

|  |  |  |
| --- | --- | --- |
| **Module Code** | : | CT071-3-3-DDAC |
| **Module Name** | : | Designing and Developing Application on the Cloud |
| **Project Name** | : | Maersk Line Container Management System |
| **Student Name** | : | Keanu Yap Yong Jun |
| **TP Number** | : | TP034739 |
| **Lecturer Name** | : | DR. KALAI ANAND A/L RATNAM |
| **Date Completed** | : | 13 April 2018 |

# Acknowledgement

I would like to thanks to my lecturer, Dr. Kalai for his guidance and providing all the necessary information and skill in the field of cloud. He also taught me a lot of skill that how to implement a web application on Azure. Besides, I would like to thank my friends who supporting me and give me feedback. Therefore, I can always in the right path in coding phrase. Last but not least, I would like to thank my parents for their support in my studies.

Table of Contents

[Acknowledgement 2](#_Toc511376773)

[1.0 Introduction 5](#_Toc511376774)

[1.1 Project Background 5](#_Toc511376775)

[1.2 Scope 6](#_Toc511376776)

[1.3 Requirement Specifications 6](#_Toc511376777)

[1.4 Summary of Major Functions/ Solution Contents 6](#_Toc511376778)

[2.0 Project Plan 7](#_Toc511376779)

[3.0 Design 8](#_Toc511376780)

[3.1 Design Consideration 8](#_Toc511376781)

[3.2 Modeling 8](#_Toc511376782)

[3.2.1 Use Case Diagram 8](#_Toc511376783)

[3.2.2 Use Case Specification 9](#_Toc511376784)

[3.2.3 Class Diagram 19](#_Toc511376785)

[3.2.4 Sequence Diagram 20](#_Toc511376786)

[3.2.5 Cloud Architecture 28](#_Toc511376787)

[4.0 Implementation 30](#_Toc511376788)

[4.1 Application Development 30](#_Toc511376789)

[4.2 Azure Publishing 32](#_Toc511376790)

[4.3 Application Scaling 35](#_Toc511376791)

[5.0 Test Plan & Testing Discussion 37](#_Toc511376792)

[5.1 Unit Testing 37](#_Toc511376793)

[5.1.1 Register Agent 37](#_Toc511376794)

[5.1.2 Search Agent 38](#_Toc511376795)

[5.1.3 Register Ship 38](#_Toc511376796)

[5.1.4 Search Ship 38](#_Toc511376797)

[5.1.5 Register Schedule 39](#_Toc511376798)

[5.1.6 Search Schedule 39](#_Toc511376799)

[5.1.7 Register Customer 40](#_Toc511376800)

[5.1.8 Search Customer 40](#_Toc511376801)

[5.1.9 Register Item 41](#_Toc511376802)

[5.1.10 Search Item 41](#_Toc511376803)

[5.1.11 Make Booking 42](#_Toc511376804)

[5.1.12 Search Booking 42](#_Toc511376805)

[5.2 Performance Testing 43](#_Toc511376806)

[5.2.1 Analysis 45](#_Toc511376807)

[6.0 Managed Databases 46](#_Toc511376808)

[7.0 Conclusion 48](#_Toc511376809)

[8.0 References 49](#_Toc511376810)

# 1.0 Introduction

## 1.1 Project Background

Maersk Line is the global container division and the largest operating unit of the A.P. Moller – Maersk Group. It is the world's largest container shipping company having customers through 374 offices in 116 countries. Maersk Line operates over 600 vessels and has a capacity of 2.6 million TEU. Maersk Line was founded in 1928.

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

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

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 Scope

Maersk wanted to design and develop a Container Management System (CMS) that can manage the logistic with efficient way.

## 1.3 Requirement Specifications

1. Manage the entire booking process.

2. Able to scale the solution to meet the needs of demands

## 1.4 Summary of Major Functions/ Solution Contents

1. Design & Develop a single tenant web application hosted on Microsoft Azure as an App Service (Web App) or on AWS Elastic Beanstalk.

2. Consume Relational Database

3. Consist of 5 - 10 interlinked pages.

4. Provide quality content and design.

5. Analyze web application performance with monitoring tools.

6. To be able to scale the solution to meet the needs of demands during peak seasons.

7. Source code to place in source control management services.

# 2.0 Project Plan

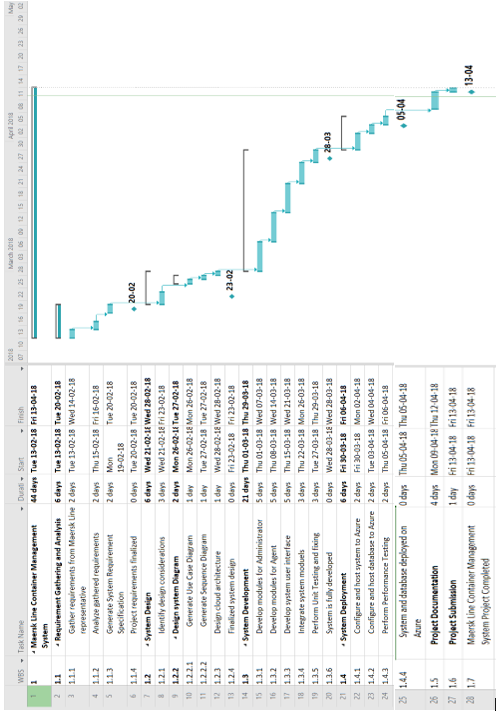


Figure 1: Maersk Line CMS's Project Plan

# 3.0 Design

## 3.1 Design Consideration

There are some assumption or considerations that have been made though the project like:

1. It is in development purpose and proof the concept, therefore the azure credit has just only $200 provided.

## 3.2 Modeling

### 3.2.1 Use Case Diagram

A close up of a map

Description generated with high confidence

Figure 2: Maesrk Line CMS's use case diagram

### 3.2.2 Use Case Specification

#### 3.2.2.1 Register Agent

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-001 |
| Use Case Title | Register Agent |
| Use Case Description | Admin creates a new Agent account with username, password, first name, last name and gender. |
| Included Use Case | - |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin is at Agent registration page. |
| Post condition | Admin registered Agent and redirect to Admin page. |
| Main Scenario | 1. Admin enters Agent’s username that have not used before. 2. Admin enters password. 3. Admin re-enters the same password. 4. Admin enters first name. 5. Admin enters last name. 6. Admin select gender. 7. Admin clicks register button. |
| Alternate Scenario | 1. Admin enters username that been used. 2. Admin enters different password in step 3. |

#### 3.2.2.2 Search Agent

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-002 |
| Use Case Title | Search Agent |
| Use Case Description | Allow Admin to search Agent |
| Included Use Case | UC-001 |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin is logged in. |
| Post condition | System show the Agent that the Admin search. |
| Main Scenario | 1. Admin enters keyword according to the category that Admin choose. 2. Admin clicks search button. |
| Alternate Scenario | 1. Admin enters the keyword that does not exist. |

#### 3.2.2.3 Register Ship

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-003 |
| Use Case Title | Register Ship |
| Use Case Description | Admin creates a new Ship with ship id, ship name, and capacity. |
| Included Use Case | - |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin is at Ship registration page. |
| Post condition | Admin registered Ship and redirect to Admin page. |
| Main Scenario | 1. Admin enters Ship’s id that have not used before. 2. Admin enters ship’s name. 3. Admin enters ship’s capacity. 4. Admin clicks register button. |
| Alternate Scenario | 1. Admin enters ship id that been used. |

