Design a mini online quiz interface that updates scores dynamically.

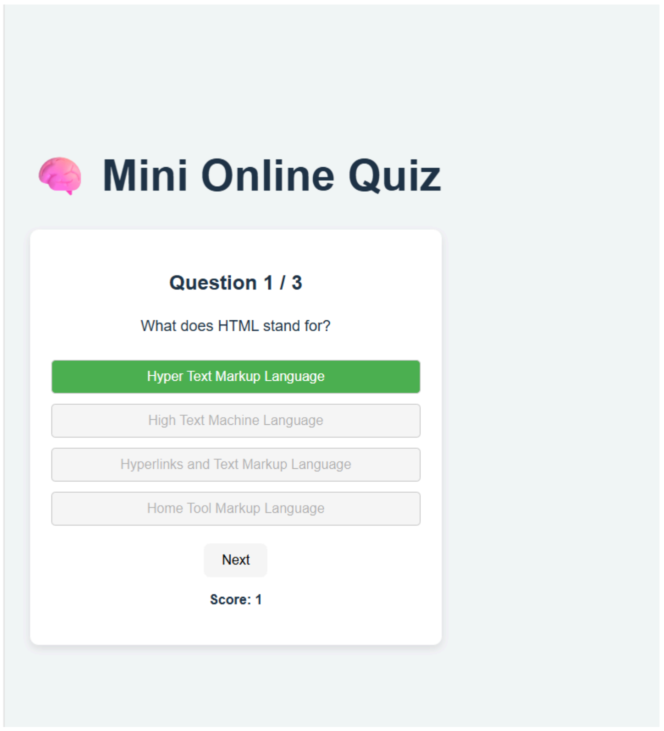
App.jsx

```
import React, { useState } from "react";
import "./App.css";

function App() {
  const questions = [
    {
      question: "What does HTML stand for?",
      options: [
        "Hyper Text Markup Language",
        "High Text Machine Language",
        "Hyperlinks and Text Markup Language",
        "Home Tool Markup Language",
      ],
      correct: 0,
    },
    {
      question: "Which language is used for styling web pages?",
      options: ["HTML", "JQuery", "CSS", "XML"],
      correct: 2,
    },
    {
      question: "Which is a JavaScript framework?",
      options: ["React", "Laravel", "Django", "Flask"],
      correct: 0,
    },
  ];

  const [currentQuestion, setCurrentQuestion] = useState(0);
  const [score, setScore] = useState(0);
  const [selected, setSelected] = useState(null);
  const [showResult, setShowResult] = useState(false);
const handleOptionClick = (index) => {
  setSelected(index);

  if (index === questions[currentQuestion].correct) {
    setScore(score + 1);
  }
};

const nextQuestion = () => {
  setSelected(null);

  if (currentQuestion + 1 < questions.length) {
    setCurrentQuestion(currentQuestion + 1);
  } else {
    setShowResult(true);
  }
};

return (
  <div className="app">
    <h1>🧠 Mini Online Quiz</h1>

     {!showResult ? (
      <div className="quiz-box">
        <h2>
          Question {currentQuestion + 1} / {questions.length}
        </h2>
        <p className="question">
          {questions[currentQuestion].question}
        </p>

        <div className="options">
          {questions[currentQuestion].options.map((option, index) => (
        <button
          key={index}
          className={`option-btn ${
            selected === index ? "selected" : ""
          }`}
          onClick={() => handleOptionClick(index)}
          disabled={selected !== null}
        >
          {option}
        </button>
      ))}
    </div>

    <button
      className="next-btn"
      onClick={nextQuestion}
      disabled={selected === null}
    >
      Next
    </button>

    <p className="score">Score: {score}</p>
  </div>
) : (
  <div className="result-box">
    <h2>🎉 Quiz Completed!</h2>
    <p>
      Your Score: <strong>{score}</strong> / {questions.length}
    </p>
    <button onClick={() => window.location.reload()}>
      Restart Quiz
    </button>
  </div>
)}
```

```
    </div>
  );
}

export default App;
```

App.css

```
body {
  background-color: #f4f6f8;
  font-family: Arial, sans-serif;
}

.app {
  text-align: center;
  padding: 30px;
}

.quiz-box,
.result-box {
  background: white;
  max-width: 500px;
  margin: auto;
  padding: 25px;
  border-radius: 10px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}

.question {
  font-size: 18px;
  margin-bottom: 20px;
}

.options {
  display: flex;
  flex-direction: column;
}

.option-btn {
  padding: 10px;
  margin: 6px 0;
  font-size: 16px;
  cursor: pointer;
  border-radius: 5px;
  border: 1px solid #ccc;
  background: #f9f9f9;
}

.option-btn:hover {
  background: #eaeaea;
}

.option-btn.selected {
  background: #4caf50;
  color: white;
}

.next-btn {
  margin-top: 15px;
  padding: 10px 20px;
  font-size: 16px;
  cursor: pointer;
}

.score {
  margin-top: 15px;
  font-weight: bold;
}
```

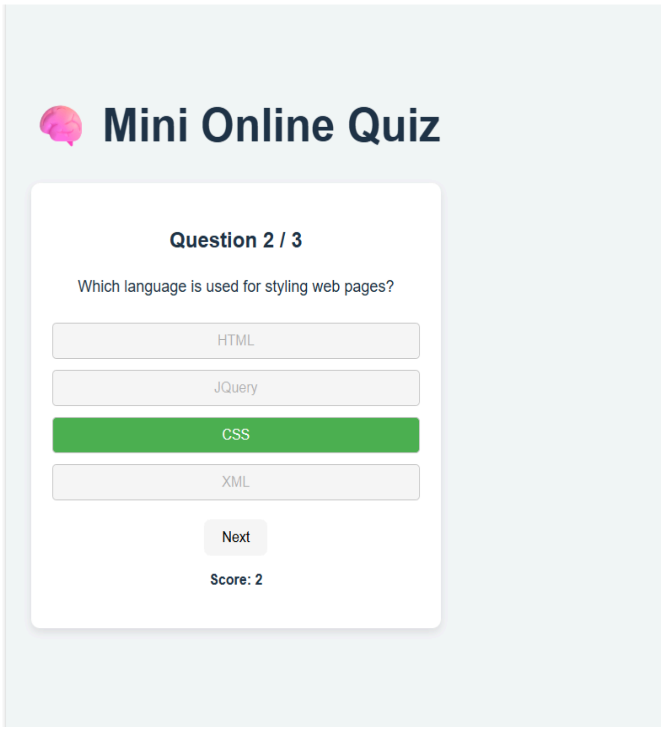

🧠 Mini Online Quiz

Question 1 / 3

What does HTML stand for?

Hyper Text Markup Language
High Text Machine Language
Hyperlinks and Text Markup Language
Home Tool Markup Language
Next

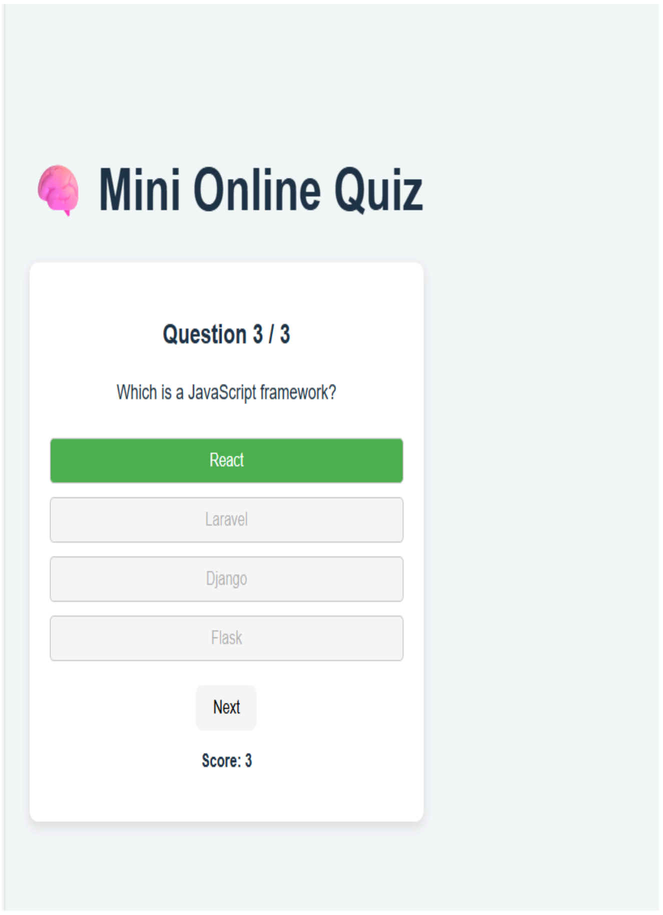
Score: 1


🧠 Mini Online Quiz

Question 2 / 3

Which language is used for styling web pages?

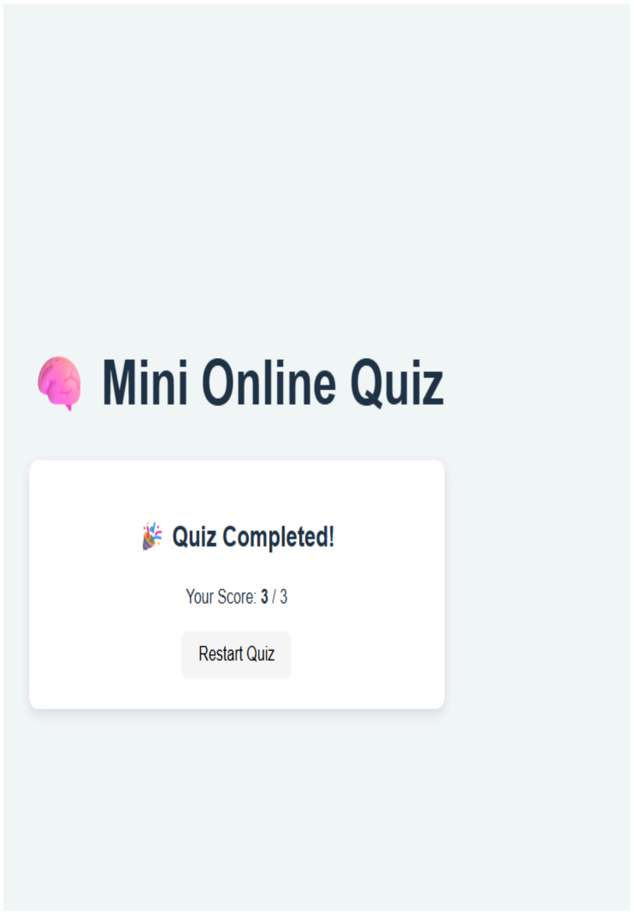
Score: 2


🧠 Mini Online Quiz

Question 3 / 3

Which is a JavaScript framework?

Score: 3


🧠 Mini Online Quiz

🎉 Quiz Completed!

Your Score: **3** / 3

Task 4: Implement a real-time polling system for classroom use.

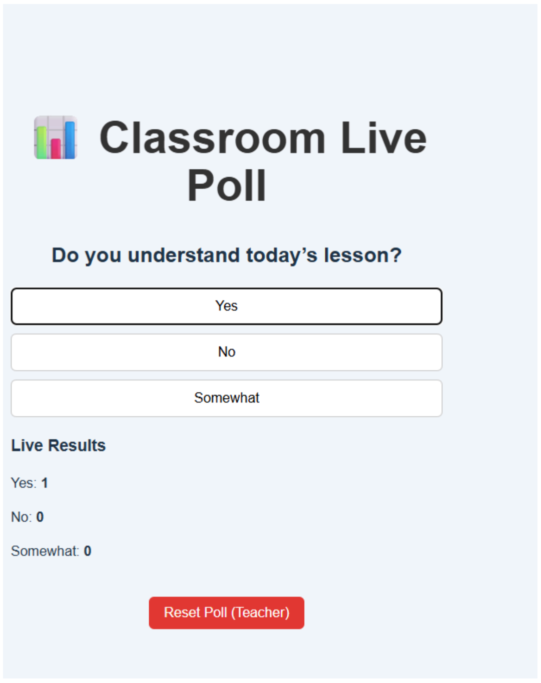
App.jsx

```
import { useEffect, useState } from "react";
import "./App.css";

const channel = new BroadcastChannel("classroom-poll");

export default function App() {
  const [poll, setPoll] = useState({
    question: "Do you understand today’s lesson?",
    options: {
      Yes: 0,
      No: 0,
      Somewhat: 0,
    },
  });

  // Listen for real-time updates
  useEffect(() => {
    channel.onmessage = (event) => {
      setPoll(event.data);
    };

    return () => channel.close();
  }, []);

  const vote = (option) => {
    const updatedPoll = {
      ...poll,
      options: {
        ...poll.options,
        [option]: poll.options[option] + 1,
      },
    };

    setPoll(updatedPoll);
    channel.postMessage(updatedPoll); // broadcast update
  };
```

```
 const resetPoll = () => {
   const reset = {
     ...poll,
     options: Object.fromEntries(
       Object.keys(poll.options).map((key) => [key, 0])
     ),
   };

   setPoll(reset);
   channel.postMessage(reset);
 };

 return (
   <div className="app">
     <h1>📊 Classroom Live Poll</h1>
     <h2>{poll.question}</h2>

     <div className="options">
       {Object.keys(poll.options).map((option) => (
         <button key={option} onClick={() => vote(option)}>
           {option}
         </button>
       ))}
     </div>

     <div className="results">
       <h3>Live Results</h3>
       {Object.entries(poll.options).map(([key, value]) => (
         <p key={key}>
           {key}: <strong>{value}</strong>
         </p>
       ))}
     </div>

     <button className="reset" onClick={resetPoll}>
       Reset Poll (Teacher)
     </button>
   </div>
 );
}
```

App.css

```
body {
  margin: 0;
  background: #f4f6fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 500px;
  margin: auto;
  text-align: center;
  padding: 30px;
}

h1 {
  color: #333;
}

.options button {
  display: block;
  width: 100%;
  margin: 10px 0;
  padding: 12px;
  font-size: 16px;
  cursor: pointer;
  border-radius: 6px;
  border: 1px solid #ccc;
  background: white;
}

.options button:hover {
  background: #e3f2fd;
}

.results {
  margin-top: 20px;
  text-align: left;
}

.reset {
    margin-top: 25px;
    padding: 10px 18px;
    background: #e53935;
    color: white;
    border: none;
    border-radius: 6px;
    cursor: pointer;
  }
```



📊 Classroom Live Poll

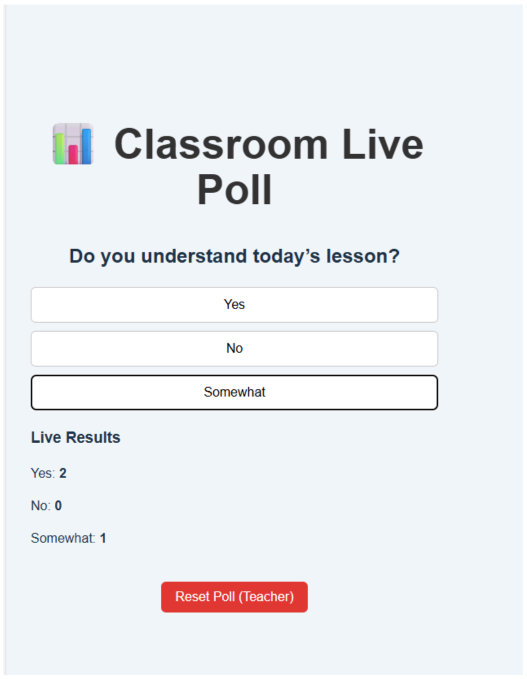
Do you understand today’s lesson?

Live Results

Yes: **1**

No: **0**

Somewhat: **0**



📊 Classroom Live Poll

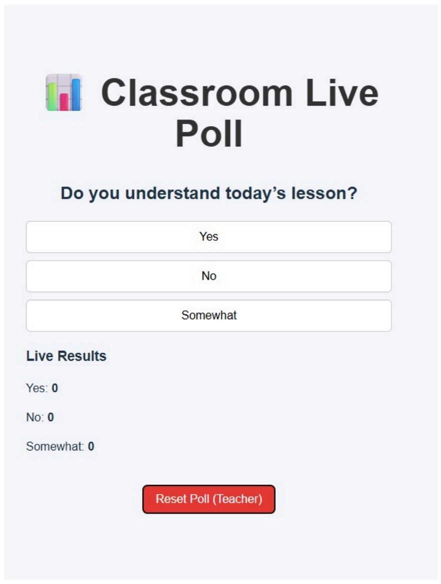
Do you understand today’s lesson?

