

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI – 590018



A MINI-PROJECT REPORT ON

“AIRLINE TICKET BOOKING SYSTEM”

Submitted in partial fulfilment of the requirements of the award of degree of

BACHELOR OF ENGINEERING IN

INFORMATION SCIENCE AND ENGINEERING

Submitted by:

MANOJ M [4VV19IS045]

MADHAVA P ACHAR [4VV19IS041]

UNDER THE GUIDANCE OF

Prof. Yashpal Gupta S

Assistant Professor

Dept of ISE

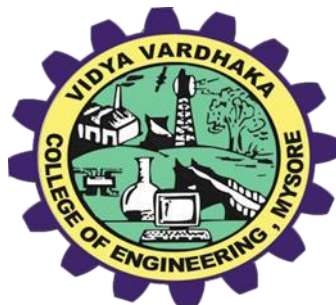
VVCE, Mysuru

Dr. Prema N S

Associate Professor

Dept of ISE

VVCE, Mysuru



2021-2022

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING
VIDYAVARDHAKA COLLEGE OF ENGINEERING MYSURU-570002



Vidyavardhaka College of Engineering
Gokulum 3rd Stage,
Mysuru-570002

Department of Information Science and Engineering

CERTIFICATE

This is to certify that the mini-project report entitled “**AIRLINE TICKET BOOKING SYSTEM**” is a bona fide work carried out by **MANOJ M** and **MADHAVA P ACHAR**, students of 6th semester Information Science and Engineering, **Vidyavardhaka College of Engineering, Mysuru** in partial fulfilment for the award of the degree of Bachelor of Engineering in **Information Science & Engineering of the Visvesvaraya Technological University, Belagavi**, during the academic year **2021-2022**. It is certified that all the suggestions and corrections indicated for the internal assessment have been incorporated in the report deposited in the department library. The report has been approved as it satisfies the requirements in respect of mini-project work prescribed for the said degree.

Signature of the Guide

(Prof. Yashpal Gupta S)

Signature of the Guide

(Dr. Prema N S)

Signature of the HOD

(Dr. A B Rajendra)

Name of the Examiners

- 1.
- 2.

Signature with Date

ACKNOWLEDGEMENT

The Mini project would not have been possible without the guidance, assistance, and suggestions of many individuals. I would like to express our deep sense of gratitude and indebtedness to each one who has helped me to make this project a success.

We heartily thank our beloved Principal, **Dr. B Sadashive Gowda** for his wholehearted support and for his kind permission to undergo the mini project.

We wish to express our deepest gratitude to **Dr. A B Rajendra**, Head of Department, Information Science and Engineering, VVCE, for his constant encouragement and inspiration in taking up this mini project.

We gracefully thank our mini-project guides, **Prof. Yashpal Gupta S**, Assistant Professor **and Dr. Prema N S**, Associate Professor, Dept. of Information Science and Engineering for their encouragement and advice throughout the course of the mini project work.

In the end, we are anxious to offer our sincere thanks to our family members and friends for their valuable suggestions and encouragement.

MANOJ M

MADHAVA P ACHAR

ABSTRACT

The purpose of Airlines ticket booking system is to automate the existing manual system by the help of computerized equipment's and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with. Airlines ticket booking System, as described above, can lead to error free, secure, reliable and fast management system.

It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources. The organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

TABLE OF CONTENTS

CHAPTERS	Page no
1. INTRODUCTION	
1.1. Overview	01
1.2. Objective	01
1.3. Proposed System	01
1.4. Application	01
2. HARWARE AND SOFTWARE REQUIREMENTS	
2.1. Hardware Requirements	02
2.2. Software Requirements	02
2.2.1. Introduction to frontend	02
2.2.2. Introduction to backend	03
2.2.2. Introduction to Text files	04
2.3. Functional Requirements	04
3. DESIGN AND IMPLEMENTATION	
3.1. UML Diagram	05
3.2. Pseudo code	06
4. RESULTS AND SNAPSHOTS	
4.1. Snapshots	08
5. CONCLUSION AND FUTURE ENHANCEMENT	13
6. REFERENCES	14

CHAPTER 1

1. INTRODUCTION

1.1 OVERVIEW

The "Airlines Reservation System" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and, in some cases, reduce the hardships faced by this existing system. Moreover, this system is designed for the particular need of the company to carry out operations in a smooth and effective manner. The application is reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus, by this all it proves it is user-friendly. Airlines Reservation System, as described above, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the record keeping. Thus, it will help organization in better utilization of resources.

1.2 OBJECTIVE

The main objective of the Project on Airlines Reservation System is to manage the details of Airlines Tickets, Flights, Bookings, Customers, Booking Counter. It manages all the information about Airlines Tickets, Vendors, Booking Counter, Airlines Tickets. The project is totally built at administrative end and thus only the administrator is guaranteed the access. The purpose of the project is to build an application program to reduce the manual work for managing the Airlines Tickets, Flights, Vendors, Bookings. It tracks all the details about the Bookings, Customers, Booking Counter.

