

|  |  |  |
| --- | --- | --- |
| **Module Code** | **:** | CT071-3-3-DDAC |
| **Student Name** | **:** | Woo Quan Feng |
| **TP Number** | **:** | TP037692 |
| **Intake Code** | **:** | UC3F1706SE |
| **Lecturer Name** | **:** | **Dr. Kalai Anand Ratnam** |
| **Assignment Title** | **:** | Container Management System (CMS) with Microsoft MVC and Azure |
| **Due Date** | **:** | 13 April 2018 |

# Acknowledgement

I want to thank the following:

**Dr. Kalai Anand Ratnam**, I can produce a quality system with Microsoft Azure Web Application and Azure SQL DB with the knowledge leaded by him. This is a very useful knowledge that will help me in the future.

Table of Contents

[Acknowledgement 2](#_Toc516323105)

[Introduction 5](#_Toc516323106)

[Project Background 5](#_Toc516323107)

[Project Scope 6](#_Toc516323108)

[Aims 6](#_Toc516323109)

[Objectives 6](#_Toc516323110)

[Deliverables 7](#_Toc516323111)

[Project Plan 8](#_Toc516323112)

[Design 9](#_Toc516323113)

[UML Use Case Diagram 9](#_Toc516323114)

[UML Use-Case Diagram Description 10](#_Toc516323115)

[Login 10](#_Toc516323116)

[Register 11](#_Toc516323117)

[Create Schedule 12](#_Toc516323118)

[Create Customer 13](#_Toc516323119)

[Create Vessel 14](#_Toc516323120)

[Create Item 15](#_Toc516323121)

[Manage Schedule 16](#_Toc516323122)

[Manage Customer 17](#_Toc516323123)

[Manage Vessel 18](#_Toc516323124)

[Manage Item 19](#_Toc516323125)

[Sequence Diagram 20](#_Toc516323126)

[Login 20](#_Toc516323127)

[Register 21](#_Toc516323128)

[Create Schedule 22](#_Toc516323129)

[Create Customer 22](#_Toc516323130)

[Create Vessel 23](#_Toc516323131)

[Create Item 23](#_Toc516323132)

[Manage Schedule 24](#_Toc516323133)

[Manage Customer 25](#_Toc516323134)

[Manage Vessel 26](#_Toc516323135)

[Manage Item 27](#_Toc516323136)

[Activity Diagram 28](#_Toc516323137)

[Register 28](#_Toc516323138)

[Create Schedule 28](#_Toc516323139)

[Create Customer 29](#_Toc516323140)

[Create Vessel 29](#_Toc516323141)

[Create Item 30](#_Toc516323142)

[Manage Schedule 31](#_Toc516323143)

[Manage Customer 32](#_Toc516323144)

[Manage Vessel 33](#_Toc516323145)

[Manage Item 34](#_Toc516323146)

[Entity Relationship Diagram 35](#_Toc516323147)

[Cloud Design Pattern 36](#_Toc516323148)

[Compute Resource Consolidation Pattern 37](#_Toc516323149)

[Cloud Architecture 38](#_Toc516323150)

[Implementation 39](#_Toc516323151)

[Application Development 39](#_Toc516323152)

[Models 39](#_Toc516323153)

[Views 40](#_Toc516323154)

[Controllers 40](#_Toc516323155)

[Web Application 41](#_Toc516323156)

[Azure Publishing 42](#_Toc516323157)

[Create App Service 42](#_Toc516323158)

[Azure SQL Database 43](#_Toc516323159)

[Application Scaling 44](#_Toc516323160)

[Configure 45](#_Toc516323161)

[Performance and Availability 46](#_Toc516323162)

[Test Plan and Testing Discussion 47](#_Toc516323163)

[Functional: Unit Testing 47](#_Toc516323164)

[Functional Discussion 49](#_Toc516323165)

[Performance Test 50](#_Toc516323166)

[Performance Test Discussion 50](#_Toc516323167)

[Conclusion 51](#_Toc516323168)

[References 52](#_Toc516323169)

[Appendix 53](#_Toc516323170)

# Introduction

## Project Background

In this assignment question, Maersk Line required a container management system. This system need to handle the overall business operations of Maersk Line. Container Management System (CMS) is basically a web application which can operate and manage the schedules, customers, vessels, and items.

Container Management System that provides an online and user-friendly platform for the Maersk users. Users in this system can be divided into two groups, which are Admin and Agent. For Admin, admin can create schedule, manage schedules, and manage vessels. As for the Agent, agent can create vessel, create customer, create item, manage vessel, manage customer, and manage item.

Microsoft ASP.NET MVC is used to build for the Container Management System and the programming language are used is C#. To manage schedules, vessels, customers, and items in a main database are consider as a system functions in CMS. Also, CMS is assumed that will ultimately improve the efficiency in collecting, securing, and distributing the important data among the users. At last, Maersk having a Container Management System (CMS) will significantly shorten the time of the schedule management and enhance the interaction between users, Admin and Agent.

## Project Scope

The system is a web application that includes well IT technology to interact with the users. The users are divided into two categories, which are agent and admin. Agent and admin must to register as system user before using the system. This system helps in collecting useful and important data which is required for Maersk planning and simplifying the vessel managing processes.

## Aims

The aims are listed below:

1. To allow better and easier interaction between agent and admin.

2. To save time.

3. To simplifying the managing process.

4. To utilize resources in efficient manner.

## Objectives

The main objective of the Container Management System (CMS) is to manage the details of schedules, customers, items, and vessels with create, edit, and delete function. To using these functions can increase the accuracy of the data and keeping the data secured. It can improve the efficiency of managing schedules, customers, items, and vessels.

## Deliverables

Container Management System is an online web application which allows the users to access the system while users are connected to internet. This system can be accessed through multiple browsers. Below are the followings being the system functions taken into consideration:

For System Users:

1. Register as admin or agent
2. Login to the system

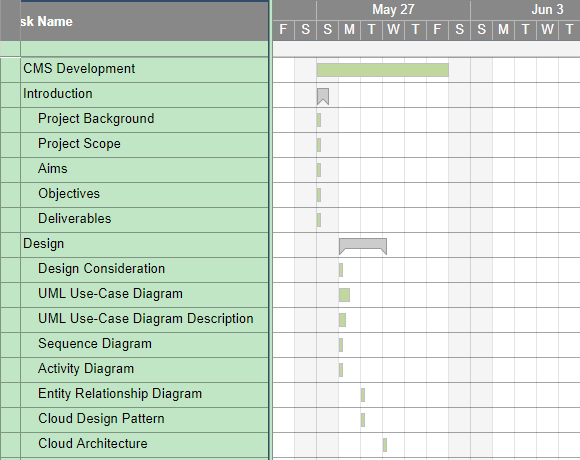
For Admin:

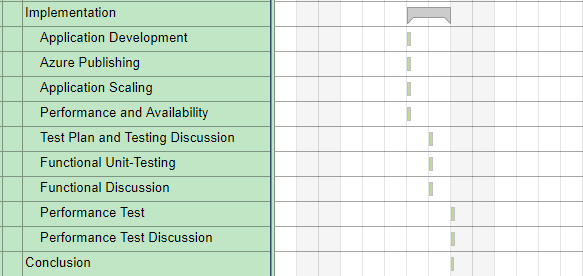
1. Manage and create schedule

For Agent:

1. Manage and create customer
2. Manage and create customer’s items
3. Manage and create customer’s vessels

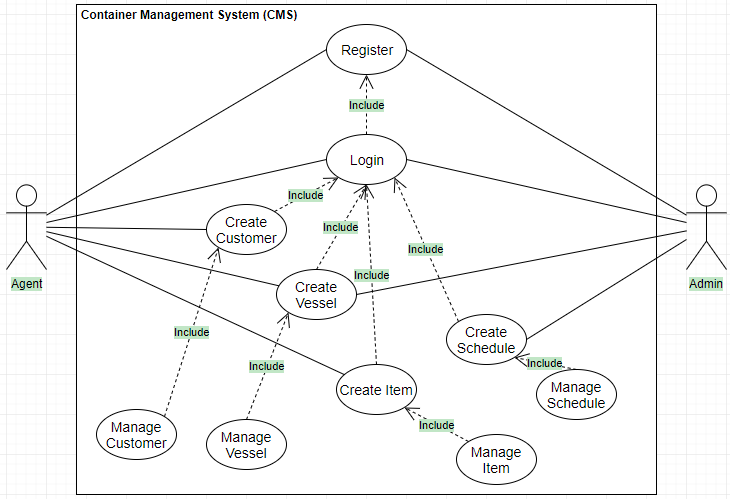
## Project Plan





# Design

## UML Use Case Diagram



## UML Use-Case Diagram Description

### Login

|  |  |
| --- | --- |
| **Use Case** | Login |
| **Summary** | Users log on the system to carry out actions. |
| **Dependency** | - |
| **Actors** | Admin, Agent |
| **Preconditions** | User required to register a valid account. |
| **Descriptive of Main Sequence** | 1. Users input their username and password.  2. System will check for the credentials of the account.  3. Login is successful if the credentials are correct.  4. System will display respective user profile page. |
| **Descriptive of Alternative Sequence** | 1. If login is unsuccessful, an error message is displayed to the users. |
| **Postcondition** | Users has successfully logged onto the system and able to perform actions. |

|  |  |
| --- | --- |
| **Use Case** | Register |
| **Summary** | Users must register before login to the system. |
| **Dependency** | - |
| **Actors** | Admin, Agent |
| **Preconditions** | - |
| **Descriptive of Main Sequence** | 1. Users input their preferred username and password.  2. System verifies the username if it has already existed.  3. System verifies if all the required fields are entered.  4. System will register the user information into the database. |
| **Descriptive of Alternative Sequence** | 1. If login is unsuccessful, an error message is displayed to the users. |
| **Postcondition** | Users has successfully logged onto the system and able to perform actions. |

### Register

|  |  |
| --- | --- |
| **Use Case** | Create Schedule |
| **Summary** | Admin creates a new schedule for vessels. |
| **Dependency** | <<Include>> Login |
| **Actors** | Admin |
| **Preconditions** | Admin is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Admin click the “Create Schedule” button.  2. System display a new creation form.  3. Admin enter the required details.  4. Admin click the “Create” button to build the schedule.  5. After that, system will publish the schedule. |
| **Descriptive of Alternative Sequence** | 1. Admin fill in the wrong information will display the error message. |
| **Postcondition** | A new schedule is created. |

### Create Schedule

### Create Customer

|  |  |
| --- | --- |
| **Use Case** | Create Customer |
| **Summary** | Agent create a new customer. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Agent click the “Create Customer” button.  2. System display a new creation form.  3. Agent enter the required details.  4. Agent click the “Create” button to build the new customer.  5. After that, system will record the customer’s details. |
| **Descriptive of Alternative Sequence** | 1. Agent fill in the wrong information will display the error message. |
| **Postcondition** | A new customer is created. |

|  |  |
| --- | --- |
| **Use Case** | Create Vessel |
| **Summary** | Agent create a new vessel for customer. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Agent click the “Create Vessel” button.  2. System display a new creation form.  3. Agent enter the required details.  4. Agent click the “Create” button to build the new vessel for customer.  5. After that, system will publish the new vessel. |
| **Descriptive of Alternative Sequence** | 1. Agent fill in the wrong information will display the error message. |
| **Postcondition** | A new vessel is created. |

### Create Vessel

### Create Item

|  |  |
| --- | --- |
| **Use Case** | Create Item |
| **Summary** | Agent create a new item for customer. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Agent click the “Create Item” button.  2. System display a new creation form.  3. Agent enter the required details.  4. Agent click the “Create” button to build the new item.  5. After that, system will record the item’s details. |
| **Descriptive of Alternative Sequence** | 1. Agent fill in the wrong information will display the error message. |
| **Postcondition** | A new item is created. |

|  |  |
| --- | --- |
| **Use Case** | Manage Schedule |
| **Summary** | Admin able to manage schedule. |
| **Dependency** | <<Include>> Login |
| **Actors** | Admin |
| **Preconditions** | Admin is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Admin click the “Manage Schedule” button.  2. System display a page with existing schedule.  3. Admin can edit the existing schedule.  4. Admin can add a new schedule.  5. Admin can delete existing schedule.  6. After that, system will update the new data. |
| **Descriptive of Alternative Sequence** | 1. There is error in customer details. |
| **Postcondition** | The schedule is created, updated, or deleted. |

### Manage Schedule

### Manage Customer

|  |  |
| --- | --- |
| **Use Case** | Manage Customer |
| **Summary** | Agent able to manage schedule. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Agent click the “Manage Customer” button.  2. System display a page with existing customer.  3. Agent can edit the existing customer.  4. Agent can add a new customer.  5. Agent can delete existing customer.  6. After that, system will update the new data. |
| **Descriptive of Alternative Sequence** | 1. There is error in customer details. |
| **Postcondition** | The customer is created, updated, or deleted. |

### Manage Vessel

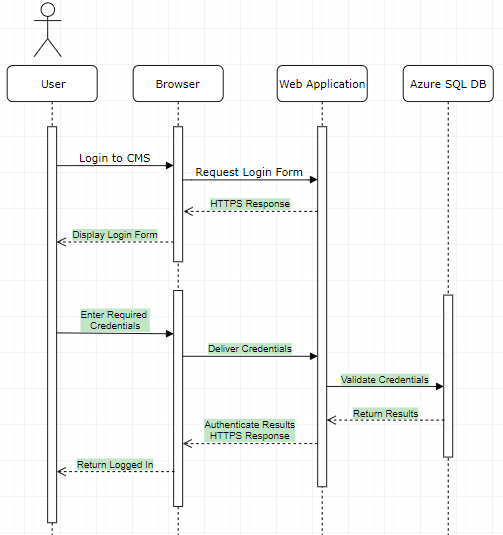
|  |  |
| --- | --- |
| **Use Case** | Manage Vessel |
| **Summary** | Agent able to manage vessel. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Agent click the “Manage Vessel” button.  2. System display a page with existing vessel.  3. Agent can edit the existing vessel.  4. Agent can add a new vessel.  5. Agent can delete existing vessel.  6. After that, system will update the new data. |
| **Descriptive of Alternative Sequence** | 1. There is error in vessel details. |
| **Postcondition** | The vessel is created, updated, or deleted. |

### Manage Item

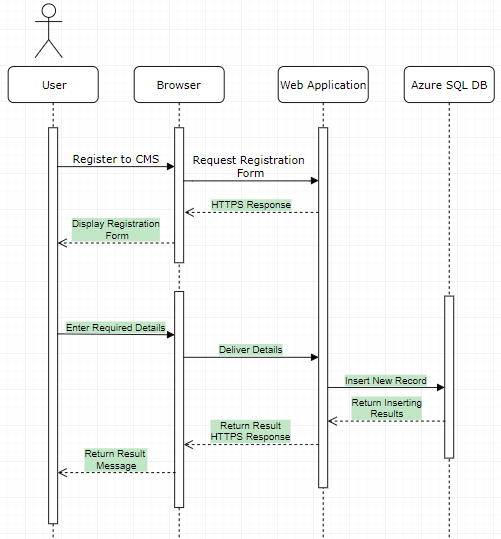
|  |  |
| --- | --- |
| **Use Case** | Manage Item |
| **Summary** | Agent able to manage item. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged onto the system. |
| **Descriptive of Main Sequence** | 1. Agent click the “Manage Item” button.  2. System display a page with existing item.  3. Agent can edit the existing item.  4. Agent can add a new item.  5. Agent can delete existing item.  6. After that, system will update the new data. |
| **Descriptive of Alternative Sequence** | 1. There is error in item details. |
| **Postcondition** | The item is created, updated, or deleted. |

