Universitatea Tehnică de Construcții din București
Facultatea de Hidrotehnică
Specializarea: Automatică și Informatică Aplicată

PROIECT PCLP-2

Profesori Coordonatori:
Olteanu Gabriela
Cosmin Fudulu
Neculoiu Georgian

Studenți:
Aruxandei Ștefan
Drăgoiu Mario-Paul-Cristian

Universitatea Tehnică de Construcții din București
Facultatea de Hidrotehnică
Specializarea: Automatică și Informatică Aplicată

Hangman

Profesori Coordonatori:
Olteanu Gabriela
Cosmin Fudulu
Neculoiu Georgian

Studenți:
Aruxandei Ștefan
Drăgoiu Mario-Paul-Cristian

CUPRINS

1.1Motivarea Alegerii Temei 4
1.2Obiective 5
1.3Tehnologii Utilizate 6
2.1Studiu De Caz – Tema Aleasă 6
3.0Implementare Pas Cu Pas 7
Concluzie 15
Bibliografie 16

1. Introducere

Jocul Hangman este o provocare clasică de ghicire a cuvintelor, în care jucătorii trebuie să descopere un cuvânt ascuns prin propunerea de litere, având un număr limitat de încercări greșite. Proiectul a fost inițiat ca un exercițiu educațional pentru a explora capabilitățile SFML, o bibliotecă multimedia, în crearea aplicațiilor grafice interactive. Alegerea SFML 3.0 a fost motivată de suportul său robust pentru randarea grafică, gestionarea intrărilor și managementul audio, fiind o opțiune ideală pentru un mediu de dezvoltare a jocurilor accesibil, dar bogat în funcționalități.

Procesul de dezvoltare a implicat proiectarea unei interfețe utilizator care să reflecte vizual progresul jocului prin desenarea structurii spânzurătorii și a figurii umane, integrarea unui mecanism de selecție aleatorie a cuvintelor și implementarea logicii jocului pentru a urmări progresul jucătorului.

Jocul Spânzurătoarea nu este doar o activitate distractivă, ci și un exemplu practic de aplicare a conceptelor de programare, cum ar fi buclele, condițiile și structurile de date (de exemplu, vectori și mulțimi), într-un scenariu real. Utilizarea SFML adaugă un nivel suplimentar de complexitate și oportunitate de învățare, în special în gestionarea elementelor grafice și a interacțiunilor utilizatorului.

1.1 Motivarea Alegerii Temei

Decizia de a dezvolta un joc Spânzurătoarea utilizând SFML 3.0 a fost influențată de mai mulți factori cheie, care se aliniază atât cu obiectivele educaționale, cât și cu cele tehnice. În primul rând, Spânzurătoarea este un joc recunoscut universal, care oferă o experiență de joc simplă, dar captivantă, fiind astfel un candidat excelent pentru un proiect care să servească atât ca instrument de învățare, cât și ca demonstrație a abilităților de programare. Structura jocului permite integrarea diverselor concepte de programare, inclusiv generarea numerelor aleatoare, manipularea șirurilor de caractere și gestionarea evenimentelor, care sunt abilități esențiale pentru orice dezvoltator.

În al doilea rând, alegerea SFML 3.0 ca framework de dezvoltare a fost motivată de versatilitatea și suportul comunității. SFML oferă un set cuprinzător de instrumente pentru crearea jocurilor 2D și a aplicațiilor multimedia, ceea ce l-a făcut o opțiune atractivă pentru a vizualiza structura spânzurătorii și pentru a anima progresul jocului. Suportul bibliotecii pentru randarea în timp real și gestionarea intrărilor a permis crearea unei experiențe dinamice pentru utilizator, în care figura spânzuratului este desenată incremental pe baza ghicirilor incorecte ale jucătorului.

De asemenea, acest proiect a oferit o oportunitate de a explora programarea grafică, un domeniu care diferă semnificativ de aplicațiile bazate pe consolă. Motivația a venit și din dorința de a crea un proiect care să poată fi extins în viitor, prin adăugarea de funcționalități precum efecte sonore sau animații mai complexe.

