Test Plan Snake

2024/2025

Cuprins

[1. Generalități 3](#_Toc187967804)

[1.1. Scopul 3](#_Toc187967805)

[1.2. Versiunea 3](#_Toc187967806)

[1.3. Stadiul de dezvoltare 3](#_Toc187967807)

[1.4. Nivelul de calitate 4](#_Toc187967808)

[2. Resurse 4](#_Toc187967809)

[2.1. Personal 4](#_Toc187967810)

[2.2. Instrumente software 5](#_Toc187967811)

[2.3. Locațiile 5](#_Toc187967812)

[2.4. Instrumente hardware 5](#_Toc187967813)

[3. Responsabilități generale 6](#_Toc187967814)

[4. Acoperirea 6](#_Toc187967815)

[5. Strategia de testare 6](#_Toc187967816)

[6. Programarea testelor 7](#_Toc187967817)

[7. Cazurile de testare 7](#_Toc187967818)

[7.1. Testarea limitelor 7](#_Toc187967819)

[7.2. Testarea funcționalității 8](#_Toc187967820)

[7.3. Testarea UI/UX 8](#_Toc187967821)

[7.4. Testarea proceselor 9](#_Toc187967822)

[8. Rapoartele de testare 10](#_Toc187967823)

[9. Metricile procesului de testare 10](#_Toc187967824)

[10. Riscuri 10](#_Toc187967825)

[11. Concluzii 11](#_Toc187967826)

1. Generalități

Acest document reprezintă planul de testare a jocului Snake, realizat în conformitate cu cerințele și specificațiile beneficiarilor.

Snake este o reinterpretare a jocului clasic cu același nume, apărut în anii 70. Jucătorii controlează un șarpe cu ajutorul butoanelor, având ca obiectiv să adune cât mai multe puncte fără a se ciocni de pereți sau de propriul corp. Pe măsură ce șarpele mănâncă punctele, acesta crește în lungime, iar dificultatea jocului crește progresiv.

* 1. Scopul

Scopul realizării planului de testare pentru aplicația Snake este de a asigura calitatea, funcționalitatea și fiabilitatea jocului prin identificarea, documentarea și remedierea posibilelor erori înainte de lansarea sa finală. Planul de testare urmărește să valideze toate funcționalitățile de bază, cum ar fi coliziunea șarpelui cu marginile sau cu propriul corp, răspunsul la inputuri, afișarea corectă a scorului și comportamentul aplicației în diferite condiții, inclusiv schimbarea limbii sau gestionarea ieșirii din joc. De asemenea, acesta verifică limitele aplicației, comportamentul în situații extreme și compatibilitatea cu diferite configurații hardware. Printr-o structurare clară a cazurilor de testare, planul ajută la prevenirea regresiilor, crește satisfacția utilizatorilor și garantează o experiență de joc fără probleme.

* 1. Versiunea

Pentru această aplicație am ales să procedăm conform conceptului de testare ciclică: se testează aplicația pe mai multe versiuni și după fiecare testare se remediază problemele observate. Acest ciclu continuu permite identificarea timpurie a defectelor, validarea fix-urilor și asigurarea că noile modificări nu afectează negativ alte funcționalități existente (prevenirea regresiilor).

Testele realizate până la data prezentării se vor axa pe 2 versiuni ale aplicației: versiunea 0.1(alpha) și versiunea 0.2(beta). Versiunea care va fi prezentată este cea beta. Produsul este unul nou realizat la comanda beneficiarilor.

* 1. Stadiul de dezvoltare

Produsul Snake se află în stadiul de prototip, ceea ce înseamnă că acesta reprezintă o versiune preliminară, cunoscută și ca varianta Alpha, destinată testării interne. În acest stadiu, aplicația conține funcționalitățile de bază, cum ar fi mișcarea șarpelui, gestionarea coliziunilor, afișarea scorului și interacțiunea cu utilizatorul prin intermediul meniului principal. Cu toate acestea, unele funcționalități pot fi incomplete sau imperfect implementate, iar produsul este predispus la erori și comportamente neașteptate.