#### 3.2.2.4 Search Ship

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-004 |
| Use Case Title | Search Ship |
| Use Case Description | Allow Admin to search Ship |
| Included Use Case | UC-003 |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin is logged in. |
| Post condition | System show the Ship that the Admin search. |
| Main Scenario | 1. Admin enters keyword according to the category that Admin choose. 2. Admin clicks search button. |
| Alternate Scenario | 1. Admin enters the keyword that does not exist. |

#### 3.2.2.5 Register Schedule

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-005 |
| Use Case Title | Register Schedule |
| Use Case Description | Admin creates a new Schedule with id, departure, arrival, departure date, arrival date and ship. |
| Included Use Case | UC-003 |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin is at Schedule registration page. |
| Post condition | Admin registered Schedule and redirect to Admin page. |
| Main Scenario | 1. Admin select departure date and arrival date 2. Admin click check available ship button. 3. Admin enters Schedule id that have not used before. 4. Admin enters departure. 5. Admin enters arrival. 6. Admin select ship. 7. Admin clicks register button. |
| Alternate Scenario | 1. Admin enters schedule that been used. |

#### 3.2.2.6 Search Schedule

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-006 |
| Use Case Title | Search Schedule |
| Use Case Description | Allow Admin to search Schedule |
| Included Use Case | UC-005 |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin is logged in. |
| Post condition | System show the Schedule that the Admin search. |
| Main Scenario | 1. Admin clicks search all button. |
| Alternate Scenario |  |

#### 3.2.2.7 Register Customer

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-007 |
| Use Case Title | Register Customer |
| Use Case Description | Agent creates a new Customer account with id, name, address and gender. |
| Included Use Case | - |
| Extended Use Case | - |
| Primary Actor | Agent |
| Precondition | Agent is at Customer registration page. |
| Post condition | Agent registered Customer and redirect to Agent page. |
| Main Scenario | 1. Agent enters Customer id that have not used before. 2. Agent enters name. 3. Agent enters address. 4. Agent select gender. 5. Agent clicks register button. |
| Alternate Scenario | 1. Agent enters id that been used. |

#### 3.2.2.8 Search Customer

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-008 |
| Use Case Title | Search Customer |
| Use Case Description | Allow Agent to search Customer |
| Included Use Case | UC-007 |
| Extended Use Case | - |
| Primary Actor | Agent |
| Precondition | Agent is logged in. |
| Post condition | System show the Customer that the Agent search. |
| Main Scenario | 1. Agent enters keyword according to the category that Agent choose. 2. Agent clicks search button. |
| Alternate Scenario | 1. Agent enters the keyword that does not exist. |

#### 3.2.2.9 Register Item

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-009 |
| Use Case Title | Register Item |
| Use Case Description | Agent creates a new Item with id, item name, customer id, and weight. |
| Included Use Case | UC-007 |
| Extended Use Case | - |
| Primary Actor | Agent |
| Precondition | Agent is at Item registration page. |
| Post condition | Agent registered Item and redirect to Agent page. |
| Main Scenario | 1. Agent enters Item’s id that have not used before. 2. Agent enters Item’s name. 3. Agent enters Customer’s id. 4. Agent enters weight. 5. Admin clicks register button. |
| Alternate Scenario | 1. Agent enters ID that been used. 2. Agent enters invalid customer id. |

#### 3.2.2.10 Search Item

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-010 |
| Use Case Title | Search Item |
| Use Case Description | Allow Agent to search Item |
| Included Use Case | UC-009 |
| Extended Use Case | - |
| Primary Actor | Agent |
| Precondition | Agent is logged in. |
| Post condition | System show the Item that the Agent search. |
| Main Scenario | 1. Agent enters keyword according to the category that Agent choose. 2. Agent clicks search button. |
| Alternate Scenario | 1. Agent enters the keyword that does not exist. |

#### 3.2.2.11 Make Booking

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-011 |
| Use Case Title | Make Booking |
| Use Case Description | Agent creates a new Booking account with id, item id and schedule id. |
| Included Use Case | UC-006, UC-009 |
| Extended Use Case | - |
| Primary Actor | Agent |
| Precondition | Agent is at make booking page. |
| Post condition | Agent make a booking and redirect to Agent page. |
| Main Scenario | 1. Agent enters Booking id that have not used before. 2. Agent selects item id. 3. Agent selects schedule id. 4. Agent clicks register button. |
| Alternate Scenario | 1. Admin enters Booking id that been used. |

#### 3.2.2.12 Search Booking

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-012 |
| Use Case Title | Search Booking |
| Use Case Description | Allow Agent to search Booking |
| Included Use Case | UC-011 |
| Extended Use Case | - |
| Primary Actor | Agent |
| Precondition | Agent is logged in. |
| Post condition | System show the Booking that the Agent search. |
| Main Scenario | 1. Agent clicks search all button. |
| Alternate Scenario |  |

#### 3.2.2.13 Manage Booking

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-013 |
| Use Case Title | Manage Booking |
| Use Case Description | Allow Admin to search all pending booking |
| Included Use Case | UC-011 |
| Extended Use Case | UC-014, UC-015 |
| Primary Actor | Admin |
| Precondition | Admin is logged in. |
| Post condition | System show the Pending booking that the Admin search. |
| Main Scenario | 1. Admin clicks search all button. |
| Alternate Scenario |  |

#### 3.2.2.14 Approve Booking

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-014 |
| Use Case Title | Approve Booking |
| Use Case Description | Allow Admin to search all pending booking |
| Included Use Case | - |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin has run UC-013. |
| Post condition | Admin update the booking. |
| Main Scenario | 1. Admin selects the pending booking. 2. Admin select “Approve”. 3. Admin click confirm button. |
| Alternate Scenario |  |

#### 3.2.2.15 Reject Booking

|  |  |
| --- | --- |
| Title | Description |
| Use Case ID | UC-015 |
| Use Case Title | Approve Booking |
| Use Case Description | Allow Admin to search all pending booking |
| Included Use Case | - |
| Extended Use Case | - |
| Primary Actor | Admin |
| Precondition | Admin has run UC-013. |
| Post condition | Admin update the booking. |
| Main Scenario | 1. Admin selects the pending booking. 2. Admin select “Reject”. 3. Admin click confirm button. |
| Alternate Scenario |  |

