# Acknowledgements

First of all, I would like to thank to my parents for their continuous support and reinforcement that help me complete this project. I also would like to thank my friends who offered to help me with their most effort. After that, I would like to thank to my lecturer DR. KALAI ANAND RATNAM for his guidance and expertise in the field of cloud computing to help me to complete this assignment properly, his suggestions have made my assignment more professional and without his encouragement and the dedication of time, my assignment would not have been a success.

Table of Contents

[Acknowledgements 1](#_Toc495319289)

[1.0 Introduction 3](#_Toc495319290)

[1.1 Project Background 3](#_Toc495319291)

[1.2 Project Scope and Objective 3](#_Toc495319292)

[1.3 Project Specification 4](#_Toc495319293)

[2.0 Project Plan 5](#_Toc495319294)

[2.1 Gannt Chart 5](#_Toc495319295)

[3.0 Design 6](#_Toc495319296)

[3.1 Cloud Architecture Diagrams 6](#_Toc495319297)

[3.2 Use case Diagram 7](#_Toc495319298)

[3.3 Data Modelling 8](#_Toc495319299)

[3.4 Sidemap 8](#_Toc495319300)

[4.0 Implementation 9](#_Toc495319301)

[4.1 Application development 9](#_Toc495319302)

[4.2 Screenshot of system 11](#_Toc495319303)

[4.2.1 Main Page 11](#_Toc495319304)

[4.2.2 Login Page 11](#_Toc495319305)

[4.2.3 Register Page 12](#_Toc495319306)

[4.2.4 Flight View Page 12](#_Toc495319307)

[4.2.5 Booking View Page 13](#_Toc495319308)

[4.2.6 Profile View Page 13](#_Toc495319309)

[4.2.7 Edit Profile Page 14](#_Toc495319310)

[4.3 Create and migrate SQL database 15](#_Toc495319311)

[4.4 Deployment Application to Microsoft Azure 17](#_Toc495319312)

[4.5 Traffic Manager 19](#_Toc495319313)

[5.0 Testing 21](#_Toc495319314)

[5.1 Unit Testing 21](#_Toc495319315)

[5.2 Performance Test 23](#_Toc495319316)

[6.0 Conclusion 24](#_Toc495319317)

[7.0 References 25](#_Toc495319318)

# Introduction

## Project Background

Online shoppers are notoriously fickle. If a website lags for even a few seconds, shoppers are just a couple of clicks away from many more options. 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 problems with its website prevented it from adequately serving customers beyond Ukraine. The site experienced severe denial-of-service (DOS) attacks, which hurt site performance and reliability, and it did not have the performance needed to host visitors from many parts of the world.

UIA has long used technology to reduce costs, innovate, and improve customer service. It has gone to a paperless cockpit and uses sophisticated software for analyzing fuel economy. The airline decided that it once again needed to innovate its way out of its web challenges.

Dmitriy Prudnikov, Chief Information Officer at Ukraine International Airlines, realized that migrating the website out of UIA datacenters into a public cloud could solve all these problems.

## Project Scope and Objective

The development of this web application is involved planning, designing, implementation and testing based on the requirements that state by Dmitriy Prudnikov.

The project was broken down into parts and the date to complete for each task was recorded by constructing a Gannt Chart. For ensure web application have high performance and availability, the application is tested and deployed to Microsoft Azure.

At the end of this project, a documentation which include introduction, project plan, development design, implementation of system and the testing of system will be produced.

## Project Specification

* **Provisioning**: To ensure the web application along with the SQL Server database to the Microsoft Azure Platform.
* **Maintainability**: To ensure the web application is able to be upgraded and other maintenance tasks are able to be performed while multiple users are using it.
* **Monitoring**: To ensure the web application is able to be monitored at all times, including how each user is using the application, in order to identify any problems and to troubleshoot them.
* **Availability**: To ensure the web application is constantly available to users and to ensure the activities of other tenants do not affect the availability of the application.
* **Scalability**: To ensure the web application scales to meet the demand of the application.
* **Functions** provided by Microsoft Azure, which have been **incorporated** into the project:
  + Deployment on the cloud
  + SQL database
  + Traffic Manager
  + Performance testing
  + Investigation and Analysis of application
  + Application Scaling

