{ES6}

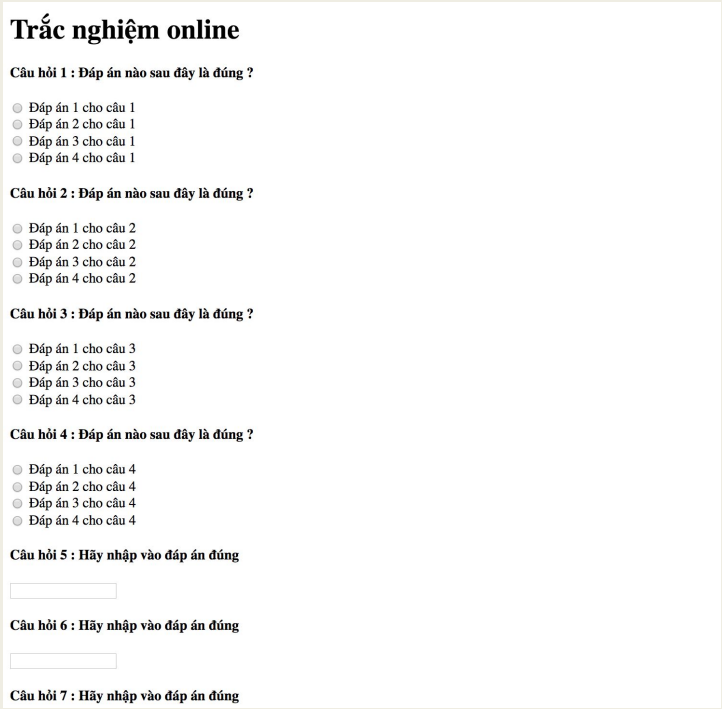
classes

```
class Polygon {
  constructor(height, width) {
    this.height = height;
    this.width = width;
  }
}
```

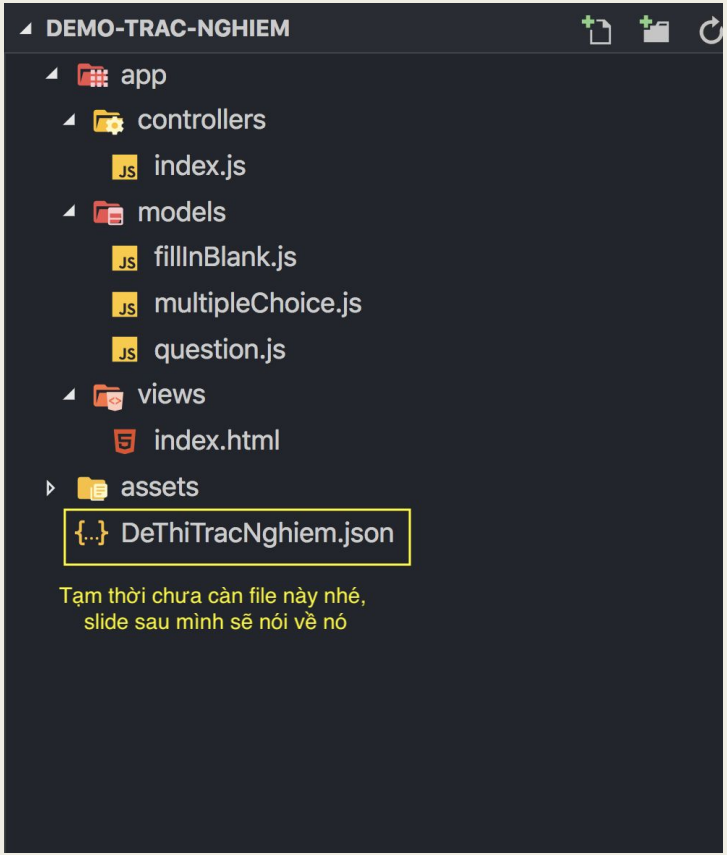
Online testing website

Yêu cầu

- ***Xây dựng trang web làm bài thi trắc nghiệm online với các tính năng sau:***
  - ***Hiển thị danh sách câu hỏi ra màn hình***
  - ***Chấm điểm***
  - ***Filter câu hỏi theo loại***
  - ***Mục lục câu hỏi***
- ***Ví dụ minh hoạ:***
- ***Trong slide này sẽ tập trung hướng dẫn các bạn làm bài tập, lý thuyết mọi người xem thêm ở slide lý thuyết nhé ^^!***



