BitDash

Alasdair Cross, Cameron Roberts, Danny Demarco, Shuai Ning, Xiaoyan Zhou, Xuan Wang, Ye Zhu, and Zhiyuan Tan

Master of Science in Computer science

University of Bath

Dec 2020

Abstract

This report would focus on the agile development life cycle of “BitDash”. “BitDash” is a dungeon parkour game developed with Unity whose main character is a ninja and runs in infinite dungeons to avoid monsters and traps. The Agile development life cycle has been used in the project is Scrum and the process has strictly followed the rules of agile.

The development life cycle has last for two months and contains three phases which are pre-game phase, game phase and post-game phase. The document is organized by sprint. In the pre-game phase, plans and architectures such as game proposal and role assignment etc. were decided and all the documents in this phase have been attached in the Appendix A (see Page 113). The game phase contains five sprints, and each sprint lasts for one week. The documents and records for the sprints have been well recorded. For each sprint, it contains Overview, Review, Customer Meeting Record, Daily Scrum Meeting, Backlog, CRC Card, User Story, Test for User Story, User Case, Test for User Case, User Interface, and Exception Handling sections. Sprint 6 has been assigned as the post-game phase and last for five days for testing and produce the final product. Tools such as Jira and Miro Online White Board have been used for guaranteeing and associating with the Agile process. Roadmaps and white boards generated have also been attached in the Appendix B (see Page 125). The quality of the product has been guaranteed by the integration test in post-game phase. The link of BitDash Repository as following: https://github.bath.ac.uk/BitDash/BitDash

Table of Contents

Acknowledgments

We, group Pink, would like to thank our Lecturer Julian for his efforts as being our customer, and for his great guidance and suggestions for everything. Also, thanks to TAs, they have offered great advice on documentation and so on. Lastly, thank all 8 team members for sparing no effort on this project and make it done and happen.

Sprint 0

0.1 Sprint 0 Overview

In this sprint, team members use the analogy system and brainstorm among team members. We confirmed the game plan. This project is a parkour game with a ninja theme. With reference to the existing system, the focus of the work of the "Super Mario" team is to determine the development tools and development process. The game will be developed using Unity 2D, and the code will be saved via GitHub. Team members communicate with Miro through the team. The files are stored in Jira. The responsibilities of the team members have been assigned, and the team roles are assigned as follows. This role assignment will be used for development in all subsequent sprints.

Role Assignments		
Development Team	Product Owner	Xiaoyan Zhou
	Scrum master	Shuai Ning
		Xuan Wang
		Ye Zhu
		Zhiyuan Tan
	Team member	Alasdair Cross
		Cameron Roberts
		Danny Demarco

Everyone in the team acts as a development team.

As the first sprint, we first implemented the most basic function, which is to open and close the game. According to the logic of the user story, after opening the game, you can see the game background and game role model. The user will also hear the background music of the game. Considering that the game character will move in the game scene later, we set the viewing angle of the game to a fixed lens. After the function is implemented, further tests are performed according to the use cases and user cases, and the test results show that there are no errors. No abnormal situation was encountered in this link. Team members confirmed the backlog of frames at daily meetings. In the next Sprint, the development of game character movement will be carried out.

0.2 Sprint 0 Review

THE 1ST REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-0 Approval of Management Plan		
Have all the Project Activities been well defined?	√	
Has Mission Need Statement been well prepared and explained technically and functionally?	√	
Have a Meeting Session Arrangement well arranged?	√	
Progress Monitor Function is defined clearly? Progress Reviews?	√	
Have the tools for development been well listed and declared?	√	
The Feasibility?	√	
CD-1 Approval of QA		
All quality assurances (for text file as well as software) are well defined?	√	
Review actions for quality assurance are well defined?	√	
A Good definition of Coding and Text file style?	√	
The Feasibility?	√	
CD-2 Approval of Reviews		

Has the review session been well defined?	√	
Has a full review circle been well arranged?	√	
Has the review rules and details been well contained?	√	
CD-3 Approval of Version Control		
The tools that are going to be used are well declared?	√	
The ways to keep different kinds of files are well declared?	√	
Does the plan clearly arrange the development productions?	√	
Does the plan avoid the risky and insecurity in development?	√	
CD-4 Software tests, fault and bug reporting		
Contains clearly testing methods?	√	
Contains clearly debugging reviews?	√	
Contains clearly bug report sessions?	√	
Whether the report keeping actions and managements well?	√	
Comments: Well structured **Reviewer: Shuai Ning**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-5 Approval of 1st Review Meeting		
Have an action list made after the session?	√	

Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: The design of the brainstorming game proposal by group members has been completed **Reviewer: Shuai Ning** **Date: 27th October 2020**		
THE 2ND REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-6 Approval of Development Progress Comparison		
A detailed comparison is made?	√	
An overall evaluation is made?	√	
A list between planned progress and real progress is made?	√	
A discus session on the solutions?	√	
A well-done record of the session?	√	
Comments: The design of the brainstorming game proposal by group members has been completed **Reviewer: Shuai Ning**		
CD-7 Approval of Development Questions		

Have actual questions reported clearly?	√	
Have all problems solved?	√	
Comments: The background and character model of the game design have been summarised **All Details of the Problems and Solutions should be recorded below:** **Problems:** None **Solutions:** None **Reviewer: Shuai Ning**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-8 Approval of 2nd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting	√	
Have all missions be defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: The team members’ design was passed without problems. **Reviewer: Shuai Ning** **Date: 30th OCT 2020**		
THE 3rd REVIEW APPROVAL		
KEY QUESTIONS FOR FINAL EVALUATION	**YES**	**NO**
CD-9 Approval of Final Evaluation		

Whether the sprint is full completeness?	√	
Full documents recorded?	√	
The sprint finished in good condition?	√	
A Final Evaluation?	√	
Comments: The sprint process has been completed. **Reviewer: Shuai Ning**		
CD-10 Approval of Debugging		
Has a final test passed?	√	
Have a Debugging track recorded?	√	
All bugs fixed?	√	
Comments: The characters and perspectives in the game have been adjusted, and now members of the game team can directly see them **Reviewer: Shuai Ning**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-11 Approval of 3rd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	

Have all missions in this meeting well finished?	√	
Comments: All missions were well finished, and everyone knows what will happen next sprint. **Reviewer: Shuai Ning**		
Sprint 0 Summary: In Sprint0, it is mainly the design of game development tools and game documents. Team members formed preliminary ideas through brainstorming and talked with customers to determine how to implement them. The game is designed as a parkour game with a Beijing story as a ninja theme. In this sprint, the focus is on the functions of entering and exiting the game. We designed the game background and the character model of the game. The game perspective lens is also fixed.		
Date: Nov 1th 2020		

0.3 Sprint 0 Customer Meeting Record

Customer Meeting Record		
Client Name: Julian **Project Start Date**: 25.10.2020		**Project Title**: BitDash **Project Deadline**: 11.12.2020
Scrum no: 0	**Meeting Scheduled at:** 12:30-13:00	**Date**: 23.10.2020
Of 3 separate game proposals, the customer showed keen interest in the parkour game proposal. It is a survival, horizontal and 2D game that allows user to run through the dungeon. The preferred control is by mouse click or simple one key click. More elements are going to be added during development.		
Analysation		
The development team will use "Super Mario" as the same type of system for comparison. Four modules were assumed in the first version of the game. The four modules in this sprint are just a fuzzy concept of system functions. In the actual development process, the types and specific functions of the function modules may change. The user enters the game through the user interface (user module). In the game, the user controls the game character to run forward, he can jump and sprint as well (motion module) to avoid obstacles. In the game, the user will encounter walls, traps and monsters to stop the user from running (obstacle module). The user runs until blocked, and the user score is displayed at the end of the game (score module).		

0.4 Sprint 0 Daily Scrum Meeting Record

Daily Scrum Meeting Record **Sprint 0: 26.10.2020 - 1.11.2020**				
Date	**Time**	**Members**	**Assigned Task**	**Summary**
26th Oct	12:00-12:30	All team members	Design sprint process and analysing tool.	Tools and documentation are discussed.
27th Oct	12:00-12:30	All team members	Function classification of motion module.	Designed the game proposal and selected the most priority function as the first sprint content.
28th Oct	12:00-12:30	All team members	Develop the function of starting and exiting the game.	All members have progressed without exception.
29th Oct	10:00-10.30	All team members	Set the game character model, game background and background music.	All members have progressed without exception.
30th Oct	13:00-13:30	All team members	Test by user story.	According to the user story without exception.
31st Oct	13:00-13:30	All team members	Test by use case.	According to the use case test without exception.
1st Nov	13:00-13:00	All team members	Sprint review and improve documentation.	Resolve functional conflicts and complete review and other documents.

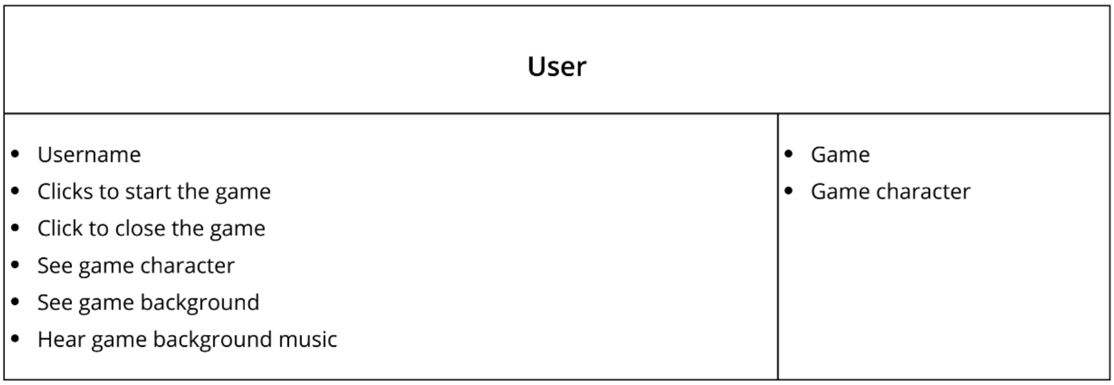
0.5 Sprint 0 Backlog

Sprint 0	**Item**	**Description**	**Priority**
Sprint backlog	Start the game	As a user, I can start the game 1.Fixed camera lens 2.Show game character model 3.Show game background 4.Start game button 5.Game background music	1

	Exit the game	As a user, I can quit the game 1.Quit game button	1
Product backlog	Game character run		
	Game character jump		
	Game character dash		

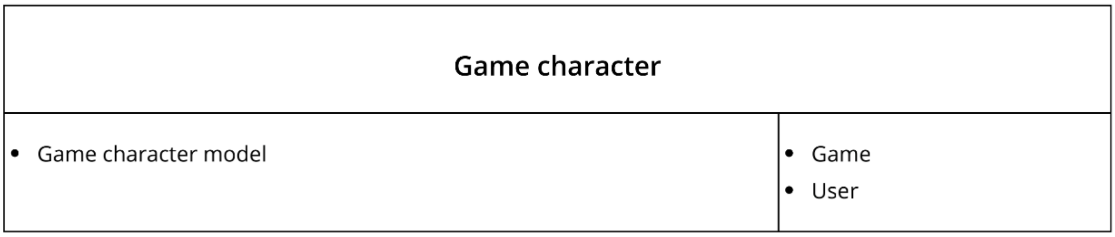
0.6 Sprint 0 CRC Cards

0.6.1 User



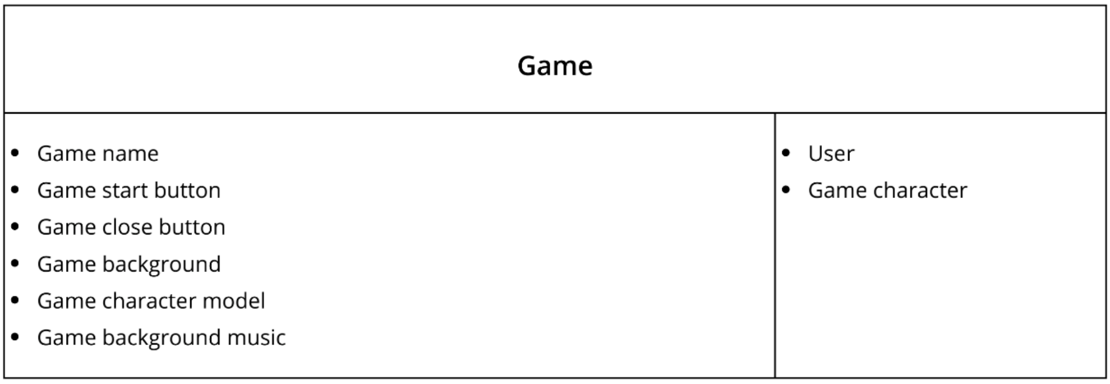
User	
• Username • Clicks to start the game • Click to close the game • See game character • See game background • Hear game background music	• Game • Game character

0.6.2 Game Character



Game character	
• Game character model	• Game • User

0.6.3 Game



Game	
• Game name • Game start button • Game close button • Game background • Game character model • Game background music	• User • Game character

0.7 Sprint 0 User Story

0.7.1 Start the Game

Title	Start the game
Story description	As a user I can start the game.
Rule description	User: Player game: Dungeon game
Testing standard	1.When the user clicks the start game button, the user can start the game. 2.The user can see the game background and game role model. 3.Users can hear game background music.
Design plan	Add button and onclick to call start function.
Checking list	Whether the branch in GitHub is start feature branch. The unity version is 2019.4.13f1c1.

0.7.2 Quit the Game

Title	Quit the game
Story description	As a user, I can quit the game.
Rule description	None
Testing standard	1.When the user clicks the exit game button, the game interface is closed, and the user returns to the computer desktop. 2.When the user closes the game, the user can exit the game.
Design plan	Add button and click to call quit function.
Checking list	Whether the branch in GitHub is quit feature branch. The unity version is 2019.4.13f1c1.

0.8 Tests for User Story

Test items	Case	When	Then	Result
Start the game	1	The user clicks the start game button.	The user can start the game.	TRUE
	2	When user enter the game.	The user can see the game background and game role model.	TRUE
	3	When the user enters the game.	Users can hear game background music.	TRUE
Quit game	1	When the user clicks the exit game button.	The game interface is closed, and the user returns to the computer desktop.	TRUE
	2	When the user closes the game.	The user can exit the game.	TRUE

0.9 Sprint 0 Use Case

Use case	Start the game
Summary	User enters the game
Actor	User (anyone playing the game)
Trigger	User opens the software
Primary scenario	1.The user clicks to enter the game [alternative scenario: The user closes the game interface]. 2.The user clicks to start the game after entering the game. [alternative scenario: User clicks to exit the game]. 3.After the user starts the game, he can see the game background and game role model. Users can hear game background music. [Exceptional scenario: the user cannot see the game background and game role model].
Alternative scenario	If user closes the game interface 1.Users can leave the game directly. [alternative scenario: The user opens the game again]. If user clicks to exit the game

	1.The user exits the game and returns to the desktop [alternative scenario: The user opens the game again]. If the user opens the game again 1.Return to primarily scenario 1.
Exceptional scenario	The user cannot see the game background and game role model. Exit the game and return to primarily scenario 1.
Pre-conditions	None
Post-conditions	Game character movement
Assumptions	None

0.10 Tests for Use Case

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user clicks to enter the game.	F	F
	2	The user clicks to start the game after entering the game.	F	F
	3	After the user starts the game, he can see the game background and game role model. Users can hear game background music.	F	F
Alternative scenario	1	If user closes the game interface.	F	F
	2	If user clicks to exit the game.	F	F
	3	If the user opens the game again.	F	F

0.11 User Interface

During this sprint we implemented a main menu that the user can interact with when the game loads up. We wanted to take a very minimalist approach towards the user interface so the user can get straight into the action of the game. Keeping the main menu simple we have to main functions, start and quit. Start initiates the game for the user getting straight to the action, meanwhile quit is a system exit for the user. This keeps the same design philosophy of the dark colours which we contrasted with white and grey so it can easily be read. When the start button is pressed, we immediately take control of our ninja character and the score starts increasing after a few seconds.

Our start menu shows bellow (Figure 1):

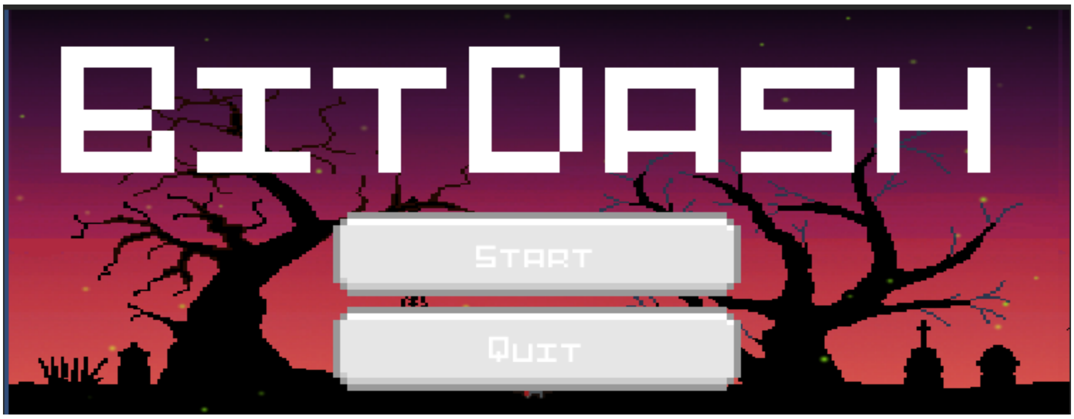


Figure 1: Start menu of BitDash

Getting into action after starting (Figure 2):

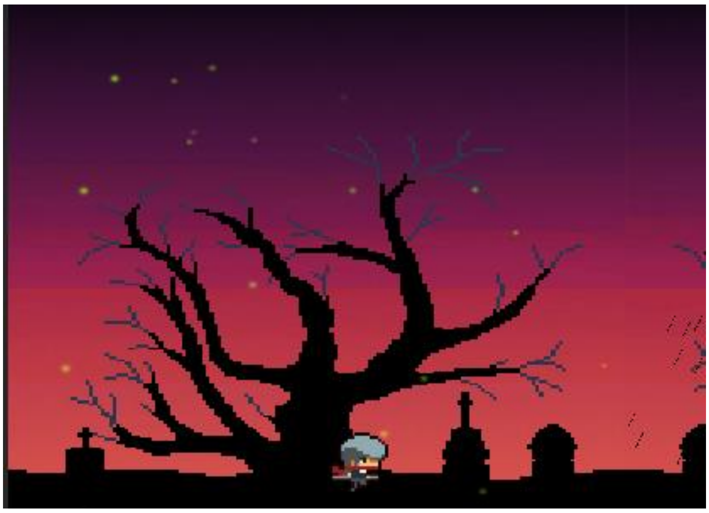


Figure 2: Action after starting

0.12 Sprint 0 Exception Handling

Exception	Time of occurrence	Cause of the exception	Solution
-	-	-	-

Sprint 1

1.1 Sprint 1 Overview

In the case that sprint0 has completed the basic settings of the game, in order to enable players to actually play the game. We set up the motion module. Among them, there are three functions of atmosphere running, jumping and dash. Running is the most basic function, and its priority is above jumping and dash. Team members first develop the function of running. After that, jump and dash functions are implemented together. When implementing the function, multiple members reported that there would be redundant files when working with brunch. At the same time, it was prone to errors for some people who came into contact with GitHub for the first time. When completing the code implementation this week, we lost a lot of time due to processing GitHub files. Finally, the team members used the ignore file processing method. Solved this problem permanently.

The development process went smoothly. The design of basic functions is perfect. But in the test, I encountered a very difficult problem. Because the two functions use the same function. So, the functions will be parallel. Finally, the problem was solved by the transfer of setting parameters. The lesson the team members learned from this is that just testing the logic is not enough, it must be integrated and tested as a whole.

1.2 Sprint 1 Review

THE 1ST REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-0 Approval of Management Plan		
Have all the Project Activities been well defined?	√	
Has Mission Need Statement been well prepared and explained technically and functionally?	√	
Have a Meeting Session Arrangement well arranged?	√	

Progress Monitor Function is defined clearly? Progress Reviews?	√	
Have the tools for development been well listed and declared?	√	
The Feasibility?	√	
CD-1 Approval of QA		
All quality assurances (for text file as well as software) are well defined?	√	
Review actions for quality assurance are well defined?	√	
A Good definition of Coding and Text file style?	√	
The Feasibility?	√	
CD-2 Approval of Reviews		
Has the review session been well defined?	√	
Has a full review circle been well arranged?	√	
Has the review rules and details been well contained?	√	
CD-3 Approval of Version Control		
The tools that are going to be used are well declared?	√	
The ways to keep different kinds of files are well declared?	√	

Does the plan clearly arrange the development productions?	√	
Does the plan avoid the risky and insecurity in development?	√	
CD-4 Software tests, fault and bug reporting		
Contains clearly testing methods?	√	
Contains clearly debugging reviews?	√	
Contains clearly bug report sessions?	√	
Whether the report keeping actions and managements well?	√	
Comments: Well structured **Reviewer: Ye Zhu**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-5 Approval of 1st Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	

Comments: Reasonable task allocation. High feasibility. Well done guys **Reviewer: Ye Zhu** **Date: Nov 2nd 2020**

THE 2ND REVIEW APPROVAL

KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	YES	NO
CD-6 Approval of Development Progress Comparison		
A detailed comparison is made?	√	
An overall evaluation is made?	√	
A list between planned progress and real progress is made?	√	
A discus session on the solutions?	√	
A well-done record of the session?	√	
Comments: Everyone is completing the work step by step. It seems that there is no need to delay or re-allocate tasks. **Reviewer: Ye Zhu**		
CD-7 Approval of Development Questions		
Have actual questions reported clearly?	√	
Have all problems solved?	√	

Comments:

Keep a backup of your code before doing any merge or test work.

