

­­­­

|  |  |
| --- | --- |
| Assignment Details | |
| Module | Designing & Developing Applications on the Cloud (CT071-3-5-3-DDAC) |
| Lecturer | Dr. Kalai Anand Ratnam |
| Intake | UC3F1706SE |
| Student Name | Erick |
| Student ID | TP033456 |
| Submission Date | 13/04/2018 |

CONTAINER MANAGEMENT SYSTEM

# Acknowledgment

First of all, I would like to thank to God for helping me to complete this Designing & Developing Applications on the Cloud assignment. By having a faith in God, I am able to finish this assignment.

After that, I would like to thank to my parents and my family for supporting me from the back. With their suggestion and support when I have a problem, I can finish this assignment.

Then, I would like to my lecturer, Dr. Kalai Anand Ratnam for guiding me on how to use this Microsoft Azure, how to build the cloud database and how to upload it. Without his guidance, I could not able to finish this assignment.

Furthermore, I would like to thank to my seniors and friends for helping me in this assignment, especially Xan and Tommy in helping me building and uploading this assignment to the Microsoft Azure.

Lastly, I would like to thank to Google for providing reliable source for my research, Starbuck for the good coffee, place and Wi – Fi for doing this assignment, Hardwell for hosting his radio show on Youtube so that I can listen to his music and his playlist and lastly, PLAY Club for a place to refresh my mind and celebrating my successful on uploading the assignment to Microsoft Azure.

Table of Contents

[Acknowledgment 2](#_Toc511228637)

[1.0 Introduction 5](#_Toc511228638)

[1.1 Background 5](#_Toc511228639)

[1.2 Objectives 6](#_Toc511228640)

[1.3 Scope 6](#_Toc511228641)

[1.4 Specification 6](#_Toc511228642)

[1.5 Deliverables 7](#_Toc511228643)

[2.0 Project Plan 8](#_Toc511228644)

[2.1 Gantt Chart 8](#_Toc511228645)

[3.0 Design 9](#_Toc511228646)

[3.1 Architectural Design 9](#_Toc511228647)

[3.2 Design Consideration 10](#_Toc511228648)

[3.3 Modelling 11](#_Toc511228649)

[3.3.1 Use Case Diagram 11](#_Toc511228650)

[3.3.2 Use Case Specification 12](#_Toc511228651)

[3.3.3 Activity Diagram 16](#_Toc511228652)

[3.3.4 Class Diagram 20](#_Toc511228653)

[3.3.5 Sequence Diagram 21](#_Toc511228654)

[4.0 Implementation 25](#_Toc511228655)

[4.1 Create the web app 25](#_Toc511228656)

[4.2 Creating SQL Database 28](#_Toc511228657)

[4.3 Deploying SQL Database to Azure 31](#_Toc511228658)

[4.4 Traffic Manager 32](#_Toc511228659)

[4.5 Deploying PHP File to Azure 37](#_Toc511228660)

[4.6 Auto Scaling 38](#_Toc511228661)

[4.7 Screenshots 40](#_Toc511228662)

[4.7.1 Homepage 40](#_Toc511228663)

[4.7.2 About 40](#_Toc511228664)

[4.7.3 Contact 41](#_Toc511228665)

[4.7.4 Login 41](#_Toc511228666)

[4.7.5 Cargo 42](#_Toc511228667)

[4.7.6 Create Cargo 43](#_Toc511228668)

[4.7.7 Update Cargo 44](#_Toc511228669)

[4.7.8 Delete Cargo 45](#_Toc511228670)

[4.7.9 Vessel 46](#_Toc511228671)

[4.7.10 Create Vessel 47](#_Toc511228672)

[4.7.11 Update Vessel 48](#_Toc511228673)

[4.7.12 Delete Vessel 49](#_Toc511228674)

[4.7.13 Shipment 50](#_Toc511228675)

[4.7.14 Create Shipment 51](#_Toc511228676)

[4.7.15 Update Shipment 52](#_Toc511228677)

[4.7.16 Delete Shipment 53](#_Toc511228678)

[4.7.17 Logout 54](#_Toc511228679)

[5.0 Test Plan 55](#_Toc511228680)

[5.1 Unit Testing 55](#_Toc511228681)

[5.1.1 Login Testing 55](#_Toc511228682)

[5.1.2 Create Cargo Testing 56](#_Toc511228683)

[5.1.3 Update Cargo Testing 57](#_Toc511228684)

[5.1.4 Create Vessel Testing 58](#_Toc511228685)

[5.1.5 Update Vessel Testing 58](#_Toc511228686)

[5.1.6 Create Shipment Testing 59](#_Toc511228687)

[5.1.7 Create Shipment Testing 60](#_Toc511228688)

[5.2 Performance Testing 61](#_Toc511228689)

[6.0 Conclusion 62](#_Toc511228690)

[7.0 References 63](#_Toc511228691)

[8.0 Appendix (GitHub & Video) 64](#_Toc511228692)

# 1.0 Introduction

## 1.1 Background

Maersk Line is the global container division and the largest operating unit of the A.P. Moller – Maersk Group, a Danish business conglomerate. It is the world's largest container shipping company having customers through 374 offices in 116 countries. It employs approximately 7,000 sea farers and approximately 25,000 land-based people. Maersk Line operates over 600 vessels and has a capacity of 2.6 million TEU. The company was founded in 1928.

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

“There was a ‘mind-opener’ where Maersk said, ‘How can we support the overall business strategy, and also 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.”

In the effort to support further business growth and increase organizational flexibility, Maersk decided to consolidate all of its data centers and server rooms operating worldwide onto a virtualized platform. Microsoft Azure was already hosting some of Maersk’s IT environment, and in March 2016 Maersk initially approached Microsoft about expanding the scope of the relationship. Moving forward, Lorenzen says Maersk is currently changing over its IT setup based on Microsoft Azure, starting with the desktop environment up to container

## 1.2 Objectives

The objective of this project is to build a HTML web application with PHP code for the back end and deploy this web application to the cloud using the Azure Cloud Services in order for the application to be used around the world

## 1.3 Scope

This web application will be used by the agent for managing the container. The application will allow the agent for managing:

* containers,
* vessels
* shipments

## 1.4 Specification

The following provides a list of the most significant goals and requirements:

* **Provisioning**: You must be able to provision the new application to the Microsoft Azure Platform.
* **Maintainability**: You must be able to upgrade the application and perform other maintenance tasks while multiple tenants are using it.
* **Monitoring**: You must be able to monitor the application at all times to identify any problems and to troubleshoot them. This includes monitoring how each tenant is using the application.
* **Availability**: Tenants want the application to be constantly available, perhaps with guarantees defined in an SLA. Again, the activities of other tenants should not affect the availability of the application.
* **Scalability**: The application scales to meet the demand of the application.

## 1.5 Deliverables

This web application is built for the agent to manage the container. Therefore, the agent able to manage:

|  |  |
| --- | --- |
| **Entity** | **Actions** |
| Cargo | Create / Read / Update / Delete |
| Vessel | Create / Read / Update / Delete |
| Shipment | Create / Read / Update / Delete |

# 2.0 Project Plan

## 2.1 Gantt Chart

Figure 1.0 Gantt Chart

# 3.0 Design

## 3.1 Architectural Design

The azure website supports many types of the web application like the ASP.Net, the PHP and the Python (Microsoft, 2018). It also allows the user to create the mobile apps, games, the E – Commerce and many more (Microsoft, 2018). No matter what web application is built by the user, it still going to work in Azure when it is deployed. Below is the picture of the architectural design.

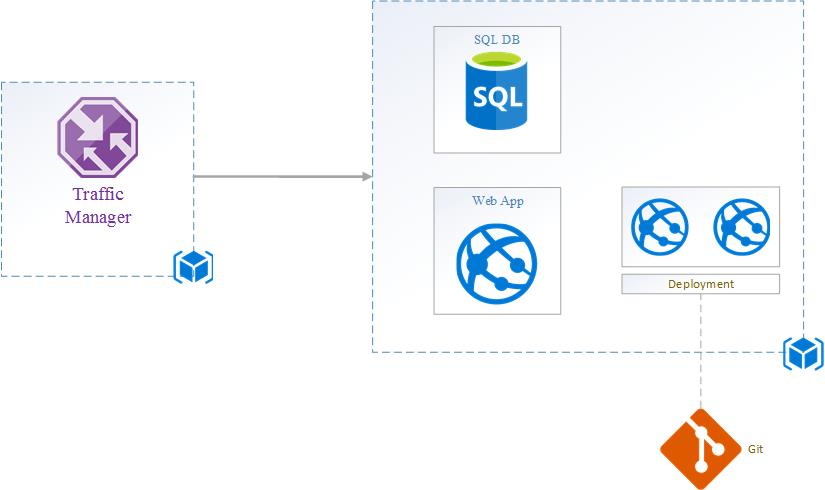


Figure 2.0 Azure Web Architecture

## 3.2 Design Consideration

Due to the lack of the specification of the system, the developer decided to make assumption that the web application was only limited to the function like managing cargos, managing vessels and managing shipments. After that, the developer decided to use PHP as the programming language to create this web application.

The domain name that was chosen by the developer for the web application that was located on South East Asia was ddac619.azurewebsites.net. The developer decided to keep the web application simple and clean in order for improving the users’ experience and allowing the users to use application straight forward without any problem. By doing so, it could achieve the high productivity when the users used the application. Furthermore, the consistency of the design also applied in this web application so that the users were able to explore the web application freely without any problem.

The functionalities of the system would be managing the cargos, managing the vessels and managing the shipments. When the users wanted to try to manage the cargos, manage the vessels and manage the shipment, the users had to login first. This was to prevent the unauthorized users to use the web application and this application and this application was design for the agent only. Once the agent login to the web application, the agent able to manage the cargo, manage the vessels and manage the shipment.

## 3.3 Modelling

### C:\Users\Owner\AppData\Local\Microsoft\Windows\INetCache\Content.Word\DDACUseCase.jpg3.3.1 Use Case Diagram

Figure 3.0 Use Case Diagram of the Web Application

The Figure above shows about use case diagram of the web application. In this application, the agent able to login and logout on the system, manage cargo, vessel and shipment.

### 3.3.2 Use Case Specification

|  |  |
| --- | --- |
| **Name** | **Login** |
| Actor | Agent |
| Priority | High |
| Description | This is where the agent login into the system |
| Extends | Logout |
| Includes | None |
| Pre-condition | The user has to be the Azure Active Directory user |
| Post-condition | Proceed to main menu |
| Flow of Events | 1. Enter username 2. Enter password 3. If username and password is invalid [A1] |
| Alternative Flows | A1. The user has to enter the username and password again |

Table 1.0 Login Use Case Specification

|  |  |
| --- | --- |
| **Name** | **Manage Cargos** |
| Actor | Agent |
| Priority | High |
| Description | View / Create / Update / Delete cargos |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | The user has to login first |
| Post-condition | Proceed to cargo page |
| Flow of Events | 1. View the list of the cargo from database 2. If there is no cargo in the database 3. Add new cargo into database 4. Enter cargo ID 5. Enter cargo name 6. Enter cargo length 7. Enter cargo width 8. Enter cargo weight 9. Enter cargo status 10. If the details are invalid [A1] 11. Save the cargo details 12. If user wishes to update cargo 13. Select the particular cargo 14. Enter new cargo name 15. Enter new cargo length 16. Enter new cargo width 17. Enter new cargo weight 18. Enter new cargo status 19. If the details are invalid [A1] 20. Save new cargo details 21. If user wishes to delete cargo 22. Select the particular cargo 23. Delete the cargo |
| Alternative Flows | A1. User has to re – enter the details again |

Table 2.0 Manage Cargos Use Case Specification

|  |  |
| --- | --- |
| **Name** | **Manage Vessels** |
| Actor | Agent |
| Priority | High |
| Description | View / Create / Update / Delete vessels |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | The user has to login first |
| Post-condition | Proceed to vessel page |
| Flow of Events | 1. View the list of the vessel from database 2. If there is no vessel in the database 3. Add new vessel into database 4. Enter vessel ID 5. Enter vessel name 6. If the details are invalid [A1] 7. Save the vessel details 8. If user wishes to update vessel 9. Select the particular vessel 10. Enter new vessel name 11. If the details are invalid [A1] 12. Save new vessel details 13. If user wishes to delete vessel 14. Select the particular vessel 15. Delete the vessel |
| Alternative Flows | A1. User has to re – enter the details again |

Table 3.0 Manage Vessels Use Case Specification

|  |  |
| --- | --- |
| **Name** | **Manage Shipment** |
| Actor | Agent |
| Priority | High |
| Description | View / Create / Update / Delete shipment |
| Extends | None |
| Includes | None |
| Pre-condition / Assumptions | The user has to login first |
| Post-condition | Proceed to shipment page |
| Flow of Events | 1. View the list of the shipment from database 2. If there is no shipment in the database 3. Add new shipment into database 4. Enter shipment ID 5. Enter vessel name 6. Enter cargo name 7. Select the date 8. If the details are invalid [A1] 9. Save the shipment details 10. If user wishes to update shipment 11. Select the particular shipment 12. Enter new vessel name 13. Enter new cargo name 14. Select new data 15. If the details are invalid [A1] 16. Save new shipment details 17. If user wishes to delete shipment 18. Select the particular shipment 19. Delete the shipment |
| Alternative Flows | A1. User has to re – enter the details again |

Table 4.0 Manage Shipment Use Case Specification

### C:\Users\Owner\AppData\Local\Microsoft\Windows\INetCache\Content.Word\DDACLoginActivityDiagram.png3.3.3 Activity Diagram

