Started on	Wednesday, 4 December 2024, 9:41 PM
State	Finished
Completed on	Wednesday, 4 December 2024, 9:51 PM
Time taken	9 mins 23 secs
Marks	1.00/10.00
Grade	**10.00** out of 100.00

Question **1**

Complete

Mark 0.00 out of 1.00

Which of the following is the correct way to update the scores of a student using a PUT request in the backend?

Select one:

a.
```
app.put("/student/:rollNo", async (req, res) => {
  const { rollNo } = req.body;
  const updatedStudent = await Student.findOneAndUpdate({ rollNo },
req.body);
  res.status(200).json({ message: "Update successful" });
});
```

b.
```
app.put("/student/:rollNo", (req, res) => {
  const rollNo = req.params.rollNo;
  Student.findOneAndUpdate({ rollNo }, req.body);
  res.send("Updated");
});
```

c.
```
app.put("/student/:rollNo", async (req, res) => {
  const rollNo = req.params.rollNo;
  const updatedStudent = await Student.updateOne({ rollNo }, req.body);
  res.json(updatedStudent);
});
```

d.
```
app.put("/student/:rollNo", async (req, res) => {
  const rollNo = req.params.rollNo;
  const updatedStudent = await Student.findOneAndUpdate(
    { rollNo },
    req.body,
    { new: true }
  );
  res.status(200).json({ message: "Update successful", updatedStudent
});
});
```

Copyright © 2024 Teleparadigm Networks Pvt. Ltd. All Rights Reserved.

Question 2

Complete

Mark 1.00 out of 1.00

What is the issue in the following backend delete route?

```
app.delete("/student/:rollNo", async (req, res) => {
  try {
    const deletedStudent = await Student.deleteMany({ rollNo: req.params.rollNo
});
    if (deletedStudent.deletedCount > 0) {
      res.status(200).json({ message: "Student deleted successfully" });
    } else {
      res.status(200).json({ message: "No student found" });
    }
  } catch (err) {
    res.status(500).json({ message: "Error deleting student", error: err });
  }
});
```

Select one:

- **a. The `deletedCount` check is unnecessary.**
- **b. The route deletes all students with the same roll number.**
- **c. The route should return a 404 status code if no student is found.**
- **d. The route throws an error if no student is found.**

Question 3

Complete

Mark 0.00 out of 1.00

In the following frontend code snippet, what will happen when the Update button is clicked?

```
<button
  className="btn btn-primary"
  onClick={() => handleUpdateClick(student.rollNo, student.scores)}
>
  Update
</button>
```

Select one:

- **a. The student's scores are updated in the database directly.**
- **b. The row becomes editable for updating scores.**
- **c. Nothing happens because the `handleUpdateClick` function is undefined.**
- **d. The component fetches updated data from the backend.**

Copyright © 2024 Teleparadigm Networks Pvt. Ltd. All Rights Reserved.

Question 4

Complete

Mark 0.00 out of 1.00

What is wrong with the following code for making a GET request in React?

```
useEffect(() => {
  axios.get("https://example.com/allstudents").then((response) => {
    setStudents(response.data);
  });
}, []);
```

Select one:

- a. **The API call should be synchronous.**
- b. **There is no error; the code works correctly.**
- c. **The `useEffect` hook does not support Promises.**
- d. **The API call should be wrapped in an async function.**

Question 5

Complete

Mark 0.00 out of 1.00

In the context of the StudentsTable component, what is the purpose of assigning dynamic IDs like `row<rollNo>` to table rows?

Select one:

- a. **To improve the performance of the component.**
- b. **To avoid duplicate table entries.**
- c. **To allow easy identification of rows for CRUD operations.**
- d. **To make the table rows visually distinct.**

Copyright © 2024 Teleparadigm Networks Pvt. Ltd. All Rights Reserved.

Question 6

Complete

Mark 0.00 out of 1.00

What is the correct way to fetch student data when the StudentsTable React component loads?

Select one:

a.
```
useEffect(() => {
  async function fetchData() {
    const response = await axios.get("https://example.com/allstudents");
    setStudents(response.data);
  }
  fetchData();
}, []);
```

b.
```
useEffect(() => {
  axios.get("https://example.com/allstudents").then((response) => {
    setStudents(response.data);
  });
}, []);
```

c.
```
useEffect(async () => {
  const response = await axios.get("https://example.com/allstudents");
  setStudents(response.data);
}, []);
```

d.
```
useEffect(() => {
  const response = axios.get("https://example.com/allstudents");
  setStudents(response.data);
}, []);
```

Question 7

Complete

Mark 0.00 out of 1.00

In the backend code snippet below, what is missing to delete a student from the database based on `rollNo`?

```
app.delete("/student/:rollNo", async (req, res) => {
  const rollNo = req.params.rollNo;
  try {
    const deletedStudent = await Student.findOneAndDelete({ rollNo });
    res.status(200).json({ message: "Student deleted successfully" });
  } catch (err) {
    res.status(400).json({ message: "Failed to delete student", error: err });
  }
});
```

Select one:

a. **A success message in case no student is found.**

b. **Validation to check if `rollNo` exists before deleting.**

c. **`await` keyword in the try block.**

d. **A `find()` query before `findOneAndDelete()`.**

Copyright © 2024 Teleparadigm Networks Pvt. Ltd. All Rights Reserved.

Question 8

Complete

Mark 0.00 out of 1.00

What does the following React code snippet do?

```
const handleDeleteClick = async (rollNo) => {
  const confirmDelete = window.confirm(
    `Are you sure you want to delete the student with Roll Number: ${rollNo}?`
  );
  if (confirmDelete) {
    const response = await axios.delete(`https://example.com/student/${rollNo}`);
    if (response.status === 200) {
      fetchStudents();
    }
  }
};
```

Select one:

- **a. It throws an error because the API response is not checked.**
- **b. It deletes the student from the frontend only.**
- **c. It only shows a confirmation dialog without deleting the student.**
- **d. It deletes the student with the specified roll number from the backend and refreshes the table.**

Question 9

Complete

Mark 0.00 out of 1.00

What will happen if the `await` keyword is omitted in the following delete route?

```
app.delete("/student/:rollNo", async (req, res) => {
  const deletedStudent = Student.findOneAndDelete({ rollNo: req.params.rollNo });
  res.status(200).json({ message: "Student deleted successfully" });
});
```

Select one:

- **a. The route will delete all students in the database.**
- **b. The route will work fine but may send the response before deletion is complete.**
- **c. The route will throw a runtime error.**
- **d. The route will wait for the deletion to complete before sending the response.**

Copyright © 2024 Teleparadigm Networks Pvt. Ltd. All Rights Reserved.

Question **10**

Complete

Mark 0.00 out of 1.00

Which of the following will correctly fetch the updated student list after deleting a student?

Select one:

a.
```
const fetchStudents = () => {
  const data = axios.get("https://example.com/allstudents");
  setStudents(data);
};
```

b.
```
const fetchStudents = async () => {
  const response = await axios.get("https://example.com/allstudents");
};
```

c.
```
const fetchStudents = () => {
  axios.get("https://example.com/allstudents").then((response) => {
    setStudents(response.data);
  });
};
```

d.
```
const fetchStudents = async () => {
  const response = await axios.get("https://example.com/allstudents");
  setStudents(response.data);
};
```

Copyright © 2024 Teleparadigm Networks Pvt. Ltd. All Rights Reserved.