected self-checkout station.
Trigger:	The customer has asked the assistance of the attendant in adding an item to their order for purchase.
Scenario: 1. Attendant: The station is selected on which the relevant customer has a session. 2. Attendant: Searches for a product by specifying some text; this should be used as a keyword search in case a product's name is in a different order than expected by the Attendant. 3. System: Displays the results of the search. If the results are satisfactory for the attendant, proceed to 4, else return to 1. 4. Attendant: A product in the displayed results is selected for addition. 5. System: Blocks the self-checkout station from further customer interaction. 6. System: Adds the product to the customer's order; updates the expected weight. 7. System: Signals to the Customer that the item should be added to the bagging area. 8. System: Detects that the weight has changed. 9. System: Unblocks the station.	
Exceptions: 1. The weight in the bagging area does not correspond to expectations; see **Weight Discrepancy**.	
Priority:	Medium, should be implemented as the system will not function well without it.
When available:	Third iteration.
Frequency of use:	A few times per day.
Channel to actor:	Hardwired.
Secondary actors:	Customer.
Channels to secondary actors: Customer: Local area network; graphical user interface; bagging area.	
Open issues:	

Text search is impractical without a physical keyboard. Presumably the attendant I/O will have to possess a physical keyboard but not the individual self-checkout stations.

Use case:	*Add Item by Visual Catalogue*
Primary actor:	Customer.
Goal in context:	To allow the customer to look through a visual catalogue of products to select the one for which they wish to add an item for purchase.
Preconditions:	The system is ready for customer input.
Trigger:	The customer does not have a PLU code or barcode available for an item, but they want to add the item to their purchase.
Scenario: 1. System: Displays the visual catalogue, allowing the customer to browse through it. 2. Customer: Selects the product of interest. 3. System: Signals to the Customer to place the item in the bagging area. 4. System: Blocks the self-checkout system from further Customer interaction. 5. Customer: Places the item in the bagging area. 6. System: Detects that the weight has changed. 7. System: Unblocks the self-checkout system.	
Exceptions: 1. The weight in the bagging area does not correspond to expectations; see **Weight Discrepancy**.	
Priority:	High, must be implemented for the system to function-well.
When available:	Third iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; bagging area.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: The form of the visual catalogue and the manner in which one could browse it are vague. Should it be in alphabetical order, where all products starting with a given letter are displayed? Should it be done like in Netflix (for example), arranged roughly by category with a long list of images that can be scrolled through?	

Use case:	*Add Item by Typing PLU Code*
Primary actor:	Customer
Goal in context:	To permit the customer to type a known PLU code in order to add an item.
Preconditions:	The system is expecting a customer to enter a PLU code.
Trigger:	The customer has an item with a PLU code and they want to add this item to their order.
Scenario: 1. System: Displays a virtual numeric keyboard to the Customer. 2. Customer: Signals to the System that an item with a given PLU code is being added. 3. Customer: Types the PLU code. 4. System: Blocks the self-checkout station from further input. 5. System: Signals to the customer to add the item to the bagging area.	

6. System: Detects that the weight has changed. 7. System: Unblocks the station. 8. System: The customer station returns to its standard display.	
Exceptions: 1. The weight in the bagging area does not correspond to expectations; see **Weight Discrepancy**.	
Priority:	High, must be implemented.
When available:	Third iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; bagging area.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: 1. Simple mechanisms for self-correction by the customer, like a backspace key, should be supported. 2. The customer ought to have the opportunity to see that the PLU does not correspond to their item. At this point, communication with the attendant is likely the best option for correcting the problem.	

