

|  |  |  |
| --- | --- | --- |
| **Module Code** | **:** | CT071-3-3-DDAC (Designing & Developing Cloud Applications) |
| **Assignment Title** | **:** | Maersk Line Container Management System (CMS) |
| **Intake Code** | **:** | UC3F1706SE |
| **Lecturer Name** | **:** | Dr. Kalai Anand A/L Ratnam |
| **Hand in Date** | **:** | 7 December 2017 |
| **Tutorial No.** | **:** | 2 |

|  |  |
| --- | --- |
| **Student ID** | **Student Name** |
| TP034688 | Chen Xin Wan |

**Acknowledgement**

The developer has taken efforts to complete Design & Developing Cloud Applications (DDAC) assignment. However, the DDAC assignment would not have been possible to be completed successfully without the kind support, help and guidance of many individuals. The developer would like to extend her sincere thanks to all of them.

First and foremost, the developer would like to express her special gratitude and thanks to Dr. Kalai Anand Ratnam who is lecturer of DDAC module for providing necessary information and guidelines on the concepts regarding the usage as well as application of Cloud services offered by Microsoft Azure into the implemented software application that serves as the main objective of DDAC module which aids the developer to complete the DDAC assignment smoothly due to clear understanding on the concepts of Cloud application development. Besides that, the thanks and appreciations of the developer also go to her course mates for their willingness in spending their time on providing useful guidelines to resolve difficulties that is faced by the developer while completing DDAC assignment which enable the completion of DDAC assignment successfully. Apart from that, the developer also would like to express her gratitude towards her family for being her pillar of support physically and mentally throughout her journey of completing DDAC assignment. They always provide advice to her on the techniques of addressing the challenges that are faced while completing DDAC assignment. This can enable the developer to have capability in overcoming the challenge that could result to delivery of DDAC assignment on time.

Table of Contents

[1.0 Introduction 1](#_Toc511380520)

[2.0 Project Plan 2](#_Toc511380521)

[3.0 Design 4](#_Toc511380522)

[3.1 Design Considerations 4](#_Toc511380523)

[3.2 Modelling 5](#_Toc511380524)

[3.2.1 Use Case Diagram 5](#_Toc511380525)

[3.2.2 Use Case Specification 5](#_Toc511380526)

[3.2.3 Activity Diagram 11](#_Toc511380527)

[3.2.4 Entity Relationship Diagram 19](#_Toc511380528)

[3.2.5 Cloud Architecture 20](#_Toc511380529)

[4.0 Implementation 22](#_Toc511380530)

[4.1 Application Development 22](#_Toc511380531)

[4.2 Microsoft Azure Publishing 26](#_Toc511380532)

[4.3 Application Scaling 31](#_Toc511380533)

[4.4 Reliability and Performance 34](#_Toc511380534)

[5.0 Testing 36](#_Toc511380535)

[5.1 Unit Testing 36](#_Toc511380536)

[5.1.1 Login (Admin and Agent) 37](#_Toc511380537)

[5.1.2 Register Agent 37](#_Toc511380538)

[5.1.3 View Agent 38](#_Toc511380539)

[5.1.4 View Booking 39](#_Toc511380540)

[5.1.5 Create Schedule 39](#_Toc511380541)

[5.1.6 Remove Schedule 40](#_Toc511380542)

[5.1.7 View Confirmed Booking 40](#_Toc511380543)

[5.1.8 View Schedule 41](#_Toc511380544)

[5.1.9 Book Vessel 41](#_Toc511380545)

[5.1.10 Register Customer 42](#_Toc511380546)

[5.1.11 Register Item 43](#_Toc511380547)

[5.2 Performance Testing 44](#_Toc511380548)

[5.2.1 Analysis 46](#_Toc511380549)

[6.0 Managed Databases 47](#_Toc511380550)

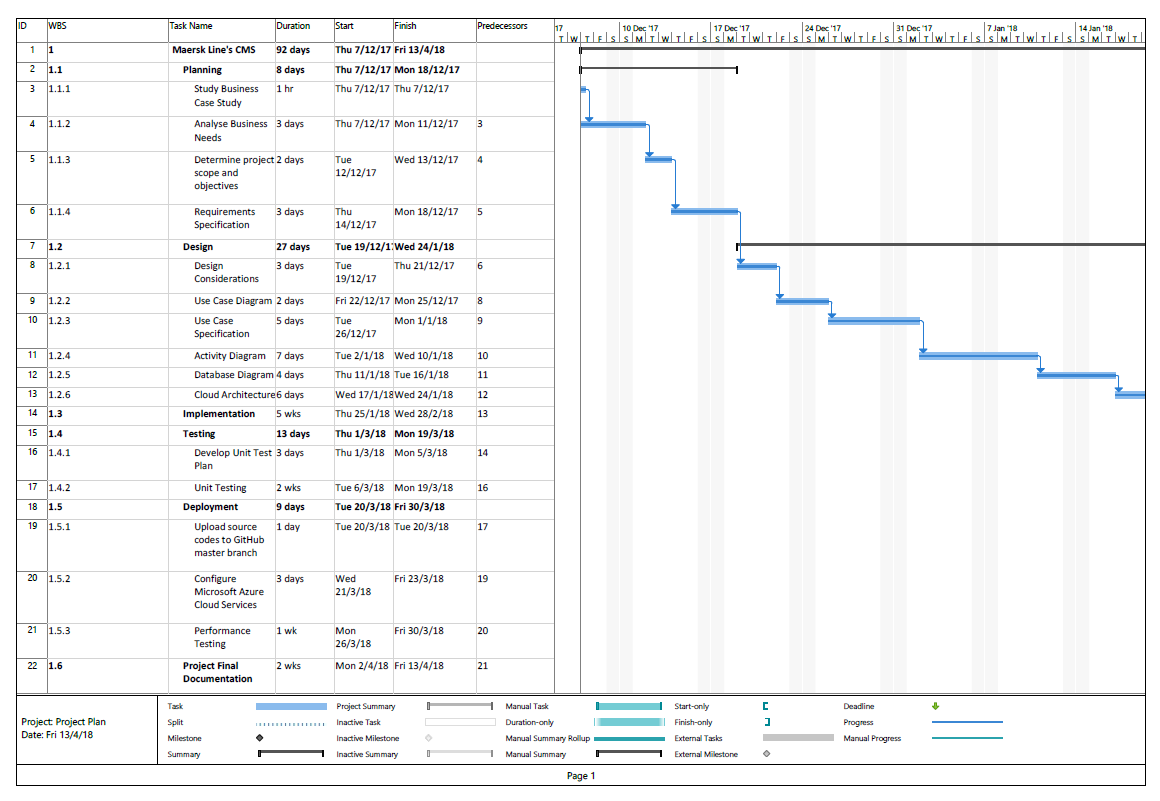
[7.0 Conclusion 51](#_Toc511380551)

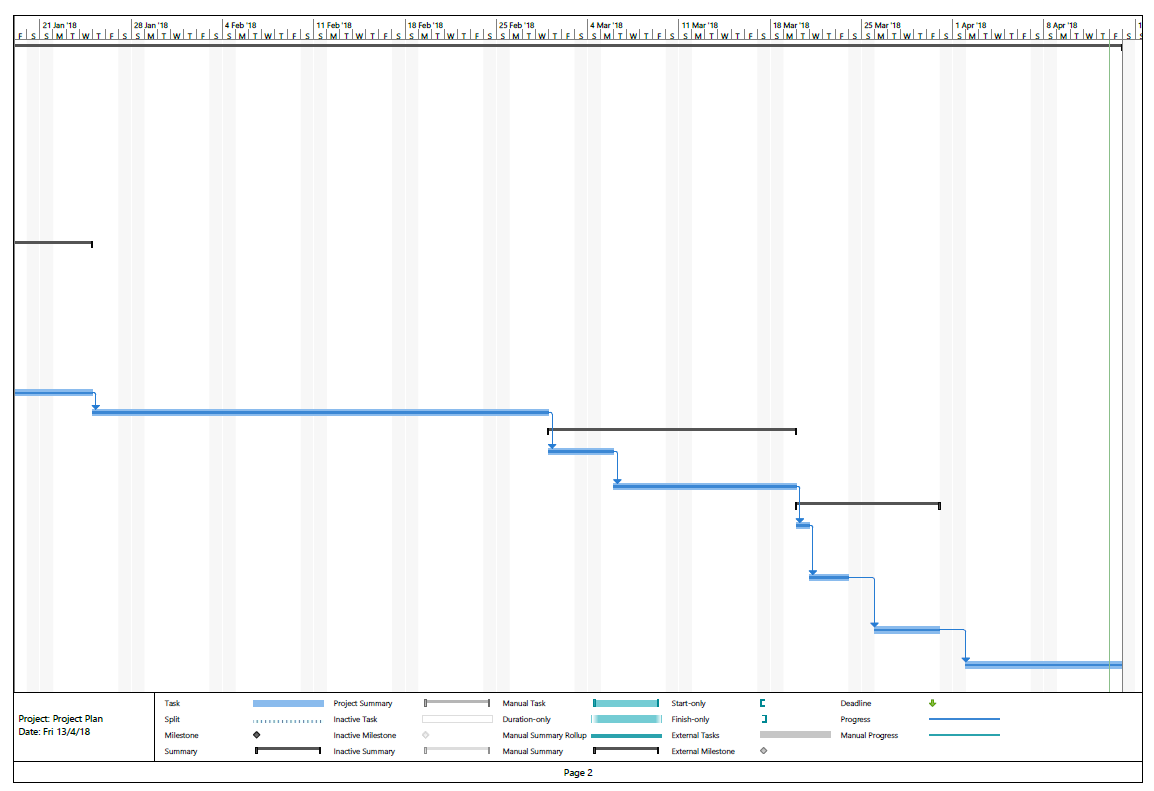
[8.0 References 53](#_Toc511380552)

# **1.0 Introduction**

Maersk Line is the global container division and biggest operating unit of Maersk Group that is known as the world’s largest container shipping company based in Denmark which was founded in 1928 that operates in 100 countries and offers the goods transportation for customers through 374 offices in 116 countries. Currently, Maersk Line has employment of approximately 7,000 sea farers and 25,000 land-based people and contains over 600 vessels and has the capacity of 2.6 million TEU. Maersk Line is looking at designing and implementation of a Container Management System (CMS) which helps to cater to manage the containers effectively and efficiently that can prevent extensive usage of overall supply chain costs and simplify the process of managing or handling logistics. Maersk Line has taken virtualised platforms options that are offered by Microsoft Azure and Amazon Web Services into consideration. However, some of Maersk Line’s IT environment are already hosted in virtualised platform offered by Microsoft Azure and currently Maersk Line has changed over its IT setup based on Microsoft Azure after successfully approached Microsoft about expanding the scope of the relationship initially. Hence, the resulting Containing Management System (CMS) will be implemented and deployed onto the Microsoft Azure virtualised platform. The features and functionalities that are offered by Maersk Line CMS provides the ability for the employees of Maersk Line to administer the process involved in transportation of goods or cargo which consists of creating accounts and schedule of vessels as well as keeping track of the bookings for goods transportation online as well as customers of Maersk Line to use the transportation service of Maersk Line for their client’s goods which consists of vessel reservation and check the reservations that have been made online. This document the details of the development of CMS which is proposed as solution to the problems that are faced by Maersk Line in managing the process involved in goods transportation as well as the result or outcome of CMS is deployed onto Microsoft Azure virtualised platform. Besides that, the decisions made in design and implementation of CMS is also explained.