### 3.2.3 Class Diagram

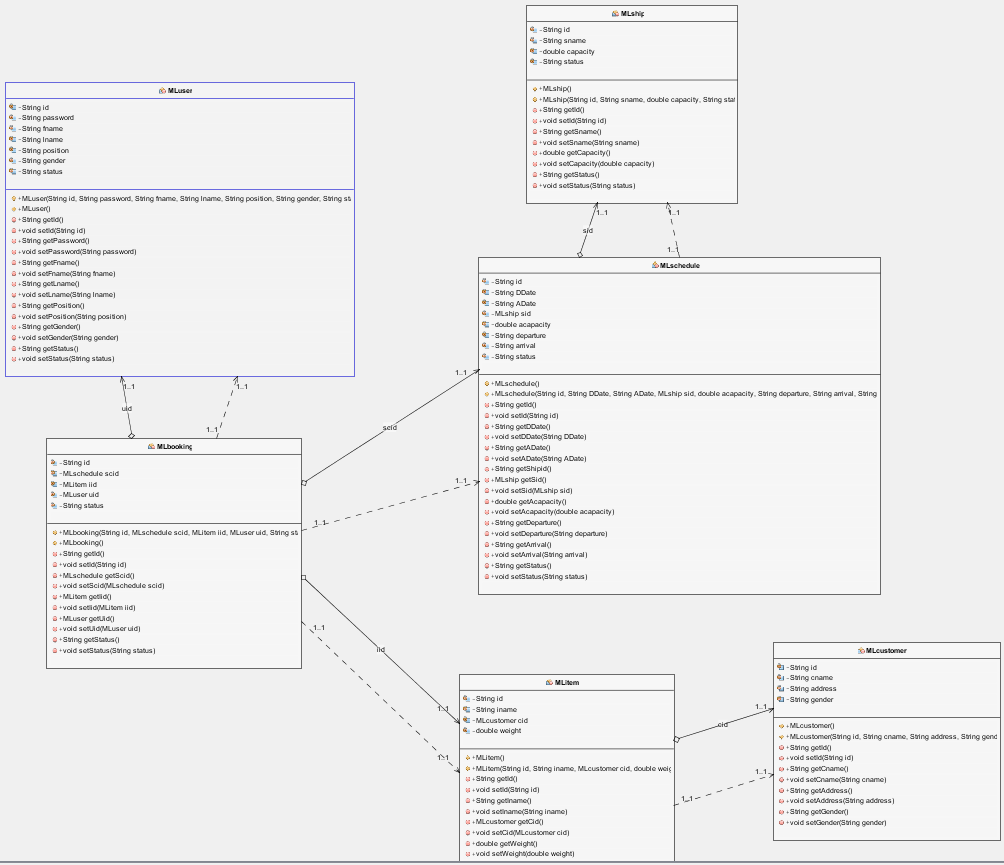


Figure 3: Maesrk Line CMS's Class Diagram

### 3.2.4 Sequence Diagram

#### 3.2.4.1 Register Agent

A screenshot of a social media post

Description generated with very high confidence

Figure 4: Register Agent

#### 3.2.4.2 Search Agent

A screenshot of a social media post

Description generated with very high confidence

Figure 5: Search Agent

#### 3.2.4.3 Register Ship

A screenshot of a social media post

Description generated with very high confidence

Figure 6: Register Ship

#### 3.2.4.4 Search Ship

A screenshot of a social media post

Description generated with very high confidence

Figure 7: Search Ship

#### 3.2.4.5 Register Schedule

A screenshot of a cell phone

Description generated with very high confidence

Figure 8: Register Schedule

#### 3.2.4.6 Search Schedule

A screenshot of a social media post

Description generated with very high confidence

Figure 9: Search Schedule

#### 3.2.4.7 Register Customer

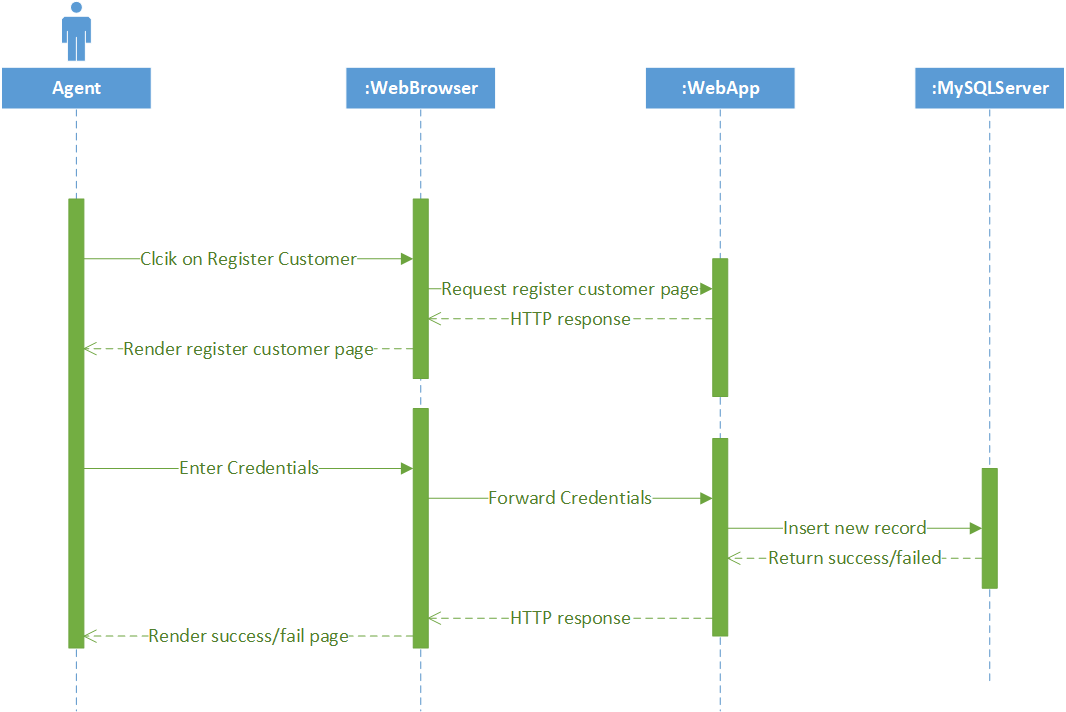


Figure 10: Register Customer

#### 3.2.4.8 Search Customer

A screenshot of a social media post

Description generated with very high confidence

Figure 11: Search Customer

#### 3.2.4.9 Register Item

A screenshot of a social media post

Description generated with very high confidence

Figure 12: Register Item

#### 3.2.4.10 Search Item

A screenshot of a social media post

Description generated with very high confidence

Figure 13: Search Item

#### 3.2.4.11 Make Booking

A screenshot of a social media post

Description generated with very high confidence

Figure 14: Make Booking

#### 3.2.4.12 Search Booking

A screenshot of a social media post

Description generated with very high confidence

Figure 15: Search Booking

#### 3.2.4.13 Manage Booking

A screenshot of a social media post

Description generated with very high confidence

Figure 16: Manage Booking

### 3.2.5 Cloud Architecture

A close up of a map

Description generated with high confidence

Figure 17: Maersk Line CMS's Cloud Architecture

Figure 17 stated that the cloud architecture that used to deploy the application to the Azure cloud platform. Marsrk Line is focusing on the Southeast Asia region. The developer will build the application by using the budget. The application is written using jsp, java servlet and MySQL. Therefore, MySQL server is provision rather than the standard SQL server. Besides, it also placed in Southeast Asia region linked with the SEA web application. Figure 17 has been implemented according to the budget restrictions.

If Maersk Line wanted to increase the budget, the ideal architecture will be added like traffic manager, geo replication and more instances. Due to Maesrk Line is a big company in different country, Maesrk Line can use scale up the plan so that they have geo-replication to handle unexpected traffic. Not only that, there is a traffic manager will be monitoring the connection between the server and the client. The purpose is to decide the best route between them.

