Question on the CRUD operations in Django

You are building a blog application. You need to create a model `BlogPost` with the following fields:

- `title` (CharField)
- `content` (TextField)
- `published_date` (DateTimeField)

Task:

1. Define the model `BlogPost`.
2. Create a Django form to add new blog posts.
3. Implement a view to handle form submission and save blog posts.

models.py

```
from django.db import models

class BlogPost(models.Model):
    title = models.CharField(max_length=200)
    content = models.TextField()
    published_date = models.DateTimeField(auto_now_add=True)

    def __str__(self):
        return self.title
```

forms.py

```
from django import forms
from .models import BlogPost

class BlogPostForm(forms.ModelForm):
    class Meta:
        model = BlogPost
        fields = ['title', ‘content']
```

views.py

```
from django.shortcuts import render, redirect
from .forms import BlogPostForm

def add_blog_post(request):
    if request.method == "POST":
        form = BlogPostForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('blog_list')
    else:
        form = BlogPostForm()
    return render(request, 'add_blog.html', {'form': form})
```

urls.py

```
from django.urls import path
from .views import add_blog_post

urlpatterns = [
    path('add/', add_blog_post, name='add_blog'),
]
```

add_blog.html

```
<h2>Add New Blog Post</h2>
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Publish</button>
</form>
```

Question 2 You need to create a view that retrieves and displays all blog posts from the database.

Task:

1. Implement a view to fetch all blog posts.
2. Display them in an HTML template.

views.py

```
from django.shortcuts import render
from .models import BlogPost

def blog_list(request):
    blogs = BlogPost.objects.all()
    return render(request, 'blog_list.html', {'blogs': blogs})
```

urls.py

```
urlpatterns += [
    path('', blog_list, name='blog_list'),
]
```

blog_list.html

```
<h2>Blog Posts</h2>
<ul>
    {% for blog in blogs %}
        <li>
            <strong>{{ blog.title }}</strong><br>
            {{ blog.content|truncatewords:20 }} <br>
            <a href="{% url 'update_blog' blog.id %}">Edit</a>
            <a href="{% url 'delete_blog' blog.id %}">Delete</a>
        </li>
    {% endfor %}
</ul>
<a href="{% url 'add_blog' %}">Add New Blog</a>
```

Question : Update a Blog Post

A user wants to edit a blog post.

Task:

1. Create a view that fetches an existing blog post.
2. Use Django forms to allow the user to edit the content.
3. Save the updated blog post back to the database.

views.py

```
from django.shortcuts import get_object_or_404

def update_blog_post(request, post_id):
    blog = get_object_or_404(BlogPost, id=post_id)
    if request.method == "POST":
        form = BlogPostForm(request.POST, instance=blog)
        if form.is_valid():
            form.save()
            return redirect('blog_list')
    else:
        form = BlogPostForm(instance=blog)
    return render(request, 'update_blog.html', {'form': form})
```

urls.py

```
urlpatterns += [
    path('update/<int:post_id>/', update_blog_post, name='update_blog'),
]
```

update_blog.html

```
<h2>Edit Blog Post</h2>
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Save Changes</button>
</form>
```

Question Delete a Blog Post

Users should be able to delete blog posts.

Task:

1. Create a view that allows users to delete a post.
2. Ask for confirmation before deleting.

views.py

```
def delete_blog_post(request, post_id):
    blog = get_object_or_404(BlogPost, id=post_id)
    if request.method == "POST":
        blog.delete()
        return redirect('blog_list')
    return render(request, 'confirm_delete.html', {'blog': blog})
```

urls.py

```
urlpatterns += [
    path('delete/<int:post_id>/', delete_blog_post, name='delete_blog'),
]
```

confirm_delete.html

```
<h2>Are you sure you want to delete "{{ blog.title }}"?</h2>
<form method="post">
    {% csrf_token %}
    <button type="submit">Yes, Delete</button>
    <a href="{% url 'blog_list' %}">Cancel</a>
</form>
```

Question : Search for Blog Posts

Users want to search for blog posts based on the title.

Task:

1. Implement a search functionality.
2. Allow users to enter a search term and display matching posts.

views.py

```
def search_blog(request):
    query = request.GET.get('q')
    blogs = BlogPost.objects.filter(title__icontains=query) if query else BlogPost.objects.all()
    return render(request, 'blog_list.html', {'blogs': blogs, 'query': query})
```

urls.py

```
urlpatterns += [
    path('search/', search_blog, name='search_blog'),
]
```

blog_list.html (Modify to include search form)

```
<h2>Blog Posts</h2>
<form method="get" action="{% url 'search_blog' %}">
    <input type="text" name="q" placeholder="Search blog posts" value="{{ query }}">
    <button type="submit">Search</button>
</form>

<ul>
    {% for blog in blogs %}
        <li>{{ blog.title }} - {{ blog.published_date }}</li>
    {% endfor %}
</ul>
```