# Project Plan

## Gannt Chart

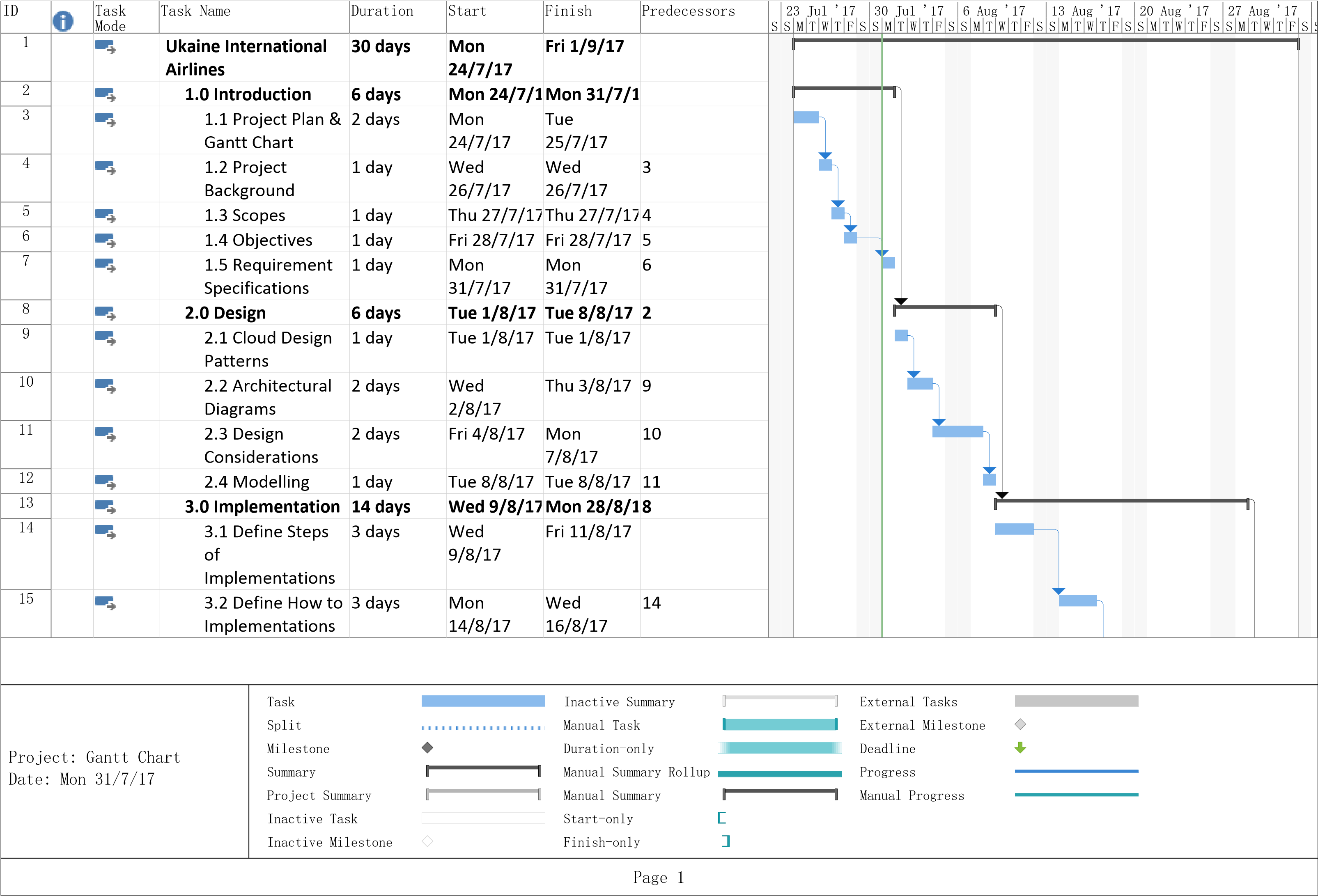


Figure 1: Gantt Chart

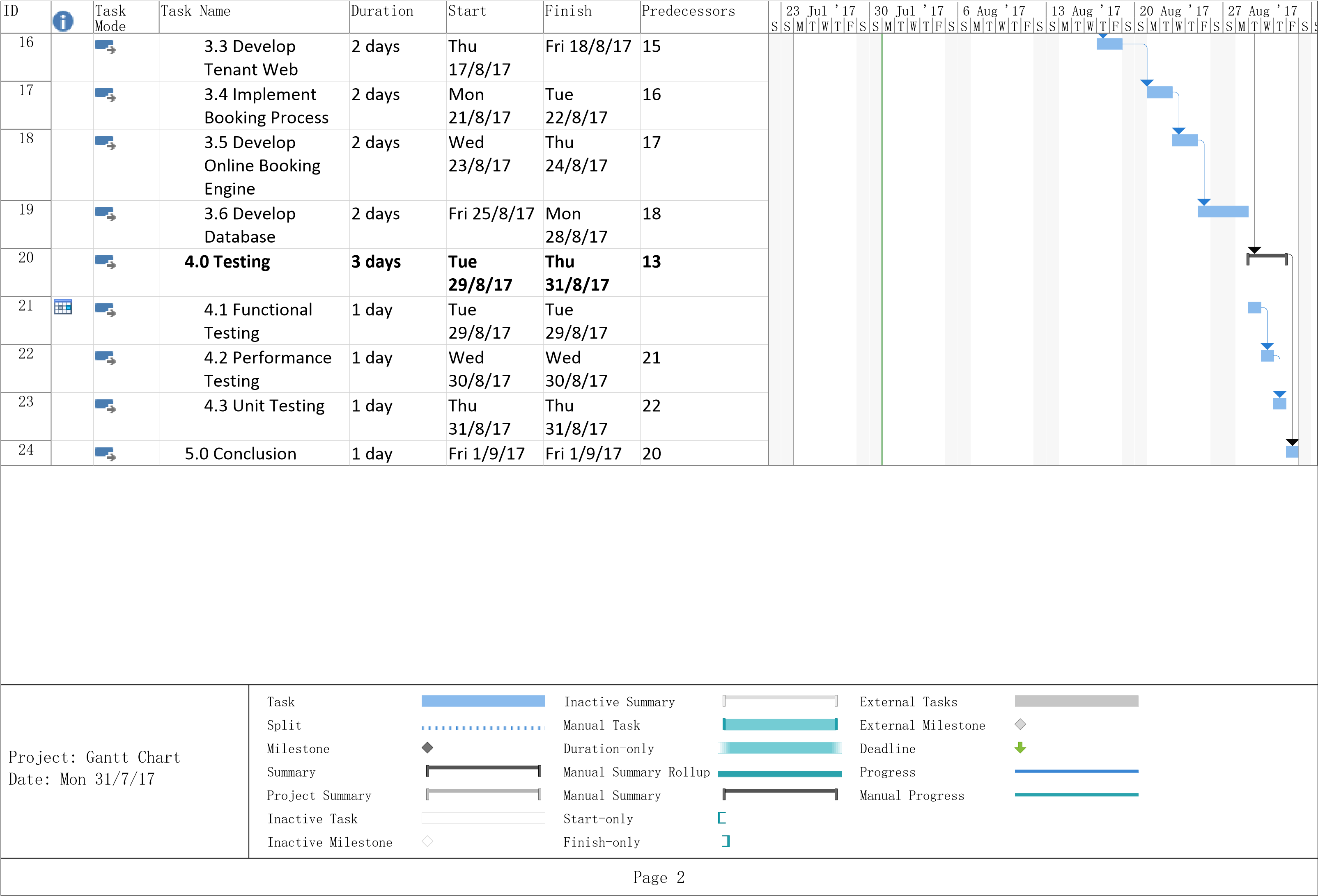


Figure 2: Gantt Chart 2

# Design

## Cloud Architecture Diagrams

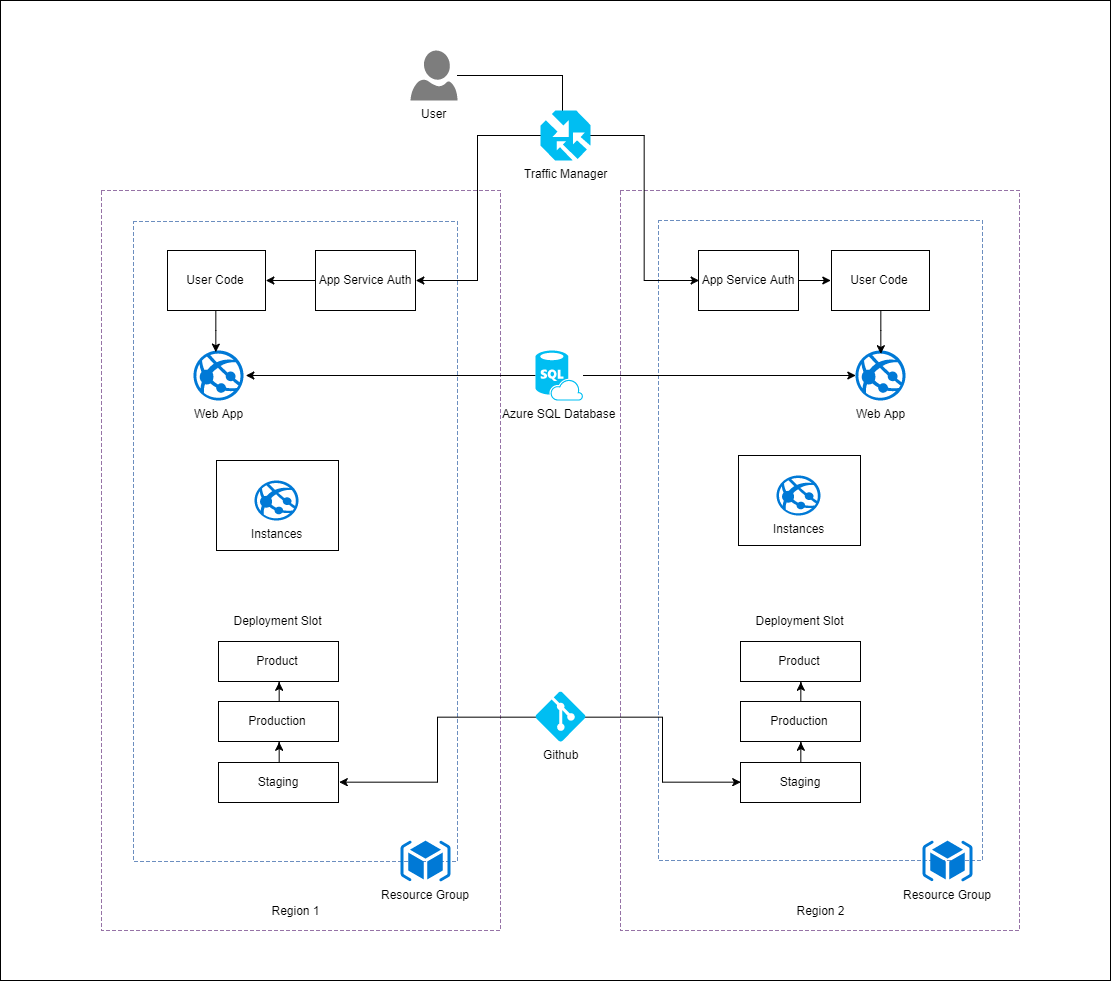


Figure 3: Cloud Architecture

Form observed from the diagram above, once the user tries to access the application using internet, the traffic manager identifies user’s location and based on the performance and the region where he is trying to connect from the routing method redirects that user to the closest and most reliable web application server.

