Designing & Developing Cloud Applications

Name : Li Kai Mun

ID number : TP033932

Intake Code : UC3F1706SE

Course Code : CT071-5-3-DDAC

Lecturer Name : DR.KALAI ANAND A/L RATNAM

Due Date : 13 – APRIL - 2018

Link: <https://ddacmaersk.azurewebsites.net/>

<http://ddac1-tm.trafficmanager.net/>

http://ddacmaersksea.azurewebsites.net/

Video: <https://web.microsoftstream.com/video/8d8365ad-6740-4319-b634-09446ec150bf>

Github: https://github.com/rkl00102/maersk

**Acknowledgement**

I would like to express my gratitude to my module lecturer, Dr Kalai for providing me the knowledge about Microsoft Azure and cloud related topics. Without his proper guidance, I would not be able to publish MAERSK web app to the Azure cloud server. I am glad that I have learned new technical skills through this module which will be useful for my career in future.

Contents

[1.0 Introduction 4](#_Toc511229670)

[1.1 Background 4](#_Toc511229671)

[1.2 Objectives & Scope 5](#_Toc511229672)

[1.3 Specification 5](#_Toc511229673)

[1.4 Deliverables 5](#_Toc511229674)

[2.0 Project Plan 6](#_Toc511229675)

[3.0 Design 7](#_Toc511229676)

[3.1 Architectural Diagram 7](#_Toc511229677)

[3.2 Design Considerations 8](#_Toc511229678)

[3.3 Use Case Diagram 9](#_Toc511229679)

[3.4 Use Case Specification 10](#_Toc511229680)

[3.5 Sequence Diagram 20](#_Toc511229681)

[4.0 Implementation 27](#_Toc511229682)

[4.1 Publishing web application to Azure 27](#_Toc511229683)

[4.2 Application Scaling 33](#_Toc511229684)

[4.3 Functional Test 36](#_Toc511229685)

[4.3.1 Unit Test 36](#_Toc511229686)

[4.3.2 Performance Test 40](#_Toc511229687)

[4.4 SQL Database 44](#_Toc511229688)

[5.0 Conclusion 47](#_Toc511229689)

[6.0 References 48](#_Toc511229690)

# 1.0 Introduction

## 1.1 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.

In an effort to support further business growth and increase organizational flexibility, Maersk decided to consolidate all of its data centers 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 Objectives & Scope

The project aims to develop an online web application using MVC for Maersk Line and deploy it to the cloud using Azure Cloud Services. The deployed web application will be accessible from anywhere as long as internet connection is available. Besides, the developer demonstrates how the web application fits into Microsoft Azure. Lastly, execute different types of test to ensure the web application meets users need and specification. The components provided by Azure should be well used in order to enhance the web application.

## 1.3 Specification

* Web application is high availability and has good performance which able to handle large amount of user requests.
* Auto-scaling should be included to assure performance of web application.
* Traffic manager is used to control the distributions of network.
* Web application allows user to book schedule of shipment.
* Web application is accessible from more than one region.

## 1.4 Deliverables

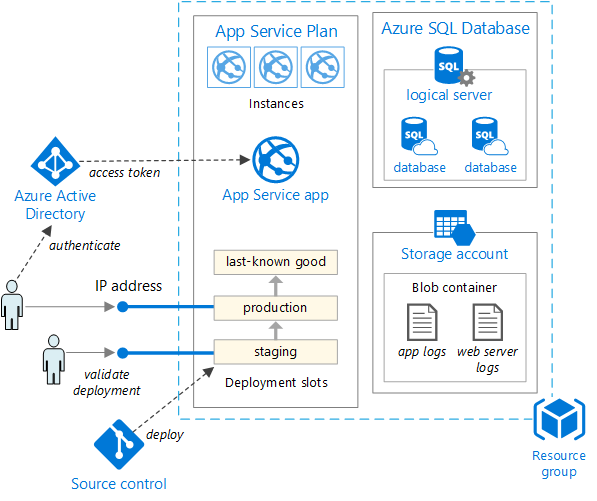
* Allow users to manage vessels. This includes create, update, read and delete.
* Allow users to manage shipment schedules. This includes create, update, read and delete.
* Allow users to book available shipment schedules. This includes create, update, read and delete.

# 2.0 Project Plan

|  |  |  |  |
| --- | --- | --- | --- |
| Task Name | Duration | Start | Finish |
| **Introduction** | **1 day** | **Sat 17/3/18** | **Sun 18/3/18** |
| Project Background | 1 hr | **Sat 17/3/18** | **Sun 18/3/18** |
| Objective & Scopes | 1 hr | **Sat 17/3/18** | **Sun 18/3/18** |
| Specification | 1 hr | **Sat 17/3/18** | **Sun 18/3/18** |
| Deliverables | 1 hr | **Sat 17/3/18** | **Sun 18/3/18** |
| **Design** | **15.25 days** | **Thu 9/7/17** | **Thu 9/21/17** |
| Architectural Diagram | 2 Days | **Mon 19/3/18** | **Wed 21/3/18** |
| Design Consideration | 1 hr | **Mon 19/3/18** | **Wed 21/3/18** |
| Use Case | 1 hr | **Mon 19/3/18** | **Wed 21/3/18** |
| Use Case Description | 2 hr | **Mon 19/3/18** | **Wed 21/3/18** |
| Sequence Diagram | 3 hr | **Mon 19/3/18** | **Wed 21/3/18** |
| **Implementation** | **5 Days** | **Fri 23/3/18** | **Wed 28/3/18** |
| Publish to Azure | **3 Days** | **Fri 23/3/18** | **Wed 28/3/18** |
| Application Scaling | **5 hrs** | **Fri 23/3/18** | **Wed 28/3/18** |
| Unit & Performance Test | **5 hrs** | **Fri 23/3/18** | **Wed 28/3/18** |
| SQL Database | **1 day** | **Fri 23/3/18** | **Wed 28/3/18** |
| **Conclusion** | **1 day** | **Wed 11/4/18** | **Wed 11/4/18** |

# 3.0 Design

## 3.1 Architectural Diagram



**Figure 1: Architecture of ASP.NET and Azure**

*Source: (https://docs.microsoft.com/en-us/dotnet/standard/modern-web-apps-azure-architecture/common-web-application-architectures)*

The figure above shows the architecture of this Azure. It consists of app service plan, SQL database, SQL server, app service and resource group.

