**Air Ticket Reservation Management System**

**ATRMS**

**Myo Thet Tun – 2918551**

submitted in partial fulfilment on the degree of

***BSc.(Hons) in Computing Science***

***Griffith College Cork***

*May 2020*

*Under the supervision of Dale Amenda*

**Disclaimer**

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the Degree of Bachelor of Science in Computing at Griffith College Cork, is entirely my own work and has not been submitted for assessment for an academic purpose at this or any other academic institution other than in partial fulfilment of the requirements of that stated above.

**Signed: \_\_\_\_\_\_\_** **\_\_\_\_\_\_\_\_\_\_ Date: \_\_\_29.05.2020\_\_\_\_\_\_\_\_\_\_**

# Acknowledgements

First of all, the author would like to tell a little bit about his student career in Griffith College Cork. Studying for a four-year honour’s degree is not an easy task. It is like running a long-distance marathon. The author started attending this BSc(Hons) in Computing Science degree in September 2016. During that four years’ time, the author has learnt many modules which are essential and important for my future career as an IT professional. During these years, the author has seen some students did not succeed, some got exhausted and left, some changed career and so on. It is a long-term process, all of us had to overcome all the challenges and difficulties from the beginning to reach this final stage.

Thus, here, in successful completion of this final year project, *Air Ticket Reservation Management System*, **ATRMS**, the author would like to thank people who taught him, who helped him, and who supported him throughout the course.

First and foremost, the author would like to thank his beloved parents from Burma who always support and encourage him for his career. And the author would like to thank Dale Amenda, Head of Computing Science faculty Cork, who is also his lecturer since the first year of the course. And all the lecturers who taught him various modules throughout these four years, the author would like to thank them as well. And finally, the author remembers all of his classmates since the first year until now and would like to thank them because they help each other since the beginning of this course.

It is obvious that without all these help and support, the author cannot reach until this stage and this project will not be accomplished.

Table of Contents

[Acknowledgements ii](#_Toc41607667)

[List of Equations iv](#_Toc41607668)

[List of Figures iv](#_Toc41607669)

[List of Tables iv](#_Toc41607670)

[Abstract v](#_Toc41607671)

[Chapter 1. Introduction 1](#_Toc41607672)

[1.1 Document Structure 1](#_Toc41607673)

[1.2 Problem Definition 1](#_Toc41607674)

[1.3 System Objectives 1](#_Toc41607675)

[1.4 Approach 1](#_Toc41607676)

[Chapter 2. Background 2](#_Toc41607677)

[2.1 Literature Review 2](#_Toc41607678)

[2.1.1 Existing System Study 2](#_Toc41607679)

[2.2 Related Work 3](#_Toc41607680)

[2.2.1 Requirements Specification 3](#_Toc41607681)

[Chapter 3. Methodology 4](#_Toc41607682)

[3.1 Technologies 4](#_Toc41607683)

[3.2 Techniques 5](#_Toc41607684)

[Chapter 4. System Design Specification 6](#_Toc41607685)

[4.1 Entity Relationship Diagram 6](#_Toc41607686)

[4.1.1 Foreign Keys and Relationships Constraints 6](#_Toc41607687)

[4.2 Use Case Diagram 7](#_Toc41607688)

[4.2.1 Use Case Descriptions 7](#_Toc41607689)

[4.3 Class Diagram 8](#_Toc41607690)

[4.4 Initial Screen Designs 8](#_Toc41607691)

[Chapter 5. Implementation 9](#_Toc41607692)

[5.1 Database Structure 9](#_Toc41607693)

[5.2 Screen Design 10](#_Toc41607694)

[Chapter 6. Testing and Evaluation 13](#_Toc41607695)

[6.1 Unit Testing 13](#_Toc41607696)

[6.2 Integration Testing 14](#_Toc41607697)

[Chapter 7. Conclusion 15](#_Toc41607698)

[7.1 Strengths and Weaknesses of the System 15](#_Toc41607699)

[7.2 Future Enhancements 15](#_Toc41607700)

[References 17](#_Toc41607701)

# List of Equations

There are no equations in this document.

# List of Figures

[Figure 4. 1: Entity Relationship Diagram 6](#_Toc41652499)

[Figure 4. 2: Use Case Diagram 7](#_Toc41652500)

[Figure 4. 3: Class Diagram 8](#_Toc41652501)

[Figure 4. 4: Initial Screen Design - 1 8](#_Toc41652502)

[Figure 4. 5: Initial Screen Design - 2 8](#_Toc41652503)

[Figure 4. 6: Initial Screen Design - 3 9](#_Toc41652504)

[Figure 5. 1: Login Form 10](#_Toc41652520)

[Figure 5. 2: Home Page 10](#_Toc41652521)

[Figure 5. 3: Signup Form 10](#_Toc41652522)

[Figure 5. 4: Passenger Form 11](#_Toc41652523)

[Figure 5. 5: Flight Form 11](#_Toc41652524)

[Figure 5. 6: Reservation Form 11](#_Toc41652525)

[Figure 5. 7: Contact Form 12](#_Toc41652526)

[Figure 5. 8: About Us Page 12](#_Toc41652527)

[Figure 5. 9: View Table Form 12](#_Toc41652528)

# List of Tables

[Table 4. 1: Use Case Descriptions 7](#_Toc41652674)

[Table 5. 1: User 9 Table 5. 2: Payment 9](#_Toc41652684)

[Table 5. 3: Passenger 9 Table 5. 4: Flight 9 Table 5. 5: Reservation 9](#_Toc41652685)

[Table 6. 1: Check Login 13](#_Toc41652732)

[Table 6. 2: Check Passenger 13](#_Toc41652733)

[Table 6. 3: Check Fight 14](#_Toc41652734)

[Table 6. 4: Check Reservation 14](#_Toc41652735)

[Table 6. 5: Check Update and Delete Record 14](#_Toc41652736)

[Table 6. 6: Check View Table 14](#_Toc41652737)

# Abstract

This project is called *Air Ticket Reservation Management System*, ***ATRMS***. The purpose of this project is to create a *Computerised Reservation System,* ***CRS*** for an air ticket reservation agency. The system can perform all common tasks which are usually available in a regular reservation system such as registering the passenger, creating new flights, ticket reservations and payments, viewing all records in various data tables. It can perform deleting and updating abilities on records in all tables.

The technology used to create the system are *HTML* and *CSS* with *Bootstrap* framework for screen design, *MySQL* for database, *JavaScript* for front-end validation and *PHP* for back-end data manipulations.

This document records all procedures and steps taken throughout the development lifecycle process of the system. It includes definition of the problem, problem’s background, and old system study and requirement specification, designing the system, implementation of the actual system and testing and evaluation process. And finally, the strengths and weaknesses of the system are defined and future enhancement of the system in the last chapter of the document.

# Chapter 1. Introduction

## 1.1 Document Structure

Chapter 1 describes the introduction of the system and it includes the problem overview, system objectives and the approach to develop the system. Chapter 2 discusses the background of the system which includes the existing system study and the requirements specification. Chapter 3 discusses the technologies and techniques used in the system. System design and diagrams are describing in chapter 4. The detail implementation of the system is discussed in chapter 5. Various test cases and results are shown in chapter 6. And finally, in the last chapter, chapter 7, summary, strengths and weaknesses of the system and future enhancements are discussed. In the last part of this document is the reference and bibliography.

## 1.2 Problem Definition

The system is called “**Air Ticket Reservation System**”. It is a web-based air ticket reservation system. It should include passenger detail and flight detail where passenger and flight information are stored. Then, the registered passenger should be able to book the flight according to the registered flight details in the reservation form. Payments should be made as soon as the booking has confirmed. The administrator should be able to view and check all the tables in a tabular format. Reports should be generated as requested.

