

|  |  |  |
| --- | --- | --- |
| **Module Code** | **:** | CT071-3-3-DDAC |
| **Student Name** | **:** | Chua Chin Hong |
| **TP Number** | **:** | TP034423 |
| **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 wish to thank the following:

My final year project supervisor, **Dr. Kalai Anand Ratnam** for his guidance, patience and advices for the system development. His undivided assistance is the key success to this assignment. With the knowledge leaded by him, I am able to produce a quality system and connect to the latest technology and services such as Microsoft Azure Web Application, SendGrid Email Services and Azure SQL Database.

Table of Contents

[Acknowledgement 1](#_Toc511399963)

[Introduction 3](#_Toc511399964)

[Project Background 3](#_Toc511399965)

[Project Scope 4](#_Toc511399966)

[Aims 4](#_Toc511399967)

[Objectives 4](#_Toc511399968)

[Deliverables 5](#_Toc511399969)

[Project Plan 6](#_Toc511399970)

[Design 9](#_Toc511399971)

[Design Consideration 9](#_Toc511399972)

[UML Use Case Diagram 9](#_Toc511399973)

[UML Use-Case Diagram Description 10](#_Toc511399974)

[Sequence Diagram 16](#_Toc511399975)

[Activity Diagrams 20](#_Toc511399976)

[Entity Relationship Diagram 23](#_Toc511399977)

[Cloud Design Pattern 24](#_Toc511399978)

[Cloud Architecture 26](#_Toc511399979)

[Implementation 27](#_Toc511399980)

[Application Development 27](#_Toc511399981)

[Email Services 31](#_Toc511399982)

[Azure Publishing 32](#_Toc511399983)

[Application Scaling 34](#_Toc511399984)

[Performance and Availability 36](#_Toc511399985)

[Test Plan & Testing Discussion 38](#_Toc511399986)

[Functional: Unit Testing 38](#_Toc511399987)

[Functional Discussion 41](#_Toc511399988)

[Performance Test 42](#_Toc511399989)

[Performance Testing Discussion 43](#_Toc511399990)

[Conclusion 44](#_Toc511399991)

[References 46](#_Toc511399992)

[Appendices 47](#_Toc511399993)

# Introduction

## Project Background

In the assignment context, a container management system is needed by the vendor, Maersk Line. This system is believed to handle the overall business operations of Maersk Group and able to carter to manage the supply chain costs.

The proposed system, “Container Management System (CMS)” is basically a web application which is able to manage the schedule, vessels and agents. As the world is becoming interconnected, this era of technology comprises a lot of software technologies that will resolve the operation with manpower and advancing the lifestyles of human beings. The proposed system will utilize the available technologies and resources in achieving the latest software environment trends.

CMS is a system that provides an online platform for the Maersk users who are anticipating manage schedule or become an agent. Hence, the users of this system are categorized into Admin and Agent. For Admin, the system will implement the functionalities such as register users, manage schedule, view agents and manage booking approval. As for the Agents, they are provided with a set of functionalities such as registering themselves to CMS, getting notification regarding booking approval, register customers, register customers’ items and make vessel booking.

Fundamentally, CMS is built using Microsoft ASP.NET MVC and it is based on C# programming language. There are several advantages of choosing this programming language including implement latest services and packages. Besides, CMS is considered as an automated management system which helps to manage vessel and agents’ information in a centralized database and assisting in solving the ordinary way of hardcopy document for managing vessels. It is assumed that using this system will ultimately improve the efficiency especially in collecting and distributing important data among the users. It also simplifies the registration of the users who wish to involve themselves in the system. Therefore, it is believed that having CMS will significantly shorten the time management for vessel schedules and enhance the interaction between admin and agents.

## Project Scope

The proposed system is considered a web application that includes useful IT technology to interact with the users. The users are mainly divided into two groups, the agent and the admin. Agent has to register to the system before using the system while admin need to proceed to the super admin for registration. The system helps in collecting important information which is needed for Maersk planning and simplifying the vessel managing processes.

## Aims

The aims for this project are listed below:

1. To allow easier interaction between agent and admin
2. To save time for creating a new schedule
3. To ease the schedule managing process
4. To allow users to be aware of the upcoming schedule
5. To utilize resources in efficient manner with the help of automation system

## Objectives

The main objective of Container Management System is to manage the details of schedule, agents and vessels. It should be able to increase the accuracy of the operation information and keeping the information secured. It can reduce the manual work leading to decreasing the workload of the Maersk staff. It also improves the efficiency of managing schedule. It combines a large diversity of vessels including a large cargo ship and normal ship.

## Deliverables

Container Management System is an online web application which allows the users to access at anytime, anywhere whenever they are connected to internet. The website of this system can be accessed through multiple browsers such as Google Chrome, Mozilla Firefox, Safari and so on. The followings are the system functions taken into consideration:

For System Users:

1. Register to the system to
2. Login to the system
3. Update Profile Picture
4. Change password and personal details

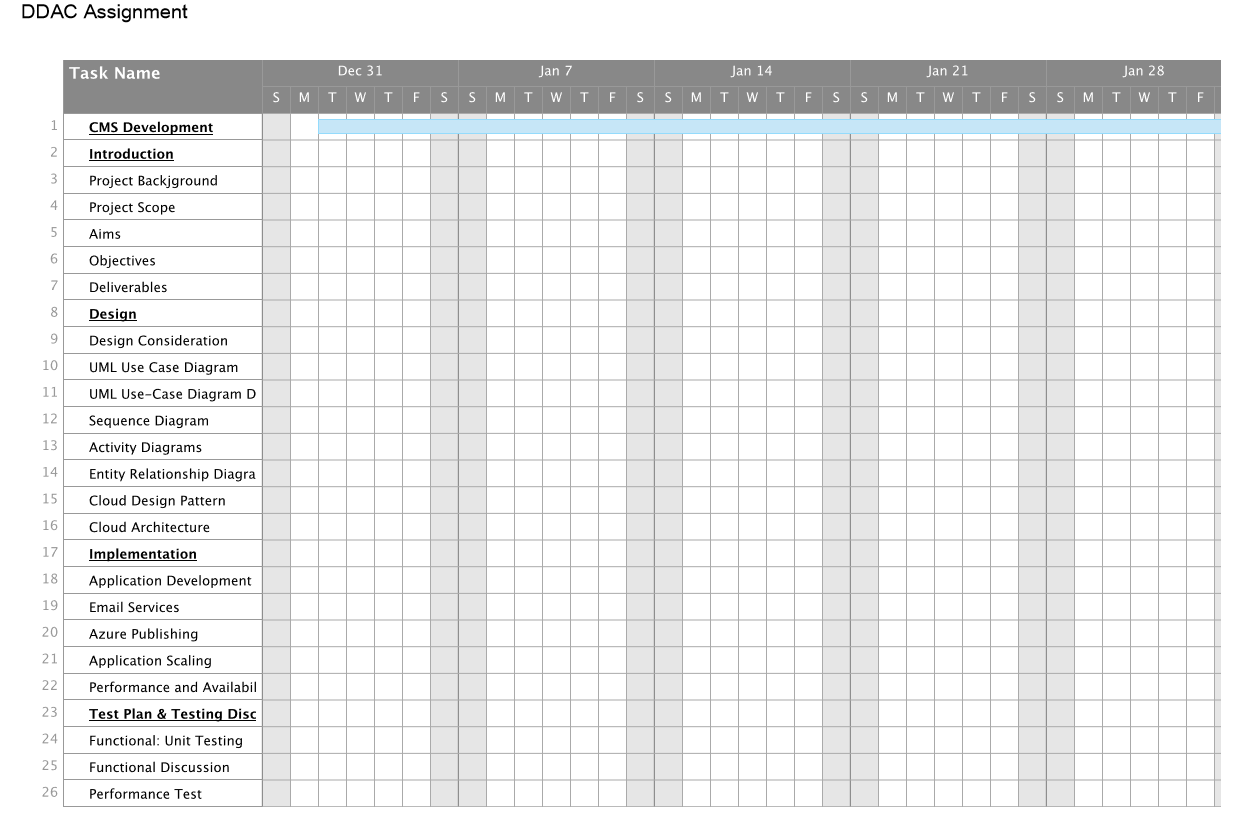
For Admin:

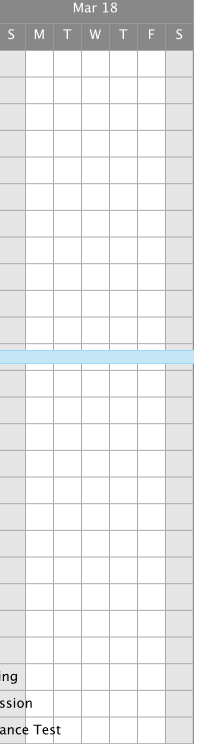
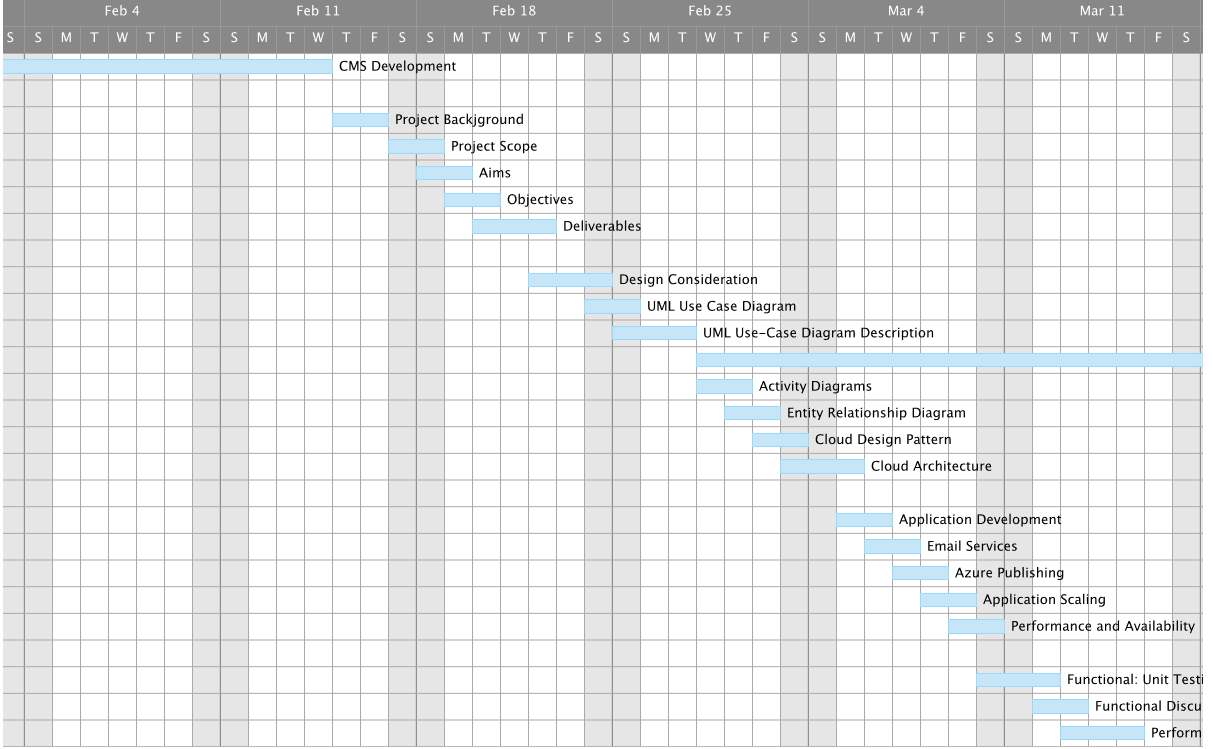
1. Manage and create schedule
2. Register all users
3. View existing agents
4. Approve or decline booking

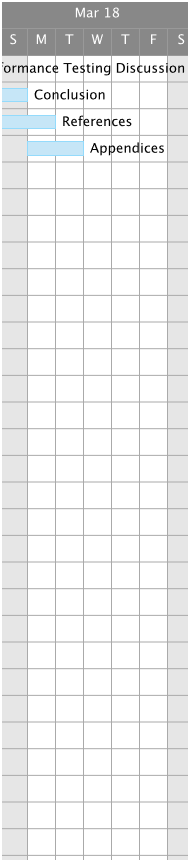
For Agent:

1. Manage and create customer
2. Manage and create customer’s items
3. Make a schedule booking
4. Receive notification regarding the booking status

# Project Plan







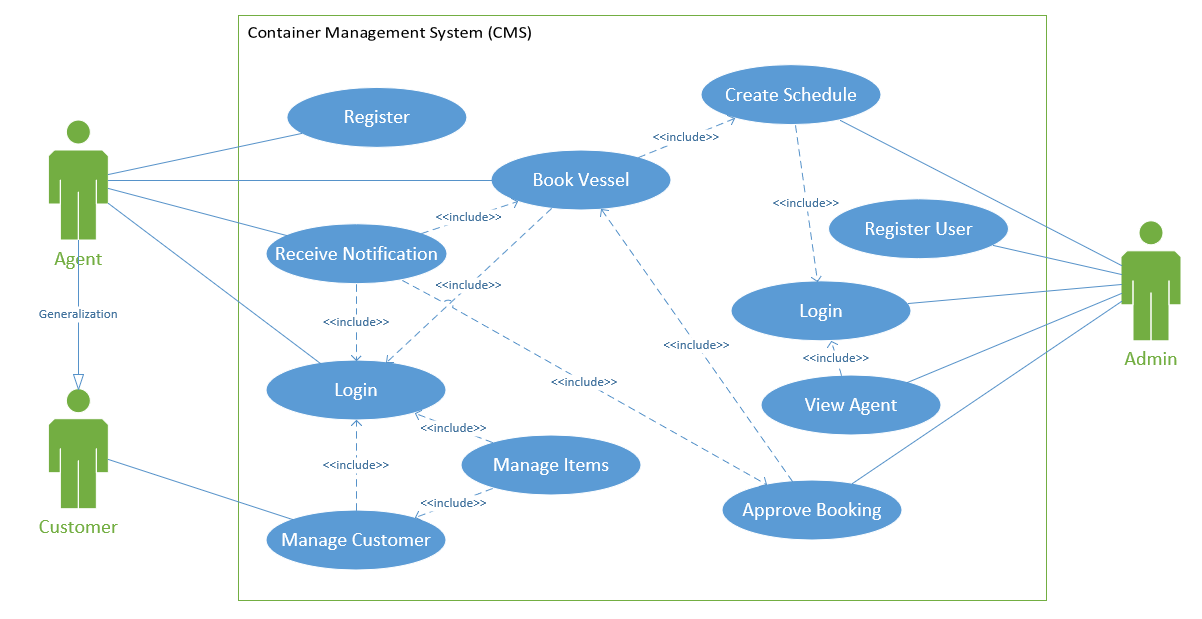
# Design

## Design Consideration

There are some elements to be aware of in this project. The considerations and assumptions are as followed:

1. Container Management System is targeting the users from global wide at which the users can be made up of different races, different countries and different individuals.
2. The storage of each vessel is should be over a large digit until few thousands to millions, but in this project, the sample data only reach about thousands.

