

# *Idle Farmer*

## **Table of Contents**

[1. ONLINE HELP](#)

[2. ABOUT](#)

[3. DO YOU LIKE IT?](#)

[4. FEATURES](#)

[5. TECH SPECS](#)

[6. EDITOR USAGE, SETUP & TEST GUIDE](#)

[7. SCRIPTS FOLDERS](#)

[7.1. Clouds](#)

[7.2. Data](#)

[7.3. Extras](#)

[7.4. Farmer](#)

[7.5. FarmHouse](#)

[7.6. Managers](#)

[7.7. Shaft](#)

[7.8. Upgrade](#)

[7.9. Warehouse](#)

[7.10. WorkerManagers](#)

## 1. ONLINE HELP

You can contact us at [damonc.studios@gmail.com](mailto:damonc.studios@gmail.com), so feel free to ask your questions and request new features!

## 2. ABOUT

With Idle Farmer - Tycoon: you can create amazing idle games for mobile devices. Create your own story or game style for new gameplay strategies!

## 3. DO YOU LIKE IT?

If so, please support us on the Unity Asset Store. You can rate ★★★★★ our asset and leave your feedback!

## 4. FEATURES

- Clouds System
- Ads simulation.
- IAP Purchases (First purchase, Remove Ads, x2 Earnings, Hire Managers) MUST ACTIVATE IAP AND SET THE CONFIGURATION AND PRODUCTS AT THE PLAY STORE\*
- Save System - Auto save and auto load between sessions
- Hire random managers to get boosts
- Unlock farm spots and upgrade them to boost profits
- Mobile ready (Android)
- Offline/idle profit calculator
- Speed and load boost multipliers
- Earnings multiplier
- Prefabs for clouds, deposits, farmers, managers, shafts, cards, upgrades and warehouse!
- Compatible with Unity 2021.3.4f1 or higher.

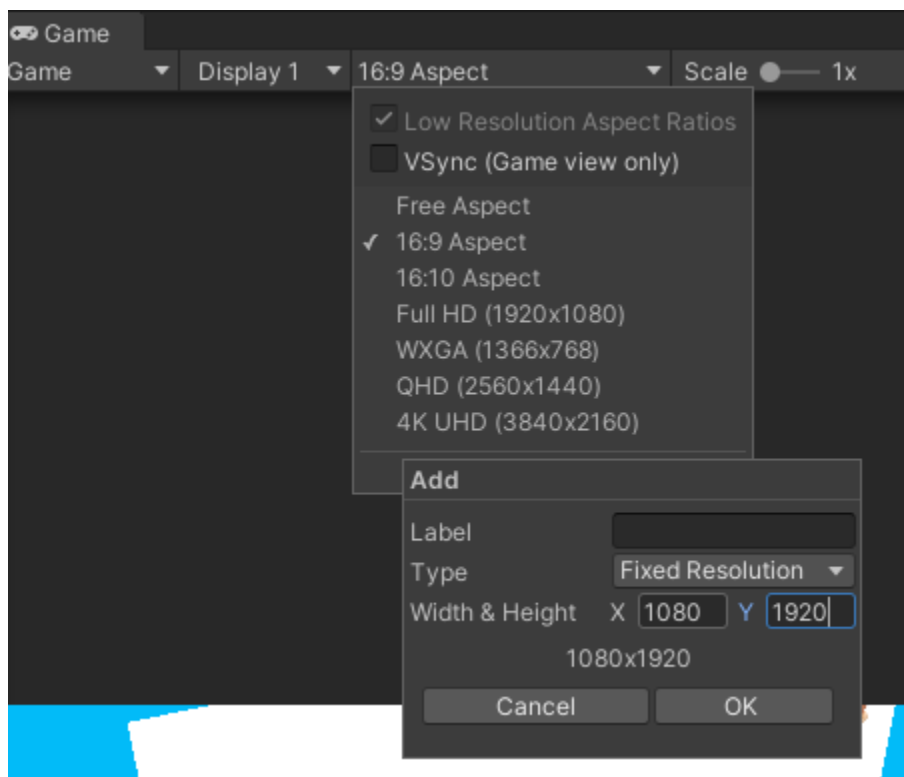
## 5. TECH SPECS

Unity 2021.

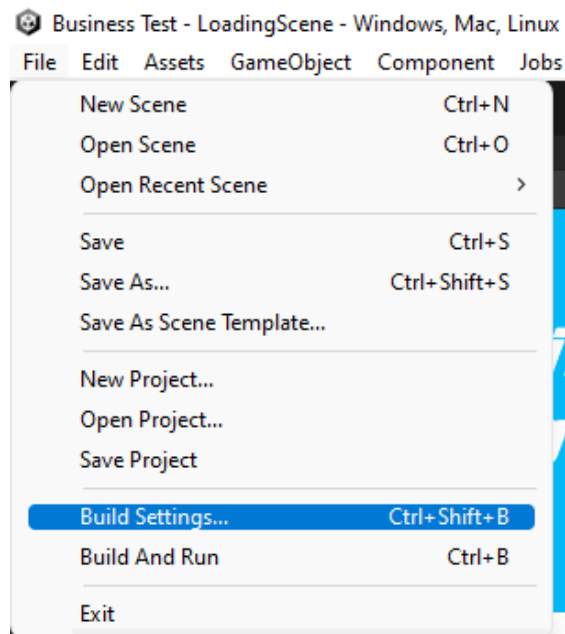
Clean C# source code.  
Mobile friendly.  
Android platform.

## 6. EDITOR USAGE, SETUP & TEST GUIDE

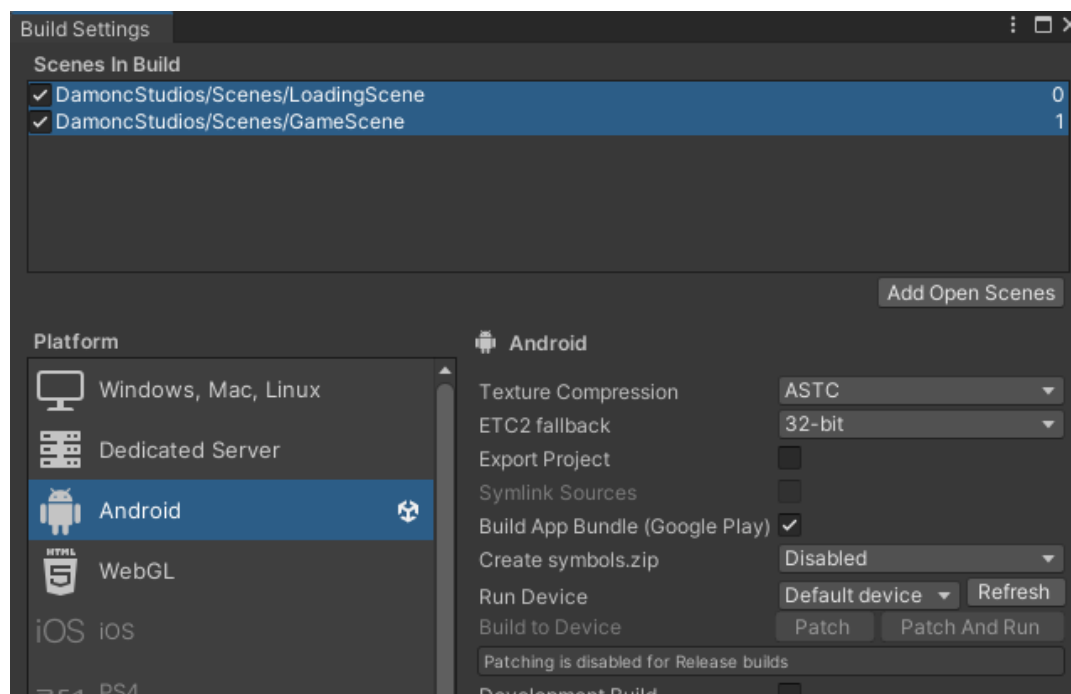
1. Download and install unity package.
2. Make sure you have DamoncStudios folder added to your project.
3. On the Game tab at the aspect section add a new aspect, 1080x1920.



4. Open the DamoncStudios folder and select the "LoadingScene" from Scenes.
5. Open "File" and "Build settings".



6. Make sure that both LoadingScene (index 0) and GameScene (index 1) are added in the same order as the screenshot below.



7. Run scene.

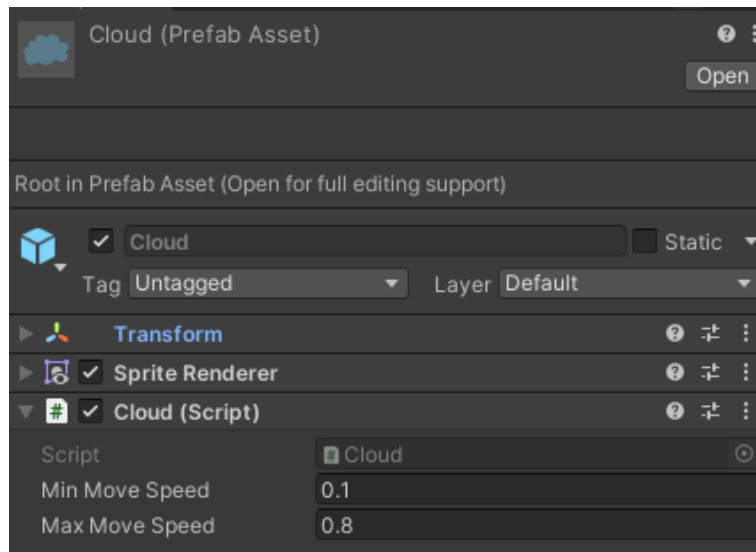
## 7. SCRIPTS FOLDERS

### 7.1. Clouds

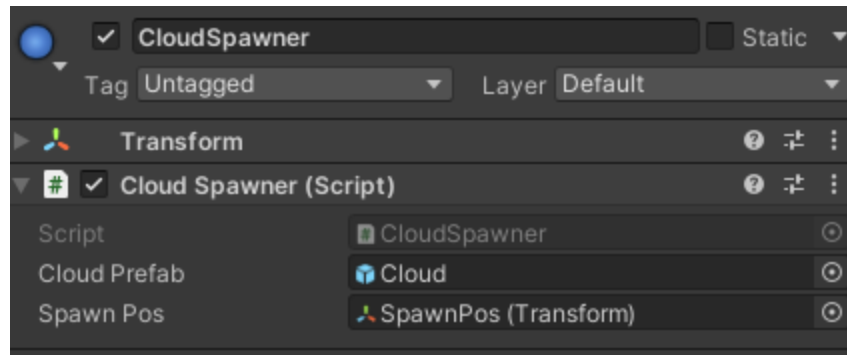
Contains all the information and functionality related to the clouds system.

Files:

- Cloud.cs: script that handles the behavior of the clouds and allows the user to adjust the clouds min and max speed. Visit the folder `DamoncStudios/Scripts/Clouds`. For example:



- CloudSpawner.cs: a script attached to the gameobject named “CloudSpawner” in the GameScene that handles the cloud spawn feature and in there you can set the cloud prefab and the spawn position for the clouds.

