PREFACE

As a part of the B.Tech. Curriculum and in order to gain practical knowledge in the field of technology, we are required to make a report on “Industrial Training in Core Java”. The basic objective behind doing this project report is to get knowledge of different tools of technology.

The project submitted in this report is “Air Ticket Booking System”. This project provides a simple user interface to book airplane tickets. 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.

**CONTENTS**

1. INSTITUTE PROFILE
   1. About TCS- ION
   2. History & Recent Developments
2. LIST OF TABLES
3. LIST OF FIGURES
4. INTRODUCTION

4.1ABOUT JAVA

* + 1. Creation of Java
    2. Java’s Lineage
    3. Features Of Java
  1. SWING
     1. The Origins Of Swing
     2. Swing is built on AWT
     3. Two Key Features of Swing
     4. MVC Connection
     5. Components
  2. JDBC

4.3.1 Connectivity to Oracle

1. SOFTWARE DESCRIPTION
   1. NetBeans IDE 8.2
   2. Oracle 10g Express Edition
2. PROJECT DESCRIPTION
   1. Overview
   2. Module Description
   3. Data Flow Diagram
   4. System Flow Diagram
   5. System Design

6.5.1 Screenshots of Interface

* 1. Database Design

6.6.1 Tables

1. CONCLUSION
2. BIBLIOGRAPHY

**1. INSTITUTE PROFILE**

**ABOUT TCS - iON**

CMC Limited is a leading IT solutions company and a subsidiary of Tata Consultancy Services Limited (TCS Ltd), one of the world's leading information technology consulting, services and business process outsourcing organisations. We are a part of the Tata group, India's best-known business conglomerate. Today, CMC Limited, an ISO 9001:2000, certified and CMMI Level V accredited organisation, is positioned as a premier IT solutions provider in the fast growing and competitive IT market. iON is a strategic unit of Tata Consultancy Services focused on Manufacturing Industries (SMB), Educational Institutions and Examination Boards. TCS iON provides technology by means of a unique IT-as-a-Service model, offering end-to-end business solutions. It caters to the needs of multiple industry segments, through innovative, easy-to-use, secured, integrated, hosted solutions in a build-as-you-grow, pay-as-you-use business model. TCS iON serves its clients with the help of best practices gained through TCS' global experience, domestic market reach, skills, and delivery capabilities. iON's Cloud Based Solution is highly modular, scalable and configurable giving businesses and educational institutions the benefits of increased efficiencies, faster go to market, predictability of technology as well as spend and better business results.

**HISTORY & RECENT DEVELOPMENTS**

CMC is committed to the highest standards of corporate governance and social responsibility. Rooted in the Tata group's long history of putting people before profits and policy of leadership with trust, CMC believes in being ethical and fair in its business dealings with all stakeholders -- employees, customers, partners, investors and the community. CMC provides a wide spectrum of unique Information Technology solutions and services to a clientele of premier organisations in the government and private sectors. Since its incorporation in 1975, CMC has an enviable record of successfully building IT solutions for massive and complex infrastructure and market projects CMC's customer orientation and service culture lead to enduring bonds with clients. Its distinct value proposition and service culture, coupled with our track record of successful service delivery, are reflected in long-standing customer relationships with dominant players in key infrastructure, services and government sectors. Its customers include some of the biggest organisations in India: Reserve Bank of India, Indian Railways, Indian Oil Corporation Limited, Bharat Petroleum Corporation Limited, Oil and Natural Gas Corporation Limited, United Western Bank, Bank of India and Bank of Baroda. Three of the many major projects undertaken by CMC Limited A passenger ticketing and reservations system for Indian Railways, which runs 6,000 passenger trains carrying over 10 million passengers a day, on a 90,000-km railway network covering around 8,000 railway stations. A cargo handling system is a comprehensive online real time cargo handling system to integrate all complex and varied activities of container terminals. This system has been implemented for several Indian and International ports. An online transaction processing system for the Bombay Stock Exchange, which handles millions of securities trading transactions every day.