## 3.2 Design Considerations

The developer has decided to use ASP.NET MVC framework as the programming language for this project. The reason being is that this framework is able to accelerate the development process. Technically, the MVC uses entity framework which able to generate the basic CRUD operations based on the models created. Since this application requires many CRUD operation, thus using this framework reduces a lot of time for the development process. Besides, this framework helps to prevent human error as the codes are auto generated based on the model. This helps developer to save time from fixing bugs. Moreover, ASP.NET MVC greatly integrate with Microsoft Azure and can easily publish the web application to the Azure cloud server. Therefore, using this framework simplifies the process of publishing web application to the cloud server and able to achieve project goal easier.

On the other hand, developer decided to deploy the web application to both Central US and SEA region to increase availability. That said, users are able to access to the web application regardless of locations. Deploying on more than one region can also increase the performance of web application because more requests can be handled.

Lastly, the user interface of the web application is simple and easy to use. It consists only three functions which is manage vessels, schedules and also bookings. The reason being is that the objective of this project is to focus on enhancing the web application rather than the design. Therefore, developer wants to put more effort on Azure solution.

## 3.3 Use Case Diagram



## 3.4 Use Case Specification

|  |  |
| --- | --- |
| **Name** | **Login** |
| Actor | Agent |
| Priority | High |
| Description | Agent log into the system in order to access to the functionalities. |
| Extends | Logout |
| Includes | None |
| Pre-condition | 1. The account being logged in is existed. |
| Post-condition | 1. The agent is logged into the system.  2. The agent can now manage vessels, schedules and bookings. |
| Flow of Events | 1. Enter registered email.  2. Enter valid password.  3. Invalid email / password [A1] |
| Alternative Flows | 1. A1. Re-enter email and password correctly |

|  |  |
| --- | --- |
| **Name** | **Manage Vessel** |
| Actor | Agent |
| Priority | High |
| Description | Agents are allowed to manage vessels |
| Extends | Create vessel, edit vessel, delete vessel |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. Agent navigates to vessel management page. |
| Post-condition | 1. The agent might have added, edited or deleted vessels. |
| Flow of Events | 1. Navigate to vessel page.  2. Choose an operation. |
| Alternative Flows | 1. Request unauthorized users to login.  2. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Create Vessel** |
| Actor | Agent |
| Priority | High |
| Description | Create new vessel into the database |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. The name of vessel being created is available. |
| Post-condition | 1. The particular vessel is registered into the database.  2. The new vessel is now available for shipment scheduling. |
| Flow of Events | 1. Enter vessel name.  2. Enter capacity of the vessel. |
| Alternative Flows | 1. Re-enter vessel name if it is already existed in the database.  2. Re-enter capacity if the input is not integer value. |

|  |  |
| --- | --- |
| **Name** | **Delete Vessel** |
| Actor | Agent |
| Priority | High |
| Description | Delete available vessels |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. List of vessels are shown. |
| Post-condition | 1. The selected vessel will be deleted. |
| Flow of Events | 1. Select particular vessel to delete.  2. Delete vessel from the database. |
| Alternative Flows | 1. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Edit Vessel** |
| Actor | Agent |
| Priority | High |
| Description | Edit the details of available vessels |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. List of vessels are shown. |
| Post-condition | 1. The new details of vessels will be saved into database. |
| Flow of Events | 1. Select particular vessel to edit.  2. Change the details of vessel. It can only be the capacity. |
| Alternative Flows | 1. Re-enter capacity if the input is not integer value. |

|  |  |
| --- | --- |
| **Name** | **Manage Schedule** |
| Actor | Agent |
| Priority | High |
| Description | Agents are allowed to manage schedules. |
| Extends | Create, edit and delete schedules. |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. Agent navigates to schedule management page. |
| Post-condition | 1. The agent might have added, edited or deleted schedules. |
| Flow of Events | 1. Navigate to schedules page.  2. Choose an operation. |
| Alternative Flows | 1. Request unauthorized users to login.  2. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Create Schedule** |
| Actor | Agent |
| Priority | High |
| Description | Create new schedule into the database |
| Extends | None |
| Includes | Available Vessels |
| Pre-condition | 1. The valid agent is logged into the system.  2. Vessels are already existed in the database. |
| Post-condition | 1. New schedule is registered into the database.  2. The new schedule is now available for booking. |
| Flow of Events | 1. Select departure time  2. Select arrival time  3. Select departure location  4. Select arrival location  5. Select vessels |
| Alternative Flows | 1. Create vessels if none of them are created. |

|  |  |
| --- | --- |
| **Name** | **Edit Schedule** |
| Actor | Agent |
| Priority | High |
| Description | Edit the details of available schedule |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. List of schedules are shown. |
| Post-condition | 1. The new details of schedules will be saved into database. |
| Flow of Events | 1. Select a particular schedule to edit.  2. Choose new departure date or arrival date.  3. Set new departure location or arrival location.  4. Select new vessels |
| Alternative Flows | 1. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Delete Schedule** |
| Actor | Agent |
| Priority | High |
| Description | Delete available schedules |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. List of schedules are shown. |
| Post-condition | 1. The selected schedule will be deleted. |
| Flow of Events | 1. Select particular schedule to delete.  2. Delete schedule from the database. |
| Alternative Flows | 1. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Manage Booking** |
| Actor | Agent |
| Priority | High |
| Description | Agents are allowed to manage bookings |
| Extends | Create, edit and delete bookings |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. Agent navigates to booking management page. |
| Post-condition | 1. The agent might have added, edited or deleted bookings. |
| Flow of Events | 1. Navigate to bookings page.  2. Choose an operation. |
| Alternative Flows | 1. Request unauthorized users to login.  2. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Create Booking** |
| Actor | Agent |
| Priority | High |
| Description | Create new bookings into the database |
| Extends | None |
| Includes | Available Schedule |
| Pre-condition | 1. The valid agent is logged into the system.  2. Schedules are already existed in the database. |
| Post-condition | 1. New booking is registered into the database. |
| Flow of Events | 1. Enter cargo size.  2. Enter number of containers.  3. Enter customer name.  4. Select listed schedule. |
| Alternative Flows | 1. Create schedules if none of them are created. |

|  |  |
| --- | --- |
| **Name** | **Edit Booking** |
| Actor | Agent |
| Priority | High |
| Description | Edit the details of available bookings |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. List of bookings are shown. |
| Post-condition | 1. The new details of bookings will be saved into database. |
| Flow of Events | 1. Select a particular bookings to edit.  2. Enter new cargo size or number of containers.  3. Select new schedules. |
| Alternative Flows | 1. Redirect authorized users to login page when they attempt to use the CRUD operation. |

|  |  |
| --- | --- |
| **Name** | **Delete Booking** |
| Actor | Agent |
| Priority | High |
| Description | Delete available bookings |
| Extends | None |
| Includes | None |
| Pre-condition | 1. The valid agent is logged into the system.  2. List of bookings are shown. |
| Post-condition | 1. The selected booking will be deleted. |
| Flow of Events | 1. Select particular booking to delete.  2. Delete booking from the database. |
| Alternative Flows | 1. Redirect authorized users to login page when they attempt to use the CRUD operation. |

