[AIRLINES RESERVATION SYSTEM]

A Major project submitted for the partial fulfilment of the requirements for the degree of

MASTERS OF COMPUTER APPLICATION

Submitted By

Subhajit Naskar

Roll No.:18871021057 Session:2021-2023

Registration. No.:211880571010057 of 2021-2023

Under The Supervision Of

Mr. Tapan Kumar Chakrabarty

PROJECT GUIDE

Mr. Tapan Kumar Chakrabarty

HOD MCA

Department Of MCA

Techno India (Hooghly – Campus)

Techno India (Hooghly – Campus)

Dharampur, G.T. Road, Near Khadina More

Chinsurah, Hooghly-712101

Website: www.technoindiahooghly.org, Email: info@technoindiahooghly.org

Project Undertaken:

FLY HIGH

Techno India Hooghly - Campus

Dharampur, Shantiniketan, Near Khadinamore
G. T. Road, Chinsurah, Hooghly - 712101
Phone: (033) 2680-2389 / 6565
E-mail: info@technoindiahooghly.org
Website: www.technoindiahooghly.org

Certificate

This is to certify that the project entitled ***"AIRLINES RESERVATION SYSTEM"*** *submitted by* ***Subhajit Naskar,*** *University Roll No:* ***18871021057 and*** *Registration No.:* ***211880571010057*** *of 2021-23 as Assigned Project (major) for the partial fulfilments of degree of Masters of Computer Application is worth of acceptance.*

Mr. Tapan kr Chakrabarty

(Project Guide)

Mr. Tapan kr Chakrabarty

(Head of the Department, MCA)

Techno India Hooghly

ACKNOWLEDGEMENT

It is our pleasure to be indebted to various people, who directly or indirectly contributed in the development of this work and who influenced our thinking, behaviour, and acts during the course of study.

We express sincere gratitude to **Dr. Mrinal Kanti Chakrabarty** worthy **Principal** for providing us an opportunity to undergo this project at **Techno India (Hooghly-Campus).**

We are grateful to our project guide "**Mr. Tapan Kr. Chakraborty**" for the guidance, inspiration and constructive suggestions that helpful us in the preparation of this project.

We are also thankful to my group with whom we have fruitful discussions which have helped us a lot in giving a final shape to the program.

PROJECT ASSOCIATES:

Subhajit Naskar (18871021057)
Parthana Ghosh (18871021049)
Ishita Naskar (18871021054)
Unnati Saha (18871021060)

ABSTRACT

The purpose of the project entitled as "**AIRLINES RESERVATION SYSTEM**" is to computerize the Front Office Management of Airport to develop software which is user friendly simple, fast, and cost – effective. It deals with the collection of passenger's information, journey details, etc. Traditionally, it was done manually. The main function of the system is register and store passengers' details and their others detail and retrieve these details as and when required, and also to manipulate these details meaningfully System input contains passengers' details, journey details, while system output is to get these details on to the screen. The "AIRLINES RESERVATION SYSTEM" can be entered using a username and password. It is accessible either by an administrator or passengers. Only they can add data into the database. The data can be retrieved easily. The data are well protected for personal use and makes the data processing very fast.

INDEX

Sr. No.	Table OF Content		Page No
1.)	Project name and description		4
2.)	Profile OF the Problem		5
3.)	Existing System		6-7
	A.)	Introduction	
	B.)	Existing Software	
	C.)	What's new in the system to be developed?	
4.)	Problem Analysis		8-10
	A.)	Product Definition	
	B.)	Feasibility Analysis	
	C.)	Project Plan	
5.)	Software Requirement Analysis		10-11
	A.)	Introduction	
	B.)	Specific Requirement	
6.)	Design		11-20
	A.)	System Design	
	B.)	Detailed Design	
7.)	Testing		20
	A.)	Functional Testing	
	B.)	Structural Testing	
	C.)	Levels of testing	
	D.)	Testing of the Project	
8.)	implementation		24
	A.)	Implementation of the project	
9.)	User Manual		27
	A.)	The complete guide for the project developed	
10.)	Source code		27
11.)	bibliography		40

1. Project Name and Description

AIRLINES RESERVATION SYSTEM

The Airline Reservations System (ARS) was one of the earliest changes to improve efficiency. ARS eventually evolved into the Computer Reservations System (CRS). A Computer Reservation System is used for the reservations of a particular airline and interfaces with a Global Distribution System (GDS) which supports travel agencies and other distribution channels in making reservations for most major airlines in a single system.

Airline Reservations Systems contain airline schedules, fare tariffs, passenger reservations and ticket records. An airline's direct distribution works within their own reservation system, as well as pushing out information to the GDS. A second type of direct distribution channel are consumers who use the internet or mobile applications to make their own reservations.

INVENTORY MANAGEMENT

An airline’s inventory contains all flights with their available seats. The inventory of an airline is generally divided into service classes (e.g., First, Business or Economy class) and up to 26 booking classes, for which different prices and booking conditions apply. Inventory data is imported and maintained through a Schedule Distribution System over standardized interfaces. One of the core functions of the inventory management is the inventory control. Inventory control steers how many seats are available in the different booking classes, by opening and closing individual booking classes for sale. In combination with the fares and booking conditions stored in the Fare Quote System the price for each sold seat is determined.

DISPLAY AND RESERVATION

Users access an airline's inventory through an availability display. It contains all offered flights for a particular city-pair with their available seats in the different booking classes. This display contains flights which are operated by the airline itself as well as code share flights which are operated in co-operation with another airline. If the city pair is not one on which the airline offers service it may display a connection using its' own flights or display the flights of other airlines.

The availability of seats of other airlines is updated through standard industry interfaces. Depending on the type of co-operation it supports access to the last seat (Last Seat Availability) in real-time. Reservations for individual passengers or groups are stored in a so-called Passenger Name Record (PNR). Among other data, the PNR contains personal information such as name, contact information or special services requests (SSRs).

e.g., for a vegetarian meal, as well as the flights (segments) and issued tickets. Some reservation systems also allow to store customer data in profiles to avoid data re-entry each time a new reservation is made for a known passenger.

Fare Quote and Ticketing

The Fares data store contains fare tariffs, rule sets, routing maps, class of service tables, and some tax information that construct the price - "the fare". Rules like booking conditions (e.g. minimum stay, advance purchase, etc.) are tailored differently between different city pairs or zones, and assigned a class of service corresponding to its appropriate inventory bucket. Inventory control can also be manipulated manually through the availability feeds, dynamically controlling how many seats are offered for a particular price by opening and closing particular classes.

2. Profile of the problem

The web based "airline reservation system" project is an attempt to stimulate the basic concepts of airline reservation system. The system enables the customer to do the things such as search for airline flights for two travel cities on a specified date, choose a flight based on the details, reservation of flight and cancellation of reservation.

The system allows the airline passenger to search for flights that are available between the two travel cities, namely the "Departure city" and "Arrival city" 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, address, city, state, and credit card number and contact number. Then it checks the validity of card and book the flight and update the airline database and user database.

Disadvantages in existing system:->

- Time consuming
- Possibly of loosing data
- Lack of security
- Difficulties in maintaining records
- Human error will be frequent
- searching the records manually leads time consuming

3. Existing System

Introduction

In the existing system all the data are stored manually to an excel sheet and filed accordingly in a filing cabinet.

SPECIFIC OBJECTIVES: -

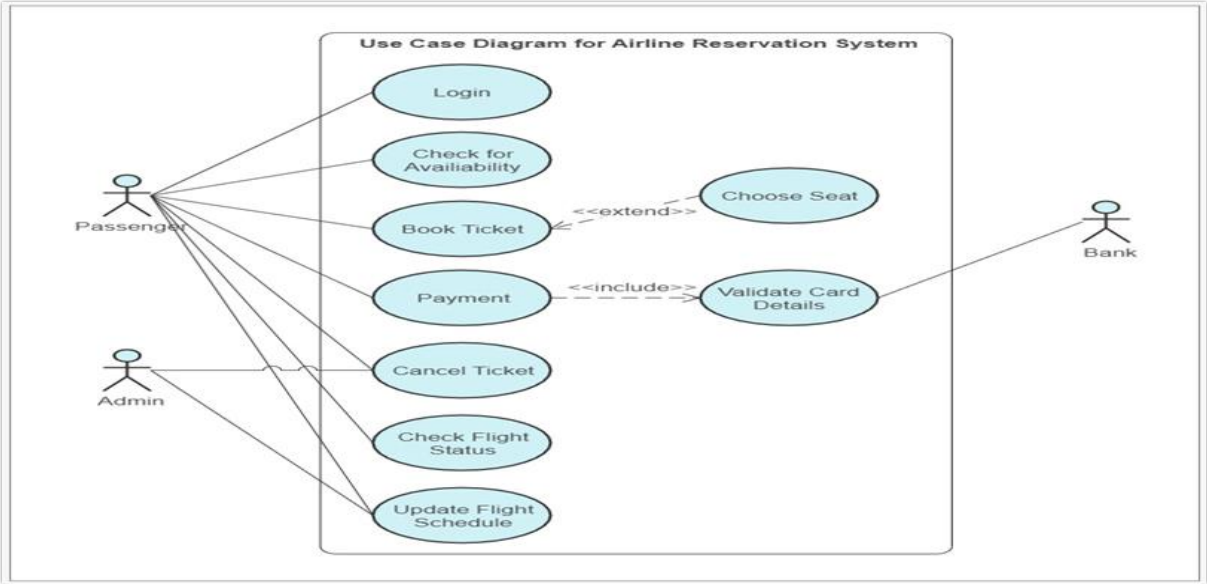
a. Maintaining safe records
b. Should be easier to find for a record in the database
c. The loss of data in any corruption of any files in the system will be avoided due to any natural case
d. Minimizing errors of the information recorded in the system while entering to the system
e. The new system should be user friendly
f. Data entry should be fast
g. There should be a method of keeping the information from unauthorized users.