# 4.0 Implementation

## 4.1 Application Development

The development of Maersk Line CMS web application is done with the Netbean IDE 8.2 with MySQL database. It also uses the concept of MVC which is Model, View and Controller. The purpose is to isolate the application’s concern (tutorialspoint, 2018). The file structure has shown below.

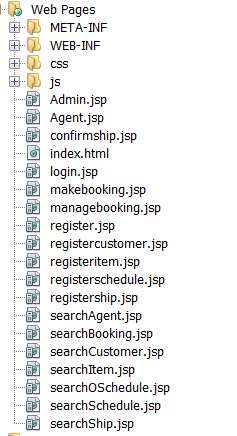
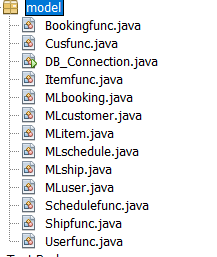
 

Figure 18: File Structure 1 Figure 19: File Structure 2

Figure 18 is the file structure that store all JSP file. In the JSP file, it contains all the front-end of the web application. The language that been used in JSP is HTML, CSS and JAVASCRIPT. Figure 19 is the file structure 2 which contain the model class which used the MVC concept. It has used Model and View in this model class. The view is represented the visualization of the result data (tutorialspoint, 2018) and the model is to represent the object that carry data. It also contains the logic to update the controller in some situation (tutorialspoint, 2018).

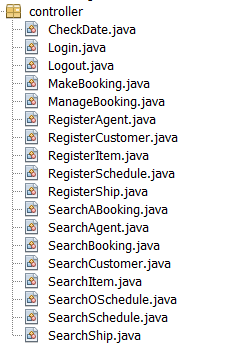


Figure 20: File Structure 3

Figure 20 is the file structure 3 which contain all the servlet. In the servlet, it contains the function that will work in the web application. It control the data flow into the model and the view which also update whenever the data has changed (tutorialspoint, 2018).

The application UI is develop combination of HTML, CSS and JAVASCRIPT. The sample of homepage of Maesrk Line CMS has shown below. The final output can be viewed in the link, <https://github.com/keanuyap/ddac>. In the link also contain the softcopy of documentation and source code.

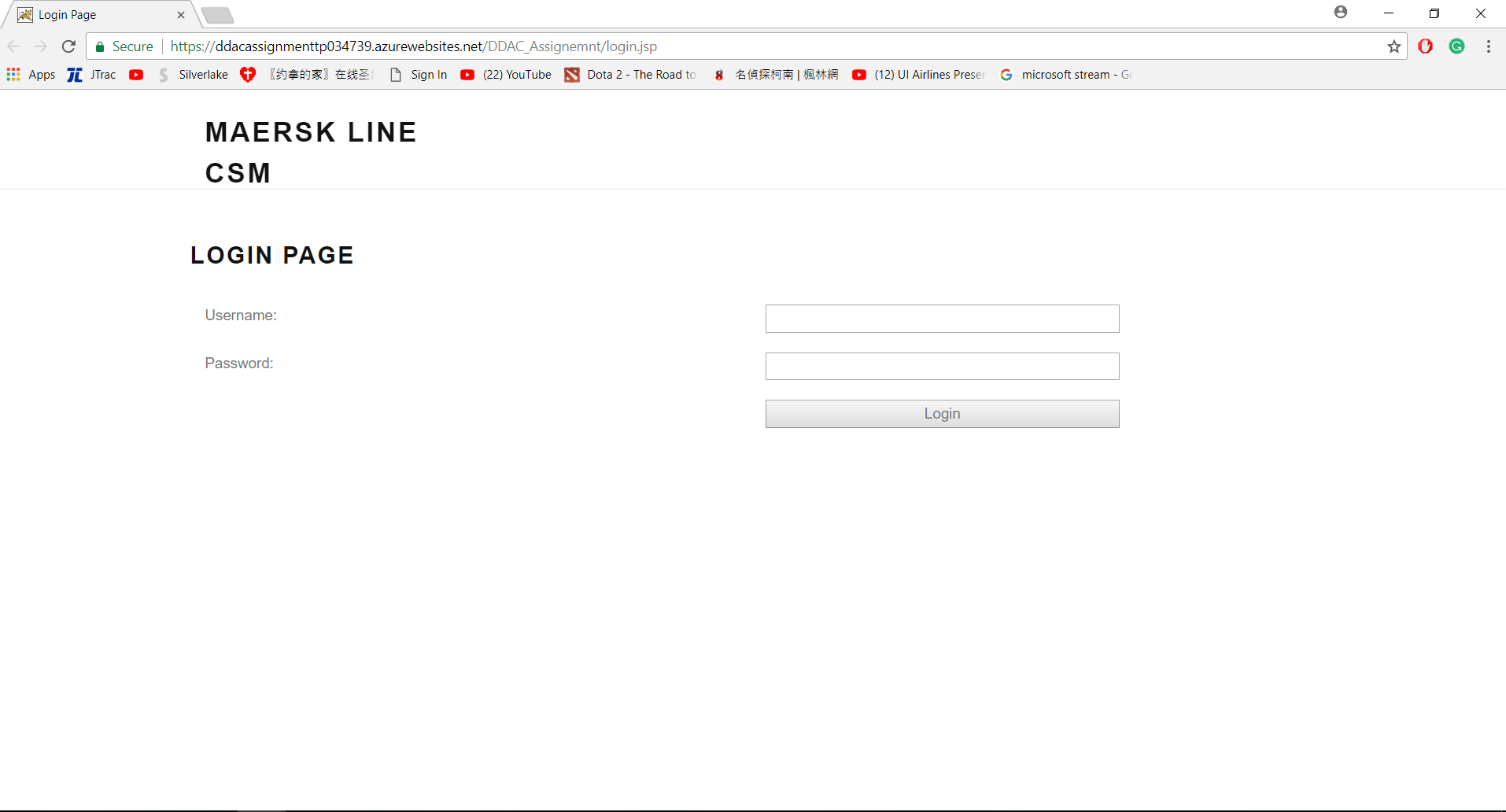


Figure 21: Home Page of Maesrk Line CMS

## 4.2 Azure Publishing

The application is published by using Azure. The first step is creating a new web application container under “Web + Mobile” category. Apache Tomcat 8 is the web application container which take app name, subscription, resource group and app service plan. Before this, it need to create a resource group. However, it need to fill in unique app name and select the region. The app name is ddacassignmenttp034739. The region that selected is Southeast Asia. The resource group is the resource group that has been created earlier. Lastly is the app service plan has been chosen the B1 basic plan. The plan is chosen by the developer due to the budget limitation. Figure 22 is the example of the Apache Tomcat 8.

