polito-WA1-AW1-2023 / **wa1-ah-weeks** Public

Code Issues Pull requests Actions Projects Security Insights

Beta Try the new code view

main ▾ wa1-ah-weeks / week07 / QA-ex9 / src / **Components.jsx** / <> Jump to ▾ Go to file ···

fulcorno complete implementation of handleSave … Latest commit a9dc48e yesterday **History**

1 contributor

130 lines (107 sloc) | 4.03 KB Raw Blame

```jsx
import { useState } from "react";
import { Badge, Button, Col, Row, Table } from "react-bootstrap";
import { AddOrEditAnswer } from "./AnswerForm";

function QuestionWithAnswers(props) {

    const [mode, setMode] = useState('view');

    const [editedAnswer, setEditedAnswer] = useState(false);

    function handleCancel() {
        setMode('view');
    }

    function handleAdd(date, text, author) {
        props.addAnswer(date, text, author);
        setMode('view');
    }

    function handleSave(id, date, text, author) {
        props.editAnswer(id, date, text, author);
        setMode('view');
    }

    function handleEdit(id) {
        setEditedAnswer(props.answers.filter((a) => (a.id === id))[0])
        setMode('edit');
    }
    const q = props.question;
    const answers = props.answers;

    if (q) {
        return (<>
            <QuestionDetails author={q.author} text={q.text} />
            <AnswerDetails answers={answers} deleteAnswer={props.deleteAnswer} upVoteAnswer={props.upVoteAnswer} handleEdit={handleEdit} />
            {mode === 'edit' && <AddOrEditAnswer mode={mode} handleCancel={handleCancel} handleSave={handleSave} initialValue={editedAnswer}
            {mode === 'add' && <AddOrEditAnswer mode={mode} handleCancel={handleCancel} handleAdd={handleAdd} />}
            {mode === 'view' && <Button variant='success' onClick={() => setMode('add')}>ADD</Button>}
        </>)

    } else {
        return <div>"QUESTION UNDEFINED"</div>
    }

}

function QuestionDetails(props) {
    return <div>
        <Row>
            <Col md={8}>
                <p className='lead'>{props.text}</p>
            </Col>
            <Col md={4} className='text-end'>
                Asked by <Badge pill bg='secondary'>{props.author}</Badge>
            </Col>
        </Row>

    </div>
}

function AnswerDetails(props) {

    const [sorted, setSorted] = useState('none');

    // LOCAL COMPUTATION
    let sortedAnswers = [...props.answers]
    let sortIcon = '-'
    if (sorted === 'up') {
        sortedAnswers.sort((a, b) => (a.score - b.score));
        sortIcon = '^'
    } else if (sorted === 'down') {
        sortedAnswers.sort((a, b) => -(a.score - b.score));
        sortIcon = 'v'
    }


    function sortByScore() {
        if (sorted === 'none')
            setSorted('up')
        else if (sorted === 'up')
            setSorted('down')
        else if (sorted === 'down')
            setSorted('none')
    }

    // DERIVED STATE - GENERALLY A BAD IDEA (1-impossible to TRACK changes; 2-duplication of information)
    // const [sortedAnswers, setSortedAnswers] = useState(props.answers);
    // function sortByScore() {
    //     setSortedAnswers((old) => {
    //         let temp = [...old]
    //         temp.sort((a, b) => -(a.score - b.score))
    //         return temp
    //     })
    // }


    return <>
        <h2>Answers:</h2>
        <Table hover>
            <thead >
                <tr>
                    <th scope="col">Date</th>
                    <th scope="col">Text</th>
                    <th scope="col">Author</th>
                    <th scope="col" onClick={sortByScore}>Score {sortIcon}</th>
                    <th scope="col">Actions</th>
                </tr>
            </thead>
            <tbody>
                {sortedAnswers.map(a => <AnswerRow key={a.id} answer={a} deleteAnswer={props.deleteAnswer} upVoteAnswer={props.upVoteAnswer}
            </tbody>
        </Table>
    </>
}

function AnswerRow(props) {
    return <tr>
        <td>{props.answer.date.format('DD/MM/YYYY')}</td>
        <td>{props.answer.text}</td>
        <td>{props.answer.author}</td>
        <td>{props.answer.score}</td>
        <td><Button variant='secondary' onClick={() => { props.upVoteAnswer(props.answer.id) }}>VOTE</Button>{' '}
            <Button variant='warning' onClick={() => { props.deleteAnswer(props.answer.id) }}>DELETE</Button>{' '}
            <Button variant='success' onClick={() => { props.handleEdit(props.answer.id) }}>EDIT</Button>
        </td>
    </tr>
}


```

```
export { QuestionWithAnswers };
```