Furthermore, there are two web application services connected to the SQL database provided by Microsoft Azure Cloud services, and the web application has in-built function to determine the region as well which identify the user location access and then showing the currency accordingly.

## Use case Diagram

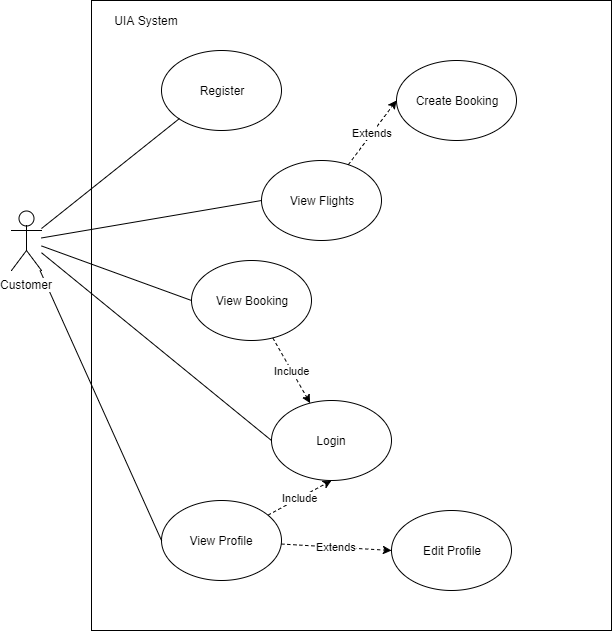


Figure 4: Use Case Diagram

## Data Modelling

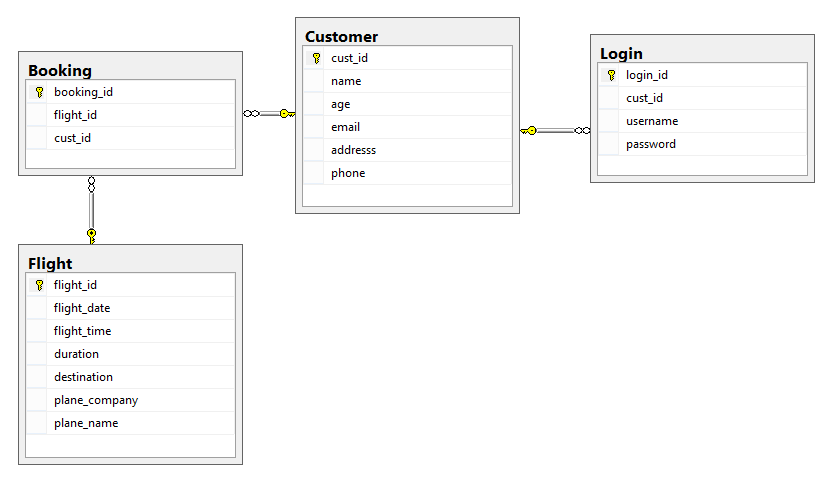


Figure 5: ERD

The diagram above is a ERD that shows the database model used for the application. The customer table contains all user information. The flight table contains all available flights, the booking table contains all bookings that made by user and the login table contains all the username and password of user.

