

UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI

FACULTATEA DE INFORMATICĂ



LUCRARE DE LICENȚĂ

ReviewFacts

propusă de

Dorian Bazgan

Sesiunea: *iulie, 2018*

Coordonator științific

Lect, Dr. Arusoaie Andrei

UNIVERSITATEA “ALEXANDRU IOAN CUZA” DIN IAȘI
FACULTATEA DE INFORMATICĂ

ReviewFacts

Dorian Bazgan

Sesiunea: *iulie, 2018*

Coordonator științific

Lect, Dr. Arusoaie Andrei

Avizat,

Îndrumător Lucrare de Licență

Titlul, Numele și prenumele _____

Data _____ Semnătura _____

DECLARAȚIE privind originalitatea conținutului lucrării de licență

Subsemnatul(a)

domiciliul în

născut(ă) la data de, identificat prin CNP,
absolvent(a) al(a) Universității „Alexandru Ioan Cuza” din Iași, Facultatea de
..... specializarea, promoția
....., 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 și 5 referitoare la plagiat, că lucrarea de licență cu titlul:

_____elaborată sub îndrumarea dl. / d-na

_____, pe care urmează să o susțină în fața
comisiei este originală, 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 originalității, consimțind inclusiv la introducerea
conținutului său într-o bază de date în acest scop.

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 diploma 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 azi,

Semnătură student

DECLARAȚIE DE CONSIMȚĂMÂNT

Prin prezenta declar că sunt de acord ca Lucrarea de licență cu titlul „*Titlul complet al lucrării*”, codul sursă al programelor și celelalte conținuturi (grafice, multimedia, date de test etc.) care însoțesc 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.

Iași, *data*

Absolvent *Dorian Bazgan*

(semnătura în original)

Contents

1. Introducere	7
1.1. Motivație.....	7
1.2. Context.....	7
1.3. Cerințe funcționale	8
1.4. Contribuții	10
2. Structura lucrării	11
2.1. Client	11
2.2. Server	19
2.3. Modelarea datelor	20
3. Implementare	24
3.1. Baza de date.....	24
3.2. Implementarea aplicației	28
4. Experimente.....	37
5. Concluzii	41
6. Bibliografie:	42
7. Appendix:	43

1. Introducere

1.1. Motivație

Întâmpinând dificultăți în alegerea unui furnizor de servicii din diferite domenii, cum ar fi construcția unei locuințe, achiziționarea unor rulouri exterioare pentru ferestre, mobilarea unui dormitor, am simțit nevoia unei aplicații orientate spre aceste domenii care să ofere un feedback, o notă, sau un review firmelor care oferă astfel de servicii, pe baza experiențelor acumulate ale altor utilizatori care au beneficiat de aceste servicii.

Această lucrare de licență are ca scop dezvoltarea unei aplicații care să creeze o comunitate pentru utilizatorii care au nevoie sau pe care i-ar ajuta părerea altor utilizatori care deja au trecut printr-o experiență asemănătoare.

1.2. Context

În prezent, majoritatea oamenilor care au, spre exemplu, nevoie fie de a construi o locuință, sau de a-și mobila propria locuință, caută oameni specializați sau firme specializate din anumite domenii de activitate. Odată cu era internetului, majoritatea căutărilor au loc online, de aceea aplicația propusă are scopul de a-i ajuta pe utilizatori să găsească cea mai bună opțiune pentru necesitățile lor, în funcție de recenziile date de alți utilizatori care au beneficiat de serviciile firmei/întreprinzătorului respectiv.

La momentul actual, mai ales pe piața internațională, există o mulțime de aplicații care își propun să aibă același rol, acela de a forma o comunitate care, pe baza experiențelor anterioare, contribuie la a ajuta viitorul utilizator în a face o alegere cât mai corectă. Printre acestea, putem enumera TrustPilot, Consumer Reports sau Yelp.

Aplicația propusă în această lucrare de licență diferă față de aplicațiile de acest tip disponibile la ora actuală pe internet prin oferirea unei experiențe simplificate nu doar pentru utilizatorii care folosesc aplicația pentru a citi review-urile, dar și pentru cei care contribuie la procesul de evaluare a unei firme, cei care scriu review-uri pe baza experiențelor trecute.

Aplicația dispune de mecanism simplificat de adăugare a unui review, utilizatorul scrie direct review-ul pe pagina principală fără a naviga către categoria și firma dorită, iar pe baza unui algoritm de Machine Learning, review-ul este asociat automat unei categorii de pe site, iar utilizatorul trebuie doar să aleagă firma dintr-o listă proprie categoriei sugerate și să dea submit.

Astfel mecanismul de navigare pas cu pas către categoria și firma dorită pe baza meniului integrat în aplicație este simplificat, tot procesul de adăugare a review-ului realizându-se pe pagina principală a aplicației.

1.3. Cerințe funcționale

Utilizatorul poate accesa aplicația în mod direct, fără autentificare, pentru a putea vedea lista de firme și review-urile aferente acestora, însă nu va putea adăuga review-uri decât după crearea unui cont și autentificarea pe baza acestuia.

a. Înregistrarea/Logarea utilizatorilor

- Pentru a-și crea un cont nou, utilizatorul are nevoie de o adresă de email validă și o parolă. O adresă de email poate fi folosită pentru un singur cont, nu pot exista două conturi cu aceeași adresă de email.
- Parola trebuie să conțină atât caractere mici, caractere mari cât și cifre, iar lungimea trebuie să fie între 6 și 20 de caractere.

b. Adăugarea de review-uri:

- Dacă utilizatorul are ca scop folosirea aplicației pentru a se documenta și pentru a citi părerile altor utilizatori în vederea realizării unei alegeri, acesta poate filtra firmele/ întreprinzătorii în funcție de domeniul din care are nevoie prin intermediul unui "search box" pus la dispoziția sa pe pagina principală a aplicației.

- După înregistrare și, respectiv, autentificare, utilizatorul poate să adauge review, folosind cele două opțiuni oferite de aplicație:
 - Navigarea pas cu pas din meniul de pe pagina principală la categoria dorită, apoi la firma dorită, unde utilizatorul are la dispoziție un formular în care scrie review-ul și acordă un număr de stele;
 - Adăugarea review-ului direct de pe pagina principală, unde utilizatorul scrie review-ul, verifică categoria sugerată de clasificatorul automat să coincidă cu opțiunea sa, alege firma dintr-o listă sugerată de clasificator și acordă un număr de stele.
 - Dacă utilizatorul a ales să adauge review-ul folosind prima opțiune oferită de către aplicație, acesta poate opta ca review-ul sau să fie postat anonim, cu un alias pe care acesta îl poate alege în formularul de adăugare a review-ului. Astfel, numele și prenumele folosit la crearea contului nu va apărea în dreptul review-ului. În schimb, identitatea utilizatorului îi va fi disponibilă administratorului la momentul aprobării review-ului pe platformă.
 - Pentru opțiunea de adăugare cu ajutorului clasificatorului automat, review-ul poate fi adăugat doar cu numele și prenumele cu care utilizatorul și-a creat contul în aplicație.
- După ce review-ul a fost adăugat cu succes, va fi aprobat după verificarea unui administrator, după care va apărea pe pagina firmei respective, la secțiunea care cuprinde evaluările utilizatorilor.

c. Evaluarea review-urilor de către alți utilizatori:

- Pentru a îmbunătăți experiența utilizatorilor în aplicație și pentru a crește calitatea feedback-ului cu review-uri obiective, am adăugat o metodă de evaluare a review-urilor pentru ceilalți utilizatori, care dacă consideră review-ul util, sau după experiența avută cu o anumită firmă, pot confirma cele spuse de un alt utilizator, îi poate aprecia review-ul ca fiind util printr-un "like".

- Dacă un utilizator a apreciat review-ul altui utilizator înainte de a avea confirmarea celor spuse, poate reveni ulterior la review și dacă îl consideră neadevărat sau inadecvat pentru respectiva firmă, poate retrage "like-ul".

1.4. Contribuții

Aplicația ReviewFacts are ca scop crearea unei comunități care să se ajute reciproc pe baza impresiilor și experiențelor acumulate privind un anumit furnizor de servicii. Utilizatorii care vor să facă o alegere, consultă părerile exprimate de alți utilizatori care au făcut respectivă alegere într-un moment trecut, și astfel, pot găsi ce este mai bun pentru ei și cerințele lor.

M-am gândit la o aplicație care să înglobeze informații de la mai mulți utilizatori accesibile atât persoanelor din exterior, care vor doar să se documenteze fără să contribuie la adăugarea de review-uri, cât și pe baza unui cont personal, cu care utilizatorii pot adăuga review-uri sau pot vota review-urile existente ale altor utilizatori pe baza unui like.

Am decis să împart lucrarea în două capitole mari, care să cuprindă procesul de dezvoltare. Primul capitol descrie modul în care este structurată aplicația, componentele din care este alcătuită, rolul lor și modul de interacțiune între ele.

Al doilea capitol prezintă modul în care aplicația a fost implementată pentru a îndeplini scopul propus.

Contribuțiile aduse în procesul de dezvoltare sunt:

- Implementarea logicii din spatele aplicației
- Dezvoltarea unei interfețe grafice pentru aplicație;

2. Structura lucrării

Structura generală a aplicației este cea a unei aplicații web obișnuite, cuprinzând trei părți principale ca și componente: clientul, serverul și baza de date.

Mecanismul de funcționare este următorul: clientul este cel care inițiază o cerere către server, care o procesează. Dacă este cazul, serverul trimite o cerere către bază de date pentru a primi date necesare pentru a construi răspunsul cererii clientului, după care este trimis către partea de client răspunsul, unde vor apărea și modificările realizate.

