

Inginerie Software

Documentatie proiect

Realizat de: Blaga Maria, Darolti Laura, Igna Alexandra, Scuturici Vlad

Aplicatie Social Media pentru animale de companie

1. Prezentarea proiectului

PetGram redefinește experiența rețelelor de socializare, plasând în centrul ei conexiunile umane, toate îmbrățișate de dragostea comună pentru animalele de companie. Aplicația oferă o modalitate simplă și rapidă de a crea conturi, facilitând împărtășirea de momente speciale prin încărcarea de imagini și descrieri. Funcționalitățile de like-uri și comentarii aduc o dimensiune suplimentară de interactivitate, iar adăugarea de prieteni extinde rețeaua socială.

PetGram nu se oprește la simpla socializare online, ci evoluează într-o comunitate virtuală unde oamenii se conectează și își dezvoltă prietenii bazate pe pasiunea comună pentru animalele de companie. De la notificări personalizate care mențin utilizatorii la curent cu activitățile lor până la interacțiuni autentice și plăcute, aplicația oferă o experiență completă și inegalabilă pentru iubitorii de animale de companie din întreaga lume.

2. Descriere

Limbajul folosit pentru implementarea proiectului este java și am folosit framework-urile Spring Boot și Vaadin.

Pachetele implementate sunt următoarele:

Model – conține clasele corespunzătoare tabelelor create in MySQL : Comments, Friends, Messages, Notifications, Pets, Posts, Users, Likes

Connection – conține clasa responsabila pentru realizarea conexiunii cu baza de date

Dataaccess- conține clasele ce furnizeaza accesul la date pentru entitățile din pachetul Model. Acestea permit metode precum inserarea, actualizarea, ștergerea și obținerea informațiilor din baza de date.

Petgram- conține clasele de implementare pentru paginile aplicației: pagina de login, register, profil, mesaje, notificari, feed.

LoginView- furnizează funcționalități pentru autentificarea utilizatorului, navigare și gestionarea erorilor.

ProfileView- Această clasă implementează o vizualizare a profilului utilizatorului, inclusiv informații despre utilizator, lista de animale de companie, fluxul de activitate și opțiuni pentru gestionarea profilului. De asemenea, oferă funcționalități precum adăugarea, ștergerea și editarea animalelor de companie, precum și crearea și vizualizarea de postări în fluxul de activitate.

MessageView- Această clasă implementează o vizualizare a mesajelor, permitând utilizatorilor să trimită și să primească mesaje. Clasa include o listă a utilizatorilor disponibili, un panou de afișare a mesajelor și un câmp de introducere a mesajelor.

SearchView- Această clasă implementează o vizualizare pentru căutarea utilizatorilor și interacțiunea cu aceștia în cadrul aplicației.

NotificationsView- Această clasă implementează afișarea notificărilor utilizatorilor, implementând modelul Observer pentru actualizări în timp real.

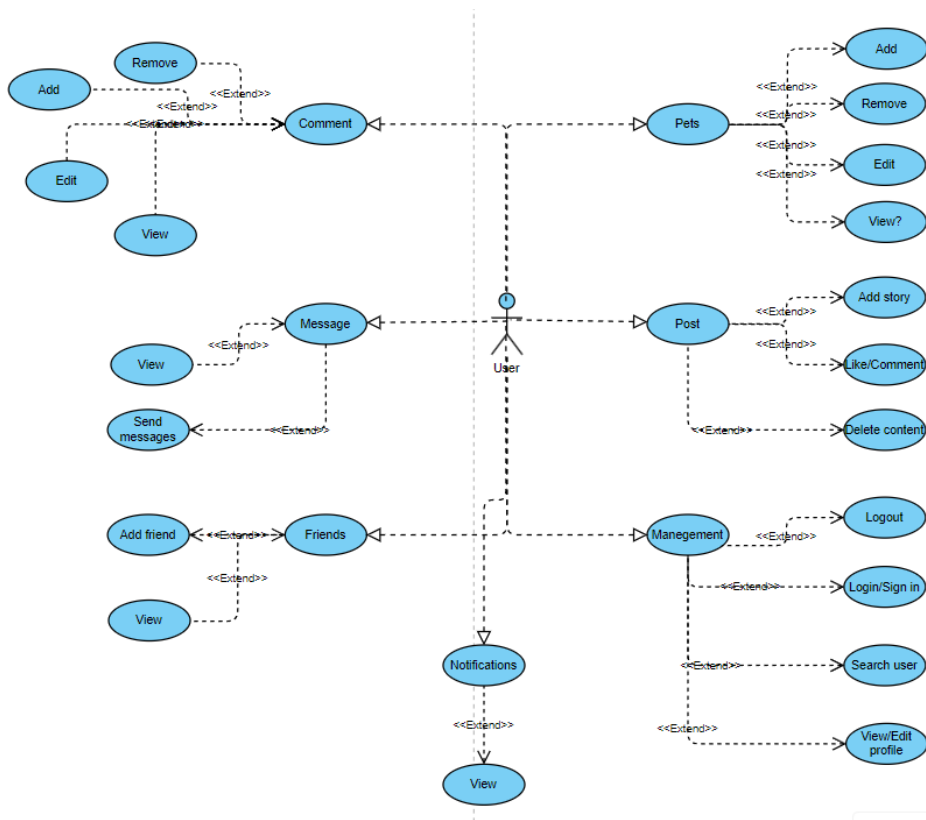
FeedView- Această clasă implementează o vizualizare pentru afișarea unui feed de postări, inclusiv interacțiuni cu utilizatorii, pet-urile și comentariile asociate.

