



A P U
ASIA PACIFIC UNIVERSITY
OF TECHNOLOGY & INNOVATION

Project Title	:	Maersk Web Application with Azure
Module Code	:	CT071-3-5-3-DDAC
Module Title	:	Designing & Developing Cloud Applications
Lecturer Name	:	Dr. Kalai Anand Ratnam
Intake Code	:	UC3F1706SE
Student Name	:	Cheong Yung Earn
Student ID	:	TP033999
Hand in Date	:	13/4/2018

Acknowledgement

Firstly, I would like to thank my lecturer, Dr. Kalai for providing us all the necessary information and skills during lectures and tutorials lab session which turns out very useful when implementing the web application on Azure. Without him, we would not be able to learn about designing and developing application on cloud in the university. I would also like to thank Asia Pacific University of Technology and Innovation for introducing this module within our degree course and providing us with all the academic resources. Besides, I would like to thank Microsoft for all the useful resources on their website, which helped me a lot in this assignment. Last but not least, I would like to acknowledge the contribution of my fellow classmates who have helped me. Throughout the process of completing this assignment, I received many bits of help from various generous people whom without them it would not have been possible for me to complete it. I would like to use this chance to express my gratitude to them.

Contents

Acknowledgement	1
Introduction.....	4
Project background	4
Objectives	4
Scope.....	4
Requirement Specification.....	5
Summary of the major functions/ contents	5
Project Plan	6
Design	7
Design considerations	7
Implementation Architectural Diagrams.....	7
Modelling (Use Case & Sequence Diagrams)	11
Use Case Diagram.....	11
ERD.....	15
Sequence Diagram	16
Implementation	23
Developing application using Visual Studio.....	23
Application scaling	32
Traffic manager.....	35
Reliability & Performance	40
Test Plan & Testing Discussion (Functional & Performance).....	40
Unit test (Agent)	40
Unit test (Admin).....	42
Performance Test	45
Analysis.....	48
Managed Database	49

Conclusion	53
References.....	54

Introduction

Project background

Maersk Line 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.

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. Maersk Line, is looking at designing and developing a Container Management System (CMS) to cater to manage the containers, a solution that reduces overall supply chain costs and an efficient way to manage logistics.

Objectives

The project involves designing and developing a single tenant web solution that meets the following objectives:

- To be able to scale the solution to meet the needs of demands during peak seasons.
- Improves profitability, reduce costs, increases productivity, eradicates errors and optimizes resources to future-proof your cargo handling business for high performance.
- Assurance & reliability through Failover Management.
- Accurately delivering an exceptional level of automation and removing human error.
- Manage entire booking process from schedule search to booking confirmation.

Scope

The following scope is defined to fulfil Maersk business requirements:

- Design & Develop a single tenant web application hosted on Microsoft Azure as an App Service (Web App) that consume Relational Database
- The web application developed consist of 5 - 10 interlinked pages that provide quality content and design.

- Analyze web application performance with monitoring tools.
- To be able to scale the solution to meet the needs of demands during peak seasons.
- Source code to place in source control management services (GitHub: https://github.com/tp033/maerskSEA_rep).
- The demo of the application is available at: <https://web.microsoftstream.com/video/1c94bd2d-64a1-445b-be86-6fba0eda088f>

Requirement Specification

The single tenant web solution supports these requirements:

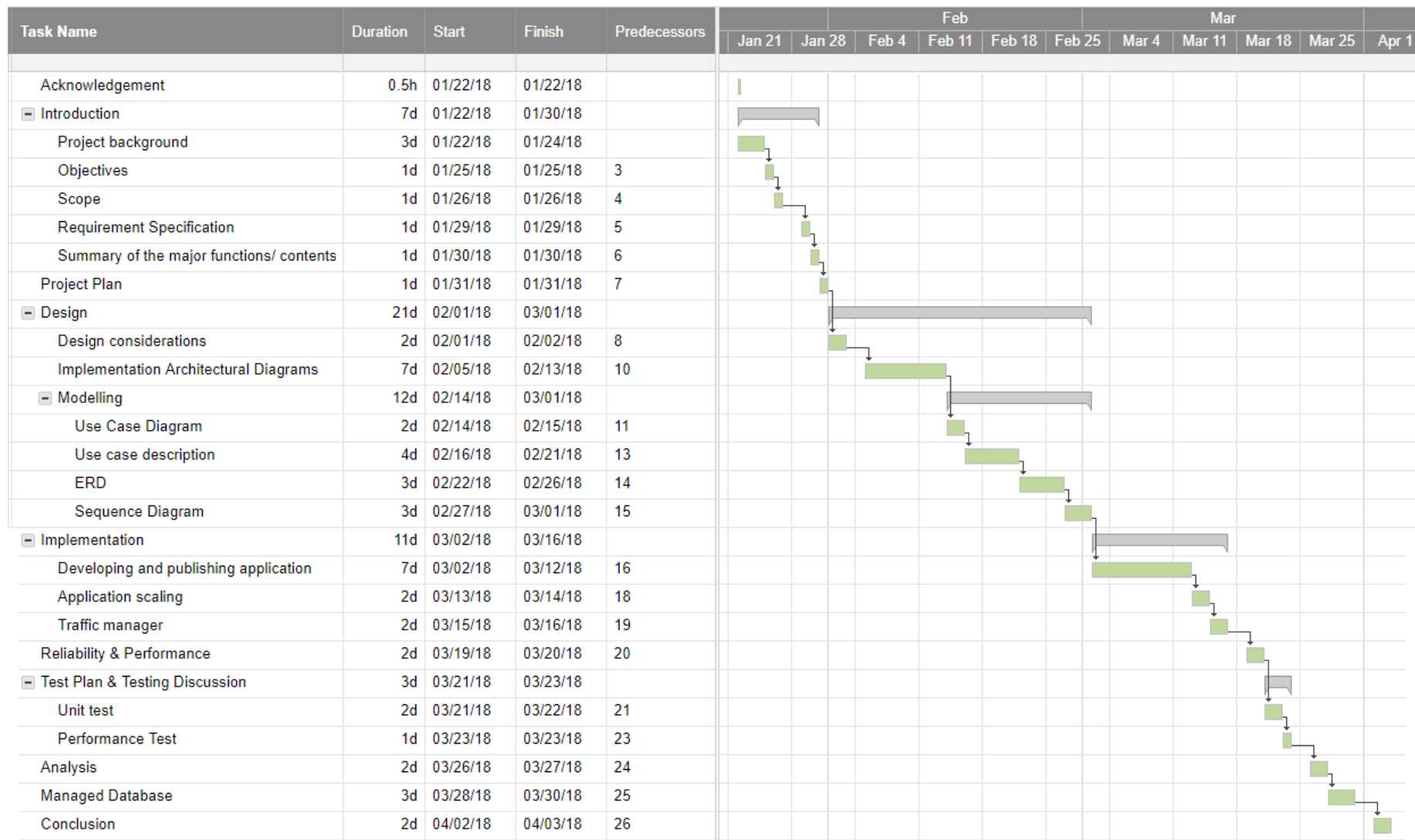
- Manage the entire booking process.
- Creation of profile.
- Retrieve relevant information for users.

Summary of the major functions/ contents

The Container Management System will allow Maersk to manage entire booking process from schedule search to booking confirmation. The web application can be accessed from any web browsers such as Internet Explorer, Opera and Google Chrome. There are two types of users in this web application, which are the Maersk admin and agents. The features that should be provided by the web application are listed below.

- Allow admin to register new account for agents.
- Allow admin to view all agent information.
- Allow admin to view all bookings.
- Allow admin to create a new schedule for vessels.
- Allow agent and admin to view shipment details such as source, destination, estimated time of arrival and departure time.
- Allow agent to create a new booking and item registration
- Allow agent to view the booking information.
- Allow agent to register customers.

Project Plan



Design

Design considerations

Maersk application would target SEA region as an initial region to deploy its resources on cloud. However, considering the needs of expansion to West US region and increased traffic would overload the SEA region, the application would be deployed on West US region as well. Due to tight budget, the development and deployment would proof the concept of cloud computing on a production work basis, the application would require further expansion to support the business operation of Maersk.

Implementation Architectural Diagrams

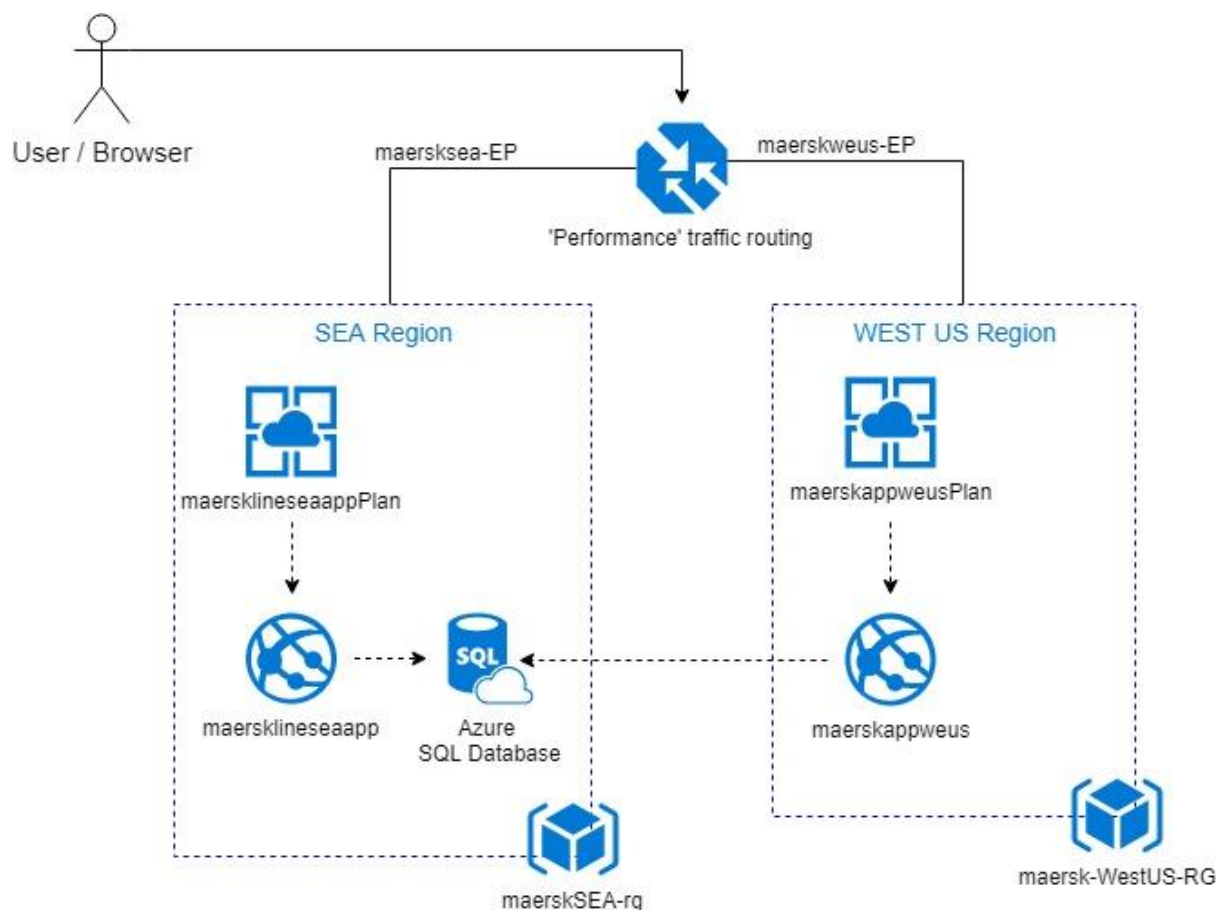


Figure 1: Current architecture

Currently, the architecture implemented consists of two regions, South-East Asia and West US regions. An Azure SQL Database that store the data, was provisioned and placed within SEA region, since it is assumed that the solution would be initiated from SEA region first, and then would be expanded to the other regions.

Due to the design consideration to run the production workload, the application plan for both region uses a Standard S1 (1 Core, 1.75 GB RAM, 50GB storage) plan that provides built-in network load balancing support which automatically distributes traffic across instances. The standard plan includes auto scale instances to match the application traffic needs. Application service plan located in each region would consists of a default three instances to be scaled up and out based on the performance need. Since this deployment is initial, the Azure SQL database would be shared across these two regions. The traffic manager uses a performance routing method so that users would be redirected to the closest endpoint, which is the endpoint with least network latency.

The diagram above is what has been implemented based on the budget restrictions. The ideal architecture would be the one shown below.

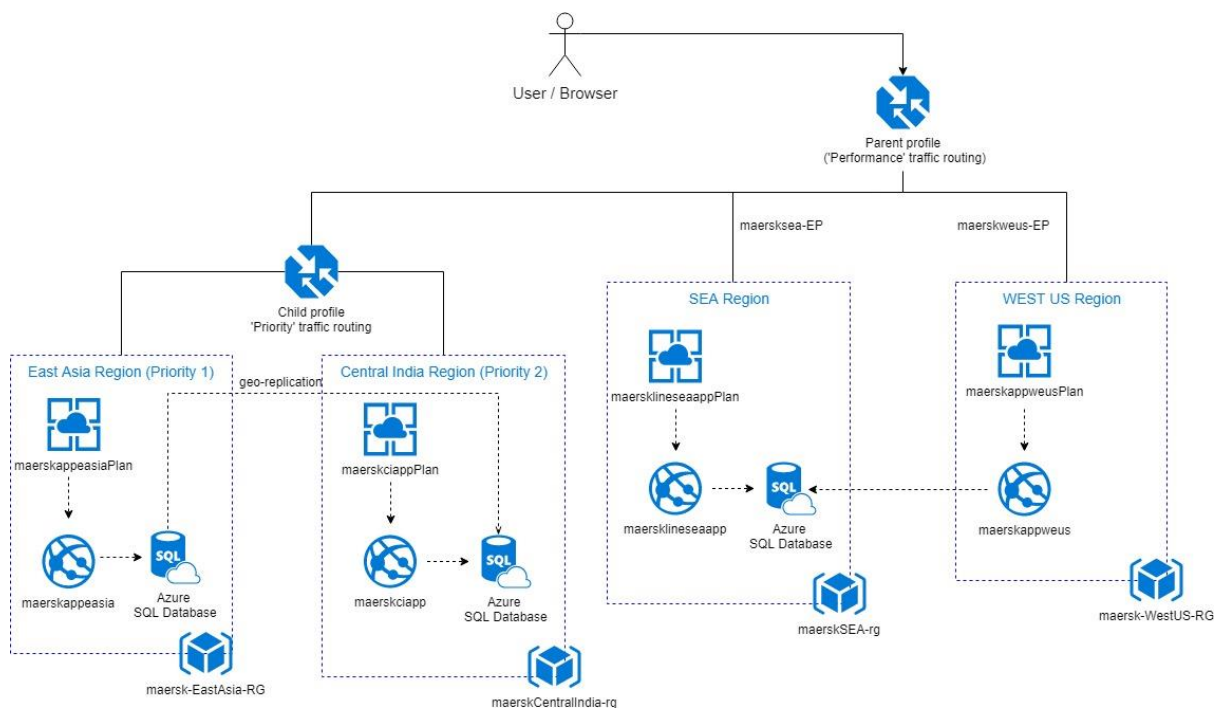


Figure 2: Proposed architecture

This deployment decision is considering additional regions for the application as the company would like to expand and have more regions to deploy their application. Other than Central India region which uses a S1 plan, all the region would use a S2 plan to cater with the increased workload.

In this case, a nested traffic manager profile is used, the parent profile with 'Performance' traffic-routing method is designed to avoid over-loading the next nearest endpoint and causing a cascading series of failures. When an endpoint fails, all traffic that would have been directed

to that endpoint is evenly distributed to the other endpoints across all regions. As noticed, a child profile with the 'Priority' traffic-routing method is designed for both East Asia and Central India, the East Asia traffic would failover to Central India, and only direct traffic to other regions when both endpoints are unavailable. Since the East Asia endpoint has higher priority than the Central India endpoint, all traffic would be sent to the East Asia endpoint when both endpoints are online. If East Asia fails, its traffic is directed to Central India. With the nested profile, traffic is directed to SEA or West US only when both Central India and East Asia fail. Of course, when all the endpoints are available, users would be directed to the endpoint with the least network latency.

