**SOFTWARE ENGINEERING**

**PROJECT TITLE:**

**AIRLINE RESERVATION SYSTEM**

**Moukhik Misra – 19BCE2190**



**Abstract**

Airline Reservation System is a computerized system used to store and retrieve

information and make transactions related to airplane travelling. The project is aimed at

exposing the importance of Airline Reservation Systems. It aims towards enhancing the relationship between customers and airline agencies through the use of ARSs, and thereby making it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations.

This software has two parts. First is user part and the administrator part. Administrator is the back end and User part is used as a front end. Administrator is used by airline authority. It will allow the customers to access database and allow new customers to sign up for online access.

The system allows the airline passenger to search for flights that are available between two cities, for example, “Delhi” and “Mumbai” for a particular departure and arrival dates. The system displays all the flight’s details such as flight no, name, price and duration of journey etc.

After search, the system display list of available flights and allows customer to

choose a particular flight. Then the system checks for the availability of seats on the

flight. If the seats are available, then the system allows the passenger to book a seat.

Otherwise, it asks the user to choose another flight.

To book a flight the system asks the customer to enter his details such as name, flight, travel areas. Then it checks the validity of card and book the flight and update the airline database and user database. The system also allows the customer to cancel his/her reservation, if any problem occurs.

The main purpose of this software is to reduce the manual errors involved in the

airline reservation process and make it convenient for the customers to book the flights as

when they require such that they can utilize this software to make reservations, modify reservations or cancel a particular reservation.

**I. Introduction**

**1.1 Background**

Flight booking segment of travel sector have some procedural complexities that require time and money. It has to deal with various concerns related to fares, schedules, rules, taxes, cancellations, and regional limits. Therefore, an effective flight reservation system is needed to reduce these limitations.

A rapid increase in the number of airline companies has resulted in cut-throat competition in the airline industry. That is why these companies are looking for a solution that helps them in generating impressive Return of investments. That is why online flight booking engine is the best option to meet these dynamic demands of these companies.

Flight Booking System shoots up the sales of an airline company and gives a competitive edge. The primary reason for choosing flight reservation system is to increase the number of bookings of the flight ticket.

**1.2 Objective**

The main objective of this project is to create a user friendly and functionality rich Airline Booking System which will:

* Create a database with appropriate tables to store vital customer data and store information of flight tickets, while avoiding redundancy and inconsistency of data.
* Creating an online airline reservation system using a backend database connection using MySQL.
* To make the Online airline reservation system efficient and user-friendly.
* Creating a secure Airline reservation system providing correct user-access levels.
* Using different development tools such as html, php, CSS, JavaScript, Bootstrap with interconnection between each of these tools.
* Provide an option that will enable a passenger(s) to cancel his/her flight due to whim or impulse.
* Provide an option where passengers can view price chart.
* Provide an information about flights.
* Provide facility to book and view the status of flight.
* User will also be able to view all the flights of a particular day irrespective of time and destination.
* We have also created a payment portal for booking of flights.

**1.3 Motivation**

Our main motivation for the project comes from the fact that the project topic is interesting. We are working on a project that has real-world relevance and we are creating our solution for it. We are working with extremely interesting development tools such as PHP, HTML, CSS, JavaScript, Bootstrap and SQL. Applying our knowledge and learning new techniques while creating our project gives us great motivation.

**II. USER REQUIREMENT SPECIFICATION**

**1. INTRODUCTION**

**1.1 PURPOSE**

The purpose of this document is to build an online system to manage flights and passengers to ease the flight management.

**1.2 DOCUMENT CONVENTIONS**

This document uses the following conventions.

* DB : Database
* ER: Entity Relationship

**1.3 INTENDED AUDIENCE AND READING SUGGESTIONS**

This project is a prototype for an airport management system which is a flight booking system and it is restricted within the college premises. This has been implemented under the guidance of Software Engineering professor Prof. Meenakshi S P . This project is useful for the airport management team and as well as to the passengers. The project is also useful for anyone who is interested in understanding the foundations of creating an airline reservation system.

**1.4 PROJECT SCOPE**

The purpose of the online flight management system is to ease flight management and to create a convenient and easy-to-use application for passengers, trying to buy airline tickets. The system is based on a relational database with its flight management and reservation functions. We will have a database supporting hundreds of major cities around the world as well as thousands of flights by various airline companies. Above all, we hope to provide a comfortable user experience along with the best pricing available.

**2. OVERALL DESCRIPTION**

**2.1 PRODUCT PERSPECTIVE**

**Description of information required:**

1. Passenger- It contains username, first name, middle name, lastname, email, mobile number, gender, DOB, password of user .These all informations about the user are stored in our DB when the user first sign up.

2. Book- It contains Book\_id, time, date, flight no., username, classtype, paid. These all information about the booking are stored in our DB when the user books the flight.

3. Class- It contains flight no., name, capacity, price.

4. Airplane- It contains airplane Id, airplane type, company. These are the information about flight which user sees while booking a seat.

5. Flight- It contains flight No., Airplane ID, Departure, Delay time, Arrival of flight.

6. Airport- It contains airport code, airport name ,city ,state, country. It contains the details of the airport which can be added

**2.2 PRODUCT FEATURES**

The major features of airline reservation system are as follows:

* An efficient method for booking flights.
* Easy retrieval of information, users can search for flights very easily as well as choose to and from location and dates of departure.
* User friendly screens to enter the data.
* User sign-in module which personalizes the information.
* Users can view the history of their purchases easily.
* Admin control to add/update/delete flight details.
* A portable and light weight application which can be further enhanced and worked upon.
* Web enabled and Fast finding of information requested.

**2.3 USER CLASS and CHARACTERISTICS**

Users of the system should be able to check the details of his booking between two given cities with the given date/time of travel from the database. A route from city A to city B is a sequence of \flights from A to B. The system will support two types of user privileges, Customer, and Employee. Customers will have access to customer functions, and the employees will have access to both customer and flight management functions. The customer should be able to do the following functions:

* **User sign in module**
* User can create their own account and sign in
* **Make a new reservation**
  + One-way
  + Round-Trip
  + Flexible Date/time
  + Confirmation
* **Cancel an existing reservation**
* **Make their payments**
* **Check their history**

The Employee should have following management functionalities:

* **CUSTOMER FUNCTIONS.**
  + Get to know if tickets are available
  + View all flights
  + To know if seats are available for a given flight.
* **ADMINISTRATIVE**
  + Add/Delete a flight
  + Update fare for flights.
  + Update the details of the flights

Each flight has a limited number of available seats. There are a number of flights which depart from or arrive at different cities on different dates and time.

**2.4 OPERATING ENVIRONMENT**

Operating environment for the airline management system is as listed below.

* distributed database
* client/server system
* Operating system: Windows.
* database: MySQL database
* platform: PHP,HTML,CSS

**2.5 DESIGN and IMPLEMENTATION CONSTRAINTS**

1. The global schema, fragmentation schema, and allocation schema.
2. SQL commands for above queries/applications
3. How the response for application 1 and 2 will be generated. Assuming these are global queries. Explain how various fragments will be combined to do so.
4. Implement the database at least using a centralized database management system.

**2.6 ASSUMPTION DEPENDENCIES**

Let us assume that this is a airline reservation system and it is used in the following application:

* For customers to book flight tickets from source to destination.
* Customers have created their own profiles.
* Customers can view the details of the flight that they will be booking.
* Customers are assumed to have made the necessary payment to confirm their booking.
* Administrators can add/delete flights and update the details of existing flights.

**3. SYSTEM FEATURES**

* **DESCRIPTION and PRIORITY**

The airline reservation system maintains information on flights, classes of seats, prices, and bookings. Of course, this project has a high priority because it is very difficult to travel across countries without prior reservations.

* **STIMULUS/RESPONSE SEQUENCES**
  + Search for Airline Flights for two Travel cities
  + Displays a detailed list of available flights and make a “Reservation” or Book a ticket on a particular flight.
  + Cancel an existing Reservation from the Shopping Cart before Payment.
* **FUNCTIONAL REQUIREMENTS**

Other system features include:

**DATABASE MANAGEMENT:**

Efficient management of the database to protect the data that is present in the database so as to ensure correct functioning of the project.

**4. EXTERNAL INTERFACE REQUIREMENTS**

**4.1 USER INTERFACES**

* Front-end software: HTML, CSS, Javascript, Bootstrap
* Back-end software: MySQL

**4.2 HARDWARE INTERFACES**

* Windows.
* A browser which supports CSS, HTML & Javascript.

**4.3 SOFTWARE INTERFACES**

Following are the software used for the flight management online application.

* **Operating System**: We have chosen Windows operating system for its best support and user-friendliness.
* **Database**: To save the flight records, passengers records, we have chosen MySQL database.
* **HTML, CSS, Javascript:** We are using these tools to create the frontend of our website.

**4.4 COMMUNICATION INTERFACES**

This project supports all types of web browsers. We are using simple electronic forms for the reservation forms, ticket booking etc.

**5. NONFUNCTIONAL REQUIREMENTS**

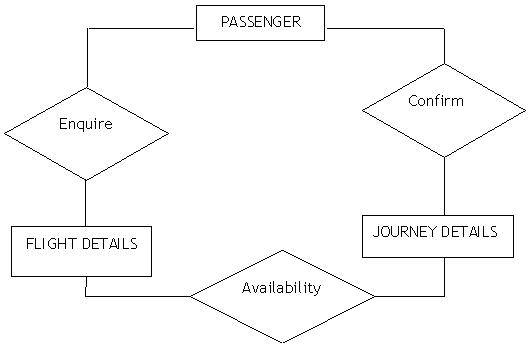
**5.1 PERFORMANCE REQUIREMENTS**

The steps involved to perform the implementation of airline database are as listed below.

**A) E-R DIAGRAM**

The E-R Diagram constitutes a technique for representing the logical structure of a database in a pictorial manner. This analysis is then used to organize data as a relation, normalizing relation and finally obtaining a relation database.

* **ENTITIES:**Which specify distinct real-world items in an application.
* **PROPERTIES/ATTRIBUTES:** Which specify properties of an entity and relationships.
* **RELATIONSHIPS:** Which connect entities and represent meaningful dependencies between them.



*Basic Default ER diagram of airline database*

**B) NORMALIZATION:**

The basic objective of normalization is to reduce redundancy which means that information is to be stored only once. Storing information several times leads to wastage of storage space and increase in the total size of the data stored.

If a database is not properly designed it can give rise to modification anomalies. Modification anomalies arise when data is added to, changed or deleted from a database table. Similarly, in traditional databases as well as improperly designed relational databases, data redundancy can be a problem. These can be eliminated by normalizing a database.

Normalization is the process of breaking down a table into smaller tables. So that each table deals with a single theme. There are three different kinds of modifications of anomalies and formulated the first, second and third normal forms (3NF) is considered sufficient for most practical purposes. It should be considered only after a thorough analysis and complete understanding of its implications.

**5.2 SAFETY REQUIREMENTS**

If there is extensive damage to a wide portion of the database due to catastrophic failure, such as a disk crash, the recovery involves locking of transactions in the database so that we are capable of rolling back from any catastrophic failure.

**5.3 SECURITY REQUIREMENTS**

Security systems need database storage just like many other applications. However, the special requirements of the security market mean that vendors must choose their database partner carefully.

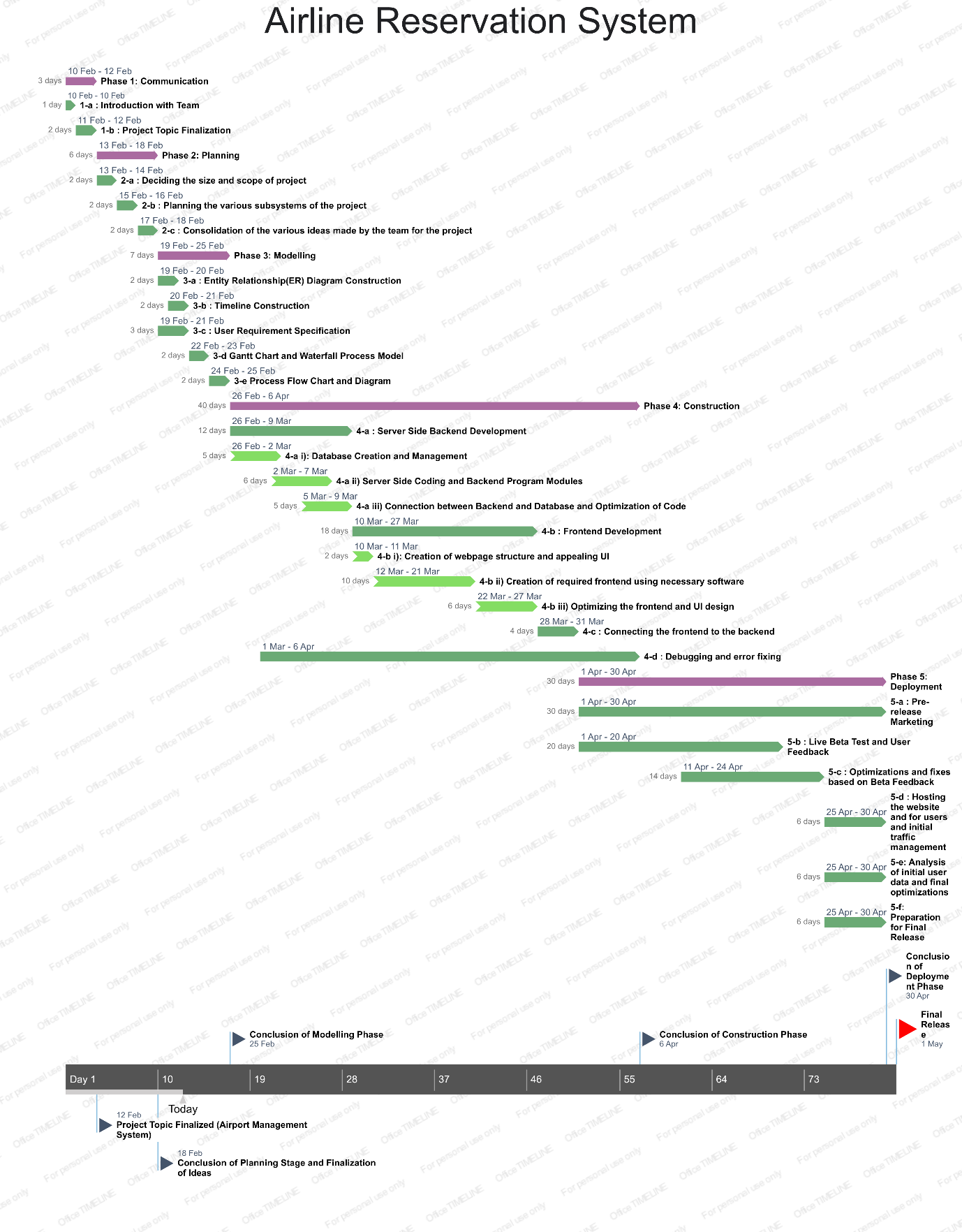
**5.4 SOFTWARE QUALITY ATTRIBUTES**

* **AVAILABILITY:** The flight should be available on the specified date and specified time as many customers are doing advance reservations.
* **CORRECTNESS:** The flight should reach start from correct start terminal and should reach the correct destination.
* **MAINTAINABILITY:** The administrators and flight in chargers should maintain correct schedules of flights.
* **USABILITY:** The flight schedules should satisfy a maximum number of customer needs.