❑ *Bước 1 : Xây dựng Cấu trúc thư mục bài tập*



❑ *Bước 2 : Xây dựng lớp đối tượng*

- Đối với bài tập này, ta sẽ có 2 loại câu hỏi chính : multiple choice và fill in blank
- Đây là là 2 loại câu hỏi khác nhau, đều có những thuộc tính giống nhau , ví dụ như loại câu hỏi ,tên câu hỏi, đáp án….Đồng thời cũng có những điểm khác nhau, ví dụ như mỗi loại câu hỏi sẽ có cách chấm điểm, giao diện hiển thị khác nhau.
- Do đó bài tập này ta sẽ cần 3 lớp đối tượng:
  - Question: chứa các thuộc tính chung của câu hỏi: content, answer,type…
  - MultipleChoice và FillInBlank: 2 lớp này sẽ kế thừa lại lớp Question đồng thời mỗi loại sẽ có các thuộc tính và phương thức riêng của mỗi loại

❑ *Bước 2 : Xây dựng lớp đối tượng*

models ▸ question.js ▸ Question

```
class Question {
  //default value : Trong trường hợp không truyền tham số , thì sẽ lấy
  // giá trị mặc đinh
  constructor(questionType = "", _id = "", content = "", answers = []) {
    this.questionType = questionType;
    this._id = _id;
    this.content = content;
    this.answers = answers;
  }
}
```

models ▸ fillInBlank.js ▸ ...

```
// extends là cú pháp kế thừa
class FillInBlank extends Question {
  /*
  rest parameters: gom tất cả tham số
  người dùng truyền vào thành mảng args
  */
  constructor(...args) {
    //spread operator : bung mảng args thành các phần tử riêng lẻ
    //vd : từ [1,2,3] => 1,2,3
    //hàm super dùng để kế thừa lại constructor của clas cha
    super(...args);
  }
}
```

Lưu ý: thứ tự tham số truyền vào giống với class cha.Vd: hoten -> tuoi -> email...

models ▸ multipleChoice.js ▸ MultipleChoice

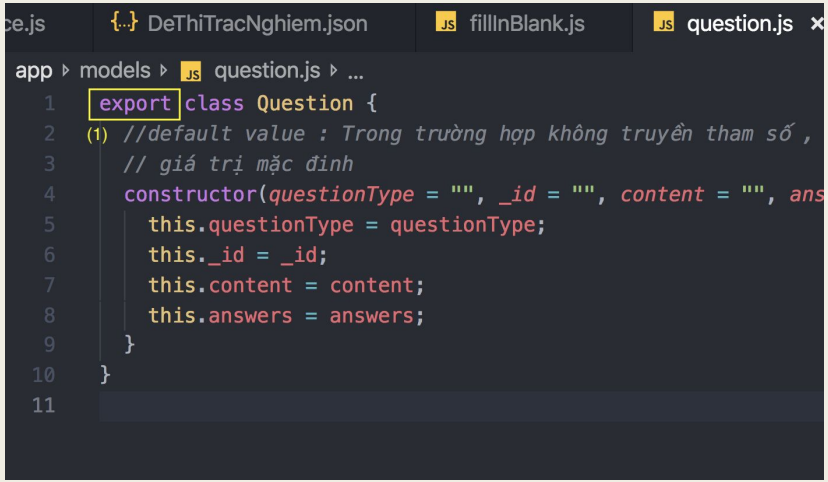
```
class MultipleChoice extends Question {
  constructor(...args) {
    super(...args);
  }
}
```