All Details of the Problems and Solutions should be recorded below:

Problems:

Error when merging into main method occur on Nov 1st.

Solutions:

Go back to the previous version and rewrite the code.

Reviewer: Ye Zhu

KEY QUESTIONS FOR MEETING REVIEW	YES	NO
CD-8 Approval of 2nd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	

Comments:

Nice work

Reviewer: Ye Zhu

Date: Nov 5th 2020

THE 3rd REVIEW APPROVAL

KEY QUESTIONS FOR FINAL EVALUATION	YES	NO
CD-9 Approval of Final Evaluation		

Whether the sprint is full completeness?	√	
Full documents recorded?	√	
The sprint finished in good condition?	√	
A Final Evaluation?	√	
Comments: The sprint is full completeness. Good. **Reviewer: Ye Zhu**		
CD-10 Approval of Debugging		
Has a final test passed?	√	
Have a Debugging track recorded?	√	
All bugs fixed?	√	
Comments: All bugs fixed and ready for the next sprint to add more functionalities. **Reviewer: Ye Zhu**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-11 Approval of 3rd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	

Have all missions in this meeting well finished?	√	
Comments: All missions were well finished, and everyone knows what will happen next sprint. **Reviewer: Ye Zhu**		
Sprint 1 Summary: During the sprint 1, everyone completed the assigned tasks on time. The items in our Sprint backlog which is “Run”. “Jump” and “Dash” has been well delivered and tested. All problems encountered were reported and resolved in time. All documents have been completed and viewed by scrum masters. The preparation for the next Sprint has been delivered by the product owner. All members have reasonable work to do in the sprint 2. Busy time periods are notified, and work assignment might be increased or decreased reasonably to ensure fairness.		
Date: Nov 8th 2020		

1.3 Sprint 1 Customer Meeting Record

Customer Meeting Record		
Client Name: Julian **Project Start Date**: 25.10.2020		**Project Title**: BitDash **Project Deadline**: 11.12.2020
Scrum no: 1	**Meeting Scheduled at:** 12:15-12:45	**Date**: 06.11.2020
Customer required specific proposals on how difficulty would be added into the game, and other elements that can add enjoyment to the players experience. He also asked for user stories, and affirmed the significance of art resources, which could make the game better. Additionally, he requested the game to have more dimensions of difficulty.		
Analysation		
The team will add more specific elements, such as different kinds of traps, monsters or obstacles to change the difficulty level, and will try to create more art resources fit to the current game style. More user cases will be added to the documentation, to give a clear description on user stories.		

1.4 Sprint 1 Daily Scrum Meeting

Daily Scrum Meeting Record

Sprint 1: 02.11.2020 - 08.11.2020

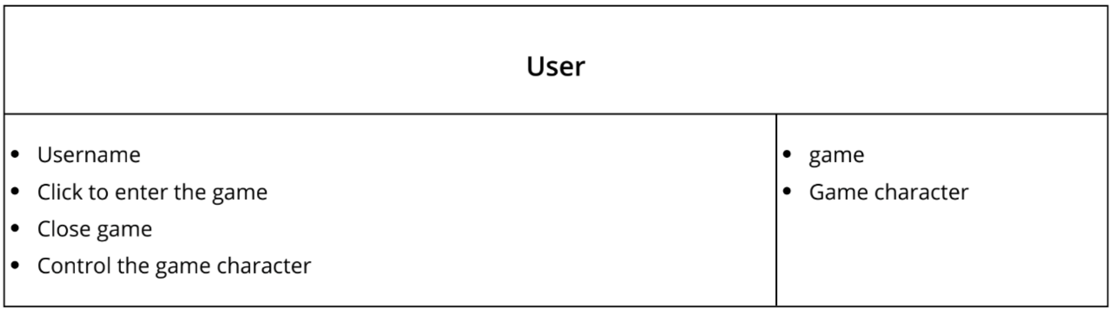
Date	Time	Members	Assigned Task	Summary
2nd Nov	15:00 - 15:30	All team members	Function classification of motion module.	The specific functions of the motion module are divided into three parts: running, jumping and dash.
3rd Nov	15:00-15:30	All team members	Establish a corresponding Unity branch and assign functions to specific members.	Write use cases and user stories and then focus on designing how to implement functions. Encountered an exception: A branch in unity is damaged.
4th Nov	15:15-15:45	All team members	Realize the functions of running, jumping and dash.	All members have progressed without exception.
5th Nov	10:00-10:30	All team members	Realize the functions of running, jumping and dash.	All members have progressed without exception.
6th Nov	13:00-13:30	All team members	Test by user story.	According to the use case test without exception.
7th Nov	13:00-13:30	All team members	Test by use case.	When testing according to the use case, the jump conflicts with the dash function. Encountered an exception: Two members work in the same branch.
8th Nov	13:00-13:30	All team members	Sprint review and improve documentation.	Resolve functional conflicts and complete review and other documents.

1.5 Sprint 1 Backlog

Sprint 1	Item	Description	Priority
Sprint backlog	Game character run	As a user, I can control the game character to run forward. 1.Game character running animation.	1
	Game character jump	As a user, I can control the game character to jump. 1.Game character jumping animation.	2
	Game character dash	As a user, I can control the game character dash. 1.Game character dashing animation.	2
Product backlog	Prefab walls		
	Prefab traps		
	Prefab monsters		
	Game character death		
	game score		

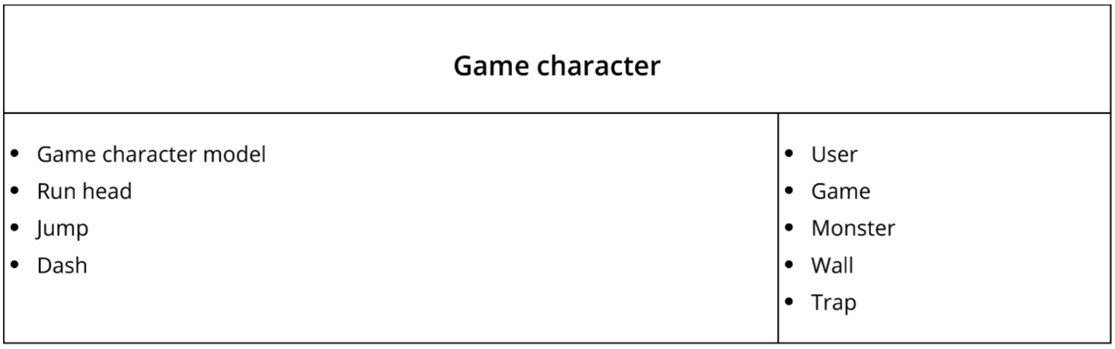
1.6 Sprint 1 CRC Cards

1.6.1 User



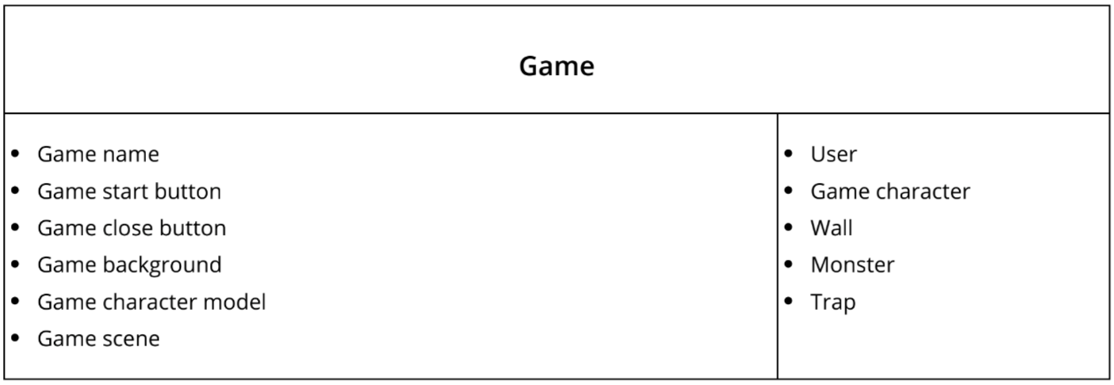
User	
• Username • Click to enter the game • Close game • Control the game character	• game • Game character

1.6.2 Game Character



Game character	
• Game character model • Run head • Jump • Dash	• User • Game • Monster • Wall • Trap

1.6.3 Game



Game	
• Game name • Game start button • Game close button • Game background • Game character model • Game scene	• User • Game character • Wall • Monster • Trap

1.7 Sprint 1 User Story

1.7.1 Game Character Movement

Title	Game character movement
Story description	As a user, I can control the movement of the game character to get points.
Rule description	User: Player Game character: the character model manipulated by the user in the game. Move: the user's control of the game character in the game. Scene: All actions of the game character are carried out in the scene. Plane: The current position of the game character (when the game character stands on an obstacle, the obstacle is equivalent to a plane).

	Running distance: the distance the user moves on the plane. Score: It increases with the length of time the game character runs in the game and is used to evaluate the user's game.
Testing standard	The game character can move in the scene.
Design plan	None
Checking list	None

1.7.2 Game Character Run

Title	Game character run
Story description	As a game character, I can run forward in the game scene.
Rule description	Run: the user controls the game character to move forward.
Testing standard	On the plane 1.The game character runs forward
Design plan	Set the running state, Boolean run. When running, add fixed speed, add smooth material to the character and the ground. Add the box collider to complete the collision. The effect of moving is achieved by calling the run state in the update function.
Checking list	Whether the branch in GitHub is run feature branch. The unity version is 2019.4.13f1c1.

1.7.3 Game Character Jump

Title	Game character jump
Story description	As a game character, I can jump in the game scene.
Rule description	Jump: The action of the game character in the game, which makes the model of the game character in a higher position in the game scene.
Testing standard	On the plane 1.The game character can jump from the current plane to a higher plane.

Design plan	After the game starts, determine whether the player is on the ground first: this process generates a ray from the outside to the inside of the box collider. In order to, determine whether the first collider box that the ray hits is the player itself, and returns a Boolean value. Return false if it is. (since the initial state of the player is on the ground. So the first collider box hit by the ray must be the player at the very beginning of the game). After the player presses the jump button, the jump function will be called in the update function, and the ground judgment result is now included in the condition at the same time, for example. In that case the jump will be triggered if the player is on the ground.
Checking list	Whether the branch in GitHub is jump feature branch. The unity version is 2019.4.13f1c1.

1.7.4 Game Character Dash

Title	Game character dash
Story description	As a game character, I can dash in the game scene.
Rule description	Dash: The action of the game character in the game, which makes the Y axis of the model of the game character shrink in the game scene.
Testing standard	On the plane 1.The game model becomes smaller without interrupting the running state.
Design plan	Setting body size to shrink the game character model. Use the invoke to control time interval. Execute through the update function.
Checking list	Whether the branch in GitHub is dash feature branch. The unity version is 2019.4.13f1c1.

1.8 Tests for User Story

Test items	Case	When	Then	Result
Game character movement	1	When the user enters the game.	Users can see the automatically movement of game characters.	TRUE
Game character run	1	When the user enters the game.	The user can control the game character to run forward.	TRUE
Game character jump	1	When the user enters the game.	The user can control the game character to jump up by clicking the up button.	TRUE
Game character dash	1	When the user enters the game.	Users can make the game character model smaller without interrupting running by clicking down button.	TRUE

1.9 Sprint 1 Use Case

Use case	Game character movement
Summary	The user controls the game character to move forward in various actions in the game scene.
Actor	User (anyone playing the game)
Trigger	User enters the game
Primary scenario	1. The user clicks to enter the game. [Alternative scenario: The user exits the game directly.]. 2. After the user enters the game, the user controls the game character to run forward. [Exception scenario: The game character does not run forward.]. 3. The game character is running, and the user controls the game character to continue running. [Alternative scenario: The user controls the game character to jump.] [Alternative scenario: the user controls the game to dash.].

Alternative scenario	If the user exits the game directly 1.The user can return to primarily scenario 1. If the user controls the game character to jump 1.The game character jumps. If the user controls the game character dash 1.Game character dash.
Exceptional scenario	The game character does not run forward. 1.The user exits the game and tries to re-enter. Back to primarily scenario 1.
Pre-conditions	Start game
Post-conditions	None
Assumptions	None

1.10 Tests for Use Case

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user clicks to enter the game.	Event handler were deleted by accident, onclick action failed.	Added event handler.
	2	The user controls the game character to run forward.	F	F
	3	The user controls the game character to continue running.	F	F
Alternative scenario	1	If the user exits the game directly.	F	F
	2	If the user controls the game character to jump.	If you press the jump and dash keys at the same time, the game character will perform these two actions simultaneously.	Added “Boolean” to judge/control the inability to enter another when it has entered the jump/ dash state.
	3	If the user controls the game character dash.		

1.11 User Interface

According to the description of the game background story in the game proposal, this game mainly tells the story of a ninja. Therefore, the model actions of the game character are designed as follows. Through the client meeting, the following model actions have been confirmed by the client.

Run:

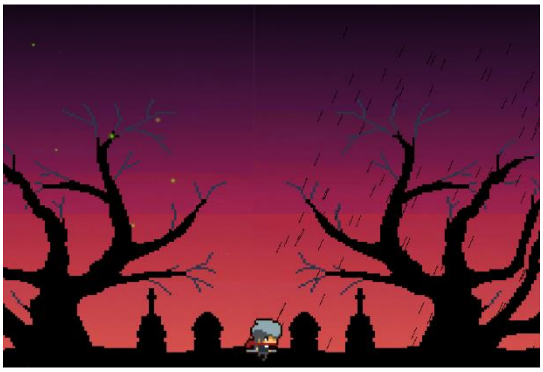


Figure 3: Character run

Jump:

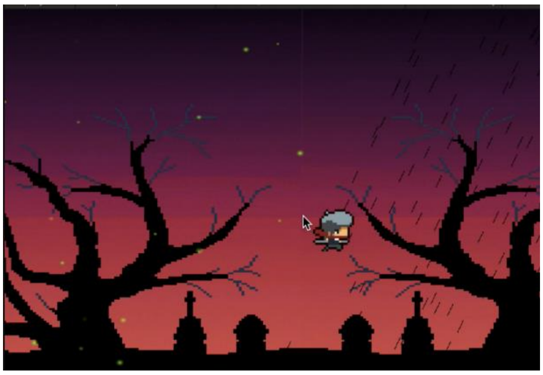


Figure 4: Character jump

Dash:

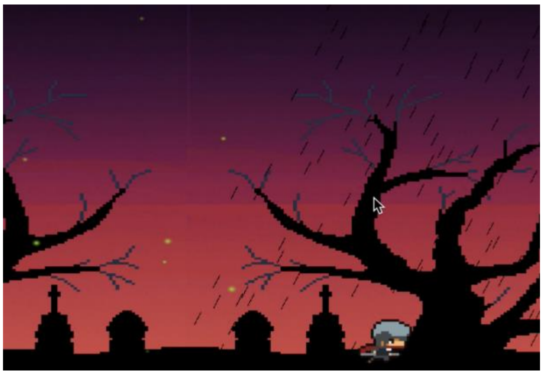


Figure 5: Character dash

1.12 Sprint 1 Exception Handling

Exception	Time of occurrence	Cause of the exception	Solution
A branch in unity is damaged.	6/11/2020	Error when merging into main method.	Go back to the previous version and rewrite the code.
Two members work in the same branch.	7/11/2020	Error when merging into main method.	Go back to the previous version and copy paste saved code so no need rewrite anymore.

Sprint 2

2.1 Sprint 2 Overview

After the game character can move normally and make jump and dash actions, in order to increase the difficulty and entertainment of the game. We must design some barriers to stop users from moving forward. Users can also make reasonable use of these obstacles to get more points. There are three types of obstacles, walls, monsters and traps. The user will physically collide with these three obstacles. The difference between the wall and the other two is that the game character will not die when touching the wall. Touching traps and monsters will instantly die. The dead function is recorded in the backlog, which is not the item of this sprint. In the process of realizing the function, the game scene is abnormal. Eventually resolved by the team members.

This sprint is mainly responsible for producing the underlying code rather than completing the game design. Only one example of each obstacle can be developed. The complete design of the game scenario will be carried out in the subsequent sprint. Team members will collect model pictures of different shapes from this sprint.

2.2 Sprint 2 Review

THE 1ST REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-0 Approval of Management Plan		
Have all the Project Activities been well defined?	√	
Has Mission Need Statement been well prepared and explained technically and functionally?	√	
Have a Meeting Session Arrangement well arranged?	√	
Progress Monitor Function is defined clearly? Progress Reviews?	√	
Have the tools for development been well listed and declared?	√	

The Feasibility?	√	
CD-1 Approval of QA		
All quality assurances (for text file as well as software) are well defined?	√	
Review actions for quality assurance are well defined?	√	
A Good definition of Coding and Text file style?	√	
The Feasibility?	√	
CD-2 Approval of Reviews		
Has the review session been well defined?	√	
Has a full review circle been well arranged?	√	
Has the review rules and details been well contained?	√	
CD-3 Approval of Version Control		
The tools that are going to be used are well declared?	√	
The ways to keep different kinds of files are well declared?	√	
Does the plan clearly arrange the development productions?	√	
Does the plan avoid the risky and insecurity in development?	√	
CD-4 Software tests, fault and bug reporting		
Contains clearly testing methods?	√	

Contains clearly debugging reviews?	√	
Contains clearly bug report sessions?	√	
Whether the report keeping actions and managements well?	√	
Comments: Generally, we catch up with the schedule. **Reviewer: Zhiyuan Tan**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-5 Approval of 1st Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: Although we just used the tools, but helped each other, on both document and coding tools. The functions were basically implemented. **Reviewer: Zhiyuan Tan** **Date: 9th November 2020**		

THE 2ND REVIEW APPROVAL

KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	YES	NO

CD-6 Approval of Development Progress Comparison		
A detailed comparison is made?	√	
An overall evaluation is made?	√	
A list between planned progress and real progress is made?	√	
A discus session on the solutions?	√	
A well-done record of the session?	√	
Comments: The design of the brainstorming game proposal by group members has been completed. **Reviewer: Zhiyuan Tan**		
CD-7 Approval of Development Questions		
Have actual questions reported clearly?	√	
Have all problems solved?	√	
Comments: Keep a backup of your code before doing any merge or test work, be careful not to change into other branch by mistake. **All Details of the Problems and Solutions should be recorded below:** **Problems:** Mistakenly changed into other branches and caused conflicts on Nov 11th. **Solutions:** Go back to the previous version and rebuilt the prefabs. **Reviewer: Zhiyuan Tan**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-8 Approval of 2nd Review Meeting		
Have an action list made after the session?	√	

Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: Just be careful on branch options. **Reviewer: Zhiyuan Tan** **Date: 12th Nov 2020**		
THE 3rd REVIEW APPROVAL		
KEY QUESTIONS FOR FINAL EVALUATION	**YES**	**NO**
CD-9 Approval of Final Evaluation		
Whether the sprint is full completeness?	√	
Full documents recorded?	√	
The sprint finished in good condition?	√	
A Final Evaluation?	√	
Comments: Generally completed the assignments. **Reviewer: Zhiyuan Tan**		
CD-10 Approval of Debugging		
Has a final test passed?	√	
Have a Debugging track recorded?	√	

All bugs fixed?	√	
Comments: All bugs fixed and ready for the next sprint to add more functionalities. **Reviewer: Zhiyuan Tan**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-11 Approval of 3rd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: The evaluation and following plan were clearly informed everyone. **Reviewer: Zhiyuan Tan**		
Sprint 2 Summary: During the sprint 2, everyone completed the assigned tasks on time. The items in our Sprint backlog which is “Prefab monster”. “Prefab walls” and “Prefab traps” has been well delivered and tested. All problems encountered were reported and resolved in time. All documents have been completed and viewed by scrum masters. The preparation for the next Sprint has been delivered by the product owner. Everyone knew how to handle the branch and the coder tried the implementation on following functions.		
Date: Nov 15th 2020		

2.3 Sprint 2 Customer Meeting Record

Customer Meeting Record

Client Name: Julian

Project Title: BitDash

Project Start Date: 25.10.2020

Project Deadline: 11.12.2020

Scrum no: 2	**Meeting Scheduled at**: 12:30-13:00	**Date**: 13.11.2020

The team decided to approach this customer meeting slightly differently with Cameron presenting to the customer. Additionally, it was decided that a PowerPoint should be produced to present to the client which will summarise the last meeting, describe our progress for the current Sprint, and make suggestions about the upcoming Sprint. This week the customer was provided with a set of 2 optional backstories for the progression of the game as a response to last week's meeting and customer emails. These were the original version, plus an alternative with an opposing set of artworks.

