ReasonLabs C# Home Assignment

Security Software Distribution System

Time Limit: 3 hours

Background

ReasonLabs distributes security software to millions of users worldwide. We need a system to manage software distribution, track installations, and detect potential security threats or license abuse.

Your Task

Build a console application that simulates a simplified version of our distribution management system. The system should handle software licenses, track deployments, and provide analytics for our security team.

Business Requirements

1. License Management

- The system should support creating and managing software licenses
- Each license should have an activation limit and expiration period
- Licenses should be associated with customers
- The system must validate license authenticity and status

2. Deployment Tracking

- Track where and when software is deployed
- Each deployment should be associated with a specific machine
- Support activation and deactivation of deployments
- Prevent unauthorized or excessive deployments

3. Security Monitoring

- Detect and log suspicious activities such as:
  - Attempts to exceed activation limits
  - Usage of expired licenses
  - Unusual deployment patterns
- Provide a way to query security events

4. Analytics & Reporting

Generate reports including:

- Product deployment statistics
- Licenses nearing expiration
- Most active deployment regions
- Security incident summaries

Technical Requirements

- Implement appropriate data models to represent the business domain
- Design a service layer that handles the business logic
- Create a simple console interface for interaction
- Include proper error handling and validation
- Write at least 5 unit tests for critical functionality

Console Application Features

Your application should provide a menu-driven interface with capabilities to:

- Manage licenses (create, view, validate)
- Handle deployments (activate, deactivate, list)
- View analytics and reports
- Monitor security events

Deliverables

1. **Source Code**: Complete C# solution
2. **Tests**: Unit tests for core functionality
3. **Documentation**:
    - README explaining how to run the application
    - Brief explanation of your design decisions
    - Any assumptions you made

Evaluation Criteria

- **Architecture & Design**: How well you structure the solution
- **Code Quality**: Readability, maintainability, and best practices
- **Problem Solving**: How you interpret requirements and handle edge cases
- **Testing**: Coverage of critical paths
- **Security Awareness**: How you handle potential security concerns

Constraints

- Target .NET 6.0 or higher
- Console Application only
- No external databases (use in-memory storage)
- Minimize external dependencies

Bonus Points

- Implement data persistence between application runs
- Add more sophisticated security threat detection
- Include performance optimizations for large datasets
- Implement a command pattern for operations

Note

We intentionally leave implementation details open to assess your ability to make design decisions. Consider:

- What data needs to be stored?
- How should components interact?
- What constitutes suspicious behavior?
- How to make the system extensible?

Good luck!