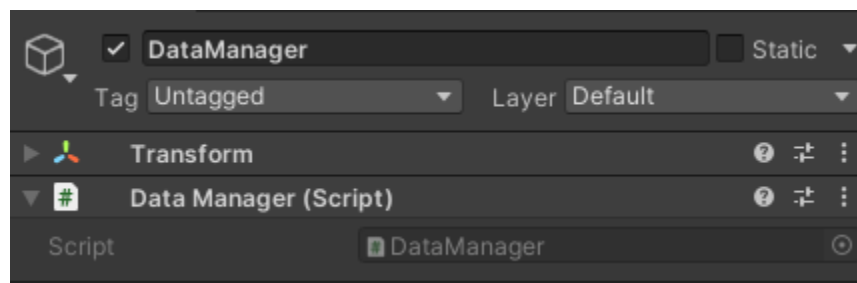


## 7.2. Data

Contains all the information and functionality related to the game data.

Files:

- DataManager: a script attached to the gameobject named “DataManager” in the LoadingScene that handles the current user session data.



- FarmHouseData.cs: a serializable script that contains the information related to the farmHouse level, upgrade cost, etc, for the current user in the GameUserProfile script.

```

namespace Assets.DamonicStudios.Scripts
{
    [Serializable()]
    3 references
    public class GameUserProfile
    {
        public string username;
        public string uid;
        public double totalEarnings;
        public double maxEarningsReached;
        public double sessionMaxEarningsReached;
        public List<UsersShaft> shafts;
        public List<WorkManagerInfo> managers;
        public FarmHouseData farmHouse;
        public WarehouseData wareHouse;
        public string offlineTime;
        public bool firstPurchaseCompleted;
        public bool removeAds;
        public bool multiplier;
        public string deviceId;
        public bool update;
        public string createdDate;
        public string lastTimePlayed;
    }
}

```

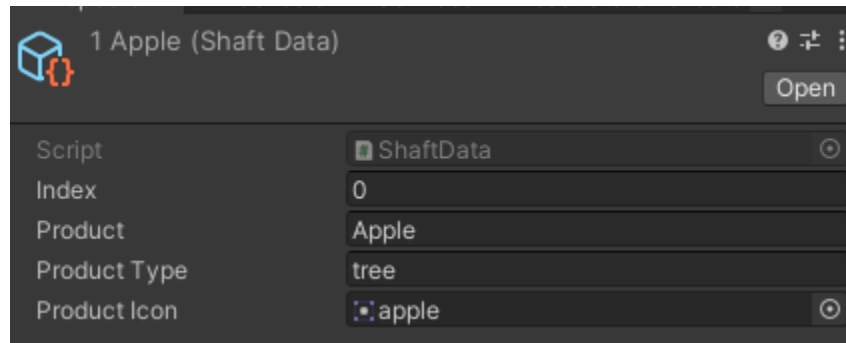
- GameUserProfile.cs: a serializable script that contains all the information related to the current user.

```

namespace Assets.DamonicStudios.Scripts
{
    [Serializable()]
    3 references
    public class GameUserProfile
    {
        public string username;
        public string uid;
        public double totalEarnings;
        public double maxEarningsReached;
        public double sessionMaxEarningsReached;
        public List<UsersShaft> shafts;
        public List<WorkManagerInfo> managers;
        public FarmHouseData farmHouse;
        public WarehouseData wareHouse;
        public string offlineTime;
        public bool firstPurchaseCompleted;
        public bool removeAds;
        public bool multiplier;
        public string deviceId;
        public bool update;
        public string createdDate;
        public string lastTimePlayed;
    }
}

```

- ShaftData.cs: ScriptableObject that handles all the data required to add new shafts to the game. Visit the folder DamoncStudios/ScriptableObjects/HarvestItems. For example:
  - Product types: tree, ground and bush.



- UsersShaft.cs: a serializable script that contains the information related to all the owned shafts for the current user in the GameUserProfile script.



```
namespace Assets.DamoncStudios.Scripts
{
    [Serializable()]
    3 references
    public class GameUserProfile
    {
        public string username;
        public string uid;
        public double totalEarnings;
        public double maxEarningsReached;
        public double sessionMaxEarningsReached;
        public List<UsersShaft> shafts;
        public List<WorkManagerInfo> managers;
        public FarmHouseData farmHouse;
        public WarehouseData wareHouse;
        public string offlineTime;
        public bool firstPurchaseCompleted;
        public bool removeAds;
        public bool multiplier;
        public string deviceId;
        public bool update;
        public string createdDate;
        public string lastTimePlayed;
    }
}
```

- WarehouseData.cs: a serializable script that contains the information related to the warehouse level, upgrade cost, etc, for the current user in the GameUserProfile script.

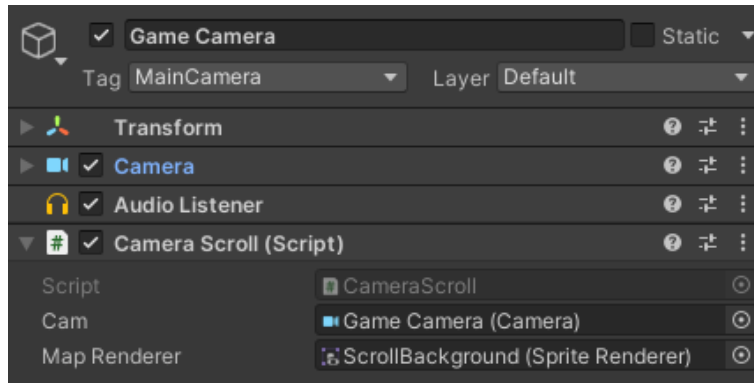
```
namespace Assets.DamonicStudios.Scripts
{
    [Serializable()]
    3 references
    public class GameUserProfile
    {
        public string username;
        public string uid;
        public double totalEarnings;
        public double maxEarningsReached;
        public double sessionMaxEarningsReached;
        public List<UsersShift> shafts;
        public List<WorkManagerInfo> managers;
        public FarmHouseData farmHouse;
        public WarehouseData wareHouse;
        public string offlineTime;
        public bool firstPurchaseCompleted;
        public bool removeAds;
        public bool multiplier;
        public string deviceId;
        public bool update;
        public string createdDate;
        public string lastTimePlayed;
    }
}
```

## 7.3. Extras

Contains all the functionality related to the camera scroll.

Files:

- CameraScroll.cs: a script attached to the gameobject named “Game Camera” in the GameScene that handles the scroll behavior for the game.



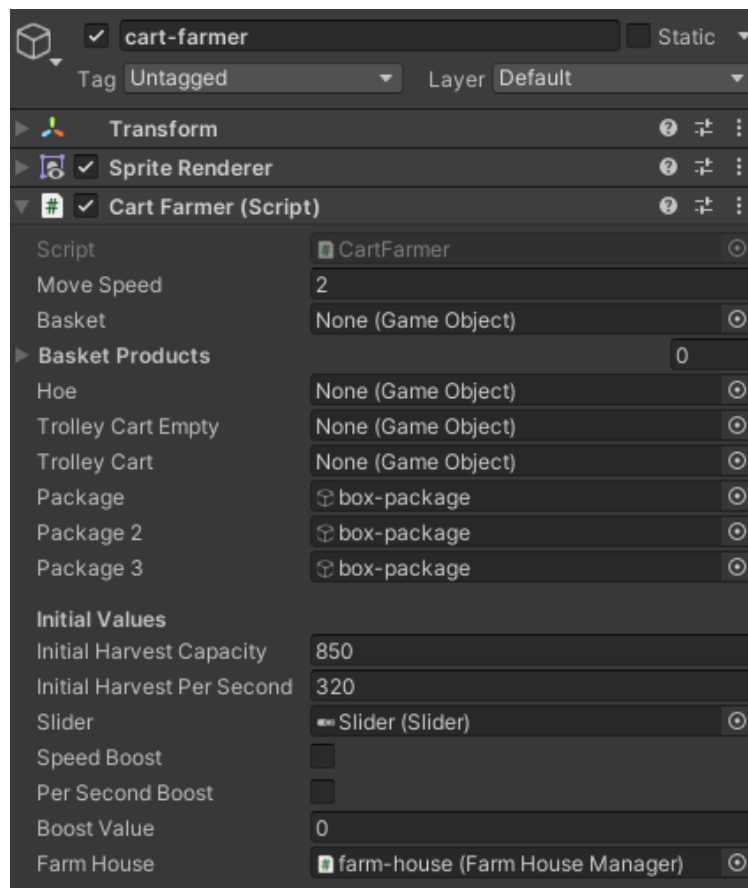
- Currency.cs: a script that handles the currency conversion of some amounts into shorter values and adds prefixes such as K, M, G...
- Deposit.cs: a script that handles the behavior of the farmhouse, warehouse and shaft deposits.
- IClickable.cs: click event that opens the managers popup used in the BaseFarmer.cs script.
- Singleton.cs: a script that handles the singleton pattern.

## 7.4. Farmer

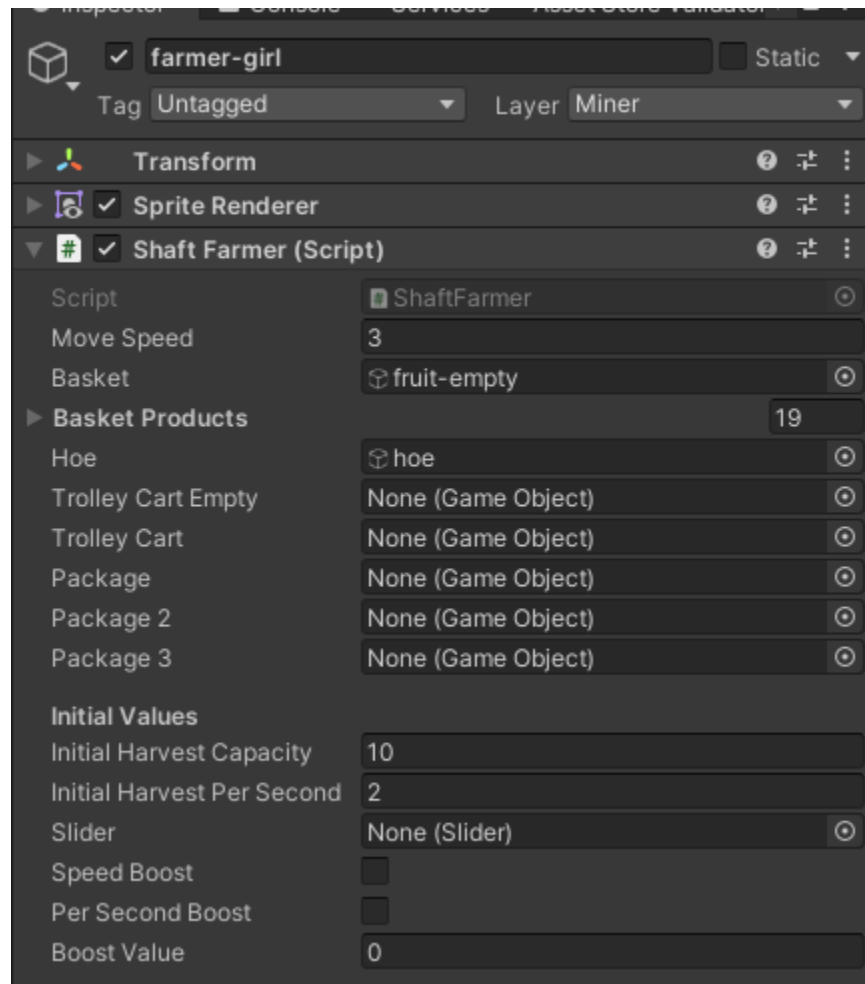
Contains all the information and functionality related to the different types of farmers in the game.

