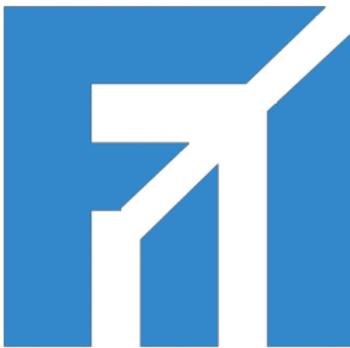


UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IASI

FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

FII Pregătit

propusă de

Rares-Vasilică Dascălu

Sesiunea: iulie, 2024

Coordonator științific

Drd. Costandache Mihai-Andrei

UNIVERSITATEA "ALEXANDRU IOAN CUZA" DIN IASI

FACULTATEA DE INFORMATICA

FII Pregătit

Rares-Vasilică Dascălu

Sesiunea: iulie, 2024

Coordonator științific

Drd. Costandache Mihai-Andrei

Avizat,

Îndrumător lucrare de licență,

Drd. Costandache Mihai-Andrei.

Data: Semnătura:

DECLARAȚIE privind autenticitatea conținutului lucrării de licență

Subsemnatul **Dascălu Rareș-Vasilică** domiciliat în **România, jud. Vaslui, com. Dănești, sat Emil Racoviță**, născut la data de **12 septembrie 2002**, identificat prin CNP **5020912374511**, absolvent al **Universității „Alexandru Ioan Cuza” din Iași**, Facultatea de **informatică** specializarea **informatică**, promoția **2024**, declar pe propria răspundere, cunoscând consecințele falsului în declarații în sensul art. 326 din Noul Cod Penal și dispozițiile Legii Educației Naționale nr. 1/2011 art.143 al. 4 si 5 referitoare la plagiat, că lucrarea de licență cu titlul **FII Pregătit** elaborată sub îndrumarea domnului **Drd. Costandache Mihai-Andrei**, este autentică, îmi aparține și îmi asum conținutul său în întregime.

De asemenea, declar că sunt de acord ca lucrarea mea de licență să fie verificată prin orice modalitate legală pentru confirmarea autenticității, consumând inclusiv la introducerea conținutului său într-o bază de date în acest scop. Declar că lucrarea de față are exact același conținut cu lucrarea în format electronic pe care profesorul îndrumător a verificat-o prin intermediul software-ului de detectare a plagiului.

Am luat la cunoștință despre faptul că este interzisă comercializarea de lucrări științifice în vederea facilitării falsificării de către cumpărător a calității de autor al unei lucrări de licență, de diplomă sau de disertație și în acest sens declar pe proprie răspundere că lucrarea de față nu a fost copiată, ci reprezintă rodul cercetării pe care am întreprins-o.

Data:

Semnătura:

DECLARAȚIE DE CONSUMĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „**FII Pregătit**”, codul sursă al programelor și celealte conținuturi (grafice, multimedia, date de test etc.) care însotesc această lucrare să fie utilizate în cadrul Facultății de Informatică.

De asemenea, sunt de acord ca Facultatea de Informatică de la Universitatea „Alexandru Ioan Cuza” din Iași, să utilizeze, modifice, reproducă și să distribuie în scopuri necomerciale programele-calculator, format executabil și sursă, realizate de mine în cadrul prezentei lucrări de licență.

Absolvent **Dascălu Rares-Vasilică**

Iași, data:

Semnătura:

Cuprins

Introducere	3
Motivație	3
Contribuții proprii	4
Structura lucrării	4
1 Aplicații similare	6
1.1 pbinfo	6
1.2 infoarena	7
1.3 Programare cu răbdare	8
1.4 Concluzii	9
2 Tehnologii utilizate	10
2.1 React.js	10
2.2 Node.js	10
2.3 REST API	11
2.4 MongoDB	11
2.5 g++	12
2.6 Docker	12
2.7 Axios	12
2.8 Formik	13
2.9 React Chart.js	13
2.10 React Bootstrap	13
2.11 React Select	13
2.12 React Router DOM	14
2.13 React Ace	14
2.14 Yup	14
2.15 BcryptJS	15

2.16 Body-parser	15
2.17 Express.js	15
2.18 Express-session	16
2.19 Mongoose	16
2.20 Connect-mongodb-session	16
2.21 Cors	16
2.22 Dotenv	17
2.23 NodeMailer	17
2.24 UUID	17
2.25 Concluzii	18
3 Arhitectură și implementare	19
3.1 Procesare răspuns	21
3.2 Creare exercițiu	24
3.3 Generare statistici	25
3.4 Gestionare exerciții	26
3.5 Gestionare utilizatori	27
3.6 Register / Login / Reset password	28
3.7 Concluzii	32
4 Scenarii de utilizare	33
4.1 Scenariul 1	34
4.2 Scenariul 2	35
4.3 Scenariul 3	36
4.4 Scenariul 4	37
4.5 Concluzii	38
Concluzii	39
Concluzii Generale	39
Posibile Îmbunătățiri	40
Bibliografie	41

Introducere

Motivație

Încă din primul an de facultate, am observat că distribuirea temelor către studenți era ineficientă din cauza lipsei de automatizare a acestui proces. De-a lungul celor trei ani, temele au fost în mare parte distribuite prin Discord sau pe site-ul cursurilor, însă corectarea acestora se făcea predominant în sala de clasă. Această metodă nu este eficientă din punct de vedere al timpului, deoarece corectarea temelor pentru o grupă de 20 de studenți durează cel puțin o oră, dacă profesorul alocă aproximativ 3 minute fiecarui student. Acest timp ar putea fi folosit mult mai productiv.

Am resimțit această problemă deoarece, în liceu, eram obișnuit cu o metodă mult mai eficientă de corectare. Se utiliza o platformă online unde temele erau rezolvate și sursele puteau fi compilate direct pe platformă pentru a verifica corectitudinea soluțiilor. Timpul economisit astfel era dedicat discuțiilor pe marginea soluțiilor trimise, pentru a descoperi diverse metode de rezolvare și pentru a ne dezvolta gândirea.

Această metodă s-a dovedit extrem de eficientă în liceu, însă, din păcate, nu există o platformă similară pentru mediul universitar. Majoritatea platformelor de acest tip sunt axate pe probleme din materia de liceu, dar nu văd de ce nu ar putea fi creată una adaptată pentru universitate. O astfel de platformă ar fi benefică atât pentru studenți, oferindu-le statistici detaliate asupra punctelor slabe și aspectelor la care trebuie să lucreze, cât și pentru profesori, economisind timpul de corectare și oferindu-le informații despre zonele în care studenții întâmpină dificultăți, astfel încât să își poată ajusta predarea în consecință.

Contribuții proprii

Pentru realizarea lucrării de licență, am dezvoltat o serie de componente cheie integrate într-o aplicație web complexă. Scopul acestei aplicații este de a facilita distribuirea și corectarea temelor pentru studenții facultății de informatică.

Componenta principală a aplicației este metoda de corectare a fișierelor sursă primite. Această metodă rulează fișierele sursă pe 10 cazuri diferite de input-output și generează un punctaj bazat pe rata de succes a codului sursă.

În ceea ce privește backend-ul, am dezvoltat o aplicație Web API responsabilă de gestionarea codurilor sursă trimise de utilizatori și de interacțiunea cu baza de date nonrelatională. În această bază de date sunt stocate sursele cu scorurile obținute, datele despre utilizatori și detaliile exercițiilor.

Ultima parte a lucrării este reprezentată de componenta vizuală: am creat o aplicație cu o interfață intuitivă și un design adaptabil la rezoluția ecranului. Funcționalitățile de autentificare permit catalogarea utilizatorilor în două categorii: administratori (profesori) și utilizatori obișnuiți (studenți). Utilizatorii obișnuiți pot naviga printre problemele încărcate și pot trimite surse pentru evaluare, având la dispoziție statistică bazată pe rata de succes a exercițiilor, care îi ajută să își identifice punctele slabe și punctele forte. Utilizatorii cu drept de administrator pot, pe lângă funcționalitățile disponibile utilizatorilor obișnuiți, să creeze noi exerciții, să gestioneze lista de utilizatori și să vadă rata de reușită a studentilor pentru fiecare exercițiu.

Structura lucrării

Această lucrare este structurată în patru capitole ce prezintă punctele esențiale ale acestei aplicații cât și pașii parcursi în crearea acesteia.

Capitolul I - oferă o scurtă prezentare a unor aplicații similare, evidențiind punctele lor forte și aspectele care ar putea fi adăugate sau îmbunătățite. Analizând aceste aplicații, se obțin perspective valoroase care ajută la conturarea unor direcții de dezvoltare pentru aplicația noastră.

Capitolul II - detaliază tehnologiile utilizate și modul în care acestea au fost implementate pentru asigurarea unei funcționări eficiente a site-ului. Printre aceste tehnologii se numără React.js pentru interfața clientului, Node.js pentru partea de server și MongoDB pentru stocarea datelor necesare aplicației. Fiecare tehnologie este explicată

în contextul specific al proiectului, subliniind contribuția sa la funcționalitatea generală a aplicației.

Capitolul III - se subliniază arhitectura aplicației și funcționalitățile principale ale acesteia. În acest capitol sunt descrise componentele site-ului și modul în care acestea sunt integrate pentru a asigura o funcționare optimă. Se vor sublinia punctele forte ale arhitecturii alese și se va argumenta alegerea fiecărei tehnologii utilizate.

Capitolul IV - prezintă scenariile de utilizare ale site-ului FII Pregătit, detaliind toate funcționalitățile prin intermediul unor imagini sugestive. Acest capitol oferă o imagine clară a modului în care utilizatorii interacționează cu aplicația și a beneficiilor pe care aceasta le oferă.

Ultima secțiune a acestei lucrări se concentrează pe concluzii și pe modalitățile de îmbunătățire și dezvoltare a aplicației.

Capitolul 1

Aplicații similare

În acest capitol vor fi prezentate câteva site-uri similare care facilitează un mod mai eficient de asignare și testare a temelor, dar care sunt orientate către mediul preuniversitar.

1.1 pbinfo

Pbinfo [1] este un portal web dedicat elevilor de liceu, oferind o gamă variată de probleme de informatică cu evaluare automată. Utilizatorii pot aborda aceste probleme utilizând opt limbaje de programare diferite, incluzând C++, C, Pascal, C#, PHP, Java, Python 2.7 și Python 3.4.

Platforma permite profesorilor să creeze clase virtuale de elevi și să le aloce teme, facilitând astfel organizarea și distribuirea sarcinilor educaționale. Pbinfo include și un mod special de examen, care restricționează accesul elevilor la soluțiile anterioare și impune un timp limitat pentru trimiterea răspunsurilor la problemele stabilite pentru evaluare.

În plus, site-ul oferă o secțiune bogată de teorie, unde elevii pot învăța și aprofunda noțiunile necesare conform curriculumului liceal.

Un alt aspect benefic al platformei este sistemul de statistici, care arată eficiența rezolvărilor, oferind o imagine asupra performanțelor generale. Cu toate acestea, site-ul nu dispune de statistici detaliate pe categorii, ceea ce ar putea ajuta elevii să identifice mai clar domeniile în care au nevoie de îmbunătățiri.

Deși Pbinfo este extrem de apreciat și folosit în rândul elevilor de liceu pentru pregătirea în informatică, nu pune la dispoziție resurse pentru studentii universitari,

limitându-se la nivelul preuniversitar.

În Figura 1 este ilustrată interfața paginii de exerciții a site-ului pbinfo.ro, prezintând aspectul pe care utilizatorul îl vede atunci când dorește să încarce o sursă pentru un exercițiu în vederea procesării.

The figure consists of two screenshots of the pbinfo.ro website. The left screenshot shows a problem submission details page. It includes a header with navigation links like 'Probleme', 'Soluții', and 'Resurse'. Below this is a table with columns: ID, Utilizator, Problema, Data încărcării, Tip, Stare, and Scor. A row is selected for user #5665410, DascaluRares (DascaluRares129), with the problem sum00, date 2017-09-19 14:40:14, C++, Evaluare finalizată, and a score of 100. Below the table are tabs for 'Enunț', 'Indicații', 'Teste de evaluare', 'Blockly', 'Soluția oficială' (with a count of 10), and 'Soluții'. The main content area contains sections for 'Cerință' (Problem statement), 'Date de intrare' (Input data), 'Date de ieșire' (Output data), 'Restricții și precizări' (Constraints and precision), and 'Exemplu:' (Example). The example shows input '12 23' and output '35'. The right screenshot shows the 'Încărcare soluție' (Upload solution) interface. It has a text area labeled 'Lipăsește codul aici' (Paste the code here) with the instruction 'Alege limbajul de programare' (Select the programming language) set to C++. A code snippet for summing two integers is pasted into the text area. A blue button at the bottom right says 'Adaugă soluția' (Add solution).

Figura 1: Pbinfo

1.2 infoarena

Infoarena [2] este un site cu funcționalități similare celor de pe pbinfo, oferind acces la o arhivă vastă de probleme, materiale educaționale pentru programa de liceu, și o arhivă de concursuri. Pe Infoarena, utilizatorii pot compila surse în diverse limbi de programare, inclusiv C++, C, FreePascal, Python3, Rust și Java.

Un aspect particular al Infoarena este modalitatea de încărcare a codului: utilizatorii trebuie să atașeze fișierul cu codul sursă dorit, să aleagă limbajul de programare corespunzător, și apoi să trimită codul spre evaluare, spre deosebire de alte platforme unde codul este introdus direct într-un editor online.

