Solution Review: Implement an Abstract Method in a Base Class

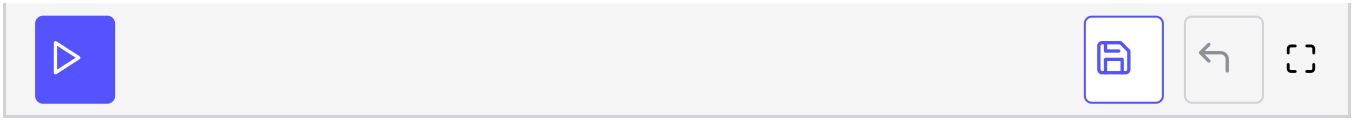
This review provides a detailed analysis to solve the 'Implement an Abstract Method in a Base Class' challenge.

WE'LL COVER THE FOLLOWING

Solution

```
// Abstarct Book Class
abstract class Book {

  // Private Fields
  private string _name;
  private string _author;
  private string _price;

  protected string Name{
    get {return this._name;}
  }
   protected string Author{
    get {return this._author;}
  }
   protected string Price{
    get {return this._price;}
  }

  // Parameterized Constructor
  public Book(string name, string author, string price) {
    this._name = name;
    this._author = author;
    this._price = price;
  }

  // Abstract Method
  public abstract string GetDetails();

}

// Class MyBook extending Book Class
class MyBook : Book {
```

```
  // Parameterized Constructor
  public MyBook(string name, string author, string price)
    : base(name, author, price)
  { }

  // Overrideing the GetDetails Abstract Method of the Base Class
  public override string GetDetails() {
   return Name + ", " + Author + ", " + Price;
  }

}

class Demo {

  public static void Main(string[] args) {
    Book myBook = new MyBook("Harry Potter", "J.k. Rowling", "100");
    Console.WriteLine(myBook.GetDetails());

  }

}
```



Explanation

- **Line 32:** Extended the `MyBook` class from the `Book` class.
- **Line 36:** Called the base class constructor.
- **Line 40:** The abstract method `GetDetails()` is overridden.
- **Line 41:** Implemented the overridden abstract method `GetDetails()` which returns the concatenated details in the form of a string.