Analysation

The customer responded very well to the new format for the meeting and showed a strong preference for the team to stick to the original, and most fleshed out version of game from the 2 optional backstories. This is due to a concern about time limits. Sticking to the parkour 2d theme with pixelated imagery was preferential. Emphasis was made on ensuring a fully completed game is ready at handover.

With regard to game difficulty: The customer suggested that in line with the parkour theme, we use our environment to create difficulty, requiring jumps off of game objects etc. Multiple routes were mentioned as a possibility, but a focus should first be made on completing one route fully. A very nice suggestion was the implementation of a chimney, or hollow where the character could jump off the walls to go up the chimney. This could also lead to other routes if there is time to implement them and could allow us to create a very cool parkour wall-to-wall jumping trick.

It was mentioned that a feasibility assessment could be made about the second backstory option if we would like to follow that idea more.

Overall, the meeting structure was preferable, the customer was happy with our direction and the team felt happy that going forward we have a clearer end goal.

2.4 Sprint 2 Daily Scrum Meeting Record

Daily Scrum Meeting Record

Sprint 2: 09.11.2020 - 15.11.2020

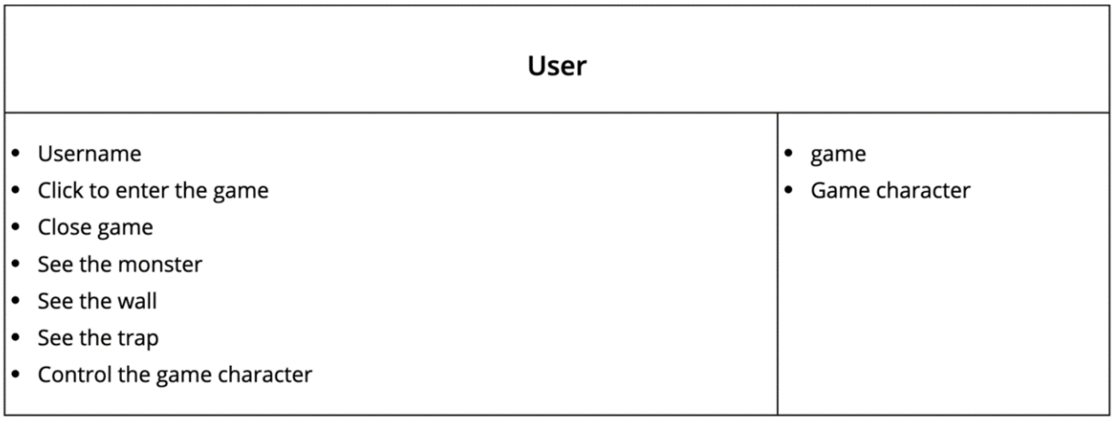
Date	Time	Members	Assigned Task	Summary
9th Nov	14:30 - 15:00	All team members	Function classification of motion module.	According to the backlog, the obstacles in the game are walls, monsters and traps.
10th Nov	13:00-13:30	All team members	Establish a corresponding Unity branch and assign functions to specific members.	Establish the corresponding Unity branch and collect the corresponding model pictures.
11th Nov	14:00-14:30	All team members	group1: Prefab Wall. group2: Prefab Trap. group3: Prefab Monster. group4: Main menu. Realize the functions.	Everyone had set up the unity and start to collect the pictures for prefabs.
12th Nov	13:00-13:30	All team members	group1: Prefab Wall. group2: Prefab Trap. group3: Prefab Monster. group4: Main menu. Realize the functions.	The main menu was done. The others would catch up. Encountered anomaly: Game scene deformation.
13rd Nov	14:00-14:30	All team members	Test by user story.	According to the user story without exception.
14th Nov	15:00-15:30	All team members	Test by use case.	According to the use case test without exception.
15th Nov	13:00-13:30	All team members	Sprint review and improve documentation.	Resolve functional conflicts and complete review and other documents.

2.5 Sprint 2 Backlog

Sprint 2	Item	Description	Priority
Sprint backlog	Prefab walls	As a user, I can control the game character to jump to the wall or avoid hitting the wall. 1.Wall model visible. 2.Walls can physically collide with game characters. 3.The wall model can be reused.	1
	Prefab monsters	As a user, I can control the game character to jump over the monster. 1.Monster model visible. 2.Monsters can physically collide with game characters. 3.The monster model can be reused.	1
	Prefab traps	As a user, I can control the game character to jump over the trap. 1.Trap model visible. 2.Traps can physically collide with game characters. 3.The trap model can be reused.	1
Product backlog	Game character death		
	Game score		
	Game over		

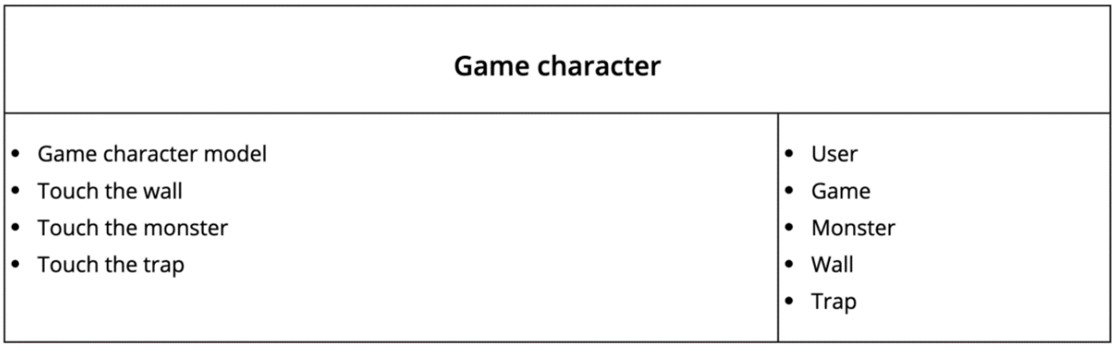
2.6 Sprint 2 CRC Cards

2.6.1 User



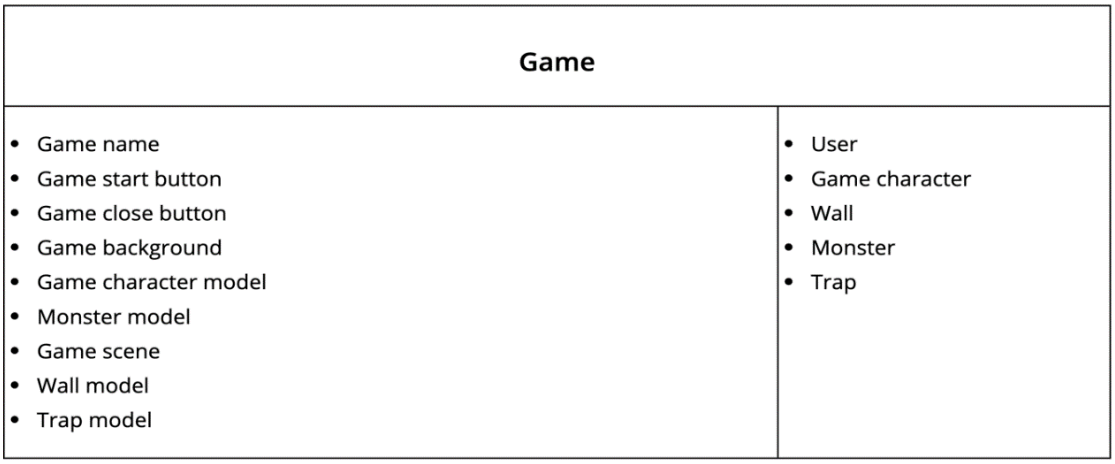
User	
• Username • Click to enter the game • Close game • See the monster • See the wall • See the trap • Control the game character	• game • Game character

2.6.2 Game Character



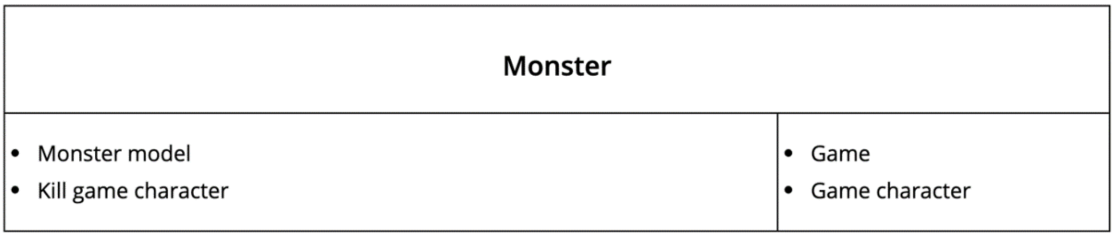
Game character	
• Game character model • Touch the wall • Touch the monster • Touch the trap	• User • Game • Monster • Wall • Trap

2.6.3 Game



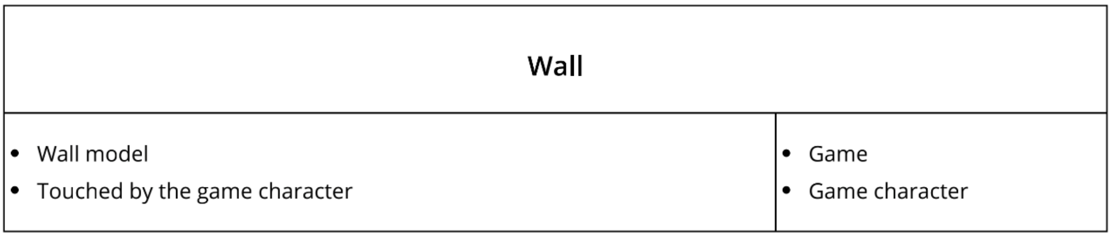
Game	
• Game name • Game start button • Game close button • Game background • Game character model • Monster model • Game scene • Wall model • Trap model	• User • Game character • Wall • Monster • Trap

2.6.4 Monster



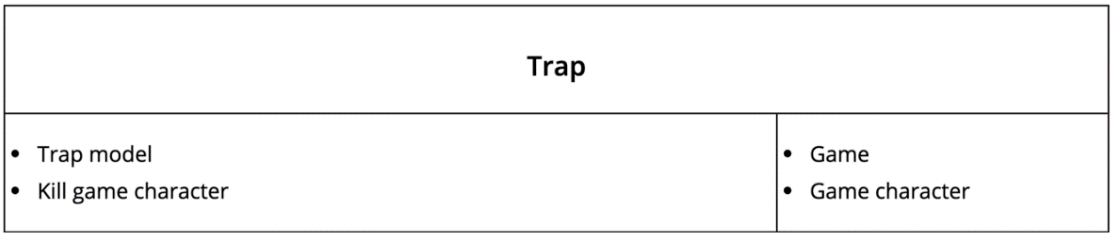
Monster	
• Monster model • Kill game character	• Game • Game character

2.6.5 Wall



Wall	
• Wall model • Touched by the game character	• Game • Game character

2.6.4 Trap



Trap	
• Trap model • Kill game character	• Game • Game character

2.7 Sprint 2 User Story

2.7.1 Prefab Monsters

Title	Prefab monsters
Story description	As a user, I can control the game character to jump over the monster.
Rule description	Monster: monsters need to be displayed in the game. Monsters kill player: when the monster touches the character, the character dies and the game ends.
Testing standard	1. Whether monsters can be displayed and distinguished with the background. 2. Whether monster can kill the character when there is a collision.

Design plan	1. Use pictures for monster prefabs carrier. 2. Use box collider 2D and tags for following judgement functions. 3. Use “rigidbody” 2D for preventing player from running through.
Checking list	Whether the branch in GitHub is jump feature branch. The unity version is 2019.4.13f1c1.

2.7.2 Prefab Walls

Title	Prefab walls
Story description	As a user, I can control the game character to jump to the wall or avoid hitting the wall.
Rule description	Wall: the wall needs to be displayed in the game and can be distinguished with the background clearly. Jump to the wall: the game character can jump to the wall when they are facing monsters chain that difficult to be avoided by a single jump. Walk on the wall: the character can walk on the wall.
Testing standard	1. Whether the wall can be displayed and distinguished with the background. 2. Whether the character can jump and stand on the wall by clicking the jump button. 3. Whether the wall will be missed by the character but still displayed on the screen if the player does not click the jump button at the right time (click the button too early/late). 4. Whether the character can walk on the wall without falling, bugs or crash the game, etc.
Design plan	1. Use piles to joint several wall pieces into a whole one. 2. Use box collider 2D and tags for following judgement functions. 3. Use “rigidbody” 2D for preventing player from running through.
Checking list	Whether the branch in GitHub is jump feature branch. The unity version is 2019.4.13f1c1.

2.7.3 Prefab Traps

Title	Prefabs traps
Story description	As a user, I can control the game character to jump over the trap.
Rule description	Traps: traps need to be displayed in the game. Traps can be avoided: when the game character notices a trap or obstacle is coming, the user can jump up to avoid been trapped or use the wall to do so. Traps can kill the player: when the player touches the trap, the character dies.
Testing standard	1. Whether traps can be displayed and distinguished with the background. 2. Whether the game character can jump up or use the wall to avoid obstacles or traps without causing any issues. 3. Whether the trap can kill the character when there is a collision.
Design plan	1. Use pictures (from resource file or create by drawing in pixel style) for traps prefabs carrier 2. Use box collider 2D and tags for following judgement functions. 3. Use "rigidbody" 2D for preventing player from running through.
Checking list	Whether the branch in GitHub is jump feature branch. The unity version is 2019.4.13f1c1.

2.8 Tests for User Story

Test items	Case	When	Then	Result
Using the prefab of monsters	1	When the game starts.	The monster can be seen.	TRUE
	2	When a monster attacks the player.	The monster has a collision with the character causing the death of Game character.	TRUE
	1	When the game starts.	The wall can be seen.	TRUE

Using the prefab of walls	2	When the player wants to jump to the wall.	The wall allows the character to jump on to and stand on it until the wall ends.	TRUE
	3	When the player does not jump to the wall.	The wall kept visible even the character missed jumping on to it.	TRUE
	4	When the player moves on the wall.	The wall allows the character to run on it without falling or bugs.	TRUE
Using the prefab of traps	1	When the game starts.	The traps can be seen.	TRUE
	2	When the player touches the trap.	The trap has a collision with the character causing the game character to die.	TRUE
	3	Players avoid traps.	The trap kept visible no matter whether the character has been avoided or not.	TRUE

2.9 Sprint 2 Use Case

2.9.1 Prefab Monsters

Use case	Prefabs Monsters
Summary	Game characters will encounter monsters in the game, Game characters will encounter monsters in the game and can avoid them.
Actor	User
Trigger	Game characters encounter monsters.
Primary scenario	1. The user clicks to enter the game [alternative scenario: the user exits the game directly]. 2. After the user enters the game, user can see the monsters somewhere in the scenes [Alternative scenario: the scenes do not contain any monsters] [Exceptional scenario: the user cannot see the monsters]. 3. When game character touch the monsters, player's character died [Alternative scenario: User does not touch the monster]

	[Exceptional scenario: the character touched the monster and not die].
Alternative scenario	If the user exits the game 1. The user can re-enter the game and return to the primary scenario 2. The scenes do not contain any monsters. 1. The user can continue to run and until there is a monster. The player does not touch the monsters. 1. The user can continue to run and until there is a monster.
Exceptional scenario	The user cannot see the monsters. 1. The user exits the game and tries to re-enter. The character touched the monster and not die. 1. The user exits the game and tries to re-enter.
Pre-conditions	Start the game
Post-conditions	Score record
Assumptions	None

2.9.2 Prefab Walls

Use case	Prefabs Walls
Summary	The user will encounter a wall in the game, and the user can walk on or avoid the wall.
Actor	User
Trigger	Users encounter a wall.
Primary scenario	1. The user clicks to enter the game [alternative scenario: the user exits the game directly] 2. After the user enters the game, user can see the walls somewhere in the scenes [Alternative scenario: the scenes do not contain any walls] [Exceptional scenario: the user cannot see the traps]

	3. The game character can walk on the wall to avoid traps and monsters [Alternative scenario: the user does not walk on the walls] [Exceptional scenario: the user cannot walk on the wall]
Alternative scenario	If the user exits the game 1. The user can re-enter the game and return to the primary scenario 2 The scenes do not contain any walls 1. The game character can continue to run and until there is a wall. The user does not walk on the walls 1. The user can walk on the wall when reach the next wall
Exceptional scenario	The user cannot see the walls 1. The user exits the game and tries to re-enter. The user cannot walk on the walls 1. The user exits the game and tries to re-enter.
Pre-conditions	Start the game
Post-conditions	None
Assumptions	None

2.9.3 Prefab Traps

Use case	Prefabs Traps
Summary	Game characters will encounter traps in the game, Game characters will trap in the game and can avoid them.
Actor	User
Trigger	Game characters encounter traps.
Primary scenario	1. The user clicks to enter the game [alternative scenario: the user exits the game directly]. 2. After the user enters the game, user can see the traps somewhere in the scenes [Alternative scenario: the scenes do not contain any traps] [Exceptional scenario: the user cannot see the traps] . 3. When game character touch the trap, player's character died [Alternative scenario: User does not touch the trap] [Exceptional scenario: the character touched the trap and not die].

Alternative scenario	If the user exits the game 1. The user can re-enter the game and return to the primary scenario 2. The scenes do not contain any traps. 1. The user can continue to run and until there is a trap. The player does not touch the trap. 1. The user can continue to run and until there is a trap.
Exceptional scenario	The user cannot see the traps. 1. The user exits the game and tries to re-enter. The character touched the trap and not die. 1. The user exits the game and tries to re-enter.
Pre-conditions	Start the game
Post-conditions	Score record
Assumptions	None

2.10 Tests for Use Case

2.10.1 Test Monsters

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user clicks to enter game.	F	F
	2	After the user enters the game, user can see the monsters somewhere in the scenes.	F	F
	3	When player touch the monsters, player's character died and game over.	F	F
Alternative scenario	1	If the user exits the game.	F	F
	2	The scenes do not contain any monsters.	F	F
	3	The player does not touch the monsters.	F	F

2.10.2 Test Walls

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user clicks to enter the game.	F	F
	2	After the user enters the game, user can see the walls somewhere in the scenes.	F	F
	3	The user can walk on the wall to avoid traps and monsters.	F	F
Alternative scenario	1	If the user exits the game.	F	F
	2	The scenes do not contain any walls.	F	F
	3	The user does not walk on the walls.	F	F

2.10.3 Test Traps

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user clicks to enter the game.	F	F
	2	After the user enters the game, user can see the traps somewhere in the scenes.	F	F
	3	When player touch the trap, player's character died.	F	F
Alternative scenario	1	If the user exits the game.	F	F
	2	The scenes do not contain any traps.	F	F
	3	The player does not touch the traps.	F	F

2.11 User Interface

During our customer meeting this week a lot of emphasis was placed upon game difficulty. Since our game is a parkour game, we will be trying to create the environment of the game to be part of the difficult by timing and executing jumps perfectly and in addition to this with the implementation of monsters and traps which will further increase the difficulty for the user. We have graphically displayed the monsters, traps, and tiles.

Tiles, Traps and Monsters (Figure 6):

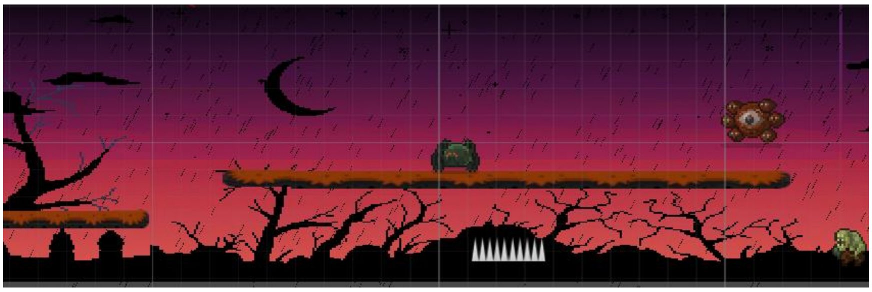


Figure 6: Tiles, Traps and Monsters

This image above shows an example of how the different prefabs can be used to create the environment for the user to traverse.

Jump over monsters and traps (Figure 7):

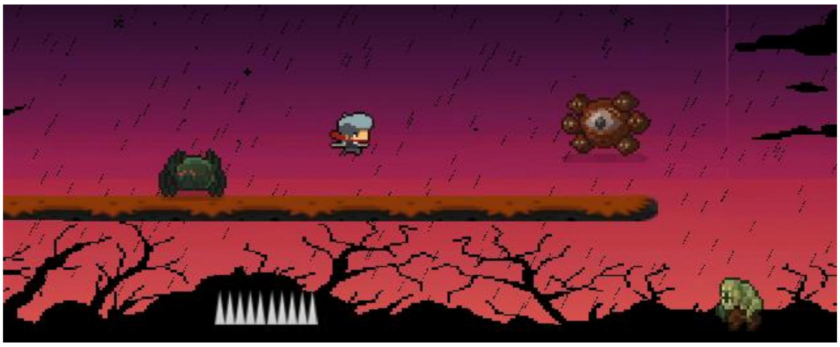


Figure 7: Jump over monsters and traps

The image above shows how the user will have to time jumps in order to traverse the environment in order to avoid death.

