**Course Code:CSC-232**

**DSA LAB**

**AIRLINE MANAGEMENT SYSTEM**

**Project Report By**

**Syed sufiyan jawaid (25761)**

**Group Members:**

Maaz Tariq Soomro (25607)

Shaikh Muhammad Zeeshan (25616)

**Table of Contents**

[1. Introduction 3](#_Toc112854775)

[2. Scope 3](#_Toc112854776)

[3. Module Description 3](#_Toc112854777)

[4. Hardware/Software Requirements 3](#_Toc112854778)

[5. Screen Shots 3](#_Toc112854779)

# Introduction

The **Airline Management System** is a Java-based desktop application developed as a Data Structures and Algorithms (DSA) project. This system is designed to streamline the management of airline operations, including customer management, flight scheduling, ticket booking, and other critical functionalities.

The application leverages Java Swing for a graphical user interface (GUI) implemented through JFrame forms, making it intuitive and user-friendly. The primary goal is to provide a user-friendly interface for booking flights, managing passenger data, and maintaining flight details. Data structures hashmaps play a role in organizing and managing information effectively. The system efficiently handles various airline management tasks. The project also serves as a learning platform to explore GUI development in Java and the integration of external libraries and databases.

### Libraries

The project employs the following tools and libraries:

* **JDK 16:** To leverage the latest Java features for robust development.
* **MySQL Connector (v8.0.28):** For seamless database connectivity to manage backend data.
* **JCalendar:** For date selection in the GUI, enhancing user convenience.
* **RS2XML:** For displaying database tables in a user-friendly tabular format within the application.

# Scope

The Airline Management System, developed with Java and NetBeans GUI, aims to simplify and automate airline operations while providing a user-friendly interface. The system incorporates the following functionalities:

* **Flight Management:**
  1. Adding, viewing, and managing flight details to ensure accurate and efficient scheduling.
  2. GUIs: addflight.java, flightsinfo.java.
* **Customer Management:**
  1. Maintaining and displaying customer details for personalized service.
  2. GUIs: customer.java, customerinfo.java, viewcustomer.java.
* **Booking and Ticketing:**
  1. Facilitating flight booking and generating boarding passes seamlessly.
  2. GUIs: flightbooking.java, boardingpass.java.
* **Authentication:**
  1. Ensuring secure access through login and signup functionalities.
  2. GUIs: login.java, signup.java.
* **Journey Details:**
  1. Managing and displaying passenger journey details to enhance travel planning.
  2. GUI: journeyDetails.java.
* **Admin Dashboard:**
  1. Providing a centralized interface for administrators to manage airline operations efficiently.
  2. GUI: dashboard.java.
* **Cancellation:**
  1. Enabling hassle-free ticket cancellation for passengers.
  2. GUI: CancelTicket.java

# Module Description

The system is divided into various modules, each handling specific functionalities such as flight management, customer data, booking, cancellations, and administrative controls. Detailed descriptions are provided for authentication, flight management, journey details, and more.

1. **Authentication Module:**
   * Handles user login and signup functionalities.
   * Ensures secure access to the system.
   * Classes: login.java, signup.java.
2. **Flight Management Module:**
   * Manages the addition, update, and retrieval of flight details.
   * Allows administrators to schedule flights.
   * Classes: addflight.java, flightsinfo.java.
3. **Customer Management Module:**
   * Handles customer data storage and retrieval.
   * Displays customer information for administrative use.
   * Classes: customer.java, customerinfo.java, viewcustomer.java.
4. **Booking Module:**
   * Facilitates booking of tickets and generating boarding passes.
   * Classes: flightbooking.java, boardingpass.java.
5. **Journey Details Module:**
   * Maintains and displays detailed information about passenger journeys.
   * Class: journeyDetails.java.
6. **Cancellation Module:**
   * Allows users to cancel booked tickets efficiently.
   * Class: CancelTicket.java.
7. **Dashboard Module:**
   * Acts as a central hub for managing all system operations.
   * Class: dashboard.java.

8. **All Info Module:**

* Displays all passenger and booking details in a centralized view.
* Information includes PNR number, ticket number, name, address, source, destination, and more.
* Class: allinfo.java

9**. Thank You Screen:**

* + Displays a confirmation page after generating boarding pass.
  + Class: thankyou.java.

