3.2 | Change Report

3.2.a | Changes Summary

Taking on a project from another team (*Evil Geese*) brings unavoidable and yet necessary changes to our manner of conduct, and this could come to limit our productivity as we strive to bring the game project to completion. It is therefore important that we adapt our conduct to fit the new project and not the reverse. Trying to force the existing project into something more familiar to us will not only take considerably more time and effort but will affect the project negatively by creating a mismatch between how *Evil Geese*'s action plan covers the how the work that remains to be done will be completed, and how we would innately complete it.

Upon choosing and receiving the new project, we inspected the methodology, implementation and requirement documents as laid out by the previous team to give us an understanding of how they created the game so far, and how we may adapt our project management and development practices going forward. Using Software Configuration Management, we identified what changes needed to be managed within the documentation, code and deliverables.

The requirement document demanded foremost attention as we see this as the basis of all other documentation; that is, the purpose of other documentation is to detail the factors that bring the requirements to fruition. We found the *Evil Geese's* requirements both similar to our initial project but different in very subtle ways that necessitated change to the way we managed this part of the project. It was then necessary to mark off these requirements against what the team had already achieved with help from the report provided by *Evil Geese*.

Changes between *Evil Geese's* documentation and ours necessitate a change procedure. During our initial planning phase we individually went through the project to come up with planned changes (as detailed in the table below).

Area	Change Management Approach
Documentation	Skim Evil Geese's documentation looking for areas of explanation that will likely conflict between the policies we use and those of *Evil Geese*. We will first detail our findings (intent to change areas) in the change report, then use this as a guide to edit the documentation.
Code	Existing code is assumed to be correct and used in the existing implementation. Nevertheless, code will evaluated for its utility as we extend the game. If any code becomes deprecated or is found to be broken, it will be commented out rather than deleted in case dependencies are later found and the code can be recovered to a working state easily.

The deliverables from *Evil Geese* are all well-written and set out their aims in a clear manner, often with justification in light of client requirements. We will not change whole pieces of existing documentation to suite our needs because as previously discussed, this can cause more harm than good. What we need to do is understand how their methods fit their team and what makes ours different. These differences will lead to an understanding of areas that must be changed as we are a different team from *Evil Geese* and will not be able to work in the exact same way.

Overall, our approach to change management followed the basics of the change management process by requesting the changes seen in the table, and assessing how this would affect the project goals - in this case being the requirements supplied by Evil Geese. As a result of this, we rejected most major changes that involved deleting code or dramatically changing the approach used, as Evil Geese's methodologies suited our team, and allowed us to save time by adopting their methodologies immediately. We only carried out additive changes, by adding new artwork and additional code to implement the game, and by assigning each member of the team to one of these changes.

3.2.b | Change Explanation and Justification

i. GUI Report

Updated GUI Report: https://ojp508.github.io/Mambo-Studios/GUI3.pdf

Evil Geese's GUI report covers their design process, GUI usability and GUI playability. Their design proposes a GUI that satisfies their requirements , whilst avoids using an approach in which the program's design is fully completed and planned before starting implementation (BDFU). This method suited our team, as the game had already been largely implemented, and meant we could follow on the GUI without following any rigorous plans, meaning more freedom and creativity to be shown. Therefore we choose to keep this approach in extending the GUI.

We aimed to create a GUI that most closely matched with the one that already been implemented, therefore following their approach to usability, we used an art style that accounted for red/green colour blindness, as they had done. This ensured the GUI was consistent within the game, as well as ensuring usability for all players, which satisfied requirement UH2. In addition to this, we maintained a top down view for the game, also to ensure internal consistency, whilst meeting the requirements they had obtained from the stakeholder F7. We felt justified in our decision to keep these factors the same, as they matched our team dynamic by supporting an agile approach, as well as ensuring the game was kept usable and playable, ensuring an enjoyable user experience and customer satisfaction by meeting the requirements *Evil Geese* had produced.

The GUI for the combat mode in *Evil Geese*'s implementation was simple, usable, and suited the play style of the game. Therefore, we also decided to leave this unchanged, and simply use some of the artwork they provided to create combat scenarios containing other characters. This combat mode met all the relevant requirements (F8 and F1), and met the need to be playable, therefore by applying no changes we were able to have a consistent and complete implementation.

We have furthered the GUI of the game by implementing a GUI for the shop (of which the back-end code was designed by *Evil Geese*), which followed their existing principles to GUI design by not using Big Design Up Front, and prioritising usability and playability. This meant it used the same simplistic and accessible art style, and was implemented in Unity to ensure continuity within the game. Moreover, we have applied the existing main menu GUI to that of the newly-implemented minigame, adapting the text for applicability, further ensuring the game is consistent despite the team change over. This satisfies the criteria for assessment 3 by supplying a minigame, whilst also ensuring requirement F13 obtained by the previous team, was met. Also by adding new areas within the map we met the assessment 3 criteria of having 10 locations within the game, to complete its implementation, meaning the map GUI had additional areas to choose from, however the style and implementation remained the same.

