Experiment - 3

Student Name: Archita Srivastava
UID: 23BCS12459
Branch: BE-CSE
Section/Group: KRG-2B
Semester: 5th
Date of Performance: 11/9/25
Subject Name: Full Stack- I
Subject Code: 23CSP-339

Aim: To build an interactive library management interface using React components with full CRUD (Create, Read, Update, Delete) functionality.

Objective: The main objective is to-

1. Design a book listing component.
2. Implement search functionality.
3. Add a form for new book entries.
4. Enable update and delete capabilities for each book.
5. Manage state using React hooks.

Hardware/Software Requirements:

1. Processor: Intel i5/Ryzen 5 or higher
2. RAM: 8GB minimum.
3. Display: 1920x1080 resolution.
4. Node.js v18+
5. React.js v18+
6. VS code with ES7 + extensions.
7. JSON server(for mock PIs).

About the Experiment -

This experiment demonstrates how to build a dynamic and responsive Library Management System using React.

Concepts covered-

1. Component-based architecture.
2. State management with hooks(useState, useEffect).
3. Controlled forms and event handling.
4. Conditional rendering.
5. RESTful API interaction with fetch.

Code implementation -

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


function App() { const [books, setBooks] = useState([]); const
 [formData, setFormData] = useState({ title: '', author: '' });
 const [searchTerm, setSearchTerm] = useState(''); const
 [editingBookId, setEditingBookId] = useState(null);


 // Fetch initial books from JSON Server
 useEffect(() => {
 fetch('http://localhost:3001/books')
    .then(res => res.json())
    .then(data => setBooks(data));
 }, []);


 // Handle form input change
 const handleChange = e => {
  setFormData({ ...formData, [e.target.name]: e.target.value });
 };
```

```
// Handle Add / Update book
const handleSubmit = e => {
e.preventDefault(); if (editingBookId) { // Update book
fetch(`http://localhost:3001/books/${editingBookId}`, {
method: 'PUT', headers: { 'Content-Type':
'application/json' }, body: JSON.stringify(formData),
})
.then(res => res.json())
.then(updatedBook => { setBooks(books.map(book => (book.id ===
editingBookId ? updatedBook :
book))); setEditingBookId(null);
setFormData({ title: '', author: ''
});
});
} else {
// Add new book
fetch('http://localhost:3001/books', { method:
'POST', headers: { 'Content-Type':
'application/json' }, body:
JSON.stringify(formData),
```

```
    })
      .then(res => res.json())
      .then(newBook => {
        setBooks([...books, newBook]);
        setFormData({ title: '', author: '' });
      });
  }
};

// Edit book const handleEdit
= book => {
  setEditingBookId(book.id); setFormData({ title:
  book.title, author: book.author });
};

// Delete book
const handleDelete = id => {
  fetch(`http://localhost:3001/books/${id}`, {
    method: 'DELETE',
```

```
  }).then(() => { setBooks(books.filter(book =>
    book.id !== id));
  });
};

// Filtered books for search const filteredBooks =
books.filter(book =>
book.title.toLowerCase().includes(searchTerm.toLowerCase())
);

return (
  <div style={{ padding: '20px' }}>
    <h2>Library Management</h2>

    {/* Add / Update Book Form */}
    <form onSubmit={handleSubmit}>
      <input
        name="title"
        placeholder="Title"
        value={formData.title}
```

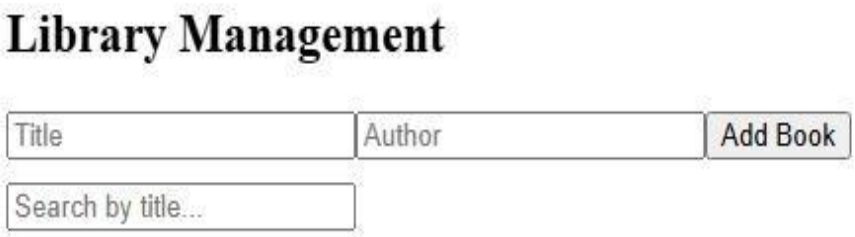
```
    onChange={handleChange}
    required
  />
  <input name="author"
    placeholder="Author"
    value={formData.author}
    onChange={handleChange
    } required
  />
  <button type="submit">{editingBookId ? 'Update' : 'Add'} Book</button>
</form>

{/* Search Bar */}
<input placeholder="Search by
  title..." value={searchTerm}
  onChange={e =>
  setSearchTerm(e.target.value)}
  style={{ marginTop: '10px' }}
/>
```

```
      {/* Book List */}
      <ul>
        {filteredBooks.map(book => (
          <li key={book.id}>
            <strong>{book.title}</strong> by {book.author}
            <button onClick={() => handleEdit(book)}>Edit</button>
            <button onClick={() => handleDelete(book.id)}>Delete</button>
          </li>
        ))}
      </ul>
    </div>
  );
}

export default App;
```

Output:


Library Management
Title
Author
Add Book
Search by title...