Damarel Placement Report

James Fernando

22nd January 2019

Contents

1 Introduction

In this report, I am going to first explain a bit about the company and my position in it before going over some of the projects I have worked on over the year and go through what I have learnt and what I can take away from those projects. And, then I will go into what skills I have gained and how I have developed over the year.

2 Company Background

Damarel Systems International were established in 1988 and are a UK based software house specialising in airport operations software. They supply a number of software packages to airports and airlines all over the world. I was mainly working on the FiNDnet Billing Application. This system allows for airports to bill the airlines for services provided to an airline's plane based on a contract with the airline.

3 Placement Role

My Job title while working at the company was Student Placement Developer. My main purpose was to use knowledge from university to develop real projects, learn about modern technologies and real-world team working practices. My tasks are to design, develop and support software amongst other small activities which were in the company's interest. As I was assigned to the FiNDnet Billing Development team all of my work was on the Billing application or on other items to help in the development of the Billing application.

4 First Weeks Task's

Read Administrator Handbook

This involved reading the administrator handbook whilst playing around with a copy of the billing software to get familiar with how it worked and what all the different functions are. As although you may expect that there isn't much to the billing application I can assure you it's far bigger and more complicated than most would expect.

C# Crash Course

My next task was to learn about some C# features I heavily used Microsoft's C# Guide[1] to learn about how to use the various features available in C#. I used this guide to learn about generics, event handlers, LINQ statements and lambda expressions.

MVVM

MVVM stands for Model View View Model. This is a design pattern for creating GUI applications. It is based on the idea of separating the display code from the logic and data. This works well in graphics packages which use XAML for the display code, for example, WPF[1] and Android Development. As this allows for all of the display code to

[1]WPF: Stands for Windows Presentation Forms is the latest display package from Microsoft to use with C#.

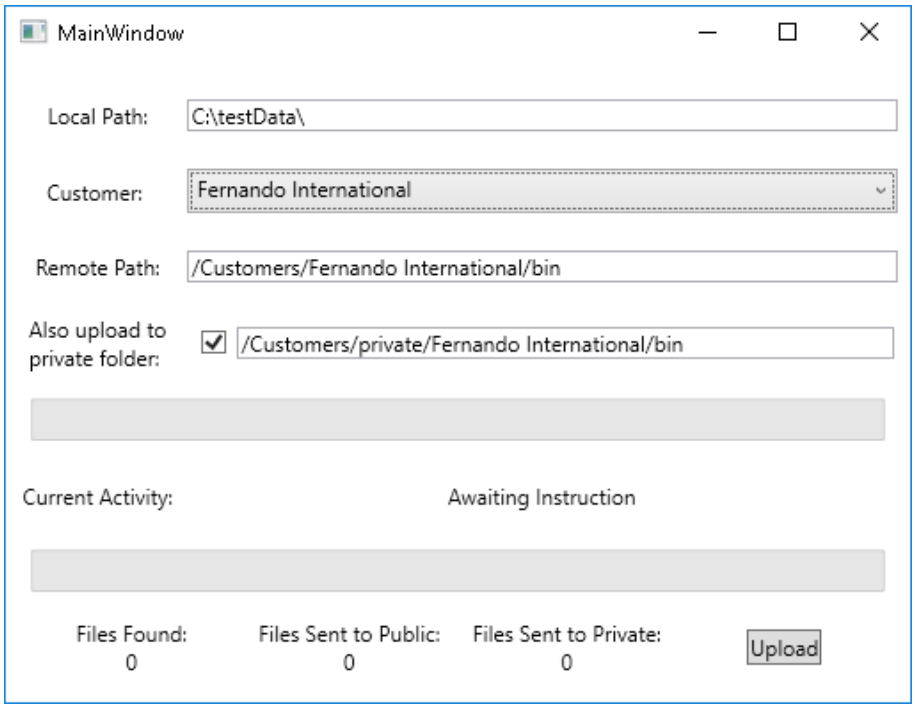

Figure 1: A Screenshot of the FTP Uploader application

be written in a different language meaning the separation of display code and the logic code is naturally enforced. It also allows for the logic to be unit tested as it should not contain any display/graphics code.

