Завдання:

Розробіть Web API для керування колекцією книг. API повинен надавати можливості для створення, отримання, оновлення та видалення книг. Використайте впровадження залежностей для організації коду та забезпечення гнучкості.

**Вимоги:**

**Модель даних:**

Створіть клас **Book** з такими властивостями:

**Id** (типу int)

**Title** (типу string)

**Author** (типу string)

**PublishedYear** (типу int)

**Інтерфейс репозиторію:**

Визначте інтерфейс **IBookService** з такими методами:

IEnumerable<Book> **GetAll**()

Book **GetById**(int id)

**Add**(Book book)

**Update**(Book book)

**Delete**(Guid id)

Створіть сервіс **BookService**, який реалізує інтерфейс **IBookService**, та зареєструйте його в DI як Singleton (щоб протягом життя нашого сервісу, дані з нього не видалялись). На рівні цього сервісу створіть приватну змінну типу List<Book> з назвою \_books, яка буде нашим тимчасовим сховищем книжок. В цілому з назв методів зрозуміло, що вони мають робити, але зробимо уточнення – всі ці дії мають впливати на змінну List<Book>, тобто метод Add має додавати книгу в список, а метод Delete видаляти.

Створіть інтерфейс **IAuthorService** з методами:

bool **IsAuthorExist**(string author)

void **Add**(string author)

**void Delete**(string author)

Створіть сервіс **AuthorService**, який реалізує інтерфейс **IAuthorService**, та має в собі приватну змінну типу List<string> \_authors. Відповідно методи мають додавати нових авторів за іменем і видаляти так само в цьому списку, а метод IsAuthorExist має перевіряти чи існує автор у списку, та повертати true, якщо автор існує і false, якщо його не існує. Сервіс додати в DI як Singleton.

Створіть інтерфейс **IBookValidationService**, який має такий метод:

Bool **AddBookValidation**(Book book)

Створіть сервіс **BookValidationService**, який реалізовує **IBookValidationService**. Метод AddBookValidation має робити просту перевірку книги перед тим, щоб її додати. Якщо назва книги більша за 30 символів, або якщо PublishedYear більший за поточний час, або якщо автора книги не існує в сервісі AuthorService, то значить валідація не буде успішною, і метод поверне false, і така книга не буде створена. Метод AddBookValidation, має виконуватися як у під час створення нових книжок, так і під час зміни існуючих книжок. Додайте цей сервіс у DI як Transient.

Створіть контролер **BooksController**, який містить такі методи:

GET /api/books — отримати всі книги

GET /api/books/{id} — отримати книгу за ідентифікатором

POST /api/books — додати нову книгу

POST /api/books/{id} — оновити існуючу книгу

POST /api/books/{id} — видалити книгу

Створіть контролер AuthorController, який містить такі методи:

POST /api/author — створити автора

POST /api/ author /{name} — видалити автора за ім’ям

В результаті виконання завдання, контролер BooksController має делегувати запити до BookService, а BookService має делегувати запит на валідацію під час додавання нової книги чи зміни старої книги до BookValidationService, а цей сервіс в свою чергу, має звертатися до сервіса AuthorService, для перевірки валідності авторів. Відповідно має працювати AuthorController, де логіка простіша, і основною задачею там є просто додавання авторів та їх перевірка.