# Acknowledgement

I would like to thank my DDACC Lecturer Dr. Kalai for providing us with the knowledge to use the Microsoft Azure Services as well as his guidance throughout the entire semester for the DDAC module and this project. Through his tutorials I was able to understand how to use of Microsoft Azure services and how it is beneficial in the growing age of cloud computing. Besides that I would like to thank my friends, Teoh Ghee Yuen and Bryton Hong for their support and guidance throughout the development of this system as well as others. Last but not least to my family, who has been very supportive throughout this assignment.

# Table of Contents

Contents

[Acknowledgement 1](#_Toc498938739)

[Table of Contents 2](#_Toc498938740)

[1.0 Introduction 4](#_Toc498938741)

[1.1 Background 4](#_Toc498938742)

[1.2 Aims 4](#_Toc498938743)

[1.3 Objectives 4](#_Toc498938744)

[1.4 Deliverables 4](#_Toc498938745)

[2.0 Project Plan 5](#_Toc498938746)

[3.0 Design 6](#_Toc498938747)

[3.1 Architectural Diagram 6](#_Toc498938748)

[3.2 Design Considerations 7](#_Toc498938749)

[3.3 Modelling 7](#_Toc498938750)

[3.3.1 Use Case Diagram 7](#_Toc498938751)

[3.3.2 Sequence Diagram 8](#_Toc498938752)

[3.3.3 Data Modelling 13](#_Toc498938753)

[3.3 Sitemap 14](#_Toc498938754)

[3.4 Web Application User Interface 15](#_Toc498938755)

[3.4.1 Home Interface 15](#_Toc498938756)

[3.4.2 Register Interface 15](#_Toc498938757)

[3.4.3 Login Interface 16](#_Toc498938758)

[3.4.4 Home Interface when Login into Account 16](#_Toc498938759)

[3.4.5 Search Flight Interface 17](#_Toc498938760)

[3.4.6 Search Flight Results Interface 17](#_Toc498938761)

[3.4.7 Flight Booking Confirmation Interface 17](#_Toc498938762)

[3.4.8 Booking Confirmation Interface 17](#_Toc498938763)

[3.4.9 About Interface 18](#_Toc498938764)

[3.4.10 Contact Interface 18](#_Toc498938765)

[3.4.11. Profile Interface 18](#_Toc498938766)

[3.4.12 View Flight Booking Interface 18](#_Toc498938767)

[4.0 Implementation 19](#_Toc498938768)

[4.1 Application Development 19](#_Toc498938769)

[4.2 Azure Publishing 21](#_Toc498938770)

[4.2.1 SQL Database 21](#_Toc498938771)

[4.2.2 Web Application publishing 22](#_Toc498938772)

[4.2.3 Azure Traffic Manager 24](#_Toc498938773)

[4.3 Application Scaling 25](#_Toc498938774)

[4.4 Investigate and Analyse Application 27](#_Toc498938775)

[4.5 Discussion of Managed Database 30](#_Toc498938776)

[5.0 Test Plan 32](#_Toc498938777)

[5.1 Unit Testing 32](#_Toc498938778)

[5.1.1 Unit Testing for Registration 32](#_Toc498938779)

[5.1.2 Unit Testing for Login 32](#_Toc498938780)

[5.1.3 Unit Testing for Search Flights 33](#_Toc498938781)

[5.1.4 Unit Testing for Booking Flights 33](#_Toc498938782)

[5.1.5 Unit Testing for Viewing Booked Flights 33](#_Toc498938783)

[5.2 Performance Testing 34](#_Toc498938784)

[6.0 Conclusion 36](#_Toc498938785)

[7.0 References 37](#_Toc498938786)

[8.0 Appendix 39](#_Toc498938787)

[8.1 Stream Video Presentation 39](#_Toc498938788)

[8.2 GitHub Repository 39](#_Toc498938789)

# 1.0 Introduction

## 1.1 Background

Ukraine International Airlines (UIA) is the flagship carrier and largest airline in Ukraine. It operates domestic and international passenger flights and cargo services to Europe, the Middle East, the United States, and Asia.

The airline is eager to expand into new markets, but its current website prevented it from adequately serving customers beyond Ukraine. The Ukraine International Airlines (UIA), is looking into the development of an Online Flight Booking System. UIA looked at both Microsoft Azure and Amazon Web Services and chose Azure. The development of the Ukraine International Airlines (UIA) will be developed and deployed onto the Azure.

The Ukraine International Airlines (UIA), will allow the users to register and login to use the Ukraine International Airlines (UIA) on the Online Flight Booking System. Using the Azure services it will help the Ukraine International Airlines (UIA) Online Flight Booking System it will be beneficial in many aspects as it will be explained through this documentation.

## 1.2 Aims

To provide an Online Flight Booking System that uses Azure services that can help increase the efficiency in flight booking for the Ukraine International Airlines (UIA).

## 1.3 Objectives

The objectives for the Ukraine International Airlines (UIA) Flight Booking System is:

1. To be able to host the Ukraine International Airlines (UIA) Online Flight Booking System on cloud.
2. The book available flights provided by the Ukraine International Airlines (UIA).
3. The ease flight bookings through an online website.

## 1.4 Deliverables

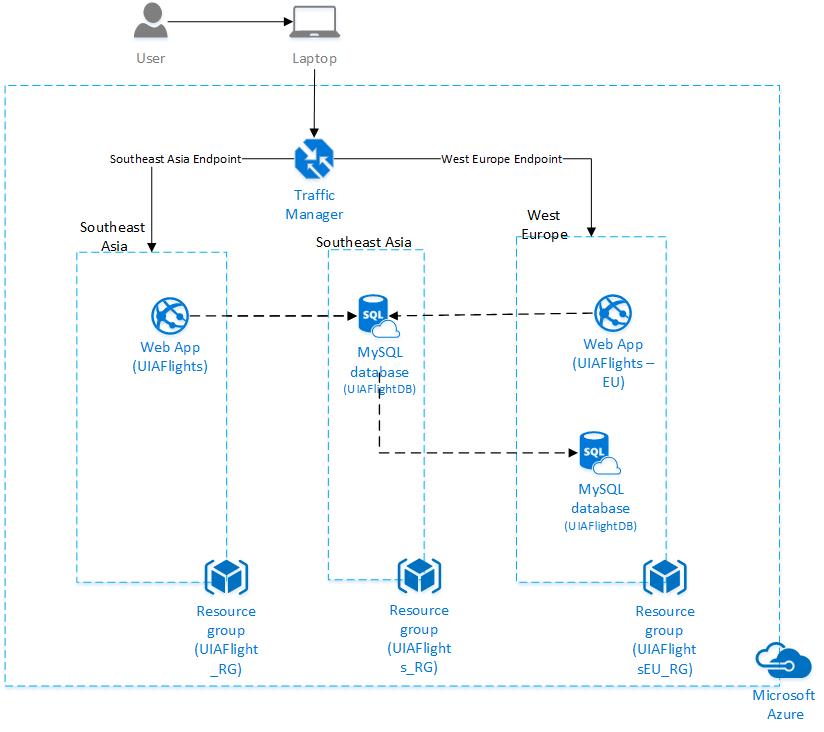
1. Allows the users to register and login into the Ukraine International Airlines (UIA) Online Flight Booking System.
2. Allows the users to search for Flights of the Ukraine International Airlines (UIA) Online Flight Booking System.
3. Allows the users to book and view the flights booked on the Ukraine International Airlines (UIA) Online Flight Booking System.

# 2.0 Project Plan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| WBS | Task Name | Duration | Start | Finish |
| **1** | **Ukraine International Airlines (UIA) Online Flight Booking System** | **90 days** | **Mon 7/17/17** | **Fri 11/17/17** |
| **1.1** | **Introduction** | **5 days** | **Mon 7/17/17** | **Fri 7/21/17** |
| 1.1.1 | Background | 1 day | Mon 7/17/17 | Mon 7/17/17 |
| 1.1.2 | Aims And Objectives | 1 day | Tue 7/18/17 | Tue 7/18/17 |
| 1.1.3 | Deliverables | 3 days | Wed 7/19/17 | Fri 7/21/17 |
| **1.2** | **Design** | **14 days** | **Mon 7/24/17** | **Thu 8/10/17** |
| 1.2.1 | Architectural Diagram | 2 days | Mon 7/24/17 | Tue 7/25/17 |
| 1.2.2 | Design Considerations | 2 days | Wed 7/26/17 | Thu 7/27/17 |
| **1.2.3** | **Modelling** | **7 days** | **Fri 7/28/17** | **Mon 8/7/17** |
| 1.2.3.1 | Use Case Diagram | 2 days | Fri 7/28/17 | Mon 7/31/17 |
| 1.2.3.2 | Sequence Diagram | 2 days | Tue 8/1/17 | Wed 8/2/17 |
| 1.2.3.3 | Data Modelling | 2 days | Thu 8/3/17 | Fri 8/4/17 |
| 1.2.3.4 | SiteMap | 1 day | Mon 8/7/17 | Mon 8/7/17 |
| 1.2.4 | Web Application User Interface | 3 days | Tue 8/8/17 | Thu 8/10/17 |
| **1.3** | **Implementation** | **41 days** | **Fri 8/11/17** | **Fri 10/6/17** |
| 1.3.1 | Local Web Application Development | 31 days | Fri 8/11/17 | Fri 9/22/17 |
| 1.3.2 | Publishing Application to Azure | 1 day | Mon 9/25/17 | Mon 9/25/17 |
| 1.3.3 | Implement SQL Database | 3 days | Tue 9/26/17 | Thu 9/28/17 |
| 1.3.4 | Application Scaling | 2 days | Fri 9/29/17 | Mon 10/2/17 |
| 1.3.5 | Investigate and Analyse Application | 4 days | Tue 10/3/17 | Fri 10/6/17 |
| **1.4** | **Test Plan** | **13 days** | **Mon 10/9/17** | **Wed 10/25/17** |
| 1.4.1 | Test Plan | 5 days | Mon 10/9/17 | Fri 10/13/17 |
| 1.4.2 | Unit Testing | 4 days | Mon 10/16/17 | Thu 10/19/17 |
| 1.4.3 | Performance Testing | 4 days | Fri 10/20/17 | Wed 10/25/17 |
| **1.5** | **Documentation** | **17 days** | **Thu 10/26/17** | **Fri 11/17/17** |
| 1.5.1 | Research | 5 days | Thu 10/26/17 | Wed 11/1/17 |
| 1.5.2 | Documentation Compilation | 5 days | Thu 11/2/17 | Wed 11/8/17 |
| 1.5.3 | Video Recording | 7 days | Thu 11/9/17 | Fri 11/17/17 |

