College of Engineering, Construction and Living Sciences
Bachelor of Information Technology
ID737001: Game Development
Level 7, Credits 15
Project: Game Development + Demo

Assessment Overview

In this assessment, you will design and develop a game using a programming language/game engine of your choice. In addition, marks will be allocated for code quality and best practices, documentation and Git usage.

Learning Outcome

At the successful completion of this course, learners will be able to:

1. Design and develop a game using industry standard tools, technologies and practices.

Assessments

Assessment	Weighting	Due Date	Learning Outcome
Assignment	30%	07-06-2024 (Friday at 4.59 PM)	1
Project: Game Development + Demo	70%	21-06-2024 (Friday at 4.59 PM)	1

Conditions of Assessment

You will complete this assessment during your learner-managed time. However, there will be time during class to discuss the requirements and your progress on this assessment. This assessment will need to be completed by **Friday, 21 June 2024** at **4.59 PM**.

Pass Criteria

This assessment is criterion-referenced (CRA) with a cumulative pass mark of **50%** over all assessments in **ID737001: Game Development**.

Authenticity

All parts of your submitted assessment **must** be completely your work. Do your best to complete this assessment without using an **AI generative tool**. You need to demonstrate to the course lecturer that you can meet the learning outcome(s) for this assessment.

However, if you get stuck, you can use an **AI generative tool** to help you get unstuck, permitting you to acknowledge that you have used it. In the assessment's repository **README.md** file, please include what prompt(s) you provided to the **AI generative tool** and how you used the response(s) to help you with your work. It also applies to code snippets retrieved from **StackOverflow** and **GitHub**.

Failure to do this may result in a mark of **zero** for this assessment.

Policy on Submissions, Extensions, Resubmissions and Resits

The school's process concerning submissions, extensions, resubmissions and resits complies with **Otago Polytechnic** policies. Learners can view policies on the **Otago Polytechnic** website located at https://www.op.ac.nz/about-us/governance-and-management/policies.

Submission

You **must** submit all application files via **GitHub Classroom**. Here is the URL to the repository you will use for your submission – https://classroom.github.com/a/dbrTesMQ. If you do not have not one, create a **.gitignore**. The latest application files in the **main** branch will be used to mark against the **Technical and Professional Proficiency** criterion. Please test before you submit. Partial marks **will not** be given for incomplete functionality. Late submissions will incur a **10% penalty per day**, rolling over at **5:00 PM**.

Extensions

Familiarise yourself with the assessment due date. Extensions will **only** be granted if you are unable to complete the assessment by the due date because of **unforeseen circumstances outside your control**. The length of the extension granted will depend on the circumstances and **must** be negotiated with the course lecturer before the assessment due date. A medical certificate or support letter may be needed. Extensions will not be granted for poor time management or pressure of other assessments.

Resits

Resits and reassessments **are not** applicable in **ID737001: Game Development**.

Instructions

Technical and Professional Proficiency - Learning Outcome 1 (50%)

- The topic for the game is **your choice**.
- The game needs to open without code or file structure modification in an appropriate **text editor**.
- Gather requirements from the client and deconstruct them into user stories.
- Design and develop a game using a chosen **programming language/game engine**.
- Demo the game on **itch.io**.

Code Quality and Best Practices - Learning Outcome 1 (30%)

- An appropriate **.gitignore** file is used.
- Appropriate naming of files, variables, methods and classes.
- Idiomatic use of values, control flow, data structures and in-built functions.
- Efficient algorithmic approach.
- Sufficient modularity.
- Each file has an **comment** located at the top of the file.
- Formatted code.
- No dead or unused code.

Documentation and Git Usage - Learning Outcome 1 (20%)

- **GitHub** project board or issues to help you organise and prioritise your development work. The course lecturer needs to see consistent use of **GitHub** project board or issues for the duration of the assessment.
- In a **Microsoft Word** document called **game-document**, explain the following:
  - Core concept
  - Design pillars
  - Main features and mechanics
  - Target platform and audience
  - Interface and controls
  - Basic story
  - Visual style
  - Audio style
  - Known issues and bugs
  - Future improvements
  - A URL to the game on **itch.io**.
- In a **Microsoft Word** document called **play-testing**, record the following:
  - Overall experience:
    - Rate your overall experience playing the game from 0-5 (0 being the worst and 5 being the best).
    - A brief explanation of your experience. Highlight one thing you enjoyed and one thing you did not enjoy.
  - Game mechanics:
    - Identify any game mechanics that felt intuitive or unintuitive.
    - Improvements to enhance the game mechanics.
  - Controls:
    - Were the controls easy to learn and use?
    - Did you encounter any issues with the controls?
  - User interface:
    - Was important information presented clearly?
    - Did the user interface enhance or detract from the game?
  - Difficulty:

        - Was the game too easy, too hard or just right?
        - A brief explanation of challenges that felt challenging or unfair.
    - Bugs:
        - Document any bugs you encountered during play testing.

- Select two interesting game mechanics that you implemented and explain in a **Microsoft Word** document called **reflections** the following:
    - What did you implement?
    - What did you research during the implementation? Provide a link to the resources you used.
    - What did you try? What worked? What did not work?
    - What did you learn?
    - How can you apply what you learned to future games?
    - During the development of your game, what did you find most challenging professionally? How did you overcome it?
- Correct spelling and grammar.
- Your **Git commit messages** should:
    - Reflect the context of each functional requirement change.
    - Be formatted using an appropriate naming convention style.

Additional Information

- **Do not** rewrite your **Git** history. It is important that the course lecturer can see how you worked on your assessment over time.
- You need to show the course lecturer the initial **GitHub** project board or issues before you start your development work. Following this, you need to show the course lecturer your **GitHub** project board or issues at the end of each week.