Live Results

Yes: **2**

No: **0**

Somewhat: **1**



📊 Classroom Live Poll

Do you understand today’s lesson?

Live Results

Yes: **0**

No: **0**

Somewhat: **0**

Task 5: Design a feedback form interface that displays submitted data dynamically on the screen.

App.jsx

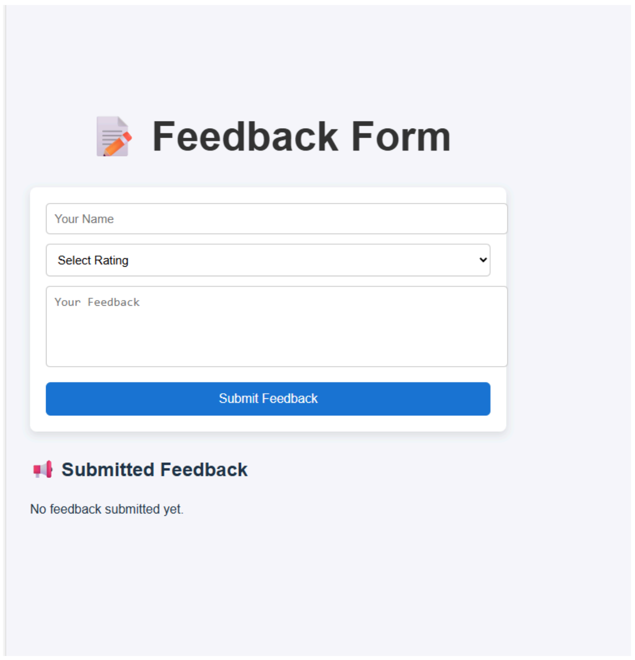
```jsx
import { useState } from "react";
import "./App.css";

export default function App() {
  const [name, setName] = useState("");
  const [rating, setRating] = useState("");
  const [comment, setComment] = useState("");
  const [feedbackList, setFeedbackList] = useState([]);

  const handleSubmit = (e) => {
    e.preventDefault();

    if (!name || !rating || !comment) return;

    const newFeedback = {
      id: Date.now(),
      name,
      rating,
      comment,
    };

    setFeedbackList([newFeedback, ...feedbackList]);

    // Clear form
    setName("");
    setRating("");
    setComment("");
  };

  return (
    <div className="app">
      <h1>📝 Feedback Form</h1>

      <form className="form" onSubmit={handleSubmit}>
        <input
          type="text"
      placeholder="Your Name"
      value={name}
      onChange={(e) => setName(e.target.value)}
    />

    <select
      value={rating}
      onChange={(e) => setRating(e.target.value)}
    >
      <option value="">Select Rating</option>
      <option value="Excellent">Excellent</option>
      <option value="Good">Good</option>
      <option value="Average">Average</option>
      <option value="Poor">Poor</option>
    </select>

    <textarea
      placeholder="Your Feedback"
      value={comment}
      onChange={(e) => setComment(e.target.value)}
    />

    <button type="submit">Submit Feedback</button>
  </form>

  <div className="feedback-section">
    <h2>📢 Submitted Feedback</h2>

    {feedbackList.length === 0 && (
      <p>No feedback submitted yet.</p>
    )}

    {feedbackList.map((item) => (
      <div key={item.id} className="feedback-card">
        <h3>{item.name}</h3>
        <p className="rating">Rating: {item.rating}</p>
        <p>{item.comment}</p>
      </div>
    ))}
  </div>
```

```
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  background: #f5f7fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 600px;
  margin: auto;
  padding: 30px;
}

h1 {
  text-align: center;
  color: #333;
}

.form {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}

.form input,
.form select,
.form textarea {
  width: 100%;
  margin-bottom: 12px;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}
```

```css
.form textarea {
  resize: none;
  height: 80px;
}

.form button {
  width: 100%;
  padding: 12px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

.form button:hover {
  background: #125aa3;
}

.feedback-section {
  margin-top: 30px;
}

.feedback-card {
  background: white;
  padding: 15px;
  margin-bottom: 12px;
  border-radius: 8px;
  box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);
}

.rating {
  font-weight: bold;
  color: #1976d2;
}
```



Feedback Form

Your Name

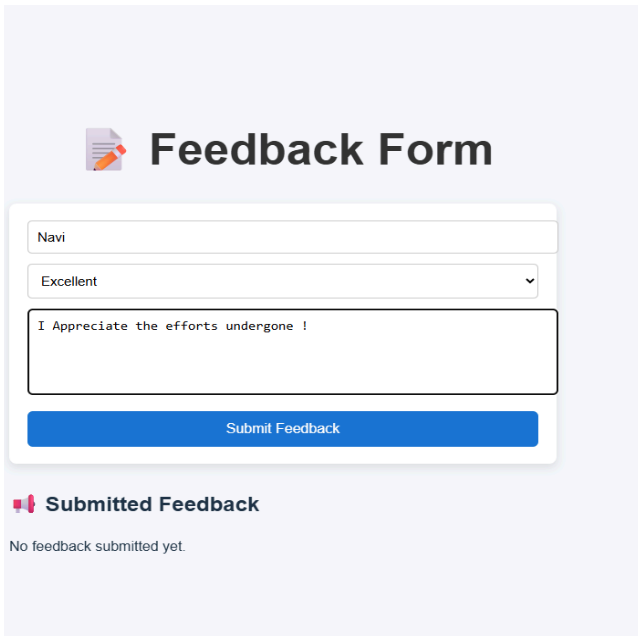
Select Rating

Your Feedback

Submit Feedback

Submitted Feedback

No feedback submitted yet.



Feedback Form

Navi

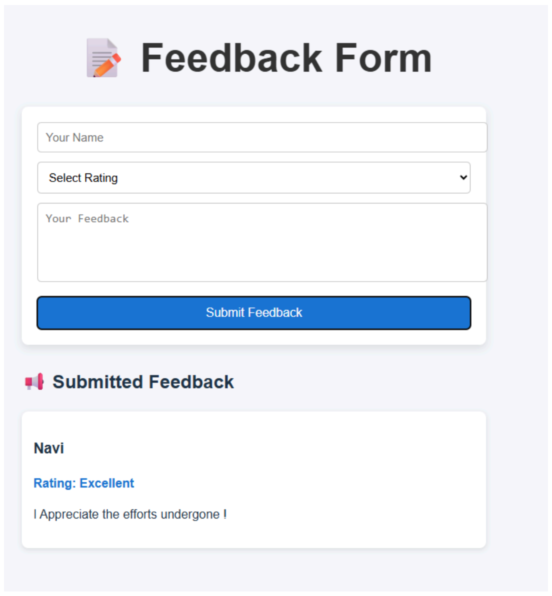
Excellent

I Appreciate the efforts undergone !

Submit Feedback

Submitted Feedback

No feedback submitted yet.



Feedback Form

Your Name

Select Rating

Your Feedback

Submit Feedback

Submitted Feedback

Navi

Rating: Excellent

I Appreciate the efforts undergone !

Task 6: Implement a simple course enrollment form that updates the enrolled course list in real time.

App.jsx

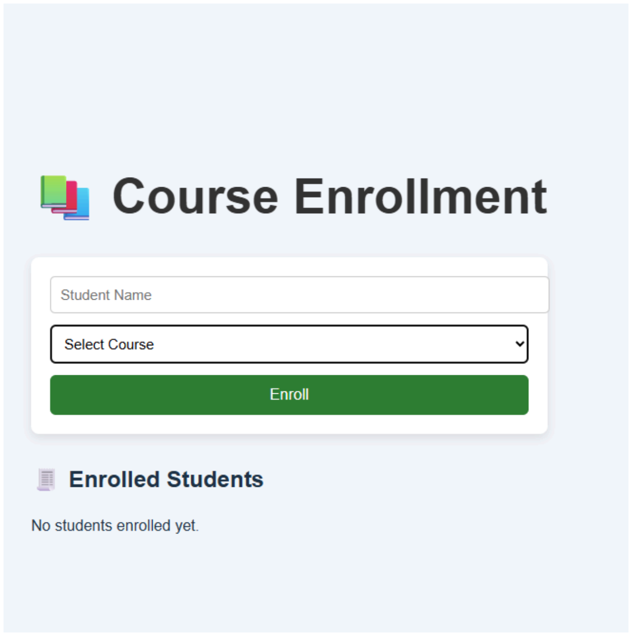
```
import { useState } from "react";
import "./App.css";
```

```
export default function App() {
  const [studentName, setStudentName] = useState("");
  const [course, setCourse] = useState("");
  const [enrollments, setEnrollments] = useState([]);

  const handleEnroll = (e) => {
    e.preventDefault();

    if (!studentName || !course) return;

    const newEnrollment = {
      id: Date.now(),
      studentName,
      course,
    };

    setEnrollments([...enrollments, newEnrollment]);

    // Clear form
    setStudentName("");
    setCourse("");
  };

  return (
    <div className="app">
      <h1>📚 Course Enrollment</h1>

      <form className="form" onSubmit={handleEnroll}>
        <input
          type="text"
          placeholder="Student Name"
          value={studentName}
          onChange={(e) => setStudentName(e.target.value)}
        />

        <select
          value={course}
          onChange={(e) => setCourse(e.target.value)}
        >
          <option value="">Select Course</option>
          <option value="Mathematics">Mathematics</option>
          <option value="Computer Science">Computer Science</option>
          <option value="Physics">Physics</option>
          <option value="Chemistry">Chemistry</option>
        </select>

        <button type="submit">Enroll</button>
      </form>

      <div className="enrollment-section">
        <h2>🧾 Enrolled Students</h2>

        {enrollments.length === 0 && (
          <p>No students enrolled yet.</p>
        )}

        <ul>
          {enrollments.map((item) => (
            <li key={item.id}>
              <strong>{item.studentName}</strong> — {item.course}
            </li>
          ))}
        </ul>
      </div>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  background: #f3f6fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 600px;
  margin: auto;
  padding: 30px;
```

```
}

h1 {
  text-align: center;
  color: #333;
}

.form {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);
}

.form input,
.form select {
  width: 100%;
  margin-bottom: 12px;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

.form button {
  width: 100%;
  padding: 12px;
  background: #2e7d32;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

.form button:hover {
  background: #1b5e20;
}

.enrollment-section {
  margin-top: 30px;
}

.enrollment-section ul {
  list-style: none;
  padding: 0;
}

.enrollment-section li {
  background: white;
  padding: 12px;
  margin-bottom: 8px;
  border-radius: 6px;
  box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);
}
```



📚 Course Enrollment

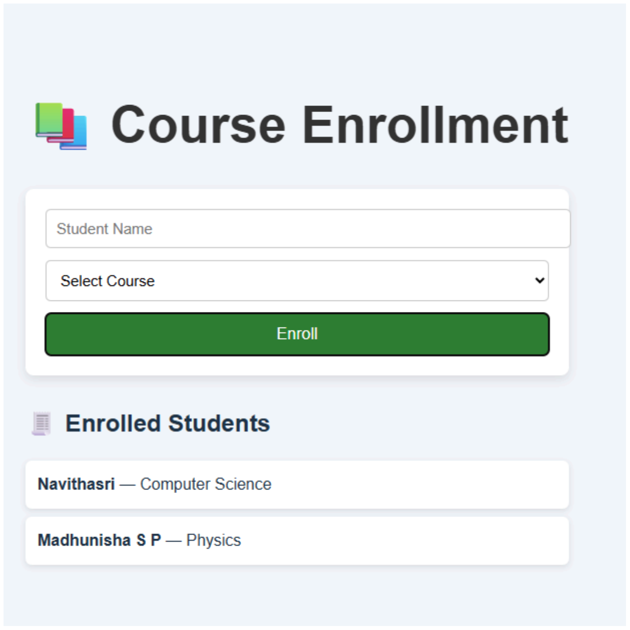
Student Name

Select Course

Enroll

Enrolled Students

No students enrolled yet.



📚 Course Enrollment

Student Name

Select Course

Enroll

Enrolled Students

Navithasri — Computer Science

Madhunisha S P — Physics

Task 7: Simulate an attendance tracker interface that marks students present or absent dynamically.

App.jsx

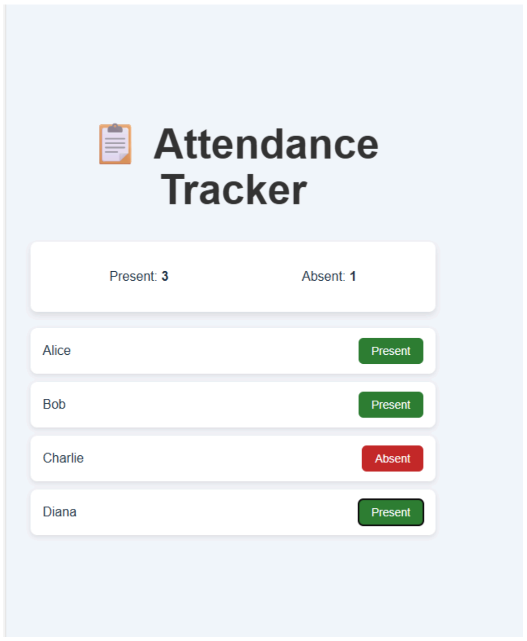
```
import { useState } from "react";
import "./App.css";
```

```
export default function App() {
  const [students, setStudents] = useState([
    { id: 1, name: "Alice", present: false },
    { id: 2, name: "Bob", present: false },
    { id: 3, name: "Charlie", present: false },
    { id: 4, name: "Diana", present: false },
  ]);

  const toggleAttendance = (id) => {
    setStudents(
      students.map((student) =>
        student.id === id
          ? { ...student, present: !student.present }
          : student
      )
    );
  };

  const presentCount = students.filter((s) => s.present).length;
  const absentCount = students.length - presentCount;

  return (
    <div className="app">
      <h1>📋 Attendance Tracker</h1>

      <div className="summary">
        <p>Present: <strong>{presentCount}</strong></p>
        <p>Absent: <strong>{absentCount}</strong></p>
      </div>

      <ul className="student-list">
        {students.map((student) => (
          <li key={student.id} className="student-item">
            <span>{student.name}</span>

            <button
              className={student.present ? "present" : "absent"}
              onClick={() => toggleAttendance(student.id)}
            >
```

```
              {student.present ? "Present" : "Absent"}
            </button>
          </li>
        ))}
      </ul>
    </div>
  );
}
```

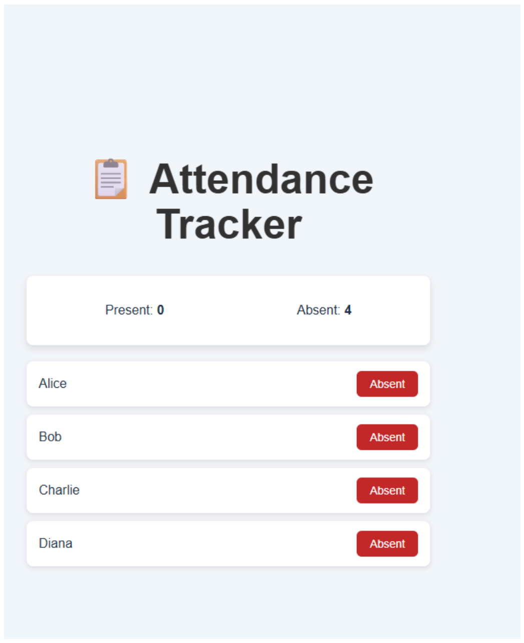
App.css

```
body {
  margin: 0;
  background: #f4f6fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 500px;
  margin: auto;
  padding: 30px;
}

h1 {
  text-align: center;
  color: #333;
}

.summary {
  display: flex;
  justify-content: space-around;
  margin-bottom: 20px;
  background: white;
  padding: 15px;
  border-radius: 8px;
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);
}

.student-list {
  list-style: none;
  padding: 0;
}

.student-item {
  background: white;
  padding: 12px 15px;
  margin-bottom: 10px;
  border-radius: 8px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);
}

.student-item span {
  font-size: 16px;
}

button {
  padding: 8px 16px;
  border: none;
  border-radius: 6px;
  cursor: pointer;
  color: white;
  font-size: 14px;
}

button.present {
  background: #2e7d32;
}

button.absent {
  background: #c62828;
}
```


📋 Attendance Tracker

Present: **3** Absent: **1**

Alice Present
Bob Present
Charlie Absent
Diana Present


📋 Attendance Tracker

Present: **0** Absent: **4**

Alice Absent
Bob Absent
Charlie Absent
Diana Absent

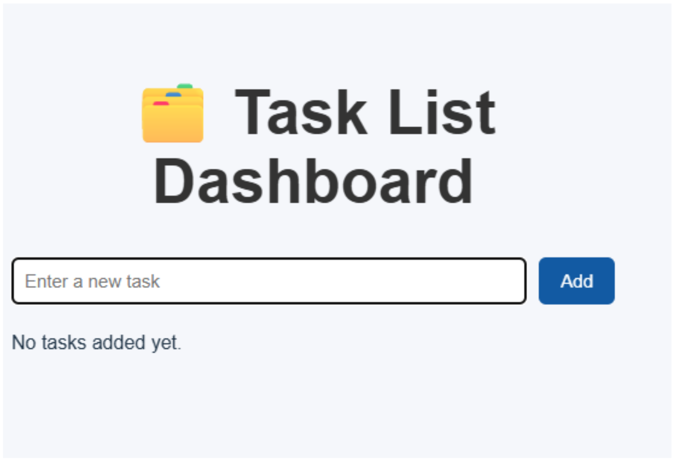
Task 8: Develop a simple task list dashboard that allows adding and removing tasks.

