Change Report 2

Cohort 1, Group 6

Group Members:

Hussain Alhabib

Ellen Matthews

Minnie Poon

Jason Ruan

Daniel Smith

Owen Smith

Process, Tools & Conventions Used

Due to the extensive documentation across this project, especially following the selection and takeover of Team 9's project for Assessment 2, we required detailed conventions and well-established tools to effectively track and monitor changes made to both the deliverable documentation and the project's code.

Building on the effective methods we used in Assessment 1 - including Slack for communication, the Google Workspace for file management, and a Gantt Chart for task management and timeline visualisation - we implemented additional methods to keep track of and manage changes between our documentation and code and Team 9s.

For the documentation

We optimised our Google Workspace drive to keep each team's documentation well organised but separate from each other and utilised its version history functionality to track edits throughout the project. To remain organised within individual pieces of documentation, we established an editing "convention" to document and highlight our changes:

1. **Small Changes:** For minor edits such as definition updates or rewordings, we replaced their original text with our edits, which were formatted in bold and used the Calibri font.
2. **Late-stage Changes:** For changes that required comparison, we retained their version and added our edits immediately after. To help differentiate, we included a dashed line before and after the edited section, titled it "OUR EDITS", and used the Calibri font.
3. **Major Changes:** For extensive edits i.e. replacing a whole page, we removed their version entirely and replaced it with ours. Like with late-stage changes, we marked these sections with dashed lines, titled them "OUR EDITS", and used the Calibri font for consistency.

For the code

We continued to use GitHub as our primary version control and development tool. To effectively track, review, and plan changes we used the following GitHub features:

1. **Commit History:** GitHub's commit history allowed us to keep track of each change made to the repository after we cloned Team 9's at the takeover stage. Furthermore, our practice of using detailed commit messages ensured transparency and accountability but also the traceability of modifications made to the code.
2. **Actions:** We utilised GitHub Actions' Pull Requests to review and approve changes as a group before integrating them into the main branch. This ensured the code met our quality standards - set and regularly reviewed by our Quality Assurance team member - and met the project's requirements.
3. **Issues:** During development, GitHub Issues were used to plan and assign tasks, track bugs, and document feature enhancements between group members. Each issue was tagged with labels including "area: game", "area: documentation", "cat: bug", and "cat: enhancement" to help easily categorise and distinguish between tasks.

These collective processes and tools helped us to efficiently and effectively plan, make, and track changes made to both the documentation and code within the team.

Updates to Team 9’s Requirements 1

Find the original Req1 at [https://uoy-team-six.github.io/a2/assets/a1-09/deliverables/Req1.pdf].
Find the updated Req1 at [https://uoy-team-six.github.io/a2/assets/docs/Req1-Updated.pdf].

Firstly, we didn’t change anything about how the initial analysis process and requirements collection was undergone. This is because, even though we had a different approach, we’re building off of their initial plan and therefore it would make no sense to replace that with our own process because we technically didn’t use that here. We did however change their Single Statement of Need (SSON) as this demonstrates how we viewed the project and helps explain additional requirements we added later. Luckily our SSON was already very similar to Team 9’s. Here, the only main difference between the two SSONs is the mention of the title, upgrading buildings as a feature, the conversion of gameplay-time to real-time and the success metrics.

Next, we made sure to add certain definitions which weren’t already mentioned or edit those we felt needed better explanations. We then changed all the priority listings. Previously, the different categories were: “Shall”, “May” and “Should”. We found these titles vague and unclear as to the hierarchy from highest to lowest, and so swapped the priority titles over to our original ones: “Essential”, “High”, “Medium” and “Low”. This also gives us the extra category of “Essential” which describes requirements that the game cannot function without, whereas “High”, “Medium” and “Low” being vocational requirements but with varying importance.

Changes to the User Requirements

- **UR_BASIC_BUILDINGS**: Title was changed from UR_BUILDING_VARIETY as the requirement also includes actions such as upgrading and demolishing, which we felt didn’t fit well under the original header.
- **UR_TIME**: Title changed from UR_GAME_PROGRESS as the countdown doesn’t necessarily indicate game progress, just the passage of time, as such the title was changed.
- **UR_SCORE**: Title was changed from UR_STUDENT_SATISFACTION as we wanted to implement various ways to progress in the game, not just student satisfaction. Therefore, the title didn’t suit as well anymore and was thus changed to a more broad idea. Specifically, the idea of university income/money being an idea we added alongside Team 9’s desire for some form of income being added to the game.
- **UR_EASE_OF_USE**: Title was changed from UR_INTUITION. Both titles have a very similar idea, but we preferred the term “Ease of use” as it implies a particular effort to make the game as accessible as possible, whereas “Intuition” implies that the user should have a knowledge of basic game structures beforehand, which we didn’t want to emphasise as anyone of any knowledge level should be able to play.
- **UR_BUILDING_LIMITS**: Title was changed from UR_CAMPUS_CREATION, as the requirement was more about restricting building placement than it was about making a campus.
- **UR_EVENTS**: Very similar but added specification that there was to be at least 3 core events.
- **UR_DEPLOYMENT**: Title was changed from UR_PERFORMANCE to improve viewer understanding.
- **UR_TIPS**: A new requirement added to introduce the idea of tips and guidance to help players.

