

EcoVenture

Software Engineering: Group Deliverable

Group 15

Aisling Broder-Rodgers

Andrew Morrison

Joel Egerton

Leonie Robinson

Nathaniel Nimmock

CSC7083 Peer Assessment: Save Our Planet

This Assessment Document is intended to provide you and your assessor with an overview of each group member's involvement delivery of the CSC7083 Project.

Each group should complete one Assessment Document and its content must be agreed by all group members. The completed form should be included at the start of your group's PDF report. **Don't forget to fill in the Group Number.**

There are three main parts to the Assessment Document – the Evaluation, the Declaration and the Personal Statements (**spaces for each team member's personal statement are provided on the reverse of this sheet**). All parts must be completed – otherwise your group's report will not be marked. Arrange a group meeting to discuss the evaluation and personal statements, and see the note below!

Evaluation Group Number: 15				
Name	Contribution to team-working and motivation ¹	Contribution to documented analysis, design and testing ^{1,2}	Contribution to working system code ^{1,2}	Peer Score (Range 85 – 115)
Aisling Broder-Rodgers	5	4	5	114
Andrew Morrison	5	5	5	115
Leonie Robinson	5	5	4	110
Joel Egerton	5	5	4	111
Nathaniel Nimmock	5	4	4	109

¹Values for contribution: 1 = Minimal Contribution; 2 = Reasonable Contribution; 3 = Good Contribution; 4 = Very Good Contribution; 5 = Excellent Contribution

²This value should consider contributions in the round – direct contributions to required deliverables, and contributions that have made the deliverables possible.

Declaration		
<p>"I declare that I have read the Queen's University regulations on plagiarism, and that any contribution I have made to the attached submission is my own original work, except for any elements that I have clearly attributed to third parties. I understand that this submission will be subject to an electronic test for plagiarism and will also be subject to the University's regulations concerning late submission if it is received after the deadline."</p>		
Name	Date	Confirmation (<i>use the words shown in the example below!</i>)
Aisling Broder-Rodgers	21/4/24	I agree to the terms of the declaration
Andrew Morrison	21/4/24	I agree to the terms of the declaration
Leonie Robinson	21/4/24	I agree to the terms of the declaration
Joel Egerton	21/4/24	I agree to the terms of the declaration
Nathaniel Nimmock	21/4/24	I agree to the terms of the declaration

A note on the Evaluation:

Complete all the columns in the Evaluation Table. The Contribution columns are intended to help team members quantify each other's input to the project, before they award agreed **Peer Scores**. There will not necessarily be a precise correlation between the Peer Score and the Contribution values. However, high Contribution values, as an indicator of the importance of the team member's work to the success of the project, should normally result in a high Peer Score for a team member. Likewise a low Peer Score would be

the expected outcome if Contribution values are low. Students who have made a high-value Contribution in all three contribution categories (e.g. 5,5,5) should expect to receive a higher Peer Score than students who have made a lower-value Contribution in one or more categories (e.g. 5,5,3).

If, having reviewed the Contribution values, the team agrees that Team Member 1 made a minimal contribution overall, a Peer Score of 85 would be appropriate for Team Member 1. If Team Member 1's contribution was excellent (critical to the success of the project in all areas of engagement), consider a peer score of 115. If Team Member 1 made a generally good contribution, doing what was expected of them, they could expect to receive a Peer Score of 100. It may be that a team member (for whatever reason) has disengaged from the project entirely, and in such circumstances a Peer Mark of 0 may be acceptable. ***Please inform the module Lecturer if a team member has left your group or has ceased to play an active role in the group.***

Each team member's overall score for the project will be calculated according to the following formula, where S_i is Team Member i 's overall score, P_i is the Peer Score received by Team Member i , N is the number of members in the team, and M is the raw mark awarded to the report by the assessor.

Any Peer Score within the range 85 – 115 will normally be accepted by the module Lecturer. **However, students are expected to award a range of marks within a team: it is very unusual in a project for everyone to display exactly the same level of ability and commitment, and the Peer Scores should reflect this.** Be fair: be prepared to recognise someone who has adopted a leading role in the project, and acknowledge the fact that some contributions will be weaker than others. Uniform marks, or marks outside the range 85 – 115, may require that the Team discuss its decision with the module Lecturer, in order to agree a fair distribution of marks. Throughout the project, team members should use appropriately named folders in [GitLab](#) to help them co-ordinate their work and maintain a record of their contributions. Where team members cannot agree a distribution, or the distribution is unreasonable, the module Lecturer's judgement will be final.

<i>Personal statement of (enter name):</i>	<i>Aisling Broder-Rodgers</i>
The following were my most significant contributions to the project (100 words or less):	
Active participation in group discussions and meetings Worked on the UML Use Case Descriptions in response to the UML Use Case Diagram Development of the ActionSquare class and the accompanying J Unit Test class Development of the SOPGame class (in collaboration with structure and methods provided by Andrew and gathering player information developed from Nathaniel) Carried out associated black-box testing Reviewing of teammates code and carrying out some integration testing Editing of group video for submission	

<i>Personal statement of (enter name):</i>	<i>Andrew Morrison</i>
The following were my most significant contributions to the project (100 words or less):	

contributed to the creation of user stories, acceptance criteria, user journeys and sprints. Modelled and designed several of the sequence diagrams. Created the basic class structure and UML, and the initial draft game logic. Coded the Board class and its associated configuration csv and its unit tests. Designed the logic for offering a development and wrote the associated methods in the SOPclass
(createFieldMap, printFieldMap, checkFieldStats, offerDevelopment, getFieldSquares, OfferMinorDevelopment, OfferMajorDevelopment and getUserInput). Carried out code reviews and made suggested edits. Contributed to the group report. Particularly the testing section.

<i>Personal statement of (enter name):</i>	<i>Leonie Robinson</i>
--	------------------------

The following were my most significant contributions to the project (100 words or less):
--

I actively participated in discussions to understand user needs, scenarios, and desired outcomes. I created a thematic framework of the game and produced a Use Case Diagram showing the range of actions within the game. I created a user story map based on user stories and priorities. I generated the Piece and Die classes and tested the code, suggesting changes where needed. I facilitated collaborative development using GitLab. I actively participated in group meetings, sprint planning sessions and sprint cycle reviews. I documented the project's requirements and design decisions, creating comprehensive and informative sections in our project report.

<i>Personal statement of (enter name):</i>	<i>Joel Egerton</i>
--	---------------------

The following were my most significant contributions to the project (100 words or less):
--

My most significant contributions were:
 Extensive testing of the SOPGame Class
 Creation of several Sequence Diagrams
 Generation of the Square Class and testing of the same
 Testing and proofreading the final working code, suggesting minor improvements where needed
 Helping to chair and arrange times for the group's meetings
 Assisting in the overall final report, in particular the agile iterations section.

<i>Personal statement of (enter name):</i>	<i>Nathaniel Nimmock</i>
--	--------------------------

The following were my most significant contributions to the project (100 words or less):
--

My most significant contributions were:
 Set up teams' channel and what's app group for communication.
 Vocal contributor to all meetings, helped drive positivity and good morale.
 Chaired several meetings, acted as scrum master.
 Recorded minutes for all meetings.
 Iteratively developed and fully tested the Player class.
 Developed the comparator for comparing knowledge points.
 Assisted with logic to record and validate players in the SOPGame class.
 Local integration testing after each iteration to hunt for bugs and opportunities, suggested changes, improvements, and additions where appropriate.
 Created a user gameplay guide, covering board, gameplay, interface, and rules

Introduction

EcoVenture is a virtual board game to be played via the console mode of the IDE platform, with a distinctive theme centred around saving our planet. Players take on the role of eco-entrepreneurs, using scientific knowledge to address environmental challenges by getting involved in their own environmental initiatives, or contributing to another player's initiative, and use this knowledge to progress in the game through minor and major developments with the aim of achieving sustainability.

The project was collaboratively developed by a team of five members. Each team member had access to a dedicated file repository on GitLab, facilitating seamless uploading, viewing, and reviewing of code. Team Meeting minutes and other essential documentation were stored on MS Teams, ensuring accessibility for all team members. The team committed to weekly and then twice-weekly meetings to monitor and discuss the progress of assigned tasks. Detailed minutes were recorded during these meetings (see Appendix 3). Backlog and sprints were recorded using Jira. The game was played via the console in Eclipse IDE and tested using JUnit framework.

The game was crafted according to the specific requirements outlined by the customer. The initial phase for the team involved gathering these requirements. User stories were developed to articulate requirements from the perspective of a customer (see Appendix 1). The user stories were refined and added to the product backlog to serve as a basis for sprint planning.

A use case diagram was created to provide an overview of the entire system, while the corresponding use case description detailed the sequence of actions required to attain a desired outcome. Additionally, class and sequence diagrams were generated to illustrate interactions between users and the game.

By the end of development, we had an end product which met all functional and non-functional requirements and had been extensively tested using white and black box testing techniques.

Requirements Engineering in Agile Software Development

In Agile software development, requirements engineering involves defining the features and scenarios of the system to meet the needs of the stakeholders. This process includes feature identification, scenario definitions and user story creation.

The team identified and documented the core features of the game based on the stakeholder's requirements. These were then split into functional and non-functional requirements. Functional requirements included functions the game should provide, and how the system should react to particular inputs and situations.

Functional requirements:

- Allow up to four players (minimum of 2), whose names should be unique and entered at the beginning of gameplay.
- The players take turns throwing two virtual dice.
- Provide information to players about the squares they land on, including obligations or opportunities.

- Allow players to take actions such as dedicating resources to take charge of a square or offering a square to another player.
- Display the resources (knowledge points, squares owned, minor and major developments) for each player when their resources have changed.
- Include a start square where players pick up resources as they pass.
- Implement a square where nothing happens (e.g., a neutral square).
- There are four 'fields', two consisting of three 'areas' and two consisting of two 'areas'.
- One of the two-area fields is the costliest field to acquire and resource.
- Another two-area field is the least costly field to acquire and resource.
- Display player positions and their ownership of squares and the properties of those squares.
- Players must own entire fields before they can use knowledge points to develop areas within those fields.
- Any area in a field owned by that player may be developed during that player's turn.
- Three minor developments are needed before a player can establish a major development.
- The more developed the area, the greater the resource consumed.
- Provide choices for a player to keep playing or leave the game.
- When a player leaves the game, the game ends.
- If a player runs out of knowledge points, the game ends.
- When a game ends, the amount of knowledge points each player holds are shown.
- The winner is the player with the highest number of knowledge points.
- Express the outcome of the game in a manner appropriate to the style of the game.

Whereas non-functional requirements primarily included system wide requirements rather than individual features.

Non-functional requirements:

- Ensure smooth gameplay experience with minimal lag or delay between player actions.
- Intuitive and user-friendly interface for easy navigation and interaction.
- Implement error handling and recovery mechanisms to prevent game crashes and data loss.
- Design the game to accommodate future updates and expansions, such as additional fields or areas.
- Balance the gameplay mechanics to ensure fair and enjoyable gameplay for all players.
- Use appropriate language and style consistent with the theme of environmental conservation and eco-friendly initiatives.

Scenarios were based on the functional requirements to capture the sequence each user takes within the game. These were later used as the basis for user stories and use cases.

1. Player registration
2. Turn-taking, roll dice
3. Square interaction
4. Action choice
5. Resource management
6. Field developments and ownership
7. End game

The scenarios were formulated into user stories to give an informal, general explanation of the features from the perspective of the end user. User stories are often used in agile software development to put the end users at the centre of the conversation. They also helped the development team understand why they are building, what they are building and its value. The user stories were as follows:

1. Player Name Entry at Gameplay Start
User Story: *As a player, I want to be able to enter the names of up to four players at the beginning of the gameplay, so that each player is identified uniquely throughout the game.*
2. Turn-Based Dice Rolling
User Story: *As a player, I want the game to implement turn-based mechanics where each player takes turns rolling two virtual dice, providing an element of chance and strategy to the gameplay.*
3. Square Interaction and Action Choice
User Story: *As a player, I want the game to inform me of my current position on the board, along with any obligations or opportunities associated with that square, allowing me to make informed decisions and take appropriate actions.*
4. Resource Change Notification and Balance Update
User Story: *As a player, I want the game system to notify me whenever there is a change in my resources, providing clear information about the reason for the change and updating my current balance accordingly.*
5. Start Square and Empty Square Functionality
User Story: *As a player, I want the game to include a Start Square where I can collect resources and an Empty Square where no action occurs, contributing to the overall gameplay experience.*
6. Field and Area Management in Save Our Planet
User Story: *As a player, I want the game to feature distinct fields representing various environmental sectors, each comprised of interconnected areas, allowing me to strategize and invest resources wisely to protect the planet.*
7. Area Development and Field Custodianship
User Story: *As a player, I want the game to enforce the rule that I must have ownership of an entire field before I can develop an area within it. Additionally, I want the flexibility to develop areas in fields I already own, even if I am not currently positioned on that specific area.*
8. Development and Major Development Establishment
User Story: *As a player, I want the game to feature a system of development, where I can invest resources to enhance specific aspects of the game world. Additionally, I want the ability to establish major developments, representing significant advancements or*

achievements, with the requirement of completing multiple minor developments beforehand.

9. Resource Exchange for Developed Areas

User Story: As a player, I want the game to incorporate a mechanism where landing on an area owned by another player incurs a cost, proportional to the level of development in that area. This ensures that interactions between players involve strategic decision-making and resource management.

10. Determining the Outcome and Purpose of the Game

User Story: As a player, I want the game to have a clear outcome when one player runs out of resources or decides to no longer participate. Additionally, I want the game to reflect the ethos of Save Our Planet, whether it emphasizes resource accumulation or cooperative support for worthy environmental ventures.

The user stories were developed further to also include tasks and acceptance criteria (see Appendix 1).

