My Algorithm:

First, we will add one row to the table which shows bank transaction information by default and on clicking it shows and hides PayPal transactions as shown in below steps.

1. Create table, with table heading row with columns

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |

2. Fill the first row with bank transaction information

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times NyTime | transfer | -2.1 | 2020-09-17 |

3. On clicking above row, show PayPal transactions

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| *The New York Times Co.* | *shopping* | *-2* | *2020-09-17* |
| *From British Pound* | *transfer* | *2* | *2020-09-17* |
| *Credit Card* | *income* | *1.92* | *2020-09-17* |
| *To Euro* | *transfer* | *-1.92* | *2020-09-17* |
| *Paypal \*ny Times Nytime* | *transfer* | *-2.1* | *2020-09-17* |

4. on clicking it again we hide PayPal transactions or show same as step2 or below figure.

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times NyTime | transfer | -2.1 | 2020-09-17 |

Hence, we will append PayPal transactions related rows or remove rows to the table based on click events on bank transaction rows.

I want to use React JS library for working solution. We will map our algorithm to React components and assume the table will contain only one row containing bank transaction information. On clicking it, we will update rows of the table with PayPal Transactions. We will treat our table as react component before that we read data from json files and create Row data as input for our table

**Step1: *Grouping Transactions together***

***var bankTransaction = require("./transaction\_revolut1.json");***

***var paypalTransactions = [***

***require("./transaction\_paypal1.json"),***

***require("./transaction\_paypal2.json"),***

***require("./transaction\_paypal3.json"),***

***require("./transaction\_paypal4.json")***

***];***

***function makeRow(bankTransaction, paypalTransactions) {***

***return {***

***bankTransaction: bankTransaction,***

***paypalTransactions: paypalTransactions***

***};***

***}***

***var rowInTable = makeRow(bankTransaction, paypalTransactions);***

So, ***rowInTable*** will be input for our component

**Step2:Creating TransactionTable class component**

class TransactionTable extends React.Component {

render() {

return (

<table>

<tr id="TransactionTable" >

<th>SOURCE</th>

<th>CATEGORY</th>

<th>AMOUNT</th>

<th>DATE</th>

</tr>

</table>

);

}

}

The below statements will call render() in above TransactionTable class

const rootElement = document.getElementById("root");

ReactDOM.render(<TransactionTable row={rowInTable} /> ,rootElement);

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |

**Step3: We will add bankTransactionRow data to render() method in above class**

render() {

***var bankTransactionRow = [***

***<tr>***

***<td>{this.props.row.bankTransaction.description}</td>***

***<td>{this.props.row.bankTransaction.category}</td>***

***<td>{this.props.row.bankTransaction.amount}</td>***

***<td>{this.props.row.bankTransaction.made\_on}</td>***

***</tr>***

***];***

return (

<table>

<tr id="TransactionTable" >

<th>SOURCE</th>

<th>CATEGORY</th>

<th>AMOUNT</th>

<th>DATE</th>

</tr>

**{bankTransactionRow}**

</table>

);

}

}

Our output after step3 looks like below

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times NyTime | transfer | -2.1 | 2020-09-17 |

**Step4: show Paypal transactions on click**

Our Table should know when to add extra rows and remove extra rows. So, it should know the state of the row. So, we define a new state element or flag which decides to add or append rowsto table on clicking the row and when we update state data, our TransactionTable’s render() , method will be revoked again. So we will add paypal transaction rows or empty rows based on state data,So we need to add code to update state on click.

class TransactionTable extends React.Component {

**constructor() {**

**super()**;

**this.state = { showExpanded: false };**

**this.clickRow = this.clickRow.bind(this);**

}

**clickRow() {**

**if (this.state.showExpanded) {**

**this.setState({ showExpanded: false });**

**} else {**

**this.setState({ showExpanded: true });**

**}**

**}**

render() {

var bankTransactionRow = [

<tr **onClick={this.clickRow**}>

<td>{this.props.row.bankTransaction.description}</td>

<td>{this.props.row.bankTransaction.category}</td>

<td>{this.props.row.bankTransaction.amount}</td>

<td>{this.props.row.bankTransaction.made\_on}</td>

</tr>

];