Următoarea imagine prezintă structura generală a aplicației:



Figura 1: Structura generala a aplicatiei

2.1. Client

În acest subcapitol voi descrie arhitectura clientului. Clientul reprezintă legătura dintre utilizator și logica aplicației, prin intermediul sau utilizatorul poate folosi funcționalitățile aplicației, din acest motiv clientul trebuie să fie intuitiv și totodată receptiv la cererile utilizatorului.

În această aplicație, clientul este format dintr-o serie de pagini web, fiecare punând la dispoziție funcționalitățile aplicației către utilizator. Principalele componente care constituie clientul, sunt:

- **Pagina principală:** este prima pagină cu care utilizatorul interacționează atunci când intră pentru prima oară în aplicație. Pagina principală este formată din mai multe componente, ea oferind și o funcționalitate importantă a aplicației.
 - Prima componentă, meniul de navigare general, care apare în toată aplicația, nu doar pe pagina principală, cuprinde:
 - logo-ul aplicației;
 - butonul "Pagina principală" odată apăsă, trimite utilizatorul pe prima pagină a aplicației indiferent de locul în care se află;
 - butonul "Despre" are rolul de a-i oferi utilizatorului o scurtă descriere despre aplicație;
 - un dropdown "Meniu" cu opțiunile de Înregistrare și Logare;
 - Cea de-a doua componentă, reprezintă meniul specific aplicației, care este constituit din mai multe submeniuri de tip dropdown, fiecare reprezentând categorii din anumite domenii de activitate, iar din dropdown utilizatorul poate alege să vadă lista de firme pentru acea categorie, sau toate review-urile din categoria respectivă.
 - Sub acest meniu, utilizatorul poate găsi un searchbox care îl poate ajuta să găsească o listă de firme pe bază un cuvânt cheie, care să reprezinte domeniul de activitate al firmei respective, spre exemplu, o listă de firme care realizează mobilier pentru dormitor.

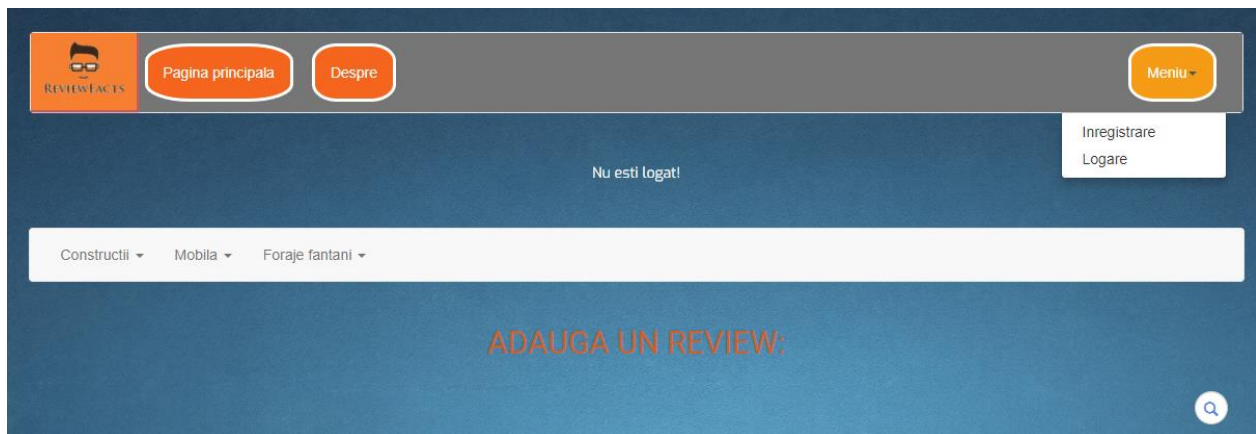


Figura 2: Meniurile de navigare de pe pagina principală

- Ultima componentă de pe pagina principală, și cea mai importantă din aplicație, din punct de vedere funcțional, reprezintă componenta de adăugare a recenziilor cu ajutorul clasificatorului automat, Bayes Naiv. Această componentă este formată dintr-o "text area" în care utilizatorul poate să scrie partea textuală care constituie recenziei:
- După adăugarea comentariului, utilizatorul are la dispoziție un buton de verificare a categoriei care va apela algoritmul de clasificare automată și va afișa pe ecran utilizatorului categoria sugerată de care aparține review-ul sau;

Dacă utilizatorul consideră categoria sugerată de algoritm ca fiind opțiunea dorită, atunci el va interacționa cu ultimele trei subcomponente:

- Odată afișată categoria pe ecran, se încarcă lista de firme care fac parte din categoria respectivă într-un dropdown, de unde utilizatorul alege pe baza numelui firma la care dorește să adauge review-ul;

- Pentru completitudinea recenziei, utilizatorul trebuie să aleagă un număr de stele, între 1 și 5, lucru pe care îl poate realiza selectând unul din cele 5 radio buttons puse la dispoziție în interfață.
- După parcurgerea acestor pași, cel din urmă este trimiterea review-ului către aprobare, acest lucru realizându-se prin intermediul ultimei subcomponente, butonul de "Trimitere".

Figura 3: Adaugarea review-ului folosind clasificatorul Bayes

- **Pagina de înregistrare:** poate fi accesată de pe prima pagină, din meniul general al aplicației și conține un formular care este constituit din patru câmpuri de intrare, pentru nume, prenume, email, parola, label-uri pentru validarea câmpurilor, un label pentru confirmarea înregistrării și un buton pentru finalizarea înregistrării.

The image displays two web forms side-by-side. The left form, titled '-Inregistrare-', is for user registration and includes input fields for 'Nume:' (Name), 'Prenume:' (Surname), 'Email:', and 'Parola:' (Password). Each field has a small icon on the right side of the input box. Below these fields is a blue 'Submit' button. The right form, titled '-Autentificare-', is for user login and includes input fields for a username (with a person icon) and a password (with a lock icon). Below these fields is a blue 'Log In' button. Both forms are enclosed in a light green border with a blue outer frame.

Figura 4: Formularele de inregistrare si autentificare

- **Pagina de logare:** de asemenea accesibilă de pe prima pagină din același meniu unde se găsește și pagina de înregistrare, și conține un formular alcătuit din 2 câmpuri de intrare, un label pentru mesaje de eroare sau succes și un buton pentru terminarea procesului de logare.
- **Pagina de listare a firmelor:** reprezintă pagina în care se încarcă toate firmele specifice unei anumite categorii pe care o alege utilizatorul din cel de-al doilea meniu de navigație de pe pagina principală. Pe această pagină utilizatorul poate vedea pentru fiecare firmă numele său, un scor(rank) care reprezintă o medie a numărului de stele din feedback-ul utilizatorilor adăugat până la momentul respectiv, și două tag-uri, "Detalii firma" și "Vezi review-uri".



FIRME		
Firme	Detalii	Review-uri
Firma1 	Detalii firma ➤	Vezi review-uri ➤
Firma2 	Detalii firma ➤	Vezi review-uri ➤

Figura 5: Lista firmelor pentru o anumita categorie

În funcție de opțiunea pe care o alege, utilizatorul este trimis pe:

- **Pagina cu detaliile firmei**, care este alcătuită dintr-un tabel care conține numele firmei, sediul firmei împreună cu orașul în care această firmă își desfășoară activitatea, și o scurtă descriere despre firma sau despre serviciile pe care aceasta le prestează.

DETALII FIRME	
Numele firmei:	Firma2
Sediu/Locatie:	Iasi
Descriere: <div>Firma cu sediul in Iasi</div>	

Figura 6: Pagina cu detalii pentru o anumita firma

- **Pagina cu review-urile firmei**, este locul unde utilizatorul vizualizează toate recenziile adăugate de alți utilizatori, care constă în numele utilizatorului care adaugă review-ul, comentariul propriu-zis în care acesta și-a exprimat opinia în legătură cu firma și cu serviciile oferite de aceasta, și un scor, o notă, reprezentată printr-un număr de stele aurii care

materializează impresiile și părerea generală pe care și-a format-o utilizatorul după experiența avută.

Ultima subcomponentă a feedback-ului este butonul de like, destinat celorlalți utilizatori ai aplicației care citesc review-urile și își exprimă aprecierea față de review-urile adăugate. Un utilizator are dreptul de a aprecia o singură dată un anumit review, iar dacă după o anumită perioadă de timp, din anumite motive, consideră că review-ul nu a reflectat realitate, poate să își retragă like-ul prin apăsarea aceluiași buton. În dreapta butonului de like este afișat numărul like-urile acumulate de un anumit review.

Tot pe această pagină este locul unde utilizatorii își pot construi propriul review pentru o anumită firmă, lucru posibil prin apăsarea butonului de "Adăugare review". Un alt buton existent pe această pagină este cel de "Admin" care trimite utilizatorii cu drepturi de administrator în aplicație pe o pagină unde le este permise următoarele acțiuni :

- Ștergerea unui utilizator existent în sistemul aplicației;
- Ștergerea unui review existent în aplicație;
- Aprobarea review-urilor ce tocmai au fost adăugate.

Un utilizator are dreptul de a adăuga un singur review pentru o anumită firmă, iar după adăugarea în aplicație, butonul "Adaugă review" va fi dezactivat pentru utilizatorul respectiv, iar la poziționarea cursorului pe butonul dezactivat, utilizatorului îi va apărea un mesaj de informare sub forma unui tooltip, prin care va fi înștiințat că a adăugat deja un review în sistem.

REVIEW-URI

Nume utilizator:	Bazgan Dorian
Comentariu:	<div>Primul meu review</div> <div>👍 0</div>
Scor:	★☆☆☆☆

Adauga Review

Admin

Ai adaugat deja un review!