* 1. Nivelul de calitate

Nivelul de calitate urmărit pentru această etapă a aplicației Snake este unul funcțional și bine definit, asigurând că toate componentele de bază ale jocului sunt implementate și funcționează corespunzător. Obiectivul este ca aplicația să ruleze fără probleme majore și să permită utilizatorilor să interacționeze cu elementele esențiale, cum ar fi mișcarea șarpelui, coliziunile și afișarea scorului. Această calitate funcțională este suficient de avansată pentru a permite beneficiarilor să evalueze în detaliu dacă cerințele lor sunt respectate și dacă experiența oferită corespunde așteptărilor.

1. Resurse
   1. Personal

Manager Proiect – Rus David-Sorin

Date contact:

[david.sorin.rus@stud.ubbcluj.ro](mailto:david.sorin.rus@stud.ubbcluj.ro)

Sediu: Cluj-Napoca, Cluj

Programator – Roșculeț Cosmin

Date contact:

[cosmin.rosculet@stud.ubbcluj.ro](mailto:cosmin.rosculet@stud.ubbcluj.ro)

Sediu: Comarnic, Prahova

Tester – Stelic Ionuț-Marius

Date contact:

[ionut.stelic@stud.ubbcluj.ro](mailto:ionut.stelic@stud.ubbcluj.ro)

Sediu: Prundu Bârgăului, Bistrița

Tester – Pusok Alex-Dănuț

Date contact:

[alex.pusok@stud.ubbcluj.ro](mailto:alex.pusok@stud.ubbcluj.ro)

Sediu: Zalău, Sălaj

* 1. Instrumente software

Pentru dezvoltarea aplicației, s-a utilizat limbajul de programare **Python**, care a oferit flexibilitate și eficiență în implementarea funcționalităților. Grafica aplicației a fost realizată cu ajutorul bibliotecii **Turtle**, care permite crearea de interfețe vizuale simple și interactive. Pentru testarea automată a aplicației, a fost folosit framework-ul **unittest**, asigurându-se astfel o validare corectă a funcționalităților. Pentru înregistrarea video a eventualelor erori care vor avea nevoie de demonstrații video se va folosi **OBS**. Codul sursă al aplicației este stocat pe platforma **GitHub**, care facilitează gestionarea versiunilor și colaborarea între dezvoltatori. Documentația proiectului a fost realizată în **Microsoft Word** și **PDF**, pentru o prezentare clară și organizată, iar comunicarea între echipă s-a realizat prin aplicațiile **WhatsApp** și **Discord**.

* 1. Locațiile

Locațiile principale de lucru pentru dezvoltarea aplicației au fost domiciliile personale ale membrilor echipei și campusul Facultății de Științe Economice și Gestiunea Afacerilor din cadrul Universității Babeș-Bolyai (UBB).

* 1. Instrumente hardware

Instrumentele principale de hardware sunt laptopurile și telefoanele personale ale membrilor echipei.

1. Responsabilități generale

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Manager  Rus David | Programator  Roșculeț Cosmin | Tester  Stelic Ionuț | Tester  Pusok Alex |
| Cerințele utilizatorului | X |  |  |  |
| Specificații | X |  |  |  |
| Plan de testare | X |  |  |  |
| Cod sursă |  | X |  |  |
| Test case |  |  |  | X |
| Bug report |  |  | X |  |
| Testare automată |  |  |  | X |

1. Acoperirea

Testele planificate nu acoperă întreaga gamă de teste disponibile în domeniul testării software. Se va pune accent pe testarea limitelor (Limit Testing), pe testele de funcționalitate (Functional Testing), pe evaluarea interfeței utilizatorului (UI/UX) și pe verificarea proceselor de închidere și gestionare a aplicației. Alte tipuri de teste, cum ar fi cele de performanță, stres, compatibilitate sau securitate, nu vor face obiectul testării în această etapă a proiectului.

