Ciobanu Constantin-Marcu,1207B

A blue and white logo

Description automatically generated

ANDREW HOOD

Context

Andrew este unul dintre singurii arcasi neinfricati ramasi dupa razboiul orcilor de acum 30 de ani. Acum Andrew isi traieste viata linistit intr-un sat departe de aglomeratia mileniala.

Intr-o zi prietenul sau, Steph, vine la el cu o veste ingrijoratoare. Ultimii mortii vii (undeads) si-au facut o armata fiind gata de recucerirea taramului de smarald. Andrew nu poate spune nu acestei provocari asa ca pleaca in misiunea de salvare a taramului.

Taramul de smarald este unul special intrucat anumite sectiuni din peisaj plutesc efectiv in aer, asadar Andrew va fi nevoit sa isi foloseasca aptitudinile deprinse in tineretea acestuia sa depaseasca orice obstacol.

Sistemul

Jocul este unul de tip RPG-Platformer. Acesta se va desfasura pe mai multe nivele cu greutate crescatoare. Din perspectiva 2D utilizatorul va trebui sa evite obstacolele din fiecare nivel pentru a cartiga. Jocul se va termina atunci cand Andrew va invinge cavalerul mortii :

Utilizatorul va trebui sa se deplaseze pe harta folosind tastele A,D pentru a se deplasa in directiile stanga si dreapta, iar pentru a sari pe platforme sa va folosi tasta

SPACE.

A video game with a cartoon character

Description automatically generated A video game with a couple of characters

Description automatically generated

Jucatorul va trebui sa invinga anumiti inamici cu armele castigate de a lungul nivelelor, si sa isi imbunatateasca arsenalul de lupta pe parcursul calatoriei.

Nivelurile

Fiecare nivel va avea un numar de inamici direct proportional cu parcursul efectuat de jucator (ex : nivel 10 -> 8 inamici) . In cadrul fiecarui nivel jucatorul trebuie sa colecteze un minim de bani ( coins ) pentru a-si putea upgrada arsenalul.

Un nivel se considera completat daca jucatorul a invins macar ½ din inamici si colectat ½ din coins. In functie de dificultatea overall selectata in meniu, aceste proportii vor creste

De-alungul jocului, jucatorul va intalni 4 tipuri de inamici, fiecare cu abilitati individuale si factor de damage amplificat asupra Player-ului.

Enemy 1:



Ghoul – Acest inamic va fi present in primul nivel si va prezenta un nivel de dificultate redus.

Abilitate: Melee Sword

Health points : 100hp

Damage to user: 10hp

Enemy 2:

A cartoon of a skull

Description automatically generated

Skeleton – Acest inamic va fi prezent incepand cu nivelul secundar si va prezenta un nivel de dificultate mediu.

Abilitate: Melee Undead Sword

Health points : 150hp

Damage to user: 15hp

Enemy 3:

A cartoon of a skull wearing a crown

Description automatically generated

Lich – Acest inamic va fi prezent incepand cu nivelul secundar si va prezenta un nivel de dificultate mediu.

Abilitate: Melee Magic Sword

Health points : 150hp

Damage to user: 15hp

Boss Enemy:

A cartoon of a warrior

Description automatically generated

Death Knight – Acest inamic va fi present in ultimul nivel si este adversarul cu cea mai mare dificultate datorita faptului ca sabia sa este una exploziva!

Abilitate: Melee Boom Sword

Health points : 250hp

Damage to user: 30hp + 20hp(Boom)

Nivelul 1

In acest nivel utilizatorul se va familiariza cu jocul si se va intalni cu 2 adversari de tip Ghoul

A screenshot of a video game

Description automatically generated

Nivelul 2

In acest nivel utilizatorul va fi pus la incercare de 2 Ghouls si 1 Skeleto fiind nevoit sa invinga acesti inamici pentru a se dovedi vrednic de nivelul final



Nivelul 3(Boss Fight)

In acest nivel Player-ul va intalni Cavalerul Mortii si 3 witches pentru a se bate cu Cavalerul Player-ul va fi nevoit sa invinga cele 3 Witches



Interfata

Interfata Pause Menu va fi formata din 3 butoane

* Quit – echivalent alt+f4 ->exit game
* Play – resume state of play

Interchimbarea intre Meniu si Joc se executa folosind gamestates ce ne permit sa schimbam statutul actual al jocului, prin randarea unei alte grafici pe ecran. Exemplu:

* Pause Button – switch game state to MENU
* Play Button - switch game state to PLAYING

A screenshot of a video game

Description automatically generated

SPRITES

Andrew:

A cartoon character with a hood

Description automatically generatedA cartoon of a person in a hood

Description automatically generatedA cartoon character holding a bow and arrow

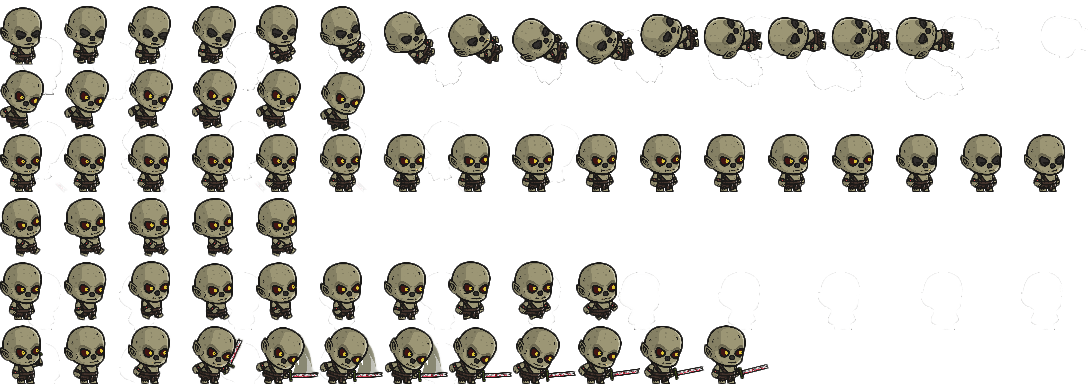
Description automatically generatedA cartoon of a person with a hood

Description automatically generated

A screenshot of a video game

Description automatically generated

Ghoul



Cavalerul Mortii :

A cartoon of a warrior

Description automatically generatedA cartoon of a warrior

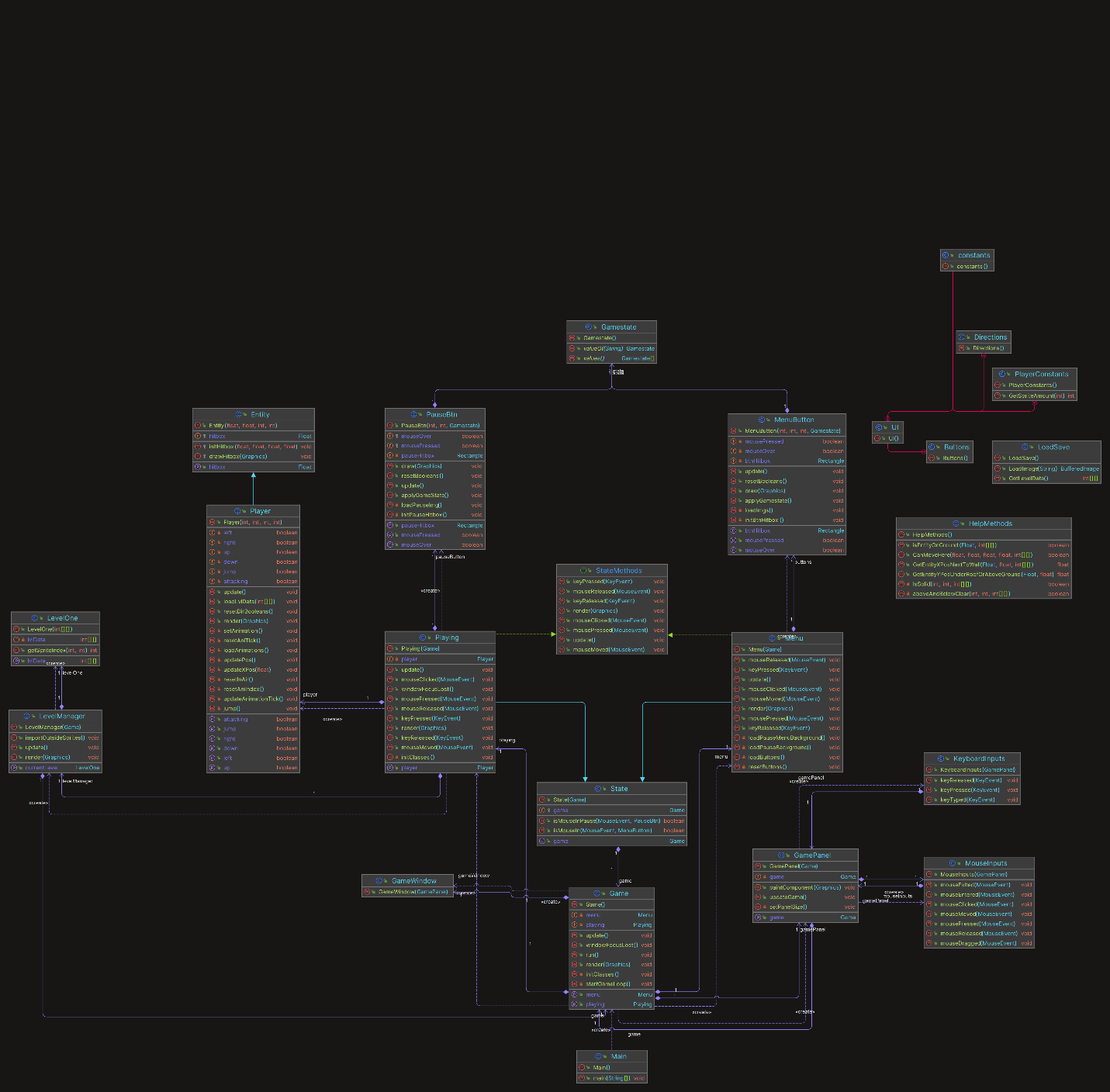
Description automatically generatedA cartoon of a warrior

