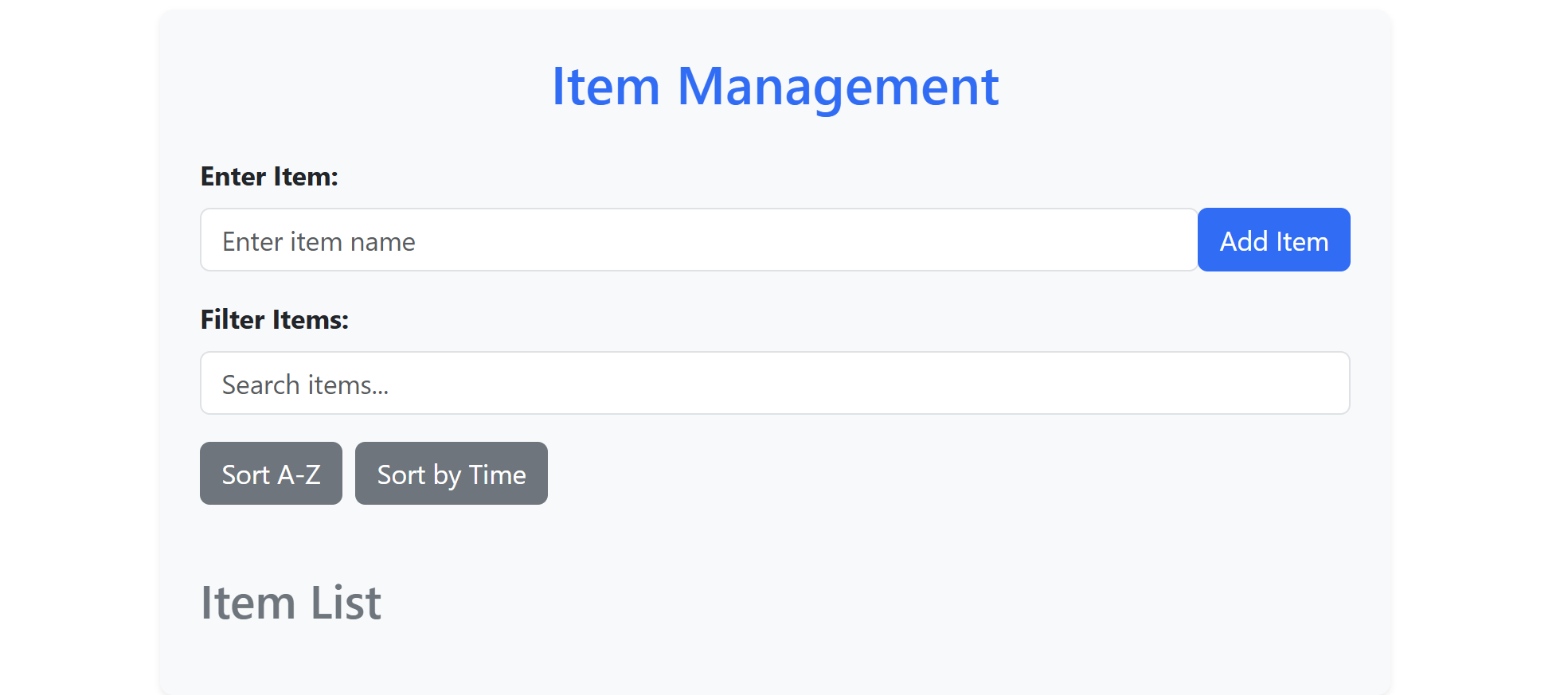
# Use Reducer

## Bài 4



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

import { Button, Form, Container, Row, Col, ListGroup, Card } from "react-bootstrap"; // Thêm Card vào import

function listReducer(state, action) {

  switch (action.type) {

    case "ADD\_ITEM":

      return { ...state, items: [...state.items, action.item] };

    case "REMOVE\_ITEM":

      return {

        ...state,

        items: state.items.filter((item) => item.id !== action.id),

      };

    case "EDIT\_ITEM":

      return {

        ...state,

        items: state.items.map((item) =>

          item.id === action.id ? { ...item, name: action.name } : item

        ),

      };

    default:

      return state;

