

Fakultet organizacionih nauka

Internet tehnologije

Projektni rad – Društvena mreža 1

Photogramy

Mentor:

Dušan Barać

Studenti:

Nikola Đorđević 124/17

Mirko Zečić 49/17

Beograd, jul 2021.

Sadržaj

1. Korisnički zahtev	4
2. Specifikacija slučajeva korišćenja.....	5
SK1: Slučaj korišćenja – Kreiranje korisničkog naloga:	5
SK2: Slučaj korišćenja - Prijavljivanje korisnika na sistem pomoću Google naloga	6
SK3: Slučaj korišćenja - Prijavljivanje korisnika na sistem.....	7
SK4: Slučaj korišćenja – Kreiranje uspomene	8
SK5: Slučaj korišćenja – Pretraživanje uspomene	9
SK6: Slučaj korišćenja – Izmena uspomene	10
SK7: Slučaj korišćenja – Brisanje uspomene	11
SK8: Slučaj korišćenja – Lajkovanje uspomene	12
SK9: Slučaj korišćenja – Lajkovanje uspomene	13
SK10: Slučaj korišćenja – Zapрати/Otprати prijatelja	14
3. Arhitektura aplikacije	15
4. Ponašanje softverskog sistema – Sistemski dijagram sekvenci	16
DS1: Kreiranje korisničkog naloga.....	16
DS2: Prijavljivanje na sistem	18
DS3: Kreiranje uspomene	20
DS4: Pretraživanje pacijenta	22
5. Specifikacija REST API-ja.....	25
fetchPost(id).....	25
fetchPosts()	26
fetchPostsBySearch(searchQuery).....	27
createPost(newPost).....	29
likePost(id)	30
commentPost(value, id).....	31
updatePost(id, updatedPost)	32
deletePost(id).....	33
signIn(userInfo)	34
signUp(userInfo)	35
follow(nas_id, id)	36
getUsers().....	37

getFollowers(id)	38
6. Model podataka	40
6.1 Struktura baze podataka (Konceptualni model)	41
7. Tehnologije korišćene u aplikaciji	44
8. Korisničko uputstvo	45
9. Delovi koda	49
10. Github repozitorijum	51

1. Korisnički zahtev

Photogramy je društvena mreža na kojoj se možete registrovati, bilo preko google naloga bilo preko našeg sistema i nakon toga deliti svoje uspomene sa svojim prijateljima. Možete ostavljati lajkove, komentare na objavama, zapratiti vaše prijatelje da biste videli njihove objave ili otpratiti ukoliko ne želite više da gledate njihove uspomene. Kada kliknete na određenu uspomenu naš sistem će ispod da Vam preporuči još neke uspomene koje će Vam se možda svideti pa ćete i njih moći da pogledate.

2. Specifikacija slučajeva korišćenja

SK1: Slučaj korišćenja – Kreiranje korisničkog naloga:

Naziv SK:

Kreiranje korisničkog naloga

Aktor:

Korisnik

Učesnici:

Korisnik, sistem

Preduslovi:

Sistem je uključen i prikazuje formu za rad sa korisničkim nalogom.

Osnovni scenario SK:

1. Korisnik unosi podatke o korisniku. (APUSO)
2. Korisnik kontroliše da li je korektno uneo podatke o korisniku. (ANSO)
3. Korisnik poziva sistem da zapamti podatke o korisniku. (APSO)
4. Sistem pamti podatke o korisniku. (SO)

Alternativni scenario:

- 4.1. Ukoliko podaci koje je korisnik uneo nisu potpuni i sistem ne može da kreira novi korisnički nalog, sistem prikazuje korisniku poruku: „Sistem ne može da sačuva podatke o korisniku!“. (IA)

SK2: Slučaj korišćenja - Prijavljivanje korisnika na sistem pomoću Google naloga

Naziv SK:

Prijavljivanje korisnika na sistem pomoću Google naloga

Aktor:

Korisnik

Učesnici:

Korisnik, sistem

Preduslovi:

Sistem je uključen i prikazuje formu za rad sa korisničkim nalogom. Korisnik nije prijavljen na sistem.

Osnovni scenario SK:

1. Korisnik poziva sistem da prijavi korisnika. (APSO)
2. Sistem prijavljuje korisnika na sistem. (SO)

Alternativni scenario:

2.1. Ukoliko podaci koje je korisnik uneo nisu potpuni ili sistem ne može da pronađe podudaranje sa korisnikom, sistem prikazuje korisniku poruku: „Neuspešno prijavljivanje na sistem!“. (IA)

SK3: Slučaj korišćenja - Prijavljivanje korisnika na sistem

Naziv SK:

Prijavljivanje korisnika na sistem

Aktor:

Korisnik

Učesnici:

Korisnik, sistem

Preduslovi:

Sistem je uključen i prikazuje formu za rad sa korisničkim nalogom. Korisnik nije prijavljen na sistem.

Osnovni scenario SK:

1. Korisnik unosi podatke o korisniku. (APUSO)
2. Korisnik kontroliše da li je korektno uneo podatke o korisniku. (ANSO)
3. Korisnik poziva sistem da prijavi korisnika. (APSO)
4. Sistem prijavljuje korisnika na sistem. (SO)

Alternativni scenario:

- 4.1. Ukoliko podaci koje je korisnik uneo nisu potpuni ili sistem ne može da pronađe podudaranje sa korisnikom, sistem prikazuje korisniku poruku: „Neuspešno prijavljivanje na sistem!“. (IA)

SK4: Slučaj korišćenja – Kreiranje uspomene

Naziv SK

Kreiranje uspomene

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa uspomenom.

Osnovni scenario SK

1. Korisnik unosi podatke uspomene. (APUSO)
2. Korisnik kontroliše da li je korektno uneo podatke uspomene. (ANSO)
3. Korisnik poziva sistem da zapamti podatke o uspomeni. (APSO)
4. Sistem pamti podatke o uspomeni. (SO)
5. Sistem prikazuje korisniku poruku: "Sistem je zapamtio uspomenu". (IA)

Alternativna scenarija

- 5.1 Ukoliko sistem ne može da zapamti podatke o uspomeni on prikazuje korisniku poruku: "Sistem ne može da zapamti uspomenu". (IA)

SK5: Slučaj korišćenja – Pretraživanje uspomene

Naziv SK

Pretraživanje uspomene

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa uspomenom.

Osnovni scenario SK

1. Korisnik unosi vrednost po kojoj pretražuje uspomenu. (APUSO)
2. Korisnik kontroliše da li je ispravno uneo podatke. (ANSO)
3. Korisnik pritiskom tipke poziva sistem da pronađe uspomenu. (APSO)
4. Sistem traži uspomene po zadatoj vrednosti. (SO)
5. Sistem prikazuje korisniku podatke o uspomeni. (IA)

Alternativna scenarija:

4.1. Ukoliko sistem ne može da pronađe uspomenu, on korisniku prikazuje poruku: „Sistem ne može da pronađe uspomenu!“. (IA)

SK6: Slučaj korišćenja – Izmena uspomene

Naziv SK

Izmena uspomene

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa pacijentom.

Osnovni scenario SK

1. Korisnik unosi (menja) podatke o uspomeni. (APUSO)
2. Korisnik kontroliše da li je korektno uneo podatke o uspomeni. (ANSO)
3. Korisnik poziva sistem da zapamti podatke o uspomeni. (APSO)
4. Sistem pamti podatke o uspomeni. (SO)
5. Sistem prikazuje korisniku poruku: "Sistem je zapamtio uspomenu." (IA)

Alternativna scenarija:

- 5.1 Ukoliko sistem ne može da zapamti podatke o uspomeni on prikazuje korisniku poruku: "Sistem ne može da zapamti uspomenu". (IA)

SK7: Slučaj korišćenja – Brisanje uspomene

Naziv SK

Brisanje uspomene

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa uspomenu. Učitana je lista uspomena.

Osnovni scenario SK

1. Korisnik poziva sistem da obriše uspomenu. (APSO)
2. Sistem briše uspomenu. (SO)
3. Sistem prikazuje korisniku poruku: "Sistem je obrisao uspomenu." (IA)

Alternativna scenarija

- 3.1 Ukoliko sistem ne može da obriše uspomenu on prikazuje korisniku poruku "Sistem ne može da obriše uspomenu".(IA)

SK8: Slučaj korišćenja – Lajkovanje uspomene

Naziv SK

Lajkovanje uspomene

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa uspomenom. Učitana je lista uspomena.

Osnovni scenario SK

1. Korisnik poziva sistem da lajkuje uspomenu. (APSO)
2. Sistem lajkuje uspomenu. (SO)
3. Sistem postavlja vizuelni efekat da je uspomena lajkovana. (SO)

Alternativna scenarija

- 2.1 Ukoliko sistem ne može da lajkuje uspomenu on prikazuje korisniku poruku "Sistem ne može da obriše uspomenu".(IA)

SK9: Slučaj korišćenja – Lajkovanje uspomene

Naziv SK

Lajkovanje uspomene

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa uspomenom. Učitana je lista uspomena.

