# ACKNOWLEDGEMENT

I would like to express my special thanks to my lecturer, Dr Kalai Anand A/L Ratnan for all his patience, motivation and immense knowledge. His guidance helped me in all the time of research and developing this Online Ticket Booking System for Ukraine International Airlines (UIA). Below is the account credential which related to this project:

Online Ticket booking System for Ukraine International Airlines (UIA). Website is available at: http://onlineticket20171119110517.azurewebsites.net/ (Southeast Asia).

The user can directly sign up to get the credential to access the system, but I already prepare one account that can be used to access the system.

Email : [rio\_afiando@yahoo.com](mailto:rio_afiando@yahoo.com)

Password: rioal

# Table of Contents

[ACKNOWLEDGEMENT 1](#_Toc498908991)

[Table of Contents 2](#_Toc498908992)

[1.0 INTRODUCTION 4](#_Toc498908993)

[1.1 Company Background 4](#_Toc498908994)

[1.2 Aims 4](#_Toc498908995)

[1.3 Objectives 4](#_Toc498908996)

[1.4 Scopes 5](#_Toc498908997)

[1.5 Goals and Requirements 5](#_Toc498908998)

[2.0 PROJECT PLAN 6](#_Toc498908999)

[2.1 Gantt Chart 6](#_Toc498909000)

[3.0 DESIGN 7](#_Toc498909001)

[3.1 Architectural Diagram 7](#_Toc498909002)

[3.2 Use Case Diagram 8](#_Toc498909003)

[3.3 Use Case Specification 9](#_Toc498909004)

[3.3.1 Login 9](#_Toc498909005)

[3.3.2 Logout 9](#_Toc498909006)

[3.3.3 Registration 10](#_Toc498909007)

[3.3.4 Modify Profile 10](#_Toc498909008)

[3.3.5 View Passenger Flight Schedule 11](#_Toc498909009)

[3.3.6 View Cargo Flight Schedule 11](#_Toc498909010)

[3.4 Activity Diagram 12](#_Toc498909011)

[3.4.1 Login 12](#_Toc498909012)

[3.4.2 Registration 12](#_Toc498909013)

[3.4.3 Modify Profile 13](#_Toc498909014)

[3.5 Design Considerations 14](#_Toc498909015)

[3.6 Modelling 14](#_Toc498909016)

[4.0 IMPLEMENTATION 15](#_Toc498909017)

[4.1 Graphical User Interface 15](#_Toc498909018)

[4.2 Database Migrations 19](#_Toc498909019)

[4.3 Development Steps from Local to Azure 20](#_Toc498909020)

[4.4 Update Database with Database Migration Technology 29](#_Toc498909021)

[5.0 TESTING 33](#_Toc498909022)

[5.1 Functional 33](#_Toc498909023)

[5.2 Performance 35](#_Toc498909024)

[6.0 CONCLUSION 36](#_Toc498909025)

[REFERENCES 37](#_Toc498909026)

[APPENDIX 38](#_Toc498909027)

# INTRODUCTION

## Company Background

Ukraine International Airlines (UIA) is the flagship carrier and largest airline in Ukraine. The Ukraine International Airlines (UIA) is operating domestic and international passenger flights, not only for passenger Ukraine International Airlines (UIA) also offer the service for cargo services to Europe, Middle East, United States and Asia. This Airlines have a will to expand their market, but they have problem with the website development for their customers that beyond Ukraine. The site experienced by Ukraine International Airlines (UIA) 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.

Ukraine International Airlines (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 analysing fuel economy. The Ukraine International Airlines (UIA) decide that it once again needed to innovate its way out of its web challenges. One of the executive board of Ukraine International Airlines (UIA) which is the Chief Information Officer (CIO) named Dmitriy Prudnikov, realized that migrating the websites out of Ukraine International Airlines (UIA) data centres into a public cloud might solve all these problems.

## Aims

Provides a Web Application for Ukraine International Airlines (UIA) into public cloud platform to serve their customers that beyond Ukraine and to prevent the denial-of-service (DOS) attacks, which hurt site performance and reality, and it did not have the performance needed to host visitors from many parts of the world.

## Objectives

To achieve the aims of this project, there are several things need to be done by the developer’s teams that will be listed in this objective session:

* To provide a system that could manage the airlines schedule.
* To provide a system that could show the flight schedule or list that can be seen by the users.
* To provide a system that only can be accessed by those who have the account to access the system only.

## Scopes

The scope of this project only to give the access to the user to access the system and to show the list of flight schedule, destination and origin of flight to the user of the system. The system that will be developed would not focus on the how the user book the flight, how user pay the price of the ticket.

## Goals and Requirements

There are mandatory goals and requirements should be considered in doing the project:

* ***Provisioning***

The developer must be able to provision the new application to the Microsoft Azure Platform.

* ***Maintainability***

The developer must be able to upgrade the application and perform other maintenance tasks while multiple tenants are using it.

* ***Monitoring***

The developer must be able to monitor the application at all times to identify any problems and to troubleshoot them. This includes monitoring how each tenant is using the application.

* ***Availability***

UIA want the application to be constantly available, perhaps with guarantees defined in an SLA. Again, the activities of other tenants should not affect the availability of the application.

* ***Scalability***

The application scales to meet the demand of the application.

# PROJECT PLAN

## Gantt Chart

|  |  |  |  |
| --- | --- | --- | --- |
| **TASK NAME** | **DURATION** | **START** | **FINISH** |
| **Ukraine international Airlines (UIA)**  **Online Ticket Booking System** | **79 Days** | **Fri 09/01/17** | **Sat 11/18/17** |
| **Introduction** | **6 Days** | **Fri 09/01/17** | **Wed 09/06/17** |
| Company Background | 1 Day | Fri 09/01/17 | Fri 09/01/17 |
| Aims | 1 Day | Sat 09/02/17 | Sat 09/02/17 |
| Objectives | 1 Day | Sun 09/03/17 | Sun 09/03/17 |
| Scopes | 2 Days | Mon 09/04/17 | Tue 09/05/17 |
| Goals and Requirements | 1 Day | Wed 09/06/17 | Wed 09/06/17 |
| **Project Plan** | **2 Days** | **Thu 09/07/17** | **Fri 09/08/17** |
| Gantt Chart | 2 Days | Thu 09/07/17 | Fri 09/08/17 |
| **Design** | **24 Days** | **Sat 09/09/17** | **Mon 10/02/17** |
| Architectural Diagram | 4 Days | Sat 09/09/17 | Tue 09/12/17 |
| Use Case Diagram | 5 Days | Wed 09/13/17 | Sun 09/17/17 |
| Use Case Specification | 5 Days | Mon 09/18/17 | Fri 09/22/17 |
| Activity Diagram | 5 Days | Sat 09/23/17 | Wed 09/27/17 |
| Design Considerations | 3 Days | Thu 09/28/17 | Sat 09/30/17 |
| Modelling | 2 Days | Sun 10/01/17 | Mon 10/02/17 |
| **Implementation** | **40 Days** | **Tue 10/03/17** | **Sat 11/17/17** |
| Graphical User Interface | 20 Days | Tue 10/03/17 | Sun 10/22/17 |
| Deploy from Local to Azure | 20 Days | Mon 10/23/17 | Sat 11/11/17 |
| **Testing** | **6 Days** | **Sun 11/12/17** | **Fri 11/17/17** |
| Functional | 3 Days | Sun 11/12/17 | Tue 11/14/17 |
| Performance | 3 Days | Wed 11/15/17 | Fri 11/17/17 |
| **Conclusion** | **1 Day** | **Sat 11/18/2017** | **Sat 11/18/17** |

# DESIGN

## Architectural Diagram

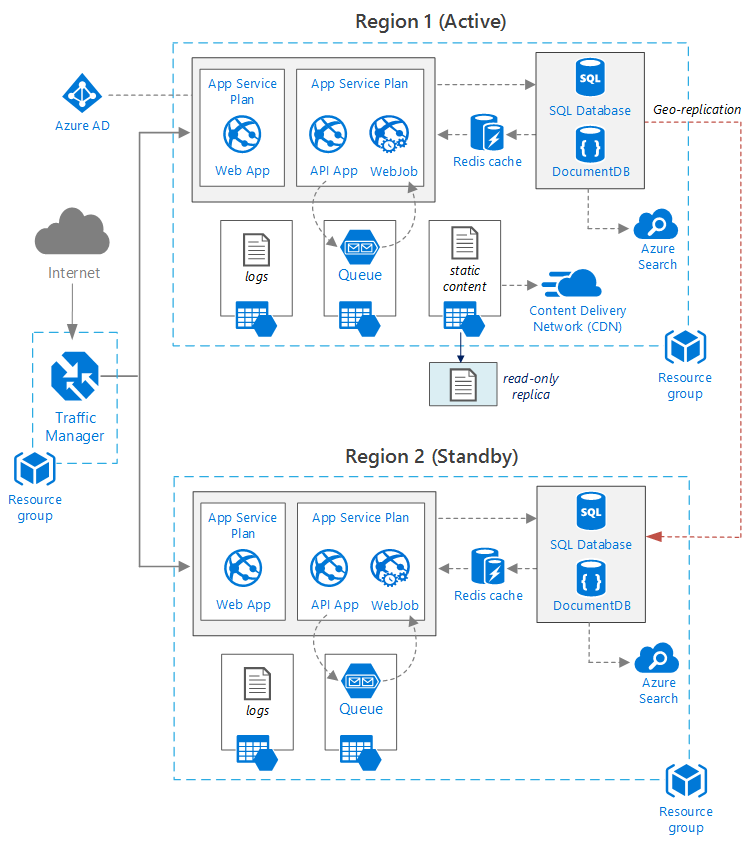


Figure Architectural Diagram

## Use Case Diagram

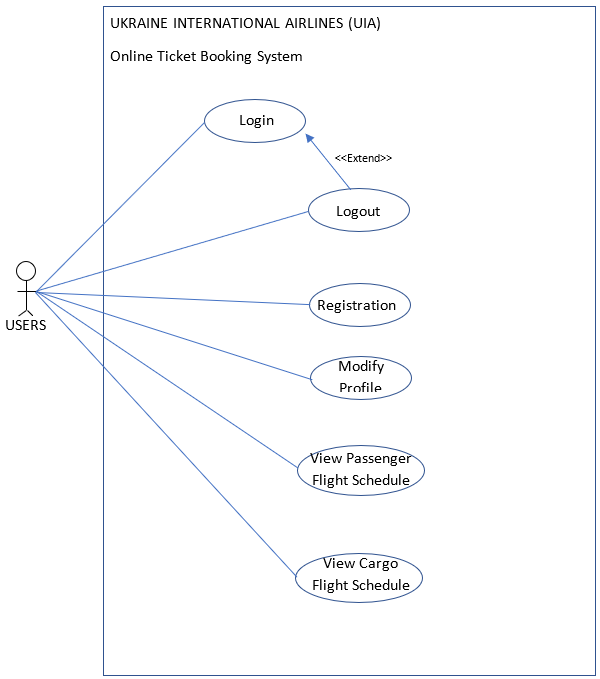


Figure Use Case Diagram

## Use Case Specification

### Login

|  |  |
| --- | --- |
| **Name** | **Login Process** |
| Actor | Users |
| Priority | High |
| Description | Login to the system |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | - User has been registered as user through the application |
| Post-condition | Proceed to main menu |
| Flow of Events | 1. Input username 2. Input password 3. If the username and password is not valid [A1] |
| Alternative Flows | 1. A1. User must re-input his/her username and password correctly |

### Logout

|  |  |
| --- | --- |
| **Name** | **Logout Process** |
| Actor | Users |
| Priority | High |
| Description | Logout from the system |
| Extends | Login |
| Includes | None |
| Pre-condition / Assumptions | - User has been login to the system |
| Post-condition | Proceed to Login Page |
| Flow of Events | * + - 1. Press Logout Button |
| Alternative Flows | None |

### Registration

|  |  |
| --- | --- |
| **Name** | **Register Process** |
| Actor | Users |
| Priority | High |
| Description | People want to have an account to access the application |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | None |
| Post-condition | People can access the system with the username and password that they set during the registration |
| Flow of Events | 1. Click Sign Up Link 2. Input All the Data Needed 3. If the Input is Not valid [A1] 4. If valid the user need to do the first-time login |
| Alternative Flows | A1. User must re-input his/her profile correctly |

### Modify Profile

|  |  |
| --- | --- |
| **Name** | **Modify Profile Process** |
| Actor | Users |
| Priority | High |
| Description | User want to modify their profile |
| Extends | None |
| Pre-condition / Assumptions | Login to the system |
| Post-condition | Profile Updated |
| Flow of Events | 1. Click modify Profile link 2. Modify data that need to be changed 3. If the Input is not valid [A1] 4. If Valid user will proceed to profile page. |
| Alternative Flows | A1. User must re-input his/her profile correctly |

### View Passenger Flight Schedule

|  |  |
| --- | --- |
| **Name** | **View Passenger Flight Schedule** |
| Actor | Users |
| Priority | High |
| Description | User want to see the flight schedule for passenger. |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | Login to the system |
| Post-condition | Display the flight schedule for passenger. |
| Flow of Events | 1. Click passenger flight schedule link 2. View the flight schedule for passenger |
| Alternative Flows | None |

### View Cargo Flight Schedule

|  |  |
| --- | --- |
| **Name** | **View Cargo Flight Schedule** |
| Actor | Users |
| Priority | High |
| Description | User want to see the flight schedule for cargo |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | Login to the system |
| Post-condition | Display the flight schedule for cargo |
| Flow of Events | 1. Click passenger flight schedule link 2. View the flight schedule for cargo |
| Alternative Flows | None |

## Activity Diagram

### Login

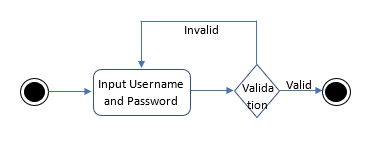


Figure Login Activity Diagram

### Registration

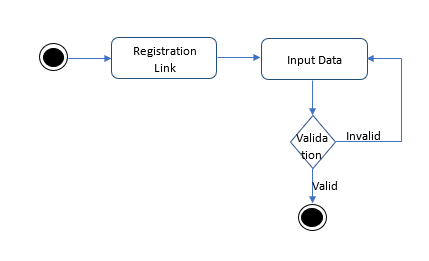


Figure Registration Activity Diagram

### Modify Profile

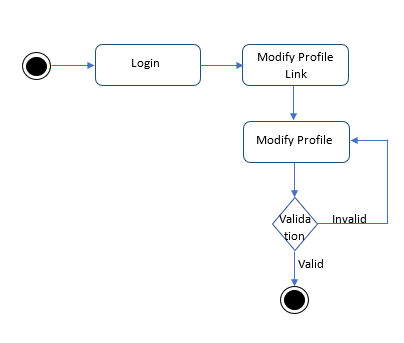


Figure Modify Profile Activity Diagram

## Design Considerations

As for the design consideration, the developer using or implement the system database using entity framework, which is provides by the visual studio. Because it is easier rather than to write the sql query, since the focus of this project is on the deployment the system to the cloud platform, not about the system structure. So, by creating the entity framework for each entity, the model of every entity also will be automatically created, and the developer can easily generate the controller and the view for that entity. The model of the data design in this system will be shown in the ERD that will be shown in the modelling session below.

## Modelling

Figure Entity Relationship Diagram (ERD)

The figure above shows the Entity Relationship Diagram of the system, to show the entity sets stored in the database ***(SmartDraw, 2015)***. The user table will be the place to store all the details or data about the users of the system, while the other two tables will be the place to store the flight schedule details.

# IMPLEMENTATION

## Graphical User Interface

In this section of documentation will focus on the implementation of the system, which in this section several screenshots of the graphical user interface of the system will be shown. And the information about every page of the system will be explain in this section.

* + - 1. **Login Page**

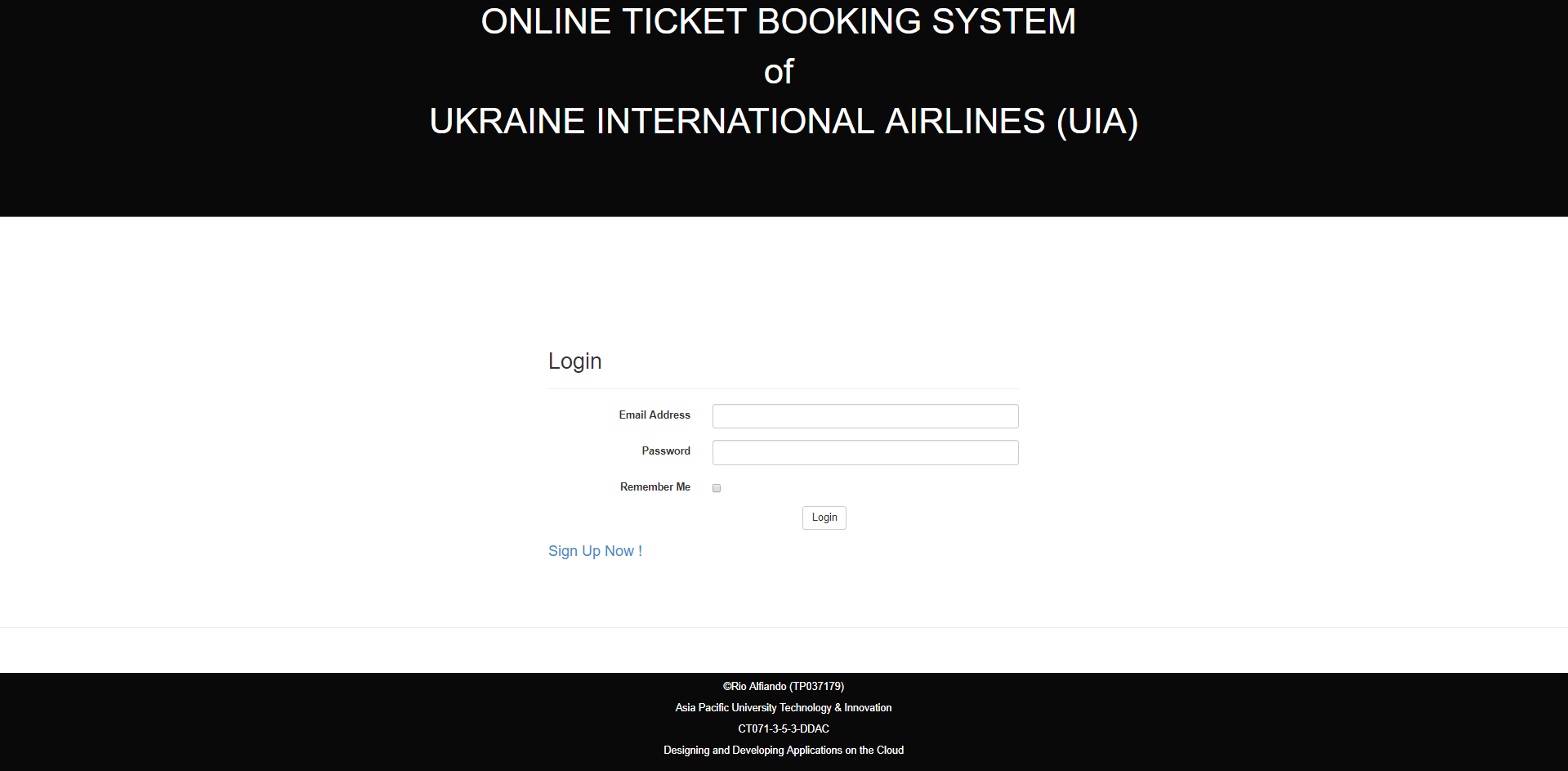
****The figure above showing the login page of the system for the user to access the system. If the users do not have the account yet, they can directly click sign up link, and they will go to the registration page, that is the only way for the user to have an access in to the system. Without doing the registration people cannot access the system, because of they do not have the credential to access the system.

