

# River's Edge Chronicles

 [Informatii despre proiect](#)

 [Descrierea Jocului](#)

 [Specificatii Tehnice](#)

 [Specificătii Tematice](#)

 [Acțiuni Disponibile și Control](#)

 [Unelte de Inteligență Artificială din Unreal Engine](#)

 [Clasele Proprii](#)

 [Lista de taskuri realizate](#)

[Etapa 0 - alegere temă și descriere](#)

[Etapa 1 - scenă simplă - plasarea terenului, skyboxului și a altor elemente statice simple](#)

[Cerințe teren](#)

[Cerințe materiale](#)

[Etapa 2 - pion/caracter, sistemul de damage](#)

[Cerințe strict pentru pion](#)

[Cerințe strict pentru caracter](#)

[Cerințe comune pentru pion/caracter](#)

[Sistemul de damage](#)

[Etapa 3 - Traекторii. Projecțile](#)

[Traекторii](#)

[Proiecțile](#)

[Etapa 4 - scenă avansată - arhitectura jocului, a nivelului. Lumini.](#)

[Arhitectura scenelor \(hărților\)](#)

[Nivelele jocului \(hărți\)](#)

[Lumini și umbre](#)

[Aleator](#)

[Etapa 5 - Evenimente. Coliziuni. Manipularea timpului](#)

[De mouse](#)

[De tastatură](#)

[Coliziuni](#)

[Manipularea timpului](#)

[Etapa 6 - Meniul. Adăugarea de sunete](#)

[Meniul jocului](#)

[Sunete](#)

[Etapa 7 - salvarea jocului](#)

[Salvarea și reîncărcarea jocului](#)

[Etapa 8 - C++. Design patterns. Programare orientată pe obiecte.](#)

[Etapa 9 - Efecte vizuale: sisteme de particule, efecte de postprocesare.](#)

[Etapa 10 - documentația](#)

[Alte categorii de punctaj](#)

[Algoritmică](#)

[Aspect](#)

[Organizare cod](#)

[Modelare](#)

[Sunete](#)

[Pachete externe](#)

 [Listă de Utilitare Folosite](#)

 [Lista de Pachete Externe Folosite](#)

 [Bibliografie](#)

[Etapa 1](#)

[Etapa 2](#)

[Etapa 6](#)

[Etapa 7](#)

 [Total Puncte](#)



## Informatii despre proiect

Nume: Georgescu Miruna-Bianca

Grupa: 332

Numele proiectului: River's Edge Chronicles

Curs: Dezvoltarea Jocurilor cu Unreal Engine 5

Link catre repository: <https://github.com/MirunaGeorgescu/rivers-edge-chronicles>



## Descrierea Jocului

Jocul se desfășoară într-o lume captivantă, în centrul căreia se află un sat pitoresc, înconjurat de un râu misterios, cu un impunător castel în mijloc. Scopul jocului este de a explora această hartă complexă, interacționând cu diverse elemente și personaje.



Unul dintre aspectele distinctive ale jocului este prezența unui gardian care apără castelul. Gardianul reprezintă o amenințare, iar jucătorul trebuie să își croiască drum prin sat și să găsească strategii pentru a-l evita sau a-l înfrunta vitejeste. O componentă importantă a jocului este implementarea sistemului de luptă, unde jucătorul poate ataca gardianul și, la rândul său, poate fi atacat. Pentru a adăuga un element de tensiune, gardianul poate afecta starea de sănătate a jucătorului, iar widget-urile afișate pe ecran îi informează pe jucători cu privire la nivelul lor de viață.



Jocul beneficiază de un sistem de checkpoint-uri, astfel încât progresul să fie salvat, oferind jucătorului satisfacția de a avansa în aventura sa.



Un element intrigant al jocului constă în existența unor materiale speciale care pot fi descoperite în timpul explorării. Aceste elemente adaugă o dimensiune suplimentară la povestea jocului.

Pentru a intensifica experiența, jocul implementează un ciclu zi-noapte realist, oferind astfel variații în atmosferă și provocări suplimentare. Această

caracteristică contribuie la realismul și imersiunea în lumea creată.



În cadrul jocului, este integrat un meniu principal care oferă jucătorului acces rapid la diverse opțiuni. Acest meniu principal include funcționalități inceperea jocului sau accesarea opțiunilor. Un meniu de opțiuni dedicat aduce o dimensiune personalizată experienței de joc.



Jucătorul are posibilitatea de a pune jocul pe pauză în orice moment, asigurându-se astfel că poate gestiona confortabil experiența de joc. Un meniu de opțiuni dedicat aduce o dimensiune personalizată experienței de joc.



## Specificatii Tehnice

- **Sisteme de Operare:** Jocul a fost dezvoltat pentru sistemele de operare Windows 11 Home, versiunea 23H2 (Build 22631.3007), cu suport pentru arhitectura de 64 de biți.
- **Spațiu pe Disc:** Dimensiunea totală a jocului este de aproximativ 6.39 GB (6,868,211,543 bytes), ocupând 6.48 GB (6,965,514,240 bytes) pe disc.
- **Procesor:** Dezvoltat pentru a funcționa eficient pe procesoare de generația a 13-a Intel Core, cu specificații minime de 2.20 GHz (exemplu: Intel(R) Core(TM) i7-1360P).
- **Versiune Unreal Engine:** 5.3.2.
- **Conexiune la Rețea:** Jocul nu necesită conexiune la rețea pentru a fi jucat.
- **Fisier de Salvare:**
  - Dimensiunea fișierului de salvare este de aproximativ 2.17 KB (2,225 bytes).
  - Ocupă 4.00 KB (4,096 bytes) pe disc.



## Specificații Tematice

- **Categorie:** Joc de aventură.
- **Motivația Alegerii Temei Jocului:** Tema jocului, River's Edge Chronicles, a fost aleasă pentru versatilitatea pe care o oferă. Atmosfera medievală, legenda antică și elementele de mister și magie creează o experiență captivantă pentru jucători.
- **Listă de Taguri Descriptive:**
  - Explorare
  - Poveste

- Mister
- Magie
- Legendă antică
- Medieval
- **Tipul Jocului:** Singleplayer (un singur jucător).
- **Timpul de Joc:** Nelimitat. Jucătorii pot explora lumea jocului și pot progrăsă în aventura lor fără restricții de timp. Aceasta oferă o experiență mai relaxată și personalizată.
- **Limba în Care Jocul Este Disponibil:** Engleză. Jocul este dezvoltat și optimizat pentru a fi jucat în limba engleză.