# 3.0 Design

## 3.1 Architectural Diagram



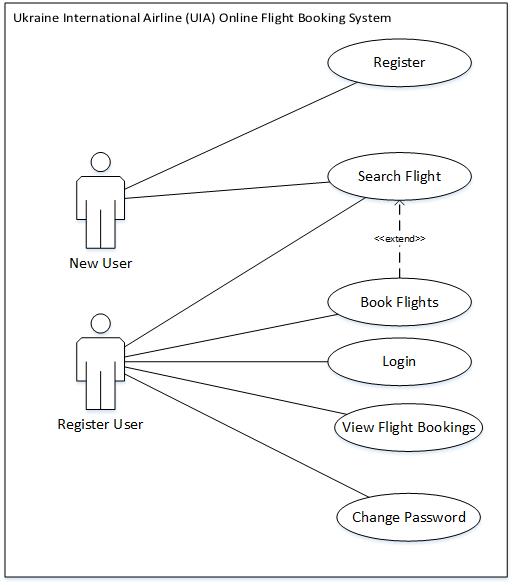
Above shows the cloud architectural diagram for the UIA Online Flight Booking System. There will be 2 web application services that will be deployed for two different locations which are Southeast Asia and West Europe and the traffic manager will handle the request based on the user location. The backup database will be placed in the West Europe resource group in case the primary database in the Southeast Asia region.

## 3.2 Design Considerations

Before the development of the Ukraine International Airlines (UIA) Online Flight Booking System, the requirements of the system has to be detailed including the assumptions and considerations for the system. Furthermore, Ukraine International Airlines (UIA) will be developing their system on Azure, which can allow them to deploy it to different regions and provide many Azure functionalities that will be beneficial for the system. However, the development team was only provided a RM150 resource budget to be used on Azure for the current development of the system and is required to work with it. Therefore, during the designing of the system when selecting the Azure architecture the right pricing tier has to be selected to ensure that the Web Application will be able to run efficiently.

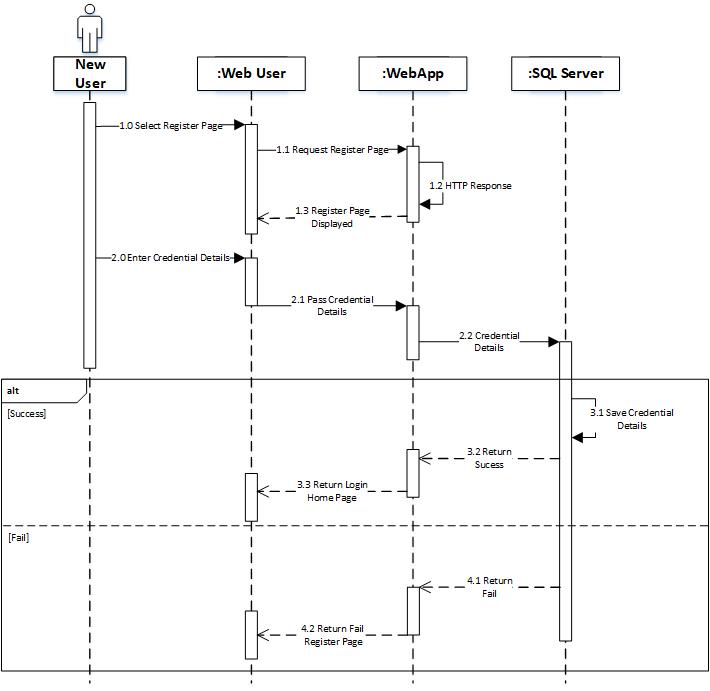
## 3.3 Modelling

### 3.3.1 Use Case Diagram

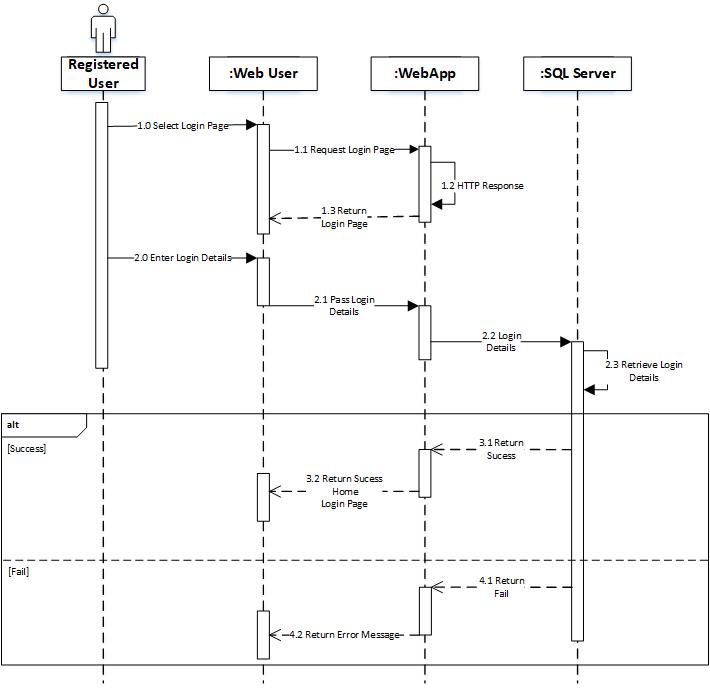


### 3.3.2 Sequence Diagram

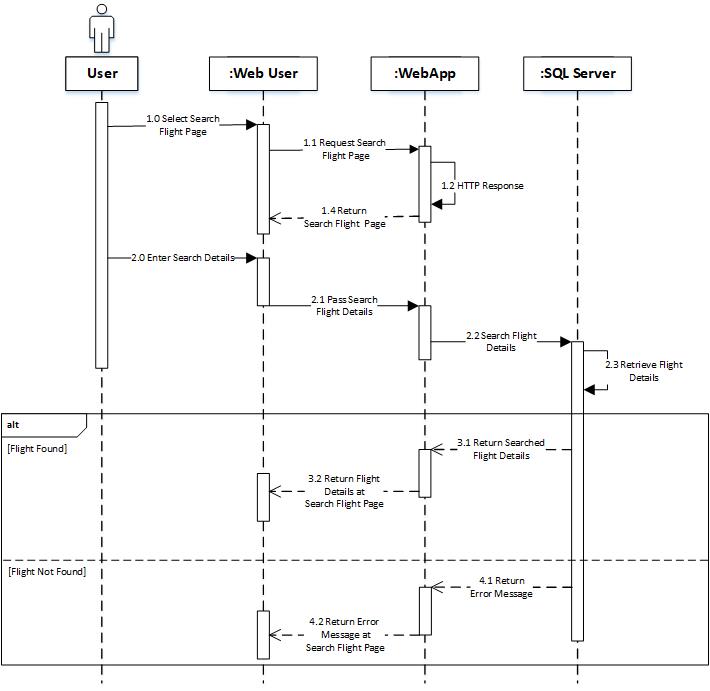
#### 3.3.2.1 Register Sequence Diagram



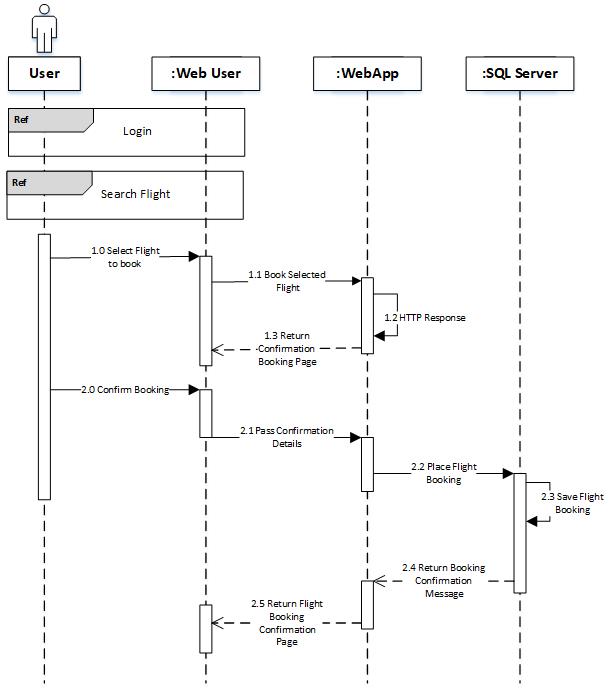
#### 3.3.2.2 Login Sequence Diagram



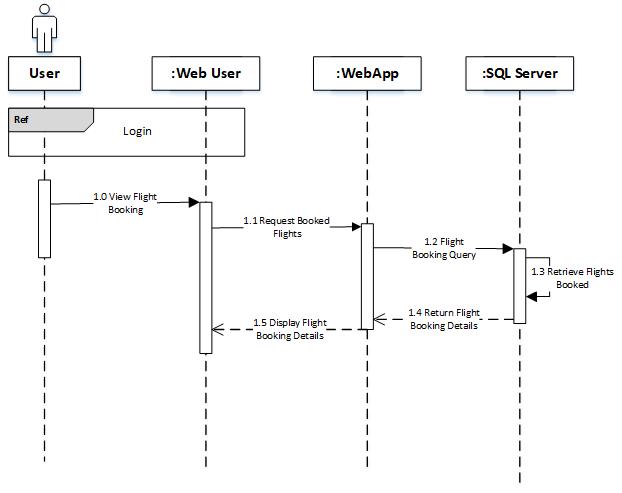
#### 3.3.2.3 Search Flight Sequence Diagram



