**1.INREGISTRARE**

-nume si prenume

-username: mai mare > 8 caractere si cu litere mici

-email: sa se verifice ca e valid

-parola: sa contina mai mult de 8 caractere, o litera mare si un caracter special

- sa aiba o bifa obligatorie pentru confirmarea de citire termeni si conditii

- sa se poata realiza doar cand da click pe un link de securitate primit prin email.

**2.LOGIN**

-username

-parola

-daca nu sunt corecte, atentionare prin mesaj

-am uitat parola si sa isi poate genera alta prin adresa de mail

**3. MENIUL PRINCIPAL**

-***incarca o creatie literara*:** la click pe actiunea asta(cum e in html-ul din aplicatia implementata), utilizatorul este obligat sa incarce: nume creatie literara, coperta(o poza specifica), creatia in format pdf, genul creatiei(va avea sa aleaga dintr-o lista afisata) si numele autorului va fi scris automat de sistem dupa datele de la inregsitrare(nu va putea modifica textbox-ul acela).

TOATE CAMPURILE SUNT OBLIGATORII!!!

Aceasta creatie literara va fi trimisa administratorului pentru a fi validata sau respinsa.

Administratorul va avea sectiunile: primite, in curs de verificare( va da asta din sectiunea primite) si validata sau respinsa(aici va si detalia motivele). Statusul va fi automat actualizat de fiecare data si in pagina utilizatorului in sectiunea „Creatiile mele literare”.

Administratorul va citi cartea si daca e posibil, va putea sa o verifice antiplagiat printr un program anume.

Utilizatorul va fi instiintat printr-o notificare in aplicatie daca a fost respinsa sau acceptata.

-***toate creatiile literare:*** aici fiecare utilizator va vedea toate creatiile literare incarcate de ceilalti(cele care au fost validate de administrator).

Aici, fiecare creatie va avea elementele puse de utilizator la incarcare, iar prin click pe „citeste creatie literara”, utilizatorului i se va deschide pdf-ul corespunzator. Utilizatorul nu va putea salva sau modifica creatia literara, ci doar va putea selecta un fragment de minim 500 de cuvinte si ii va putea asocia un anumit sentiment. Printr un buton aparut jos, daca va da click pe el, va scrie „asociaza sentiment” si daca selecteaza un fragment corespunzator, va putea alege dintr o lista data sentimentul (fericire, iubire, tristete, motivatie) acolo intr un text box si el va fi trimis in baza de date cu titlul operei si textul fragmentului.

Tot aici, creatia literara va avea in dreptul ei butonul „Recomanda” si i se vor deschide utilizatorii aplicatiei, cat si „acorda nota” sau „adauga recenzie”.

-***creatiile mele literare:*** fiecare utilizator va vedea ce a incarcat el si va putea filtra dupa: toate, primite, validate, in curs de validare sau respinse.

-***formeaza un grup:*** un utilizator va putea forma un grup cu alti utilizatori, iar ceilalti vor putea accepta sau refuza intrarea in grup.

-***performanta mea:*** sa apara un grafic de performanta specific fiecarui utilizator in functie de media notelor primite pentru fiecare creatie literara.

-***cauta dupa sentiment*:** alege sentimentul dintr-o lista data si acolo sunt afisate toate framentele din operele corespunzatoare, care au avut asociat sentimentul ales.

-***profilul utilizatorului:*** va putea incarca o poza sau isi va putea schimba parola sau adresa de mail. Sa apara in stanga sus.

As elimina partea cu peretele, daca ia prea mult timp, daca nu, e ok sa apara si el in meniu, ca „Perete”.