App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [task, setTask] = useState("");
  const [tasks, setTasks] = useState([]);

  const addTask = (e) => {
    e.preventDefault();

    if (!task.trim()) return;

    const newTask = {
      id: Date.now(),
      text: task,
    };

    setTasks([...tasks, newTask]);
    setTask("");
  };
  const removeTask = (id) => {
    setTasks(tasks.filter((t) => t.id !== id));
  };

  return (
    <div className="app">
      <h1>🗂️ Task List Dashboard</h1>

      <form className="task-form" onSubmit={addTask}>
        <input
          type="text"
          placeholder="Enter a new task"
          value={task}
          onChange={(e) => setTask(e.target.value)}
        />
        <button type="submit">Add</button>
      </form>

      <ul className="task-list">
        {tasks.length === 0 && <p>No tasks added yet.</p>}

        {tasks.map((t) => (
          <li key={t.id} className="task-item">
            <span>{t.text}</span>
            <button
              className="delete"
              onClick={() => removeTask(t.id)}
            >
              ❌
            </button>
          </li>
        ))}
      </ul>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  background: #f5f7fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 500px;
  margin: auto;
  padding: 30px;
}

h1 {
  text-align: center;
  color: #333;
}

.task-form {
  display: flex;
  gap: 10px;
  margin-bottom: 20px;
}

.task-form input {
  flex: 1;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

.task-form button {
  padding: 10px 18px;
  font-size: 15px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
}

.task-form button:hover {
  background: #125aa3;
}

.task-list {
  list-style: none;
  padding: 0;
}

.task-item {
  background: white;
  padding: 12px;
  margin-bottom: 8px;
  border-radius: 6px;
  display: flex;
  justify-content: space-between;
  align-items: center;
  box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);
}

.delete {
  background: #e53935;
  border: none;
  border-radius: 6px;
  color: white;
  cursor: pointer;
  padding: 6px 10px;
  font-size: 14px;
}
```

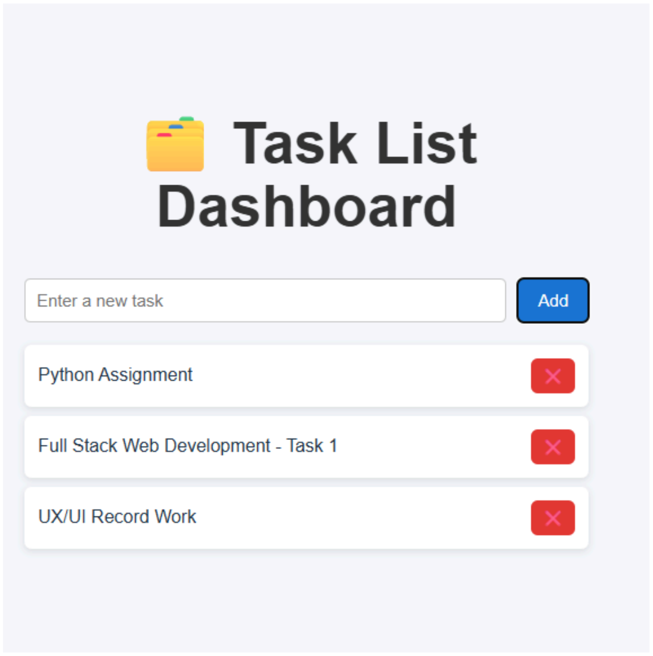


🗂️ Task List Dashboard

Enter a new task

Add

No tasks added yet.



🗂️ Task List Dashboard

Enter a new task

Add

Python Assignment

Full Stack Web Development - Task 1

UX/UI Record Work

Task 9: Flight Ticket Booking using JavaScript Dialog Boxes. (Alerts, Confirmations, and User Input (Prompt) Dialogs).

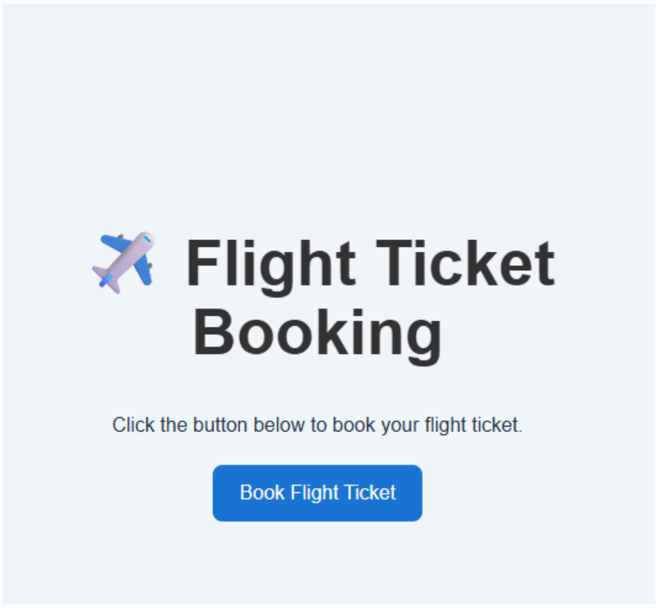
App.jsx

```
import "./App.css";

export default function App() {
  const bookTicket = () => {
    const name = prompt("Enter your name:");
    if (!name) {
      alert("Booking cancelled: Name is required.");
      return;
    }

    const destination = prompt("Enter destination city:");
    if (!destination) {
      alert("Booking cancelled: Destination is required.");
      return;
    }

    const confirmBooking = confirm(
      `Confirm booking?\n\nPassenger: ${name}\nDestination: ${destination}`
    );

    if (confirmBooking) {
      alert(
        `✅ Booking Confirmed!\n\nPassenger: ${name}\nDestination: ${destination}\nHave a
safe journey ✈️`
      );
    } else {
      alert("❌ Booking cancelled by user.");
    }
  };

  return (
    <div className="app">
      <h1>✈️ Flight Ticket Booking</h1>
      <p>Click the button below to book your flight ticket.</p>

      <button onClick={bookTicket}>Book Flight Ticket</button>
    </div>
  );
}
```

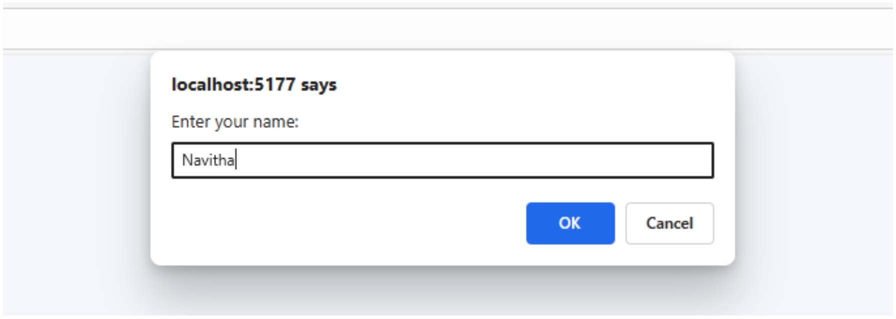
App.css

```
body {
  margin: 0;
  background: #f4f6fb;
  font-family: Arial, sans-serif;
}
```

```css
.app {
  max-width: 500px;
  margin: auto;
  padding: 40px;
  text-align: center;
}

h1 {
  color: #333;
}

p {
  margin-bottom: 20px;
  font-size: 16px;
}

button {
  padding: 14px 22px;
  font-size: 16px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}
```

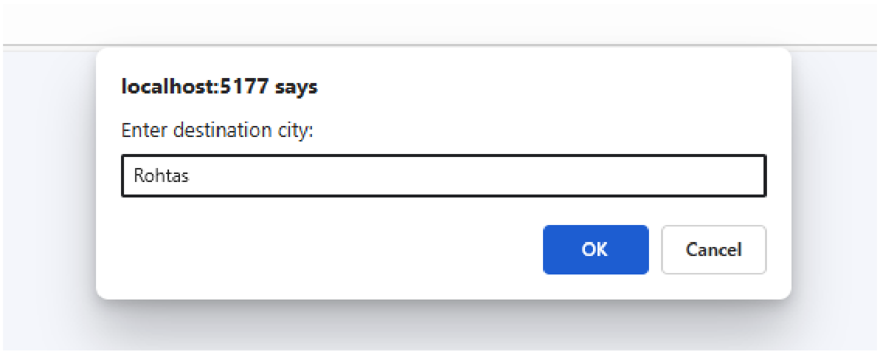

✈️ Flight Ticket Booking

Click the button below to book your flight ticket.


localhost:5177 says

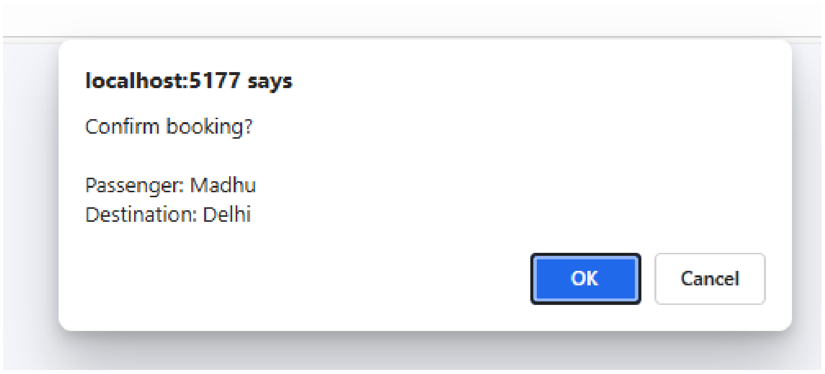
Enter your name:

Navitha


localhost:5177 says

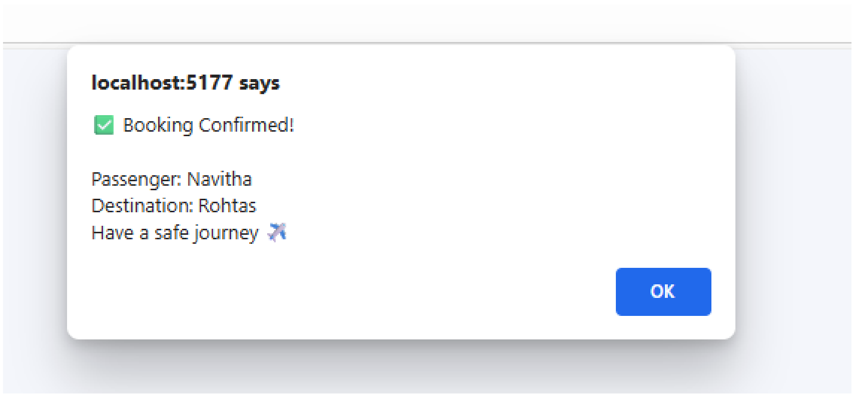
Enter destination city:

Rohtas


localhost:5177 says

Confirm booking?

Passenger: Madhu
Destination: Delhi


localhost:5177 says

✅ Booking Confirmed!

Passenger: Navitha
Destination: Rohtas
Have a safe journey ✈️

Task 10: Hotel Reservation System using JavaScript Dialog Boxes

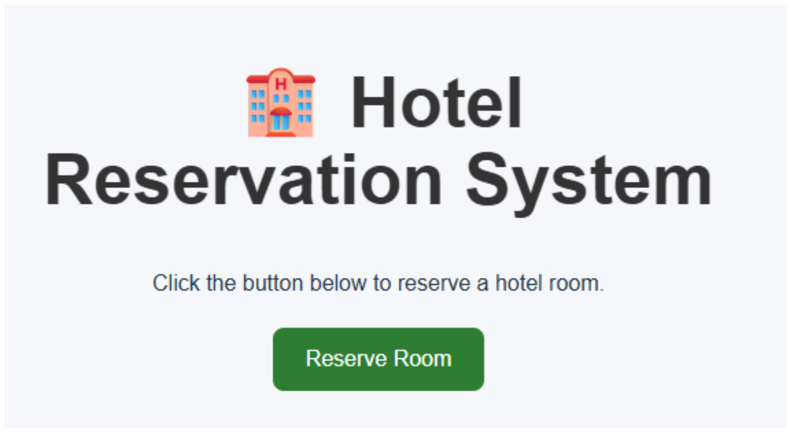
App.jsx

```
import "./App.css";

export default function App() {
  const reserveRoom = () => {
    const name = prompt("Enter guest name:");
    if (!name) {
      alert("Reservation cancelled: Name is required.");
      return;
    }

    const roomType = prompt(
      "Enter room type (Single / Double / Suite):"
    );
    if (!roomType) {
      alert("Reservation cancelled: Room type is required.");
      return;
    }

    const nights = prompt("Enter number of nights:");
    if (!nights || isNaN(nights) || nights <= 0) {
      alert("Reservation cancelled: Invalid number of nights.");
      return;
    }

    const confirmReservation = confirm(
      `Confirm Reservation?\n\nGuest: ${name}\nRoom Type: ${roomType}\nNights: ${nights}`
    );

    if (confirmReservation) {
      alert(
        `🏨 Reservation Confirmed!\n\nGuest: ${name}\nRoom: ${roomType}\nNights:
${nights}\nEnjoy your stay!`
      );
    } else {
      alert("❌ Reservation cancelled by user.");
    }
  };

  return (
    <div className="app">
      <h1>🏨 Hotel Reservation System</h1>
      <p>Click the button below to reserve a hotel room.</p>

      <button onClick={reserveRoom}>Reserve Room</button>
    </div>
  );
}
```

App.css

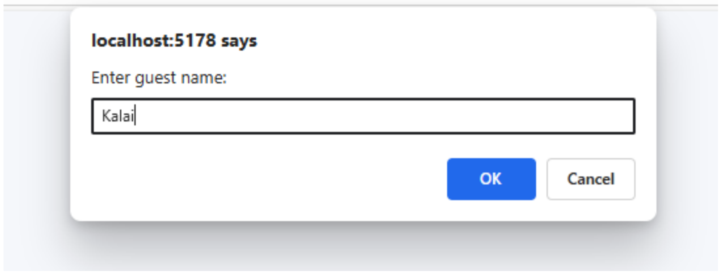
```
body {
  margin: 0;
  background: #f5f7fb;
  font-family: Arial, sans-serif;
}
```