## UML Use Case Diagram



## UML Use-Case Diagram Description

|  |  |
| --- | --- |
| **Use Case** | Login |
| **Summary** | Admin or Agent log on to the system to carry out actions |
| **Dependency** | - |
| **Actors** | Admin, Agent |
| **Preconditions** | User has registered an valid account |
| **Descriptive of Main Sequence** | 1. Users input their username and password into the selected field.  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** | 2.a. Just in case of incorrect credentials is entered, the users will be required to re-enter the correct information.  3.a. Login is unsuccessful, an error message is displayed to the user. |
| **Postcondition** | User has successfully logged on to the system and able to perform their granted actions. |

|  |  |
| --- | --- |
| **Use Case** | Register |
| **Summary** | User has to register before login |
| **Dependency** | - |
| **Actors** | Agent |
| **Preconditions** | - |
| **Descriptive of Main Sequence** | 1. User enters 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** | 2.a. Just in case of existing username entered, the system will request customer to enter a new one.  3.a. If any field is empty, the system will display error message “invalid register”. |
| **Postcondition** | User has registered his/ her unique personal account and able to log on to the system. |

|  |  |
| --- | --- |
| **Use Case** | View Agents |
| **Summary** | Users able to view existing agents |
| **Dependency** | <<Include>> Login |
| **Actors** | Admin |
| **Preconditions** | User is logged on to the system |
| **Descriptive of Main Sequence** | 1. User press the “View Agent” tab.  2. System displays page with existing agents.  3. User can view the agent details. |
| **Descriptive of Alternative Sequence** | 2.a. There is no agent created or description of the agent is not related. |
| **Postcondition** | Existing agents can be seen. |

|  |  |
| --- | --- |
| **Use Case** | Create Schedule |
| **Summary** | Admin creates a new schedule for vessels. |
| **Dependency** | <<Include>> Login |
| **Actors** | Admin |
| **Preconditions** | Admin is logged on to the system |
| **Descriptive of Main Sequence** | 1. User press the “Create” button under schedule tab.  2. System displays a new creation form.  3. Admin enters the required event details.  3. Admin press the “create” button to build the schedule.  4. System will publish the schedule, |
| **Descriptive of Alternative Sequence** | 2.a. The description of the schedule is not related or the schedule start date greater than deadline triggers error message.  3.a. Admin does not fill in the required information will display the error message. |
| **Postcondition** | A new schedule is created. |

|  |  |
| --- | --- |
| **Use Case** | Manage Booking |
| **Summary** | Admin manages existing booking. |
| **Dependency** | <<Include>> Book Vessel |
| **Actors** | Admin |
| **Preconditions** | Admin is logged on to the system |
| **Descriptive of Main Sequence** | 1. Agent has to create a booking first.  2. Admin click on “Booking” tab.  3. System will display a page of the agent created schedule.  3. Admin select booking and click on approve or decline button.  4. System will update the booking details accordingly. |
| **Descriptive of Alternative Sequence** | 2.a. There is no booking created by the agent.  3.a. Agent does not fill in the required information to change will display the error message. |
| **Postcondition** | Booking is approved. |

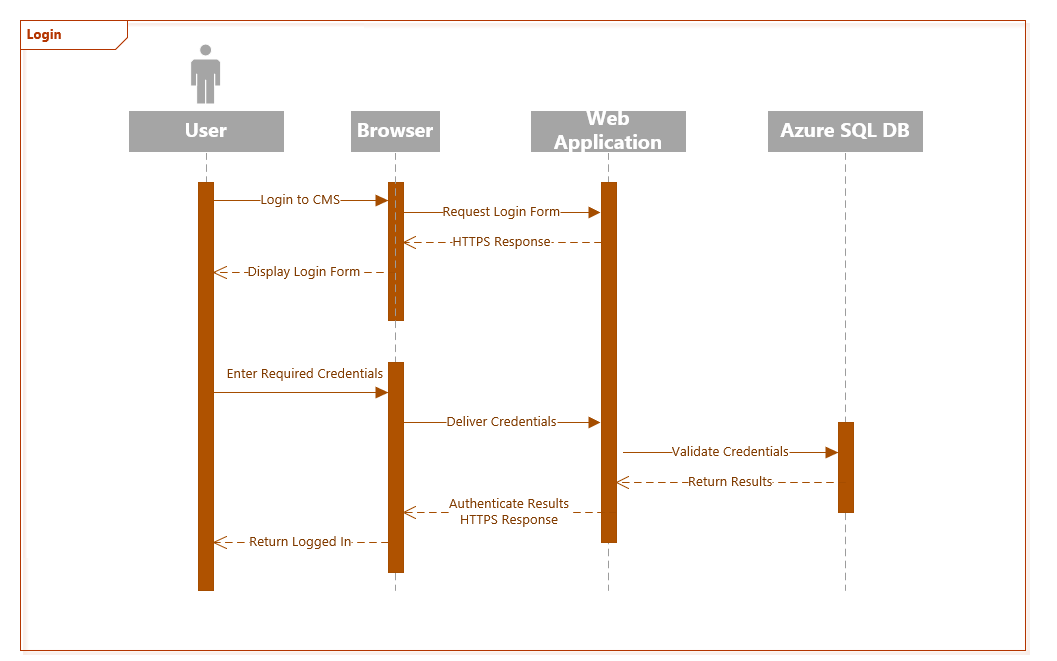
|  |  |
| --- | --- |
| **Use Case** | Manage Customer |
| **Summary** | Users able to manage customer. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | User is logged on to the system |
| **Descriptive of Main Sequence** | 1. User press the “Manage Customer” tab.  2. System displays page with existing customer.  3. User can edit the customer.  4. User can add a new customer.  5. System will insert the new customer data. |
| **Descriptive of Alternative Sequence** | 2.a. There is error in customer details. |
| **Postcondition** | New Customer is created. |

|  |  |
| --- | --- |
| **Use Case** | Manage Item |
| **Summary** | Users able to manage item. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | User is logged on to the system |
| **Descriptive of Main Sequence** | 1. User press the “Manage Item” tab.  2. System displays page with existing item.  3. User can edit the item.  4. User can add a new item.  5. System will insert the new item data. |
| **Descriptive of Alternative Sequence** | 2.a. There is no customer.  3a. There is error in item details. |
| **Postcondition** | New Item is created. |

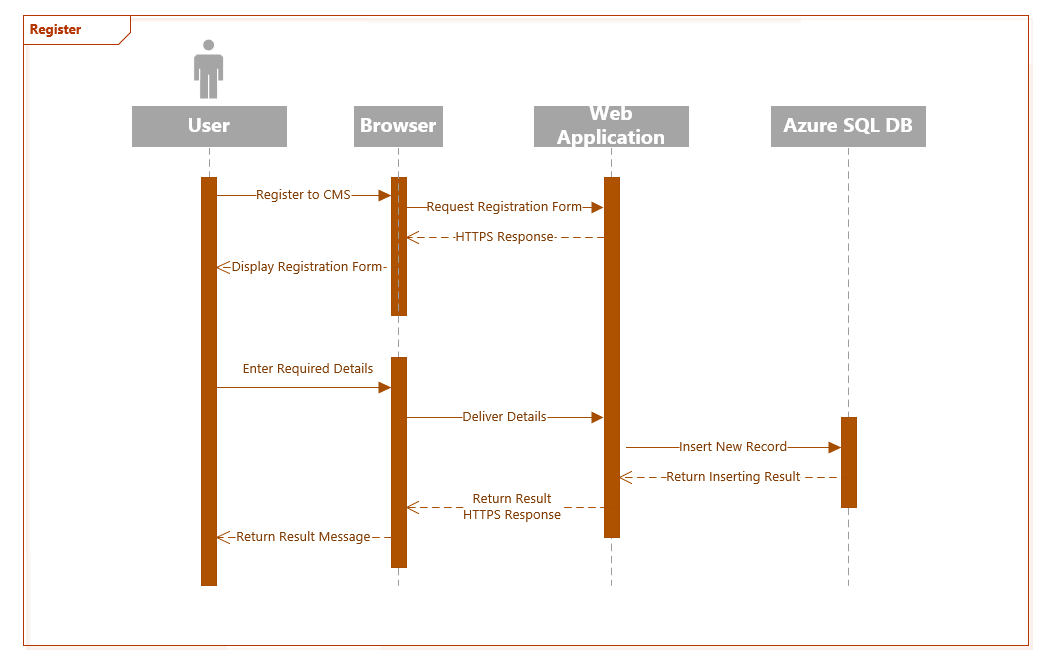
|  |  |
| --- | --- |
| **Use Case** | Book Vessel |
| **Summary** | Agent book a vessel. |
| **Dependency** | <<Include>> Login |
| **Actors** | Agent |
| **Preconditions** | Agent is logged on to the system |
| **Descriptive of Main Sequence** | 1. User press the “Book Vessel” tab.  2. System displays a page with available schedule.  3. User choose a schedule.  4. User click on a vessel type.  5. User enter the required details.  5. System will add the booking for the selected schedule and admin will be notified. |
| **Descriptive of Alternative Sequence** | 2.a. There is no schedule created.  3.a. There is no customer to choose. |
| **Postcondition** | Vessel is booked. |