## Sequence Diagram

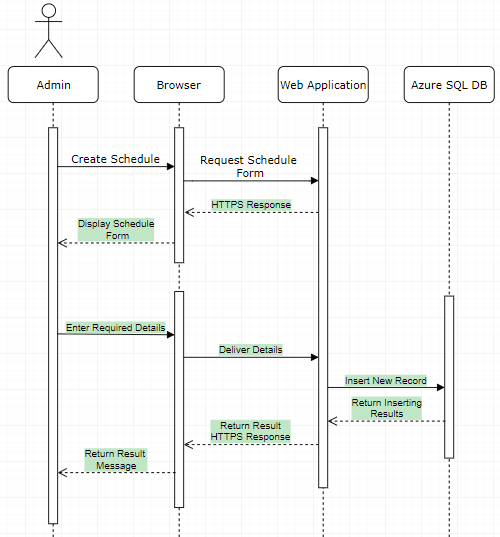
### Login



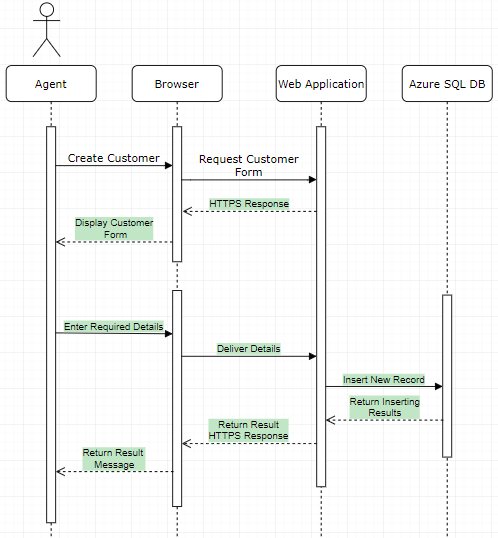
### Register



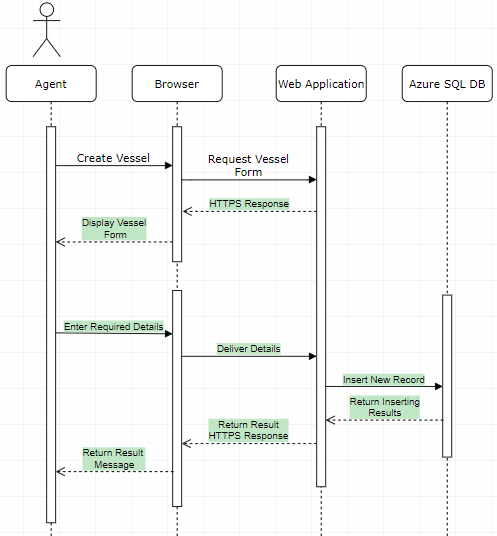
### Create Schedule



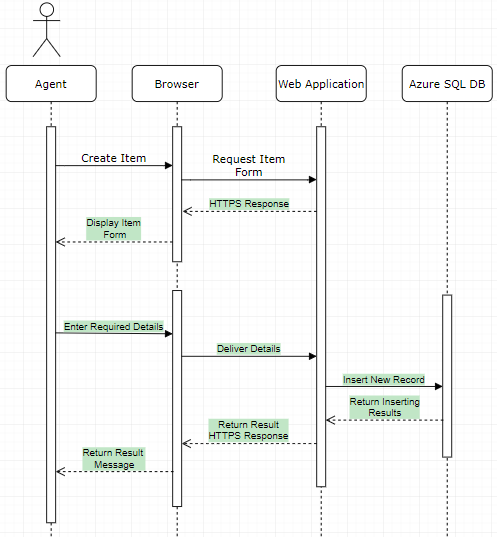
### Create Customer



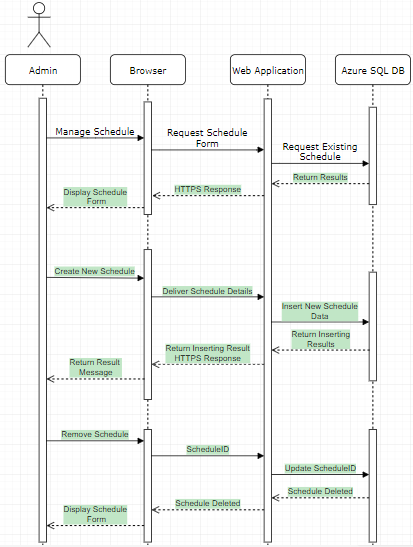
### Create Vessel



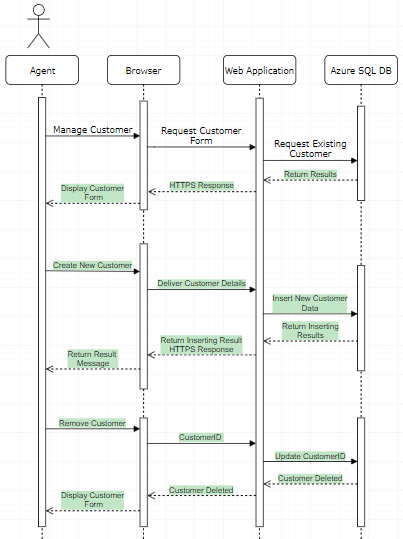
### Create Item



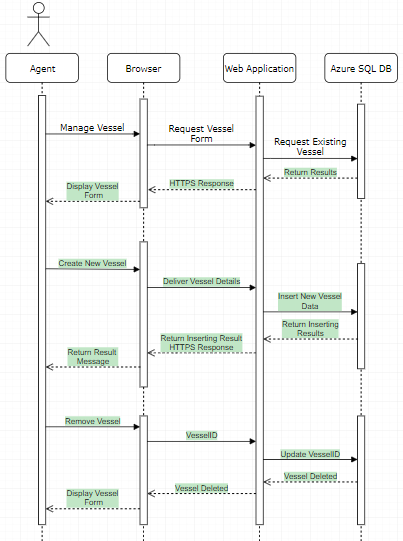
### Manage Schedule



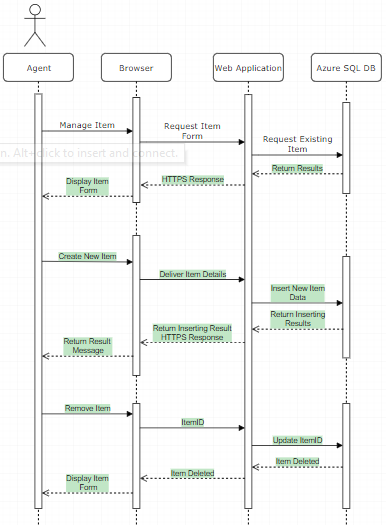
### Manage Customer



### Manage Vessel

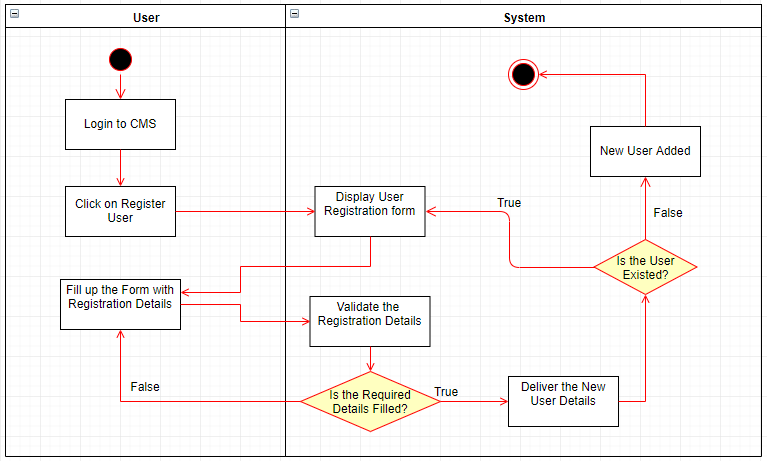


### Manage Item

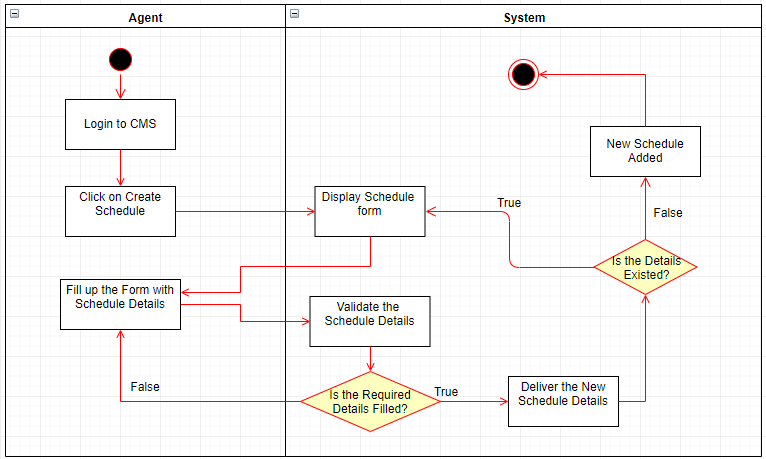


## Activity Diagram

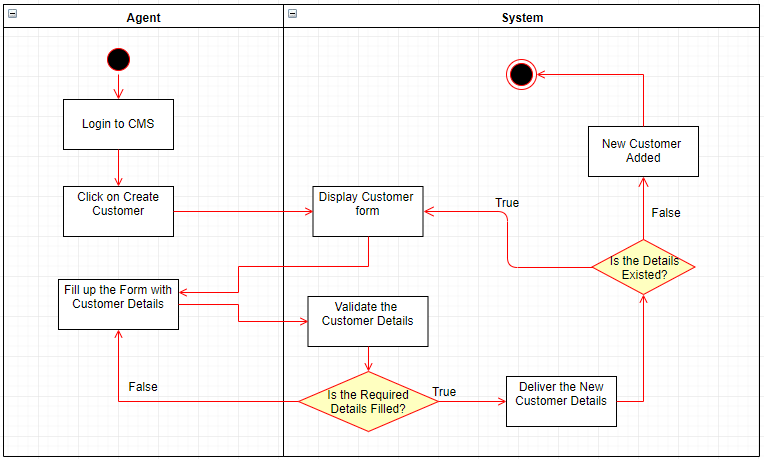
### Register



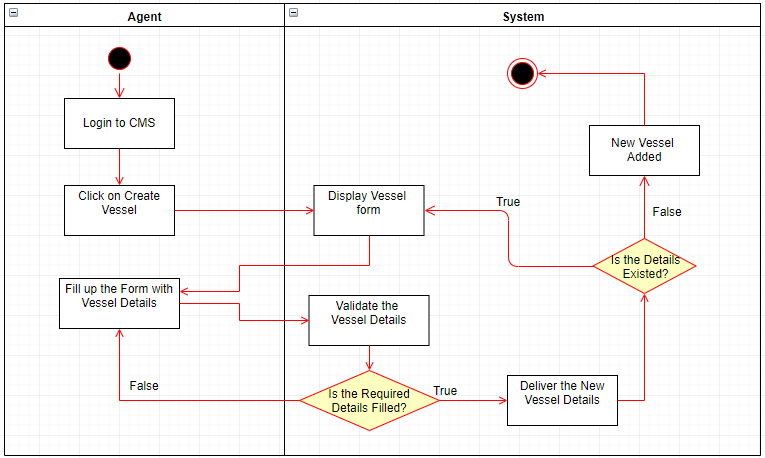
### Create Schedule



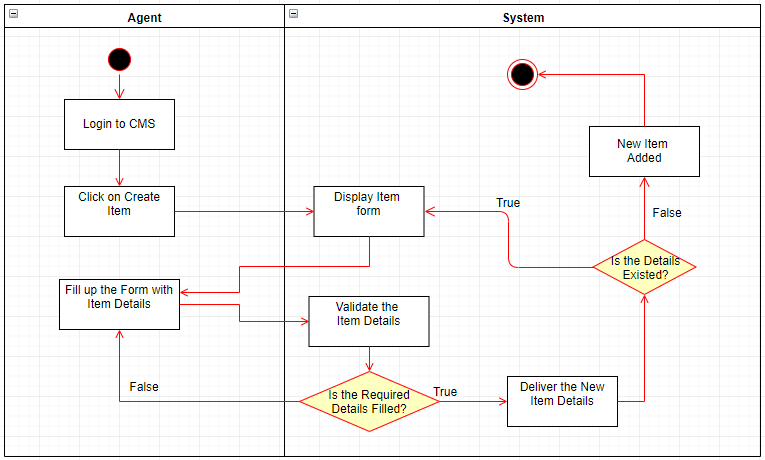
### Create Customer



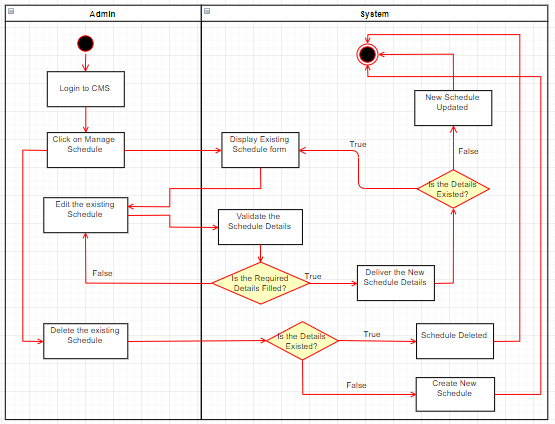
### Create Vessel



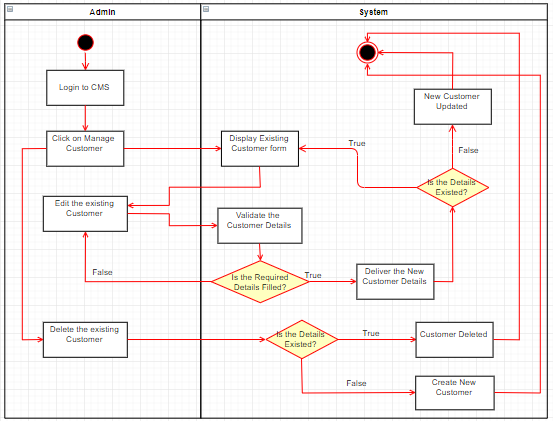
### Create Item



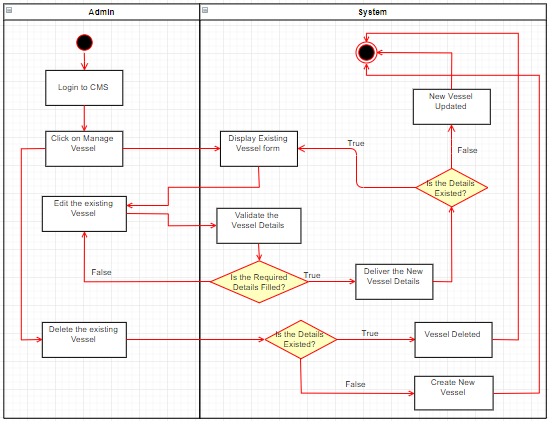
### Manage Schedule



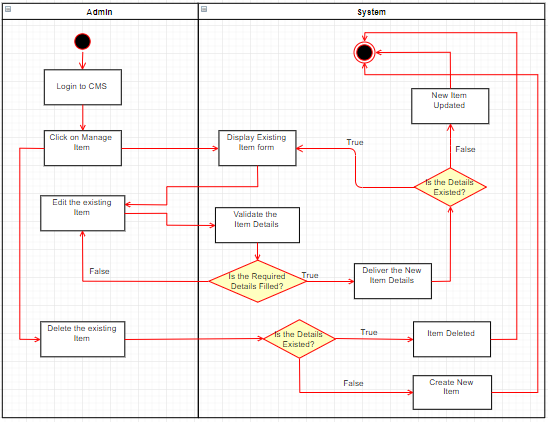
### Manage Customer



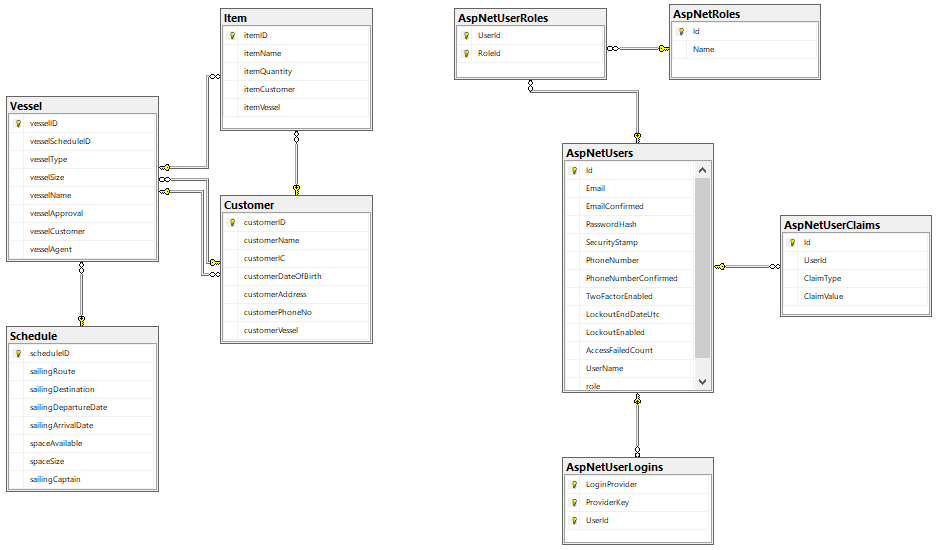