Desi prezintă similitudini cu pbinfo, Infoarena pune un accent mai mare pe pregătirea elevilor pentru olimpiadele de informatică și alte concursuri de profil, propunând o gamă largă de probleme de o complexitate mai ridicată. Platforma oferă, de asemenea, resurse valoroase pentru elevii care aspiră la perfectionarea tehniciilor de programare și dezvoltarea competențelor necesare pentru competițiile de top.

Este de remarcat faptul că, spre deosebire de alte platforme, Infoarena nu furnizează statistici privind rata de succes a surselor evaluate, ceea ce constituie o diferență notabilă și ar putea influența experiența utilizatorilor care doresc să-și monitorizeze progresul în mod continuu. Această caracteristică face ca Infoarena să fie mai puțin orientată spre feedback instant, concentrându-se mai mult pe dezvoltarea pe termen lung a abilităților de programare.

În Figura 2 este prezentată interfața grafică a site-ului infoarena.ro, mai exact pagina pe care utilizatorul o vede atunci când dorește să acceseze o problemă pusă la dispoziție.

The screenshot shows the Infoarena website interface. At the top, there's a navigation bar with links for Home, Arhiva de probleme, Arhiva educatională, Arhiva monthly, Arhiva ACM, Concursuri, Concursuri virtuale, Clasament, Articole, Downloads, Links, Documentație, Despre Infoarena, Monitorul de evaluare, Trimite soluții, Contul meu, and Căutare. Below the navigation bar, there's a sidebar with 'In curand...', '166418 membri înregistrati', and '14:50:49'. The main content area shows a recent submission for problem 'adunare.in'. The submission details are as follows:

Fisierul intrare/iesire:	adunare.in, adunare.out	Sursă:	Info-arena 1.0
Author:	Din Fodor	Adăugată de:	
Timp execuție pe test:	0.025 sec	Limită de memorie:	65536 kbytes
Scorul tău:	N/A	Dificultate:	★☆☆☆☆

Below the table, there are sections for 'A+B', 'Adunare simplă', 'Date de intrare', 'Date de ieșire', 'Exemplu', and 'Indicii de rezolvare'. There's also a section for 'Vizualizare soluție trimisă' with a 'Fisier' dropdown set to 'Choose File' and a 'Concurs' dropdown set to 'Alegeți runda'. A note at the bottom says: 'Acestă problemă face parte din mai multe concursuri. Selectaază-l pe cel la care participă'.

Figura 2: Infoarena

1.3 Programare cu răbdare

Programare cu Răbdare [3] este un alt site educativ dedicat elevilor de liceu, oferind o varietate de probleme și resurse aliniate la programa școlară. Deși nu diferă semnificativ de alte platforme similare, acest site se distinge prin capacitatea de a compila cod nu doar în limbajele Python3, Pascal, C++ și C, ci și în pseudocod. Aceasta este o caracteristică unică care permite elevilor să experimenteze cu soluții algoritmice într-un format mai accesibil și mai apropiat de logica matematică, fără a fi constrânsi de sintaxa strictă a unui limbaj de programare.

Utilizatorii pot trimite soluțiile fie prin copierea directă într-un câmp text pe site, fie prin încărcarea fisierului sursă. Programare cu Răbdare oferă statistici pentru fiecare problemă individual, generând date despre căte surse au fost necesare pentru a ajunge la

o sursă cu punctaj maxim. Totuși, aceste statistici nu sunt detaliate pe categorii, ceea ce ar putea ajuta elevii să identifice mai precis zonele în care trebuie să își îmbunătățească cunoștințele.

Cu toate acestea, la fel ca și alte platforme menționate anterior, site-ul nu oferă resurse sau suport pentru studenții universitari, fiind conceput exclusiv pentru nevoile și obiectivele elevilor de liceu.

Observăm în *Figura 3* pagina pe care site-ul Programare cu Răbdare o pune la dispoziție utilizatorului în momentul în care acesta dorește accesarea unui exercițiu în vederea rezolvării sale.

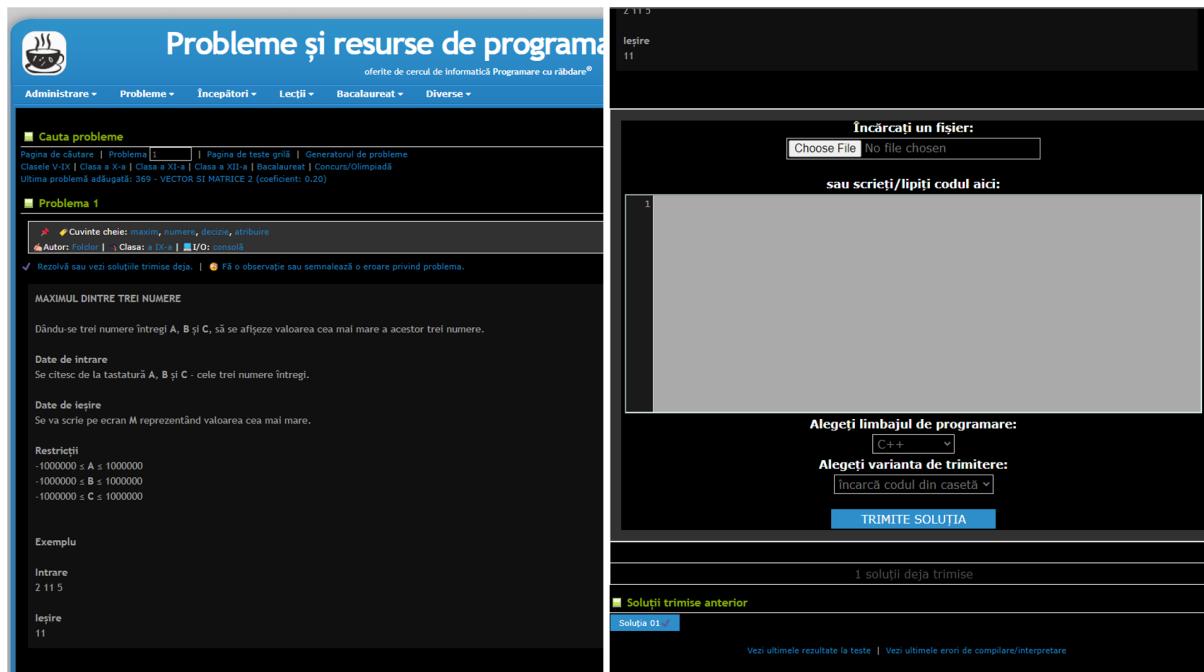


Figura 3: Programare cu răbdare

1.4 Concluzii

Capitolul 1 descrie aplicații similare care au inspirat dezvoltarea aplicației FII Pregătit, extinzând funcționalitățile existente cu caracteristici unice. Deși toate aplicațiile au trăsături comune, fiecare aduce implementări distincte, oferind un mediu de utilizare diferit.

Capitolul 2

Tehnologii utilizate

În acest capitol, vom prezenta tehnologiile utilizate în cadrul aplicației FII Pregătit, precum și numeroasele librării care au contribuit la buna funcționare a site-ului.

2.1 React.js

React.js [4] este o bibliotecă open-source bazată pe JavaScript, creată de Meta (fostul Facebook), care simplifică procesul complex de creare a frontend-ului pentru interacțiunea cu utilizatorul.

Deoarece React.js se ocupă doar de interfața cu utilizatorul și de randarea componentelor în DOM, aplicațiile React.js se bazează adesea pe alte librării pentru rutare și alte funcționalități pe partea de client. Un avantaj cheie al React.js este că evită randarea inutilă a elementelor neschimbate din DOM, ajutând astfel la optimizarea performanței aplicației.

Întreaga interfață a clientului a fost creată folosind React.js datorită flexibilității sale și a numeroaselor librării ajutătoare pe care le oferă.

2.2 Node.js

Node.js [5] este un mediu de rulare JavaScript open-source, popular pentru diverse tipuri de proiecte. Utilizează motorul V8 JavaScript, același motor folosit de Google Chrome, dar în afara browserului, ceea ce îi conferă o performanță ridicată.

O aplicație Node.js rulează într-un singur proces, fără a crea un thread nou pentru fiecare solicitare. Node.js oferă primitive I/O asincrone în librăria sa standard, preve-

nind blocarea codului JavaScript. Majoritatea librăriilor din Node.js sunt scrise folosind paradigmă neblocante, astfel încât blocarea este excepția, nu norma.

Când Node.js efectuează o operațiune I/O, cum ar fi citirea din rețea, accesarea unei baze de date sau a unui sistem de fișiere, nu blochează firul de execuție și nu irosește cicluri CPU așteptând. În schimb, Node.js continuă operațiunile când răspunsul este disponibil.

Această abordare permite lui Node.js să gestioneze mii de conexiuni simultane cu un singur server, fără a complica gestionarea concurenței thread-urilor, care poate fi o sursă majoră de erori.

Node.js este principalul limbaj utilizat în dezvoltarea aplicației, deoarece, fiind bazat pe JavaScript la fel ca React.js, facilitează integrarea cu partea de frontend. În plus, Node.js oferă o multitudine de librării care permit crearea diverselor funcționalități necesare.

2.3 REST API

REST API [6] este un stil arhitectural pentru interfețele de programare a aplicațiilor, care utilizează cereri HTTP pentru a accesa și manipula date. Aceste cereri permit efectuarea operațiunilor GET, PUT, POST și DELETE, adică citirea, actualizarea, crearea și ștergerea resurselor. REST API este folosit pentru a facilita comunicarea dintre server și client, asigurând un mod standardizat și eficient de interacțiune între acestea.

Un avantaj major al REST API este flexibilitatea sa, permitând dezvoltatorilor să construiască aplicații scalabile și extensibile. De asemenea, REST API este independent de tipul de platformă sau limbaj de programare, ceea ce înseamnă că serverul și clientul pot fi dezvoltate folosind tehnologii diferite și totuși pot comunica eficient.

În aplicație, REST API a fost folosit pentru comunicarea între server și client datorită simplității și eficienței sale în gestionarea cererilor și răspunsurilor HTTP.

2.4 MongoDB

MongoDB [7] este un sistem de gestionare a bazelor de date non-relaționale open-source, care utilizează documente flexibile în loc de tabele și rânduri pentru a procesa și stoca diverse tipuri de date.

Fiind o soluție de bază de date NoSQL, MongoDB nu necesită un sistem de gestionare a bazelor de date relationale, oferind un model de stocare a datelor elastic. Acest model permite utilizatorilor să stocheze și să interogheze cu ușurință tipuri de date diverse. Astfel, MongoDB simplifică gestionarea bazelor de date pentru dezvoltatori și creează un mediu extrem de scalabil pentru aplicații și servicii multiplatforme.

MongoDB este baza de date utilizată pentru stocarea datelor necesare aplicației datorită flexibilității sale și a integrării excelente cu Node.js.

2.5 g++

g++ [8] este un compilator pentru limbajul C++ inclus în colecția GNU Compiler Collection (GCC). Aceasta efectuează preprocesarea, compilarea, asamblarea și legarea codului sursă pentru a crea un fișier executabil. Comanda g++ oferă diverse opțiuni care permit oprirea procesului în diferite etape intermediare, asigurând flexibilitate în gestionarea și analizarea fazelor de compilare.

Acest compilator este utilizat pentru a permite aplicației să ruleze și să proceseze coduri sursă scrise în limbajul C++.

2.6 Docker

Docker [9] este o platformă open-source pentru dezvoltarea, livrarea și rularea aplicațiilor. Aceasta oferă capacitatea de a împacheta și rula o aplicație într-un mediu izolat numit container. Containerele asigură izolarea și securitatea necesare pentru a permite rularea simultană a mai multor containere pe aceeași mașină. Fiecare container conține tot ceea ce este necesar pentru a rula aplicația, eliminând astfel dependențele de software-ul instalat pe gazdă.

În aplicația FII Pregătit, Docker a fost utilizat pentru a crea containere în care codul sursă să fie procesat, prevenind astfel eventualele daune cauzate de un cod malitios.

2.7 Axios

Axios [10] este o bibliotecă JavaScript bine-cunoscută, utilizată pentru efectuarea solicitărilor HTTP dintr-un browser web sau Node.js. Aceasta simplifică trimiterea cererilor HTTP asincrone către un server și gestionarea răspunsurilor. Axios oferă

caracteristici precum interceptoare, manipularea antetelor cererilor și răspunsurilor, precum și gestionarea diferitelor formate de date, cum ar fi JSON. Este frecvent folosită în dezvoltarea web pentru obținerea de date de la API-uri și interacțiunea cu serverele.

2.8 Formik

Formik [11] este o bibliotecă open-source cunoscută pentru crearea și gestionarea datelor de formular în aplicațiile React. Aceasta furnizează o soluție intuitivă pentru construirea formularelор în React, având un API simplu și funcții de validare integrate. Formik simplifică colectarea și manipularea datelor de intrare în cadrul aplicațiilor React.

2.9 React Chart.js

React Chart.js [12] este o bibliotecă JavaScript utilizată pentru crearea de diagrame și grafice în aplicațiile React. Aceste grafice pot fi folosite pentru a vizualiza date pe site-uri web. React Chart.js simplifică procesul de creare a diagramelor pentru utilizatori și este adecvat pentru seturi mari de date.

Această librărie a fost utilizată pentru crearea diagramelor care informează utilizatorul despre performanțele sale în urma rezolvării exercițiilor.

2.10 React Bootstrap

React Bootstrap [13] este o bibliotecă open-source ce îmbină flexibilitatea React cu abilitățile de stilizare ale Bootstrap. Aceasta oferă o varietate de componente UI, precum butoane, bare de navigare, formulare și multe altele. Componentele sunt concepute pentru a fi ușor de personalizat și reutilizat în diverse proiecte.