Osnovni scenario SK

1. Korisnik poziva sistem da lajkuje uspomenu. (APSO)
2. Sistem lajkuje uspomenu. (SO)
3. Sistem postavlja vizuelni efekat da je uspomena lajkovana. (SO)

Alternativna scenarija

- 2.1 Ukoliko sistem ne može da lajkuje uspomenu on prikazuje korisniku poruku "Sistem ne može da lajkuje uspomenu".(IA)

SK10: Slučaj korišćenja – Zapрати/Otprати prijatelja

Naziv SK

Zapрати/Otprати prijatelja

Aktori SK

Korisnik

Učesnici SK

Korisnik i sistem (program)

Preduslov: Sistem je uključen i korisnik je ulogovan pod svojom šifrom. Sistem prikazuje formu za rad sa uspomenom. Učitana je lista korisnika.

Osnovni scenario SK

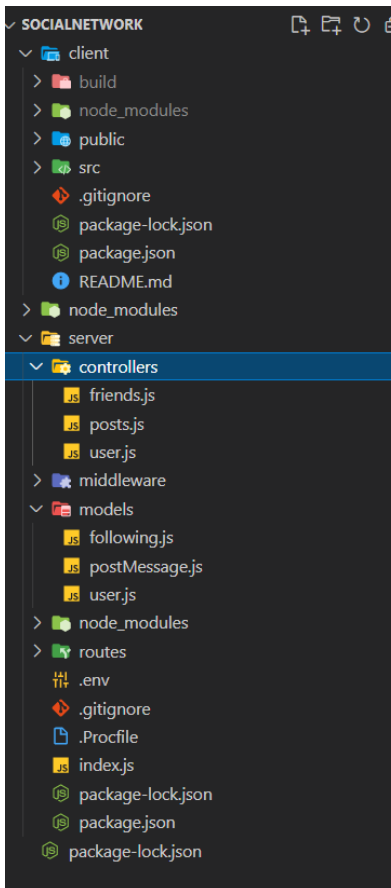
1. Korisnik poziva sistem da **zapрати/otprати** prijatelja. (APSO)
2. Sistem **prати/prestaje da prати** prijatelja. (SO)
3. Sistem postavlja vizuelni efekat da je prijatelj **zapaćen/otpraćen**. (SO)

Alternativna scenarija

- 2.1 Ukoliko sistem ne može da zapрати/otprati prijatelja on prikazuje korisniku poruku "Sistem ne može da zapрати/otprati prijatelja ".(IA)

3. Arhitektura aplikacije

Projekat je urađen na MVC (Model – View - Controller) arhitekturi. Sam klijentski deo koda, koji je rađen u React.js predstavlja View deo u slagalici. Controller se nalazi na serverskoj strani, on obrađuje zahteve i pravi upite na bazu podataka. Model se nalazi na serverskoj strani i to su klase i objekti koje smo koristili. Za povezivanje frontend i backend dela korišćena je REST arhitektura.



Slika 1 – Struktura projekta

```
const PORT = process.env.PORT || 5000;

mongoose.connect(process.env.CONNECTION_URL, { useNewUrlParser: true, useUnifiedTopology: true
  .then(() => app.listen(PORT, () => console.log(`Server running on port: ${PORT}`)))
  .catch((error) => console.log(error.message));

mongoose.set('useFindAndModify', false);
```

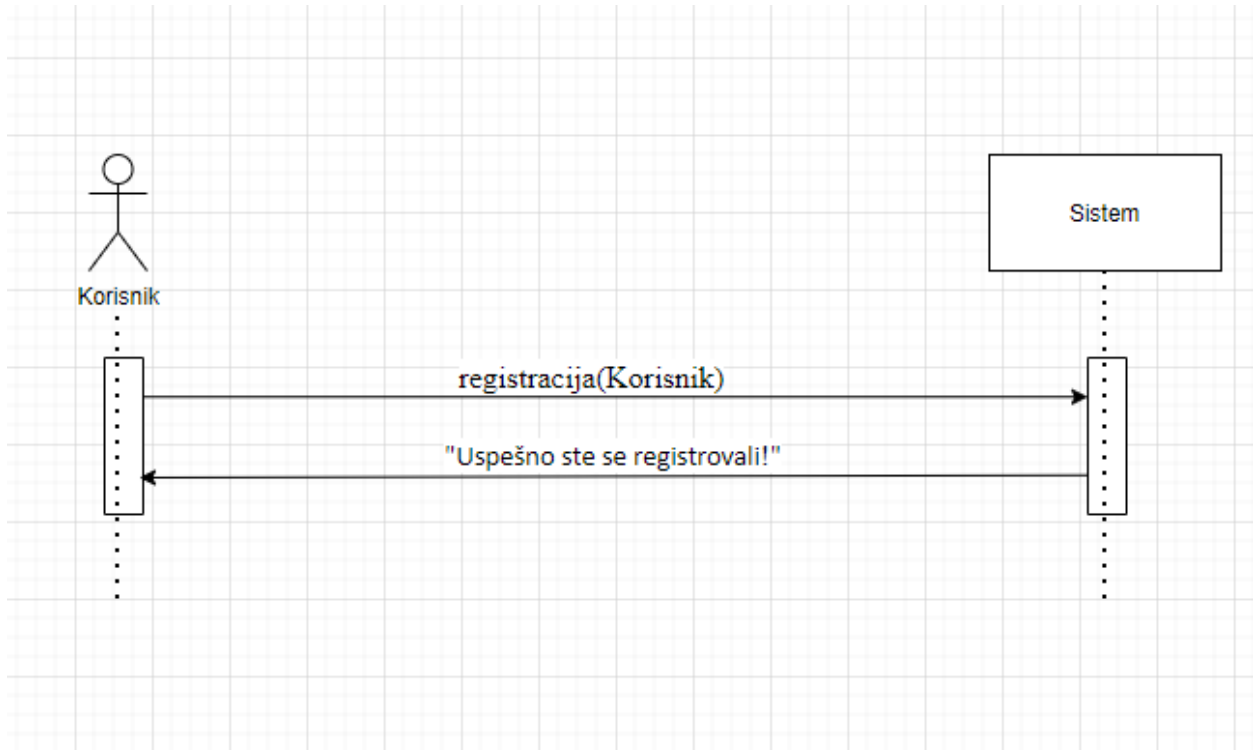
Slika 2 – Povezivanje sa bazom podataka

4. Ponašanje softverskog sistema – Sistemski dijagram sekvenci

DS1: Kreiranje korisničkog naloga

Osnovni scenario SK:

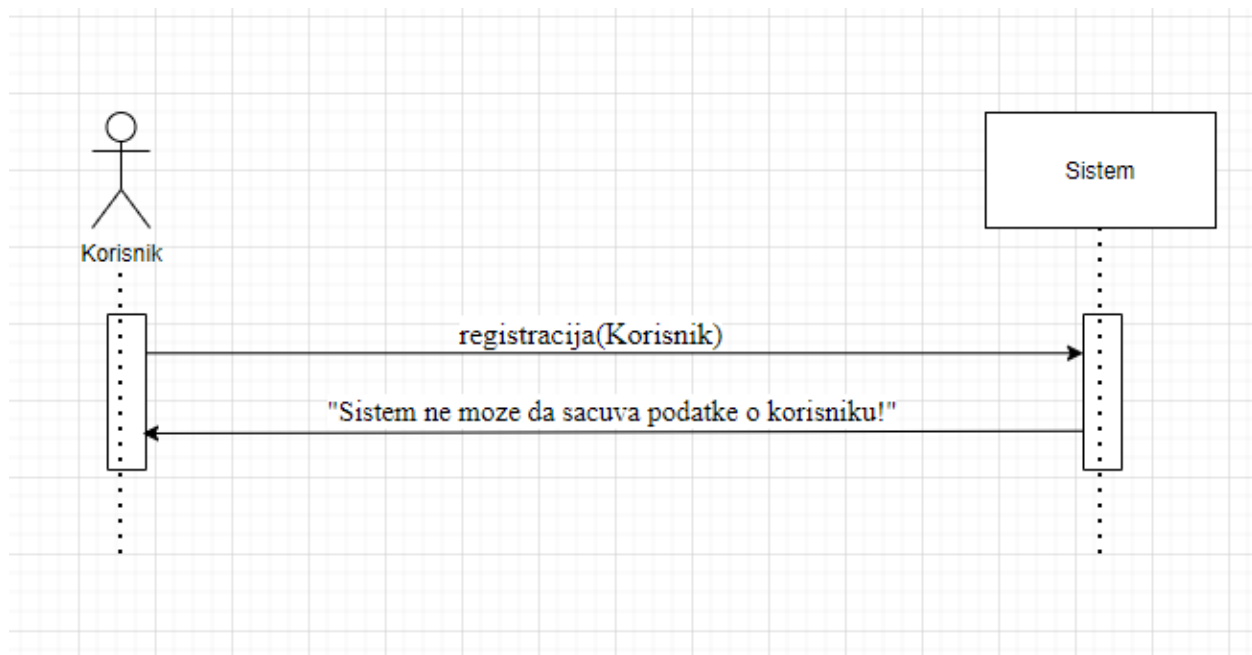
1. Korisnik poziva sistem da zapamti podatke o korisniku. (APSO)
2. Sistem prikazuje korisniku poruku: „Uspešno ste se registrovali!“. (IA)



