|  |  |
| --- | --- |
| TextUI | calls |
| TextUI | None |
| MainMenu | None |
| RouteChoice | Store->DisplayInventory(), Store->Items(), ItemMenu(), Store->AddItemToCart(), Store->DisplayCart, Store->DisplayInventory, Store->CartItems(), Store->RemoveItemFromCart, Store->Checkout(), Store->ClearCart() |
| ItemMenu | None |

|  |  |
| --- | --- |
| ShoppingCart | calls |
| ShoppingCart | None |
| AddItem | Item->get\_id() , Item->IncreaseQuantity() |
| RemoveItem | Item->get\_quantity(), Item->DecreaseQuantity() |
| DisplayCart | Item->ToString() |
| ClearCart | None |
| get\_items() | None |

|  |  |
| --- | --- |
| Item | calls |
| Item | None |
| get\_id() | None |
| get\_quantity() | None |
| get\_cost() | None |
| get\_type() | None |
| IncreaseQuantity | None |
| DecreaseQuantity | None |
| ToString | None |
| Clone | Item() |
| operator<< | ToString() |

|  |  |
| --- | --- |
| Store | Calls |
| Store | Item(), ShoppingCart() |
| DisplayInventory | Item->ToString() |
| Items | Item->ToString(), Item-> get\_quantity() |
| CartItems | ShoppingCart->get\_items(), Item-> get\_id(), Item->ToString(), Item->get\_quantity() |
| AddItemToCart | Item->get\_Id(), Item->DecreaseQuantity()  Item->Clone(), ShoppingCart->AddItem() |
| RemoveItemFromCart | ShoppingCart->get\_items(), Item->get\_id(), ShoppingCart->RemoveItem(), Item->IncreaseQuantity() |
| DisplayCart | ShoppingCart->DisplayCart() |
| Checkout | ShoppingCart->get\_items(), Item->get\_cost(), Item->get\_quantity(), Item->get\_type(), ClearCart() |
| ClearCart | ShoppingCart->ClearCart() |

Part 2:

The clone method creates a new item and copies all the current item’s properties to the new item (except sets qty to 1) and returns a pointer to the new object. It does this for the purpose of removing one item from inventory so that it can be added to the shopping cart. It is included in the class definition for item because that is where you have access to all of the item’s properties.

Part 3:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Trait | TextUI | ShoppingCart | Item | Store |
| Cohesive | Yes, this is cohesive as it allows the user to create a ui while hiding the hard parts. | Yes, this allows for a single abstraction for the class | Yes, this allows for a single abstraction for the class, and all functionality for item is contained. | Yes, this allows for a single abstraction for the class, and allows all functionality for store and other classes easily. |
| Complete | Yes, this class provides all necessary functionality to create a ui. | Yes, contains all functionality necessary for the object | Yes, contains all functionality necessary for the object including cloning the object for moving to cart. | No, store does not currently handle transactions or amounts of money. |
| Clear | Yes, this class is clear in what it does and what each member function does | Yes, functions are named clearly and do the expected operation | Yes, functions are named clearly and do the expected operation | Yes, functions are named clearly and do the expected operation |
| Convienient | Yes, it abstracts away a lot of the details making it convenient to implement. | Yes, allows access to every operation needed for object, in a easy to work with way. | Yes, allows access to every operation needed for object, in a easy to work with way. | Yes, allows access to every operation needed for object, in a easy to work with way. |
| Consistent | Yes, the naming is consistent, and the behavior does what is expected. | Yes, naming follows guidelines and is clear for what is happening. | Yes, naming follows guidelines and is clear for what is happening. | Yes, naming follows guidelines and is clear for what is happening. |