## 3.5 Sequence Diagram

**Login**



**Create Vessel**



**Edit Vessel**



**Create Schedule**



**Edit Schedule**



**Create Booking**



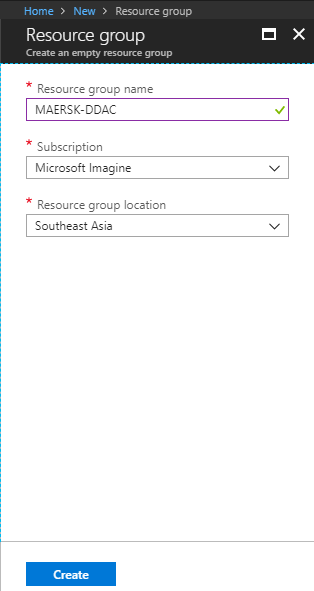
**Edit Booking**



# 4.0 Implementation

## 4.1 Publishing web application to Azure

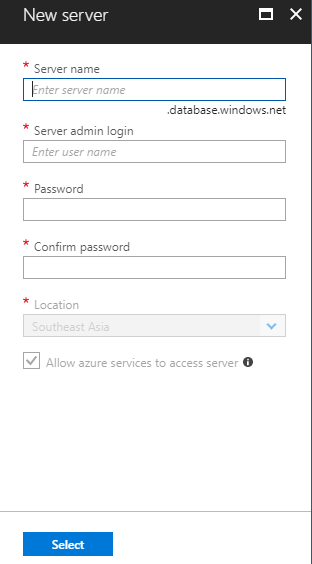
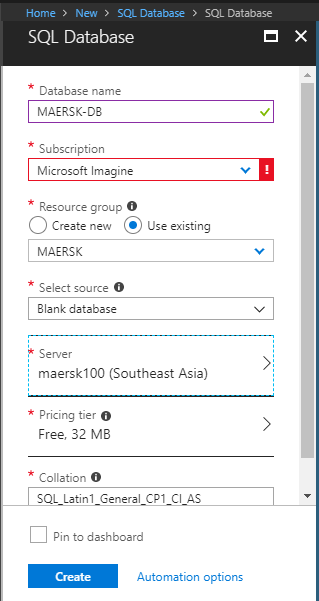
**Step 1: Create Resource Group**



**Figure 2: Create Resource Group**

In order to publish web application to Azure, resource group has to be first created. Resource group is a container that stores related resources for an Azure solution. It allows the developer to manage and monitor these resources as a group. For this project, web app, SQL server and Azure Database have been stored in the resource group. Additional resource that may enhance the application can be added anytime in the future. There are three inputs required while creating the resource group. Firstly, the name of the resource group must be new and distinctive since one Azure account can contain more than one resource group. Existed name will be prohibited while creating the resource group. Secondly, select the type of subscription that suits the application the most. In this project, Microsoft Imagine is selected as it is free yet sufficient for the web application. Lastly, select the region to determine which group of users are allowed to access to the web application.

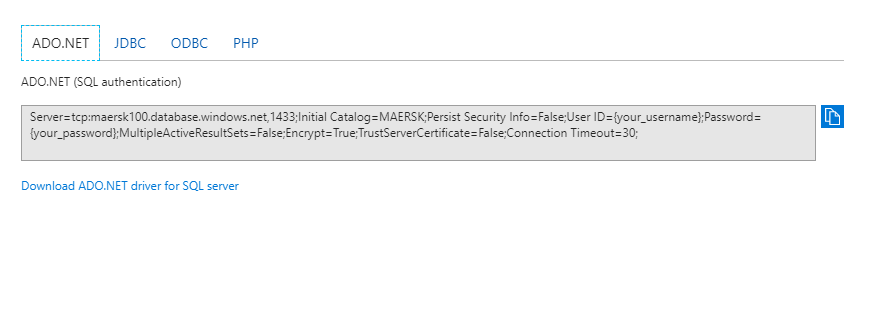
**Step 2: Create SQL Database**



**Figure 3: SQL Database**

The next step is to create SQL database for the web application to store all related data. There are multiple inputs required while creating the database. Similar to resource group, the name of the database has to be new and distinctive as well. The next thing is to choose the subscription that suits the database. In this case, Microsoft Imagine is chosen as it is free yet effective for this web application. Moving on, developer needs to specify the resource group that will manage this database. Server has to be created in order to host the database. Simple details such as server name, username and password are required for creating new server. The username and password registered must be remembered as they will be used for connecting the SQL management studio which will manage the database. Lastly, select the pricing tier for the database. There are variety of plans available for database but since the subscription is Microsoft Imagine, the plan provided is free which allocated 32MB. The plan differs according to the types of subscription.

