Homework 3

Pop Marc Alexandru

Justifications:

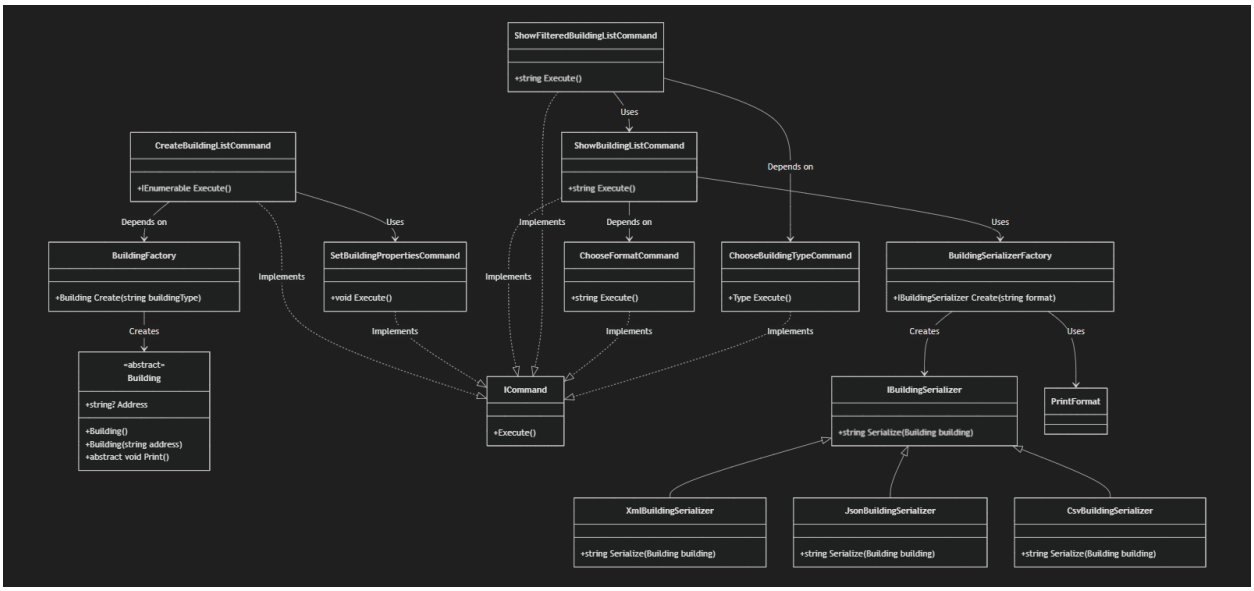
Factory Method pattern:

I chose the Factory Method pattern for this project to decouple the object creation logic from the client code, allowing for easier extension and maintenance. It simplifies the process of creating different types of Building objects based on input, making the system more flexible and scalable. The factory method pattern is suitable here because it provides a single method to create specific types of buildings based on a given type, keeping the logic centralized and straightforward. Abstract Factory would be unnecessary, as it is typically used to create families of related objects, which isn't required for this scenario.

Bridge pattern:

I chose the Bridge design pattern for this implementation because it allows me to create a bridge between an existing interface (IBuildingSerializer) and a new format or system without changing the original code. The Bridge pattern is ideal when you need to integrate a class or method into an existing system, and it enables seamless compatibility between different interfaces. This approach is more suitable than Adapter, which is focused on separating abstraction and implementation, or Decorator, which is used for dynamically adding functionality to objects.

UML Diagram (simplified – derived building types are not included for easier visualization):


ShowFilteredBuildingListCommand
+string Execute()
Uses
CreateBuildingListCommand
+IEnumerable Execute()
ShowBuildingListCommand
+string Execute()
Depends on
Depends on
Uses
Implements
Depends on
Uses
BuildingFactory
+Building Create(string buildingType)
SetBuildingPropertiesCommand
+void Execute()
ChooseFormatCommand
+string Execute()
ChooseBuildingTypeCommand
+Type Execute()
BuildingSerializerFactory
+IBuildingSerializer Create(string format)
Implements
Implements
Creates
Implements
Implements
Implements
Creates
Uses
«abstract»
Building
+string? Address
+Building()
+Building(string address)
+abstract void Print()
ICommand
+Execute()
IBuildingSerializer
+string Serialize(Building building)
PrintFormat
XmlBuildingSerializer
+string Serialize(Building building)
JsonBuildingSerializer
+string Serialize(Building building)
CsvBuildingSerializer
+string Serialize(Building building)