
Coding Exercise: Secure Windows Application

Estimated Duration: 5 days part-time
Target Role: Senior Windows Application Developer
Output: Secure native Windows desktop + background service app with full installer (MSI)

Project Overview

Design and build a Windows-native application that can publish a request for a google search on https. It will consist of:

1. A persistent background **Windows Service** that manages the secure network connection.

2. A **Desktop UI** for user input, live status monitoring, and basic configuration.

3. An **MSI installer** to package and deploy both components as a cohesive product.

Requirements

1. Windows Service

- Written in **C++ or C#**
- Runs at system startup
- Publish a request for a google search on HTTPS.
- Sends:
    - Hostname
    - IP address
    - Timestamp
- Implements:
    - Retry logic
    - Logging to disk

2. Desktop UI

- UI to:
    - Input server address and port
    - Connect/disconnect trigger
    - View current status (connected, failed, retrying)
    - View real-time logs

- Framework options:
    - WinForms / WPF (C#)
    - MFC (C++)
    - Flutter (bonus if applicable)

- IPC or messaging layer to communicate with service

3. MSI Installer

- Create an installer using:

    - **WiX Toolset**, **InstallShield**, or **Visual Studio Installer Projects**

- Installer must:

    - Deploy both service and desktop app
    - Set service to start on boot
    - Create uninstall entry
    - Store settings in registry or `%PROGRAMDATA%`
    - Include TLS certs (if self-signed)

Security Expectations

- Secure connection using TLS 1.2+

- Validate cert chain (self-signed OK with comment)

- Avoid hardcoded credentials

- Basic input sanitization and error handling

Deliverables

- Full **source code** (GitHub or zip)
- Working **MSI installer**
- Sample self-signed certs (if applicable)
- **README** including:
    - Build instructions
    - Installer usage
    - Architecture and design notes
    - Description of tools used (e.g., MSI generator, libraries)

Evaluation Criteria

Area	Focus
Code Quality	Readable, modular, secure, and well-documented
API Usage	Proper use of Windows networking and system APIs
UI/UX	Functional, clear, responsive
Security	TLS, cert handling, input validation
MSI Packaging	Proper setup, install/uninstall behavior
Reliability	Error handling, retry, logging, service control
Documentation	README clarity, tradeoffs, rationale
Bonus Points	CI workflow, Flutter integration, use of metrics or system tray icon

---

You can deliver this project through Git or bundle up and send it to Centripetal for review.