**SWIFT AIRLINES**

**1. Overview**

* 1. **Purpose**

The purpose of this document is to describe the requirements that are necessary for developing the Airline Reservation System. It also helps to gather and analyze the new ideas incorporated to the existing reservation system. The intended audience is any person who wants to view, book or cancel air tickets.

* 1. **Scope**

The system provides an online interface where users can search for specific flights from any source to destination. The user can view flight timings, book and cancel tickets. There are several features offered in this system that sets it apart from the many existing systems, such as allowing booking till the last minute as emergency services.

* 1. **Objective**

**General Objective:**

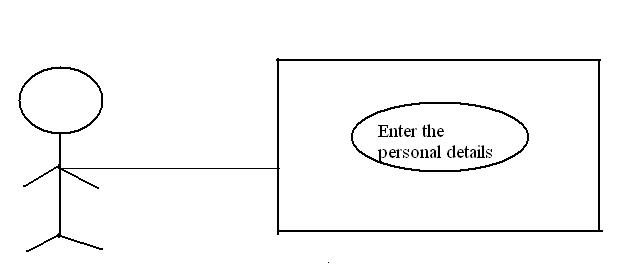
The main goal of the proposed system is to make the process of airline reservation online.

**Specific Objectives:**

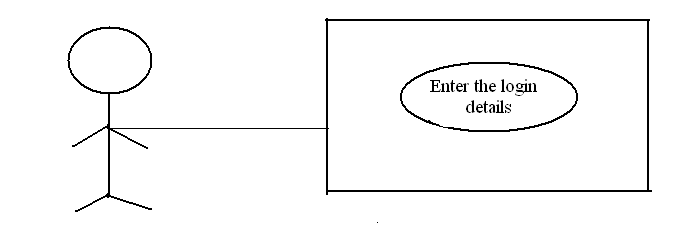
* To create a system that makes booking and cancelling easy and user friendly.
* Make the entire process of itinerary decisions On-The-Go.
* To provide the customer more options like last minute emergency booking, low-cost chartered flights etc.
* Reduce the work load of ticket officers and counters.
* To enable bulk amount of booking and cancellations in a very short time.
* Instant refund of payments
  1. **Requirements**
     1. **Business Requirements**
* Investment, to host and advertise the application
* A technical and managerial team, to maintain the system and update it on a consistent basis
  + 1. **System Requirements**
* PHP 5.3+, HTML5, Twitter Bootstrap v3, jQuery 1.7, MySQL DB
* Windows/Fedora operating system
* Server Bandwidth : 100TB, Space: unlimited
* Database space: Unlimited
  + 1. **Functional Requirements**

The functional requirements of the system are divided among the Customers and Administrator of the application. It can be explained in detail:

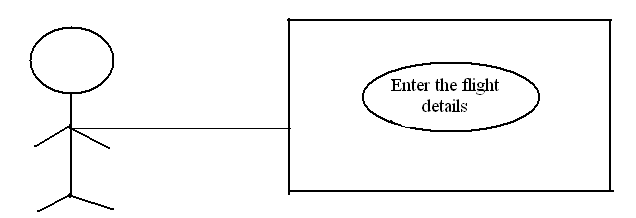
* + - 1. **Use case name: User Registration**
* **Description:** This use case describes the scenario where the customer should provide all their details in order to create an account for themselves to use for login purposes.
* **Actor:** User or the customer
* **Input:** Personal details such as Name, Address, Phone Number and Email ID etc.
* **Output:** All the details will be saved into a database and can be used to fill in the appropriate fields at the time of booking.

Use Case: ****Fig 1.1

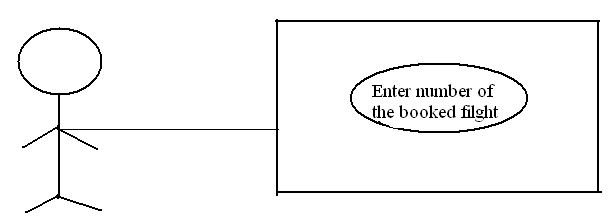
* + - 1. **Use case name: User Login/Logout**
* **Description:** This use case describes the scenario where the customer logs into their account by entering the username and password provided at the time of registration.
* **Actor:** User or the customer
* **Input:** Username, password and/or any other form of verification (eg: captcha). Click the logout button.
* **Output:** The entered fields will be matched with the data from the database. If the case and content match, the user is logged into their profile or asked to re-enter the details. The user will be logged out of their account as and when they wish. This prevents their personal information from being shared with subsequent users.