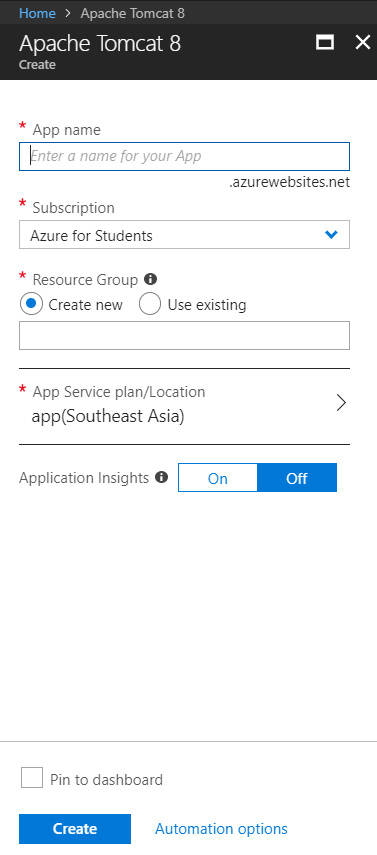


Figure 22: Apache Tomcat 8

After that, it need to set up the Azure Database for MySQL server. It need to fill in a unique server name, subscription, resource group, server admin login name and password, location, version and the pricing tier. The reason I choose Azure Database for MySQL server is because it provides all the feature like scaling and monitoring but with MySQL (Microsoft, 2018). The server will be created to host the database in the region that chosen with username and password to access the server. Figure 23 is the example of how to set up the Azure Database for MySQL server.

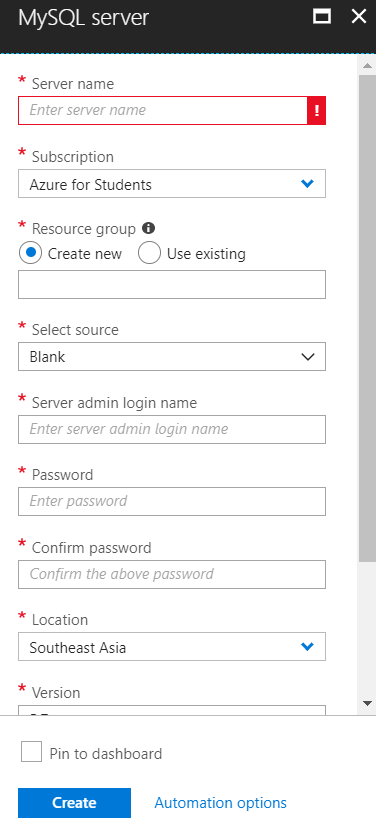


Figure 23: Azure Database for MySQL server

Then, publish the web application to azure with the web service that created earlier. For using Netbeans IDE, it need to use the advanced tool to publish the web application. It uses power shell to publish. Figure 24 is the way to open advance tool. After open the advanced tool, select the power shell in the debug console tab above. The purpose is to drag the web application war file from Netbeans IDE to it. The location is site\wwwroot\bin\apache-tomcat-8.5.24\webapps which stated in Figure 25. There are also some configurations need to be done.

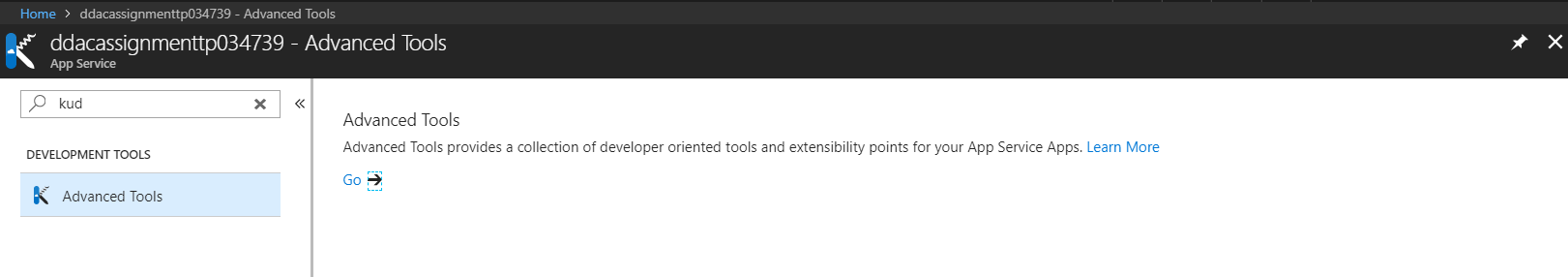


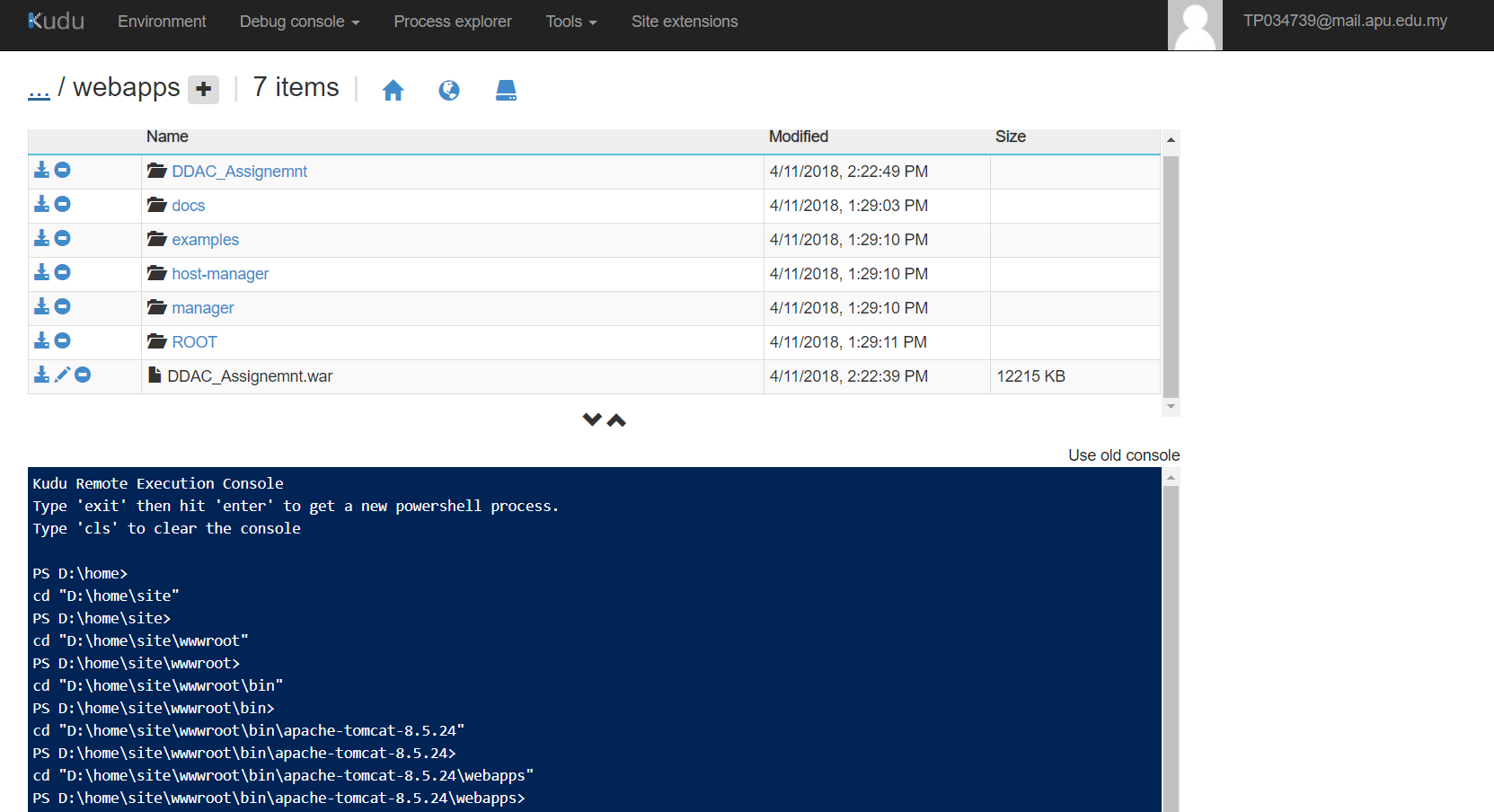
Figure 24: How to find Advanced tools

Figure 25: Location

After published the web application to Azure, it need to link the Azure Database for MySQL server to MySQL workbench. It need to set the hostname, username and password to connect it. Therefore, the web application can update the database and the result can be shown in MySQL workbench.

