|  |  |
| --- | --- |
|  | **Escola Secundária Damião de Goes** |
| The Horde  **Relatório no âmbito da Prova de Aptidão Profissional**    **Curso Programador de Informática**    **Nome:** Nicolae Malai  **N**º: 17 **Turma**: PI  **Professor/a Orientador/a**: Sandra Cristina Paiva Resende | |

**Agradecimentos**

Primeiramente quero agradecer à escola por ter aberto este curso, o que me deu a oportunidade de estudar em uma área que gosto e tenho muito interesse.

Em segundo lugar quero agradecer á minha professora orientadora, Sandra Resende, por me ajudar e incentivar a trabalhar em algo que gostasse mesmo sendo diferente do que se realiza habitualmente para esta prova. Também quero agradecer á professora e coordenadora deste curso, Clara Silva e ao professor Álvaro Guedes pela ajuda oferecida ao longo destes três anos, em todos os aspetos necessários.

Por fim quero agradecer à minha família e a todos os meus amigos por me ajudarem a experimentar o projeto diversas vezes, pelas ideias proporcionadas e pela motivação para não desistir nos momentos mais difíceis.

A todos, o meu mais sincero obrigado!

**Índice**

|  |  |  |
| --- | --- | --- |
| **Capa** | ............................................................................................................ | **1** |
| **Agradecimentos** | ............................................................................................................ | **2** |
| **Índice** | ............................................................................................................ | **3** |
| **Dicionário** | ............................................................................................................ | **4** |
| **Introdução** | ............................................................................................................ | **5** |
| **Descrição do projeto** | ............................................................................................................ | **6-10** |
| **Desenvolvimento do projeto** | ............................................................................................................ | **11-29** |
| **Recursos Utilizados** | ............................................................................................................ | **30** |
| **Conclusão**  **Bibliografia** | ............................................................................................................  ............................................................................................................ | **31**  **32** |

**Dicionário**

**Unity** – Software criado pela empresa **Unity** para desenvolvimento de jogos.

**Unity Editor** – Ambiente gráfico dentro do **Unity**.

**Digital Designer** – Profissão que consiste em criar arte para o mundo digital.

**Sound Designer** – Profissão que consiste em criar sons para diversas necessidades.

**Script** – Código atribuído a algum *GameObject* para realizar uma certa tarefa.

**GameObject** – Objeto criado dentro do *Unity Editor*.

**Dash** – Ato de correr certa distância a grande velocidade.

**Death Screen** – Janela mostrada ao jogador após este perder o jogo.

**Leaderboard** – Tabela de classificações.

**Trello** – Website usado para a organização de projetos.

**Sprite** **Renderer** – Componente no **Unity** que permite dar uma aparência ao *GameObject*.

**Rigidbody** – Componente no **Unity** que permite ao *GameObject* interagir com as físicas do **Unity**.

**Collider** – Componente no **Unity** que permite ao *GameObject* colidir com outros *GameObjects*.

**Função** **MovePosition()** – Função pertencente ao *Rigidbody* que permite adicionar movimento a um *GameObject* que tenha o componente *Rigidbody*.

**Raycast** – No **Unity**, um raio enviado para um espaço 2D ou 3D que se move em uma certa direção.

**AI** – Inteligência artificial.

**Asset** – Objeto exterior ao projeto atual(som, imagens Etc.).

**UI** – Interface do utilizador.

**Canvas** – No **Unity**, área onde todos os elementos do *UI* estão.

**Start()** – Função chamada sempre que o script iniciar pela primeira vez.

**Vignette** – Vinheta, neste caso no sentido fotográfico.

**INTRODUÇÃO**

O meu projeto de aptidão profissional consiste na criação de um jogo 2D para computador, junto com um website relacionado ao mesmo.

O objetivo do jogo é criar um loop infinito e testar o jogador quanto às suas capacidades de sobreviver a uma quantidade enorme de inimigos.

Quanto ao website, o seu propósito é servir de entrada para o jogo, ou seja, poderá realizar o download por lá, assim como um conjunto de opções quanto à sua conta e aos seus recordes.

A prova visa aprofundar e desenvolver os meus conhecimentos e capacidades nestas duas àreas da programação, assim como criar algo jogável, divertido e apreciado por várias pessoas de todas as idades.

**DESCRIÇÃO DO PROJETO**

A criação de um website era uma escolha certa, porém, necessitava de algo mais complexo e ao que o website poderia dar uso. Visto o meu interesse na área de desenvolvimento de jogos, ocorreu-me juntar um ao meu projeto final.

O jogo tem um estilo **2D** ***Singleplayer*** ***Survival****.* Isto significa que o jogador terá de sobreviver o máximo que conseguir tendo uma pontuação no final para provar tal.

Necessitava de algo que cativasse o jogador não pela complexidade do jogo mas sim pela sua jogabilidade interessante e mecânicas diferentes do comum.

O objetivo seria nem o website nem o jogo fazerem sentido sozinhos, ou seja, cada um precisaria do outro para funcionarem como o pretendido e creio ter conseguido realizar tal tarefa.

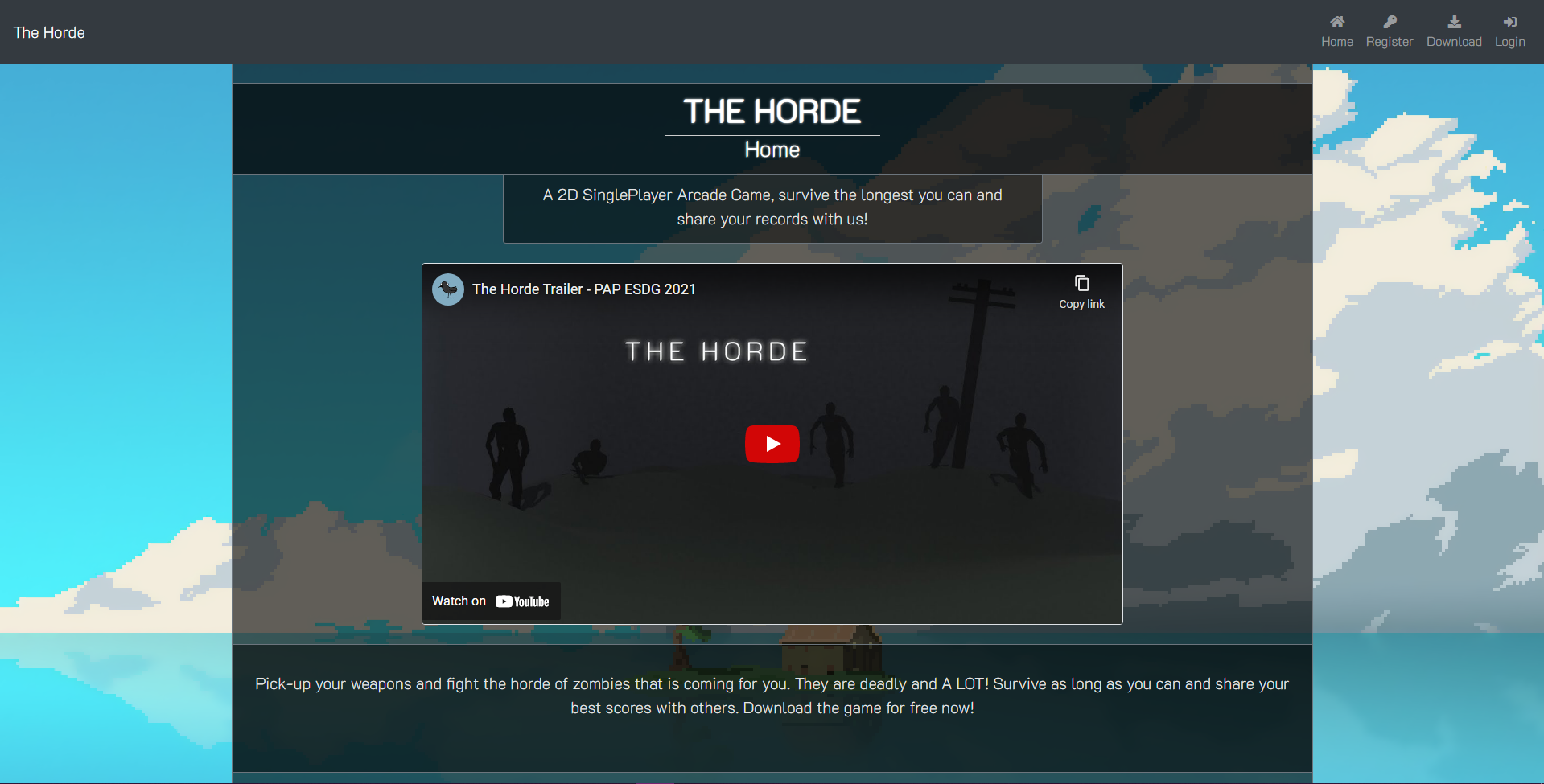


Figura - Página inicial do website

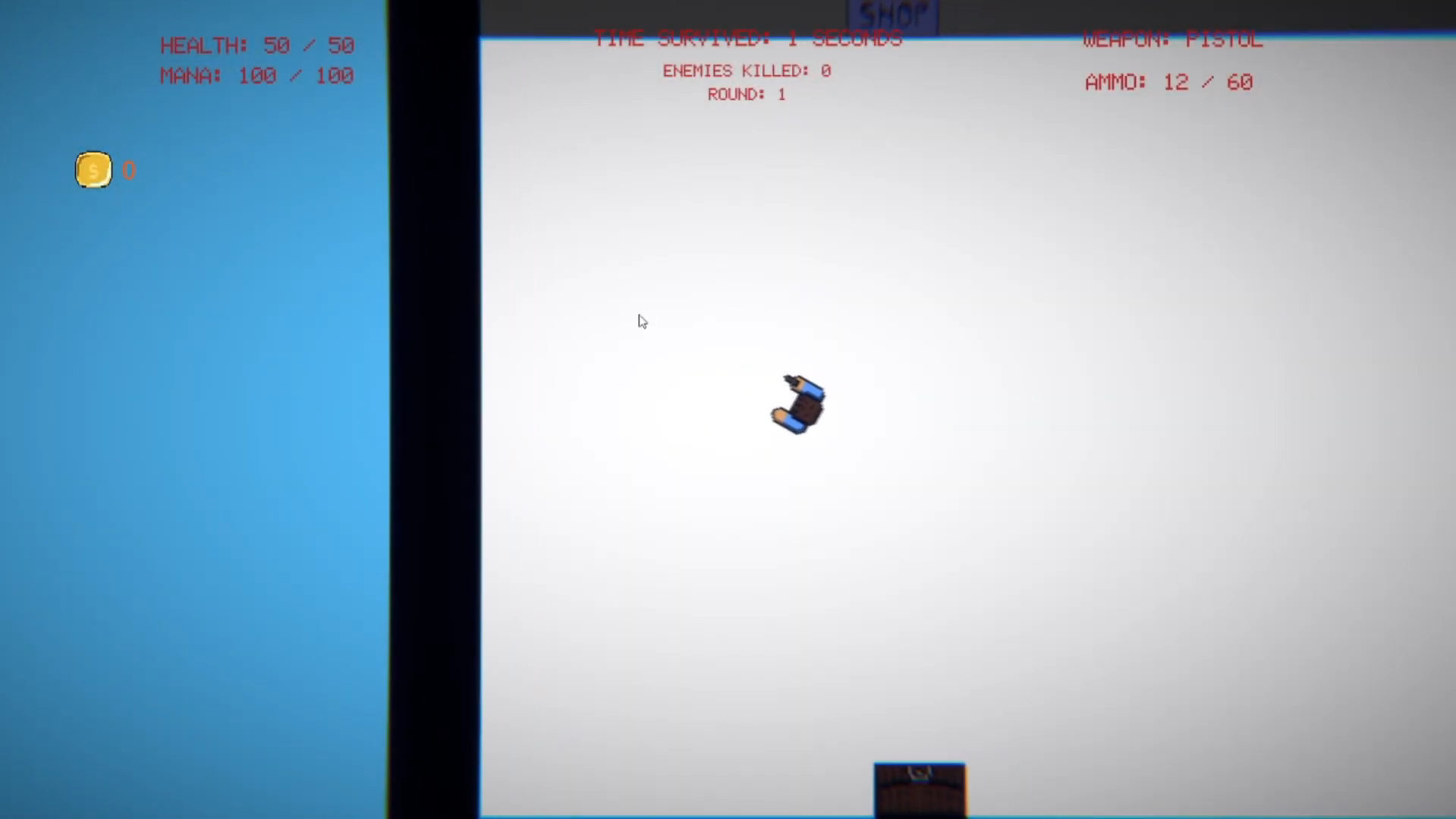


Figura 2 - Imagem do jogo

Após vários dias de conversas com amigos e reflexão sobre o tema do projeto, surgiram várias

ideias, no entanto percebi que a minha motivação e vontade era realizar um jogo 2D do estilo ***Massive Multiplayer Online Role-Playing Game***(MMORPG). Então iniciei a jornada por estudar tudo o que conseguia encontrar sobre este estilo de jogo.