MessageMediator- conține clasele de implementare pentru funcționalitatea de mesaje cu ajutorul design pattern-ului mediator

Composite- conține clasele de implementare pentru funcționalitatea de vizualizare a profilelor altor useri cu ajutorul design pattern-ului composite

Patterns- conține clasele de implementare pentru vizualizarea paginii de feed cu ajutorul design pattern-urilor observer si strategy

3. Diagrama de use case



4. Diagrame

Diagrama de activitate logare in cont:

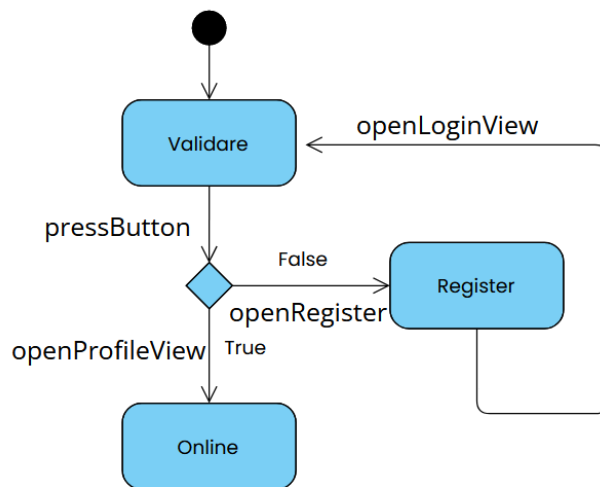


Diagrama de comunicare mesaje private:

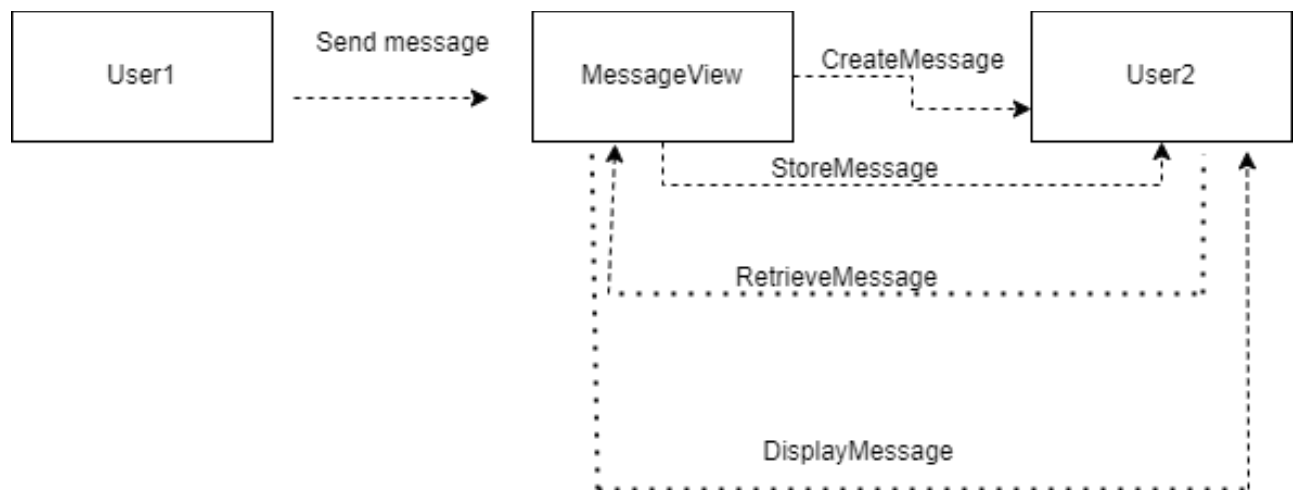


Diagrama de secventa adauga postare:

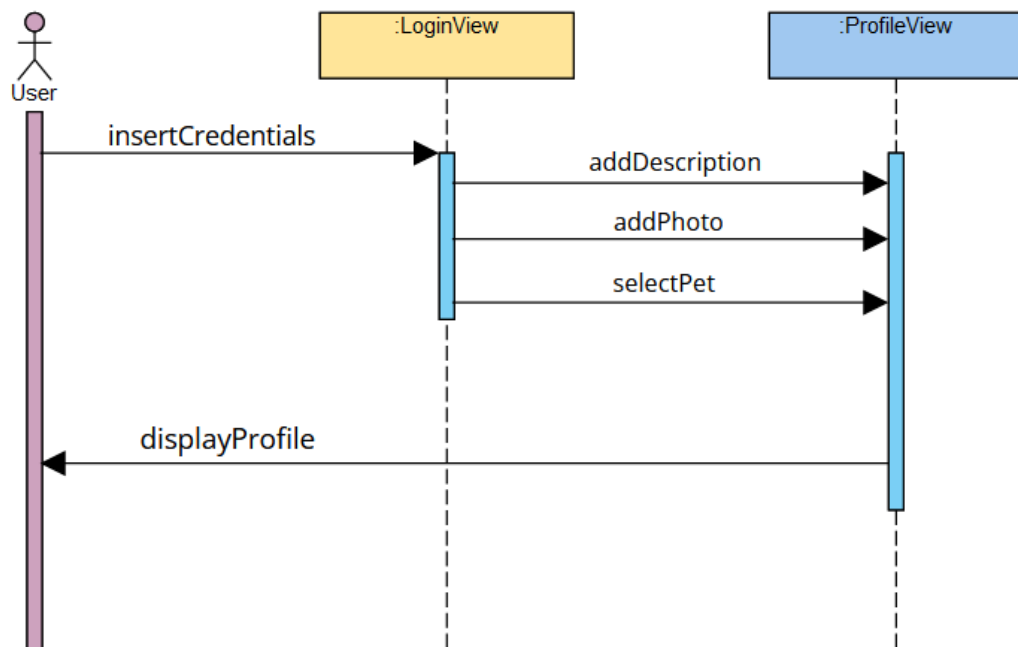
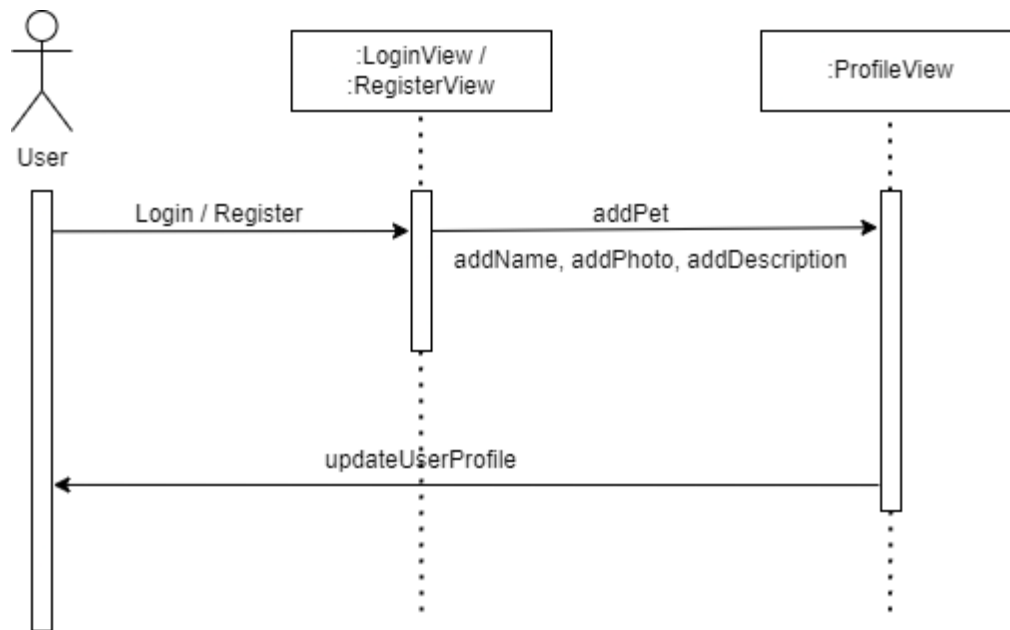
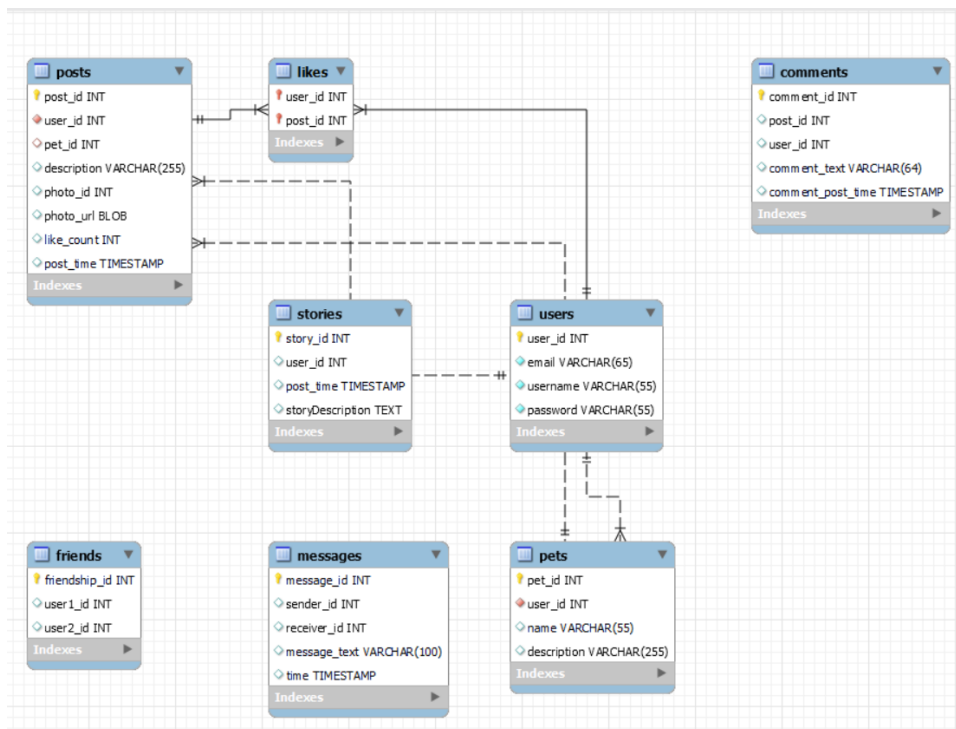


Diagrama de secventa adauga animal de companie:



5. Diagrama bazei de date cu tabele generate de sql



6. Design Pattern

În proiectul PetGram, am implementat design pattern-ul Strategy pentru gestionarea afișării imaginilor în postări în pagina de feed. Această abordare oferă flexibilitate și ușurință în schimbarea modului în care imaginile sunt afișate fără a modifica structura principală a codului. Am creat o interfață `ImageDisplayStrategy` care definește metoda `displayImage(Posts post)`. Cu strategiile concrete `ExistingImageDisplayStrategy` care implementează `ImageDisplayStrategy` și se ocupă de afișarea imaginilor existente în post și `MissingImageDisplayStrategy` care implementează `ImageDisplayStrategy` pentru situația în care postul nu conține o imagine.

Design pattern-ul ales de mine este Observer, pe care l-am implementat pentru a gestiona notificările din aplicație. Acest pattern activează notificările în momentele cheie ale interacțiunii utilizatorilor cu aplicația, în special când aceștia dau "like" sau comentează la o postare. Structura patternului în aplicația mea se prezintă astfel: am definit o interfață `Observer`, care include metoda `update`. Aceasta metodă este cheia patternului, fiind folosită pentru a primi notificări de la subiect. Clasa `NotificationsView` implementează interfața `Observer` și, prin urmare, funcționează ca un observator concret. Când metoda `update` este invocată, `NotificationsView` procesează textul notificării, actualizând interfața utilizatorului corespunzător.

În acest proiect, am implementat design pattern-ul mediator pentru funcționalitatea de mesaje între utilizatori.