Librarii/Framework-uri folosite:

* Font-end: Angular 9, Angular Material (pentru componentele ui)
* Back-end: Spring Boot (spring-security, spring-data-jpa, spring-mail), [Project Lombok](https://projectlombok.org/), [ModelMapper](http://modelmapper.org/)

Caracteristici Angular:

* Arhitectura bazata pe componente, practic fiecare functionalitate mare a aplicatiei poate fi implementata intr-o componenta/ mai multe componente
* Servicii – pentru separarea logicii ce nu are legatura cu un view(pagina html). Prin servicii se fac requesturile HTTP catre server, iar Angular pune la dispozitie un client simplificat, HttpClient
* Module – mai multe componente ce deservesc aceeasi funtionalitate se pot grupa in module
* Rounting – modul ce pune la dispozitie un serviciu ce permite definirea de rute in aplicatie

**Ce e mai jos am luat dintr-o documentatie la o aplicatie asemanatoare**

Pentru partea de back-end am ales Spring Boot, deoarece acesta introduce o serie de avantaje:

* Set standard de dependințe pentru a simplifica build-ul si configurarea aplicației
* Server web încorporat(Tomcat) pentru o lansare simplă
* Configurări automate, atunci când este posibil, pentru funcționalitățile Spring

Pentru interogarea bazei de date am folosit interfata JPARepository din Spring Data JPA. Aceasta ușurează procesul de dezvoltare prin expunerea unor metode (query methods) astfel încât interogările uzuale să se realizeze doar prin definirea acestor metode (de exemplu „*findByNUME\_CAMP*”).

Pentru a construi răspunsurile ce sunt folosite in partea de front-end, am creat clase separate (DTO – Data Transfer Object), care sunt bazate pe entități. Aceasta este o bună practică, deoarece este posibil să dorim ca unele câmpuri din entități să nu fie vizibile in aplicația client. Maparea dintre entități și obiectele de transfer se realizează cu ajutorul librăriei „*modelmapper*”.

Structura proiectelor

Proiectul back-end este structurat în felul următor:

* controller: conține clase anotate cu @RestController. Acestea sunt tipuri specializate de controllere, ce includ anotările @Controller și @ResponseBody, și, ca rezultat simplifică implementarea. Acestea oferă posibilitatea de a expune URL-uri pentru a fi apelate de către clienți.
* domain: conține entitățile JPA, anotate cu @Entity, ce mapează tabelele din baza de date.
* dto: conține clase ale căror instanțe vor fi întoarse ca răspuns în controller
* repository: conține interfețe ce extind una din variantele Spring Data Repositories, ce ajută la crearea într-un mod simplu a interogărilor prin definirea unor metode cu nume specifice.
* service: conține interfețele pentru servicii, cât și implementările acestora. Implementările sunt anotate cu @Service, iar în acestea se găsește logica/prelucrarea datelor

REFERINȚE

* Documentația oficială Spring: <https://spring.io/docs>
* Documentația oficială Angular: <https://angular.io/docs>
* Angular Material: <https://material.angular.io>

**Securitate**

Este o buna practica sa nu se stocheze parolele utilizatorilor in clar in baza de date. De aceea am folosit implementarea standard a interfetei *PasswordEncoder* din spring-security (*BCryptPasswordEncoder*). Prin aceasta se aplica un algoritm de criptare *one-way*, ceea ce înseamnă ca odată ce parola s-a criptat, rezultatul poate fi obținut doar prin aplicarea aceluiași algoritm pe același set de caractere.

**Project Lombok**

Este o librarie utlitara Java ce ofera anotari pentru generarea codului. De exemplu putem anota clasele cu *@Getter* pentru generarea de getter-ilor, *@Setter*  pentru generarea setter-ilor, s.a.m.d.

**Spring Email**

Prin utilizarea dependintei *spring-boot-starter-mail* se ofera acces la diferite interfete si implementari pentru trimiterea de e-mailuri. Configurarea contului si a serverului de pe care se vor trimite e-mailurile se face prin fisierul de configurare spring boot (application.properties)

Pe partea de front-end am folosit utilitare precum:

* [ng2-pdf-viewer](https://www.npmjs.com/package/ng2-pdf-viewer): librarie ce ofera suport pentru afisarea fisierelor pdf direct in pagina html.
* [bootstrap-utilities](https://www.npmjs.com/package/bootstrap-utilities) : puune la dispozitie clase css pentru padding, margin, etc.