Here's a simple CRUD example using React (with Axios) as frontend and Django (with Django REST Framework) as backend. This example focuses only on a single model, like Post, and shows the minimal setup to perform Create, Read, Update, and Delete operations.

🛠️ Backend: Django + Django REST Framework

1. Install Requirements

```
pip install django djangorestframework
```

2. Set Up Django Project and App

```
django-admin startproject backend
cd backend
python manage.py startapp api
```

3. Configure settings.py

```
INSTALLED_APPS = [
    ...
    'rest_framework',
    'api',
]
```

4. Create a Model

api/models.py

```
from django.db import models

class Post(models.Model):
    title = models.CharField(max_length=100)
    content = models.TextField()
```

5. Create Serializer

api/serializers.py

```
from rest_framework import serializers
from .models import Post

class PostSerializer(serializers.ModelSerializer):
    class Meta:
        model = Post
        fields = '__all__'
```

6. Create API View

api/views.py

```
from rest_framework import viewsets
from .models import Post
from .serializers import PostSerializer

class PostViewSet(viewsets.ModelViewSet):
    queryset = Post.objects.all()
    serializer_class = PostSerializer
```

7. Set Up URLs

```
# api/urls.py

from django.urls import path, include
from rest_framework.routers import DefaultRouter
from .views import PostViewSet

router = DefaultRouter()
router.register(r'posts', PostViewSet)

urlpatterns = [
    path('', include(router.urls)),
]
```

```
# backend/urls.py

from django.contrib import admin
from django.urls import path, include
```

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('api/', include('api.urls')),
]
```

8. Run Migrations and Start Server

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

Now your API is available at http://localhost:8000/api/posts/.

💻 Frontend: React + Axios

1. Set Up React Project

```
npx create-react-app frontend
cd frontend
npm install axios
```

2. Basic Axios CRUD Component

```
// src/App.js

import React, { useEffect, useState } from 'react';
```

```
import axios from 'axios';

const API_URL = 'http://localhost:8000/api/posts/';

function App() {
    const [posts, setPosts] = useState([]);
    const [title, setTitle] = useState('');
    const [content, setContent] = useState('');
    const [editId, setEditId] = useState(null);

    useEffect(() => {
        fetchPosts();
    }, []);

    const fetchPosts = async () => {
        const res = await axios.get(API_URL);
        setPosts(res.data);
    };

    const createPost = async () => {
        await axios.post(API_URL, { title, content });
        setTitle('');
        setContent('');
        fetchPosts();
    };
```

```
const updatePost = async () => {
    await axios.put(`${API_URL}${editId}/`, { title, content });
    setTitle('');
    setContent('');
    setEditId(null);
    fetchPosts();
};

const deletePost = async (id) => {
    await axios.delete(`${API_URL}${id}/`);
    fetchPosts();
};

const editPost = (post) => {
    setEditId(post.id);
    setTitle(post.title);
    setContent(post.content);
};

return (
    <div>
        <h1>Simple CRUD</h1>
        <input value={title} onChange={(e) => setTitle(e.target.value)} placeholder="Title" />
        <input value={content} onChange={(e) => setContent(e.target.value)} placeholder="Content" />
        <button onClick={editId ? updatePost : createPost}>
            {editId ? 'Update' : 'Create'}
```

```
            </button>
            <ul>
                {posts.map((post) => (
                    <li key={post.id}>
                        <strong>{post.title}</strong>: {post.content}
                        <button onClick={() => editPost(post)}>Edit</button>
                        <button onClick={() => deletePost(post.id)}>Delete</button>
                    </li>
                ))}
            </ul>
        </div>
    );
}

export default App;
```

✅ Summary

Operation	Method	URL	Axios Call
Create	POST	/api/posts/	axios.post()
Read	GET	/api/posts/	axios.get()
Update	PUT	/api/posts/<id>/	axios.put()
Delete	DELETE	/api/posts/<id>/	axios.delete()

Let's break down the React + Axios CRUD example in detailed explanation, section by section, so you fully understand what each part does and how it connects to the Django backend.

🧠 Overview

This app does Create, Read, Update, and Delete (CRUD) operations on a list of "posts" (each with title and content). It connects to a Django backend using the Django REST Framework.

🔍 File: App.js

✅ 1. Importing React and Axios