**var paypalTransactions = [];**

**if (this.state.showExpanded) {**

**for (var i = 0; i < this.props.row.paypalTransactions.length; i++) {**

**var paypalTransaction = (**

**<tr>**

**<td>{this.props.row.paypalTransactions[i].description}</td>**

**<td>{this.props.row.paypalTransactions[i].category}</td>**

**<td>{this.props.row.paypalTransactions[i].amount}</td>**

**<td>{this.props.row.paypalTransactions[i].made\_on}</td>**

**</tr>**

**);**

**paypalTransactions.push(paypalTransaction);**

**}**

**}**

return (

<table id="TransactionTable" >

<tr>

<th>SOURCE</th>

<th>CATEGORY</th>

<th>AMOUNT</th>

<th>DATE</th>

</tr>

{bankTransactionRow**.concat(paypalTransactions)**} //

</table>

);

}

}

By default we hide paypal transactions so we set state.showExpanded to false.

On clicking the row we call **clickRow()**  which updates state showExpanded to true and revokes render()

In render we will see the state and fill extra rows as

var paypalTransactions = [];

if (this.state.showExpanded) {

for (var i = 0; i < this.props.row.paypalTransactions.length; i++) {

var paypalTransaction = (

<tr>

<td>{this.props.row.paypalTransactions[i].description}</td>

<td>{this.props.row.paypalTransactions[i].category}</td>

<td>{this.props.row.paypalTransactions[i].amount}</td>

<td>{this.props.row.paypalTransactions[i].made\_on}</td>

</tr>

);

**paypalTransactions.push(paypalTransaction)**;

}

}

And then we will add extra rows as below

return (

<table id="TransactionTable" >

<tr>

<th>SOURCE</th>

<th>CATEGORY</th>

<th>AMOUNT</th>

<th>DATE</th>

</tr>

{bankTransactionRow**.concat(paypalTransactions)**}

</table>

);

}

}

On clicking the first row we get paypal transactions

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times NyTime | transfer | -2.1 | 2020-09-17 |

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| *The New York Times Co.* | *shopping* | *-2* | *2020-09-17* |
| *From British Pound* | *transfer* | *2* | *2020-09-17* |
| *Credit Card* | *income* | *1.92* | *2020-09-17* |
| *To Euro* | *transfer* | *-1.92* | *2020-09-17* |
| *Paypal \*ny Times Nytime* | *transfer* | *-2.1* | *2020-09-17* |

Now we set state.showExpanded to true.

On clicking the row we call **clickRow()**  which updates state.showExpanded to false which revoke render() with empty paypalTransactions.so we will see only bank transaction row

var paypalTransactions = [];

if (this.state.showExpanded) {

… }

|  |  |  |  |
| --- | --- | --- | --- |
| SOURCE | CATEGORY | AMOUNT | DATE |
| Paypal \*ny Times NyTime | transfer | -2.1 | 2020-09-17 |

**Step5: extending solution for multiple rows**

When we have multiple rows we need to track status for each row. We use rowIds Of each row. We will display only banktransactions by default.

//Grouping Transactions together

var bankTransaction = require("./transaction\_revolut1.json");

var paypalTransactions = [

require("./transaction\_paypal1.json"),

require("./transaction\_paypal2.json"),

require("./transaction\_paypal3.json"),

require("./transaction\_paypal4.json")

];

function makeRow(bankTransaction, paypalTransactions) {

return {

bankTransaction: bankTransaction,

paypalTransactions: paypalTransactions

};

}

I have duplicated another bank transaction, so now we will have two rows

**var rowInTable = [**

**makeRow(bankTransaction, paypalTransactions),**

**makeRow(bankTransaction, paypalTransactions)**

**];**

Expecting Same bank transaction rows

|  |  |  |  |
| --- | --- | --- | --- |
| **SOURCE** | **CATEGORY** | **AMOUNT** | **DATE** |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |

we keep state.expandedRowIds empty in the constructor. On clicking we will add rowId to status.expandedRowIds which will be used to determine in appending PayPal Transaction rows at the end of each bank Transaction row.

class TransactionTable extends React.Component {

constructor() {

super();

this.state = { expandedRowIds: [] };

}

we will update our clickRow function to take rowId as we need to maintain state for each row.

