# Competitors

# Oceanhorn: Monster of the Uncharted Seas / Oceanhorn 2: Knights of the lost Realm

* PEGI 7
* £10.99 on steam

## Moonlighter

* PEGI 7
* Approx. £15 on steam

# Hyper Light Drifter

* PEGI 12
* Approx. £9 on steam

Hyper light drifter is an indie RPG which involves a significant amount of action and adventure. The game was developed by Abylight Studios. Their key staff has 8 members, meaning this is a viable competitor for us due to a relatively small team size.

Hyper light drifter has an atmosphere which is unique in comparison to other games on the market. We strive to create an atmosphere which is not like the one seen in Hyper Light Drifter, but one which holds a similar amount of uniqueness. It is a suitable competitor as we will be competing for audiences which are of the artistic type and play games mainly due to the art style/graphics.

The story of Hyper Light Drifter follows the story of a protagonist who is venturing out to find a cure for his own illness, similarly to the concept of The Cure. Whilst travelling through “The Lands of Time”, the character is faced with many obstacles (enemies) which he must fight with his sword; in our game the player will be required to face combat during random parts of the game with either Malum’s men or other enemies.

# Prototypes & Playtesting

We intend on making a small prototype for the game that we can playtest to test features and gain feedback from our target audience. This prototype will be made immediately after all documentation and concept details are complete. This will not be a digital prototype but will be a prototype that is paper based (similar to a standard board game). This prototype will mainly serve to test some of the game’s core mechanics, features and general concept. Feedback from this prototype’s playtesting sessions will be taken in the form of anonymous feedback forms, to ensure all play testers are 100% honest and to gain the most reliable feedback possible. The feedback will help the team to make any last-minute concept changes to the game before digital development can commence.

After these changes are made, a second prototype will begin to be developed. This prototype will be a digital prototype and will further steer the team in the correct direction for the remaining period of game development. The second prototype will consist of one core level of the game, and this will be the tutorial level. This is a suitable level to prototype because it will allow the play testers to come to grips with the game’s mechanics and controls in the short time frame they have access to the prototype (1 – 2 hours maximum). This level will have the essential features implemented, however smaller, more unnecessary features (such as dialogue from non-important characters, certain pick up items which don’t cause progression within the prototyped level, non-interactable objects etc) will not be required for this early prototype. It will be essentially be one full section from the minimum viable product.

Other than the playtests which will occur after the completion of the initial prototype, there will also be a third playtest when the game is mostly, or fully complete. This playtest will allow play testers to identify any faults or bugs within the game and this will be presented to the team via feedback forms. In relation to the release of the game, the final playtest should be set to occur a couple of weeks before intended release date.

The playtests mentioned are the essential playtests which without a doubt need to occur to progress with the development of the game. However, to ensure that our game is of an acceptable standard and quality, we will have many other smaller playtests throughout the entire development process.

The playtesting team will be comprised of the development team themselves, and any other volunteers who are interested in playing the game. The playtest will be advertised to mostly students; however, other groups of people are also desired.

# Technical Documentation

**Primary Algorithms:**

* **A\* Search Algorithm:** This algorithm will be helpful whenever generating the game’s map, more specifically to find the most suitable, shortest path between two locations on the map to create a waypoint system. This feature will benefit the player as it will help them to navigate an open world environment at ease.
* **Sorting Algorithm**: This algorithm could be useful for things such as the shop that player can buy items from and could sort the items into the correct alphabetical order, high to low (price) and vice versa etc, to enhance the player’s experience.
* **[ARTIFICIAL INTELLIGENCE] Algorithms for Steering Behaviours:** Behaviours such as fleeing, following and arriving at waypoints will be useful for certain AI in our game to both make NPCs seem more human-like and to also contribute to the game world (i.e. Witch Doctor fleeing from player until he arrives at certain point on map and comes to a halt). This can be programmed by using basic object transformations and a set of switch/if else statements.
* **Finite State Machines:** A finite state machine is required for AI to enter different states (i.e. *Enemies:* Attacking the player, Idle, Wandering etc). This can be via general behaviours or the different states of animations of the player character and NPCs.

Player is close enough for the NPC to attack and take health from player

Player enters the NPC’s line of sight at a relatively close distance

Player approaching the area of NPC

Attack Player

Idle

Wandering

Chase Player

Player is not close enough for the NPC to attack and take health from player

Player leaves the NPC’s line of sight at a relatively far distance

Player leaves the area of NPC

An example of a finite state machine for a random NPC which spawns whilst the player is exploring the land (i.e. one of Malum’s men)

**Game Cheats for Testing:**

*Cheats should be used by the testers by entering a secret cheat code which will be supplied to them by the programmer.*

* Cheat which allows the user to skip to different levels in the game.
* Cheat for teleporting throughout the game’s map on the input of x and y values
* Cheat to skip to time of day by inputting the time in the 24-hour time format (i.e. 13:00 as opposed to 1:00pm)
* Cheat for increasing/decreasing the money player has
* Cheat for increasing/decreasing player reputation
* Cheat to increase/decrease health state
* Cheat to have unlimited money/health
* Cheat to add items to inventory
* Cheat to navigate the map via flying

\* Some of these cheats may also be used by the player in “New Game +” to enhance user experience and the reward for completing the game.