2.12 Sprint 2 Exception Handling

Exception	Time of occurrence	Cause of the exception	Solution
Game scene deformation	14/11/2020	When designing monster, wall, and trap models, the game scene was damaged because multiple members moved positions.	Re-merge and delete extra files.

Sprint 3

3.1 Sprint 3 Overview

In this sprint, all controllable behaviours in the user's game will be completed. User-controlled game characters will die when encountering monsters and traps. The user can view the score of the game after the death of the game character. At the same time, the team members collected enough pictures to make the next sprint.

This sprint is by far the most problematic in the implementation of functions. Through testing, there are some bugs in the death and scoring functions of the preliminary design. This causes the game character to die and roll when touching all obstacles. And the user's score will continue to accumulate and will not end with the death of the game character. These two are currently the most serious problems. The team members meet urgently and discuss solutions. Finally, all functional modifications will be completed at the end of the sprint.

3.2 Sprint 3 Review

THE 1ST REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-0 Approval of Management Plan		
Have all the Project Activities been well defined?	√	
Has Mission Need Statement been well prepared and explained technically and functionally?	√	
Have a Meeting Session Arrangement well arranged?	√	
Progress Monitor Function is defined clearly? **Progress Reviews?**	√	
Have the tools for development been well listed and declared?	√	

The Feasibility?	√	
CD-1 Approval of QA		
All quality assurances (for text file as well as software) are well defined?	√	
Review actions for quality assurance are well defined?	√	
A Good definition of Coding and Text file style?	√	
The Feasibility?	√	
CD-2 Approval of Reviews		
Has the review session been well defined?	√	
Has a full review circle been well arranged?	√	
Has the review rules and details been well contained?	√	
CD-3 Approval of Version Control		
The tools that are going to be used are well declared?	√	
The ways to keep different kinds of files are well declared?	√	
Does the plan clearly arrange the development productions?	√	
Does the plan avoid the risky and insecurity in development?	√	
CD-4 Software tests, fault and bug reporting		

Contains clearly testing methods?	√	
Contains clearly debugging reviews?	√	
Contains clearly bug report sessions?	√	
Whether the report keeping actions and managements well?	√	
Comments: The test except for some bugs, such as the hero will roll after collision and the score will increase when the start button is not clicked. Finally, successfully fixed these bugs. **Reviewer: Xuan Wang**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-5 Approval of 1st Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: Reasonable task allocation. High feasibility. Sufficient and necessary communication. **Reviewer: Xuan Wang** **Date: Nov 16th 2020**		

THE 2ND REVIEW APPROVAL

KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	YES	NO
CD-6 Approval of Development Progress Comparison		
A detailed comparison is made?	√	
An overall evaluation is made?	√	
A list between planned progress and real progress is made?	√	
A discuss session on the solutions?	√	
A well-done record of the session?	√	
Comments: The session enumerated the problems encountered and proposed corresponding solutions through discussion. **Reviewer: Xuan Wang**		
CD-7 Approval of Development Questions		
Have actual questions reported clearly?	√	
Have all problems solved?	√	

Comments:

Keep a backup of your code before doing any merge or test work.

All Details of the Problems and Solutions should be recorded below:

Problems:

After the game is over, the score pauses and the display of the game over animation are out of sync.

Solutions:

The original independent decision logic of the score suspension was deleted, and a new decision condition was added to the death function.

Reviewer: Xuan Wang

KEY QUESTIONS FOR MEETING REVIEW	YES	NO
CD-8 Approval of 2nd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	

Comments:

Despite some technical difficulties, all tasks in this sprint were completed on time.

Reviewer: Xuan Wang

Date: Nov 19th 2020

THE 3rd REVIEW APPROVAL

KEY QUESTIONS FOR FINAL EVALUATION	YES	NO
CD-9 Approval of Final Evaluation		

Whether the sprint is full completeness?	√	
Full documents recorded?	√	
The sprint finished in good condition?	√	
A Final Evaluation?	√	
Comments: The sprint is full completeness. **Reviewer: Xuan Wang**		
CD-10 Approval of Debugging		
Has a final test passed?	√	
Have a Debugging track recorded?	√	
All bugs fixed?	√	
Comments: Some bugs appear because of insufficient communication when designing functions. Specifically, different logics are used to determine the score suspension and the hero's death. **Reviewer: Xuan Wang**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-11 Approval of 3rd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	

Have all missions in this meeting well finished?	√	

Comments:

All missions were well finished, and everyone knows what will happen next sprint.

Reviewer: Xuan Wang

Sprint 3 Summary:

During the sprint 3, everyone completed the assigned tasks on time. The items in our Sprint backlog which is “Run”. “Score” and “Death” has been well delivered and tested. All problems encountered were reported and resolved in time. All documents have been completed and viewed by scrum masters. The preparation for the next Sprint has been delivered by the product owner. All members have reasonable work to do in the sprint 4. Busy time periods are notified, and work assignment might be increased or decreased reasonably to ensure fairness.

Date: Nov 22nd 2020

3.3 Sprint 3 Customer Meeting Record

Customer Meeting Record

Client Name: Julian		**Project Title**: BitDash
Project Start Date: 25.10.2020		**Project Deadline**: 11.12.2020
Scrum no: 3	**Meeting Scheduled at:** 12:15-12:45	**Date**: 20.11.2020

This week we continued with new format in the meeting with the PowerPoint presentation in the format of:

Response to the last meeting - Current game state - Where we plan to go in the next sprint.

From the response to last meeting we showed we have implemented daily meetings, made a firm decision on the backstory of our game, and implemented aspects of difficulty into the game.

The game state we showed that we have the main aspects implemented. The jump and Dash functions are working, and prefabs have been created for tiles, monsters, and traps

Our aim for next week is made up of 3 main aspects. Aggregating all prefab work from separate branches, constructing the main route, and testing.

In addition, we gave some potential futures for the game to show the customer that the game may be further developed beyond the original scope.

Analysation
The customer responded well to the presentation. Some suggestion by the customer was made about the 'health' of the player being damaged in instances where jumps had not been made correctly in keeping with the parkour aspect. Health could improve over time. It was also suggested that once injured a player may be slowed down, however, this would make things slightly easier for the player as the speed of the game is an element of its difficulty. The customer overall was happy and said that we are on track and doing a good job.

3.4 Sprint 3 Daily Scrum Meeting

Daily Scrum Meeting Record

Sprint 3: 16.11.2020 - 22.11.2020

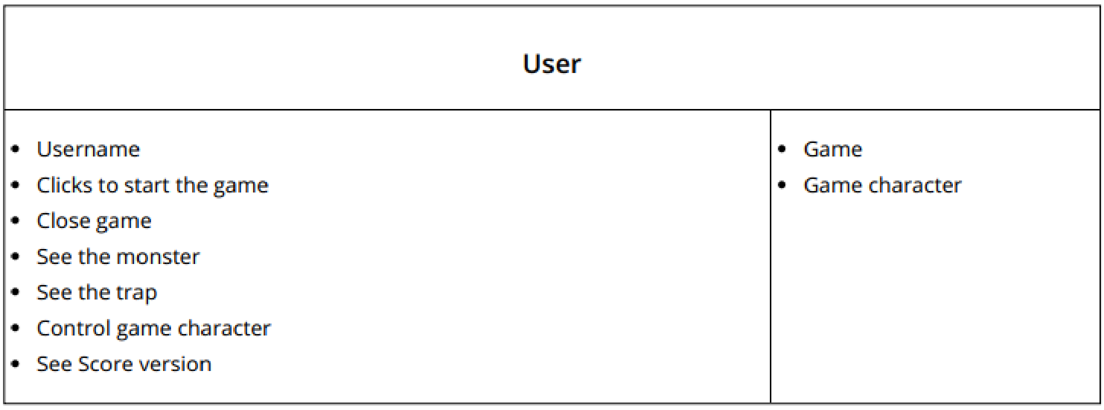
Date	Time	Members	Assigned Task	Summary
16th Nov	15:15 - 15:45	All team members	Function classification of motion module.	According to the backlog, the function is divided into two parts: death and points.
17th Nov	15:15 - 15:45	All team members	Establish a corresponding Unity branch and assign functions to specific members.	Write use cases and user stories and then focus on designing how to implement functions.
18th Nov	15:15-15:45	All team members	Book a customer meeting and organize previous documents.	All members have progressed without exception.
19th Nov	15:15-15:45	All team members	Realize death and scoring function.	Encountered an exception: Github file conflict.
20th Nov	12:30-13:00	All team members	Test by user story.	According to the use case test without exception.
21st Nov	15:15-15:45	All team members	Test by use case.	1.The score function damage 2.Roll over after hitting something when character die. 3.The score will continue to increase over time.
22nd Nov	15:15-15:45	All team members	Sprint review and improve documentation.	Resolve functional conflicts and complete review and other documents.

3.5 Sprint 3 Backlog

Sprint 3	Item	Description	Priority
Sprint backlog	Game character death	As a user, the game characters I control will die when they hit monsters or traps. 1.Game character death model.	1
	game score	As a user, when I control the cumulative survival time of the game character in the game, my score will increase. 1.User score panel.	2
Product backlog	Game over		
	Design game levels		

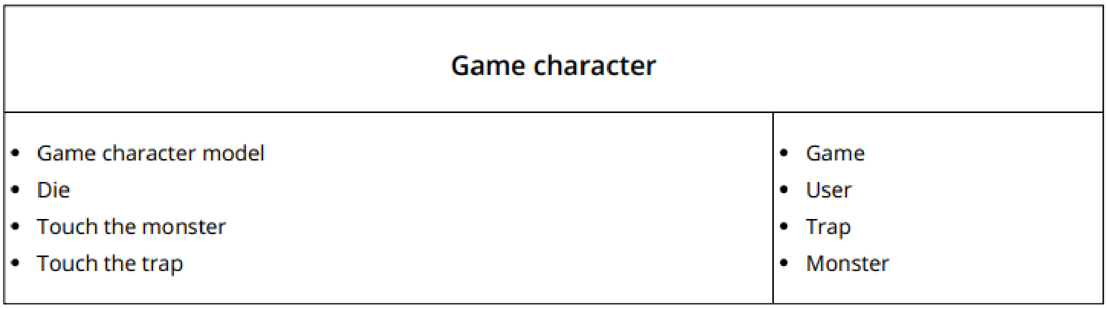
3.6 Sprint 3 CRC Cards

3.6.1 User



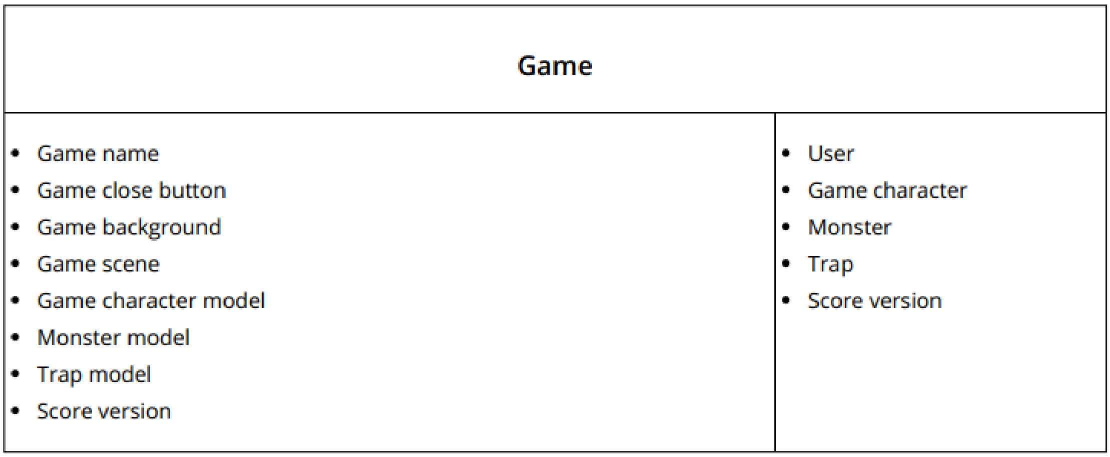
User	
• Username • Clicks to start the game • Close game • See the monster • See the trap • Control game character • See Score version	• Game • Game character

3.6.2 Game Character



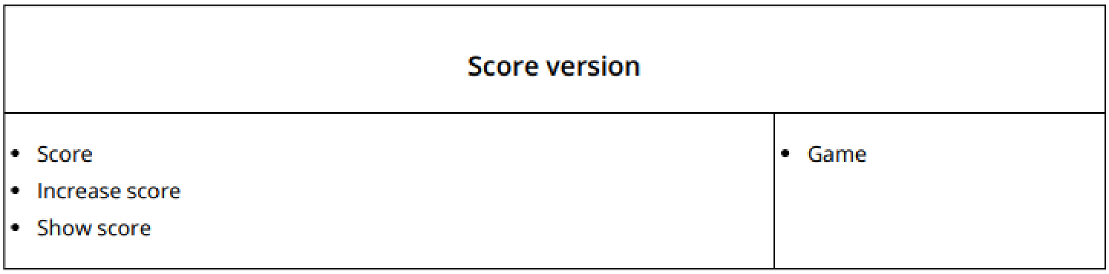
Game character	
• Game character model • Die • Touch the monster • Touch the trap	• Game • User • Trap • Monster

3.6.3 Game



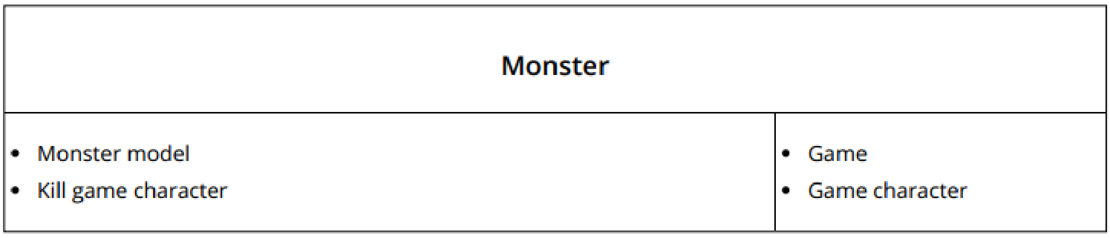
Game	
• Game name • Game close button • Game background • Game scene • Game character model • Monster model • Trap model • Score version	• User • Game character • Monster • Trap • Score version

3.6.4 Score Version



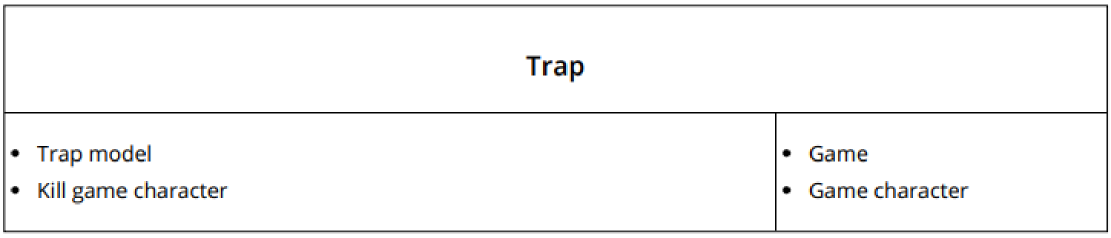
Score version	
• Score • Increase score • Show score	• Game

3.6.5 Monster



Monster	
• Monster model • Kill game character	• Game • Game character

3.6.6 Trap



Trap	
• Trap model • Kill game character	• Game • Game character

3.7 Sprint 3 User Story

3.7.1 Game Character Die

Title	Game character die
Story description	As a user, the game character I control will die when it touches a monster or trap.
Rule description	Game character death: The user can no longer control the game character.
Testing standard	1.Game characters will die if they touch the trap. 2.The game character will die when touching the monster.
Design plan	When the player detects a collision with a tag of death, stop the player's actions, stop scoring, call the death function, change the game running state to stop, and activate the background text to display the content GAME OVER.
Checking List	Whether the branch in GitHub is death feature branch. The unity version is 2019.4.13f1c1.

3.7.2 Score

Title	Score
Story description	As a user, I can increase my score by surviving longer in the game.
Rule description	score: A measure used to evaluate the user's achievements in the game.
Testing standard	1. The user's living time increases, and the user's score increases. 2. The game character dies, and the score no longer increases.
Design plan	After the game starts (the player clicks the start button), the score starts to increase smoothly and consistently. To achieve this function, you need to change the content displayed by “scoreText” in the “gameManger” script to a float type “scoreAmount” that increases over time, and then add this script as a component to UI “CanVas” and associate it with the start button and the end button respectively. When the game ends, call the death function to stop scoring.
Checking List	Whether the branch in GitHub is score feature branch. The unity version is 2019.4.13f1c1.

3.8 Tests for User Story

Test items	Case	When	Then	Result
Game character die	1	When Game character touches the trap.	Game characters will die.	TRUE
	2	When Game character touches the monster.	The game character will die.	TRUE
Score	1	The user's living time increases.	The user's score increases.	TRUE
	2	When game character dies	The score no longer increases.	TRUE

3.9 Sprint 3 Use Case

3.9.1 Game Character Death

Use case	Game character death
Summary	When the game character touches the walls, monsters, or traps in the level, the game character die.
Actor	User (anyone playing the game)
Trigger	The game character touches the monsters or traps.
Primary scenario	1. The user is playing a game. 2. When the user plays, the cumulative score increases. 3. The user-controlled game character touches a trap or monster. [Exceptional scenario: Game character cannot touch monsters or traps.]. 4. The game character died. [Exceptional scenario: The game character is not dead.].
Alternative scenario	None
Exceptional scenario	The game character cannot touch monsters or traps. 1.Exit the game and re-enter. The game character is not dead. Exit the game and re-enter.
Pre-conditions	Game character movement

Post-conditions	Score
Assumptions	None

3.9.2 Score

Use case	Score
Summary	When the game character touches the walls, monsters, or traps in the level, the game character's death will show the game score.
Actor	User (anyone playing the game)
Trigger	The game character touches the walls, monsters, and traps in the level.
Primary scenario	1. The user-controlled game character dies . 1. After the game character is died, the user's score is displayed. [Exceptional scenario: There is no cumulative increase in user scores.] [Exceptional scenario: After the Game character death, the user score is not displayed.].
Alternative scenario	None
Exceptional scenario	1. No cumulative increase in user scores. ● The user exits the game and tries to re-enter. 2. The user score is not displayed after the game is over. ● The user exits the game and tries to re-enter.
Pre-conditions	Game character death
Post-conditions	Game over
Assumptions	None

3.10 Tests for Use Case

3.10.1 Test Character Death Function

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user-controlled game character dies.	F	F

	2	After the game character is died, the user's score is displayed.	The score will continue to increase over time.	The programmer responsible for developing the score function module cannot solve this problem independently. After discussing with the programmer responsible for the death module, the judgment condition of the score suspension was added to the death judgment method to solve the problem.
Alternative scenario	**-**	-	F	F

3.10.2 Test of Score

Test items	**Case**	**Scenario description**	**Accidents**	**Solution**
Primary scenario	1	The user is playing a game.	F	F
	2	When the user plays, the cumulative score increases.	The score can only be incremented by the number of seconds, specifically, one-second increases by one point.	Modify the parameters in Unity to make the effect of score increase coherent and smooth.
	3	The user-controlled game character touches a trap or monster.	F	F
	4	The game character died.	Roll over after hitting something.	Two solutions. The first is to lock the Z axis; the second is to modify the collision frame of the monster to prevent the judgment range from being too large.
Alternative scenario	**-**	-	F	F

3.11 User Interface

For the current Sprint, the aim was to provide the user with a further two main features. These were to display the death of the character in the game, and to display the users score in real-time. These features present themselves in the user interface in the following format.

Death of the Character:

The death of the character occurs when he makes contact with a monster or a trap (Figure 8&9). This is evident in the interface through a number of changes in the gameplay. The scoring function stops accruing and remains fixed at the final game score. The character stops running and the user controls for dash and jump cease to operate. The 'Game Over' text will also be displayed centre screen.

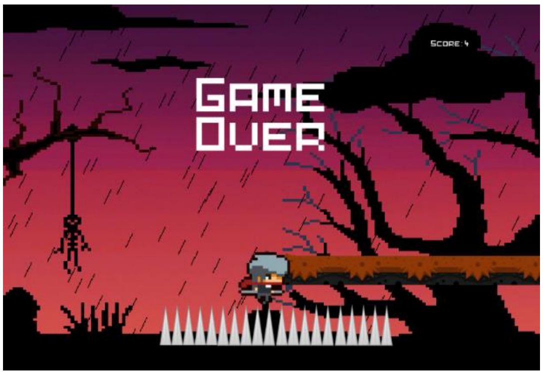


Figure 8: Character hit trap

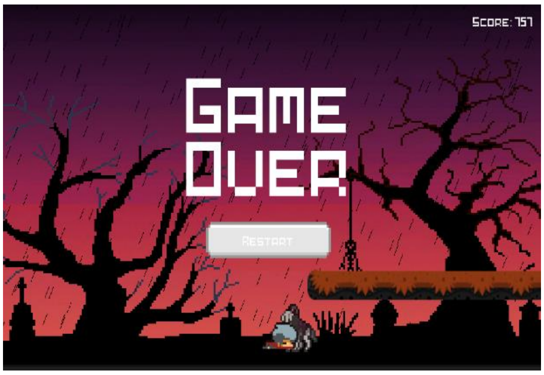


Figure 9: Character died

Users Score:

The users score is set to be linked with the distance that the character can make it through the game, which also accounts for a time alive. This feature will provide a constant display of the real-time accruing score in the left (Figure 10) of the screen and serves to both encourage the player and allow them to keep track of progress against previous attempts. The function is shown below as it accrues with time and distance (Figure 11).