## Actiuni Disponibile și Control

- **Deplasare în față și înapoi:** W pentru înainte, S pentru înapoi.
- **Deplasare laterală:** A pentru mișcare la stânga, D pentru mișcare la dreapta.
- **Sărit:** Bara de spațiu (Space).
- **Schimbare culoare emisivă sac:** Apăsare continuă pe tasta Enter.
- **Pauză:** Tasta P.
- **Rotirea camerei:** Folosirea mouse-ului pentru a direcționa camera în jurul personajului.
- **Schimbare perspectivă:** Tasta C pentru a trece între modul third-person și first-person.
- **Atac:** Click stânga pe mouse pentru a efectua acțiunea de atac.



## Unelte de Inteligență Artificială din Unreal Engine

### AIMoveTo

- **Descrierea functiei**

- AIMoveTo este o funcție din sistemul de inteligență artificială (AI) în Unreal Engine.
- Această funcție permite agenților AI să calculeze și să execute un traseu către o destinație specificată.
- **Mod de Aplicare în Joc**
  - **Scenariu în Joc:**
    - Gardianul este o entitate AI responsabilă cu apărarea unei anumite zone.
    - Atunci când jucătorul se apropie, AIMoveTo este utilizată pentru a ghida gardianul către poziția actuală a jucătorului.



## Clasele Proprii

### 1. Clasa Character Extinsă:

- **Rol:** Reprezintă jucătorul și gardianul în joc.
- **Proprietăți Importante:**
  - Informații despre starea și abilitățile jucătorului/gardianului.
  - Implementare a logicii de mișcare și atac.
- **Metode Implementate:**
  - Funcții pentru manipularea mișării, atacului și gestionarea stării de sănătate.

### 2. Clasa Widget Extinsă:

- **Rol:** Afisează starea de sănătate a jucătorului și gardianului în interfața utilizatorului.
- **Proprietăți Importante:**
  - Componente vizuale pentru bară de sănătate și alte informații relevante.
- **Metode Implementate:**
  - Funcții pentru actualizarea vizuală a barei de sănătate.

### 3. Clase pentru Meniuri (Principal, Optiuni, Game Over, Pauză):

- **Rol:** Gestionarea diferitelor ecrane de meniu în joc.
- **Proprietăți Importante:**
  - Elemente de interfață vizuală pentru fiecare meniu.

### 4. Clasa GamemodeBase Extinsă:

- **Rol:** Gestionarea fluxului general al jocului și inițializarea elementelor de bază.
- **Proprietăți Importante:**
  - Configurări pentru inițializarea și gestionarea jocului.
- **Metode Implementate:**
  - Funcții pentru gestionarea tranzitilor între stările de joc.

### 5. Clasa Actor Extinsă pentru Checkpoint:

- **Rol:** Marchează locațiile de checkpoint în joc.
- **Proprietăți Importante:**
  - Informații despre locația checkpoint-ului.
- **Metode Implementate:**
  - Funcții pentru activarea checkpoint-ului și restaurarea stării jocului la checkpoint.

### 6. Clasa SaveGame Extinsă:

- **Rol:** Gestionarea salvărilor jocului.
- **Proprietăți Importante:**
  - Informații despre starea jocului care trebuie salvate.
- **Metode Implementate:**
  - Funcții pentru salvarea și încărcarea stării jocului.



## Listă de taskuri realizate

## Etapa 0 - alegere temă și descriere

1. (0.02) Alegerea temei (găsirea unui titlu pentru joc)
2. Descrierea succintă a temei (google docs, 1/2-1 pagina). Va contine următoarele informații:
  - a. (0.13) o descriere generală a jocului (0.13 împărțit în):
    - (0.05) Contextul (cadrul, povestea jocului; despre ce este vorba)
    - (0.05) Contextul (cadrul, povestea jocului; despre ce este vorba)
    - (0.05) Regulile de joc: care sunt acțiunile posibile ale jucătorului. În ce situații se termină jocul. Cum se decide dacă a câștigat sau pierdut, cum se calculează eventualul scor.
  - b. (0.1) Stabilirea interacțiunii cu utilizatorul. Poate fi dată ca descriere în cuvinte sau diagramă UML (doar pentru acțiunile principale, nu tot jocul). Aceasta trebuie să cuprindă scenarii de utilizare. Care e prima interfață pe care o vede utilizatorul. Ce acțiuni disponibile are când intră în aplicație etc.
  - c. (0.02) Identificarea categoriei de utilizatori (vârstă, personalitate, interese, cadru social). Exemplu: jocul este dedicat vegetarienilor de toate vîrstele, cu nivel mediu spre ridicat de cultură generală, pasionați de gătit, strategie și ecologie
  - d. (0.03) Stabilirea cuvintelor/sintagmelor cheie (o listă cu minim 5-6 cuvinte cheie care descriu jocul). Motiv: unele platforme pentru publicarea jocurilor cer astfel de sintagme cheie (keywords, tags etc.)
  - e. (0.2) Căutarea unor jocuri similare (4-5 jocuri) ca temă pentru a observa lucrurile pro și contra (minim 2 aspecte pozitive și minim 2 aspecte negative pentru fiecare joc). Jocurile pot fi evaluate fi pe baza unui demo, a vizionării unor streaming-uri sau videoclipuri gratuite care arată desfășurarea jocului, și chiar cu ajutorul review-urilor. Nu este necesară descărcarea/instalarea jocurilor. Pentru fiecare joc veți scrie numele urmat de observații. De exemplu dacă doriți să faceți un joc de simulare a unui sat veți enumera jocuri de simulare pentru un fel de comunitate (sat, oraș, trib etc.) Jocurile găsite nu trebuie să fie identice ca idee cu jocul vostru ci doar să aibă niște caracteristici similare.

**Total: 0.5 puncte**

# Etapa 1 - scenă simplă - plasarea terenului, skyboxului și a altor elemente statice simple

## Cerințe teren

- (0.05) În scenă trebuie să existe un teren. Nu este obligatorie deplasarea pe teren, poate servi drept peisaj în jurul platformei de joc. 
- (0.15) Terenul trebuie să aibă un relief variat(să existe multiple zone joase și înalte). Terenul va avea alocat un material ce cuprinde multiple (minim 3) texturi (de exemplu, textură de iarbă, de nisip, de rocă etc). Texturile asociate trebuie pictate pe teren astfel încât să fie în concordanță cu forma terenului (de exemplu o groapă adâncă va avea textură de rocă și nu cu iarbă/floricele) 
- (0.05) Pe teren trebuie să existe minim o rampă (meniul Sculpt→ ramp) 
- (0.05) Pe teren trebuie să existe două zone simetrice (de exemplu doi munți). (vezi meniul Sculpt→ mirror)

