**Aplicație de tip checker**

**-CheckerApp-**



C-112C

Sd.Cap. Heașcă Antonio

Sd.Cap. Sîrbu Bianca

Cuprins

Capitolul 1 – INTRODUCERE

* 1. Scopul proiectului
  2. Lista definițiilor
  3. Structura DCS

Capitolul 2 – DESCRIEREA GENERALĂ A PRODUSULUI SOFTWARE

* 1. Descrierea produsului software
  2. Detaliera platformei HW/SW
  3. Constrângeri

Capitolul 3 – DETALIEREA CERINȚELOR SPECIFICE

* 1. Cerințele funcționale
  2. Cerințele ne-funcționale

## Tabel versiuni

|  |  |
| --- | --- |
| **Numărul versiunii** | **Modificări adăugate** |
| Versiunea 1 | Versiune inițială |

## 

## Introducere

### 1.1 Scopul proiectului

Scopul proiectului este de a eficientiza timpul petrecut de către profesori pentru a corecta temele și proiectele legate de programare, implementate de către elevi/studenți.

De asemenea, ne dorim să dezvoltăm o aplicație care să permită utilizatorilor să verifice și să compare diferite tipuri de fișiere sursă, să compileze și să compare rezultatele pe baza criteriilor definite de profesor și să ofere funcționalități de administrare și gestionare a utilizatorilor.

Aplicația propusă vine să rezolve dificultățile întâlnite în procesul de evaluare și comparare a codului sursă, oferind un instrument puternic și ușor de utilizat pentru profesori și elevi/studenți.

### Lista definițiilor

* HW – HARDWARE
* SW – SOFTWARE
* RAM – RANDOM ACCES MEMORY
* UI – USER INTERFACE

**1.4. Structura DCS**

Introducere: Cuprinde o prezentare generală a proiectului și a scopului său, evidențiind necesitatea și beneficiile aplicației "CheckerApp".

Descrierea generală a produsului software: Oferă o imagine de ansamblu a aplicației, inclusiv contextul în care este utilizată și funcționalitățile sale principale.

Detalierea platformei HW/SW: Specifică cerințele hardware și software pentru utilizarea aplicației, inclusiv procesorul, memoria RAM, mediul de dezvoltare, framework-ul UI și sistemul de operare.

Constrângeri: Enumeră diversele constrângeri cu care se confruntă dezvoltarea și implementarea aplicației, precum compatibilitatea cu diferite platforme sau actualizările și mentenanța sistemului.

Descrierea cerințelor specifice: Împarte cerințele în două categorii principale: funcționale și nefuncționale.

Cerințele funcționale: Detaliază comportamentul și funcționalitățile aplicației, inclusiv autentificarea, verificarea și compararea fișierelor sursă, administrarea utilizatorilor etc.

Cerințele nefuncționale: Evidențiază aspectele legate de performanță, scalabilitate, ușurința de utilizare, eficiența resurselor etc.

Descrierea modelului logic al sistemului: Detaliază modul în care aplicația este structurată și cum funcționează logic, inclusiv componente precum clientul, baza de date și serverul.

Tabel versiuni: Oferă un istoric al modificărilor aduse documentației, evidențiind numărul versiunii și modificările adăugate.

### 2. Descrierea generală a produsului software

**2.1. Descrierea produsului software**

**Context:**

Într-o zi obișnuită de școală, Alex, un elev pasionat de informatică, se pregătea să își verifice temele pentru examenul săptămânal. După ce și-a terminat codul pentru problemele propuse, Alex și-a dat seama că ar fi util să aibă o modalitate de a verifica dacă soluțiile sale sunt corecte.

Așa că, deschizând laptopul, Alex a accesat aplicația de tip checker pe care o descoperise recent. Aplicația avea funcționalități pentru verificarea și compararea fișierelor sursă, compilarea codului.

În primul rând, Alex a încărcat fișierul său sursă în aplicație și a rulat verificarea. Aplicația a identificat rapid eventualele erori de sintaxă și a oferit sugestii pentru corectarea acestora.

