**ĐỀ THI THỬ SỐ 1**

**MÔN: PHÁT TRIỂN ỨNG DỤNG WEB NÂNG CAO**

**LỚP CTK44 – THỜI GIAN: 90 PHÚT**

**CÂU 1: (2 điểm)**

1. Học phần Phát triển ứng dụng Web nâng cao đã trải qua bao nhiêu buổi thực hành?
2. Có bao nhiêu bài thực hành (Lab) chính thức? Trong số đó, bạn thấy thích nhất là bài thực hành nào? Vì sao?
3. Những nội dung chính của bài thực hành (Lab) số 3 là gì?

***Trả lời:***

**CÂU 2: (2 điểm)**

Cho ba lớp Category (danh mục), Candy (kẹo) và CandyContext như sau:

public class Category

{

// Mã số danh mục

public int Id { get; set; }

// Tên danh mục

public string Name { get; set; }

// Cho biết hiển thị lên menu hay không

public bool ShowOnMenu { get; set; }

}

public class Candy

{

// Mã số

public int Id { get; set; }

// Tên mặt hàng kẹo

public string Name { get; set; }

// Giá bán (đơn vị là USD, luôn lớn hơn 0 và

// có phần lẻ thập phân. Ví dụ: 5.99).

public decimal Price { get; set; }

// Ngày hết hạn sử dụng

public DateTime ExpirationDate { get; set; }

// Mã số danh mục (Mỗi mặt hàng kẹo phải thuộc

// đúng một danh mục)

public int CategoryId { get; set; }

public Category Category { get; set; }

}

public class CandyContext : DbContext

{

public DbSet<Category> Categories { get; set; }

public DbSet<Candy> Candies { get; set; }

}

1. Hãy xác định kiểu dữ liệu cho các thuộc tính ShowOnMenu, Price, ExpirationDate, và CategoryId. (Sinh viên sửa trực tiếp mã nguồn ở trên bằng cách thay tên kiểu dữ liệu vào chỗ [……])
2. Giả sử, ta có lớp CandyService thực thi interface ICandyService chứa phương thức GetCandiesAsync. Hãy định nghĩa phương thức này để tìm các mặt hàng kẹo chưa hết hạn sử dụng theo tên mặt hàng (nếu giá trị truyền vào khác rỗng), đơn giá (miền giá min-max có giá trị hợp lệ), danh mục (nếu có truyền giá trị) hoặc tên danh mục (tìm chính xác theo tên danh mục nếu truyền giá trị khác rỗng) và sắp xếp kết quả tăng dần theo tên mặt hàng kẹo.

public async Task<IList<Candy>> GetCandiesAsync(

string name = null,

int? categoryId = null,

string categoryName = null,

decimal? minPrice = null,

decimal? maxPrice = null,

CancellationToken cancellationToken = default)

{

IQueryable<Candy> queryable = \_dbContext.Set<Candy>();

// Tìm kẹo chưa hết hạn sử dụng

queryable = queryable.Where(i => DateTime.Now <= i.ExpirationDate);

if (!string.IsNullOrWhiteSpace(name))

{

queryable = queryable.Where(i => i.Name.Contains(name));

}

if (categoryId.HasValue)

{

queryable = queryable.Where(c => c.CategoryId == categoryId);

}

else if (!string.IsNullOrWhiteSpace(categoryName))

{

queryable = queryable.Where(i => i.Category.Name == categoryName);

}

if (minPrice.HasValue)

{

queryable = queryable.Where(c => c.Price >= minPrice);

}

if (maxPrice.HasValue)

{

queryable = queryable.Where(c => c.Price <= maxPrice);

}

queryable = queryable.OrderBy(c => c.Name);

return await queryable.Include(i => i.Category).Select(i => new Candy()

{

Name = i.Name,

Price = i.Price,

ExpirationDate = i.ExpirationDate,

CategoryId = i.CategoryId,

Category = i.Category,

}).ToListAsync(cancellationToken);

}

**CÂU 3: (3 điểm)**

Bạn An sử dụng kết quả ở câu 2 để định nghĩa một API Endpoint như sau:

app.MapGet("/candies", (ICandyService service, CandyFilterModel model) =>

{

var candies = await service.GetCandiesAsync(

model.Name, model.CategoryId, model.CategoryName,

model.MinPrice, model.MaxPrice);

return candies;

})

.WithName("GetCandies")

.Produces<Candy>()

.WithOpenApi();

1. Hãy giúp bạn An định nghĩa lớp CandyFilterModel.
2. Hãy giúp bạn An tìm những lỗi sai trong đoạn mã trên và sửa lại cho đúng.
3. Hãy giải thích ý nghĩa của toàn bộ đoạn mã trên (sau khi đã sửa).
4. Hãy nêu một đề xuất để cải thiện hiệu suất cho API Endpoint trên.