Use Case: ****Fig 1.2

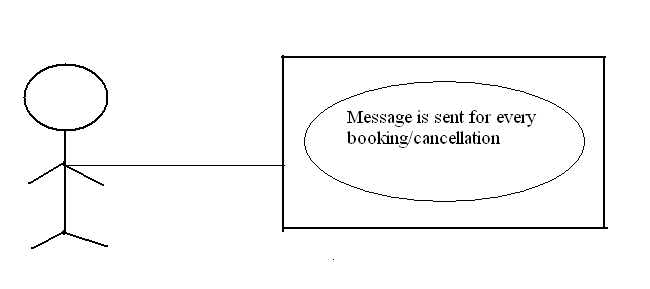
* + - 1. **Use case name: Book Flights**
* **Description:** This use case describes the scenario where the customer searches for flights and goes ahead with booking.
* **Actor:** User or the customer
* **Input:** Source, Destination names or flight codes, Date of travel, number of passengers and class.
* **Output:** Display all the flights with departure, arrival time (24hr format) and fare

Use Case: ****Fig 1.3

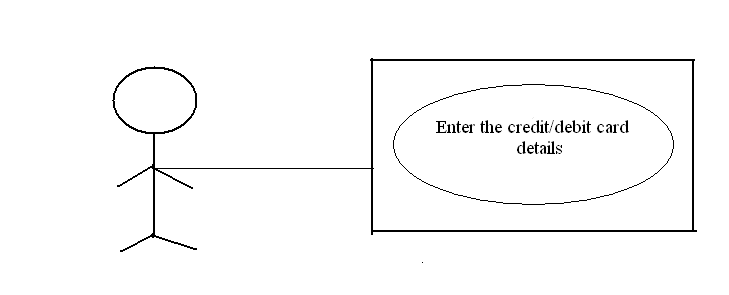
* + - 1. **Use case name: Cancel/Modify Flights**
* **Description:** This use case describes the scenario where the customer wishes to cancel/modify any bookings that have been made. Only those bookings can be cancelled/modified whose departure time has not passed.
* **Actor:** User or the customer
* **Input:** Form of identification of flight (eg: PNR number), email ID, phone number
* **Output:** If the details of the flight given at the time of booking match, the selected booking can be cancelled/modified. A portion of the flight fare is retained as cancellation/modification fee and/or service charges. However, the reversal of a part of the fare applies to only refundable bookings.

Use Case: Fig 1.4

* + - 1. **Use case name: Booking/Cancellation Confirmations**
* **Description:** This use case describes the scenario where the customer receives confirmation from the website upon any booking or cancellations made in their name.
* **Actor:** Administrator
* **Input:** Administrator makes the system with an automatic feature to send an alert.
* **Output:** The customer gets alert, as an Email and/or an SMS when a booking or cancellation has been confirmed. To avoid any misunderstand from the customer’s part, the system has it mandatory to re-confirm such actions.

Use Case: ****Fig 1.5

* + - 1. **Use case name: Payment Mode**
* **Description:** This use case describes the scenario where the customer is required to pay for the booking(s) made.
* **Actor:** User or the customer
* **Input:** Credit Card details, each time, for non-account holders, email ID, phone number, security pin/questions
* **Output:** Confirmation of booking if all the bank details get verified OK. Error in payment if they do not match.

Use Case: ****Fig 1.6

* + - 1. **Administrator Activities:**
* **Add/Modify Flight Information:** The Administrator has the sole rights to add, delete or modify the flight information. Sometimes, flights get delayed or cancelled. The details of such flights need to be removed/ modified. The changes must also be reflected in the databases.
* **Cancellation of Reservations:** The administrator has a duty to approve the cancellation of reservations made. Send an acknowledgement of the cancellation and approve for the reversal of the amount previously paid.
* **E-mail confirmations:** Whenever a customer makes or cancels a reservation, the administrator is responsible for sending confirmation e-mails to the customer, confirming the booking with details.
  + 1. **Non-Functional Requirements:**
       1. **Performance Requirements:**
* The system is not run from a single server as that will not be able handle the load when the site visits go beyond a threshold limit. Too many page visits may slow down page navigations and also payment authorization, eventually leading to a server crash. To handle bulk amount of traffic, multiple servers are to be set up. This will ensure smooth functionality of the booking system.
* The beta version of application is run on localhost hence the response time depends on the RAM speed of the processor being used
  + - 1. **Security Requirements:**
* Users are required to login with their user ID and password to book/cancel flight details and modify personal details
* The system permits only authorized members to do administrative tasks.
* The system allows users to view only their information and no other users’.
* This system is SQL Injection proof and safe from XSS attack.

1. **Literature Survey:**
   1. **Existing Systems:**

The existing online airline booking systems come with many banners of advertisements and pop-ups on the sides of the website that it hinders with the entire booking process. It is known that a booking is not finalized till the transaction of payment is authorized and confirmed. Such have been instances where the user books their ticket provisionally but fails to go through with the transaction. Such tickets are not booked however the reversal of their bookings is not reflected instantly instead take an average of 20 minutes.

Existing system allows only those bookings that are confirmed 5 hours before the departure of the flight. These happenings invite many customer care calls towards the company. To attend these queries, man hours are put on the line.

Many systems crash during the peak hour of their local time. The servers are not enough to handle the load then. Too many users make the page navigations slow, hindering the overall number of bookings that go through till the end.