## Cerințe materiale

- (0.05) Obiect cu material transparent 
- (0.05) Obiect cu luciu metalic care reflectă mediul înconjurător 
- (0.1) Obiect cu material lucios(care reflectă mediul) pe anumite zone și nelucios pe altele în funcție de un anumit pattern (rezolvarea se va face prin blueprints) 
- (0.05) Existența unui obiect cu culoare emisivă 
- (0.05) Obiect care are un material cu gradient în 2 culori (gradientul se va realiza prin blueprints fără a folosi o textură externă) 
- (0.05) Obiect care are transparență în gradient 
- (0.15) Obiect care are un material cu gradient în 3 culori. Gradientul realizat prin blueprints, fără utilizarea unei texturi externe 
- (0.1) Obiect cu material transparent și opac pe portiuni. Portiunile de transparență decide printr-un pattern (exemplu: altă textură) 

- (0.1) Combinarea culorilor a două texturi în funcție de un pattern dat de o a treia textură 
- (0.2) Simularea culorii cu sclipici (puncte sclipitoare dispuse aleator). Folosind un nod de zgomot, fără textură externă 
- (0.15) Material complex cu pattern creat prin minim 5 funcții matematice 
- (0.05) Folosirea unui normal map pentru a crea senzația de asperități pe un obiect, fără modificarea vertecșilor 

**Total: 1.4 puncte**

## **Etapa 2 - pion/caracter, sistemul de damage**

### **Cerințe strict pentru pion**

(0.1) Realizare pion prin extinderea clasei Pawn sau DefaultPawn

### **Cerințe strict pentru caracter**

(0.1) Realizare caracter prin extinderea clasei Character 

(0.2-0.5) Crearea unei/unor animații pentru caracter care să fie folosite în joc. Punctajul se dă în funcție de cât de complexe sunt animațiile, contextul în care sunt folosite

(0.1) Clasa pentru pionul/caracterul din scenă e făcută în C++

(0.1-0.3) Caracterul trebuie să aibă acțiuni (Action Mappings) definite în inputs din Project Settings și implementate în blueprint (exemple de acțiuni: jump, crouch, fly etc.) 

(0.1) Asocierea animațiilor pentru acțiunile de mai sus (minim o asociere) 

### **Cerințe comune pentru pion/caracter**

(0.05) Pionul/caracterul va avea o cameră (de înregistrare) adăugată în components pentru a urmări pionul în stil first person sau third person. 

(0.15) Posibilitatea de a schimba din urmărire first person în third person prin apăsarea unei taste. 

(0-0.4) Crearea unor variabile pentru pion/caracter sau alți actori, care să reflecte starea jucătorului, anumite proprietăți. Punctajul se dă doar dacă informațiile memorate sunt relevante pentru salvarea jocului:

- Intreg (integer sau Integer64)
- Boolean 
- Rațional (Float) 
- Sir de caractere (String sau Text)
- Vector (de exemplu, pentru memorarea unei culori, locații etc.)
- Tablou de date (Array) de orice tip
- Mulțime (Set) de orice tip
- Hartă (Map) de orice tip

(0.1) Pionul/caracterul trebuie să aibă mișările pe axe (Axis Mappings) definite în inputs din Project Settings.

- (0.1) Se adună la punctaj dacă se poate translata pe minim 2 axe definite astfel. 
- (0.1) Se adună la punctaj dacă se poate roti față de măcar o axă. 
- (0.05) Se adună la punctaj dacă măcar un mapping este făcut pentru mouse. 

(0.1) Pionul/caracterul își poate schimba(mări/micșora) viteza de deplasare. 

(0.1) Se va trata coliziunea pionului/caracterului cu alte obiecte, folosind un box de coliziune. Pionul/caracterul va putea fi capabil să treacă prin anumite obiecte dar nu prin altele (în minim una dintre aceste situații, se vor schimba unul sau mai multe attribute ale pionului/caracterului: de exemplu îi scade sănătatea dacă atinge un inamic).

(0.05-0.1) Un sistem de calculare a scorului. În funcție de realizările în joc se va calcula un număr care să arate cât de bine s-a descurcat jucătorul.

(0.35) Sistem de highscore. Pentru jocuri cu finalitate, după terminarea jocului, scorul se va salva într-un fișier, cu scorurile tuturor utilizatorilor, ordonate de la cel mai bun la cel mai slab (eventual numărul de scoruri memorate poate fi maxim N,

și orice performanță sub primele N nu va fi salvată). Utilizatorul va avea opțiunea de a vedea scorurile.

## Sistemul de damage

(0.25) Se va implementa sistemul implicit de damage din Unreal fie asupra pionului/caracterului fie asupra actorilor cu care interacționează jucătorul. Se va folosi metoda `ApplyDamage` în urma unui eveniment din joc. Cu ajutorul unui eveniment `AnyDamage` actorul asupra căruia se aplică distrugerea va avea niste parametri afectați. Se va implementa un caz pentru o distrugere cu valoare mică (obiectul își poate schimba culoarea, se poate micșora etc) și un altul pentru o distrugere cu valoare mare (de exemplu obiectul poate să dispară sau să își schimbe culoarea în mod diferit față de damage-ul mic, sau să oferim un mesaj scris pe ecran). 

**Total: 1.2 puncte**

## Etapa 3 - Traекторii. Proiectile

### Traекторii

(0.2) Traекторii statice. Vor exista în scenă traекторii definite prin curbe spline create static cu ajutorul editorului. Pe traекторie se vor deplasa actori

(0.1) Viteză ce poate fi accelerată/decelerată pe întreaga traectorie

(0.25) Viteză accelerată/decelerată doar pe anumite porțiuni ale traectoriei (predeterminate sau calculate prin program)

(0.05) Oprirea deplasării pe traectorie

(0.05) Repornirea deplasării pe traectorie din punctul în care s-a pornit obiectul

(0.05) Reînceperea plimbării pe traectorie din punctul inițial

(0.05) Plimbare infinită pe o traectorie.

(0.1) Realizarea traseului pe traectorie de un număr finit de ori (N) după care se întâmplă o anumită acțiune în joc

(0.3 + bonus 0.1) Generarea unei curbe spline în mod dinamic, prin program în blueprint. Dacă se face prin C++ se vor primi 0.1 puncte bonus

- (0.2) Schimbarea în mod dinamic a traectoriei în urma unui eveniment.
- (0.1) Existența mai multor obiecte pe aceeași traectorie
- (0.1) Adăugarea dinamică (în urma unui eveniment sau o anumită stare a jocului) a unor obiecte suplimentare pe traectorie.
- (0.1) Schimbarea direcției de mers pe traectorie, prin program.

## Proiectile

- (0.2) Implementarea unui actor special cu rol de proiectil. Acesta va fi lansat pe o traectorie în urma unui eveniment. Proiectul va porni din spate un actor (poate fi și pion/caracter) cu scopul de a ajunge la anumite coordonate. Proiectilul dispare la atingerea unei ținte (un alt actor)
- (0.2-0.4) Proiectul poate urmări o țintă mobilă (își ajustează traectoria în funcție de coordonatele țintei). Punctajul depinde de tipul de urmărire și naturalețea traectoriei
- (0.1-0.5) Aplicarea fizicii asupra proiectilului: proiectil afectat de forța de gravitație, efecte de vânt, precipitații, forță de frecare diferită în medii diferite

**Total: 0 puncte**

## Etapa 4 - scenă avansată - arhitectura jocului, a nivelului. Lumini.

### Arhitectura scenelor (hărților)

(0.1-0.5) se dă pentru complexitatea construcției scenei (numărul de elemente, modul de așezare, construcții create prin așezarea unor forme elementare pentru a obține forme mai complexe). Folosirea modului Foliage pentru realizarea anumitor zone. 

(0.1-0.5) Se dă pentru generarea prin program a actorilor cu anumite locații, rotații, dimensiuni în scopul de a crea construcții complexe (exemplu: o tablă de șah formată din cubulete, un labirint, o casă formată din obiecte de tip perete și acoperiș care au fost plasate prin blueprint pentru a obține aspectul de casă). Generarea actorilor în scenă se va face în blueprint cu metode precum Spawn

Actor from Class. Minim o caracteristică a actorilor va fi calculată prin blueprint (de exemplu, locația, rotația)

## Nivelele jocului (hărți)

- (0.2) Jocul este multilevel cu hărți diferite. Se trece de la un nivel la altul în urma unor realizări în joc.
- (0.1-0.5 per nivel; max 2p pt 4 nivele) se dă până la maxim 0.5 pentru fiecare nivel suplimentar, până la un maxim de 4 nivele (primul nivel este punctat în alte categorii de punctaj) în funcție de complexitatea arhitecturii acestuia (din punct de vedere al terenului, skybox-ului (sau skysphere), luminilor, obiectelor, așezate pe hartă static (cu ajutorul editorului) sau în mod dinamic (prin program) skybox/skysphere, elemente atmosferice etc.
- (0.2) Folosirea de subnivele pentru optimizarea hărții

## Lumini și umbre

(0.05-0.1) Folosirea relevantă a minim unei surse direcționale de lumină (Directional Light). Modificarea (statică, manuală a) proprietăților acesteia.

(0.05-0.1) Folosirea relevantă a minim unei surse punctiforme de lumină (Point Light). Modificarea (statică, manuală a) proprietăților acesteia.

(0.05-0.1) Folosirea relevantă a minim unei surse spot de lumină (Spot Light). Modificarea (statică, manuală a) proprietăților acesteia.

(0.05-0.1) Folosirea relevantă a minim unei surse dreptunghiulare de lumină (Rect Light). Modificarea (statică, manuală a) proprietăților acesteia.

(0.05-0.1) Folosirea relevantă a minim unei surse atmosferice de lumină (Sky Light). Modificarea (statică, manuală a) proprietăților acesteia.

(0.05-0.1) Asocierea luminilor (sub formă de componente) unor actori (exemplu: crearea unei lanterne, asociind mesh-ului de lanternă un Spot Light)

(0.1-0.5) Modificarea caracteristicilor luminilor (precum culoare/intensitate, faptul că e stinsă/aprinsă) în funcție de evenimente/starea jucătorului/timpul din joc. Se punctează în funcție de numărul de lumini afectate, numărul de tipuri diferite de modificări și complexitatea acestora. Căteva exemple (ca să vă faceți o idee, dar puteți veni cu ceva nou):

- dacă jocul simulează succesiunea zi/noapte, SkyLight poate varia 
- Sau avem unele lumini care sunt aprinse pentru 5 secunde și stinse pentru 2 secunde, apoi iar aprinse și tot aşa)
- lumină care se aprinde când intrăm într-o cameră

(0.1-0.5) Animarea luminilor prin schimbarea direcției pozitiei, distanței de atenuare, în mod treptat și continuu. Se punctează în funcție de numărul de lumini afectate, numărul de tipuri diferite de animații și complexitatea acestora. Căteva exemple (ca să vă faceți o idee, dar puteți veni cu ceva nou):

- O sursă de lumină punctiformă care se plimbă pe o traекторie
- Simularea unei stele a cărei stralucire variază periodic în timp mergând treptat de la o lumină intensă la una slabă și tot aşa.
- O sursă spot care se rotește, ca un reflector automat
- Sursă de lumină direcțională care își schimbă treptat direcția

(0.05) Sursă de lumină care nu proiectează umbre 

(0.05) Obiect (actor) care nu lasă umbre desi alte obiecte luminate de aceeași sursă lasă umbre. 

(0.1) Obiect care nu e afectat de lumină (material de tip Unlit) 

## Aleator

Folosirea unor numere aleatoare în:

- (0.05) generarea unei culori
- (0.05) coordonate, rotații și/sau dimensiuni aleatoare pentru unul sau mai multe obiecte sau pion/caracter
- (0.1) string aleator - de exemplu pentru o parolă sau parte din username-ul implicit, ori pentru salvarea jocului
- 0.1) Amestecarea aleatoare a elementelor unui vector folosit apoi în joc
- (0.2-0.3) Comportamente determinate probabilist (se dă 0.2 pentru 2 probabilități complementare și 0.3 pentru mai multe). Exemplu: cu o

probabilitate de 20% să se genereze elemente de culoare c1, cu o probabilitate de 30% culoare c2 și restul de culoare c3. Se poate alege orice element care să depindă de probabilitate (culoare, locație, formă, tipul de obiect, acțiune desfășurată etc.) 

**Total: 1.7 puncte**

## **Etapa 5 - Evenimente. Coliziuni. Manipularea timpului**

### **De mouse**

- (0.1) Folosirea relevantă a unui eveniment de click în cadrul jocului 
- (0.1) Folosirea relevantă a unui eveniment de begin cursor over în cadrul jocului
- (0.1) Folosirea relevantă a unui eveniment de end cursor over în cadrul jocului

### **De tastatură**

- (0.1) Folosirea relevantă a unui eveniment de keydown (tastă apăsată) în cadrul jocului 
- (0.1) Folosirea relevantă a unui eveniment de keyup (tastă eliberată) în cadrul jocului 
- (0.1) Tratarea unei combinații de taste (dintre o tastă specială - shift, ctrl, alt - și una afișabilă, de exemplu Shift+q, ctrl+w etc.)

