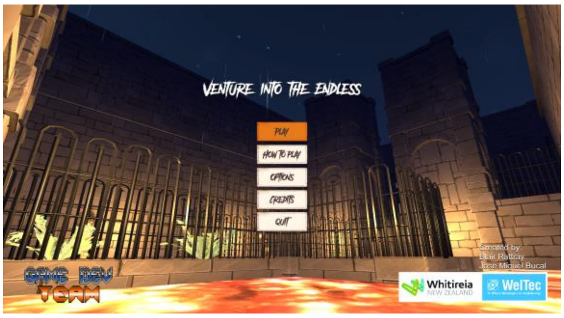
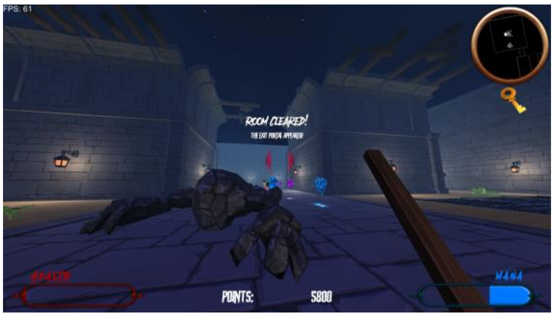
3D Game – Venture into the Endless

By: Blair Rattray, Jose Miguel Bucal **Advisor:** Reza Moosa **Client:** Self-client



Venture into the Endless Main Menu



Boss battle aftermath

INTRODUCTION

Venture into the Endless is a 3D first-person rogue-like shooter game. The player will be spawned in a procedurally generated level, meaning the rooms are pre-made but laid out in a random order. This game was developed using the Unity engine.

Our intention was to build a one player game which will be very visual and designed to stop people walking past expo stands to capture their attention. The game itself will be student branded and Weltec branded to both further raise the profile of student achievement and the institute.

DEVELOPMENT

We developed this project using Unity, a cross platform game engine with a built-in IDE developed by Unity Technologies. It is a real-time game development platform that uses C# scripts, which are written using Visual Studio.

The development was accomplished using the SCRUM methodology, where the team works in two-week sprints to achieve their goal. The product and sprint backlogs were set up in Google Drive, and Unity Collaboration services were used for version control. All assets used within the game were either provided by our (former) Client or freely downloaded from the Unity asset store. Also, most sounds used came with the assets provided by the (former) Client. The main menu music was created by Eric Matyas using a Creative Commons license.

While the procedural room generation was being developed, the other members of the team were working on the player class, including their weapon, movement, stats, and inventory. Once the player was completely developed the team then moved into implementing the enemy characters and their interaction with the player.

The game's features and functionality were tested using debugging and unit tests with Unity Test Runner. User Testing surveys were also sent out to testers to give the team feedback about improvements and recommendations regarding future development. Integration testing was done manually in test cases for Player input to check the function is correctly communicating with other modules inside the game. Compatibility testing was done to check for game compatibility on software, hardware, devices etc.

Once the main core features and functionality was fully developed and tested, we moved into adding event modes for rooms for the player to interact with. After the final feature was implemented, we worked on game optimizations to ensure that the product worked as intended before deploying as an executable.

CONCLUSION

Our initial project plan proved to be quite accurate in the outcome of our project. We did have some issues with managing our (former) client. Due to the difficulties that arose, we had to let them go and undergo the project without a client.

With the project planned out, we completed our sprints within the designated time frame. There were a few tasks that were not completed in a specific sprint, these were quickly sorted in subsequent sprints.

Testing took place to highlight any errors or helpful missing features. We got a lot of feedback from testers and implemented quite a few extra features that were suggested in our final development/touch-ups sprint.