

**ST. JOSEPH'S COLLEGE (AUTONOMOUS)
LALBAGH ROAD, BANGALORE - 560027**



MAJOR PROJECT LAB

Submitted by:

Varshitha A R Gowda (19PMC23062)

Surabhi U (19PMC23068)

Saba Santhosh (19PMC23069)

Akash Mayya (19PMC23072)

Under the Guidance of

Prof. Dr. B.G. Prashanthi

Department of Computer Science

VI SEMESTER B.Sc.

2021-2022

ST. JOSEPH'S COLLEGE (AUTONOMOUS) LALBAGH ROAD, BANGALORE - 560027



CERTIFICATE

This is to certify that **Varshitha AR Gowda (19PMC23062), Surabhi U (19PMC23068), Saba Santhosh (19PMC23069), Akash Mayya (19PMC23072)** have successfully completed the project titled “**Airlines Reservation System**” at **ST. JOSEPH'S COLLEGE (AUTONOMOUS)** under the supervision and guidance in the fulfilment of requirements of Sixth Semester, Bachelor of Science (Computer Science) of Bengaluru City University, Bengaluru.

Dr. B.R. Prashanthi

Prof. Sandhya N
Head of the Department
Computer Science

ACKNOWLEDGEMENT

We take this opportunity to place on record, our sincere gratitude and a deep sense of thankfulness, to the following individuals, without whose strong support, this project would not have been possible. We wish to express our deepest gratitude to our respected principal Fr. Dr. Victor Lobo S.J, for his enthusiasm and helpful nature in making our education in St. Joseph's college as a memorable one.

I would like to extend my sincere gratitude to Prof. Dr. B.G. Prashanthi, Department of Computer Science, St. Joseph's College (Autonomous), for their continuous support, guidance, knowledge and insights in assisting us in our Major project work.

They inspired and encouraged us to go out into the field and gain hands-on experience. We are very grateful for all the time that they have sacrificed in helping us sort out the fine details of this project.

We would also like to convey our gratitude towards St. Joseph's College (Autonomous) for giving us an opportunity to do a web development project.

We would also like to express our gratitude to all of our friends, parents, and well-wishers for their help in completing our major project.

Table of Contents

SL NO	TITLE	PAGE NUMBER
1.	Abstract	6
2.	Introduction	7
3.	Study of Existing System	7
4.	Study of Proposed System	8
5.	System Requirements	9
6.	Requirement Analysis	11
7.	System Design <ul style="list-style-type: none"> • Entities and Relation • Key Attributes • Data Flow Diagram • E-R Model • Conversion of E-R Model to Relational Model • Relational Model • Participation Constraints and Cardinality Ratios • Normalization 	12 13 15 16 18 20 21 22
8.	Software Design	27
9.	Code	36
10.	Software Testing	74
11.	Future Enhancement & Conclusion	76
12.	References	77

ABSTRACT

By convergence of the world into a global village, traveling great distances has become so common for not only business but also for pleasure purposes. Due to the fast-paced nature of the world today, air travel is considered as one of the first choice among several modes of transportation to save time and one of the tools passengers use to speed up the process of travelling by air is the Airline Reservation System (ARS),

An Airline Reservation system is very important because it has the strong ability to reduce errors that might have occurred when using a manual system of reservation and helps speed up the boarding process.

The need of this system was realized since the beginning stages of the airline industry when information such as routes, aircrafts, schedules and fares about flights was published by airlines in large books. Travel agents had the difficult task of looking into separate books for reservations that involve multiple airlines. It was a dream to get a real time picture of available seats because airlines share information at day end only.

The following programming languages were used: PHP, JavaScript, HTML and CSS for designing the interface of the system, and SQL for the database. Especially passengers who don't have time to or don't want to visit a travel agent. Keeping in mind the fact that people might not have enough time out of their busy routine to go and visit a travel agent in order to make a reservation, most airlines have offered their services over the World Wide Web. Therefore, internet has become the integral part of the flight reservation process not only for travel agents, who sits in the office and makes use of web servers of different airlines to find out the most suitable flight for a particular customer, but also for the customers who want to reserve flights online without necessarily having to contact air travel agents and get all the details at their fingertips.

INTRODUCTION

One of the most common modes of travel is traveling by air. Customers who wish to travel by air nowadays have a wide variety of flights and a range of timings to choose from. Nowadays competition is so fierce between flights that there are a lot of discounts and a lot of luxuries given to customers that will give an edge to that particular flight.

The World Wide Web has become tremendously popular over the last four years, and currently most of the Flights have made provision for online reservations of their flights. The Internet has become a major resource for people looking for making reservations online without the hassle of meeting travel agents. Airline Reservation System intends to serve these purposes.

It intends to check all the available Flight databases and return a string of results, which can help them in their travel plans.

STUDY OF EXISTING SYSTEM

Every Airline company is charged with the responsibility of ensuring to give safe and comfortable service to its customers. To ensure this quality of service, the companies which work in this business should reach out to their customers and give quality service to compete in the market. And this includes an easy ticket reservation and cancellation system. In the manual system, the daily reservation list for all flights is generated and sent. Very often, they cannot put up the reservation list on time. Owing to the inefficiencies in the manual system, when passengers cancel tickets, the reservation list is not updated in time. They maintain a waiting list, which is used to update the reservation list when passengers cancel tickets. Currently, the manual system handles all requests for changes in the reservations. It treats a change of this kind as a cancellation, and reissues tickets. Reservation opens a few days before the scheduled departure date. Based on the availability of seats the tickets are issued. Each ticket, whether confirmed or wait-listed, has a unique identification number. This number is generated in a serial order. The reservation clerk records the amount of fare paid for the ticket in the Cash Collection Register. A passenger

can cancel tickets by submitting a cancellation form with the ticket. Depending on the difference in hours between the departure and cancellation the passenger loses a certain percentage of the fare.

Limitations of the manual system

- Only a few bookings could be made due to manual operation.
- It takes an enormous amount of time for recording transactions.
- Requires a large number of manual laborers.
- Customers should go to ticket offices to reserve a ticket and cancel it.

STUDY OF PROPOSED SYSTEM

In our proposed system we have the provision for adding the details of the customers by the ticketing manager with the help of a predefined format and drop-down lists. So the overhead of the ticketing manager becomes less. Another advantage of the system is that it is very easy to edit the details of the customers and delete a ticket entry when it found unnecessary.

Our proposed system has several advantages

- User friendly interface
- Fast access to database
- Less error
- More Storage Capacity
- Look and Feel Environment
- Quick transaction

All the manual difficulties in managing the customer details in an airline reservation database have been rectified by implementing computerization.

The proposed system has the strong ability to reduce errors that might have occurred when using a manual system of reservation and helps speed up the boarding process. Airways has an existing Airline Reservation System, but there are paper analyzed the problems of the existing system. The problems are: inability of passengers to select

their preferred seat(s) from the reservation system, No option to passengers for printing their boarding pass from the existing system, no notification of passengers of flight cancellation or delays and passengers don't have access to aircraft maintenance report to ease the fears associated with air travel and its disasters. In this paper, an Improved Airline Reservation System that is convenient for passengers to solve the before mentioned problems. The Improved Airline Reservation system is designed and implemented using data obtained from interviewing airline personnel, passengers, and materials on Airline Reservation Systems. In this regard, the Improved Airline Reservation System will assist Airways in variety of airline administration tasks and service needs from time of initial reservation through completion of the task. The following programming languages were used: PHP, JavaScript, HTML and CSS for designing the interface of the system, and SQL for the database.

SYSTEM REQUIREMENTS

To be used efficiently, all computer software needs certain hardware components or other software resources to be present on a computer. These prerequisites are known as system requirements and are often used as a guideline as opposed to an absolute rule. Most software defines two sets of system requirements: minimum and recommended. Airline Reservation System requires hardware like required amount of RAM, basic required processor and software like browser that supports, server that is required.

Hardware Requirements:

- OS Name: Windows 10/11 (64 bit)
- RAM: 8 GB
- Hard disk: Minimum 1 GB
- Processor: Intel® Core i5-6500 CPU@ 3.20 GHz

Software Requirements:

- Front End: Xampp v8.8.1, VS Code
- Back End: Apache tomcat server
- Web Server: Apache
- Compatible Browser: Google Chrome

Software Description:

MySQL:

XAMPP is a free and open-source cross-platform web server solution stack package developed by Apache Friends consisting mainly of the Apache HTTP Server, MariaDB database, and interpreters for scripts written in the PHP and Perl programming languages.

VS Code:

Visual Studio Code is a source code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python, C++ and Fortran. It is based on the Electron framework which is used to develop Node.js Web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Out of the box, Visual Studio Code includes basic support for most common programming languages. This basic support includes syntax highlighting, bracket matching, code folding, and configurable snippets. Visual Studio Code also ships with IntelliSense for JavaScript, TypeScript, JSON, CSS, and HTML, as well as debugging support for Node.js. Support for additional languages can be provided by freely available extensions on the VS Code Marketplace.

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse. This allows it to operate as a language-agnostic code editor for any language. It supports many programming languages and a set of features that differs per language. Unwanted files and folders can be excluded from the project tree via the settings. Many Visual Studio Code features are not exposed through menus or the user interface but can be accessed via the command palette.

MySQL Server:

MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter,

and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available and offer additional functionality. MySQL is a central component of the LAMP open-source web application software stack (and other "AMP" stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python". Applications that use the MySQL database include: TYPO3, MODX, Joomla, WordPress, Simple Machines Forum, phpBB, MyBB, and Drupal. MySQL is also used in many high-profile, large scale websites, including Google, Facebook, Twitter, Flickr, and YouTube.

ERDPlus:

ERDPlus is a web-based database modeling tool that lets the user quickly and easily create

- Entity Relationship Diagrams (ERDs)
- Relational Schemas (Relational Diagrams)
- Star Schemas (Dimensional Model)

Draw IO:

It is a useful, free diagramming service with a strong collaboration feature via Google. It's handy and occasionally needs to make diagrams. It is a free speed service that helps in drawing.

REQUIREMENT ANALYSIS

The Airline Reservation System:

- Stores the personal information of passengers such as their name, address, phone number, etc.

- Allows the passenger to log in as a user consisting of username/email, password, etc.
- Allows the user to book tickets stored as scheduled which includes source destination, net fare, arrival, departure, seats availability and date of travel.

Using this function, a user can:

- The user represents staff but has limited access to the system.
- Remove a passenger record if the passenger cancels a booked ticket.
- Book a new flight in one way.
- Cancel a booked ticket.
- Admin can manage all bookings, approve, create new user and add new airlines.
- Displays information about the website on the home page.

SYSTEM DESIGN

Once all the requirements have been collected and analyzed, the next step is to create a conceptual schema for the Airline Reservation System, using a high-level conceptual data model. Design is the creation of a plan or convention for the construction of an object, system or measurable human interaction.

Designing often necessitates considering the aesthetic, functional, economic and sociopolitical dimensions of both the design object and design process of Airline Reservation System. Thus "design" may be a substantive referring to a categorical abstraction of a created thing or things (the design of something) or a verb for the process of creation. It is an act of creativity and innovation. Thus, Airline Reservation System is designed to fulfill the user requirement and it is user friendly.

Entities and Relations

Entities and their attributes

- AIRLINES_LIST – id, airlines, logo_path.

- AIRPORT_LIST – id, airport, location.
- FLIGHT_LIST – id, airline_id, plane_no, departure_airport_id, arrival_airport_id, departure_datetime, arrival_datetime, seats, price, date_created.
- BOOKED_FLIGHT – id, flight_id, name, address, contact.
- USERS – id, name, address, contact, username, password, type.
- SYSTEM_SETTINGS – id, name, email, contact, cover_img, about_content

Key Attributes

Key attributes used in Airline Reservation System

Entities	Primary Key	Foreign Key
AIRLINES_LIST	id	-
AIRPORT_LIST	id	-
FLIGHT_LIST	id	airline_id, departure_airport_id, arrival_airport_id
BOOKED_FLIGHT	id	flight_id
USERS	id	-
SYSTEM_SETTINGS	id	id

Super Key: Super Key is defined as a set of attributes within a table that can uniquely identify each record within a table. Super Key is a superset of Candidate key.

Candidate Key: Candidate keys are defined as the minimal set of fields, which can uniquely identify each record in a table. It is an attribute or a set of attributes that can act as a Primary Key for a table to uniquely identify each record in that table. There can be more than one candidate key.

- A candidate key can never be NULL or empty. And its value should be unique
- .There can be more than one candidate keys for a table.
- A candidate key can be a combination of more than one column (attributes).

Primary key: Primary key is a set of one or more fields/columns of a table that uniquely identify a record in database table. It cannot accept null, duplicate values. Only one candidate key can be primary key. The primary keys are named id in each and every entity types. **Foreign key:** A foreign key is a field in a table that matches another field

of another table. A foreign key places constraint on data in the related tables, which enables SQL to maintain referential integrity.

Unique key: Unique key is a set of one or more fields/columns of a table that uniquely identifies a record in database table. It is like primary key but it can accept only one null value and it cannot have duplicate values. For more help refer the article [Difference between primary key and unique key](#).

Composite Key: Key that consists of two or more attributes that uniquely identify any record in a table is called Composite key. But the attributes, which together form the Composite key, are not a key independently or individually.

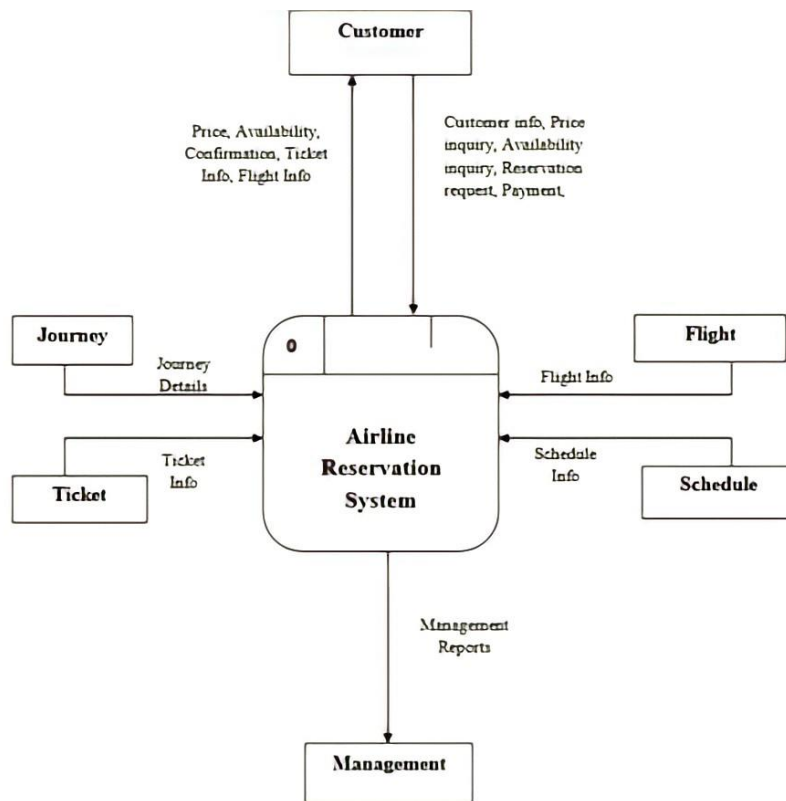
NOT NULL: The NOT NULL constraint enforces a column to NOT accept NULL values. This enforces a field to always contain a value, which means that one cannot insert a new record, or update a record without adding a value to this field.