1.3 PROPOSED SYSTEM

This project aims to develop an Airline ticket booking web application where it provides facility for users to login to the application and it displays the flight data by fetching data from the API and provides facility for booking according to the users need. This system is proposed to carry out a smooth and simple operation in effective manner

1.4 APPLICATION

This Airline ticket booking system is designed for all the users who all need to travel in airlines.

CHAPTER 2

2. HARDWARE AND SOFTWARE REQUIREMENTS

2.1 HARDWARE REQUIREMENTS

- Processor Core i3
- Minimum Disk Drive 500GB
- RAM 1GB

2.2 SOFTWARE REQUIREMENTS

- Operating system: Windows 7 or above
- Frontend: EJS, CSS, JavaScript
- Backend: Node.js, Express.js, API's
- File type user: Text file

2.2.1 INTRODUCTION TO FRONTEND

EJS

EJS or Embedded JavaScript Templating is a templating engine used by Node.js. Template engine helps to create an HTML template with minimal code. Also, it can inject data into HTML template at the client side and produce the final HTML. EJS is a simple templating language which is used to generate HTML markup with plain JavaScript. It also helps to embed JavaScript to HTML pages. The thing we need to do is to set EJS as our templating engine with Express which is a Node.js web application server framework, which is specifically designed for building single-page, multi-page, and hybrid web applications. It has become the standard server framework for node.js. [1]

CSS

Cascading Style Sheets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS handles the look and feel part of a web page. Using CSS, you can control the colour of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colours are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects. CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. [2]

JavaScript

JavaScript often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. Over 97% of websites use JavaScript on the client side for web page behaviour, often incorporating third-party libraries. All major web browsers have a dedicated JavaScript engine to execute the code on users' devices. JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object orientation, and first-class functions. It is multi-paradigm, supporting event driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model. [3]

2.2.2 INTRODUCTION TO BACKEND

Node.js

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

Npm bundled with Node.js. It runs on the command npm. It is a package manager that downloads packages into a node_modules folders. You call the downloaded packages through `const libraryModule = require("libraryname")`. [4]

Express.js

Express is a fast, assertive, essential and moderate web framework of Node.js. You can assume express as a layer built on the top of the Node.js that helps manage a server and routes. It provides a robust set of features to develop web and mobile applications. [5]

Core features of Express framework:

- It can be used to design single-page, multi-page and hybrid web applications.
- It allows to setup middleware's to respond to HTTP Requests.
- It defines a routing table which is used to perform different actions based on HTTP method and URL and allows pages to render in the browser.

API's

Application Programming Interfaces (APIs) are constructs made available in programming languages to allow developers to create complex functionality more easily. They abstract more complex code away from you, providing some easier syntax to use in its place.

As a real-world example, think about the electricity supply in your house, apartment, or other dwellings. If you want to use an appliance in your house, you plug it into a plug socket and it works. You don't try to wire it directly into the power supply — to do so would be really inefficient and, if you are not an electrician, difficult and dangerous to attempt. [6]

2.2.3 INTRODUCTION TO TEXT FILES

A file with .TXT extension represents a text document that contains plain text in the form of lines. Paragraphs in a text document are recognized by carriage returns and are used for better arrangement of file contents. A standard text document can be opened in any text editor or word processing application on different operating systems. All the text contained in such a file is in human-readable format and represented by sequence of characters. Text files can store large amount of data as there is no limitation on the size of contents. However, text editors opening such large files need to be smart for loading and displaying these. Almost all operating systems come with text editors that allow you to create and edit text files. For example, Windows OS comes with Notepad and WordPad for this purpose. Similarly, MacOS comes with TextEdit for creating and editing Text Documents. There are, however, other free text editors available as well over the internet that provide you the capability to work with Text Documents like Notepad++ which is far more advanced in terms of functionality.

2.3 FUNCTIONAL REQUIEMENTS

- System needs store information about new entry of Airlines Tickets.
- System needs to help the internal staff to keep information of Flights and find them as per various queries.
- System needs to maintain quantity record.
- System needs to keep the record of Bookings.
- System needs to update and delete the record.
- System also needs a search area
- It also needs a security system to prevent data

CHAPTER 3

3. DESIGN AND IMPLEMENTATION

3.1 UML Diagram

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

- UML stands for **Unified Modelling Language**.
- UML is different from the other common programming languages such as C++, Java, COBOL, etc.
- UML can be described as a general-purpose visual modelling language to visualize, specify, construct, and document software system.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard.

Sequence diagram

The sequence diagram represents the flow of messages in the system and is also termed as an event diagram. It helps in envisioning several dynamic scenarios. It portrays the communication between any two lifelines as a time-ordered sequence of events, such that these lifelines took part at the run time. In UML, the lifeline is represented by a vertical bar, whereas the message flow is represented by a vertical dotted line that extends across the bottom of the page. It incorporates the iterations as well as branching.