From the developed user stories, the team created a product backlog that allowed us to identify and prioritise system functionality. The backlog included all tasks that needed to be completed to ensure delivery of a successful, working system that met all functional and non-functional requirements. Requirements were prioritised based on their importance to the overall gameplay experience. Top priorities included core gameplay mechanics for a functional game, such as creation of Board, Die, Piece, Square and Player classes. Medium-priority requirements included additional features that enhance gameplay but are not critical, for example player actions and interactions. Meanwhile, some low-priority requirements included polishing the user interface and endgame conditions. We created a visual representation of the user stories, arranging them in a logical order and providing a high-level view of the game's functionality.

Figure 1 below shows the user story map which includes the epics, and corresponding user stories and tasks used in the backlog.

Figure 1 – User Story Map



The last step in requirement engineering involved creating a graphical representation of the virtual game, serving multiple purposes for our team. It offered a visual depiction of the game board layout, including squares and fields, which facilitated the team's understanding of the game's overall structure. This visual aid ensured that all team members were on the same page regarding the design and layout of the game board. With a visual representation available, everyone could easily reference and comprehend the game's features and components. Additionally, it helped ensure that the functional requirements of the board were fulfilled. For example, in figure 2, we can see there is a neutral square where nothing happens (*'Carbon Neutral'*). There are four 'fields', two consisting of three squares (*'Renewable Energy Initiative'* and *'Waste Management'* fields) and two consisting of two squares (*'Sustainable Agriculture'* and *'Climate Change'*). One of the two-area fields is the costliest field to acquire and resource (*'Climate Change'*) and another two-area field is the least costly field to acquire and resource (*'Sustainable Agriculture'*).

Figure 2 – Graphical game board

Hydro Power Renewable Energy Initiative Price: 95 KP Minor Dev: 25 KP Major Dev: 75 KP	Organic Farming Sustainable Agriculture Price: 50 KP Minor Dev: 45 KP Major Dev: 65 KP	Agroforestry Sustainable Agriculture Price: 60 KP Minor Dev: 55 KP Major Dev: 70 KP	<i>Carbon Neutral!</i>
Wind Power Renewable Energy Initiative Price: 90 KP Minor Dev: 20 KP Major Dev: 80 KP	<div style="text-align: center;"> <h2>EcoVenture</h2> <p><i>Invest in and develop environmentally friendly schemes to contribute to the sustainability of the planet!</i></p> </div>		Recycling Waste Management Price: 130 KP Minor Dev: 65 KP Major Dev: 100 KP
Solar Power Renewable Energy Initiative Price: 100 KP Minor Dev: 30 KP Major Dev: 90 KP			Composting Waste Management Price: 120 KP Minor Dev: 55 KP Major Dev: 95 KP
<i>The Green Beginning!</i> Collect 200 KP!	Climate Resilience Planning Climate Change Price: 275 KP Minor Dev: 120 KP Major Dev: 180 KP	Carbon Capture & Storage Climate Change Price: 250 KP Minor Dev: 100 KP Major Dev: 160 KP	Circular Economy Waste Management Price: 125 KP Minor Dev: 125 KP Major Dev: 107 KP

Figure 2 shows a graphical representation of the game board, which is included in the user guide. A full user guide, including the game rules, can be found in the appendices of this report (Appendix 4).

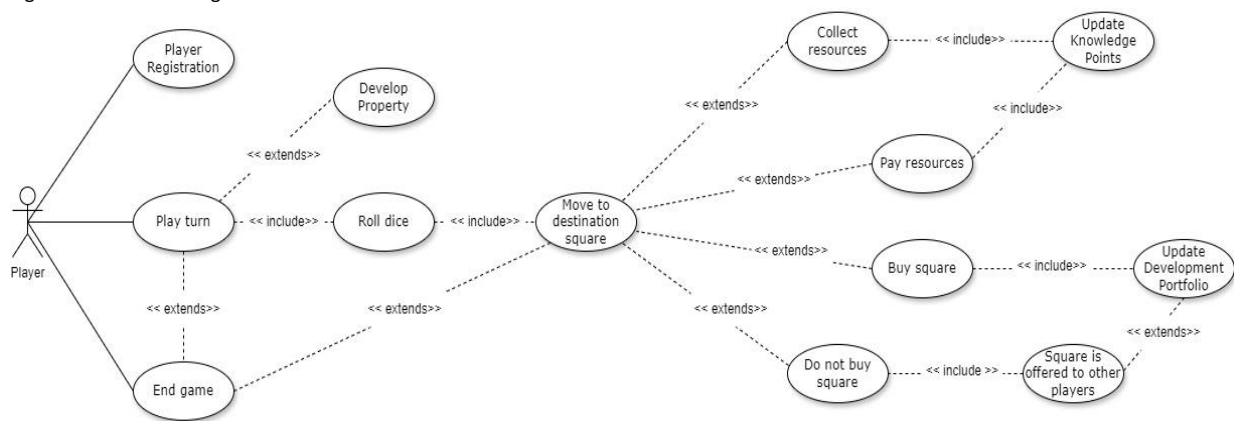
Designing

System modelling was utilized to enhance the team's comprehension of the system and ensure a shared understanding during development, facilitating the design process to align with user requirements. This included Use Case, Sequence and Class Diagrams as well as User Story Mappings.

Each Use Case describes a set of interactions between the actor and the system to achieve a particular goal. A Use Case Diagram was used to provide a visual representation of the interactions between actors and the system's functionality, helping the team understand the system's behaviour and requirements. Include relationships indicate that one use case includes another use case's functionality and can help avoid redundancy. Extend relationships indicate that one use case can extend another by adding optional behaviour, allowing for the modularisation of system behaviour. A description of each use case can be seen in Appendix 2.

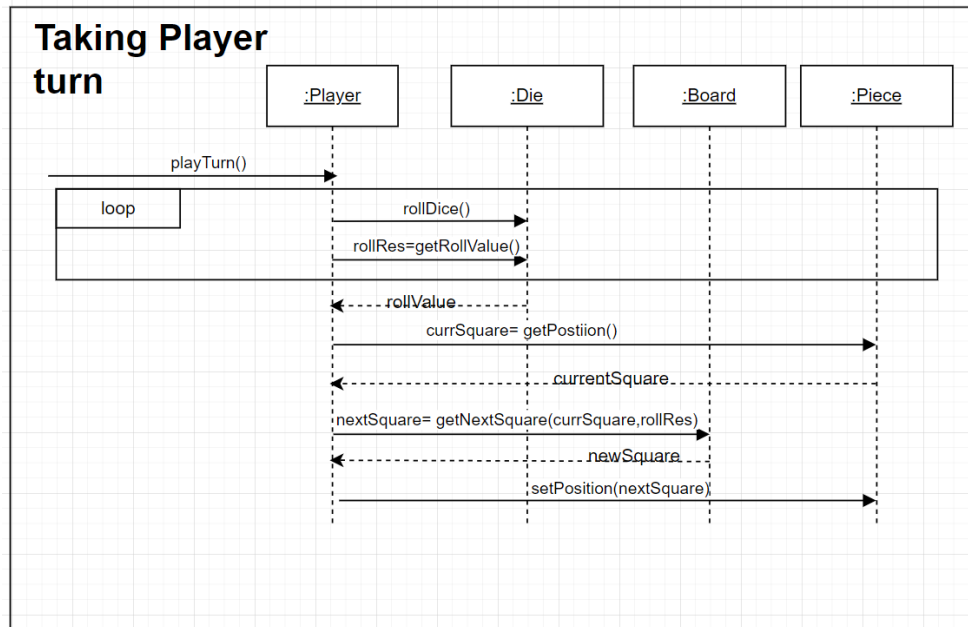
The Use Case Diagram in Figure 3 shows a high-level view of the range of actions within the game.

Figure 3 – Use Case Diagram



Drawing Sequence Diagrams, to illustrate the sequential flow of interactions between different components or objects within the game, was essential to the design phase. They helped us to understand how different components collaborated to achieve specific functionality and identify any areas for optimisation. We first identified key objects within the system, and how these objects interact with each other to achieve specific tasks. We then identified the messages or interactions between objects, indicating how they communicate and cooperate to achieve the desired functionality. Lifelines were utilized to represent the lifespan of each object during the sequence of interactions. Arrows were added to illustrate the flow of messages. Examples of these diagrams are shown in figures 4-9 below.

Figure 4 – Sequence Diagram: Taking Player Turn



In the sequence diagram above (Figure 4), the loop calls each player to take a turn. During turn the player rolls die, the roll value is stored in die object. Player calls `getRollValue()`, which returns the total roll, player then calls and stores `getPosition()` from piece to get the index of the current square. Piece returns the index of square. Player calls `getNextSquare()` from board to get and store index of new square on board based on old position index (`currSquare`) and roll value. Player calls `setPosition()` to store the new square index in piece object.

Figure 5 – Sequence Diagram: Enter Player Names

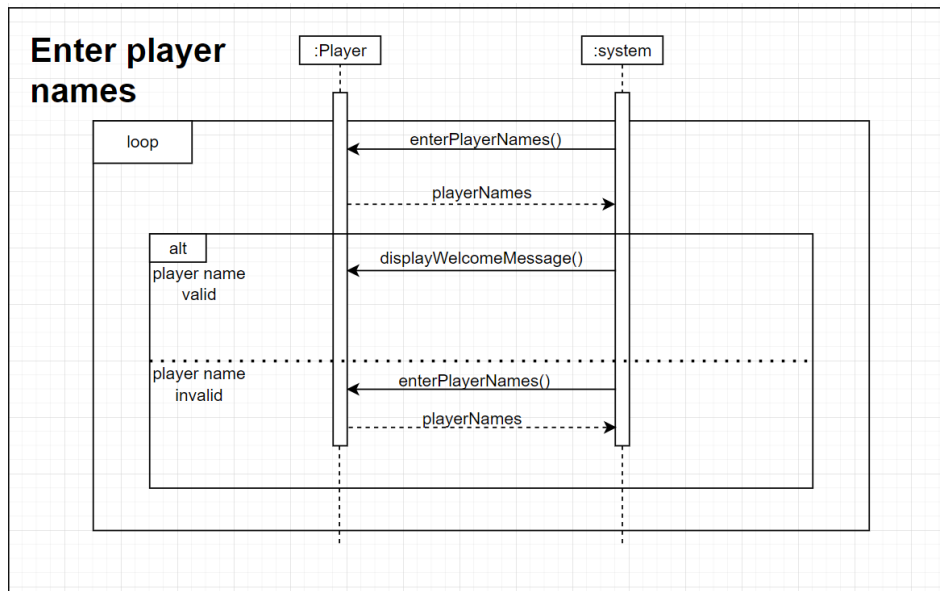


Figure 5 shows a sequence diagram for entering player names into the game for the number of players playing. The system prompts the player to enter player names and returns these. If the names are valid, players are shown a welcome message. If invalid, they are re-prompted to enter a valid name.

Figure 6 – Sequence Diagram: Game Main Loop

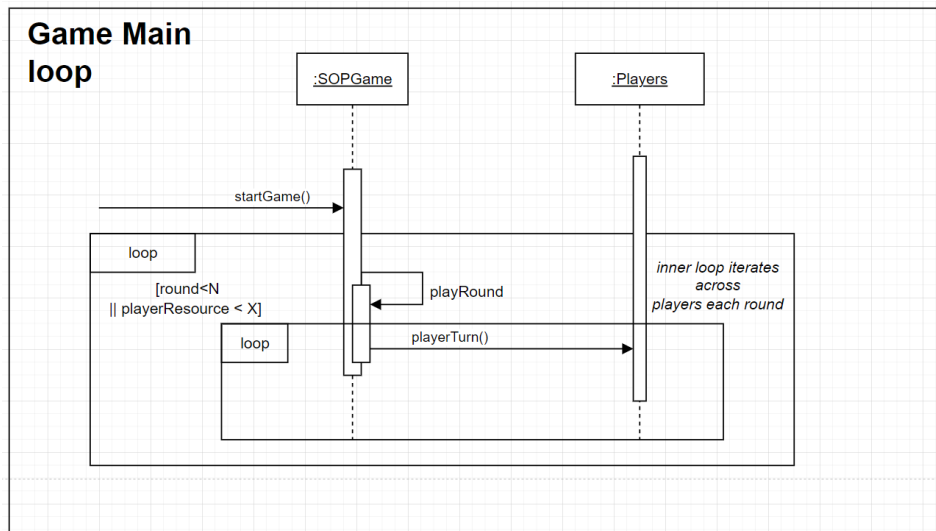
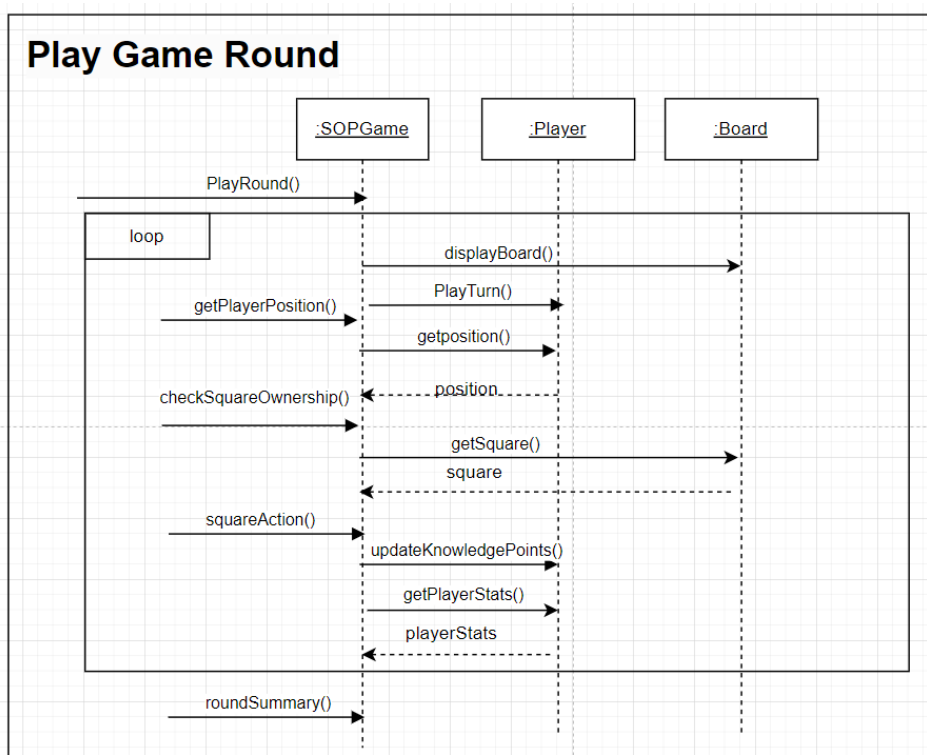


Figure 6 shows the game main loop. The game object will control the main loop and will iterate until N rounds hit or player resource exceeds X knowledge points. The game will enter a period of interaction where the game will call `playRound()` and then enter an inner loop where the game will call `playerTurn()` iterating across each player for each round.

