Online Quiz Platform

Project   Description:

  Building an online quiz platform where users can take quizzes on various topics. The quizzes are multiple-choice questions with a single correct answer. Users can sign up, create quizzes, take quizzes, and view their results. The project is built using Django Rest Framework with JWT authentication

The following features are implemented:

1. User Authentication
2. Quiz Creation
3. Quiz Taking
4. Quiz Results
5. Quiz Listing
6. Quiz Filtering
7. Quiz Analytics
8. User Profile
9. User Management

## Requirements:

- Django
- Pillow
- Markdown
- Djangorestframework
- Django-filter
- Djangorestframework-simplejwt

USER AUTHENTICATION

Users should be able to sign up, log in, and log out. The authentication system should use JWT tokens.

- **Registration (Sign up)**
    - → User can register into the app by passing the email and password. Here the email is used as username for login purpose
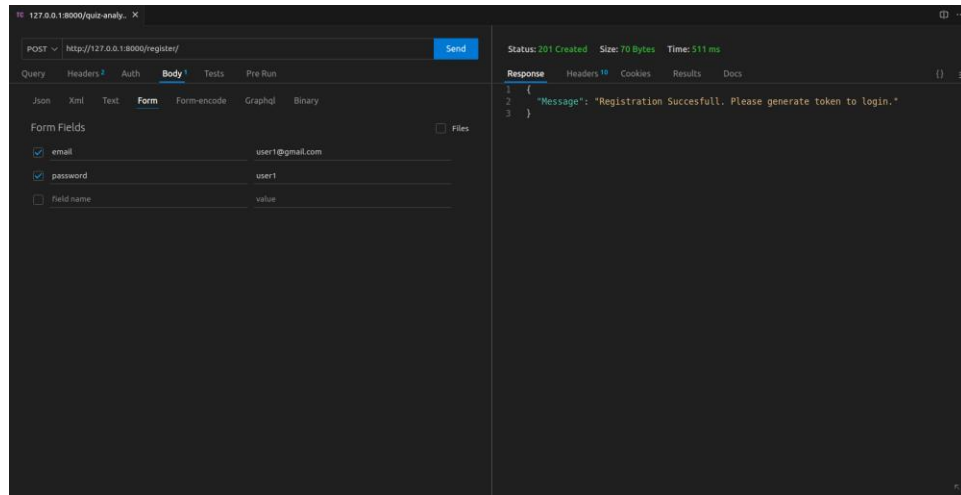
- **Login**
  $\rightarrow$ User can login into the app by using the email as username and password
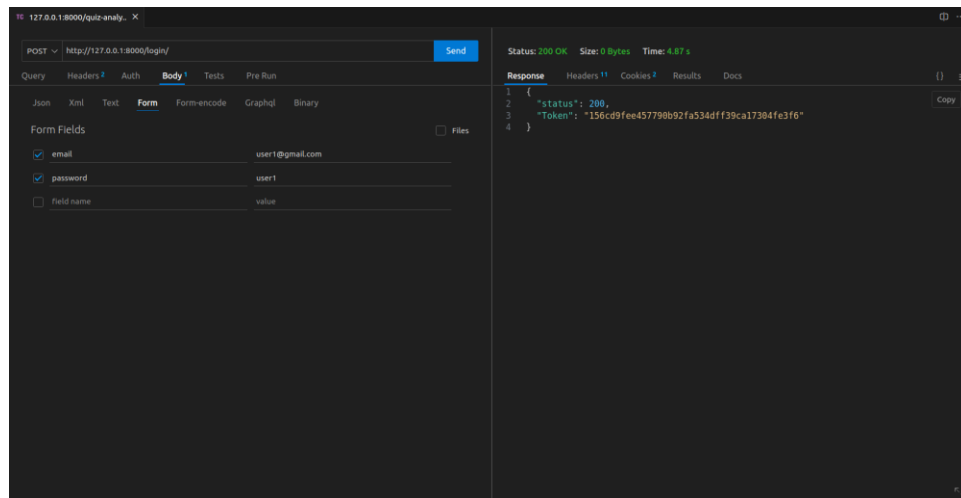
- **Token Generation**
  $\rightarrow$ By submitting the username and password, the token will be generated for the user

- **Token verification**
  → By submitting the user generated token, the token will be validated

- **Login View (profile view)**
  → By pasting the verified token in bearer space user can access the app

QUIZ CREATION

Users should be able to create quizzes. Each quiz should have a title and a set of questions with multiple choices

- **Create Quiz**
    → Authenticated user can access to create a quiz under the category

- **Create Multiple Choice Question**
    → Authenticated user can access to create multiple choice questions

QUIZ TAKING

Users should be able to take quizzes. Each quiz should have a set of questions with multiple choices, and the user should be able to select their answer. After the user has completed the quiz, the system should display their score.