Files:

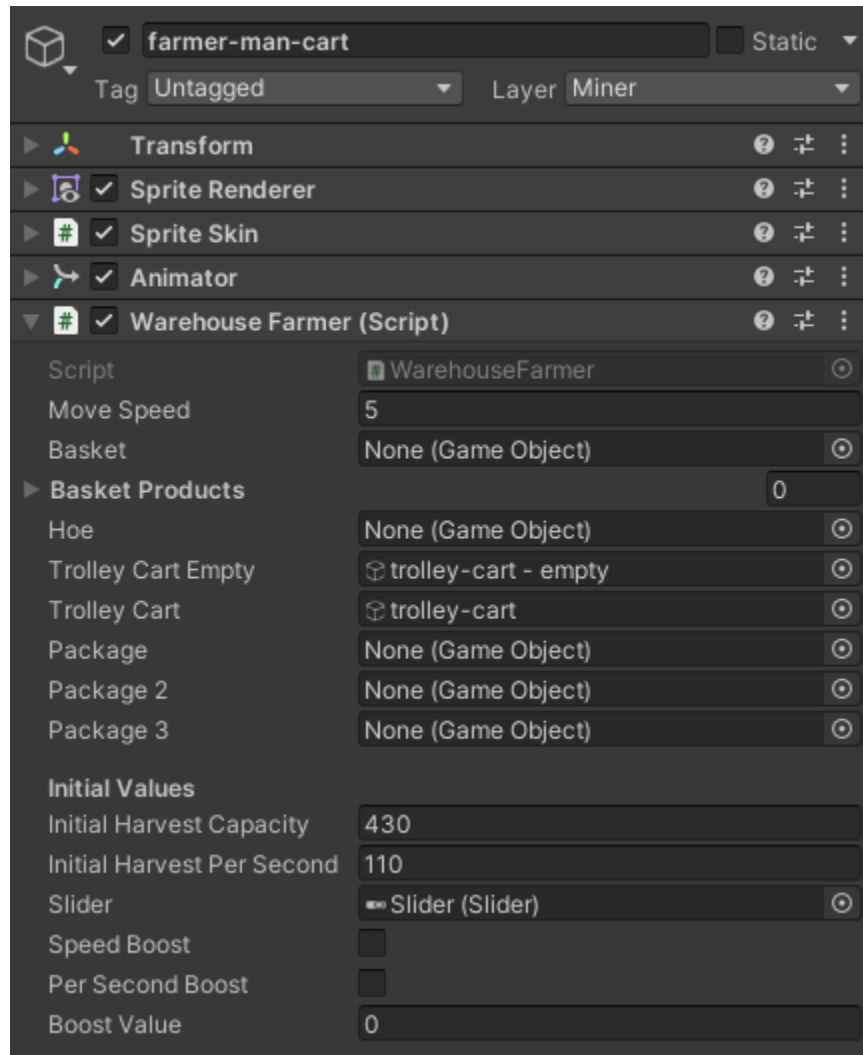
- BaseFarmer.cs: base script that contains all the needed functions for the farmers.
- CartFarmer.cs: a script attached to the gameobject named “cart-farmer” in the GameScene that handles the cart-farmer behavior, where the user can set the initial harvest capacity and harvest speed.



- ShaftFarmer.cs: a script attached to the prefab named “farmer-girl” in the folder DamoncStudios/Prefabs that handles the farmer-girl behavior, where the user can set the initial harvest capacity and harvest speed.



- WarehouseFarmer.cs: a script attached to the prefab named “farmer-man-cart” in the folder DamoncStudios/Prefabs that handles the farmer-man-cart behavior, where the user can set the initial harvest capacity and harvest speed.

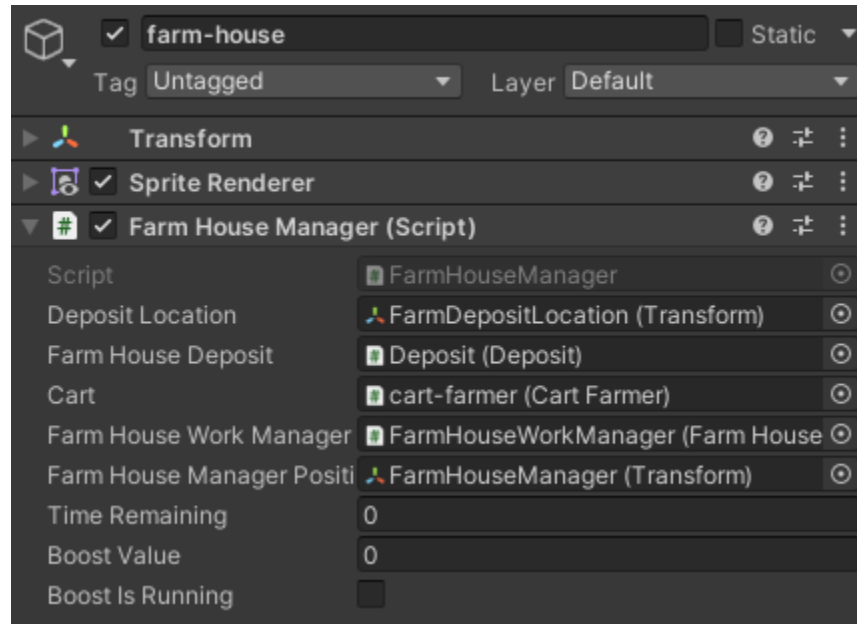


## 7.5. FarmHouse

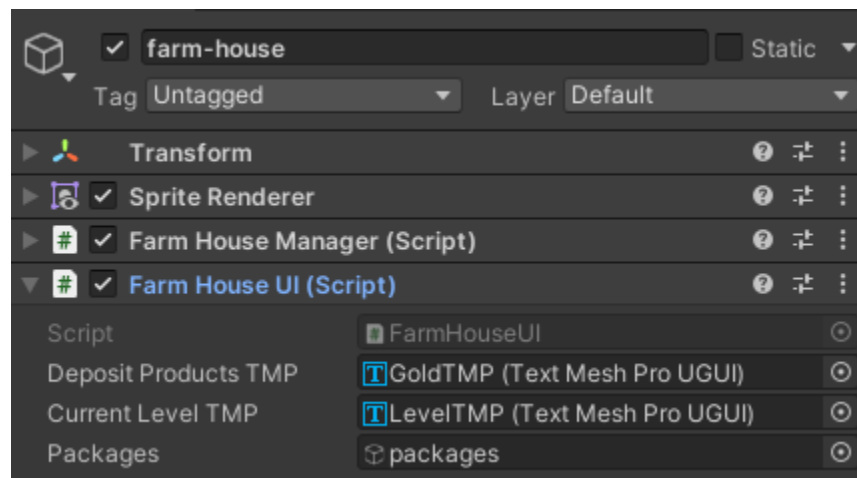
Contains all the functionality and logic related to the farmHouse feature.

Files:

- FarmHouseManager.cs: a script attached to the gameobject named “farm-house” in the GameScene inside the Background gameobject that handles the farmhouse behavior.



- FarmHouseUI.cs: a script attached to the gameobject named “farm-house” in the GameScene inside the Background gameobject that handles the farmhouse UI.

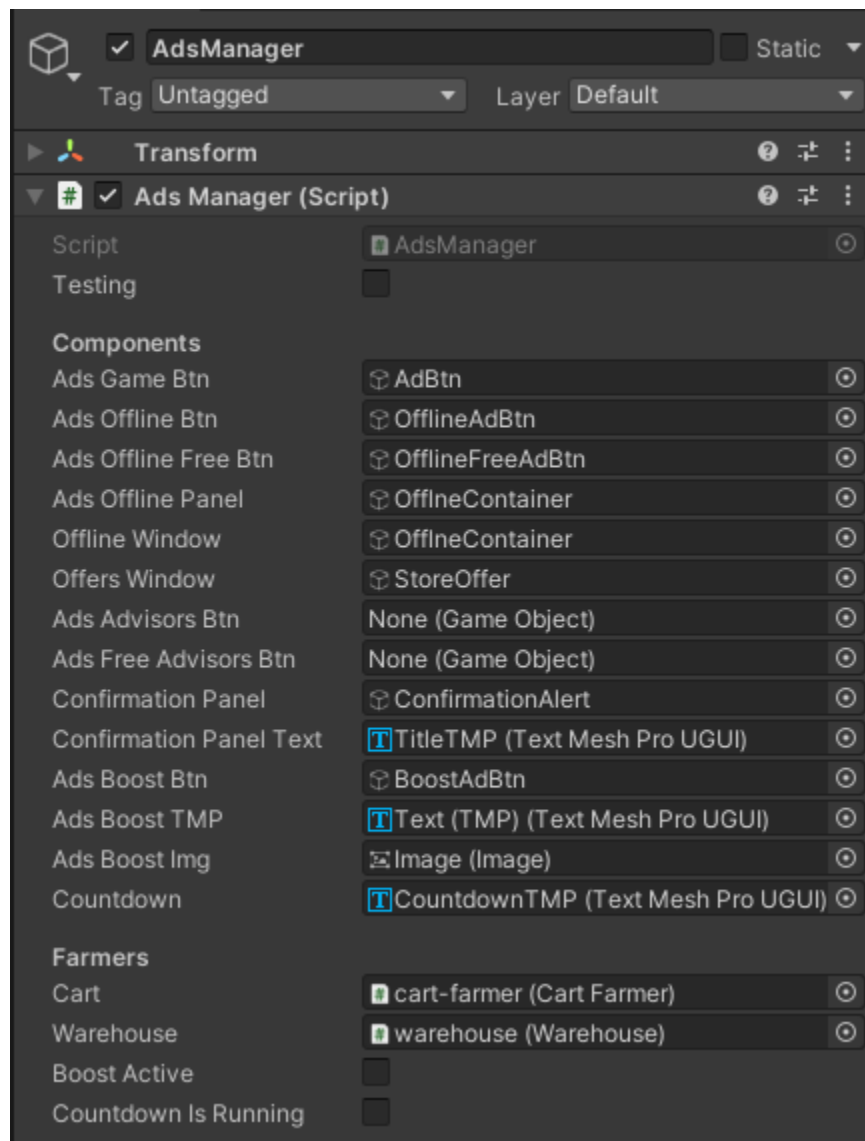


## 7.6. Managers

Contains all the functionality and logic related to the managers that handle all the different game features.