The DEFAULT SQL constraint: At times, a person might want to automatically insert a value into a column when no other value is provided. That's where the DEFAULT SQL constraint comes in. One can use the constraint to define a column's default value. This can be a handy way to add the current date and time to a column or to avoid having to use NULL values.

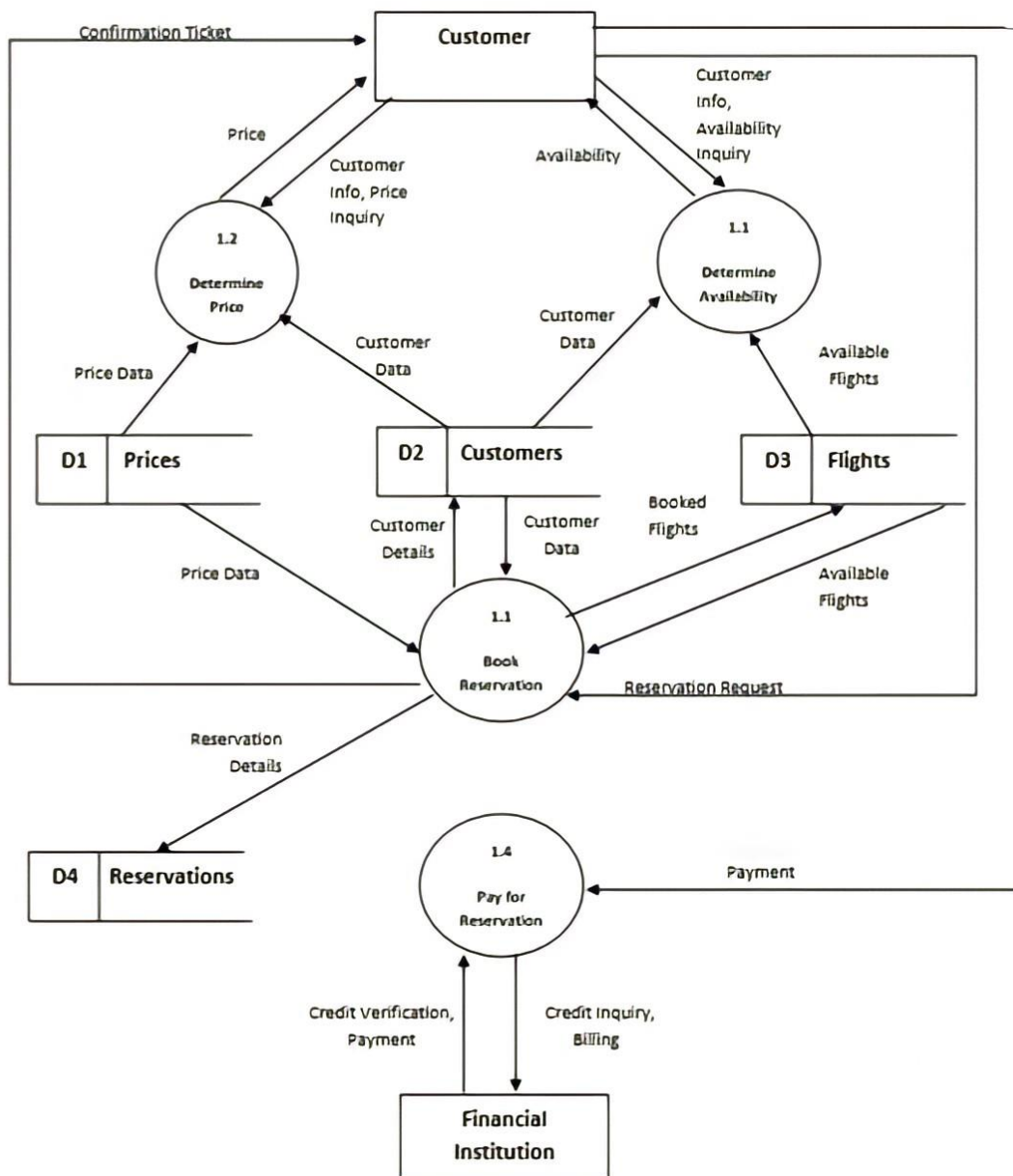
- `airlines_list(id, airlines, logo_path)` – id is the primary key.
- `airport_list(id, airport, location)` - id is the primary key.
- `flight_list(id, airline_id, plane_no, departure_airport_id, arrival_airport_id, departure_datetime, arrival_datetime, seats, price, date_created)` - id is the primary key.
- `booked_flight(id, flight_id, name, address, contact)` - id is the primary key.
- `users(id, name, address, contact, username, password, type)` - id is the primary key.
- `system_settings(id, name, email, contact, cover_img, about_content)` - id is primary key

Data Flow Diagram

LEVEL 0



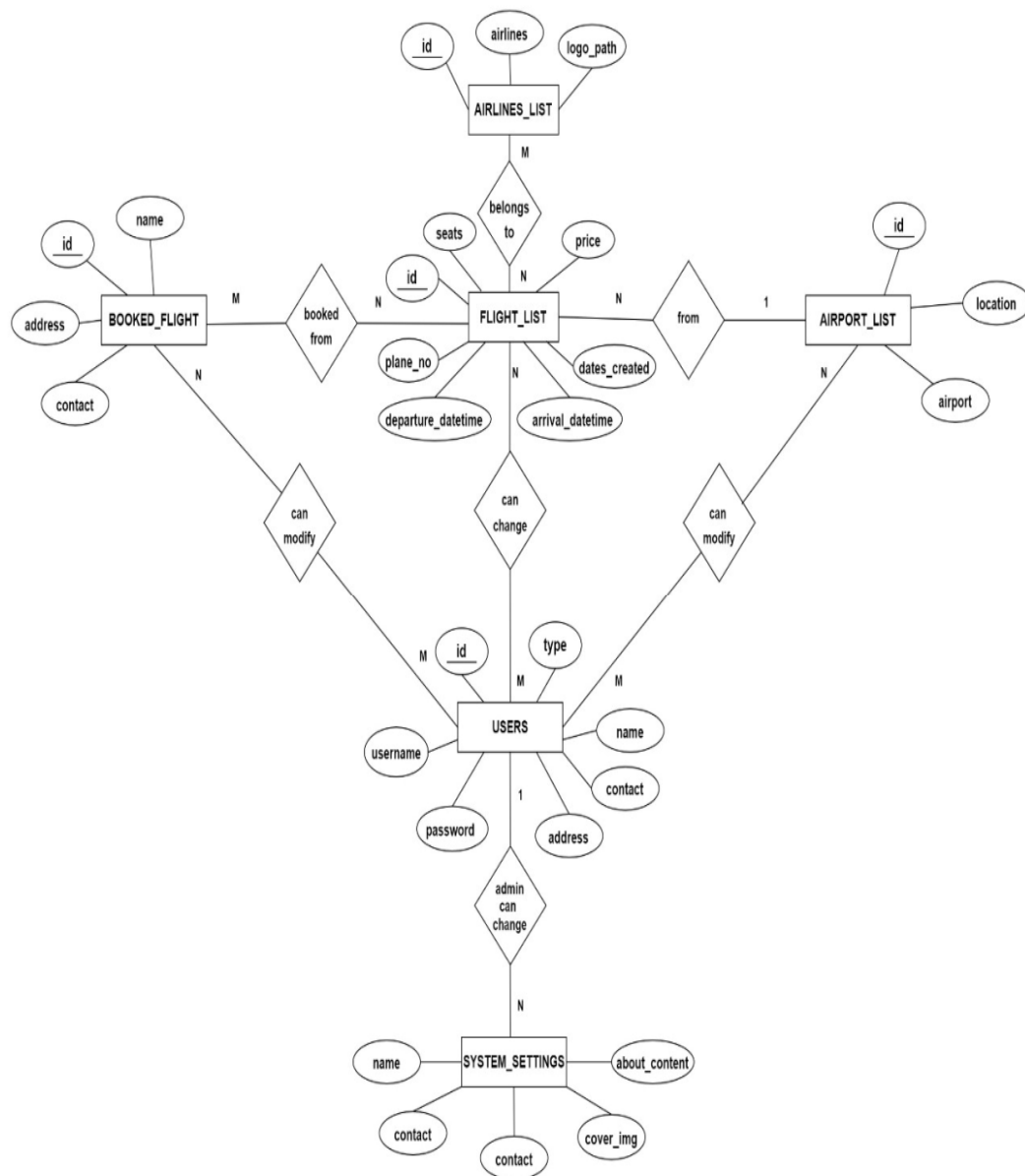
LEVEL 1



E-R Model

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties. Entity relationship diagrams as shown in Fig.2.1 provide a visual starting point for database design that can also be used to help determine information system requirements throughout an organization. After a relational database is rolled out, an

ERD can still serve as a referral point, should any debugging or business process re-engineering be needed later. However, while an ERD can be useful for organizing data that can be represented by a relational structure, it can't sufficiently represent semi-structured or unstructured data. It's also unlikely to be helpful on its own in integrating data into a pre-existing information system.



E-R diagram for Airline Reservation System

Conversion of E-R Model to Relational Model

Step 1 - Mapping of Regular Entity Types: For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E. Include simple components of composite attributes. Choose one of Es key attributes to be the primary key of R. Such relations are sometimes called entity relations because each tuple represents an entity instance.

Step 2 - Mapping of Weak Entity Types: For each weak entity type W in the ER schema with owner entity type E, create a relation R and include all simple attributes (or simple components of composite attributes) of Was attributes of R. Include as foreign key attributes of R, the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s). If there is a weak entity type E2 whose owner is also a weak entity type E1, then E1 should be mapped before E2 to determine its primary key first.

Step 3 - Mapping of Binary 1:1 Relationship Types: For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R. There are three possible approaches: 1. the foreign key approach, 2. the merged relationship approach, and 3. the cross-reference or relationship relation approach. The first approach is the most useful.

A. Foreign key approach: Choose one of the relations (we'll say S) and include as a foreign key in S the primary key of T. It is better to pick an entity type with total participation in R in the role of S. Include all the simple attributes (or the simple components of composite attributes) of the 1:1 relationship type R as attributes of S.

B. Merged relation approach: This involves merging the two entity types and the relationship into a single relation. This may be appropriate when both participations are total.

C. Cross-reference or relationship relation approach: Set up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types. The relation R is called a relationship relation or lookup table, because each tuple in R represents a relationship instance that relates one tuple from S with one tuple of T.

Step 4 - Mapping of Binary 1: N Relationship Types: For each regular binary 1: N relationship R, identify the relation S that represents the participating entity type at the N-side of the relationship type. Include as foreign key in S the primary key of the relation T that represents the other entity type.

Step 5 - Mapping of Binary M: N Relationship Types: For each binary M:N relationship type R, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. Their combination will form the primary key of S. Include in S all the attributes from the M: N relationship type.

Step 6 - Mapping of Multivalued Attributes: For each multivalued attribute A, create a new relation R. This relation will include an attribute corresponding to A, plus the primary key attribute K—as a foreign key in R—of the relation that represents the entity type or relationship type that has A as an attribute. The primary key of R is the combination of A and K. If the multivalued attribute is a composite, the user includes its simple components.

Step 7 - Mapping of N-ary Relationship Types: For each n-ary relationship type R, where $n > 2$, create a new relation S to represent R. Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types. Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.

The primary key of S is usually a combination of all the foreign keys that reference the relations. However, if the cardinality constraints on any of the entity types E participating in R is 1, then the primary key of S should not include the foreign key attribute that references the relation E' corresponding to E.

Step 8 - Options for Mapping Specialization or Generalization: Convert each specialization with m subclasses $\{S_1, S_2, \dots, S_m\}$ and (generalized superclass C, where the attributes of C are $\{k, a_1, \dots, a_n\}$ and k is the (primary) key, into relation schemas.

A. Create different relations (tables) for each sub class and one for the superclass.

B. Integrate the superclass into each 'M' relation so there are no more relationships.

C. Union all the attributes into one relation and add an indicator, like an integer, that specifies the type of entity it is. Obviously, there would be nulls in cells that did not correspond to the type of class that entity was.

D. Similar to option 'C' but use binary flags to determine the entity type after the user union all the attributes.

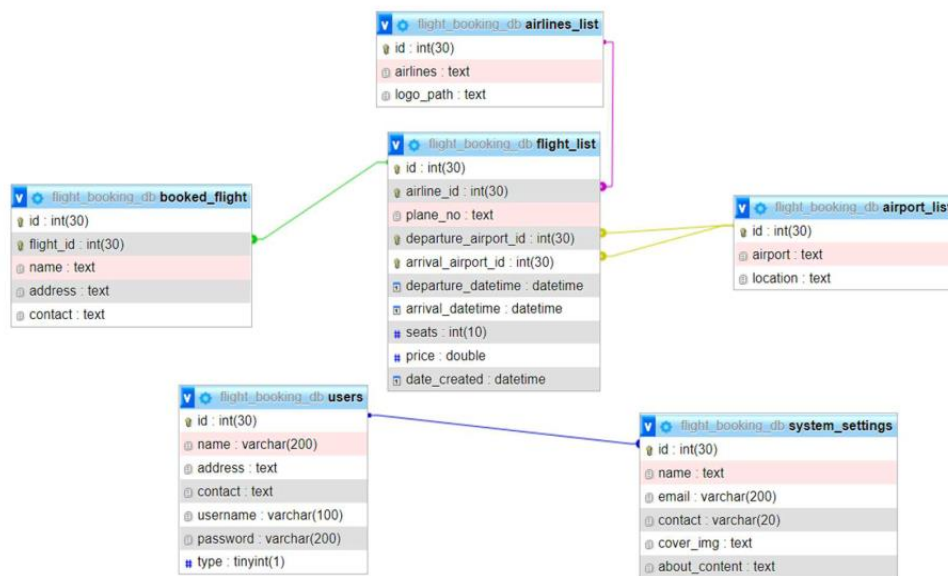
Step 9 - Mapping of Categories (Union Types): For mapping, a category whose defining superclasses have different keys, it is customary to specify a new key attribute, called a surrogate key, when creating the relation. Because the keys are different, the user cannot use just one of them exclusively to identify all tuples in the relation. A user can create a relation to correspond to the category, and include any attributes of the category in this relation.