Description automatically generatedA cartoon of a warrior

Description automatically generatedA collage of cartoon characters

Description automatically generated

Diagrama UML

\*diagrama UML detaliata se regaseste poza atasata arhivei\*

STRUCTURA CODULUI SURSA

Modelul arhitectural (Architectural Design)

Arhitectura aplicației este bazată pe modelul de proiectare State, cu diferite stări reprezentate de clase precum Playing, Pause, Menu, și LevelCompleted. Aceste stări sunt gestionate de clasa Game, care conține o referință la starea curentă a jocului. De asemenea, există clase pentru gestionarea intrărilor de la utilizator (MouseInputs), gestionarea nivelurilor (LevelManager), și gestionarea entităților de joc (EnemyManager, Player). Această arhitectură permite separarea responsabilităților și face codul mai ușor de înțeles și de întreținut.

Modelul interfeței cu utilizatorul (User Interface Design)

Modelul interfeței cu utilizatorul (User Interface Design): Interfața cu utilizatorul este gestionată prin diverse clase, cum ar fi PauseMenuOptionBtn și PauseBtn, care reprezintă butoanele din meniul de pauză. Aceste clase se ocupă de desenarea butoanelor și de gestionarea interacțiunilor cu utilizatorul. De asemenea, există clase precum GameOverOverlay și LevelCompleted care reprezintă diferite ecrane sau overlay-uri din joc.

Sablonul de proiectare ( Design Pattern )

Folosind sablonul State, acesta permite unui obiect să-și schimbe comportamentul atunci când starea sa internă se modifică. Acest lucru se realizează prin delegarea comportamentelor specifice stării către diferite obiecte reprezentând acele stări.