  }

}

const initialState = { items: [] };

function ItemList() {

  const [state, dispatch] = useReducer(listReducer, initialState);

  const [newItemName, setNewItemName] = useState("");

  const [editItem, setEditItem] = useState(null);

  const [editName, setEditName] = useState("");

  const [filterText, setFilterText] = useState("");

  const handleAddItem = () => {

    if (newItemName.trim()) {

      const newItem = { id: Date.now(), name: newItemName.trim() };

      dispatch({ type: "ADD\_ITEM", item: newItem });

      setNewItemName("");

    }

  };

  const handleRemoveItem = (id) => dispatch({ type: "REMOVE\_ITEM", id });

  const handleEditSave = () => {

    if (editName.trim()) {

      dispatch({ type: "EDIT\_ITEM", id: editItem, name: editName.trim() });

      setEditItem(null);

      setEditName("");

    }

  };

  const handleSort = (type) => {

    const sortedItems = [...state.items];

    if (type === "alpha") sortedItems.sort((a, b) => a.name.localeCompare(b.name));

    else sortedItems.sort((a, b) => a.id - b.id);

    dispatch({ type: "ADD\_ITEM", item: { items: sortedItems } }); // Sử dụng ADD\_ITEM để cập nhật

  };

  return (

    <Container className="my-5">

      <Row className="justify-content-center">

        <Col md={8}>

          <Card className="p-4 shadow-sm border-0 rounded-3 bg-light">

            <h2 className="text-center mb-4 text-primary">Item Management</h2>

            <Form>

              <Form.Group controlId="formItem" className="mb-3">

                <Form.Label className="fw-bold">Enter Item:</Form.Label>

                <div className="input-group">

                  <Form.Control

                    type="text"

                    value={newItemName}

                    onChange={(e) => setNewItemName(e.target.value)}

                    placeholder="Enter item name"

                    className="rounded-end"

                  />

                  <Button variant="primary" onClick={handleAddItem} className="rounded-start">

                    Add Item

                  </Button>

                </div>

              </Form.Group>

              <Form.Group controlId="filterItem" className="mb-3">

                <Form.Label className="fw-bold">Filter Items:</Form.Label>

                <Form.Control

                  type="text"

                  value={filterText}

                  onChange={(e) => setFilterText(e.target.value)}

                  placeholder="Search items..."

                />

              </Form.Group>

              <div className="d-flex gap-2 mb-3">

                <Button variant="secondary" onClick={() => handleSort("alpha")}>Sort A-Z</Button>

                <Button variant="secondary" onClick={() => handleSort("time")}>Sort by Time</Button>

              </div>

            </Form>

            <h3 className="mt-4 mb-3 text-secondary">Item List</h3>

            <ListGroup>

              {state.items

                .filter((item) =>

                  item.name.toLowerCase().includes(filterText.toLowerCase())

                )

                .map((item) => (

                  <ListGroup.Item

                    key={item.id}

                    className="d-flex justify-content-between align-items-center p-3 border rounded mb-2 bg-white shadow-sm"

                  >

                    <span className="fw-semibold">{item.name}</span>

                    <div>

                      <Button

                        variant="danger"

                        size="sm"

                        onClick={() => handleRemoveItem(item.id)}

                        className="me-2"

                      >

                        Remove

                      </Button>

                      <Button

                        variant="warning"

                        size="sm"

                        onClick={() => {

                          setEditItem(item.id);

                          setEditName(item.name);

                        }}

                      >

                        Edit

                      </Button>

                    </div>

                  </ListGroup.Item>

                ))}

            </ListGroup>

            {editItem && (

              <Form className="mt-4">

                <Form.Group controlId="editItem" className="mb-3">

                  <Form.Label className="fw-bold">Edit Item:</Form.Label>

                  <div className="input-group">

                    <Form.Control

                      type="text"

                      value={editName}

                      onChange={(e) => setEditName(e.target.value)}

                      className="rounded-end"