Use case:	*Handle Bulky Item*
Primary actor:	Customer.
Goal in context:	To permit the customer to avoid placing an item in the bagging area, presumably because it is too big or too heavy.
Preconditions:	The system is expecting the customer to place an item in the bagging area.
Trigger:	The customer signals to the system that they do not want to bag their item.
Scenario: 1. Customer: Signals that they do not want to bag their item. 2. System: Blocks the self-checkout station from further customer input. 3. System: Signals to the Attendant that a no-bagging request is in progress. 4. Attendant: Signals to the System that the request is approved. 5. System: Reduces the expected weight in the bagging area by the expected weight of the item. 6. System: Unblocks the station.	
Exceptions: 1. The customer adds the item to the bagging area anyways; see **Weight Discrepancy**.	
Priority:	Medium, should be implemented to avoid legal consequences of a customer becoming injured due to moving a heavy item or as a result of collision with a bulky item.
When available:	Second iteration.
Frequency of use:	A few times per week.
Channel to actor:	Graphical user interface.
Secondary actors:	Attendant.
Channels to secondary actors: Attendant: Local area network; attendant station.	
Open issues: 1. Care has to be taken to ensure that “blocking the station” does not inhibit the customer from making this request. 2. What happens if the attendant does not want to approve the request?	

Use case:	*Weight Discrepancy*
Primary actor:	None.
Goal in context:	To react to a difference in the actual weight of the bagging area and the expected weight, due to items being moved improperly into or out of the bagging area, or due to true weights differing from ideal weights or measured weights.
Preconditions:	The system is in the midst of a customer's session.
Trigger:	The System detects a weight change, and it deems that this weight is unacceptably different from expectations.

Scenario:

1. System: Blocks the self-checkout station from further customer interaction.
2. System: Signals to the Customer regarding the weight discrepancy.
3. System: Signals to the Attendant regarding the weight discrepancy.
   There are three options:
   a. Customer: Adds or removes an item in response; OR,
   b. Customer: Signals the System about a do-not-bag request (see **Do Not Place Item in bagging area**); OR,
   c. Attendant: Signals the System of a weight-discrepancy approval.
4. System: Unblocks the self-checkout station.

Exceptions:

1. If the attendant does not approve the weight discrepancy, they will need to manually investigate what has been placed in the bagging area and what is supposed to be there.
2. If the weight change from the bagging area still does not concur with expectations, the weight discrepancy will continue.

Priority:	High, must be implemented for the system to function.
When available:	Second iteration.
Frequency of use:	Potentially constant.
Channel to actor:	N/A.
Secondary actors:	Customer; Attendant.

Channels to secondary actors:

Customer: Graphical user interface; bagging area.

Attendant: Local area network; graphical user interface.

Open issues:

1. If the scale is untrustworthy, spurious weight discrepancies will occur, upsetting customers and attendants alike.
2. Scales cannot be trusted 100%, so it is important that attendants be attentive to the stations, and that they be provided with the means to correct problems.

Use case:	*Signal for Attendant's Attention*
Primary actor:	Customer.
Goal in context:	To allow the customer to ask for help from an attendant.
Preconditions:	The system is ready for customer input.
Trigger:	The customer seeks assistance from an attendant.

Scenario:

1. Customer: Signals to the System that they would like the attention of the Attendant.
2. System: Signals to the Attendant that their attention is desired at that Customer's station.

3. Attendant: Clears the request once they have attended to the Customer.	
Exceptions: Multiple requests could arrive concurrently.	
Priority:	Low, may be skipped without a major impact on system usability.
When available:	Third iteration.
Frequency of use:	Multiple times per day.
Channel to actor:	Graphical user interface.
Secondary actors:	Attendant.
Channels to secondary actors: Local area network; graphical user interface.	
Open issues: 1. What if the attendant and customer do not speak a common language?	

Use case:	*Start Session*
Primary actor:	Customer.
Goal in context:	To allow the customer to start a new session.
Preconditions:	The system not currently in a session.
Trigger:	The customer wishes to place an order.
Scenario: 1. System: Displays an initial splash screen, with an indication of "Touch Anywhere to Start" or similar message. 2. Customer: Touches the screen. 3. System: Ready for further customer interaction.	
Exceptions: None.	
Priority:	High, essential to system operation.
When available:	First iteration.
Frequency of use:	Every customer.
Channel to actor:	Graphical user interface.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: 1. How should this interact with language support and support for accessibility?	