**PLANNING CHART**

**Gantt Chart**

**Purple: Main Phases , Dark Green: Procedures, Light Green: Sub-Tasks**



**System Modelling**

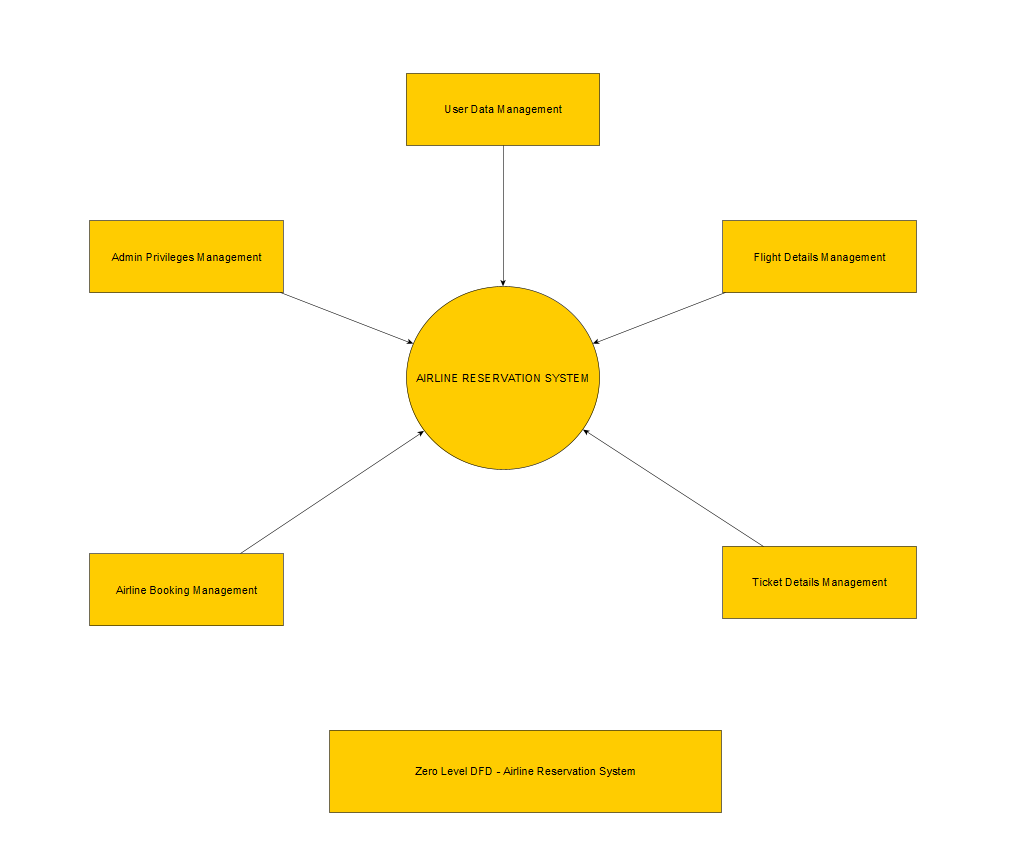
**1. Data Flow Diagram (DFD)**

DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system. The visual representation makes it a good communication tool between User and System designer. Structure of DFD allows starting from a broad overview and expand it to a hierarchy of detailed diagrams. DFD has often been used due to the following reasons:

* Logical information flow of the system
* Determination of physical system construction requirements
* Simplicity of notation
* Establishment of manual and automated systems requirements

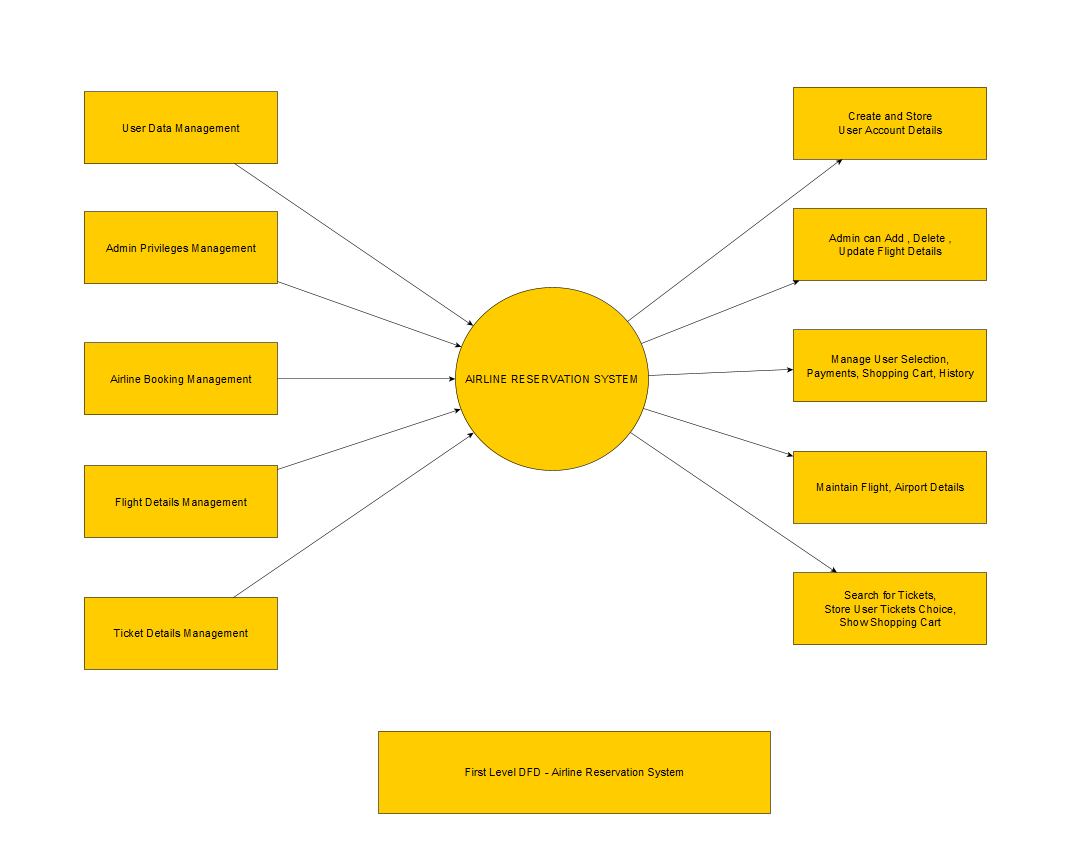
**I. 0 – level DFD of Airline Reservation System**

* Main aspects of the Airline Reservation System ae shown in the 0 level DFD



**II. 1st Level DFD**

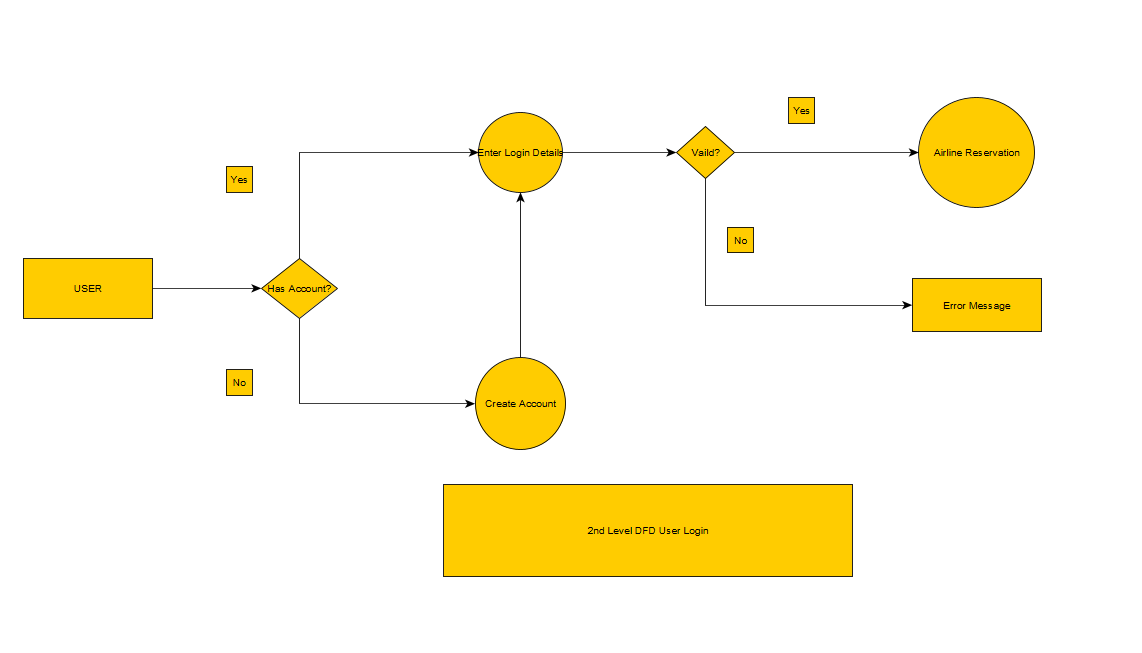
* First Level DFD of Airline Reservation System describes the main functions of its various components.



**III. 2nd Level DFD**

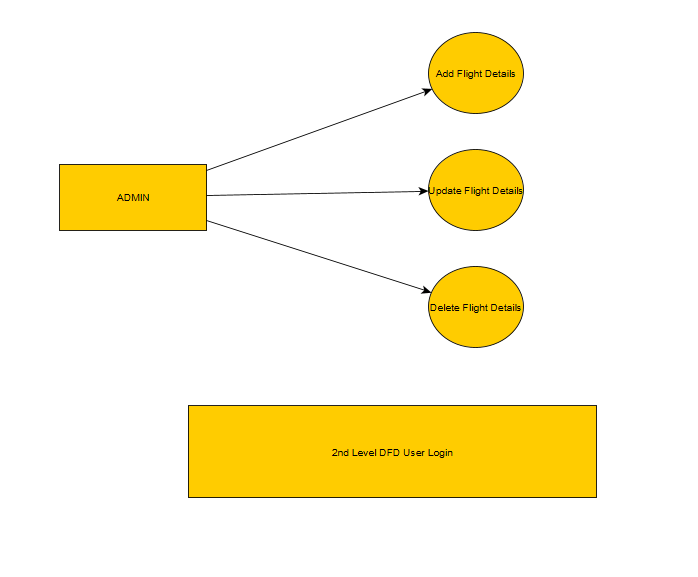
**1. User Login**

* Details the steps required for the user to create an account or login to the account



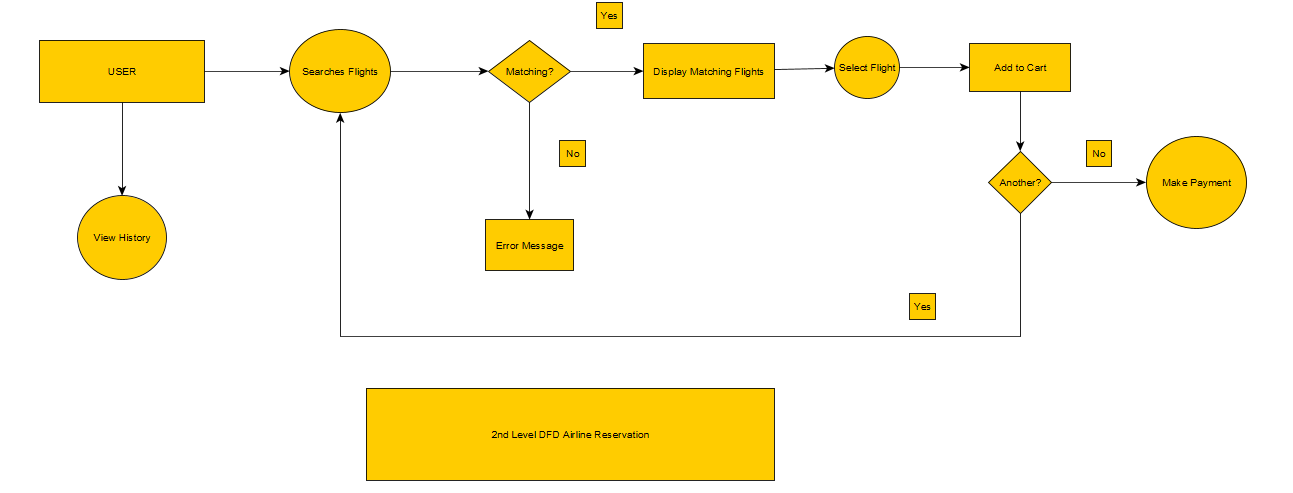
**2. Admin**

* Admin Functionalities are represented here



**3. Airline Reservation**

* Describes procedure for booking a ticket

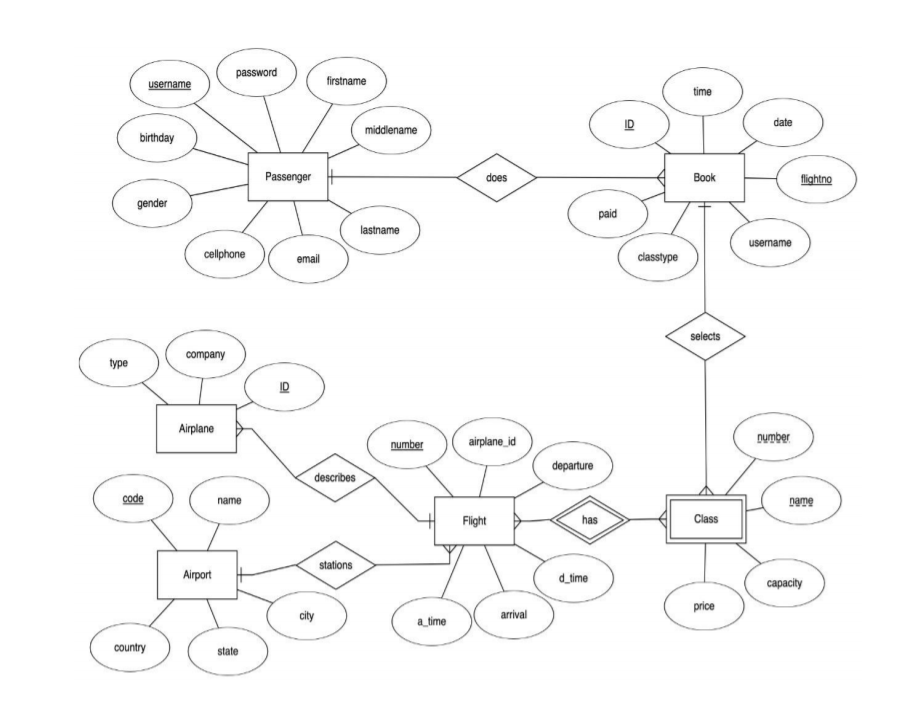


**2. Context Modelling with ER Diagram**

A context model defines how context data are structured and maintained (It plays a key role in supporting efficient context management). It aims to produce a formal or semi-formal description of the context information that is present in a context-aware system. In other words, the context is the surrounding element for the system, and a model provides the mathematical interface and a behavioural description of the surrounding environment.

ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database. In other words, ER diagrams help to explain the logical structure of databases. ER diagrams are created based on three basic concepts: entities, attributes and relationships. ER Diagrams contain different symbols that use rectangles to represent entities, ovals to define attributes and diamond shapes to represent relationships. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.

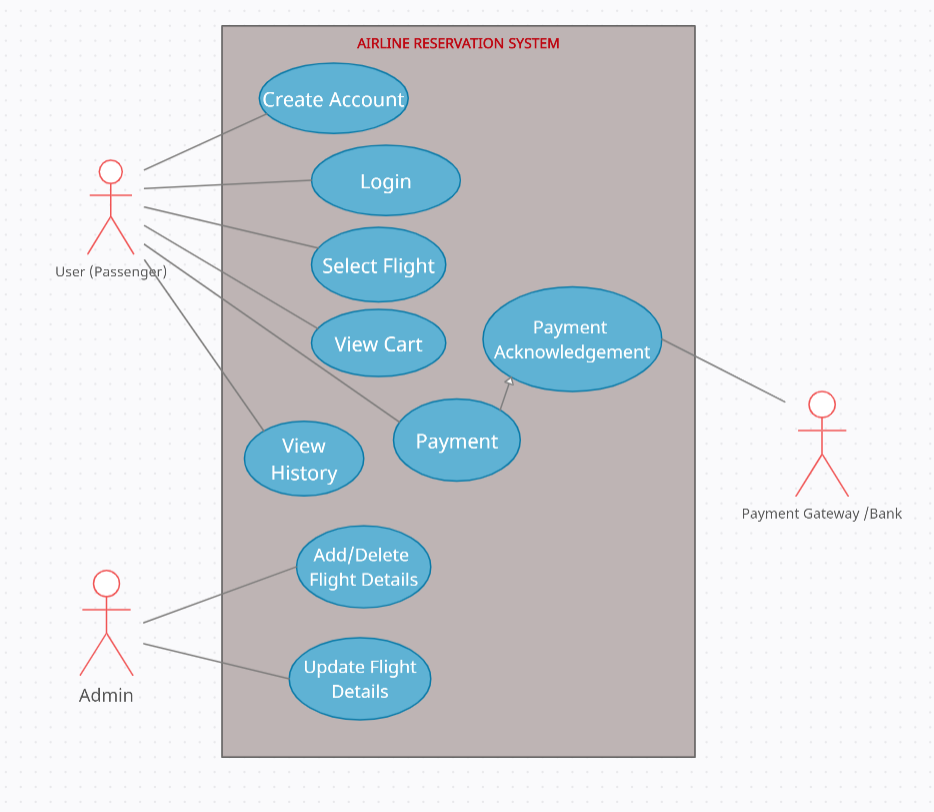
The ER diagram of the Airline Reservation System shows the various sections of information that is being stored and the relations between the various components of the system. We can see the passenger section and the various information that is being stored there. We can see the relation between passenger, booking, flight, airport, airplane , class and so on.



**3. Use Case Diagram**

A UML Use Case Diagram is the primary form of system/software requirements for a new software program underdeveloped. Use cases specify the expected behavior (what), and not the exact method of making it happen (how). Use cases once specified can be denoted both textual and visual representation (i.e. use case diagram). A key concept of use case modeling is that it helps us design a system from the end user's perspective. It is an effective technique for communicating system behavior in the user's terms by specifying all externally visible system behavior.

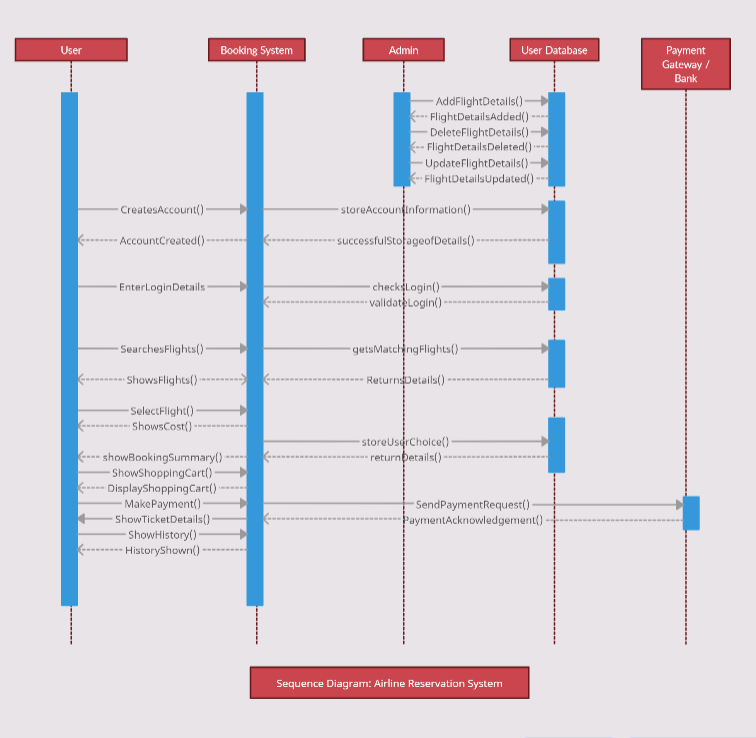
Here we represent the Airline Reservation System as a Use Case Diagram. In our Airline Reservation System, we can see that the user is capable of creating their own account and login to the same. Users can search the flights to their necessary destinations from necessary airport and can select their seats as well. Users also can make payments to book the ticket as well. Users can view their shopping cart as well as their booking history. The application also has administrator functions where the admin can add or delete flights and can update flight details such as capacity, price and destination.



