**UNIVERSITATEA NAȚIONALĂ DE ȘTIINȚĂ ȘI TEHNOLOGIE POLITEHNICA BUCUREȘTI**

Facultatea de Electronică, Telecomunicații și Tehnologia Informației

**Proiect**

**Programarea Interfețelor pentru Baze de Date**

Student: Cernat Alexandru Florin

Grupa: 431D

Profesor coordonator:

Ș.l. Dr. Ing. Pupezescu Valentin

București 2024

**Cuprins**

[Cerințe 3](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264816)

[Tehnologii utilizate 3](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264817)

* [MySQL 3](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264818)
* [Java Spring Boot 3](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264819)
* [Thymeleaf 4](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264820)
* [Apache Tomcat 4](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264820)
* [HTML 4](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264821)

[Descrierea aplicației 5](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264822)

* [Baza de date 5](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264823)
* [Diagrama ERD 6](file:///C:\Users\iWar10\AppData\Local\Microsoft\Windows\INetCache\IE\JH14VKES\Gheorghiu_Mihai_SpringBoot%5b1%5d.docx#_Toc158264824)
* Implementarea aplicaţiei 7
* Implementarea HTML 10

Concluzii ..............................................................................................................................12

Bibliografie ..........................................................................................................................12

**Cerințe**: Creaţi două aplicaţii care să conţină o bază de date creată în sistemul de gestiune a bazelor de date MySql şi două interfeţe la aceasta (baza de date este comună).

La crearea interfeţelor se vor folosi două tehnologii (la alegere - ex.: JSP, Hibernate, JPA, .NET, Python, JavaFX, Spring Boot etc.). Baza de date va fi compusă din tabelele: Pupils şi Disciplines, cu asociere de tip M:N.

Interfeţele vor trebui să permită utilizatorului să execute următoarele operaţii pe toate tabele (inclusiv pe cele de legătura dacă aveti asocieri de tip M:N): vizualizare, adăugare, modificare şi ştergere de date. Vizualizarea tabelelor de legătură va presupune vizualizarea datelor referite din celelalte tabele.

Pentru tema primită, se aleg tehnologiile diferite JSP și Java Spring Boot. În prezentul document, se va face referire la tehnologia JSP.

## **Tehnologii utilizate:**

1.MySQL este un sistem de gestionare a bazelor de date relaționale, utilizat frecvent în dezvoltarea aplicațiilor web și a altor soluții software. Acesta permite stocarea și administrarea eficientă a datelor, cum ar fi cele referitoare la angajați, produse, vânzări sau inventar. La fel ca alte sisteme de baze de date, precum Microsoft Access, MySQL oferă funcționalități esențiale pentru organizarea și interogarea datelor. Totuși, MySQL este preferat pentru proiecte de mare amploare datorită capacității sale de scalare și performanței ridicate, în timp ce Microsoft Access este mai potrivit pentru aplicații de dimensiuni mici și medii datorită ușurinței în utilizare.1]

2.JavaServer Pages (JSP) este o tehnologie care le permite dezvoltatorilor să creeze pagini web dinamice prin combinarea codului Java cu limbaje de marcare precum HTML sau XML. Paginile JSP includ atât elemente statice (HTML, XML), cât și componente dinamice scrise în Java, generând astfel conținut personalizat pe server înainte de a fi trimis către utilizator. Această metodă asigură portabilitate, scalabilitate și eficiență, fiind o alegere populară pentru dezvoltatorii care construiesc aplicații web complexe și adaptabile. [2]

3.Apache Tomcat este un server web open-source și un container de servlet-uri Java dezvoltat de Apache Software Foundation. Acesta furnizează un mediu de rulare pentru aplicații web Java, permițând implementarea și gestionarea lor într-un mod eficient și scalabil.