Figura 7: Lista cu review-uri adăugate de utilizatori

Am ilustrat în imaginea următoare, folosind o diagramă use case, interacțiunea utilizatorului cu partea de client din aplicație:

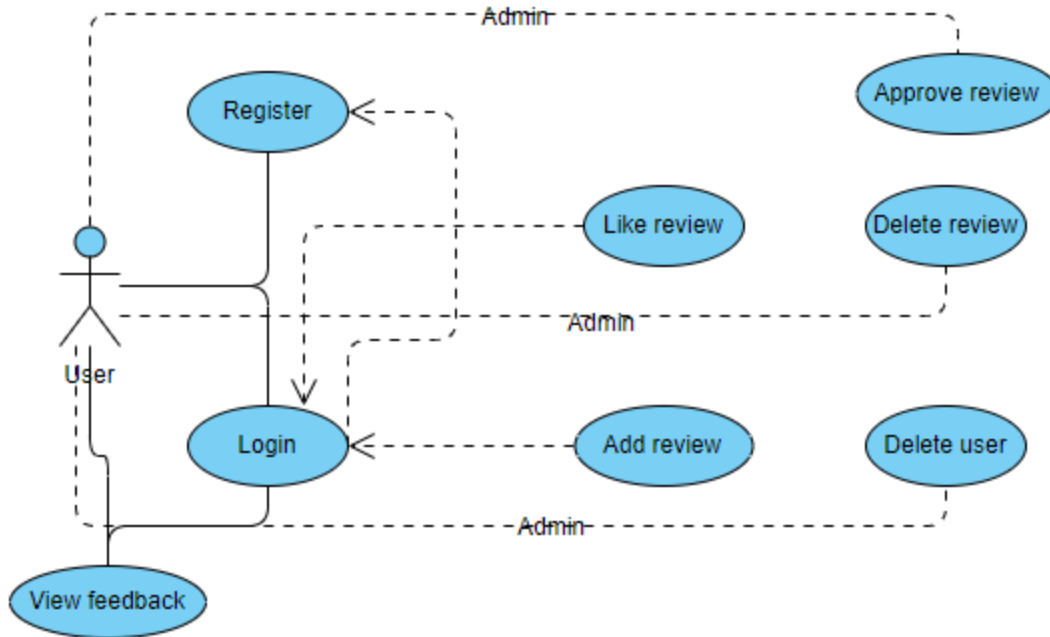


Figura 8: Diagrama use-case a aplicației

2.2. Server

Partea de server a aplicației ReviewFacts este construită în limbajul de programare C#, și este partea de aplicație care are rolul de a asculta și rezolva orice cerere din partea clientului. Când un request ajunge de la client la server, o succesiune de operații au loc înainte de trimiterea răspunsului către client. Fiind o aplicație construită folosind framework-ul de web development ASP .NET Web Forms, cererile clientului sunt procesate pe partea de client de către serverul IIS.

Aplicația este hostată pe serverul local furnizat de mediul de dezvoltare, IIS(Internet Information Services), care are propriul său Process Engine ASP .NET pentru a rezolva cererile ASP .NET. Atunci când un request ajunge de la client la server, IIS preia acel request, îl procesează și trimite răspunsul înapoi la client.

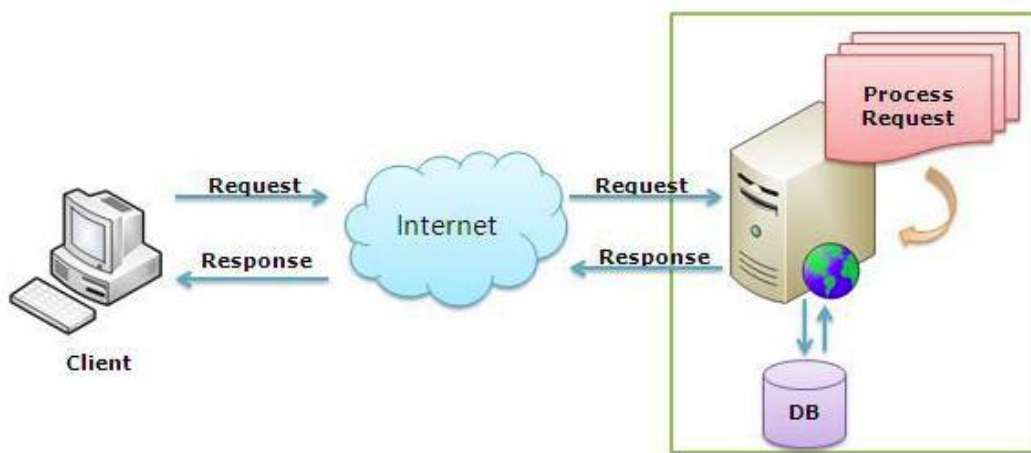


Figura 9: Arhitectura client server adoptată de aplicație

Fiind un proiect ASP .NET Web Forms, partea de client se află în fișierele cu extensia .aspx, iar cele în care este scris codul sursă al serverului au extensia aspx.cs.

Pe partea de server se permite accesul utilizatorilor în aplicație pe baza a două roluri, cel de User și de Admin. Astfel în funcție de rolul pe care îl deține, utilizatorul va avea acces sau nu la anumite părți ale aplicației. Pe baza acestui rol, utilizatorului i se asignează un grad de încredere în a gestiona aplicația.

2.3. Modelarea datelor

Pentru a realiza operațiunile specifice acestei aplicații, serverul comunică cu o bază de date relațională. Aici sunt salvate date despre useri, firme și review-uri. Bază de date construită pentru această aplicație are următoarea structură:

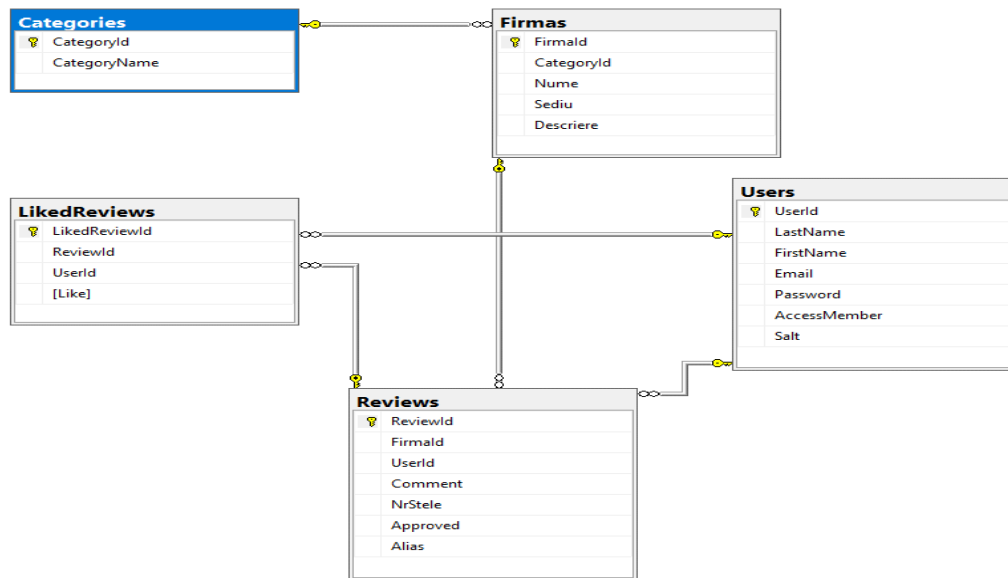


Figura 10: Arhitectura bazei de date

Entitati:

- **USERS:** conține informațiile legate de un utilizator. Date stocate în această tabelă sunt completate în momentul în care un utilizator își creează un cont în aplicație, prin intermediul paginii de înregistrare. Câmpul **UserId** este de tip **primary-key**, eliminând astfel posibilitatea de a avea doi utilizatori cu același **Id**.
- **CATEGORIES:** conține toate categoriile la care sunt asigurate firmele din aplicație, câmpurile memorate fiind **id-ul categoriei** și **numele**.
- **FIRMAS:** conține informații despre firmele care au, sau vor avea feedback în aplicație. În acest tabel avem ca și cheie primară câmpul **FirmId**, iar ca și cheie străină câmpul **CategoryId** din tabela **CATEGORIES**, pentru a asigura firma la o anumită categorie.
- **REVIEWS:** această tabelă stochează toate review-urile adăugate de utilizatori în aplicație, împreună cu **id-ul firmei** de care aparține review-ul, câmp de tip cheie străină și **id-ul user-ului** care a adăugat review-ul, de asemenea, câmp de tip cheie străină în tabela **Reviews**.

Pe lângă aceste două informații, se stochează toate componentele recenziei propriu-zise: comentariul, numărul de stele, statusul review-ului (câmpul Approved) la un anumit moment, care atunci când va fi adăugat, va avea valoarea setată pe 0, iar după intervenția unui administrator, acest câmp își poate schimba valoarea în Approved = 1, când review-ul este aprobat să fie afișat în aplicație. Este stocat și alias-ul utilizatorului care a adăugat review-ul, dacă acest a optat pentru opțiunea de a adăuga review-ul sub anonim.

- **LIKEDREVIEWS:** este tabela care stochează informații despre utilitatea unui review, stocând like-urile utilizatorilor pentru un anumit review. Conține câmpul ReviewId și UserId ca și chei străine pentru a se reține informații despre review-ul care este apreciat și despre user-ul care a realizat aprecierea.

Relații între entități:

Relațiile între entitățile din baza de date proiectată pentru această aplicație pot fi de tip 1:M. Astfel, între entitățile prezentate mai sus, există următoarele relații:

- **Firmas – Reviews:** această relație este de tip one-to-many, astfel că unei singure firme îi pot corespunde mai multe review-uri, însă un review este asignat unei singure firme;
- **Categories – Firmas:** relație de tip one-to-many, reprezintă faptul că unei singure categorii, îi pot corespunde mai multe firme, însă o firmă poate face parte dintr-o singură categorie;
- **Users – Reviews:** este o relație de tip one-to-many, un user având posibilitatea să adauge mai multe review-uri, însă un review aparține unui singur user;
- **Users – LikedReviews:** relație de tip one-to-many, reprezintă faptul că un user poate da like-uri la mai multe review-uri, însă un review poate fi apreciat de același utilizator o singură dată;

- **Review – LikedReview:** relație de tip one-to-many, în sensul că un review poate fi apreciat de mai multe ori(poate avea mai multe like-uri), în timp ce o entitate de tip LikedReview, care reprezintă un review apreciat, se poate identifica cu o singură entitate de tip Review.