The primary key of the new relation is the surrogate key. User also include this surrogate key as a foreign key in each relation corresponding to a superclass of the category. For a category whose superclasses have the same key, there is no need for a surrogate key.

Relational Model

In relational database theory, a relation, as originally defined by E. F. Codd, is a set of tuples (d_1, d_2, \dots, d_n) , where each element d_j is a member of D_j , a data domain. Codd's original definition notwithstanding, and contrary to the usual definition in mathematics, there is no ordering to the elements of the tuples of a relation. Instead, each element is termed an attribute value. An attribute is a name paired with a domain (nowadays more commonly referred to as a type or data type). An attribute value is an attribute name paired with an element of that attribute's domain, and a tuple is a set of attribute values in which no two distinct elements have the same name. Thus, in some accounts, a tuple is described as a function, mapping names to values.

Relational diagram for Airline Reservation System



Relational Diagram for Airline Reservation System

Participation Constraints and Cardinality Ratios

The relations between various entities are:

- **CAN MODIFY:** A relation between USERS and BOOKED_FLIGHT, USERS and AIRPORT_LIST. 'M' USERS can modify 'N' BOOKED_LIST and AIRPORT_LIST attributes. Cardinality- M:N Participation- Complete
- **ADMIN CAN CHANGE:** A relation between USERS and SYSTEM_SETTINGS. 1 USER who is an admin can modify / change 'N' SYSTEM_SETTINGS. Cardinality- 1:N Participation-Complete
- **CAN CHANGE:** A relation between USERS and FLIGHT_LIST. 'M' USERS (staff) can change the 'N' FLIGHT_LIST details. Cardinality- M:N Participation- Complete
- **BOOKED FROM:** A relation between BOOKED_FLIGHT and FLIGHT_LIST. 'M' number of flights can be booked in 'N' Flights. Cardinality- M:N Participation- Complete
- **FROM:** A relation between AIRPORT_LIST and FLIGHT_LIST. 1 Airport can contain 'N' Flights. Cardinality- 1:N Participation- Complete

- **BELONGS TO:** A relation between AIRLINES_LIST and FLIGHT_LIST. 'N' number of flights belongs to 'M' number of Airlines. Cardinality- M:N

Normalization

Normalization involves arranging attributes in relations based on dependencies between attributes, ensuring that the dependencies are properly enforced by database integrity constraints. Normalization is accomplished by applying some formal rules either by a process of synthesis or decomposition. Synthesis creates a normalized database design based on a known set of dependencies. Decomposition takes an existing (insufficiently normalized) database design and improves it based on the known set of dependencies.

1NF:

First normal form (1NF) is a property of a relation in a relational database. A relation is in first normal form if and only if the domain of each attribute contains only atomic (indivisible) values, and the value of each attribute contains only a single value from that domain. First normal form is an essential property of a relation in a relational database. Database normalization is the process of representing a database in terms of relations in standard normal forms, where first normal is a minimal requirement.

First normal form enforces these criteria:

- Eliminate repeating groups in individual tables.
- Create a separate table for each set of related data.

1. AIRLINES_LIST: id, airlines, logo_path have atomic values. Thus, the relation is in 1NF.

2. AIRPORT_LIST: id, airport, location have atomic values. Thus, the relation is in 1NF.

3. FLIGHT_LIST: id, airline_id, plane_no, departure_airport_id, arrival_airport_id, departure_datetime, arrival_datetime, seats, price, date_created have atomic values. Thus, the relation is in 1NF.

4. BOOKED_FLIGHT: id, flight_id, name, address, contact have atomic values. Thus, the relation is in 1NF.

5. USERS: id, name, address, contact, username, password, type have atomic values. Thus, the relation is in 1NF.

6. SYSTEM_SETTINGS: id, name, email, contact, cover_img, about_content have atomic values. Thus, the relation is in 1NF.

2NF:

Second normal form (2NF) is a normal form used in database normalization. A relation that is in first normal form (1NF) must meet additional criteria if it is to qualify for second normal form.

Specifically: a relation is in 2NF if it is in 1NF and no non-prime attribute is dependent on any proper subset of any candidate key of the relation. A non-prime attribute of a relation is an attribute that is not a part of any candidate key of the relation.

Put simply, a relation is in 2NF if it is in 1NF and every non-prime attribute of the relation is dependent on the whole of every candidate key. A functional dependency on part of any candidate key is a violation of 2NF. In addition to the primary key, the relation may contain other candidate keys; it is necessary to establish that no non-prime attributes have part-key dependencies on any of these candidate keys.

1. AIRLINES_LIST: id, airlines, logo_path have atomic values, no non-prime attribute is dependent on candidate key and the relation is 1NF. Hence the relation is in 2NF.

2. AIRPORT_LIST: id, airport, location have atomic values, no non-prime attribute is dependent on candidate key and the relation is 2NF.

3. FLIGHT_LIST: id, airline_id, plane_no, departure_airport_id, arrival_airport_id, departure_datetime, arrival_datetime, seats, price, date_created have atomic values, no non-prime attribute is dependent on candidate key and the relation is 1NF. Hence the relation is in 2NF.

4. BOOKED_FLIGHT: id, flight_id, name, address, contact have atomic values, no non-prime attribute is dependent on candidate key and the relation is 1NF. Hence the relation is in 2NF.

5. USERS: id, name, address, contact, username, password, type have atomic values, no non-prime attribute is dependent on candidate key and the relation is 1NF. Hence the relation is in 2NF.

6. SYSTEM_SETTINGS: id, name, email, contact, cover_img, about_content have atomic values, no non-prime attribute is dependent on candidate key and the relation is 1NF. Hence the relation is in 2NF.

3NF:

Third normal form is a normal form that is used in normalizing a database design to reduce the duplication of data and ensure referential integrity by ensuring that:

1. All the attributes in a table are determined only by the candidate keys of that relation and not by any non-prime attributes. Every non-prime attribute of R is non-transitively dependent on every key of R.
2. The entity is in 2NF

3NF was designed to improve database processing while minimizing storage costs. 3NF data modeling was ideal for online transaction processing (OLTP) applications.

1. AIRLINES_LIST: id, airlines, logo_path have atomic values, no non-prime attribute is transitively dependent on primary key and the relation is 1NF, 2NF. Hence the relation is in 3NF.

2. AIRPORT_LIST: id, airport, location have atomic values, no non-prime attribute is transitively dependent on primary key and the relation is 1NF, 2NF. Hence the relation is in 3NF.

3. FLIGHT_LIST: id, airline_id, plane_no, departure_airport_id, arrival_airport_id, departure_datetime, arrival_datetime, seats, price, date_created have atomic values, no non-prime attribute is transitively dependent on primary key and the relation is 1NF, 2NF. Hence the relation is in 3NF.

4. BOOKED_FLIGHT: id, flight_id, name, address, contact have atomic values, no non-prime attribute is transitively dependent on primary key and the relation is 1NF, 2NF. Hence the relation is in 3NF.

5. USERS: id, name, address, contact, username, password, type have atomic values, no non-prime attribute is transitively dependent on primary key and the relation is 1NF, 2NF. Hence the relation is in 3NF.

6. SYSTEM_SETTINGS: id, name, email, contact, cover_img, about_content have atomic values, no non-prime attribute is transitively dependent on the primary key and the relation is 1NF, 2NF. Hence the relation is in 3NF.

Relational Database Schema:

A schema is a collection of database objects (as far as this hour is concerned tables) associated with one particular database username. This username is called the Schema owner, or the owner of the related group of objects. The user may have one or multiple schemas in a database. The user is only associated with the schema of the same name and often the terms will be used interchangeably.

Basically, any user who creates an object has just created it in her own schema unless specifically instructs it to be created in another one. So, based on a user's privileges within the database, the user has control over objects that are created, manipulated, and deleted. A schema can consist of a single table and has no limits to the number of objects that it may contain, unless restricted by a specific database implementation.

Below points show how to go about creating a Relational Database Schema.

1. Identify all the entities in the system.
2. Identify relationships between entities.
3. Add attributes for entities

In the Reservation System Schema Diagram, there are 6 entities namely:

1. AIRLINES_LIST
2. AIRPORT_LIST
3. FLIGHT_LIST
4. BOOKED_FLIGHT
5. USERS
6. SYSTEM_SETTINGS

AIRLINES_LIST

<u>id</u>	airlines	logo_path
-----------	----------	-----------

AIRPORT_LIST

<u>id</u>	airport	location
-----------	---------	----------

FLIGHT_LIST

<u>id</u>	airline_id	plane_no	departure_airport_id	arrival_airport_id	departure_datetime	arrival_datetime	seats	price	date_created
-----------	------------	----------	----------------------	--------------------	--------------------	------------------	-------	-------	--------------

BOOKED_FLIGHT

<u>id</u>	flight_id	name	address	contact
-----------	-----------	------	---------	---------

USERS

<u>id</u>	name	address	contact	username	password	type
-----------	------	---------	---------	----------	----------	------

SYSTEM_SETTINGS

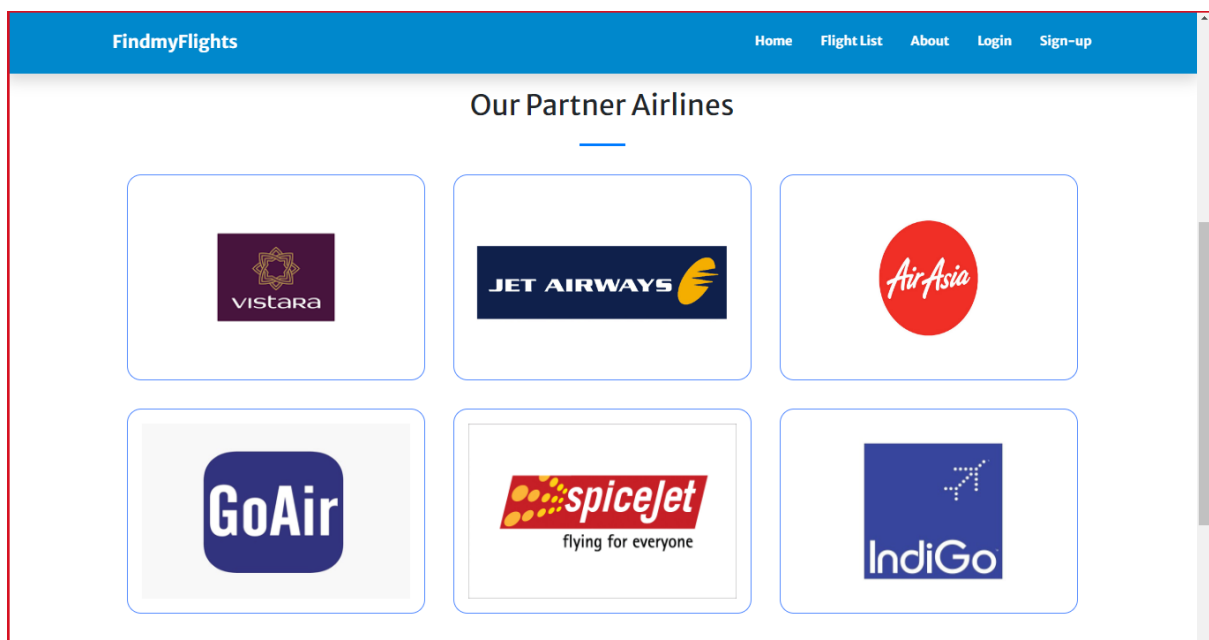
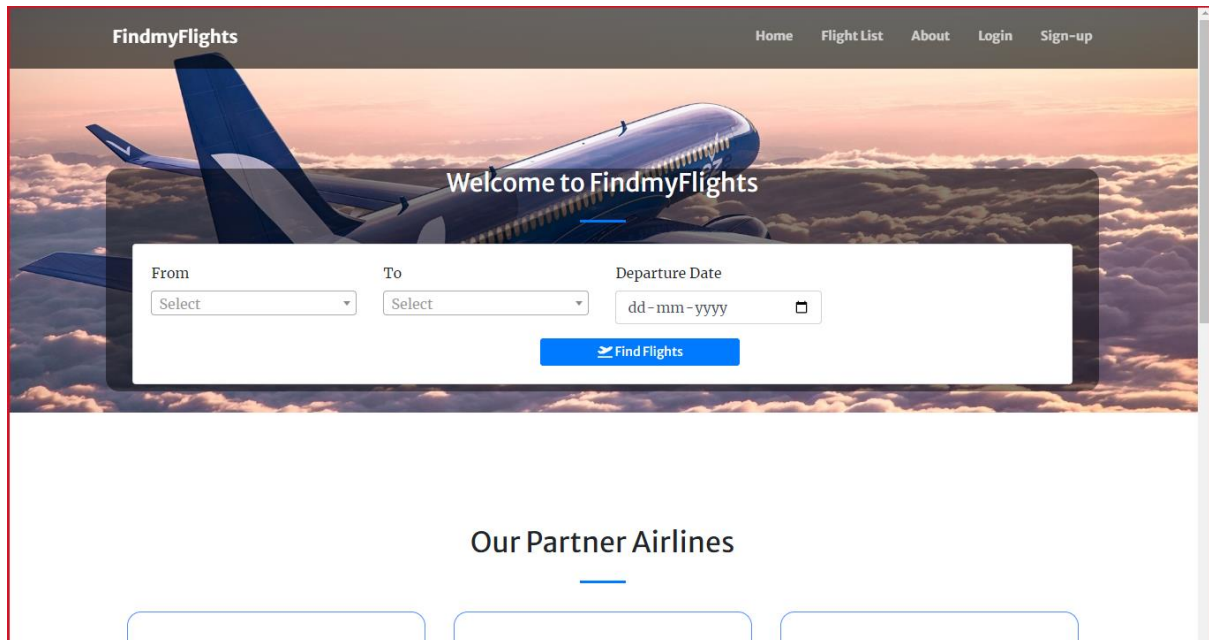
<u>id</u>	name	email	contact	cover_img	about_content
-----------	------	-------	---------	-----------	---------------

