NAME	SHASANK SUMAN
UID	23BCS12489
CLASS	622-A

- REACT PRACTISE 1

Practice 1 – React : Title ProductCard Component Using Props

Objective –

Build a reusable React component that displays product details using props. This task helps you understand how to pass and render dynamic data in React components.

Task Description –

Create a React component named ProductCard that accepts props for product name, price, and stock status. The component should display all three details clearly, for example in a card layout or a simple styled box. You should demonstrate how different product data can be passed into the component and rendered dynamically without changing the component code.

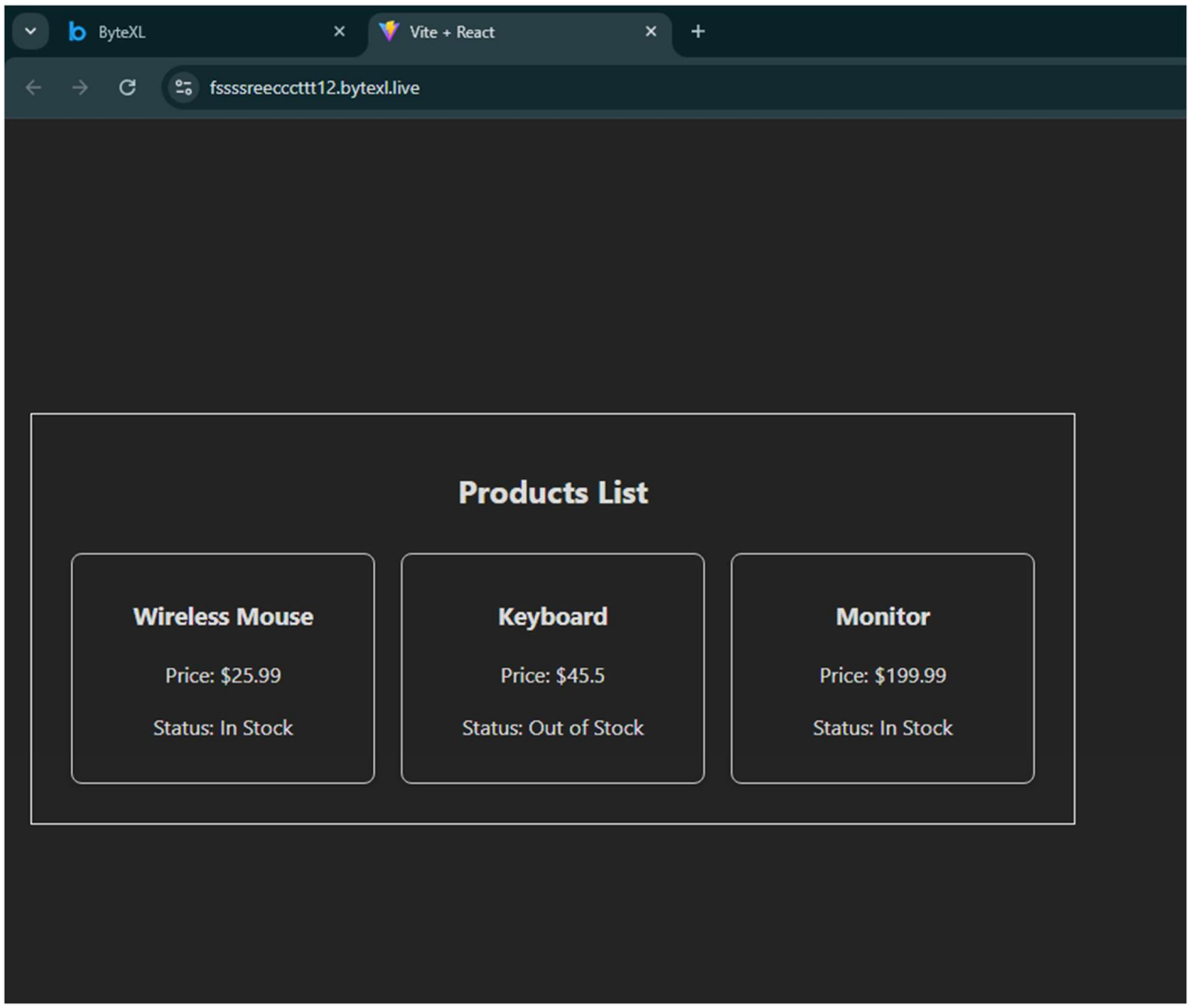
- CODE

```
import React from "react";

const ProductCard = ({ name, price, status }) => {
  return (
    <div style={styles.card}>
      <h3 style={{ fontWeight: "bold" }}>{name}</h3>
      <p>Price: ${price}</p>
      <p>Status: {status}</p>
    </div>
  );
};

const styles = {
  card: {
    border: "1px solid #ddd",
    borderRadius: "8px",
    padding: "15px",
    margin: "10px",
    width: "200px",
    textAlign: "center",
    boxShadow: "0px 2px 5px rgba(0,0,0,0.1)",
  },
};

export default ProductCard
```

```
import React from "react";
import ProductCard from "./productcard";

function App() {
  return (
    <div style={styles.container}>
      <h2 style={{ textAlign: "center" }}>Products List</h2>
      <div style={styles.list}>
        <ProductCard name="Wireless Mouse" price="25.99" status="In Stock"
/>
        <ProductCard name="Keyboard" price="45.5" status="Out of Stock" />
        <ProductCard name="Monitor" price="199.99" status="In Stock" />
      </div>
    </div>
  );
}

const styles = {
  container: {
    border: "1px solid white",
    padding: "20px",
    margin: "20px",
  },
  list: {
    display: "flex",
    justifyContent: "space-around",
  },
};

export default App;
```