System Design: ERD along with table Structure

• **Name:**	
• **Address:**	
• **Phone Number:**	
• **Date OF Birth:**	
• **Passport No:**	
• **Nationality:**	
• **Destination:**	
• **Airline:**	
• **Day of Departure:**	
• **Time:**	
• **Time of Arrival:**	

When the customer asks to book a flight at first the booking form is been filled with his/her full:->
Name, Address, National ID, Email, Address, Contact Number, Destination, booking date and the Retuned date as shown above the form. After that the document is been signed by the office staff.

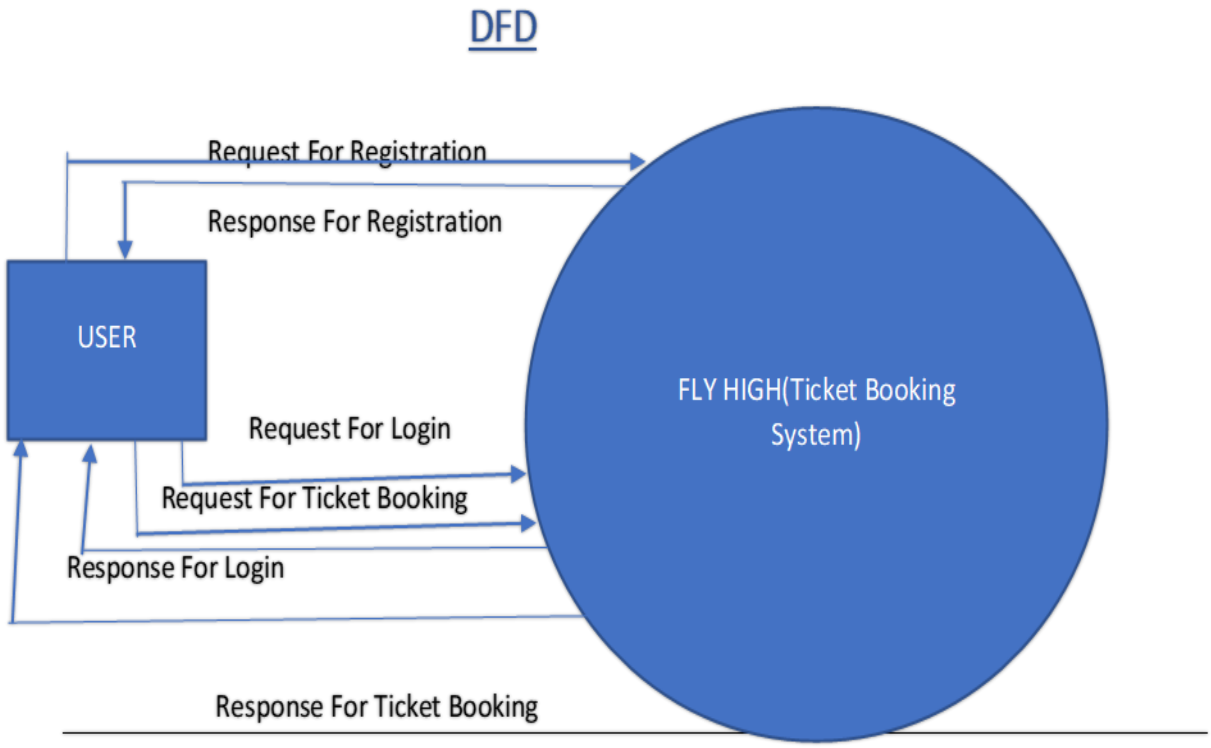
USECASE DIAGRAM



GNATT CHART

Task	Nov'22	Dec'22	Jan'23	Feb'23	Mar'23	Apr'23	May'23
Planning							
Research							
Design and development							
Implementation							
Follow-up and action							

System Design: Data Flow Diagram



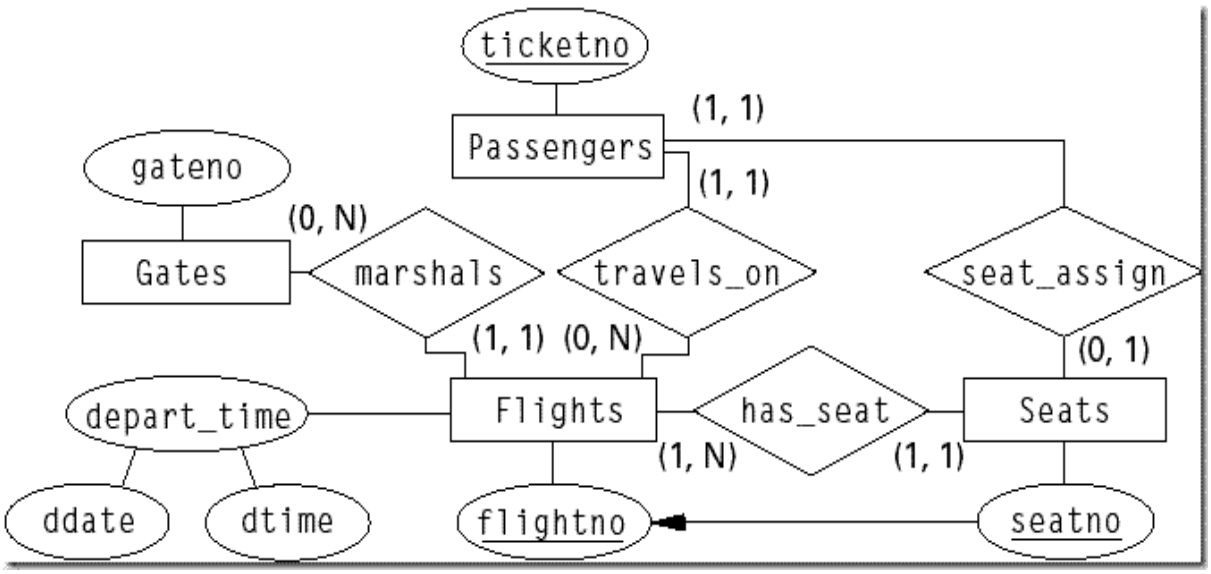
- **Exiting software**

METHODOLOGIES

HARDWARE:
PROCESSOR: PENTUIUM IV 2.6 GHz
RAM: 512MB DD RAM
MONITOR: 15" COLOR
HARD DISK :250 GB
CDDRIVE: LG52X
KEYBOARD: STANDARD 102 KEYS
MOUSE: OPTICAL MOUSE

SOFTWARE:
FORNT END: HTML, CSS, JAVA SCRIPT, REACT, NODEJS
BACKEND: NODE JS, MONGO DB
OPERATING SYSTEM: WINDOS, WINDOWS 10

- **DFD for Present system**



- **What's new in the system to be developed?**

APPLICABILITY

- This project solves the problem of the traditional reservation system.
- With certain changes it can be applicable on any online reservation field.
- One of the most benefits in today's life is that reservation can be made from any place of the world.
- The user need not to be present the physically to draw a reservation slip. It will automatically do by the system

ADVANTAGES

1. It easy to learn and adjust to the system
2. This system does not require the staff to be highly educated
3. The requirements to tackle this job may be limited to
4. Willing to work long hours
5. Data is not easily lost
6. It easy to manage the system due to the high number of staff working

4. Problem Analysis

- **Product definition**

1. Plane **type**: This defines the physical type of the plane. It dictates the capacity of first, executive, business and economy seats that a flight can have.

2. **Airport**: An airport consists of a name, the city it is in, and its airport id.

3. **Flight**: A flight is identified by its flight ID. A flight denotes a unique "plane", i.e., one which is scheduled to run at a certain time, from one place to another. A flight runs over a set of routes.

4. **Route**: A route is simply a tuple of airports: (Start Airport, End Airport), and every route has a unique route id. A flight runs over a route only if it runs from the start airport to the end airport, possibly halting in between at other airports. A route is elementary for a flight if the flight runs nonstop from the start airport to the end airport.

5. **Ticket**: A ticket is uniquely identified by a ticket ID. The ticket may be a passenger ticket or a cargo ticket, and can be booked under a passenger profile or a user profile. A ticket is booked on a flight for a route that the flight is associated with. A passenger ticket contains details about the passenger, and a cargo ticket about a cargo. The

6. **Scheme**: A scheme consists of discount percentages on various classes awarded on certain flights, and for certain people or round trips. Scheme ids have a type code defining what they

are valid for, and a period code showing whether they are valid as of now or no. A scheme is defined for a flight and for a particular route.

7. **Official**: An official is a person who can book tickets for others, and can find retrieve the complete list of passengers boarding a flight. An official works at an airport.

8. **Profile**: A profile denotes that a person has been verified to be genuine and can book tickets/ execute certain queries.

Feasibility Analysis

1. **Flight Route Scheme**: This is a ternary relation that says that a flight runs over a route using a particular scheme. The scheme can be null, but not the flight id and route id. The attributes in this include:

a. Fare for the flight between the two stops given by the route specified.
b. Any scheme valid on this flight, for this route
c. Number of booked seats on this flight, route.
d. A flag value indicating whether this is an elementary or complex route for this flight.

2. **Flight Route Ticket**: This ternary relation says that a ticket is booked on a certain flight over one route that the flight allows. It has only the primary keys of each entity.