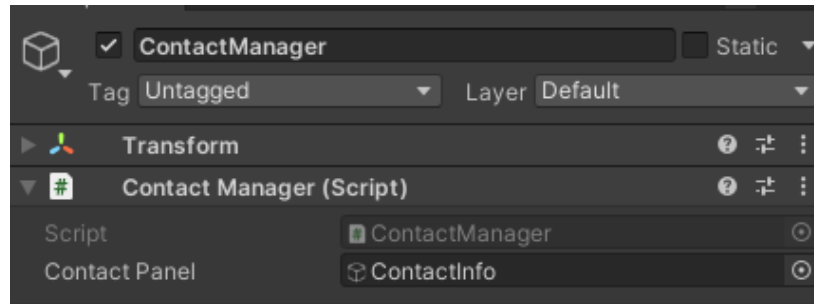
Files:

- AdsManager.cs: a script attached to the gameobject named “AdsManager” in the GameScene that handles the simulation of an ad integration.

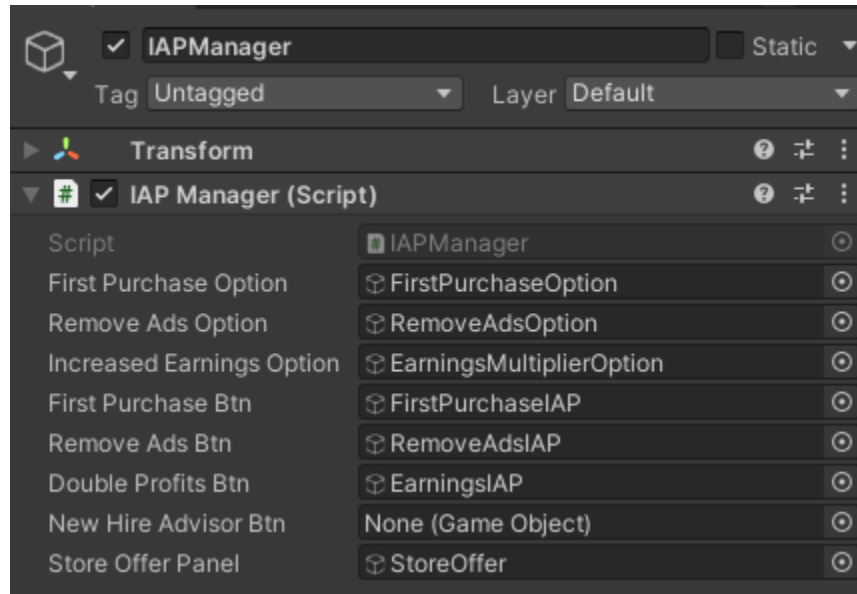




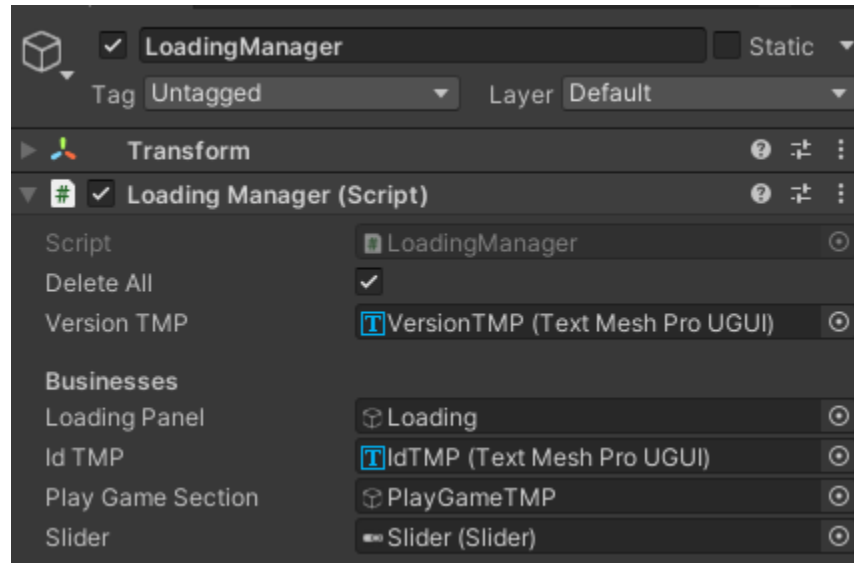
- ContactManager.cs: a script attached to the gameobject named “ContactManager” in the GameScene that handles contact info UI features.



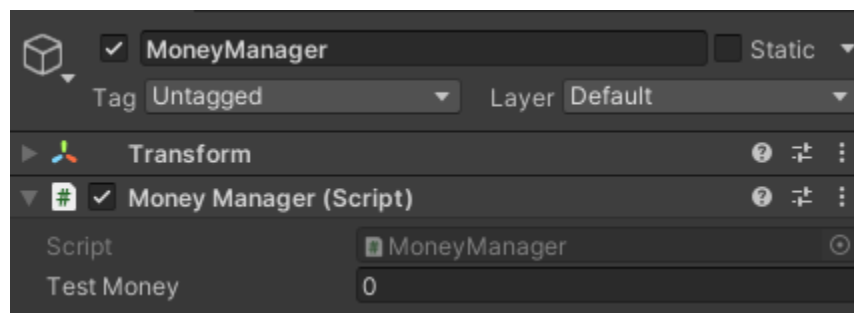
- IAPManager.cs: a script attached to the gameobject named “IAPManager” in the GameScene that handles the IAP UI components and IAP features for buying different game improvements such as removing ads, first purchase and increase earnings.



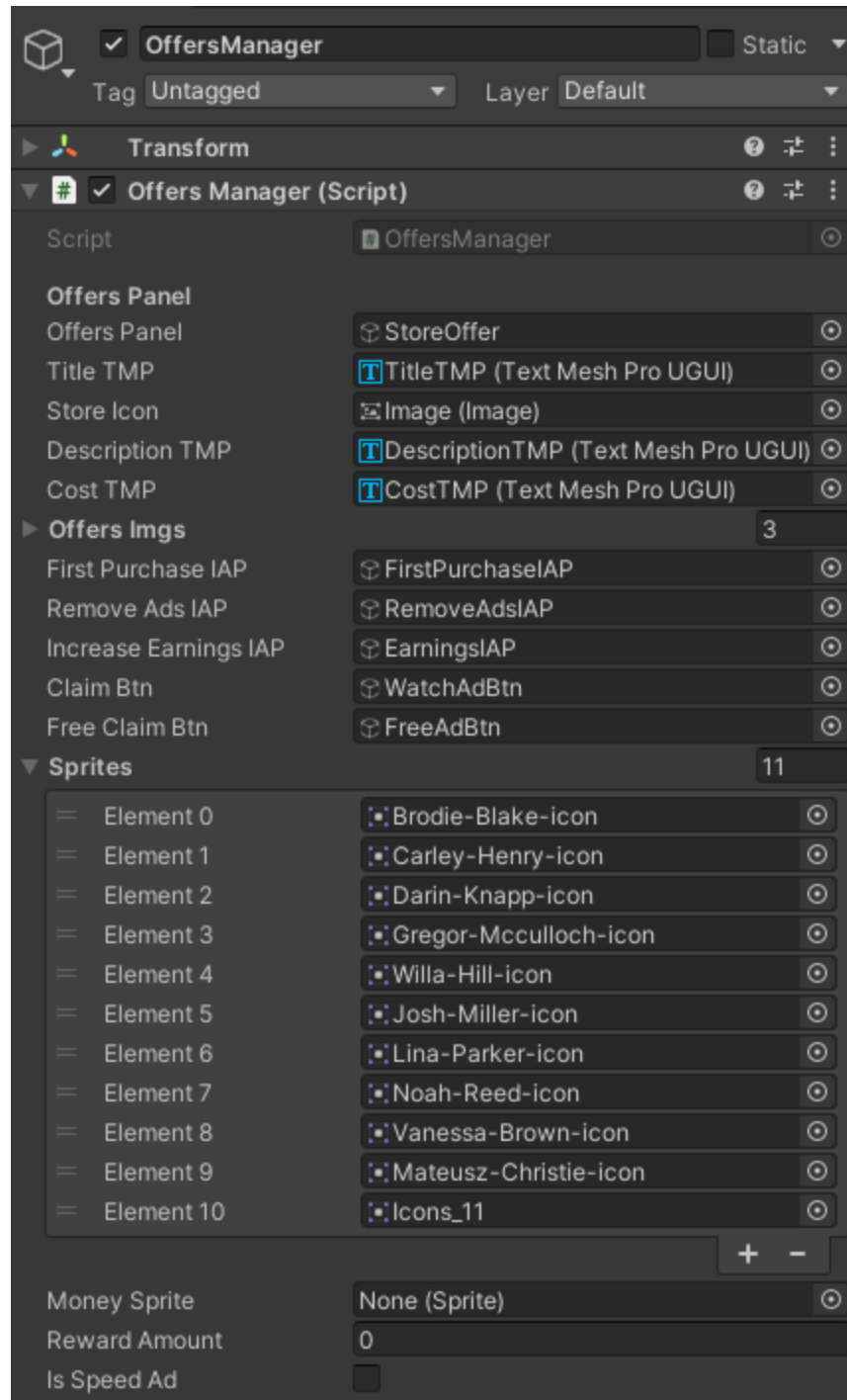
- LoadingManager.cs: a script attached to the gameobject named “LoadingManager” in the LoadingScene that handles the main scene UI components and starts the game.



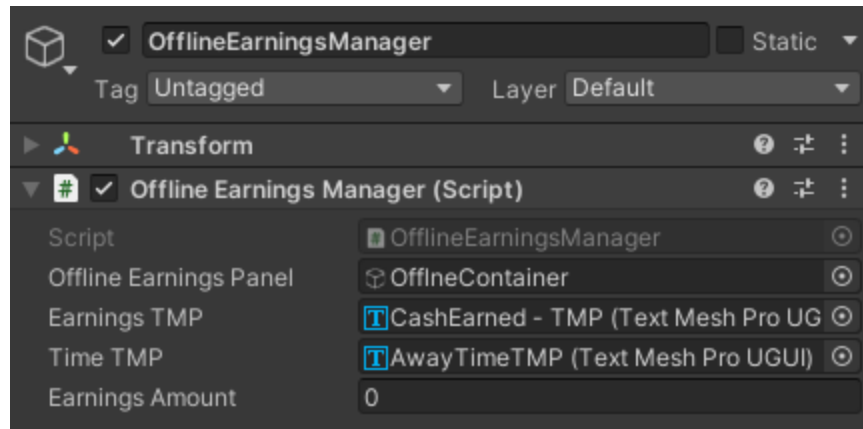
- MoneyManager.cs: a script attached to the gameobject named “MoneyManager” in the GameScene that handles the total earnings for the game session, it allows the user to set a starting amount for testing purposes.