## Sidemap

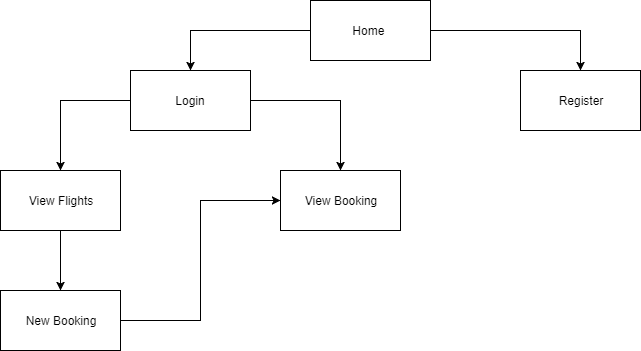


Figure 6: Sidemap

The diagram above is a Sidemap of the system, it shows the flow of the application pages.

# Implementation

## Application development

The development of the web application was used C# language in framework of ASP.Net and MySQL database. MVC structure was used on development this web application, which consists of controller, model and view.

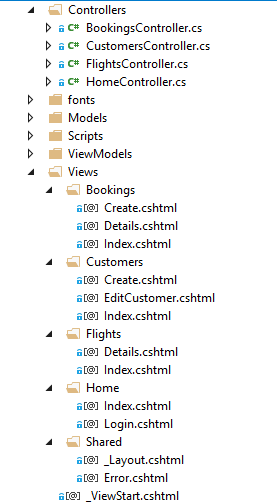


Figure 7: System Structure

The diagram above shows the project structure on the system. All views are control by Controllers, for instance, BookingController is control all the Views in the Booking folder. Controller is uses to manage the post and get of those views that under it.

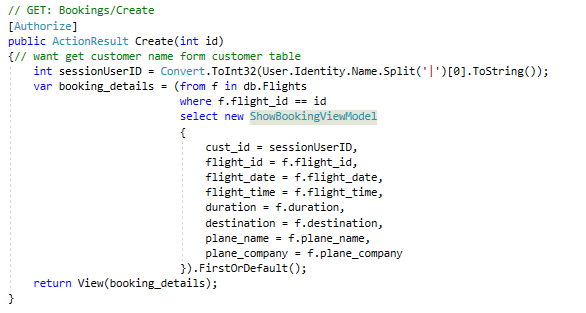


Figure 8: "Get" Codes

The diagram above is the “Get” of create booking page, it writes in the booking controller to let the create booking page to get what it wants. Create booking page will get all the variables that needed.



Figure 9: "Post Code"

The diagram above is the “Post” of create booking page, it use to let the view to give the variables to the controller to perform task.

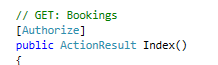


Figure 10: Authorize Code

For the page that need to login before access to it, [Authorize] will added at above the “Get” code of the page in controller. [Authorize] will redirect to login page if the user not yet login.

