



INDIVIDUAL ASSIGNMENT
TECHNOLOGY PARK MALAYSIA

CT071-3-3-DDAC

Designing & Developing Cloud Applications

UC3F1706SE

Name : Gan Wei Teck
TP No. : TP038168
Hand-Out Date : 28th November 2017
Hand-In Date : 13rd April 2018
Lecture Name : Dr. Kalai Anand Ratnam

Acknowledgement

First of all, I would like to express my deepest gratitude to my lecturer, Dr. Kalai Anand Ratnam, who has been giving me lots of knowledge and idea about the Microsoft Azure Cloud Computing. Besides, he was also very helpful and provides professional advice on how to deploy a web application to Azure App Service. Not only that, he provides useful materials and tools that help me to save a lot of time on deploying the web application and complete the assignment on time.

I also want to take this opportunity thank to my course mates, who teach and guide me all the time in this assignment. They are willing to give help and teach me about the C# programming and some features of Microsoft Visual Studio 2017 that I have never used before. At the same time, I appreciate that my course mates who are willing to share their knowledge and help me to complete this assignment.

In conclusion, I have gained more knowledge about the Azure Cloud Computing which is very useful, because it could help me to apply it in my future works.

Contents

Acknowledgement	2
1.0 Introduction.....	5
1.1 Project Background.....	5
1.2 Project Objectives	6
1.3 Project Scope.....	6
1.4 Requirement Specification	7
1.5 Summary of Major Features / Content Solutions.....	7
2.0 Project Plan	8
2.1 Gantt Chart	8
3.0 Design	9
3.1 Design Considerations.....	9
3.2 Modelling	10
3.2.1 Use Case Diagram.....	10
3.2.2 Use Case Specification	11
3.2.3 Sequence Diagram	21
3.2.4 Entity Relationship Diagram.....	23
3.3 Cloud Architectural Design.....	24
4.0 Implementation	25
4.1 Application Development	25
4.2 Azure Publishing	27
4.2.1 App Service Publish.....	27
4.2.2 Database Publish.....	34
4.3 Application Scaling	41
4.3.1 Web Application Scaling.....	41
4.3.2 SQL Database Scaling	42
4.4 Reliability & Performance	Error! Bookmark not defined.

5.0	Test Plan & Testing Discussion	43
5.1	Unit Testing	43
5.2	Performance	44
6.0	Managed Databases	47
7.0	Conclusion	48
References	Error! Bookmark not defined.

1.0 Introduction

1.1 Project Background

Maersk Line is the global container division and the largest operating unit of the A.P. Moller – Maersk Group, a Danish business conglomerate. It is the world's largest container shipping company having customers through 374 offices in 116 countries. It employs approximately 7,000 sea farers and approximately 25,000 land-based people. Maersk Line operates over 600 vessels and has a capacity of 2.6 million TEU. The company was founded in 1928.

Operating in 100 countries and transporting goods around the globe, at first glance it would appear Danish shipping company Maersk Line is already handling all the cargo it can manage. But when Maersk determined that the volume of most of the goods it was shipping had grown to full capacity, the company decided that cloud powered solutions would be a crucial part of rectifying the situation.

“There was a ‘mind-opener’ where Maersk said, ‘How can we support the overall business strategy, and also from an IT perspective,’” says Soeren Lorenzen, an account general manager with Hewlett-Packard company who is involved first-hand with Maersk’s ITO efforts. “There was a new CIO who wanted to outsource every part of IT, but without [negatively] impacting shipping.”

In an effort to support further business growth and increase organizational flexibility, Maersk decided to consolidate all of its data centres and server rooms operating worldwide onto a virtualized platform. Microsoft Azure was already hosting some of Maersk’s IT environment, and in March 2016 Maersk initially approached Microsoft about expanding the scope of the relationship. Moving forward, Lorenzen says Maersk is currently changing over its IT setup based on Microsoft Azure, starting with the desktop environment up to container management.

1.2 Project Objectives

The objectives of the project as below:

- i. To develop a web application and deploy it on the Microsoft Azure Cloud Services.
- ii. To allow admin to manage the Maersk cargo and shipping through the website.
- iii. To allow admin to make reservation for the company who want to ship their parcel by using the container.
- iv. To allow admin to register a new cargo, vessel, and port.
- v. To allow admin to view the reservation and other information that had been registered in this system.

1.3 Project Scope

- i. The design and development of the web application based on the requirements of Maersk Line.
- ii. The system requires to store the reservation details and other information by using SQL database from Azure.
- iii. The web application provides login page for admin to login.
- iv. The system must have high security to prevent data stealing.
- v. Traffic manager will be implements to control the request from web client.
- vi. The system will distribute to the nearest point based on the user request location.
- vii. The performance testing will be done and analysed if there is any possibility of existence of bug, to ensure the system runs smoothly.

1.4 Requirement Specification

Below are list of requirements and goals that are to be implemented on the web application.

Provisioning: Developer able to provision the new application to the Microsoft Azure Platform.

Maintainability: Developer able to upgrade the application and perform other maintenance tasks while multiple tenants are using it.

Monitoring: Developer 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: Tenants 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 must be scalable to meet the demand of the application.

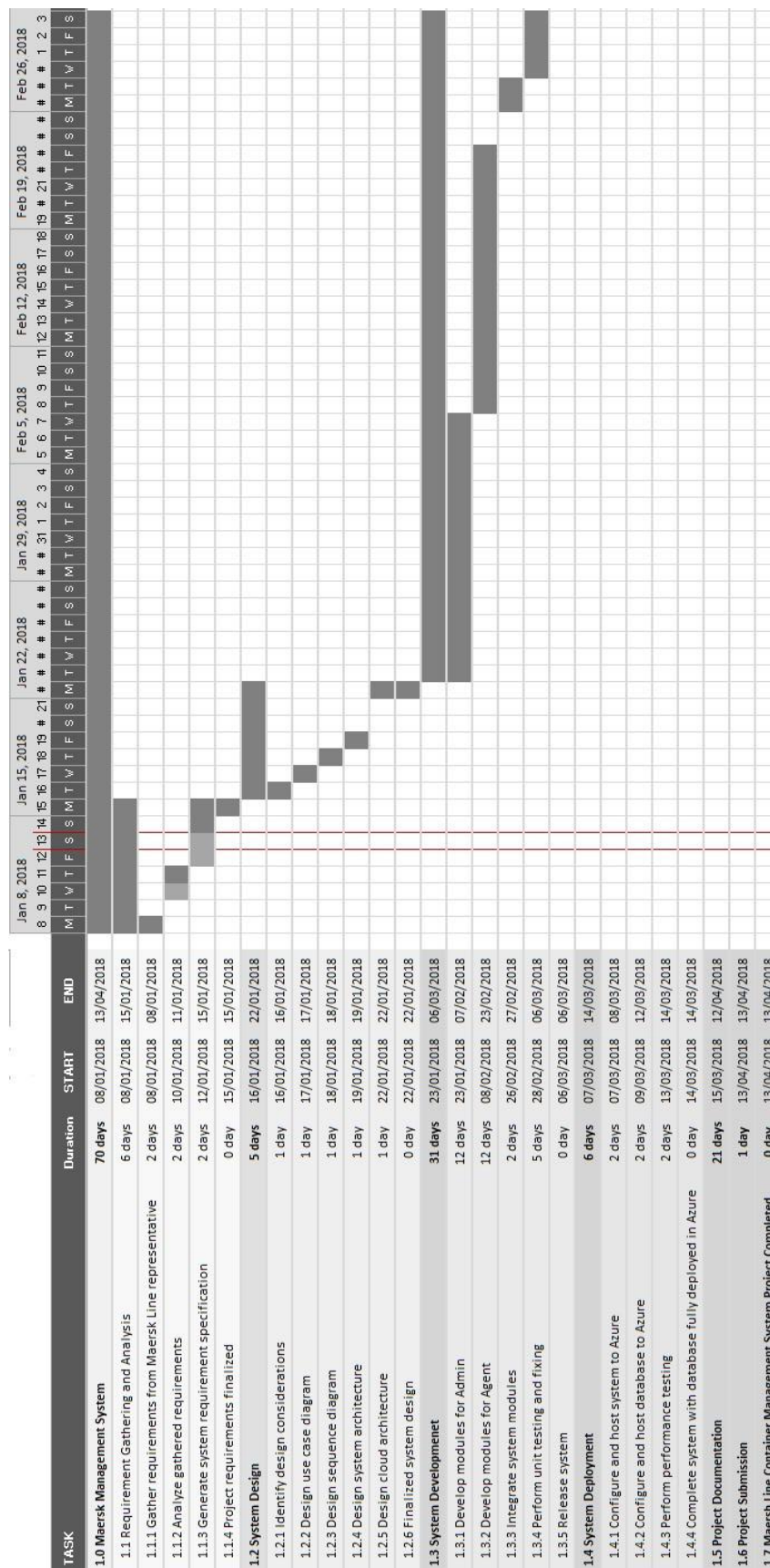
1.5 Summary of Major Features / Content Solutions

The Maerks Line web application is available and compatible with different types of browser, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and others. Users will be able to perform all the functionalities that listed below.

- User able to register vessel, cargo, and port in the web application.
- User able to make reservation for the shipment.
- User able to perform update and delete the reservation, vessel, cargo, and port in this web application.
- User able to view the details, such as registration information.
- User able to view the details of shipment information, such as departure time, arrival time, price, departure shipyard, and arrival shipyard.

2.0 Project Plan

2.1 Gantt Chart



3.0 Design

3.1 Design Considerations

There are several consideration and assumptions are made before the designing of the system. The biggest intention of Maersk Line is can reduce the overall supply chain cost and improve the efficient way to manage their logistics by developing a highly available cloud-based application to accelerate the business process.

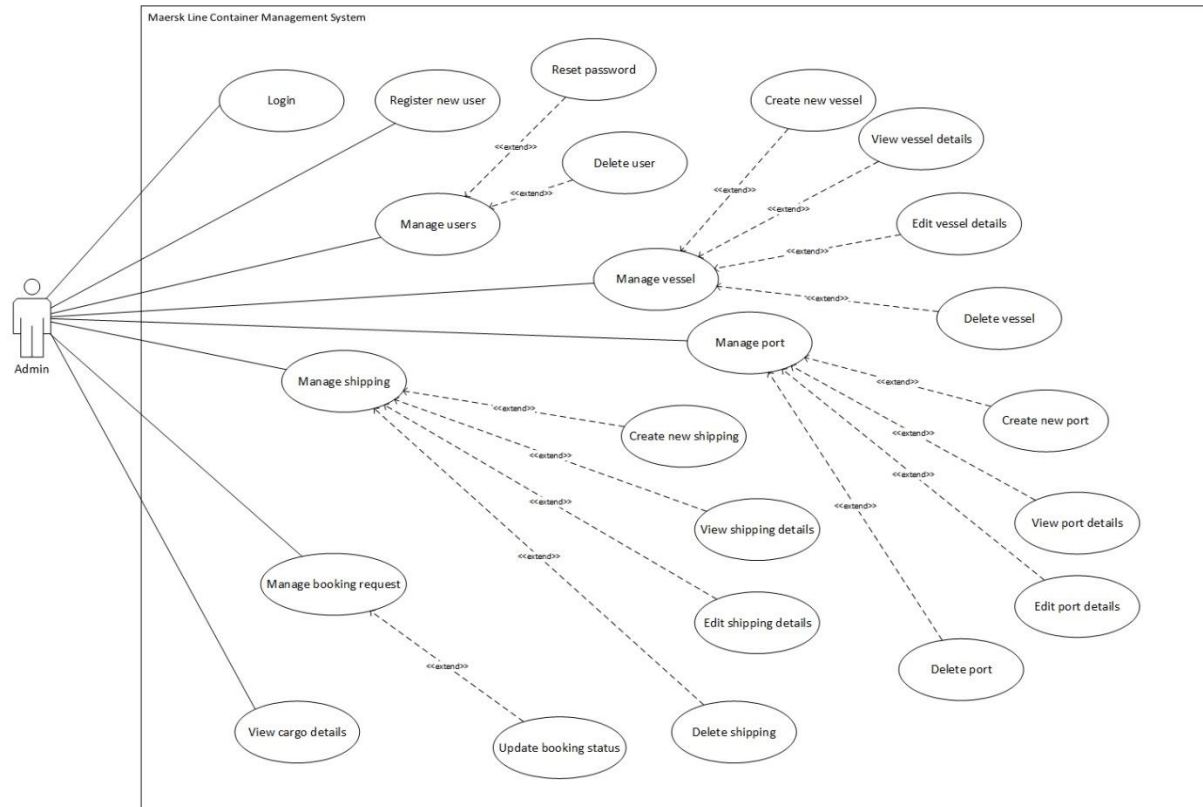
Other than that, Maersk Line IT personnel are familiar with ASP.Net development with Microsoft SQL Server which would make the development to be smooth as they have better understanding on the programming languages as well as easy to be maintained by them.