Interfața și clasa din pachetul `MediatorMessages` - `ChatMediator` și `ChatMediatorImpl` - lucrează împreună cu clasa `MessageView` pentru a implementa funcționalitatea de mesaje.

Interfața ChatMediator definește un set de metode pentru a gestiona comunicarea între utilizatori. Metodele includ trimiterea de mesaje (sendMessage), adăugarea de utilizatori (addUser), și obținerea tuturor utilizatorilor (getAllUsers).

Clasa ChatMediatorImpl: implementează ChatMediator și furnizează logica de bază pentru gestionarea mesajelor între utilizatori. În cadrul metodei sendMessage, adaugă un nou mesaj în baza de date folosind obiectul MessagesDAO. În metoda addUser, adaugă un utilizator la lista de utilizatori. În metoda getAllUsers, obține toți utilizatorii din baza de date folosind UsersDAO.

Clasa MessageView: folosește o instanță a ChatMediatorImpl pentru a facilita comunicarea între utilizatori. În metoda sendMessage, utilizează ChatMediator pentru a trimite mesajul și actualizează interfața pentru afișarea mesajelor.

Prin intermediul acestor clase, MessageView poate interacționa cu ChatMediator pentru a gestiona și afișa mesajele între utilizatori.

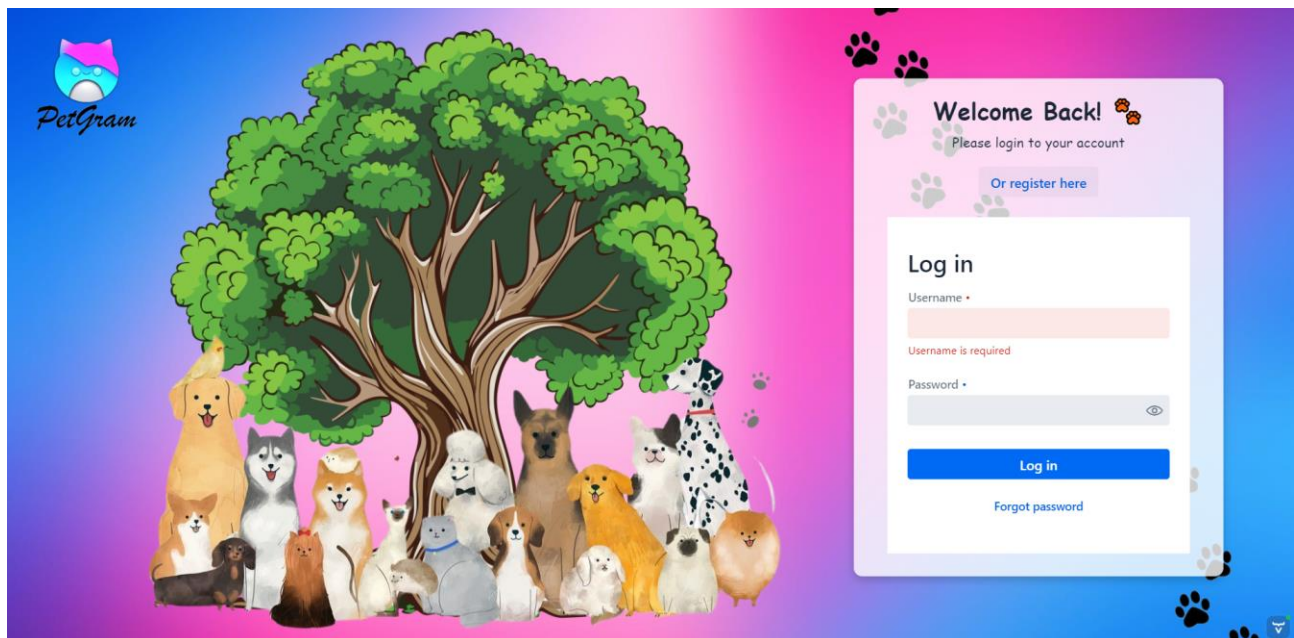
Design pattern-ul Composite este folosit eficient în implementarea unui profil de utilizator, permițând organizarea și gestionarea diferitelor componente ale profilului într-o structură ierarhică și unitară. În acest cadru, profilul unui utilizator poate fi considerat un 'composite', incluzând diverse 'sub-componente' sau 'leaf objects', cum ar fi detalii personale, animale, postări și alte elemente caracteristice. Fiecare dintre aceste sub-componente poate avea, la rândul ei, propriile sub-componente, formând astfel o structură de tip arbore. Pattern-ul Composite facilitează managementul acestor componente, oferind posibilitatea de a le trata în mod uniform, fie că sunt obiecte simple sau complexe. De exemplu, când se actualizează profilul, modificările pot fi aplicate atât componentelor individuale, cât și celor compuse, folosind aceeași interfață sau metodă, ceea ce simplifică logica de programare și mentenanța codului.

7. Manual de utilizare

Pentru a putea folosi aplicatie sunt necesare: pentru programatorul care a proiectat interfata trebuie sa aiba acces la IntelliJ, programul unde s-a scris codul in java, la MySQL Workbench care contine baza de date cu tabelele necesare aplicatiei si un motor de cautare, precum google chrome. Aplicatia se ruleaza din IntelliJ, iar site-ul apare in motorul de cautare. Utilizatorul are nevoie doar de browser pentru a utiliza aplicatia. Se poate loga sau inregistra daca nu are inca cont, iar apoi poate posta sau interactiona cu ceilalti utilizatori.