                    />

                    <Button

                      variant="success"

                      onClick={handleEditSave}

                      className="rounded-start"

                    >

                      Save

                    </Button>

                    <Button

                      variant="secondary"

                      onClick={() => setEditItem(null)}

                      className="ms-2"

                    >

                      Cancel

                    </Button>

                  </div>

                </Form.Group>

              </Form>

            )}

          </Card>

        </Col>

      </Row>

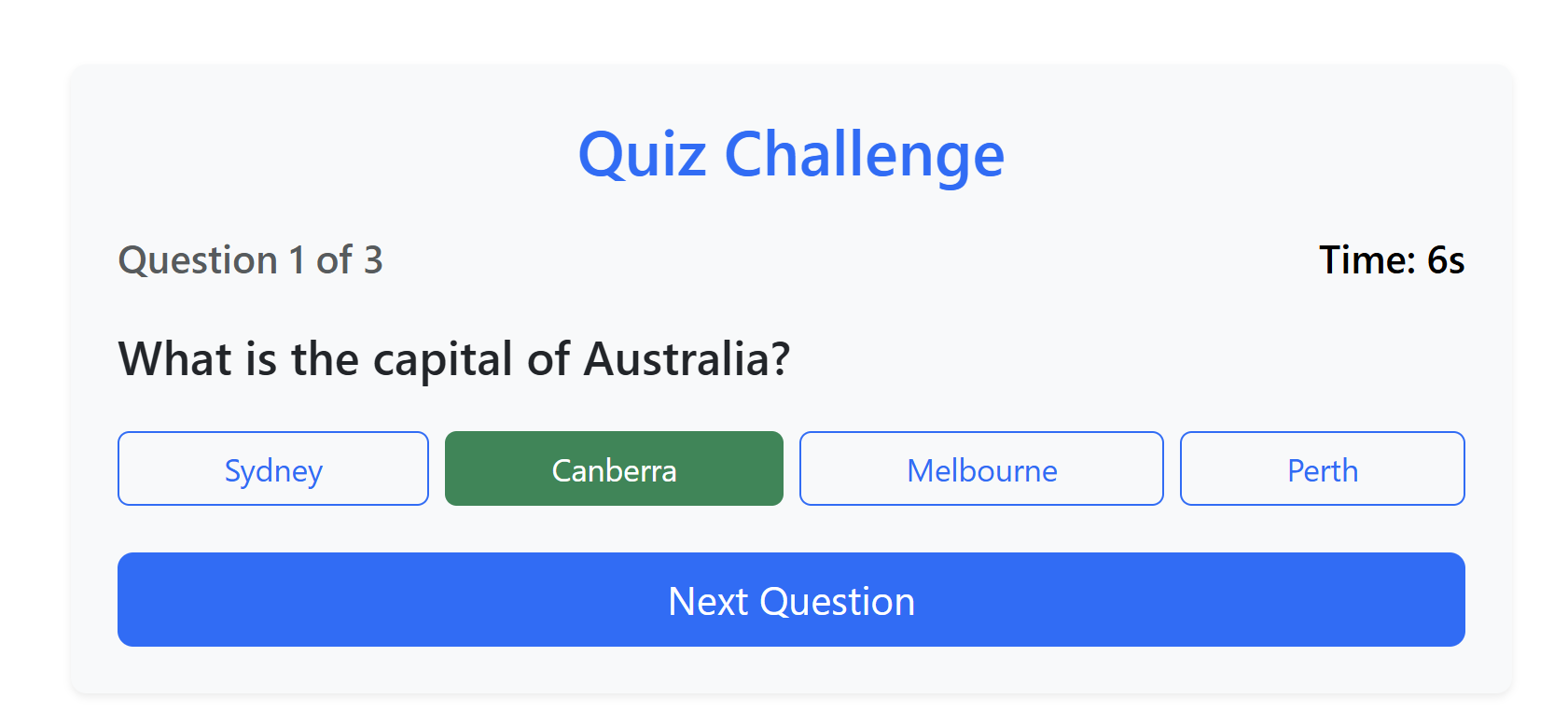
    </Container>

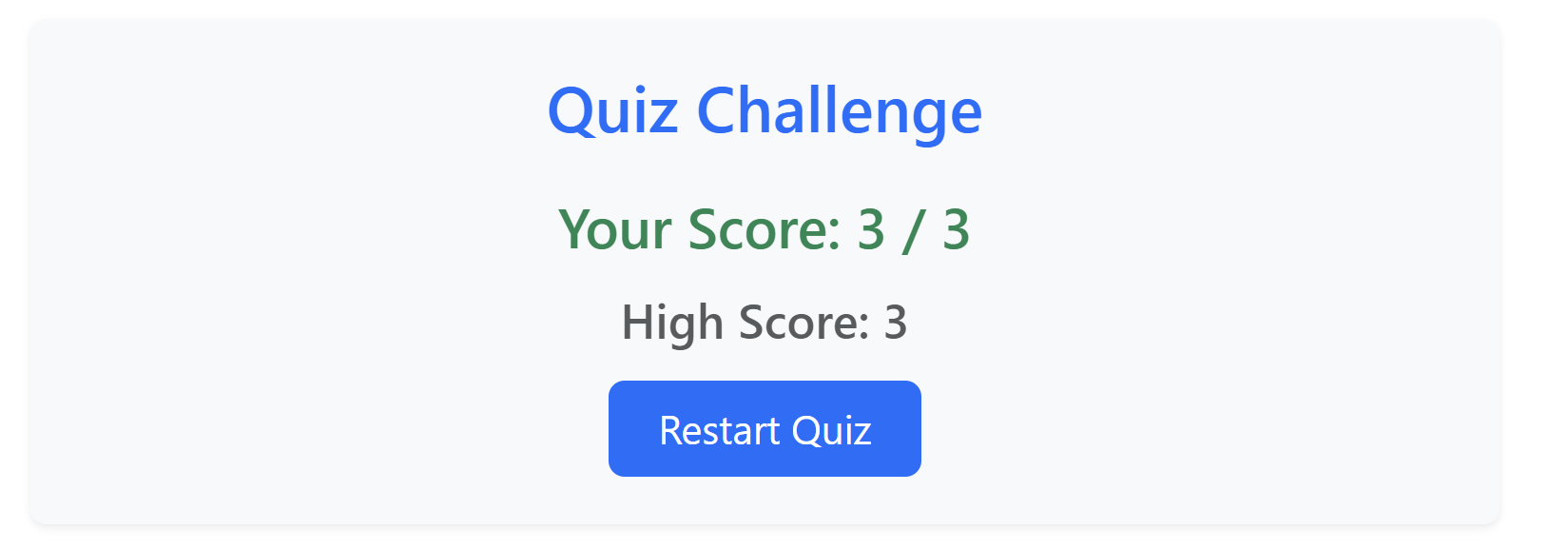
  );

}

export default ItemList;

## Bài 5 & 6





import React, { useReducer, useEffect, useState } from "react";

import { Button, Container, Card, Row, Col } from "react-bootstrap";

import { FaCheckCircle, FaTimesCircle } from "react-icons/fa";

const initialState = {

  questions: [

    {

      id: 1,

      question: "What is the capital of Australia?",

      options: ["Sydney", "Canberra", "Melbourne", "Perth"],

      answer: "Canberra",

    },

    {

      id: 2,

      question: "Which planet is known as the Red Planet?",

      options: ["Venus", "Mars", "Jupiter", "Saturn"],

      answer: "Mars",

    },

    {

      id: 3,

      question: "What is the largest ocean on Earth?",

      options: ["Atlantic Ocean", "Indian Ocean", "Pacific Ocean", "Arctic Ocean"],

      answer: "Pacific Ocean",

    },

  ],

  currentQuestion: 0,

  selectedOption: "",

  score: 0,

  showScore: false,

};

