**Contribution to project:**

One aspect I was responsible for was designing the game map and the player movement. As a group, we decided to go with a birds-eye view map where the player can move in all four directions. To start myself off, I watched a video guide series by Shaw Code on YouTube1. This was very useful in providing some underlying logic on how to implement collisions and 2D movement in four directions. However, I wrote my code from the ground up, making many adjustments to the approach taken by Shaw Code. For example, I used a list of lists rather than a list of strings for representing the game map. This allowed a much wider variety of game tiles to be placed because 2-digit numbers can be used as elements of the lists. I also came up with alternatives for several methods not taught in lectures. For example, I used a nested for loop instead of ‘enumerate’ to generate the game map, and I used the ‘super’ function as an alternative method for automatically adding sprites to sprite groups. I additionally simplified several ideas for increased efficiency, for example removing redundant position and speed components when implementing wall collisions and camera adjustments.

To create the game map, I adapted a set of Modern Interior graphics from LimeZu. In order to create an authentic PC shop, I customised colours and combinations of these graphics to make objects like computer shelves and displays. In total, I assembled 60 different wall objects that were inserted into the map in various locations. I designed the composition of the map using these graphics as elements.

To assemble the main character sprite and animations, I used an RPG spritesheet by Pipoya3.

I was also responsible for designing the money, score, and time mechanics. I designed the money and score mechanics using my own intuition and based on the banking object classes in an earlier COMP4008 lecture. For the time mechanics, I took inspiration from TokyoEdtech’s YouTube video4 which showed me how to import the ‘time’ module and access UNIX time.

To add the PC components to the map, I coded collisions, random spawning, and statistic boxes (upon mouse hover) using the methods I had learnt from earlier tutorial videos that I watched and using my own intuition. For the component images, I made use of Mikalaimanyshau’s library. I also used an interactive approach between the Player class and the event loop in order to code the item basket, navigation, and item deletion. I used a similar method to implement the checkout menu and lock the player in place until they have made a decision. Following this, I added the ability to end the game and navigate to a game over screen.

After creating my code, I used a refactoring approach to add all non-sprite functionality to an overarching Game class so that each type of menu screen could be called using conditionals. Each type of menu screen has a separate method to condense the event loop and the rest of the game loop.

Other contributions: helping to set up the functionality of the leaderboard along with using the ‘pickle’ library to permanently save high score data, helping to create the enemy health bars, helping with some of the logic behind enemy movement, coming up with the general storyline behind the game.

1https://www.youtube.com/watch?v=crUF36OkGDw&ab\_channel=ShawCode

2https://limezu.itch.io/moderninteriors

3https://pipoya.itch.io/pipoya-free-rpg-character-sprites-32x32

4https://www.youtube.com/watch?v=juSH7hmYUGA&ab\_channel=TokyoEdtech

5https://stock.adobe.com/uk/search?creator\_id=204610883&filters%5Bcontent\_type%3Aphoto%5D=1&filters%5Bcontent\_type%3Aillustration%5D=1&filters%5Bcontent\_type%3Azip\_vector%5D=1&filters%5Bcontent\_type%3Avideo%5D=1&filters%5Bcontent\_type%3Atemplate%5D=1&filters%5Bcontent\_type%3A3d%5D=1&filters%5Bis\_editorial%5D=all&filters%5Bcontent\_type%3Aimage%5D=1&order=relevance&safe\_search=1&k=pc+component&search\_page=1&search\_type=autosuggest&acp=0&aco=pc+compo&get\_facets=0