Furthermore, the SQL server in East Asia region is geo-replicated in the Central India region for disaster recovery. After an outage in the primary region (priority 1), the SQL Database service detects that the primary database is not accessible and triggers failover to the secondary region (priority 2). The secondary database automatically synchronizes with the primary. The advantage of this architecture is that the same web application could be deployed to different regions without any region-specific configuration and doesn't require additional logic to manage failover. Also, the application performance is not impacted by failover as the web application and the database are always co-located. However, there is a trade-off for the priority traffic routing method, the application resources in priority 2 region are underutilized most of the time, therefore, the application plan for that region would be S1, which is slightly low performance than other regions.

In fact, a similar geo-replication could be deployed for West-US region, however, in order to reduce cost for additional database in West US region, the developer remains the architecture of a shared database across SEA and West US region, which might slightly reduce the performance as the West US region need to access database in SEA region; the developer has decided to tolerate acceptable latency to save cost, instead of incurring higher cost for performance.

Estimated cost for proposed deployment:

Service type	Region	Description	Estimated Cost
App Service	Southeast Asia	3 instance(s) x 730 Hours, Size: S2, Standard tier	RM1,839.60
App Service	East Asia	3 instance(s) x 730 Hours, Size: S2, Standard tier	RM1,839.60
App Service	Central India	3 instance(s) x 730 Hours, Size: S1, Standard tier	RM1,011.78
App Service	West US	3 instance(s) x 730 Hours, Size: S2, Standard tier	RM1,839.60
Azure SQL Database	Southeast Asia	Single Database, DTU Purchase Model, Standard Tier, S2: 50 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	RM309.16
Azure SQL Database	East Asia	Single Database, DTU Purchase Model, Standard Tier, S2: 50 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	RM309.16
Azure SQL Database	Central India	Single Database, DTU Purchase Model, Standard Tier, S2: 50 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	RM353.68
Traffic Manager	Global	3 million DNS queries/mo, 4 Azure endpoint(s)	RM12.85
		Monthly Total	RM7,515.42
		Annual Total	RM90,185.01

Modelling (Use Case & Sequence Diagrams)

Use Case Diagram

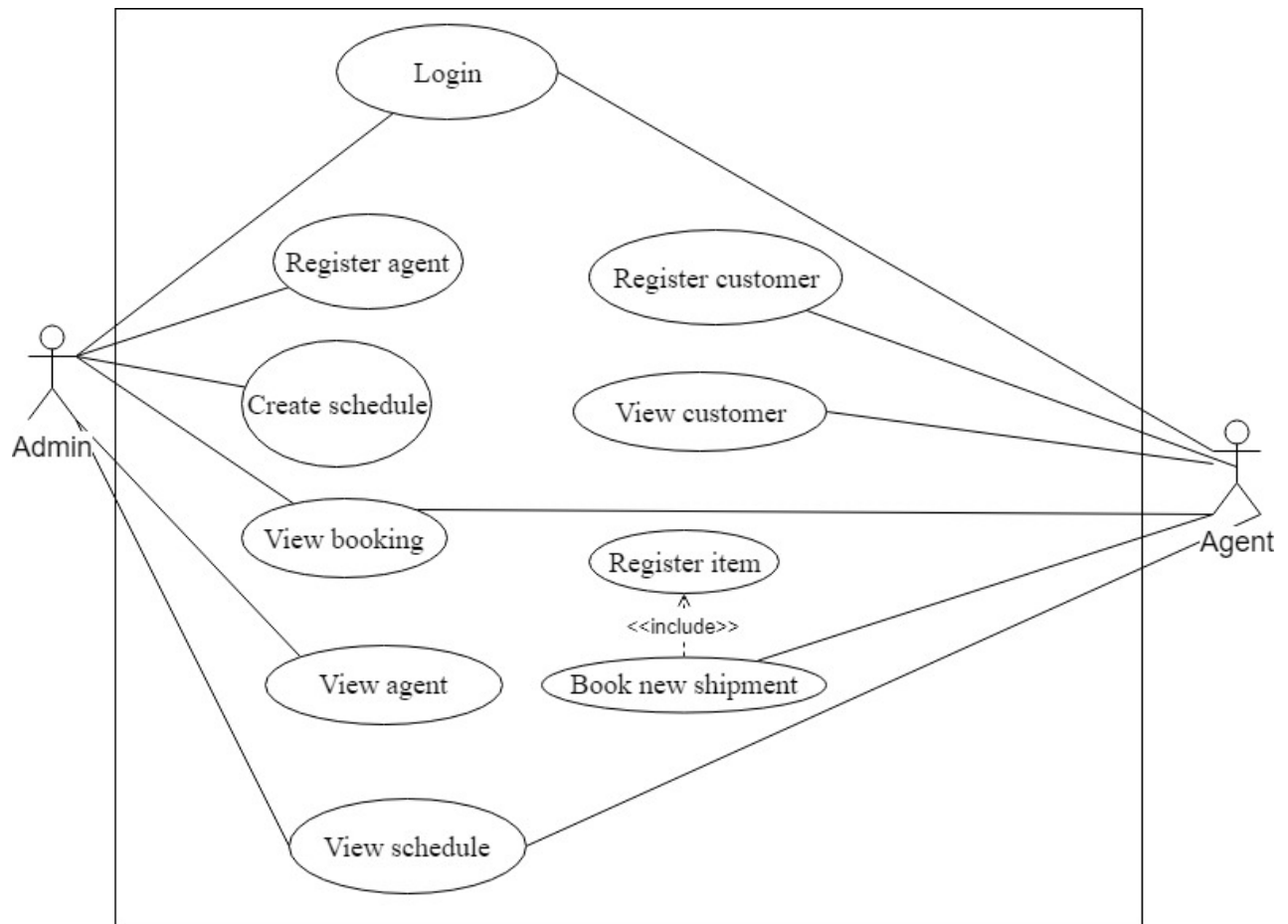


Figure 3: Use case diagram

Use case description

Use case:	Login
Actors:	Agent, Admin
Brief description:	Login to the account with specific credential
Dependency	N/A
Pre-condition:	The credential should be registered first before users could log in
Basic flow:	<ol style="list-style-type: none"> 1. Click Login tab on the top right corner 2. Enter username 3. Enter password 4. Click Login

Alternate Flow (Exception):	If credential entered is invalid, error message would be shown
Post-condition:	Redirect to the view booking page.

Use case:	Register agent
Actors:	Admin
Brief description:	Create new agent by entering agent details
Dependency	N/A
Pre-condition:	The admin must log in first
Basic flow:	<ol style="list-style-type: none"> 1. Click on the register agent tab 2. Enter agent details. 3. Click create
Alternate Flow (Exception):	<p>3a. If any detail is missed out, detailed error message would be shown.</p> <p>3b. If the username is associated with an existing username, a message indicating user exists would be shown</p>
Post-condition:	Show agent registration successful message

Use case:	Create schedule
Actors:	Admin
Brief description:	The agent creates schedule for vessel
Dependency	N/A
Pre-condition:	The admin must log in first and should have planned and confirm the schedule details before entering it to the system
Basic flow:	<ol style="list-style-type: none"> 1. Click create schedule tab 2. Enter schedule details 3. Click create
Alternate Flow (Exception):	<p>3a. If any detail is missed out, detailed error message would be shown.</p> <p>3b. If the schedule clashes with the existing schedule, a message indicating schedule clashes would be shown</p>
Post-condition:	Display booking successful message

Use case:	View booking
Actors:	Agent, Admin
Brief description:	The admin could view all the bookings; on the other hand, the agent could view only his/her specific bookings
Dependency	N/A
Pre-condition:	There must be a booking, so that it could be viewed
Basic flow:	1. Click view bookings tab 2. Click view schedule to view the schedule of the bookings
Alternate Flow (Exception):	N/A
Post-condition:	Display all the specific bookings and the details

Use case:	View agent
Actors:	Admin
Brief description:	Allow admin to view list of agents and their details
Dependency	N/A
Pre-condition:	There must be an agent, so that it could be viewed
Basic flow:	1. Click view agents tab
Alternate Flow (Exception):	N/A
Post-condition:	Display all the agents and their details

Use case:	View schedule
Actors:	Agent, Admin
Brief description:	Allow agent and admin to view the schedule details
Dependency	N/A
Pre-condition:	There must be a schedule added by agent, so that it could be viewed
Basic flow:	1. Click view schedule tab
Alternate Flow (Exception):	N/A
Post-condition:	Display the schedule details

Use case:	View customer
Actors:	Agent
Brief description:	Allow agent to view the customer details
Dependency	N/A
Pre-condition:	The agent must log in first
Basic flow:	1. Click on the View customer tab
Alternate Flow (Exception):	N/A
Post-condition:	Display the customer details

Use case:	Book new shipment
Actors:	Agent
Brief description:	The agent books a vessel to ship his/her item
Dependency	<<include>> Register item
Pre-condition:	The agent should know which schedule is appropriate for shipping his/her item. The owner of the item must be registered as a customer.
Basic flow:	1. Select a time slot from the list of schedule details 2. Click register item 3. Enter item details 4. Click register item and booking
Alternate Flow (Exception):	4a. If any detail is missed out, detailed error message would be shown.
Post-condition:	Display a booking successful message

Use case:	Register customer
Actors:	Agent
Brief description:	Create new customer by entering customer details
Dependency	N/A
Pre-condition:	The agent must log in first
Basic flow:	1. Click on the register customer tab 2. Enter customer details. 3. Click create
Alternate Flow (Exception):	3a. If any detail is missed out, detailed error message would be shown. 3b. If the NRIC/ passport number is associated with the username, a message indicating user exists would be shown
Post-condition:	Show agent registration successful message