Design Patterns

I also had a look at some of the design patterns listed on the website Source Making - Design Patterns [2] Although at the time they didn't make much sense over the year I was able to understand and use the abstract factory, adapter and composite patterns in my development.

Unit Testing Frameworks

I also had a brief look at some unit testing frameworks such as Moq and Autofixture. Moq allows you to implement and set-up interfaces without having to define an entire class. And, Autofixture is an Object constructor library which allows for objects to be constructed using random values.

5 Ftp Uploader

My first task was to create an FTP Uploader, this was to be used internally as a tool to upload releases of the application to an FTP server where it could then be accessed by the customers. The requirements for this tool were to:

- Have a GUI for easy use
- Manage options for different releases

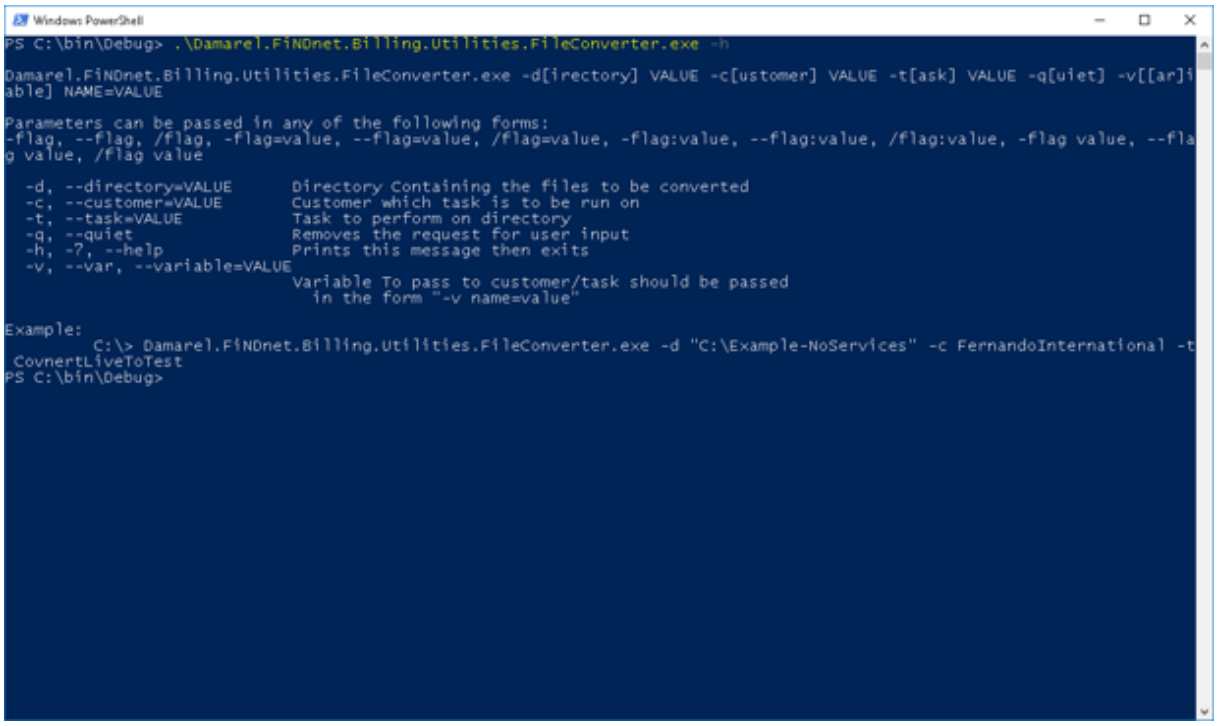

Figure 2: A Screenshot of the File Editor console application

- Have an option to also upload the release to a private folder
- Automatically have values set based on defaults

The FTP Uploader was a good introduction into developing an MVVM application as it allowed me to get to grips with how the model works without too much complexity added from the application logic. I was then able to use this base knowledge about MVVM and WPF applications to develop GUI programs later on. This project still needs some work on it before it can be used. I have had to clean up the ticket for this project and add some more information on it so that it can be picked up by someone else after I leave.