Slika 3 – DS1 Osnovni scenario

Alternativni scenario

- 2.1. Ukoliko podaci koje je korisnik uneo nisu potpuni i sistem ne može da kreira novi korisnički nalog, sistem prikazuje korisniku poruku: „Sistem ne može da sačuva podatke o korisniku!“. (IA)

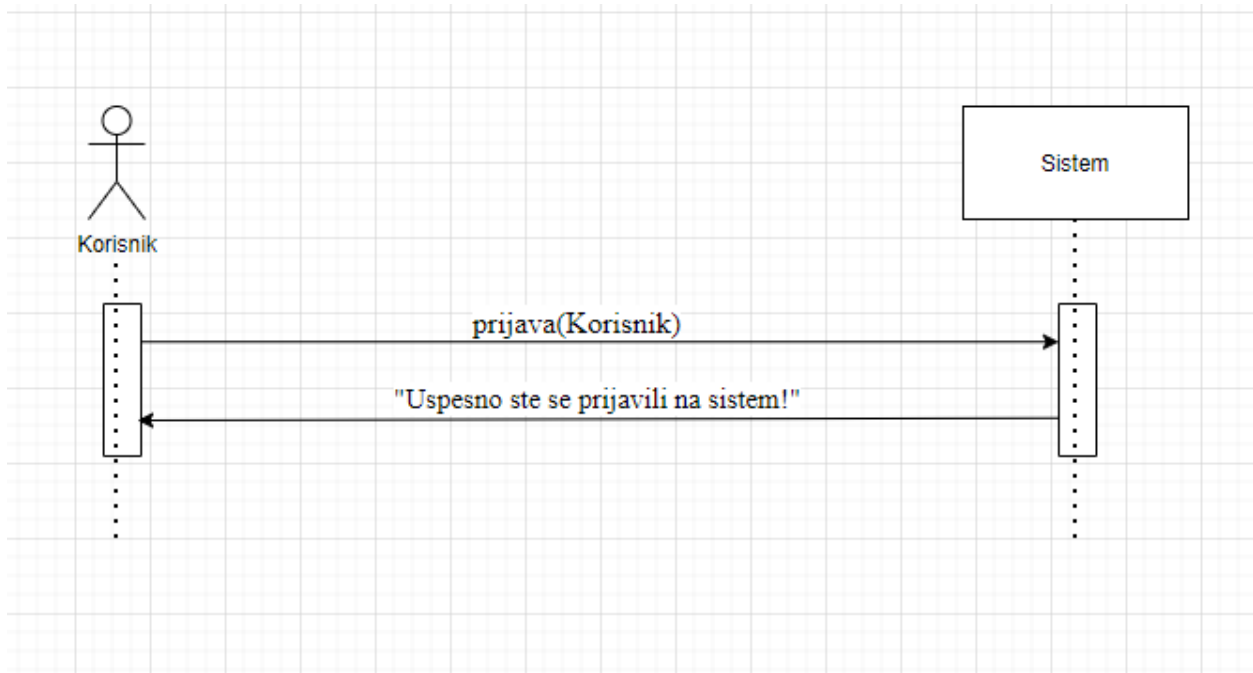


Slika 4 – DS1 Alternativni scenario

DS2: Prijavljanje na sistem

Osnovni scenario SK:

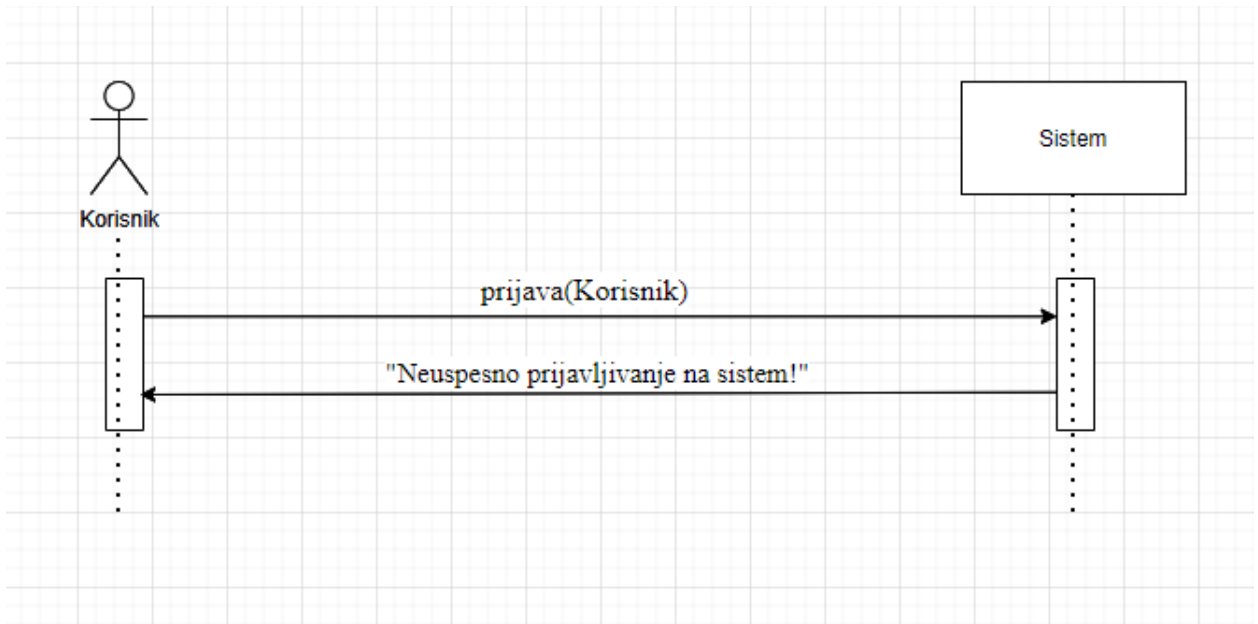
1. Korisnik poziva sistem da prijavi korisnika. (APSO)
2. Sistem prikazuje korisniku poruku: „Uspesno ste se prijavili na sistem!“. (IA)



Slika 5 - DS2 – Osnovni scenario

Alternativni scenario

- 2.1. Ukoliko podaci koje je korisnik uneo nisu potpuni ili sistem ne može da pronađe podudaranje sa korisnikom, sistem prikazuje korisniku poruku: „Neuspešno prijavljivanje na sistem!“. (IA)

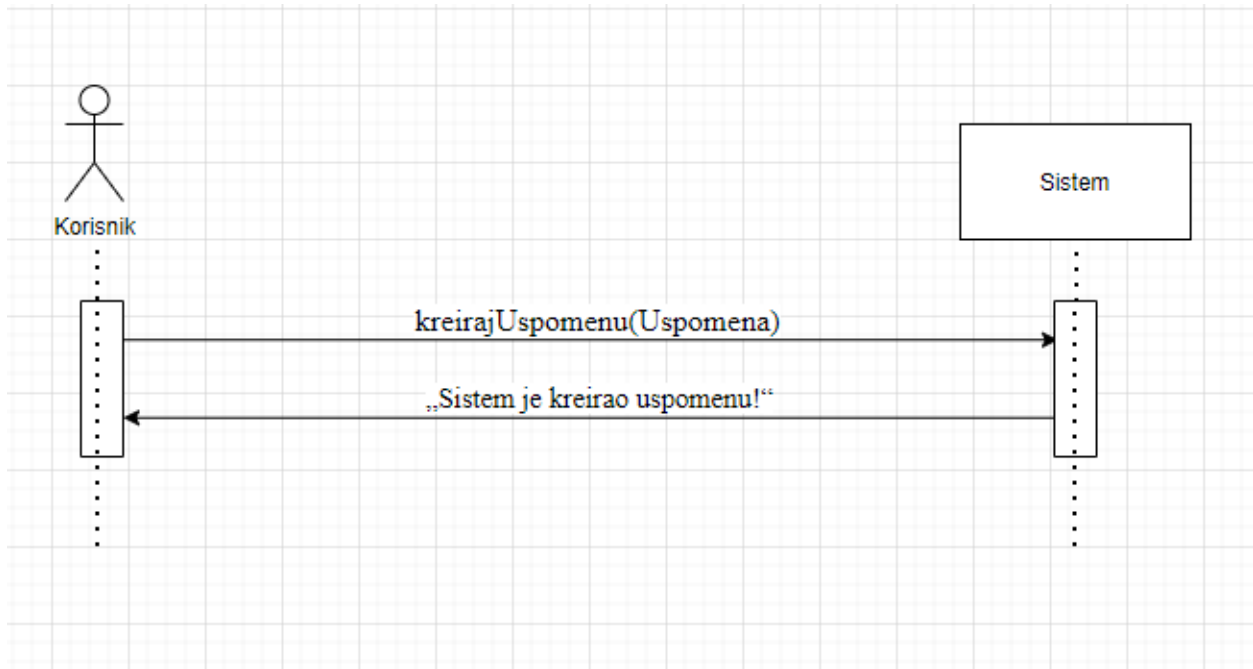


Slika 6 – DS2 Osnovni scenario

DS3: Kreiranje uspomene

Osnovni scenario SK:

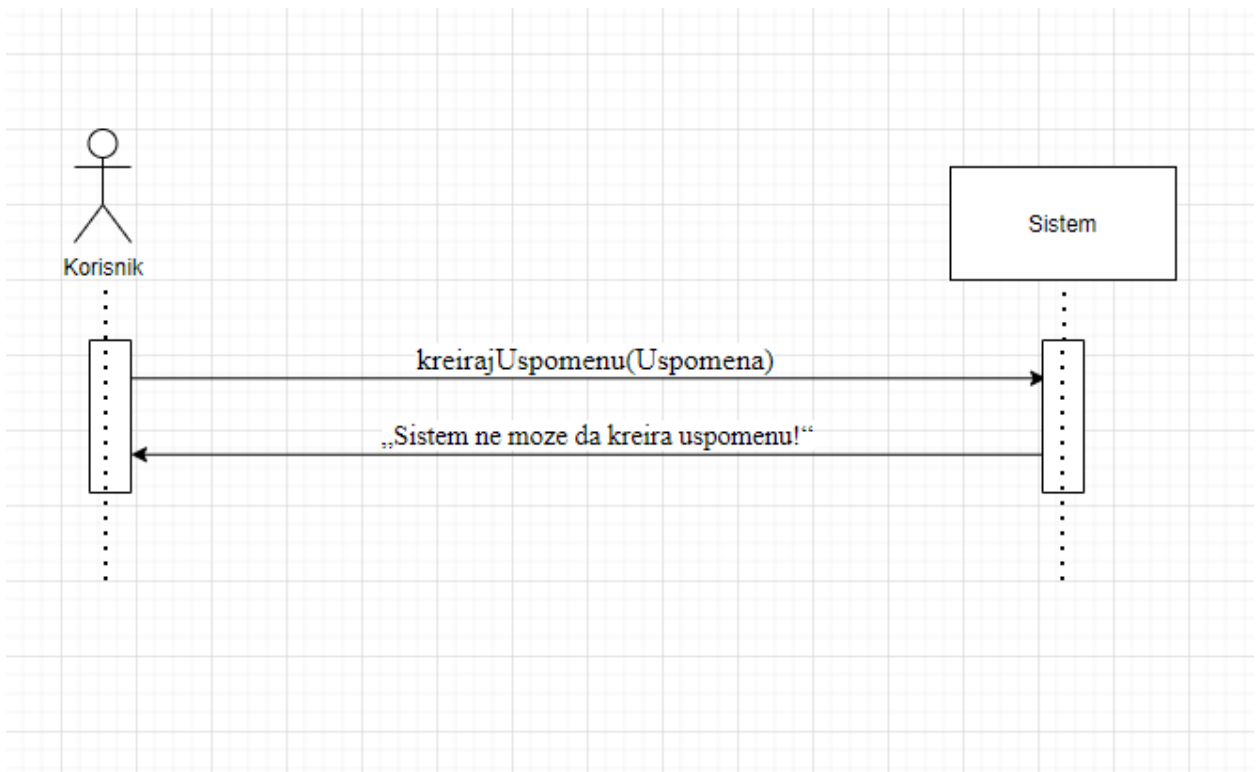
1. Korisnik poziva sistem da zapamti podatke o uspomeni. (APSO)
2. Sistem prikazuje korisniku poruku: „Sistem je zapamtio uspomenu!“. (IA)



Slika 7 – DS3 Osnovni scenario

Alternativna scenarija

- 2.1. Ukoliko sistem ne može da zapamati uspomenu, on korisniku prikazuje poruku: „Sistem ne može da kreira uspomenu!“. (IA)

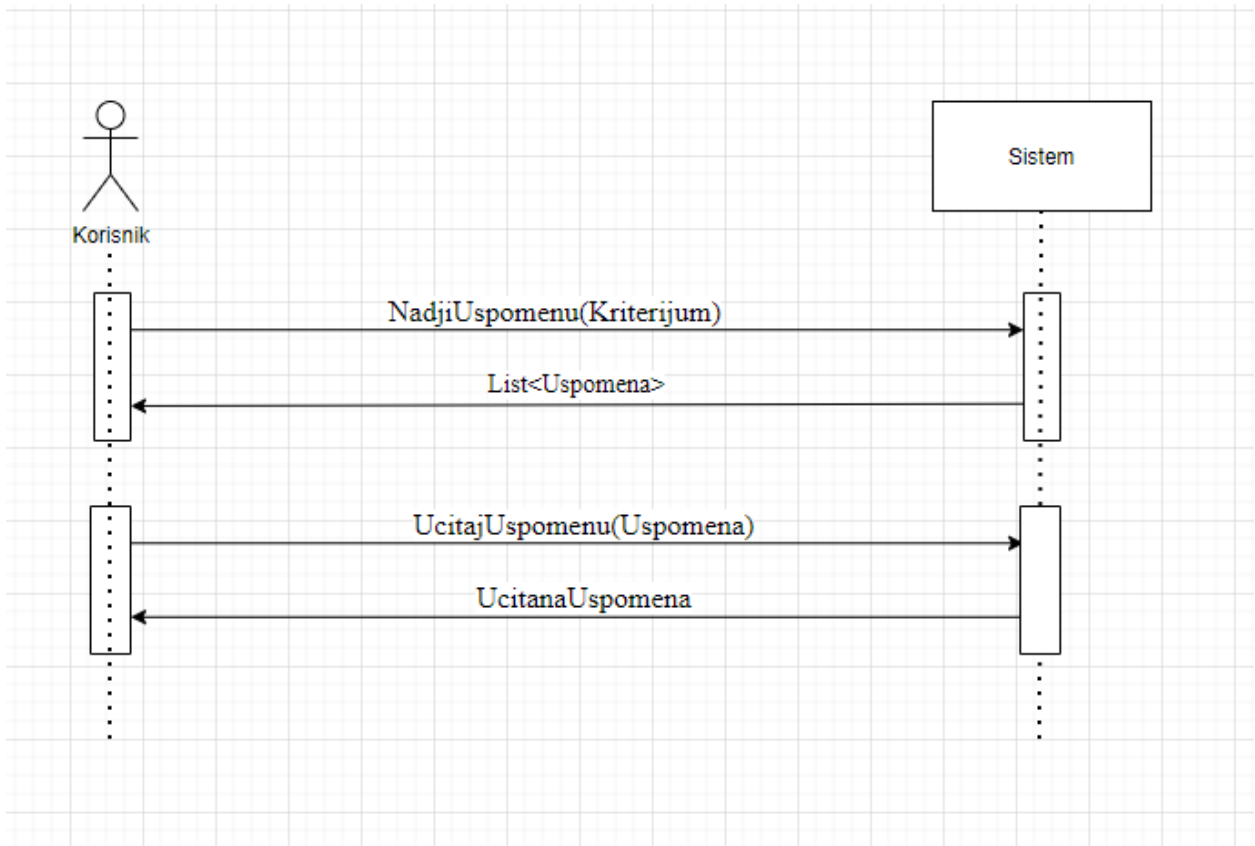


Slika 8 – DS3 Alternativni scenario

DS4: Pretraživanje pacijenta

Osnovni scenario SK:

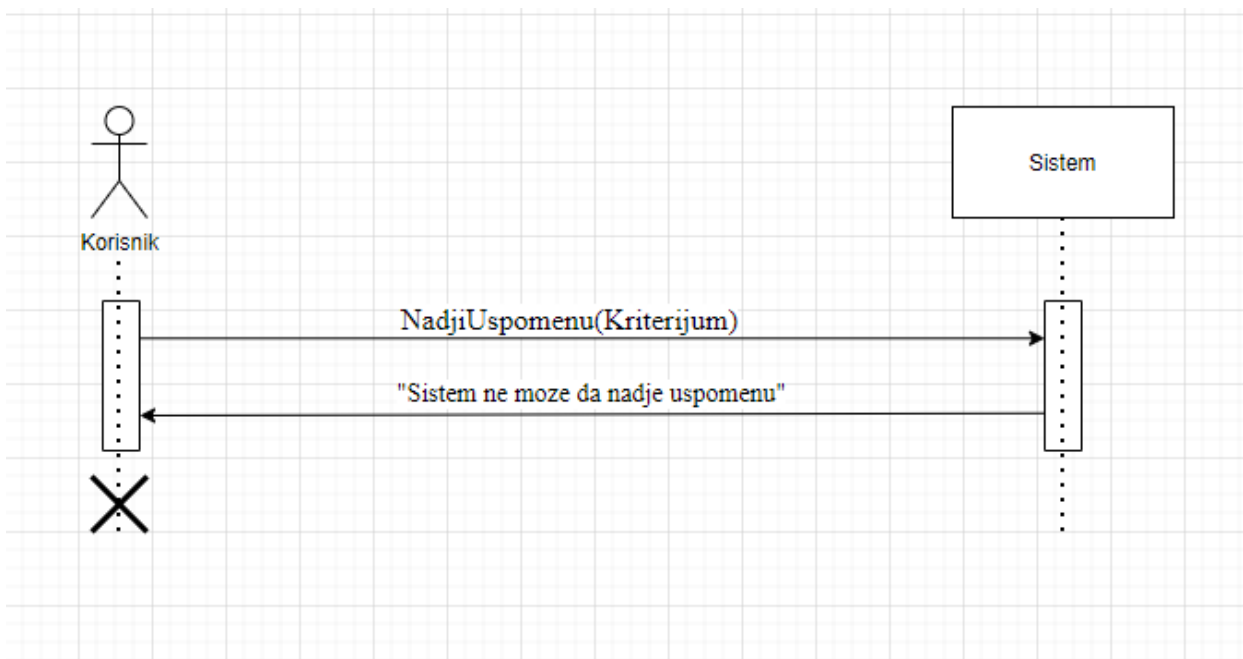
1. Korisnik unosom vrednosti putem tipke poziva sistem da pronade uspomenu. (APSO)
2. Sistem prikazuje korisniku podatke o uspomeni. (IA)
3. Korisnik poziva sistem da učitaj uspomenu.(APSO)
4. Sistem prikazuje korisniku učitani uspomenu. (IA)