# **2.0 Project Plan**





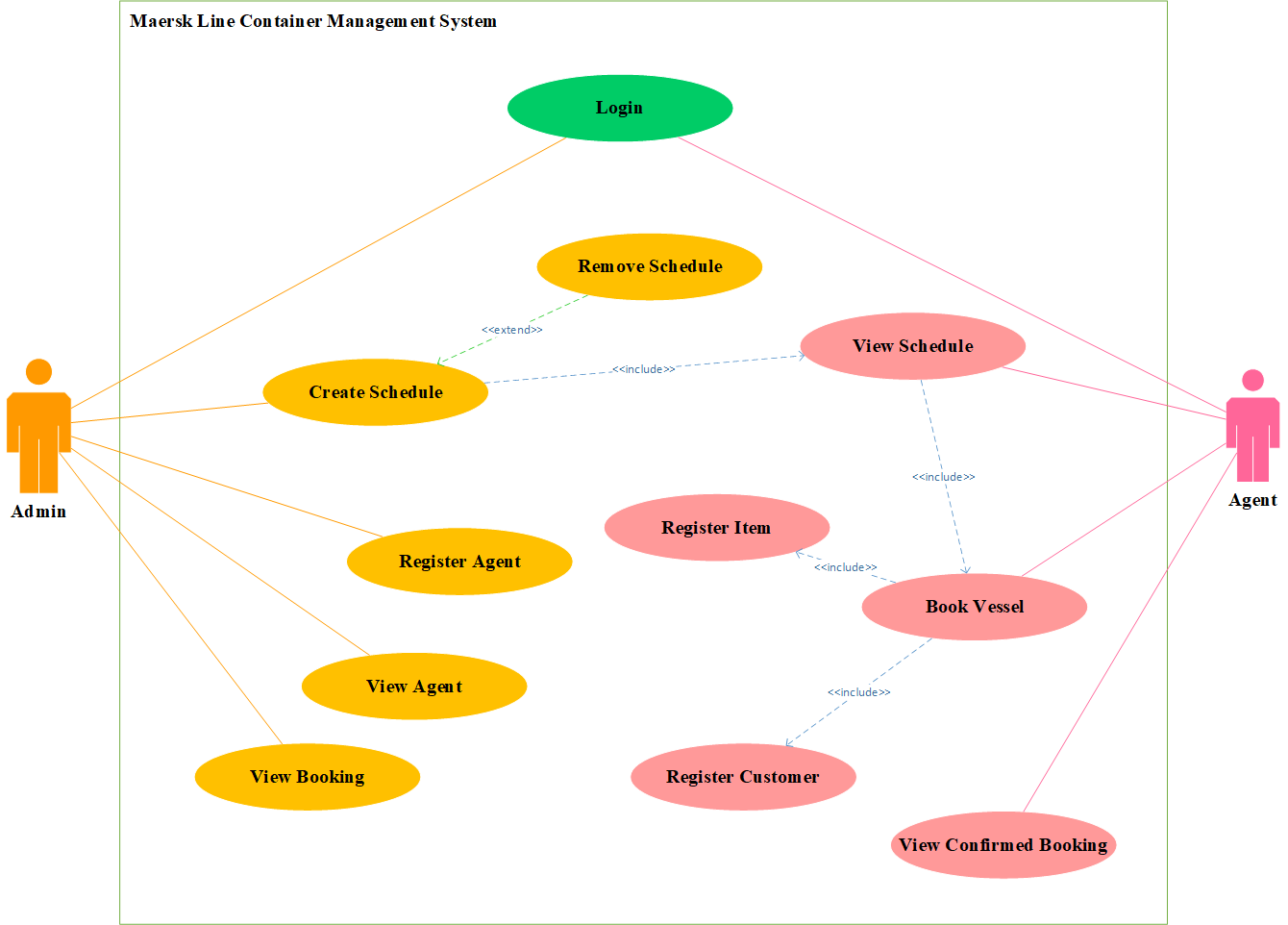
# **3.0 Design**

## **3.1 Design Considerations**

There are numbers of assumptions and considerations that have been identified and determined during the process of designing and implementing Maersk Line’s CMS. The first assumption is the Maersk Line’s CMS currently target towards the users who consists of admins that are known as employees of Maersk Line and agents who are known as the users that obtain the transportation service offered by Maersk Line for their client’s goods that are located at the region where the main office of Maersk Line is positioned which is North Europe only. This is to ensure that Maersk Line’s CMS that is newly implemented has high level of performance which enable effective cargo handling business, demonstrated the ability of meeting the needs of demands during peak seasons as well as well-maintained over time which result in achieving reliability and quality assurance before expanding the system to other regions where Maersk Line is operating at to increase organisational flexibility and increase market share of container shipping industry which can help to further business growth of Maersk Line. Hence, the developed CMS is currently tested in North Europe region only. In addition, the second assumption is development environment of Maersk Line’s CMS in Microsoft Azure virtualized platform is production environment which means that there is no continuous delivery environment created for the web-based application in Microsoft Azure virtualized platform is because there is local environment has been created for Maersk Line’s CMS that is localhost which means that the localhost is treated as testing environment for Maersk Line’s CMS and the final version of the web-based application is then only deployed into production environment which is at Microsoft Azure virtualized platform unless there are changes have been occurred which have been tested on localhost to ensure correctness and then only committed to Microsoft Azure virtualized platform which this process is classified as maintenance. Besides that, the validity of the development for Maersk Line’s CMS associated with the concept is examined on Microsoft Azure Cloud platform through the usage of credit amount RM850.00 that is offered by Microsoft Azure.

## **3.2 Modelling**

### **3.2.1 Use Case Diagram**



**Figure 3.2.1.1: Maersk Line Container Management System Use Case Diagram**

### **3.2.2 Use Case Specification**

**Login**

|  |  |
| --- | --- |
| **Use Case** | Login |
| **Summary** | User logs into the system in order to perform tasks using the functions offered by the system. |
| **Dependency** | None |
| **Actors** | Admin, Agent |
| **Preconditions** | User is already registered and has an account for the system. |
| **Descriptive of Main Sequence** | 1. User provides username and password.  2. System verifies the username and password given by the user.  3. System displays “Congratulations you have successfully login!” message.  4. System grant access. |
| **Descriptive of Alternative Sequence** | 2a. If the username or password entered by the user is wrong, the user is required to re-enter the username and password. |
| **Postcondition** | User is redirected to main menu and proceeds to use the system for performing tasks. |

**Register Agent**

|  |  |
| --- | --- |
| **Use Case** | Register Agent |
| **Summary** | Admin registers the agent who is the user that obtains the shipping service of Maersk Line for his or her client’s goods. |
| **Dependency** | None |
| **Actors** | Admin |
| **Preconditions** | Admin is already logged into the system. |
| **Descriptive of Main Sequence** | 1. Admin selects dropdown of “Agents”.  2. Admin selects ‘Register Agents’  3. System retrieves and displays the empty registration form.  4. Admin enters registration details.  5. System displays “You have successfully created an agent” message. |
| **Descriptive of Alternative Sequence** | 5a) If the form is not completed, system returns ‘Please fill out this field’ message.  5b) If details that are provided by the admin do not match the format required for the fields or duplicated, admin will prompt to re-enter the details. |
| **Postcondition** | Agent can gain access to the system as well as perform activities using the system. |

**View Agent**

|  |  |
| --- | --- |
| **Use Case** | View Agent |
| **Summary** | Admin views the information of agents who are added into the system. |
| **Dependency** | None |
| **Actors** | Admin |
| **Preconditions** | Admin is already logged into the system. |
| **Descriptive of Main Sequence** | 1. Admin selects dropdown of “Agents”.  2. Admin chooses ‘View Agents’.  3. System retrieves and displays the information of all agents who are added into the system. |
| **Descriptive of Alternative Sequence** | 3a) If there are no available agent accounts have been created for the system, returns empty list. |
| **Postcondition** | Admin is redirected to his or her account dashboard page of the system. |

**View Booking**

|  |  |
| --- | --- |
| **Use Case** | View Booking |
| **Summary** | Admin views information of all bookings that have been made for the vessel schedule that is not removed yet. |
| **Dependency** | None |
| **Actors** | Admin |
| **Preconditions** | Admin is already logged into the system. |
| **Descriptive of Main Sequence** | 1. Admin chooses ‘View Booking’.  2. System retrieves and displays the information of all vessel schedule that are not removed yet.  3. Admin clicks on ‘View Booking’ button for the row of the specific vessel schedule that is wanted to be viewed.  4. System retrieves and displays the information of all bookings for the chosen vessel schedule. |
| **Descriptive of Alternative Sequence** | 2a) If there are no available vessel schedule, returns empty list.  4a) If there are no available bookings for the chosen vessel schedule, displays ‘There are no available bookings for this schedule’ message. |
| **Postcondition** | Admin is redirected to his or her account dashboard page of the system. |

**Create Schedule**

|  |  |
| --- | --- |
| **Use Case** | Create Schedule |
| **Summary** | Admin adds the vessel schedule |
| **Dependency** | <<extend>> Remove Event  <<include>> View Schedule |
| **Actors** | Admin |
| **Preconditions** | Admin is already logged into the system. |
| **Descriptive of Main Sequence** | 1. Admin selects dropdown of ‘Schedule’.  2. Admin chooses ‘Create Vessel Schedule’.  3. System retrieves and displays the empty vessel schedule form.  4. Admin enters vessel schedule details.  5. System displays “You have successfully created a vessel schedule” message.  6. Admin can choose to remove the vessel schedule that has been created. |
| **Descriptive of Alternative Sequence** | 5a) If the form is not completed, system returns ‘Please fill out this field’ message.  5b) If details that are provided by admin do not match the format required for the fields or duplicated, admin will prompt to re-enter the details. |
| **Postcondition** | Agent can view the vessel schedule |

**View Confirmed Booking**

|  |  |
| --- | --- |
| **Use Case** | View Confirmed Booking |
| **Summary** | Agent views the information of all bookings that have been successfully made by him or herself. |
| **Dependency** | None |
| **Actors** | Agent |
| **Preconditions** | Agent is already logged into the system. |
| **Descriptive of Main Sequence** | 1. Agent chooses ‘View Confirmed Booking’.  2. System retrieves and displays the information of all bookings that have been made by the agent |
| **Descriptive of Alternative Sequence** | 2a) If there are no available bookings have been made by the agent, returns empty list. |
| **Postcondition** | Agent is redirected to his or her account dashboard page of the system. |

**View Schedule**

|  |  |
| --- | --- |
| **Use Case** | View Schedule |
| **Summary** | Agent views all vessel schedule |
| **Dependency** | <<include>> Book Vessel |
| **Actors** | Agent |
| **Preconditions** | Agent is already logged into the system. |
| **Descriptive of Main Sequence** | 1. Agent chooses ‘Book Vessel’.  2. System retrieves and displays the information of all vessel schedule.  3. Agent reserves vessel schedule for his or her client’s goods transportation |
| **Descriptive of Alternative Sequence** | 2a) If there are no available vessel schedule, returns empty list. |
| **Postcondition** | Agent is redirected to his or her account dashboard page of the system. |

**Book Vessel**

|  |  |
| --- | --- |
| **Use Case** | Book Vessel |
| **Summary** | Agent books the vessel schedule for his or her client’s goods transportation |
| **Dependency** | <<include>> Register Item  <<include>> Register Customer |
| **Actors** | Agent |
| **Preconditions** | Agent is already viewed the information of all vessel schedule. |
| **Descriptive of Main Sequence** | 1. Admin clicks on ‘Book’ button for the row of the specific vessel schedule that is wanted to be reserved.  2. System retrieves and displays the empty booking form  3. Agent registers the customer  4. Agent registers the item of the customer which would like to be transported  5. System displays “You have successfully booked the vessel” message. |
| **Descriptive of Alternative Sequence** | 1a) If the departure date is earlier than the current date, system displays ‘You are not allowed to book the vessel which the departure date is over or on today.’ message  5a) If the form is not completed, system returns ‘Please fill out this field’ message.  5b) If details that are provided by agent do not match the format required for the fields or duplicated, agent will prompt to re-enter the details. |
| **Postcondition** | Agent proceed to view the details of all bookings that have been successfully made. |

### **3.2.3 Activity Diagram**

**Login**



**Register Agent**



**View Agent**



**View Booking**



**Create Schedule**



**View Confirmed Booking**



**View Schedule**



**Book Vessel**



### **3.2.4 Entity Relationship Diagram**



### **3.2.5 Cloud Architecture**



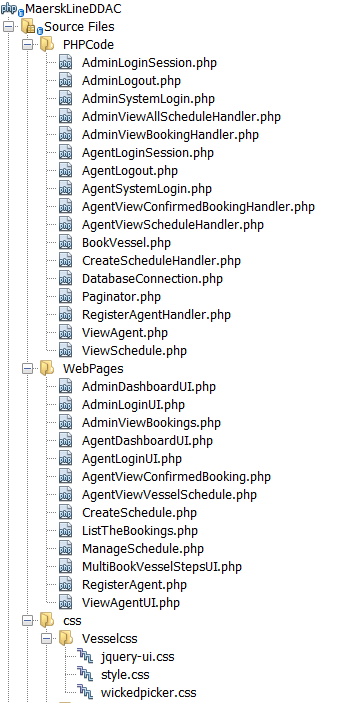
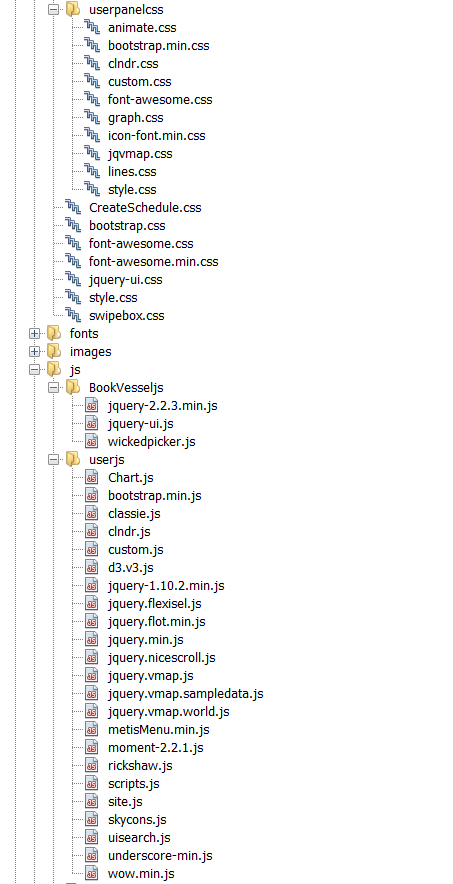
**Figure 3.2.5.1: Maersk Line’s CMS Cloud Architecture**

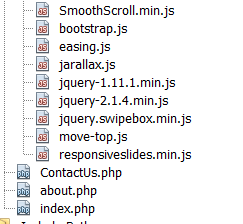
Figure 3.2.5.1 shows the cloud architecture that is designed to deploy the implemented Maersk Line’s CMS into Microsoft Azure virtualized platform. As mentioned in the design considerations section which is at sub-section 3.1 under section 3.0, Maersk Line’s CMS is currently only target towards one region where the main office of Maersk Line is positioned which is North Europe. Hence, there is only one primary web app is created for the developed Maersk Line’s CMS and place in North Europe region as well as under one resource group. Besides that, a MySQL database will be configured and deployed on North Europe region as well as under the same resource group where the web app is created to deploy Maersk Line’s CMS as the programming language that used to implement the web-based application is PHP and MySQL that is classified as the most suitable and common Database Management System (DBMS) used by the web-based applications that are developed using PHP. Furthermore, auto scaling will be configured to ensure high level of effectiveness in allocation of resources that can ensure performance of Maersk Line’s CMS despite peak seasons as well as avoid extensive usage of cost that could lead to over budget. Other than that, as mentioned in the design considerations section which is at sub-section 3.1 under section 3.0, Maersk Line’s CMS local environment which is localhost is classified as testing environment whereas Maersk Line’s CMS that deployed at Microsoft Azure virtualized platform is classified as production environment which therefore there is no utilisation of Continuous Delivery (CD) in Microsoft Azure virtualized platform. In addition, GitHub which is the repository that keeps the source codes for the developed Maersk Line’s CMS that are written in a set of files will be linking with the web app which will deploy the web-based application automatically as well as synced consistently if there are modifications performed on the set of files.

# **4.0 Implementation**

## **4.1 Application Development**

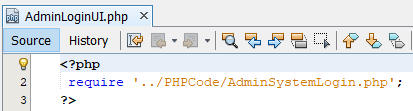
The development of Maersk Line’s Container Management System (CMS) as a web-based application is performed through web-based programming languages which are HTML, CSS and JavaScript. These programming languages are used to implement and support front-end of Maersk Line’s CMS that enable direct interaction among the users and the application such as screens. Apart from that, PHP which is an acronym of PHP Hypertext Preprocessor is also selected as the scripting language to be used in implementation back-end Maersk Line’s CMS. PHP is also known as programming language that can be embedded in HTML or used as a standalone binary (Converse, Park and Morgan, 2007). Besides that, PHP is a programming language that also plays an important role as a script-parsing engine that works in unison to assist the developer in creating dynamic content that helps to retrieve information out of database and inserting them into the web page of the web-based program or application (Cullen, 2002). Hence, PHP is used to enable interaction among Maersk Line’s CMS and its database such as retrieval of information from the database and display at the application as well as perform validation on the input that are provided by the users from front-end to ensure accuracy or correctness of the information in the application. Furthermore, MySQL is chosen as the database of Maersk Line’s CMS. MySQL is a type of Relational Database Management System (DBMS) that is developed by Oracle Corporation that supports multithreaded and multiusers and contains more than 10 million installations (Bassil, 2012). The primary query language that is used by MySQL is Structured Query Language (SQL) which was formalised by the American National Standards Institute (ANSI) in year 1986 (Darmawikarta, 2014). Therefore, MySQL is used to enable Maersk Line’s CMS for storing and retrieving the data to satisfy the information needs of users. **Figure 4.1.1** and **Figure 4.1.2** shows the structure of the files that are created for Maersk Line’s CMS.

**Figure 4.1.1: File Structure of Maersk Line’s CMS**

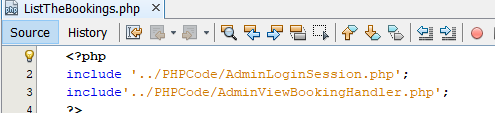


**Figure 4.1.2: File Structure of Maersk Line’s CMS**

The default file that will be executed first is index.php when Maersk Line’s CMS starts running which it displays the main or home page of the web-based application. The main page includes navigations to all web pages that have created for Maersk Line’s CMS which means that all requests of accessing to the features and functionalities that are offered by Maersk Line’s CMS made by the users are going through index.php file. Besides that, the web pages which are the front-end that are created to demonstrate the features and functionalities of Maersk Line’s CMS are stored inside WebPages folder shown in figure 4.1.1. Apart from that, the files that created for back-end of Maersk Line’s CMS are stored in PHPCode folder shown in figure 4.1.1. When the specific web page of Maersk Line’s CMS is accessed by the user to perform specific task such as reserve vessel schedule. The web page invokes the backend of the web page from PHPCode folder through ‘require’ or ‘include’ statement.

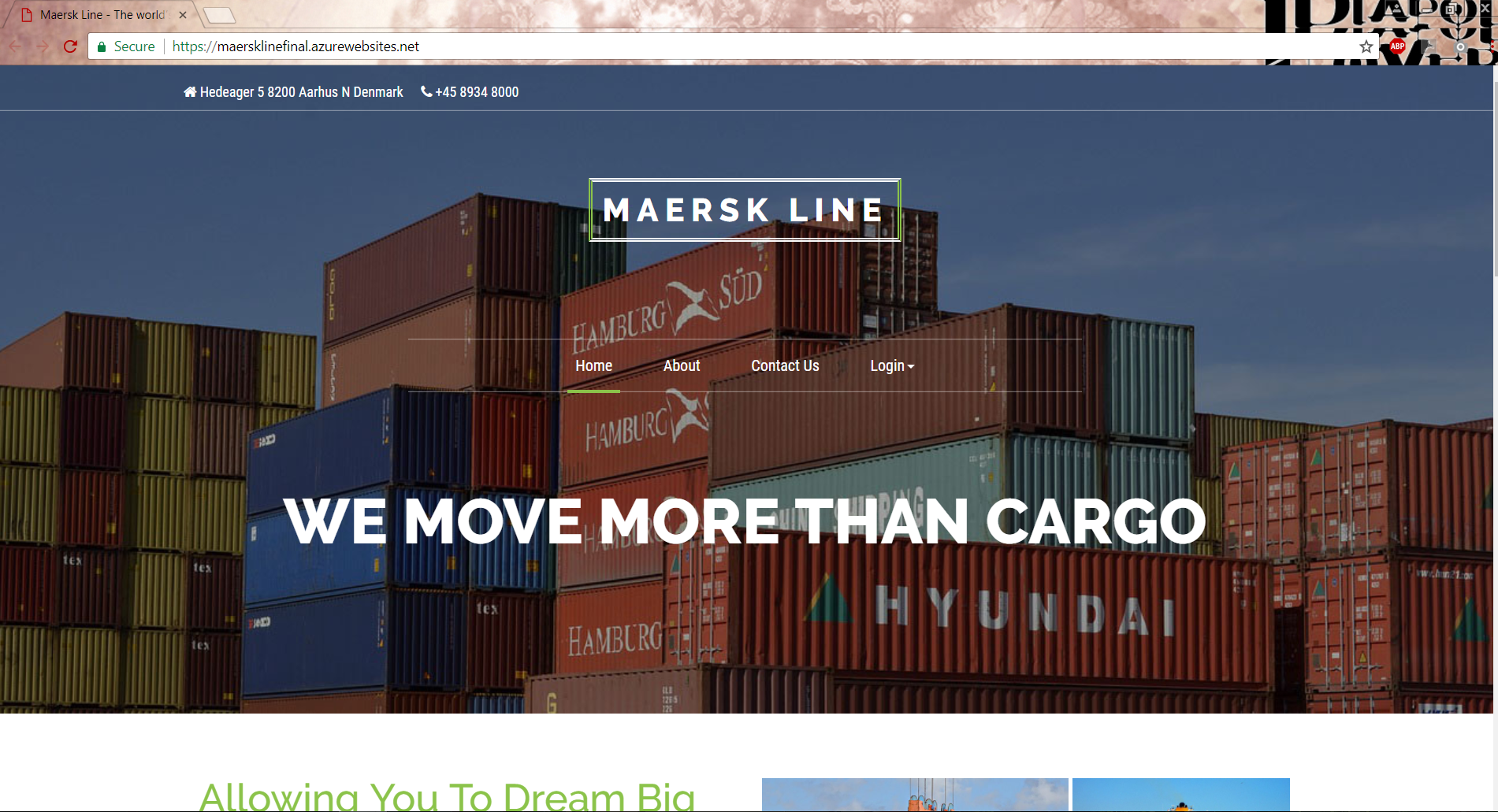


**Figure 4.1.3: Invoke backend statement**



**Figure 4.1.4: Invoke backend statement**

Figure 4.1.3 shows the login page of Maersk Line’s CMS for admin user which is AdminLoginUI.php that invokes the backend to help performing verification of the login credentials which are provided by the users through retrieval of information from the database for performing comparison with the input that is provided by the users through ‘require’ statement. Besides that, figure 4.1.4 shows the page of Maersk Line’s CMS that displays the list of bookings that have been made for the specific vessel schedule for the admin user which is ListTheBookings.php that invokes the backend to help verifying that the admin user has successfully logged into the web-based application before accessing the web page as well as retrieving the information from the database through ‘include’ statements. The development of Maersk Line’s CMS is concluded that the interaction between the front-end and back-end is very direct which means that the front-end that is the web page files invokes the back-end files inside the web page files itself to enable back-end tasks take place for ensuring the correctness and accuracy of the information in Maersk Line’s CMS. In addition, a screenshot is provided as an example of User Interface (UI) that is implemented for Maersk Line’s CMS which is the home page that designed using the web-based programming languages which have been mentioned above that consists of HTML, CSS and JavaScript shown in figure 4.1.5.



**Figure 4.1.5: Maersk Line’s CMS Home Page**

Finally, the version control of Maersk Line’s CMS is also performed throughout the development process of the web-based application to record changes that are performed on a set of files over time and enable recall of specific versions of the files. By performing version control on Maersk Line’s CMS, this can simplify the process of reverting the changes back to the previous state of the web-based application if there are any errors or negative impact occurs after performing modifications or changes on the files. The tool that chosen to assist the version control throughout the implementation of Maersk Line’s CMS is GitHub which stores the source code of the web-based application into a set of files which the link that allows viewing of the source code is <https://github.com/xinwanc/MaerskLineDDAC>. Other than that, a video that demonstrates the implemented Maersk Line’s CMS is also created and the link that allows watching the video to gain insights or deeper understanding on how the web-based application works is <https://web.microsoftstream.com/video/e46b2b5f-64a6-4805-ac4a-ae81ba384ba7>. Other than that, the link that allows to view the softcopy of the report that documents the entire Software Development Life Cycle (SDLC) of Maersk Line’s CMS is available at <https://github.com/xinwanc/TP034688DDACAssignment>.

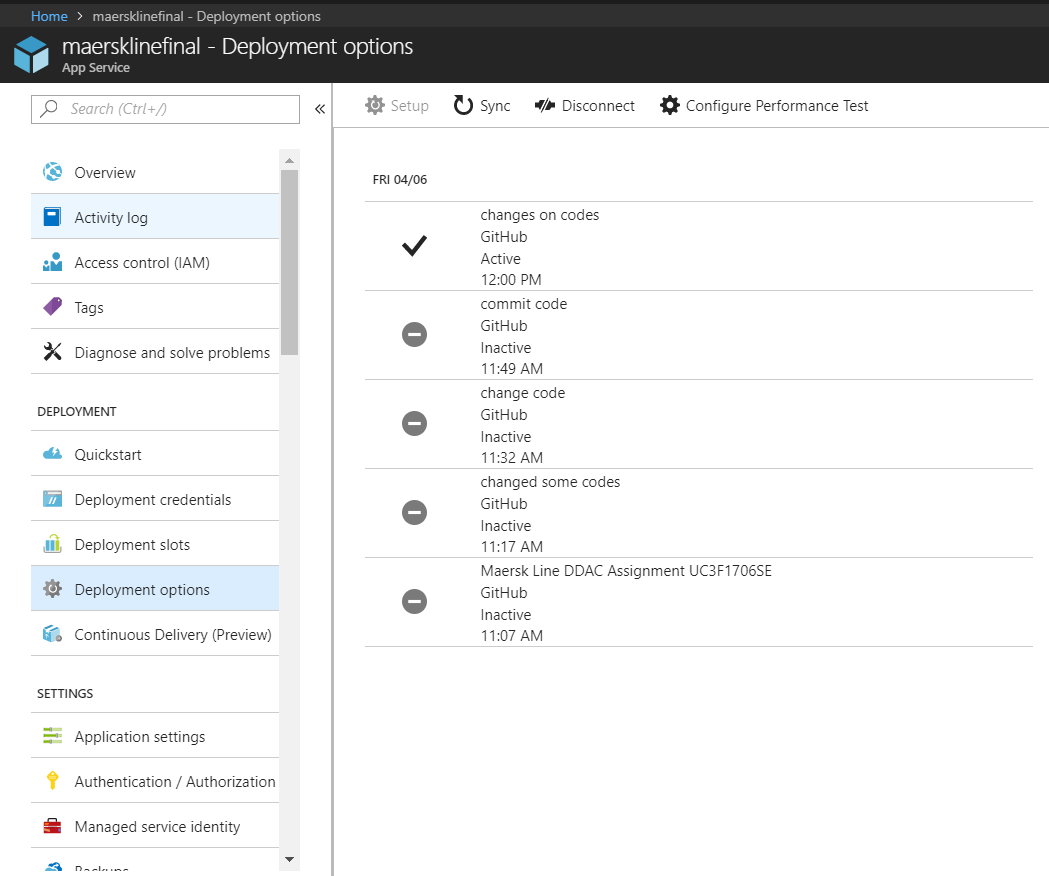
## **4.2 Microsoft Azure Publishing**

Maersk Line’s CMS is published into Microsoft Azure virtualised platform by creating a new resource which is ‘Web App+MySQL’ and then provide the app name which will be used to help executing Maersk Line’s CMS on web browser, resource group that keeps all the resources of the deployed Maersk Line’s CMS which consists of the source code, app service plan or region and MySQL database, selection of the app service plan which is the demographic region to host Maersk Line’s CMS as well as creation of MySQL database. Figure 4.2.1 shows the information of the resource that is created for deploying Maersk Line’s CMS in Microsoft Azure.



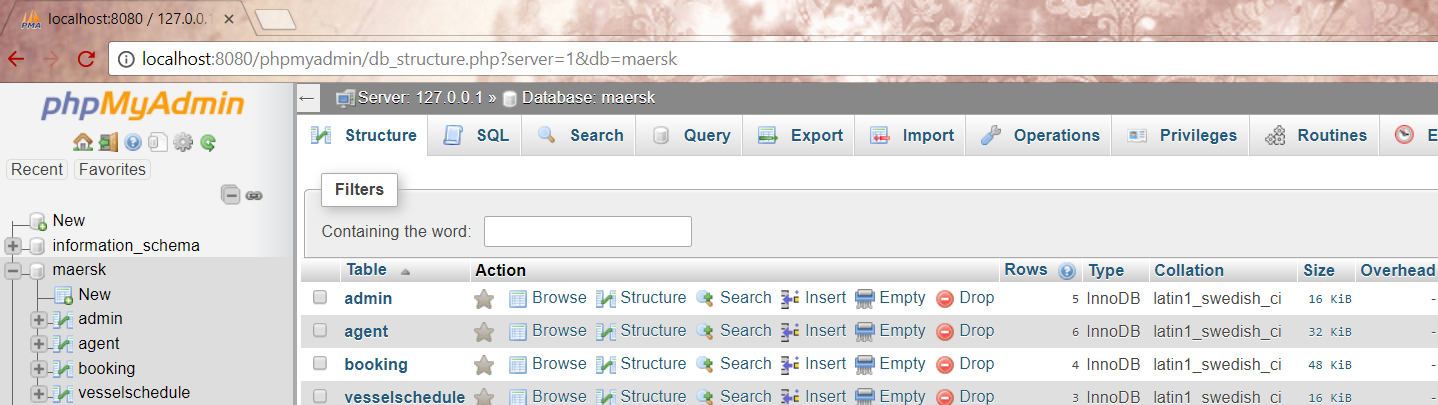
**Figure 4.2.1: Resource Created for Maersk Line’s CMS in Microsoft Azure**

After creating the resource for Maersk Line’s CMS, the deployment process of the web-based application into Microsoft Azure virtualized platform is performed by moving the source codes which are stored in a set of files for Maersk Line’s CMS into the created resource. The deployment method that is chosen to publish the web-based application is GitHub by linking the created resource with the selected name of the project in GitHub after permitting access to the projects that are available in the user account of GitHub. GitHub option is known as automatic deployment options which it deploys the changes that are made on a set of files which are already committed and pushed to the specific branch in GitHub into Microsoft Azure virtualized platform automatically and consistently. This has provided convenience in performing necessary changes on the set of files of Maersk Line’s CMS as they will be automatically deployed into Microsoft Azure virtualized platform instead of keep re-upload the specific file through File Transfer Protocol (FTP). Figure 4.2.2 shows the branch history of GitHub which is also consistently deployed into Microsoft Azure if there are modifications have been made.



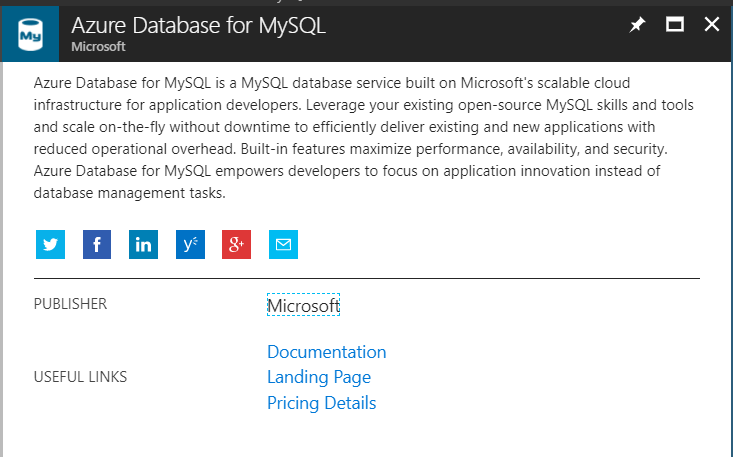
**Figure 4.2.2: Branch history**

After success deployment of the implemented Maersk Line’s CMS into Microsoft Azure virtualized platform, the process of integrating MySQL database that is created at Microsoft Azure virtualized platform with the deployed Maersk Line’s CMS is performed as Maersk Line’s CMS is currently integrating with MySQL database that has been created in localhost which the data of the database can be accessed and maintained through phpMyAdmin that is shown in figure 4.2.3. Hence, exception will be throw when the web page of Maersk Line’s CMS that requires database operation is accessed in Microsoft Azure virtualized platform.



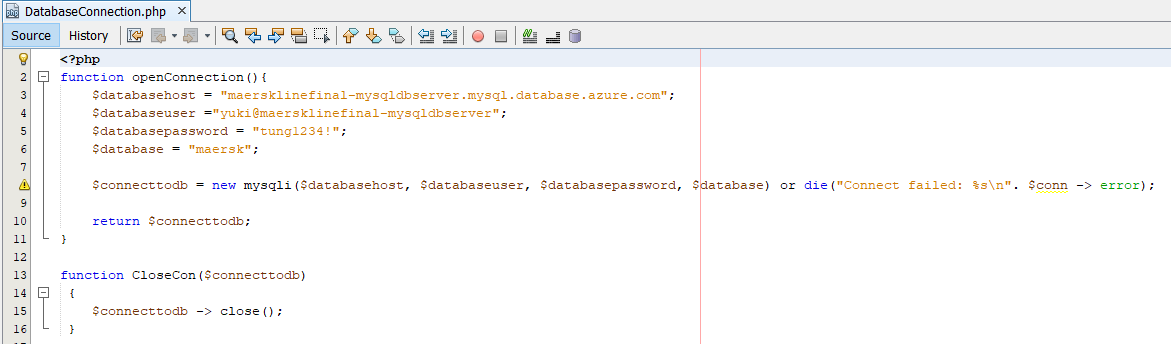
**Figure 4.2.3: MySQL database in local environment**

In Microsoft Azure virtualized platform, the database that has been created for integrating with Maersk Line’s CMS is MySQL server instead of a SQL server which is shown in figure 4.2.4 because the it has been mentioned that the implementation of Maersk Line’s CMS is performed using PHP programming language and MySQL as DBMS. Apart from that, the features that are offered by Microsoft Azure virtualized platform for MySQL database is same as SQL database such as scaling and monitoring (Microsoft, 2017).

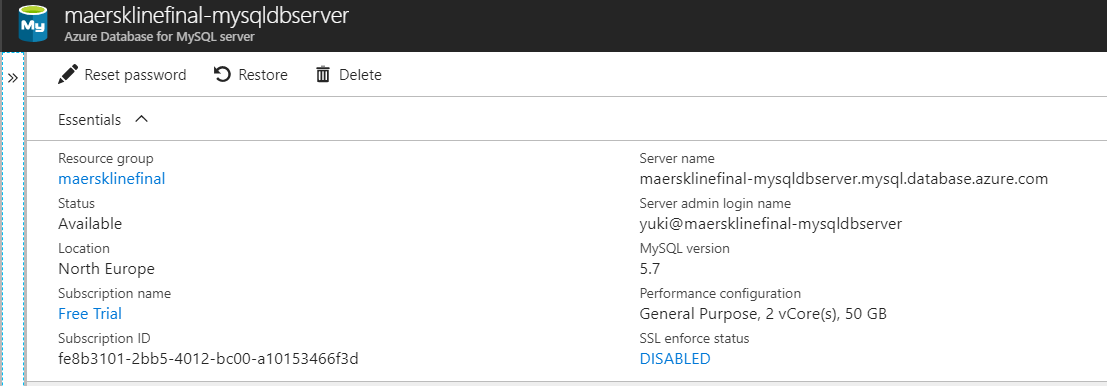


**Figure 4.2.5: Azure database for MySQL**

The integration process of MySQL database that is created with Maersk Line’s CMS in Microsoft Azure virtualized platform is initiated by modifying the connection string of Maersk Line’s CMS to the MySQL database in Microsoft Azure virtualized platform in DatabaseConnection.php that is shown in figure 4.2.6 by referring to the details of MySQL database in Microsoft Azure virtualized platform which consist of server name, server admin login name, server admin password and name of database schema that is shown in figure 4.2.7. This enable Maersk Line’s CMS to grant the authority of accessing MySQL database that is stored and maintained at Microsoft Azure virtualized platform.