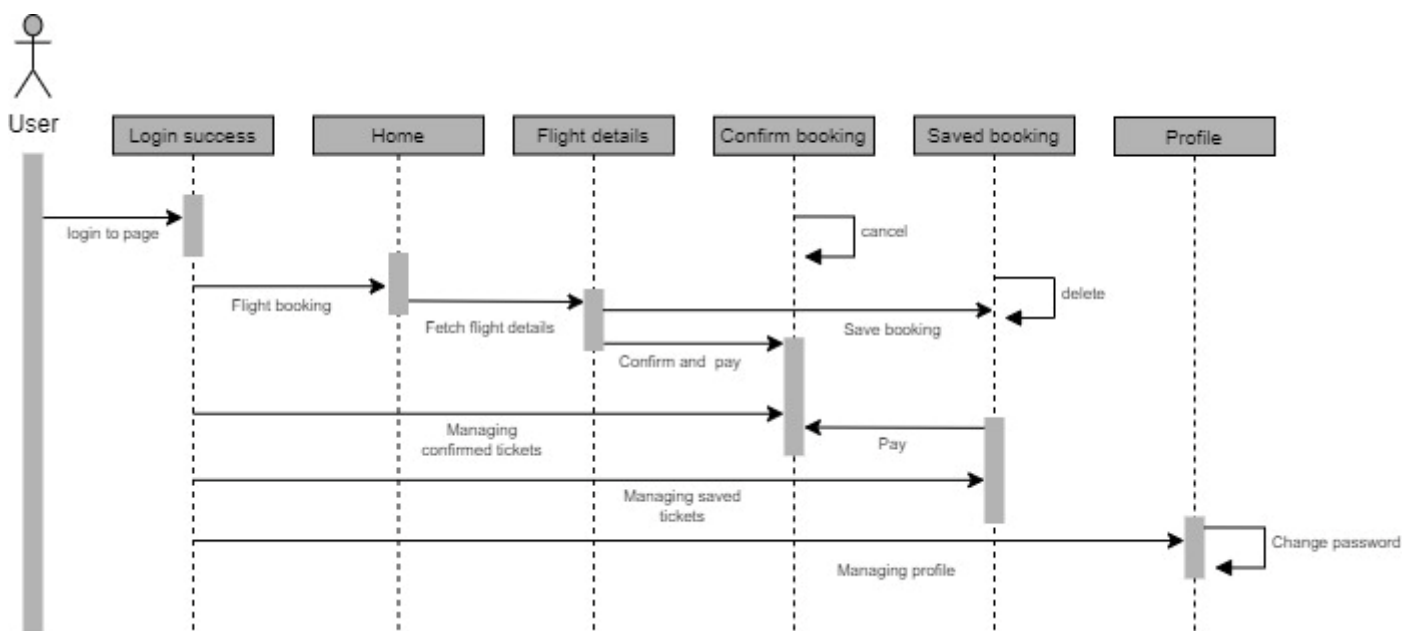


Fig 3.1.1 Sequence diagram of User

3.2 PSUEDO CODE

```

const getDataFromFile = (address, defaultList) => {
  let data = new String();
  try {
    data = fs.readFileSync(address, { encoding: "utf8", flag: "r" });
  } catch (error) {
    fs.writeFileSync(address, JSON.stringify(defaultList));
    data = JSON.stringify(defaultList);
  }

  return data;
};

const fetchFlights: (req, res) => {
  var session = req.session;
  if (session.user) {
    const { from, to, date } = req.body;
    if (from && date && to) {
      var mydate = new Date(date);
      mydate = year + "/" + month + "/" + day;
      const felDetUrl =
        "https://api.flightstats.com/flex/schedules/rest/v1/json/from/${from}/to/${to}/arriving/${mydate}?appId=2b3f4415&appKey=b7b1c948b008fa981b0a4c7620c44b2f&Access-Control-Allow-Origin=*";
      var FligOptions = {
        method: "GET",
        url: felDetUrl,
        headers: {
          appId: "2b3f4415",
          appKey: "b7b1c948b008fa981b0a4c7620c44b2f",
        },
      };
      request(options, function (error, flgResponse) {
        if (error) throw new Error(error);
        flightName = JSON.parse(flgResponse.body);
        flightNameArr.push(flightName.airline.name);
        if (flightNameArr.length === data.scheduledFlights.length) {
          res.status(200).render("user/booking.ejs", {
            title: "Booking",
            length: length,
            flights,
            flightNameArr,
            main: false,
            adminDash: false,
            userDash: true,
            name: session.user,
            from,
            to,
            mydate: date,
          });
        }
      });
    }
  }
};

} else {
  res.status(401).redirect("/userdashboard");
}
}
}

```

```

function create_saved_UUID() {
  var dt = new Date().getTime();
  var uuid = "xxx-xyx-yxx".replace(/[xy]/g, function (c) {
    var r = (dt + Math.random() * 16) % 16 | 0;
    dt = Math.floor(dt / 16);
    return (c == "x" ? r : (r & 0x3) | 0x8).toString(16);
  });
  return uuid;
}

function checkValidity(date) {
  const today = new Date();
  if (today.toDateString() === date.toDateString()) {
    return 0;
  } else if (today < date) {
    return 2;
  } else {
    return 1;
  }
}

const userproj: (req, res) => {
  var session = req.session;
  const id = session.user;
  const histArr = [];
  if (session.user) {
    let data = getDataFromFile("./files/userInfo.txt", defaultList);
    let histData = getDataFromFile("./files/booked.txt", defaultList);

    data = JSON.parse(data);
    histData = JSON.parse(histData);

    for (let i = 0; i < data.length; i++) {
      if (data[i].id === id) {
        for (let j = 0; j < histData.length; j++) {
          histArr.push(histData[j]);
        }
        console.log(histArr);
        res.status(200).render("user/userprofile.ejs", {
          title: "User Profile",
          main: false,
          adminDash: false,
          userDash: true,
          user: session.user,
          name: data[i].name,
          phone: data[i].phone,
          histArr: histArr,
          length: histArr.length,
        });
        return;
      }
    }
    res.status(404).send("User not found");
  } else {
    res.redirect("/logout");
  }
}