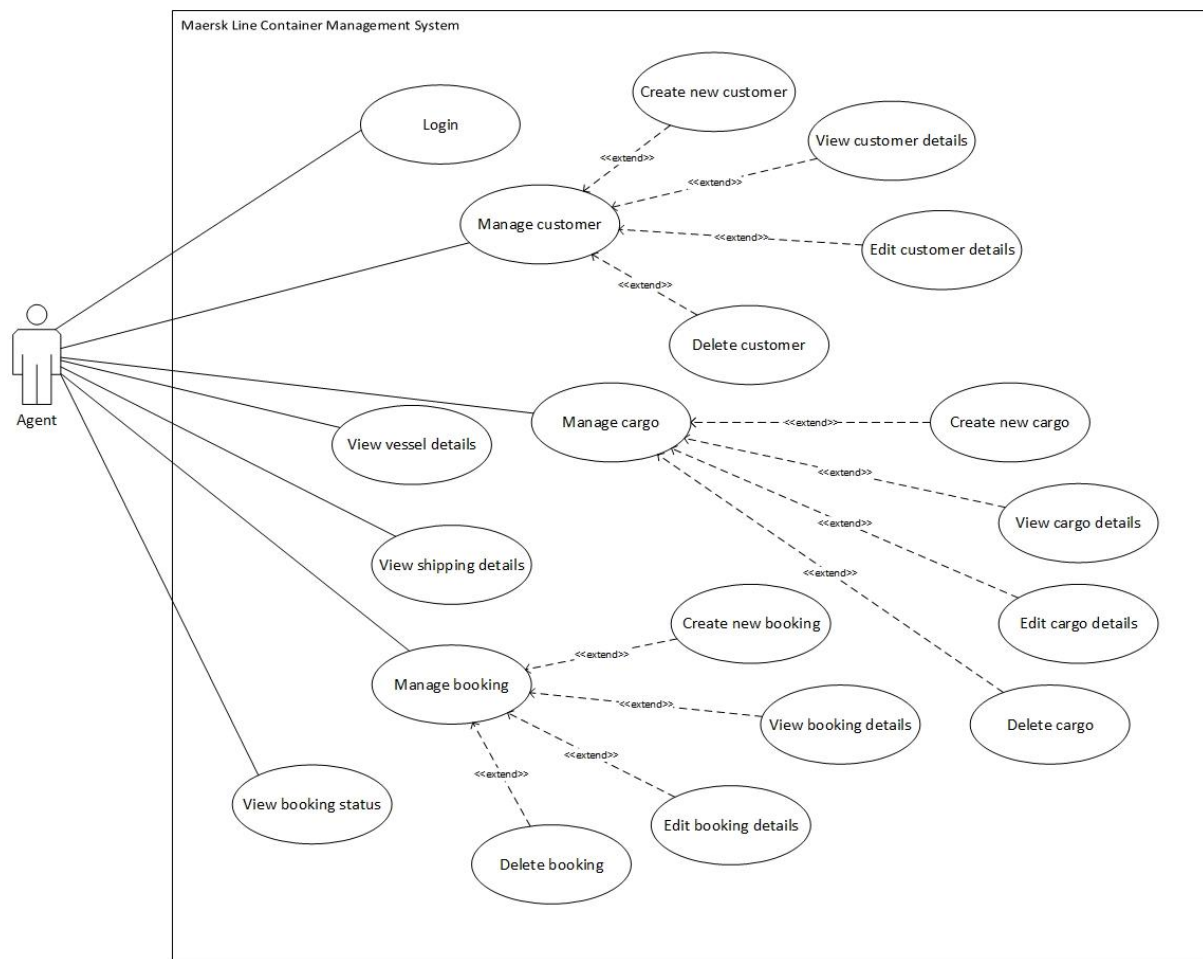
Hence, the development team has been given a limited credit on Azure, so that the team only had to host the system on Southeast Asia region to achieve cut of unnecessary cost. Furthermore, this can also increase the system flexibility according the needs of user access, an autoscaling service will be implement in the web application in each region.

3.2 Modelling

3.2.1 Use Case Diagram

Admin model



Agent model**3.2.2 Use Case Specification**

Use Case Name	Login
Description	Allow admin and agent to login to the system.
Dependency	-
Actor	Admin, Agent
Pre-conditions	System must not be logging with any user.
Event Flow	<ol style="list-style-type: none"> 1. Admin enter valid e-mail and password. 2. System perform login validation. 3. Admin is redirected to the Admin Home Page.
Alternative Flow	<ol style="list-style-type: none"> 1. Agent enter valid e-mail and password. 2. System perform login validation. 3. Agent is redirected to the Agent Home Page.
Post-condition	Admin or Agent will be redirected to the particular home page with the correct e-mail displayed on the top right corner.

Use Case Name	Register new user
Description	Allow admin to register a new agent account or new admin account.
Dependency	-
Actor	Admin
Pre-conditions	Admin must log in to the system.
Event Flow	<ol style="list-style-type: none"> 1. Admin enter valid new user account details. 2. System perform agent registration validation. 3. New user account is registered into the database.
Alternative Flow	<ol style="list-style-type: none"> 1. Admin enter invalid new user account details. 2. System perform agent registration validation. 3. Error message showed.
Post-condition	New user able to login the system by using valid information.

Use Case Name	Manage users
Description	Admin can help user to reset password or delete user account.
Dependency	<<extend>> Reset password <<extend>> Delete user
Actor	Admin
Pre-conditions	Admin must log in to the system.
Event Flow	<ol style="list-style-type: none"> 1. Look for specific account. 2. Admin can help user to reset password or delete user account.
Alternative Flow	-
Post-condition	<ol style="list-style-type: none"> 1. User password will be reset successfully. 2. User account will be delete and not able to login to the system.

Use Case Name	Manage vessel
Description	Admin can create, view, edit, and delete vessel information.
Dependency	<<extend>> Create new vessel

	<<extend>> View vessel details <<extend>> Edit vessel details <<extend>> Delete vessel
Actor	Admin
Pre-conditions	Admin must log in to the system.
Event Flow	Create new vessel: <ol style="list-style-type: none"> 1. Create new vessel record. 2. Enter new vessel information. 3. New vessel created successfully. View vessel details: <ol style="list-style-type: none"> 1. Select a vessel record to view. 2. System displays the vessel information. Edit vessel details: <ol style="list-style-type: none"> 1. Select a vessel record to edit. 2. Enter new vessel information. 3. Vessel updated successfully. Delete vessel: <ol style="list-style-type: none"> 1. Select a vessel record to delete. 2. Vessel deleted successfully.
Alternative Flow	Create new vessel: 3(a). Invalid vessel information entered. Edit vessel details: 3(a). Invalid vessel information entered. Delete vessel: 2(a). Fail to delete vessel due to one or more schedules or bookings have been assigned to this vessel.
Post-condition	Agent able to view the information of the vessel.

Use Case Name	Manage port
----------------------	--------------------

Description	Admin can create, view, edit, and delete port information.
Dependency	<<extend>> Create new port <<extend>> View port details <<extend>> Edit port details <<extend>> Delete port
Actor	Admin
Pre-conditions	Admin must log in to the system.
Event Flow	Create new port: 1. Create new port record. 2. Enter new port information. 3. New port created successfully. View port details: 1. Select a port record to view. 2. System displays the port information. Edit port details: 1. Select a port record to edit. 2. Enter new port information. 3. Port updated successfully. Delete port: 1. Select a port record to delete. 2. Port deleted successfully.
Alternative Flow	Create new port: 3(a). Invalid port information entered. Edit port details: 3(a). Invalid port information entered.
Post-condition	Agent able to make booking on the available port.

Use Case Name	Manage shipping schedule
Description	Admin can create, view, edit, and delete shipping record.
Dependency	<<extend>> Create new shipping

	<<extend>> View shipping details <<extend>> Edit shipping details <<extend>> Delete shipping
Actor	Admin
Pre-conditions	Admin must log in to the system.
Event Flow	Create new shipping: 1. Create new shipping record. 2. Enter new shipping information. 3. New shipping created successfully. View shipping details: 1. Select a shipping record to view. 2. System displays the shipping information. Edit shipping details: 1. Select a shipping record to edit. 2. Enter new shipping information. 3. Shipping updated successfully. Delete shipping: 1. Select a shipping record to delete. 2. Shipping deleted successfully.
Alternative Flow	Create new shipping: 3(a). Invalid shipping information entered. Edit shipping details: 3(a). Invalid shipping information entered. Delete shipping: 2(a). Fail to delete shipping due to one or more schedules is on-going.
Post-condition	Agent able to view shipping scheduled.

Use Case Name	Manage booking request
----------------------	-------------------------------

Description	Allow admin to view booking request list and update the booking status.
Dependency	<<extend>> Update booking status
Actor	Admin
Pre-conditions	System is logged in with an Admin account and at least one booking is requested.
Event Flow	<ol style="list-style-type: none"> 1. Admin select the booking request and click on the update booking status. 2. Admin can select confirmed, shipped, arrived, and cancelled. 3. The booking request status will be update accordingly.
Alternative Flow	-
Post-condition	The status of the booking request will be updated in the database and the agent will be notified.

Use Case Name	View cargo details
Description	Admin able to view all cargo details.
Dependency	-
Actor	Admin
Pre-conditions	System is login with Admin account and has at least one cargo record stated.
Event Flow	<ol style="list-style-type: none"> 1. Admin click on Booking tab. 2. Admin click on View Cargo Detail button.
Alternative Flow	-
Post-condition	The cargo details showed.

Use Case Name	Manage customer
Description	Agent can create, edit, delete, and view customer record.
Dependency	<<extend>> Create new customer <<extend>> Edit customer details <<extend>> Delete Customer <<extend>> View customer details
Actor	Agent

Pre-conditions	Agent must login to the system.
Event Flow	<p>Create new customer:</p> <ol style="list-style-type: none"> 1. Create new customer record. 2. Enter new customer information. 3. New customer created successfully. <p>Edit customer details:</p> <ol style="list-style-type: none"> 1. Select a customer record to edit. 2. Enter new customer information. 3. Customer updated successfully. <p>Delete customer:</p> <ol style="list-style-type: none"> 1. Select a customer record to delete. 2. Customer deleted successfully. <p>View customer details:</p> <ol style="list-style-type: none"> 1. Select a customer record to view. 2. System displays the customer information.
Alternative Flow	<p>Create new customer:</p> <p>3(a). Invalid customer information entered.</p> <p>Edit customer details:</p> <p>3(a). Invalid customer information entered.</p>
Post-condition	-

Use Case Name	Manage cargo
Description	Agent can create, edit, delete, and view cargo record.
Dependency	<<extend>> Create new cargo <<extend>> Edit cargo details <<extend>> Delete cargo <<extend>> View cargo details
Actor	Agent
Pre-conditions	Agent must login to the system.
Event Flow	<p>Create new cargo:</p> <ol style="list-style-type: none"> 1. Create new cargo record. 2. Enter new cargo information. 3. New cargo created successfully.

	Edit cargo details: <ol style="list-style-type: none"> 1. Select a cargo record to edit. 2. Enter new cargo information. 3. Cargo updated successfully. Delete cargo: <ol style="list-style-type: none"> 1. Select a cargo record to delete. 2. Cargo deleted successfully. View cargo details: <ol style="list-style-type: none"> 1. Select a cargo record to view. 2. System displays the cargo information.
Alternative Flow	Create new cargo: <ol style="list-style-type: none"> 3(a). Invalid cargo information entered. Edit cargo details: <ol style="list-style-type: none"> 3(a). Invalid cargo information entered. Delete cargo: <ol style="list-style-type: none"> 2(a). Fail to delete cargo due to one or more schedules is on-going.
Post-condition	Agent able to view cargo information.

Use Case Name	View vessel details
Description	Agent can view the information of all vessel.
Dependency	-
Actor	Agent
Pre-conditions	System is login with Agent account and has at least one vessel record stated.
Event Flow	<ol style="list-style-type: none"> 1. Agent click on View Vessel tab. 2. System shows all vessel details.
Alternative Flow	-
Post-condition	The vessel details showed.

Use Case Name	View shipping details
----------------------	------------------------------

Description	Agent can view the information of shipping.
Dependency	-
Actor	Agent
Pre-conditions	System is login with Agent account and has at least one shipping record stated.
Event Flow	<ol style="list-style-type: none"> 1. Agent click on View Shipping tab. 2. System shows all shipping details.
Alternative Flow	-
Post-condition	The shipping details showed.

Use Case Name	Manage booking
Description	Agent can create, edit, delete, and view booking record.
Dependency	<<extend>> Create new booking <<extend>> Edit booking details <<extend>> Delete booking <<extend>> View booking details
Actor	Agent
Pre-conditions	Agent must login to the system.
Event Flow	Create new booking: <ol style="list-style-type: none"> 1. Create new booking record. 2. Enter new booking information. 3. New booking created successfully. Edit booking details: <ol style="list-style-type: none"> 1. Select a booking record to edit. 2. Enter new booking information. 3. Booking updated successfully. Delete booking: <ol style="list-style-type: none"> 1. Select a booking record to delete. 2. Booking deleted successfully. View booking details: <ol style="list-style-type: none"> 1. Select a booking record to view. 2. System displays the booking information.
Alternative Flow	Create new booking:

	<p>3(a). Invalid booking information entered.</p> <p>Edit booking details:</p> <p>3(a). Invalid booking information entered.</p> <p>Delete booking:</p> <p>2(a). Fail to delete cargo due to one or more schedules is on-going.</p>
Post-condition	-

Use Case Name	View booking status
Description	Agent can view the information of booking status.
Dependency	-
Actor	Agent
Pre-conditions	System is login with Agent account and has at least one booking record stated.
Event Flow	<ol style="list-style-type: none"> 1. Agent click on Shipping Status tab. 2. System shows all shipping status.
Alternative Flow	-
Post-condition	Shipping status showed.