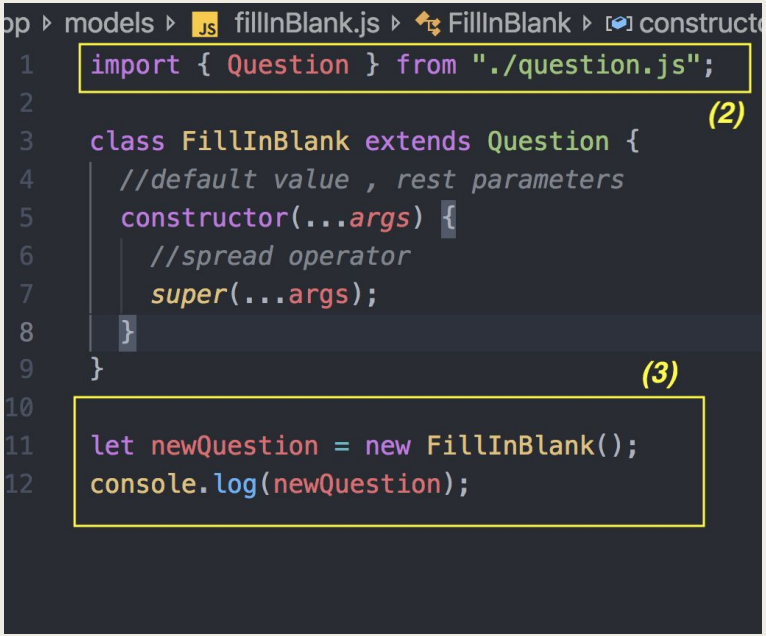
❑ *Bước 2 : Xây dựng lớp đối tượng*

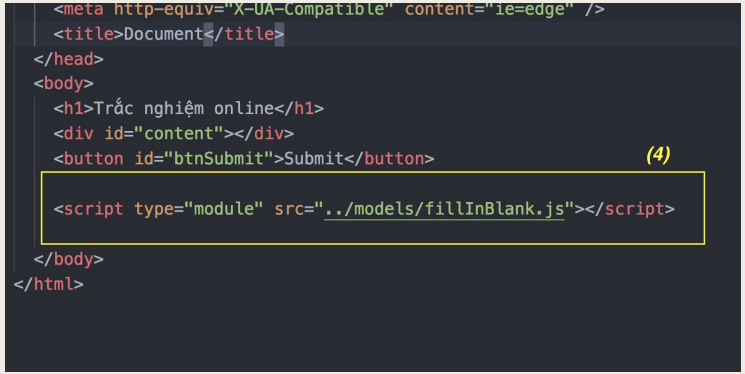
- Trước kia, để sử dụng được file javascript, ta cần gắn vào html thông qua thẻ script, có nghĩa ở đây, ta cần phải gắn 3 file question , multipleChoice và fillInBlank vào html theo đúng thứ tự.
- Sang Es6 , ta sẽ viết code theo hướng module, có nghĩa là mỗi file được coi là một module riêng biệt , ta sẽ dùng cú pháp import và export
- Vd: để sử dụng dụng Class Question trong file multipleChoice.js, ở file question.js, mình sẽ export class Question ra, sau đó ở file multipleChoice.js, mình sẽ import vào để sử dụng
- Sau đó, trong file index.js, nơi ta sẽ sử dụng 2 lớp MultipleChoice và FillInBlank, cũng làm tương tự , import vào để sử dụng
- Cuối cùng, ở index.html, ta chỉ cần link 1 file index.js vào là được

=> Thử nha :))