* 1. **Proposed System:**
* The proposed system towards online airline reservation is faster. Unlike many existing systems, there are multiple servers installed to run the website. This reduces almost all possibilities for a website to hang as a result of server crash.
* When transactions do not get authorized, the reversals of the reservations appear on the system without much delay so that they are open for the next user. Such a feature is helpful when multiple users do bulk booking.
* For travel security reasons, the bookings are expected to be done a few hours prior to the flight departure. This system allows last minute booking for emergencies, if the flight is not fully booked, as long as it does not challenge the airlines’ security.

Overall, the proposed system is robust as it caters to all the needs of present day users expect from an online booking.

1. **Implementation**
   1. **Coding**

**3.1.1**Index.php

<div class="col-lg-8">

<div class="well bs-component">

<form class="form-horizontal" action="book.php?count\_a=2" method="GET">

<?php

if(isset($\_GET['path'])===true

&&isset($\_GET['from\_city'])===true &&isset($\_GET['to\_city'])===true

&&isset($\_GET['departure\_date'])===true

&&isset($\_GET['count\_a'])===true &&isset($\_GET['count\_c'])===true &&isset($\_GET['class'])===true) {

$from = $\_GET['from\_city'];

$to = $\_GET['to\_city'];

$departdate = $\_GET['departure\_date'];

$class = $\_GET['class'];

$path = $\_GET['path'];

$counta = $\_GET['count\_a'];

if($path==='oneway') {

echo '<legend>Flights from '.$from.' to '.$to.'</legend>';

$query = "SELECT \* FROM `flight\_search` WHERE `from\_city`= '$from' AND `to\_city` = '$to' AND `departure\_date` = '$departdate'";

$result = mysql\_query($query);

if($result) {

if(mysql\_num\_rows($result) > 0) {

while($row = mysql\_fetch\_assoc($result)) {

?>

<?php if($class==='Economy') { ?>

<td><input type="radio" name="chose\_to" value="<?php echo $row['fno']; ?>"/><?php echo $row['fno']; ?></td>

<td><?php echo $row['departure\_time']; ?></td>

<td><?php echo $row['arrival\_time']; ?></td>

<td><?php echo $row['e\_seats\_left']; ?></td>

<td><?php echo $row['e\_price']; ?></td>

<?php } else if($class==='Business') { ?>

<td><input type="radio" name="chose\_to" value="<?php echo $row['fno']; ?>"/><?php echo $row['fno']; ?></td>

<td><?php echo $row['departure\_time']; ?></td>

<td><?php echo $row['arrival\_time']; ?></td>

<td><?php echo $row['b\_seats\_left']; ?></td>

<td><?php echo $row['b\_price']; ?></td>

<?php } else { 'Not enough seats left, please search again!'; }

}?>

</tr>

</tbody>

</table>

<input type="hidden" name="count\_a" value="<?php echo $counta; ?>"/>

<center><button type="submit" id="class" name="class" value="<?php echo $class; ?>" class="btnbtn-primary">Choose Flights</button></center>

<?php

} else { echo 'No flights found';}

}

else { die(mysql\_error()); }

}

else if($path==='return') {

echo '<legend>Flights from '.$from.' to '.$to.'</legend>';

$query1 = "SELECT \* FROM `flight\_search` WHERE `from\_city`= '$from' AND `to\_city` = '$to' AND `departure\_date` = '$departdate'";} ?>

* + 1. Book.php

<?php

if(f\_logged\_in()===true) {

if(isset($\_GET['class'])===true &&isset($\_GET['count\_a'])===true &&isset($\_GET['count\_c'])===true) {

if(isset($\_GET['chose\_to'])===true) {

if(isset($\_GET['chose\_fro'])===true) {

$to = $\_GET['chose\_to'];

$fro = $\_GET['chose\_fro'];

$counta = $\_GET['count\_a'];

$countc = $\_GET['count\_c'];

$class = $\_GET['class'];

$q1 = "SELECT \* FROM `flight\_search` WHERE `fno`='$to'";

$r1 = mysql\_query($q1);

while($row1 = mysql\_fetch\_assoc($r1)) {

echo '<legend>Your flight from '.$row1['from\_city'].' to '.$row1['to\_city'].'</legend>';

echo 'Flight Number: '.$row1['fno'].'<br>';

if($class==='Economy') {

if(($counta+$countc)<=$row1['e\_seats\_left']) {

echo 'Departure Time: '.$row1['departure\_time'].'<br>';

echo 'Arrival Time: '.$row1['departure\_time'].'<br>';

echo 'Number of Adults: '.$counta.'<br>';

echo 'Number of Children: '.$countc.'<br><br>';

$eat1 = $counta \* $row1['e\_price'];

echo 'Cost of '.$counta.' adult(s): '.$counta.' \* Rs. '.$row1['e\_price'].' = Rs. '.$eat1.'<br>';

$ect1 = ($countc \* $row1['e\_price']) \* 0.75;

echo 'Cost of '.$countc.' child: 75% of '.$countc.' \* Rs. '.$row1['e\_price'].' = Rs. '.$ect1.'<br>';

$etax1 = 500;

echo 'Service Tax: Rs. '.$etax1.'<br><br>';

$etot1 = $eat1 + $ect1 + $etax1;

echo '<legend>Cost of one trip is Rs. '.$etot1.'</legend><hr>';

