

Platforma pentru evaluarea cunostintelor pe domenii

Profesor coordonator: Student:

**Ș.l. dr. ing. Mihai Vlase Sandu Alexandru**

**Introducere**

O platformă de evaluare este o aplicaţie (utilizată de regulă în spaţiul Web) complexă creată cu scopul testării cunoştinţelor acumulate într-un domeniu, abilităţilor motorii, psihomotorii, dar şi evoluţiei copiilor preşcolari conform nivelului specific de dezvoltare intelectuală, testarea capacităţii de efectuare a unor operaţii aritmetico-logice.

Platformele de evaluare prezintă un trend ascendent pe piaţa din România, ele fiind utilizate nu doar în scop educaţional, dar mai ales în domenii precum mediciă, inginerie şi multe altele. Practic, aproape că nu există domeniu al vieţii ştiinţifice ori sociale, în care evaluarea cu ajutorul unei aplicaţii – platformă să nu fie prezentă. Ele pot fi utilizate în cadru organizat, în interiorul unor grupuri de lucru, cu un scop comun sau individual, de curiozitate, de nevoie, ori de plictiseală.

In momentul de fata, trend-ul sa spunem asa este catre mediul online, cu tot mai multe site-uri si platforme fiind pe internet pentru a ajuta elevii si studentii sa se pregateasca la diferite materii/pe diferite domenii.

Doua astfel de exemple, din doua tabere sa spunem asa relativ diferite ar fi platforma AEL si platforma ExamenulTau, prima avand mai mult rol pedagogic, si fiind folosite de catre cadrele didactice, iar a doua fiind folosita mai mult de catre elevi pentru verificarea cunostintelor inainte de examen, spre exemplu.

Ambele au la baza parcurgerea unor cursuri sau lectii preliminare si apoi sustinerea unui test pentru verificarea sedimentarii cunostintelor respectivului.

Totodata, cu atatea surse de informatie la indemana prin intermediul internetului, deja nu mai revine in rolul platformei de evaluare sa fie totodata cea care se ocupa si de pare de instruire inainte de evaluarea elevilor/studentilor.

Astfel, se poate folosi efortul depus in crearea lectiilor si tutorialelor pentru imbunatatirea strategiilor si algoritmilor de evaluare, sau pentru acoperirea a mai multor arii.

**Justificarea temei / Motivatie**

Am ales crearea unei astfel de platforme intrucat consider ca poate ajuta elevii si studentii sa isi poate imbunatati cunsotintele de cultura generala, avand in vedere ca acestea nu sunt intotdeauna corelate cu studiile si totodata nu intotdeauna cunostintele si lectiile din sistemul de invatamant se translateaza mai apoi in mentalitatea individului ca si notiuni de cultura generala.

Implementarea aplicatiei este relativ necesara pentru persoanele foarte concentrate pe un singur domeniu, caz in care cunostintele din alte domenii cumva sunt destul de scazute, si se vrea schimbarea acestui aspect. Totodata, este util si pentru imbunatatirea cunostintelor pe un domeniu selectat, indiferent de nivel, intrucat se poate permite ajustarea nivelului de dificultate.

Ca si aspecte ce ar face diferenta de ce este deja pe piata, se poate scoate in evidenta in primul rand faptul ca, odata cu inregistrarea utilizatorului, cand acesta acceseaza platforma pentru prima oara, are ocazia sa sustina o evaluare initiala, in functie de care i se va stabili nivelul de dificultate de la care incepe.

Acesta este un lucru foarte benefic pentru utilizatorii care au deja un nivel in domeniul respectiv, si ii ajuta sa treaca peste primele capitole, relativ usoare pentur ei, comparativ cu nivelul estimat, lucru care de altfel i-ar putea plictisi. Un modul nou, menit sa confere aceeasi experienta initiala utilizatorilor, indiferent de nivelul lor initial.

Un alt lucru cu care vine ca si noutate este sistemul de knowledge safeguard, care nu permite utilizatorului sa sustina testul pentru promovare la nivelul urmator decat daca are scoruri bune la utlimele 3 sesiuni de training, Asta previne invatarea mecanica/norocul, intrucat intrebarile sunt aleatorii din baza de date, si deci se inlatura astfel probabilitatea ca utilizatorul sa promoveze la nivelul urmator fara sa aiba cunostintele necesare, facand astfel sistemul de evaluare cat mai stabil, complet si precis.

Totodata, problema cu platformele existente este faptul ca incearca sa implice si partea de dinainte de testarea propriu-zisa, care ar trebui sa fie scopul principal, insa pe platformele respective accentul este pus pe partea pedagogica si pe tutoriale, evaluarea cunostintelor la final fiind doar o formalitate.

Fiind astfel concentrate pe partea preparativa, de dinainte de evaluarea propriu-zisa, ele pierd insasi esenta, si anume testarea individului, punand accentul pe aspectele de dinaintea evaluarii. Acest lucru va fi evitat de aplicatia in cauza, intrucat aceasta nu are alt scop decat evaluarea elevilor si studentilor, iar acestia sunt responsabili singuri de pregatirea de dinainte de evaluare.

Un lucru inovativ cu care vine aceasta aplicatie este ca, prin conceptul de a se concentra doar pe evaluarea studentului, se lasa la latitudinea studentului/elevului ca acesta sa se pregateasca cum considera, cultivandu-se astfel simtul autodidact, de care acesta are nevoie din ce in ce mai mult pe parcursul ciclului didactic cat si dupa terminarea acestuia, in special in domeniile tehnice, si nu numai.

Totodata, platforma poate fi folosita si in scop recreativ, in sensul ca utilizatorii pot concura intre ei, cu scopul de a ajunge la un nivel cat mai avansat, oferindu-li-se astfel posibilitatea de a fi intr-o competitie recreationala, unde se stimuleaza totodata activitatea intelectuala a tinerilor.

Un lucru care are potentialul de a face aceasta aplicatie mai usor de utilizat fata de concurenta ar putea fi faptul ca are un design simplist, orientat pe functionalitate si rezultate, nu pe aspect. Majoritatea aplicatiilor au mult bloatware (soft si module aditionale), care de cele mai multe ori distrag cursantul de la scopul principal, si anume evaluarea cunostintelor.

In aceasta aplicatie, asemenea distrageri sunt evitate printr-un design minimalist, unde utilizatorul are doar strictul necesar, prezentat intr-o metoda cat mai lipsita de distrageri, astfel ca acesta se poate concentra pe scopul principal al platformei, si anume, evaluarea cunostintelor.

**Cerinte si specificatii**

Scopul general al aplicatiei este acela de a evalua si stimula prin testare cursantii pe diferite domenii de cultura generala.

Acest lucru se realizeaza pe baza unor nivele de dificultate, si avansarea cursantilor se face prin promovarea la urmatorul nivel de dificultate, in urma sustinerii unui test, acesta continand atat intrebari din nivelul curent cat si intrebari din toate nivelele anterioare. In momentul in care utilizatorul ajunge la un punctaj de peste 60%, atunci este promovat la nivelul urmator.

Promovarea la un nivel superior nu impiedica un cursant sa acceseze nivelele anterioare, sa sustina sesiuni de training sau evaluari la nivelele respective. Totodata, se mentioneaza faptul ca nici rezultatele la evaluari, nici la sesiunile de training nu se iau in calcul, in special nu se iau in calcul pentru nivelul curent al utilizatorului.

Totodata, platforma are un sistem de asigurarea consistentei rezultatelor, realizat prin introducerea sesiunilor de training. Acestea sunt o modalitate prin care utilizatorul poate sustine o versiune simplificata a examenelor, care contine doar intrebari din nivelul curent.

Aplicatia mentine inregistrate rezultatele de la ultimele trei sesiuni de training, astfel ca utilizatorul nu poate promova la nivelul urmator, ba chiar nu poate sustine examenul catre promovare, daca nu are un punctaj mediu de peste 60% realizat pe baza rezultatelor la ultimele trei sesiuni de training.

Intrebarile, atat pentru sesiunile de training, cat si pentru cele de examen, sunt alese in mod aleatoriu, la momentul sustinerii testelor, asigurandu-se astfel o evaluare cat mai impartiala si mai corecta a cursantilor.

La primul contact cu platforma, se va prezenta doar un ecran cu posibilitate de logare in aplicatie, si o optiune de inregistrare, in cazul in care utilizatorul este acolo pentru prima oara si nu beneficiaza inca de un cont pe platforma.

La inregistrarea cursantilor, se va specifica un nume de utilizator specific, care trebuie sa nu fie deja luat de catre un alt cursant, parola, impreuna cu confirmarea acesteia, precum si domeniul de cunostinte ales de catre utilizator.

Dupa logarea utilizatorilor pe platforma, se ajunge la un landing page, cu informatii generale atat despre utilizator, cat si despre aplicatie. Deasemenea, de pe aceasta pagina de home utilizatorul isi poate schimba domeniul, avand totusi ca amendament faptul ca, odata schimbat domeniul, cursantul isi va pierde nivelul de dificultate la care a ajuns, acesta refiind resetat la zero.

In partea din stanga a paginii sunt situate butoanele folosite pentru navigarea pe platforma, si anume in zona superioara butoanele pentru ajungerea pe pagina de home, butonul folosit pentru delogare, precum si butonul pentru pagina de administrare, in cazul administratorilor.

Mai apoi, in zona inferioara, se gasesc butoanele pentru navigarea catre diferitele nivele de dificultate ale aplicatiei. Toate butoanele pentru accesul la nivele sunt vizibile inca de la inceput, insa acestea nu devin functionale decat in momentul in care cursantul a reusit sa promoveze la nivelul respectiv.

Dintre cerintele si specificatiile aplicatiei, un aspect extraordinar de important, care diferentiaza aplicatia de restul concurentei, este faptul ca cursantii, la logarea pe platforma, in cazul in care este prima oara cand acestia se conecteza, utilizatorii pot sustine o evaluare initiala, prin care li se poate stabili nivelul initial de cunostinte in domeniul ales.

Acest aspect este foarte important, intrucat prin stabilirea nivelului initial, se poate evita parcurgerea primelor nivele de catre cursantii cu un nivel ceva mai ridicat de cunostinte, lucru care ar fi putut sa ii plictiseasca sau sa ii faca sa isi piarda interesul in platforma, intrucat aceasta nu ii stimuleaza la nivelul adecvat.

Totodata, se pune accentul si pe controlul administratorilor asupra continutului platformei, astfel ca acestia au o pagina separata pentru activitatile specifice de administrare ale aplicatiei. Aceasta pagina permite administratorilor sa adauge noi intrebari pentru diferite nivele si domenii, totodata editarea celor deja existente precum si stergerea acestora.