Tomcat acționează ca un server web sau un mediu de execuție pentru aplicațiile Java, gestionând cererile HTTP de la clienți și facilitând executarea servlet-urilor și JSP-urilor pentru generarea răspunsurilor dinamice.[3]

**4.**HyperText Markup Language (HTML) este limbajul standard pentru crearea și structurarea paginilor web. Prin utilizarea unei varietăți de elemente și atribute, HTML permite dezvoltatorilor să definească structura conținutului, incluzând paragrafe, titluri, linkuri și imagini. Această structură semantică este esențială nu doar pentru prezentarea vizuală și organizarea conținutului în browser, ci și pentru accesibilitate și optimizarea pentru motoarele de căutare. HTML funcționează în strânsă legătură cu CSS pentru stilizare și JavaScript pentru funcționalități interactive, formând împreună fundamentul dezvoltării web moderne. Această combinație permite crearea de site-uri web interactive și atractive.[4]

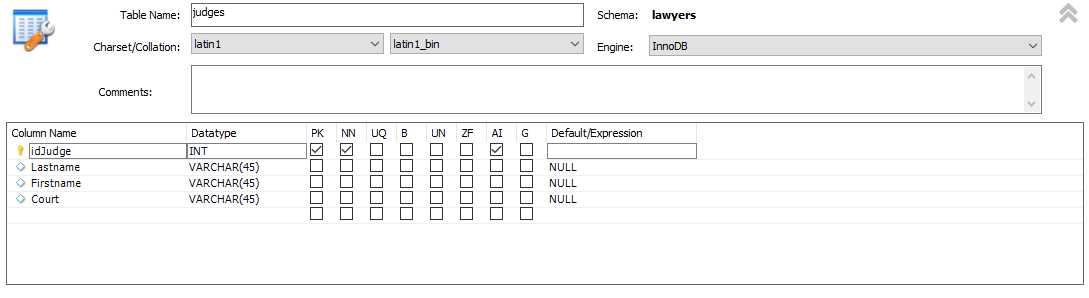
## **Descrierea aplicației:**

Tema dată presupune asocierea M:N a două tabele, „Judges” și „Trials”.

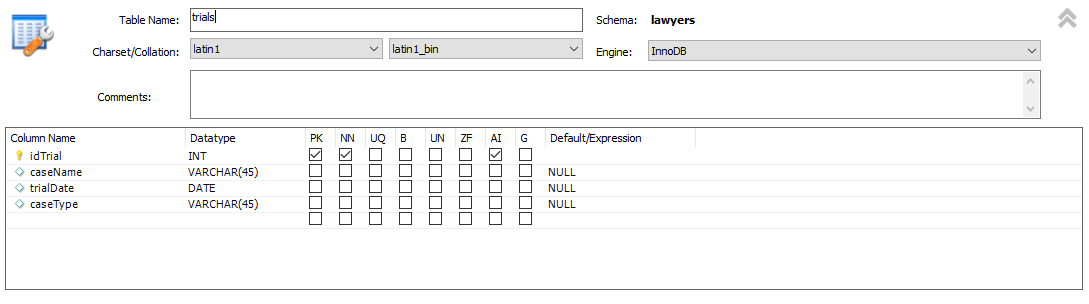
Asocierea M:N (Many-to-Many) se caracterizează prin faptul că fiecărui element dintr-o tabelă îi pot fi asociate mai multe elemente din cealaltă tabelă și invers.

În acest caz, unui „judge” îi pot fi asociate mai multe „trials”, la fel cum și unui „trial” îi pot fi asociați mai mulți judecători.

În cele ce urmează, se vor prezenta relațiile dintre aceste entități, împreună cu atributele specifice fiecăreia.  
**Tabel 1 - Judges:**

**

**Tabel 2 - Trials**

**

Pentru o asociere M:N este necesară crearea unei relații de legătură, în acest caz, „verdicts”, o a treia tabelă de joncțiune. Scopul acesteia este de a diviza asocierea many-to-many în două asocieri de tip 1:N.

În această nouă tabelă, atributele ce au fost selectate drept chei primare pentru tabelele de mai sus, vor deveni chei străine pentru tabela de legătură.

**Tabel 3 – Verdicts:**

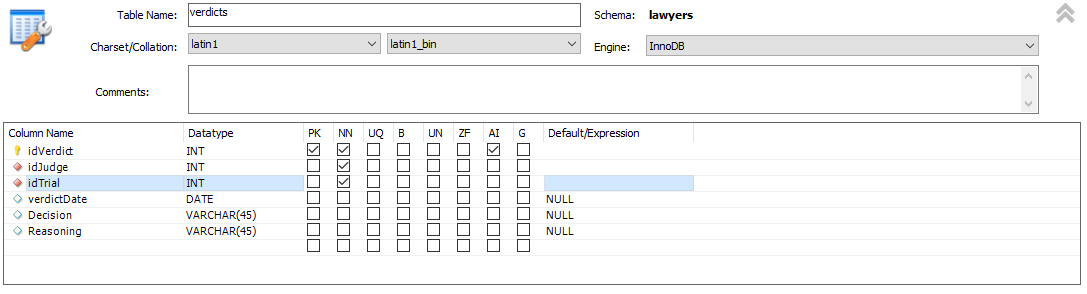


Diagrama ERD oferă o reprezentare vizuală compleza care iltrusrează entitățile, relațiile și contrângerile acestora, oferint o perspectivă detaliată a structurii și interconexiunilor din cadrul bazei de date.