Cu ajutorul aplicației de tip checker, Alex a reușit să-și îmbunătățească rapid și eficient codul, eliminând orice greșeli și având încredere că va obține un rezultat bun la examenul de informatică. Profesorul său a fost impresionat de munca sa și de atenția la detalii, iar Alex a obținut o notă excelentă, datorită pregătirii sale atente și folosirii inteligente a tehnologiei.

Profesorul s-a ajutat și el de aplicație, verificând sursele lui Alex cu cele ale altor elevi și urmărind statisticile acestora.

Această aplicație este o platformă pentru verificarea și evaluarea codului sursă scris în diferite limbaje de programare, cum ar fi C și C++. Utilizatorii acestei aplicații pot fi împărțiți în trei roluri principale: student, profesor și administrator.

Funcționalitățile principale ale aplicației includ:

* Verificarea diferitelor tipuri de fișiere sursă și limbaje de programare, cum ar fi C și C++.
* Compararea fișierelor pentru identificarea gradului de similitudine, utilă pentru a preveni plagierea.
* Compilarea fișierelor și compararea rezultatelor pe baza criteriilor definite de către profesor.
* Profesorul are capacitatea de a defini scenarii de verificare și elementele aferente, cum ar fi template-urile pentru rezolvare.
* Administratorul poate gestiona utilizatorii, inclusiv adăugarea sau ștergerea profesorilor și a studenților.
* Generarea de statistici pentru a oferi o vedere de ansamblu asupra performanței studenților și a altor metrici relevante.
* Logarea acțiunilor la nivelul aplicațiilor client și server, pentru a asigura securitatea sistemului.

Această aplicație facilitează procesul de evaluare și feedback pentru studenți și profesori. Ea oferă un set complet de instrumente pentru gestionarea și evaluarea codului sursă într-un mod eficient și structurat.

Aplicația în sine va avea la bază 3 componente:

1. Clientul: aplicație bazată pe o interfață grafică, care permite gestionarea rapidă și eficientă a checker-ului în funcție de utilizatorul logat. Acesta trimite cereri la server în scopul primirii unui răspuns la întrebările puse(query SQL).

2. Baza de date: stochează toate informațiile aferente aplicatiei și trimite la cerere, dacă există, informații referitoare la acestea. De asemenea, ca orice bază de date, implementează operațiile de bază read, update, delete, create.

3. Server: face legătura între client și baza de date, primind cererea de la user și trimițând un răspuns de la baza de date înapoi acestuia.

**2.2. Detalierea platformei HW/SW**

În această secțiune, vom detalia platforma noastră HW/SW, evidențiind specificațiile cheie și interacțiunile dintre componentele hardware și software.

* Procesor: Avem nevoie de un procesor Intel sau AMD care să suporte efectuarea unor sarcini complexe in ceea ce privește analiza și verificarea datelor. Procesorul ar trebui să ofere suficientă putere de calcul pentru a gestiona operațiuni de procesare intensivă în timp real.
* Memorie RAM: Cel puțin 4GB de memorie RAM pentru a susține încărcarea eficientă a datelor și rularea fluentă a algoritmilor de verificare.
* Mediu de dezvoltare: Dezvoltarea aplicației se realizează utilizând Microsoft Visual Studio, care oferă un mediu de dezvoltare puternic și ușor de utilizat pentru dezvoltarea și depanarea aplicațiilor Windows.
* Framework UI: Utilizăm sistemul inter-platformă de dezvoltare Qt pentru a dezvolta interfața grafică prezentată utilizatorului, asigurând o experiență consistentă și plăcută.
* Sistem de operare: Aplicația este dezvoltată pentru platforma sistemului de operare Windows 10 sau versiuni ulterioare, asigurând compatibilitatea cu mediul de utilizare predominant.

**2.3. Constrângeri**

* Resurse hardware limitate: Aplicația fiind dezvoltata pentru platforma sistemului de operare de tip Windows, utilizatorii care rulează alte sisteme de operare ar putea să întălnească dificultăți în a folosi aplicația.
* Actualizări și mentenanță: Odată cu evoluția tehnologiei și a cerințelor utilizatorilor, este posibil să fie necesare actualizări și îmbunătățiri continue ale aplicației.