Deasemenea, un administrator poate crea noi domenii pe care sa le foloseasca utilizatorii, si tot acesta poate sterge domenii. Astfel, administratorii au control asupra continutului bazei de date cu intrebari si domeniile acestora, asigurandu-se astfel o experienta cat mai completa si mai buna pentru utilizatori.

**Analiza problemei**

**Tipuri de utilizatori**

Aplicatia are ca target principal utilizatorii tineri, eventual in diferite stagii ale ciclului de invatamant, care vor sa isi testeze cunostintele in diferite domenii.

Acestia de cele mai multe ori aleg aplicatia pentru a isi imbunatati pregatirea pentru un anumit domeniu fie in scop didactic, cum ar fi pregatirea inainte de un examen la o materie pe domeniul respectiv, cat si in scop recreational, intr-o competitie amicala cu colegii spre exemplu, cu scopul de a ajunge la un nivel de dificultate cat mai avansat.

Totodata, aplicatia poate fi folosita si de utilizatori ceva mai inaintati in varsta, datorita mecanismului complet de asigurare a consistentei evaluarii cursantului, astfel ca firecare examen pentru promovarea la nivelul urmator implica atat sesiuni de training, cat si evaluarea propriu-zisa contine intrebari din nivelele anterioare.

Pe langa utilizatorii cu drepturi normale, mai avem o alta categorie, total diferita, si anume administratorii. Acestia au rolul de moderatori ai aplicatiei sa spunem asa, in sensul ca acestia controleaza contentul platformei. Ei pot deasemenea actiona ca si utilizatorii standard, dar pe langa asta au si optinea unui panel administrativ de unde pot adauga, sterge sau edita intrebari, precum si domenii de cunostinte.

**Elementele aplicatiei**

Elementele esentiale ale aplicatie se contureaza dupa functionalitatile platformei si dupa scopul acesteia. Astfel, se contureaza ca si entitati ale aplicatiei insasi intrebarile, ele fiind unitatea functionala, si caracateristici auxiliare ale acestora, cum ar fi domeniile de activitate.

Totodata, acestea sunt structurate si grupate sub forma de diferite tehnici de evaluare, acestea fiind trei la numar, constituite de evaluarea initiala, sesiunea de training si evaluarea pentru promovarea la urmatorul nivel de dificultate.

Evaluarea initiala are ca scop principal realizarea unui examen preliminar in care se stabileste nivelul cursantului de dinainte de a avea contact cu paltforma. Rolul acestei evaluari este acela de a facilita accesul la platforma al utilizatorului de la un nivel cat mai apropiat al acestuia de cel deja existent, astfel evitandu-se plictisirea sau pierderea interesului prin fortarea lui in a trece prin primele nivelel, in cazul in care acesta poseda deja cunostinte ceva mai avansate in domeniul respectiv. Aceasta evaluare se realizeaza printr-un sistem care se auto-ajusteaza in functie de utilizator, astfel ca mereu urmatoarea intrebare va fi ajustata la un nivel superior sau inferior fata de cea precedenta in functie de raspunsul cursantului.

Sesiunile de training au rolul de a sedimenta cunostintele cursantului, prin executarea unor evaluari la oscara ceva mai mica, continand doar intrebari din nivelul curent. Acestea sunt totodata mecanismul prin care platforma asigura un anumit nivel de consistenta al cunostintelor utilizatorului, astfel ca acesta nu poate sustine examenul pentru promovare la urmatorul nivel pana nu a are o medie de peste 60% la utlimele trei sesiuni de training la nivelul respectiv.

Testul pentru promovarea la urmatorul nivel al cursantului este testul prin care utilizatorul are sansa, dupa ce a dovedit ca stapaneste nivelul curent prin acumularea unui punctaj de trecere la sesiunile de training, sa sustina examenul si, prin luarea unui puncatj satisfacator, sa promoveze la urmatorul nivel de dificultate. Totodata, acest examen contine si intrebari aleatorii din nivelele anterioare, astfel ca se asigura o evaluare cat mai completa si precisa a cursantului, asigurandu-se ca acesta stapnaeste in continuare si nivelele anterioare.

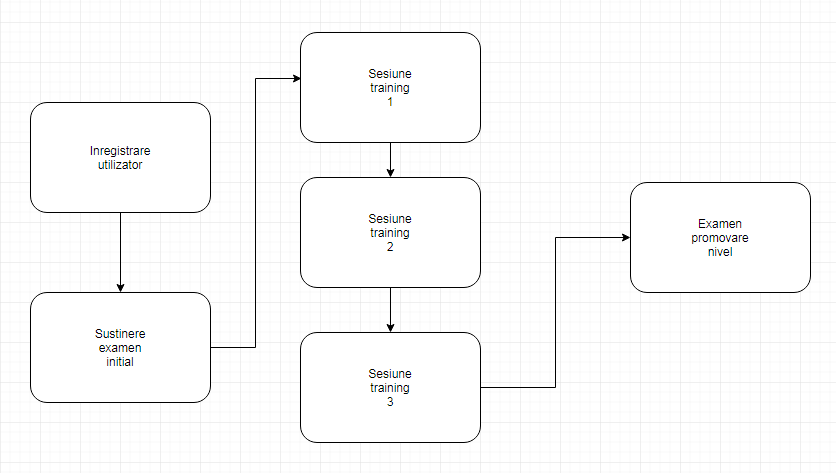
Intrebarile, ca si strucutra, au enuntul, insasi intrebarea in sine, din domeniul din care face parte si nivelul de dificultate caruia ea apartine, si apoi trei variante de raspuns, dintre care una dintre ele este cea corecta. Aceasta organizare este folosit pentru simplificarea ramificarii algoritmicii de evaluare, si totodata pentru a familiariza usor cursantul cu functionalitatea structurii de evaluare.

Domeniile constituie gruparea dupa context a intrebarilor, si sunt baza formarii diferitelor categorii de intrebari, constituind un criteriu important de diferentiere a acestora, impreuna cu nivelele de dificultate.

Nivelele de dificultate ale platformei de evaluare constituie al doilea criteriu principal de diferentiere a intrebarilor, si nu numai, intrucat acesta este mijlocul de departajare a insasi utilizatorul, si anume nivelul de dificultate curent al acestora.

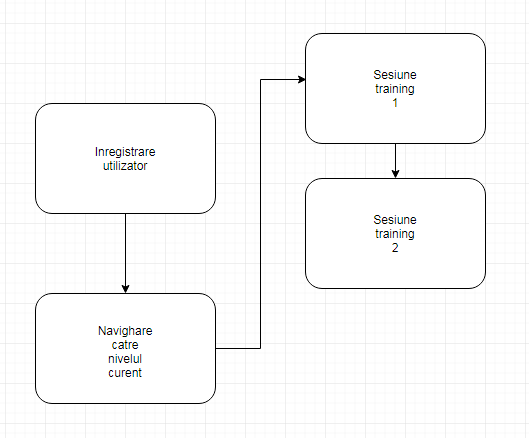
**Cazuri de utilizare**

1. Inregistrarea unui utilizator pentru prima oara pe platforma, sustinerea examenlui initial de catre acesta si eventual executia sesiunilor de training si apoi a examenului pentru promovarea la nivelul urmator.

Acesta este cazul principal de utilizare al aplicatiei, in care utlizatorii noi iau contact pentru prima oara cu aplicatia, si de cele mai multe , ajung sa parcurga mai toate flow-urile majore ale aplicatiei.

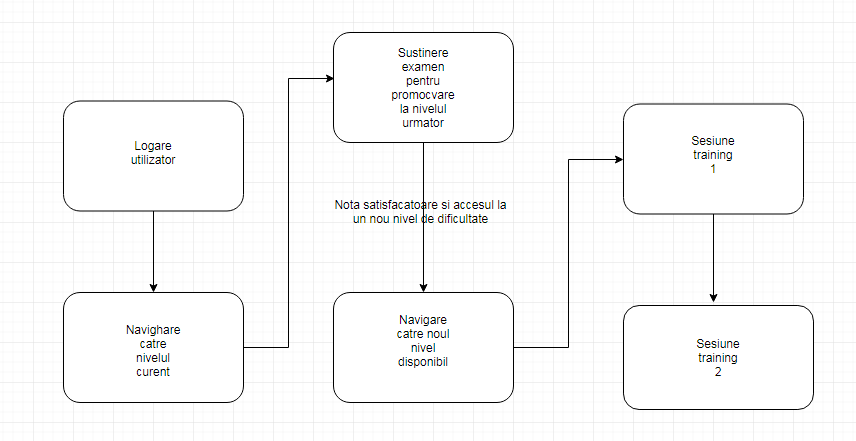
2. Logarea unui utilizator deja existent pentru executarea catorva sesiuni de training pentru nivelul curent.

Acesta este cazul cel mai des intalnit pe platforma, in care utilizatorul viziteaza zilnic aplicatia pentru a se antrena, executand cateva sesiuni de training pentru a se pregati de un viitor eventual examen de promovare la nivelul urmator, in momentul in care acesta se simte pregatit sa il sustina, si desigur, intruneste criteriile necesare, si anume sa aiba un punctaj mediu de peste 60% la ultimele trei sesiuni de training.



3. Sustinerea examenului de promovare la nivelul urmator de catre cursant si executarea unei sesiuni de training pe noul nivel disponibil.

Acesta este un caz de utilizare destul de comun intre cursanti, implicand sustinerea examenului de promovare la urmatorul nivel de dificultate, si odata luat acesta cu success, utilizatorul in cele mai multe cazuri executa o sesiune sau doua training pe nould nivel de dificultate pentru a-si face o idee despre noua dificultate la care acces acum.



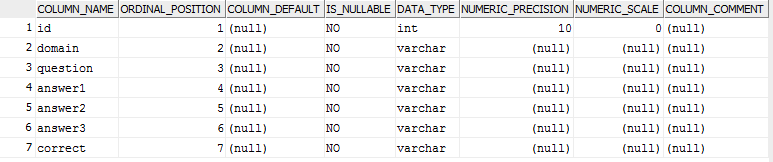
**Proiectarea bazei de date**

Pentru proiectarea bazei de date s-a folosit ca si arhitectura un sistem descentralizat pentru stocarea intrebarilor si datelor utilizatorilor, in sensul ca nu s-au folosit chei secundare si nu s-au legat tabelele la nivel relational, pentru evita dependintele intre date si blocarea executiei unor operatii tranzactionale pe baza legaturilor datelor respective cu alte tabele.

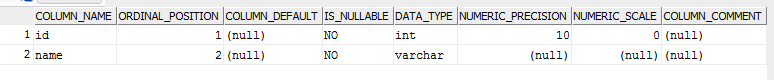
Astfel, s-a folosit un sistem de baze de date relational, avand la baza ca si stack de tehnologii MySQL , urmand mai apoi ca tabelele respective sa fie mapate in entitati la nivel logic pentru implementarea algoritmilor prin framework-ul Spring, in limbajul Java.