## Sequence Diagram

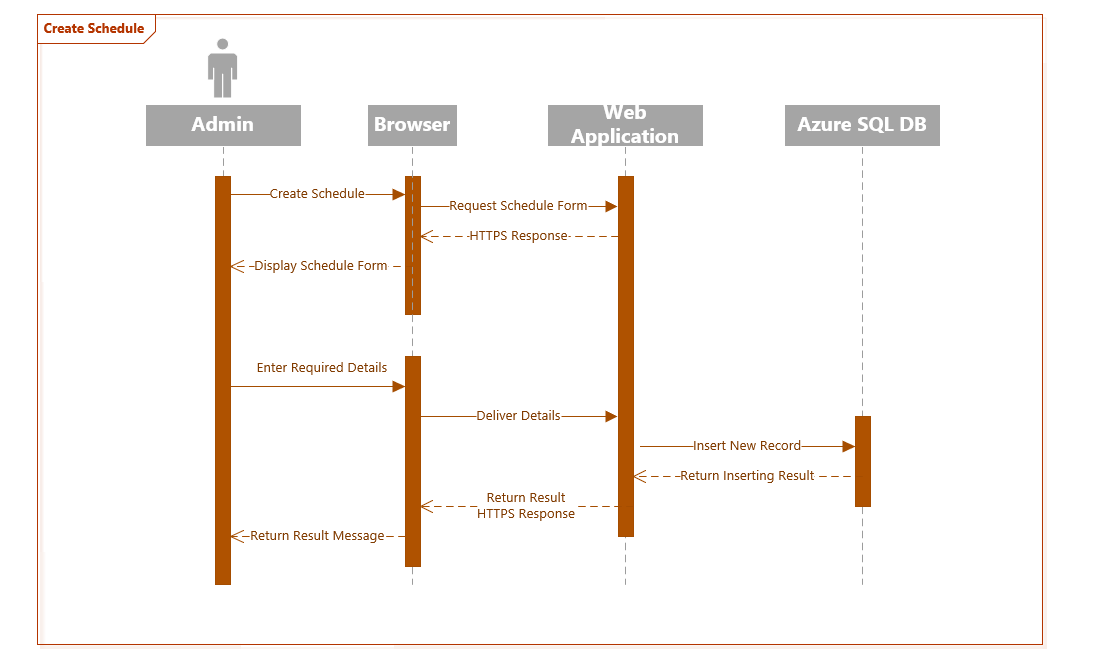
1. Login



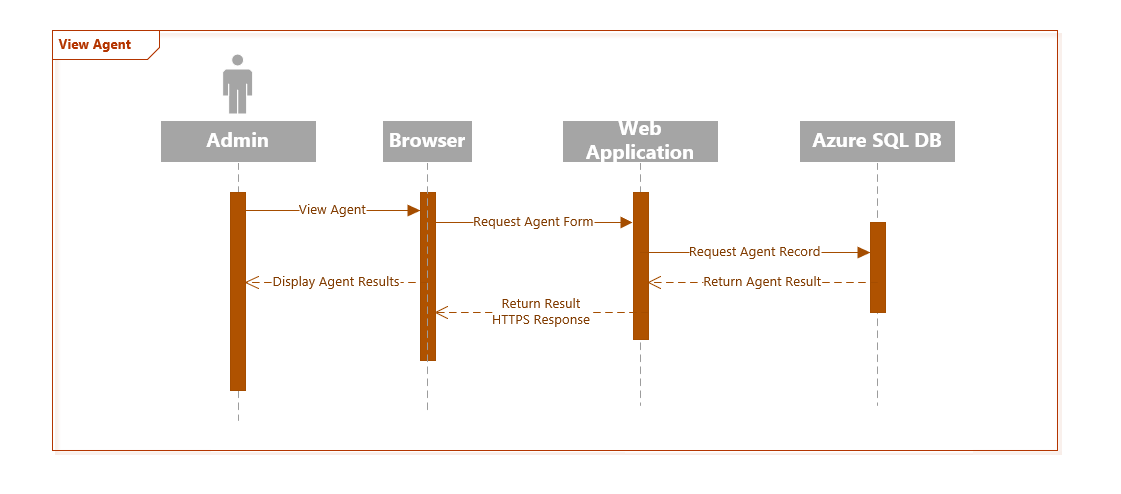
1. Register



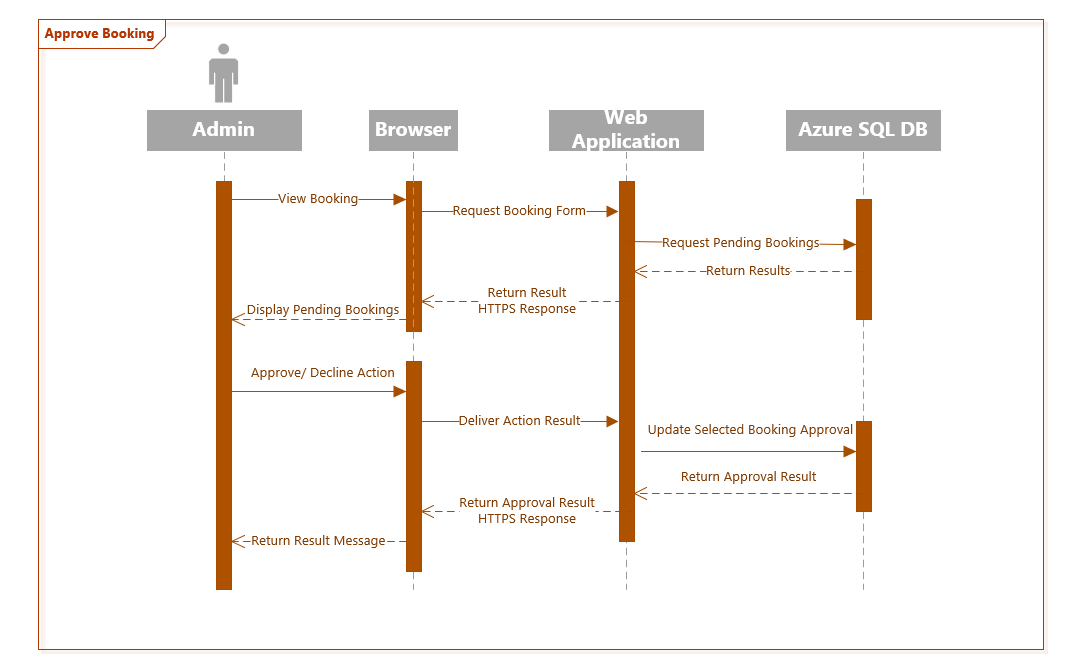
1. Create Schedule



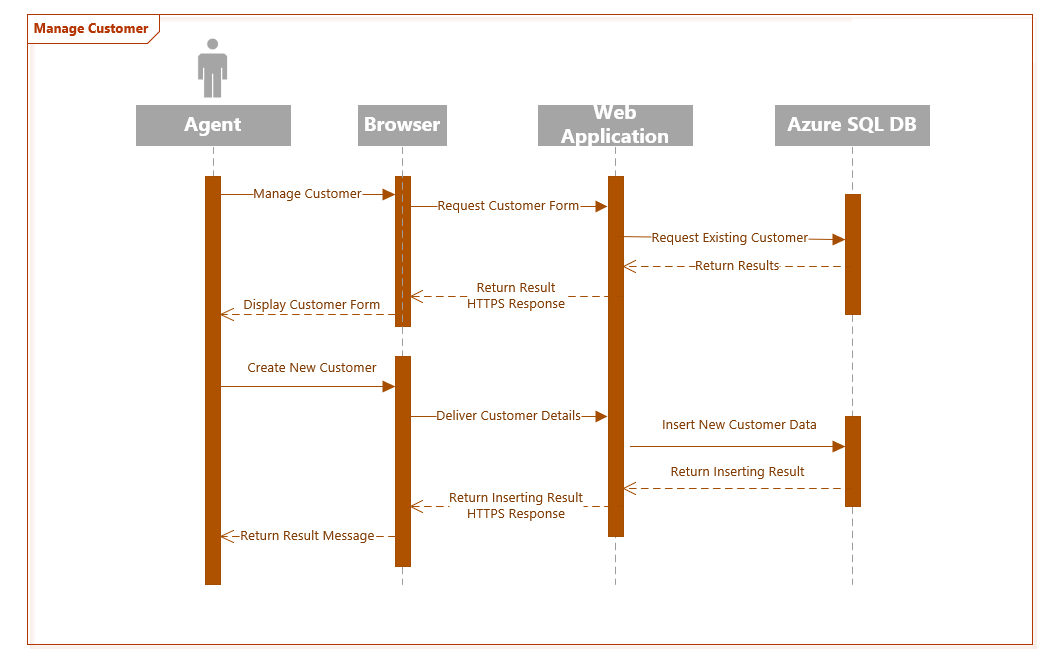
1. View Agent



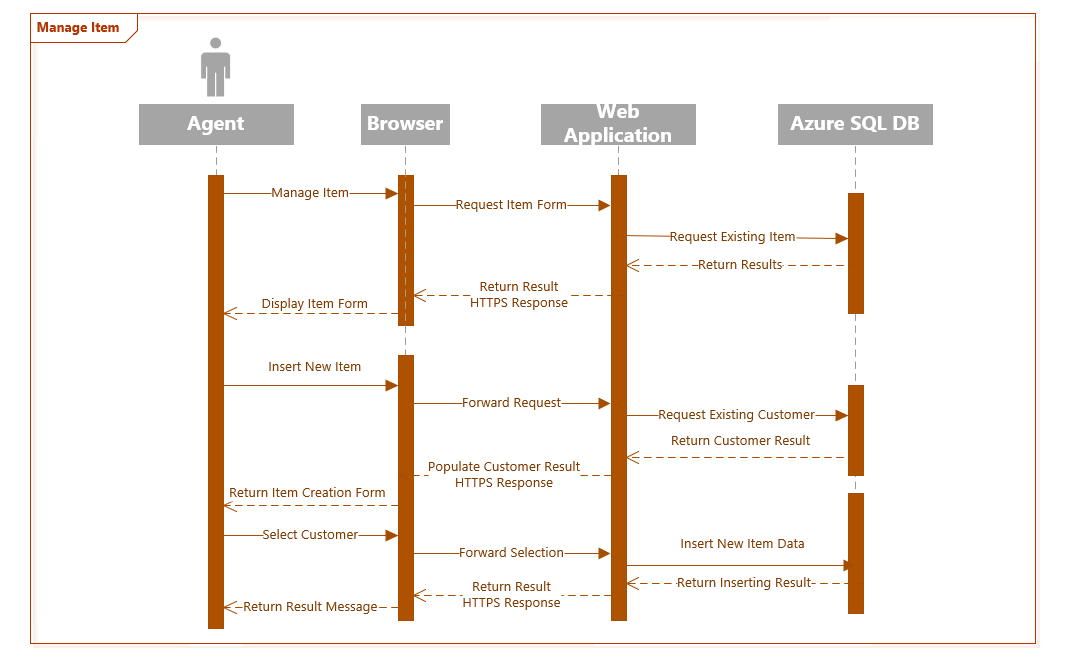
1. View Agent



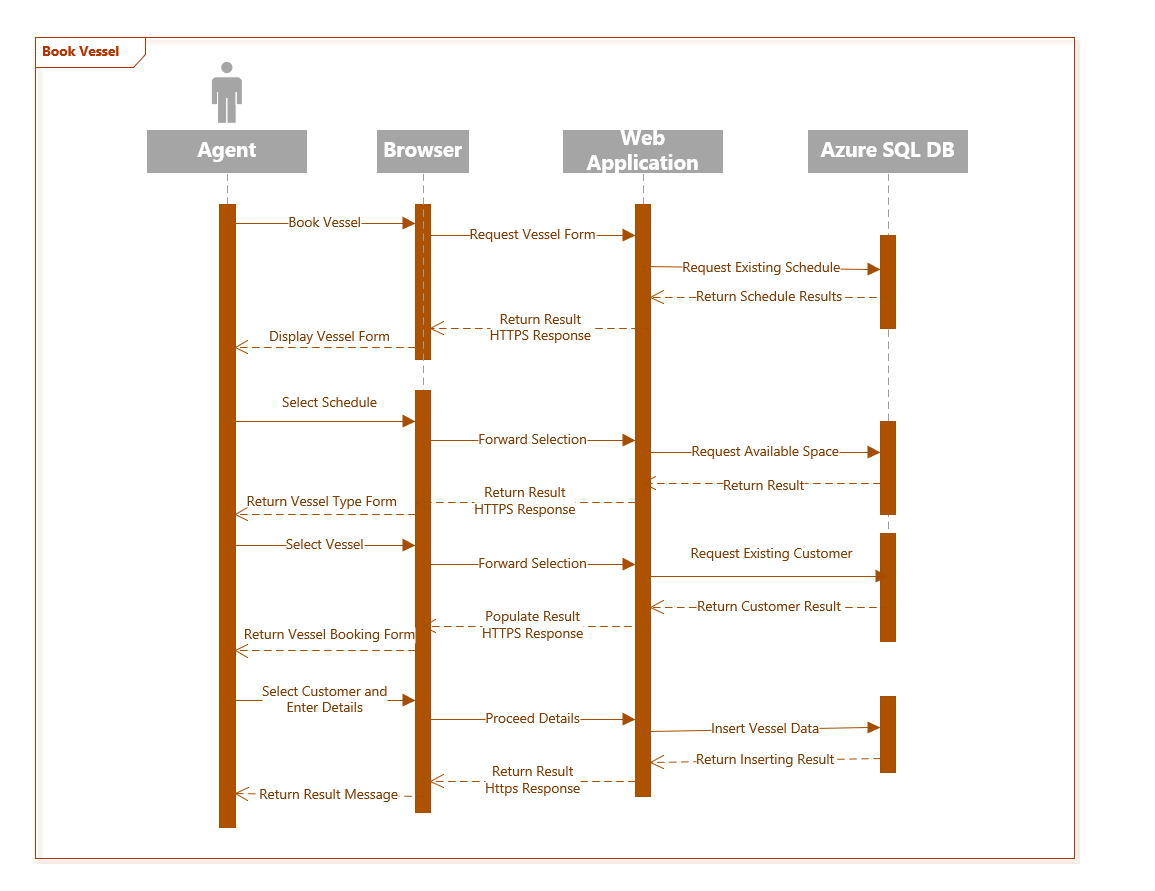
1. Manage Customer



1. Manage Item

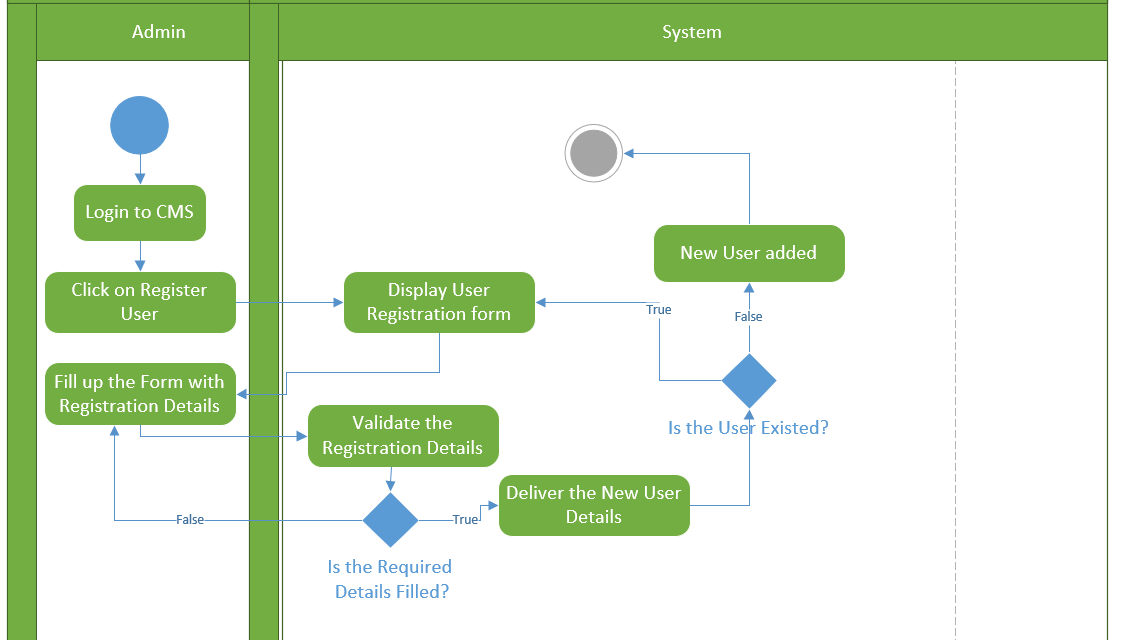


1. Book Vessel

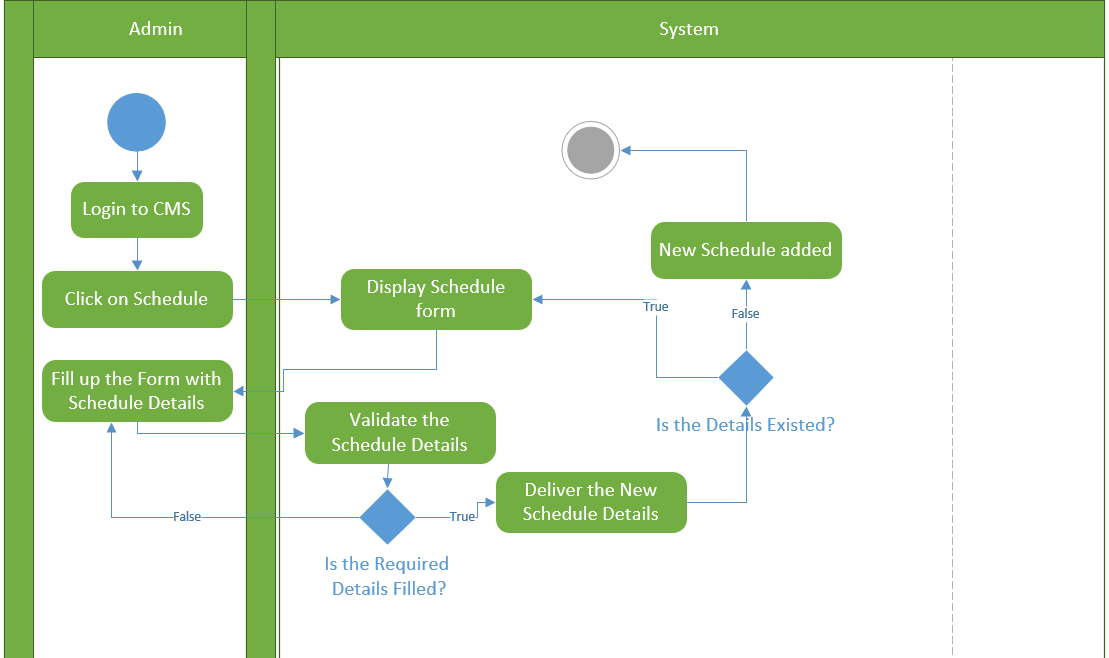


## Activity Diagrams

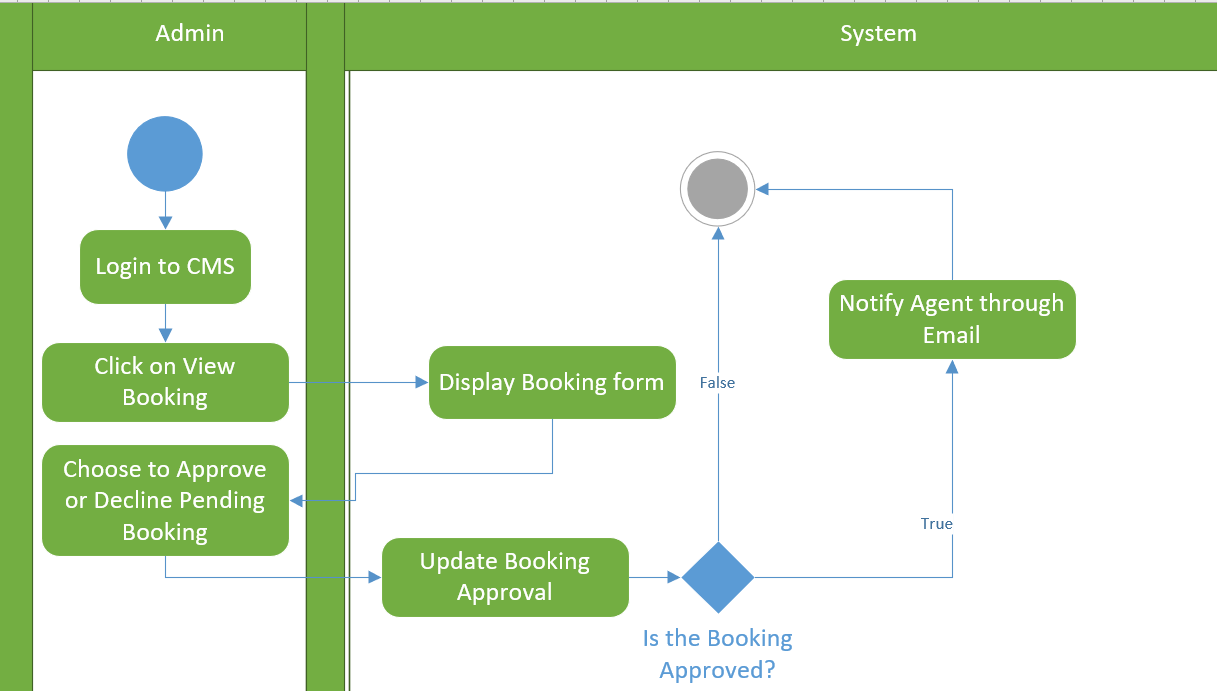
1. Login



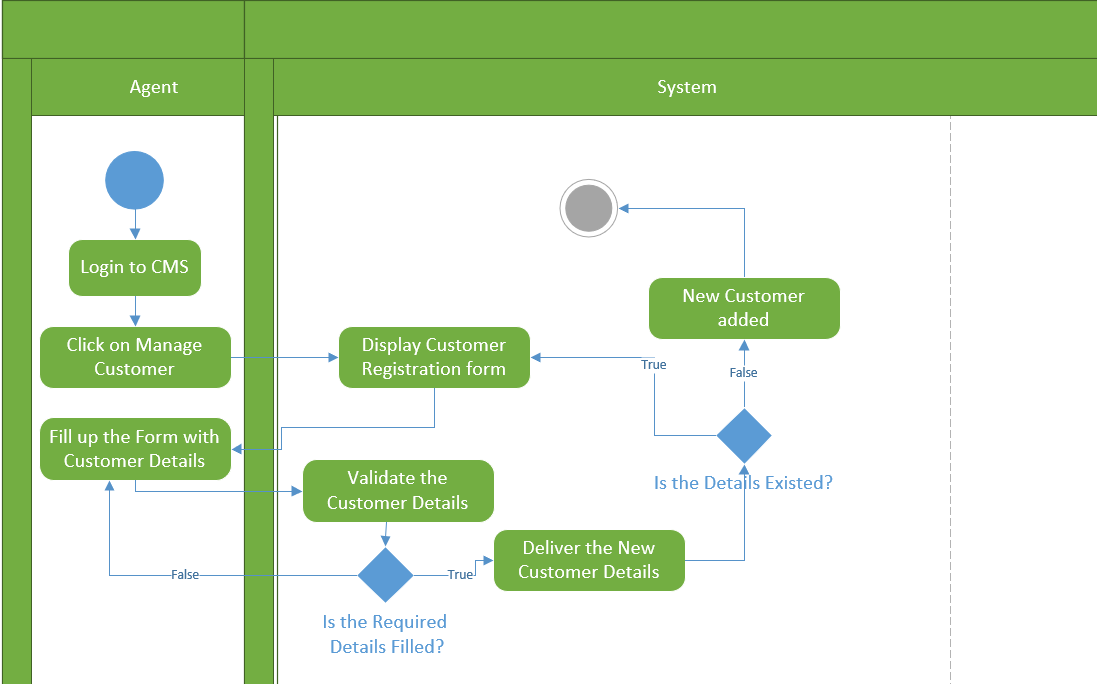
1. Create Schedule



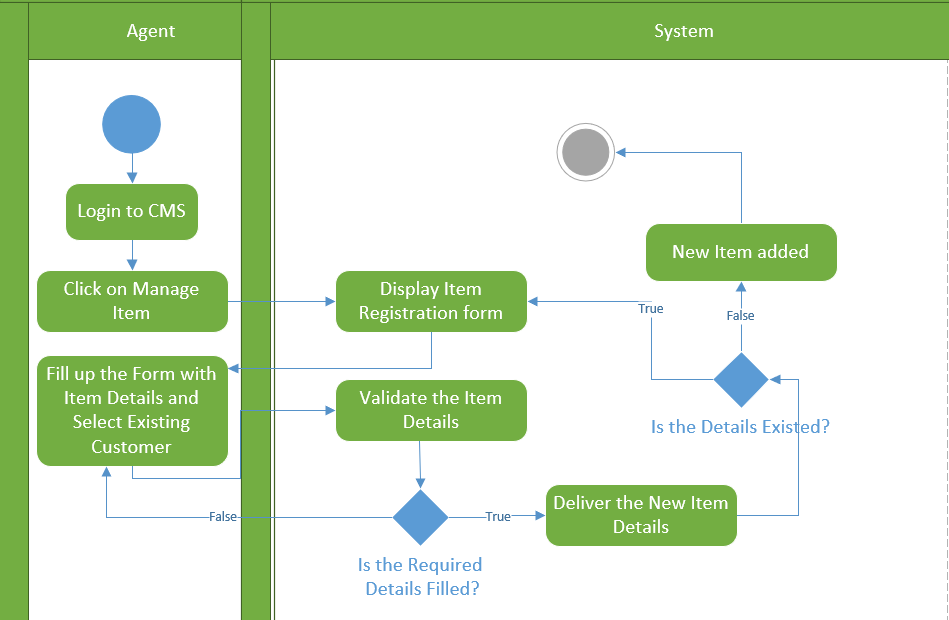
1. Manage Booking



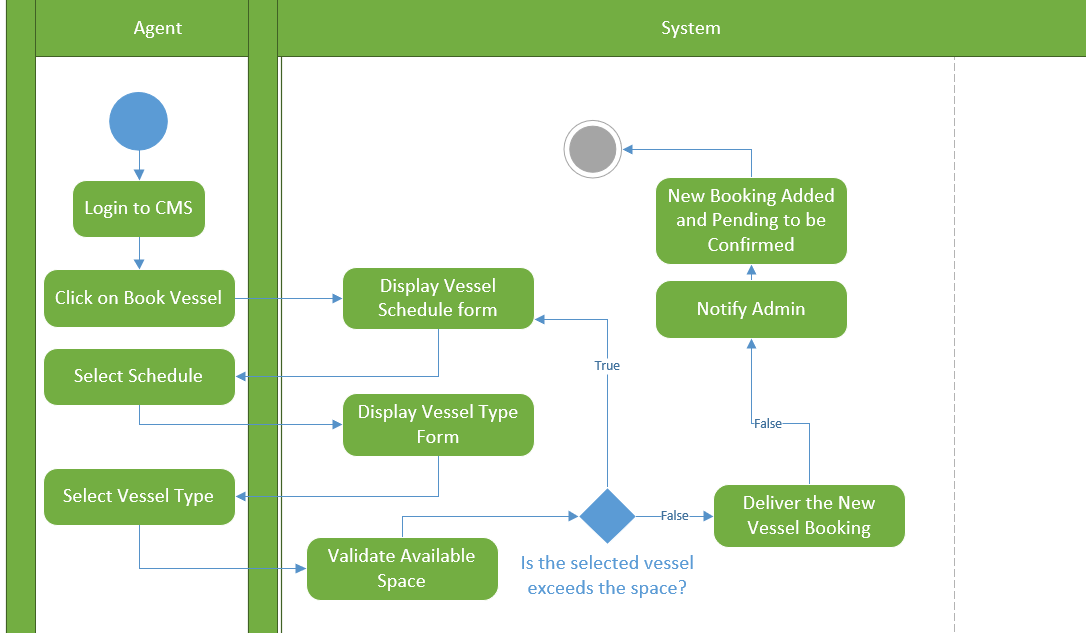
1. Manage Customer



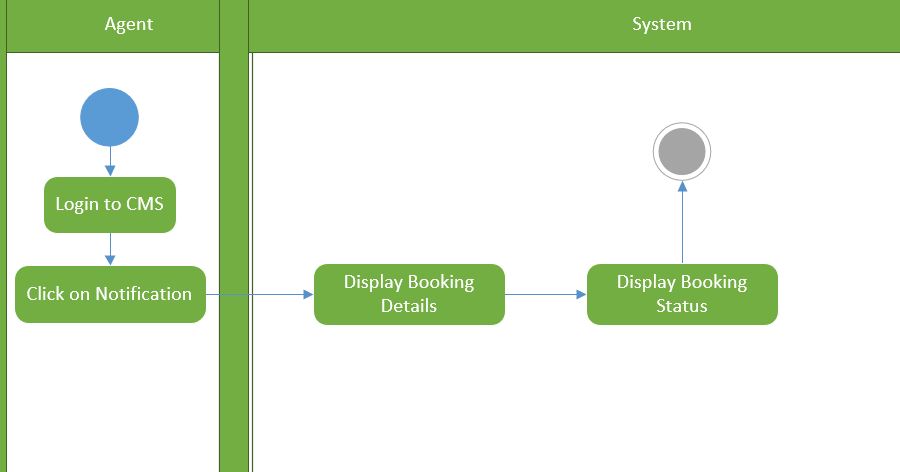
1. Manage Item



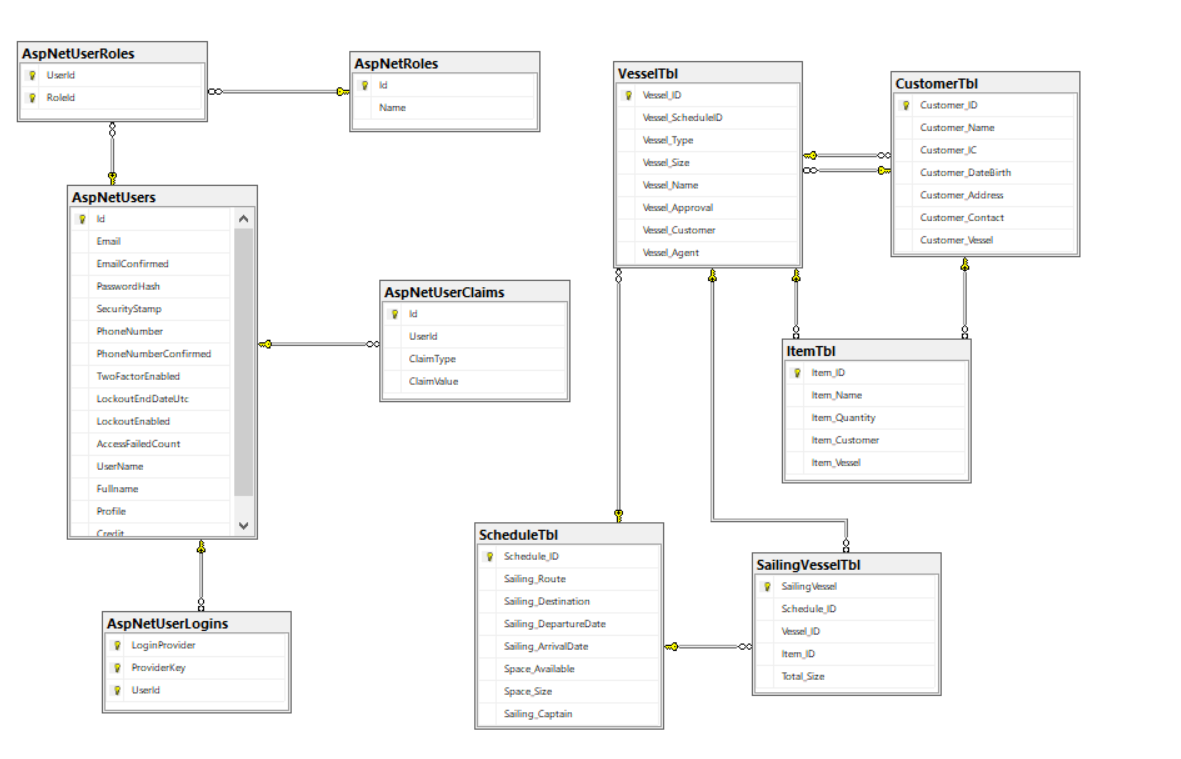
1. Book Vessel



1. Receive Notification



## Entity Relationship Diagram



## Cloud Design Pattern

This assignment has implemented one of the cloud design patterns. According to Amazon (2012), the Cloud Design Patterns (CDP) are a collection of solutions and design ideas for using cloud technology to solve common systems design problems. The design patterns are categorized by reviewing numerous designs made by the cloud architects. As for this project, Compute Resource Consolidation Pattern is used.

There are usually a lot of operations in cloud. In most of the solutions, it most probably initials to follow the design principle of separation of concerns and divide these operations into discrete computational units which are hosted and deployed individually. Publishing a large variety of computational units as part of the same application will significantly increase runtime hosting costs and make management of the system to be more complex although this strategy is able to simplify the logical design of the solution.