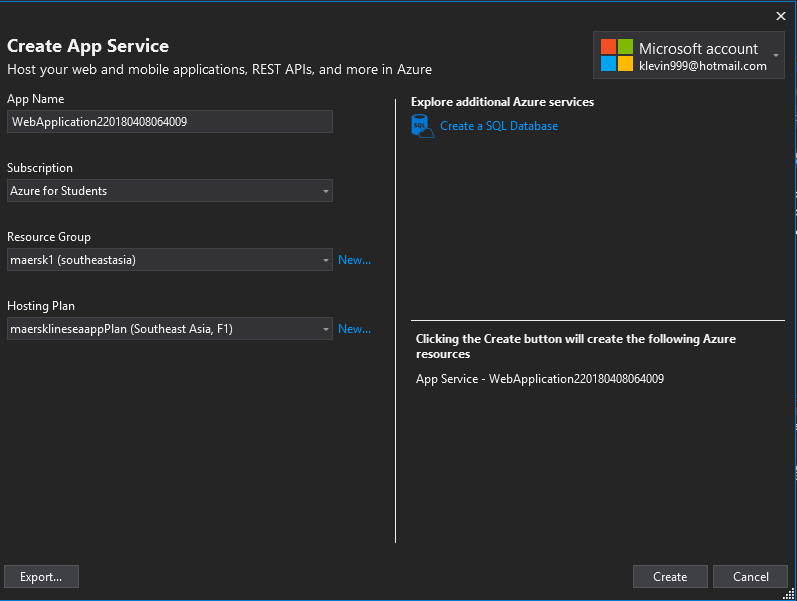
**Step 3: Connect to SQL Database**



**Figure 4: Connection String**

The next step is to connect the web application to SQL database. In order to achieve this, the connection strings provided by Azure need to be copied and replaced with the current one in ASP.NET. The username and password must match with the SQL server which created previously in order to connect successfully. After connecting to the SQL server, the web application in Visual Studio is linked with Microsoft Azure. The developer can easily publish the web application to Azure once the connection is settled.

**Step 4: Publish**

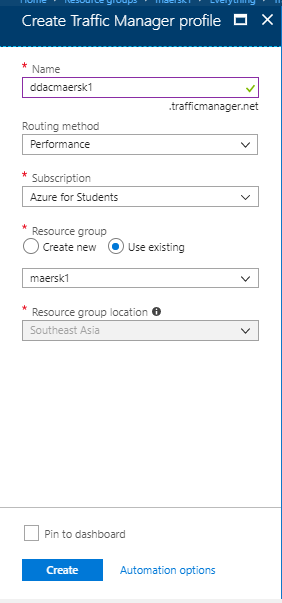


**Figure 5: Create new app service**

The final step is to publish the web application to Azure. Firstly, new app server has to be created first. The details such as subscription, resource group and hosting plan will be retrieved once login in Microsoft Visual Studio. Secondly, fill in the required details and create the app service. The system will check for app name availability to prevent repetitive name. The app name must be new and distinctive. Afterwards, select the resource group that will handle this web app and create the profile. Once created, the link of the web application will be shown on browser. The web app can be enhanced and configured through Microsoft Azure.

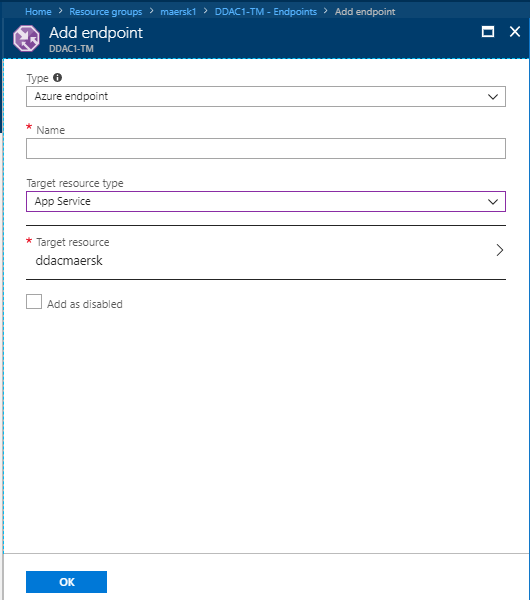
**Step 5: Traffic Manager**

According to Azure, Traffic Manager has the ability to control the distribution of user traffic for service endpoints in different datacenters. It uses the DNS to direct client request to the most suitable endpoint based on a traffic-routing method and the health of the endpoints. The aims of Traffic Manager it to provide a range of traffic-routing methods and endpoint monitoring options to suit different application needs.



**Figure 6: Traffic Manager Creation**

The figure above shows the step to create traffic manager. First, go to the resource group that will hold the traffic manager profile and add a new one. Provide the necessary details for the traffic manager profile such as name, routing method, subscription type. Once every detail has been validated, the new traffic manager profile will be created.



**Figure 6.1: Endpoint creation**

After traffic manager profile is created, now move on and create endpoint. The endpoint is created for traffic manager to monitor the application. This way when the user access the traffic manager link, the traffic manager will choose from these endpoints depending on the routing method chosen. Different region will require different endpoint. In this project, two endpoints are created which used to monitor Central US and SEA region. Only standard and above app service plan will activate the traffic manager status.

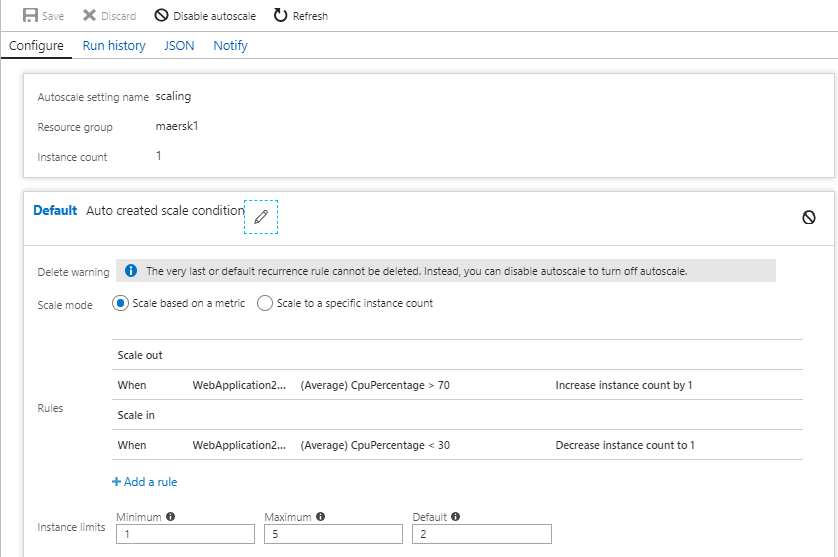
## 4.2 Application Scaling

According to Azure 2017, auto-scaling involves process of allocating resources to match performance requirements dynamically. As the volume of tasks grow, the MAERSK web application requires more resources in order to maintain the desired performance levels and satisfy service-level agreements (SLAs). The great thing about auto-scaling is that extra resources will be de-allocated automatically when they are no longer needed for the web application. This helps the developers to save cost. Additionally, it helps monitoring the server performance all the time and decide whether to allocate or de-allocate the resources. Therefore, it is worth to choose a plan that offers auto-scaling which suitable for MAERSK web application.