# Hardware/Software Requirements

#### Hardware Requirements

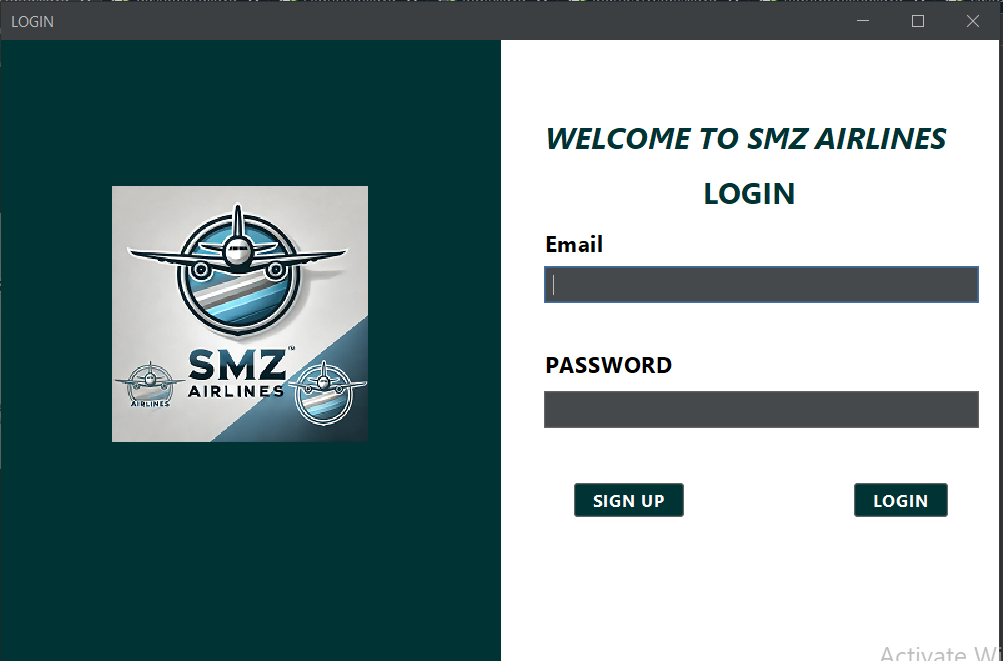
* **Processor:** Intel Core i3 or higher.
* **RAM:** 4 GB (minimum), 8 GB or more
* **Hard Disk:** 500 MB free space (minimum).

#### Software Requirements

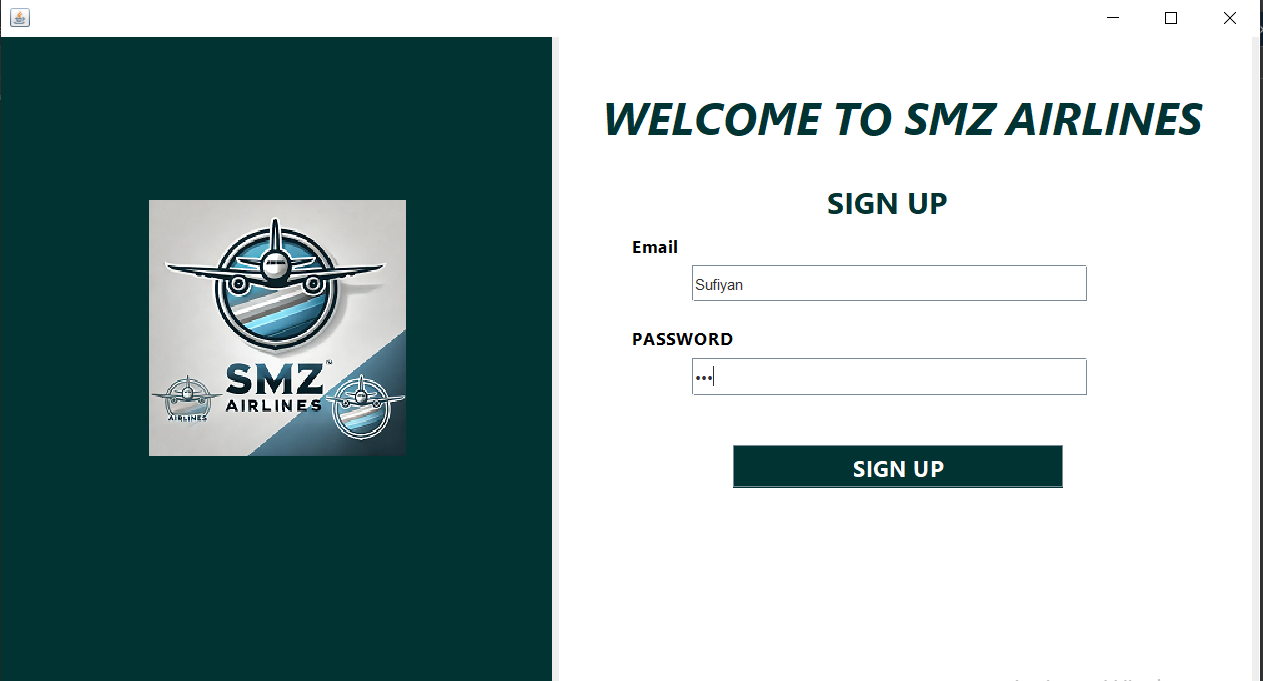
* **Operating System:** Windows 10/11, macOS, or Linux
* **Java Development Kit (JDK):** JDK 16.
* **Database:** MySQL Server 8.0.28 or higher.
* **IDE:** NetBeans or IntelliJ IDEA for development.
* **Libraries:**
  + JCalendar for date selection.
  + RS2XML for rendering tables in the GUI.
  + MySQL Connector for database connectivity.