### **Coliziuni**

- (0.1) Folosirea relevantă a unui eveniment de overlap în cadrul jocului 
- (0.1) Folosirea relevantă a unui eveniment de hit în cadrul jocului 
- (0.1-0.4) Actualizarea datelor pionului/characterului și/sau actor la coliziune (hit/overlap) Se punctează în funcție de complexitatea tratării coliziunii, De exemplu, dacă un actor(poate fi chiar pionul) se află în coliziune cu diferiți actori (su diferite tipuri de actori) să se întâmpile acțiuni diferite (de exemplu, la coliziunea cu o bară de energie, bara dispare și pionul câștigă sănătate, dar la coliziunea cu un inamic, inamicul doar își schimbă culoarea iar pionul pierde sănătate. 

### **Manipularea timpului**

(0.1) Realizarea unei acțiuni la un interval de timp t după ce s-a întâmplat un eveniment (De exemplu, la 2 secunde după ce porneste jocul, se întâmplă ceva), de exemplu cu un nod de tip "Set Timer by Function Name"

(0.1) Repetarea apelului unei funcții la intervale de timp egale, cu un nod de tip "Set Timer by Function Name"

(0.1) În urma unui eveniment sau a unei stări atinse de joc, o funcție apelată repetitiv (la intervale de timp egale) cu ajutorul lui "Set Timer by Function Name", va fi pusă în așteptare cu "Pause Timer by Function Name". Apelarea repetitivă va fi reluată în urma altui eveniment, cu "Unpause Timer by Function Name"

(0.05) În urma unui eveniment sau a unei stări atinse de joc, o funcție apelată repetitiv (la intervale de timp egale) cu ajutorul lui "Set Timer by Function Name", va fi anulată (apelurile repetate vor fi opriate definitiv) cu "Clear Timer by Function Name".

(0.2) Afisarea datei (de exemplu, într-un widget) folosind nodul now și spargând structura DateTime pe componente. Data se va afișa în format zi/lună/an (iar dacă un număr e sub 10, va fi precedat de cifra 0)

(0.2) Afisarea pe ecran, pe parcursul jocului, a timpului care s-a scurs de la începutul jocului sau de la începutul sesiunii, sau de la un anumit eveniment încolo.

(0.3) Pentru o informație de timp (câte secunde mai durează până la un eveniment sau cate secunde au trecut de la un moment t, timpul în loc să se afișeze ca un număr întreg de secunde se va afișa în formatul hh:mm:ss (h - oră, m - minute, s - secunde) . Dacă vreun număr din cele 3 categorii este sub 10, se va afișa precedat de un 0).

(0.25) Afisarea într-un widget a timpului ultimei accesări sau a intervalului de timp care s-a scurs de la ultima accesare.

**Total: 0.6 puncte**

## **Etapa 6 - Meniul. Adăugarea de sunete**

### **Meniul jocului**

(0.2p) La intrarea în joc se va afișa meniul principal al jocului. Jocul nu este pornit (de exemplu, este în pauză) până nu se ieșe din meniu. 

(0.05) Folosirea unei imagini într-un panou 

(0.05) Folosirea panourilor de tip HorizontalBox și/sau VerticalBox 

(0.2p) Trecerea printre ecranele meniului folosind WidgetSpinner

(0.2-0.4p) Crearea unei clase custom pentru butoane. (se punctează în funcție de căt de complexă este.)

Meniul principal poate conține următoarele butoane:

- (0.2p) Butonul de pornire a unui joc nou. Butonul va avea un text sugestiv, de exemplu "Start". La intrarea în aplicație, jocul este în pauză, și rămâne așa până îl activează utilizatorul 
- (0.3p) Butonul de continuare a ultimului joc început. La click pe acest buton, pornește jocul afișând exact starea în care a fost lăsat de utilizator înainte de ultima închidere (sau ultima salvare) 
- (0.2p) Ecranul de setări generale (pentru profilul jucătorului sau caracteristicile unui joc nou). Ecranul de setări va fi accesat printr-un buton din meniul principal. Ecranul de setări va conține diverse inputuri și un buton de trimitere a datelor. La click pe buton setările se vor salva în proprietățile pionului/characterului. 
- Inputurile folosite în ecranul de setări (sau alte ecrane care cer informații de la utilizator pot fi de următoarele tipuri (atenție, se punctează pentru fiecare tip distinct, nu input în sine):
  - (0.05) EditText
  - (0.05) TextBox 
  - (0.1) Slider. Se vor seta parametri precum: valorile minime și maxime și pasul.
  - (0.1) SpinBox. Se vor seta parametri precum: valorile minime și maxime, numrul de cifre zecimale, pasul (delta), exponentul de creștere (creșterea pasului pentru valori mai mari)
  - (0.05) Checkbox

- (0.1) ComboBox cu minim 2 opțiuni
- (0.1) RadialSlider Se vor seta parametri precum: pasul, valoarea implicită etc.
- (0.1) Se dă suplimentar 0.1 pentru ComboBox dacă opțiunile sunt adăugate dinamic.
- Se adună separat 0.1 pentru fiecare tip de input care e inclus într-un widget custom în scopul adăugării unor funcționalități noi la completare (Exemplu: un borderbox care își schimbă culoarea de background la fiecare apăsare de tastă
- (0 - 0.3) Aspectul ecranului de setări. Se acordă puncte în funcție de:
  - Cel mai important: Alinierea elementelor (de exemplu cu un grid)
  - Faptul că fiecare input are etichetă asociată (text în dreptul lui care să spună rolul) și/sau tool/tip,
  - Schimbarea culorilor隐含的
  - Text lizibil (culori alese cu contrast bun; textul nu e prea transparent sau suprapus cu o imagine)
  - Folosirea unui background plăcut.
- (0.1) Adăugarea dinamică (prin program) a unor elemente în widget, folosind metode de tipul "Add Child to [container]"
- (0.1) Stergerea dinamică (prin program) a unor elemente în widget, folosind metode de tipul "Remove Child"
- (0.4p) Butonul de încărcare a unui joc vechi. La click pe acest buton se va deschide un ecran cu salvările anterioare ale utilizatorului din care acesta poate să aleagă ce joc dorește. Salvările pot fi listate prin butoane sau printr-un combobox. Ecranul va fi generat dinamic în funcție de fișierele din folderul de jocuri salvate. Identificarea jocurilor care corespund jucătorului curent se va face prin username.
- Butonul de afișare a informațiilor despre joc:
  - (0.1) va duce spre un ecran cu un text despre joc care explică povestea/contextul. Ecranul are un buton de revenire la meniul principal.