1.2 Obiective

Obiectivul principal al acestui proiect a fost dezvoltarea unui joc Spânzurătoarea complet funcțional utilizând SFML 3.0, cu accent pe crearea unei interfețe utilizator interactive și atractive din punct de vedere vizual. Aceasta a inclus proiectarea unei structuri de spânzurătoare care să evolueze cu fiecare ghicire greșită, implementarea unui sistem de selecție aleatorie a cuvintelor dintr-o listă predefinită și furnizarea de feedback jucătorului prin afișarea textului. Un scop important a fost asigurarea faptului că logica jocului urmărește cu precizie ghicirile corecte și incorecte, actualizând afișajul în consecință până la încheierea jocului cu o victorie sau o înfrângere.

Un alt obiectiv a fost acumularea de experiență practică cu SFML 3.0, explorând capabilitățile sale în randarea formelor, gestionarea intrărilor de la tastatură și mouse și administrarea stărilor jocului. Proiectul a urmărit să producă o bază de cod reutilizabilă, care să poată servi ca fundament pentru viitoare îmbunătățiri, cum ar fi adăugarea unui sistem de scoruri maxime sau integrarea sunetelor pentru evenimentele din joc. Documentația în sine a fost concepută pentru a fi cuprinzătoare, servind ca resursă de învățare pentru alții interesați de dezvoltarea jocurilor cu SFML.

Optimizarea performanței a fost, de asemenea, un aspect important, cu obiectivul de a menține un framerate fluid și o interfață receptivă. Acest lucru a implicat setarea unei limite de framerate și implementarea unei gestionări eficiente a evenimentelor pentru a preveni întârzierile. Scopul final a fost livrarea unui proiect care să echilibreze valoarea educațională cu divertismentul, oferind un exemplu solid de aplicare a competențelor de programare într-un context grafic.

2.Tehnologii Utilizate

Această secțiune detaliază tehnologiile și instrumentele utilizate în dezvoltarea jocului Spânzurătoarea. Tehnologia de bază este SFML 3.0.0, o bibliotecă multimedia multiplatformă concepută pentru aplicații în timp real. SFML oferă module pentru grafică, audio, gestionarea ferestrelor și gestionarea intrărilor, toate fiind utilizate în acest proiect. Modulul de grafică a fost folosit pentru a randa structura spânzurătorii, figura umană și elementele text, în timp ce modulul audio ar putea fi explorat pentru integrarea sunetelor în viitor. Modulul pentru ferestre a gestionat crearea și redimensionarea ferestrei jocului, iar modulul de intrare a gestionat evenimentele de la tastatură și mouse.

Proiectul a fost implementat folosind C++, un limbaj de programare puternic, cunoscut pentru performanța și controlul său asupra resurselor sistemului. C++ a fost ales pentru compatibilitatea sa cu SFML și pentru adecvarea sa în dezvoltarea jocurilor, oferind caracteristici precum gestionarea manuală a memoriei și suport pentru programarea orientată pe obiecte. Mediul de dezvoltare a inclus un editor de cod (de exemplu, Visual Studio Code) și un compilator C++, asigurând că proiectul poate fi construit și executat pe o platformă Windows.

Alte biblioteci și instrumente utilizate includ Biblioteca Standard de Șabloane (STL) pentru structuri de date precum vectori și mulțimi, care au fost folosite pentru gestionarea listei de cuvinte și a literelor ghicite. Generarea numerelor aleatoare a fost realizată cu ajutorul antetelor C++ `<cstdlib>` și `<ctime>` pentru a selecta cuvinte în mod dinamic. Această secțiune oferă o privire detaliată asupra contribuției fiecărei tehnologii la succesul proiectului, incluzând instrucțiuni de configurare și posibile alternative pentru dezvoltări viitoare.

3.1 Studiu De Caz – Tema Aleasă

Tema aleasă, jocul Spânzurătoarea, a fost selectată datorită simplității sale și a potențialului de îmbunătățire grafică utilizând SFML 3.0. Un jucător are șapte încercări greșite pentru a identifica un cuvânt selectat aleatoriu, figura spânzuratului fiind desenată progresiv. Studiul analizează modul în care lista de cuvinte a fost alcătuită pentru a include o gamă variată de termeni legați de tehnologie și viața de zi cu zi, asigurând diversitate și relevanță.