3.2.3 Sequence Diagram

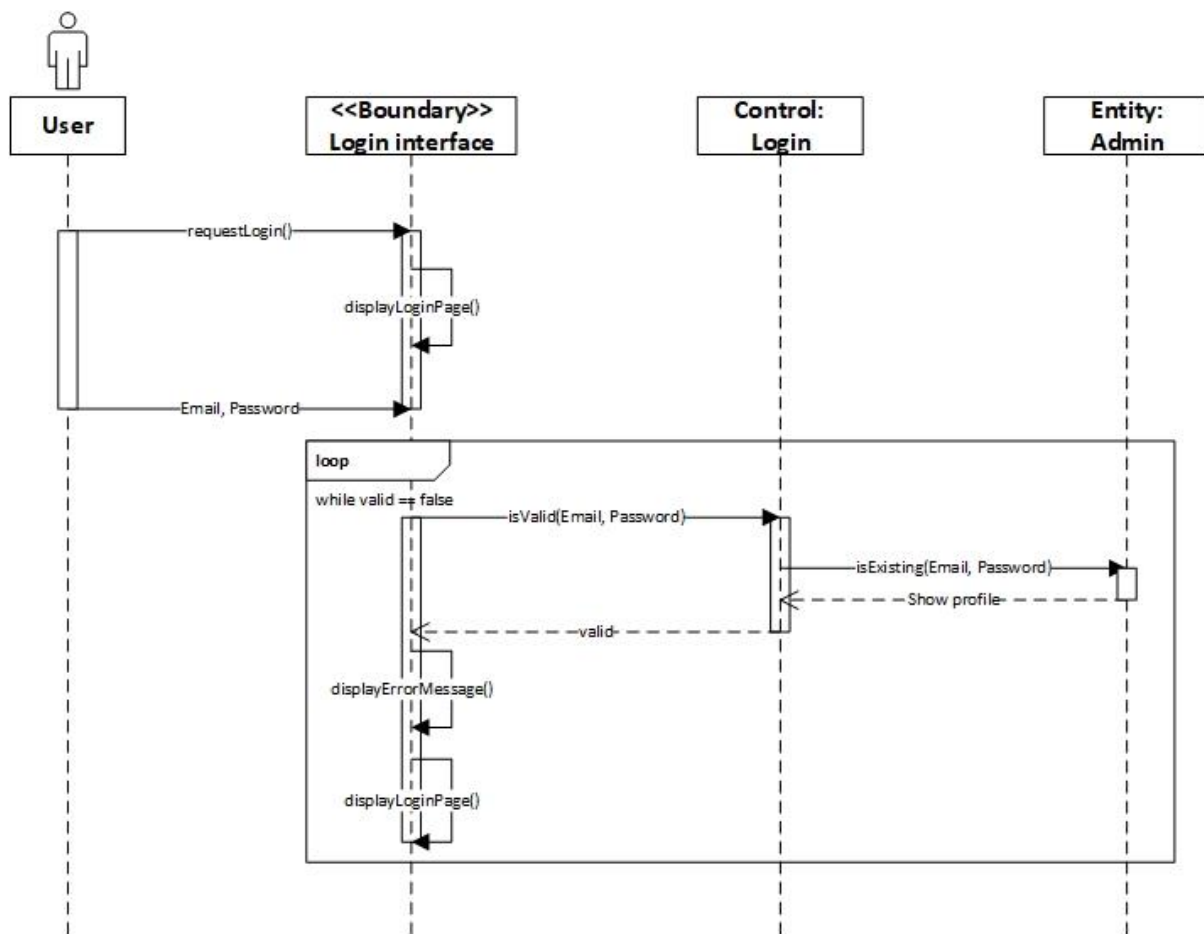


Figure 1 Sequence Diagram for Login

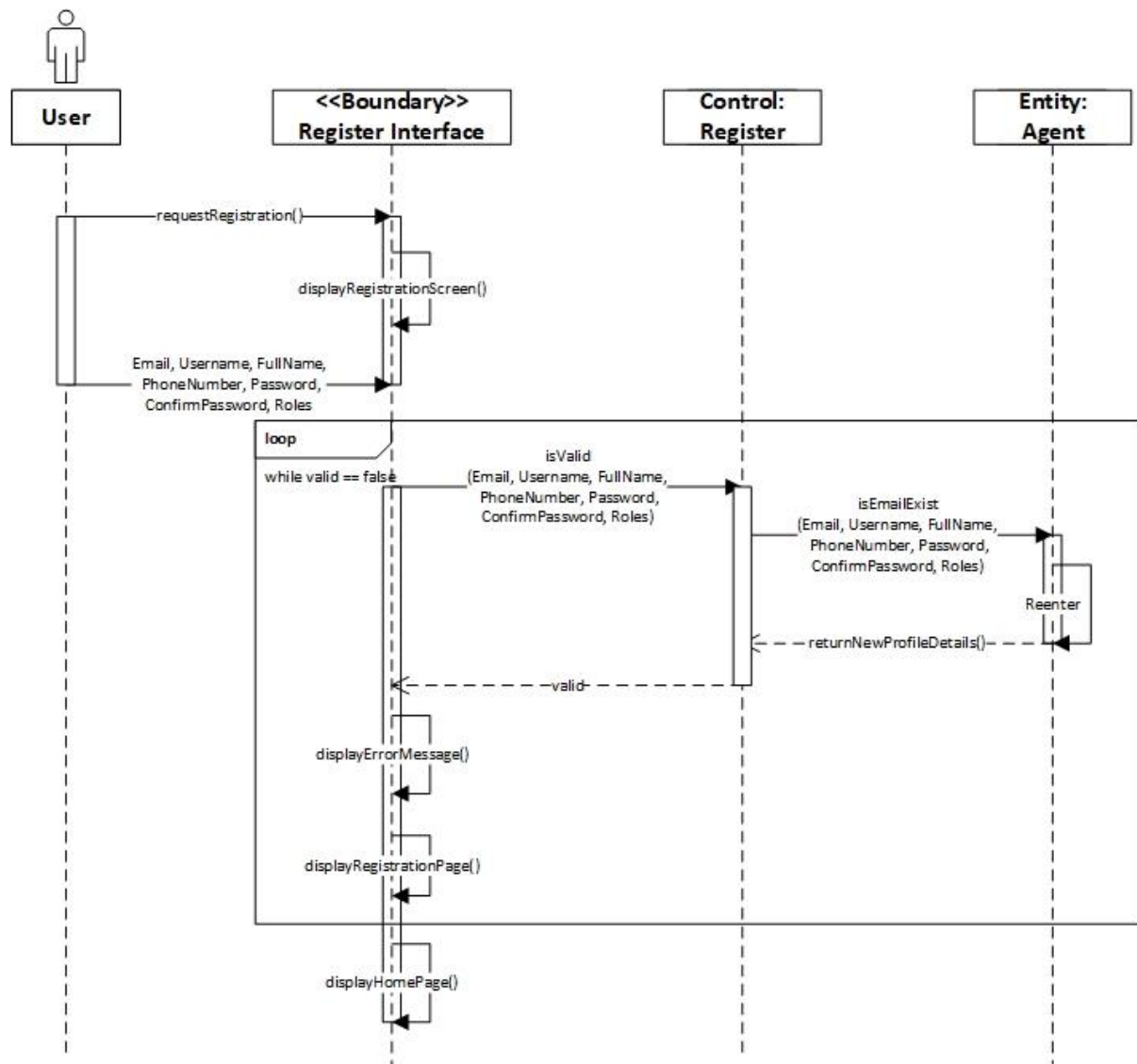


Figure 2 Sequence Diagram for Register new user

3.2.4 Entity Relationship Diagram

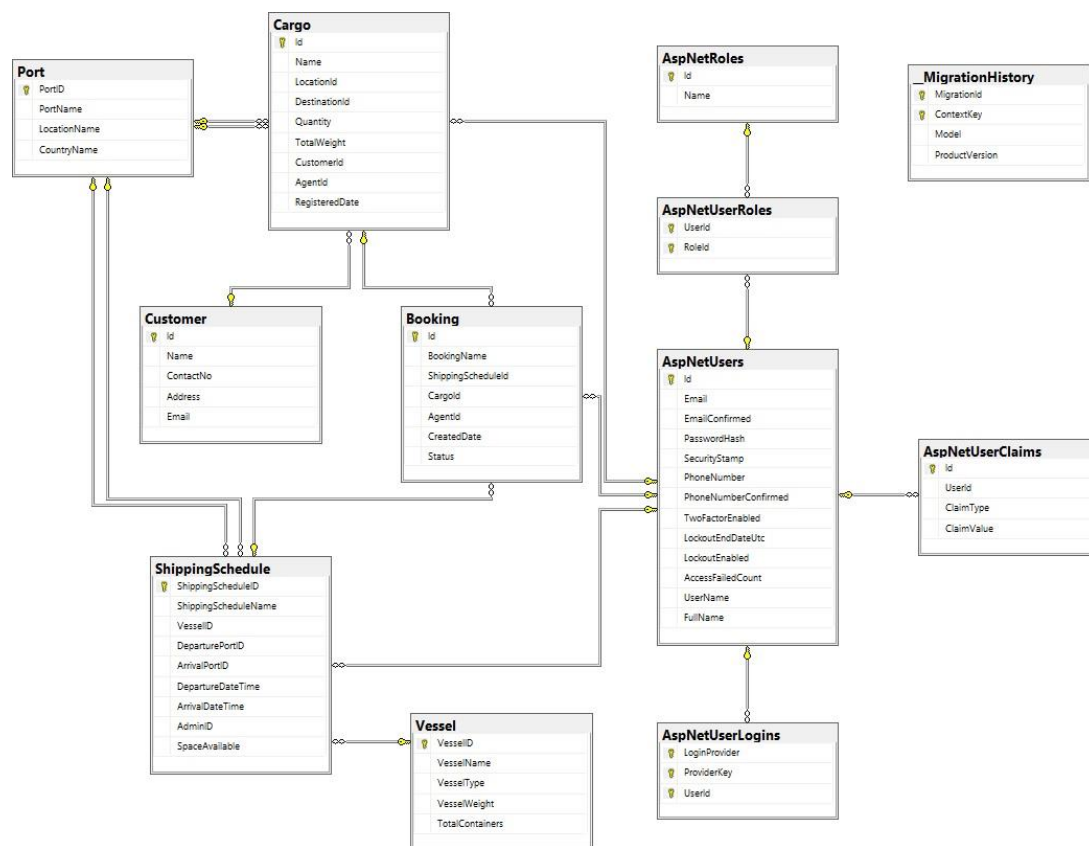


Figure 3 ERD of CRM

3.3 Cloud Architectural Design

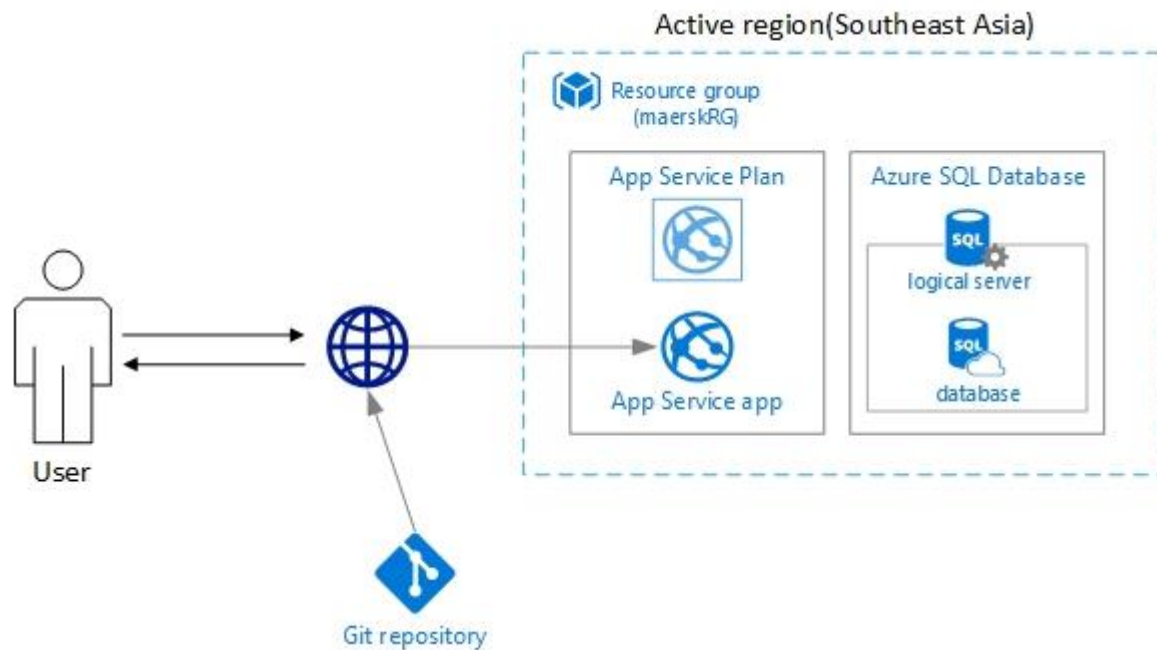


Figure 4 Cloud Architecture Diagram

Total cost incurred by implementing the architecture above is shown in the table below:

Microsoft Azure Estimate				
Your Estimate				
Service type	Custom name	Region	Description	Estimated Cost
App Service		Southeast Asia	1 instance(s) x 730 Hours, Size: S1, Standard tier, 0 SNI connection(s), 0 IP connection(s)	RM306.60
Azure SQL Database		Southeast Asia	Single Database, DTU Purchase Model, Standard Tier, S0: 10 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	RM61.82
Support			Support	RM0.00
			Monthly Total	RM368.42
			Annual Total	RM4,421.02
Disclaimer				
All prices shown are in Malaysian Ringgit (RM\$). This is a summary estimate, not a quote. For up to date pricing information please visit this link . This estimate was created at 4/13/2018 7:35:22 AM UTC.				

Figure 5 Microsoft Azure Estimate

4.0 Implementation

4.1 Application Development

The development of Maersk Line Container System Management (CSM) web application is done with Visual Studio 2017 IDE and using the MVC structure to implement the user interface which mainly broken down into three categories, Model, Controller, and View. Besides, SQL database is integrated to store and retrieve the application data based on user needs. The file structure is demonstrated with the figure below.

