

MINISTERUL EDUCAȚIEI NAȚIONALE



UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

DOCUMENTAȚIE

TEMA PROIECTARE SOFTWARE NUMARUL_1

NUME: Nistor Cristian-Emil
GRUPA: 30231

CUPRINS

1.	Enuntul problemei.....	Error! Bookmark not defined.
2.	Instrumente utilizate.....	Error! Bookmark not defined.
3.	Justificarea limbajului de programare ales	4
4.	Descrierea diagramelor UML	4
5.	Descrierea aplicatiei	6
6.	Concluzii	12

1. Enunțul problemei

Dezvoltati o aplicatie care poate fi utilizata pentru organizarea unui turneu de tenis. Aplicatia va avea 4 tipuri de utilizatori: tenismen, arbitru, organizator turneu si administrator.

Utilizatorii de tip tenismen pot efectua urmatoarele operatii fara autentificare:

- Inscrierea la turneu;
- Vizualizarea programului turneului pe probe;
- Vizualizarea scorului fiecarei partied de tenis;

Utilizatorii de tip arbitru pot efectua urmatoarele operatii dupa autentificare:

- Vizualizarea programului propriu;
- Gestionarea scorului partied pe care o arbitreaza;

Utilizatorii de tip organizator turneu pot efectua urmatoarele operatii dupa autentificare:

- Acceptarea sau respingerea unui tenismen (notificare prin email);
- Operatii CRUD in ceea ce priveste persistenta tenismenilor, arbitrilor si a partidelor de tenis;
- Generarea programului turneului;
- Filtrarea listei de tenismeni dupa diferite criterii;
- Salvare liste filtrate cu informatii despre partide in mai multe formate: csv, json, xml, doc.

Utilizatorii de tip administrator pot efectua urmatoarele operatii dupa autentificare:

- Operatii CRUD pentru informatiile legate de utilizatorii aplicatiei care necesita autentificare;
- Vizualizarea listei tuturor utilizatorilor care necesita autentificare.

2. Instrumente utilizate

Java este un limbaj de programare extrem de popular și robust, cunoscut pentru portabilitatea și securitatea sa încorporate. Furnizează programatorilor o platformă puternică pentru dezvoltarea diverselor tipuri de aplicații software, de la cele enterprise la jocuri și aplicații mobile. Acest limbaj își are originea în C și C++, dar aduce un model de obiecte mai simplu și oferă mai puține facilități la nivel inferior. Un program Java corect scris și compilat poate fi executat fără modificări pe orice platformă care are instalată o mașină virtuală Java (JVM). Acest grad de portabilitate, inexistent în cazul limbajelor mai vechi precum C, este posibil datorită faptului că sursele Java sunt compilate într-un format standard numit cod de octeți (byte-code),

care reprezintă un intermediar între codul mașină (dependent de arhitectura calculatorului) și codul sursă.

MySQL este un sistem de gestionare a bazelor de date relaționale, recunoscut pentru performanța sa excelentă, fiabilitate și un set bogat de funcționalități pentru manipularea datelor. Este deseori alegerea preferată în industrie pentru aplicațiile care necesită gestionarea eficientă a datelor.

Swing este o bibliotecă grafică pentru Java, ce permite dezvoltatorilor să creeze interfețe grafice flexibile și estetice pentru aplicațiile lor desktop. Cu ajutorul Swing, pot fi construite interfețe interactive și atractive, cu suport pentru diverse componente grafice și personalizări.

Aceste instrumente, folosite împreună, constituie un set puternic pentru dezvoltarea aplicațiilor desktop în Java, furnizând dezvoltatorilor o abordare comprehensivă pentru crearea de aplicații sofisticate și eficiente, care să gestioneze datele cu MySQL și să ofere o experiență plăcută utilizatorilor finali prin intermediul interfeței grafice dezvoltate cu Swing.

3. Justificarea limbajului de programare ales

Alegerea limbajului de programare pentru dezvoltarea unei aplicații este esențială pentru succesul proiectului. În cazul meu, am optat pentru Java din mai multe motive bine întemeiate.

În primul rând, Java este un limbaj de programare extrem de popular și robust, cunoscut pentru portabilitatea și securitatea sa încorporate.