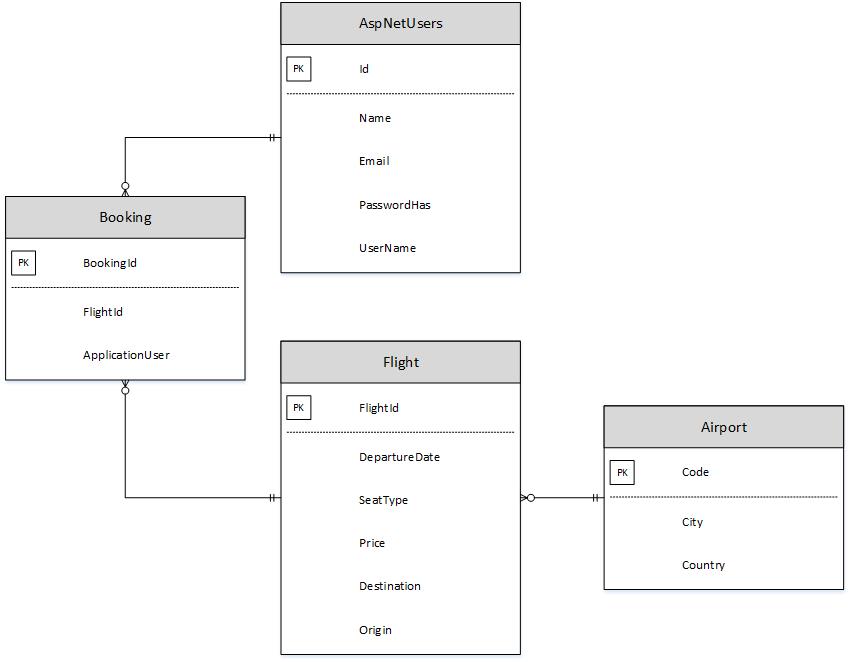
#### 3.3.2.4 Book Flight Sequence Diagram



#### 3.3.2.5 View Flight Booked Sequence Diagram

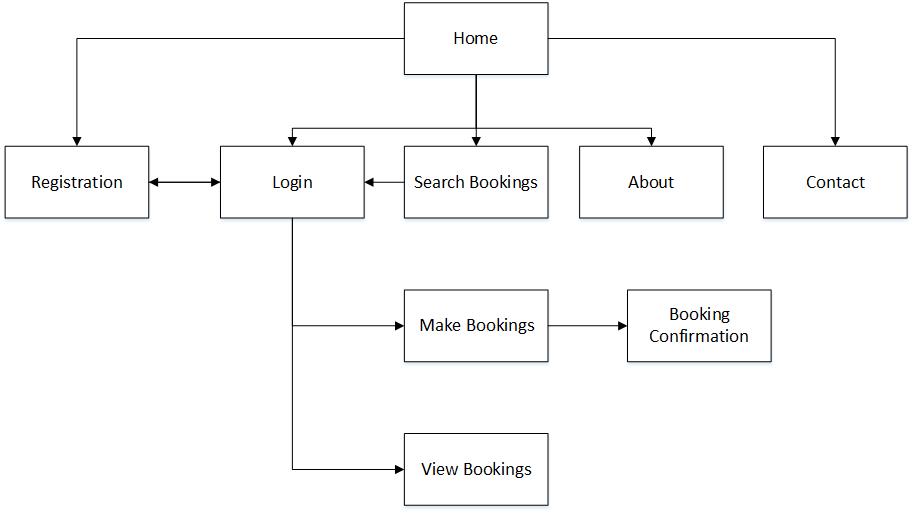


### 3.3.3 Data Modelling



The Entity Relationship Diagram (ERD) above describes the relationship of the databased used for the UIA Online Flight Booking System. These are the 4 database that will be required for this system. The Airport database stores all the codes, city and country of the different airports which will be referenced by the Flight database when displaying the flights of the Ukraine International Airlines. The AspNetUsers database stores all the information of the users that registers inside the UIA Online Flight Booking System. The users will be required to register to make the bookings, in which the Booking database will store the flights that the registered user makes and will save the flight identification number and the user details.

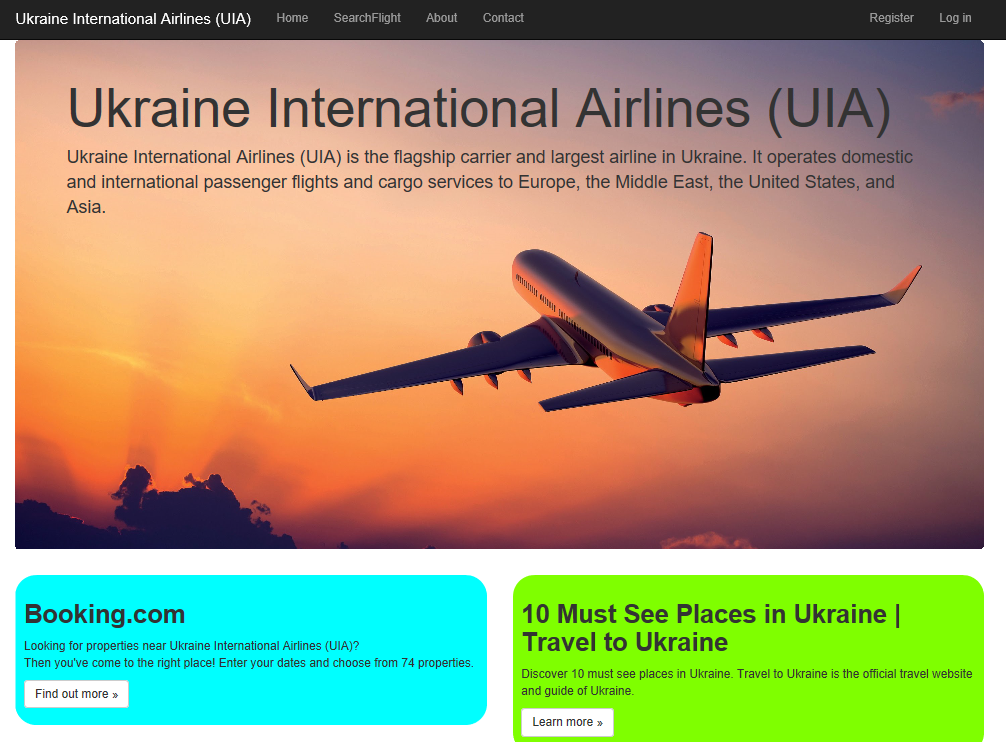
### 3.3 Sitemap



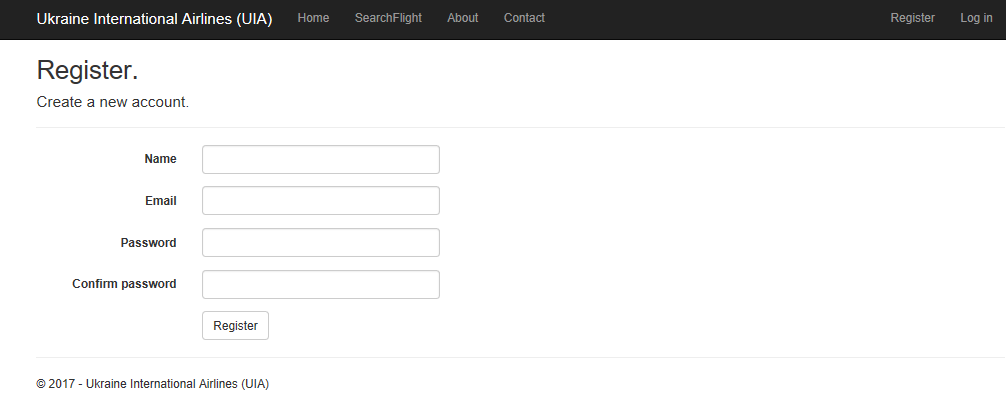
Above shows the sitemap for the UIA Online Flight Booking System which describes the flow of the application pages. When the user first accesses the website the user will be shown the Home page. The unregistered user of the website will be able to view 5 of the web pages as shown above, but to make bookings on the UIA Online Flight Booking System the user will be required to login to view the other pages of the website.