**4. Sequence Diagram**

A sequence diagram simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function.

The sequence diagram given shows the sequence of functions of the Airline Reservation System. In the Sequence Diagram, user creates an account and logs in to the service. Then the user searches flight details and the system shows the necessary flights. User selects the flight required and then can see his shopping cart. User then makes the payment. After the user makes the payment, he can see his previous bookings.

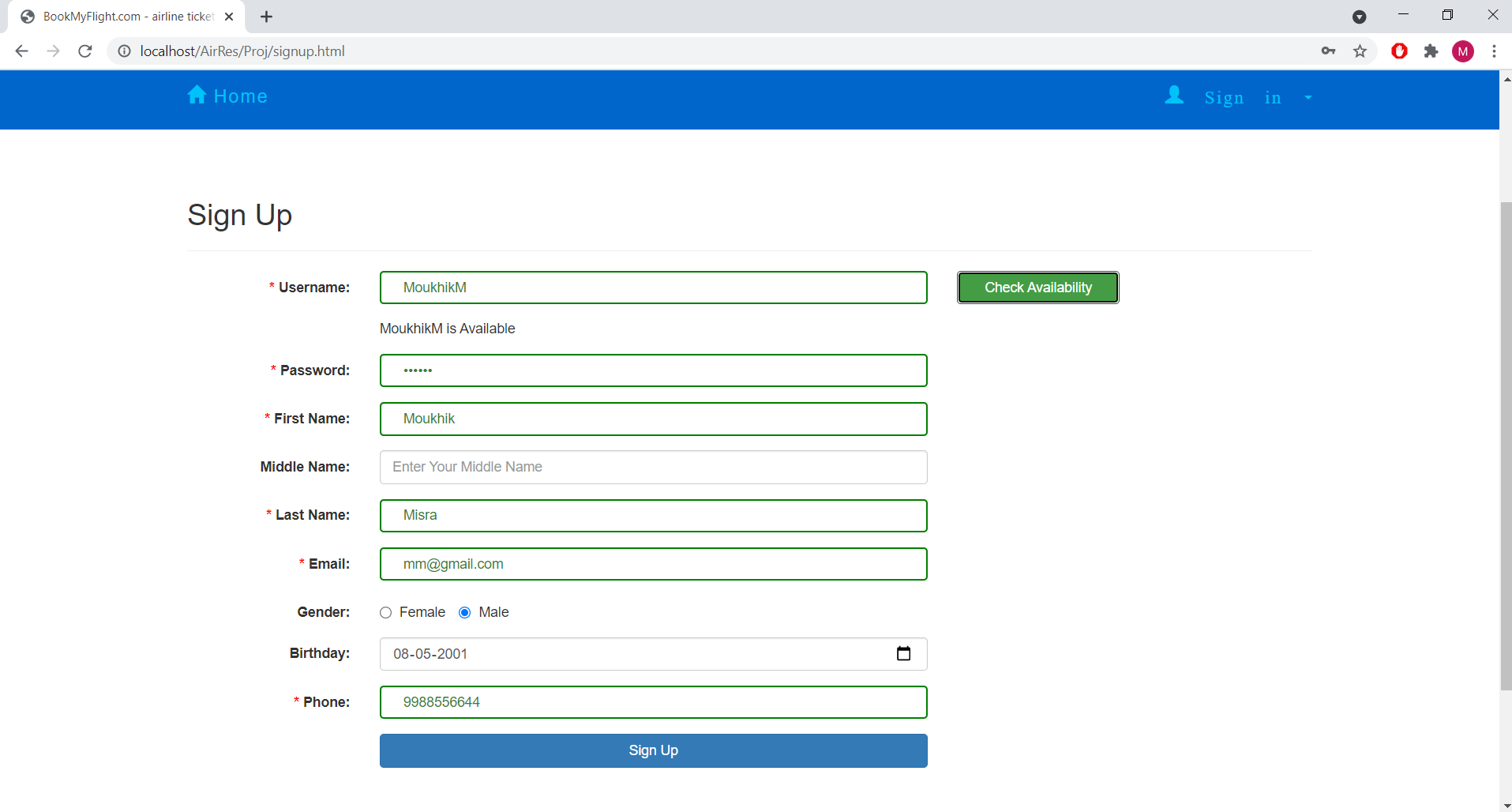


**Testing**

**1) Account Creation Module**

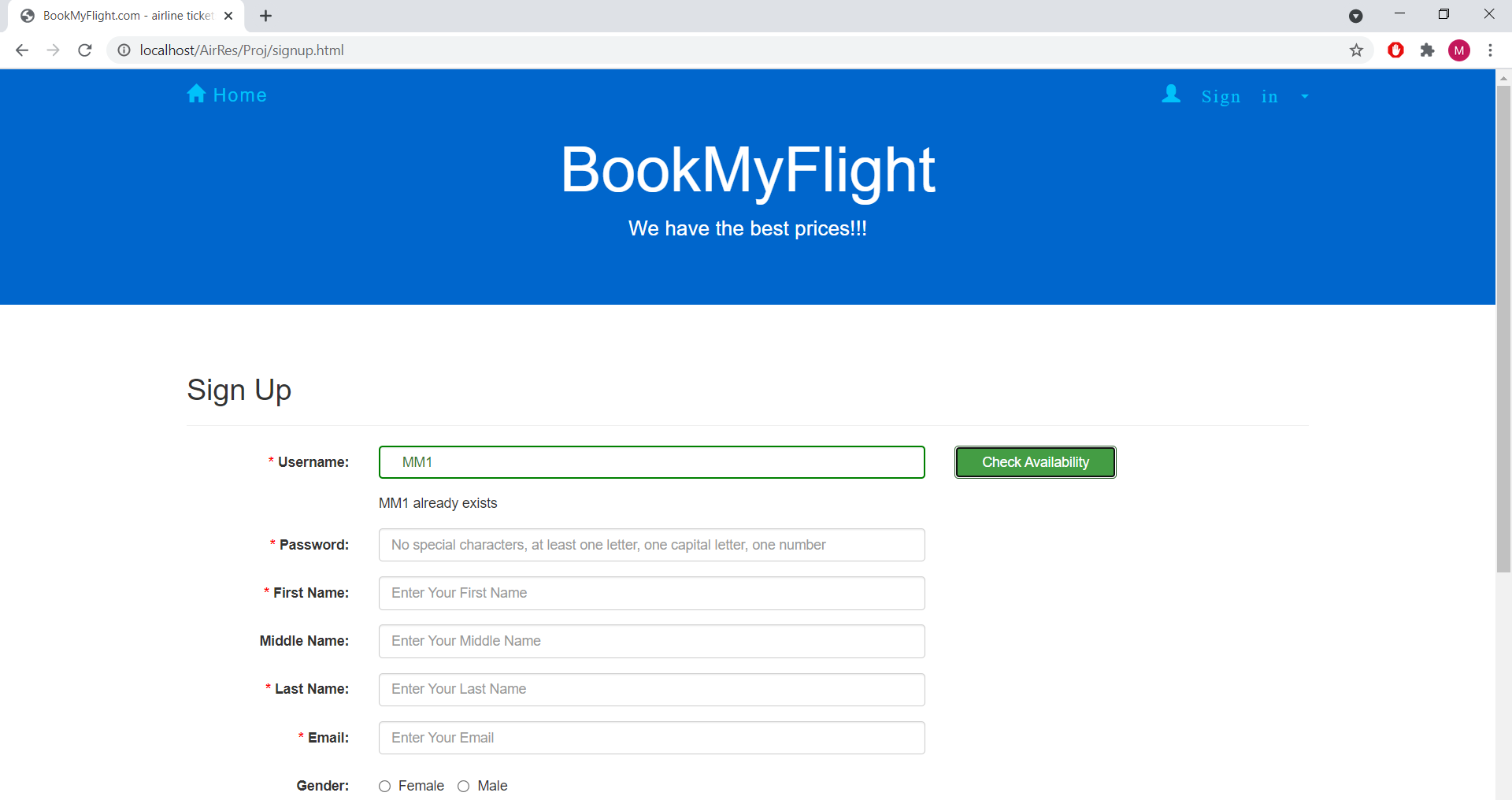
|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | TC\_001 | **Test Case Description** | | Test the Account Creation Module Functionality | | | | | |
| **Created By** | | Moukhik Misra | **Reviewed By** | | Moukhik Misra, Mohit Suhasaria, Eshana Dubey | | **Version** | | 1.0 | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | | Review Module Functionality for Account Creation |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | | Moukhik Misra | **Date Tested** | | May 9, 2021 | | **Test Case (Pass/Fail/Not Executed)** | | Pass | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | |
| 1 | Access to Chrome Browser | | |  | 1 | Usernames: MoukhikM, MM1 | | | | |
| 2 | Connection to Localhost | | |  | 2 | Passwords: Abc123, asd1 | | | | |
| 3 | Working Database Connection | | |  | 3 | First Name: Moukhik | | | | |
| 4 | Internet Connection | | |  | 4 | Last Name: Misra | | | | |
|  |  |  |  |  | 5 | email: mm@gmail.com, mmmmm | | | | |
|  |  |  |  |  | 6 | Birthday: 08-05-2001 | | | | |
|  |  |  |  |  | 7 | Phone: 9988556644 | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Test Scenario 1** | | Verify the functionality and working of account creation by supplying correct information | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Start the website from localhost | | Site should open | | As Expected | | | Pass | | |  |
| 2 | Enter desirable username | | Username can be entered | | As Expected | | | Pass | | |  |
| 3 | Check username availability | | Username should be unique | | As Expected | | | Pass | | |  |
| 4 | Enter desired password | | Password can be entered | | As Expected | | | Pass | | |  |
| 5 | Check Password validity | | Password should be valid according to given rules | | As Expected | | | Pass | | |  |
| 6 | Enter First Name, Middle Name(optional) Last Name | | Name can be entered | | As Expected | | | Pass | | |  |
| 7 | Enter email id | | Email Id can be entered | | As Expected | | | Pass | | |  |
| 8 | Enter Gender | | Gender can be selected | | As Expected | | | Pass | | |  |
| 9 | Enter Birthday | | Birthday can be entered |  | As Expected | | | Pass | | |  |
| 10 | Enter Phone Number | | Phone Number can be entered | | As Expected | | | Pass | | |  |
| 11 | Press Sign Up | | Account should be created | | As Expected | | | Pass | | |  |

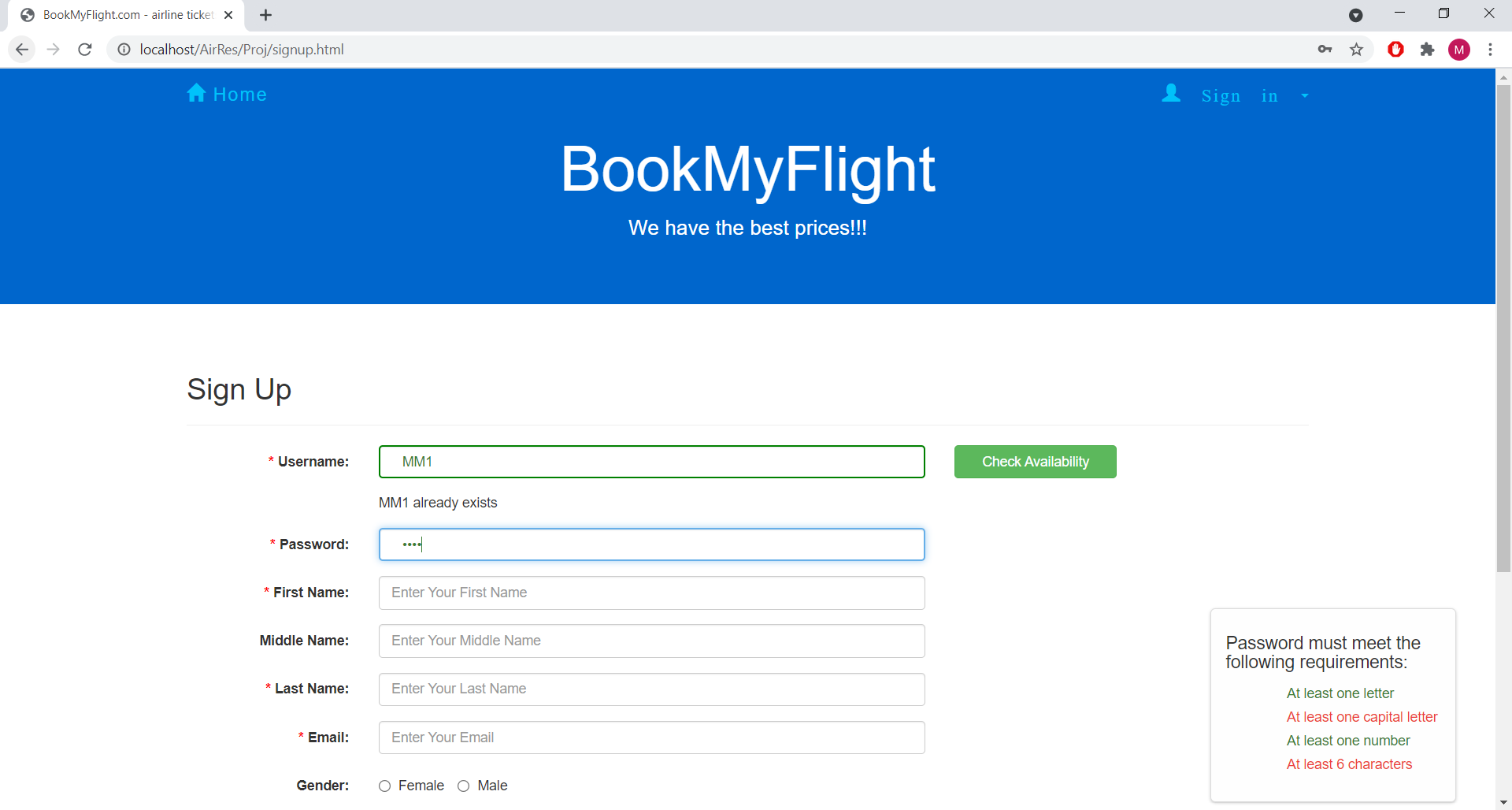
Result of Testing Account Creation with correct information:

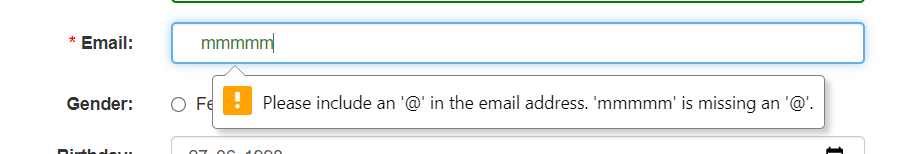


|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario 2** | | Verify the functionality and working of account creation by supplying wrong information | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Start the website from localhost | | Site should open | | As Expected | | | Pass | | |  |
| 2 | Enter username which has already been taken | | Username can be entered | | As Expected | | | Pass | | |  |
| 3 | Check username availability | | Username should not be unique | | As Expected | | | Pass | | |  |
| 4 | Enter available username | | Username is unique | | As Expected | | | Pass | | |  |
| 5 | Enter password which does not follow rules | | Password can be entered | | As Expected | | | Pass | | |  |
| 6 | Check Password validity | | Password is not valid | | As Expected | | | Pass | | |  |
| 7 | Enter valid Password | | Password is valid | | As Expected | | | Pass | | |  |
| 8 | Enter First Name, Middle Name(optional) Last Name | | Name can be entered | | As Expected | | | Pass | | |  |
| 9 | Enter invalid email id | | Email Id will not be accepted | | As Expected | | | Pass | | |  |
| 10 | Enter valid email id | | Email can be entered | | As Expected | | | Pass | | |  |
| 11 | Enter Gender | | Gender can be selected | | As Expected | | | Pass | | |  |
| 12 | Enter Birthday | | Birthday can be entered |  | As Expected | | | Pass | | |  |
| 13 | Enter Phone Number | | Phone Number can be entered | | As Expected | | | Pass | | |  |
| 14 | Press Sign Up | | Account should be created | | As Expected | | | Pass | | |  |

Result of Testing Account Creation with invalid data fields or wrong information :