**2. LIST OF TABLES**

Table1. Domestic Flights

Table2. Domestic Fares

Table3. International Flights

Table4. International Fares

Table5. Login

Table6. User Details

**3. LIST OF FIGURES**

Fig.1 JDBC Connection

Fig.2 Flow Diagram of JDBC Working

Fig.3 Data Flow Diagram

Fig.4 System Flow Diagram of Booking

Fig.5 System Flow Diagram of Cancellation

Fig.6 Welcome Screen

Fig.7 About Us Screen

Fig.8 Travel Details

Fig.9 Booking Details

Fig.10 Available Flights & Other Details

Fig.11 Login Screen

Fig.12 Ticket Details

Fig.13 Thanks Screen

Fig.14 Cancellation Screen

**4. INTRODUCTION**

**4.1. ABOUT JAVA**

**4.1.1. Creation of Java**

Java was conceived by James Gosling, Patrick Naughton, Chris Warth, Ed Frank, and Mike Sheridan at Sun Microsystems, Inc. in 1991. It took 18 months to develop the first working version. This language was initially called “Oak,” but was renamed “Java” in 1995. The primary motivation was the need for a platform-independent (that is, architecture-neutral) language that could be used to create software to be embedded in various consumer electronic devices, such as microwave ovens and remote controls. Although Java has become inseparably linked with the online environment of the Internet, it is important to remember that Java is first and foremost a programming language. Computer language innovation and development occurs for two fundamental reasons:

• To adapt to changing environments and uses

• To implement refinements and improvements in the art of programming

**4.1.2. Java’s Lineage**

Java is related to C++, which is a direct descendant of C. Much of the character of Java is inherited from these two languages. From C, Java derives its syntax. Many of Java’s object oriented features were influenced by C++. In fact, several of Java’s defining characteristics come from—or are responses to—its predecessors. Moreover, the creation of Java was deeply rooted in the process of refinement and adaptation that has been occurring in computer programming languages for the past several decades.

**4.1.3. Features of Java**

* **Security:** Java achieved protection by confining an applet to the Java execution environment and not allowing it access to other parts of computer.
* **Portability:** Java follows “Write once, run anywhere”. Java codes are portable and same code can be run on different machines.
* **Byte Code:** The key that allows Java to solve both the security and the portability problems just described is that the output of a Java compiler is not executable code. Rather, it is bytecode. Bytecodeis a highly optimized set of instructions designed to be executed by the Java run-time system, which is called the Java Virtual Machine *(JVM)*. In essence, the original JVM was designed as an interpreter for bytecode. This may come as a bit of a surprise since many modern languages are designed to be compiled into executable code because of performance concerns. However, the fact that a Java program is executed by the JVM helps solve the major problems associated with web-based programs.
* **Simple:** Java is simple to learn and use effectively.
* **Object Oriented:** Java is an object oriented language and supports its basic concepts of encapsulation, polymorphism and inheritance.
* **Robust:** Java is robust in the sense that it helps to detect errors at early stages of program development. The multi-platformed environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. Thus, the ability to create robust programs was given a high priority in the design of Java.
* **Multithreaded:** Java was created to meet the real world requirements. And so it supports multithreaded programming, which allows you to write programs that can do many things simultaneously.
* **Architecture-neutral:** With evolution, operating systems upgrade, processor upgrades, and changes in core system resources can all combine to make a program malfunction. Java designers made various decisions in an attempt to alter this situation.
* **Distributed:** Java is designed to for the distributed environment of the Internet because it handles TCP/IP protocols.

* 1. **SWING**

Very simple user interfaces can be built with the AWT classes. Although the AWT is still a crucial part of Java, its component set is no longer widely used to create graphical user interfaces. Today, most programmers use Swing or JavaFX for this purpose. Swing is a framework that provides more powerful and flexible GUI components than does the AWT. As a result, it is the GUI that has been widely used by Java programmers for more than a decade.

**4.2.1. The Origins of Swing**

Swing did not exist in the early days of Java. Rather, it was a response to deficiencies present in Java’s original GUI subsystem: the Abstract Window Toolkit. The AWT defines a basic set of controls, windows, and dialog boxes that support a usable, but limited graphical interface. One reason for the limited nature of the AWT is that it translates its various visual components into their corresponding, platform-specific equivalents, or peers. This means that the look and feel of a component is defined by the platform, not by Java. The use of native peers led to several problems. First, because of variations between operating systems, a component might look, or even act, differently on different platforms. This potential variability threatened the overarching philosophy of Java: write once, run anywhere. Second, the look and feel of each component was fixed (because it is defined by the platform) and could not be (easily) changed. Third, the use of heavyweight components caused some frustrating restrictions. For example, a heavyweight component was always opaque. Not long after Java’s original release, it became apparent that the limitations and restrictions present in the AWT were sufficiently serious that a better approach was needed. The solution was Swing. Introduced in 1997, Swing was included as part of the Java Foundation Classes (JFC). Swing was initially available for use with Java 1.1 as a separate library. However, beginning with Java 1.2, Swing (and the rest of the JFC) was fully integrated into Java.

**4.2.2. Swing Is Built on the AWT**

Before moving on, it is necessary to make one important point: although Swing eliminates a number of the limitations inherent in the AWT, Swing *does not* replace it. Instead, Swing is built on the foundation of the AWT. This is why the AWT is still a crucial part of Java. Swing also uses the same event handling mechanism as the AWT. Therefore, a basic understanding of the AWT and of event handling is required to use Swing.