127.0.0.1:8000/quiz-analy... ✕

POST ⌄   http://127.0.0.1:8000/quiz-play/   Send

Query   Headers ²   Auth ¹   **Body** ¹   Tests   Pre Run

Json   Xml   Text   Form   Form-encode   Graphql   Binary

Json Content   Format
```
1  {
2      "question" : 7,
3      "answered" : "3"
4  }
```

Status: 400 Bad Request   Size: 42 Bytes   Time: 8 ms

Response   Headers ¹⁰   Cookies   Results   Docs   {} ☰
```
1  {
2      "Error": "Already answered this question"
3  }
```
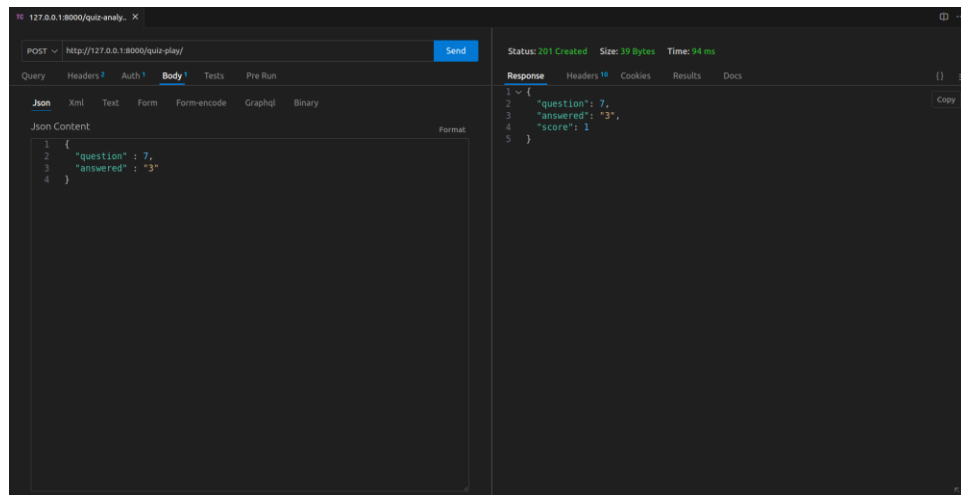Copy

127.0.0.1:8000/quiz-analy... ✕

POST ⌄   http://127.0.0.1:8000/quiz-play/   Send

Query   Headers ²   Auth ¹   **Body** ¹   Tests   Pre Run

Json   Xml   Text   Form   Form-encode   Graphql   Binary

Json Content   Format
```
1  {
2      "question" : 6,
3      "answered" : "MVT"
4  }
```

Status: 201 Created   Size: 41 Bytes   Time: 311 ms

Response   Headers ¹⁰   Cookies   Results   Docs   {} ☰
```
1  {
2      "question": 6,
3      "answered": "MVT",
4      "score": 0
5  }
```

QUIZ RESULT

Users should be able to view their quiz results. The system should display the user's score for each quiz they have taken
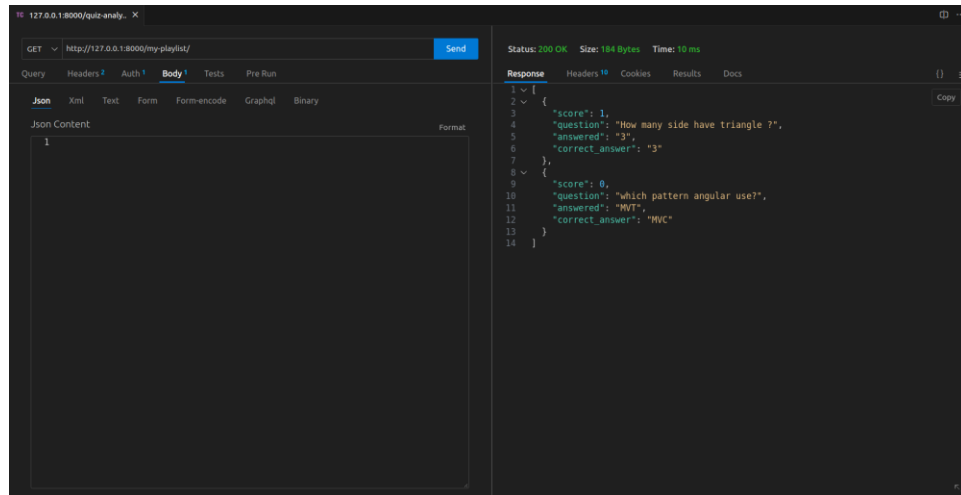
QUIZ LISTING