◦ (0-0.4) Stilizare specială a ecranului cu textul despre joc. Punctajul se dă în funcție de cât de complexă și frumoasă e stilizarea:

- folosirea unui scrollPane
- folosirea culorilor diferite în cadrul textului
- folosirea stilurilor diferite: bold/italic
- stilizarea textului sub formă de secțiuni cu titluri
- folosirea listelor
- folosirea imaginilor în cadrul textului

● (0.1p) Butonul de ieșire din aplicație (la click pe el se închide jocul) 

(0.1) Cu ajutorul unui widget se va crea un meniu afișat pe parcursul jocului care va avea butoanele (punctate suplimentar după cum urmează):

- (0.1) Pauza - la Click pe el, jocul intră în pauză, iar când dăm iar click pe el reîncepe. Textul butonului ar trebui să difere în tipul pauzei, de exemplu să scrie "Reîncepe" 
- (0.1) Un buton de ieșire din joc. 
- (0.2) Un buton/shortcut cu tastă care pune în pauză jocul și afișează meniul principal. În această situație meniul trebuie să aibă un buton suplimentar cu textul "Continua".

(0.1-0.3) Afișarea informațiilor legate de starea jucătorului în timpul jocului. Se punctează în funcție de cât de complexă e afișarea. 

(0.1) Folosirea unei bare de progres în afișarea informațiilor pentru jucător 

(0.2) Opțiunea de a ascunde și reafipa afișajul din timpul jocului, de exemplu, la apăsarea unei taste.

(0.2) Simularea unor radio buttons folosind butoane custom

(0.3) Simularea unor radio buttons folosind checkbox-uri custom

(0.2-0.5) Unul sau mai multe ecrane informative care apar în urma unui eveniment sau a unei stări în care ajunge jocul. De exemplu, un ecran în care jucătorul e informat că a intrat într-un nivel nou sau că a "murit". Se punctează în funcție de numarul lor și de complexitatea afișării. 

(0.1) Buton de restart într-un ecran informativ, pentru cazul în care jucătorul a murit sau s-a terminat nivelul 

(0.2-0.4) Loading screen - se punctează în funcție de complexitate

## Sunete

(0.1) Adăugarea unui sunet în cadrul jocului

Se punctează suplimentar:

(0.1) Sunetul a apărut în cadrul unui widget în urma unui eveniment (de exemplu, click pe buton)

(0.1) Sunetul a apărut în urma unei coliziuni

(0.1-0.2) Sunetul depinde de acțiunile jucătorului și de mediu: deplasarea prin nisip generează alt sunet decât cea prin băltoace)

**Total: 2.4 puncte**

## Etapa 7 - salvarea jocului

### Salvarea și reîncărcarea jocului

(0.1) Crearea unei clase derivate din SaveGame 

(0.2) Optiunea de salvare a jocului în timpul derulării jocului într-un fișier, fără trecerea prin meniul principal. De exemplu la o combinație de taste sau la click pe elemente speciale din joc. 

(0.2) Se oferă punctaj suplimentar dacă la salvarea jocului utilizatorul e întrebat printr-un widget, dacă dorește să salveze peste fișierul curent corespunzător sesiunii prezente de joc sau vrea un fișier nou caz în care poate opta pentru o parte din nume (de exemplu salvarea e de forma [username][timestamp][nume-dat-de-utilizator]

(0-0.4) Salvarea informațiilor relevante pentru repornirea jocului din punctul rămas și afișarea setărilor deja făcute de utilizator. (Fiecare tip diferit de date din cele enumerate 0.05). Punctajul se dă doar dacă informațiile memorate sunt relevante pentru salvarea jocului:

- Intreg (integer sau Integer64)

- Boolean 
- Rațional (Float) 
- Sir de caractere ( String sau Text)
- Vector (de exemplu, pentru memorarea unei culori, locații etc.)
- tablou de date (Array) de orice tip
- mulțime (Set) de orice tip
- hartă (Map) de orice tip

(0.2) Crearea unei funcții în Blueprint pentru citirea fișierului de salvare și setarea unor date în joc. 

(0.2) Opțiunea de AutoSave - se va salva jocul la fiecare interval de timp t.

(0.2) Salvarea la checkpoints. Dacă utilizatorul ajunge să realizeze ceva deosebit precum atingerea unei locații sau găsirea unui artefact sau înfrângerea unui inamic, jocul se salvează automat și la reintrarea în joc se va porni de la checkpoint-ul anterior. 

**Total: 0.8**

## **Etapa 8 - C++. Design patterns. Programare orientată pe obiecte.**

## **Etapa 9 - Efecte vizuale: sisteme de particule, efecte de postprocesare.**

## **Etapa 10 - documentația**

Va cuprinde următoarele părți/capitole:

1. (0.05) Prima pagină cu nume, prenume, grupă, data examenului, și titlu jocului. În josul paginii, numele materiei. Un cuprins către capitole (în caz că sunt mai multe pagini). Numele capitolelor cerute sunt cele scrise cu bold, culoare neagră. Paginile (dacă sunt mai multe) vor fi numerotate. 
2. (0.1) Descrierea detaliată a jocului (nu planul inițial ci doar ce ați reușit să implementați. Ce face jocul, care este povestea (această parte poate conține

și fragmente din etapa 0). Poate conține printscreens-uri din joc pentru clarificări 

3. (0.05) Specificații tehnice: sistemele de operare pentru care a fost dezvoltat jocul, cât ocupă pe disc, memoria aproximativă de care are nevoie, pachete adiționale care ar trebui instalate, necesită sau nu conexiune la rețea, ce fel de date salvează și aproximativ cât de mare poate ajunge un fișier de salvare, versiunea de Unreal Engine pe care a fost dezvoltat jocul. 
4. (0.05) Specificații tematice (necesare pentru eventuala publicare a jocului): categoria/categoriile jocului, motivația alegerii temei jocului, o listă de taguri descriptive, jocul este single sau multiplayer, timpul de joc (estimativ în cate ore poate fi terminat jocul; pentru jocurile nelimitate se va preciza că timpul de joc poate fi oricât de mare), limbile în care jocul este disponibil. 
5. (0.1) Acțiuni disponibile. Control. Ce acțiuni avem disponibile în joc și cu ce dispozitive periferice de intrare le putem realiza (mouse, tastatură, joystick etc.), cu ce combinație de taste și butoane. Ce shortcut-uri există. Ce opțiuni are utilizatorul de a-și defini propriile combinații de evenimente de mouse/tastatură pentru diverse acțiuni 
6. (0.1) Clasele proprii. Descrierea claselor create în cadrul jocului: rolul lor, proprietățile importante, ce metode implementează, cum sunt integrate în joc. Ce design patterns au fost implementate (și descrierea succintă a fiecărui design pattern). 
7. (0.1-0.2) Algoritmii. Descrierea algoritmilor utilizați (de exemplu, algoritmi pentru generarea unei structuri, algoritmi speciali pentru calcularea punctajului, algoritmii de inteligență artificială care controlează pioni/caractere din joc. Enumerarea elementelor implementate în C++. În explicații se vor insera capturi de ecran cu blueprints sau secevențe din codul C++. Analiza complexității și eficienței algoritmilor atât din punct de vedere al timpului cât și al memoriei. Punctajul depinde de numărul de algoritmi, complexitatea și modul de analiză.
8. (0.05-0.1) Unelte de inteligență artificială din Unreal. Descrierea uneltelelor de inteligență artificială (din cadrul Unreal) folosite și modul de aplicare în joc. 
9. (0.1) Lista de taskuri realizate. Obligatoriu pentru prezentare. Aici vor fi cerințele preluate cu copy-paste din barem, pe care studentul le-a realizat.