Use case:	*Select Language*
Primary actor:	Customer.
Goal in context:	To allow the customer to interact with the system in a language of their choice.
Preconditions:	The system is ready for customer input.
Trigger:	The customer wishes to interact with the system in a particular language.
Scenario: 1. System: Displays the language options available at the station. 2. Customer: Signals to the System the selected language. 3. System: Ready for further customer interaction.	

Exceptions: 1. If a desired language is not present, or if the customer changes their mind, the option to cancel should be available.	
Priority:	Low, may be skipped without a major impact on system usability.
When available:	Third iteration.
Frequency of use:	Infrequent; a few times per week.
Channel to actor:	Graphical user interface.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: 1. How many languages must be supported? 2. Is the standard language to be English unless something else is selected? Or should this be a configuration setting when the system is installed? 3. How will this interact with the customer if they are vision-impaired? 4. The cost of maintaining the system and its attendant databases in additional languages could lead to unacceptably high costs. Poor translation could lead to customer misunderstanding or offense.	

Use case:	*Select Audio I/O*
Primary actor:	Customer.
Goal in context:	To allow the customer to interact with the system in an auditory manner, speaking into a microphone and hearing from a speaker.
Preconditions:	The system is ready for customer input.
Trigger:	The customer desires to interact with the system verbally.
Scenario: 1. Customer: Signals to the System that verbal interaction is to be used.	
Exceptions: None.	
Priority:	Low, as most customers can be expected to prefer visual interaction. However, failure to provide this functionality could be legally challenged as discriminatory.
When available:	Third iteration.
Frequency of use:	Once a week.
Channel to actor:	Hardwired.
Secondary actors:	None.
Channels to secondary actors: N/A.	
Open issues: 1. Interacting with all the functions of the system verbally could be challenging, requiring significantly different user interface design. 2. Multiple stations using verbal interaction could lead to confusion for the system and for customers, especially if volume levels need to be elevated for some customers. 3. The costs of this additional development could be significant compared to the costs of the base functionality. 4. Presumably the customer should be able to change their mind and also switch back to visual/touch based interaction.	

5. Should attendants also be able to interact verbally with their station? Failure to support this could be deemed as discriminatory hiring practices.

Use case:	*Pay*
Primary actor:	Customer.
Goal in context:	To permit the customer to pay for their items.
Preconditions:	There are items on the customer's order for which sufficient payment has yet to be made.
Trigger:	The customer wishes to pay for their items.
Scenario: 1. (Abstract use case) The customer indicates the mode of payment that they want to use. 2. (Abstract use case) The customer provides payment, signalling this to the System. 3. System: The amount due is reduced by the payment. 4. System: Signals to the Customer the remaining amount due.	
Exceptions: 1. If the payment is not made or not accepted, the remaining amount due is not reduced, but the customer session is otherwise unaffected.	
Priority:	High, must be implemented.
When available:	First iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Graphical user interface; payment-based user interfaces according to the payment mode.
Secondary actors:	Unknown.
Channels to secondary actors: N/A.	
Open issues: 1. Should the customer be forced to make full payment, or can they make partial payment and then return to adding/removing items?	

Use case:	*Pay with Cash*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide coins and/or banknotes as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay cash for their order.
Scenario: 1. (Abstract use case) Customer: Inserts coins and/or banknotes in the System. 2. System: Reduces the remaining amount due by the value of the inserted cash. 3. System: Signals to the Customer the updated amount due after the insertion of each coin or banknote. 4. System: If the remaining amount due is greater than 0, go to 1. 5. System: If the remaining amount due is less than 0, dispense the amount of change due. 6. Once payment in full is made and change returned to the customer, see **Print Receipt**.	
Exceptions: 1. If the Customer inserts cash that is deemed unacceptable, this will be returned to the customer without involving the System, presumably handled in hardware.	

2. If insufficient change is available, the Attendant should be signalled as to the change still due to the Customer and the station should be suspended so that maintenance can be conducted on it.	
Priority:	Medium, essential in some business contexts, undesired in others.
When available:	First iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; cash-handling interfaces.
Secondary actors:	Attendant.
Channels to secondary actors: Local-area network; graphical user interface.	
Open issues: 1. The hardware has to handle invalid cash, to reject it without involving the control software. 2. Should mixed modes of payment be supported (e.g., part cash, part crypto)?	

Use case:	*Pay with Coin*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a coin as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay with a coin for their order.
Scenario: 1. Customer: Inserts a coin in the System. 2. System: Reduces the remaining amount due by the value of the inserted coin. 3. System: Signals to the Customer the updated amount due after the insertion of the coin. 4. System: If the remaining amount due is greater than 0, go to 1. 5. System: If the remaining amount due is less than 0, dispense the amount of change due. 6. Once payment in full is made and change returned to the customer, see **Print Receipt**.	
Exceptions: 1. If the Customer inserts a coin that is deemed unacceptable, this will be returned to the customer without involving the System, presumably handled in hardware. 2. If insufficient change is available, the Attendant should be signalled as to the change still due to the Customer and the station should be suspended so that maintenance can be conducted on it.	
Priority:	Medium, essential in some business contexts, undesired in others.
When available:	First iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; cash-handling interfaces.
Secondary actors:	Attendant.
Channels to secondary actors: Local-area network; graphical user interface.	
Open issues: 1. The hardware has to handle invalid cash, to reject it without involving the control software. 2. Should mixed modes of payment be supported (e.g., part cash, part crypto)? 3. This use case seems to imply that the customer has to signal to the system for every coin entry. Won't this be really monotonous?	

Use case:	*Pay with Banknote*
Primary actor:	Customer.

Goal in context:	To permit the customer to provide a banknote as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay with a banknote for their order.

Scenario:

1. Customer: Inserts a banknote in the System.
2. System: Reduces the remaining amount due by the value of the inserted banknote.
3. System: Signals to the Customer the updated amount due after the insertion of the banknote.
4. System: If the remaining amount due is greater than 0, go to 1.
5. System: If the remaining amount due is less than 0, dispense the amount of change due.
6. Once payment in full is made and change returned to the customer, see **Print Receipt**.

Exceptions:

1. If the Customer inserts a banknote that is deemed unacceptable, this will be returned to the customer without involving the System, presumably handled in hardware.
2. If insufficient change is available, the Attendant should be signalled as to the change still due to the Customer and the station should be suspended so that maintenance can be conducted on it.

Priority:	Medium, essential in some business contexts, undesired in others.
When available:	First iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; cash-handling interfaces.
Secondary actors:	Attendant.

Channels to secondary actors:
Local-area network; graphical user interface.

Open issues:

1. The hardware has to handle invalid cash, to reject it without involving the control software.
2. Should mixed modes of payment be supported (e.g., part cash, part crypto)?
3. This use case seems to imply that the customer has to signal to the system for every banknote entry. Won't this be really monotonous?

Use case:	*Pay with Credit*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a credit card as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via credit card for their order.

Scenario:

1. (Abstract use case) Customer: Presents the credit card to the System.
2. (Abstract use case) Customer: Provides validation as to their identity.
3. (Abstract use case) System: Signals to the Bank the details of the credit card, validation, and the amount to be charged.
4. Bank: Signals to the System the hold number against the account of the credit card.
5. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of credit available.
6. Bank: Signals to the System that the transaction was successful.
7. System: Updates the amount due displayed to the customer.
8. <**Print Receipt extension point**>.

Exceptions:

1. If the Bank does not approve the transaction, the remaining amount due will not change.

Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: Does the Customer get to try their validation more than once? Will the card be block if not validated correctly?	