2.11 React Select

React Select [14] este o componentă drop-down care oferă multiple funcționalități, precum selecția multiplă și completarea automată. De asemenea, React Select permite stilizarea personalizată și selecția fixă.

2.12 React Router DOM

React Router DOM [15] este un pachet npm esențial pentru adăugarea rutării dinamice în aplicații web. Aceasta bibliotecă ajută la gestionarea navegării și afișării diferitelor pagini într-o aplicație React. React Router DOM este compatibil atât pentru clienți cât și pentru servere, oferind o soluție integrată de rutare în ecosistemul React.

Această bibliotecă facilitează crearea unui flux de navigare fluid și intuitiv, îmbunătățind experiența utilizatorului prin tranziții lin între pagini și actualizări ale interfeței fără reîncărcare completă.

Biblioteca React Router DOM gestionează rutarea completă a aplicației, oferind o soluție specifică pentru React. Aceasta este flexibilă și ușor de implementat, facilitând navigarea între diferite pagini și componente ale aplicației.

2.13 React Ace

React Ace [16] este un editor de cod robust și bogat în funcții, destinat aplicațiilor web. Este un wrapper React pentru Ace Editor, un editor de cod autonom scris în JavaScript. Dezvoltatorii React preferă adesea React Ace datorită ușurinței sale de integrare și a gamei largi de funcționalități, inclusiv evidențierea sintaxei, completarea codului și personalizarea.

Acest editor a fost ales deoarece oferă un mediu similar cu cel al unui editor de cod, permitând utilizatorilor să editeze codul direct în aplicație, fără a mai fi necesară copierea acestuia într-un editor local pentru modificări.

2.14 Yup

Yup [17] este o bibliotecă JavaScript care permite crearea de scheme pentru obiectele de date și validarea acestora pentru a asigura conformitatea cu schemele respective. De asemenea, poate transforma datele pentru a le conforma. O schemă Yup definește structura datelor și tipurile de valori așteptate.

Această librărie a fost utilizată pentru a valida datele înainte de trimis, minimizând astfel riscul de a trimite date incorecte către server.

2.15 BcryptJS

BcryptJS [18] este o implementare JavaScript a funcției de hashing a parolei bcrypt. Este conceput pentru a fi sigur și eficient, ceea ce îl face o alegere ideală pentru hasharea parolelor în aplicațiile Node.js. Această bibliotecă este ușor de integrat și utilizează algoritmi de hashing puternici pentru a asigura securitatea datelor sensibile.

2.16 Body-parser

Body-parser [19] este o bibliotecă Node.js utilizată pentru extragerea informațiilor dintr-o solicitare HTTP primită. Aceasta procesează informațiile și le trece prin un middleware.

Middleware-ul Body-parser primește datele clientilor sub formă de formular HTML sau obiect JavaScript, în funcție de tipul solicitării HTTP. Datele prelucrate sunt stocate în ‘req.body’, putând fi ulterior folosite de alte middleware-uri pentru prelucrare suplimentară.

2.17 Express.js

Express.js [20] este un framework web open source pentru Node.js, cunoscut pentru viteza, flexibilitatea și minimalismul său. Acesta simplifică semnificativ dezvoltarea aplicațiilor web și API-urilor server-side folosind JavaScript. Dezvoltat și întreținut de fundația Node.js, Express.js oferă numeroase funcționalități care îmbunătățesc productivitatea dezvoltatorilor.

Framework-ul permite o organizare eficientă a aplicațiilor prin middleware și rutare, extinde capacitatele obiectelor Node HTTP cu utilități suplimentare și facilitează generarea conținutului HTTP dinamic. Aceste caracteristici fac din Express.js o alegere populară pentru dezvoltarea rapidă și eficientă a aplicațiilor web.

În aplicația noastră web, Express.js a fost utilizat pentru a asigura o gestionare optimă a rutelor și a facilita dezvoltarea API-urilor server-side.

2.18 Express-session

Middleware-ul express-session [21] permite crearea și stocarea datelor de sesiune pentru autentificare și preferințele utilizatorilor. Utilizând acest middleware, se poate menține eficient interacțiunea cu stare între serverul Express.js și client. Gestionarea sesiunilor îmbunătățește securitatea aplicației și optimizează experiența utilizatorului.

Acesta a fost utilizat pentru a crea sesiuni noi la fiecare autentificare, permitând astfel verificarea constantă a stării de conectare a utilizatorilor.

2.19 Mongoose

Mongoose [22] este o bibliotecă Object Data Modeling (ODM) pentru MongoDB și Node.js. Aceasta gestionează relațiile dintre date, oferă validare a schemelor și traduce obiectele din cod în reprezentările lor corespunzătoare în MongoDB. Este un instrument esențial pentru manipularea eficientă a datelor în aplicațiile Node.js.

Biblioteca Mongoose a simplificat gestionarea bazei de date pe server, oferind o integrare facilă a MongoDB cu serverul scris în Node.js.

2.20 Connect-mongodb-session

Acest modul [23] exportă o singură funcție care primește o instanță de Connect sau Express și returnează o clasă MongoDBStore. Această clasă poate fi utilizată pentru a stoca sesiuni în MongoDB, asigurând gestionarea eficientă și scalabilă a sesiunilor utilizatorilor. Modulul este ideal pentru aplicațiile care necesită stocarea persistentă a sesiunilor într-o bază de date MongoDB.

2.21 Cors

Pachetul cors [24] este un middleware foarte popular pentru Express.js, conceput pentru a simplifica configurarea CORS în aplicațiile Node.js. Acesta oferă o modalitate flexibilă și puternică de a defini politicile CORS, adaptându-se la cerințele specifice ale aplicației. Utilizând pachetul cors, dezvoltatorii pot gestiona cu ușurință cererile cross-origin, îmbunătățind securitatea și funcționalitatea aplicațiilor lor. În plus, acest

middleware este ușor de integrat și configurat, asigurând compatibilitatea cu diverse scenarii de utilizare, cum ar fi API-uri RESTful și aplicații single-page.

2.22 Dotenv

Dotenv [25] este un pachet npm (Node Package Manager) foarte popular folosit în aplicațiile NodeJS pentru gestionarea variabilelor de environment. Aceste variabile, definite încă din afara codului sursă, devin accesibile aplicației în timpul execuției. Ele sunt adesea folosite pentru a stoca informații sensibile, precum chei API, siruri de conexiune la baze de date sau alte setări de configurare critice.

În această aplicație, a fost utilizat pentru a stoca informații precum datele de autentificare ale contului Gmail folosit pentru trimiterea emailurilor către utilizatori și URL-ul de conectare la baza de date.

2.23 NodeMailer

NodeMailer [26] este un modul Node.js care vă permite să trimiteți e-mailuri de pe serverul dvs. cu ușurință. Este un modul fără dependențe externe, compatibil cu toate aplicațiile Node.js. E-mailurile trimise pot include text simplu, atașamente sau HTML. Se pot utiliza conturi Gmail sau Mailtrap pentru a configura servere SMTP false pentru testare; în cazul nostru, folosim Gmail. NodeMailer oferă numeroase funcționalități, ceea ce îl face unul dintre cele mai bune module pentru trimiterea de e-mailuri în Node.js.

În aplicația FII Pregătit, Nodemailer utilizează un cont Gmail special creat pentru această aplicație. Acesta este folosit pentru funcția de resetare a parolei, prin trimiterea unui link unic către adresa de email a utilizatorilor, permitându-le să-și reseteze parola în cazul în care aceasta a fost uitată.

2.24 UUID

Este un generator de UUID (Identifier Unic Universal) [27] simplu și flexibil pentru Node.js, care oferă o metodă fiabilă de a crea identificatori unici pentru o varietate de aplicații și scenarii de utilizare. Acest instrument este ideal pentru dezvoltatorii care

au nevoie de identificatori unici în mod consistent, asigurând unicitate și securitate în gestionarea datelor și a resurselor.

2.25 Concluzii

Fiecare dintre tehnologiile și librăriile prezentate a avut un rol esențial în dezvoltarea proiectului, fiind atent selectate pentru a crea o aplicație modernă, accesibilă, ușor de modificat, cu un design plăcut și un comportament stabil. Baza de date MongoDB oferă un mod sigur și ușor de utilizat pentru stocarea tuturor datelor necesare, integrându-se eficient cu ajutorul librăriei Mongoose. Docker asigură securitatea necesară prin crearea de containere izolate pentru fiecare rulare a surselor, prevenind astfel daunele cauzate de codul malitios. Fiecare tehnologie și librărie a contribuit semnificativ la rezultatul final al site-ului.

De asemenea, utilizarea React.js pentru dezvoltarea interfeței de utilizator a permis crearea unui front-end dinamic și responsiv, oferind o experiență de utilizare îmbunătățită. Express.js, folosit pentru dezvoltarea back-end-ului, a oferit un cadru robust și flexibil pentru gestionarea cererilor HTTP și interacțiunea cu baza de date. În plus, implementarea Node.js a permis utilizarea JavaScript atât pe server, cât și pe client, simplificând fluxul de dezvoltare și reducând complexitatea codului. În concluzie, selecția atentă a acestor tehnologii și librării a fost crucială pentru succesul proiectului, asigurând performanță, securitate și scalabilitate.

Capitolul 3

Arhitectură și implementare

În acest capitol, vom detalia funcționalitățile aplicației, atât pentru utilizatorii obișnuiți cât și pentru cei cu drepturi de administrator.

În *Figura 4* sunt ilustrate principalele funcționalități ale aplicației, pe care le vom descrie pe fiecare în parte în secțiunile următoare. Vom explora cum fiecare caracteristică contribuie la experiența generală a utilizatorului și cum acestea pot fi eficient utilizate pentru a maximiza eficiența și performanța aplicației.

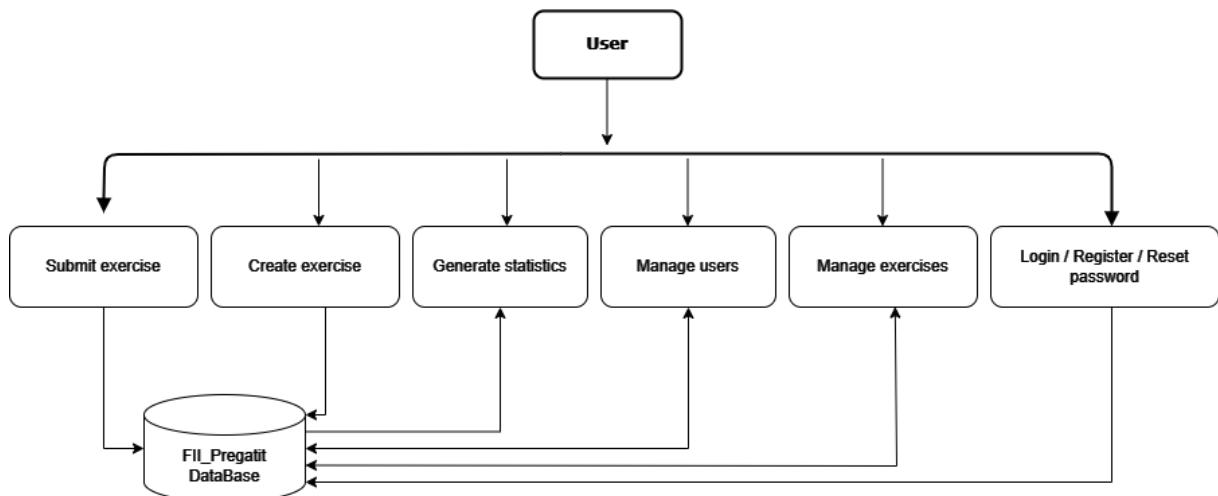


Figura 4: Arhitectura generală a aplicației

În *Figura 5*, este prezentată organizarea informațiilor în baza de date. Această figură evidențiază colecțiile necesare și modul eficient în care acestea au fost structurate pentru a optimiza utilizarea spațiului.