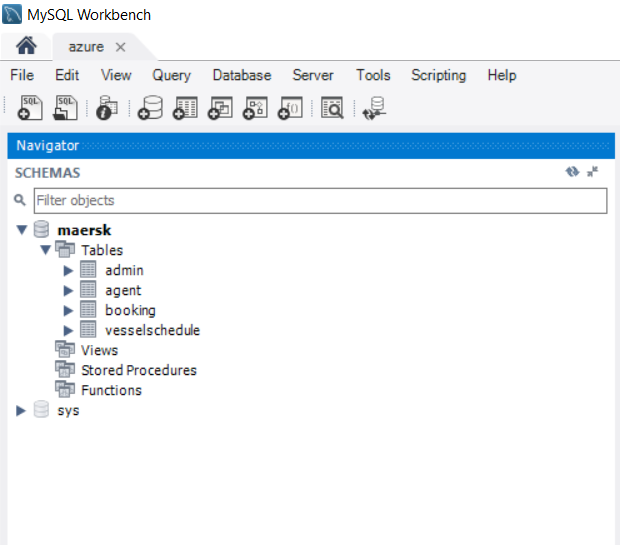


**Figure 4.2.6: Connection String**



**Figure 4.2.7: Details of Azure MySQL database**

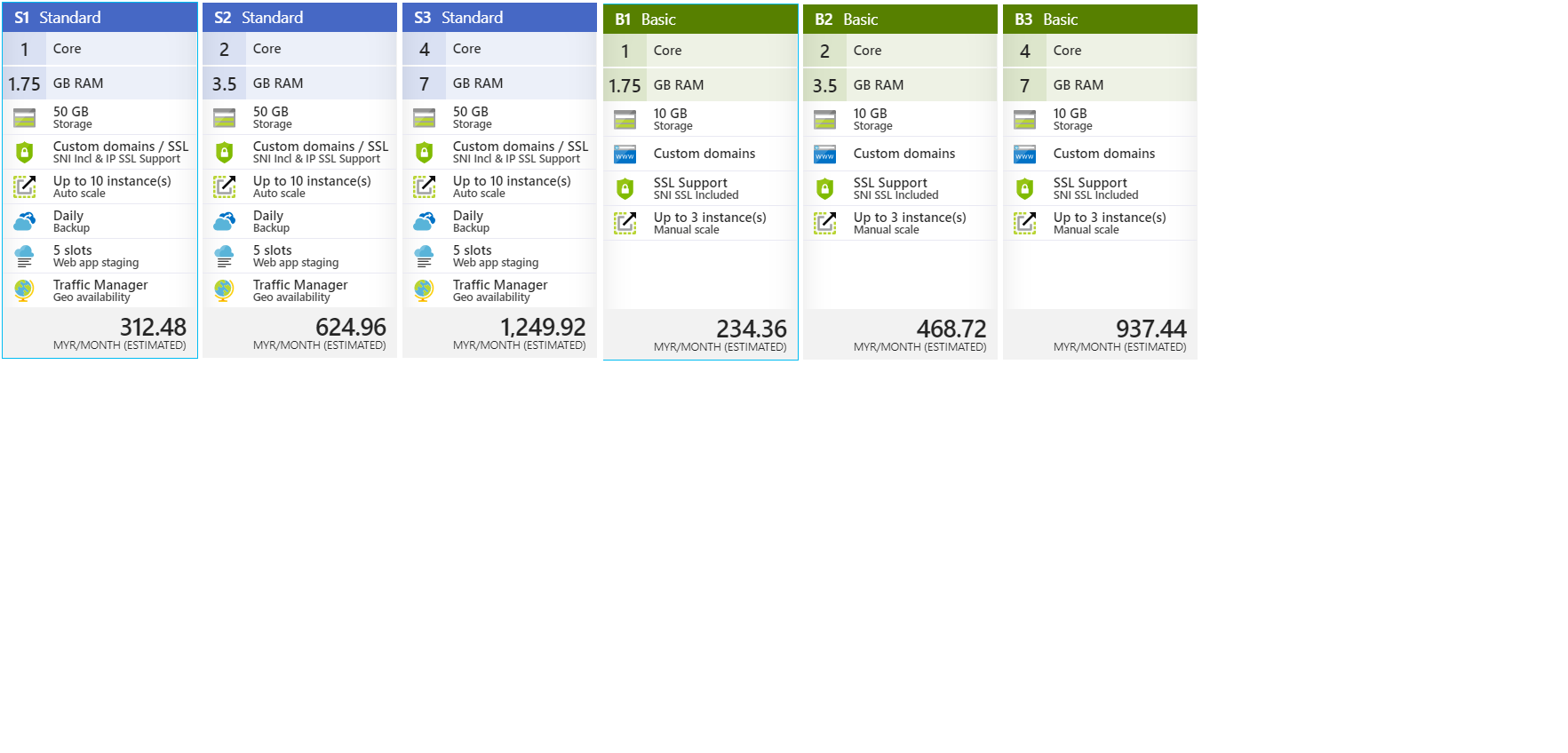
Apart from that, as there is already a database has been created and integrated with Maersk Line’s CMS when the web-based application is running at local environment before deploying in Microsoft Azure virtualized platform. Hence, there are tables and data have been existed for Maersk Line’s CMS before deploying it in Microsoft Azure virtualized platform which means that migration of tables and data from the existing database at phpMyAdmin into MySQL database in Microsoft Azure virtualized platform is required to be performed for ensuring consistency of the data. To perform migration process, MySQL Workbench which is an open-source software tool that is developed by Oracle Corporation for helping to manage MySQL servers and databases is used to grant access to MySQL database in Microsoft Azure virtualized platform for performing import of existing tables and data into the specific database schema that shown in figure 4.2.8 (Dev.mysql.com, 2018). Besides that, MySQL Workbench also allows maintaining the data and tables that are created in MySQL database at Microsoft Azure virtualized platform easily.



**Figure 4.2.8: MySQL Workbench**

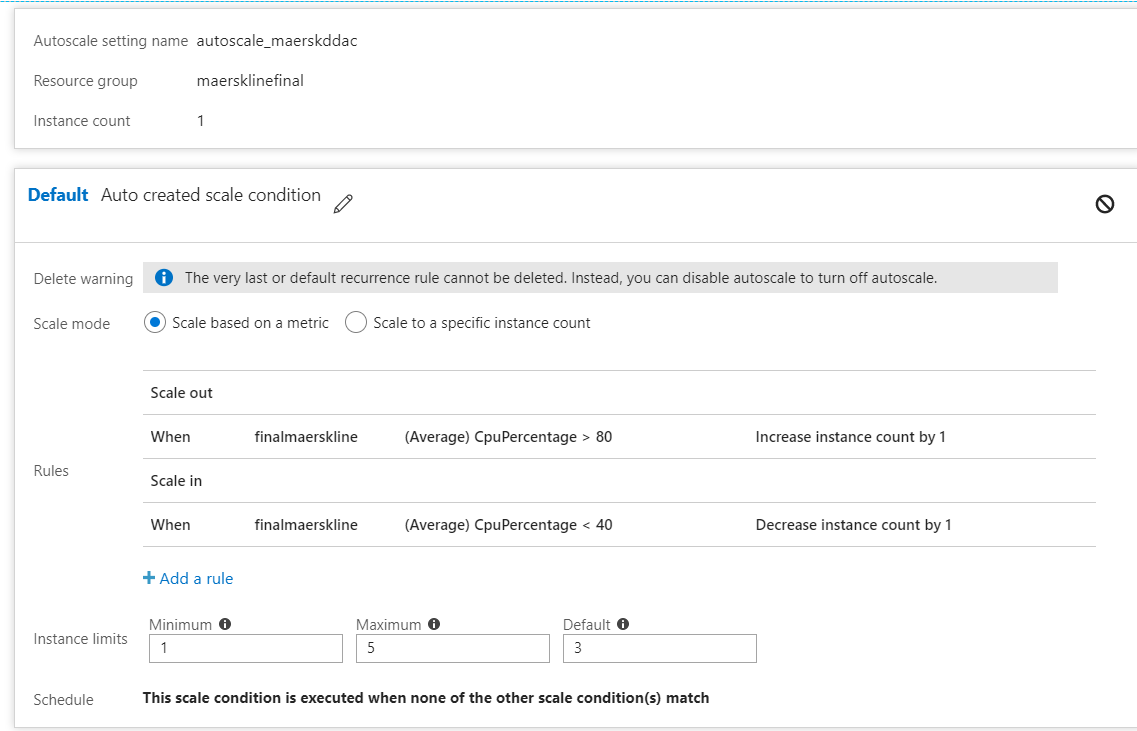
## **4.3 Application Scaling**

Microsoft Azure virtualized platform enables auto scaling which is a feature that helps to allocate the resources dynamically for satisfying the performance requirements which can ensure that the application has the ability of maintaining the desired performance levels and satisfy service-level agreements (SLAs) even the volume of work increases which therefore additional resources is may be required by the web-based application as well as has the ability of de-allocating the resources if the demand loosens which therefore additional resources are no longer required to minimise the costs (Microsoft Azure, 2017). Hence, auto scaling plays an essential role for Maersk Line’s CMS for promoting high level of efficiency level in allocating the resources throughout the execution of Maersk Line’s CMS which can accomplish performance quality goal of the web-based application running at Microsoft Azure virtualized platform and at the same time maintaining high level of cost-effectiveness because auto scaling helps to prevent wastage on allocation of extra resource during high peak when they are automatically de-allocated if they are no longer required in order to minimise the operation cost. Apart from that, auto scaling also provides the advantage of eliminating the need of hiring a dedicated personal or expert to always supervise the performance of the web-based application and making decision on whether to proceed with allocating or de-allocating more resources. Figure 4.3.1 shows the price tier for the category of app service plan that are taken into consideration for supporting Maersk Line’s CMS.



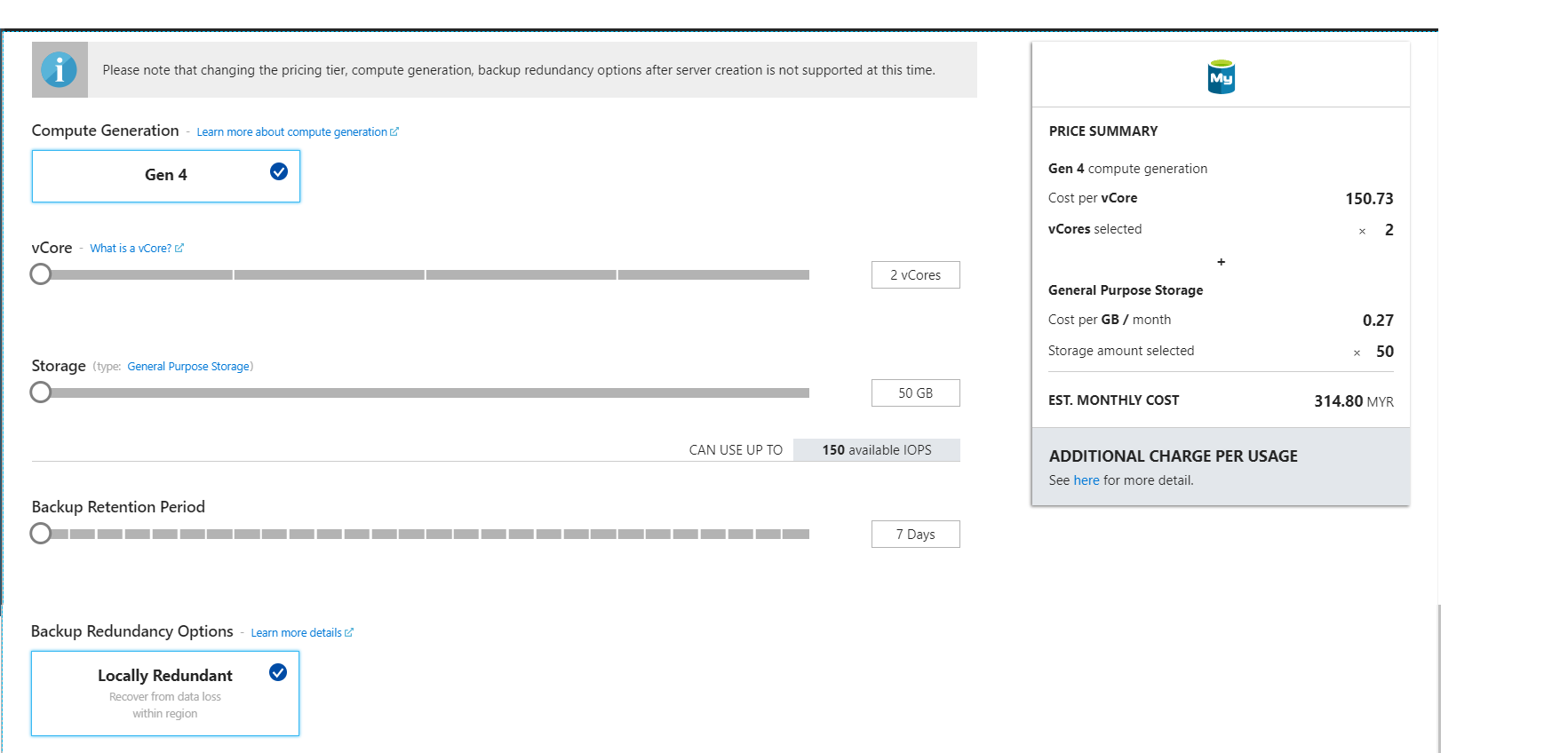
**Figure 4.3.1: Price Tier of App Service Plan**