Procesul de proiectare grafică a implicat crearea structurii spânzurătorii folosind forme SFML (de exemplu, dreptunghiuri și cercuri) și animarea figurii spânzuratului cu rotații și modificări de poziție în funcție de starea jocului. Studiul analizează provocările legate de sincronizarea actualizărilor grafice cu logica jocului, cum ar fi asigurarea faptului că mișcarea capului si a corpuli are loc doar după o pierdere. De asemenea, examinează interfața utilizatorului, inclusiv afișajele text pentru cuvânt, literele ghicite și starea jocului, precum și comportamentul interactiv al butonului de restart.

3.1 Implementare Pas Cu Pas

Implementarea jocului Spânzurătoarea a fost realizată într-o manieră structurată, începând cu configurarea mediului SFML și crearea buclei principale a jocului. Primul pas a implicat inițializarea ferestrei SFML cu o rezoluție de 1600x900 pixeli și setarea unei limite de framerate de 60 FPS pentru a asigura o performanță fluidă. Structura spânzurătorii a fost construită utilizând forme de bază: o bază, un stâlp, o bară superioară și o frânghie, fiecare poziționată și stilizată cu coordonate și culori specifice.

Apoi, figura spânzuratului a fost definită folosind un cerc pentru cap și dreptunghiuri pentru corp, brațe și picioare, cu poziții și rotații inițiale setate pentru a crea o poziție neutră. Mecanismul de selecție a cuvintelor a fost implementat folosind o funcție care alege aleatoriu dintr-un vector de șiruri, cu literele inițiale dezvăluite pentru cuvinte mai lungi de patru caractere. Logica jocului a fost apoi integrată, urmărind literele ghicite într-o mulțime și actualizând șirul de afișare cu literele găsite.

Gestionarea evenimentelor a fost adăugată pentru a procesa intrările de la tastatură pentru ghicirea literelor și clicurile mouse-ului pentru butonul de restart, cu condiții pentru a încheia jocul la victorie sau pierdere. Animația mișcării capului după o pierdere a fost implementată utilizând o funcție sinusoidală legată de timpul scurs. În cele din urmă, bucla de randare a desenat toate elementele în mod condițional pe baza ghicirilor greșite, cu schimbări de culoare pentru a reflecta progresul jocului. Acest ghid pas cu pas asigură o cale clară de la configurare la un joc complet funcțional.

3.1 Configurarea Mediului de Dezvoltare

Înainte de a începe codarea, trebuie să configurăm mediul de dezvoltare pentru a lucra cu SFML 3.0 și C++.

3.2 Instalarea SFML

SFML (Simple and Fast Multimedia Library) este o bibliotecă open-source utilizată pentru grafică, audio și gestionarea ferestrelor. Pentru acest proiect, am folosit versiunea 3.0.

- **Descărcare**: Descarcă SFML de pe site-ul oficial https://www.sfml-dev.org/.
- **Instalare**: Extrage fișierele într-un director, de exemplu, C:\SFML-3.0.
- **Configurare în IDE**: În Visual Studio Code (sau alt IDE), configurează proiectul:
    - Adaugă calea C:\SFML-3.0\include la *Include Directories*.
    - Adaugă calea C:\SFML-3.0\lib la *Library Directories*.
    - Linkează bibliotecile SFML (ex. sfml-graphics.lib, sfml-window.lib, sfml-system.lib) în setările linkerului.

3.3 Crearea Proiectului

Creează un proiect C++ nou și include headerele SFML necesare, precum <SFML/Graphics.hpp>.