- **UR_SETTINGS**: A new requirement added to introduce settings such as in-game sound volume etc.
- **UR_LEADERBOARD:** A new requirement added to introduce a leaderboard to the game.
- **UR_ACHIEVEMENTS:** A new requirement added to introduce unlockable achievements to the game.

Changes to the Functional Requirements

- **FR_BUIDLINGS**: A merging of two requirements, FR_BUILDINGS and FR_BUILDING_TYPES. We felt it would be easier to combine these requirements as the types of buildings available should be described alongside building implementation. We also added mention of the player being able to demolish buildings which was previously omitted and removed the idea of a time constraint on building buildings.
- **FR_EVENTS**: A merging of two requirements, FR_EVENT_TYPES and FR_EVENT_VARIETY. We felt it would be easier to group all event requirements together.
- **FR_USER_INTERFACE**: We included mention of a satisfaction metric being visible to the player at all time and interactive elements that react when clicked or moved.
- **FR_BUIDLING_EFFECTS**: Changed title from FR_SATISFACTION_BUILDING as we felt our title was more clear and explanatory.
- **FR_BACKGROUND**: Fixed error where FR_BACKGROUND was listed as a non-functional requirement rather than a functional requirement.
- **FR_SETTINGS**: A new requirement added to introduce settings as shown above.
- **FR_LEADERBOARD:** A new requirement added to introduce the leaderboard.
- **FR_ACHIEVEMENTS_SYSTEM:** A new requirement added to ensure achievements are unlocked in real time and that players are informed about it.
- **FR_ACHIEVEMENTS_EFFECT:** A new requirement added to ensure achievements affect the player’s final score.

Changes to the Non-Functional Requirements

- **NFR_PERFORMANCE**: Removed the idea of a specific frame rate and focused instead on the more abstract idea of smooth gameplay.
- **NFR_OPERABILITY**: Added the idea of the game being accessible to as wide an audience as possible and being accommodating to different needs.
- **NFR_LICENSE**: Added the requirement for the game to be appropriately licensed and use licensed sounds and music assets to improve the gameplay experience.
- **NFR_DEPLOYMENT**: Fixed error where NFR_DEPLOYMENT was listed as a functional requirement rather than a non-functional requirement.
- **NFR_LEADERBOARD_VISIBILITY:** A new requirement added to ensure the leaderboard can be accessed and is easy to understand.
- **NFR_ACHIEVEMENTS_NOTIFICATION:** A new requirement added to ensure players clearly know which achievements they have unlocked and when.

Changes to the Constraint Requirements

- **CR_LEGAL**: Removed as considered redundant.
- **CR_LOCAL**: Removed as considered redundant.

Updates to Team 9’s Architecture 1

Find the original Arch1 at [https://uoy-team-six.github.io/a2/assets/a1-09/deliverables/Arch1.pdf].
Find the updated Arch1 at [https://uoy-team-six.github.io/a2/assets/docs/Arch1-Updated.pdf].

Other than certain key diagrams, we did not change much of Team 9’s Architecture deliverable. For example, we kept their description of their architecture design process and of their architecture evolution as it is related to their progression across Assessment 1 and thus relevant as a baseline for our Assessment 2 architecture. Furthermore, as we wished to keep the architecture similar to theirs we followed the same architecture design process as them.

One thing of note, is that within this amended deliverable, we have slightly changed the editing “convention” by leaving their old architecture that we have redesigned as we believe it provides fundamental understanding as to the development of the game’s architecture across both assessments.

In addition to what is mentioned above, we also kept the latter part of the documentation the same too. This includes their sequence diagrams, and the platform they used to create their architecture designs. We kept these as we also used PlantUML for the creation of the new architecture, and we kept their sequence diagrams as we did not really change anything within the game related to how the user interacts with the map. However, the two things we did change can be found below.