## 1.3 System Objectives

The system objective is to produce a complete working *Air Ticket Reservation System****.*** The system is a kind of *CRS (Computerised Reservation System)*, an information system which promotes ticket sales and provides fast and reliable information of bookings, prices and availability of services and products in tourism industry. CRS is later upgraded into *GDS (Global Distribution System)* and are largely used by travel agencies for selling tickets from multiple airlines.

## 1.4 Approach

Firstly, the required information is collected from the tables and form designs from the existing systems to determine what type of data fields and values should be included in our new system.

Then, the next step is to prepare a set of requirement specifications for the system and the types of technology to be considered, such as which programming language and database format to be used, and to submit these initial approach details to the project’s supervisor. And the required duration and tasks are set and ‘*Tasks, Durations and Dependencies Table*’ is created and draw ‘*Gantt Chart*’ according to this table.

After the approval of the supervisor, the various tasks of the design phase are started, such as drawing Entity Relationship Diagram, Use Case Diagram, Class Diagram, and Initial Screen Designs.

The next step is the ‘*Implementation*’ phase. Firstly, the database structure is created using MySQL and the actual screen designs are created using Bootstrap framework. After that is the implementation of all the forms and the connection with the database. This phase is the main development phase of the system.

The last phase is the ‘*Testing and Evaluation*’ phase. In this phase, many testing cases are performed such as individual form testing, integrated testing, and whole system testing. Finally, the system is fully developed, and the strengths and weaknesses of the system are identified, and the weaknesses are to be better in the future updated versions.

# Chapter 2. Background

## 2.1 Literature Review

This stage involves studying the existing system or similar systems and defining the requirements specification in accordance of the existing system study. According to the *Problem Definition* defined above in the *section 1.2*, *Existing Systems Study* and *Requirements Specification* stages are performed.

### 2.1.1 Existing System Study

The author studies many existing systems and they are YouTube videos and some web references as well. The list is as follows.

<https://www.youtube.com/watch?v=PkuQCBBpS28>

This is a video of a nice java-based *Air Ticket Reservation System*. This video gives an idea of how an air ticket reservation system look like and the usage of appropriate controls on the data forms such as where to use calendar controls, combo boxes, radio options, etc.

<https://www.youtube.com/watch?v=H28jJYk0Xds>

This is another good example of how a reservation system looks like. The original system is created as Java-based system.

<https://1000projects.org/airline-ticket-reservation-system-project-in-java.html/comment-page-1#comments>

<https://creately.com/diagram/example/iid8qkkq2/Airline+Reservation+System+ER+Diagram>

These websites give an idea of diagrams and specifications for the system.

<https://www.youtube.com/watch?v=_Kvw5fZkOMc>

This video series give a complete guide to create a CRUD application with PHP and MySQL database.

## 2.2 Related Work

### 2.2.1 Requirements Specification

User’s requirements specification is defined according to the following possible scenarios.

**Note:** the underlined words will be the entities for the system.

**Scenario A (Book the ticket)**

A passenger wants to travel from origin (A) to destination (B). Therefore, he searches for the flight information on the website whether there is a flight available from A to B at a given date and time (departure and arrival). If he finds the flight information at a desired date and time, he will book the flight for that date and time. (The system should display either the eco or business class has chosen and show the available seats in the specified class.) If the seat is available, he will book the seat and fill up the form for passenger details. And proceed to payment details form and pay the amount due. After the payment is confirmed, a printable receipt copy will be generated and sent to passenger’s email address for record. (As cancellation is still possible before the travel date, a given allowance period for cancellation prior to the actual departure date should be identified here).

**Scenario B (Cancel the ticket)**

If the passenger wants to cancel the ticket within the allowance period, the system should check the ticket details and accept the cancellation, i.e. delete the reservation and move it into cancellation table, if possible, to know how many has been cancelled at a later period).

**Scenario C (Administrator create the various entries)**

Administrator performs CRUD for the flight information, can be able to view and correct errors in passenger information, reservation, and cancellation details.

# Chapter 3. Methodology

## 3.1 Technologies

**Bootstrap Framework**

Bootstrap is a front-end framework with the combination of *HTML*, *CSS*, and *JavaScript* together. [1]It helps building mobile responsive websites more quickly and easily. It was [1]originally developed by Twitter. But because of its [1]free, versatile, and intuitive nature, it is now used for various purposes from developing web applications to WordPress themes.

One [1]can build complex web pages using standard *HTML* and customise with Bootstrap according to the needs. It comes with [1]jQuery plugins which provides additional functionality such as buttons, tooltips, carousels, etc. It also has *JavaScript* form validations in many form controls.

**MySQL**

MySQL is one of the most powerful *Relational Database Management System* **(RDBMS)**. Moreover, it is [3]open source and is freely available which uses the most popular language for database manipulation, Structured Query Language, SQL. MySQL [2]becomes the leading database for creating web-based applications due to its performance, reliability, and user-friendly features. It plays an important role for [3]almost every open-source PHP application. MySQL is used by many popular web industries including [2]Facebook, YouTube, Twitter, etc.

**XAMPP Control Panel**

[4]XAMPP is an open-source software package developed by Apache Friends. And it includes [4]Apache distributions for Apache server, MariaDB, PHP and Perl. The name XAMPP stands for [4]*Cross-platform Apache MariaDB PHP and Perl*. XAMPP works as a local host or local server installed on developer’s own computer. And it allows the developer to test the sever-side web applications such as [4]MySQL, PHP, Apache, and Perl projects, without actually uploading to the remote server.

**JavaScript**

[5]JavaScript is a scripting language which allows the implementation of more complex dynamic features on web pages rather than just static contents created with HTML and CSS. JavaScript is [5]the third layer of the standard web technologies after HTML and CSS. Even though some bit of front-end JavaScript form-validations come with Bootstrap, some functionality for the validations are implemented in the separate “*.js*” files for the system. These JavaScript features help the author to perform most form-validations in front-end instead of validating everything in the database back-end manipulations.

**PHP**

PHP is an [6]open-source general-purpose scripting language. It [6]stands for “*Hypertext Pre-processor*” but the acronym does not match exactly. PHP is a [6]complete programming language and it can be embedded in HTML tags. It is used for [6]server-side scripting to create dynamic web pages and to perform database manipulations on web pages.

## 3.2 Techniques

Initial screen designs are created with MS Visual Studio according to the diagrams defined on the *System Design Specification* stage. These designs are submitted to the client to check if they meet the requirements. At this point, if some part of the design does not meet the requirements, this specific part of the design is to be changed in order to meet the requirements.

Web-based screen designs are built using *Bootstrap* framework according to the *System Design Specification*. After that, it is to test the front-end form validations in each form using JavaScript. Then, integrates each form with the whole system and test the system validations again.

The next step is to create the SQL database depending on the diagrams and creating PHP files and test the connection to the database and perform data manipulations on the local server with XAMPP.