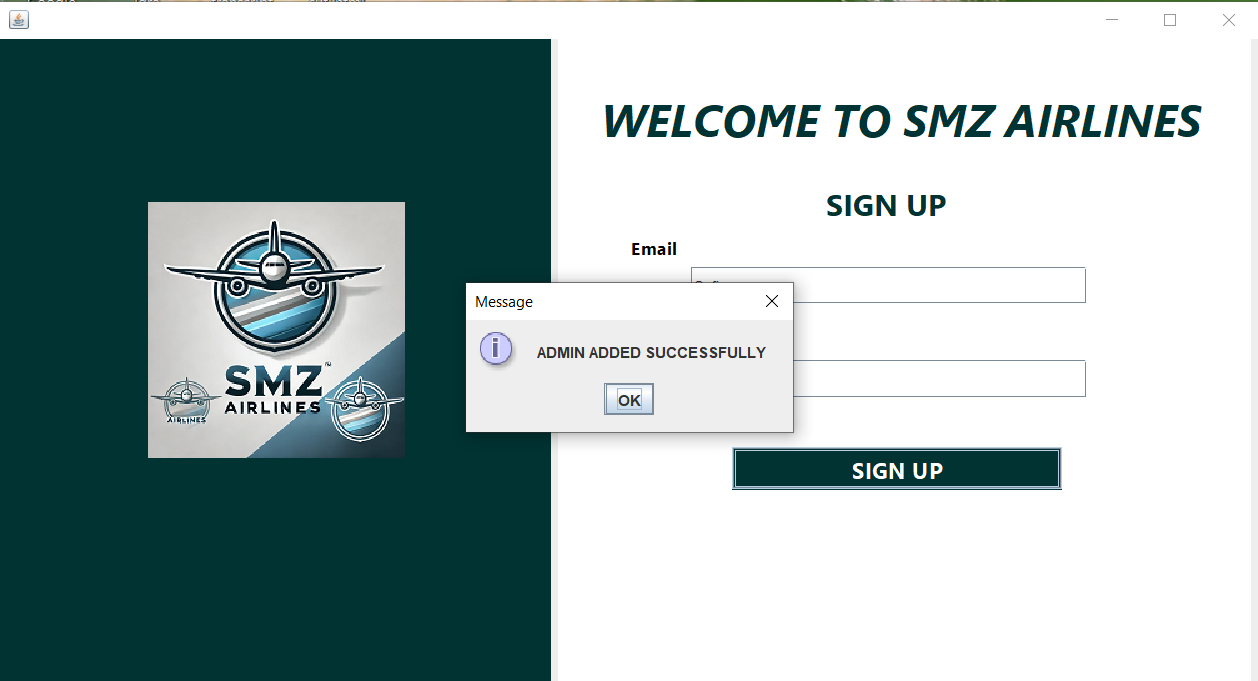
# 5. Screen Shots

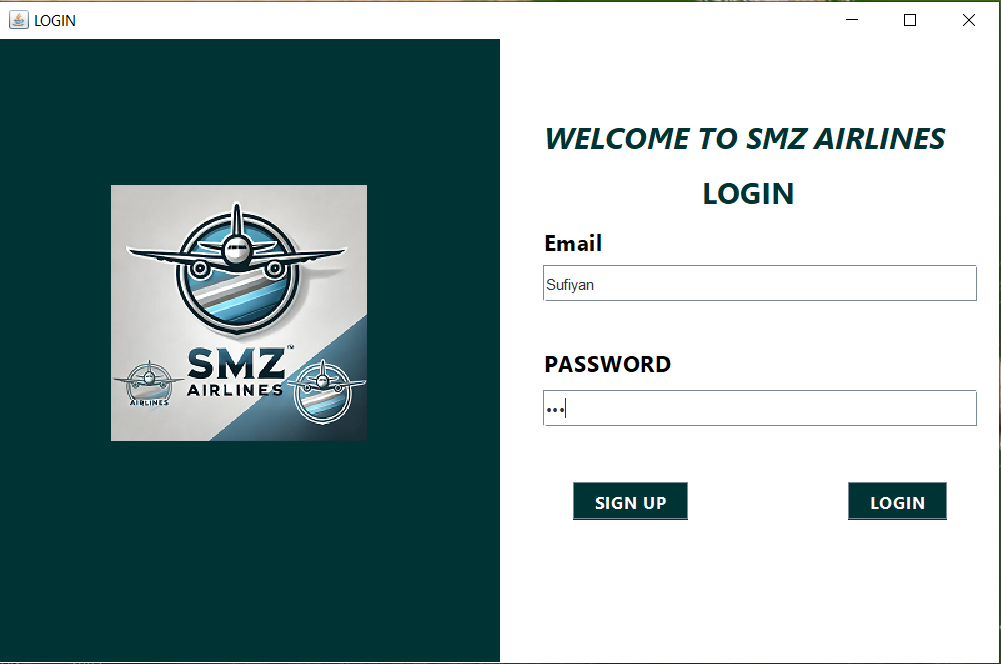