```
import React, { useEffect, useState } from 'react';
import axios from 'axios';
```

- useState – lets you manage local component state.
- useEffect – runs code after the component renders (like loading data).
- axios – HTTP client used to send requests to your Django backend.

🌍 2. Backend API URL

```
const API_URL = 'http://localhost:8000/api/posts/';
```

This is the base URL for your Django backend API. It maps to:

GET /api/posts/ → List all posts

POST /api/posts/ → Create a new post

PUT /api/posts/1/ → Update post with ID 1

DELETE /api/posts/1/ → Delete post with ID 1

🛠️ 3. Component State Variables

```
const [posts, setPosts] = useState([]);
const [title, setTitle] = useState('');
const [content, setContent] = useState('');
const [editId, setEditId] = useState(null);
```

These manage the UI state:

posts – an array of all posts (fetched from backend).

title & content – input fields to create/update a post.

editId – stores the ID of the post being edited. If null, a new post is being created.

🚀 4. Fetching Posts (Read)

```
useEffect(() => {
  fetchPosts();
}, []);
```

This runs once when the component loads.

It calls fetchPosts() which sends a GET request.

```
const fetchPosts = async () => {
    const res = await axios.get(API_URL);
    setPosts(res.data);
};
```

axios.get(API_URL) fetches the list of posts.

setPosts() updates the state so they show up in the UI.

✍️ 5. Creating a New Post

```
const createPost = async () => {
    await axios.post(API_URL, { title, content });
    setTitle('');
    setContent('');
    fetchPosts();
};
```

Sends a POST request with { title, content } as data.

After creating, it clears the input fields and reloads the post list.

✏️ 6. Editing a Post (Update)

```
const editPost = (post) => {
    setEditId(post.id);
    setTitle(post.title);
    setContent(post.content);
};
```

When user clicks "Edit", this populates the inputs with the post's current data.

It also sets editId so the app knows we're updating, not creating.

```
const updatePost = async () => {
    await axios.put(`${API_URL}${editId}/`, { title, content });
    setTitle('');
    setContent('');
    setEditId(null);
    fetchPosts();
};
```

Sends a PUT request to update the post.

After update, it resets the inputs and clears editId.

❌ 7. Deleting a Post

```
const deletePost = async (id) => {
    await axios.delete(`${API_URL}${id}/`);
    fetchPosts();
};
```

Sends a DELETE request to /posts/<id>/.

After deletion, it refreshes the list.

🧩 8. Rendering the UI

```
return (
  <div>
    <h1>Simple CRUD</h1>
    <input value={title} onChange={(e) => setTitle(e.target.value)} placeholder="Title" />
    <input value={content} onChange={(e) => setContent(e.target.value)} placeholder="Content" />
    <button onClick={editId ? updatePost : createPost}>
      {editId ? 'Update' : 'Create'}
    </button>
    <ul>
      {posts.map((post) => (
        <li key={post.id}>
          <strong>{post.title}</strong>: {post.content}
          <button onClick={() => editPost(post)}>Edit</button>
          <button onClick={() => deletePost(post.id)}>Delete</button>
        </li>
      ))}
    </ul>
  </div>
);
```

- Input fields for title and content.
- Button toggles between "Create" and "Update" based on editId.
- Lists all posts with "Edit" and "Delete" buttons.

🔁 React-Axios-Django Communication Summary

Action	HTTP Verb	URL	Description
List Posts	GET	/api/posts/	Load all posts
Add Post	POST	/api/posts/	Add new post
Edit Post	PUT	/api/posts/<id>/	Update existing post
Delete	DELETE	/api/posts/<id>/	Delete a post

🧪 Test Flow

Run Django backend: python manage.py runserver

Run React frontend: npm start

Go to http://localhost:3000

Create, edit, or delete posts using the UI.