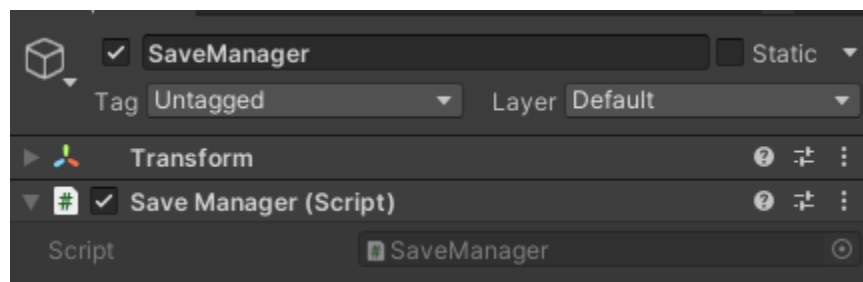
- OffersManager.cs: a script attached to the gameobject named “OffersManager” in the GameScene that handles the offers UI components and shows different ad offers like earn X amount or increase the farm production if you watch this ad.



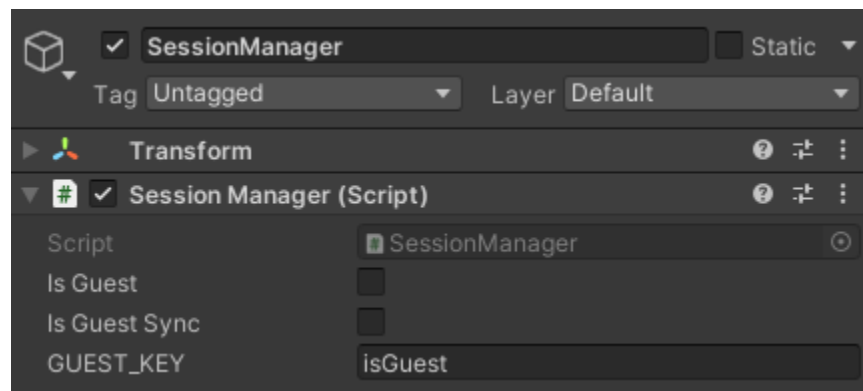
- OfflineEarningsManager.cs: a script attached to the gameobject named “OfflineEarningsManager” in the GameScene that handles the UI offline earnings components and features.



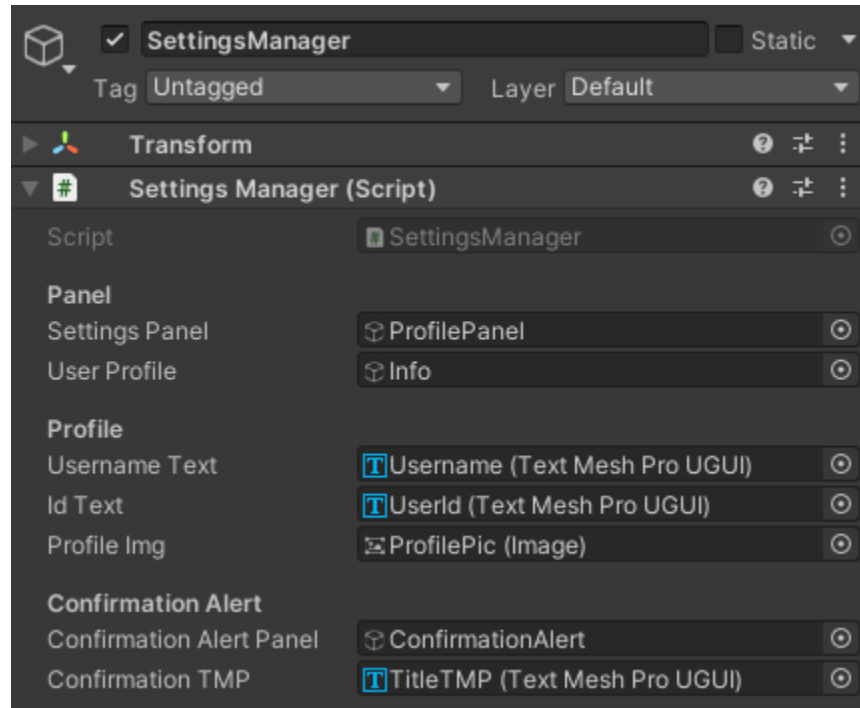
- SaveManager.cs: a script attached to the gameobject named “SaveManager” in the GameScene that saves the session progress every 15 seconds.



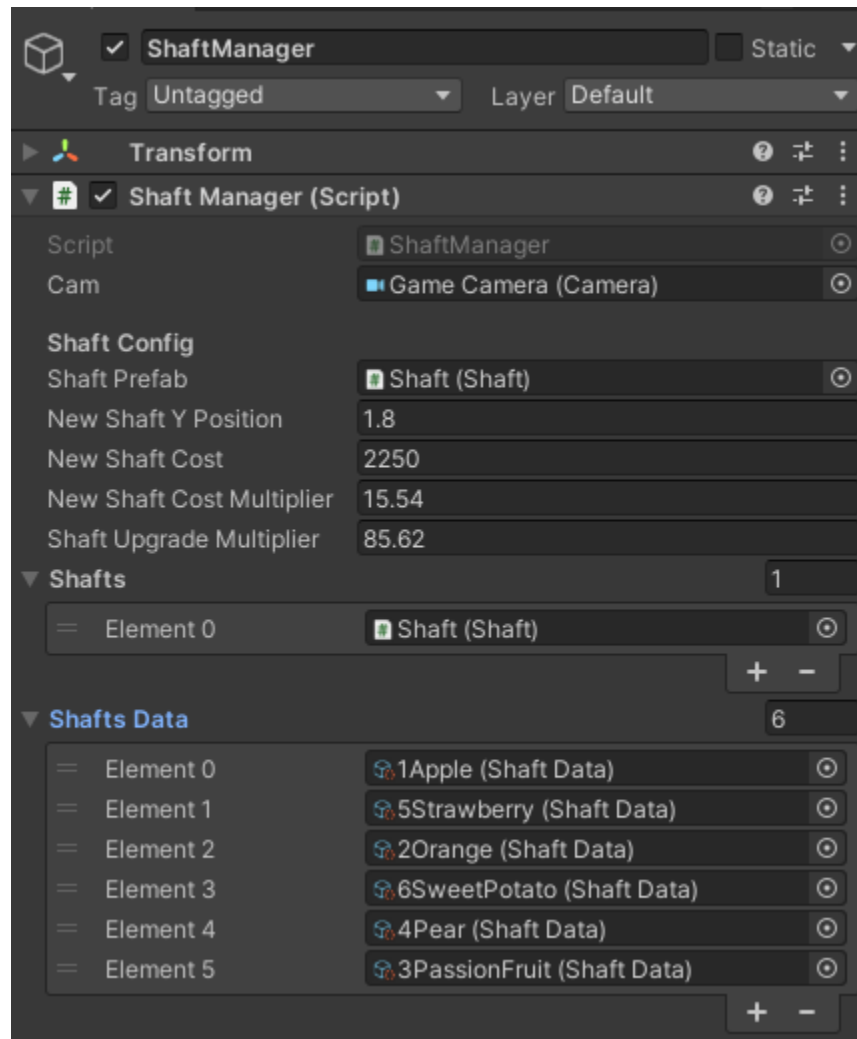
- SessionManager.cs: a script attached to the gameobject named “SessionManager” in the LoadingScene that handles the session of the current user.



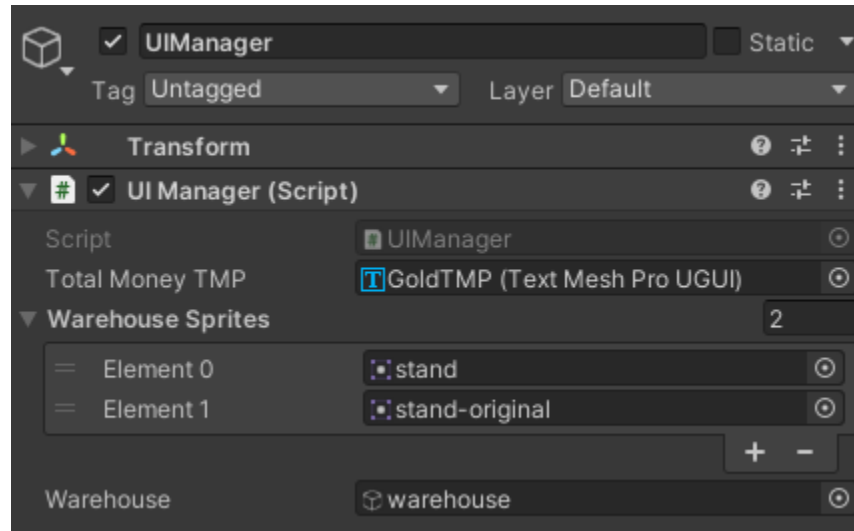
- SettingsManager.cs: a script attached to the gameobject named “SettingsManager” in the GameScene that handles the UI settings components and features in the settings section.



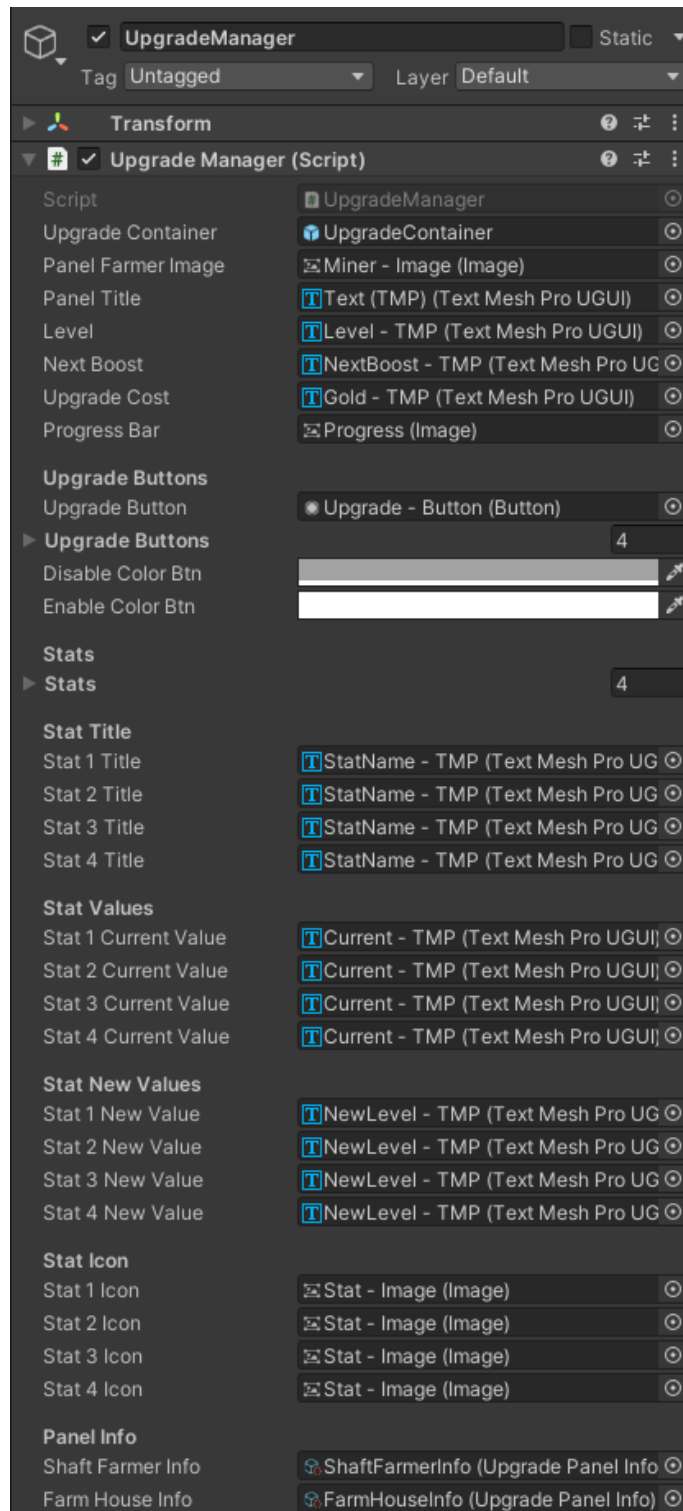
- ShaftManager.cs: a script attached to the gameobject named “ShaftManager” in the GameScene that handles all the possible products that can be added to a shaft, shafts earnings, costs and level/upgrade multipliers. This is the place where you add you new products as a scriptableObjects from DamoncStudios/ScriptableObjects/HarvestItems.



