Too Many Eyes / Shop System

System Design Document

# Changes

## V 1.0

Editor: Perrin Peterson

02/24/2022

* Document Created, and initial version

# Introduction

This document details the design and purpose of a locomotion system designed for Too Many Eyes. Designed for the Blight Brew Game, the document will cover;

* Usage for the designers.
* UML for the Programmers.

# Design Goals

The goal of the Shop System is to serve as an interface for the physical shop, and give the shopping AI a way to find, collect, and pay for items. This system works with the Economy System to make a more fluid economy, and make prices matter.

# Behaviour

The system will allow the player to interface with it through a shop book (not yet implemented), and the shop book will display information about the shop, such as history, purchasable upgrades, purchasable items, and a UI to price items currently in stock. The shop will then communicate with the economy system, which handles the value of the economy and the value of items, and passes it to shoppers, which make a decision whether or not to buy the item. After this, the player can “barter” to try and get a bit more money out of the AI. This will serve as the main way to gather money in the game.

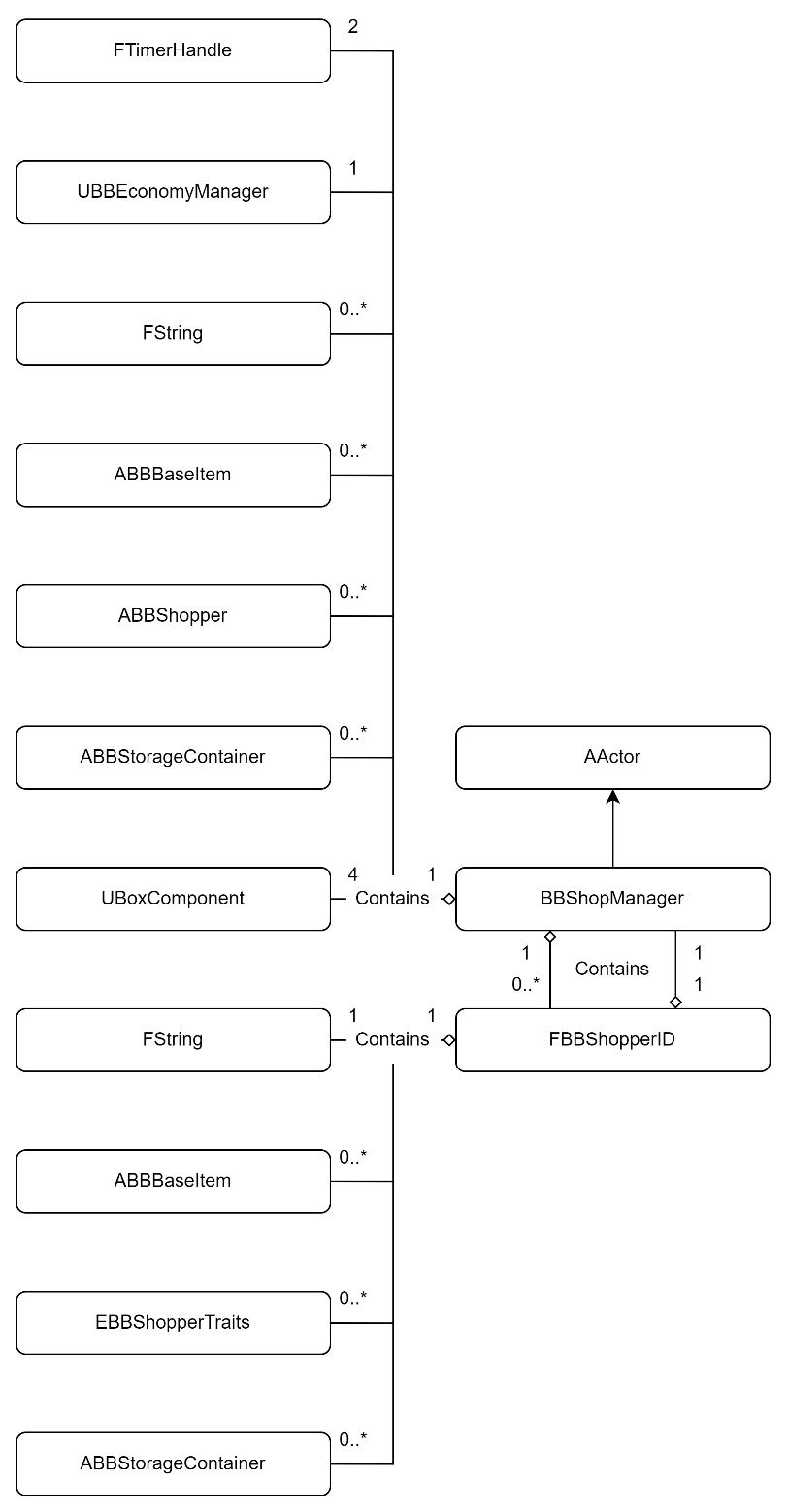
# High Level Design

Graphical user interface

Description automatically generated

* Player system - the system is our core system and is what the player interacts with to use all our other systems.
* Locomotion system - an advanced movement system we’re using to give us more control over the players movement. This gives us variables we can control, such as stamina, and additional functionality, such as climbing and vaulting.
* Potion system – a system meant to be easy to use for designers. The system allows for the designers as much freedom as possible, while giving the coders a minimal amount of updating to do. This is the main system the player will use to complete puzzles and generate income.
