כן, בהחלט! תוכל ליצור פונקציה שמקבלת מספר פרמטרים אופציונליים, כך שתוכל לבצע חיפוש לפי אחד או יותר מהם. ניתן להשתמש ב-`Nullable` כדי לאפשר ערכים ריקים ולבדוק אילו פרמטרים נמסרו על ידי המשתמש.

### דוגמה לפונקציה לחיפוש ספרים

```csharp

[HttpGet("search")]

public IActionResult SearchBooks([FromQuery] int? id, [FromQuery] string name, [FromQuery] string category)

{

var results = Lists\_Of\_The\_Librery.Books.AsQueryable();

if (id.HasValue)

{

results = results.Where(book => book.Code == id.Value);

}

if (!string.IsNullOrEmpty(name))

{

results = results.Where(book => book.Name.Contains(name, StringComparison.OrdinalIgnoreCase));

}

if (!string.IsNullOrEmpty(category))

{

results = results.Where(book => book.Category.Equals(category, StringComparison.OrdinalIgnoreCase));

}

return Ok(results.ToList());

}

```

### הסבר על הקוד:

1. \*\*פרמטרים אופציונליים\*\*:

- `int? id` - פרמטר אופציונלי שיכול להיות `null`.

- `string name` - גם פרמטר אופציונלי, ואם הוא ריק או `null`, לא נבצע חיפוש לפי שם.

- `string category` - פרמטר נוסף לחיפוש לפי קטגוריה.

2. \*\*שימוש ב-`AsQueryable`\*\*:

- זה מאפשר לנו לבנות שאילתות דינמיות על בסיס הפרמטרים שהתקבלו.

3. \*\*תנאים לחיפוש\*\*:

- עבור כל פרמטר, אם הוא נמסר (כלומר לא `null`), אנחנו מסננים את התוצאות.

- לדוגמה, אם `id` נמסר, אנו מסננים לפי הקוד. אם `name` נמסר, אנו מחפשים ספרים שכוללים את השם (באופן לא רגיש לרישיות).

4. \*\*החזרת התוצאות\*\*:

- בסוף, מחזירים את התוצאות כרשימה.

### דוגמה לקריאת ה-API

תוכל לבצע קריאות שונות כמו:

1. חיפוש לפי ID:

```

GET /api/books/search?id=123

```

2. חיפוש לפי שם:

```

GET /api/books/search?name=Harry

```

3. חיפוש לפי קטגוריה:

```

GET /api/books/search?category=Fiction

```

4. חיפוש לפי מספר פרמטרים:

```

GET /api/books/search?id=123&name=Harry

```

### סיכום

באמצעות הגישה הזו, אתה יכול לאפשר למשתמשים לחפש ספרים בצורה גמישה על פי קריטריונים שונים. אם יש לך שאלות נוספות או שאתה רוצה להרחיב על משהו מסוים, אני כאן!