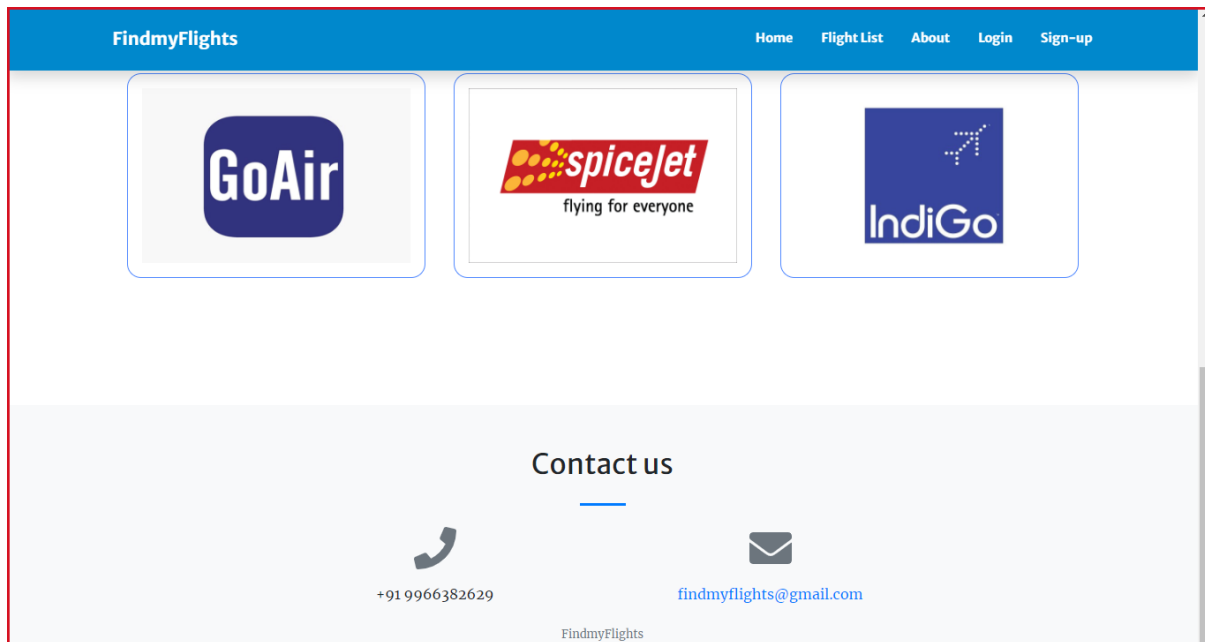
Schema Diagram for Airline Reservation System**DESCRIPTION OF THE SCHEMA DIAGRAM:**

- The AIRLINES_LIST has 3 attributes and a single primary key(id).
- The AIRPORT_LIST entity has 3 attributes and a single primary key(id).
- The FLIGHT_LIST entity has 10 attributes, a single primary key (id) and 3 foreign keys
(airline_id,departure_airport_id,arrival_airport_id).
- The BOOKED_FLIGHT entity has 5 attributes, single primary key (id) and 1 foreign key
(flight_id).
- The USERS entity has 7 attributes, a single primary key (id).
- The SYSTEM_SETTINGS has 6 attributes, one primary key(id) and one foreign key(id).

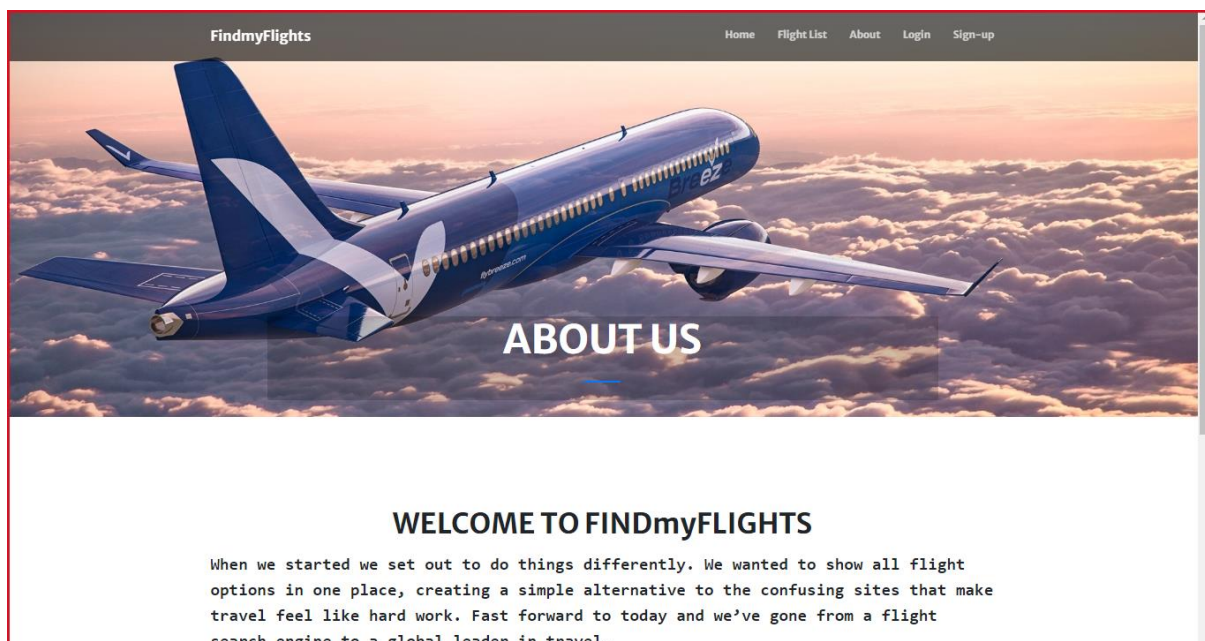
SYSTEM DESIGN

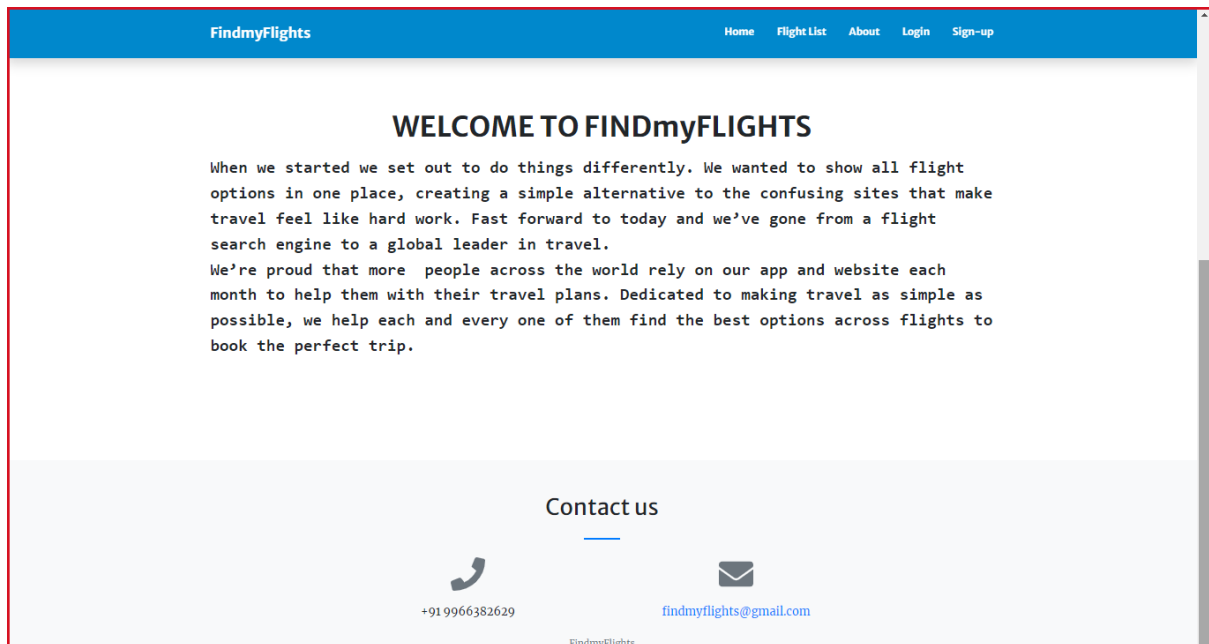
This chapter shows the implementation of every part of Airline Reservation System and it can perform all the operations that any reservation system can do. It contains screenshots of the functions added.



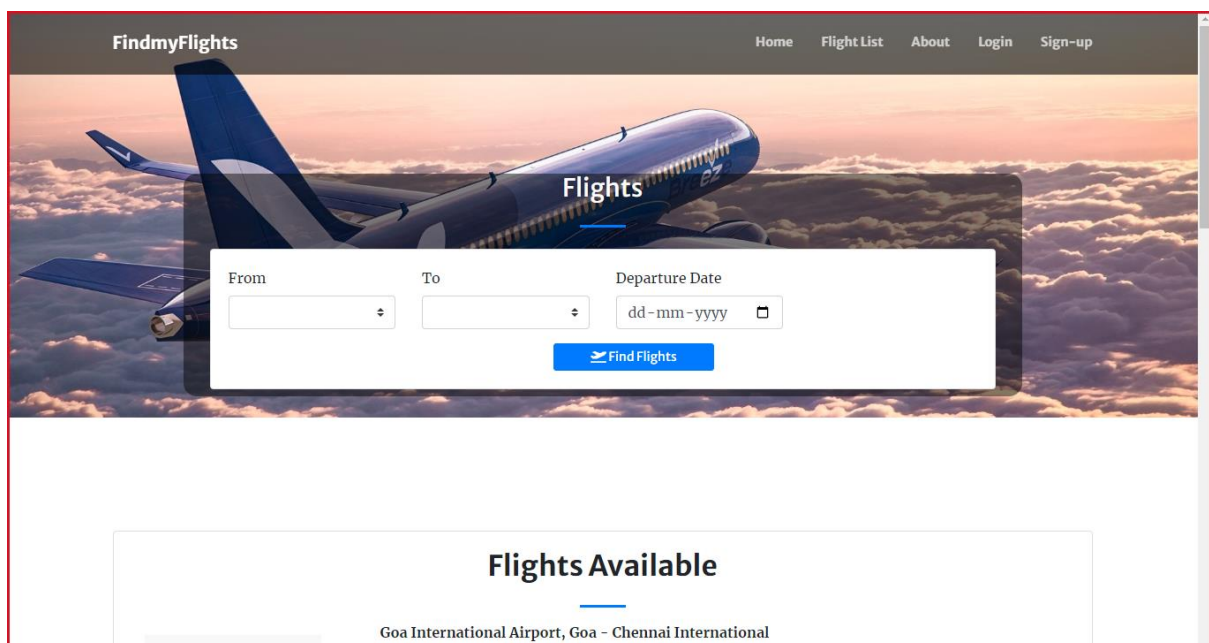


This is the default page that is loaded when user enters the website. User can use the home page to find flight availability giving the required credentials. The credentials include from(source), to(destination) and Departure date. Our partner Airlines include GoAir, SpiceJet, AirAsia and many others which have been displayed in the home page.






About Us page includes some of the information about some specified Domestic Airlines. It also includes detailed description of the website. If there is any problem or issues user can access admin's email or phone number in this page.




FindmyFlights
Home
Flight List
About
Login
Sign-up

Flights Available




Goa International Airport, Goa - Chennai International Airport, Chennai
Airline: GoAir Airlines
Departure: 07:23 PM
Arrival: Jul 06, 2022 11:23 PM
Available Seats : **250**

3,500.00
Book Now



Sardar Vallabhbhai Patel International Airport, Ahmedabad, Gujarat - Goa International Airport, Goa
Airline: Jet Airways
Departure: 07:25 PM
Arrival: Jul 08, 2022 07:25 PM
Available Seats : **180**

6,000.00
Book Now



Chhatrapati Shivaji Maharaj International Airport, Mumbai - Kempegowda International Airport Bangalore, Bangalore
Airline: Spice Jet
Departure: 08:00 AM

3,600.00

When user enters the appropriate credentials, he/she can check the available flights for their departure. The dialogue box that appears shows the airline type, Departure time, Arrival time and the available seats. For User convenience the ticket price is also provided in the box.