3. **Profile Ticket relation**: This consists of two relations: user profile related to ticket and the official profile related to ticket. This is done to keep the user and official profiles separate and independent from each other.

Project Plan

The plan of this project is to make a software which is:->>

- Interpreted and high performance
- Distributed
- Dynamic
- Secure

Services Reservation in Airlines Abstract: -
This Abstract report on a study about examining airlines' reservation services.

An apparatus, method, and program for determining a price of an option to purchase an airline ticket, and for facilitating the sale & exercise of those options.

Pricing of the options may be based on departure location criteria, destination location criteria, and travel criteria. The attributes selected for examination included….
(1) Requirements of reservation services.
(2) Provision of extra benefits.

(3) Factors affecting reservation time.
(4) Provision of additional services/facilities.

Empirical results indicated that some airlines did not provide all components in the chosen attributes and that airlines in these regions differed significantly in certain dimensions of the chosen attributes.

5. Software requirement Analysis

Introduction

Requirements analysis is critical to the success of a development project. Requirements must be documented, actionable, measurable, testable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design. Requirements can be architectural, structural, behavioural, functional, and non-functional.

General Description

Visual Basic .NET (**VB.NET**) is an object-oriented computer programming language that can be viewed as an evolution of the classic Visual Basic (VB) which is implemented on the .NET Framework. Microsoft currently supplies two major implementations of Visual Basic: Microsoft Visual Studio, which is commercial software and Microsoft Visual Studio Express, which is free of charge.

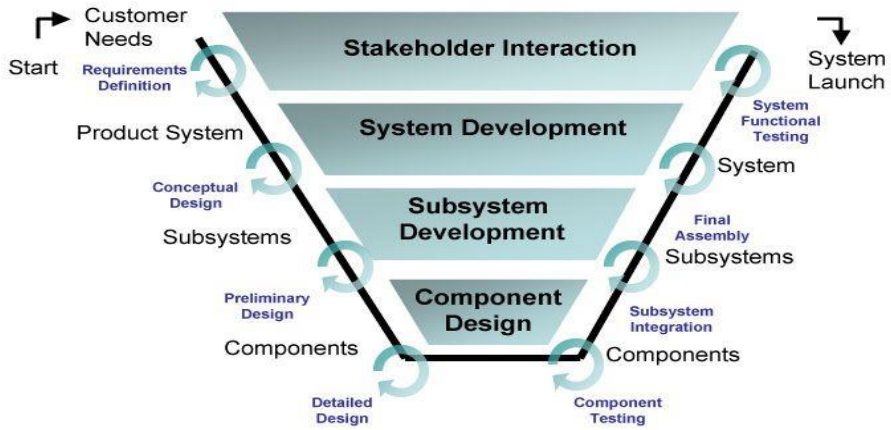
Specific Requirements

- **System**
  - Windows XP Service Pack 2 or above (for 2010 release, Service Pack 3)
  - Windows Server 2003 Service Pack 1 or above
  - Windows Server 2003 R2 or above
  - Windows Vista
  - Windows Server 2008
  - Windows 7
  - Windows 10
  - Linux with Mono (only works with .NET 2.0 applications)
- **Hardware**
  - Minimum: 1.6 GHz CPU, 384 MB RAM, 1024×768 display, 5400 RPM hard disk
  - Recommended: 2.2 GHz or higher CPU, 1024 MB or more RAM, 1280×1024 display, 7200 RPM or higher hard disk

6. Design

System Design

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. One could see it as the application of systems theory to product development.

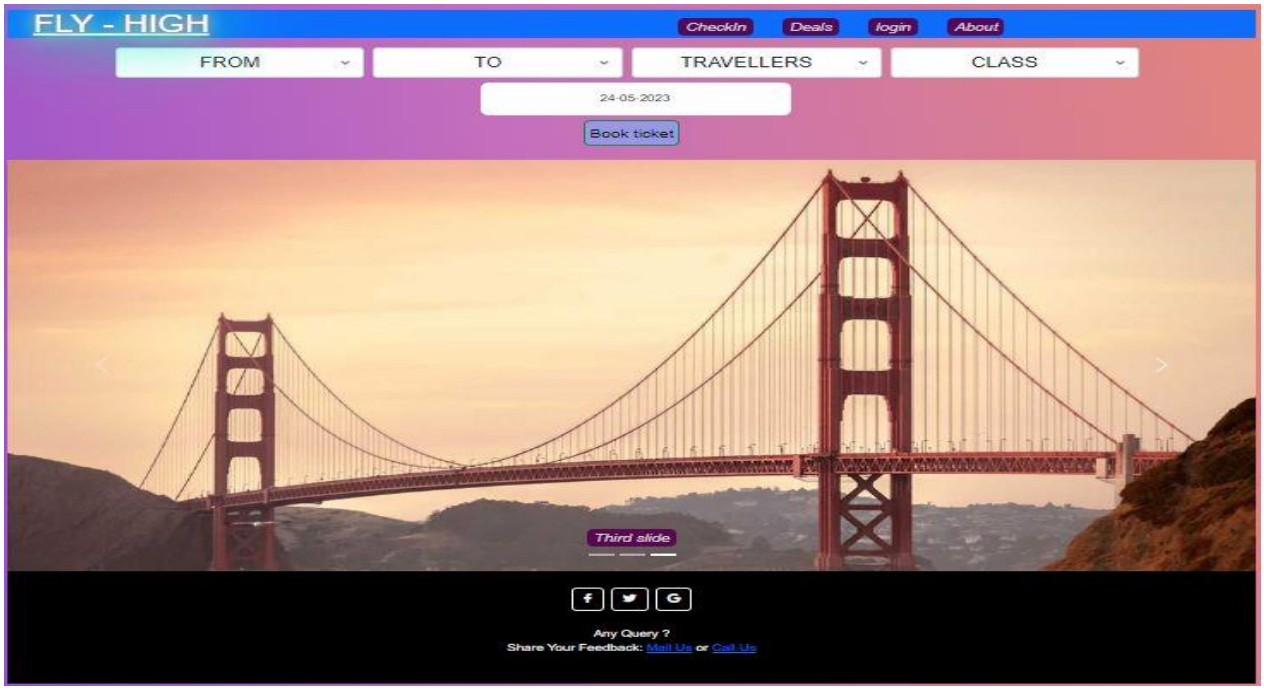


There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development, then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user

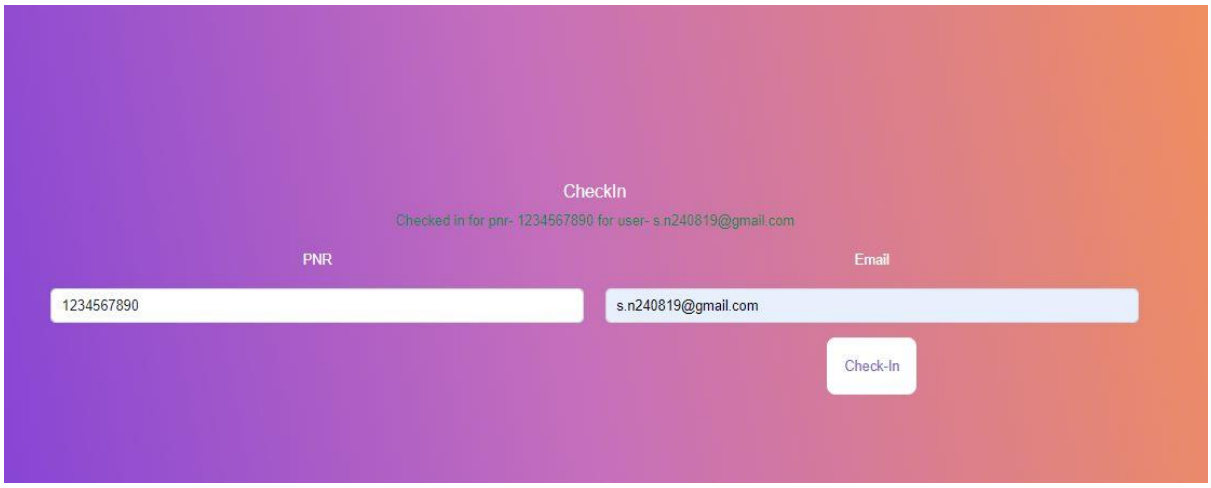
Detailed design: -

Screen shots:

HOME PAGE: -



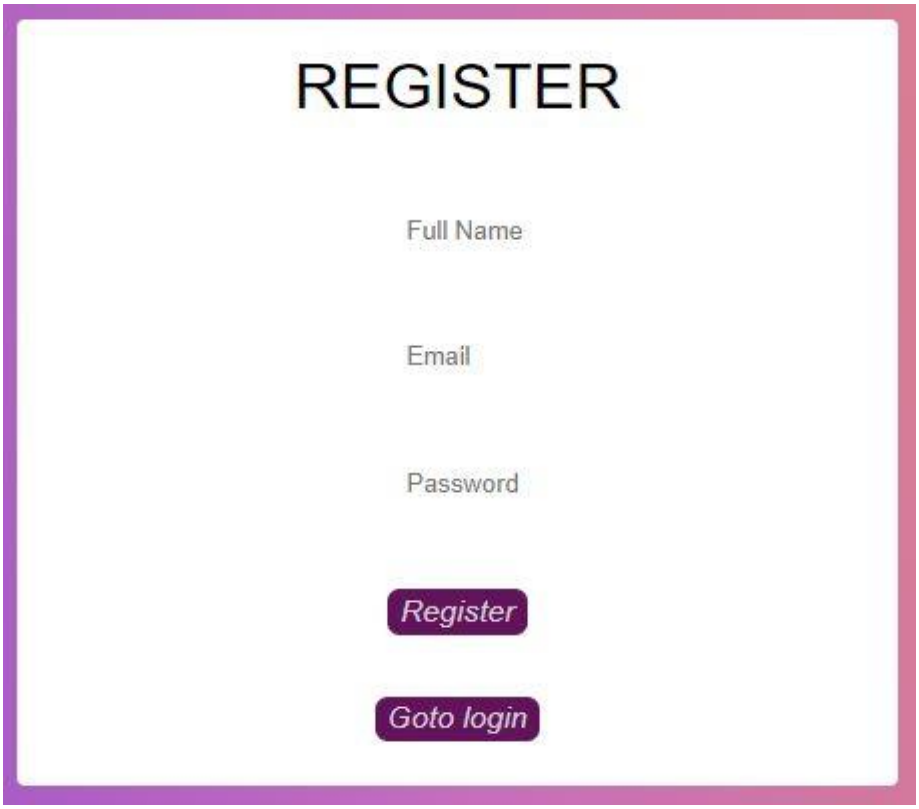
CHECK-IN PAGE: -



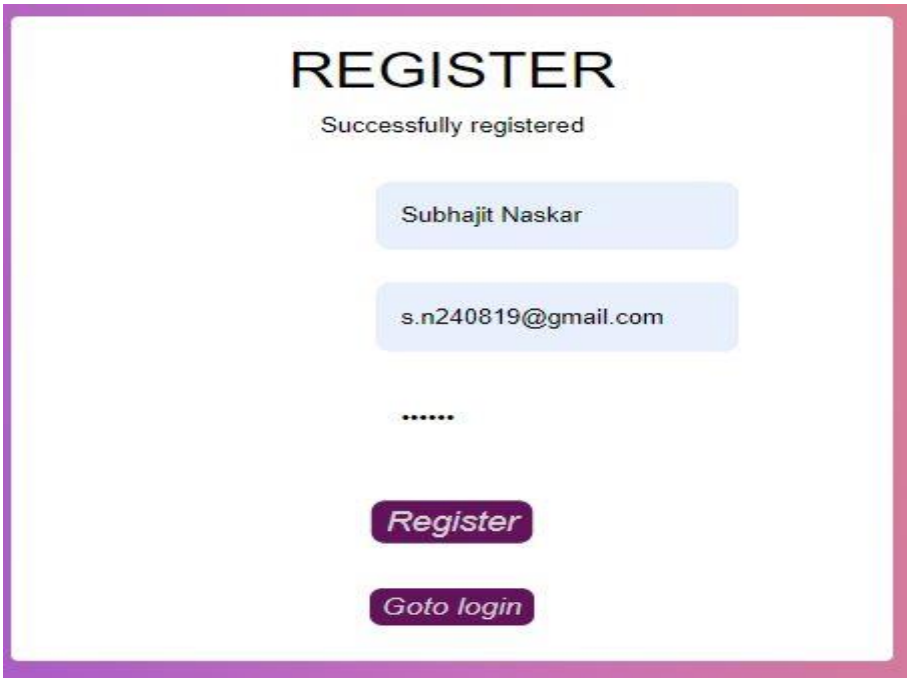
DEALS PAGE: -

LOG-IN PAGE: -

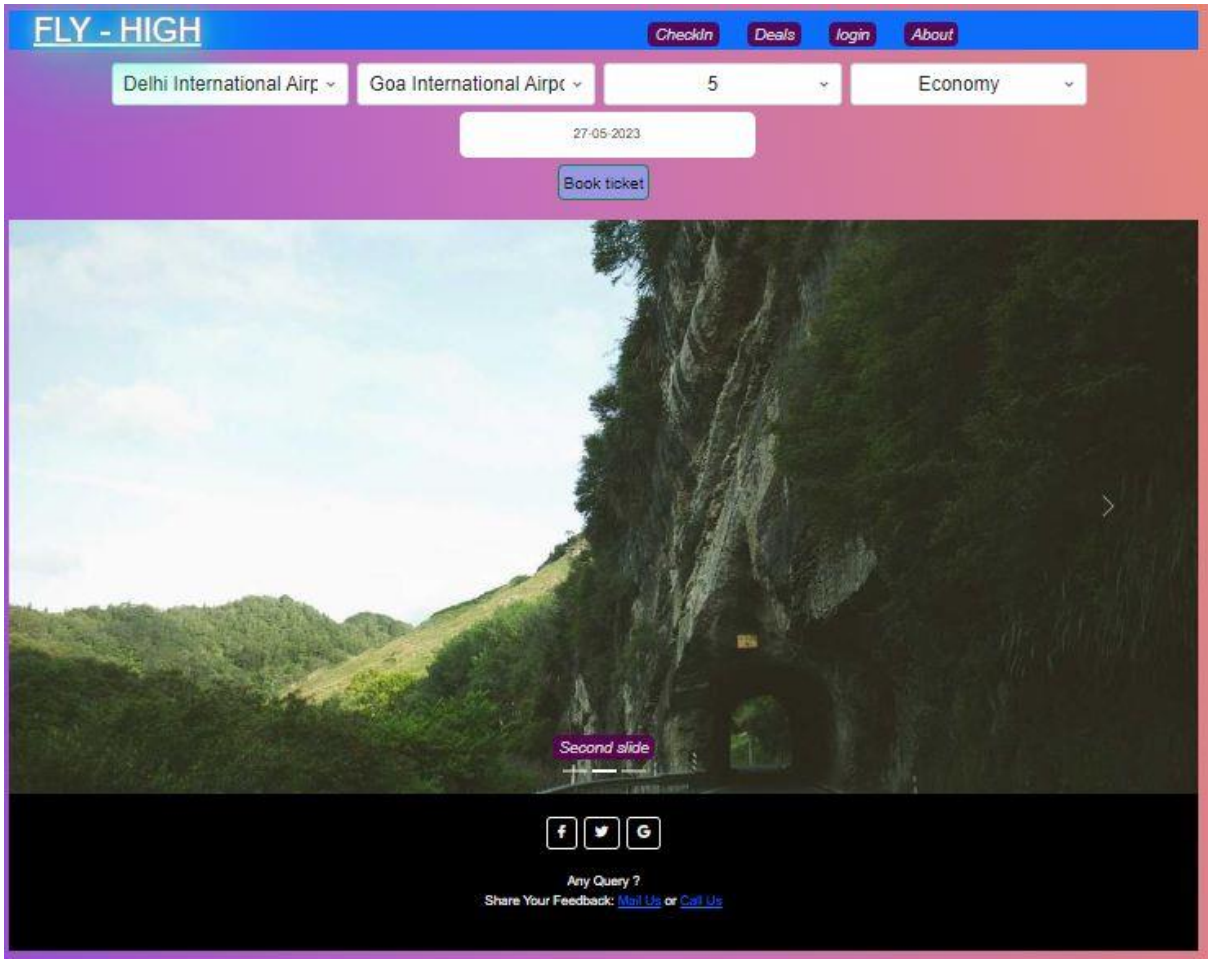
REGISTATION PAGE: -



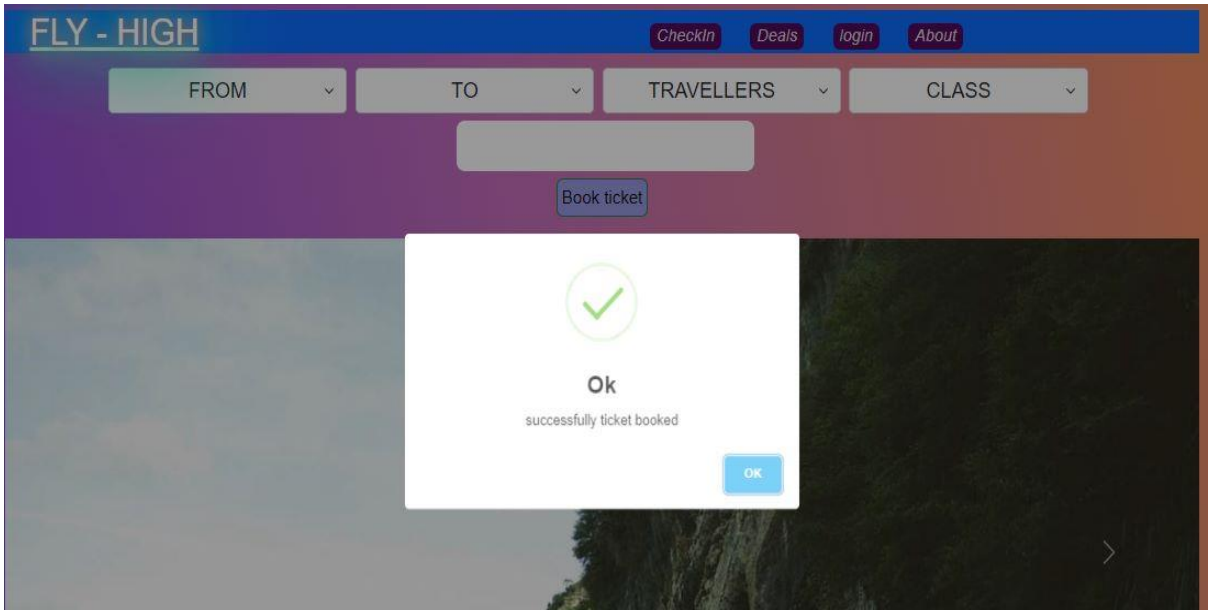
REGISTATION-SUCCESS PAGE: -



BOOK-TICKET PAGE: -



TICKET-BOOKED PAGE: -



ABOUT US: -

7. Testing

Functional Testing

Functional testing is a type of black box testing that bases its test cases on the specifications of the software component under test. Functions are tested by feeding them input and examining the output, and internal program structure is rarely considered (Not like in white-box testing).