```

CHAPTER 4

4. RESULTS AND SNAPSHOTS

4.1 SNAPSHOTS

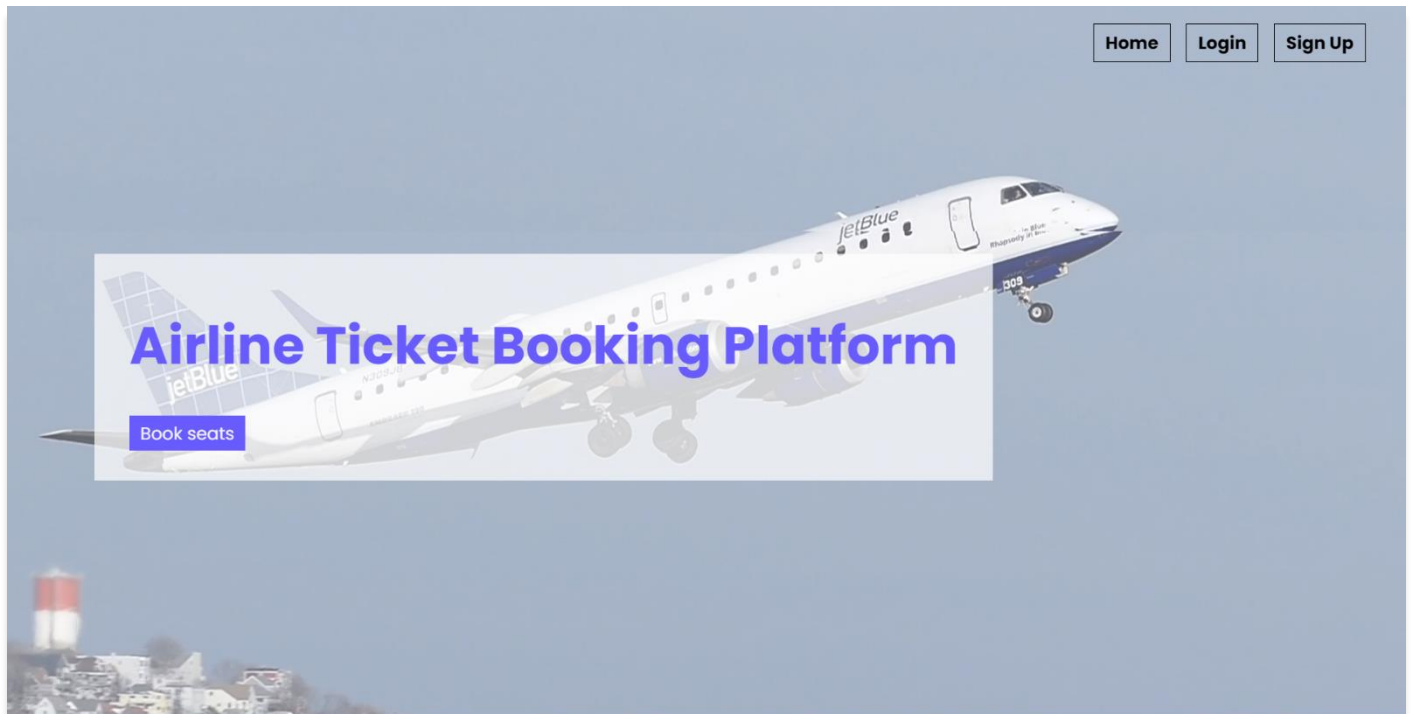
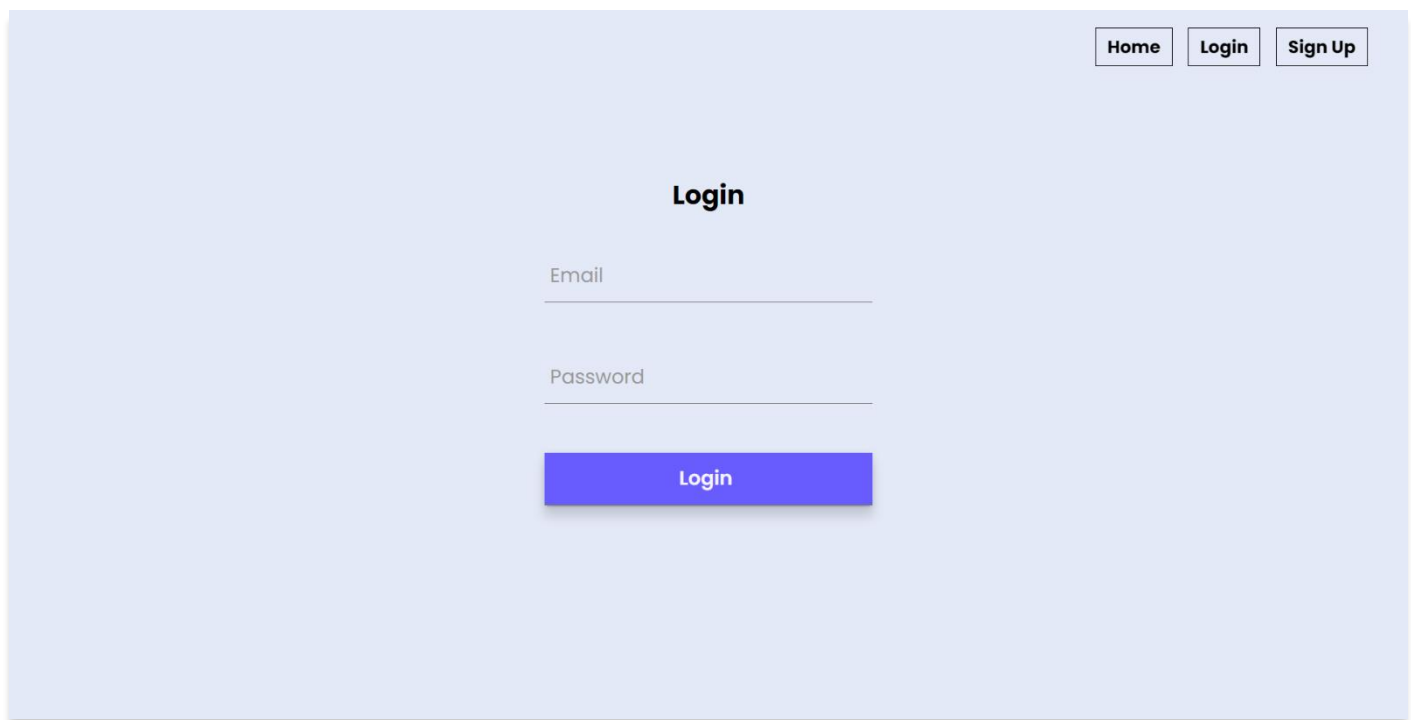