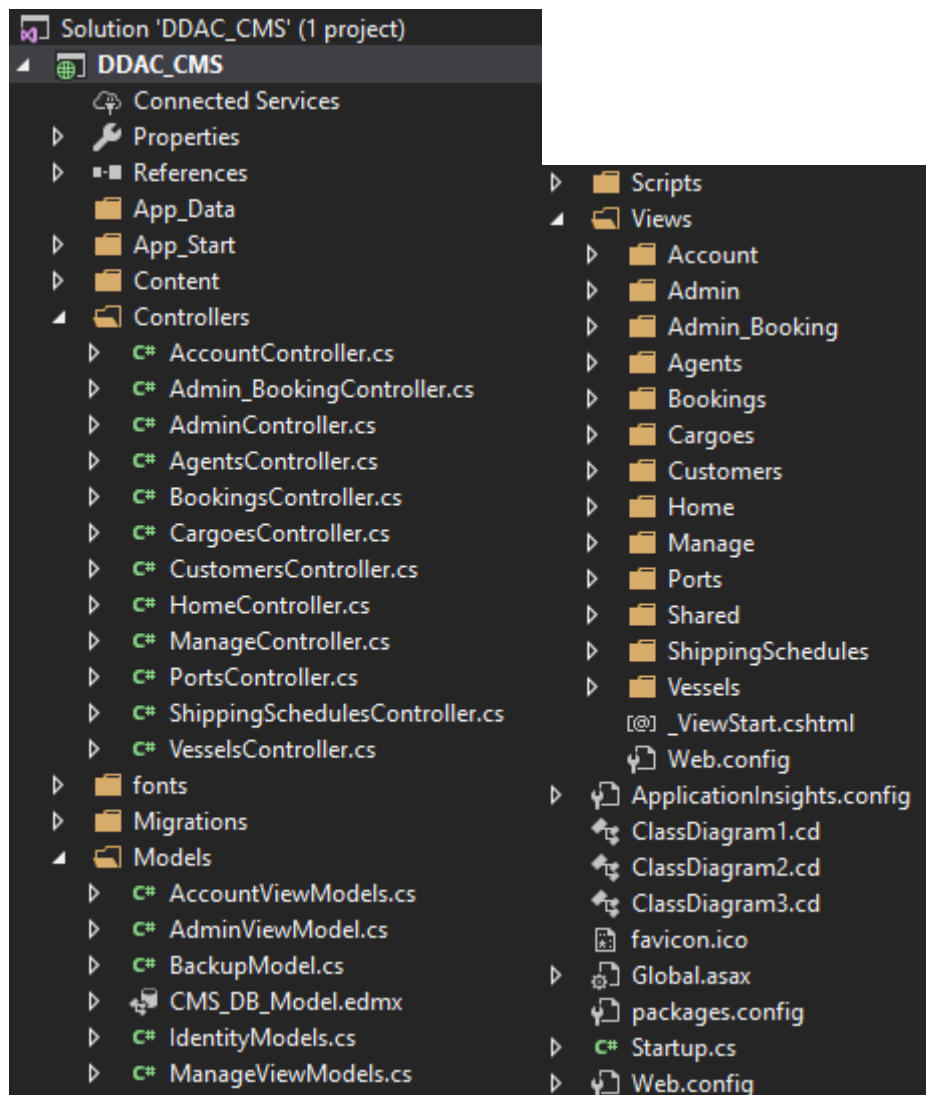


Figure 6 Visual Studio Project Directory

The models and views were generated automatically through Database First Method (DFM) which the database structure was first created then only mapping it in application with Entity Framework to fetch the database objects and build the models and view according to database semantics. The constraints and stored procedures created for each table can be imported as well

and applied to each model to verify the input when modifying object data. Besides that, the data annotation can also be defined for model attribute to define the metadata for ASP.NET MVC and ASP.NET data controls to further improve on the data integrity.

A full demo of the functional system is included in the attached CD or online at <https://web.microsoftstream.com/video/7dc1cf7e-a6dd-4ca3-8714-25e8d72324ff?list=studio> .

The document and source code of the web application is available at https://github.com/dkgan/DDAC_CMS_TP038168.

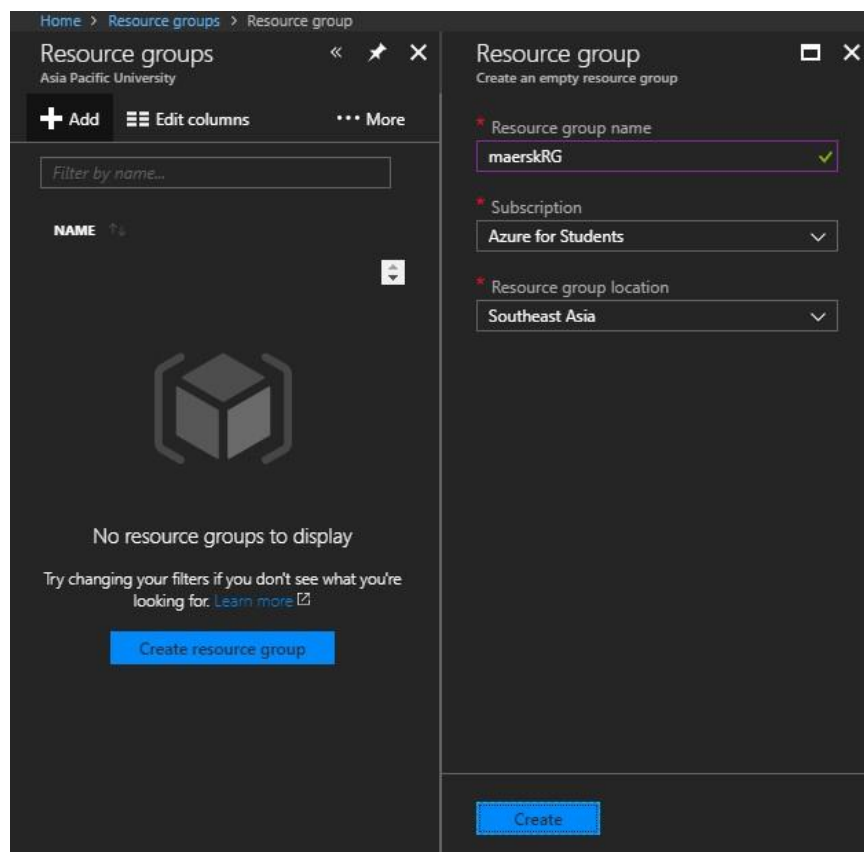
4.2 Azure Publishing

As mentioned before, the web application will be host on Microsoft Azure. There are several steps to takes to fully deploy the web application and SQL database onto Microsoft Azure. However, the selection of programming language and database to develop the web application will be affect the process of deployment onto Microsoft Azure. These are the followings steps of the deployment.

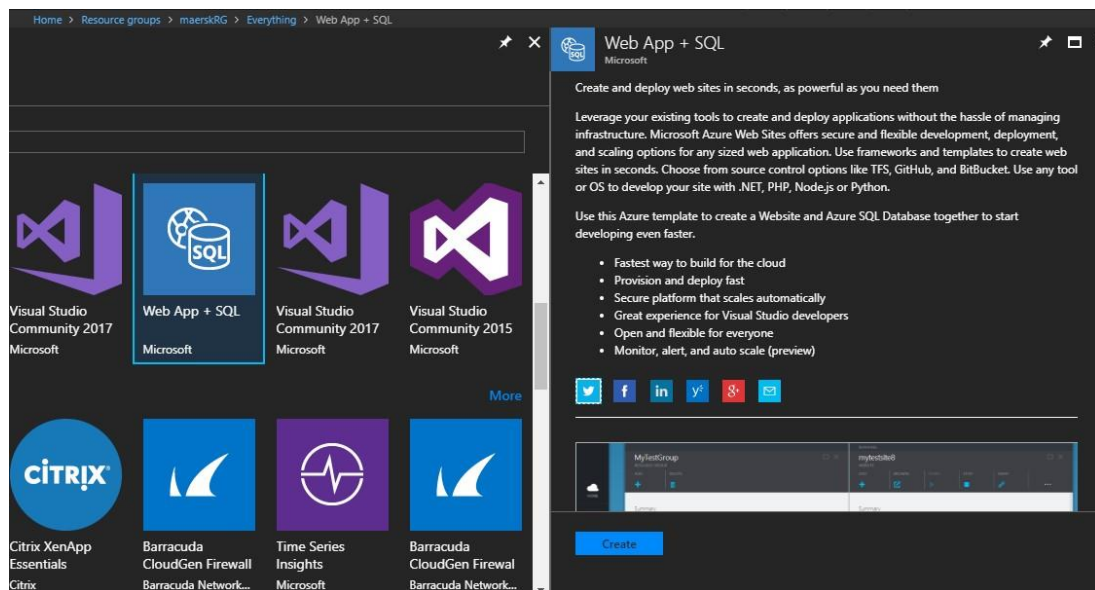
4.2.1 App Service Publish

Step 1 – Create new web service on Azure

Firstly, the developer creates a new resource group named as ‘maerskRG’ and locate at Southeast Asia. The figure below shows the creation of resource group.

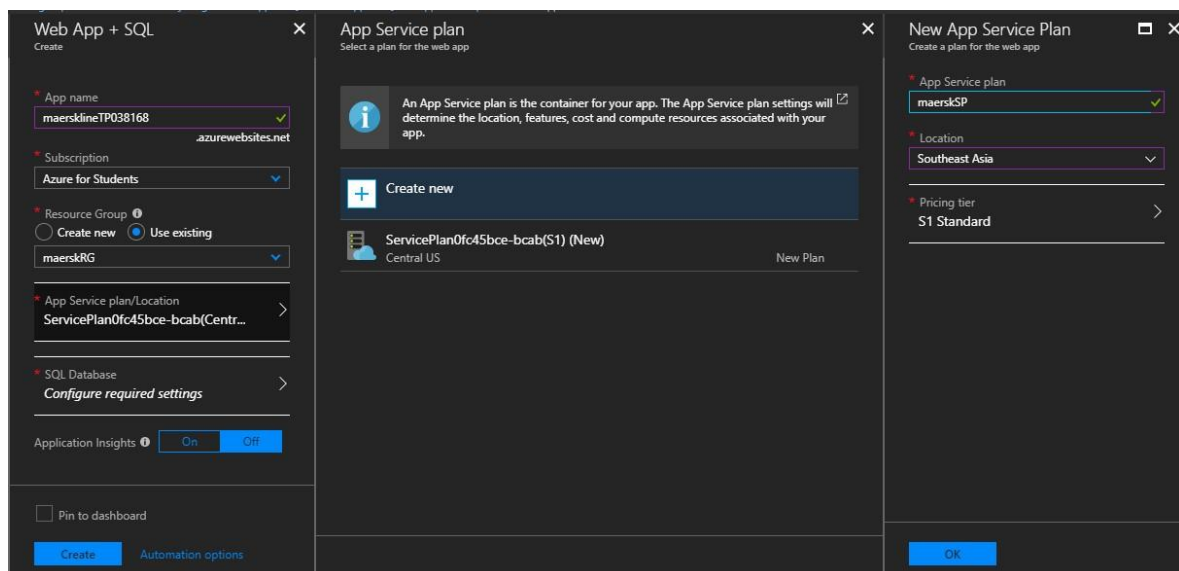


Then, the developer chooses Web App + SQL service for better deployment.



Then, the developer creates a new web application which requires the user to register app name which is the URL for the website, choose subscription, resource group, app service plan, and operating system.

By setting up the App Service plan, the developer requires to create a new App Service Plan, with a plan name, and location at Southeast Asia.



Then, the developer also requires to setting up SQL Database for the web application.

The screenshot displays the Azure portal interface for creating a new Web App + SQL. The 'Web App + SQL' pane on the left contains the following fields:

- App name: maersklineTP038168 (with a green checkmark)
- Subscription: Azure for Students (dropdown menu)
- Resource Group: maerskRG (dropdown menu)
- App Service plan/Location: maerskSP(Southeast Asia) (dropdown menu)
- SQL Database: CMS_DB (dropdown menu)

The 'Database' pane in the center shows 'No databases found'.

The 'SQL Database' pane on the right shows the following fields:

- Name: CMS_DB (with a green checkmark)
- Target server: Configure required settings (dropdown menu)
- Pricing tier: Configure required settings (dropdown menu)
- Collation: SQL_Latin1_General_CP1_CI_AS (dropdown menu)

A 'Select' button is located at the bottom right of the 'SQL Database' pane.

In target server, the developer must create a new server. By setting up, the server name must be unique, yet the developer must prepare the server admin login and password to access the database server.

The screenshot displays the Azure portal interface for creating a new server. The 'SQL Database' pane on the left contains the following fields:

- Name: CMS_DB (with a green checkmark)
- Target server: Configure required settings (dropdown menu)
- Pricing tier: Configure required settings (dropdown menu)
- Collation: SQL_Latin1_General_CP1_CI_AS (dropdown menu)

The 'Server' pane in the center shows 'No servers found'.

The 'New server' pane on the right shows the following fields:

- Server name: maerskservtp038168 (with a green checkmark)
- Server admin login: maerskdk (with a green checkmark)
- Password: (masked) (with a green checkmark)
- Confirm password: (masked) (with a green checkmark)
- Location: Southeast Asia (dropdown menu)
- Allow azure services to access server: (checked)

A 'Select' button is located at the bottom right of the 'New server' pane.

After setting up target server, next is pricing tier. In Resource Configuration and Pricing, Azure will let the developer to choose the specific setting for the system. Other than that, price will be given is different setting. As the figure below, the developer has chosen 10 units of DTUs, which cost RM63 per DTU. Besides, the max data size to store data can store up to 250GB.

Resource Configuration & Pricing

Feedback

Free
A basic level database with shared storage.
Starting at 20.96 MYR / month

Basic
For less demanding workloads
Starting at 20.96 MYR / month

Standard
For most production workloads
Starting at 63.00 MYR / month

Premium
For IO-intensive workloads.
Starting at 1953.00 MYR / month

elastic-based purchasing options
(Premium SQL Server for customerized performance using elastic)

DTUs [What is a DTU?](#)

20 50 100 200 400 800 1600 3000 10 (\$0)

Max data size

100 MB 250 GB 250 GB

Cost Summary

Cost per DTU (in MYR)	6.30
DTUs selected	x 10
EST. COST PER MONTH	63.00 MYR

Apply

In the figure below, the developer has finish setting up Web App + SQL service.