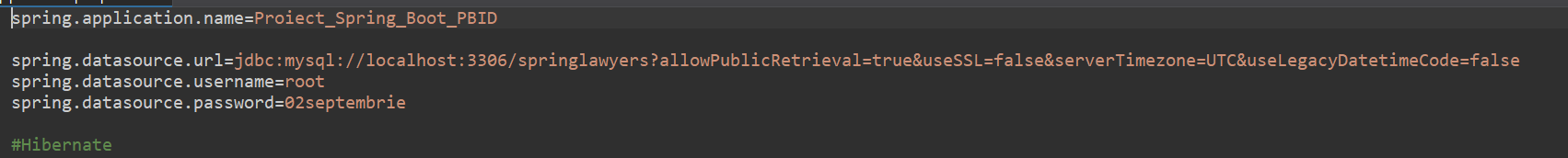
Asocierile dintre tabele:

* Între „judges” și „trials” este o asociere M:N;
* Între „judges” și „verdicts” este o asociere 1:N;
* Între „trials” și „verdicts” este o asociere 1:N.

**Implementarea aplicaţiei:**

**Iniţializare şi conectare la baza de date**

Conexiunea la baza de date din MySQL se realizează în fisierul application.properties, folosind credențialele specificate.

****

O aplicație Spring Boot bazată pe arhitectura **MVC (Model-View-Controller)** este formată din mai multe pachete, fiecare având clasele pentru cele 3 tabele, fiecare cu funcţionalităţile ei.

**Demo.controller**

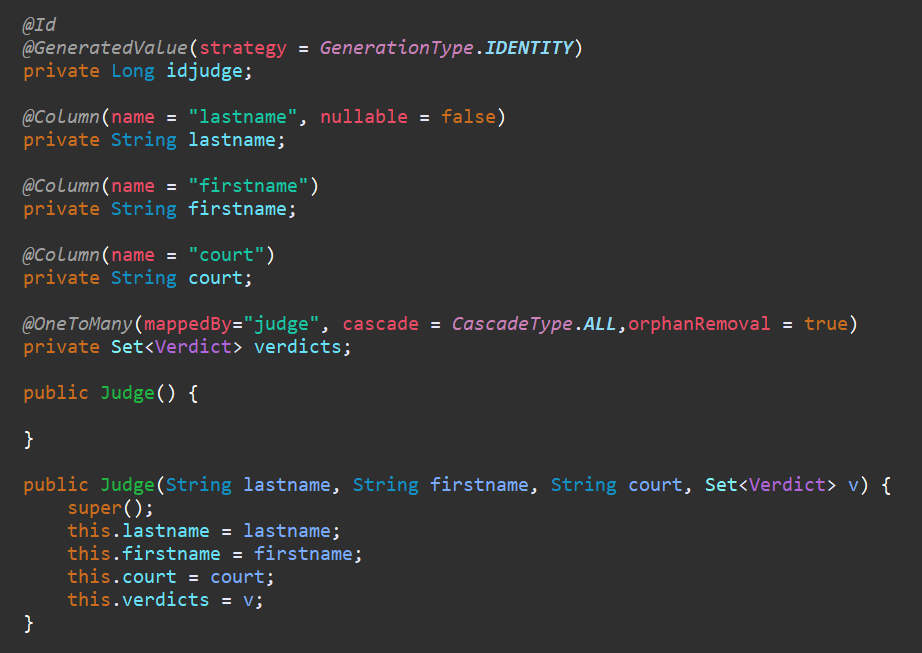
Controller-ul este responsabil pentru gestionarea cererilor HTTP și maparea acestora la metodele corespunzătoare din aplicație. Controller-ul primește cererile de la client, procesează datele și returnează răspunsurile adecvate. De exemplu, PupilController gestionează cererile legate de entitatea Pupil, cum ar fi crearea, actualizarea, ștergerea sau obținerea informațiilor despre elevi.





