

# Postman の使い方

## ① 新しい API テストの作成

- **Create new collection**  
→ 新しい API の接続(テスト用のまとまり)を作成
  - **ADD request**  
→ その API でテストしたいリクエストを追加
- 

## ② リクエストの設定

- **POST / GET** などの HTTP メソッドを選択

**ENTER URL** にテストしたいエンドポイントを入力

`http://localhost:8000/user`

- - 送りたい内容を書く
    - `Body → raw → JSON`
    - JSON 形式で API に送りたいデータを書く
  - **SEND**
    - API にリクエストを送信する
- 

# Postman の認証について (Authorization)

- **Authorization** とは
  - API 側が  
「この送り主は安全かどうか」

を判定するための仕組み

- Postman 側では  
「誰が・どのような方法で送るか？」  
を指定する
- 

## API 側の認証設定 (Django)

- API 側では  
「この人ならアクセスしてよい」  
という設定が必要
  - Django では  
**User** モデルの **username** / **password** が  
そのまま認証に使われる
- 

## Django の認証について

- Django には標準の **User** モデル が存在する

```
from django.contrib.auth.models import User
```

- User モデルにある
    - **username**
    - **password**
  - このデータが **Authorization** ヘッダーに含まれていれば  
→ API の許可が下りる仕組み
- 

## User モデルを拡張したいとき

## ① accounts アプリを作成する

- **User** 管理専用のアプリとして `accounts` を作成
  - その中で **User** モデルを定義・拡張する
- 

## ② **User** モデルを定義・拡張 (例)

```
# accounts/models.py
from django.contrib.auth.models import AbstractUser
from django.db import models

class User(AbstractUser):
    power = models.IntegerField(default=1)
    role = models.CharField(max_length=20, default="user")
```

---

## ③ `settings.py` に追加

```
AUTH_USER_MODEL = 'accounts.User'
```

- これにより Django は  
標準 **User** ではなく、拡張した **User** を使用するようになる
- 

## ④ 他のモデルから **User** を参照するとき

- `settings` から **User** モデルを参照する必要がある

```
from django.conf import settings

class Post(models.Model):
    user = models.ForeignKey(
        settings.AUTH_USER_MODEL,
        on_delete=models.CASCADE
    )
```

---

# マイグレーション

- 各アプリでモデルを定義した後は、以下を実行する

```
python manage.py makemigrations  
python manage.py migrate
```

- これにより  
それぞれのアプリで定義したモデルを一括でマイグレーションできる