**Compute Resource Consolidation Pattern**

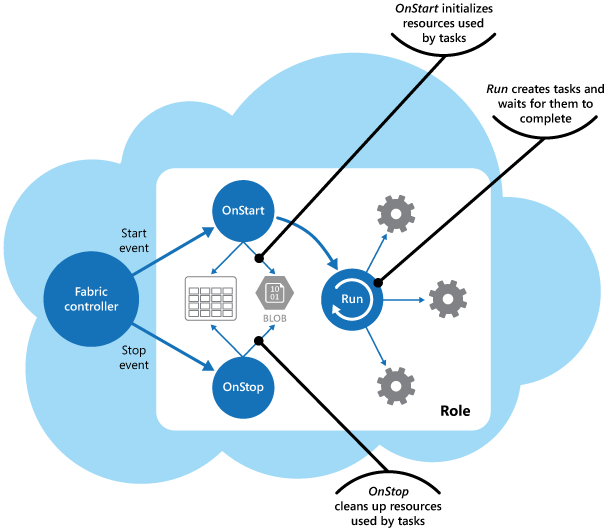


Diagram C

The Diagram C above shows a fabric controller is controlling all the resource units. The units and roles are awaiting the calls from the controller before executing any task. It is believed that this design pattern will help reduce costs, increase utilization, improve communication speed, and ease the management effort of Maersk Line. Through this pattern it is also possible to consolidate multiple tasks or operations into a single computational unit. Tasks are categorized according to a variety of criteria based on the features provided by the environment, and the costs associated with these features. Categorizing these features together will allows them to be scaled as a unit. It also provides additional instances of a computational unit to be started and stopped according to the workload. For instances, Windows Azure offers autoscaling which can be applied to roles in a Cloud Service, Web Applications, and Virtual Machines. To be generalized, the greater resources specified, the higher the cost. To improve financial status, it is crucial to maximize the amount of work an expensive computational unit performs, and not let it to be offline or inactive for an extended instance.

## Cloud Architecture

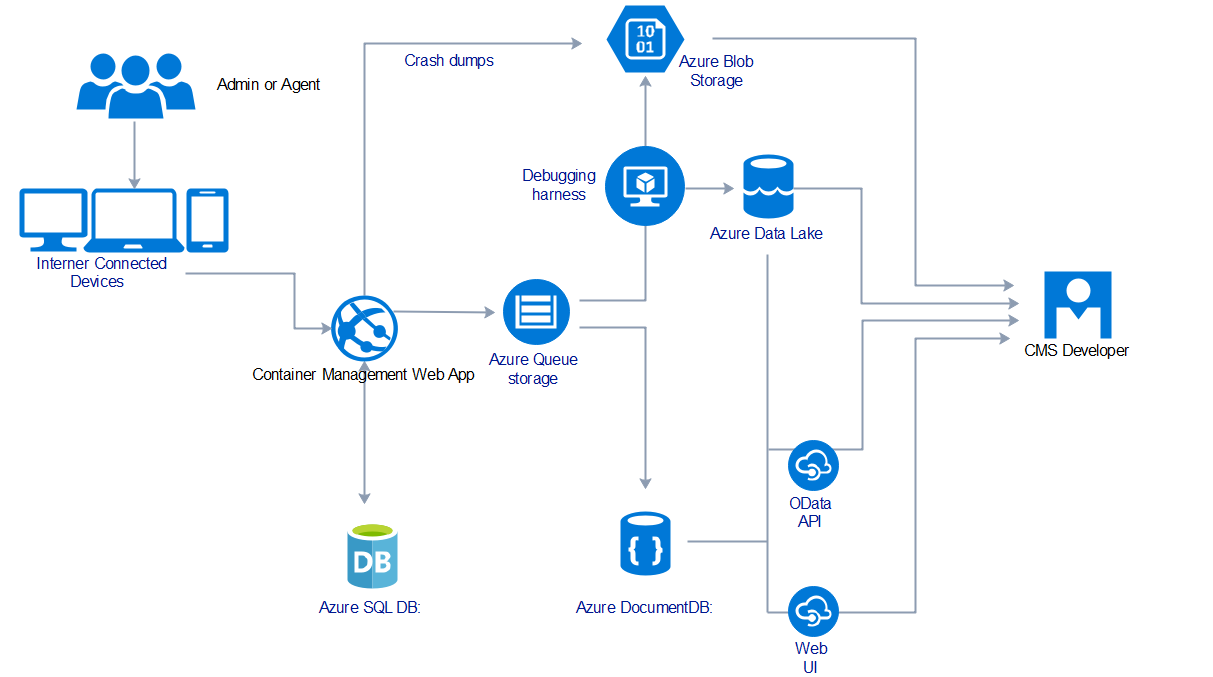


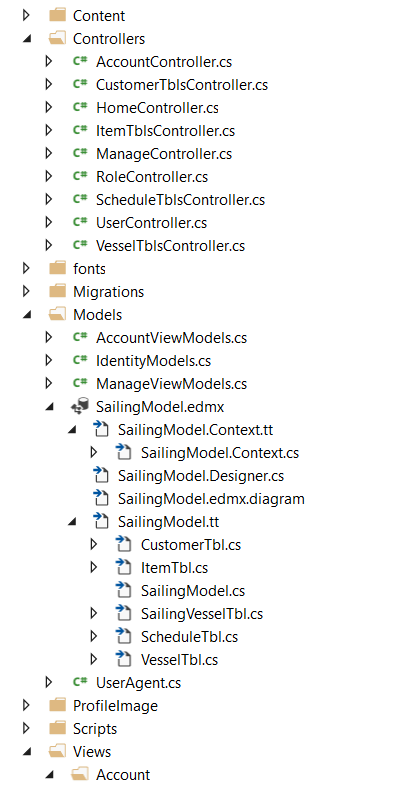
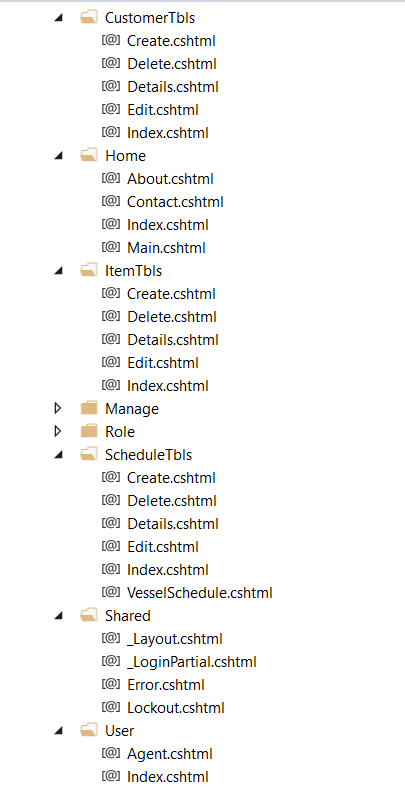
Diagram D

Diagram D shows the cloud architecture diagram for azure designing. The users like admin and agent have to be connected with internet access devices. Then, they are able to access the container management web application hosted by Microsoft Azure. The web application is subscribed to South East Asia region for better performance and speed. When the user attempts to login, it will trigger the Azure SQL Database for the data retrieving. If