Web App + SQL

Create

* App name
maersklineTP038168 ✓
.azurewebsites.net

* Subscription
Azure for Students

* Resource Group ⓘ
☐ Create new ☒ Use existing
maerskRG

* App Service plan/Location
maerskSP(Southeast Asia)

* SQL Database
CMS_DB

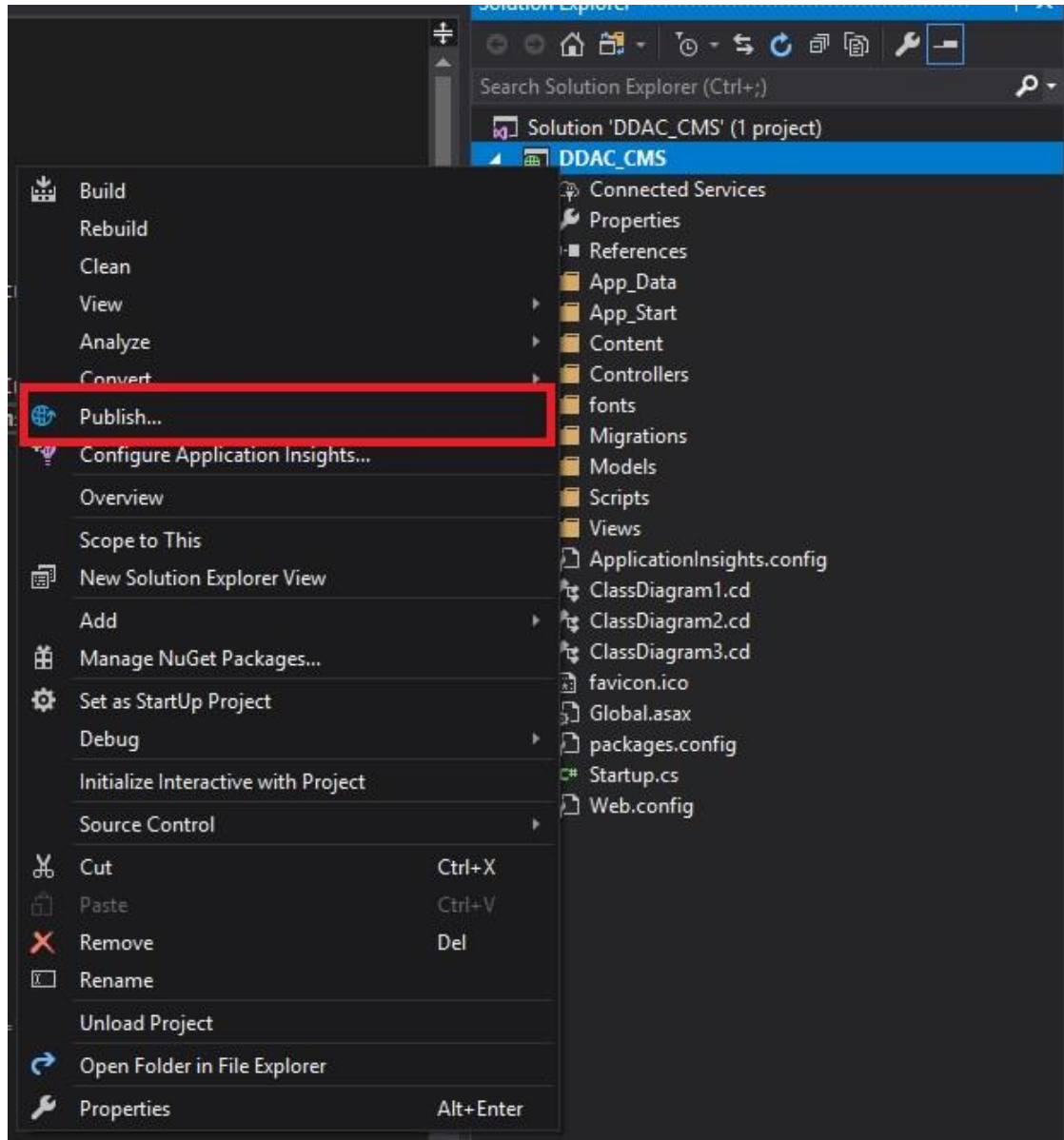
Application Insights ⓘ ☒ On ☐ Off

☐ Pin to dashboard

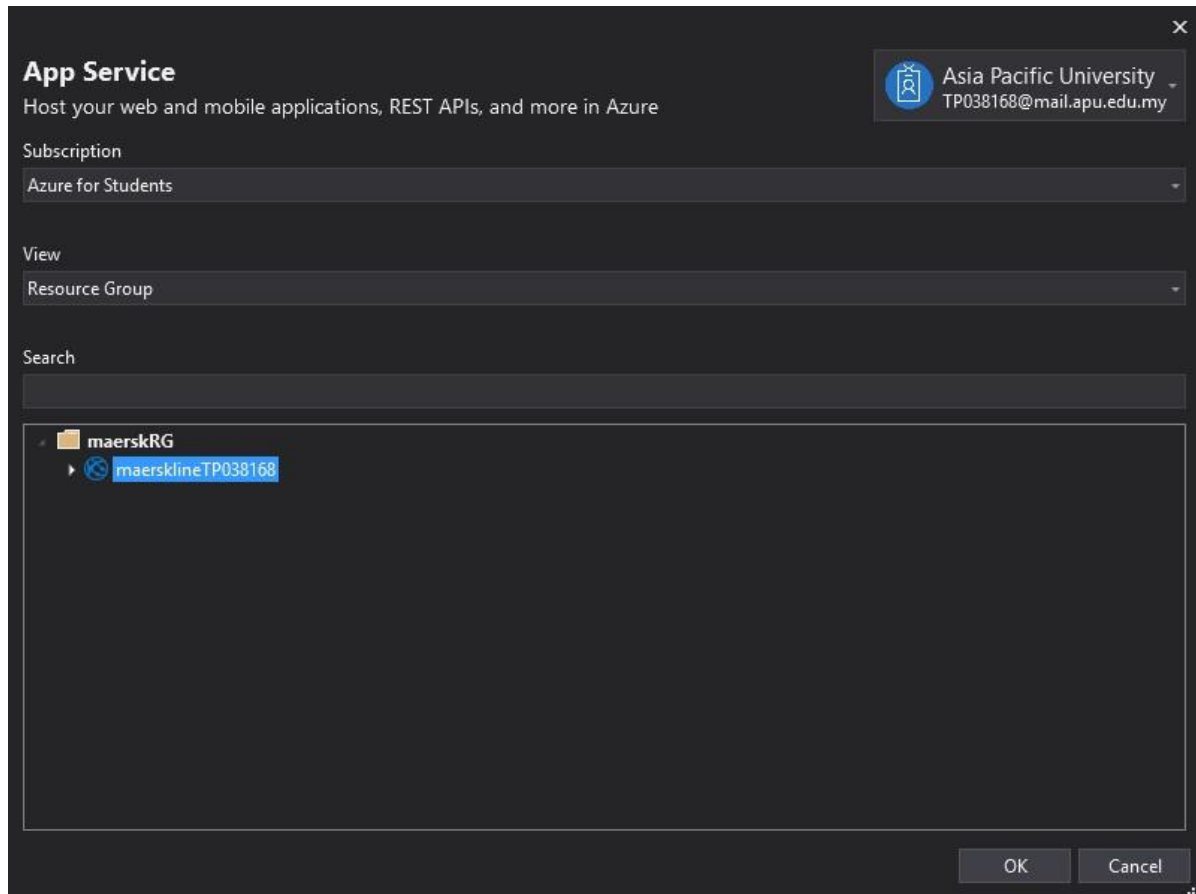
Create Automation options

Step 2 – Publish Web Application

Open Visual Studio and open the application's project. Find the main project in solution explorer. Right click the main project and press "Publish". The following shows the snapshot of this process.



Then, a window will prompt. It requires user to login to the Microsoft Azure account. Choose the correct subscription, in this case, “Azure for Students” is used. A list of resource groups will be listed and choose the correct one and press “OK” button to publish the web application.

