Moviehub API-dokumentaatio

BaseURL: https://localhost:5000

1. Käyttäjän rekisteröityminen:

Endpoint: POST /api/register
Kuvaus: Rekisteröi uuden käyttäjän MovieHub-palveluun.
Pyyntö:

```
{
  "name": "Etunimi",
  "lname": "Sukunimi",
  "password": "Salasana123",
  "email": "kayttaja@example.com",
  "username": "kayttaja123",
  "pronouns": "she/her"
}
```

Vastaus:
Onnistunut:

```
{
  "success": true,
  "message": "Registration successful"
}
```

Error:

```
{
  "error": "Internal server error",
  "status": 500
}
```

2. Käyttäjän kirjautuminen:

Endpoint: POST/api/login

Kuvaus: Autentikoi käyttäjän kirjautumisen avulla. Tarkistaa tietokannasta usernamen sekä passwordin.

Pyyntö:

```
{
  "username": "kayttaja123",
  "password": "salasana123"
}
```

Vastaus:

Onnistunut:

```
{
  "success": true,
  "userId": 123,
  "message": "Login successful"
}
```

Error:

```
{
  "error": "User not found",
  "status": 404
}
```

Error:

```
{
  "error": Invalid password
  "status": 401
}
```

Error:

```
{
  "error": " Internal server error ",
  "status": 500
}
```

3. Päivitä käyttäjän tietoja:

Endpoint: POST /api/user/:userId

Kuvaus: Päivittää käyttäjän tietoja.

Pyyntö:

Parametrit:
- userId : userId, kenen dataa muokataan

Vastaus:

Status OK:


```
{
  "user_id": "int",
  "firstname": "string",
  "email": "string",
  "pronouns": "string",
  "birthdate": "string",
  "phone_number": "string",
  "address": "string",
  "profile_picture": "string (base64 encoded)"
}
```


Error:


```
{
 "error": "Internal Server Error",
"status": 500,
}
```


4. Päivitä käyttäjän salasana:

Endpoint: POST /api/password/:userId
Kuvaus: Päivitä käyttäjän salasana annetun UserId:n ja käyttäjän syöttämän uuden salasanan perusteella.
Pyyntö:


```
{
  "newPassword": "string"
}
```


Parametrit:
- userId : userId, jonka salasana päivitetään.

Tiedot:

- newPassword: Uusi salasana.

Vastaus:

Onnistunut:

```
{
 "user_id": "123",
// ... lisää käyttäjän tietoja
 }
```

Error:

```
{
"error": " User not found",
"status": 404
 }
```

Error:

```
{
"error": "Internal server error",
"status": 500
 }
```

5. Päivitä käyttäjän username:

Endpoint: POST /api/username/:userId
Kuvaus: Päivitä käyttäjän käyttäjänimi annetun userId:n ja uuden käyttäjänimen perusteella.

Pyyntö:

```
{
  "username": "string"
}
```

Parametrit:

- userId: userId, jonka käyttäjänimi päivitetään.

Tiedot:

- username: Uusi käyttäjänimi.

Vastaus:

Onnistunut:

{
"user_id": "123",
// ... lisää käyttäjän tietoja
}

Error:

{
"error": "User not found",
"status": 404
}

Error:

{
"error": "Internal server error",
"status": 500
}

6. Poista käyttäjätili ja sen tiedot:

Endpoint: DELETE /api/user/:userId

Kuvaus: Poista käyttäjätili ja siihen liittyvät tiedot.

Pyyntö:

Parametrit:

- userId: userId, jonka tiedot halutaan poistaa.

Vastaus:

Onnistunut:

{
"message": "Account and data deleted successfully",
"deletedUser": ”123”
}

Error:

{

"error": "User not found",
"status": 404
}

Error:
{
"error": "Internal server error",
"status": 500
}

7. Hae käyttäjän tiedot:

Endpoint: GET /api/user/:userId
Kuvaus: Hae käyttäjän tiedot annetun userId:n perusteella.
Pyyntö:
Parametrit:
- userId: userId, jonka tiedot halutaan hakea.