**4.2.3. Two Key Features of Swing**

1. Swing Components Are Lightweight: With very few exceptions, Swing components are lightweight. This means that they are written entirely in Java and do not map directly to platform-specific peers. Thus, lightweight components are more efficient and more flexible.

1. Swing supports a pluggable look and feel: Swing supports a pluggable look and feel(PLAF). Because each Swing component is rendered by Java code rather than by native peers, the look and feel of a component is under the control of Swing. This fact means that it is possible to separate the look and feel of a component from the logic of the component, and this is what Swing does. Separating out the look and feel provides a significant advantage: it becomes possible to change the way that a component is rendered without affecting any of its other aspects.

**4.2.4. The MVC Connection**

In general, a visual component is a composite of three distinct aspects:

* The way components look when rendered to the screen.
* The way the user reacts to the user.
* The state information associated with the component.

No matter what architecture is used to implement a component, it must implicitly contain these three parts. Over the years, one component architecture has proven itself to be exceptionally effective: Model- View- Controller or MVC for short. The MVC architecture is successful because each piece of the design corresponds to an aspect of a component. In MVC terminology, the *model* corresponds to the state information associated with the component. For example, in the case of a check box, the model contains a field that indicates if the box is checked or unchecked. The *view* determines how the

Component is displayed on the screen, including any aspects of the view that are affected by the current state of the model. The *controller* determines how the component reacts to the user. For example, when the user clicks a check box, the controller reacts by changing the model to reflect the user’s choice (checked or unchecked). This then results in the view being updated. By separating a component into a model, a view, and a controller, the specific implementation of each can be changed without affecting the other two.

**4.2.5. Components**

In general, Swing components are derived from JComponent class.JComponentprovides the functionality that is common to all components. All of Swing’s components are represented by classes defined within the package **javax.swing**. The following table shows the class names for Swing components (including those used as containers).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| JApplet | JButton | JCheckBox | JCheckBoxMenuItem | JColorChooser |
| JComboBox | JComponent | JDesktopPane | JDialog | JEditorPane |
| JFileChooser | JFormattedTextField | JFrame | JInternalFrame | JLabel |
| JLayer | JLayeredPane | JList | JMenu | JMenuBar |
| JMenuItem | JOptionPane | JPanel | JPasswordField | JPopupMenu |
| JProgressBar | JRadioButton | JRootPane | JScrollBar | JScrollPane |
| JSeparator | JSlider | JSpinner | JSplitPane | JTabbedPane |
| JTable | JTextArea | JTextField | JTextPane | JToggleButton |
| JToolBar | JToolTip | JTree | JViewPort | JWindow |

The components used in the project along with their description are as follows:

* JLabel: JLabelis Swing’s easiest-to-use component.JLabelcan be used to display text and/or an icon. It is a passive component in that it does not respond to user input.
* JTextField: JTextFieldis the simplest Swing text component. It is also probably its most widely used text component. JTextField allows you to edit one line of text.
* JButton: The JButton class provides the functionality of a push button. It allows an icon, a string, or both to be associated with the push button. When the button is pressed, an ActionEvent is generated. Using the ActionEvent object passed to the actionPerformed( ) method of the registered ActionListener, you can obtain the *action command* string associated with the button.
* JCheckBox: JCheckBox class provides the functionality of a check box.
* JComboBox: Swing provides a *combo box* (a combination of a text field and a drop-down list) through the JComboBox class. A combo box normally displays one entry, but it will also display a dropdown list that allows a user to select a different entry.
* JTable: JTable is a component that displays rows and columns of data. You can drag the cursor on column boundaries to resize columns. You can also drag a column to a new position. Depending on its configuration, it is also possible to select a row, column, or cell within the table, and to change the data within a cell.
  1. **JDBC**

Java JDBC is a java API to connect and execute query with the database. JDBC API uses jdbc drivers to connect with the database.