The categories of app service plan that are taken into consideration for Maersk Line’s CMS are basic and standard which offers different features with different prices. The final decision has been made on the selection of app service plan for Maersk Line’s CMS which is Standard app service plan. Maersk Line’s CMS which is deployed to North Europe region will be using S1 Standard app service plan because of the limited amount of credit or budget that is given by Microsoft Azure virtualized platform which is RM850.00. The reason of having more preference in selecting Standard app service plan instead of Basic app service plan is because it provides essential features that facilitates the process of measuring the performance of Maersk Line’s CMS in North Europe region before approving the web-based application to be ready for releasing to other regions for public use. First and foremost, Standard app service plan offers the minimum of 50GB of storage which is greater than any basic plan and this feature is greatly considered because it enables Maersk Line’s CMS to have the ability of handling a larger amount of transactions within a given period easily. Besides that, Standard app service plan also offers custom domain and SSL which enable support of SNI Incl & SSL Support that is additional benefit compared to basic plan. Other than that, Standard app service plan also offers auto scaling feature which can be reached up to 10 instances and this enable the web-based application to be auto scaled out and in according to the metrics such as server load, memory usage or even on a set schedule (Luijbregts, 2017). This feature is greatly considered as it is extremely useful to determine the accurate performance of Maersk Line’s CMS which will be visited by all users in North Europe region and expanded to other regions in future. Besides that, the daily backup of the configuration and server data is offered by Standard app service plan which this feature is greatly considered because it can help to avoid permanent loss of critical data and simplifies the recovery process which can maintain the ability of Maersk Line in dealing with business activities as usual.



**Figure 4.3.2: Maersk Line’s CMS Scaling Plan**

Figure 4.3.2 shows the scaling plan that is created for Maersk Line’s CMS for helping to adjust the allocated resources to ensure high level of performance for the web-based application while it is executed in Microsoft Azure virtualized platform that is mainly targeted at North Europe region. The application instances is fixed to 3 only as default in order to ensure that the cost has been spent is within the budget amount that is given which is RM850.00 because the higher application instances requires higher amount of cost to be spent. There are two rules have been created in terms of average CPU utilisation while configuring the auto scale feature for Maersk Line’s CMS in North Europe server or region which consist of scale out and scale in. The first rule is when the average percentage of CPU usage is more than 80% for the past 10 minutes, the application instance will be increased by 1. Besides that, the second rule is when the average percentage of CPU usage is less than 40% for the past 10 minutes, the application instance will be reduced by 1. Furthermore, the application instance limit is also configured which the minimum application instance is 1 and maximum application instance 3 to ensure efficient allocation of resources which can avoid Maersk Line’s CMS to scale more than the amount that could be afforded which could lead to over budget. Apart from that, the application service plan subscription will be upgraded to premium or isolated app service plan to accommodate the expanded business needs in future as premium or isolated app service plan offers more and improved features such as higher backup frequencies and better hardware such as SSDs (Lin, 2016).



**Figure 4.3.3: MySQL Database Pricing Tier**

Figure 4.3.3 shows the pricing tier that is chosen for MySQL database of Maersk Line’s CMS in Microsoft Azure virtualized platform. The plan that is selected for MySQL database is general purpose plan as it can support up to 32 vCore enable prediction of IO performance. Besides that, the MySQL database is configured to use Gen 4 as compute generation, 2vCores which should be sufficient for serving Maersk Line’s CMS which currently has only one region, 50GB for storage size and backup retention period of seven days which is the shortest period and locally redundant as option of backup redundancy because Maersk Line’s CMS is deployed in one region only. The lowest configuration possible has been selected for MySQL database in order to prevent occurrence of over budget. Apart from that, a greater plan will be subscribed to handle greater amount of transaction that is required for supporting the growing business needs as well as the backup redundancy option will change to geo-redundant if Maersk Line’s CMS has expanded its usage to other regions in future.

## **4.4 Reliability and Performance**

Reliability and performance are the key quality goals that mandatory to be achieved by Maersk Line’s CMS to provide the ability for Maersk Line in supporting the business operation of the organisation over time which enable meeting the needs of demands despite peak seasons. Measures have been taken to ensure reliability and performance of Maersk Line’s CMS executed in Microsoft Azure virtualized platform. As Maersk Line’s CMS is focusing on one region only, therefore the default application instances for running the web-based application is 3 for promoting higher reliability which could scaled out to 5 instances. Besides that, load balancer is offered by Microsoft Azure virtualized platform that integrates with Maersk Line’s CMS which distributes the tasks among all the application instances (Microsoft, 2017). This can help to increase the effectiveness level in performing resources allocation by Maersk Line’s CMS that could result in better throughput as well as reliability. Apart from that, Microsoft Azure virtualized platform also offers automation on the process of scaling out the application instances according to metrics that are prefixed by the users such as schedule or response time (Cavale, et al., 2017). Furthermore, auto scaling is also applied for the application instances to handle the unexpected workload which can help to ensure that Maersk Line’s CMS can be operated and accessible by the users despite peak seasons which creates congestion of workloads. Apart from that, proper indexing is also used to optimise MySQL server instance internally as suggested by database tuning advisors. As for now, there is only one instance of MySQL server kept in North Europe which the one and only region focussed by Maersk Line’s CMS. Besides that, geo-replication will be configured for increasing the data integrity as well as minimise the possibility of massive data loss when Maersk Line’s CMS is expanded to other regions in the future. This can enable Maersk Line’s CMS has the ability to continue assist business activities performed by the organisation as well as provide airtight protection or security for the data from the disaster.

# **5.0 Testing**

## **5.1 Unit Testing**

Unit testing is conducted on the unit or function of Maersk Line’s CMS separately to verify the individual units or functions are working properly which can help to ensure that they are behaving in an expected manner. Besides that, focusing in individual units or functions only that is the concept of unit testing enables the tracing of root cause on defects or errors easily which therefore allows fixing of the defects or errors that are identified in Maersk Line’s CMS earlier as well as ease the debugging process as the defects or errors can be easily detected through testing that is conducted on small unit or functions. Unit testing is performed on Maersk Line’s CMS locally which is localhost that classified as testing environment for the web-based application unless otherwise stated and can be conducted on the production environment of Maersk Line’s CMS which is deployed in Microsoft Azure virtualized platform by changing the instances of ‘localhost’ to the hosted domain name if there are any modifications occurred associated to enhancement.

**5.1.1 Login (Admin and Agent)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML01-01 | Enter correct username and password | Display ‘Congratulations you have successfully login!’ message | Display ‘Congratulations you have successfully login!’ message | Pass |
| ML01-02 | Enter invalid username or password | Display ‘Sorry, your username or password is wrong or the account is not exist. Please double check.’ message | Display ‘Sorry, your username or password is wrong or the account is not exist. Please double check.’ message | Pass |
| ML01-03 | Empty field | Display ‘Please fill out this field’ message | Display ‘Please fill out this field’ message | Pass |

**5.1.2 Register Agent**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML02-01 | Fill in all the fields correctly | Display ‘You have successfully created an agent’ message | Display ‘You have successfully created an agent’ message | Pass |
| ML02-02 | Does not fill in all the fields | Display ‘Please fill out this field’ message | Display ‘Please fill out this field’ message | Pass |
| ML02-03 | Enter exist username | Display ‘Username has exist’ message | Display ‘Username has exist’ message | Pass |
| ML02-04 | Enter special characters and numeric into first name | Display ‘Please key in alphabets only’ message | Display ‘Please key in alphabets only’ message | Pass |
| ML02-05 | Enter special characters and numeric into last name | Display ‘Please key in alphabets only’ message | Display ‘Please key in alphabets only’ message | Pass |
| ML02-06 | Enter special characters and spaces into username | Display ‘Only alphanumeric characters allowed’ message | Display ‘Only alphanumeric characters allowed’ message | Pass |
| ML02-07 | Enter different passwords for ‘Password’ and ‘Confirm Password’ field | Display ‘Password Not Match’ message | Display ‘Password Not Match’ message | Pass |
| ML02-08 | Enter special characters or spaces for passwords | Display ‘Only alphanumeric characters allowed’ message | Display ‘Only alphanumeric characters allowed’ message | Pass |
| ML02-09 | Enter exist email | Display ‘Email Exist’ message | Display ‘Email Exist’ message | Pass |
| ML02-10 | Enter wrong email format | Display ‘Email format is wrong’ message | Display ‘Email format is wrong’ message | Pass |
| ML02-11 | Enter duplicated details | Display ‘The agent has been registered’ message | Display ‘The agent has been registered’ message | Pass |

**5.1.3 View Agent**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML03-01 | Agent data is available | Display data of all agents in table and pages of 5 agents per page | Display data of all agents in table and pages of 5 agents per page | Pass |
| ML03-02 | Agent data is unavailable | Display empty table | Display empty table | Pass |

**5.1.4 View Booking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML04-01 | Booking data is available for the chosen vessel schedule | Display data of all bookings in table and pages of 5 bookings per page | Display data of all bookings in table and pages of 5 bookings per page | Pass |
| ML04-02 | Booking data is unavailable for the chosen vessel schedule | Display ‘There are no available bookings for this schedule’ message | Display ‘There are no available bookings for this schedule’ message | Pass |

**5.1.5 Create Schedule**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML05-01 | Fill in all the details correctly | Display ‘You have successfully created a vessel schedule’ message | Display ‘You have successfully created a vessel schedule’ message | Pass |
| ML05-02 | Enter details of vessel schedule that has been existed | Display ‘The vessel schedule has been created’ message | Display ‘The vessel schedule has been created’ message | Pass |
| ML05-03 | Didn’t fill in all the fields | Display ‘Please fill out this field’ message | Display ‘Please fill out this field’ message | Pass |
| ML05-04 | Enter date of the departure date is earlier or same as current date | Display ‘The departure should not be less than or equal to current date’ message | Display ‘The departure should not be less than or equal to current date’ message | Pass |
| ML05-05 | Enter date of the arrival date is earlier or same as departure date | Display ‘The arrival date should not be same as or before departure date’ message | Display ‘The arrival date should not be same as or before departure date’ message | Pass |

**5.1.6 Remove Schedule**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML06-01 | Click on ‘Delete’ button located in the specific row of the vessel schedule table which its arrival date is later than current date | Display ‘You are not allowed to remove the schedule as the arrival date is not over yet.’ message | Display ‘You are not allowed to remove the schedule as the arrival date is not over yet.’ message | Pass |
| ML06-02 | Click on ‘Delete’ button located in the specific row of the vessel schedule table which its arrival date is earlier or same as current date | Display ‘You have successfully remove the vessel schedule’ message | Display ‘You have successfully remove the vessel schedule’ message | Pass |

**5.1.7 View Confirmed Booking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML07-01 | Booking data is available | Display data of all bookings in table and pages of 5 bookings per page | Display data of all bookings in table and pages of 5 bookings per page | Pass |
| ML07-02 | Booking data is unavailable | Display empty table | Display empty table | Pass |

