CSC2044 Software Development Processes and Practice

Project Submission Report

This document will provide you with really important information on how your individual contribution will be determined.

At the end of what has hopefully been an educational and fun experience, albeit one that might also have been hectic or demanding, the final submission completes the project development within this phase of the module. The following section should be carefully scrutinised as it holds key questions and the information you provide will enable me to best assess your project.

Section 1

Your team

Please provide team details below. Note, you should include your QUB email address (*please do double check you've entered in the correct email address*)

Name	Student Number	QUB Email Address
Chloe McMullan		
Courtney Shek		
Michael Grundie		
Jodie Burnside		
Dariusz Jerzewski		
Chloe McMullan		

Section 1

Your submission

Please detail the name(s) of the GitLab project(s) containing your submission.
Important: If you have submitted several different projects, e.g. card game and deck editor, or, projects from different team members, then it is essential each project is detailed below.
Important: If your project(s) contains branches off the master that you would like me to consider then it is essential these are also detailed.

What has been submitted?
csc2043-1617-56 : Master Branch csc2043-1617-56 : Courtney Branch

How can your card game be played?

- After you choose the map level, you appear in the tile-map, walk about and interact with the characters, some have advice.

- The red haired character is your enemy, walk into him and accept the battle challenge to start the game.(Characters generate randomly but quickly, walk about until the enemy pops up)

- Your hand is at the bottom of the main battle screen, above is your bench then active. Above them is the AI's active and bench. You can not see the AI's hand on purpose.

- Your cards start off unevolved; they display the possible stats for each school's evolve.
  - Shools:
    - Humanitarian - Yellow (Has 10 defence against STEM).
    - Medical – Red (Has 10 defence against Humanitarian).
    - STEM – Blue (Has 10 defence agains Medical).

- Once you evolve a unimon card with a school card you will only see its evolved ability and basic ability.
  - Drag and drop a school card onto an unevolved unimon card to evolve it.

- Each time you want to use your evolved ability you have to charge it, you can use any school card for this no matter the evolved to school. To an evolved Unimon card, school cards become energy cards. Charged unimon cards display a flashing blue border.
  - Again drag and drop a school card onto an evolved unimon card to charge.

- There is a limit of 1 evolve and 1 charge per turn!

- Setup Turn: You can not attack on your first turn, simply fill you bench, active and use up your turn evolve and charge if you have any school cards. The game will detect when you have done this.

- You can enlarge/select cards by long-pressing them.

- Attacking: Select your active card(long press) you will only see the button for your basic attack when the card is unevolved or when the card is evolved but uncharged or cooling down.
  - Your turn (after the setup turn) can not end until you attack or use your special. This is your turn ending move.

- Each school has a 10hp defence against one of the other schools. On an attack where the school defence has been used the defending school star will flash from the attacked card.

- When ordinary defence has been used, a shield will flash from the attacked card.

- When an attack has decreased a Unimon's health a health down animation will travel down the attacked card.

- Specials (evolved attack/ability):
  - Larger attack

  - Health increase
  - Defence (only works for the enemy's next turn then drops to 0 again).

- Some specials have cooldowns and some can only be used once (noted by '!')

- Stats from left to right = School / Special Type(attack,health,defence) / Cool-Down / Special Value (how much attack,defence,health).
- Retreating: Dragging a card back from active is free on the first turn, after that it costs you 1 witpoint, a pop-up will confirm that you want to. You can only retreat to an empty bench slot.

- Witpoints: You as a player have 20 witpoints, when a unimon is defeated, or when you retreat, you lose 1 witpoint. If you lose all your witpoints GAME OVER!

- End Game: You can play right up until your last unimon, if all your unimon are defeated before the enemy's, GAME OVER! Though you'll likely run out of witpoints first.