The system should display a list of available quizzes that users can take
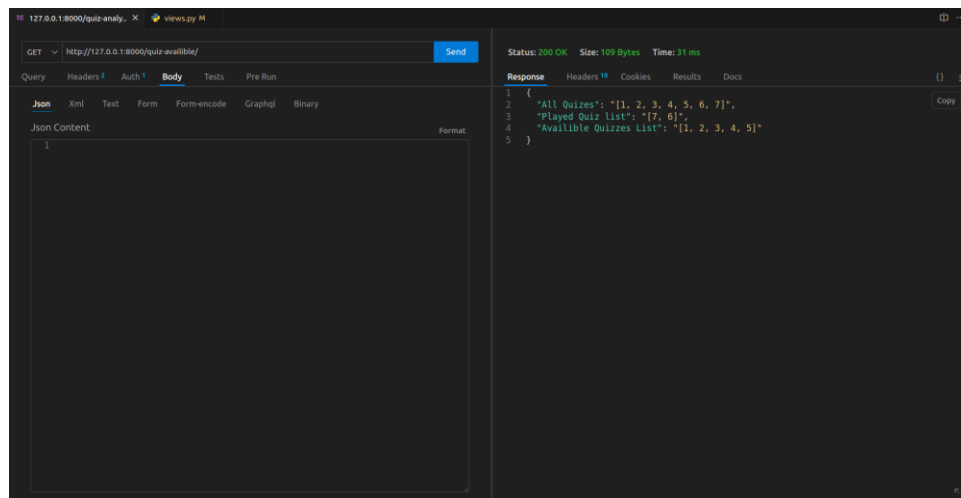
QUIZ FILTERING

Users should be able to filter quizzes by topic, difficulty level, and date created

GET http://127.0.0.1:8000/question-list/?search=angular **Send**

Status: 200 OK Size: 446 Bytes Time: 15 ms

Query Headers² Auth¹ **Body** Tests Pre Run

Json Xml Text Form Form-encode Graphql Binary

Json Content Format

```
1  [
2      {
3          "id": 1,
4          "quiz": 1,
5          "difficulty": 1,
6          "question": "what is angular",
7          "option_one": "front end framework",
8          "option_two": "backend framework",
9          "option_three": "none of the above",
10         "correct_answer": "front end framework",
11         "date_created": "2023-03-02",
12         "user": "admin"
13     },
14     {
15         "id": 6,
16         "quiz": 1,
17         "difficulty": 1,
18         "question": "which pattern angular use?",
19         "option_one": "MVC",
20         "option_two": "MVT",
21         "option_three": "MVM",
22         "correct_answer": "MVC",
23         "date_created": "2023-03-02",
24         "user": "admin"
25     }
26 ]
```
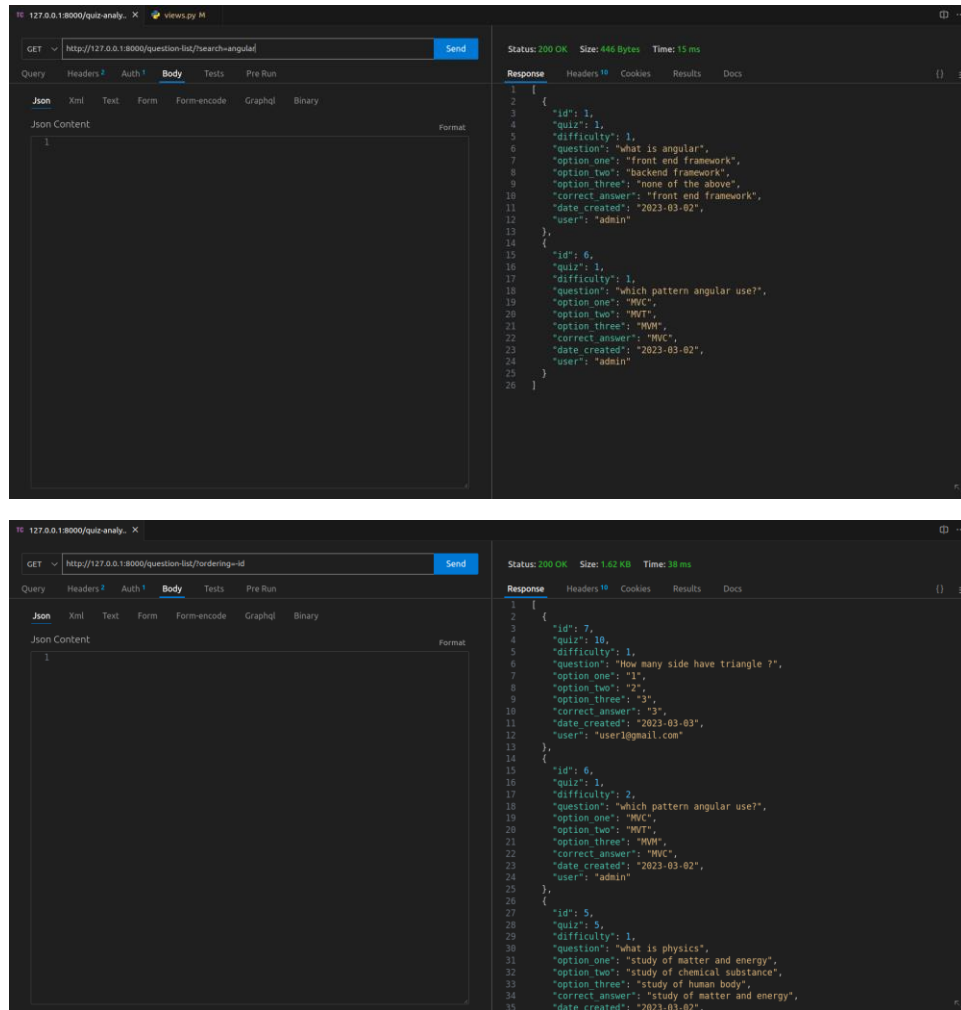
GET http://127.0.0.1:8000/question-list/?ordering=id **Send**

Status: 200 OK Size: 1.62 KB Time: 38 ms

Query Headers² Auth¹ **Body** Tests Pre Run

Json Xml Text Form Form-encode Graphql Binary

Json Content Format

```
1  [
2      {
3          "id": 7,
4          "quiz": 10,
5          "difficulty": 1,
6          "question": "How many side have triangle ?",
7          "option_one": "1",
8          "option_two": "2",
9          "option_three": "3",
10         "correct_answer": "3",
11         "date_created": "2023-03-03",
12         "user": "user1@gmail.com"
13     },
14     {
15         "id": 6,
16         "quiz": 1,
17         "difficulty": 2,
18         "question": "which pattern angular use?",
19         "option_one": "MVC",
20         "option_two": "MVT",
21         "option_three": "MVM",
22         "correct_answer": "MVC",
23         "date_created": "2023-03-02",
24         "user": "admin"
25     },
26     {
27         "id": 5,
28         "quiz": 5,
29         "difficulty": 1,
30         "question": "what is physics",
31         "option_one": "study of matter and energy",
32         "option_two": "study of chemical substance",
33         "option_three": "study of human body",
34         "correct_answer": "study of matter and energy",
35         "date_created": "2023-03-02",
```
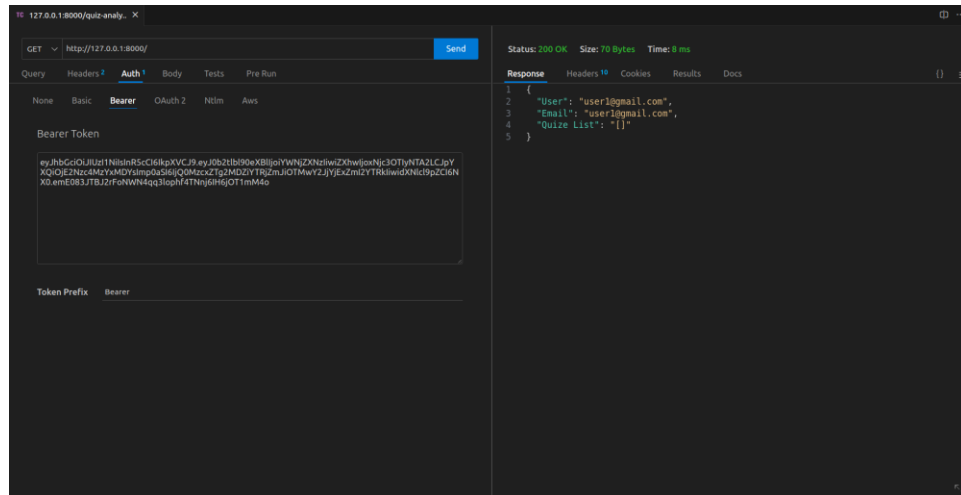
QUIZ ANALYSIS

The system should display analytics on each quiz, such as the average score, the number of times the quiz has been taken, and the percentage of users who have passed the quiz

- **User Profile**
  → Users should be able to view their profile, which should display their username, email, and a list of quizzes they have created
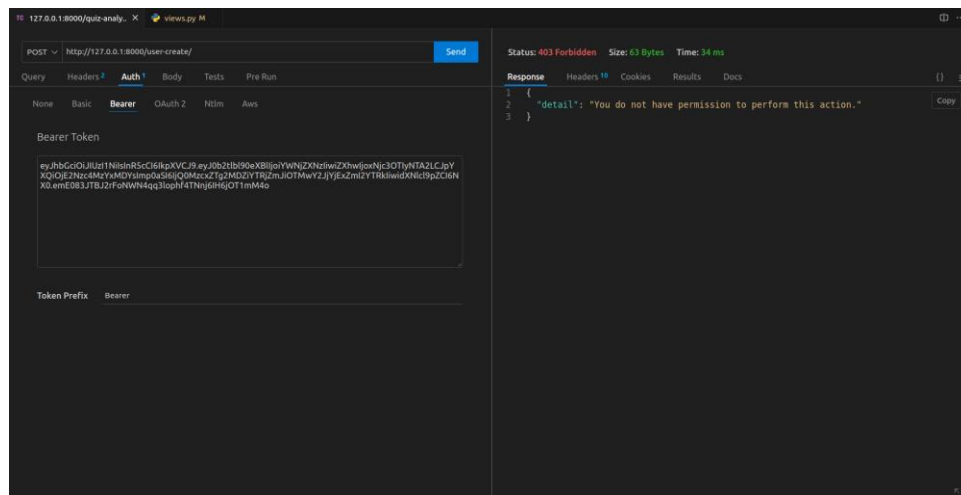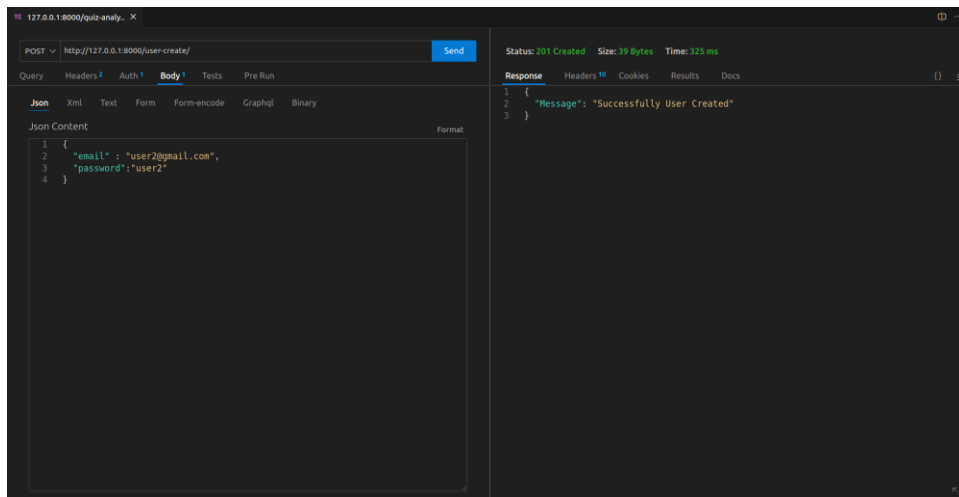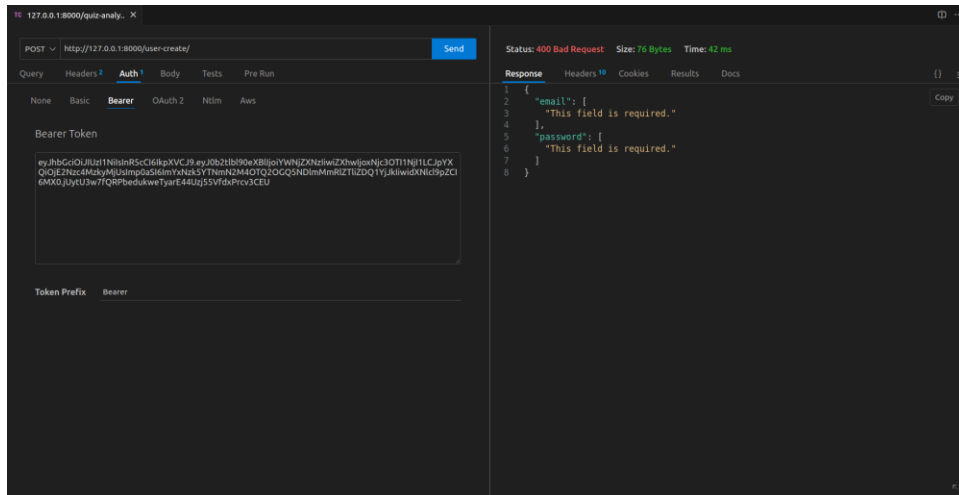
USER MANAGEMENT