3. Implementare

3.1. Baza de date

În realizarea bazei de date, am folosit SQL Server 2017 în combinație cu Entity Framework Code First, baza de date este alcătuită din 5 entități, descrise în detaliu la capitolul Modelarea datelor.

Pentru crearea acestei baze de date, am folosit Entity Framework Code First Approach, care mi-a permis să structurez baza de date sub formă de clase în C#, după care am migrat baza de date în SQL Server.

Pentru a detalia modul de gândire în realizarea acesteia, voi atașa câteva imagini cu structura obiectelor:

- **Firma:**

Clasa Firmă, care se va materializa în entitatea Firms după migrare, conține următoarele câmpuri(le voi aduce în discuție pe cele care au o anumită legătură cu restul bazei de date):

- FirmaId: este cheia primară din tabelul Firms;
- CategoryId: este cheia străină din tabelul Firms care pointează către cheia primară din tabela Categories;
- Categorie: este o proprietate virtuală care de tipul clasei Category ,are un rol bine definit în modelarea relației one-to-many cu tabela Firms, care semnifică faptul că o firmă aparține unei singure categorii;
- ReviewurileFirmei: reprezintă o colecție virtuală de obiecte de tip Review, cu rolul de a modela relația one-to-many dintre tabela Firms și Review, și reprezintă faptul că o firmă poate avea o colecție de review-uri.

- **Category:**

Clasa Category reprezintă modelul pentru tabela Categories, și conține:

- câmpul CategoryId ca și cheie primară;
- colecția virtuală de obiecte de tip Firmă, ListOfFirmas, care are rolul de a stoca lista de firme specifice categoriei respective;

- **User:**

Clasa User reprezintă modelul pentru tabela Users, cu rolul de a stoca toate datele legate de un utilizator al aplicației și conține următoarele câmpuri relevante:

- UserId: cheia primară în viitorul tabel cu rol de a identifica în mod unic un utilizator;
- ReviewsPerUser: colecție virtuală cu rol în modelarea relației one-to-many cu tabela Reviews, evidențiază faptul că un User poate adăuga mai multe review-uri;
- LikesPerUser: colecție virtuală cu rol în modelarea relației one-to-many cu tabela LikedReviews, reprezintă faptul că un user poate da like la mai multe like-uri;

- **Review:**

Clasa Review conține toate informațiile care vor fi stocate în tabela Reviews, iar ca și câmpuri relevante amintim:

- ReviewId: câmp de tip cheie primară, cu rol de identificator unic al unui review;
- FirmaId: câmp de tip cheie străină în tabela Reviews, care pointează către tabela Firmas;
- UserId: câmp de tip cheie străină în tabela Reviews, care pointează către tabela Users;

- Approved: reprezintă o valoare numerică cu semnificație booleană, atunci când un review este aprobat, acest câmp își schimbă valoarea din 0 în 1.
- Firma și User, două proprietăți virtuale de tipul claselor Firma și User, cu rol în modelarea relațiilor one-to-many dintre tabela Reviews și tabelele Firms și Users, semnifică faptul că un review este asociat unei singure firme și scris de un singur utilizator;
- LikesForReview, este o colecție virtuală, pentru modelarea relației dintre entitățile Review și LikedReview, stochează like-urile pentru un review.

- **LikedReview:**

Clasa LikedReview a fost contruită pentru tabela care va conține like-urile primite de un review, valorificându-l în viziunea utilizatorilor care îl citesc. Ca și câmpuri importante, conține:

- LikedReviewId: camp de tip cheie primară, cu rol de a identifica în mod unic un review apreciat;
- ReviewId: camp de tip cheie străină, care pointează către tabelul Reviews;
- UserId: câmp de tip cheie străină, care pointează către tabelul Users;
- ReviewForLike și UserForLike, două proprietăți virtuale care sunt folosite de Entity Framework la modelarea relației 1:M dintre tabela LikedReviews și cele două tabele Review și User;

Pe baza acestor câmpuri și a proprietăților virtuale, după migrare, bază de date va conține exact toate legăturile dorite la proiectare.

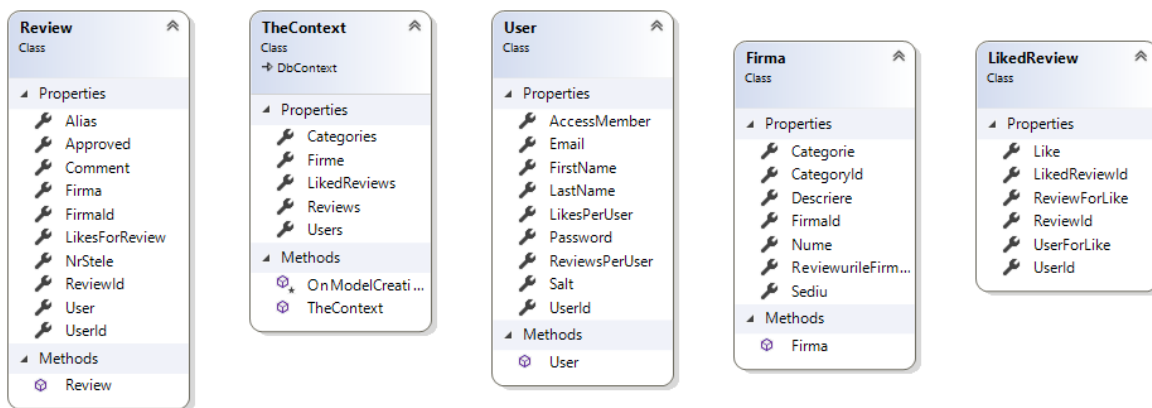


Figura 11: Structura claselor care se vor concretiza in tabele dupa migrare

Pentru a marca un câmp ca fiind cheie primară în viitorul tabel, numele proprietății în clasa creată în cod, trebuie să respecte o convenție de numire, și anume numele proprietății să conțină numele clasei urmate de "Id". Astfel, Entity Framework-ul va seta toate câmpurile care respectă această convenție ca și cheie primară în tabelele pe care le va crea la migrare.

Pentru a seta câmpurile de tip cheie străină, se respectă o convenție de numire asemănătoare, astfel în tabelul în care câmpul se dorește a fi de tip cheie străină, acel câmp trebuie să aibă același nume cu cheia primară din tabelul către care pointează. Putem exemplifica prin următoarea imagine:

```
public class Firma
{
    6 references | Dorian, 301 days ago | 1 author, 1 change | 0 exceptions
    public int FirmaId { get; set; }
    0 references | CODE40\dorian.bazgan, 51 days ago | 1 author, 1 change | 0 exceptions
    public int? CategoryId { get; set; }
}

public class Category
{
    2 references | CODE40\dorian.bazgan, 51 days ago | 1 author, 1 change | 0 exceptions
    public int CategoryId { get; set; }
}
```

Figura 12: Ilustrarea modelarii cheilor

Câmpul `CategoryId` este cheie primară în tabela `Categories`, iar în tabela `firma` este cheie străină.

3.2. Implementarea aplicației

În următorul capitol voi analiza aplicația din punct de vedere tehnic, felul cum a fost abordată fiecare componentă în procesul de dezvoltare.

- Autentificarea și înregistrarea:

În pagina de înregistrare, utilizatorul are un formular unde va completa câmpurile necesare, care vor fi trimise către server după apăsarea butonului de Înregistrare.

Înainte de a fi trimise către server, are loc la nivel de client validarea câmpurilor introduse, astfel utilizatorul nu poate să omită completarea unor câmpuri, fiindu-i afișat un mesaj de atenționare în caz contrar.

Acest lucru este implementat cu ajutorul controlului `RequiredFieldValidator` din `ASP.NET` care este legat către controlul care are nevoie de validare, în cazul de față, `textbox`-urile în care se completează datele.

Email-ul este validat o dată prin `RequiredFieldValidator` și încă o dată cu ajutorul unui validator pe bază de expresii regulate, care verifică dacă email-ul introdus are o anumită structură specifică, să fie urmat de "`@provider.com`".

De asemenea este realizată o validare a parolei pe baza căreia utilizatorul dorește să-și creeze un cont în aplicație, care trebuie să conțină o combinație de caractere alfanumerice și simboluri și să aibă o lungime cuprinsă între 6 și 20 de caractere.

Validarea parolei este implementată pe server, cu ajutorul clasei `PasswordAdvișor`, care conține două metode de verificare și care returnează `true` sau `false`, în funcție de condițiile îndeplinite sau nu de parolă aleasă de utilizator.

Dacă parola nu trece de o anumită condiție, este afișat în labelul de informații din formular-ul de înregistrare mesajul specific validării eșuate.

Implementarea validărilor pentru email și pentru parola se pot vizualiza în figurile 13-17 din Appendix.

The image contains two screenshots of a web registration form. The left screenshot shows the full form titled '-Inregistrare-'. It has four input fields: 'Nume:' (Name) with the value 'Bazgan', 'Prenume:' (Surname) which is empty and has a message 'Prenumele trebuie completat!' (Surname must be completed!), 'Email:' with the value 'dorian12995@gmail.com', and 'Parola:' (Password) which is masked with dots. A blue 'Submit' button is at the bottom. The right screenshot is a close-up of the 'Parola:' field. It shows the same masked input field and a validation message below it: 'Parola trebuie sa contina o combinatie de caractere mari, mici si cifre!' (Password must contain a combination of uppercase, lowercase, and digits). A blue 'Submit' button is also visible below the message.