**Figure 7: Pricing Tier**

The figure above shows the pricing plan offered by Azure. Each of the plan provides different offers. Both basic and standard plan offer the same core and GB RAM while the only difference is that standard plan provides more features. For MAERSK web application, standard plans will be taken into considerations rather than basic plan as more essential features are provided. First of all, standard plan offers more storage than basic plan which allows the server to handle larger amount of transactions. Moreover, it allows auto-scaling up to 10 instances which is relatively useful as the number of customer increases over time. Additionally, standard plan also provide daily backup to prevent loss of data which is great for the web application as a lot of data will be stored. Furthermore, it provides 5 deployment slot which holds the recent deployment restore point in case any unexpected situation happens. Lastly, the traffic manager is able to handle the connection between client and server to ensure the stable communication by connecting them to the nearest region.



**Figure 7.1: Scale-out Plan**

The figure above shows the scale-out plan for MAERSK web application. The plan created is whenever the CPU usage is greater than 70%, one instance will be increased to handle the extra work load and vice versa an instance will be decreased whenever the CPU usage is lower than 30%. The plan can be further improved by subscribing to higher tier plan such as premium or isolated plans as they provide additional instances for scaling.

## 4.3 Functional Test

### 4.3.1 Unit Test

**Login**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| LT1 | username: klevin  password: tmnet123 | Username and password correct | Login successful | Login successful | Pass | High |
| LT2 | username: klevin  password: Tmnet123 | Username correct but password wrong | Login failed. Error message is displayed | Login failed. Error message is displayed | Pass | High |
| LT3 | username:kklevin  password:tmnet123 | username wrong but password correct | Login failed. Error message is displayed | Login failed. Error message is displayed | Pass | High |
| LT4 | username:kklevin  password: | Password field is empty | Login failed. Error message is displayed | Login failed. Error message is displayed | Pass | High |
| LT5 | username:  password:tmnet123 | Username is empty | Login failed. Error message is displayed | Login failed. Error message is displayed | Pass | High |

**Create Vessel**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| CV1 | Name: ShipA  Capacity: 20 | Valid name and capacity | Register successful | Registered successful | Pass | High |
| CV2 | Name: ShipA  Capacity: a | Valid name and invalid capacity | Register unsuccessful. Capacity field only allow integer value. | Registered unsuccessful. Capacity field only allow integer value. | Pass | High |
| CV3 | Name:  Capacity: | Both fields are empty | Register unsuccessful. These field are required. | Registered unsuccessful. These field are required. | Pass | High |

**Edit Vessel**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| EV1 | Name: ShipA  New Name: ShipB  Capacity: 20 | Valid name and capacity | Edit successful | Edited successful | Pass | High |
| EV2 | Name: ShipA  New Name: ShipB  Capacity: a | Valid name and invalid capacity | Edit unsuccessful. Capacity field only allow integer value. | Unsuccessful. Capacity field only allow integer value. | Pass | High |

**Create Schedule**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| CS1 | DepartTime: 10/3/2018  ArrivalTime: 11/3/2018  DepartLocation: KL  ArriveLocation: Ipoh  VesselID: ShipB | Valid inputs | Schedule created | Schedule created successfully | Pass | High |
| CS2 | DepartTime: 10/3/2018  ArrivalTime: 09/3/2018  DepartLocation: KL  ArriveLocation: Ipoh  VesselID: ShipB | Arrival time is before departure time. | Fail to create schedule as departure time exceeds arrival time. | Schedule created successfully | Fail | High |
| CS3 | DepartTime:  ArrivalTime:  DepartLocation:  ArriveLocation:  VesselID: | Fields are empty | Register unsuccessful. These field are required. | Registered unsuccessful. These field are required. | Pass | High |

**Edit Schedule**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| ES1 | New DepartTime:20/3/18  New ArrivalTime: 21/3/18 | Valid inputs | Edit successful | Edited successful | Pass | High |

**Create Booking**

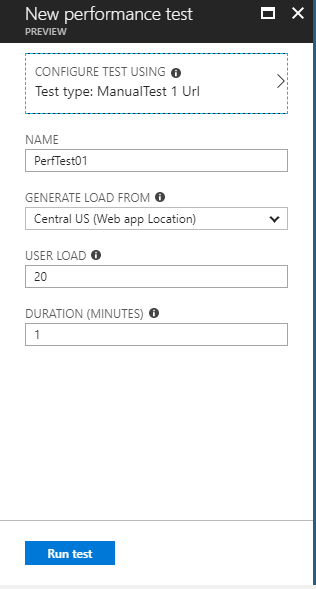
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| CB1 | CargoSize: 20  Cotainer: 20  Customer: Kelvin  Schedule: 3/6/2018 KL->Selangor | Valid inputs | Booking created | Booking created successfully | Pass | High |
| CB2 | CargoSize: abc  Container: 20  Customer: Kelvin  Schedule: 3/6/2018 KL->Selangor | Cargo size is not integer value | Only integer value is allowed in cargo size field | Error message displayed: Only integer value is allowed in cargo size field | Fail | High |
| CB3 | CargoSize: 20  Container: abc  Customer: Kelvin  Schedule: 3/6/2018 KL->Selangor | Container field is not integer value | Only integer value is allowed in container field | Error message displayed: Only integer value is allowed in container field. | Fail | High |

**Edit Booking**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case ID | Conditions | Description | Expected Result | Actual Result | Status | Priority |
| EB1 | New schedule: 3/7/2018 KL->Selangor | Valid inputs | Edit successful | Edited successful | Pass | High |

### 4.3.2 Performance Test

The performance of MAERSK web app can be tested through conducting performance test on Azure. Performance test works by allocating certain amount of virtual users to visit the website. Different application plan will have different amount of virtual users available. The free plan only allow maximum of 40 users to be tested. Besides, the duration is only limited to 1 minute with the free plan. On the other hand, S1 plan allows maximum of 250 users to be tested and the duration is limited to 5 minutes. For this web application, the free plan and S1 plan will be used to conduct performance test and find out the difference and determine which is suitable for this web application.