Lista de cerințe va fi organizată pe categorii și subcategorii exact ca prezentul barem. În dreptul fiecarui task va fi scris și punctajul din barem, iar studentul va face o sumă estimativă a punctelor sub fiecare categorie dar și la final, pentru verificare (baremul e mare și complex și e ușor să se uite ceva la prezentare, cu această masură putem verifica dacă nu am sărit nimic). 

10. (0.05-0.1) Lista de utilitare folosite. Veți enumera toate utilitare folosite pentru a crea active (modele, texturi, sunete etc.) pentru joc. Veți enumera activele și veți explica pe scurt cum le-ați realizat. 
11. (0.05) Lista de pachete externe folosite. Veți enumera toate pachetele adăugate în joc, toate activele preluate din alte surse (imagini, și în ce context le-ați folosit. Veți oferi și câte un link către pachetul respectiv. 
12. (0.05-1) Bibliografie. Veți lista cartile, tutorialele (scrise sau video), forumurile de unde ați preluat idei, bucăți de cod/blueprint. Dacă sursa este online, veți scrie și linkul. Pentru fiecare sursă veți preciza ce anume ați preluat de acolo. 

**Total: 0.75 puncte**

## Alte categorii de punctaj

### Algoritmică

Un algoritm se punctează în funcție de:

- relevanța în joc
- eficiența în timp și spațiu
- structurile de date speciale folosite (vectori, matrici, grafuri, rețele neurale)
- formulele matematice implicate
- interacțiunea cu mediul din joc, cu starea jucătorului
- numărul de sarcini pe care le îndeplinește
- scalabilitate
- Observație: în general cei care au puzzle games sau jocuri de strategie vor avea spre maxim aici. Jocul de la curs de exemplu ar primi cam 1-1.5 datorită

labirintului și a inamicilor. Un shooter simplu, doar cu calcularea sănătății, mici upgrade-uri etc, ia pe la 0.5-0.7. Un joc unde nu se prea întâmplă nimic (a încercat și omul ceva ca să nu zică că nu a deschis Unreal în viața lui) ia 0.1

## Aspect

Se va puncta luând în considerare următoarele

- Așezarea obiectelor în scenă în mod logic și ordonat (de exemplu mai mulți copaci grupați pe o zonă dăteren să formeze o pădurice și nu suspendați în aer sau pe capul vreunui personaj - decât dacă povestea jocului impune să fie acolo. O să faceți toți copaci zburători în joc acum, nu?)
- Terenul are o formă naturală, logică, îngrijită, plăcută. De exemplu nu e doar un teren plan cu trei gropițe puse unde s-a nimerit doar ca să fie).
- Sky Sphere are o textură relevantă (nu avem nori și soare pentru un joc care se petrece în cosmos)
- Folosirea unor culori potrivite (care să nu deranjeze ochiul) și a unor texturi frumoase. Texturile pot fi luate de pe internet cât timp nu se încalcă drepturile de autor și sursa e trecută în documentație.
- Materialele se potrivesc cu mesh-ul (de exemplu un obiect despre care știm că e metalic va avea luciu metalic).
- Modele sunt potrivite ca formă și dimensiune.
- Arhitectura scenei și cromatică se potrivesc cu tema jocului și categoria de utilizatori vizată.
- Iluminarea scenei este potrivită (de exemplu dacă suntem într-o încăpere în care căutăm ceva, să avem o sursă de lumină să distingem obiectele (decât dacă povestea cere asta)
- Widget-urile afișate sunt ușor de utilizat. Textele nu se suprapun. Contrastul între culoarea textului și a fundalului este potrivit (se poate citi ușor).
- Animatiile sunt potrivite, cu tranzitii line, nu sunt prea multe ca să facă scena obosită.

**Total: 1 punct**

## Organizare cod

Se poate da pentru:

- idei foarte interesante de joc
- implementări deosebite
- resurse (create de student) deosebite, interesante

## Modelare

(0.1-0.3 per model; maxim 0.9 pt 3 modele) Se punctează în funcție de complexitate modelele modelate de către student în Blender (sau alt utilitar de modelare). Se punctează până la maxim 3 modele diferite (Deci prezentați cele mai complexe modele; dacă diferențele între modele sunt mici, se consideră grad mic de complexitate). Studentul trebuie să arate dovada creării modelelor (de exemplu fișierele create de utilitar, printscreens-uri cu progresul muncii, evoluția modelului pe parcurs), explicația modului de realizare în documentație (ce unelte a folosit).

## Sunete

(0.1-0.2 per fișier; maxim 1p pt 5 fișiere) Se punctează în funcție de complexitate sunetele create/ înregistrate chiar de către student. în orice utilitar dorește (de exemplu, Audacity). Se punctează până la maxim 5 sunete diferite (Deci prezentați cele mai complexe fișiere; dacă diferențele între sunete sunt mici, se consideră grad mic de complexitate). Studentul trebuie să arate dovada creării fișierelor sunet (de exemplu fișierele create de utilitar, printscreens-uri cu progresul muncii, evoluția track-ului pe parcurs), explicația modului de realizare în documentație (ce unelte a folosit).

## Pachete externe

(0.1 per pachet; maxim 0.3 pt 3 pachete) Adăugarea unui pachet extern și folosirea a minim un mesh/textură/material din el. Se punctează până la maxim 3 pachete diferite

(0.05-0.1) Modificarea parametrilor (cu excepția celor de poziționare pentru actori) a minim unui element introdus din pachetul extern. Se punctează în funcție de complexitate.