6 File Editor

The File editor was my next project and is a console program where the initial scope for the application was to convert some configuration files from live to test values. However since then, it has become a complete file conversion application so that it is possible to set a customer and conversion type, and it will manage the conversions for those settings. It can also be developed so that it can be expanded to deal with any file type and different conversion types. As an example, it has been expanded by one of my colleagues and it is now used as a part of the build process.

7 About Box

Next, I had to create a new WPF About box to replace the old about box written in Centura. The aims of this project were to provide more debugging information about the environment set up to our customers and support team. This was developed using

Figure 3: A Screenshot of the About box window

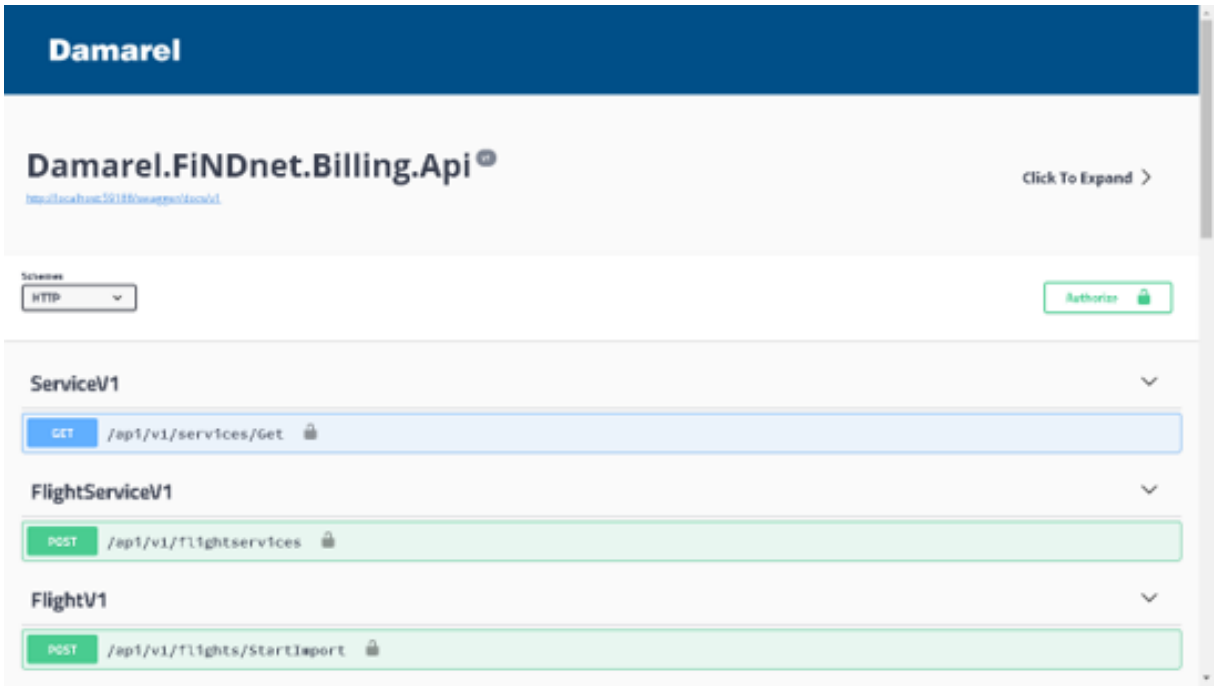

Figure 4: A Screenshot of the Swagger UI

MVVM. Meaning I had to learn a lot of WPF mainly about how the XAML works and the different features which were available in it, for example, I had to use triggers to change the display format depending on the value of a property in the view model. This project was completed and included in the Billing release.