**5.1.8 View Schedule**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML08-01 | Vessel schedule data is available | Display data of all vessel schedule in table and pages of 5 vessel schedule per page | Display data of all vessel schedule in table and pages of 5 vessel schedule per page | Pass |
| ML08-02 | Vessel schedule data is unavailable | Display empty table | Display empty table | Pass |

**5.1.9 Book Vessel**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML09-01 | Click on ‘Book’ button located in the specific row of the vessel schedule table which its departure date is earlier or same as current date | Display ‘You are not allowed to book the vessel which the departure date is over or on today.’ message | Display ‘You are not allowed to book the vessel which the departure date is over or on today.’ message | Pass |
| ML09-02 | Click on ‘Book’ button located in the specific row of the vessel schedule table which its departure date is later than current date | Display empty vessel booking form | Display empty vessel booking form | Pass |
| ML09-03 | Click on ‘Book’ button located in the specific row of the vessel schedule table which its cargo capacity is 0 which is not available | Display ‘Sorry, the vessel schedule is fully booked’ message | Display ‘Sorry, the vessel schedule is fully booked’ message | Pass |

**5.1.10 Register Customer**

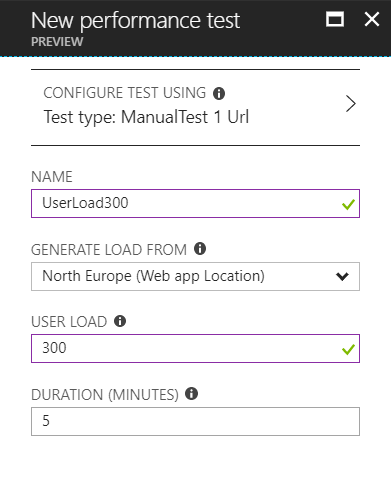
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML10-01 | Fill in all the fields correctly | Display ‘You have successfully created an agent’ message | Display ‘You have successfully created an agent’ message | Pass |
| ML10-02 | Does not fill in all the fields | Display ‘Please fill out this field’ message | Display ‘Please fill out this field’ message | Pass |
| ML10-03 | Enter exist contact number | Display ‘The contact number has been existed for another customer’ message | Display ‘The contact number has been existed for another customer’ message | Pass |
| ML10-03 | Enter wrong contact number format | Display ‘The contact number format is invalid, the format should be like 03-12345780’ message | Display ‘The contact number format is invalid, the format should be like 03-12345780’ message | Pass |
| ML10-04 | Enter exist email | Display ‘The email has existed for another customer’ message | Display ‘The email has existed for another customer’ message | Pass |
| ML10-05 | Enter wrong email format | Display ‘Please include an @ in the email address.’ message | Display ‘Please include an @ in the email address.’ message | Pass |
| ML10-06 | Enter duplicated details | Display ‘You have already registered the customer and item for this vessel schedule’ message | Display ‘You have already registered the customer and item for this vessel schedule’ message | Pass |

**5.1.11 Register Item**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Description** | **Expected Results** | **Actual Results** | **Status (Pass/Fail)** |
| ML11-01 | Enter cargo capacity that is more than available cargo capacity | Display ‘Your cargo space exceed. Please re-enter’ message | Display ‘Your cargo space exceed. Please re-enter’ message | Pass |

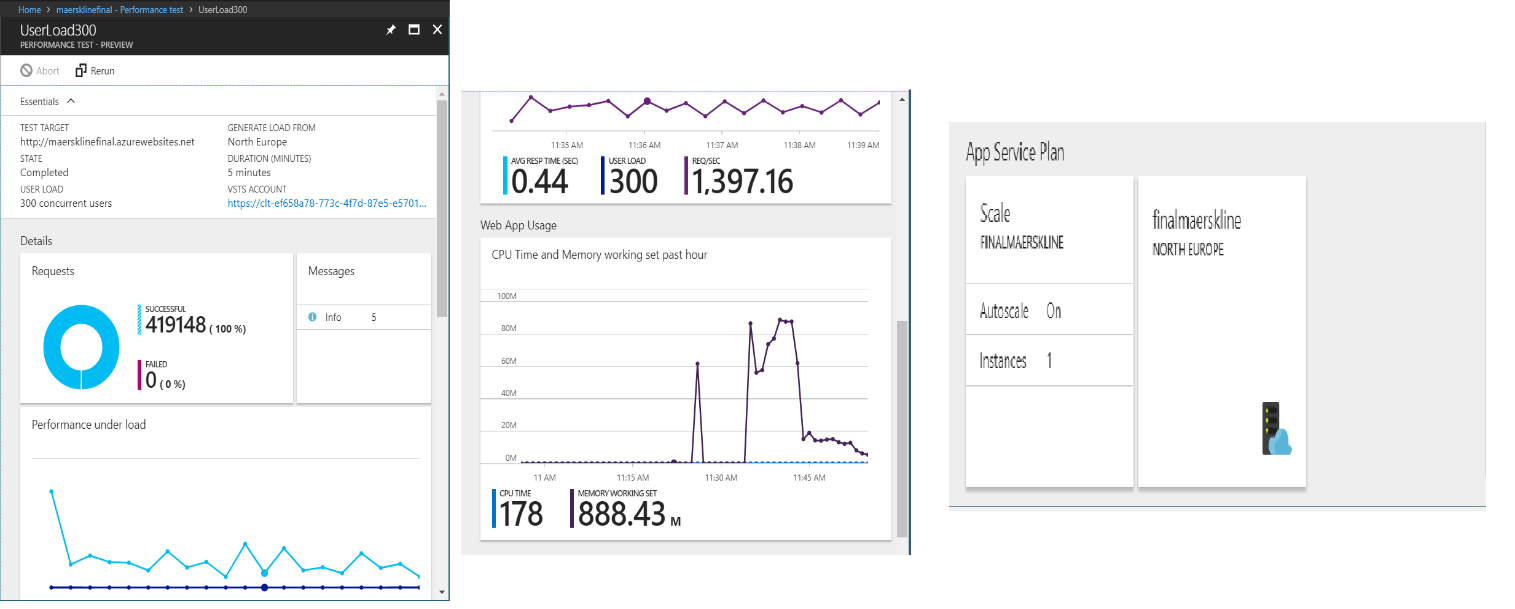
## **5.2 Performance Testing**

Performance testing is conducted on Maersk Line’s CMS to determine the responsiveness of the web-based application to execute any action within a given time interval. Besides that, performance testing is performed on Maersk Line’s CMS that is deployed in Microsoft Azure virtualized platform which is classified as production environment for the web-based application to ensure that Maersk Line’s CMS has the capability of accomplishing high level of performance level when it is accessed by huge numbers of users to perform tasks using the features and functionalities offered by the web-based application throughout North Europe region which can provide sufficient confidence for the organisation in expanding the usage of Maersk Line’s CMS to other regions in future due to its performance assurance. The performance testing of Maersk Line’s CMS is conducted through simulation of users visiting the web-based application which is known as the feature or functionality offered by Microsoft Azure during creation of the resource for deploying Maersk Line’s CMS. In addition, performance testing will be performed using three different categories of Standard app service plan that consists of S1, S2 and S3 as well as numbers of user load within a given period. The target number of user load is between **300** to **900** with each test increases by 200 user increments in user load. Figure 5.2.1 shows an example for configuration of simulation on numbers of users in accessing Maersk Line’s CMS associated to performance testing.



**Figure 5.2.1: Performance Test**

According to figure 5.2.1, the performance testing is taking place at North Europe which is the one and only region where Maersk Line’s CMS is deployed at and with simulation of 300 users in accessing the web-based application as well as lasting for 5 minutes. Additionally, the results that is produced by the performance testing consist of a pie chart that illustrates successful and failed request in percentage, status messages, a graph that illustrates performance under load, a graph that illustrates CPU usage and memory time as well as application scaling operation that are shown in figure 5.2.2.



**Figure 5.2.2: Performance Testing Results**

The gathering of results from performance testing that is conducted for different user loads and category of Standard app service plan is mainly focusing on average response time recorded in seconds and numbers of failed requests which are documented and showed in table 5.2.3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Concurrent Users  App Service Plan | 300 | 500 | 700 | 900 |
| S1 | 0.44 Sec  0 Failed | 0.72 Sec  0 Failed | 1.06 Sec  0 Failed | 1.01 Sec  0 Failed |
| S2 | 0.22 Sec  0 Failed | 0.77 Sec  0 Failed | 0.43 Sec  2 Failed | 0.47 Sec  2 Failed |
| S3 | 0.17 Sec  0 Failed | 0.82 Sec  0 Failed | 0.41 Sec  0 Failed | 0.73 Sec  0 Failed |

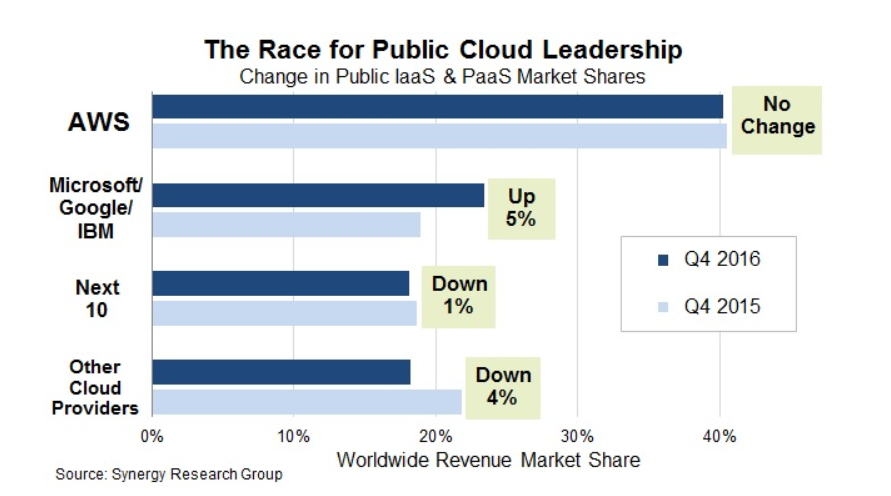
**Table 5.2.3: Maersk Line’s CMS Performance Testing in Different Service Plan**

**5.2.1 Analysis**

According to the results that are gained from the performance testing conducted towards Maersk Line’s CMS executing at Microsoft Azure virtualized platform, it is concluded that the higher tier or category of app service plan results in better ability of handling huge number of concurrent requests. Based on the testing results that are received from different app service plan for different numbers of user load, it can be seen that S3 has showed a significant increase in performance over both S2 and S1 as it has the ability in handling 300 to 900 concurrent users in less than 1 seconds in each test that increases the number of concurrent users by 200 without occurrence of failures. This has demonstrated a convincing evidence on the statement that mentioned above which is the higher tier of app service plan is more capable in maintaining the high level performance of Maersk Line’s CMS. From the perspective of average response time which is recorded in seconds, it can be seen that both S2 and S3 are able to demonstrate rapid average response time which is less than 1 seconds in overall compared to S1 despite the average response time of S2 and S3 in handling 300 concurrent users is slightly slow compared to S1. However, from the perspective of reliability which is recorded in numbers of failed requests, it can be seen that S1 and S3 can reliably handling 300 to 900 concurrent users without occurrence of failures compared to S2 which has 2 failed requests while handling 700 and 900 concurrent users. Hence, it is determined that S3 is the most preferred tier of app service plan to be used for supporting Maersk Line’s CMS as it not only has faster overall response time which is less than 1 seconds and also can reliably handling 300 to 900 concurrent users without occurrence of failures.

# **6.0 Managed Databases**

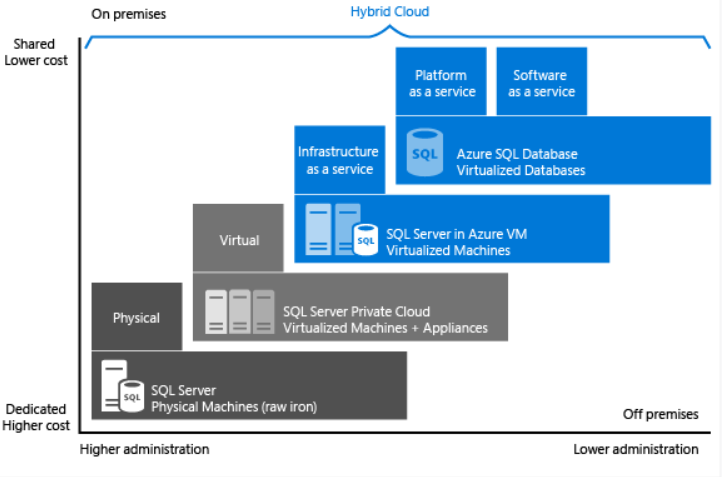
The growing popularity of cloud and managed services which are available online have increase the availability for huge numbers of services that were only accessible through a locally provisioned server previously through the click of a button. The concept of Platform as a Service (PaaS) has played an essential role to the market as a tool that tries to shave the time between development and deploying off constantly. Platform as a service (PaaS) is known as the development and deployment environment that is performed completely in the cloud which the required resources are purchased from a cloud service provider through pay-as-you-go basis to enable delivery of everything from simple cloud-based apps to sophisticated, cloud -enabled enterprise applications (Azure.microsoft.com, 2018).