Figure Login Page Interface

1. **Registration Page**

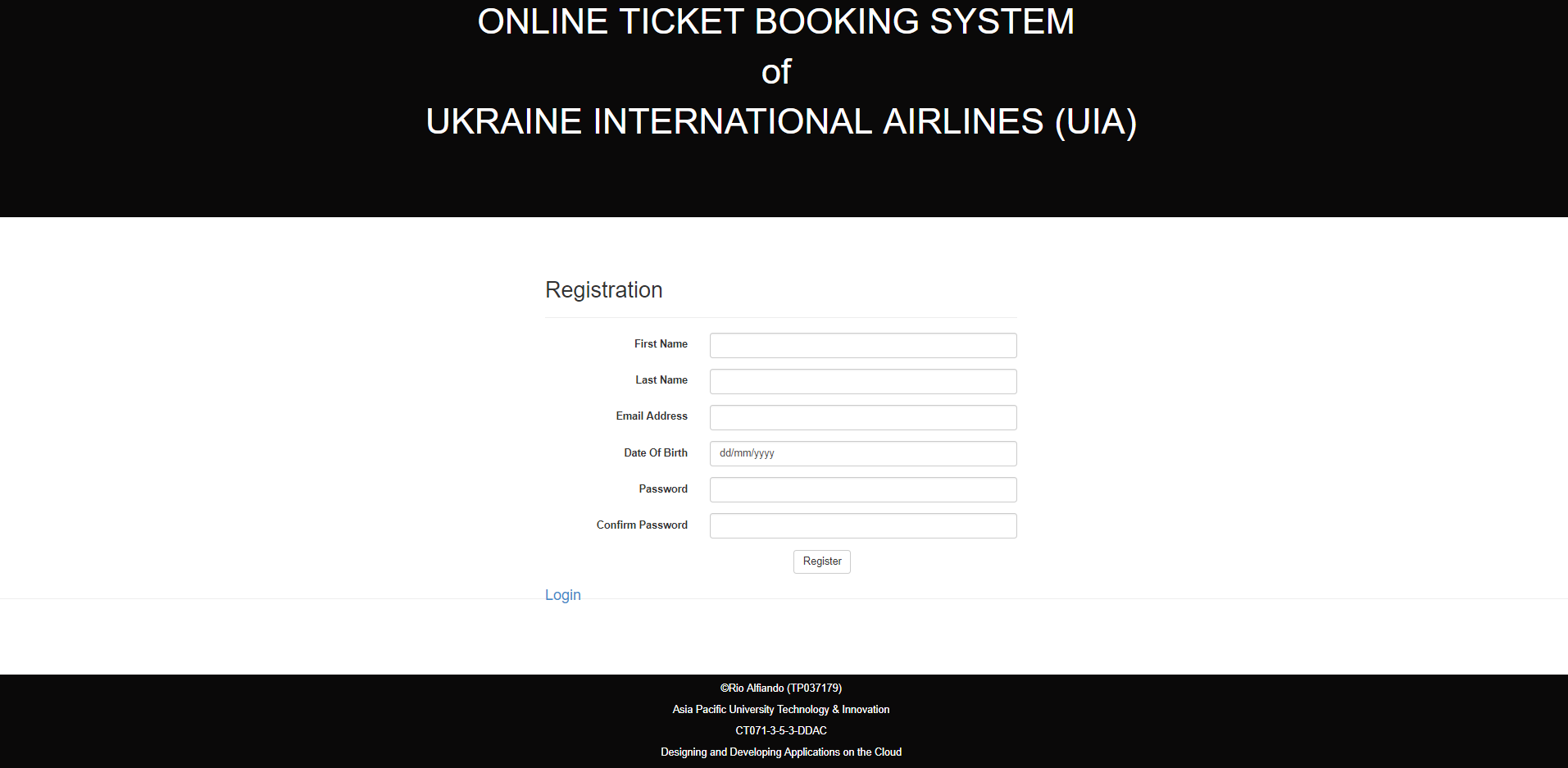
****

Figure Registration Page Interface

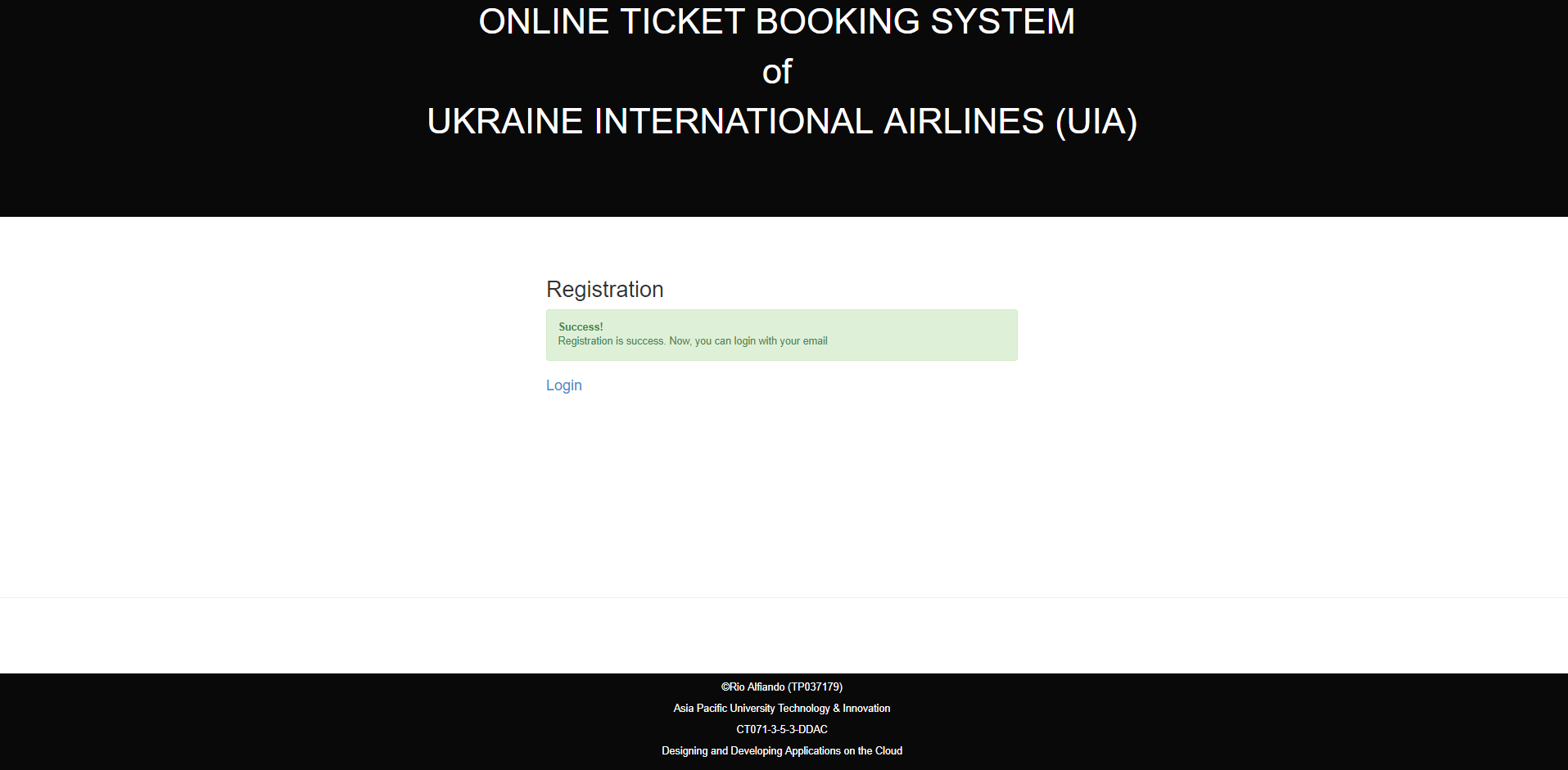
When the users click the sign-up link in the login page, they will go to the page that shown in the figure above. In this page, the developer also implements the validation for every field. So, the users cannot leave it blank, because all the data is needed in this page. After the users finish all the registration process, they can directly login with their email and the password that they input in this registration page. But, if the users already have the account or credential, they can directly login to the system, by clicking the login link in this registration page. If the registration is success, the system will show the success message like shown in the figure below.