```
.app {
  max-width: 500px;
  margin: auto;
  padding: 40px;
  text-align: center;
}

h1 {
  color: #333;
}

p {
  margin-bottom: 20px;
  font-size: 16px;
}

button {
  padding: 14px 24px;
  font-size: 16px;
  background: #2e7d32;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
}

button:hover {
  background: #1b5e20;
}
```

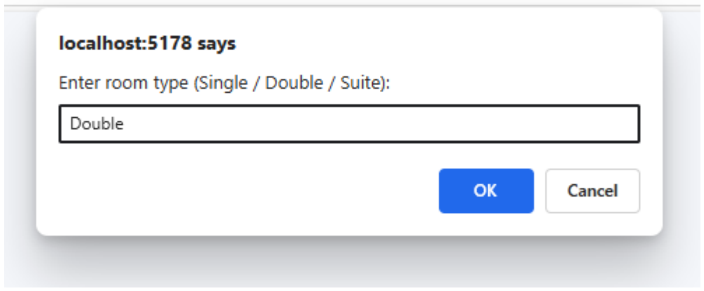


🏨 Hotel Reservation System

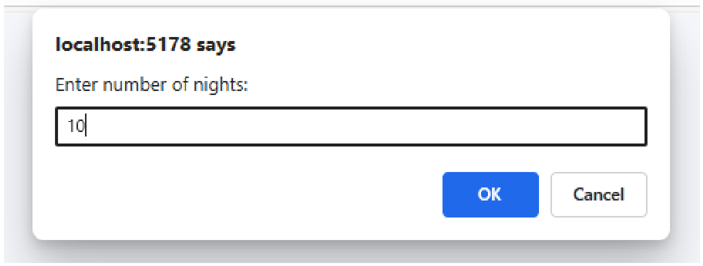
Click the button below to reserve a hotel room.



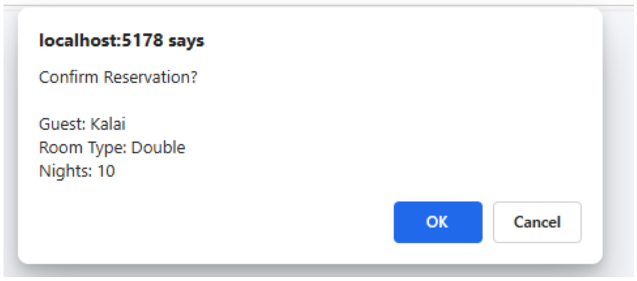
```
localhost:5178 says
Enter guest name:
Kalai
```



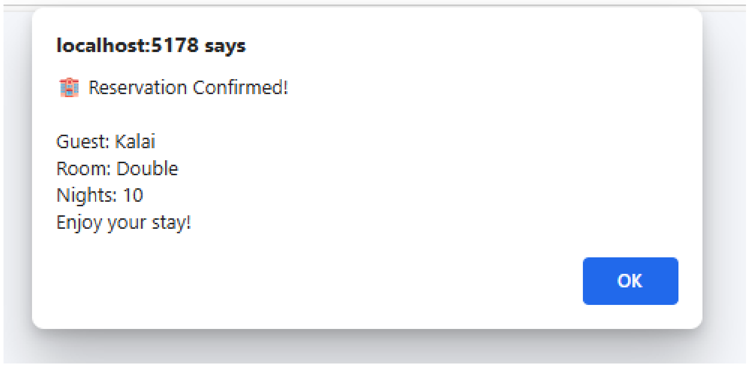
```
localhost:5178 says
Enter room type (Single / Double / Suite):
Double
```



```
localhost:5178 says
Enter number of nights:
10
```



```
localhost:5178 says
Confirm Reservation?

Guest: Kalai
Room Type: Double
Nights: 10
```



```
localhost:5178 says
🏨 Reservation Confirmed!

Guest: Kalai
Room: Double
Nights: 10
Enjoy your stay!
```

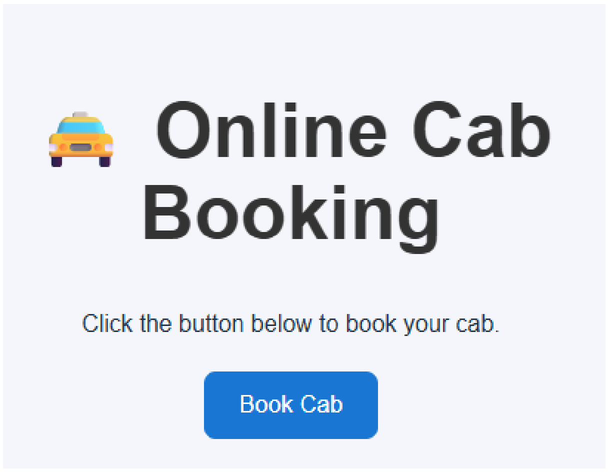
Task 11: Online Cab Booking Application using JavaScript Dialog Boxes.

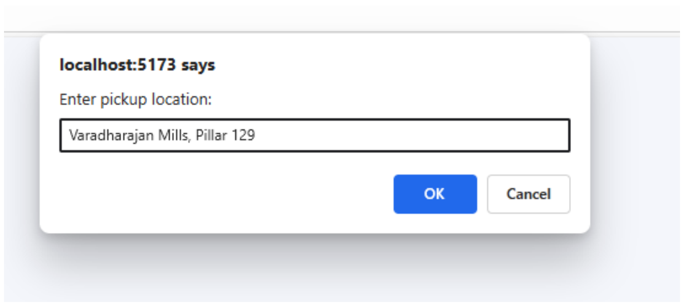
App.jsx

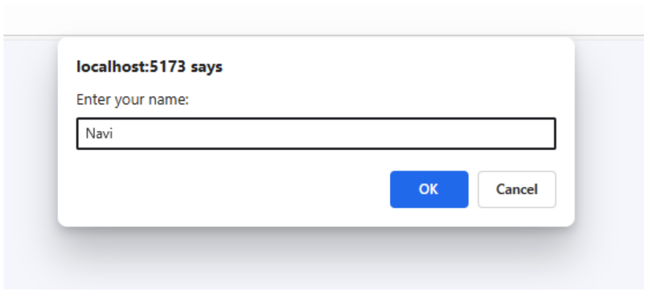
```
import "./App.css";

export default function App() {
  const bookCab = () => {
    const name = prompt("Enter your name:");
    if (!name) {
      alert("Booking cancelled: Name is required.");
      return;
    }
    const pickup = prompt("Enter pickup location:");
    if (!pickup) {
      alert("Booking cancelled: Pickup location is required.");
      return;
    }

    const drop = prompt("Enter drop location:");
    if (!drop) {
      alert("Booking cancelled: Drop location is required.");
      return;
    }

    const confirmBooking = confirm(
      `Confirm Cab Booking?\n\nPassenger: ${name}\nPickup: ${pickup}\nDrop: ${drop}`
    );

    if (confirmBooking) {
      alert(
        `🚖 Cab Booking Confirmed!\n\nPassenger: ${name}\nPickup: ${pickup}\nDrop:
${drop}\nDriver will arrive shortly.`
      );
    } else {
      alert("❌ Booking cancelled by user.");
    }
  };

  return (
    <div className="app">
      <h1>🚖 Online Cab Booking</h1>
      <p>Click the button below to book your cab.</p>

      <button onClick={bookCab}>Book Cab</button>
    </div>
  );
}
```

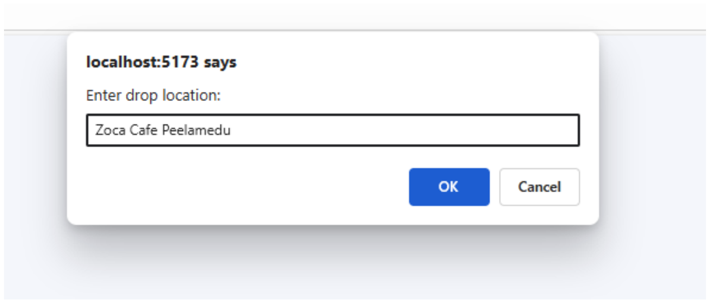
App.css

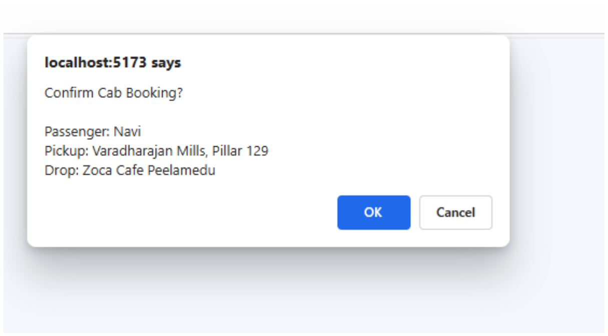
```
body {
  margin: 0;
  background: #f4f6fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 500px;
  margin: auto;
  padding: 40px;
  text-align: center;
}

h1 {
  color: #333;
}

p {
  margin-bottom: 20px;
  font-size: 16px;
}

button {
  padding: 14px 24px;
  font-size: 16px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 8px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}
```



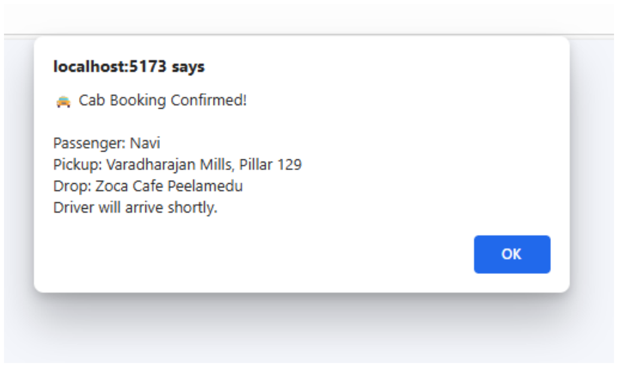

localhost:5173 says
Enter pickup location:
Varadharajan Mills, Pillar 129


localhost:5173 says
Enter your name:
Navi


localhost:5173 says
Enter drop location:
Zoca Cafe Peelamedu


localhost:5173 says
Confirm Cab Booking?

Passenger: Navi
Pickup: Varadharajan Mills, Pillar 129
Drop: Zoca Cafe Peelamedu


localhost:5173 says
🚕 Cab Booking Confirmed!

Passenger: Navi
Pickup: Varadharajan Mills, Pillar 129
Drop: Zoca Cafe Peelamedu
Driver will arrive shortly.

Task 12: Design an event registration interface with confirmation alerts.

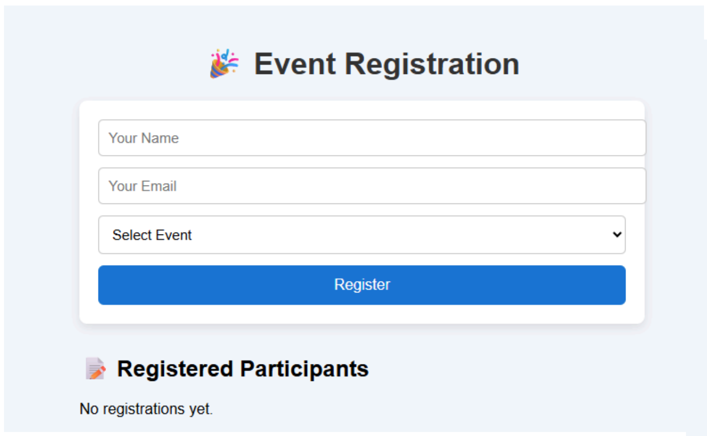
App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [name, setName] = useState("");
  const [email, setEmail] = useState("");
  const [eventType, setEventType] = useState("");
const [registrations, setRegistrations] = useState([]);

const handleRegister = (e) => {
  e.preventDefault();

  if (!name || !email || !eventType) {
    alert("❌ Please fill all fields before registering.");
    return;
  }

  const newRegistration = { id: Date.now(), name, email, eventType };
  setRegistrations([newRegistration, ...registrations]);

  alert(
    `✅ Registration Successful!\n\nName: ${name}\nEmail: ${email}\nEvent: ${eventType}`
  );

  // Clear form
  setName("");
  setEmail("");
  setEventType("");
};

return (
  <div className="app">
    <h1>🎉 Event Registration</h1>

    <form className="form" onSubmit={handleRegister}>
      <input
        type="text"
        placeholder="Your Name"
        value={name}
        onChange={(e) => setName(e.target.value)}
      />

      <input
        type="email"
        placeholder="Your Email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
        />

        <select
          value={eventType}
          onChange={(e) => setEventType(e.target.value)}
        >
          <option value="">Select Event</option>
          <option value="Workshop">Workshop</option>
          <option value="Seminar">Seminar</option>
          <option value="Webinar">Webinar</option>
        </select>

        <button type="submit">Register</button>
      </form>

      <div className="registration-list">
        <h2>📝 Registered Participants</h2>
        {registrations.length === 0 && <p>No registrations yet.</p>}

        {registrations.map((r) => (
          <div key={r.id} className="registration-card">
            <p><strong>Name:</strong> {r.name}</p>
            <p><strong>Email:</strong> {r.email}</p>
            <p><strong>Event:</strong> {r.eventType}</p>
          </div>
        ))}
      </div>
    </div>
  );
}
```

App.css

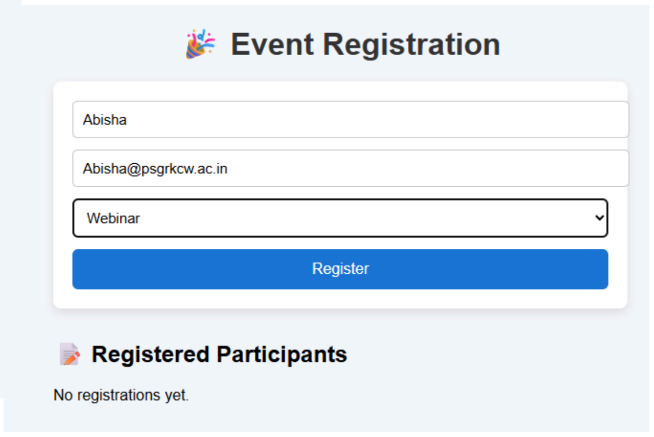
```
body {
  margin: 0;
  background: #f4f6fb;
  font-family: Arial, sans-serif;
}

.app {
  max-width: 600px;
  margin: auto;
  padding: 30px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.form {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  margin-bottom: 30px;
}

.form input,
.form select {
  width: 100%;
  margin-bottom: 12px;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

.form button {
  width: 100%;
  padding: 12px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}
```

```
.form button:hover {
  background: #125aa3;
}

.registration-list {
  text-align: left;
}

.registration-card {
  background: white;
  padding: 12px 15px;
  margin-bottom: 10px;
  border-radius: 6px;
  box-shadow: 0 2px 6px rgba(0,0,0,0.1);
}
```



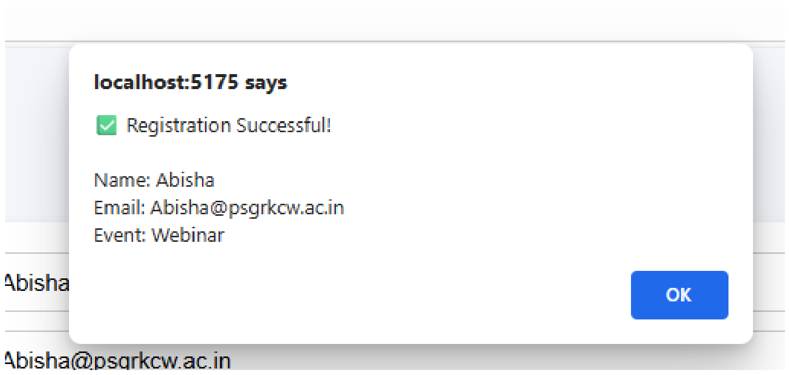
📝 Registered Participants

No registrations yet.



📝 Registered Participants

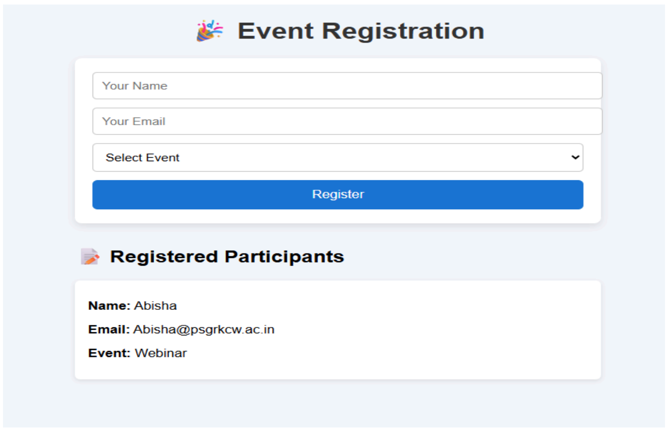
No registrations yet.



localhost:5175 says

✅ Registration Successful!