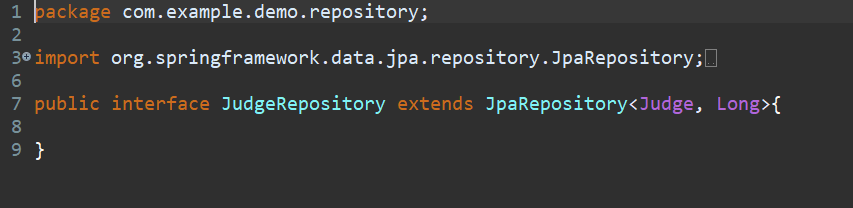
**Demo.entity**

Clasele din pachetul demo.entity reprezintă entitățile de bază de date ale aplicației tale. Fiecare clasă din acest pachet corespunde unei tabele din baza de date și conține atributele care vor deveni coloane ale respectivei tabele.



**Demo.repository**

Repository-ul este responsabil pentru interacțiunea cu baza de date. Acesta definește metode pentru operațiuni CRUD (Create, Read, Update, Delete) și alte operațiuni specifice pe entități. Repository-urile sunt de obicei interfețe care extind alte interfețe de la Spring Data JPA, cum ar fi JpaRepository.



**Demo.service**

Service-ul definește logica aplicației. Acesta conține metode care implementează funcționalitățile necesare pentru a îndeplini cerințele aplicației. Service-ul este un strat intermediar între controller și repository, asigurând separarea logicii de afaceri de logica de prezentare și de acces la date.



**Demo.service.impl**

Service.Impl este implementarea concretă a interfeței Service. Acesta implementează metodele definite în interfața Service și utilizează repository-urile pentru a accesa datele din baza de date.

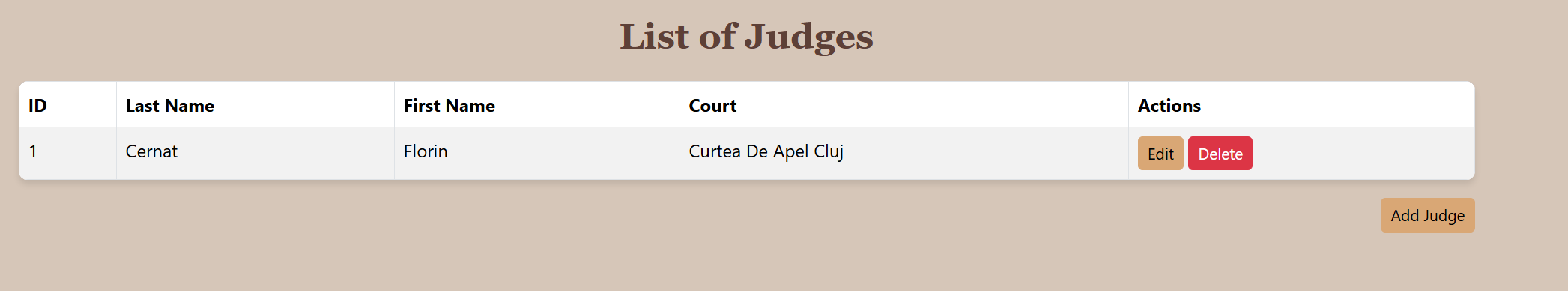


**3.4 Implementarea HTML**

**Afişarea datelor**

Următorul fragment de cod este folosit pentru a afișa datele referitoare la ”judge”.