Figura 13: Mesajele de validare afisate in formularul de inregistrare

Dacă toate informațiile introduse au trecut de validatori, atunci utilizatorul este salvat în baza de date.

Pentru a salva în baza de date parola utilizatorului, am realizat un mecanism de criptare a parolei, care constă în următorii pași:

- Parola este preluată din textbox-ul formularului de înregistrare și este convertită într-un
- byte array;
- Se generează un număr criptografic random numit salt, de asemenea convertit în byte array;
- Peste parola în bytes se adaugă salt-ul în bytes;
- Byte array-ul returnat este criptat folosind algoritmul de criptare SHA256:

- Parola criptată este salvată în baza de date sub formă de plaintext;

La partea de autentificare, utilizatorul are la dispoziție un formular cu două textbox-uri, unde acesta completează email-ul și parola cu care și-a creat contul la pasul de înregistrare.

La completarea email-ului și a parolei și după apăsarea butonului de logare, se verifică în baza de date dacă există un utilizator înregistrat cu email-ul introdus, dacă da, atunci se parcurg pașii de criptare a parolei introduse, iar dacă criptarea parolei corespunde cu cea din baza de date, utilizatorul este autentificat.

- Adăugarea review-urilor:

După înregistrare și autentificare, utilizatorul poate să adauge la rândul său review-uri în aplicație, prin cele două metode descrise la capitolul Structura lucrării.

Voi detalia cele două metode, din punct de vedere al implementării, în cele ce urmează:

Dacă utilizatorul optează pentru varianta de adăugare a review-ului direct de pe pagina principală, acesta va folosi clasificarea cu ajutorul algoritmului Bayes Naiv. În momentul în care utilizatorul a terminat de scris review-ul în textarea de pe pagina principală, următorul pas este de a verifica la ce categorie sugerează clasificatorul că ar trebui adăugat review-ul.

Acest lucru se realizează la apăsarea butonului "Sugerează o categorie", moment în care se preia review-ul de pe client din textarea și se realizează un call ajax de tip POST către metoda de pe backend care apelează clasificatorul.

Astfel review-ul este primit ca parametru de metodă de pe server ReviewValue; în această metodă este apelat clasificatorul Bayes după cum urmează:

Metoda ReviewValue trece prin toți pașii necesari apelării algoritmului Bayes Naiv, prin urmare voi prezenta fiecare linie din această metodă, care are legătură cu clasificatorul.

- Se verifică dacă review-ul introdus de către utilizator conține vreun cuvânt cheie din datele de antrenament, lucru implementat în metoda HasKeyword, care ia ca parametri review-ul și datele de antrenament pentru clasificarea review-ului. (Figura 23)
- Dacă review-ul conține cuvinte specifice unei anumite categorii, atunci putem merge mai departe la clasificarea apartenenței review-ului unei categorii.
- Pentru a stoca datele de antrenament, am construit o clasă Document, care stochează în două proprietăți clasa și propoziția pentru clasificare. (Figura 24)
- Folosind o listă de obiecte de tip Document, am creat datele de antrenament pentru clasificarea review-urilor, cât și pentru clasificarea firmelor în funcție de domeniul de activitate. (Figura 25)

Pentru a putea aplica algoritmul de clasificare Bayes Naiv, avem nevoie de o prelucrare a review-ului, prelucrare care a fost implementată după cum urmează:

- Pentru a clasifica review-ul ca aparținând unei anumite clase, principiul algoritmului constă:
 - extragerea tuturor cuvintelor din datele de antrenament;
 - extragerea tuturor cuvintelor din review-ul introdus de utilizator;
 - verificarea apartenenței fiecărui cuvânt la lista de cuvinte pentru fiecare clasă în parte;
 - dacă un cuvânt este găsit în lista de cuvinte pentru o clasă anume, incrementăm o variabilă cu rol de probabilitate; (Figura 26)
 - pentru că în lista de cuvinte pentru o anumită clasă pot exista cuvinte care să aibă o probabilitate de apariție mai mare, cuvinte mai comune pentru o anumite categorie, probabilitatea acestor cuvinte se calculează împărțind întregul la numărul de apariții ale cuvântului, astfel asigurând faptul că aceste cuvinte comune, folosite mai des într-un review, au o probabilitate

mai scăzută și gradul de încredere în această probabilitatea pentru a realiza clasificarea este mai mic. Ca exemplu putem să ne gândim la cuvântul cheie lemn, fiind un cuvânt mai comun, ar trebui să aibă o “greutate” mai mică în luarea deciziei asupra clasei, decât cuvântul masa, sau comodă. (Figura 27)

- Pentru a ne apropia de clasificarea propriu-zisă, stocăm într-un dicționar probabilitatea obținută de fiecare clasă în parte, calculăm probabilitatea maximă și returnăm clasa care a obținut-o.
- Dacă review-ul nu conține nici un cuvânt cheie din toate categoriile, atunci probabilitățile pentru fiecare clasă în parte vor fi 0, moment în care afișăm utilizatorului un mesaj de informare, review-ul nefiind clasificat::

În cazul în care categoria a fost returnată de pe backend cu succes către client, va fi afișată în label-ul de informare din formularul de adăugare a review-ului; dacă dintr-un anumit motiv, a apărut o eroare, va fi afișată pe client cu ajutorul unui alert;

După ce categoria sugerată a ajuns pe client, este realizat un alt apel Ajax către o altă metodă de pe backend, "LoadData" care returnează toate firmele aparținând categoriei sugerate din baza de date către client, acestea fiind încărcate într-un dropdown din formularul de adăugare review.

După alegerea numărului de stele și apăsarea butonului de submit, dacă utilizatorul este logat, review-ul este salvat în baza de date; Este realizată pe partea de server și o validare a câmpurilor introduse, pentru a se evita tentativa de a introduce valori nule în baza de date; dacă utilizatorul nu este autentificat, butonul de Adăugare îl va trimite către pagina de Autentificare:

Tot prin intermediul clasificatorului Bayes Naiv, am implementat pe pagina principală funcționalitatea de a căuta o anumită firmă după domeniul de activitate. Ca exemplu, putem căuta în search box-ul de pe pagina principală o listă de firme care se ocupă de realizarea de dormitoare:



Figura 14: Bara de cautare

De această dată, toată logica legată de clasificarea firmelor pe baza domeniului introdus în searchbox este realizată pe partea de server, în metoda `GetFirmasByDomain`, care are aceeași implementare ca metoda de clasificare pentru review, însă de această dată datele de antrenament sunt schimbate, cuvintele cheie fiind acum domeniile specifice fiecărei firme.

Metoda `GetFirmasByDomain` primește domeniul sub formă de string ca și parametru din bara de căutare, fiind apelată la apăsarea tastei Enter sau a butonului de search reprezentat prin imaginea unei lupe albastre, de către event-ul asociat butonului.

Dacă utilizatorul optează pentru cea de-a doua metodă de a adăuga review-ul în aplicație, acesta trebuie să navigheze pas cu pas din meniul de pe pagina principală la categoria dorită, apoi la firma dorită, unde, dacă nu a mai adăugat nici un review pentru acea firmă, butonul "Adaugă review" este enable și la apăsarea lui îl va trimite pe utilizator la un formular de adăugare.

Nume:

Bazgan

Prenume:

Dorian

Introduceți review-ul:

Ultima experienta cu aceasta firma a fost una destul de reusita, sunt multumit de calitatea realizarii comodel de care am dat comanda. Recomand!

Acordați un număr de stele:

1 2 3 4 5

☐ Send as anonim

Submit

Figura 15: Formularul de introducere review

Aici, spre deosebire de pagina principală, utilizatorul are posibilitatea să adauge review-ul sub anonim, pe baza unui alias pe care îl alege în acel moment. Dacă utilizatorul bifează checkbox-ul "Trimite anonim", câmpurile Nume și Prenume dispar, apare un singur textbox cu numele Alias.

Pe partea de server, se verifică dacă a fost bifat checkbox-ul, și în funcție de această variabilă, sunt construită două entități Review diferite, având sau nu câmpul Alias setat. După adăugarea review-ului, câmpurile formularului sunt resetate.

- Afișarea review-urilor:

Pagină unde sunt afișate review-urile este accesibilă de pe pagina cu lista firmelor, secțiunea "Vezi review-uri". Pentru a afișa toate review-urile pentru o anumită firmă, am realizat următoarea implementare:

Dacă utilizatorii au drept de administrator asupra aplicației, pe această pagină va fi disponibil butonul de "Admin" care conduce utilizatorul pe o pagină de management a aplicației.

Butonul de adăugare review va fi dezactivat dacă utilizatorul a adăugat deja un review pentru firma respectivă. Metoda CanSubmit verifică dacă utilizatorul stocat în sesiunea curentă(utilizatorul autentificat) a mai adăugat un review în baza de date la firma respectivă.

Pentru afișarea review-urilor, am construit un tabel dinamic pe partea de backend, care se încarcă odată cu pornirea paginii, în metoda Page_Load.

Metoda SelectReviewFirma aduce din baza de date toate review-urile aprobate de admin pentru o anumită firmă, iar

Pentru fiecare review din lista "revs" se crează tabela dinamic din code behind folosind controalele de tip TableRow, TableCell, Label, TextBox, ImageButton.

