**Lab 6: About Redux, Redux Thunk and Redux Toolkit**

**Tạo componets/CheckAnswers.js**

import React from "react";

import PropTypes from "prop-types";

import {

  Button,

  Card,

  CardBody,

  CardTitle,

  Container,

  Row,

  Col,

} from "reactstrap";

const CheckAnswers = ({ answers, onRetakeQuiz }) => {

  return (

    <Container className="my-4">

      <h2 className="text-center bg-dark text-white py-3">Quiz Review</h2>

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

        {answers && answers.length > 0 ? (

          answers.map((answer, index) => (

            <Col key={index} sm="6" md="4" lg="2" className="mb-3">

              <Card

                className="h-100 border-0"

                style={{

                  backgroundColor: answer.isAnswered ? "#d1e7dd" : "#f8d7da",

                  color: answer.isAnswered ? "#155724" : "#721c24",

                }}

              >

                <CardBody className="text-center">

                  <CardTitle tag="h6">Question No {index + 1}</CardTitle>

                  <p

                    className={

                      answer.isAnswered ? "text-success" : "text-danger"

                    }

                  >

                    {answer.isAnswered ? "Answered" : "Unanswered"}

                  </p>

                </CardBody>

              </Card>

            </Col>

          ))

        ) : (

          <p className="text-center mt-4">No answers available to review.</p>

        )}

      </Row>

      <div className="text-center mt-3">

        <Button color="primary" onClick={onRetakeQuiz}>

          Back to Quiz

        </Button>

      </div>

    </Container>

  );

};

CheckAnswers.propTypes = {

  answers: PropTypes.arrayOf(

    PropTypes.shape({

      isAnswered: PropTypes.bool.isRequired,

    })

  ).isRequired,

  onRetakeQuiz: PropTypes.func.isRequired,

};

export default CheckAnswers;

**Tạo componets/CustomNavbar.js**

import React from "react";

import { Link } from "react-router-dom";

import { Navbar, NavbarBrand, Nav, NavItem, NavLink } from "reactstrap";

const CustomNavbar = () => {

  return (

    <Navbar color="dark" dark expand="md">

      <NavbarBrand tag={Link} to="/">

        Home

      </NavbarBrand>

      <Nav className="me-auto" navbar>

        <NavItem>

          <NavLink tag={Link} to="/">

            About

          </NavLink>

        </NavItem>

        <NavItem>

          <NavLink tag={Link} to="/">

            News

          </NavLink>

        </NavItem>

        <NavItem>

          <NavLink tag={Link} to="/">

            Quiz

          </NavLink>

        </NavItem>

        <NavItem>

          <NavLink tag={Link} to="/">

            Contact

          </NavLink>

        </NavItem>

      </Nav>

    </Navbar>

  );

};

export default CustomNavbar;

**Tạo componets/Quiz.js**

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

import { useSelector, useDispatch } from "react-redux";

import {

  setQuestions,

  setCorrectAnswers,

  selectAnswer,

} from "../slices/quizSlice";

import CheckAnswers from "./CheckAnswers";

import Review from "./Review";

const Quiz = () => {

  const questions = useSelector((state) => state.quiz.questions);

  const selectedAnswers = useSelector((state) => state.quiz.userAnswers);

  const dispatch = useDispatch();

  const [currentQuestionIndex, setCurrentQuestionIndex] = useState(0);

  const [showSummary, setShowSummary] = useState(false);

  const [showReview, setShowReview] = useState(false); // New state for review

  useEffect(() => {

    // Initialize questions and correct answers once when the component mounts

    dispatch(

      setQuestions([

        {

          id: 1,

          text: "Inside which HTML element do we put the JavaScript?",

          options: ["javascript", "scripting", "script", "js"],

        },

        {

          id: 2,

          text: "What is the correct syntax to refer to an external script called 'script.js'?",

