

Acknowledgement

First of all, I would like to express my deepest gratitude to my lecturer, Dr.Kalai Anand A/L Ratnam, who has been giving me lots of knowledge and idea about the Microsoft Azure Cloud Computing. Besides, he was also very helpful and provides professional advice on how to deploy a web application to Azure App Service. Not only that, he provides useful materials and tools that help me to save a lot of time on deploying the web application and complete the assignment on time.

I also want to take this opportunity thank to my course mates, who teach and guide me all the time in this assignment. They are willing to give help and teach me about the PHP programming language and some features of NetBeans IDE 8.2 that I have never used before. At the same time, I appreciate that my course mates who are willing to share their knowledge and help me to complete this assignment.

In conclusion, I have gained more knowledge about the Azure Cloud Computing which is very useful, because it could help me to apply it in my future works.

Table of Contents

Acknowledgement	1
1.0 Introduction.....	3
1.1 Project Background.....	3
1.2 Project Objective	4
1.3 Project Scope.....	4
1.5 Project Deliverables	5
2.0 Project Plan	6
3.0 Design	7
3.1 Cloud Architectural Diagrams	7
3.2 Design Consideration	7
3.3 Modelling	8
3.3.1 Use Case Diagram	8
3.3.2 Use Case Specification	9
3.4 Sequence Diagram.....	19
4.0 Implementation	30
4.1 Publish Web Application	30
4.2 Application Scaling	38
4.3 Managed Databases (PAAS).....	39
5 Testing.....	41
5.1 Performance Testing	41
5.2 Unit Testing.....	44
6.0 Conclusion	46
7.0 References.....	47

1.0 Introduction

1.1 Project Background

Maersk Line is the global container division and the largest operating unit of the A.P. Moller – Maersk Group, 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 an 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 management.

1.2 Project Objective

The aims of the project are to develop a Web application and deploy it on the Microsoft Azure Cloud Service that allows administrators to manage Maersk Line shipping through the website. In this system, administrators can book for companies that want to ship packages through containers. In addition, it also allows administrators to register items, ships, agent and customers. Not only that, the administrator can also view the reservations and other information already registered in the system.

1.3 Project Scope

The scope of the project is included the design and development of Web applications based on Maersk Line requirements. The system needs to use Azure and SQL database to store reservations and other information. The web application provides a login page for administrator to login, this is to ensure the security of the site and prevent anonymous information theft. After the website development is completed, it will be deployed on the Microsoft Azure cloud service. The traffic manager will be implemented to control requests from web clients. Finally, if there is any possibility of a bug, the performance test will be completed and analysed. This is to ensure that the system performs smoothly when deployed.

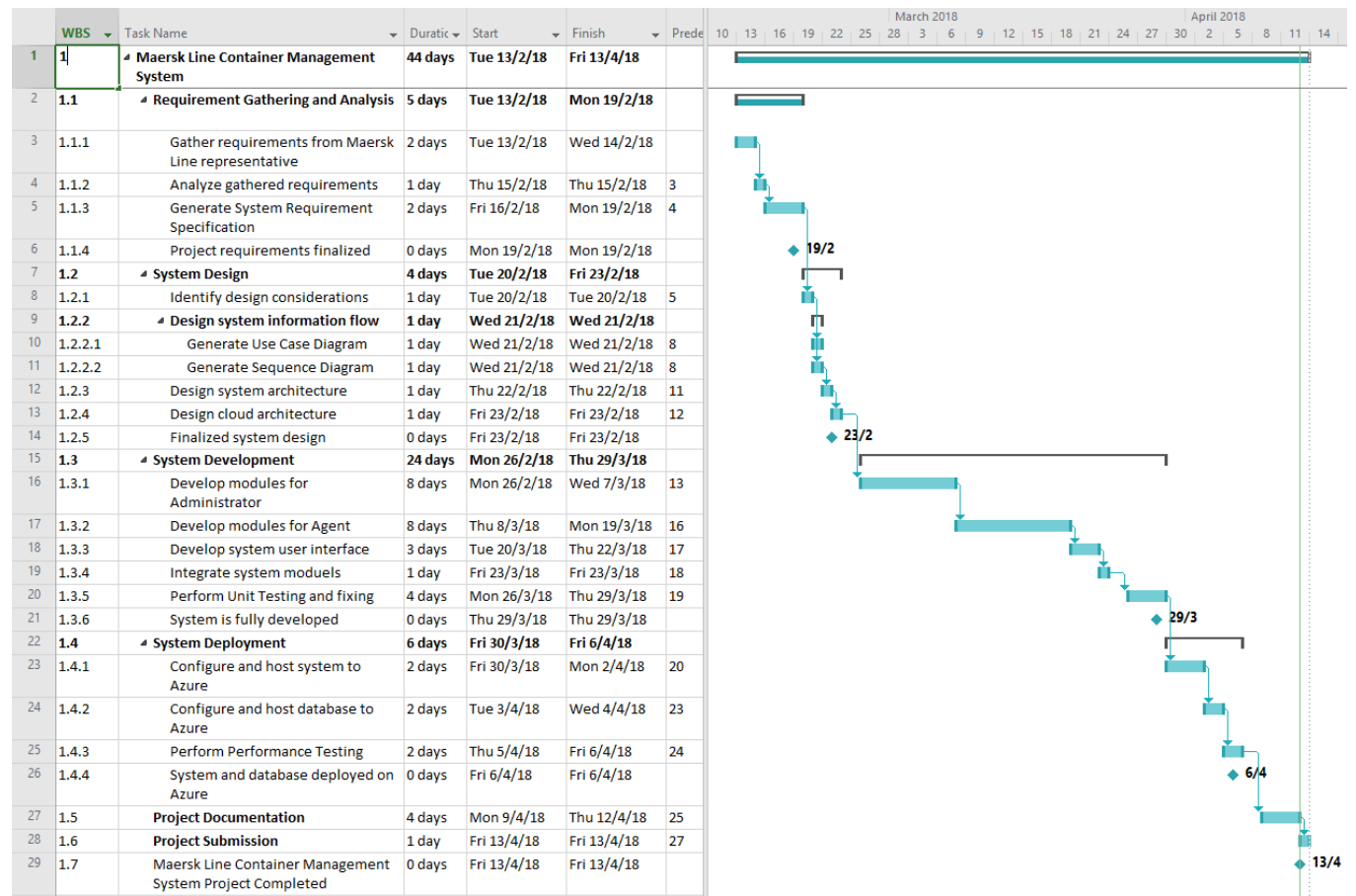
1.5 Project Deliverables

The Maerks Line web application is available and compatible with different types of browser, such as Google Chrome, Mozilla Firefox, Microsoft Edge, and others. Users will be able to perform all the functionalities that listed below:

- User able to view the details of shipment information, such as departure time, arrival time, departure place, and arrival place in the web application.
- User able to view their reservation information in the web application.
- User able to register customers and items in the web application.
- User able to make reservation for the shipment in the web application.
- Admin User able to perform update and delete the reservation, ship, container and account in this web application.

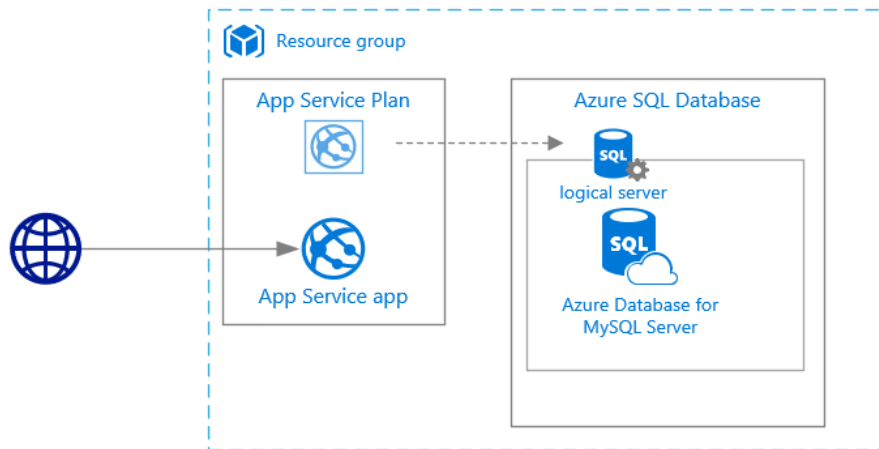
2.0 Project Plan

Gantt Chart



3.0 Design

3.1 Cloud Architectural Diagrams



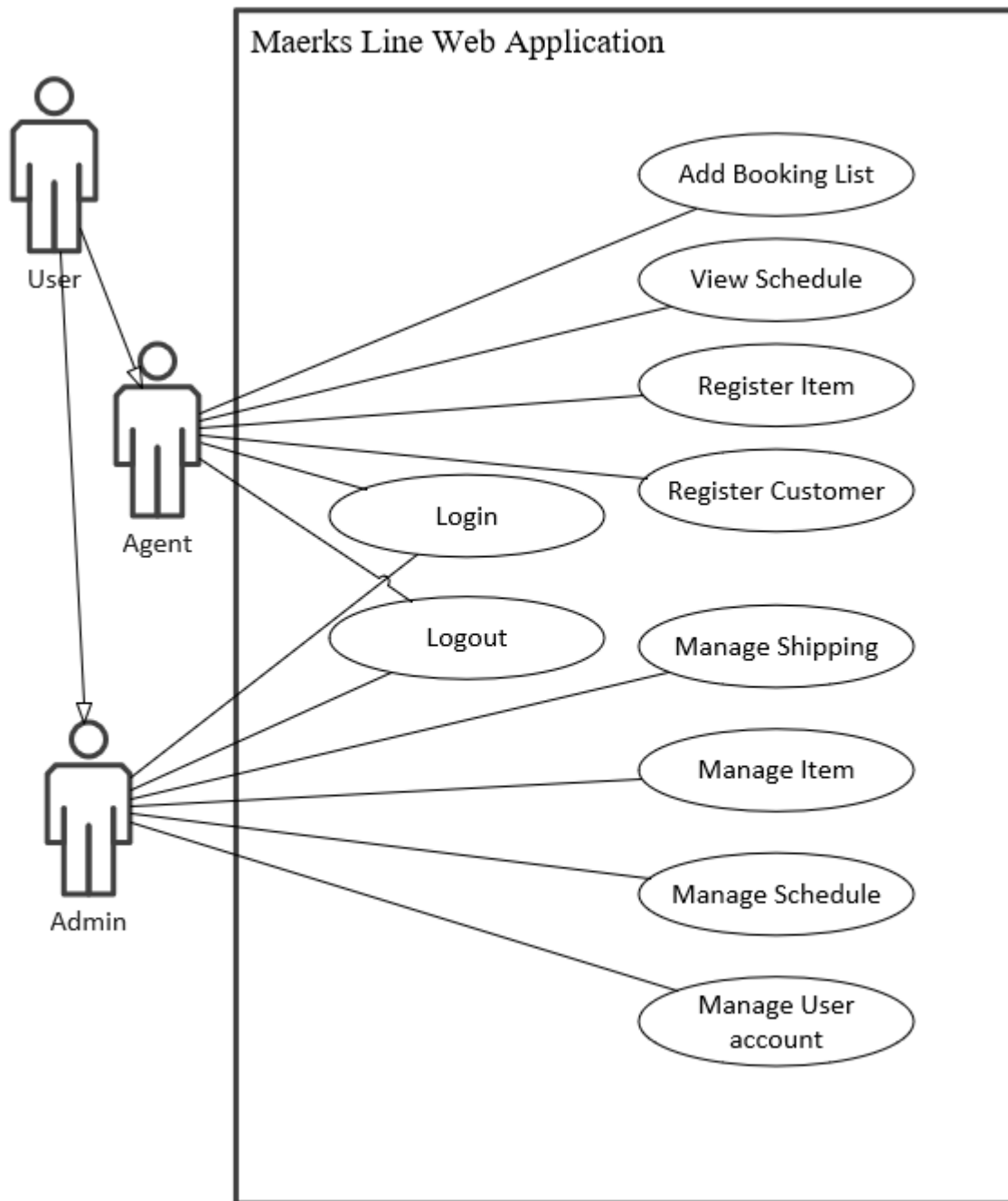
3.2 Design Consideration

Microsoft Azure Estimate				
Your Estimate				
Service type	Custom name	Region	Description	Estimated Cost
App Service		Southeast Asia	1 instance(s) x 1 Months, Size: S1, Standard tier, 0 SNI connection(s), 0 IP connection(s)	RM306.60
Azure Database for MySQL		Southeast Asia	Basic Tier, 1 Gen 4 (1 vCore) x 1 Months, 5 GB Storage, 100 GB Additional Backup storage - LRS redundancy	RM212.14
Support			Support	RM0.00
			Monthly Total	RM518.74
			Annual Total	RM6,224.90
Disclaimer				
All prices shown are in Malaysian Ringgit (RM\$). This is a summary estimate, not a quote. For up to date pricing information please visit this link . This estimate was created at 4/12/2018 3:49:50 PM UTC.				

Some assumptions or considerations have been made throughout the project, which is Azure will costs around RM518.74 permonth to provided to the system developer for initial development purposes, so the validity of the system concept has been verified.

3.3 Modelling

3.3.1 Use Case Diagram



3.3.2 Use Case Specification

Table 1: Login

Use Case ID:	1
Use Case Name:	Login
Summary:	Users able to login into this web application
Dependency:	N/A
Actors:	User
Precondition:	User requires username and password to login to the web application.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Login page is displayed. 2. The system prompts user to fill in username and password. 3. The user inserts username and password 4. System verifies the login information that the user entered. 5. User logged in to the web application successfully.
Description of Alternative Sequence:	<ol style="list-style-type: none"> 4 a) If login details are verified, then the users will be login into the web application. 4 b) If login details are invalid, then the users need to re-enter the user name password.
Post-condition	The user is logged in to the web application with correct account.