}

else {

echo 'Not enough available seats. Sorry, please check some other flight!';

header("refresh:10; url=index.php");

}

* + 1. Loginact.php

<?php

$title = 'Swift Airlines | Login Error';

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/core/init.php';

if(empty($\_POST) === false) {

$f\_uname = $\_POST['f\_uname'];

$f\_password = $\_POST['f\_password'];

if(empty($f\_uname) === true || empty($f\_password) === true){

$errors[] = 'You need to enter both, the username and the password!';

}

else if (f\_exists($f\_uname)===false) {

$errors[] = 'No such user exists! Please register!';

}

else if(f\_active($f\_uname)===false) {

$errors[] = 'Please activate your account!';

}

else {

if(strlen($f\_password)>32) {

$errors[] = 'Password too long!';

}

$f\_login = f\_login($f\_uname, $f\_password);

if($f\_login===false) {

$errors[] = 'Username and Password do not match!';

}

else {

$\_SESSION['f\_id'] = $f\_login;

header('Location: http://localhost/swift/index.php');

exit();

}

}

}

else {

$errors[] = 'No Log In credentials received!';

}

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/includes/overall/header.php';

if(empty($errors) === false) {

?>

<br/><h4>We tried to log you in, but : </h4><br/>

<?php

echooutput\_errors($errors);

}

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/includes/overall/footer.php';

?>

* + 1. Card.php

<?php

if(f\_logged\_in()===true) {

$puname = $\_GET['puname'];

$puadd = $\_GET['puadd'];

$puphno = $\_GET['puphno'];

$pumail = $\_GET['pumail'];

if(isset($\_GET['chose\_to'])===true) {

$to = $\_GET['chose\_to'];

$class = $\_GET['class'];

$counta = $\_GET['count\_a'];

$countc = $\_GET['count\_c'];

if(isset($\_GET['chose\_fro'])===true) {

$fro = $\_GET['chose\_fro'];

if($class==='Economy') {

$eat1 = $\_GET['eat1'];

$eat2 = $\_GET['eat2'];

$ect1 = $\_GET['ect1'];

$ect2 = $\_GET['ect2'];

$etax1 = $etax2 = 500;

$etot1 = $eat1 + $ect1 + $etax1;

$etot2 = $eat2 + $ect2 + $etax2;

$etotr = $etot1 + $etot2;

if($countc==='0') {

if($counta==='1') {

$aname1 = $\_GET['aname1'];

$aage1 = $\_GET['aage1'];

$asex1 = $\_GET['asex1'];

}

else if($counta==='2') {

$aname1 = $\_GET['aname1'];

$aage1 = $\_GET['aage1'];

$asex1 = $\_GET['asex1'];

$aname2 = $\_GET['aname2'];

$aage2 = $\_GET['aage2'];

$asex2 = $\_GET['asex2'];

}

}

* + 1. Confirm.php

<?php

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/core/init.php';

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/includes/overall/header.php';

if(f\_logged\_in()===true) {

$uid = $f\_data['f\_id'];

if(isset($\_POST['cardnum'])===true &&isset($\_POST['cvvnum'])===true) {

$cardnum = $\_POST['cardnum'];

$cvvnum = $\_POST['cvvnum'];

$puname = $\_POST['puname'];

$puphno = $\_POST['puphno'];

$puadd = $\_POST['puadd'];

$pumail = $\_POST['pumail'];

if(isset($\_POST['chose\_to'])===true) {

if(isset($\_POST['chose\_fro'])===true) {

$to = $\_POST['chose\_to'];

$fro = $\_POST['chose\_fro'];

$counta = $\_POST['count\_a'];

$countc = $\_POST['count\_c'];

$class = $\_POST['class'];

if($class==='Economy' && $counta==='1' && $countc==='0') {

$aname1 = $\_POST['aname1'];

$asex1 = $\_POST['asex1'];

$aage1 = $\_POST['aage1'];

$ect1 = $\_POST['ect1'];

$ect2 = $\_POST['ect2'];

$eat1 = $\_POST['eat1'];

$eat2 = $\_POST['eat2'];

$etotr = $\_POST['etotr'];

$totc = $counta + $countc;

$status = 'Booked';

$q1 = "SELECT \* FROM `flight\_search` WHERE `fno`='$to'";

$r1 = mysql\_query($q1);

while($row1 = mysql\_fetch\_assoc($r1)) {

$from\_city = $row1['from\_city'];

$to\_city = $row1['to\_city'];

$depart\_date = $row1['departure\_date'];

$arr\_date = $row1['arrival\_date'];

$depart\_time = $row1['departure\_time'];

$arr\_time = $row1['arrival\_time'];

}

$q10 = "SELECT `c\_balance` FROM `card\_details` WHERE `c\_cvv`='$cvvnum' AND `c\_cnum`='$cardnum'";

$r10 = mysql\_query($q10);

if($r10) {

if(mysql\_num\_rows($r10)>0) {

while($row10 = mysql\_fetch\_assoc($r10)) {

$balance = $row10['c\_balance'];

if($balance>=$etotr) {

$deduct = $balance - $etotr;

mysql\_query("UPDATE `card\_details` SET `c\_balance`='$deduct' WHERE `c\_cvv`='$cvvnum' AND `c\_cnum`='$cardnum'");

