



# Python | Extending and customizing django-allauth

Last Updated : 22 May, 2024

Prerequisite: [Django-allauth setup and Configuration](#)

Let's deal with customizing django-allauth signup forms, and intervening in registration flow to add custom processes and validations.

## Extending the Signup Form or adding custom fields in Django-allauth:

One of the most common queries about allauth is about adding additional fields or custom fields to the signup form. You can extend the SignupForm class from allauth.account.forms. All you need to do is create a custom class pass the SignupForm to the custom class and define the custom fields and save it. It's *vital* to return the user object as it will be passed to other modules for validation. You also need to include a variable in settings.py.

Let's see this using an example *forms.py*.

### Python3



```
1 from allauth.account.forms import SignupForm
2 from django import forms
3
4
5 class CustomSignupForm(SignupForm):
6     first_name = forms.CharField(max_length=30, label='First
    Name')
7     last_name = forms.CharField(max_length=30, label='Last
    Name')
8
9     def save(self, request):
10         user = super(CustomSignupForm, self).save(request)
11         user.first_name = self.cleaned_data['first_name']
12         user.last_name = self.cleaned_data['last_name']
```

```
13         user.save()  
14     return user
```

In the above snippet, CustomSignupForm is extended the class which inherits all the features of SignupForm class and adds the necessary features. Here custom fields by the name first\_name and last\_name are created and saved using the signup module in the same class.

The ACCOUNT\_FORMS in settings.py for the above code is

#### Python3



```
1 ACCOUNT_FORMS = {  
2     'signup': 'YourProject.forms.CustomSignupForm',  
3 }
```

Any other custom forms created can be extended in ACCOUNT\_FORMS. The list is illustrated in the documentation.

Similarly, *LoginForm* *UserForm* *AddEmailForm* and others can be extended. However, remember that when you extend these forms and link them in settings.py. Don't forget to pass the original form (For example *SignupForm*) as a parameter to your class and sometimes you might have to handle custom validations and return users or some other value. Please refer to the [source code](#) to follow the correct flow.

### User Intervention and Custom validations:

Let's discuss adding custom validations and intervening in the user registration flow. *DefaultAccountAdapter* is very useful and indeed can be used to solve most of the customization problems that a user might encounter while using *django-allauth*.

#### Example #1: Restricted List of email's

After figuring out a way to store and fetch the restricted list, you can use the adapters and raise validation errors in the registration form when a restricted email tries to register. Extend a *DefaultAccountAdapter* and override the *clean\_email* method. Create an adapter.py in your project directory and extend the default adapter class.

## Python3



```
1 from allauth.account.adapter import DefaultAccountAdapter
2 from django.forms import ValidationError
3
4 class RestrictEmailAdapter(DefaultAccountAdapter):
5     def clean_email(self, email):
6         RestrictedList = ['Your restricted list goes here.']
7         if email in RestrictedList
8             raise ValidationError('You are restricted from
registering.\
9                                     Please
contact admin.')
10         return email
```

Finally, point the account adapter in settings.py to your extended class.

```
ACCOUNT_ADAPTER='YourProject.adapter.RestrictEmailAdapter'
```

### Example #2: Add a Maximum length to a username

As `ACCOUNT_USERNAME_MAX_LENGTH` doesn't exist in the allauth library, *DefaultAccountAdapter* can be used to achieve this feature without much pain. Extend the *DefaultAccountAdapter* class and overriding the *clean\_username* method. You need to also reference the *clean\_username* once again after our custom validation to complete other inbuilt validations.

The last sentence in the above paragraph is the key to work with *DefaultAccountAdapter*. You should never forget to reference the original module name for the module to complete other validations.

## Python3



```
1 from allauth.account.adapter import DefaultAccountAdapter
2 from django.forms import ValidationError
3
4 class UsernameMaxAdapter(DefaultAccountAdapter):
5     def clean_username(self, username):
6         if len(username) > 'Your Max Size':
```

```
7         raise ValidationError('Please enter a username
    value\
8                                     less than the current
    one')
9
10        # For other default validations.
11        return DefaultAccountAdapter.clean_username(self,
    username)
```

Finally, point to the subclass in your settings.py

```
ACCOUNT_ADAPTER = 'YourProject.adapter.UsernameMaxAdapter'
```

You can refer to the *adapters.py* [file](#) in the source code and extend other modules and add a process or change their flow. The modules such as *populate\_username*, *clean\_password* (which can be customized to restrict commonly used passwords) can have the custom process and flow without rewriting them.

*DefaultAccountAdapter* can be a potent tool if used in the right circumstances to intervene in allauth's default process. allauth comes with a vast variety of inbuilt settings, and they are here.

You can also download the entire source code from the link in the references below.