ERD

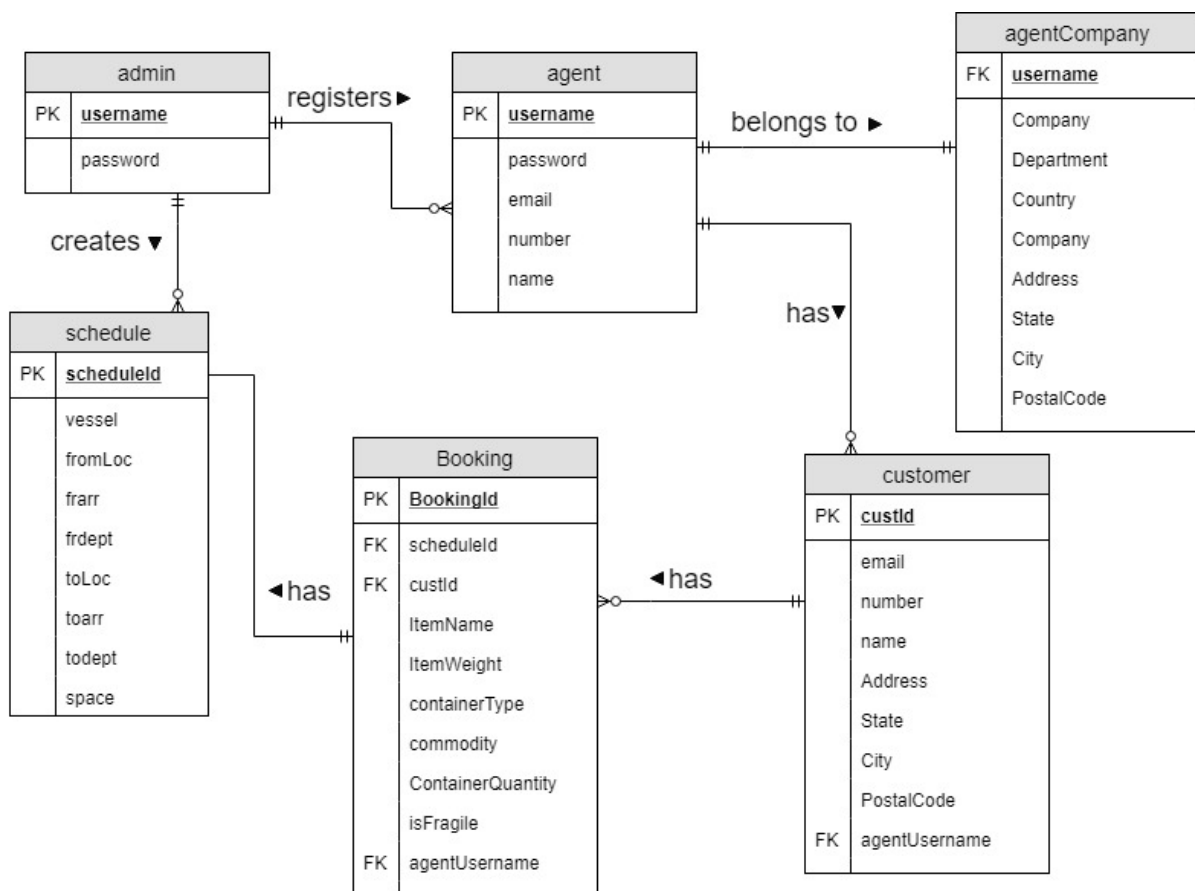


Figure 4: Database ERD

Sequence Diagram

Login

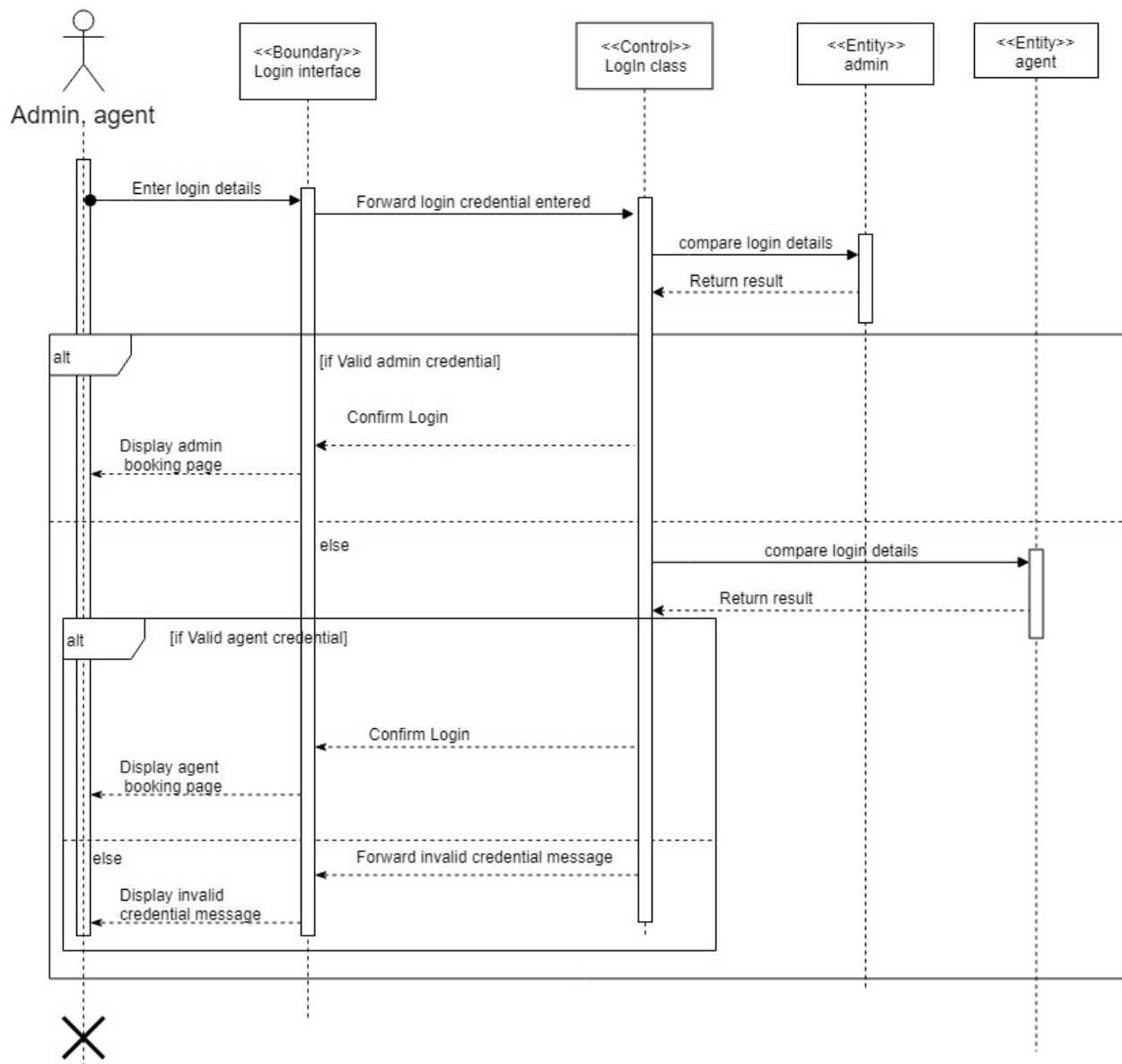


Figure 5: Login sequence diagram

Register agent

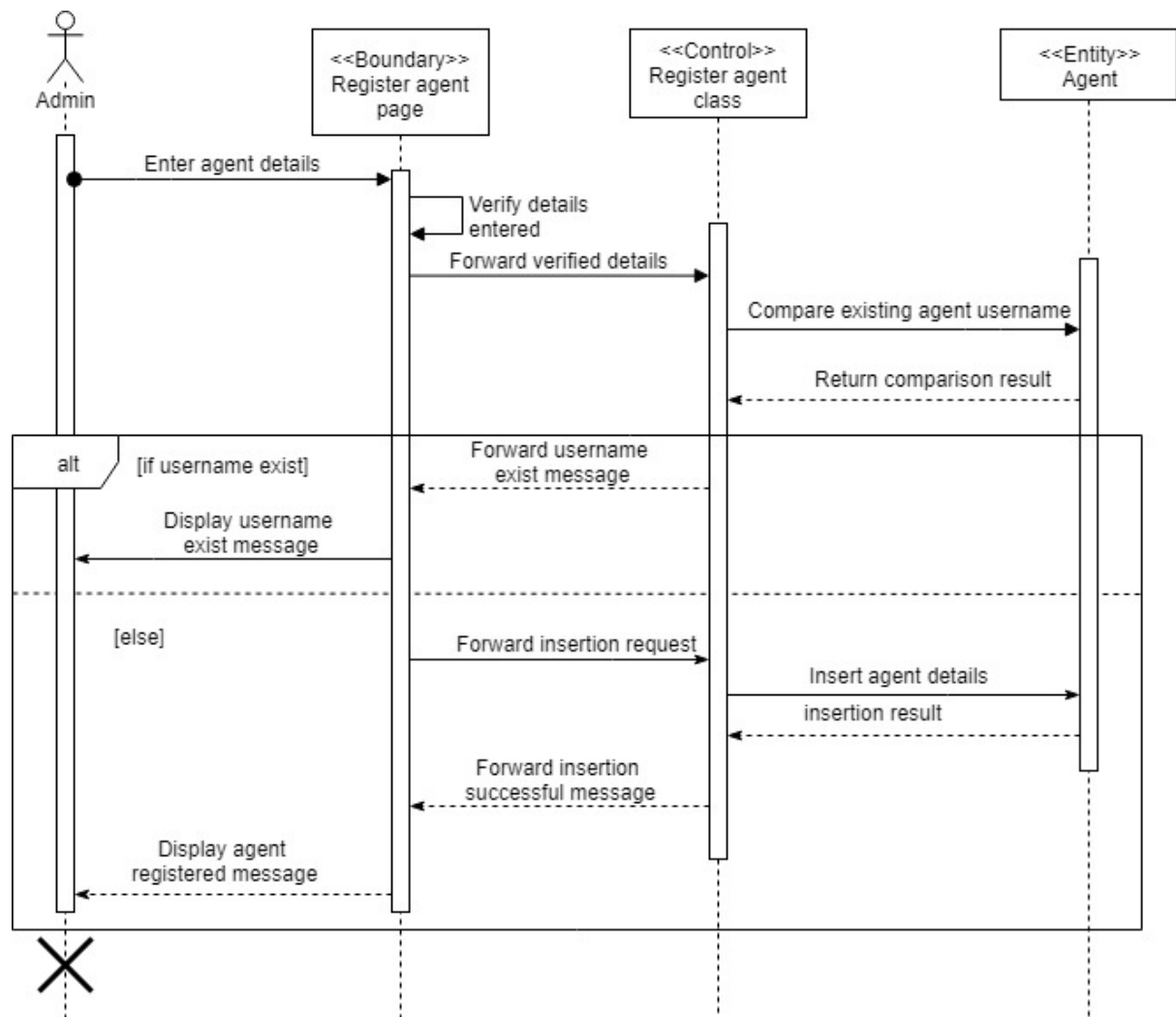


Figure 6: Register agent sequence diagram

Create Schedule

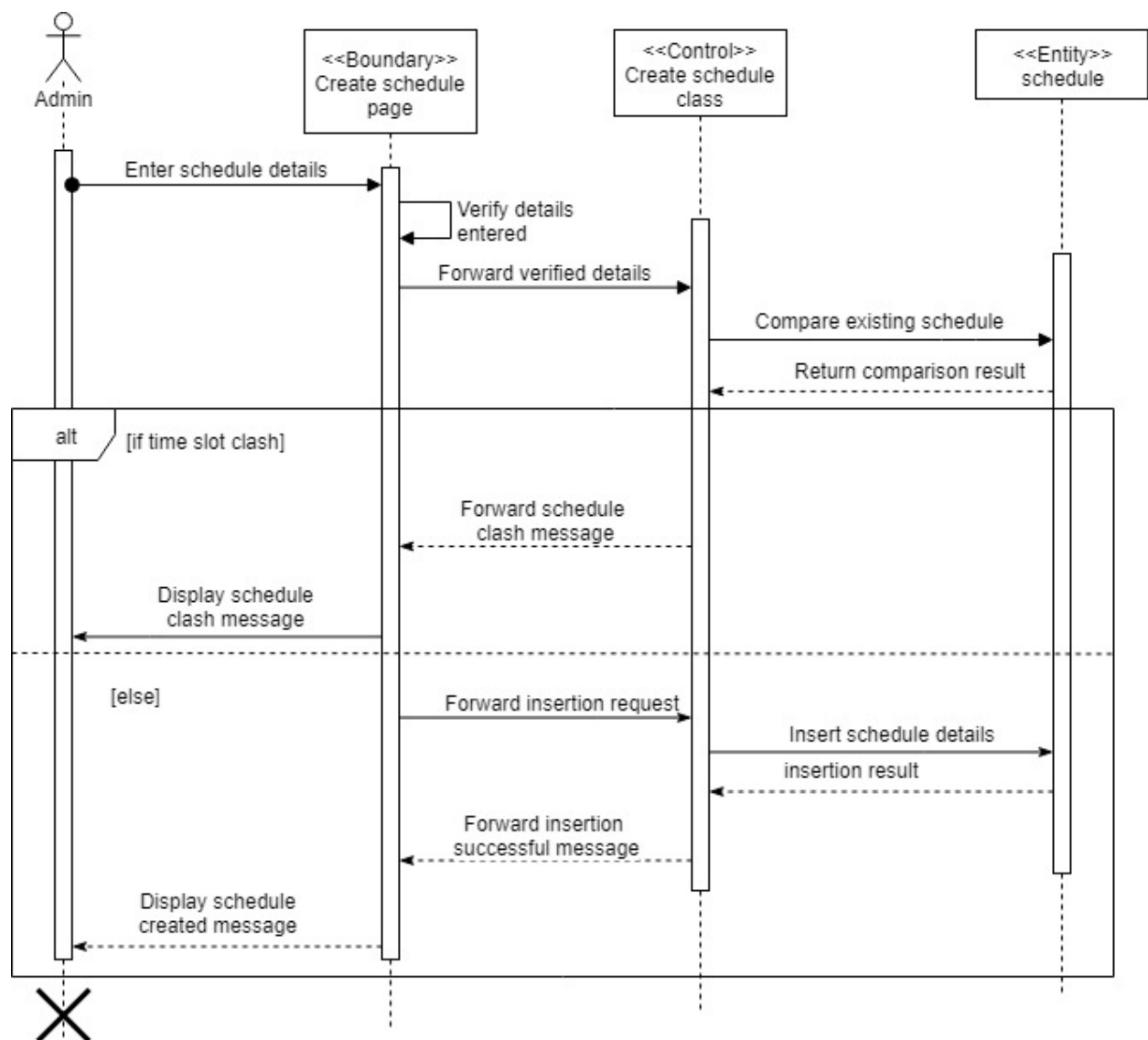


Figure 7: Create schedule sequence diagram

View agent

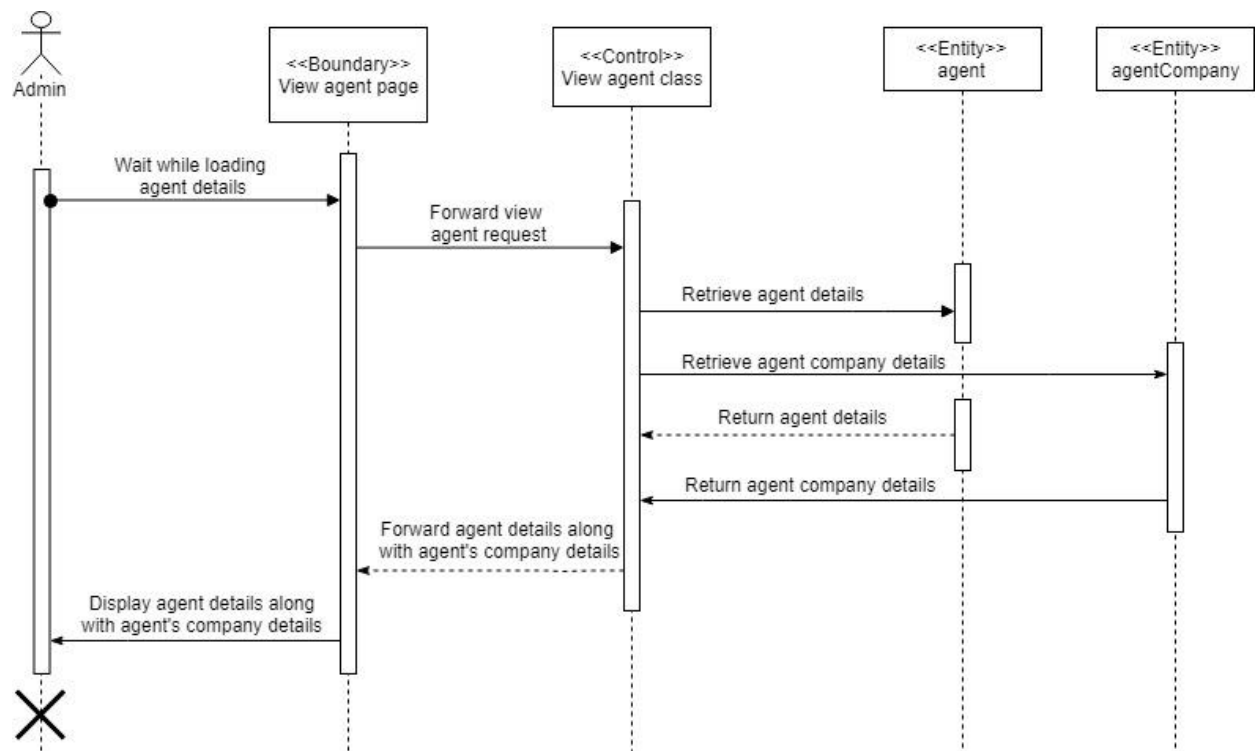


Figure 8: View agent sequence diagram

View booking

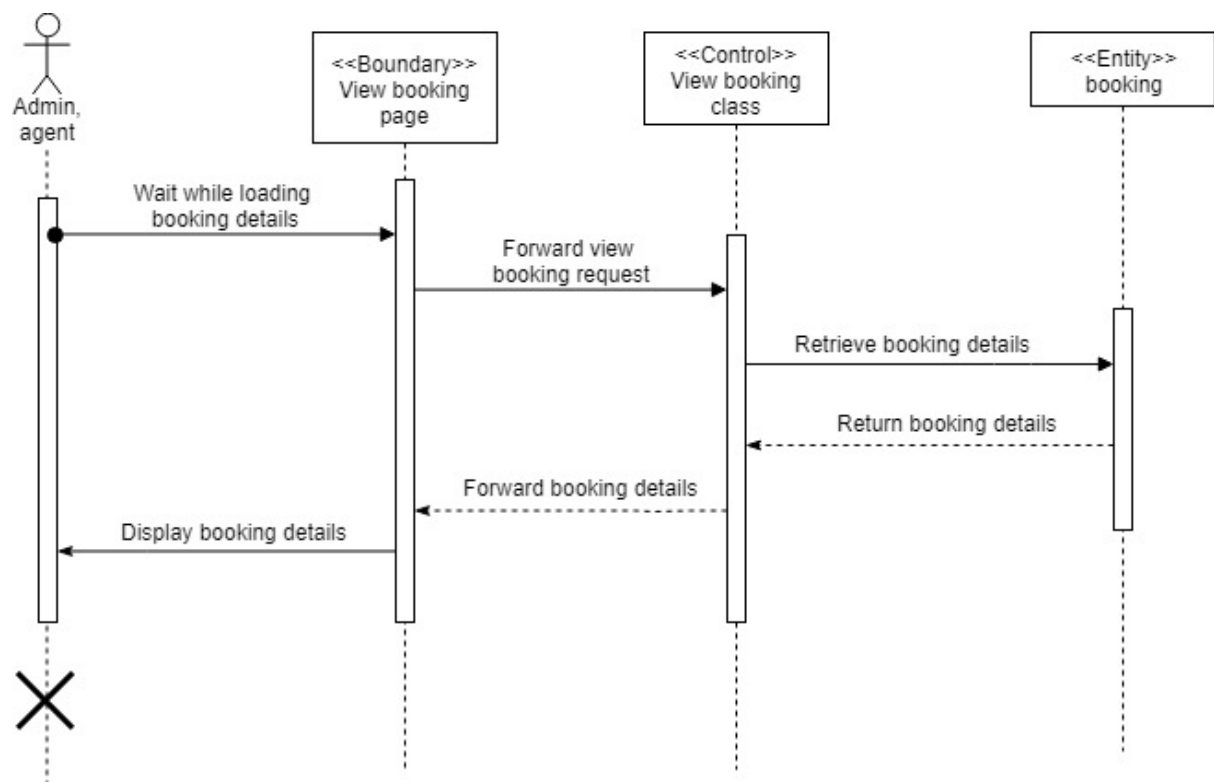


Figure 9: View booking sequence diagram

View schedule

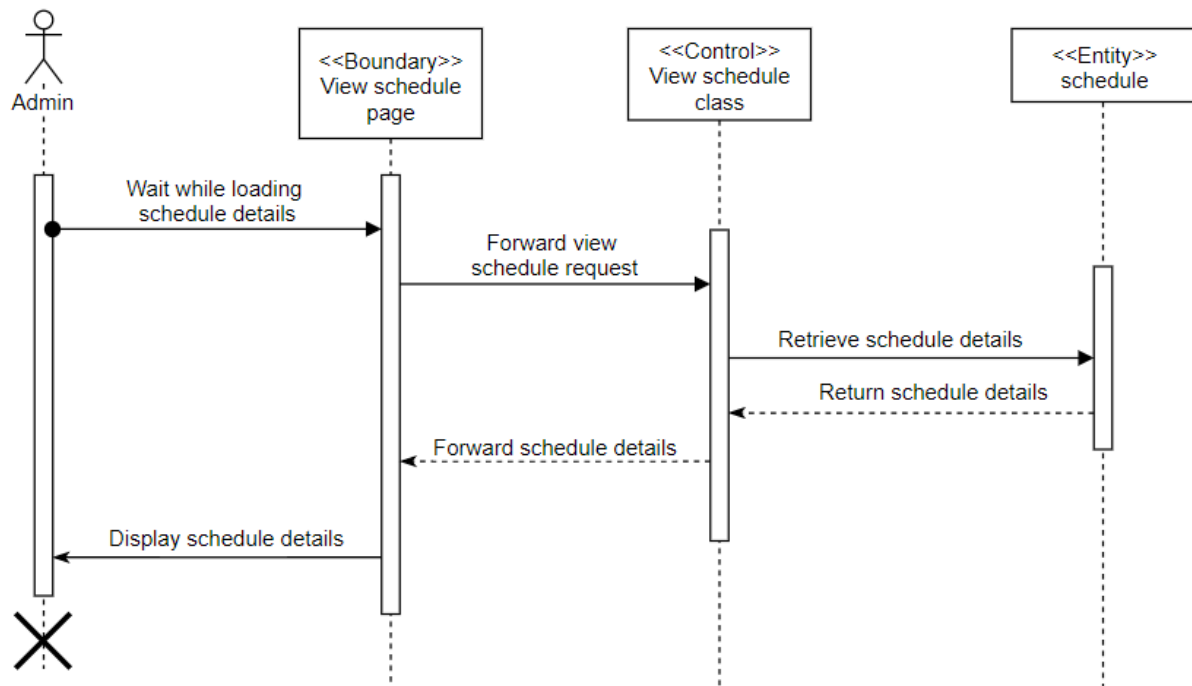


Figure 10: View schedule sequence diagram

View customer

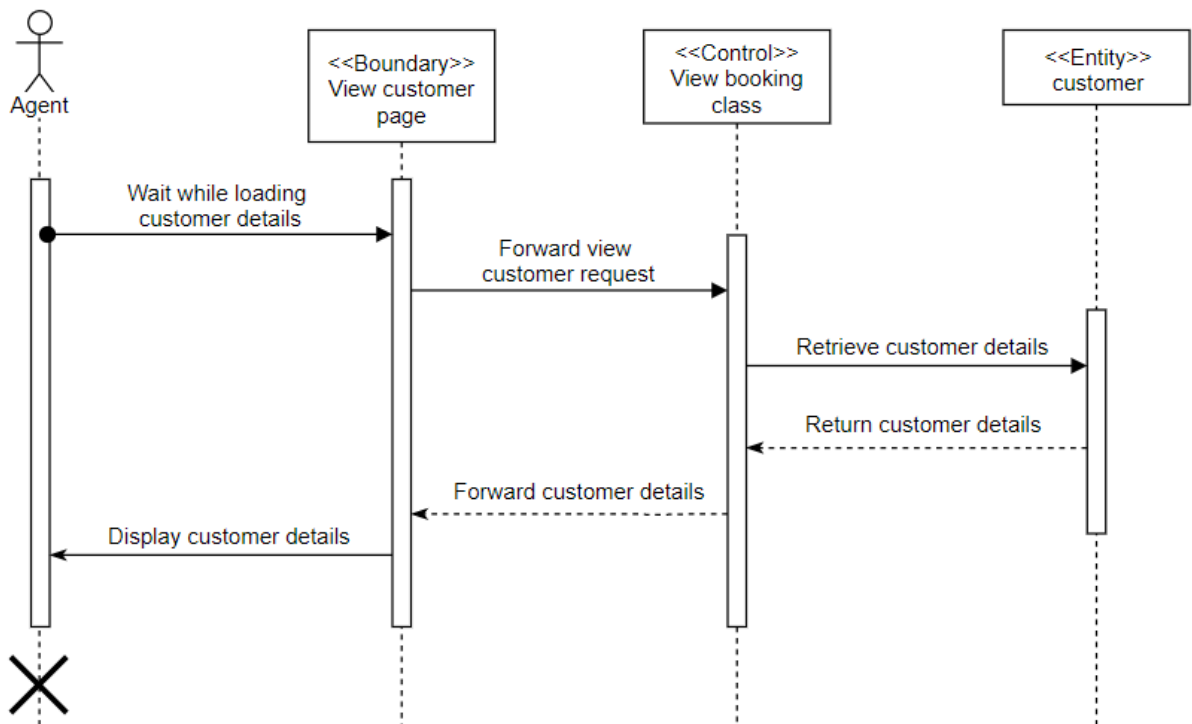


Figure 11: View customer sequence diagram

Register customer

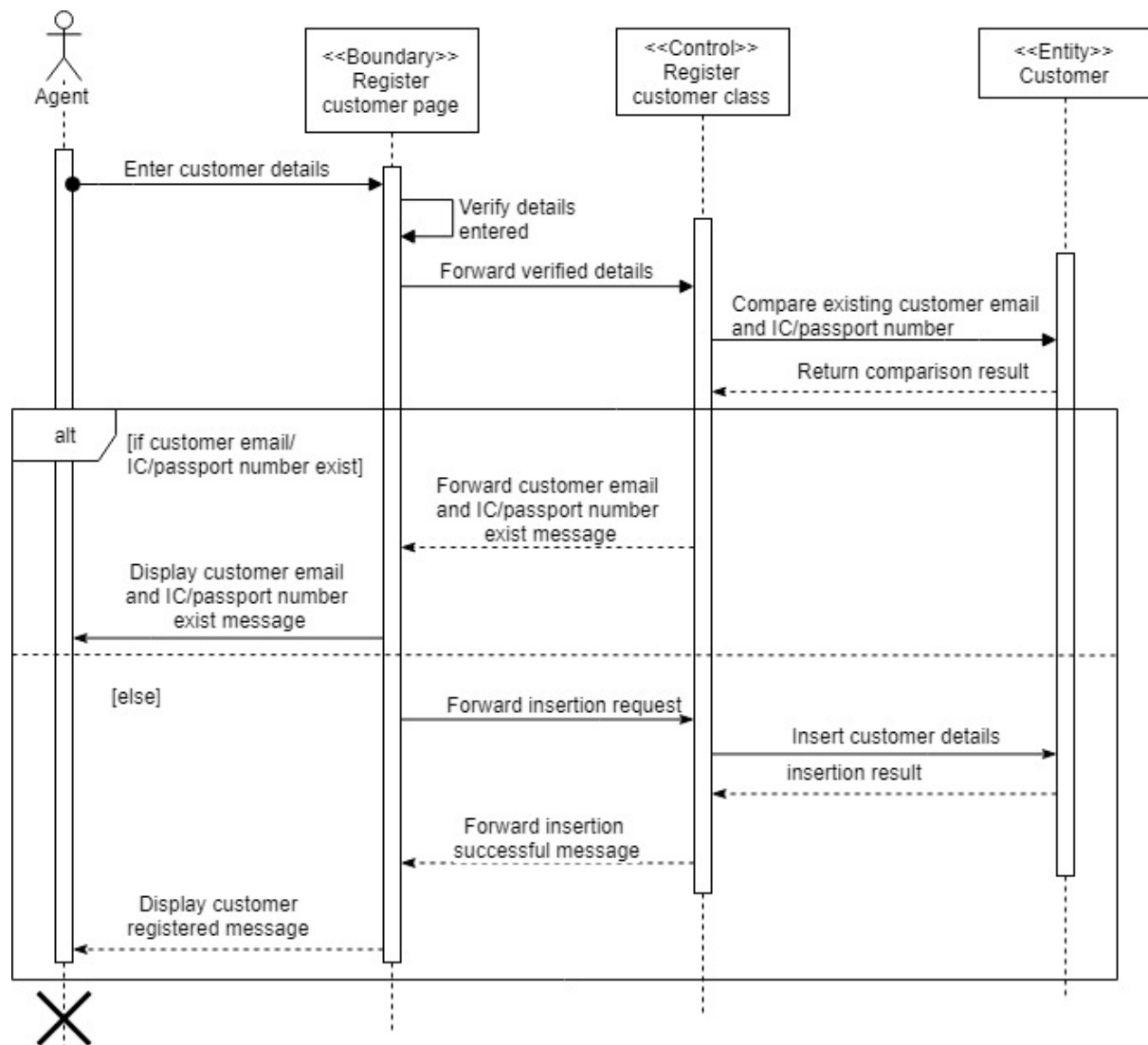


Figure 12: Register customer sequence diagram

Book new shipment

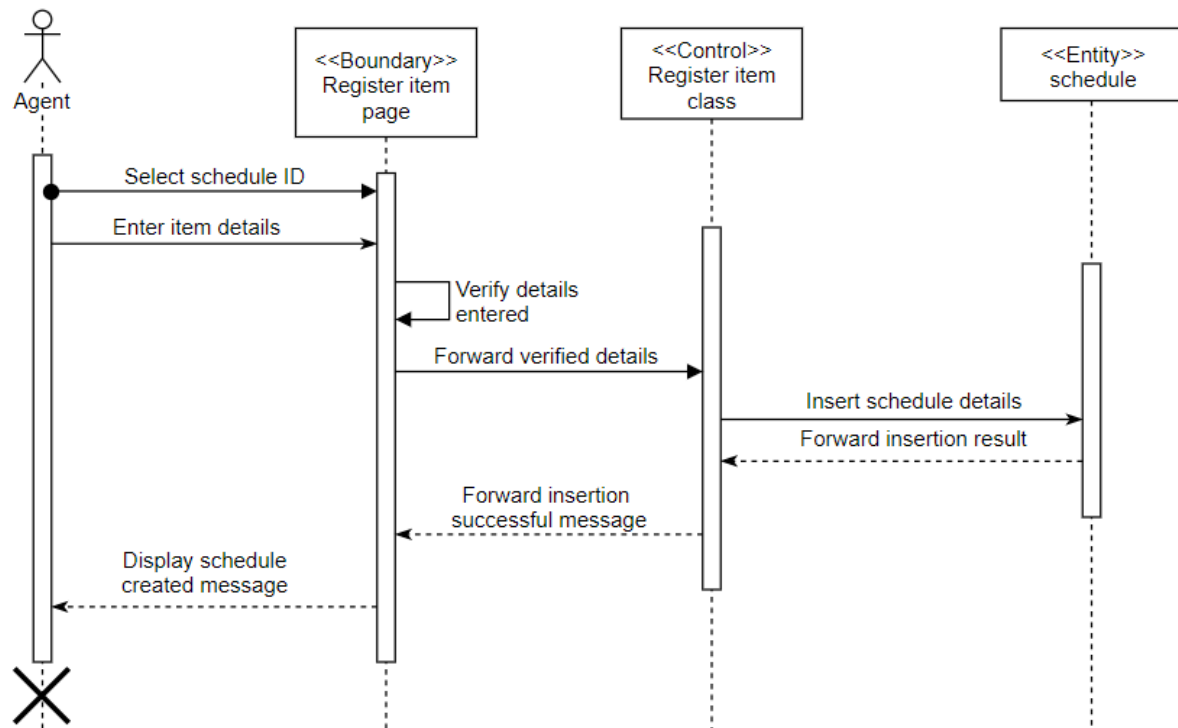


Figure 13: Book new shipment sequence diagram

Implementation

Developing application using Visual Studio

The application was developed using ASP.NET Web Forms and C# with a SQL database to store and retrieve the application data based on user needs. The file structure is demonstrated as shown below.

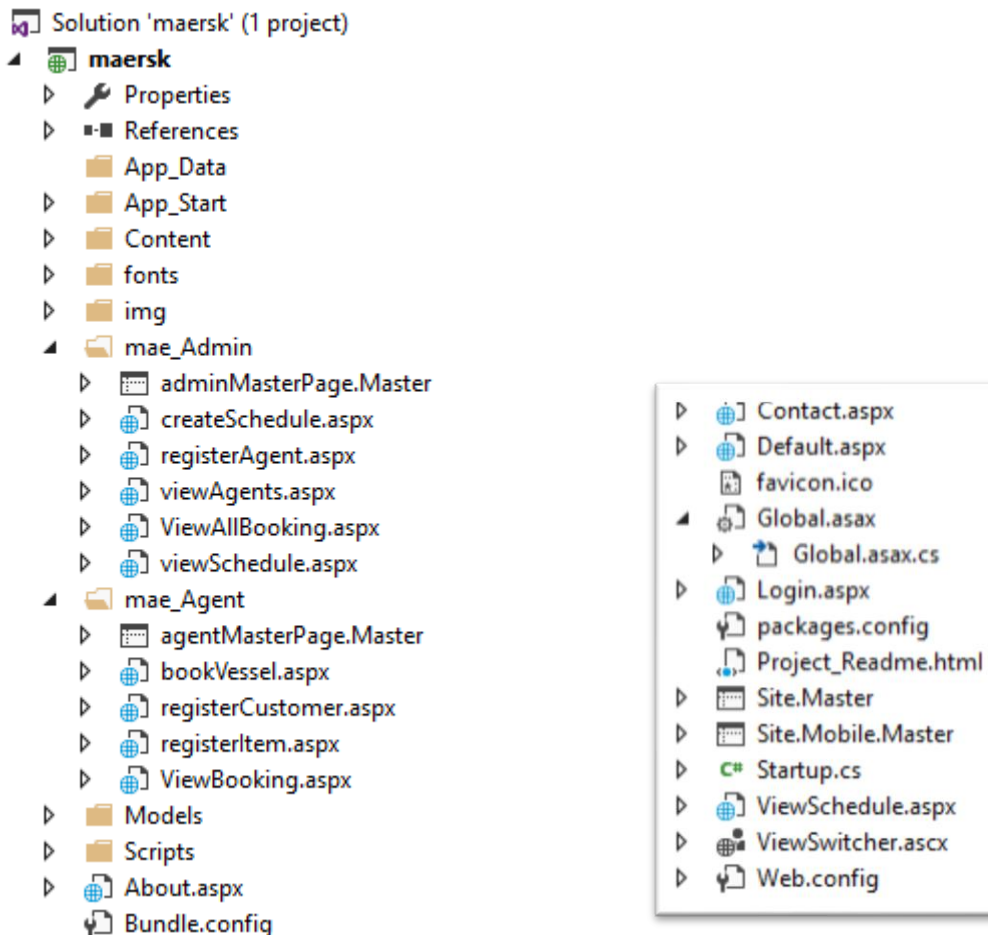


Figure 14: File Structure

There is a folder called mae_Admin and mae_Agent which store the admin and agent web forms respectively. On the other hand, the .aspx files out of the folders are accessible by users that are neither admin or agent, in other words, which are users that are not logged in. For each .aspx file, there is a C# code behind file which was