.Net Core Web Service – MongoDB

# Thiết lập csdl

## Cài đặt mongodb

Vào trang <https://www.mongodb.com/docs/manual/installation/> download và cài đặt

Cài đặt Mongo Compass

A screenshot of a computer

Description automatically generated

Tạo database

A screenshot of a computer

Description automatically generated

Thêm 2 document

|  |
| --- |
| [  {  "Name": "Design Patterns",  "Price": 54.93,  "Category": "Computers",  "Author": "Ralph Johnson"  },  {  "Name": "Clean Code",  "Price": 43.15,  "Category": "Computers",  "Author": "Robert C. Martin"  }  ] |

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

# Api

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Cài đặt MongoDB Driver: vào menu Tools->NuGet Packages manager->…

A screenshot of a computer

Description automatically generated

Thêm một thư mục mới Models

A screenshot of a computer

Description automatically generated

## Thêm configuration model

Class Book

A computer screen shot of text

Description automatically generated

Class BookStoreDatabaseSettings

A screen shot of a computer

Description automatically generated

Mở file Program.cs, them code, them đoạn code được chọn vào. (nhớ using SampleAPI.Models)

A computer screen shot of a program

Description automatically generated

## Thêm các tác vụ CRUD cho service

Thêm thư mục services vào project

Thêm một lớp mới có tên BookService

|  |
| --- |
| using Microsoft.Extensions.Options;  using MongoDB.Driver;  using SampleAPI.Models;  namespace SampleAPI.Services  {  public class BooksService  {  private readonly IMongoCollection<Book> \_booksCollection;  public BooksService(  IOptions<BookStoreDatabaseSettings> bookStoreDatabaseSettings)  {  var mongoClient = new MongoClient(  bookStoreDatabaseSettings.Value.ConnectionString);  var mongoDatabase = mongoClient.GetDatabase(  bookStoreDatabaseSettings.Value.DatabaseName);  \_booksCollection = mongoDatabase.GetCollection<Book>(  bookStoreDatabaseSettings.Value.BooksCollectionName);  }  public async Task<List<Book>> GetAsync() =>  await \_booksCollection.Find(\_ => true).ToListAsync();  public async Task<Book?> GetAsync(string id) =>  await \_booksCollection.Find(x => x.Id == id).FirstOrDefaultAsync();  public async Task CreateAsync(Book newBook) =>  await \_booksCollection.InsertOneAsync(newBook);  public async Task UpdateAsync(string id, Book updatedBook) =>  await \_booksCollection.ReplaceOneAsync(x => x.Id == id, updatedBook);  public async Task RemoveAsync(string id) =>  await \_booksCollection.DeleteOneAsync(x => x.Id == id);  }  } |

Chú ý, trong trường hợp này, đối tượng BookStoreDatabaseSettings được lấy theo thông qua kỹ thuật DI từ constructor.

Thêm code sau vào Program.cs

A screenshot of a computer

Description automatically generated

## Thêm Controller

Thêm lớp BooksController.cs với nội dung sau vào project

|  |
| --- |
| using Microsoft.AspNetCore.Mvc;  using SampleAPI.Models;  using SampleAPI.Services;  namespace SampleAPI.Controllers  {  [ApiController]  [Route("api/[controller]")]  public class BooksController : ControllerBase  {  private readonly BooksService \_booksService;  public BooksController(BooksService booksService) =>  \_booksService = booksService;  [HttpGet]  public async Task<List<Book>> Get() =>  await \_booksService.GetAsync();  [HttpGet("{id:length(24)}")]  public async Task<ActionResult<Book>> Get(string id)  {  var book = await \_booksService.GetAsync(id);  if (book is null)  {  return NotFound();  }  return book;  }  [HttpPost]  public async Task<IActionResult> Post(Book newBook)  {  await \_booksService.CreateAsync(newBook);  return CreatedAtAction(nameof(Get), new { id = newBook.Id }, newBook);  }  [HttpPut("{id:length(24)}")]  public async Task<IActionResult> Update(string id, Book updatedBook)  {  var book = await \_booksService.GetAsync(id);  if (book is null)  {  return NotFound();  }  updatedBook.Id = book.Id;  await \_booksService.UpdateAsync(id, updatedBook);  return NoContent();  }  [HttpDelete("{id:length(24)}")]  public async Task<IActionResult> Delete(string id)  {  var book = await \_booksService.GetAsync(id);  if (book is null)  {  return NotFound();  }  await \_booksService.RemoveAsync(id);  return NoContent();  }  }  } |

File Program.cs sẽ có dạng như sau

|  |
| --- |
| using SampleAPI.Models;  using SampleAPI.Services;  var builder = WebApplication.CreateBuilder(args);  // Add services to the container.  builder.Services.Configure<BookStoreDatabaseSettings>(  builder.Configuration.GetSection("BookStoreDatabase"));  builder.Services.AddSingleton<BooksService>();  builder.Services.AddControllers()  .AddJsonOptions(  options => options.JsonSerializerOptions.PropertyNamingPolicy = null);  // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle  builder.Services.AddEndpointsApiExplorer();  builder.Services.AddSwaggerGen();  var app = builder.Build();  // Configure the HTTP request pipeline.  if (app.Environment.IsDevelopment())  {  app.UseSwagger();  app.UseSwaggerUI();  }  app.UseHttpsRedirection();  app.UseAuthorization();  app.MapControllers();  app.Run(); |

# Chạy ứng dụng

Khi chạy ứng dụng, windows sẽ hỏi về việc trust cho SSL certificate. Chọn YES cho 2 cửa sổ sau

A screenshot of a computer error

Description automatically generated

A screenshot of a computer security warning

Description automatically generated

Truy cập trang <https://localhost:7295/swagger/index.html>

A screenshot of a computer

Description automatically generated

Truy cập và thực thi các api với swagger

A screenshot of a computer

Description automatically generated