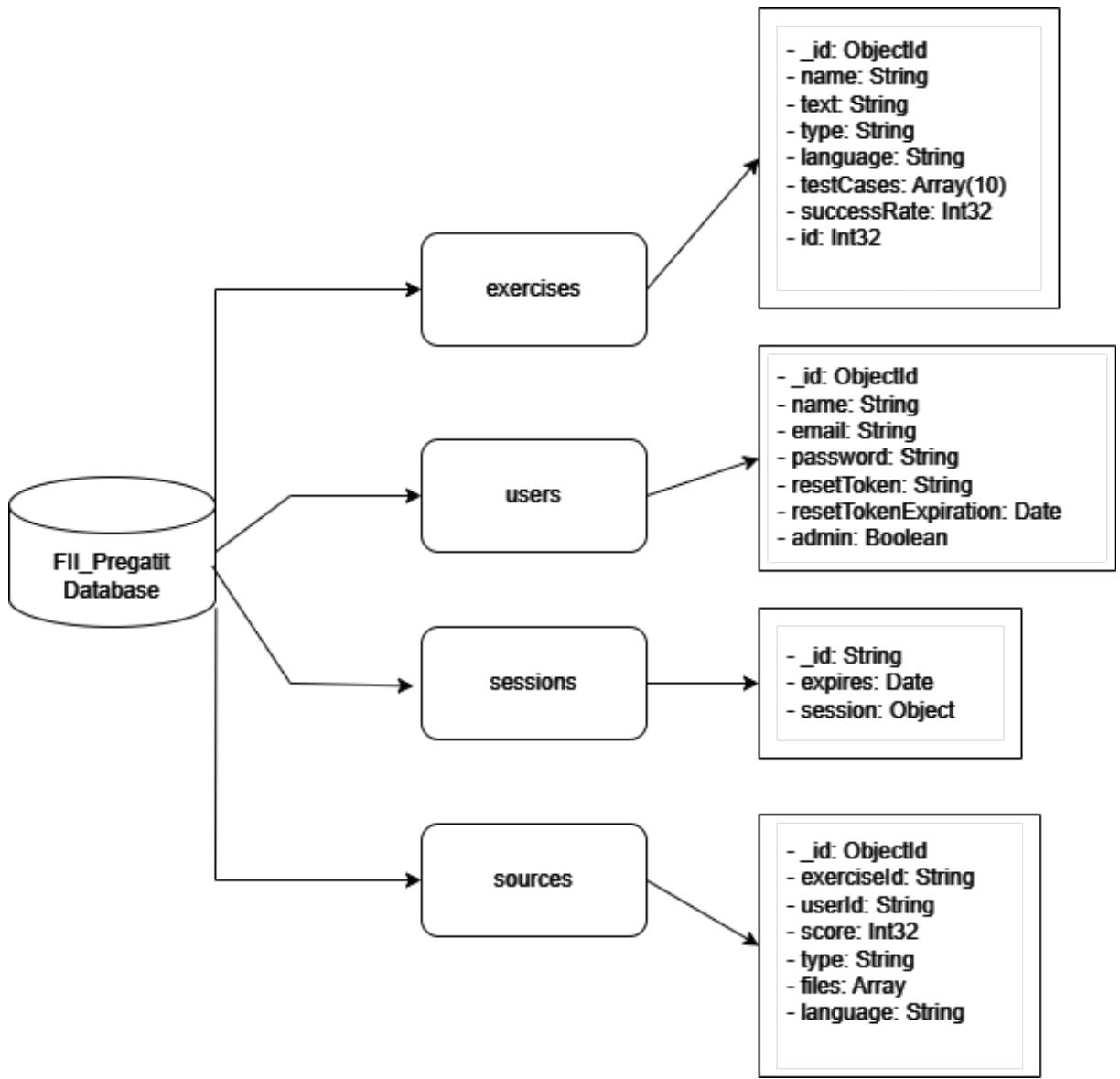


Figura 5: Organizarea bazei de date

Platforma gestionează patru colecții principale în baza de date MongoDB, fiecare având un rol esențial în funcționarea aplicației.

Prima colecție, denumită „exercises”, stochează informații vitale pentru fiecare exercițiu. Aceasta include un ID unic generat automat de MongoDB pentru identificarea fiecărui exercițiu, numele și descrierea exercițiului, categoria acestuia (field-ul „type”), și eventualele restricții de limbaj (field-ul „language”). De asemenea, conține un array de 10 perechi de date input-output utilizate pentru calcularea scorului surselor trimise, un câmp ce reflectă rata de succes a exercițiului, și un ID suplimentar ce facilitează căutarea exercițiilor de către utilizatori.

A doua colecție, „users”, este dedicată informațiilor utilizatorilor. Aceasta include un ID unic generat de MongoDB, numele utilizatorului, adresa de email a contului, parola criptată, un „resetToken” pentru resetarea parolei și un câmp „resetTokenExpi-

ration” care definește durata de valabilitate a acestui token. De asemenea, un câmp „admin” indică dacă utilizatorul are drepturi de administrator.

Cea de-a treia colecție, „sessions”, se ocupă de gestionarea sesiunilor active ale utilizatorilor. Aceasta înregistrează un ID unic, perioada de valabilitate a sesiunii (setată la 24 de ore de la conectare), și un obiect ce cuprinde toate datele legate de sesiunea creată.

Ultima colecție, „sources”, conține sursele încărcate de utilizatori. Aici sunt stocate: ID-ul unic al sursei, ID-ul exercițiului asociat, ID-ul utilizatorului care a încărcat sursa, scorul obținut de utilizator pentru acea sursă, categoria problemei pentru care sursa a fost încărcată, un array cu perechile nume fisier - cod sursă din fiecare fișier încărcat, și limbajul în care a fost procesat codul.

Toate aceste colecții au fost concepute și structurate pentru a asigura o gestionare eficientă și organizată a datelor, evitând duplicarea informațiilor și optimizând accesul la acestea. Această organizare ajută la menținerea unei baze de date curate și la eficientizarea proceselor de căutare și accesare a datelor necesare, fie că este vorba de gestionarea exercițiilor, utilizatorilor, sesiunilor sau surselor încărcate. Această structură raționalizată contribuie la o performanță îmbunătățită a platformei și la o experiență de utilizare mai fluidă și intuitivă.

3.1 Procesare răspuns

Această funcționalitate este esențială în cadrul aplicației, deoarece gestionează procesarea codului sursă trimis de utilizatori și furnizează feedback bazat pe corectitudinea codului respectiv.

În *Figura 6* este prezentată pagina de unde utilizatorul poate consulta cerințele problemei și poate trimite soluția sa pentru procesare.

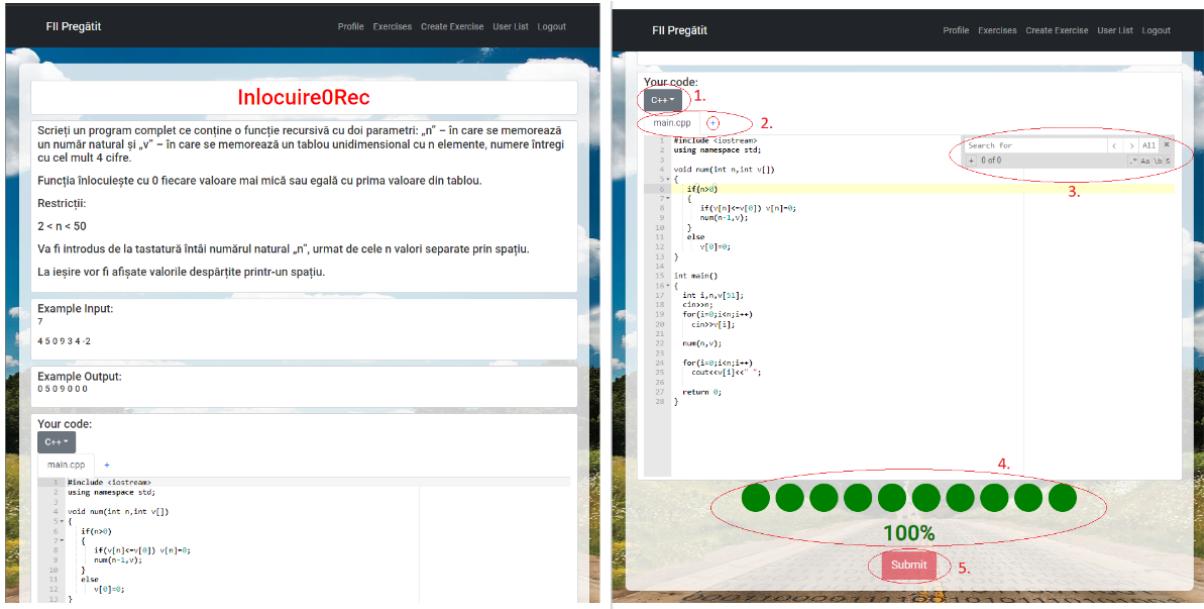


Figura 6: Pagina problemelor de pe aplicația FII Pregătit

De la prima interacțiune, utilizatorul poate observa numele, cerința exercițiului, și un exemplu de input-output oferit de autorul problemei. Sub aceste informații se află un editor de cod, care reprezintă spațiul în care utilizatorul va introduce soluția problemei. Editorul oferă diverse funcționalități, printre care posibilitatea de a alege limbajul de programare pentru scrierea soluției (*vezi 1. în Figura 6*). Această funcție poate fi limitată de autorul problemei la crearea acesteia.

Utilizatorul poate crea, pe lângă fisierul principal denumit *main*, și alte fișiere auxiliare, precum headere, clase sau alte fișiere de tip .cpp, .py etc., pentru elaborarea unui clean code (*vezi 2. în Figura 6*). În cadrul aplicației FII Pregătit, editorul include și o bară de căutare pentru a facilita găsirea variabilelor sau a liniilor de cod dorite (*vezi 3. în Figura 6*).

În funcție de corectitudinea codului, va fi afișat un procentaj care reprezintă punctajul obținut după procesarea soluției. Acest scor variază de la 0 la 100, fiind afișat împreună cu indicatoare colorate specific fiecărui interval de punctaj pentru a oferi un feedback vizual clar (*vezi 4. în Figura 6*). Dacă nu a fost încărcată nicio sursă, scorul și indicatoarele nu vor apărea, urmând să fie vizibile doar după încărcarea unei surse. Datorită timpului necesar procesării codului, până la primirea punctajului de la server, se va afișa un spinner pentru a informa utilizatorul că sursa este în curs de evaluare.

Butonul de trimitere (*vezi 5. în Figura 6*) este strict pentru expedierea codului sursă către server, fiind dezactivat dacă nu s-au efectuat modificări în editor, sau după ce o sursă a fost trimisă, necesitând modificări ulterioare în editor pentru reactivarea sa.

Acest mecanism previne trimitera repetitivă de surse identice și reduce suprasolicitarea serverului.

Odată ce codul a fost trimis de către utilizator, serverul verifică limbajul de programare selectat de utilizator pentru a asigura că procesarea este adecvată limbajului respectiv. Pentru codul scris în C++, se utilizează compilatorul g++ din GNU Compiler Collection. Odată ce codul este trimis către server, acesta este plasat într-o coadă și procesat într-un container Docker cu un ID unic (vezi Figura 7) pentru a identifica eventualele greșeli de sintaxă. Dacă sunt detectate erori, clientul este notificat cu un scor de 0 și mesajul „*Compilation error*”. În absența erorilor, sunt create alte 10 containere pentru a rula teste care vor determina scorul final.

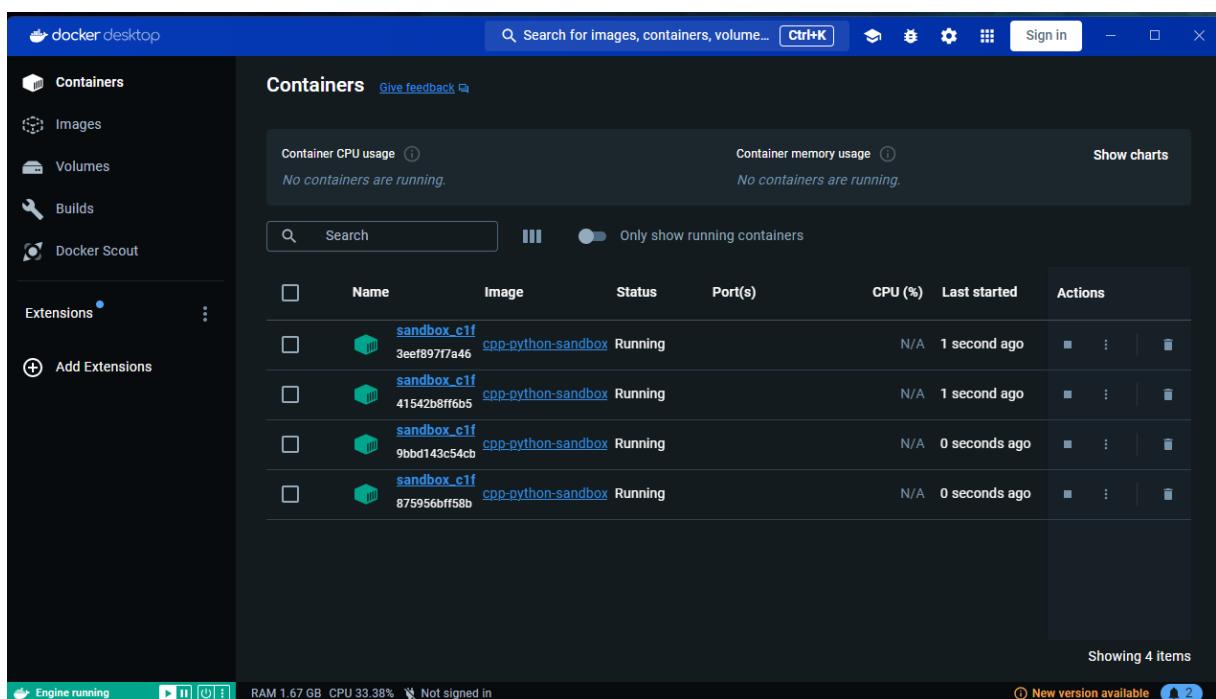


Figura 7: Containere generate cu ajutorul Docker

Utilizarea Dockerului asigură securitatea aplicației prin prevenirea problemelor generate de codul malicios și evită duplicarea accidentală a containerelor, fiecare container fiind creat cu un ID unic. După finalizarea testelor, containerele și fișierele temporare sunt eliminate pentru a menține curățenia și eficiența serverului.

Serverul gestionează și cazurile în care codul conține bucle infinite, aplicând un timeout fiecărui test, care oprește execuția și distrugе resursele temporare, urmând să informeze clientul cu mesajul „*Time limit exceeded*”. Această metodă protejează serverul de suprasolicitare și informează utilizatorul despre problemele din codul său.

După rularea celor 10 teste, scorul bazat pe numărul de teste trecute este calculat și trimis către client. Sursa, împreună cu scorul, limbajul în care a fost scrisă, ID-ul

problemei, ID-ul utilizatorului și categoria problemei sunt stocate în baza de date pentru utilizări viitoare și pentru generarea de statistici.