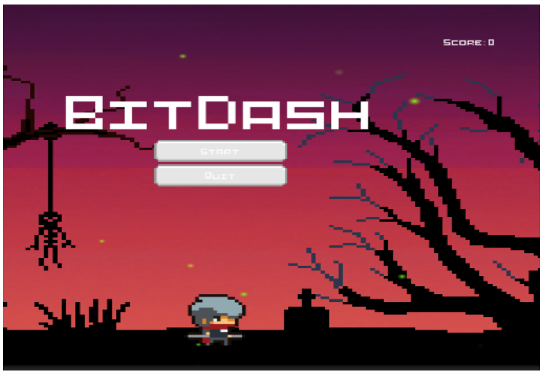


Figure 10: Start as Score 0

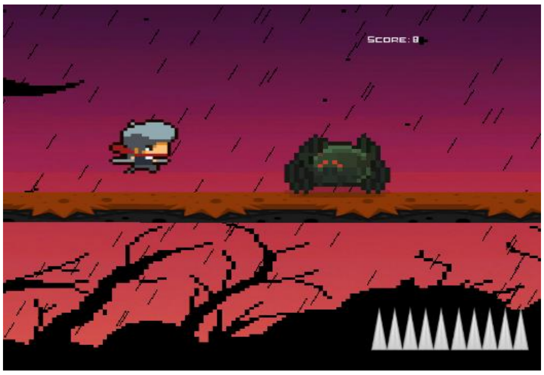


Figure 11: Score change over time

3.12 Sprint 3 Exception Handling

Exception	Time of occurrence	Cause of the exception	Solution
Github file conflict	20/11/2020	There are a lot of unnecessary Unity configuration files in the upload file including cache and text.	The version is returned to reposition the model.

Sprint 4

4.1 Sprint 4 Overview

Before the start of this sprint, team members made changes to the design of the game framework. Mentioned in the customer meeting in sprint0. Originally this game will have a similar user score ranking interface. However, this game will eventually be treated as a simple casual standalone game. So, in the development process, we removed this part of the content. This sprint is different from sprint2, which is just a model. And this sprint is designed to put multiple obstacles into the scene.

Sprint4 is the most difficult Sprint. In addition to designing the levels in the game, the game also needs to be over. So far, a basic design of the game has been completed. However, for users, the game still needs to continue to improve, for which we designed the game restart function. Because the design of a single map is limited. The team members decided to design the level mode as a loop.

4.2 Sprint 4 Review

THE 1ST REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-0 Approval of Management Plan		
Have all the Project Activities been well defined?	√	
Has Mission Need Statement been well prepared and explained technically and functionally?	√	
Have a Meeting Session Arrangement well arranged?	√	
Progress Monitor Function is defined clearly? Progress Reviews?	√	
Have the tools for development been well listed and declared?	√	

The Feasibility?	√	
CD-1 Approval of QA		
All quality assurances (for text file as well as software) are well defined?	√	
Review actions for quality assurance are well defined?	√	
A Good definition of Coding and Text file style?	√	
The Feasibility?	√	
CD-2 Approval of Reviews		
Has the review session been well defined?	√	
Has a full review circle been well arranged?	√	
Has the review rules and details been well contained?	√	
CD-3 Approval of Version Control		
The tools that are going to be used are well declared?	√	
The ways to keep different kinds of files are well declared?	√	
Does the plan clearly arrange the development productions?	√	
Does the plan avoid the risky and insecurity in development?	√	
CD-4 Software tests, fault and bug reporting		

Contains clearly testing methods?	√	
Contains clearly debugging reviews?	√	
Contains clearly bug report sessions?	√	
Whether the report keeping actions and managements well?	√	
Comments: The tasks were assigned according to the workflow of the team members. For sprint 4, the workflow was reduced because of the heavy work for everyone. However, the plan considered the situation and covered the aspects such as the quality control etc. The tasks were acceptable for all the team members. **Reviewer: Xiaoyan Zhou**		

KEY QUESTIONS FOR MEETING REVIEW	YES	NO
CD-5 Approval of 1st Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: The meeting was involved two parts, 1st part everyone gave the feedback for the deadlines and current work and 2nd part was assign the moderate amount of work to guarantee the process of the development. **Reviewer: Xiaoyan Zhou** **Date: 23rd Nov 2020**		

THE 2ND REVIEW APPROVAL

KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	YES	NO
CD-6 Approval of Development Progress Comparison		
A detailed comparison is made?	√	
An overall evaluation is made?	√	
A list between planned progress and real progress is made?	√	
A discus session on the solutions?	√	
A well-done record of the session?	√	
Comments: The meeting was for checking the workflow at the middle of the sprint and it became especially significant since the busy period of all the members. Works flow were slightly behind than the plan. The goal would be finished at the next half of the sprint. **Reviewer: Xiaoyan Zhou**		
CD-7 Approval of Development Questions		
Have actual questions reported clearly?	√	
Have all problems solved?	√	
Comments: The code was well kept and backup, coding process was slightly behind. No new bugs had occurred yet. **All Details of the Problems and Solutions should be recorded below:** **Problems:** None		

Solutions: None

Reviewer: Xiaoyan Zhou

KEY QUESTIONS FOR MEETING REVIEW	YES	NO
CD-8 Approval of 2nd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	

Comments:

All meeting details had recorded. Working flow would keep the same as the 1st review meeting decided.

Reviewer: Xiaoyan Zhou

Date: 26th Nov 2020

THE 3rd REVIEW APPROVAL

KEY QUESTIONS FOR FINAL EVALUATION	YES	NO
CD-9 Approval of Final Evaluation		
Whether the sprint is full completeness?	√	
Full documents recorded?	√	
The sprint finished in good condition?		√
A Final Evaluation?	√	

<table>
<tr><td colspan="3"><b>Comments:</b>
The sprint was finished in a hurry although no documents and codes were missing. Most of the work was finished at the daily sprint of this sprint. Work quality was still needed to check when using the functions.
<b>Reviewer: Xiaoyan Zhou</b></td></tr>
<tr><td colspan="3"><b>CD-10 Approval of Debugging</b></td></tr>
<tr><td><b>Has a final test passed?</b></td><td>√</td><td></td></tr>
<tr><td><b>Have a Debugging track recorded?</b></td><td>√</td><td></td></tr>
<tr><td><b>All bugs fixed?</b></td><td>√</td><td></td></tr>
<tr><td colspan="3"><b>Comments:</b>
No new bugs had been detected when running the tests, but since the functions would be used in more place later, more tests were required.
<b>Reviewer: Xiaoyan Zhou</b></td></tr>
<tr><td><b>KEY QUESTIONS FOR MEETING REVIEW</b></td><td><b>YES</b></td><td><b>NO</b></td></tr>
<tr><td colspan="3"><b>CD-11 Approval of 3rd Review Meeting</b></td></tr>
<tr><td><b>Have an action list made after the session?</b></td><td>√</td><td></td></tr>
<tr><td><b>Have all members attended the meeting?</b></td><td>√</td><td></td></tr>
<tr><td><b>Have all missions be defined and separated to all members clearly?</b></td><td>√</td><td></td></tr>
<tr><td><b>Have all missions in this meeting well finished?</b></td><td>√</td><td></td></tr>
<tr><td colspan="3"><b>Comments:</b>
The meeting was happened immediately after the daily sprint. Overall, all the work had been finished.
<b>Reviewer: Xiaoyan Zhou</b></td></tr>
</table>

Sprint 4 Summary:

The sprint 4 became much difficult for everyone because of the working flow. Although all the work had been finished and tested, the quality was slightly lower than the ones before. The tests were not enough even though the functions were working fine. This sprint had made us to consider more about the flow control at the beginning of the sprint and the quality control of the works. The following sprint, we would consider more about it. However, overall, the sprint was still successful.

*More thing to be covered: suitable workflow control and quality control.

Date: Nov 29th 2020

4.3 Sprint 4 Customer Meeting Record

Customer Meeting Record

Client Name: Julian

Project Title: BitDash

Project Start Date: 25.10.2020

Project Deadline: 11.12.2020

Scrum no: 4	**Meeting Scheduled at:** 12:15-12:45	**Date**: 27.11.2020

This week the following ideas were presented to the client in the continuing format:

Response to the last meeting:

We are continuing to focus on documentation and general daily meetings. We have merged all the separate branches and implemented further level design and gameplay features since the last scrum.

Current game state:

A video of the dash feature was presented to demonstrate the feature and where we were at with the game. A picture of the level design showing a certain amount of collections of prefabs already implemented was also shown to demonstrate additional work in other branches.

Where we plan to go in the next sprint:

In the next sprint we aim to have the first route completed, to have enabled the scoring and death functions, have the game in a working state and be starting the post-game documentation.

In addition to the usual meeting format, the latest version of unity was demonstrated. Prefab tiles were shown to demonstrate ease of replication, to create additional levels and the latest version of unity was demonstrated.

Analysation
In the pre-meeting it was mentioned that we should now be starting to focus on our final document and bringing all our documentation together for the final submission. It was decided that we would now do some test exports in Jira and decide on our final format. The presentation was given with very little feedback throughout from the client. The client enquired about documentation and so the user manual was mentioned as post game documentation with no real response either. The customer finished with 2 distinct points of feedback: 1. Upon presenting the game to the client through video, pictures and a presentation of the latest main branch of unity, the customer said that physics were unrealistic, the character jumped an unusually large distance and that we could adjust parameters that control the game. We felt this could be done at testing stage and there was no real resistance to this. 2. Client enquired whether joining tiles and assets together is going to be done manually? We responded that we would deal with these issues in later test stages. However, his response was that instead of manual construction of runs, you could construct these runs programmatically – auto produce different levels. We would however need to think through the logic to control variation of assets as they are compiled. The client finished by saying that what really matters is there is a connection between user stories, user cases etc. ‘Traceability’ from the customer desire -> final implementation.

4.4 Sprint 4 Daily Scrum Meeting Record

Daily Scrum Meeting Record

Sprint 4: 23.11.2020 - 29.11.2020

Date	Time	Members	Assigned Task	Summary
23rd Nov	15:15-15:45	All team members	Function classification of motion module.	According to the backlog, various obstacles of the game will be combined into game levels in this sprint. And the game character will die when hitting an obstacle.
24th Nov	15:15-15:45	All team members	Establish a corresponding Unity branch and assign functions to specific members.	Write use cases and user stories and then focus on designing how to implement functions.

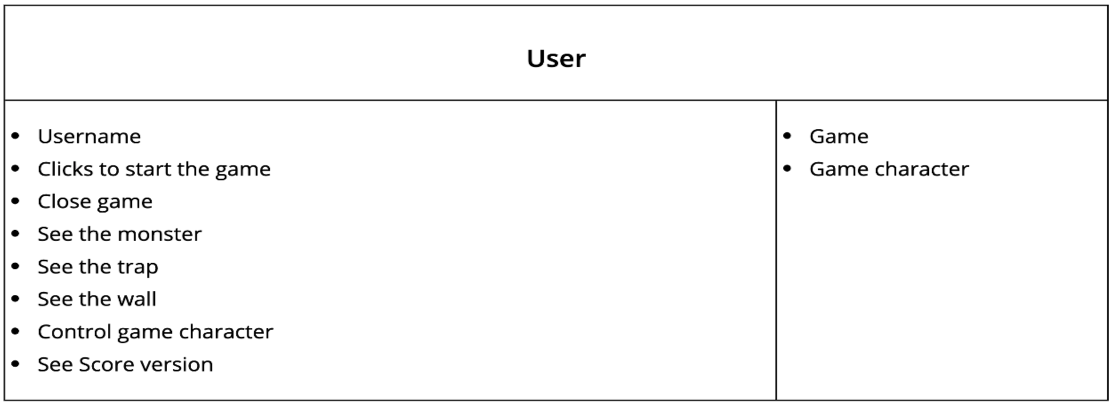
25th Nov	15:15-15:45	All team members	Collect more prefab pictures.	All members have progressed without exception.
26th Nov	14:15-14:45	All team members	Realize game over function.	All members have progressed without exception.
27th Nov	12:45-13:15	All team members	Test by user story.	According to the user story without exception.
28th Nov	15:15-15:45	All team members	Test by use case.	1.The character kept running after the score showed. 2.The character would sometimes crash on the wall and stop moving forward.
29th Nov	15:15-15:45	All team members	Sprint review and improve documentation.	Resolve functional conflicts and complete review and other documents.

4.5 Sprint 4 Backlog

Sprint 4	Item	Description	Priority
Sprint backlog	Game over	As a user, I can end the game after checking the score. 1.Game over interface.	1
	Game levels	As a game character, I can avoid obstacles through various actions and increase my survival time in the game. 1.Design a variety of monster models. 2.Design a variety of wall models. 3.Design multiple trap models. 4.Apply the model to the game scene. 5.Model can be reused.	1
Product backlog	Level loop		
	Restart		

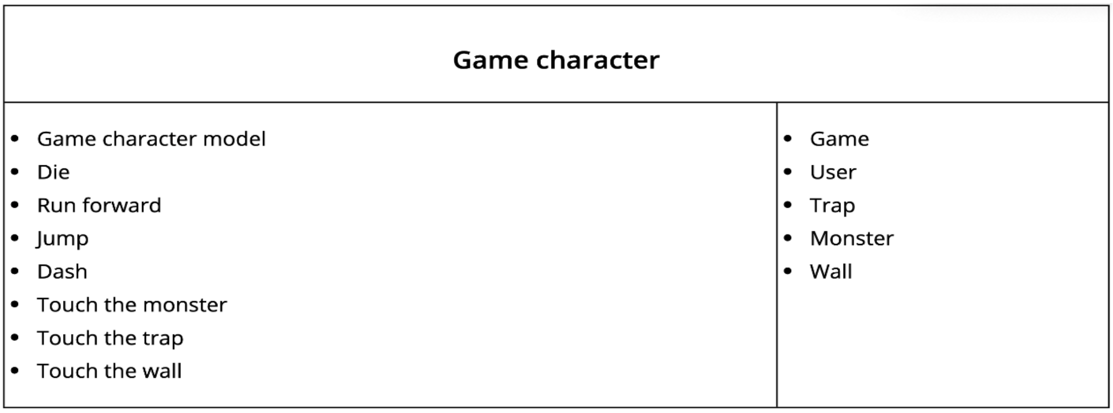
4.6 Sprint 4 CRC Cards

4.6.1 User



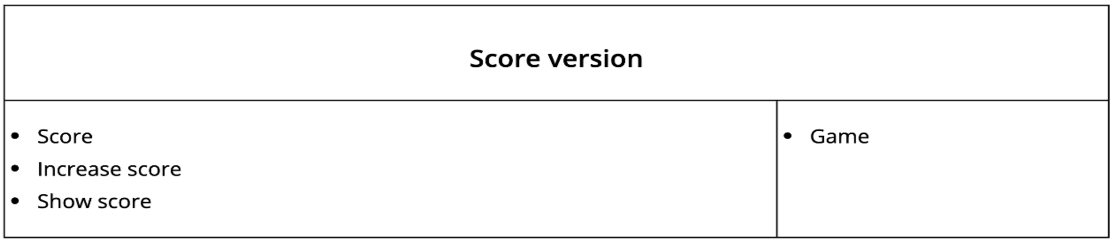
User	
• Username • Clicks to start the game • Close game • See the monster • See the trap • See the wall • Control game character • See Score version	• Game • Game character

4.6.2 Game Character



Game character	
• Game character model • Die • Run forward • Jump • Dash • Touch the monster • Touch the trap • Touch the wall	• Game • User • Trap • Monster • Wall

4.6.3 Score Version



Score version	
• Score • Increase score • Show score	• Game

4.6.4 Game

Game	
• Game name • Game start button • Game close button • Game background • Game scene • Game character model • Monster model • Trap model • Wall model • Score version • Show game over	• User • Game character • Monster • Trap • Wall • Score version

4.6.5 Monster

Monster	
• Monster model • Kill game character	• Game • Game character

4.6.6 Trap

Trap	
• Trap model • Kill game character	• Game • Game character

4.6.6 Wall

Wall	
• wall model • Touched by game character	• Game • Game character

4.7 Sprint 4 User Story

4.7.1 User Passes the Level

Title	User passes the level
Story description	As a user, I can extend my survival time in the game by running and jumping dash in the game.
Rule description	Level: consists of monsters, traps and walls. The elements that make up the level can be repeated.
Testing standard	1.The character avoids the monster or trap game continues. 2.The game character hits a wall, trap or monster. The game character dies.
Design plan	The elements of monsters, traps and tiles are stored as prefabs. By combining different design of the tiles, traps and monsters, the game level would increase step by step.
Checking list	The unity version is 2019.4.13f1c1.

4.7.2 Game Over

Title	Game over
Story description	As a user, I know that I can end the game after I score.
Rule description	End the game: The user cannot operate the game character and can choose to exit the game.
Testing standard	1.The user cannot operate the game character. 2.User can exit the game.
Design plan	The character would eventually reach an invisible box collider at the end of the backgrounds. The box collider would large enough to guarantee the character to touch it. And after the touch the game over and show the final score.
Checking list	The unity version is 2019.4.13f1c1.

4.8 Tests for User Story

Test items	Case	When	Then	Result
User passes the level	1	The character avoids the monster or trap.	The game continues.	TRUE
	2	The game character hits traps or monsters.	The game character died.	TRUE
Game over	1	When the game is over.	The user cannot operate the game character.	TRUE
	2	When the game is over.	User can exit the game.	TRUE

4.9 Sprint 4 Use Case

4.9.1 User Passes the Level

Use case	User passes the level
Summary	The user controls the game character through a series of actions to pass through the level composed of monsters, walls, and traps, and try to avoid the death of the game character and prolong the survival time in the game.
Actor	User (anyone playing the game)
Trigger	User enters the game
Primary scenario	1.The user clicks to enter the game [alternative scenario: the user exits the game directly]. 2.After the user enters the game, control the game character to run forward [Alternative scenario: the user stops controlling the game character] [Exceptional scenario: the user cannot control the game character]. 3.The game character keeps running [alternative scenario: the game character encounters a wall and jumps] [alternative scenario: the game character encounters a wall and dash] [alternative scenario: the game character encounters a monster and jumps] [alternative scenario: the game character encounters Trap and jump] [Exception: the game character did not encounter any walls，monsters or traps.].

	4.The game character touches a monster, trap or wall, and the game character dies. [Alternative scenario: The game character crosses the wall, monster or trap and returns to the main scene 3]. 5.After the game character dies, the user score is displayed. [Exceptional scenario: the game ends but the score is not displayed].
Alternative scenario	If the user exits the game. 1.The user can re-enter the game and return to the primary scenario 2. The user stops controlling the game character. 1.The user can continue to control the game character. Return to the primary scenario 3. If the game character hits the wall and jump. 1.The game character jumps over the wall and return to the main scenario 3. [Alternative scenario: the user does not cross the wall, return to the primary scenario 4]. If the game character encounters wall and dash. 1.The game character dash passes the wall and continues to run [Alternative scenario: the user does not pass the wall, return to the primary scenario 4]. If game character encounters a monster and jumps. 1.The game character jumps over the monsters l and continues to run [Alternative scenario: the user does not cross the monsters , return to the main scenario 4][Alternative scenario: The game character jumps to the wall.]. If the game character encounters a trap and jumps. 1.The game character jumps over the trap and continues to run [Alternative scenario: the learning character does not pass the trap. Return to the primary scenario 4] [Alternative scenario: The game character jumps to the wall.]. If the game character jumps to the wall. 1. Return to the main scenario 3.
Exceptional scenario	The user cannot control the game character. 1.The user exits the game and tries to re-enter. Return to primary scenario 1. Users cannot encounter wall, monsters or traps.