Como breve explicação, este jogo é normalmente feito em um mundo aberto gigante, com imensos detalhes, missões, objetos com quais o jogador pode interagir, mecânicas sem conta, uma grande história e por aí adiante.

Não tardou muito para perceber que era impossível para mim, sem experiência em desenvolvimento de jogos e mesmo no **Unity**, desenvolver um jogo deste gênero com o tempo dado. Mesmo com experiência, o problema seria desenvolver algo cativante e divertido para o jogador em tão pouco tempo.

Para dar um exemplo mais concreto do trabalho que é fazer um jogo deste gênero podemos ter em conta que, em média, estes são desenvolvidos por equipas inteiras de profissionais, sendo eles programadores, *digital* *designers*, *sound* *designers* e/ou historiadores, demorando até anos para conseguir o produto final. Tudo isto tendo em conta que têm pessoas para investir tempo e imenso dinheiro no projeto, possivelmente alcançando 7 dígitos.



Figura 3 - Jogo "World of Warcraft"



Figura 4 - Jogo "Guild of Wars 2"

Logo após perceber que a minha ideia inicial não era fazível em termos de tempo decidi criar algo

mais simples. Passei o resto do dia a pesquisar sobre estilos de jogos que seriam relativamente mais fáceis e possíveis de desenvolver em tempo útil tendo em conta os prazos de entrega previamente estabelecidos.

Não encontrando muitas ideias nas quais tivesse muito interesse, decidi criar um projeto no **Unity** e começar por desenvolver a mecânica mais simples e essencial, a movimentação do jogador.

Este início decidiu o rumo do meu jogo, visto que, após ter o *script* de movimentação feito, me ocorreram várias ideias boas que poderia implementar. Sempre gostei de ter um *dash* nos jogos que jogava e isto deu-me a vontade de implementar esta mecânica no meu.

Tendo os scripts de movimentação do jogador e *dash* programados, percebi o estilo de jogo que queria fazer. Lembrei-me de vários jogos do gênero que joguei em que me diverti imenso e cheguei à conclusão de que um jogo de *singleplayer survival* era a ideia mais apropriada para alcançar os meus objetivos.

Após ter tomado esta decisão as ideias vieram umas atrás das outras, sem parar. O objetivo agora era aproveitar as melhores e desenvolver algo divertido e simples.

Foi assim que o nome “**The Horde**” me veio à cabeça. A descrição principal seria um jogo em que o objetivo é fugir de mortos-vivos e ao mesmo tempo matá-los para conseguir passar para a próxima ronda.



Figura 5 - Jogo "Call of Duty Black Ops Zombies 4"

Para tal iriam haver várias armas diferentes assim como a habilidade de usar um *dash* para situações mais complicadas. Para conseguir as armas ou balas o jogador precisaria de dinheiro, este viria dos mortos-vivos que o mesmo mataria, assim como a chance destes deixarem cair uma caixa com um carregador de munição.

Com o dinheiro suficiente o jogador poderia dirigir-se a certas caixas de armas distribuídas pelo mapa, ao pagar o devido esta iria dar ao jogador uma arma ao calhas dentro das existentes.

O jogador também se poderia dirigir a outras localizações marcadas como “*shop*”no chão, para comprar munição para a sua arma atual.

Depois de conseguir uma arma mais poderosa o objetivo seria sobreviver o maior número de rondas possível. O problema seria que, à medida que as rondas passariam, a quantidade de mortos-vivos aumentaria, tendo os mais poderosos tipos a aparecer muito mais frequentemente.

Ao não conseguir sobreviver mais e morrer, o jogador seria levado para a sua *death screen* onde teria os seus resultados, mais concretamente a ronda a que chegou, a sua pontuação e dinheiro.



Figura - Quando o jogador abre uma caixa no jogo



Figura 7 - "Shop" no chão do jogo

Aqui entraria o website com a sua função de unir a comunidade do jogo. Nesta *death screen* estariam todas as informações necessárias para partilhar o recorde com outros jogadores através do website.

Para desenvolver um sistema sem burlas, o website necessitaria de administradores para controlar os recordes dos jogadores, isto é, averiguar as provas que o jogador enviou quanto aos valores que inseriu no recorde, após isso o administrador poderia rejeitar ou aceitar o recorde.

Caso o recorde fosse aceite este iria para a *leaderboard* global, caso contrário seria eliminado completamente dos registos do website.

Este sistema evitaria recordes falsos enviados por jogadores.

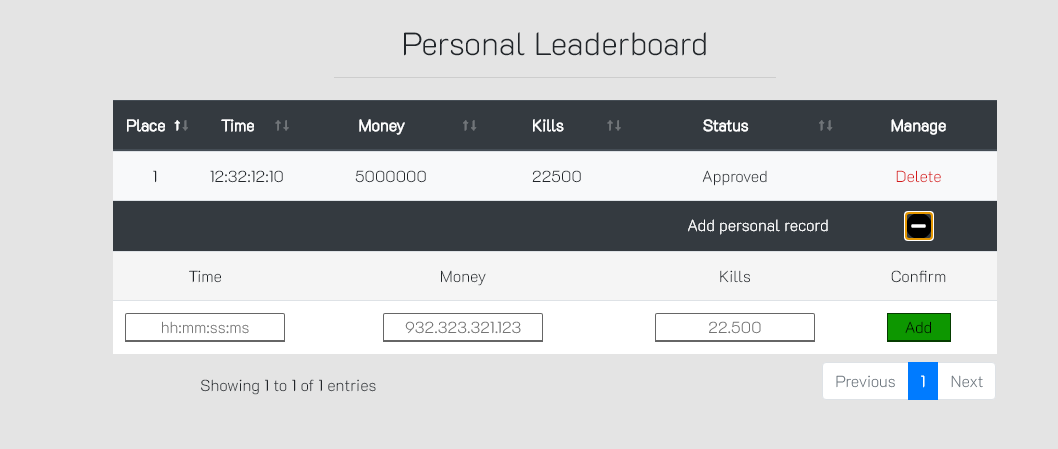


Figura - Leaderboard Pessoal

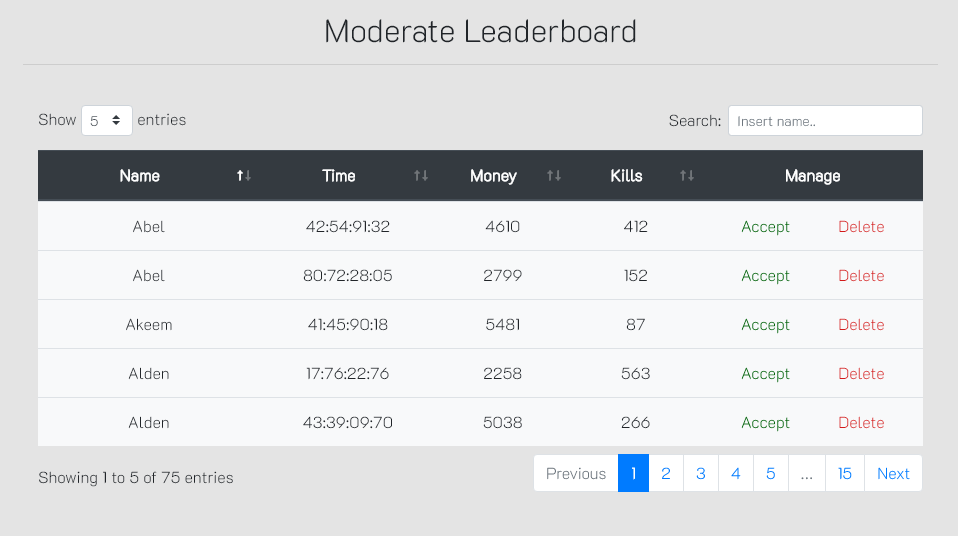


Figura 9 - Moderação da Leaderboard

**DESENVOLVIMENTO DO PROJETO**

Após ter a ideia geral em mente comecei a pô-la em prática. Iniciei esta etapa por desenvolver o jogo, assim poderia fazer as modificações necessárias até achar que o jogo estaria pronto sem ter de modificar o website à mínima mudança.

Antes de começar a programar, decidi criar e organizar uma tabela no **Trello**. Criei as listas “To do”, “Priority”, “Doing”, “Done” e “Ideas”. Traduzido significa, respetivamente, “Por fazer”, “Prioridade”, “A fazer”, “Feito” e “Ideias”.

Inseri vários cartões sobre tudo o que me vinha à mente relacionado o meu projeto, organizei pelas diferentes listas e decidi o que faria primeiro.

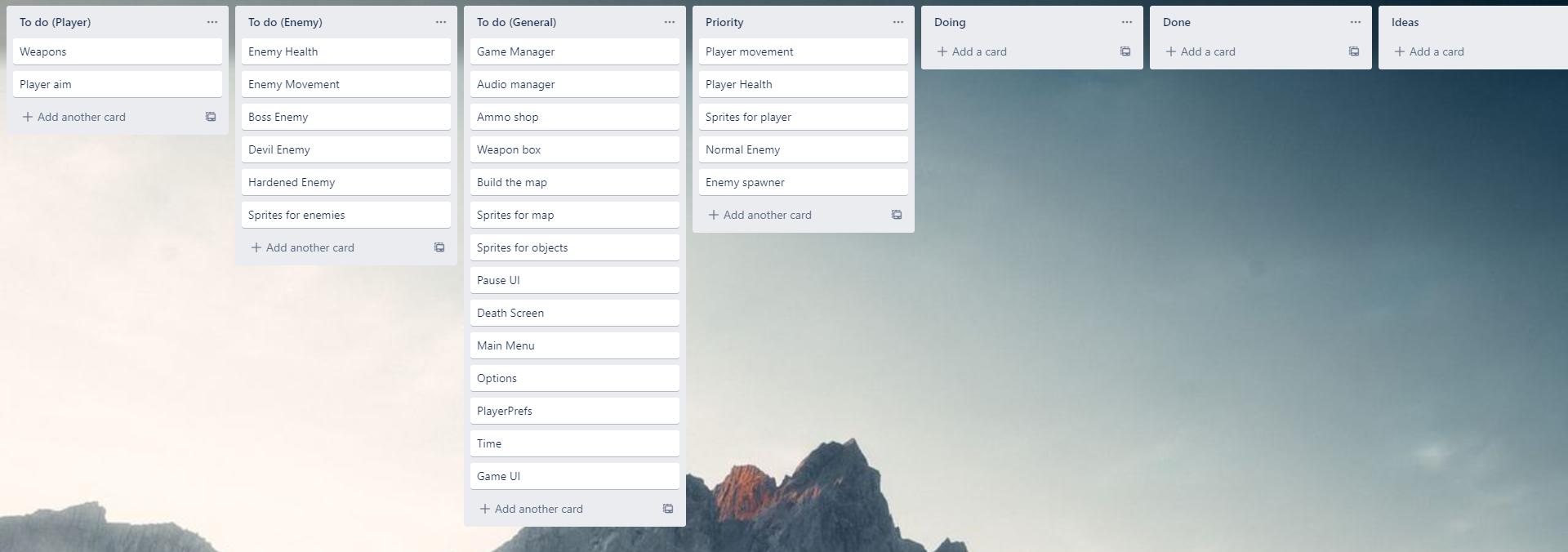


Figura - Trello sobre o jogo

Para o jogador, criei um *GameObject* e adicionei-lhe um *Sprite Renderer*, isto faria possível eu dar uma aparência ao personagem. Adicionei um *Rigidbody* para conseguir usar as físicas do jogo e também um *Collider* para ativar as colisões para o jogador.

