CHANGE REPORT

Group 3

Liam Martin

Aaliya Williams

Lucy Crabtree

Kai Nichol

Sammy Hori

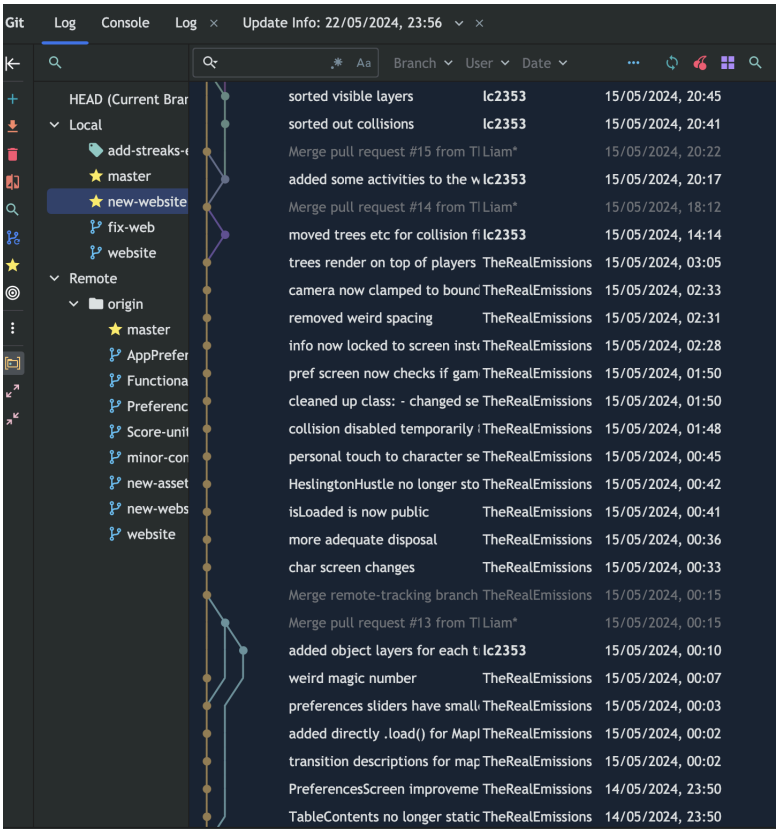
Tim Gorst

Zac Ribbins

Introduction

To begin with, the documentation of the game we took over was originally in PDF format, and so for the sake of updating their preexisting documents, we emailed them in order to obtain the Google Doc for each deliverable. Having the documentation in this form also enabled us to directly monitor each change that had been made to the deliverables via the built-in Version History function Google Docs contained, thus acting as an auxiliary in case an individual potentially forgot to document the changes made via the two methods outlined later.

In order to aggregate the alterations made to the previous deliverables, our team opted to use two distinct methods: a Trello Board and a Google Doc. The Google Doc contained subheadings for each deliverable, in which our team would document the date and nature of the change being made, which in turn, prevented the rise of potential miscommunication or inconsistencies when creating the change report. Simultaneously, the use of the Trello Board allowed us to keep track of what things needed changing as and when we discovered them and allowed us to sort these various tasks into their respective stages of progress. The Board was set up to notify the team via Discord every time an update or change was made, in a similar way to how we set up notifications for GitHub pull requests, ultimately making the tracking and review process straightforward.



Github change log, detailed commits

Requirements

After having received all of Group 1's original Requirements documentation, ensuring that the pre-established requirements were met was essential as that would set a precedent for how our team went about making changes to their pre-existing work. To begin, we opted to make minor adjustments to their introductory paragraph for a more precise and detailed rundown as to how the development process was initialised. Following this, a detailed explanation of the approaches and methods used for the establishment and formatting of the requirements was added to the documentation as this was missing initially, a notable change was the decision to use the MoSCoW method as this was deemed to be a better fit for the development of the game, especially after considering the extent in which changes needed to be made. This meant changing their priorities from 'Shall' to 'Must'.

Once these changes had been implemented, the most notable change made to the requirements table besides the addition of new requirements detailed later on, was the removal of the two Constraint Requirements, [TECH_CONSTRAINT_CODE_LANG] and [TECH_CONSTRAINT_JAVA_VER] changed to [CR_DESIGN_TECH] to remain consistent with the MoSCoW method.

The product brief for the second assessment demanded that we add new components to the game, a leaderboard and an achievements system to the game, this in turn, required us to add new user and functional requirements: [UR_LEADERSHIP], [UR_ACHIEVEMENTS], [FR_LEADERBOARD], [FR_ACTIVITY_COUNTER] and [FR_ACTIVITY_STREAK]. These new requirements saw that the game would include all components specified in the product brief, as well as acting as a guideline for future changes and additions to the architecture of the system.