## Screenshot of system

### Main Page

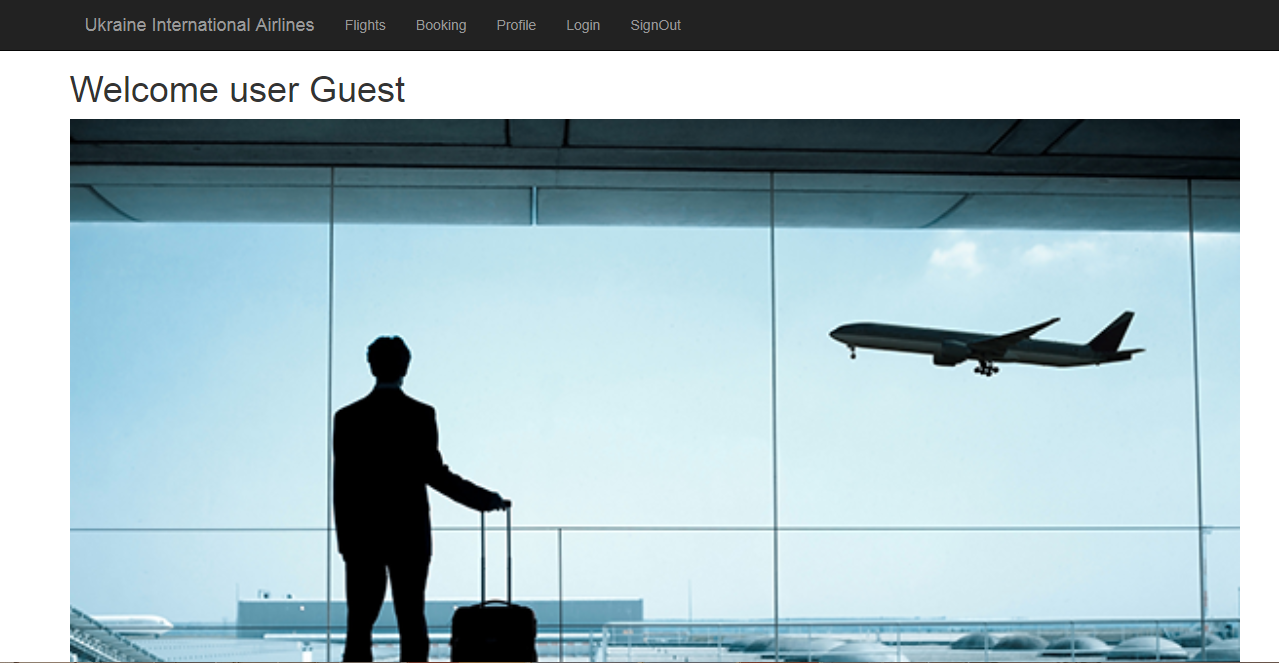


Figure 11: MainPage Screenshot

### Login Page

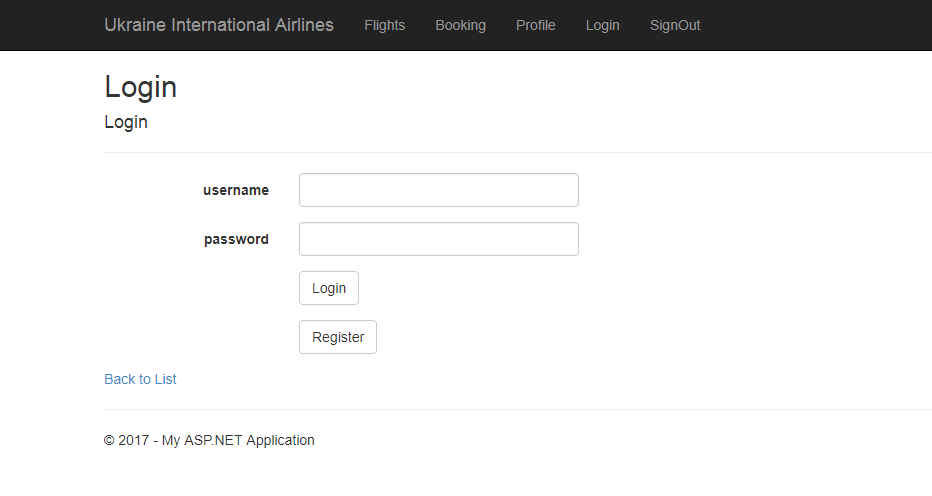


Figure 12: Login Screenshot

### Register Page

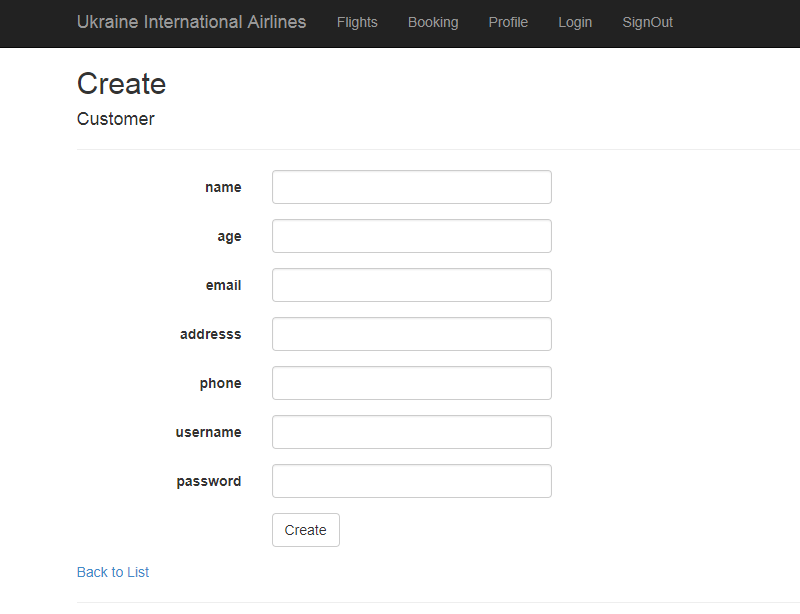


Figure 13: Register Screenshot

### Flight View Page

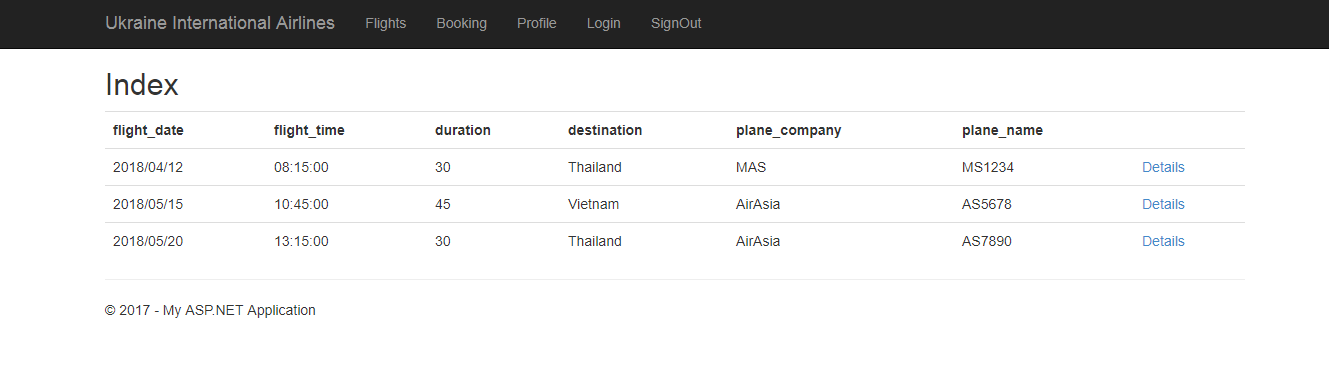


Figure 14: Flight Page Screenshot

### Booking View Page

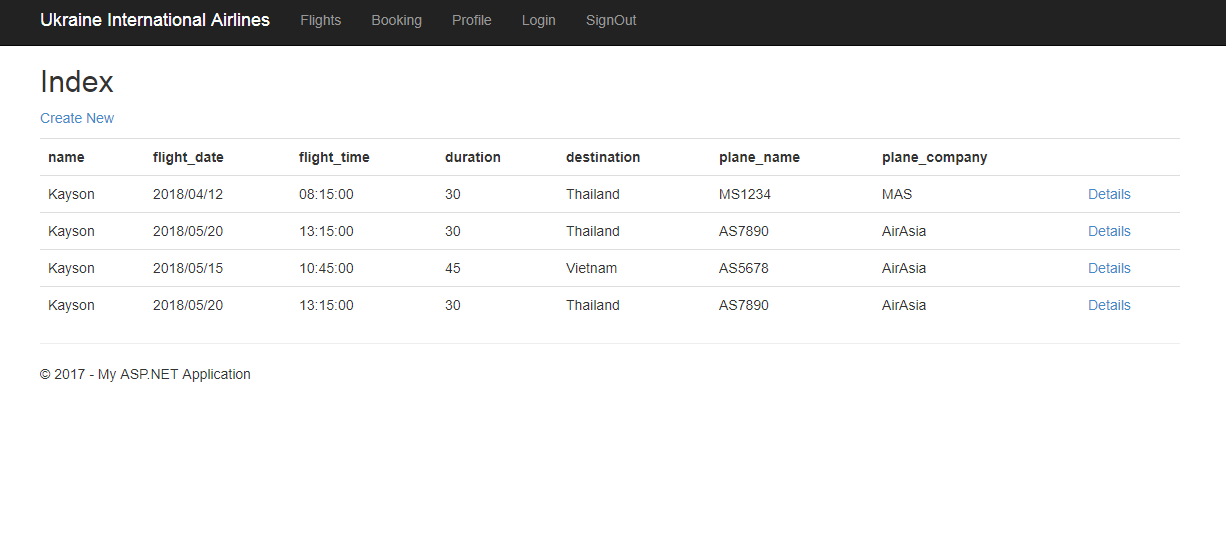


Figure 15: Booking Page Screenshot

### Profile View Page

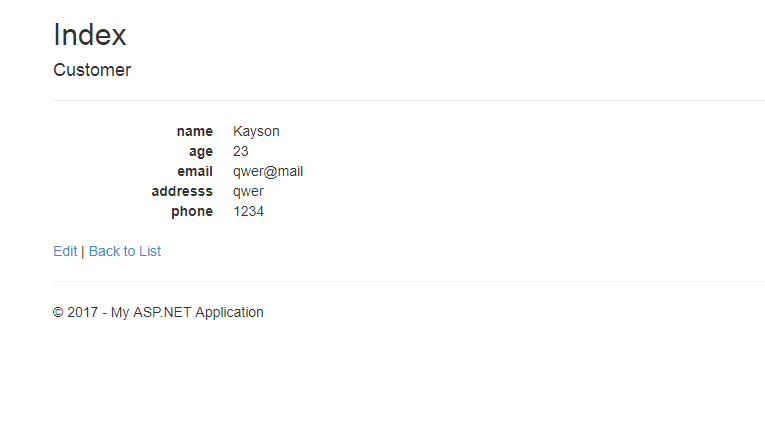


Figure 16: Profile Page Screenshot

### Edit Profile Page

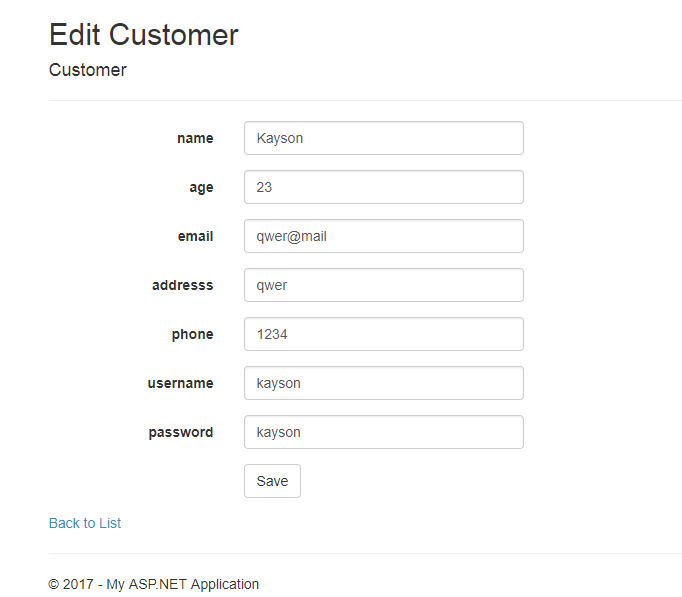


Figure 17: Edit Profile Page Screenshot

## Create and migrate SQL database

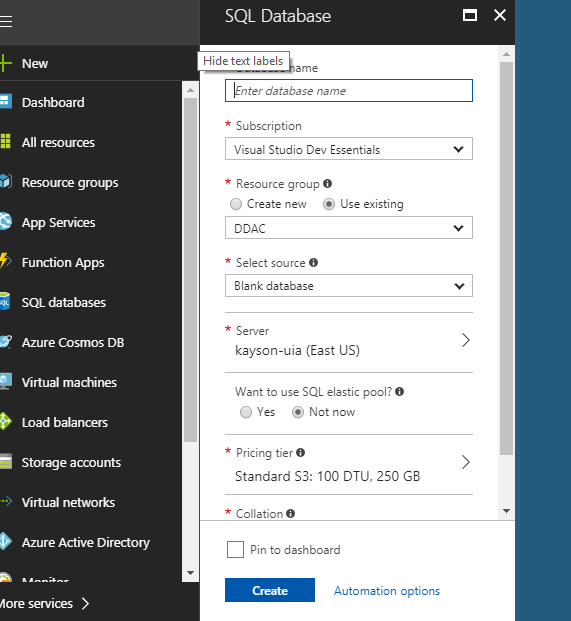


