Note:

- All programs should be written in C#.
- If not specified, technology can be chosen freely.
- The usage of ASP.NET MVC 5 or ASP.NET MVC Core is mandatory!
- Think professional code!

1. Programming exercise

Write a (little) program that shows how you handle the full stack from front-end to back-end. As noted already, ASP.NET MVC 5/ASP.NET MVC Core should be used.

A user should be able to enter a new flight which has a departure airport and a destination airport. This new flight should be persisted into the system so that it can be retrieved later on. Business logic should be able to calculates the distance between the two airports their GPS positions and to calculate the fuel needed for this flight, given the aircraft fuel consumption per distance/flight time + takeoff effort.

The program should be able to list the different flights which are loaded from a persistence medium, technology for this can be chosen freely. It should also be possible to edit a flight and persist these changes.

A report page should be included that shows a summary of all entered data including the calculated data.
Please consider that the code should be production quality and not POC!

Also note that any missing requirements are left for to you provide.

2. How would you rate the quality of the following piece of code? Can you provide an alternative?

```
if (message is MessageA)
{
    var messageA = message as MessageA;
    messageA?.MyCustomMethodOnA();
}
else if (message is MessageB)
{
    var messageB = message as MessageB;
    messageB?.MyCustomMethodOnB();
    messageB?.SomeAdditionalMethodOnB();
}
else if (message is MessageC)
{
    var messageC = message as MessageC;
    messageC?.MyCustomMethodOnC();
}
```

3. Implement a file reading "library" that provides the following functionalities:

Our business wants us to write a file reading implementation and assigns a user story to us. This user story states that "A user should be able to read a text file". Please implement this a simplistic version of this in C# and record this version inside a Git repository, please tag this as version 1 at the end of the implementation.

Some time later our business asks us to extend the library and creates a new user story stating that "A user should be able to read an XML file". Extend your implementation so that this behaviour is possible, record the changes again in the Git repository and tag them as version 2 at the end of the implementation.

Again some time later our business asks us to change the library so that they are able to read encrypted TEXT files. They create a user story stating that "A user should be able to read an encrypted TEXT file". Our user story contains some more detail stating that the user will tell the system that it needs to read an encrypted TEXT file. Please note that the encryption algorithm is of no concern and can be for example a simple reverse of the text. Also note that we should be able to switch the encryption without actually changing the code. Implement your changes and record them again in the Git repository tagging them as version 3 at the end of the implementation.

Later in time the business requires us to again change the library an extend it with role based security for XML files. A user story is assigned stating that "A user should be able to read XML files in role based security context". Some examples are stated such as "eg. admin can read everything, other roles can only read limited set of files" and us such we're not interested in the actual role based security system. As such you can provide a simplistic implementation, however make sure that the switch to a real role based security system should be possible without actually changing the code! Implement your changes and record them again in the Git repository tagging them as version 4 at the end of the implementation.

At some point in time the business asks us to enable the encrypted reading feature also for XML files. Implement your changes and record them again in the Git repository tagging them as version 5 at the end of the implementation.

Even some more time later the business asks us to enable the role based security reading feature also for TEXT files. Implement your changes and record them again in the Git repository tagging them as version 6 at the end of the implementation.

As a last change the business asks us to also add the ability to read JSON files. They create 3 user stories: "A user should be able to read JSON files" "A user should be able to read encrypted JSON files" "A user should be able to read JSON files in role based

security context" Implement your changes for each feature separatly and record them again in the Git repository tagging each version as a new version 7, 8 & 9 at the end of the implementation of each feature.

BONUS: Implement a simple GUI/CLI application allowing to specify which file type you want to read, specify to use the encryption system and specify if role based security is needed or not. As such I can start the application, it will ask me the file type, to use the encryption system and if role based security should be used or not. If role based security is used it will ask my role and then we will read the file and show the output. After this I can read another file in another way without needing to restart the application.