the web application encounters any problem such as error reading scale from Azure DB, the errors will be stored as crash dumps to Azure Blob Storage. For normal circumstance, the web application will interact to queue storage and store the debugging data in Azure Data Lake. All the log records and crash reports can be received by the system developer.

# Implementation

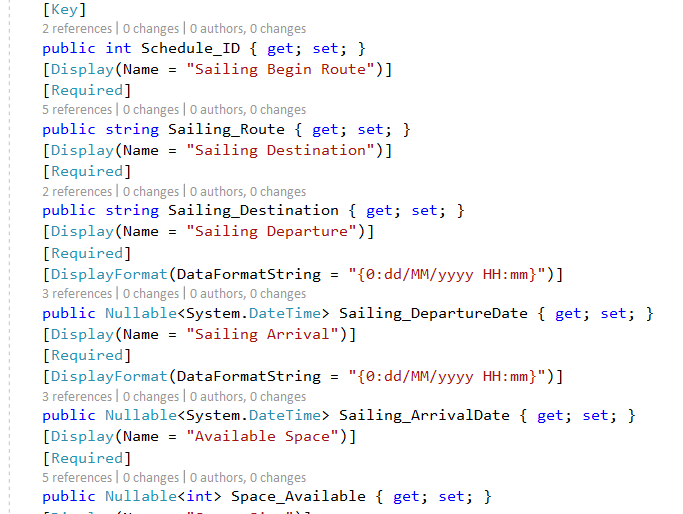
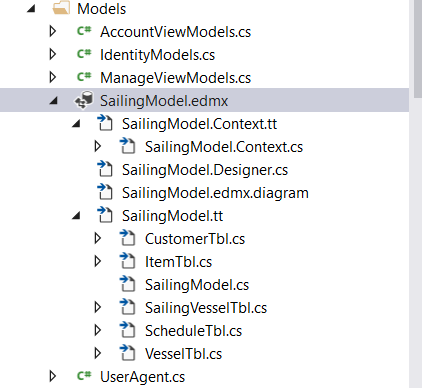
## Application Development

**Controller & View**

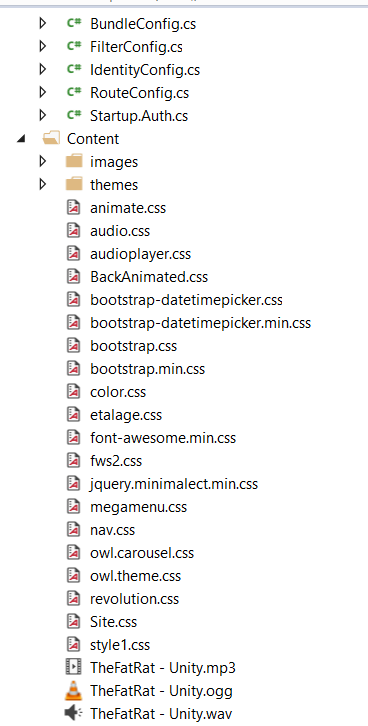
The development of the Container Management web application is produced using Microsoft ASP.NET MVC with C# Programming Language. The web application uses an Model, View and Controller structures to work. All the website pages are controlled by the specific controller. For example, if a user wants to book a vessel, the application will proceed to VesselTbls Controller first to execute the Return View() action in order to generate the vessel booking page

**Model**



This system also makes uses of ADO.NET Entity Data Model which utilize the use of Code-First Database of Microsoft SQL Server before publishing to Azure SQL. There are few crucial models related to the development of the system like Schedule, Vessel, Customer and Item Model. These models significantly define the attributes stored in the database and carry out the validation checking when entering the data.

**CSS & JavaScript**



The fundamental part of system development would be utilizing use of CSS files and Java Scripts files. All the related CSS files are stored under Content whereas Java Scripts files are stored under Scripts folder. These files are then referred from the BundleConfig.cs. From the Bundle Config, the required CSS and JavaScript are defined and the corresponded design for the page can be rendered by calling @Styles.Render("~/Content/css").