          options: [

            "<script src='script.js'>",

            "<script href='script.js'>",

            "<script ref='script.js'>",

            "<script name='script.js'>",

          ],

        },

        {

          id: 3,

          text: "Which of the following is used to create a function in JavaScript?",

          options: [

            "function myFunction()",

            "def myFunction()",

            "create myFunction()",

            "fn myFunction()",

          ],

        },

        {

          id: 4,

          text: "How can you add a comment in JavaScript?",

          options: [

            "<!-- This is a comment -->",

            "// This is a comment",

            "' This is a comment",

            "# This is a comment",

          ],

        },

        {

          id: 5,

          text: "How do you round the number 7.25 to the nearest integer?",

          options: [

            "Math.rnd(7.25)",

            "Math.round(7.25)",

            "round(7.25)",

            "rnd(7.25)",

          ],

        },

        {

          id: 6,

          text: "Which event occurs when a user clicks on an HTML element?",

          options: ["onmouseover", "onchange", "onmouseclick", "onclick"],

        },

        {

          id: 7,

          text: "How do you declare a JavaScript variable?",

          options: [

            "var myVar;",

            "variable myVar;",

            "v myVar;",

            "declare myVar;",

          ],

        },

        {

          id: 8,

          text: "Which operator is used to assign a value to a variable?",

          options: ["\*", "-", "=", "x"],

        },

        {

          id: 9,

          text: "What will the following code return: Boolean(10 > 9)?",

          options: ["true", "false", "NaN", "undefined"],

        },

        {

          id: 10,

          text: "How do you write 'Hello World' in an alert box?",

          options: [

            "msgBox('Hello World');",

            "alertBox('Hello World');",

            "msg('Hello World');",

            "alert('Hello World');",

          ],

        },

        {

          id: 11,

          text: "How does a WHILE loop start?",

          options: [

            "while (i <= 10; i++)",

            "while (i <= 10)",

            "while i = 1 to 10",

            "while (i++ <= 10)",

          ],

        },

        {

          id: 12,

          text: "How does a FOR loop start?",

          options: [

            "for (i = 0; i <= 5)",

            "for i = 1 to 5",

            "for (i <= 5; i++)",

            "for (i = 0; i <= 5; i++)",

          ],

        },

      ])

    );

    dispatch(

      setCorrectAnswers({

        1: "script",

        2: "<script src='script.js'>",

        3: "function myFunction()",

        4: "// This is a comment",

        5: "Math.round(7.25)",

        6: "onclick",

        7: "var myVar;",

        8: "=",

        9: "true",

        10: "alert('Hello World');",

        11: "while (i <= 10)",

        12: "for (i = 0; i <= 5; i++)",

      })

    );

  }, [dispatch]);

  const handleAnswerSelect = (answer) => {

    const questionId = questions[currentQuestionIndex]?.id;

    if (questionId) {

      dispatch(selectAnswer({ questionId, answer }));

    }

  };

  const navigateQuestion = (direction) => {

    if (direction === "next" && currentQuestionIndex < questions.length - 1) {

      setCurrentQuestionIndex(currentQuestionIndex + 1);

    } else if (direction === "prev" && currentQuestionIndex > 0) {

      setCurrentQuestionIndex(currentQuestionIndex - 1);

    }

  };

  const submitQuiz = () => {

    setShowReview(true); // Show the review component after submission

  };

  const resetQuiz = () => {

    setShowSummary(false);

    setShowReview(false);

    setCurrentQuestionIndex(0);

    // You might also want to reset selected answers in the redux store if needed

  };

  // Generate answers array for CheckAnswers component

  const answers = questions.map((question) => ({

    isAnswered: selectedAnswers[question.id] !== undefined,

  }));

  // Conditionally render the Review, CheckAnswers component, or the main quiz view

  if (showReview) {

    return <Review onRetakeQuiz={resetQuiz} />;

  }

  if (showSummary) {

    return <CheckAnswers answers={answers} onRetakeQuiz={resetQuiz} />;

  }

  // Main quiz view

  return (

    <div className="quiz-container">

      <header className="quiz-header">

        <h1>JavaScript Quiz</h1>

      </header>

      {questions[currentQuestionIndex] && (

        <div className="question-container">

          <h3 className="question-text">{`Q${currentQuestionIndex + 1}. ${

            questions[currentQuestionIndex].text

          }`}</h3>

          <div className="options">

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

              <button

                key={index}

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

                className={`option ${

                  selectedAnswers[questions[currentQuestionIndex].id] === option

                    ? "selected"