- Cool Things:
  - Dragging a card to the top/bottom of the viewport will make the viewport shift making it easier to reach card slots off the screen.
  - You can't drag to the AI's area.
  - Turn on your sound!
  - Check out our card animations.
  - The AI is progressively harder to beat (We think anyway).
  - The help guide in the pause menu explains which schools have a defence to other schools.
  - In the battlescreen class decrease the values of unimonPerDeck, and schoolCardsPerDeck to 6 and 10 to try a quick game. Or lower if you wish. Also you adjust other values such as evolve limit charge limit, game rules etc.
  - For testing: In the battle screen classes construtor uncomment the line

    ```
    setUpEndTurn();
    ```

    to enable a button that allows you to skip turns instantly. (Play it without this first.)
  - Check out our logging!

Courtney's Branch:

- The main screen should take you directly to the tile map, from there explore the world and talk to non-playable characters for helpful battle advice.
- But also be aware of your opponent who will be looking for a battle!
- (enemy)

Are there any known bugs in your game?

[[Overview of any known bugs in the game. This will help me understand and avoid them while playing the game (which often means I can see/play more of the game)]]

- AI sometimes places a card bottom left of the human hand. **Possibly fixed:**
- AI sometimes instantly moves middle card in AI bench without animation. Efforts to fix: change removeFromCardContainer method, changed moveToCardContainer(), added a workaround, tested assigning card to cardHolders at different points of the code, tried syncing in AIPerformMoves instead, tried setCardWithoutChangingPosition, added new method to CardHolderContainers to test if finding the empty container was breaking there. Seems to centre around the 2nd cardholder in the aibench – perhaps something to do with how AI is reverse of player containers?
- In Courtney's Branch – NPC players sometimes spawn in corridors and block your pathway so you can't access part of the map
- Main-Branch Tile Map – Characters sometimes spawn in walls, this was only unsolved because of time restraints.

Does your game contain any non-obvious features?

- Pop-ups work in screen viewport or layer viewport space.
- The unimon cards are created by reading the unimon stats file. You could add your own unimon easily (Make sure to add an image with the unimon's name exactly, into the unimon images folder).
- The AI scales to a larger difficulty. This affects both ingame choices, and the rock paper scissors choices.
- Release button has been modified to suit our needs for screen and viewport.
- The game only allows 3 max school cards into your hand at a time.
- Focus shift is intentional to allow for different point of views depending on whose turn it is, or if an attack is about to happen.
- Different animations and sound occur depending on the attacks.
- Different sounds happen depending on the type of movement.
- Tile map is able to be loaded from file and randomly generated using a seed. The random generator spawns enemies as you walk around.

How does your game map onto the assessment criteria?

PROFESSIONALISM (10): We had a fornightly team meeting, which has mapped accurately into our commits as you can see consistent commits across the whole year (excluding major holiday periods). The game is very stable, with minor bugs that do not crash nor majorly affect the game. The game is well presented, with animations throughout to hold attention. We have made the game scale to screen size so that it can be used on multiple platforms.

QUALITY OF ARCHITECTURAL DESIGN (15): Our code has been structured into appropriate packages. Through pre-planning we have implemented inheritance throughout our classes. We achieved polymorphism throughout many classes through method overloading. We implemented functionality using interfaces for extensibility. Our classes are cohesive, robust and loosely coupled. Our methods tend to be atomic in nature, they have one distinct purpose. We have implemented and extended upon available libraries (GAGE).

USE OF INPUTS, GRAPHICS, AND SOUND (11): We've used consistent graphics throughout to provide a 'QUB' theme. We've used rounded buttons consistently that allow for a 'touch down' appearance. The objects on the mapscreen have a different appearance depending on what way you are walking, and the popups used are consistent with those in the battlescreen. The battlescreen has nice smooth dragging animations, with smooth movements back to the original position if the card has been placed incorrectly. There are animations for each particular type of attack, as well as the AI smoothly moving cards across the screen. We've allowed for different types of touch events, such as long touch and confirmed single taps. For the RockPaperScissors, the cards nicely animate from side to side and zoom in/out. We have appropriate music throughout the game, and sounds for each different type of action you take.