**3.** **Detalierea cerințelor specifice**

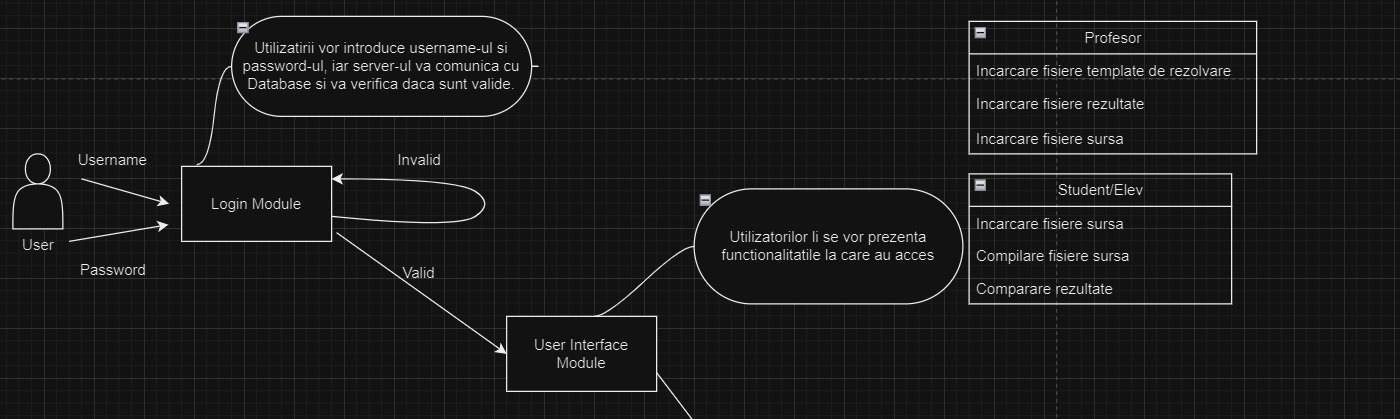
**3.1. Cerințele funcționale**

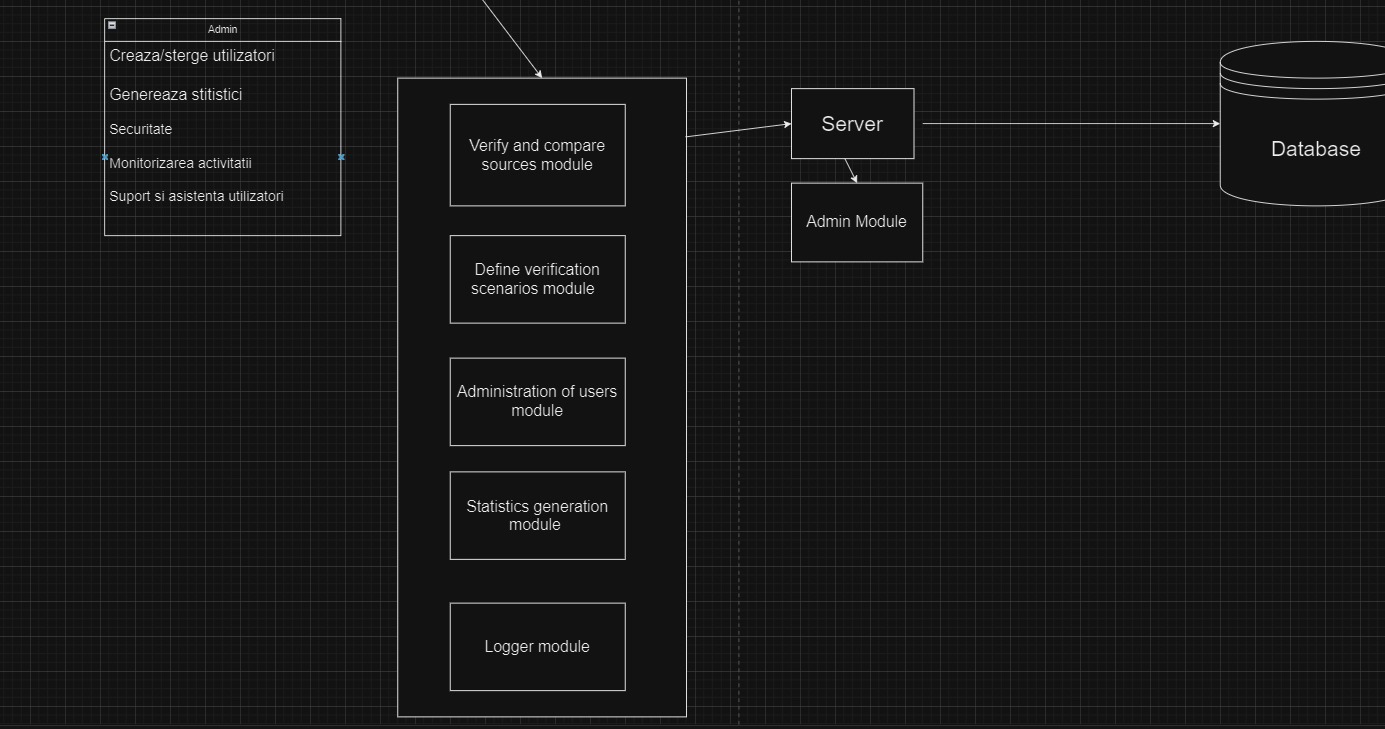
* Autentificare și gestionare a rolurilor: Implementarea unui sistem de autentificare , cu gestionare a rolurilor (admin, student, profesor). Recunoașterea automată a rolului în funcție de datele de autentificare (username și parolă).
* Verificare și comparare a fișierelor sursă: Citirea si interpretarea fișierelor sursă în limbajul de programare specificat.
* Compilare a codului și comparare a rezultatelor: Implementarea unui mediu de compilare pentru fișierele sursă. Compararea rezultatelor obținute cu cele definite de profesor.
* Definirea scenariilor de verificare și a elementelor aferente: Dezvoltarea unei interfețe pentru definirea scenariilor de verificare și a elementelor aferente (template pentru rezolvare, vectorul rezultatelor corecte etc.).
* Administrare a utilizatorilor: Implementarea funcționalităților de adăugare și ștergere a utilizatorilor, disponibile doar pentru administrator (acesta se poate conecta de pe server printr-un username și o parolă).
* Generarea de statistici: Algoritmi pentru generarea de statistici legate de activitatea utilizatorilor și rezultatele obținute.
* Înregistrare a acțiunilor la nivelul aplicațiilor client și server: Implementarea unui sistem de înregistrare a acțiunilor la nivelul aplicațiilor client și server (Logger)

**3.2. Cerințele ne-funcționale**

* Performanța: Aplicația ar trebui să ofere un timp de răspuns rapid, indiferent de numărul de utilizatori sau de volumul datelor procesate.