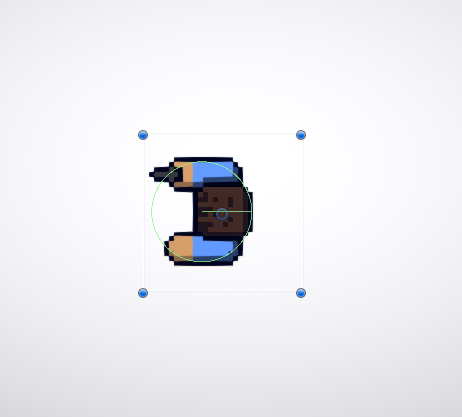


Figura - Personagem do jogador

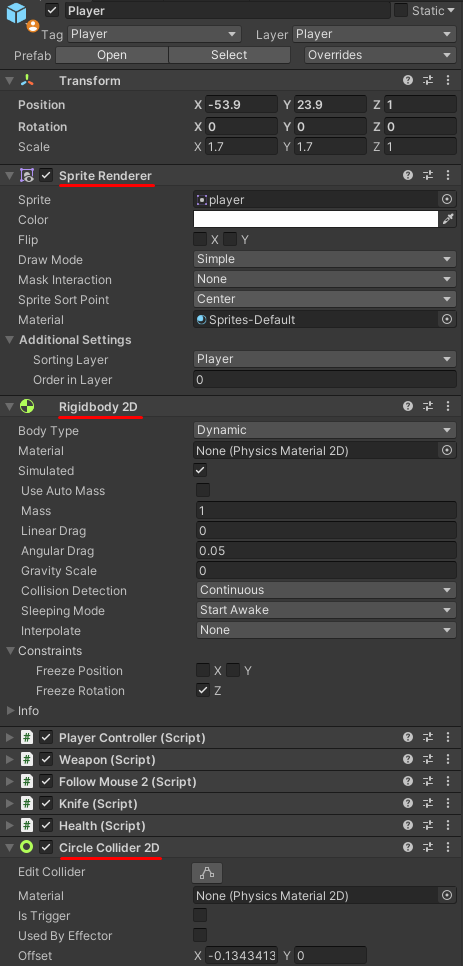


Figura - Script do personagem do jogador

De seguida criei um *script*, dei-lhe o nome de “Player Controller” e atribuí-o ao jogador. Este *script* contém todo o código necessário para a movimentação do personagem. Decidi utilizar a função **MovePosition()** do *Rigidbody* para realizar a movimentação do personagem, funciona perfeitamente para o que necessito e não faz o jogador teletransportar pelo mapa.

O *dash* no entanto seria um bocado mais complicado devido a este, no fundo, funcionar como um teletransporte para uma certa localização, isto daria a oportunidade ao jogador de sair para fora do mapa.

Para resolver este problema recorri às *Raycasts*, sempre que o jogador carregasse na tecla do *dash*, um *Raycast* seria enviado para a direção a qual este iria, verificando a posição do jogador pós-*dash*, caso não ultrapassasse a parede este faria o *dash* normalmente, caso contrário faria um *dash* porém para a mesma posição inicial.

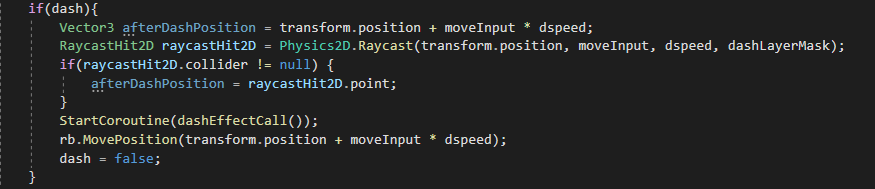


Figura - Código sobre o dash

Com estas duas mecânicas concluídas, desenvolvi um script para o personagem seguir o cursor. Para alcançar tal objetivo precisei converter as coordenadas do cursor na janela para coordenadas no jogo usando a função **ScreenToWorldPoint()** e de seguida trocar a rotação do personagem para estar de frente para onde o cursor apontaria.

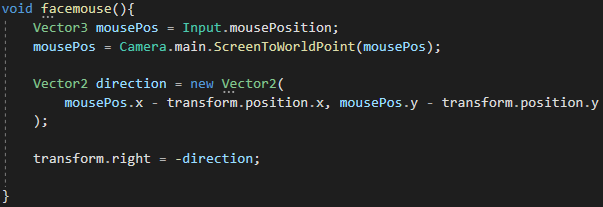


Figura - Código sobre o personagem seguir o cursor

Prossegui para o desenvolvimento das armas, criei um *script* e defini algumas variáveis para conseguir simular vários tipos de armas usando apenas um único script. Algo como velocidade de disparo, tempo entre cada disparo, dano de cada bala, tamanho do carregador entre outras variáveis igualmente importantes, serviriam para simular essa diferença de arma para arma.

No fundo, quando o jogador apanharia uma nova arma, os valores da sua atual arma seriam alterados para outros, isto em prática simula uma arma completamente diferente.

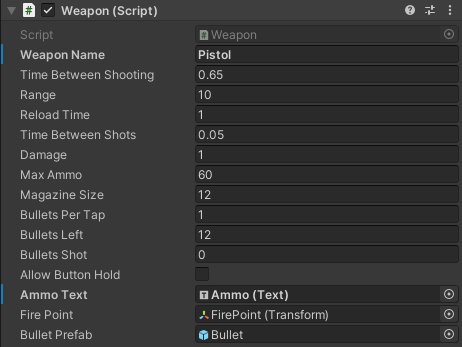


Figura - Script da arma

Para o jogador poder perder precisei criar um script de vida, este define a vida do jogador, recupera com o tempo e tira quando necessário. Também implementei um tipo de energia para o jogador conseguir usar apenas dois *dashes* ate ficar sem energia, esta também se recupera com o tempo.

Por fim faltaria dar ao jogador uma maneira de atacar caso a sua arma ficasse sem balas, para tal criei uma espécie de espada/faca que tiraria vida se acertasse no inimigo. Dei-lhe uma aparência, um *collider* e criei uma animação que imita um ataque, adicionei um *script* ao jogador para conseguir acionar a espada e animação quando carregasse na tecla certa.

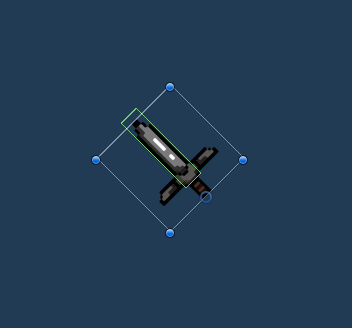


Figura - Faca do jogador

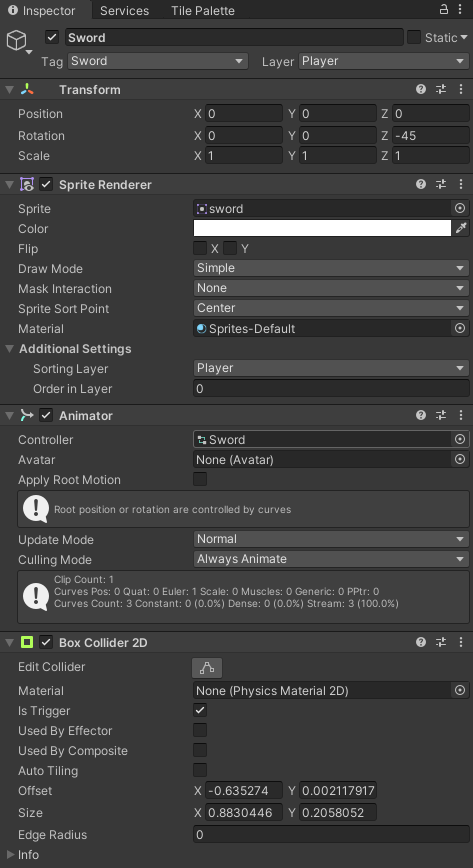


Figura 17 - Código da faca do jogador

De seguida iniciei a fase dos mortos-vivos, queria ter quatro tipos de inimigos diferentes, cada um mais difícil do que o anterior, com mais vida e mais dano por ataque mas também um bocado mais lento.

Para aprender sobre *AI* segui um mini-curso do **Unity** que ensina tudo o que é necessário aprender. Após visualisar os vídeos, experimentei em um projeto à parte para me acostumar com as funções de *AI*. Já no meu projeto, criei um novo *GameObject*, atribui-lhe um *Sprite Renderer* e uma aparência, um *Collider* e um *Rigidbody*, tal como no jogador.



Figura - Código do "Zombie"

Criei um *script* novo e atribuí-o ao *GameObject* do morto-vivo. Este, em conjunto com um *asset* chamado **NavMeshPro**, este ajudaria o morto-vivo a ir atrás do jogador procurando a melhor rota, isto faria os inimigos irem dar a volta à parede quando necessário e não irem contra ela caso o jogador tivesse do outro lado. Acrescentei também outro *script* que trataria de atacar o jogador caso estivesse a uma certa distância. Também é este *script* que verifica a vida do inimigo, ou seja, se tiver com 0 de vida este irá ser eliminado, tendo a chance de deixar cair uma caixa com munição.

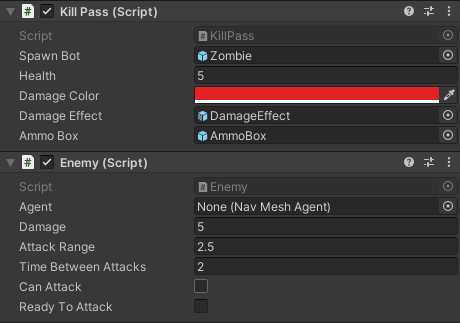


Figura - Scripts do "Zombie"

Com este *GameObject* feito, bastava duplicar e mudar o necessário para fazer outros inimigos, com mais dano e mais vida, entre outros fatores. Assim criei 4 inimigos diferentes, o “Zombie”, “Reinforced Zombie”, “Devil” e o “Boss” que aparece a cada 3 rondas.



Figura - Personagem do "Zombie"



Figura - Personagem do "Reinforced Zombie"



Figura - Personagem do "Devil"



Figura - Personagem do "Boss"

Estava na hora de trabalhar no sistema de rondas e inimigos, onde apareceriam, em que quantidade, quando é que passaria para a próxima ronda e assim por diante. Este sistema foi muito complexo pois requeria dos cálculos certos para o jogo ficar perfeito. Precisava também de aumentar o nível de dificuldade em a cada ronda mas com limites, isto é, a partir de uma certa ronda teriam de aparecer a mesma quantidade de inimigos porém os tipos mais fortes em maior quantidade a cada ronda que passaria.

Comecei por tentar encontrar funções matemáticas para conseguir o número que queria em perfeição, criei um ficheiro **Excel** para tentar representar o resultado antes de o colocar no *script*.

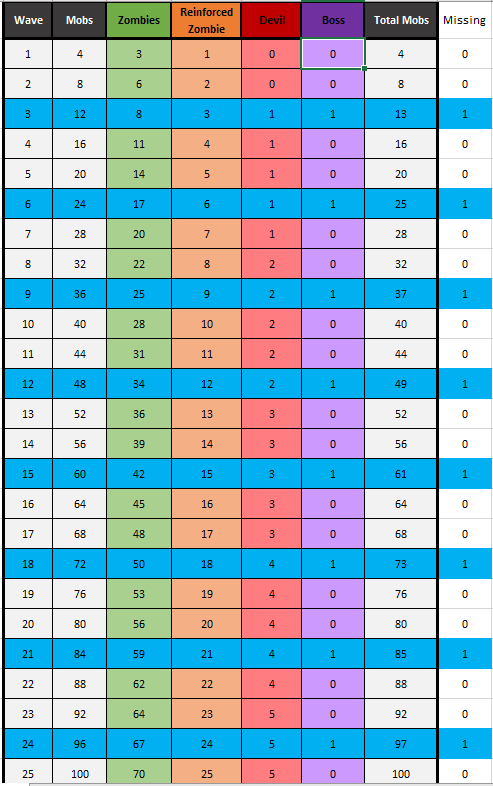


Figura - Excel sobre o sistema de rondas

* Para o número total de inimigos decidi que multiplicar o número da ronda por 4 seria uma boa quantidade. Claro que isto não daria sempre número certo, então usei uma função para arredondar o número para um inteiro.
* Como os “Zombie” seriam os inimigos mais fracos, precisariam de aparecer em maior quantidade logo dei-lhes 70% do número total de inimigos na ronda.
* Para os “Reinforced Zombie” dei 25% pois precisava que estivessem la em boa quantidade mas não exagerada.
* Os “Devil” iriam aparecer cada vez mais dependendo da ronda, para tal decidi que usar o restante dos inimigos seria uma boa opção, ou seja, calcularia a soma dos “Zombie” e “Reinforced Zombie” e faria o resto da divisão com o número total de inimigos na ronda.