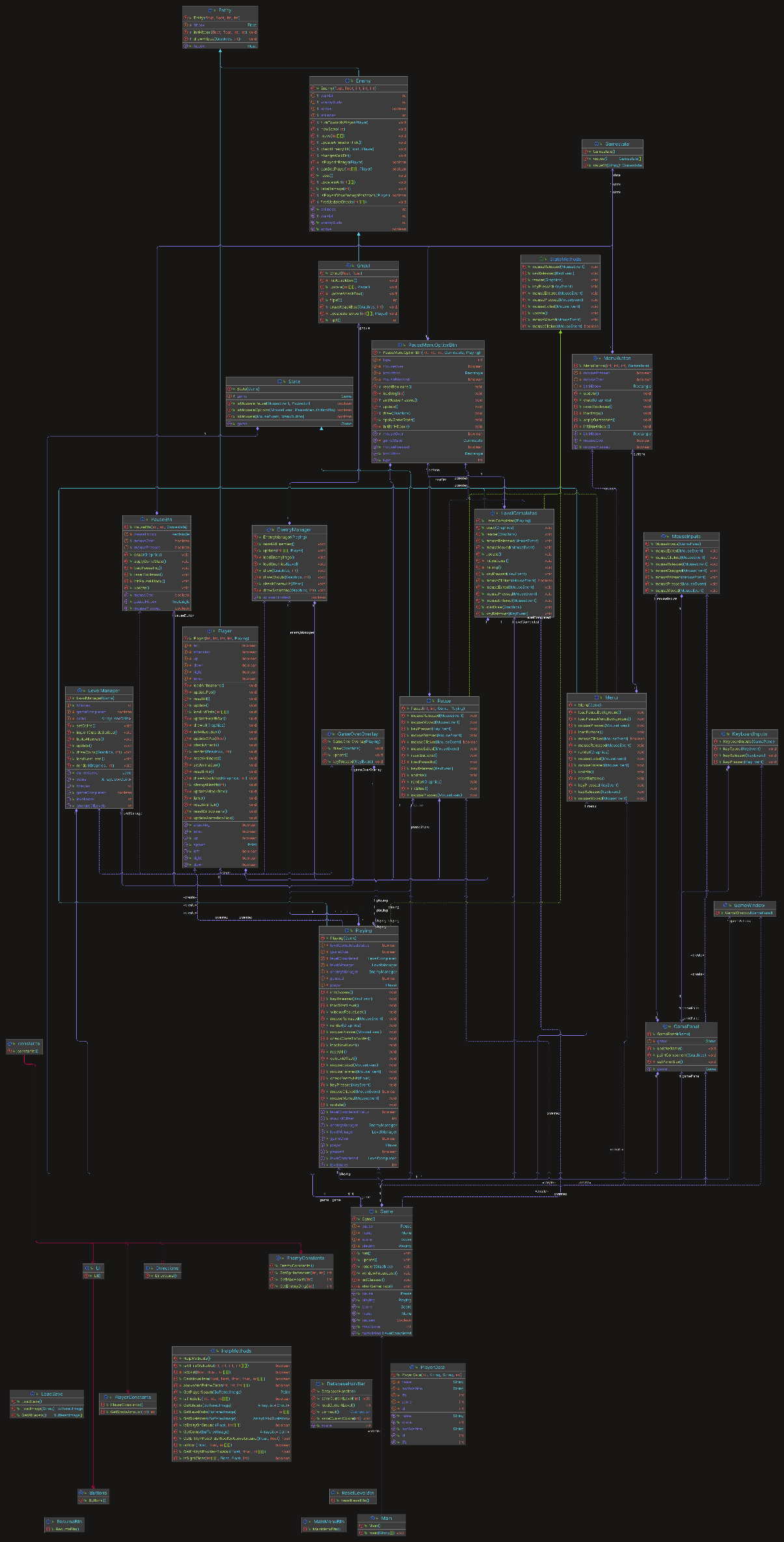
Clasa abstractă State: Aceasta servește ca bază pentru toate stările posibile în jocul dvs. Include metode precum update(), render(), mouseClicked(), mousePressed(), mouseReleased(), mouseEntered(), mouseExited(), mouseMoved(), keyPressed(), și keyReleased() care sunt implementate în fiecare stare concretă. Aceasta este o caracteristică cheie a sablonului State, unde fiecare stare are propriul său set de comportamente.

Clasele de stare concrete: Acestea includ Playing, Pause, Menu, și LevelCompleted. Fiecare dintre aceste clase extinde clasa State și implementează comportamentele specifice pentru starea respectivă. Aceasta este o altă caracteristică cheie a sablonului State, unde comportamentele specifice stării sunt încapsulate în propriile lor clase.

Clasa Game: Aceasta conține o referință la starea curentă a jocului și gestionează schimbarea stărilor. De exemplu, atunci când jocul este pus în pauză, starea curentă este schimbată în Pause. Aceasta este o altă caracteristică cheie a sablonului State, unde contextul (în acest caz, clasa Game) deține o referință la starea curentă și poate să o schimbe.

Clasa MouseInputs: Aceasta gestionează intrările de la mouse și apelează metodele corespunzătoare în funcție de starea curentă a jocului. Acesta este un alt indiciu că ați implementat sablonul de proiectare State.

Prin utilizarea sablonului State, încapsulăm comportamentele specifice fiecărei stări în propriile lor clase, făcând codul mai ușor de înțeles și de întreținut.



BAZA DE DATE

Baza de date utilizată este o bază de date SQLite, care este o bază de date relațională încorporată. SQLite nu necesită un server separat pentru a rula, ceea ce o face potrivită pentru aplicații mici sau pentru testare.

În acest cod, baza de date este utilizată pentru a stoca și recupera starea jocului. Mai precis, se salvează scorul curent și nivelul curent al jocului.

Conexiunea la baza de date se realizează prin metoda `connect()`, care returnează un obiect `Connection`. Acest obiect este folosit pentru a executa interogări SQL asupra bazei de date.

Metoda `getScore()` execută o interogare SQL pentru a obține scorul curent din baza de date. În mod similar, metoda `loadCurrentLevel()` execută o interogare SQL pentru a obține nivelul curent.

Metodele `saveCurrentScore(int score)` și `saveCurrentLevel(int levelIndex)` sunt utilizate pentru a actualiza scorul și nivelul curent în baza de date. Acestea execută o interogare SQL de actualizare pentru a seta noile valori.

Bibliografie:

<https://www.aseprite.org/>

<https://craftpix.net/>

https://opengameart.org/