FindmyFlights
Flight List
About

Indira Gandhi International Airport, New Delhi - Chhatrapati Shivaji Maharaj International Airport, Mumbai

Person/s

Go
Cancel

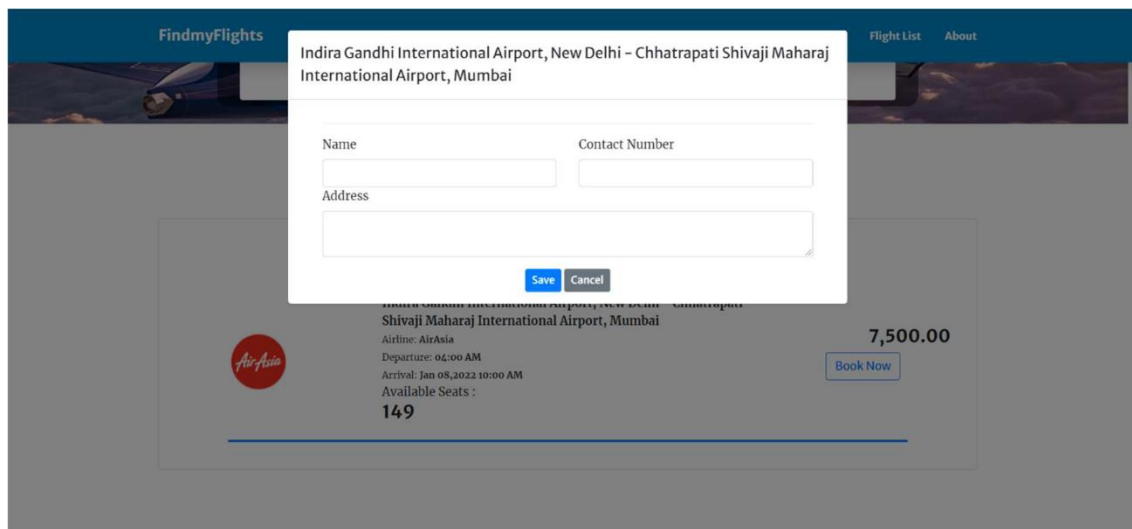
Flights Available



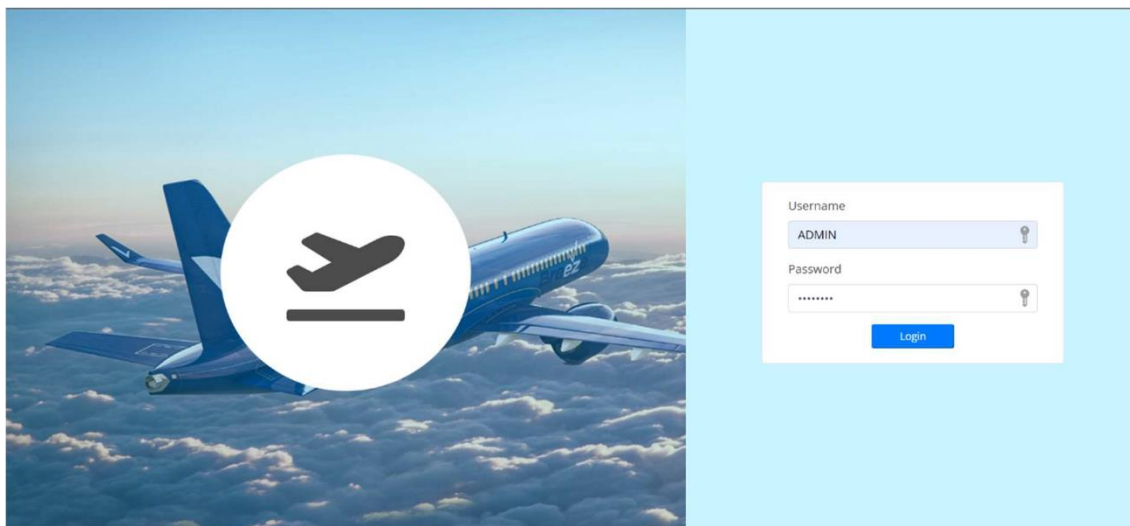
Indira Gandhi International Airport, New Delhi - Chhatrapati Shivaji Maharaj International Airport, Mumbai
Airline: AirAsia
Departure: 04:00 AM
Arrival: Jan 08, 2022 10:00 AM
Available Seats : **149**

7,500.00
Book Now

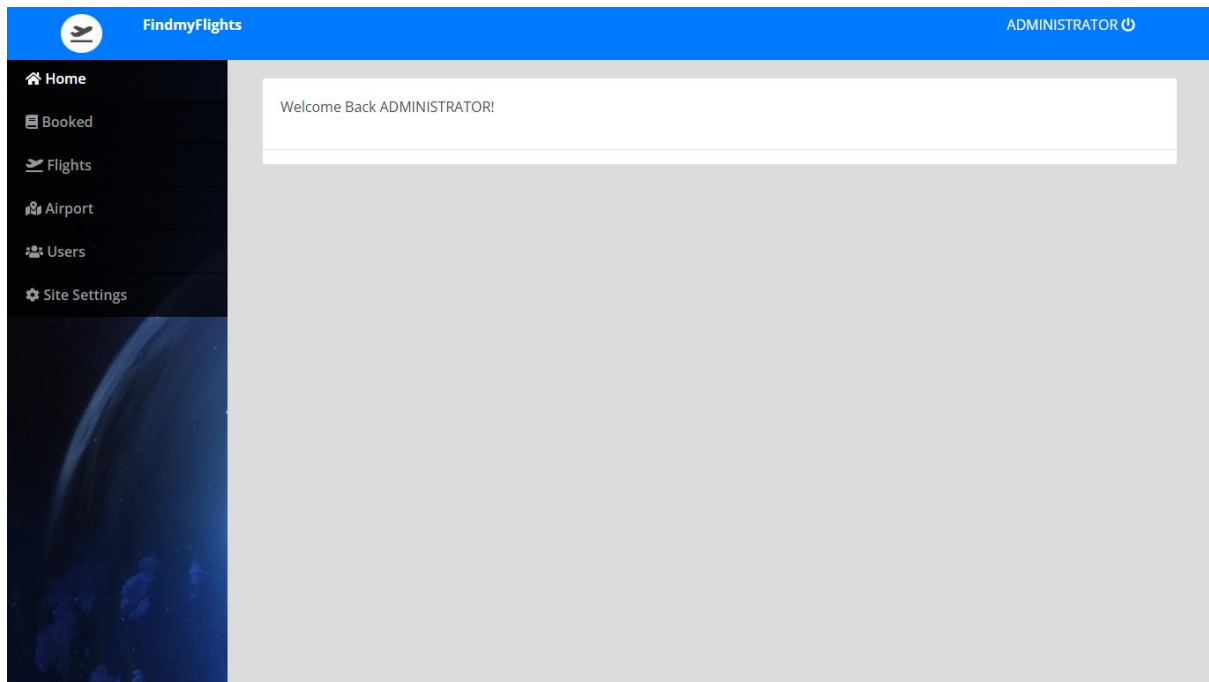
After entering the credentials and selecting the Book Now option the following window is prompted. The number of person/s boarding the flight must be entered for confirming the seats and generating tickets by calculating the price.







The user checks for seat availability and enters the required details for booking. Then the user is also prompted for entering his/her name, address, and contact number which will be saved in the database.



This page is used for the login of users for FindmyFlights website. Here admin is logging in to the website by giving the required credentials. The username of the admin is set to “admin” by default and he/she can set the password of their own. The admin will be logged in to the website and can administer functions.

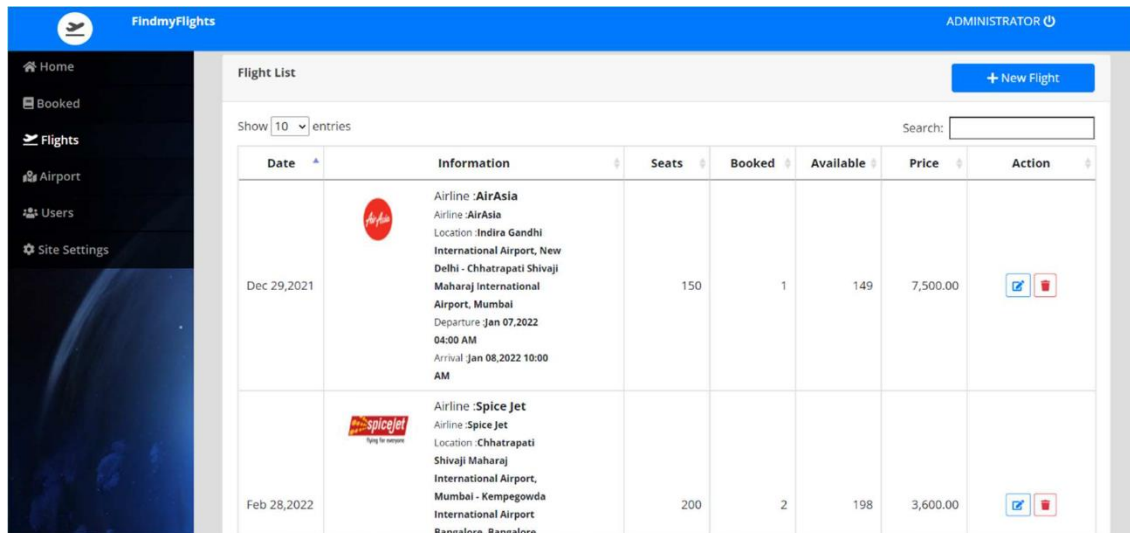


Users can login to the website by giving their respective username and password. There is some restriction access for users like they cannot view other users list. Users can access the information on flights, airport and the booked flights for travel.

#	Information	Flight Info	Action
1	Name :Eddie Munson Contact # :9845367812 Address :Bangalore	Airline :Spice Jet Plane :ABC103 Airline :Spice Jet Location :Chhatrapati Shivaji Maharaj International Airport, Mumbai - Kempegowda International Airport Bangalore, Bangalore Departure :Jun 30,2022 08:00 AM Arrival :Jul 01,2022 08:45 AM	 
2	Name :Mahima Hebbar Contact # :7878787878 Address :Bangalore	Airline :Spice Jet Plane :ABC103 Airline :Spice Jet Location :Chhatrapati Shivaji Maharaj International Airport, Mumbai - Kempegowda International Airport Bangalore, Bangalore Departure :Jun 30,2022 08:00 AM Arrival :Jul 01,2022 08:45 AM	 
		Airline :AirAsia	

This is the home page of admin who has selected booked option in the left bar to check for the reservation status and information. The booked flight list shows admin the



names, contact and address of the user who has booked it. It also includes the information about the type of flight that is booked including the date, time and an edit or remove option.



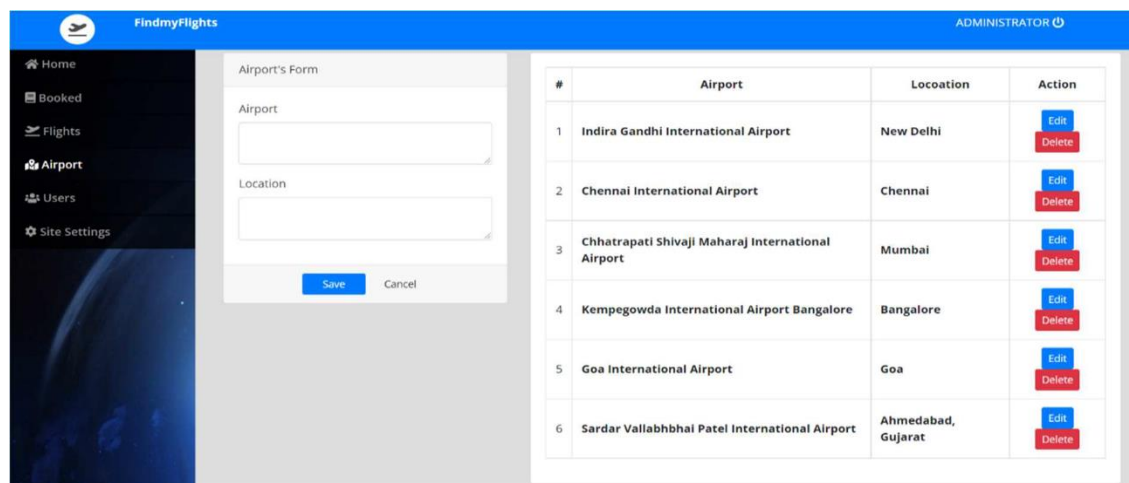
FindmyFlights ADMINISTRATOR

Flight List

Show 10 entries

Date	Information	Seats	Booked	Available	Price	Action
Dec 29, 2021	 <p>Airline : AirAsia Airline : AirAsia Location : Indira Gandhi International Airport, New Delhi - Chhatrapati Shivaji Maharaj International Airport, Mumbai Departure : Jan 07, 2022 04:00 AM Arrival : Jan 08, 2022 10:00 AM</p>	150	1	149	7,500.00	Edit Delete
Feb 28, 2022	 <p>Airline : Spice Jet Airline : Spice Jet Location : Chhatrapati Shivaji Maharaj International Airport, Mumbai - Kempegowda International Airport, Bangalore, Bangalore</p>	200	2	198	3,600.00	Edit Delete

This is the home page of admin who has selected flights option in the left bar to check for the status and information of flights. The flight list shows admin the total seats, seats booked, available seats, and price for each ticket. It also includes the information about the type of flight that can be booked including the date, time and an edit or remove option.



FindmyFlights ADMINISTRATOR

Airport's Form

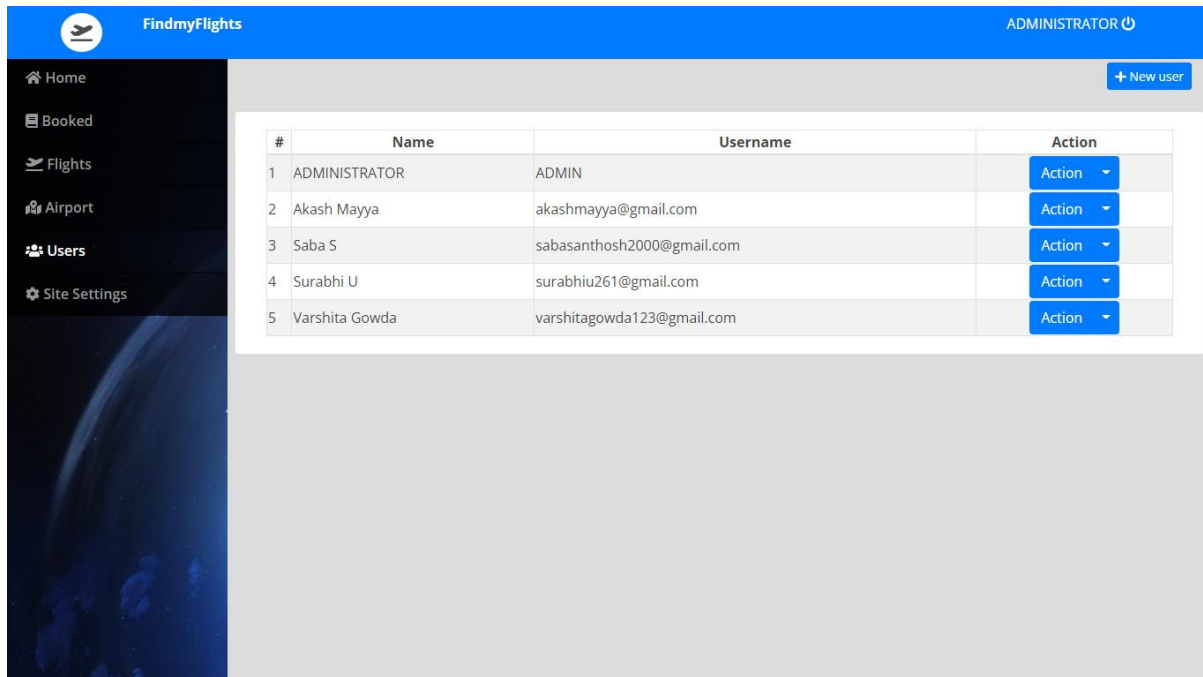
Airport

Location

[Save](#) [Cancel](#)