JDBC (Java Database Connectivity) 

Fig.1 JDBC Connection

There are 5 steps to connect any java application with the database in java using JDBC. They are as follows:

* Register the driver class - The forName () method of Class ‘class’ is used to register the driver class. This method is used to dynamically load the driver class.
* Creating connection -The getConnection () method of DriverManager class is used to establish connection with the database.
* Creating statement- The createStatement () method of Connection interface is used to create statement. The object of statement is responsible to execute queries with the database.
* Executing queries - The executeQuery () method of Statement interface is used to execute queries to the database. This method returns the object of ResultSet that can be used to get all the records of a table.
* Closing connection - By closing connection object statement and ResultSet will be closed automatically. The close() method of Connection interface is used to close the connection.
  + 1. **Connectivity to Oracle:**

1. DriverManager Class - The DriverManager class acts as an interface between user and drivers. It keeps track of the drivers that are available and handles establishing a connection between a database and the appropriate driver. The DriverManager class maintains a list of Driver classes that have registered themselves by calling the method DriverManager.registerDriver ().
2. Connection - A Connection is the session between java application and database. The Connection interface is a factory of Statement, PreparedStatement, and DatabaseMetaData i.e. object of Connection can be used to get the object of Statement and DatabaseMetaData. The Connection interface provide many methods for transaction management like commit (), rollback () etc.
3. Statement - The Statement interface provides methods to execute queries with the database. The statement interface is a factory of ResultSet i.e. it provides factory method to get the object of ResultSet.
4. ResultSet- The object of ResultSet maintains a cursor pointing to a row of a table. Initially, cursor points to before the first row.
5. PreparedStatement- The PreparedStatement interface is a sub interface of Statement. It is used to execute parameterized query.  The performance of the application will be faster if you use PreparedStatement interface because query is compiled only once.

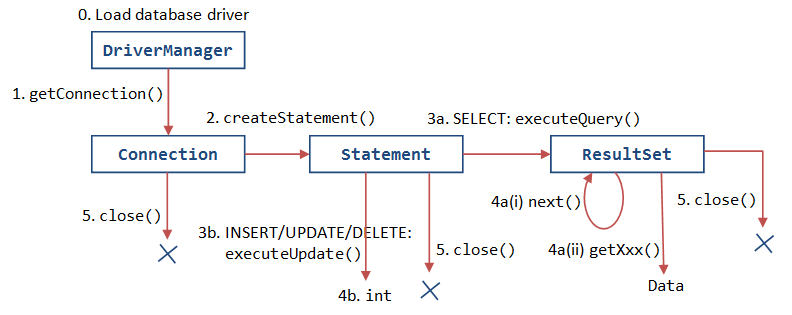


Fig. 2 Flow Diagram of JDBC Working

1. **SOFTWARE DESCRIPTION**

**5.1 NETBEANS IDE 8.2**

NetBeans is a software development platform written in Java. The NetBeans Platform allows applications to be developed from a set of modular software components called *modules*. Applications based on the NetBeans Platform, including the NetBeans integrated development environment (IDE), can be extended by third party developers. The NetBeans IDE is primarily intended for development in Java, but also supports other languages, in particular PHP, C/C++ and HTML5.NetBeans is cross-platform and runs on Microsoft Windows, Mac OS X, Linux, Solaris and other platforms supporting a compatible JVM. Fast & smart code editing. The editor supports many languages from Java, C/C++, XML and HTML, to PHP, Groovy, Javadoc, JavaScript and JSP. Because the editor is extensible, you can plug in support for many other languages.

An IDE is much more than a text editor. The NetBeans Editor indents lines, matches words and brackets, and highlights source code syntactically and semantically. It lets you easily refactor code, with a range of handy and powerful tools, while it also provides code templates, coding tips, and code generators. Keeping a clear overview of large applications, with thousands of folders and files, and millions of lines of code, is a daunting task. NetBeans IDE provides different views of your data, from multiple project windows to helpful tools for setting up your applications and managing them efficiently, letting you drill down into your data quickly and easily, while giving you versioning tools via Subversion, Mercurial, and Git integration out of the box.

A new version was released 8.2 October 3, 2016. NetBeans IDE is the official IDE for Java 8. With its editors, code analysers, and converters, you can quickly and smoothly upgrade your applications to use new Java 8 language constructs, such as lambdas, functional operations, and method references. NetBeans IDE 8.2 provides out-of-the-box code analysers and editors for working with the latest Java 8 technologies--Java SE 8, Java SE Embedded 8, and Java ME Embedded 8.

**5.2. ORACLE 10g EXPRESS EDITION**

Oracle Database 10g Express Edition (Oracle Database XE) is a free version of the world's most capable relational database. Oracle Database XE is easy to install, easy to manage, and easy to develop with.

With Oracle Database XE, one can use an intuitive, browser-based interface, to:

•Administer the database

•Create tables, views, and other database objects

•Import, export, and view table data

•Run queries and SQL scripts

•Generate reports

1. **PROJECT DESCRIPTION**
   1. **OVERVIEW**

The Air Ticket Booking 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, under two tabs namely “From” and “To” for a particular departure 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 create an account and enter his details such as username, credit-card number and contact number and create a password also. Then it checks the validity of card and book the flight and update the airline database and user database. The system also allows the customer to cancel his/her reservation, if any problem occurs. The main purpose of this software is to reduce the manual errors involved in the airline reservation process and make it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations, modify reservations or cancel a particular reservation.