- UIManager.cs: a script attached to the gameobject named “UIManager” in the GameScene that handles the total earnings displayed in the game scene and the stand sprites.



- UpgradeManager.cs: a script attached to the gameobject named “UpgradeManager” in the GameScene that handles the shafts, farmhouse and warehouse upgrades.



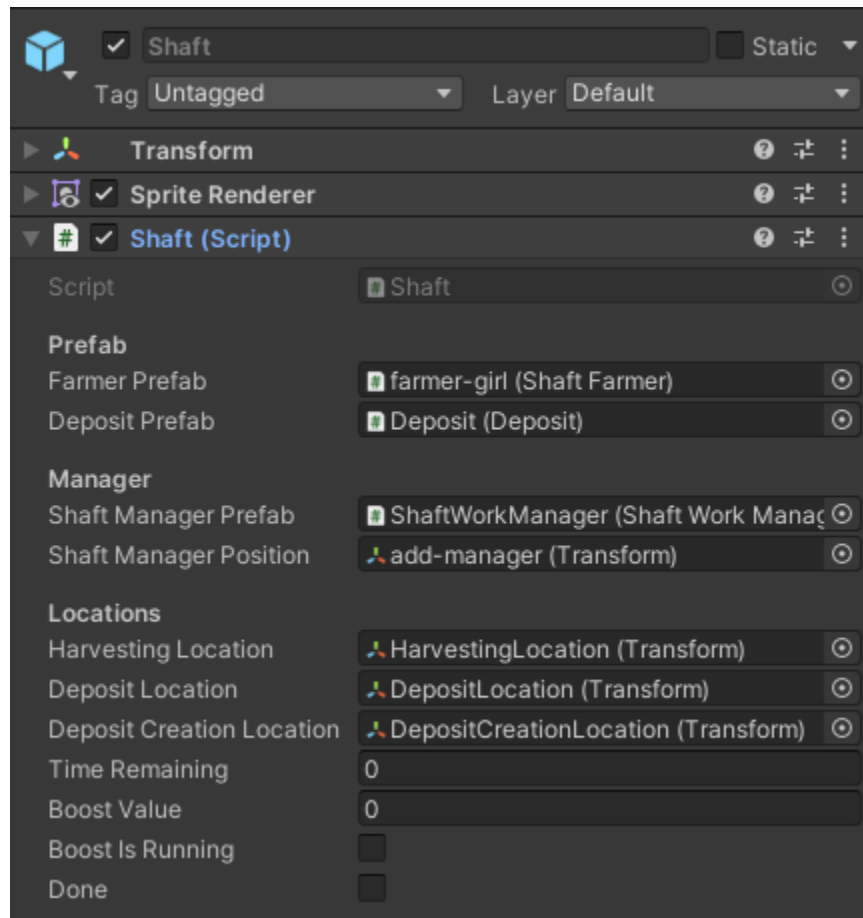


## 7.7. Shaft

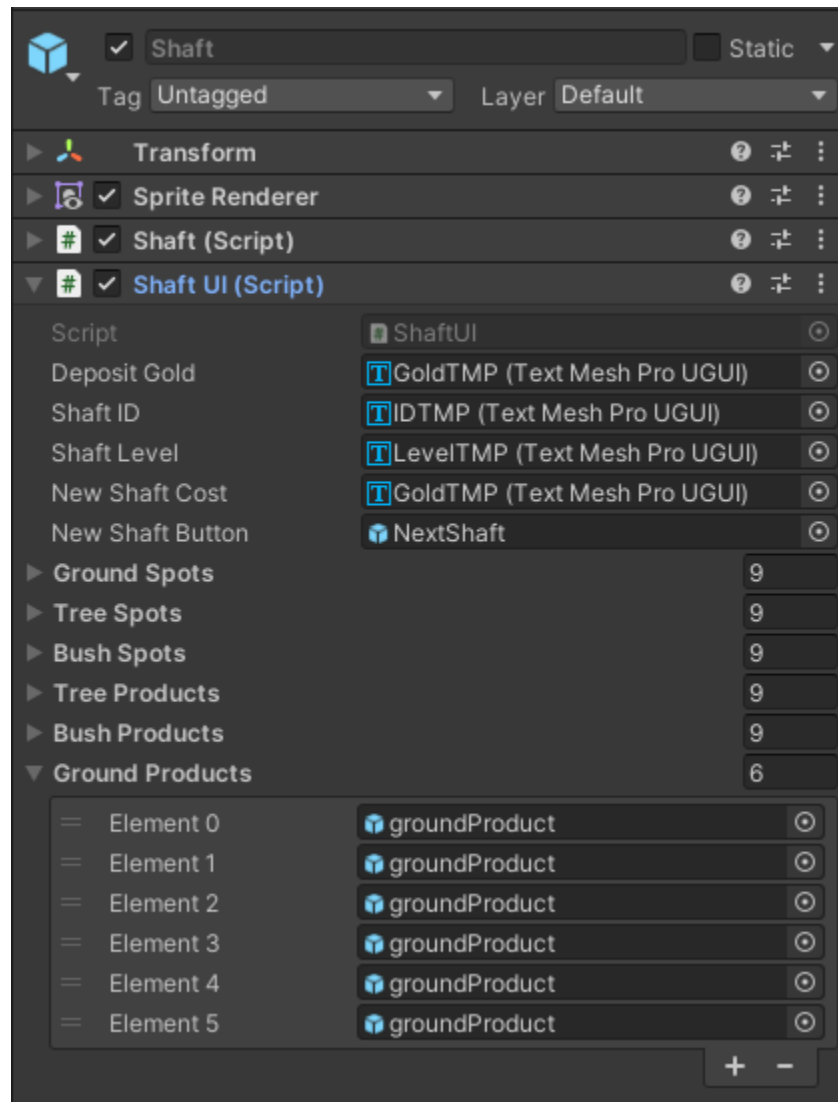
Contains all the functionality and logic related to the shafts feature.

Files:

- Shaft.cs: a script attached to the prefab named “Shaft” in the folder DamoncStudios/Prefabs handles the shafts behavior.



- ShaftUI.cs: a script attached to the prefab named “Shaft” in the folder DamoncStudios/Prefabs handles the shafts UI.



## 7.8. Upgrade

Contains all the information and functionality related to the list of businesses.

Files:

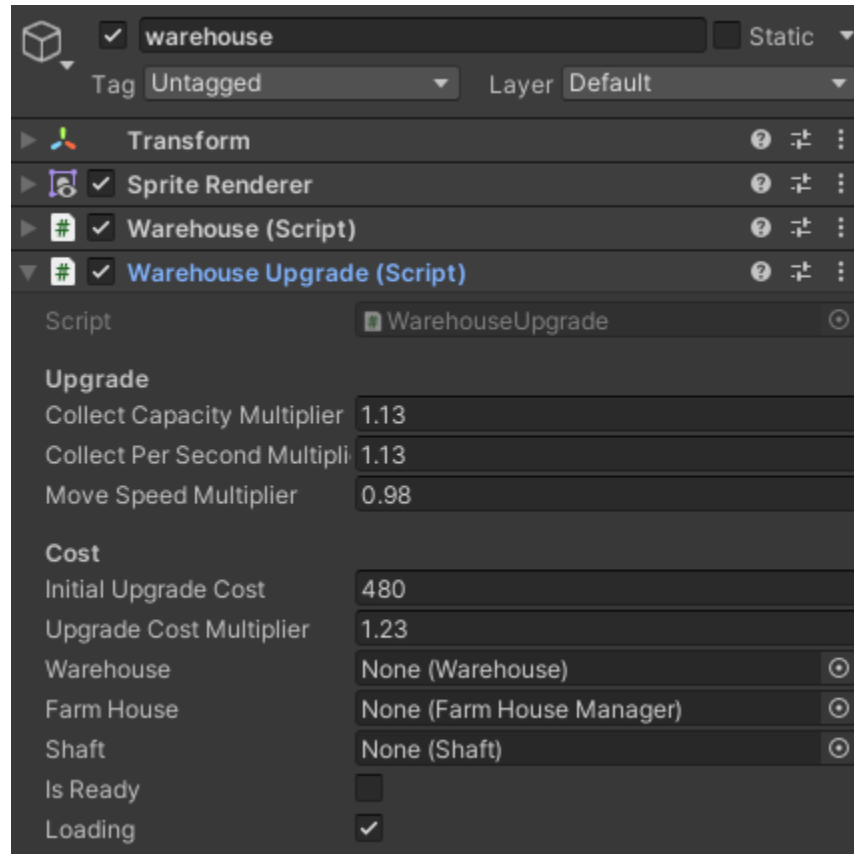
- BaseUpgrade.cs: base script that contains all the needed functions for the upgrades.
- FarmHouseUpgrade.cs: a script attached to the gameobject named “farm-house” in the GameScene inside the Background gameobject that handles the upgrade values and multipliers for the farmHouse.



- ShaftUpgrade.cs: a script attached to the prefab named “Shaft” in the folder DamoncStudios/Prefabs that handles the upgrade values and multipliers for the Shafts.



- WarehouseUpgrade.cs: a script attached to the gameobject named “warehouse” in the GameScene inside the Background gameobject that handles the upgrade values and multipliers for the warehouse.