8 Swagger

Before going on to describe the work I completed using Swagger, Firstly I shall explain what it is and how it can be used. Swagger is an API Description tool it allows for information about the endpoints of the API to be described. For example, it can state the formats of the inputs and outputs of the endpoint. The Swagger document is the file which contains all of the information on the API and is stored in the JSON format. This JSON file can then be parsed by a number of different programs to create interfaces which allow you to use/test the API. Alex my mentor set-up the Swashbuckle package in the API project. Swashbuckle is a tool which can generate a swagger document using the XML comments and some attributes from a C# Web Project. Also, included in the swashbuckle release is the Swagger UI package. Swagger UI is a website which generates a testing ground for an API given a Swagger document input. Swagger have a test page which can be found at `http://petstore.swagger.io/` which can better demonstrate how it can be used. My job was to make some modifications to the Swagger document created by Swashbuckle and also to make some modifications to the Swagger UI to give it more of a Damarel theme. I learn quite a bit working on this project like how open source pieces of software are used in commercial environments. Also, I did find a bug in the Swagger UI while working with it, therefore, I needed to submit a bug report to get it fixed. This introduced me to how Github works with their issues system.

9 Unit Testing

As one of my tasks, I was asked to implement some unit tests for the API project this was mainly to focus on the validation and population logic.

Moq

While working on the unit tests I needed to learn how to use Moq this would enable me to test some of the existing API code. Moq is a .NET library which allows you to implement interfaces without having to define an entire class. This makes it very useful in Unit testing as it enables you to construct the input classes for a method/class without having to go through the set-up process for that input. This also taught me the importance of using Interfaces where possible when programming as it allows for easier unit testing. Also, if for example, you are writing a method it means you can abstract the function of your method away from its input meaning that it can be used in other places and with other classes.

dotCover

I also used dotCover[3] to produce a coverage report before and after implementing the unit tests, although this was useful in showing my superiors the work which was carried out. It did clearly show that some parts of the code were not covered. Although you could argue that this is a valid issue to have, I feel and, I believe that my superiors agreed that this does distract you from another possible problem which is if the code which is covered is tested properly.

Demonstration

Before finishing the placement year I was asked to perform a demonstration to the development team about the best practices for Unit testing and what would be the house style going forward. I also produced a document and emailed it round to everyone so that they have some reference if they need to check what the best practices are in the future.

10 Turn Report Generator

The turn report generator was one of my main projects over the year and I spend a lot of time researching and developing the project. A Turn in my sense was the inbound and outbound flight for an aircraft and all of the services carried out on that aircraft from the airport's point of view. In the end, I didn't have enough time to complete it, however, I was able to start it and hopefully provide enough guidance and knowledge to Jonathan(The following replacement student) to finish the project. The turn report generator part of the billing application needed to be able to use a number of different templates depending on the which customer the turn report was for and what type of invoice it was. And then given this it needed to produce a PDF of the turn report. Originally the template was supposed to be created by a designer program produced by Damarel. However to cut the development time we decided that we should make and release the generator first and then make and release the designer. When we release the generator then we would also provide some templates created by hand so that the customer will still be able to use the generator.

Template and Template Designer

Template The Template was supposed to store information on what will be displayed in the turn report and how it would be displayed.

Designer The designer would then allow our customers to build the templates. In the research carried out for the designer I needed to work out how I would create this, it was requested that the designer should be of a what you see is what you get style, therefore I created a drag-drop system which would allow users to drag in the columns they wanted to be displayed on the report and drag to re-arrange them. This would make it as simple as possible for our customers to use. They would then have more options on the columns when they are dragged into the document. This drag-drop system was very complicated and a number of times I felt it was not possible to complete it. However, in the end, I did produce a proof of concept piece of software which was able to handle the desired drag-drop functionality, this did teach me that even if I think that something is not possible or more specifically it's not possible to do it cleanly and efficiently I need to look into other ways of completing the task at hand. I do feel that from a company viewpoint that all of the time and effort working on the drag-drop was a waste as a different implementation could have been carried out a lot quicker and that would not be much more difficult for the customers to understand. However, from a personal point of view, the learning experience is invaluable.