La final, containerele și fișierele temporare sunt eliminate, sursa este scoasă din coadă și procesul este repetat pentru următoarea sursă din coadă. În prezent, pot fi procesate simultan 10 surse, cu posibilitatea de extindere a acestei limite în funcție de capacitatea de procesare a serverului. Restul surselor rămân în coadă până când resursele se eliberează pentru a le procesa.

3.2 Creare exercițiu

Una dintre funcționalitățile specifice administratorilor este posibilitatea de a crea exerciții, proces ilustrat în *Figura 8*.

Administratorii își pot adăuga textul problemei și apoi selecta dintr-un searchable dropdown categoria corespunzătoare problemei pe care doresc să o creeze. Alegerea categoriei contribuie la generarea statisticilor pentru utilizatori. După selectarea categoriei, urmează un alt searchable dropdown unde autorul poate impune restricții de limbaj pentru problema creată. Dacă dorește ca problema să accepte răspunsuri doar într-un anumit limbaj din lista disponibilă, va selecta acel limbaj specific; în caz contrar, pentru a permite rezolvarea în toate limbajele, va opta pentru „*All languages*”.

The figure consists of two side-by-side screenshots of a web application interface. Both screenshots have a header with the text "FII Pregătit" and navigation links: Profile, Exercises, Create Exercise, User List, and Logout. The left screenshot shows the "Create Problem" page. It has fields for "Problem Name" (empty), "Problem Text" (empty), "Problem Type" (a dropdown menu showing "Select Problem Type"), "Language" (a dropdown menu showing "Select language"), and "Test Case 1" with "Input" and "Output" fields (both empty). The right screenshot shows the same page after filling in the first test case. It now includes "Test Case 8", "Test Case 9", and "Test Case 10", each with "Input" and "Output" fields. At the bottom right of the right screenshot, there is a red "Create" button. The background of both screenshots features a blue sky with clouds.

Figura 8: Pagina de creare a exercițiilor

Autorul trebuie să introducă apoi cele 10 seturi de date pentru input-output care vor fi utilizate pentru calcularea scorului fiecărei surse trimise. Ideal, aceste cazuri

ar trebui să fie variate pentru a testa diferite aspecte, cum ar fi memoria, timpul de răspuns al codului și situații specifice problemei generate. Primul test de input-output va fi folosit și ca exemplu pentru utilizatori. La finalizarea acestor pași și apăsarea butonului de trimitere, sistemul verifică completitudinea câmpurilor, iar dacă totul este în ordine, autorul este notificat cu mesajul „*Problem successfully created*” și redirecționat către pagina cu lista de exerciții.

După primirea datelor de la server, se efectuează o verificare pentru a asigura că nu există o altă problemă cu același nume. Dacă o astfel de problemă există deja, autorul va primi mesajul „*An exercise with this name already exists*”. În caz contrar, clientul va fi informat că exercițiul a fost creat cu succes. Exercițiul este salvat în baza de date cu un ID unic, facilitând identificarea acestuia în lista de exerciții.

3.3 Generare statistici

După ce un utilizator încarcă cel puțin o sursă, pagina profilului său afișează statistici care ilustrează punctele sale forte și slabe, pe baza scorurilor obținute. Cu cât mai multe surse sunt încărcate de un utilizator, cu atât statisticile devin mai detaliate, reflectând diversitatea problemelor abordate. În diagrama profilului sunt incluse doar categoriile de probleme pentru care utilizatorul a trimis soluții. Categoriile fără soluții încărcate nu sunt reprezentate în diagramă. Plasând cursorul peste o coloană din diagramă, apare un procentaj ce indică rata de succes a utilizatorului pentru acea categorie de probleme.

Statisticile sunt generate în timp real când utilizatorul accesează pagina de profil, nu în momentul încărcării soluțiilor, pentru a reduce sarcina de procesare a serverului. Diagrama este creată utilizând biblioteca React Chart.js, care asigură o prezentare intuitivă și estetică (vezi Figura 9). Generarea statisticilor se bazează pe sursele salvate în baza de date, unde serverul identifică categoriile de probleme abordate de utilizator și calculează statistici bazate pe numărul de surse cu scor maxim pentru fiecare categorie.

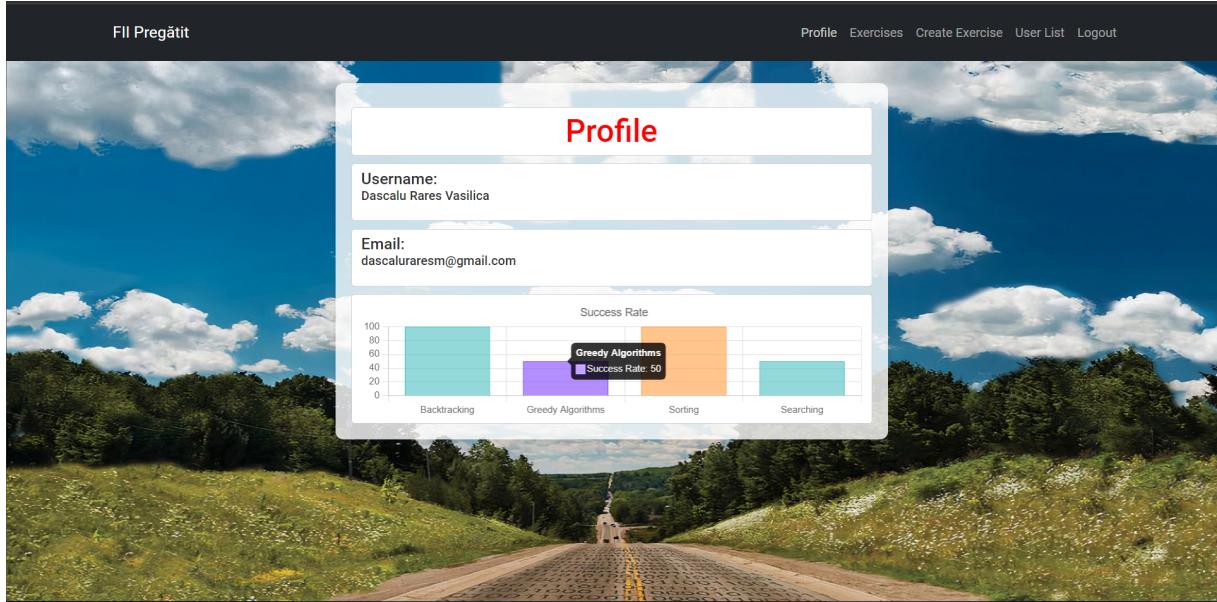


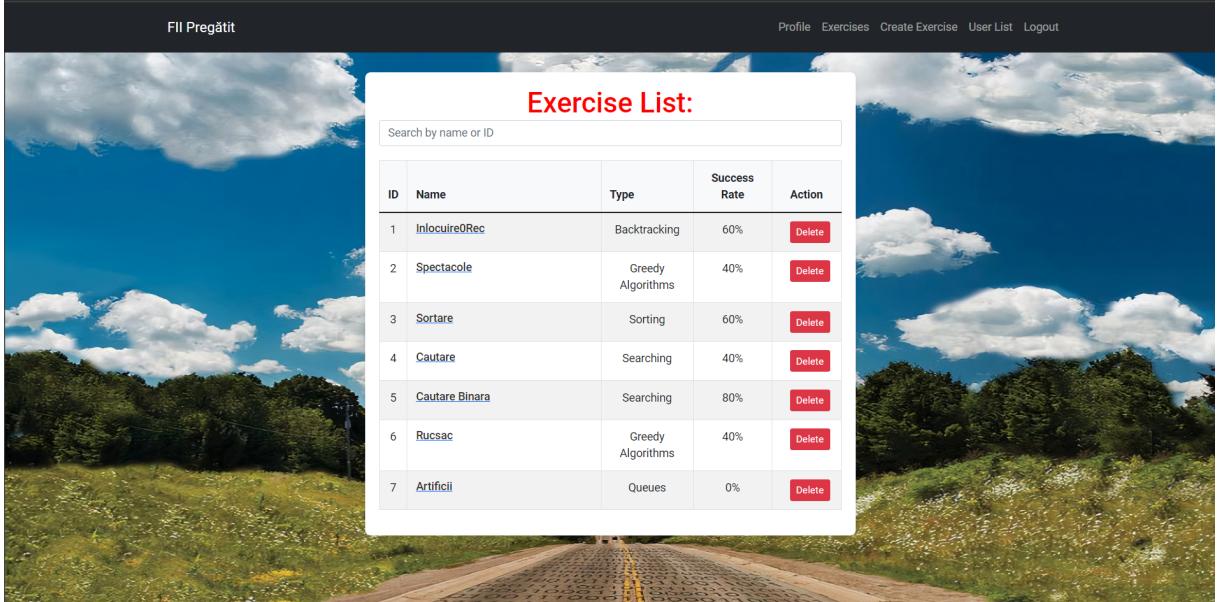
Figura 9: Pagina cu statisticile utilizatorului

Pe pagina de profil sunt afișate, de asemenea, informații personale ale utilizatorului, precum adresa de email și numele, completând astfel panorama resurselor disponibile pentru monitorizarea progresului individual.

3.4 Gestionare exerciții

O pagină esențială în platformă este lista completă a exercițiilor disponibile. Această listă oferă acces la exercițiile ce pot fi rezolvate, iar utilizatorii pot folosi bara de căutare pentru a localiza un exercițiu specific după nume sau ID. Odată ce un exercițiu este selectat, utilizatorii sunt redirectionați către pagina dedicată acestuia, unde pot consulta cerința și încărca codul sursă pentru procesare.

Pentru utilizatorii cu drepturi de administrator, în dreptul fiecărui exercițiu apar două funcții suplimentare, vizibile în *Figura 10*. Prima este o rubrică ce afișează rata de succes a problemei, calculată pe baza surselor trimise de utilizatori. A doua este un buton de delete, care, odată apăsat, elimină atât datele problemei cât și toate sursele asociate, asigurând eliminarea datelor reziduale. Aceste instrumente suplimentare oferă administratorilor un control mai mare asupra gestionării conținutului și facilitează menținerea curățeniei și eficienței bazei de date, contribuind la o administrare eficientă a resurselor și la optimizarea experienței de utilizare pe platformă.

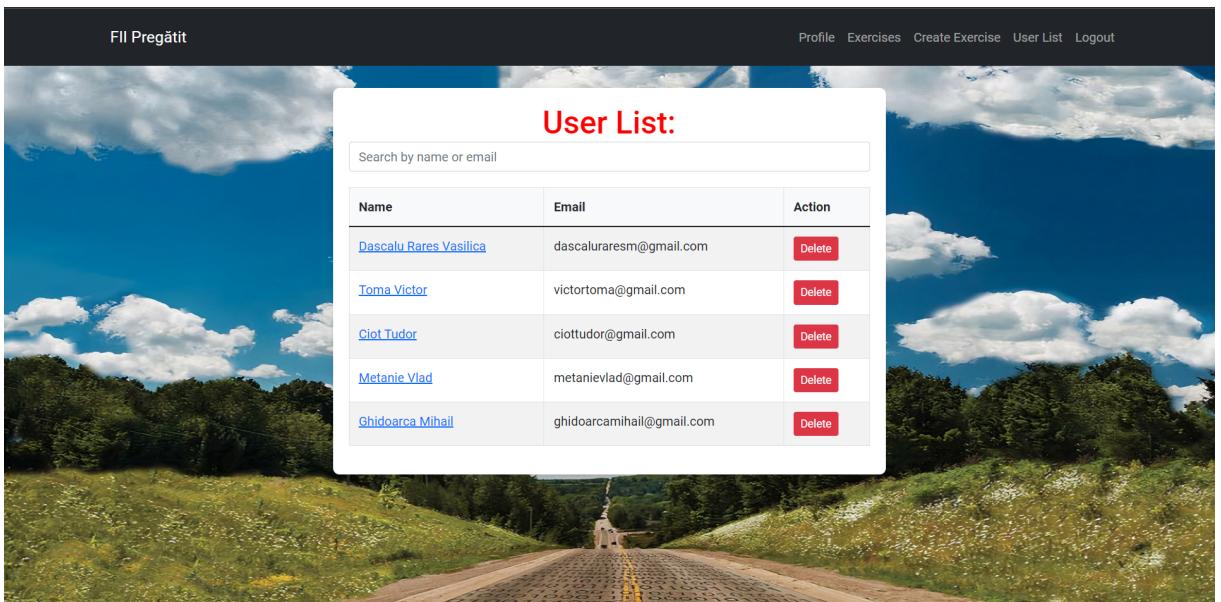


Exercise List:				
Search by name or ID				
ID	Name	Type	Success Rate	Action
1	Inlocuire0Rec	Backtracking	60%	Delete
2	Spectacole	Greedy Algorithms	40%	Delete
3	Sortare	Sorting	60%	Delete
4	Cautare	Searching	40%	Delete
5	Cautare Binara	Searching	80%	Delete
6	Rucsac	Greedy Algorithms	40%	Delete
7	Artificii	Queues	0%	Delete

Figura 10: Lista cu exercițiile disponibile

3.5 Gestionare utilizatori

O altă funcționalitate esențială pentru utilizatorii cu drepturi de administrator este gestionarea utilizatorilor. Administratorii pot accesa lista de utilizatori și au opțiunea de a șterge conturi prin apăsarea butonului „delete” situat în dreptul fiecărui utilizator. De asemenea, pot căuta utilizatori în bara de căutare după nume sau adresă de email. În momentul ștergerii unui utilizator, toate sursele asociate aceluia cont sunt de asemenea eliminate din baza de date, asigurând astfel că nu rămân date reziduale.