## 3.4 Web Application User Interface

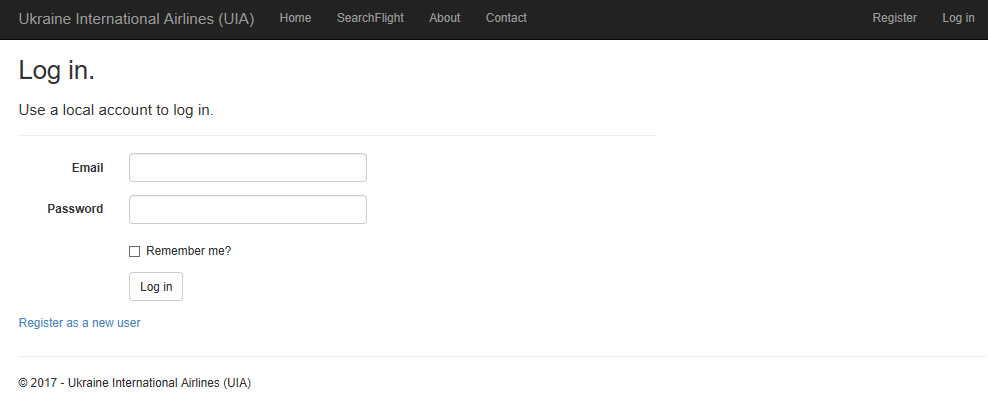
### 3.4.1 Home Interface



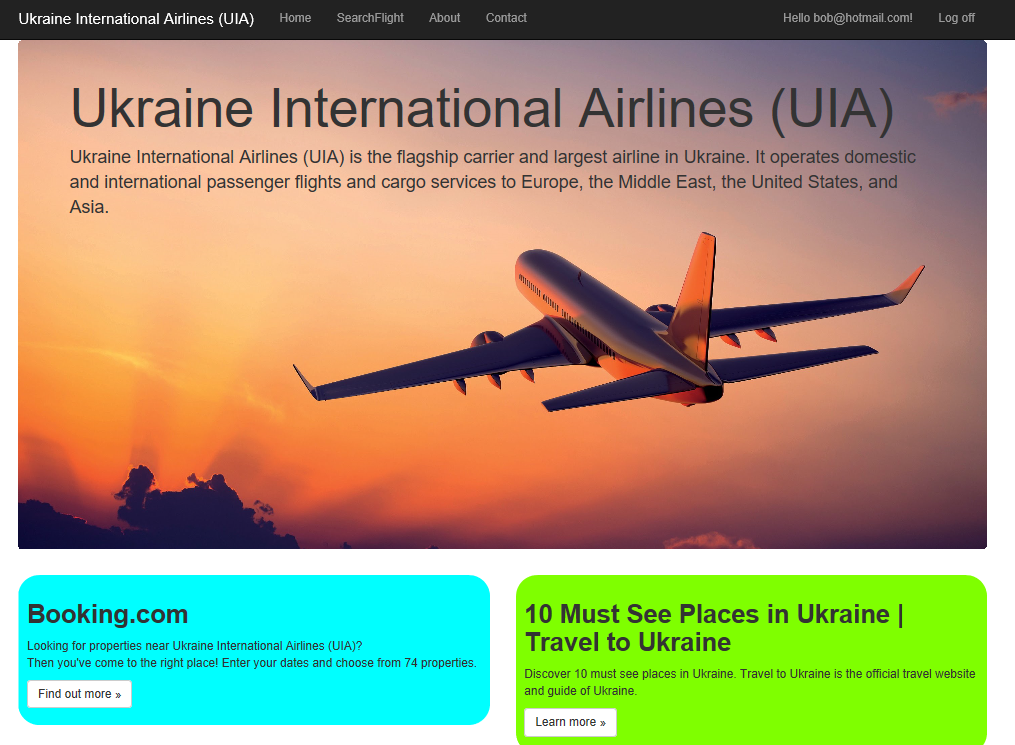
### 3.4.2 Register Interface



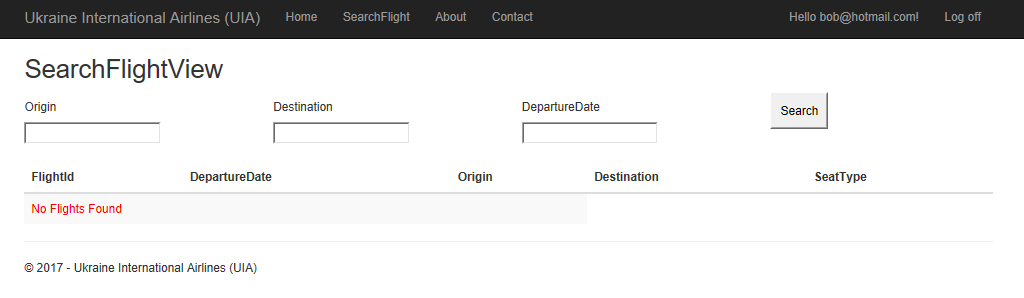
### 3.4.3 Login Interface



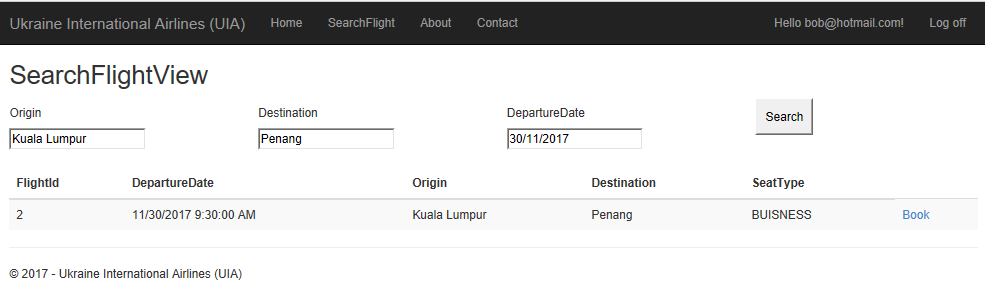
### 3.4.4 Home Interface when Login into Account



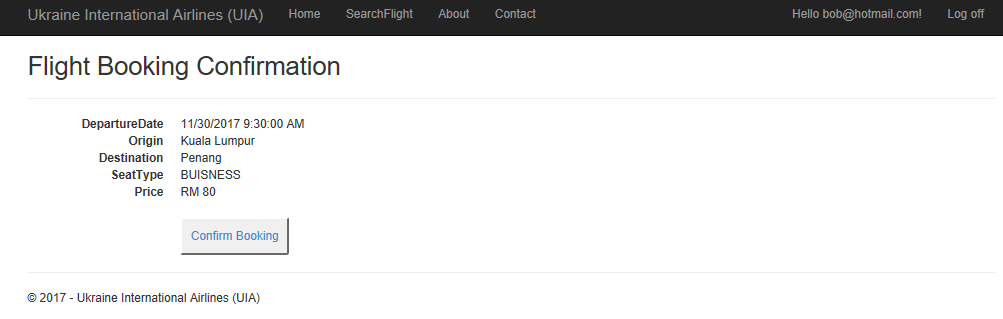
### 3.4.5 Search Flight Interface



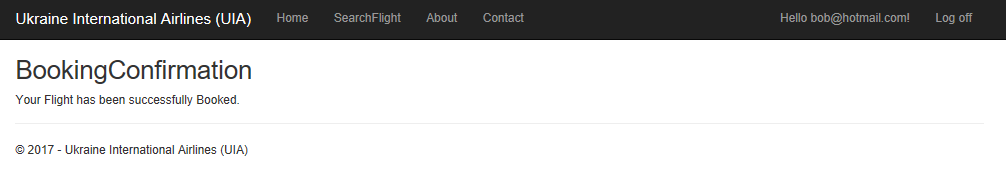
### 3.4.6 Search Flight Results Interface



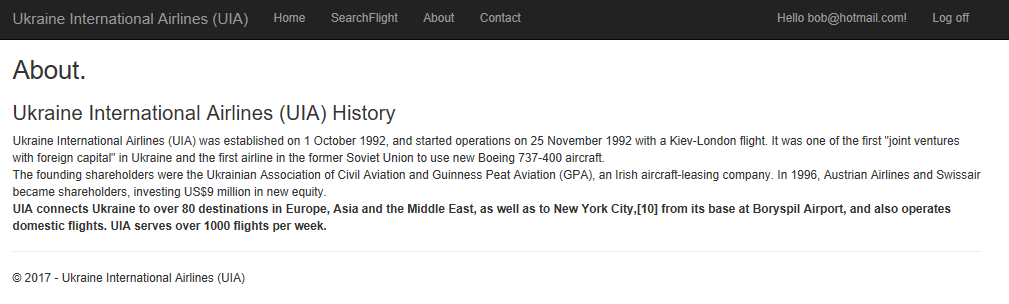
### 3.4.7 Flight Booking Confirmation Interface



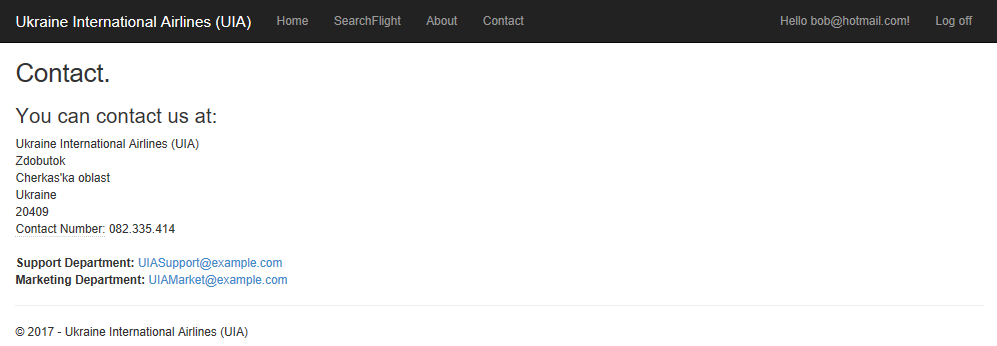
### 3.4.8 Booking Confirmation Interface



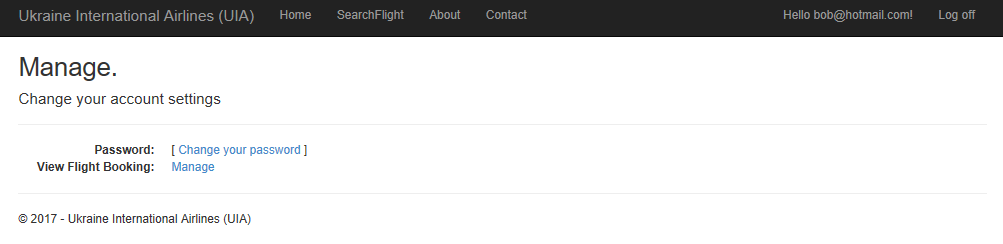
### 3.4.9 About Interface



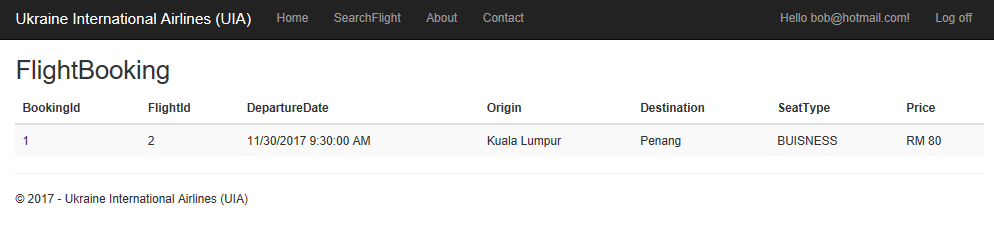
### 3.4.10 Contact Interface



### 3.4.11. Profile Interface



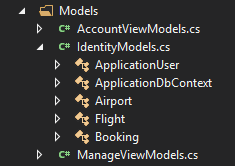
### 3.4.12 View Flight Booking Interface



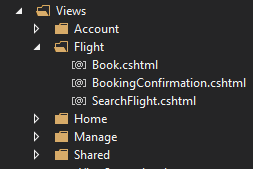
# 4.0 Implementation

## 4.1 Application Development

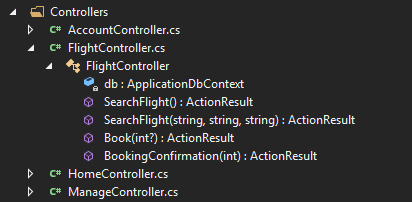
The development of the Ukraine International Airlines (UIA) Online Flight Booking System was done on Microsoft Visual Studio 2017 using the ASP.NET MVC and MySQL database. The system will be using the MVC structure which consists of the Model, View and Controller. The reason for using the ASP.NET MVC Framework provides a different pattern for web applications in comparison to the ASP.NET Web Forms. (Microsoft, 2017) The Model objects stores the data for the application and will be used to retrieve the information from the database. The flight, airport and booking objects are stored in the IdentityModel.cs as seen in below.