# Chapter 4. System Design Specification

## 4.1 Entity Relationship Diagram

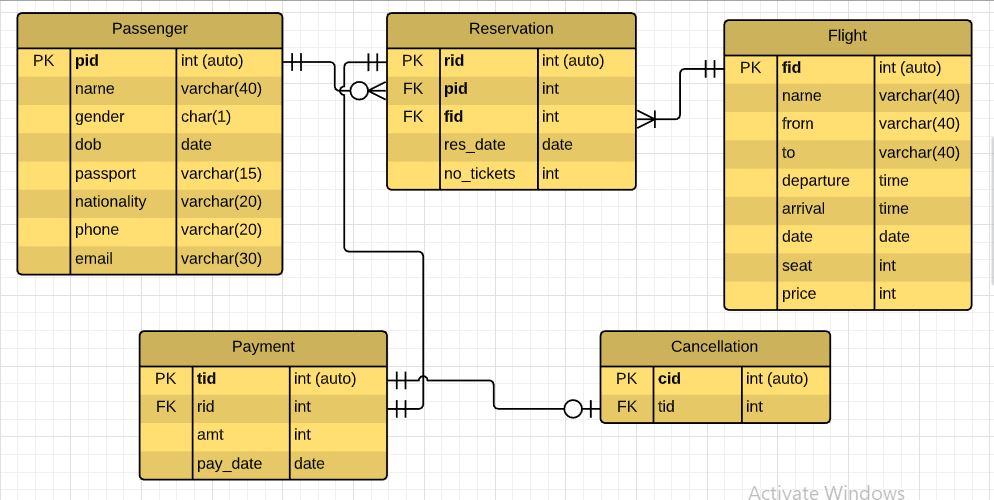


Figure 4. : Entity Relationship Diagram

### 4.1.1 Foreign Keys and Relationships Constraints

**Note: primary keys** in all tables use *auto-numbering system*.

The relationship constraints for the tables are as follows.

**Passenger - Reservation**

One passenger may book zero or many reservations. But one reservation is made by one passenger only.

**Flight - Reservation**

One reservation can have only one flight. But a flight has one or many reservations. \*(Assume all the flights are running well and have at least one reservation for each one.)

**Payment – Reservation**

One reservation has only one payment and similarly, one payment can be made for only one reservation.

**Payment – Cancellation**

Zero or one payment should be cancelled, but only one cancellation is for one payment.

**Note:** When cancellation is made, the cancelled ticket will be moved to cancellation table for recording purpose and the original record in the “Payment” table will be deleted.

## 4.2 Use Case Diagram

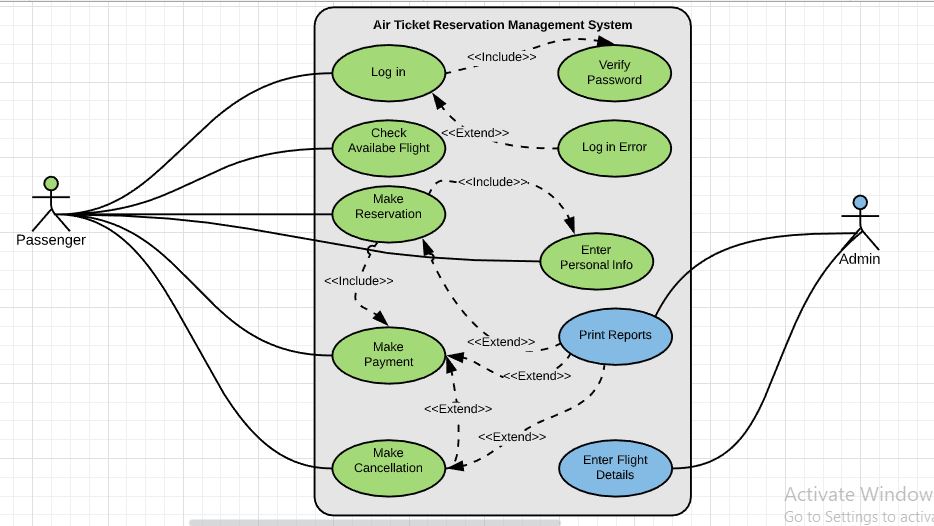


Figure 4. : Use Case Diagram

### 4.2.1 Use Case Descriptions

|  |  |  |  |
| --- | --- | --- | --- |
| No | Case | Actor | Description |
| 1 | User Login | User | Passenger enters username and password to login |
| 2 | Verify Login | User | It is included in the login use case to verify that the username and the password are correct. |
| 3 | Login Error | User | This case is extended from Login to show error message when the username or password does not match. |
| 4 | Check Available Flight | Passenger | A passenger can check whether a flight is available for his or her destination at a desired date and time. |
| 5 | Make Reservation | Passenger | A passenger can make a reservation for the ticket(s) if he or she finds the available flight. |
| 6 | Enter Personal Details | Passenger | A passenger must enter his or her details when reservation is made. This use case is included as part of make reservation. |
| 7 | Make Payment | Passenger | After reservation, user will be taken to make payment form. This use case is included as part of make reservation, too. |
| 8 | Make Cancellation | Passenger | This use case is extended from Make Payment use case because a passenger can make a cancellation after the payment or maybe not. |
| 9 | Enter Flights Details | Admin | Admin can perform the operations to flight details information. |
| 10 | Print Reports | Admin | Admin may print out the reports of payments, reservations, and cancellations. This use case is extended from three use cases, Make Payment, Make Reservation and Make Cancellation. |