* 1. **MODULE DESCRIPTION**

The Air Ticket Booking System has two main modules:

1. Checking for available flights along with their schedule and fares, and subsequently booking of the flight.
2. Cancellation of flight.

The user can browse through the flights available and can also book a flight using the system, after which a confirmation ticket will be issued to the user. In order to confirm a booking a user has to login to his id or if he/she is a new user they can also sign up and create a new account. The user is also provided the facility to cancel a flight by logging into their account and entering the flight number.

**6.3 DATA FLOW DIAGRAM**

Data flow diagrams focus on the data flowing into and out of the system and the processing of the data. These basic components of every computer program can be described in detail and used to analyse the system for accuracy and completeness.

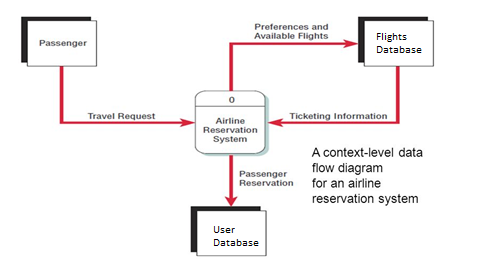


Fig.3 Data Flow Diagram

The most basic elements of an airline reservation system are represented in this data flow diagram. The passenger (an entity) initiates a travel request (data flow). The passenger’s preferences and the available flights are sent to the flights database, which sends the details of the available flights back to the process. After the user confirms the booking, the data is updated in the user details and flights’ databases and a ticket is issued to the user and he/she can take a printout of the ticket as well.

**6.4 SYSTEM FLOW DIAGRAM**

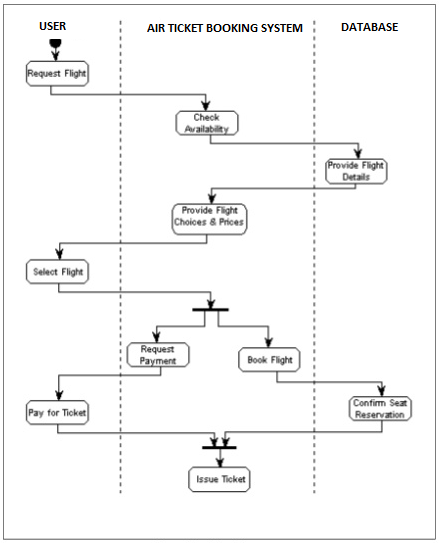


Fig. 4 System Flow Diagram of Booking Process

The system flow diagram describes the flow of the booking process. The user requests a flight by choosing two cities from the two tabs of “from” and “to”, selecting the date of travel and entering the number of passengers and selecting the class of travel. Then the system checks for the availability of the flight and provides the details of the flight like flight number, airlines, airplane type, time of departure and time of arrival along with the fare details. Then the user can opt to choose a flight and has to login using the username and password. If he/she is a new user then they can sign up to create a new account. After confirmation by user the flight is booked and the database is updated. A ticket is also generated in the end for the user to take its printout.

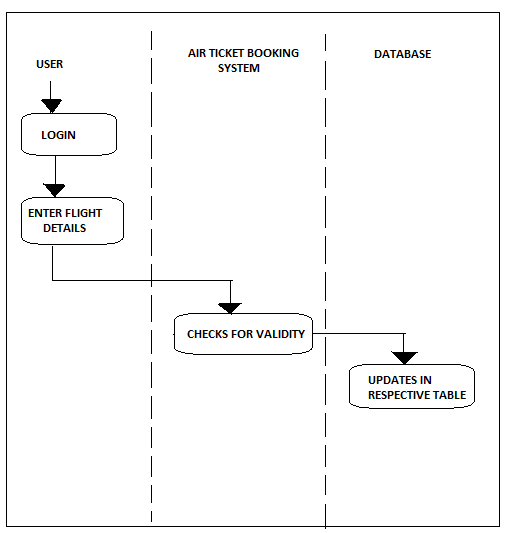


Fig.5 System Flow Diagram of Cancellation

This is a system flow diagram of Cancellation process. The user has to login into the account by using username and password. And has to fill up the details like flight number, date of travel and class of travel. Then the system checks the validity of the details entered and makes necessary updates in the database of the flight as well as the user’s.