Figure 7 – Sequence Diagram: Play Game Round



The play game round sequence diagram (Figure 7) shows the outer game loop calling `playRound()`, that iterates across players and calls `playTurn()`. The player moves to new square and then outer loop `getPosition()` checks ownership stats/ developments of current square, `squareAction()` is called that will have conditionals to offer the correct response (buy square, pay rent, add development offer to

other player) call update KnowledgePoints for relevant player and then get the player summary for each player.

Figure 8 – Sequence Diagram: Dice Roll and Developing Owned Square

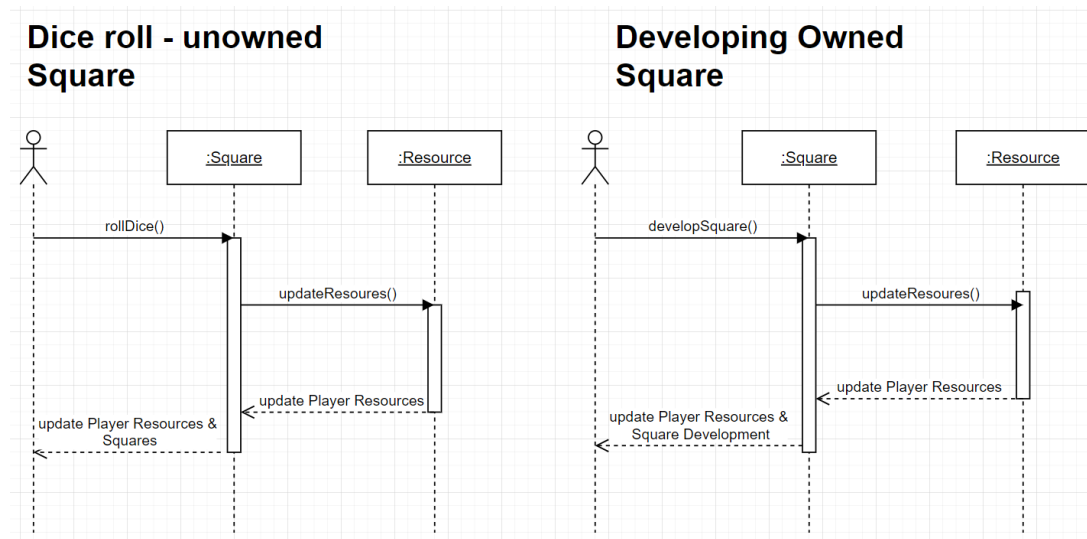
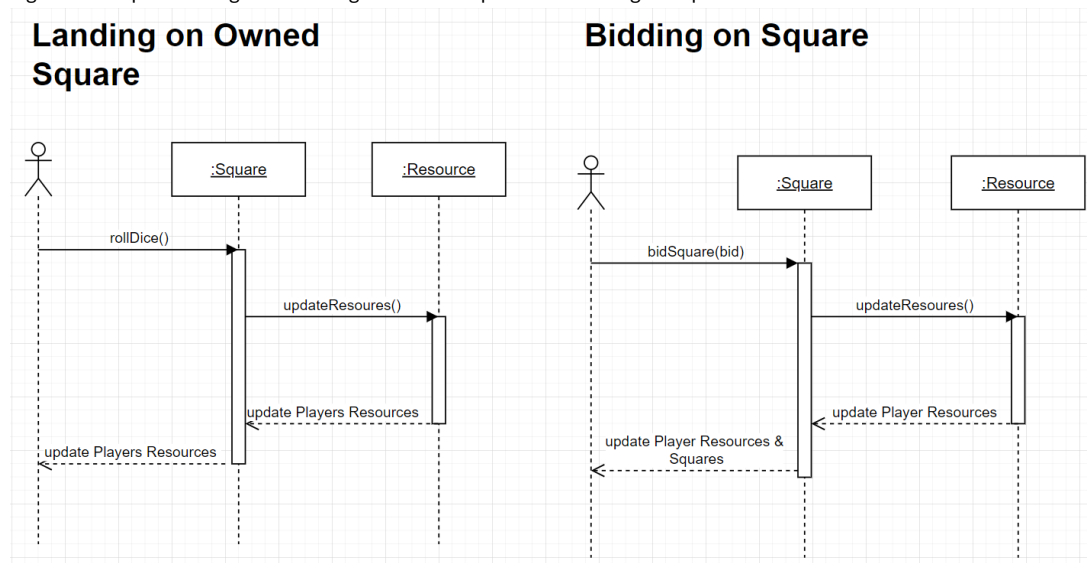
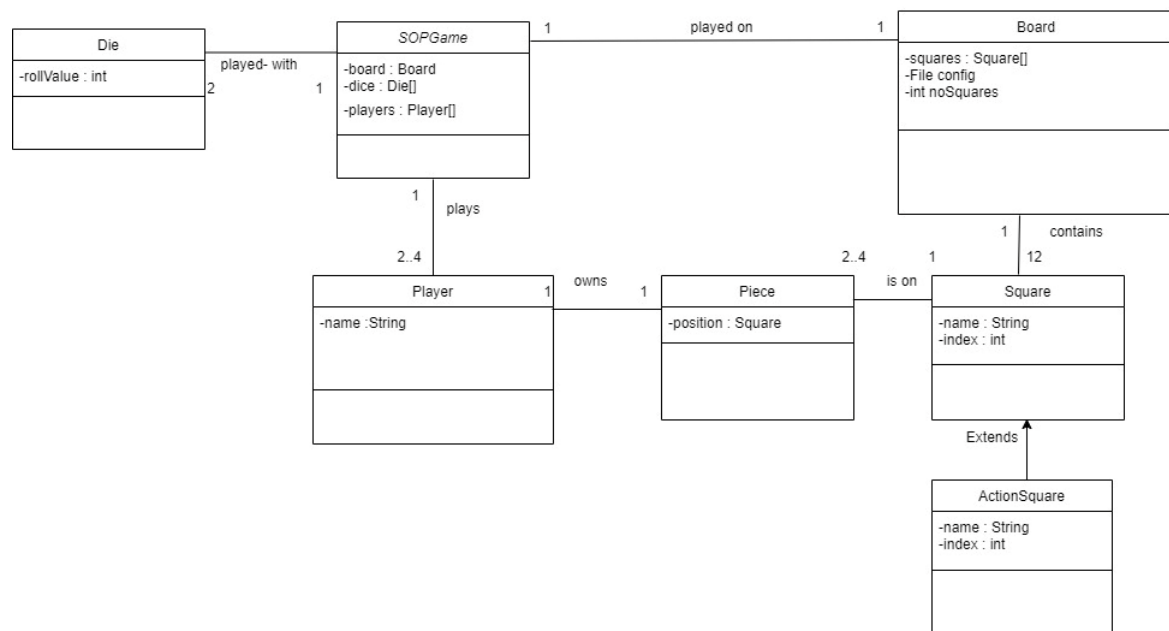


Figure 9 – Sequence Diagram: Landing on Owned Square and Bidding on Square



The above system diagrams (figure 8 & 9) show the process of the player resources being updated through a variety of scenarios; purchasing a square, developing a square, paying rent for a square and purchasing a square unwanted by the original player to land on it. Initially, we had considered implementing some sort of auction for unwanted squares but opted for a simpler process of offering the square at the original price to a random player; future versions of the code could easily be updated with an auction system though.

Figure 10 - High-level UML Class Diagram



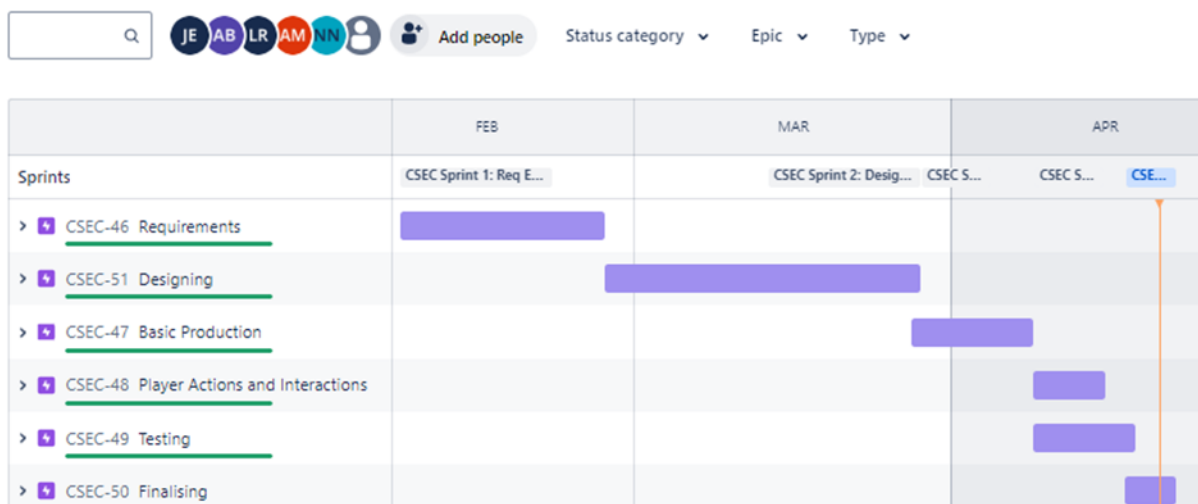
The final stage of the design phase was creating a high-level UML Class Diagram which showed the final structural model of the game. The high-level UML Class Diagram in figure 10 shows the classes and their relationships. For example, Square has attributes name and index, there is a composition relationship between the Board class and the Square class, with each Board containing 12 Squares, each Piece occupies a specific Square on the game board and there can be 2-4 Piece positions on Squares. ActionSquare is a subclass of Square, inheriting attributes and behaviours from the Square class. The SOPGame is played on 1 Board with 2 Die and 2-4 Players can play SOPGame.

Agile Iterations

For this project we used an Agile framework, this allowed a great degree of flexibility, collaboration, and iterative development, which allowed us to speedily produce a working prototype of the code we were working on and then gradually introduce improvements and additional features through iterative sprints week on week. As can be seen in the figure below we had 6 Sprints: Requirements; Designing; Basic Production; Player Actions and Interactions; Testing; and Finalising. Each of these sprints lasted between one and four weeks; it is worth noting that during the Designing Sprint we were also completing coursework for the Web Development component of our course, so factoring this into discussions we allowed ourselves more time for this Sprint to compensate. As the project continued, we also found certain Sprints overlapped more than we had originally foreseen.

Projects / CSC7083 Software Engineering Coursework

Timeline

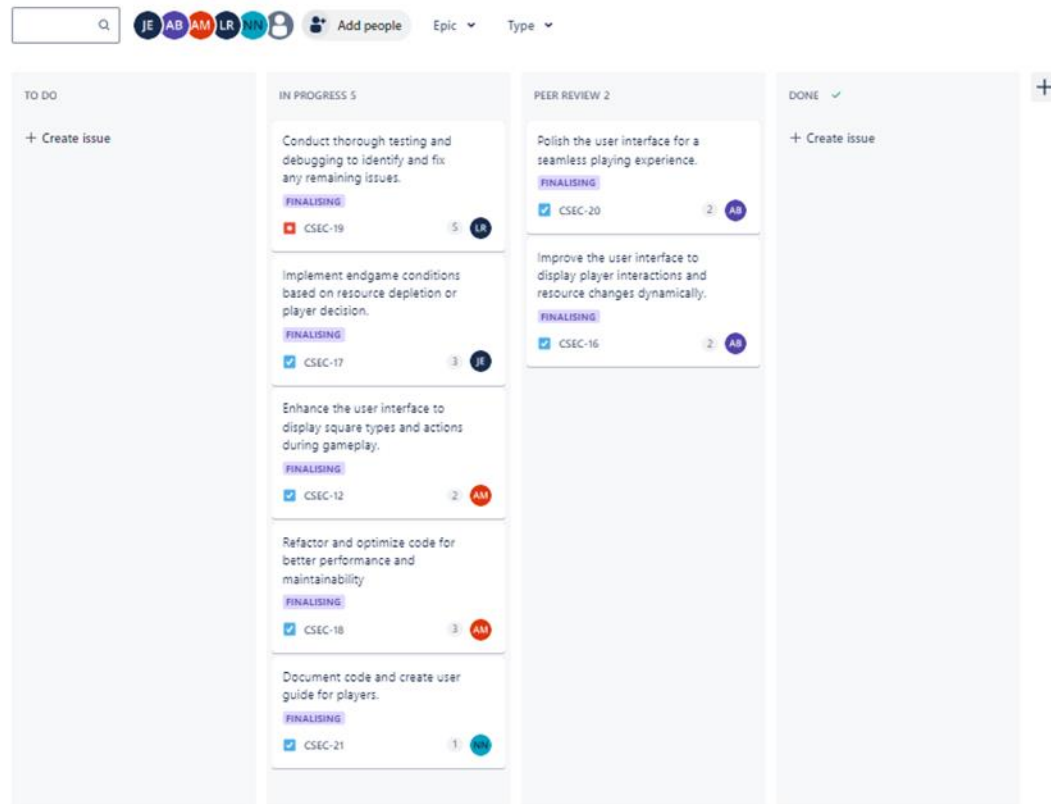


Within each sprint we used the Scrum board to visualise the tasks that needed to be completed, showing both who tasks were allocated to, and where they were in the development cycle, as can be seen in the example below.