Figure 4.0 Login Activity Diagram

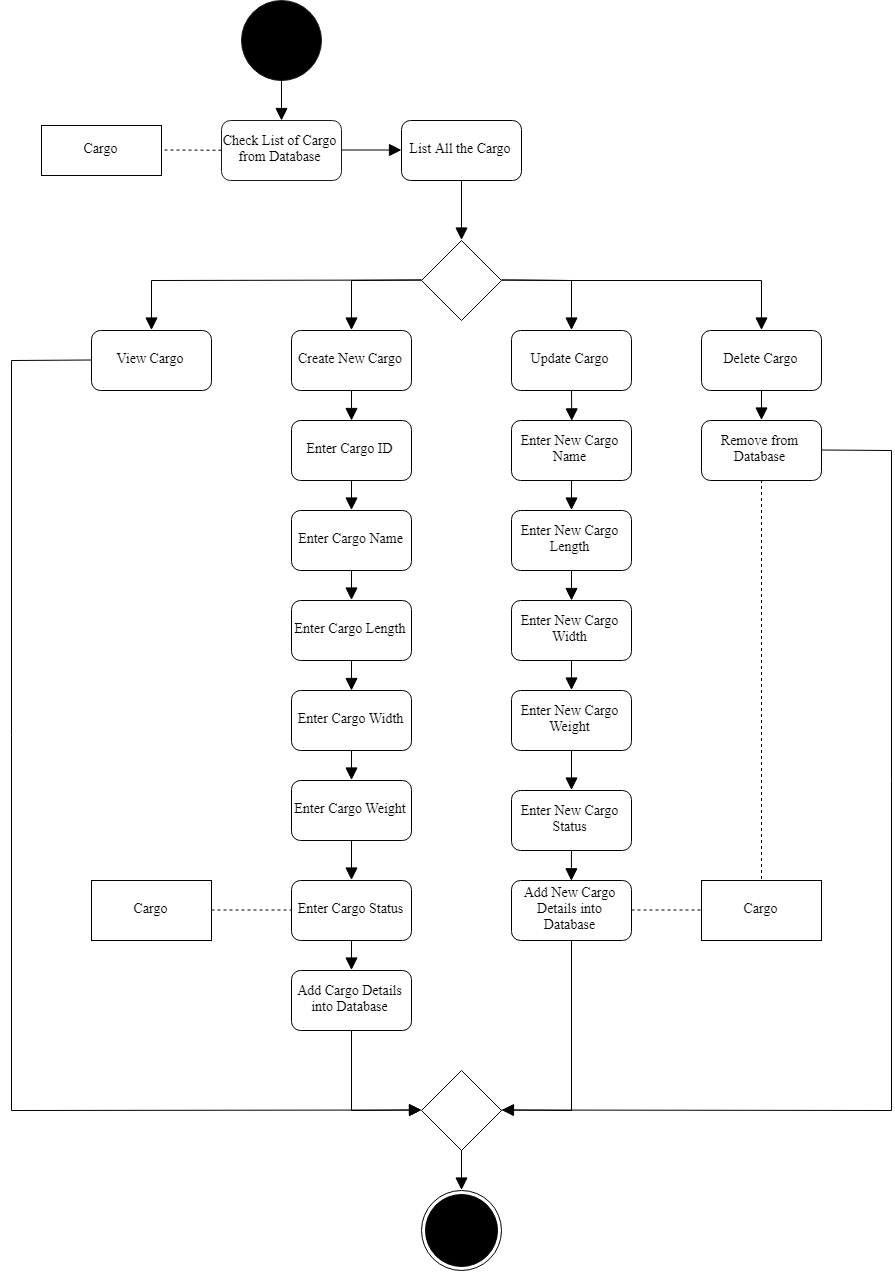


Figure 5.0 Manage Cargo Activity Diagram

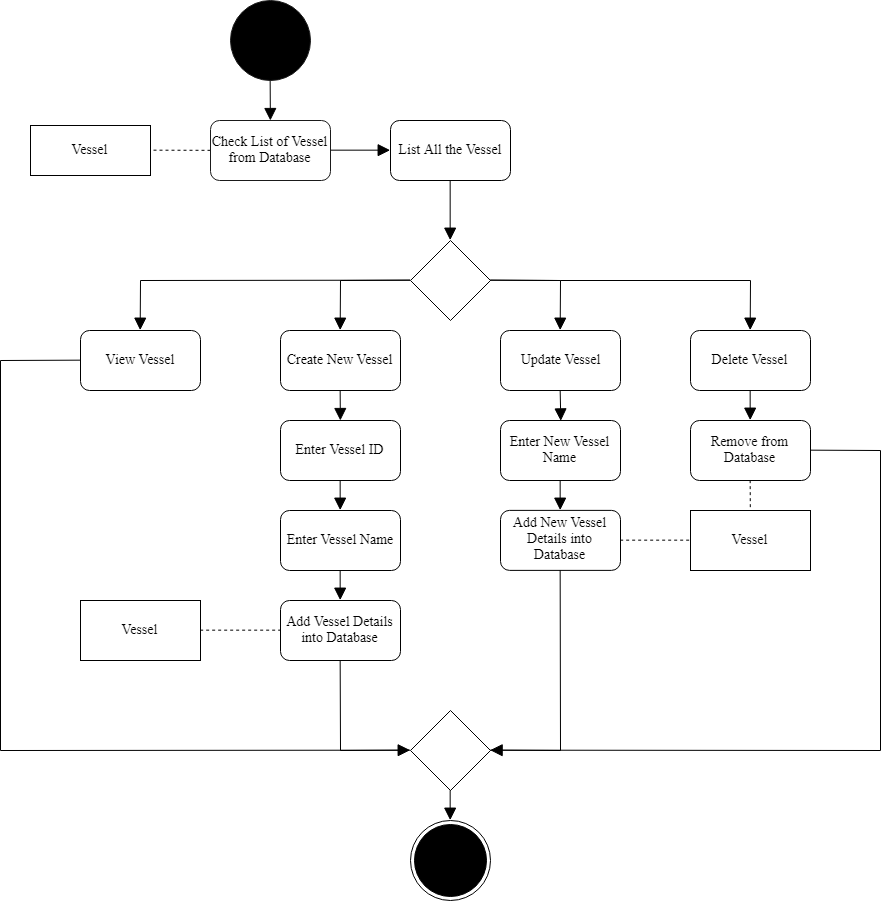


Figure 6.0 Manage Vessel Activity Diagram

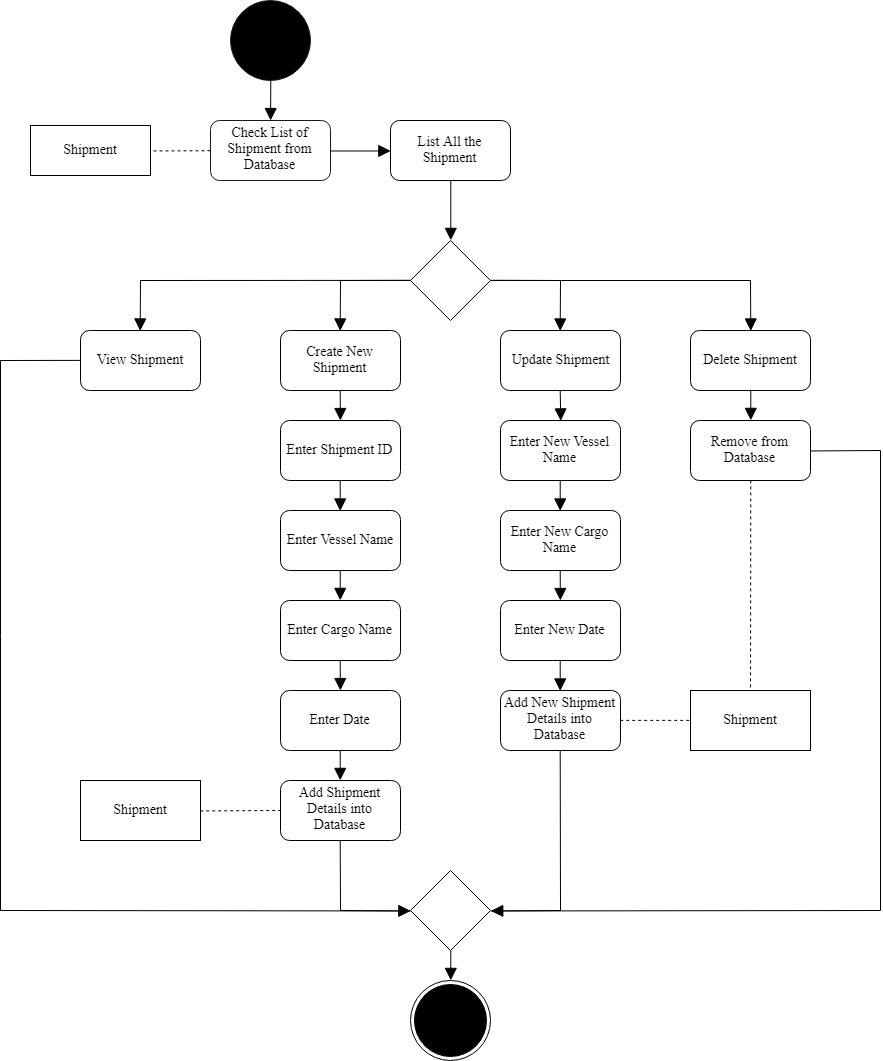


Figure 7.0 Manage Shipment Activity Diagram

### 3.3.4 Class Diagram

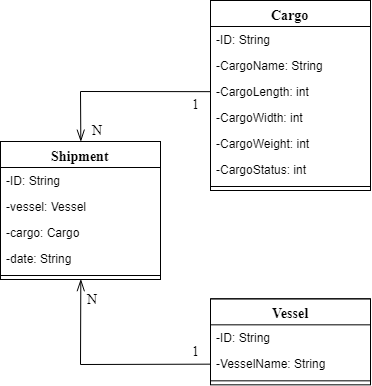


Figure 8.0 Class Diagram

### 3.3.5 Sequence Diagram

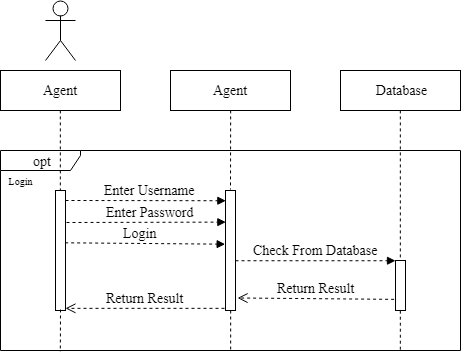


Figure 9.0 Login Sequence Diagram

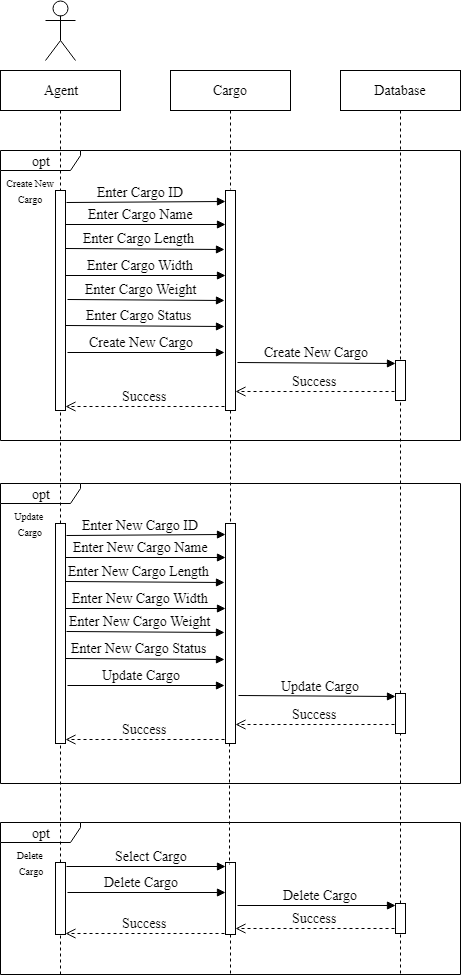


Figure 10.0 Manage Cargo Sequence Diagram

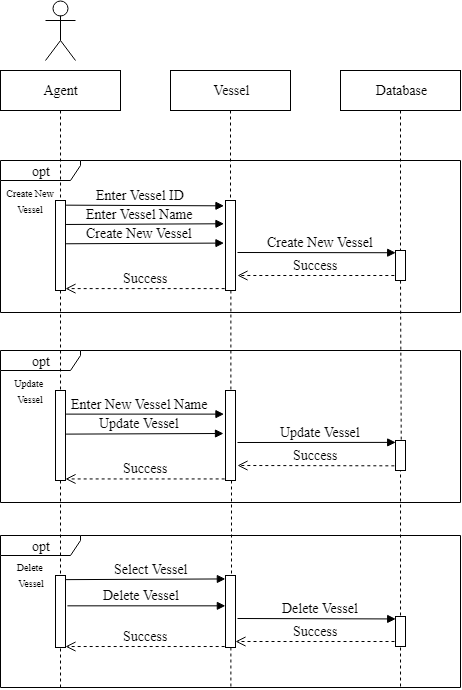


Figure 11.0 Manage Vessel Sequence Diagram

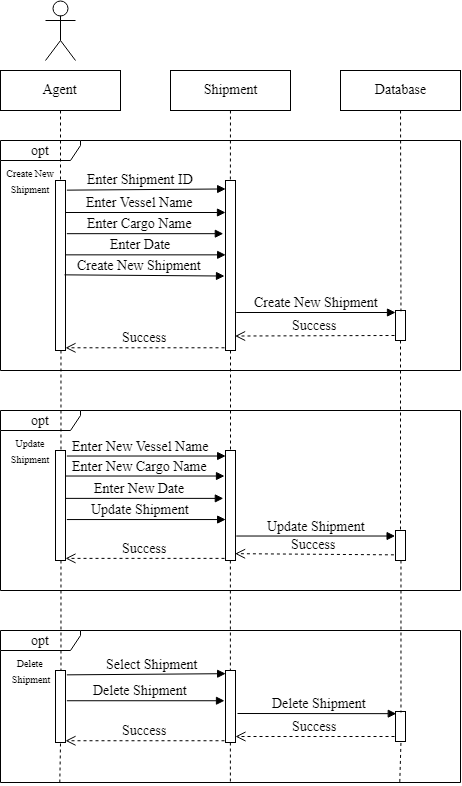


Figure 12.0 Manage Shipment Sequence Diagram

# 4.0 Implementation

## 4.1 Create the web app

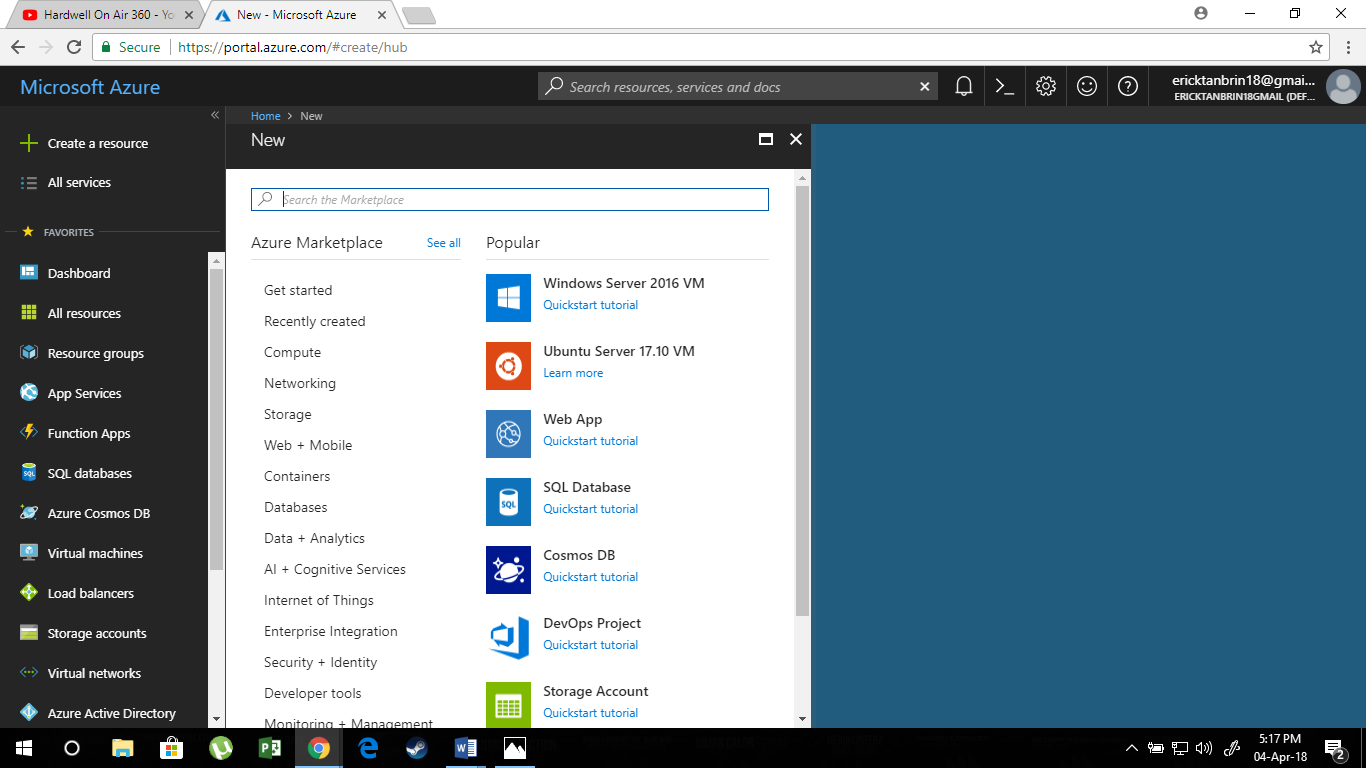


Figure 13.0 Creating New Web App

The Figure above shows about creating new app. First of all, click the “Create a resource” and then choose the “Web App”.

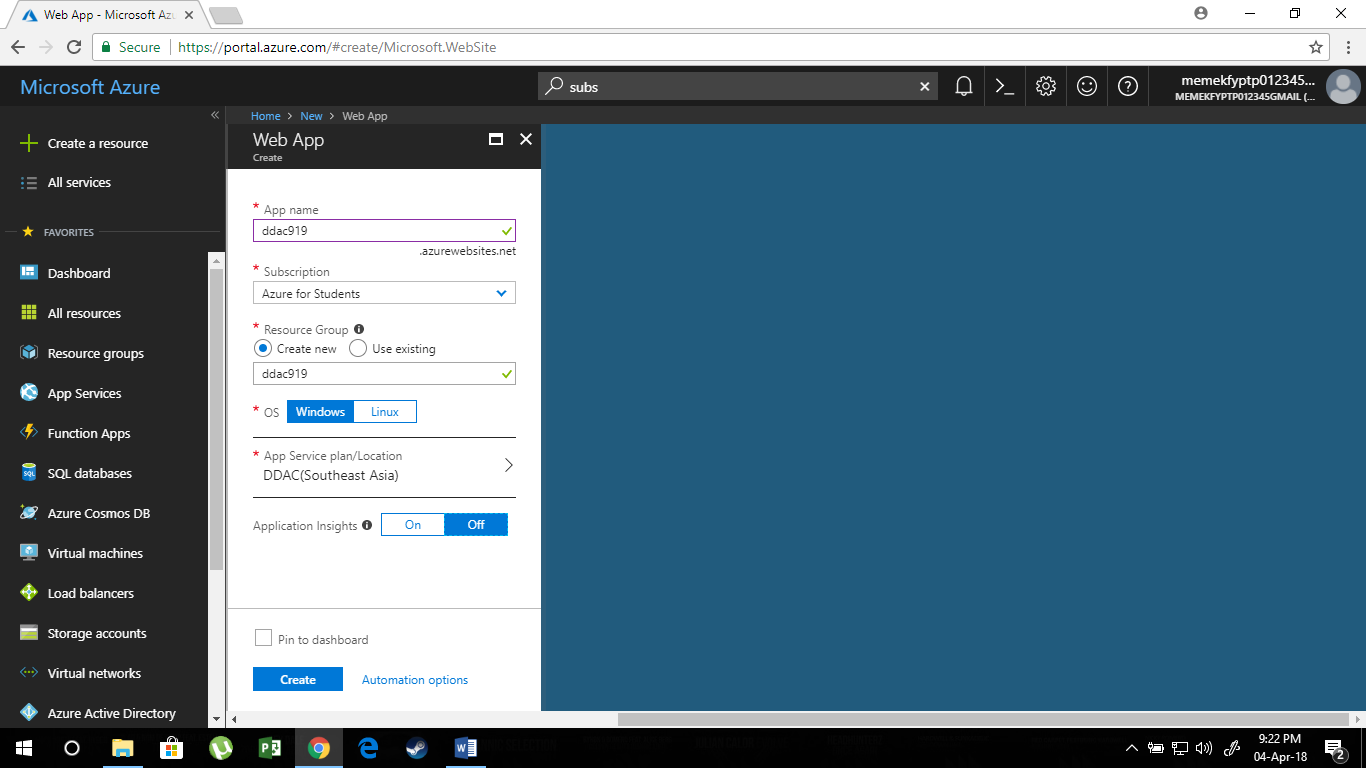


Figure 14.0 Result After Clicking Web App

The Figure above the result after clicking the web app. Once the web app is clicked, enter the app name, which is ddac919 in this scenario. After that, choose the subscription, which is Azure for Student. After that, create the resource group, where in this case is ddac919. After that, choose the OS, which is windows in this scenario. Then, choose the App Service Plan/Location, which is South East Asia in this scenario. The Figure 14.0 will show about how to select the app service plan/location.