- prin instanțierea clasei TableRow se crează un rând în viitorul tabel;
- prin instanțierea clasei TableCell se creează o celulă din rândul creat anterior;
- instanțierea clasei Control oferă posibilitatea de a crea diferite controale, cum ar fi Label, Textbox; acestea pot fi stilizate atribuindu-le o clasă CSS definită în fișierul de style, putem modifica fontul, putem seta textul dorit, toate acestea direct pe partea de server;
- în final se urmează o ierarhia pentru a poziționa fiecare control la poziția lui finală: label-urile, textbox-urile trebuie adăugate în celulă, celula trebuie adăugată la rând, iar rândurile se adaugă la tabelul definit în html:

```
<asp:Table ID="tablerev" runat="server" CssClass="table table-responsive table-bordered table-striped my-class"></asp:Table>
```

```
tablerev.Rows.Add(row1);  
tablerev.Rows.Add(row2);  
tablerev.Rows.Add(row3);  
tablerev.Rows.Add(row4);
```

Figura 16: Adăugarea rândurilor create pe server în tabelul de pe client

Pentru a implementa butonul de like de lângă fiecare review am folosit un control de tip ImageButton căruia i-am atribuit o imagine, și i-am setat evenimentul de adăugare în baza de date a like-ului:

```

Control ctrlx = new Control();
ctrlx = new ImageButton();
((ImageButton)ctrlx).ImageUrl = "images/like.png";
((ImageButton)ctrlx).ID = reviewId.ToString();
((ImageButton)ctrlx).Click += ImageBtn_Click;
((ImageButton)ctrlx).CssClass = "like-image";

```

Figura 17: Crearea butonului de like

În evenimentul asociat ImageButton-ului, ImageBtn_Click este implementată logică de adăugare a like-ului, se verifică dacă utilizatorul autentificat a mai dat like la acel review, iar dacă a mai dat, la o nouă apăsare a butonului, se șterge like-ul din baza de date, funcționalitate ce permite utilizatorului să își retragă la un anumit moment aprecierea față de review-ul altui utilizator. Dacă utilizatorul nu a mai apreciat review-ul, like-ul sau va fi salvat în baza de date.

La partea de design a aplicației, am definit anumite stiluri proprii în fișierele locale, și o mare parte din stiluri cum ar fi butoanele, tabelele, formularele, fac parte din Bootstrap:

```

<div>
    <asp:Table ID="tablerev" runat="server" CssClass="table table-responsive table-bordered table-striped my-class"></asp:Table>
</div>

<asp:Button ID="addreview_btn" CssClass="btn btn-primary" runat="server" OnClick="addreview_btn_Click" Text="A" />
<asp:Button ID="admin_btn" CssClass="btn btn-default" runat="server" OnClick="admin_btn_Click" Text="Admin" />

```

Figura 18: Folosirea claselor Bootstrap pentru stilizarea butoanelor și a tabelor

4. Experimente

La partea de experimente am ales să testez acuratețea clasificatorului Bayes, fiind un algoritm de învățare automată, pot exista diferențe între realitate și clasificarea returnată de acest algoritm. Prin urmare am creat un proiect nou de tip Console Application pentru a putea testa acest algoritm și am stocat într-o listă câteva review-uri cu anumite caracteristici specifice unui mediu de test:

- Primele trei review-uri conțin câte un cuvânt cheie din fiecare categorie disponibilă în aplicație: mobilă, construcții și foraje fântâni
- Următoarele review-uri conțin anumite combinații de cuvinte cheie din fiecare categorie pentru a vedea cum clasifică algoritmul în anumite situații mai complicate.

```
static string pattern = "Am avut o experienta cu aceasta firma, ";
public static List<string> ListOfReviews = new List<string>
{
    pattern + "am comandat o masa, care a iesit cum ma asteptam. Recomand!", // 1 cuvânt mobila
    pattern + "mi-au construit o cabana de vara, duritatea betonului a iesit cam slaba.", // 1 cuvânt constructii
    pattern + "mi-au construit un put de mare adancime, cu tuburi de 60 de cm.", // 1 cuvânt foraje fantani
    pattern + "au facut o finisare la o masa, care a iesit cum ma asteptam. Recomand!", // 1 cuv constructi, 1 cuvânt mobila
    pattern + "mi-au turnat beton pe o terasa, pe care au amenajat o masa si o " +
    "comoda si a iesit cum ma asteptam. Recomand!", // cate un cuvânt din fiecare categorie
    pattern + "mi-au realizat o cabana din caramida si beton, si o masa, care au iesit cum ma asteptam. Recomand!", // 2 cuv constructii,
    pattern + "mi-au turnat un trotuar de beton langa fantana si au lucrat bine. Recomand!", // 2 cuv constructii, 1 cuvânt foraje fantan:
    pattern + "au sapat un put de mare adancime, si au placat-o cu caramida. Au lucrat bine." //1 cuv fantani, 1 cuv constructii
};
```

Figura 19: Review-urile alese pentru realizarea experimentelor

- Am stocat într-o listă clasificările reale pentru aceste review-uri, iar pentru fiecare review din lista de review-uri de test am salvat într-un dicționar index-ul fiecărui review și clasificarea returnată de algoritm la rulare;

După ce s-au stocat aceste date de test, se compară clasificările corecte cu clasificările returnate de algoritm:

```

Expected: 0 -> mobila ----- Classified: mobila
Expected: 1 -> constructii ----- Classified: constructii
Expected: 2 -> foraje fantani ----- Classified: foraje fantani
Expected: 3 -> mobila ----- Classified: mobila
Expected: 4 -> constructii ----- Classified: mobila
Expected: 5 -> constructii ----- Classified: constructii
Expected: 6 -> constructii ----- Classified: constructii
Expected: 7 -> foraje fantani ----- Classified: constructii

```

Figura 20: Rezultatele experimentelor

Astfel, am observat că dintr-un număr de 8 review-uri alese pentru test, algoritmul a clasificat corect 6 dintre ele, și 2 greșit. Acest lucru s-a datorat următoarelor aspect:

Algoritmul Bayes Naiv calculează un scor de apariție pentru fiecare cuvânt din review într-o anumită clasă din totalul de 3. Dacă review-ul conține un cuvânt cheie dintr-o categorie, și alt cuvânt cheie din altă categorie, calculul probabilității apartenenței review-ului fiecărei clase poate fi, spre exemplu, următorul:

```

{
  Mobila: 1
  Constructii: 1
  Foraje fantani: 0
}

```

Acest lucru înseamnă că algoritmul a găsit un cuvânt cheie specific categoriei mobilă, un cuvânt cheie specific categoriei construcții și 0 cuvinte cheie din categoria foraje fântâni.

Pentru a putea face o clasificare finală, următorul pas din clasificare este calcularea maximului celor trei probabilități, categoria cu probabilitatea maximă fiind predicția dată de algoritm. Având două categorii cu aceeași probabilitate, algoritmul nu poate da un răspuns 100%, maximul dintre două numere egale, este unul din cele două, algoritmul returnând fie categoria mobilă, fie categoria construcții ca și clasificare finală.

Același lucru se întâmplă și în momentul în care review-ul introdus de utilizator conține câte un cuvânt din toate cele trei categorii:

{
 Mobila: 1
 Constructii: 1
 Foraje fantani: 1
 },

clasificarea finală fiind una din cele trei categorii.

Review	Cuvinte cheie	Clasificare corecta	Clasificare Bayes	Predictie
1	1 cuvant mobila	categoria mobila	categoria mobila	corect
2	1 cuvant constructii	categoria constructii	categoria constructii	corect
3	1 cuvant foraje fantani	categoria foraje fantani	categoria foraje fantani	corect
4	1 cuvant constructii, 1 cuvant mobila	categoria mobila	categoria mobila	corect
5	cate un cuvant din fiecare categorie	categoria constructii	categoria mobila	gresit
6	2 cuvinte constructii, 1 cuvant mobila	categoria constructii	categoria constructii	corect
7	2 cuvinte constructii, 1 cuvant foraje fantani	categoria constructii	categoria constructii	corect
8	1 cuvant fantani, 1 cuvant constructii	categoria foraje fantani	categoria constructii	gresit

Probabilitatea de prezicere este de 6 review-uri clasificate corect / 8 review-uri în total = 0,75.

Din datele de test, am constatat că algoritmul de clasificare Bayes Naiv a avut un procent de acuratețe de 75%, rezultat care poate diferi în funcție de review-urile introduse pentru test, în cazul de față.

5. Concluzii

Scopul aceste lucrări de licență a fost acela de a crea o comunitate online care să ajute utilizatorii să aleagă o firmă prestatoare de servicii pentru anumite domenii (mobilă, construcții, foraje fântani) pe baza review-urilor altor utilizatori care au trecut deja printr-o experiență cu o anumită firmă.

Astfel, utilizatorii care au nevoie de o a doua părere înainte de a angaja o anumită firmă, pot citi review-urile acestei firme și își pot crea o opinie proprie în legătură cu firma, care îi va ajuta să ia sau nu o anumită decizie.

În această lucrare am încercat să descriu principalele etape ale dezvoltării aplicației web, punând accent atât pe partea de structură, cât și pe partea de implementare adăugând fragmente de cod care descriu anumite funcționalități din aplicație.

În implementare s-au folosit tehnologii ca ASP .NET WebForms, cu C# ca limbaj de programare pe partea de server, iar la partea de client s-a folosit Javascript.

Această aplicație ar putea fi îmbunătățită prin optimizarea algoritmului folosit pentru clasificarea automată a review-urilor și a domeniilor, prin îmbogățirea datelor de antrenament, sau extinderea sa la alte culturi.