1. Strategia de testare

În perioada de testare, vor fi utilizate strategii precum testarea black box, care se va concentra pe verificarea funcționalității aplicației fără a analiza codul sursă, evaluând doar rezultatele obținute în urma interacțiunii cu utilizatorul. Testarea manuală va implica rularea unor scenarii de test pentru a verifica comportamentul aplicației în diferite condiții. Testarea automată va fi folosită pentru a verifica rapid și repetat funcționalitățile aplicației. Testarea pozitivă va asigura că aplicația funcționează corect în condiții normale, iar error forcing va încerca să provoace erori pentru a evalua cum gestionează aplicația situațiile neprevăzute. Aceste strategii vor ajuta la o testare completă a aplicației.

1. Programarea testelor

Testarea primei versiuni a aplicației va începe imediat după finalizarea acesteia și se va desfășura pe o perioadă de 3 zile. În acest interval, echipa de testare va evalua funcționalitățile aplicației, identificând eventualele erori și disfuncționalități. Problemele descoperite vor fi documentate și raportate pentru a fi remediate în cel mai scurt timp. După implementarea modificărilor, versiunea beta a aplicației va fi lansată pentru testare extinsă, pe o perioadă de o săptămână. Acest proces va permite verificarea îmbunătățirilor și identificarea unor eventuale probleme suplimentare în condiții mai apropiate de utilizarea reală, asigurând astfel o stabilitate mai bună a produsului.

1. Cazurile de testare

În continuare vor fi prezentate principalele categorii de teste care vor fi realizate, obiectivele pe care dorim să le atingem cu ajutorul acestora și testele efective care vor fi efectuate.

* 1. Testarea limitelor

Testarea limitelor este un proces esențial în evaluarea unei aplicații, concentrându-se pe verificarea comportamentului acesteia la valori extreme sau aproape de limitele funcționale definite. Această metodă este utilizată pentru a identifica erori care pot apărea atunci când sistemul este supus unor condiții neobișnuite sau de graniță, cum ar fi marginea ecranului, lungimea maximă a șarpelui sau alte situații de limită.

Obiective:

În Snake, testarea limitelor urmărește să verifice comportamentul aplicației în situații extreme, cum ar fi coliziunile cu marginea spațiului de joc. Obiectivul este să ne asigurăm că jocul gestionează corect astfel de situații, declanșând mesajul "Game Over" și oprind șarpele din mișcare.

Teste efective:

* Încheierea jocului la lovirea marginii: Verificăm dacă jocul detectează corect coliziunea șarpelui cu marginea spațiului de joc și dacă afișează mesajul "Game Over" atunci când această coliziune are loc.
* Șarpele nu detectează pereții jocului și continuă să iasă din raza ecranului: Testăm situația în care șarpele este orientat spre marginea ecranului pentru a observa dacă aplicația permite șarpelui să părăsească spațiul de joc sau dacă gestionează corect această limită.
  1. Testarea funcționalității

Testarea funcționalității are ca scop verificarea principalelor mecanisme ale jocului pentru a asigura o experiență stabilă și fără erori majore. Scopul acestei testări este să confirmăm că aplicația se comportă conform specificațiilor și că logica jocului este implementată corect.

Obiective:

Dorim să ne asigurăm că șarpele se mișcă conform comenzilor utilizatorului, coliziunile sunt detectate corect, iar consumarea mâncării declanșează toate acțiunile necesare, cum ar fi creșterea șarpelui și generarea unei noi bucăți de mâncare. Aceste teste sunt esențiale pentru validarea logicii de bază a jocului.

Teste efective:

* Încheierea jocului când șarpele se lovește singur: Testăm dacă jocul detectează coliziunea dintre capul șarpelui și corpul său, afișând corect ecranul "Game Over".
* Răspunderea șarpelui la inputuri: Verificăm dacă șarpele răspunde corect la comenzile utilizatorului, menținând restricțiile, cum ar fi imposibilitatea de a schimba direcția cu 180 de grade.
* Consumarea mâncării: Testăm dacă șarpele crește cu o unitate atunci când consumă mâncarea și dacă mâncarea reapare în alte poziții pe ecran.
  1. Testarea UI/UX

Testarea UI/UX este realizată pentru a asigura o interfață intuitivă și plăcută din punct de vedere vizual, precum și o experiență ușor de înțeles pentru utilizator. Aceasta se concentrează pe verificarea afișajului și a interacțiunilor vizibile, astfel încât utilizatorul să aibă acces la informațiile esențiale în timp real.