### Manage Vessel



### Manage Item



## Entity Relationship Diagram

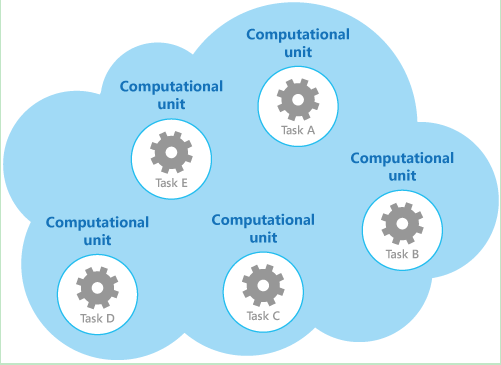


## Cloud Design Pattern

This system is implemented one of the cloud design patterns. There are a lot of patterns in cloud design, each pattern describes the problem that the pattern addresses. Many design patterns are included code snippets that guide how to implement the design patterns on Azure. For example, ambassador patterns are created a service to send the network requests on behalf of a consumer application.

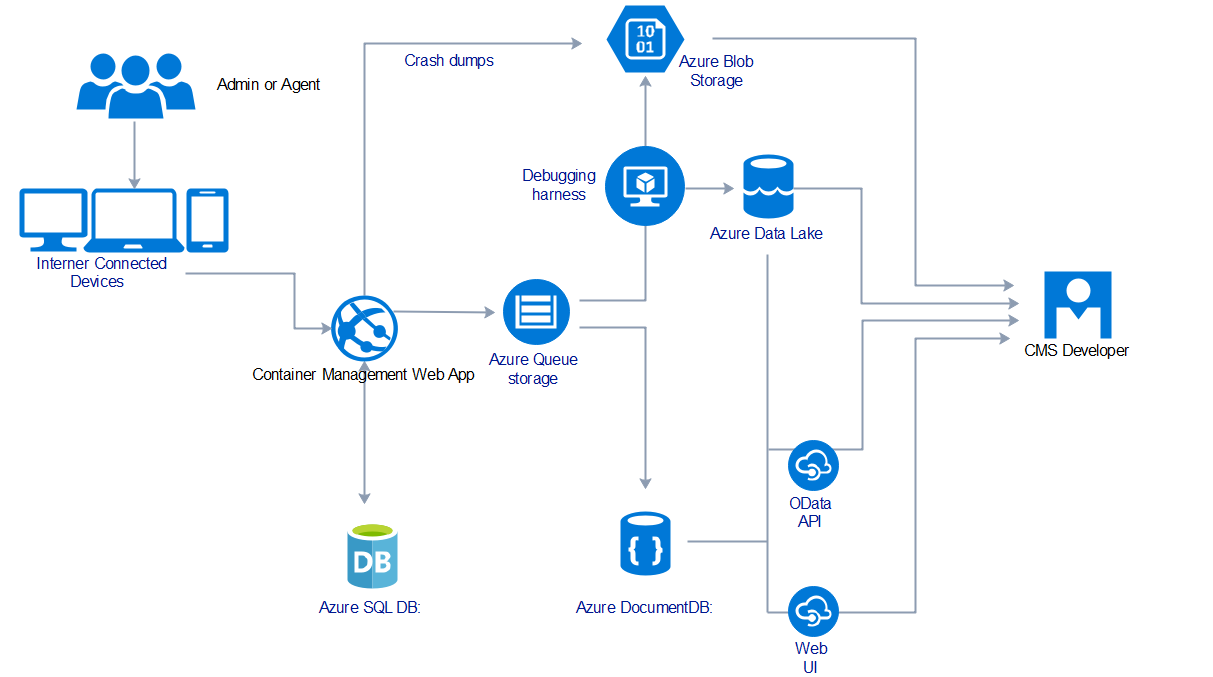
## Compute Resource Consolidation Pattern

A single computational unit manage multiple tasks can increase utilization, reduce costs and time, improve efficiency of the management. Depending on the functions provided by the environment and the costs associated with these functions, tasks can be grouped according to criteria. Grouping those tasks together to allows to scale as a unit. It enables additional instances of a computational unit to be started and stopped according to the workload.