Fig 4.1.1 Home page

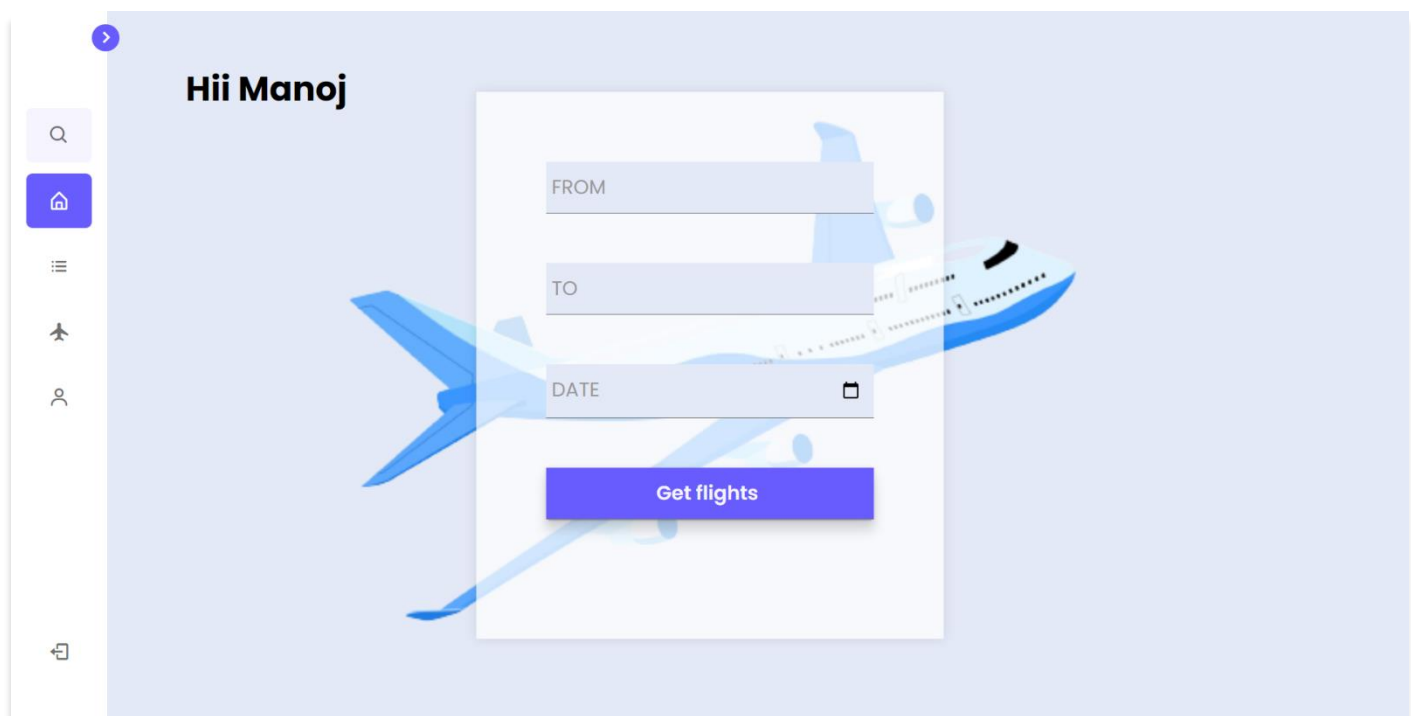
The image shows the 'Register' page of the 'Airline Ticket Booking Platform'. The background is a solid light blue. In the top right corner, there are three small, light blue buttons with black text: 'Home', 'Login', and 'Sign Up'. In the center of the page, the word 'Register' is written in a bold, black font. Below 'Register', there are four text input fields, each with a label to its left: 'Name', 'Email', 'Phone', and 'Password'. Each label is in a light gray font. Below the 'Password' field, there is a large, solid blue button with the word 'Login' in white text.