Figure POST registration page

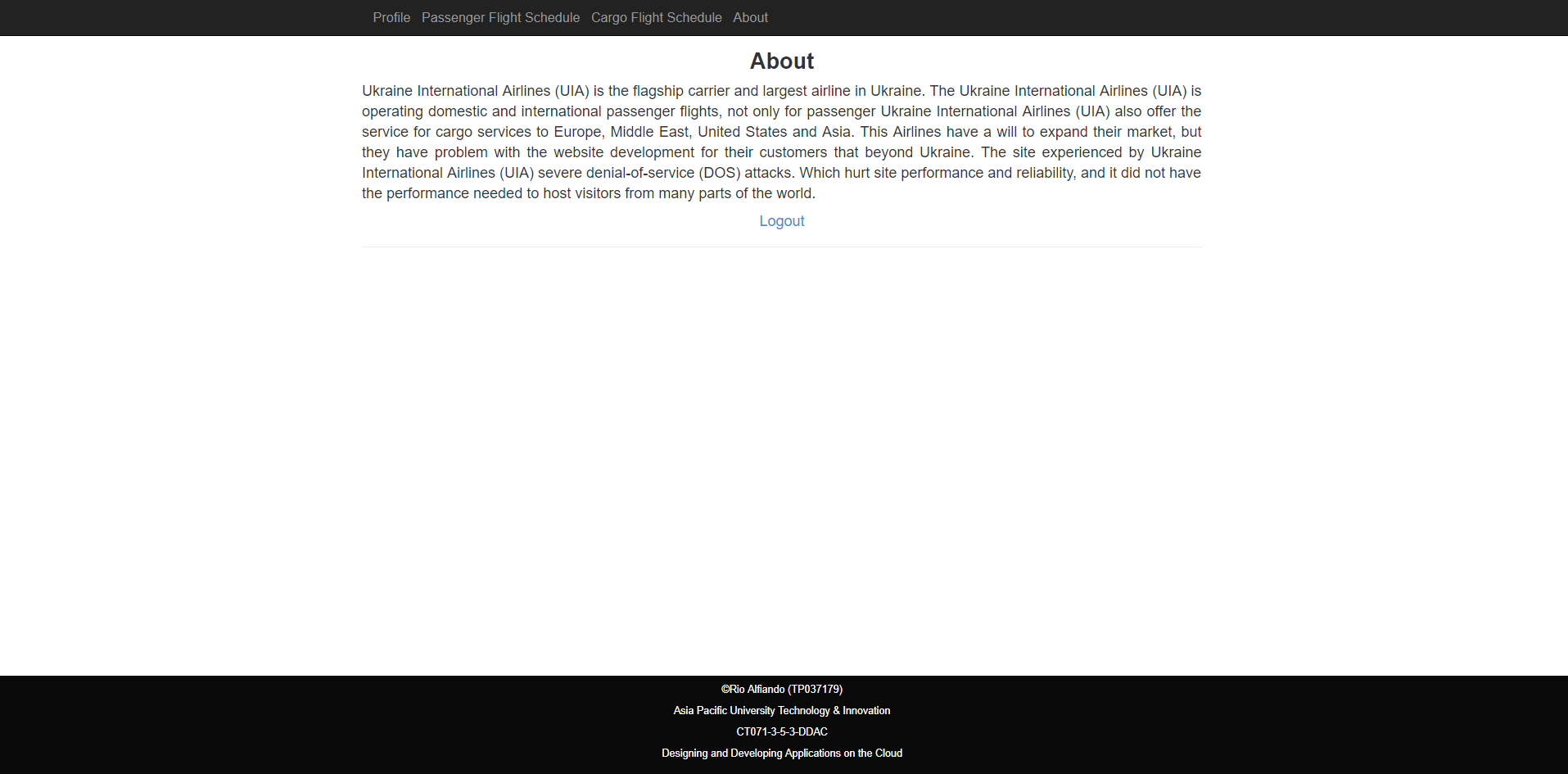
1. **About Page**

Figure About Page Interface

After the user successfully, login they will see the about page, which in this page the user will know about the background of the company. They can read all the information about the company here in this page.

1. **Cargo Flight Schedule**

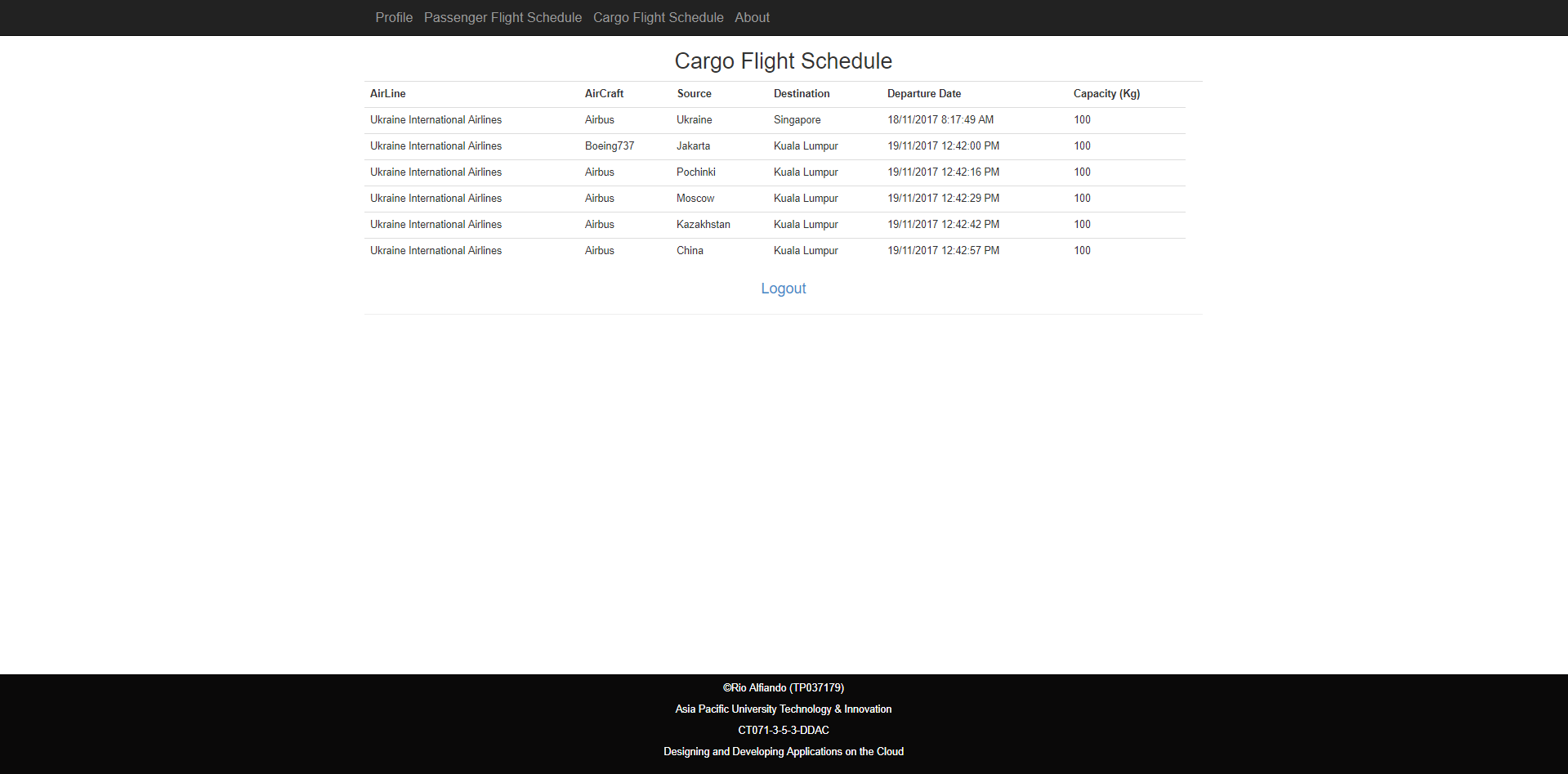
****

Figure Cargo Flight Schedule Interface

This is the page where users can see all the flight schedule for cargo, while they can see the flight schedule for the passenger in the other page that provides by the Ukraine International Airlines.

1. **Passenger Flight Schedule**

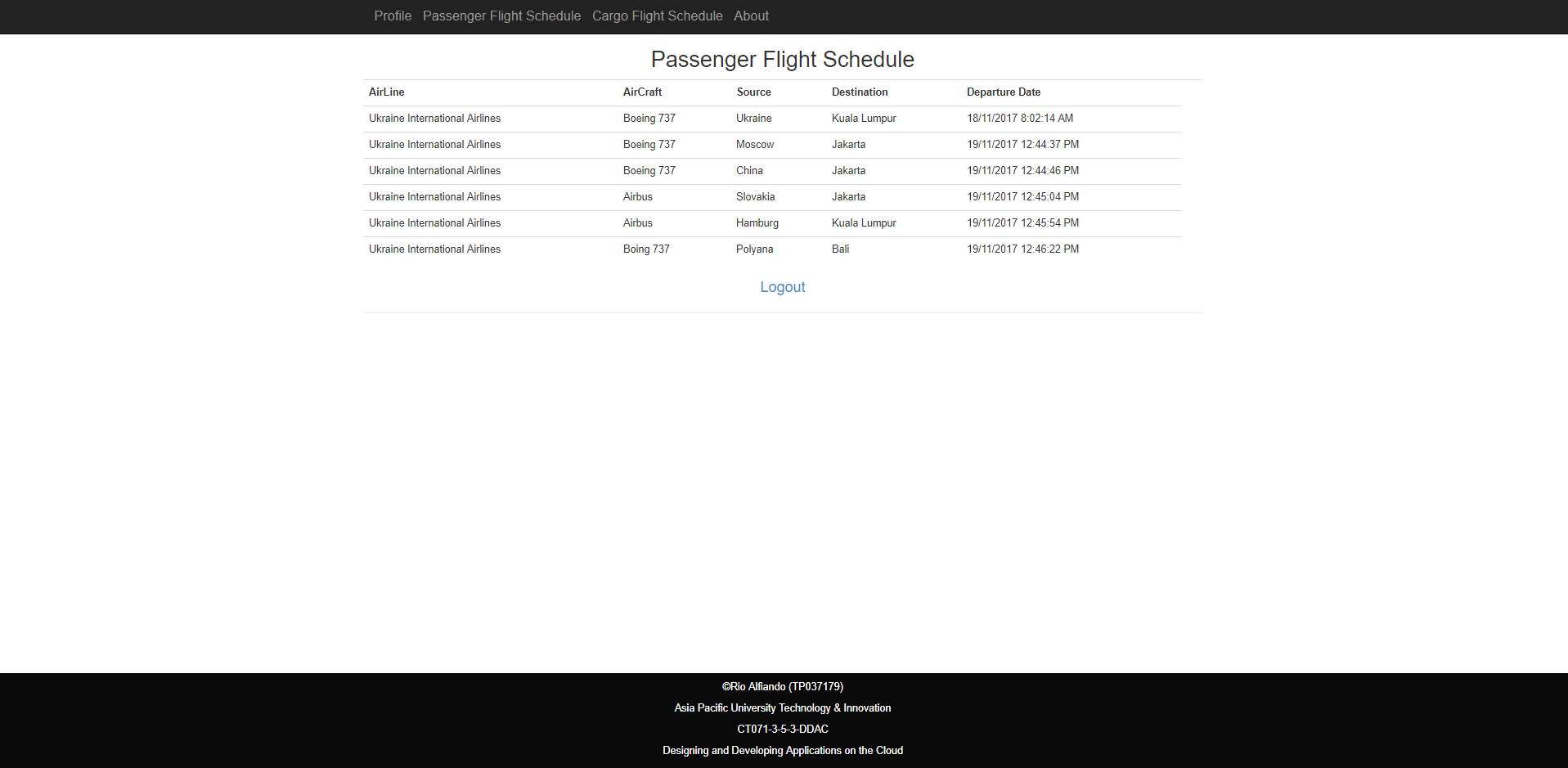
****

Figure Passenger Flight Schedule Interface

This is the page where users can see the flight schedule for the passengers.