Functional testing differs from system testing in that functional testing a program by checking it against ... design document or specification", while system testing "validate a program by checking it against the published user or system requirements.

Functional testing typically involves five steps:->

1. The identification of functions that the software is expected to perform
2. The creation of input data based on the function's specifications
3. The determination of output based on the function's specifications
4. The execution of the test case
5. The comparison of actual and expected outputs

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic.

Structural Testing

It determines the durability and integrity of complete structures or sub-assemblies, usually in multi-axis testing systems that replicate end use conditions. The product range extends from single-axis component testing through to complex testing systems for the simulation of almost all service loads affecting a vehicle or component. IST offers a wide range of modular standard testing systems for use in the automotive industry to test car components such as suspensions and steering systems. Alternatively, systems can be engineered to meet specific customer requirements.

Levels of testing

1) Unit testing
2) Integration testing
3) System testing
4) User acceptance testing

1)unit testing: individual software components of application are tested in isolation from other part of the program.

big ban testing: individual software components of an applications are combined at once into system. every module is first in it tested. Then the entire system is tested for communication interfaces between them.

bottom-up testing: in bottom-up integration testing begins from bottom of the module hierarchy and work up to the top to simulate higher level modules. every module is first unit tested then modules are added in ascending hierarchical order. Lower-level modules are tested first then the next set of higher-level modules are tested with previously tested lower-level modules.

top-down testing: begins testing from top of the module hierarchy and work down to the bottom to simulate lower interfacing modules. Every module is first unit tested then the modules are added in descending hierarchical order. Higher level modules are tested first then the next set of lower-level modules are tested with previously tested higher-level modules.

- **Testing of the Project**

Proofreading
Proofreading (also proof-reading) traditionally means reading a proof copy of a text in order to detect and correct any errors. Modern proofreading often requires reading copy at earlier stages as well.

Double Entry
In the double entry system transactions are entered twice in the accounts. For example, the same car purchased will result an increase in the "vehicle" account and a decrease in "cash" account. Therefore, the difference between the two is that in single entry system, transaction is entered only once and in double entry system it is entered twice

This is to find if any errors are present in the system. To check for the errors an artificial made database are fed to system and been checked and the errors will be verified accordingly.
Two main ways of data verifications are: -

Single Method: this is also known as proof reading method-

Typing the data twice and comparing the two databases at the same time. This method is also known as double entry method

8. Project Legacy

- **Current status of project**

  - **Data has been added**
  - **New customer login**
  - **Deletion in the data**
  - **Updating**
  - **Generate id automatically**

9. Source Code: -

-:FRONTEND:-

CODING IN REACT JS: -

CODING FOR LOGIN PAGE:-

```
import React, { useEffect, useState } from "react";
import { NavLink } from "react-router-dom";
import { Navigate, useNavigate } from 'react-router-dom';
import '../css/Login.css';
import Footer from './Footer';

const Login = () => {
    const navigate = useNavigate();
    const [login, setLogin] = useState({});
    useEffect(() => {
        setLogin({
            email:"",
            password:"",
        })
      },[]);

      const handleChange = e => {
        const { name, value } = e.target;
        setLogin(prevState => ({
            ...prevState,
            [name]: value
        }));
    };


    const loginUser =(event) =>{
        fetch('http://localhost:8080/login', {
            method: 'POST',
            mode: 'cors',
            headers: {
                "Content-Type": 'application/json'
            },
            body: JSON.stringify(login)
        }).then(async data => {
            const response = await data.json();
            if (response.status == 200) {
                localStorage.setItem('email', login.email)
                navigate('/');
            }
            if (response.status > 200) {
```

```
                document.querySelector('.response-message').innerText =
response.message;
            setTimeout(()=> {
                document.querySelector('.response-message').innerText = '';
            }, 2000)
            }
        })
        setLogin({
            email:"",
            password:"",
        })
    }
        return(
            <div>
                <div className="response-message"></div>
                <h1>Login</h1>
                <input type="email" name="email"  value={login.email}
onChange={handleChange} placeholder="Email"/><br/><br/>
                <input type="password" name="password"  value={login.password}
onChange={handleChange} placeholder="Password"/><br/><br/>
                <button onClick={() => loginUser()}>LOG IN</button>
                <NavLink to="/Register" class="nav">
                    <p2 style={{fontSize:'25px', color: 'white',}}>Don't have
Fly High id? </p2> <h3>Register</h3>
                </NavLink>
                <Footer/>
            </div>
        );
}

export default Login;
```

CODING FOR HOME PAGE:-

```
import React from "react";
import Header from "./Header";
import Destination from "./Destinations";
import Gallery from "./Gallery";
import Footer from "./Footer";
import "../css/Home.css";



function Home() {

  return (
    <div>
      <div className="header">
        <Header />
```

```
      </div>
      <div className="">
        <Destination/><br/>
      </div>
      <Gallery/>
      <Footer/>


    </div>
   );
}

export default Home;
```

CODING FOR REGISTER PAGE:-

```
import React, {useState} from "react";
import {Navigate, useNavigate} from 'react-router-dom';
import '../css/Register.css';


function Register(_props) {
    const navigate=useNavigate();

    const registerUser = () => {
        const fullName = document.querySelector('#fullName').value;
        const email = document.querySelector('#email').value;
        const password = document.querySelector('#password').value;

        fetch('http://localhost:8080/register', {
            method: 'POST',
            mode: 'cors',
            headers: {
                "Content-Type": 'application/json'
            },
            body: JSON.stringify({
                fullName,
                email,
                password
            })
        }).then(data => {
            document.querySelector('.response-message').innerText =
'Successfully registered';
            setTimeout(()=> {
                document.querySelector('.response-message').innerText = '';
                navigate('/login');
            }, 2000)
        })
```

```
    }

    return(
      <div className="form">
      <h1>REGISTER</h1>
        <div className="response-message"></div>
          <div className="form-body">
              <div className="username">
                  <label className="form__label" for="fullName">Full Name
</label>
                  <input className="form__input" type="text" id="fullName"
placeholder="Full Name"/>
              </div>
              <div className="email">
                  <label className="form__label" for="email">Email </label>
                  <input  type="email" id="email" className="form__input"
placeholder="Email"/>
              </div>
              <div className="password">
                  <label className="form__label" for="password">Password
</label>
                  <input className="form__input" type="password"
 id="password" placeholder="Password"/>
              </div>
          </div>
          <div class="footer">
              <button type="submit" class="btn"
onClick={registerUser}><h3>Register</h3></button>
          </div>
          <br></br>
          <div class="footer">
          <button type="submit" class="btn" onClick={() =>
navigate('/login')}><h3>Goto login</h3></button>
      </div>

      </div>
    ) ;
};
export default Register;
```