**LOGIN PAGE**



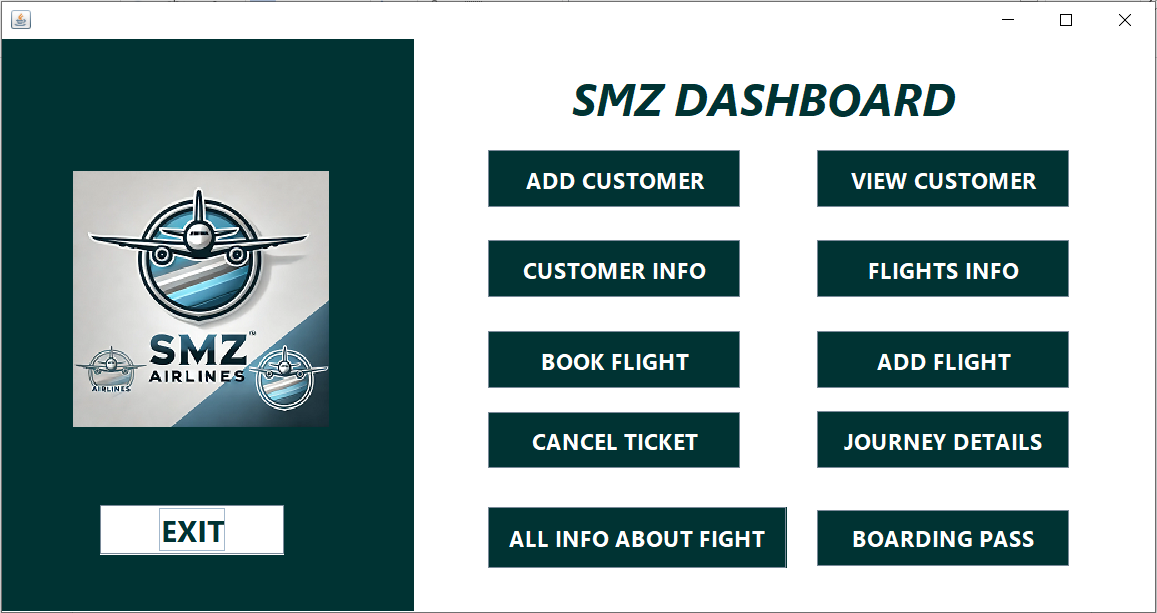
**FIRST I DO (SIGN UP):**