Admin users should be able to manage users, including creating, editing, and deleting user accounts

- **User Creation**
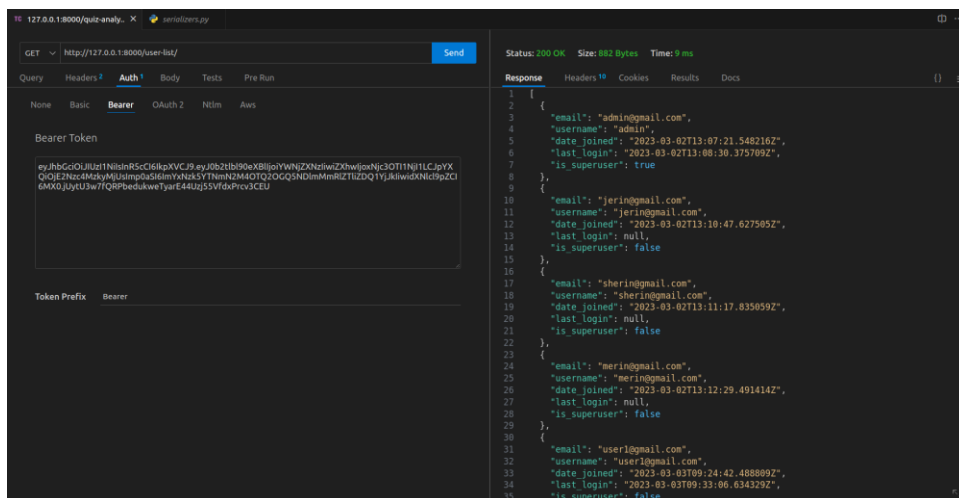 → Only admin user has permission to create the user

→ To access this page, admin is to generate token and copy the access toke and paste in bearer token space, now the admin user can access the page to create a user
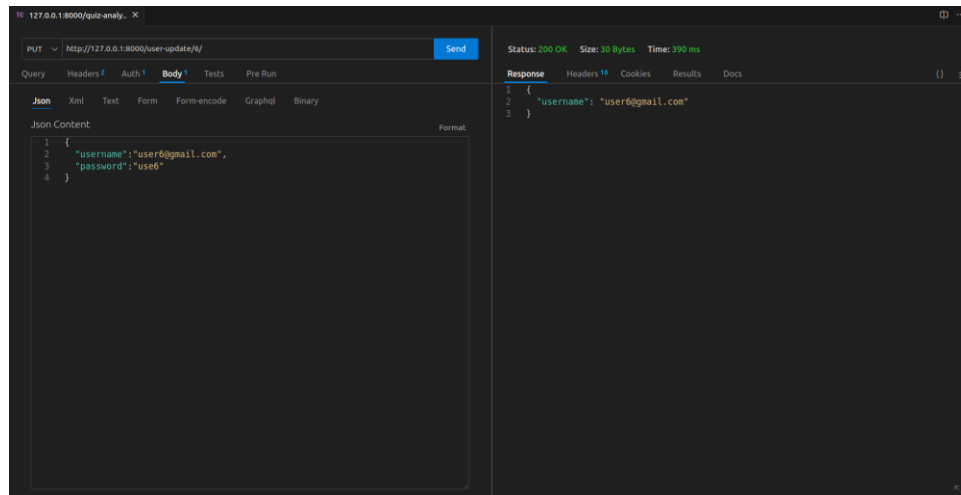
Screenshot 1 (POST user-create - 400 Bad Request):

127.0.0.1:8000/quiz-analy..