	1. The user exits the game and tries to re-enter. Return to primary scenario 1. End of game score is not displayed. The user exits the game and tries to re-enter. Return to primary scenario 1.
Pre-conditions	start the game
Post-conditions	Score
Assumptions	None

4.9.2 Game Over

Use case	Game over
Summary	When the user knows his final score, the game ends.
Actor	User (anyone playing the game)
Trigger	User view score
Primary scenario	1. User view score. 2. After the user checks the score, it shows that the game is over. [Exceptional scenario: Do not show game over].
Alternative scenario	None
Exceptional scenario	Do not show game over. 1. User exits game.
Pre-conditions	score
Post-conditions	None
Assumptions	None

4.10 Tests for Use Case

4.10.1 Test of User Passes the Level

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user clicks to enter the game.	F	F
	2	After the user enters the game, control the game character to run forward.	F	F
	3	The game character keeps running.	F	F
	4	The game character touches a monster, trap or wall, and the game character dies.	F	F
	5	After the game character dies, the user score is displayed.	The character kept running after the score showed.	Adding new statement called death. When the death state is true, the character would stop running. Adding death animation to replace the running animation after death.
Alternative scenario	1	If the user exits the game directly.	F	F
	2	The user stops controlling the game character.	F	F
	3	If the game character hits the wall and jump.	F	F
	4	If the game character encounters wall and dash.	F	F

	5	If game character encounters a monster and jumps.	F	F
	6	If the game character encounters a trap and jumps.	F	F
	7	If the game character jumps to the wall.	The character would sometimes crash on the wall and stop moving forward.	The problem happened because of the box collider size. resize the box collider.

4.10.2 Test of Game Over Function

Test items	Case	Scenario description	Accidents	Solution
Primary scenario	1	User view score.	F	F
	2	After the user checks the score, it shows that the game is over.	F	F
Alternative scenario	1	-	F	F

4.11 User Interface

For the current Sprint the aim was to provide the user with a further two main features. These were to display the 'Game Over' message in the game, and to allow the user to pass the current level by proving adequate tiles for the user to just about manage to navigate through the course. These features present themselves in the user interface in the following format.

Game Over:

In the previous Sprint, the function for the death of the character was implemented when he makes contact with a monster or a trap. This signals the death of the player and the end of the game (Figure 12). To make it clear this has occurred, 'Game Over' will be displayed in large text centre screen.

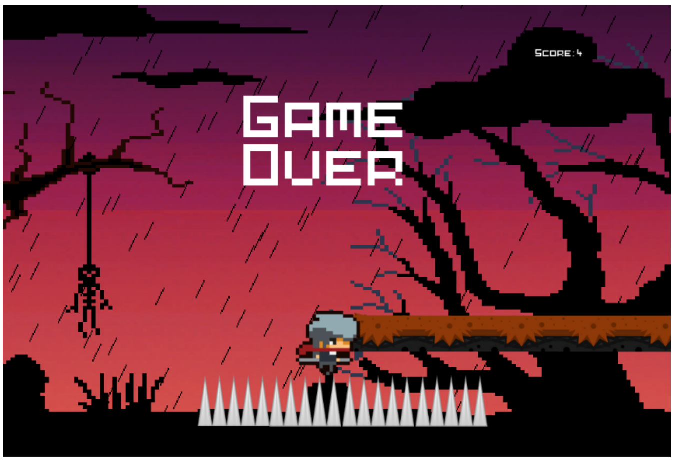
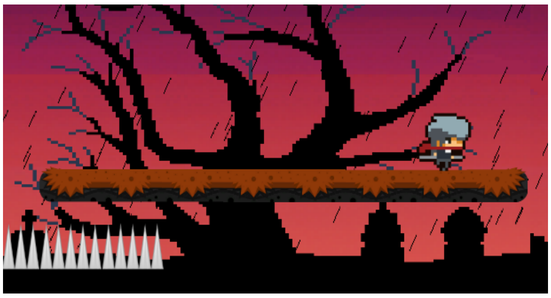


Figure 12: Game Over

Level Passing:

A range of tiles were fabricated that allow the player to jump upon, run along, under and to traverse dangers such as the monsters and traps that are implemented to kill the player (Figure 13). For the player to successfully navigate through the level, he will need to use these tiles to his advantage in the avoidance of obstacles. A selection of these tiles is shown below in original format and one that has been implemented in the game to allow avoidance of a trap (Figure 14).



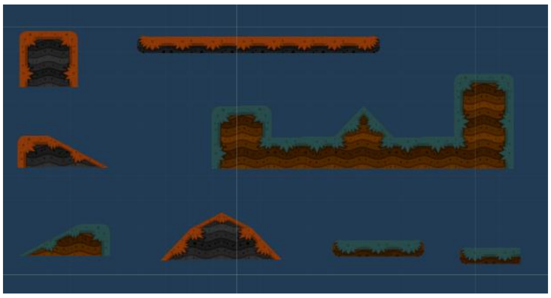


Figure 13: Prefabs of Tiles

Figure 14: A selection from tiles

4.12 Sprint 4 Exception Handling

Exception	Time of occurrence	Cause of the exception	Solution
-	-	-	-

Sprint 5

5.1 Sprint 5 Overview

In the design of the last sprint, there was a contentious question. Should we let the game pass the game at the end or in an infinite scene where users can keep running forward? Finally, considering the actual situation, if the game has a clearance function, the highest score is fixed. This is very unfair to players, so we designed the game as a level loop mode. Players can keep playing the game in the scene to refresh their records. Also designed in this sprint is the re-game. Taking into account the actual situation of the player, the player should be able to re-play instead of exiting the game and re-entering after losing a game.

When editing the code, the realization of this function is easier than imagined. Only need to make two more scenes, and the end of the game scene triggers to enter the next scene. The final function is completed by everyone.

5.2 Sprint 5 Review

THE 1ST REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-0 Approval of Management Plan		
Have all the Project Activities been well defined?	√	
Has Mission Need Statement been well prepared and explained technically and functionally?	√	
Have a Meeting Session Arrangement well arranged?	√	
Progress Monitor Function is defined clearly? Progress Reviews?	√	
Have the tools for development been well listed and declared?	√	
The Feasibility?	√	

CD-1 Approval of QA		
All quality assurances (for text file as well as software) are well defined?	√	
Review actions for quality assurance are well defined?	√	
A Good definition of Coding and Text file style?	√	
The Feasibility?	√	
CD-2 Approval of Reviews		
Has the review session been well defined?	√	
Has a full review circle been well arranged?	√	
Has the review rules and details been well contained?	√	
CD-3 Approval of Version Control		
The tools that are going to be used are well declared?	√	
The ways to keep different kinds of files are well declared?	√	
Does the plan clearly arrange the development productions?	√	
Does the plan avoid the risky and insecurity in development?	√	
CD-4 Software tests, fault and bug reporting		
Contains clearly testing methods?	√	
Contains clearly debugging reviews?	√	

Contains clearly bug report sessions?	√	
Whether the report keeping actions and managements well?	√	
Comments: Well structured. **Reviewer: Shuai Ning**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-5 Approval of 1st Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions been defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: The group work is fully distributed. **Reviewer: Shuai Ning** **Date: 30th November 2020**		
THE 2ND REVIEW APPROVAL		
KEY QUESTIONS FOR ALL BASIC PLANS REVIEW AND APPROVAL	**YES**	**NO**
CD-6 Approval of Development Progress Comparison		
A detailed comparison is made?	√	

An overall evaluation is made?	√	
A list between planned progress and real progress is made?	√	
A discus session on the solutions?	√	
A well-done record of the session?	√	
Comments: The level loop and restart function are under development. **Reviewer: Shuai Ning**		
CD-7 Approval of Development Questions		
Have actual questions reported clearly?	√	
Have all problems solved?	√	
Comments: Level loop function completed. **All Details of the Problems and Solutions should be recorded below:** **Problems:** None **Solutions:** None **Reviewer: Shuai Ning**		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-8 Approval of 2nd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	

Have all missions in this meeting well finished?	√	

Comments:

Team documentation work is in progress without exception.

Reviewer: Shuai Ning

Date: 4th December 2020

THE 3rd REVIEW APPROVAL

KEY QUESTIONS FOR FINAL EVALUATION	YES	NO
CD-9 Approval of Final Evaluation		
Whether the sprint is full completeness?	√	
Full documents recorded?	√	
The sprint finished in good condition?	√	
A Final Evaluation?	√	

Comments:

The sprint process has been completed.

Reviewer: Shuai Ning

CD-10 Approval of Debugging		
Has a final test passed?	√	
Have a Debugging track recorded?	√	
All bugs fixed?	√	

Comments:

Restart function completed.

Reviewer: Shuai Ning		
KEY QUESTIONS FOR MEETING REVIEW	**YES**	**NO**
CD-11 Approval of 3rd Review Meeting		
Have an action list made after the session?	√	
Have all members attended the meeting?	√	
Have all missions be defined and separated to all members clearly?	√	
Have all missions in this meeting well finished?	√	
Comments: All missions were well finished. **Reviewer: Shuai Ning**		
Sprint 5 Summary: In the development of sprint5, this is the final step to complete all game functions. The team members need to make new game scenes, which is somewhat difficult. But we are finally done. Then we realized that before the final release of a software development, targeted testing should be done. It is designed to ensure that there is no problem with the user after leaving the programming environment.		
Date: 6th December 2020		

5.3 Sprint 5 Customer Meeting Record

Customer Meeting Record		
Client Name: Julian **Project Start Date**: 25.10.2020		**Project Title**: BitDash **Project Deadline**: 11.12.2020
Scrum no: 5	**Meeting Scheduled at:** 12:00-12:30	**Date**: 27.11.2020
This week the following ideas were presented to the client in the continuing format:		

Response to the last meeting:

First level design is now in a playable state with new death and scoring functions added.

Physics in the game have been altered to be more realistic at clients' request.

Dash function has been implemented into the main game following last weeks demo.

Current game state:

A variety of video demos were presented demonstrating the new functions that have been added to the game that included the ability to walk on tiles, the revised physics, and the death and scoring functions.

Where we plan to go in the next sprint:

Focus on the next sprint will be on extension of the existing game to provide longer levels.

One idea that the team is floating is the possibility of infinite levels.

Another idea is the auto-generation of new levels.

The main branch of the current game in unity was demonstrated to show the game in its entirety.

Analysation

The pre-meeting team seemed to focus quite strongly on the user manual as part of the documentation and highlighted that we should be thinking about this at this stage of the project. We informed that this will be arranged and implemented in our big Monday meeting where we delegate the works for the upcoming Sprint.

The client seemed to be happy with the presentation and with the responses to last weeks meeting. Although very little actual feedback was given, there were no objections to any of the current implementations. Additionally, all points that the client had addressed in response to last weeks presentation had been implemented/resolved.

The Client enquired about documentation and a demonstration of our current documents was given, the client brought up whether we had attempted the document exports that were advised in our previous meeting and we advised this had been tested and we had exported and backed up current documents.

The client echoed the concerns in the pre-meeting about the progress of the game. Level length and play time seemed to be key concerns. The team does feel that we are in good stead in this regard and that we have time to complete the additional.

5.4 Sprint 5 Daily Scrum Meeting Record

Daily Scrum Meeting Record

Sprint 5: 30.11.2020 - 6.12.2020

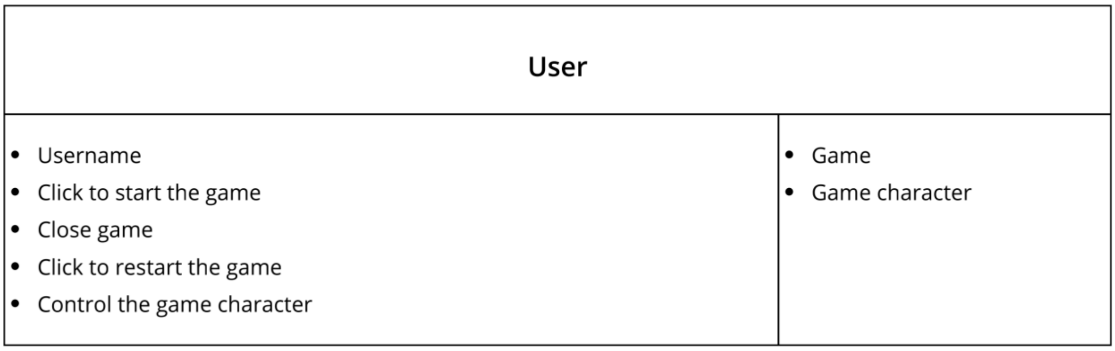
Date	Time	Members	Assigned Task	Summary
30th Nov	15:15-15:45	All team members	Function classification of motion module.	Make restart function and level loop function according to Backlog.
1st Dec	15:15-15:45	All team members	Establish a corresponding Unity branch and assign functions to specific members.	All members have progressed without exception.
2nd Dec	15:15-15:45	All team members	Realize the re-start function.	All members have progressed without exception.
3rd Dec	14:15-14:45	All team members	Implement level loop function.	All members have progressed without exception.
4th Dec	12:45-13:15	All team members	Test by user story.	According to the user story without exception.
5th Dec	15:15-15:45	All team members	Test by use case.	According to the use case test without exception.
6th Dec	15:15-15:45	All team members	Sprint review and improve documentation.	Resolve functional conflicts and complete review and other documents.

5.5 Sprint 5 Backlog

Sprint 5	Item	Description	Priority
Sprint backlog	Level loop	As a user, I can go to the next level after passing the current level and keep looping. Until the game character dies. 1.End of level.	1
	Restart	As a user, when the game character dies in the game, I can directly restart the game.	1
Product backlog	-		

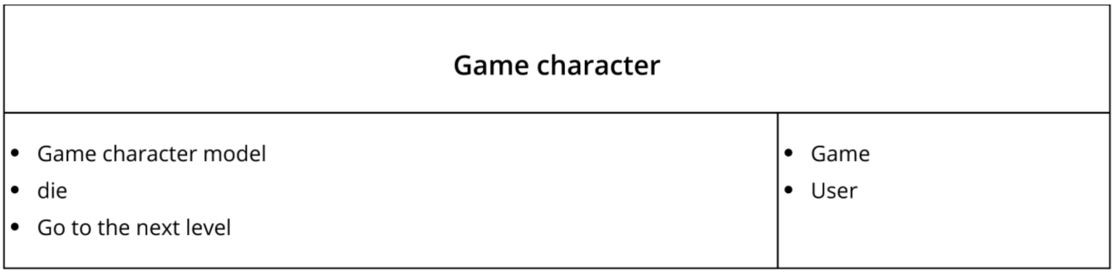
5.6 Sprint 5 CRC Cards

5.6.1 User



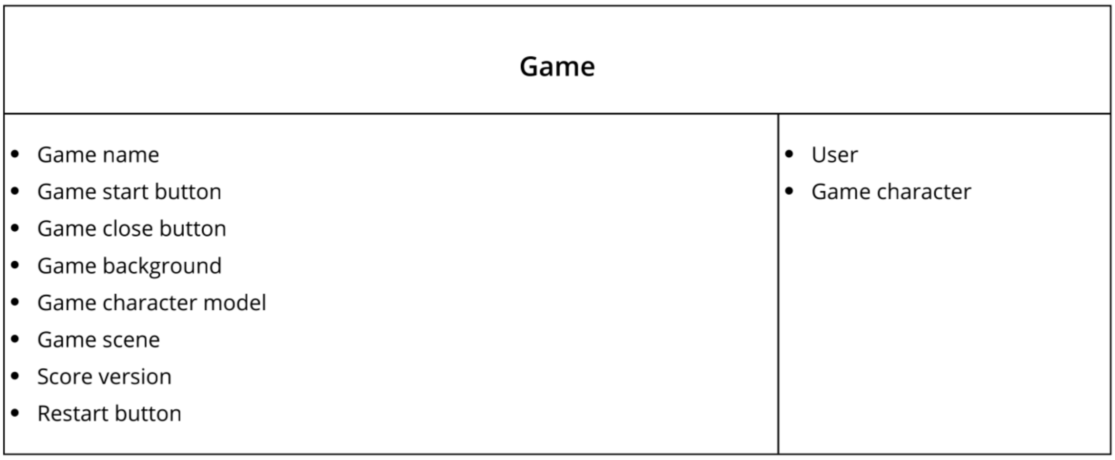
User	
• Username • Click to start the game • Close game • Click to restart the game • Control the game character	• Game • Game character

5.6.2 Game Character



Game character	
• Game character model • die • Go to the next level	• Game • User

5.6.3 Game



Game	
• Game name • Game start button • Game close button • Game background • Game character model • Game scene • Score version • Restart button	• User • Game character

5.7 Sprint 5 User Story

5.7.1 Level Loop

Title	Level loop
Story description	As a user, I can control the game character to enter the next level until the game character dies.
Rule description	Loop: Different game levels, when the user passes through one level, it switches to the next.
Testing standard	1. The user can control the game character to enter the next level. 2. Cannot enter the next level when the game character dies.
Design plan	Arrange a new game scene and keep the start stage to the end. Set up an invisible box when the game character touches the box and loads the next game level.
Checking list	The unity version is 2019.4.13f1c1.

5.7.2 Restart

Title	Restart
Story description	As a user, I can restart the game directly when the game character dies.
Rule description	None
Testing standard	1. When the game character dies, the restart button is displayed. 2. The user clicks the restart button and the game returns to the first level. 3. After the game returns to the first level, the user score is cleared.
Design plan	Restart: on click restart button and load scene 0.
Checking list	The unity version is 2019.4.13f1c1.

5.8 Tests for User Story

Test items	Case	When	Then	Result
Restart the game	1	The user manipulates the player and the player dies.	The user can see the restart button.	TRUE
	2	The user clicks the restart button.	The user returns to first level.	TRUE
	3	The user returns to the first level.	The user can play with game reset.	TRUE
Level loop	1	The user manipulates the player and let it reach the portal in the end.	The user can play in next level.	TRUE
	2	The user manipulates the player and the player die.	The user could not play in next level.	TRUE

5.9 Sprint 5 Use Case

5.9.1 Loop Level

Use case	Loop Level
Summary	When the user passes the current level, the user enters the next level.
Actor	User (anyone playing the game)
Trigger	The user touches the box at the end of the level.
Primary scenario	1.The game character reaches the end of the level. [Alternative scenario: Game character death]. 2.The game character enters the next game level. [Exceptional scenario: The game character did not enter the next level] [Exceptional scenario: User scores are not accumulated.].
Alternative scenario	Game character death 1.User restarts or exits the game.
Exceptional scenario	The game character did not enter the next level. 1.User re-enters the game. User scores are not accumulated. 1.User re-enters the game.
Pre-conditions	Movement
Post-conditions	Game character death
Assumptions	None

5.9.2 Restart

Use case	Restart
Summary	After the death of the game character, the user clicks in and restarts to return to the first level.
Actor	User (anyone playing the game)
Trigger	Game character death

Primary scenario	1. User-controlled game character death. 2. When the game character dies, the user can see the restart button. [Exceptional scenario: The restart button is not displayed]. The user clicks the restart button to return to the first level. [Exceptional scenario: User score is not cleared]. [Alternative scenario: the user clicks out the game].
Alternative scenario	User clicks to exit the game. 1. The user exits the game.
Exceptional scenario	The restart button is not displayed. 1.User re-enters the game. The user clicks out the game. 1.User re-enters the game.
Pre-conditions	Game character death
Post-conditions	Game over
Assumptions	None

5.10 Tests for Use Case

Test items	Case	Scenario Description	Accidents	Solution
Primary scenario	1	The user could play in next level after the player goes through the portal at the end of the level.	F	F
	2	The user could click the restart button to restart the game after player win.	F	F
	3	The user could play in a reset game from the first level after the restart button is clicked.	F	F
Alternative scenario	1	The user restarts or exit the game.	F	F

5.11 User Interface

The two functions in this sprint are level looping and game restart. The user can enter the next level after passing a level. After the death of the game character, the user can start a new game again.

Level loop:

The user enters the next level without any prompts, which is exactly the same as the previous level with different background (Figure 15&16).

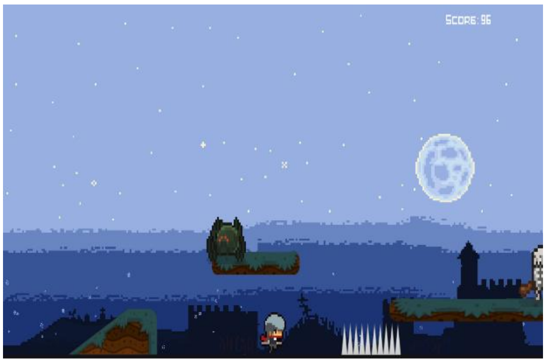


Figure 15: Level loop of BitDash

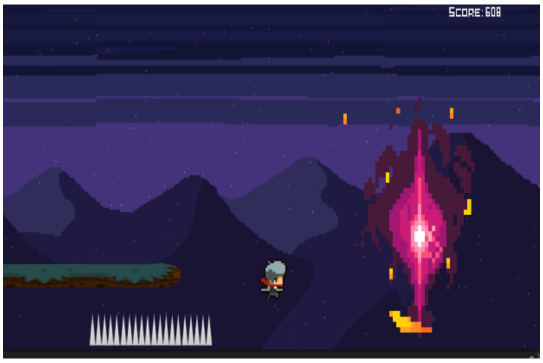


Figure 16: BitDash with different background

Restart:

The restart button is displayed when the user dies (Figure 17). After the user selects restart, the score will be cleared and return to the first level.