**6.5 SYSTEM DESIGN**

The Graphics User Interface (GUI) has been created using JFrame. The components used actively throughout the program are:

* JButton
* JTextField
* JLabel
* JTable
* JPasswordField
* JPanel
* DateChooserCombo
* JComboBox
* JSeparator
* JCheckBox

**6.5.1. SCREENSHOTS OF THE INTERFACE**



Fig. 6 Welcome Screen

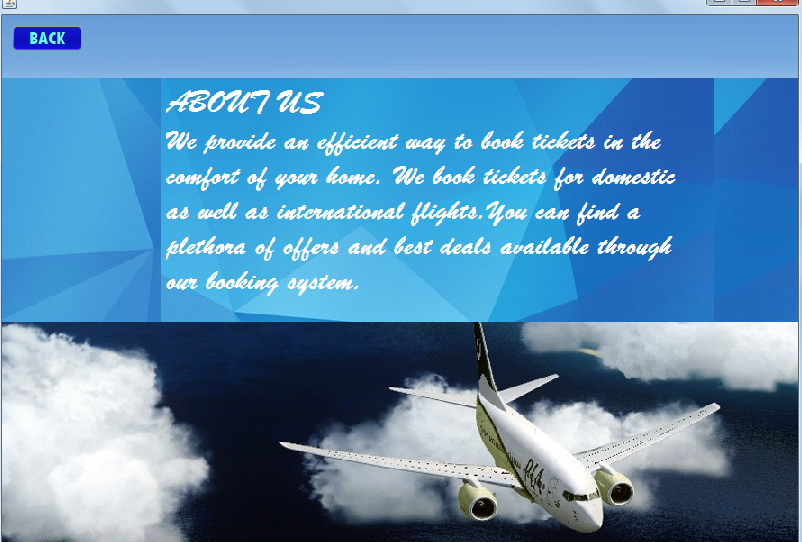


Fig. 7 About Us Screen



Fig. 8 Travel Details

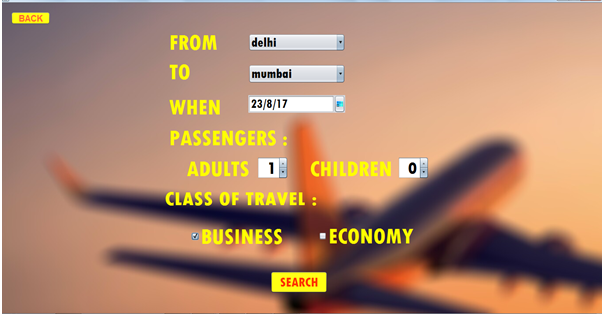


Fig. 9 Booking Details

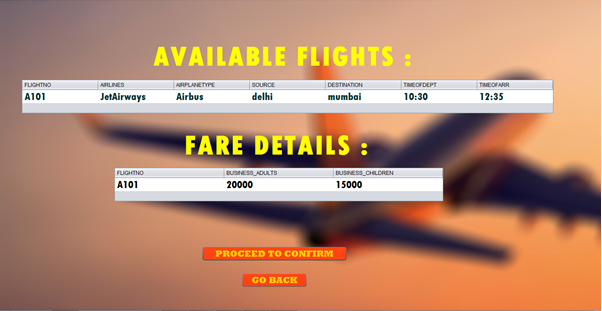


Fig. 10 Available Flights and Other Details