The views are used to store the user interface of the application. They are separated differently because different user interfaces will be based on the Model. (Microsoft, 2017) Most of the Flight user interfaces will be referenced using the models from the IdentitiyModels.cs.



Lastly, the controller will be used to handle user interaction and will work alongside the model which will select the view to display the user interface. (Microsoft, 2017) As seen below, the Flight Controller will access the IdentitiyModels.cs which contains the objects needed for the controller and will be displayed in the Flight View.



The reason for using the MVC Pattern is that is helps provide the separation between the different logics as it will helps promote parallel development so that the developers will be able to develop the MVC application separately. As different developers can work on different part such as the business logic in the model, or the view, or the controller logic.

## 4.2 Azure Publishing

### 4.2.1 SQL Database

Before publishing the Web Application the Azure SQL Database deployment has to be done before hand.

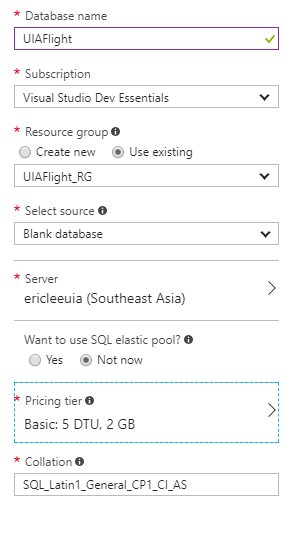


Figure 1: SQL Database Properties

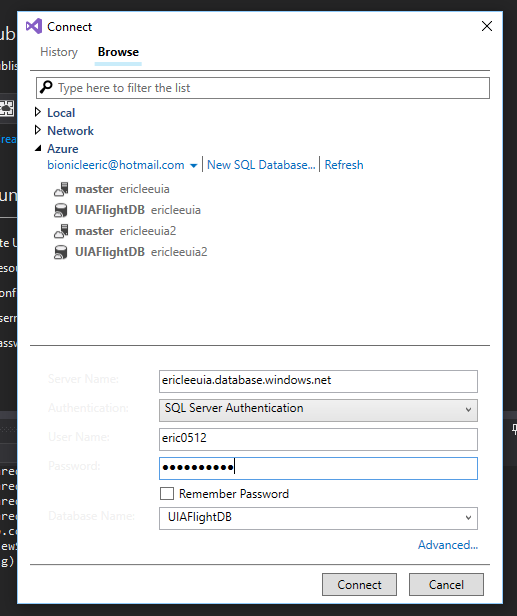


Figure 2: Connection to the Azure Server containing the database

Once the Azure SQL Database has been created on Azure, the connection to the database on the SQL Server will be done on Microsoft Visual Studio 2017. Once the connection has been established, later on after the web application has been deployed on Azure, the database connection string has to be attached onto the Azure application to ensure the database can be used. After that, the database data is needed to be updated using the Nuget Package Management which the team will just have to enter Update-Database inside the Package Manager Console which will create the tables required for the database and later inserts query will be developed to insert the data into the database for usage later.

### 4.2.2 Web Application publishing

Next, the Ukraine International Airline (UIA) Online Flight Booking System will be published on Azure Web Application Services whereby the application will be deployed on a server. Below shows the image screenshot of the deployment for the Azure Web App.

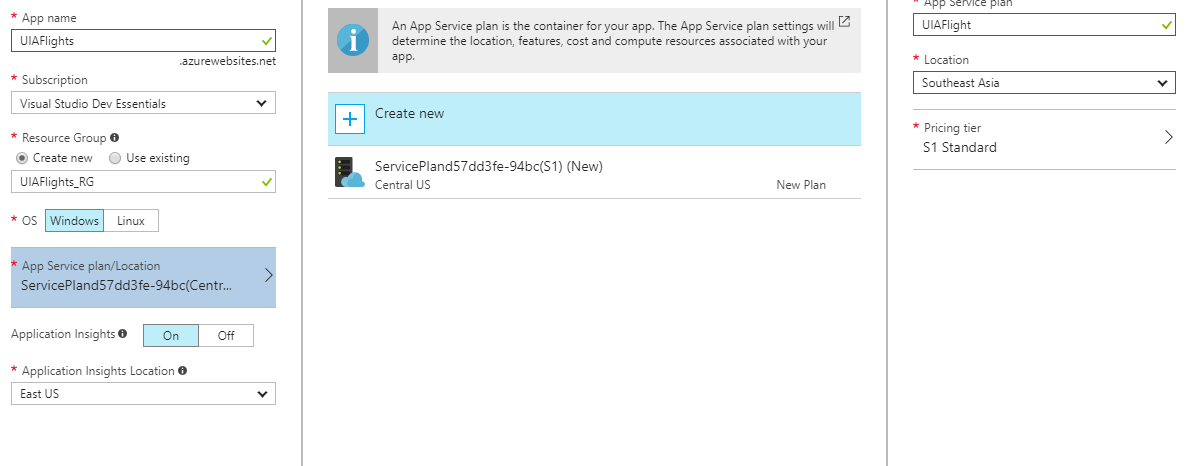


Figure 3: Azure Web App Deployment

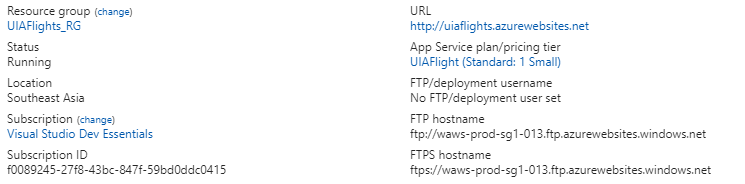


Figure 4: Web Application details

After the Azure Web Application service was deployed on Azure, the MVC ASP.NET Ukraine International Airlines (UIA) Online Flight Booking System will be deployed using the Microsoft Visual Studio 2017. The team will select existing as the Microsoft Azure App Service has been created on the Azure platform beforehand. Therefore, all the development team have to do is to select the right Resource Group containing the Web Application Services and deploy it based on the region.

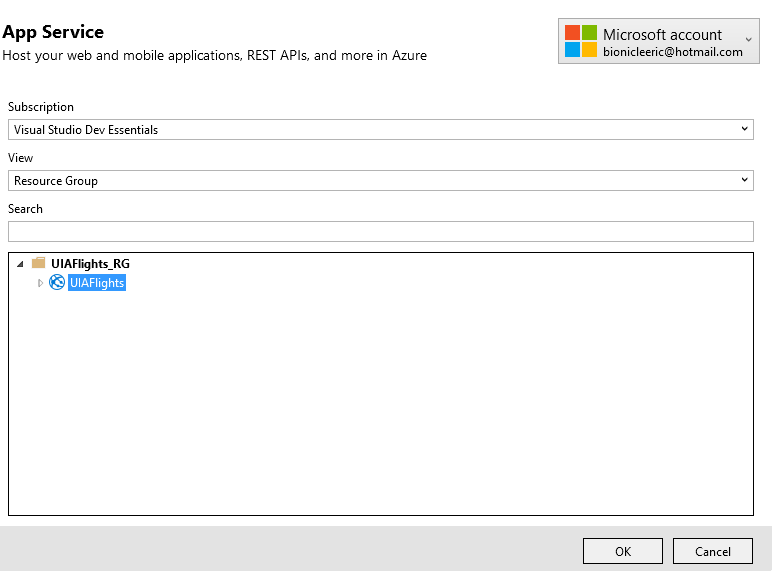


Figure 5: Selecting App Service

The web application will then be published through Microsoft Visual Studio accordingly, based on the different regions.

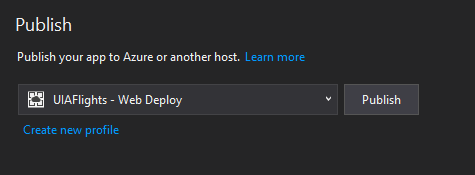


Figure 6: Publish System on the Sea Region Microsoft Visual Studio 2017

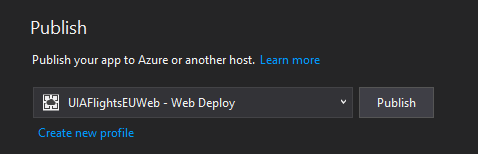


Figure 7: Publishing System on the West Europe region on Microsoft Visual Studio 2017

### 4.2.3 Azure Traffic Manager

When both of the applications have been published based on the different region, the traffic manager will be implemented to control the user traffic. The Azure Traffic Manager will use the Domain Name System (DNS) to direct that users based on their endpoint varying from the current health of the endpoints and traffic routing method. (Dwivedi & Wheeler, 2017) The traffic manager will be used to help redirect the customers to a new location if a failure does occur and also distribute the user traffic over multiple locations. (Azure, 2017)

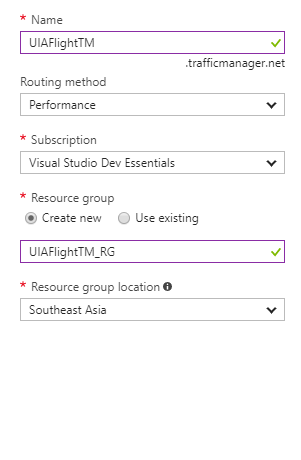


Figure 8: Implementing Traffic Manager for Sea Region

The Traffic Manager will be implemented for both the Southeast Asia region and West Europe region. When the traffic manager has been implemented the users will be able to visit the website through the traffic manager DNS. This will relocate the web app services based on the URL link of the web application. Below shows the traffic manager directing Southeast Asia users to the website. The URL will be <http://uiaflighttm.trafficmanager.net/> in comparison with to the West Europe URL which will be <http://uiaflighttmeu.trafficmanager.net/> for the West Europe users.