1. **Modify Profiles**

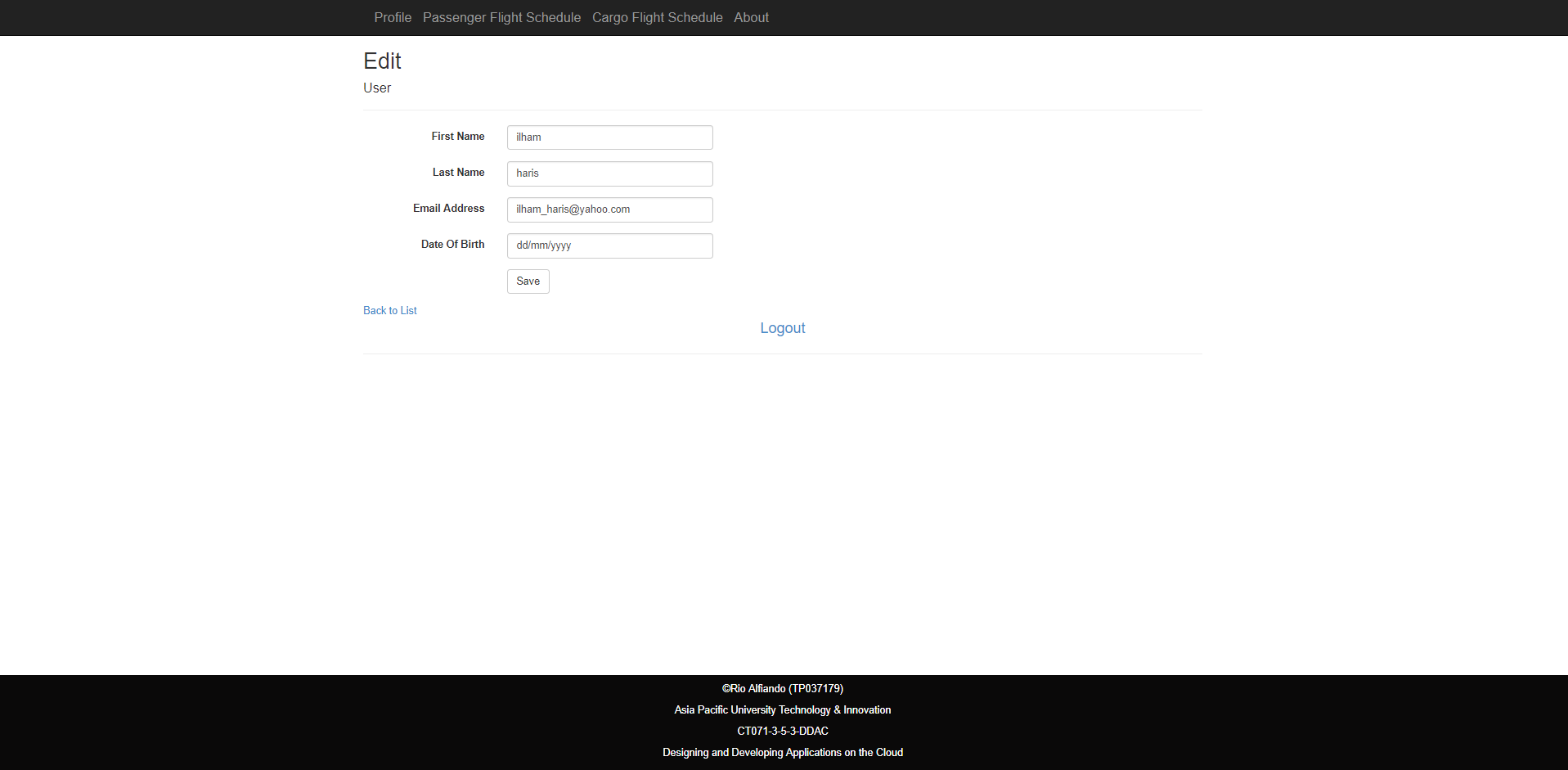
In this page the users can modify their profile, according what they want.

Figure Modify Profile Page

## Database Migrations

Database Migration also being implemented in the system, in order to create new database easier whenever the developer want to move the project from the local server to the cloud platform. What developer do to implement the database migration is using the NuGet Package Manager tools provided by the Visual Studio, to create the database migration files. The command line used to create the migration technology are as follows:

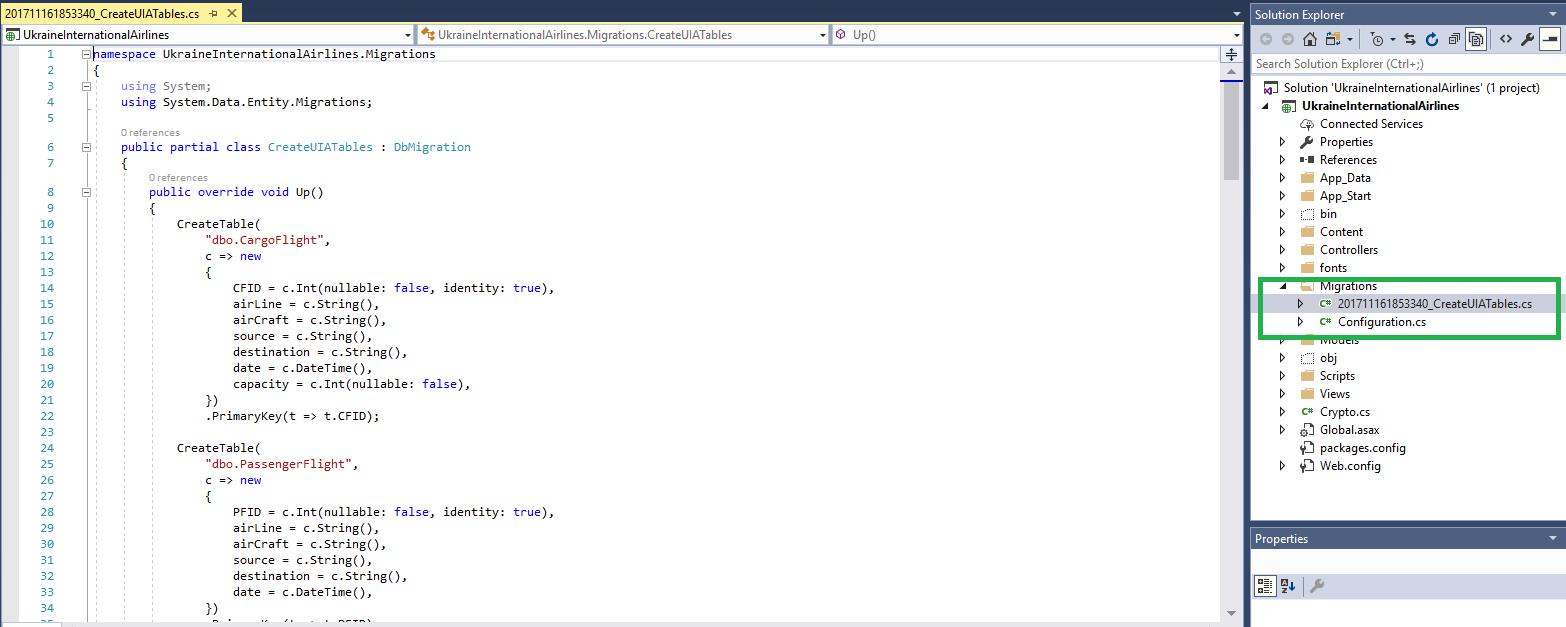
* + - 1. **Enable-migrations:** To enable the system to implement the migration technology, which the table structure can be implemented to another server later.
      2. **Add-migration CreateUIATables:** it will create the table structure query needed, so whenever the developer moves the system to new environment, the developer can directly update the database through NuGet Package Manager, and the developer will get all the table needed for this system.

Figure Database Migration

Like shown in the green box in the figure above, it’s all the migration files and configuration needed to create the table in the new environment in the future.

## Development Steps from Local to Azure

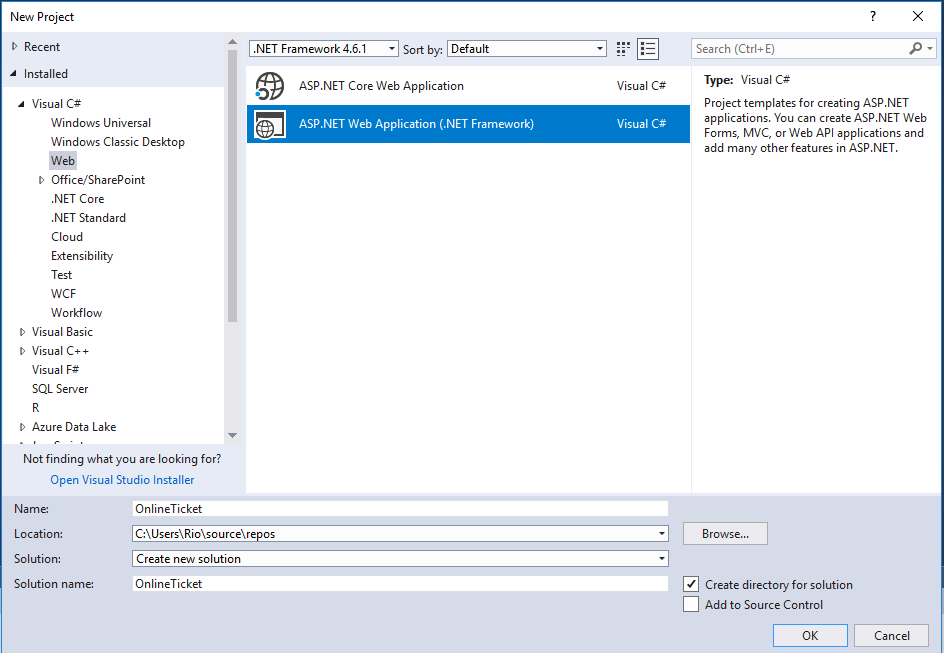
**STEP 1**

Figure Create New Project

To start the project the developer, need to have visual studio on their desktop first, to start developing the project and publish it to azure. To start the project, the developer can open their visual studio and then click **File** -> **Project -> New Project,** and then the pop-up windows like shown in the figure above will come outto give option to the developer what kind of project they want to build. In this case we want to develop a web application, so the developer need to choose the ASP.NET Web Application (.NET Framework). Finally the developer need to type in the name for the project and press OK.