**Web Application**

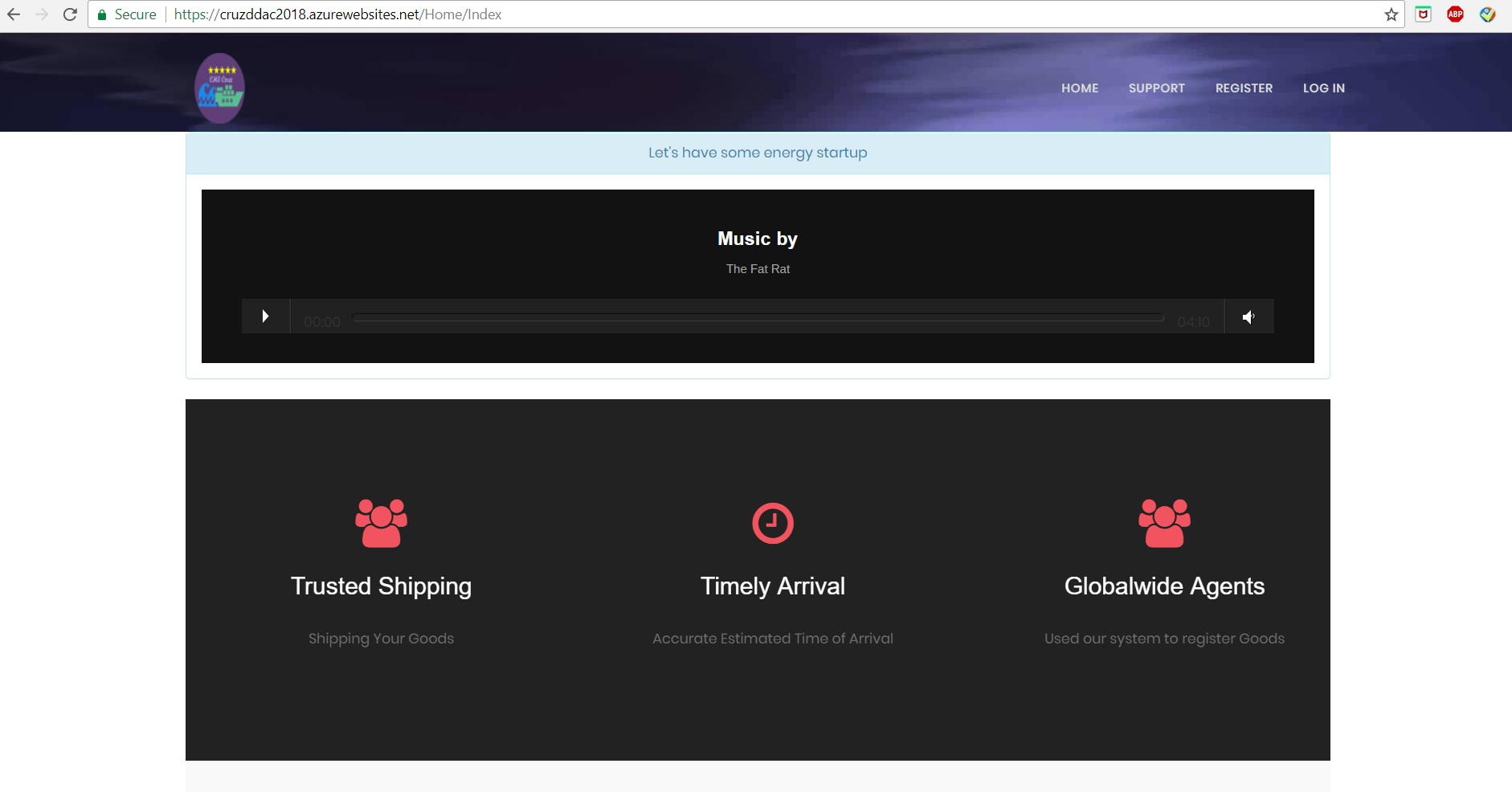
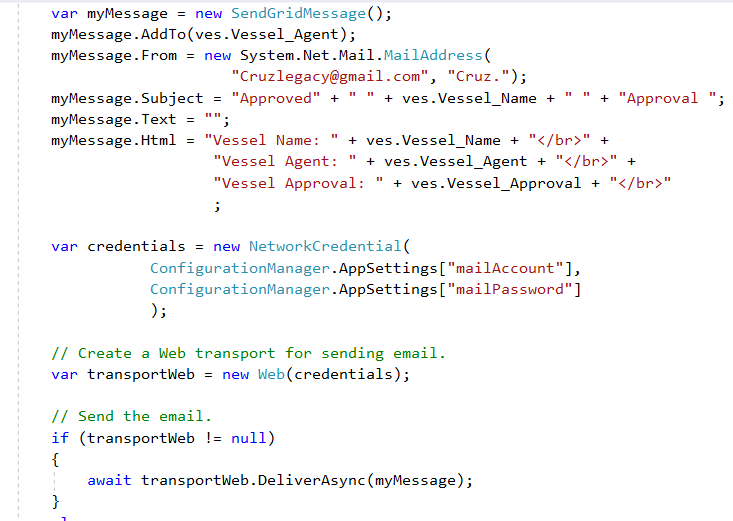
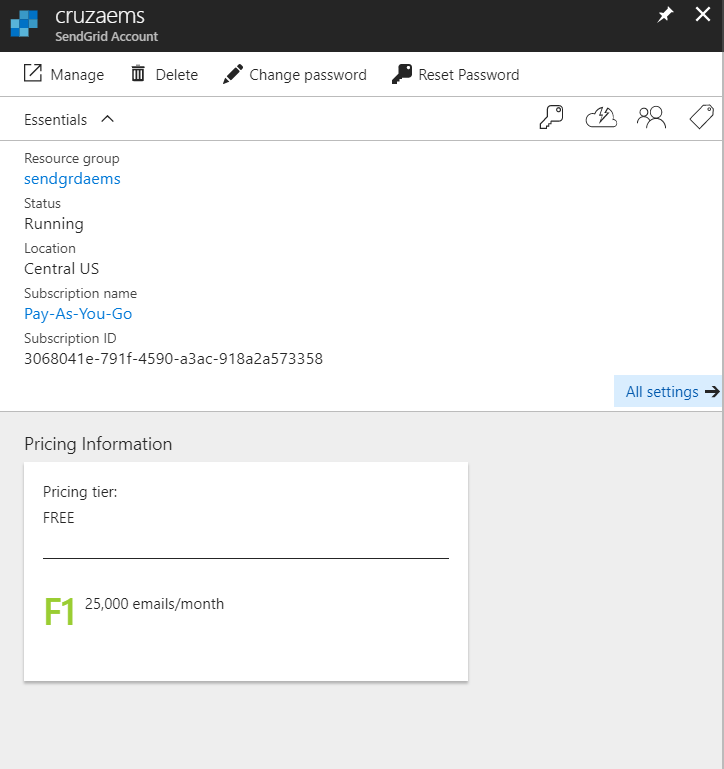


Diagram above indicates the home page of the Container Management Web Application. From this page, the developer has implemented a Music Box and some icons with reference to Bootstrap. A logo has also been designed for the application, it is shown on the top left corner which is a circle with five stars, ship and wave.

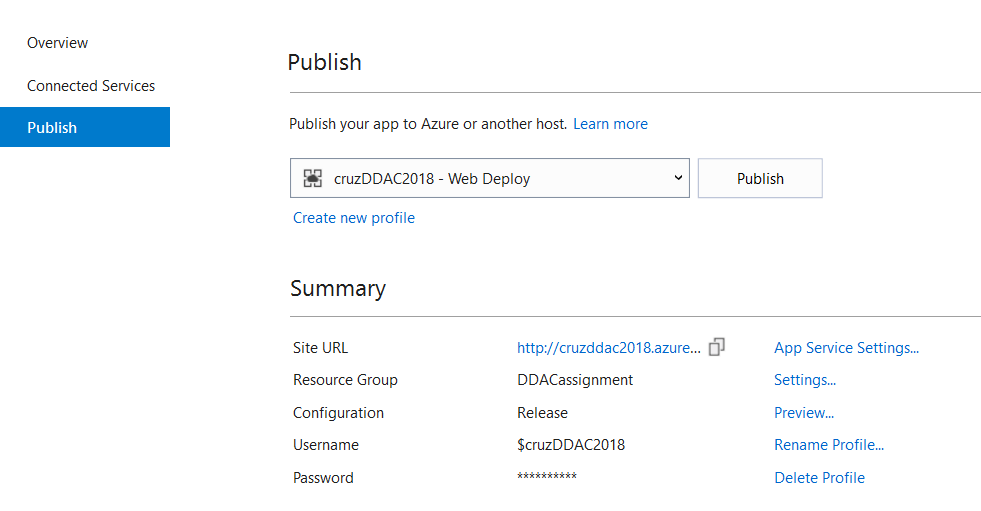
For further understanding of how the application works, the application source codes can be accessed through GitHub link: https://github.com/cruzlegacy/cruzDDAC and the final output of the application through <https://cruzddac2018.azurewebsites.net/> .

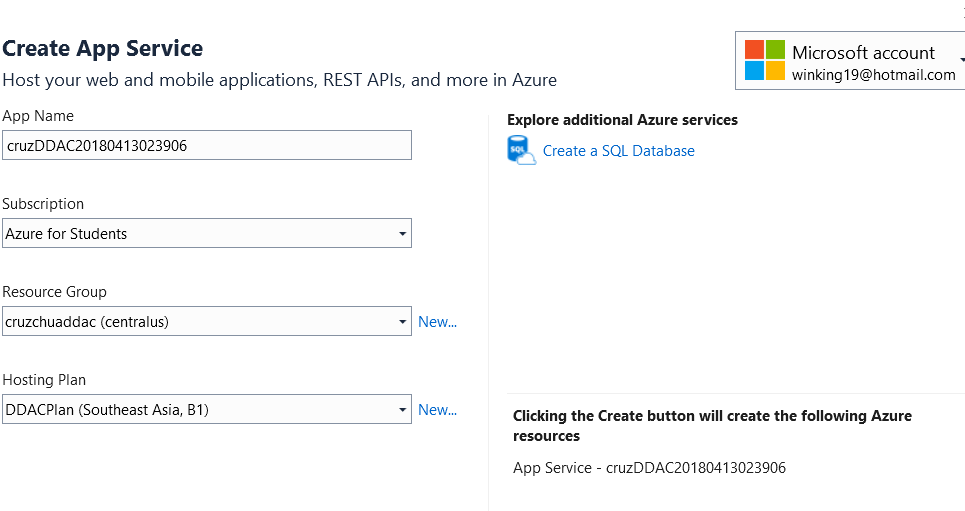
## Email Services



This application also makes uses of the available resources and services from Microsoft Azure. Through Azure, a email service vendor, SendGrid can be connected and its service can be accessed. It is an email service which allow the users to send the email information with the codes defined. The credentials for accessing the email is defined in the Web.Config files which is not visible for security reasons. As shown in the diagrams above, when the admin approves a booking, the email service will be activated at which the system will generate an automated email to the selected agent email with the booking details.

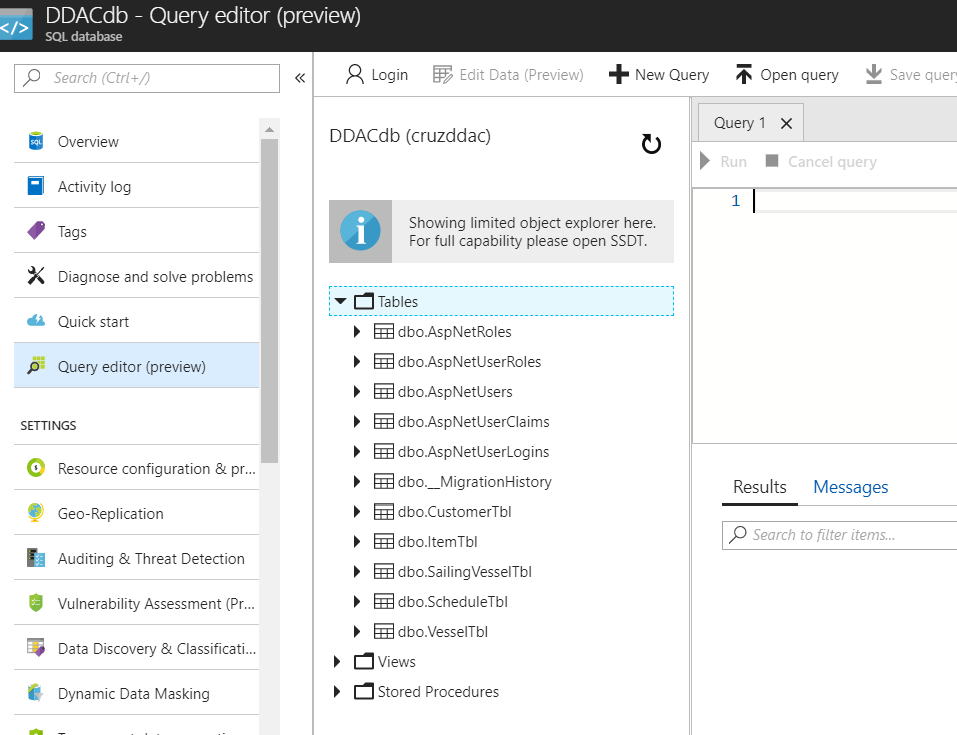
## Azure Publishing



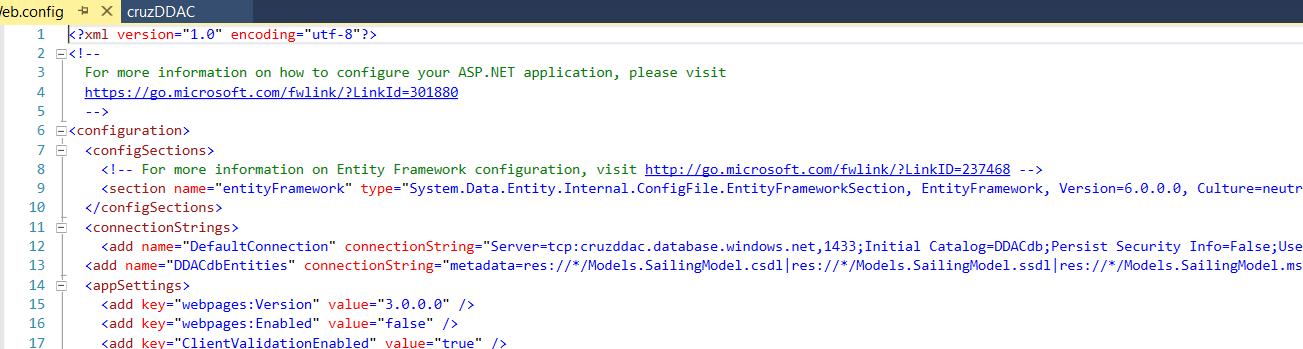


The web application is published to Microsoft Azure using Visual Studio 2017 built-in extension. At first, the developer needs to sign in the email which is registered to Microsoft Azure in the Visual Studio. Then, create a new profile under publishing the solution section. All the required Azure Service details needed to be filled in such as Application Name, Subscription, Resource Group and Hosting Plan.