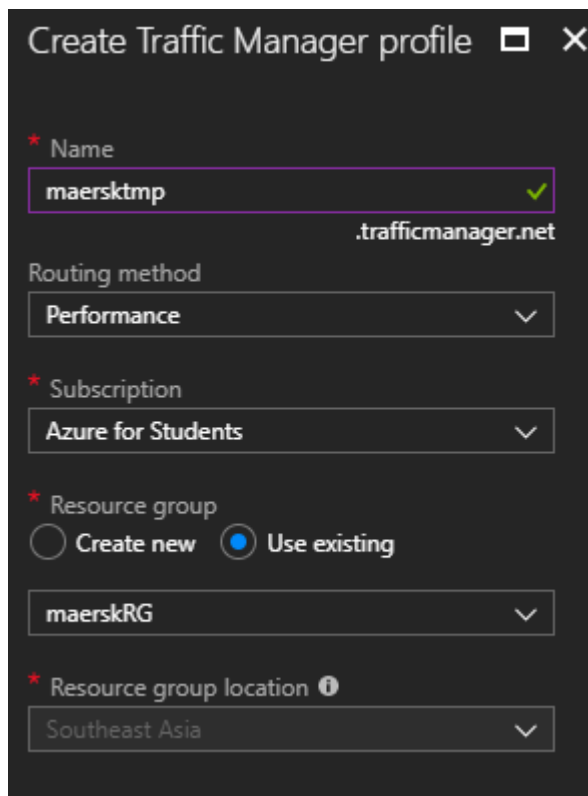


Wait until the process completes, and the following messages will be shown.

```
2>obj\Release\Package\PackageTmp.
2>Start Web Deploy Publish the Application/package to https://maerskmanagement.scm.azurewebsites.net/msdeploy.axd?site=MAERSKManagement ...
2>Adding ACLs for path (MAERSKManagement)
2>Adding ACLs for path (MAERSKManagement)
2>Updating file (MAERSKManagement\bin\DDAC.dll).
2>Updating file (MAERSKManagement\bin\DDAC.pdb).
2>Adding ACLs for path (MAERSKManagement)
2>Adding ACLs for path (MAERSKManagement)
2>Publish Succeeded.
2>Web App was published successfully http://maerskmanagement.azurewebsites.net/
===== Build: 1 succeeded, 0 failed, 3 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

Step 3 – Set up Traffic Manager Profile

Find the service of “Traffic Manager Profile” and press it. It will come up a registration form that require user to fill up the information such as name of the manager, routing method, subscription, and resource group. Fill up all required information and press “Create” button to create the profile.



Create Traffic Manager profile

* Name
maersktmp ✓
.trafficmanager.net

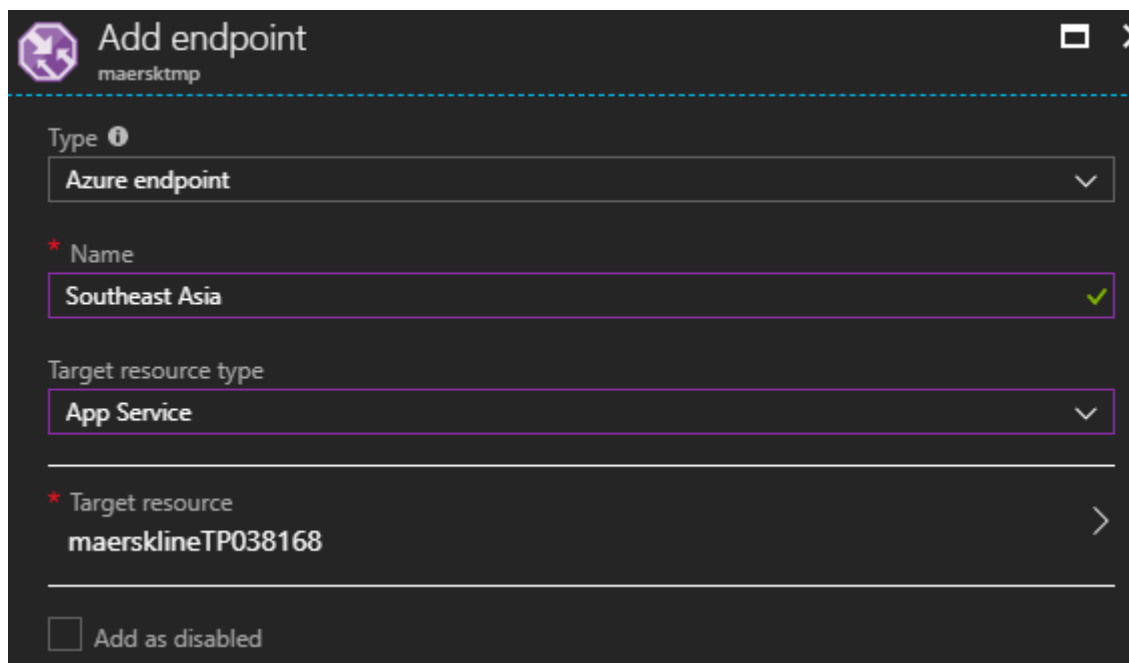
Routing method
Performance

* Subscription
Azure for Students

* Resource group
☐ Create new ☒ Use existing
maerskRG

* Resource group location ⓘ
Southeast Asia

Then, go to “Endpoints” in settings this traffic manager profile. Press to add the Azure endpoint. Fill up all the information, especially the correct target resource. The following figure shows the configuration of endpoint.



Add endpoint
maersktmp

Type ⓘ
Azure endpoint

* Name
Southeast Asia ✓

Target resource type
App Service

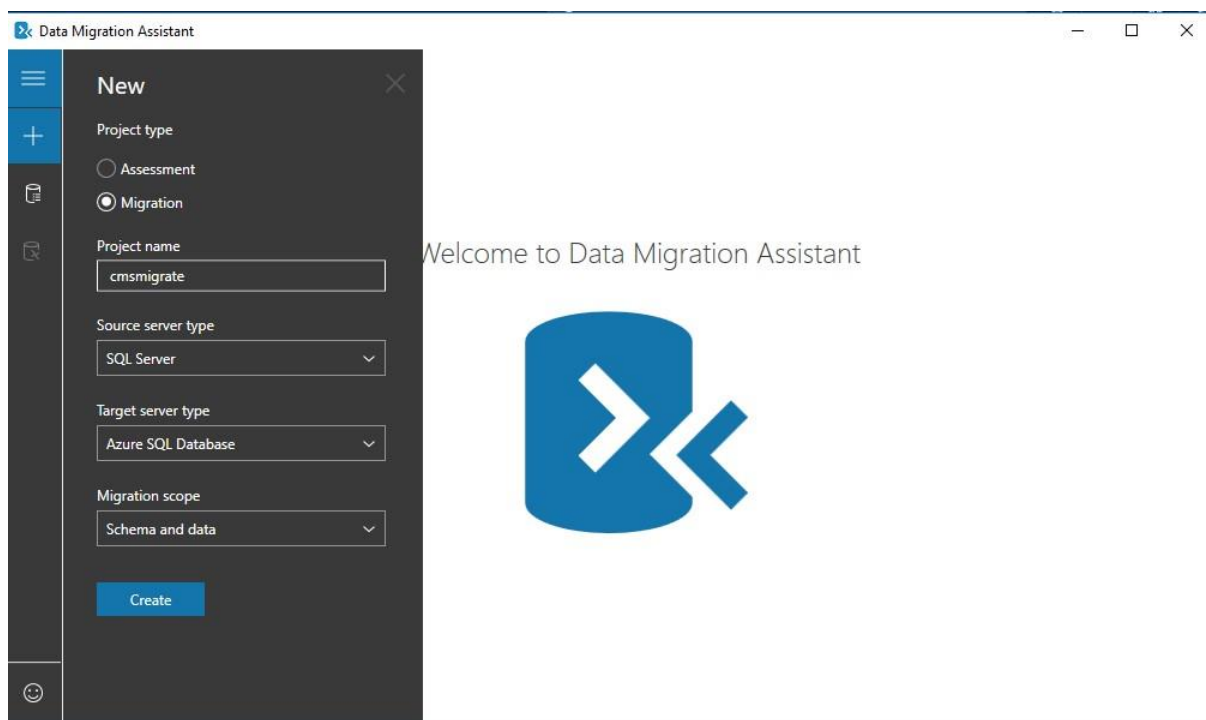
* Target resource
maersklineTP038168 >

☐ Add as disabled

4.2.2 Database Publish

Step 1 – Create new web service on Azure

After creating SQL database service, install an application called “Microsoft Data Migration Assistant (DMA)” to migrate the database from local server to Azure. Open DMA and press create to create a new migration project. Fill up all required information and press create button. The following figure shows the configuration of migration project.



Insert the server name of local server so that the database can be retrieved and migrated. Choose the correct database that needs to be migrated to Azure. Press “Next” button to proceed to next steps.

The screenshot shows the 'Data Migration Assistant' window with the 'cmsmigrate' title. The progress bar at the top indicates the current step is '1 Select source', followed by '2 Select target', '3 Select objects', '4 Script & deploy sch', '5 Select tables', and '6 Migrate data'. On the left, there are icons for a menu, a plus sign, a document, and a refresh button. The main area is titled 'Connect to source server'. It includes a 'Server name' dropdown set to 'DK', an 'Authentication type' dropdown set to 'Windows Authentication', and 'Connection properties' with checkboxes for 'Encrypt connection' and 'Trust server certificate'. Below these is a 'Source SQL Server permissions' section with a note about the 'CONTROL SERVER' permission and a 'Connect' button. On the right, a table lists databases to select a single one for migration to Azure SQL Database. The table has columns 'Name' and 'Compatibility Level'. The databases listed are ADVBS_DB (130), ADVBS_TSL_DB (140), CMS_DB (130, selected), DWConfiguration (140), DWDiagnostics (140), and DWQueue (140). A 'Next' button is at the bottom right.

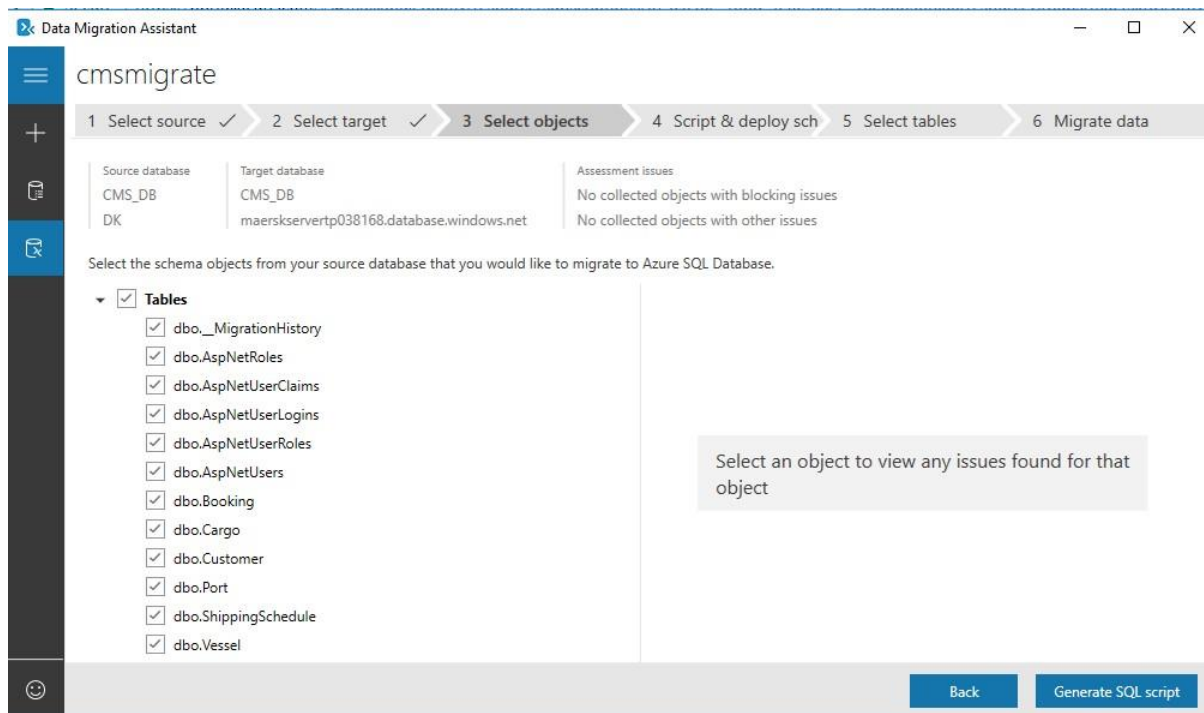
Name	Compatibility Level
ADVBS_DB	130
ADVBS_TSL_DB	140
CMS_DB	130
DWConfiguration	140
DWDiagnostics	140
DWQueue	140

Insert the name of the server name of SQL Database in Azure and connect to the server. It requires the username and password to access the SQL Server and this username and password is configured in previous step. Choose the Database as the destination of the migration. In this case, the database called “CMS_DB”. Press “Next” button to proceed to next step.

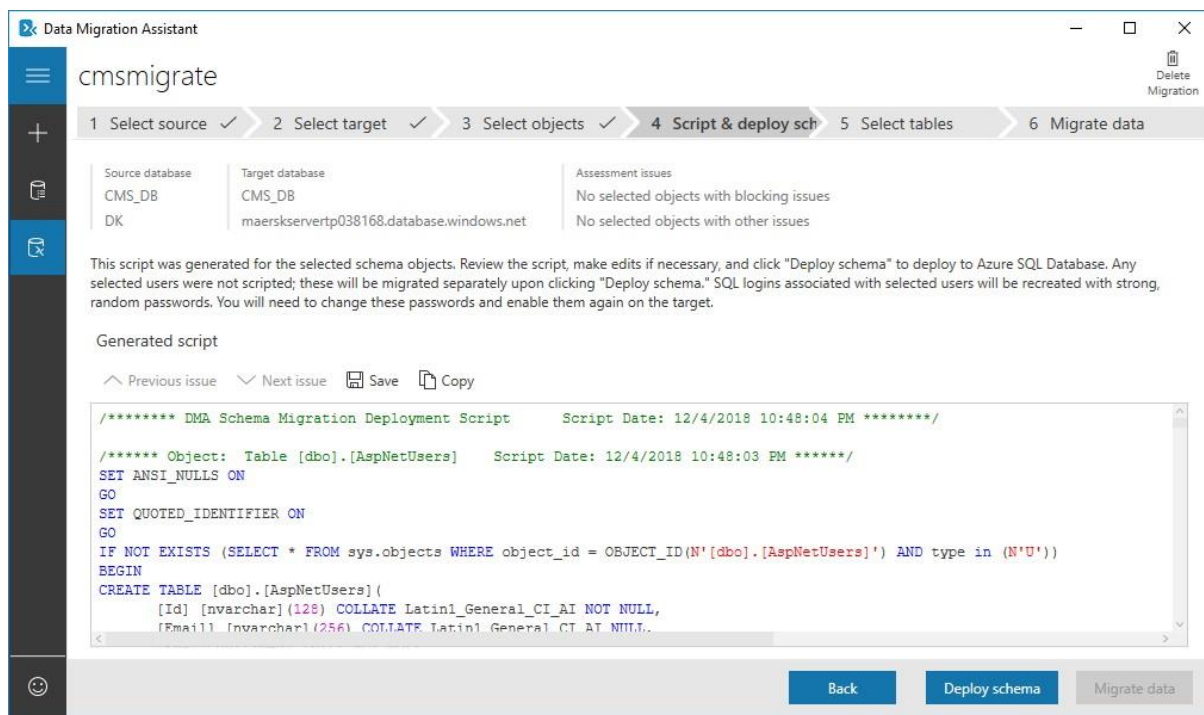
The screenshot shows the 'Data Migration Assistant' window with the 'cmsmigrate' title. The progress bar at the top indicates the current step is '2 Select target', with '1 Select source' marked as complete. On the left, there are icons for a menu, a plus sign, a document, and a refresh button. The main area is titled 'SQL Authentication credentials'. It includes a 'Username' field with 'maerskdk', a 'Password' field with masked characters, and 'Connection properties' with checkboxes for 'Encrypt connection' and 'Trust server certificate'. Below these is a 'Target Azure SQL Database permissions' section with a note about the 'CONTROL DATABASE' permission and a 'Connect' button. On the right, a table lists databases to select a single one for migration from the target Azure SQL Database server. The table has columns 'Name' and 'Compatibility Level'. The database listed is CMS_DB (140, selected). A 'Back' button is at the bottom left and a 'Next' button is at the bottom right.

Name	Compatibility Level
CMS_DB	140

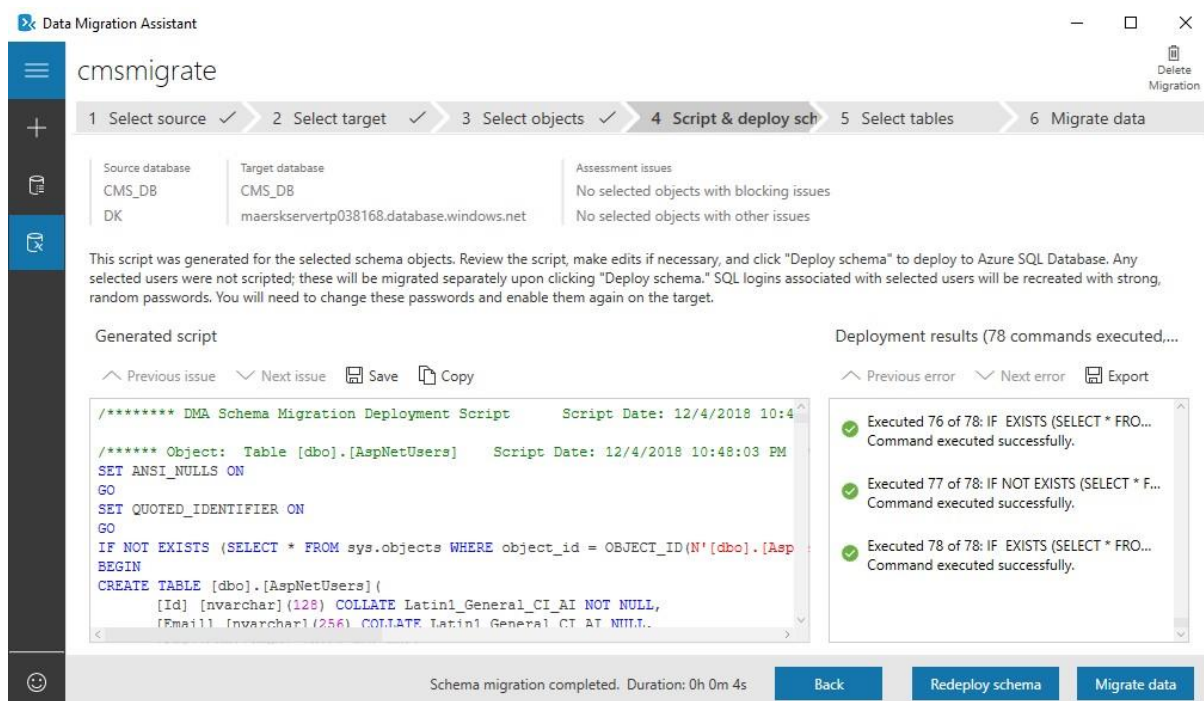
Tick the table that needs to be migrated to the Azure. Press “Generate SQL Script” to proceed to the next step. The following figure shows the table that needs to be migrated to Azure.



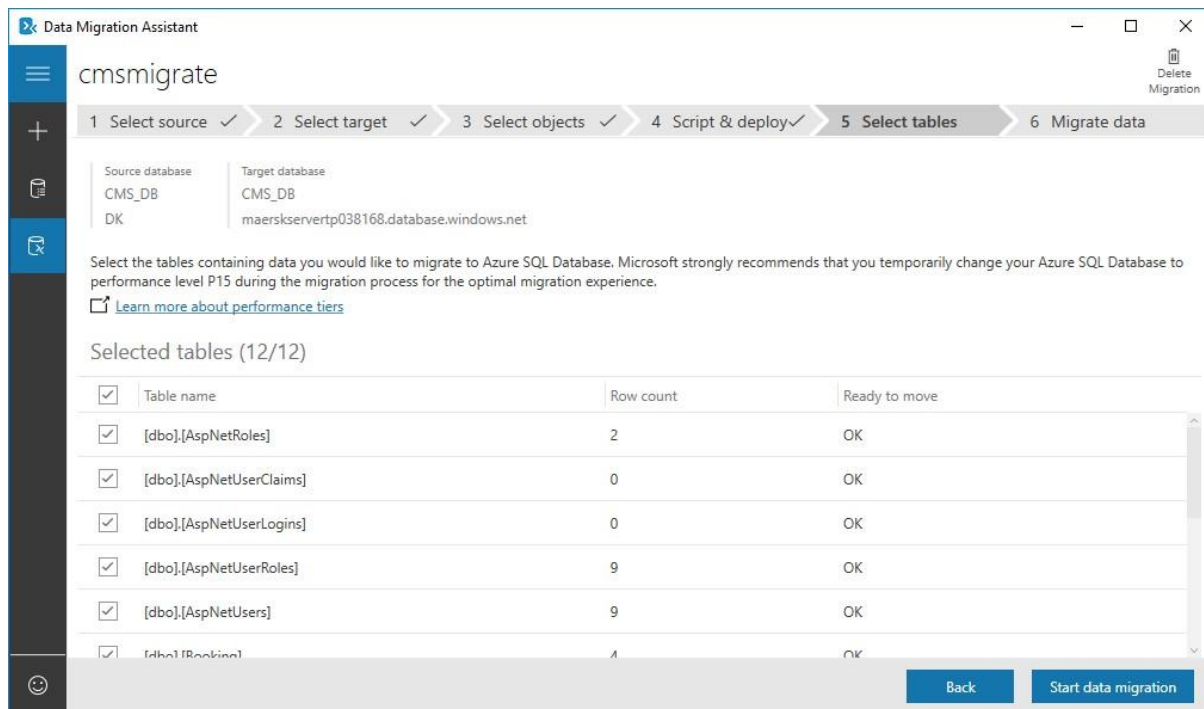
The system will automatically generate the SQL script that re-create the whole database on Azure. If there is no any issue, press “Deploy Schema” to migrate the database to Azure. The following figure shows the migration script of the database in this system.



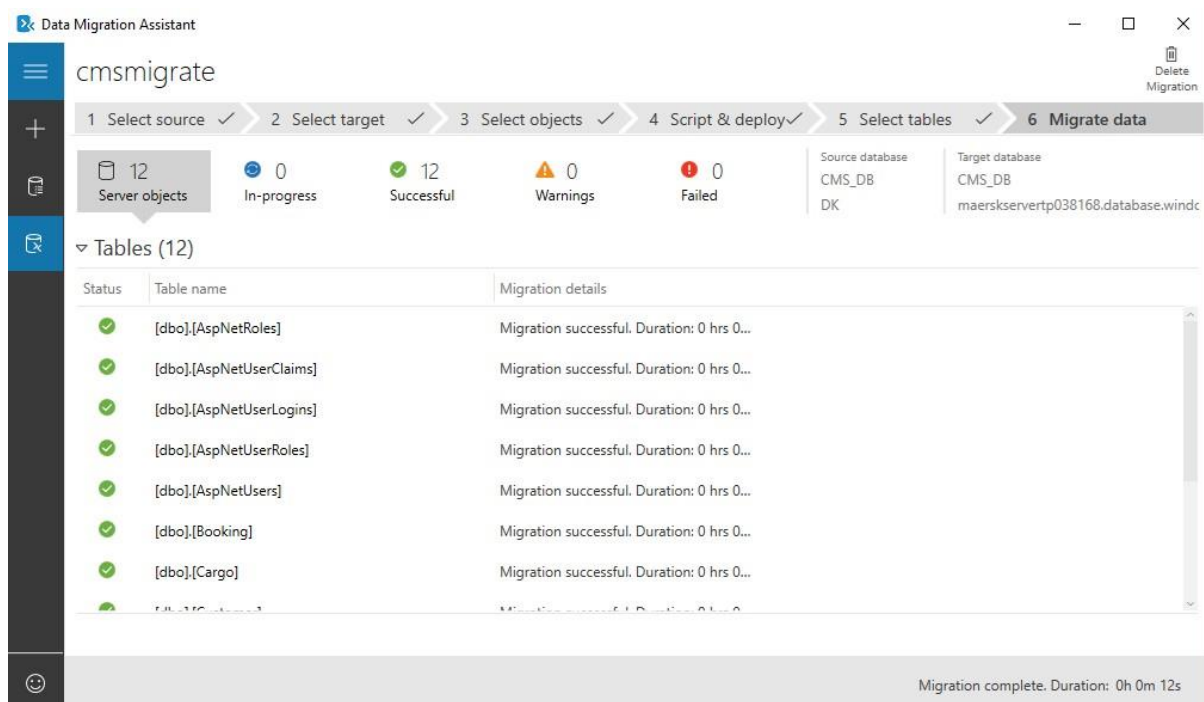
The DMA will execute all the query and ready for the migration of database to Azure. Press “Migrate Data” to proceed to next step.



In this step, DMA will show the status of each table whether it is ready to be migrated to Azure. Again, this is the last confirmation that which table will be migrated to Azure. In this case, tick all tables and press “Start data migration” to complete migration process.