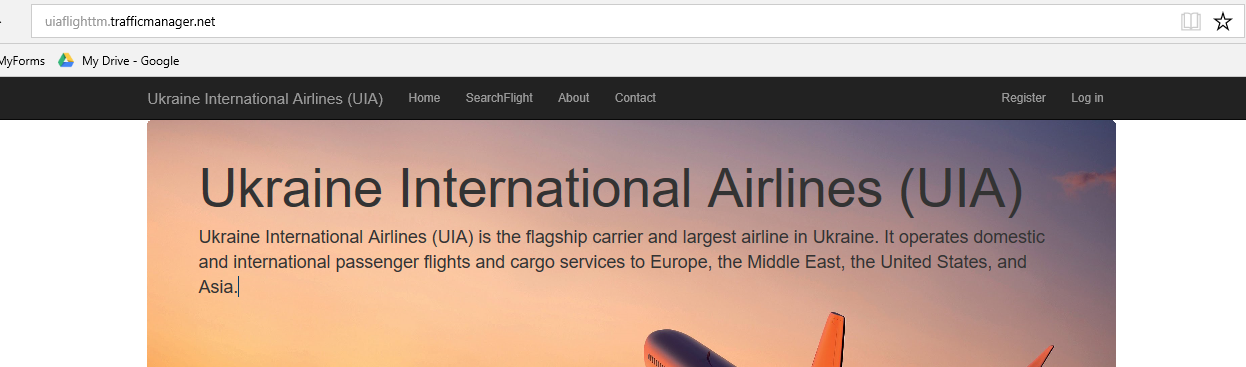


Figure 9: Traffic Manager directing Southeast Asia users

## 4.3 Application Scaling

The deployment of the web application will be using the S1 tier App Service plan. There are many selections available for the App Service but due to the Azure budget restrictions of RM150, the S1 Standard tier App Service Plan will be selected when deploying to the Southeast Asia region and the West Europe region. The Standard Plan helps provide important features for the web application development such as custom domain. Besides that, the standard plan also provides daily backup for the data to ensure no loss in data as well as web application staging to allow the application deployment to be rolled back to the last functional slot in event of any error occurring in time to come. Moreover, the system hasn’t been fully published live yet for real time usage, as the system will need to be verified by the Ukraine International Airlines Company. Once it has been verified and meets the requirement the service plan of the system will be scale up to the S3 Standard tier App Service Plan or even higher as it will allow the website to have more higher capabilities server. By using the S3 Standard tier App Service Plan it can provide a faster services as the increase in the number of cores and processing memory for the server to handle the traffic based on the region.



Figure 10: App Service Plan

The traffic manager is also important as it will be used to redirect the traffic of the users of the website based on their location. For example, people that are using the website from Southeast Asia will be directed to the web app deployed on Southeast Asia however those from West Europe will be transferred accordingly.

For Ukraine International Airlines (UIA) users that want to access the site from the same domain scaling out but still able to access it from a different region is critical. This is because For the SEA region the fixed instance will be 4 as many of the users. This is crucial to save the costs as more instances require higher cost. However for the West Europe Region, 4 application instances will be set along with auto scaling enabled based on the server load. This is because many people that will use the airline will be from Ukraine West Europe.

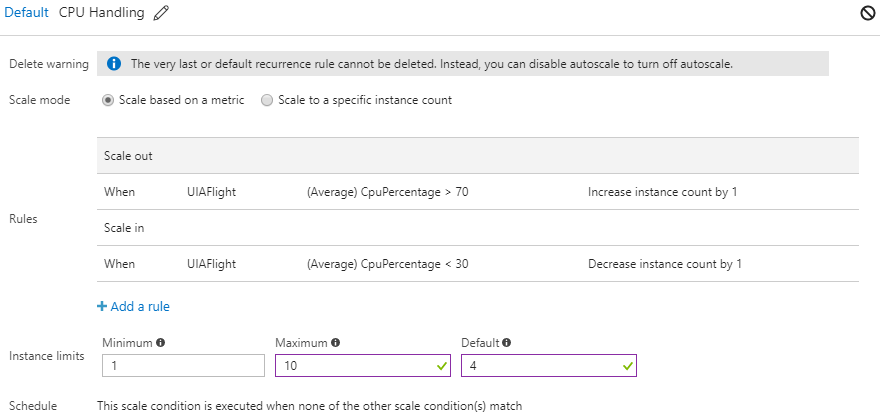


Figure 11: Scale In and Scale out West Europe Region

The scale out will only occur if the CPU handling or usage is more than 70% whereby an additional instance will be added. If the CPU usage is less than 30% an instance will be removed. Finally for the MySQL Server uses a basic level currently for the testing stages, but will be increased to the standard tier as it will be able to handle more traffic from different regions. Also if more flights and bookings are made it has to be increased as well.

## 4.4 Investigate and Analyse Application

Using Azure we are able to help investigate and analyse the UIA Online Flight Booking System performances through the tools provided by Azure. This is essential as the UIA Online Booking System might grow as time to come or if there are performance issues through Azure we will be able to view it. Azure provide many web app performance tools to help monitor the activity of our Azure web application. (Lin & Hinton, 2017)

Below figure displays the overview of the web application statistics regarding the performance of the web application which are the HTTP Server Errors, Data In, Data Out, Requests, and Average Response Time of the web application. The app service helps provide monitoring functionality through the Azure Portal to review quotas and metrics for an application. (Tardif & Lin, 2016)

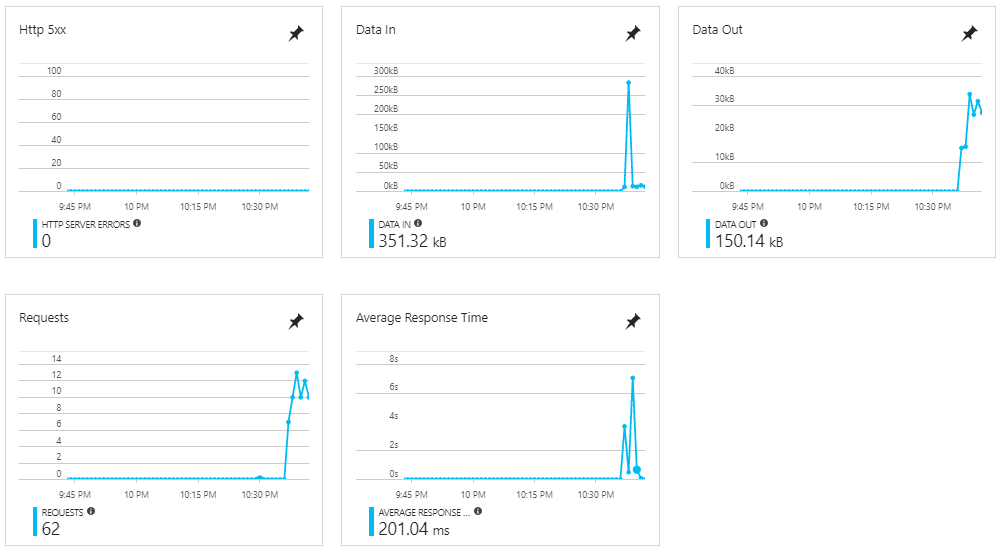


Figure 12: Instrumentation Data from App Service Overview.

Moreover, through the Application Insights for the web application we are able to get actionable insights through application performance management and instant analytics of the web application. (Azure, 2017) The Application Insights allow us to get live data of the web application used and the server response time and request from the application. It can also display how many users are on the web application as well. Below diagram shows the overview of the Application Insights. It can be seen that the HTTPGet Function shows the slowest request in comparison to the HTTPPost function.

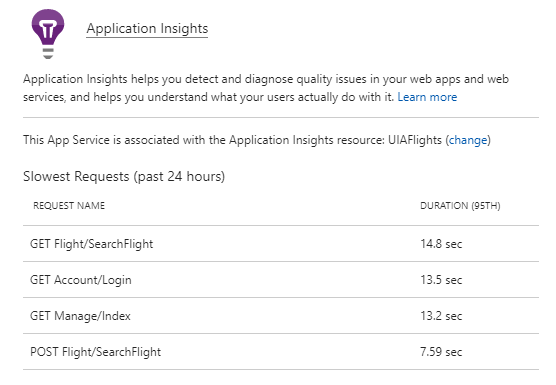


Figure 13: Application Insights

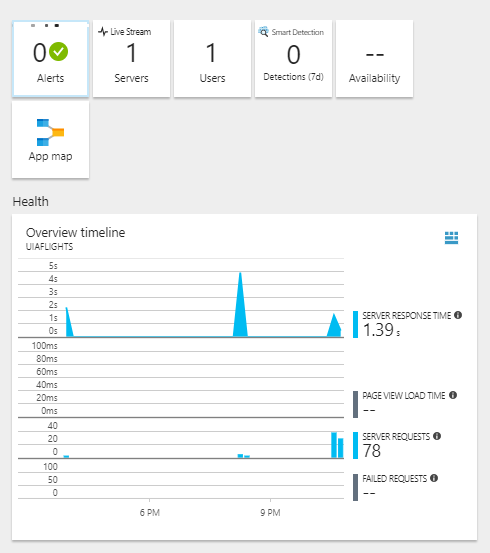


Figure 14: Application Insights Overview

Through the Azure monitoring services we are also able to review the CPU and Memory Percentage of the UIA Flight service plan. As seen below we can see there is an activity that happens around 3pm onwards for 19th of November.