- OUTPUT

DEPLOY LINK :- https://fssssreeccctttt12.bytexl.live/



- REACT PRACTISE 2

Title

Library Management UI with Search, Add, and Remove Book Functionality

Objective

Build an interactive library management interface using React that allows users to view a list of books and perform actions like searching, adding, and removing books. This task strengthens your understanding of state management, event handling, and dynamic rendering in React.

Task Description

Create a React-based UI that displays a list of books (each with a title and author). Add a search input box to filter books by title or author as the user types. Provide an input form to add new books to the list, and include a remove button next to each book to delete it from the list. All actions (searching, adding, removing) should work dynamically without reloading the page, demonstrating React's ability to update the UI based on state changes.

- CODE

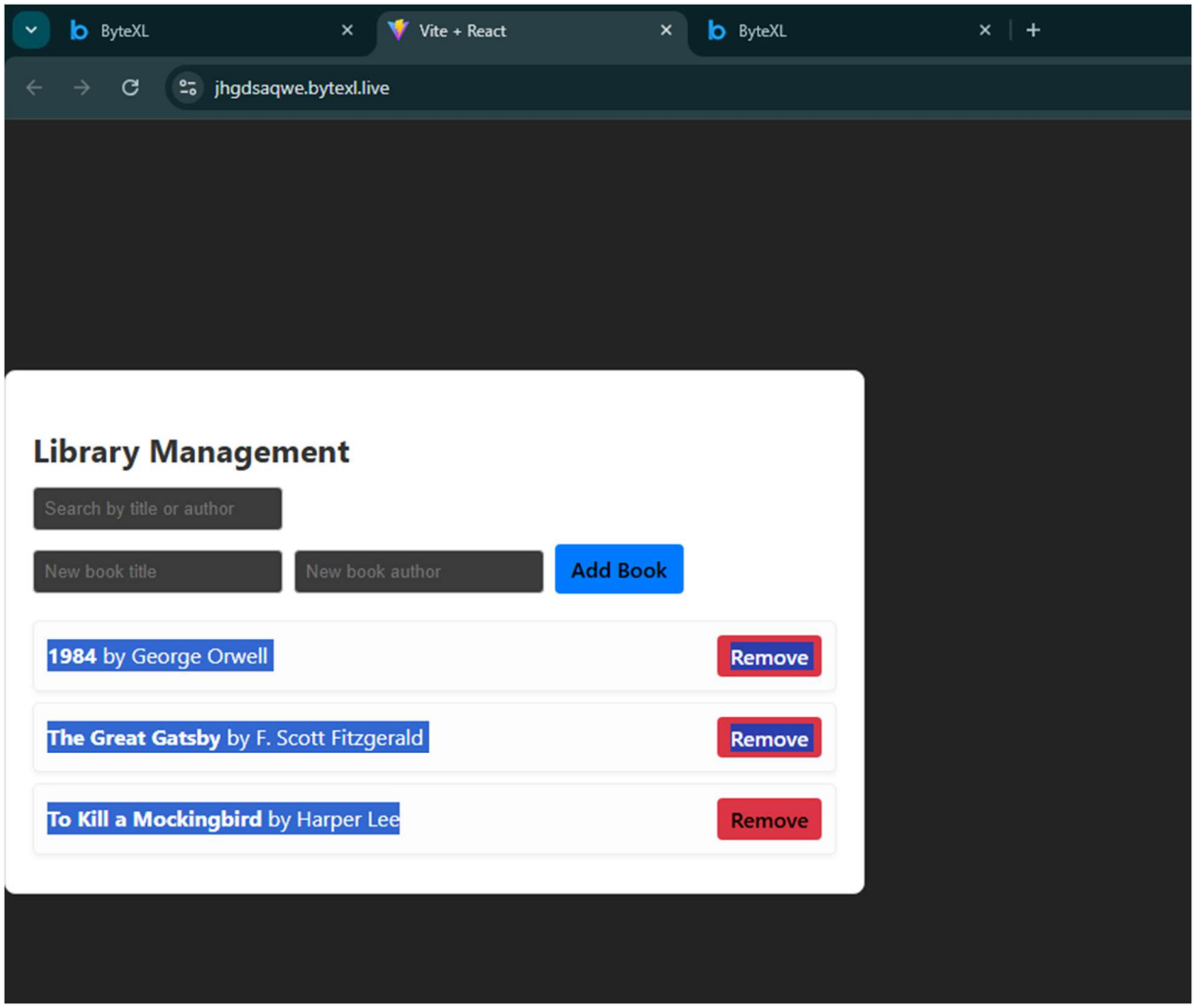
```
import React, { useState } from "react";

function App() {
  const [books, setBooks] = useState([
    { title: "1984", author: "George Orwell" },
    { title: "The Great Gatsby", author: "F. Scott Fitzgerald" },
    { title: "To Kill a Mockingbird", author: "Harper Lee" },
  ]);

  const [search, setSearch] = useState("");
  const [newTitle, setNewTitle] = useState("");
  const [newAuthor, setNewAuthor] = useState("");

  const addBook = () => {
 if (newTitle.trim() && newAuthor.trim()) {
setBooks([...books, { title: newTitle, author: newAuthor }]);
      setNewTitle("");
```

```
    setNewAuthor("");
  }
};

const removeBook = (index) => {
  setBooks(books.filter((_, i) => i !== index));
};

const filteredBooks = books.filter(
  (book) =>
    book.title.toLowerCase().includes(search.toLowerCase()) ||
    book.author.toLowerCase().includes(search.toLowerCase())
);

return (
  <div style={styles.container}>
    <h2 style={styles.heading}>Library Management</h2>

    {/* Search */}
    <input
      type="text"
      placeholder="Search by title or author"
      value={search}
      onChange={(e) => setSearch(e.target.value)}
      style={styles.input}
    />

    {/* Add Book */}
    <div style={{ marginTop: "10px" }}>
      <input
        type="text"
        placeholder="New book title"
        value={newTitle}
