Projektni zadatak iz predmeta Programiranje mobilnih aplikacija

Projektni zadatak predstavlja aplikaciju koja omogućava korisnicima, koji su registrovani u okviru sistema, da samostalno kreiraju pitanja iz različitih oblasti i nivoa obrazovanja i da odgovaraju na postojeća pitanja. Omogućava lako ponvaljanje gradiva, efikasan i zabavan način učenja.

Funkcije sistema

Registracija i prijavljivanje na sistem

Neprijavljenom korisniku se prikazuje stranica za prijavu na sistem. Stranica sadrži polja za unos email adrese i lozinke i link za prelazak na stranicu za registraciju novog korisnika.

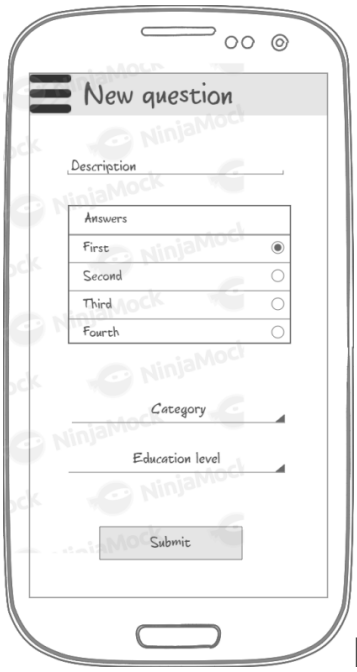
Registracija korisnika obuhvata unos imena, prezimena, email adrese i lozinke. Email adresa predstavlja korisničko ime i potrebno je da bude jedinstvena.

Prikaz osnovne stranice

Nakon prijave na sistem, prikazuje se osnovna stranica koja sadrži pregled pitanja svih korisnika, dugme za kreiranje pitanja i meni koji omogućava prelazak na stranicu za:

- filtriranje po kategorijama,
- filtriranje po nivou obrazovanja,
- filtriranje po lokaciji,
- arhiva urađenih pitanja i
- prikaz korisničkog profila.

Klikom na jedno pitanje, prikazuje se stranica koja sadrži pregled odabranog pitanja i tu se pitanje korisnik može da reši pitanje (više o tome u nastavku teksta).



Kreiranje pitanja

Postupak kreiranja pitanja:

- Korisnik unosi pitanje.
- Korisnik unosi četiri odgovora i označava koji je tačan.
- Korisnik bira kategoriju pitanja.
- Korisnik bira nivo obrazovanja kome pripada pitanje.

Prilikom kreiranja svakog pitanja, svi korisnici koju su "pretplaćeni" dobijaju notifikaciju da je kreirano novo pitanje.

Kategorija i nivo obrazovanja pitanja su predefinisani u sistemu.

Primeri kategorija:

- matematika,
- engleski,
- fizika,
- hemija…

Primeri nivoa obrazovanja:

- osnovna škola,
- srednja škola,
- doktorske studije,
- drugo…

Arhiva urađenih pitanja

Korisniku se prikazuju sva pitanja na koja je odgovorio.

Filtriranje

Korisnik ima mogućnost da odabre filter, nakon čega će mu na osnovnoj stranici biti prikazna samo ona pitanja koja odgovaraju zadatom kriterijumu. Mogućnost filtriranja po:

- kategoriji,
- nivou obrazovanja i
- lokaciji.

Prikaz koriničkog profila

Korisnik ima mogućnost da ažurira lične podatke.