Storage Some parts of the templates needed to be stored in the application database, this meant that I had to design and implement some scripts to add the table/columns to an existing database and to the create scripts for the database. This meant that I had to learn a lot about the Oracle database system which is annoying because if you wish to make any small changes to a database you need to understand how all of it works and it's is quite different from MySQL which is what I had known previously. Although I did learn that in the industry they aren't as strict to the forms of normalisation as we are in university as my superiors were happy with me giving indexes to rows/items with unique keys in them. Now looking back at this design is very useful especially in the very recent case of the EU's GDPR law as this stops you from using personal customer data as a primary key as it may be deleted at the request of the customer. As we stored some information in the database this meant we had to implement a data access repository for the template. The data access repository allows us to simply call a few commands to store and retrieve the templates from the database when we give the repository a connection object.

Turn Report Generator

When it comes to generating the template we need to make sure that it is generated as a PDF then displayed and printed alongside the Invoice.

PDF Generation When it came to producing the report as a PDF I was asked to do some research in the PDF packages I could use. I was given some restrictions mainly it had to be free/open source as we already had crystal reports which the customers were paying for. And, this meant we could not justify charging them for another PDF generation tool. I, therefore, did some research into free/open source PDF generation tools although this lead me down the wrong path because although a piece of software is open source that

does not allow you to use it as you wish, this is because there are some terms in the licence saying that if you use this software in your own software your software needs to be open source which is a problem in a commercial environment. After this, I learnt a lot more about software licences and what I can and can't use. While producing the Interface for the PDF generator we (Jonathan and I) had to make sure that we created so that it had enough functionality(or the functionally could be added) so that it could be used for the invoice generator as well.

Integration into current application The turn report generation needed to be integrated into the billing application so that when a user clicked to view an invoice they would also be able to see the turn report as well. I conducted a small amount of research into how to integrate it and produced a couple of mock-ups on how it could look once it was all integrated.

11 Invoice Generator

The invoice generator project is the larger project where the turn report generator came from. This project was for me to replace the current invoice generation which utilises crystal reports and implement my own invoice generation using another PDF generation tool.

Current Invoice Generation

Currently, we are using Crystal Reports to generate the invoices and reports. This is a third party report generation tools. Some expertise is needed to create a Crystal Report templates, therefore, these templates need to be created by our development team. This takes up valuable development time for trivial tasks like moving an address field or changing the contact person name. Therefore with the use of a designer the customers themselves would be able to carry out these tasks.

Invoice Research

My first step was to conduct some research into what the invoice generation needed to support, this involved looking at some invoice templates online and some of the current crystal reports templates. This allowed me to find what features were required by the PDF generation tool e.g. Tables, Unicode etc.

Invoice Design

The designer for the invoice was to follow the same principal as the Turn report designer and would allow for the dragging and dropping of elements on the invoice page however this would be more complex as some of the elements on the page would need to be positioned absolutely on the page, for example, so that addresses could be lined up with the envelope windows. Again these would need to be stored as templates.

Invoice Generation

This would need to be done using the same generator as used in the Turn report generator which is why research on the requirements for the invoice was completed before choosing

the PDF generator for the Turn report generator. While I and Jonathan were developing the PDF generator for the turn report generator we made sure that the PDF generation interface we created would be able to also handle the Invoice generation so that the code could be reused.

12 Skills Gained

I can comfortably say that I have gained far more skills than I expected to over the placement year. I also have found that I appear to work better in a work environment compared to an academic one. I feel that the ability to focus on one or two things at a time suites how my brain works a bit better than focusing on 4 modules at a time.

12.1 Programming Skills

Over the past year, I have developed my programming skills a tremendous amount and not just in language specific skills but overall programming skills which can be transferred between languages.

Programming Principals

I've learnt a lot about programming principals mainly from my mentor Alex and technical lead Gihan. I have learnt a bit about design patterns like some of those listed at `https://sourcemaking.com/design_patterns`[2]. Unsurprisingly I learnt quite a bit about creating code with unit testing in mind when I was going through the API project unit testing that. The main idea is to use interfaces where possible so that inputs can be mocked.

