REST API 2

Onze API is klaar maar we moeten nog iets tweaken aan het outputformaat.

Stap 1, Output formaat veranderen

Dit is wat we **nu hebben**:

```
[
  {
    "vraag": "Wat is de hoofdstad van Frankrijk?",
    "antwoord1": "New York",
    "antwoord2": "London",
    "antwoord3": "Paris",
    "antwoord4": "Dublin",
    "juiste_antwoord": "3"
  },
```

En dit is wat we **nodig** hebben in onze React app:

```
[
  {
    "questionText": "Wat is de hoofdstad van Frankrijk?",
    "answerOptions": [
      {
        "answerText": "New York",
        "isCorrect": "false"
      },
      {
        "answerText": "London",
        "isCorrect": "false"
      },
      {
        "answerText": "Paris",
        "isCorrect": "true"
      },
      {
        "answerText": "Dublin",
```

```
        "isCorrect": "false"
      }
    ]
  },
```

We moeten onze API output dus omzetten (transformeren) van het ene formaat naar het andere formaat.

We lopen in een loop vraag-voor-vraag door de vragen heen en zetten elke vraag in het juiste formaat.

Met de volgende aangepaste functie zetten we het JSON-formaat om.

```
public function actionApiGet() {

    $sql = "select vraag, antwoord1, antwoord2, antwoord3, antwoord4, juiste_antwoord from
quiz";
    $result = Yii::$app->db->createCommand($sql)->queryAll();

    $output = [];
    foreach ($result as $elem) {
        array_push( $output,
            [   'questionText' => $elem['vraag'],
                'answerOptions' =>
                [
                    ['answerText'=>$elem['antwoord1'],'isCorrect'=>( $elem['juiste_antwoord']
== 1 ? 1 : 0) ],
                    ['answerText'=>$elem['antwoord2'],'isCorrect'=>( $elem['juiste_antwoord']
== 2 ? 1 : 0) ],
                    ['answerText'=>$elem['antwoord3'],'isCorrect'=>( $elem['juiste_antwoord']
== 3 ? 1 : 0) ],
                    ['answerText'=>$elem['antwoord4'],'isCorrect'=>( $elem['juiste_antwoord']
== 4 ? 1 : 0) ],
                ]
            ] );
    }

    return json_encode($output);
}
```

Dit is een stukje lastige code. Allereerst gebruiken we de PHP-data structuren voor een array. Het array wordt dan later (in regel 20) omgezet in JSON.

Zet deze regels (tijdelijk) vlak voor de return zodat je beter kan zien wat er gebeurt.

```
echo "<pre>";
var_dump($output);
```

Dit is het array wat we in de loop (regel 7, *foreach*.....) opbouwen. Dit array wordt in regel 20 (*json_encode*) vertaald in een JSON structuur.

Stap 2, aanpassen REACT code

We passen de REACT code in twee kleine stapjes aan. We zetten eerst onze bestaande JSON in een file en laten onze code een file lezen. Als dat werkt dan hoeven we alleen nog maar de verwijzing van de file te veranderen in een verwijzing naar de REST-API (de URL dus).

We doen dit weer stap-voor-stap en zetten eerst onze bestaande JSON in een file zodat we die moeten inlezen. Het inlezen van een file is namelijk bijna hetzelfde als het inlezen van data via een REST API.

In REACT zetten we in onze public folder een nieuwe file met de naam '*data.json*'.

data.json inhoud:

```
[
   {
       "questionText": "Wat is de hoofdstad van Frankrijk?",
       "answerOptions": [
           { "answerText": "New York", "isCorrect": "false" },
           { "answerText": "London", "isCorrect": "false" },
           { "answerText": "Paris", "isCorrect": "true" },
           { "answerText": "Dublin", "isCorrect": "false" }
       ]
   },
   {
       "questionText": "Wie is de oprichter van Tesla?",
       "answerOptions": [
           { "answerText": "Jeff Bezos", "isCorrect": "false" },
           { "answerText": "Elon Musk", "isCorrect": "true" },
           { "answerText": "Bill Gates", "isCorrect": "false" },
           { "answerText": "Tony Stark", "isCorrect": "false" }
       ]
```

```
    },
    {
        "questionText": "Welk bedrij was eind jaren 90 bijna failliet?",
        "answerOptions": [
            { "answerText": "Apple", "isCorrect": "true" },
            { "answerText": "Intel", "isCorrect": "false" },
            { "answerText": "Amazon", "isCorrect": "false" },
            { "answerText": "Microsoft", "isCorrect": "false" }
        ]
    }

]
```

Test door via de url:

`localhost:3000/data.json`

de json data op te vragen.

Nu moeten we de file inlezen. Dat doen we door eerst de volgende twee methods (functies) toe te voegen aan de class *App*.

```
componentDidMount() {
  this.getData();
}

getData() {
  fetch('./data.json')
  .then(res => res.json())
  .then((data) => {
    this.setState({
      jsonLoaded: true,
      jsonData: data
    });
  })
  .catch(console.log);
}
```

De method compnentDidMount is een standaard React functie die wordt uitgevoerd als een component (class) wordt geladen.

De getdata() method hebben we zelf gemaakt en die leest de file asynchroon in. En zet de data (de vragen dus) in een state variabele; de variabale *jsonData* bevat de vragen en de *jsonLoaded* wordt op *true* gezet om te laten zien dat alle data ingeladen is.

Asynchroon betekent dat de 'code' niet wacht tot de data is ingelezen. Het inlezen gebeurt als het ware in de achtergrond.

In de render method moeten we daarom eerst wachten tot de data is ingeladen. Op de eerste regel van de render methode zetten we daarom de volgende code:

```
if ( ! this.state.jsonLoaded ) {
  return (  <div className="app">loading...</div>)
}
```

In de praktijk gaat het laden erg snel maar als het laden te lang duurt wil je niet dat je App een foutmelding geeft. We wachten dus totdat de data is ingelezen. Als dat gedaan is dan doen we:

```
var questions=this.state.jsonData;
```

Hiermee is de variabele question gelijk aan de state variabele en werkt de rest van de code weer.

We kunnen ook overal waar in de code *questions* wordt gebruikt dat vervangen door *this.state.jsonData.*

Test je code en oops.... er gaat waarschijnlijk nog iets fout.

In *handleAnswerOtionClicked* hebben we een if-statement waarmee we bepalen of we moeten stoppen. Daarin gebruiken we de constante *questions* nog. Deze moeten we vervangen in *this.state.jsonData*

Het if-statement wordt dus:

```
if ( this.state.vraagNr < this.state.jsonData.length-1) {
```

Tenslotte kunnen we bovenaan in onze code de constante *questions* weghalen.

Test nog een keer goed of alles werkt voordat je verder gaat!

De laatste stap is nu heel eenvoudig als alles goed werkt.

Stap 3, aanpassen verwijzing naar file in verwijzing naar web (Yii) web site.

Als de vorige stap goed is uitgevoerd en alles werkt dan hoeven alleen nog maar de verwijzing in de fetch naar de './json' te vervangen in de url van onze Yii app.

```
getData() {
    fetch('http://localhost:8080/quiz/api-get')
    ...
```

Je zult nu waarschijnlijk ook zien dat de data even moet laden (afhankelijk van hoe snel alles is

Nu kan je in jouw Yii app vragen aanpassen en/of toevoegen.

Inleveren

App.js

--

Revision #34
Created 2021-08-10 08:37:10 UTC by Max
Updated 2022-08-28 08:50:14 UTC by Max