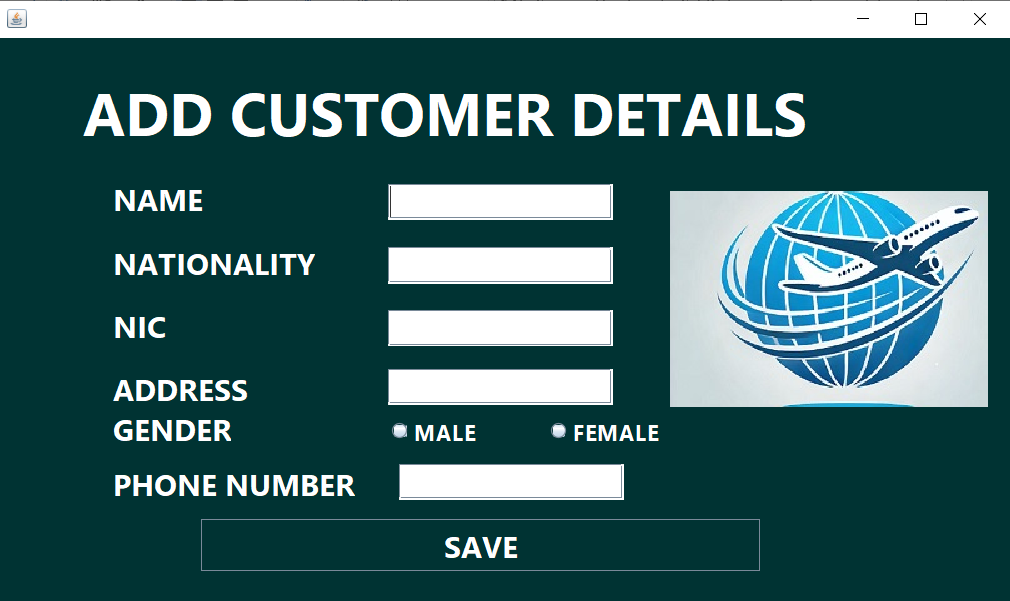


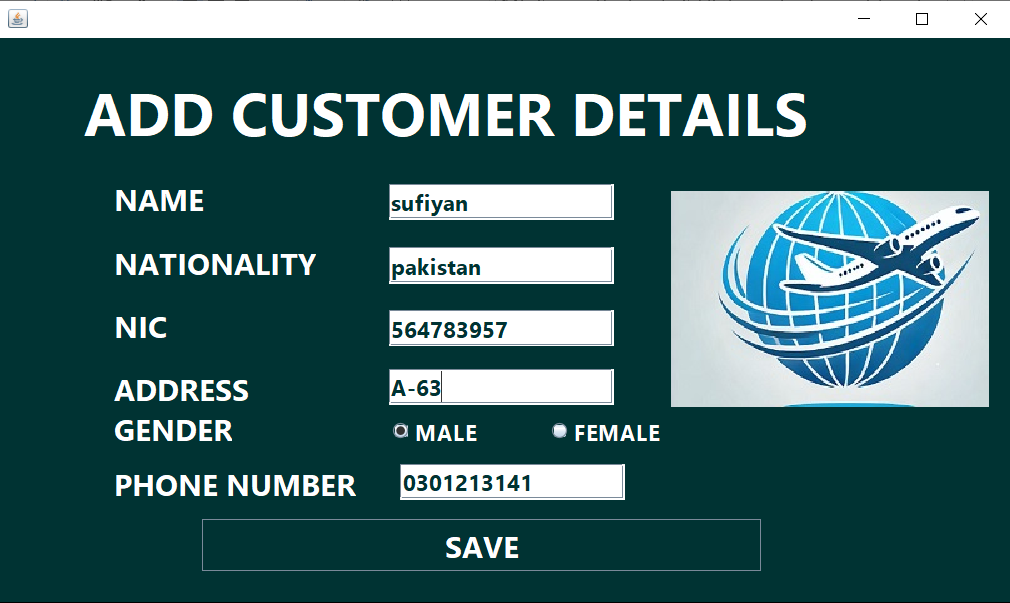
SIGN UP DONE (ADMIN ADDED):