❑ *Cú pháp import , export trong ES6*



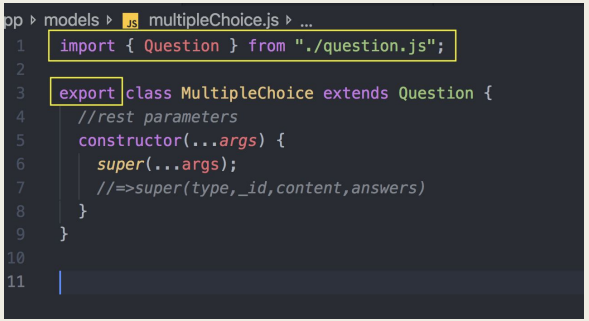


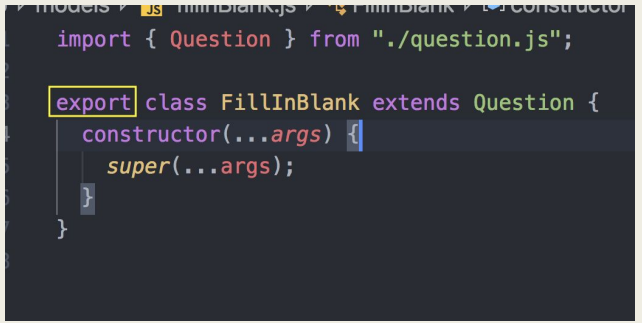


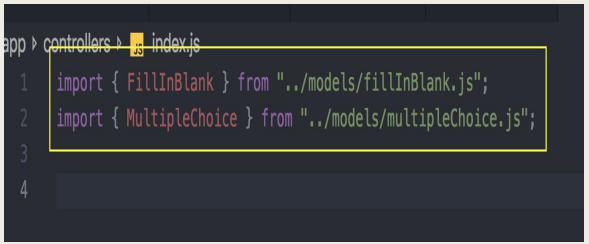
❑ *Cú pháp import , export trong ES6*

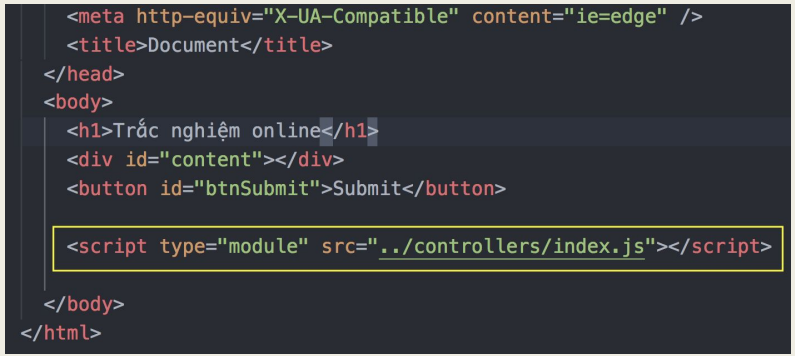
- (1): export class Question từ file question.js. Đây là 1 trong 2 loại export chúng ta sẽ học.Tạm thời gọi loại này là export “Có định danh” nhé các bạn.
- (2): import class Question vào file fillInBlank để sử dụng với cú phép như hình. Đối với cách export “Có định danh” này, thì cái ta export ra từ file question.js là 1 object như vầy {Question: Question}. Giả sử trong file question.js, ta export thêm 1 class A nữa , thì cái export ra sẽ là {Question:Question, A : A}. Do đó khi import ta phải dùng cú pháp như trong hình, lấy ra Question từ object được export.
- (3): mình thử khởi tạo 1 instance để check xem có hoạt động không nhé.
- (4): ở file index.html, mình script link file fillInBlank.js vào với **type là module,** không hề link file question.js
- => kết quả: code vẫn chạy bình thường nhé
- Okay, v giờ sẽ hoàn tất toàn bộ follow nha, làm tương tự với multipleChoice.js và index.js