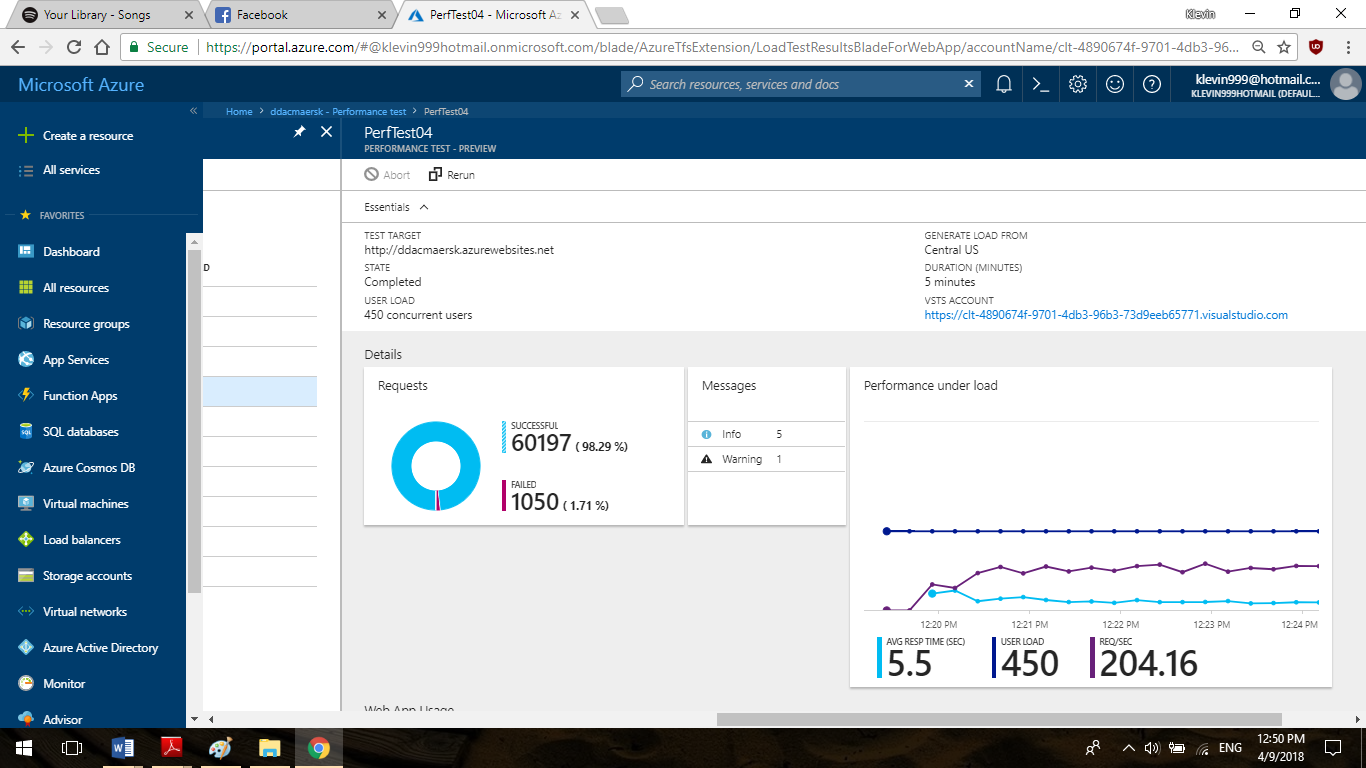


**Figure 8: Adding new performance test**

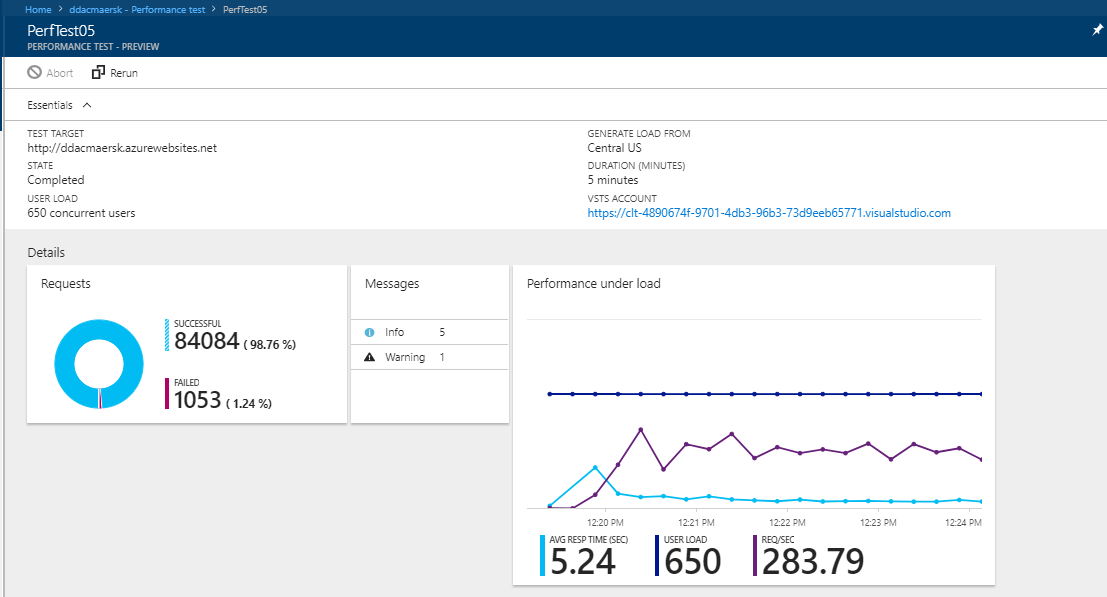
The figure above shows the performance test creation. Firstly, provide a name for the performance test as different name have different criteria for testing. Click on configure test and provide the link of web app to be tested. After that, allocate the amount of virtual users involved and duration. Lastly, click on run test to execute the performance test. The test will take approximately 15 minutes to finish depending on the duration and amount of users. The results will be analyzed in the following section.