CODING FOR DESTINATION:-

```
import React, { useState } from 'react';
import DatePicker from 'react-datepicker';
import 'react-datepicker/dist/react-datepicker.css';
import '../css/Destination.css';
import swal from 'sweetalert';
```

```
function Destinations() {
  const [fromAirport, setfromAirport] = useState();
  const [toAirport, settoAirport] = useState();
  const [travellerCount, settravellerCount] = useState();
  const [classType, setclassType] = useState();
  const [selectedDate, setSelectedDate] = useState(new Date());

  const bookticket = () => {

    fetch('http://localhost:8080/destinations', {
      method: 'POST',
      mode: 'cors',
      headers: {
        "Content-Type": 'application/json'
      },
      body: JSON.stringify({
        fromAirport: fromAirport,
        toAirport: toAirport,
        travellerCount: travellerCount,
        classType: classType,
        selectedDate: selectedDate,
        email: localStorage.getItem('email')
      })
    }).then(async data => {
      const response = await data.json();
      swal("Ok", "successfully ticket booked", "success")

    })
    setfromAirport("")
    settoAirport("")
    settravellerCount("")
    setclassType("")
    setSelectedDate("")
  }



  const handleSubmit = (e) => {
    e.preventDefault()
    console.log(fromAirport)
    console.log(toAirport)
    console.log(travellerCount)
    console.log(classType)
    console.log(selectedDate)
  }

  return (
    <>
```

```
      <form onSubmit={handleSubmit}>
        <select value={fromAirport} onChange={e =>
setfromAirport(e.target.value)}
          className="form-select" id="fromAirport" aria-label="Default select
example">
          <option selected>FROM</option>
          <option value="AJL"> Aizawl Airport</option>
          <option value="IXU"> Aurangabad Airport</option>
          <option value="ATQ"> Amritsar International Airport</option>
          <option value="IXB"> Bagdogra International Airport</option>
          <option value="IXG"> Belagavi International Airport</option>
          <option value="BLR"> Bengaluru International Airport</option>
          <option value="BBI"> Bhubaneswar International Airport</option>
          <option value="BHO"> Bhopal International Airport</option>
          <option value="IXC"> Chandigarh International Airport</option>
          <option value="MAA"> Chennai International Airport</option>
          <option value="CJB"> Coimbatore International Airport</option>
          <option value="DED"> Dehradun International Airport</option>
          <option value="DEL"> Delhi International Airport</option>
          <option value="DIB"> Dibrugarh Airport</option>
          <option value="DMU"> Dimapur International Airport</option>
          <option value="RDP"> Durgapur Airport</option>
          <option value="GAY"> Gaya International Airport</option>
          <option value="GOY"> Goa International Airport</option>
          <option value="GOP"> Gorakhpur International Airport</option>
          <option value="GAU"> Guwahati International Airport</option>
          <option value="HBX"> Hubli International Airport</option>
          <option value="HYD"> Hyderabad International Airport</option>
          <option value="IMF"> Imphal International Airport</option>
          <option value="IDR"> Indore International Airport</option>
          <option value="JLR"> Jabalpur International Airport</option>
          <option value="JAI"> Jaipur International Airport</option>
          <option value="IXJ"> Jammu International Airport</option>
          <option value="JDH"> Jodhpur International Airport</option>
          <option value="JRH"> Jorhat Airport</option>
          <option value="CNN"> Kannur International Airport</option>
          <option value="KJB"> Kurnool Airport</option>
          <option value="COK"> Kochi Airport</option>
          <option value="KLH"> Kolhapur Airport</option>
          <option value="CCU"> Kolkata International Airport</option>
          <option value="CCJ"> Kozhikode International Airport</option>
          <option value="LKO"> Lucknow International Airport</option>
          <option value="IXM"> Madurai International Airport</option>
          <option value="IXE"> Mangaluru International Airport</option>
          <option value="BOM"> Mumbai International Airport</option>
          <option value="MYQ"> Moisuru Airport</option>
          <option value="NAG"> Nagpur International Airport</option>
          <option value="PAT"> Patna International Airport</option>
```

```
          <option value="IXD"> Prayagraj Airport</option>
          <option value="PNQ"> Pune International Airport</option>
          <option value="IXZ"> Portblair Airport</option>
          <option value="RPR"> Raipur International Airport</option>
          <option value="RJA"> Rajahmundry International Airport</option>
          <option value="IXR"> Ranchi International Airport</option>
          <option value="SHL"> Shillong Airport</option>
          <option value="SAG"> Shirdi Airport</option>
          <option value="IXS"> Silchar International Airport</option>
          <option value="SXR"> Srinagar International Airport</option>
          <option value="STV"> Surat International Airport</option>
          <option value="TRV"> Tiruvananthapuram InternationalAirport</option>
          <option value="TRZ"> Tiruchirappalli International Airport</option>
          <option value="TIR"> Tirupati Airport</option>
          <option value="TCR"> Tuticorin International Airport</option>
          <option value="UDR"> Udaipur International Airport</option>
          <option value="BDQ"> Vadodara International Airport</option>
          <option value="VNS"> Varanasi International Airport</option>
          <option value="VGA"> Vijaywada International Airport</option>
          <option value="VTZ"> Visakhapatnam International Airport</option>
        </select>
        <select value={toAirport} onChange={e => settoAirport(e.target.value)}
          className="form-select" id="toAirport" aria-label="Default select
example">
          <option selected>TO</option>
          <option value="AJL"> Aizawl Airport</option>
          <option value="IXU"> Aurangabad Airport</option>
          <option value="ATQ"> Amritsar International Airport</option>
          <option value="IXB"> Bagdogra International Airport</option>
          <option value="IXG"> Belagavi International Airport</option>
          <option value="BLR"> Bengaluru International Airport</option>
          <option value="BBI"> Bhubaneswar International Airport</option>
          <option value="BHO"> Bhopal International Airport</option>
          <option value="IXC"> Chandigarh International Airport</option>
          <option value="MAA"> Chennai International Airport</option>
          <option value="CJB"> Coimbatore International Airport</option>
          <option value="DED"> Dehradun International Airport</option>
          <option value="DEL"> Delhi International Airport</option>
          <option value="DIB"> Dibrugarh Airport</option>
          <option value="DMU"> Dimapur International Airport</option>
          <option value="RDP"> Durgapur Airport</option>
          <option value="GAY"> Gaya International Airport</option>
          <option value="GOY"> Goa International Airport</option>
          <option value="GOP"> Gorakhpur International Airport</option>
          <option value="GAU"> Guwahati International Airport</option>
          <option value="HBX"> Hubli International Airport</option>
          <option value="HYD"> Hyderabad International Airport</option>
          <option value="IMF"> Imphal International Airport</option>
```

```
          <option value="IDR"> Indore International Airport</option>
          <option value="JLR"> Jabalpur International Airport</option>
          <option value="JAI"> Jaipur International Airport</option>
          <option value="IXJ"> Jammu International Airport</option>
          <option value="JDH"> Jodhpur International Airport</option>
          <option value="JRH"> Jorhat Airport</option>
          <option value="CNN"> Kannur International Airport</option>
          <option value="KJB"> Kurnool Airport</option>
          <option value="COK"> Kochi Airport</option>
          <option value="KLH"> Kolhapur Airport</option>
          <option value="CCU"> Kolkata International Airport</option>
          <option value="CCJ"> Kozhikode International Airport</option>
          <option value="LKO"> Lucknow International Airport</option>
          <option value="IXM"> Madurai International Airport</option>
          <option value="IXE"> Mangaluru International Airport</option>
          <option value="BOM"> Mumbai International Airport</option>
          <option value="MYQ"> Moisuru Airport</option>
          <option value="NAG"> Nagpur International Airport</option>
          <option value="PAT"> Patna International Airport</option>
          <option value="IXD"> Prayagraj Airport</option>
          <option value="PNQ"> Pune International Airport</option>
          <option value="IXZ"> Portblair Airport</option>
          <option value="RPR"> Raipur International Airport</option>
          <option value="RJA"> Rajahmundry International Airport</option>
          <option value="IXR"> Ranchi International Airport</option>
          <option value="SHL"> Shillong Airport</option>
          <option value="SAG"> Shirdi Airport</option>
          <option value="IXS"> Silchar International Airport</option>
          <option value="SXR"> Srinagar International Airport</option>
          <option value="STV"> Surat International Airport</option>
          <option value="TRV"> Tiruvananthapuram InternationalAirport</option>
          <option value="TRZ"> Tiruchirappalli International Airport</option>
          <option value="TIR"> Tirupati Airport</option>
          <option value="TCR"> Tuticorin International Airport</option>
          <option value="UDR"> Udaipur International Airport</option>
          <option value="BDQ"> Vadodara International Airport</option>
          <option value="VNS"> Varanasi International Airport</option>
          <option value="VGA"> Vijaywada International Airport</option>
          <option value="VTZ"> Visakhapatnam International Airport</option>
        </select>
        <select value={travellerCount} onChange={e =>
settravellerCount(e.target.value)}
          className="form-select" id="travellerCount" aria-label="Default
select example">
          <option>TRAVELLERS</option>
          <option>1</option>
          <option> 2</option>
          <option> 3</option>
```

```
          <option> 4</option>
          <option> 5</option>
          <option> 6</option>
          <option> 7</option>
          <option> 8</option>
          <option> 9</option>
          <option> 9+</option>
        </select>
        <select value={classType} onChange={e => setclassType(e.target.value)}
          className="form-select" id="classType" aria-label="Default select
example">
          <option selected>CLASS</option>
          <option value="1"> Economy</option>
          <option value="2"> Business</option>
          <option value="3"> Premium Economy</option>
        </select>
        <div className="date-picker-wrapper">
          <DatePicker
            selected={selectedDate} onChange={(date) => setSelectedDate(date)}
            dateFormat="dd-MM-yyyy"
            minDate={new Date()}
            className="col-sm-3"
            showYearDropdown
            showCurrentDate
          />
        </div>
        {/* <DatePicker */}
        {/* selected={selectedDate} onChange={selectedDate =>
setSelectedDate(selectedDate)} */}
        {/* className="col-sm-5" id="selectedDate" /> */}

        <button type="submit"
          class="btn btn-outline-success btn-lg"
          onClick={bookticket}>
          Book Ticket
        </button>
      </form>
    </>
  )
export default Destinations;
```

CODING FOR CHECK-IN:-

```
import React, { useState } from "react";
import { Navigate, useNavigate } from 'react-router-dom';
import "../css/CheckIn.css";

function CheckIn(_props) {
  const navigate = useNavigate();
  const [formValue, setFormValue] = useState({ pnr: "", email: "" });
```

```
  const [responseMessage, setResponseMessage] = useState("");

  const handleInput = (e) => {
    const { name, value } = e.target;
    setFormValue({ ...formValue, [name]: value });
  };

  const handleSubmit = (e) => {
    e.preventDefault();
    console.log(formValue);
  };

  const checkIn = () => {
    const { pnr, email } = formValue;

    fetch("http://localhost:8080/checkin", {
      method: "POST",
      mode: "cors",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        pnr: pnr,
        email: email,
      }),
    })
      .then(async (data) => {
        const response = await data.json();
        setResponseMessage(response.message);
        setTimeout(() => {
          setResponseMessage("");
          navigate('/');
        }, 2000);
      })
      .catch((error) => {
        console.error("Error:", error);
      });
  };

  return (
    <>
      <div className="container">
        <div className="row">
          <div className="col-md-12">
            <h5 className="mt-2">CheckIn</h5>
            {responseMessage && (
              <p className="text-success response-
message">{responseMessage}</p>
```

```
            )}
            <form onSubmit={handleSubmit}>
              <div className="row">
                <div className="col-md-6">
                  <div className="mb-3">
                    <label className="form-label">PNR</label>
                    <input
                      type="numbers"
                      name="pnr"
                      className="form-control"
                      value={formValue.pnr}
                      placeholder="PNR"
                      errorMessage="PNR should be 10 Numbers"
                      pattern="(str.match(/^\d{10}$/)"
                      rrorMessage="PNR should be 10 Numbers"
                      onChange={handleInput}
                    />
                  </div>
                </div>
                <div className="col-md-6">
                  <div className="mb-3">
                    <label className="form-label">Email</label>
                    <input
                      type="email"
                      name="email"
                      className="form-control"
                      value={formValue.email}
                      placeholder="Email"
                      onChange={handleInput}
                    />
                  </div>
                  <button type="submit" onClick={checkIn}>
                    Check-In
                  </button>
                </div>
              </div>
            </form>
          </div>
        </div>
      </div>
    </>
  );
};

export default CheckIn;
```