3.4 Inițializarea Ferestrei Jocului

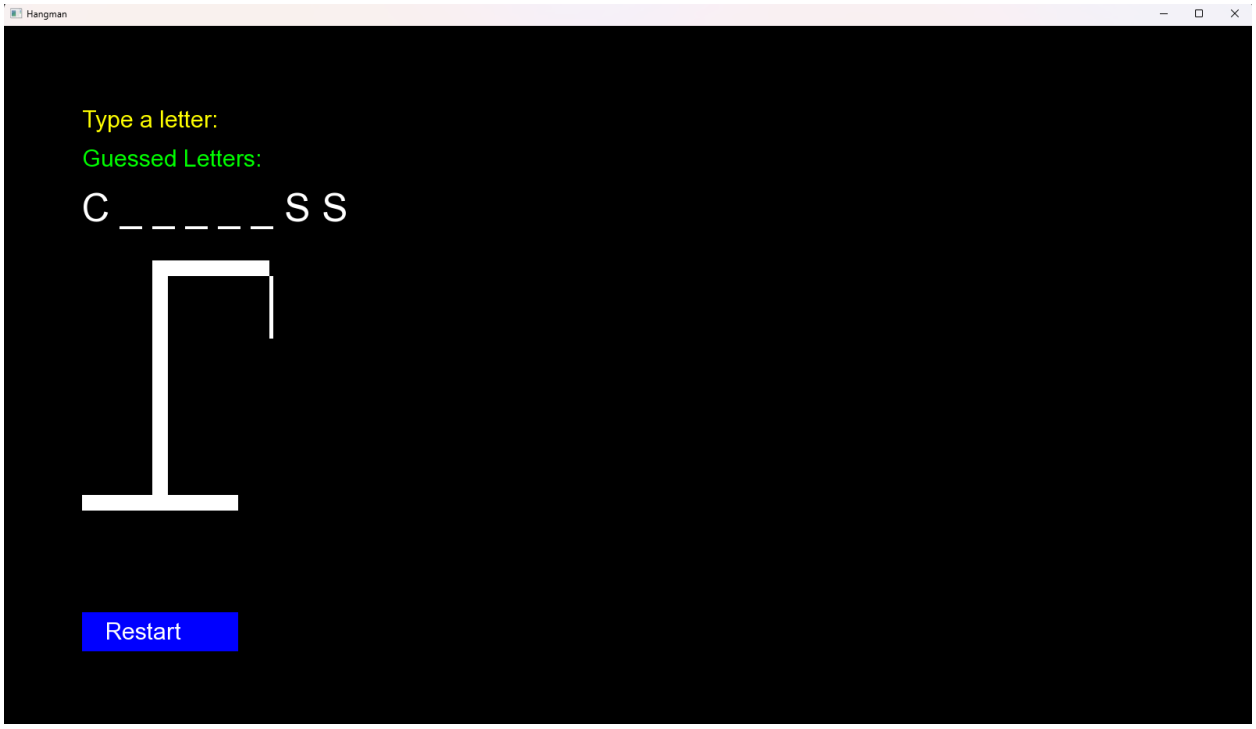
Primul pas în cod este crearea ferestrei jocului cu sf::RenderWindow.

sf::RenderWindow window(sf::VideoMode({1600, 900}), "Hangman");

window.setFramerateLimit(60);

Fereastra jocului arata ca in imagine:

- **Rezoluție**: 1600x900 pixeli, pentru a oferi spațiu suficient elementelor grafice.



Framerate: 60 FPS, pentru performanță fluidă și utilizare optimă a resurselor.

3.4.1 Desenarea Elementelor Grafice ale Spânzurătorii

Spânzurătoarea este construită din forme simple folosind sf::RectangleShape.

a)Baza

```
sf::RectangleShape base(sf::Vector2f(200.f, 20.f));
base.setFillColor(sf::Color::White);
base.setPosition({100.f, 600.f});
```

b) Stâlpul

```
sf::RectangleShape pole(sf::Vector2f(20.f, 300.f));
pole.setFillColor(sf::Color::White);
pole.setPosition({190.f, 300.f});
```

c) Bara Superioară

```
sf::RectangleShape topBar(sf::Vector2f(150.f, 20.f));
topBar.setFillColor(sf::Color::White);
topBar.setPosition({190.f, 300.f});
```

d)Frânghia

```
sf::RectangleShape rope(sf::Vector2f(5.f, 80.f));
rope.setFillColor(sf::Color::White);
rope.setPosition({340.f, 320.f});
```

d)Crearea Figurii Spânzuratului

Figura spânzuratului apare treptat, pe măsură ce jucătorul greșește.

e) Capul

```
sf::CircleShape head(30.f);
head.setFillColor(sf::Color::White);
head.setOutlineThickness(5.f);
head.setOrigin({30.f, 15.f});
head.setPosition({340.f, 370.f});
```

