2. API Integration with Error Handling

jsx

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

// Custom hook for API calls
function useApiCall(url, options = {}) {
  const [data, setData] = useState(null);
  const [loading, setLoading] = useState(false);
  const [error, setError] = useState(null);

  const executeCall = async (customUrl = url, customOptions =
options) => {
    try {
      setLoading(true);
      setError(null);

      const response = await fetch(customUrl, {
        ...customOptions,
        headers: {
          'Content-Type': 'application/json',
          ...customOptions.headers
        }
      });

      if (!response.ok) {
        throw new Error(`HTTP ${response.status}:
${response.statusText}`);
      }

      const result = await response.json();
      setData(result);
      return result;
    } catch (err) {
      setError(err.message);
      throw err;
    } finally {
      setLoading(false);
    }
  };

  return { data, loading, error, executeCall };
}

// Main component
function UserManagement() {
  const [users, setUsers] = useState([]);
  const { data, loading, error, executeCall } = useApiCall();
  // Load users on mount
  useEffect(() => {
    loadUsers();
  }, []);

  const loadUsers = async () => {
    try {
      const userData = await
executeCall('https://jsonplaceholder.typicode.com/users');
      setUsers(userData);
    } catch (err) {
      console.error('Failed to load users:', err);
    }
  };

  const createUser = async (userData) => {
    try {
      const newUser = await
executeCall('https://jsonplaceholder.typicode.com/users', {
        method: 'POST',
        body: JSON.stringify(userData)
      });
      setUsers([...users, { ...newUser, id: Date.now() }]);
    } catch (err) {
      console.error('Failed to create user:', err);
    }
  };

  const updateUser = async (id, userData) => {
    try {
      const updatedUser = await
executeCall(`https://jsonplaceholder.typicode.com/users/${id}`, {
        method: 'PUT',
        body: JSON.stringify(userData)
      });
      setUsers(users.map(user => user.id === id ? {
...updatedUser, id } : user));
    } catch (err) {
      console.error('Failed to update user:', err);
    }
  };

  const deleteUser = async (id) => {
    try {
      await
executeCall(`https://jsonplaceholder.typicode.com/users/${id}`, {
        method: 'DELETE'
      });
      setUsers(users.filter(user => user.id !== id));
    } catch (err) {
      console.error('Failed to delete user:', err);
    }
  };

  if (loading && users.length === 0) {
    return <div className="loading">Loading users...</div>;
  }

  return (
    <div className="user-management">
      <h1>User Management</h1>

      {error && (
        <div className="error-banner">
          <p>Error: {error}</p>
          <button onClick={loadUsers}>Retry</button>
        </div>
      )}

      <UserForm onSubmit={createUser} />

      <div className="user-list">
        {users.map(user => (
          <UserCard
            key={user.id}
            user={user}
            onUpdate={updateUser}
            onDelete={deleteUser}
          />
        ))}
      </div>
    </div>
  );
}

function UserForm({ onSubmit, initialData = {} }) {
  const [formData, setFormData] = useState({
    name: '',
    email: '',
    phone: '',
    ...initialData
  });

  const handleSubmit = (e) => {
    e.preventDefault();
    onSubmit(formData);
    setFormData({ name: '', email: '', phone: '' });
  };

  return (
    <form onSubmit={handleSubmit} className="user-form">
      <input
        type="text"
        placeholder="Name"
        value={formData.name}
        onChange={(e) => setFormData({ ...formData, name:
e.target.value })}
        required
      />
      <input
        type="email"
        placeholder="Email"
        value={formData.email}
        onChange={(e) => setFormData({ ...formData, email:
e.target.value })}
        required
      />
      <input
        type="tel"
        placeholder="Phone"
        value={formData.phone}
        onChange={(e) => setFormData({ ...formData, phone:
e.target.value })}
        required
      />
      <button type="submit">Add User</button>
    </form>
  );
}

export default UserManagement;
```