**Code file : ch84**

**Video file : 94**

**Update View in Django 5**

# 📘 UpdateView in Django 5 (Class-Based View)

The **UpdateView** is a **generic class-based view** in Django used to update existing model objects.  
It is very similar to CreateView, but instead of creating new records, it **loads an existing record into the form** (using pk or slug) and lets the user update it.

## ✅ Basic Example

### views.py

from django.views.generic.edit import UpdateView

from .models import Student

class StudentUpdateView(UpdateView):

    model = Student

    fields = ['name', 'email', 'password']   # fields we want to update

urls.py

from django.urls import path

from myapp import views

urlpatterns = [

    # pk = primary key of student record to update

    path('update/<int:pk>/', views.StudentUpdateView.as\_view(), name='update'),

]

### Default Template Requirement

By default, UpdateView looks for a template named:

<app\_name>/<model\_name>\_form.html

👉 In this case → student\_form.html

## ✅ Default Template (student\_form.html)

<!DOCTYPE html>

<html>

<head>

    <title>Update Student</title>

</head>

<body>

    <form method="POST">

        {% csrf\_token %}

        {{ form.as\_p }}

        <input type="submit" value="Submit">

    </form>

</body>

</html>

## ✅ Adding Success URL

After updating, Django needs a **redirect URL** (otherwise it won’t know where to go).  
We can provide it using success\_url:

### views.py

class StudentUpdateView(UpdateView):

    model = Student

    fields = ['name', 'email', 'password']

    success\_url = '/thanks/'   # redirect after successful update

urls.py

from django.views.generic.base import TemplateView

urlpatterns = [

    path('update/<int:pk>/', views.StudentUpdateView.as\_view(), name='update'),

    path('thanks/', TemplateView.as\_view(template\_name="myapp/thankyou.html"), name='thanks'),

]

### thankyou.html

<!DOCTYPE html>

<html>

<head>

    <title>Thank You</title>

</head>

<body>

    <h1>Thank you! Student details updated successfully.</h1>

</body>

</html>

## ✅ Custom Template

You don’t have to stick with the default student\_form.html.  
We can use our own:

class StudentUpdateView(UpdateView):

    model = Student

    fields = ['name', 'email', 'password']

    success\_url = '/thanks/'

    template\_name = 'myapp/register.html'   # custom template

## ✅ Customizing Form Without forms.py

If you want to add CSS classes, placeholders, etc. directly:

from django import forms

class StudentUpdateView(UpdateView):

    model = Student

    fields = ['name', 'email', 'password']

    success\_url = '/thanks/'

    template\_name = 'myapp/register.html'

    def get\_form(self):

        form = super().get\_form()

        form.fields['name'].widget = forms.TextInput(attrs={'class': 'myform', 'placeholder': 'Enter Name'})

        form.fields['password'].widget = forms.PasswordInput(attrs={'class': 'mypass'})

        return form

## ✅ Using ModelForm (Recommended Way)

Instead of overriding get\_form, the professional way is to use a **ModelForm**.

### forms.py

from django import forms

from .models import Student

class StudentForm(forms.ModelForm):

    class Meta:

        model = Student

        fields = ['name', 'email', 'password']

        widgets = {

            'name': forms.TextInput(attrs={'class': 'myclass'}),

            'password': forms.PasswordInput(render\_value=True, attrs={'class': 'mypass'})

        }

### views.py

class StudentUpdateView(UpdateView):

    model = Student

    form\_class = StudentForm      # use custom form

    success\_url = '/thanks/'

    template\_name = 'myapp/register.html'

# ⚡ Key Points About UpdateView

1. **Update existing record** (needs pk or slug in URL).
2. **Default template name** → <model>\_form.html.
3. **Requires success\_url or get\_absolute\_url()** for redirect.
4. **Customization options:**
   * Override get\_form() → inline customization.
   * Use forms.py with ModelForm → recommended, cleaner approach.

👉 In short:

* **CreateView** → insert new data.
* **UpdateView** → edit existing data.