**Azure SQL Database**



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



After creating the Azure SQL DB, the connection string for the DB is referred and stored under Web.Config file of the project solution. The server name, initial catalogue, user id and password must be corresponded to the connection string in the Azure SQL DB. At the end, the connection of the project to the database is successful.

## Application Scaling

There are several Pricing Tier for the application to subscribe in Microsoft Azure Web Services. Microsoft Azure also provides web service for auto scaling for the application best performance. Autoscaling can be said as a built-in feature of Cloud Services, Virtual Machines, and Websites which helps the applications to perform their best when demand changes (Microsoft Azure, 2018). Performance can be differed for different size and scaled of applications. If the developed web application , Container Management System tends to increase its performance. The pricing tier of the web services can be up scaled to the affordable pricing. It is crucial to increase the tier because when the Container Management Web Application increase the workload, more resources are needed to maintain the desired performance levels and output.

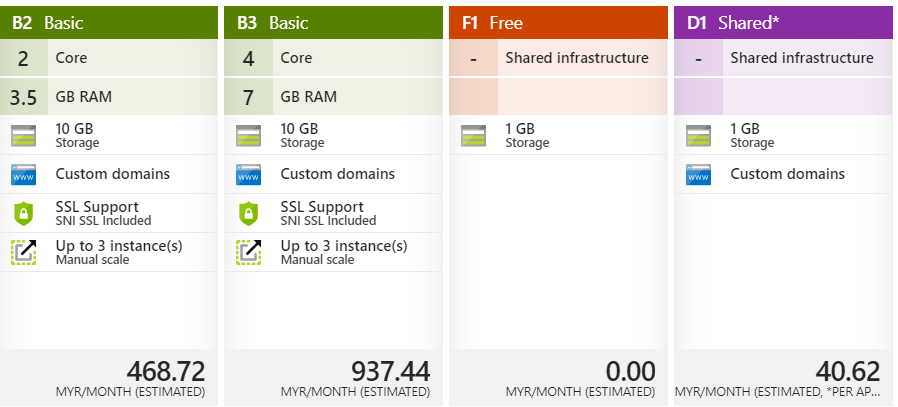
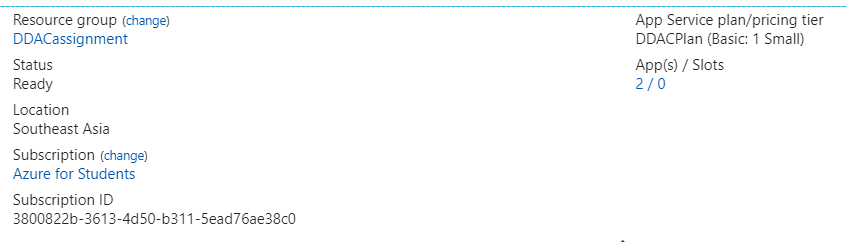
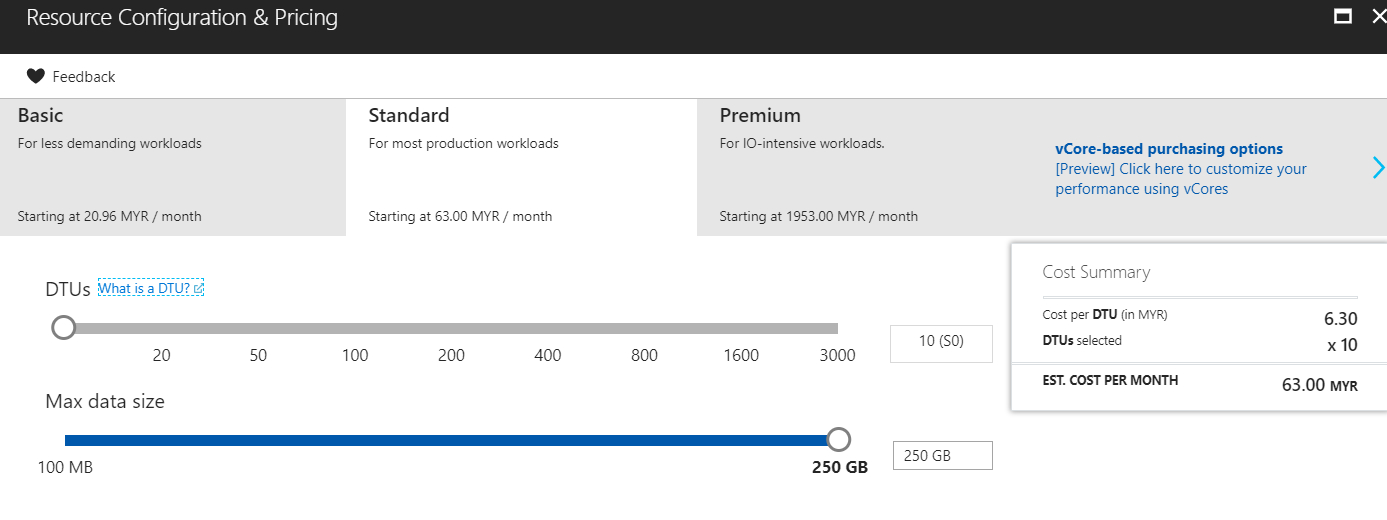


Diagram B: Pricing Tier of Container Management System

With the standards set by Microsoft Azure, all web applications other than Student Subscription must be subscribed to a basic, B1 plan. The basic plan is the standard of most of the personal development project with a small variety of developers. The domain for the web application is set at the South East Asia (SEA) Region because the web application is developed in the country, Malaysia which belongs to SEA region. Subscripting to domain in other country such as United State region will significantly slows down the performance of the web application as the traffic in between the communication network is usually far and complex.

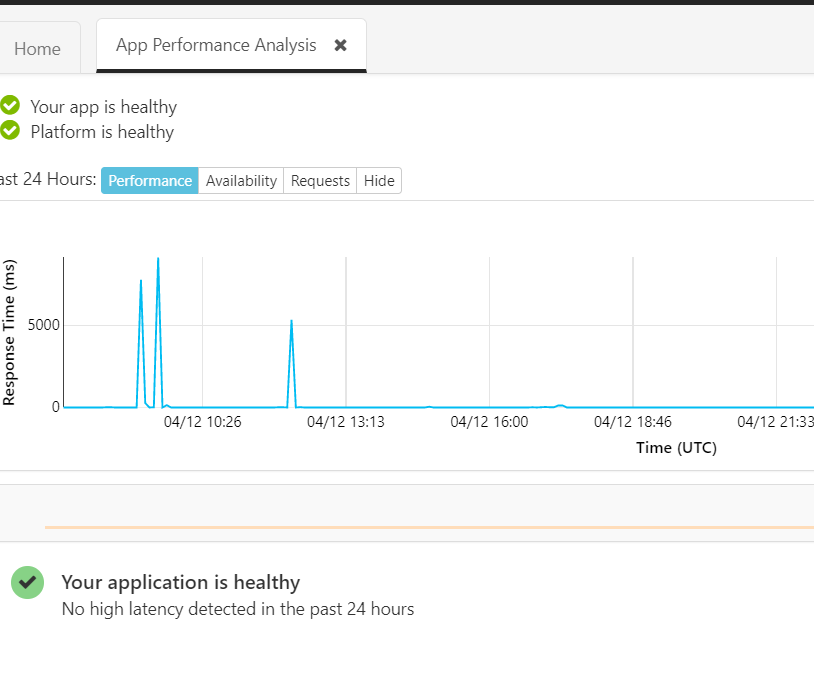


The current subscription of the web application is B1, Basic, at which it only provides one core with 1.75GB Ram. Since the system is still being developed for developer use, it is sufficient to stay the pricing tier at B1. If there is sponsor or the Maersk owner wish to scale up the application for operation use, it is recommended to upgrade the solution to B3 for better processors and ram.

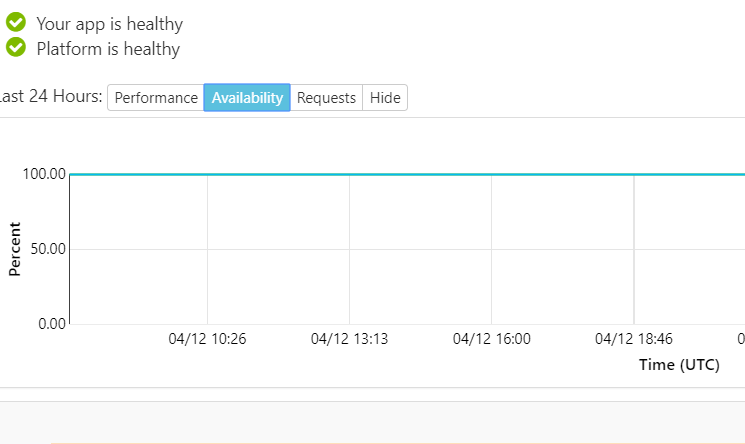


The Azure SQL Database will be using a Standard Tier of Pricing. It comes with 10 compute units and maximum data size of 250 GB with RM63.00 per month. The price of the subscription is affordable for the developer and its performance is good for the web application. If there is any further improvement on the system that will require a large amount of data to store, the pricing tier can be upscaled to Premium for better performance and data storage.

## Performance and Availability



The above chart is generated from the Diagnosis Tools provided by Microsoft Azure. It is analysed based on the web application performance and health. It comes with several tabs such as Performance, Availability and Requests. It shows the response of Container Management System for a specific of time. From the chart, the response time is going incline up as the receiving the user action. However, it stays at initial state when there is no user accessing the application. From the results, it clearly indicates that the system will not consume excess usage when it is not in use.



The chart above shows the availability of the web application. Availability can be said as the ability of a system to be functional under stated conditions for a specified period of time or when it is in need. The chart above is essential to measure and evaluate the availability of the system. It clearly shows that the application is remain at 100% all the time which indicates the system runs all the time without going down or offline status. To maximize the availability, the application can be hosted by multiple web services to prevent any errors occur within a single web service and provides a backup for the website to recover.

# Test Plan & Testing Discussion

## Functional: Unit Testing

Unit Testing is basically used to ensure the system code meets the design and user requirements. It is a type of testing which is done by software developers in which the smallest testable module of an application including functions, procedures or interfaces (TheEconomicTimes). It segregates every part of a program and test that the individual parts are working without error. For example, if a set of possible inputs are given to the system, it should return the proper and accurate values. Having unit testing will surely help in maintaining and simplifying the debugging processes.

