# Blue Coding .NET Challenge

**Note:** *Please, take your time and read these instructions thoroughly. This project is configured for .NET 6. As always, feel free to use your normal C# IDE/Editor to complete the task.*

## Deliverables

A Github public repository with at least 2 commits:

- Initial commit with the code as originally received.
- Solution to Issues (you can split it into 1 commit per Issue or just 1 for all of them)

**Note:** *Don't mention or reference Blue Coding in any of the links or names*

## Summary

You're working on a monitoring platform for IoT devices. The platform includes an API where devices regularly submit readings from internal sensors. The system evaluates all incoming readings and if a reading or readings are outside the normal range an alert is generated and returned to the device.

You've created an endpoint named `readings/evaluate` in the `ReadingsController` class, that will take the readings from a single device and return any potential alerts.
While testing this endpoint, you discover the following issues that we would like you to review and fix.

## Important Considerations

- Do not modify nor commit changes to the solution `.sln` or project `.csproj` files.
- All existing tests must pass and should not be changed.
- The tests are the source of truth for the minimum expected behavior and expectations for the REST API.
- No new tests need to be created.

- Only out-of-the-box functionality already present in the projects can be used to solve the above issues.
- Use best practices for C# and ASP.NET when writing your code, code quality is important.

# Tasks

## Issue 1

The device should send a special “secret” in the header of the request but something is wrong and the endpoints that rely on this header have their tests failing. Please find the cause and fix the problem so that the related tests pass.

See test: `CorrectStatusCodeBasedOnSecretValidity`

## Issue 2

In addition to sensor readings, the submitted data will also contain the firmware version of the reporting device. This version data must be validated to ensure it follows standard “semantic versioning”. This validation needs to be added.

A C# compatible regex to validate the firmware version (taken from the official semver page) is:

```
^(0|[1-9]\d*)\.(0|[1-9]\d*)\.(0|[1-9]\d*)(?:-((?:0|[1-9]\d*|\d*[a-zA-Z-][0-9a-zA-Z-]*)(?:\.(?:0|[1-9]\d*|\d*[a-zA-Z-][0-9a-zA-Z-]*))*))?(?:\+([0-9a-zA-Z-]+(?:\.[0-9a-zA-Z-]+)*))?$
```

See tests: `SuccessWhenValidFirmware` and `ReturnErrorWhenInvalidFirmware`

## Issue 3

The reading data contains both humidity and temperature values. We have discovered that the appropriate alerts are not being returned when these readings are out of range.

Fixing this will resolve the remaining failing tests.