Slika 9 – DS4 Osnovni scenario

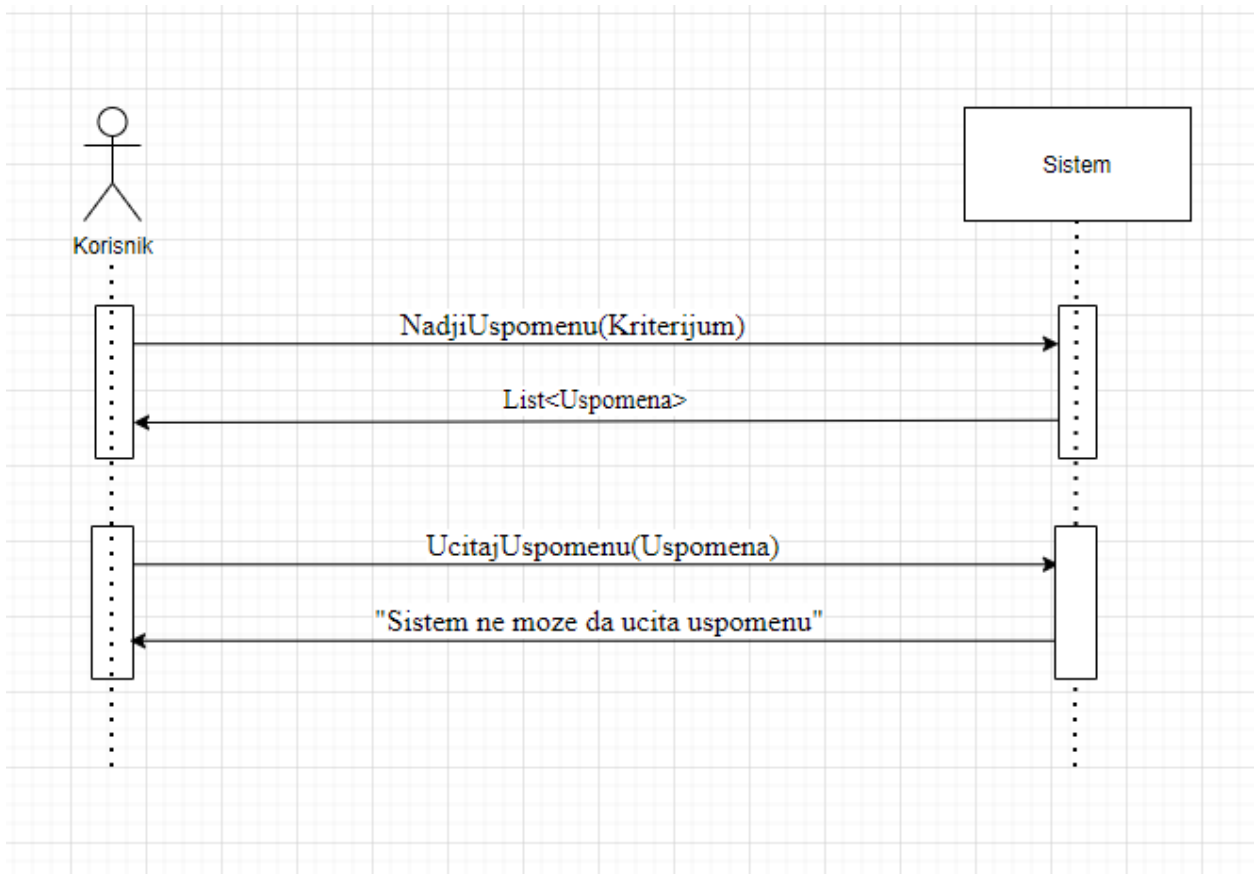
Alternativna scenarija

- 1.1 Ukoliko sistem ne može da pronade uspomenu on prikazuje korisniku poruku: "Sistem ne može da pronade uspomenu!" Prekida se izvršenje scenaria. (IA)



Slika 10 – DS4 Alternativni scenario 1

3.1 Ukoliko sistem ne može da učita uspomenu on prikazuje korisniku poruku: “Sistem ne može da učita uspomenu!” Prekida se izvršenje scenaria. (IA)



Slika 11 – DS4 Alternativni scenario 2

5. Specifikacija REST API-ja

fetchPost(id)

GET posts/:id

Resource URL

<https://iphotogramy.herokuapp.com/posts/:id>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
404	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/posts/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
404	Not found.

fetchPosts()

GET /posts?page=pageNumber

Resource URL

<https://iphotogramy.herokuapp.com/posts?page=pageNumber>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes
Requests / 8-min window (user auth)	8

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
404	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  {"id": "1234567890",
   "url": "https://your_domain.com/posts/1234567890",
   "valid": true,
   "created_at": "2016-06-02T23:54:02Z"},
  {"id": "1234889989",
   "url": "https://your_domain.com/posts/1234889989",
   "valid": true,
   "created_at": "2016-06-02T23:54:02Z"}
}
```

Error Messages

Code	Message
404	Not found.

fetchPostsBySearch(searchQuery)

GET /posts/search?searchQuery='none'&tags=

Resource URL

<https://iphotogramy.herokuapp.com/posts/search?searchQuery='none'&tags=>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes
Requests / 8-min window (user auth)	8

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
404	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  {"id": "1234567890",
    "url": "https://your_domain.com/posts/1234567890",
    "valid": true,
    "created_at": "2016-06-02T23:54:02Z"
  },
  {"id": "1234889989",
    "url": "https://your_domain.com/posts/1234889989",
    "valid": true,
    "created_at": "2016-06-02T23:54:02Z"
  }
}
```

Error Messages

Code	Message
404	Not found.

createPost(newPost)

POST posts/

Resource URL

<https://iphotogramy.herokuapp.com/posts/>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
201	Webhook URL is registered to the provided application
409	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/posts/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
409	Bad request.

likePost(id)

PATCH posts/:id/likePost

Resource URL

<https://iphotogramy.herokuapp.com/posts/:id/likePost>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/posts/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

No support.

commentPost(value, id)

POST posts/:id/commentPost

Resource URL

<https://iphotogramy.herokuapp.com/posts/:id/commentPost>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/posts/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
```

```
}
```

Error Messages

No support.

updatePost(id, updatedPost)

PATCH posts/:id

Resource URL

<https://iphotogramy.herokuapp.com/posts/:id>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
201	Webhook URL is registered to the provided application

Example Response – Success

HTTP 200

```
{  
  "id": "1234567890",  
  "url": "https://your_domain.com/posts/1234567890",  
  "valid": true,
```



```
"created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

No support.

deletePost(id)

DELETE posts/:id

Resource URL

<https://iphotogramy.herokuapp.com/posts/:id>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
201	Webhook URL is registered to the provided application

Example Response – Success

HTTP 204 No Content

Error Messages

No support.

signIn(userInfo)

POST user/signin

Resource URL

<https://iphotogramy.herokuapp.com/user/signin>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
500	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/user/signin/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
500	Something went wrong.

signUp(userInfo)

POST user/signup

Resource URL

<https://iphotogramy.herokuapp.com/user/signup>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
500	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/user/signup/1234567890",
  "valid": true,
```

```
"created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
500	Something went wrong.

follow(nas_id, id)

PATCH friends/follow/:id

Resource URL

<https://iphotogramy.herokuapp.com/friends/follow/:id>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/friends/follow/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
500	Something went wrong.

getUsers()

GET friends/

Resource URL

<https://iphotogramy.herokuapp.com/friends/>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
500	There is an error with your request. See error messages section below.