Table 2: Add Booking

Use Case ID:	2
Use Case Name:	Add Booking
Summary:	User creates a new booking for the company in the web application.
Dependency:	N/A
Actors:	Agent
Precondition:	User login into the web application as agent account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Booking” on side menu. 3. User click the “New” button. 4. The user inserts booking details. 5. User click the “Insert” button. 6. Booking is created and saved into the database.
Description of Alternative Sequence:	5 a) If user click the “Close” button, the Booking will not create and save into the database.
Post-condition	Booking information are stored into the database system.

Table 3: View Schedule

Use Case ID:	3
Use Case Name:	View Schedule
Summary:	User view schedule of shipping on the web application.
Dependency:	N/A
Actors:	Agent
Precondition:	User login into the web application as agent account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Schedule” on side menu. 3. The system display schedule of shipping in a table form. 4. User able to view the details of shipping in a table form.
Description of Alternative Sequence:	4 a) User able to search the key word to find the specific information.
Post-condition	N/A

Table 4: Register Item

Use Case ID:	4
Use Case Name:	Register Item
Summary:	User registers item on the web application.
Dependency:	N/A
Actors:	Agent
Precondition:	User login into the web application as agent account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Register Item” on side menu. 3. User click the “New” button. 4. The user inserts Item details. 5. User click the “Insert” button. 6. Item data is created and saved into the database.
Description of Alternative Sequence:	5 a) If user click the “Close” button, the Item data will not create and save into the database.
Post-condition	The Item information are stored into the database system.

Table 5: Register Customer

Use Case ID:	5
Use Case Name:	Register Customer
Summary:	User registers Customer on the web application.
Dependency:	N/A
Actors:	Agent
Precondition:	User login into the web application as agent account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Register Customer” on side menu. 3. User click the “New” button. 4. The user inserts Customer details. 5. User click the “Insert” button. 6. Customer data is created and saved into the database.
Description of Alternative Sequence:	5 a) If user click the “Close” button, the Customer data will not create and save into the database.
Post-condition	The Customer information are stored into the database system.

Table 6: Manage Shipping

Use Case ID:	6
Use Case Name:	Manage Shipping
Summary:	User able to Manage the Shipping details include Insert, Delete and Update.
Dependency:	N/A
Actors:	Admin
Precondition:	User login into the web application as admin account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Ship” on side menu. 3. User able to click the “New” button, inserts Ship details, and click “Insert” button to insert data. 4. Ship data will have created and saved into the database. 5. User able to click the “Edit” label, change the data that system display, and click the “Update” button to update data. 6. The new Ship data will have saved into the database. 7. User able to click “Delete” button, click the “OK” button on confirmation dialog that system prompt. 8. The specific Ship data has been deleted in database.
Description of Alternative Sequence:	<p>3 a) If user click the “Close” button, the Ship data will not create and save into the database.</p> <p>5 a) If user does not click the “Update” button, the Ship data will not have updated and save into the database.</p> <p>7 a) If user click the “Cancel” button, the Ship data will not have deleted in the database.</p>
Post-condition	The Ship information are change in the database system depend on user selected.

Table 7: Manage Item

Use Case ID:	7
Use Case Name:	Manage Item
Summary:	User able to Manage the Item details include Insert, Delete and Update.
Dependency:	N/A
Actors:	Admin
Precondition:	User login into the web application as admin account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Store” on side menu. 3. User able to click the “New” button, inserts Item details, and click “Insert” button to insert data. 4. Item data will have created and saved into the database. 5. User able to click the “Edit” label, change the data that system display, and click the “Update” button to update data. 6. The new Item data will have saved into the database. 7. User able to click “Delete” button, click the “OK” button on confirmation dialog that system prompt. 8. The specific Item data has been deleted in database.
Description of Alternative Sequence:	<p>3 a) If user click the “Close” button, the Item data will not create and save into the database.</p> <p>5 a) If user does not click the “Update” button, the Item data will not have updated and save into the database.</p> <p>7 a) If user click the “Cancel” button, the Item data will not have deleted in the database.</p>
Post-condition	The Item information are change in the database system depend on user selected.

Table 8: Manage Schedule

Use Case ID:	8
Use Case Name:	Manage Schedule
Summary:	User able to Manage the Schedule details include Insert, Delete and Update.
Dependency:	N/A
Actors:	Admin
Precondition:	User login into the web application as admin account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Schedule” on side menu. 3. User able to click the “New” button, inserts Schedule details, and click “Insert” button to insert data. 4. Schedule data will have created and saved into the database. 5. User able to click the “Edit” label, change the data that system display, and click the “Update” button to update data. 6. The new Schedule data will have saved into the database. 7. User able to click “Delete” button, click the “OK” button on confirmation dialog that system prompt. 8. The specific Schedule data has been deleted in database.
Description of Alternative Sequence:	<p>3 a) If user click the “Close” button, the Schedule data will not create and save into the database.</p> <p>5 a) If user does not click the “Update” button, the Schedule data will not have updated and save into the database.</p> <p>7 a) If user click the “Cancel” button, the Schedule data will not have deleted in the database.</p>
Post-condition	The Schedule information are change in the database system depend on user selected.

Table 9: Manage User Account

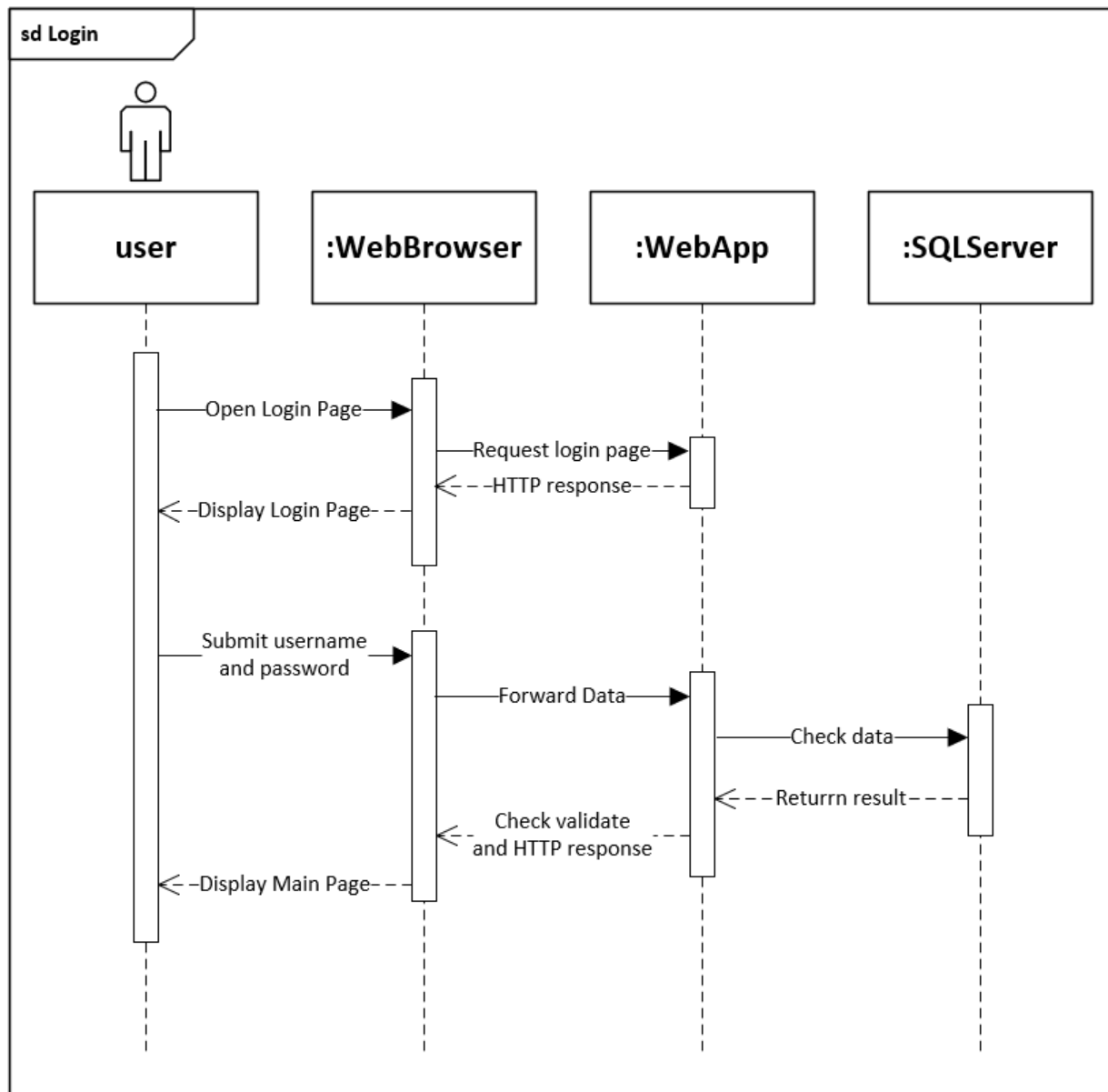
Use Case ID:	9
Use Case Name:	Manage User Account
Summary:	User able to Manage the Account details include Insert, Delete and Update.
Dependency:	N/A
Actors:	Admin
Precondition:	User login into the web application as admin account.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User selects “Users” on side menu. 3. User able to click the “New” button, inserts Ship details, and click “Insert” button to insert data. 4. Account data will have created and saved into the database. 5. User able to click the “Edit” label, change the data that system display, and click the “Update” button to update data. 6. The new Account data will have saved into the database. 7. User able to click “Delete” button, click the “OK” button on confirmation dialog that system prompt. 8. The specific Account data has been deleted in database.
Description of Alternative Sequence:	<p>4 a) If user click the “Close” button, the Account data will not create and save into the database.</p> <p>8 a) If user does not click the “Update” button, the Account data will not have updated and save into the database.</p> <p>12 a) If user click the “Cancel” button, the Account data will not have deleted in the database.</p>
Post-condition	The Account information are change in the database system depend on user selected.

Table 10: Logout

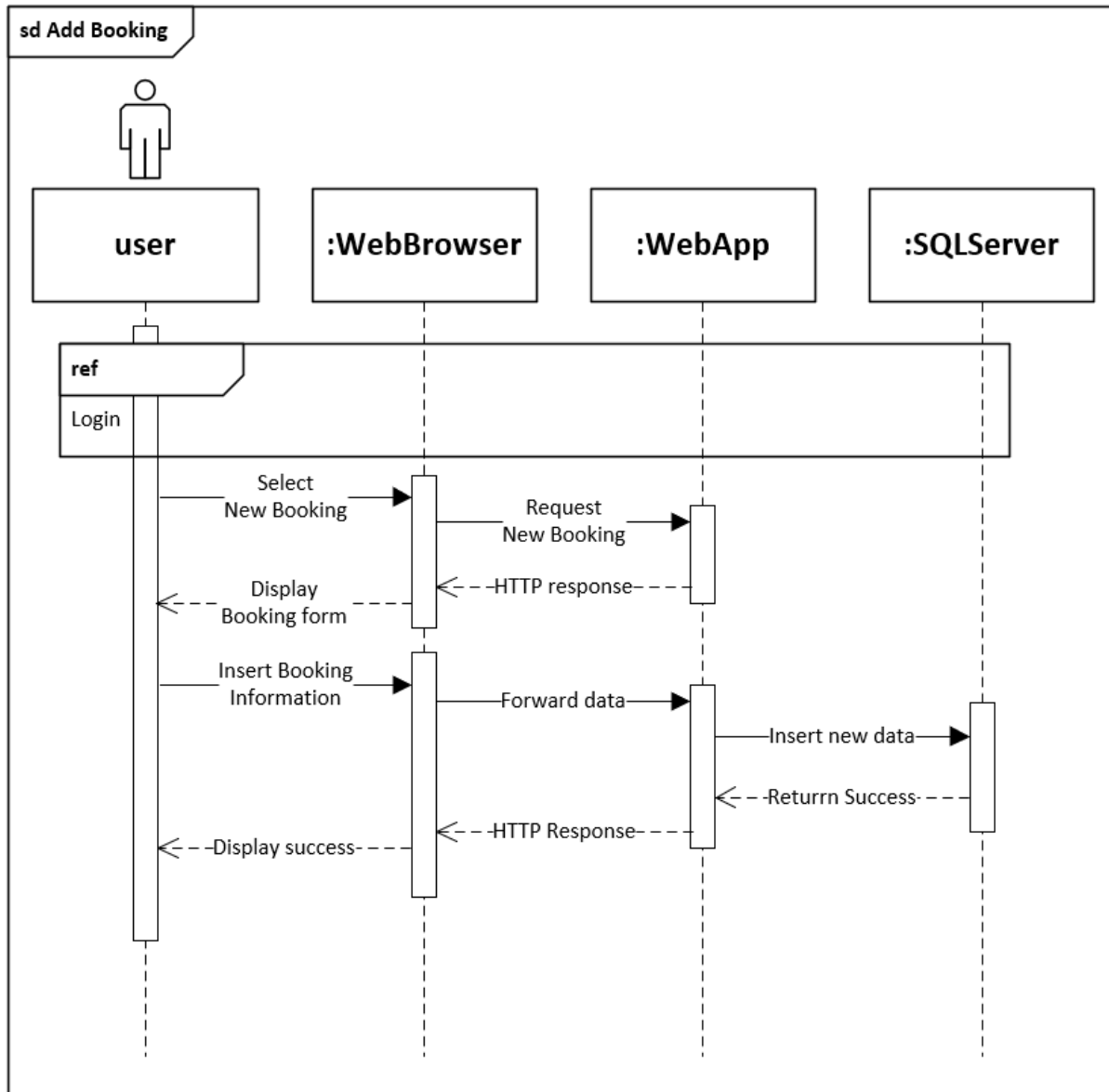
Use Case ID:	10
Use Case Name:	Logout
Summary:	User able to logout from the web application
Dependency:	N/A
Actors:	User
Precondition:	User is login the web application.
Description of the Main Sequence:	<ol style="list-style-type: none"> 1. Home Page is displayed. 2. User select the “Logout” option. 3. System display the login page. 4. User logout the system successfully.
Description of Alternative Sequence:	<ol style="list-style-type: none"> 1. If user does not login into the web application, then the user unable performs logout.
Post-condition	User able to logout from the web application.