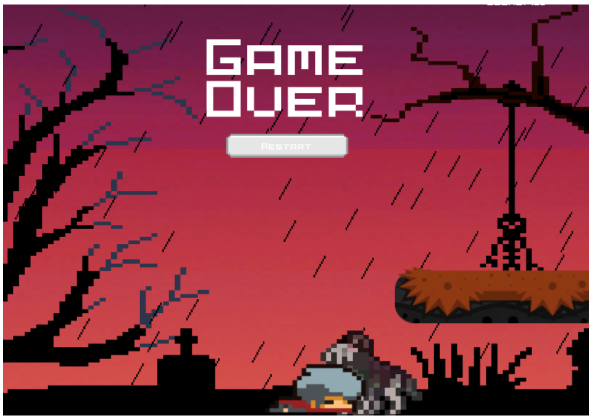


Figure 17: Restart menu of BitDash

5.12 Sprint 5 Exception handling

Exception	Time of occurrence	Cause of the exception	Solution
-	-	-	-

Sprint 6/ Post-game phase

6.1 Sprint 6 Overview

This part of the team has completed the development. All information has been edited. In order to be able to publish the game. At the end of this link, we test the overall function and logic and export the game into an installation package for players to download. The realization of functions is not involved in this part. It is just tests of document processing and the game.

6.2 Sprint 6 Integration Test

Test Platform	Test items	Case	When	Then	Result
Windows	**Test Module: Open the Application**				
	Open the Application	**1**	**The user clicks the BitDash.exe file**	**The game window opens**	**TRUE**
	Test Module: Start and Quit				
	Start the game	1	The user clicks the start game button.	The user can start the game.	TRUE
		2	When user enter the game.	The user can see the game background and game role model.	TRUE
		3	When the user enters the game.	Users can hear game background music	TRUE
	Quit game	1	When the user clicks the exit game button.	The game interface is closed, and the user returns to the computer desktop.	TRUE
		2	When the user closes the game.	The user can exit the game.	TRUE
	Test Module: Character Movement				

	Game character movement	1	When the user enters the game.	Users can control the movement of game characters.	TRUE
	Game character run	1	When the user enters the game.	The user can control the game character to run forward.	TRUE
	Game character jump	1	When the user enters the game.	The user can control the game character to jump up.	TRUE
	Game character dash	1	When the user enters the game.	Users can make the game character model smaller without interrupting running.	TRUE
	Test Module: Prefabs				
	The prefab of monsters	1	When the game starts.	The monster can be seen.	TRUE
		2	When a monster attacks the player.	The monster has a collision with the character causing the death of Game character.	TRUE
	The prefab of walls	1	When the game starts.	The wall can be seen.	TRUE
		2	When the player wants to jump to the wall.	The wall allows the character to jump on to and stand on it until the wall ends.	TRUE
		3	When the player does not jump to the wall.	The wall kept visible even the character missed jumping on to it.	TRUE

		4	When the player moves on the wall.	The wall allows the character to run on it without falling or bugs.	TRUE
	The prefab of traps	1	When the game starts.	The traps can be seen.	TRUE
		2	When the player touches the trap.	The trap has a collision with the character causing the game character to die.	TRUE
		3	Players avoid traps.	The trap kept visible no matter whether the character has been avoided or not.	TRUE
	Test Module: Level Pass and Game Over				
	Game character die	1	When Game character touches the trap.	Game characters will die.	TRUE
		2	When Game character touches the monster.	The game character will die.	TRUE
	Score	1	The user's living time increases.	The user's score increases.	TRUE
		2	When game character dies.	The score no longer increases.	TRUE
	User passes the level	1	The character avoids the monster or trap.	The game continues.	TRUE
		2	The game character hits traps or monsters.	The game character died.	TRUE
	Game over	1	When the game is over.	The user cannot operate the game character.	TRUE

	Test Module: Restart and Map Loop				
	Restart the game	1	The user manipulates the player and the player dies.	The user can see the restart button.	TRUE
		2	The user clicks the restart button.	The user returns to first level.	TRUE
		3	The user returns to the first level.	The user can play with game reset.	TRUE
	Level loop	1	The user manipulates the player and let it reach the portal in the end.	The user can play in next level.	TRUE
		2	The user manipulates the player and the player die.	The user could not play in next level.	TRUE
MacOS	**Test Module: Open the Application**				
	Open the Application	**1**	**The user hold on control and clicks the BitDash_Mac file**	**The game window opens**	**TRUE**
	Test Module: Start and Quit				
	Start the game	1	The user clicks the start game button.	The user can start the game.	TRUE
		2	When user enter the game.	The user can see the game background and game role model.	TRUE
		3	When the user enters the game.	Users can hear game background music.	TRUE
	Quit game	1	When the user clicks the exit game button.	The game interface is closed, and the user	TRUE

				returns to the computer desktop.	
		2	When the user closes the game.	The user can exit the game.	TRUE
	Test Module: Character Movement				
	Game character movement	1	When the user enters the game.	Users can control the movement of game characters.	TRUE
	Game character run	1	When the user enters the game.	The user can control the game character to run forward.	TRUE
	Game character jump	1	When the user enters the game.	The user can control the game character to jump up.	TRUE
	Game character dash	1	When the user enters the game.	Users can make the game character model smaller without interrupting running.	TRUE
	Test Module: Prefabs				
	The prefab of monsters	1	When the game starts.	The monster can be seen.	TRUE
		2	When a monster attacks the player.	The monster has a collision with the character causing the death of Game character.	TRUE
	The prefab of walls	1	When the game starts.	The wall can be seen.	TRUE
		2	When the player wants to jump to the wall.	The wall allows the character to jump on to and stand on it until the wall ends.	TRUE

		3	When the player does not jump to the wall.	The wall kept visible even the character missed jumping on to it.	TRUE
		4	When the player moves on the wall.	The wall allows the character to run on it without falling or bugs.	TRUE
	The prefab of traps	1	When the game starts.	The traps can be seen.	TRUE
		2	When the player touches the trap.	The trap has a collision with the character causing the game character to die.	TRUE
		3	Players avoid traps.	The trap kept visible no matter whether the character has been avoided or not.	TRUE
	Test Module: Level Pass and Game Over				
	Game character die	1	When Game character touches the trap.	Game characters will die.	TRUE
		2	When Game character touches the monster.	The game character will die.	TRUE
	Score	1	The user's living time increases.	The user's score increases.	TRUE
		2	When game character dies.	The score no longer increases.	TRUE
	User passes the level	1	The character avoids the monster or trap.	The game continues.	TRUE
		2	The game character hits traps or monsters.	The game character died.	TRUE

	Game over	1	When the game is over.	The user cannot operate the game character.	TRUE
	Test Module: Restart and Map Loop				
	Restart the game	1	The user manipulates the player and the player dies.	The user can see the restart button.	TRUE
		2	The user clicks the restart button.	The user returns to first level.	TRUE
		3	The user returns to the first level.	The user can play with game reset.	TRUE
	Level loop	1	The user manipulates the player and let it reach the portal in the end.	The user can play in next level.	TRUE
		2	The user manipulates the player and the player die.	The user could not play in next level.	TRUE

6.3 Sprint 6 Customer Meeting Record

Customer Meeting Record		
Client Name: Julian		**Project Title**: BitDash
Project Start Date: 25.10.2020		**Project Deadline**: 11.12.2020
Scrum no: 6	**Meeting Scheduled at:** 12:00-12:30	**Date**: 4.12.2020

The presentation this week presented the following ideas to the client:

In response to the feedback from last week:

There was a definite focus that the client wanted us to focus effort on ensuring all the documentation was in good order and so the team this week has been very focused on documentation. The documents prepped were: Reviews and CRC cards along with the usual Sprint paperwork for the current Sprint. Work was also carried out in regards of game testing and collating bugs for fixes in the final Sprint. The 2 main bugs highlighted this week were the Player still runs after death and that the score number increasing is not continuous. As an addition to the game, a number of additional levels were also prepared.

What the team have completed this Sprint:

As the final Sprint is approaching, the team felt it a good idea to present an overview of the documentation that has been collated for approval to the client. As CRC cards were a one area of documentation lacking until recently, the team demonstrated the work completed on these, which included cards for: User, Game Character, Game, Score Version, Traps, and Walls.

The focus for the next Sprint:

The presentation then iterated the teams focus for the upcoming final Sprint. This weekend will be focused on game development and testing. There are still some bug fixes to sort out. The team also still need to focus on the User Manual, Installation Guide, and Maintenance Guide which is a large part of the work required for the final Sprint. The team will conduct the final branch merge this coming weekend and assemble the final version of the game.

Analysation

Feedback from premeeting team was that the final document should be handed in as one large, collated document and that it should be in Sprint order.

Upon demonstration of the collated Sprint documents, the client mentioned that there was missing an overview of each Sprint, to give the reader an overview of what has happened in that week, this should then be followed by a weekly review that should then detail what has happened in response. It was highlighted that this should consist of a narrative review of the sprint comprising Overview followed by review…

The client made it very clear that he feels there is still a lot of work to do on this project. Although, the team did seem to feel they are in good stead to get the final product complete, to a high standard, by the handover date.

The client also enquired as to whether the team felt it would be productive to have final meeting before the handover. The team decided that one last customer meeting would be a good chance to have all the final documents ready and the game complete so we can show the client as a final check.

Overall, a productive and positive meeting with a clear route decided for the team to bring the project to a successful conclusion.

6.4 Sprint 6 Daily Scrum Meeting Record

Daily Scrum Meeting Record **Sprint 6: 7.12.2020 - 10.12.2020**				
Date	**Time**	**Members**	**Assigned Task**	**Summary**
7th Dec	15:15-15:45	All team members	Perfect the documentation and testing.	Bring all sprint contents from Jira into the formal documentations.
8th Dec	15:15-15:45	All team members	Perfect the documentation and codes and testing.	Finish the rest document works and perfect the implementation on codes.
9th Dec	15:15-15:45	All team members	Perfect the documentation and codes and testing.	Finish the rest work and check deficiency.
10th Dec	15:15-15:45	All team members	Perfect the documentation and codes and testing.	Finish the rest work and check deficiency, ready for the final delivery.

6.5 Sprint 6 Backlog

Sprint0	Start the game Exit the game
Sprint1	Game character run Game character jump Game character dash
Sprint2	Prefab wall Prefab monster Prefab trap
Sprint3	Game character death game score
Sprint4	Game over Design game levels
Sprint5	Level loop Game rrestart

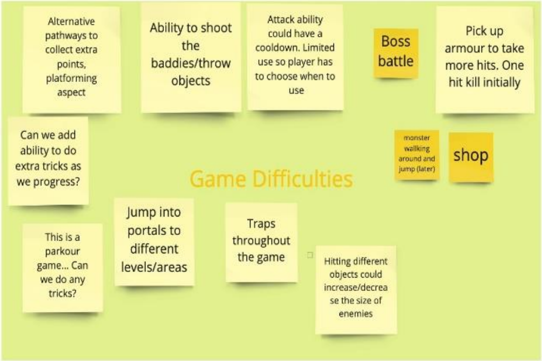
6.6 Sprint 6 Exception handling

Exception	Time of occurrence	Cause of the exception	Solution
-	-	-	-

Appendix A (Pre-game phase)

A.1 White Board for Brainstorms

We have used multiple tools to guarantee and assist with our agile process. Jira Software and Miro White Board were some of the most important tools. The following attachments are the roadmap created by Jira from pre-game phases to Sprint 6 and white boards for the development process.



Alternative pathways to collect extra points, platforming aspect

Ability to shoot the baddies/throw objects

Attack ability could have a cooldown. Limited use so player has to choose when to use

Boss battle

Pick up armour to take more hits. One hit kill initially

Can we add ability to do extra tricks as we progress?

monster walking around and jump (later)

shop

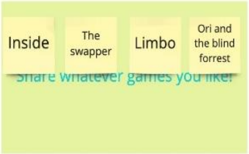
Game Difficulties

This is a parkour game... Can we do any tricks?

Jump into portals to different levels/areas

Traps throughout the game

Hitting different objects could increase/decrease the size of enemies



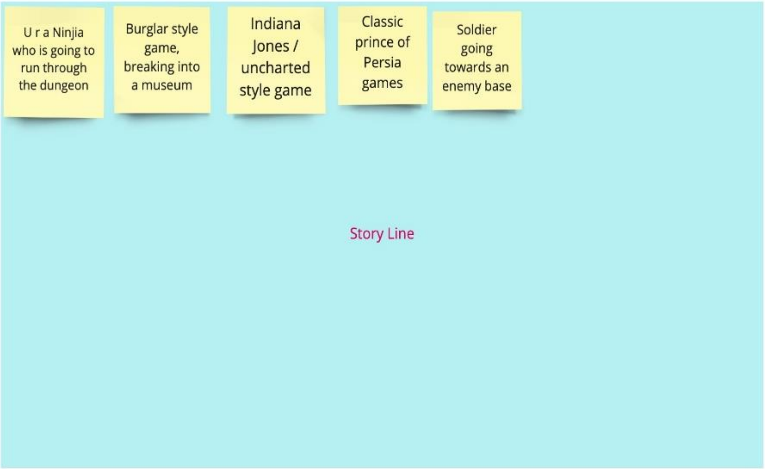
Inside

The swapper

Limbo

Ori and the blind forrest

Share whatever games you like!



U r a Ninjia who is going to run through the dungeon

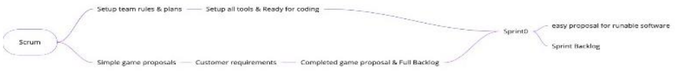
Burglar style game, breaking into a museum

Indiana Jones / uncharted style game

Classic prince of Persia games

Soldier going towards an enemy base

Story Line



Scrum

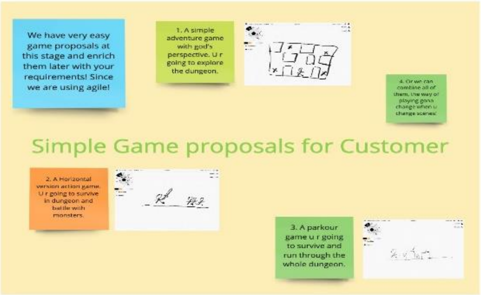
Setup team rules & plans — Setup all tools & Ready for coding

Simple game proposals — Customer requirements — Completed game proposal & Full Backlog

Sprint0

easy proposal for runable software

Sprint Backlog



We have very easy game proposals at this stage and enrich them later with your requirements! Since we are using agile!

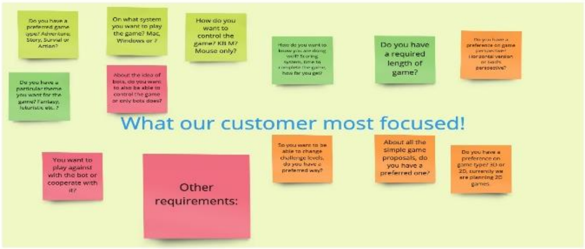
1. A simple adventure game with god's perspective. U r going to explore the dungeon.

4. Or we can combine all of them, the way of playing gona change when u change scenes!

Simple Game proposals for Customer

2. A Horizontal version action game. U r going to survive in dungeon and battle with monsters.

3. A parkour game u r going to survive and run through the whole dungeon.



Do you have a preferred game type? Adventure, Story, Survial or Action?

On what system you want to play the game? Mac, Windows or ?

How do you want to control the game? KB M? Mouse only?

How do you want to know you are doing well? Scoring system, time to complete the game, how far you get?

Do you have a required length of game?

Do you have a preference on game perspective? Horizontal version or god's perspective?

Do you have a particular theme you want for the game? Fantasy, futuristic etc. ?

About the idea of bots, do you want to also be able to control the game or only bots does?

What our customer most focused!

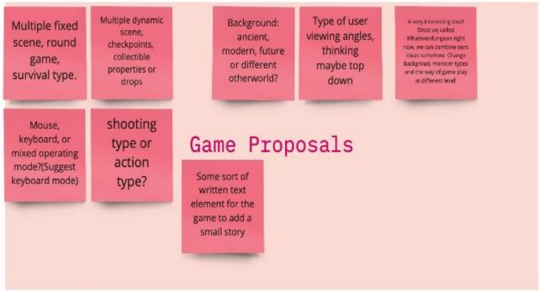
You want to play against with the bot or cooperate with it?

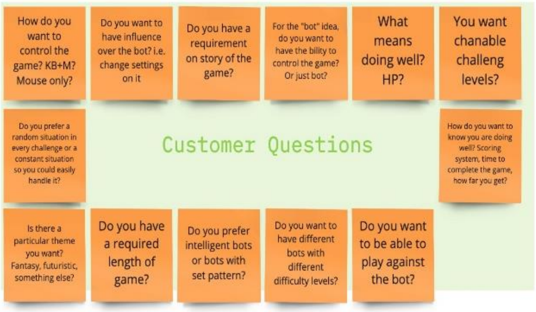
Other requirements:

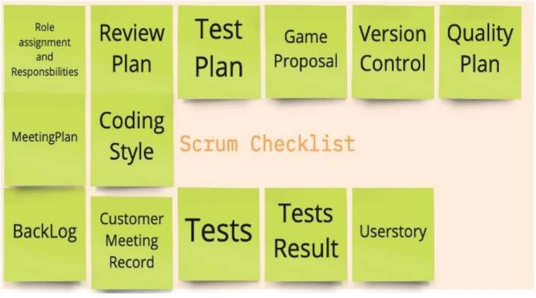
So you want to be able to change challenge levels, do you have a preferred way?

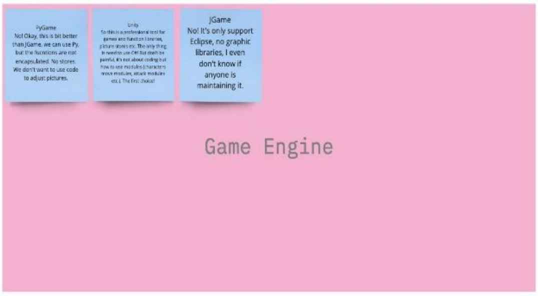
About all the simple game proposals, do you have a preferred one?

Do you have a preference on game type? 3D or 2D, currently we are planning 2D games


Multiple fixed scene, round game, survival type.
Multiple dynamic scene, checkpoints, collectible properties or drops
Background: ancient, modern, future or different otherworld?
Type of user viewing angles, thinking maybe top down
A very interesting idea! Since we called Whateverdungeon right now, we can combine ours ideas somehow. Change backgroud, monster types and the way of game play in different level
Mouse, keyboard, or mixed operating mode?(Suggest keyboard mode)
shooting type or action type?
Game Proposals
Some sort of written text element for the game to add a small story


How do you want to control the game? KB+M? Mouse only?
Do you want to have influence over the bot? i.e. change settings on it
Do you have a requirement on story of the game?
For the "bot" idea, do you want to have the bility to control the game? Or just bot?
What means doing well? HP?
You want chanable challeng levels?
Do you prefer a random situation in every challenge or a constant situation so you could easily handle it?
Customer Questions
How do you want to know you are doing well? Scoring system, time to complete the game, how far you get?
Is there a particular theme you want? Fantasy, futuristic, something else?
Do you have a required length of game?
Do you prefer intelligent bots or bots with set pattern?
Do you want to have different bots with different difficulty levels?
Do you want to be able to play against the bot?




PyGame
No! Okay, this is bit better than JGame, we can use Py, but the fucntions are not encapsulated. No stores. We don't want to use code to adjust pictures.
Unity
So this is a professional tool for games and function libraries, picture stores etc. The only thing is need to use C#! But don't be painful, it's not about coding but how to use modules (characters move modules, attack modules etc.). The First choice!
JGame
No! It's only support Eclipse, no graphic libraries, I even don't know if anyone is maintaining it.
Game Engine

A.2 Game proposal

According to the requirements of customers and referring to games of the same type, the analysis of the proposal for this project is as follows.

This is a parkour game. In this game, the user plays a ninja who has just escaped from the dungeon. He is in the background of deserted walls and dark forest. He needs to escape from this barren land. There are various obstacles, monsters and traps blocking him on the way to escape. He must use his agility to cross this wasteland.

The ninja must pass these levels carefully, because he will lose his life and end the game as long as he encounters danger. When the ninja dies, he will get points according to his progress.

For the player, the player should try to increase the survival time and running distance of the game character in the game to obtain higher scores.

A.3 Meeting plan

Introduction

The meeting plan is used to determine the meeting time of the group members, formulate the meeting process, record the meeting method, and summarize the tools needed in the meeting.

Time management

This item is used to determine the time of the meeting, including meetings between group members and meetings with mentors. The meeting schedule has been deposited in Jira. If a team member disagrees with the meeting time or is absent from the meeting due to personal reasons, other team members need to be notified in time through teams.

Meeting process

Personal work

Team members should summarize their work in the sprint and answer the following three questions:

1. What parts of the sprint plan have been completed by yourself. Which functions are implemented? What functions have not been completed by me?

2. During this sprint, what problems you encountered that affected your work, including technical problems (code problems when programming). Non-technical issues (such as the impact of other models, Rider will not be used) What kind of support is needed (including the help needed in programming technology, and what pre-parts are needed in the part that you are responsible for).

3. Adjustments to future work. Plan to complete which parts of the content in the next phase.

Get support

After the team members raise the questions, the solution to each problem is proposed in the meeting. How to support group members to complete their work.