Após imensos testes e tempo investido, consegui alcançar uma fórmula perfeita para o que precisava. Agora bastaria colocar os “Boss” a aparecer a cada 3 rondas, o que foi fácil. Calculando o resto da divisão do número total de inimigos na ronda pelo número 3 conseguiria o resultado 0 ou maior. Caso o resultado fosse 0 seria uma ronda de chefe, caso contrário seria uma ronda normal.

Com a fórmula perfeita para o meu objetivo, apenas restava aplicá-la em código, dentro de um *script*. Dito desta maneira parece fácil, porém este foi, de longe, o *script* mais complexo e complicado que tive de escrever ate ao dia, bastava a mínima desatenção para perder todo o raciocínio.

Iniciei o *script* por criar as variáveis que achasse necessárias, algumas delas precisariam ser em formato de arrays para otimizar o código.

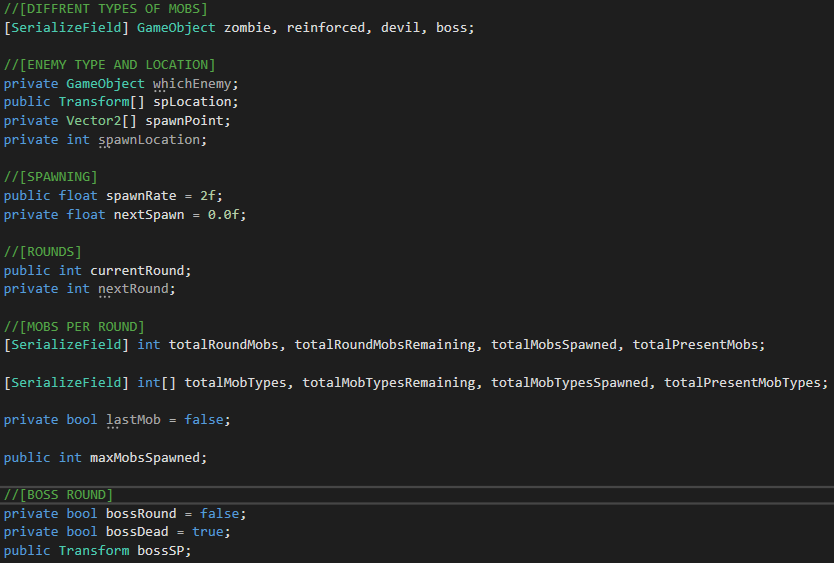


Figura - Variáveis do script de rondas

Após ter as variáveis criadas, defini tudo que era necessário para quando o jogo fosse iniciado, ou seja, funcionaria como um reset.

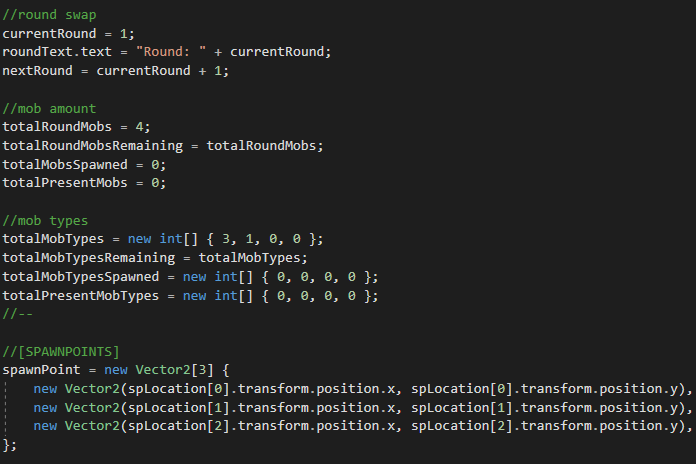


Figura - Função Start() do script de rondas

De seguida escrevi o código necessário para a mudança de ronda, isto aconteceria quando o número total de inimigos presentes atingisse o número 0 e caso fosse uma ronda de chefe, este precisaria estar morto também.



Figura - Parte do código de rondas onde troca a ronda

Aqui estariam as funções matemáticas necessárias para conseguir o resultado que consegui na folha de **Excel**. Com os números certos definidos, faltava-me fazer os inimigos aparecerem no mapa. Para conseguir alcançar este objetivo criei 3 localizações no mapa por onde os inimigos poderiam aparecer e atribuí as coordenadas dentro de uma array. Utilizei a função **Random.Range()** para conseguir um número ao calhas entre 0 e 3 para usar como índice da minha array, desta maneira conseguiria escolher uma localização aleatória para o inimigo aparecer.

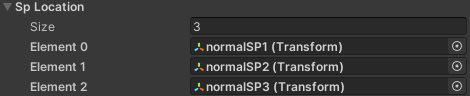


Figura 28 - Array das localizações de aparecimento dos inimigos

Parecia estar tudo feito porém os inimigos apareciam do mais fraco para o mais forte, ou seja, em um exemplo de 10 fracos, 5 médios e 3 fortes, apareceriam os 10 fracos primeiro, seguidos pelos 5 médios e por fim os 3 fortes, o que não estava certo.

Para corrigir esta situação inconveniente utilizei a mesma função que anteriormente, desta vez iria conseguir um número entre 1 e a soma da quantidade de todos os tipos de inimigos.

Seguidamente escrevi um código que me permitisse saber que inimigo é que iria aparecer tendo em conta o número aleatório conseguido.

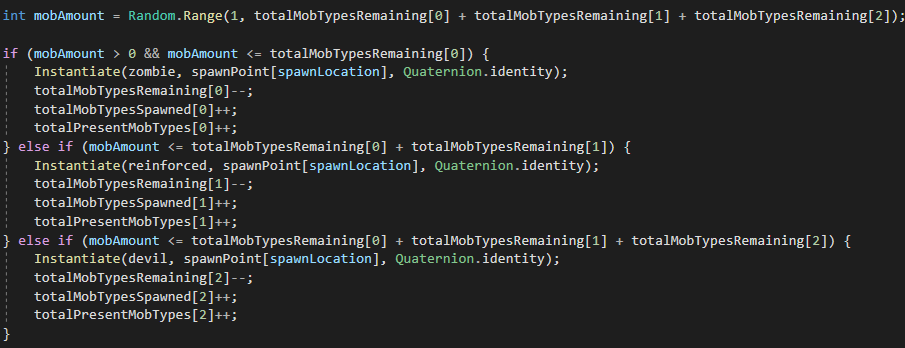


Figura - Parte do código de rondas que trata de fazer os inimigos aparecerem

Como também haveriam rondas de chefe, precisaria que o este também tivesse uma chance de aparecer, assim conseguiria evitar que fosse o primeiro ou o último inimigo a aparecer, usei a mesma técnica que anteriormente porém apenas para o chefe. Assim concluiria este script essencial para o jogo.

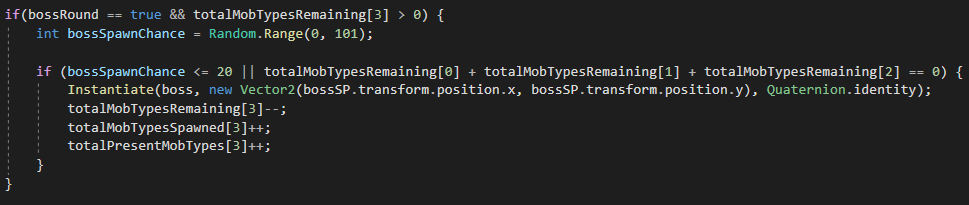


Figura 30 - Parte do codigo de rondas que trada de fazer o chefe aparecer

Para o *UI* criei um *Canvas* com a opção de *Scale With Screen Size* e defini a resolução inicial para 1920x1080 píxeis, visto ser a resolução do meu monitor. Esta opção faria com que todos os componentes colocados no *UI* fossem redimensionados conforme a resolução do ecrã, mantendo-os sempre proporcionais.

Em primeiro lugar criei um painel para quando o jogo estivesse a carregar. De seguida criei o menu principal com os botões de **iniciar**, **opções** e **sair**. Por fim criei o menu das opções onde o jogador poderia manipular o volume do jogo, este teria 3 opções, o **volume global** que iria manipular todo o resto, o **volume da música** que apenas manipularia a música de fundo, e o **volume dos efeitos**, esta opção controlaria todos os efeitos sonoros como o som das balas, dos monstros e por aí adiante.



Figura - Janela de carregamento do jogo



Figura - Menu principal do jogo



Figura - Menu de opções do jogo

Como não queria ter problemas com direitos de autor, pesquisei por vídeo-tutoriais sobre como criar efeitos sonoros para um jogo. Aprendi que ao gravar um som no meu microfone e editá-lo utilizando o programa Audacity conseguia alcançar inúmeros sons diferentes para qualquer situação que me fosse necessária.

Não tendo os materiais necessários para fazer sons de balas e de recarregamento de uma arma, pesquisei por sons de uso gratuito, desta maneira não iria ter problemas com direitos de autor.

O resto dos sons foram gravados no meu microfone e modificados por mim, sons como o do *dash*, o de apanhar um carregador do chão, os inimigos a levarem dano e a morrerem entre outros.

Abaixo está um exemplo do som em que o jogador leva dano. Gravei um audio parecido ao pretendido e modifiquei usando os diferentes efeitos disponíveis.

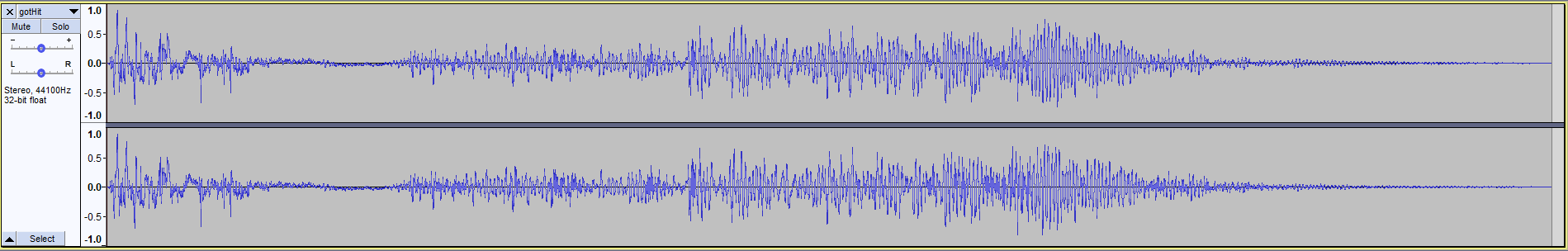


Figura - Áudio gravado por mim

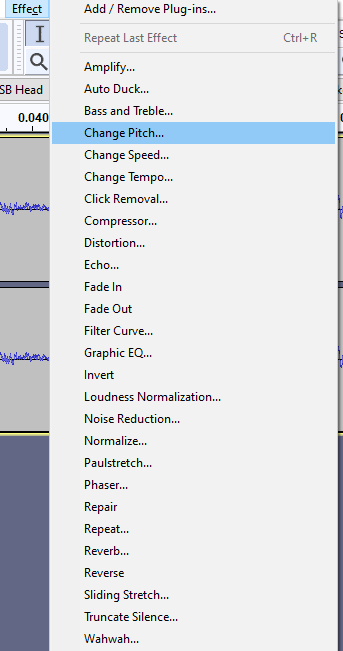


Figura - Menu de efeitos aplicáveis ao áudio utilizando o Audacity

Após ter os audios todos necessários faltava colocá-los no meu projeto e distribuí-los aos diferentes *GameObjects*, para tal aprendi a fazer um *script* que funciona como um grupo global, adiciona-se os sons que precisamos dentro de uma array e modificamos com as devidas definições que precisamos. Quando precisarmos utilizar um certo som, podemos utilizar uma função pública deste script em outros, esta irá acionar o som.