## 4.3 Application Scaling

According to (Azure, 2017), Azure cloud environment provide auto scaling for the web service which the resources can be allocated dynamically according to the needs which match with the performance requirement. The web application deployed in Southeast Asia Region which will use the B1 tier App Service. The reason is because of the budget restriction. Figure 26 below is the plans of that Azure provided.

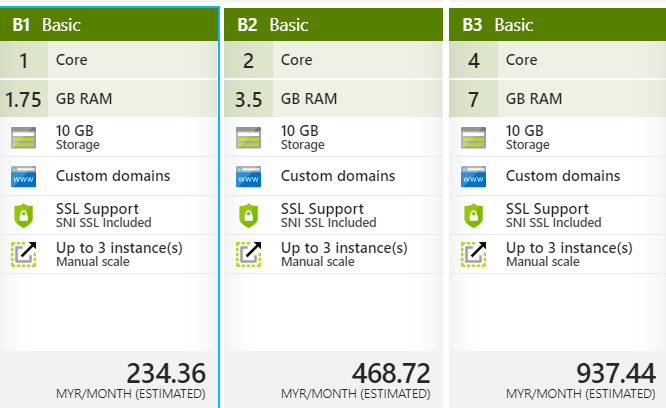


Figure 26: Pricing tier

First of all, it has a great storage which contain 10 GB. Due to Maersk Line CMS is still in development phrase, 10 GB is more than enough for it. Next, it has custom domains with SSL support which included SNI and SSL. Although it does not have auto scaling, but it has allowed to scale up to 3 instances which is suitable for Maesrk Line CMS web application. However, if Maesrk Line decided to increase the budget, the Standard plan is highly recommended due to it has autoscaling, 50GB storage, daily backup feature, 5 deployment slot which can hold recent deployment restore point incase some unexpected situation happened, and it has traffic manager which can handle the connection between client and server. The purpose is to ensure that it has stable communication.



Figure 27: Scale out

Figure 27 stated that it can scale the instances, but manually which maximum is 3 instances. If upgraded to S1 plan, it has auto scaling feature. For example, the plan is crated whenever the CPU usage is greater than 80%, another instance will have deployed automatically. The instance will have removed whenever the CPU usage is lower than 30%.

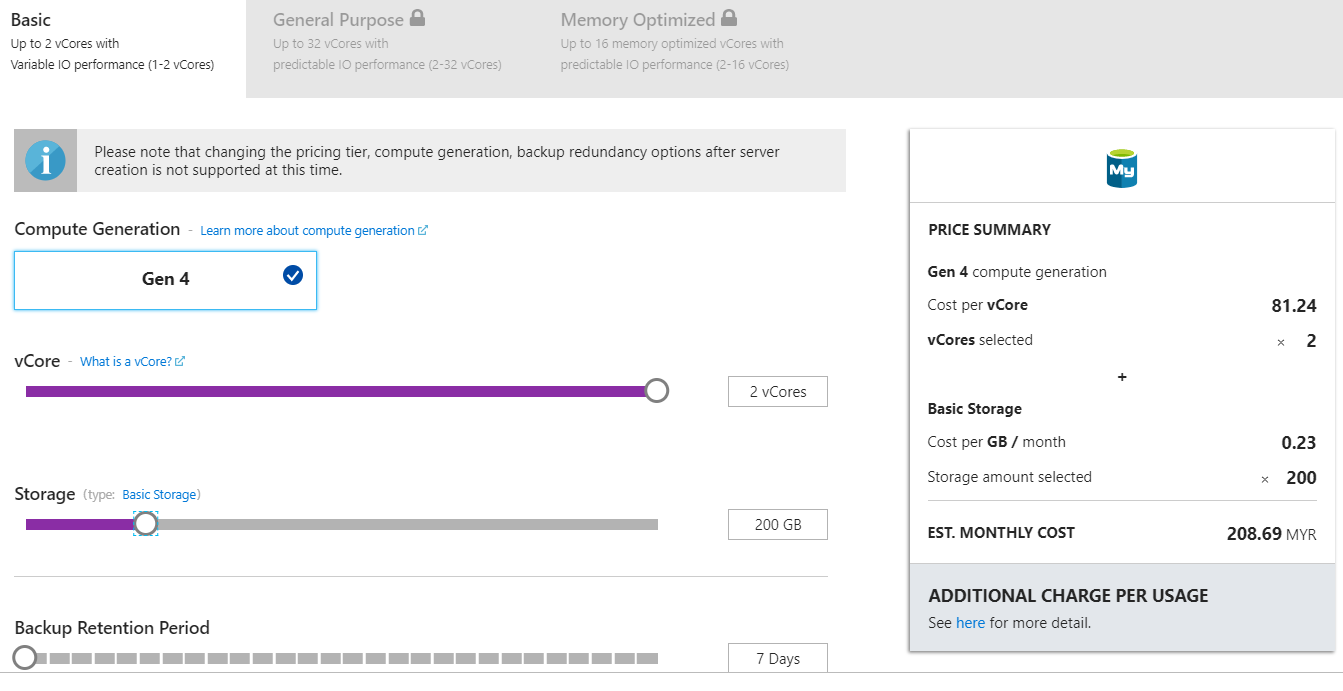


Figure 28: Pricing tier of MySQL server

Figure 28 is the pricing tier of MySQL server. For Maersk Line CMS web application, the Azure MySQL will be used the basic plan. The current plan that chosen is 2 vCore and 200GB storage. It will have backup retention every 7 days. If Maersk Line grow very fast and decide to increase the budget, a greater plan like standard plan can be subscripted to handle larger amount of data.