3.4 Sequence Diagram

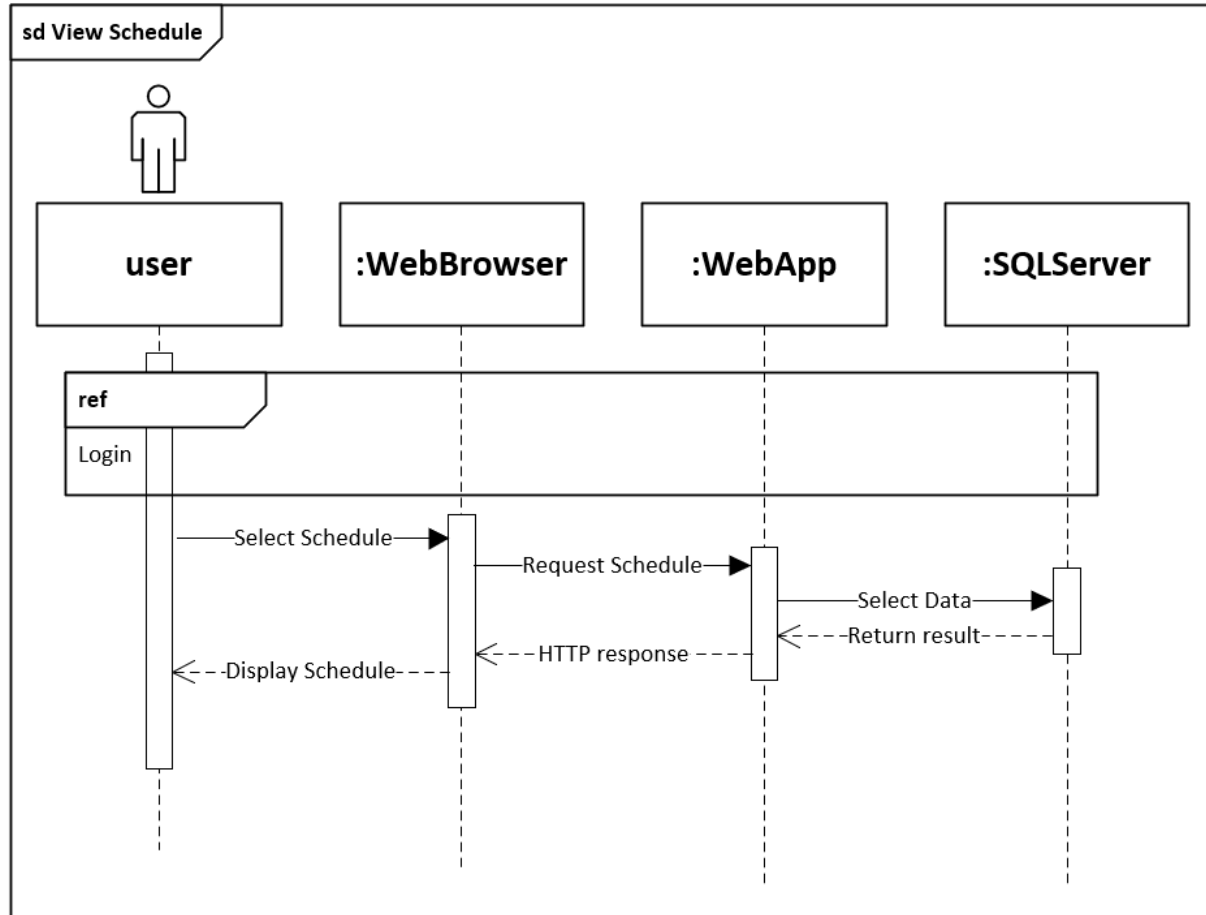
3.4.1 Login



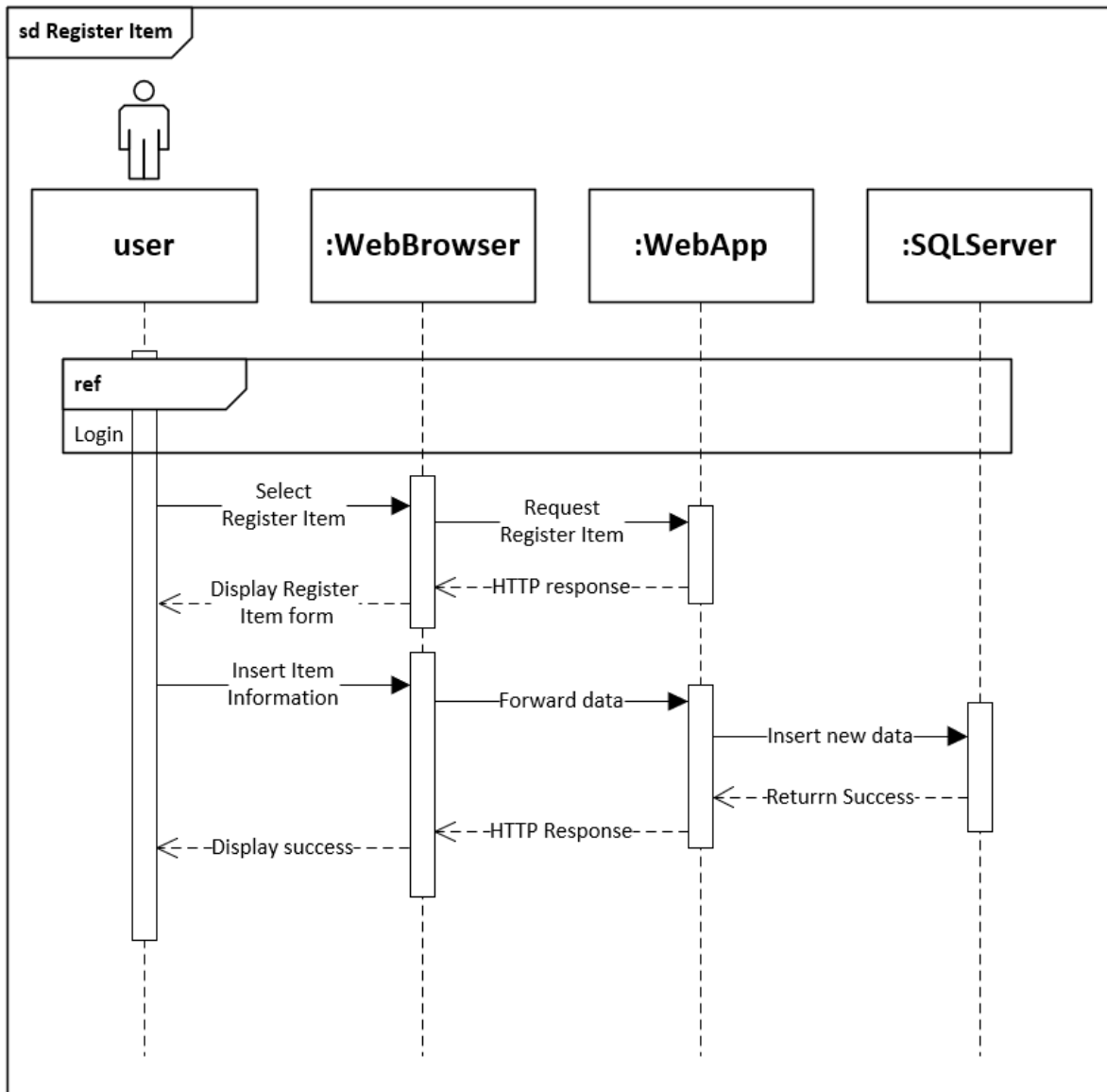
3.4.2 Add Booking



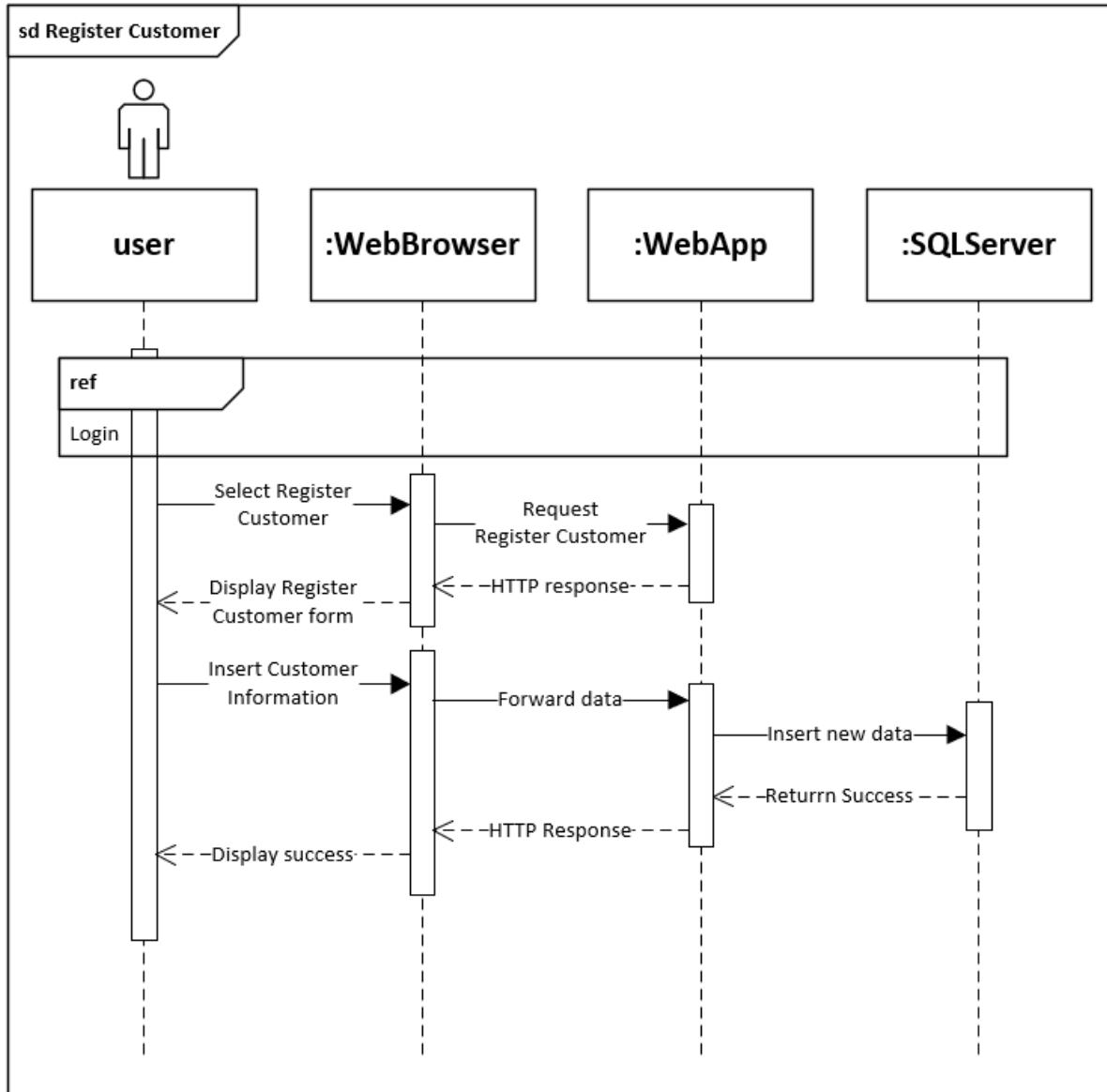
3.4.3 View Schedule



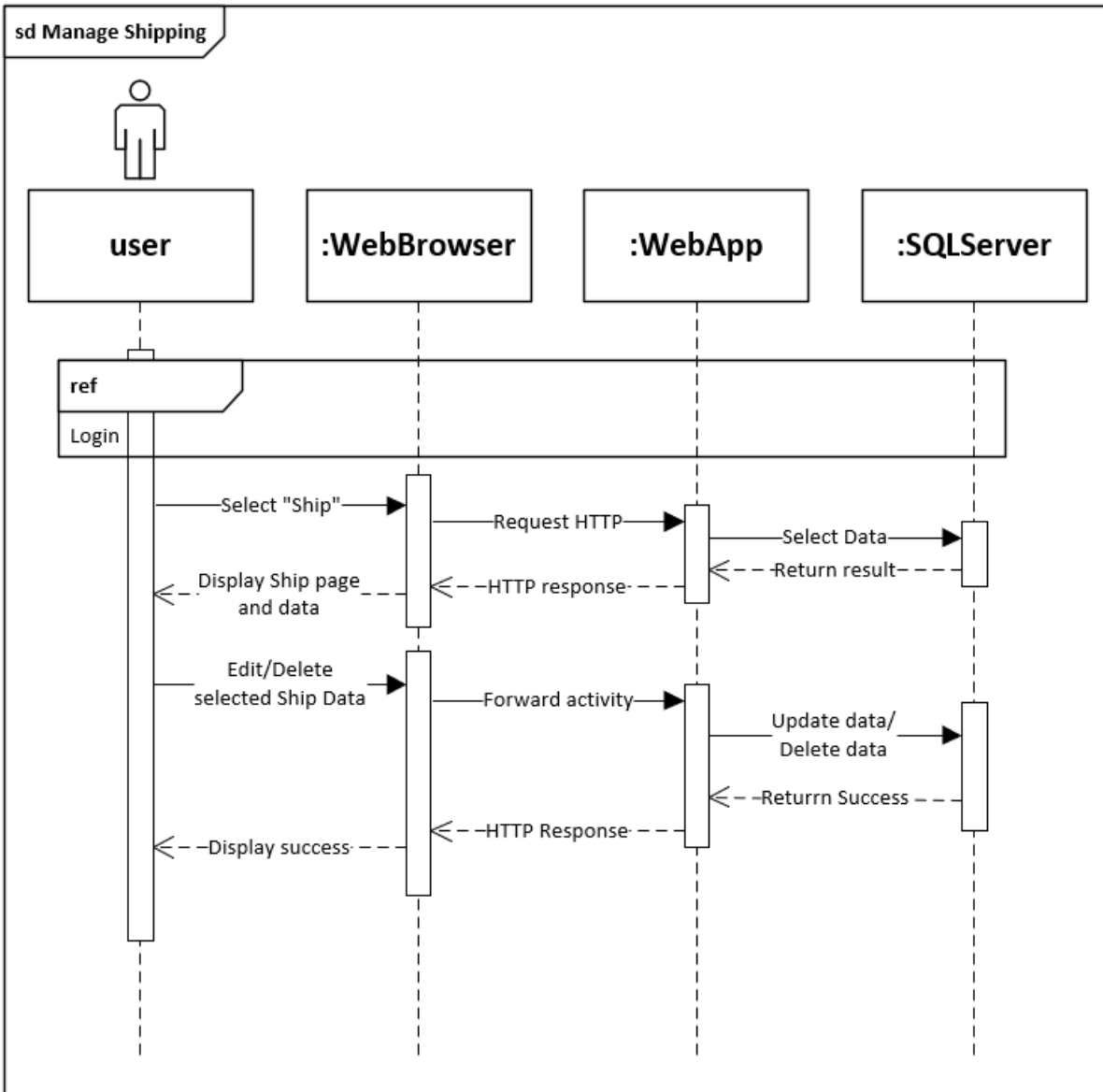
3.4.4 Register Item



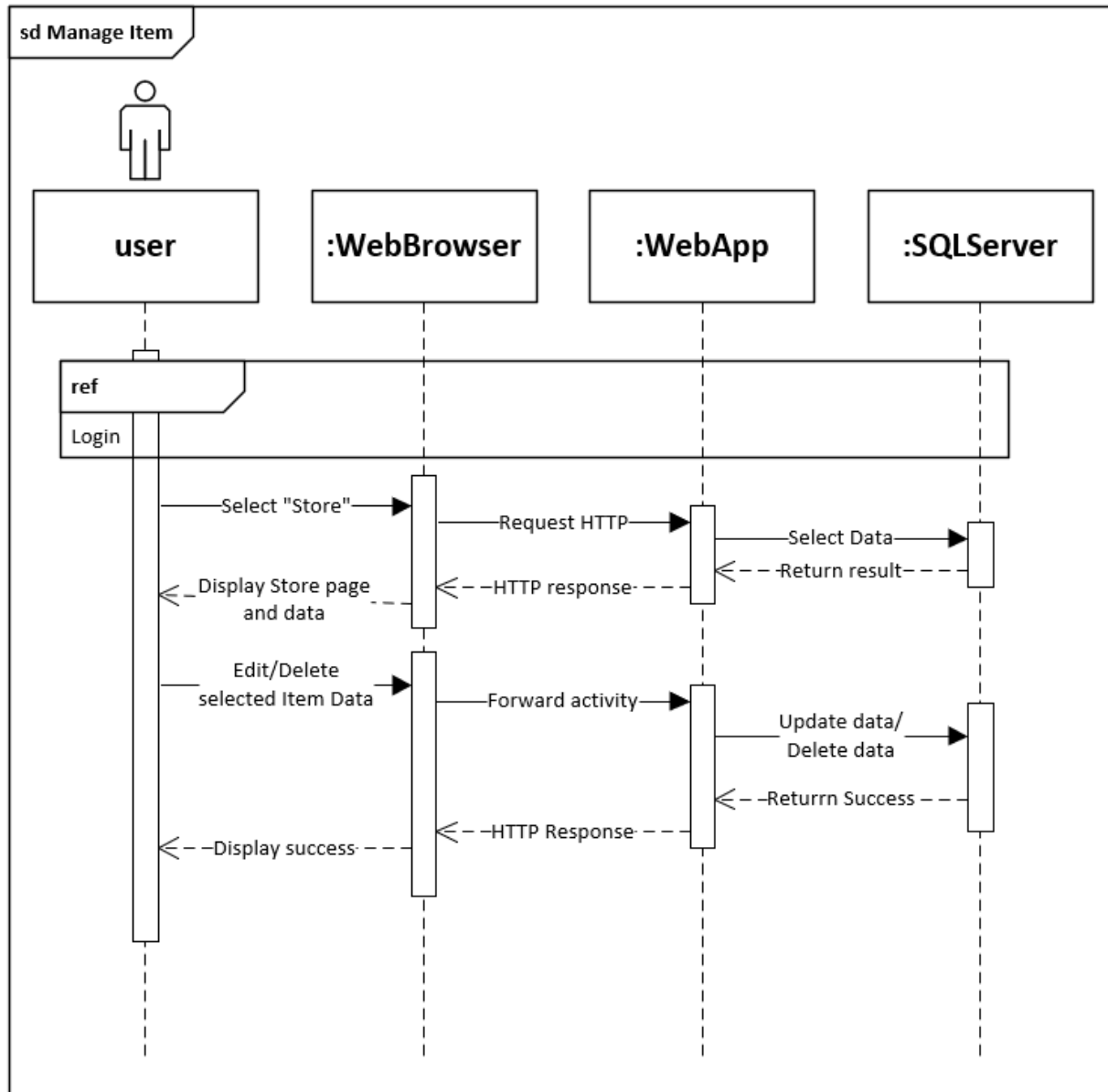
3.4.5 Register Customer



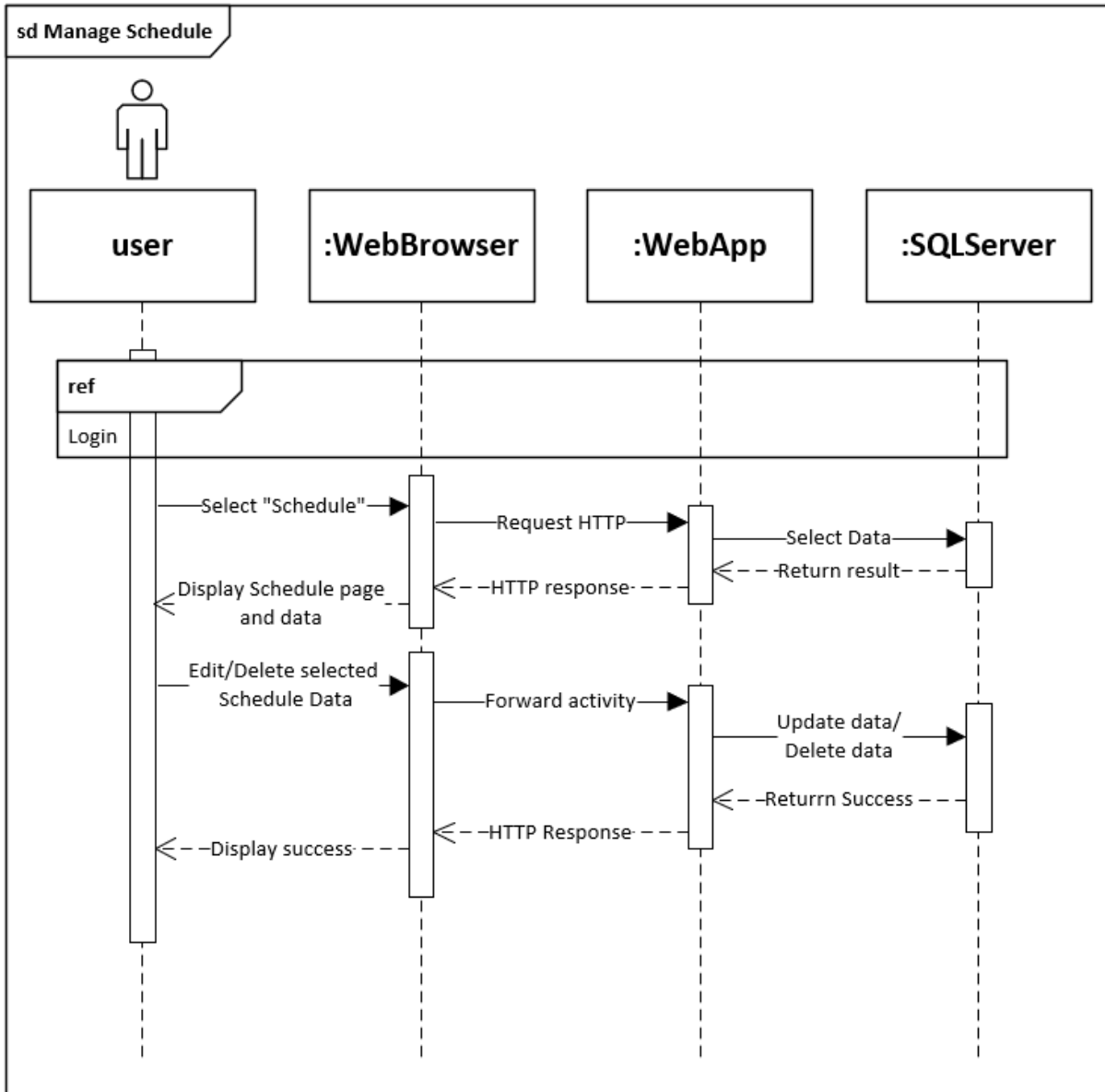
3.4.6 Manage Shipping



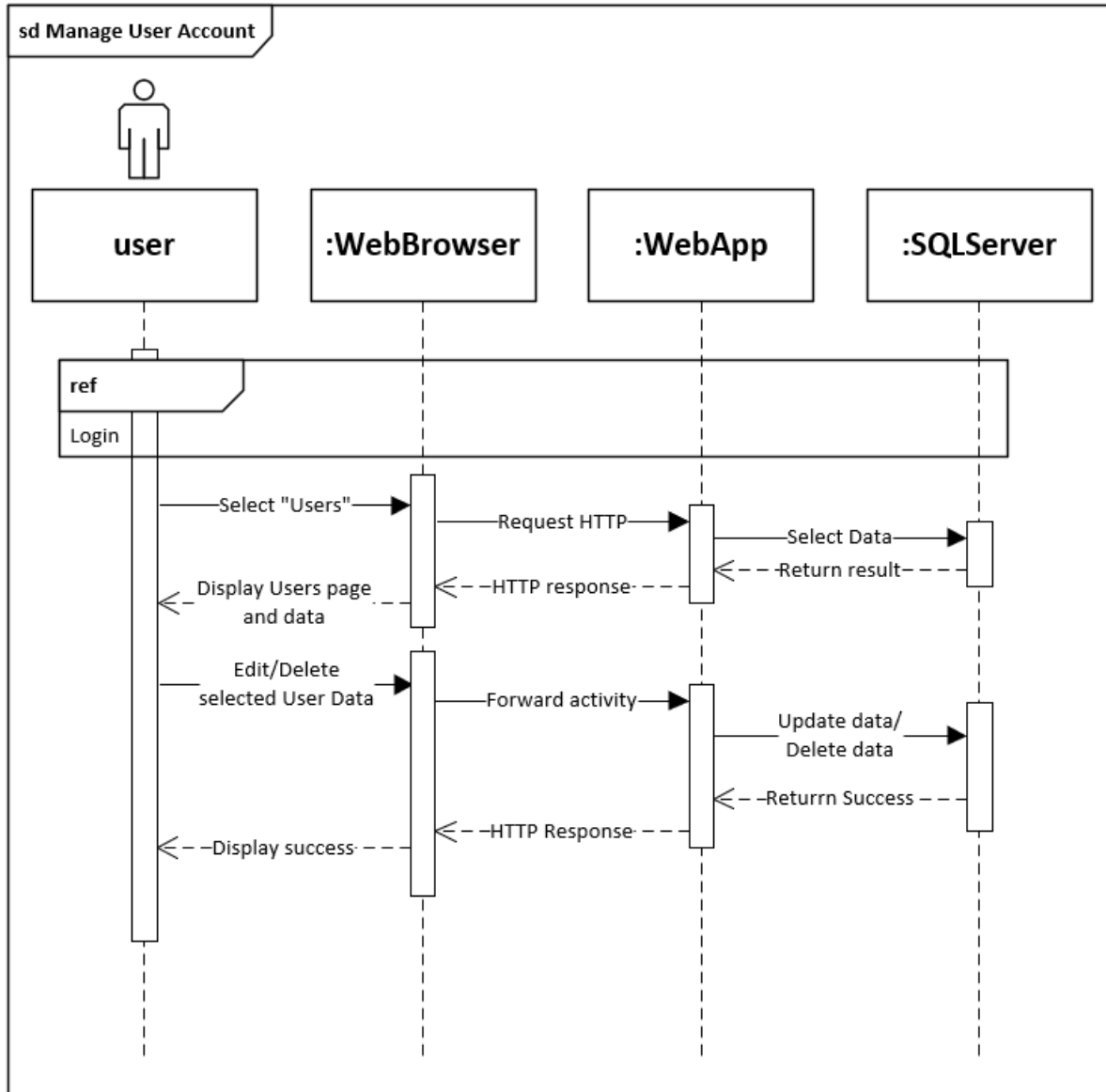
3.4.7 Manage Item



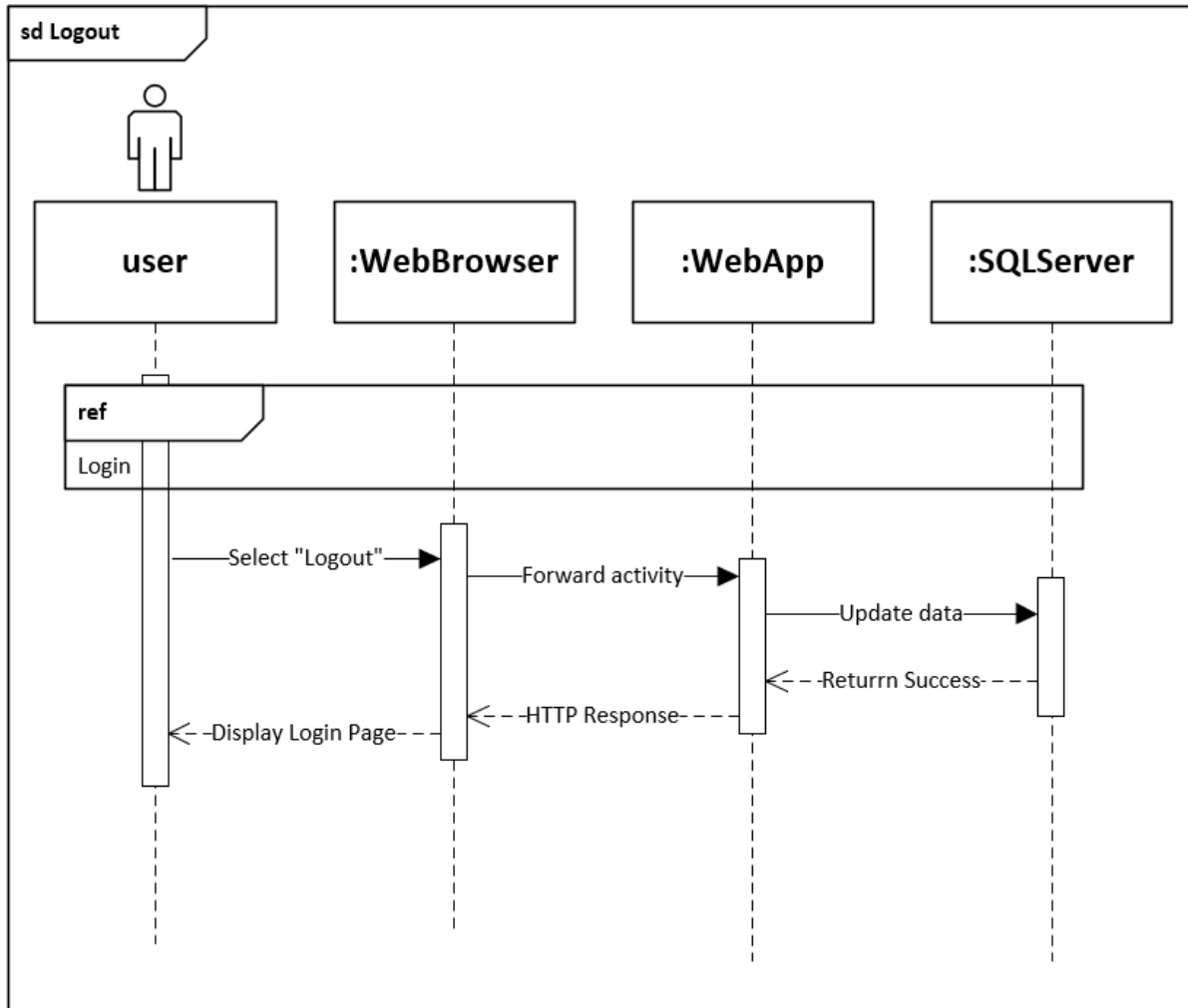
3.4.8 Manage Schedule



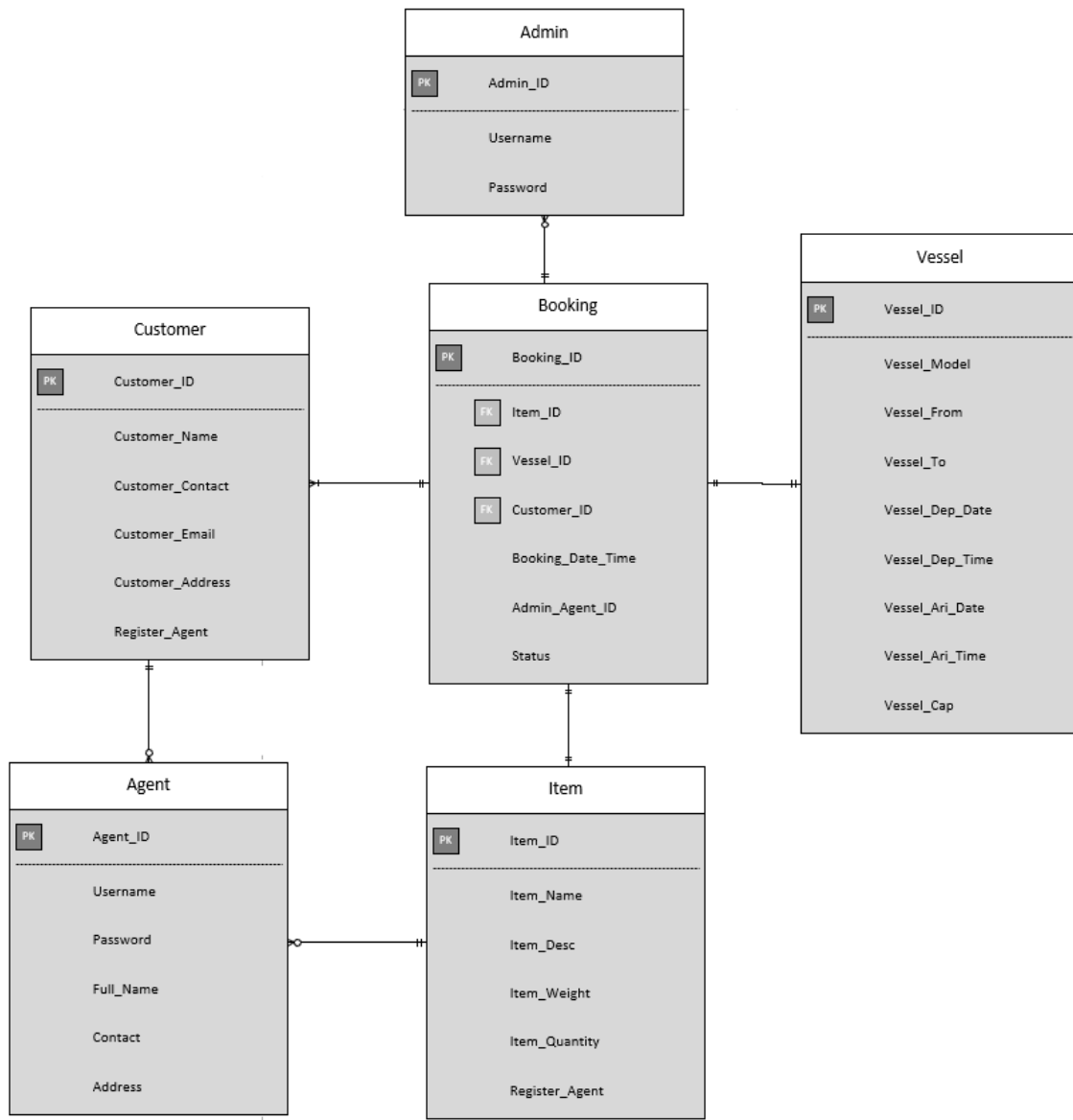
3.4.9 Manage User Account



3.4.10 Logout



3.5 Entity Relationship Diagram (ERD)



4.0 Implementation

4.1 Publish Web Application

Publish Source Code on GitHub

The screenshot shows the GitHub repository page for 'hygron/ddac'. The repository has 0 stars and 0 forks. The 'Code' tab is selected, showing the 'Quick setup' section. A red box highlights the commands to create a new repository on the command line:

```
echo "# ddac" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:hygron/ddac.git
git push -u origin master
```

Below this, the commands to push an existing repository from the command line are shown:

```
git remote add origin git@github.com:hygron/ddac.git
git push -u origin master
```

On the right, a terminal window titled 'MINGW64: c:/xampp/htdocs/ddac' shows the output of the commands:

```
create mode 100644 user_guide/libraries/zip.html
create mode 100644 user_guide/license.html
create mode 100644 user_guide/objects.inv
create mode 100644 user_guide/overview/appflow.html