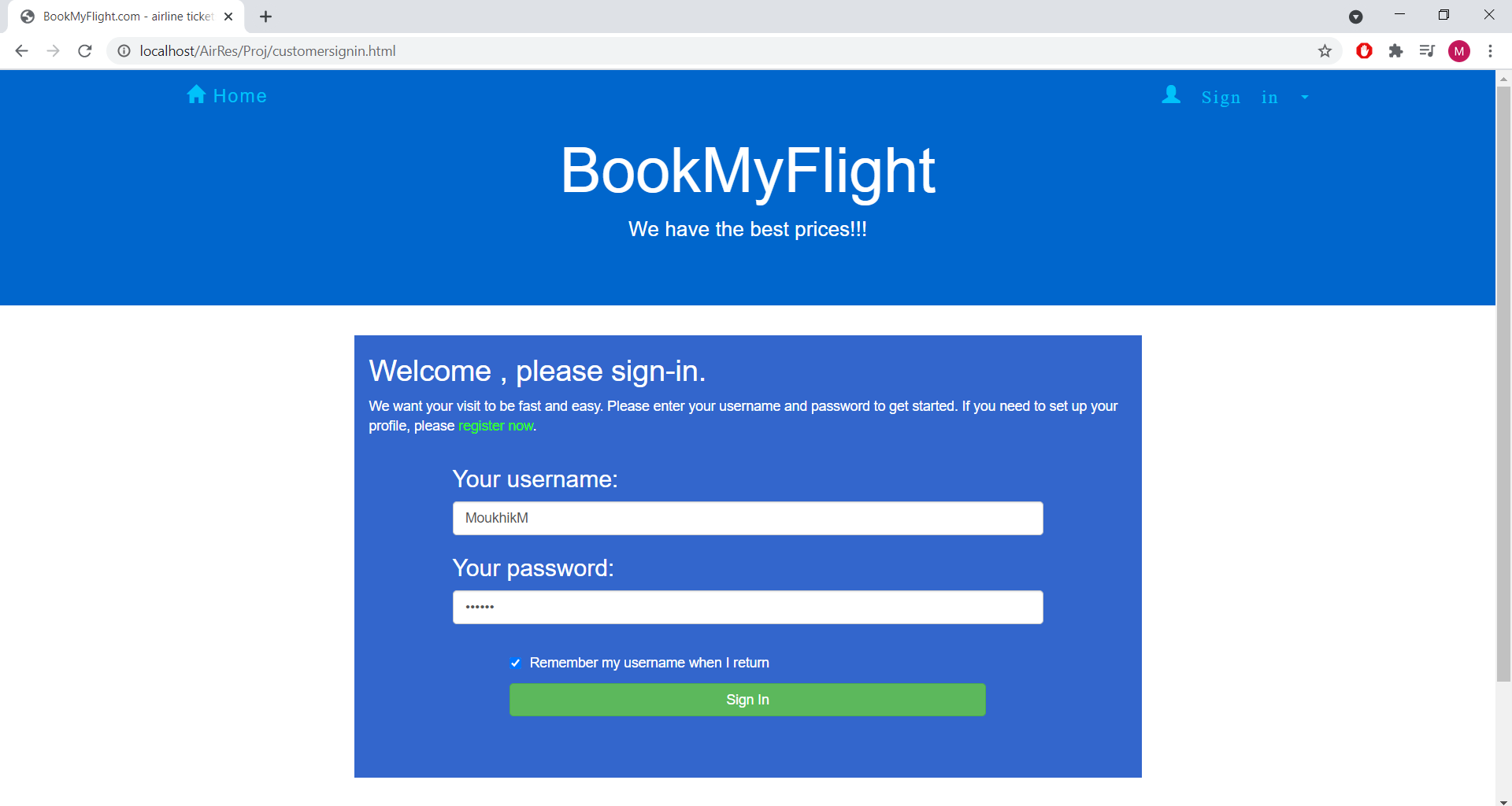


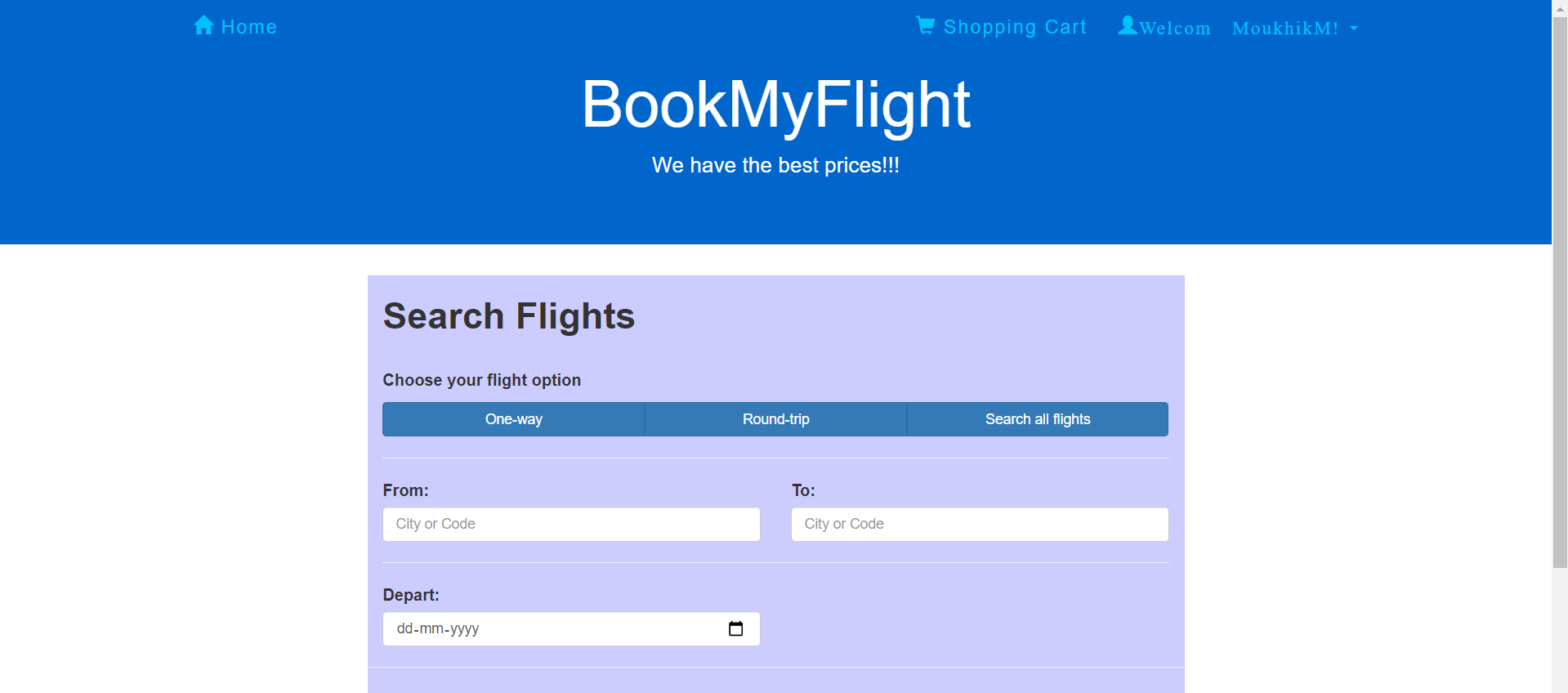


**2) Login Module:**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | TC\_002 | **Test Case Description** | | Test the Login Module Functionality | | | | | |
| **Created By** | | Moukhik Misra | **Reviewed By** | | Moukhik Misra, Mohit Suhasaria, Eshana Dubey | | **Version** | | 1.0 | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | | Review Module Functionality for Login |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | | Moukhik Misra | **Date Tested** | | May 9, 2021 | | **Test Case (Pass/Fail/Not Executed)** | | Pass | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | |
| 1 | Access to Chrome Browser | | |  | 1 | Usernames: MoukhikM (valid username), MS (invalid username) | | | | |
| 2 | Connection to Localhost | | |  | 2 | Passwords: Abc123 (valid password), asd1 (invalid password) | | | | |
| 3 | Working Database Connection | | |  |  |  |  |  |  |  |
| 4 | Internet Connection | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Test Scenario 1** | | Verify the functionality and working of login module by entering valid details | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Navigate to login page | | Login Page should open | | As Expected | | | Pass | | |  |
| 2 | Enter username | | Username can be entered | | As Expected | | | Pass | | |  |
| 3 | Enter password | | Password can be entered | | As Expected | | | Pass | | |  |
| 5 | Select Remember User option | | Option can be selected | | As Expected | | | Pass | | |  |
| 6 | Press Sign In | | User has signed in | | As Expected | | | Pass | | |  |

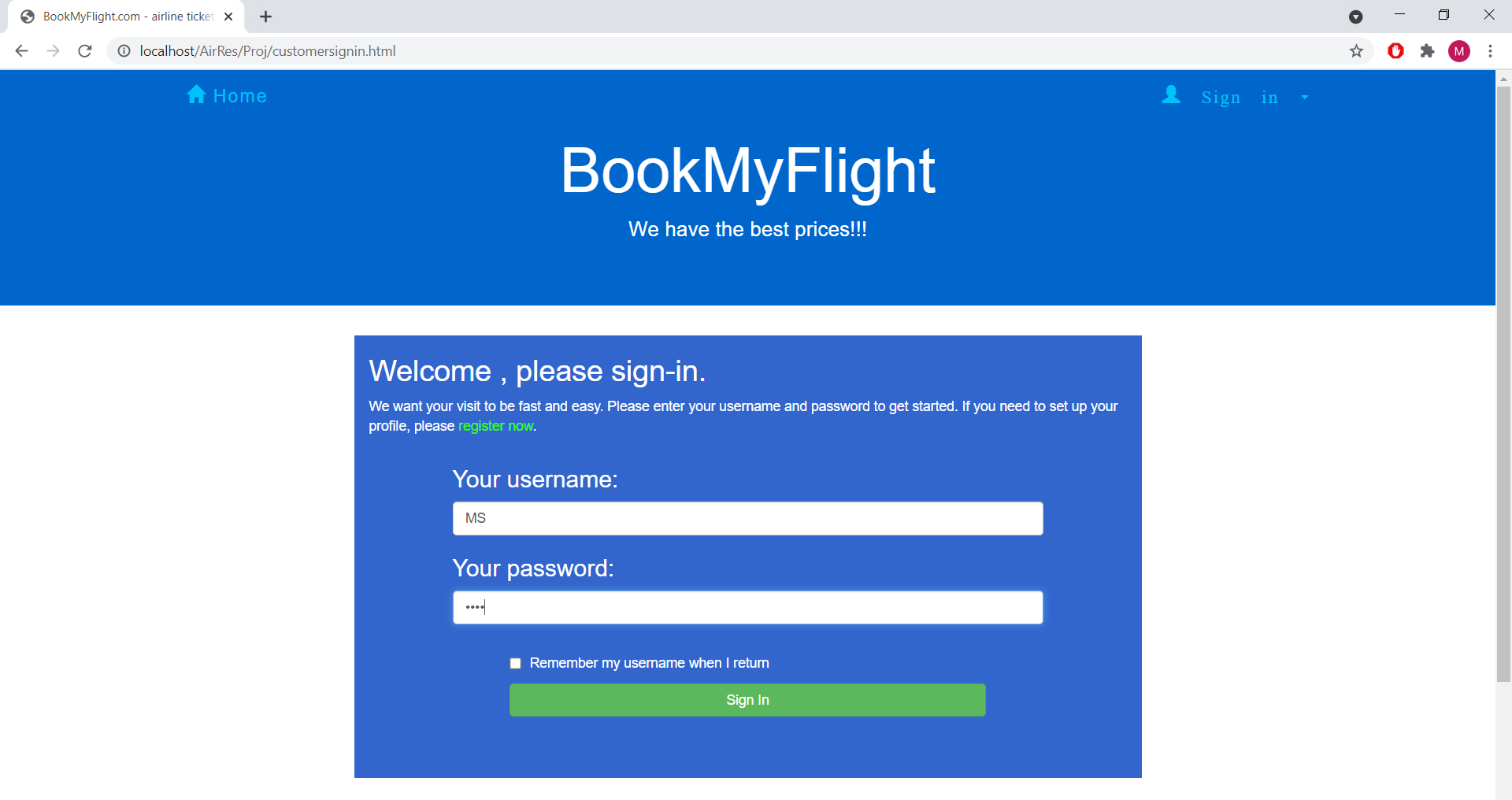
Result of Testing Login Module with valid details:

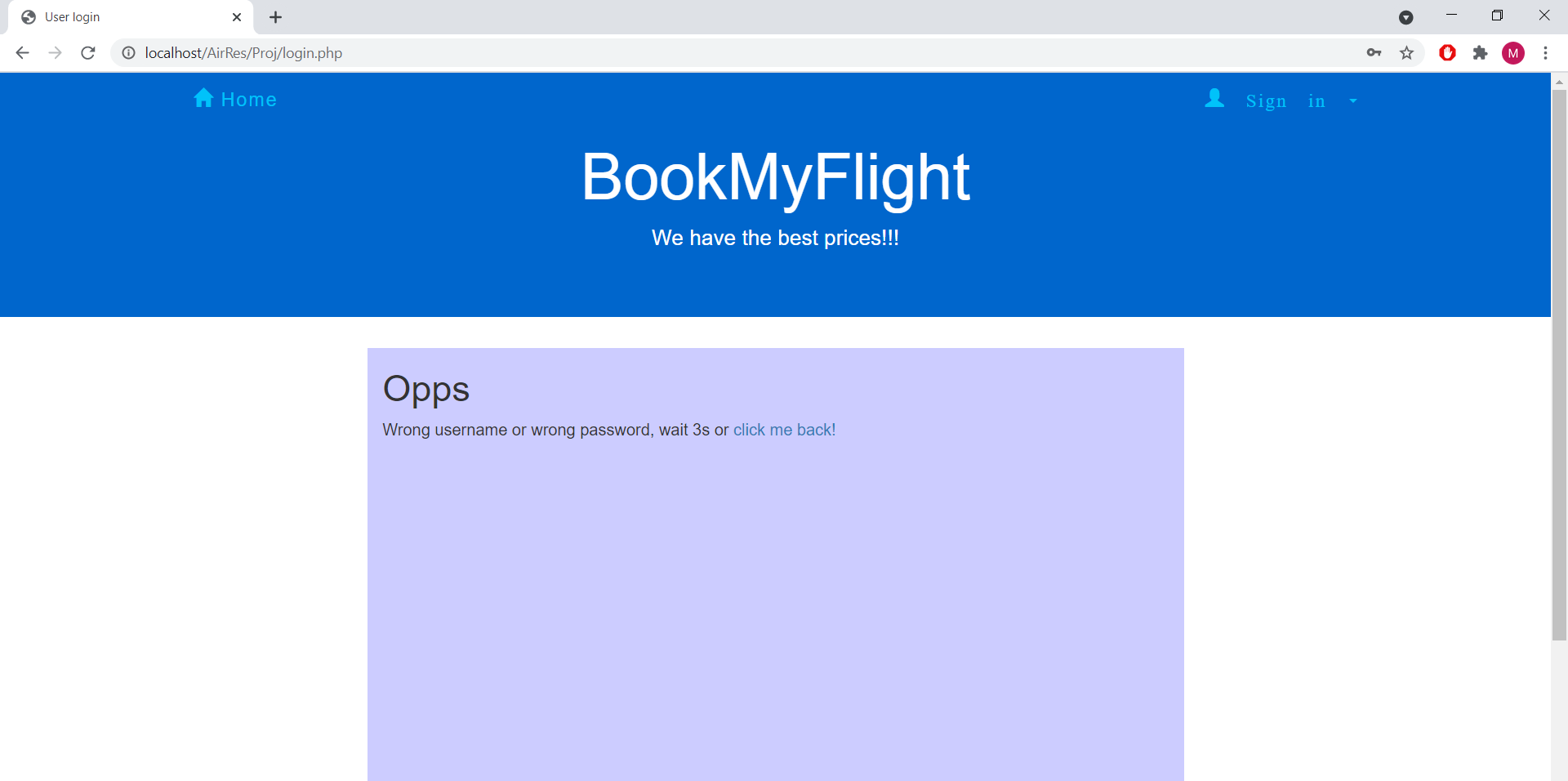




|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario 2** | | Verify the functionality and working of login module by entering invalid details | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Navigate to login page | | Login Page should open | | As Expected | | | Pass | | |  |
| 2 | Enter invalid username | | Username can be entered | | As Expected | | | Pass | | |  |
| 3 | Enter invalid password | | Password can be entered | | As Expected | | | Pass | | |  |
| 5 | Select Remember User option | | Option can be selected | | As Expected | | | Pass | | |  |
| 6 | Press Sign In | | User cannot sign in. Redirected back to login page after 3s. | | As Expected | | | Pass | | |  |

Result of Testing Login Module with invalid details:

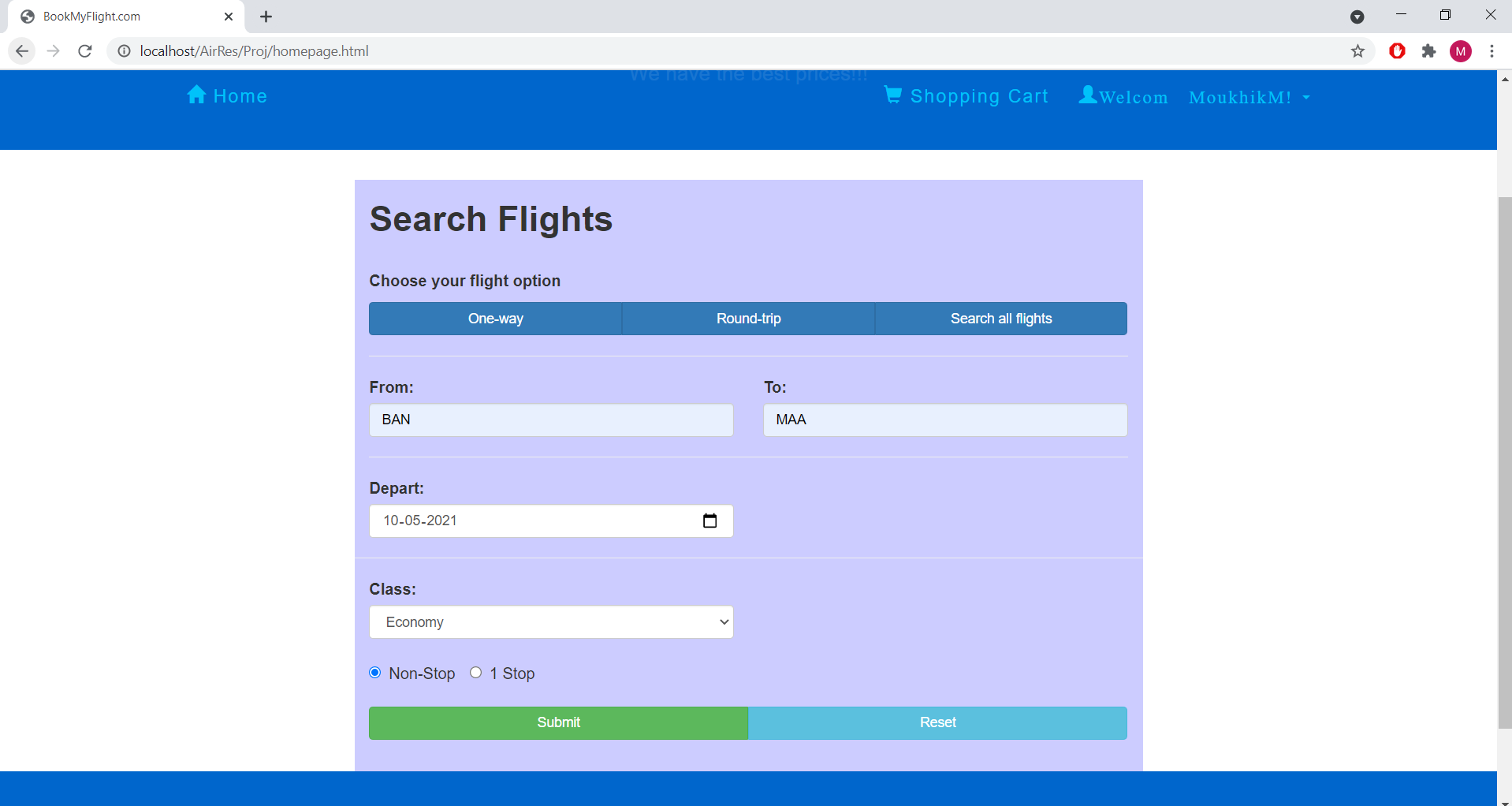


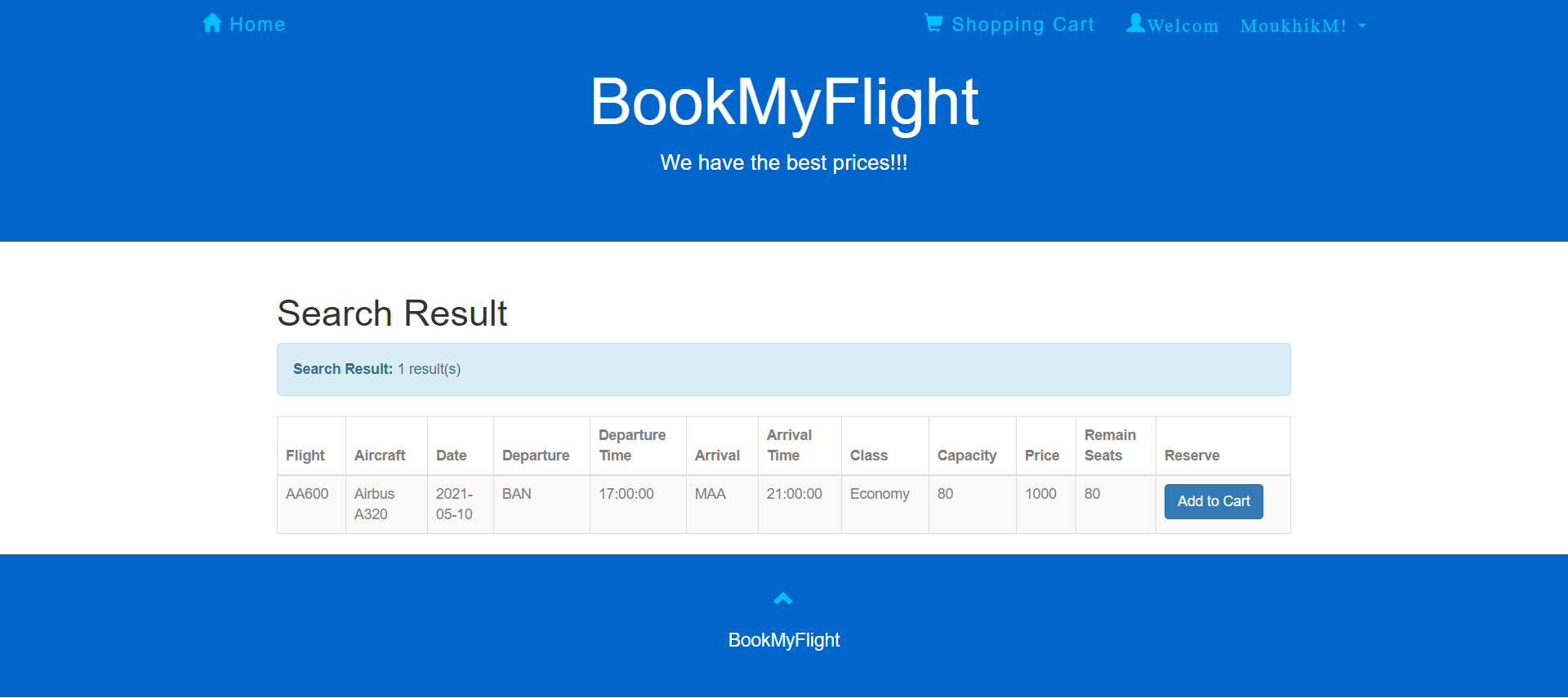


**3) Search Flights Module**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | TC\_003 | **Test Case Description** | | Test the Search Flight Module Functionality | | | | | |
| **Created By** | | Mohit Suhasaria | **Reviewed By** | | Mohit Suhasaria, Moukhik Misra, Eshana Dubey | | **Version** | | 1.0 | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | | Review Search Flight Module Functionality | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | | Mohit Suhasaria | **Date Tested** | | May 9, 2021 | | **Test Case (Pass/Fail/Not Executed)** | | Pass | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | |
| 1 | Access to Chrome Browser | | |  | 1 | Mode: Round Trip | | | | |
| 2 | Connection to Localhost | | |  | 2 | From: Bangalore (BAN) | | | | |
| 3 | Working Database Connection | | |  | 3 | To: Chennai (MAA) | | | | |
| 4 | Internet Connection | | |  | 4 | Depart: 10-05-2021 | | | | |
|  |  |  |  |  | 5 | Return: 12-05-2021 | | | | |
|  |  |  |  |  | 6 | Class: Economy | | | | |
|  |  |  |  |  | 7 | Stoppage: Non-Stop | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
| **Test Scenario 1** | | Verify the functionality and working of One-Way trip flight search | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Go to the search flight page via localhost | | Page should open | | As Expected | | | Pass | | |  |
| 2 | Enter Mode (One-Way, Round-trip) details | | One-Way can be selected | | As Expected | | | Pass | | |  |
| 3 | Enter From details | | Details can be entered | | As Expected | | | Pass | | |  |
| 4 | Enter To details | | Details can be entered | | As Expected | | | Pass | | |  |
| 5 | Enter Departure date | | Date can be selected | | As Expected | | | Pass | | |  |
| 6 | Enter Class details | | Class Details can be entered | | As Expected | | | Pass | | |  |
| 7 | Enter Stoppage details | | Stoppage Details can be entered | | As Expected | | | Pass | | |  |
| 8 | Click Submit | | Available flights for given parameters are shown | | As Expected | | | Pass | | |  |

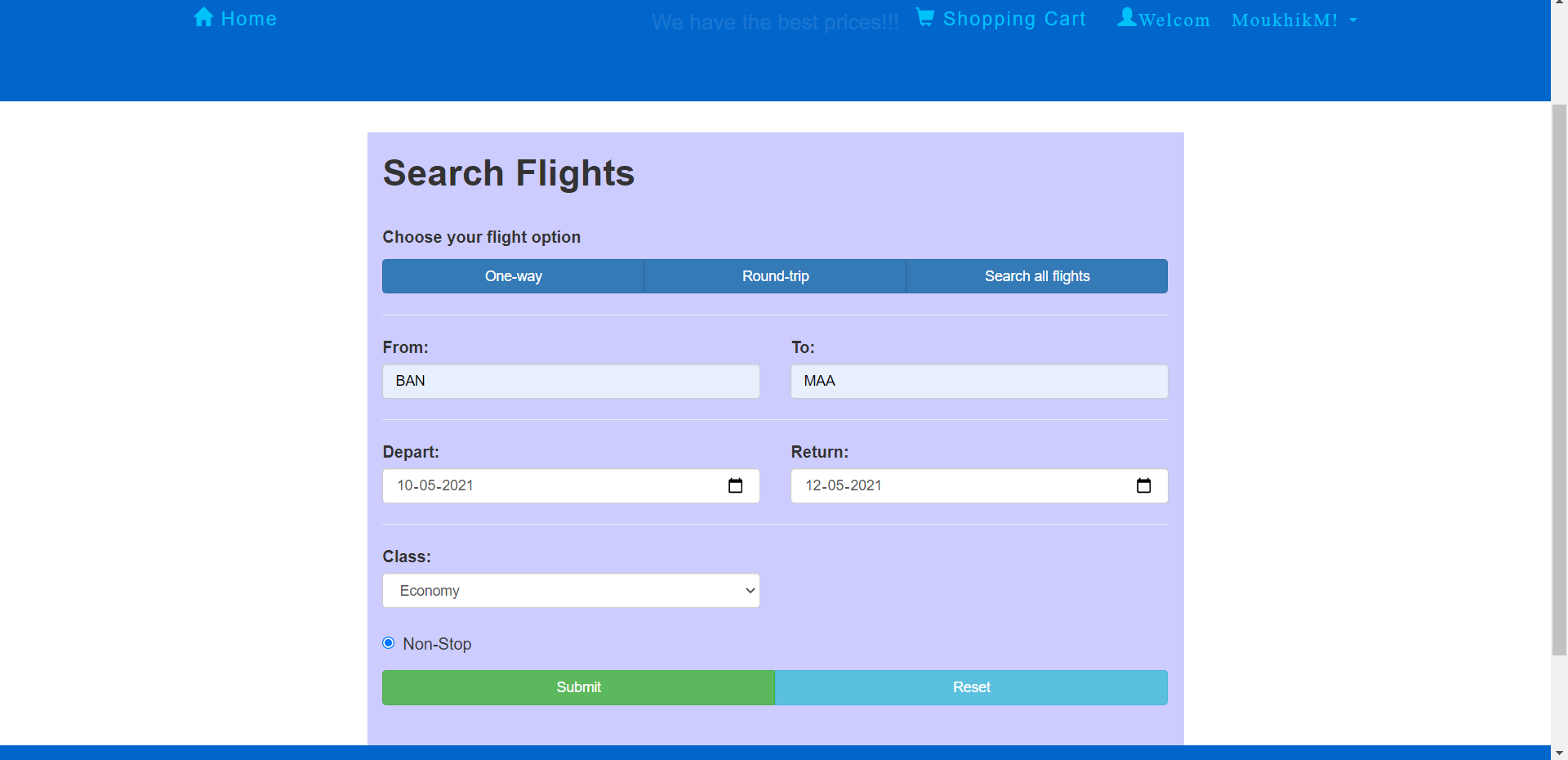
Results after entering One-Way Details:

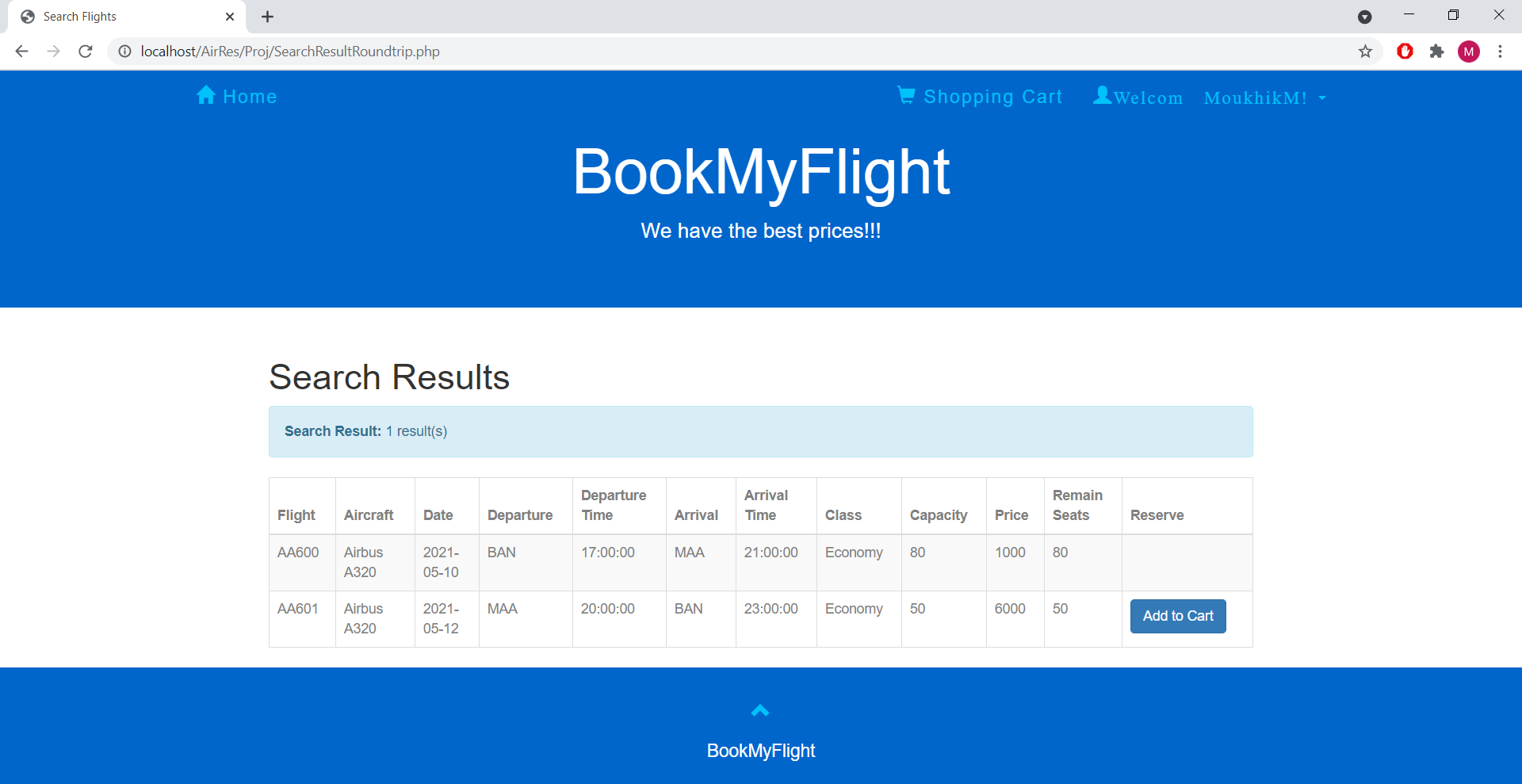




|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario 2** | | Verify the functionality and working of Round-trip flight search | | | |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Go to the search flight page via localhost | | Page should open | | As Expected | | | Pass | | |  |
| 2 | Enter Mode (One-Way, Round-trip) details | | Round trip can be entered | | As Expected | | | Pass | | |  |
| 3 | Enter From details | | Details can be entered | | As Expected | | | Pass | | |  |
| 4 | Enter To details | | Details can be entered | | As Expected | | | Pass | | |  |
| 5 | Enter Departure date | | Date can be selected | | As Expected | | | Pass | | |  |
| 6 | Enter Return date | | Date can be selected | | As Expected | | | Pass | | |  |
| 7 | Enter Class details | | Class Details can be entered | | As Expected | | | Pass | | |  |
| 8 | Enter Stoppage details | | Stoppage Details can be entered | | As Expected | | | Pass | | |  |
| 9 | Click Submit | | Available flights for given parameters are shown | | As Expected | | | Pass | | |  |

Results after entering Round Trip details:

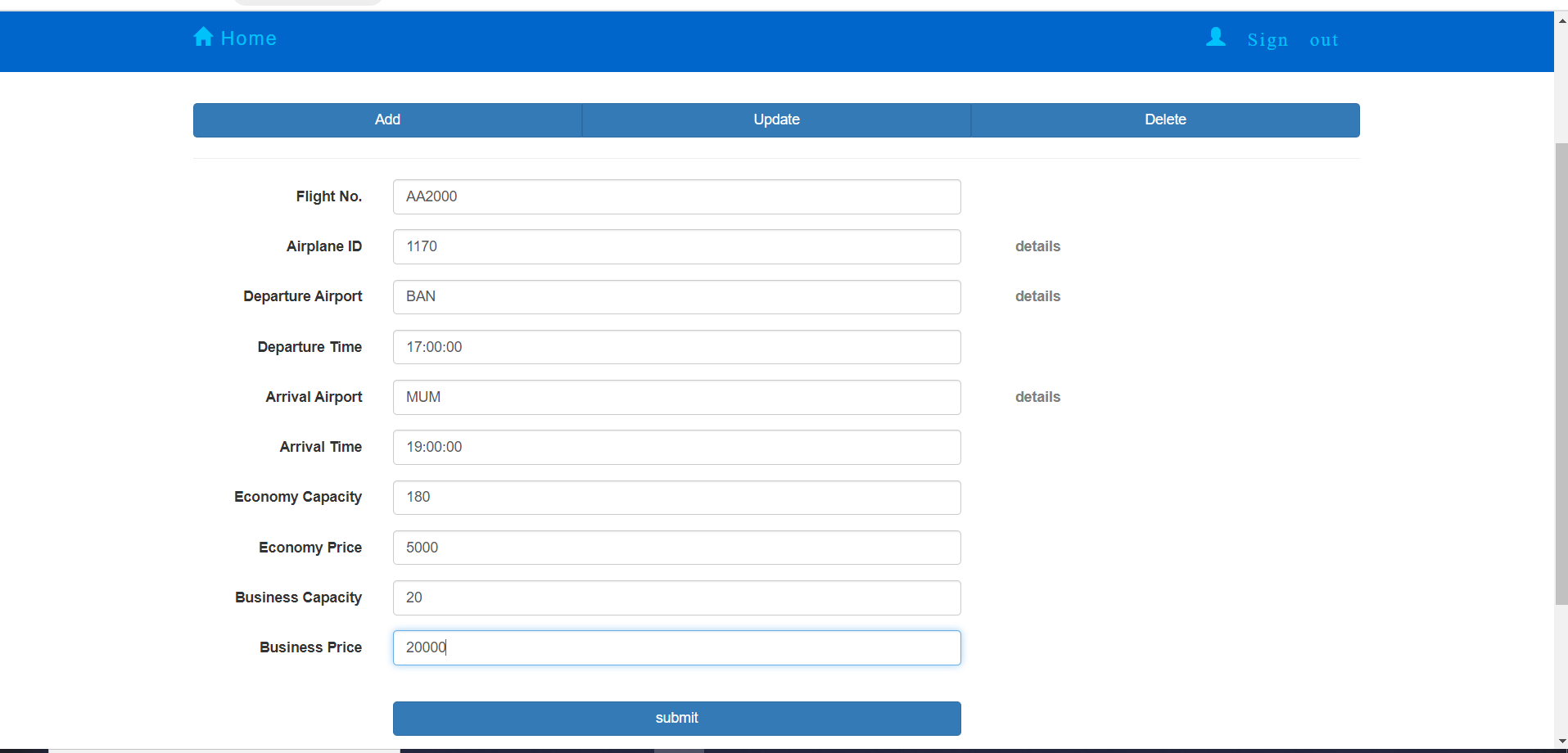




**4) Admin Module**

**Add Flight**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Case ID** | | TC\_004 | **Test Case Description** | | Test the Admin Module Functionality | | | | | |
| **Created By** | | Moukhik Misra | **Reviewed By** | | Moukhik Misra, Mohit Suhasaria, Eshana Dubey | | **Version** | | 1.0 | |
|  |  |  |  |  |  |  |  |  |  |  |
| **QA Tester’s Log** | | Review Admin Module Functionality |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Tester's Name** | | Moukhik Misra | **Date Tested** | | May 9, 2021 | | **Test Case (Pass/Fail/Not Executed)** | | Pass | |
|  |  |  |  |  |  |  |  |  |  |  |
| **S #** | **Prerequisites:** | | |  | **S #** | **Test Data** | | | | |
| 1 | Access to Chrome Browser | | |  | 1 | Flight No: AA2000 | | | | |
| 2 | Connection to Localhost | | |  | 2 | Airplane Id: 1170 | | | | |
| 3 | Working Database Connection | | |  | 3 | Departure Airport: BAN | | | | |
| 4 | Internet Connection | | |  | 4 | Departure Time: 17:00:00 | | | | |
|  |  |  |  |  | 5 | Arrival Airport : MUM | | | | |
|  |  |  |  |  | 6 | Arrival Time: 19:00:00 | | | | |
|  |  |  |  |  | 7 | Economy Capacity: 180 | | | | |
|  |  |  |  |  | 8 | Economy Price: 5000 | | | | |
|  |  |  |  |  | 9 | Business Capacity: 20 | | | | |
|  |  |  |  |  | 10 | Business Price: 20000 | | | | |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Test Scenario 1** | | Verify the functionality and working of Add Flight option in Admin Module | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Start the website from localhost | | Site should open | | As Expected | | | Pass | | |  |
| 2 | Enter Flight No | | Flight No. can be entered | | As Expected | | | Pass | | |  |
| 3 | Enter Aiplane Id | | Airplane Id can be entered | | As Expected | | | Pass | | |  |
| 4 | Enter Departure Details | | Departure details can be entered | | As Expected | | | Pass | | |  |
| 5 | Enter Arrival Details | | Arrival Details Can be Entered | | As Expected | | | Pass | | |  |
| 6 | Enter Economy details | | Economy Details can be entered | | As Expected | | | Pass | | |  |
| 7 | Enter Business details | | Business details can be entered | | As Expected | | | Pass | | |  |
| 8 | Press Submit | | Flight should be created | | As Expected | | | Pass | | |  |



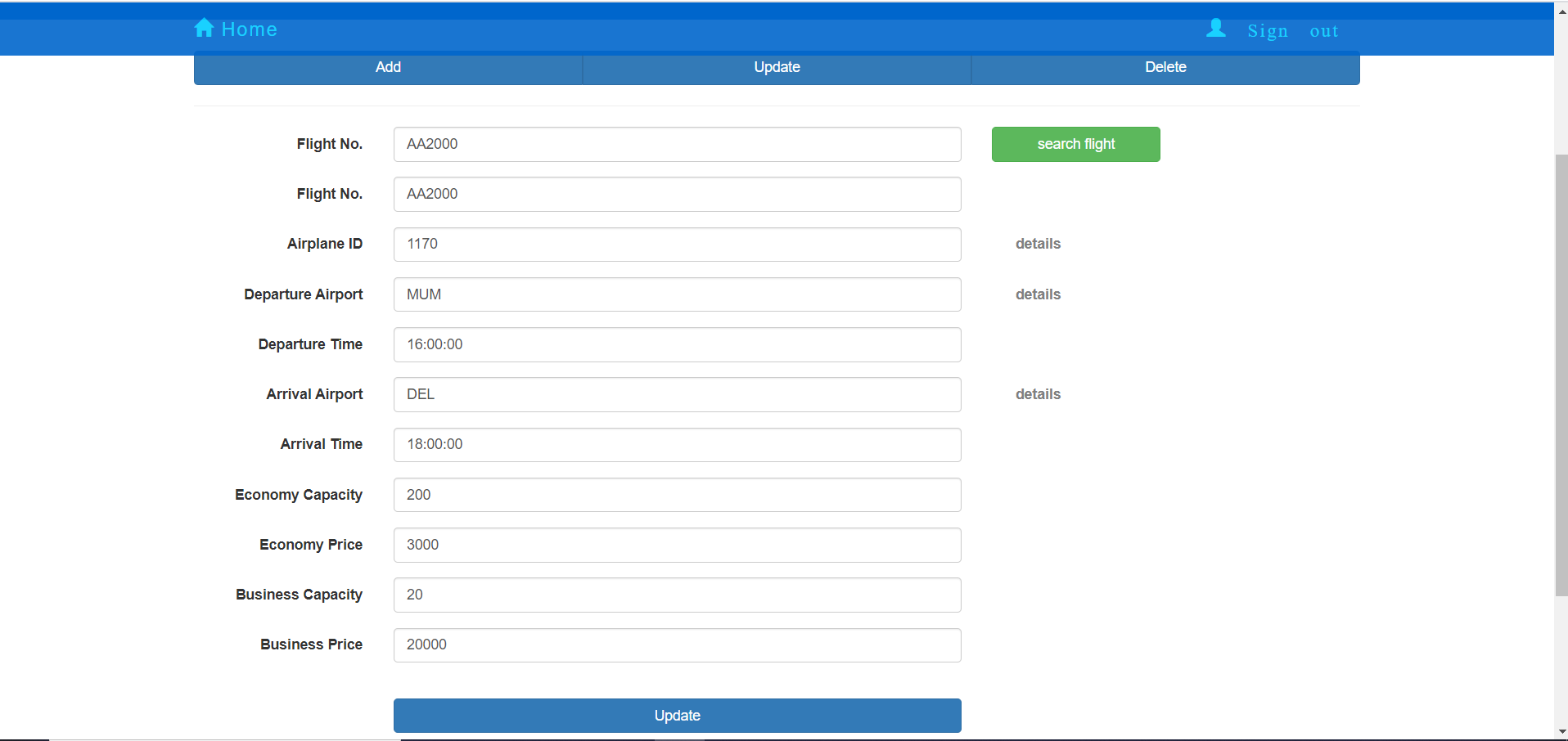


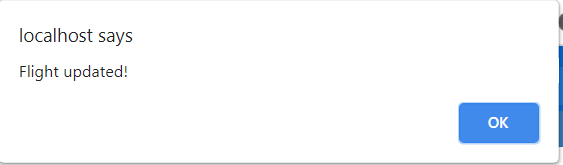
**Database:**



**Update Flight**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario 2** | | Verify the functionality and working of Update Flight Option in Admin Module | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Start the website from localhost | | Site should open | | As Expected | | | Pass | | |  |
| 2 | Enter Flight Number | | Flight Number Can be entered | | As Expected | | | Pass | | |  |
| 3 | Press Search Flight | | Details should autofill | | As Expected | | | Pass | | |  |
| 4 | Update Desired Details | | Details can be entered | | As Expected | | | Pass | | |  |
| 5 | Press Update | | Flight Details should update | | As Expected | | | Pass | | |  |



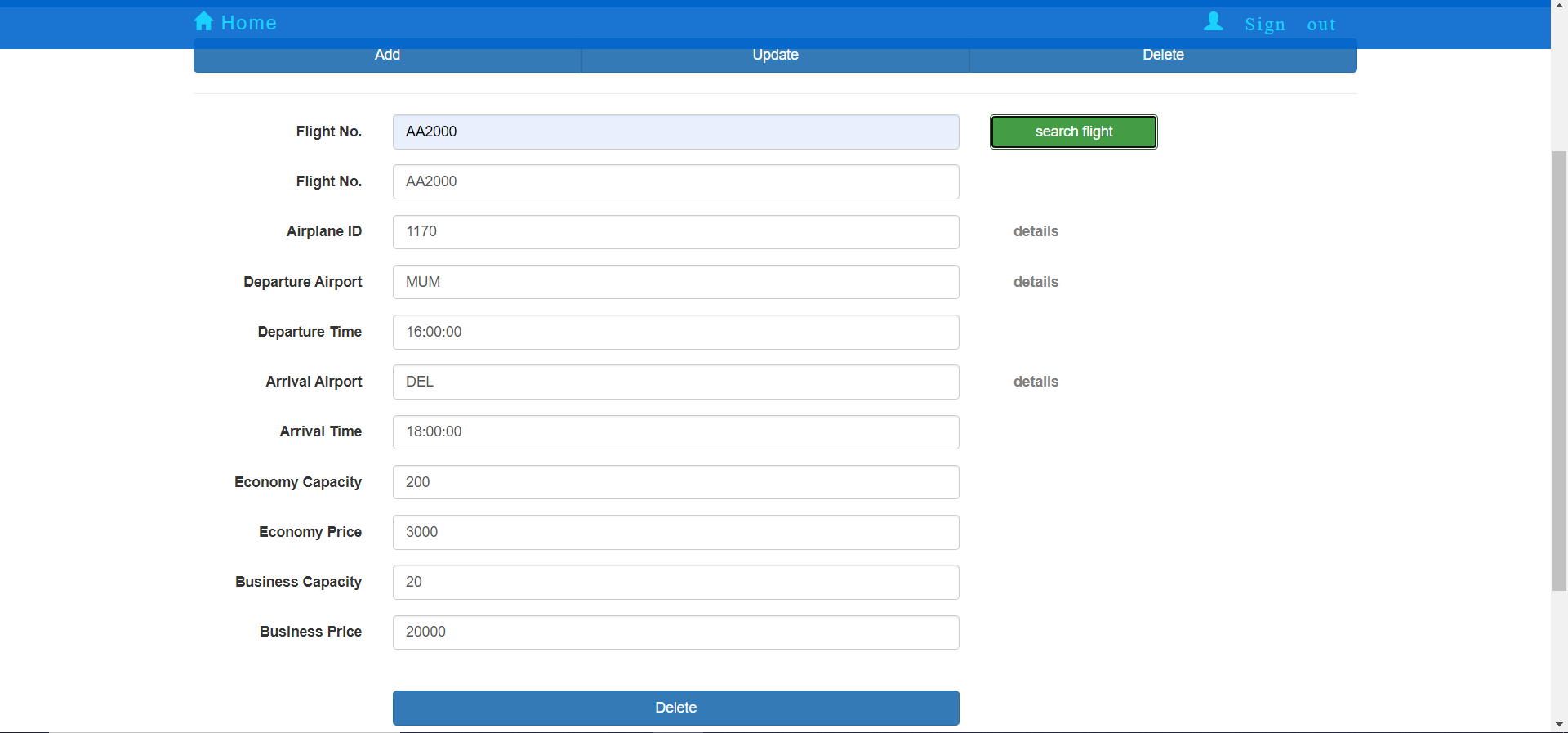


Database:



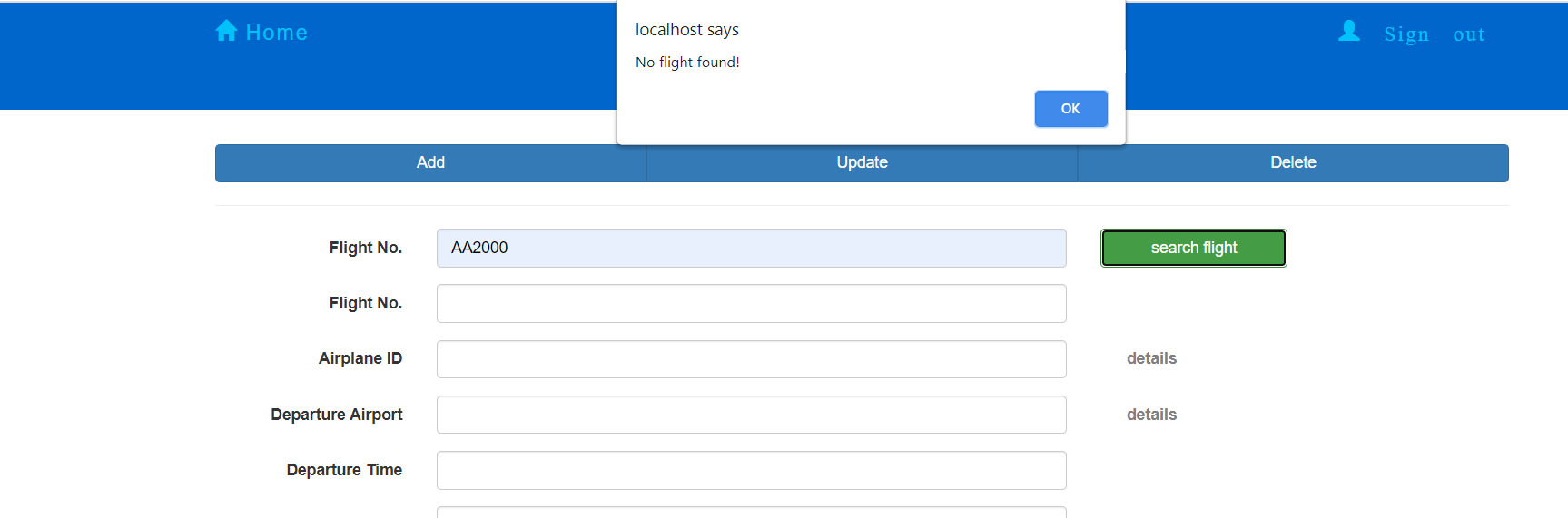
**Delete Flight**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Test Scenario 3** | | Verify the functionality and working of Delete Flight Option in Admin Module | | |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |
| **Step #** | **Step Details** | | **Expected Results** | | **Actual Results** | | | **Pass / Fail / Not executed / Suspended** | | |
|  |
| 1 | Start the website from localhost | | Site should open | | As Expected | | | Pass | | |  |
| 2 | Enter Flight Number | | Flight Number Can be entered | | As Expected | | | Pass | | |  |
| 3 | Press Search Flight | | Details should autofill | | As Expected | | | Pass | | |  |
| 4 | Press Delete | | Flight Details should get deleted | | As Expected | | | Pass | | |  |