1. A new Class Architecture Diagram

The most substantial amendment was the redesign of Team 9’s Class Architecture diagram. During development, the lead programmer renamed some of their classes and also implemented a few new parent classes to control functionalities. In addition, we created two new Screens to assist with the game’s functionalities; a Leaderboard Screen and a Help Screen - the classes of which can now be seen in the new class architecture diagram.

A more detailed description of some of the changes made to the class architecture can be found below:

- The addition of Events
    - Class: EventManager, GameEvent, EventType
    - This section allows the GameLogic class to process and display events on the game screen randomly to players throughout gameplay.
    - Related Requirements: UR_EVENTS, FR_EVENTS
- The addition of Achievements
    - Class: AchievementManager and Achievement
    - This section was a new functionality added to the product brief for Assessment 2, and interacts with the GameLogic class to monitor the players progress and give them achievements if the conditions are met.
    - Related Requirements: UR_ACHIEVEMENTS, FR_ACHIEVEMENTS_SYSTEM, FR_ACHIEVEMENTS_EFFECT, NFR_ACHIEVEMENTS_NOTIFICATION
- The addition of the Leaderboard
    - Class: Leaderboard

    - This section was another new functionality added to the product brief for Assessment 2, and interacts with the Score class to calculate the top 5 performing players of the game.
    - Related Requirements: UR_LEADERBOARD, FR_LEADERBOARD, NFR_LEADERBOARD_VISIBILITY
- The addition of Settings and Difficulty
    - Class: Settings and Difficulty
    - Although the SettingsScreen already existed, Team 9 never added any functionality to it. For assessment 2, we implemented the requirement for players to be able to select their desired level of difficulty; Easy, Normal or Hard. These classes help to do that. We also added sound assets thus making the already implemented volume slider functional.
    - Related Requirements: UR_SETTINGS, UR_DIFFICULTY_SETTINGS, FR_SETTINGS, FR_DIFFICULTY_SELECTION
- The addition of Animated Students
    - Class: PeopleManager and Person
    - These classes were added to ensure that we could introduce animated students to the background of the map to provide the player with a more engaging and enjoyable experience.
    - Related Requirements: FR_BACKGROUND

2. An updated Behavioural Diagram

We updated their Diagram 3 (Behavioural Diagram) to highlight the changes in game states from when we took it over to now. Key additions include; a new Leaderboard screen, a new Help screen, an updated Settings screen which now includes a toggle for debugging keys and a dropdown selector for difficulty levels, and an updated GameOver Screen which now displays the players score including a breakdown by metric, and the ability to add their name for a Leaderboard entry.

Updates to Team 9’s Method Selection and Planning 1

Find the original Plan1 at [https://uoy-team-six.github.io/a2/assets/a1-09/deliverables/Plan1.pdf].
Find the updated Plan1 at [https://uoy-team-six.github.io/a2/assets/docs/Plan1-Updated.pdf].

Aside from the obvious changes in Team Organisation, the majority of Team 9’s deliverable was similar to ours. This includes the fact that we both utilised GitHub and GitHub Projects for the coding aspect of the project, we both developed our game using IntelliJ and VSCode, and we both used LibGDX as our game development framework to supplement the coding aspect. A summary of the few changes we did make can be viewed below.

1. A refined selection of chosen methodology

Although we both selected Agile as our team’s software engineering methodology, we added an additional section to their Plan1 documentation to reflect both the feedback we received for Assessment 1 and the process we had actually been following. As a result, we added a short description about our evaluation and selection of the type of Agile development framework we wished to follow during this project; Scrum.

2. A different method of team communication

Unlike Team 9, we did not evaluate Slack and Discord, instead we compared Slack and Microsoft Teams. Therefore, we updated the deliverable to reflect our decision making process and to highlight our differences in opinion and chosen platform for Team Communication.

3. Our approach to team organisation

As each team’s approach will be different, we added a new section to detail our approach to team organisation as well as the current roles and responsibilities for each team member - similar to our process in Assessment 1. Although different, a few similarities to note, are the fact that we both decided to have a team member lead each main section, and assigned roles based on personal interests and skills.

4. Our systematic plan

This section was less a change, and more an addition to purely show our method of organising tasks and tracking progress was different. We decided to simply include a screenshot of our Gantt Chart and a brief description of the benefits of using our chosen design to enhance the deliverables accuracy.

Updates to Team 9’s Risk Assessment and Mitigation 1

Find the original Risk1 at [https://uoy-team-six.github.io/a2/assets/a1-09/deliverables/Risk1.pdf].
Find the updated Risk1 at [https://uoy-team-six.github.io/a2/assets/docs/Risk1-Updated.pdf].