În al doilea rând, Java oferă dezvoltatorilor o platformă puternică și bogată în funcționalități pentru crearea de aplicații sofisticate. Aceasta înseamnă că pot să mă concentrez mai mult pe logica aplicației și mai puțin pe gestionarea detaliilor tehnice.

De asemenea, Java este recunoscut pentru performanța și fiabilitatea sa, fiind utilizat frecvent în industrie pentru dezvoltarea unor aplicații critice și de mare scalabilitate. Acest lucru îmi oferă încrederea că aplicația de organizare a turneului de tenis va funcționa eficient și fără erori în timpul desfășurării evenimentului.

În plus, Java este susținut de o comunitate vastă și activă de dezvoltatori, ceea ce înseamnă că avem acces la o gamă largă de resurse, biblioteci și suport tehnic pentru a mă ajuta în procesul de dezvoltare.

4. Descrierea diagramelor UML

Pentru a gestiona eficient cerințele specifice ale aplicației noastre destinate organizării unui turneu de tenis, am conceput o arhitectură clară și bine definită, care să permită fiecărui tip de utilizator să-și îndeplinească sarcinile într-un mod intuitiv și eficient.

Diagrama cazurilor de utilizare ne furnizează o înțelegere clară a acțiunilor pe care fiecare tip de utilizator le poate întreprinde în cadrul aplicației. Fiecare utilizator, fie el tenismen, arbitru, organizator de turneu sau administrator, are atribuite operațiuni specifice, adaptate rolului său în cadrul procesului de organizare a turneului de tenis.

În conformitate cu necesitățile aplicației noastre, fiecare utilizator trebuie să se autentifice inițial pentru a accesa funcționalitățile oferite. După autentificare, fiecare utilizator are acces la un set specific de operațiuni, în funcție de rolul său.

Tenisman: Poate efectua operații precum înscrierea la turneu, vizualizarea programului competiției fără a fi necesară autentificarea și vizualizarea scorului partidelor.

Arbitru: După autentificare, poate vizualiza programul său personal și poate modifica scorul partidelor de tenis arbitrate de el.

Organizator de Turneu: Are privilegii extinse, putând efectua operațiuni CRUD asupra tenismenilor, arbitrilor și partidelor de tenis, precum și generarea programului turneului, filtrarea listei de tenismeni după vârstă și categorie și multe altele.

Administrator: Deține controlul asupra datelor utilizatorilor, putând efectua operații CRUD pentru informațiile acestora și având acces la lista tuturor utilizatorilor care necesită autentificare.

Propunerea de arhitectură pentru aplicația mea este organizată în patru module principale, care sunt evidențiate într-o diagramă de clase: Model, ViewModel, View și Service. Această abordare urmează direcțiile și standardele arhitecturale ale Model-View-ViewModel (MVVM), cu includerea unui modul adițional pentru administrarea serviciilor.

Pachetul Model

Pachetul Model constituie centrul vital al aplicației și își asumă responsabilitatea pentru administrarea datelor și logicii de afaceri. Aici sunt definite clasele și structurile de date care reprezintă obiectele și entitățile esențiale din sfera de aplicare a aplicației. Aceste clase includ, printre altele, entități precum jucatorii de tenis, informațiile despre utilizatori și alte elemente relevante pentru funcționarea corespunzătoare a aplicației. În plus, în acest pachet sunt implementate logica de acces la baza de date, inclusiv operațiile CRUD (Create, Read, Update, Delete) necesare pentru manipularea eficientă a datelor.

Relația cu pachetul Service: Serviciile din pachetul Service utilizează clasele din pachetul Model pentru a accesa și manipula datele.

Pachetul Repository

În cadrul acestui pachet, fiecare entitate sau obiect din structura de date a aplicației este asociat cu un repository dedicat, care furnizează metode specializate pentru realizarea operațiunilor de citire, scriere, actualizare și ștergere în baza de date. Aceste metode abstractizează detaliile specifice ale interacțiunii cu baza de date, permițând o utilizare simplificată și uniformă a acestora. Repository-urile pot beneficia de utilizarea framework-urilor și bibliotecilor specializate, precum Hibernate pentru persistența în JPA (Java Persistence API) sau Spring Data, care simplifică implementarea operațiilor CRUD. Aceste instrumente reduc considerabil complexitatea gestionării bazei de date și contribuie la dezvoltarea rapidă și eficientă a aplicației.