**Hardware Requirements:**

* PC equipped with mouse and keyboard
* Minimum system requirement of:

OS: Windows 10 / MacOS Catalina

Processor: Intel Core i5 / Ryzen 5

Memory: 8 GB RAM

Graphics: GeForce GTX 660 / Radeon R9 270

Storage: 4 GB available space

Sound Card: DirectX compatible

**Game Engine:**

The game engine the team have agreed to use is ***Unity.*** This is because all team members have had prior experience with this game engine. It is also widely used and therefore a vast amount of information and solutions can be easily found on perhaps, online forums, if we were to encounter any critical errors during the development process. Unity is also considered rather easy to use in comparison to many other game engines, and therefore will be a lot easier for our small team to handle and navigate.

Overall, the main reason why we have chosen to use Unity as the game engine to develop The Cure on is due to the fact of it being free to use (as long as a significant profit is not made), and will suit the restrictions we have budget-wise with the project.

**Good Will:**

A method will be required to set and store the percentage of good will the player has. Good will refers to the reputation that the player has with the NPCs in the game. The player can gain good will by doing tasks for NPCs or interacting with them in the correct way. This will essentially be the main point system within the game and will determine the quality of the ending the player receives; it works as follows:

Criteria for the percentage of good will required for each ending:

Possible good will which can be gained by the player at certain points in game:

**Ending 1 (“Bad” Ending):** <=59%

**Ending 2 (“Normal” Ending):** <=94%

**Ending 3 (“Golden” Ending):** >=95%

**Tutorial Level**: 12%

**Witch Doctor:** 13%

**City of Haven:** 40%

**Healer Level:** 18%

**Ingredients Level:** 18% / 45% **(dependant on success of previous level**)

This method will be comprised of constantly checking if the Good Will is increased/decreased and running through perhaps an if statement to compare the current good will percentage to the criteria whenever an ending is required to be determined.

# MVP Systems & Features

**Multiple Endings**

If the product were to be developed to a bare minimum the team have agreed that it would still be suitable to keep the multiple endings feature. This is due to it being one of our major selling points and we would not like to reduce the amount of current selling points.

However, if we were to produce the project to a minimum, the game would consist of 2 core endings as opposed to 3. This is because it would be a lot more time consuming to produce less endings, more computationally efficient and more cost efficient. It would be more cost efficient due to less voice lines needing to be produced, which would allow the team to pay the voice actors less. It would also require less production of cutscenes and other relevant and costly assets.

The endings which we intend on producing in the minimum viable product is Ending 1 (The “Bad” Ending) and Ending 3 (The “Golden” Ending). This is because we believe that these endings are the most diverse in comparison to one another. Also, allowing the player to receive the most dissatisfying ending will give them the most incentive to replay the game to receive the most satisfying ending.

**New Game +**

An option for New Game + is also necessary to produce the minimum viable product. This is a necessity as it allows the user to replay the game with perhaps certain items/upgrades they have collected, an animal companion (horse or a wolf), the option to use cheats and extra good will points. This will further give the player an incentive to replay the game to achieve a better ending if they were dissatisfied in their first playthrough and in turn, this will increase their opinion of the game as a whole- leading to The Cure to be promoted via word of mouth.

**Good Will**

To determine the multiple endings, we will need to keep the concept of good will in the minimum viable product. Also, the algorithms and methods which are associated with the good will system should be simple to implement. However, the tasks that need to be completed to achieve a high percentage good will can be lessened.

Such tasks include tasks that do not have a direct impact on the general story, such as delivering presents to in-laws or helping the shepherd find his sheep. (i.e. “Short term goals” listed on the Player Objectives/Progression page.

**Technical Specifications**

* Minimum system requirement of:

OS: Windows 10

Processor: Intel Core i5 / Ryzen 5

Memory: 8 GB RAM

Graphics: GeForce GTX 660 / Radeon R9 270

Storage: 4 GB available space

Sound Card: DirectX compatible

If the user is intending on playing on a PC which does not meet or exceed these specifications, they should not run it. We expect most standard laptops that are owned by casual PC gamers (who are of the student demographic) to meet/exceed these specifications.

At minimum, we would like to release The Cure on Windows operating systems and exclude Mac operating systems. This is because the majority of the development team own Windows computers and programming/testing/releasing in two different operating systems could be time consuming. Also, we believe that most PC gamers own computers with the Windows operating system, so we would be catering to the majority.

Users that do not understand these specifications will be advised to check this using an online tool such as <https://www.systemrequirementslab.com/cyri> .

# Localisation

We intend on making the game purely English due to our small budget and team who do not have any speaking knowledge on other languages. Therefore, any means of translation will not be required.

However, if the game does prove significant success in the future during the post release period, other language options will slowly be introduced. We will accomplish this by hiring translators.