Table 4. 1: Use Case Descriptions

## 4.3 Class Diagram

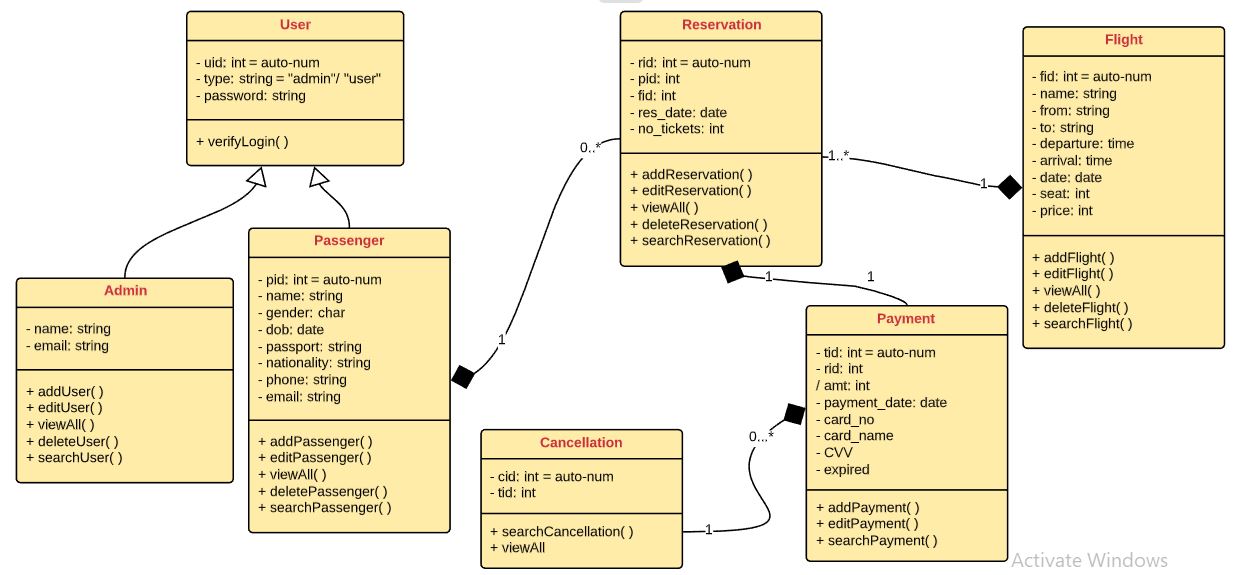


Figure 4. : Class Diagram

## 4.4 Initial Screen Designs

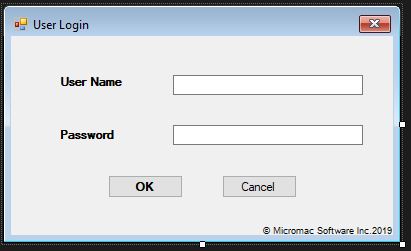


Figure 4. : Initial Screen Design - 1

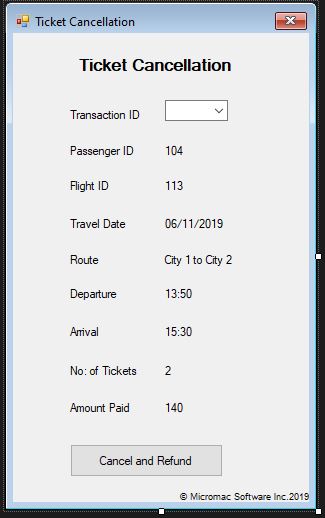
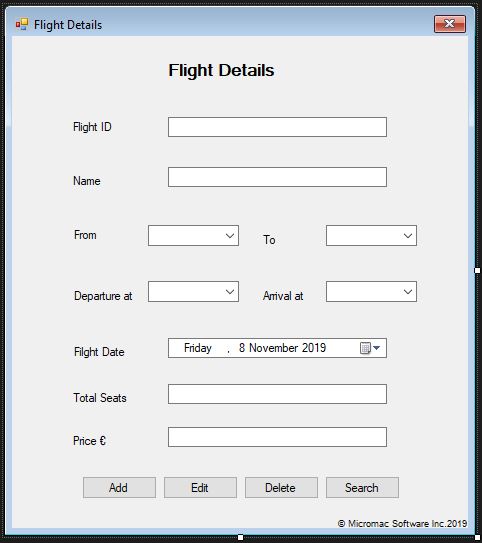
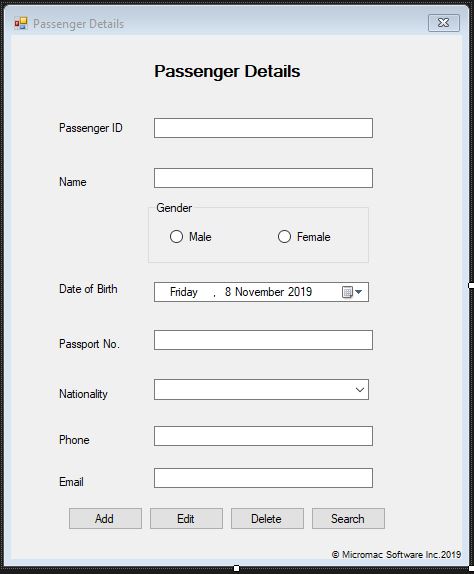


Figure 4. : Initial Screen Design - 2

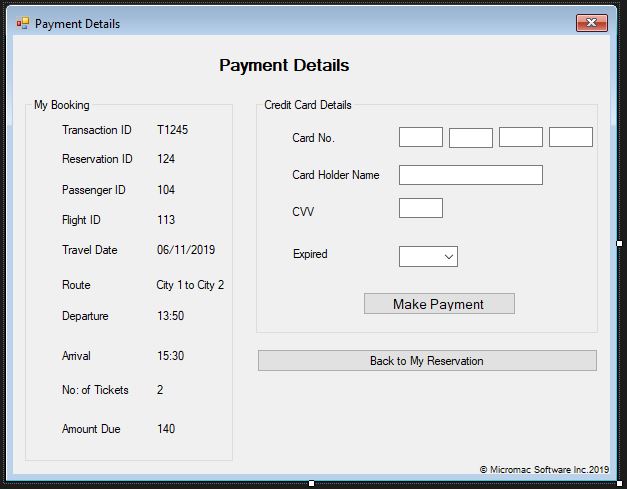
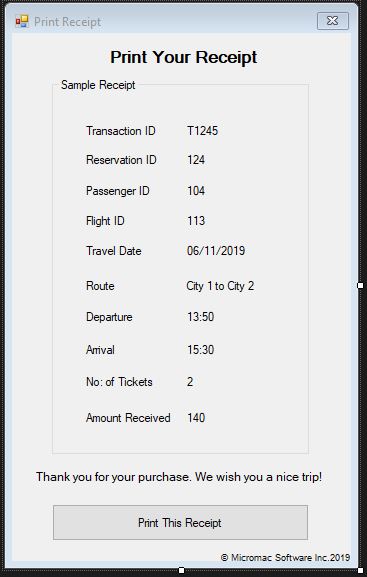
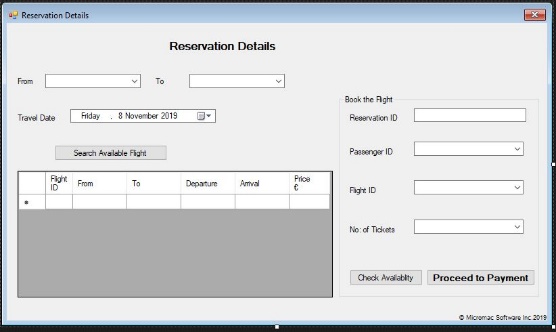


Figure 4. : Initial Screen Design - 3

These initial screen designs are created in Microsoft Visual Studio and they are submitted to the organisation to check whether the system design meet the user’s requirements.

# Chapter 5. Implementation

This chapter describes the structure of the system. This includes database structure and screen design.

## 5.1 Database Structure

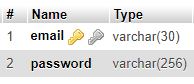
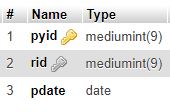
 