Baza de date a aplicatiei contine ca si entitati tabele pentru domenii, utilizatori, precum si tabele pentru intrebari, acestea fiind structurate pe nivele, astfel, exista atatea tabele de intrebari cat nivele de dificultate.

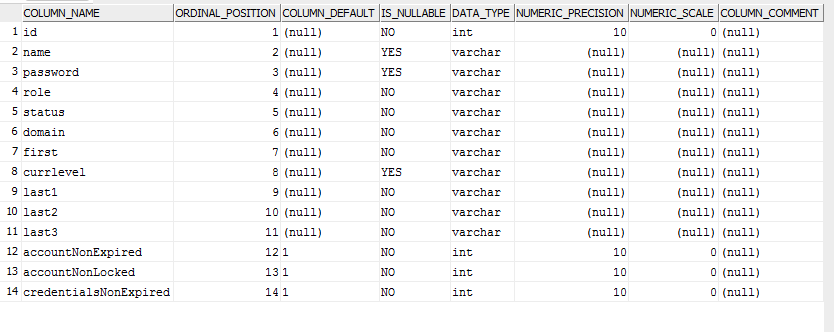
Tabelele unde sunt stocate intrebarile sunt organizate pe nivele, si structura coloanelor acestora consta intr-un id unic, care se autoincrementeaza, domeniul intrebarii, enuntul acesteia, precum si cele trei variante de raspuns, si totodata un camp aditional unde avem raspunsul corect, acesta trebuind sa coincida cu una din cele trei variante de raspuns.



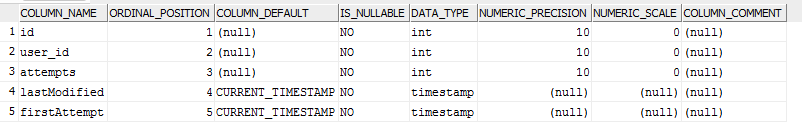
Domeniile sunt intr-un tabel separat, ca si entitate, intrucat au un scop vast atat in administrarea intrebarilor, cat si conturilor utilizatorilor.



O alta entitate complexa a bazei de date este tabela de utilizatori, unde se tin datele personale ale cursantilor si totodata statistici ale acestora specifice platformei, dintre care amintim ca si exemplu nivelul curent sau ultimele trei rezultate la sesiunile de training, fara de care nu s-ar putea face evaluarea pentru acceptanta sustinerii examenului de promovare la nivelul urmator.

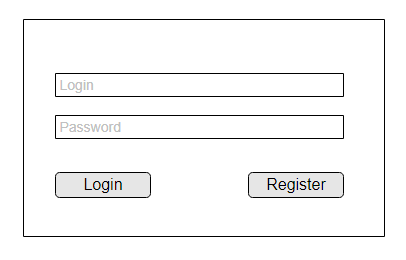


O tabela aditionala introdusa in schema bazei de date este tabela folosita pentru securitatea platformei, cu scopul principal de a tine evidenta incercarilor utilizatorului de a se loga, folosindu-se mai apoi o implementare in aplicatia pentru a preveni logarea gresita a utilizatorului pentru un numar prea mare de incercari.

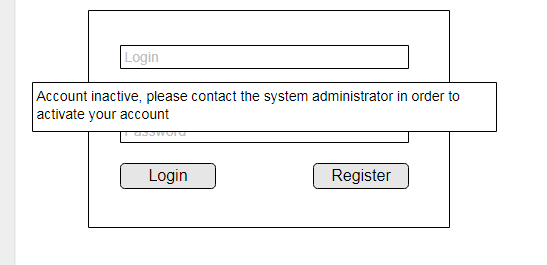


**Proiectarea interfetei**

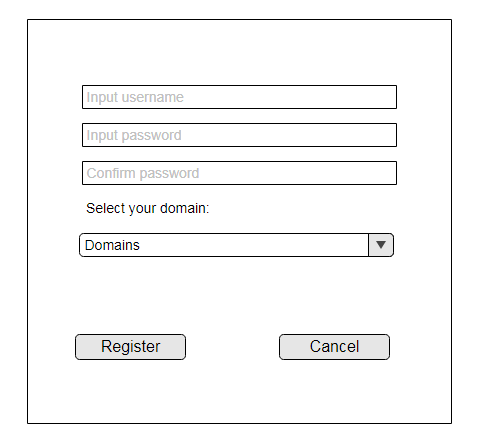
Proiectarea interfetei incepe in primul rand cu interfata cu care ia contact pentru prima oara utilizatorul, si anume interfata de logare. Aceasta trebuie sa fie simpla, minimalista, si sa contina campurile pentru introducerea numelui de utilizator, parola acestuia si butonul de login. Aditional, aceasta contine si un buton pentru inregistrare, in cazul in care cursantul nu are inca un cont inregistrat pe aceasta platforma.



Totodata, in cazul in care utilizatorul a depasit numarul de incercari de logare disponibile, el va fi atentionat printr-un banner informativ ca acum contul lui este blocat si si trebuie sa vorbeasca cu un admin pentru a i se activa din nou.



Pentru interfata de inregistrare, o interfata prin care vor trece toti utilizatorii standard, intrucat acestia trebuie sa isi creeze cont pentru a putea accesa platforma, structura acestei interfete consta in asezarea campurilor folosite pentru alegerea numelui de utilizator, parola, confirmarea acesteia , precum si o lista cu domeniile disponibile pentru intrebari, mai apoi venind butoanele pentru register si cancel.

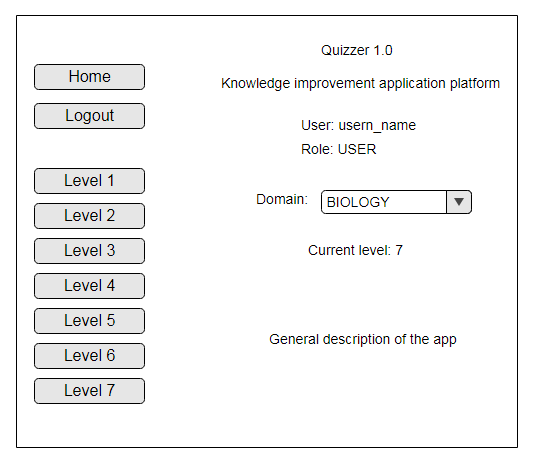


Dupa apasarea butonului pentru inregistrare, se va verifica daca valorile introduse pentru campurile pentru parola si respectiv confirmarea ei coincid, si restul criteriilor sunt indeplinite, cum ar fi numele de utilizator sa nu fie deja existent.

Odata logat, utilizatorului va avea acces la platforma, unde in zona superioara din stanga sunt butoanele administrative, respectiv butonul pentru home, logout, eventual si butonul pentru administratie, in cazul in care utilizatorul este administratorul.

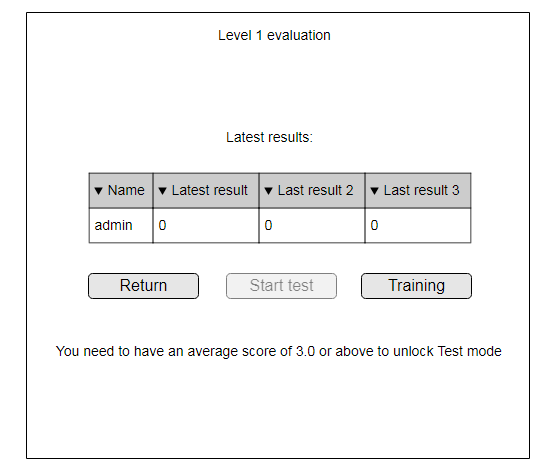
In zona inferioara din stanga avem prezente butoanele pentru navigarea pe diferite nivele de dificultate.

In zona din dreapta a landing-page-ului avem o descriere sumara a aplicatiei si totodata un combobox de unde utilizatorul isi poate schimba domeniul pentru intrebari.



Interfata pentru fiecare nivel in partea va avea un tabel ce contine numele utilizatorului si rezultatele acestuia la ultimele trei sesiuni de training, precum si butoanele pentru start test, in cazul in care criteriile sunt indeplinite, butonul de training, pentru pornirea unei sesiuni de training, si totodata butonul de return, pentru a intoarce utilizatorul la pagina de home.

In partea de jos a ecranului se va afisa si o nota informativa in legatura cu nota medie la care trebuie sa ajunga studentul pentru a putea sustine examenul pentru promovarea la urmatorul nivel de dificultate.



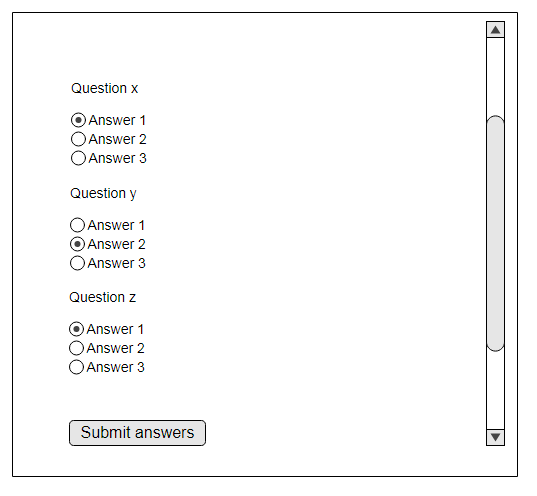
Interfata de utilizator responsabila pentru teste in sne, fie ele cele de promovare la nivelul urmator, sesiuni de training sau testarea initiala a userului, trebuie sa aiba o strucutra cat mai minimalista si mai simpla.

Astfel, se asigura o experienta cat mai lipsita de distrageri pentru cursant, scopul principal fiind acela de a-l ajuta sa se concentreze cat mai eficient pe toata perioada sustinerii testului, indiferent de scopul acestuia.

Folosindu-se o astfel de structura, intrebarile sunt organizate intr-o lista, avand enuntul in partea superioara si mai apoi dedesubt cele trei variante posibile de raspuns, cu o singura varianta posibila fiind corecta.