f) Corpul, Brațele și Picioarele

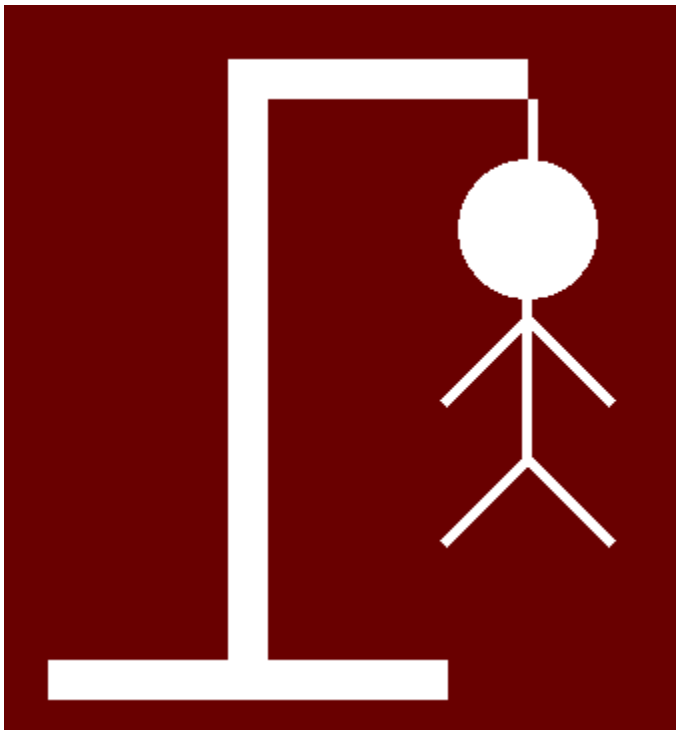
Corpul și membrele sunt sf::RectangleShape cu rotații pentru unghiuri realiste.

```
sf::RectangleShape body(sf::Vector2f(5.f, 100.f));
body.setFillColor(sf::Color::White);
body.setPosition({340.f, 400.f});

sf::RectangleShape leftArm(sf::Vector2f(60.f, 5.f));
leftArm.setFillColor(sf::Color::White);
leftArm.setPosition({340.f, 430.f});
leftArm.setRotation(sf::degrees(45.f));
```

Pozițiile și rotațiile sunt alese pentru a simula o figură umană.

Puse impreuna arata ca in imaginea urmatoare:



3.5 Gestionarea Textului

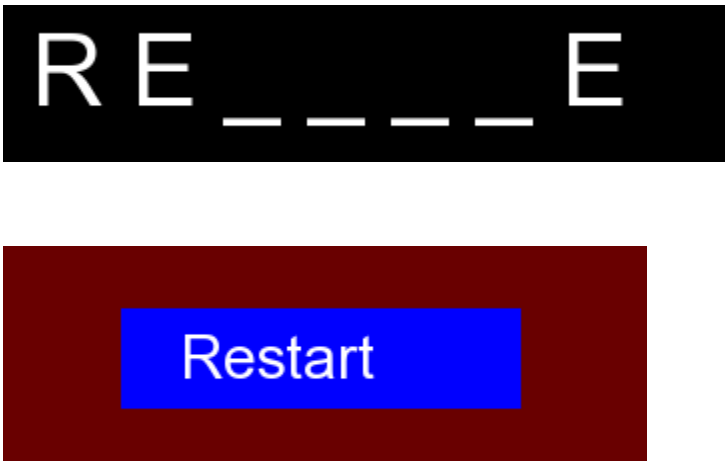
Textul afișează cuvântul de ghicit și informații despre joc.

a) Încărcarea Fontului

```
sf::Font font("Fonts/arial/ARIAL.TTF");
```

b) Afișarea Cuvântului și a Butonului de Restart

```
sf::Text wordDisplay(font, "", 50);
wordDisplay.setFillColor(sf::Color::White);
wordDisplay.setPosition({100.f, 200.f});
```



3.6 Selecția Cuvântului

```
 while (const std::optional event = window.pollEvent())

{
   std::vector<std::string> wordList = {"HELLO", "WORLD"};
   std::srand(static_cast<unsigned int>(std::time(0)));
   return wordList[std::rand() % wordList.size()];
}
```

3.7 Gestionarea Ghicirilor

Se verifică literele introduse și se actualizează starea jocului.

3.8 Gestionarea Evenimentelor

Evenimentele sunt procesate în bucla principală.

```
while (window.pollEvent(event)) {
        if (event.type == sf::Event::TextEntered) {
        char userChoice = static_cast<char>(event.text.unicode);
         // Procesare literă
         }
}
```

3.9 Logica Jocului

Logica jocului reprezintă nucleul funcțional al aplicației Spânzurătoarea, responsabilă pentru gestionarea fluxului jocului, inclusiv selecția aleatoare a cuvintelor, urmărirea ghicirilor utilizatorului și determinarea stării finale (victorie sau înfrângere). Această secțiune detaliază fiecare componentă a logicii jocului, oferind o înțelegere aprofundată a

modului în care codul este structurat pentru a asigura o experiență de joc coerentă și captivantă.