create mode 100644 user_guide/overview/at_a_glance.html
create mode 100644 user_guide/overview/features.html
create mode 100644 user_guide/overview/getting_started.html
create mode 100644 user_guide/overview/goals.html
create mode 100644 user_guide/overview/index.html
create mode 100644 user_guide/overview/mvc.html
create mode 100644 user_guide/search.html
create mode 100644 user_guide/searchindex.js
create mode 100644 user_guide/tutorial/conclusion.html
create mode 100644 user_guide/tutorial/create_news_items.html
create mode 100644 user_guide/tutorial/index.html
create mode 100644 user_guide/tutorial/news_section.html
create mode 100644 user_guide/tutorial/static_pages.html
pc@hyGrON MINGW64 /c:/xampp/htdocs/ddac (master)
$ git remote add origin git@github.com:hygron/ddac.git
pc@hyGrON MINGW64 /c:/xampp/htdocs/ddac (master)
$ git push -u origin master
Enter passphrase for key '/c:/Users/pc/.ssh/id_rsa':
```

To publish completed Maerks Line web application on to the Azure Cloud Services, developer need to open a GitHub account and upload their web application codes on the GitHub. This is because Azure Cloud Services able to connect with the GitHub and publish online. To publish to GitHub, developer can use Git Bash to upload to the GitHub with accessed passphrase key.

The GitHub URL is <https://github.com/hygron/ddac>.

Create Web Application Service on Azure