Use case:	*Pay with Credit via Swipe*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a credit card as payment, swiping it.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via credit card for their order.
Scenario: 1. Customer: Swipes the credit card to the System. 2. System: Prompts the Customer for a signature. 3. Customer: Provides a signature. 4. System: Signals to the Bank the details of the credit card, signature, and the amount to be charged. 5. Bank: Signals to the System the hold number against the account of the credit card. 6. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of credit available. 7. Bank: Signals to the System that the transaction was successful. 8. System: Updates the amount due displayed to the customer. 9. <**Print Receipt extension point**>.	
Exceptions: 1. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: Does the Bank compare the signature against the one on record? If they don't match, does the Customer get to try again?	

Use case:	*Pay with Credit via Insert*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a credit card as payment, swiping the card.
Preconditions:	The system has to be ready to take payment.

Trigger:	The customer has indicated that they want to pay via credit card for their order.
Scenario: 1. Customer: Inserts the credit card and provides the PIN. 2. System: Validates the PIN against the credit card. 3. System: Signals to the Bank the details of the credit card and the amount to be charged. 4. Bank: Signals to the System the hold number against the account of the credit card. 5. System: Requires a signature from the Customer. 6. Customer: Provides a signature. 7. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of credit available. 8. Bank: Signals to the System that the transaction was successful. 9. System: Updates the amount due displayed to the customer. 10.<**Print Receipt extension point**>.	
Exceptions: 1. If the customer enters the wrong PIN three times in a row, the card will be blocked. 2. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: How does the System validate the PIN?	

Use case:	*Pay with Credit via Tap*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a credit card as payment, using tap.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via credit card for their order.
Scenario: 1. Customer: Taps the credit card. 2. System: Signals to the Bank the details of the credit card and the amount to be charged. 3. Bank: Signals to the System the hold number against the account of the credit card. 4. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of credit available. 5. Bank: Signals to the System that the transaction was successful. 6. System: Updates the amount due displayed to the customer. 7. <**Print Receipt extension point**>.	
Exceptions: 1. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.

Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: There seems to be no validation mechanism. How can we be sure that the tap was intentional and that the credit card belongs to this Customer?	

Use case:	*Pay with Debit*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a debit card as payment.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via debit card for their order.
Scenario: 1. (Abstract use case) Customer: Presents the debit card to the System. 2. (Abstract use case) Customer: Provides validation as to their identity. 3. (Abstract use case) System: Signals to the Bank the details of the debit card, validation, and the amount to be charged. 4. Bank: Signals to the System the hold number against the account of the debit card. 5. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of money available. 6. Bank: Signals to the System that the transaction was successful. 7. System: Updates the amount due displayed to the customer. 8. <**Print Receipt extension point**>.	
Exceptions: 1. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: Does the Customer get to try their validation more than once? Will the card be block if not validated correctly?	

Use case:	*Pay with Debit via Swipe*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a debit card as payment, swiping it.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via debit card for their order.
Scenario: 1. Customer: Swipes the debit card to the System. 2. System: Prompts the Customer for a signature. 3. Customer: Provides a signature.	

4. System: Signals to the Bank the details of the debit card, signature, and the amount to be charged. 5. Bank: Signals to the System the hold number against the account of the debit card. 6. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of money available. 7. Bank: Signals to the System that the transaction was successful. 8. System: Updates the amount due displayed to the customer. 9. <**Print Receipt extension point**>.	
Exceptions: 1. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: Does the Bank compare the signature against the one on record? If they don't match, does the Customer get to try again?	