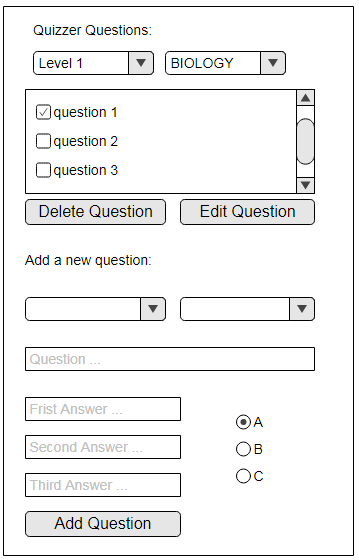
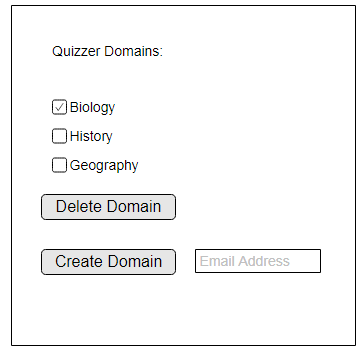
O astfel de asezare simplificata face mai usoara familiarizarea cursantului cu platforma si totodata il ajuta sa se poata concentra cat mai usor pe verificarea notiunilor respective.

Deasemenea, dupa parcurgerea listei de intrebari, cursantul va gasi in partea inferioara a paginii butonul folosit pentru trimiterea raspunsului si inceperea procesului de validare al acestora, astfel ca acesta mai apoi primeste intr-o nota informativa pe pagina viitoare rezultatul in urma sustinerii testului si eventuale remarci cu privire la performanta acestuia.



Aditional, pentru proiectarea interfetei de utilizator folosita de catre administratori, acesta va avea o zona pentru adaugarea/stergerea de domenii, precum si o zona similara, folosita pentru stergere sau editare a intrebarilor, cu posibilitatea de a le filtra dupa domeniul acestora si nivelul de dificultate.

Totodata, panel-ul administrativ va contine si o zona desemnata pentru adaugarea de noi intrebari, unde adminul va specifica exact enuntul acesteia, precum si variantele de raspuns, impreuna cu cea corecta.



**Tehnologii**

In ceea ce priveste stack-ul de tehnologii folosit la aplicatie, pentru proiectarea bazei de date s-a folosit MySQL ca si sistem de gestionare a bazelor de date relationale, fiind o alternativa destul de lightweight la Oracle, cu o sintaxa similara.

Pentru proiectarea efectiva a aplicatiei si a arhitecturii serviciilor folosite de aceasta s-a folosit pentru programare limbajul Java, datorita scalabiliatatii oferite, suportului foarte bun pentru partea de multi-threading si totodata datorita portabilitatii datorate JVM-ului, facand astfel aplicatia cross-platform compatible.

Ca si framework-uri aditionale implementarii in Java, pentru partea de back-end a aplicatiei Spring a fost nucleul sa spunem asa, folosindu-ne de facilitatile acestuia pentru a exercita cat mai clar si eficient logica aplicatiei la nivel programatic.

Spring este un framework redus ca dimensiuni, dar în acelaşi timp flexibil şi universal, orientat spre a facilita crearea de aplicații Java SE şi Java EE. Spring Framework urmăreşte să ofere infrastructura necesară pentru realizarea aplicațiilor Java, astfel încât programatorul să se poată concentra asupra dezvoltării efective de cod. Spring utilizează pe scară largă obiecte Java clasice (POJOs – Plain Old Java Objects).

La bază, o aplicație Java constă din obiecte care colaborează între ele pentru soluționarea unei probleme – obiectele dintr-o aplicație având dependențe între ele.

Platforma Java nu dispune de mijloacele necesare pentru a organiza obiectele, această sarcină revenind dezvoltatorilor şi arhitecților care au la dispoziție pattern-urile clasice (design patterns) pentru a organiza clasele şi obiectele. Totuşi, acestea trebuie implementate de către programator.

Facilitatea de inversiune a controlului (Inversion of Control – IoC) pusă la dispoziție de Spring Framework adresează această problemă – organizarea obiectelor – oferind mijloacele necesare pentru a permite componentelor disparate să fie asamblate într-o aplicație funcțională.

S-a folosit Hibernate ORM (Object-Relational Mapping) pentru maparea tabelelor din baza de date la nivel de entitati in contextul Spring, pentru o translatare cat mai usoara mai apoi in obiecte Java pentru o prelucrare cat mai usoara a informatiilor din baza de date.

Object / Relational Mapping (ORM) este o tehnică de programare ce face posibilă accesarea și manipularea obiectelor fără ca programatorii să fie interesați de sursa de date de unde provin aceste obiecte. Această tehnică a apărut din nevoia de a depăși diferențele de paradigmă dintre modelul orientat pe obiecte (susținut de limbajele de programare de nivel înalt actuale) și modelul relațional (utilizat de cele mai populare sisteme de gestiune a bazelor de date).

Limbajele de programare orientate pe obiecte reprezintă datele într-un graf interconectat de obiecte, pe când bazele de date relaționale folosesc un mod tabelar de reprezentare. Efortul de a conecta atributele claselor definite prin intermediul unui limbaj orientat pe obiecte cu câmpurile tabelelor din baza de date nu poate fi ignorat, iar scopul unui ORM este acela de a crea o relație naturală, transparentă, fiabilă și de durată între cele două modele. Această nepotrivire de paradigmă pare să nu își fi găsit încă o soluționare definitivă care să fie aprobată de toți actorii din industria IT, însă opinia generală este aceea că framework-urile ORM reprezintă un important pas înainte.

Procesul automat de stocare a obiectelor într-o bază de date relațională folosind un framework ORM, constă în maparea obiectelor la tabelele corespunzătoare, asociereadintre ele fiind descrisa folosind metadata**.**

Hibernate reprezintă o soluție de tip ORM pentru asigurarea persistenței și a interogării eficiente a informațiilor stocate în baze de date relaționale, prin intermediul limbajului de programare Java. Este un produs open-source ce realizează maparea dintre tabelele bazei de date și clase Java, asigurând astfel interacțiunea dintre sistemeul de gestiune pentru baze de date și serverul de aplicații ce implementează logica aplicației.

Mecanismul prin care sunt reprezentate legăturile dintre baza de date și clasele Java este reprezentat de fișiere XML, astfel încât nu este necesară dezvoltarea de cod sursă pentru obținerea unei astfel de funcționalități. Astfel, în situația în care apare o schimbare la nivelul bazei de date sau la nivelul claselor Java, este suficient ca aceasta să fie realizată și la nivelul fișierului XML corespunzător pentru ca legătura dintre entități să fie menținută. API-ul pus la dispoziție atât pentru asigurarea persistenței cât și pentru interogarea diferitelor informații din baza de date este flexibil și scalabil, fiind accesibil tuturor programatorilor, indiferent de nivelul de pregătire. Se oferă o abstractizare a operațiilor de la nivelul sistemului de gestiune pentru baze de date, astfel încât detaliile de implementare de la nivelul acestuia nu trebuie să fie cunoscute, interacțiunea făcându-se la nivelul obiectelor Java ce corespund unor concepte specifice afacerii sau culturii organizației respective. Pot fi modelate astfel legături oricât de complexe dintre tabelele entitate de la nivelul bazei de date. Mai mult, accesul la date se realizează optimizat, folosindu-se diferite strategii inteligente pentru interogarea informațiilor, menținând în același timp ușurința în utilizare. De asemenea, poate fi utilizat independent de un server de aplicații.

Pentru operatiile la nivel de baze de date cum ar fi adaugarea, stergerea sau modificarea datelor din tabele s-au folosit JPA Repositories ale Spring, astfel ca fiecare tabel are corespondent un repository la nivel de Java ce permite efectuarea unor astfel de operatiuni, si totodata permite adaugarea unor operatiuni noi, definite de ingineri, prin adaugarea noii metode folosite si adnotarea acesteia cu query-ul pe baza de date care se vrea a fi efectuat.

JPA (Java Persistence API) reprezintă o colecție de clase (în limbajul de programare Java), oferind funcționalități legate de interogarea eficientă a informațiilor stocate în baze de date relaționale, mai ales în situația în care volumul acestora este considerabil. Această interfață de programare implementează un nivel de abstractizare, toate informațiile fiind stocate la nivel de obiecte, astfel încât programatorul nu este obligat să cunoască detaliile specifice unui sistem de gestiune al bazelor de date. Prin urmare, JPA realizează o punte între informațiile reprezentate relațional - așa cum sunt reținute într-o bază de date - și sub formă de instanțe ale unor clase, permițând un transfer facil între aceste moduri de organizare, implementând în același timp funcționalități de acces la date prin diferite mecanisme de interogare.

JPA reprezintă un mecanism prin care entitățile care își găsesc un corespondent în logica aplicației sunt convertite în date relaționale și viceversa. Acesta descrie modul în care pot fi descrise obiectele de tip POJO (Plain Oriented Java Object) precum și modul în care pot fi gestionate relațiile dintre acestea.

Spre exemplu, un obiect de tip Entity reprezintă o entitate persistentă, fiind utilizat pentru stocarea și încărcarea informațiilor din baze de date relaționale. Acesta deține un singur atribut (opțional) - name, denumirea entității respective. De regulă, se folosește ca adnotare pentru a indica faptul că trebuie asigurată persistența clasei în cauză, la nivelul unei tabele în baza de date.

Pentru partea de front-end a aplicatiei s-a folosit ca si framework cu implementare in Java Vaadin, un framework relativ similar cu Swing, avand ca diferenta esentiala aplicabilitatea acestuia la aplicatiile pe o platforma Web, cum este cazul aici, astfel ca Vaadin translateaza la runtime codul implementat in Java in cod JavaScript, pentru a putea fi interpretat de browser. Majoritatea componentelor folosite sunt standard, folosindu-se astfel un design simplist.

Pentru organizarea efectiva a proiectului, pentru organizarea dependintelor, plugin-urilor, librariilor si tuturor configurarilor necesare, precum si pentru orchestrarea instalarii acestora, a artefactelor si a deploy-ului aplicatiei s-a folosit ca si tool principal Maven, acesta ajutand totodata si la organizarea structurii directoarelor, avand la radacina proiectului ca si fisier esential fisierul pom.xml, bazandu-se pe conceptul Project-Object-Model, aceasta fiind unitatea principala pe care se bazeaza Maven cand executa goal-uri, acestea fiind actiuni specifice, cum ar fi clean, install, deploy, etc.

Maven este un sistem de build și management al proiectelor, scris în Java. Face parte din proiectele găzduite de Apache Software Foundation. Funcționalitățile sale principale sunt descrierea procesului de build a softwareului și descrierea dependențelor acestuia. Un fișier XML descrie proiectul care urmează să fie build-uit, dependențele acestuia sau ale module și componente de care depinde, ordinea în care se execută build-ul, directoarele și plug-in-urile necesare. Maven descarcă dinamic bibliotecile Java si plug-in-uri necesare, din unul sau mai multe repository-uri.