$q11 = "SELECT `e\_seats\_left` FROM `flight\_search` WHERE `fno`='$to'";

* + 1. Profile.php

<?php

$title = 'Swift Airlines | Profile';

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/core/init.php';

user\_protect\_page();

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/includes/overall/header.php';

if(isset($\_GET['f\_uname']) === true && empty($\_GET['f\_uname']) === false) {

$f\_uname = $\_GET['f\_uname'];

if(f\_exists($f\_uname) === true ) {

$f\_id = f\_id\_from\_username($f\_uname);

$profile\_data = f\_data($f\_id, 'f\_fname','f\_lname','f\_mailid','f\_sex','f\_regdate');

?>

<h4><?php echo $profile\_data['f\_fname']; ?>&#39;s Profile</h4>

<br/>

<div class="container">

<imgsrc="http://localhost/<?php echo $profile\_data['f\_dp'] ?>" height="150" width="150" />

</div>

<hr>

<strong>Name:</strong><?php echo $profile\_data['f\_fname'].' '.$profile\_data['f\_lname']; ?><br/><br/>

<strong>Sex:</strong><?php echo $profile\_data['f\_sex']; ?><br/><br/>

<strong>Campus:</strong><?php echo $profile\_data['f\_campus']; ?><br/><br/>

<strong>Department:</strong><?php echo $profile\_data['f\_dept']; ?><br/><br/>

<strong>Email ID:</strong><?php echo $profile\_data['f\_mailid']; ?><br/><br/>

<strong>Member Since:</strong><?php echo $profile\_data['f\_regdate']; ?><br/><br/>

<?php

}

else {

echo 'Sorry, that user doesn\'t exists!';

}

}

else {

header('Location: http://localhost/swift/index.php');

exit();

}

include $\_SERVER["DOCUMENT\_ROOT"].'/swift/includes/overall/footer.php';

?>

* + 1. Logout.php

<?php

session\_start();

session\_destroy();

header('Location: http://localhost/swift/index.php');