Re-planning of the entire project

Collect new ideas that the team members think of while working and join the discussion. The team members can show the overall plan of the project accordingly.

After getting the work progress of the team members and the problems encountered in the work. According to the work progress and the analysis of the entire project, each member is assigned the work that needs to be completed in the next sprint (combined with the team members' own plans for future work.)

Meeting minutes

The team members will organize the meeting records. Record the work done by the team members. Record the problems encountered by the team members and their future plans. Record what needs to be completed in the next sprint.

When a group member is absent from the meeting, you can learn the contents of the current meeting through the meeting minutes.

Meeting records are also used to document the sprint. When the project process encounters problems, you can review the work of the team members to solve the problem. At the same time, the team members can view the meeting minutes at any time, and if there are problems in the records or they are deemed to need to be modified, they can be raised in time.

Meeting tools

- Meeting planning tool: Jira is used to determine the time for group meetings.
- Problem tool: GitHub is used to track the code of the team members and summarize the problems encountered in programming. At the same time, team members can review each other's work through GitHub. Miro is used to record the ideas proposed by the team members and the non-technical problems encountered at various stages.
- Communication tool: teams group members meet through teams.

A.4 Quality Plan

Introduction

This document explains the Quality Assurance Plan for the dungeon game.

Responsibilities

The responsibilities for all members include, but are not limited to:

○ Develop the requirement specification for the project.

○ Develop the cost estimation for the project

○ Develop the design plan for the project

○ Develop and maintain the software quality plan

○ Generate and maintain a schedule of software quality assurance activities

○ Develop the test plan and assess all aspects of quality issues. Implement and test the product and deliver the product along with the necessary documentation

○ Review the work and communicate the prioritised changes

Documentation Relating to Quality

This section identifies the other documentation governing the requirements, development, verification, validation and maintenance of the software.

○ Requirement

○ Meeting Plan

○ Meeting Minute

○ Version Control Plan

○ Software Tests Plan

○ Review Plan

○ Coding Style

○ Review Evidence

○ Integration Test Results

○ Daily Scrum Meeting Record

○ Customer Meeting Record

○ Product Backlog List

○ Role Assignment and Responsibility

Project Quality Assurance

This section identifies the quality assurance activities for each development process. All members are responsible for inspection in three areas:

1. Personal work

Everyone checks whether the increments produced during the sprint are in line with their own plan.

2. Demand response

Check whether the function of the product corresponds to the demand.

3. Personal summary

Whether there are deviations in the product functions that you are responsible for in the sprint, and whether your work plan fully corresponds to the needs.

Project Checkpoints

Each stage of development will have at least one formal checkpoint called a stage exit. When a stage has been successfully exited, it indicates that all draft deliverables due to date have been completed, all outstanding issues have acceptable action plans, and there is a sound plan for the remainder of the project (detailed for the next stage).

Product Identification & Traceability

The requirement specification will be used to check off the deliverables. The Project Review will ensure that each of the requirements is met by both the design and test functions.

Project Review

This section identifies the numbers and type of system reviews. Review will occur at the end of each development stage. For each review, members will assess the review deliverable to assure that the deliverable meets the requirements and communicate the prioritised changes. The further details are defined in the “Review Plan”.

Testing & Quality Check

Each group member will assure that the test management processes and products are being implemented per Test Plans. This includes all types of testing of software system components as described in the test plan. In addition, test results will be documented, addressed and tracked to discuss in the testing stages of the project. The further details are defined in the “Software Test Plan”.

Problem Reporting & Corrective Action

The “Issues” section in GitHub will be used to itemise, document, track and resolve the problem reported through the project.

Tools

Project Tools:

○ Miro

○ MySQL

○ GitHub

○ Unity Hub

○ Rider

○ Jira

○ Google Drive

Quality Tools:

○ Microsoft Office Tools (e.g. Word, PowerPoint, Excel) ○ Google Drive (Doc, Sheet, Slide)

○ Issues Tracker (GitHub)

A.5 Coding Style

Introduction

This document describes the coding style for the source code which is used in the Software Engineering Coursework 2

Formatting

a. Indentation

All indents will be four spaces.

Matching braces should always line up vertically in the same column, and each brace should be by itself on one line.

All if-else, while and for statements must use braces.

b. Spacing

Use a space around binary operators.

Commas and semicolons are always followed by a space.

All method names should be immediately followed by an open parenthesis '('.

c. Class Member Ordering

Field

Constructors

Methods

d. Max. Line Length

Lines should contain no more than 120 characters to maintain readability.

Identifiers

All identifiers must be meaningful names and use letters and numbers only. (A-Z, a-z, 0-9)

a. Class

All class identifiers are singular nouns and start with a capital letter.

b. Method & variables

All method and variable names should start with lowercase letters. Use uppercase for the first letter of each word in the middle to increase readability of compound identifiers.

c. Constant

The constant is written in uppercase.

d. Test Code

Use underscores in identifiers for methods and fields

Coding

a. Access

Specify all fields and methods as either private, public or protected. All fields must be private, except for some constants.

b. 'Break'

'break' is used only for switch statement control.

Documentation

a. Class Comment

Every class must have a comment at the top containing at least a general description of the class, author name and version number

b. Method Comment

Every method must have a method comment

c. Code Comment

Only comment in the code where the code is difficult to understand.

A.6 Test Plan

Introduction

The document contains the description of test plan for Project [Game Name]. A good software design should meet the rule of loose coupling and strong cohesion. It gives us a better-designed code that is easier to maintain.

During the software development process, developers should look for defects early and repeatedly, there are several benefits in early testing. They could offer us valuable experience with a system, this can make it possible to spot problems early enough to fix them at a much lower cost, the later a defect is noticed and corrected, the more expensive it will be.

Also, we can build up a series of test cases and results that can be used repeatedly as the program is developed (regression tests). The test cases will allow us to check we have not carelessly made errors into the rest of the system as a result of the changes.

Testing

In this section, black-box testing could meet our needs. Black-box testing will be designed to check if each function could produce out expected output. Before the test, we can also use software inspection to analyze and check the system requirements, design models, the program source code. Inspections mostly focus on the source code of a system, but any readable representation of the software, such as its requirements or a design model, can be inspected. There are 2 advantages of software inspection:

1. If an error occurred in the testing, that error leads to unwanted outputs. you can never be sure if later output anomalies are due to a new error or are side effects of the original error. And inspection does not involve executing the system, so no need to worry about interactions between errors.

2. Un-finished versions of a system can be inspected without additional costs. If program is incomplete, then you need to develop specialized test harnesses to test the parts that are available. This obviously adds to the system development costs.

During the test, testers should have made at least one test for every requirement in the requirements document and we should design the tests to provide coverage of all the features of the object. This means that we should test all operations associated with the object. And we will include all the use cases for black-box testing. One of the most important benefits of black-box testing is that it starts from the user’s point of view, it will be easy to know which features that user will use and which problems they will encounter.

Also, it is based on the software development requirements, we can also know which functions in the documents have been implemented. And the equivalence class partitioning (ECP) will be used in the test. We will not start the next coding part until the previous code passed all its tests. Here are the ECP rules we need to consider when we do the tests:

1. When the input condition specifies the value range or the number of values, a valid equivalence class and an invalid equivalence class can be established.

2. Under the condition that the input condition specifies the set of input values or the condition of "must follow", an effective equivalence class and multiple invalid equivalence classes can be established.

3. When the input condition is a Boolean, a valid equivalence class and an invalid equivalence class can be determined

4. Under the condition of knowing equivalence classes that have been divided and each element is treated differently in the program. The equivalence class is supposed to be further divided into smaller equivalence classes. In the next step, if all classes conform to the passing requirements of the test cases, integration testing (check that several software units/components work well together) will be executed to test the whole use case. All the test results will be recorded in the following test log template.

Test Log Template

ID	[Module name] *E.g. CharacterMovements*
Sprint	[Sprint name] *e.g. Sprint 1*
Test Function	The function of module] *e.g. Movements of character*
Test Objective	[Objective] *e.g. Make sure to let character move in 4 directions with keyboard*
Description	[A more detailed description] *e.g. When press key "go up", character goes up; when press key "go right", character go right; …… when press key "go up" and "go right" at the same time, the character go up right directly; ……*
Pass/Fail	[Check if the function meets the objective] *e.g. Pass*
Time	[Test time] *e.g. 26/12/2020*
Tester	[Tester name]

Debugging

Software debugging is the work that starts after a successful test. It is different from software testing. The task of debugging is to further diagnose and correct potential errors in the program. The debugging activity consists of two parts: determine the exact nature and location of suspicious errors in the program, modify the program (design, coding) to eliminate this error.

Debug can help us clear the flow of a complex program to help read the code. Luckily, we could simply use "Devtools" inside Chrome to debug. During the process of debugging, the developer could set a breakpoint at that they want to start through "Event Listener Breakpoints" and use "Step" functions to walk through the program. The breakpoint is to be placed on the left side of a statement in the body of the method that may be problematic. Developers can hit multiple breakpoints, but the less the better. Breakpoints can be added before Debug runs and can be added at runtime.

When we find the bug, the record of the bug is absolutely needed, and we should explain how exactly the product is broken. We will use the following bug report template to keep it being recorded. The point of the writing bug report is to get bugs fixed. During the bug report, “where did the bug take place”, “when did it happen”, “who made that bug”, and “what happened” must be written in it.

In addition, the steps to reproduce is the most important part of the bug report. If the developer is able to reproduce the bug, the bug is very likely to be fixed. If the steps are unclear, it might even be possible to know whether the bug has been fixed. The precise steps to reproduce the bug are essential.

Developers should indicate whether you can reproduce the bug at will, occasionally, or not at all and precisely describe the observed result and the expected result. Clearly separate facts from speculations. All in all, the debugging process should follow this workflow. Find them → Document them →Reproduce them →Fix them Bug Reports Template

<u>Bug Reports Template</u>

ID	ID in order e.g.#1
Bug Name	
Reporter	
Submit Date	DD/MM/YYYY
Description	
URL	
Screenshot	
Platform	
Operating System	
Browser	
Severity	
Assigned to	
Priority	

A.7 Version Control

Introduction

Since version control is a vital part of any project, it is extremely important that we keep a record of how we plan to carry out version control. Version control creates save points as work is done by the team and in the event of failure, the project can be reverted to the last point when it was working correctly. It is the purpose of this document to formally layout the process of version control throughout the project BitDash.

Centralised Repository

With regards to using a centralised repository, the team will be using GitHub, as this will allow the team to collaborate easily on the project files. It will record the changes made to files over time and can allow for different team members to work on the project at the same time without confusion and creating multiple documents. It will also allow us to be able to revert back to a previous version if needed. Since we will be working over several time zones, the files are accessible to all members at any time.

Commit

Save points, also known as commits in git will also be used in the project and will consist of a detailed message marking the amendments made. Documents should be committed to the central depository when code compiles and also passes our test requirements set out during the project. If code is incomplete and does not compile, it should still be pushed onto the central repository, however it must be put in a separate branch until it passes our tests, then can be added to the master branch.

Branches

Furthermore, git enables the use of the 'branch' function. This is extremely beneficial to those engaging in incremental work, just as we shall be doing during the project, as our development process will be using the agile methodology. A branch is a linked list of commits which grows as new commits are continually added. Using branches, team members can split off from the master branch to develop new features, without the risk of disturbing the flow of work in the master branch. This means it is ideal for any team members working simultaneously, allowing for new features to be added, tested and either added back to the master branch if it works or removed if not. If merging the new feature back into the master branch, GitHub gives the option to open a discussion between the parties involved. Also, if a problem arises during the merging process, then GitHub tells the team what the problems are and how to fix them.

Contribution

Using version control means that contributions can be attributed to individuals or programming pairs in the team. This feature means that communication channels are made more open to see who made changes and why something is being changed.

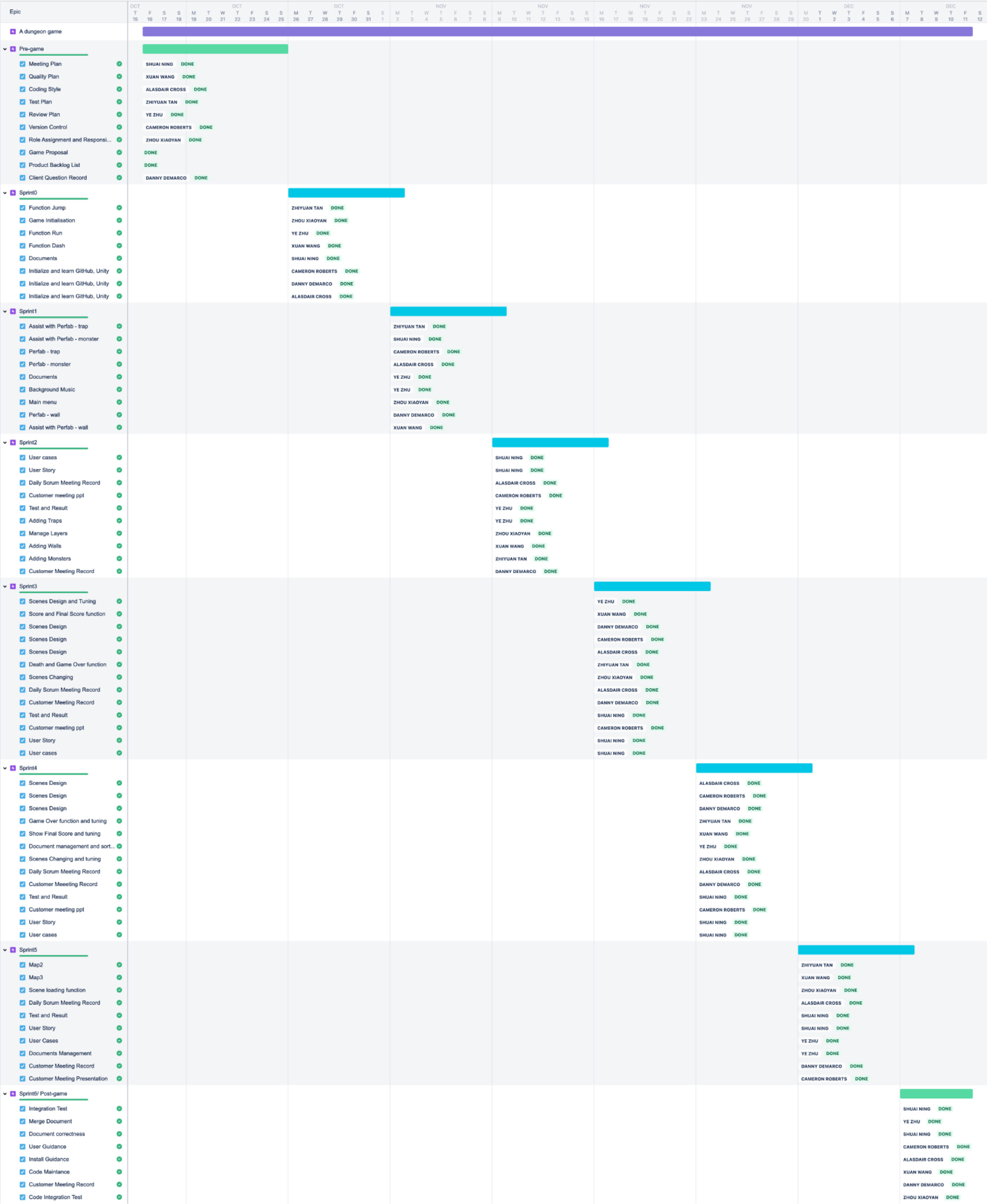
Versions

Additionally, versions for the project means that any member can retrieve any document or code from any point in time. This is extremely beneficial as code or documents can be improved and amended by the rest of the team from a singular central repository.

Documentation

For the process of documentation of the project, the team has decided upon google docs for most of the written documentation. This is mainly due to its versatility and accessibility for all team members. It provides version history, and edits to the documents are stored, allowing all members to look back through revision history, making it easy to see what changes have been made and by who. Google docs also supports comments, meaning that we can provide quality

Appendix B: Roadmap for BitDash



Epic		
A dungeon game		
Pre-game		
☑ Meeting Plan	SHUAI NING	DONE
☑ Quality Plan	XUAN WANG	DONE
☑ Coding Style	ALASDAIR CROSS	DONE
☑ Test Plan	ZHIYUAN TAN	DONE
☑ Review Plan	YE ZHU	DONE
☑ Version Control	CAMERON ROBERTS	DONE
☑ Role Assignment and Responsi...	ZHOU XIAOYAN	DONE
☑ Game Proposal		DONE
☑ Product Backlog List		DONE
☑ Client Question Record	DANNY DEMARCO	DONE
Sprint0		
☑ Function Jump	ZHIYUAN TAN	DONE
☑ Game Initialisation	ZHOU XIAOYAN	DONE
☑ Function Run	YE ZHU	DONE
☑ Function Dash	XUAN WANG	DONE
☑ Documents	SHUAI NING	DONE
☑ Initialize and learn GitHub, Unity	CAMERON ROBERTS	DONE
☑ Initialize and learn GitHub, Unity	DANNY DEMARCO	DONE
☑ Initialize and learn GitHub, Unity	ALASDAIR CROSS	DONE
Sprint1		
☑ Assist with Perfab - trap	ZHIYUAN TAN	DONE
☑ Assist with Perfab - monster	SHUAI NING	DONE
☑ Perfab - trap	CAMERON ROBERTS	DONE
☑ Perfab - monster	ALASDAIR CROSS	DONE
☑ Documents	YE ZHU	DONE
☑ Background Music	YE ZHU	DONE
☑ Main menu	ZHOU XIAOYAN	DONE
☑ Perfab - wall	DANNY DEMARCO	DONE
☑ Assist with Perfab - wall	XUAN WANG	DONE
Sprint2		
☑ User cases	SHUAI NING	DONE
☑ User Story	SHUAI NING	DONE
☑ Daily Scrum Meeting Record	ALASDAIR CROSS	DONE
☑ Customer meeting ppt	CAMERON ROBERTS	DONE
☑ Test and Result	YE ZHU	DONE
☑ Adding Traps	YE ZHU	DONE
☑ Manage Layers	ZHOU XIAOYAN	DONE
☑ Adding Walls	XUAN WANG	DONE
☑ Adding Monsters	ZHIYUAN TAN	DONE
☑ Customer Meeting Record	DANNY DEMARCO	DONE
Sprint3		
☑ Scenes Design and Tuning	YE ZHU	DONE
☑ Score and Final Score function	XUAN WANG	DONE
☑ Scenes Design	DANNY DEMARCO	DONE
☑ Scenes Design	CAMERON ROBERTS	DONE
☑ Scenes Design	ALASDAIR CROSS	DONE
☑ Death and Game Over function	ZHIYUAN TAN	DONE
☑ Scenes Changing	ZHOU XIAOYAN	DONE
☑ Daily Scrum Meeting Record	ALASDAIR CROSS	DONE
☑ Customer Meeting Record	DANNY DEMARCO	DONE
☑ Test and Result	SHUAI NING	DONE
☑ Customer meeting ppt	CAMERON ROBERTS	DONE
☑ User Story	SHUAI NING	DONE
☑ User cases	SHUAI NING	DONE
Sprint4		
☑ Scenes Design	ALASDAIR CROSS	DONE
☑ Scenes Design	CAMERON ROBERTS	DONE
☑ Scenes Design	DANNY DEMARCO	DONE
☑ Game Over function and tuning	ZHIYUAN TAN	DONE
☑ Show Final Score and tuning	XUAN WANG	DONE
☑ Document management and sort...	YE ZHU	DONE
☑ Scenes Changing and tuning	ZHOU XIAOYAN	DONE
☑ Daily Scrum Meeting Record	ALASDAIR CROSS	DONE
☑ Customer Meeeting Record	DANNY DEMARCO	DONE
☑ Test and Result	SHUAI NING	DONE
☑ Customer meeting ppt	CAMERON ROBERTS	DONE
☑ User Story	SHUAI NING	DONE
☑ User cases	SHUAI NING	DONE
Sprint5		
☑ Map2	ZHIYUAN TAN	DONE
☑ Map3	XUAN WANG	DONE
☑ Scene loading function	ZHOU XIAOYAN	DONE
☑ Daily Scrum Meeting Record	ALASDAIR CROSS	DONE
☑ Test and Result	SHUAI NING	DONE
☑ User Story	SHUAI NING	DONE
☑ User Cases	YE ZHU	DONE
☑ Documents Management	YE ZHU	DONE
☑ Customer Meeting Record	DANNY DEMARCO	DONE
☑ Customer Meeting Presentation	CAMERON ROBERTS	DONE
Sprint6/ Post-game		
☑ Integration Test	SHUAI NING	DONE
☑ Merge Document	YE ZHU	DONE
☑ Document correctness	SHUAI NING	DONE
☑ User Guidance	CAMERON ROBERTS	DONE
☑ Install Guidance	ALASDAIR CROSS	DONE
☑ Code Maintance	XUAN WANG	DONE
☑ Customer Meeting Record	DANNY DEMARCO	DONE
☑ Code Integration Test	ZHOU XIAOYAN	DONE