Table 5. : User Table 5. : Payment

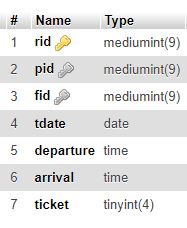


Table 5. : Passenger Table 5. : Flight Table 5. : Reservation

## 5.2 Screen Design

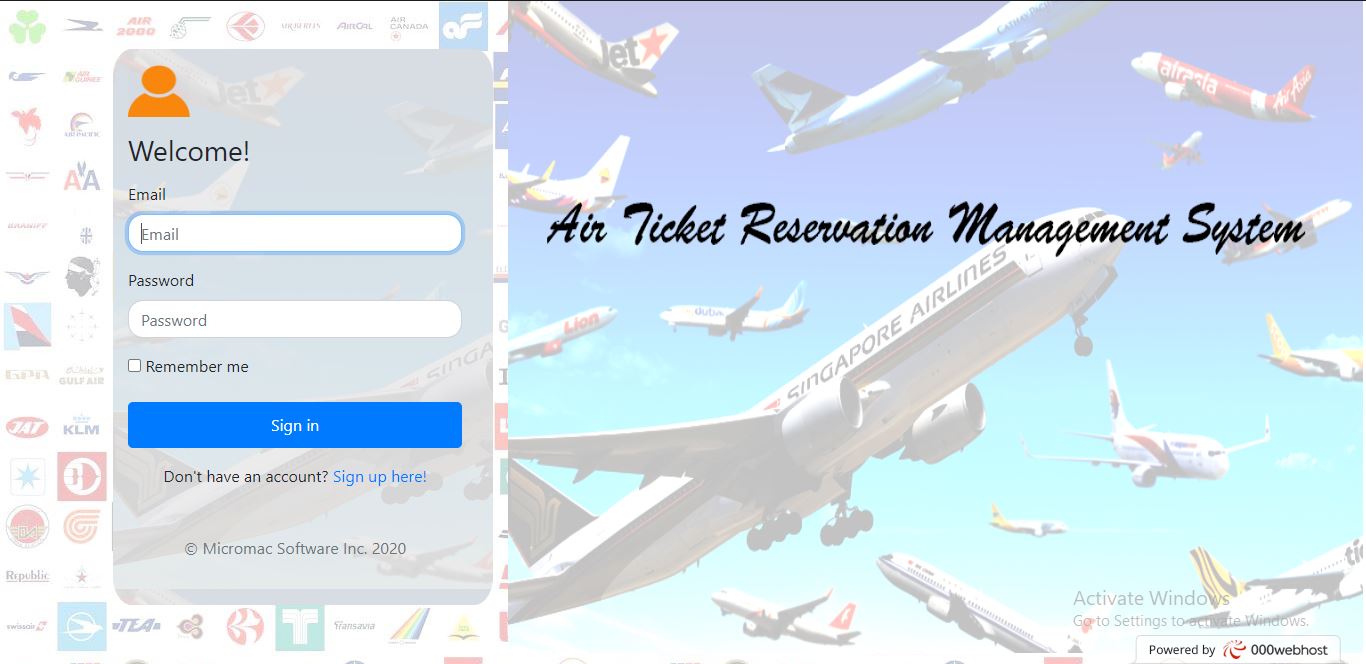


Figure 5. : Login Form

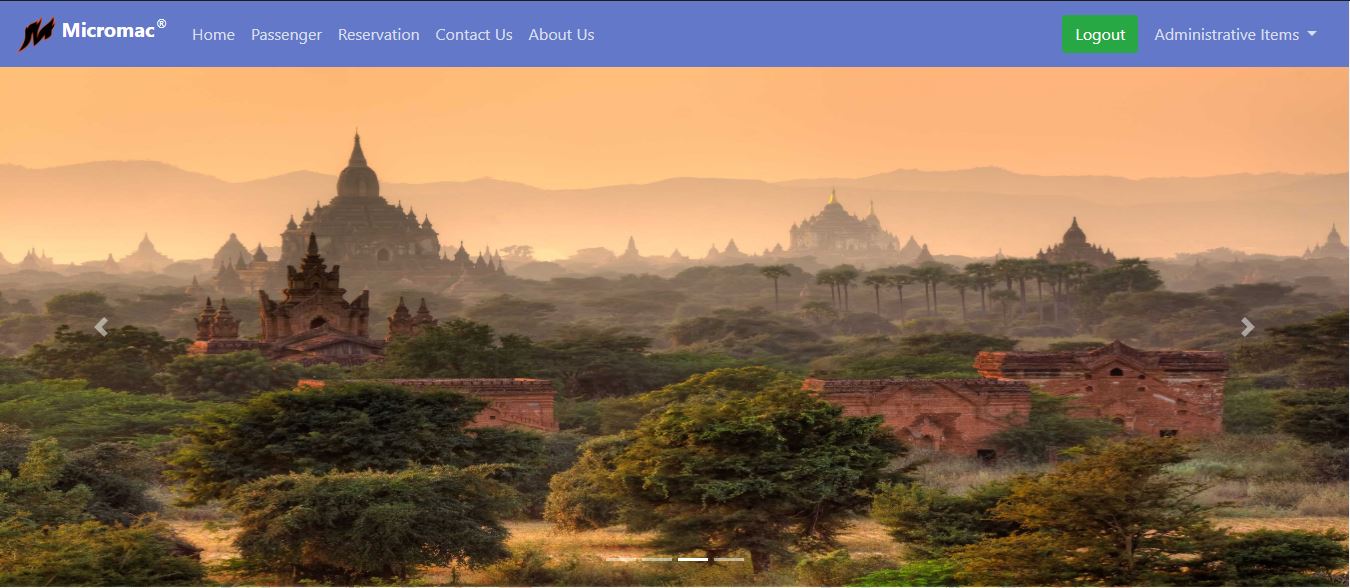


Figure 5. : Home Page

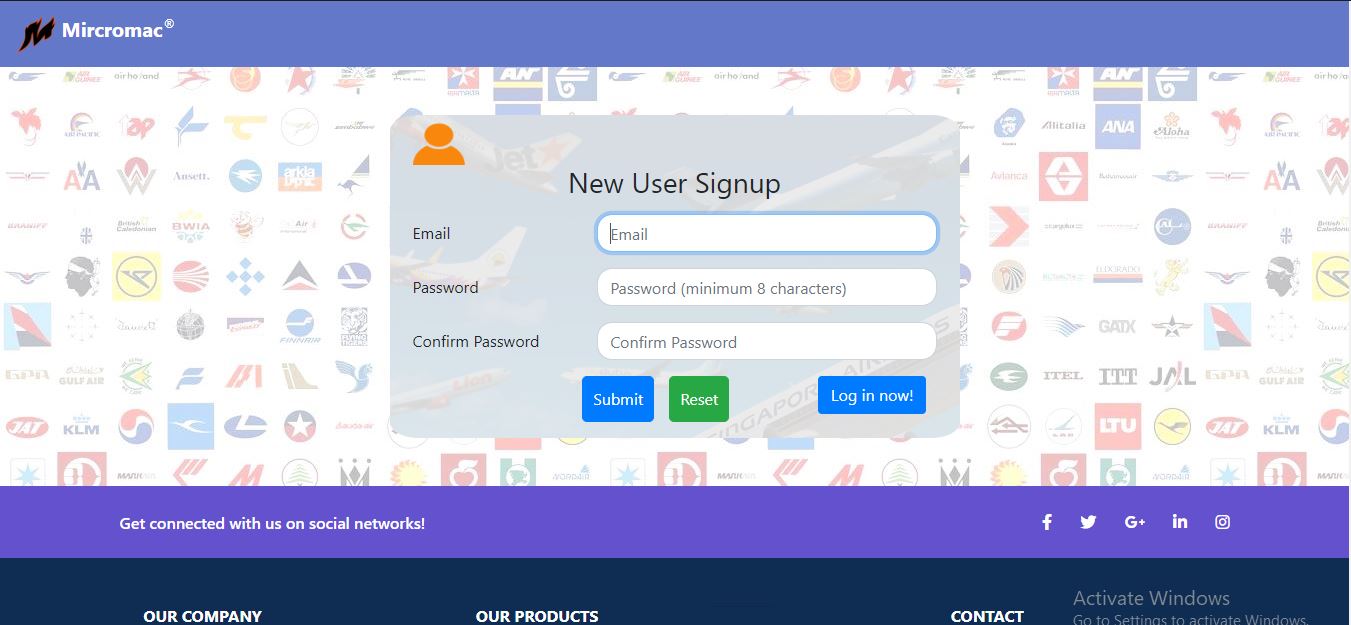


Figure 5. : Signup Form

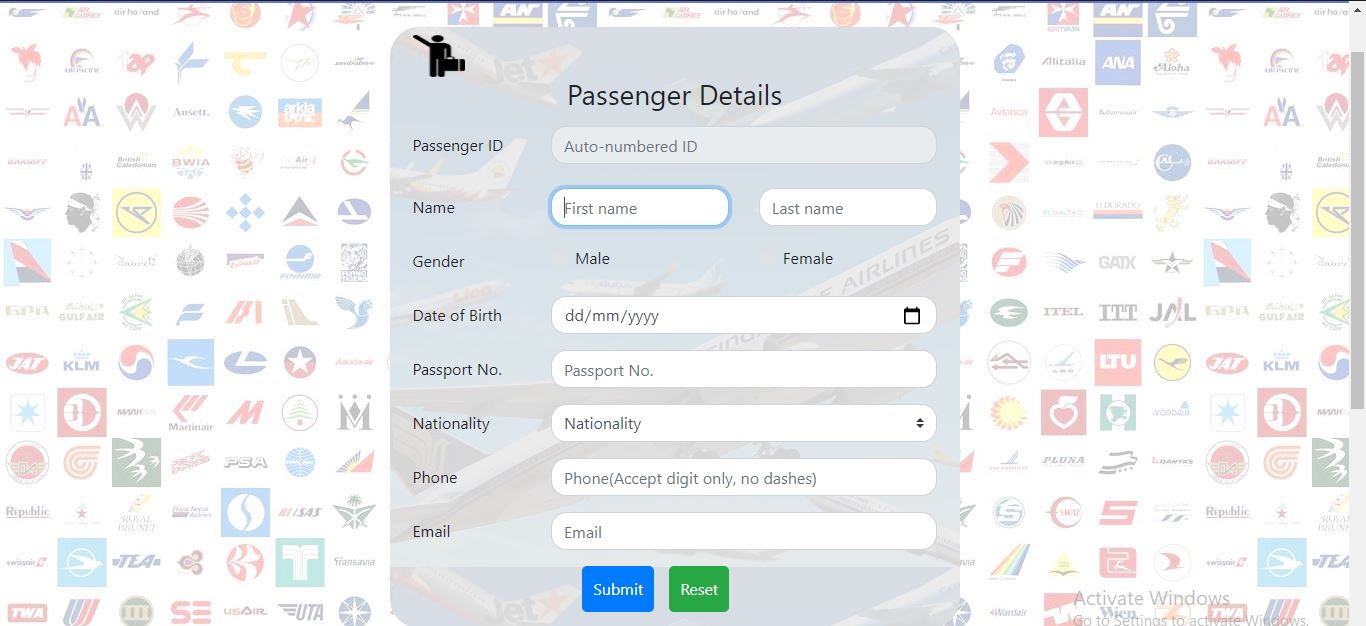


Figure 5. : Passenger Form

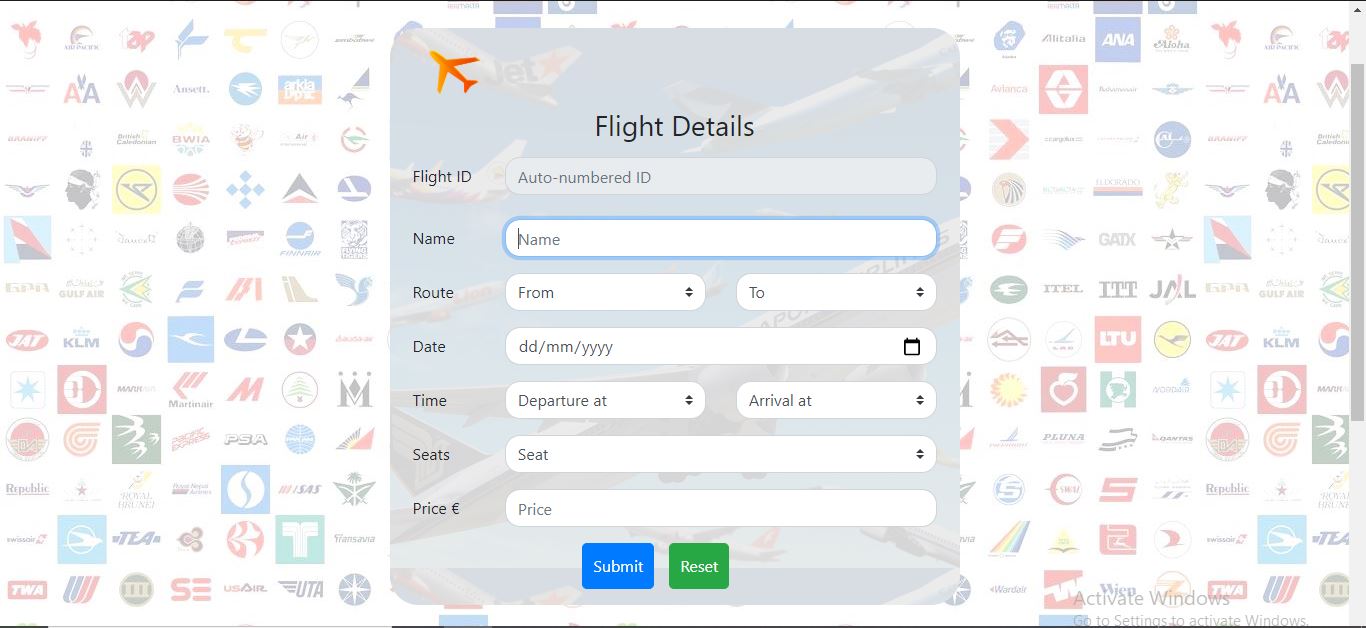