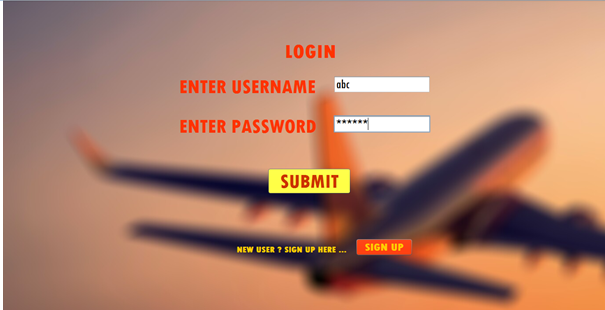


Fig. 11 Login Screen

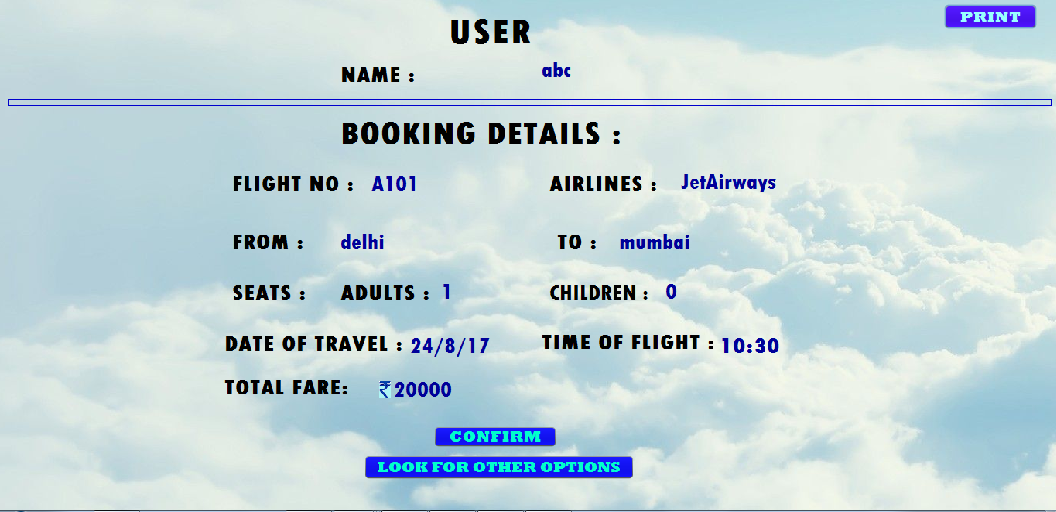


Fig. 12 Ticket Details

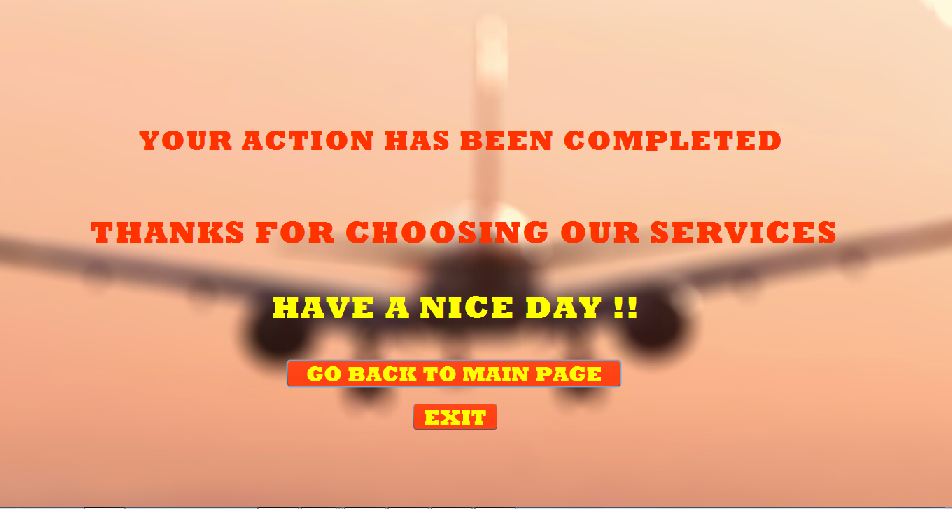


Fig.13 Thanks Screen

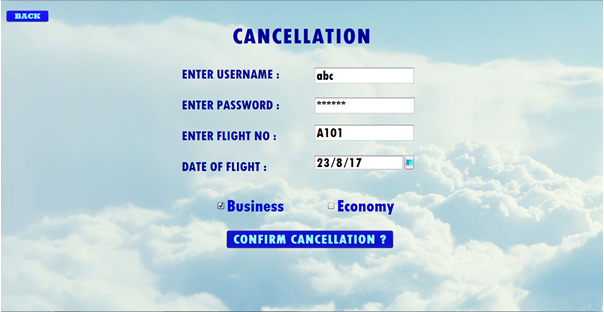


Fig. 14 Cancellation Screen

Each of the frames represents a java class which extends javax.swing.JFrame.

**6.6 DATABASE DESIGN**

Database has been created using Oracle 10g Express Edition. There are six different tables being used in the entire program namely: Domestic, Domestic Fares, International, International Fares, Login and User Details.