Pentru deploy-ul aplicatiei s-a folosit ca si webserver Tomcat, intrucat are o compatibilitate excelenta cu tehnologiile bazate pe JVM si este foarte eficient pentru orchestrarea deploy-urilor.Apache Tomcat, de altfel cunoscut sub numele de „Tomcat” este un server web open-source, fiind dezvoltat și menținut tot de Apache Software Foundation. Este o implementare software pentru tehnologiile Java şi rulează pe Java Servlets și pagini JavaServer, cunoscute sub numele de JSP-uri.

Ca si tool de versionare, folosit pentru a avea in primul rand siguranta unui back-up al aplicatiei, si totodata posibilitatea de a face rollback la o versiune anterioara, in cazul in care au aparut probleme in cea curenta, s-a folosit Git, intrucat este un tool la indemana, open-source, cu multe resurse de informare asupra lui pe internet, care este totodata si foarte versatil.

Un sistem de versionare este un mod de management al fișierelor care permite păstrarea istoricului tuturor modificărilor aduse fișierelor urmărite.

GitHub, serviciul folosit in acest caz, este un serviciu de hosting al proiectelor git (un proiect git este numit repository). Acesta permite păstrarea unei copii a proiectului online și vizualizarea tuturor fișierelor și a modificărilor aduse acestora.

**Implementarea aplicatiei**

Primul stagiu al implementarii aplicatiei a fost alegerea tehnologiilor folosite pentru dezvoltare, detaliate in capitolul Tehnologii, a tool-urilor aditionale implementarii, cum ar fi IDE-ul si workbench-ul pentru baza de date.

Dupa alegerea tehnologiilor si instalarea/configurarea acestora, urmatorul pas a fost configurarea IDE-ului, in cazul acesta fiind folosit IntelliJ IDEA, un IDE bazat pe Java, pentru dezvoltare de aplicatii complexe folosind tehnologiile Java. Pentru accesul si modificarile aditionale asupra bazei de date, ca si tool a fost folosit in faza incipienta mysqld, tool-ul default cu care vine baza de date MySQL, ulterior, pentru opratii mai complexe fiind ales SQL Developer, un tool dezvoltat de Oracle, cu un plugin pentru MySQL.

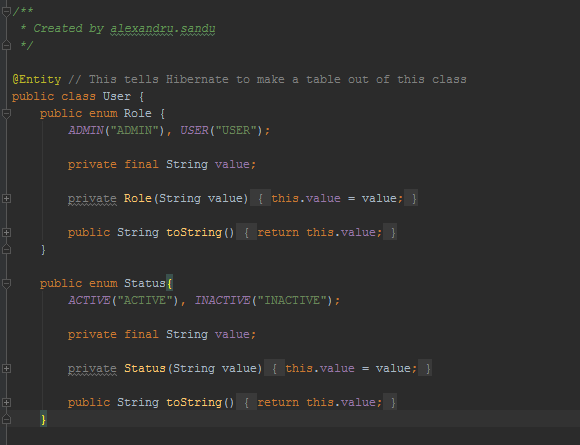
Ca un prim pas in inceperea dezvoltarii aplicatiei avem proiectarea bazei de date, intrucat aceasta constituia fundatia fara de care nu s-ar fi putut continua dezvoltarea tuturor celorlalte module ale aplicatiei.

Dupa hotararea asupra arhitecturii bazei de date, asupra tabelelor care vor fi folosite, s-a continuat cu adauagrea efectiva a acestora in baza de date. Detalii aditionale cu privire la structura tabelelor si detalii specifice se pot gasi la capitolul pentru Proiectarea bazei de date.

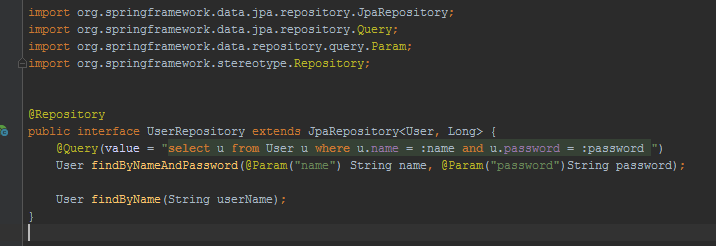
Urmatorul pas, dupa stabilirea bazei de date, a fost reprezentat de construirea scheletului aplicatiei. Acesta lucru a fost facilitat de folosirea framework-ului SpringBoot, astfel ca folosind tehnologiile specificea acestuia, am adaugat in fisierul specific Maven, si anume pom.xml dependintele necesare pentru rularea aplicatiei, diferite librarii si plugin-uri necesare pentru dezvoltare, si eventuale guidelines cu privire la artefactele rezultatele in urma compilarii si asa mai departe. Totodata, in acest stagiu s-au adaugat configurarile necesare pentru framework-ul de Spring, prin fisierul de application.properties .

Dupa stabilirea unei structuri incipiente a aplicatiei, urmatorul pas a fost construirea fundatiilor aplicatiei pentru parte de back-end.

Acestea au fost reprezentate in prima faza de maparea tabelelor din baza de date ca si entitati in contextul Spring, create la pasii anteriori la nivel de obiecte Java, pentru a putea exercita la nivel programatic operatiuni si business logic asupra lor cu usurinta.

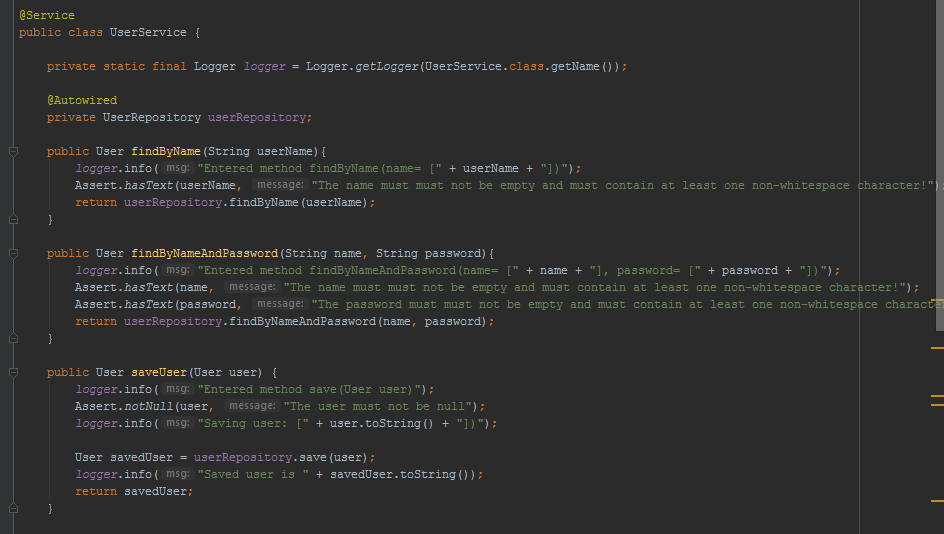


Totodata, acesta este pasul in care au fost create si repositoarele bazate pe JPA, facilitandu-se astfel executarea operatiunilor asupra bazei de date. Felul in care acestea au fost folosite in arhitectura aplicatiei a constat in adaugarea unor metode si functii la nivelul lor, acestea fiind programate sa corespunda unor query-uri parametrizate, astfel ca de fiecare data cand o metoda era chemata din limbajul Java, aceasta de fapt in spate apela aceste repository-uri pentru executia unor query-uri stabilite de dezvoltatori asupra bazei de date.



Astfel, am putea spune ca maparea tabelelor la nivel de entitati este o mapare informationala, iar maparea query-urilor pe baza de date in metode ale repositoarelor este o mapare functionala.

Tot in acest stagiu au fost create si serviciile aplicatiei, aceasta actionand in arhitectura aplicatiei ca niste providere de informatii, ele reprezentand de fapt o grupare asupra unor functii aditionale, un layer aditional de abstractizare am putea spune asupra repositoarelor, astfel ca, spre exemplu, toate intrebarile, indiferent de nivelul de dificultate al acestora sau de domeniul de care apartin, pot fi accesate sau modificate drect dintr-un sngur nivel, in cazul acesta, level service, creat special pentru acest scop.



Designul aplicatiei, bazat pe acest tip de back-end implica un EntiyManager care va crea niste bean-uri (beans - unitatea structurala a contextului Spring) pe baza entitatilor definite programatic de catre dezvoltator, acesta putand fi apoi initializate si totodata accesate de oriunde din contextul aplicatiei, folosindu-se conceptul injectiei de dependinte (dependency injection). Astfel, printr-o adnotare de tipul @Autowired, un repository declarat in contextul aplicatiei este legat (wired) automat de entitatea corespondenta, fara a mai fi nevoie de adaugarea unor configuratii aditionale.

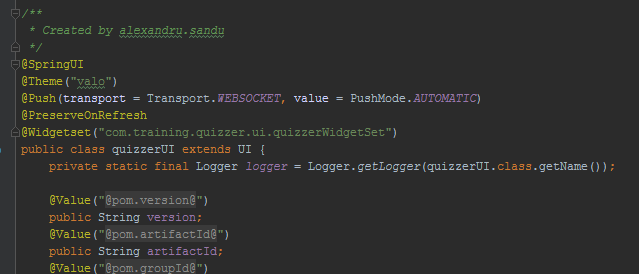
Folosindu-se o arhitectura a aplicatiei bazate pe obiecte, multi-threading, operatiuni asincrone, si context, si in primul rand o aplicatie bazate pe interfata cu utilizatorul, s-a folosit ca si concept principal conceptul de inversare a controlul (IOC- Inversion of Control).

Printr-un astfel de concept, spre deosebire de o aplicatie dezvoltata in C spre exemplu, unde dezvoltatorul foloseste ca si tehinca de programare programarea imperativa, spre deosebire de cea orientata pe obiecte, deci aplicatia in sine controleaza flow-ul aplicatiei, aici flow-ul aplicatiei este dictat de utilizator, in functie de ce doreste acesta sa activeze. Acestuia nu i se poate dicta ce comportament sa aiba, ce flow-uri sa activeze, ci decide singur asuprea aceastor lucruri, aplicatia doar punandu-i la dispozitie diferitele alegeri posibile pe care acesta le poate face.

Odata creat back-endul aplicatiei, urmatorul pas a fost implemetarea modului pentru front-end. Aici, folosind ca si framework Vaadin, bazat si acesta tot pe Java, s-a putut realiza o integrare foarte facila cu celelalte componente, bazandu-se toata implementarea aplicatiei pe tehnocologii cu aceeasi baza, in cazul de fata, limbajul de programare Java.