Interfata aplicatiei de social media:

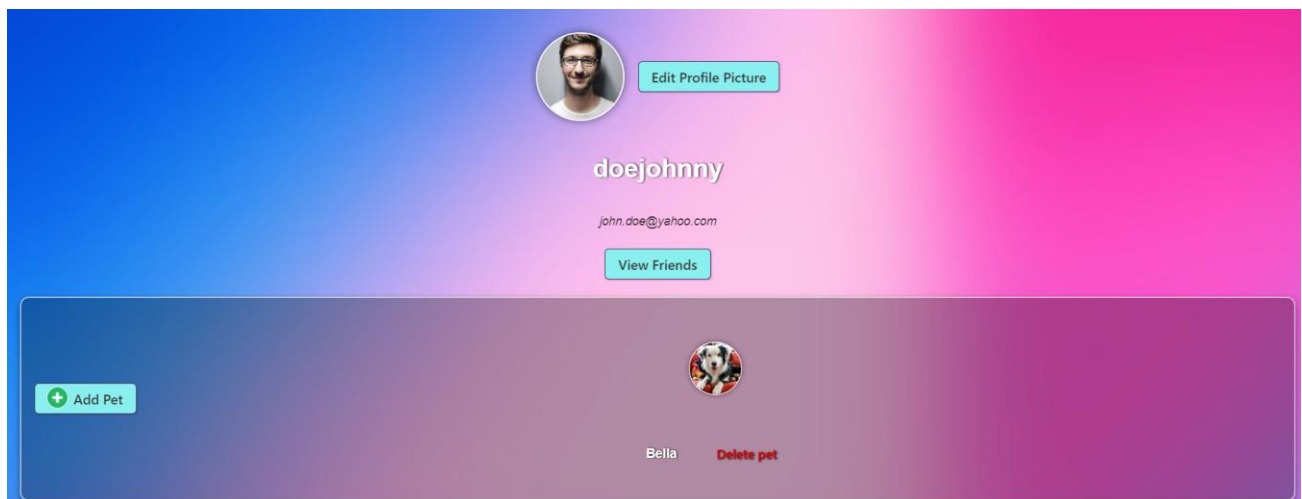
Pagina de logare:



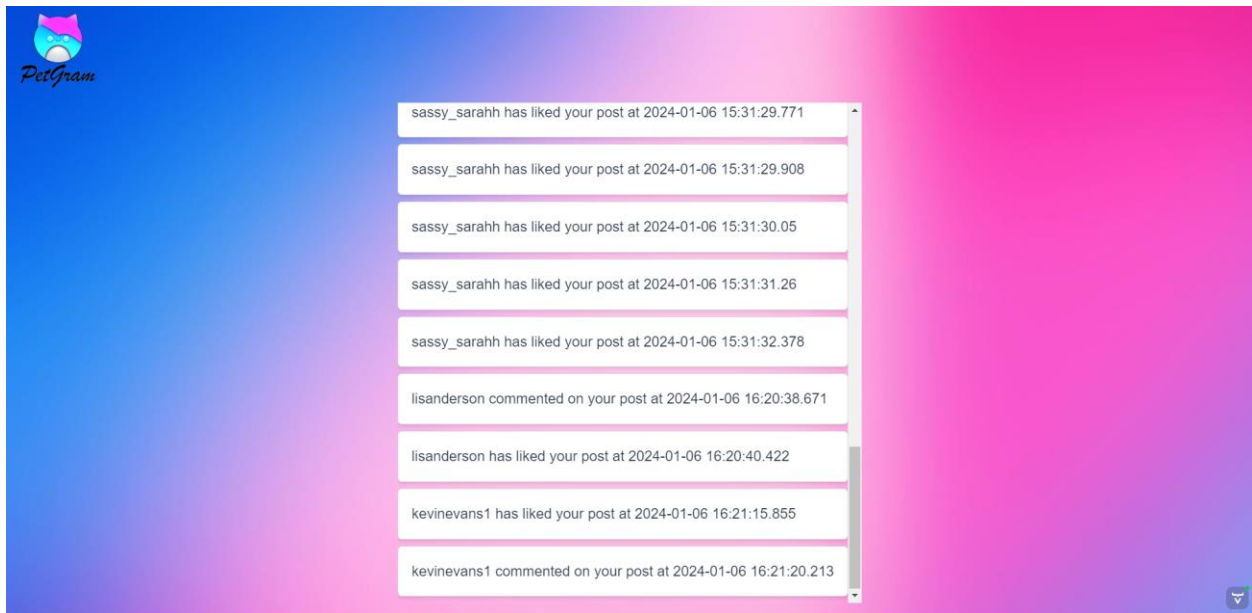
Pagina pentru inregistrare:



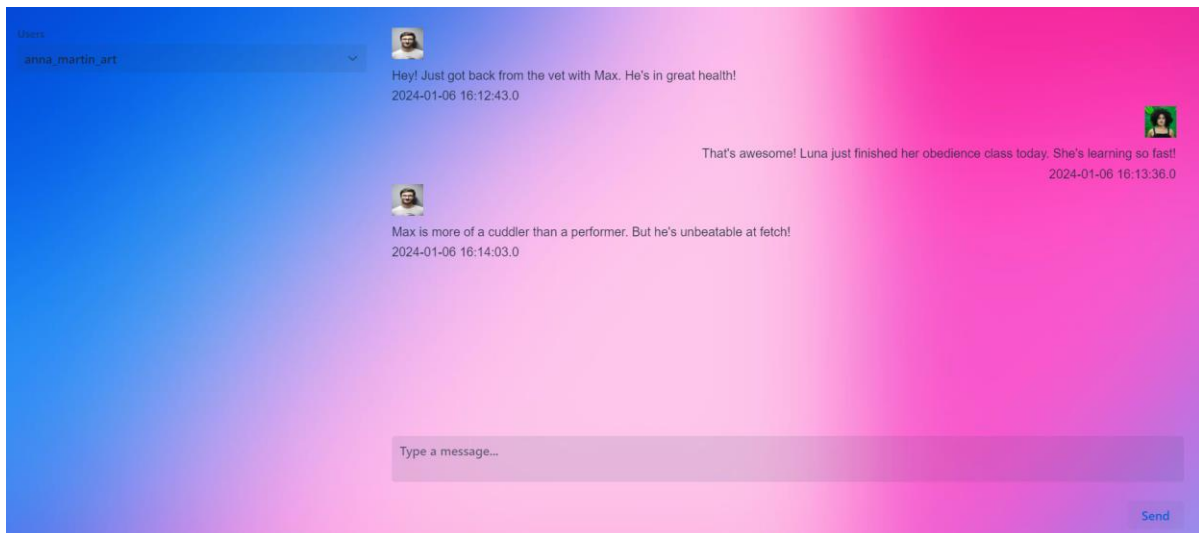
Pagina pentru profilul utilizatorului:



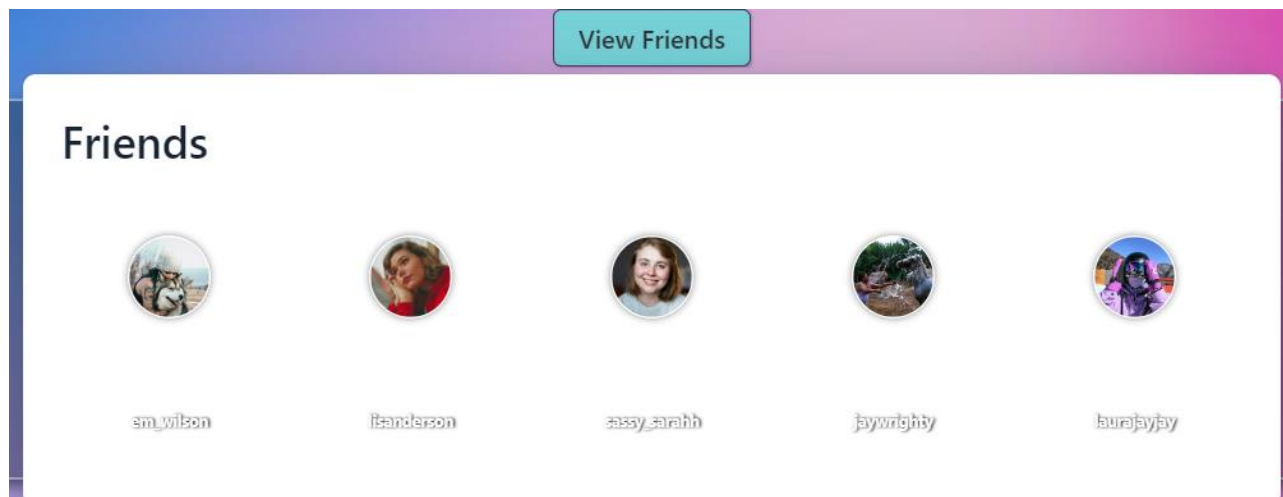
Pagina pentru notificari:



Pagina pentru mesage private:



Lista de prieteni pentru un utilizator:



Utilizatorul poate sa isi schimbe poza de profil si sa adauge animale de companie:



Edit Profile Picture

Change Profile Picture

Upload File...



Drop file here



profilepic.png



Save

Add New Pet

Pet Name

Billy

Pet Description

Turtle

Upload File...



Drop file here




pet2.png



Save

Un animal de companie are propriul mini-profil:


Pet Details




Name: Bella


Description: Graceful and friendly, loves belly rubs and long walks in the park. 🐶

Utilizatorul poate sa posteze o poza, etichetand un animal de companie din lista acestuia:


duojohnny

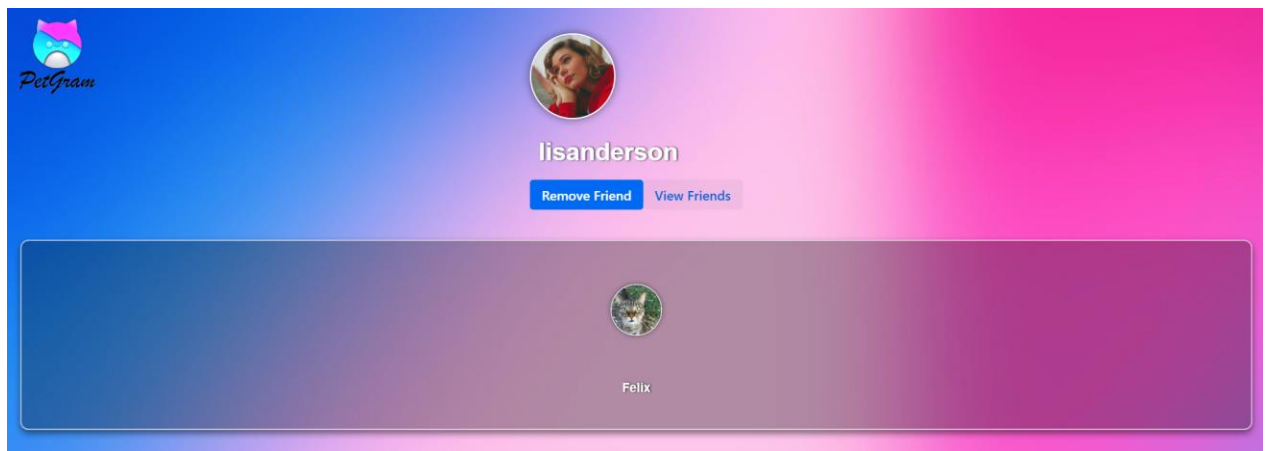

Bella

Just realized my dog's toy collection has officially taken over the living room. It's a small sacrifice for his endless joy, right? 🐶