**References:** [stack overflow thread on Example 1](#)

Looking to dive into the world of programming or sharpen your Python skills? Our [Master Python: Complete Beginner to Advanced Course](#) is your ultimate guide to becoming proficient in Python. This course covers everything you need to build a solid foundation from fundamental programming concepts to advanced techniques. With **hands-on projects**, real-world examples, and expert guidance, you'll gain the confidence to tackle complex **coding challenges**. Whether you're starting from scratch or aiming to enhance your skills, this course is the perfect fit. Enroll now and master Python, the language of the future!



5

## Previous Article

[Python | Django Admin Interface](#)

## Next Article

[Django - Dealing with warnings](#)

## Similar Reads

### Creating custom user model API extending AbstractUser in Django

Every new Django project should use a custom user model. The official Django documentation says it is “highly recommended” but I’ll go a step further and say...

4 min read

### Customizing Object Level Permissions - Django REST Framework

In this article, we will discuss how to customize Object Level Permissions in Django REST Framework. To customize permission classes in Django REST...

5 min read

### Customizing Filters in Django REST Framework

Prerequisite: Adding Filtering in APIs – Django REST Framework [link needed article on published yet] Django filters facilitate filtering the queryset to retrieve...

4 min read

### Customizing Phone Number Authentication in Django

As we know, Introducing phone number-based authentication in the Django Admin Panel is a strategic move to modernize and enhance your web...

3 min read

### Extending a list in Python (5 different ways)

In this article, we are going to learn different methods of extending a list in Python. The list is the widely used Python data structure and it can be extended...

3 min read

### Customizing Styles in Matplotlib

Here, we'll delve into the fundamentals of Matplotlib, exploring its various classes and functionalities to help you unleash the full potential of your data...

12 min read

## Customizing Swagger UI

FastAPI is a state-of-the-art, high-performance web framework that uses normal Python type hints to develop APIs with Python 3.7+. FastAPI's ability to...

6 min read

## Adding Tags Using Django-Taggit in Django Project

Django-Taggit is a Django application which is used to add tags to blogs, articles etc. It makes very easy for us to make adding the tags functionality to our django...

2 min read

## How to customize Django forms using Django Widget Tweaks ?

Django forms are a great feature to create usable forms with just few lines of code. But Django doesn't easily let us edit the form for good designs. Here, we...

3 min read

## Styling Django Forms with django-crispy-forms

Django by default doesn't provide any Django form styling method due to which it takes a lot of effort and precious time to beautifully style a form. django-crispy...

1 min read

Article Tags : [Articles](#) [Python](#) [Technical Scripter](#)

Practice Tags : [python](#)



Corporate & Communications Address:-  
A-143, 9th Floor, Sovereign Corporate  
Tower, Sector- 136, Noida, Uttar Pradesh  
(201305) | Registered Address:- K 061,

Tower K, Gulshan Vivante Apartment,  
Sector 137, Noida, Gautam Buddh  
Nagar, Uttar Pradesh, 201305



## Company

About Us  
Legal  
In Media  
Contact Us  
Advertise with us  
GFG Corporate Solution  
Placement Training Program  
GeeksforGeeks Community

## Languages

Python  
Java  
C++  
PHP  
GoLang  
SQL  
R Language  
Android Tutorial  
Tutorials Archive

## DSA

Data Structures  
Algorithms  
DSA for Beginners  
Basic DSA Problems  
DSA Roadmap  
Top 100 DSA Interview Problems  
DSA Roadmap by Sandeep Jain  
All Cheat Sheets

## Data Science & ML

Data Science With Python  
Data Science For Beginner  
Machine Learning  
ML Maths  
Data Visualisation  
Pandas  
NumPy  
NLP  
Deep Learning

## Web Technologies

HTML  
CSS  
JavaScript  
TypeScript  
ReactJS  
NextJS  
Bootstrap  
Web Design

## Python Tutorial

Python Programming Examples  
Python Projects  
Python Tkinter  
Web Scraping  
OpenCV Tutorial  
Python Interview Question  
Django

## Computer Science

Operating Systems  
Computer Network  
Database Management System  
Software Engineering  
Digital Logic Design

## DevOps

Git  
Linux  
AWS  
Docker  
Kubernetes

Engineering Maths  
Software Development  
Software Testing

Azure  
GCP  
DevOps Roadmap

System Design

High Level Design  
Low Level Design  
UML Diagrams  
Interview Guide  
Design Patterns  
OOAD  
System Design Bootcamp  
Interview Questions

Inteview Preparation

Competitive Programming  
Top DS or Algo for CP  
Company-Wise Recruitment Process  
Company-Wise Preparation  
Aptitude Preparation  
Puzzles

School Subjects

Mathematics  
Physics  
Chemistry  
Biology  
Social Science  
English Grammar  
Commerce  
World GK

GeeksforGeeks Videos

DSA  
Python  
Java  
C++  
Web Development  
Data Science  
CS Subjects

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved