Lemon Inc – Project Document

Names: Elad Moshe, Omer Gabay  
Workshop: Game Development with Unity  
Instructor: Moshe Sulamy

Project No: 231402

Project Description

**General Overview**Lemon Inc is a virtual reality experience where the player manages a lemonade stand that is set up on the street of a bustling city.  
It is a game that requires an agile response from the player as it is a race against time.  
The player receives orders from arriving customers and needs to serve those orders before the customers lose interest.  
If the player manages to serve the orders on time, they receive cash.  
The main goal is to earn as much money as possible and to grow the business into a thriving one. 

**The Scene**The player spawns next to a lemonade stand which is composed of a table on top of which there are tools and ingredients used to follow the recipes of the orders requested by customers.  
Some examples of tools are: a lemon press, a cutting board, a knife…  
Examples of ingredients: lemons, pink lemonade syrup, mint, sugar cubes…  
Across from the player is a road with moving cars and buildings that represent other businesses.  
In order to make the experience even more realistic, we’ve added housing behind the player to represent the aura of a partially residential area.  
The customers spawn in random locations along the road and approach the player as they get close. A cartoon city with buildings and a street

Description automatically generated**A video game of a house with a car parked in front of it

Description automatically generated**

**The Customers**As customers approach the stand, they form a line in front of it.  
The first customers to arrive then walk over to the stand and a timer appears above each one’s head.  
The timer represents the time a customer will wait until they eventually lose interest and leave.  
The waiting time is assigned as a random value within a certain range that is determined by the customer’s demeanor:

* Difficult / busy customers (doctors, businessmen): 15 – 30 seconds
* Intermediate / patient customers (punks, waiters): 30 – 45 seconds
* Easy / laid-back customers (farmers): 45 – 60 seconds

When the player hovers over a customer, they’re met with an icon that describes the requested order. 

**The Orders**Customers can request lemonade from the following menu:

* Basic lemonade (no added ingredients)
* Sugared lemonade (basic lemonade with varying amounts of sugar measured in cubes)
* Mint lemonade (basic lemonade with varying amounts of mint)
* Mint sugared lemonade (sugared lemonade with varying amounts of mint)
* Pink lemonade (basic lemonade with pink lemonade syrup)
* Sugared pink lemonade (pink lemonade with varying amounts of sugar measured in cubes)
* Mint pink lemonade (pink lemonade with varying amounts of mint)
* Mint sugared pink lemonade (sugared pink lemonade with varying amounts of mint)

**The Recipes**The player needs to follow these recipes to create the respective items:

* Basic lemonade:
  + Grab a lemon from the lemon box.
  + Place the lemon on the cutting board.
  + Slice the lemon with the knife.
  + Take one slice and place in the appropriate slot on the lemon press.
  + Take a glass from the glass stack.
  + Place the glass in the appropriate slot on the lemon press.
  + Point at the lemon press’ handle and click.
  + The glass is then replaced with a glass filled with lemonade.
* Sugared lemonade:
  + Grab a sugar cube from the sugar bowl.
  + Drop the sugar cube down from above a glass of basic lemonade.
  + Repeat until reached amount of requested sugar.
  + The lemonade will turn a lighter shade of yellow to represent a glass of sugared lemonade.
* Mint lemonade:
  + Grab a leaf of mint.
  + Drop the mint down from above a glass of basic lemonade.
  + Repeat until reached amount of requested mint.
  + The lemonade will turn green to represent a glass of mint lemonade.
* Mint sugared lemonade:
  + Grab a leaf of mint / a sugar cube.
  + Drop the mint / sugar down from above a glass of sugared / mint lemonade.
  + Repeat until reached amount of requested mint / sugar.
  + The lemonade will turn light green to represent a glass of mint sugared lemonade.
* Pink lemonade:
  + Grab the bottle of pink lemonade syrup.
  + Lift the bottle so that the top is facing down and above a glass of basic lemonade.
  + The syrup will spray into the glass and the lemonade will turn pink to represent a glass of pink lemonade.
* Sugared pink lemonade:
  + Grab a sugar cube from the sugar bowl.
  + Drop the sugar cube down from above a glass of pink lemonade.
  + Repeat until reached amount of requested sugar.
  + The lemonade will turn a lighter shade of pink to represent a glass of sugared pink lemonade.
* Mint pink lemonade:
  + Grab a leaf of mint.
  + Drop the mint down from above a glass of pink lemonade.
  + Repeat until reached amount of requested mint.
  + The lemonade will turn brown to represent a glass of mint pink lemonade.
* Mint sugared pink lemonade:
  + Grab a leaf of mint / a sugar cube.
  + Drop the mint / sugar down from above a glass of sugared / mint pink lemonade.
  + Repeat until reached amount of requested mint / sugar.
  + The lemonade will turn light brown to represent a glass of mint sugared lemonade.