User List:		
Search by name or email		
Name	Email	Action
Dascalu Rares Vasilica	dascaluraresm@gmail.com	Delete
Toma Victor	victortoma@gmail.com	Delete
Ciot Tudor	cioottudor@gmail.com	Delete
Metanie Vlad	metanievlad@gmail.com	Delete
Ghidarcă Mihail	ghidoarcamihail@gmail.com	Delete

Figura 11: Lista cu utilizatori

Pagina de administrare a utilizatorilor a fost concepută pentru a fi cât mai intuitivă și placută, aşa cum se observă în *Figura 11*. Aceasta include funcționalități suplimentare care permit vizualizarea surselor fiecărui student. După localizarea studentului dorit, accesând numele acestuia, veți fi redirecționați către o listă cu exercițiile disponibile. Selectând exercițiul dorit, se va deschide o pagină similară cu cea a problemei, diferență constând în faptul că codul studentului este prezentat fără opțiunea de a fi editat sau retrimis la server. În plus, atunci când vizualizați o sursă, puteți vedea și scorul obținut de utilizator pentru acea problemă, detalii ilustrate în *Figura 12*. Aceste caracteristici oferă administratorilor un control eficient și direct asupra gestionării conținutului și activităților utilizatorilor.

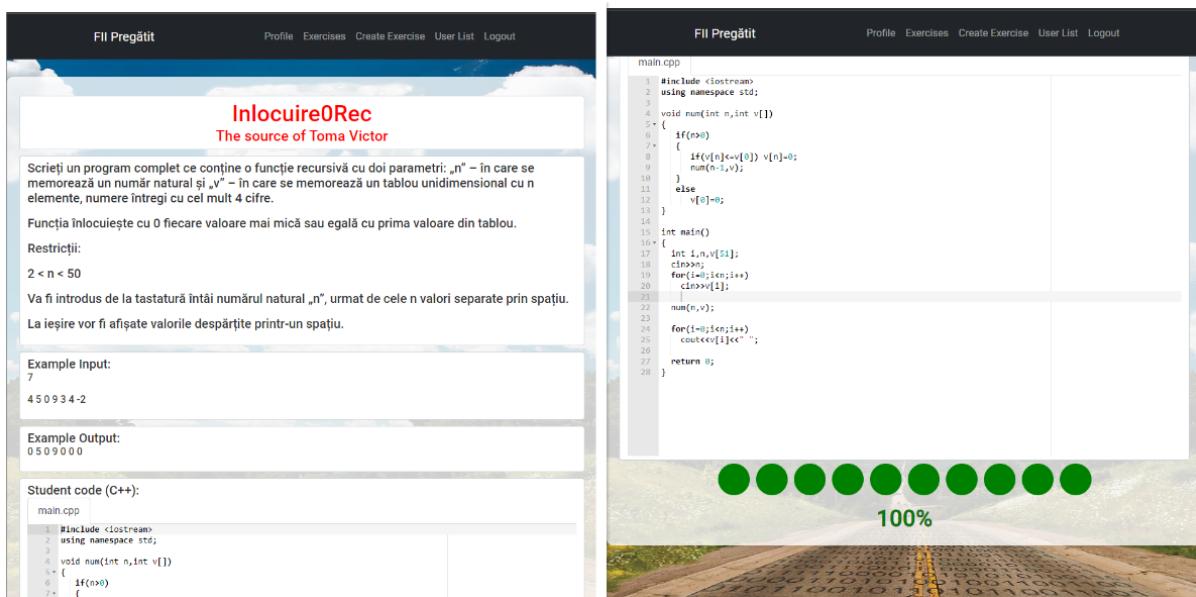


Figura 12: Pagina cu sursa utilizatorului

3.6 Register / Login / Reset password

Acste trei funcționalități sunt interconectate și esențiale pentru experiența utilizatorului pe platformă.

Prima funcție, cea de înregistrare, le permite utilizatorilor să creeze un cont asociat tuturor surselor trimise, acces indispensabil pentru rezolvarea exercițiilor disponibile. Formularul de înregistrare solicită completarea următoarelor câmpuri: un nume de utilizator, de preferință numele complet al persoanei, un email pentru resetarea parolei în caz de necesitate și o parolă. Parola trebuie să respecte anumite criterii de securitate: minimum 8 caractere, cel puțin o cifră, o literă mare și una mică, și cel puțin un caracter special. Lângă câmpul pentru parolă există un buton care permite vizualizarea

parolei introduse pentru a evita erori. Un ultim câmp necesită confirmarea parolei prin reintroducerea acesteia, asigurându-se astfel corectitudinea acesteia.

Butonul de trimitere a datelor ("Submit") inițiază verificarea condițiilor de validare pentru fiecare câmp și afișează mesaje de eroare adecvate sub fiecare câmp, în cazul în care informațiile introduse nu îndeplinesc cerințele.

Odată ce datele sunt verificate, acestea sunt trimise către server, care verifică dacă există deja un cont asociat cu adresa de email furnizată. Dacă un astfel de cont există, serverul notifică utilizatorul cu mesajul „*A user with this email already exists*”. Dacă adresa de email nu este asociată cu un alt cont, serverul folosește biblioteca bcrypt.js pentru a cripta parola, sporind astfel securitatea aplicației, și înregistrează noile date ale utilizatorului în baza de date, creând astfel un cont nou. La final, serverul confirmă crearea contului, trimițând utilizatorului mesajul „*User created successfully!*”.

O funcționalitate complementară celei de înregistrare este cea de autentificare, prin care utilizatorii trebuie să introducă adresa de email și parola asociate contului pentru a accesa exercițiile disponibile. Butonul de trimitere ("Submit") încarcă aceste date pe server pentru verificare și, dacă informațiile sunt corecte, permite accesul la conținut.

După primirea datelor de la utilizator, serverul efectuează o căutare în baza de date pentru adresa de email specificată. Dacă găsește un utilizator corespunzător, compară parola introdusă cu cea din baza de date folosind funcțiile din biblioteca bcrypt.js. Dacă autentificarea eșuează, fie din cauza unei adrese de email inexistente, fie din cauza unei parole incorecte, serverul trimită înapoi mesajul „*Invalid email or password*”.

Dacă autentificarea reușește, serverul inițiază o sesiune nouă pentru utilizator, pe care o salvează și în baza de date iar apoi trimită către client mesajul „*Logged in successfully*”. Această sesiune permite monitorizarea stării de conectare a utilizatorului în orice moment și este încheiată fie prin apăsarea butonului de deconectare din bara de navigație, fie la închiderea browserului.

Odată creată sesiunea, serverul verifică autenticitatea conexiunii unui utilizator de fiecare dată când acesta accesează o pagină specifică a aplicației, comparând datele de sesiune primite de la utilizator cu cele din baza de date. Există, de asemenea, un control suplimentar pentru paginile dedicate administratorilor, prin care serverul confirmă dacă utilizatorul posedă drepturi de administrator.

Această verificare a stării de conectare se efectuează și atunci când un utilizator încearcă să trimită o sursă spre procesare, când se încarcă o nouă problemă sau când un

administrator intenționează să șteargă un utilizator sau o problemă.

Acste verificări sunt esențiale, mai ales după repornirea serverului, când toate sesiunile sunt șterse, dar utilizatorul poate fi încă activ pe pagină. Fără aceste teste, serverul ar întâmpina dificultăți în asocierea surselor trimise cu conturile corespunzătoare, ceea ce ar duce la neconcordanțe în baza de date. Această metodă asigură integritatea și coerenta datelor gestionate de aplicație.

Ambele funcționalități, de înregistrare și autentificare, beneficiază de un design intuitiv și atractiv, aşa cum este ilustrat în *Figura 13*.

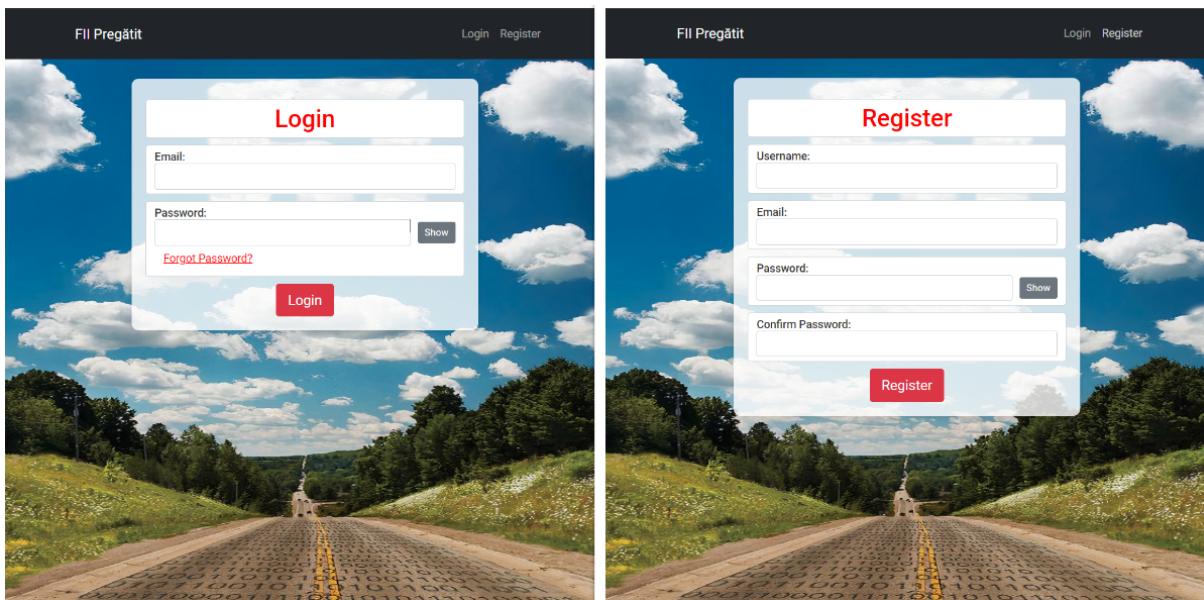


Figura 13: Paginile de Login și Register

O altă funcționalitate esențială a platformei, strâns legată de procesele de înregistrare și autentificare, este opțiunea de resetare a parolei. Aceasta asigură că utilizatorii nu își pierd accesul la conturile lor chiar dacă uită parola, protejând în același timp datele asociate contului și menținând securitatea acestora.

În momentul creării contului, în baza de date sunt generate două câmpuri suplimentare pentru a facilita resetarea parolei. Primul câmp, denumit *resetToken*, este inițializat cu valoarea *null* și va stoca un token unic necesar procesului de resetare a parolei. Al doilea câmp, *resetTokenExpiration*, este de asemenea inițializat cu *null* și va înregistra data de expirare a tokenului. Această dată limitează validitatea tokenului la o oră, crescând securitatea procesului și prevenind utilizarea neautorizată.

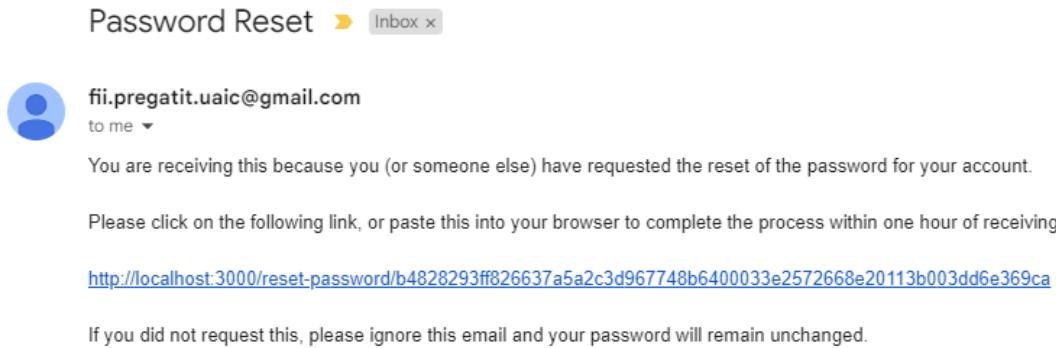


Figura 14: Mail-ul ce conține link-ul unic

Când un utilizator își uită parola, trebuie să selecteze opțiunea „*Forgot password?*” pe pagina de login. Aceasta îl redirectionează către o pagină unde poate introduce adresa de email pentru a primi un link unic de resetare (vezi Figura 14) folosind libraria NodeMailer și un cont de gmail creat special pentru această funcționalitate. După introducerea adresei de email, sistemul verifică dacă aceasta există în baza de date. Dacă adresa este găsită, se generează un token unic asociat cu contul utilizatorului și i se setează un termen de expirare.

Link-ul unic trimis pe email redirectionează utilizatorul către o pagină unde poate seta o nouă parolă. Utilizatorul trebuie să introducă noua parolă în câmpurile „*Password*” și „*Confirm password*” pentru a preveni erorile de tastare. După confirmarea parolei și apăsarea butonului de resetare, serverul extrage tokenul din URL, îl caută în baza de date și verifică validitatea acestuia. Dacă tokenul este invalid sau expirat, serverul informează utilizatorul printr-un mesaj „*Invalid or expired reset token*”. Dacă procesul este valid, parola este resetată și utilizatorul primește mesajul „*Password has been reset*”, fiind apoi redirectionat către pagina de login. Câmpurile *resetToken* și *resetTokenExpiration* sunt reinitializate cu *null* după resetarea parolei.