                    : ""

                }`}

              >

                {option}

              </button>

            ))}

          </div>

        </div>

      )}

      <div className="navigation-buttons">

        <button onClick={() => setCurrentQuestionIndex(0)}>First</button>

        <button

          onClick={() => navigateQuestion("prev")}

          disabled={currentQuestionIndex === 0}

        >

          Prev

        </button>

        <button

          onClick={() => navigateQuestion("next")}

          disabled={currentQuestionIndex === questions.length - 1}

        >

          Next

        </button>

        <button onClick={() => setCurrentQuestionIndex(questions.length - 1)}>

          Last

        </button>

      </div>

      <div className="action-buttons">

        <button onClick={submitQuiz} className="submit-button">

          Submit

        </button>

        <button

          onClick={() => setShowSummary(true)}

          className="check-answer-status-button submit-button"

        >

          Check Answers

        </button>

      </div>

    </div>

  );

};

export default Quiz;

**Tạo componets/Review.js**

import React from "react";

import { useSelector } from "react-redux";

import { Button } from "reactstrap";

import "./QuizReview.css";

const QuizReview = ({ onRetakeQuiz }) => {

  const questions = useSelector((state) => state.quiz.questions);

  const selectedAnswers = useSelector((state) => state.quiz.userAnswers);

  const correctAnswers = useSelector((state) => state.quiz.correctAnswers);

  return (

    <div className="review-container">

      <header className="review-header">

        <h1>Quiz Review</h1>

      </header>

      {questions.map((question, index) => {

        const userAnswer = selectedAnswers[question.id];

        const isCorrect = userAnswer === correctAnswers[question.id];

        const isAnswered = userAnswer !== undefined;

        return (

          <div

            key={question.id}

            className={`review-question ${

              isAnswered && isCorrect ? "correct" : "incorrect"

            }`}

          >

            <p className="question-text">

              <strong>{`Q${index + 1}. ${question.text}`}</strong>

            </p>

            <div className="options">

              {question.options.map((option, idx) => (

                <div

                  key={idx}

                  className={`option ${

                    option === userAnswer

                      ? option === correctAnswers[question.id]

                        ? "correct"

                        : "incorrect"

                      : option === correctAnswers[question.id]

                      ? "correct-answer"

                      : ""

                  }`}

                >

                  <input

                    type="radio"

                    checked={option === userAnswer}

                    readOnly

                  />

                  <label>{option}</label>

                </div>

              ))}

            </div>

            <p className="correct-answer-text">

              Right answer is: <strong>{correctAnswers[question.id]}</strong>

            </p>

          </div>

        );

      })}

      <div className="text-center mt-3">

        <Button color="primary" onClick={onRetakeQuiz}>

          Back to Quiz

        </Button>

      </div>

    </div>

  );

};

export default QuizReview;

**Tạo slices/quizSlide.js**

// quizSlice.js

import { createSlice } from "@reduxjs/toolkit";

const initialState = {

  questions: [],

  correctAnswers: {},

  userAnswers: {},

  showReview: false,

};

const quizSlice = createSlice({

  name: "quiz",

  initialState,

  reducers: {

    setQuestions: (state, action) => {

      state.questions = action.payload;

    },

    setCorrectAnswers: (state, action) => {

      state.correctAnswers = action.payload;

    },

    selectAnswer: (state, action) => {

      const { questionId, answer } = action.payload;

      state.userAnswers[questionId] = answer;

    },

    toggleReview: (state) => {

      state.showReview = !state.showReview;

    },

  },

});

export const { setQuestions, setCorrectAnswers, selectAnswer, toggleReview } =

  quizSlice.actions;

export default quizSlice.reducer;

**Tạo store/store.js**

// store.js

import { configureStore } from "@reduxjs/toolkit";

import quizReducer from "../slices/quizSlice";

const store = configureStore({

  reducer: {

    quiz: quizReducer,

  },

});

export default store;

**App.js**

import React from "react";

import { Provider } from "react-redux";

import store from "./store/store";

import Quiz from "./components/Quiz";

import "./App.css";