6. Bibliografie:

1. Dejan Sarka, Milos Radivojevic, Wiliam Durkin, SQL Server 2017 Developer's Guide
2. Tom Archer, *Inside C#, 2nd Edition*
3. Adrian Turtzchi, *C# .NET Web Developer's Guide*
4. Hungry Minds, *C# Bible*
5. GABRIEL, Enea, *WebForms, o tehnologie ASP.NET*
6. ASP .NET Web Forms
<https://docs.microsoft.com/en-us/aspnet/web-forms/>
7. JavaScript
<https://www.javascript.com/>
8. Bootstrap Components
<https://getbootstrap.com/docs/3.3/components/>
9. Entity Framework
<https://docs.microsoft.com/en-us/ef/>
10. SQL Server
<https://www.microsoft.com/en-us/sql-server/sql-server-2017>

7. Appendix:

```
<asp:TextBox ID="passwd_textbox" runat="server" CssClass="form-control register-textboxes" TextMode="Password"></asp:TextBox>
<asp:RequiredFieldValidator ID="PasswordRequiredFieldValidator" runat="server"
    ControlToValidate="passwd_textbox" ErrorMessage="Parola trebuie completata!"
    SetFocusOnError="True" Display="Dynamic">
</asp:RequiredFieldValidator>
```

Figura 21: Validator pentru camp obligatoriu

```
<asp:TextBox ID="email_textbox" runat="server" CssClass="form-control register-textboxes"></asp:TextBox>
<asp:RequiredFieldValidator ID="EmailRequiredValidator" runat="server"
    ControlToValidate="email_textbox" ErrorMessage="Emailul trebuie completat!"
    SetFocusOnError="True" Display="Dynamic">
</asp:RequiredFieldValidator>
<asp:RegularExpressionValidator ID="RegularExpressionEmailValidator" runat="server"
    ErrorMessage="Email invalid!" ControlToValidate="email_textbox"
    SetFocusOnError="True"
    ValidationExpression="\w+([-+.']\w+)*@\w+([-.]\w+)*\.\w+([-+]\w+)*" Display="Dynamic">
</asp:RegularExpressionValidator>
```

Figura 22: Validare email pe baza expresiilor regulate

```

public class PasswordAdvisor
{
    1 reference | DESKTOP-F7T6ER8\Dorian, 6 days ago | 1 author, 2 changes | 0 exceptions
    public static bool CheckStrength(string password)
    {
        string patdi = @"\d+"; //match digits
        string patupp = @"[A-Z]+"; //match upper cases
        string patlow = @"[a-z]+"; //match lower cases

        Match id = Regex.Match(password, patdi);
        Match upp = Regex.Match(password, patupp);
        Match low = Regex.Match(password, patlow);

        if (id.Success && upp.Success && low.Success)
        {
            return true;
        }

        return false;
    }

    1 reference | DESKTOP-F7T6ER8\Dorian, 6 days ago | 1 author, 1 change | 0 exceptions
    public static bool CheckLength(string password)
    {
        var passwordCharArray = password.ToCharArray();
        bool hasRequiredLength = passwordCharArray.Length >= 6 && passwordCharArray.Length <= 20;

        if (hasRequiredLength)
        {
            return true;
        }

        return false;
    }
}

```

Figura 23: Validare parola pe partea de server

```

if(!isPasswordStrength)
{
    successregister_lbl.Text =
        "Parola trebuie sa contina o combinatie de caractere mari, mici, cifre si simboluri!";
}
if(!hasOptimumLength)
{
    successregister_lbl.Text =
        "Parola trebuie sa contina intre 6 si 20 de caractere!";
}

```

Figura 24: Mesajele afișate la validarea câmpurilor

```

using (var db = new TheContext())
{
    var _email = (from u in db.Users
        where u.Email == email
        select u.Email).SingleOrDefault();
    if (_email != null)
    {
        successregister_lbl.Text = "Exista deja un cont cu email-ul introdus!";
    }
    else
    {
        var user = new User { LastName = nume, FirstName = prenume, Email = email, Password = finalEncryptedPasswordString, Salt = salt };
        db.Users.Add(user);
        db.SaveChanges();
        successregister_lbl.Text = "Te-ai inregistrat cu succes!";
        Response.AddHeader("REFRESH", "2;URL=Logare.aspx");
    }
}

```

Figura 25: Salvarea utilizatorului in baza de date

```

static byte[] ConvertPlainTextToBytes(string plainText)
{
    return Encoding.UTF8.GetBytes(plainText);
}

```

Figura 26: Convertirea parolei in byte array

```

private static byte[] CreateSalt(int size)
{
    //Genereaza un numar cryptographic random
    RNGCryptoServiceProvider rng = new RNGCryptoServiceProvider();
    byte[] buff = new byte[size];
    rng.GetBytes(buff);

    return buff;
}

```

Figura 27: Crearea salt-ului pentru alipirea cu parola

```

static byte[] GenerateSaltedHash(byte[] plainText, byte[] salt)
{
    HashAlgorithm algorithm = new SHA256Managed();

    byte[] plainTextWithSaltBytes = new byte[plainText.Length + salt.Length];

    for (int i = 0; i < plainText.Length; i++)
    {
        plainTextWithSaltBytes[i] = plainText[i];
    }

    for (int i = 0; i < salt.Length; i++)
    {
        plainTextWithSaltBytes[plainText.Length + i] = salt[i];
    }
    return algorithm.ComputeHash(plainTextWithSaltBytes);
}

```

Figura 28: Concatenarea salt-ului la parola și criptarea parolei obținute

```

$.ajax({
    type: "POST",
    url: classificationUrl,
    async: false,
    data: JSON.stringify({ myReview: review }),
    contentType: 'application/json; charset=utf-8',
    dataType: 'json',
    success: function (data) {
        labelClassificationName.innerHTML = "Categorie sugerata: " + data.d;
        classificationCategory = data.d;
    },
    error: function (data, success, error) {
        alert("Error: " + error);
    }
});

```

Figura 29: Apelul Ajax către metoda de pe server care returnează clasificarea algoritmului Bayes Naiv

[WebMethod]

0 references | CODE40\dorian.bazgan, 1 hour ago | 2 authors, 6 changes

```
public static string ReviewValue(string myReview)
{
    var finalClassification = string.Empty;
    var hasCategoryKeyword = HasKeyword(myReview, TrainingDictionaryDataReviews);
    if (myReview != string.Empty && hasCategoryKeyword)
    {
        finalClassification = Classify(myReview, TrainingDictionaryDataReviews);
    }
    return finalClassification;
}
```

Figura 30: Metoda de pe server care apelează algoritmul de clasificare

2 references | CODE40\dorian.bazgan, 1 hour ago | 2 authors, 2 changes

```
public static bool HasKeyword(string review, Dictionary<string, List<string>> trainingDocuments)
{
    List<string> words = trainingDocuments.SelectMany(x => x.Value).ToList();
    List<string> wordsFromReview = review.Split(' ').ToList();

    foreach (var word in wordsFromReview)
    {
        if (words.Contains(word))
        {
            return true;
        }
    }
    return false;
}
```

Figura 31: Metoda care verifică dacă review-ul conține cuvinte cheie specifice uneia dintre categorii

13 references | DESKTOP-F7T6ER8\Dorian, 14 hours ago | 2 authors, 2 changes

```
public class Document
{
    2 references | DESKTOP-F7T6ER8\Dorian, 14 hours ago | 1 author, 1 change
    public string TheClass { get; set; }
    2 references | DESKTOP-F7T6ER8\Dorian, 14 hours ago | 1 author, 1 change
    public string Sentence { get; set; }

    6 references | DESKTOP-F7T6ER8\Dorian, 14 hours ago | 2 authors, 2 changes
    public Document(string theClass, string sentence)
    {
        TheClass = theClass;
        Sentence = sentence;
    }
}
```

Figura 32: Clasa ajutătoare în procesul de clasificare, stochează categoriile și propozițiile

```
//datele de antrenament pentru clasificarea review-urilor
static List<Document> trainDataCategories = new List<Document>
{
    new Document("mobila",
        "masa mesei masuta masutei comoda comodei biblioteca bibliotecii PAL nuc fag extensibil rectangular picioare finisaje " +
        "lac lemn imbinari lac nitrolac slefuit incleiat fin"),
    new Document("constructii",
        "beton betonului nisip nisipului var varului fier fierului etrier etriere tencuiala finisare finisaj manopera acoperis " +
        "lemn caramida ciment materiale BCA bca sarma striat"),
    new Document("foraje fantani",
        "freza frezei apa apei huma lut lutului var pietre piatra motor motorului metri metru m tuburi margaritar cm centimetri " +
        "forat foraj sol pamant")
};

//datele de antrenament pentru clasificarea domeniilor
private static List<Document> trainDataDomains = new List<Document>
{
    new Document("mobila",
        "dormitoare dormitor bucatarie bucatarii baie bai living sufragerie sufragerii"),
    new Document("constructii",
        "case casa acoperis magazine magazii beci beciuri anexe anexa cabana cabane"),
    new Document("foraje fantani",
        "puturi fantani arteriziana fantana put")
};
```

Figura 33: Datele de antrenament pentru clasificarea review-uri și a firmelor pe domenii

```
1 reference | DESKTOP-F7T6ER8\Dorian, 15 hours ago | 1 author, 1 change
public static double CalculateScoreForClass(string sentence, string className,
    Dictionary<string, List<string>> corpus)
{
    double probability = 0;
    var countWords = CountingWords(corpus);

    List<string> wordsForClass = corpus.Where(x => x.Key == className).SelectMany(x => x.Value).ToList();

    List<string> wordsFromReview = Regex.Replace(sentence, "\\p{P}+", "").Split(' ').ToList();

    foreach (var word in wordsFromReview)
    {
        if (wordsForClass.Contains(word))
        {
            probability += 1 / (double) countWords[word.ToLower()];
        }
    }

    return probability;
}
```

Figura 34: Metoda care calculează probabilitatea apartenenței unui review la o clasă


```

//calculeaza numarul de aparatii a fiecarui cuvânt din datele de antrenament
1 reference | DESKTOP-F7T6ER8\Dorian, 15 hours ago | 1 author, 1 change
public static Dictionary<string, int> CountingWords(Dictionary<string, List<string>> trainingDocuments)
{
    Dictionary<string, int> countingWords = new Dictionary<string, int>();

    List<string> words = trainingDocuments.SelectMany(x => x.Value).ToList();

    foreach (var word in words)
    {
        if (countingWords.ContainsKey(word))
        {
            countingWords[word]++;
        }
        else
        {
            countingWords.Add(word, 1);
        }
    }

    return countingWords;
}

```

Figura 35: Metoda care contorizează numărul de apariții a unui cuvânt într-o propoziție