The screenshot shows the Microsoft Azure portal interface for creating a Web Application Service. The left sidebar contains the 'Create a resource' button and a list of services. The main area displays the 'Web App + MySQL' creation wizard. The 'Web App + MySQL' panel (left) is highlighted with a red box labeled '1' around the 'Create' button. The 'Database Server' panel (right) is highlighted with a red box labeled '3' around the 'OK' button. The 'Web App + MySQL' panel contains the following fields:

- App name:** ddactp038206
- Subscription:** Azure for Students
- Resource Group:** ddactp038206
- Database Provider:** Azure Database for MySQL
- App Service plan/Location:** ServicePlane3e38e96-8408/Centr...
- Database:** Database Settings Required

The 'Database Server' panel contains the following fields:

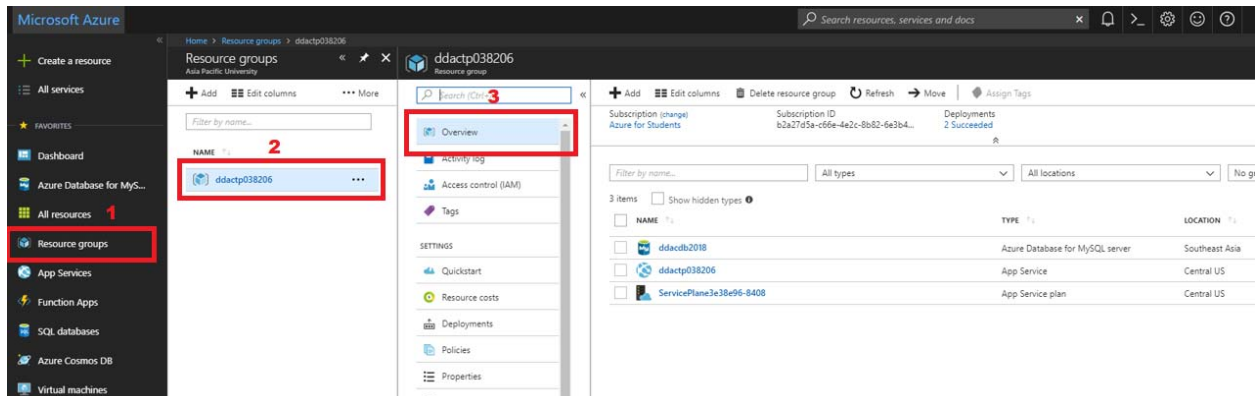
- Server name:** ddactp038206-mysqldbserver
- Server admin login name:** tp038206
- Password:** [masked]
- Confirm password:** [masked]
- Version:** 5.7
- Pricing tier:** Basic, 1 vCore(s), 5 GB
- Database name:** db_admin

Red boxes with numbers 1, 2, 3, and 4 highlight the 'Create' button, the 'App name' field, the 'OK' button, and the 'Create' button respectively.

To publish Web Application to the Azure Cloud Services, developer should do the following step to Create a Web App Service:

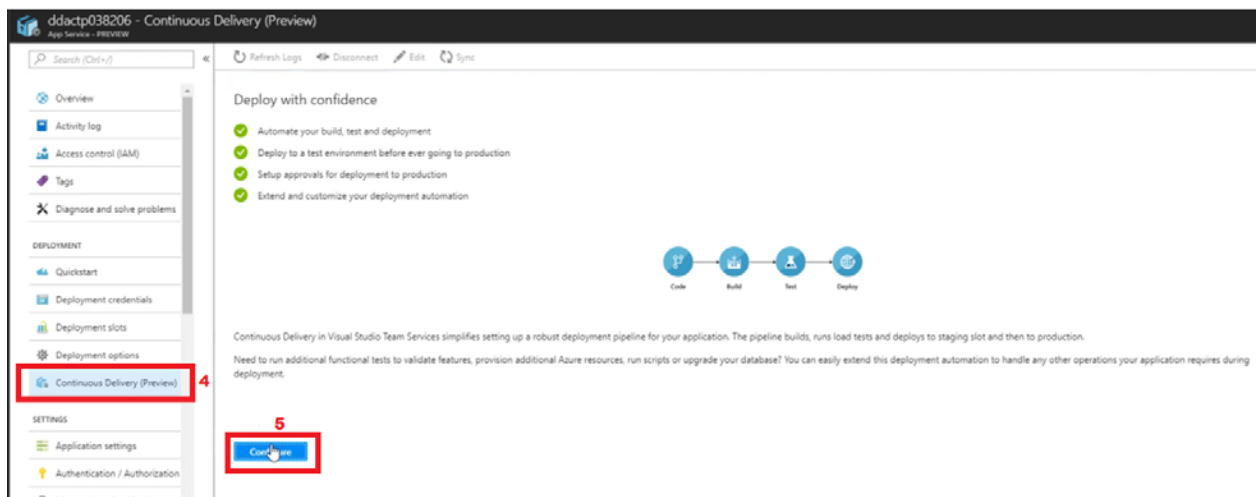
1. Choose the “Web App + MySQL” that can be search on side menu.
2. Select or insert the “App Name”, “Subscription”, “Resource Group”, “Database Provider”, “App Service Plan/Location” and “Database”.
3. Select or insert data of Database Server include “Server Name”, “Login Name”, “Version”, “Database Name”, and choose the plan of Database for the Web Application.
4. Press “OK” button, and then press “Create” Button.