EXTENT OF GAME FEATURES (14): We've added: a splash screen, a new game feature and a load game feature (saves your current level progression), a character creation screen to customize a character, a tutorial screen for before you load into a game, and a helper book for when in game, a candycrush style progression map which brings you into an rpg-style tile map for walking around in. The Battlescreen has a fully functional AI which gets stronger as you progress, very smooth dragging effects with appropriate rules for dragging, a 'rock paper scissors' styled game to decide who gets first turn, a battle end screen which allows you to choose a defeated unimon to add to your deck. There is also an accessory store to spend points you gain from winning games.

COMPLEXITY OF GAME ALGORITHMS (25): We have a dynamic unimon stat input that reads text from a text file. In the rpg style map screen, we have one style of map generation which reads maps from text files and ensures NPCs are generated into valid positions. We have another type of endless map generation which uses Perlin Noise to generate 'endless' maps, which uses a seed to reproduce maps. Differentiation between dragging the layerviewport and objects on the screen (or both simultaneously). Popups that allow feedback from the user. The algorithms behind deciding if a card drag was valid, as well the ability to zoom in on cards. Sounds and animations differing depending on what moves have been used, or if moves are invalid. Cards that rotate on the screen from side to side in rockpaperscissors. An intelligent AI that can be scaled in terms of difficulty, and is able to perform moves along the screen. Multiple classes to allow for different styles of animations. A mergeBitmap() method that allows multiple bitmaps to be combined and positioned relative to each other. Methods that allow for grids to be created out of gameobjects and rects. Methods that allow for the scaling of text/the ability to find text size. Game objects can be scaled to the screen viewport by not passing in a layerviewport. A textparser that allows for the creation of text from bitmaps. Release button extended and modified to be able to work on both screenviewports and layerviewports. Throughout the battleScreen we have state dectectors that drive the game.

CODING STYLE AND CODE QUALITY (25): We have extensive commenting and sectioning in every class to make the classes more readable. We have logged files accurately, and have placed files into appropriate folders. We have used extensible classes through interfaces, enums, and inheritance. We have used polymorphism throughout a range of classes, such as Player and ImageTextHolder. Methods have been moved into accurate classes throughout. Each class is unipurpose and split up accordingly. We used release buttons for all of our buttons throughout the game. We have appropriate and descriptive names for methods, variables and classes. Complex algorithms are broken up into smaller methods to make them easier to manage. We have removed unused code from our classes and we have removed all “TODO” comments and unnecessary comments. Where possible we have tried to keep methods in classes to 25 lines or less.

Section 2

Assessment Weighings

Each project is marked out of 100 for which a total of 50 marks are distributed between three different categories, namely: Use of Input/Graphics/Sound, Extent of Game Features and Complexity of Game Algorithms (with each category having a minimum of 5 marks and a maximum of 25 marks). For your submitted project you can decide how the 50 marks are to be distributed between the three categories. Using this, you can tailor the assessment scheme to your project.

Update the Use of Input/Graphics/Sound, Extent of Game Features and Complexity of Game Algorithms weightings in the table below such that each category has a minimum of 5 and a maximum of 25 marks and the **total marks across all categories sums to 100**.

Assessment Topic	Weighting (total 100 marks)
Professionalism	Fixed 10 marks
Quality of Architectural Design	Fixed 15 marks
Use of Input/Graphics/Sound	11
Extent of Game Features	14
Complexity of Game Algorithms	25
Coding Style and Code Quality	Fixed 25 marks

Section 3 Peer Evaluations

You should discuss the contribution and effort that each member has made towards the project. Following this you should then decide how the marks will be distributed within the project and record the team's decision in the table shown below.

To help you complete this process, the entire team should sit down and complete the following tables. To be fair to everyone in the team, it is of the upmost importance that the tables are honestly and accurately completed. To be blunt, someone who has provided a stronger contribution to the project should receive a greater mark as all the team members have benefitted from their contribution in terms of the final awarded mark.