By default we will not have rowId in expandedRowIds array but we it on clicking for the first time and then remove on clicking again. So below code track expandedRowIds

clickRow(**id**) {

var expandedRowIds = this.state.expandedRowIds;

var isRowExpanded = expandedRowIds.includes(id);

if (!isRowExpanded) {

this.setState({

expandedRowIds: expandedRowIds.concat(id)

});

} else {

this.setState({

expandedRowIds: expandedRowIds.filter(

(expandedRowIds) => expandedRowIds !== id

)

});

}

}

I have created separate methods to fetch details related to each row based on rowIds

bankTransactionRow(rowId) {

const clickRow = () => this.clickRow(rowId);

var bankTransaction = (

<tr onClick={clickRow}>

<td>{this.props.rows[rowId].bankTransaction.description}</td>

<td>{this.props.rows[rowId].bankTransaction.category}</td>

<td>{this.props.rows[rowId].bankTransaction.amount}</td>

<td>{this.props.rows[rowId].bankTransaction.made\_on}</td>

</tr>

);

return bankTransaction;

}

paypalTransactions(rowId) {

var paypalTransactions = [];

for (var i = 0; i < this.props.rows[rowId].paypalTransactions.length; i++) {

var paypalTransaction = (

<tr>

<td>{this.props.rows[rowId].paypalTransactions[i].description}</td>

<td>{this.props.rows[rowId].paypalTransactions[i].category}</td>

<td>{this.props.rows[rowId].paypalTransactions[i].amount}</td>

<td>{this.props.rows[rowId].paypalTransactions[i].made\_on}</td>

</tr>

);

paypalTransactions.push(paypalTransaction);

}

return paypalTransactions;

}

render() {

var allrows = [];

for (var i = 0; i < this.props.rows.length; i++) {

allrows.push(this.bankTransactionRow(i)); //  **we will add banktransaction rows by default, I also //introduced owid here**

if (**this.state.expandedRowIds.includes(i)**) {

const paypalTransactionsForRow = this.paypalTransactions(i);

for (var j = 0; j < paypalTransactionsForRow.length; j++) {

allrows.push(paypalTransactionsForRow[j]); //  **we will add paypal transaction rows to //banktransaction row if it clicked.**

}

}

}

return (

<table id="TransactionTable">

<tr>

<th>SOURCE</th>

<th>CATEGORY</th>

<th>AMOUNT</th>

<th>DATE</th>

</tr>

{allrows};

</table>

);

}

}

const rootElement = document.getElementById("root");

ReactDOM.render(<TransactionTable rows={rowInTable} />, rootElement);

Outputs:

On clicking first row in the below table state,

|  |  |  |  |
| --- | --- | --- | --- |
| **SOURCE** | **CATEGORY** | **AMOUNT** | **DATE** |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |

We get the following table state,

|  |  |  |  |
| --- | --- | --- | --- |
| **SOURCE** | **CATEGORY** | **AMOUNT** | **DATE** |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| The New York Times Co. | shopping | -2 | 2020-09-17 |
| From British Pound | transfer | 2 | 2020-09-17 |
| Credit Card | income | 1.92 | 2020-09-17 |
| To Euro | transfer | -1.92 | 2020-09-17 |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |

On clicking last row in above table state, we get

|  |  |  |  |
| --- | --- | --- | --- |
| **SOURCE** | **CATEGORY** | **AMOUNT** | **DATE** |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| The New York Times Co. | shopping | -2 | 2020-09-17 |
| From British Pound | transfer | 2 | 2020-09-17 |
| Credit Card | income | 1.92 | 2020-09-17 |
| To Euro | transfer | -1.92 | 2020-09-17 |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| The New York Times Co. | shopping | -2 | 2020-09-17 |
| From British Pound | transfer | 2 | 2020-09-17 |
| Credit Card | income | 1.92 | 2020-09-17 |
| To Euro |  |  |  |

On clicking first row in the above table state,

|  |  |  |  |
| --- | --- | --- | --- |
| **SOURCE** | **CATEGORY** | **AMOUNT** | **DATE** |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| The New York Times Co. | shopping | -2 | 2020-09-17 |
| From British Pound | transfer | 2 | 2020-09-17 |
| Credit Card | income | 1.92 | 2020-09-17 |
| To Euro | transfer | -1.92 | 2020-09-17 |

On clicking second row in above table, we get

|  |  |  |  |
| --- | --- | --- | --- |
| **SOURCE** | **CATEGORY** | **AMOUNT** | **DATE** |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |
| Paypal \*ny Times Nytime | transfer | -2.1 | 2020-09-17 |

I have attached files single\_row.js and multiple\_row.js files with this document for complete code with minimal css for my table in style.css