Pachetul SinglePointAcces

Pachetul SinglePointAcces a fost proiectat pentru a oferi un singur punct centralizat de acces către toate funcționalitățile și serviciile furnizate de aplicație. Acest pachet acționează ca un strat de abstractizare între interfețele utilizatorului și logica de afaceri, facilitând modularitatea și reutilizarea codului. Aici sunt definite clasele și metodele care expun funcționalitățile cheie ale aplicației și gestionează interacțiunile cu celelalte pachete.

Pachetul ViewModel

Pachetul ModelView conține logica de prezentare a datelor și coordonarea interacțiunii dintre Model și View. Aici sunt găsite clasele care procesează cererile utilizatorului, interacționează cu datele și actualizează interfața grafică în consecință. Principalul obiectiv al ViewModel-ului este de a menține o separare clară între logica de business și interfața grafică.

Pachetul View

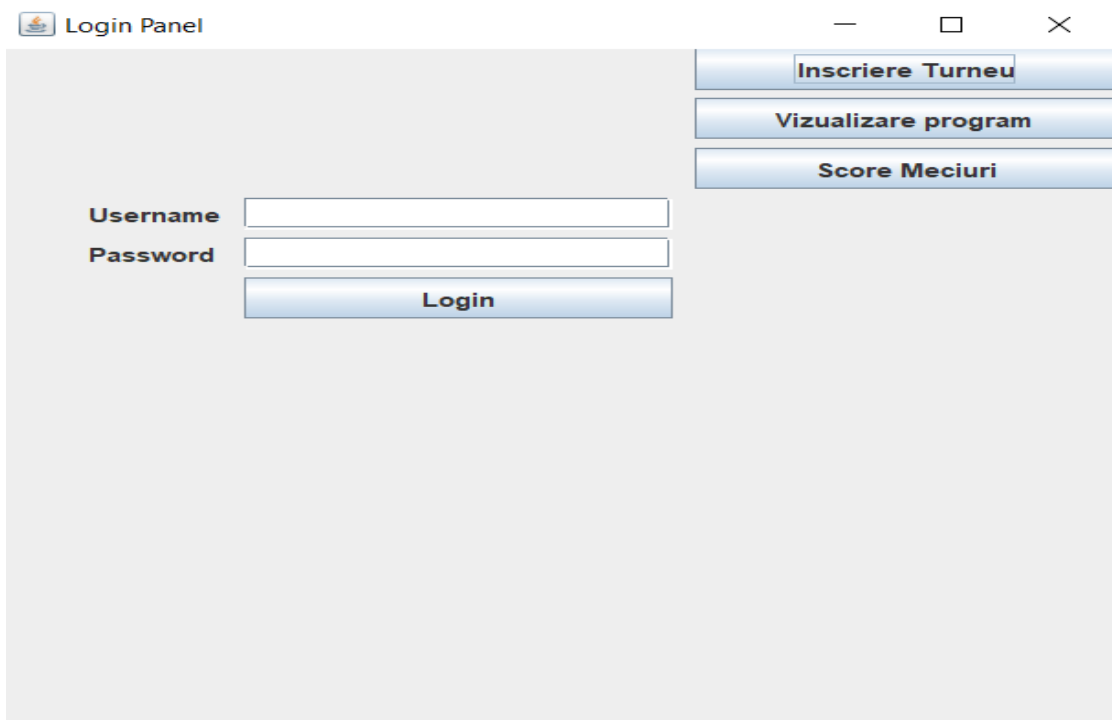
Pachetul View cuprinde componentele grafice folosite pentru comunicarea cu utilizatorul. Aici sunt definite elementele precum ferestrele, panourile, butoanele, etc., esențiale pentru configurarea aspectului interfeței grafice a aplicației. Este crucial ca acest pachet să se concentreze exclusiv pe aspectul vizual și pe interacțiunea cu utilizatorul, fără a include logica de afaceri sau de manipulare a datelor. În legătură cu pachetul ViewModel: ViewModel-ul interacționează cu view-urile pentru a gestiona evenimentele generate de utilizator și pentru a actualiza interfața grafică în funcție de acțiunile acestuia.

5. Descrierea aplicatiei

Aplicația a fost completată conform cerințelor specificate și voi detalia în continuare modul de utilizare.