?>

* 1. **Screenshots**

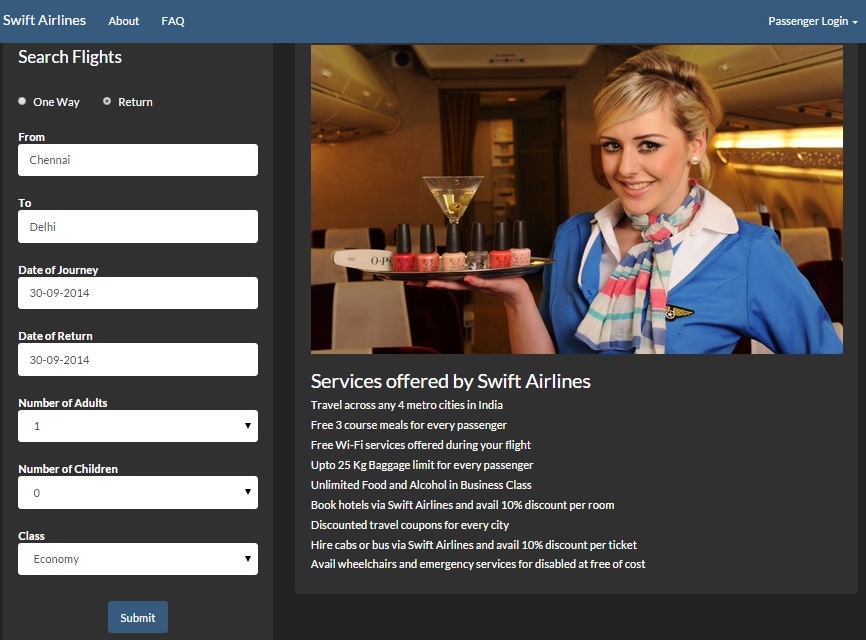
****

Fig 3.1

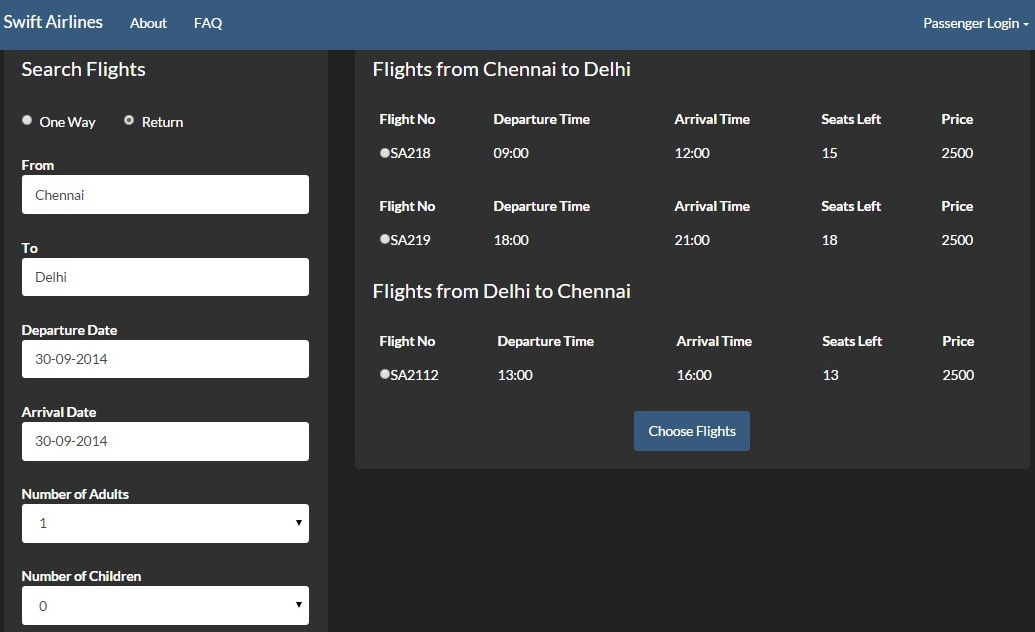
****

Fig 3.2

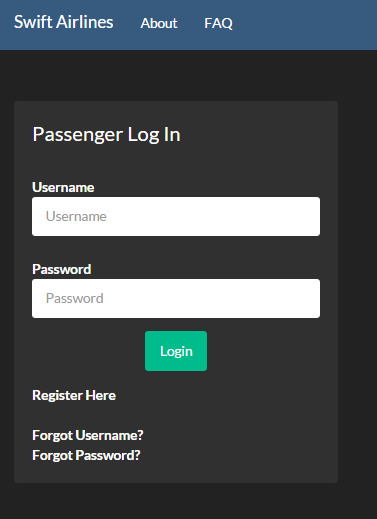
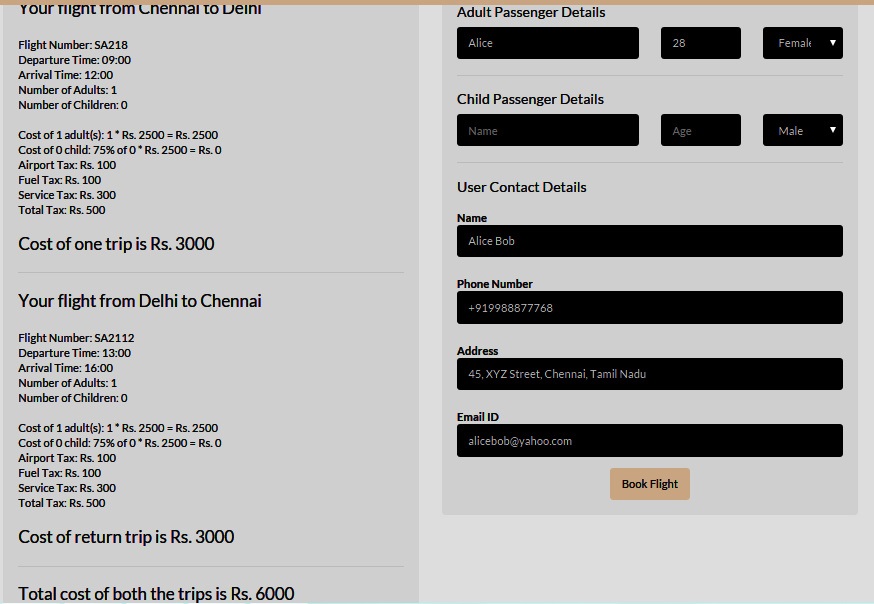
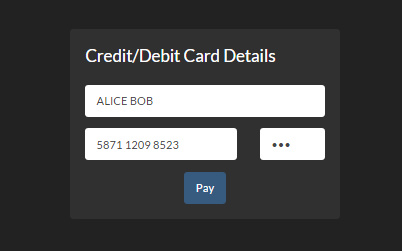
****