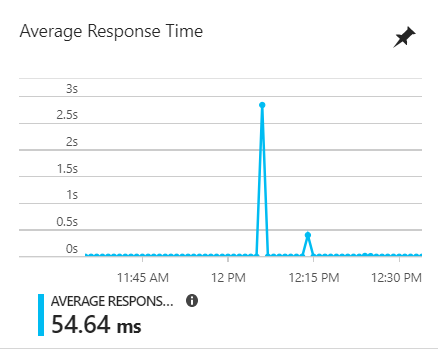
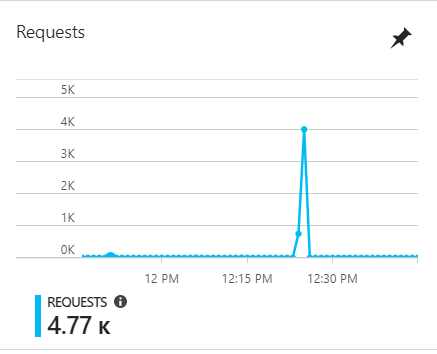
Unit Testing Plan:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **No.** | **Testing**  **ID** | **Testing Function** | **Testing Description** | **Expected**  **Results** | **Actual**  **Results** | **Status** |
| 1 | UT1 | Register as new Agent | Enter the information required for registering a new account to CMS. | New Agent is registered to CMS. | New Agent is registered to CMS. | Success |
| 2 | UT2 | Login to CMS | Enter the user email and password in the specific field and click on login button. | User account is logged in to the system. | User account is logged in to the system. | Success |
| 3 | UT3 | Register User | Click on Register User and enter required details for registration form. | New User is added to the system. | New User is added to the system. | Success |
| 4 | UT4 | Manage Schedule | Click on Schedule and Display all existing schedule. | Existing schedules are displayed. | N Existing schedules are displayed | Success |
| 5 | UT5 | Create Schedule | Click on Create in Schedule Form and Enter the required details. | New schedule is added. | New schedule is added. | Success |
| 6 | UT6 | View Agent | Click on view agent tab. | Existing agents are displayed. | Existing agents are displayed. | Success |
| 7 | UT7 | View Booking | Click on view booking tab. | New pending booking is displayed. | New pending booking is displayed. | Success |
| 8 | UT8 | Approve/ Decline Booking | Click on view booking and select action. | Bookings are either approved or declined. | Bookings are either approved or declined. | Success |
| 9 | UT9 | Update Profile Image | Click on Update Profile button under setting and upload a new profile image. | User’s profile image is changed with the uploaded image. | User’s profile image is changed with the uploaded image | Success |
| 10 | UT10 | Change Password | Click on Change Password button under setting and enter existing password and new password. | User’s password is changed. | User’s password is changed. | Success |
| 11 | UT11 | Log Off | Click on Log Off button on the right of navigation bar. | User is logged out from the system. | User is logged out from the system. | Success |
| 12 | UT12 | Manage Customer | Click on Manage Customer tab. | A list of existing customers is populated. | A list of existing customers is populated. | Success |
| 13 | UT13 | Create Customer | Click on Create button under the Manage Customer tab and enter required details. | New Customer is added. | New Customer is added. | Success |
| 14 | UT14 | Manage Item | Click on Manage Item tab. | A list of existing items is populated. | A list of existing items is populated. | Success |
| 15 | UT15 | Create Item | Click on Create button under the Manage Item tab and enter required details. | New item is added. | New item is added. | Success |
| 16 | UT16 | Book Vessel | Click on Book Vessel tab. | A list of existing schedules is populated. | A list of existing schedules is populated. | Success |
| 17 | UT17 | Select Schedule | Select a schedule under Book Vessel tab. | Schedule is selected. | Schedule is selected. | Success |
| 18 | UT18 | Select Vessel Type | Select a schedule under Book Vessel tab and Select a Vessel Type. | Vessel Type is selected. | Vessel Type is selected. | Success |
| 19 | UT19 | Create Vessel Booking | After selecting schedule and vessel type, enter the required information. | New booking is created. | New booking is created. | Success |
| 20 | UT20 | View Notification | Click on Notification tab. | The status of the booking which is made by the user is displayed. | The status of the booking which is made by the user is displayed. | Success |

## Functional Discussion

As a summary of system validation with the unit testing, all the unit functions of Container Management System had been tested. Unit testing is used by the developer to make sure that the system functions are good without flaw. All the unit testing functions are done successfully and able to achieve the expectation of the developer. No error occurs from this testing, which also means that the system is ready to go through further testing or deploying to public. Therefore, unit testing ensures the system is ideal and able to run in the target environment when deploying to outsiders.

## Performance Test



The diagrams above indicate that the amount of requests by the users and the average response times for the web application site. The average response time of the website after 4770 requests is 54.64ms which is normal and within the requirement of 100ms. The performance test is conducted with the help of Azure web application built-in performance testing feature. These tests are conducted on the main web app resource and will test a user load of 40 to 520 for one minute with each test increasing in user load by 160 user increments. The results that will be collected regardless of response time and failed requests. These tests are carried out on three app service plans such as Standard 1, Standard 2 and Standard 3. The response times are recorded for these tests and are taken into consideration for the time taken of the website to respond.

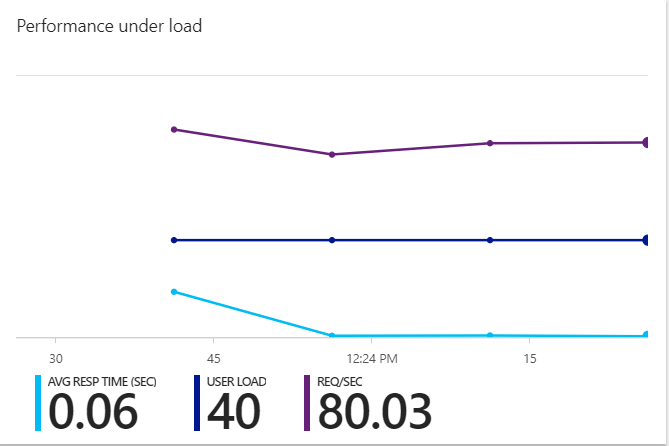
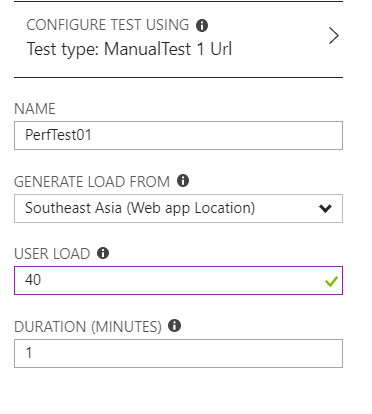


Table F

The Table F above shows the output of performance testing on different standard plans with different amount of user load in 1 minute with 160 users increment.

## Performance Testing Discussion

From the Table F, it clearly indicates that the performance improve as the Service Plan goes higher. It can be said that the better tier of Service Plan has better control of the concurrent request from the users. Regarding to the performance testing result, when running the test on 520 users, the Standard 2 Service Plan shows a crucial improvement over Standard 1 on the response time and less failure. However, Standard 3 Plan does not show significant improvement over Standard 2, meanwhile increase in failure. This can be due to the maximum acceptable threshold for the user load at the specific timing. To summarize, it defines that the Standard 1 Plan can hold up to 360 concurrent users for good performance while Standard 2 and Standard 3 Plans can manage up to 220 concurrent users. Therefore, it is important to consider the amount of concurrent user to access the website before choosing the subscription plan. It is the practice to reduce operation cost and meanwhile make use of the available plan for the best output.

# Conclusion

Throughout building up the system and conducting the Maersk Line project, a lot of insights and understanding has been gained by the project developer. For instances, there is a need for the developer to acquire latest information from Microsoft in order to keep the system up to date, most of the versions of the project references like jQuery, Bootstrap and JavaScript have to be compliable to the platform used for developing the system and the Microsoft Azure Documentation has to be referred for correct system deploying. The system developer has managed to overcome the limitation of knowledges through online learning and articles related to the programming language and Azure SQL database setup for the system. Secondly, the developer comes with basic skills in designing with HTML, CSS and Razor View. However, the developer is keened to improve the designing skills by studying the designs used on the latest websites. Moreover, the platform chosen, Microsoft ASP.NET MVC 5 is considered a new technology used to develop website recently. It is difficult for the developer to look for the sources or solution when encounter errors such as the system has failed to compile, the target page is not redirected correctly, the value inserted by the users are not passed into the controller and so on. As an alternative solution to the errors encountered, the developer has utilized the use of JavaScript in solving these issues. In the essence of manipulating the problems face, the developer must be determined and analytical to figure out the actual sources to cause the errors.

In a nutshell of completing this project, there are numerous vital elements have to be considered for constructing the system, Container Management System (CMS). The aspects of technology skills and analytical thinking the developer possess is very limited from the start of this project. Meanwhile, after going through several crucial researches and the study of Maersk Line operations as well as journal relating to vessel schedule management, the developer has sprouted out the ways to make a successful system. For instances, the requirements and functionalities of the system, the need of technology for the system and the public acceptance of the system. The researcher has also investigated a lot of academic explorations on the key areas of managing event and come out with a conclusion that having CMS will improve the situation of current trends of Maersk Line to manage the daily operations. With the useful findings and knowledge, this proposed system has been developed and deployed online with appropriate guidance from the lecturer, Dr. Kalai Anand Ratnamand mastering skills for developing this system. A functional CMS has been deployed for the public use and can be accessed through: https://cruzddac2018.azurewebsites.net/.

To summarize, the developer has done sufficient researches on completing this assignment. This assignment has allowed the developer to generate useful idea by analyzing the Maersk Line needs and the ways to improve the current situation of the business operation. Although it is challenging, the researcher has referred a lot of journal and online resources for the need of the proposed system, Container Management System, (CMS). It is strongly believed that having CMS for the Maersk will improve the quality of Schedule and Agent management as well as increase the awareness of beneficial Vessel Booking.

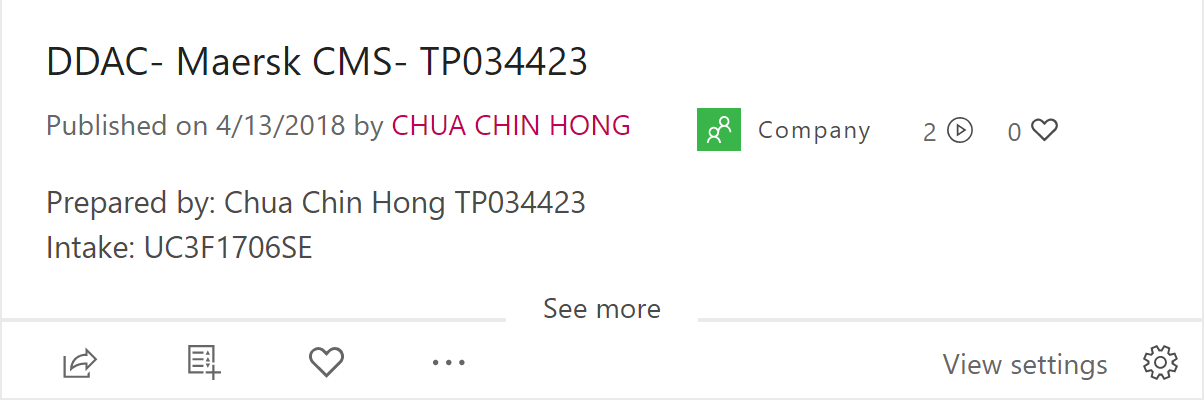
# References

1. Microsoft Azure, 2018, Azure Autoscale, [ONLINE] Available at: <https://azure.microsoft.com/en-us/features/autoscale/>. [Accessed 28 February 2018].
2. Amazon, 2012, What are AWS Cloud Design Patterns?. [ONLINE] Available at: <http://en.clouddesignpattern.org/index.php/Main_Page>. [Accessed 6 March 2018].
3. TheEconomicTimes, 2018, What are AWS Cloud Design Patterns?. [ONLINE] Available at: <https://economictimes.indiatimes.com/definition/unit-testing>. [Accessed 8 March 2018].

# Appendices

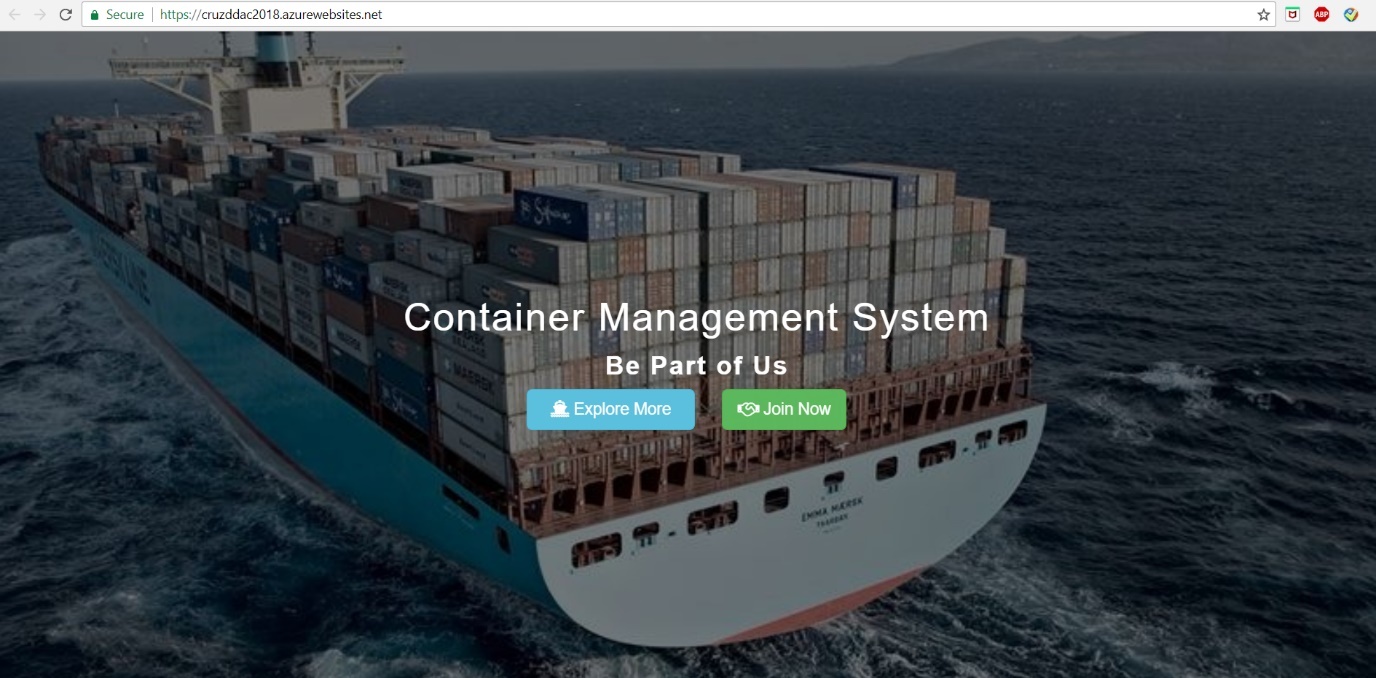
1. Microsoft Stream Link:

**https://web.microsoftstream.com/video/53eaade7-e35c-4591-aaef-23d2776ad576?list=studio**



1. Container Management System Link:

**https://cruzddac2018.azurewebsites.net/**



1. GitHub Link:

**https://github.com/cruzlegacy/cruzDDAC**