La deschiderea aplicației, utilizatorul va întâlni o fereastră de login, unde este necesar să se autentifice folosind username-ul și parola asociate contului său. Dacă datele introduse sunt corecte și se potrivesc cu un utilizator din sistem, autentificarea va fi reușită și utilizatorul va fi redirecționat automat către pagina corespunzătoare rolului său. În caz contrar, dacă informațiile introduse sunt incorecte, va fi afișat un mesaj de eroare pentru a notifica utilizatorul cu privire la această situație.

Pe scurt, odată ce autentificarea este finalizată cu succes, fiecare utilizator va fi direcționat către pagina destinată rolului său în sistem. De exemplu, un administrator va fi redirecționat către pagina de administrare, în timp ce un arbitru va fi trimis către pagina dedicată activităților specifice rolului sau.



The image shows a software window titled "Login Panel". It features a standard Windows-style title bar with minimize, maximize, and close buttons. The window's content is split into two main areas. The left area contains a login form with two text input fields labeled "Username" and "Password", and a "Login" button positioned below them. The right area contains a vertical column of three buttons: "Inscriere Turneu", "Vizualizare program", and "Score Meciuri".

Jucatorii de tenis nu au nevoie de autentificare, ci pot accesa functionalitatile destinate lor prin butoanele Inscriere Turneu, Vizualizare program si Scor Meciuri.

În pagina destinată jucatorilor de tennis, fiecare jucator de tenis poate sa se inscrie la turenu dupa ce isi completeaza numele, prenumele, varsta si categoria la care se inscrie, iar dupa ce in cadrul turneului sunt inscrisi 16 jucatori de tenis, se va genera un program al acestui turneu, iar acesta va putea fi vizualizat de catre fiecare jucator prin intermediul butonului Vizualizare program.

Register Panel

First Name

Last Name

Age

☐ Male

☐ Female

☐ Unde18

Register

Back

Match	Time	First Player	Second Player	Referee
Last 16				
1	10:00	9	22	1
2	10:50	11	13	2
3	11:40	18	7	3
4	12:30	15	16	4
5	13:20	14	17	6
6	14:10	12	10	1
7	15:00	21	8	2
8	15:50	19	20	3
Last 8				
9	16:00	Match Winner 1	Match Winner 2	
10	17:00	Match Winner 3	Match Winner 4	
11	18:00	Match Winner 5	Match Winner 6	
12	19:00	Match Winner 7	Match Winner 8	
Semi Final				
13	20:00	Match Winner 9	Match Winner 10	
14	21:00	Match Winner 11	Match Winner 12	
Final				
15	22:00	Match Winner 13	Match Winner 14	

Nu in ultimul rand, utilizatorul cu rolul de organizator, poate efectua operatii de CRUD pe arbitrii, iar in momentul in care un arbitru este eliminat, tuturor meciurilor de tenis atribuite acelui arbitru li se vor atribui un arbitru random dintre cei care sunt in cadrul turneului.

Referee CRUD Panel

First Name

Last Name

Id

Add Update Delete

Id	First Name	Last Name
1	arbitru1	arbitru
2	arbitru2	arbitru
3	arbitru	arbitru
4	arbitru	arbitru
5	arbitru5	arbitru

Back

Totodata, organizatorul poate efectua operatii de CRUD si pe meciurile de tenis, sau asupra jucatorilor de tenis.

In cadrul ferestrei asociate meciurilor de tenis, se vor afisa 4 tabele, dintre care 2 contin jucatorii de tenis, unul arbitrii, iar celalalt meciurile de tenis asociate turneului. Pentru a adauga un meci de tenis se vor selecta jucatorii si arbitrul dorit.

În concluzie, aplicația Turneu de tenis reprezintă o soluție sofisticată și eficientă pentru administrarea unui turneu de tenis, integrând armonios principiile arhitecturale ale Model View ViewModel pentru a garanta o funcționare solidă și extensibilă. Prin intermediul interfețelor grafice distincte utilizatorii au acces și pot gestiona diversitatea funcționalităților oferite de aplicație, adaptate în funcție de rolurile și responsabilitățile fiecăruia. Implementarea logicilor de afișare, model și prezentare într-un mod modular și coerent asigură o separare eficientă a sarcinilor și facilitează dezvoltarea, întreținerea și extinderea pe termen lung a aplicației.