Rešavanje pitanja

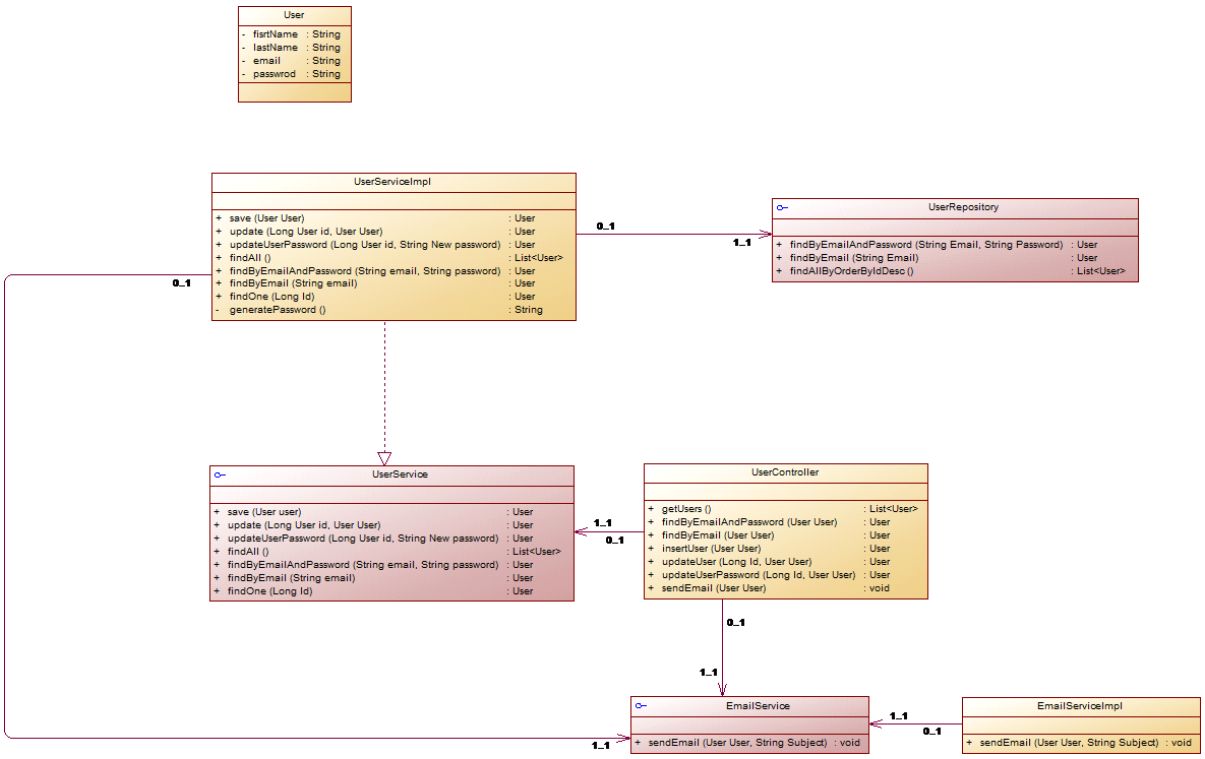
Korisnik ima mogućnost da odgovara na postojeća pitanja, nakon čega se prikazuje rezultat i mogućnost komentarisanja.

Web servisi (back-end i db)

Backend aplikacije će biti napisan korišćenjem Java + Spring. Spring će biti iskorišćen da se napiše REST preko kojeg će aplikacija komunicirati i razmenjivati podatke sa bazom podataka. Podaci će biti čuvani u Mongo bazi podataka, jer je brza, efikasna i omogućava pretraživanje po lokaciji.

Model podataka

Na slici 1 je predstavljen User class dijagram.



Slika 1 - User class

User - sadrži:

- String firstName,
- String lastName,
- String email,
- String password

UserRepository:

- User findByEmailAndPassword(String email, String password);
- User findByEmail(String email);
- List<User>findAllByOrderByIdDesc();

UserService:

- User save(User user);
- User update(Long userId, User user);
- User updateUserPassword(Long userId, String newPassword);

- List< User > findAll();
- User findByEmailAndPassword(String email, String password);
- User findByEmail(String email);
- User findOne(Long id);