Architecture

It was apparent once obtaining Group 1's Architecture documentation, that Group 1 had not met certain aspects of the assessment brief, and thus it was apparent many alterations would be necessary.

The first course of action we undertook was to justify the usage of UML and PlantUML as this was not present to begin with. The reason for this change was to make the document in line with industry-standard architectural documentation. Another notable change was the removal of their 'Class Responsibility Collaborator (CRC) Cards, which were transferred to the website for ease of access in conjunction with increasing the readability of the document. A few lesser changes were made, including the introduction of headers for each individual section of the document and the inclusion of a brief description of how the architecture further evolved after our team had inherited Group 1's work.

We then changed the Class Diagram and accompanying description to update it according to the changes we had made to their class architecture. Namely, we added an achievement class, and we removed some of the screen classes that we felt made the diagram cluttered, had already been mentioned in the HestlingtonHustle Class, and didn't add anything new.

Among the other changes detailed above, a major stylistic alteration was made, with the purpose of improving the layout and overall appearance of the document, this being in the introduction of a table. Initially, the classes were presented as sentences, containing a brief description of the class and how it linked to the requirements previously outlined. However given the number of classes and requirements needed to be included, a tabular format was by far the most desirable.

Once these issues had been addressed, we made changes to the the justification of their system architecture as it was apparent they did not implement an ECS style architecture despite saying that they did, and therefore, this section of the documentation was removed, and replaced with a revised statement justifying the usage of the OOP approach in order to reflect the code accurately.

Lastly, as the project had evolved drastically from the state in which Group 1 had left it in, it was deemed appropriate to document some of the noteworthy parts of this evolution. The 'Evolution of the System' section has been updated, now containing a description of a couple of quality-of-life changes made to particular classes and subsequently the architecture of the system as a means of simplifying certain

components. Changes made to the architecture with the intention of meeting the new requirements established were briefly detailed in this section.

Overall, the architecture document received the most changes in comparison with the other documents, given the extent to which alterations were made to the system itself.

Method Selection and Planning

We received Group 1's Method Selection and Planning Document and got to work updating it for our specific workflows. To our surprise, we found many similarities in how we worked, such as their development tools: LibGDX, Tiled and IntelliJ, which were identical to our own.

The first thing that we changed was their software engineering methodology section, as they discussed at length their use of both Waterfall and Agile methodologies, whereas we used exclusively Agile, so we updated the references to waterfall throughout the document.

Next was the Collaboration tools, where I found that they used similar applications to us, but on the whole fewer. Where they used Jira for kanban-style list making, we used Trello, and their communication methods were limited to WhatsApp, whereas we also opted for Discord in order to separate general communications (Whatsapp) from more technical discussions (Discord). Discord also allowed us to set up automatic messages for GitHub and Trello updates. Other than these changes, however, our collaboration tools were much the same as theirs, with PlantUML and Google Drive for organisation, and GitHub for version control.

In the discussion of their planning timeline, they included images of a number of UML Diagrams, but we felt they were too small to be legible, so we removed them and included links to the images that have been put on our website instead.
We didn't need to change much else on their Method Selection and Planning document, other than updating the planning sections with the plans for Weeks 7-14, and generally updating it for the second part of the assessment.

Our updated version of their Method and Selection Planning Document can be found on our website, with all new additions to the document made in red.

Risk Assessment and Mitigation

Once we had taken over the production of their system, when considering Group 1's Risk Assessment and Mitigation document, the majority of the pre-established risks still held some relevance however, it was apparent that some of these risks were no longer relevant to the developmental process of the system. One risk, in particular, R5, specified that one team member had to commute from outside of York which may have posed an issue if said member had not turned up to their meeting and thus, missed key details of the development process. Our team did not face this same dilemma and ultimately, this risk was promptly removed.

As the assessment involved making several changes to an already pre-existing system, a few potential risks were promptly brought to our attention and so needed to be strictly identified. As their system came with a fair amount of issues, of differing severity, risks R11 through to R15 were swiftly added to the document, outlining their type, alongside a brief description, likelihood, severity, means of mitigation and the team members who had been assigned the task of handling said risk.

To summarise, not many changes were made to this particular piece of documentation as the majority of pre-established risks were still relevant to our team and the overall development of the game.

Conclusion

Overall, the original deliverables received numerous changes over the course of our project in order to reflect the changes made to the system itself, these being highlighted in the 'requirements' and 'architecture' sections in conjunction with the changes in developmental procedures in the aforementioned 'method selection and planning' and 'risk assessment and mitigation' sections. For consistency purposes, all changes made within these documents are in red font to contrast the original black text.