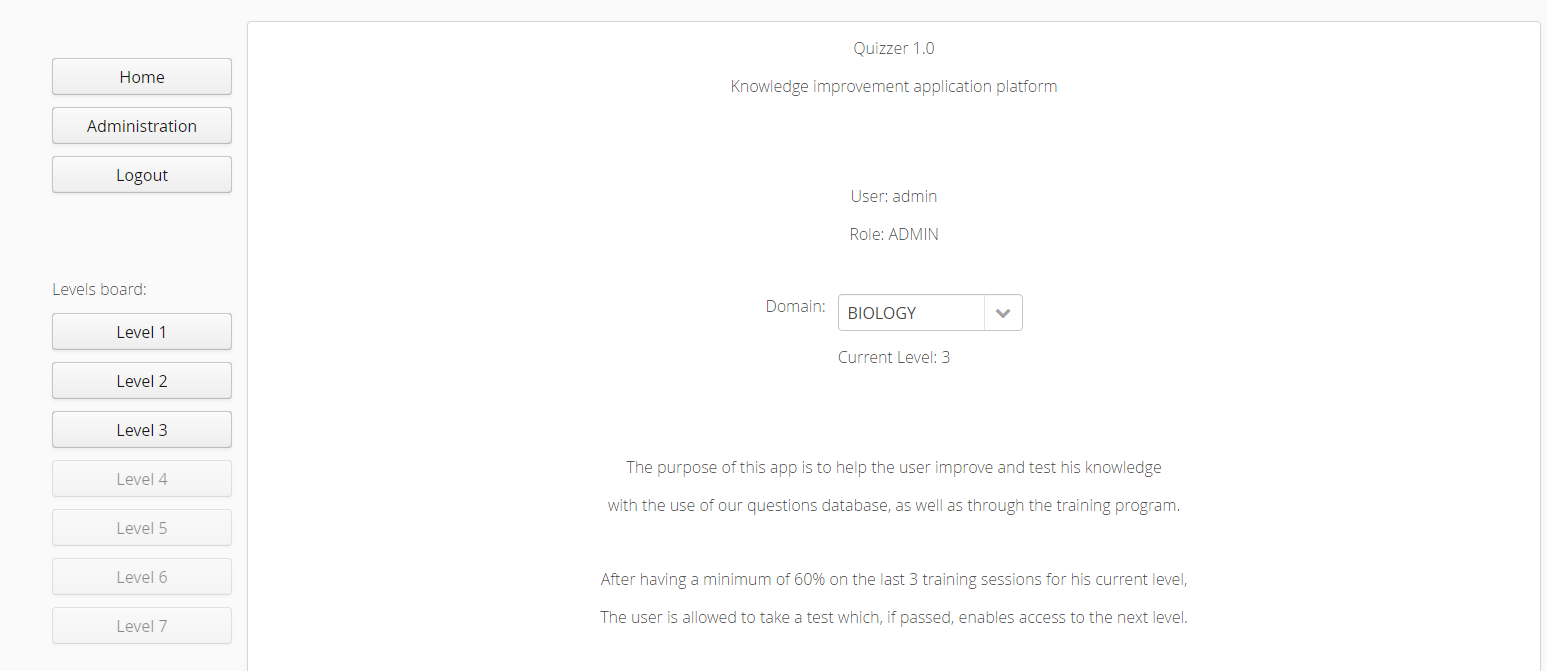
Fiind un framework realtiv similar cu Swing, totusi optimizat pentru web development, astfel ca acesta transforma codul Java in cod Javascript pentru a putea fi interpretata de browser, nemaifiind astfel nevoie de vrei un limbaj adițional pentru a putea exercita business logic-ul aplicației.

La baza interfeței de utilizator stă un element de tipul UI, astfel ca toate elementele interfeței de utilizator sunt coordonate de către acesta, așadar acesta acționează ca un middle tier între front-end și back-end, putând unei astfel exercita logica prin intermediul acestei componente.

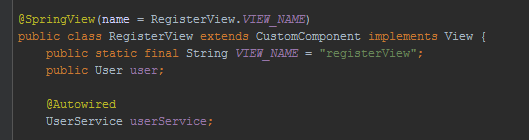


Aceasta componenta sta la baza arhitecturii fron-end-ului, intrucat ea este responsabila de routarea sa spunem asa view-urilor, de implementarea unui navigator, care face posibila mutarea intre diferite views ale aplicatiei.

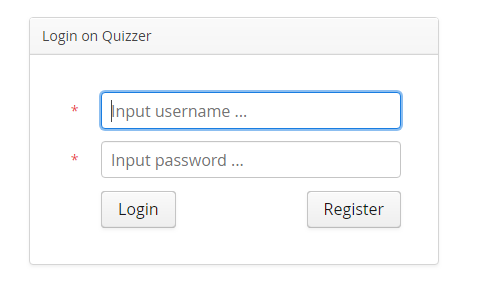
Structura de bază a interfeței de utilizator, odată ce acesta este logat pe platformă, este constituită din panel în zona din dreapta a paginii, Unde este conținutul efectiv al meniurilor și al informațiilor aferente platformei, iar în partea din stânga a paginii, în partea superioară, avem prezente butoanele  Administrative ale sesiunii, în speță butonul pentru home și butonul pentru delogare, tot aici fiind prezent și butonul pentru  deschiderea panelului de administrare, în cazul în care utilizatorul are rolul de administrator al platformei. în partea inferioară avem butoanele folosite pentru navigarea între diferitele nivele de dificultate ale aplicației,  Deci navigarea către sesiunile și examenele specifice fiecărui nivel de dificultate.



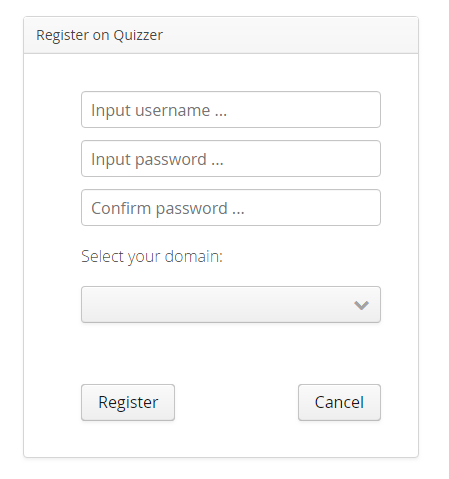
Implementarea paginilor consta de fapt in implementarea unor view-uri, avand la baza Vaadin, care sunt mai apoi introduse si ele in contextul Spring, ca si SpringViews, avand un nume distinct fiecare dintre ele. In baza acestui nume distinct se face mai apoi redirectionarea catre ele prin intermediul navigatorului, care este responsabil de modificarea contentului din panel-ul principal, din partea dreapta a interfetei, butoanele de navigare din partea stanga a interfetei ramand in aceeasi stare, totodata din ele aparand posibilitatea de trigger a evenimentelor de navigare.



Pentru prima fază a proiectării frontend-ului, eforturile s-au concentrat asupra creării interfeței pentru logare. Astfel, o dată cu generarea interfeței care conține un câmp pentru introducerea numelui de utilizator, un câmp pentru introducerea parolei, apoi butonul pentru logare și butonul pentru înregistrare, în momentul logării, se folosește un Binder, astfel că contul utilizatorului este legat de sesiunea aplicației pe care acesta tocmai a deschis-o. Astfel, se asigură o unitate a experienței pentru fiecare utilizator și totodată o măsură de siguranță pentru conectarea utilizatorului la secțiunea corespunzătoare.

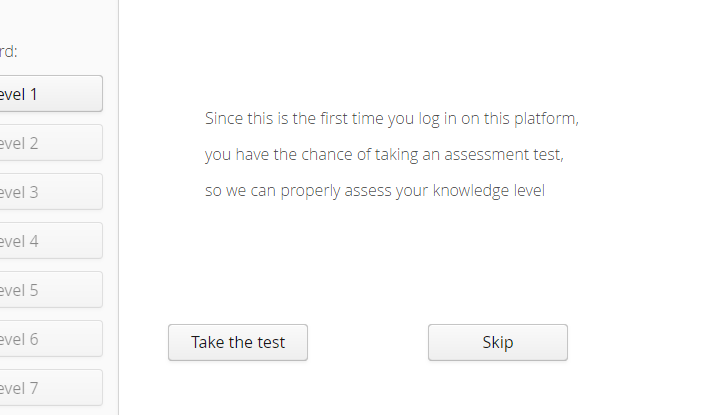


Mai apoi, odată ce aceasta a fost creată, următorul pas a fost reprezentat de crearea interfeței pentru înregistrarea utilizatorului nou în cazul în care utilizatorul contact cu platforma pentru prima oară. Astfel, interfața de utilizator a fost concepută ca având un câmp pentru introducerea numelui de utilizator, totodată verificarea acestuia pentru unicitate la nivel de bază de date, mai apoi câmpurile pentru introducerea parolei și totodată pentru confirmarea acesteia,urmând ca mai apoi următorul aliment să fie o listă derulantă cu domeniile disponibile, asa ca fiecare utilizator trebuie să își aleagă un domeniu de activitate în care acesta va fi testat.  mai apoi în zona inferioară a paginii sunt prezente butoanele pentru confirmarea înregistrării, și butonul pentru cancel, în cazul în care utilizatorul  nu vrea să salveze modificările.



După apăsarea butonului pentru înregistrare, aplicația face o interogare la baza de date pentru a verifica unicitatea a numelui de utilizator. In cazul în care acest criteriu este îndeplinit, aplicația verifică mai apoi dacă Câmpurile pentru introducerea parolei și respectiv verificarea acesteia coincid, urmând ca mai apoi, ultima verificare înaintea salvării înregistrări pentru utilizatorul respectiv  să fie selecția corectă a unui domeniu.

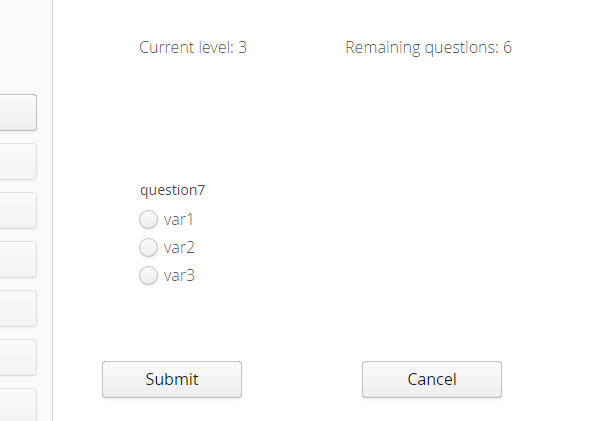
O dată înregistrat, utilizatorul se poate loga pe platformă. în cazul în care utilizatorul este pentru prima oară logat, acesta va fi informat că are posibilitatea de a susține un examen pentru o evaluare inițială, acesta având posibilitatea atât de a susține examenul, cât și de a nu face acest lucru, cazul în care acesta va începe automat cu nivelul 1 de dificultate.