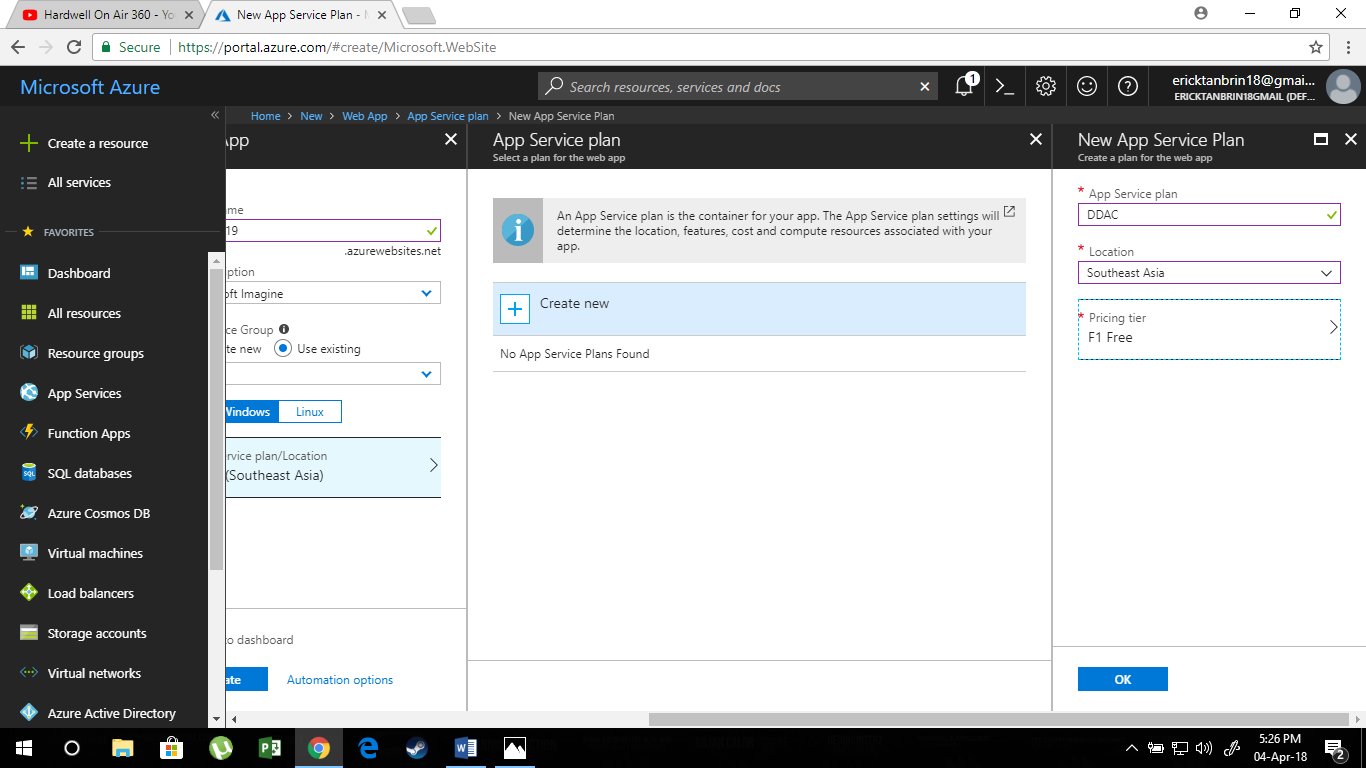


Figure 15.0 How to Create App Service Plan/Location

The Figure above shows about how to create app service plan/location. First of all, click the “>” button. After that, click the “Create New”. After that, fill in the information. In this scenario, the app service plan name will be DDAC, the location will be in South East Asia and the pricing tier will be F1 Free. Once it is done, click the “OK” button. After that, click the “Create” button to create the web application.

## 4.2 Creating SQL Database

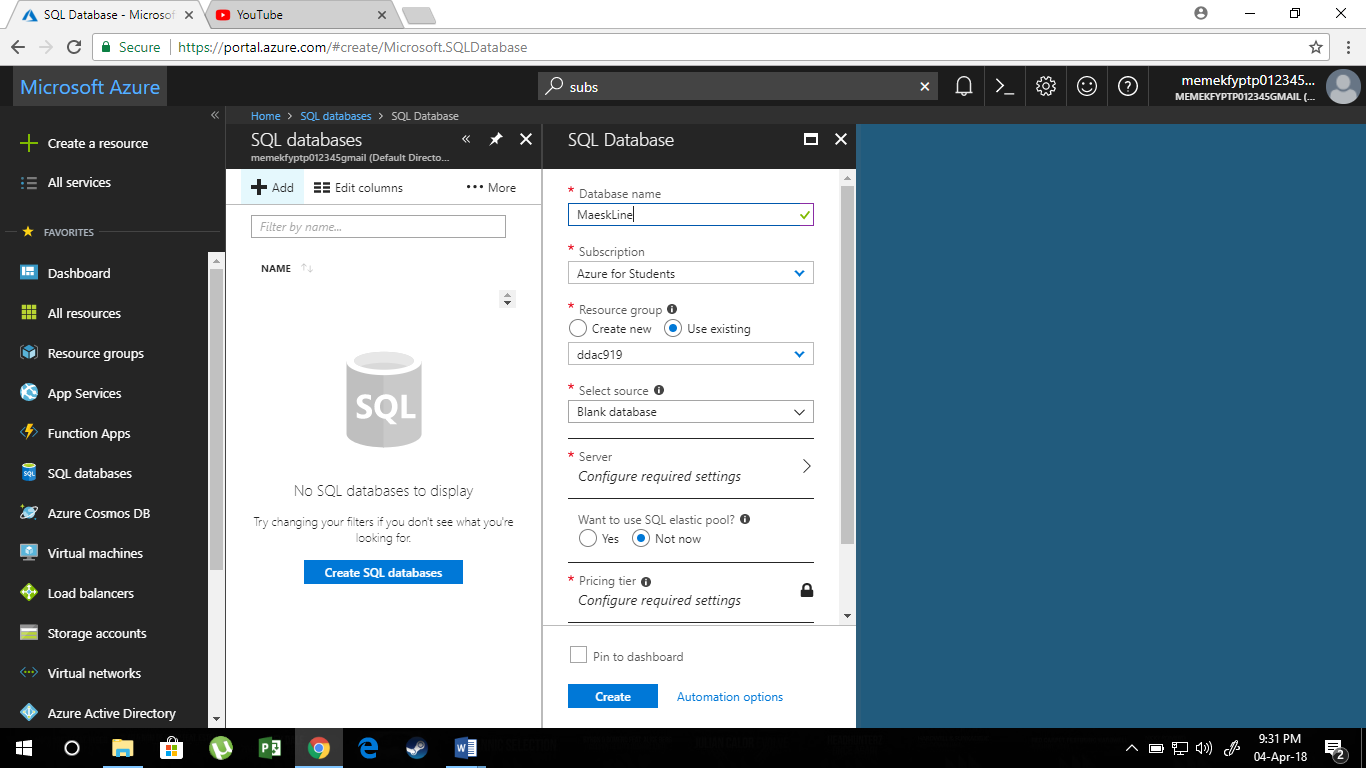


Figure 16.0 Creating SQL Database

The Figure above shows about creating new SQL database. First, press the “SQL database”. After that, click the “Add” button. Then, enter the database name which is in this case is “MaerskLine”. After that, choose the subscription, which is in this case is Azure for Students. After that, choose “use existing” as the resource group and select ddac919 in this case. After that, select Blank Database for the select source. For the server configuration will be shown in the Figure 16.0

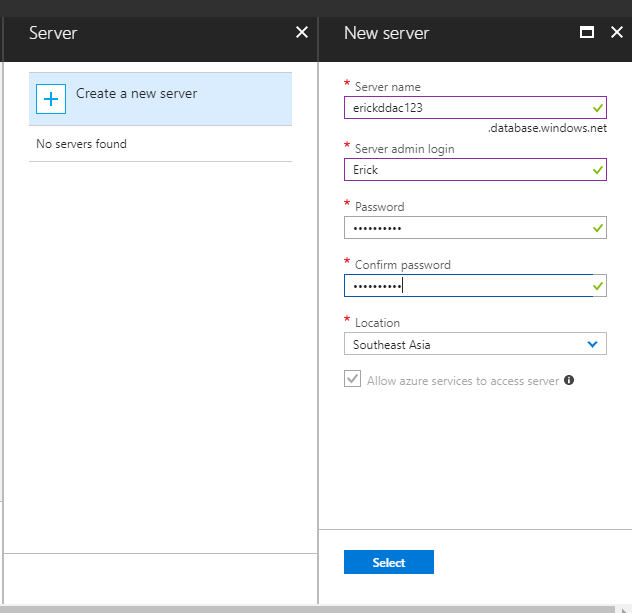


Figure 17.0 Server Configuration

The Figure above shows about the server configuration. For the server name, it will be erickddac123. For the server admin login, it will be Erick. For the password and the confirm password will be 12321232Aq. For the location, it will be Southeast Asia. After that, press “select” button.

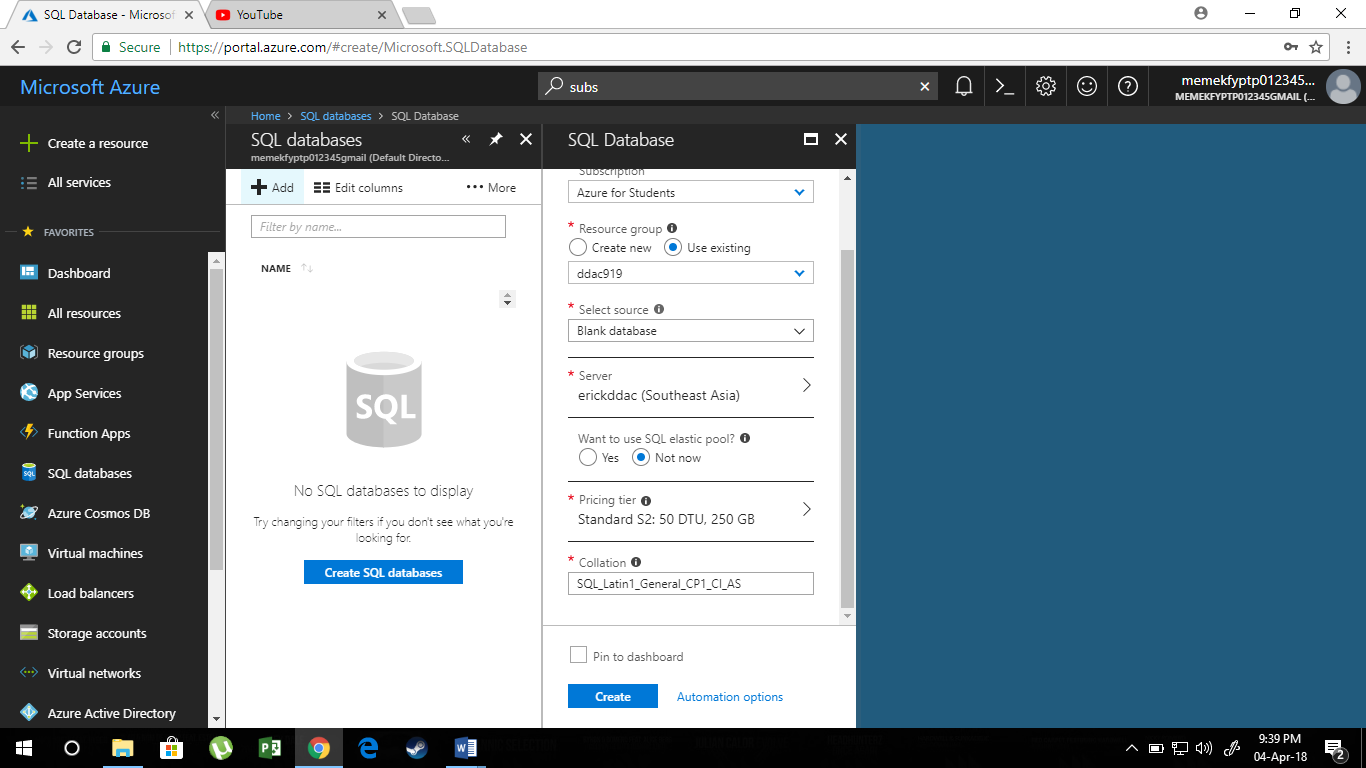


Figure 18.0 Creating SQL Database

The Figure above shows about the creating SQL Database. Once the server is already configured, choose “Not Now” for the want to user SQL elastic pool?. After that, for the pricing tier, choose Standard S2: 50 DTU, 250 GB. For the collation, enter SQL\_Latin1\_General\_CP1\_CI\_AS. After that, press the “Create” button.

## 4.3 Deploying SQL Database to Azure

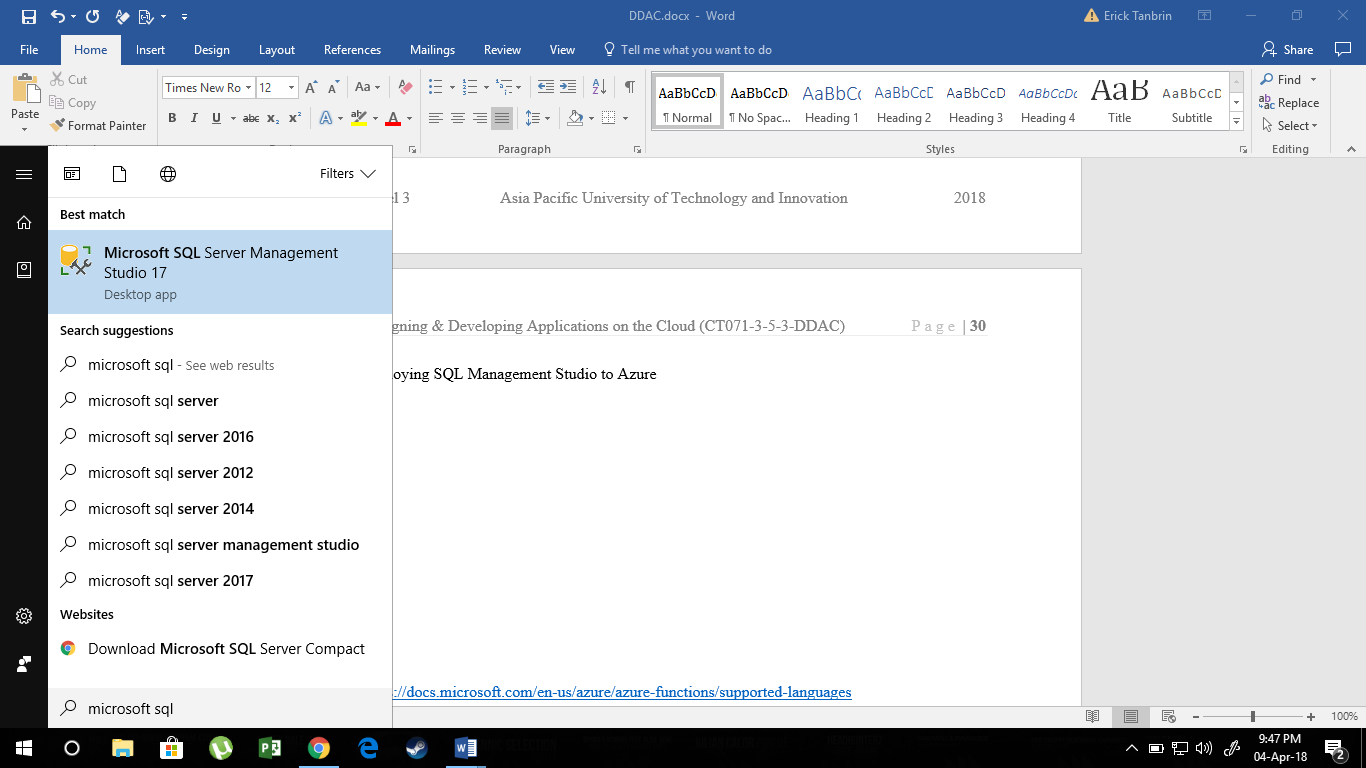


Figure 19.0 Deploying SQL Database to Azure

Figure above shows about deploying SQL database to azure. First, the Microsoft SQL Server Management Studio 17 has to be installed first. Then, run the application.