## Cloud Architecture

In this system, admin or agent must connected with internet access devices in order to use the Container Management System. The container management web application is hosted by Microsoft Azure. And the data will be stored in the Azure SQL Database. For the better performance, the web application is subscribed to South East Asia region. So, when the users are login to the system, it will trigger the database for the data retrieving. The Azure Blob Storage will store any error or problem if the web application faced. The system developer will receive all the log records and crash reports.

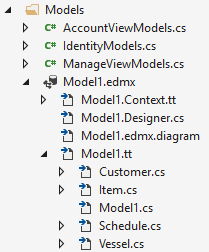


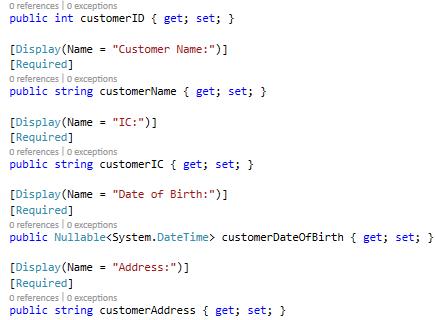
# Implementation

## Application Development

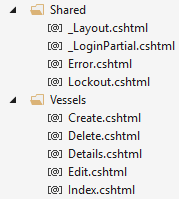
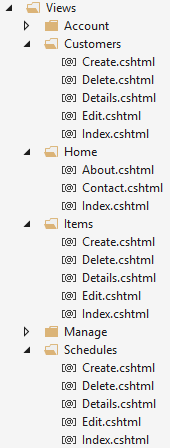
The development of the CMS web application is using Microsoft ASP.NET MVC with C# Programming Language. Model, View and Controller structures perform the web application to work. And all the website pages are controlled by the unique controller. View is the interface of system that users used. Model is defined the attributes stored in the Azure SQL DB and carry out the validation checking when users entering data.