* Inventory system – a system meant to be easy to use for designers. The system allows for the designers as much freedom as possible, while giving the coders a minimal amount of updating to do. The system works as a container to hold items that the player collects. This includes the players backpack, as well as storage containers around the world.
* Resource system – a system meant to be the first of the steps for the player to generate income. This system gives the player ways to harvest materials and shows off behavioural logic for when the system is used.
* Crafting System – the system in between the Resource system, and the Shop/Potion systems, in terms of actual gameplay. This systems job is to refine the resources the player gathers into other items or potions for use in one of the other systems.
* Shop system - a system to generate income, and allow for the player to improve. The system interfaces with AI, the player, and Storage objects, to give the player a way of selling items to NPC’s. This can have numerous effects on the AI and is the primary source of income, a necessary resource for improving the players arsenal, and serves to break the monotony of just grinding for resources.
* AI System - The system dealing with the NPC’s in the world, including fauna. The system interfaces with the shop system, allowing for NPC’s to be customers, as well as gives the AI their logic for movement, interaction, and anything else.
* Economy System – A system that goes unseen by the player, changes the value of items based on selling history, value of the local economy, and a degree of randomness. This should make the world feel a little bit more alive, and give the player a reason to continually adjust prices based on customer responses.
* User Interface System – A system used to show, hide, and allow the player to control the Interface. The interface controls almost every system in some way, but mainly through the player.

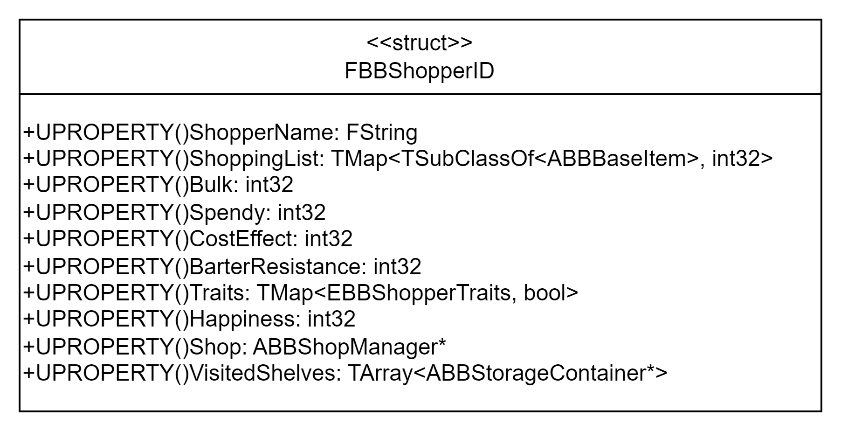
# Mid Level View



*For more detailed information about ABBBaseItem, or ABBStorageContainer, please see the Inventory System MDD.   
For more detailed information about UBBEconomyManager, please see the Economy System MDD.  
For more detailed information about ABBShopper, please see the AI System MDD.*