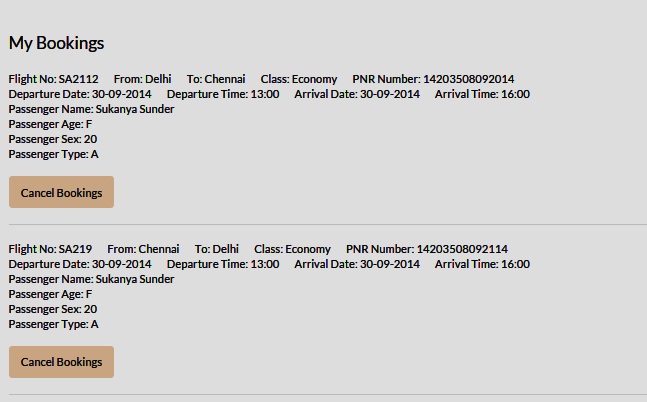
Fig 3.3

****

**Fig 3.4**

****

**Fig 3.5**

****

**Fig 3.6**

1. **Design and Structure**
   1. **Data Flow Diagram**

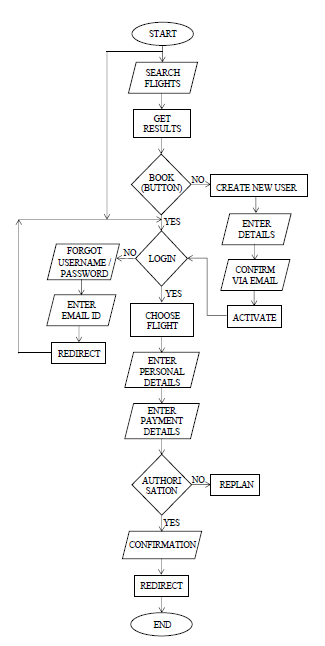
****

Fig 4.1

* 1. **Database Structures**
     1. **flight\_search**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** |
| ***id*** | int(11) | No |  |
| fno | varchar(10) | No |  |
| from\_city | varchar(20) | No |  |
| to\_city | varchar(20) | No |  |
| departure\_date | varchar(20) | No |  |
| arrival\_date | varchar(20) | No |  |
| departure\_time | varchar(20) | No |  |
| arrival\_time | varchar(20) | No |  |
| e\_seats\_left | int(3) | No | 20 |
| b\_seats\_left | int(3) | No | 10 |
| e\_price | int(5) | No |  |
| b\_price | int(5) | No |  |

Table 4.1

* + 1. **flight\_users**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** |
| ***f\_id*** | int(11) | No |  |
| f\_fname | varchar(20) | No |  |
| f\_lname | varchar(20) | No |  |
| f\_sex | varchar(10) | No |  |
| f\_uname | varchar(32) | No |  |
| f\_password | varchar(32) | No |  |
| f\_mailid | varchar(100) | No |  |
| f\_mailcode | varchar(100) | No |  |
| f\_active | int(11) | No |  |
| f\_regdate | datetime | No |  |
| f\_passrec | int(11) | No |  |
| f\_address | varchar(500) | No |  |
| f\_phone | varchar(10) | No |  |

Tbale 4.2

* + 1. **card\_details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** |
| ***c\_id*** | int(11) | No |  |
| c\_name | varchar(50) | No |  |
| c\_cvv | int(3) | No |  |
| c\_cnum | varchar(16) | No |  |
| c\_balance | int(8) | No |  |

Table 4.3

* + 1. **passenger\_details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** |
| ***p\_id*** | int(11) | No |  |
| p\_pnr | varchar(25) | No |  |
| p\_name | varchar(50) | No |  |
| p\_age | varchar(3) | No |  |
| p\_sex | varchar(10) | No |  |
| p\_fno | varchar(10) | No |  |
| p\_from | varchar(10) | No |  |
| p\_to | varchar(10) | No |  |
| p\_dedate | varchar(20) | No |  |
| p\_ardate | varchar(20) | No |  |
| p\_detime | varchar(20) | No |  |
| p\_artime | varchar(20) | No |  |
| p\_status | varchar(20) | No |  |
| p\_class | varchar(10) | No |  |
| p\_passtype | varchar(1) | No |  |

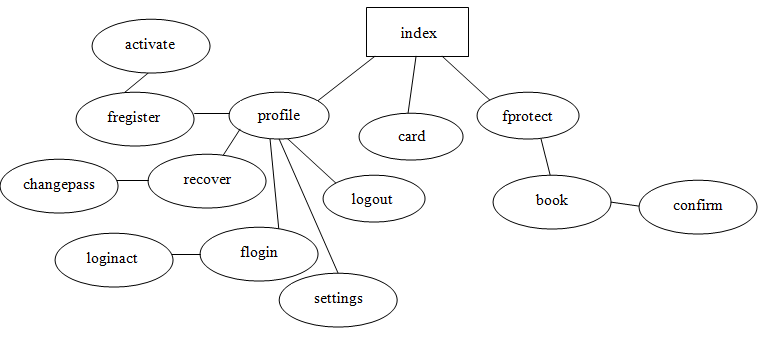
Table 4.4

* + 1. **booking\_details**

|  |  |  |  |
| --- | --- | --- | --- |
| **Column** | **Type** | **Null** | **Default** |
| ***b\_id*** | int(11) | No |  |
| b\_name | varchar(50) | No |  |
| b\_phno | varchar(15) | No |  |
| b\_mail | varchar(50) | No |  |
| b\_add | varchar(100) | No |  |
| b\_price | varchar(10) | No |  |
| b\_child | varchar(3) | No |  |
| b\_adults | varchar(3) | No |  |
| b\_total | varchar(3) | No |  |
| b\_status | varchar(10) | No |  |
| b\_pnr | varchar(20) | No |  |
| b\_card | varchar(16) | No |  |