auto generated once a web form is created. The first page that would be shown is the Default.aspx, which is also the home page in this case. The Web.config file stores the configuration information of the web application, such as the SQL connection string.

Creating a SQL server

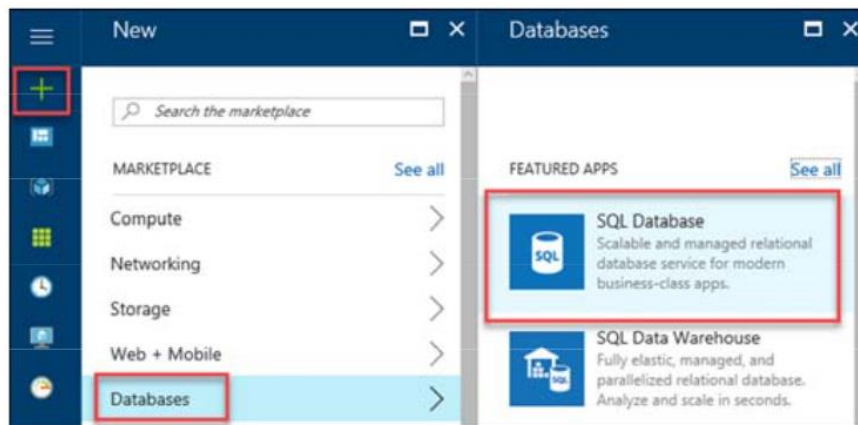


Figure 15: Creating Azure SQL database

Before the database was created, a server must be created by specifying the server name, login credential of the server and location of the server as shown below. The server name given was ‘maersk-server’.

* Server name

.database.windows.net

* Server admin login

* Password

* Confirm password

* Location

☒ Allow azure services to access server ⓘ

Figure 16: Create new server

Resource group (change) maerskSEA-rg	Server name maersk-server.database.windows.net
Status Online	Elastic database pool No elastic pool
Location Southeast Asia	Connection strings Show database connection strings
Subscription (change) Azure for Students	Pricing tier Basic
Subscription ID 5b050bae-dcba-4286-a254-e9ed4266a434	Oldest restore point No restore point available

Figure 17: Configuration of database

The database was created based on the configuration above. When creating the database, it is important to choose a pricing plan based on application usage, to be specific on performance, when creating this application, a basic plan with 5 DTUs is chosen as the developer would like to demonstrate the application only, however, in a real scenario like Maersk, the developer would recommend a standard or premium pricing plan. This is because, both plan has a higher storage as well as DTUs.