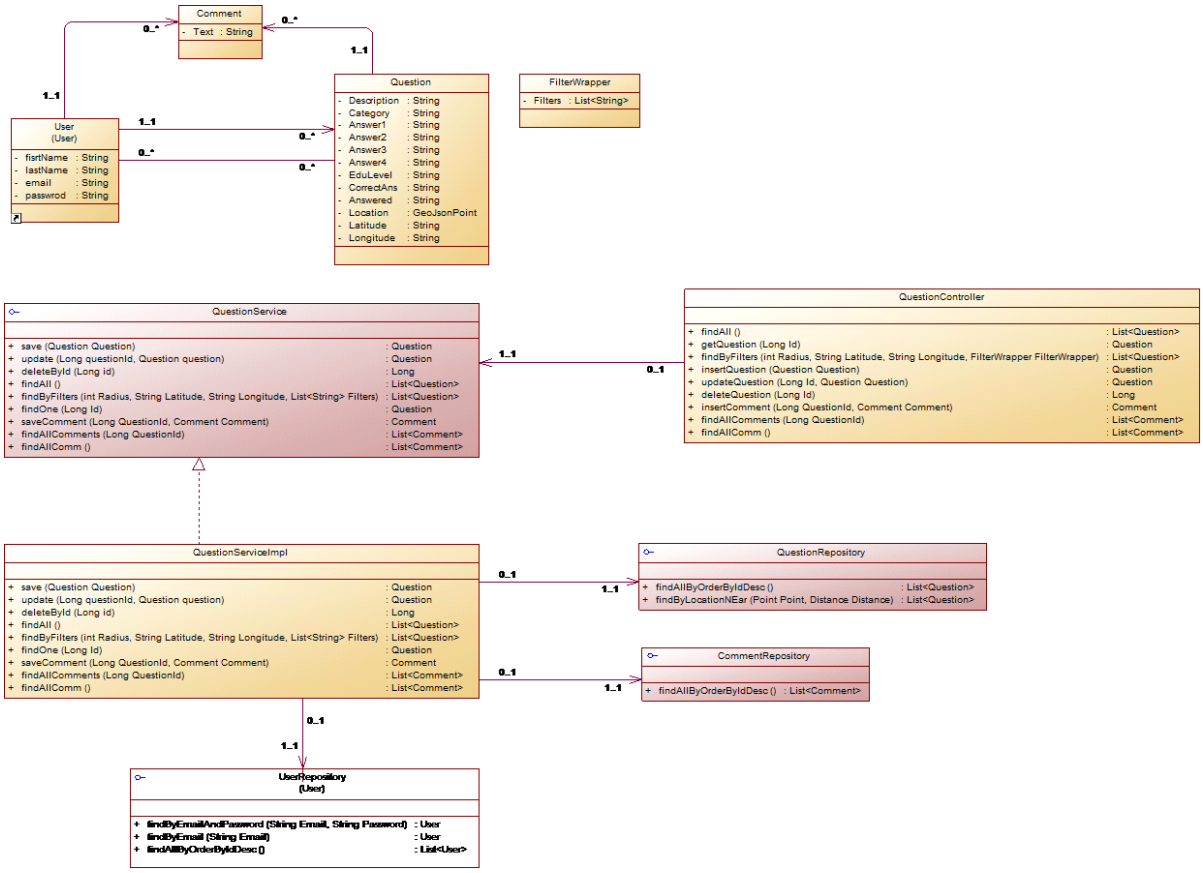
EmailService:

- void sendEmail(User user, String subject)

UserController:

- List<User> getUsers()
- User findByEmailAndPassword(User user)
- User findByEmail(User user)
- User insertUser(User user)
- User updateUser(Long id,User user)
- User updateUserPassword(Long id, User user)
- Void sendEmail(User user)

Na slici 2 je prikazan Question class dijagram.



Slika 2 - Question class

Question - sadrži:

- String description,
- String category,
- String answer1,
- String answer2,
- String answer3,

- String answer4,
- String eduLevel,
- String correctAns,
- String answered,
- String latitude,
- String longitude,
- GeoJsonPoint location

QuestionRepository:

- List<Question> findAllByOrderByIdDesc();
- List<Question> findByLocationNear(Point point, Distance distance);

QuestionService:

- Question save(Question question);
- Question update(Long questionId, Question question);
- Long deleteById(Long id);
- List<Question> findAll();
- List<Question> findByFilters(int radius, String latitude, String longitude, List<String> filters);
- Question findOne(Long id);
- Comment saveComment(Long questionId, Comment comment);
- List<Comment> findAllComments(Long questionId);
- List<Comment> findAllComm();

QuestionController:

- List<Question> findAll()
- Question getQuestion(Long id)
- List<Question> findByFilters(int radius, String latitude, String longitude, FilterWrapper filters)

- Question insertQuestion(Question question)
- Question updateQuestion(Long id, Question question)
- Long deleteQuestion(Long id)
- Comment insertComment(Long questionId, Comment comment) {
- List<Comment> findAllComments(Long questionId)
- List<Comment> findAllComm()