Connect to GitHub from Azure Web App Service



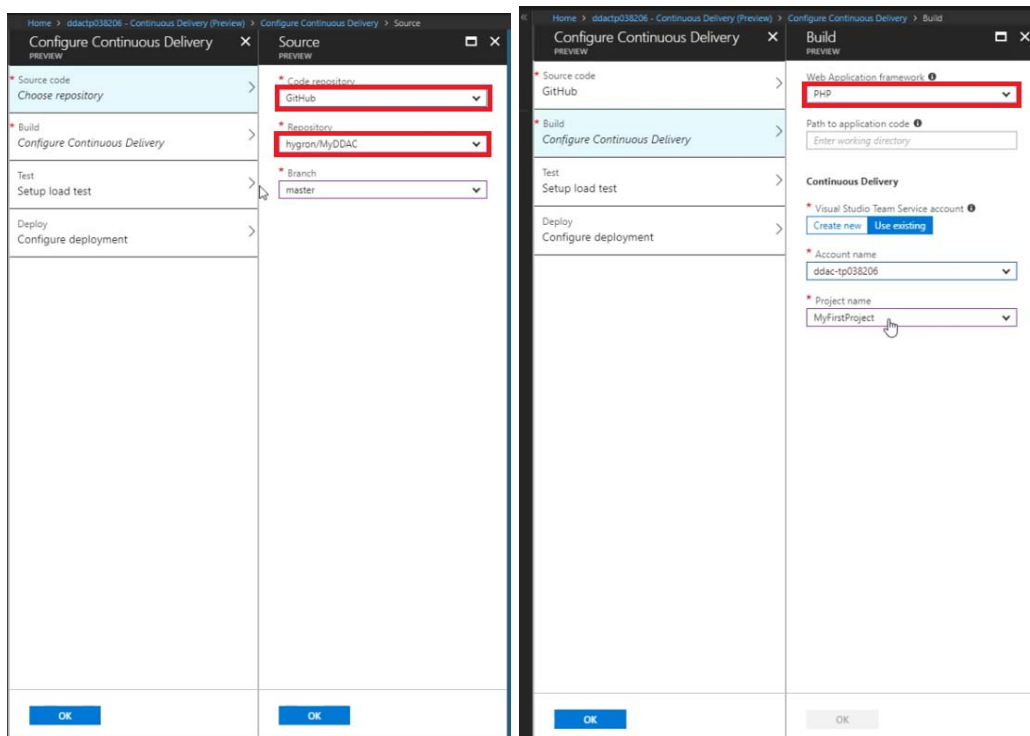
After created the Web App Service, developer need to connect the Azure to the Github php file that developer published by following step:

1. Choose the “Resource Group” that can be search on side menu.
2. Select the Resource Group Name that created before.
3. Select the created App Service in the “Overview” option.

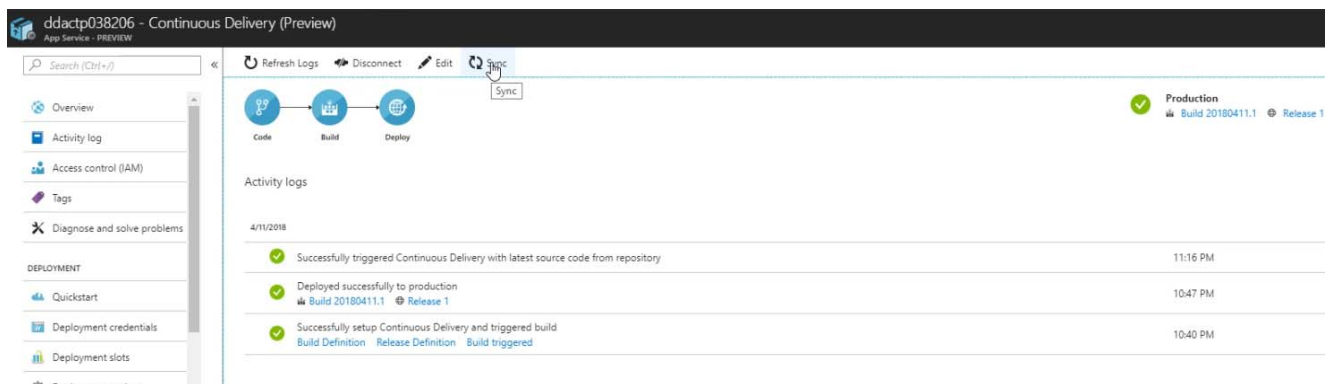


4. Select “Continuous Delivery (Preview)” on side menu of Web Service.
5. Click “Configure” button.

After that the System will prompt out the Configure information.



6. Choose the “Source code” Option and select the correct Repository on Github that developer uploaded and press OK button
7. Choose the “Build” Option and select the correct programing language depend on the source code and press OK button.



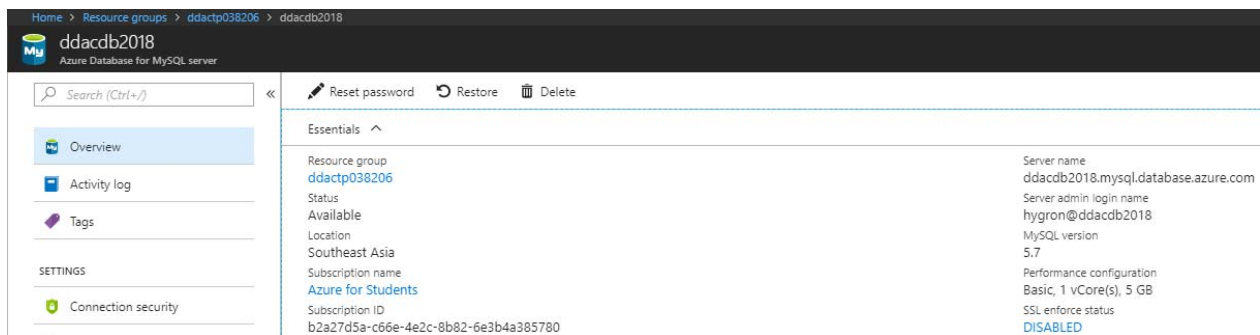
8. Now just wait Microsoft Azure to build and deploy the web application.

After all of the steps above, the website should be link to the source code. But developer still need to set up some setting to manage the database.

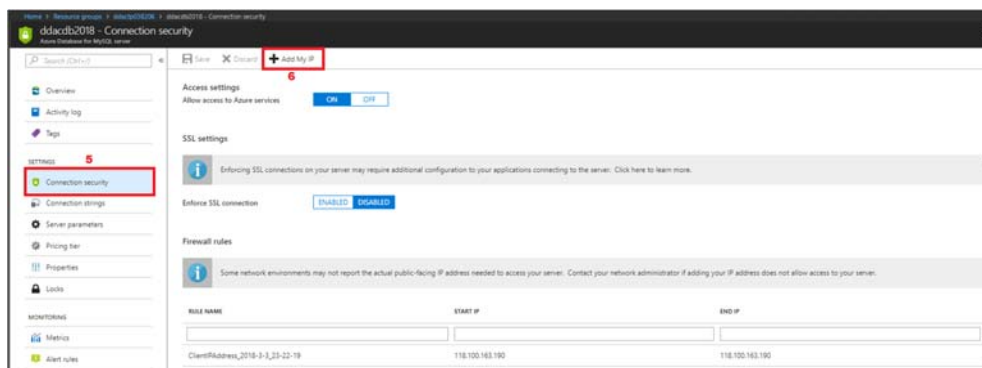
Setting of the Database in Azure

After that, to connect the database to the Azure, developer should setting some option by following step below:

1. Choose the “Resource Group” that can be search on side menu.
2. Select the Resource Group Name that created before.
3. Select the created database that type is “Azure Database for MySQL server” in the “Overview” option.

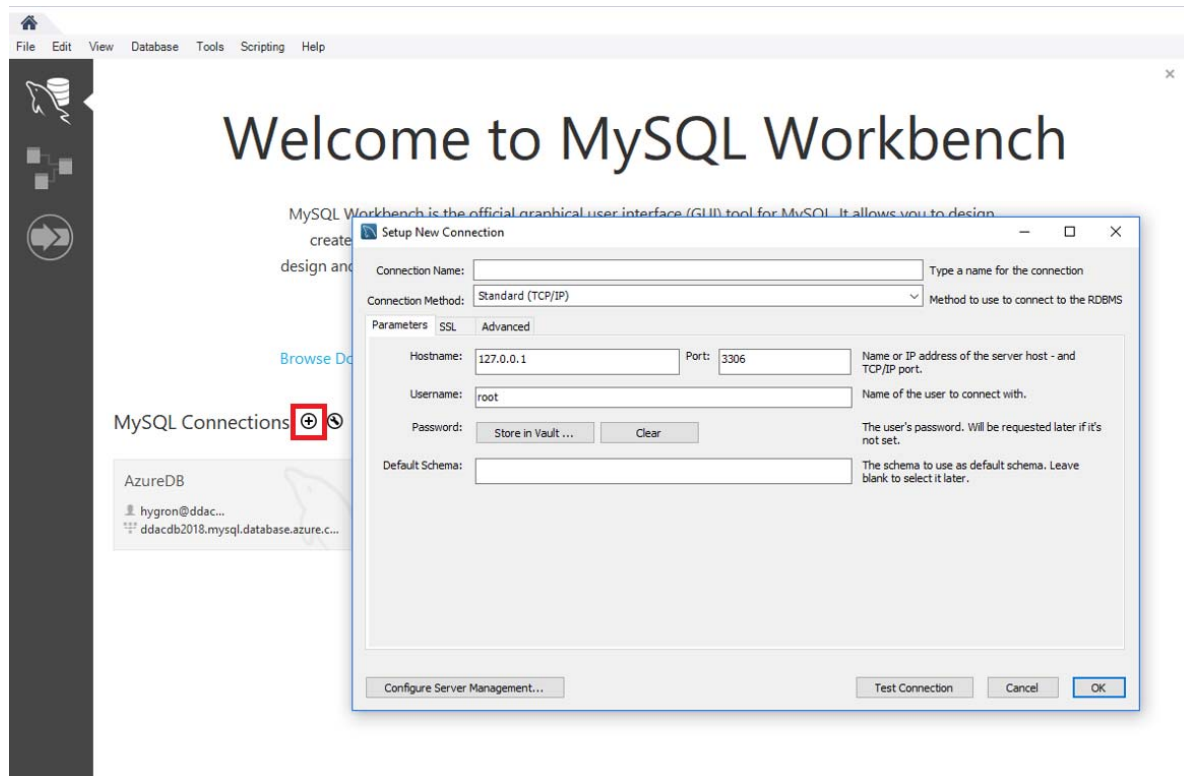


4. The main information of the Database will be show on “Overview” Option.



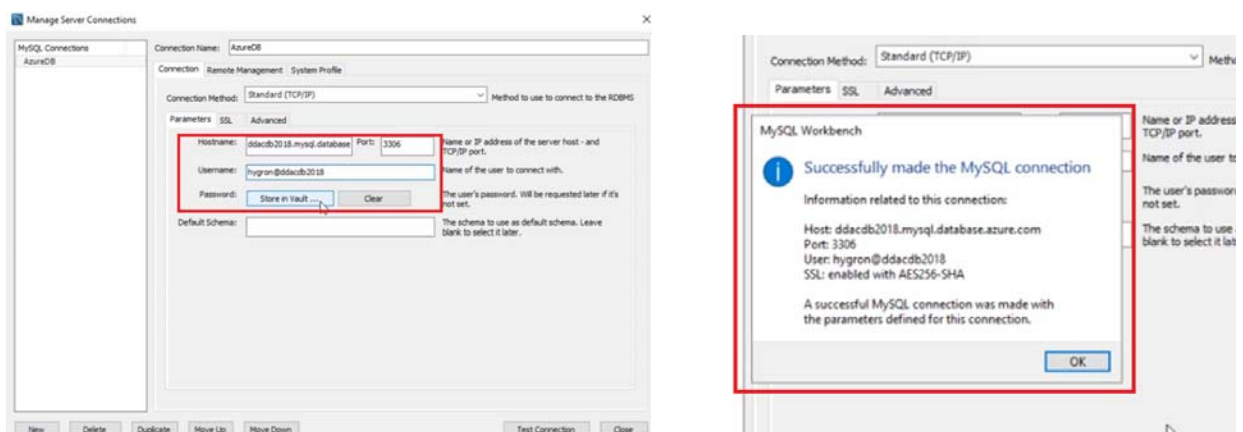
5. Select “Connection Security” in side menu.
6. Click “Add My IP” on top.