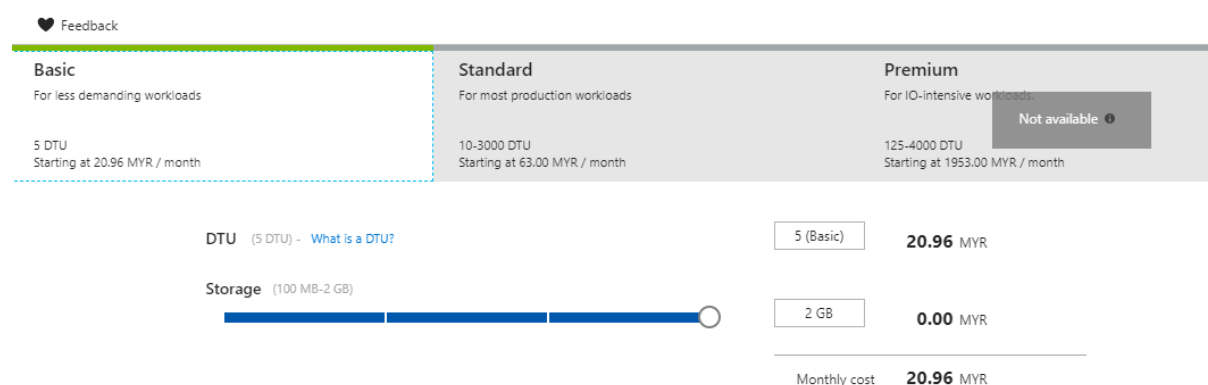
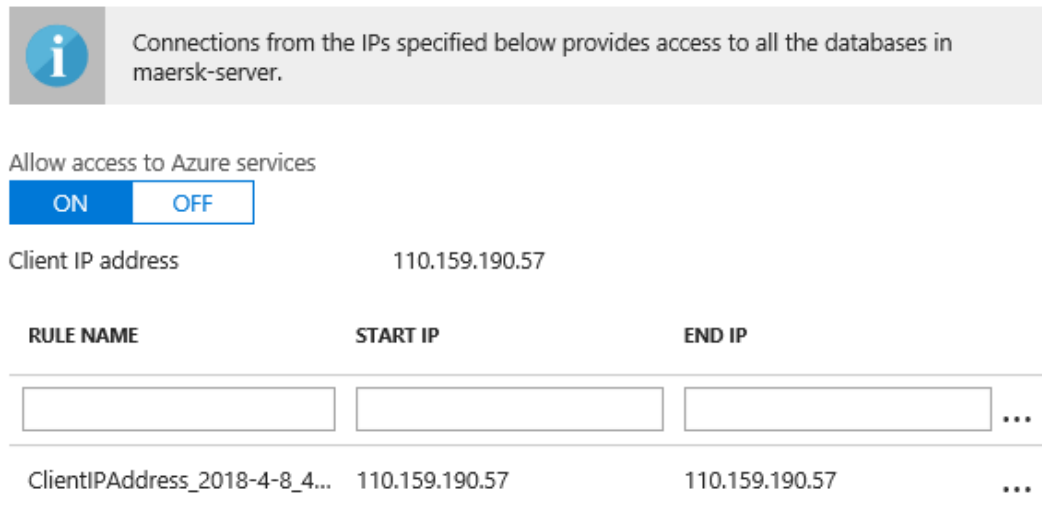


Figure 18: Database pricing

Doubling the DTUs equates to doubling the set of resource available to that database, in other word, increase the performance of the database. If maximum DTUs is achieved, query latency increases, as these queries may be queue and would only be executed when resources are available, hence, the workload would be slowed down. If the queries start timing out, an error might occur. In terms of reaching maximum limit of database storage, insertion and update actions that would increase the data size, would fail and displays an error message to the client, however, the delete and select statement is still executable. Therefore, choosing a pricing plan is important to ensure that the application is resilient and low latency (Microsoft Azure, 2017).



Connections from the IPs specified below provides access to all the databases in maersk-server.

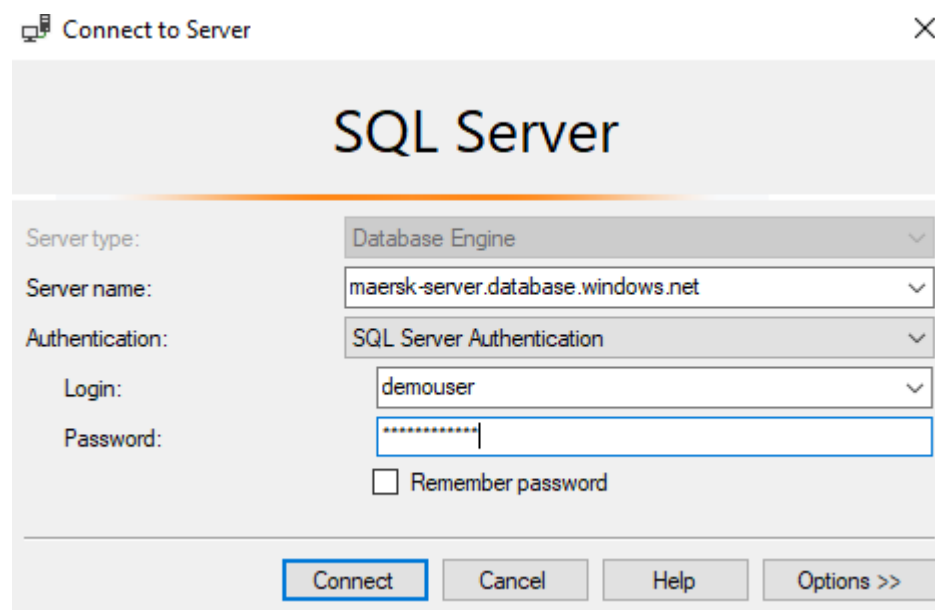
Allow access to Azure services
ON OFF

Client IP address 110.159.190.57

RULE NAME	START IP	END IP
ClientIPAddress_2018-4-8_4...	110.159.190.57	110.159.190.57

Figure 19: Add client IP

Once the database deployment has completed, the server firewall is set by creating a firewall rule to allow access from the developer's current client IP address. A new firewall rule need to be added for the new IP address if the IP address changes in the future.



Connect to Server

SQL Server

Server type: Database Engine

Server name: maersk-server.database.windows.net

Authentication: SQL Server Authentication

Login: demouser

Password: *****

☐ Remember password

Connect Cancel Help Options >>

Figure 20: SQL server authentication

After that, the server could be connected with the credential from SQL server management studio with SQL server authentication.

The existing SQL server database is migrated to Azure SQL server database by following these steps:

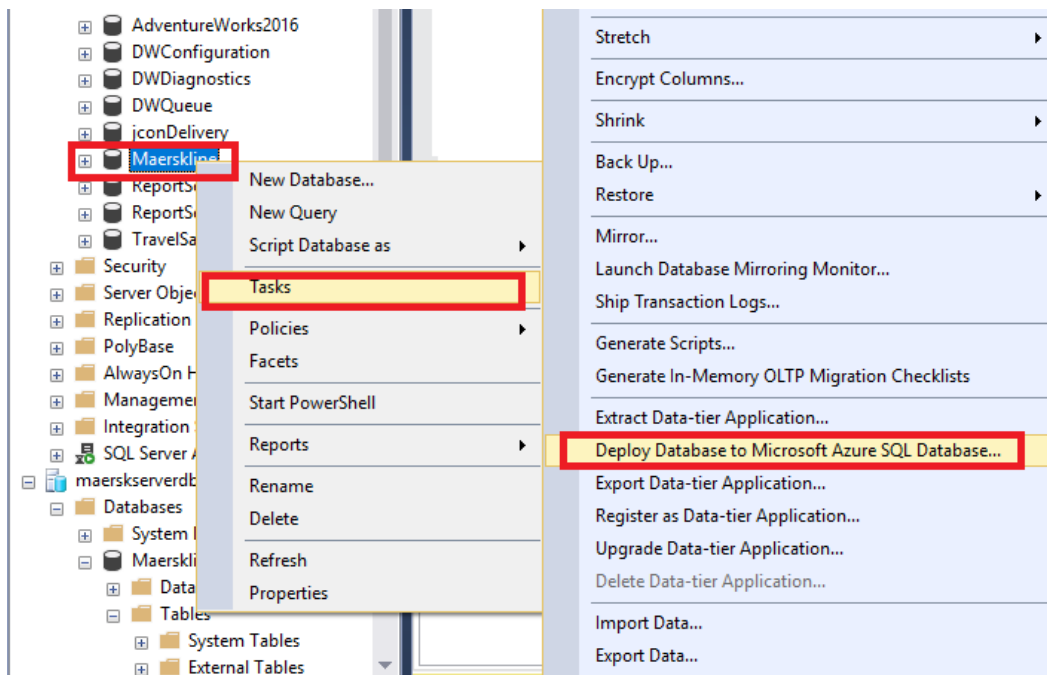


Figure 21: Deploy local database to Azure SQL database

Click Next on the Introduction window.

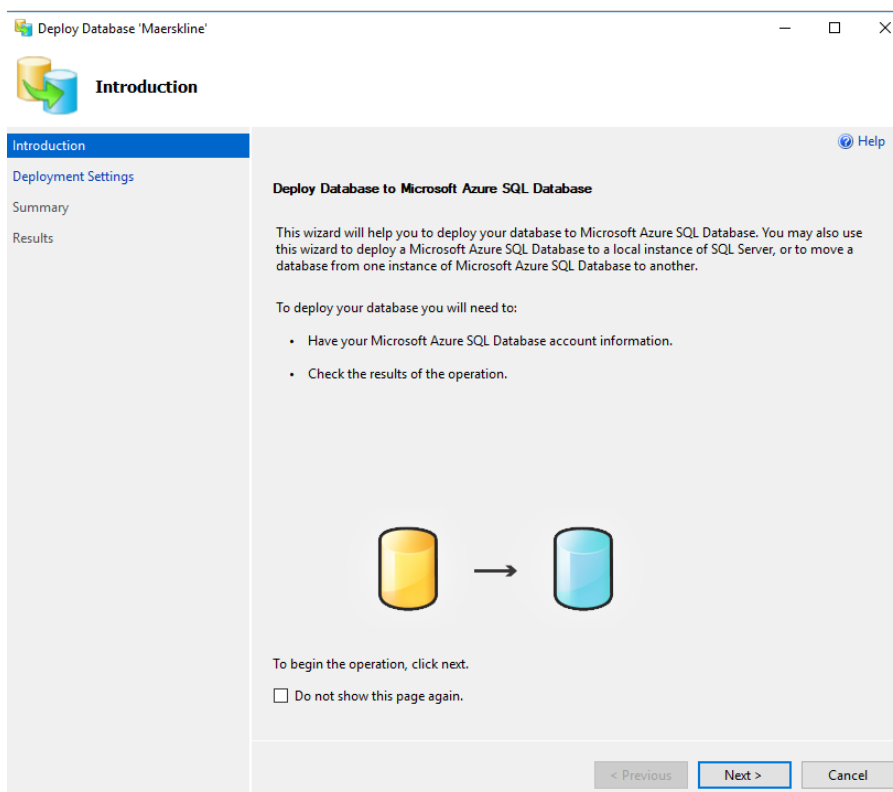


Figure 22: Deploy database wizard

On the Deployment Settings window and click the Connect button and connect to the Azure SQL Server. The deployment is configured as shown below, and click next.

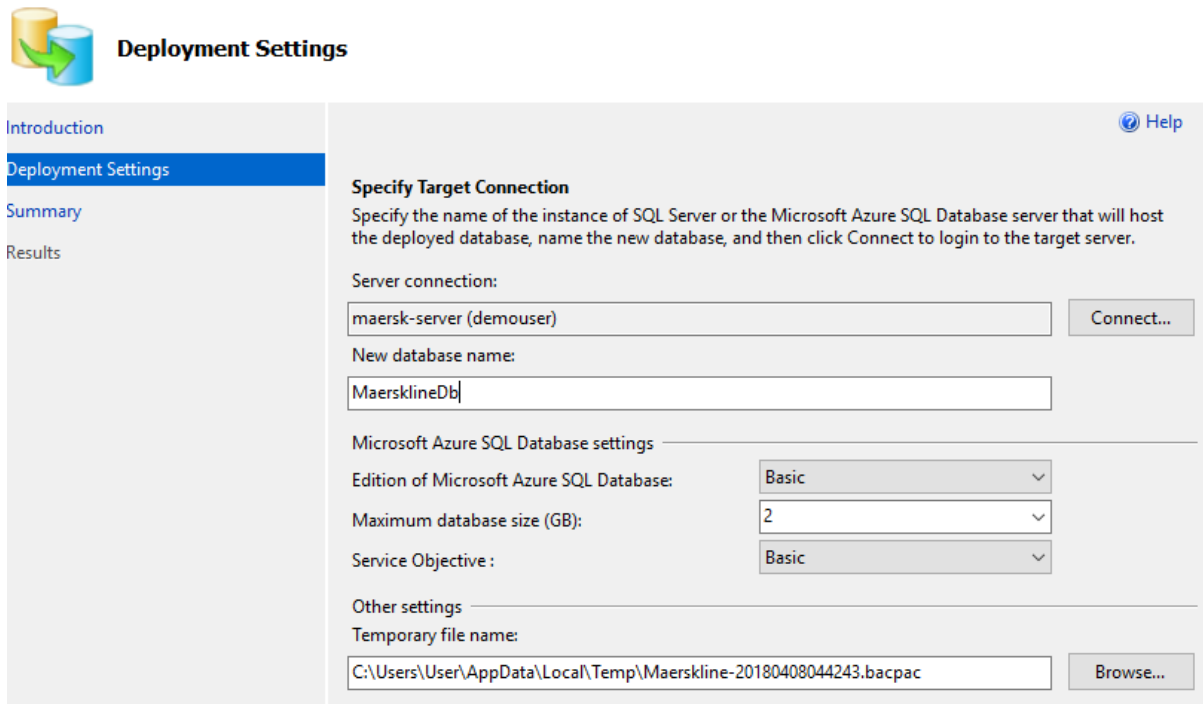


Figure 23: Deployment settings

After clicking next and validating the summary window and click finish and the migration takes place.

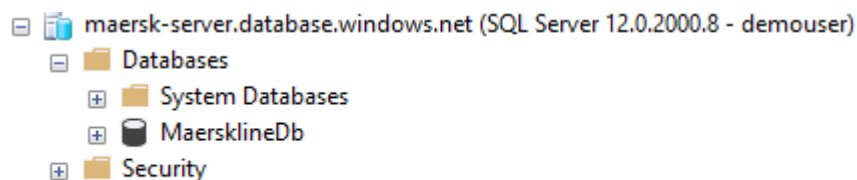


Figure 24: Validate database migration

Lastly, in SQL Server Management Studio, expand the Databases folder, then expand the database migrated to would be shown, this verified that the migration was successful.

Publishing web App on Azure

Before publishing the web app, the connection string in the Web.config file is modified by including the database connection string given by Azure database as shown below.

Connection strings
[Show database connection strings](#)

```
Server=tcp:maersk-server.database.windows.net,1433;Initial Catalog=MaersklineDb;Persist Security Info=False;User ID={your_username};Password={your_password};MultipleActiveResultSets=False;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;
```



Figure 25: Database connection string

The web app developed is published on Azure from Visual studio directly. The publish page can be found by right-clicking the solution and select “Publish”, where it is located under the Solution Explorer.

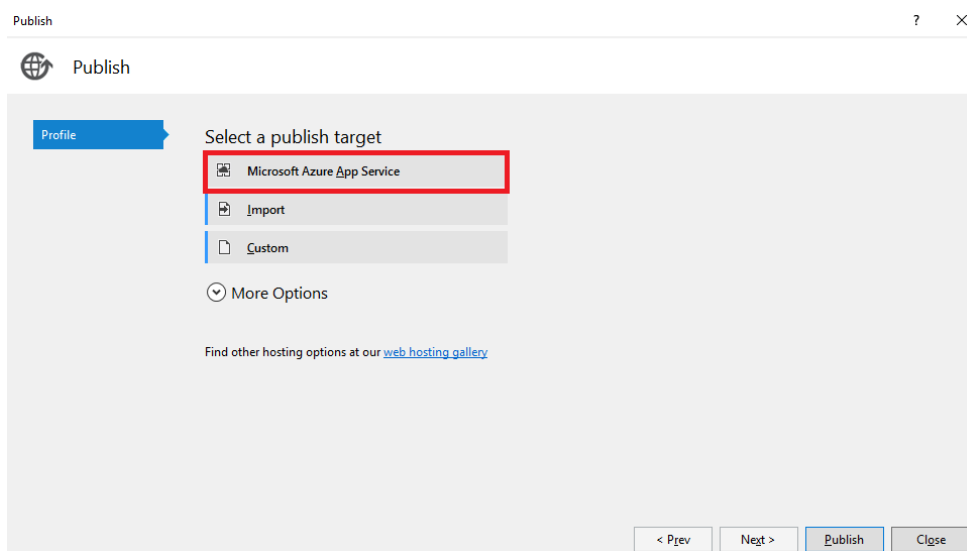
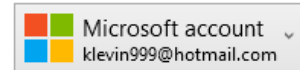


Figure 26: Publish as Microsoft Azure App Service

After that, the Microsoft Azure App Service would be selected as a publish target.