Use the following scale to complete the table:

1. Less than average
2. A bit less than average
3. Average
4. A bit more than average
5. More than average

Team member 1: Jodie Burnside	Contribution
• Amount of time and effort applied throughout the project	• **[2]**
• Organisational (i.e. planning) and motivational contribution	• **[3]**
• Contribution to solve problems, develop complex code, remove bugs, etc.	• **[2]**
Team member 2: Michael Grundie	Contribution
• Amount of time and effort applied throughout the project	• **[5]**
• Organisational (i.e. planning) and motivational contribution	• **[5]**
• Contribution to solve problems, develop complex code, remove bugs, etc.	• **[5]**
Team member 3: Dariusz Jerzewski	Contribution
• Amount of time and effort applied throughout the project	• **[3]**
• Organisational (i.e. planning) and motivational contribution	• **[4]**
• Contribution to solve problems, develop complex code, remove bugs, etc.	• **[5]**
Team member 4: Chloe McMullan	Contribution
• Amount of time and effort applied throughout the project	• **[5]**
• Organisational (i.e. planning) and motivational contribution	• **[5]**
• Contribution to solve problems, develop complex code, remove bugs, etc.	• **[5]**
Team member 5: Courtney Shek	Contribution
• Amount of time and effort applied throughout the project	• **[4]**
• Organisational (i.e. planning) and motivational contribution	• **[4]**
• Contribution to solve problems, develop complex code, remove bugs, etc.	• **[4]**

The table should provide a rough overview of contribution from each team member (although I stress it is a rough measurement).

Important: the following table is not intended to be judgemental – for example someone might not have been able to contribute much time and effort to the project due to other commitments. Whilst this explains and justifies the situation, it remains important that the imbalance is still reported.

Section 4

Code Contribution

On the next page the team must complete the table by listings all the .java files contained within your project (including the size of the .java file in Lines of Code (LOC)). Against each source file, the team must then agree the contribution of each team member to the ***design and development*** of that piece of source code. If needed, extend the table across multiple pages for large projects.

A fictitious example is provided below:

Source File	LOC	J.Blogs	S.Smith	P.Laverty	Z.Zhou
Game.java	258	40%	10%	20%	30%
Player.java	105	0%	40%	0%	60%
Level.java	145	60%	40%	0%	0%
Collectable.java	56	0%	0%	0%	100%
Splash.java	25	0%	100%	0%	0%

Important: Any code that you want me to assess must be listed below. This includes code from different projects and/or different branches that you want me to assess.

Important: Copy and paste the table for each different project/branch, although, source code in a branch that does not differ from the master branch does not need to be listed in the branch table.

Important: To assist with the above process, please do ensure that you comment your code (class headers, method headers) so that the author is specified. This is particularly important for a source file that more than one team member has contributed to.

Important: Not being able to come to a clear allocation is a valid outcome. If this is the case, then either:

- [Recommended] Make sure the source code explicitly indicates the author of each method and I will evaluate myself.
- Talk to me and we can arrange a team meeting. As part of the arbitration process I will meet with the entire team and go through the contested source files to identify the sections that each team member has developed. I will then decide on how the contested code will be separately marked.

Project/Branch	csc2043-1617-56 : Master Branch					
Source File	LOC	[Michael]	[Chloe]	[Courtney]	[Dariusz]	[Jodie]
Animation/AnimationFactory	24				100	
Animation/Animator	42				100	
Animation/FlipBook	26				100	
Animation/GameObjectFlipBook	77				100	
Animation/iAnimation	15				100	
Graphics/Animation	223		100			
Graphics/CanvasGraphics2d	190		25	25	50	
Graphics/IGraphics2d	117		25	25	50	
Graphics/ImageHolder	210		100			
Graphics/ImageTextHolder	37		100			
Graphics/TextHolder	260		100			
Graphics/TextParser	280				100	
Input/Input	208	100				
Input/TouchEvent	39		100			
Input/TouchEvenHandler	346		100			
IO/FileIO	264	100				
Engine/AssetStore	253	100				
AI/AI	478		100			
AI/AILogic	587		100			
AI/AIMove	105		100			
AI/AIPerformMoves	416		100			
CardClasses/Card	168	25	25	25	25	
CardClasses/School	10		100			
CardClasses/SchoolCard	50	100				
CardClasses/Specials	57					100
CardClasses/Unimon	210	45	45	10		
CardClasses/UnimonCard	601	50		10	40	
CardClasses/UnimonMove	367	50		50		
CardContainingStructures/ CardContainer	196	5	15		80	
CardContainingStructures/ CardHolder	63				100	
CardContainingStructures/ CardHolderContainer	138		5		95	
Shop.StoreItems/StoreItem	27					100
Shop.StoreItems/Item	21					100
Shop.StoreItems/ItemTypeENUM	13					100
TileMapElements/Camera	21				100	
TileMapElements/Destructible	43				100	
TileMapElements/Direction	12				100	
TileMapElements/Enemy	32			75	25	
TileMapElements/Entity	57			75	25	
TileMapElements/EntityType	8			75	25	
TileMapElements/Floor	47				100	
TileMapElements/IDestructibleTile	11				100	
TileMapElements/IEntity	17				100	
TileMapElements/IModifyingTile	10				100	
TileMapElements/IMovable	10				100	
TileMapElements/ITile	9				100	
TileMapElements/Movement	71			75	25	
TileMapElements/NPC	61			50	50	