Vastaus:
Onnistunut:
{
"user_id": "123",
"firstname": "Joku",
"lastname": "Jokunen",
"email": "joku@example.fi",
"username": "johndoe",.... yms
}
Error:
{
"error": "User not found",
"status": 404
}

Error:api/user
{

"error": "Internal Server Error"
"status": 500

}

Error:

{
"error": "Internal server error",
"status": 500
}

8. Lisää elokuva suosikkeihin:

Endpoint: POST /api/add-to-favorites
Kuvaus: Lisää elokuvan käyttäjän suosikkeihin MovieHub-palvelussa.
Pyyntö:

{
"userId": "kayttaja123",
"movieId": "elokuva456",
"username": "kayttaja123"
}

Vastaus:
Onnistunut:

{
"success": true
}

Error:

{
"error": " Movie already favorited by this user",
"status": 400
}

Error:

```
{
  "error": "Internal server error",
  "status": 500
}
```

9. Lisää tähtiarvostelu:

Endpoint: POST /api/add-rating
Kuvaus: Lisää tähtiarvostelu elokuvalle.
Pyyntö:

```
{
"userId": "123",
"movieId": "456",
"rating": 4,
"username": "user123"
 }
```

Vastaus:

Onnistunut:

```
{
”success": true
 }
```

Error:

```
{
"error": "Internal Server Error",
"status": 500
 }
```

10. Lisää tekstiarvostelu:

Endpoint: POST /api/add-a-review
Kuvaus: Käyttäjä voi antaa tekstimuotoisen arvostelun.
Pyyntö:

```
{
  "movieId": "int",
  "reviewText": "string",
  "username": "string",
  "userId": "int"
}
```

Vastaus:
Onnistunut:

```
{
  "success": true,
  "message": "Review added successfully"
}
```

Epäonnistunut:

```
{
  "success": false,
  "error": "Internal Server Error"
}
```

11. Kaikki elokuva-arvostelut:

Endpoint: GET /api/data
Kuvaus: Hakee kaikki datan movie_reviews taulukosta.
Vastaus:
Onnistunut:

```
[
  {
    "review_id": "int",
```

```
    "user_id": "int",
    "movie_id": "int",
    "rating": "number",
    "review_text": "string",
    "created_at": "string",
  },
  // ... muut arvostelut
]
```


Error:


```
{
  "error": "Internal Server Error"
}
```


12. Hae käyttäjän suosikit:

Endpoint: GET /api/favorites/:userId
Kuvaus: Hae käyttäjän suosikit annetun userId:n perusteella.
Pyyntö:
Parametrit:

- userId: userId, jonka suosikit halutaan hakea.

Vastaus:
Onnistunut:


```
{
"user_id": "123",
"movie_id": "456",
"username": "kayttajanimi",
"favorite_id":"2",
  }
```


Error:


```
{
"error": " Favorites not found",
"status": 404
 }
```


Error:

{
"error": "Internal user error",
"status": 500
}

13. Hae kaikkien käyttäjien suosikit:

Endpoint: GET/api/data-fav
Kuvaus: Hakee kaikki käyttäjien suosikit.
Pyyntö: Ei tarvita parametrejä, valitsee kaikki taulusta favorites.
Vastaus:

Onnistunut:

{

"user_id": "123",

"movie_id": "456",

"username": "user123"

}

Error:

{
"error": "Internal Server Error",
"status": 500
}

14. Lähetä Lomake

Endpoint: POST /submit-form

Kuvaus: Lähetä contact us-sivustolta viesti käyttäjän tiedoilla.
Pyyntö:

```
{
  "name": "string",
  "email": "string",
  "message": "string"
}
```

• name: Käyttäjän nimi.

• email: Käyttäjän sähköpostiosoite.

• message: Käyttäjän lähettämä viesti.

Vastaus:
Onnistunut:

```
{
"success": true
 }
```

Error:

```
{
"success": false,
"error": " Error submitting form",
"status": 500
 }
```

15. Tallenna käyttäjän preferenssit:

Endpoint: POST /api/user-preferences
Kuvaus: Tallenna käyttäjän preferenssit omalle kustomoidulle sivulle (uutiset, elokuvat, arvostelut ja TV-ohjelmat).
Pyyntö:

{
  "userId": "int",
  "newsPreference": "boolean",
  "moviesPreference": "boolean",
  "reviewsPreference": "boolean",
  "tvPreference": "boolean"
}

- userId: UserId preferensseille.
- newsPreference (boolean): Käyttäjän asetus uutissisällölle.
- moviesPreference (boolean): Käyttäjän asetus elokuviin liittyvälle sisällölle.
- reviewsPreference (boolean): Käyttäjän asetus arvostelusisällölle.
- tvPreference (boolean): Käyttäjän asetus TV-ohjelmasisällölle.

Vastaus:

Onnistunut:

{
"user_id": "123",
"news_preference": true,
"movies_preference": true,
"reviews_preference": false,
"tv_preference": true
 }

Error:

{
"error": "Internal server error",
"status": 500
 }

16. Hae käyttäjän preferenssit:

Endpoint: GET /api/user-preferences/:userId
Kuvaus: Hae käyttäjän preferenssit annetun userId:n perusteella.
Pyyntö:

Parametrit:

- userId: userId, jonka asetukset halutaan hakea.

