ATG Software Engineering Code Test

You will be given a refactoring exercise based on a C# Visual Studio solution which currently has several maintainability problems, bugs and no unit tests.

# The problem

A LotService class has a GetLot method with the following signature-

**public Lot GetLot (int id, bool isLotArchived)**

The main Lot data store is a PaaS service (which has availability issues in the past), so a failover data store has been created which stores a backup copy of the Lot records. The **GetLot** method retrieves lot and returns them to the caller based on the following logic:

* If the *isLotArchived* parameter is true, retrieve the Lot from the archive data store.
* If the system is in failover mode, try retrieve the Lot from the failover store.
  + The method evaluates if the system should be in failover mode based on a given number of failed requests in each time period (currently 10 minutes).
  + There is a class which maintains number of failed requests and that number should be above 50 failed requests.
  + There is *isFailoverModeEnabled* flag which must be true.
* Otherwise try retrieve the Lot from the main Lot data store.

The response from the both the Failover and main Lot data access may indicate that the lot has been archived. If so, the lot is then retrieved from the archive data store.

# Your Task

We would like you to begin refactoring the GetLot method in the LotService class in order to make the class easier to maintain.  You should assume that this service is part of a larger system.

Following classes representing the 3 data stores to load a lot based on logic described above. Feel free to change anything (method signatures, constructors, etc.): -

1. The ArchivedRepository class.
2. The FailoverLotRepository class.
3. The LotRepository class.

During the refactoring process you should consider the SOLID principles, the readability of the code and where tests might be appropriate.

***The unit test project should use NUnit, and Moq (feel free to use any nuget packages you are familiar with).***

**Task2 :** Summarize your last project in a json document, this could include description, tech stack and tools etc.

# Your solution

You should aim to make the solution more maintainable, applying basic engineering principles such as SOLID, DRY, YAGNI and KISS.

We want to see how you break down the problem and we’re looking for simple, clean, readable code to demonstrate this. Feel free to use the internet to look up anything you need during the code test.

You must prepare a small document to explain your approach and changes you have made as part of your solution.