#	Airport	Location	Action
1	Indira Gandhi International Airport	New Delhi	Edit Delete
2	Chennai International Airport	Chennai	Edit Delete
3	Chhatrapati Shivaji Maharaj International Airport	Mumbai	Edit Delete
4	Kempegowda International Airport Bangalore	Bangalore	Edit Delete
5	Goa International Airport	Goa	Edit Delete
6	Sardar Vallabhbhai Patel International Airport	Ahmedabad, Gujarat	Edit Delete

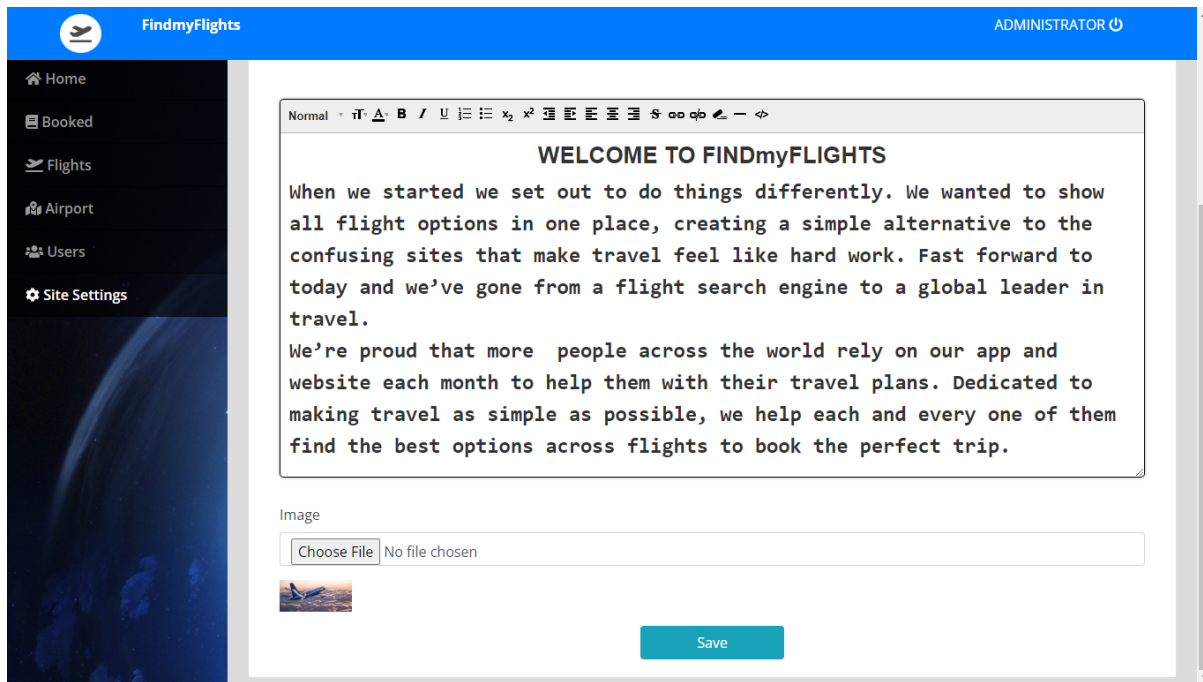
This is the home page of admin who has selected airport option in the left bar to check for the status and information of flights. The airport list shows admin the Airport name, location of it and the action that he wishes i.e either to edit or remove. Admin has an option to add the airport name and the location of it.



The screenshot displays the 'FindmyFlights' admin dashboard. The top navigation bar is blue with the 'FindmyFlights' logo on the left and 'ADMINISTRATOR' with a power icon on the right. A '+ New user' button is located in the top right corner. The left sidebar is dark blue with icons for 'Home', 'Booked', 'Flights', 'Airport', 'Users', and 'Site Settings'. The 'Airport' option is currently selected. The main content area shows a table with 5 rows of airport data. Each row includes a serial number, the airport name, its username, and an 'Action' button with a dropdown arrow.

#	Name	Username	Action
1	ADMINISTRATOR	ADMIN	Action ▾
2	Akash Mayya	akashmayya@gmail.com	Action ▾
3	Saba S	sabasanthosh2000@gmail.com	Action ▾
4	Surabhi U	surabhiu261@gmail.com	Action ▾
5	Varshita Gowda	varshitagowda123@gmail.com	Action ▾

This is the home page of admin who has selected Users option in the left bar to check for the status and information of users. The Users list shows admin the total users and their email that is been provided by them for the website. Admin has the option to take action on the users that are currently available.



This is the home page of admin who has selected site settings option in the left bar to check for the information of the site and edit the same. This gives admin a function to change the system name, email, contact and the about content of the website. Admin can also change the image of the website home page in this site settings option.

CODE

DATABASE:

```
-- phpMyAdmin SQL Dump
-- version 5.1.1
-- https://www.phpmyadmin.net/
--
-- Host: 127.0.0.1
-- Generation Time: Jan 12, 2022 at 08:43 PM
-- Server version: 10.4.22-MariaDB
-- PHP Version: 8.1.1

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
START TRANSACTION;
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `flight_booking_db`
--

--
-- Table structure for table `airlines_list`
--

CREATE TABLE `airlines_list` (
  `id` int(30) NOT NULL,
  `airlines` text NOT NULL,
  `logo_path` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```

--
-- Dumping data for table `airlines_list`
--

INSERT INTO `airlines_list` (`id`, `airlines`, `logo_path`) VALUES
(1, 'AirAsia', 'airasia-feat-logo.jpg'),
(2, 'Indigo Airlines', 'indigo-airlines-logo.jpg'),
(3, 'Spice Jet', 'GopikaChowflaDesign-Spicejet-logo.jpg'),
(4, 'Vistara', 'images.jpg'),
(5, 'GoAir Airlines', 'goair-agencies.jpg'),
(6, 'Jet Airways', '146784-jet.jpg');

-----

--
-- Table structure for table `airport_list`
--

CREATE TABLE `airport_list` (
  `id` int(30) NOT NULL,
  `airport` text NOT NULL,
  `location` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `airport_list`
--

INSERT INTO `airport_list` (`id`, `airport`, `location`) VALUES
(1, 'Indira Gandhi International Airport', 'New Delhi'),
(2, 'Chennai International Airport', 'Chennai'),
(3, 'Chhatrapati Shivaji Maharaj International Airport', 'Mumbai'),
(4, 'Kempegowda International Airport Bangalore', 'Bangalore'),
(5, 'Goa International Airport', 'Goa'),
(6, 'Sardar Vallabhbhai Patel International Airport', 'Ahmedabad, Gujarat');

```

```

-- -----

--

-- Table structure for table `booked_flight`

--

CREATE TABLE `booked_flight` (
  `id` int(30) NOT NULL,
  `flight_id` int(30) NOT NULL,
  `name` text NOT NULL,
  `address` text NOT NULL,
  `contact` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `booked_flight`

--

INSERT INTO `booked_flight` (`id`, `flight_id`, `name`, `address`, `contact`) VALUES
(4, 101, 'Deepashri', 'Bangalore', '9467453211'),
(5, 103, 'Namrutha', 'Bangalore', '7878787878'),
(6, 103, 'Kavana', 'Bangalore', '9845367812');

-- -----

--

-- Table structure for table `flight_list`

--

CREATE TABLE `flight_list` (
  `id` int(30) NOT NULL,
  `airline_id` int(30) NOT NULL,
  `plane_no` text NOT NULL,
  `departure_airport_id` int(30) NOT NULL,
  `arrival_airport_id` int(30) NOT NULL,
  `departure_datetime` datetime NOT NULL,
  `arrival_datetime` datetime NOT NULL,

```

```

`seats` int(10) NOT NULL DEFAULT 0,
`price` double NOT NULL,
`date_created` datetime NOT NULL DEFAULT current_timestamp()
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `flight_list`
--

INSERT INTO `flight_list` (`id`, `airline_id`, `plane_no`, `departure_airport_id`, `arrival_airport_id`,
`departure_datetime`, `arrival_datetime`, `seats`, `price`, `date_created`) VALUES
(101, 1, 'ABC101', 1, 3, '2022-01-07 04:00:00', '2022-01-08 10:00:00', 150, 7500, '2021-12-29 11:23:52'),
(102, 2, 'ABC102', 2, 3, '2022-02-03 11:00:00', '2022-02-04 09:00:00', 100, 5000, '2022-01-25 11:46:12'),
(103, 3, 'ABC103', 3, 4, '2022-03-16 08:00:00', '2022-03-16 08:45:00', 200, 3600, '2022-02-28 11:57:31'),
(104, 4, 'ABC104', 4, 5, '2022-01-25 19:20:30', '2022-01-26 19:20:30', 350, 4000, '2022-01-12 23:50:30'),
(105, 5, 'ABC105', 5, 2, '2022-02-12 19:23:06', '2022-01-12 23:23:07', 250, 3500, '2022-01-13 23:53:07'),
(106, 6, 'ABC106', 6, 5, '2022-03-29 19:25:04', '2022-03-30 19:25:04', 180, 6000, '2022-01-25 23:55:04');

-- -----

--

-- Table structure for table `system_settings`
--

CREATE TABLE `system_settings` (
  `id` int(30) NOT NULL,
  `name` text NOT NULL,
  `email` varchar(200) NOT NULL,
  `contact` varchar(20) NOT NULL,
  `cover_img` text NOT NULL,
  `about_content` text NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--

-- Dumping data for table `system_settings`
--

```

```
INSERT INTO `system_settings` (`id`, `name`, `email`, `contact`, `cover_img`, `about_content`) VALUES
(1, 'FindmyFlights', 'findmyflights@gmail.com', '+91 9966382629', '1642003980_airline.jpg', '<h1
style="text-align: center; background: transparent; position: relative; user-select:
auto;"><b style="user-select: auto;">WELCOME TO
FINDmyFLIGHTS</b></h1><h1 style="text-align: center; background: transparent;
position: relative; user-select: auto;"><font color="#000000" face="Open
Sans, Arial, sans-serif" style="user-select: auto;"><b style="user-select:
auto; color: rgb(51, 51, 51)"><span style="color: rgb(7, 55, 99); user-select: auto; font-
size: 14px;"><span style="font-size:20px;color: rgb(7, 55, 99); user-select:
auto;"><b style="font-size:20px;color: rgb(7, 55, 99); user-select: auto;">On our
website, you can find details of some of the specified Domestic Airlines.</b></span><br
style="user-select: auto;"></span></b></font><p style="user-
select: auto;"></p><p style="user-select: auto;"></p></h1>');

```

```
-- -----
```

```
--
```

```
-- Table structure for table `users`
```

```
--
```

```
CREATE TABLE `users` (
  `id` int(30) NOT NULL,
  `doctor_id` int(30) NOT NULL,
  `name` varchar(200) NOT NULL,
  `address` text NOT NULL,
  `contact` text NOT NULL,
  `username` varchar(100) NOT NULL,
  `password` varchar(200) NOT NULL,
  `type` tinyint(1) NOT NULL DEFAULT 2 COMMENT '1=admin , 2 = user'
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
--
```

```
-- Dumping data for table `users`
```

```
--
```

```
INSERT INTO `users` (`id`, `doctor_id`, `name`, `address`, `contact`, `username`, `password`, `type`)
VALUES
(1, 0, 'ADMINISTRATOR', '', 'ADMIN', '123ADMIN', 1),
(7, 0, 'Sarvath Anjum', 'Bangalore', '7899236478', 'sarvathanjum@gmail.com', 'sarvathanjum123', 2),
(9, 2, 'Gagan A', 'Mandya', '9878231235', 'gagana@gmail.com', 'gagana123', 2),
(10, 3, 'Sushmita S', 'Bangalore', '9923456323', 'sushmitas@gmail.com', 'sushmitas123', 2),
(11, 0, 'Pooja S', 'Bangalore', '9878784554', 'poojas@gmail.com', 'poojas123', 2),
```



```
(15, 9, 'Sparsha B S', 'Shivmoga', '9778344356', 'sparshabs@gmail.com', 'sparshabs123', 2);
```

```
--
```

```
-- Indexes for dumped tables
```

```
--
```

```
--
```

```
-- Indexes for table `airlines_list`
```

```
--
```

```
ALTER TABLE `airlines_list`  
  ADD PRIMARY KEY (`id`);
```

```
--
```

```
-- Indexes for table `airport_list`
```

```
--
```

```
ALTER TABLE `airport_list`  
  ADD PRIMARY KEY (`id`);
```

```
--
```

```
-- Indexes for table `booked_flight`
```

```
--
```

```
ALTER TABLE `booked_flight`  
  ADD PRIMARY KEY (`id`);
```

```
--
```

```
-- Indexes for table `flight_list`
```

```
--
```

```
ALTER TABLE `flight_list`  
  ADD PRIMARY KEY (`id`);
```

```
--
```

```
-- Indexes for table `system_settings`
```

```
--
```

```
ALTER TABLE `system_settings`  
  ADD PRIMARY KEY (`id`);
```

```
--
```

```
-- Indexes for table `users`  
  
--  
ALTER TABLE `users`  
  ADD PRIMARY KEY (`id`);  
  
--  
  
-- AUTO_INCREMENT for dumped tables  
--  
  
--  
  
-- AUTO_INCREMENT for table `airlines_list`  
--  
ALTER TABLE `airlines_list`  
  MODIFY `id` int(30) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;  
  
--  
  
-- AUTO_INCREMENT for table `airport_list`  
--  
ALTER TABLE `airport_list`  
  MODIFY `id` int(30) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;  
  
--  
  
-- AUTO_INCREMENT for table `booked_flight`  
--  
ALTER TABLE `booked_flight`  
  MODIFY `id` int(30) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;  
  
--  
  
-- AUTO_INCREMENT for table `flight_list`  
--  
ALTER TABLE `flight_list`  
  MODIFY `id` int(30) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=107;  
  
--  
  
-- AUTO_INCREMENT for table `system_settings`  
--  
ALTER TABLE `system_settings`
```

```

MODIFY `id` int(30) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=2;

--
-- AUTO_INCREMENT for table `users`
--

ALTER TABLE `users`
  MODIFY `id` int(30) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=16;
COMMIT;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;

```

Admin class:

```

<?php
session_start();
ini_set('display_errors', 1);
Class Action {
    private $db;

    public function __construct() {
        ob_start();
        include 'db_connect.php';

        $this->db = $conn;
    }

    function __destruct() {
        $this->db->close();
        ob_end_flush();
    }

    function login(){
        extract($_POST);

        $qry = $this->db->query("SELECT * FROM users where username =
        ".$username." and password = ".$password." ");
    }
}