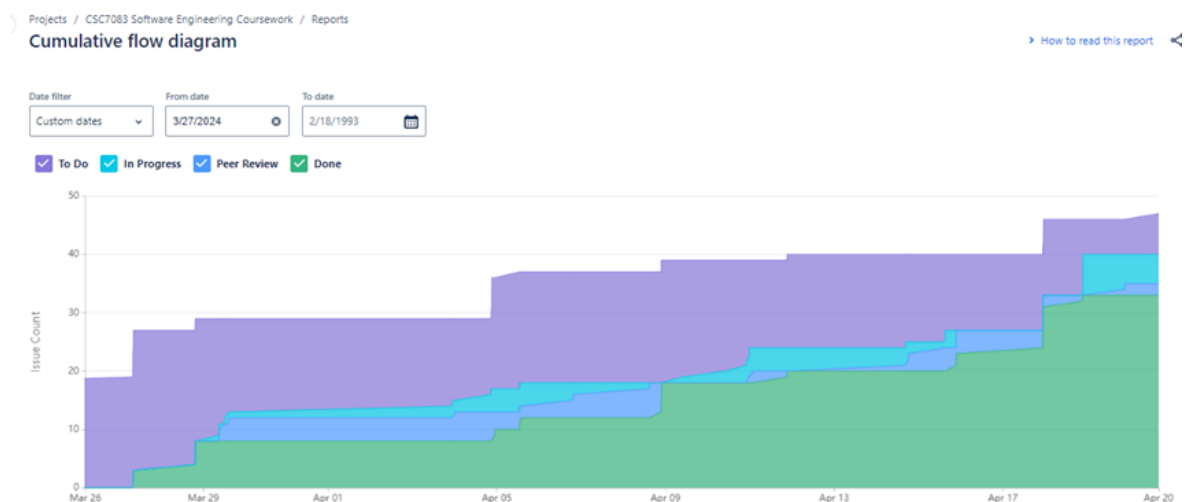
Projects / CSC7083 Software Engineering Coursework

CSEC Sprint 5: integrate

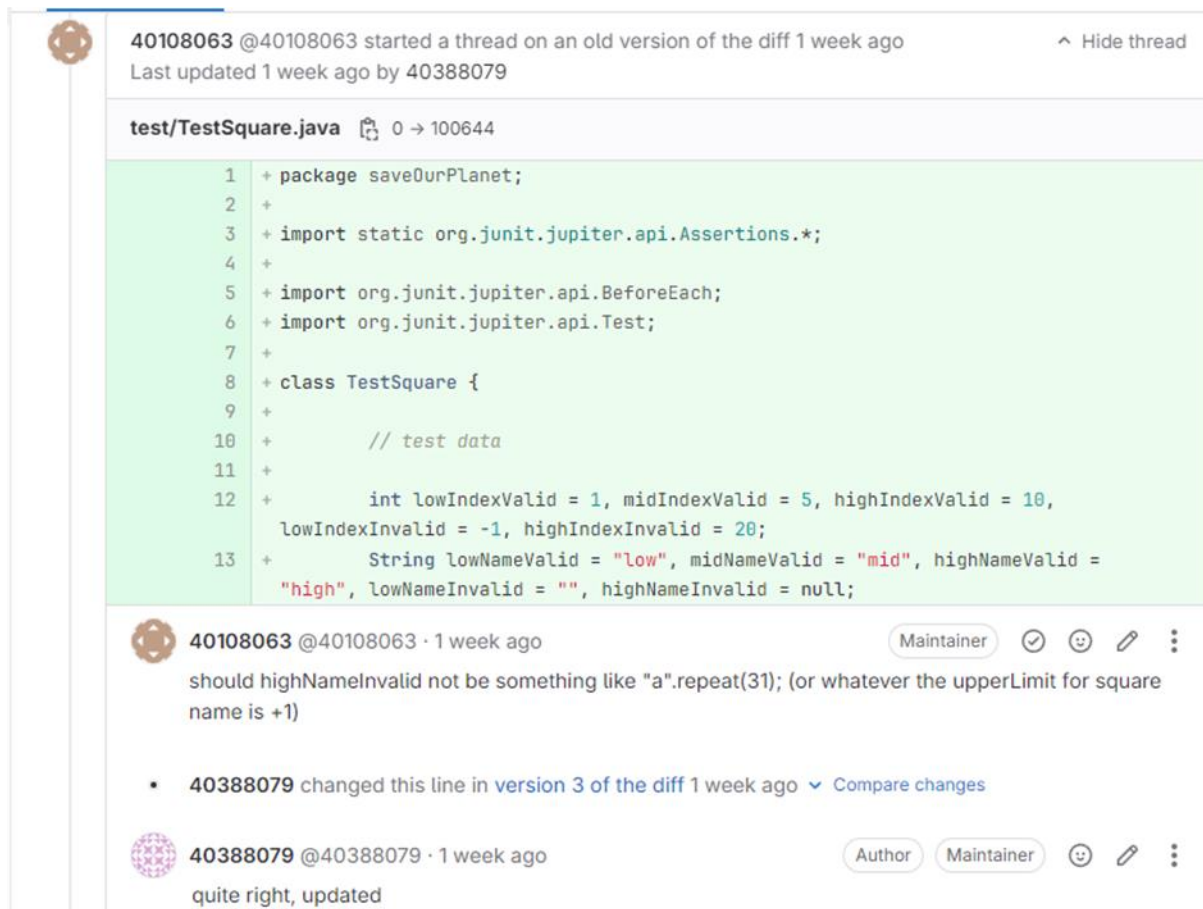
Integration



Unfortunately, due to the way our Jira workspace was initially setup the Sprint Burndown charts don't appear to be overly meaningful, but the below Cumulative flow diagram shows a steady progress from the initial setup of the projects issues to the completion of the same; with additional issues being added throughout the life of the project as and when they came to light during the iterative design flow.



During Sprint cycle reviews, we would discuss the work that had been completed during the Sprint making comments and suggestions for improvements where necessary (see team meeting minutes in Appendix 3). This can perhaps be seen most clearly during the Production and Testing stages where each new or amended piece of code was being added through a merge request which was reviewed by each member of the group; this was often done in the days leading up to group meeting, but generally approve and merge in new request during the meeting, below can be seen a snipped showing a merge request along with a suggested improvement. Using the tools available to us through GitLab and Jira allowed us to streamline the development and easily see where each of the group members were in terms of their workload and advice and assist them, as necessary.



40108063 @40108063 started a thread on an old version of the diff 1 week ago
Last updated 1 week ago by 40388079

test/TestSquare.java 0 → 100644

```

1 + package saveOurPlanet;
2 +
3 + import static org.junit.jupiter.api.Assertions.*;
4 +
5 + import org.junit.jupiter.api.BeforeEach;
6 + import org.junit.jupiter.api.Test;
7 +
8 + class TestSquare {
9 +
10 +     // test data
11 +
12 +     int lowIndexValid = 1, midIndexValid = 5, highIndexValid = 10,
13 +     lowIndexInvalid = -1, highIndexInvalid = 20;
14 +     String lowNameValid = "low", midNameValid = "mid", highNameValid =
15 +     "high", lowNameInvalid = "", highNameInvalid = null;

```

40108063 @40108063 · 1 week ago Maintainer ✓ 😊 ✎ ⋮
should highNameInvalid not be something like "a".repeat(31); (or whatever the upperLimit for square name is +1)

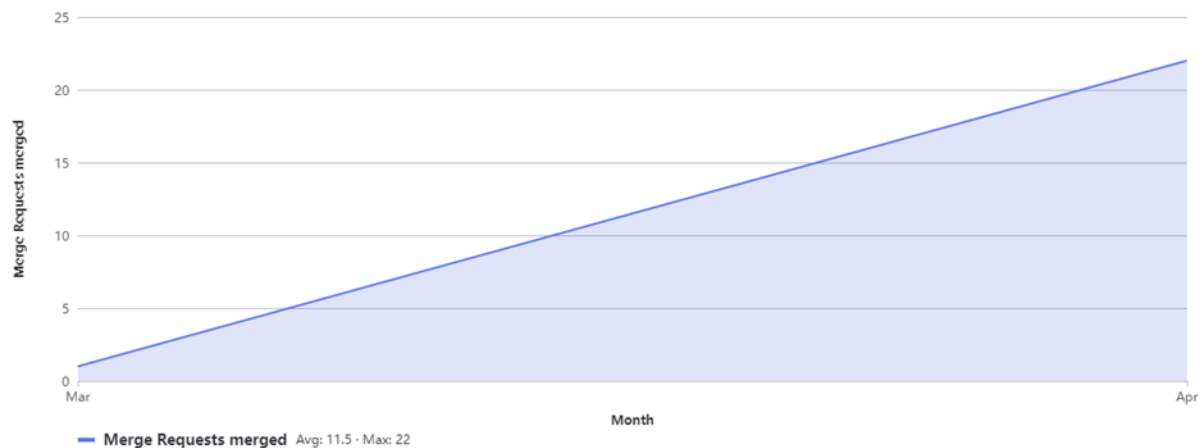
- 40388079 changed this line in [version 3 of the diff](#) 1 week ago Compare changes

40388079 @40388079 · 1 week ago Author Maintainer 😊 ✎ ⋮
quite right, updated

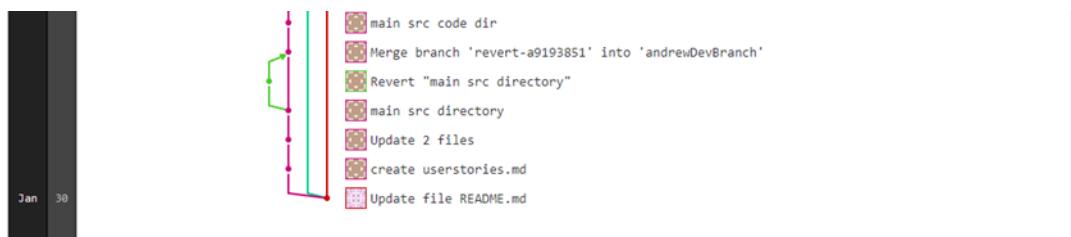
Our GitLab repository below shows a similar steady progress with number of merges to our main code branch steadily increasing as we came towards the end of project work cycle.

Throughput

The number of merge requests merged by month.



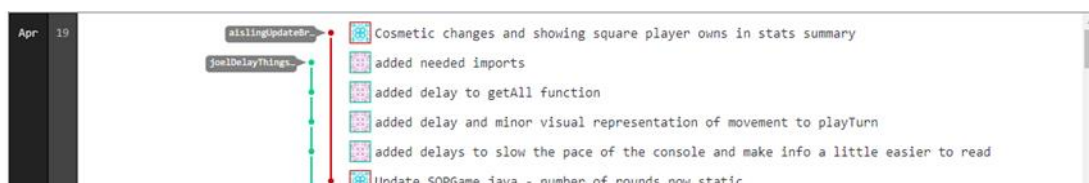
A few snips from our Repository graph further highlight the timescales and buildup of work. With the initial setup of our repository and code there were only a couple of branches.



But as the project rolled on and more amendments were being made the working branches increased in number, each dealing with a specific improvement or change.



Now as we are approaching the project's end only a few minor changes need to be merged with all other working branches closed off.



Testing

During the second code sprint we each turned our attention to the unit testing of our respective classes, which were written individually and merged to master. This allowed us to ensure code integrity during all phases, by providing fast feedback on the correctness of code changes, facilitating continuous integration, and enabling the team to deliver high-quality software iteratively. Furthermore, Unit tests provide a safety net that helps ensure that modifications to the codebase do not inadvertently introduce issues. By including a unit testing suite in the final code repository, it is much easier for a user to customise or extend the software to suit their specific needs improving the scalability of the final product.

After each iteration of code, we performed integration and acceptance testing (in coherence with the predetermined acceptance criteria) to verify new features/logic changes were consistent with the overall existing game mechanics and that they satisfied the requirements.

For example, in the first phase, verifying integration between Player, Square (and ActionSquare), and Gameboard classes. To ensure the core game loop functions correctly and the basic user interface elements display player turns and game state accurately. At every stage of integration regression testing was performed, rerunning our unit tests and integrated code, to ensure that we had not introduced any unintended bugs.

At the end of each sprint, we met as a group to carry out exploratory testing, using our collective experience and intuition to reveal any defects that may not have been identified throughout our automated testing.

During the final sprint, our attention turned to increasing our efforts in performance and user interface testing, ensuring the user interface functioned correctly in line with the design specifications, providing a seamless experience for players.

In conclusion, our use of an Agile methodology enabled us to deliver high-quality software iteratively and responsively. Through rigorous testing, continuous integration, and collaborative efforts, we ensured code integrity, alignment with requirements, and a quality user experience. By embracing Agile principles, we successfully navigated each sprint, delivering incremental value. The user guide in Appendix 4, shows our final product met all of the functional and non-functional requirements.

Appendix 1 – User Stories

1. Title: Player Name Entry at Gameplay Start

User Story:

As a player, I want to be able to enter the names of up to four players at the beginning of the gameplay, so that each player is identified uniquely throughout the game.

Acceptance Criteria:

1. When the game starts, a prompt should be displayed requesting the names of up to four players.
2. The system should allow the entry of alphanumeric characters for each player's name.
3. Each player name field should have a character limit of at least 20 characters to accommodate longer names.
4. Each player name should be unique.
5. The system should validate that at least one player name is entered before proceeding with the gameplay.
6. If fewer than four players are participating, the system should allow skipping the entry for the remaining players.
7. Once the names are entered, they should be displayed and stored for reference throughout the game.
8. The system should proceed to the gameplay stage using the entered player names.
9. Error handling should be in place to handle invalid inputs or unexpected behaviour during the name entry process.

Tasks:

1. Design the user interface for entering player names.
2. Implement the logic for capturing player names and validating inputs.
3. Integrate the player name entry functionality into the game's initialization phase.
4. Implement error handling mechanisms for invalid inputs or unexpected behaviour.
5. Test the player name entry feature thoroughly to ensure it functions as expected under various scenarios.
6. Document the process for entering player names and any relevant user instructions.

Additional Considerations:

- Ensure the user interface for entering player names is intuitive and user-friendly.
- Provide feedback to the user during the name entry process to indicate successful input or any errors encountered.