Această evaluare inițială are ca scop principal stabilirea nivelului utilizatorului de dinainte ca acesta să fi folosit platforma putîndu-i-se  astfel atribui un nivel mai apropiat de cel deja existent.Motivul pentru care este recomandat o astfel de evaluare este acela de a evita intr-o oarecare măsură plictisirea utilizatorului sau forțarea acestuia în a parcurge nivele sub nivelul lui curent de dificultate. Astfel, Se asigură o experiență cât se poate de unitară asupra platformei, indiferent de gradul de cunoștințe pe care le posedă cursantul.

Algoritmul din spatele acestei testări inițiale constituie în primul rând plecarea de la nivelul 1 de dificultate ca și nivel inițial, atenția fiind îndreptată asupra răspunsurilor pe care le dă cursantul în urma întrebărilor primite.

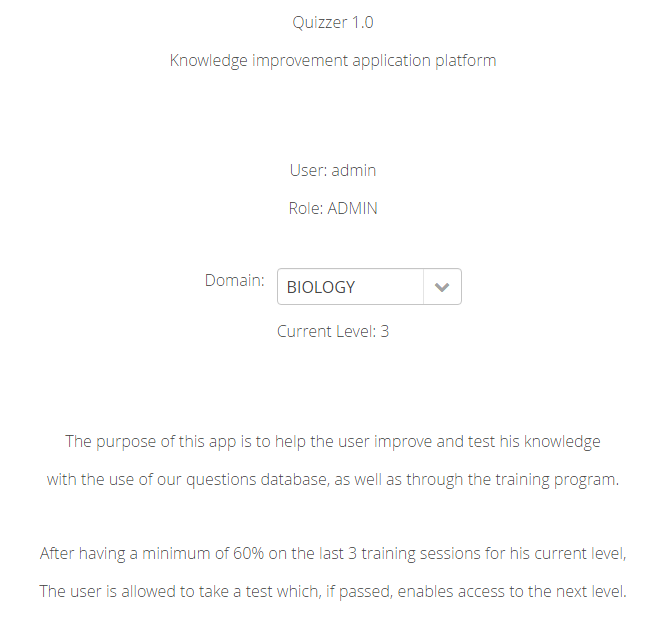
Acesta nu are vizibilitate, și nu poate răspunde decât la o singură întrebare pentru fiecare pas, astfel că următoarea întrebare pe care o va primi va fi cu un nivel superior sau inferior de dificultate față de cea curentă, în funcție de răspunsul acestuia. Cu alte cuvinte, dacă cursantul răspunde corect la întrebarea curentă, următoarea întrebare va fi cu un nivel de dificultate incrementat, iar în cazul în care cursantul răspunde cu un răspuns greșit, atunci următoarea întrebare va veni cu un nivel de dificultate decrementat.



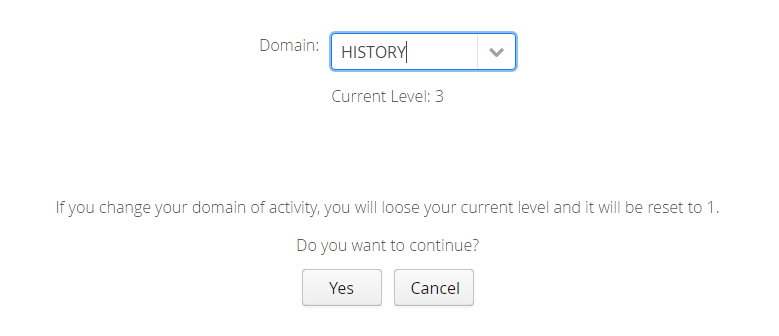
Pe toată durata testului, se ține minte nivelul curent al cursantului, în funcție de nivelul de dificultate al întrebării curente. Astfel, la finalul evaluării inițiale, platforma are un nivel mediu al răspunsurilor cursantului în funcție de rezultatele acestuia și de nivelul curent al ultimei întrebări, care este dependent de  de răspunsurile utilizatorului pe toată durata execuției testului, astfel că are locul evaluare cât mai completă a cunoștințelor cursantului și o evaluare a nivelului acestuia cât se poate de precisă putând fii categorisit într-unul din nivelele de dificultate ale platformei.

Odată cu finalizarea evaluării inițiale, utilizatorul are deblocate toate nivelele de dificultate inferioare nivelului curent. Astfel, Spre exemplu, dacă utilizatorul are evaluat nivelul de dificultate ca fiind corespunzător nivelului de dificultate 3 al platformei, atunci acestuia i se deblochează atât nivelul curent în cazul acesta 3, cât și nivelele inferioare, în speță nivelele 2 și 1. Folosind o astfel de logică, i se permite utilizatorului să poată susține examene și sesiuni de training și de la nivelele inferioare, chiar dacă acesta nu a avut contact cu ele.

O dată logat pe platformă, utilizatorului ii este prezentată pagina de start a aplicației, care conține informații generale despre platformă, despre contul utilizatorului și rolul acestuia, informații generale despre regulile de utilizare a acesteia, precum și o listă derulantă din care utilizatorul își poate schimba domeniul de activitate în care acesta dorește să fie testat.



în cazul în care utilizatorul dorește să schimbe domeniul întrebărilor, cursantul este avertizat că în momentul în care acesta dorește să schimbe domeniul, el își va pierde nivelul curent la care acesta a ajuns în momentul de față, și va începe evaluarea pe domeniul selectatîncepând de la nivelul de dificultate 1.



Odată navigat pe unul dintre nivelele de dificultate deblocate pentru utilizator, ne este prezentată interfața specifică nivelelor de dificultate care conține un notă informativă cu nivelul curent selectat, un tabel cu numele utilizatorului și rezultatele acestuia la ultimele trei sesiuni de antrenament, întrucât acestea sunt esențiale pentru a debloca oportunitatea de a susține examenul de promovare către nivelul următor.

Algoritmul aplicației presupune că un utilizator nu poate susține examenul pentru promovare la următorul nivel până când nu a îndeplinit un criteriu de bază, și anume, realizarea unui punctaj de peste 60% la ultimele trei sesiuni de training.Această metodă, acest mecanism de asigurare a consistenței cunoștințelor utilizatorului asigura o pregătire cât mai completă și mai complexă, în urma căreia prin confirmare, se poate susține examenul pentru promovare la următorul nivel de dificultate.

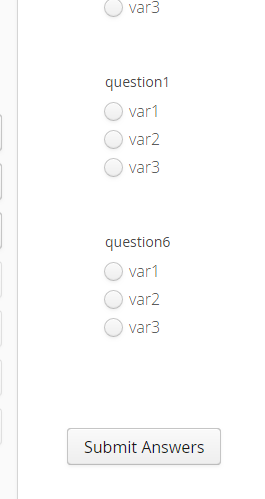


Sesiunile de training sunt una dintre bazele platformei pentru evaluarea de cunoștințe, întrucât utilizatorul pe trece o foarte mare parte din timpul alocat pe platformă, în aceste stadii.în urma execuției unei sesiuni de antrenament, rezultatul acesteia, indiferent de cuantum, este salvat în înregistrările bazei de date referitoare la utilizatorul respectiv, făcând dus felul lor în prospectare a statisticilor pe care acesta le vede la nivelul de interfață de utilizator.

Examenele pentru promovarea la următorul nivel de dificultate sunt parte esențială a platformei în sine, ele fiind de fapt modalitatea prin care cursantul poate verifica și cuantifica progresul pe care acesta l-a făcut până în momentul de față, și totodată este mijlocul principal de evaluare pe care această platformă de evaluare a cunoștințelor îl oferă.

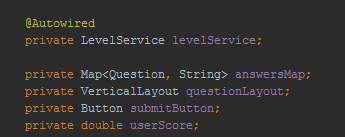
Aceste teste, spre deosebire de sesiunile de antrenament, conțin atât întrebări din nivelul de dificultate curent, cât și o serie de întrebări aleatoare specifice nivelelor de dificultate anterioare. Astfel, prin acest mecanism de asigurare a  persistenței cunoștințelor, se asigură o evoluție cât mai precisă, și o evaluare cât mai completă și corectă a cursantului, întrucât acesta este nevoit să dovedească că posedă un continuare cunoștințele de la nivelele inferioare, criteriu esențial în promovarea la următorul nivel de dificultate.

 Interfața de utilizator responsabilă pentru susținerea efectivă a examenelor a fost gândită să fie cât mai simplistă și cât mai minimalistă scopul unui astfel de design fiind acela de a permite utilizatorului un mediu cât mai lipsit de distrageri, acesta putându-se astfel concentrara cât se poate de eficient și de complet asupra întrebărilor din textul pe care acesta îl efectuează.

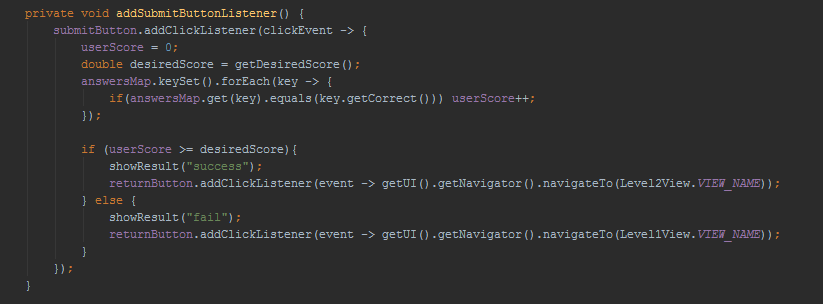


Algoritmul aplicației din spatele execuției testelor implică utilizarea serviciilor atât celui pentru utilizator cât și celui pentru nivele. Pe baza Serviciul de utilizator se fac legăturile cu sesiunea curentă și cu rezultatele din baza de date asupra utilizatorului, în speță nivelul curent, etc. Iar pe baza serviciului de nivele se face providing-ul cu întrebări, în funcție de nivelele necesare, mai precis nivelul curent și nivelele inferioare acestuia.

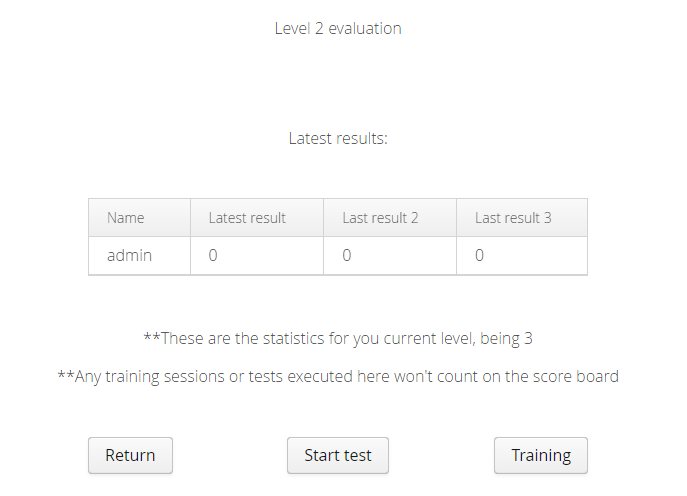
Totodată o altă parte foarte importantă algoritmice este reprezentată de mape, o structură de date de tipul cheie evaluare, care ne permiteți să mapăm întrebările și selecția utilizatorului ca fiind răspunsul corect. în baza acestei selecții, și a întrebării corespunzătoare acesteia, se face compararea între răspunsul utilizatorului și răspunsul știind că ei fiind cel corect pentru întrebarea respectivă, pentru validarea punctajului la întrebarea în cauză.