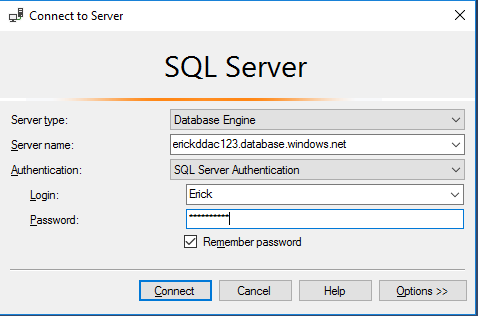


Figure 20.0 Entering the Data

The Figure above shows about entering the data. For the server name, enter erickddac123.database.windows.net, the login will be Erick and the password will be 12321232Aq. After that, you can start build the table for storing the information

## 4.4 Traffic Manager

Traffic manager allows the developer to take control on the distribution of the user traffic for the service endpoints in the different data centers (Microsoft, 2018). The service endpoints are supported by the Traffic Manager which include the Azure VMs, the Web Application and the Cloud Service (Microsoft, 2018). In order to create the traffic manager, the new resource group has to be created first for the traffic manager, just like shown in the Figure 18.0.

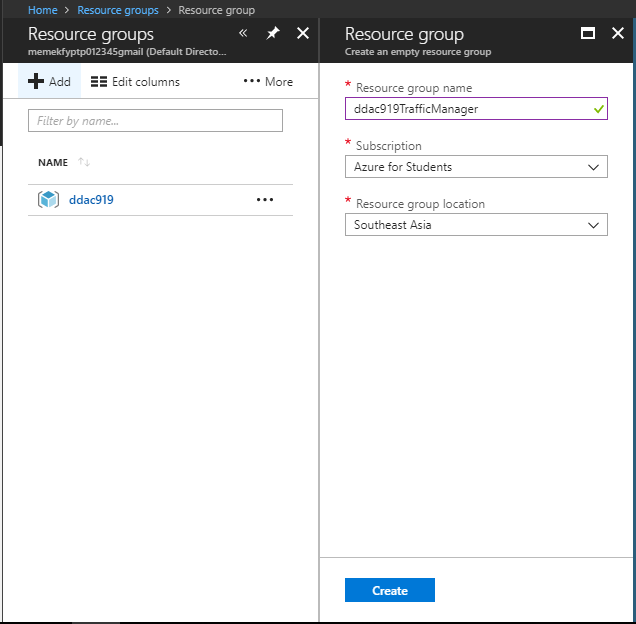


Figure 21.0 Creating New Resource Group for Traffic Manager

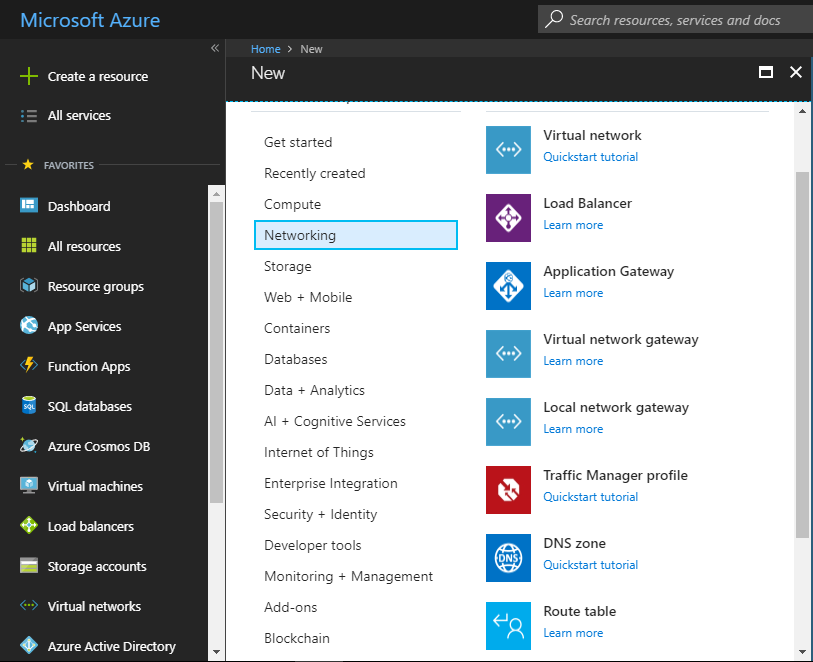


Figure 22.0 Creating Traffic Manager Profile

The Figure above shows about creating traffic manager profile. First, click “Create a resource”, then choose “Networking” and then choose the “Traffic Manager Profile”.

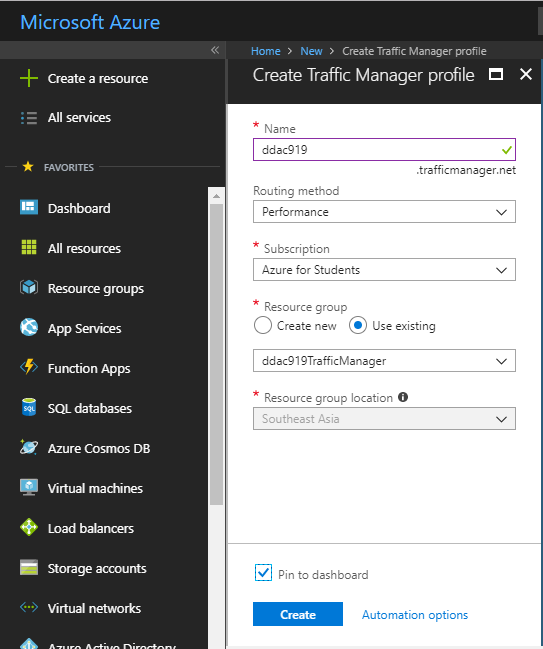


Figure 23.0 Creating Traffic Manager Profile

The Figure above shows about creating traffic manager profile. First, enter the name, which is ddac919 in this case. After that, the developer has to choose the routing method. There are 4 types of the routing method, which are Performance, Weighted, Priority and Geographic. In this case, Performance will be suitable and it will redirect the user to the endpoint which is close to the users geographically for reducing the network latency. After that, choose the resource group, which is existing and choose ddac919TrafficManager in this case. After that, press the “Create” button.

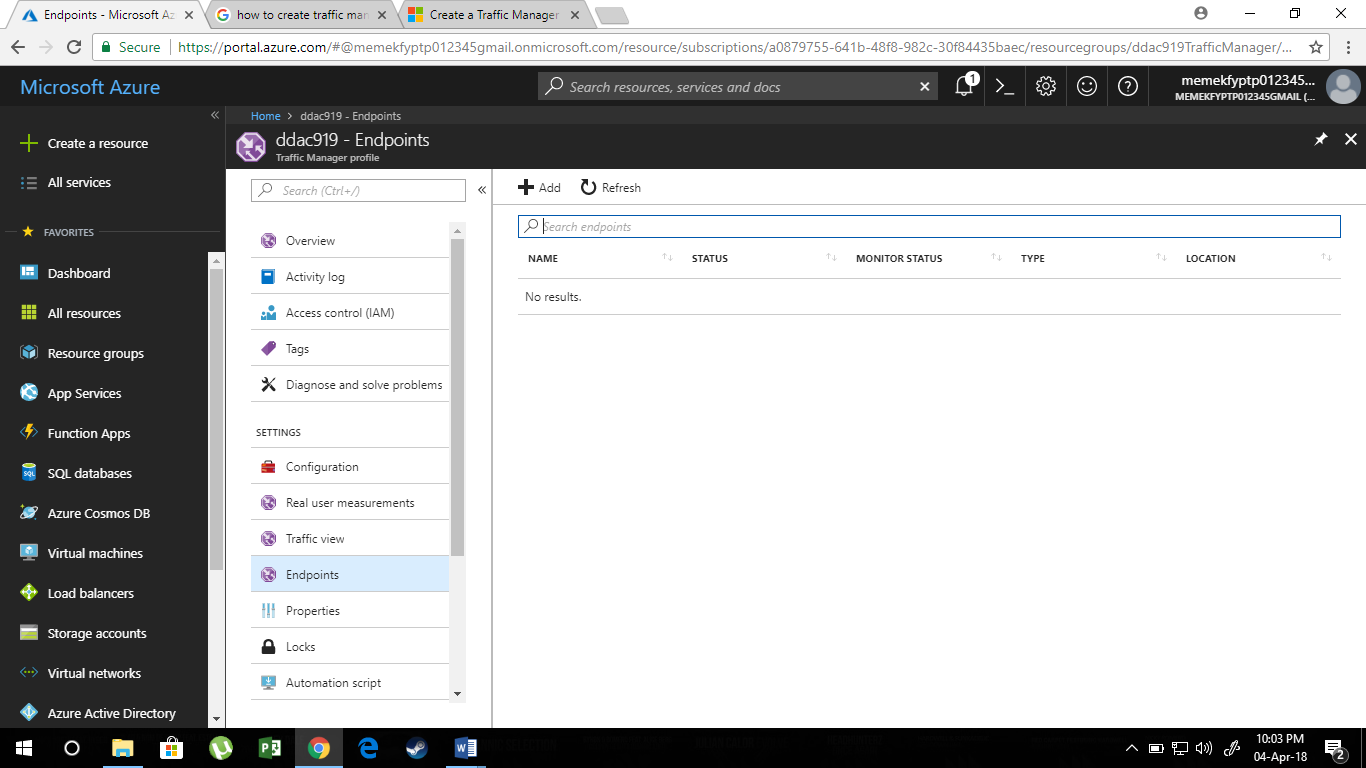


Figure 24.0 Result After Creating the Traffic Manager Profile

The Figure above shows about the result after creating the traffic manager profile. Once it is created, the end points have to be created. For creating the end points, press the “Endpoints” and then press the “Add” button.

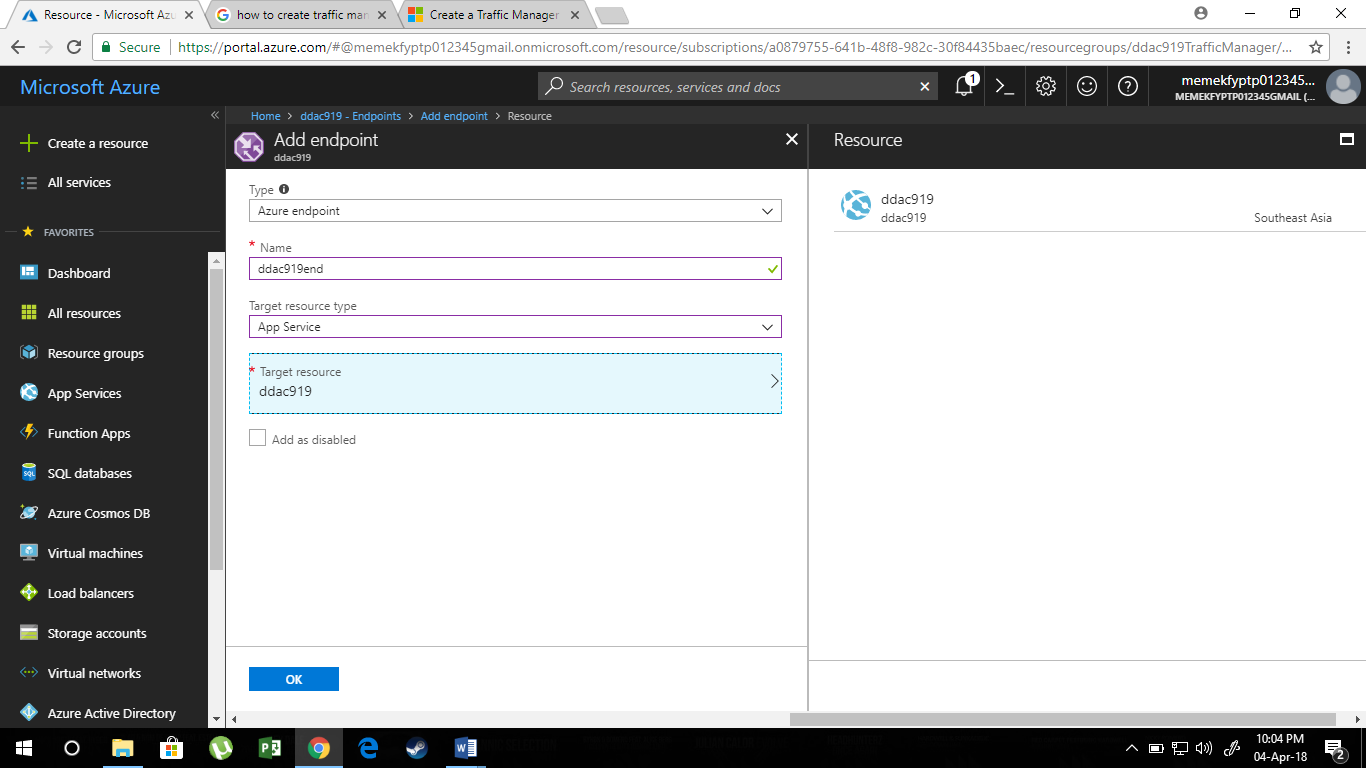


Figure 25.0 Result After Pressing the “Add” Button

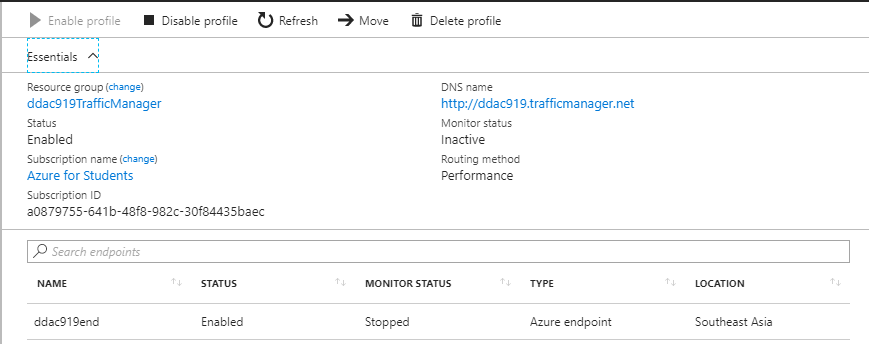
The Figure above shows about the result after pressing the “Add” button. First, choose the type, which is “Azure endpoint” in this case. Then, enter the name, which is ddac919end in this case. Then, choose the target resource type, which is app service in this case and choose the target resource, which is ddac919 in this case. After that, press the “OK” button and the result will be shown in Figure 21.0