### Models

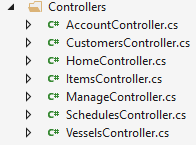




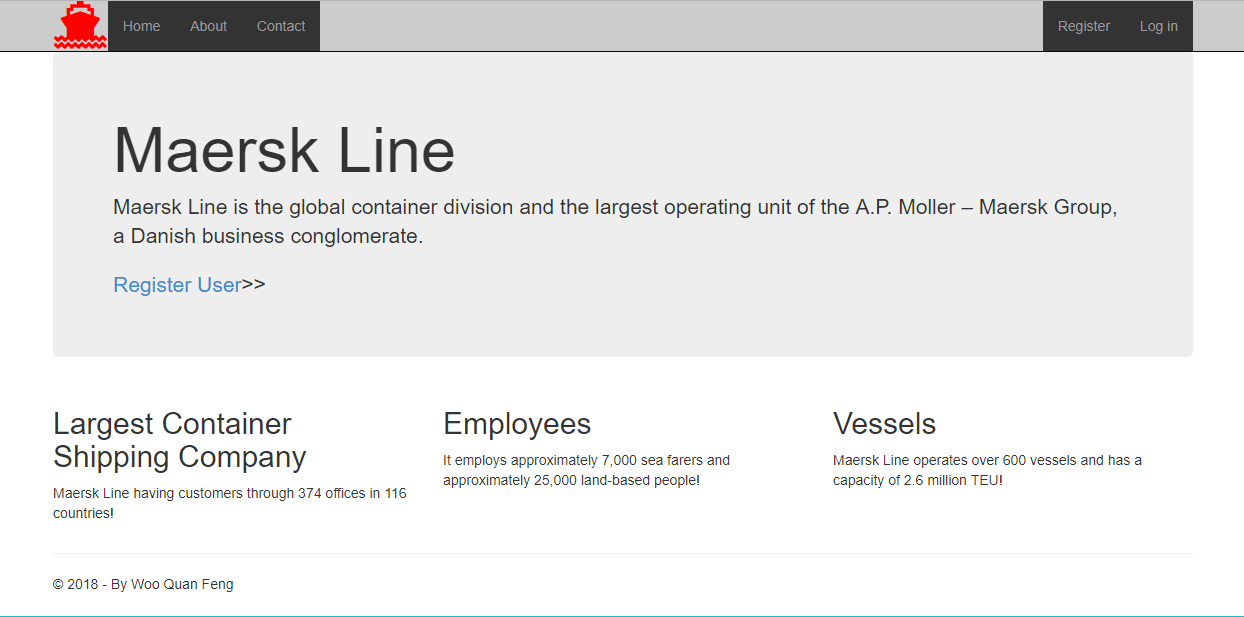
### Views



### Controllers

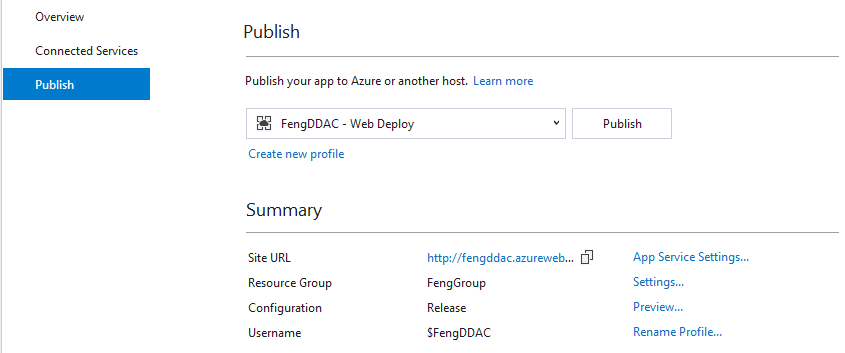


## Web Application

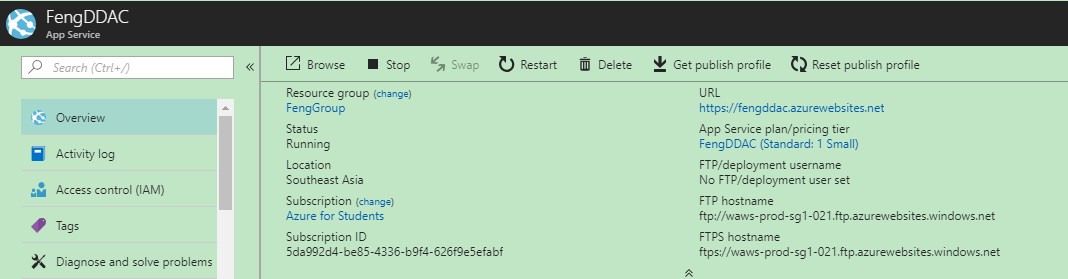


This diagram is above indicating the home page pf the Container Management Web Application. And there is a logo shown on the top left corner. For further understanding about this web application, can through <http://fengddac.azurewebsites.net>.

## Azure Publishing

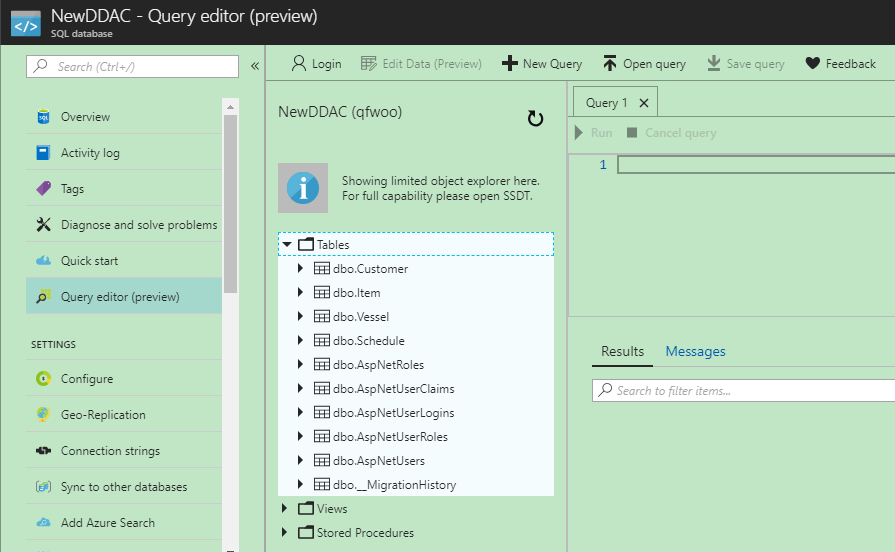


## Create App Service



The web application is published to Microsoft Azure using Visual Studio 2017 built-in extension. But first, developer need to create app service in Microsoft Azure. All the required Azure Service details needed to be filled in such as Application Name, Subscription, Resource Group and Hosting Plan.

## Azure SQL Database

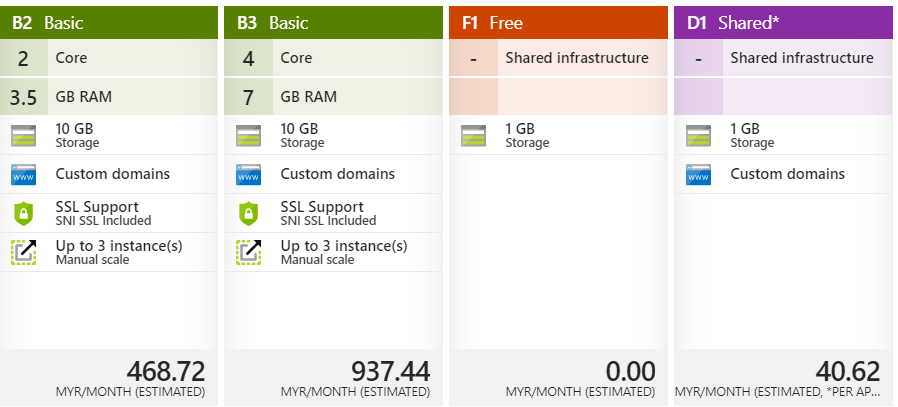


Azure SQL DB is used as the database for the web application. The developer has created all the tables required by logging to the Server Admin Account. And, execute a list of queries to the Azure SQL DB.