Example Response – Success

HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/user/signin/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
500	Something went wrong.

getFollowers(id)

GET friends/:id

Resource URL

<https://iphotogramy.herokuapp.com/friends:id>

Resource Information

Response Format	JSON
Requires Authentication	Yes (user context - all consumer and access tokens)
Rate Limited	Yes

Parameters

url (required)	Encoded URL for the callback endpoint.
----------------	--

HTTP Responses

HTTP Code	Message
200	Webhook URL is registered to the provided application
500	There is an error with your request. See error messages section below.

Example Response – Success

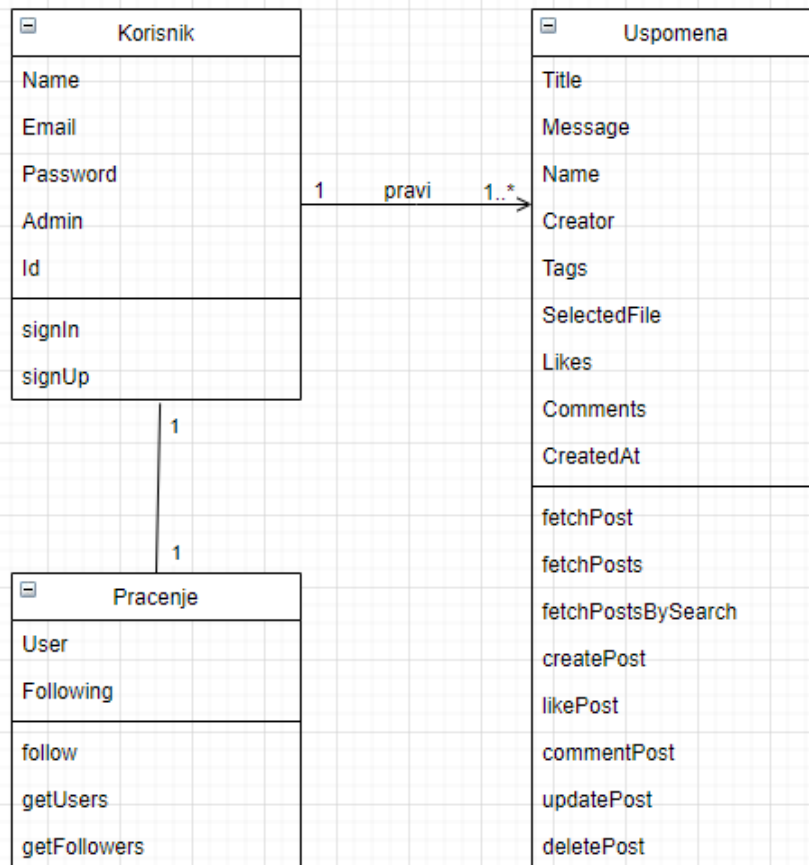
HTTP 200

```
{
  "id": "1234567890",
  "url": "https://your_domain.com/user/signin/1234567890",
  "valid": true,
  "created_at": "2016-06-02T23:54:02Z"
}
```

Error Messages

Code	Message
500	Something went wrong.

6. Model podataka

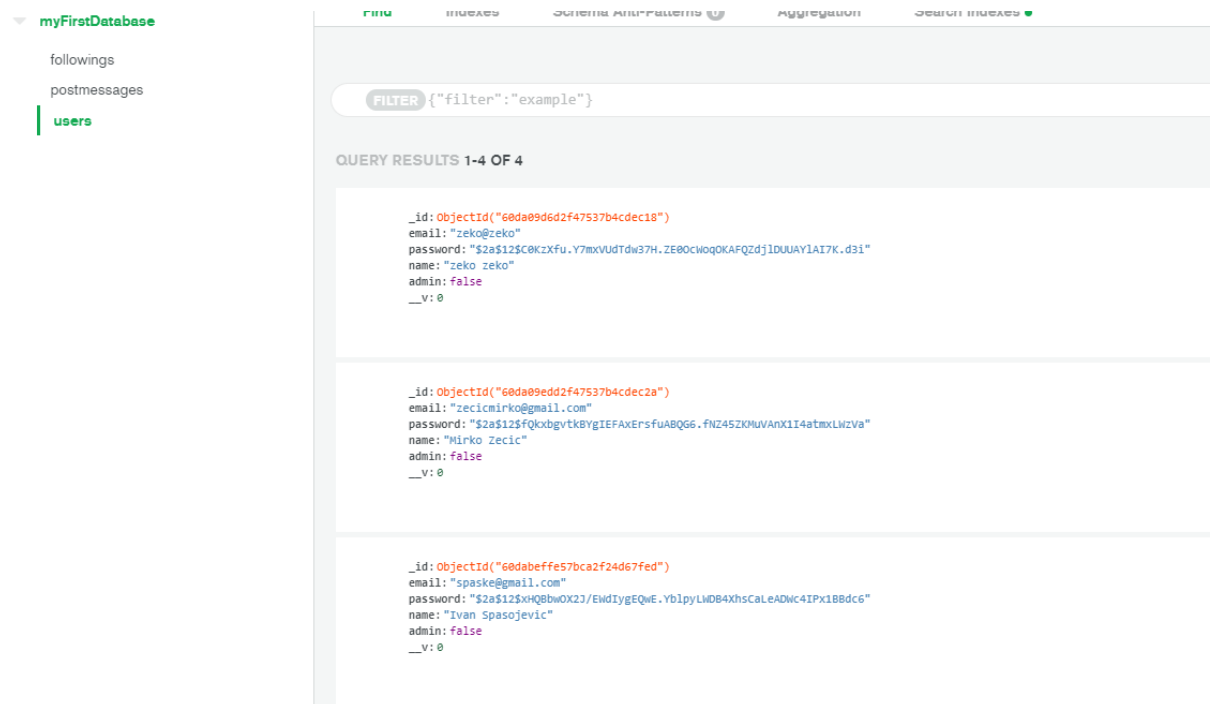


Slika 12 – Dijagram klasa

6.1 Struktura baze podataka (Konceptualni model)



Slika 13 – Konceptualni model



Slika 14 – Mongo DB – users dokument

myFirstDatabase

- followings
- postmessages**
- users

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

FILTER {"filter": "example"}

QUERY RESULTS 1-2 OF 2

```
_id: ObjectId("60da09e3d2f47537b4cdec23")
> tags: Array
> likes: Array
> comments: Array
  createdAt: 2021-06-28T17:41:55.803+00:00
  title: "zekov titl"
  message: "zekov post"
  selectedFile: ""
  name: "zeko zeko"
  creator: "60da09d6d2f47537b4cdec18"
  __v: 0
```

```
_id: ObjectId("60da09fdd2f47537b4cdec3c")
> tags: Array
> likes: Array
> comments: Array
  createdAt: 2021-06-28T17:42:21.840+00:00
  title: "Mirkov titl"
  message: "Mirkova poruka"
  selectedFile: ""
  name: "Mirko Zecic"
  creator: "60da09edd2f47537b4cdec2a"
  __v: 0
```

Slika 15 – Mongo DB – postmessages dokument

▼ myFirstDatabase

followings

postmessages

users

find

indexes

Schema Anti-Patterns

Aggregation

FILTER

{ "filter": "example" }

QUERY RESULTS 1-4 OF 4

_id: ObjectId("60da09d6d2f47537b4cdec1b")

> following: Array

user: "60da09d6d2f47537b4cdec18"

__v: 0

_id: ObjectId("60da09eed2f47537b4cdec2d")

> following: Array

user: "60da09edd2f47537b4cdec2a"

__v: 0

_id: ObjectId("60dabeffe57bca2f24d67ff0")

> following: Array

user: "60dabeffe57bca2f24d67fed"