Name: Abisha
Email: Abisha@psgrkcw.ac.in
Event: Webinar


🎉 Event Registration

Your Name
Your Email
Select Event
Register

📝 Registered Participants

Name: Abisha
Email: Abisha@psgrkcw.ac.in
Event: Webinar

\

Task 13: Implement a simple user profile editing system with form validation.

App.jsx

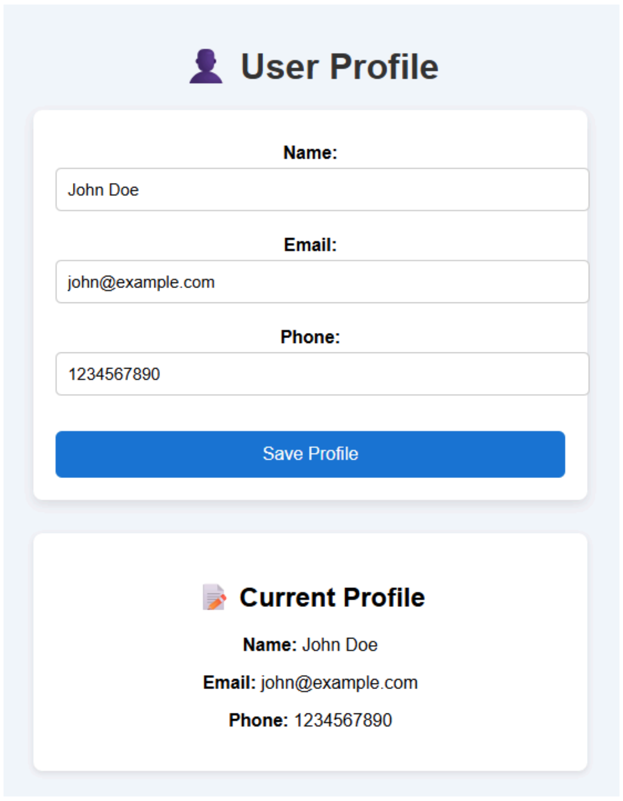
```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [profile, setProfile] = useState({
    name: "John Doe",
    email: "john@example.com",
    phone: "1234567890",
  });

  const [form, setForm] = useState(profile);
  const [errors, setErrors] = useState({});

  const validate = () => {
    const newErrors = {};

    if (!form.name.trim()) newErrors.name = "Name is required.";
    if (!form.email.match(/^\S+@\S+\.\S+$/)) newErrors.email = "Invalid email.";
    if (!form.phone.match(/^\d{10}$/)) newErrors.phone = "Phone must be 10 digits.";

    setErrors(newErrors);
    return Object.keys(newErrors).length === 0;
```

```
};

const handleSave = (e) => {
  e.preventDefault();
  if (!validate()) return;
  setProfile(form);
  alert("✅ Profile updated successfully!");
};

return (
  <div className="app">
    <h1>👤 User Profile</h1>

    <form className="form" onSubmit={handleSave}>
      <label>Name:</label>
      <input
        type="text"
        value={form.name}
        onChange={(e) => setForm({ ...form, name: e.target.value })}
      />
      {errors.name && <p className="error">{errors.name}</p>}

      <label>Email:</label>
      <input
        type="email"
        value={form.email}
        onChange={(e) => setForm({ ...form, email: e.target.value })}
      />
      {errors.email && <p className="error">{errors.email}</p>}

      <label>Phone:</label>
      <input
        type="text"
        value={form.phone}
        onChange={(e) => setForm({ ...form, phone: e.target.value })}
      />
      {errors.phone && <p className="error">{errors.phone}</p>}

      <button type="submit">Save Profile</button>
    </form>
```

```
      <div className="profile-display">
        <h2>📝 Current Profile</h2>
        <p><strong>Name:</strong> {profile.name}</p>
        <p><strong>Email:</strong> {profile.email}</p>
        <p><strong>Phone:</strong> {profile.phone}</p>
      </div>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 500px;
  margin: auto;
  padding: 30px;
}

h1 {
  text-align: center;
  color: #333;
  margin-bottom: 20px;
}

.form {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  margin-bottom: 30px;
}

.form label {
  display: block;
  margin-top: 10px;
  font-weight: bold;
}

.form input {
  width: 100%;
  padding: 10px;
  margin-top: 4px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

button {
  margin-top: 20px;
  width: 100%;
  padding: 12px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}

.error {
  color: #e53935;
  margin-top: 4px;
  font-size: 14px;
}

.profile-display {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 2px 6px rgba(0,0,0,0.1);
}
```


👤 User Profile

Name:
John Doe

Email:
john@example.com

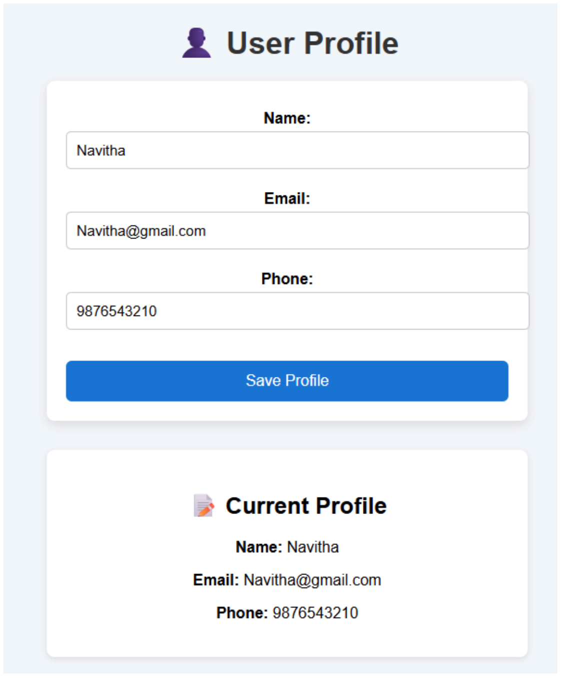
Phone:
1234567890

📝 Current Profile

Name: John Doe

Email: john@example.com

Phone: 1234567890


👤 User Profile

Name:
Navitha

Email:
Navitha@gmail.com

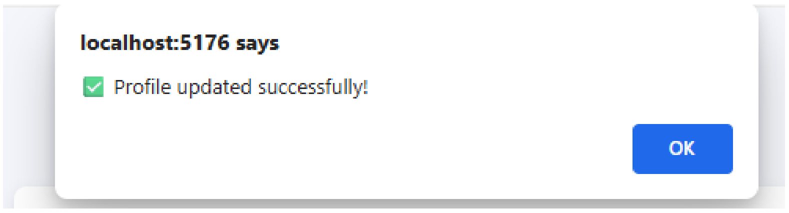
Phone:
9876543210

📝 Current Profile

Name: Navitha

Email: Navitha@gmail.com

Phone: 9876543210


localhost:5176 says

✅ Profile updated successfully!

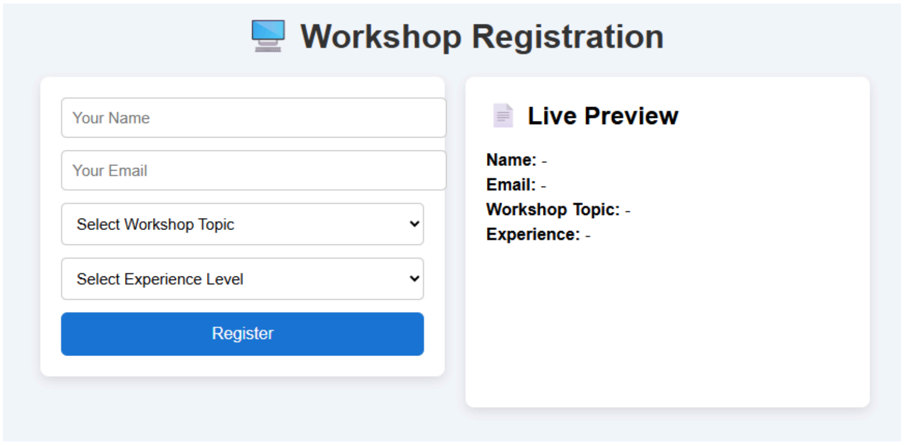
Task 14: Design a dynamic registration form for an online workshop with live preview.

App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [form, setForm] = useState({
    name: "",
    email: "",
    topic: "",
    experience: "",
  });

  const handleChange = (e) => {
    setForm({ ...form, [e.target.name]: e.target.value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();

    if (!form.name || !form.email || !form.topic || !form.experience) {
      alert("❌ Please fill all fields before submitting.");
      return;
    }

    alert(
      `✅ Registration Successful!\n\nName: ${form.name}\nEmail: ${form.email}\nTopic:
${form.topic}\nExperience: ${form.experience}`
    );

    setForm({ name: "", email: "", topic: "", experience: "" });
  };

  return (
    <div className="app">
      <h1>🖥️ Workshop Registration</h1>

      <div className="container">
        <form className="form" onSubmit={handleSubmit}>
          <input
            type="text"
            name="name"
            placeholder="Your Name"
            value={form.name}
            onChange={handleChange}
          />
          <input
            type="email"
            name="email"
            placeholder="Your Email"
            value={form.email}
            onChange={handleChange}
          />

          <select name="topic" value={form.topic} onChange={handleChange}>
            <option value="">Select Workshop Topic</option>
            <option value="React Basics">React Basics</option>
            <option value="Advanced JavaScript">Advanced JavaScript</option>
            <option value="UI/UX Design">UI/UX Design</option>
          </select>

          <select
            name="experience"
            value={form.experience}
            onChange={handleChange}
          >
            <option value="">Select Experience Level</option>
            <option value="Beginner">Beginner</option>
            <option value="Intermediate">Intermediate</option>
            <option value="Advanced">Advanced</option>
          </select>

          <button type="submit">Register</button>
        </form>

        <div className="preview">
          <h2>📄 Live Preview</h2>
          <p><strong>Name:</strong> {form.name || "-"}</p>
          <p><strong>Email:</strong> {form.email || "-"}</p>
          <p><strong>Workshop Topic:</strong> {form.topic || "-"}</p>
          <p><strong>Experience:</strong> {form.experience || "-"}</p>
        </div>
      </div>
    </div>
  );
}
```

App.css

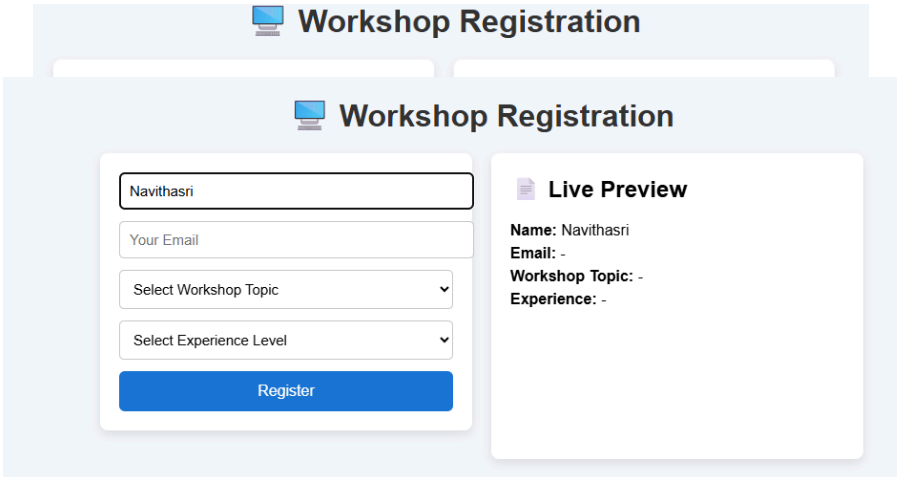
```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 800px;
  margin: auto;
  padding: 30px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.container {
  display: flex;
  gap: 20px;
  justify-content: center;
  flex-wrap: wrap;
}

.form {
  background: white;
  padding: 20px;
  border-radius: 8px;
  flex: 1;
  min-width: 300px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  display: flex;
  flex-direction: column;
}

.form input,
.form select {
  margin-bottom: 12px;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

button {
  padding: 12px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}

.preview {
  background: white;
  padding: 20px;
  border-radius: 8px;
  flex: 1;
  min-width: 250px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  text-align: left;
}

.preview h2 {
  margin-top: 0;
}

.preview p {
  margin: 6px 0;
}
```


🖥️ Workshop Registration

Your Name
Your Email
Select Workshop Topic
Select Experience Level
Register

Live Preview

Name: -
Email: -
Workshop Topic: -
Experience: -

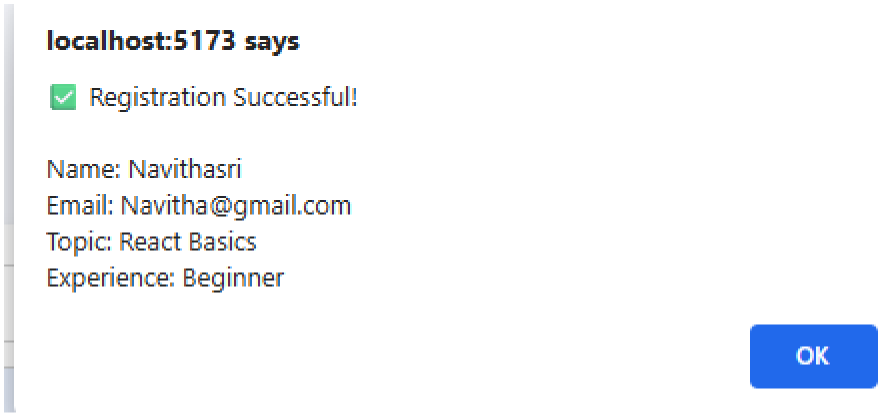

🖥️ Workshop Registration

🖥️ Workshop Registration

Navithasri
Your Email
Select Workshop Topic
Select Experience Level
Register

Live Preview

Name: Navithasri
Email: -
Workshop Topic: -
Experience: -


localhost:5173 says

✅ Registration Successful!

Name: Navithasri
Email: Navitha@gmail.com
Topic: React Basics
Experience: Beginner

OK

Task 15: Design a login interface that redirects users based on role selection.
App.jsx

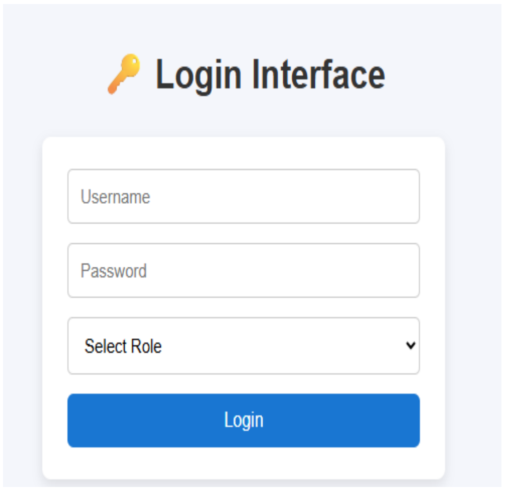
```
import { useState } from "react";
import "./App.css";

export default function App() {
```

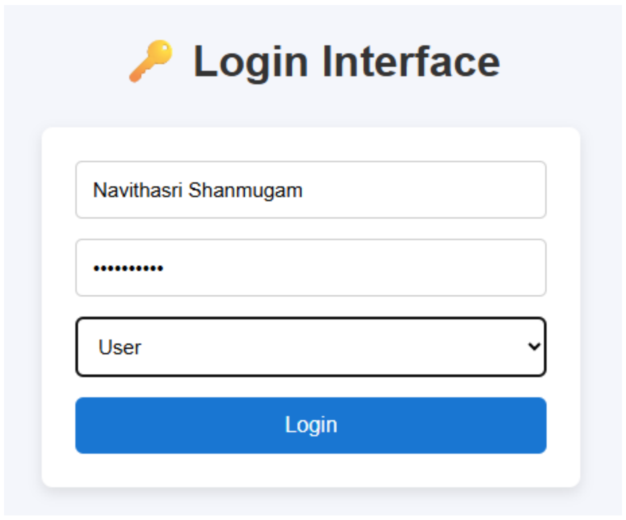
```
const [username, setUsername] = useState("");
const [password, setPassword] = useState("");
const [role, setRole] = useState("");

const handleLogin = (e) => {
  e.preventDefault();

  if (!username || !password || !role) {
    alert("❌ Please fill all fields and select a role.");
    return;
  }