**Investigate & Analyze Application**

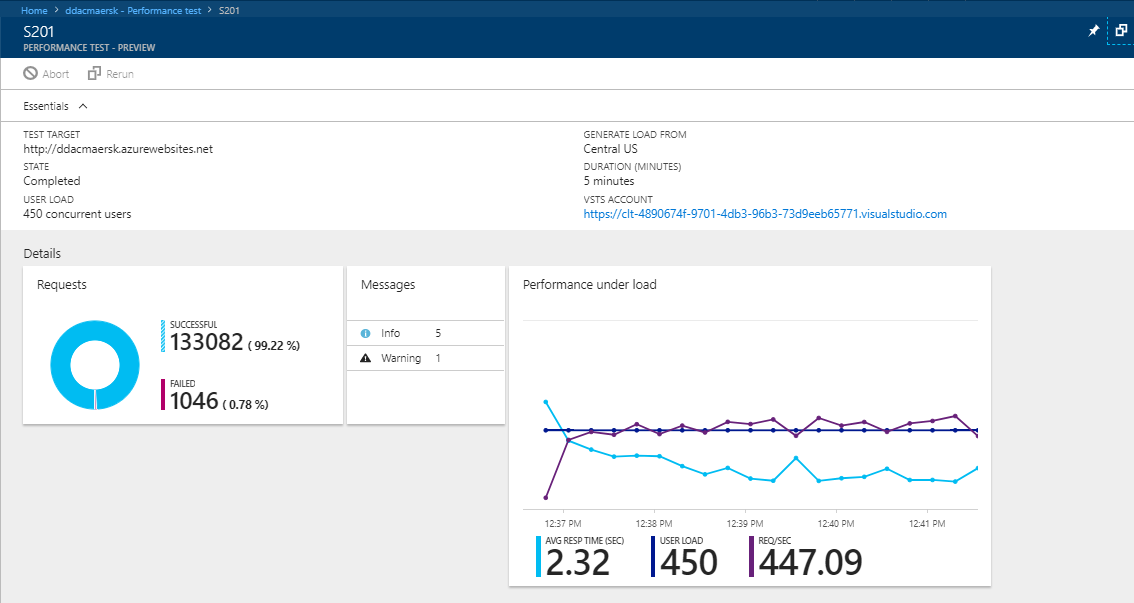
After the performance test is done, the results are used to compare as developer want to select the service plan that suitable for the application. Four performance test have been performed for this web application using two different plans. S1 and S2 plans are used and tested with 450 and 650 concurrent users at duration of 5 minutes. The figure below are the screenshot for results of the performance tests.



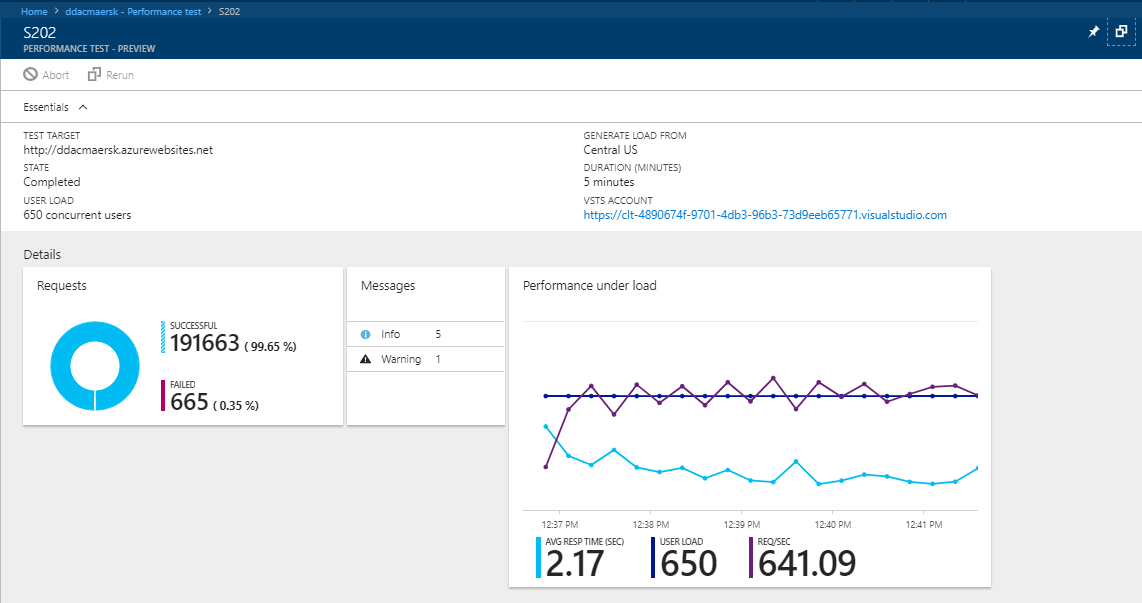
**Figure 8.1: S1 450 Users**



**Figure 8.2: S1 650 Users**



**Figure 8.3: S2 450 Users**



**Figure 8.4: S2 650 Users**

|  |  |  |
| --- | --- | --- |
| Plan | Concurrent Users | |
| 450 | 650 |
| Standard 1 | 5.5  1050 Failed | 5.24  1053 Failed |
| Standard 2 | 2.32  1046 Failed | 2.17  665 Failed |

**Figure 8.5: Results of performance test**

Based on the results of these four performance tests, it can be concluded that the higher tier plan is greater at handling large amount of concurrent users. The average response time of the app service while using Standard 1 plan is around 5.5 and 5.24. The success rate of handling requests is around 98%. However, the number of failed request is high which is not good for the web application. On the other hand, the average response time of the app service while using Standard 2 is much faster compared to Standard 1. It takes only about 2.17 sec to handle the requests which is efficient for the web application. Besides, although the number of concurrent users are as high as 650 with total number of requests of 191663, the number of failed request are only 665. Therefore, Standard 2 plan will be used for this web application as it is efficient and the price is reasonable.

# 4.4 SQL Database

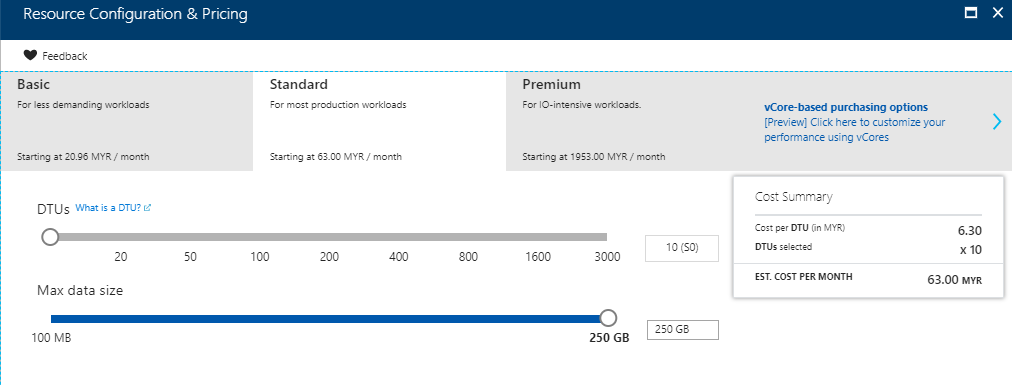
According to (Microsoft, 2017), Azure SQL Database is native to the cloud which serve as a platform as a service (PaaS) database or a database as a service (DBaaS) which is optimized for software-as-a-service (SaaS) app development. It offers compatibility with most of the SQL Server features. It has nothing difference with typical SQL Server which is available across physical on premises machines, private cloud environments, third party hosted private cloud environments and public cloud. The same set of server products, development tools and expertise across theses environment is provided. The hardware and software that is owned by Microsoft is built in the SQL Database. With SQL Database, development can be done directly on the service using built-in features and functionality. When using SQL Database, developer can easily scale up or out for greater power with no interruption.