Figure 26.0 Traffic Manager Overview

## 4.5 Deploying PHP File to Azure

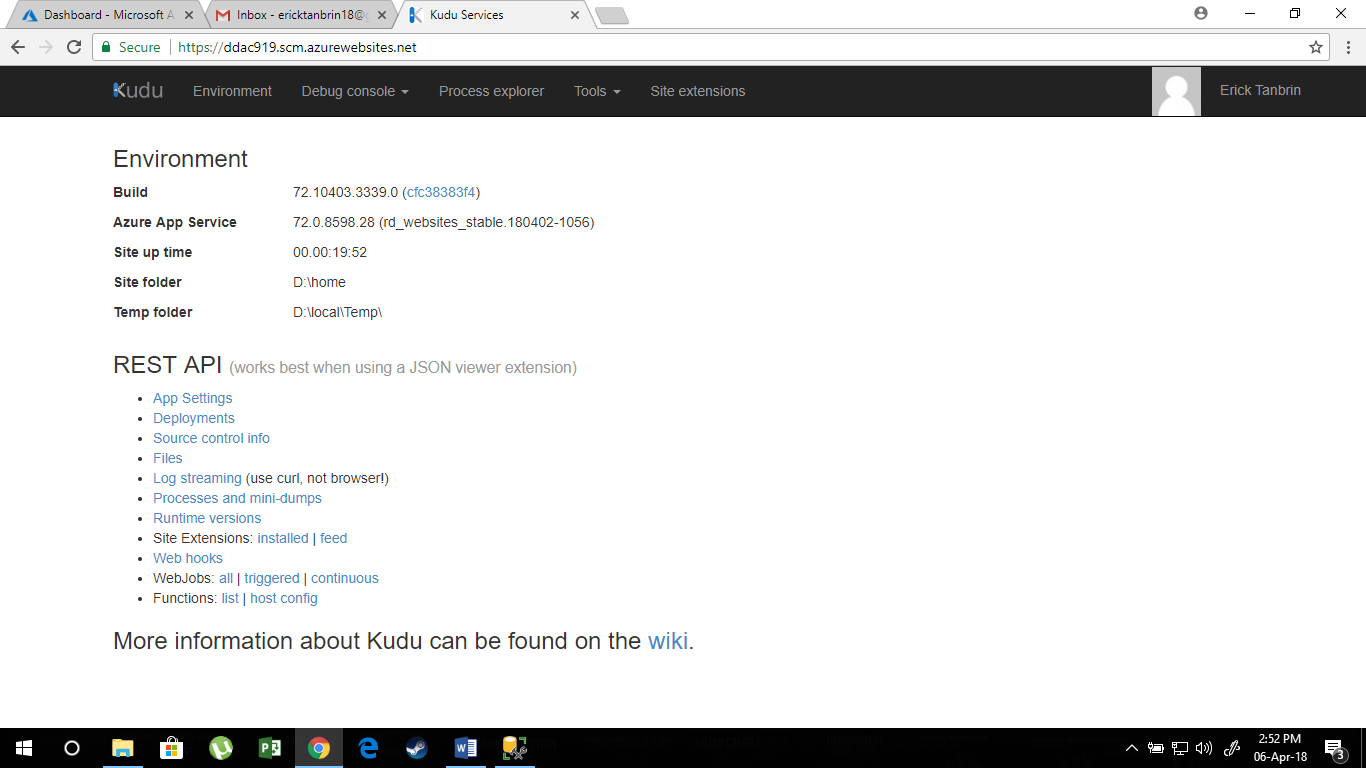


Figure 27.0 Deploying PHP File to Azure

The Figure above shows about deploying PHP file to azure. First, go to https://ddac919.scm.azurewebsites.net/ZipDeploy. After that, compress the PHP file into the zip and upload it, just like shown in Figure 28.0

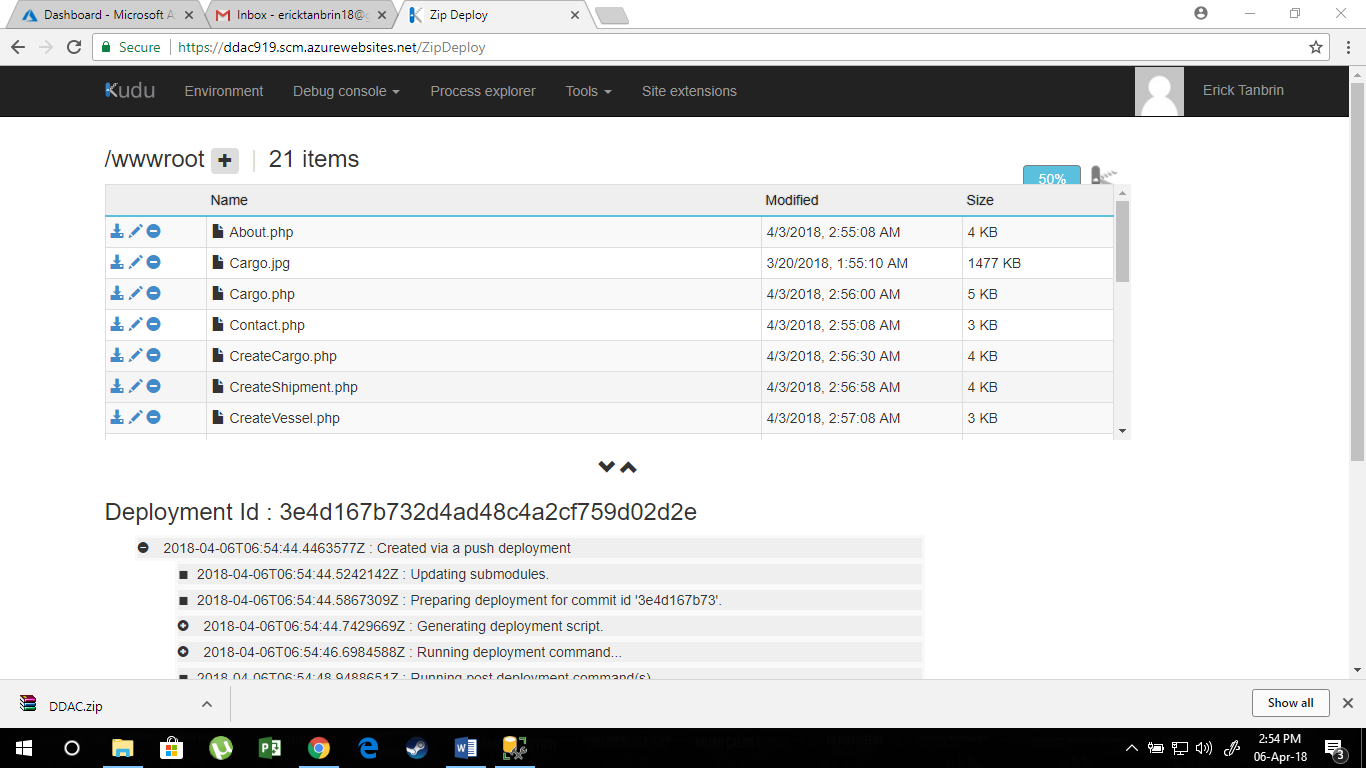


Figure 28.0 Upload PHP file to Azure

## 4.6 Auto Scaling

Since there were lots of users accessing the web application, the scaling process need to be automated. Azure provides the service where the web application can be auto scale up and scale down and it depends on the rules that are set by the developer. In this case, the developer decided to set the rule, which in CPU percentage where the CPU percentage was 70% for the 5 minutes. After that, the developer just had to enter the scale rule, name, number of the instance, maximum number of the instance and the percentage of the CPU. After that, the azure would start implementing this scaling process into the web application.

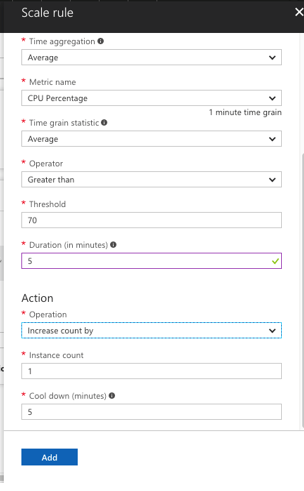


Figure 29.0 Scale Rule

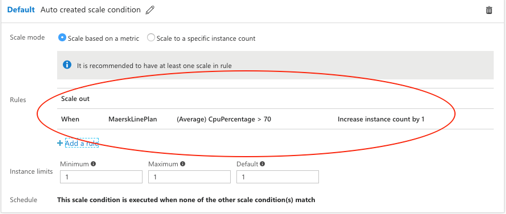


Figure 30.0 Auto Scaling Overview

## 4.7 Screenshots

### 4.7.1 Homepage

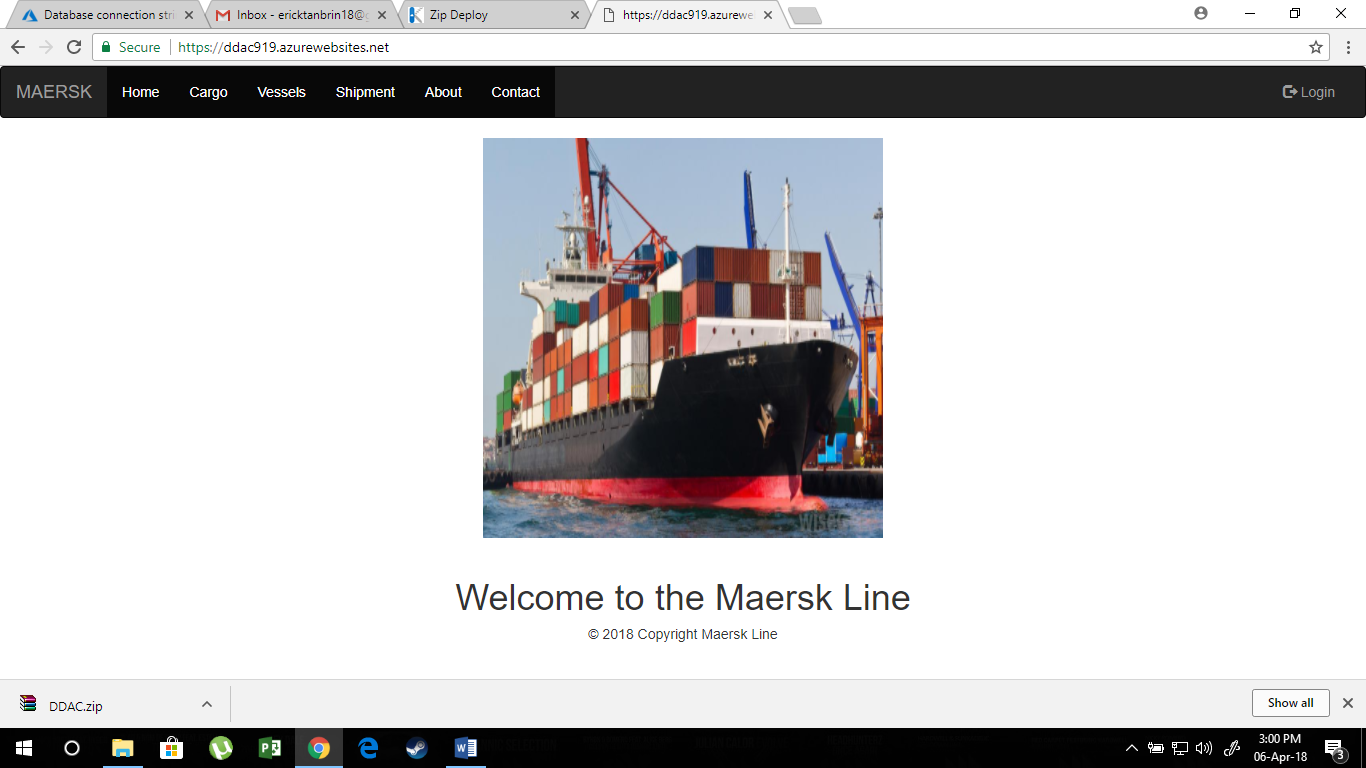
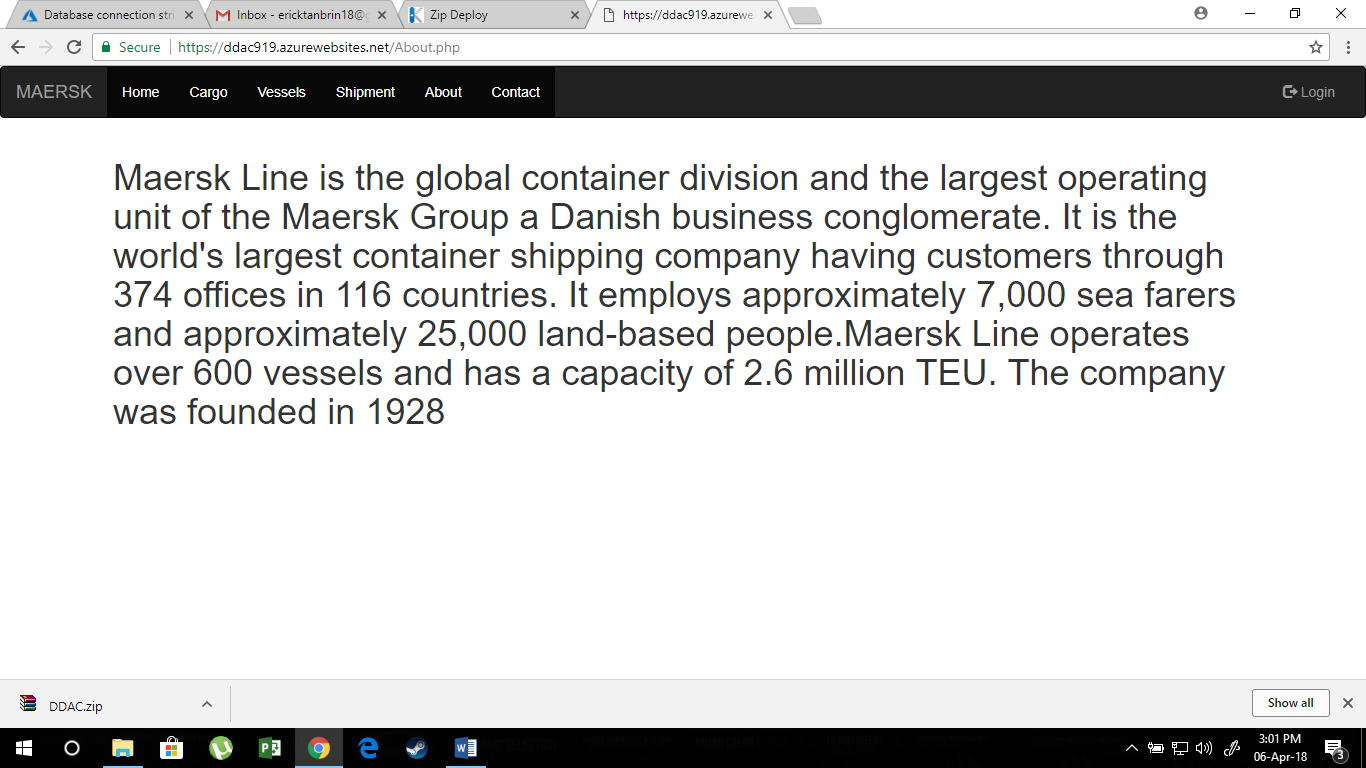


Figure 31.0 Homepage

The Figure above shows about homepage. This is where the main page of the Maersk Line.

### 4.7.2 About

  
Figure 32.0 About

The Figure above shows about the about page. This is where the information about the Maersk Line is written here.

### 4.7.3 Contact

Figure 33.0 Contact

The Figure above shows about the contact page. This is where the address and the contact number of the Maersk Line is written here.

### 4.7.4 Login

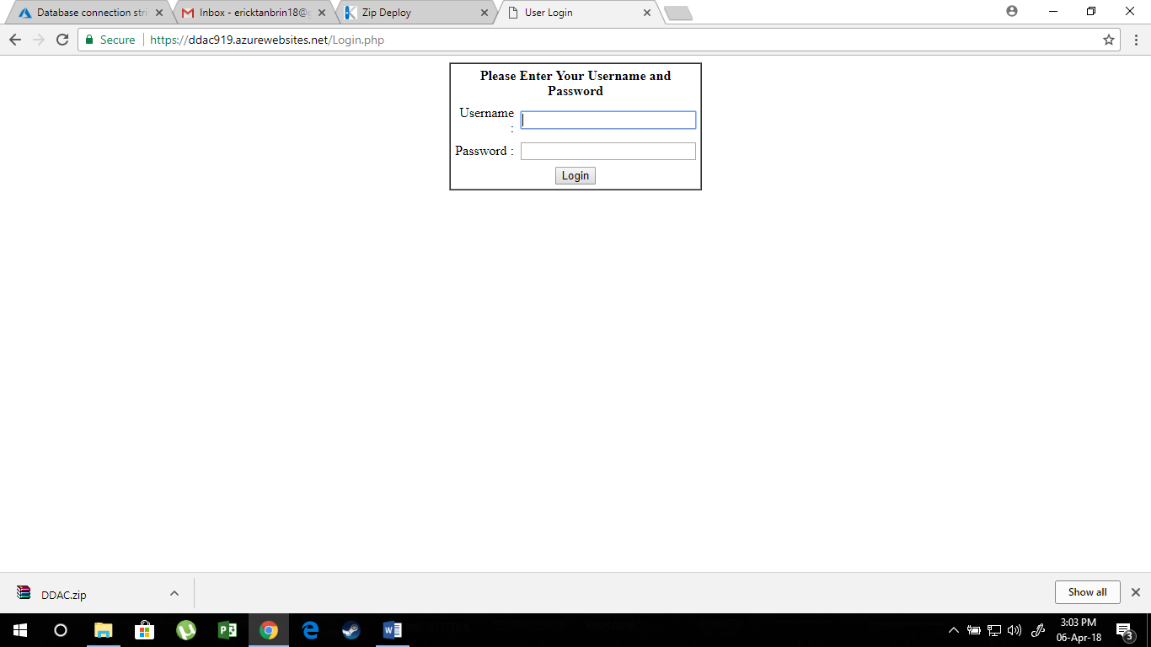


Figure 34.0 Login

The Figure above shows about login. This is where the agent just to enter the username and the password. The username is Agent and the password is agent.