CODING FOR DEALS:-

```
import React, { useState } from "react";
import { NavLink } from "react-router-dom";
import "../css/Details.css";

function Offers(){
    const [cards] = useState([
        {
            title:'Get Flat 12% off with SBI Credit Cards',

            text:'Get 12% instant off upto Rs.1500 on domestic flights.Use
code:ISVBIC for regular booking of domestic flights '
        },
        {
            title:'Get Flat 13% off with UPI',

            text:'Get 13% off via UPI payments on domestic flights.'
        },
        {
            title:'Get Flat 10% off with PayPal for international flights',

            text:'There are many variations of passages of Lorem Ipsum
available,but the majority have suffered alteration in some form, by injected
humour'
        },
        {
            title:'Get Flat Rs.800 off on flights with HDFC Credit Cards',

            text:'Get Rs.800 off on domestic flights with HDFC Bank Diners &
Infinia Credit Cards.Use code:ISYBC for booking '
        },
        {
            title:'Win Free flight tickets daily',

            text:'Book your flight using code FREEFLIGHT and get a chance to
win a free flight ticket worth Rs.6000'
        },
        {
            title:'Get Flat 12% off on domestic flights with ICICI Credit
Cards',

            text:'Get 12% off on your domestic flight booking.Use coupon
code:ISZBC and pay via ICICI Credit Cards'
        },
    ])
    return (

        <div>
        <section>
```

```
        <div className="container">
        <h1>Flight Offers</h1>
        <div className="cards">
         {
            cards.map((card,i) => (

        <div key={i} className="card">

        <h9>{card.title}</h9>
        <p style={{fontSize:'25px', color: 'black',}}>{card.text}</p>
        <NavLink to="/Details" class="nav">
         <h3>Grab Now</h3>
        </NavLink>
        </div>
    ))
            }

        </div>
        </div>
        </section>
        </div>

    );
};
export default Offers;
```

CODING FOR HEADER:-

```
import React from "react";
import { Link } from "react-router-dom";
import "../css/Header.css";

const Header = () => {
  const isLoggedIn = false;
  return (
    <div>
      <React.Fragment>
        <nav className="navbar navbar-expand-lg navbar-light bg-primary">
          <a className="neonText" style={{ fontSize: '40px', marging: '20px',
padding: '30px' }} href="#!">
            FLY - HIGH
          </a>
          <button
            className="navbar-toggler"
            type="button"
            data-toggle="collapse"
            data-target="#navbarSupportedContent"
            aria-controls="navbarSupportedContent"
            aria-expanded="false"
```

```
            aria-label="Toggle navigation">
            <span className="navbar-toggler-icon" />
          </button>
          <div className="collapse navbar-collapse" style={{ marginLeft:
'500px' }} id="navbarSupportedContent">
            <ul className="navbar-nav mr-auto">

              <li class="nav-item">
                <Link class="nav-link" to='/CheckIn'><h3>CheckIn</h3></Link>
              </li>

              <li className="nav-item">
                <Link className="nav-link" to="/Deals">
                  <h3>Deals </h3>
                </Link>
              </li>


              <li className="nav-item">
                {isLoggedIn ?
                  <Link className="nav-link" to="/Login">
                    <h3>Login</h3>
                  </Link> :
                  <Link className="nav-link" to="/Login" onClick={() =>
localStorage.removeItem("email")}>
                    <h3>login</h3>
                  </Link>}
              </li>

              <li className="nav-item">
                <Link className="nav-link" to="/About">
                  <h3>About</h3>
                </Link>

              </li>
            </ul>
          </div>
        </nav>
      </React.Fragment>
    </div>
  )
};

export default Header;
```

CODING FOR FOOTER:-

```
import React from "react";
```

```
function Footer() {
    return (
        <footer className="bg-black text-center text-white">
            <div className="container p-4">
                <section className="mb-4">
                    <a className="btn btn-outline-light btn-floating m-1"
href="/" role="button"><i className="fab fa-facebook-f"></i></a>

                    <a className="btn btn-outline-light btn-floating m-1"
href="/" role="button"><i className="fab fa-twitter"></i></a>

                    <a className="btn btn-outline-light btn-floating m-1"
href="/" role="button"><i className="fab fa-google"></i></a>
                </section>

                <section className="mb-4">
                    <p>
                    Any Query ? <br/>
                    Share Your Feedback: <a
href="mailto:s.n240819@gmail.com">Mail Us</a> or <a href="tel:6291338577">Call
Us</a> <br/>
                    </p>
                </section>
            </div>
        </footer>

    );
};
export default Footer;
```

CODING FOR GALLERY:-

```
import React from 'react';
import { Carousel } from 'react-bootstrap';

const CarouselComponent = () => {
  return (
    <Carousel >
      <Carousel.Item>
        <img
          className="d-block w-100"
          src="https://picsum.photos/1080/520?text=First slide&bg=373940"
          alt="First slide"
        />
        <Carousel.Caption>
          <h3>First slide</h3>
        </Carousel.Caption>
      </Carousel.Item>
      <Carousel.Item>
```

```
        <img
          className="d-block w-100"
          src="https://picsum.photos/1080/520?text=Second slide&bg=282c34"
          alt="Second slide"
        />

        <Carousel.Caption>
          <h3>Second slide</h3>
        </Carousel.Caption>
      </Carousel.Item>
      <Carousel.Item>
        <img
          className="d-block w-100"
          src="https://picsum.photos/1080/520?text=Third slide&bg=20232a"
          alt="Third slide"
        />
        <Carousel.Caption>
         <h3>Third slide</h3>
        </Carousel.Caption>
      </Carousel.Item>
    </Carousel>
  );
};

export default CarouselComponent;
```

CODING FOR ABOUT:-

```
import React from "react";
import Footer from './Footer';
import '../css/About.css';

function About(){
    return(
        <div>
            <h1>About Us</h1><br/>
            <div className="details">
            MCA Minor Project<br/>
            TECHNO COLLEGE HOOGHLY  (TIH)<br/>
            SUBHAJIT NASKAR - 18871021057<br/>
            ISHITA NASKAR - 18871201054<br/>
            PARTHANA GHOSH - 18871201049<br/>
            UNNATI SAHA - 18871021060
            </div>
            <div>
                <Footer/>

            </div>
```

```
        </div>
    );
};
export default About;
```

CODING FOR APP:-

```
import './App.css';
import {BrowserRouter, Routes, Route} from 'react-router-dom';
import Home from './component/Home';
import About from './component/About';
import Login from './component/Login';
import Register from './component/Register';
import Deals from './component/Deals';
import Bookings from './component/Bookings';
import CheckIn from './component/CheckIn';
import Dropdown from './component/Dropdown';
import Details from './component/Details';
import Search from './component/Search';



function App() {
  return (
    <div className="App">
    <BrowserRouter>
      <Routes>
        <Route exact path="/" element={<Home/>}>Home</Route>
        <Route exact path="/About" element={<About/>}>About</Route>
        <Route exact path="/Login" element={<Login/>}>Login</Route>
        <Route exact path="/Register" element={<Register/>}>Register</Route>
        <Route exact path="/Deals" element={<Deals/>}>Deals</Route>
        <Route exact path="/Bookings" element={<Bookings/>}></Route>
        <Route exact path='/Booking/dropDown' element={<Dropdown/>}/>
        <Route exact path="/CheckIn" element={<CheckIn/>}>CheckIn</Route>
        <Route exact path="/Details" element={<Details/>}>Details</Route>
        <Route exact path="/Search" element={<Search/>}>Search</Route>
      </Routes>
    </BrowserRouter>
    <div className="text-wrapper">
        </div>
    </div>

}
```

-: BACKEND :-

CODING FOR SERVER:-

```
const { MongoClient } = require('mongodb');
const express = require('express');
const app = express();
const bodyParser = require('body-parser');
const cors = require('cors');


const url = 'mongodb://localhost:27017';
const client = new MongoClient(url);

const dbName = 'flyhigh';

const PORT=8080;

app.use(bodyParser.json());
app.use(cors())

app.post('/register', async (req, res) => {
    const {fullName,email, password} = req.body;
    await app.dbConnection.collection('users').insertMany([{fullName, email,
password}]);
    // console.log(username, password);
    return res.send({status: 201});
});

app.post('/login', async (req, res) => {
    const {email, password} = req.body;
    const user = await app.dbConnection.collection('users').find({email,
password}).toArray();
    if (user.length == 0) {
        return res.send({status: 401, message: 'Invalid email/Password'});
    }
    await app.dbConnection.collection('users').updateOne({email}, {$set:
{active: true}});
    return res.send({ status: 200, message: 'Successfully logged in' , user});
});

app.post('/logout', async (req, res) => {
    const {username} = req.body;
    await app.dbConnection.collection('users').updateOne({username}, {$set:
{active: false}});
    return res.send({ status: 200, message: 'Successfully logged out'});
});

app.post('/checkin', async (req, res) => {
```

```
    const {pnr,  email} = req.body;
    const users = await
app.dbConnection.collection('users').find({email}).toArray();
    if (users.length == 0 || users[0].active != true) {
        return res.send({status: 401, message: 'Please log in first'});
    }
    await app.dbConnection.collection('checkins').insertMany([{email, pnr}]);
    const message = `Checked in for pnr- ${pnr} for user- ${email}`;
    return res.send({status: 201, message})
})

app.post('/destinations', async (req, res) => {
    const { fromAirport, toAirport, travellerCount, classType, selectedDate,
email } = req.body;
    const users = await app.dbConnection.collection('users').find({ email
}).toArray();
    if (users.length === 0 || !users[0].active) {
      return res.send({ status: 401, message: 'Please log in first' });
    }
    await app.dbConnection.collection('destinations').insertMany([{
fromAirport, toAirport, travellerCount, classType, selectedDate }]);
    const message = `Your ${fromAirport} ticket booked successfully to
${toAirport}`;
    return res.send({ status: 201, message });
  });

app.listen(PORT, async () => {
    console.log('came here');
    await client.connect();
    app.dbConnection = client.db(dbName);
    console.log(`server listening on ${PORT}`);
})
```