```

```

        if($qry->num_rows > 0){
            foreach ($qry->fetch_array() as $key => $value) {
                if($key != 'passwords' && !is_numeric($key))
                    $_SESSION['login_'].$key = $value;
            }
            return 1;
        }else{
            return 3;
        }
    }

    function login2(){
        extract($_POST);
        $qry = $this->db->query("SELECT * FROM users where username = ".$email."
and password = ".md5($password)." ");
        if($qry->num_rows > 0){
            foreach ($qry->fetch_array() as $key => $value) {
                if($key != 'passwords' && !is_numeric($key))
                    $_SESSION['login_'].$key = $value;
            }
            return 1;
        }else{
            return 3;
        }
    }

    function logout(){
        session_destroy();
        foreach ($_SESSION as $key => $value) {
            unset($_SESSION[$key]);
        }
        header("location:login.php");
    }

    function logout2(){
        session_destroy();
        foreach ($_SESSION as $key => $value) {
            unset($_SESSION[$key]);

```

```

    }
    header("location:../index.php");
}

function save_user(){
    extract($_POST);
    $data = " name = '$name' ";
    $data .= ", username = '$username' ";
    $data .= ", password = '$password' ";
    $data .= ", type = '$type' ";
    if(empty($id)){
        $save = $this->db->query("INSERT INTO users set ".$data);
    }else{
        $save = $this->db->query("UPDATE users set ".$data." where id =
".$id);
    }
    if($save){
        return 1;
    }
}

function signup(){
    extract($_POST);
    $data = " name = '$name' ";
    $data .= ", contact = '$contact' ";
    $data .= ", address = '$address' ";
    $data .= ", username = '$email' ";
    $data .= ", password = '".md5($password)."' ";
    $data .= ", type = 3";
    $chk = $this->db->query("SELECT * FROM users where username = '$email' ")-
>num_rows;
    if($chk > 0){
        return 2;
        exit;
    }

    $save = $this->db->query("INSERT INTO users set ".$data);

```

```

        if($save){
            $qry = $this->db->query("SELECT * FROM users where username =
            ".$email." and password = ".md5($password)." ");
            if($qry->num_rows > 0){
                foreach ($qry->fetch_array() as $key => $value) {
                    if($key != 'passwords' && !is_numeric($key))
                        $_SESSION['login_'.$key] = $value;
                }
            }
            return 1;
        }
    }

function save_settings(){
    extract($_POST);
    $data = " name = '".str_replace("'", "&#x2019;", $name)." ";
    $data .= ", email = '$email' ";
    $data .= ", contact = '$contact' ";
    $data .= ", about_content = '".htmlentities(str_replace("'", "&#x2019;", $about))."'
";

    if($_FILES['img']['tmp_name'] != ""){
        $fname = strtotime(date('y-m-d
H:i')).'_'. $_FILES['img']['name'];
        $move =
move_uploaded_file($_FILES['img']['tmp_name'], './assets/img/'. $fname);
        $data .= ", cover_img = '$fname' ";
    }

    // echo "INSERT INTO system_settings set ".$data;
    $chk = $this->db->query("SELECT * FROM system_settings");
    if($chk->num_rows > 0){
        $save = $this->db->query("UPDATE system_settings set ".$data);
    }else{
        $save = $this->db->query("INSERT INTO system_settings set ".$data);
    }
}

```

```

        if($save){
            $query = $this->db->query("SELECT * FROM system_settings limit 1")-
>fetch_array();
            foreach ($query as $key => $value) {
                if(!is_numeric($key))
                    $_SESSION['setting_'. $key] = $value;
            }

            return 1;
        }
    }

function save_airlines(){
    extract($_POST);
    $data = " airlines = '$airlines' ";
    if(!empty($_FILES['img']['tmp_name'])){
        $fname = strtotime(date("Y-m-d H:i"))." _". $_FILES['img']['name'];
        $move = move_uploaded_file($_FILES['img']['tmp_name'],
'../assets/img/'.$fname);
        if($move){
            $data .= ", logo_path = '$fname' ";
        }
    }
    if(empty($id)){
        $save = $this->db->query("INSERT INTO airlines_list set ".$data);
    }else{
        $save = $this->db->query("UPDATE airlines_list set ".$data." where
id=".$id);
    }
    if($save)
        return 1;
}

function delete_airlines(){
    extract($_POST);
    $delete = $this->db->query("DELETE FROM airlines_list where id = ".$id);

```

```

        if($delete)
            return 1;
    }

    function save_airports(){
        extract($_POST);
        $data = " airport = '$airport' ";
        $data .= ", location = '$location' ";

        if(empty($id)){
            $save = $this->db->query("INSERT INTO airport_list set ".$data);
        }else{
            $save = $this->db->query("UPDATE airport_list set ".$data." where
id=".$id);
        }
        if($save)
            return 1;
    }

    function delete_airports(){
        extract($_POST);
        $delete = $this->db->query("DELETE FROM airport_list where id = ".$id);
        if($delete)
            return 1;
    }

    function save_flight(){
        extract($_POST);
        $data = " airline_id = '$airline' ";
        $data .= ", plane_no = '$plane_no' ";
        $data .= ", departure_airport_id = '$departure_airport_id' ";
        $data .= ", arrival_airport_id = '$arrival_airport_id' ";
        $data .= ", departure_datetime = '$departure_datetime' ";
        $data .= ", arrival_datetime = '$arrival_datetime' ";
        $data .= ", seats = '$seats' ";
        $data .= ", price = '$price' ";

        if(empty($id)){

```



```

        // echo "INSERT INTO flight_list set ".$data;
        $save = $this->db->query("INSERT INTO flight_list set ".$data);
    }else{
        $save = $this->db->query("UPDATE flight_list set ".$data." where
id=".$id);
    }
    if($save)
        return 1;
}

function delete_flight(){
    extract($_POST);
    $delete = $this->db->query("DELETE FROM flight_list where id = ".$id);
    if($delete)
        return 1;
}

function book_flight(){
    extract($_POST);
    foreach ($name as $k => $value) {
        $data = " flight_id = $flight_id ";
        $data .= " , name = '$name[$k]' ";
        $data .= " , address = '$address[$k]' ";
        $data .= " , contact = '$contact[$k]' ";

        $save[] = $this->db->query("INSERT INTO booked_flight set ".$data);
    }
    if(isset($save))
        return 1;
}

function update_booked(){
    extract($_POST);
    $data = " name = '$name' ";
    $data .= " , address = '$address' ";
    $data .= " , contact = '$contact' ";

```

```

        $save= $this->db->query("UPDATE booked_flight set ".$data." where id
        =".$id);

        if($save)
            return 1;
    }

    function delete_booked(){
        extract($_POST);
        $delete = $this->db->query("DELETE FROM booked_flight where id = ".$id);
        if($delete)
            return 1;
    }
}

```

AIRLINES:

```
<?php include('db_connect.php');?>
```

```
<div class="container-fluid">
```

```
    <div class="col-lg-12">
```

```
        <div class="row">
```

```
            <!-- FORM Panel -->
```

```
            <div class="col-md-4">
```

```
                <form action="" id="manage-airlines">
```

```
                    <div class="card">
```

```
                        <div class="card-header">
```

```
                            Airlines Form
```

```
                        </div>
```

```
                        <div class="card-body">
```

```
                            <input type="hidden" name="id">
```

```
                            <div class="form-group">
```

```
                                <label class="control-
```

```
label">Airlines</label>
```

```
                                <textarea name="airlines" id=""
```

```
                                cols="30" rows="2" class="form-control"></textarea>
```

```

        </div>
        <div class="form-group">
            <label for="" class="control-
label">Logo</label>
            <input type="file" class="form-
control" name="img" onchange="displayImg(this,$(this))">
        </div>
        <div class="form-group">
            <img src="" alt="" id="cimg">
        </div>

    </div>

    <div class="card-footer">
        <div class="row">
            <div class="col-md-12">
                <button class="btn btn-sm btn-
primary col-sm-3 offset-md-3"> Save</button>
                <button class="btn btn-sm btn-
default col-sm-3" type="button" onclick="_reset()"> Cancel</button>
            </div>
        </div>
    </div>
</div>
</div>
</form>
</div>
<!-- FORM Panel -->

<!-- Table Panel -->
<div class="col-md-8">
    <div class="card">
        <div class="card-body">
            <table class="table table-bordered table-hover">
                <thead>
                    <tr>

```

```

center">#</th>
center">Image</th>
center">Name</th>
center">Action</th>

<th class="text-
center">#</th>
<th class="text-
center">Image</th>
<th class="text-
center">Name</th>
<th class="text-
center">Action</th>

</tr>
</thead>
<tbody>
<?php
$i = 1;
$cats = $conn->query("SELECT *
FROM airlines_list order by id asc");
while($row=$cats-
>fetch_assoc()):
?>
<tr>
<td class="text-
center"><?php echo $i++ ?></td>
<td class="text-center">

</td>
<td class="">
<b><?php echo
$row['airlines'] ?></b>
</td>
<td class="text-center">
<button class="btn
btn-sm btn-primary edit_airline" type="button" data-id="<?php echo $row['id'] ?>" data-
airlines="<?php echo $row['airlines'] ?>" data-logo_path="<?php echo $row['logo_path'] ?>"
>Edit</button>
<button class="btn
btn-sm btn-danger delete_airline" type="button" data-id="<?php echo $row['id']
?>">Delete</button>
</td>
</tr>
<?php endwhile; ?>

```

```

        </tbody>
    </table>
</div>
</div>
</div>
<!-- Table Panel -->
</div>
</div>

</div>
<style>

    td{
        vertical-align: middle !important;
    }
    td p{
        margin: unset
    }
    img{
        max-width:100px;
        max-height: :150px;
    }
</style>
<script>
    function _reset(){
        $('#cimg').attr('src','');
        $('[name="id"]').val("");
        $('#manage-airlines').get(0).reset();
    }

    $('#manage-airlines').submit(function(e){
        e.preventDefault()
        start_load()
        $.ajax({
            url:'ajax.php?action=save_airlines',

```

```

        data: new FormData($(this)[0]),
        cache: false,
        contentType: false,
        processData: false,
        method: 'POST',
        type: 'POST',
        success:function(resp){
            if(resp==1){
                alert_toast("Data successfully added",'success')
                setTimeout(function(){
                    location.reload()
                },1500)
            }
            else if(resp==2){
                alert_toast("Data successfully updated",'success')
                setTimeout(function(){
                    location.reload()
                },1500)
            }
        }
    })
})

$('#.edit_airline').click(function(){
    start_load()
    var cat = $('#manage-airlines')
    cat.get(0).reset()
    cat.find("[name='id']").val($(this).attr('data-id'))
    cat.find("[name='airlines']").val($(this).attr('data-airlines'))
    cat.find("#cimg").attr("src","./assets/img/"+$(this).attr('data-logo_path'))
    end_load()
})

$('#.delete_airline').click(function(){

```

```

        _conf("Are you sure to delete this airline?", "delete_airline", [$(this).attr('data-
id')]))
    })
    function displayImg(input, _this) {
    if (input.files && input.files[0]) {
        var reader = new FileReader();
        reader.onload = function (e) {
            $('#cimg').attr('src', e.target.result);
        }

        reader.readAsDataURL(input.files[0]);
    }
}

function delete_airline($id){
    start_load()
    $.ajax({
        url:'ajax.php?action=delete_airlines',
        method:'POST',
        data:{id:$id},
        success:function(resp){
            if(resp==1){
                alert_toast("Data successfully deleted", 'success')
                setTimeout(function(){
                    location.reload()
                },1500)
            }
        }
    })
}
}
</script>

```

FLIGHTS:

```
<?php include('db_connect.php');?>
```

```

<div class="container-fluid">

    <div class="col-lg-12">
        <div class="row">
            <!-- FORM Panel -->
            <div class="col-md-4">
                <form action="" id="manage-airlines">
                    <div class="card">
                        <div class="card-header">
                            Airlines Form
                        </div>
                        <div class="card-body">
                            <input type="hidden" name="id">
                            <div class="form-group">
                                <label class="control-
label">Airlines</label>
                                <textarea name="airlines" id=""
cols="30" rows="2" class="form-control"></textarea>
                            </div>
                            <div class="form-group">
                                <label for="" class="control-
label">Logo</label>
                                <input type="file" class="form-
control" name="img" onchange="displayImg(this,$(this))">
                            </div>
                            <div class="form-group">
                                <img src="" alt="" id="cimg">
                            </div>
                        </div>
                    </div>
                </form>
            </div>
            <div class="card-footer">
                <div class="row">
                    <div class="col-md-12">
                        <button class="btn btn-sm btn-
primary col-sm-3 offset-md-3"> Save</button>

```



```
  | = $row['airlines'] ? | Edit Delete |
```

```


```

```

td{
    vertical-align: middle !important;
}
td p{
    margin: unset
}

```

```

        img{
            max-width:100px;
            max-height: :150px;
        }
</style>
<script>
    function _reset(){
        $('#cimg').attr('src','');
        $('[name="id"]').val("");
        $('#manage-airlines').get(0).reset();
    }

    $('#manage-airlines').submit(function(e){
        e.preventDefault()
        start_load()
        $.ajax({
            url:'ajax.php?action=save_airlines',
            data: new FormData($(this)[0]),
            cache: false,
            contentType: false,
            processData: false,
            method: 'POST',
            type: 'POST',
            success:function(resp){
                if(resp==1){
                    alert_toast("Data successfully added",'success')
                    setTimeout(function(){
                        location.reload()
                    },1500)
                }
                else if(resp==2){
                    alert_toast("Data successfully updated",'success')
                    setTimeout(function(){
                        location.reload()

```

```

        },1500)

    }

}

}))

))

$('#edit_airline').click(function(){
    start_load()
    var cat = $('#manage-airlines')
    cat.get(0).reset()
    cat.find("[name='id']").val($(this).attr('data-id'))
    cat.find("[name='airlines']").val($(this).attr('data-airlines'))
    cat.find("#cimg").attr("src","../assets/img/"+$(this).attr('data-logo_path'))
    end_load()
})

$('#delete_airline').click(function(){
    _conf("Are you sure to delete this airline?", "delete_airline",[$(this).attr('data-
id')]))
})

function displayImg(input,_this) {
if (input.files && input.files[0]) {
    var reader = new FileReader();
    reader.onload = function (e) {
        $('#cimg').attr('src', e.target.result);
    }

    reader.readAsDataURL(input.files[0]);
}
}

function delete_airline($id){
    start_load()
    $.ajax({
        url:'ajax.php?action=delete_airlines',
        method:'POST',
        data:{id:$id},