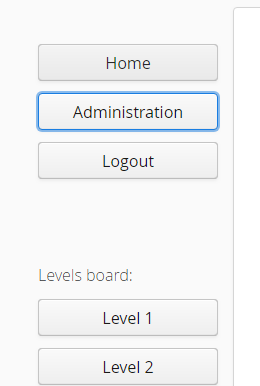
Validările au loc în momentul în care utilizatorul apasă pe butonul pentru trimiterea rezultatelor, implicit astfel confirmare a finalizării testului.Fiecare nivel are un așa a zis desired score, bazat pe numărul de întrebări pe care acesta îl conține, astfel ca punctajul necesar promovării în urma executiei unui test diferă de la nivel la nivel, în funcție de numărul de întrebări. Acest desired score este calculat în momentul afișării întrebărilor pentru testul respectiv, și este luat în calcul pentru Evaluarea rezultatului la final, după ce utilizatorul apasă pe butonul pentru trimiterea răspunsurilor.



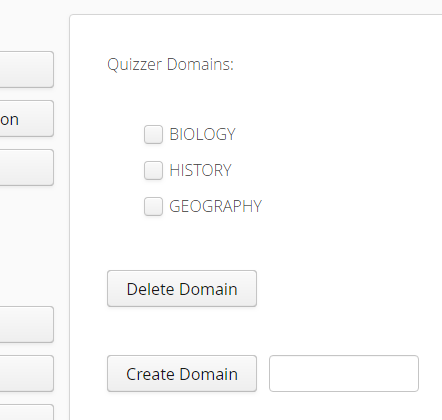
In cazul în care utilizatorul nu se află pe nivelul curent și este în panelul specificul unui alt nivel, inferior, acesta va fi atenționat că rezultatele în urma a execuției sesiunilor atât de antrenament cât și cele de promovare din nivelul respectiv nu se vor lua în calcul, și mai ales nu se vor lua în calcul pentru nivelul curent la care acesta activează. Aceste nivele sunt în momentul de față pentru cursant acolo doar cu caracter informativ.



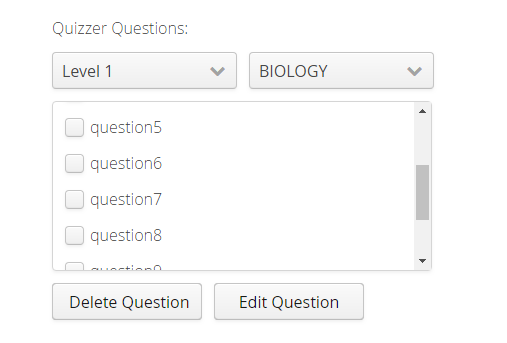
Pe lângă interfeței de utilizator specifice utilizatorilor standard, în speță cursanților, mai avem o interfață de utilizator specifică administratorilor, și anume panelul pentru administrare, prin care se poate ajunge printr un buton amplasat în zona butoanelor pentru home și pentru delogare, buton care nu este vizibil utilizatorilor standard ci doar administratorilor.



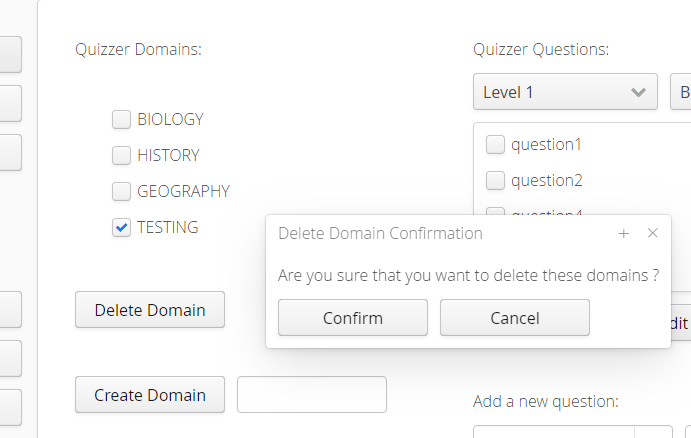
Acest panou administrativ conține trei zone importante, unde dintre ele fiind zona pentru administrarea domeniilor platformei, unde se pot vedea domeniile existente la momentul respectiv pe platformă, și totodată administratorii au abilitatea de a adăuga sau de a șterge domenii de pe platformă.



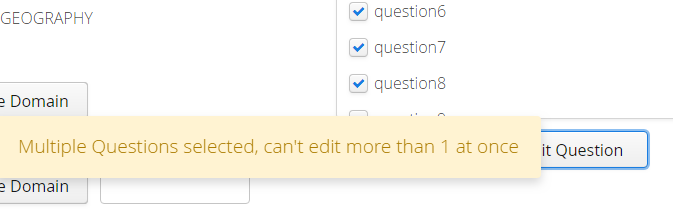
O altă zonă importantă a panelului administrativ este cea corespunzătoare administrării întrebărilor existente pe platformă. Acestea se pot vedea printr o modalitate de filtrare bazată pe nivelul acestora și pe domeniile corespunzătoare lor, cu posibilitatea de a fi șterse, respectiv editate.

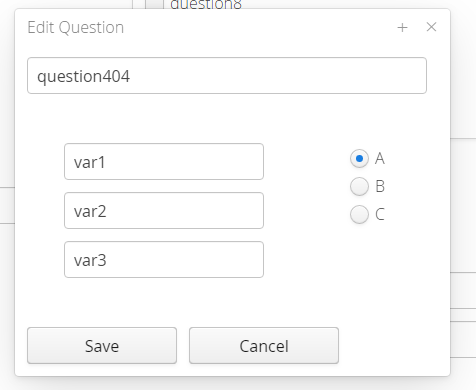


Atât în cazul ștergerii unui domeniu cât și a întrebărilor utilizatorul este informat să confirme acțiunea pe care urmează să o desfășoare, pentru a preveni ștergerea accidentală a datelor.

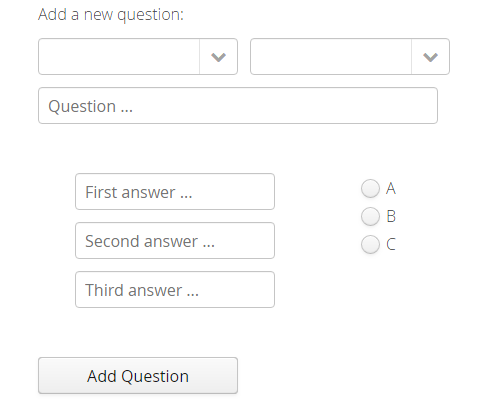


Pentru editarea întrebărilor administratorul nu poate selecta decât o singură întrebare. In cazul în care acesta a selectat multiple întrebări, sau nu a selectat nici una, fiind informat de utilizare a defectuoasă a editarii întrebărilor. După selectarea unei singure întrebări și apăsarea butonului pentru editare, administratorului i se va deschide o fereastră adițională, de unde acesta poate modifica oricare din câmpurile întrebării, în speță enunțul acesteia sau vreuna dintre cele trei variante de răspuns.





Cea de a treia zonă importantă din panelul administrativ este reprezentată de zona de unde administratorul poate adăuga noi întrebări pe platformă această zonă conține toate câmpurile specifice unei întrebări, enunțul acesteia, cele trei variante posibile de răspuns, precum și o zonă de selecție, de unde administratorul poate specifica care din cele trei răspunsuri este cel corect, și de asemenea administratorul este prezentat cu două liste derulante de unde acesta poate selecta domeniul, respectiv nivelul de dificultate ale întrebării nou create.



Prin panelul administrativ, administratorii sunt diferențiați de utilizatorii standard, in speță cursanții,  aceștia vând în plus abilitatea de a controla conținutul efectiv al platformei, putând adăuga, șterge atât domenii de activitate cât și întrebările aferente acestora.

**Concluzii**

Ca și concluzii, putem observa ca și lucru adus nou în acest domeniu, față de restul aplicațiilor, testarea inițială, având ca scop principal garanția unei experiențe similare pentru toți cursanții, indiferent de nivelul acestora de pregătire inițială, înainte de luarea contactului cu platforma.

Un alt lucru esențial pe care îl aduci această platformă este simplitatea și minimalismul arhitecturii interfeței grafice, astfel că utilizatorul se poate concentra eficient și complet asupra examenului pe care acesta îl susține în momentul respectiv, fără distrageri.

Un alt mecanism eficient al platformei pentru asigurarea consistenței și a persistenței cunoștințelor cursanților este reprezentat de sesiunile de antrenament, prin care utilizatorul trebuie să treacă înainte de a putea susține examenul pentru promovarea la următorul nivel de dificultate. Tot aici intră și cuprinsul examenului pentru promovare, și anume, prezența în conținut a întrebărilor de la nivelele inferioare nivelului curent, astfel asigurând-se o plajă de cunoștințe cât mai completă a cursanților.

O remarcă a abordării diferite față de majoritatea platformelor similare este constituită de concentrarea asupra testelor și evaluărilor susținute, și totodată de lipsa materialelor ajutătoare pentru pregătirea cursanților virgula întrucât trăim într-un mediu în care informația este accesibilă oricui, astfel nemaifiind necesară implicarea platformei în pregătirea utilizatorilor.

în concluzie, această platformă destinată evaluării cunoștințelor iese în evidență prin concentrarea asupra evaluărilor utilizatorilor a și totodată prin algoritmică folosită pentru a asigura are a nivelului de cunoștințe satisfăcător al cursanților.

**Bibliografie**

http://master.cedcsv.ro/cedc/infos/informatica/platforme\_de\_evaluare.pdf

http://spring.io/projects/spring-framework

https://spring.io/projects/spring-boot

https://docs.spring.io/spring-data/jpa/docs/current/reference/html/

https://www.journaldev.com/3531/spring-mvc-hibernate-mysql-integration-crud-example-tutorial

https://vaadin.com/docs

https://maven.apache.org/

http://blog.infoeducatie.ro/tutorial/2015/04/14/github-101.html

http://aipi2015.andreirosucojocaru.ro/laboratoare/laborator02

https://moqups.com/