Figure 5. : Flight Form

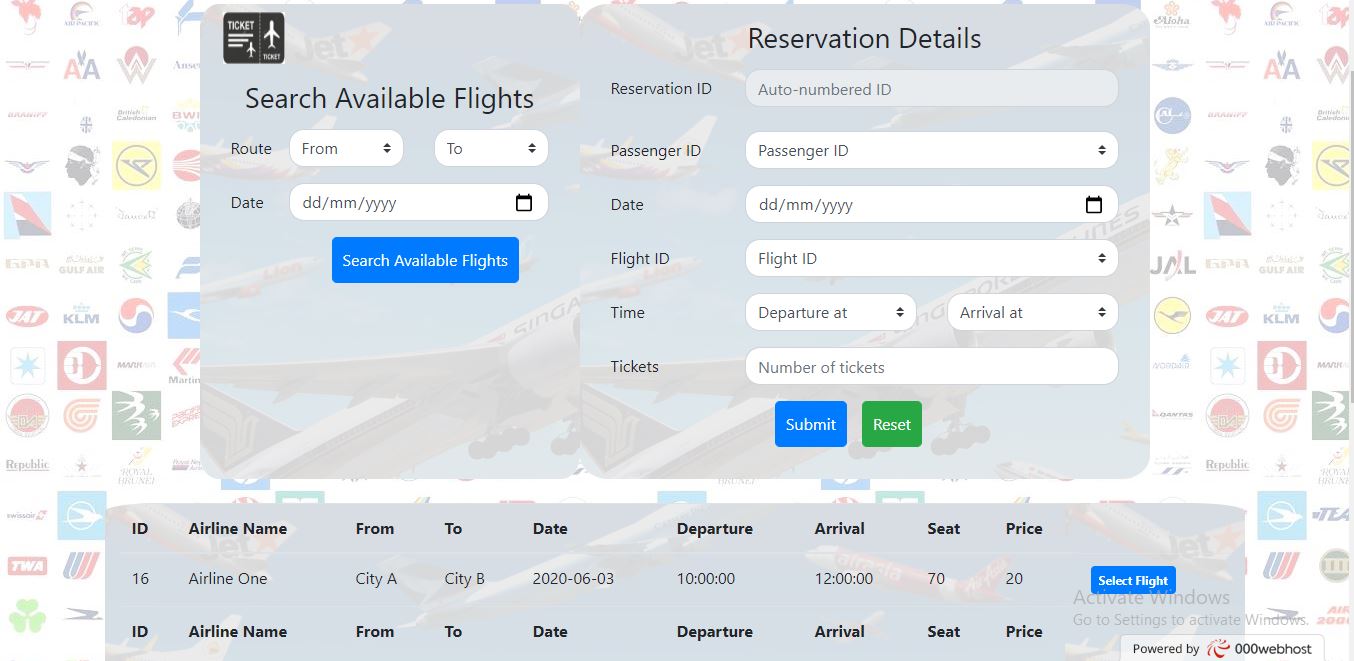


Figure 5. : Reservation Form

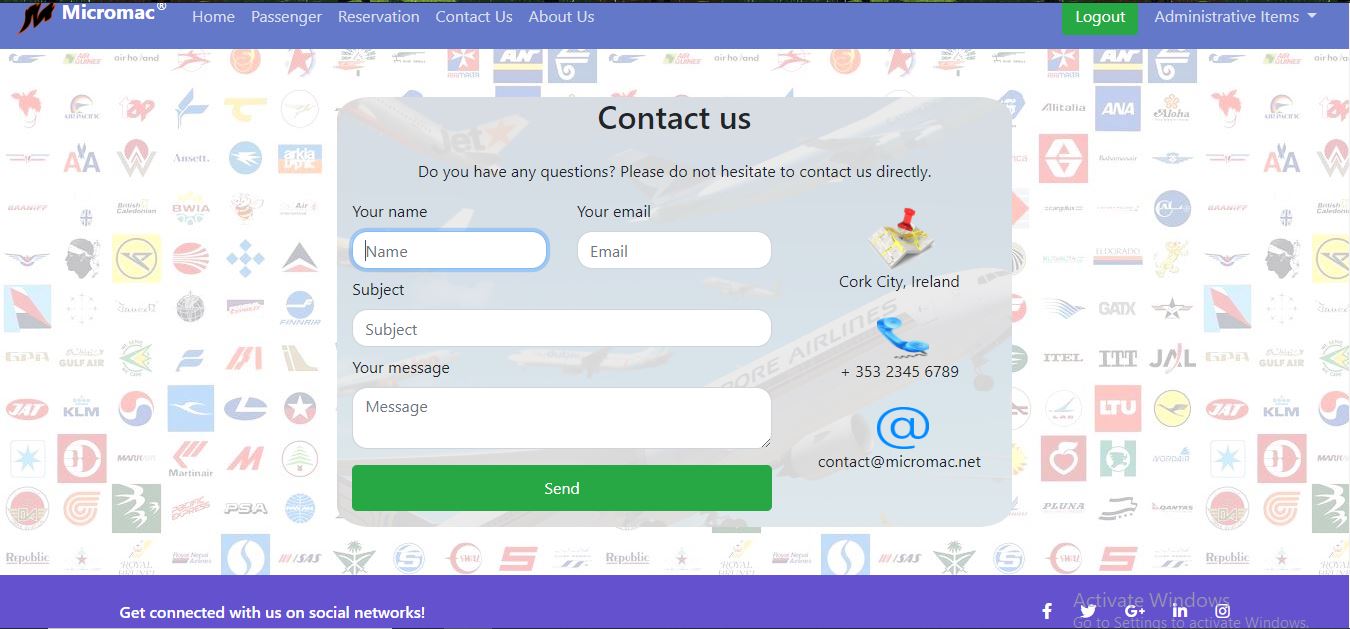


Figure 5. : Contact Form

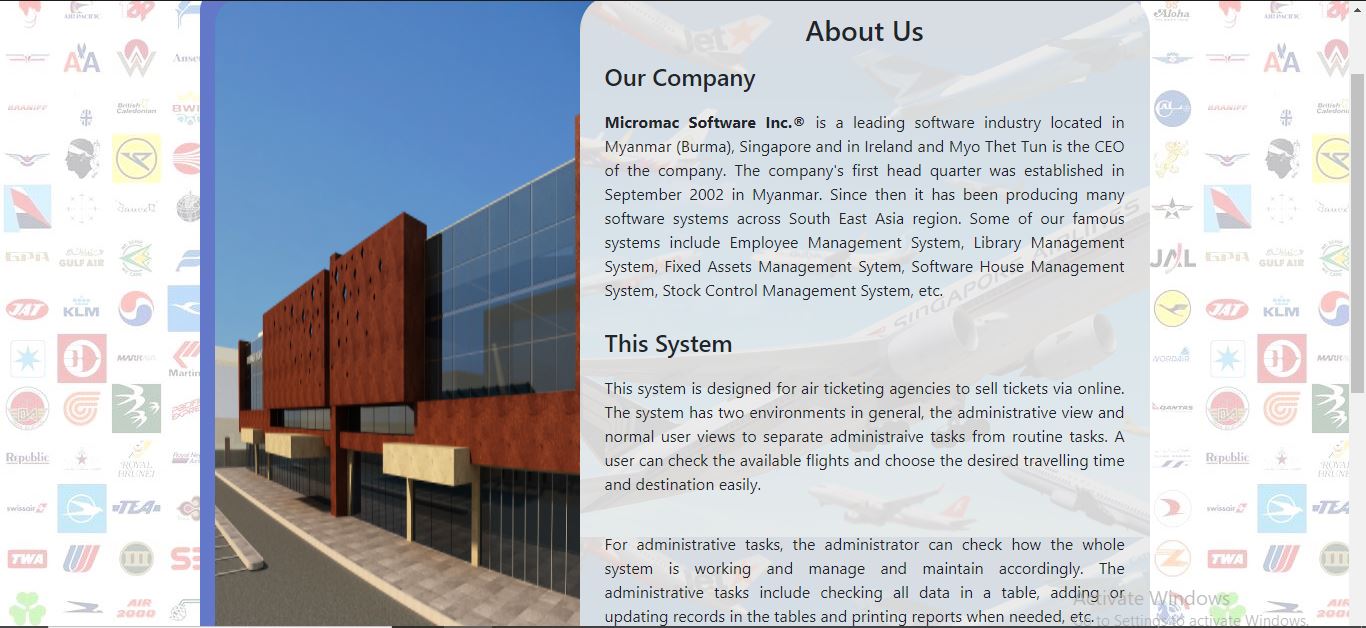


Figure 5. : About Us Page

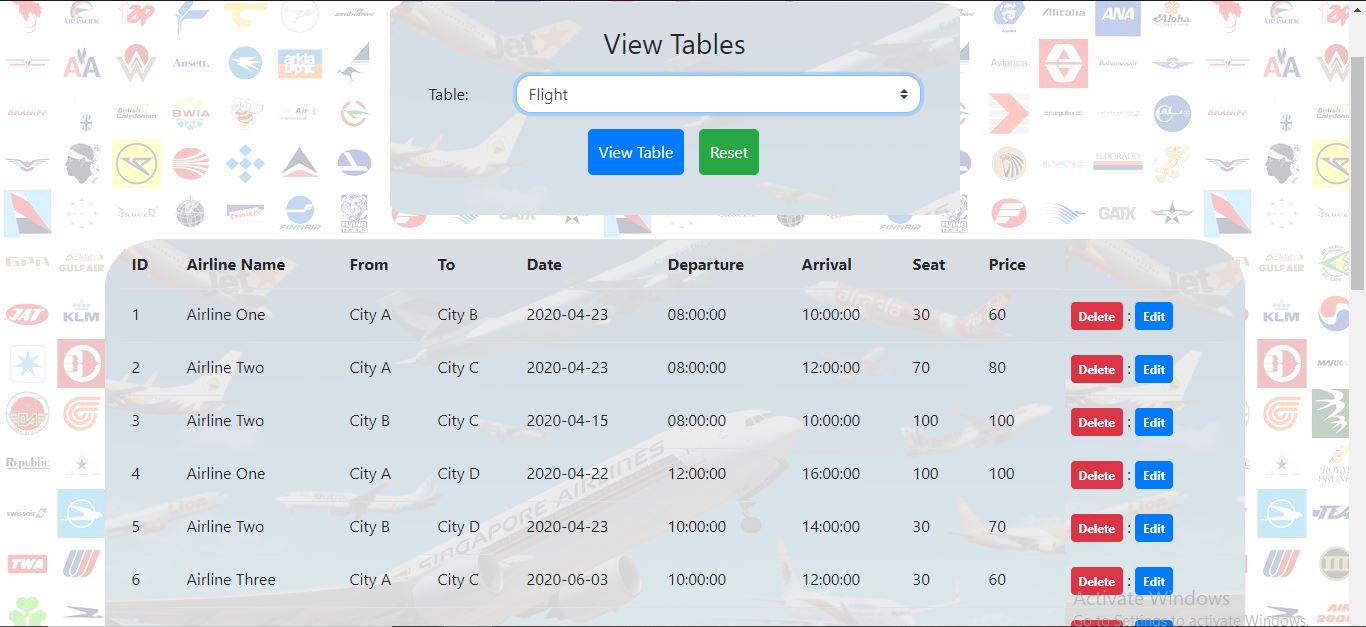


Figure 5. : View Table Form

# Chapter 6. Testing and Evaluation

Software testing can be defined as [7]an activity to check whether the actual results meet the expected results to make sure that the software system is a defect-free system. In another term, it is also known as [7]*Verification of Application Under Test* **(AUT)**. There are many stages involves in the testing process. For simplicity, only two main types of testing process are used here. [7]They are *‘Unit Testing’* which performs test cases on each individual component and *‘Integration Testing’* which performs the testing on integration of the individual components with the whole system.