**Serving**When the player finishes making an order, they need to serve it to satisfy the customer and receive cash.  
Serving the order is as simple as placing it on top of the green tray above the table and selecting the appropriate customer, or alternatively calling the customer’s name which is displayed above them.  
When an order is successfully served, it disappears and the customer leaves immediately (the timer ends).  
If the order was as requested by the customer, the player consequently receives the appropriate amount of cash.  
Otherwise, no money is received.

Project Structure

**Game Engine**The project runs on the [Unity](https://unity.com/) game engine and was made with the Unity editor (version: 2021.3.20f) as a Universal Render Pipeline project.

**Virtual Reality**Unity’s [XR interaction toolkit (version 2.2.0](https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit@2.2/manual/index.html)) is used for all virtual reality features and the game was built to be played on a Meta Quest 2 virtual reality headset and as such, is deployed as an android package kit file (APK).

**Voice Recognition**Meta’s [Wit.AI](https://wit.ai/) natural language processing application programming interface is used to analyze any vocal input from the player.  
This API is available for online use only and therefore the game requires internet connection for any of its voice recognition features.

**AI Navigation**Unity’s built in AI navigation tools were used for non-playable character movement around the scene.  
Every customer was assigned a navmesh agent for pathfinding along marked areas as they’re making their way towards various targets set up throughout the game’s timeline.

**3D Modeling**Unity’s ProBuilder was used for editing 3D models of outsourced assets.  
For example, the model that was used for the lemon press was divided into different parts using ProBuilder’s Face Selection & Detach Faces tools.  
Additionally, some experimental features of ProBuilder were used such as the Boolean (CSG) Tool.

**Assets:**Artistic assets for the game were collected from various sources:

* The Unity Asset Store (<https://assetstore.unity.com>)
* 3D Model Database (<https://3dmdb.com>)
* Quick Sounds (<https://quicksounds.com>)
* Code Monkey (<https://unitycodemonkey.com/>)

**Diving Into The Code**

**CustomerManager.cs**

A script that is responsible for spawning customers into the scene.

The customers spawned into the scene are divided into two collections:

1. customersWaitingAtTable – the customers which are currently waiting for their order to be taken or received when the order is ready.
2. customersWaitingAtLine – customers that are still waiting for their place at the table. A customer will be spawned in a random location from a set of predefined locations in the scene stored in an array of vectors called spawnPositions and will go to the end of the line of customers waiting. In the Update() loop we check if there is an empty place in front of the lemonade stand. If so, the first customer in line will approach the lemonade stand.

Each customer has a NavMeshAgent component

In the CustomerManger.cs Update() loop we check for customers that have received their order or their allotted time to wait had run out and if so, we call the function SendCustomerAway(GameObject customer) – this function sets the destination for NavMeshAgent to the right side of the stand. The customer walks along the new path and is destroyed when the destination is reached.

**Interactable Game Objects**

Juicer – The juicer is the most important tool for lemonade preparation. Without it, the player can’t serve any drinks to their customers.

The Juicer has an XR Simple Interactable component and when its handle is selected, it toggles between two states: Juicer (Opened) and Juicer (Closed). Both are child objects of Juicer and only one of them is active at a time depending on the state of the parent object.

In addition, the Juicer has 2 XR Socket Interactor components.  
One which has an interaction layer mask for Glass where a drink is placed after a lemon is squeezed.

The second socket is for lemons and is only active when the juicer is opened.

When operating the juicer there is a possibility for the player to make mistakes.

For example, closing the Juicer when a lemon is placed in the appropriate socket but no glass is underneath.

In the script SetJuicerState.cs we handle this scenario by keeping the juicer open and using our HapticsController to send a vibration to the controllers. In addition, we also play a sound to convey to the player that they did something wrong.