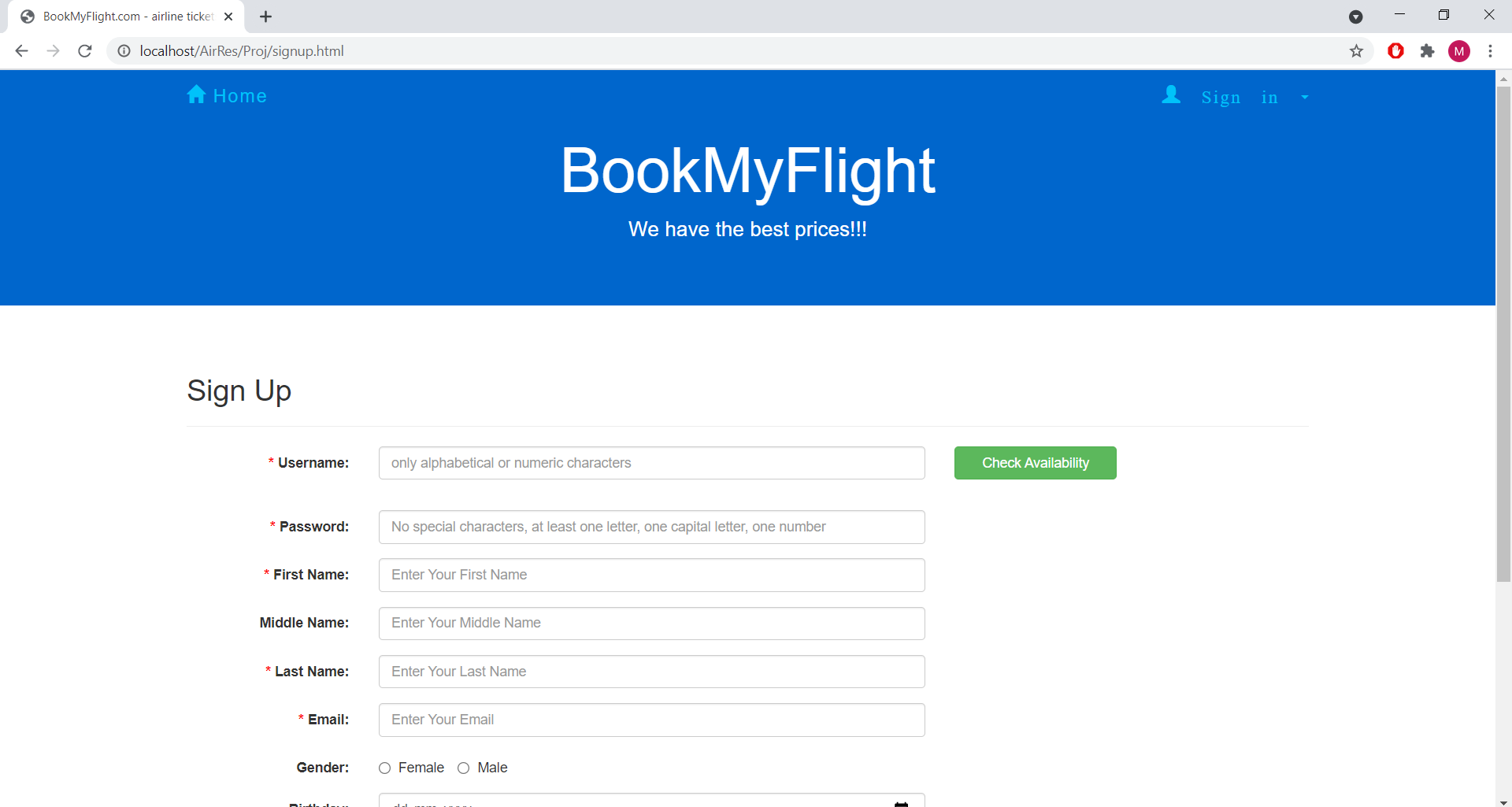


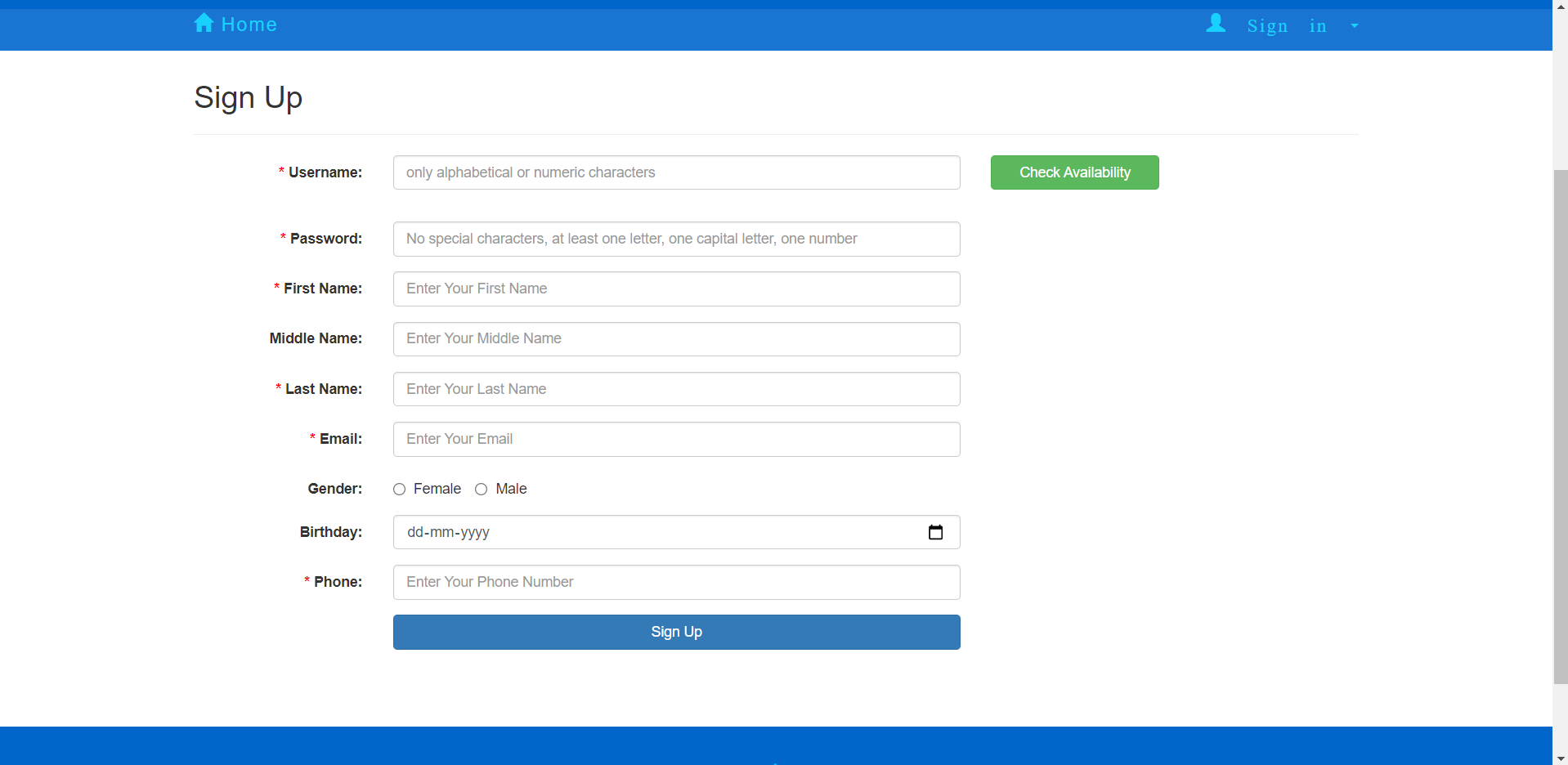
After Deletion on Searching same Flight Again:



**GUI Design**

**Create Account :**

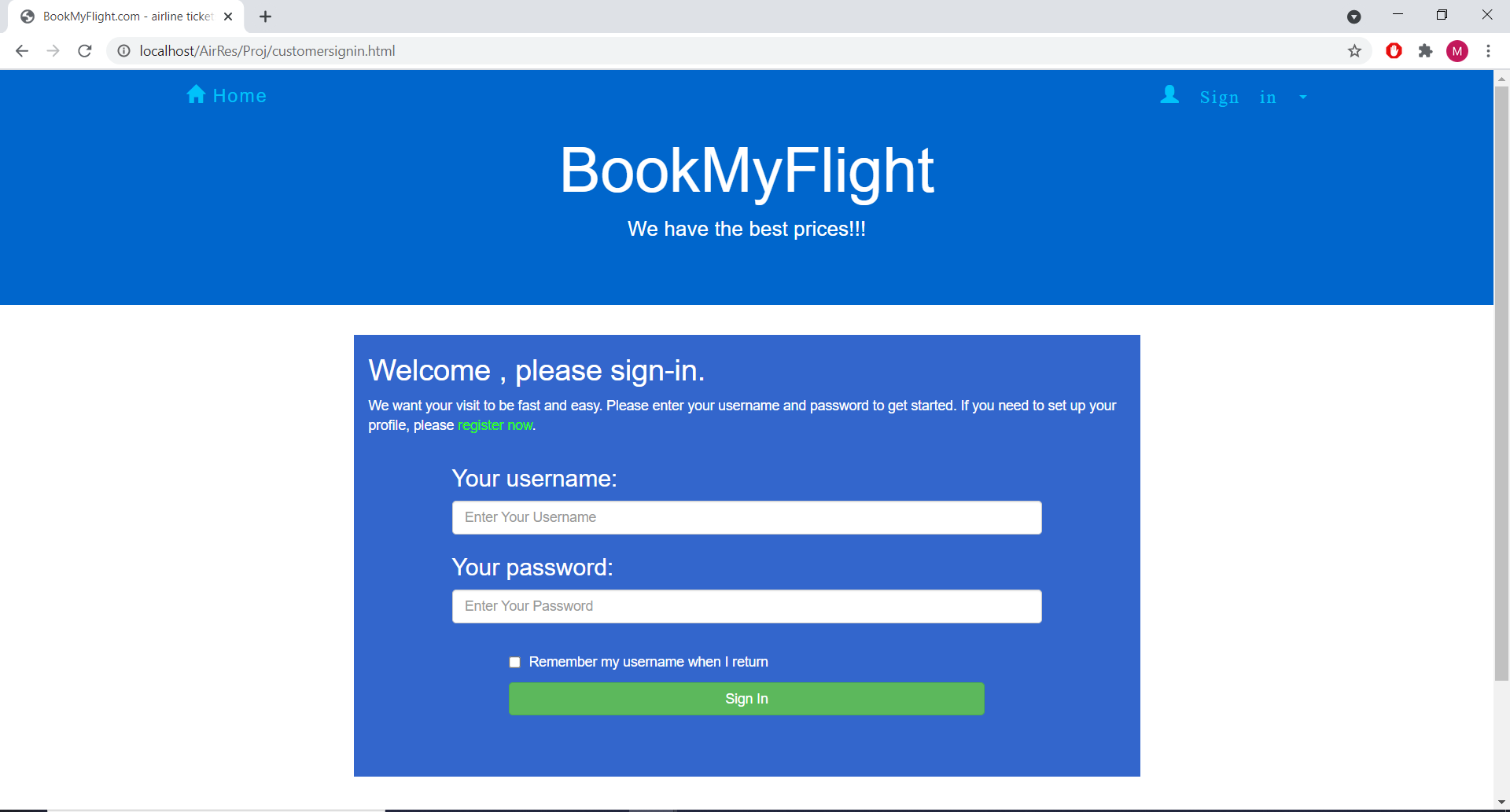




**Features:**

* Users can create an account by supplying their name, email, phone number, gender and providing a password.
* The interface is simple and clean and is an efficient GUI design.
* Users only need to provide basic information before they use the service.
* Not too cluttered and clean colours.

**Login:**

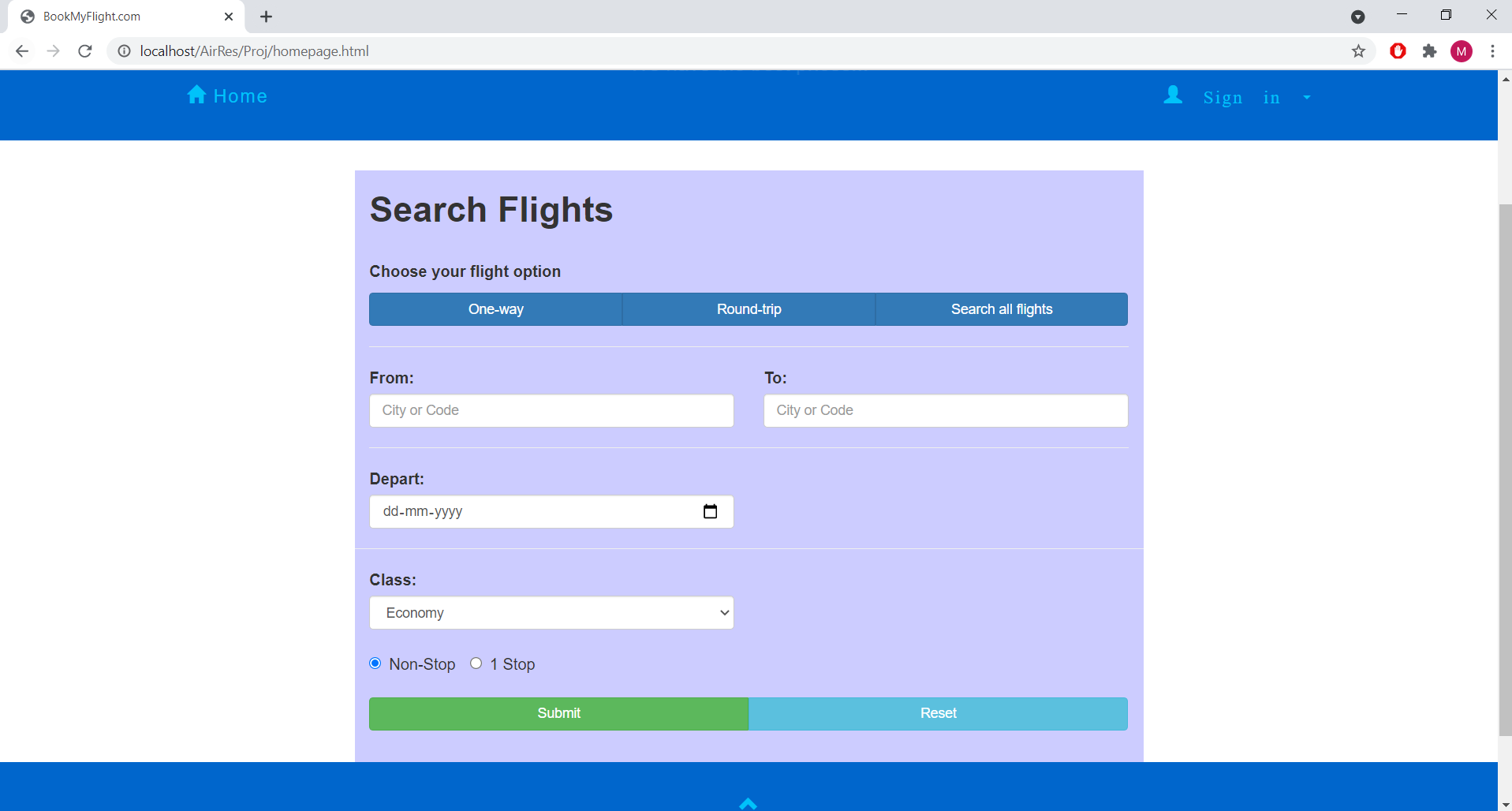


**Features:**

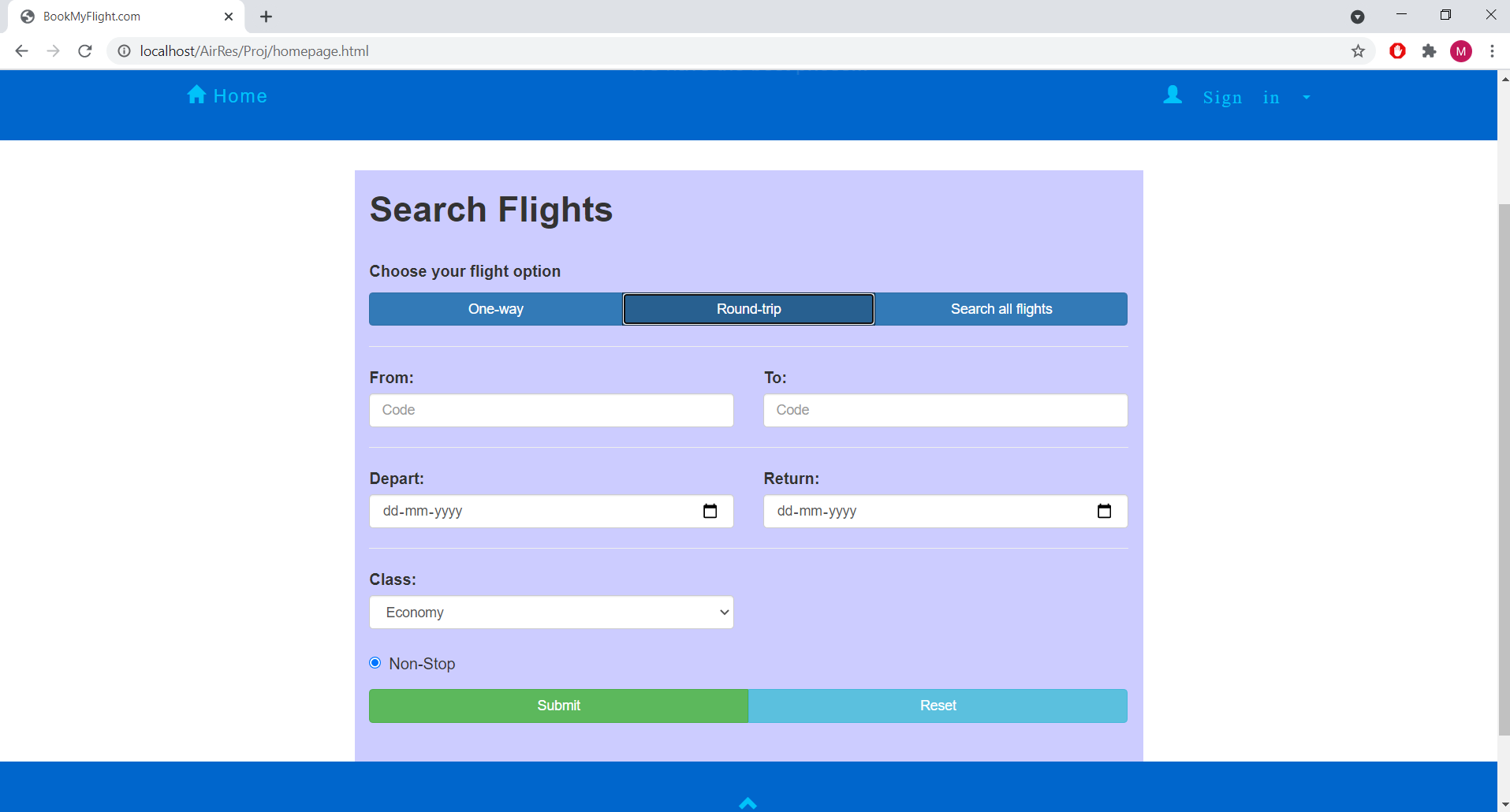
Basic and clean Login Interface in which users just have to enter their username and password to start using the service.