function quizReducer(state, action) {

  switch (action.type) {

    case "SELECT\_OPTION":

      return { ...state, selectedOption: action.payload };

    case "NEXT\_QUESTION":

      const isCorrect = state.selectedOption === state.questions[state.currentQuestion].answer;

      return {

        ...state,

        score: isCorrect ? state.score + 1 : state.score,

        currentQuestion: state.currentQuestion + 1,

        selectedOption: "",

        showScore: state.currentQuestion + 1 === state.questions.length,

      };

    case "RESTART\_QUIZ":

      return { ...initialState };

    default:

      return state;

  }

}

function QuestionBank() {

  const [state, dispatch] = useReducer(quizReducer, initialState);

  const { questions, currentQuestion, selectedOption, score, showScore } = state;

  const [feedback, setFeedback] = useState("");

  const [time, setTime] = useState(10);

  useEffect(() => {

    if (!showScore && time > 0) {

      const timer = setInterval(() => setTime(t => t - 1), 1000);

      return () => clearInterval(timer);

    } else if (time === 0) handleNextQuestion();

  }, [time, showScore]);

  useEffect(() => {

    if (showScore) {

      const highScore = localStorage.getItem("highScore") || 0;

      if (score > highScore) localStorage.setItem("highScore", score);

    }

  }, [showScore]);

  const handleOptionSelect = (option) => dispatch({ type: "SELECT\_OPTION", payload: option });

  const handleNextQuestion = () => {

    const isCorrect = selectedOption === questions[currentQuestion].answer;

    setFeedback(isCorrect ? <><FaCheckCircle className="text-success me-2" /> Correct! 🎉</> : <><FaTimesCircle className="text-danger me-2" /> Incorrect! The correct answer is {questions[currentQuestion].answer}</>);

    setTimeout(() => setFeedback(""), 2000);

    dispatch({ type: "NEXT\_QUESTION" });

    setTime(10);

  };

  const handleRestartQuiz = () => {

    dispatch({ type: "RESTART\_QUIZ" });

    setTime(10);

    setFeedback("");

  };

  return (

    <Container className="my-5">

      <Row className="justify-content-center">

        <Col md={8}>

          <Card className="p-4 shadow-sm border-0 rounded-3 bg-light">

            <h2 className="text-center mb-4 text-primary">Quiz Challenge</h2>

            {showScore ? (

              <div className="text-center">

                <h3 className="text-success mb-3">Your Score: {score} / {questions.length}</h3>

                <h4 className="text-muted mb-3">High Score: {localStorage.getItem("highScore") || 0}</h4>

                <Button variant="primary" size="lg" onClick={handleRestartQuiz} className="px-4">

                  Restart Quiz

                </Button>

              </div>

            ) : (

              <div>

                <div className="d-flex justify-content-between mb-3">

                  <h5 className="text-muted">Question {currentQuestion + 1} of {questions.length}</h5>

                  <h5 style={{ color: time < 5 ? "red" : "black" }}>Time: {time}s</h5>

                </div>

                {feedback && <div className="alert alert-info d-flex align-items-center mb-3">{feedback}</div>}

                <h4 className="mb-4">{questions[currentQuestion].question}</h4>

                <div className="d-flex flex-wrap gap-2 mb-4">

                  {questions[currentQuestion].options.map((option, index) => (

                    <Button

                      key={index}

                      variant={selectedOption === option ? "success" : "outline-primary"}

                      className="flex-fill"

                      onClick={() => handleOptionSelect(option)}

                    >

                      {option}

                    </Button>

                  ))}

                </div>

                <Button

                  variant="primary"

                  size="lg"

                  className="w-100"

                  disabled={!selectedOption}

                  onClick={handleNextQuestion}

                >

                  {currentQuestion === questions.length - 1 ? "Finish Quiz" : "Next Question"}

                </Button>

              </div>

            )}

          </Card>

        </Col>

      </Row>

    </Container>

  );

}

export default QuestionBank;