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";

import CustomNavbar from "./components/CustomNavbar";

import CheckAnswers from "./components/CheckAnswers";

import Review from "./components/Review";

const App = () => {

  return (

    <Provider store={store}>

      <Router>

        <CustomNavbar />

        <Routes>

          <Route path="/" element={<Quiz />} />

          <Route path="/review" element={<CheckAnswers />} />

          <Route path="/SubmitReview" element={<Review />} />

        </Routes>

      </Router>

    </Provider>

  );

};

export default App;

**index.js**

import React from "react";

import ReactDOM from "react-dom/client";

import "./index.css";

import App from "./App";

import reportWebVitals from "./reportWebVitals";

import "bootstrap/dist/css/bootstrap.min.css";

const root = ReactDOM.createRoot(document.getElementById("root"));

root.render(

  <React.StrictMode>

    <App />

  </React.StrictMode>

);

reportWebVitals();

**App.css**

/\* Container for the entire quiz component \*/

.quiz-container {

  max-width: 800px;

  margin: 0 auto;

  padding: 20px;

  background-color: #f4f4f9;

  border-radius: 8px;

  text-align: center;

  box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);

}

/\* Header styling \*/

.quiz-header {

  background-color: #333;

  color: white;

  padding: 15px 0;

  border-radius: 8px 8px 0 0;

  margin-bottom: 20px;

}

h1 {

  margin: 0;

  font-size: 1.8rem;

}

/\* Question container styling \*/

.question-container {

  margin-bottom: 20px;

}

/\* Styling for the question text \*/

.question-text {

  font-size: 1.2rem;

  margin-bottom: 10px;

  text-align: left;

  font-weight: 600;

  color: #333;

}

/\* Options styling \*/

.options {

  display: flex;

  flex-wrap: wrap;

  gap: 15px;

  justify-content: center;

}

.option {

  background-color: #dfe7fd;

  border: 2px solid transparent;

  padding: 15px 20px;

  border-radius: 8px;

  cursor: pointer;

  flex: 1 1 45%;

  text-align: center;

  font-size: 1rem;

  transition: background-color 0.3s, border-color 0.3s;

}

.option.selected {

  background-color: #a6c8ff;

  border-color: #007bff;

  color: #333;

}

/\* Hover effect for options \*/

.option:hover {

  background-color: #c3d6ff;

}

/\* Navigation buttons styling \*/

.navigation-buttons {

  display: flex;

  justify-content: center;

  gap: 10px;

  margin-top: 20px;

}

.navigation-buttons button {

  padding: 10px 20px;

  background-color: #007bff;

  color: white;

  border: none;

  border-radius: 4px;

  cursor: pointer;

  font-size: 1rem;

  font-weight: 500;

  transition: background-color 0.3s;

}

.navigation-buttons button:hover {

  background-color: #0056b3;

}

.navigation-buttons button:disabled {

  background-color: #cccccc;

  cursor: not-allowed;

}

/\* Action buttons styling for Quiz Review and Submit \*/

.action-buttons {

  display: flex;

  justify-content: space-between;

  margin-top: 20px;

  gap: 10px;

}

.review-button,

.submit-button {

  padding: 12px 20px;

  font-size: 1rem;

  font-weight: 600;

  border: none;

  border-radius: 8px;

  cursor: pointer;

  width: 48%;

  transition: background-color 0.3s;

}

.review-button {

  background-color: #17a2b8;

  color: white;

}

.review-button:hover {

  background-color: #138496;

}

.submit-button {

  background-color: #28a745;

  color: white;

}

.submit-button:hover {

  background-color: #218838;

}

/\* Container for the review component \*/

.review-container {

  max-width: 800px;

  margin: 0 auto;

  padding: 20px;

  background-color: #f4f4f9;

  border-radius: 8px;

  box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);

}

/\* Header styling \*/

.review-header {

  background-color: #333;

  color: white;

  padding: 15px 0;

  border-radius: 8px 8px 0 0;

  text-align: center;

  margin-bottom: 20px;

}

h1 {

  margin: 0;

  font-size: 1.8rem;

}

/\* Styling for each review question \*/

.review-question {

  padding: 20px;

  border-radius: 8px;

  margin-bottom: 20px;

}

/\* Correct and incorrect answer background colors \*/

.review-question.correct {

  background-color: #d4edda; /\* Light green \*/

  border: 1px solid #c3e6cb;