Import Local Database to Azure Database

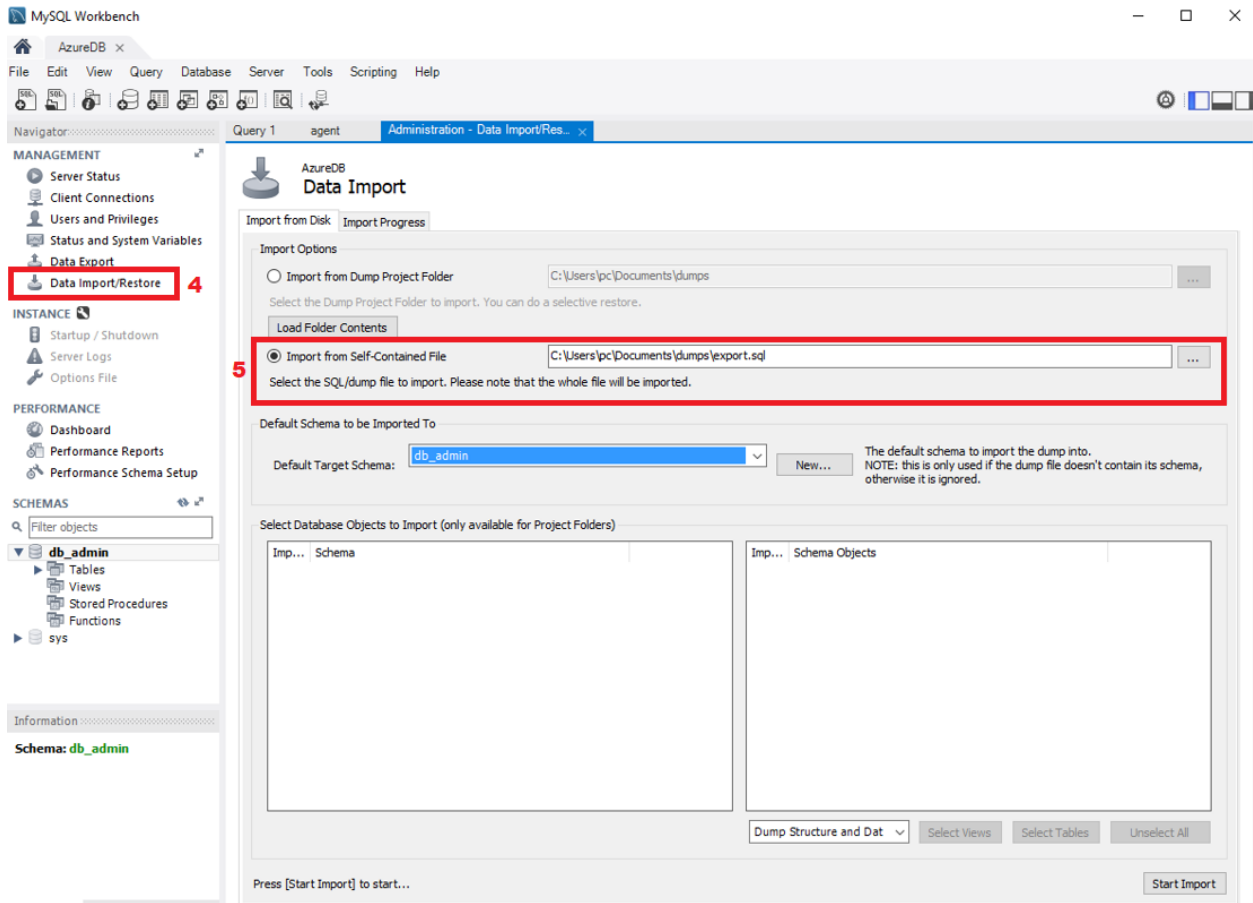


To import the database, developer need to use “MySQL Workbench” to connect local database to Azure database.

1. Press the “+” Add button to create the Azure Connection.



2. Insert the data that refer to MySQL database on Azure.
3. After the Successful Message prompt, click OK and select the Connection that u created.



4. Select “Data Import/Restore”.
5. Select import sql file and wait it done.

Now, the Web application should be able to running the source code on GitHub successfully.

The Web application website is <https://ddactp038206.azurewebsites.net>.

Here is the video about the system demonstration is upload at <https://web.microsoftstream.com/video/cd6473ca-fdfb-422d-8abf-6710cbccb474?list=studio>.

Others

During developer doing all the step above, sometimes developer will face some error or problem, so here is some common solution for some problem that developer may meet.

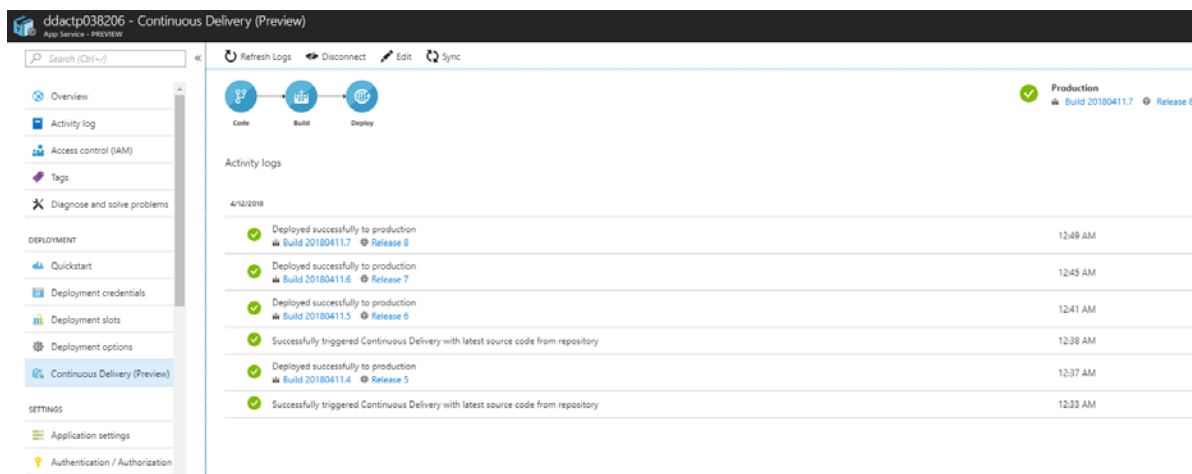


```

69 |
70 | The $query_builder variables lets you determine whether or not to load
71 | the query builder class.
72 |
73 $active_group = 'default';
74 $query_builder = TRUE;
75
76 $db['default'] = array(
77     'dsn' => '',
78     'hostname' => 'ddacdb2018.mysql.database.azure.com',
79     'username' => 'hygron@ddacdb2018',
80     'password' => 'Ddac038206',
81     'database' => 'db_admin',
82     'ddriver' => 'mysql',
83     'dbprefix' => '',
84     'pconnect' => FALSE,
85     'db_debug' => (ENVIRONMENT !== 'production'),
86     'cache_on' => FALSE,
87     'cachedir' => '',
88     'char_set' => 'utf8',
89     'dbcollat' => 'utf8_general_ci',
90     'swap_pre' => '',
91     'encrypt' => FALSE,
92     'compress' => FALSE,
93     'stricton' => FALSE,
94     'failover' => array(),
95     'save_queries' => TRUE

```

1. Do not forget to change the 'hostname', 'username', 'password', and 'database' on the php code file that publish on the GitHub.



2. Sometimes, developer need to Sync and wait the Azure Release new version to update the changers. There is not things go wrongs but developer just need to be patient.

4.2 Application Scaling

S1 Standard	S2 Standard	S3 Standard
1 Core	2 Core	4 Core
1.75 GB RAM	3.5 GB RAM	7 GB RAM
50 GB Storage	50 GB Storage	50 GB Storage
Custom domains / SSL SNI Incl & IP SSL Support	Custom domains / SSL SNI Incl & IP SSL Support	Custom domains / SSL SNI Incl & IP SSL Support
Up to 10 instance(s) Auto scale	Up to 10 instance(s) Auto scale	Up to 10 instance(s) Auto scale
Daily Backup	Daily Backup	Daily Backup
5 slots Web app staging	5 slots Web app staging	5 slots Web app staging
Traffic Manager Geo availability	Traffic Manager Geo availability	Traffic Manager Geo availability
312.48 MYR/MONTH (ESTIMATED)	624.96 MYR/MONTH (ESTIMATED)	1,249.92 MYR/MONTH (ESTIMATED)

Azure's App service can scale vertically (scale up) and horizontally (scale out). Although both scaling methods can help improve the responsiveness of applications, the way that they implement performance improvements are fundamentally different. Vertical scaling deals with the capacity of the resource, such as its storage size, CPU clock speed and RAM capacity (Microsoft, 2017). On the horizontal scaling, it handles instances of a resource by copying or deleting the copy of the application service.

When the application service created for this project is configured to expand according to the indicator. Whenever the average CPU load exceeds 80% for at least 15 minutes, the network application instance will automatically increase by one. Of course, you can create more instances as long as you do not exceed the limits set by the pricing tier. However, the number of instances remains low here because it is only used for testing.

Application services can also be manually expanded when needed. Assuming the current pricing hierarchy, standard S1 does not have enough performance quotations, Maersk Line has may choose other higher pricing tiers, such as standard S1 or even advanced pricing tiers (if they have a budget) if Maersk Line knows the website is in a specific time period, for example, if there is high traffic during the holiday season or when they are in a promotional campaign, it is best to use this scaling method. Extending the application service will undoubtedly increase the responsiveness of the Application service, and it can also support more application instances when scaling.

4.3 Managed Databases (PAAS)

The cloud database may be one of the more widely used services on the cloud platform, such as Microsoft Azure and Google Cloud Platform, because it provides many advantages over traditional databases, and at the same time, it is affordable.

The most popular cloud database type is the relational database as a service (DBaaS), which is essentially a hosted database hosted on the cloud. Such a database that belongs to the Platform as a Service (PaaS) category because the user only needs to manage the data itself, not the platform hosting the database, as the latter is managed by the service provider. Most cloud service providers will also provide the option of hosting databases through virtual machines, basically providing Infrastructure as a Service (IaaS).

This is different from the PaaS database because the operating system, DBMS, and other software must be installed or maintained by the user (Microsoft, 2017). Therefore, compared to IaaS, people can think of PaaS as more user-friendly and easy, because most aspects or components are managed automatically. In fact, a argue can be say that hosting a database from a dedicated server provides more control and can be tailored to specific needs. However, some of the advantages provided by the PaaS database compared to other database hosting methods are unparalleled. As mentioned before, setting up and running the PaaS database is actually much cheaper because the user does not need to obtain the necessary dedicated hardware in addition to the resources needed to maintain the hardware and software.

The PaaS database also typically provides features such as disaster recovery, firewall security, data encryption, database replication, and geographical distribution. This is especially useful for users who are seeking peace of mind when hosting data but are unwilling to do all the work.

The nature of the PaaS database makes it suitable for teams who need limited budgets and limited development time, especially small teams that cannot to hire large among people to develop and manage databases. However, the PaaS database lacks the amount of control that is hosted in the database on the dedicated server. If one person really wants more control over their database while still looking for a cost-effective solution, they can choose the IaaS database because it provides this.

For examples, for the fully-managed cloud databases such as Microsoft Azure SQL Database and Cloud SQL for Google Cloud Platform. These databases can manage themselves, such as automatically performing backups and replications, and therefore require almost zero maintenance. Although the PaaS databases provided by Google Cloud Platform and Microsoft Azure are powerful, easy to access, scalable, and available, they have limited support for database engines. Now, the Azure SQL database supports only SQL Server Engine and will soon have MySQL support (under Azure Database written in MySQL, currently in preview). On the other hand, Google Cloud Platform Cloud SQL supports MySQL 5.6 and 5.7, and will soon support PostgreSQL.

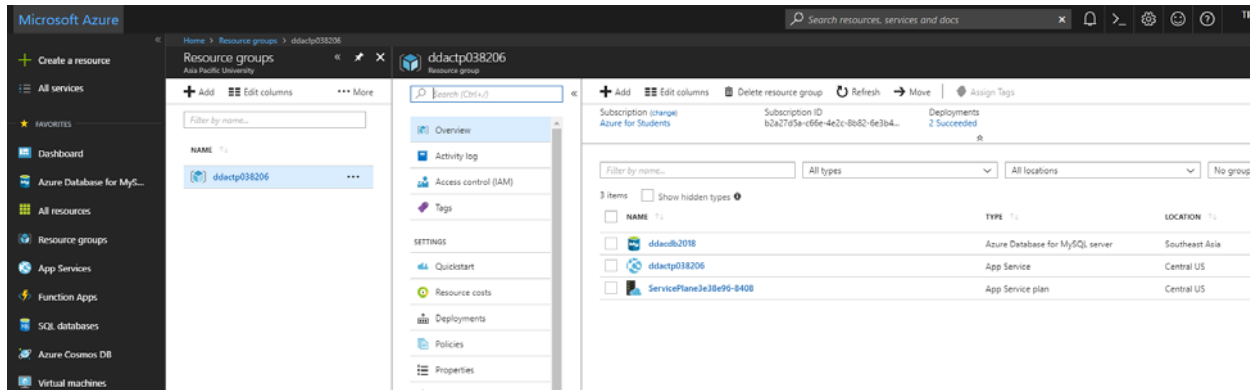
For the Amazon AWS Relational Database Service (RDS), because of the previously database solution lacks self-management capabilities, it tends to favor container services. Despite this, AWS RDS still offers many features and has many similarities with competitors. AWS RDS has the advantage of supporting more database engines such as SQL Server, MySQL, PostgreSQL, Oracle, and MariaDB. All of the above database engines are hosted by AWS on Elastic Block Store (EBS) volumes. They also have a fifth database engine called Aurora, which uses a different infrastructure and claims to provide up to five times the throughput of standard MySQL running on the same hardware. This is achieved by tightly integrating the database engine with the SSD-backed virtualized storage tier dedicated to database workloads (Amazon, 2017).