Figure 18: Create and Migrate SQL Database Screenshot

Step 1: Before created a Web SQL, a resource group is needed to create first. After that, click on Add and select SQL Database. Next fill in the database name, select resource group that created, create a new server with fill server name, user name, password and location, and select pricing tier (Standard S0: 10 DTU, 250GB), then click create. After created, click on it can copy the server name of the database.

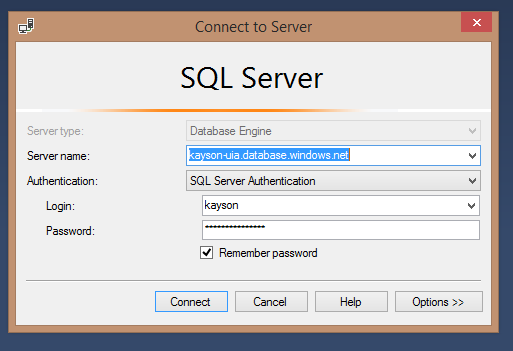


Figure 19: Create and Migrate SQL Database Screenshot 2

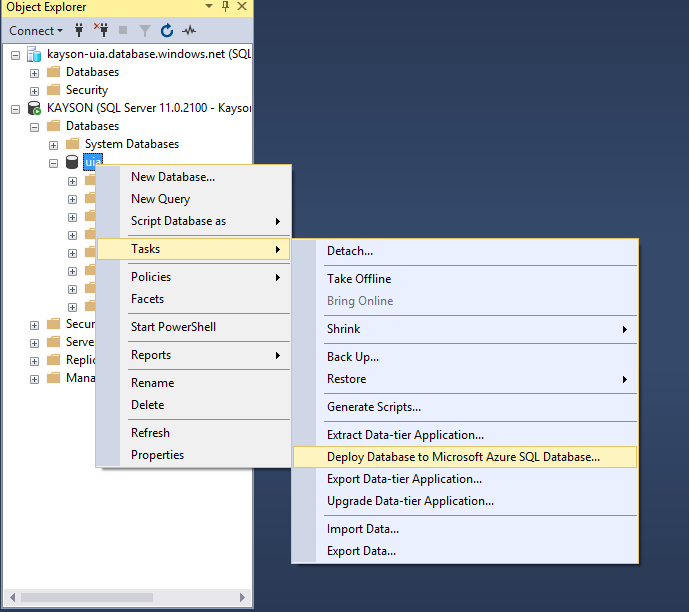
Step 2: Open Microsoft SQL Server Management Studio, the click connect and paste the server name into the textbox and choose SQL Server Authentication with fill in the username and password of server, then click connect.