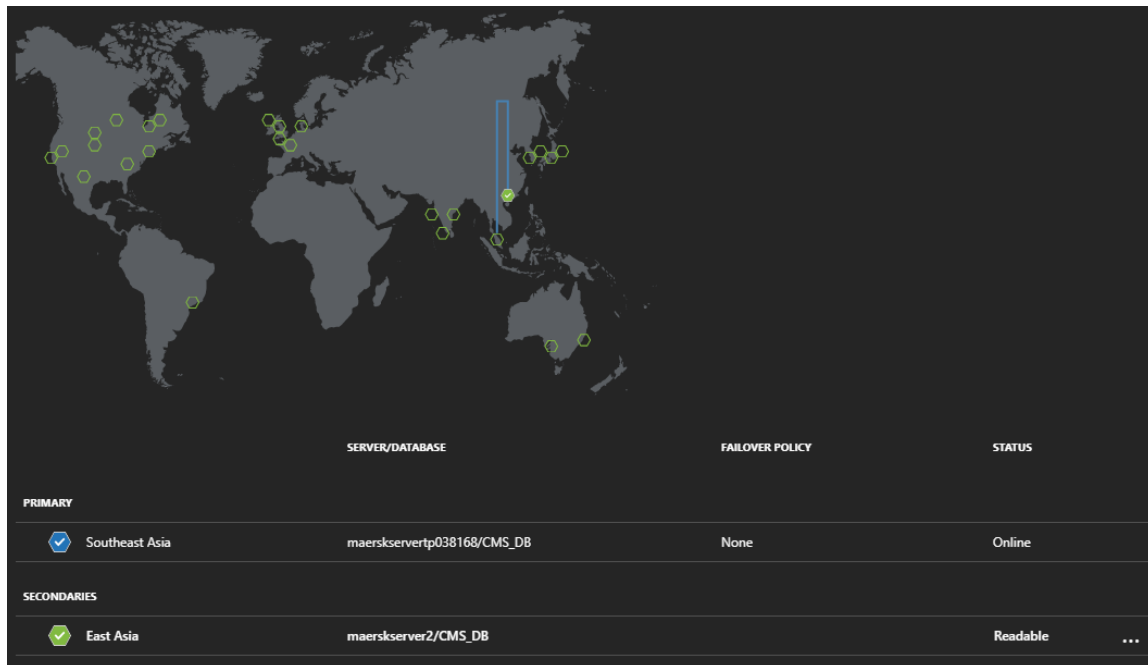


Once done migration, it will show as the figure below. This is the last time of data migration.

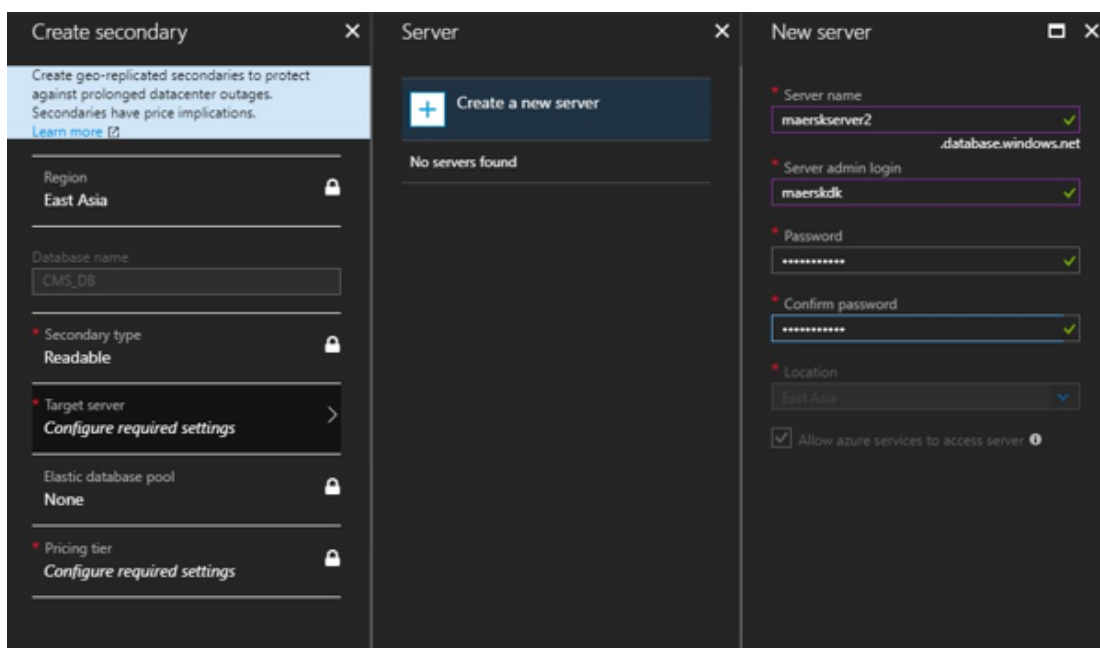


Step 2 – Setup Geo-Replication

Go to SQL Database in Azure that created in previous step. Find “Geo-Replication” in settings to configure the geo-replication of database. Then, press “East Asia” as it is the nearest server location to South East Asia.



A registration form will be shown up and requires the user to fill up the registration of new secondary database. Same as the creation of primary database, type the database name, server name, pricing tier, location, etc. After creation, the configuration of secondary database is done.



Secondary database completely created after setting up the information as figure below.

Create secondary

Create geo-replicated secondaries to protect against prolonged datacenter outages. Secondaries have price implications. [Learn more](#)

Region

East Asia

Database name

CMS_DB

Secondary type

Readable

Target server

maerskserver2 (East Asia)

Elastic database pool

None

Pricing tier

Standard S0: 10 DTUs, 250 GB

















☐ Pin to dashboard

OK

4.3 Application Scaling

Maersk Lin CMS is hosting on S1 Standard price tier and the SQL database is hosting on Standard pricing tier with maximum data size and minimum amount of Database Transaction Unit (DTU).

4.3.1 Web Application Scaling

S1 Standard	S2 Standard	S3 Standard
1 Core	2 Core	4 Core
1.75 GB RAM	3.5 GB RAM	7 GB RAM
 50 GB Storage	 50 GB Storage	 50 GB Storage
 Custom domains / SSL SNI Incl & IP SSL Support	 Custom domains / SSL SNI Incl & IP SSL Support	 Custom domains / SSL SNI Incl & IP SSL Support
 Up to 10 instance(s) Auto scale	 Up to 10 instance(s) Auto scale	 Up to 10 instance(s) Auto scale
 Daily Backup	 Daily Backup	 Daily Backup
 5 slots Web app staging	 5 slots Web app staging	 5 slots Web app staging
 Traffic Manager Geo availability	 Traffic Manager Geo availability	 Traffic Manager Geo availability
312.48 MYR/MONTH (ESTIMATED)	624.96 MYR/MONTH (ESTIMATED)	1,249.92 MYR/MONTH (ESTIMATED)

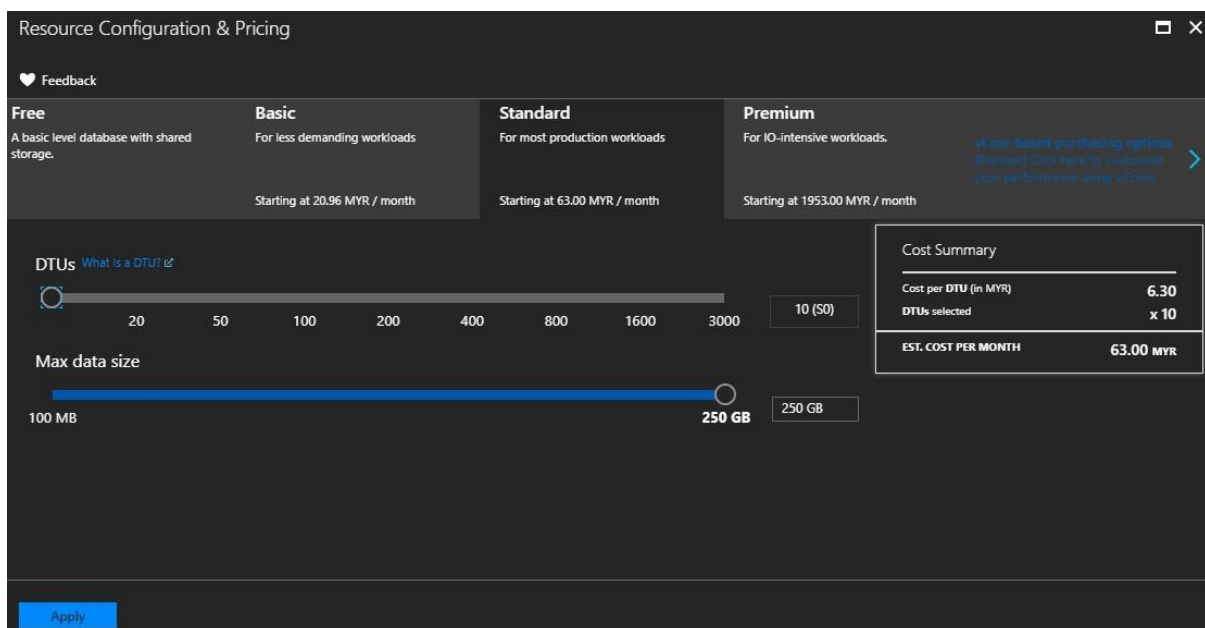
For the Maersk Management Web Application, S1 Standard pricing tier will be used to host the web application. In standard pricing tier, the storage provided for user is 50 GB, compare to basic pricing tier which only provide 10GB storage for user. Standard plan is more reliable and scalable as the system might be improved in the future and the size of the web application will increase as well. Therefore, the basic plan is not scalable enough. Besides, the standard plan provides auto scaling for user while standard plan provides manual scaling. Auto scaling is important as the app service will automatically scale up or down the number of virtual machine's instances based on the average performance of the CPU (Microsoft Azure, 2018). Unlike basic plan, the user has to manually adjust the number of instances.

On top of that, in standard plan, traffic manager is also provided for the user while basic plan does not provide traffic manager. Traffic manager is also an important feature that helps the user to route to the best performed internet location based on the available endpoint (Microsoft Azure, 2017). This could reduce the latency of using this web application. Moreover, Standard plan allows the MAERSK Line to use their own domain name instead of the default Azure's domain name. Also, Standard plan provides the feature of daily backup for the system

and 5 slots of Web App staging. The benefit of deploying the web application in slot is to validate the system changes in a staging development before swapping it with the development slot. It allows the developer to test the system first in another slot.

4.3.2 SQL Database Scaling

The SQL Database will be using the standard plan for initial deployment phase which does not require much storage and computation. In the current situation, there will be 10 DTUs and 100 MB provided for storage size and this is sufficient enough for the initial deployment testing.



The above figure shows the current scale of the SQL database in Azure. The basic plan of this Azure SQL Database service is not suitable for this system. This is because the maximum amount of database size is only 2GB and only 5 DTUs are provided. It is not enough for this system to handle the large amount of data from different countries and the performance of the database might slow down. For the current state, standard plan is the most suitable plan to be used for initial deployment stage as it is slowly deploying to other country.

As the business grows, more data and DTU might be required to be assigned to the system depends on the situation. If current plan is not sufficient to support the larger load of the system, then a greater plan such as Premium tier will be considered in the future in order to handle the greater amount of transaction.

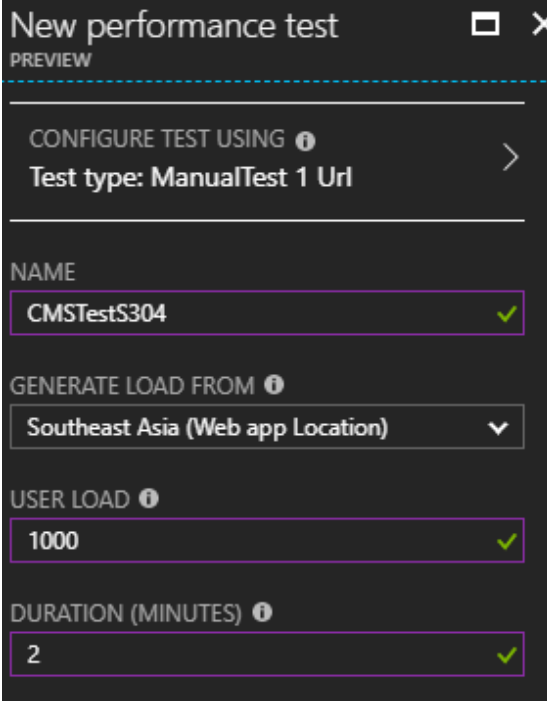
5.0 Test Plan & Testing Discussion

5.1 Unit Testing

Test Case ID	Test Parameter	Test Description	Expected Result	Actual Result	Status
A1	Login	Enter correct credentials	Login Successfully	Login Successfully	Pass
		Enter wrong credential	Login failed	Login failed	Pass
A2	Register new user	Submit incomplete registration form	Show error message	Show error message	Pass
		Submit complete registration form	New data saved to database successfully	New data saved to database successfully	Pass
		Submit the form with existing e-mail	Show error message	Show error message	Pass

5.2 Performance

