Implementation Report

For Assessment 3, all requirements given by the brief and our inherited Requirements Document were implemented and are described in more detail below. All code and related files can be found on our GitHub repository. The original documents can be found on Geese Lightning's website.

While the inherited code contained a completely different Collision Detection system to which we were familiar and so caused a few problems over the assessment period, the team decided not to change what had already been implemented in the game and expanded on what were given. We tried to change as little as possible in the architecture and requirements, however, some modifications had to be made as key parts of the brief were not included in the requirements.

No major changes to the original architecture were made: the table below highlights any significant changes made, along with the section of code which fulfills each requirement where possible.

As each change we have made relates to a requirement, the table of changes shows which requirement changes are linked to. We state the classes in which the changes were made and have a section where we go into some detail about what the changes actually did and how they were implemented. Finally, we have the Relevant Code References which can be used to directly find the sections of code that were added/modified. These references are found by going to the class in the code referenced to by "In <classname>:" and then using ctrl-f and typing in the reference id to then locate the change in code. Every change has both a comment to show the start and the comment to show the end of a block of changes. As an example, looking at "In TicTacToeScreen: TICTACTOE" would direct you to go to the TicTacToeScreen and then use ctrl-f to find the TICTACTOE reference.

Requirements ID	Class	Change	Relevant Code References
P2	Level TextScreen	There existed a bug in the original code which would cause an EXCEPTION ACCESS VIOLATION error and then crash the game. After much debugging, we found that this was due to a bug in the Stage.dispose() function, in particular with disposing of the batch. Removing explicit calls to Stage.dispose() fixed this bug. It was also noticed that the skin in TextScreen was not disposed of, so this was also remedied.	In Level: REMOVESTAGEDISPOSE In TextScreen: CALLDISPOSE DISPOSESKIN
F5	TicTacToeScreen	We added a Naughts and	In TicTacToeScreen:

	Constants MinigameVars	Crosses mini-game to the main menu. It is accessible via a button on the main menu and it will start to run without affecting the main game. The game is implemented as a 3x3 grid of 9 buttons which can be clicked to choose a cell for the noughts and crosses game.	TICTACTOE In MinigameVars: MINIGAMEVARIABLES In Constants: POINTGAINVALUES
F6	PowerUpSlow PowerUpDamage Constants Level	New power-ups were added to the game (Slow and Damage). They have their own classes respectively. Additionally, we added some timer values to the constant class to allow their effects to wear off. As with all power ups, these are made by extending the Power Up class that already existed when we started.	In PowerUpSlow: SLOWPOWERUP In PowerUpDamage: DAMAGEPOWERUP In Constants: DAMPOWERMAGNITUDE DAMPOWERTIME SLOWMAGNITUDE SLOWTIME In Level: SPAWNPOWERUPS
F1	BusStopLevel ComputerScienceLevel GlassHouseLevel	There is a variety of new locations added to the software each with the own class respectively. And each has their own buttons added to the main menu. These buttons are greyed out when areas are not meant to be accessible. Finishing each level increments the progression variable in Zepr if the level has not already been finished.	In BusStopLevel: BUSSTOP In ComputerScienceLevel: COMPSCI In Level: ENDLEVEL LEVELCOMPLETE In GlassHouseLevel: GLASSHOUSE
F7	Level Constants	In the game, we have added two new boss entities. Each has its own texture and in-game stats. Both are significantly larger than the other zombie characters in order to make it easy to see that they are more powerful, and they have higher stats than normal zombies. The first boss appears on level 3 (Courtyard) and the final boss appears on level 6 (Glasshouse)	In Level: BOSSSPAWNED FIRSTBOSSSPAWN FINALBOSSSPAWN SPAWNBOSSES In Constants: FIRSTBOSSSTATS FINALBOSSSTATS
F3	Constants Player SelectLevelScreen	We have made one player and just like the new boss entities it has its own texture and in-game statistics. This character is called stJohn and is typified by having both the lowest speed and lowest health of any playable character in	In Constants: STJOHNSTATS In Player: STJOHNTYPE STJOHNGRAPHIC In SelectLevelScreen: INITNEWCHARBUTTON NEWCHARDESCRIPTION

		order to provide the player a ‘hard mode’.	DISPLAYNEWCHARBUTTON CHARBUTTONCLICKLISTENER
F4	Constants Level	In addition to the new players, there are two new zombies added to go alongside the already existing one, with their own texture and in-game statistics. These zombies all differ from one another in both health and speed, and when zombies are spawned the game randomly chooses between these 3 types of zombie. We have also improved the health bars system so that it can be used for zombies with varying health levels.	In Constants: ZOMBHEALTH ZOMBSTATMOD In Level: SPAWNRANDOMZOMBIE BETTERHEALTHBARS
F8	SelectLevelScreen	A new save function has been implemented this allows the player to pause there progress between levels, by the function saving the state of the game into a text file. This text file stores only 2 numbers, the points and the number associated with the current progression state of the game. A load function has also been implemented which simply reads this text file and updates the points and progression values before reloading the page so as to display these changes.	In SelectLevelScreen: IMPLEMENTSAVE IMPLEMENTLOAD
F12	Level Constants Zombie	There is now a new safe area implemented into the software allowing the player to walk around without the enemies attacking them as they will not be able to spawn. Entering this area will also cause the zombies throughout the game to be more difficult to combat by increasing all of their stats. This is built into the functionality of the zombies.	In Level: SAFEAREA In Constants: SAFEAREAPOINTS In Zombie: SAFEAREADIFFICULTYINCREASE
F11	Zepr Constants Zombie Level	A points system has been added so that the player can accumulate points by avoiding and defeating enemies,	In Zepr: POINTSVAR IMPLEMENTPOINTS In Constants:

	SelectLevelScreen	finishing the minigame and by entering the safe area for the first time.	POINTGAINVALUES In Zombie: GRANTZOMBIEPOINTS In Level: POINTCOUNTERLABEL INITAVOIDTIMER SAFEAREA UPDATEAVOIDTIMER POINTCOUNTERSTRING SETPOINTCOUNTERSTRING DISPLAYPOINTCOUNTER In SelectLevelScreen: SELECTLEVELPOINTLABEL INITSELECTLEVELPOINTCOUNTER POINTCOUNTERMENUBAR UPDATESELECTLEVELPOINTS
N2	Zepr SelectLevelScreen	The state system that already existed within Zepr to keep track of the player's progress through the game has been updated to include the new levels, the level select screen has been updated to allow access to these levels in order to facilitate the progression and the levels with bosses will only end when the bosses have been defeated (with only the final level displaying the Game Complete screen).	In Zepr: LEVELPROGRESS LEVELLOAD In SelectLevelScreen: INITNEWLEVELBUTTONS NEWLEVELDESCRIPTIONS DISPLAYNEWLEVELBUTTONS LEVELBUTTONCLICKLISTENERS

References

[1] "Geese Requirements" [Online] Available: https://github.com/mh1753/AbstractDelete/raw/master/Documentation/Assessment%203/Req3.pdf
[Accessed 17-Feb-2019]