❑ *Cú pháp import , export trong ES6*

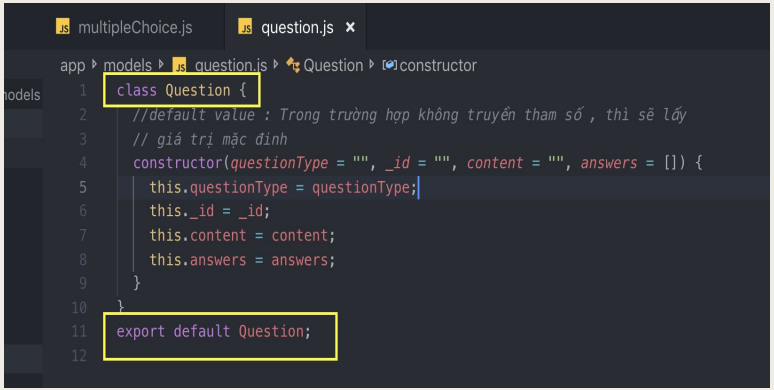


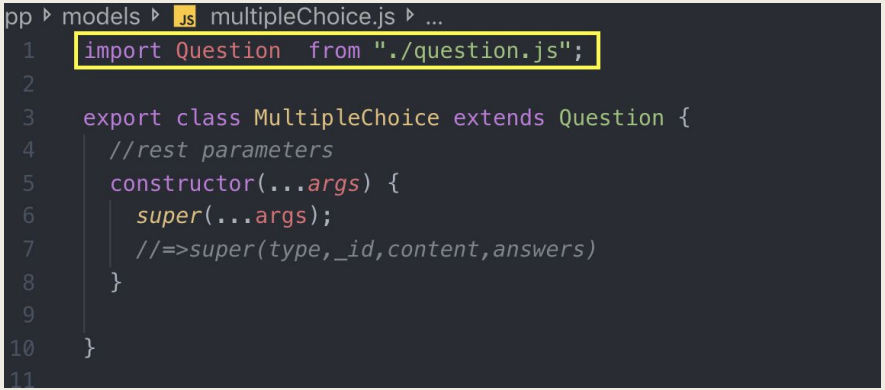






- Export loại 2 : Default
  - Lấy ví dụ: ở file question.js, ta chỉ export duy nhất class Question, do vậy thay vì dùng export “định danh”, ta có thể sử dụng export “default"
  - Với export default, thì khi import sẽ không cần cặp ngoặc {} và có thể dùng tên tuỳ ý





- Xây dựng phương thức render giao diện cho mỗi loại câu hỏi
  - Trước tiên, ta sẽ xem qua một câu hỏi có hình hài như nào, tức là một đối tượng câu hỏi bao gồm các thuộc tính nào.
  - Có 2 loại câu hỏi như sau

DeThiTracNghiem.json ...

```json
[
  {
    "questionType": 1,
    "_id": 1,
    "content": "Đáp án nào sau đây là đúng ?",
    "answers": [
      {
        "_id": "2",
        "exact": "false",
        "content": "Đáp án 1 cho câu 1",
        "STT": null
      },
      {
        "_id": "3",
        "exact": "false",
        "content": "Đáp án 2 cho câu 1",
        "STT": null
      },
      {
        "_id": "4",
        "exact": "false",
        "content": "Đáp án 3 cho câu 1",
        "STT": null
      },
      {
        "_id": "5",
        "exact": "true",
        "content": "Đáp án 4 cho câu 1",
        "STT": null
      }
    ]
  },
```

```json
  {
    "questionType": 2,
    "_id": 7,
    "content": "Hãy nhập vào đáp án đúng",
    "answers": [
      {
        "_id": "7",
        "exact": null,
        "content": "Đáp án 7",
        "STT": null
      }
    ]
  },
  {
```

- Xây dựng phương thức render giao diện cho mỗi loại câu hỏi
  - Phương thức render của mỗi loại: multiple choice bên trái, fill in blank bên phải
  - Phương thức render mình đề cập ở đây là một hàm, hàm này khi chạy trả ra một đoạn code html để hiện ra giao diện.

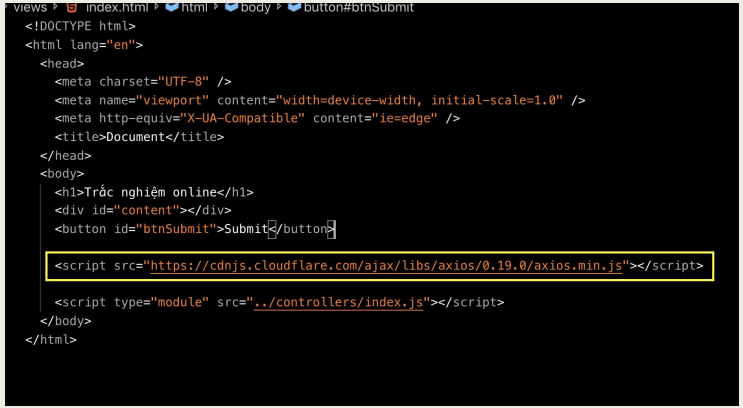
```
renderAnswer() {
  let answersText = "";
  for (let ans of this.answers) {
    answersText += `
          <div>
              <input type="radio" name="${this._id}" />
              <span> ${ans.content} </span>
          </div>
      `;
  }
  return answersText;
}

render(index) {
  //   string template
  return `
          <div>
              <h4>Câu hỏi ${index} : ${this.content} </h4>
              ${this.renderAnswer()}
          </div>
      `;
}
```

```
//
render(index) {
  return `
          <div>
              <h4>Câu hỏi ${index} : ${this.content} </h4>
              <input type="text"  />
          </div>
      `
}
```