POST http://127.0.0.1:8000/user-create/   Send

Query   Headers ²   Auth ¹   Body   Tests   Pre Run

None   Basic   **Bearer**   OAuth 2   Ntlm   Aws

Bearer Token

ey.JhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjc3OTI1NjI1LCJpYXQiOiJE2Nzc4MzkyMjUsImp0aSI6ImYxNzk5N2M4OTQ2YT5mMmRIZTlliZDQ1YjJJkIiwidXNlcl9pZCI6MMX0.jUytU3w7fQRPbedukweTyarE44Uzj55VfdxPrcv3CEU

Token Prefix   Bearer

Status: **400 Bad Request**   Size: **76 Bytes**   Time: **42 ms**

Response   Headers ¹⁰   Cookies   Results   Docs   {}   ☰   Copy

```
1  {
2    "email": [
3      "This field is required."
4    ],
5    "password": [
6      "This field is required."
7    ]
8  }
```

Screenshot 2 (POST user-create - 201 Created):

127.0.0.1:8000/quiz-analy..

POST http://127.0.0.1:8000/user-create/   Send

Query   Headers ²   Auth ¹   **Body** ¹   Tests   Pre Run

Json   Xml   Text   Form   Form-encode   GraphQl   Binary

Json Content   Format

```
1  {
2    "email" : "user2@gmail.com",
3    "password":"user2"
4  }
```

Status: **201 Created**   Size: **39 Bytes**   Time: **325 ms**

Response   Headers ¹⁰   Cookies   Results   Docs   {}   ☰

```
1  {
2    "Message": "Successfully User Created"
3  }
```

- **User List**
  → It will show the list of all the user they are registered

Screenshot 3 (GET user-list - 200 OK):

127.0.0.1:8000/quiz-analy.. ×   serializers.py

GET http://127.0.0.1:8000/user-list/   Send

Query   Headers ²   **Auth** ¹   Body   Tests   Pre Run

None   Basic   **Bearer**   OAuth 2   Ntlm   Aws

Bearer Token

ey.JhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ0b2tlbl90eXBlIjoiYWNjZXNzIiwiZXhwIjoxNjc3OTI1NjI1LCJpYXQiOiJE2Nzc4MzkyMjUsImp0aSI6ImYxNzk5N2M4OTQ2YTVmMmRIZTlliZDQ1YjJJkIiwidXNlcl9pZCI6MMX0.jUytU3w7fQRPbedukweTyarE44Uzj55VfdxPrcv3CEU

Token Prefix   Bearer

Status: **200 OK**   Size: **882 Bytes**   Time: **9 ms**

Response   Headers ¹⁰   Cookies   Results   Docs   {}   ☰

```
1   [
2     {
3       "email": "admin@gmail.com",
4       "username": "admin",
5       "date_joined": "2023-03-02T13:07:21.548216Z",
6       "last_login": "2023-03-02T13:08:30.375709Z",
7       "is_superuser": true
8     },
9     {
10      "email": "jerin@gmail.com",
11      "username": "jerin@gmail.com",
12      "date_joined": "2023-03-02T13:10:47.627505Z",
13      "last_login": null,
14      "is_superuser": false
15    },
16    {
17      "email": "sherin@gmail.com",
18      "username": "sherin@gmail.com",
19      "date_joined": "2023-03-02T13:11:17.835059Z",
20      "last_login": null,
21      "is_superuser": false
22    },
23    {
24      "email": "merin@gmail.com",
25      "username": "merin@gmail.com",
26      "date_joined": "2023-03-02T13:12:29.491414Z",
27      "last_login": null,
28      "is_superuser": false
29    },
30    {
31      "email": "user1@gmail.com",
32      "username": "user1@gmail.com",
33      "date_joined": "2023-03-03T09:24:42.488809Z",
34      "last_login": "2023-03-03T09:33:06.634329Z",
35      "is_superuser": false
```

- **Edit User**
  → Only the admin has permission to edit the user

- **Delete User**
  → Only the admin has permission to edit the user