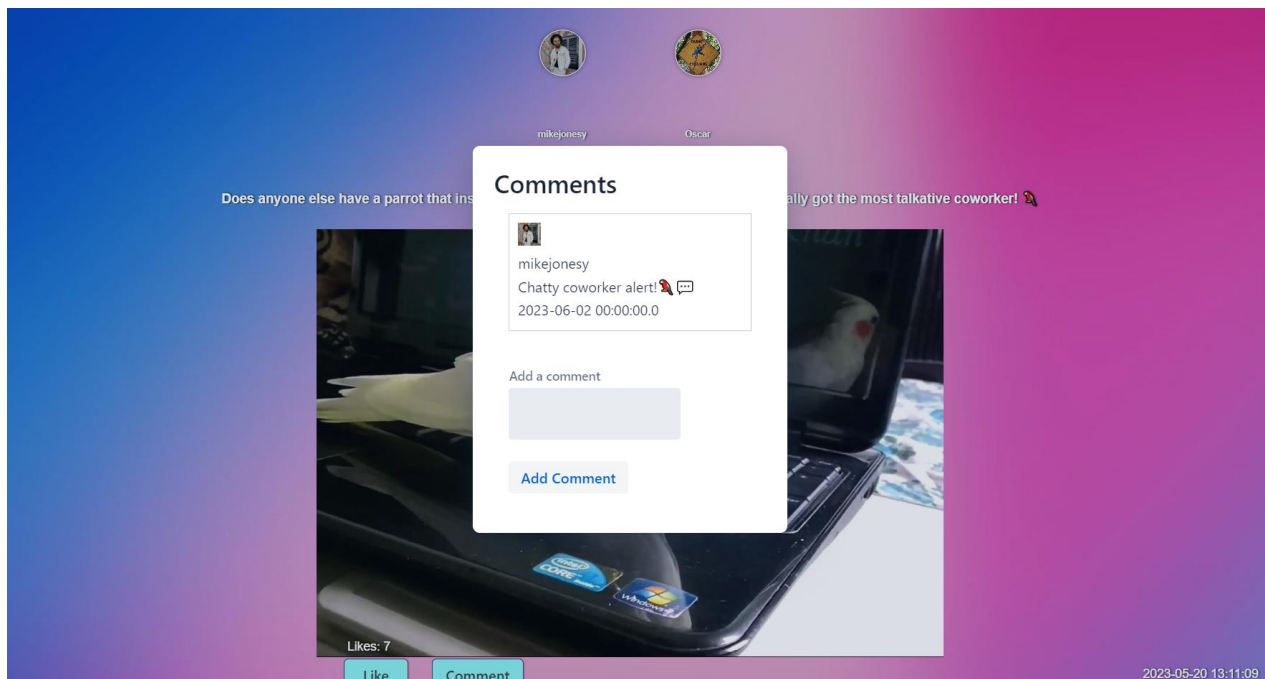


Likes: 20

Utilizatorul poate sa adauge sau sa stearga un user din lista de prieteni:



Utilizatorul poate sa dea like sau sa scrie un comentariu la o postare:



Utilizatorul poate sa adauge un story pe profilul acestuia, care se va sterge automat dupa 24 de ore:

Add your story

Story Description

My first story

Upload File...