        onChange={(e) => setNewTitle(e.target.value)}
        style={styles.input}
      />
      <input
        type="text"
        placeholder="New book author"
        value={newAuthor}
        onChange={(e) => setNewAuthor(e.target.value)}
        style={styles.input}
      />
      <button onClick={addBook} style={styles.button}>
        Add Book
```

```
        </button>
      </div>

      {/* Book List */}
      <div style={{ marginTop: "20px" }}>
        {filteredBooks.map((book, index) => (
          <div key={index} style={styles.bookItem}>
            <span>
              <strong>{book.title}</strong> by {book.author}
            </span>
            <button onClick={() => removeBook(index)}
 style={styles.removeButton}>
              Remove
            </button>
          </div>
        ))}
      </div>
    </div>
  );
}


const styles = {
  container: {
    backgroundColor: "#ffffff",
    border: "1px solid #ddd",
    borderRadius: "8px",
    padding: "20px",
    margin: "20px auto",
    width: "600px",
    boxShadow: "0px 4px 10px rgba(0,0,0,0.05)",
  },
heading: {
  marginBottom: "10px",
    color: "#333",
  },
  input: {
 padding: "8px",
    marginRight: "8px",
    border: "1px solid #ccc",
    borderRadius: "4px",
  },
  button: {
    padding: "8px 12px",
    backgroundColor: "#007bff",
```

```
    color: "Black",
    border: "none",
    borderRadius: "4px",
    cursor: "pointer",
  },
  bookItem: {
    display: "flex",
    justifyContent: "space-between",
    alignItems: "center",
    border: "1px solid #eee",
    padding: "10px",
    marginBottom: "8px",
    borderRadius: "5px",
    backgroundColor: "#fdfdfd",
    boxShadow: "0px 2px 4px rgba(0,0,0,0.05)",
  },
  removeButton: {
    padding: "5px 10px",
    backgroundColor: "#dc3545",
    color: "Black",
    border: "none",
    borderRadius: "4px",
    cursor: "pointer",
  },
};

export default App;
```

- OUTPUT
  DEPLOY LINK - https://jhgdsaqwe.bytexl.live/


ByteXL
Vite + React
ByteXL
jhgdsaqwe.bytexl.live
Library Management
Search by title or author
New book title
New book author
Add Book
1984 by George Orwell
Remove
The Great Gatsby by F. Scott Fitzgerald
Remove
To Kill a Mockingbird by Harper Lee
Remove

- REACT PRACTISE 3

Title –
Person Class Hierarchy with Student and Teacher Subclasses.