2. Title: Turn-Based Dice Rolling

User Story:

As a player, I want the game to implement turn-based mechanics where each player takes turns rolling two virtual dice, providing an element of chance and strategy to the gameplay.

Acceptance Criteria:

1. The game should have a defined starting player, either randomly selected or based on a predetermined order.
2. Each player's turn should begin with a prompt indicating it is their turn to roll the dice.

3. When prompted, the active player should be able to initiate the dice roll.
4. The system should simulate the roll of two virtual dice, each displaying a random number between 1 and 6.
5. The sum of the numbers rolled on the two dice should be calculated and displayed to the player.
6. The game should update the player's position based on the result of the dice roll.
7. After the dice roll and subsequent actions are resolved, the turn should pass to the next player in the predefined order.
8. The game should maintain a record of each player's turn and their respective resources.
9. The turn-based mechanics should continue until a player runs out of knowledge points, a player leaves the game or a player reaches the maximum number of knowledge points.

Tasks:

1. Design the user interface for displaying the dice roll results and prompting the player for their turn.
2. Implement the logic for simulating the roll of two virtual dice and calculating the total sum.
3. Integrate the dice rolling functionality into the turn-based system of the game.
4. Update the game state and player positions based on the result of the dice roll.
5. Implement the mechanism for passing the turn to the next player in the sequence.
6. Test the turn-based dice rolling feature to ensure it functions correctly for all players and under various game conditions.
7. Document the turn-based mechanics and how they interact with the dice rolling aspect of the game.

Additional Considerations:

- Allow for customization of turn order or game settings to accommodate different play preferences.

3. Title: Square Interaction and Action Choice

User Story:

As a player, I want the game to inform me of my current position on the board, along with any obligations or opportunities associated with that square, allowing me to make informed decisions and take appropriate actions.

Acceptance Criteria:

1. When a player lands on a square, the game should display information about the square, including its name, type, and any associated obligations or opportunities.
2. The information presented should be clear and concise, providing relevant details about the consequences of landing on that square.
3. If the square is unowned, an opportunity for the player to own square by dedicating resources should be presented to the player.
4. If the player chooses to take charge of the square, the game should deduct the required resources from the player's inventory and transfer ownership of the square to them.
5. If the player chooses not to take charge of the square, the game should offer the square to other players at the usual cost.
6. The game should handle any transactions or transfers of ownership smoothly and update the game state accordingly.
7. Once the player has made their decision or taken any necessary actions, the game should proceed to the next player's turn.

Tasks:

1. Design the user interface for displaying square information and action choices to the player.
2. Implement the logic for retrieving and presenting relevant details about each square on the board.
3. Develop the functionality for allowing players to take charge of unowned squares by dedicating resources.
4. Implement the mechanism for offering unowned squares to other players for purchase if the current player declines ownership.
5. Update the game state and handle any transactions or changes in ownership resulting from player actions.
6. Test the square interaction and action choice feature to ensure it functions correctly for all players and under various game conditions.
7. Document the square interaction mechanics and the various actions players can take when landing on distinct types of squares.

Additional Considerations:

- Provide cues to indicate the type of square and the available actions.

4. Title: Resource Change Notification and Balance Update**User Story:**

As a player, I want the game system to notify me whenever there is a change in my resources, providing clear information about the reason for the change and updating my current balance accordingly.

Acceptance Criteria:

1. Whenever a player's resources change, the game system should display what the change was and the reason for the change.
2. The message should be prominently displayed to the player.
3. The game system should update the player's current resources to reflect the new number of resources available.
8. The system should handle simultaneous resource changes for multiple players, ensuring each player receives accurate messages and resource updates relevant to the game play (i.e. if a player lands on a square owned by another player, the first player must pay rent to the second player therefore both players resources must be updated).

Tasks:

1. Design the message system to handle resource changes and balance updates in real-time.
2. Implement the logic for generating messages based on distinct types of resource changes (e.g., buying a square, developing a square, paying rent, passing start square).
3. Develop the functionality for updating the player's balance after each resource change, ensuring it reflects the correct amount of resources available.
4. Integrate the message and resource update features seamlessly into the game's existing interface and gameplay flow.
5. Test the message and balance update functionality to ensure it works correctly.
6. Document the message system and resource update process, including any relevant error handling procedures or edge cases.

5. Title: Start Square and Empty Square Functionality

User Story:

As a player, I want the game to include a Start Square where I can collect resources and an Empty Square where no action occurs, contributing to the overall gameplay experience.

Acceptance Criteria:

1. The game should include a designated Start Square where players begin their journey around the board.
2. Upon landing on the Start Square, players should automatically collect resources.
3. The Start Square should have a unique and creative name that reflects its purpose, setting the tone for the game's theme and narrative.
4. The resources collected from the Start Square should be displayed to the player.
5. There should be a separate Empty Square on the board where no specific action occurs when a player lands on it.
6. The Empty Square should have a distinct name or label that adds flavour to the game's environment.
7. When a player lands on the Empty Square, the game should display a brief message indicating that nothing happens and prompt the player to continue their turn.
8. The Empty Square should not have any direct impact on the player's resources, score, or game state.
9. Both the Start Square and Empty Square should be positioned appropriately on the game board, ensuring they contribute to the flow and balance of gameplay.

Tasks:

1. Design the layout of the game board, including the placement and virtual representation of the Start Square and Empty Square.
2. Define the rules and mechanics for resource collection upon landing on the Start Square.
3. Implement the logic for resource collection and display when a player lands on the Start Square.
4. Create a thematic name for the Start Square that aligns with the game's theme and narrative.
5. Determine the role and purpose of the Empty Square within the game's mechanics and story.
6. Implement the logic for handling player interactions with the Empty Square, ensuring it does not affect the game state.
7. Test the functionality of both the Start Square and Empty Square to ensure they function correctly and do not introduce any unintended gameplay issues.
8. Document the characteristics and behaviour of the Start Square and Empty Square for reference during development and gameplay.

6. Title: Field and Area Management in Save Our Planet

User Story:

As a player, I want the game to feature distinct fields representing various environmental sectors, each comprised of interconnected areas, allowing me to strategize and invest resources wisely to protect the planet.

Acceptance Criteria

1. The game should include four fields, each representing a significant environmental sector or initiative.
2. Two of the fields should consist of three areas each, while the remaining two fields should consist of two areas each, creating a balanced and varied gameplay experience.

3. Each field should have a unique name that clearly communicates its theme and purpose within the game.
4. The areas within each field should be thematically related to the field they belong to, fostering coherence and immersion.
5. Players should be able to acquire ownership of areas, with the cost of acquisition varying based on the field's significance and impact.
6. One of the fields with two areas should be the most expensive to acquire and resource, reflecting its critical importance in environmental conservation and sustainability efforts.
7. The other field with two areas should be the least expensive to acquire and resource, providing players with a more accessible option for expanding their influence on the board.
8. The game should display clear indicators of field ownership and resource allocation, allowing players to track their progress and assess their standing in the game.

Tasks:

1. Define the thematic concepts and names for the four fields based on environmental sectors and initiatives.
2. Identify and select appropriate areas to include within each field, ensuring they align with the field's theme and contribute to gameplay depth.
3. Determine the acquisition cost and resource value for each field, taking into account their significance and impact within the game.
4. Implement the logic for field acquisition and resource management, incorporating mechanisms for players to invest resources and gain ownership of fields.
5. Design the user interface elements to display field information, including ownership status, resource allocation, and associated benefits.
6. Test the field and area management mechanics to ensure they function correctly and provide engaging gameplay experiences.
7. Document the characteristics and attributes of each field and area for reference during development and gameplay.

Additional Considerations:

- Consider incorporating educational elements or information about real-world environmental initiatives and challenges within the game to promote awareness and engagement.

7. Title: Area Development and Field Custodianship**User Story:**

As a player, I want the game to enforce the rule that I must have ownership or custodianship of an entire field before I can develop an area within it. Additionally, I want the flexibility to develop areas in fields I already own, even if I am not currently positioned on that specific area.

Acceptance Criteria:

1. When a player decides to develop an area within a field, the game should check if the player has ownership or custodianship of the entire field.
2. If the player does not own or have custodianship of the entire field, the game should prevent them from developing any areas within that field.
3. If the player owns or has custodianship of the entire field, the game should allow them to proceed with developing an area within it.
4. The game should provide a clear indication to the player of their custodianship status for each field, helping them understand which fields they can develop areas in.

5. Players should be able to develop areas within fields they already own, even if they are not currently positioned on the specific area they wish to develop.
6. The game should allow players to initiate area development actions on their turn, regardless of their current position on the game board.
7. After developing an area, the game should update the player's resources or game state, reflecting the changes made to the field.

Tasks:

1. Define the criteria for custodianship of a field within the game's logic.
2. Implement the custodianship check mechanism to ensure players meet the requirements before attempting to develop areas within fields.
3. Design the user interface elements to display custodianship status for each field, providing clear feedback to players.
4. Develop the functionality for players to initiate area development actions on their turn, regardless of their current position on the game board.
5. Test the area development and custodianship mechanics to verify they function correctly and enforce the intended rules.
6. Document the custodianship rules and area development process for reference during development and gameplay.

8. Title: Development and Major Development Establishment**User Story:**

As a player, I want the game to feature a system of development, where I can invest resources to enhance specific aspects of the game world. Additionally, I want the ability to establish major developments, representing significant advancements or achievements, with the requirement of completing multiple minor developments beforehand.

Acceptance Criteria:

1. The game should include a mechanic for development, allowing players to enhance various areas within the game.
2. Each development should have a unique name and represent a tangible improvement or advancement within the game.
3. The cost of each development should be defined, in terms of resources.
4. Players should have the option to invest resources to initiate the development process.
5. Once a development is initiated, the game should update the game state to reflect the changes brought about by the development.
6. Other players may be required to invest in a development to utilize its benefits or features when they land on it, fostering interaction and competition among players.
7. Three minor developments must be completed before a player can establish a major development.
8. The major development should have a significant impact on the game, with the player gaining higher knowledge points than before when another player lands on it.
9. The cost of establishing a major development should be higher than that of a minor development, reflecting its significance within the game.
10. Upon establishing a major development, the game should provide substantial benefits or advantages to the player who initiated it.

Tasks:

1. Define the nature and significance of developments within the game, considering various aspects such as technology, infrastructure, or environmental conservation.
2. Determine the cost of each development and establish a mechanism for players to invest resources to initiate them.
3. Design the user interface elements to facilitate the development process, including options for players to select and initiate developments.
4. Implement the logic for tracking the progress of developments and updating the game state accordingly.
5. Define the interactions and transactions between players regarding the use or utilization of completed developments.
6. Develop the system for establishing major developments, including the requirement of completing three developments beforehand.
7. Test the development and major development mechanics to ensure they function correctly and provide engaging gameplay experiences.
8. Document the characteristics and effects of each development and major development for reference during development and gameplay.

9. Title: Resource Exchange for Developed Areas**User Story:**

As a player, I want the game to incorporate a mechanism where landing on an area owned by another player incurs a cost, proportional to the level of development in that area. This ensures that interactions between players involve strategic decision-making and resource management.

Acceptance Criteria:

1. When a player lands on an area within a field owned by another player, the game should trigger a resource exchange based on the development level of that area.
2. The cost of landing on an area owned by another player should be directly proportional to the level of development in that area.
3. The cost should be deducted from the resources of the player landing on the area.
4. The more developed the area (higher level of development), the greater the amount of resources consumed.
5. The game should clearly communicate the cost of landing on an area to the player before deducting the resources.
6. If a player does not have enough resources to cover the cost, the game will be over.
7. After the resource exchange, the game should update the resource balance for both the player who landed on the area and the owner of the area.
8. The game should provide messages to players to indicate the resource exchange and its impact on the players involved.

Tasks:

1. Determine the formula for calculating the cost of landing on an area based on its development level.
2. Implement the logic for deducting resources from the player landing on an area and updating the resource balance accordingly.
3. Develop the user interface elements to display the cost of landing on an area and facilitate resource exchange between players.
4. Integrate the resource exchange mechanism into the game's existing gameplay flow and mechanics.

5. Test the resource exchange feature to ensure it accurately calculates costs and updates resource balances for both players.
6. Document the resource exchange rules and mechanics for reference during development and gameplay.

Additional Considerations:

- Consider providing players with the option to negotiate or trade resources instead of paying the full cost when landing on another player's area.
- Ensure the resource exchange mechanism is balanced and does not excessively penalize players for landing on developed areas, maintaining fairness and strategic depth in gameplay.

10. Title: Determining the Outcome and Purpose of the Game**User Story:**

As a player, I want the game to have a clear outcome when one player runs out of resources or decides to no longer participate. Additionally, I want the game to reflect the ethos of Save Our Planet, whether it emphasizes resource accumulation or cooperative support for worthy environmental ventures.

Acceptance Criteria:

1. When one player runs out of resources, reaches the maximum number of resources or expresses the desire to no longer play, the game should conclude.
2. The outcome of the game should be communicated in a manner consistent with the overarching theme and style of Save Our Planet.
3. The player with the most knowledge points should be declared the winner.
4. The game should display the number of knowledge points held by each placing player (ie 1st, 2ⁿ and 3rd place).
5. The language used to convey the outcome should align with the values of environmental conservation, sustainability, and cooperation.
6. The game should gracefully handle the conclusion of the game and transition to a summary screen or end-game sequence.

Tasks:

1. Determine the overarching theme and ethos of the game: resource accumulation or cooperative support for environmental ventures.
2. Define the criteria for determining the winner or outcome of the game based on the chosen theme.
3. Implement the logic for detecting when one player runs out of resources or decides to exit the game.
4. Develop the user interface elements and end-game sequence to display the outcome and resource holdings of each player.
5. Craft appropriate language and messaging to communicate the outcome of the game in a manner consistent with the chosen theme and style.
6. Test the end-game scenarios to ensure they are triggered correctly and the outcome is conveyed accurately.
7. Document the rules and mechanics governing the game's conclusion and outcome for reference during development and gameplay.

Additional Considerations:

- Consider offering players the option to view a summary of their contributions to environmental ventures or achievements unlocked during the game, reinforcing the theme of Save Our Planet.

Appendix 2 – Use Case Descriptions

Name	Player registration
Objective	To start gameplay, obtaining the number of players (2-4) and names of the players.
Actors	All players
Precondition	Eclipse must be open.
Main Flow	<ol style="list-style-type: none"> 1. System prompts for number of players. 2. User enters the number of players. 3. System prompts for name for each player. 4. Users enter name of each player. 5. The system orders users in alphabetical order. 6. The system welcomes players. 7. The system assigns 200 knowledge points to all players. 8. The system places all players at "The Green Beginning".
Alternative Flows	<p>A. At 1., if the number of players is not between 2-4 inclusive, the system will ask for the number of players until it falls within this range.</p> <p>B. At 4., if a user enters a name that another user in the game has taken, the system will prompt for a different name to be chosen.</p>
Post condition	The number of players and player name is set, each player holds 200 knowledge points. All players are placed at "The Green Beginning".

Name	Play Turn
Objective	The system prompts the player whose turn it is if they want to roll the dice.
Actors	Player
Precondition	Gameplay is set up with 2-4 named players.
Main Flow	<ol style="list-style-type: none"> 1. The system asks if the player would like to roll (Y/N). 2. Player enters Y.
Alternative Flows	<ol style="list-style-type: none"> 1. At 1., if a player selects N, the system asks them to select N again if they are sure. 2. If the player selects N again the game is over. 3. If the player selects Y after initial N, the user will be prompted again if they would like to roll. <p>1. At 1., if a player enters anything other than Y/N, the system asks for Y/N again.</p>
Post condition	Player System reacts to Player entry accordingly.

Name	Roll Dice
Objective	The system prompts the player whose turn it is if they want to roll the dice
Actors	Player
Precondition	Gameplay is set up with 2-4 named players and player had selected Y to play turn
Main Flow	<ol style="list-style-type: none"> 1. The system generates two random numbers (each 1-6 inclusive) 2. The system combines them to determine the player's roll value 3. Roll value is set 4. Roll value is displayed to console

Alternative Flows	N/A
Post condition	Player roll value has been set

Name	Move to Destination Square
Objective	Displays square player landed on and options
Actors	Player
Precondition	It is a player's turn
Main Flow	<ol style="list-style-type: none"> 1. The system moves the player's piece along the virtual game board according to the roll value 2. The system informs the player of the square that they have landed on, it is type and associated costs and ownership if of type field 3. The system informs the player of the actions they can take 1. 4. The player selects which action they would like to take 1.(if applicable)
Alternative Flows	<ol style="list-style-type: none"> 1. At 3, if the player has landed on the Carbon Neutral or Green Beginning Square, the system will inform the player. The system displays the player's current knowledge point profile and square ownership status. 2. The player's turn terminates.
Post condition	Player has moved on the virtual board and now could collect resources, buy square, or do not buy square or, if square is already owned, pay resources.

Name	Collect Resources
Objective	Displays square player landed on (' <i>The Green Beginning</i> ') and automatically receives a set number of resources.
Actors	Player
Precondition	It is a player's turn and they have landed on or passed 'The Green Beginning' Square
Main Flow	<ol style="list-style-type: none"> 1. The system informs the player they passed 'The Green Beginning' square 2. The system calls method to update knowledge points
Alternative Flows	N/A
Post condition	Player has moved on the virtual board and will collect knowledge points

Name	Pay Resources
Objective	The current Player must pay the Player that owns the square the appropriate knowledge points
Actors	Player
Precondition	Player lands on a square owned by another player
Main Flow	<ol style="list-style-type: none"> 1. The system displays the name of the square and field, any developments on the square, the name of the owner of the square and the rental fee (X knowledge points) for landing on the square. 2. The system calls method to update knowledge points

Alternative Flows	N/A
Post condition	Knowledge points are updated

Name	Update Knowledge Points – collect resources / pay resources
Objective	When a player lands on a square owned by another player, they must pay the owner of the square the appropriate number of knowledge points. Or when a player lands on/passes <i>'The Green Beginning,'</i> they will receive 200 knowledge points.
Actors	Players (Player A and Player B)
Precondition	Player A has landed on a square owned by Player B. Or player has landed on / passed <i>'The Green Beginning.'</i>
Main Flow	<ol style="list-style-type: none"> 1. Knowledge points owed are deducted from Player A. 2. Player B's knowledge points are increased by amount owed. 3. The system displays "X knowledge points transferred from Player A to Player B. Player A's new balance: X knowledge points. Player B's new balance Z knowledge points." 4. Gameplay continues to <i>Develop Resources</i> if player is in possession of all squares within a category.
Alternative Flows	<ol style="list-style-type: none"> 1. At point 2, if player A does not have sufficient funds to pay player B, the game is over. 2. At point 4, if not able to develop resources, game moves on to next player's turn.
Post condition	Transfer of points has occurred. Player A has settled balances with Player B.

Name	Buy Square
Objective	When a player lands on an unowned square, they will have the opportunity to buy it.
Actors	Player
Precondition	Player A has landed on a square that is not currently owned by any other players.
Main Flow	<ol style="list-style-type: none"> 1. The system displays the name, the field, and the cost of the square (X knowledge points). 2. The system displays player's current balance. 3. If the player has sufficient funds to buy the square, the system asks the player if they would like to buy the square (Y/N) 4. The player enters Y 5. The system congratulates player and they now own [Square Name]
Alternative Flows	<ol style="list-style-type: none"> 1. N/A
Post condition	Update Development Portfolio

Name	Do Not Buy Square
Objective	When a player lands on an unowned square, and they chose not to buy it, it is offered to other players to buy it.
Actors	All Players

Precondition	Player has landed on a square that is not currently owned by any other players and does not want to buy it.
Main Flow	<ol style="list-style-type: none"> 1. The system displays the name, the field, and the cost of the square (X knowledge points). 2. If the player has sufficient funds to buy the square, the system asks the player if they would like to buy the square (Y/N) 3. The player enters N
Alternative Flows	N/A
Post condition	Square is offered to next player

Name	Square offered to other players
Objective	When a player lands on an unowned square, and they chose not to buy it, it is offered to other players to buy it.
Actors	All Players
Precondition	Player has landed on a square that is not currently owned by any other players and does not want to buy it.
Main Flow	<ol style="list-style-type: none"> 1. The system shuffles the remaining players to determine the order in which to ask 2. The system checks if the next player has sufficient funds to purchase the square. 3. The system prompts the next player to enter Y/N if they would like to purchase the square. 4. Next player enters Y 5. The square is now purchased and the system displays “[Square Name], belonging to [Square Field] purchased by [Player]”
Alternative Flows	<ol style="list-style-type: none"> 1. If at step 3, the player does not have sufficient funds, the square remains unowned. 2. The system displays “[Square Name], belonging to [Square Group] remains unowned” <ol style="list-style-type: none"> 1. If at step 3, the player enters N, the system asks the next player in the shuffled list.
Post condition	Portfolio is updated and play goes back to the initial player to continue turn

Name	Update Development Portfolio
Objective	When a player buys a square their knowledge points are decreased, and square is updated as owned by player x
Actors	All Players
Precondition	Player has chosen to buy a square and has sufficient knowledge points
Main Flow	<ol style="list-style-type: none"> 1. The square is updated as owned and the system displays “[Square Name], belonging to [Square Field] purchased by [Player]” 2. Knowledge points to purchase square are deducted from player 3. The system displays player’s current balance.

Alternative Flows	N/A
Post condition	Play goes back to the initial player to continue turn

Name	Develop Resources / Property
Objective	To enable a player to develop resources on a square.
Actors	Player
Precondition	The player is currently in possession of all squares within a field. It is the player's turn and they have rolled the dice and moved on the board. If applicable rent has already been paid for the square landed on.
Main Flow	<ol style="list-style-type: none"> 1. The system displays information on the field(s) and areas that the player possesses, including the costs of development (minor and major developments) 2. The system asks the player if they would like to develop on a square (Y/N) 3. If the player owns multiple complete fields, the system prompts the player to choose a field to develop. 4. The player enters the name of the square. 5. The system asks the player if they would like to build development (if minor developments <3, it will ask if they would like to build a minor development, if minor developments = 3, it will ask if they want to build a major development). 6. The player enters Y/N 7. If Y and the player has sufficient knowledge points, the appropriate knowledge points are deducted from the player's balance and the rent price is adjusted. 8. The system displays "[Square Name], belonging to [Square Group] now contains [Development]. Rent price has increased to [price]" 9. The system displays Player A's current balance and portfolio. 10. Gameplay continues to the next player's turn
Alternative Flows	<ol style="list-style-type: none"> 1. If N at point 6, the player's turn is terminated and gameplay continues to the next player's turn. 1. If at step 7, the player has insufficient funds to proceed, the system prints "Sorry, you do not have the appropriate funds to develop at this time." 2. The player's turn is terminated and gameplay continues to the next player's turn
Post condition	If player has developed on a square, the rent price of the square has increased.

Name	End Game
Objective	To finish the gameplay and declare a winner.
Actors	Players
Precondition	A player has achieved a specified level of scientific knowledge points (maximum points or bankrupt) or a specified number of rounds have been played or player(s) have chosen not to roll / progress to next round.
Main Flow	<ol style="list-style-type: none"> 1. System orders players based on knowledge points

	<ol style="list-style-type: none"> 2. System displays congratulatory message winning player to the player with the most knowledge points. 3. System displays statistics for all players 4. Gameplay terminates
Alternative Flows	1. At 1., if two players have the same amount of knowledge points, rank order is determined by alphabetical order.
Post condition	Winner determined and displayed. Gameplay terminated.

Appendix 3 – Minutes of team meetings.

January meetings:

Date and Location: 11/1/24 – Teams.

Meeting Objectives: Ice breaker

Attendees: Joel Egerton, Aisling Broder-Rodgers
Nathaniel Nimmock,

Apologies: Andrew Morrison, Leonie Robinson

Agenda: Ice breaker.

Discussions (challenges, and progress by each team member on previous tasks):

How we have found the course so far, general personal information exchange to better know each other.

Teams and what's app group set up by Nathaniel.

Jira set up by Joel – invites sent to the rest of the team.

First meeting – no previous tasks set.

Action Plan (with allocations to each team member)

N/A

No action plan in place as whole team had not met yet.

Date and Location: 15/1/24, Teams.

Meeting Objectives: Ice breaker.

Attendees: Joel Egerton, Aisling Broder-Rodgers
Nathaniel Nimmock, Andrew Morrison, Leonie Robinson

Apologies: N/A

Agenda: Ice Breaker.

Discussions (challenges, and progress by each team member on previous tasks):

First full house, more ice breakers, whole team acquainted with each other.

Collectively discussed user stories, and current understanding of specification.

Action Plan (with allocations to each team member):

Continue with learnings from lectures and individual study of the specification.

Come back and amalgamate ideas at next meeting.

Date and Location: 25/1/24 - teams

Meeting Objectives: Formulate a starting point based on the spec provided

Attendees: Nathaniel Nimmock, Leonie Robinson, Joel Egerton, Andrew Morrison

Apologies: Aisling Broder-Rodgers

Agenda: Formulate a starting point based on the spec provided.

Discussions (challenges, and progress by each team member on previous tasks):

Where to start –

-Defining squares

-Defining zones

-Looking at use cases

-Andrew and Leonie have created draft user stories.

-Cutting through jargon in spec – believed to be deliberately wordy to simulate how a customer from a different background may present a concept.

-Agile approach to better reveal concepts over development phase.

-Considering time frame available with reference to other responsibilities.

-Getting everyone set up to use JIRA collectively and discussing its use.

-Discussing potential use of confluence to store documentation.

-Epics, how to implement with reference to what we have so far when moving to Jira.

-Getting items on the backlog to move forward.

-Discussed getting started using GitLab- but not yet provided to use.

Action Plan (with allocations to each team member):

- no specific allocations at this time.

-User stories doc's to be amalgamated then added to Jira so this can be modified collectively

-Look further at UML so we can collectively get a start on this.

February meetings

Date and Location: 1.2.24 – Teams.

Meeting Objectives: No set objective

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

s

Apologies: N/A

Agenda:

No set agenda.

Discussions (challenges, and progress by each team member on previous tasks):

Invite Janak to next meeting.

Discuss project sprints overview – created by Andrew.

Agreed framework with option of flexibility.

Discuss User Journey Map – created by Andrew.

Agreed looks good to begin acting upon by team.

Dashboard created on Jira to keep track of tickets.

Discussed that although this is designed to replicate a real-life situation, without a designated project lead disseminating specific tasks the situation is difficult.

Logging into EEECS to access Gitlab.

Action Plan (with allocations to each team member):

Break for a week, persona learning to better understand how to progress

Date and Location: 5.2.24 – Teams.

Meeting Objectives: Discuss personal learnings and apply this to project

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Aisling Broder-Rodgers

Apologies: Andrew Morrison

Agenda:

No set agenda – non full house so general discussion.

Deferred invite to Janak as not everyone is present

Discussions (challenges, and progress by each team member on previous tasks):

Discussed general project requirements and documentation created so far with respect to what learning and new ideas we have from the last meeting, nailing down details and what still needs to be done.

epic, user stories.

User story map

User journey mapping

Flowcharts

Use case diagrams (and use case descriptions – tables)

Sequence diagrams. Class diagram – final structure model.

Creation of game and virtual game board – section 4 and 5.

Theme – names of fields and squares and sections.

Points system – knowledge points.

Decide who wins – potential scenario where everyone loses.

Decide who starts – random / alphabetical / youngest player?

What happens when a player runs out of resources?

Everyone in attendance happy with current progress and documentation

Action Plan (with allocations to each team member):

Leonie updating current draft of user stories in line with discussions, making this available or further comment.

Look at Requirements in more details and potentially finalise this at next meeting.

Same for any other documentation we are happy to move forward with.

Invite Janak to next meeting where everyone is available.

Date and Location: 8.2.24 – teams.

Meeting Objectives: Refine the requirements and user stories to a near final version.

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison

Apologies: Aisling Broder-Rodgers

Agenda: Refine the requirements and user stories to a near final version.

Discussions (challenges, and progress by each team member on previous tasks):

Meeting not recorded – minutes created post meeting so some details may be missing.

Progress –

EcoVenture document update by Leonie and made available in teams for review.

User map made by Leonie on Mira made available on teams – potential epics user stories and tasks.

Gameboard diagram by Joel on Draw.io made available on Teams for review.

Agreed documentation we have is good to move forward with and to be amended as required as development need arises.

Action Plan (with allocations to each team member):

Break for reading week/Web dev submission – resume meetings after

Date and Location: 19.2.24 – Teams.

Meeting Objectives: Meet with Janak to get an idea of where we are at and out best next steps.

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Aisling Broder-Rodgers, Janak Adhikari

Apologies: N/A

Agenda: Meet with Janak to get an idea of where we are at and out best next steps.

Discussions (challenges, and progress by each team member on previous tasks):

Meeting not recorded so some details made be missing -

Showed the work we had compiled so far, Janak said we were on track.

Nothing else discussed at this time.

Action Plan (with allocations to each team member):

No action plan set.

Date and Location: 26.2.24 – Teams.

Meeting Objectives: collectively run through run through sequence diagram

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison

Apologies: Aisling Broder-Rodgers

Agenda: collectively run through run through sequence diagram

Discussions (challenges, and progress by each team member on previous tasks):

Updated Sequence diagram

Define resource as either class variable or object – seems best to be class variable.

Piece- i.e. item representing player, hat dog etc, required? Will look at this further as classes are put together.

Implement leaderboard function, can this be called at any time from a menu for example so a player can see where they rank at any point. Spec indicates this should be provided at the end of each turn. Perhaps hashmap can be used here to make sure of key value pairs.

Collective challenge – Web development CW due on 14.3.24 – best endeavours to get what we can done over the coming few weeks.

Action Plan (with allocations to each team member):

Andrew - Develop overall relationship sequence diagram.

Andrew – Develop further on initial UML draft.

Nathaniel – continue refactoring minutes documents.

Rest of team – continue research of problem spec and various ways to implement.

Potentially meet again with Janak next week to run through what we have

March meetings

Date and Location: 11/3/24 – Teams.

Meeting Objectives: speak with Janak, show what we have completed so far.

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers, Janak Adhikari

Apologies: N/A

Agenda: speak with Janak, show what we have completed so far.

Discussions (challenges, and progress by each team member on previous tasks):

Advice presented during meeting.

Janak - Split overall relationship into separate boards as full gameplay loop can leave things out of place.

Janak - Combine square diagrams- Bidding can remain separate.

Janak - Use case descriptions – ?? – didn't quite catch what he said.

Janak - just write use case names - Map branching of game play flow.

Janak - Aisling/Leonie diagram include fail sequence – end if run out of resource etc.

Remove – no action use case.

Remove start game implementation level details for start game. Enter player name etc.

Janak feedback – Use, use case descriptions to create sequence diagram.

Action Plan (with allocations to each team member):

Implement changes suggested by Janak – meet again and form action plan to begin coding

Date and Location: 23/3/24-Teams.

Meeting Objectives: production

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda:

Andrew has prepared a starting point for getting this moving.

Review this, separate out jobs and proceed

Discussions (challenges, and progress by each team member on previous tasks):

Sequence diagram of various classes and overall game loop

UML, 1st iteration – classes variable and methods

Method calls will be synchronous.

CSV file to be read to configure squares.

Opportunities for multithreading – write out results to csv?

Opportunity to give option to save current players, writes out player object data to csv -Will also require option to read player data.

All agree after first run through of ULM that we have enough to move forward.
Though UML is more or less ready, we don't have this split into actionable tasks on Jira

Action Plan (with allocations to each team member):

Joel to take UML and create tasks for back log – tasks to be divided out on next meeting.

Date and Location: 28.3.24 – Teams.

Meeting Objectives: Get third sprint created (production) and assign tasks.

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda: Get third sprint created (production) and assign tasks.

Discussions (challenges, and progress by each team member on previous tasks):

Requirements engineer (designing) sprint completed.

Contents and goals of sprint 3 determined.

Assign tasks in order to commence testing on the classes.

Tackle Game logic collectively.

Meeting frequency, keep to once a week or add mid-week meeting also? – undecided.

Action Plan (with allocations to each team member):

Nathaniel Nimmock, Player class, minutes

Joel Egerton, square, Square class, Square Enum

Leonie Robinson, Piece class, Dice class

Andrew Morrison, Gameboard class, csv config

Aisling Broder-Rodgers ActionSquare class, core game loop

Reconvene to determine testing approach.

April meetings

Date and Location: 4.4.2024-Teams.

Meeting Objectives: Review sprint 3 progress

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda: Review sprint 3, address concerns, opportunities, and progress.

Discussions (challenges, and progress by each team member on previous tasks):

- Removal of array from die class as not required.
- Field, Enum or object – went with Enum, simplifies use in board class.

- Use of HashMap to keep track of squares fields and developments.
- Use of burndown charts? - not yet implemented assumed to work in background as we progress sprint(s).
- Context of next sprint – unit testing and error handling.
 - -Decided we would test our own classes – familiarity with class implementation should lead to faster.
 - -completion of testing – additionally where overlap of classes occur – can testing be shared?
 - -Challenge identified here as full project is unavailable at this stage – tests will need revisited.
- Comparator for knowledge points required to display KP stats and determine player rank.
- Win conditions revisited and discussed –
 - -first to max KP or first with major development?
 - -Last person standing? i.e. last to run out of KP or last to quit?
 - -Timed condition – rank of best stated at a pre-determined interval?
- ActionSquare 1st iteration completed – Aisling
- SOPGame 1st iteration completed – Aisling
- Basic class structure completed - Andrew
- Board class 1st iteration completed – Andrew
- Config csv for Board class 1st iteration completed – Andrew
- SquareType enum 1st iteration completed – Joel
- Square class 1st iteration completed – Joel
- Piece class 1st iteration completed – Leonie
- Die class 1st iteration completed – Leonie
- Player class 1st iteration completed – Nathaniel
- Previous meetings minutes recorded – Nathaniel

Final elements of sprint 3 after review to be polished, then sprint closed.

sprint 4 created – testing, error handling and validation elements added from backlog.

Action Plan (with allocations to each team member):

Aisling - Basic user interface for displaying turns and game state, develop resource management mechanisms to track and update player resources based on game actions, test ActionSquare.

Andrew - test Board class, test Field class, create methods to facilitate offering a development to player in Main game class,

Joel - test Square Class, enhance error handling and validation for player actions.

Leonie – enhance error handling and validation for player actions, implement player interactions for acquiring fields and exchanging resources between players, test Piece class, test Die class.

Nathaniel – create comparator for KP, test Player class, record minutes

Date and Location: 8.4.24 – Teams.

Meeting Objectives: Review final elements of sprint 3

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda: Review progress on sprint 3 completion, Planning for sprint 4

Discussions (challenges, and progress by each team member on previous tasks):

- Most recent class data no git pulled down and ran locally for review.

- Discussed complexities of creating a class that contains dependencies from other classes you have not seen yet.
- Revisited die class, consensus is array to be removed replaced by one die object and two rolls to simulate two dice, changes also made to player and other classes class to reflect this.
- Revisited player class, play turn to be simplified.
- Revisited Piece class – PlayerPositions method not required so removed.
 - Move logic for going back to the start and passing go to Board class,
- Board class to be updated to separate out get current and next square
- Should game logic be within SOPgame class or in a separate driver class?
- Revisited Player class UpdateKnowledgePoints method split to separate methods for increasing and decreasing knowledge points.
- Reviewed offer development logic – looks good
- Decided on knowledge points being the winning condition for the game.
 - Discussed potential implementation.
 - Discussed how knowledge points would be gained, pass go rent etc.
 - How knowledge points comparator would be used
- Reviewed remaining merge requests for sprint 3 – ActionSquare Class merged to main, Field class merged to main, a few minor changes still required in Player class, Die class and SOPGame class, then to be merged.
- Sprint 3 to be closed when changes and merges are completed.
- Assigned items from board for Sprint 4 Testing
- Basic user interface for displaying turns and game state, develop resource management mechanisms to track and update player resources based on game actions, done with some minor adjustment required, the merge. test ActionSquare started - Aisling
- create methods to facilitate offering a development to player in Main game class, done - Test Board class, test Field class, started – Andrew
- enhance error handling and validation for player actions. Done, test Square Class started - Joel
- enhance error handling and validation for player actions, implement player interactions for acquiring fields and exchanging resources between players done, test Piece class, test Die class started - Leonie
- create comparator for KP done, minor changed to Player class review need some tweaks, record minutes, done, test Player class, started. – Nathaniel

Action Plan (with allocations to each team member):

Final elements of sprint 3 after review to be polished, then sprint closed.

sprint 4 – complete assigned tasks, fully test and identify issues

- Aisling – finalise and merge SOPGame, proceed with test ActionSquare.
- Andrew – proceed with testing Board class and Field class. Create methods to offer developments.
- Joel – proceed with testing Square Class.
- Leonie – finalise and merge Die class, proceed with testing Piece class and Die class.
- Nathaniel – finalise and merge Player class and comparator for KP, proceed with testing Player class, record minutes.
- ALL – check all classes locally to see all method name, parameters etc are correct

Date and Location: 11.4.24 – Teams.

Meeting Objectives: Discuss progress on sprint 4

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda: Discuss progress on sprint 4

Discussions (challenges, and progress by each team member on previous tasks):

- How to implement upper and lower bound for knowledge points.

- Initial local integration of everyone's classes working as is but needs further refinement.
- Review current merge requests.
- Discussed progress on testing.
- Discussed challenges of testing methods vs business rules.
- Reviewed current updates and merge requests, addressed comments.
- Reviewed square states, pricing etc to make sure game was playable – all fine as is.
- Discussed potential changes rising from issues discovered during testing.
- Testing of ActionSquare class completed, SOPGame class updated – Aisling.
- Testing of Board class, test Field class, completed, created methods to offer developments – Andrew.
- Testing of Square Class completed – Joel.
- Testing of Piece class completed, Die class updated, testing of Die class, completed – Leonie.
- Testing of Player class completed, minutes recorded – Nathaniel.
- Checked over local copy of main for issue - completed. – All.

Action Plan (with allocations to each team member):

sprint 4 – complete assigned tasks, fully test and identify issues.

- ALL – Update testing to reflect planned and impromptu changes identified, update and merged to main in first round of testing.
- ALL – Proceed with identifying any further issues identified by updated testing.
- ALL – start looking at cleaning up and preparing materials for reporting.
- Aisling – proceed with updating ActionSquare class tests.
- Andrew – proceed with updating Board class and Field class tests.
- Joel – proceed with updating Square Class tests.
- Leonie – proceed with updating Piece class and Die class tests.
- Nathaniel – proceed with updating Player class tests, record minutes.

Date and Location: 15.4.24 – Teams.

Meeting Objectives: Discuss progress of and potentially close sprint 4

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda: Discuss progress and potentially close sprint 4

Discussions (challenges, and progress by each team member on previous tasks):

- Discussed current progress of report – initial structure created by Leonie.
 - Discussed potential elements to be included in report.
 - Discussed potential elements to be included in the appendix.
- Discussed current use of user journey and user story mappings.
 - Potential rejig epic epics required

- Review sprint 4 progress and merge requests.
 - TestSquare and comments reviewed - merged to main. Joel
 - TestSOPGame – and comments reviewed - merged to main. Joel
 - Update to SOPGame Class and comments reviewed - merged to main. Aisling
 - ActionSquareTest and comments reviewed - merged to main. Aisling
 - Update to Player Class and comments reviewed - merged to main. Nathaniel
 - TestPlayer and comments reviewed - merged to main. Nathaniel
 - DieTest and comments reviewed - merged to main. Leonie
 - PieceTest and comments reviewed - merged to main. Leonie
 - Update to Board Class and comments reviewed - merged to main. Andrew
 - BoardTest and comments reviewed - merged to main. Andrew
- Need to update SOPGame with final game logic.
- Need to clean up console output for game.

Action Plan (with allocations to each team member):

prepare for sprint 5 – to be set up on next meeting.

- Aisling – updating SOPGame
- Andrew – finalise sequence diagrams.
- Joel – Additional tests for SOPGame
- Leonie – Continue with report structure, make copy available to work on.
- Nathaniel – Game guide, Record minutes.
- ALL – draw down current updated files locally and identify any issues.

Date and Location: 18.4.24 – Teams

Meeting Objectives: Set up sprint 5, discuss reporting

Attendees: Nathaniel Nimmock, Joel Egerton, Leonie Robinson, Andrew Morrison, Aisling Broder-Rodgers

Apologies: N/A

Agenda: Set up sprint 5, discuss reporting

Discussions (challenges, and progress by each team member on previous tasks):

- Discussed general progress on the assignment and what is outstanding.
- Discussed how we would approach recording the video requirement.
- Refinement of basic structure of group report, made available – Leonie.
- Peer review started; scores given by each member declaration provided- All.
- Issues and potential improvements identified discussed –
 - Delay of text appearing as turns are played.
 - Formatting of Enum fields to title case.
 - Other general cosmetic improvements
- Reviewed update to SOPGame class – few changes left but otherwise ready to merge – Aisling.

- Additional testing for SOPGame class reviewed. Tests to be updated in line with final changes to SOPGame class – Joel.
- Updated sequence diagrams reviewed – Andrew.
- Reviewed progress on game guide – needs updated to reflect current changes to interface – Nathaniel.
- Final items from backlog added to sprint 5 and assigned.

Action Plan (with allocations to each team member):

- Aisling – final changes to SOPGame, enhance user interface.
- Andrew – finalise sequence diagrams, enhance user interface, find potential optimisations in code.
- Joel – complete Additional tests for SOPGame, implement end game conditions.
- Leonie – Conduct overall testing and debugging.
- Nathaniel – Complete Game guide, Record minutes.
- ALL – draw down current updated files locally and identify any issues.
- ALL – see what can be added to the report.