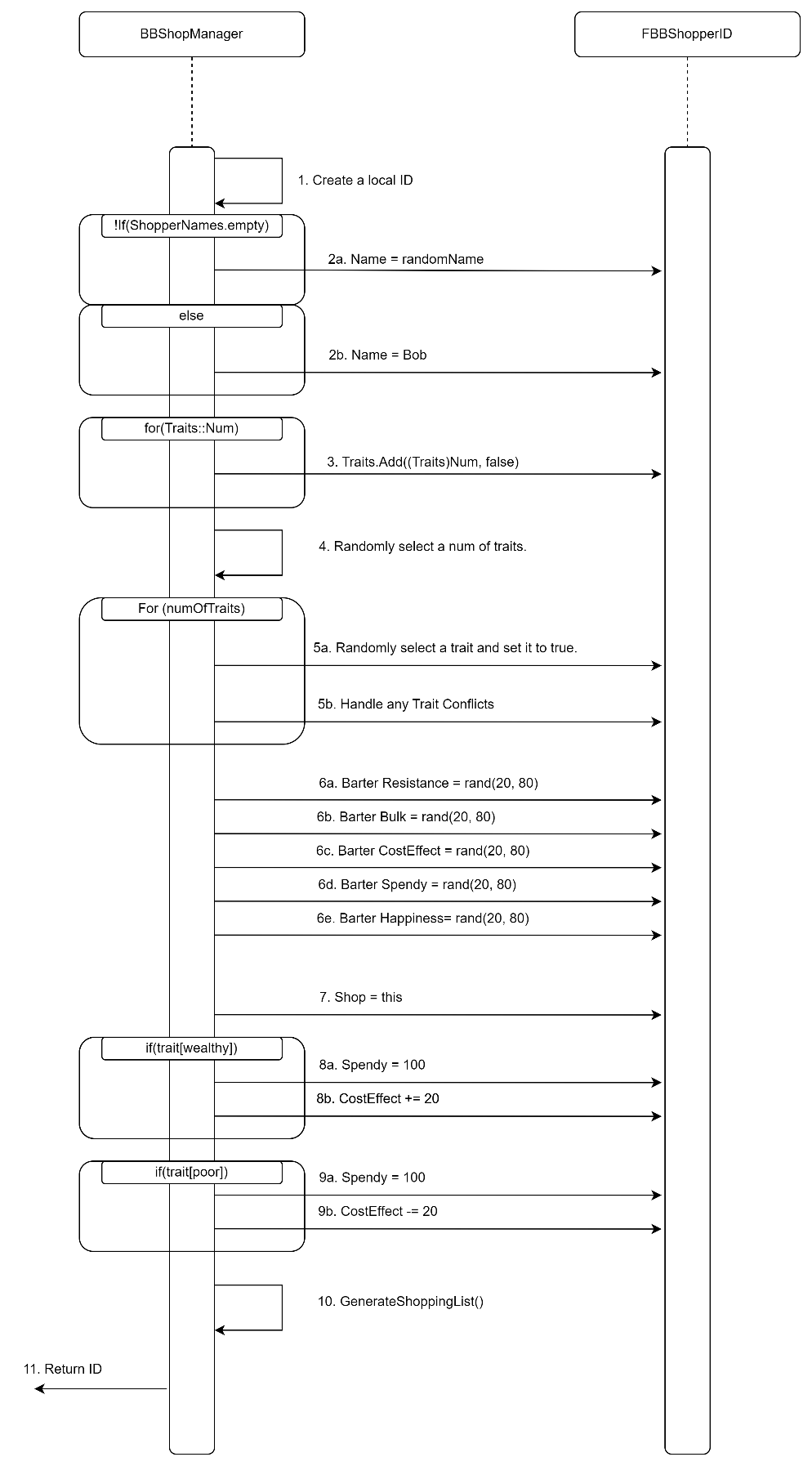
* BBShopManager is a placeable area, which allows for custom AI’s to interact with objects in that area. It also allows us to place areas that the AI needs to know about. It also serves as logic mid point between the ShopperAI, the Player, and the Economy System.
* FBBShopperID is a container that holds everything that makes each AI unique, making the AI’s more personable.
* EBBShopperTraits is an Enum that can be assigned and checked against for the Shoppers to have specific traits. In this situation, there is a map of the traits, and they’re assigned either true, or false.

# Logical View



# Process View

## Generating a Random Shopper ID

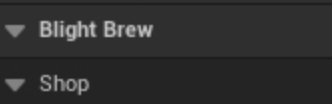
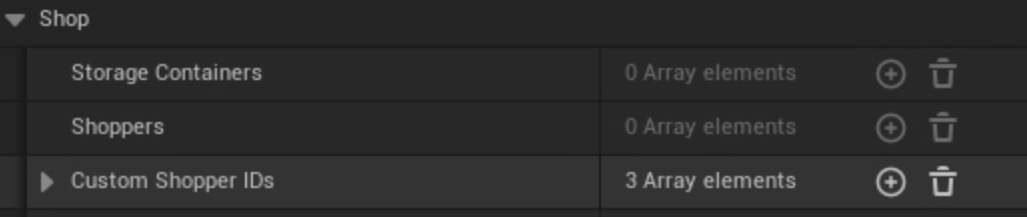


1. Create a local ID that we can modify.
2. Set a random name.
   1. If the array has options, use a random one from the array.
   2. If the array is empty, use “Bob”.
3. Initialize the traits list, setting them all to false.
4. Get a random number for the number of traits.
5. For the number of traits;
   1. Set a random trait to true.
   2. If there’s a trait conflict, set the conflicting trait to false.
6. Set all the other values to a random value between 20 and 80.
7. Set the shop to itself.
8. If the Wealthy trait is true, adjust the value.
9. If the Poor trait is true, adjust the value.
10. Call the GenerateShoppinglist(), passing in the list.
11. Return the result.

# Use Case View

## ShopManager

### Set a custom ShopperID

1. Select the ShopManager in the world.
2. In the details, open the Shop menu.  
   
3. Open the Custom Shoppers dropdown menu.  
   
4. Click the plus button to add a new ID.
5. Set the parameters and traits that you want.  
   