Lab Assignment: Unit Testing Fundamentals with .NET

You are tasked with creating a library management system that will be used to manage books and their lending operations. You need to implement the core functionality and write comprehensive unit tests to ensure the system works as expected.

Required Classes

1. `Book` class with properties:
   - ISBN (string)
   - Title (string)
   - Author (string)
   - PublicationYear (int)
   - IsAvailable (bool)
2. `LibraryManager` class with methods:
   - `AddBook(Book book)`
   - `RemoveBook(string isbn)`
   - `GetBookByISBN(string isbn)`
   - `LendBook(string isbn)`
   - `ReturnBook(string isbn)`
   - `GetAllAvailableBooks()`

Testing Requirements

Your task is to create comprehensive unit tests for the `LibraryManager` class. Your tests should cover:

1. **Positive Testing Scenarios:**
   - Adding a new book successfully
   - Retrieving an existing book by ISBN
   - Lending an available book
   - Returning a borrowed book
   - Getting all available books
   - Removing an existing book
2. **Negative Testing Scenarios:**
   - Adding a book with duplicate ISBN
   - Retrieving a non-existent book
   - Lending a book that's already lent out
   - Returning a book that wasn't lent
   - Removing a non-existent book
3. **Boundary Testing:**
   - Adding books with empty/null ISBN

- Adding books with empty/null Title
- Adding books with publication year in the future
- Testing the system with zero books
- Testing the system with a large number of books

4. **Edge Cases:**

   - Testing with special characters in ISBN/Title
   - Testing with very long strings for Title/Author
   - Testing with minimum/maximum valid years for PublicationYear

Implementation Requirements

1. The `AddBook` method should:

   - Not allow duplicate ISBNs
   - Validate that ISBN, Title, and Author are not null or empty
   - Validate that PublicationYear is not in the future

2. The `LendBook` method should:

   - Throw an exception if the book doesn't exist
   - Throw an exception if the book is already lent out
   - Update the IsAvailable status appropriately

3. The `ReturnBook` method should:

   - Throw an exception if the book doesn't exist
   - Throw an exception if the book is already available
   - Update the IsAvailable status appropriately

Testing Guidelines

1. Use NUnit framework for writing tests
2. Implement proper test organization using TestFixture and Test attributes
3. Use appropriate Assert methods for validations
4. Implement SetUp and TearDown methods where necessary
5. Use meaningful test names that describe the scenario being tested
6. Include both positive and negative assertions in your tests
7. Use test categories to organize different types of tests
8. Implement proper exception handling and test for exceptions where appropriate

Bonus Challenges

1. Implement and test a search functionality that can search books by title or author
2. Add and test a reservation system for books that are currently lent out
3. Implement and test a due date system for lent books

Deliverables

1. Complete solution with both projects (LibraryManagement and LibraryManagement.Tests)
2. Well-documented test classes with all scenarios covered
3. A brief report explaining your testing approach and any assumptions made
4. List of edge cases you identified and how you handled them

Evaluation Criteria

1. Test coverage and completeness
2. Test organization and clarity
3. Proper use of testing patterns and best practices
4. Code quality and documentation
5. Handling of edge cases and error conditions

Note

- Make sure to follow C# coding conventions
- Use meaningful names for your test methods
- Include comments explaining complex test scenarios
- Use test data builders or helper methods to reduce code duplication in tests