Obiective:

Se urmărește verificarea coerenței interfeței, inclusiv a schimbării limbii între engleză și română, pentru a asigura accesibilitatea și o experiență plăcută, indiferent de preferințele utilizatorului. Prin aceste teste, ne dorim să eliminăm orice confuzie și să creăm o interfață clară și atractivă.

Teste efective:

* Afișarea scorului curent în timpul jocului: Se testează dacă scorul curent este afișat corect pe ecran și dacă acesta crește corespunzător atunci când șarpele consumă mâncarea.
* Schimbarea limbii interfeței: Verificăm funcționalitatea opțiunii de schimbare a limbii, testând dacă aplicația trece cu succes între limba engleză și română, fără să afecteze performanța sau alte elemente vizuale ale jocului.
  1. Testarea proceselor

Testarea proceselor validează gestionarea corectă a proceselor aplicației, în special în ceea ce privește închiderea și afișarea informațiilor post-joc. Aceasta asigură că aplicația se comportă conform așteptărilor atunci când utilizatorul încheie o sesiune de joc.

Obiective:

Ne dorim ca aplicația să se închidă complet, fără a lăsa procese active, și să afișeze corect scorurile, atât pentru runda curentă, cât și pentru high score. Acest lucru garantează o experiență finalizată corect pentru utilizator și contribuie la funcționarea aplicației fără probleme pe termen lung.

Teste efective:

* Închiderea aplicației la apăsarea butonului Quit: Testăm dacă aplicația se închide complet, fără a lăsa procese active în fundal, în special după terminarea unei runde de joc.
* Afișarea high score-ului corect după fiecare rundă: Verificăm dacă scorurile sunt afișate corect în ecranul "Game Over" și dacă high score-ul este actualizat corespunzător în funcție de performanța fiecărei runde.

1. Rapoartele de testare

În cadrul procesului de testare a aplicației Snake, rapoartele vor fi întocmite manual pentru a documenta în detaliu rezultatele fiecărui test. Aceste rapoarte vor include descrierea erorilor întâlnite, pașii necesari pentru a reproduce problemele și observațiile relevante pentru echipa de dezvoltare. Documentația va fi realizată în formate simple, precum fișiere Word sau Notepad, iar în cazurile în care este necesară o demonstrație vizuală a erorilor, acestea vor fi filmate și stocate pentru referință.

1. Metricile procesului de testare

Metricile utilizate pentru a evalua eficiența procesului se vor rezuma la erorile detectate de un tester. Fiecare tester va documenta erorile găsite, iar performanța individuală va fi monitorizată pe parcursul întregii perioade de testare. Aceste metrici vor fi importante pentru a evalua modul în care fiecare tester reușește să identifice problemele și pentru a urmări progresul general al testării.

1. Riscuri

Principalele riscuri identificate se referă la impactul activităților academice asupra disponibilității echipei pentru realizarea testării. Având în vedere că membrii echipei sunt și studenți, există riscul ca timpul disponibil pentru testare să fie limitat din cauza altor responsabilități școlare. Acest lucru ar putea duce la întârzieri în procesul de testare și remediere a erorilor.

De asemenea, dacă termenul de livrare este depășit, se poate produce o presiune suplimentară asupra echipei, care poate influența calitatea testării și implementarea corectă a tuturor schimbărilor necesare. Gestionarea eficientă a timpului și prioritizarea activităților este esențială pentru a minimiza aceste riscuri și a asigura finalizarea proiectului la un nivel de calitate corespunzător cerințelor.

1. Concluzii

Planul de testare propus are ca scop identificarea și corectarea erorilor într-un mod organizat și eficient, utilizând metode diverse, precum testarea manuală și automată, alături de testarea limitelor, funcțională și a interfeței utilizator. Aceste acțiuni vor contribui la obținerea unui produs final de calitate, care să îndeplinească cerințele utilizatorilor și să fie funcțional în diverse scenarii. Testarea ciclică va permite remedierea constantă a problemelor și îmbunătățirea aplicației pe măsură ce aceasta progresează.