Objective -
Understand and apply the concept of inheritance in JavaScript (ES6 classes) by creating a base class and extending it into specialized subclasses. This helps build strong foundational skills in object-oriented programming within a modern JavaScript context.

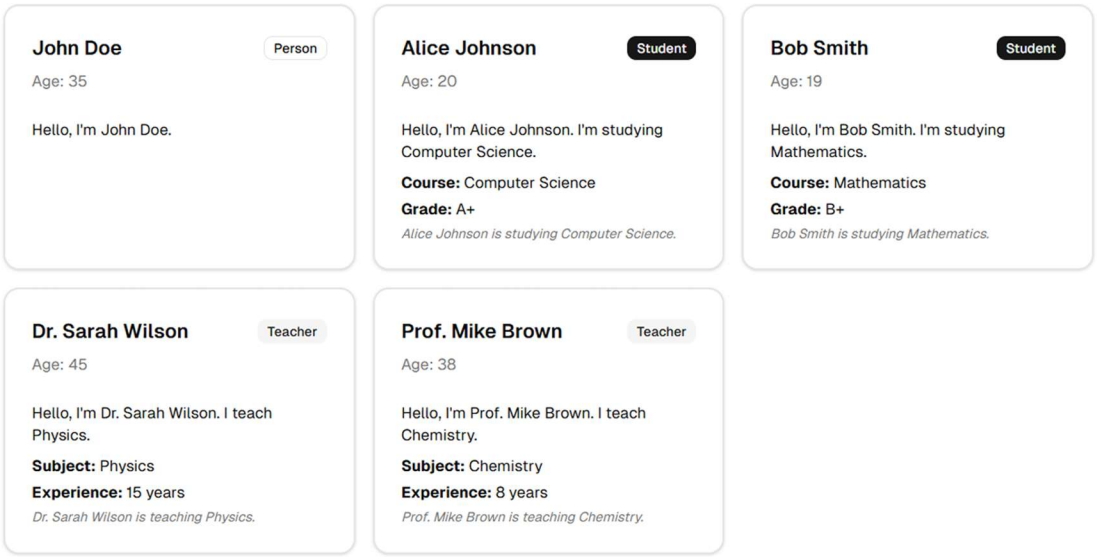
Task Description –
Create a base Person class that has properties like name and age, and a method to display basic information. Then, create two subclasses: Student and Teacher, each extending Person. The Student class should include an additional property like grade or course, and the Teacher class should include a property like subject or department. Each subclass should override or extend methods as needed to display complete details. Finally, create instances of both subclasses and demonstrate calling their methods to show how inheritance and method overriding work.

- CODE

```
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }

  getInfo() {
    return `${this.name} is ${this.age} years old.`;
  }
}
```

```
class Student extends Person {
  constructor(name, age, course) {
    super(name, age); // call parent constructor
    this.course = course;
  }

  getInfo() {
    return `${this.name} is ${this.age} years old and
studies ${this.course}.`;
  }
}
class Teacher extends Person {
  constructor(name, age, subject) {
    super(name, age);
    this.subject = subject;
  }

  getInfo() {
    return `${this.name} is ${this.age} years old and
teaches ${this.subject}.`;
  }
}

const student1 = new Student("Alice", 20, "Computer
Science");
const teacher1 = new Teacher("Mr. Smith", 40,
"Mathematics");

console.log(student1.getInfo());
console.log(teacher1.getInfo());
```

- OUTPUT


John Doe
Person
Age: 35
Hello, I'm John Doe.
Alice Johnson
Student
Age: 20
Hello, I'm Alice Johnson. I'm studying Computer Science.
Course: Computer Science
Grade: A+
Alice Johnson is studying Computer Science.
Bob Smith
Student
Age: 19
Hello, I'm Bob Smith. I'm studying Mathematics.
Course: Mathematics
Grade: B+
Bob Smith is studying Mathematics.
Dr. Sarah Wilson
Teacher
Age: 45
Hello, I'm Dr. Sarah Wilson. I teach Physics.
Subject: Physics
Experience: 15 years
Dr. Sarah Wilson is teaching Physics.
Prof. Mike Brown
Teacher
Age: 38
Hello, I'm Prof. Mike Brown. I teach Chemistry.
Subject: Chemistry
Experience: 8 years
Prof. Mike Brown is teaching Chemistry.