***Question #4 (10 pts.)***

During our discussions of design patterns over the last two weeks we’ve discussed many reasons why incorporating them into your software design can be advantageous. Briefly discuss one instance where using a design pattern may not be beneficial. Maximum points will be earned on this question for providing discussion presented during lecture and not on information obtained from the internet.

Design patterns may not be beneficial when the complexity it introduces outweighs its benefits, especially in small or simple applications or MVP applications.   
One of the principles for clean code is KISS (Keep It Simple, Stupid) and sometimes using design patterns violates this principle. The goal should be to write clear, maintainable, and understandable code.

With small applications, it might lead to overengineering. For example, applying the Abstract Factory pattern to create only two simple objects might be overkill, introducing unnecessary layers of abstraction that make the code harder to follow. In such cases, a straightforward implementation may be more effective than force-fitting a pattern. Patterns should serve the problem, not the other way around.

**Reference :**

[1] A Checklist on Principles and Strategies(KISS ), lecture

[2] The Downside, An Introduction to Design Patterns

[2] https://www.geeksforgeeks.org/java-design-patterns/