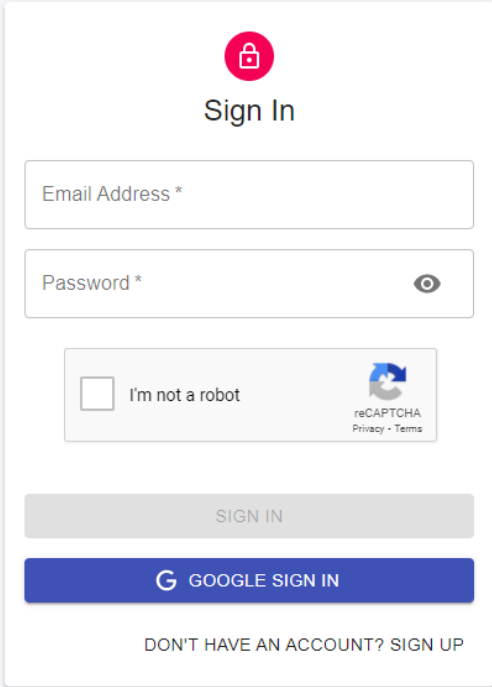
__v: 0

Slika 16 – Mongo DB – followings dokument

7. Tehnologije korišćene u aplikaciji

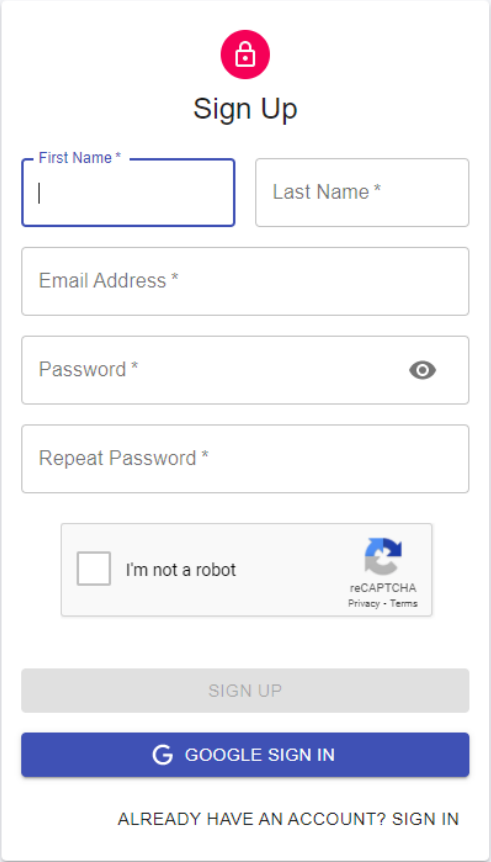
Projekat je rađen na MERN steku. M stoji ispred Mongo DB, odnosno baza podataka koju smo koristili je Mongo baza. Ona je nerelaciona baza, što znači da nije strogo struktuirana kao relaciona, već određena polja mogu imati različiti dužine. Mongo baza je dokument tip nerelacione baza, tj. podaci se predstavljaju kao dokumenti. To možete videti na slikama iznad. E stoji ispred Express.js i to je Node.js okvir, koji nam olakšava i ubrzava rad sa Node.js backend tehnologijom. Besplatan je i otvorenog koda. R stoji ispred React.js, a to je JavaScript biblioteka otvorenog koda koja nam služi da razvijamo korisnički interfejs, odnosno UI komponente. Razvijena je od strane Facebook-a. React se može koristiti kao baza za razvoj SPA (Single Page Applications) ili mobilnih aplikacija. N stoji za Node.js, tačnije backend tehnologija koja nam olakšava i ubrzava pravljenje skalabilnih web aplikacija. Node.js je “event-driven”, “non-blocking” I/O model koji na efikasan način omogućava pravljenje real-time aplikacija koji se izvršavaju na različitim platformama.

8. Korisničko uputsvo

A sign-in form with a white background and rounded corners, centered on a light blue background. At the top is a red circular icon with a white padlock. Below it is the text "Sign In". The form contains two input fields: "Email Address *" and "Password *". The password field has a small eye icon to its right. Below the password field is a reCAPTCHA section with a checkbox labeled "I'm not a robot" and a reCAPTCHA logo with links for "Privacy" and "Terms". At the bottom of the form are two buttons: a grey "SIGN IN" button and a blue "GOOGLE SIGN IN" button with the Google logo. Below the buttons is a link that says "DON'T HAVE AN ACCOUNT? SIGN UP".

Slika 17 – Prijavljivanje na sistem

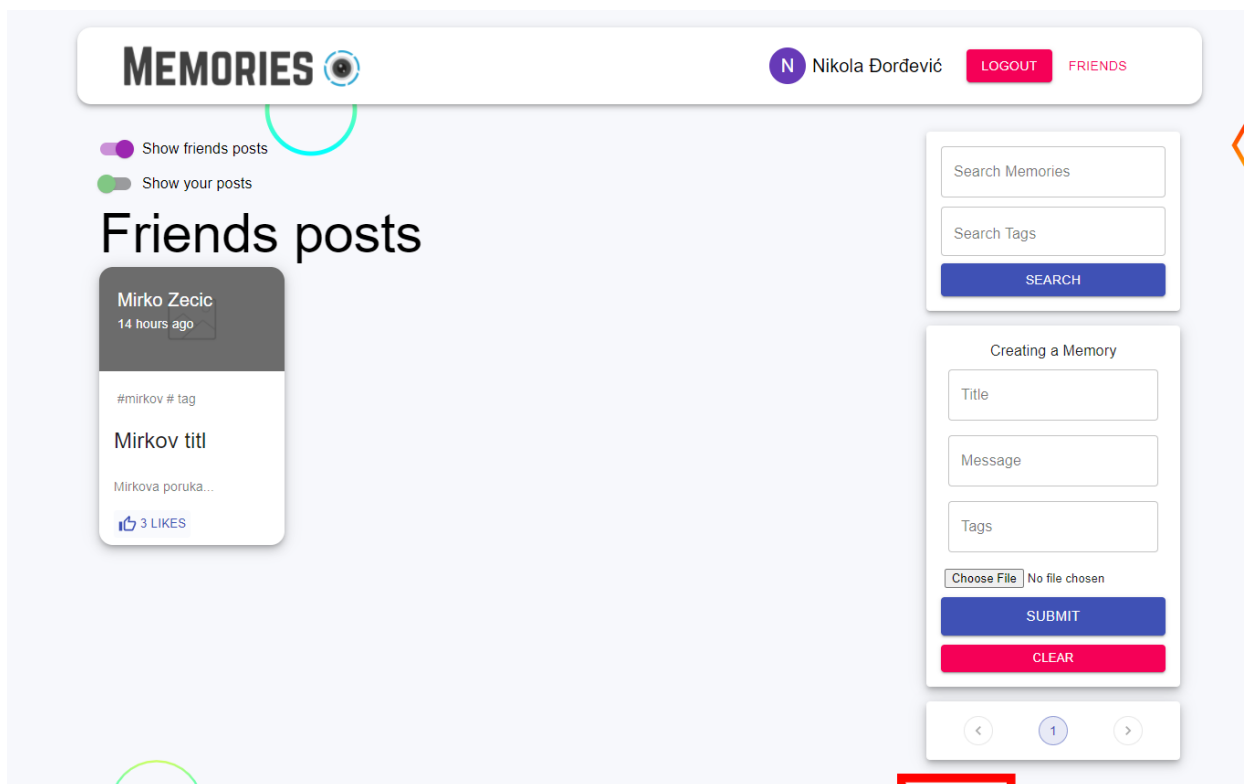
Ukoliko ste se registrovali na sistem, možete se prijaviti sa vašim kredencijalima, ali je potrebno da potvrdite da niste robot. Ukoliko ne želite na taj način da se prijavite, možete koristiti Google nalog. Ukoliko nemate nalog na našoj platformi, potrebno je da ga napravite. Sličan je rad kao i sa prethodnim slučajem korišćenja.



The image shows a 'Sign Up' form with a red lock icon at the top. The form includes input fields for 'First Name *', 'Last Name *', 'Email Address *', 'Password *' (with a toggle eye icon), and 'Repeat Password *'. Below these is a reCAPTCHA section with a checkbox labeled 'I'm not a robot' and a reCAPTCHA logo with links for 'Privacy' and 'Terms'. At the bottom, there is a grey 'SIGN UP' button, a blue 'GOOGLE SIGN IN' button, and a link that says 'ALREADY HAVE AN ACCOUNT? SIGN IN'.

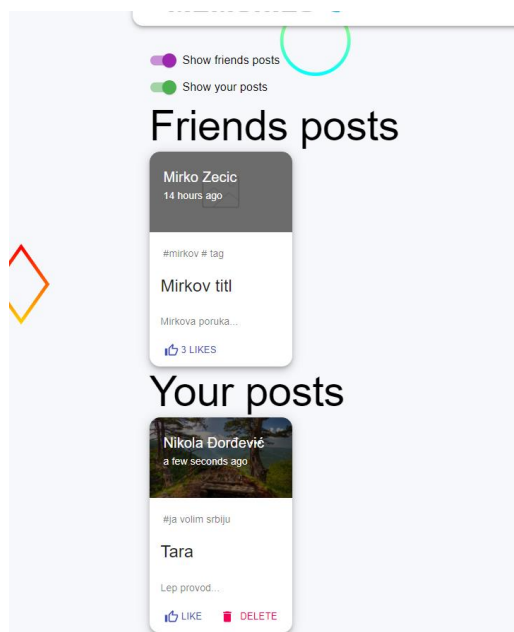
Slika 18 – Registracija

Kada se prijavite pristupate sistemu. Automatski će Vam se prikazati uspomene prijatelja koje pratiti kao i vaše samostalno kreirane uspomene.