# 5.0 Test Plan & Testing Discussion

## 5.1 Unit Testing

### 5.1.1 Register Agent

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Test Case | Username Text box | Password Text box | Retype Password Textbox | First Name Textbox | Last Name Textbox | Expected Message | Actual Message |
| UT-001 | Empty | Empty | Empty | Empty | Empty | “Please fill out this required Field” | “Please fill out this required Field” |
| UT-002 | Invalid Input | Valid Input | Valid Input | Valid Input | Valid Input | “Username has been used” | “Username has been used” |
| UT-003 | Valid Input | Valid Input | Invalid Input | Valid Input | Valid Input | “Password does not match | “Password does not match |
| UT-004 | Valid Input | Valid Input | Valid Input | Valid Input | Valid Input | “Register Successfully” | “Register Successfully” |

### 5.1.2 Search Agent

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Attraction Text box | Expected Result | Actual Result |
| UT-005 | Empty | There is no result in the table. | As Expected |
| UT-006 | Invalid Input | There is no result in the table. | As Expected |
| UT-007 | Valid Input | Display the Agent’s details. | As Expected |

### 5.1.3 Register Ship

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case | Ship ID Text box | Ship Name Text box | Capacity Textbox | Expected Message | Actual Message |
| UT-008 | Empty | Empty | Empty | “Please fill out this required Field” | “Please fill out this required Field” |
| UT-009 | Invalid Input | Valid Input | Valid Input | “Ship ID has been used” | “Ship ID has been used” |
| UT-010 | Valid Input | Valid Input | Valid Input | “Register Successfully” | “Register Successfully” |

### 5.1.4 Search Ship

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Attraction Text box | Expected Result | Actual Result |
| UT-011 | Empty | There is no result in the table. | As Expected |
| UT-012 | Invalid Input | There is no result in the table. | As Expected |
| UT-013 | Valid Input | Display the Ship’s details. | As Expected |

### 5.1.5 Register Schedule

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case | Schedule ID Text box | Departure Text box | Arrival Textbox | Expected Message | Actual Message |
| UT-014 | Empty | Empty | Empty | “Please fill out this required Field” | “Please fill out this required Field” |
| UT-015 | Invalid Input | Valid Input | Valid Input | “Schedule ID has been used” | “Schedule ID has been used” |
| UT-016 | Valid Input | Valid Input | Valid Input | “Register Successfully” | “Register Successfully” |

### 5.1.6 Search Schedule

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Show All Button | Expected Result | Actual Result |
| UT-017 | Pressed | Display the Schedule’s details. | As Expected |

### 5.1.7 Register Customer

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case | Customer ID Text box | Customer name Textbox | Address Textbox | Expected Message | Actual Message |
| UT-018 | Empty | Empty | Empty | “Please fill out this required Field” | “Please fill out this required Field” |
| UT-019 | Invalid Input | Valid Input | Valid Input | “Customer ID has been used” | “Customer ID has been used” |
| UT-020 | Valid Input | Valid Input | Valid Input | “Register Successfully” | “Register Successfully” |

### 5.1.8 Search Customer

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Attraction Text box | Expected Result | Actual Result |
| UT-021 | Empty | There is no result in the table. | As Expected |
| UT-022 | Invalid Input | There is no result in the table. | As Expected |
| UT-023 | Valid Input | Display the Customer’s details. | As Expected |

### 5.1.9 Register Item

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Case | Item ID Text box | Item name Textbox | Weight Textbox | Expected Message | Actual Message |
| UT-024 | Empty | Empty | Empty | “Please fill out this required Field” | “Please fill out this required Field” |
| UT-025 | Invalid Input | Valid Input | Valid Input | “Item ID has been used” | “Item ID has been used” |
| UT-026 | Valid Input | Valid Input | Valid Input | “Register Successfully” | “Register Successfully” |

### 5.1.10 Search Item

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Attraction Text box | Expected Result | Actual Result |
| UT-027 | Empty | There is no result in the table. | As Expected |
| UT-028 | Invalid Input | There is no result in the table. | As Expected |
| UT-029 | Valid Input | Display the Item’s details. | As Expected |

### 5.1.11 Make Booking

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Booking ID Text box | Expected Message | Actual Message |
| UT-030 | Empty | “Please fill out this required Field” | “Please fill out this required Field” |
| UT-031 | Invalid Input | “Booking ID has been used” | “Booking ID has been used” |
| UT-032 | Valid Input | “Register Successfully” | “Register Successfully” |

### 5.1.12 Search Booking

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Show All Button | Expected Result | Actual Result |
| UT-033 | Pressed | Display the Schedule’s details. | As Expected |

## 5.2 Performance Testing

The performance testing has been conducted by Maesrk Line CMS. It is one of the feature of Azure Portal. The performance testing has been conducted in 3 different plan which is B1, B2, S1. The performance testing is test according to the number of user load within a given time period. The user load that will be test is 500-1000 user in 2 minutes. Figure 29 stated the performance test which cover name, server, user load and the given time.

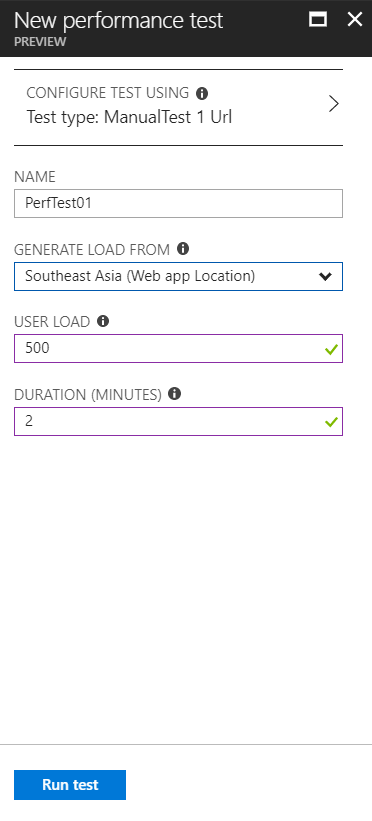


Figure 29: Performance Test

There are some sample results from the performance test which take in SEA server, 500 users load and last for 2 minutes. There is a pie chart which stated that how many it success and fail with the percentages. Moreover, it also has stated the performance under load and the CPU time and memory working usage. Figure 30 and 31 is the example of the sample results.

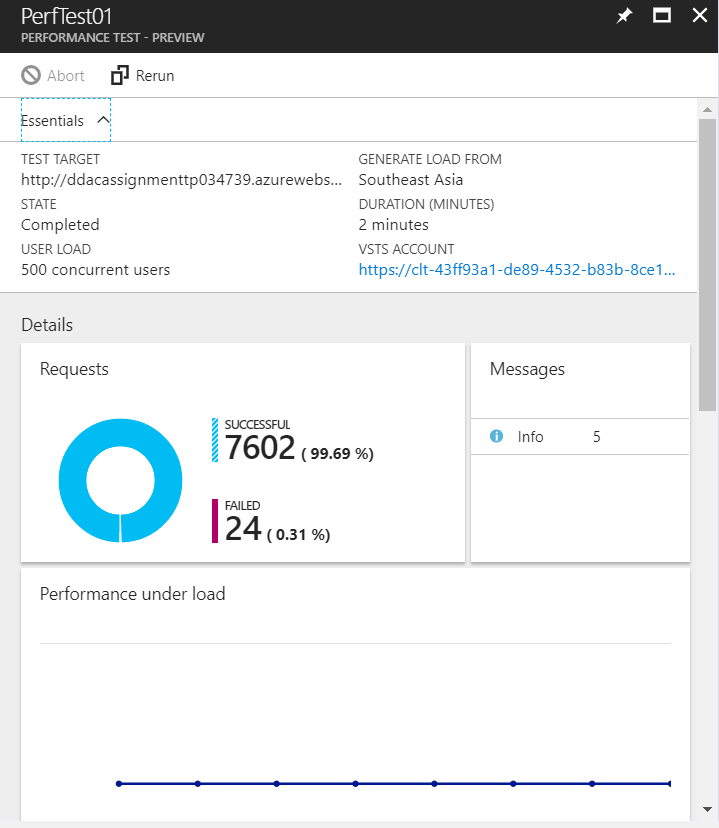
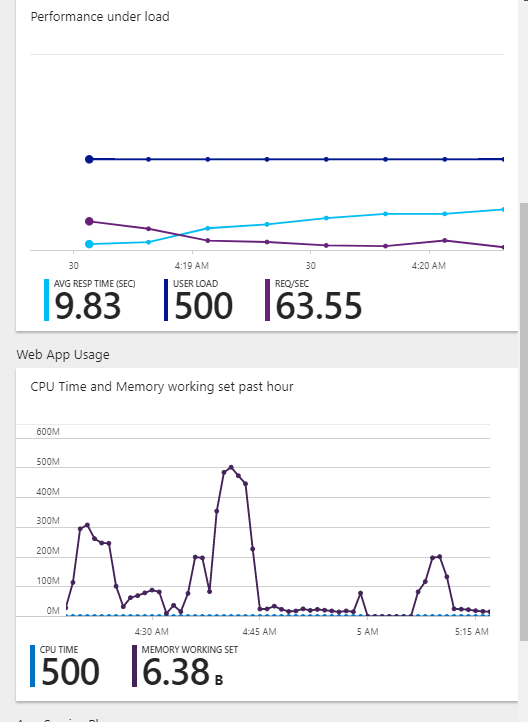
 