By hosting the system on Azure, performance testing can be done within a few clicks with various configurations that facilitates the analysing processes. In this performance testing, the system will be using three different plans within the Standard pricing tier for comparison purposes and performance testing will be conducted within 2 minutes with 250, 500, 750, and 1000 user load. Figure on the bottom shows the configuration for one of the performance testing done. The performance testing will take up to several minutes to run and generate results. The performance testing results includes the number of successful and failed requests, performance under load, and web app usage.



New performance test

PREVIEW

CONFIGURE TEST USING ⓘ

Test type: ManualTest 1 Url >

NAME

CMSTestS304 ✓

GENERATE LOAD FROM ⓘ

Southeast Asia (Web app Location) ▼

USER LOAD ⓘ

1000 ✓

DURATION (MINUTES) ⓘ

2 ✓

Figure 7 Performance Testing Configuration

Below shows the result from one of the performance testing.

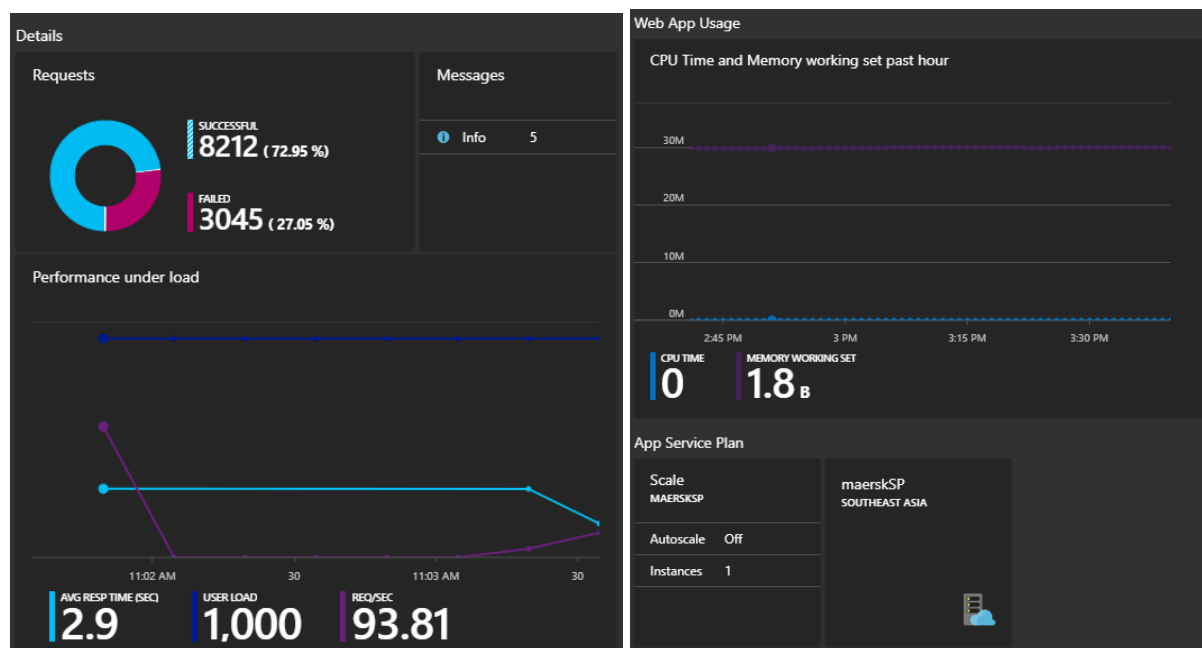


Figure 8 Performance Testing Result

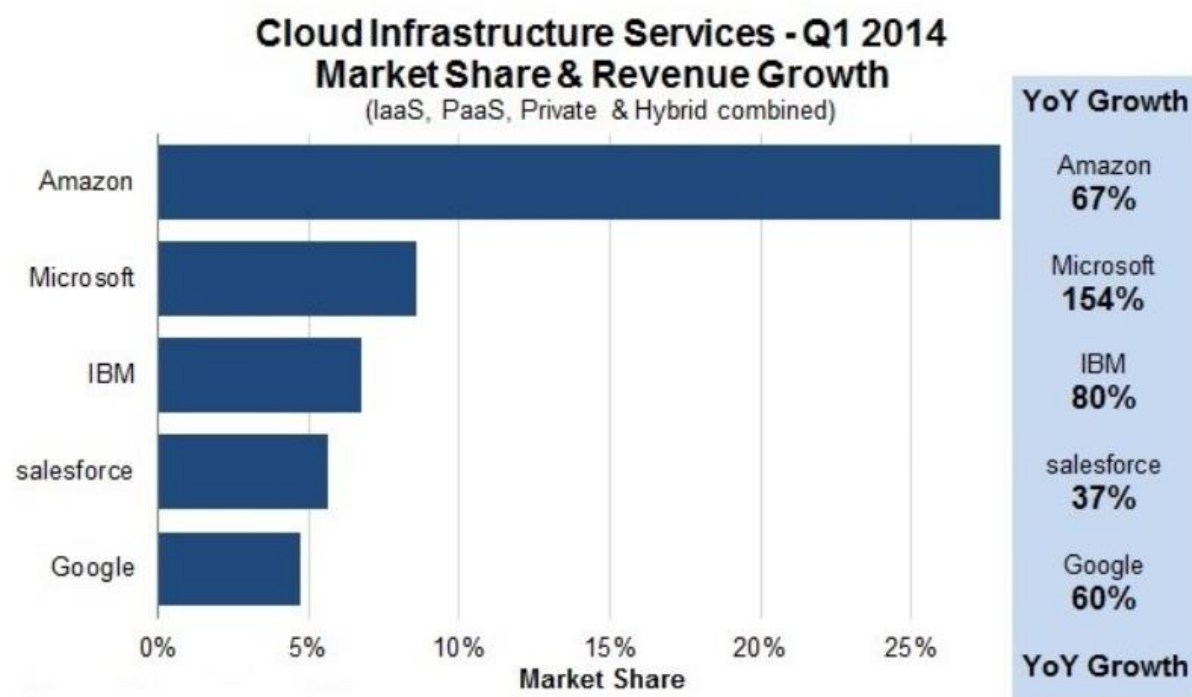
Obviously, different plan within the Standard pricing tier provides different amount of core and RAM. S1 Standard provides 1 Core and 1.75 GB of RAM, while S2 Standard provides double of the Core and RAM provided by S1 Standard and S3 Standard provides double of the Core and RAM provided by S2 Standard. At the bottom shows the comparison table of performance testing done while using different plan within the Standard pricing tier with auto scaling configured.

App Service Plan	Concurrent Users	250	500	750	1000
Aspect of Comparison					
S1	Percentage of failed request	48.38%	43.64%	46.4%	42.7%
	Average response time	0.42 sec	2.1 sec	1.08 sec	2.56 sec
S2	Percentage of failed request	42.52%	42.34%	43.05%	40.92%
	Average response time	1.81 sec	1.67 sec	1.81 sec	1.87 sec
S3	Percentage of failed request	48.4%	31.52%	45.59%	27.05%
	Average response time	0.44 sec	2.49 sec	0.76 sec	2.9 sec

As shown in the table above, the better the app service plan, the better the system performance and the ability to handle concurrent requests. While S1 app service plan performs with average response time ranging from 0.42 seconds to 2.56 seconds, S2 app service plan greatly outperformed S1 app service plan with average response time ranging from 1.81 seconds to 1.87 seconds. However, there are not much improvements when the system is scaled up to S3 service plan which performs with average time ranging from 0.44 seconds to 2.9 seconds. Considering that the pricing for S3 app service plan is double of S2 and S2 is double of S1, it can be concluded that S2 app service plan will be the ideal app service plan for this system. This is because S2 app service plan has a significant improvement compared to S1 app service plan which made the doubled price worth it, but the same case does not apply on the S3 app service plan.

6.0 Managed Databases

Platform as a Service which also known as PaaS, is a cloud model designed for software developers that simplify the development process by shifting specific aspects of systems management to the service provider, it often used to develop web and mobile applications using components that are pre-configured and maintained by the service provider (Sullivan, 2014). Microsoft Azure provide a complete development and deployment environment in the cloud. Besides Microsoft, there are also Amazon, IBM, Google and many others are proving the cloud service in current market. Microsoft is second largest cloud services provider beside the Amazon (Gacinga, 2014).



In infrastructure as a service (IaaS), it offers server/storage, networking firewalls and datacentre infrastructure for the user, these hardware is all provided and managed by the provider, which allow the user to reduce the cost on buying physical servers, etc. PaaS is similar with IaaS, include infrastructure such as servers, storage, etc., it also has middleware, development tools, business intelligence (BI) services, database management systems, etc (Azure, 2018).

7.0 Conclusion

In conclusion, the development and deployment of Maersk Line Container Management System are completed successfully with all the requirements met. Throughout this project, the developer has explored the Cloud Computing knowledge region, which is entirely new for the developer. From only concerning with application development, the developer has widened his capability to deploy and host the application on cloud services. The developer has learned the basic concepts of cloud computing and practiced with actual implementation on Microsoft Azure. Other than learning and practicing the theoretical and practical operations of hosting, the developer has utilized the tools and services provided by Microsoft Azure to ease the development, deployment, and testing processes. All in all, this project exposed the developer to the cloud computing knowledge base which prepares the developer for future career as cloud computing has become commonly used in the industry nowadays.

Reference

Azure, M., 2018. *What is PaaS?*. [Online] Available at: <https://azure.microsoft.com/en-us/overview/what-is-paas/> [Accessed 10 4 2018].

Gacinga, J., 2014. *Microsoft Corporation Becomes the Second-Largest Cloud Services Provider*. [Online] Available at: <https://www.fool.com/investing/general/2014/05/11/microsoft-becomes-second-largest-cloud-services-pr.aspx> [Accessed 10 4 2018].

Sullivan, D., 2014. *PaaS Providers List: Comparison And Guide*. [Online] Available at: <http://www.tomsitpro.com/articles/paas-providers,1-1517.html> [Accessed 10 4 2018].

Microsoft Azure (2017). *What is Traffic Manager*. [Online] Available from: <https://docs.microsoft.com/en-us/azure/traffic-manager/traffic-manager-overview>. [Accessed: 11 April 2018].

Microsoft Azure (2018). *Failover groups and active geo-replication - Azure SQL Database*. [Online]. Available from: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-geo-replication-overview>. [Accessed: 11 April 2018].

Microsoft Azure, 2017. *Autoscaling*. [Online]. Available from: <https://docs.microsoft.com/en-us/azure/architecture/best-practices/auto-scaling> [Accessed 11 April 2018].