Test Description

You have been tasked with implementing a search functionality for a web application that allows users to find relevant documents based on a search query. The search query can be a long text, and the application should return documents that contain all or any of the words in the search query.

Your task is to implement a C# method that performs the search based on the requirements above. You will be given a SQL Server database that contains a single table, documents, with the following columns:

- **`id`** (int): the unique identifier of the document.
- **`title`** (nvarchar(255)): the title of the document.
- **`content`** (nvarchar(max)): the content of the document.

The method signature is as follows:

```
public static List<Document> SearchDocuments(string query, bool matchAll);
```

The method should return a list of **`Document`** objects that match the search query, based on the **`matchAll`** parameter. If **`matchAll`** is true, the method should return documents that contain all the words in the query. If **`matchAll`** is **`false`**, the method should return documents that contain any of the words in the query.

A **`Document`**object has the following properties:

- **`Id`** (int): the unique identifier of the document.
- **`Title`** (string): the title of the document.
- **`Content`** (string): the content of the document.

Requirements

- The search query should be case-insensitive.
- The search should not be sensitive to leading or trailing whitespace.
- The search should not be sensitive to punctuation or special characters.
- The search should be optimized for performance.

Evaluation Criteria

Your code will be evaluated based on the following criteria:

1. **Correctness** - your code should produce the correct search results.
2. **Efficiency** - your code should be efficient and not cause unnecessary database queries.
3. **Readability** - your code should be easy to read and understand.
4. **Maintainability** - your code should be easy to maintain and extend.
5. **Testability** - your code should be easy to test.
6. **Error Handling** - your code should handle errors and exceptions gracefully.

Tips

- You may use any libraries or frameworks that are commonly used in .NET Framework development.
- You may assume that the database connection string is provided and that the database schema is already set up.
- You may write additional helper methods or classes as needed.
- You may write unit tests for your implementation if you have time.

Good luck!