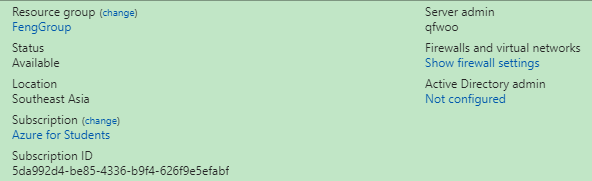


The connection string for the database is referred and stored under Web.config file of the project solution. The server name, initial catalogue, user id and password must be correct and connect to the Azure SQL DB. So, the connection of the project to the Azure SQL DB is successful.

## Application Scaling

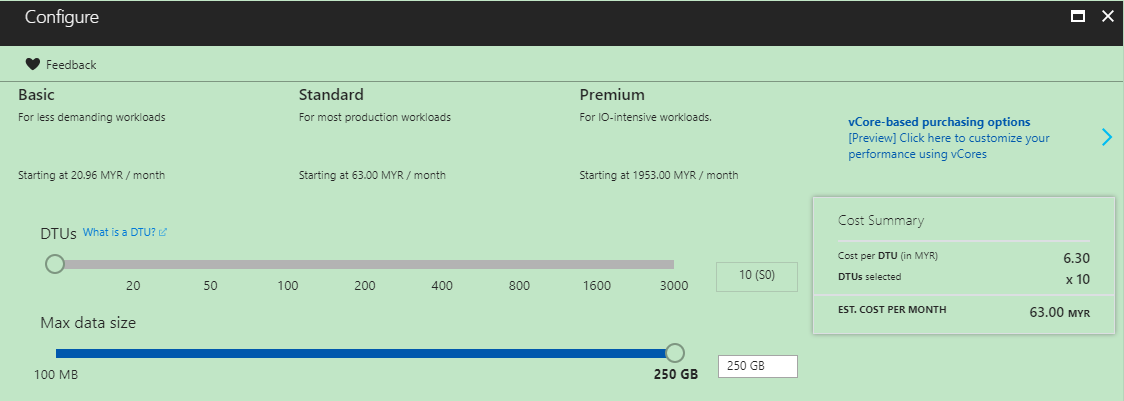


With the standards set by Microsoft Azure, all web applications other than Student Subscription must be subscribed to a basic, B1 plan. This basic plan is the standard of most of the personal development project with a small variety of developers. South East Asia (SEA) Region is the domain of this web application, the reason for choosing this region because this web application is developed in the Malaysia which belong to South East Asia (SEA) region. If choosing other region such as United State region will significantly slow down the performance of this web application, the communication network is far.



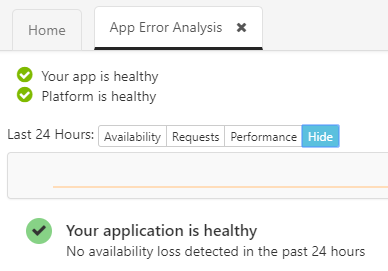
The current subscription of this web application is B1, Basic, at which provides one core with 1.75GB Ram.

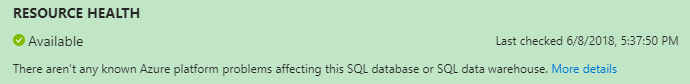
## Configure



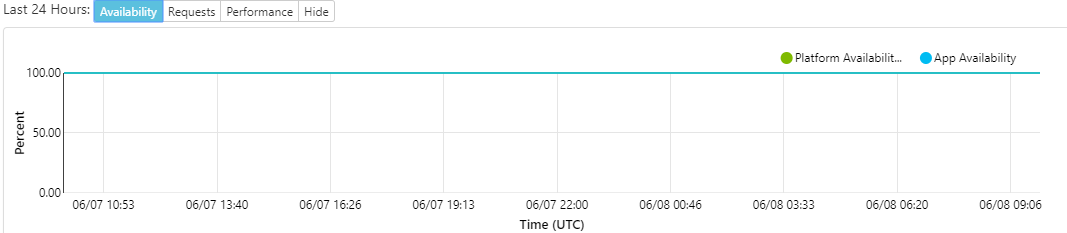
The Azure SQL DB will be using a Standard Tier of Pricing with 10 compute units and maximum data size of 250GB with RM63.00 per month.

## Performance and Availability





The above chart is generated from the Diagnosis Tools provided by Microsoft Azure. Diagnosis tools can analyse the performance and health of this web application. It comes with several tabs which are Availability, Requests and Performance. When there are no users accessing the web application, will stays at initial state. From this analysis, the developer can know the web application will not consume excess usage when it is free.



This chart above shows the availability of the web application. And shows that this web application is remain at 100% all the time which indicates the system runs all the time without going down or offline status.

# Test Plan and Testing Discussion

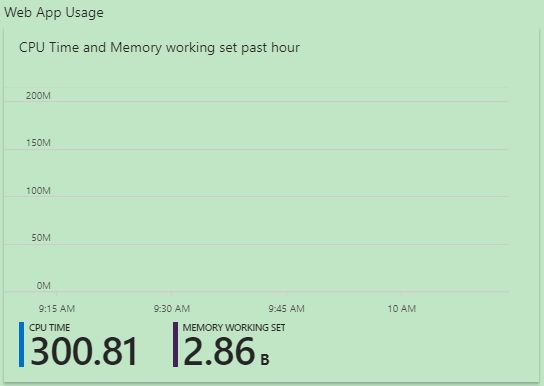
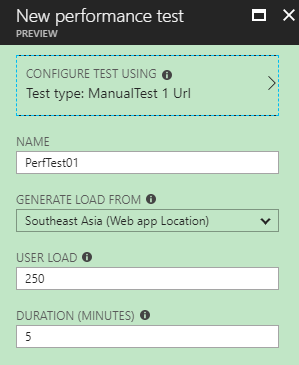
## Functional: Unit Testing