### 4.7.5 Cargo

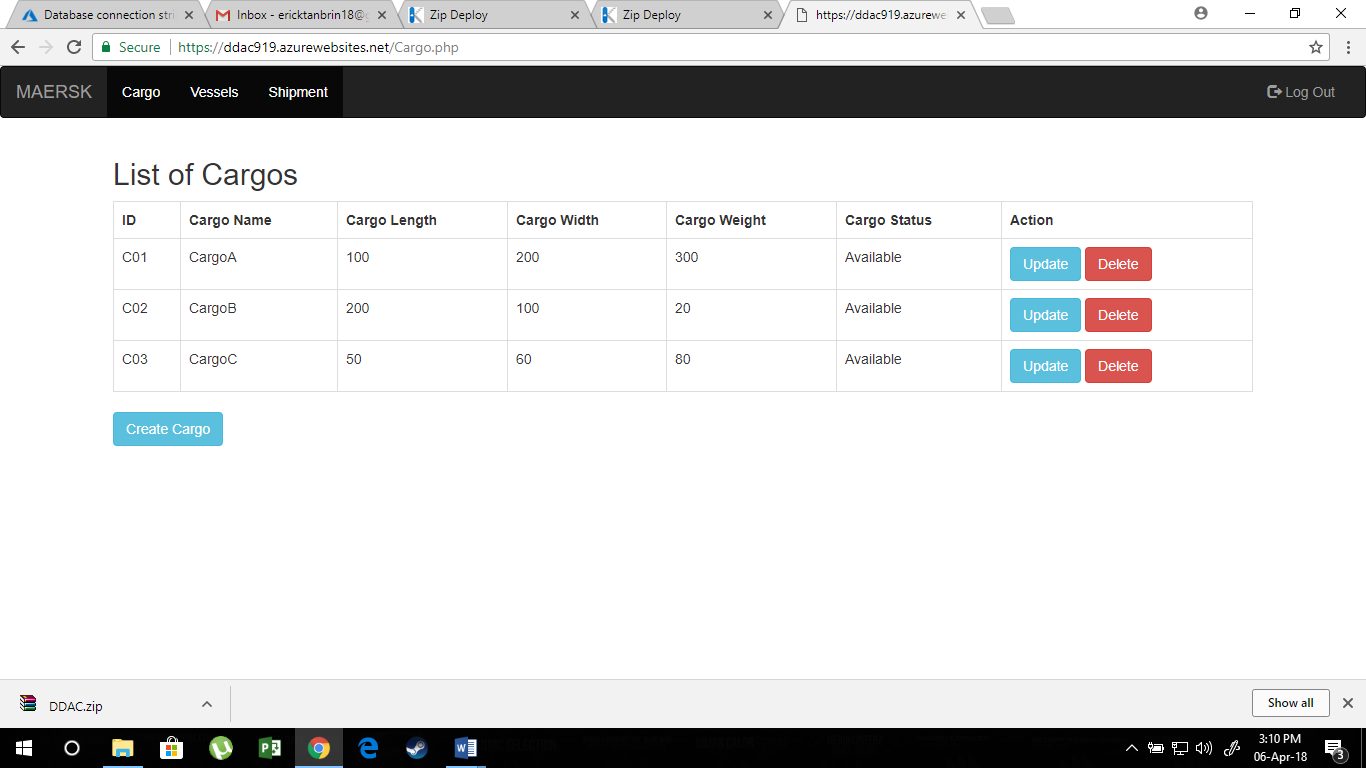


Figure 35.0 List of Cargo

The Figure above shows about the list of cargo. This is where the agent can view, create, update and delete the cargo.

### 4.7.6 Create Cargo

Figure 36.0 Create Cargo

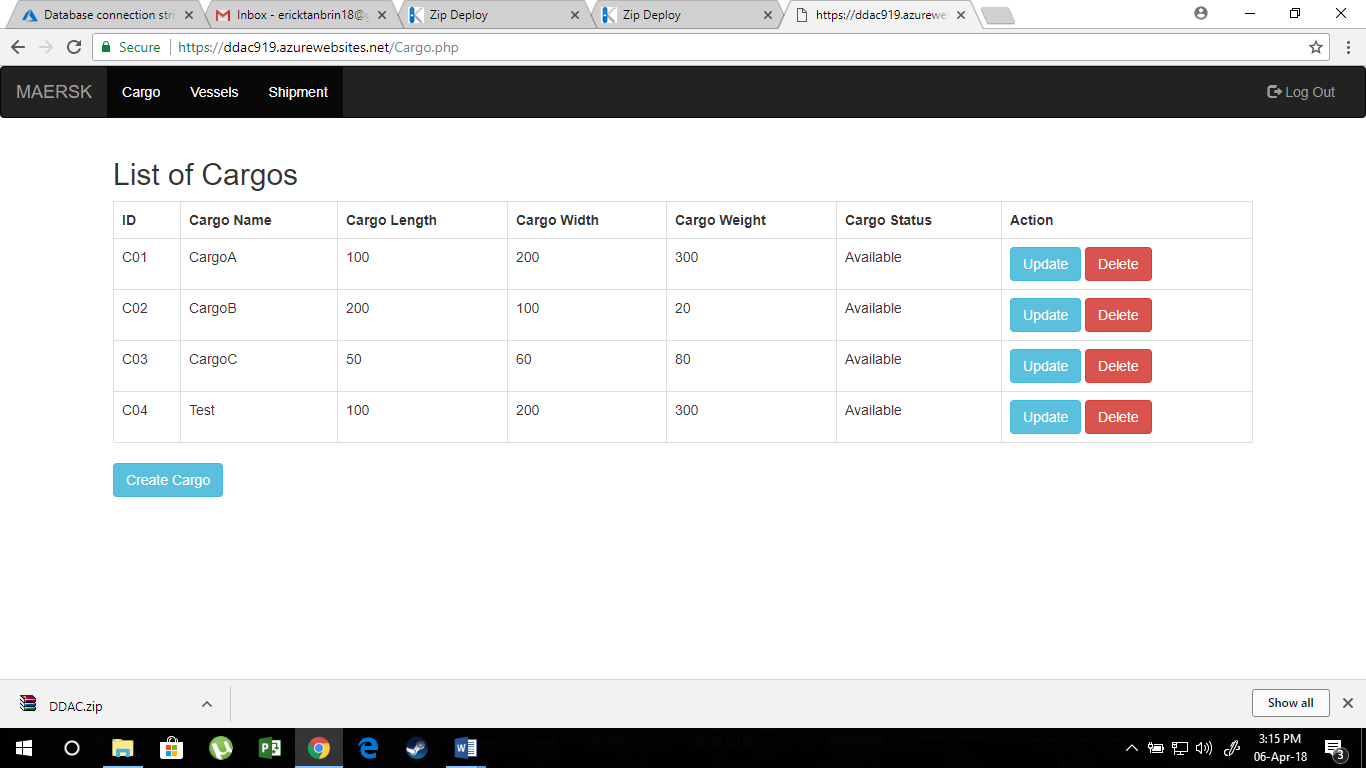
The Figure above shows about create cargo. This is where the agent just has to fill in the information and press the “Create Cargo” button to create the cargo and the result is shown in Figure 37.0. The previous button will bring the agent to the cargo page, just like shown in Figure 35.0.

Figure 37.0 Result After Creating Cargo

### 4.7.7 Update Cargo

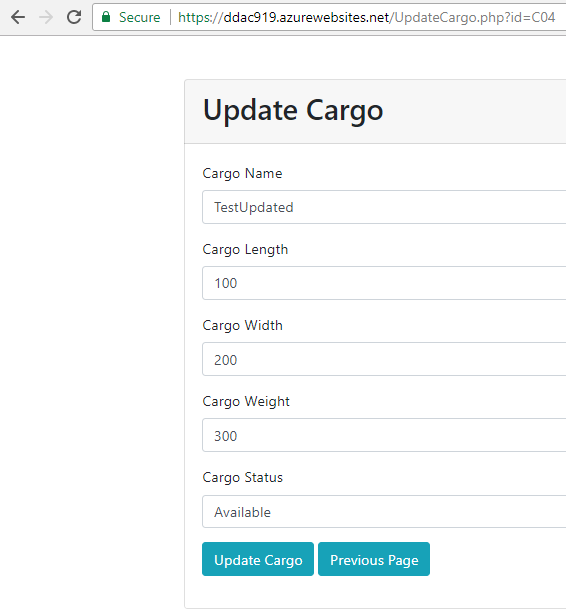


Figure 38.0 Cargo Update Figure 38.0 Cargo – Update

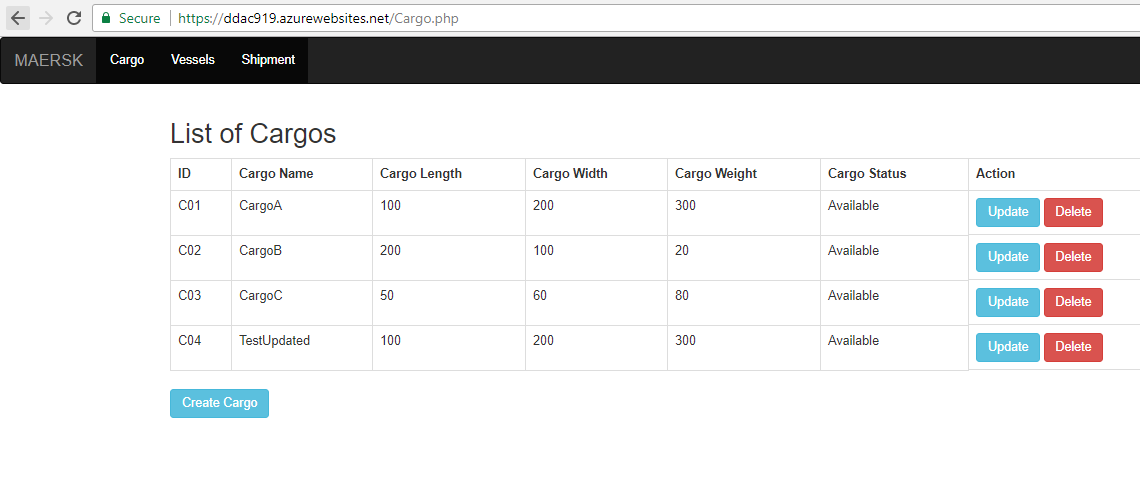
The Figure above shows about update cargo. For updating the cargo, the agent just has to click the update button that is provided on each row of the cargo. The name of the cargo, which is test is changed into testupdated. Once it is change, the agent just has to press the “Update Cargo” button and the result is shown in the Figure 39.0 below.

Figure 39.0 Result After Updating the Cargo Name

### 4.7.8 Delete Cargo

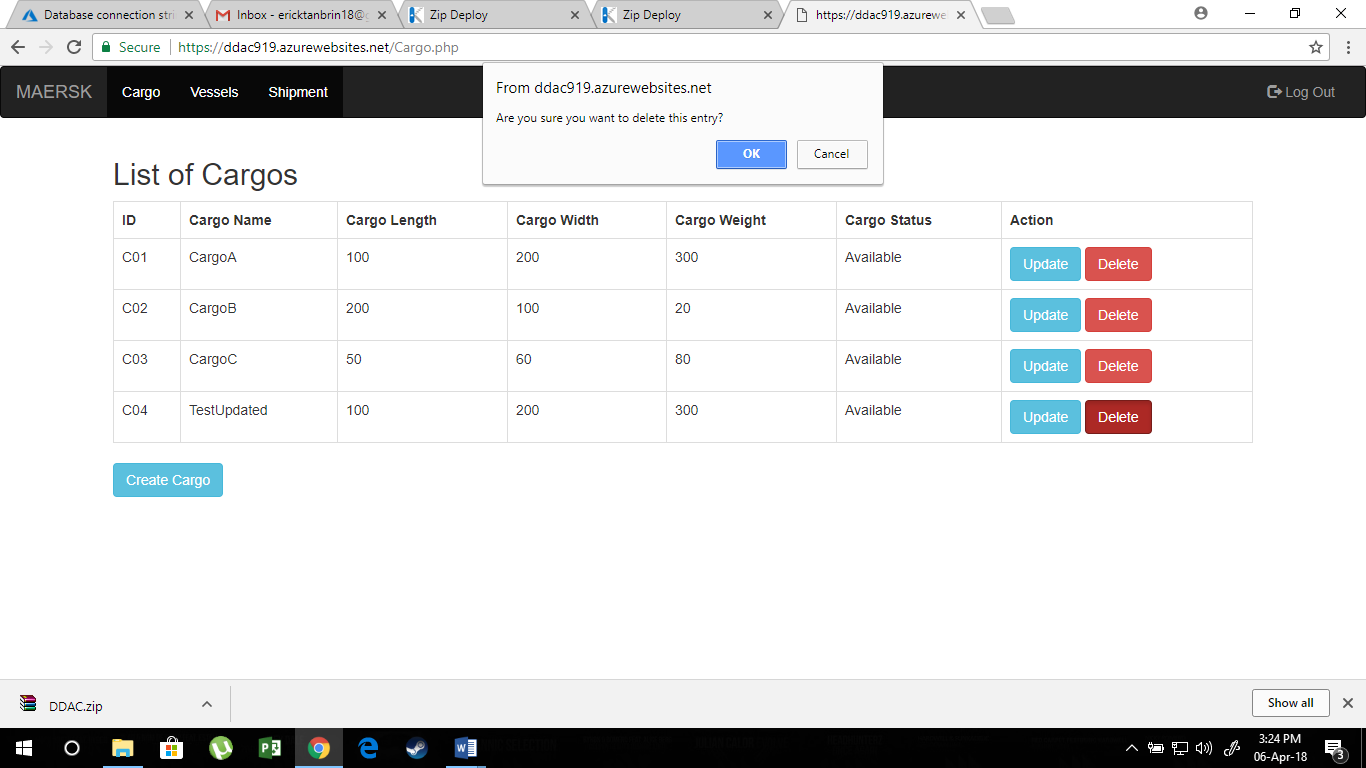


Figure 40.0 Delete Cargo

The Figure above shows about delete cargo. The agent just has to click the “Delete” button on the same row of the data that the agent wants to delete. After that, the confirmation will pop up and ask the agent whether that data wants to be deleted or not. If the agent presses ok, the data will be deleted from the database, just like shown in Figure 41.0.