Noua parolă trebuie să respecte anumite criterii pentru a asigura securitatea contului: minim 8 caractere, cel puțin un caracter special, o literă mare și una mică, și cel puțin o cifră. Există și un buton pentru vizualizarea parolei, facilitând introducerea corectă a acesteia și prevenind erorile de tastare.

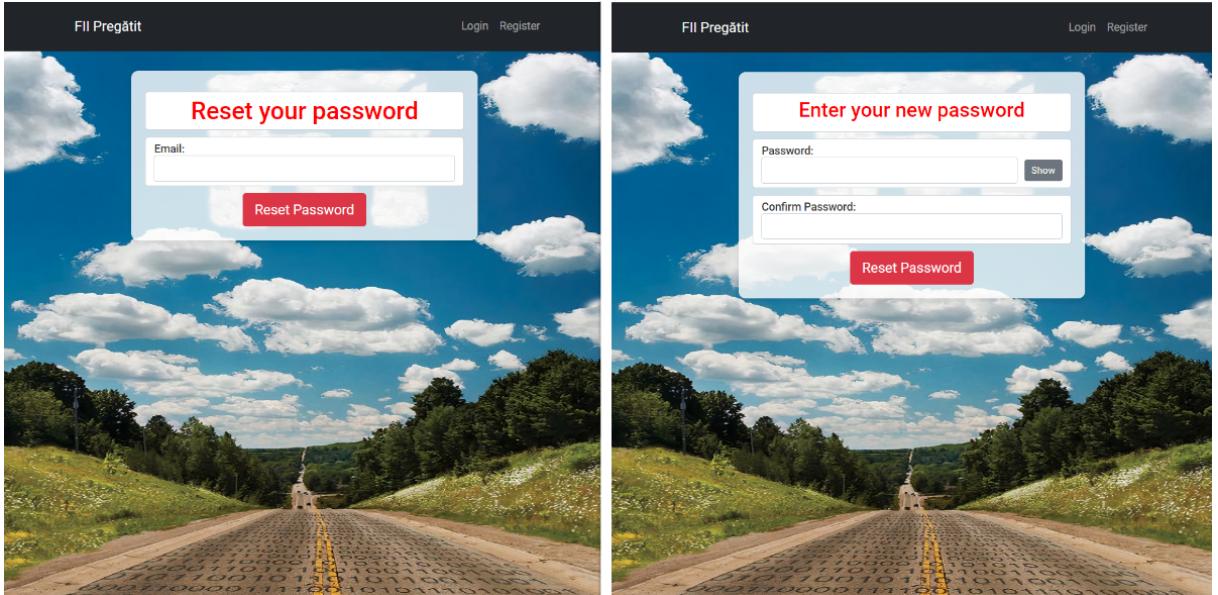


Figura 15: Paginile responsabile de resetarea parolei

Această funcționalitate beneficiază de un design intuitiv și atractiv, aşa cum este prezentat în *Figura 15*, îmbunătățind experiența utilizatorului și asigurând o interfață prietenoasă și accesibilă. Implementarea acestei opțiuni nu doar că oferă o soluție robustă pentru gestionarea parolelor uitate, dar și contribuie la securitatea generală a platformei, asigurând că utilizatorii pot recupera accesul la conturile lor fără riscul compromiterii datelor personale. Astfel, procesul de resetare a parolei devine o parte integrantă și esențială a experienței de utilizare a platformei, consolidând încrederea utilizatorilor în sistemul de securitate oferit.

3.7 Concluzii

Acest capitol este destinat să ofere o imagine de ansamblu asupra arhitecturii și implementării aplicației. În această secțiune, am inclus diagrame și imagini ale paginilor care ilustrează fiecare funcționalitate, precum și descrieri ale modului în care acestea funcționează și au fost implementate.

Capitolul 4

Scenarii de utilizare

O aplicație reușește să capteze și să mențină interesul utilizatorilor prin intermediul funcționalităților sale. Atunci când utilizatorii explorează pentru prima dată o aplicație, ei testează fiecare funcționalitate disponibilă pentru a evalua cât de bine răspunde aceasta nevoilor și preferințelor lor. Dacă cel puțin una dintre aceste funcționalități reușește să le fie pe plac, utilizatorii vor continua să utilizeze aplicația, integrând-o în rutina lor zilnică. În caz contrar, dacă niciuna dintre funcționalitățile testate nu le satisfac așteptările, utilizatorii vor înceta să folosească aplicația și vor căuta alternative care să le ofere o experiență mai bună.

În acest capitol, voi prezenta câteva scenarii de utilizare pentru a ilustra modul în care diferitele funcționalități ale aplicației pot atrage și reține utilizatorii. Fiecare scenariu va fi însoțit de diagrame Use Case, care vor oferi o reprezentare vizuală a interacțiunilor dintre utilizatori și funcționalitățile aplicației. Aceste diagrame vor ajuta la o înțelegere mai clară a modului în care diferite tipuri de utilizatori interacționează cu aplicația și cum diversele funcționalități pot contribui la succesul acesteia.

4.1 Scenariul 1

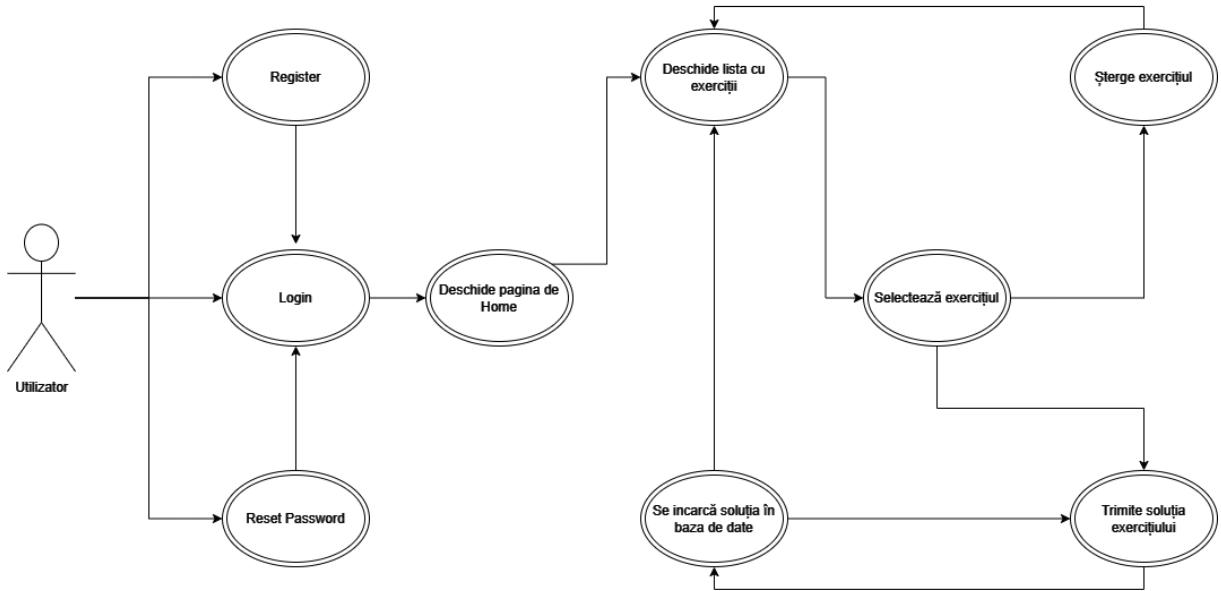


Figura 16: Scenariu de utilizare pentru rezolvarea și gestionarea exercițiilor

În *Figura 16* observăm diagrama UseCase a scenariului de rezolvare și gestionare a exercițiilor.

Când un utilizator dorește să acceseze aplicația, acesta trebuie să se autentifice pentru a i se atribui drepturile necesare navigării prin aplicație. Dacă utilizatorul nu are un cont, poate accesa pagina de înregistrare pentru a crea unul nou, unde vor fi asociate toate informațiile necesare. În cazul în care utilizatorii și-au uitat parola, aceștia pot opta pentru resetarea parolei prin introducerea adresei de email asociate contului, urmând să primească un link unic generat pe email care le va permite să își schimbe parola.

Odată autentificat cu succes, utilizatorul este redirectionat către pagina principală (Home), de unde, cu ajutorul butonului situat pe bara de navigație, poate accesa lista cu toate exercițiile disponibile puse la dispoziție de administratori. După găsirea exercițiului dorit, utilizatorii pot accesa pagina acestuia pentru a începe rezolvarea.

Utilizatorii cu drepturi de administrator au acces la funcții suplimentare, cum ar fi vizualizarea ratelor de succes ale exercițiilor și posibilitatea de a șterge un exercițiu, inclusiv toate sursele asociate acestuia.

După ce un utilizator rezolvă un exercițiu, primește un scor generat automat, iar fișierele sursă trimise sunt încărcate în baza de date. Utilizatorii pot reveni oricând pentru a modifica ultima sursă trimisă în încercarea de a obține punctaj maxim, în cazul în care nu au reușit acest lucru de prima dată.

Acet sistem asigură o gestionare eficientă și transparentă a exercițiilor, oferind utilizatorilor posibilitatea de a-și monitoriza progresul și de a îmbunătăți performanțele. De asemenea, permite administratorilor să mențină o organizare clară și eficientă a conținutului educațional disponibil pe platformă.

4.2 Scenariul 2

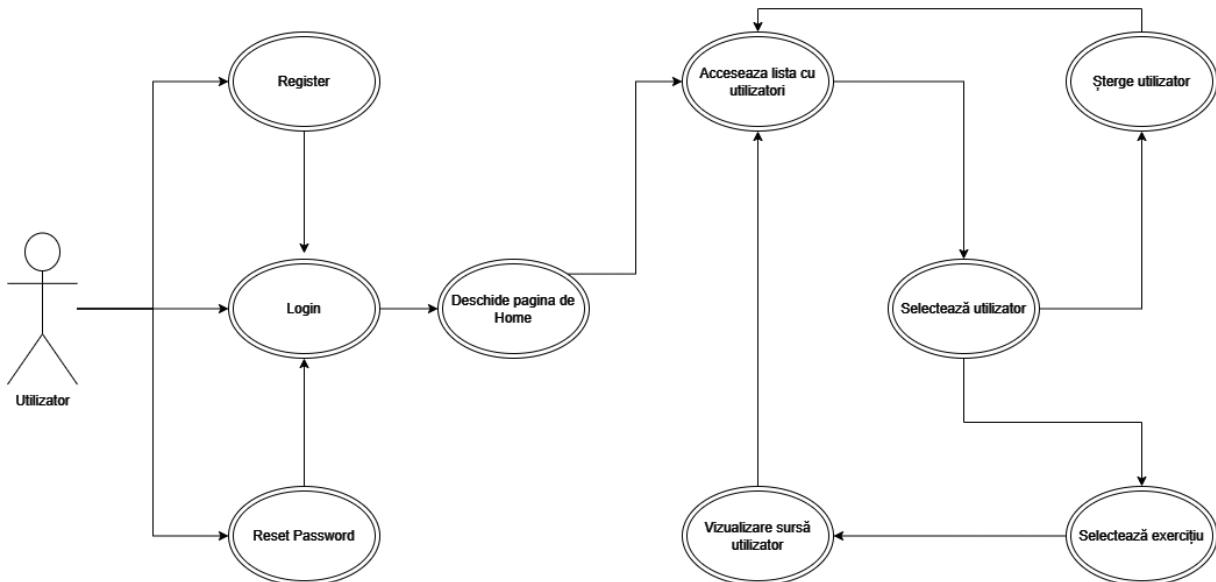


Figura 17: Scenariu de utilizare pentru gestionarea utilizatorilor și vizualizarea surselor

În Figura 17 este ilustrată diagrama UseCase asociată scenariului de utilizare pentru gestionarea utilizatorilor și vizualizarea surselor trimise.

După autentificarea cu succes a unui utilizator cu drepturi de administrator, acesta capătă acces la o funcționalitate specială prin intermediul butonului "User List" situat pe bara de navigație. Acest buton redirecționează administratorul către o listă care include toți utilizatorii aplicației, oferindu-i posibilitatea de a șterge un utilizator anume, împreună cu toate sursele asociate contului acestuia.

Dacă administratorul dorește să examineze o sursă pentru o posibilă corectare sau evaluare suplimentară, poate căuta utilizatorul specific în lista menționată anterior. Odată ce numele utilizatorului este selectat, administratorul este redirecționat către o listă de exerciții. Aici, poate căuta exercițiul specific pentru care dorește o evaluare suplimentară.

La selectarea exercițiului dorit, administratorul este dus către o pagină unde poate vizualiza detalii ale problemei și fișierele sursă trimise de utilizatorul vizat. Alături de fișierele sursă, este afișat și scorul obținut de utilizator în urma evaluării automate. Este

important de menționat că fișierele sursă ale utilizatorului nu pot fi modificate de către administrator, acestea putând fi doar vizualizate pentru o evaluare suplimentară mai detaliată.

Această facilitate asigură o gestionare eficientă și transparentă a activităților utilizatorilor în cadrul platformei, permitând administratorilor să monitorizeze și să asigure integritatea procesului de evaluare.