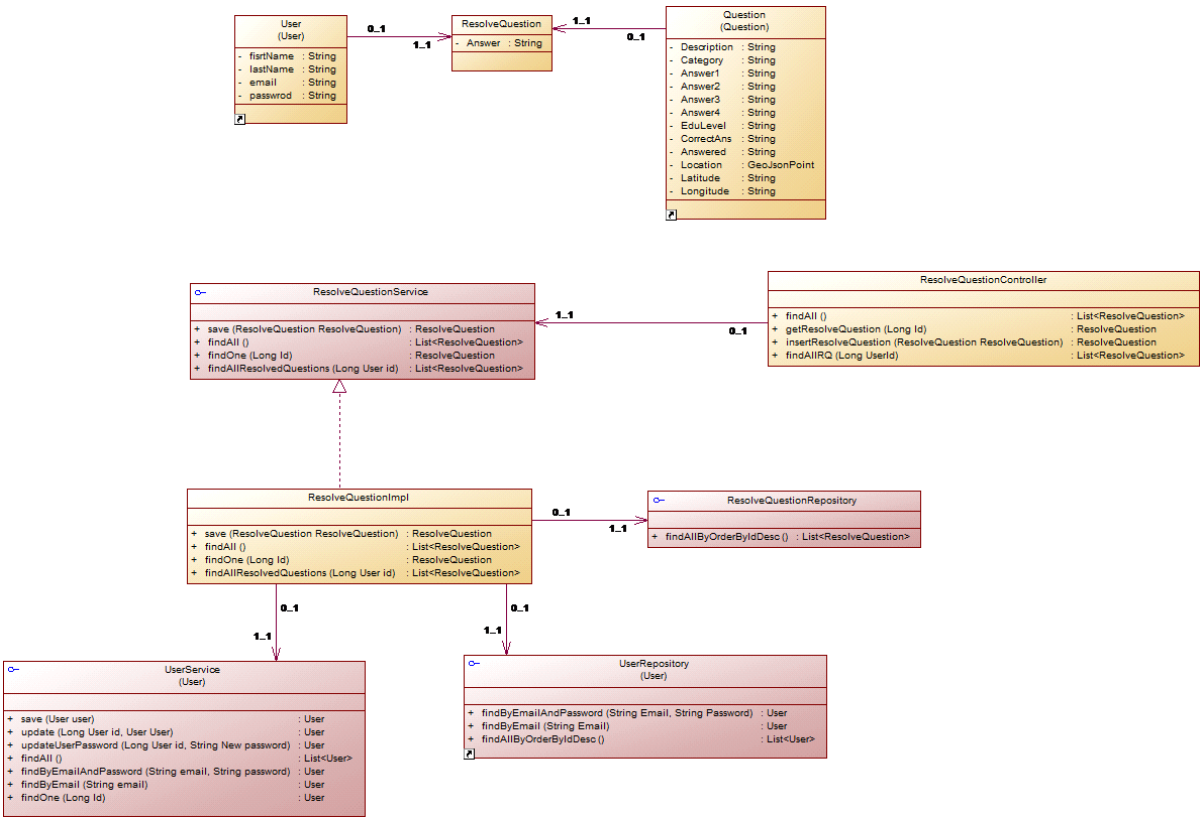
Comment:

- String text

CommentRepository:

- List<Comment> findAllByOrderByIdDesc()

Na slici 3 je prikazan ResolvedQuestion class dijagram.



ResolveQuestion:

- String answer
- String correctAns
- String questionText

ResolveQuestionRepository:

- List<ResolveQuestion> findAllByOrderByIdDesc()

ResolveQuestionService:

- ResolveQuestion save(ResolveQuestion question);
- List<ResolveQuestion> findAll();
- ResolveQuestion findOne(Long id);
- List<ResolveQuestion> findAllResolvedQuestions(Long userId);

ResolveQuestionController:

- List<ResolveQuestion>> findAll()
- ResolveQuestion> getResolveQuestion(Long id)
- ResolveQuestion insertResolveQuestion(ResolveQuestion question)
- List<ResolveQuestion> findAllRQ(Long userId)

Tim 16:
Svetlana Đurić E239/2017
Milena Lalić E236/2017
Danilo Zeković R121/2017