  // Simulate role-based redirection
  if (role === "Admin") {
    alert(`✅ Welcome Admin ${username}!\nRedirecting to Admin Dashboard...`);
  } else {
    alert(`✅ Welcome ${username}!\nRedirecting to User Homepage...`);
  }

  // Clear form
  setUsername("");
  setPassword("");
  setRole("");
};

return (
  <div className="app">
    <h1>🔑 Login Interface</h1>

    <form className="login-form" onSubmit={handleLogin}>
      <input
        type="text"
        placeholder="Username"
        value={username}
        onChange={(e) => setUsername(e.target.value)}
      />

      <input
        type="password"
        placeholder="Password"
```

```
          value={password}
          onChange={(e) => setPassword(e.target.value)}
        />

        <select value={role} onChange={(e) => setRole(e.target.value)}>
          <option value="">Select Role</option>
          <option value="Admin">Admin</option>
          <option value="User">User</option>
        </select>

        <button type="submit">Login</button>
      </form>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 400px;
  margin: auto;
  padding: 50px 20px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 30px;
}

.login-form {
  background: white;
  padding: 25px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  display: flex;
  flex-direction: column;
}

.login-form input,
.login-form select {
  margin-bottom: 15px;
  padding: 12px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

button {
  padding: 12px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}
```

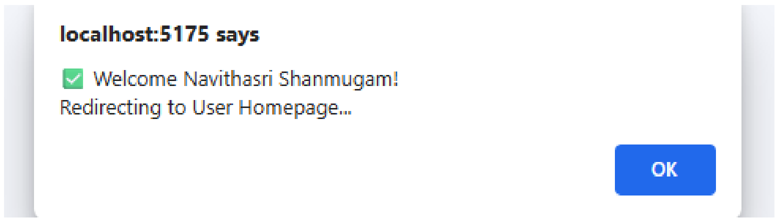

🔑 Login Interface


🔑 Login Interface

Navithasri Shanmugam

••••••••••

User



localhost:5175 says

✅ Welcome Navithasri Shanmugam!
Redirecting to User Homepage...

Task 16: Simulate a basic authentication system with error alerts.

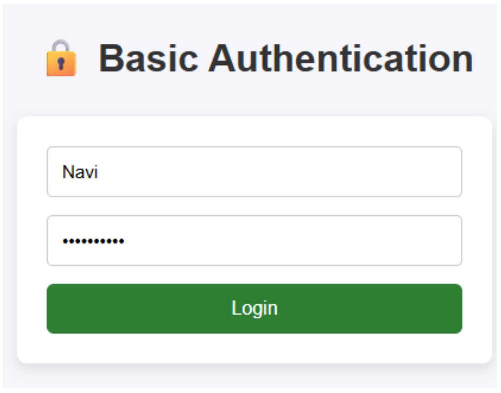
Apo.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [username, setUsername] = useState("");
  const [password, setPassword] = useState("");

  // Hardcoded credentials
  const validUser = {
    username: "admin",
    password: "12345",
  };

  const handleLogin = (e) => {
    e.preventDefault();

    if (!username || !password) {
      alert("❌ Please enter both username and password.");
      return;
    }

    if (username === validUser.username && password === validUser.password) {
      alert(`✅ Login Successful! Welcome ${username}.`);
      // Clear form
      setUsername("");
      setPassword("");
    } else {
      alert("❌ Invalid username or password.");
    }
  };
  return (
    <div className="app">
      <h1>🔒 Basic Authentication</h1>

      <form className="auth-form" onSubmit={handleLogin}>
        <input
          type="text"
          placeholder="Username"
          value={username}
          onChange={(e) => setUsername(e.target.value)}
        />

        <input
          type="password"
          placeholder="Password"
          value={password}
          onChange={(e) => setPassword(e.target.value)}
        />

        <button type="submit">Login</button>
      </form>
    </div>
  );
}
```

App.css

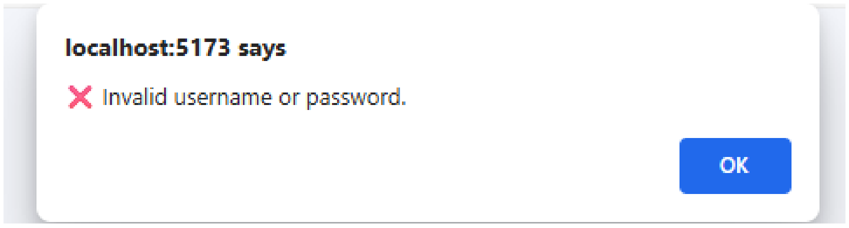
```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 400px;
  margin: auto;
  padding: 50px 20px;
  text-align: center;
}
```

```css
h1 {
  color: #333;
  margin-bottom: 30px;
}

.auth-form {
  background: white;
  padding: 25px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  display: flex;
  flex-direction: column;
}

.auth-form input {
  margin-bottom: 15px;
  padding: 12px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

button {
  padding: 12px;
  background: #2e7d32;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

button:hover {
  background: #1b5e20;
}
```


Basic Authentication

Navi

••••••••••


localhost:5173 says

❌ Invalid username or password.

Task 17: Implement a student feedback system with a rating and comments section.

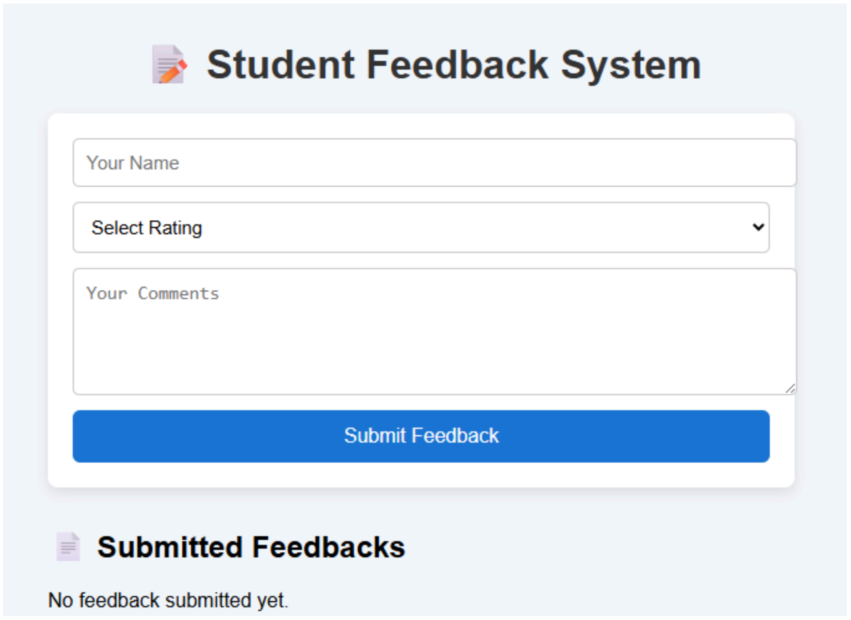
App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [name, setName] = useState("");
  const [rating, setRating] = useState("");
  const [comment, setComment] = useState("");
  const [feedbacks, setFeedbacks] = useState([]);

  const handleSubmit = (e) => {
    e.preventDefault();

    if (!name || !rating || !comment) {
      alert("❌ Please fill all fields before submitting.");
      return;
    }

    const newFeedback = {
      id: Date.now(),
      name,
      rating,
      comment,
    };

    setFeedbacks([newFeedback, ...feedbacks]);
    alert("✅ Feedback submitted successfully!");

    // Clear form
    setName("");
    setRating("");
    setComment("");
  };

  return (
    <div className="app">
      <h1>📝 Student Feedback System</h1>

      <form className="feedback-form" onSubmit={handleSubmit}>
        <input
          type="text"
          placeholder="Your Name"
          value={name}
          onChange={(e) => setName(e.target.value)}
        />

        <select value={rating} onChange={(e) => setRating(e.target.value)}>
          <option value="">Select Rating</option>
          <option value="1">1 ⭐</option>
          <option value="2">2 ⭐⭐</option>
          <option value="3">3 ⭐⭐⭐</option>
          <option value="4">4 ⭐⭐⭐⭐</option>
          <option value="5">5 ⭐⭐⭐⭐⭐</option>
        </select>

        <textarea
          placeholder="Your Comments"
          value={comment}
          onChange={(e) => setComment(e.target.value)}
        />

        <button type="submit">Submit Feedback</button>
      </form>

      <div className="feedback-list">
```

```
        <h2>📄 Submitted Feedbacks</h2>
        {feedbacks.length === 0 && <p>No feedback submitted yet.</p>}

        {feedbacks.map((f) => (
          <div key={f.id} className="feedback-card">
            <p><strong>Name:</strong> {f.name}</p>
            <p><strong>Rating:</strong> {f.rating} ⭐</p>
            <p><strong>Comments:</strong> {f.comment}</p>
          </div>
        ))}
      </div>
    </div>
  );
}
```

App.css

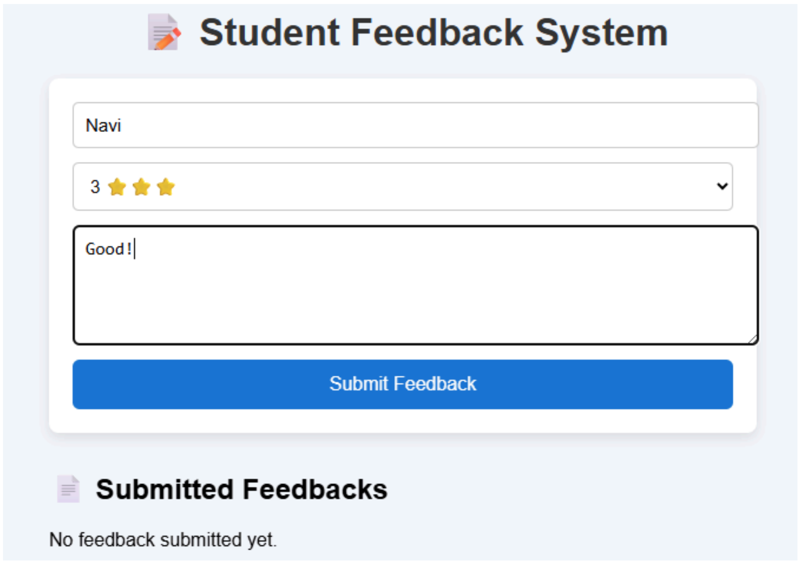
```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 600px;
  margin: auto;
  padding: 30px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.feedback-form {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  margin-bottom: 30px;
  display: flex;
  flex-direction: column;
}

.feedback-form input,
.feedback-form select,
.feedback-form textarea {
  width: 100%;
  margin-bottom: 12px;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
}

.feedback-form textarea {
  resize: vertical;
  min-height: 80px;
}

button {
  padding: 12px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}

.feedback-list {
  text-align: left;
}

.feedback-card {
  background: white;
  padding: 12px 15px;
  margin-bottom: 10px;
  border-radius: 6px;
  box-shadow: 0 2px 6px rgba(0,0,0,0.1);
}
```



Student Feedback System

Submitted Feedbacks

No feedback submitted yet.



Student Feedback System

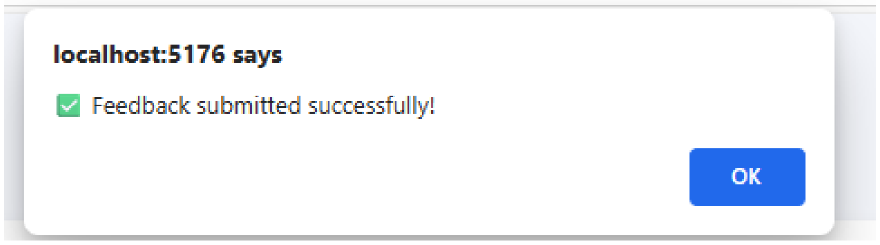
Navi

3 ⭐⭐⭐

Good!

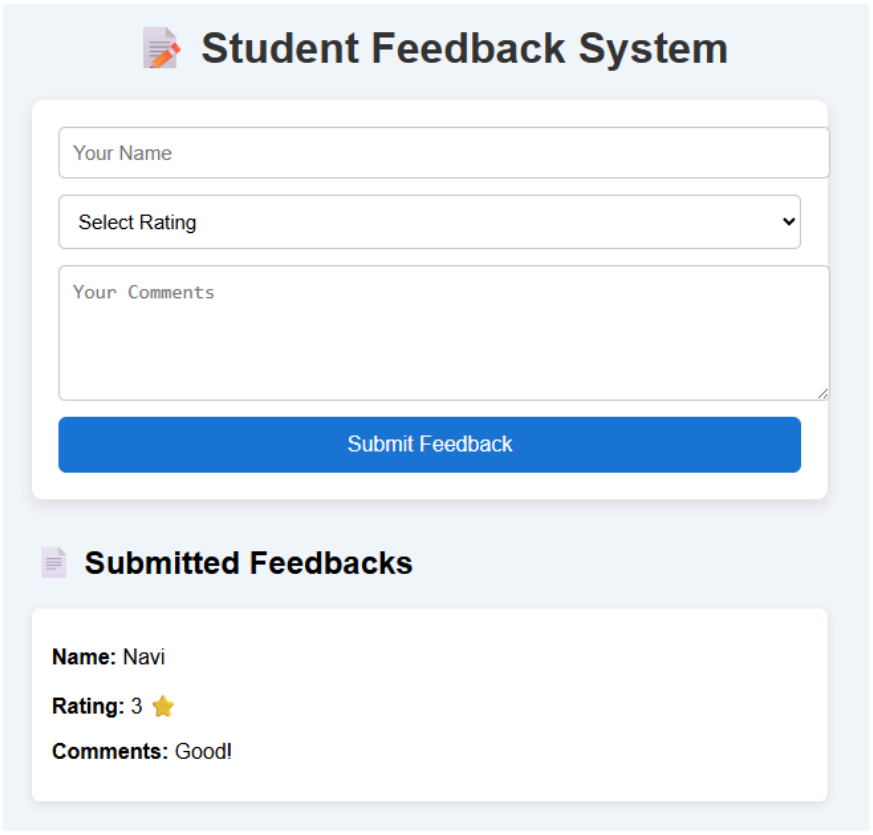
Submitted Feedbacks

No feedback submitted yet.



localhost:5176 says

✅ Feedback submitted successfully!



Name: Navi

Rating: 3 ⭐

Comments: Good!

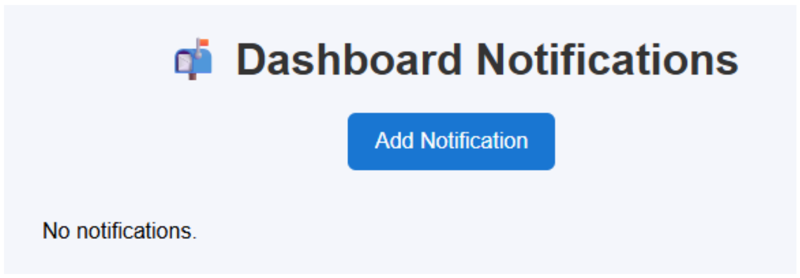
Task 18: Simulate a notification system for a dashboard application.

App.jsx

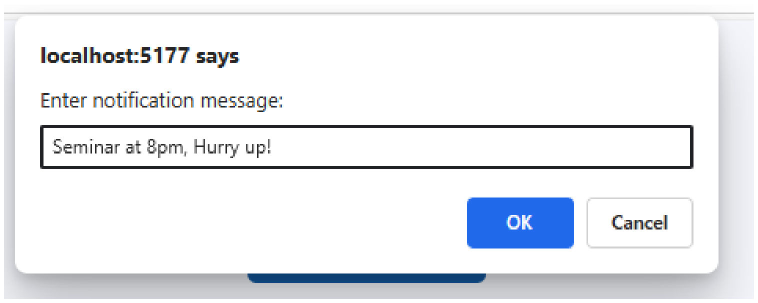
```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [notifications, setNotifications] = useState([]);

  const addNotification = () => {
    const message = prompt("Enter notification message:");
    if (!message) return;

    const newNotification = {
      id: Date.now(),
      message,
    };

    setNotifications([newNotification, ...notifications]);
  };

  const dismissNotification = (id) => {
    setNotifications(notifications.filter((n) => n.id !== id));
  };

  return (
    <div className="app">
      <h1>📬 Dashboard Notifications</h1>
      <button onClick={addNotification}>Add Notification</button>