- UpgradePanelInfo.cs: ScriptableObject that handles the upgrade options UI Panel for the shafts, farmhouse and warehouse. Visit the folder DamoncStudios/ScriptableObjects/Upgrade. For example:

This screenshot shows the configuration window for the 'Farm House Info (Upgrade Panel Info)'. The window has a title bar with a blue cube icon, the title 'Farm House Info (Upgrade Panel Info)', and standard window controls. An 'Open' button is in the top right. The configuration is organized into sections: 'Script' (UpgradePanelInfo), 'Panel Title' (Farm House), 'Panel Miner Icon' (farm-house-icon), and 'Location' (Farm House). Below these are four 'Stat Title' fields: 'Load', 'Move Speed', 'Load Speed', and 'Capacity'. The 'Stat Icon' section contains four fields: 'Icons\_4', 'Icons\_5', 'Icons\_6', and 'Icons\_8'.




Property	Value
Script	UpgradePanelInfo
Panel Title	Farm House
Panel Miner Icon	farm-house-icon
Location	Farm House
<b>Stat Title</b>	
Stat 1 Title	Load
Stat 2 Title	Move Speed
Stat 3 Title	Load Speed
Stat 4 Title	Capacity
<b>Stat Icon</b>	
Stat 1 Icon	Icons_4
Stat 2 Icon	Icons_5
Stat 3 Icon	Icons_6
Stat 4 Icon	Icons_8

This screenshot shows the configuration window for the 'Shaft Farmer Info (Upgrade Panel Info)'. The window has a title bar with a blue cube icon, the title 'Shaft Farmer Info (Upgrade Panel Info)', and standard window controls. An 'Open' button is in the top right. The configuration is organized into sections: 'Script' (UpgradePanelInfo), 'Panel Title' (Farm Shaft), 'Panel Miner Icon' (Icons\_9), and 'Location' (Shaft). Below these are four 'Stat Title' fields: 'Farmer', 'Walk Time', 'Harvest Speed', and 'Capacity'. The 'Stat Icon' section contains four fields: 'Icons\_0', 'Icons\_3', 'Icons\_1', and 'Icons\_8'.

Property	Value
Script	UpgradePanelInfo
Panel Title	Farm Shaft
Panel Miner Icon	Icons_9
Location	Shaft
<b>Stat Title</b>	
Stat 1 Title	Farmer
Stat 2 Title	Walk Time
Stat 3 Title	Harvest Speed
Stat 4 Title	Capacity
<b>Stat Icon</b>	
Stat 1 Icon	Icons_0
Stat 2 Icon	Icons_3
Stat 3 Icon	Icons_1
Stat 4 Icon	Icons_8



Warehouse Info (Upgrade Panel Info)



Open

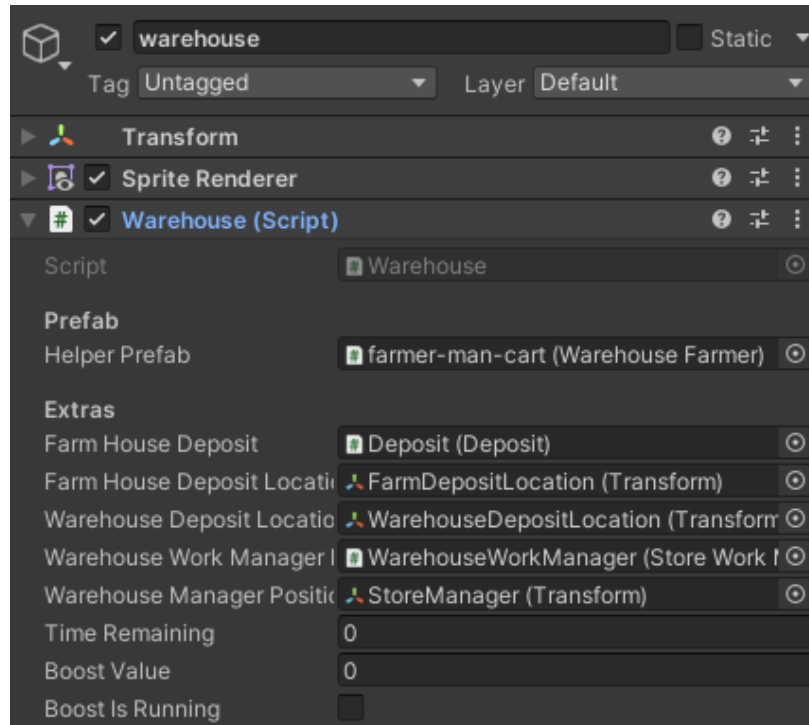
Script	UpgradePanelInfo
Panel Title	Store
Panel Miner Icon	Icons_10
Location	Warehouse
Stat Title	
Stat 1 Title	Transporters
Stat 2 Title	Walk Speed
Stat 3 Title	Load Speed
Stat 4 Title	Capacity
Stat Icon	
Stat 1 Icon	Icons_0
Stat 2 Icon	Icons_3
Stat 3 Icon	Icons_7
Stat 4 Icon	Icons_8

## 7.9. Warehouse

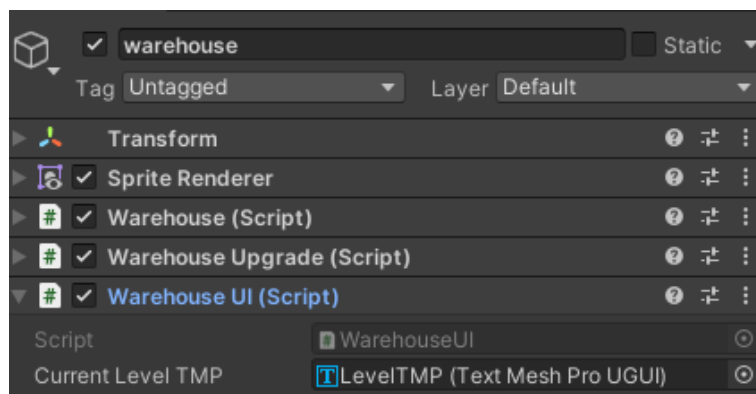
Contains all the functionality and logic related to the warehouse feature.

Files:

- Warehouse.cs: a script attached to the gameobject named “warehouse” in the GameScene inside the Background gameobject that handles the warehouse behavior.



- WarehouseUI.cs: a script attached to the gameobject named “warehouse” in the GameScene inside the Background gameobject that handles the warehouse UI.



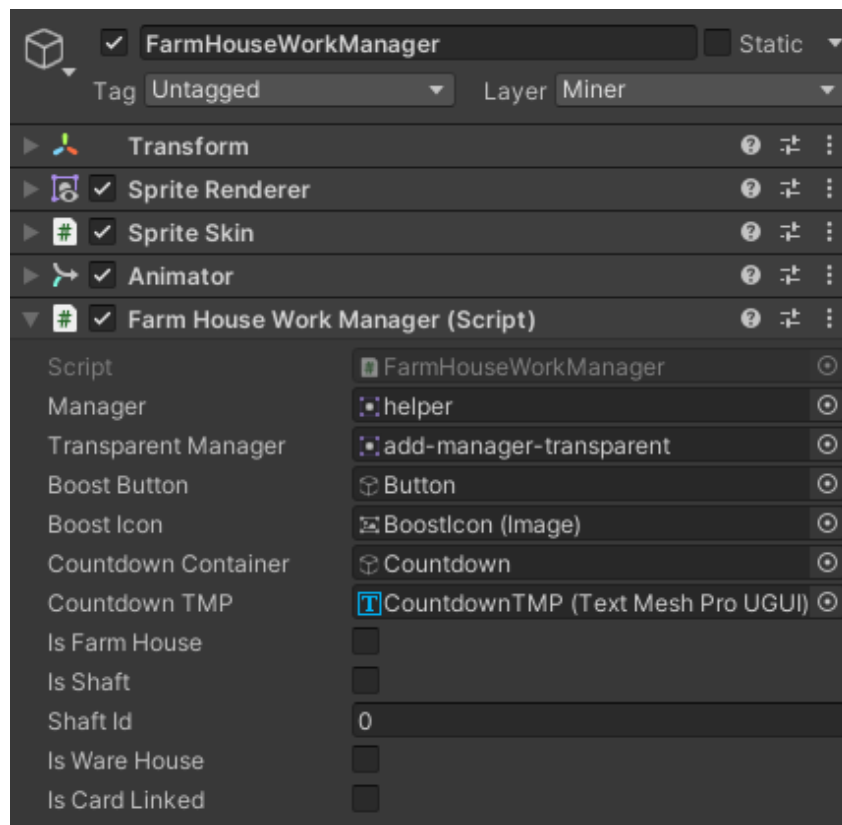


## 7.10. WorkerManagers

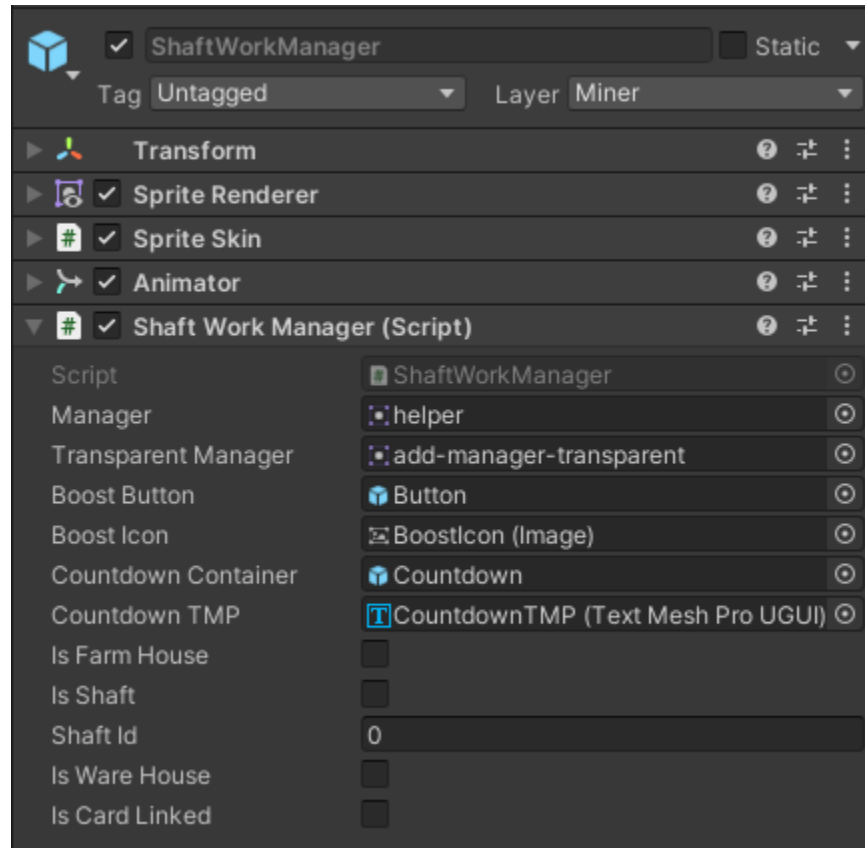
Contains all the functionality and logic related to the managers.