***Trả lời:***

a. public class CandyFilterModel

{

public string Name { get; set; }

public decimal? MinPrice { get; set; }

public decimal? MaxPrice { get; set; }

public int? CategoryId { get; set; }

public string CategoryName { get; set; }

}

b. A picture containing text, screenshot, software

Description automatically generated

c. Lấy danh sách candy theo các tiêu chí (name, categoryId, …)

d. Sử dụng caching để giảm truy vấn, kỹ thuật phân trang và đánh index cho CSDL để tăng tốc độ truy vấn

**CÂU 4: (3 điểm)**

Sau khi xây dựng API Endpoint (ở câu 3) hoàn chỉnh, bạn An tạo một React component – tên là CandyList – để hiển thị danh sách mặt hàng kẹo lên ứng dụng Web. Bạn An định nghĩa mã nguồn cho component này như sau:

import React, { useState } from 'react'

const CandyList = () => {

    const [candies, setCandies] = useState([]);

    fetch(`https://localhost:7076/candies`)

        .then(response => response.json())

        .then(data => setCandies(data));

    return (

        <div class="product-list">

            {candies.map((item) => (

               <div className="product-item">

                    <h1>item.name</h1>

                    <p>$ { item.price }</p>

                </div>

            ))}

        </div>

    )

}

export default CandyList;

1. Hãy cho biết đoạn mã trên chứa những lỗi sai hoặc cảnh báo nào? Giải thích nguyên nhân gây ra lỗi.
2. Hãy giúp bạn An cách sửa lỗi và viết lại mã nguồn đúng.
3. Hãy bổ sung mã lệnh để hiển thị dòng thông báo “***Đang tải dữ liệu …***” trong khi chờ kết quả trả về từ API hoặc thông báo lỗi “***Không thể tải dữ liệu***” nếu gặp lỗi khi gọi API.

***Trả lời:***

a.

Cảnh báo 1: cảnh báo thuộc tính class của thẻ html phải thay bằng className, do trong JavaScript có từ khóa class bị trùng

Cảnh báo 2: cảnh báo thiếu key cho mỗi item khi dùng map để render trong JSX, do thuộc tính key giúp cho DOM ảo của React xác định được item nào có sự thay đổi, mỗi key bên trong mảng khi map có giá trị là duy nhất

Lỗi 1: lỗi bị vòng lặp vô hạn, do setCadies trong kết quả trả về từ api gọi ở hàm fetch lặp lại mỗi khi component bị re-render, khi set lại state dẫn đến việc component bị re-render 🡪 dòng code sử dụng gọi api lại được chạy lại (do không có điều kiện gọi lại) 🡪 mà trong kết quả trả về lại setCadies 🡪 dẫn đến việc xảy ra tình trạng vòng lặp vô hạn

Lỗi 2: tên candy không hiển thị ra giao diện do dòng code item.name thiếu cặp dấu ngoặc ngọn bao bọc trong JSX như sau: {item.name}

b.

import React, { useState, useEffect } from "react";

const CandyList = () => {

   const [candies, setCandies] = useState([]);

   useEffect(() => {

      fetch(`https://localhost:7095/candies`)

         .then((response) => response.json())

         .then((data) => {

            console.log(data);

            setCandies(data);

         });

   }, []);

   return (

      <div className="product-list">

         {candies.map((item) => (

            <div className="product-item" key={item.id}>

               <h1>{item.name}</h1>

               <p>$ {item.price}</p>

            </div>

         ))}

      </div>

   );

};

export default CandyList;

c.

import React, { useState, useEffect } from "react";

const CandyList = () => {

   const [candies, setCandies] = useState([]);

   const [errorMessage, setErrorMessage] = useState("");

   useEffect(() => {

      fetch(`https://localhost:7095/candies?Name=abc&MinPrice=fabwfawf`)

         .then((response) => response.json())

         .then((data) => {

            if (data.length === 0) {

               setErrorMessage("Không có dữ liệu");