(0.1-0.2 per clasă; maxim 1p pt 5 clase) Extinderea unei clase de orice fel din cadrul pachetului și modificarea parametrilor, adăugarea de parametri/metode noi. Se punctează în funcție de complexitate până la maxim 5 clase diferite modificate

**Total: 0.2**



## Listă de Utilitare Folosite

### 1. Unreal Engine Marketplace:

- **Tipul de Utilitar:** Platformă de achiziționare de resurse pentru Unreal Engine.
- **Active Folosite:**
  - **Good Sky Asset:** Descărcat de pe Unreal Engine Marketplace pentru a îmbunătăți aspectul cerului și al iluminării în joc.
  - **Mod de Aplicare:** Integrarea Good Sky Asset a adus un nivel suplimentar de realism și atmosferă în cadrul jocului, oferind o experiență vizuală îmbunătățită.

### 2. Mixamo:

- **Tipul de Utilitar:** Platformă de animații și rigging online.
- **Active Folosite:**
  - **Personaje Rigged și Animatii:** Descărcate de pe Mixamo pentru a furniza personaje animate pentru joc.
  - **Mod de Aplicare:** Utilizarea Mixamo a adus personaje animate și animații de calitate în proiect, fără necesitatea creării propriilor rigging și animații.



## Lista de Pachete Externe Folosite

### 1. Good Sky (Unreal Engine Marketplace):

- **Link:** Good Sky on Unreal Engine Marketplace
- **Descriere:** Pachetul Good Sky a fost descărcat de pe Unreal Engine Marketplace și a fost utilizat pentru a îmbunătăți aspectul cerului și al

iluminării în joc. Acesta a adus o atmosferă realistă și a contribuit la crearea unei experiențe vizuale captivante.

## 2. Mixamo:

- **Link:** [Mixamo](https://mixamo.com)
- **Descriere:** Platforma Mixamo a fost folosită pentru a obține personaje rigate și animații de calitate. Personajele animate descărcate au adăugat diversitate și realism în joc, fără a necesita efort suplimentar pentru rigging și animații personalizate.



## Bibliografie

Pentru realizarea scenei utilizate în acest proiect, am consultat și aplicat metodele prezentate în tutorialul "Building Medieval Worlds" disponibil pe platforma O'Reilly (<https://learning.oreilly.com/course/building-medieval-worlds/9781805124221/>), dar care poate fi găsit și pe Udemy (<https://www.udemy.com/course/building-medieval-worlds-unreal-engine-5-modular-kitbash/>).

Instuctor: **3D Tudor Limited By Neil Ian Bettison**

Titlu:

**Building Medieval Worlds - Unreal Engine 5 Modular Kitbash**

Platformă: O'Reilly

URL:

<https://learning.oreilly.com/course/building-medieval-worlds/9781805124221/>

## Etapa 1

Pentru crearea unui material cu culoare emisivă am urmat acest tutorial pe

YouTube: <https://www.youtube.com/watch?v=kb84BruB7I8&t=167s>

## Etapa 2

Personajul principal este un caracter numit Peasant Girl pe care l-am descărcat de la Mixamo (<https://www.mixamo.com/#/?page=1&type=Character>)

Animatiile pentru personaj le-am descărcat tot de pe Mixamo

- Idle: <https://www.mixamo.com/#/?page=2&type=Motion%2CMotionPack>

- Running: <https://www.mixamo.com/#/?page=2&type=Motion%2CMotionPack>
- Fight 1: <https://www.mixamo.com/#/?genres=Combat&page=1&type=Motion%2CMotionPack>
- Fight 2: <https://www.mixamo.com/#/?genres=Combat&page=2&type=Motion%2CMotionPack>
- Fight 3: <https://www.mixamo.com/#/?genres=Combat&page=1&type=Motion%2CMotionPack>

Pentru Castle Guard, am descrat personajul de pe Mixamo (<https://www.mixamo.com/#/?page=1&type=Character>) si animatia

- Idle: <https://www.mixamo.com/#/?page=1&query=&type=Motion%2CMotionPack>
- Fight: <https://www.mixamo.com/#/?genres=Combat&page=3&type=Motion%2CMotionPack>

Pentru a importa acest personaj in proiect am urmarit acest tutorial de pe YouTube: <https://www.youtube.com/watch?v=PyXeB1QtC0A>

Pentru a implementa Action Mappings si animatii am urmarit acest tutorial pe YouTube: <https://www.youtube.com/watch?v=2xqeci73UG0>

Pentru a implementa schimbarea intre 3rd person si 1st person view am urmarit acest tutorial pe YouTube: <https://www.youtube.com/watch?v=lMuinh0SXU>

Pentru gardian am implementat pasii din acest tutorial de pe YouTube: <https://www.youtube.com/watch?v=xm-7m5Fw1HU&list=PLiSIOaRBfgkfE5tMuS9AQoG41UeToM1Ar&index=4>

Pentru sistemul de Combat am urmat cele trei parti ale acestui tutorial de pe YouTube:

- <https://www.youtube.com/watch?v=eH2ONHEITGA&list=PLiSIOaRBfgkfE5tMuS9AQoG41UeToM1Ar&index=1>
- <https://www.youtube.com/watch?v=-8BTWTf-vrs&list=PLiSIOaRBfgkfE5tMuS9AQoG41UeToM1Ar&index=2>
- <https://www.youtube.com/watch?v=kZVla2uWOiM&list=PLiSIOaRBfgkfE5tMuS9AQoG41UeToM1Ar&index>

Pentru Health System am urmarit acest tutorial de pe YouTube:

<https://www.youtube.com/watch?v=kZVla2uWOiM&list=PLiSIOaRBfgkfE5tMuS9AQoG41UeToM1Ar&index=7>

## **Etapa 6**

Pentru a crea meniul principal am urmarit acest tutorial:

<https://www.youtube.com/watch?v=zWI-36floDQ>

Pentru meniul de pauza am urmarit acest tutorial de pe YouTube:

<https://www.youtube.com/watch?v=Yr8beK9FMe0&t=39s>

Pentru Enhanced input action mapping am urmarit acest video:

<https://www.youtube.com/watch?v=CYiHNbAlp4s&t=188s>

Pentru meniul de optiuni am urmarit urmatorul videoclip de pe YouTube:

<https://www.youtube.com/watch?v=Ff67XtqgSxc>

Pentru ecranul de game over am urmarit acest tutorial de pe YouTube:

[https://www.youtube.com/watch?v=ZpQmL\\_TzSPo](https://www.youtube.com/watch?v=ZpQmL_TzSPo)

## **Etapa 7**

Pentru a crea sistemul de salvare a jocului am urmarit acest tutorial de pe YouTube: <https://www.youtube.com/watch?v=H6rqJbwjRlk&t=1857s>



## **Total Puncte**

10.55 puncte