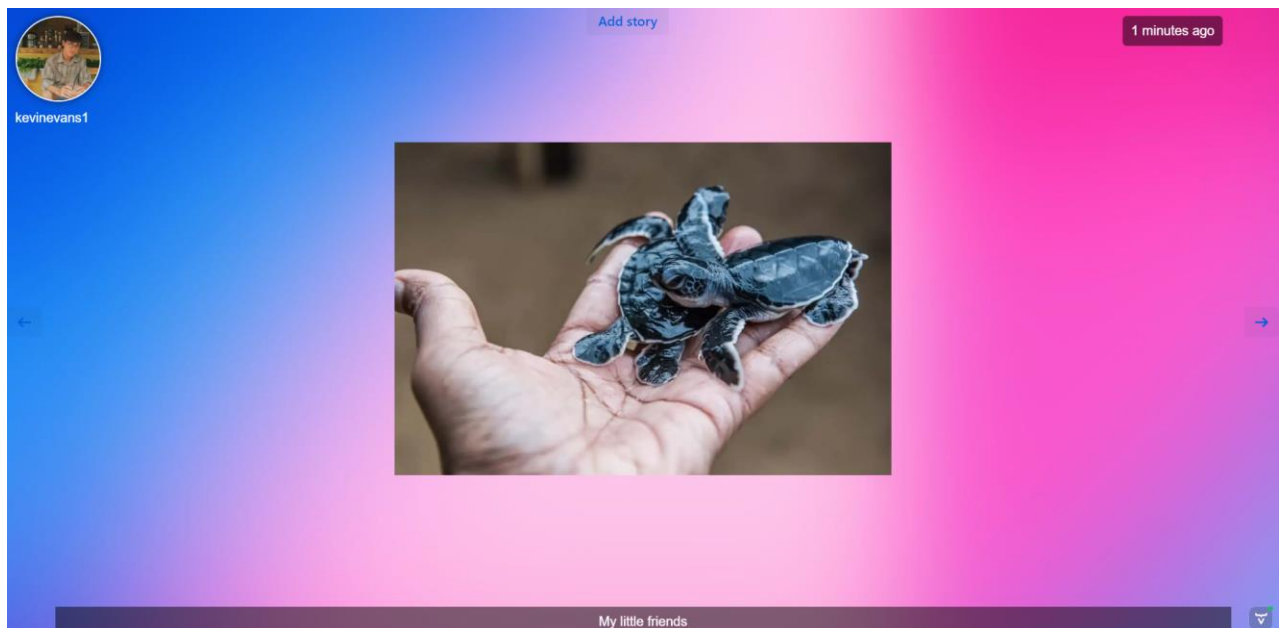


Drop file here

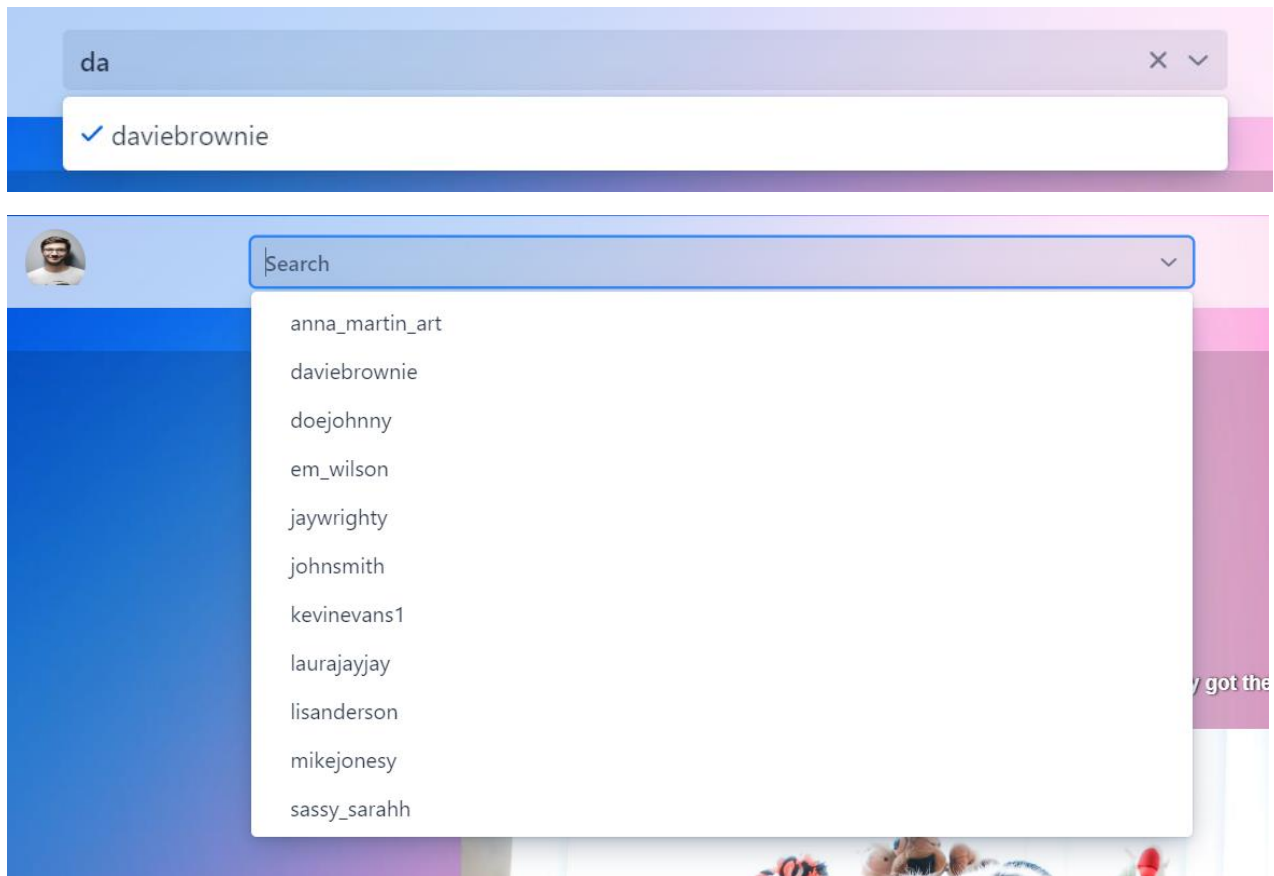
✓ test.png



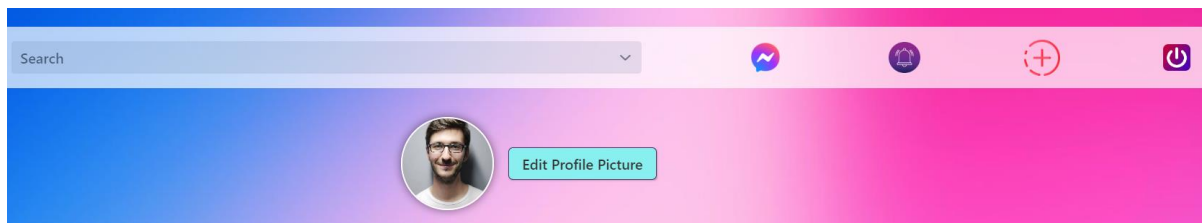
Save



Utilizatorul poate sa caute profilul altor utilizatori:



Bara de cautare si meniul aplicatiei:



8. Tutorial de utilizare

Aplicatia este implementata astfel incat sa fie foarte usor de utilizat pentru orice utilizator.

Ceea ce se intampla in spate: se ruleaza programul din IntelliJ avand MySQL deschis, acolo generandu-se tabelele. Utilizatorul trebuie doar

sa deschida un browser, sa caute si sa intre la adresa <http://localhost:8080>. Se va deschide pagina de logare reprezentata mai sus. Daca are deja cont, utilizatorul poate sa intre in cont cu emailul si cu parola, iar daca nu are se poate inregistra, sa-si creeze un cont apasand pe “Register here”, unde trebuie sa isi introduca numele, prenumele, emailul si parola.

Cand utilizatorul a intrat in cont, va apare pagina cu feed-ul. Acesta poate vedea, odata intrat in cont, o bara de cautare si un meniu deasupra, de unde poate sa aleaga unde sa navigheze. Poate sa isi vada propriul profil, sa caute profilul altor utilizatori, sa trimita mesaje private unui prieten, sa isi vada notificarile, sa adauge un story, si sa se deconecteze din cont. De asemenea, utilizatorul poate naviga prin feed si sa vada postarile prietenilor lui, sa dea like si comentarii la acestea. Pe profilul propriu, utilizatorul poate sa adauge un animal de companie si sa posteze o imagine etichetand un animal la alegere.

9. Bibliografie

<https://vaadin.com/docs/latest/components>

<https://spring.io/projects/spring-framework/>

<https://refactoring.guru/design-patterns>

<https://www.site.uottawa.ca/school/research/lloseng/supportMaterial/>