Use case:	*Pay with Debit via Insert*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a debit card as payment, swiping the card.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via debit card for their order.
Scenario: 1. Customer: Inserts the debit card and provides the PIN. 2. System: Validates the PIN against the debit card. 3. System: Signals to the Bank the details of the debit card and the amount to be charged. 4. Bank: Signals to the System the hold number against the account of the debit card. 5. System: Requires a signature from the Customer. 6. Customer: Provides a signature. 7. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of money available. 8. Bank: Signals to the System that the transaction was successful. 9. System: Updates the amount due displayed to the customer. 10.<**Print Receipt extension point**>.	
Exceptions: 1. If the customer enters the wrong PIN three times in a row, the card will be blocked. 2. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors:	

Bank: Internet via secure protocol.
Open issues: How does the System validate the PIN?

Use case:	*Pay with Debit via Tap*
Primary actor:	Customer.
Goal in context:	To permit the customer to provide a debit card as payment, using tap.
Preconditions:	The system has to be ready to take payment.
Trigger:	The customer has indicated that they want to pay via debit card for their order.
Scenario: 1. Customer: Taps the debit card. 2. System: Signals to the Bank the details of the debit card and the amount to be charged. 3. Bank: Signals to the System the hold number against the account of the debit card. 4. System: Signals to the Bank that the transaction identified with the hold number should be posted, reducing the amount of money available. 5. Bank: Signals to the System that the transaction was successful. 6. System: Updates the amount due displayed to the customer. 7. <**Print Receipt extension point**>.	
Exceptions: 1. If the Bank does not approve the transaction, the remaining amount due will not change.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Many times per day.
Channel to actor:	Graphical user interface; card-handling interface.
Secondary actors:	Bank.
Channels to secondary actors: Bank: Internet via secure protocol.	
Open issues: There seems to be no validation mechanism. How can we be sure that the tap was intentional and that the credit card belongs to this Customer?	

Use case:	*Pay with Crypto*
Primary actor:	Customer
Goal in context:	To allow the customer to pay for their order with cryptocurrency.
Preconditions:	The system is ready to receive customer payment.
Trigger:	The customer wishes to pay with cryptocurrency.
Scenario: 1. Customer: Signals to System that a cryptocurrency payment is to be made. 2. System: Communicates with Crypto Authority regarding the payment. 3. Crypto Authority: Authorizes the payment? 4. System: Reduces the remaining amount due by the amount authorized by the Crypto Authority? 5. System: Signals to the Customer the remaining amount due.	
Exceptions: 1. If communication with the Crypto Authority is not established securely, the transaction should be cancelled and the remaining amount due will not be changed.	

Priority:	Low, may be skipped since the number of customers wishing to use this is likely to be few.
When available:	Third iteration.
Frequency of use:	Infrequent, a few times per week.
Channel to actor:	Graphical user interface?
Secondary actors:	Crypto Authority.
Channels to secondary actors: Secure internet connection.	
Open issues: 1. What kinds of cryptocurrency are to be supported? 2. What kind of hardware is needed to permit the customer to provide details of their cryptocurrency? 3. How is conversion between crypto and local currency to be supported? 4. What protocols are to be followed? 5. How is security to be handled?	

Use case:	*Print Receipt*
Primary actor:	Customer.
Goal in context:	To provide a receipt for the purchases and payments made by the customer.
Preconditions:	Payment in full has been received for the customer's order.
Trigger:	Payment in full has been received for the customer's order.
Scenario: 1. System: The payment record will be up-to-date with details of the payment(s). 2. System: Signals to the receipt printer to print the payment record. 3. System: Thanks the Customer. 4. System: Ends the session, ready for a new one.	
Exceptions: 1. If the receipt printer runs out of paper or ink in the middle of printing the receipt, the printing will be aborted, the station will be suspended, and the Attendant informed that a duplicate receipt must be printed and that the station needs maintenance.	
Priority:	High, must be implemented.
When available:	Second iteration.
Frequency of use:	Every customer transaction.
Channel to actor:	Receipt printer.
Secondary actors:	Attendant.
Channels to secondary actors: Local area network; graphical user interface.	
Open issues: 1. Should electronic receipts also be supported? 2. If a problem occurs after payment is made, how can a duplicate receipt be created for the customer? 3. Can the customer decline printing a receipt?	