2 references | DESKTOP-F7T6ER8\Dorian, 16 hours ago | 1 author, 1 change

```
public static string Classify(string sentence, Dictionary<string, List<string>> corpusDictionary)
{
    Dictionary<string, double> classificationDictionary = new Dictionary<string, double>();

    var countingZeros = 0;
    foreach (var @class in corpusDictionary.Keys)
    {
        var probability = CalculateScoreForClass(sentence, @class, corpusDictionary);
        classificationDictionary.Add(@class, probability);
    }

    var allValues = classificationDictionary.Values.ToList();
    foreach (var value in allValues)
    {
        if (Math.Abs(value) < 0.01)
        {
            countingZeros++;
        }
    }

    if (countingZeros == classificationDictionary.Count)
    {
        return "Nici o potrivire";
    }

    var classifiedCategory = classificationDictionary
        .FirstOrDefault(x => Math.Abs(x.Value - classificationDictionary.Values.Max()) < 0.01).Key;

    return classifiedCategory;
}
```

Figura 36: Metoda care returnează categoria către client pe baza probabilității returnate de clasificator

```
[WebMethod]
0 references | CODE40\dorian.bazgan, 51 days ago | 2 authors, 8 changes | 0 exceptions
public static List<Firma> LoadData(string myCategory)
{
    var categoryId = GetCategoryId(myCategory);
    _context = new TheContext();
    var firme = _context.Firme.Include(x => x.Categorie).Where(x => x.Categorie.CategoryId == categoryId)
        .ToList();
    return firme;
}
```

Figura 37: Aducerea firmelor din baza de date care corespund categoriei returnate de clasificator

```

$.ajax({
    type: "POST",
    url: dropdownUrl,
    contentType: 'application/json; charset=utf-8',
    data: JSON.stringify({ myCategory: classificationCategory }),
    success: function (result) {
        var ddlDrop = $("[id*=firmeDropdownList]");
        if (result.d.length === 0) {
            ddlDrop.empty();
            ddlDrop.append('<option>' + 'Nu exista firme' + '</option>');
        }
        for (var i = 0; i < result.d.length; i++) {
            if (ddlDrop[0][i] !== undefined && ddlDrop[0][i] !== result.d[i].Nume) {
                ddlDrop.empty();
                ddlDrop.append('<option>' + result.d[i].Nume + '</option>');
            } else {
                ddlDrop.append('<option>' + result.d[i].Nume + '</option>');
            }
        }
    },
    failure: function () {
        alert("Error");
    }
})

```

Figura 38: Încărcarea firmelor specifice categoriei sugerate de clasificator într-un dropdown pe client

```

if (CanSubmitDefault(viewreviews.SessionUserId, firmaId))
{
    using (var db = new TheContext())
    {
        var review = new Review
        {
            FirmaId = firmaId,
            UserId = viewreviews.SessionUserId,
            Comment = comment,
            NrStele = rating
        };
        db.Reviews.Add(review);
        db.SaveChanges();
    }
    classifyReviewTextarea.Value = string.Empty;
    successLabel.Text =
        "Review-ul a fost adaugat cu succes! Va apare in sectiune de review-uri dupa aprobarea unui admin.";
}
else
{
    successLabel.Text = "Ai mai adaugat un review la firma selectata!";
}

```

Figura 39: Adăugarea review-ului în baza de date de pe pagina principală

```
protected void submitReview_LinkBtn_Click(object sender, EventArgs e)
{
    if (Session["USER_EMAIL"] == null)
    {
        Response.Redirect("~/Logare.aspx");
    }
}
```

Figura 40: Redirecționarea către pagina de autentificare

```
protected void btnSearchBox_Click(object sender, EventArgs e)
{
    var userSearch = Request.Form[searchBox.UniqueID];
    var bayesClassification = GetFirmasByDomain(userSearch);
    var categoryId = GetCategoryId(bayesClassification);
    Response.Redirect("1.aspx?id="+categoryId);
}
```

Figura 41: Clasificarea firmelor pe baza domeniilor de activitate

```
function EnableControl() {
    var txtboxNume = document.getElementById('<%= nume_textbox.ClientID %>');
    var txtboxPrenume = document.getElementById('<%= prenume_textbox.ClientID %>');
    var checkboxAnonim = document.getElementById('<%= AnonimOption.ClientID %>');
    var numeLabel = document.getElementById('<%= numerev_lbl.ClientID %>');
    var prenumeLabel = document.getElementById('<%= prenumerev_lbl.ClientID %>');
    var prenumeValue = '<%= viewreviews.SessionFirstName%>';

    if (checkboxAnonim.checked) {
        prenumeLabel.innerHTML = "Alias";
        txtboxPrenume.removeAttribute('disabled');
        txtboxPrenume.value = "";
        txtboxNume.style.visibility = "hidden";
        numeLabel.style.visibility = "hidden";
    } else {
        txtboxNume.style.visibility = "visible";
        numeLabel.style.visibility = "visible";
        txtboxPrenume.disabled = true;
        txtboxNume.disabled = true;
        txtboxPrenume.value = prenumeValue;
        prenumeLabel.innerHTML = "Prenume";
    }
}
```

Figura 42: Afișarea și ascunderea input-urilor pentru numele utilizatorului din feedback

```

    if (AnonimOption.Checked)
    {
        var review = new Review
        {
            FirmaId = id,
            UserId = viewreviews.SessionUserId,
            Comment = comment,
            NrStele = intRating,
            Alias = alias
        };
        db.Reviews.Add(review);
        db.SaveChanges();
    }
    else
    {
        var anonimReview = new Review
        {
            FirmaId = id,
            UserId = viewreviews.SessionUserId,
            Comment = comment,
            NrStele = intRating
        };
        db.Reviews.Add(anonimReview);
        db.SaveChanges();
    }
}
nume_textbox.Text = string.Empty;
prenume_textbox.Text = string.Empty;
TextArea1.Value = string.Empty;
Label1.Text = "Review-ul a fost adaugat cu succes!";
Response.AddHeader("REFRESH", "2;URL=viewreviews.aspx?id=" + id);

```

Figura 43: Crearea review-ului care conține numele utilizatorului sau alias-ul

```

using (var db = new TheContext())
{
    var email = (from u in db.Users
                 where u.AccessMember == "admin"
                 select u.Email).SingleOrDefault();

    if (Session["USER_EMAIL"] != null)
    {
        if (Session["USER_EMAIL"].ToString() == email)
        {
            admin_btn.Visible = true;
        }
        else
        {
            admin_btn.Visible = false;
        }
    }
    else
    {
        admin_btn.Visible = false;
    }
}

```

Figura 44: Afișarea butonului pentru administrator

1 reference | Dorian Bazgan, 243 days ago | 1 author, 2 changes | 0 exceptions

```

public bool CanSubmit(int userId)
{
    using (var db = new TheContext())
    {
        var id = int.Parse(Request.QueryString["id"]);
        var review = (from r in db.Reviews
                      where r.FirmaId == id
                      where r.UserId == SessionUserId
                      select r).SingleOrDefault();
        if (review != null)
        {
            return false;
        }
        return true;
    }
}

if (!CanSubmit(SessionUserId))
{
    addreview_btn.Enabled = false;
    addreview_btn.ToolTip = "Ai adaugat deja un review!";
}

```

Figura 45: Limitarea utilizatorului de a adăuga un singur review pentru o firmă

1 reference | Dorian Bazgan, 243 days ago | 2 authors, 5 changes | 0 exceptions

```

public List<Review> SelectReviewFirma(int id)
{
    using (var db = new TheContext())
    {
        var review = (from r in db.Reviews
                      where r.FirmaId == id
                      where r.Approved == 1
                      select r).ToList();
        return review;
    }
}

var revs = SelectReviewFirma(_id);

```

Figura 46: Încărcarea din baza de date a review-urilor aprobate pentru o anumită firmă

```

foreach (var item in revs)
{
    var reviewId = item.ReviewId;

    //randul 1
    TableRow row1 = new TableRow();

    //celula1
    TableCell cell1 = new TableCell();
    Control ctrl1 = new Control();
    ctrl1 = new Label();
    ((Label)ctrl1).CssClass = "formLabel";
    ((Label)ctrl1).Text = "Nume utilizator: ";
    ((Label)ctrl1).Font.Bold = true;
    cell1.Controls.Add(ctrl1);
    row1.Cells.Add(cell1);
}

```

Figura 47: Crearea tabelului dinamic care va conține tot feedback-ul utilizatorilor

```

using (var db = new TheContext())
{
    var likedReview = new LikedReview
    {
        ReviewId = btnId,
        UserId = SessionUserId,
        Like = 1
    };
    if (AlreadyLiked(SessionUserId, btnId))
    {
        var reviewLiked = ReturnExistLikedReview(SessionUserId, btnId);
        db.LikedReviews.Attach(reviewLiked);
        db.LikedReviews.Remove(reviewLiked);
        db.SaveChanges();
    }
    else
    {
        db.LikedReviews.Add(likedReview);
        db.SaveChanges();
    }
}

```

Figura 48: Adăugarea sau ștergerea like-ului pentru un review

```

public static List<string> ExpectedClassifications = new List<string>()
{
    "mobila",
    "constructii",
    "foraje fantani",
    "mobila",
    "constructii",
    "constructii",
    "constructii",
    "foraje fantani"
};

```

Figura 49: Clasificările corecte ale review-urilor de test din capitolul Experimente

```
Dictionary<int, string> ClassifiedCategories = new Dictionary<int, string>();  
  
foreach (var item in ListOfReviews)  
{  
    ClassifiedCategories.Add(ListOfReviews.IndexOf(item), _Default.Classify(item, _Default.TrainingDictionaryDataReviews));  
}
```

Figura 50: Salvarea index-ului review-ului și clasificarea sa returnată de algoritmul Bayes