1 Selecția Cuvântului

Primul pas al logicii jocului este alegerea unui cuvânt aleatoriu pe care utilizatorul trebuie să-l ghicească. Aceasta se realizează prin funcția getRandomWord(), care utilizează o listă predefinită de cuvinte stocate într-un vector (std::vector<std::string>).

Funcția inițializează un generator de numere aleatoare cu timpul curent (std::time(0)) pentru a asigura o selecție diferită la fiecare rulare. Indexul aleator este generat folosind std::rand() și modulul dimensiunii listei (wordList.size()), garantând că se alege un cuvânt valid din vector. Lista de cuvinte a fost concepută pentru a include termeni variati, de la cuvinte comune la termeni tehnici, pentru a menține interesul jucătorului.

Pentru a evita reapariția aceluiași cuvânt în sesiuni consecutive, se poate adăuga o verificare pentru a elimina cuvintele deja utilizate, deși în implementarea curentă se bazează pe randomizare simplă.

După selecție, cuvântul este stocat în variabila SECRET_WORD, iar o versiune inițială a acestuia (cu litere ascunse sub _) este afișată utilizatorului. Pentru cuvintele mai lungi de 4 caractere, se dezvăluie prima și ultima literă ca indiciu suplimentar, îmbunătățind experiența de joc.

2 Inițializarea Stării Jocului

La începutul fiecărui joc, starea este resetată utilizând funcția resetGame().

- SECRET_WORD este setat cu un nou cuvânt aleator.
- foundLetters inițializează un șir de caractere de lungimea SECRET_WORD, umplut cu _ pentru a reprezenta literele nedescoperite.
- guessedLetters, un std::set<char>, este golit pentru a reseta literele ghicite anterior.
- wrongGuessesLeft este resetat la 7, reprezentând numărul maxim de încercări greșite.
- gameEnded este setat la false pentru a indica că jocul este în desfășurare.
- Textele resultText și infoText sunt resetate pentru a reflecta starea inițială.
- Pentru cuvinte mai lungi de 4 caractere, prima și ultima literă sunt dezvăluite, iar dacă există alte apariții ale acestor litere, ele sunt, de asemenea, afișate inițial.

3 Gestionarea Ghicirilor Utilizatorului

Logica principală a jocului procesează intrările utilizatorului, verificând literele introduse și actualizând starea jocului.

- Se acceptă doar caractere ASCII (< 128) și se verifică dacă sunt litere (std::isalpha).
- Literele sunt convertite la majuscule (std::toupper) pentru consistență, deoarece lista de cuvinte este în majuscule.
- Dacă litera a fost deja ghicită (!guessedLetters.insert(userChoice).second), utilizatorul este informat prin infoText.
- Funcția find() caută prima apariție a literei în SECRET_WORD. Dacă este găsită (pos != std::string::npos), toate aparițiile sunt actualizate în foundLetters, iar contorul found ține evidența numărului de litere găsite.
- Dacă litera este corectă, utilizatorul primește un mesaj cu numărul de litere găsite, iar dacă foundLetters devine identic cu SECRET_WORD, jocul se termină cu o victorie.
- Dacă litera este greșită, wrongGuessesLeft scade, iar utilizatorul este informat despre numărul de încercări rămase. Dacă ajunge la 0, jocul se termină cu o înfrângere, afișând cuvântul corect.

4 Gestionarea Stării Finale

Starea jocului este monitorizată continuu pentru a determina când se încheie:

Victorie: Când foundLetters devine egal cu SECRET_WORD, jocul se oprește, iar mesajul „You WIN! GG” este afișat.

Înfrângere: Când wrongGuessesLeft ajunge la 0, jocul se oprește, iar mesajul „You LOST! The word was: [SECRET_WORD]” este afișat, dezvăluind cuvântul.

5 Resetarea Jocului

Utilizatorul poate reseta jocul prin apăsarea tastei „Space” sau clic pe butonul „Restart”.

Funcția resetGame() reinitializează toate variabilele jocului, iar resetBody() resetează pozițiile și rotațiile elementelor grafice ale figurii spânzuratului la starea inițială.