**6.6.1 Tables**

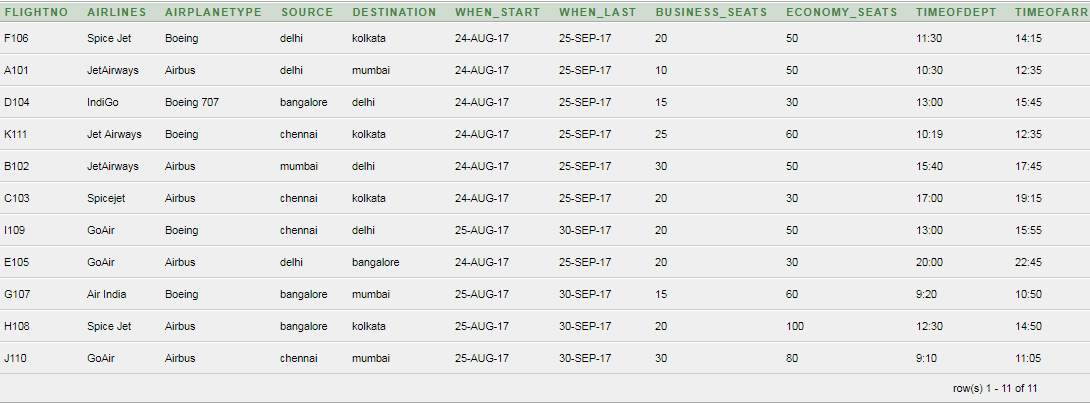


Table1. Domestic Flights

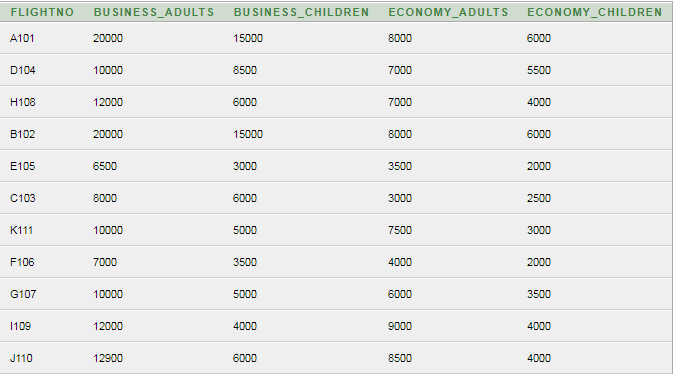


Table2. Domestic Fares



Table3. International Flights

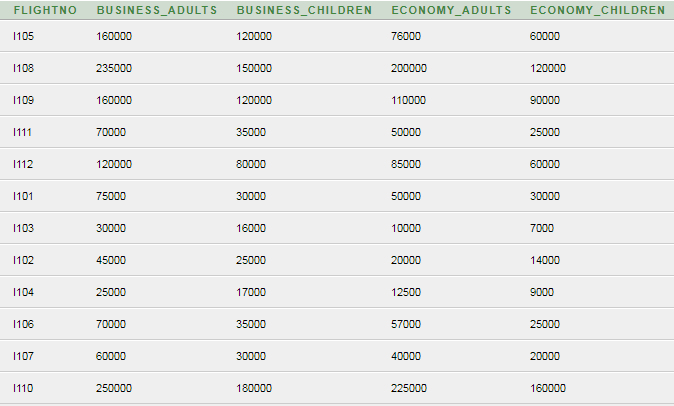


Table4. International Fares

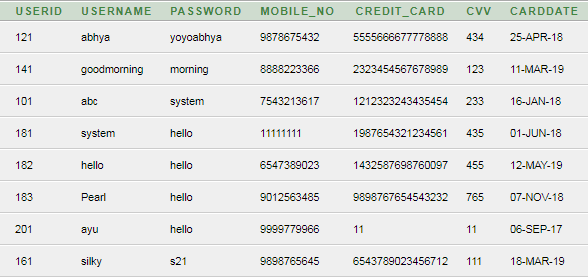


Table5. Login

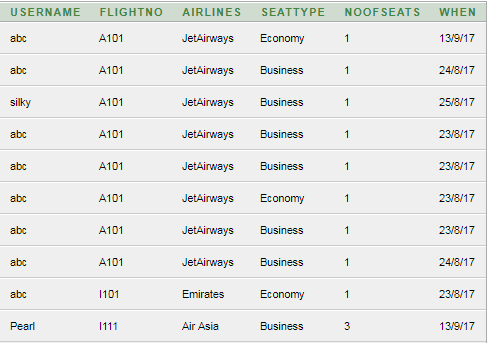


Table6. User Details

All the tables store the entire data being used in the program. Table 1 and Table 3 store the data of domestic and international flights respectively which includes, flight number, airlines, airplane type, source and destination cities, dates and departure and arrival time. Table 2 and Table 4 hold the data about fare details of domestic and international flights respectively. These tables include fares for adults and children for both business and economy classes. Table 5 holds the Login details of the users along with their credit card details. Table 6 holds the booking details of users.

1. **CONCLUSION**

The Project is working satisfactorily & has been tested several times for different test cases. It is a wholesome project which can be used for commercial purpose. The interfaces are user-friendly and relatable.

From improvement point of view, the payment module interface in the project has been temporarily designed to show a virtual transaction. However, for commercial use the real transactions can be implemented by a payment gateway. The database can also be extended to include all major cities of India and abroad.

1. **BIBLIOGRAPHY**
2. Java: The Complete Reference, Ninth Edition by Herbert Schildt.
3. Websites:

* [www.javatpoint.com/jdbc-driver](http://www.javatpoint.com/jdbc-driver)
* [www.netbeans.org](http://www.netbeans.org)