# Performance Approach

The evaluation of the performance of the project is paramount due to being able to look at how the project can be added to, maintained and also how it runs if it were an actual program being used. This is done by looking how fast it compiles and how easy it is to be taken elsewhere and still ran and how much memory it takes up when loading.

## Reliability

The fact that the system will always complete its required job regardless proves that it is reliable, however, as stated later on in this evaluation, it does require the specified files for the program to work properly as it doesn’t create its own XML files. However, as the code is modular in its programming style, it can edited and still work in full, for when a part of the program is under construction.

## Security

The security of the program is easily compromised. It holds user data as static strings in XML files that can be easily opened and viewed using file viewers, including passwords and email combinations. If this was to become a release it would have to be hashed and then decrypted when passwords are entered. This is also an issue due to the fact sensitive plane data is also set up as static information so can also be opened, which could compromise the planes and also be a threat to the passengers safety.

## Portability

The way the program is set up is it users various XML files that are scattered in the source directory, meaning that if this was moved around different machines, it would have to have all the files added to the same directory otherwise it wouldn’t input correctly. It wouldn’t let the user know, either, when a file cannot be found and would just stop working, which would take away from the user experience.

To improve this, we would have to add a directory that holds all the user information and the XML files, then this can moved around with the release rather than having to add everything later on. Or, another way of getting around this issue, is to be able to create the XML file if it doesn’t exist previously.

## Maintainability

The program can be looked after constantly using the very easily viewable, split up functions and then code can be added and changed. Due to the way the menu is set up, when a menu item is being worked on it can simply be taken out of the menu and not change how the program functions. Obviously, it wouldn’t be able to be inputted either. This is a slow way of maintaining a program but it is effective, although basic.

One of the only big issues with maintaining the program is the large size of it, as it would need someone who knows what they’re doing to change the program, as someone who didn’t and didn’t research the program enough could cause fatal errors to the system.

## Scalability

The way the code is set up allows the program to be easily added to later on in development, as each function is called from the menu and then the functions are kept separately. This also means that it can be synchronised when multiple people are adding to the code and Github commits can be completed more successfully. Also, due tot eh fact it is a command line based program, it is able to be added to slightly easier as new graphical interfaces don’t have to be made and instead one can just create a new function and add a text based number code that locates it in the menu.

Another aspect of how scalable the program is being the fact it can work with only a few flights and plane information and also millions of the same data. How efficient the program is could be tester however due to the data structure of SQL it would allow millions of data entries and still be as easy to read from, although it might take a little longer to search through the file. This would be necessary for the system due to the fact it’s an airport system that would contain lots of flight codes and passenger information.