* Scalabilitatea: Sistemul ar trebui să poată crește în capacitate și să se adapteze la un număr variabil de utilizatori și la volumul crescut de date procesate.
* Ușurința de utilizare: Interfața utilizator ar trebui să fie intuitivă și ușor de înțeles, fără a necesita instruire extensivă pentru utilizatori.
* Eficiența și utilizarea resurselor: Aplicația ar trebui să utilizeze resursele hardware și software în mod eficient, evitând suprautilizarea sau subutilizarea acestora.

## 3.3 Descrierea modelului logic al sistemului





Arhitectura propusă pentru implementare constă într-un sistem client-server, în care serverul va gestiona baza de date și va oferi servicii de verificare și comparare a fișierelor sursă, compilare a codului și generare a statisticilor. Interfața grafică va fi implementată la nivelul aplicației client, care va comunica cu serverul pentru a executa diverse operațiuni.

Modulele componente ale aplicației includ:

a. Modulul de autentificare(login) și gestionare a rolurilor (admin, student, profesor)

b. Modulul de verificare și comparare a fișierelor sursă

c. Modulul de compilare a codului și comparare a rezultatelor

d. Modulul de definire a scenariilor de verificare și a elementelor aferente (template pentru rezolvare, vectorul rezultatelor corecte etc.)

e. Modulul de administrare a utilizatorilor (adăugare, ștergere)

f. Modulul de generare a statisticilor

g. Modulul de înregistrare a acțiunilor la nivelul aplicațiilor client și server