Change Report

Cohort 1 Group 10

Cai Hughes<cabh500@york.ac.uk>
Ben Slater<bs1463@york.ac.uk>
Adeola Adeniji<aa3098@york.ac.uk>
Mathew Riedy<mr1723@york.ac.uk>
Riad Kasmi<rmk526@york.ac.uk>
Simon Konieczny<sk2144@york.ac.uk>

link to T11 website: https://engteam11.github.io/

Introduction:
Summary of Planning, Tracking, and Review Processes for Assessment 1

Our team employed a structured approach to plan, develop, and review changes to the Assessment 1 deliverables, including documentation and code inherited from the other team. The following processes, tools, and conventions were utilized:

1. **Initial Assessment and Documentation Review:**
    - We began by conducting a thorough assessment of the inherited deliverables, including codebase, documentation, and project requirements.
    - Detailed documentation review sessions were conducted to understand the existing architecture, design decisions, and implementation details.
2. **Establishment of Communication Channels:**
    - Effective communication channels, such as Slack channels and regular team meetings, were established to facilitate collaboration and information exchange among team members.
    - Clear communication protocols were defined to ensure timely updates, address concerns, and coordinate tasks.
3. **Version Control and Code Management:**
    - Git version control system, coupled with the platform GitHub, was used to manage code changes, track revisions, and facilitate collaborative development.
    - Branching strategies, such as feature branches and pull requests, were implemented to organise development efforts and ensure code integrity.
4. **Task Management and Tracking:**
    - Tasks and action items were organised using project management tools like google drive etc.
    - Tasks were assigned, prioritised, and tracked throughout the development cycle to monitor progress and ensure alignment with project milestones.
5. **Code Review and Quality Assurance:**
    - Regular code review sessions were conducted to assess the quality, readability, and adherence to coding standards of the inherited codebase and new contributions.
    - Code review feedback was provided constructively, focusing on identifying areas for improvement and ensuring consistency across the codebase.
6. **Documentation Updates and Maintenance:**
    - Documentation, including design documents, user manuals, and technical specifications, was updated iteratively to reflect changes and enhancements made to the system.
    - Emphasis was placed on maintaining clear, concise, and up-to-date documentation to support future development and facilitate knowledge transfer.
7. **Testing and Validation:**
    - Comprehensive testing procedures, including unit tests, integration tests, and user acceptance testing, were conducted to validate the functionality, performance, and usability of the system.
    - Test cases were designed, executed, and documented to ensure robustness and reliability of the software.

By implementing these processes, tools, and conventions, our team successfully managed the planning, development, and review of changes to the Assessment 1 deliverables, fostering collaboration, quality assurance, and continuous improvement throughout the project lifecycle.

Requirements

Previous Document
Updated Document

The existing Requirements document needed several adjustments and additions to better address previous and new requirements gathered by the team as well as specified in the Product Brief.

In the User Requirements Table, several of the existing requirements were moved to a 'SHALL' priority. These were: UR_MOVEMENT, UR_INTERACTION, and UR_WINNING_CRITERIA. These requirements result directly from the Product Brief and form the core aspects of the game. UR_MOVEMENT specifies that 'The user can move around the map', this is an essential feature of the game and therefore necessitates the 'SHALL' priority. Similarly, UR_INTERACTION refers to the players ability to interact with map elements, this is a feature without which the game wouldn't be playable (for example, lack of ability to interact with objects to sleep/relax/study would break the game and interfere with the user experience) so also requires the highest level of priority. Lastly, UR_WINNING_CRITERIA dictates the requirements for the end of the game, also requiring 'SHALL' priority to preserve the idea of the game from the Product Brief.

Two new User Requirements were added to address the new requirements specified in the updated product brief, these were: UR_VIEW_LEADERBOARD, which specifies that the user should be able to bring up the leaderboard for the game on a given computer and see the top 10 scores and the names of the corresponding players, and UR_HIDDEN_ACHIEVEMENTS, which describes the new streak mechanic, where doing a certain activity for multiple days in a row scores more points and unlocks achievements. Both of these requirements were assigned 'SHALL' priority to correspond to their critical nature in fulfilling the requirements set in the brief.

Moving on to the Functional Requirements Table, the title was changed to avoid the repetition of the word 'Requirements'. Several requirement descriptions were modified to more closely match the team's vision of how the game should match the brief. FR_MAP was modified to state 'Map of all required buildings for player activities' so that the requirement addressed the game better. Several other requirements were modified to be more descriptive, for example FR_CLOCK's description previously stated 'Display in-game time', this was changed to 'Keep track of game time, making sure each day isn't longer than 2-2.5 minutes. Display the current time for the player to see.'. Several other requirements were modified to add clarity to the requirements and make them easier to understand.