App Service

Host your web and mobile applications, REST APIs, and more in Azure



Subscription

Azure for Students

View

Resource Group

Search

New...

OK

Cancel

Figure 27: Enter app service details

A window that prompt to enter the app service details would be popped up as shown above.

Hosting
Services

App Name Change Type ▼

Subscription

Resource Group
 New...

App Service Plan
 New...

Figure 28: Configure app service details

The figure above shows the configuration of the app service. The app service plan was configured according to the following step.

Configure App Service Plan

An App Service plan is the container for your app. The App Service plan settings will determine the location, features, cost and compute...

App Service Plan

Location

Size

Figure 29: Configure app service plan

The plan selected currently is 'Free', this would be scaled up later. After that proceed with the instruction by clicking next and publish

```
2>Adding ACLs for path (maersklineseaapp)
2>Adding ACLs for path (maersklineseaapp)
2>Publish Succeeded.
2>Web App was published successfully http://maersklineseaapp.azurewebsites.net/
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

Figure 30: Publish succeeded

Secure | <https://maersklineseaapp.azurewebsites.net>

Maersk Line Home About Contact View Schedule Log in

Maersk (SEA region)

With the world's largest shipping fleet and a comprehensive inland network, we provide regular and fast connections to almost anywhere on the globe. The result? You can connect to more customers than ever before – wherever in the world they happen to be.

[Contact us»](#)

Responding to customer needs

In September, Maersk Line introduced Remote Container Management (RCM) to all its refrigerated cargo customers.

[More »](#)

Maersk Line

Maersk Line is the world's largest container shipping company, known for reliable, flexible and eco-efficient services. The Maersk Liner business also includes Safmarine, Seago Line, SeaLand and MCC. The company operates all over the world and has a fleet of 639 ships which sail every major trade lane on the globe.

[More »](#)

The venture adventure

Maersk's Growth organisation is taking bright ideas and developing and commercialising them, with the aim of supporting the company's strategic growth agenda by creating new world-class businesses.

[More »](#)

Figure 31: Application published

After a few minutes, the application is published on Azure as shown on web publish activity tab on Visual studio, and the browser would show the application published (where the link is no longer a localhost).

Application scaling

Scaling allows resources to be dynamically allocated based on the application needs, for example, scaling could quickly and handle unexpected rapid growth or seasonal shifts in demand (Schoeb, 2018); Azure has two workflows for scaling, scale up and scale out. Scale up includes getting more CPU, memory, disk space, and extra features like dedicated virtual machines (VMs), custom domains and certificates, and more. Scale up is done by changing the pricing tier of the App Service plan that suits the app. On the other hand, scale out is done by increasing the number of instances that run the app. This is particularly important when it comes to businesses such as Maersk Web Application, whereas, the workload is expected to grow, hence, more resource is needed to maintain the desired performance level. Scaling is concerned about overall resource optimization, it does not only concern about how to maintain or increase the performance when there is an increased workload; it is also concerned about reducing resource consumption during non-peak. Hence, companies like Maersk need not to worry for the extra resource allocated during high peak will be a waste as they will be automatically deallocated when no longer needed to reduce the operation cost.

Scale up of Maersk application



















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)	

Figure 32: Standard plans (Source: Microsoft Azure)













B1 Basic		B2 Basic		B3 Basic	
1	Core	2	Core	4	Core
1.75	GB RAM	3.5	GB RAM	7	GB RAM
	10 GB Storage		10 GB Storage		10 GB Storage
	Custom domains		Custom domains		Custom domains
	SSL Support SNI SSL Included		SSL Support SNI SSL Included		SSL Support SNI SSL Included
	Up to 3 instance(s) Manual scale		Up to 3 instance(s) Manual scale		Up to 3 instance(s) Manual scale
234.36 MYR/MONTH (ESTIMATED)		468.72 MYR/MONTH (ESTIMATED)		937.44 MYR/MONTH (ESTIMATED)	

Figure 33: Basic plans (Source: Microsoft Azure)

The Standard service plan is designed for running production workloads. Pricing is based on the size and number of instances run. Microsoft claimed that the standard service plan (S1, S2, S3) has a built-in network load balancing support that automatically distributes traffic across instances. A standard plan is chosen over the basic plan as it has greater storage than any basic plan which is 50 GB that could handle a larger amount of transaction within a given time period.

The web application deployed to both the regions will be using the S1 tier App Service. This is mainly due to the restriction of available cost currently. S1 plan provides custom domains which is required for Maersk application. It also allows for up to 10 instances to be created and auto scaled (out or in) based on metrics such as server load, memory usage, or even on a set schedule (Microsoft Azure, 2017) so that the Maersk application developed could match the traffic needs. On the standard plan, a daily backup of the configuration and server data is provided, which is important to ensure no data is lost in any circumstances as data loss is not affordable for any company, for Maersk, losing the booking information of agents or customers would result in chaos. Furthermore, at least a standard plan is required so that the Traffic Manager is supported, users could access the site from the same domain whilst still accessing the most appropriate endpoint based on the traffic routing method and the health of the endpoint (Microsoft Azure, 2017) (Ramees, 2017). When the application is released and growing, both applications should be upgraded to the S2 plan or higher as at least 2 cores are required to handle the traffic which result in a more resilient application and provide a greater specification such as more backup frequencies and better hardware such as SSDs.



Figure 34: Register microsoft.insights for enabling auto-scale

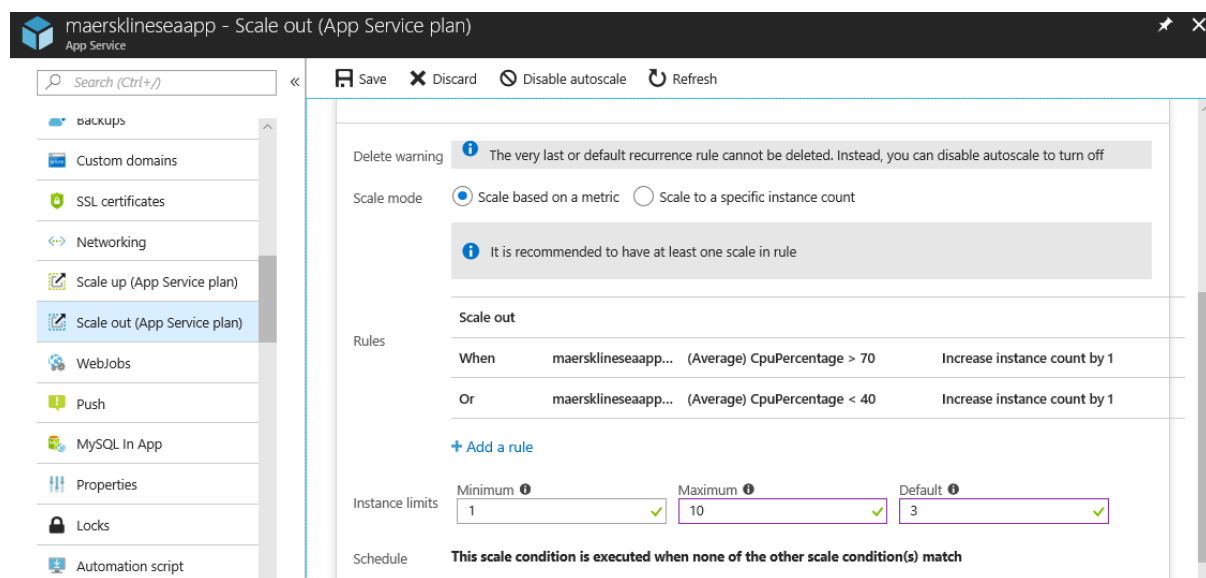


Figure 35: Scale out based on metrics

Once the application is scaled up to S1, a scale out is performed by enabling auto-scale. Before that, the subscription must be registered to microsoft.insights provider. The scale setting is applicable to maersklineseaapp, which is an Azure App Service plans. According to the rule,

this auto scale setting is triggered based on metrics that indicate load or performance (in this case, CPU percentage). For scaling out, the SEA region and West US region will use a default of 3 application instances with auto scaling enabled based on the rules configured. An additional instance will be added whenever the server CPU utilization increases above 70% and an instance will be removed when it drops below 40%.

Other than the scale condition created above, another scale condition is created to suit the business nature. This auto scale setting is triggered at a scheduled date and time. For example, it is foreseen that the application might have more traffic on Monday and Tuesday because the admin is assumed to create and update their new schedules, also, shipment booking from agents during these days are expected to increase since it's the beginning of the week. Similarly, during festive season like Christmas and New Year, where the traffic is expected to be increased as compared to normal days (due to increased booking and shipments), scaling could also be performed, to suit this kind of special occasions. When the CPU percentage is greater than 70% instance count would be increased by 3 during the specified schedule.

Scale out			
Rules	When	maersklineseaapp... (Average) CpuPercentage > 70	Increase instance count by 3
	Or	maersklineseaapp... (Average) CpuPercentage < 30	Increase instance count by 1
+ Add a rule			
Instance limits	Minimum ⓘ	Maximum ⓘ	Default ⓘ
	1	10 ✓	5 ✓
Schedule	<input type="radio"/> Specify start/end dates <input checked="" type="radio"/> Repeat specific days		
Repeat every	<input checked="" type="checkbox"/> Monday <input checked="" type="checkbox"/> Tuesday <input type="checkbox"/> Wednesday <input type="checkbox"/> Thursday <input type="checkbox"/> Friday <input type="checkbox"/> Saturday <input type="checkbox"/> Sunday		
Timezone	(UTC+08:00) Kuala Lumpur, Singapore ▼		
Start time	08:00 ✓		
End time	15:00 ✓		

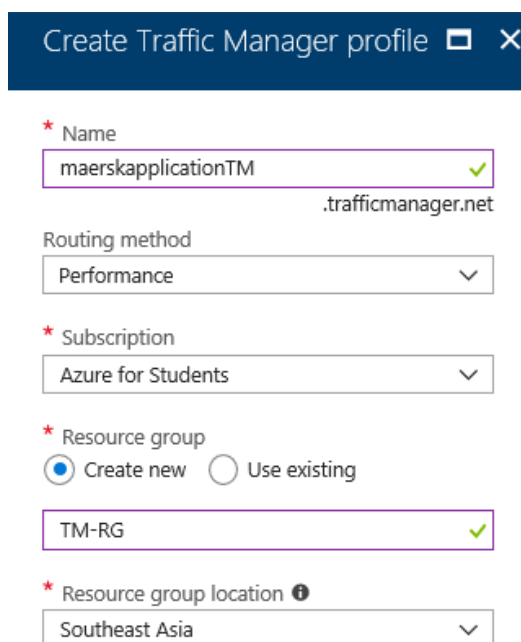
Figure 36: Scale out based on schedule

Traffic manager

Microsoft Azure Traffic Manager allows control the distribution of user traffic for Maersk web app endpoints in different regions. Traffic Manager is resilient to failure, including the failure of an entire Azure region (Microsoft Azure, 2017), therefore, the developer decides to

implement it as the objective is to assure the reliability, hence the traffic manager would act as a failover management. Traffic Manager uses the Domain Name System (DNS) to direct client requests to the most appropriate endpoint based on a traffic-routing method and the health of the endpoints (Microsoft Azure, 2017). Performance routing method is selected as the application is deployed in two regions across the globe, and the developer would like to improve the responsiveness by routing traffic to the location that is 'closest' to the user. 'Closest' does not necessary represented by closest geographic location, instead, the closest endpoint is measured by network latency (lowest network latency). Traffic Manager delivers high availability for Maersk applications by monitoring those endpoints and providing automatic failover when an endpoint goes down.

The steps of configuring the Traffic manager is shown below:



Create Traffic Manager profile

* Name
maerskapplicationTM ✓
.trafficmanager.net

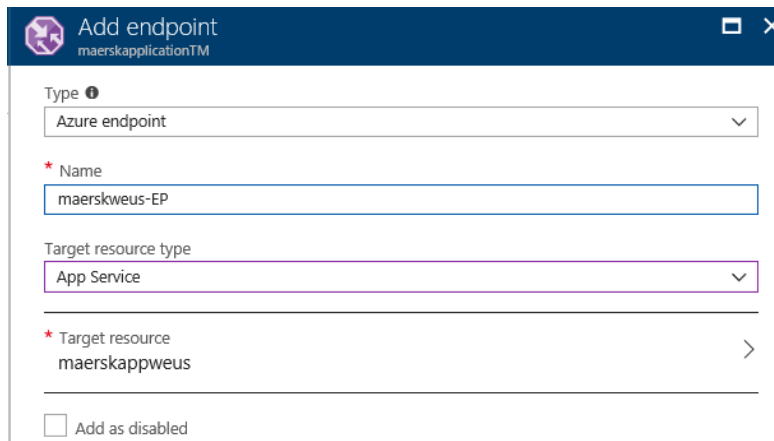
Routing method
Performance ▼

* Subscription
Azure for Students ▼

* Resource group
☒ Create new ☐ Use existing
TM-RG ✓

* Resource group location ⓘ
Southeast Asia ▼

Figure 37: Create Traffic manager profile



Add endpoint
maerskapplicationTM

Type ⓘ
Azure endpoint

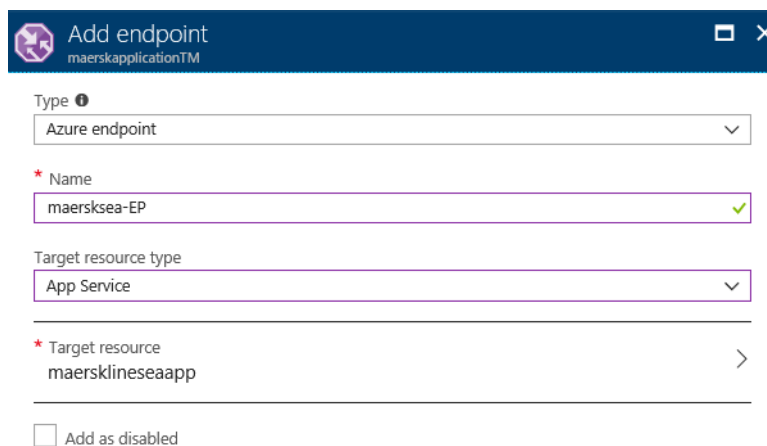
* Name
maerskweus-EP

Target resource type
App Service

* Target resource
maerskappweus

☐ Add as disabled

Figure 38: Adding West US endpoint



Add endpoint
maerskapplicationTM

Type ⓘ
Azure endpoint

* Name
maersksea-EP

Target resource type
App Service

* Target resource
maersklineseaapp

☐ Add as disabled

Figure 39: Adding SEA endpoint

Essentials ^

Resource group (change) TM-RG	DNS name http://maerskapplicationtm.trafficmanager.net
Status Enabled	Monitor status Online
Subscription name (change) Azure for Students	Routing method Performance
Subscription ID 5b050bae-dcba-4286-a254-e9ed4266a434	

Search endpoints

NAME	STATUS	MONITOR STATUS	TYPE	LOCATION
maerskweus-EP	Enabled	Online	Azure endpoint	West US
maersksea-EP	Enabled	Online	Azure endpoint	Southeast Asia

Figure 40: Overview of traffic manager

Once the configuration is completed, the overview of traffic manager is shown as above. The traffic manager has two endpoints, which are in West US and SEA separately. Both endpoints are enabled.

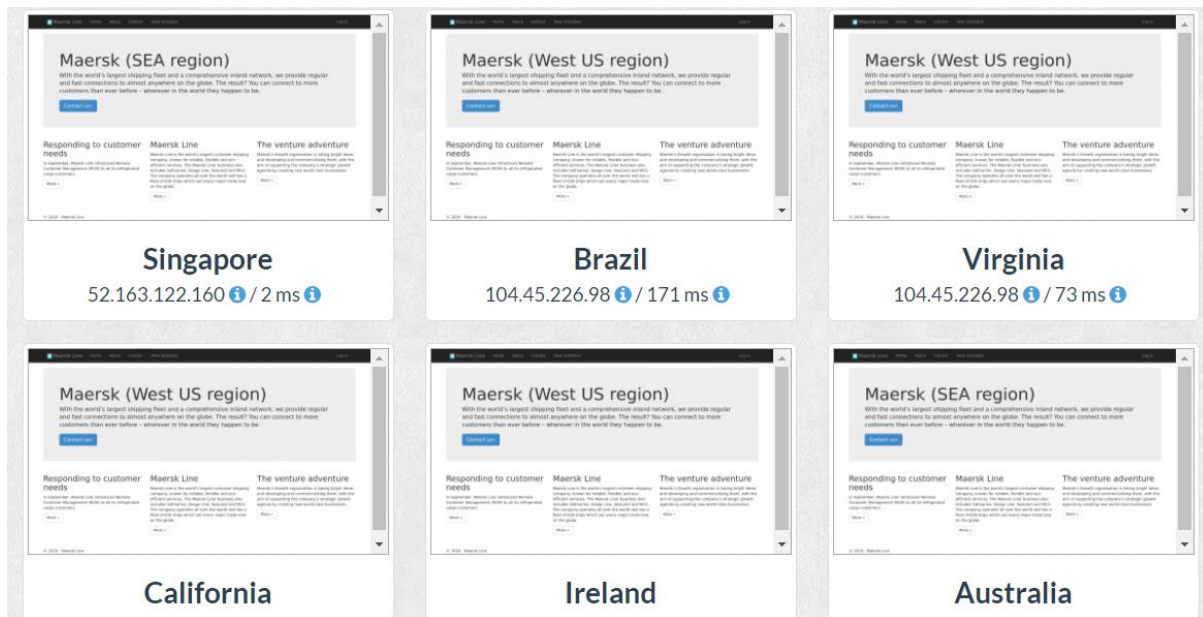


Figure 41: Accessing the site from various regions

Since both endpoints are available, users are directed to different endpoints that has the lowest latency. The URL used for all the regions are the DNS name of the traffic manager.

[▶ Enable profile](#)
[■ Disable profile](#)
[↻ Refresh](#)