Fig 4.1.2 Register page



The login page features a light blue background. At the top right, there are three buttons: 'Home', 'Login', and 'Sign Up'. In the center, the word 'Login' is displayed in bold. Below it, there are two input fields: 'Email' and 'Password'. At the bottom center, there is a blue button labeled 'Login'.

Fig 4.1.3 Login Page



The user dashboard has a light blue background. On the left, there is a vertical sidebar with a search icon, a home icon, a menu icon, an airplane icon, a user profile icon, and a clipboard icon. The main area displays 'Hii Manoj' at the top. Below this, there is a white card with a blue airplane illustration in the background. The card contains three input fields: 'FROM', 'TO', and 'DATE'. At the bottom of the card is a blue button labeled 'Get flights'.

Fig 4.1.4 User dashboard

Book Tickets

From : BLR To : VNS Date : 2022-07-15

- 1. IndiGo**
Code:897 Date:2022-07-15 Departure:05:30 Arrival:07:55
Book : 1999.00/-
- 2. FlexFlight**
Code:403 Date:2022-07-15 Departure:16:30 Arrival:19:00
Book : 1999.00/-
- 3. Go First**
Code:8762 Date:2022-07-15 Departure:16:30 Arrival:19:00
Book : 1999.00/-

Fig 4.1.5 Booking page

Confirm Details

Flight name : **IndiGo**
 Flight code : **897**
 Departure date : **2022-07-15**
 Departure time : **05:30**
 Arrival time : **07:55**