DATABASE

CODING FOR API.jsx:-

```
import React from 'react'

const Api = () => {
  return (
    <div>Api</div>
  )
}

export default Api;
```

CODING FOR DB.json:-


```
{
    "users":[
        {
          "id": 1,
          "name": "Leanne Graham",
          "username": "Bret",
          "email": "Sincere@april.biz",
          "address": {
            "street": "Kulas Light",
            "suite": "Apt. 556",
            "city": "Gwenborough",
            "zipcode": "92998-3874",
            "geo": {
              "lat": "-37.3159",
              "lng": "81.1496"
            }
          },
          "phone": "1-770-736-8031 x56442",
          "website": "hildegard.org",
          "company": {
            "name": "Romaguera-Crona",
            "catchPhrase": "Multi-layered client-server neural-net",
            "bs": "harness real-time e-markets"
          }
        },
        {
          "id": 2,
          "name": "Ervin Howell",
          "username": "Antonette",
          "email": "Shanna@melissa.tv",
          "address": {
            "street": "Victor Plains",
            "suite": "Suite 879",
            "city": "Wisokyburgh",
            "zipcode": "90566-7771",
            "geo": {
              "lat": "-43.9509",
              "lng": "-34.4618"
            }
          },
          "phone": "010-692-6593 x09125",
          "website": "anastasia.net",
          "company": {
            "name": "Deckow-Crist",
            "catchPhrase": "Proactive didactic contingency",
            "bs": "synergize scalable supply-chains"
          }
```

```
    },
    {
      "id": 3,
      "name": "Clementine Bauch",
      "username": "Samantha",
      "email": "Nathan@yesenia.net",
      "address": {
        "street": "Douglas Extension",
        "suite": "Suite 847",
        "city": "McKenziehaven",
        "zipcode": "59590-4157",
        "geo": {
          "lat": "-68.6102",
          "lng": "-47.0653"
        }
      },
      "phone": "1-463-123-4447",
      "website": "ramiro.info",
      "company": {
        "name": "Romaguera-Jacobson",
        "catchPhrase": "Face to face bifurcated interface",
        "bs": "e-enable strategic applications"
      }
    },
    {
      "id": 4,
      "name": "Patricia Lebsack",
      "username": "Karianne",
      "email": "Julianne.OConner@kory.org",
      "address": {
        "street": "Hoeger Mall",
        "suite": "Apt. 692",
        "city": "South Elvis",
        "zipcode": "53919-4257",
        "geo": {
          "lat": "29.4572",
          "lng": "-164.2990"
        }
      },
      "phone": "493-170-9623 x156",
      "website": "kale.biz",
      "company": {
        "name": "Robel-Corkery",
        "catchPhrase": "Multi-tiered zero tolerance productivity",
        "bs": "transition cutting-edge web services"
      }
    },
    {
```

```
    "id": 5,
    "name": "Chelsey Dietrich",
    "username": "Kamren",
    "email": "Lucio_Hettinger@annie.ca",
    "address": {
      "street": "Skiles Walks",
      "suite": "Suite 351",
      "city": "Roscoeview",
      "zipcode": "33263",
      "geo": {
        "lat": "-31.8129",
        "lng": "62.5342"
      }
    },
    "phone": "(254)954-1289",
    "website": "demarco.info",
    "company": {
      "name": "Keebler LLC",
      "catchPhrase": "User-centric fault-tolerant solution",
      "bs": "revolutionize end-to-end systems"
    }
  },
  {
    "id": 6,
    "name": "Mrs. Dennis Schulist",
    "username": "Leopoldo_Corkery",
    "email": "Karley_Dach@jasper.info",
    "address": {
      "street": "Norberto Crossing",
      "suite": "Apt. 950",
      "city": "South Christy",
      "zipcode": "23505-1337",
      "geo": {
        "lat": "-71.4197",
        "lng": "71.7478"
      }
    },
    "phone": "1-477-935-8478 x6430",
    "website": "ola.org",
    "company": {
      "name": "Considine-Lockman",
      "catchPhrase": "Synchronised bottom-line interface",
      "bs": "e-enable innovative applications"
    }
  },
  {
    "id": 7,
    "name": "Kurtis Weissnat",
```

```
    "username": "Elwyn.Skiles",
    "email": "Telly.Hoeger@billy.biz",
    "address": {
      "street": "Rex Trail",
      "suite": "Suite 280",
      "city": "Howemouth",
      "zipcode": "58804-1099",
      "geo": {
        "lat": "24.8918",
        "lng": "21.8984"
      }
    },
    "phone": "210.067.6132",
    "website": "elvis.io",
    "company": {
      "name": "Johns Group",
      "catchPhrase": "Configurable multimedia task-force",
      "bs": "generate enterprise e-tailers"
    }
  },
  {
    "id": 8,
    "name": "Nicholas Runolfsdottir V",
    "username": "Maxime_Nienow",
    "email": "Sherwood@rosamond.me",
    "address": {
      "street": "Ellsworth Summit",
      "suite": "Suite 729",
      "city": "Aliyaview",
      "zipcode": "45169",
      "geo": {
        "lat": "-14.3990",
        "lng": "-120.7677"
      }
    },
    "phone": "586.493.6943 x140",
    "website": "jacynthe.com",
    "company": {
      "name": "Abernathy Group",
      "catchPhrase": "Implemented secondary concept",
      "bs": "e-enable extensible e-tailers"
    }
  },
  {
    "id": 9,
    "name": "Glenna Reichert",
    "username": "Delphine",
    "email": "Chaim_McDermott@dana.io",
```

```
          "address": {
            "street": "Dayna Park",
            "suite": "Suite 449",
            "city": "Bartholomebury",
            "zipcode": "76495-3109",
            "geo": {
              "lat": "24.6463",
              "lng": "-168.8889"
            }
          },
          "phone": "(775)976-6794 x41206",
          "website": "conrad.com",
          "company": {
            "name": "Yost and Sons",
            "catchPhrase": "Switchable contextually-based project",
            "bs": "aggregate real-time technologies"
          }
        },
        {
          "id": 10,
          "name": "Clementina DuBuque",
          "username": "Moriah.Stanton",
          "email": "Rey.Padberg@karina.biz",
          "address": {
            "street": "Kattie Turnpike",
            "suite": "Suite 198",
            "city": "Lebsackbury",
            "zipcode": "31428-2261",
            "geo": {
              "lat": "-38.2386",
              "lng": "57.2232"
            }
          },
          "phone": "024-648-3804",
          "website": "ambrose.net",
          "company": {
            "name": "Hoeger LLC",
            "catchPhrase": "Centralized empowering task-force",
            "bs": "target end-to-end models"
          }
        }
      ]
}
```

10. Bibliography

1. Lakhani, Karim R., & von Hippel, Eric (2003). How Open Source Software Works: Free User to User Assistance. *Research Policy*, 32, 923–943 doi:10.2139/ssrn.290305

2. Choudhary, S.R. (2014). "Cross-platform testing and maintenance of web and mobile applications". *Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014*.

3. Pouliquen, Louis Y. Risk Analysis in Project Appraisal. World Bank Staff Occasional Papers, no. 11. Baltimore: Johns Hopkins University Press, 1970.

4. Boehm, B, "Spiral Development: Experience, Principles,andRefinements", Special Report CMU/SEI-2000-SR-008, July 2000

5. Malcolm, D. G., J. H. Roseboom, C. E. Clark, W. Fazar. "Application of a Technique for Research and Development Program Evaluation," *OPERATIONS RESEARCH*, Vol. 7, No. 5, September– October 1959, pp. 646–669

6. Boehm, Barry (1981). *Software Engineering Economics*. PrenticeHall. ISBN 0-13-822122-7.

7. A.P.G. Brown, "Modelling a Real-World System and Designing a Schema to Represent It", in Douque and Nijssen (eds.), *Data Base Description*, North-Holland, 1975, ISBN 0-7204-2833-5.

8. Bruza, P. D.; van der Weide, Th. P. (1990-11-01). "Assessing the quality of hypertext views". *ACM SIGIR Forum*. 24 (3): 6–25.doi:10.1145/101306.101307. ISSN 0163-5840

9. Gemino, A., Parker, D.(2009) "Use case diagrams in support of use case modeling: Deriving understanding from the picture", *Journal of Database Management*, 20(1), 1-24.

10. Imielinski, T.;Lipski, W. (1982). *A systematic approach to relational database theory*. *Proceedings of the 1982 ACM SIGMOD International Conference on Management of Data (SIGMOD '82)*. New York, NY: ACM. pp. 8–14.doi:10.1145/582353.582356. ISBN 978-0897910736.

11. Störrle, Harald, and J. H. Hausmann. "semantics of uml 2.0 activities." Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing. 2004.