Files:

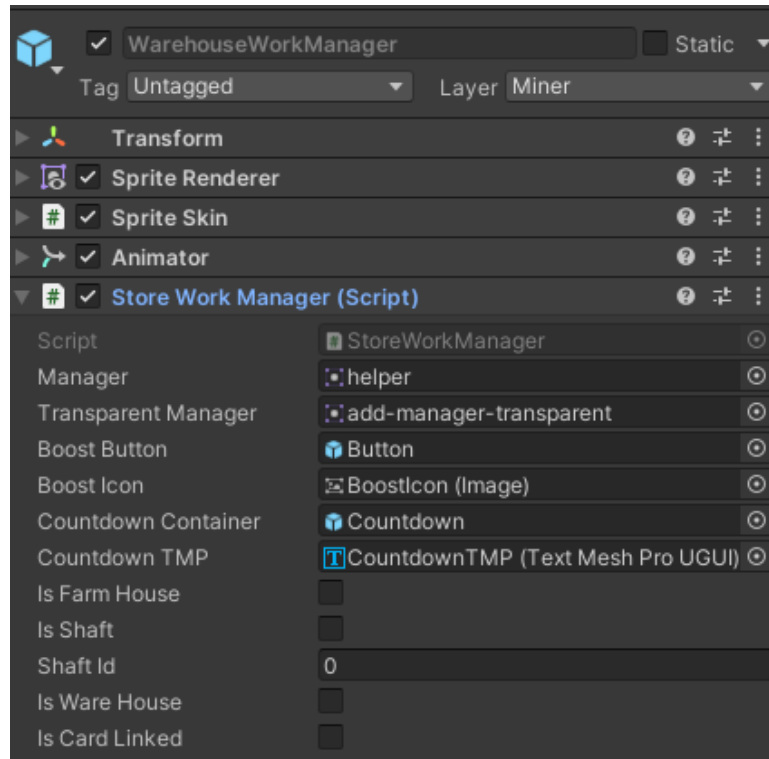
- BaseWorkManager.cs: base script that contains all the needed functions for the managers.
- FarmHouseWorkManager.cs: a script attached to the prefab named "FarmHouseWorkManager" in the folder DamoncStudios/Prefabs that handles the FarmHouse manager behavior and boosts.



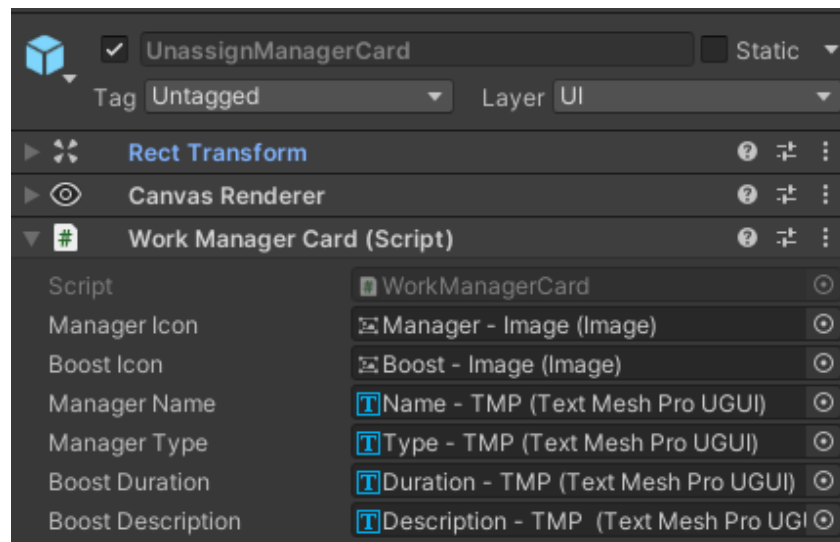
- ShaftWorkManager.cs: a script attached to the prefab named “ShaftWorkManager” in the folder DamoncStudios/Prefabs that handles the shafts manager behavior and boosts.



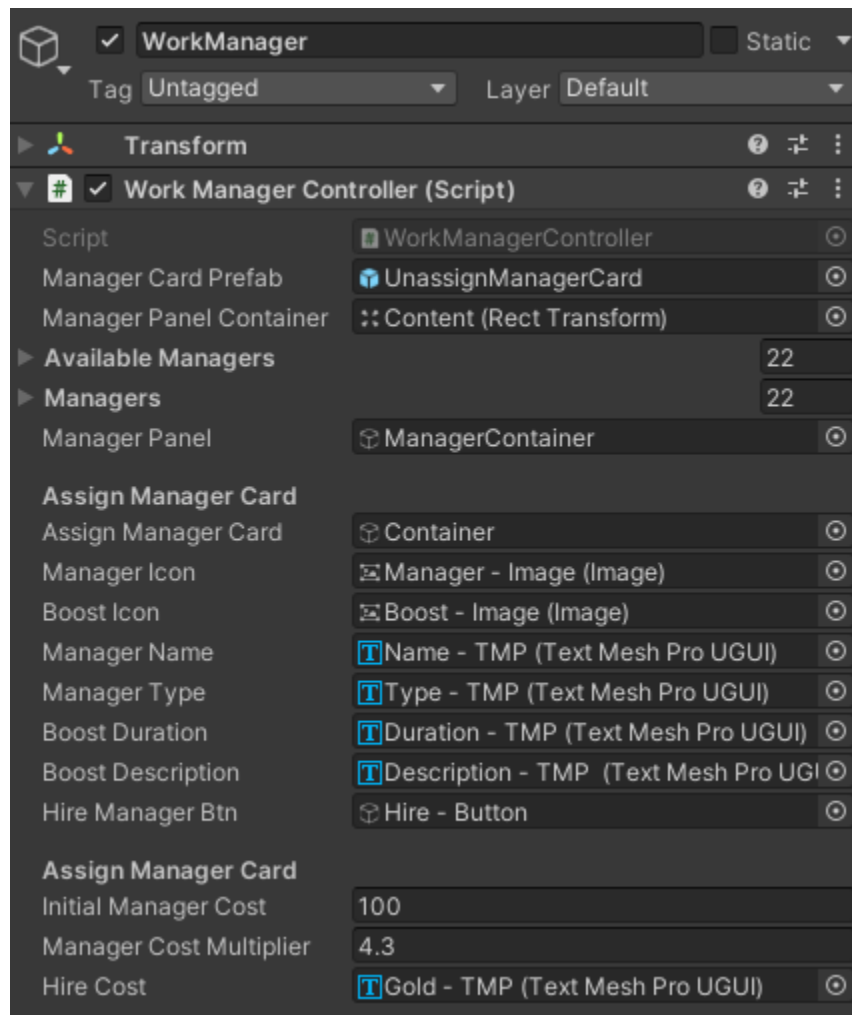
- StoreWorkManager.cs: a script attached to the prefab named “WarehouseWorkManager” in the folder DamoncStudios/Prefabs that handles the warehouse manager behavior and boosts.



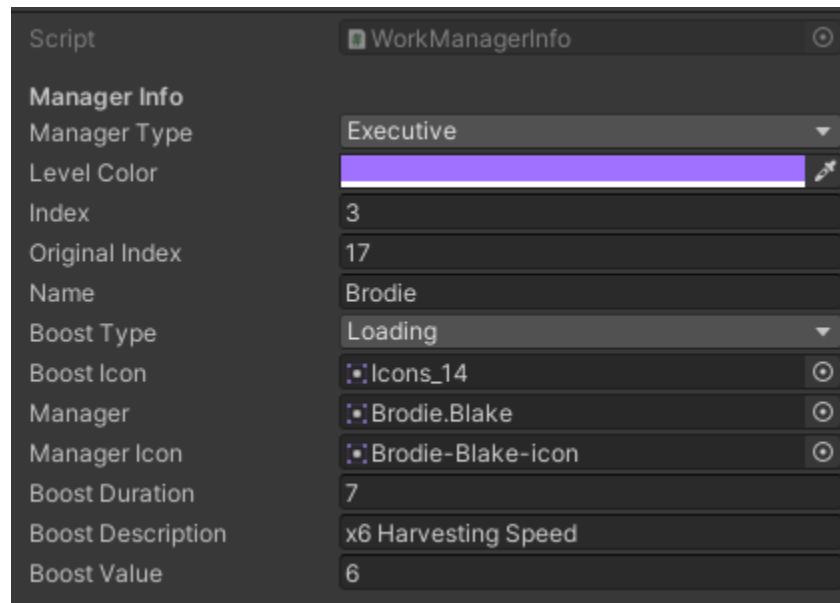
- WorkManagerCard.cs: a script attached to the prefab named “UnassignManagerCard” in the folder DamoncStudios/Prefabs that handles the managers UI for unassigned managers.



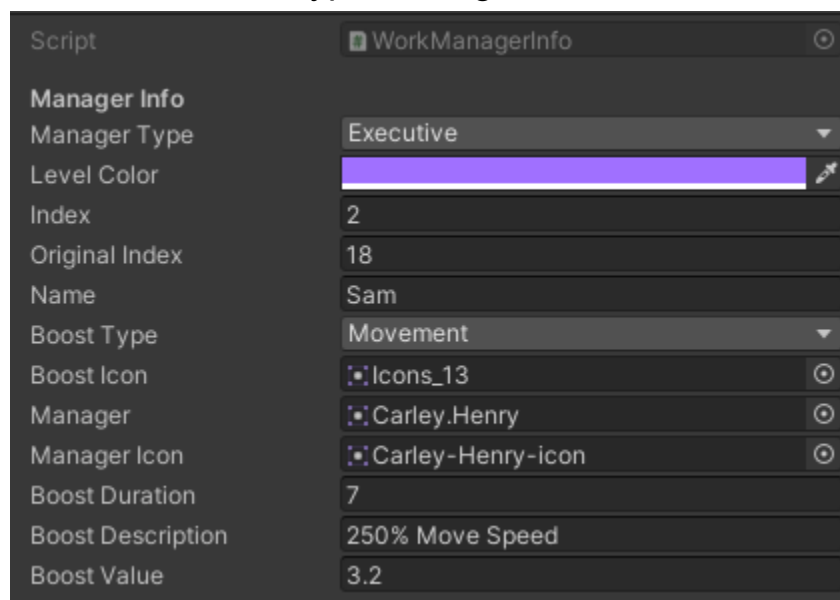
- WorkManagerController.cs: a script attached to the gameobject named “WorkManager” that handles the managers and allows the user to add new ones, you can also set the managers initial cost and multiplier.



- WorkManagerInfo.cs: ScriptableObject that contains all the required information such as name, boost type, boost icon, manager sprite, etc for the different managers. Visit the folder [DamoncStudios/ScriptableObjects/Managers](#). For example:
  - Loading Boost Type Manager:



- Movement Boost Type Manager:



**For any further question please don't hesitate to  
contact us at:  
[damonc.studios@gmail.com](mailto:damonc.studios@gmail.com)**