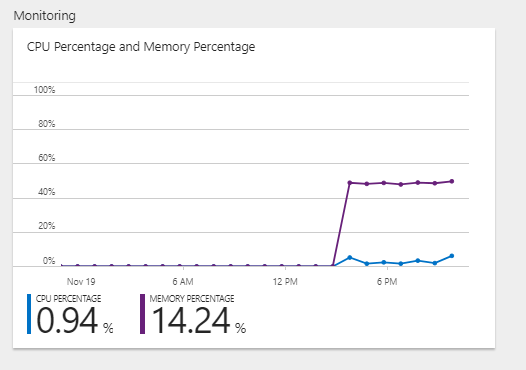


Figure 15: Service Plan Overview

For the database of the flight, we are able to view the database query performance insights of the queries. Through this we will be able to making tuning for the queries to help make improvement to the performance. (Stein & Rebeler, 2017)

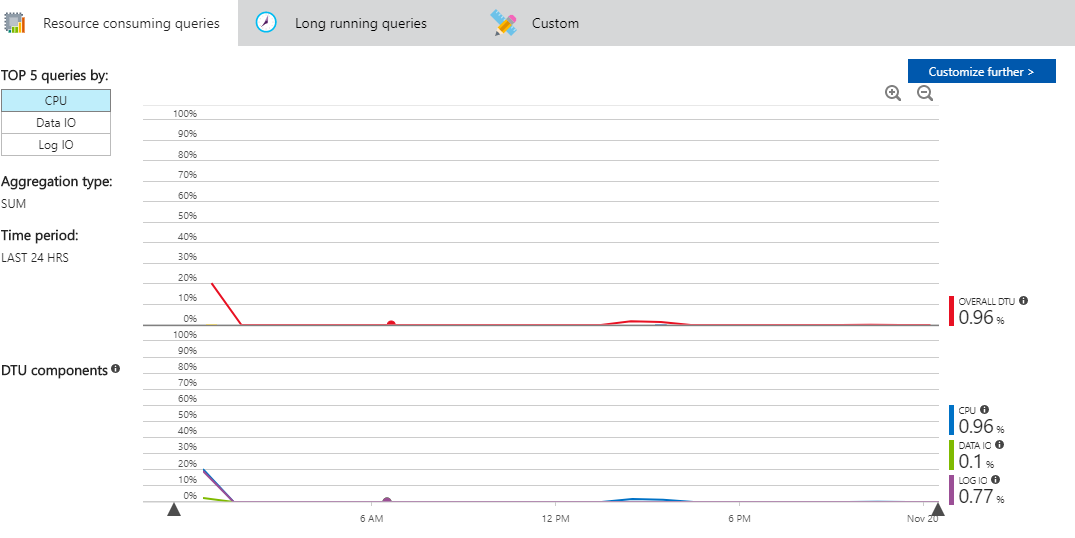


Figure 16: Resource Consuming Queries

## 4.5 Discussion of Managed Database

The cloud platform and online managed services has been increasing in popularity as well as market growth as time passes. Microsoft Azure has been continuously growing and has shown the largest growth in the market share for the 2nd quarter. (Hall, 2017) This is because cloud platforms are introducing new types of business model and more globally based integration networks in the year 2017. (Columbus, 2017)

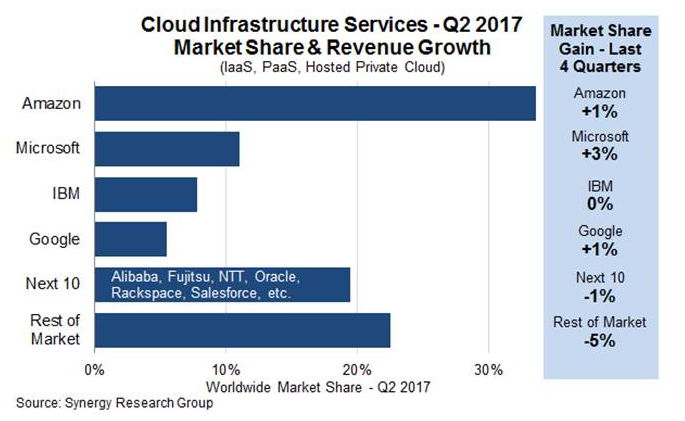


Figure 17: Cloud Infrastructure Services Growth Q2 2017 (Hall, 2017)

Furthermore, the Platform-as-a-Service adoption is currently going to be the fastest growing sector among the cloud platforms based on KPMG analysis. Many companies can be seen adopting cloud services due to data intensive operations performed on the cloud. (Columbus, 2017)

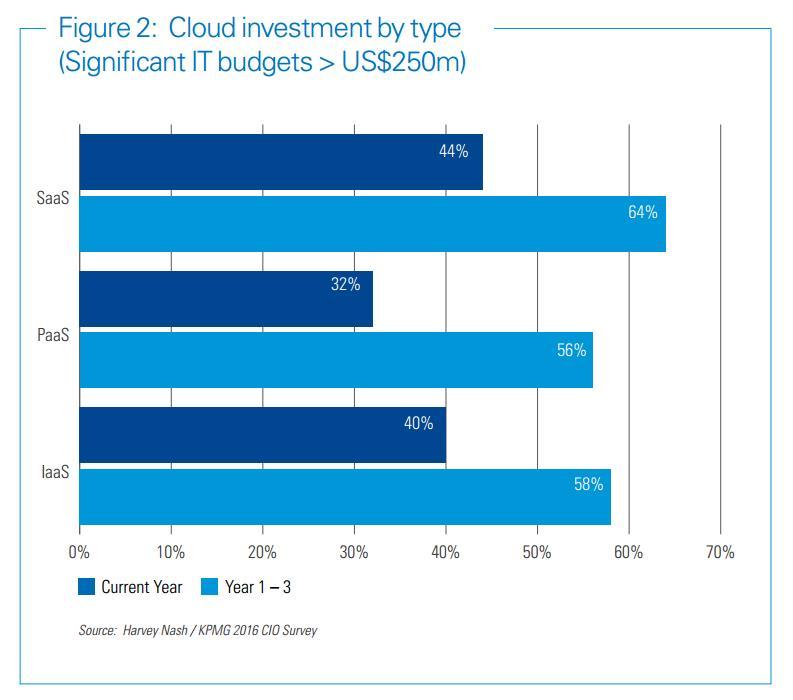


Figure 18: Cloud investment (Source: KPMG 2016 CIO Survey)

It can be seen very clearly that cloud infrastructure services is growing in the industry. The Platform-as-a-Service refers to a complete development and deployment which is done through the cloud. (Azure, 2017) It helps us deliver simple cloud-based applications to complicated cloud-enabled enterprise applications. PaaS helps handles the cloud service provider which are the backend services and developer just has to manage the developed application along with the services developed. (Azure, 2017)

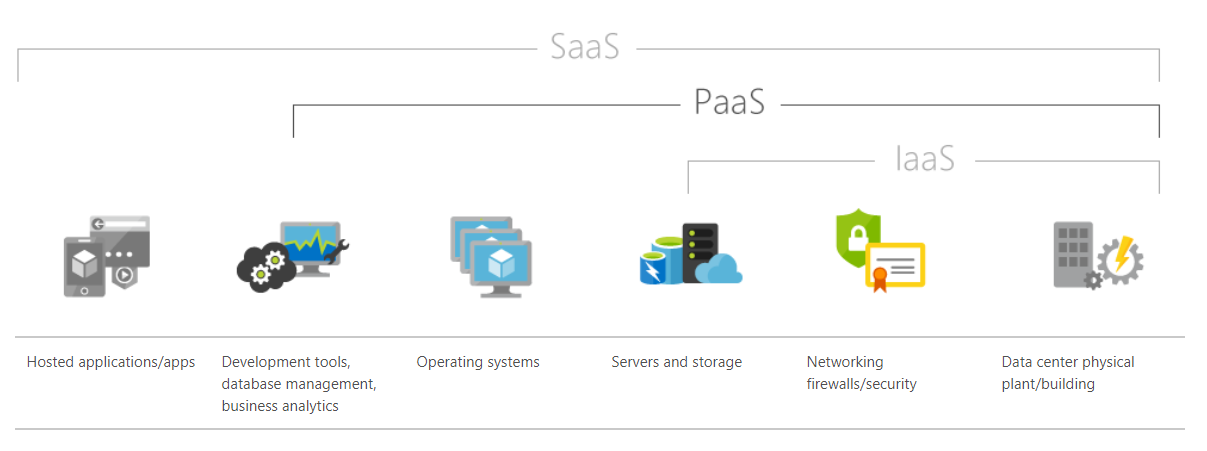


Figure 19: PaaS Cloud Service

PaaS is mainly used for development framework as it helps provide a framework to help developers build their applications on. Not to mention PaaS allows developers to analyse business data, patterns and findings which can be used to help improve forecasting and investment returns. (Azure, 2017) Finally, the advantages that PaaS provides through Microsoft Azure is that it helps cut coding time, support geographically distributed development teams, and also manage application lifecycle efficiently. Azure SQL Database is a relational cloud database that helps provide advance built-in protection and security features. (Azure, 2017) This allows the reduction in administrative tasks of the Azure SQL Database rather than using local SQL Database as this will help provide the developers more time to focus on the application development. Besides that, with PaaS it helps also provide sophisticated tools affordably such as Microsoft Azure that provides many types of cloud services at different price range to help developers with their application development including database services.