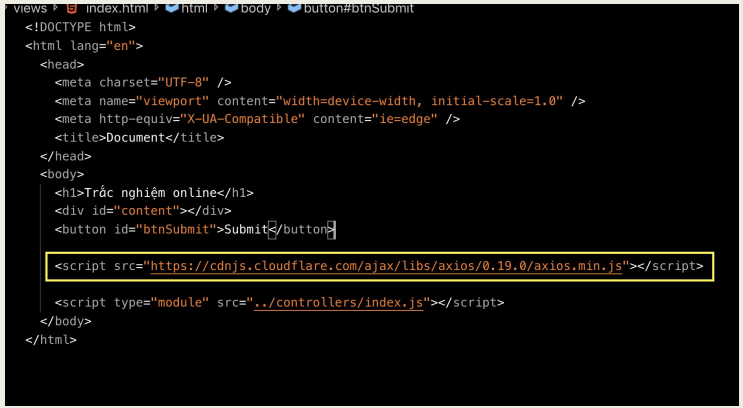
❑ *Xây dựng controller index.js*

- Giờ ta sẽ tiến hành xây dựng code chức năng cho trang web sau khi đã xây xong các lớp đối tượng
- Đầu tiên sẽ lấy data về trước, ở bài tập này , ta sẽ dùng axios để send request lên server backend lấy dữ liêu. Gắn cdn vào html (hình trái) và ở index.js, tiến hành get data ngay lúc đầu (hình phải)



❑ *Xây dựng controller index.js*

- Giờ ta sẽ tiến hành xây dựng code chức năng cho trang web sau khi đã xây xong các lớp đối tượng
- Đầu tiên sẽ lấy data về trước, ở bài tập này , ta sẽ dùng axios để send request lên server backend lấy dữ liêu. Gắn cdn vào html (hình trái) và ở index.js, tiến hành get data ngay lúc đầu (hình phải). Cách sử dụng như ajax , then tương ứng với done, catch tương ứng với fail.




```
const getData = () => {
  axios({
    method: "GET",
    url: "../../DeThiTracNghiem.json"
  })
    .then(res => {
      console.log(res);
    })
    .catch(err => {
      console.log(err);
    });
};
getData();
```

❑ *Xây dựng controller index.js*

- Sau khi lấy data, tiến hành map từ đối tượng server trả về sang đối tượng của mình để sử dụng phương thức render

```
import { FillInBlank } from "../models/fillInBlank.js";
import { MultipleChoice } from "../models/multipleChoice.js";

let questionList = [];
// lấy data từ DB
// arrow function

const getData = () => {
  axios({
    method: "GET",
    url: "../../DeThiTracNghiem.json"
  })
    .then(res => {
      mapDataToObject(res.data);
      renderQuestion();
    })
    .catch(err => {
      console.log(err);
    });
};
getData();

const mapDataToObject = (data = []) => {
  //destructuring : bóc tách phần tử

  // => const questionType = question.questionType
  questionList = data.map(question => {
    const { questionType, _id, content, answers } = question;
    if (questionType === 1) {
      return new MultipleChoice(questionType, _id, content, answers);
    }
    return new FillInBlank(questionType, _id, content, answers);
  });
};
```