Overall, since the handover, the game GUI has remained largely the same, with advancements in the map, as well as additions to the game such as a minigame and shop extending the original GUI presented by *Evil Geese*. We feel this is justified as it meets the requirements they obtained as a team, in addition to allowing the game to be playable and usable for all users.

ii. Testing

Updated Testing Report: https://ojp508.github.io/Mambo-Studios/Test3.pdf

Further links for testing can be found in Assessment 3 at: https://ojp508.github.io/Mambo-Studios/#

Evil Geese provided two testing documents; one that provided an overview - their testing philosophy and aims, again with intimate reference to game requirements - and another, that detailed results of unit and performance testing.

To ensure continued functionality of all the features *Evil Geese* had implemented, we decided that all tests that *Evil Geese* had created during their time with the project must return the same result both during our continued development of the project and in the final build release. This ensures none of the game's original functionality is lost during our development of it.

Our own testing focused on the new features we added/expanded. This involved testing:

- Every new item (checking they responded with the correct name and effect)

The Shop we added was tested through play testing, checking that dialog boxes ran correctly and that the correct items were given. The MiniGame was also tested through play testing as its simple code made unit tests meaningless.

Therefore, we will replace results (screenshots) in *Evil Geese*'s document that details testing results to prove the same tests are working after our development cycle. We will additionally provide new testing results to show that the added functionality is working we as intend. We will not change the testing philosophy and aims document, aside from minor continuity changes. We believe the information it provides (testing methods and purpose) remains just as relevant and applicable now, as it was during the previous development stage. Furthermore, their reference to requirements remains relevant (the referenced requirements have not changed).

iii. Methods & Plans

Updated Methods & Plans Report: https://ojp508.github.io/Mambo-Studios/Plan3.pdf
Updated Gantt Chart: https://ojp508.github.io/Mambo-Studios/Gantt3.pdf

Many of *Evil Geese*'s methods as detailed in their documentation are similar to the methods *Un-Two* used during assessment 2 and below. Therefore, the proportion of content that demands change represents a minority of that which is present in *Evil Geese*'s method documentation.

Further, many of the development and team organisational tools are identical; only the choice of communication tool differed between the two teams.

Evil Geese outline their intent to use Agile development and give rationale for doing so, namely how an agile development approach will deliver the best results given the prospect of changing requirements. Not only does Agile already correspond to the approach our team has become accustomed to, but the rationale remains relevant as we progress into stage 3 of the development cycle and hence the approach to development is unchanged.

Upon reading the management approach taken by Evil Geese we decided to adopt their approach of assigning each team member a management role for each area of the project. We felt this method was justified as it meant we would not need to change their plan, and we could produce a game that would be as consistent as possible by following their method and management procedure. This method was also useful when identifying risks for the risk assessment as we each had a more intimate understanding of specific areas (assigned areas) of the project and could identify potential risks easier. We plan to take this approach forward with us in to assessment 4 as it has proven to be effective.

iv. Risk Assessment & Mitigation

Updated Risk Assessment: https://ojp508.github.io/Mambo-Studios/Risk3.pdf

Evil Geese took a very similar approach to Risk and Mitigation analysis and this made our approach to risk fairly straightforward. *Evil Geese* used a tabular approach recording risks with an associated likelihood and probability. This straightforward approach is simple to understand and requires no change as our group already has a good understanding and experience with this approach.

Their system of Risk Ownership was slightly different. They associated risk completely to one owner who was in charge of identifying, managing and resolving a particular risk. While we agree with this approach of assigning ownership we feel resolution should not be assessed by the same person who is fixing the risk. We think it is better that an outside opinion is introduced to properly decide when the problem has been fixed. We will say the Risk Manager from *Evil Geese*'s set Methodology is in charge of deciding when risk is mitigated. In cases where the Risk Manger is the Risk Owner, the Group Manager will have the responsibility.

Evil Geese's presentation as stated is tabular and simple, a clear key is set out which explains each separate field to ensure its easy to understand, each field is both necessary and useful with a clear boundary of what it affects or describes. The triple fields of Description,Triggers,Preventative Measures makes a clear set and responsive risk process making the whole process run smoothly, for this reason we are not going to make changes to the set out as it is exemplary in its design and all group members are able to understand and implement there necessary processes.

The Risks themselves cover the majority of the project, while not having as many risks as the group previously used the ones they do have cover a much broader range of the project, this is an improvement over previous methods as it allows assignment of ownership to be far more simple. The owners are in charge of less risks so they can watch the ones they are in charge of more closely making identification easier and more likely.

However the Risk table currently has no plan in place for dealing with Risks due to change in the Project from team switching, this can't be allowed to continue. As discussed, the major debilitating factor for this section of the assessment is confusion due to changes between the teams' design. To mitigate this a new section in the Risk Table has been added the C section which identifies, assigns level and mitigates risks affecting the team due to change of group. This section will identify and solve the risk of confusion caused by changing methodology and engineering techniques.

The rest of the Risk assessment was left unchanged, due to the work provided by Evil Geese being satisfactory and well managed, as well as providing an effective Risk Management process if we kept up to date with a proper series of identification meetings as set out by their methodology.