The above comparison and discussion is only talking about some of the SQL databases provided by cloud platform providers and the differences between them. The main of the view is obviously limited because it does not involve other aspects include pricing, availability and location. When selecting which cloud platform provider to use for the project, they should also need to consider other products and services they provide, such as NoSQL database services, storage, and networking products, because it is easier and more manageable to adhere to a cloud platform provider.

In the conclusion, the PaaS database offers many benefit and advantage than traditional databases and also can improve application development, publishing, and maintenance

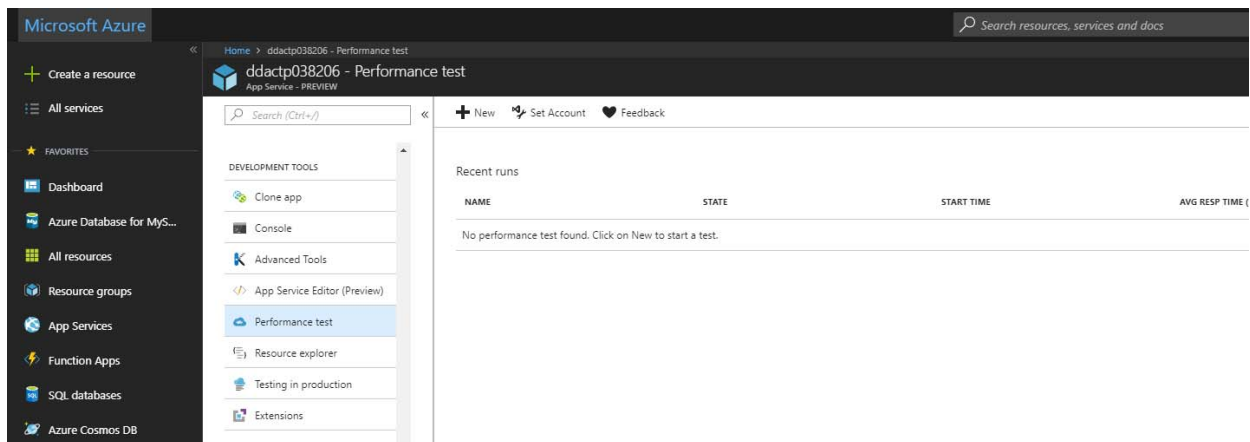
5 Testing

5.1 Performance Testing

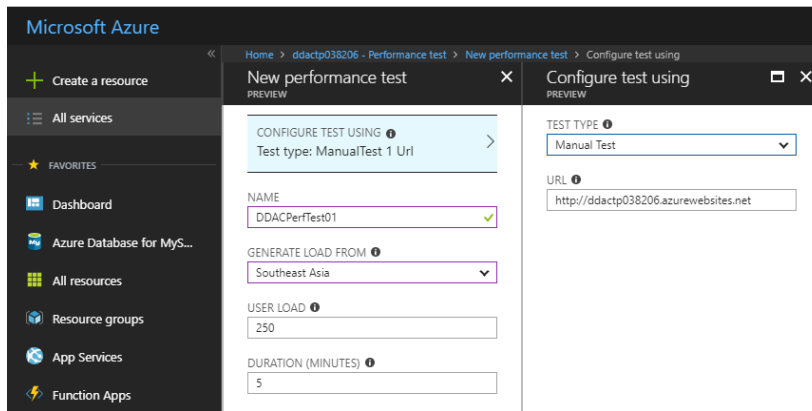


To connect the do the performance testing in Azure developer should:

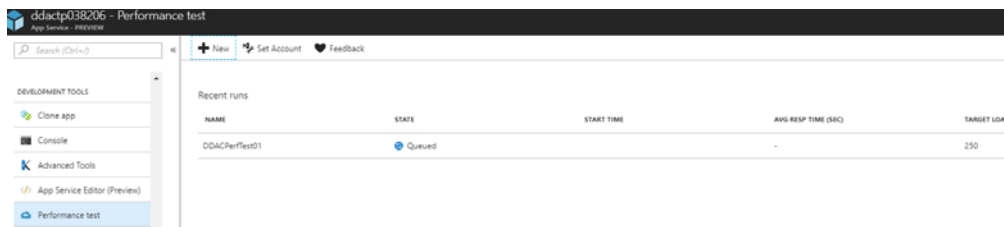
1. Choose the “Resource Group” that can be search on side menu.
2. Select the Resource Group Name of the Web Application.
3. Select the created App Service(ddactp038206) in the “Overview” option.



4. Select “Performance Test” on side menu of Web Service.
5. Click “New” to create new performance test.

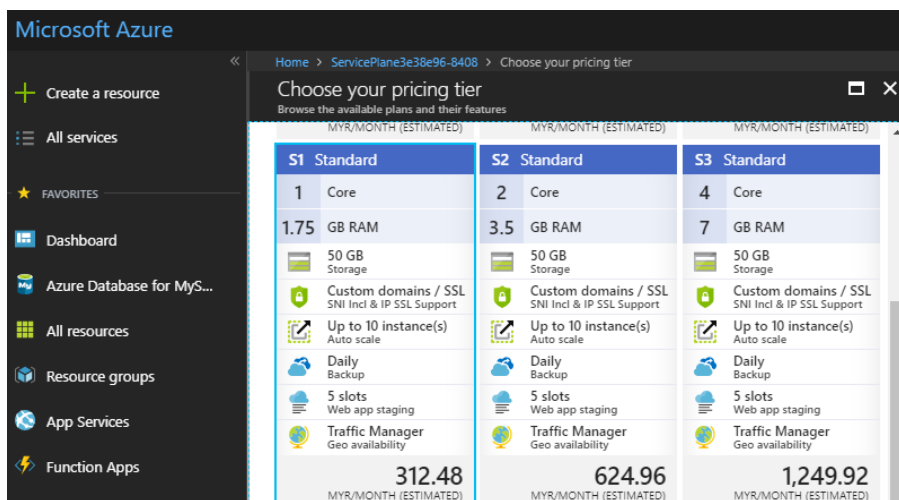


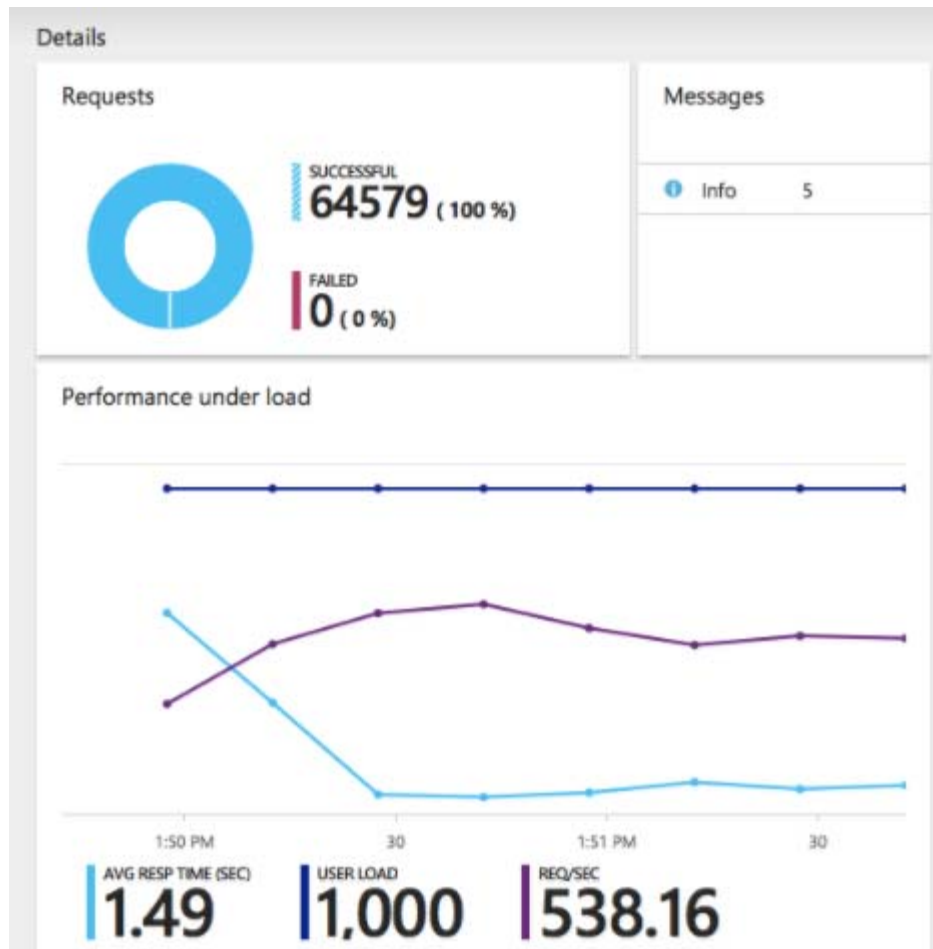
6. Insert all details and click Done button and “Run Test” button.



7. Wait the System finish the performance test and view the result.

In this part, user needs to conduct a performance test to test the performance of the web application. This test will be conducted in 1000 user loads in 2 minutes, in Standard 1 plan.





From the data gained in the performance tests above, it can be seen that the result has 100% successful rate, and the average response time is 1.49 second. Assume that maximum acceptable threshold for page load times on this application is 3 seconds, which means this plan is accepted and can handle 1000 concurrent users.

5.2 Unit Testing

Test ID	Test Case	Test Case Description	Expected Result	Actual Result	Status
T1	Login	User able to login Web application as admin.	Login Successful as admin account.	Login Successful as admin account.	Pass
T2	Login	User able to login Web application as agent.	Login Successful as agent account.	Login Successful as agent account.	Pass
Below Test Case are using Admin Account					
TAD1	Insert Ship	User able to insert Ship data into database.	Insert ship data Successful.	Insert ship data Successful.	Pass
TAD2	Insert Item	User able to insert Item data into database.	Insert item data Successful	Insert item data Successful	Pass
TAD3	Insert Schedule	User able to insert Booking Schedule data into database.	Insert booking schedule data Successful	Insert booking schedule data Successful	Pass
TAD4	Insert User Account	User able to insert User Account data into database.	Insert user account data Successful	Insert user account data Successful	Pass
TAD5	Update Ship	User able to update Ship data into database.	Update ship data Successful.	Update ship data Successful.	Pass
TAD6	Update Item	User able to update Item data into database.	Update item data Successful	Update item data Successful	Pass

TAD7	Update Schedule	User able to update Booking Schedule data into database.	Update booking schedule data Successful	Update booking schedule data Successful	Pass
TAD8	Update User Account	User able to update User Account data into database.	Update user account data Successful	Update user account data Successful	Pass
TAD9	Delete Ship	User able to delete Ship data from database.	Delete ship data Successful.	Delete ship data Successful.	Pass
TAD10	Delete Item	User able to delete Item data from database.	Delete item data Successful	Delete item data Successful	Pass
TAD11	Delete Schedule	User able to delete Booking Schedule data from database.	Delete booking schedule data Successful	Delete booking schedule data Successful	Pass
TAD12	Delete User Account	User able to delete User Account data from database.	Delete user account data Successful	Delete user account data Successful	Pass
Below Test Case are using Agent Account					
TAG1	Insert Booking	User able to insert Booking data into database.	Insert booking data Successful.	Insert booking data Successful.	Pass
TAG2	Register Item	User able to register Item data into database.	Insert item data Successful.	Insert item data Successful.	Pass
TAG3	Register Customer	User able to register Customer data into database.	Insert customer data Successful.	Insert customer data Successful.	Pass

6.0 Conclusion

In conclusion, Maerks Line web application allows the user to make reservation for shipment, and also able to create container, shipyards, and manage ships. The web application is developed based on the requirement that stated in the assignment question paper. Besides, this web application has achieved to goal of providing a good platform for the user to manage the shipment.

Microsoft Azure is a good platform for the web application to deploy on it. This platform provides a great solution to the Maerks Line Company as it meets the needs of Maerks Line web application. Besides, it is very easy and convenient for the software developer to deploy their web application with few simple steps. Unlike the others platform, it only takes few minutes to deploy the web application to the cloud. Not only that, Microsoft Azure also provides SQL Database that helps a lot of software developer to solve the problem of storing the data.

The software developer had gained lots of knowledge in this area, which is develop and deploy the web application on Microsoft Azure App Service, and traffic control. In this assignment, I had learned the web application deployment, traffic control, and other useful functions. I believe that I could apply it in my future jobs and help to improve my career as it provides many useful features.

7.0 References

- 1) Azure.microsoft.com. (2018). *Pricing Calculator / Microsoft Azure*. [online] Available at: <https://azure.microsoft.com/en-us/pricing/calculator/> [Accessed 10 Apr. 2018].
- 2) Digital Trends. (2017). *Battle of The Browsers: Edge vs. Chrome vs. Firefox vs. Opera vs. Vivaldi*. [Online] Available from: <https://www.digitaltrends.com/computing/best-browser-internet-explorer-vs-chrome-vs-firefox-vs-safari-vs-edge/> [Accessed 10 Apr. 2018].
- 3) GitHub. (2018). *Build software better, together*. [online] Available at: <https://github.com/settings/keys> [Accessed 10 Apr. 2018].
- 4) Hafling, J. (2017). *Advantages of A Simple Web Design*. [Online] Available from: <https://www.jayhafling.com/articles-for-seo/advantages-of-a-simple-web-design/> [Accessed 10 Apr. 2018].
- 5) Microsoft. (2017). *ASP.NET MVC Overview*. [Online] Available from: [https://msdn.microsoft.com/en-us/library/dd381412\(v=vs.108\).aspx](https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx) [Accessed 10 Apr. 2018].
- 6) Tutorials Point. (2017). *MVC Framework – Introduction* [Online] Available from: https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm [Accessed 10 Apr. 2018].