**Game Report**

Game Back Ground:

The game is based on the 2013 Southeast Asian Haze, where the haze period was caused by large-scale burning in many parts of Sumatra and Borneo. The haze was notable for causing record high levels of pollution in Singapore where the 3-hour Pollution Standard Index (PSI) readings reached a record high of 401.

In this game, you play as a store owner, stocking up on gas masks at the start of each day, and trying to make a profit on the sales!

Gameplay:

At the beginning, the player starts with a sum of money and 3 stores. The player chooses the amount of gas masks to purchase, before assigning them to whichever stores at chosen prices. The day will start after confirmation of stock placement, counting down the day and simulating a situation where the haze is drifting into Singapore.

At the start of each day, the player gets to replenish their stock, and reassign prices and amounts to the store of their choice. The player also gets to move their stores on the map to how they see fit, depending on the sometimes in-accurate weather forecast or the dynamic haze movements.

The game ends after 7 in- game days, where the PSI readings will reach up to even 500 PSI. Success or failure will be determined if the player has earned a profit or loss.

Features:

* Fully randomized haze generation and gas type diffuse system.
* A forecast system that can read patterns and predict the weather.
* Buyers with their own unique preferences.
* Saving and loading of game.
* A singleton sound class.
* A system of money where buyers will buy from the stalls and you get to earn money.
* A singleton time class.
* The world is on a 2.5D grid.
* Tile Class
* Game States
* 2.5D camera with zoom in and out.
* Clicking Class

Haze:

By Roland:

The haze class consists of three parts, the generation of the haze PSI readings, the haze rendering, and the haze diffuse. The generation of the PSI readings was done with a sine graph, with Perlin Noise added in. The rendering of the haze was done in such a way that each tile would contain four tilechilds. All the tilechilds would be interlinked, and they would then render the haze according to its own haze reading, using a HSL to RGB converter for the haze colour. The alpha values of the haze would also be changed to how thick the haze is. Lastly, the diffusing of the haze throughout the tilechilds was done by linking all the childs, then using an application of cellular automata with a diffuse rule. The program runs by taking a value out of the generated sine reading and giving it a factor, than seed it onto the world where it would instantly diffuse itself and render.

Buyer:

By Roland:

The buyer class consists of two parts, the finite state AI itself, and a personality factor. The personality factor is randomized to create three different parameters; distance preference, haze value preference, and money preference. These preferences are just a ratio, and would ultimately be used to determine which shop the player would buy a haze mask from. The finite state AI consists of three states, **Idle walking, Walking to buy, Cannot Buy walking**. **Idle walking** just lets the AI walk to where it is targeted to go, and despawn. If haze conditions are right, it will decide to buy a mask and walk towards a shop, while changing state to **walking to buy**. **Walking to buy** lets the AI walk to the appointed shop. If the shop has masks to sell, it will buy a mask. Else, it would go to the next shop it can buy from. If there are no shops with masks left, they will change to **Cannot Buy Walking,** where they will just walk to their original target and despawn.

Highscore Class:

By Roland:

Reads a file name called “Highscore.txt” and gets the high score list, if it has one. After that, it contains an insert function, where the player can insert their username with their high score and show it out.

Knowledge Applied:

Knowledge gained from all the modules we studied for the past 4 semesters has been applied into the making of this game.

By YingTzu:

Stalls and Money Class:

The money system that the buyers can buy from the stalls and earn money.

If the player buys too many masks, it might cause a loss in profits at the end of the day.

At the end of every day, if the player does not manage to sell all the masks, the masks will not be refunded.

Button Class:

The main control of the whole game play is to use the mouse to click the buttons and the transition from states to states. The player can see the shop they have selected when they click on the shop texture.

States:

Menu:

Before the menu appears, the Singapore map will emerge before the menu appears. The background animation shows a haze simulation.

Tutorial State:

Newspaper animation and brief tutorial explanation.

Buy Mask State:

This state allows the player to buy the masks to sell by clicking the 3 buttons. The cost is stated at the top of the button.

Start of the Day:

This state allows the player to decide the number of masks and price to sell for every shop by clicking the buttons.

End of the Day:

Shows the player how much each shop has earned and total money at the end of the day, and how many masks were sold or left. However, the masks left will not be returned to the player.

Bar class:

Shows the timing to the player.

Money Animation Class:

Animation of money flying up when a customer buys a mask.

Textures:

Most of the textures in the game (Background, buttons and shops).

Wei Qi-

Forecast Class

This class uses a few algorithms.

First, it stores preloaded PSI values from the haze class.

Second, it checks the current day and choose an appropriate range of PSI that the current day would be.

As the days goes on, it checks the values and patterns of the previous days.  
If the values are going up, there’s a high chance to go up and a low chance to go down, and vice versa.

If there’s an up and down (or vice versa) pattern from the previous 2 days, it will consider whether if it will rise or fall using previous values.

Then using an appropriate range of PSI, it use a random number generator. The range with the best possibility will have a higher chance than the other numbers in the range while the numbers outside the range is best not considered since a forecast is not 100% correct.

It gets the result 3 times to get an average and return the average range it generated.

SaveLoad Class, SaveState and LoadState

The States reads information from .txt files and store them temporary in a SaveLoad class.

It will load the values inside the save file you choose and the game will use it.

When the player saves their progress, it will overwrite and show the new information.

Models

The models of the stalls and people were done using simple shapes using maya. As well as UV mapping the stall while someone else did it for the people.

Buying during Gameplay

Using a simple function, it will add more masks to the current shop selected and deducts money.

It shows the current shop’s information like the price of the mask and the remaining amount of masks. Also shows the current money the player have.