            }

            setCandies(data);

         })

         .catch((err) => {

            setErrorMessage("Không thể tải dữ liệu");

         });

   }, []);

   if (errorMessage) {

      return <p>{errorMessage}</p>;

   }

   return (

      <div className="product-list">

         {candies.length > 0 ? (

            candies.map((item) => (

               <div className="product-item" key={item.id}>

                  <h1>{item.name}</h1>

                  <p>$ {item.price}</p>

               </div>

            ))

         ) : (

            <p>Đang tải dữ liệu</p>

         )}

      </div>

   );

};

export default CandyList;

**--- HẾT ---**

**ĐỀ THI THỬ SỐ 2**

**MÔN: PHÁT TRIỂN ỨNG DỤNG WEB NÂNG CAO**

**LỚP CTK44 – THỜI GIAN: 90 PHÚT**

**CÂU 1: (2 điểm)**

1. Những nội dung chính của bài thực hành (Lab) số 5 là gì?
2. Bài thực hành số 4 (Web API) có một phiên bản cập nhật. Hãy trình bày vắn tắt nội dung thay đổi và lý do của việc cập nhật đó.

***Trả lời:***

**CÂU 2: (3 điểm)**

Cho hai lớp Category (danh mục) và CandyContext như sau:

public class Category

{

// Mã số danh mục

public int Id { get; set; }

// Tên danh mục

public string Name { get; set; }

// Cho biết hiển thị lên menu hay không

public bool ShowOnMenu { get; set; }

}

public class CandyContext : DbContext

{

public DbSet<Category> Categories { get; set; }

}

1. Hãy định nghĩa lớp **CategoryEditModel** chứa các thuộc tính cần thiết để nhận thông tin từ phía Client, được dùng cho việc thêm mới hoặc cập nhật một danh mục.
2. Giả sử thư viện **FluentValidation** được sử dụng để kiểm tra tính hợp lệ của dữ liệu nhập. Hãy cài đặt lớp **CategoryValidator** để thiết lập ràng buộc giá trị của thuộc tính **Name** phải khác rỗng. Nội dung thông báo lỗi tương ứng là “***Tên danh mục không được để trống***”.
3. Giả sử, ta có lớp CandyService thực thi interface ICandyService chứa phương thức AddOrUpdateCategoryAsync. Hãy định nghĩa phương thức này để thêm mới hoặc cập nhật một danh mục dựa vào mã số danh mục.

public async Task<Category> AddOrUpdateCategoryAsync(

string categoryName,

bool showOnMenu = false,

int? categoryId = null,

CancellationToken cancellationToken = default)

{

// Sinh viên viết mã lệnh ở đây

}

***Trả lời:***

a.

public class CategoryEditModel

{

public string Name { get; set; }

public bool ShowOnMenu { get; set; }

}

b.

public class CategoryValidator : AbstractValidator<CategoryEditModel>

{

public CategoryValidator()

{

RuleFor(a => a.Name)

.NotEmpty()

.WithMessage("Tên chủ đề không được để trống");

}

}

c.

public async Task<Category> AddOrUpdateCategoryAsync(

string categoryName,

bool showOnMenu = false,

int? categoryId = null,

CancellationToken cancellationToken = default

)

{

Category category = new Category()

{

Id = categoryId ?? 0,

Name = categoryName,

ShowOnMenu = showOnMenu,

};

if (category.Id == 0)

{

\_dbContext.Categories.Add(category);

} else

{

\_dbContext.Categories.Update(category);

}

bool isSuccess = await \_dbContext.SaveChangesAsync(cancellationToken) > 0;

return isSuccess ? category : null;

}

**CÂU 3: (3 điểm)**

Bạn Bình sử dụng kết quả ở câu 2 để định nghĩa một API Endpoint như sau:

app.MapPost("/categories", (ICandyService service, CategoryEditModel model) =>

{

var category = await service.AddOrUpdateCategoryAsync(

model.Id, model.Name, model.ShowOnMenu);

return category;

})

.WithName("AddCategory")

.Produces<Candy>()

.WithOpenApi();

1. Hãy giúp bạn Bình tìm những lỗi sai trong đoạn mã trên và sửa lại cho đúng.
2. Hãy bổ sung mã lệnh vào đoạn mã trên để kiểm tra tính hợp lệ của dữ liệu nhập. Nếu dữ liệu không hợp lệ, trả về mã 400 Bad Request kèm theo thông báo lỗi. Giả sử rằng các thiết lập cần thiết để sử dụng thư viện FluentValidation đã được cài đặt hợp lệ ở tập tin Program.cs.
3. Hãy giải thích ý nghĩa của toàn bộ đoạn mã trên (sau khi đã sửa và bổ sung).