Due to the short length and similarity between our Risk Documentation and Team 9’s Risk Deliverable, there were not many changes to make. However, for the changes we did make, see below a description of each.

To note, we did not change their Risk Management Process, as we felt their process closely mirrored ours with a few exceptions - that to the most part we have now adopted ourselves for the Risk Register in Assessment 2.

1. A refined risk register

After analysing Team 9’s Risk Register, and comparing their identified risks with ours, we decided to merge our tables by including some of theirs, and omitting parts of both. A summary of these changes can be found below, and a more detailed index of changes can be found on the next page;

- **6** of Team 09’s Risks not used.
- **4** of Team 09’s Risks used solely.
- **2** of Team 06’s Risks used solely.
- **6** of Team 06’s Risks and **4** of Team 09’s Risks merged.
- **1** newly created risk.

Furthermore, following the feedback we received from Assessment 1 we followed Team 9’s utilisation of colors and qualitative measures for each of the risks, and also implemented a Risk Type column. In addition, we utilised their method of distinguishing Avoidance, Mitigation, and Contingency Plans for each risk due to the fact we felt it improved the traceability and readability of the Risk Register.

2. A new risk

After completing the comparison process of both our risks and Team 9’s risk, and following our feedback for Assessment 1, we decided to create a new risk. R11 is a brand new risk that focuses on game enjoyment/playability - something mentioned in the feedback for Assessment 1 that we did not do. The risk ensures that we prioritise testing throughout the game’s development to ensure maximum user satisfaction at the final product.

Risk Register Changes Index

In the table below, you can see an index denoting each of our Risks and Team 09's risks and a comparison as to what we did with them i.e. kept them, did not use them, or merged them with another risk.

New ID	Old ID [Team|ID]		Kept?	Reasoning
R1	Team 06	1	Merged	These risks were merged as both risks addressed the issue of team member availability. And by combining them, we have created a more robust mitigation strategy that focuses on the bus factor of certain tasks.
	Team 09	R1		
R2	Team 06	2	Yes	We kept this risk as it is crucial to mitigate the loss of documentation which could be devastating to the group's progress and result in us missing the deadline.
-	Team 09	R2	No	We felt this risk was redundant since all the libraries we chose are well-established, reliable, and do not require further research.
R3	Team 06	3	Merged	We merged these risks as the risks addressed different avenues related to communication issues within the team. This allowed us to streamline the mitigation approach by focusing on our Team's methods inc. Slack and the Gantt Chart.
	Team 09	R11		
	Team 06	5		
	Team 06	7		
R4	Team 06	4	Merged	We merged both these risks as they addressed challenges with unfamiliar or overcomplicated tools. This merged risk focuses on group discussions and team training.
	Team 09	R9		
R5	Team 06	6	Yes	We retained this risk as software policy changes are extremely relevant and can significantly affect the project's progress if it occurs.
R6	Team 09	R6	Yes	We adopted this risk as we believe it is possible that customer requirements could change, and we also utilise an agile methodology - relevant to mitigating this risk.
-	Team 09	R12	No	We felt this risk was not relevant as our project team communicates regularly, and the Gantt Chart easily ensures task deadlines are clear and followed clearly.
R7	Team 06	8	Merged	These risks were merged as both risks dealt with team members' skills and their alignment to project roles - something we focus on as a team.
	Team 09	R7		
R8	Team 09	R5	Yes	We kept this risk as unclear or ambiguous are a possible risk especially when our requirements follow a more textual format.

-	Team 09	R14	No	We felt this risk was redundant as the role of one of our group members as Report Editor is to go through and check all documentation is properly formatted.
-	Team 09	R3	No	We felt this risk was not relevant as our team has clear processes in place to maintain code and readability - something monitored by our Head Developer.
R9	Team 09	R4	Yes	We retained this risk as compatibility with hardware and OSs is a valid concern - in fact it is stated in our requirements.
R10	Team 09	R8	Yes	We adopted this risk as time management and meeting deadlines are critical - especially when being assessed - and is easily tracked using our Gantt Chart.
-	Team 09	R10	No	We felt this risk was not needed as our Gantt Chart is designed flexibly, based on the team's ongoing progress, and thus does not require fixed time buffers.
-	Team 09	R13	No	We felt this risk was not relevant as our clear project schedule and focus on meeting requirements prevent scope creep from delaying important tasks.
R11	n/a	n/a	New	This risk is newly created. See the Change Report for our reasoning.

To compare further, see the original Risk Register

- of Team 06: https://uoy-team-six.github.io/assets/Risk1.pdf
- of Team 09: https://uoy-team-six.github.io/a2/assets/a1-09/deliverables/Risk1.pdf