      <div className="notifications">
        {notifications.length === 0 && <p>No notifications.</p>}

        {notifications.map((n) => (
          <div key={n.id} className="notification">
            <span>{n.message}</span>
            <button onClick={() => dismissNotification(n.id)}>Dismiss</button>
          </div>
        ))}
      </div
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 600px;
  margin: auto;
  padding: 40px 20px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

button {
  padding: 12px 20px;
  margin-bottom: 20px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  font-size: 16px;
  cursor: pointer;
}

button:hover {
  background: #125aa3;
}

.notifications {
  text-align: left;
}

.notification {
  display: flex;
  justify-content: space-between;
  align-items: center;
  background: white;
  padding: 12px 15px;
  margin-bottom: 10px;
  border-radius: 6px;
  box-shadow: 0 2px 6px rgba(0,0,0,0.1);
}

.notification button {
  padding: 6px 10px;
  background: #e53935;
  color: white;
  border: none;
  border-radius: 4px;
  cursor: pointer;
}

.notification button:hover {
  background: #b71c1c;
}
```

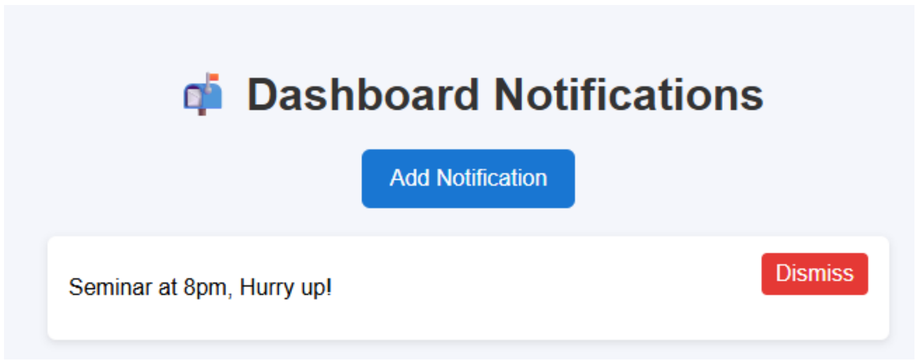


No notifications.


localhost:5177 says

Enter notification message:

Seminar at 8pm, Hurry up!



Seminar at 8pm, Hurry up!

Task 19: Construct a responsive blog layout with real-time comment update.
App.jsx

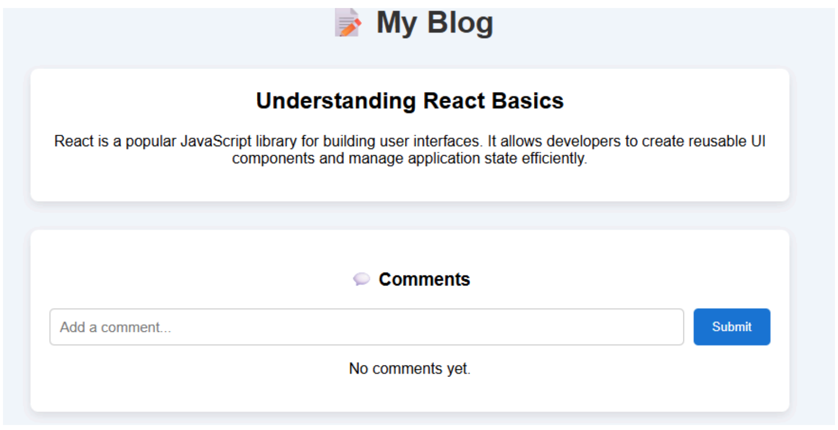
```
import { useState } from "react";
import "./App.css";
```

```
export default function App() {
  const [comments, setComments] = useState([]);
  const [commentInput, setCommentInput] = useState("");

  const handleCommentSubmit = (e) => {
    e.preventDefault();
    if (!commentInput.trim()) return;

    const newComment = {
      id: Date.now(),
      text: commentInput,
    };

    setComments([newComment, ...comments]);
    setCommentInput("");
  };

  return (
    <div className="app">
      <h1>📝 My Blog</h1>

      <div className="blog-post">
        <h2>Understanding React Basics</h2>
        <p>
          React is a popular JavaScript library for building user interfaces.
          It allows developers to create reusable UI components and manage
          application state efficiently.
        </p>
      </div>

      <div className="comments-section">
        <h3>💬 Comments</h3>
        <form onSubmit={handleCommentSubmit}>
          <input
            type="text"
            placeholder="Add a comment..."
            value={commentInput}
            onChange={(e) => setCommentInput(e.target.value)}
          />
          <button type="submit">Submit</button>
        </form>

        {comments.length === 0 && <p>No comments yet.</p>}
        {comments.map((c) => (
          <div key={c.id} className="comment">
            {c.text}
          </div>
        ))}
      </div>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 800px;
  margin: auto;
  padding: 20px;
}

h1 {
  text-align: center;
  color: #333;
  margin-bottom: 30px;
}

.blog-post {
  background: white;
  padding: 20px;
  border-radius: 8px;
  margin-bottom: 30px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}
```

```
.blog-post h2 {
  margin-top: 0;
}

.comments-section {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.comments-section form {
  display: flex;
  flex-wrap: wrap;
  margin-bottom: 15px;
}

.comments-section input {
  flex: 1;
  padding: 10px;
  font-size: 15px;
  border-radius: 5px;
  border: 1px solid #ccc;
  margin-right: 10px;
}

.comments-section button {
  padding: 10px 20px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 5px;
  cursor: pointer;
}

.comments-section button:hover {
  background: #125aa3;
}

.comment {
  background: #f0f4ff;
  padding: 10px 15px;
  margin-bottom: 10px;
  border-radius: 6px;
  word-wrap: break-word;
}

/* Responsive */
@media (max-width: 600px) {
  .comments-section form {
    flex-direction: column;
  }

  .comments-section input {
    margin-right: 0;
    margin-bottom: 10px;
  }
}
```



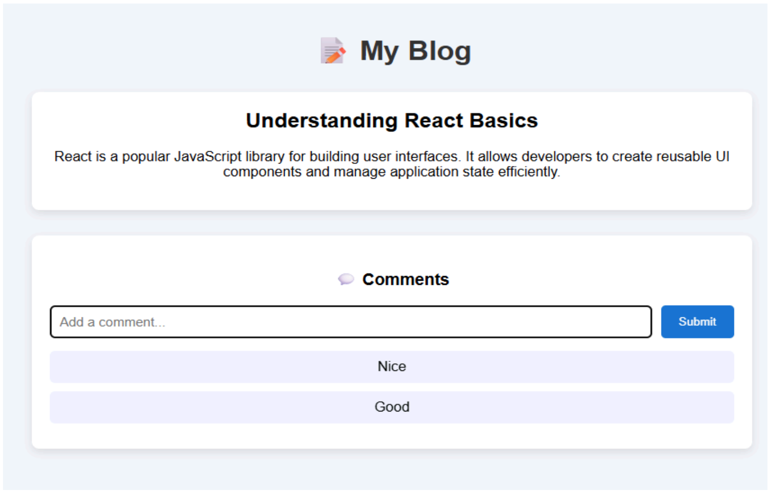
📝 My Blog

Understanding React Basics

React is a popular JavaScript library for building user interfaces. It allows developers to create reusable UI components and manage application state efficiently.

💬 Comments

No comments yet.



📝 My Blog

Understanding React Basics

React is a popular JavaScript library for building user interfaces. It allows developers to create reusable UI components and manage application state efficiently.

💬 Comments

Nice

Good

Task 20: Simulate a basic e-commerce product listing interface with add-to-cart feature.

App.jsx

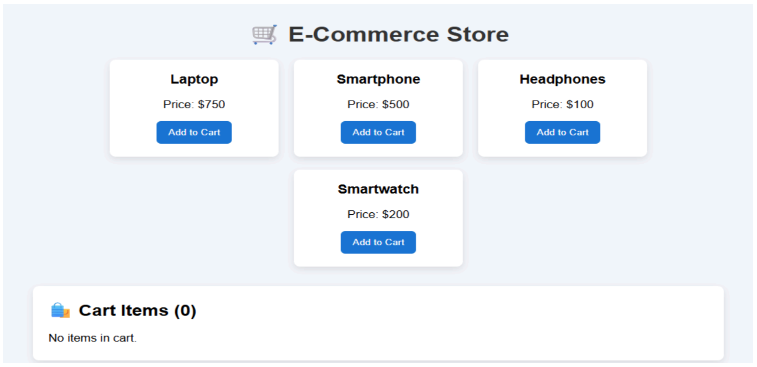
```
import { useState } from "react";
import "./App.css";

export default function App() {
  const products = [
    { id: 1, name: "Laptop", price: 750 },
    { id: 2, name: "Smartphone", price: 500 },
    { id: 3, name: "Headphones", price: 100 },
    { id: 4, name: "Smartwatch", price: 200 },
  ];

  const [cart, setCart] = useState([]);

  const addToCart = (product) => {
    setCart([...cart, product]);
  };

  return (
    <div className="app">
      <h1>🛒 E-Commerce Store</h1>

      <div className="product-list">
        {products.map((p) => (
          <div key={p.id} className="product-card">
```

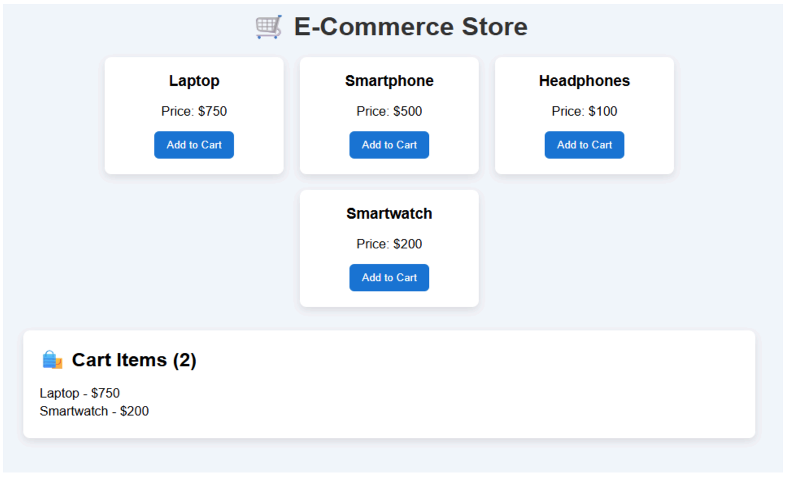
```
            <h3>{p.name}</h3>
            <p>Price: ${p.price}</p>
            <button onClick={() => addToCart(p)}>Add to Cart</button>
          </div>
        ))}
      </div>

      <div className="cart">
        <h2>🛍️ Cart Items ({cart.length})</h2>
        {cart.length === 0 && <p>No items in cart.</p>}
        {cart.map((item, index) => (
          <p key={index}>
            {item.name} - ${item.price}
          </p>
        ))}
      </div>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 900px;
  margin: auto;
  padding: 30px 20px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.product-list {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  justify-content: center;
  margin-bottom: 30px;
}

.product-card {
  background: white;
  padding: 20px;
  border-radius: 8px;
  width: 180px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.product-card h3 {
  margin-top: 0;
}

.product-card button {
  padding: 10px 15px;
  background: #1976d2;
  color: white;
  border: none;
  border-radius: 6px;
  cursor: pointer;
}

.product-card button:hover {
  background: #125aa3;
}

.cart {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  text-align: left;
}
```

```
.cart h2 {
  margin-top: 0;
}

.cart p {
  margin: 5px 0;
}
```





Task 21: Design a simple student management dashboard with navigation and data display.

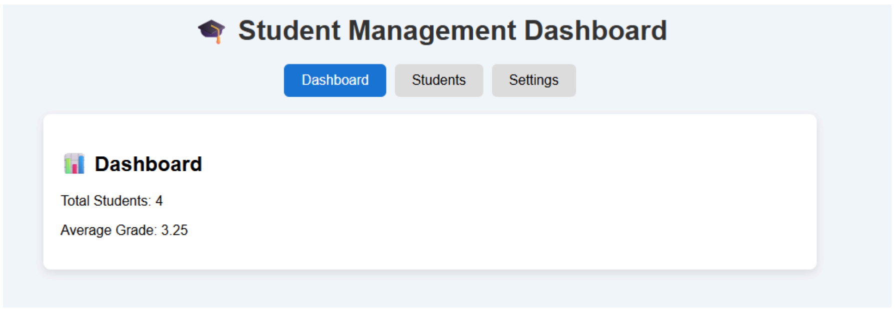
App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const [activeTab, setActiveTab] = useState("Dashboard");
const students = [
  { id: 1, name: "Alice Johnson", grade: "A" },
  { id: 2, name: "Bob Smith", grade: "B" },
  { id: 3, name: "Charlie Brown", grade: "A" },
  { id: 4, name: "Diana Prince", grade: "C" },
];

const renderContent = () => {
  switch (activeTab) {
    case "Dashboard":
      return (
        <div>
          <h2>📊 Dashboard</h2>
          <p>Total Students: {students.length}</p>
          <p>Average Grade: {calculateAverageGrade()}</p>
        </div>
      );
    case "Students":
      return (
        <div>
          <h2>👩‍🎓 Students List</h2>
          <table>
            <thead>
              <tr>
                <th>ID</th>
                <th>Name</th>
                <th>Grade</th>
              </tr>
            </thead>
            <tbody>
              {students.map((s) => (
                <tr key={s.id}>
                  <td>{s.id}</td>
                  <td>{s.name}</td>
                  <td>{s.grade}</td>
                </tr>
              ))}
            </tbody>
          </table>
        </div>
      );
    case "Settings":
      return (
        <div>
          <h2>⚙️ Settings</h2>
          <p>Here you can configure dashboard settings.</p>
        </div>
      );
    default:
      return null;
  }
};

const calculateAverageGrade = () => {
  const gradePoints = { A: 4, B: 3, C: 2, D: 1, F: 0 };
  const total = students.reduce((acc, s) => acc + gradePoints[s.grade], 0);
  return (total / students.length).toFixed(2);
};

return (
  <div className="app">
    <h1>🎓 Student Management Dashboard</h1>
    <div className="tabs">
      {["Dashboard", "Students", "Settings"].map((tab) => (
        <button
          key={tab}
          className={activeTab === tab ? "active" : ""}
          onClick={() => setActiveTab(tab)}
        >
          {tab}
        </button>
      ))}
    </div>

    <div className="content">{renderContent()}</div>
  </div>
);
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 900px;
  margin: auto;
  padding: 30px 20px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.tabs {
  display: flex;
  justify-content: center;
  margin-bottom: 20px;
  flex-wrap: wrap;
  gap: 10px;
}

.tabs button {
  padding: 10px 20px;
  border: none;
  border-radius: 6px;
  background: #e0e0e0;
  cursor: pointer;
  font-size: 16px;
}

.tabs button.active {
  background: #1976d2;
  color: white;
}
```

```
.tabs button:hover {
  background: #bdbdbd;
}

.content {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  text-align: left;
}

table {
  width: 100%;
  border-collapse: collapse;
  margin-top: 10px;
}

table, th, td {
  border: 1px solid #ccc;
}

th, td {
  padding: 10px;
  text-align: left;
}

th {
  background: #f0f0f0;
}

@media (max-width: 600px) {
  table, th, td {
    font-size: 14px;
  }

  .tabs button {
    padding: 8px 12px;
    font-size: 14px;
  }
}
```