***Trả lời:***

a.

// AddCategory

app.MapPost("/categories", async (

ICategoryService service,

CategoryEditModel model,

IValidator<CategoryEditModel> validator

) =>

{

model.Name = model.Name.Trim();

var validationResult = await validator.ValidateAsync(model);

if (!validationResult.IsValid)

{

return Results.BadRequest(validationResult.Errors.ToResponse());

}

Category categoryResult = await service.AddOrUpdateCategoryAsync(model.Name, model.ShowOnMenu);

return Results.Ok(categoryResult);

})

.WithName("AddCategory")

.Accepts<Category>("application/json")

.Produces<Category>()

.WithOpenApi();

c. Tạo endpoint thêm category với phương thức POST, cắt bỏ khoảng trắng đầu và cuối thuộc tính Name của dữ liệu category thêm mới, kiểm tra tính hợp lệ dữ liệu gửi kèm với FluentValidation, cuối cùng là thực thi việc thêm danh mục vào CSDL

**CÂU 4: (2 điểm)**

Sau khi xây dựng API Endpoint (ở câu 3) hoàn chỉnh, bạn Bình tạo một React component – tên là EditCategory – để tạo biểu mẫu cho phép người dùng nhập thông tin và tạo một danh mục mới. Bạn Bình định nghĩa mã nguồn cho component này như sau:

import React, { useState } from 'react';

const EditCategory = () => {

    const [name, setName] = useState('');

    const [showOnMenu, setShowOnMenu] = useState(false);

    return (

        <div>

            <h1>Add New Category</h1>

            <form>

                <label> Name:

                    <input type="text"

                        onChange={(e) => setName(e.target.value)} />

                </label>

                <label> Show On Menu:

                    <input type="checkbox"

                        onChange={(e) => setShowOnMenu(!showOnMenu)} />

                </label>

                <button type="submit">Submit</button>

            </form>

        </div>

    )

}

export default EditCategory;

1. Bình cho rằng có những biến trạng thái không được sử dụng và việc xử lý các sự kiện onChange là không cần thiết. Vì vậy, cậu ấy quyết định sử dụng hook useRef thay cho hook useState. Bạn hãy giúp Bình cài đặt lại component trên.
2. Định nghĩa hàm xử lý sự kiện để khi người dùng nhấn nút Submit thì chương trình gửi dữ liệu nhập đến API Endpoint đã tạo ở câu 3. (Sinh viên tùy chọn sử dụng fetch API hoặc thư viện axios).
3. Hãy bổ sung mã lệnh để hiển thị dòng thông báo “***Đang gửi dữ liệu …***” trong khi chờ kết quả trả về từ API hoặc thông báo lỗi “***Đã có lỗi xảy ra***” nếu gặp lỗi khi gọi API.

***Trả lời:***

import React, { useState, useRef } from "react";

const EditCategory = () => {

   const nameRef = useRef(null);

   const showOnMenuRef = useRef(null);

   const messageRef = useRef(null);

   function handleSubmit(e) {

      e.preventDefault();

      const category = {

         name: nameRef.current.value,

         showOnMenu: showOnMenuRef.current.checked,

      };

      messageRef.current.innerText = "Đang gửi dữ liệu...";

*// Submit*

      fetch("https://localhost:7095/categories", {

         method: "POST",

         headers: {

            "Content-Type": "application/json",

         },

         body: JSON.stringify(category),

      })

         .then((response) => response.json())

         .then((response) => (messageRef.current.innerText = ""))

         .catch((err) => {

            messageRef.current.innerText = "Đã có lỗi xảy ra";

         });

   }

   return (

      <div>

         <h1>Add New Category</h1>

         <form onSubmit={handleSubmit}>

            <label>

               {" "}

               Name:

               <input ref={nameRef} type="text" />

            </label>

            <label>

               {" "}

               Show On Menu:

               <input ref={showOnMenuRef} type="checkbox" />

            </label>

            <button type="submit">Submit</button>

            <p ref={messageRef}></p>

         </form>

      </div>

   );

};

export default EditCategory;

**--- HẾT ---**