Azure SQL Database helps in reducing overall costs to the minimum for provisioning and managing many databases. Besides, it reduces the administration costs as virtual machines, operating system and database software will be managed by the cloud. Additionally, Azure SQL Database can dramatically increase the number of databases managed by a single IT or development resource.

**Characteristics of SQL Database**

|  |  |
| --- | --- |
| Characteristics | Description |
| Suitability | * New cloud-designed applications that have to be developed in short time. * Database with high availability, failure recovery and upgrades * Teams that are not familiar with managing operating system and configuration settings * Databases with large storage |
| Resources | * Focus on the application layer only without employ IT resources for configuration and management of the underlying infrastructure. |
| Total cost of ownership | * Reduce administrative costs and hardware costs |
| Business continuity | * Built-in failure tolerance infrastructure capabilities such as automated backups, Point-in-time restore, geo-restore and active geo-replication. |
| Hybrid Cloud | * Data in Azure SQL database can be accessed through on-premises application. |

**Scaling Database**

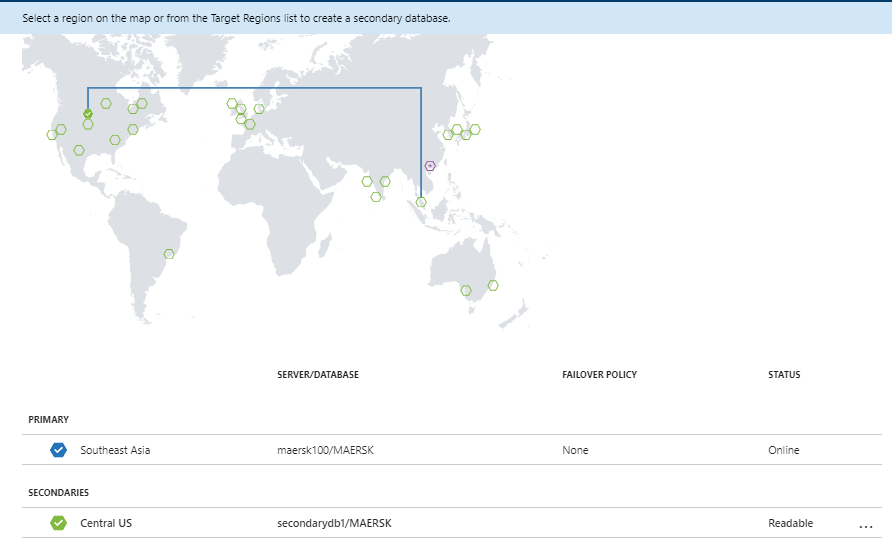


**Figure 9: Scaling**

The SQL database can be configured and change the plan anytime. For MAERSK web application, standard plan is used for the database. This is because it provides 250GB of storage which can store many data. Besides, the cost is relatively cheap. Besides, the standard plan offers geo-replication which is extremely useful for the web application. Further explanation of geo-replication will be explained in the section below.

**Geo-replication**

Azure SQL Database includes geo-replication feature which allows configure up to four readable secondary databases in the same or different data center locations (regions). Whenever the primary database is unable to connect or if there is any failover happens, secondary databases are available for querying and for failover.



**Figure 10: Geo-replication**

The figure above shows the geo-replication of SQL database. As you can see, the SEA region is the primary database while the secondary database is Central US region. If any failure happens on the primary database, the secondary database will be switched and replace the primary database. However, the secondary database is only readable.

# 5.0 Conclusion

To conclude, MAERSK web application has been successfully deployed to Azure cloud. Besides, the application has been enhanced with features such as traffic manager and auto scaling. With Azure SQL Database, the web application can be easily developed as the database is PaaS. Most of the architectures, operating system and maintenance are managed by Azure. Therefore, it has accelerated the development process and also the deployment process. Throughout this project, developer has gained more knowledge regarding to cloud services. Developer also learned about the differences between IaaS, PaaS and SaaS. Last but not least, million thanks to Microsoft Azure for providing 100USD for students and allow them to have a chance to test out the components provided in Azure.

# 6.0 References

CarlRabeler. 2018. Azure portal: SQL Database geo-replication | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-geo-replication-portal>. [Accessed 11 April 2018].

tdykstra. 2018. Getting Started with Entity Framework 6 Code First using MVC 5 | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/getting-started-with-ef-using-mvc/creating-an-entity-framework-data-model-for-an-asp-net-mvc-application>. [Accessed 11 April 2018].

Microsoft Azure. 2018. SQL Database – Cloud Database as a Service | Microsoft Azure. [ONLINE] Available at: <https://azure.microsoft.com/en-us/services/sql-database/>. [Accessed 11 April 2018].

CarlRabeler. 2018. What is the Azure SQL Database service? | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-technical-overview>. [Accessed 11 April 2018].

CarlRabeler. 2018. SQL (PaaS) Database vs. SQL Server in the cloud on VMs (IaaS) | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-paas-vs-sql-server-iaas>. [Accessed 11 April 2018].

bonova. 2018. Azure SQL Database Managed Instance Overview | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-managed-instance>. [Accessed 11 April 2018].

KumudD. 2018. What is Traffic Manager | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/azure/traffic-manager/traffic-manager-overview>. [Accessed 11 April 2018].

anosov1960. 2018. Failover groups and active geo-replication - Azure SQL Database | Microsoft Docs. [ONLINE] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-geo-replication-overview>. [Accessed 11 April 2018].