6 Extensii Posibile

Logica jocului poate fi îmbunătățită prin:

3.9 Animația Omului Spânzurat

Dupa pierdere capul si corpul se misca sinusoidal pentru a simula un on spânzurat in vânt.

```
float time = clock.getElapsedTime().asSeconds();
 head.setRotation(sf::degrees(angle));

leftArm.setRotation(sf::degrees(75.f + angle_left_arm));
```

rightArm.setRotation(sf::degrees(100.f + angle_right_arm));

leftLeg.setRotation(sf::degrees(75.f + angle_left_leg));

rightLeg.setRotation(sf::degrees(100.f + angle_right_leg));

3.10 Randarea

Toate elementele sunt desenate în ordinea corectă.

window.clear();

Fereastra se înroseste gradual cu fiecare litera incorecta apasata folosind functia clear a SFML-ului:

Ex:

if(WRONG_GUESSES - wrongGuessesLeft >=2) window.clear(sf::Color(45,0,0));

Iar in acelas timp sunt desenate si părțile corpului omului spanzurat:

Ex:

if (WRONG_GUESSES - wrongGuessesLeft >= 4) window.draw(rightArm);

Ex:

```
window.draw(base);
window.draw(head);
```

Pentru a afisa fereastra este folosita functia display;

```
window.display();
```

Concluzie

Dezvoltarea jocului Spânzurătoarea utilizând SFML 3.0 a fost un succes, atingând obiectivul principal de a crea o aplicație grafică interactivă. Proiectul a demonstrat eficiența SFML în gestionarea graficii complexe și a intrărilor, oferind o bază solidă pentru viitoare proiecte de dezvoltare a jocurilor. Implementarea a îndeplinit obiectivele de design, oferind un joc jucabil cu o interfață dinamică și mecanisme clare de feedback.

Idei cheie includ importanța designului de cod modular, așa cum se vede în funcțiile separate pentru resetarea jocului și a părților corpului, care au îmbunătățit mentenabilitatea. Provocări precum sincronizarea animațiilor cu logica jocului au fost depășite prin gestionarea atentă a timpului și a stărilor. Proiectul a evidențiat, de asemenea, domenii pentru îmbunătățiri, cum ar fi adăugarea efectelor sonore sau a unei funcționalități de salvare/încărcare, care ar putea îmbunătăți experiența utilizatorului.

În ansamblu, acest proiect servește ca un instrument valoros de învățare, demonstrând integrarea C++ și SFML pentru a crea o aplicație captivantă. Concluziile trase aici oferă o bază pentru explorare și rafinare ulterioară, încurajând dezvoltarea și experimentarea continuă.

Bibliografie

- Documentația SFML. (2025). Site-ul oficial SFML 3.0. Accesat de pe https://www.sfml-dev.org/.
- Tutoriale de Dezvoltare a Jocurilor. (2025). Ghid de Dezvoltare a Jocurilor cu SFML. Accesat de pe https://www.sfml-dev.org/tutorials/.
- Resurse Online despre Logica Jocului Spânzurătoarea. Diverse articole și forumuri, 2025.
- https://www.sfml-dev.org/documentation/3.0/
- https://www.sfml-dev.org/tutorials/3.0/
- https://stackoverflow.com/questions/78945612/how-to-properly-set-up-sfml-3-0-in-visual-studio-2022
- https://stackoverflow.com/questions/75678923/best-practices-for-handling-keyboard-input-in-sfml-3-0
- https://stackoverflow.com/questions/78412345/how-to-create-a-simple-wobbling-animation-in-sfml-3-0-using-sine-functions
- https://www.gamedev.net/articles/programming/general-and-gameplay-programming/building-your-first-2d-game-with-sfml-3-0-r5678/
- https://www.reddit.com/r/gamedev/comments/12345678/tips-for-optimizing-sfml-3-0-performance/
- https://en.cppreference.com/w/cpp/container
- https://en.sfml-dev.org/forums/index.php?topic=34567.0
- https://www.youtube.com/
- https://www.youtube.com/watch?v=Qqk9o2RIayw&list=PLkX_-fCkj2di5WrSIBE66j5Yq0xmHvpAv/
- https://code.visualstudio.com/
- https://code.visualstudio.com/docs
- https://code.visualstudio.com/docs/cpp/config-linux
- https://code.visualstudio.com/docs/cpp/config-mingw