2. 6 | Updates on Assessment 1 Deliverables

A. Requirements

Ref	Old Requirement	New Requirement	Description of Change	Explanation of Change
U5	10 different world locations	5 different world locations	We decided to change this requirement to 5 world map locations, still based around the University campus.	Ensure quality of the game whilst delivering it to the customer on time, and to simplify the game.
U6	Ability to encounter enemies throughout the world	Ability to encounter enemies throughout the world (except in the town)	This was slightly altered by making the town a safe zone in which the character could not engage in combat.	This change meant the town could be used to trade, and the player wouldn't be interrupted by unwanted mob fights. It also made more sense that the mobs were only in the dungeons.
S1	The user should be able to customise a character to some degree, in terms of appearance and skill.	The user should be able to customise a character's skill to some degree	This was changed by removing the ability to change the character's appearance, meaning only the skill could be customised.	We felt that using the market place in the game would provide the user with enough opportunity to customise their character in terms of weapons and armour, therefore appearance was less relevant.
S2	The system will have a GUI that will display the chosen character +companions (using side-scrolling format)	The system will have a GUI that will display the chosen character (using side-scrolling format)	We altered this requirement so that the GUI will not show companions while roaming around the world, they will only be displayed in combat.	We felt displaying companions at all times would cause the GUI to become congested, and meant more focus on where the character was going, meaning better gameplay.
S4	The system will have a viewable map that can be accessed anytime, showing icons that represent the progress of the 6 playable characters	The system will have a viewable map, showing an icon that represents the progress of the character group.	We decided to change this requirement to only viewing the world map when exiting a location. As the character's group of companions will be all together this can be represented as a single point on the map displaying location, rather than showing the location of 6 characters separately.	We felt viewing the map at the end would allow the player to always know where they were, and move on to the next location more easily. Showing the group as a single point on the map would also make the game more understandable, and again focused on the main character.
F4	Display a world map from the pause menu that displays progress	Display a world map upon exiting locations, to choose next destination.	We decided that the world map would be viewed upon exiting the town and dungeons, allowing the player to select and "fast travel" to their next location, rather than from the pause menu.	We felt this fulfilled the customer demands, and made the game more intuitive by allowing the player to follow the story logically. It also allowed constant tracking of the player's location.

B. Models, Methods and Tools

Methods

Our methods needed clarifying as to what approach we were following as a team. We are approaching the project using an agile process. This is because we felt splitting the work into teams, and working collaboratively to elicit and meet the user requirements naturally follows the agile process and is therefore ideal to our software engineering task, despite having a small time frame. Although we used some aspects from RAD, such as more focus on process rather than planning in our software development, our idea of customer collaboration by encouraging the client to specify requirements and follow them as closely as possible follows agile methodologies. Also communicating over the holiday period was achieved using facebook messenger and GitHub to communicate and collaborate as developers was supported by our agile approach.

Tools

Most of our tools remained the same during the second stage of development, where we made heavy reliance on Unity as a game engine, due to the experience with the software from one of our team members. We continued to use visual studio as an IDE to use and edit our code. One of the largest changes to the tools used was using GitHub in order to push code updates to our repository over the christmas holidays. We did this as it allowed version control, and ease of use for all members of the team due to familiarity, whilst also allowing compatibility with Visual Studio. This is why we felt this was a stronger choice than Visual Studio Team Services. Google Docs and Facebook messenger were also very useful, as they ensured ease of use and collaboration frequently, on all platforms used by our team members, to ensure we meet our requirements.

Team Organisation

One of the key changes we made to our team organisation during assessment 2 was splitting the group into pairs to work on coding the project, but also support the creation of documents and deliverables. We split the group into 3 groups, one pair working on the map implementation, another working on the combat system and character abilities, and the final group working on the shop and market town within the game. Doing so split the workload effectively, and meant pairs could communicate over christmas with more relevance and focus on specific tasks. We then all put our ideas together in regular updates in the facebook chat, as well as pushes to our team GitHub repository, and adding new documents to the google drive. This was done in order to maximise our personal strengths as a team, by pairing members of the team with high confidence in unity and C# to support other members. We felt this also fit in with our agile methodologies.

Updated Plan for Rest of SEPR

We have a full outline of our plan on our website (available on under updated methods). Our plan for Assessment 3 is also detailed in a Gantt chart. This details how the tasks in assessment 3 will be structured by ensuring a change report is completed to first understand our new team project, before continuing to implement the design. Once we have seen the new project we will be working on a plan to assign tasks to team members in much the same way as before, by using a pairing of those who are more confident in specific areas such as the game engine software or specific programming languages, with those who are less confident. This ensures a high quality project whilst splitting the workload equally and fairly.

C. Risk Assessment

For the most part, the risks of the project remained the same. Many of the previously identified low-risk problems occurred and were mitigated according to the prescribed mitigation techniques. We avoided any high hazard risks with proper group discussion prior to beginning work. For this reason we do not believe that our method of risk identification requires change as it has worked to our, and the project's, needs. The risk ranking system and prescribed mitigation technique meant we had a course of action to alleviate issues and prevent an accumulative slowing of project progress.

What needed changing was our approach to mitigation once a problem had been identified. What often happened during the project is one member would identify a problem that had occurred and notify everyone else. Thereafter, there was no organised delegation of the problem to a team member to be alleviated and so the issue could persist throughout development. Our new mitigation system includes an updated leadership system which places a member in charge of mitigation until the problem has been resolved. Appended mitigation decisions are found in red text in the old sections, they will have replaced any old methods.

The new system is as follows:

- Risk identified and a meeting called.
- Mitigation assigned to one person (either the person who found the issue or the one responsible according to the scope of the risk).
- At next weekly meeting all live risks are discussed to see if fixed.
- If not fixed, assign a second member to the problem to check first member's progress.

This new method is more efficient at fixing problems that are identified: by giving control of the situation to specific members, the time taken to resolve problems is greatly reduced as the need to discuss it number of times over several meetings is cut down to one meeting and a decision. While this could cause problems in dissatisfaction in the group as people resent the choices made by others over their work we feel this will not be a problem, firstly because we have already identified and mitigated that risk and secondly because each member is in charge of the choice an equal number of time so there is an element of fairness.

Some problems that occurred during the engineering process had not been identified throughout the planning stage. As such, we were not able to immediately resolve them. We have since updated (updates are highlighted in red) the risk assessment to include these risks and prescribe an agreed-upon sequence of mitigation to provide immediate support if they are to occur again.

These risks included:

- A member's deficiency with a language or technique
- Unsupported language by system
- Inconsistent Unity versions

These risks were found by problems that occured in the group, our meeting system meant they were discovered early and we were able to append them to the risk table to ensure it would not happen again.