**Figure 6.0.1: Market Share of Cloud Computing Industry (Synergy Research Group, 2017)**

Figure 6.0.1 shows the market share of cloud computing providers in Q4 2016 compared with Q4 2015 (Synergy Research Group, 2017). According to the statistics that is shown in figure 6.0.1, Amazon Web Services (AWS) which is the cloud computing solution delivered by Amazon still successfully maintain its position as dominator of cloud computing market by maintaining its market share of 40% throughout Q4 2015 and Q4 2016. Besides that, Microsoft, Google or IBM is ranked as second place in the cloud computing market by obtaining its market share of 23% in Q4 2016 that is 5% more than Q4 2015 which is 18%. The leadership of Amazon, Microsoft, Google and IBM which are known as large companies in Cloud Computing market has concluded that PaaS is becoming more essential and trendy for various organisations as the services delivered has high level of accessibility with wide variety of features otherwise extensive usage of effort and cost are required to obtain on a locally provisioned server.

Azure Database for MySQL offers fully managed, enterprise-ready community MySQL relational database service for application development and deployment (Microsoft Azure, 2018). Besides that, Azure Database for MySQL is also serves as a PaaS database or database as a service (DBaaS) which is same as Azure SQL Database that is optimized for software as service (SaaS) app development as it is native to the cloud or virtualized platform. Furthermore, it is also has the capability to support most of the MySQL features which makes it does not have any differences with typical MySQL database that is available across premises machines physically, private cloud environments, third party hosted private cloud environments and public cloud. Furthermore, Azure Database for MySQL also offers same set of server products, development tools and expertise with Azure SQL Database.



**Figure 6.0.2: Level of Administration and Degree of Cost Efficiency Quadrant (Microsoft, 2018)**

Figure 6.0.2 shows the diagram that illustrates each offering that is characterized by the level of administration that is obtained over the infrastructure which is located on the X axis and the degree of cost efficiency that is achieved by database level consolidation and automation which is located on the Y axis (Microsoft, 2018). According to the diagram showed in figure 6.0.2, it has concluded that the managed database is capable to achieve the highest degree of cost efficiency and with lower administration level is required which can help to minimise the operation cost. Hence, this can enable the detail work on database to be handed over to the expertise which makes it as a great service to move on as it allows the organisation to focus on performing business activities that could promote business growth which aids to increase profitability of the organisation. Besides that, the managed database also serves as an optimise solution that helps to simplify the process of provisioning and managing the database by allowing the direct creation of database on the service using the built-in features or functionality which can help to save cost by eliminating the unnecessary resources and manpower in provisioning and managing the database. Furthermore, the managed database offered by Microsoft Azure virtualized platform is also beneficial because it performs the administration tasks automatically on the database which can help to prevent unnecessary workloads and efforts that could eventually reduce time spent in administrating the managed database. By using database offered by Microsoft Azure, this could improve productivity in database management that could eventually promote high level of effectiveness and efficiency in managing the information that flow throughout the organisation.

There are various features and functionalities that are offered by Azure Database for MySQL which is beneficial or helpful for managing the information of the organisation that can increase effectiveness and efficiency in database management. First and foremost, Azure Database for MySQL offers fully managed service provider which is combined with MySQL Community Edition (Microsoft Azure, 2018). This can help to avoid complexity of the infrastructure which eventually helps to simplify the process of managing the database and enable putting the main focus on building exceptional web-based applications. Besides that, Azure Database for MySQL offers several service tiers with different pricing which consist of Basic, General Purpose and Memory Optimized where each tier has different capability in supporting lightweight to heavyweight database workloads (Microsoft, 2018). Apart from that, flexibly scaling which provision in minutes, compute or storage independently within seconds is also offered by Azure Database for MySQL (Microsoft Azure, 2018). This can simplify the effort required by the organisation to maintain performance of the database as well as has high capability in satisfying the needs of the solution by adjusting the scale. This allows the organisation to have high capability in responding towards rapid changing resources requirements. Furthermore, Azure Database for MySQL also keeps the applications of the organisations up and running without the need of extra configuration, replication or cost to ensure that the applications are operational over time (Microsoft Azure, 2018). This can enable the organisation to have the ability of delivering the applications that are accessible by the users at any time which meets the quality goal of availability. Other than that, this also can enable the organisation to perform business operation using the applications smoothly which can eventually enable business growth. Apart from that, Azure Database for MySQL also offers airtight protection for the data of the applications with features that consists of limit access, protect data at-rest and in-motion as well as monitor activity throughout the database (Microsoft, 2018). Besides that, Azure Database for MySQL also offers automated backup feature for the data that stored into the database. This can aid the organisation to prevent unauthorised access to the data of the organisation as well as leakage of data especially for those that are critical to support business growth of the organisation. Additionally, this also can facilitate the process of restoring the data of the organisation that is accidentally deleted or loss which can prevent extensive usage of cost and effort in recovering the loss data as well as maintain the ability of the organisation in performing business operations as usual despite the occurrence of disaster. With the features and functionalities that offered by Azure Database for MySQL that provides various advantages, this has demonstrated the convincing evidence that motivates the business to consider of selecting Azure Database for MySQL for its applications developed.

# **7.0 Conclusion**

To sum it up, the development of Maersk Line’s CMS is able to be completed on time as well as successfully deployed on Cloud environment which is Microsoft Azure. The development life cycle of Maersk Line’s CMS is initiated by determining requirements specification which consist of the features and functionalities that are needed to be demonstrated in the implemented Maersk Line’s CMS after analysing the business case regarding the organisation which is Maersk Line. After that, design of Maersk Line’s CMS is performed through identification of design considerations which resulted in creation of diagrams that consist of Use Case Diagram, Activity Diagram, Entity Relationship Diagram and Cloud Architectural Diagram. Next, the configuration of implemented Maersk Line’s CMS in Microsoft Azure virtualized platform involves purchasing Platform as a Service (PaaS) by creating one Azure Database for MySQL and publishing the application to Azure App Service in one region only. Besides that, testing is conducted on Maersk Line’s CMS to ensure that it is behaving as expected.

During the process of developing Maersk Line’s CMS, the developer is able to gain a good understanding on the concept of Cloud Computing in its various forms as well as knowing how to apply the usage of Cloud Computing service towards the usage of Maersk Line’s CMS through Microsoft Azure. In addition, the developer also can familiarise the usage of Cloud development environment of Microsoft Azure through exploration of the features and functionalities offered by Microsoft Azure as well as making use of them which can help to improve the effectiveness and efficiency in managing the web-based application. The realization of concept and practical tasks performed on designing, implementing and deploying Maersk Line’s CMS to Microsoft Azure virtualized platform can help to enhance the cloud application development skill which can transform the developer who had zero knowledge about practical usage of Cloud into the developer who is able to slowly master and adapt into Cloud development environment. This can provide strong foundation for the developer when proceeding to career path of cloud application development which has high level of popularity nowadays. Apart from that, the developer also leant the importance of having comprehensive planning before starting to develop Maersk Line’s CMS such as developing UML diagrams and designing Cloud architecture of the web-based application when it is deployed in Microsoft Azure for ensuring high level of balance or relevance by considering required resources with the allocated budget to present the best solution which can aid the developer to gain clearer and deeper understanding on the scope of Maersk Line’s CMS implemented and deployed in Microsoft Azure. Apart from that, the developer also has learnt the importance of time management which can help the developer to complete the implementation of Maersk Line’s CMS without any bugs occurred as well complete deployment of the web-based application into Microsoft Azure and ensure smooth execution of Maersk Line’s CMS on the platform successfully and on time.

All in all, Cloud computing is classified as the new trend in this era of globalisation which is widely used by various organisations especially the large corporations due to its speed, agility and global reach. In addition, Platform as a Service (PaaS) which is known as common concept of Cloud computing adopts pay-as-you-use model which can provide flexibility in terms of resource usage and eliminates the high level of complexity on the infrastructure which could simplify the process of managing the applications that are migrated into Cloud platform. In a nutshell, Maersk Line’s CMS is completed with the best effort as well as effectively and efficiently managed as best as possible in Microsoft Azure due to time as well as budget constraint and it is believed that there are few aspects to be enhanced in the future which can help to make Maersk Line’s CMS even robust.

# **8.0 References**

Azure.microsoft.com. (2018). What is PaaS? Platform as a Service | Microsoft Azure. [online] Available at: https://azure.microsoft.com/en-us/overview/what-is-paas/ [Accessed 11 Apr. 2018].

Bassil, Y. (2012). A Comparative Study on the Performance of the Top DBMS Systems. *Journal of Computer Science & Research (JCSCR)*, 1(1), pp.20-31.

Cavale, A., Boucher, R. J. & Kemnetz, J., 2017. Scale instance count manually or automatically. [Online] Available at: https://docs.microsoft.com/en-us/azure/monitoring-and-diagnostics/insights-how-to-scale [Accessed 10 Apr. 2018].

Converse, T., Park, J. and Morgan, C. (2007). PHP5 and MySQL Bible. Enskede: TPB.

Cullen, K. (2002). PHP: An open source solution for Web programming and dynamic content. *Information Technology and Libraries*, 21(3), pp.116-118.

Darmawikarta, D. (2014). SQL For MySQL: A Beginner's Tutorial. Canada: Brainy Software.

Dev.mysql.com. (2018). MySQL :: MySQL Workbench Manual :: 1 General Information. [online] Available at: https://dev.mysql.com/doc/workbench/en/wb-intro.html [Accessed 10 Apr. 2018].

Lin, C., 2016. Scale up an app in Azure. [Online] Available at: https://docs.microsoft.com/en-us/azure/app-service/web-sites-scale [Accessed 10 Apr. 2018].

Luijbregts, B., 2017. How to Autoscale Azure App Services & Cloud Services. [Online] Available at: https://stackify.com/autoscale-azure-app-services-cloud-services/ [Accessed 10 Apr. 2018].

Microsoft Azure. (2017). Autoscaling. [online] Available at: https://docs.microsoft.com/en-us/azure/architecture/best-practices/auto-scaling [Accessed 10 Apr. 2018].

Microsoft Azure. (2018). App Service Plans | Microsoft Azure. [online] Available at: https://azure.microsoft.com/en-us/pricing/details/app-service/plans/ [Accessed 10 Apr. 2018].

Microsoft Azure. (2018). Azure Database for MySQL – Fully-Managed Service | Microsoft Azure. [online] Available at: https://azure.microsoft.com/en-us/services/mysql/ [Accessed 11 Apr. 2018].

Microsoft, 2018. *Azure Database for MySQL.* [Online] Available at: https://azure.microsoft.com/en-us/services/mysql/ [Accessed 10 April 2018].

Microsoft. (2018). Overview of Azure Database for MySQL relational database service. [online] Available at: https://docs.microsoft.com/en-us/azure/mysql/overview [Accessed 11 Apr. 2018].

Microsoft. (2018). SQL (PaaS) Database vs. SQL Server in the cloud on VMs (IaaS). [online] Available at: https://docs.microsoft.com/en-us/azure/sql-database/sql-database-paas-vs-sql-server-iaas [Accessed 11 Apr. 2018].

Synergy Research Group (2017). Microsoft, Google and IBM Public Cloud Surge is at Expense of Smaller Providers | Synergy Research Group. [online] Srgresearch.com. Available at: https://www.srgresearch.com/articles/microsoft-google-and-ibm-charge-public-cloud-expense-smaller-providers [Accessed 11 Apr. 2018].