## 6.1 Unit Testing

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | Check login functionality | | |
| **Case** | Valid email and password | Invalid email or password | Blank email or password |
| **Pre-condition** | None | None | None |
| **Test Data** | myothetun@gmail.com  abcdefgh | myothettun@gmail.com  password | myothettun@gmail.com  Null (nothing is typed in) |
| **Expected** | Navigate to Home page | Show message “Invalid” | Show message to fill the text box |
| **Results** | Navigate to Home page | Show “Invalid Login” | Show “Please fill in this field” |
| **Pass(Y/N)** | Y | Y | Y |

Table 6. : Check Login

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | Check passenger form | | |
| **Case** | Blank in any user control | Invalid phone | Save information |
| **Pre-condition** | Valid Login | Valid Login | Valid Login |
| **Test Data** | Nothing is typed in | Abcd1234 | Myo Thet,Tun,M,05/11/2019,  M1234567J,Burma,45671234,myothettun@gmail.com |
| **Expected** | Show “Please fill in this field” | Show message “Invalid” | Show message data is successfully inserted |
| **Results** | Show “Please fill in this field” | Show invalid phone number | Show “Data is successfully inserted!” |
| **Pass(Y/N)** | Y | Y | Y |

Table 6. : Check Passenger

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | Check flight form | | |
| **Case** | Blank in any user control | Same source and destination | Save information |
| **Pre-condition** | Valid Login | Valid Login | Valid Login |
| **Test Data** | Nothing is typed in | City A, City A | 16, Airline One, City A, City B, 03/06/2020, 10:00, 12:00, 70, 20 |
| **Expected** | Show “Please fill in this field” | Show message “source/destination cannot be the same” | Show message data is successfully inserted |
| **Results** | Show “Please fill in this field” | Show message “source/destination cannot be the same” | Show “Data is successfully inserted!” |
| **Pass(Y/N)** | Y | Y | Y |

Table 6. : Check Fight

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | Check reservation form | | |
| **Case** | Blank in any user control | Same source and destination | Save information |
| **Pre-condition** | Registered Passenger, Flight | Registered Passenger, Flight | Registered Passenger, Flight |
| **Test Data** | Nothing is typed in | City A, City A | 14, Airline One, City A, City D, 03/06/2020, 08:00,12:00, 100,120 |
| **Expected** | Show “Please fill in this field” | Show message “source/destination cannot be the same” | Show message data is successfully inserted |
| **Results** | Show “Please fill in this field” | Show message “source/destination cannot be the same” | Show “Data is successfully inserted!” |
| **Pass(Y/N)** | Y | Y | Y |

Table 6. : Check Reservation

## 6.2 Integration Testing

|  |  |  |
| --- | --- | --- |
| **Scenario** | Check update and delete record via data grid | |
| **Case** | Update in passenger form | Delete in passenger form |
| **Pre-condition** | Old record must exist | Old record must exist |
| **Test Data** | Edit ‘first name field’ in record no. 9 to ‘Myo T’ | Delete record no. 2 in passenger form |
| **Expected** | Show “Successfully updated” | Show “Successfully deleted” |
| **Results** | Show “Successfully updated” | Show “Successfully deleted” |
| **Pass(Y/N)** | Y | Y |

Table 6. : Check Update and Delete Record

|  |  |  |
| --- | --- | --- |
| **Scenario** | Check view table | |
| **Case** | Select flight table | Select passenger table |
| **Pre-condition** | Table must exist | Table must exist |
| **Test Data** | Select Flight and press “View Table” | Select Passenger and press “View Table” |
| **Expected** | Show flight table in a data grid | Show passenger table in a data grid |
| **Results** | Show flight table in a data grid | Show passenger table in a data grid |
| **Pass(Y/N)** | Y | Y |

Table 6. : Check View Table

# Chapter 7. Conclusion

In summary, the system is called “*Air Ticket Reservation Management System*” **(ATRMS)**. It is created as a web-based air ticket reservation system. Since the very beginning, the author started everything from scratch and every step is documented here as detail as possible. And this is also the first project for the author to create a whole web-based system from scratch.

This document includes defining the problem scope, gathering requirements specifications, techniques and technologies used in the system, various diagrams for system design, implementation of the system, test cases and results for the system testing. Lastly, the conclusion, strengths and weaknesses, and future enhancements of the system.

## 7.1 Strengths and Weaknesses of the System

**Strengths**

Be able to perform fully functional *CRUD* system as create, retrieve, update, and delete options for all records are available in every table. Use data grid style when viewing all records in the table to have overall structure of the whole table. Update and delete buttons are available in each record when viewing the table in data grid. Pressing update button will bring back to the corresponding form filled up with the requested data record. Thus, this makes it ready to update the record.

Messages and notifications are displayed accordingly when and where necessary making the system user-friendly and interactive. Overall GUI design is optimised to be user-friendly and positions of the form components and controls, such as textboxes, buttons, combo boxes, etc. are designed to be consistence and convenience for users in the whole system.

**Weaknesses**

Even though much effort has been made throughout the process of developing the system, due to limited amount of time and limited knowledge on the resources used, there are some weak points found in the system as follows.

Change password option should be included for existing users. When updating a record, user must select or filled with data in all data fields again to be able to update the record. Otherwise, the field would be regarded as null and would request the user to fill in. Ticket cancellation is not included in the current version.

## 7.2 Future Enhancements

In the future version of the system, all the weaknesses mentioned above will be improved. A registered user should be able to change the password, should not need to select or fill all the data field again when updating. The system should be improved in terms of security and in terms of GUI design as well. The overall system design should be improved as well for better performance. Moreover, many new features, such as ticket cancellation and user levels, are expected to be included in the next version of the system.

***\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\****

# References

[1] [schäferhof](https://websitesetup.org/about-and-contact/)f, N. (2019), Bootstrap Tutorial-How to setup and use bootstrap(Step-by-Step) [online], Website Setup, available at: https://websitesetup.org/bootstrap-tutorial-for-beginners/ [accessed 28 May 2020]

[2] Unknown (Unknown), MySQL: About MySQL [online], MySQL, available at: https://www.mysql.com/about/ [accessed 28 May 2020]

[3] Unknown (Unknown), What is MySQL [online], Site Ground, available at: https://www.siteground.com/tutorials/php-mysql/mysql/ [accessed 28 May 2020]

[4] Ganesan, P. (2017), What is XAMPP? And how to install XAMPP on local computer? [online], WPBLogX, available at: https://www.wpblogx.com/what-is-xampp/ [accessed 28 May 2020]

[5] MDN Contributors (2020), What is JavaScript? [online], Mozilla MDN web docs, available at: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript [accessed 28 May 2020]

[6] Unknown (Unknown), What is PHP? [online], Free servers, available at: http://www.freeservers.com/WebHosting101/WhatIsPHP.html [accessed 28 May 2020]

[7] [Unknown](https://websitesetup.org/about-and-contact/) (Unknown), What is Software Testing? Introduction, Definition, Basics & Types [online], Guru99, available at: https://www.guru99.com/software-testing-introduction-importance.html [accessed 28 May 2020]