# 5.0 Test Plan

## 5.1 Unit Testing

### 5.1.1 Unit Testing for Registration

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tester Name:** | | Lee Jhoe Yeuw, Eric | | **Test Date:** | **18/11/2017** |
| **Test Case** | **Test Scenario Description** | | **Expected Results** | | **Actual Results** |
| 1. | 1. The user will select the register page. 2. Fill up the required fields. 3. Once filled up, click on “Register” button. | | Account will be successfully created and user will be login to his/her account. | | As Expected |
| 2. | 1. The user will select the register page. 2. Do not fill up the required fields. 3. Click on “Register” button. | | Alert Message will be displayed requesting use to fill up required fields | | As Expected |

### 5.1.2 Unit Testing for Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tester Name:** | | Lee Jhoe Yeuw, Eric | | **Test Date:** | **18/11/2017** |
| **Test Case** | **Test Scenario Description** | | **Expected Results** | | **Actual Results** |
| 1. | 1. The user will select the login page. 2. Enter username and password. 3. Click on “Login” Button. | | Direct user to the home page with the username being displayed at header saying Hello. | | As Expected |
| Display error message that, username and password is incorrect and doesn’t match any data in the database. | | As Expected |
| 2. | 1. The user will select the Login page. 2. Click on “Login” Button. | | Display error message to ask user to insert value into requested field. | | As Expected |

### 5.1.3 Unit Testing for Search Flights

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tester Name:** | | Lee Jhoe Yeuw, Eric | | **Test Date:** | **18/11/2017** |
| **Test Case** | **Test Scenario Description** | | **Expected Results** | | **Actual Results** |
| 1. | 1. The user will select the search flight page. 2. Enter the origin, destination and departure date. 3. Click on “Search” Button. | | Display searched flight details. | | As Expected |
| 2. | 1. The user will select the search flight page. 2. Click on “Search” Button. | | Alert Message will be displayed requesting use to fill up required fields. | | As Expected |

### 5.1.4 Unit Testing for Booking Flights

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tester Name:** | | Lee Jhoe Yeuw, Eric | | **Test Date:** | **18/11/2017** |
| **Test Case** | **Test Scenario Description** | | **Expected Results** | | **Actual Results** |
| 1. | 1. After finding a flight. 2. Click on “Book button”. | | Display Booking confirmation page. | | As Expected |
| Redirect to Login Page | | As Expected |
| 2. | 1. After finding a flight. 2. Click on “Book button”. 3. Click on “Confirm Booking” button. | | Display Booking Confirmation Page, and after confirmation button was clicked, Flight booking is saved. | | As Expected |

### 5.1.5 Unit Testing for Viewing Booked Flights

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Tester Name:** | | Lee Jhoe Yeuw, Eric | | **Test Date:** | **18/11/2017** |
| **Test Case** | **Test Scenario Description** | | **Expected Results** | | **Actual Results** |
| 1. | 1. The user has login to system 2. The user clicks view flights booked button. | | Display flight booked details table with flights booked. | | As Expected |
| Display empty flight booked details table | | As Expected |

## 5.2 Performance Testing

To test the performance of the web application, the azure performance testing tool can be used as this will allow us to simulate a specific amount of users from a certain region that requests a specific URL of the web application. Below shows the configuration of the different tests varying from different amount of users which will request the Home page of the web application for 5 minutes.

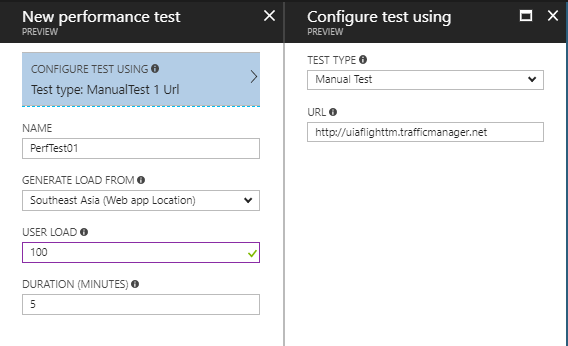


Figure 20: Performance Test Configuration

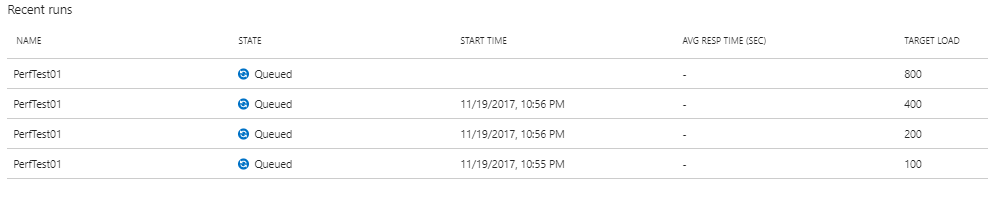


Figure 21: 4 Performance Test varying number of users

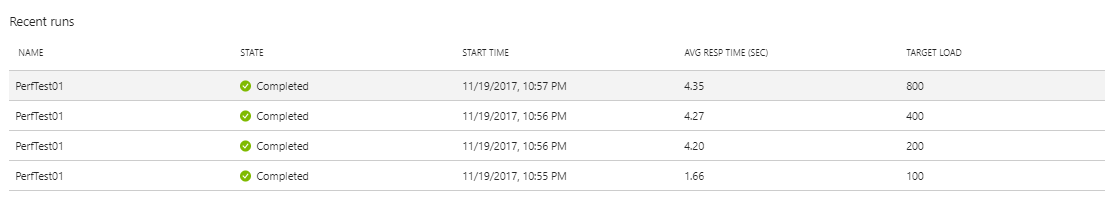


Figure 22: Results of Performance Test

A total of 4 performance test was conducted which will simulate different amount of users, such as a 100, 200, 400 and 800 users that will be accessing the same webpage for 5 minutes in the same region. Below shows the table of the documented results retrieved from the performance test.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| User Load  (Concurrent) | 100 | 200 | 400 | 800 |
| #Successful Requests | 11441(100%) | 12165(100%) | 24149(100%) | 50811(100%) |
| #Failed Requests | 0 | 0 | 0 | 0 |
| Average Response Time (secs) | 1.66 | 4.2 | 4.27 | 4.35 |
| Requests/secs | 38.14 | 40.55 | 80.5 | 169.37 |

Through the results above we can see that if there is more than 200 users using the system, it will take a longer time to handle the request in comparison to less than 200 users.

# 6.0 Conclusion

In conclusion, the development of the Ukraine International Airline (UIA) Online Flight Booking System was successful. It was developed using ASP.NET MVC5 and C# through the Microsoft Visual Studio 2017 IDE. The application was hosted through Azure using the Web Application Services as well as the SQL Database service and Traffic Manager Services that will be required for the system to run successfully. Many different aspects when developing the system were needed to be looked at instead of a normal development of an application. This is because this system will be deployed for international usage. Therefore, the right selection of pricing plans for the different Azure services provided are needed to be made wisely.

Through this project it was a great lesson as it helps give developers and insight of how development works in the real world. As there are many different factors that are needed to be looked at rather than just coding an application. Publishing an application using a cloud service also helps the developers as the cloud platform provides many services that can ease the development of the application while the cloud platform handles the backend stuff. Using Microsoft Azure also helps us analyse the performance of our system and allows us to scale in or out accordingly. Moreover through the traffic manager services it can help handle the traffic when different users from different regions are accessing the system and will divert them to the respective web application based on the region accordingly.

All in all, this assignment has proven to be quite a challenge as we have to manage our time wisely due to the budget constraints provided and ensure the requirements for the Ukraine International Airlines (UIA) Online Flight Booking System are fully met. Finally, without Microsoft Azure the development of the system would not have been successful.

# 7.0 References

1. Azure, M., 2017. *Application Insights.* [Online]   
   Available at: https://azure.microsoft.com/en-us/services/application-insights/  
   [Accessed 19 November 2017].
2. Azure, M., 2017. *SQL Database.* [Online]   
   Available at: https://azure.microsoft.com/en-us/services/sql-database/  
   [Accessed 19 November 2017].
3. Azure, M., 2017. *Traffic Manager.* [Online]   
   Available at: https://azure.microsoft.com/en-us/services/traffic-manager/  
   [Accessed 18 November 2017].
4. Azure, M., 2017. *What is PaaS?.* [Online]   
   Available at: https://azure.microsoft.com/en-us/overview/what-is-paas/  
   [Accessed 19 November 2017].
5. Columbus, L., 2017. *Roundup Of Cloud Computing Forecasts, 2017.* [Online]   
   Available at: https://www.forbes.com/sites/louiscolumbus/2017/04/29/roundup-of-cloud-computing-forecasts-2017/#63b204ab31e8  
   [Accessed 19 November 2017].
6. Dwivedi, K. & Wheeler, S., 2017. *Overview of Traffic Manager.* [Online]   
   Available at: https://docs.microsoft.com/en-us/azure/traffic-manager/traffic-manager-overview  
   [Accessed 18 November 2017].
7. Hall, C., 2017. *Microsoft's Cloud Market Share Grew More than Anyone Else's Last Quarter -- Analysts.* [Online]   
   Available at: http://www.datacenterknowledge.com/business/microsofts-cloud-market-share-grew-more-anyone-elses-last-quarter-analysts  
   [Accessed 19 November 2017].
8. Lin, C. & Hinton, B., 2017. *Monitor Azure web app performance.* [Online]   
   Available at: https://docs.microsoft.com/en-us/azure/application-insights/app-insights-azure-web-apps  
   [Accessed 18 November 2017].
9. Microsoft, 2017. *ASP.NET MVC Overview.* [Online]   
   Available at: https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx  
   [Accessed 18 November 2017].
10. Stein, S. & Rebeler, C., 2017. *Azure SQL Database Query Performance Insight.* [Online]   
    Available at: https://docs.microsoft.com/en-us/azure/sql-database/sql-database-query-performance  
    [Accessed 19 November 2017].
11. Tardif, B. & Lin, C., 2016. *How to: Monitor Apps in Azure App Service.* [Online]   
    Available at: https://docs.microsoft.com/en-us/azure/app-service/web-sites-monitor  
    [Accessed 18 November 2017].

# 8.0 Appendix

## 8.1 Stream Video Presentation

Link:

<https://web.microsoftstream.com/video/8fe46fc6-698e-42eb-87132a22c5c46ec4?list=studio>

## 8.2 GitHub Repository

Link: <https://github.com/emertXE94/DDACUIA>