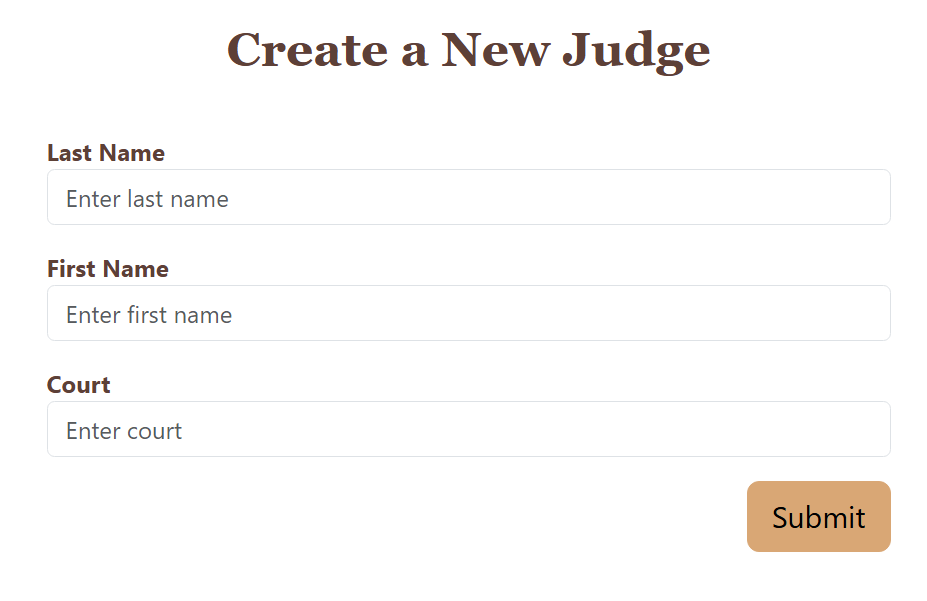




**Adăugarea datelor**

Adăugarea se realizează prin apăsarea butonului „Add” din pagina web. Acest fragment de cod se ocupă cu realizarea formularului ce trebuie completat.

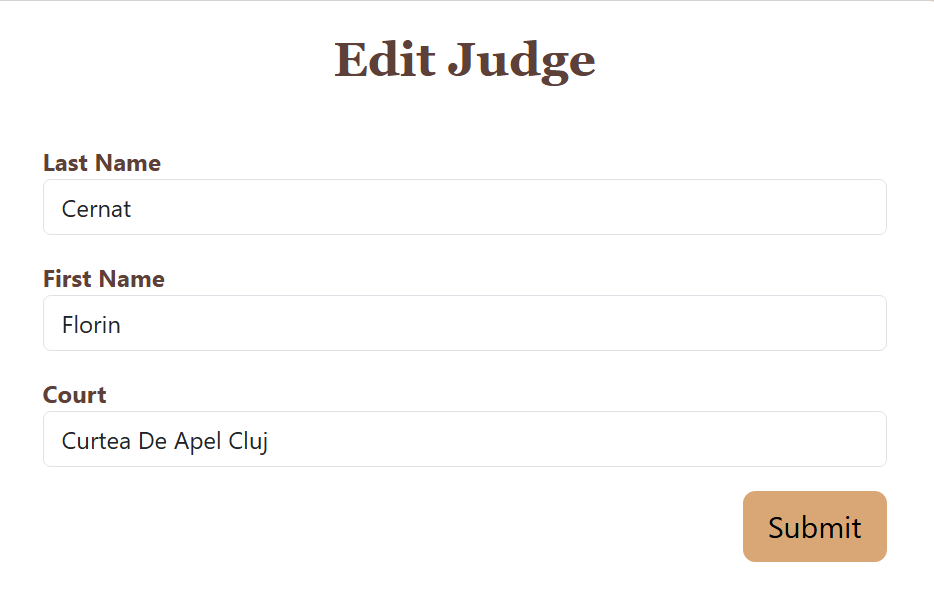




**Modificarea/ştergerea datelor**

Modificarea/ştergerea datelor se realizează prin apăsarea butonului de „Edit”/ „Delete” din pagina de vizualizare a tabelelor. Acest fragment de cod se ocupă cu realizarea formularului de editare a datelor deja existente.





**Concluzii:**

Aplicația, prin integrarea operațiunilor esențiale CRUD (creare, citire, actualizare, ștergere), oferă un mod simplu și eficient de a gestiona clienții, de a organiza tipurile de asigurări și de a clasifica contractele. Aceasta reprezintă o soluție optimă pentru industria asigurărilor, contribuind la o administrare eficientă a datelor.  
Prin designul său simplificat și orientat pe funcționalitate, aplicația permite utilizatorilor să gestioneze și să acceseze baza de date cu ușurință, devenind un instrument indispensabil pentru agenții de asigurări.  
  
  
**Bibliografie:**

1. Spring.io (https://spring.io/projects/spring-boot)

2. Oracle Java Server Pages Technology (https://www.oracle.com/java/technologies/jspt.html)

3. Thymeleaf (https://www.thymeleaf.org/)

4. Tomcat (https://tomcat.apache.org/)

5. FreeCodeCamp (https://www.freecodecamp.org/news/html-css-and-javascript-explained-for-beginners)