Figure 30: Sample Result 1 Figure 31: Sample Result 2

Table 1: Maersk Line CMS performance testing with different plan

|  |  |  |  |
| --- | --- | --- | --- |
| Concurrent Users  App  Service Plan | 500 | 750 | 1000 |
| B1 | 9.83 Sec  24 Failed | 14.98 Sec  11 Failed | 19.35 Sec  427 Failed |
| B2 | 2.33 Sec  2 Failed | 3.55 Sec  0 Failed | 3.61 Sec  25 Failed |
| S1 | 19.33 Sec  0 Failed | 16.6 Sec  0 Failed | 20.12 Sec  0 Failed |

### 5.2.1 Analysis

The conclusion is B2 plan is the most suitable plan for Maersk Line CMS. Although it has some failure in 1000 concurrent user, however it has the fastest speed compare to B1 and S1. B2 plan actually is quite expensive compare to B1 and S1 due to it has higher RAM and 2 Core. The qualification that to become a suitable plan is depend on the speed and the low failure rate. Compare the three plans, B2 is the best valued tier and the price is still affordable under the budget restriction which is RM 468.72 per month

# 6.0 Managed Databases

According to (Microsoft, 2018), Azure SQL Database is a SQL database that native to the cloud. It also known as platform as a service (PaaS) database or a database as a service (DBaaS). It is optimized for software as a service (SaaS) application development. Besides, it also offers compatibility with the SQL Server features (Microsoft, 2018). There are few advantages of PaaS like cut coding time, add development capabilities without hiring additional staff, and able to manage the application lifecycle efficiently.

First of all, PaaS development tools can cut the time in coding the new application with pre-coded application components that have built into a platform (Microsoft, 2018). It really saves a lot of time a cost that can prevent waste resources on unnecessary place. Besides, PaaS also can give the development team a lot of new capabilities without hiring the additional staff that need required skill (Microsoft, 2018). Moreover, PaaS provides all capability that needed to build a web application (Microsoft, 2018). For example, testing, deploying and managing in a same integrated environment.

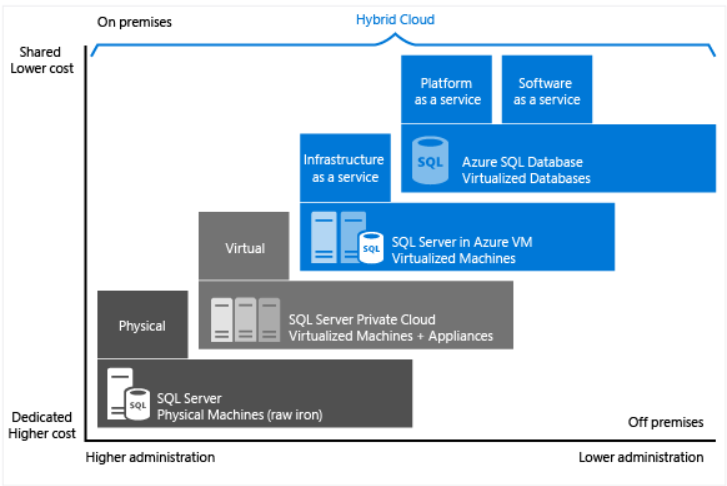


Figure 32: Characteristic of level of administration (Microsoft, 2018)

Figure 32 stated that the managed database could cost a less and less administration. It really reduces a lot operation cost if using PaaS as the database. Therefore, it is an excellent choice for the company that wanted to grow their business but lack of human resource to handle it. The developer can use the built-in feature to reduce the overall cost spent for provisioning and managing the databases (Microsoft, 2018). There is another benefit that use PaaS which is it does not need to manage the upgrades or high availability or back up (Microsoft, 2018). Azure SQL database can handle all of it.

In addition, there are some business motivation or factor that can cause a company to choose Azure SQL Database. For example, Cost, Administration, Service Level Agreement and Time to market.

First of all, Azure SQL database has supported few service tiers which does not cost too much (Microsoft, 2018). For example, Basic plan, Standard Plan and Premium Plan. Each plan has provided different level of performance with the cost that covered to handle the internet traffic etc. It also will be configured automatically, upgraded and patched by Microsoft. Moreover, Administration is another factor that a company will consider Azure SQL database. The reason is the business can be still monitor the database without need to manage the database engine, server, hardware and operating system (Microsoft, 2018). Furthermore, Service Level Agreement (SLA) provides an availability of 99.99% that can cover the IT department which help to support the business plan. Last but not least, it can achieve high productivity of the developer and fast time-to-market. For example, it provides PowerShell Cmdlets that can manage the administrative operation for databases with automated (Microsoft, 2018).

# 7.0 Conclusion

In the nutshell, the development of the cloud-based application for Maersk Line CMS has been run successfully. The developer has gained a lot of knowledge and experience on cloud computing and how Microsoft Azure connection with web application by publishing the Maersk Line CMS. Last but not least, it is a strong foundation for the developer that can strengthen their skill that may benefit their future career, especially the development of cloud application is getting hot and in trend nowadays.

# 8.0 References

1. Azure, M., 2017. *Autoscaling.* [Online]   
   Available at: https://docs.microsoft.com/en-us/azure/architecture/best-practices/auto-scaling  
   [Accessed 13 April 2018].
2. Microsoft, 2018. *Azure Database for MySQL.* [Online]   
   Available at: https://azure.microsoft.com/en-us/services/mysql/  
   [Accessed 13 April 2018].
3. Microsoft, 2018. *Choose a cloud SQL Server option: Azure SQL (PaaS) Database or SQL Server on Azure VMs (IaaS).* [Online]   
   [Accessed april 2018].
4. Microsoft, 2018. *What is Paas?.* [Online]   
   Available at: https://azure.microsoft.com/en-us/overview/what-is-paas/  
   [Accessed 13 April 2018].
5. tutorialspoint, 2018. *Design Patterns - MVC Pattern.* [Online]   
   Available at: https://www.tutorialspoint.com/design\_pattern/mvc\_pattern.htm  
   [Accessed 13 April 2018].