[➔ Move](#)
[🗑 Delete profile](#)

Essentials ^

Resource group (change) TM-RG	DNS name http://maerskapplicationtm.trafficmanager.net
Status Enabled	Monitor status Online
Subscription name (change) Azure for Students	Routing method Performance
Subscription ID 5b050bae-dcba-4286-a254-e9ed4266a434	

🔍 Search endpoints

NAME	STATUS	MONITOR STATUS	TYPE	LOCATION
maerskweus-EP	Enabled	Online	Azure endpoint	West US
maersksea-EP	Disabled	Disabled	Azure endpoint	Southeast Asia

Figure 42: Disable SEA endpoint

The screenshot shows a web browser at the URL maerskapplicationtm.trafficmanager.net/. The page features the Maersk Line logo and navigation links (Home, About, Contact, View Schedule) and a 'Log in' button. The main content area is titled 'Maersk (West US region)' and contains the text: 'With the world's largest shipping fleet and a comprehensive inland network, we provide regular and fast connections to almost anywhere on the globe. The result? You can connect to more customers than ever before – wherever in the world they happen to be.' Below this text is a blue button labeled 'Contact us»'. At the bottom, there are three columns of text: 'Responding to customer needs' (with a link to <http://maerskapplicationtm.trafficmanager.net/>), 'Maersk Line' (describing the company's fleet and services), and 'The venture adventure' (describing the company's growth strategy).

Figure 43: West US endpoint

If the SEA region endpoint is disabled, users would be redirected to the West US endpoint, however, the URL is still the same.

Reliability & Performance

The reliability of the application is improved by deploying the application in two regions, namely, SEA and West US region. The traffic manager is configured using performance routing method across the two regions, this can improve both reliability and performance. In term of reliability, when the West US region experience an outage, they would be directed to the SEA region, hence the architecture is resilient to failure, which is reliable. In terms of performance, the traffic manager would improve the Maersk application responsiveness by directing traffic to the endpoint with the lowest network latency for the user, hence the user would have a great browsing experience.

Besides that, each region's web application is assigned with a default of 3 instances for high reliability which could scaled up to 10 instances. The resource allocation could be done effectively with Azure load balancer to distribute the task among all the instances. The auto scaling is applied based on the rule configured as shown previously to handle workload as well as maintain performance during non-peak and peak period, which result in a resilient application. Furthermore, the selected pricing plan provides daily backup which ensure minimal data loss. However, the performance could be improved by scaling to a higher pricing plan as a higher plan provides more cores and RAM.

Test Plan & Testing Discussion (Functional & Performance)

Unit test (Agent)

Login			
Test case	Test Description	Expected Result	Actual Result
Valid Login	1. Enter correct username (agent 012) 2. Enter correct password (password)	Navigate to the agent bookings page and display all the bookings of the agent	As expected
Invalid Login	1. Enter incorrect username (agent 0012) 2. Enter correct password (password)	Display error message	As expected
	1. Enter correct username (agent 012) 2. Enter incorrect password (invalidpassword)	Display error message	As expected

	1. Enter incorrect username (agent 0012)	Display error message	As expected
	2. Enter incorrect password (invalidpassword)		

Register customer			
Test case	Test Description	Expected Result	Actual Result
Valid customer registration	1. Click on Register customer tab at navigation bar located on top of the page 2. Enter valid customer details. 3. Click create	Display registration successful message	As expected
Invalid customer registration	1. Enter invalid email for customer email field 2. Click create	Display error message	As expected
	1. Enter existing NRIC/ passport number for customer NRIC/ passport number field 2. Click create	Display error message	As expected
	1. Enter existing email for customer email field 2. Click create	Display error message	As expected

View booking			
Test case	Test Description	Expected Result	Actual Result
View booking	1. Click on View booking tab at navigation bar located on top of the page	Display all bookings of the agent	As expected
View booking and its associated schedule	1. Click on View booking tab at navigation bar located on top of the page 2. Click View Schedule	Display all bookings of the agent and the schedule details of the selected booking	As expected

Book New Shipment			
Test case	Test Description	Expected Result	Actual Result
Valid item registration	1. Click on Book New Shipment tab at navigation bar located on top of the page 2. Select desired time slot from schedule list 3. Enter valid item details 4. Click Register Item and Booking	Display item registration successful message	As expected
Invalid item registration	1. Click on View booking tab at navigation bar located on top of the page 2. Click View Schedule 3. Click Register Item and Booking	Display error message	As expected

View Schedule			
Test case	Test Description	Expected Result	Actual Result
View Schedule	1. Click on View schedule tab at navigation bar located on top of the page	Display all the schedule details	As expected

Unit test (Admin)

Login			
Test case	Test Description	Expected Result	Actual Result
Valid Login	1. Enter correct username (maerskAdmin) 2. Enter correct password (admin@maersk)	Navigate to the admin bookings page and display all the bookings of the agent	As expected
Invalid Login	1. Enter incorrect username (incorrect username) 2. Enter correct password (admin@maersk)	Display error message	As expected

	1. Enter correct username (maerskAdmin) 2. Enter incorrect password (invalidpassword)	Display error message	As expected
	1. Enter incorrect username (maerskAdmi) 2. Enter incorrect password (invalidpassword)	Display error message	As expected

Register agent			
Test case	Test Description	Expected Result	Actual Result
Valid agent registration	1. Click on Register agent tab at navigation bar located on top of the page 2. Enter valid agent details 3. Click create	Display registration successful message	As expected
Invalid agent registration	1. Enter invalid email for agent email field 2. Click create	Display error message	As expected
	1. Enter existing username for username field 2. Click create	Display error message	As expected

View booking			
Test case	Test Description	Expected Result	Actual Result
View booking	1. Click on View booking tab at navigation bar located on top of the page	Display all bookings	As expected
View booking and its associated schedule	1. Click on View booking tab at navigation bar located on top of the page 2. Click View Schedule	Display all bookings and the schedule details of the selected booking	As expected

View agent			
Test case	Test Description	Expected Result	Actual Result
View agent	1. Click on View agent tab at navigation bar located on top of the page	Display all agent details	As expected

View Schedule			
Test case	Test Description	Expected Result	Actual Result
View Schedule	1. Click on View schedule tab at navigation bar located on top of the page	Display all the schedule details	As expected

Create schedule			
Test case	Test Description	Expected Result	Actual Result
Valid schedule creation	1. Click on create schedule tab at navigation bar located on top of the page 2. Enter valid item details 3. Click Create	Display schedule created message	As expected
Invalid schedule creation	1. Click on create schedule tab at navigation bar located on top of the page 2. Enter invalid item details 3. Click Create	Display error message	As expected
Invalid schedule creation	1. Click on create schedule tab at navigation bar located on top of the page 2. Enter a clashing time slot 3. Click Create	Display error message	As expected

Performance Test

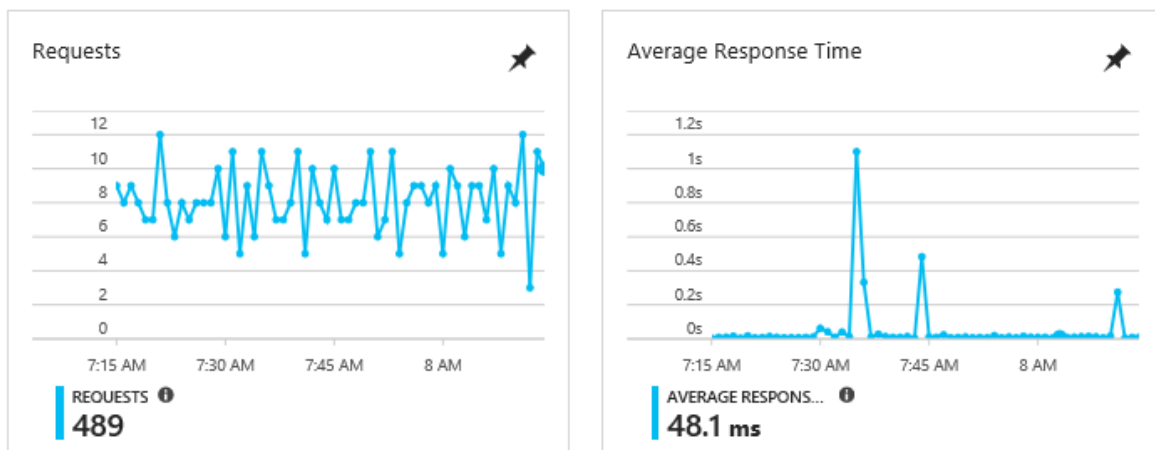


Figure 44: Average response time and requests for the web application

The graph above shows the average response time for the average usage of the web application.

The performance test intended to check the web app's performance before launching it, by assessing whether the app is ready for release based on its ability to handle the traffic during peak use.

The test simulates a user load of 200 to 800 for 3 minutes with each test having 200 user load increments. The figure below shows the one of the test details (400 users for 3 minutes with S3 plan).

The screenshot shows the 'New performance test' configuration interface in Azure. On the left, the 'Performance test' option in the sidebar is highlighted with a red box. Above it, the '+ New' button is also highlighted with a red box. The main configuration area on the right includes a 'CONFIGURE TEST USING' section with 'Test type: ManualTest 1 Url'. Below this, there are input fields for 'NAME' (PeTest01IS3), 'GENERATE LOAD FROM' (Southeast Asia (Web app Location)), 'USER LOAD' (400), and 'DURATION (MINUTES)' (3).

Figure 45: Step to conduct performance test

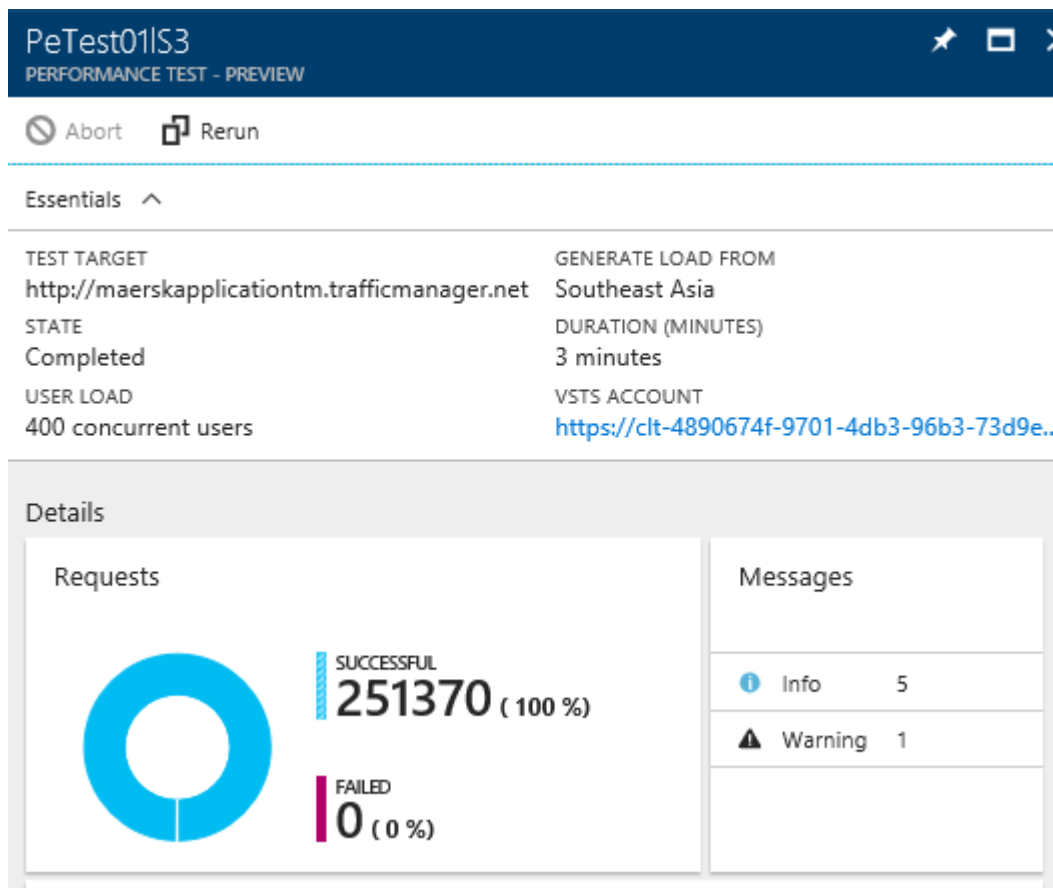


Figure 46: Test details

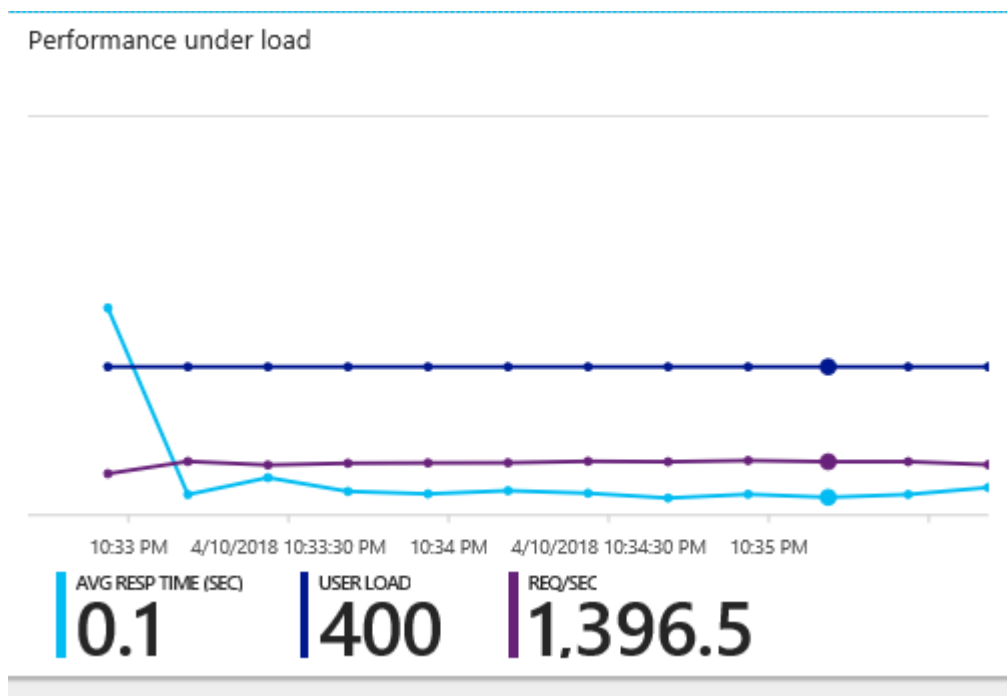


Figure 47: Performance under load

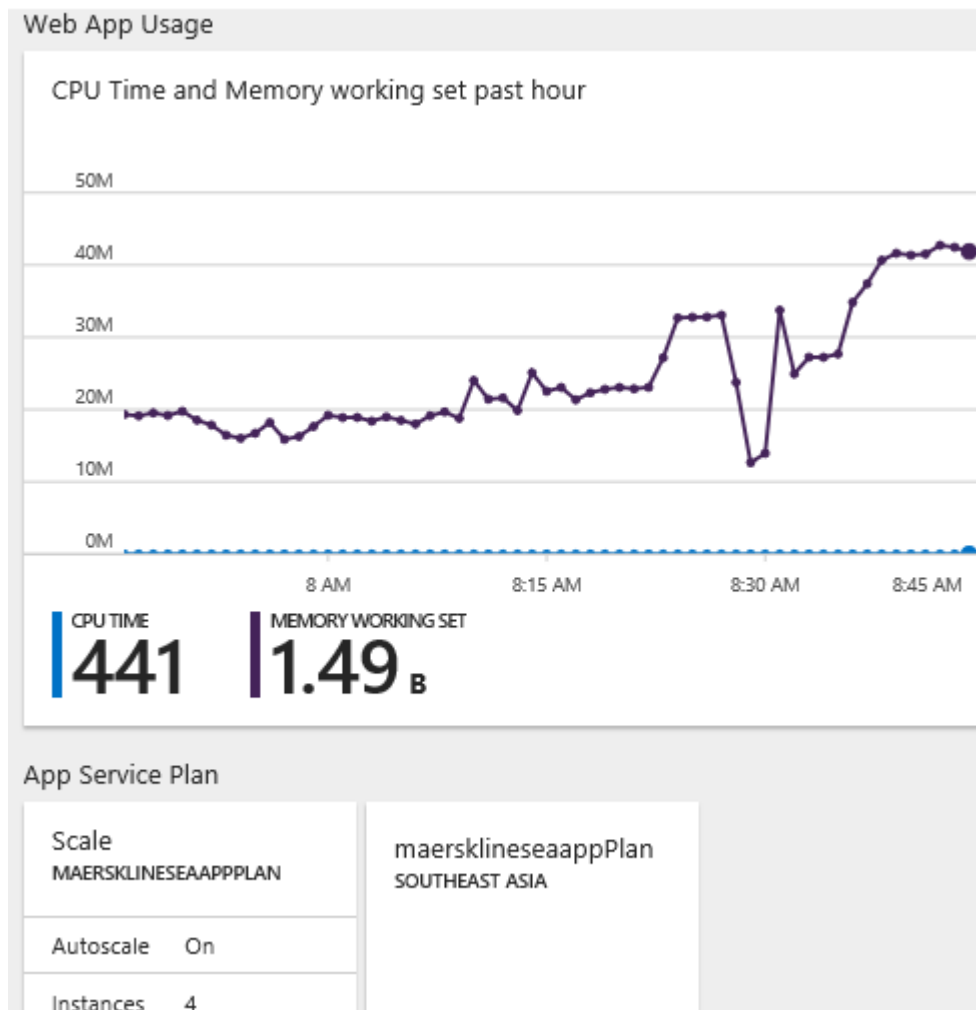


Figure 48: Web app usage

After the test is done, the final result in the form of test details, performance under load, web app usage, and app service plan are viewed. The failed request and the average response time data were used to compare the test result for different application service plan and concurrent users.

These tests are conducted on three app service plans – S1, S2 and S3. The results that were gathered include response time and failed requests. The table below shows the data gathered from the tests described.

Concurrent users \ App Service Plan	200	400	600	800
S1	1.97 3 failed	2.32 0 failed	2.63 13 failed	3.41 46 failed
S2	0.32 0 failed	0.79 0 failed	1.12 5 failed	2.48 28 failed
S3	0.02 0 failed	0.10 0 failed	1.17 7 failed	2.46 24 failed

Analysis

Based on the performance test, it can be seen that a higher pricing plan provides a better performance in terms of handling concurrent users. S2 shows significant improvement over S1, the performance was double in S2 as compared to S1. It can be seen that S1 was able to handle 200 users for 1.97 seconds, while S2 could handle in 0.32 seconds without any failure; also, S2 could handle 600 users faster and lesser failure than S1. On the other hand, S3 does not show much improvement over S2, as the improvements of response time was less than 1 second and the number of failures is more than S2 for 600 users. The maximum acceptable threshold for page load times of this application is 2 seconds. Therefore, it is justified that S1 could only handle 200 concurrent users, while S2 and S3 could handle up to 600 concurrent users. Hence, it is justified that S2 plan is the most suitable plan, as the performance is higher than (about doubled) the S1 plan with lesser failures; also, since S3 does not show a significant improvement over S2, and considering that the cost doubled the S2 plan, hence S2 is the most cost-effective plan for Maersk application.

Managed Database

Azure SQL Database is a relational database-as-a service using the Microsoft SQL Server Engine. Azure SQL database is also known as a platform as a service (PaaS) database. A PaaS is a complete development and deployment environment in the cloud, with resources that enable applications to be delivered (Devi & R, 2015). Users (mostly companies or App and web developers) purchase the required resources from Azure on a pay-as-you-go basis and access them over a secure Internet connection (Microsoft Azure, 2018), they would only be charged for what have been used. Amazon, Google and Windows are well known cloud computing service providers that provides PaaS. All of them provide a managed database service.