Slika 19 – Početna strana

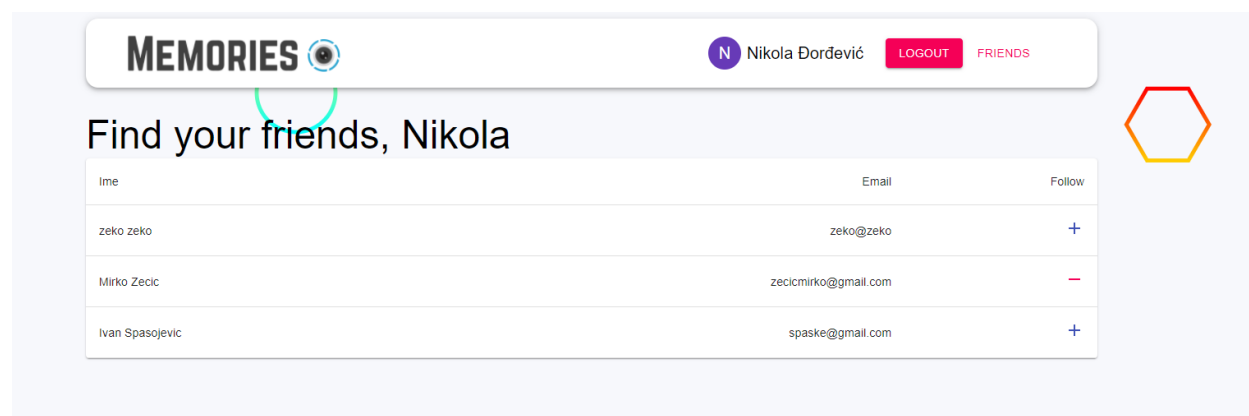
Ukoliko nemate kreirane uspomene možete to uraditi na formi sa desne strane na kojoj piše Creating a Memory. Popunite formu i klikom na submit, ona će stajati u sekciji your posts, koju je potrebno da uključiti da biste videli svoje uspomene.



Slika 20 – Uspomene

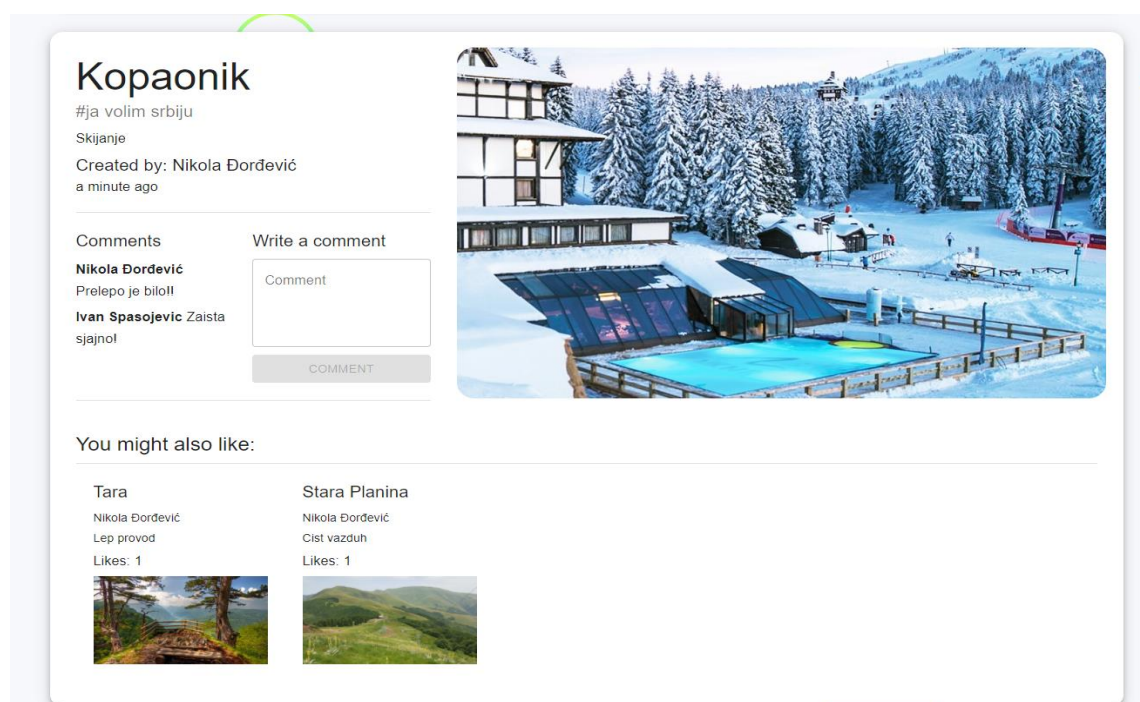
Uspomene koje ste sami kreirali možete obrisati ili izmeniti, lajkovati i komentarisati. Uspomene koje su drugi kreirali možete samo lajkovati i komentarisati. Brisanje je intuitivno, klikom na kantu za smeće. Dok je za izmenu potrebno kliknuti na tri tačkice u gornjem desnom uglu slike. Nakon toga sve je identično kao i kreiranje uspomene.

Klikom na Friends u gornjem desnom uglu, otvara se tabela sa svim korisnicima platforme, gde možete da ih zapratite ili otpratite.



Slika 21 – Friends

Klikom na određenu uspomenu, otvoriće Vam se ta uspomene, gde možete videti sliku u većoj rezoluciji, kao i komentarisati ili videti ostale komentare. Ispod naš sistem će Vam preporučiti neke uspomene koje biste možda želeli da pogledate.



Slika 22 – Uspomena detalji

Klikom na Logout, Vaša sesija se zatvara.

9. Delovi koda

API pozivi na naš bekend.

```
import axios from 'axios';
//http://localhost:5000
//https://iphotogramy.herokuapp.com/
const API = axios.create({ baseURL: 'http://localhost:5000' });

API.interceptors.request.use((req) => {
  if (localStorage.getItem('profile')) {
    req.headers.Authorization = `Bearer ${JSON.parse(localStorage.getItem('profile')).token}`;
  }
  return req;
})

export const fetchPost = (id) => API.get(`/posts/${id}`);
export const fetchPosts = (page) => API.get(`/posts?page=${page}`);
export const fetchPostsBySearch = (searchQuery) => API.get(`/posts/search?searchQuery=${searchQuery.search} || 'none'&tags=${searchQuery.tags}`);
export const createPost = (newPost) => API.post('/posts', newPost);
export const likePost = (id) => API.patch(`/posts/${id}/likePost`);
export const commentPost = (value, id) => API.post(`/posts/${id}/commentPost`, { value });
export const updatePost = (id, updatedPost) => API.patch(`/posts/${id}`, updatedPost);
export const deletePost = (id) => API.delete(`/posts/${id}`);

export const signIn = (formData) => API.post('/user/signin', formData);
export const signUp = (formData) => API.post('/user/signup', formData);
export const getUsers = () => API.get('/friends');

export const follow = (nas_id, id) => API.patch(`/friends/${id}`, { nas_id });
export const getFollowers = (id) => API.get(`/friends/${id}`);
```

Slika 23 – API pozivi

Rute na bekend delu.

```
import express from 'express';

import { getPostsBySearch, getPosts, getPost, createPost } from '../controllers/post';
import auth from '../middleware/auth.js';

const router = express.Router();

router.get('/search', getPostsBySearch);
router.get('/', getPosts);
router.get('/:id', getPost);

router.post('/', auth, createPost);
router.patch('/:id', auth, updatePost);
router.delete('/:id', auth, deletePost);
router.patch('/:id/likePost', auth, likePost);
router.post('/:id/commentPost', commentPost);

export default router;
```

Slika 24 – Rute na bekindu

Kontroler na backend strani koji će obraditi zahtev za prijavljivanje sa frontenda.

```
export const signin = async (req, res) => {
  const { email, password } = req.body;

  try {
    const existingUser = await User.findOne({ email });

    if (!existingUser) return res.status(404).json({ message: "User doesn't exists!" });

    const isPasswordCorrect = await bcrypt.compare(password, existingUser.password);

    if (!isPasswordCorrect) return res.status(400).json({ message: "Invalid credentials. " });

    //Ako smo pronasli user-a izvlacimo token za njega i vracamo ga na front
    //Stavljamo test kao secret, inace bi trebalo to da se izvuče u .env fajl
    const token = jwt.sign({ email: existingUser.email, id: existingUser._id }, 'test', { expiresIn: "1h" });

    res.status(200).json({ result: existingUser, token });
  } catch (error) {
    res.status(500).json({ message: "Something went wrong. " });
  }
}
```

Slika 25 – Kontroler i callback funkcija signin

Middleware koji će nam omogućiti da samo onaj ko je kreirao uspomenu može da je izmeni i obriše.

```
const auth = async(req, res, next) => {
  try {
    //ideja ovde je da kada se korisnik prijavi on dobije token
    //Mi sada zelimo da proverimo da je to stvarno on, kao i da mu nakon toga damo privilegije
    const token = req.headers.authorization.split(" ")[1];
    //Postoji i Google-ov i nas token (customAuth) pa moramo da proverimo
    const customAuth = token.length < 500;

    let decodedData;

    if(token && customAuth) {
      //Hocemo preko verify funkcije da izvučemo podatke o tom korisniku iz tokena
      //saljemo secret koji smo ranije kreirali u controlleru
      decodedData = jwt.verify(token, 'test');
      req.userId = decodedData?.id;
      //ovi req.userId ce biti dostupni u funkcijama u kontrolerima pre kojih smo
      //unutar ruta stavili auth
    } else {
      //Google
      decodedData = jwt.decode(token);
      //sub je google-ov naziv za id
      req.userId = decodedData?.sub;
    }

    next();
  } catch (error) {
    console.log(error);
  }
}
```

Slika 26 – Middleware

10. Github repozitorijum

<https://github.com/nikollace/SocialNetwork>