Appendix 4 – User Guide



Eco Venture User guide

Contents

[Gameboard](#)

[Gameplay, rules, and interface](#)


[Creators](#)



Gameboard

The game is text based but in fact utilises a game board, below you can see how it looks.

You can create your own pieces to move around this page as you play if you like.

Hydro Power Renewable Energy Initiative Price: 95 KP Minor Dev: 25 KP Major Dev: 75 KP	Organic Farming Sustainable Agriculture Price: 50 KP Minor Dev: 45 KP Major Dev: 65 KP	Agroforestry Sustainable Agriculture Price: 60 KP Minor Dev: 55 KP Major Dev: 70 KP	<i>Carbon Neutral!</i>
Wind Power Renewable Energy Initiative Price: 90 KP Minor Dev: 20 KP Major Dev: 80 KP	 <p>EcoVenture</p> <p><i>Invest in and develop environmentally friendly schemes to contribute to the sustainability of the planet!</i></p> <p>ECO VENTURE</p>		Recycling Waste Management Price: 130 KP Minor Dev: 65 KP Major Dev: 100 KP
Solar Power Renewable Energy Initiative Price: 100 KP Minor Dev: 30 KP Major Dev: 90 KP			Composting Waste Management Price: 120 KP Minor Dev: 55 KP Major Dev: 95 KP
<i>The Green Beginning!</i> Collect 200 KP!	Climate Resilience Planning Climate Change Price: 275 KP Minor Dev: 120 KP Major Dev: 180 KP	Carbon Capture & Storage Climate Change Price: 250 KP Minor Dev: 100 KP Major Dev: 160 KP	Circular Economy Waste Management Price: 125 KP Minor Dev: 125 KP Major Dev: 107 KP

Gameplay, rules, and interface

Overview

EcoVenture is a virtual board game with a distinctive theme centred around saving our planet. Players take on the role of eco-entrepreneurs, striving to invest in and develop environmentally friendly schemes to contribute to the sustainability of the planet. Players use scientific knowledge to address environmental challenges by getting involved in their own environmental initiatives, or contributing to another player's initiative, and use this knowledge to progress in the game through minor and major developments with the aim of achieving sustainability.

Getting started:

This is a text game played through a terminal, you can use an integrated development environment (IDE) such as Eclipse, this is recommended, or you can open a terminal (you will need to compile the files manually).

You begin by choosing the amount of players between 2 and 4 people may play.

You cannot proceed with one player.

```
Welcome to the Save Our Planet Board Game.  
Let's get this Game Started  
How many players are there? Please enter a number between 2 and 4:
```

When the amount of players has been chosen, proceed with entering player names when prompted.

- The same player's name cannot be entered twice.
- Player name must be between 2 and 40 characters.

```
Setting up the game for 2 players...  
  
Please enter the name of each player, one at a time.  
Enter name for Player [1]: Rob  
Enter name for Player [2]: NN
```

The game begins...

Gameplay:

- Note you can choose to stop the game before completion by pressing 'N' twice.
- To proceed with your turn, Choose 'Y'.

```
ROUND 1  
*****  
  
Are You Ready to Play?  
Entering 'N' twice will exit the game.  
  
Enter your choice (Y/N):
```

When the game begins, each player will be prompted to take their turn, keep an eye out for your name!

```

Nn would you like to roll the dice?
If you enter 'N' twice, game play will terminate for all players.

**=====**
Player : Nn - Knowledge Points : 200.00
**=====**

Your response (Y/N):

```

Each player begins with 200 Knowledge points on 'The Green Beginning' square.

Your actions will either increase or decrease your knowledge points as you play.

The first person to gain 1000 Knowledge points wins.

Or the person with the most Knowledge points at the end of 15 rounds wins.

During your turn two dice are rolled, the total represents the number of places you will move.

Each time you land on or pass 'The Green Beginning' square you will receive another 200 Knowledge points.

Keep an eye out for the "Carbon Neutral" square, this is a neutral zone where nothing happens, you effectively skip your go!... but the air quality is great!

```

Moving: . . . . .
You have landed on the square Carbon Neutral of type Neutral. at position 6
> This square is not available for purchase.
> No Action Required.

```

The rest of the squares are split into fields, within the fields are environmental initiatives you can choose to invest your knowledge (points) into!

Renewable Energy Initiatives:

- **Solar Power:** Harnessing energy from the sun using solar panels.
- **Wind Energy:** Generating electricity from wind turbines.
- **Hydropower:** Utilizing the energy of flowing water to generate power.

Sustainable Agriculture Practices:

- **Organic Farming:** Cultivating crops without synthetic pesticides and fertilizers.
- **Agroforestry:** Integrating trees and shrubs into agricultural landscapes for environmental and economic benefits.

Waste Management and Recycling Programs:

- **Recycling:** Collecting and processing materials for reuse, reducing the demand for new resources.
- **Composting:** Decomposing organic waste to create nutrient-rich soil.
- **Circular Economy:** Designing products with a focus on reuse, repair, and recycling to minimize waste.

Climate Change Mitigation and Adaptation Strategies:

- **Carbon Capture and Storage (CCS):** Capturing carbon dioxide emissions from industrial processes and power plants.
- **Climate Resilience Planning:** Developing strategies to adapt to the impacts of climate change, such as rising sea levels and extreme weather events.

When you land on one of these you will be prompted to act!

```

Rolling dice...
Moving Piece 4 places...
You have landed on the square Organic Farming at position 4 which belongs to the field of Sustainable Agriculture.

This square is currently unowned.
> The cost to buy this square is 50.00 Knowledge Points and initial rental charge will be 10.00 Knowledge Points.
> The cost to build a minor development will be 45.00 Knowledge Points.
> The cost to build a major development will be 65.00 Knowledge Points.
Nn, you have 200.00 Knowledge Points. Would you like to buy this square for 50.00 Knowledge Points? (Y/N):

```

Each square can be developed - You must first buy the square before you can develop upon it!

- A player must own all squares in a field before they can begin development.

If you choose to buy the square this will be recorded, and your knowledge points will be updated!

```

Congratulations Nn you now own Organic Farming

**=====**
Player : Nn - Knowledge Points : 150.00
**=====**

```

If another player lands on this square they must now pay rent to remain there until their next turn!

Both players knowledge points will be updated.

```

Rolling dice...
Moving Piece 4 places...
You have landed on the square Organic Farming at position 4 which belongs to the field of Sustainable Agriculture.
> This square is owned by Nn.
> There are currently 0 minor developments and 0 major developments on this square.
> The rent to stay on this square is 10.00 Knowledge Points.

Transferring 10.00 from Rob to Nn.

**=====**
Player : Rob - Knowledge Points : 190.00
**=====**

**=====**
Player : Nn - Knowledge Points : 160.00
**=====**

```

Beware, if the rent paid takes your knowledge points below 0, the game will end, whoever has the most knowledge points at the time wins!

```

Moving: . . . . .
You have landed on the square Solar Power at position 1 which belongs to the field of Renewable Energy Initiatives.
> This square is owned by Bob.
> There are currently 0 minor developments and 0 major developments on this square.
> The rent to stay on this square is 500.00 Knowledge Points.
Rent required is 500.0, but you only have 448.75
You have insufficient funds to proceed - gameplay terminated.

```

If you do not have enough knowledge points to buy a square you will be informed, and this will then be offered to the other player(s).

```

You have landed on the square Carbon Capture & Storage at position 10 which belongs to the field of Climate Change.

This square is currently unowned.
> The cost to buy this square is 250.00 Knowledge Points and initial rental charge will be 40.00 Knowledge Points.
> The cost to build a minor development will be 100.00 Knowledge Points.
> The cost to build a major development will be 160.00 Knowledge Points.
Rob, you have 190.00 Knowledge Points, unfortunately you do not have sufficient funds to buy this square.

Carbon Capture & Storage belonging to Climate Change will now be offered to other players.
Nn would you like to purchase this square?

```

If you have enough knowledge points but simply do not wish to buy the square, this will then be offered to the other player(s) also.

```
Nn, you have 360.00 Knowledge Points. Would you like to buy this square for 125.00 Knowledge Points? (Y/N): n
Please confirm by typing 'N' again: n

Circular Economy belonging to Waste Management will now be offered to other players.

Rob would you like to purchase this square?
Rob, you have 190.00 Knowledge Points. Would you like to buy this square for 125.00 Knowledge Points? (Y/N):
```

At the end of each round a summary will be displayed, showing players knowledge points and the current purchase status for squares within their respective fields.

***** CURRENT PLAYER STATS *****	***** CURRENT PLAYER STATS *****
<pre> **=====** Player : Nn - Knowledge Points : 360.00 **=====** **=====** Player : Rob - Knowledge Points : 190.00 **=====** </pre>	<pre> **=====** Player : Nn - Knowledge Points : 360.00 **=====** **=====** Player : Rob - Knowledge Points : 145.00 **=====** </pre>
***** CURRENT SQUARE STATS *****	***** CURRENT SQUARE STATS *****
<pre> Field: Waste Management Owners: - Square Currently Unowned - Square Currently Unowned - Square Currently Unowned Field: Sustainable Agriculture Owners: - Nn - Square Currently Unowned Field: Climate Change Owners: - Square Currently Unowned - Square Currently Unowned Field: Renewable Energy Initiatives Owners: - Square Currently Unowned - Square Currently Unowned - Square Currently Unowned </pre>	<pre> Field: Waste Management Owners: - Square Currently Unowned - Rob - Rob Field: Sustainable Agriculture Owners: - Nn - Square Currently Unowned Field: Climate Change Owners: - Square Currently Unowned - Square Currently Unowned Field: Renewable Energy Initiatives Owners: - Square Currently Unowned - Square Currently Unowned - Square Currently Unowned </pre>

When all squares are owned the owning player will be prompted to create a development –

```

You have landed on the square Agroforestry at position 5 which belongs to the field of Sustainable Agriculture.

This square is currently unowned.
> The cost to buy this square is 60.00 Knowledge Points and initial rental charge will be 12.00 Knowledge Points.
> The cost to build a minor development will be 55.00 Knowledge Points.
> The cost to build a major development will be 70.00 Knowledge Points.
Nn, you have 748.00 Knowledge Points. Would you like to buy this square for 60.00 Knowledge Points? (Y/N): y

Congratulations Nn you now own Agroforestry

**=====**
Player : Nn - Knowledge Points : 688.00
**=====**

Would you like to develop your owned areas in the field : Sustainable Agriculture? (Y/N)
```

```

Would you like to purchase a minor development for Organic Farming?
This will cost 45.00 knowledge points. You currently have 688.00 knowledge points.
Enter 'Y' or 'N':
```

If you choose yes, you will be informed of your success.

```

Y
Successfully added Development to Organic Farming
There is now 1 Minor Development on Organic Farming
```

If you choose no, your turn will ned.

```

N
No development carried out.
```

The rent cost of landing on each owned square in each field is modified as developments are added.

```
You have landed on the square Agroforestry at position 5 which belongs to the field of Sustainable Agriculture.
> This square is owned by Nn.
> There are currently 2 minor developments and 0 major developments on this square.
> The rent to stay on this square is 13.23 Knowledge Points.

Transferring 13.23 from Rob to Nn.

**=====**
Player : Rob - Knowledge Points : 368.77
**=====**

**=====**
Player : Nn - Knowledge Points : 311.23
**=====**
```

When all minor developments in a field are purchased, you will be prompted to add a major development.

```
You have landed on the square Agroforestry at position 5 which belongs to the field of Sustainable Agriculture
> This square is owned by Nn.
> There are currently 2 minor developments and 0 major developments on this square.
> The rent to stay on this square is 13.23 Knowledge Points.
You already own this square. Hang here until your next roll

Would you like to develop your owned areas in the field : Sustainable Agriculture? (Y/N) y

Would you like to purchase a Major development for Organic Farming?
This will cost 65.00 knowledge points. You currently have 311.23 knowledge points.
Enter 'Y' or 'N':
```

If you choose yes, you will be informed of your success.

```
Enter 'Y' or 'N': y
Successfully added Major Development to Organic Farming
```

If you choose no, your turn will ned.

```
Enter 'Y' or 'N': n
No development carried out on Circular Economy
```

If you have purchased all possible developments in a field, you will be informed.

```
Would you like to develop your owned areas in the field : Sustainable Agriculture? (Y/N) y
Maximum amount of Developments reached.
```

When winning conditions are met (first to 1000 KP or highest KP when 15 rounds are completed) the final game stats will be presented

Note there are only 3 winning positions, if a 4th player is present, they will not qualify and will be informed.

```
***** CURRENT PLAYER STATS *****

**=====**
Player : Nn - Knowledge Points : 107.44
**=====**

**=====**
Player : Rob - Knowledge Points : 30.56
**=====**

***** CURRENT SQUARE STATS *****

Field: Waste Management
Owners:
- Rob
- Rob
- Rob

Field: Sustainable Agriculture
Owners:
- Nn
- Nn

Field: Climate Change
Owners:
- Nn
- Nn

Field: Renewable Energy Initiatives
Owners:
- Square Currently Unowned
- Square Currently Unowned
- Nn

***** FINAL LEADERBOARD *****

In 1st Place: Nn with 107.44 Knowledge Points
In 2nd Place: Rob with 30.56 Knowledge Points
```


Should 2 or more player finish with the same number of points they will be displayed alphabetically in the leaderboard.

```
Setting up the game for 4 players...
```

```
Please enter the name of each player, one at a time.
```

```
Enter name for Player [1]: Ad
```

```
Enter name for Player [2]: Ac
```

```
Enter name for Player [3]: Ab
```

```
Enter name for Player [4]: Aa
```

```
***** FINAL LEADERBOARD *****
```

```
In 1st Place: Aa with 200.00 Knowledge Points
```

```
In 2nd Place: Ab with 200.00 Knowledge Points
```

```
In 3rd Place: Ac with 200.00 Knowledge Points
```

```
Unfortunately, Ad didn't place with 200.00 Knowledge Points
```



Creators

Aisling Broder-Rodgers, Andrew Morrison, Joel Egerton, Leonie Robinson, Nathaniel Nimmock