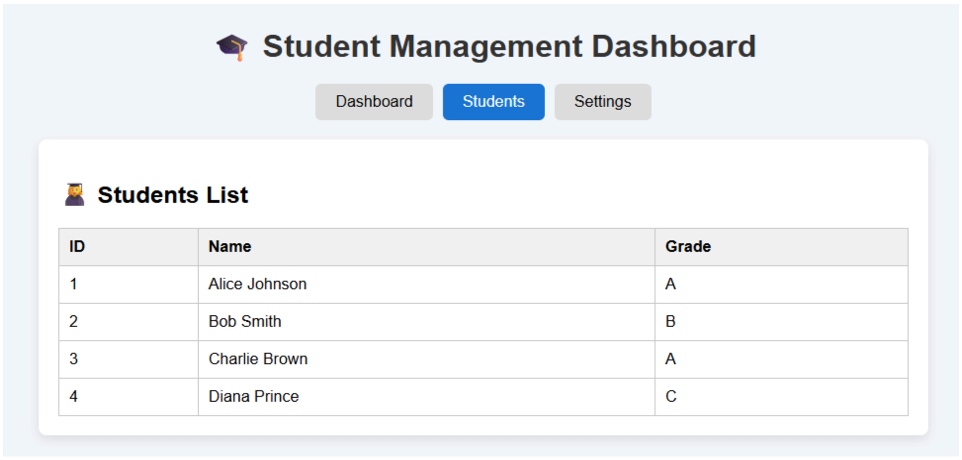


🎓 Student Management Dashboard

📊 Dashboard

Total Students: 4

Average Grade: 3.25



🎓 Student Management Dashboard

🧑‍🎓 Students List

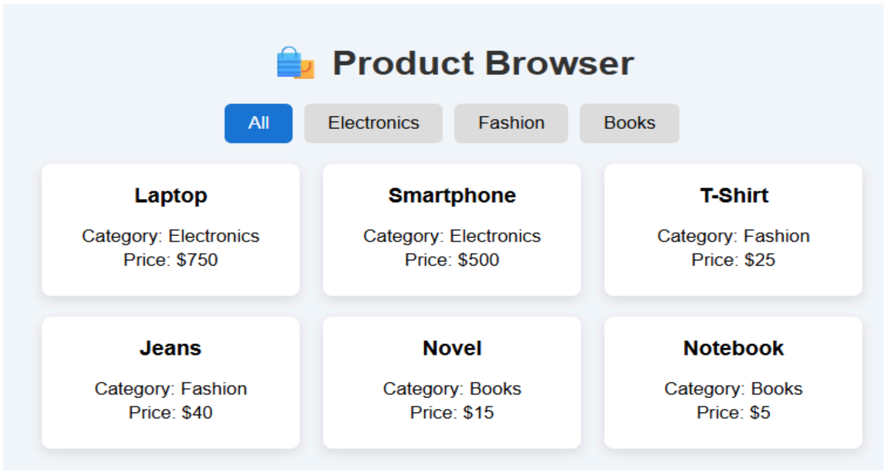
ID	Name	Grade
1	Alice Johnson	A
2	Bob Smith	B
3	Charlie Brown	A
4	Diana Prince	C

Task 22: Simulate a simple product browsing page with category-based routing.
App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
  const categories = ["All", "Electronics", "Fashion", "Books"];

  const products = [
    { id: 1, name: "Laptop", category: "Electronics", price: 750 },
    { id: 2, name: "Smartphone", category: "Electronics", price: 500 },
    { id: 3, name: "T-Shirt", category: "Fashion", price: 25 },
    { id: 4, name: "Jeans", category: "Fashion", price: 40 },
    { id: 5, name: "Novel", category: "Books", price: 15 },
    { id: 6, name: "Notebook", category: "Books", price: 5 },
  ];
  const [activeCategory, setActiveCategory] = useState("All");

  const filteredProducts =
    activeCategory === "All"
      ? products
      : products.filter((p) => p.category === activeCategory);

  return (
    <div className="app">
      <h1>🛍️ Product Browser</h1>

      <div className="category-buttons">
        {categories.map((cat) => (
          <button
            key={cat}
            className={activeCategory === cat ? "active" : ""}
            onClick={() => setActiveCategory(cat)}
          >
            {cat}
          </button>
        ))}
      </div>

      <div className="product-list">
        {filteredProducts.map((p) => (
          <div key={p.id} className="product-card">
            <h3>{p.name}</h3>
            <p>Category: {p.category}</p>
            <p>Price: ${p.price}</p>
          </div>
        ))}
        {filteredProducts.length === 0 && <p>No products in this category.</p>}
      </div>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 900px;
  margin: auto;
  padding: 30px 20px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.category-buttons {
  margin-bottom: 20px;
  display: flex;
  justify-content: center;
  flex-wrap: wrap;
  gap: 10px;
}

.category-buttons button {
  padding: 10px 20px;
  border: none;
  border-radius: 6px;
  background: #e0e0e0;
  cursor: pointer;
  font-size: 16px;
}

.category-buttons button.active {
  background: #1976d2;
  color: white;
}

.category-buttons button:hover {
  background: #bdbdbd;
}

.product-list {
  display: flex;
  flex-wrap: wrap;
  gap: 20px;
  justify-content: center;
}

.product-card {
  background: white;
  padding: 20px;
  border-radius: 8px;
  width: 180px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
}

.product-card h3 {
  margin-top: 0;
}

.product-card p {
  margin: 5px 0;
}

/* Responsive */
@media (max-width: 600px) {
  .product-card {
    width: 100%;
  }
}
```



Laptop
Category: Electronics
Price: $750

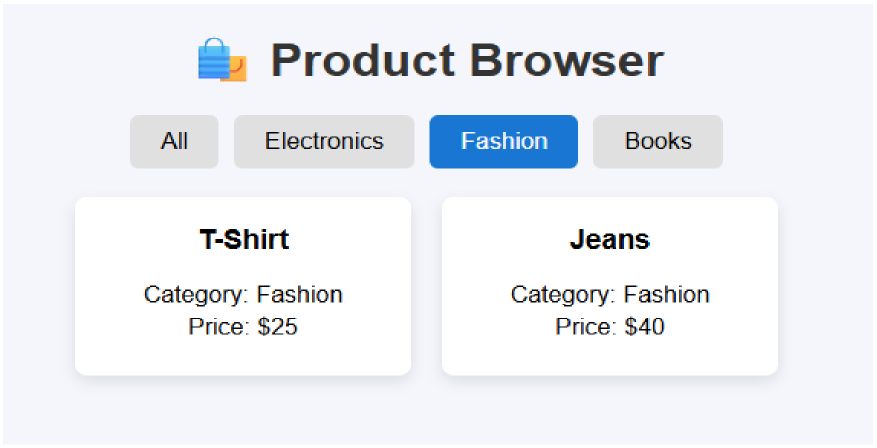
Smartphone
Category: Electronics
Price: $500

T-Shirt
Category: Fashion
Price: $25

Jeans
Category: Fashion
Price: $40

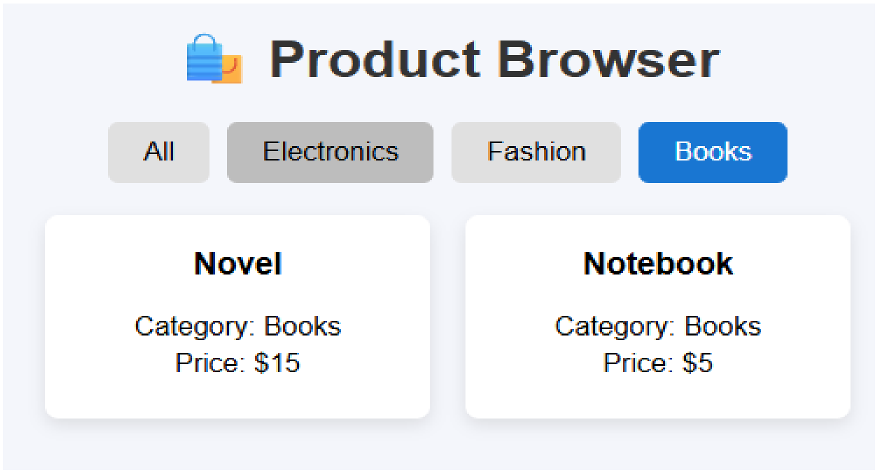
Novel
Category: Books
Price: $15

Notebook
Category: Books
Price: $5



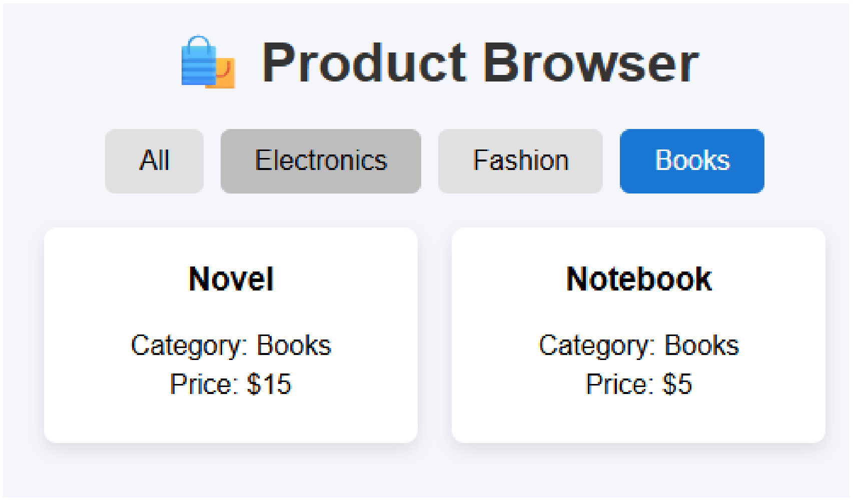
T-Shirt
Category: Fashion
Price: $25

Jeans
Category: Fashion
Price: $40



Novel
Category: Books
Price: $15

Notebook
Category: Books
Price: $5



Novel

Category: Books
Price: $15

Notebook

Category: Books
Price: $5

Task 23: Design a multi-page student dashboard with navigation using React Router.

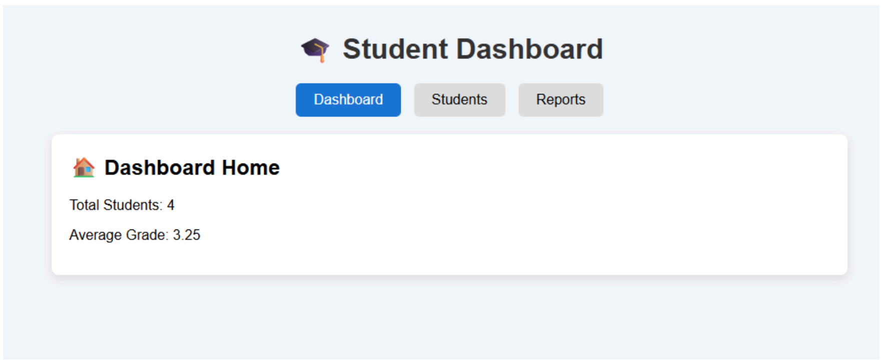
App.jsx

```
import { useState } from "react";
import "./App.css";

export default function App() {
 const [activeTab, setActiveTab] = useState("Dashboard");

 const students = [
  { id: 1, name: "Alice Johnson", grade: "A" },
  { id: 2, name: "Bob Smith", grade: "B" },
  { id: 3, name: "Charlie Brown", grade: "A" },
  { id: 4, name: "Diana Prince", grade: "C" },
 ];

 const renderContent = () => {
  switch (activeTab) {
   case "Dashboard":
    return (
     <div className="page">
      <h2>🏠 Dashboard Home</h2>
      <p>Total Students: {students.length}</p>
      <p>Average Grade: {calculateAverageGrade()}</p>
     </div>
    );
    case "Students":
      return (
        <div className="page">
          <h2>👩‍🎓 Students List</h2>
          <table>
            <thead>
              <tr>
                <th>ID</th>
                <th>Name</th>
                <th>Grade</th>
              </tr>
            </thead>
            <tbody>
              {students.map((s) => (
                <tr key={s.id}>
                  <td>{s.id}</td>
                  <td>{s.name}</td>
                  <td>{s.grade}</td>
                </tr>
              ))}
            </tbody>
          </table>
        </div>
      );
    case "Reports":
      return (
        <div className="page">
          <h2>📊 Reports</h2>
          <p>View analytics and performance reports of students.</p>
        </div>
      );
    default:
      return null;
  }
};

const calculateAverageGrade = () => {
  const gradePoints = { A: 4, B: 3, C: 2, D: 1, F: 0 };
  const total = students.reduce((acc, s) => acc + gradePoints[s.grade], 0);
  return (total / students.length).toFixed(2);
  };

  return (
    <div className="app">
      <h1>🎓 Student Dashboard</h1>

      <div className="tabs">
        {["Dashboard", "Students", "Reports"].map((tab) => (
          <button
            key={tab}
            className={activeTab === tab ? "active" : ""}
            onClick={() => setActiveTab(tab)}
          >
            {tab}
          </button>
        ))}
      </div>

      <div className="content">{renderContent()}</div>
    </div>
  );
}
```

App.css

```
body {
  margin: 0;
  font-family: Arial, sans-serif;
  background: #f4f6fb;
}

.app {
  max-width: 900px;
  margin: auto;
  padding: 30px 20px;
  text-align: center;
}

h1 {
  color: #333;
  margin-bottom: 20px;
}

.tabs {
  display: flex;
  justify-content: center;
  gap: 15px;
  margin-bottom: 20px;
  flex-wrap: wrap;
}

.tabs button {
  padding: 10px 20px;
  border: none;
  border-radius: 6px;
  background: #e0e0e0;
  cursor: pointer;
  font-size: 16px;
}

.tabs button.active {
  background: #1976d2;
  color: white;
}

.tabs button:hover {
  background: #bdbdbd;
}

.content {
  background: white;
  padding: 20px;
  border-radius: 8px;
  box-shadow: 0 4px 10px rgba(0,0,0,0.1);
  text-align: left;
}

.page h2 {
  margin-top: 0;
}
```

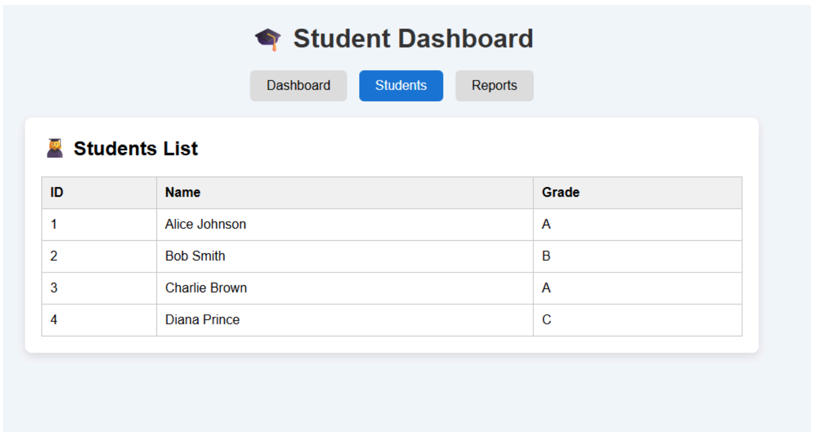
```
table {
  width: 100%;
  border-collapse: collapse;
  margin-top: 10px;
}

table, th, td {
  border: 1px solid #ccc;
}

th, td {
  padding: 10px;
  text-align: left;
}

th {
  background: #f0f0f0;
}

@media (max-width: 600px) {
  table, th, td {
    font-size: 14px;
  }

  .tabs button {
    padding: 8px 12px;
    font-size: 14px;
  }
}
```



🏠 Dashboard Home

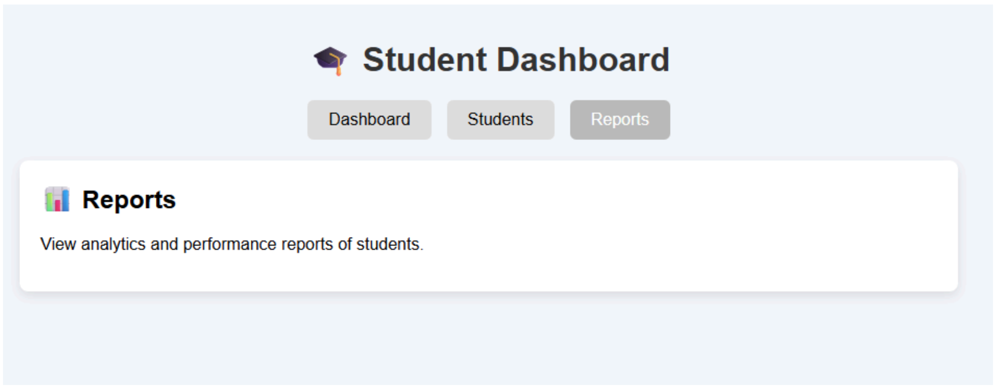
Total Students: 4

Average Grade: 3.25



🧑‍🎓 **Students List**

ID	Name	Grade
1	Alice Johnson	A
2	Bob Smith	B
3	Charlie Brown	A
4	Diana Prince	C



📊 **Reports**

View analytics and performance reports of students.