TileMapElements/PerlinMap	210				100	
TileMapElements/Tile	27				100	
TileMapElements/TiledMovement	16				100	
TileMapElements/TileType	12			50	50	
TileMapElements/Wall	23				100	
GameElements/Board	331	45	10		45	
GameElements/CardMovement	577	95	5			
GameElements/DeckBuilder	169	100				
GameElements/GameState	75				100	
GameElements/PlayArea	6	100				
GameElements/Player	271	40	5	5	45	5
GameElements/PopUp	312	100				
GameElements/RockPaperScissors	564		100			
GameElements/TypeOfLoss	24	100				
GameElements/User	71					100
GameElements/WitPointsBar	172	50		50		
GameElements/AssetLoading	331			5		95
GameScreens/AccessoriesShop	168					100
GameScreens/BattleEndScreen	204					100
GameScreens/BattleLoadingScreen	57					100
GameScreens/BattleScreen	1241	73	15	5	5	2
GameScreens/CharacterCreation	148					100
GameScreens/EndlessScreen	400			10	90	
GameScreens/HelperBook	213					100
GameScreens/LanyonLevel0	99					100
GameScreens/MapScreen	256	10	15	10		65
GameScreens/MenuScreen	212			50		50
GameScreens/PauseScreen	397			100		
GameScreens/SplashScreen	109				100	
GameSpecificAnimations/ moveUpOrDownGameObjectAnimation	172	100				
GameSpecificAnimations/ PopOutFromCentreOfGameObjectAnimation	99	100				
UI/LayerViewportReleaseButton	95	100				
UI/ReleaseButton	253	100				
UI/ScreenViewportReleaseButton	63	100				
Util/CyclicQueue	114				100	
Util/GameObjectMovement	56		100			
Util/GraphicsHelper	349		80		20	
Util/LayoutHelper	136	20	20		60	
Util/MathHelper	44				100	
Util/PerlinNoise	114				100	
Util/RNG	84				100	
Util/Vector2	204				100	
World/GameObject	256	100				
World/GameScreen	248			100		
Assets/UnimonStats.txt	26	100				
InstrumentedTest/DeckBuilderTest	111	100				
JUnitTest/AIMoveTest	61		100			

JUnitTest/AITest	435		100			
JUnitTest/CardContainerTest	157			100		
JUnitTest/CardTest	43			100		
JUnitTest/PlayerTest	98					100

Project/Branch	csc2043-1617-56 : Courtney Branch					
Source File	LOC	Courtney	Michael	Chloe	Jodie	Dariusz
TileMapElements/NPC	180	100				
TileMapElements/NPCType	13	100				
TileMapElements/TileMap	182	100				
TileMapElements/TileMapPlayer	267	100				
GameScreens/TileMapScreen	391	100				

Section 5 Overall Evaluation

With reference to the above two completed sections (peer evaluation and individual code contribution), the team should discuss and try to agree an overall contribution measure for each team member (as covered below).

A percentage figure must be entered against each team member's name. A value of 100 entails that the person contributed ***as expected by the team*** to the project. A value of more than 100 entails that the team felt the individual's contribution is deserved of additional recognition (a value of 110-120 represents a notable additional contribution; a value of 120-125 represents a very significant additional contribution). A value of less than 100 entails that the team felt the individual did not contribute to the team as expected.

