Overview

The CPSAM Authentication Module provides REST API endpoints for managing user accounts, including listing metadata, creating accounts, and authenticating users. It is built using Django REST Framework (DRF) and integrates with the SiteUser model.

Base URL

`/api/users/`

Endpoints

1. List Users and Metadata

URL: `/api/users/`

Method: GET

Description: Returns metadata about the `SiteUser` model fields, available departments, and staff designations.

Response:

```
{
    "fields": ["id", "username", "email", "first_name", "last_name", "department", "staff_designation", "is_active", "date_joined"],
    "department": [
        {"id": 1, "name": "Computer Science"},
        {"id": 2, "name": "Mechanical"}
    ],
    "staff_designations": [
        {"id": 1, "name": "Professor"},
        {"id": 2, "name": "Assistant"}
    ]
}
```

Notes: - `fields` lists all model fields in `SiteUser`. - `department` lists all available departments. - `staff_designations` lists all staff designations.

---

2. Create User Account

URL: `/api/users/`

Method: POST

Description: Creates a new `SiteUser` account. Password is hashed automatically.

Request Body:

```json
{
    "username": "johndoe",
    "email": "johndoe@example.com",
    "password": "SecurePassword123",
    "first_name": "John",
    "last_name": "Doe",
    "department": 1,
    "staff_designation": 2
}
```

Response:

```json
{
    "id": 5,
    "username": "johndoe",
    "email": "johndoe@example.com",
    "first_name": "John",
    "last_name": "Doe",
    "department": 1,
    "staff_designation": 2,
    "is_active": true
}
```

Notes: - The password is **never returned** in the response. - Fields not included in the request will take default values.

---

3. Authenticate User

URL: `/api/users/authenticate/`

Method: POST

Description: Authenticates a user using username and password.

Request Body:

```
{
    "username": "johndoe",
    "password": "SecurePassword123"
}
```

Successful Response:

```
{
    "id": 5,
    "username": "johndoe",
    "email": "johndoe@example.com",
    "first_name": "John",
    "last_name": "Doe",
    "department": 1,
    "staff_designation": 2,
    "is_active": true
}
```

Failure Response:

```
{
    "success": false,
    "message": "invalid credentials"
}
```

HTTP Status: `403 FORBIDDEN`

Notes: - Password is never returned. - Ensure HTTPS is used to protect credentials in transit.

Models & Serializers Used

- `SiteUser`: Main user model.
- `department`: Stores available departments.
- `staff_designation`: Stores staff titles.
- `AccountSerializer`: Serializes SiteUser data.
- `DepartmentSerializer`: Serializes department data.
- `StaffDesignationSerializer`: Serializes staff designation data.

Authentication & Security

- Passwords are hashed using Django's `make_password`.
- Authentication endpoint uses `authenticate` from `django.contrib.auth`.
- Passwords are **write-only**; never exposed in responses.

Notes

- All endpoints follow standard DRF conventions.
- List endpoint also serves as a metadata API for front-end forms.
- The authenticate endpoint is a **custom action** (`@action`) on the `User_Accounts` ViewSet.

Example Usage (Python Requests)

```
import requests

# List Metadata
resp = requests.get('https://example.com/api/users/')
print(resp.json())

# Create User
resp = requests.post('https://example.com/api/users/', json={
    'username': 'johndoe',
    'email': 'johndoe@example.com',
    'password': 'SecurePassword123'
})
print(resp.json())

# Authenticate User
resp = requests.post('https://example.com/api/users/authenticate/', json={
    'username': 'johndoe',
    'password': 'SecurePassword123'
})
print(resp.json())
```

Change Log

- v1.0: Initial API implementation