**use AJAX requests in your web application.**

**{% load static %}<!doctype html>**

**<html>**

**<head>**

**<meta charset="utf-8">**

**<title>{% block title %}Default Title{% endblock %}</title>**

**<link rel="stylesheet" type="text/css" href="{% static 'css/app.css' %}">**

**{% block stylesheet %}{% endblock %}**

**</head>**

**<body>**

**<header>**

**...**

**</header>**

**<main>**

**{% block content %}**

**{% endblock %}**

**</main>**

**<footer>**

**...**

**</footer>**

**<script src="https://code.jquery.com/jquery-3.1.0.min.js"></script>**

**<script src="{% static 'js/app.js' %}"></script>**

**{% block javascript %}{% endblock %}**

**</body>**

**</html>**

**views.py**

**from django.contrib.auth.forms import UserCreationForm**

**from django.views.generic.edit import CreateView**

**class SignUpView(CreateView):**

**template\_name = 'core/signup.html'**

**form\_class = UserCreationForm**

**urls.py**

**from django.conf.urls import url**

**from core import views**

**urlpatterns = [**

**url(r'^signup/$', views.SignUpView.as\_view(), name='signup'),**

**]**

**signup.html**

**{% extends 'base.html' %}**

**{% block content %}**

**<form method="post">**

**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button type="submit">Sign up</button>**

**</form>**

**{% endblock %}**

**<input type="text" required="" name="username" maxlength="150" id="id\_username" autofocus="">**

**{% extends 'base.html' %}**

**{% block javascript %}**

**<script>**

**$("#id\_username").change(function () {**

**var username = $(this).val();**

**$.ajax({**

**url: '/ajax/validate\_username/',**

**data: {**

**'username': username**

**},**

**dataType: 'json',**

**success: function (data) {**

**if (data.is\_taken) {**

**alert("A user with this username already exists.");**

**}**

**}**

**});**

**});**

**</script>**

**{% endblock %}**

**{% block content %}**

**<form method="post">**

**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button type="submit">Sign up</button>**

**</form>**

**{% endblock %}**

**views.py**

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

**from django.http import JsonResponse**

**def validate\_username(request):**

**username = request.GET.get('username', None)**

**data = {**

**'is\_taken': User.objects.filter(username\_\_iexact=username).exists()**

**}**

**return JsonResponse(data)**

**Add a route to this view: urls.py**

**from django.conf.urls import url**

**from core import views**

**urlpatterns = [**

**url(r'^signup/$', views.SignUpView.as\_view(), name='signup'),**

**url(r'^ajax/validate\_username/$', views.validate\_username, name='validate\_username'),**

**]**

**And then the simpliest implementation would be: signup.html**

**{% extends 'base.html' %}**

**{% block javascript %}**

**<script>**

**$("#id\_username").change(function () {**

**var username = $(this).val();**

**$.ajax({**

**url: '/ajax/validate\_username/',**

**data: {**

**'username': username**

**},**

**dataType: 'json',**

**success: function (data) {**

**if (data.is\_taken) {**

**alert("A user with this username already exists.");**

**}**

**}**

**});**

**});**

**</script>**

**{% endblock %}**

**{% block content %}**

**<form method="post">**

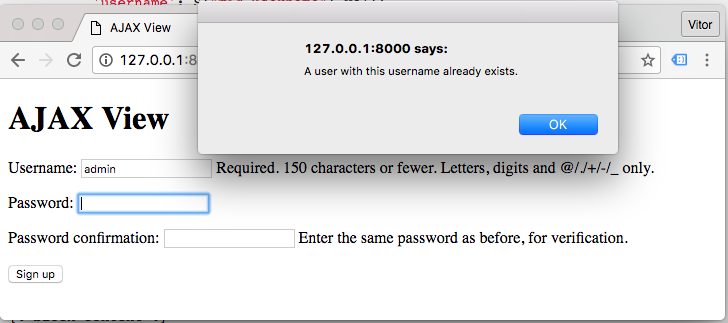
**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button type="submit">Sign up</button>**

**</form>**

**{% endblock %}**



Improving the Example

The example is working, and that’s great. But there are a few details we can improve:

* Hardcoded URL inside the Ajax function
* User message embeded in the JavaScript

Since we are inside a Django template, we could do something like that:

**$.ajax({**

**url: '{% url "validate\_username" %}',**

**...**

**});**

It is already better, but, I don’t really like this strategy. Two reasons: You won’t be able to extract this JavaScript code to a external file. We have an external script modifying our JavaScript. If you find yourself writing code to write a JavaScript code, give it another thought. There might be a better solution.

Now, what I usually like to do:

**views.py**

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

**from django.http import JsonResponse**

**def validate\_username(request):**

**username = request.GET.get('username', None)**

**data = {**

**'is\_taken': User.objects.filter(username\_\_iexact=username).exists()**

**}**

**if data['is\_taken']:**

**data['error\_message'] = 'A user with this username already exists.'**

**return JsonResponse(data)**

**Extract the user message to the Python code, this way is easier to work with translations.**

**signup.html**

**{% extends 'base.html' %}**

**{% block javascript %}**

**<script>**

**$("#id\_username").change(function () {**

**var form = $(this).closest("form");**

**$.ajax({**

**url: form.attr("data-validate-username-url"),**

**data: form.serialize(),**

**dataType: 'json',**

**success: function (data) {**

**if (data.is\_taken) {**

**alert(data.error\_message);**

**}**

**}**

**});**

**});**

**</script>**

**{% endblock %}**

**{% block content %}**

**<form method="post" data-validate-username-url="{% url 'validate\_username' %}">**

**{% csrf\_token %}**

**{{ form.as\_p }}**

**<button type="submit">Sign up</button>**

**</form>**

**{% endblock %}**

**Ideally I would put the data-validate-username-url attribute directly in the username field. But here we would need to expose all the fields in the {{ form.as\_p }}. So in this case I can live with the attribute in the form tag.**

Generally speaking: try to avoid modifying JavaScript code with Python code, keep the URL references in the HTML and manage user messages in the Python code.

Another tip: if you have direct access to the HTML field, prefer adding a class name like this:

**<input type="text" name="username" id="id\_username" class="js-validate-username">**

**And then you hook the change event to the class js-validate-username instead. The js- prefix suggests that there is a JavaScript code that interacts with this element. And then use this prefix for JavaScript only, never use it for styling the component as well via css.**