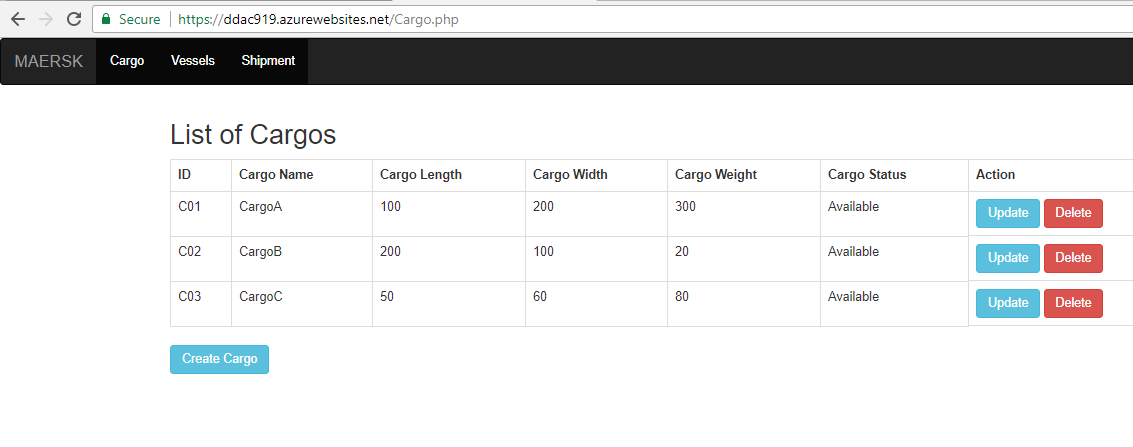


Figure 41.0 Result After the Data is Deleted

### 4.7.9 Vessel

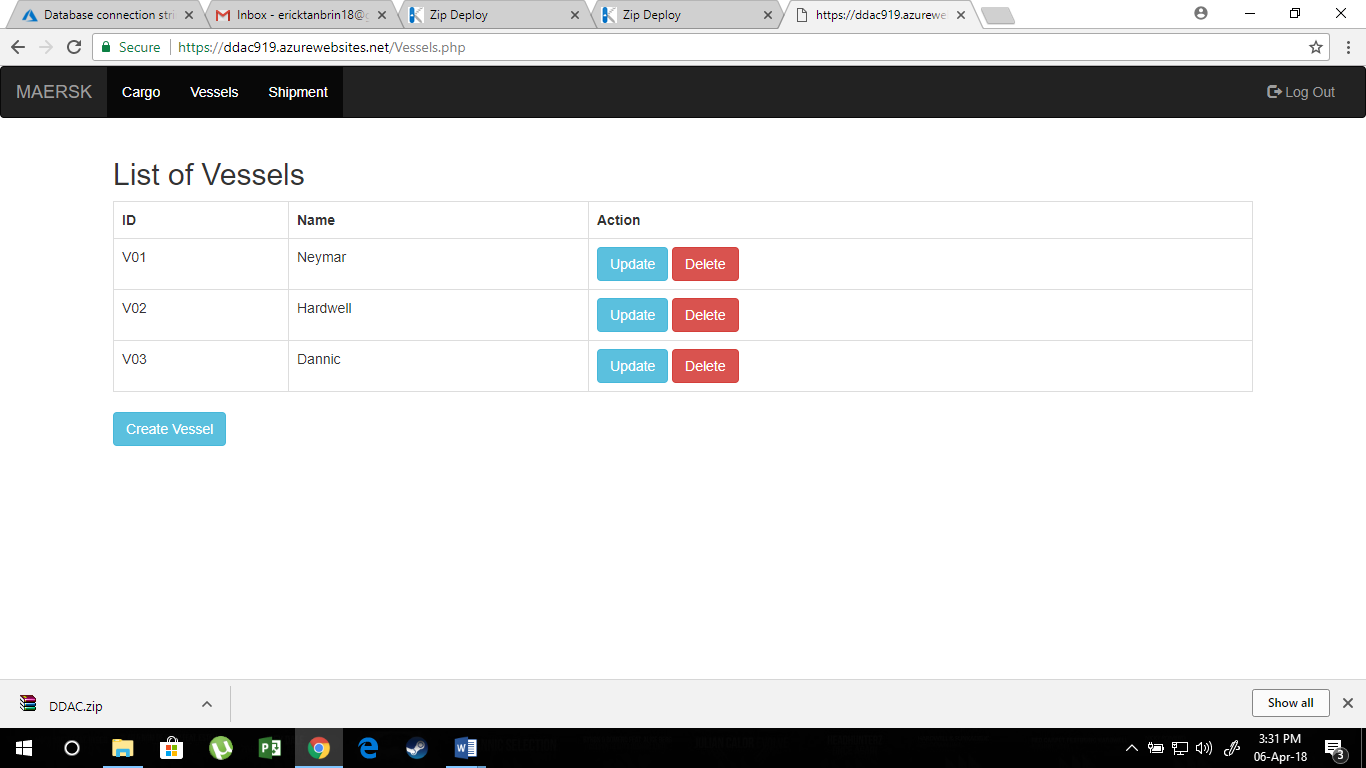


Figure 42.0 List of Vessel

The Figure above shows about the list of vessels. This is where the agent can view, create, update and delete the vessel.

### 4.7.10 Create Vessel

Figure 43.0 Create Vessel

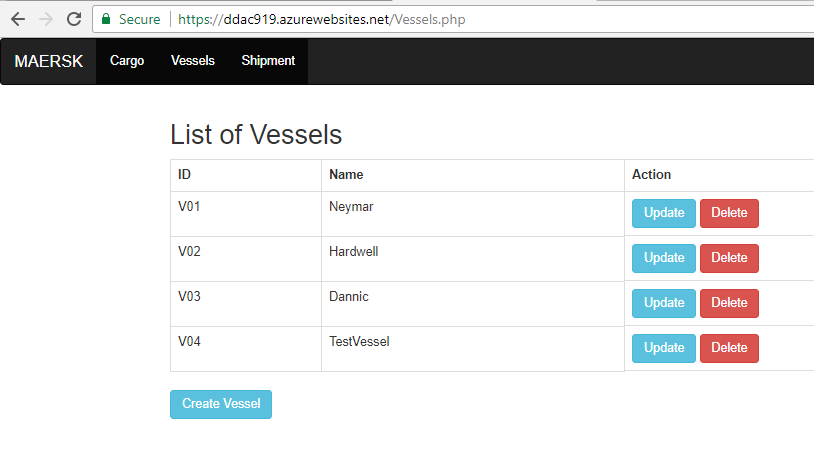
The Figure above shows about create vessel. This is where the agent just has to fill in the information and press the “Create Vessel” button to create the vessel and the result is shown in Figure 44.0. The previous button will bring the agent to the vessel page, just like shown in Figure 42.0.

Figure 44.0 Result After Creating New Vessel

### 4.7.11 Update Vessel

Figure 45.0 Update Vessel Figure 45.0 Update – Vessel

The Figure above shows about update vessel. For updating the vessel, the agent just has to click the update button that is provided on each row of the vessel. The name of the cargo, which is TestVessel is changed into TestVesselUpdated. Once it is change, the agent just has to press the “Update Vessel” button and the result is shown in the Figure 46.0 below. The previous button will bring the agent to the vessel page, just like shown in Figure 42.0

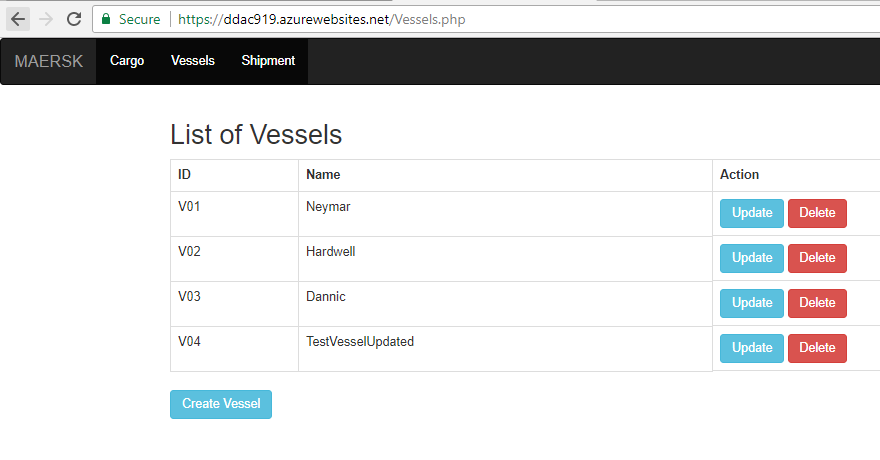


Figure 46.0 Result After Updating the Vessel Name

### 4.7.12 Delete Vessel

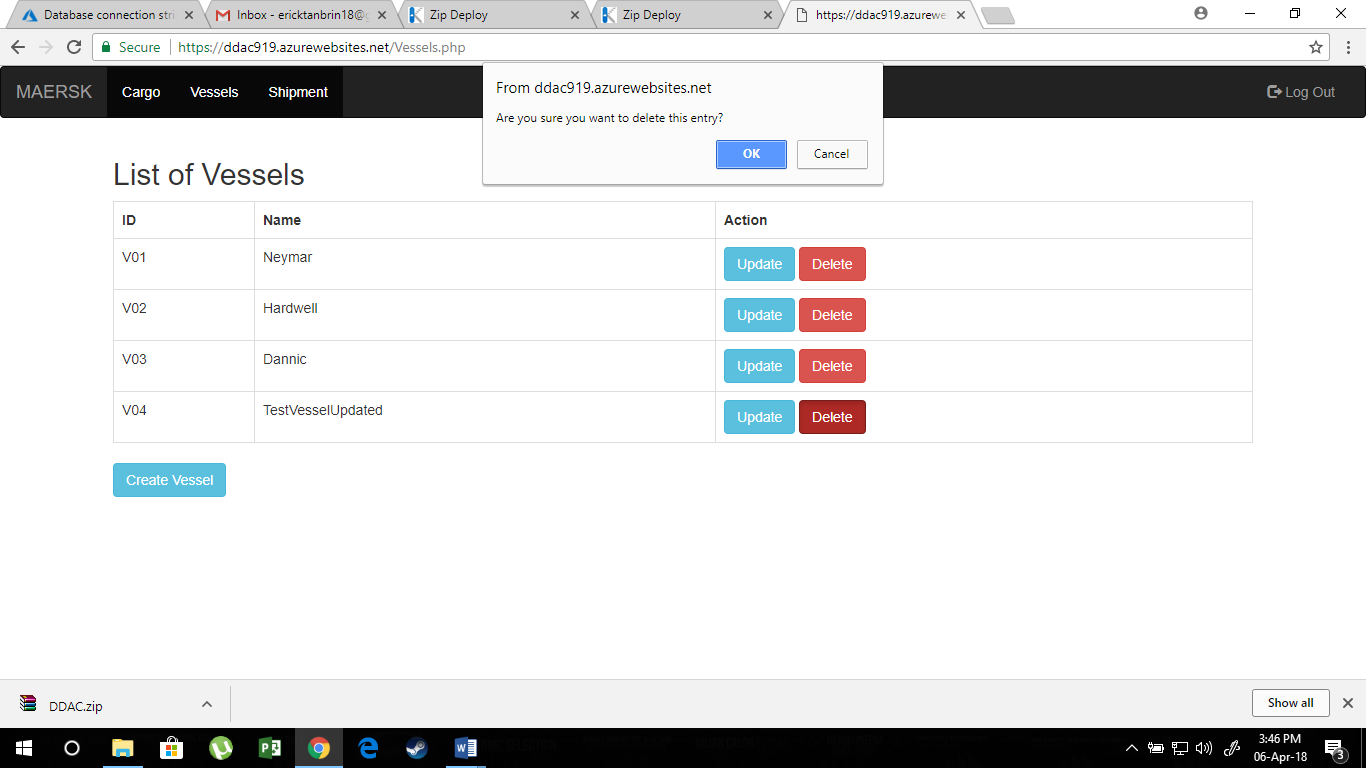


Figure 47.0 Delete Vessel

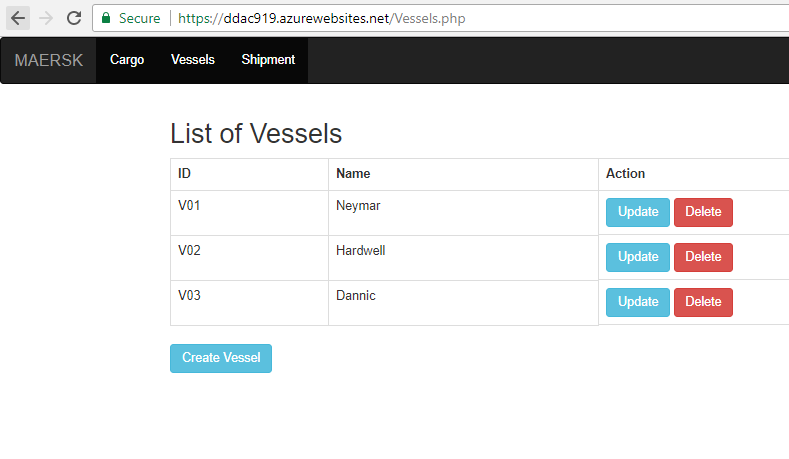
The Figure above shows about delete vessel. The agent just has to click the “Delete” button on the same row of the data that the agent wants to delete. After that, the confirmation will pop up and ask the agent whether that data wants to be deleted or not. If the agent presses ok, the data will be deleted from the database, just like shown in Figure 48.0.

Figure 48.0 Result After the Data is Deleted

### 4.7.13 Shipment

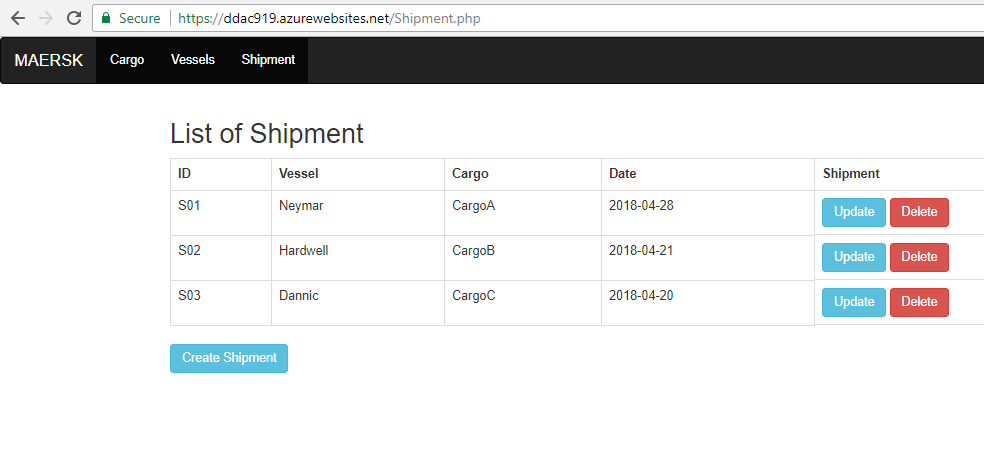


Figure 49.0 List of Shipment

Figure above shows about the list of shipment. This is where the agent can view, create, update and delete the shipment.

### 4.7.14 Create Shipment

Figure 50.0 Create New Shipment

The Figure above shows about create new shipment. The agent just has to enter the shipment ID, select the vessel name, select the cargo and select the shipment date. After that, the agent just has to click the “Create Shipment” button and the result is shown in the Figure 51.0. The previous button will bring the agent to the shipment page, just like shown in Figure 49.0.

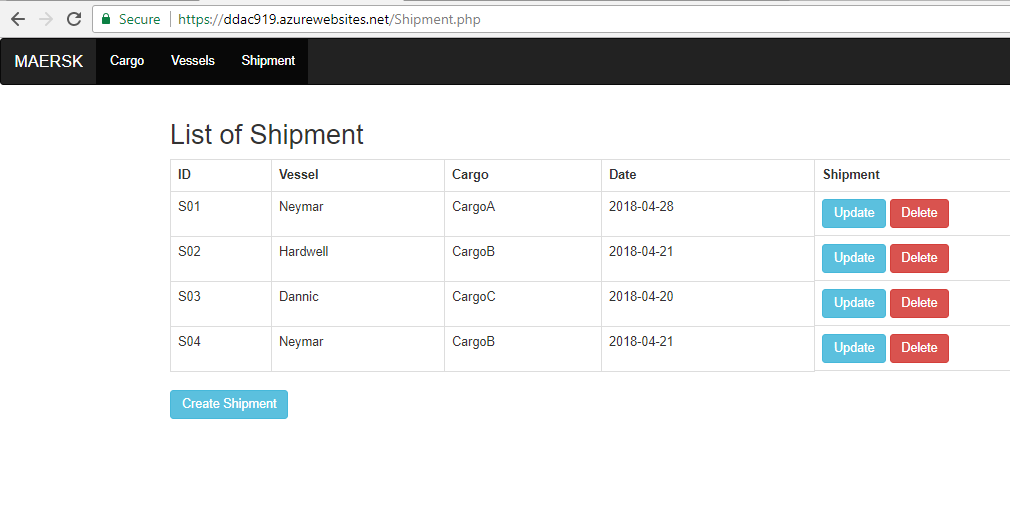


Figure 51.0 Result After Creating New Shipment

### 4.7.15 Update Shipment

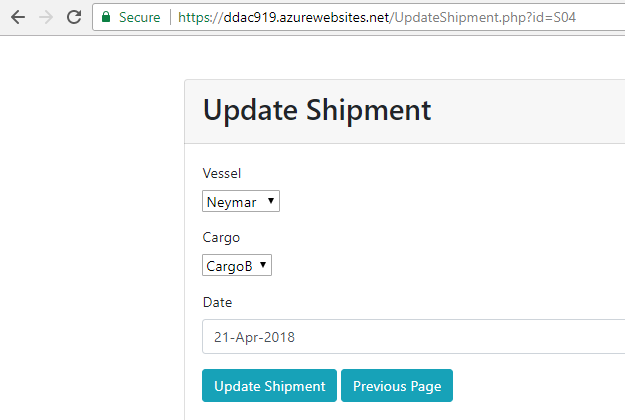
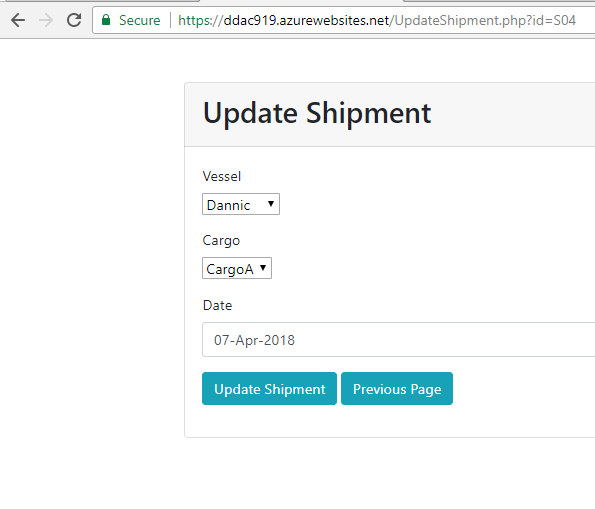