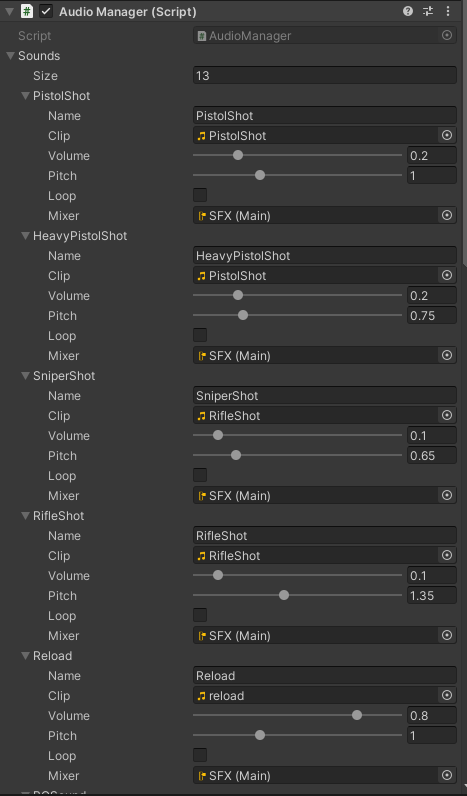


Figura - Script que trata de gerir o áudio

No caso do som de fundo, queremos que este seja iniciado logo quando o jogo comecar, para tal apenas chamamos a função de tocar música dentro da função **Start()** do script.

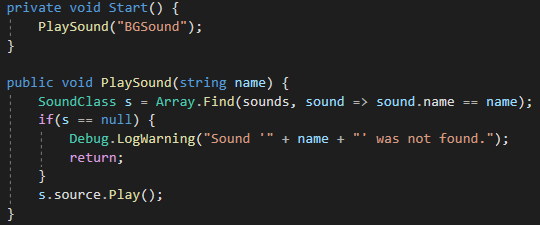


Figura - Código do script de áudio que contém a função PlaySound()

Por fim e para acabar o jogo faltava adicionar efeitos de pós-processamento, isto daria ao jogo uma aparência mais amigável e bonita. Para alcançar este objetivo utilizei o *asset* de pós-processamento do **Unity**.

Criei um volume de efeitos, adicionei os efeitos que pretendia e modifiquei os valores consoante o necessário.

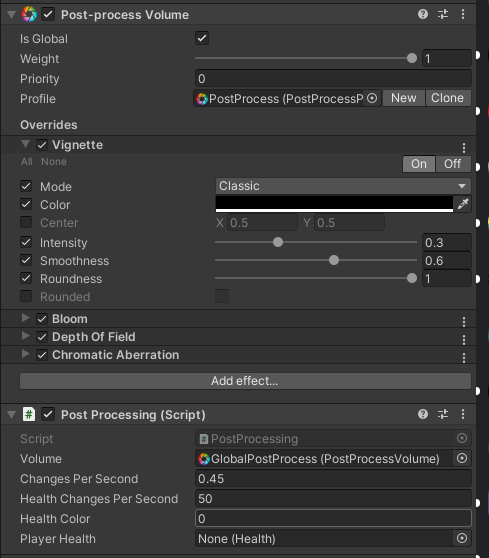


Figura - Volume de pós-processamento e script respetivo

Para adicionar mais interação com o jogo, adicionei um script neste *GameObject* que trataria de mudar certos valores nos efeitos de pós-processamento. Por exemplo, quando o player utilizaria o *dash*, este iria ver a câmera um bocado mais escura seguido de um lento aclaramento até ficar normal, dando uma imersão maior. Também foi com este script que modifiquei a cor da *Vignette* para ficar vermelha quando o jogador estivesse abaixo de uma certa quantidade de vida, dando-lhe a entender que estava quase a perder.

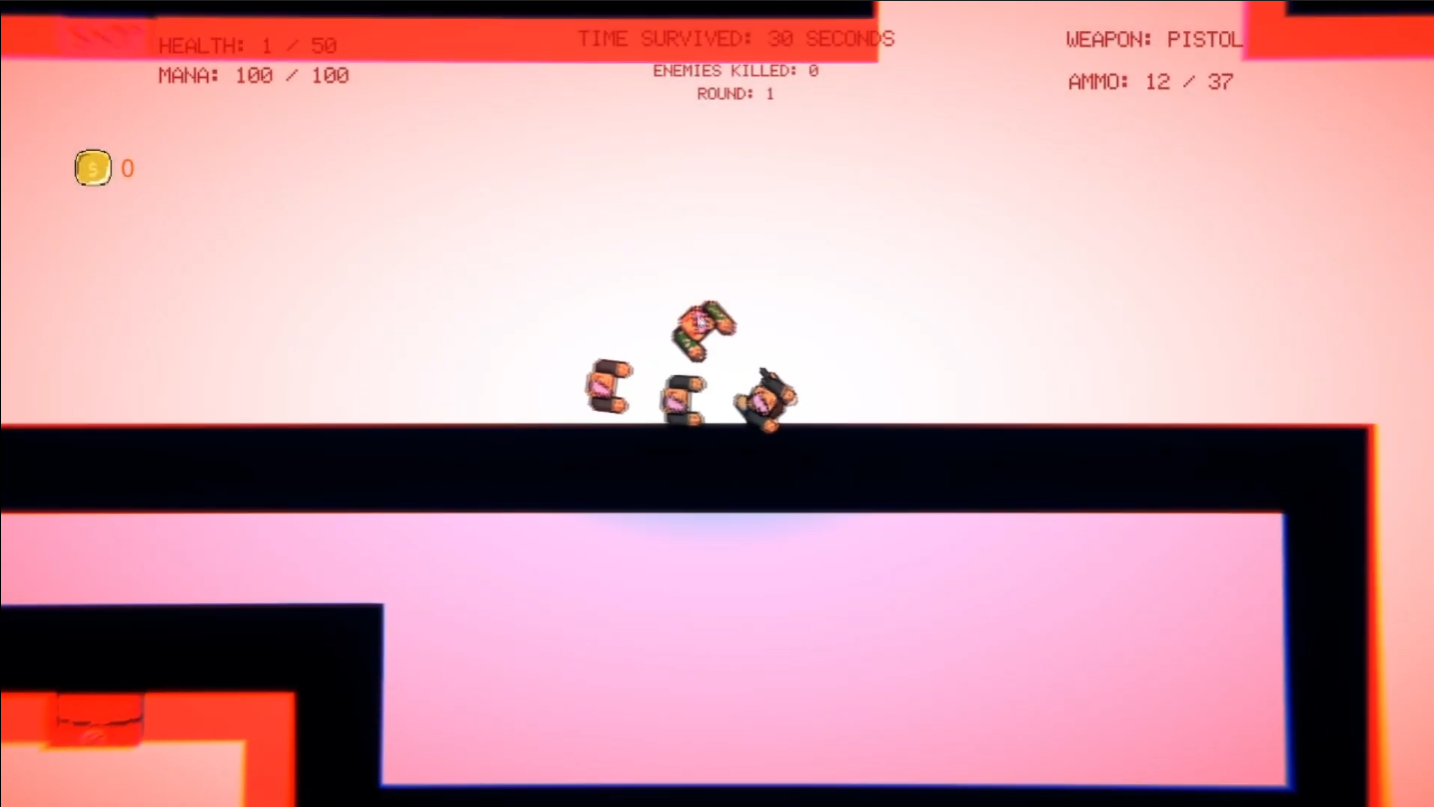


Figura - Imagem de quando o jogador está quase a perder no jogo

Com o jogo concluído faltava desenvolver o website. Para a realização deste, decidi aprender a utilizar a ferramenta **Bootstrap**, que me daria a oportunidade de programar de forma mais eficaz, simples e bonita.

Em primeiro lugar desenvolvi a barra de navegação, seria a parte mais essencial pois é utilizando esta que se conseguiria navegar pelas diferentes páginas do website. Coloquei o nome do jogo na esquerda e quatro opções na direita.



Figura - Barra de navegação do website

A primeira, “Home”, levaria á página inicial. Aqui estaria apresentado um subtítulo para o jogo, mais abaixo estaria o trailer do jogo e no final estaria uma descrição sobre o jogo.

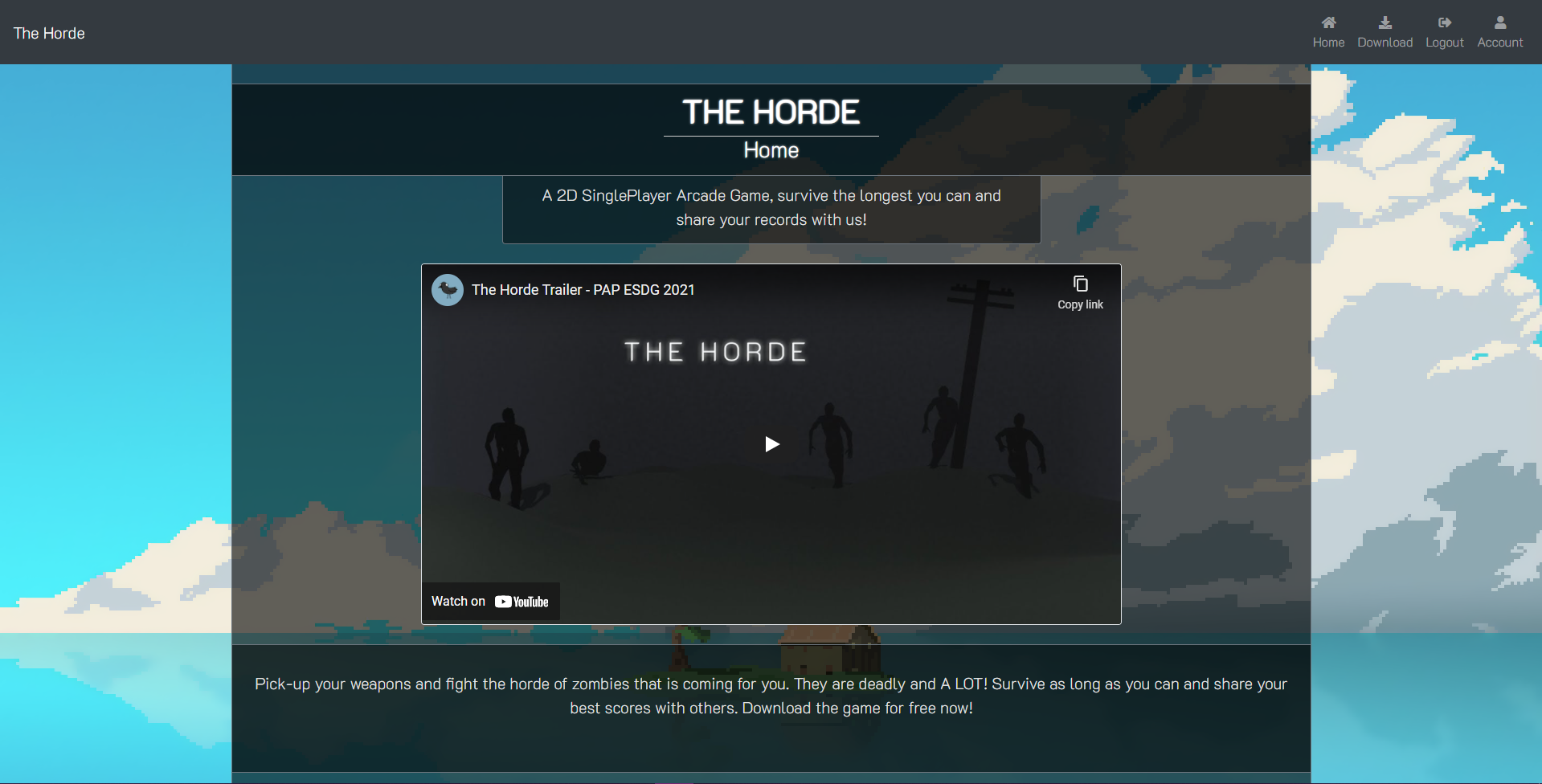


Figura - Página inicial do website

A opção “Download” levaria o utilizador á pagina de descarregamento do jogo, esta poderia ser feita de várias maneiras, contendo também uma hiperligação que leva para um website para provar que os ficheiros não contém vírus.