Additionally, two new Functional Requirements were added to correspond to the new User Requirements resulting from the updated brief, FR_LEADERBOARD was added, formalizing the requirement of storing player scores to form the leaderboard as a functional requirement of the game. FR_ACHIEVEMETNS was added to correspond to UR_HIDDEN_ACHIEVEMENTS that specifies that activities performed should be stored to keep track of repetition for the purpose of establishing streaks, and then scoring them appropriately. A third functional requirement, FR_STREAK_BONUS was added describing the point system for streaks.

Following the modification of the FR Table, the NFR Table was addressed. The fit criteria column was heavily modified since the the team's goal is fully meeting the requirements, therefore criteria such as '99% of users should not face any data leaks' was changed to 'users should not face any data leaks', eliminating the somewhat inconsequential 99%, since the testing of the game will not be robust enough to measure factors to such precision. This was a recurring theme in the Fit Criteria column that was fixed alongside minor changes in the descriptions of the NFRs.

Architecture

Previous Document
Updated Document

The Architecture document required several updates to be made due to both incorrect details and the updated user requirements.

Additional information regarding the use of the tools used to create architectural diagrams (UML and PlantUML) were added, with justification given for why they were used.

The class diagram needed to be updated to reflect the additional functionality added to meet the client's updated requirements. A new class was created in order to implement the leaderboard functionality and the StatsTracker class received a few changes to allow the tracking of streaks for the additional requirements. The Project and MainMenu classes were removed from the diagram as no class with this name or specific set of attributes and methods existed in the project files we inherited, and no other mention of this class or its purpose could be found in the architecture documentation. The DesktopLauncher class was also removed as it relates more to the required functionality of LibGDX framework rather than our own architecture. The method canPlayerMove() was also removed from the GameMap class on the diagram as the implementation did not contain this method.

Additionally the inheritance arrows showing the inheritance relationship between the subclasses MenuScreen, SettingScreen, EndScreen, inGameMenu and the parent class Screen were reversed, as initially they were facing the wrong direction and showing an incorrect relationship.

The relationships between all classes were updated to better illustrate the interactions among them. Previously all classes were depicted as being solely contained by the game class, which was not entirely true.

Three sequence diagrams were added to show some of the expected behaviour for certain components of the game (movement, interacting with activities and displaying the leaderboard) as the previous architecture document did not contain any behavioural diagrams and only lightly touched on the behaviour of certain components in relation to the requirements they fulfilled.

Method Selection and Planning

Previous Document
Updated Document

Both teams had a lot in common when it comes to how they went about cooperation and collaboration. While both teams used scrums as the methodology of choice, the approach to a few other aspects of collaboration differed. In particular, the current team all worked using IntelliJ as the IDE of choice, to streamline the workflow for everyone, if a team member had an issue getting something to run it was very easy to help them as everyone was working on the same platform. Other than that one change, all of the remaining aspects of methodology were the same across both teams, so were simply touched up on to clarify certain points.

Larger differences appeared in the Team Organization section, where the entire section had to be changed since the approach of the current team is much more individualistic and self-organized, which has proven to work well during the first part of the Assessment. This lead to several follow-through changes, the team was more divided along the lines of deliverables, where people stuck to a certain aspect of the process, and more democratic in their approach, no leader was necessary since everyone was very organized and held themselves accountable.

Lastly, the end of the planning section was updated with the work of the current team on Part 2 of the Assessment and a final Gantt chart was added for the days leading up to project submission.

Risk Assessment and Mitigation

Previous Document
Updated Document

In the Risk Assessment and Mitigation document, the introduction needed to be rewritten to reflect the approach used by the team in assessing and assigning risk ownership. Since the team follows a very clear four-step process of risk management, that was described in the introduction, replacing the previous team's methodology.

The Risk Register contained risks previously identified by the team, ownership of them had to be transferred to appropriate members of the current team. One risk (ID1) was removed as it referred to a previous consideration of the initial team, not applicable to the current team. Additionally new risks were identified and introduced to the risk register more in line with the thinking of team members. For example, scope creep, hardware/software failures, collaboration issues, poor user feedback, and issues with scaling the game were added to the risk register. These were risks identified by the team members as described in the methodology section and were seen as significant, with high or moderately high danger to the project.