Figure 52.0 Update Shipment Figure 52.0 Update – Shipment

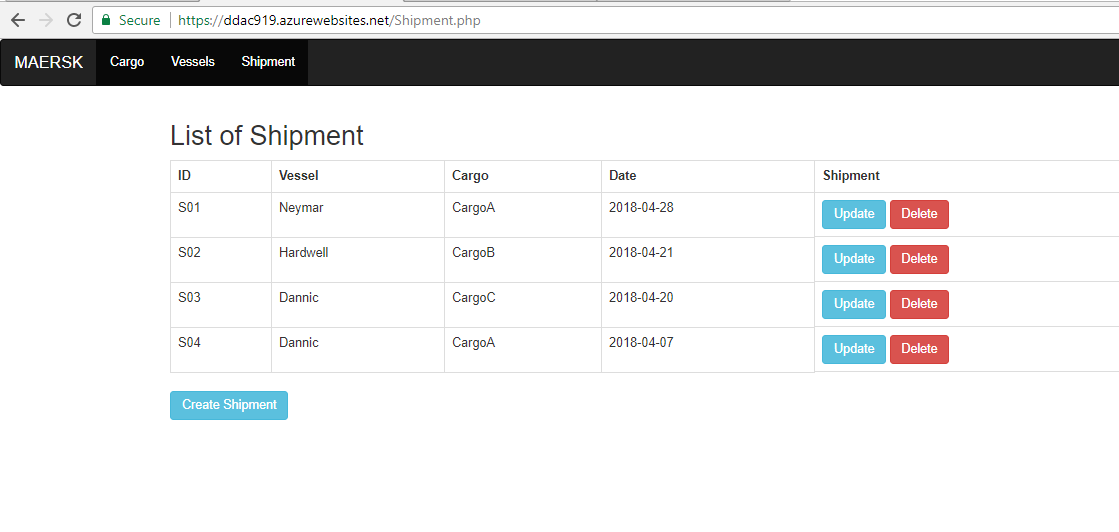
The Figure above shows about update shipment. For updating the cargo, the agent just has to click the update button that is provided on each row of the cargo. The agent can change the vessel name by selecting new vessel name, change the cargo name by selecting new cargo name and change the shipment date by selecting new shipment date. Once it is changed, the agent just has to press the “Update shipment” button and the result is shown in the Figure 53.0 below. The previous button will bring the agent to the shipment page, just like shown in Figure 49.0.

Figure 53.0 Result After Updating the Shipment Details

### 4.7.16 Delete Shipment

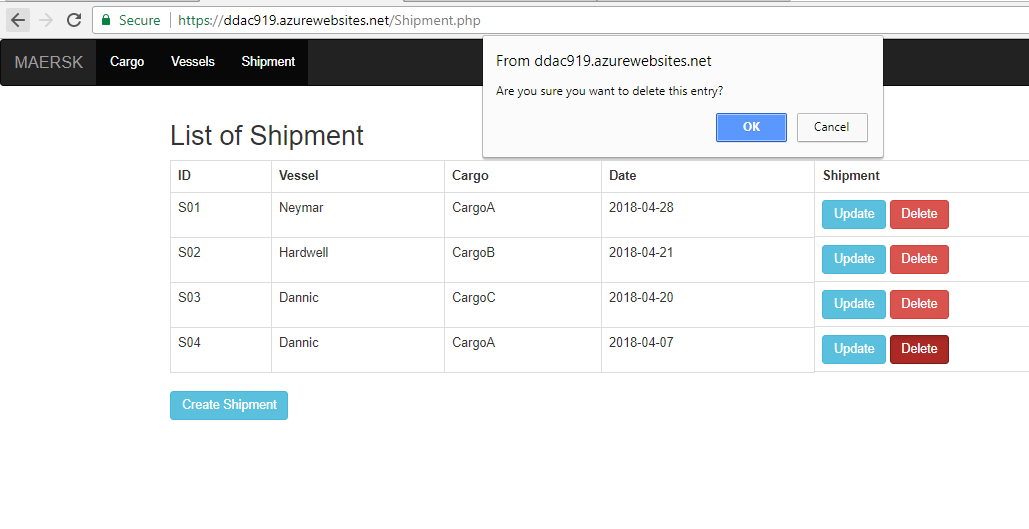


Figure 54.0 Delete Shipment

The Figure above shows about delete shipment. The agent just has to click the “Delete” button on the same row of the data that the agent wants to delete. After that, the confirmation will pop up and ask the agent whether that data wants to be deleted or not. If the agent presses ok, the data will be deleted from the database, just like shown in Figure 55.0.

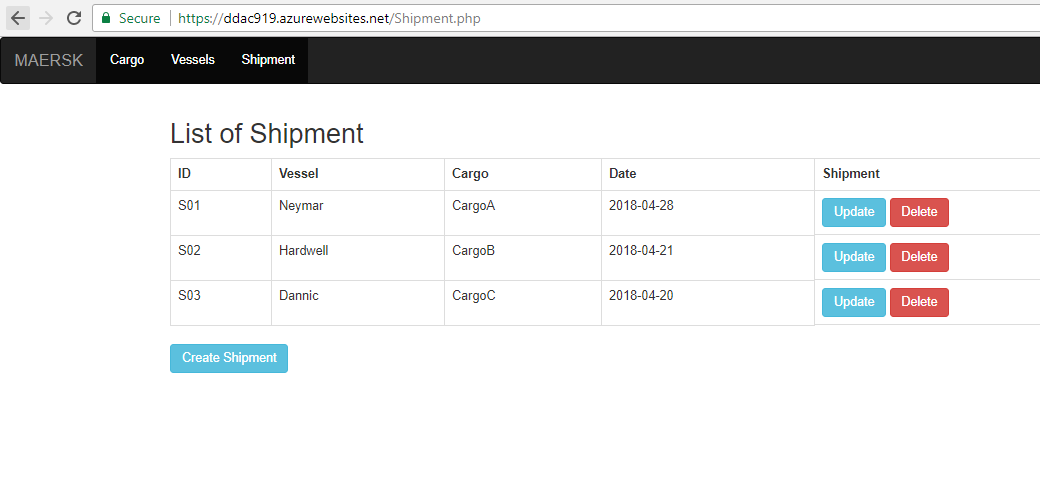


Figure 55.0 Result After the Data is Deleted

### 4.7.17 Logout

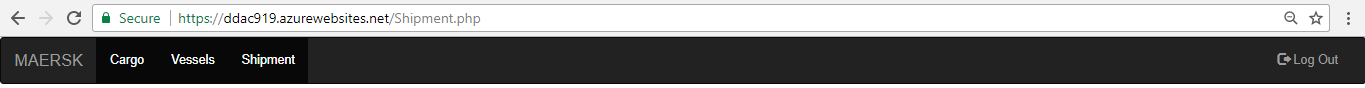


Figure 56.0 Logout

The Figure above shows about logout. If the agent wants to log out from the system, the agent just has to press the “Log Out” button and it will redirect the agent to the homepage, just like shown in Figure 31.0

# 5.0 Test Plan

## 5.1 Unit Testing

### 5.1.1 Login Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| LT1 | Username: Agent  Password: agent | The username and the password are correct | Login success | Login success | Pass | High |
| LT2 | Username: Agent  Password: (wrong input) | The username is correct, but the password is incorrect | Login failed!  Please enter the correct details | Login failed!  Please enter the correct details | Pass | High |
| LT3 | Username: (wrong input)  Password: agent | The username is incorrect, but the password is correct | Login failed!  Please enter the correct details | Login failed!  Please enter the correct details | Pass | High |
| LT4 | Username: Agent  Password: (blank) | The username is correct, but the password is blank | Login failed!  Please enter the password | Login failed!  Please enter the password | Pass | High |
| LT5 | Username: (blank)  Password: agent | The username is correct, but the password is blank | Login failed!  Please enter the username | Login failed!  Please enter the username | Pass | High |
| LT6 | Username: (blank)  Password: (blank) | The username and password is not entered | Login failed!  Please enter the username and the password | Login failed!  Please enter the username and the password | Pass | High |

Table 5.0 Login Testing

### 5.1.2 Create Cargo Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| CC1 | Cargo ID: C01  Cargo Name: CargoA  Cargo Length: 100  Cargo Width: 200  Cargo Weight: 300  Cargo Status: Available | The agent fills all the information correctly | Create cargo success | Create cargo success | Pass | High |
| CC2 | Cargo ID: (wrong input)  Cargo Name: (wrong input)  Cargo Length: (wrong input)  Cargo Width: (wrong input)  Cargo Weight: (wrong input)  Cargo Status: (wrong input) | The agent fills in all the information wrongly | Create cargo fail. Please enter the correct information | Create cargo fail. Please enter the correct information | Pass | High |
| CC3 | Cargo ID: (blank)  Cargo Name: (blank)  Cargo Length: (blank)  Cargo Width: (blank)  Cargo Weight: (blank)  Cargo Status: (blank) | The agent does not fill in all the information | Create cargo fail. Please fill in all the information | Create cargo fail. Please fill in all the information | Pass | High |

Table 6.0 Create Cargo Testing

### 5.1.3 Update Cargo Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| UC1 | Cargo Name: CargoAEdited  Cargo Length: 100  Cargo Width: 200  Cargo Weight: 300  Cargo Status: Available | The agent fills all the information correctly | Update cargo success | Update cargo success | Pass | High |
| UC2 | Cargo ID: (wrong input)  Cargo Name: (wrong input)  Cargo Length: (wrong input)  Cargo Width: (wrong input)  Cargo Weight: (wrong input)  Cargo Status: (wrong input) | The agent fills in all the information wrongly | Update cargo fail. Please enter the correct information | Update cargo fail. Please enter the correct information | Pass | High |
| UC3 | Cargo ID: (blank)  Cargo Name: (blank)  Cargo Length: (blank)  Cargo Width: (blank)  Cargo Weight: (blank)  Cargo Status: (blank) | The agent does not fill in all the information | Create cargo fail. Please fill in all the information | Create cargo fail. Please fill in all the information | Pass | High |

Table 7.0 Update Cargo Testing

### 5.1.4 Create Vessel Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| CV1 | Vessel ID: V01  Vessel Name: Neymar | The agent fills all the information correctly | Create vessel success | Create vessel success | Pass | High |
| CV2 | Vessel ID: (wrong input)  Vessel Name: (wrong input) | The agent fills in all the information wrongly | Create vessel fail. Please enter the correct information | Create vessel fail. Please enter the correct information | Pass | High |
| CV3 | Vessel ID: (blank)  Vessel Name: (blank) | The agent does not fill in all the information | Create vessel fail. Please fill in all the information | Create vessel fail. Please fill in all the information | Pass | High |

Table 8.0 Create Vessel Testing

### 5.1.5 Update Vessel Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| UV1 | Vessel Name: NeymarEdited | The agent fills all the information correctly | Update vessel success | Update cargo success | Pass | High |
| UV2 | Vessel Name: (wrong input) | The agent fills in all the information wrongly | Update vessel fail. Please enter the correct information | Update vessel fail. Please enter the correct information | Pass | High |
| UV3 | Vessel Name: (blank) | The agent does not fill in all the information | Create vessel fail. Please fill in all the information | Create vessel fail. Please fill in all the information | Pass | High |

Table 9.0 Update Vessel Testing

### 5.1.6 Create Shipment Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| CS1 | Shipment ID: C01  Vessel Name: Neymar  Cargo Name: CargoA  Shipment Date: 28 – 04 – 2018 | The agent fills all the information correctly | Create Shipment success | Create Shipment success | Pass | High |
| CS2 | Shipment ID: (wrong input)  Vessel Name: (wrong input)  Cargo Name: (wrong input)  Shipment Date: (wrong input) | The agent fills in all the information wrongly | Create shipment fail. Please enter the correct information | Create shipment fail. Please enter the correct information | Pass | High |
| CS3 | Shipment ID: (blank)  Vessel Name: (blank)  Cargo Name: (blank)  Shipment Date: (blank) | The agent does not fill in all the information | Create shipment fail. Please fill in all the information | Create shipment fail. Please fill in all the information | Pass | High |

Table 10.0 Create Shipment Testing

### 5.1.7 Create Shipment Testing

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | **Test Conditions** | **Description** | **Expected Result** | **Actual Result** | **Status** | **Test Priority** |
| US1 | Vessel Name: Hardwell  Cargo Name: CargoB  Shipment Date: 21 – 04 – 2018 | The agent fills all the information correctly | Update Shipment success | Update Shipment success | Pass | High |
| US2 | Shipment ID: (wrong input)  Vessel Name: (wrong input)  Cargo Name: (wrong input)  Shipment Date: (wrong input) | The agent fills in all the information wrongly | Update shipment fail. Please enter the correct information | Update shipment fail. Please enter the correct information | Pass | High |
| US3 | Shipment ID: (blank)  Vessel Name: (blank)  Cargo Name: (blank)  Shipment Date: (blank) | The agent does not fill in all the information | Update shipment fail. Please fill in all the information | Update shipment fail. Please fill in all the information | Pass | High |

Table 11.0 Update Shipment Testing

## 5.2 Performance Testing

For the test performance, it took 15 minutes to prepare the result of the performance test. For this performance test, the developer chose to simulate 450 concurrent users in 5 minutes and the result was shown below.

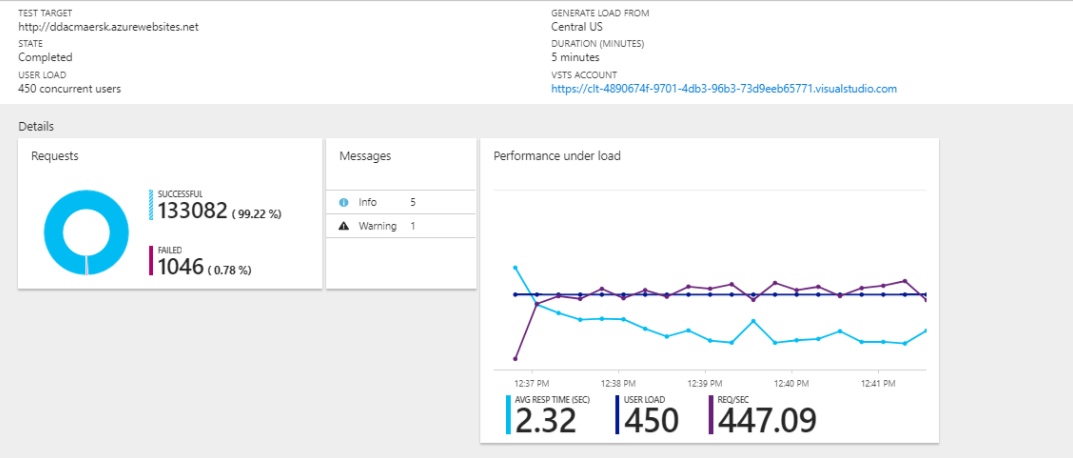


Figure 57.0 Performance Test

The Figure above shows about the performance test. In the performance test, it also shows average response time, user load and request per second.

# 6.0 Conclusion

In conclusion, the web application was successfully uploaded on the azure. Even though the developer struggled with the development of the web application and uploading the website, still the developer managed to overcome the obstacle. After that, the developer also learned how to create web application using azure, uploading the web application into the azure, the auto scaling and the traffic manager. Hopefully, the developer can use this knowledge for the future when it comes in developing a website and upload it on the azure and also how the cloud computing works.

# 7.0 References

* (Microsoft, 2018)

Microsoft. (2018). *Supported languages in Azure Functions*. [online] Available at: https://docs.microsoft.com/en-us/azure/azure-functions/supported-languages [Accessed 9 Apr. 2018].

* (Microsoft, 2018)

**Microsoft**. (2018). *Azure solutions | Microsoft Azure*. [online] Available at: https://azure.microsoft.com/en-us/solutions/ [Accessed 9 Apr. 2018].

* (Microsoft, 2018)

**Microsoft**. (2018). *What is Traffic Manager*. [online] Available at: https://docs.microsoft.com/en-us/azure/traffic-manager/traffic-manager-overview [Accessed 9 Apr. 2018].

# 8.0 Appendix (GitHub & Video)

The web application source code is published to GitHub

<https://github.com/ericktanbrin18/DDAC>

The video is published to Microsoft Stream

<https://web.microsoftstream.com/video/10f122df-5837-4cc0-bafc-2598d7b9021a?list=studio>