Esta página apenas mostraria as diferentes hiperligações de descarregamento do jogo caso o utilizador estivesse logado no website.

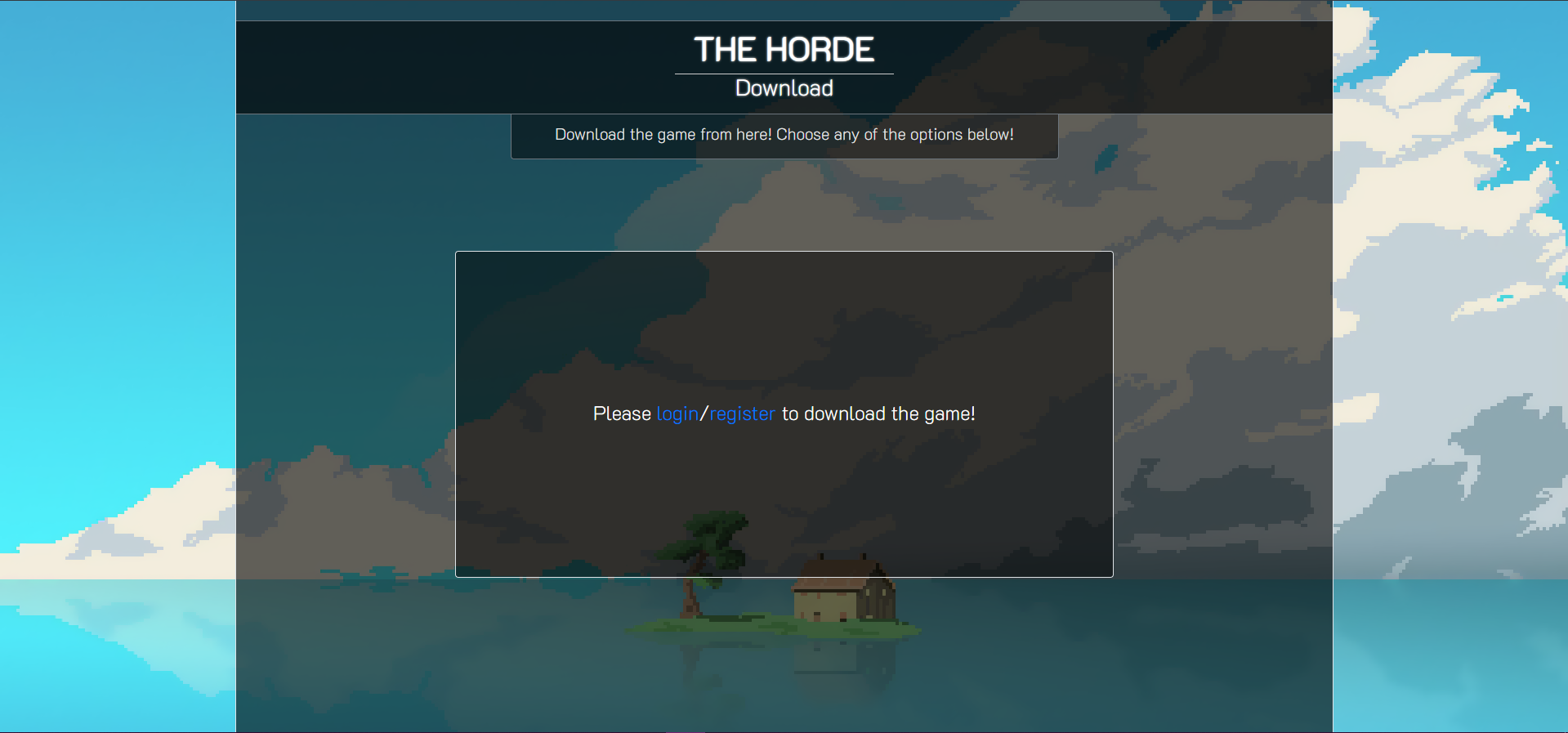


Figura - Página de download do website quando o utilizador não está logado

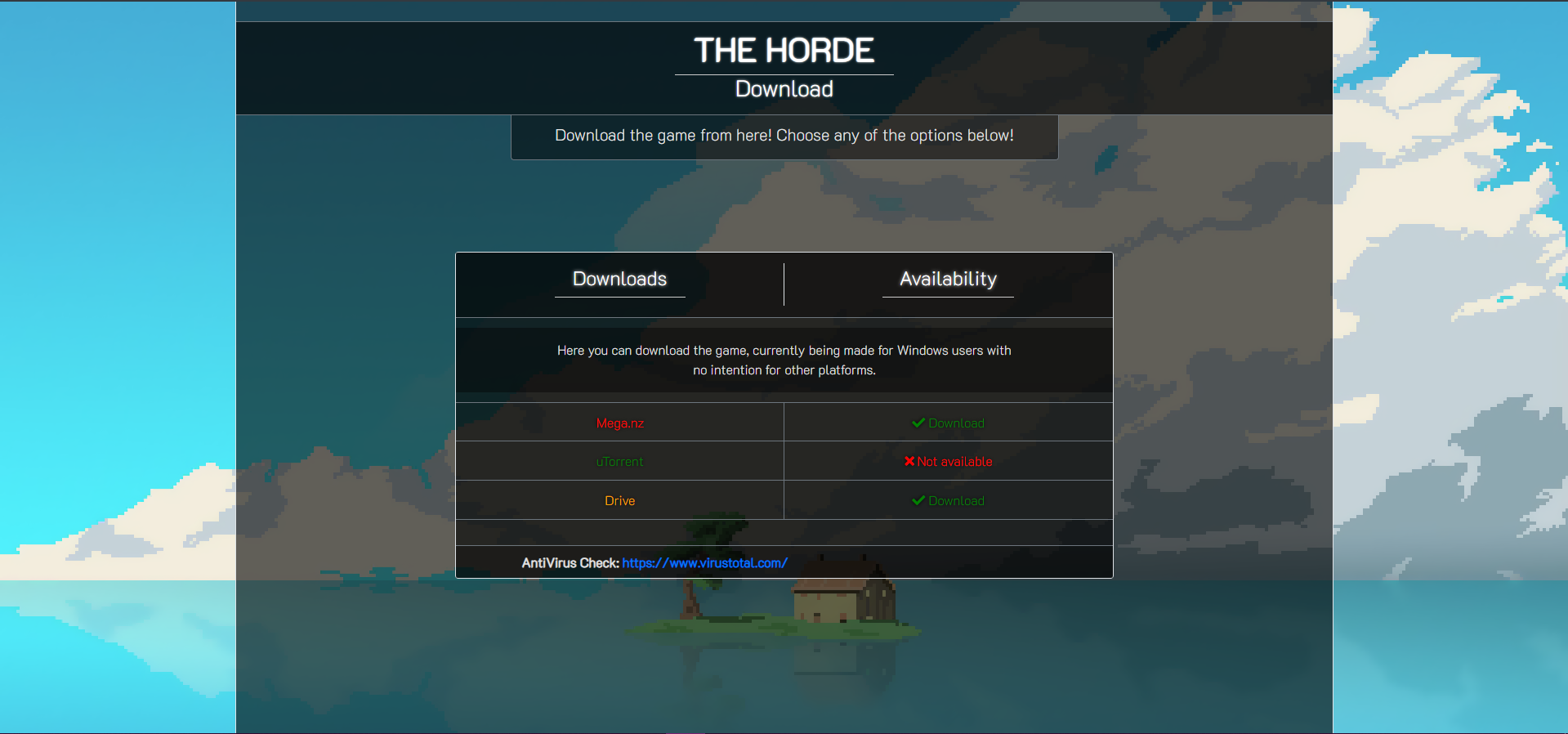


Figura - Página de download do website quando o utilizador está logado

A opção “Register” levaria à página de registo, para efetuar este seriam necessários vários dados:

* Um nome de utilizador que seria utilizado posteriormente para efetuar o login.
* Um email válido, pois serviria para confirmação mais tarde.
* Uma palavra-passe e a confirmação da mesma.
* A data de nascimento, restringindo o registo a menores de 16 anos.
* O país onde o utilizador está, este seria usado para dados estatísticos apenas.
* (**Opcional**) Um Referral ID, este sistema serviria para os administradores do website saberem de onde é que os jogadores viriam. Estes ID’s seriam dados a divulgadores do website/jogo.
* Concordar com os termos e condições do website.

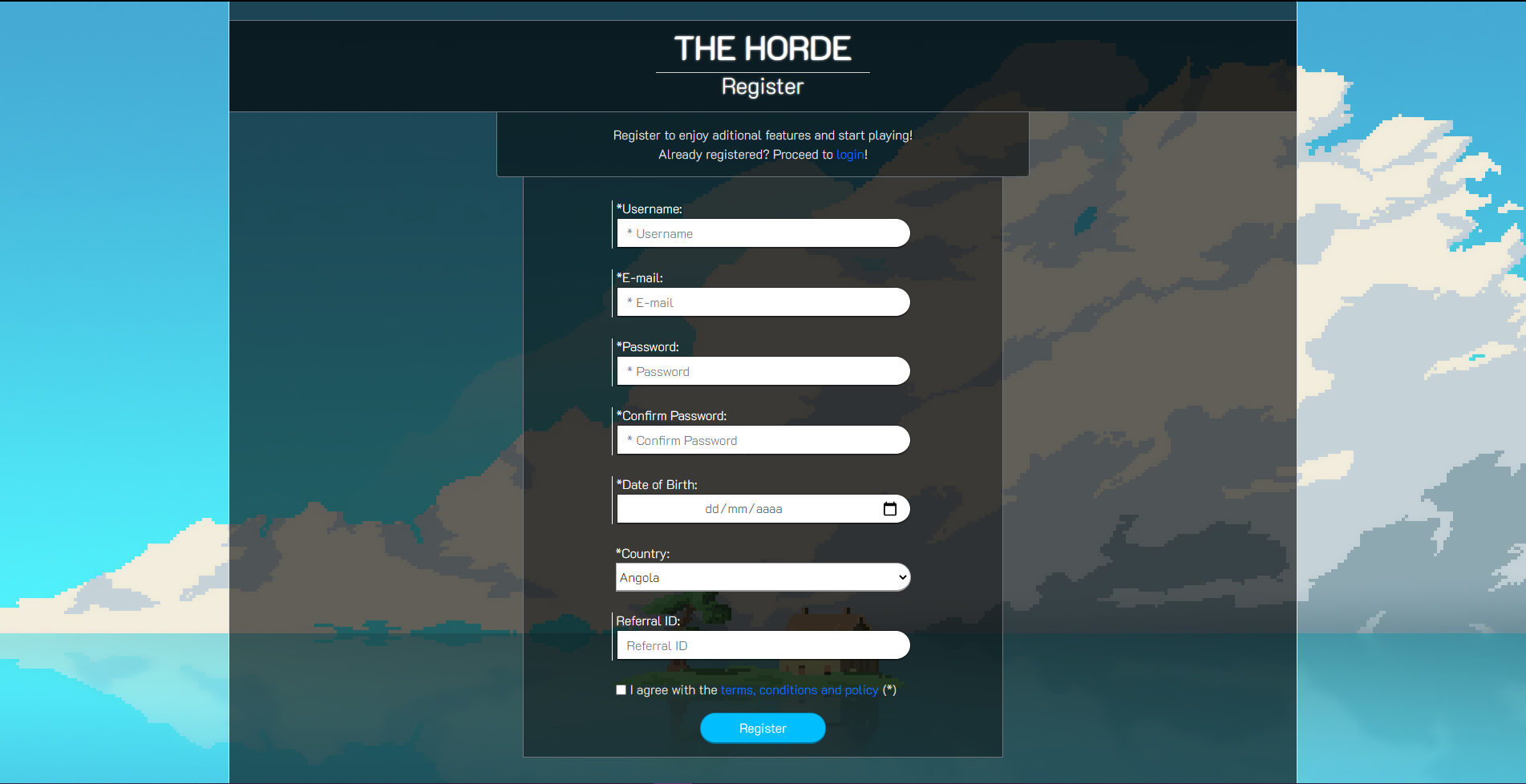


Figura - Página de registo do website

A opção de “Login” levaria à página de login no website. Esta precisaria de um nome de utilizador e uma palavra-passe corretos para levar o utilizador às abas de utilizador registado.