4.3 Scenariul 3

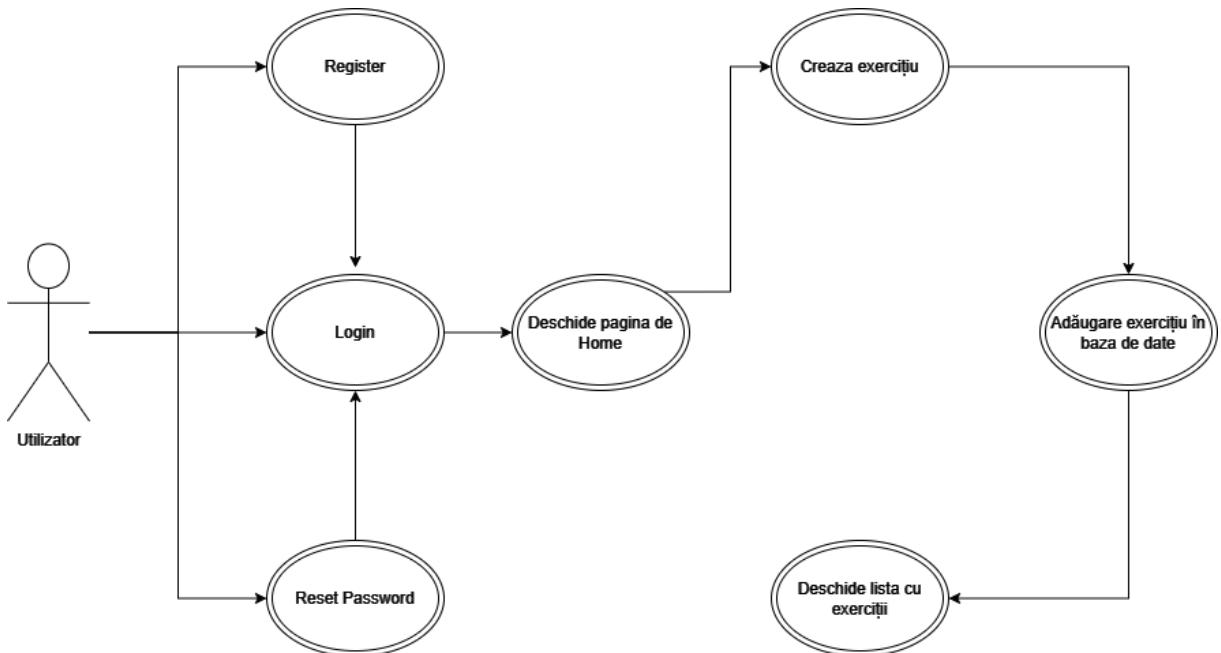


Figura 18: Scenariu de utilizare pentru crearea exercițiilor

În Figura 18 este prezentată diagrama UseCase pentru scenariul de utilizare al creării exercițiilor.

Utilizatorii cu drepturi de administrator au acces la o funcție specială care le permite să creeze exerciții. După autentificare și validarea drepturilor de administrator, aceștia pot accesa o pagină dedicată prin intermediul unui buton suplimentar apărut pe bara de navigație.

Această pagină permite administratorilor să introducă numele și cerințele exercițiului, precum și să impună restricții privind limbajul de programare utilizat pentru rezolvarea exercițiului, dacă doresc. Există și opțiunea de a permite utilizatorilor să folosească orice limbaj de programare disponibil. De asemenea, administratorul trebuie să selecteze una dintre categoriile disponibile pentru a clasifica exercițiul corespunzător și să introducă 10 perechi de input-output, care vor fi folosite pentru evaluarea surselor

trimise de utilizatori.

Odată ce exercițiul a fost creat, acesta este adăugat în baza de date și devine disponibil pentru toți utilizatorii, care pot încerca să îl rezolve și să încarce soluțiile lor.

4.4 Scenariul 4

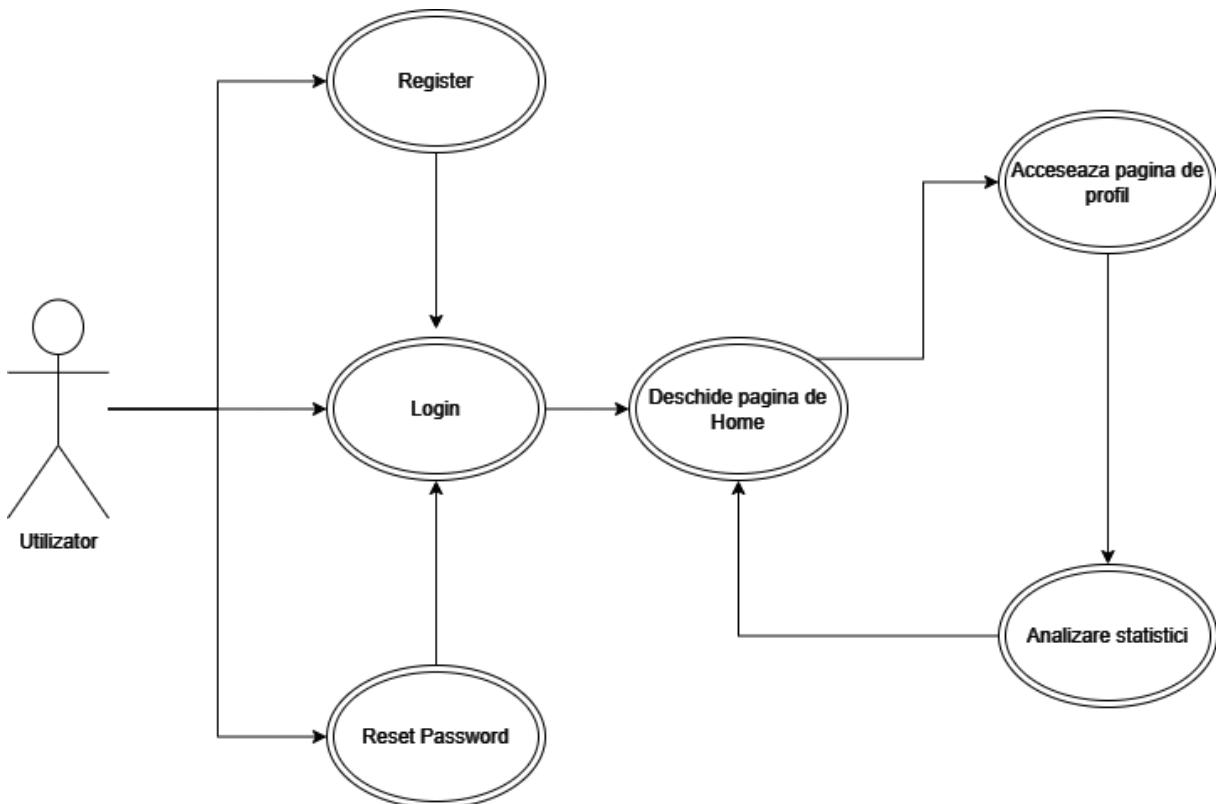


Figura 19: Scenariu de utilizare pentru vizualizarea statisticilor

În Figura 19 este prezentată diagrama UseCase pentru scenariul de utilizare al vizualizării statisticilor.

Un utilizator care are cel puțin o sursă încărcată își poate analiza performanțele accesând pagina numită „Profile”, disponibilă prin intermediul butonului poziționat pe bara de navigație. Această pagină poate fi accesată doar după ce utilizatorul s-a autenticat cu succes și este disponibilă atât pentru utilizatorii cu drepturi de administrator, cât și pentru cei fără aceste drepturi.

Statisticile oferă informații valoroase, ajutând utilizatorul să identifice atât punctele sale slabe, cât și pe cele forte. Având exercițiile catalogate, utilizatorul își poate vizualiza rata de succes a surselor trimise pentru o anumită categorie.

Dacă utilizatorul nu a încărcat nicio sursă asociată unui exercițiu dintr-o anumită categorie, acea categorie nu va fi vizibilă pe diagrama cu statisticile utilizatorului.

Aceste statistici sunt unice și private pentru fiecare utilizator, oferind informații precise referitoare la rata de succes pentru fiecare categorie cu care acesta a interacționat în rezolvarea exercițiilor.

Cu cât un utilizator a trimis surse pentru un număr mai mare de exerciții, cu atât statisticile sale devin mai complexe, oferind mai multe informații detaliate. Aceasta permite utilizatorului să aibă o perspectivă mai clară asupra progresului său și să identifice mai bine ariile care necesită îmbunătățiri.

4.5 Concluzii

Acest capitol vizează să ofere o perspectivă generală asupra funcționalităților principale ale aplicației, subliniind frecvența de utilizare a fiecărei funcționalități. Deși funcționalitățile de creare a exercițiilor și vizualizare a statisticilor pot fi utilizate mai rar, primele două funcționalități — rezolvarea exercițiilor și vizualizarea surselor — sunt cele mai frecvent utilizate și constituie unele dintre cele mai esențiale aspecte ale aplicației.

Concluzii

Concluzii Generale

Această lucrare descrie dezvoltarea aplicației FII Pregătit, de la motivația inițială până la soluția propusă, prezintând arhitectura, tehnologiile utilizate și scenariile de utilizare. Scopul principal al aplicației este de a eficientiza procesul de asignare și corectare a temelor pentru studenți, economisind astfel timp prețios care poate fi utilizat în moduri mai productive.

Aplicația este concepută pentru a fi rapidă, intuitivă și ușor de utilizat, oferind funcționalități accesibile atât studenților, cât și cadrelor didactice. Flexibilitatea sa permite adaptarea la un număr mare de discipline care implică scrierea de cod.

O funcționalitate cheie este posibilitatea de a crea mai multe fișiere sursă pentru un singur exercițiu, oferind posibilitatea de a trimite soluții complexe și menținerea unui cod curat și organizat.

Statisticile generate de aplicație pentru fiecare utilizator oferă un feedback valoros despre punctele slabe, ajutând utilizatorii să identifice domeniile care necesită atenție suplimentară. De asemenea, funcționalitatea de statistici pot ajuta la crearea unui mediu competitiv între studenți, motivându-i să își îmbunătățească performanțele.

Pentru cadrele didactice, un alt set de statistici relevante arată rata de succes a fiecărei probleme, permitându-le să identifice subiectele cu care studenții se confruntă și să se concentreze pe acestea în cadrul lecțiilor, pentru a îmbunătăți înțelegerea materiei.

În concluzie, consider că această soluție reprezintă un ajutor substanțial atât pentru studenți, cât și pentru cadrele didactice, aducând procesul de examinare și învățare la un nivel de eficiență timpului remarcabil.

Possible Îmbunătățiri

Deși aplicația FII Pregătit oferă funcționalități utile, există potențial pentru a deveni mult mai complexă.

Una dintre îmbunătățirile posibile este adăugarea mai multor limbaje de programare pentru evaluarea codului sursă trimis de utilizatori, inclusiv posibilitatea de a adăuga pseudocod, similar cu opțiunea disponibilă în aplicația Programare cu Răbdare prezentată în capitolul 1.

În plus, ar putea fi creată o secțiune dedicată resurselor, permitând cadrelor didactice să încarce cursurile pe această platformă, centralizând astfel toate resursele într-o singură aplicație.

O altă îmbunătățire posibilă este integrarea inteligenței artificiale pentru generarea automată de noi exerciții și nu numai, ceea ce ar mări considerabil potențialul de dezvoltare al aplicației.

Deși inițial a fost concepută pentru o singură instituție, aplicația are potențialul de a fi extinsă pentru a fi utilizată de mai multe instituții. Există posibilitatea de a adăuga o funcție care să listeze toate instituțiile care utilizează aplicația, necesitând ca utilizatorii să se înscrie printr-o invitație primită de la instituție pentru a accesa informațiile și exercițiile disponibile.

Aceste sugestii reprezintă doar o parte din posibilele îmbunătățiri, dar nu și singurele, deoarece există întotdeauna posibilitatea de a se găsi noi funcționalități ce pot fi implementate.

Bibliografie

- [1] <https://www.pbinfo.ro/>
- [2] <https://infoarena.ro/>
- [3] <https://probleme.programarecurabdare.ro/>
- [4] <https://www.geeksforgeeks.org/reactjs-introduction/>
- [5] <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs>
- [6] <https://www.techtarget.com/searchapparchitecture/definition/RESTful-API>
- [7] <https://www.geeksforgeeks.org/what-is-mongodb-working-and-features/>
- [8] <https://www.geeksforgeeks.org/compiling-with-g-plus-plus/>
- [9] <https://docs.docker.com/guides/docker-overview/>
- [10] <https://axios-http.com/docs/intro>
- [11] <https://www.freecodecamp.org/news/build-react-forms-with-formik-library/>
- [12] <https://browsee.io/blog/using-chart-js-in-react/>
- [13] <https://www.pluralsight.com/blog/software-development/react-bootstrap-explained>
- [14] <https://www.geeksforgeeks.org/what-is-react-select/>
- [15] <https://www.geeksforgeeks.org/what-is-react-router-dom/>
- [16] <https://www.dhiwise.com/post/a-guide-to-using-react-ace-for-efficient-coding>
- [17] <https://www.contentful.com/blog/yup-validate-forms-in-react/>
- [18] <https://www.npmjs.com/package/bcryptjs>

- [19] <https://www.npmjs.com/package/body-parser>
- [20] <https://www.geeksforgeeks.org/express-js/>
- [21] <https://www.geeksforgeeks.org/what-is-express-session-middleware-in-express>
- [22] <https://www.geeksforgeeks.org/mongoose-tutorial/>
- [23] <https://www.npmjs.com/package/connect-mongodb-session>
- [24] <https://www.geeksforgeeks.org/npm-cors/>
- [25] <https://www.geeksforgeeks.org/npm-dotenv/>
- [26] <https://www.turing.com/kb/comprehensive-guide-to-sending-an-email-using-nodemailer>
- [27] <https://www.geeksforgeeks.org/node-js-npm-uuid/>

*Toate link-urile au fost accesate ultima dată în Iunie 2024