Gọi hàm map data chạy sau khi đã lấy dữ liệu thành công

Tiến hành duyệt mảng data, kiểm tra loại câu hỏi, 1 là multiple choice, 2 là fillinblank, tiến hành tạo đối tượng tương ứng. Hàm map ở đây trả cho chúng ta một mảng mới

```
const renderQuestion = () => {
  let content = "";
  for (let i in questionList) {
    content += questionList[i].render(i * 1 + 1);
  }

  document.getElementById("content").innerHTML = content;
};
```

❑ *Xây dựng controller index.js*

- Sau khi lấy data, tiến hành map từ đối tượng server trả về sang đối tượng của mình để sử dụng phương thức render

```
import { FillInBlank } from "../models/fillInBlank.js";
import { MultipleChoice } from "../models/multipleChoice.js";

let questionList = [];
// lấy data từ DB
// arrow function

const getData = () => {
  axios({
    method: "GET",
    url: "../../DeThiTracNghiem.json"
  })
    .then(res => {
      mapDataToObject(res.data);
      renderQuestion();
    })
    .catch(err => {
      console.log(err);
    });
};
getData();

const mapDataToObject = (data = []) => {
  //destructuring : bóc tách phần tử

  // => const questionType = question.questionType
  questionList = data.map(question => {
    const { questionType, _id, content, answers } = question;
    if (questionType === 1) {
      return new MultipleChoice(questionType, _id, content, answers);
    }
    return new FillInBlank(questionType, _id, content, answers);
  });
};
```

Gọi hàm map data chạy sau khi đã lấy dữ liệu thành công

Tiến hành duyệt mảng data, kiểm tra loại câu hỏi, 1 là multiple choice, 2 là fillinblank, tiến hành tạo đối tượng tương ứng. Hàm map ở đây trả cho chúng ta một mảng mới

```
const renderQuestion = () => {
  let content = "";
  for (let i in questionList) {
    content += questionList[i].render(i * 1 + 1);
  }

  document.getElementById("content").innerHTML = content;
};
```

❑ *Xây dựng phương thức tính điểm cho multipleChoice*

Cho 4 radio button chung 1 class, mỗi radio button lưu 1 value là id của câu trả lời tương ứng

```
renderAnswer() {
  let answersText = "";
  for (let ans of this.answers) {
    answersText += `
        <div>
            <input type="radio" class="question-${this._id}" value="${ans._id}" name="${this._id}" />
            <span> ${ans.content} </span>
        </div>
    `;
  }
  return answersText;
}
checkExact(){
  //dom theo class trả ra 1 collection các thẻ html có class đó,
  //nên ở đây ta dùng spread operator để bung các phần tử của collection ra
  //và bỏ vào 1 mảng => listRadioTag là một mảng và ta có thể duyệt được
  const listRadioTag = [...document.getElementsByClassName('question-'+ this._id)];
  for(let radioTag of listRadioTag){
    //kiểm tra trong 4 câu trả lời, câu nào đang được chọn
    if(radioTag.checked){
      // lấy ra id câu trả lời đó dựa vào value của radio button
      const ansId = radioTag.value;
      // tìm ra câu trả lời đó là câu nào trong list 4 câu, nếu tìm ko thấy thì mặc đinh
      // là {}
      const ansObj = this.answers.find(ans => ans._id === ansId) || {};
      //trả về thuộc tính exact của câu đó, nếu đúng là true, sai là false
      return ansObj.exact;
    }
  }
  return false
}
```

- ***Xây dựng phương thức tính điểm cho multipleChoice***
  - Ở index.js, gắn sự kiên cho nút submit

```
const calcScore = () => {
  const result = questionList.reduce((rightAnwser, question) => {
    //hàm reduce phục vụ cho việc tính toán, ở đây, reduce sẽ giúp
    // duyệt mảng questionList mỗi lần duyệt sẽ công dồn biến rightAnswer (ban đầu là 0) lên
    // quy tắc cộng, false coi như 0, true là 1
    return (rightAnwser += (question.checkExact && question.checkExact()) || 0);
  }, 0);
  alert(result)
};

document.getElementById("btnSubmit").addEventListener("click", calcScore);
```

- ***Bài tập về nhà : xây dựng phương thức tính điểm tương tự cho loại câu hỏi fill in blank***