150 –	Outstanding contribution
125 –	Very strong contribution
110 –	More than expected
105 –	A bit more than expected
100 –	As expected
95 –	A bit less than expected
90 –	Less than expected
75 –	Weak contribution
50 –	Very weak contribution
0 -	No contribution

Note: In order to calculate each individual score (S_i) from the team score (T) and the peer assessment scores (P_1,...P_N) where N is the number of team members, the following formula will be used:

$$S_i = \frac{P_i}{\frac{1}{N}\sum_{j=N}^{j=1} P_j} T$$

For example, for an individual with a peer score of 110 (with other peer scores of 100, 100) and a team score of 68% will have a final score of = 110 / (1/3 * (110+100+100)) * 68% = 73%

Team member	Peer weighting
Michael Grundie	150
Chloe McMullan	150
Jodie Burnside	150
Courtney Shek	150
Dariusz Jerzewski	150

Utterly Important: I feel strongly that the peer assessment should be fair. Those who have contributed more to the project should receive a higher project mark. If the peer weightings largely agree with my own assessment of individual code contribution, then I will use the peer weightings to assign the final marks. However, if I feel the peer weightings do not reflect my assessment of individual contribution then I will go with my own assessment based on the individual code contributions.

Testing Bonus Challenge

Using the same scale as on the pervious page, provide weighting for those team members who attempted the testing bonus challenge. If the team did not attempt the bonus challenge, then please leave this section blank.

Team member	Peer weighting
Michael Grundie	150
Jodie Burnside	150
Chloe McMullan	150
Courtney Shek	150
Dariusz Jerzewski	150

Submission Information

You must submit the following and must adhere to the instructions for submission:

- A completed version of this document – including mark distribution, peer distribution, project information and, most importantly, the completed declaration of academic integrity. If the declaration is not signed then your project **will not** be marked.

- Your project to include all source files used by your project along with all images, sounds, and other files needed for your project to run. It is your responsibility to ensure that all content needed to run your project is included.

You may submit the following (i.e. it is not compulsory that you submit the following):

- Any documentation (e.g. design documents, storyline, artwork development, etc.) and/or additional code (e.g. partial code fragments, non-working code, etc.) that didn't make it into the final game.

Note that materials submitted after the hand-in date will not normally be considered unless prior permission has been granted for late submission.

How the project must be submitted:

You must:

- Ensure the latest version of the project is available on Gitlab and email any documentation, including this form, to P.Hanna@qub.ac.uk by 12:00 on the 8th May deadline.

- [Optional] You may also wish to send me your project using a service such as QUB's Dropbox (https://dropbox.qub.ac.uk).

Declaration of Academic Integrity

To electronically complete the declaration shown below, enter your name, today's date and within the 'Declaration' box type 'I agree to the terms of the declaration'. For example, if I were to complete the declaration I would enter:

Name Date Declaration
Philip Hanna 07/05/17 I agree to the terms of the declaration

As this is a team project, every member of the team must complete the declaration shown below:

Before completing the declaration below, check that the submission:

1. contains full acknowledgement of all secondary sources used (paper-based and electronic)
2. all code is original unless clearly referenced as otherwise

I declare that I have read the Queen's University regulations on plagiarism, and that the attached submission is original work unless clearly referenced as otherwise. I understand that my submission will be subject to an electronic test for plagiarism and will also be subject to the University regulations concerning late submission if received after the deadline.

If the team has submitted a set of agreed peer contribution weightings: I declare that the peer review process was conducted in accordance with the guidelines and the specified weightings provide an accurate assessment of individual contribution.

Name	Date	Declaration
Michael Grundie	07/05/17	*I agree to the terms of the declaration*
Chloe McMullan	07/05/17	*I agree to the terms of the declaration*
Courtney Shek	07/05/17	*I agree to the terms of the declaration*
Dariusz Jerzewski	07/05/17	*I agree to the terms of the declaration*
Jodie Burnside	07/05/17	*I agree to the terms of the declaration*