Figure 20: Create and Migrate SQL Database Screenshot 3

Step 3: Then right click on local database, then select Tasks and select Deploy Database to Microsoft Azure SQL Database.

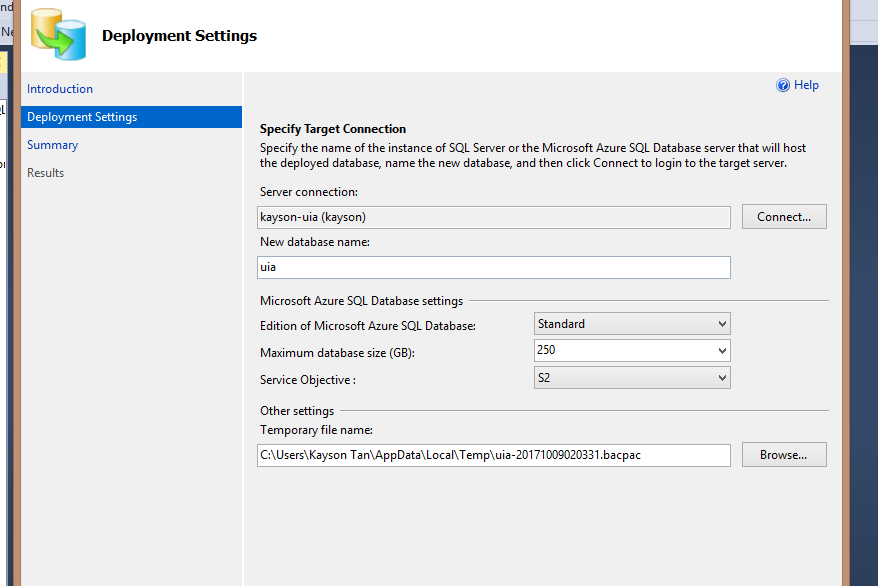


Figure 21: Create and Migrate SQL Database Screenshot 4

Step 4: In the pop up window, click connect and connect to server that create in Azure. Then select Edition of Microsoft Azure SQL Database to Standard. Then click Next and wait the database to migrate to new database.

## Deployment Application to Microsoft Azure

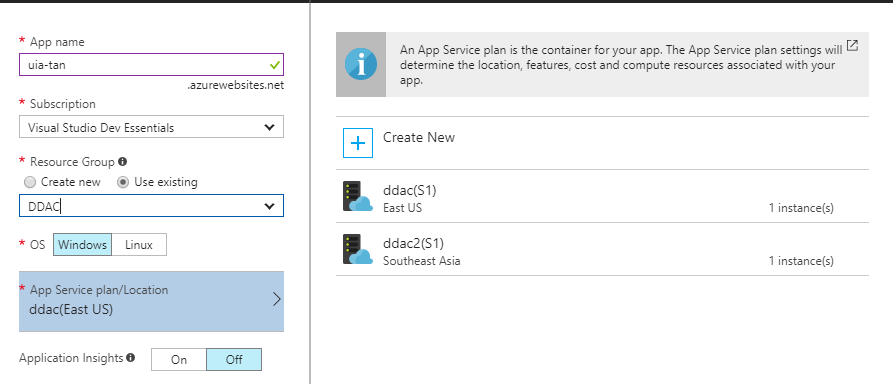


Figure 22: Deployment of Web Application Screenshot

Step 1: A GitHub account should registered and the project of system should uploaded to the GitHub. On Microsoft Azure click Add then select Web App. Next, fill in App name and select Resource group. Then, click on App Service plan and create an App Service plan with fill in the App Service plan name and select Location.

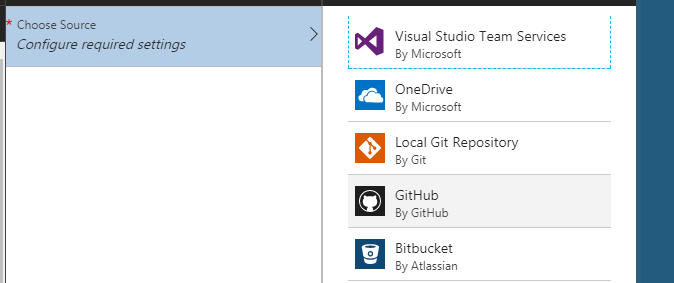


Figure 23: Deployment of Web Application Screenshot 2

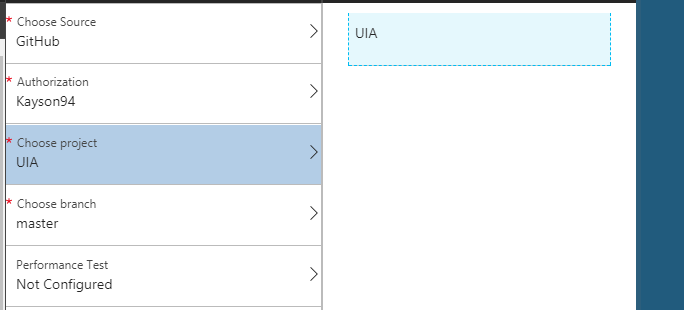


Figure 24: Deployment of Web Application Screenshot 3

Step 2: Click the Web App that just created and select Deployment options in the Deployment. Then click Choose Source and select GitHub, then select project and click OK.

## Traffic Manager

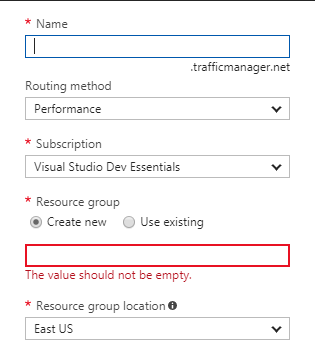


Figure 25: Traffic Manager Screenshot

Step 1: Two Web app are created with different region and resource group. Then click Add and search for Traffic Manager Profile to create. Next, fill in the name and select Resource group.

Step 2: Click on the Traffic Manager Profile and select Endpoints that in the Setting.

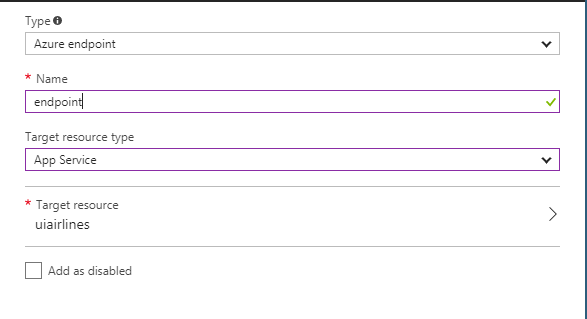


Figure 26: Traffic Manager Screenshot 2

Step 3: Then click add. Fill in the name, select Target resource type to App Service and select the Web App in the Target resource. Then click OK.