Programming Objectives

I've mainly learnt about what the programming objectives in a commercial environment are like. These focus more on having readable maintainable code rather than focusing on the speed of the code. There is also more of a focus on making code understandable rather than adding comments to confusing code. Although I always understood the benefit of reusing code. I've learnt of the importance of being able to abstract problems to their individual parts and then what is the best way of mapping that to code.

Language Specific skills

I've learnt a lot of C# skills over the year such as:

- Event Handlers
- Generics
- Tasks and Threads
- LINQ statements
- Reflection
- Lambda Functions/Delegates

Although I have learnt these in C# it shouldn't be too difficult to transfer over these skills to Java or any other language if I need to.

12.2 Development Skills

I've learnt quite a bit about the development process and how I can use review and QA cycles to verify code and reduce the number of bugs in it. I can also take this into my own personal development by leaving the code for a week before I review it myself. I've also learnt quite a bit about open source software development while looking at the various pieces of open source software we use. Over the year we have been using SVN for the development of the billing application and I feel that that is a very good step before learning how to use GIT. As I had tried to use GIT before but didn't really get anywhere. But SVN allows you to learn how to branch and the basics of merging and committing without the complexity of staging and pushing etc. This has meant I was able to develop this document using GIT.

12.3 Communication skills

I've learnt a lot about communication skills. Surprisingly I learnt a lot on my last day which I wasn't expecting as I had to go round the entire office and say goodbye to everyone. Other than that, I've learnt how to communicate and some not really important things such as email etiquette etc. I have mainly gained a lot of confidence over the year, for example, I felt that if you asked me to give a presentation on some programming aspect to a group of experienced developers I would have never imagined it would happen.

12.4 Time Management Skills

I've had to do some work on my time management although this was for a learning purpose as any deadlines were self-imposed internal deadlines which didn't really hold any weight. This was useful for me personally as it reduced stress. Although meant that my main focus for time management was on getting maximum efficiency out of mine and everyone else's time. We managed our projects using Target process[4] this allowed us to manage the review cycles and what projects were currently being worked on. I mainly had to focus on getting my projects into a review ready state so that they could then be reviewed by one of the other developers. Target Process[4] was useful in allowing us to split up work into tickets and then put each ticket through the review process. Which allowed us to review tickets as smaller chunks of code. I feel that this is something I can use while at University to allow me to better manage my projects and keep a track of what needs doing.

12.5 Planning Skills

Although a lot of planning and proof of concept was carried out on the Turn report generator and the invoice generator projects I feel that the amount of planning carried out for my other projects were about right as it was enough planning to break down the project into manageable steps without wasting time attempting to create a structure for something without actually creating it. I feel if possible I'll bring this into my own development, however, I don't think I can use it for university work as that usually requires a lot more planning than what was done while on placement.

13 Conclusion

In Conclusion, I feel that the placement year was a success for me as I learnt a lot not only about programming skills but also in other skills like time and project management skills. I feel that these will help me to be able to manage my work at University better and some of the other skills may be able to help me improve, Such as the confidence I have gained I hope this can lead me to question things I don't understand. And, I expect my programming skills will be able to improve my work. The main things I take away from the placement year is that I have now produced a few pieces of software which are actually being used commercially in addition to the various other skills I have gained.

References

[1] M. Wenzel, B. Wagner, L. Latham, R. Petrusha, M. Hoffman, M. B, and J. Beaudoin. (May 2017). C# programming guide, [Online]. Available: `https://docs.microsoft.com/en-gb/dotnet/csharp/programming-guide/`.

[2] A. Shvets, G. Frey, and M. Pavlova. (Feb. 2007). Design patterns, [Online]. Available: `https://sourcemaking.com/design_patterns`.

[3] JetBrains, *Dotcover*, version 2018.1, Apr. 2018. [Online]. Available: `https://www.jetbrains.com/dotcover/`.

[4] TargetProcess, *Target process*, Oct. 2017. [Online]. Available: `https://www.targetprocess.com/`.