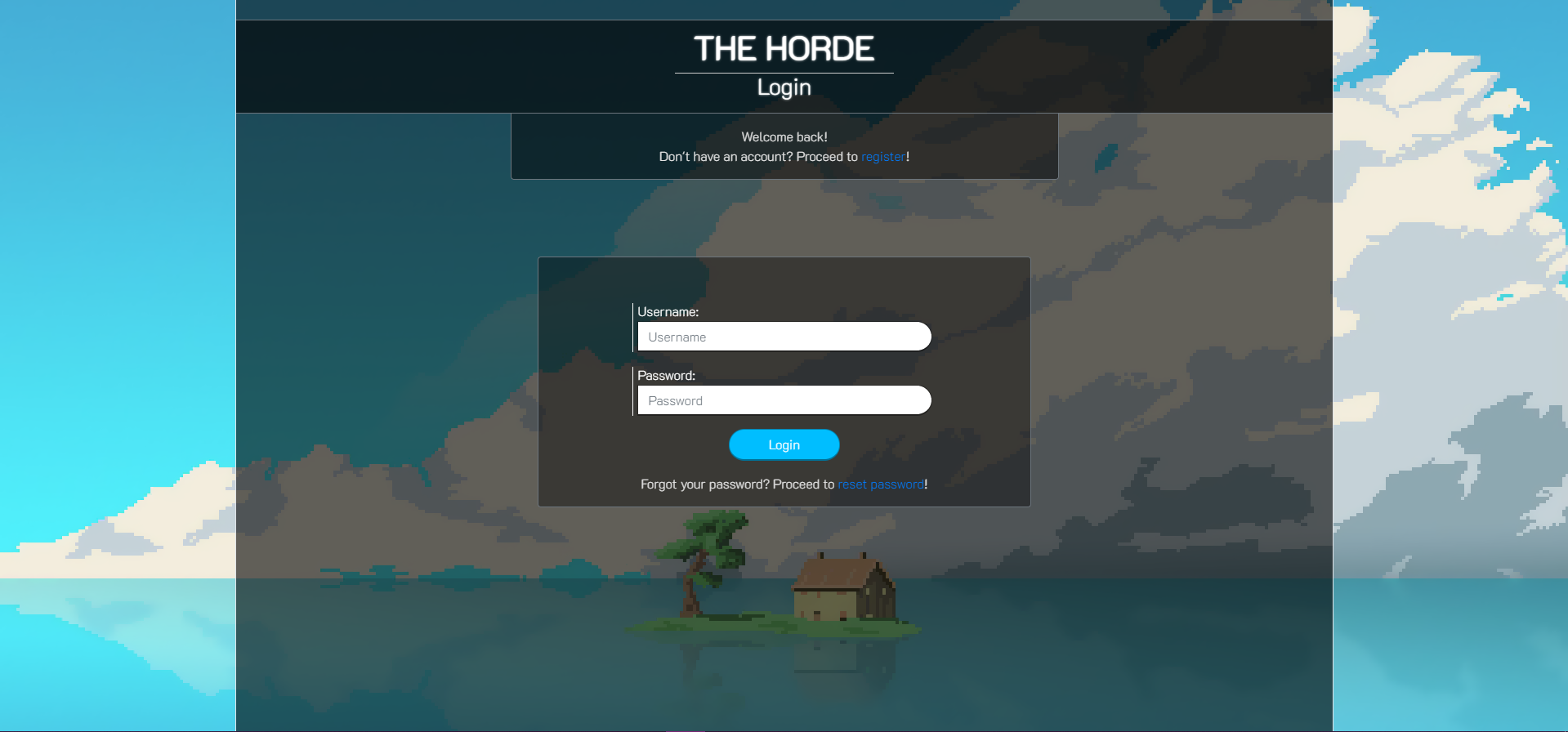


Figura - Página de login do website

Após estar logado, o utilizador receberia acesso a uma opção nova chamada “Account”. Carregar nesta opção abriria um painel no website do utilizador com outras novas opções.

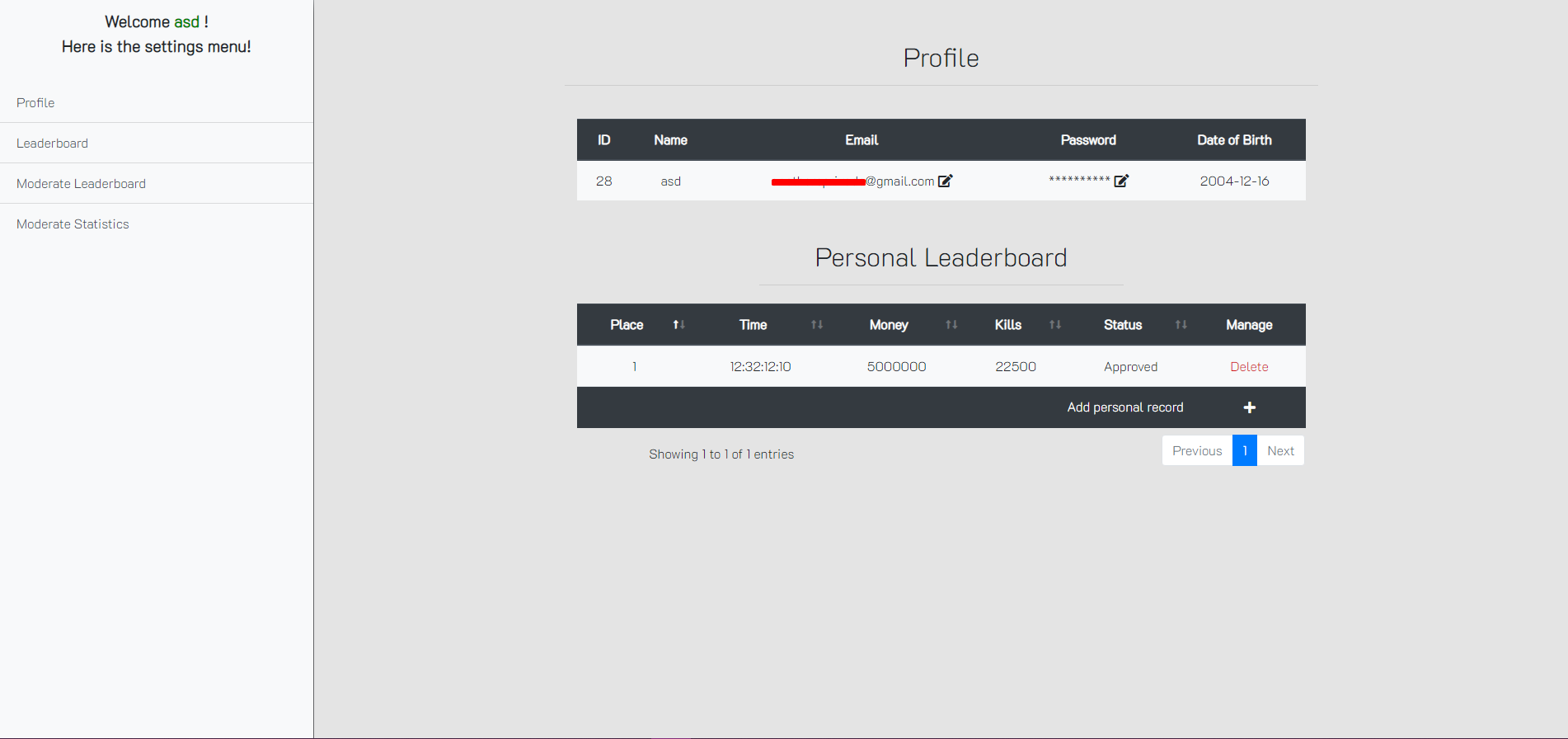


Figura - Página do perfil do utilizador no website

Estando na página “Profile” mostraria ao utilizador os seus dados assim como a opção de editar alguns deles. Também é nesta página que o utilizador consegue visualizar todos os seus recordes pessoais. Estes estão divididos em 6 colunas:

* A primeira indica o lugar do recorde na leaderboard global
* A segunda indica o tempo que o jogador sobreviveu.
* A terceira indica o dinheiro do jogador no momento em que perdeu.
* A quarta indica o número de inimigos que o jogador derrotou.
* A quinta coluna indica o estado do recorde, se este já foi aceite pelos administradores ou se ainda está em espera.
* A sexta coluna permite ao jogador eliminar recordes pessoais caso não os queira ter mais.

Abaixo dos recordes o utilizador tem a possibilidade de enviar novos pedidos, para tal precisa colocar todos os dados necessários, estes dados serão todos demonstrados no jogo, após o jogador perder.

A última página à qual o utilizador tem acesso é “Leaderboard” onde se encontra a leaderboard global. Nesta o jogador consegue pesquisar por um certo recorde utilizando as diversas opções de pesquisa e organização.

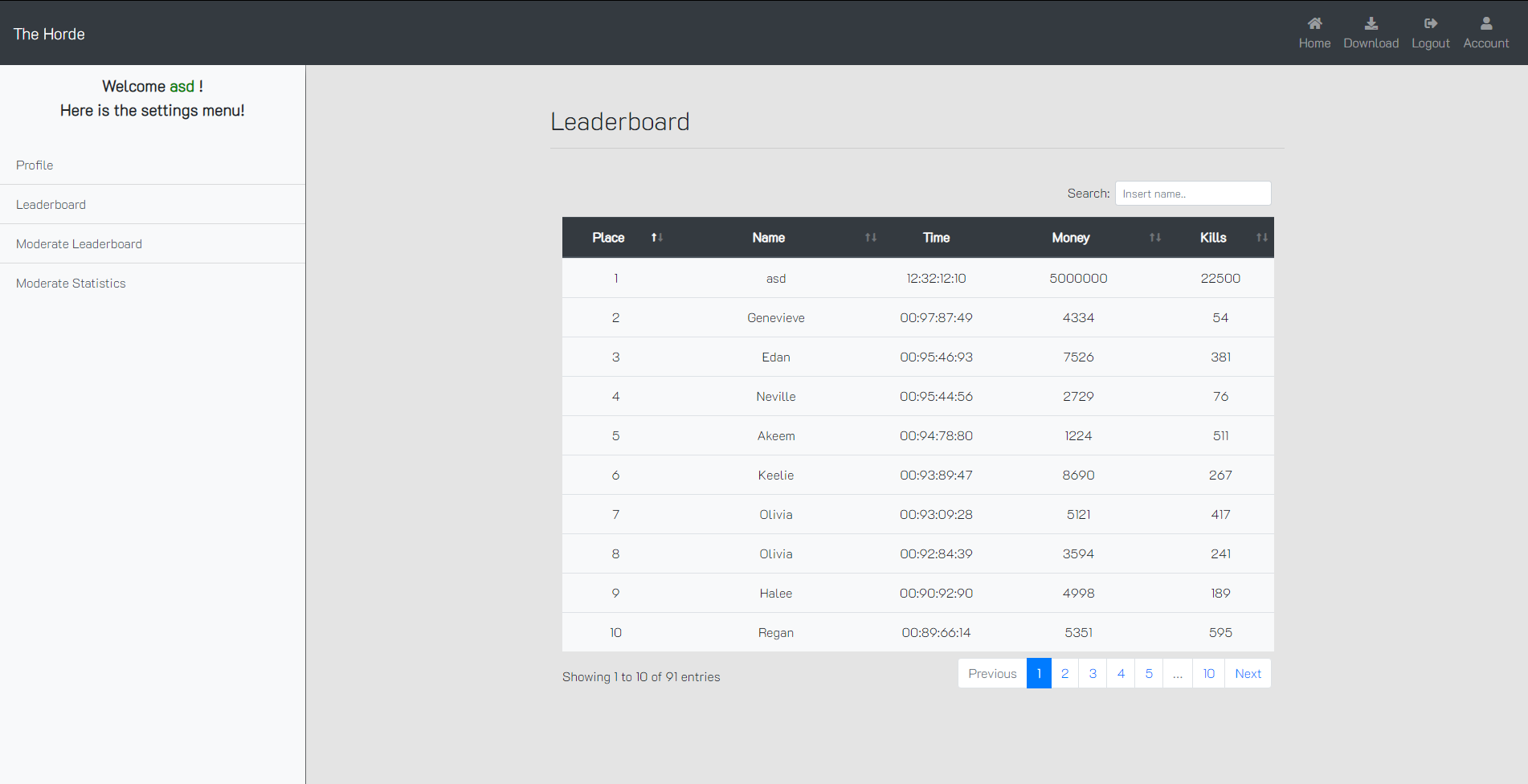


Figura - Página da Leaderboard no website

A página “Moderate Leaderboard” é restrita apenas para os administradores, nesta página estes conseguem gerir os recordes que vão sendo adicionados.