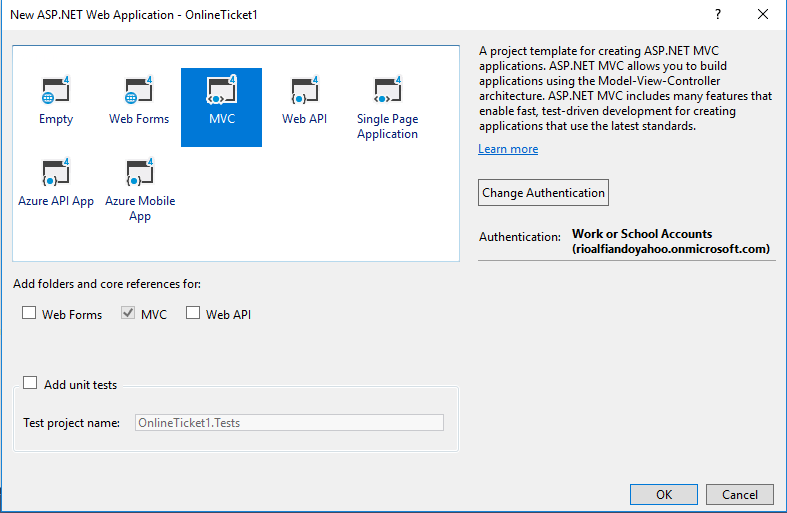
**STEP 2**

Figure Web Application templates or framework that developer wants to choose.

Once the developer has chosen the ASP.NET Web Application, the second popup menu will appear like shown in the figure above, and developer will choose MVC to develop the project. MVC stands for Model View Controller which separates the application to provide alternative to the ASP.NET Web Forms pattern for creating Web applications ***(Microsoft, 2017)***.

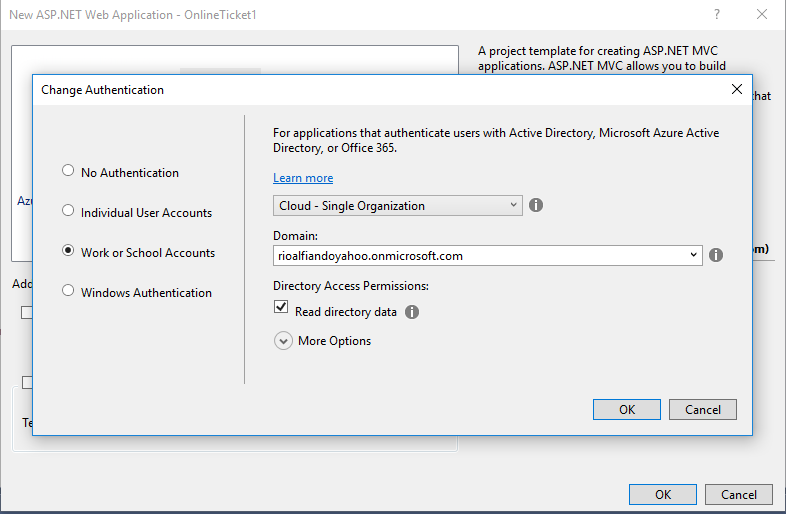
**STEP 3**

Figure Ading Azure Active Directory Authentication

Before creating the project, developer need to change the authentication method first into “Work or School Accounts” in order to implement Azure Active Directory authentication. If the developer has already logged in to visual studio using their account, the domain will be showed up by itself. On top of that, tick the “Read directory data” to grant directory access permission to user. Finally, new MVC can be created.

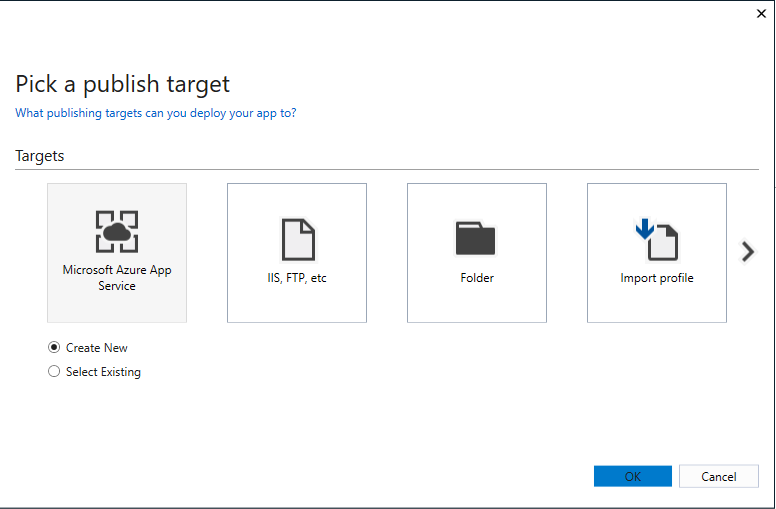
**STEP 4**

Figure Publishing to Azzure Application Service

Once the project is ready, developer now can publish his/her project into Azure App Service. This page can be viewed by right click on the project name and choose to publish. For this scenario, developer does not have existing app service, so he must create new service.

**STEP 5**

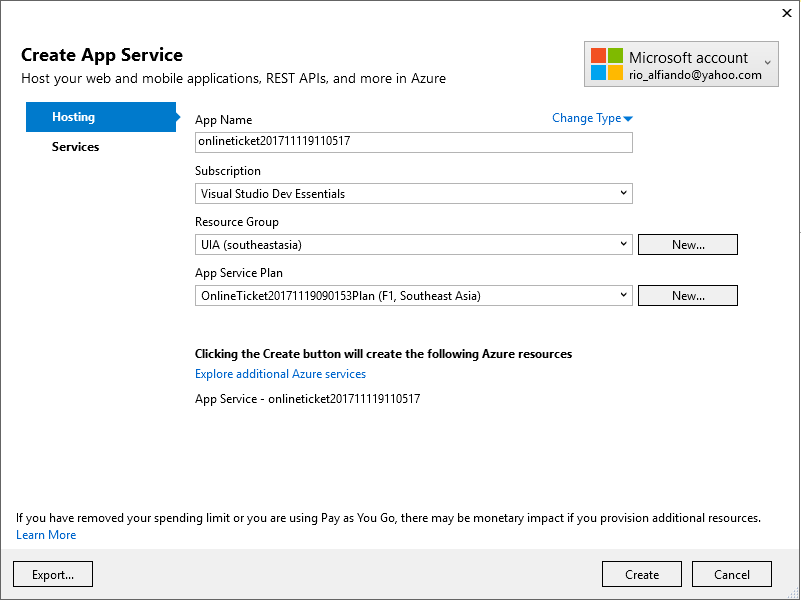
**** After choosing the Microsoft Azure App Service and click OK, the developer will see the page like shown the figure above where the user can fill up the App name that they want, which is will be used as an URL when the system is deployed. The developer also need to choose or create the App Service Plan they want to use, the region or location need to be set as well. After everything is fill up correctly, then developer will click the “Explore additional Azure Services” to create and configure the database that will be used to store the data.

Figure Application Service Configuration

**STEP 6**

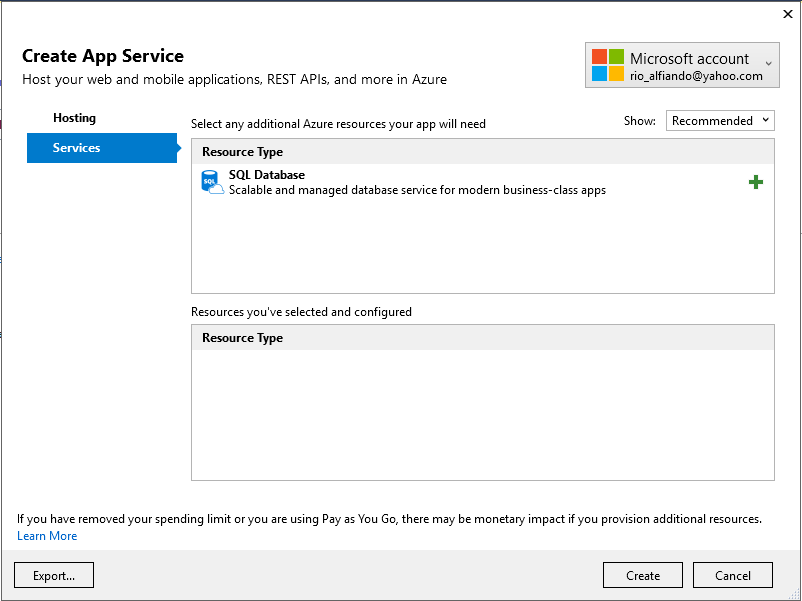
 SQL database is one of the services that provided by the Microsoft azure for the user to stored and retrieve data to the website. At this page the developer will click the + sign next to the SQL Database.

Figure SQL database configuration in Azure

**STEP 7**

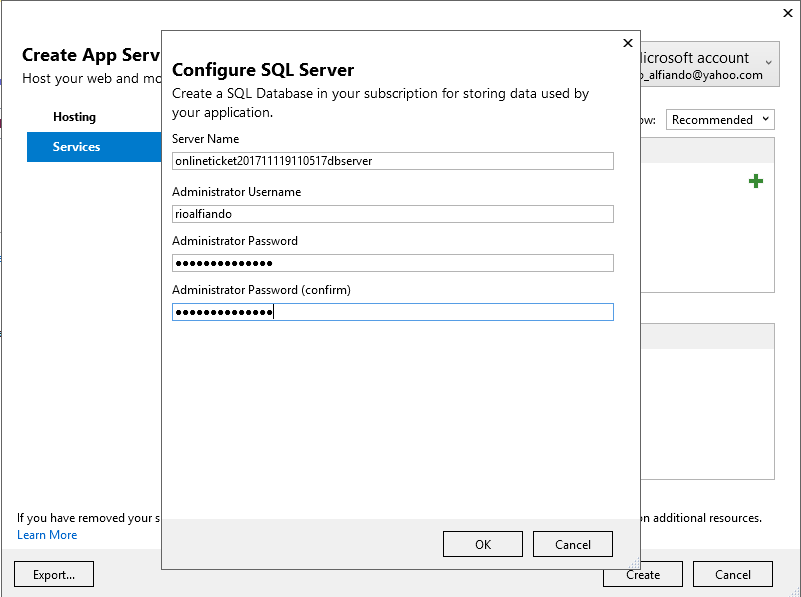
**** On this page the developer will create and configure the SQL server, which the developer will decide the server name, set the administrator username and the password for the database server.