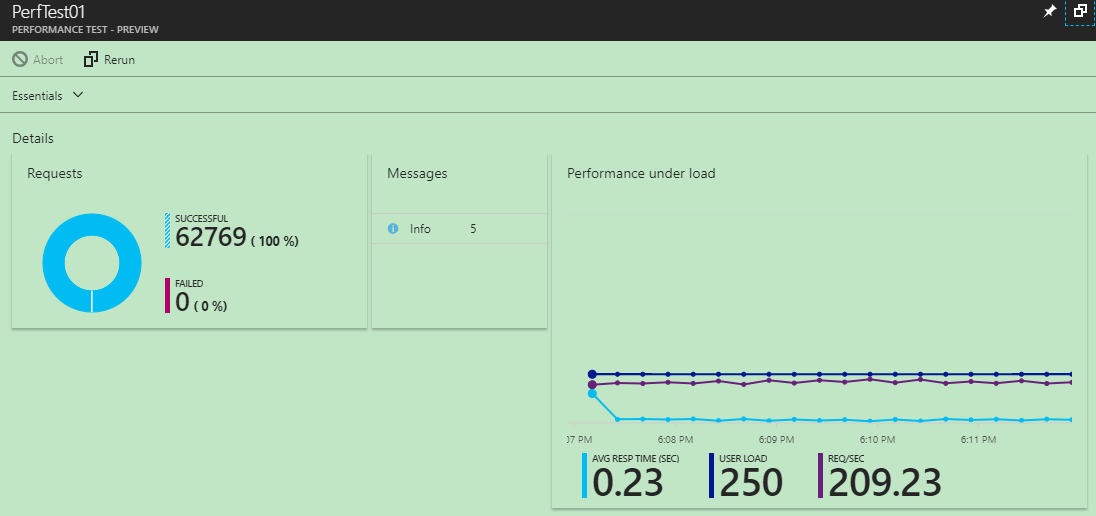
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Testing ID** | **Testing Function** | **Testing Description** | **Expected Results** | **Actual Results** | **Status** |
| 1 | UT1 | Register as Admin or Agent | Enter the required information for registering a new account to CMS. | New Admin or Agent is registered to CMS. | New Admin or Agent is registered to CMS. | Success |
| 2 | UT2 | Login to CMS | Enter the user’s email and password in the specific field and click “Login” button. | User’s account is logged into the system. | User’s account is logged into the system. | Success |
| 3 | UT3 | Create Schedule | Click on “Create Schedule” button and Enter the required details. | New Schedule is added. | New Schedule is added. | Success |
| 4 | UT4 | Create Customer | Click on “Create Customer” button and Enter the required details. | New Customer is added. | New Customer is added. | Success |
| 5 | UT5 | Create Vessel | Click on “Create Vessel” button and Enter the required details. | New Vessel is added. | New Vessel is added. | Success |
| 6 | UT6 | Create Item | Click on “Create Item” button and Enter the required details. | New Item is added. | New Item is added. | Success |
| 7 | UT7 | Manage Schedule | Click on “Manage Schedule” and Display all the existing schedules. | Existing schedules are displayed. | Existing schedules are displayed. | Success |
| 8 | UT8 | Manage Customer | Click on “Manage Customer” and Display all the existing customers. | Existing customers are displayed. | Existing customers are displayed. | Success |
| 9 | UT9 | Manage Vessel | Click on “Manage Vessel” and Display all the existing vessels. | Existing vessels are displayed. | Existing vessels are displayed. | Success |
| 10 | UT10 | Manage Item | Click on “Manage Item” and Display all the existing items. | Existing items are displayed. | Existing items are displayed. | Success |
| 11 | UT11 | Edit Schedule | Click on “Manage Schedule” and click “Edit” to Edit the details. | Schedules updated. | Schedules updated. | Success |
| 12 | UT12 | Edit Customer | Click on “Manage Customer” and click “Edit” to Edit the details. | Customers updated. | Customers updated. | Success |
| 13 | UT13 | Edit Vessel | Click on “Manage Vessel” and click “Edit” to Edit the details. | Vessels updated. | Vessels updated. | Success |
| 14 | UT14 | Edit Item | Click on “Manage Item” and click “Edit” to Edit the details. | Items updated. | Items updated. | Success |
| 15 | UT15 | Delete Schedule | Click on “Manage Schedule” and click “Delete” to delete the schedules. | Schedules deleted. | Schedules deleted. | Success |
| 16 | UT16 | Delete Customer | Click on “Manage Customer” and click “Delete” to delete the customers. | Customers deleted. | Customers deleted. | Success |
| 17 | UT17 | Delete Vessel | Click on “Manage Vessel” and click “Delete” to delete the vessels. | Vessels deleted. | Vessels deleted. | Success |
| 18 | UT18 | Delete Item | Click on “Manage Item” and click “Delete” to delete the items. | Items deleted. | Items deleted. | Success |

### Functional Discussion

As a conclusion of system validation with the unit testing, all the unit functions of Container Management System (CMS) had been tested. The developer use unit testing to test the system functions are work well without error or defect. So that, if there are no error occurs in the unit testing which means that the system is ready to go through to publish. At last, unit testing ensures the system can execute and run in the specific target environment.

## Performance Test





### Performance Test Discussion

These diagrams above show the output of performance testing on different standard plans with different amount of user load in 5 minutes with 250 users increment. A very important point is the developer must consider the amount of concurrent user to access the website before choosing the subscription plan. In conclusion, Standard 2 and Standard 3 Plans can operate and manage up to 220 concurrent users, but Standard 1 Plan only can hold up maximum with 360 concurrent users.

# Conclusion

In this project, through building up this web application, I learned a lot of understanding for building a system with Microsoft Azure. For the platform chosen, is Microsoft ASP.NET MVC 5, it is a new technology that are used to develop a website recently. In the beginning, it is very difficult to using MVC 5 to build the web application. But after done a lot of research from the network and some practice, it was very helpful for me to build the system. I believed this knowledge and experience will be helpful in my future career.

# References

Hoffman, C., 2018. *What is Microsoft Azure, Anyway?.* [Online]   
Available at: https://www.howtogeek.com/337961/what-is-microsoft-azure/  
[Accessed 27 May 2018].

Microsoft, 2014. *ASP.NET MVC 5.* [Online]   
Available at: https://docs.microsoft.com/en-us/aspnet/mvc/mvc5  
[Accessed 2017 May 2018].

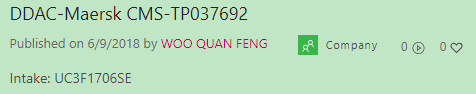
Sabnis, M., 2014. *What’s New in ASP.NET MVC 5 that make your MVC web sites shine.* [Online]   
Available at: http://www.dotnetcurry.com/aspnet-mvc/975/new-features-aspnet-mvc-5  
[Accessed 27 May 2018].

Times, T. E., 2018. *Definition of 'Unit Testing'.* [Online]   
Available at: https://economictimes.indiatimes.com/definition/unit-testing  
[Accessed 27 May 2018].

# Appendix

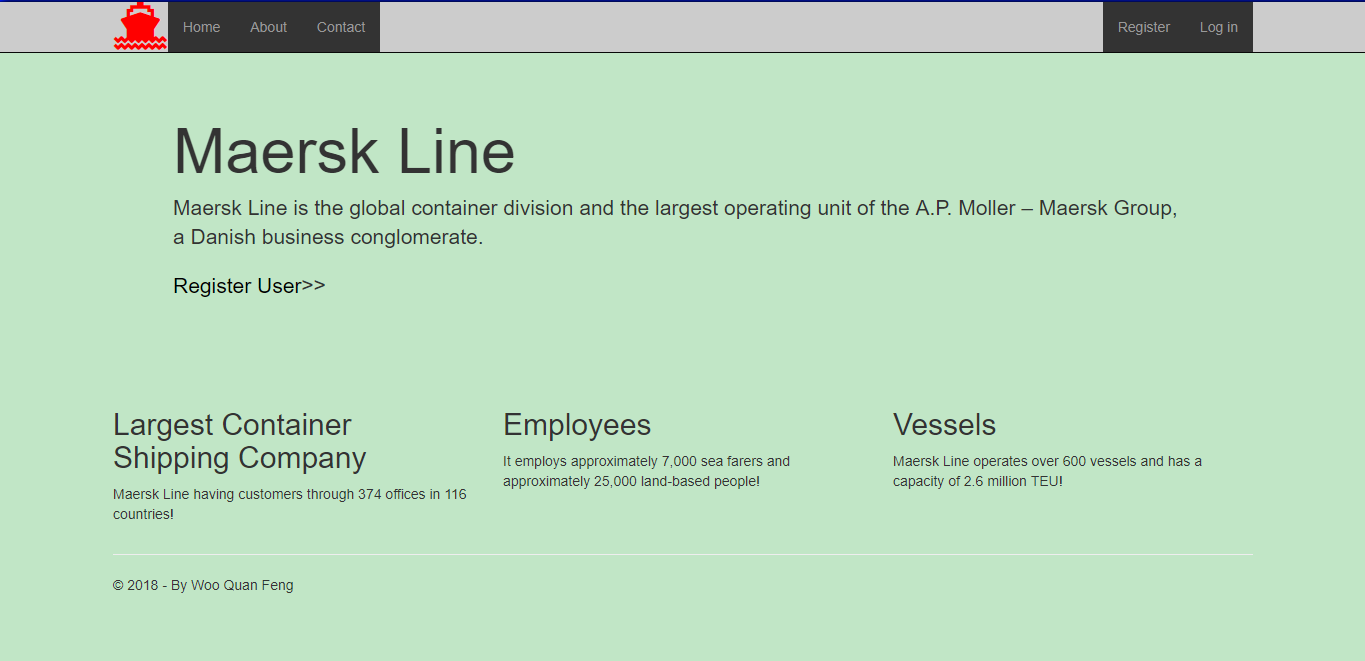
A) Microsoft Stream Link:

<https://web.microsoftstream.com/video/df0b682b-b0d4-414a-ba7f-d3ab526c879d?list=studio>



B) Container Management System Link:

<http://fengddac.azurewebsites.net>



C) GitHub Link:

<https://github.com/FengDDAC/QFWOO>