Estes também têm a opção de pesquisar e organizar de diversas maneiras.

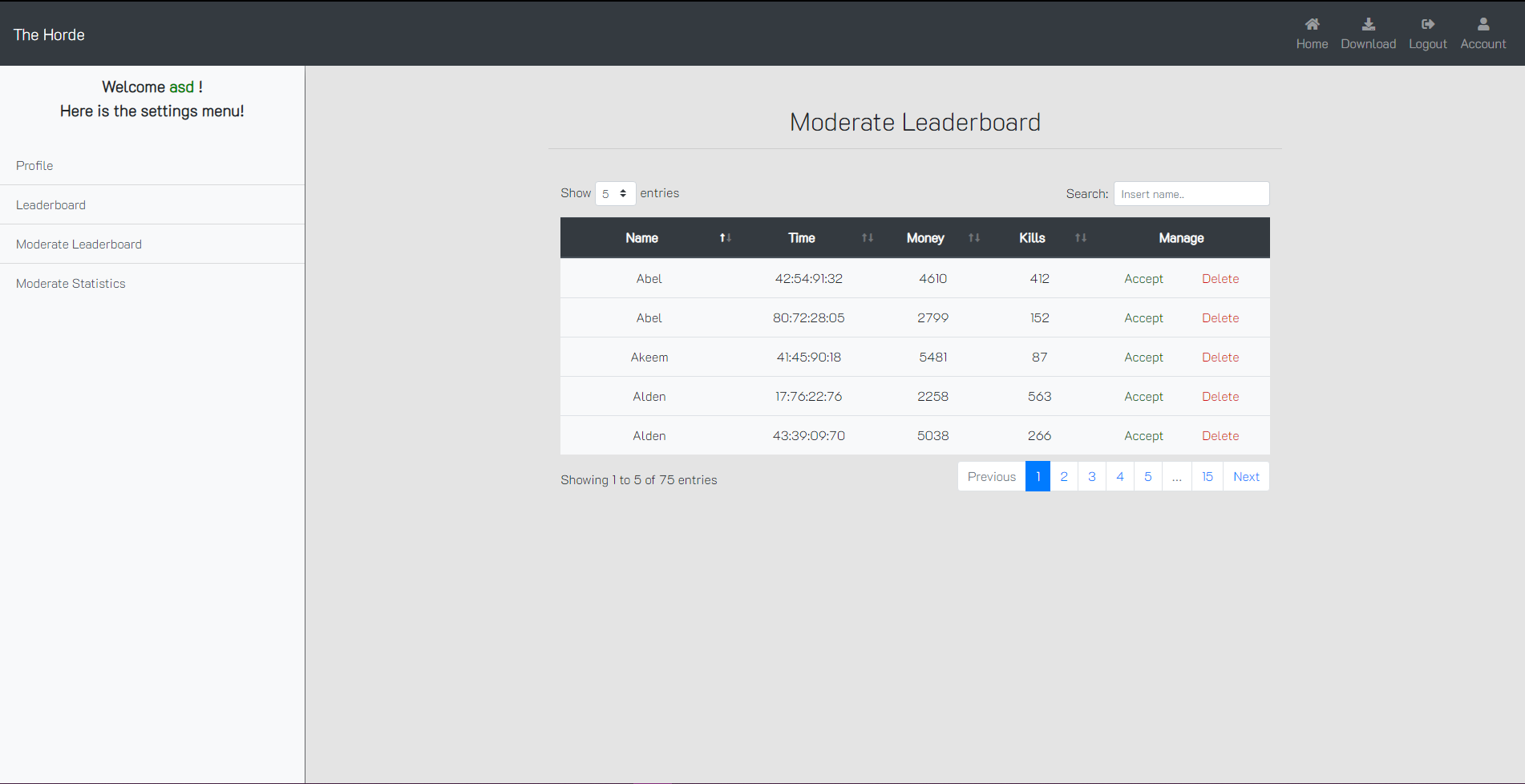


Figura - Página de moderação da leaderboard no website

Por fim a página “Moderate Statistics”, também restrita apenas para os administradores, contém um mapa com alguns países que estão disponíveis no registo, este irá preencher o mapa contando os utilizadores registados de cada país. Os administradores também conseguem ver o número total de utilizadores registados.

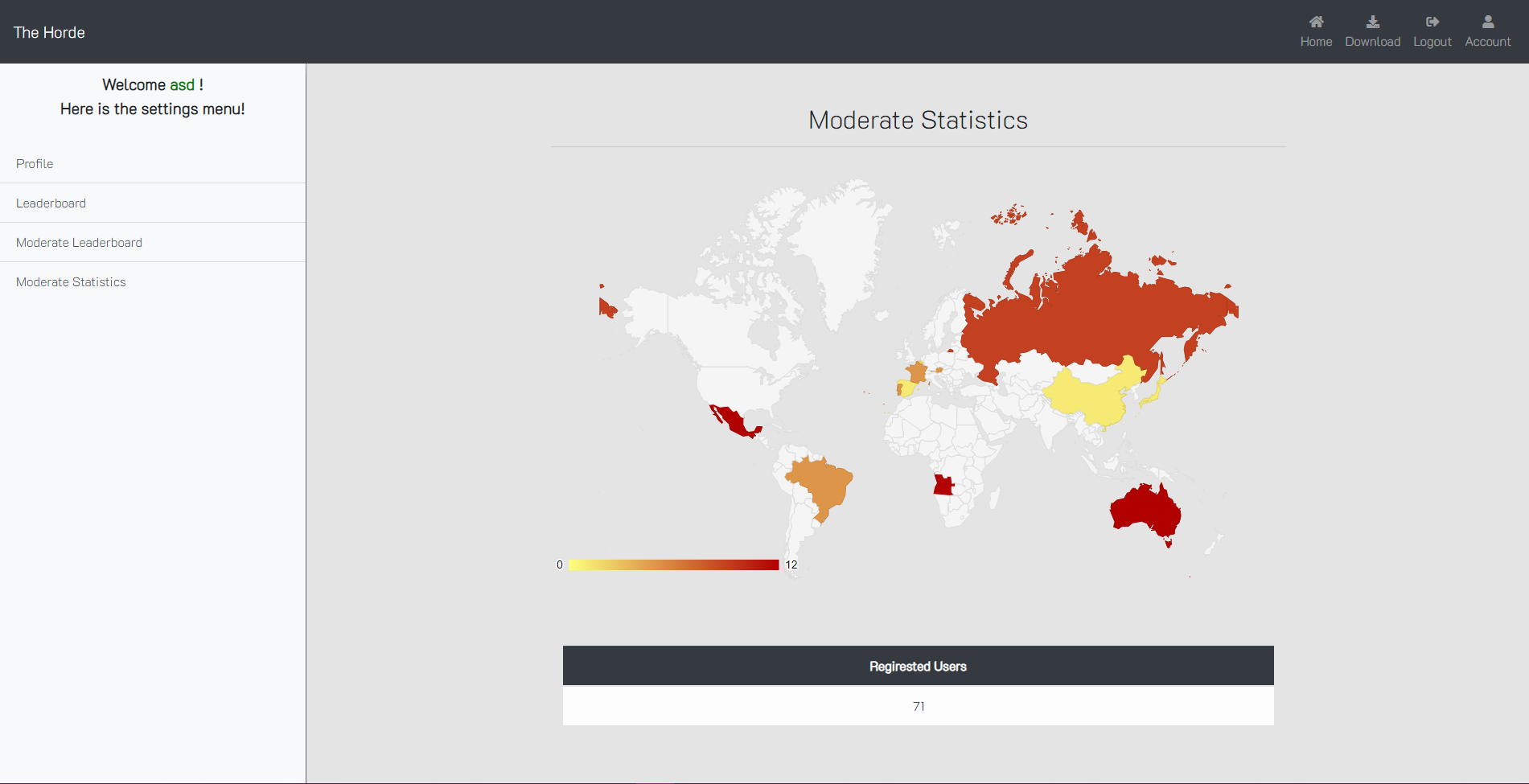


Figura - Página de estatísticas do website

Para conseguir este mapa utilizei um código disponível no **Google Charts**. Modifiquei o tamanho do mapa, os países e as cores ao meu gosto e implementei o código no website.

Com esta etapa feita concluí o desenvolvimento do website.

**RECURSOS UTILIZADOS**

**Unity** – Engine principal utilizado para o desenvolvimento do jogo.

**Audacity** – Software utilizado para a criação de efeitos sonoros para o jogo.

**Photoshop** – Software utilizado para a criação de diversas imagens para o website/jogo.

**Aseprite** – Software utilizado para a criação de pixel arts para o jogo, como a aparência dos personagens, inimigos entre outros.

**Visual studio 2019** – Software utilizado para a criação e edição de código para o jogo.

**Visual studio code** – Software utilizado para a criação e edição de código para o website.

**XAMPP** – Software utilizado para conseguir uma base de dados e um servidor de website local.

**Trello** – Plataforma utilizada para organização do meu jogo.

**FileZilla** – Software utilizado para conseguir aceder ao servidor website do domínio que me foi dado pela escola, assim como para mover/editar/criar ficheiros no mesmo.

**PHPMailer** – Código aberto usado para o envio de emails utilizando código.

**Gmail** – Utilizado, junto com o **PHPMailer** para realizar o envio de emails do website.

**Excel** – Utilzado para criar uma simulação do que seria o sistema de rondas do jogo.

**CONCLUSÃO**

Felizmente consegui concluir o projeto na totalidade, dentro do que tinha planeado apesar das várias dificuldades em resolver problemas durante a realização do projeto.

A maior dificuldade que tive foi a aprender a utilizar o **Unity** ao mesmo tempo que aprendia a linguagem de programação **C#**, utilizada para a criação dos *scripts*, para conseguir desenvolver o jogo. Isto deve-se à quantidade de scripts complexos e dúvidas que fui tendo ao longo do projeto.

Realizar esta prova ajudou-me imenso a aumentar e melhorar as minhas capacidades e conhecimento nestas áreas de programação.

Estou satisfeito com o resultado final e penso que seja um bom início para um projeto que possa ser integrado no mundo e mercado atual.

Bibliografia

**Ficheiro Excel do sistema de rondas** - <https://drive.google.com/file/d/1XzTolNVH2-uFEaMoLPyZwfazRekCYtde/view?usp=sharing>

**Ficheiro word sobre a interface do Unity, criado por mim para um estágio de 80h** - <https://drive.google.com/file/d/1czvh0H90zv1PJ4T9zL6_dThETsmQn5Lq/view?usp=sharing>

**Trello** - <https://trello.com/b/wWF1RA7V/pap>

**Discord oficial do Unity** - <https://discord.com/invite/unity>

**Unity docs** - <https://docs.unity3d.com/Manual/index.html>

**Brackeys** (YouTuber que faz vídeo-tutoriais sobre o **Unity**) <https://www.youtube.com/channel/UCYbK_tjZ2OrIZFBvU6CCMiA>

**Audacity** – <https://www.audacityteam.org/>

**XAMPP** – <https://www.apachefriends.org/index.html>

**Excel** - <https://www.office.com/>

**PHPMailer** – <https://github.com/PHPMailer/PHPMailer>

**FileZilla** - <https://filezilla-project.org/>

**Photoshop** - <https://www.adobe.com/pt/products/photoshop.html>

**Visual Studio 2019** - <https://visualstudio.microsoft.com/>

**Visual Studio Code** – <https://code.visualstudio.com/>

**Aseprite** - <https://www.aseprite.org/>

**IMPORTANTE: Todos os links acima foram visitados e revisitados inúmeras vezes durante todo o tempo da realização do projeto.**