Vastaus:

Onnistunut:

{ "
"user_id": "123",
"news_preference": true,
"movies_preference": true,
"reviews_preference": false,
"tv_preference": true
}

Error:

{
"error": "User preferences not found",
"status": 404
}

17. Luo jaettava linkki käyttäjän preferensseistä:

Endpoint: POST /api/generate-link

Kuvaus: Luo jakokelpoinen linkki käyttäjän preferensseistä.

Pyyntö:

{
"userId": "int"
}

Tiedot:

- userId: userId, jonka asetuksista linkki luodaan.

Vastaus:

Onnistunut:

{
"link": "jakokelpoinen-linkki"
}

Error:

{
"error": "User not found",
"status": 404
}

Error:

{
"error": "Internal server error",
"status": 500
}

18. Hae käyttäjän preferenssit linkin avulla:

Endpoint: GET /share/:link
Kuvaus: Hae käyttäjän preferenssit jakokelpoisella linkillä.
Pyyntö:
Parametrit:

- link: Jakokelpoinen linkki, jonka avulla haetaan käyttäjän asetukset.

Vastaus:
Onnistunut:
{ "userPreferences": { // ... käyttäjän asetukset } }

Error:

{
"error": " Link not found",
"status": 404
}

Error:

{
"error": "Internal server error",
"status": 500
}

19. Hae käyttäjän ryhmät:

Endpoint: GET /api/groups/user/:id
Kuvaus: Hae käyttäjän ryhmät annetun UserId:n perusteella.
Pyyntö:
Parametrit:

- id: userId, jonka ryhmät halutaan hakea.

Vastaus:
Onnistunut:

```
{
"group_id": "123",
"name": "Elokuvaharrastajat",
"description": "Ryhmä elokuvista kiinnostuneille",
 // ... lisää ryhmätietoja
}
```

Error:

```
{
"error": " User not found",
"status": 404
}
```

Error:

```
{
"error": " Internal server error",
"status": 500
}
```

20. Hae kaikki ryhmät:

Endpoint: GET /api/groups
Kuvaus: Hae kaikki olemassa olevat ryhmät.
Vastaus:
Onnistunu:

```
{
```

"group_id": "ryhma123",
"name": "Elokuvaharrastajat",
"description": "Ryhmä elokuvista kiinnostuneille",
}

Error:

{
"error": "Internal server error",
"status": 500
}

21. Hae suosituimmat ryhmät:

Endpoint: GET /api/groups-most-members/:limit
Kuvaus: Hae eniten jäseniä sisältävät ryhmät annetun rajoituksen perusteella.
Pyyntö:
Parametrit:

- limit (integer): Määrä ryhmiä, jotka halutaan hakea.

Vastaus:

Onnistunut:

{ "group_id": "ryhma123",
"name": "Elokuvaharrastajat",
"description": "Ryhmä elokuvista kiinnostuneille",
}

Error:

{
"error": "Internal server error",
"status": 500
}

22. Luo uusi ryhmä:

Endpoint: POST /api/create-a-group
Kuvaus: Luo uusi ryhmä käyttäjän antamilla tiedoilla.
Pyyntö:
{

```
  "groupName": "string",
  "groupDescription": "string",
  "members": ["string"],
  "ownerId": "int"
}
```

Tiedot:

- groupName: Ryhmän nimi.
- groupDescription: Ryhmän kuvaus.
- members: Ryhmän jäsenet.
- ownerId: Ryhmän omistajan id.

Vastaus:

Onnistunut:

```
{
"success": true,
"group": { // ... ryhmän tiedot }
 }
```

Error:

```
{
"success": false,
"error": "Internal server error",
"status": 500
 }
```

23. Päivitä ryhmän nimi:

Endpoint: PUT /api/groups/:group_id

Kuvaus: Päivitä ryhmän tiedot annetun ryhmäID:n ja uuden nimen perusteella.

Pyyntö:

```
{
  "name": "string"
}
```

Parametrit:

- group_id: jonka tietoja päivitetään.

Tiedot:

- name: Uusi nimi ryhmälle.

Vastaus:

Onnistunut:

{
"success": true,
"group": { // ... ryhmän tiedot }
}

Error:

{
"success": false,
"error": "No group found",
"status": 404
}

Error:

{
"success": false,
"error": "Internal server error",
"status": 500
}

24. Poista ryhmä:

Endpoint: DELETE /api/delete-groups/:group_id
Kuvaus: Poista ryhmä ja siihen liittyvät tiedot annetun ryhmäID:n perusteella.

Pyyntö:

Parametrit:

- group_id: jonka tiedot halutaan poistaa.

Vastaus:

Onnistunut Vastaus:

{ "message": "Ryhmä ja tiedot poistettu onnistuneesti",
"deletedGroup": { // ... ryhmän tiedot } }

Error:

{
"error": " Group not found",
"status": 404
}

Error:

{
"error": "Internal server error",
"status": 500
}