```

```

                success:function(resp){
                    if(resp==1){
                        alert_toast("Data successfully deleted",'success')
                        setTimeout(function(){
                            location.reload()
                        },1500)
                    }
                }
            })
        }
    }
</script>

```

LOGIN:

```

<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8">
    <meta content="width=device-width, initial-scale=1.0" name="viewport">

    <title>Admin | FindmyFlight</title>

    <?php include('./header.php'); ?>
    <?php include('./db_connect.php'); ?>
    <?php
        session_start();
        if(isset($_SESSION['login_id']))
            header("location:index.php?page=home");

    ?>

</head>

```

```
<style>
  body{
    width: 100%;
    height: calc(100%);
    /*background: #007bff;*/
  }
  main#main{
    width:100%;
    height: calc(100%);
    background:white;
  }
  #login-right{
    position: absolute;
    right:0;
    width:40%;
    height: calc(100%);
    background:white;
    display: flex;
    align-items: center;
  }
  #login-left{
    position: absolute;
    left:0;
    width:60%;
    height: calc(100%);
    background:white;
    display: flex;
    align-items: center;
    background: url(../assets/img/1642003860_websiteairline.jpg);
    background-repeat: no-repeat;
    background-size: cover;
  }
  #login-right .card{
    margin: auto;
    z-index: 1
  }
```

```

    }
    .logo {
margin: auto;
font-size: 8rem;
background: white;
padding: .5em 0.7em;
border-radius: 50% 50%;
color: #000000b3;
z-index: 10;
}
div#login-left::before,div#login-right::before {
content: "";
position: absolute;
top: 0;
left: 0;
width: calc(100%);
height: calc(100%);
background: #00c4ff36;
}

</style>

<body>

<main id="main" class=" bg-dark">
    <div id="login-left">
        <div class="logo">
            <span class="fa fa-plane-departure"></span>
        </div>
    </div>
    <div id="login-right">
        <div class="card col-md-8">
            <div class="card-body">
                <form id="login-form" >

```

```

                                <div class="form-group">
                                    <label for="username" class="control-
label">Username</label>
                                    <input type="text" id="username"
name="username" class="form-control">
                                </div>
                                <div class="form-group">
                                    <label for="password" class="control-
label">Password</label>
                                    <input type="password" id="password"
name="password" class="form-control">
                                </div>
                                <center><button class="btn-sm btn-block btn-
wave col-md-4 btn-primary">Login</button></center>
                                </form>
                            </div>
                        </div>
                    </div>
                </div>

</main>

<a href="#" class="back-to-top"><i class="icofont-simple-up"></i></a>

</body>
<script>
    $('#login-form').submit(function(e){
        e.preventDefault()
        $('#login-form button[type="button"]').attr('disabled',true).html('Logging in...');
        if($(this).find('.alert-danger').length > 0 )
            $(this).find('.alert-danger').remove();
        $.ajax({
            url:'ajax.php?action=login',
            method:'POST',
            data:$(this).serialize(),
            error:err=>{

```



```

        console.log(err)

        $('#login-form button[type="button"]').removeAttr('disabled').html('Login');

    },
    success:function(resp){
        if(resp == 1){
            location.href ='index.php?page=home';
        }else if(resp == 2){
            location.href ='voting.php';
        }else{
            $('#login-form').prepend('<div class="alert alert-
danger">Incorrect Username or Password.</div>')
            $('#login-form
button[type="button"]').removeAttr('disabled').html('Login');
        }
    }
})
})
</script>
</html>

```

HOME:

```

<?php
include 'admin/db_connect.php';
?>
<style>
#portfolio .img-fluid{
    width: calc(100%);
    height: 30vh;
    z-index: -1;
    position: relative;
    padding: 1em;
}
</style>
<header class="masthead">

```

```

<div class="container-fluid h-100">
    <div class="row h-100 align-items-center justify-content-center text-center">
        <div class="col-lg-10 align-self-end mb-4 page-title">
            <h3 class="text-white">Welcome to <?php echo $_SESSION['setting_name'];
?></h3>
            <hr class="divider my-4" />
            <div class="col-md-12 mb-2 text-left">
                <div class="card">
                    <div class="card-body">
                        <form id="manage-filter" action="index.php?page=flights"
method="POST">
                            <div class="row form-group">
                                <div class="col-sm-3">
                                    <label for="" class="control-label">From</label>
                                    <select name="departure_airport_id" id="departure_location"
class="custom-select browser-default select2">
                                        <option value=""></option>
                                        <?php
                                            $airport = $conn->query("SELECT * FROM airport_list order by
airport asc");
                                            while($row = $airport->fetch_assoc()):
                                                ?>
                                                    <option value="<?php echo $row['id'] ?>" <?php echo
isset($departure_airport_id) && $departure_airport_id == $row['id'] ? "selected" : " ?>><?php
echo $row['location'].", ".$row['airport'] ?></option>
                                                    <?php endwhile; ?>
                                                </select>
                                            </div>
                                            <div class="col-sm-3">
                                                <label for="" class="control-label">To</label>
                                                <select name="arrival_airport_id" id="arrival_airport_id"
class="custom-select browser-default select2">
                                                    <option value=""></option>
                                                    <?php
                                                        $airport = $conn->query("SELECT * FROM airport_list order by
airport asc");

```

[illegible]

```

<div class="row">
    <div class="col-lg-12 text-center">
        <h2 class="mb-4">Our Partner Airlines</h2>
        <hr class="divider">

    </div>
</div>
<div class="row no-gutters">
    <?php
        $cats = $conn->query("SELECT * FROM airlines_list order by rand() asc");
        while($row=$cats->fetch_assoc()):
            ?>
            <div class="col-lg-4 col-sm-6">
                <div class="portfolio-box" href="#">
                    

                    <div class="port-content text-center">

                </div>
            </div>
        <?php endwhile; ?>

    </div>
</div>
</div>
<script>

    $('view_prod').click(function(){
        uni_modal_right('Product','view_prod.php?id='+$(this).attr('data-id'))
    })
    $('select2').select2({
        placeholder:'Select Departure',
        width:'100%'
    })

```

```

    })
    $('[name="trip"]').on("keypress change keyup",function(){
        if($(this).val() == 1){
            $('#rdate').hide()
        }else{
            $('#rdate').show()
        }
    })
</script>
</section>

```

INDEX:

```

<!DOCTYPE html>
<html lang="en">
    <?php
        session_start();
        ob_start();
        include('header.php');
        include('admin/db_connect.php');

        $query = $conn->query("SELECT * FROM system_settings limit 1")->fetch_array();
        foreach ($query as $key => $value) {
            if(!is_numeric($key))
                $_SESSION['setting_'].$key = $value;
        }
        ob_end_flush();
    ?>

    <style>
        header.masthead {
            background: url(assets/img/<?php echo $_SESSION['setting_cover_img'] ?>);
            background-repeat: no-repeat;
            background-size: cover;
        }
    </style>

```

```

</style>
<body id="page-top">
    <!-- Navigation-->
    <div class="toast" id="alert_toast" role="alert" aria-live="assertive" aria-atomic="true">
        <div class="toast-body text-white">
        </div>
    </div>

    <nav class="navbar navbar-expand-lg navbar-light fixed-top py-3" id="mainNav">
        <div class="container">
            <a class="navbar-brand js-scroll-trigger" href="/"><?php echo
$_SESSION['setting_name'] ?></a>

            <button class="navbar-toggler navbar-toggler-right" type="button" data-
toggle="collapse" data-target="#navbarResponsive" aria-controls="navbarResponsive" aria-
expanded="false" aria-label="Toggle navigation"><span class="navbar-toggler-
icon"></span></button>

            <div class="collapse navbar-collapse" id="navbarResponsive">
                <ul class="navbar-nav ml-auto my-2 my-lg-0">

                    <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="index.php?page=home">Home</a></li>

                    <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="index.php?page=flights"></span>Flight List</a></li>

                    <li class="nav-item"><a class="nav-link js-scroll-trigger"
href="index.php?page=about">About</a></li>

                </ul>
            </div>
        </div>
    </nav>

    <?php
$page = isset($_GET['page']) ? $_GET['page'] : "home";
include $page.'.php';
?>

<div class="modal fade" id="confirm_modal" role='dialog'>

```

```

<div class="modal-dialog modal-md" role="document">
  <div class="modal-content">
    <div class="modal-header">
      <h5 class="modal-title">Confirmation</h5>
    </div>
    <div class="modal-body">
      <div id="delete_content"></div>
    </div>
    <div class="modal-footer">
      <button type="button" class="btn btn-primary" id='confirm'
onclick="">Continue</button>
      <button type="button" class="btn btn-secondary" data-dismiss="modal">Close</button>
    </div>
  </div>
</div>
<div class="modal fade" id="uni_modal" role='dialog'>
  <div class="modal-dialog modal-md" role="document">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title"></h5>
      </div>
      <div class="modal-body">
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-primary" id='submit' onclick="$('#uni_modal
form').submit()>Save</button>
        <button type="button" class="btn btn-secondary" data-
dismiss="modal">Cancel</button>
      </div>
    </div>
  </div>
</div>
<div class="modal fade" id="uni_modal_right" role='dialog'>
  <div class="modal-dialog modal-full-height modal-md" role="document">
    <div class="modal-content">

```

```

<div class="modal-header">
<h5 class="modal-title"></h5>
<button type="button" class="close" data-dismiss="modal" aria-label="Close">
  <span class="fa fa-arrow-right"></span>
</button>
</div>
<div class="modal-body">
</div>
</div>
</div>
</div>
<div id="preloader"></div>
<footer class="bg-light py-5">
  <div class="container">
    <div class="row justify-content-center">
      <div class="col-lg-8 text-center">
        <h2 class="mt-0">Contact us</h2>
        <hr class="divider my-4" />
      </div>
    </div>
    <div class="row">
      <div class="col-lg-4 ml-auto text-center mb-5 mb-lg-0">
        <i class="fas fa-phone fa-3x mb-3 text-muted"></i>
        <div><?php echo $_SESSION['setting_contact'] ?></div>
      </div>
      <div class="col-lg-4 mr-auto text-center">
        <i class="fas fa-envelope fa-3x mb-3 text-muted"></i>
        <!-- Make sure to change the email address in BOTH the anchor text and the link
target below!-->
        <a class="d-block" href="mailto:<?php echo $_SESSION['setting_email']
?>"><?php echo $_SESSION['setting_email'] ?></a>
      </div>
    </div>
  </div>
<br>

```



```
<div class="container"><div class="small text-center text-muted"> <?php echo  
$_SESSION['setting_name'] ?> </div></div>
```

```
</footer>
```

```
<?php include('footer.php') ?>
```

```
</body>
```

```
<?php $conn->close() ?>
```

```
</html>
```

SOFTWARE TESTING

Home Module

TEST CASE ID	TEST CONDITION	EXPECTED RESULT	TEST RESULT
1.	On click on the home button	It displays the option of booking flights for the user according to their requirements	Successful
2.	On click on the Flight List button	It displays the Flight available for the user according to their required data and time	Successful
3.	On click on the About button	The about page is displayed	Successful
4.	On clicking on the Login page	It displays a page where the user should enter the username and password to log in.	Successful
5.	On clicking on the Sign-up button	New page should be displayed where new users can create a profile.	Successful

Admin Module

TEST CASE ID	TEST CONDITION	EXPECTED RESULT	TEST RESULT
1.	When booked button is clicked.	It displays the table which contains the list of booked flights.	Successful
2.	When clicked on flights.	It displays the flights available with details such as seats, number of available seats, price and number of seats which have been booked.	Successful

3.	When clicked on Airport.	It displays a list of Airports.	Successful
4.	Clicking on the User button.	It displayed the user who have already created a profile.	Successful
5.	When clicked on site-settings.	It displays the page where the Admin can change the displaying details of the about page.	Successful

User Module

TEST CASE ID	TEST CONDITION	EXPECTED RESULT	TEST RESULT
1.	When clicked on the Booked button.	It displays the flights booked.	Successful
2.	When clicked on the Flights button.	It displays the flights available with details such as seats, number of available seats, price and number of seats which have been booked.	Successful
3.	When clicked on the Airport button.	Displays all the Airports that are listed	Successful

FUTURE ENHANCEMENT

Each day technologies change to suit the current market trends. This newly developed software must suit itself to sustain itself in its environment. Thus, the software can be enhanced in the future as follows:

1. Database backup and portability.
2. Enhance the communication over the network such that the database contents can be accessed anywhere with the use of servers.
3. Recover the application during crashes such that the database is kept secure.
4. Database security and permission management.
5. Alert message to the administrator on any change of database contents.
6. Mobile app development to manage the system online.

CONCLUSION

In this project, the Airline Reservation system is created using MY SQL in the backend and HTML in the front end. And they are connected using PHP and JavaScript codes. SQL is used for storing the data in the backend (DBMS). HTML is used for creating front-end pages. PHP code is used to link/connect the backend and frontend together.

This system allows customers to look for all the flights available for the entries like from and to airports and departure dates. Then when the list of available flights is listed, the customer can choose a flight of his/her choice and can book flights by giving credentials like name contact number and address, etc. The project has a very vast scope in the future. The project is flexible in terms of expansion and has advantages like:

- Airline Reservation System helps the user save time and effort by reducing the processing time and volume of errors.
- The web-based Airline Reservation system has offered an advantage to both customers and Flight companies to effectively manage the business and satisfy customers' needs at the click of a button.
- Also helps in generating report flexibility and an easy implementation environment.

- This system, therefore, helps people to book flights from anywhere easily through the portal.
- This system provides good GUI support.

REFERENCES

Text Books:

- PHP and MySQL Web Development by Luke Welling published by: Pearson's Publications
- 2. PHP: The Complete Reference by Steven Holzner published by: Tata McGraw-Hill Educations
- Database System: The Complete Book by Jeffrey Ullman published by: Tata McGraw-Hill

Web Site Links:

- www.ERDPlus.com
- www.DrawIO.com
- www.spoken-tutorials.org
- www.w3schools.com

YouTube Links:

<https://youtu.be/ueWpNe0PG34>