Table 4.5

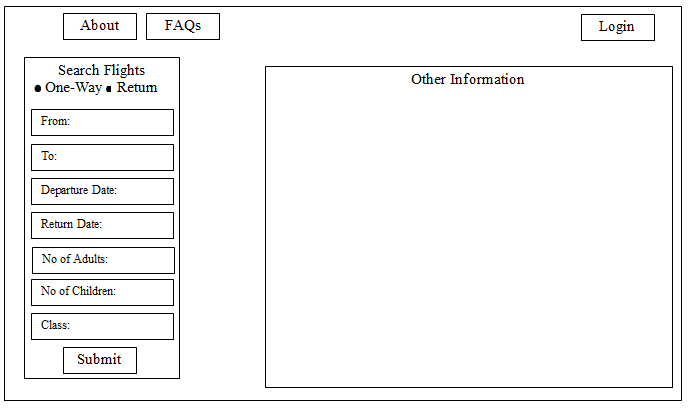
* 1. **Entity – Relationship Diagram**

**** Fig 4.2

1. **External Interfaces**

The different interfaces we would come across while developing the Airline Reservation System are:

* User Interface
* Hardware Interface
* Software Interface
  1. **User Interface**

**** Fig 5.1

* 1. **Hardware Interface**

This Airline Reservation System does not use any hardware interfaces.

* 1. **Software Interface**

This application can run on any Operating System. The beta version of it is run on Windows 7 OS. The application requires a database to store details of flight, user details and booking details etc, MySQL is used for the same. All the coding is done in PHP using HTML and CSS for styling. Xampp Control Panel is used to host the application on web on local port.

1. **Testing**

The approach followed for testing of the Airline Reservation System ensures that the major features of the project are adequately tested. The testing would be carried out on the Airline Reservation System while logging into the system as a customer and administrator of the system. As the system is dynamic, i.e., there is no correct or incorrect output. The output varies depending on the given input and its acceptance by the system. Therefore, no one testing software can be used; rather the tests must be done manually.

The lists of items to be tested in the system are:

* User Login
* User Registrations
* Search Flights
* Book Flights
* Seats Deduction
* Booking Cancellation

**Test Cases:**

|  |  |
| --- | --- |
| **Module** | **User Login** |
| Purpose | To check if the user is a registered user |
| Incorrect input | Incorrect combination of username or password |
| Pass Criteria | A notification appears on the screen that indicates the invalid username or password |
| Correct Input | Valid username and the matching password |
| Pass Criteria | The user is directed to the webpage they intend to go to after logging in. |

Table 6.1

|  |  |
| --- | --- |
| **Module** | **User Registration** |
| Purpose | To create an account for a new user |
| Incorrect input | Incorrect format entered in the input fields for the registration page |
| Pass Criteria | A notification appears on the screen that indicates incorrect input format in the fields. |
| Correct Input | The correct format entered by the user into the input fields of the registration page. |
| Pass Criteria | Successful registration of the user into the Airline Reservation System website. The system would log the user into the system afterwards. |

Table 6.2

|  |  |
| --- | --- |
| **Module** | **Search Flights** |
| Purpose | To search for flights from different source to destination |
| Incorrect input | Attempt to enter a past date according to local time or an invalid city name. |
| Pass Criteria | A notification appears on the screen asking to enter a valid city name |
| Correct Input | A date same as or post the date of booking. City name that is present in the list. |
| Pass Criteria | Accepts the correct date or city name. Moves forward to booking a reservation. |

Table 6.3

|  |  |
| --- | --- |
| **Module** | **Book Flights** |
| Purpose | To book a particular flight |
| Incorrect input | Incorrect format entered in the data fields; leaving any mandatory fields un entered. |
| Pass Criteria | A message has to be generated to the user indicating the wrong entry that he has made in the fields. |
| Correct Input | A correct input would be entering the data into the data entry fields in a correct format. |
| Pass Criteria | A correct input would be entering the data into the data entry fields in a correct format. |

Table 6.4

|  |  |
| --- | --- |
| **Module** | **Seat Deduction** |
| Purpose | To show the exact number of seats left after each booking |
| Input | Book for a given number of seats |
| Output | Number of seats booked subtracted from the number of seats available will be shown as seats left. |

Table 6.5

|  |  |
| --- | --- |
| **Module** | **Booking Cancellation** |
| Purpose | To confirm the cancellation of a booked seat in a flight |
| Incorrect Input | Unmatched details of a flight to go forth with cancellation |
| Pass Criteria | A notification appears showing that the entered details do not match. |
| Correct Input | Given details of a booked flight are matched |
| Pass Criteria | Booking is cancelled and a confirmation message is generated. |

Table 6.6

1. **Future Enhancements**

The proposed system is equipped with all the basic functionalities required to book/cancel airline reservations. There is scope for enhancement of the system in the future as well. Some of the factors that are planned to be included in a possible Version 2.0 are:

* SMS confirmations for Booking and Cancellations
* Availability of a new class called Standees that allows passengers of short routes to stand for the duration of the flight.
* Introduction of Multi-Stop flight booking
* Option to select the Food Menu

1. **References**

* [www.makemytrip.com](http://www.makemytrip.com)
* [www.codeacademy.com](http://www.codeacademy.com)
* [www.stackoverflow.com](http://www.stackoverflow.com)
* [www.projectacademy.com](http://www.projectacademy.com)
* [www.100projects.com](http://www.100projects.com)
* Introduction to Software Engineering 8th Edition
* SQL Bible 2nd Edition