According to (Google Cloud, 2018), Google has Cloud SQL that fully-managed relational PostgreSQL and MySQL databases in the cloud. Cloud SQL is suitable for WordPress sites and any other application that is compatible with MySQL or PostgreSQL database (DB Best Technologies, LLC, 2018). Cloud SQL does not require any software installation, the database backups, replication, patches, and updates are managed by Google, while ensuring greater than 99.95% availability (Google Cloud, 2018). Cloud SQL delivers high performance and scalability with up to 10TB of storage capacity, 40,000 IOPS, and 416GB of RAM per instance (Google Cloud, 2018).

AWS (Amazon Web Services) from Amazon offers a wide range of database services. One of the services, Amazon Relational Database Service (Amazon RDS), is claimed to be an easy to administer, highly scalable database's resources, available and durable, secured and affordable solution. Amazon RDS is available for Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle, and Microsoft SQL Server (Amazon Web Services, Inc., 2018). Amazon RDS synchronously replicates the data to a standby instance in a different Availability Zone. Amazon RDS offers automated backups, database snapshots, and automatic host replacement to enhance database reliability (Amazon Web Services, Inc., 2018). Web and mobile applications of Airbnb used Amazon RDS because it reduces time for managing administrative tasks for managing database (Amazon Web Services, Inc., 2018).

In fact, all these companies offer a similar managed database service, where all of them are highly scalable, ensure high availability, ease administration as well as secured. In terms of pricing, all companies have a different pricing plans and the bill is charged based on the usage.

This section would discuss about Azure SQL Database offered by Microsoft as it is used for Maersk Application.

Azure SQL Database is a high-performance, reliable, and secure database, it allows developer to develop websites with any programming language without needing to manage infrastructure.

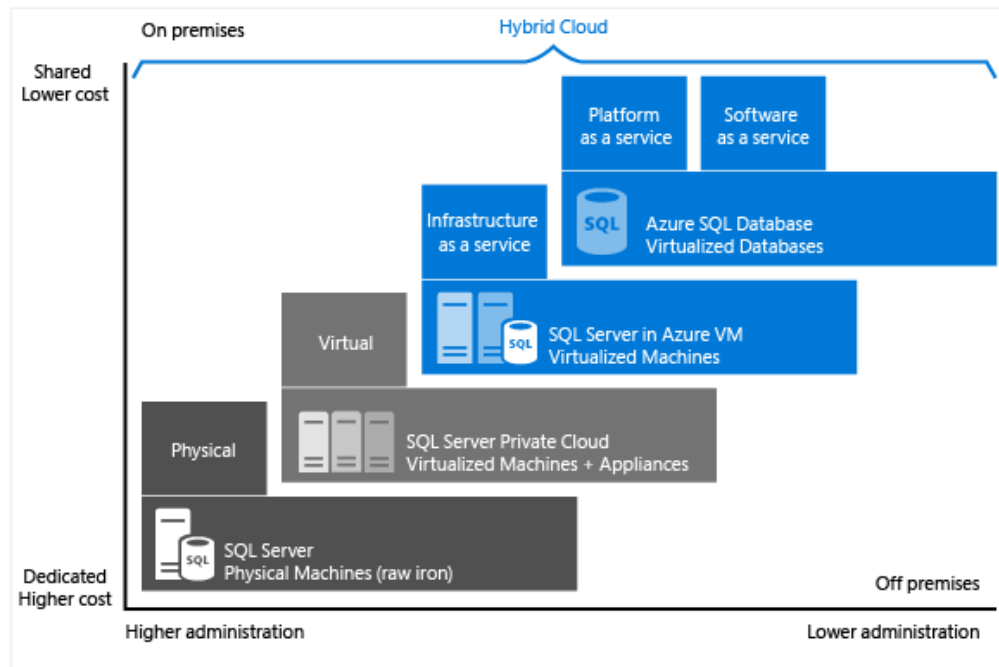


Figure 49: Microsoft's data platform (Source: (Microsoft Azure, 2018))

As seen above, the Azure SQL database required least administration and lower cost due to the database level consolidation and automation, also the administration over the infrastructure is handled by Microsoft.

Azure SQL database is built on standardized hardware and software that is managed by Microsoft. It reduces overall costs for provisioning and managing databases as it is not required to manage virtual machines, operating system or database software, upgrades, high availability, or backups (Microsoft Azure, 2018). Companies no longer need to employ IT resources for configuration and management of the underlying infrastructure, however the on-premises application can access data in Azure SQL Database; hence, they can focus on the application layer that provide value to customers. In terms of hybrid cloud, Azure SQL Server supports SQL Server transactional replication as a subscriber to replicate data (Microsoft Azure, 2018).

According to (Microsoft Azure, 2018), these are the business motivations for choosing Azure SQL Database:

<u>Cost</u>
<p>Currently, SQL Database is billed at a fixed hourly rate based on the service tier and performance level for outgoing Internet traffic at regular data transfer rates. The Azure SQL Database service tiers can be switched based on required performance levels to match application's varied throughput needs without affecting the availability of the database. The Azure SQL require a near-zero administration, therefore, the cost of administer hardware could be reduced. Cost is a key factor where PaaS, to be specific, Azure SQL Database is favourable for many companies, because, companies could save a lot in the long run, by letting Microsoft to handle the configuration and administration of the database.</p> <p>Nowadays, the performance required for an application is unpredictable, as the usage could vary from time-to time, PaaS provides the ability to scale up the application resource to accommodate growth periods or peak workload demand. The resources could be rescaled easily without database downtime and there is no need to rewrite the application code. The pricing is reasonable as well, users just need to pay for what they need when they need it, in other word, it's a pay as you use basis. Therefore, Azure SQL database is a good alternative of investing on physical infrastructure.</p>
<u>Administration</u>
<p>With Azure SQL Database, Microsoft administers the underlying hardware, for example, data replication, configuration and upgrade database software and perform failover management to ensure system availability. Companies can continue to access the database on premise and administer database such as logins, index and query tuning, auditing and security. However, Microsoft would handle the administration of the database engine, server operating system or hardware. Hence, companies need not to hire a team of experts to administer these complicated tasks.</p>
<u>Time to market</u>
<p>Azure provide a programmatic DBA-like functionality, which reduces the concern on managing the underlying operating system and database. Rather, the focus can be put on the application layer and deliver the solution to the market faster. Therefore, the time to market is likely to be shorten as the complexity of administration is offloaded and by transiting to cloud service. This allows companies to gain competitive advantage by releasing their application faster than their competitors.</p>

<u>Availability</u>
Azure provides a 99.99% availability service level agreement (SLA), which helps to keep application running 24/7. Furthermore, SQL Database provides built-in business continuity and global scalability features including, automatic backups, point-in-time restores, active geo-replication, failover groups, zone-redundant databases (Microsoft Azure, 2018).

Based on these factors, Azure SQL database is suitable for applications that aimed for cost savings and performance optimization, as Azure SQL database, which is a PaaS, fully manage the underlying hardware and infrastructure as well as common management operations on databases, which in turn helps reduces time-to-market and provide long-term cost optimization. Furthermore, since Microsoft provides strong availability SLAs for databases, application that requires databases to have a high up time, could make use of Azure DQL database.

The developer has chosen an Azure SQL database over SQL server running on Azure VMs to the eliminate hardware costs and reduces administration costs. As compared to SQL server running on Azure VMs which require a team to configure and manage high availability and disaster recovery for SQL server, a PaaS like Azure SQL has built in high availability, disaster recovery, and upgrade for the database. Hence, the developer (Maersk company) need not to worry about employing IT resources to handle configuration and management of the underlying infrastructure, and more effort could be placed on the application layer.

Conclusion

In conclusion, the project was successfully developed and deployed on Azure cloud environment, whereby the work may be considered for further expansion based on the suggestions provided. The application has developed according to the requirements specifications, application scaling and traffic manager is configured to assure the application reliability and performance.

Throughout the project, the developer has gained a good understanding on cloud computing in its various forms by developing and deploying Maersk container management system to Azure. The Microsoft Azure development environment including the Azure web portal has been thoroughly explored to determine the services that are appropriate to be implemented for the application, so the web application is more effective and efficiency. As noticed, the services used in developing this application include, application service, traffic manager, SQL server, and SQL database.

In addition, the developer has gained knowledge in designing architecture of efficient application for deploying on Azure. In a nutshell, the developer believes that the knowledge and skills gained in this module provide a strong foundation to progress in the career path and the developer has gained a deep interest in cloud as the cloud application development is becoming a trend nowadays due to its cost effectiveness and optimum resource usage.

It can be concluded that Microsoft Azure cloud application is very useful in terms of resource and cost savings for Maersk application as well as many other applications out there.

Last but not least, the developer has researched the top cloud solution providers including Amazon, Microsoft and Google. The research allows the developer to gain an insight on the evolve of information technology throughout these years, cloud computing is trending due to its ability to ensure a high availability, highly scalable, and provide performance according to application needs. With cloud computing, companies could save cost as investment on physical infrastructure and expertise to manage underlying hardware could be drastically reduced. Top companies such as AirAsia, Airbnb, Expedia, and others are adopting cloud computing in their business. Therefore, in the future, the developer would like to learn more about cloud computing due its increased demand and its capabilities to contribute in the information technology areas.

References

Microsoft Azure, 2018. *What is PaaS? Platform as a Service*. [Online] Available at: <https://azure.microsoft.com/en-us/overview/what-is-paas/> [Accessed 22 March 2018].

Amazon Web Services, Inc., 2018. *Amazon Relational Database Service (RDS)*. [Online] Available at: <https://aws.amazon.com/rds/> [Accessed 12 March 2018].

DB Best Technologies, LLC, 2018. *Google Cloud Data Platform*. [Online] Available at: <https://www.dbbest.com/technologies/google-cloud-data-platform/> [Accessed 12 March 2018].

Devi, T. & R, G., 2015. Platform-as-a-Service (PaaS): Model and Security Issues. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 15(1), pp. 151- 161.

Google Cloud, 2018. *Cloud SQL*. [Online] Available at: <https://cloud.google.com/sql/> [Accessed 21 March 2018].

Microsoft Azure, 2017. *Autoscaling*. [Online] Available at: <https://docs.microsoft.com/en-us/azure/architecture/best-practices/auto-scaling> [Accessed 2 March 2018].

Microsoft Azure, 2017. *Database Transaction Units (DTUs) and elastic Database Transaction Units (eDTUs)*. [Online] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-what-is-a-dtu> [Accessed March 3 2018].

Microsoft Azure, 2017. *Traffic Manager routing methods*. [Online] Available at: <https://docs.microsoft.com/en-us/azure/traffic-manager/traffic-manager-routing-methods> [Accessed 21 March 2018].

Microsoft Azure, 2018. *What is the Azure SQL Database service?*. [Online] Available at: <https://docs.microsoft.com/en-us/azure/sql-database/sql-database-technical-overview> [Accessed 13 March 2018].

Ramees, 2017. *Azure Traffic Manager - Part One*. [Online]
Available at: <https://www.c-sharpcorner.com/article/azure-traffic-manager-part-one/>
[Accessed 21 March 2018].

Schoeb, L., 2018. *CLOUD SCALABILITY: SCALE UP VS SCALE OUT*. [Online]
Available at: <https://turbonomic.com/blog/on-technology/cloud-scalability-scale-vs-scale/>
[Accessed 21 MARCH 2018].

External Links:

- Source code (GitHub: https://github.com/tp033/maerskSEA_rep).
- The demo of the application is available at:
<https://web.microsoftstream.com/video/1c94bd2d-64a1-445b-be86-6fba0eda088f>