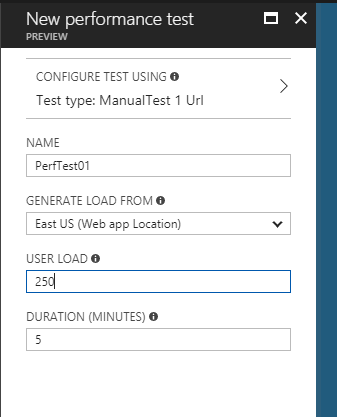
Step 4: Repeat Step 3 for the second Web App.

# Testing

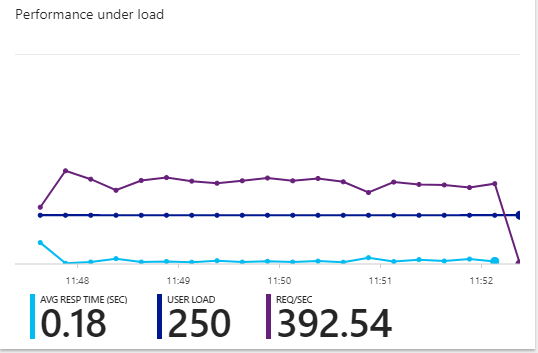
## Unit Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Function | Test Case ID | Test Condition | Description | Expected Result | Actual Result |
| Login | 1.0 | Username: kayson  Password: kayson | This test case is to test the login function. System need verify the username and password that user insert. | Login should be success if that username and password is valid. | As Expected |
| Register | 2.0 | Name: Kok Seng  Age: 23  Email: kokseng@mail.com  Address:  1, jln1, tmn 1  Phone: 123456890  Username: kokseng  Password: 12345678 | User can insert the valid value in all the field in order to register an account in the system. System should verify the inputs of the user and insert them into database. | Register should be success if all the values are valid. | As Expected |
| Flights Details | 3.0 | Select a flight. | System should display the selected flight details. | Selected flight details displayed. | As Expected |
| Booking | 4.0 | Logged in | System should display all the booking of the user. | All booking of the user displayed. | As Expected |
| Booking Details | 5.0 | Select a booking | System should display the selected booking details. | Selected booking details displayed. | As Expected |
| View Profile | 6.0 | Logged in | System will display the details of user | User details was displayed. | As Expected |
| Edit Profile | 7.0 | Name: tan  Age:23  Email: tan@mail.com  Address:  2, jln2, tmn 2  Phone  0987654321  Username:  tan  Password:  1234 | User can edit the value in all the field System should verify the inputs of the user and update the database. | Profile updated. | As Expected |

## Performance Test



Firstly, a new performance test will be created in the web app to perform performance test.



The graph above shows the average response time of the web app is 0.18 second in condition of 392.54 request per second.

# Conclusion

In conclusion, a web application for Ukraine International Airlines (UIA) has developed to let the users manage the details efficiently. The web application has developed to satisfy the requirements defined by Ukraine International Airlines (UIA) to host the developed web application on Microsoft Azure as App Service has made to meet the needs of Dmitriy Prudnikov. Other than that, by using the cloud services provided by Microsoft Azure is able to accelerate the setup and deployment of the solution. For example, a new server will take approximately 8~16 weeks to deploy in an on-premise environment. However, it only takes few minutes to deploy in the cloud computing environment.

Furthermore, Azure Traffic Manager is implemented to control request from web clients. Performance traffic routing is also done by deploying endpoints in two or more locations across the globe. It will help to improve the responsiveness of the web application by routing traffic to the location that is closest to the client. Lastly, performance test is also done to ensure the web application is capable to handle certain user load and make sure it’s up for scalability in the near the future without impacting the performance. In this assignment, the developer had gained valuable experience and knowledge of hosting web application on Microsoft Azure as App Service.

# References

Cloud Computing Design Patterns. (2017). *Cloud Computing Design Patterns*. [online] Available at: http://www.cloudpatterns.org/ [Accessed 9 Oct. 2017].

Docs.microsoft.com. (2017). *Azure Web Apps Documentation - Tutorials, API Reference*. [online] Available at: https://docs.microsoft.com/en-us/azure/app-service/ [Accessed 9 Oct. 2017].

Docs.microsoft.com. (2017). *Cloud Design Patterns*. [online] Available at: https://docs.microsoft.com/en-us/azure/architecture/patterns/ [Accessed 9 Oct. 2017].

Docs.microsoft.com. (2017). *Create an ASP.NET web app in Azure*. [online] Available at: https://docs.microsoft.com/en-us/azure/app-service/app-service-web-get-started-dotnet [Accessed 9 Oct. 2017].

Docs.microsoft.com. (2017). *Migrate SQL Server DB to Azure SQL Database*. [online] Available at: https://docs.microsoft.com/en-us/azure/sql-database/sql-database-migrate-your-sql-server-database [Accessed 9 Oct. 2017].

Docs.microsoft.com. (2017). *Monitor Azure web app performance*. [online] Available at: https://docs.microsoft.com/en-us/azure/application-insights/app-insights-azure-web-apps [Accessed 9 Oct. 2017].

Docs.microsoft.com. (2017). *Test your Azure web app performance under load from the Azure portal*. [online] Available at: https://docs.microsoft.com/en-us/vsts/load-test/app-service-web-app-performance-test [Accessed 9 Oct. 2017].

Google Cloud Platform. (2017). *Cloud Architect Certification  |  Google Cloud Certifications  |  Google Cloud Platform*. [online] Available at: https://cloud.google.com/certification/cloud-architect [Accessed 9 Oct. 2017].

SearchCloudComputing. (2017). *What is cloud architect? - Definition from WhatIs.com*. [online] Available at: http://searchcloudcomputing.techtarget.com/definition/cloud-architect [Accessed 9 Oct. 2017].

www.tutorialspoint.com. (2017). *Cloud Computing Architecture*. [online] Available at: https://www.tutorialspoint.com/cloud\_computing/cloud\_computing\_architecture.htm [Accessed 9 Oct. 2017].