Figure Create the SQL Server

**STEP 8**

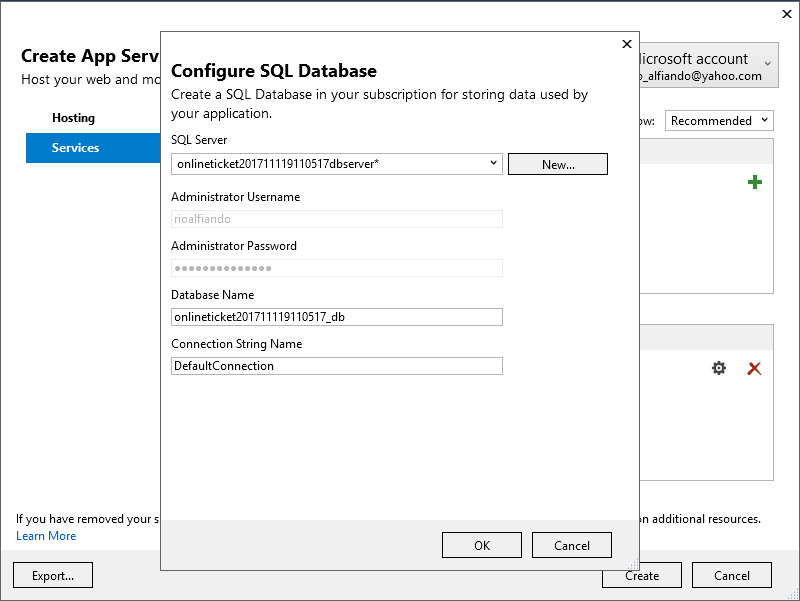
**** On this page the developer need to make sure that the Connection String Name is match with the connection string that exist in the web.config.

Figure Creating Database

**STEP 9**

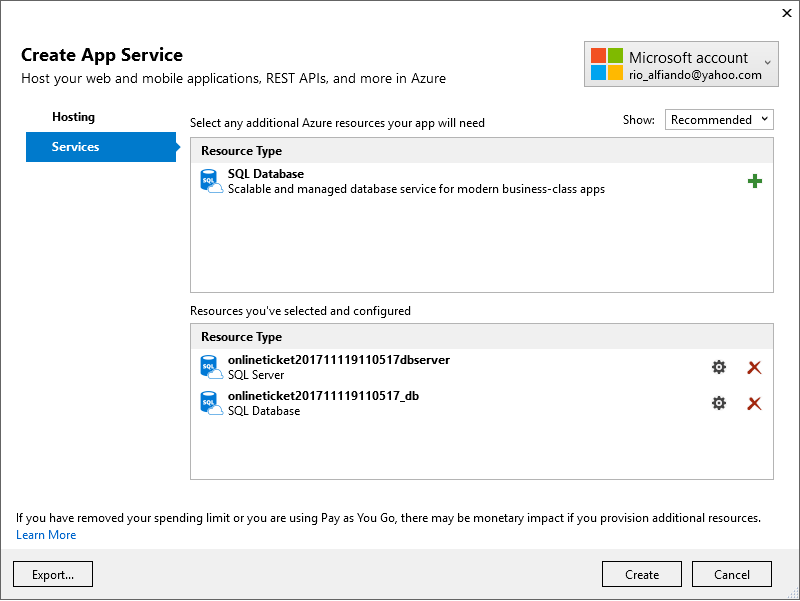
Once it is ready, project now is ready to be published.****

Figure Create Everything, deploy the website.

## Update Database with Database Migration Technology

In Section 4.2 developer already explain about how configure the database migration so the system would have implemented the database migration technology. But in this section, the developer will explain about how to migrate all the data not only the table to the new database in Azure.

After the deployment process is completely done, and the website already publish on the azure, then the website already has the database in azure platform. How to configure it, developer will list all the step needed to configure it:

**STEP 1**

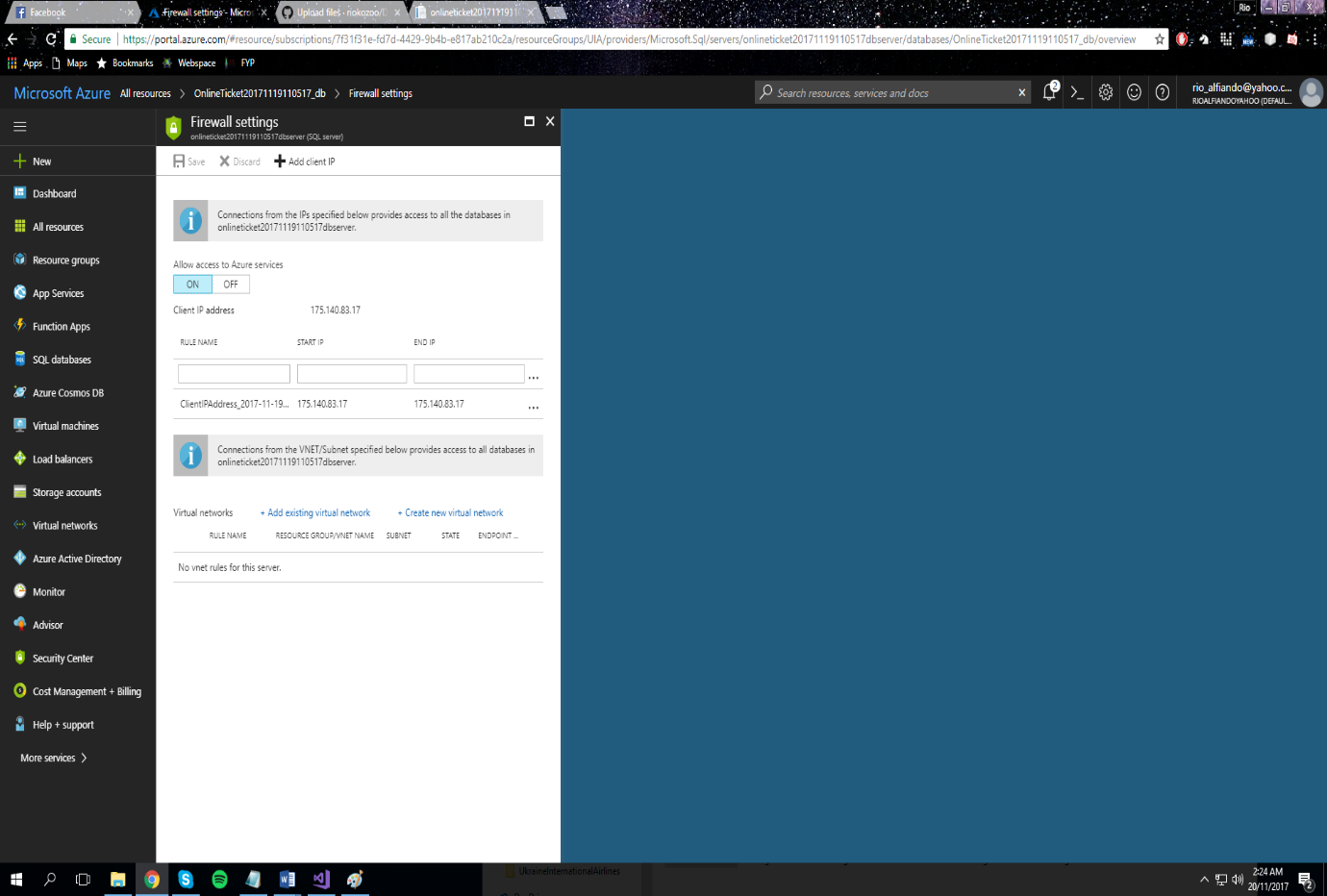
Make sure the IP address using by the developer has the permission to access the database. Go to **portal.azure.com -> sign in with the subscription account -> go to sql database used for the website -> click Set Server Firewall**

Figure Set Firewall Setting

Click Add Client IP, then it will automatically read the current IP address, and the developer need to click save, to complete the process.