Lemon Box and Lemons – the game starts with a few lemons on the cutting board ready to be squeezed but the player will surly run out of lemons early in the game.  
Therefore, a box is placed on the player’s side where they will be able to pick as many lemons as they need. The box implements an XR Simple Interactable and when the player points the XR ray towards it and selects it, a lemon is spawned in their hand.

Each Lemon object implements the ISliceable interface with the void Slice() function that is called when a knife collides with it. When collision is triggered, the Slice() function spawns 2 halves of the fruit instead of the original fruit. We created the sliced lemon using Unity’s ProBuilder package.

Glass – on the right side of the table, 2 stacks of glasses can be found. When the player points their XR ray and selects the Glass stacks a new glass is spawned in their hand (*GrabNewGlass.cs*).

The glass is a container for lemonade and it interacts with all the ingredients needed to make a drink. When a sugar cube is added to basic lemonade, it gets brighter, to signify sugared lemonade.

When a glass is released and dropped, a shuttering sound is played.  
This is performed using the GlassCollision.cs script.

Sugar Container – A sugar container is placed on the table.  
When pointing with the XR ray and selecting it, a sugar cube is spawned at the player’s hand.  
The order specifies the amount of sugar cubes required by the customer.

**UI components**

Dispatch – the Dispatch is where an order is placed when it is ready. It appears as a green button at the edge of the desk that contains the text “Place your order”. **A table with a green sign and a knife and a cutting board

Description automatically generated**

The dispatch has a socket interactor component that accepts drinks exclusively.

After placing the drink on the Dispatch an order summary canvas pops up and shows the ingredients that were used in the preparation of the drink.**A screenshot of a video game

Description automatically generated** The order summary has 3 fields:

* Drink Type – Lemonade / Pink Lemonade
* Sugar - amount of sugar cubes used in the drink
* Extra ingredient included: Yes/No (Mint)

After the order summary is displayed, the player will need to select the customer that the drink is for.

This is where an order is evaluated and given a score based on how much it matches the customer’s order.  
An order review is shown on a canvas that pops up where the prepared drink is compared to what the customer ordered.

**Customers**

In order to make the game challenging for the player we decided to implement timers for all customers. When the customer arrives at the lemonade stand the timer kicks off.

We use the customer’s NavMeshAgent component to get the customer’s position and if the customer is in front of the table, an event called *ReachedOrderContactPosition* isinvoked.This event also starts the timer for the customers. Customers wait for their order for a period of 15-60 sec (differs between characters). The timer changes colors as time passes from green to orange and in the last few seconds becomes red.

****

**Live Game Environment**

We’ve added a few scripts to make our city environment lively and dynamic.   
In VehicleSpawnManager.cs vehicles are spawned into the scene.   
We have different types of vehicles and they each have unique audio clips.   
For example, when the Ambulance passes by the player can hear a siren.   
The sounds have a 3D effect and when the car is close by, the sound becomes louder.

In order to make emergency vehicle sounds tolerable and not repetitive, we’ve also made sure that there is only a 15% chance for an emergency vehicle to spawn.

**Voice Control Implementation Using Wit.AI**

We strived to make our game intuitive and simple to play which led us to implement voice commands. In order to activate the microphone, the player must press the Trigger button.

The voice command we’ve decided to implement in our game are for calling the customer to pick up their order by their name.

In order for the command to work, a drink must be placed on the Dispatch.

If the player doesn’t remember the customer’s name, they can hover over the character with the XR ray and a chat bubble will be revealed with their name.

Using Wit.AI’s ML models we were able to train the model with many possible utterances that the player can say to tell a customer that an order is ready.

Example utterances include:

* “Your order is ready, {NAME}”
* “Your lemonade is ready, {NAME}”
* “{NAME}, take your order please”
* “{NAME}, pick up your order”

For the customer names we’ve used common names like Michael, Alex, Emily.

The Wit.ai speech recognition service requires internet connection and each response received from the API has a confidence score. The confidence score reflects how likely the utterance recognized matches the intent. Currently, we’ve decided to implement only one intent in our game which is called *order\_ready*.

In our intent, the model is also trying to detect the customer’s name which is an entity of type “wit/contact”.

Wit/contact entity is the customer’s name which is very important for the model to get right because in case of misinterpreting the customer’s name we may give the order to the wrong customer.

Similar to the intent of the utterance, Wit/contact Entity also has a confidence score and we decided to act upon it only when the confidence score is above 90%.