Passengers

Dhruva

Passengers
 Number of Passengers:
 1

Payment Details
 Total payment:
 1999

Confirm And Pay Confirm And Save Reset

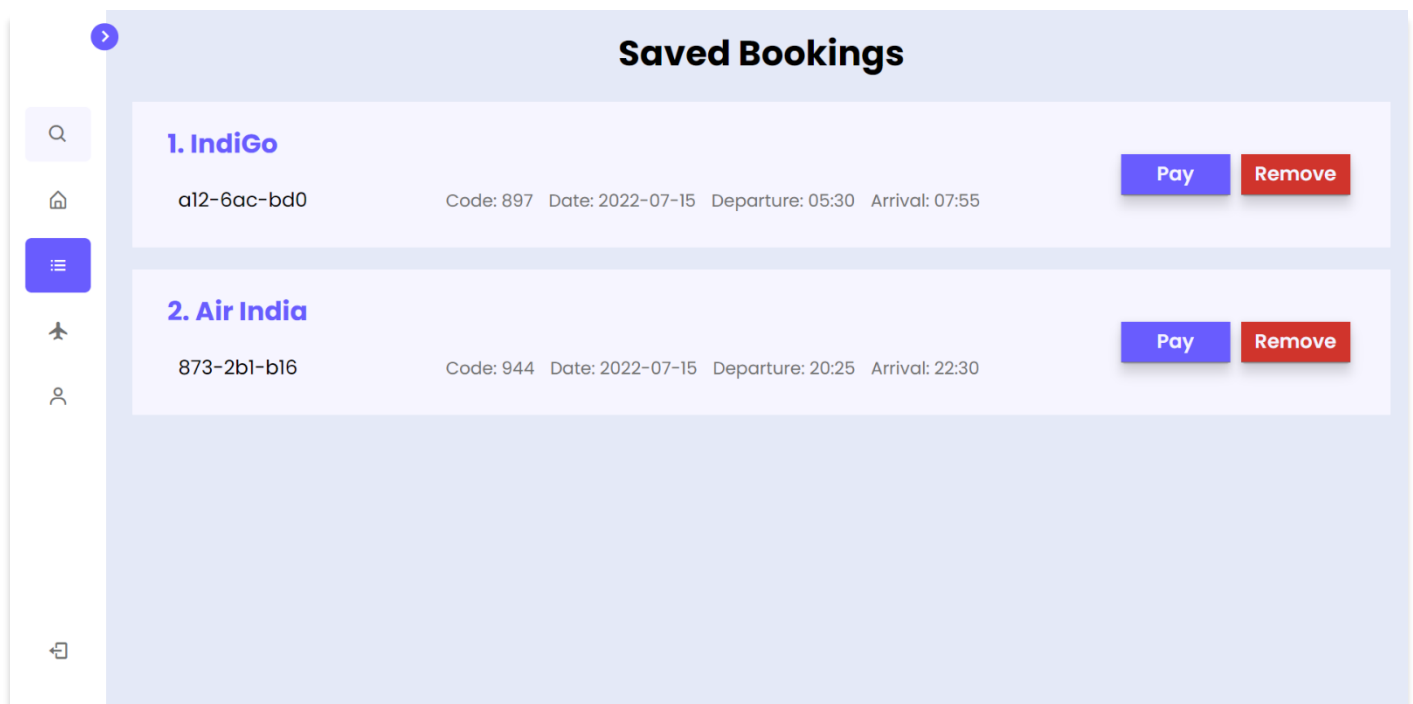
Fig 4.1.6 Confirming booking



The 'Confirmed Bookings' interface features a sidebar with navigation icons (search, home, menu, airplane, user, and a back icon) and a main content area with a title bar and four booking cards. Each card displays the airline name, a unique ID, a sequence number, a date, a time, a status indicator, and a 'Cancel' button.

Airline	ID	Seq	Date	Time	Status	Action
1. FlexFlight	4c2-2fa-9aa-bdb	6288	2022-07-14	05:50 07:30	Today	Cancel
2. Air India	725-3d8-18c-91c	946	2022-07-12	23:55 02:10	Done	Cancel
3. IndiGo	c90-778-d80-99f	356	2022-07-15	17:25 18:30	Upcoming	Cancel
4. KLM	6cf-59b-b80-86b	861	2022-07-22	08:50 11:10	Upcoming	Cancel

Fig 4.1.7 Confirmed Bookings



The 'Saved Bookings' interface features a sidebar with navigation icons (search, home, menu, airplane, user, and a back icon) and a main content area with a title bar and two booking cards. Each card displays the airline name, a unique ID, a code, a date, a departure time, an arrival time, and 'Pay' and 'Remove' buttons.

Airline	ID	Code	Date	Departure	Arrival	Actions
1. IndiGo	a12-6ac-bd0	897	2022-07-15	05:30	07:55	Pay, Remove
2. Air India	873-2b1-b16	944	2022-07-15	20:25	22:30	Pay, Remove