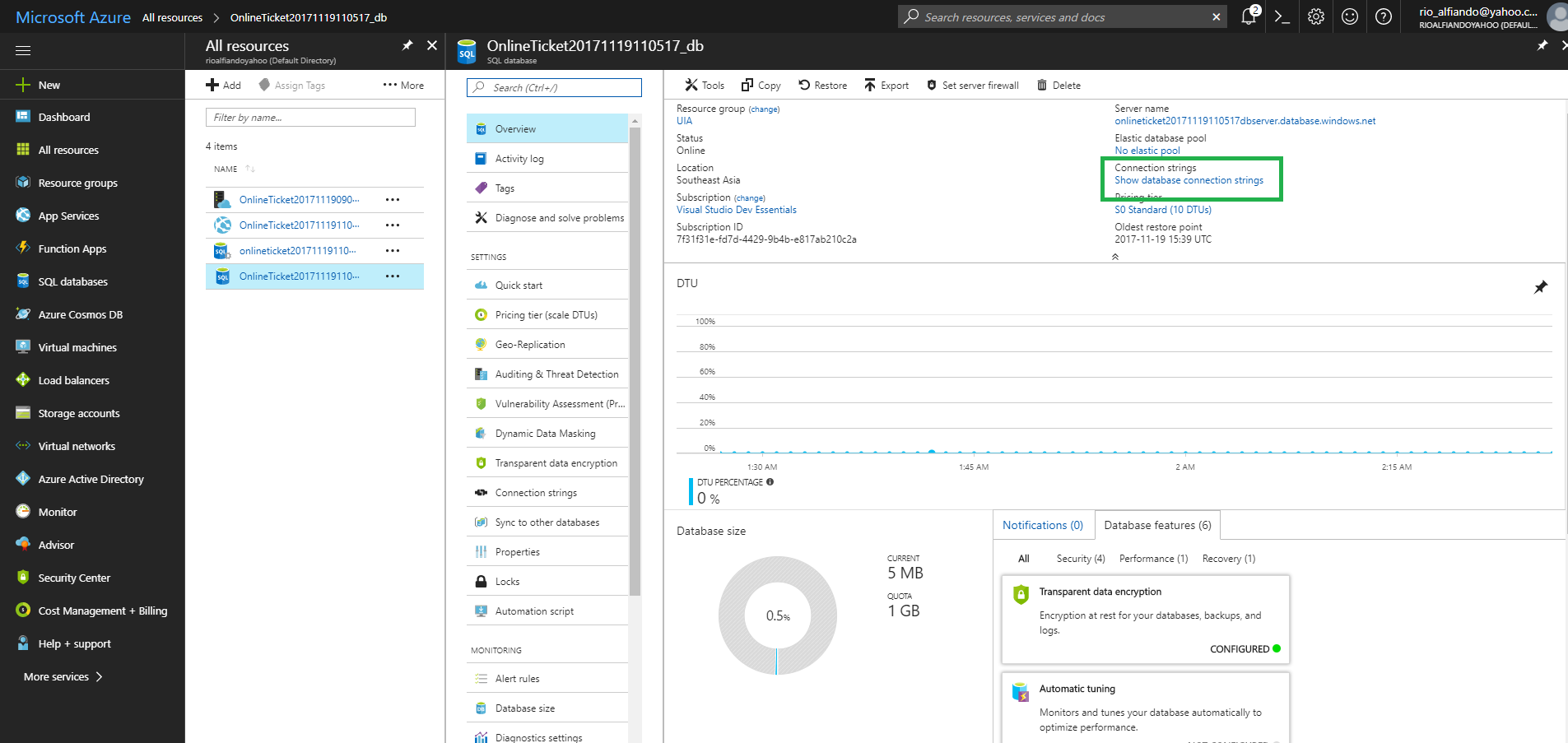
**STEP 2**

Figure Azure SQL Database

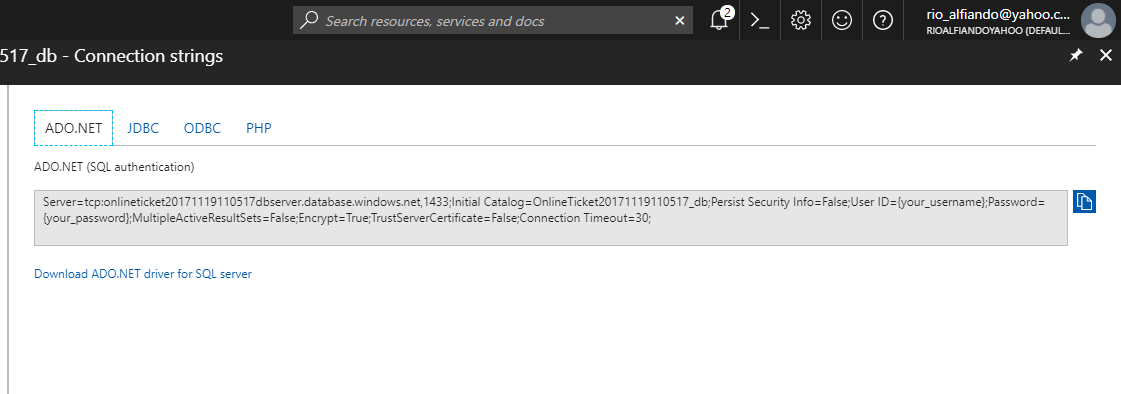
 Click Show Database Connection String, like shown in the green box in the figure above. And the developer will have the connection string like shown in the figure below.

Figure Connection String

Make sure to copy everything from this connection string.

**STEP 3**

 After copy everything, now the developer need to paste it in to the connection string in the web.config in the system in the visual studio 2017 like shown in the green box in the figure below.

Figure Web.Config of OnlineTicket.sln

**STEP 4**

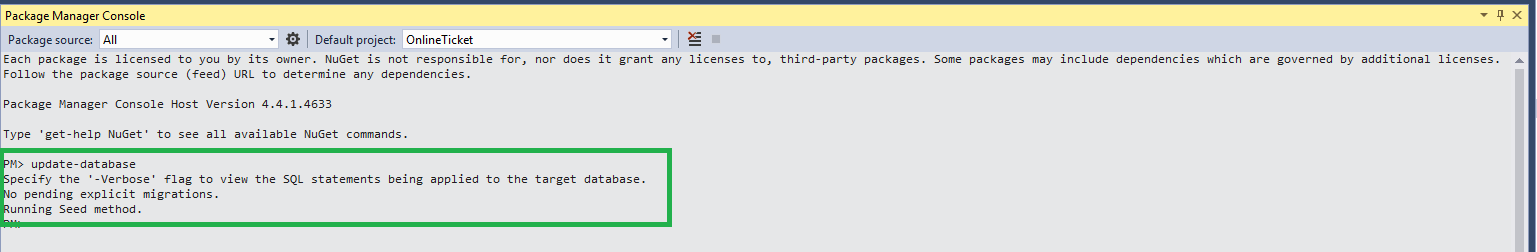
At last the developer just need to update the database using NuGet Package Manager command prompt. How to get it, select **Tools -> NuGet Package Manager -> Package Manager Console ->** type **update-database,** the enter. Like shown in the figure below, the result will say the Running seed method.

Figure NuGet Package Manager Console

Running seed method here is the one that created by the developer in the migration configuration like shown in the figure below.



Figure Seed Method In configuration of Migrations.

# TESTING

## Functional

The functional of the system will be test with the Unit Testing process, which is ever function in the system will be testing to see if it works or not. The unit testing will be drawn in the table below.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test No.** | **Activity** | **Page** | **Expected Output** | **Marks** | **Date** |
| T01 | Blank Input | Login | Error Message says field is required | *Pass* | 17 November 2017 |
| T02 | Input not exist username or invalid password. | Login | Error Message says username or password is invalid | *Pass* | 17 November 2017 |
| T03 | Input valid username and password | Login | Go to the About Page | *Pass* | 17 November 2017 |
| T04 | Click Sign Up Link | Login | Go to the registration page | *Pass* | 17 November 2017 |
| T05 | Blank Input | Registration | Error Message says field is required | *Pass* | 17 November 2017 |
| T06 | Invalid Input and click register | Registration | Error message says the regulation for that field | *Pass* | 17 November 2017 |
| T07 | Input all valid data | Registration | Success message of registration will show up. | *Pass* | 17 November 2017 |
| T08 | Input does not match password confirmation | Registration | Error Message says password does not match | *Pass* | 17 November 2017 |
| T09 | Input password less than 4 characters | Registration | Error message says password must be at least 4 characters. | *Pass* | 17 November 2017 |
| T10 | Click Login link | Registration | Go to the Login Page | *Pass* | 17 November 2017 |
| T11 | View the information about the Company background | About | Display information of company background | *Pass* | 17 November 2017 |
| T12 | View the Cargo Flight schedule | Cargo Flight Schedule | Display the list of cargo flight schedule | *Pass* | 17 November 2017 |
| T13 | View the Passenger Flight Schedule | Passenger Flight Schedule | Display the list of the passenger flight schedule | *Pass* | 17 November 2017 |
| T14 | View Profile | Profile | Display the profile details of the customers in the table. | *Pass* | 17 November 2017 |
| T15 | Click Edit link | Profile | Go to the modify profile page | *Pass* | 17 November 2017 |
| T16 | Modify profile with invalid data | Modify Profile | Error Message says the regulation of the field | *Pass* | 17 November 2017 |
| T17 | Delete one of the field, and submit the modification | Modify Profile | Error Message says that field is required. | *Pass* | 17 November 2017 |
| T18 | Modify with valid data | Modify Profile | Go to the Profile page. | *Pass* | 17 November 2017 |

## Performance

To complete this phase of the project, the built-in performance testing tool in Microsoft Azure has been used. The following figures demonstrate the completion of this process in the project.

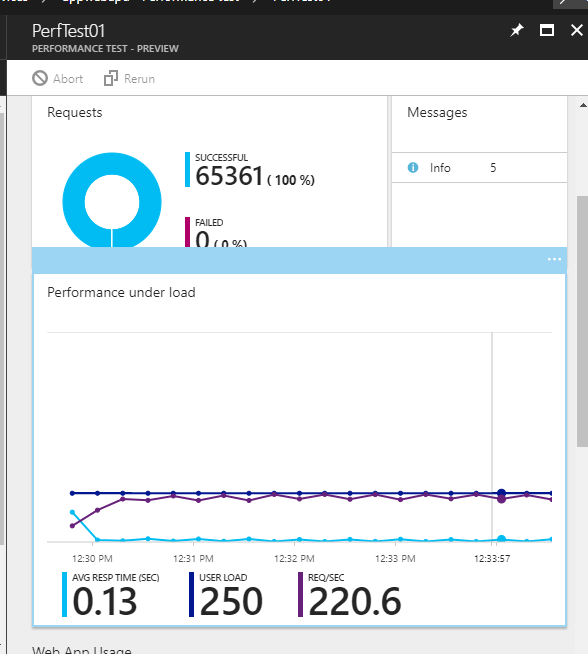


Figure Result of Performance Testing

# CONCLUSION

The conclude everything, the developer would like to say that the Online Ticket Booking System of Ukraine International Airlines has been developed successfully, and deployment has been done by the developer very smoothly. Problems stated in earlier session was clearly identified by developers, thus it gives an ease for developer to keep track on the project development progression. By achieving objectives and maintain beyond the scopes, the project can be finished within the given time. For the implementation stage, developer built all interfaces and backend programming using C# with MVC project. MVC project is dynamic and easy-to-use as it originally separates the design, controllers and its views. With MVC, faster development can be achieved, and the project can be published to Azure cloud service. Once the application was deployed, the performance of the deployed system was tested in the cloud platform which was determined to be reasonable and suitable for commercial environment.

# REFERENCES

Microsoft, 2017. *ASP.NET MVC Overview.* [Online]   
Available at: https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx  
[Accessed 15 November 2017].

2. SmartDraw, 2015. *Entity Relationship Diagram.* [Online]   
Available at: https://www.smartdraw.com/entity-relationship-diagram/  
[Accessed 15 November 2017].

# APPENDIX

The video about how to deploy the application into cloud platform is uploaded in the stream.

Video Link: <https://web.microsoftstream.com/video/824481ff-3ef8-4bdf-a9d1-c86cbaeb06c5>

The source code of web application is uploaded in the GitHub.

GitHub Link: https://github.com/riokozoo/DDAC---Online-Ticket-Booking-System