**Select Flight:**

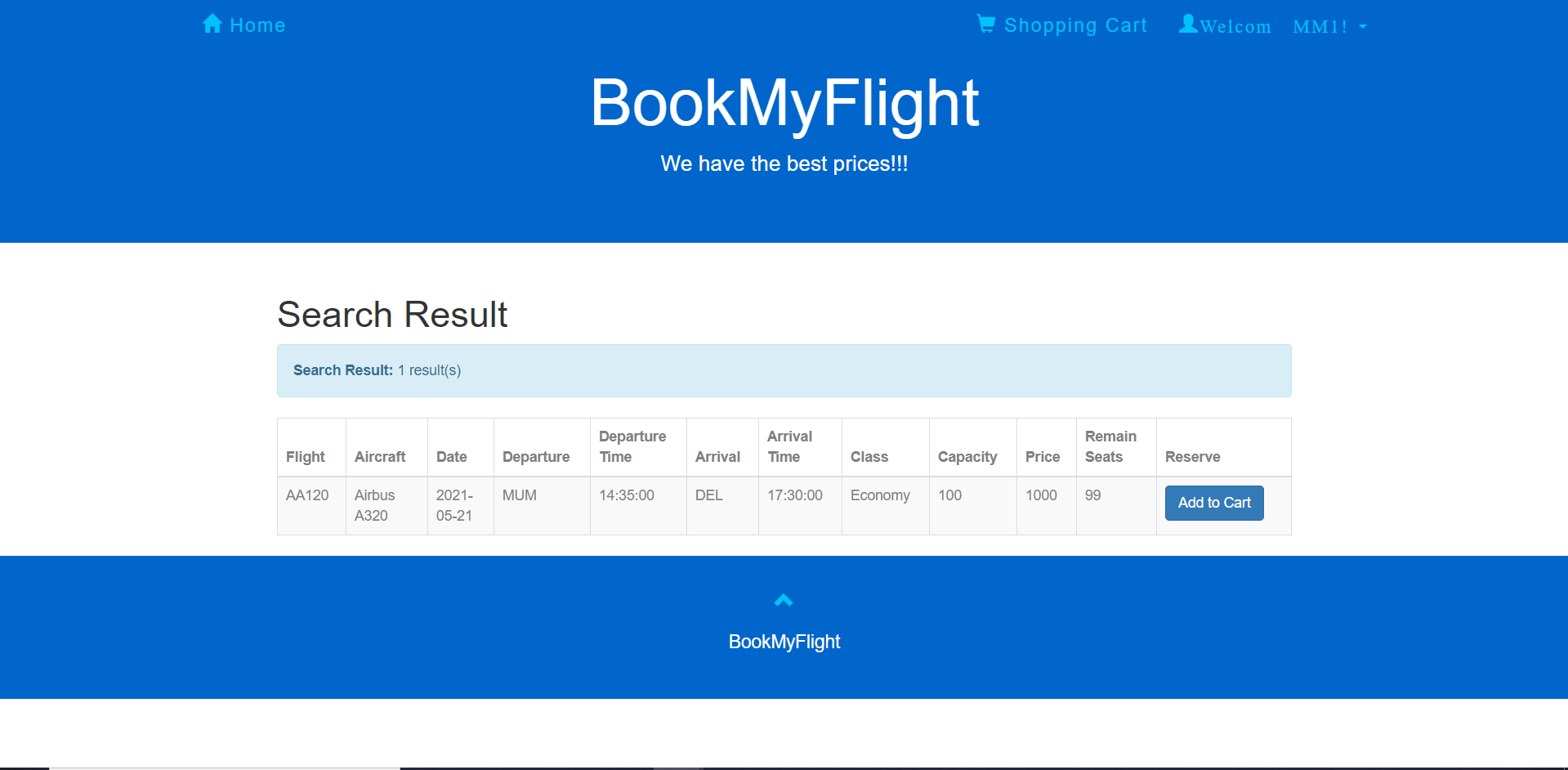
**One way:**



**Round Trip:**



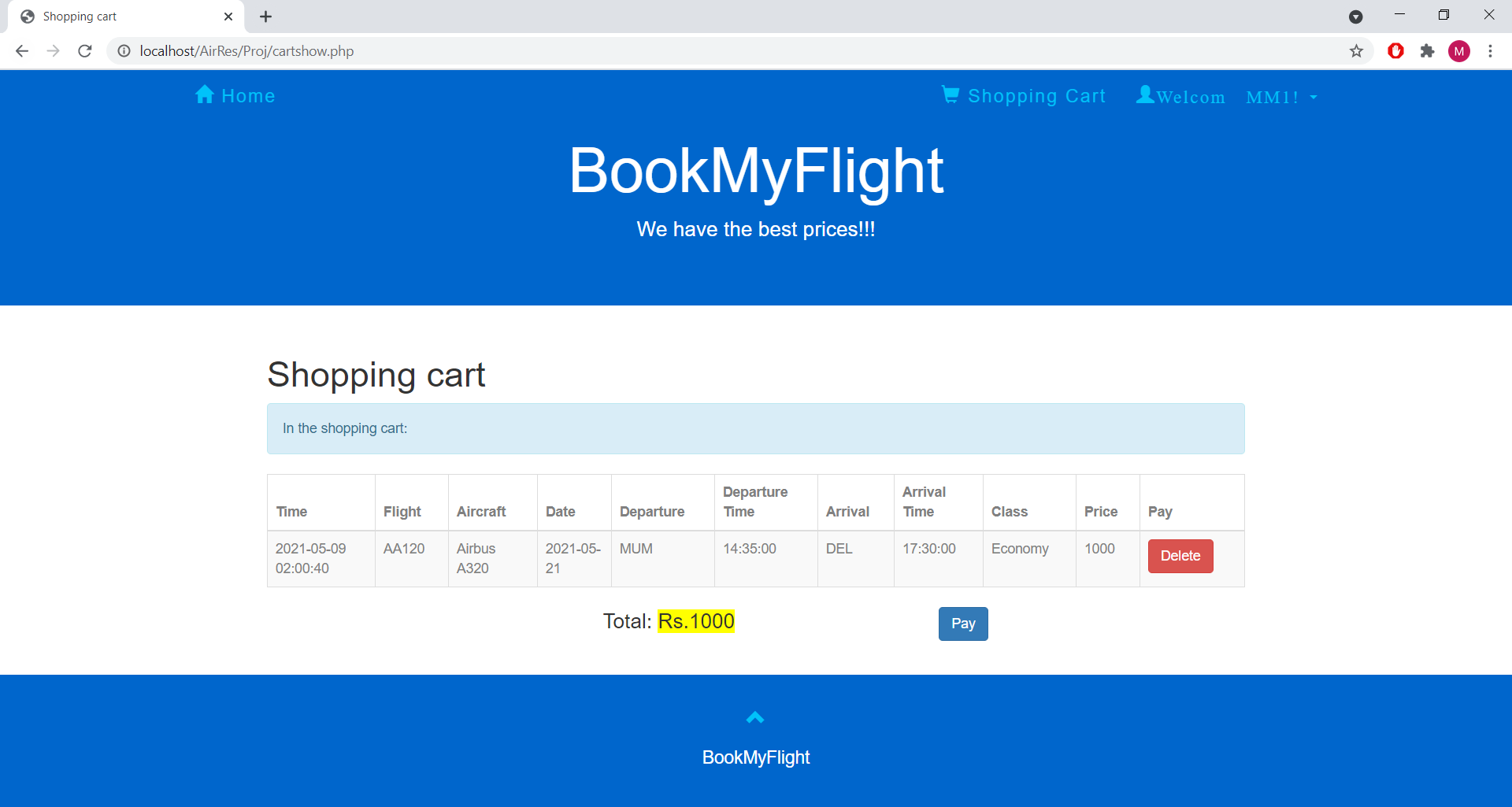
**Search results:**



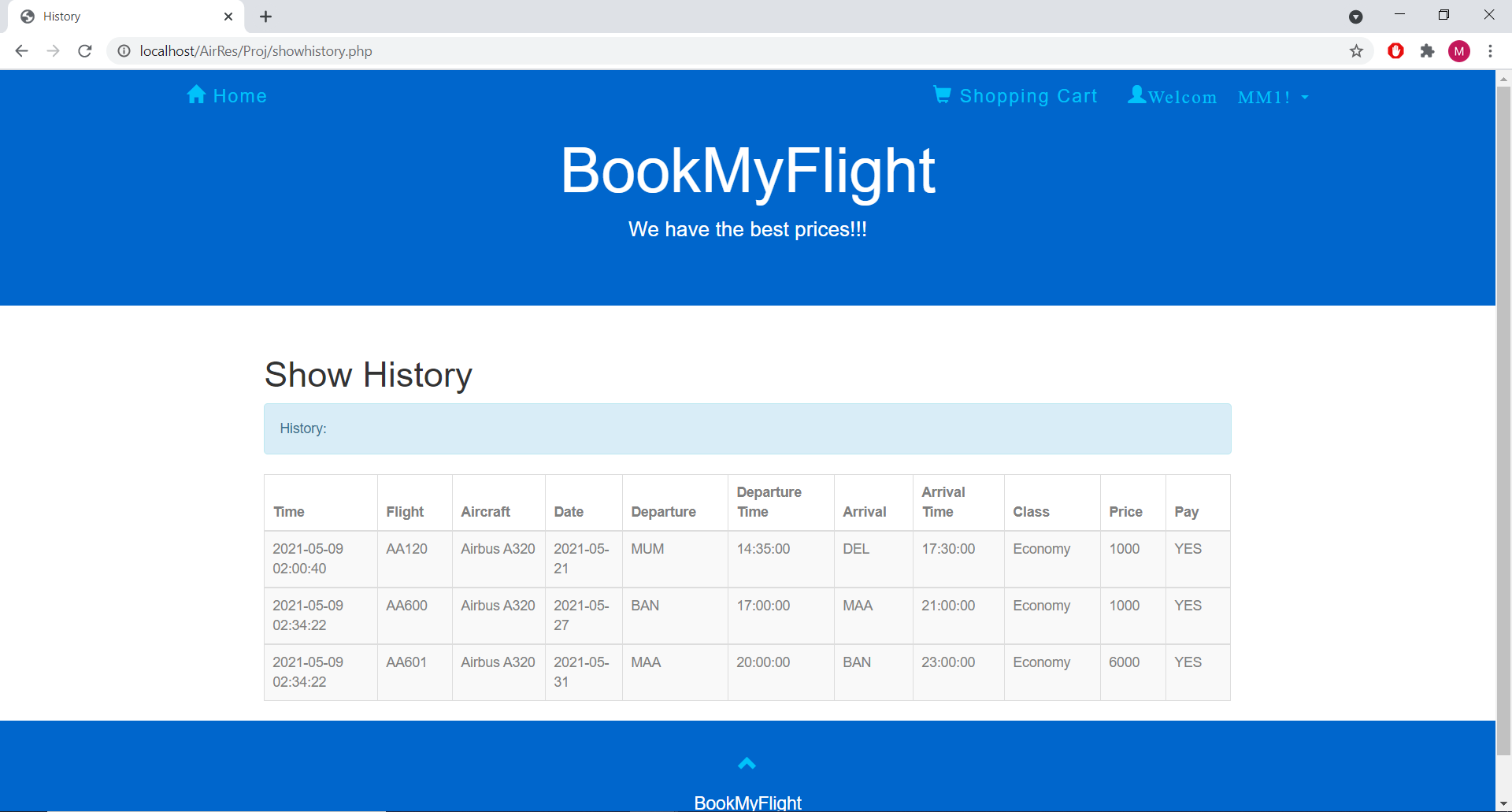
**Features:**

* Users can easily search flights desired departure and arrival destinations.
* Users can opt for either one way or round trip journeys and book accordingly.
* Search results are accurate and details are transparent and clearly understandable.
* UI is smooth and clean with no pop-ins and consistent colour theme.

**Shopping Cart and Payment:**



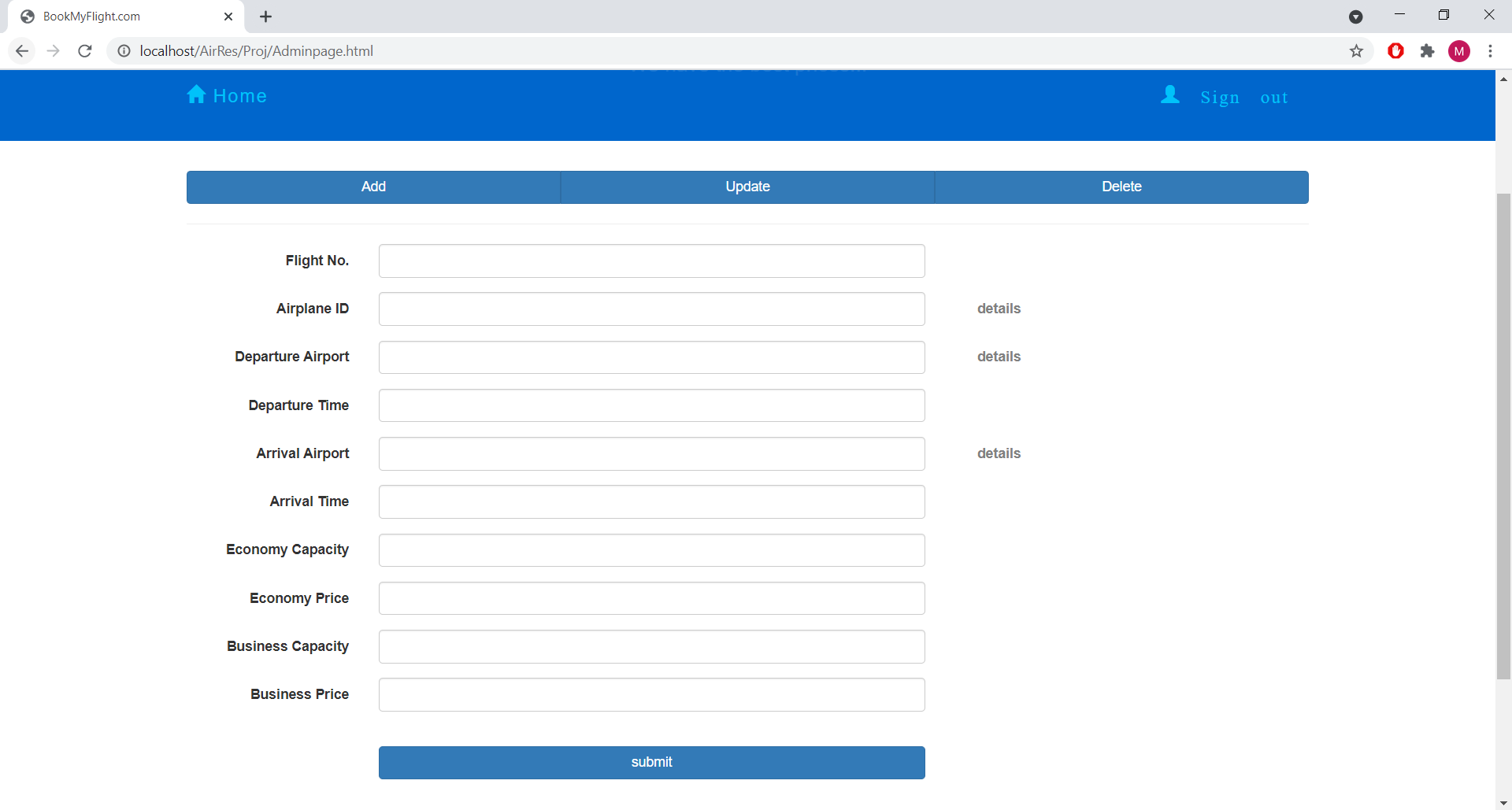
**History:**



**Features:**

* In the shopping cart feature, users can book multiple flight tickets and can pay for them in one place instead of separate payments for each ticket.
* It provides ease of access to users and decreases confusion.
* It is an optimized solution.
* The history feature allows users to keep track of their purchases and flight tickets.
* It is an efficient memory solution for users to keep track of booked tickets and past journeys.

**Admin:**



**Features:**

* Admin has the ability to Add, Update or Delete Flight details by entering the required data fields.
* Admin does most of the job on the backend (not user centric). Hence the admin is provided with a basic UI design which will optimize their workflow.

**Conclusion**

Our Airline Reservation System achieves the objectives that it sets out to reach. The Airline Reservation System is created by applying the fundamentals of Software Engineering and the report contains User Requirements Specification, Planning Chart, System Models with ER diagrams, Data Flow Diagrams, Use Case Diagrams, Sequence Diagrams, various Testing Cases and GUI Design. The project successfully creates a platform where users can create and login via their own account, select flights as per their own convenience and check out their shopping cart and booking history. It also incorporates admin functionalities such as add, update and delete flight details.