Fig 4.1.8 Saved booking

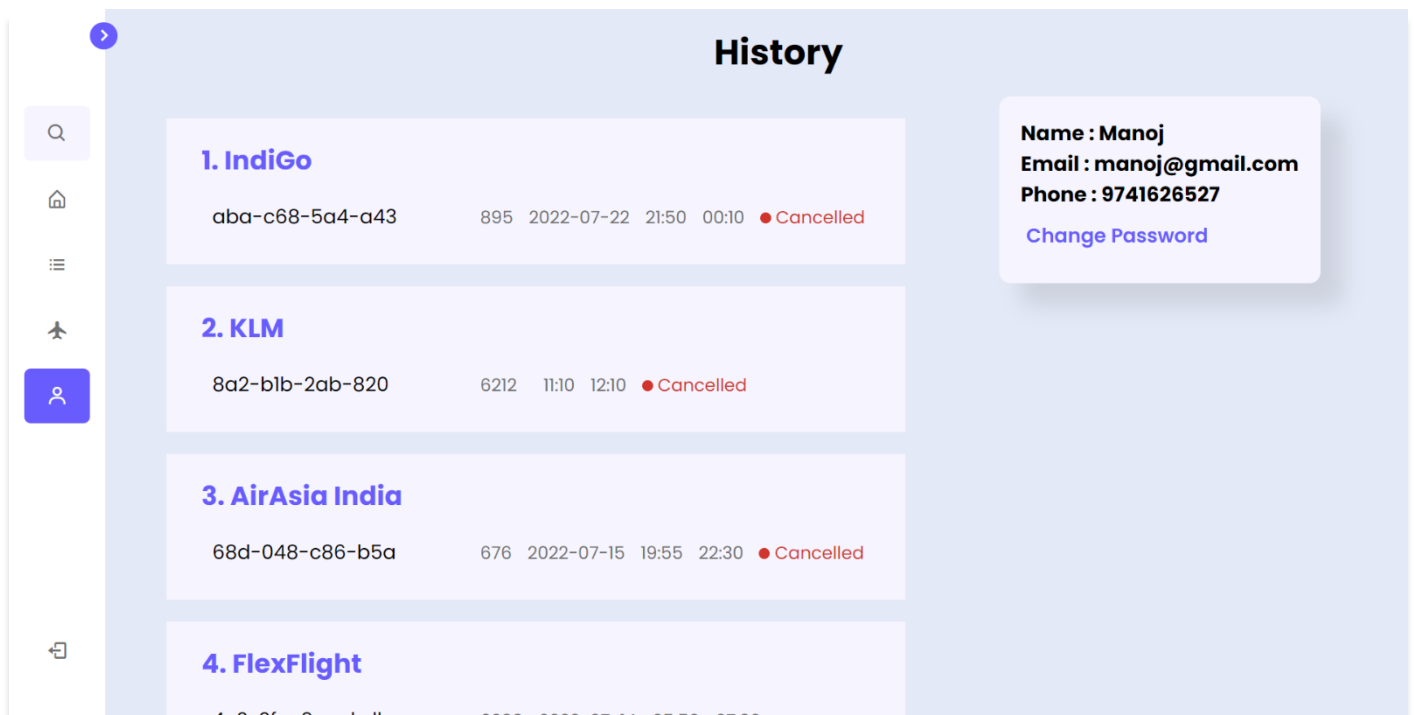


Fig 4.1.9 User profile and history

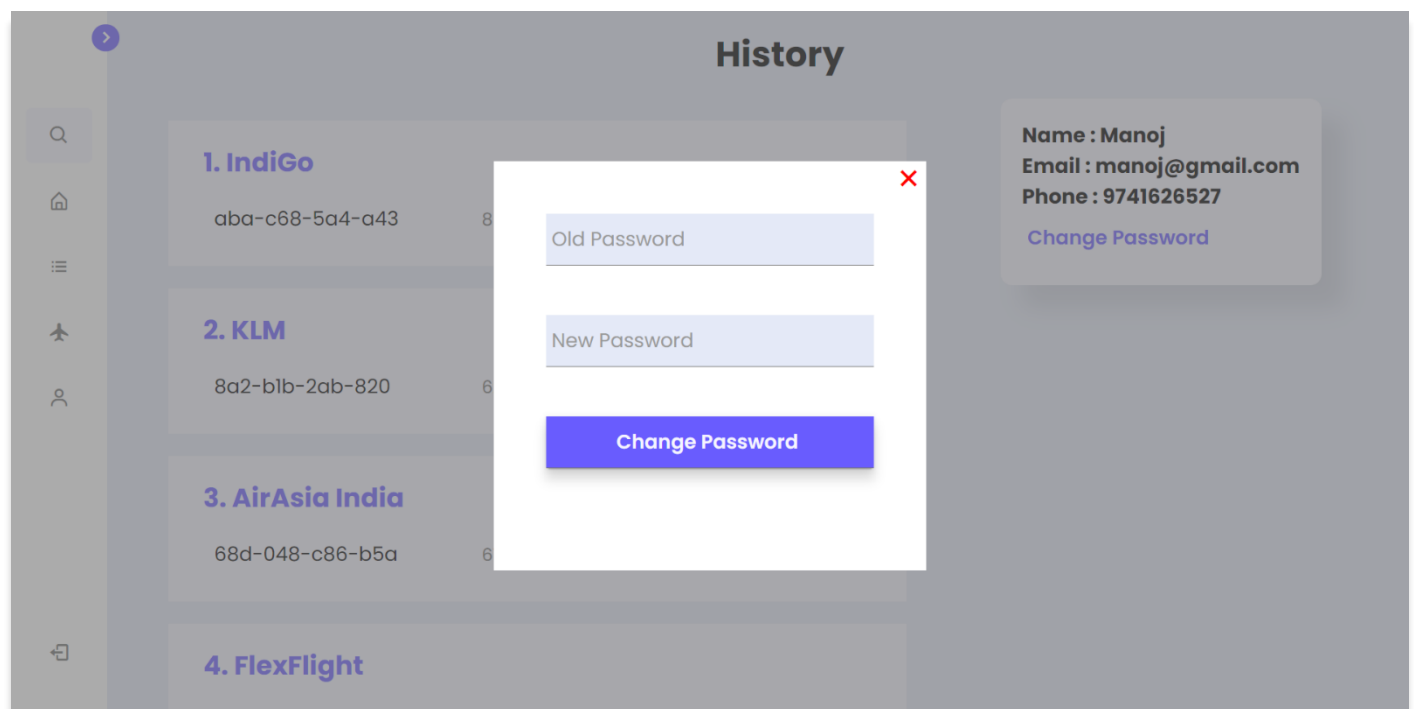


Fig 4.1.10 Changing password

CHAPTER 5

5. CONCLUSION AND FUTURE ENHANCEMENT

5.1 CONCLUSION

Based on the preparation & analysis of Airline Reservation System, it can be concluded that Airline Reservation system is better than the manual System can be used by the ticketing manager to make transactions for the customers regarding flights. The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses.

5.1 FUTURE ENHANCEMENT

Future Scope and further enhancement of the Project the Online Airline Reservation system is the next generation address book which will provide these two basic services like portability, security. The future scope includes expand the technologies like HTML and PHP we can also add new technologies like HTML, pup many more for improving the efficiency of the software. The project will be useful for any schools and colleges with slightly modification. Project is flexible i.e. any change /modification in database may be performing easily. Also, this project could be made web enabled. Assumptions, if any None Assumptions: The user is familiar with basic computer components and operations. Dependencies: The system should work on all systems.

CHAPTER 6

6. REFERENCES

1. EJS: <https://ejs.co/>
2. CSS: <https://developer.mozilla.org/en-US/docs/Web/CSS>
3. JavaScript: <https://developer.mozilla.org/en-US/docs/Web/JavaScript>
4. Node.js: <https://nodejs.org/en/docs/>
5. Express.js: <https://expressjs.com/en/guide/routing.html>
6. Flightstats api's: <https://developer.flightstats.com/>