}

.review-question.incorrect {

  background-color: #f8d7da; /\* Light red \*/

  border: 1px solid #f5c6cb;

}

/\* Styling for the question text \*/

.question-text {

  font-size: 1.1rem;

  font-weight: bold;

  margin-bottom: 10px;

}

/\* Options styling \*/

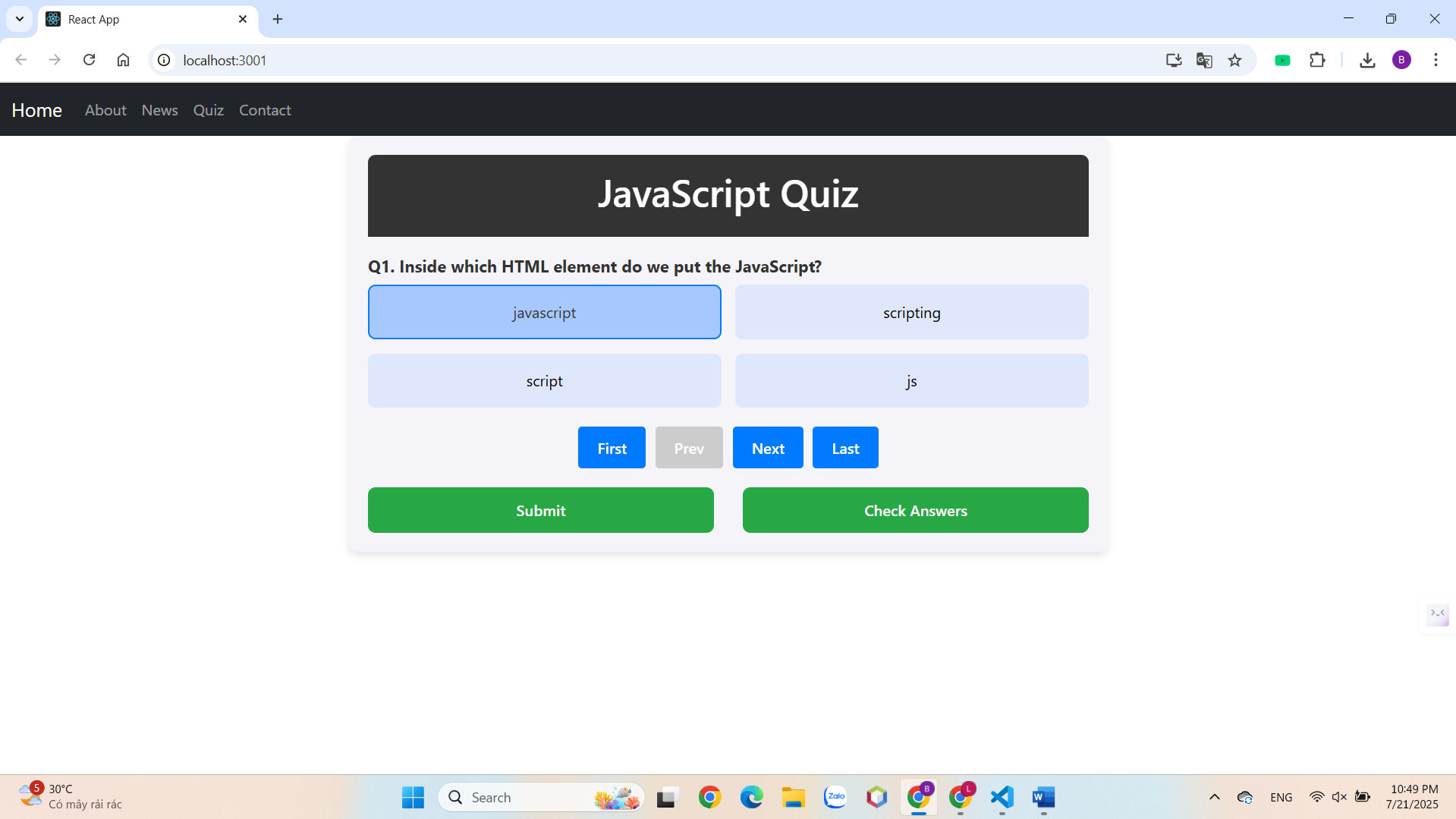
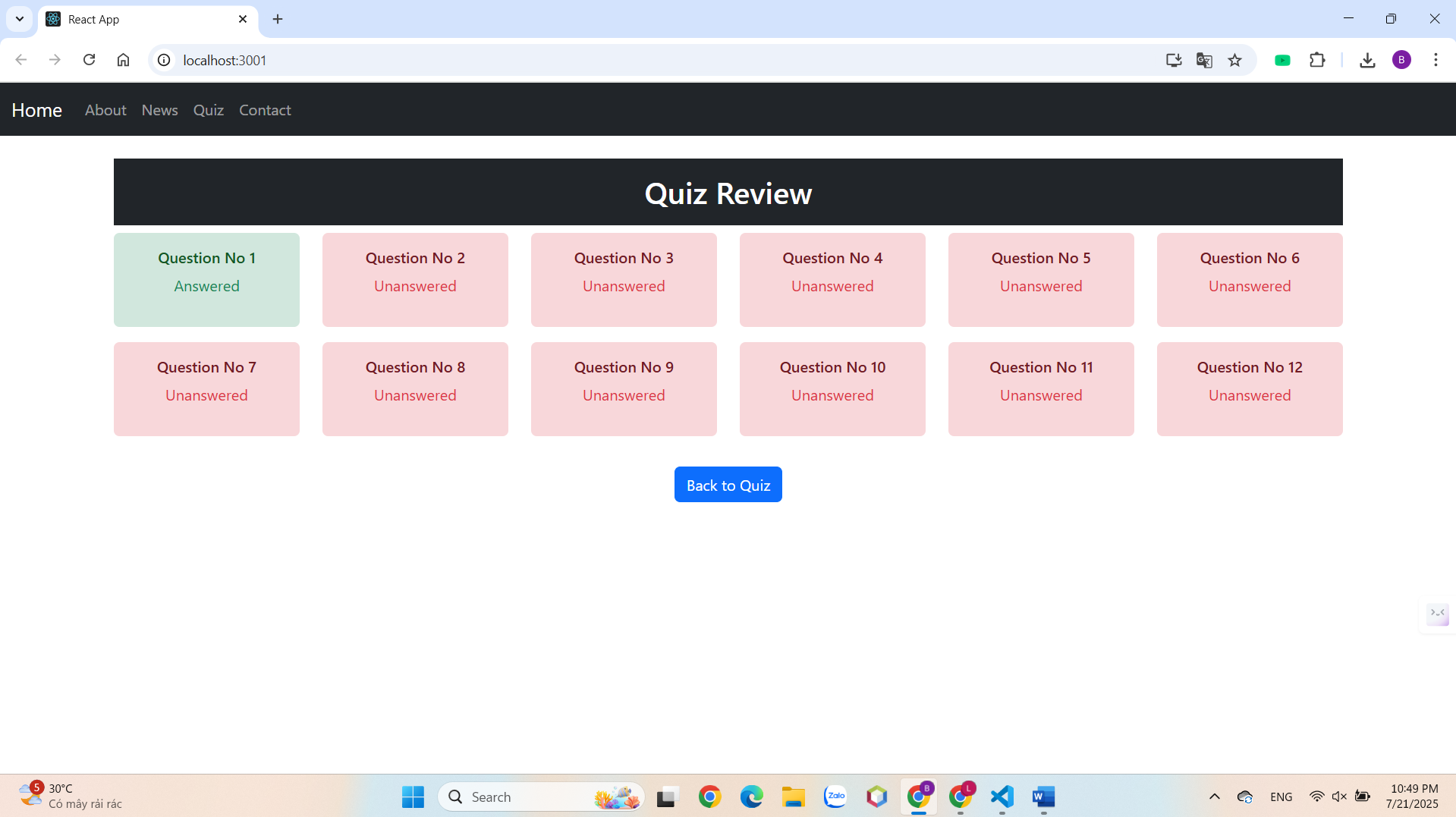
.options {

  margin: 10px 0;

  display: flex;

}

**KẾT QUẢ**

**** **** 