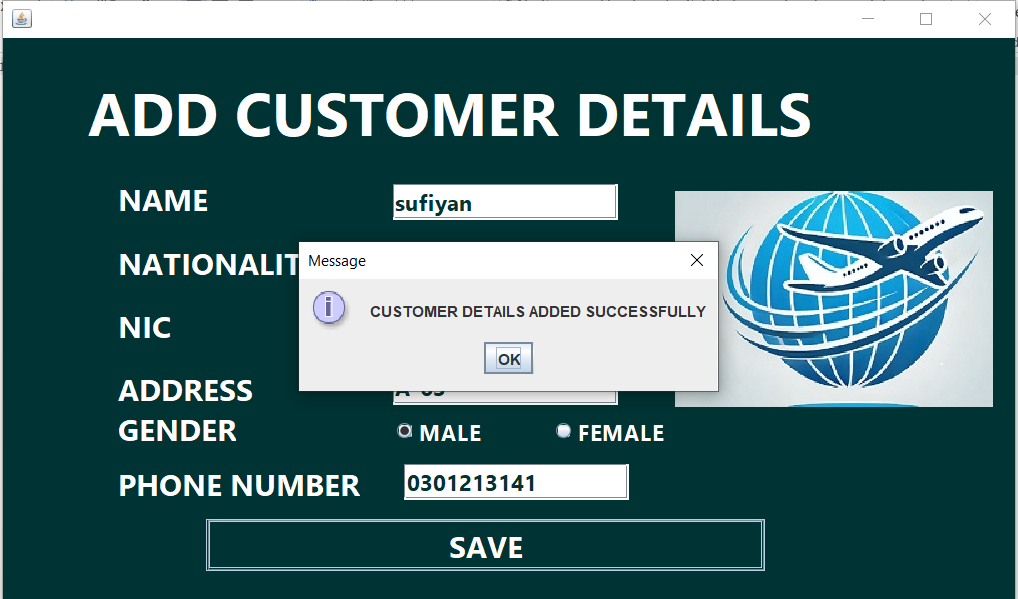
NOW LOGIN (WITH NEWLY ADDED ADMIN):

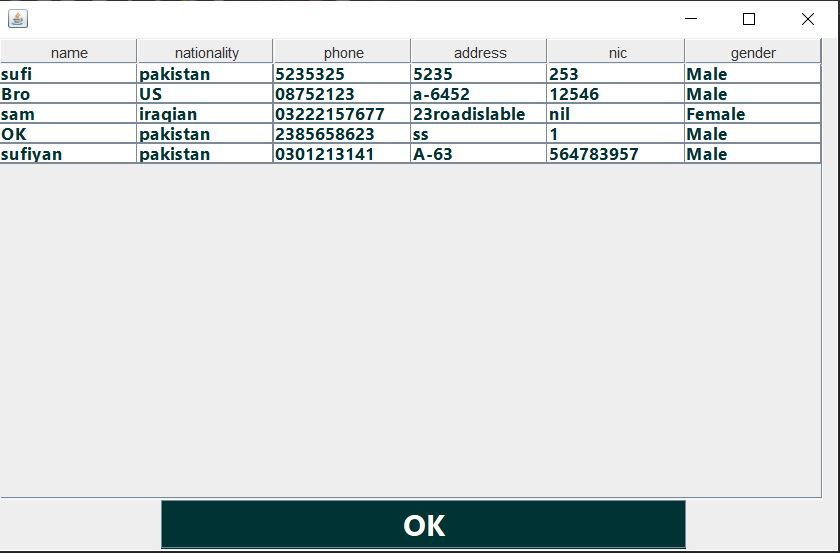
AFTER CLICK LOGIN BUTTON YOU GO ON DASHBOARD:

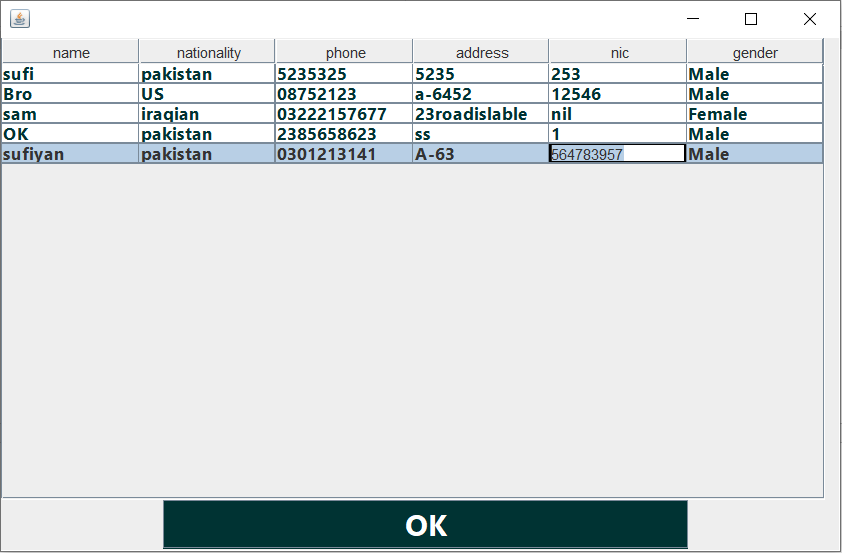


FIRST WE CLICK ADD CUSTOMER BUTTON (FROM DASHBOARD):

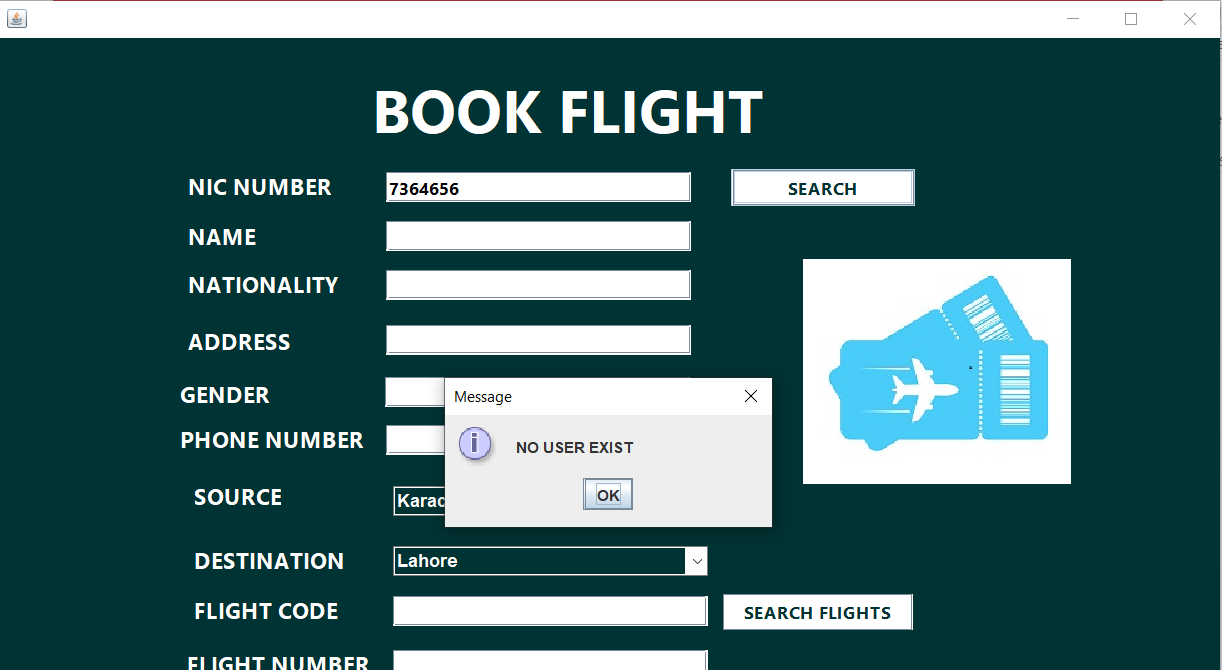
NOW ADD DETAILS OF CUSTOMER:

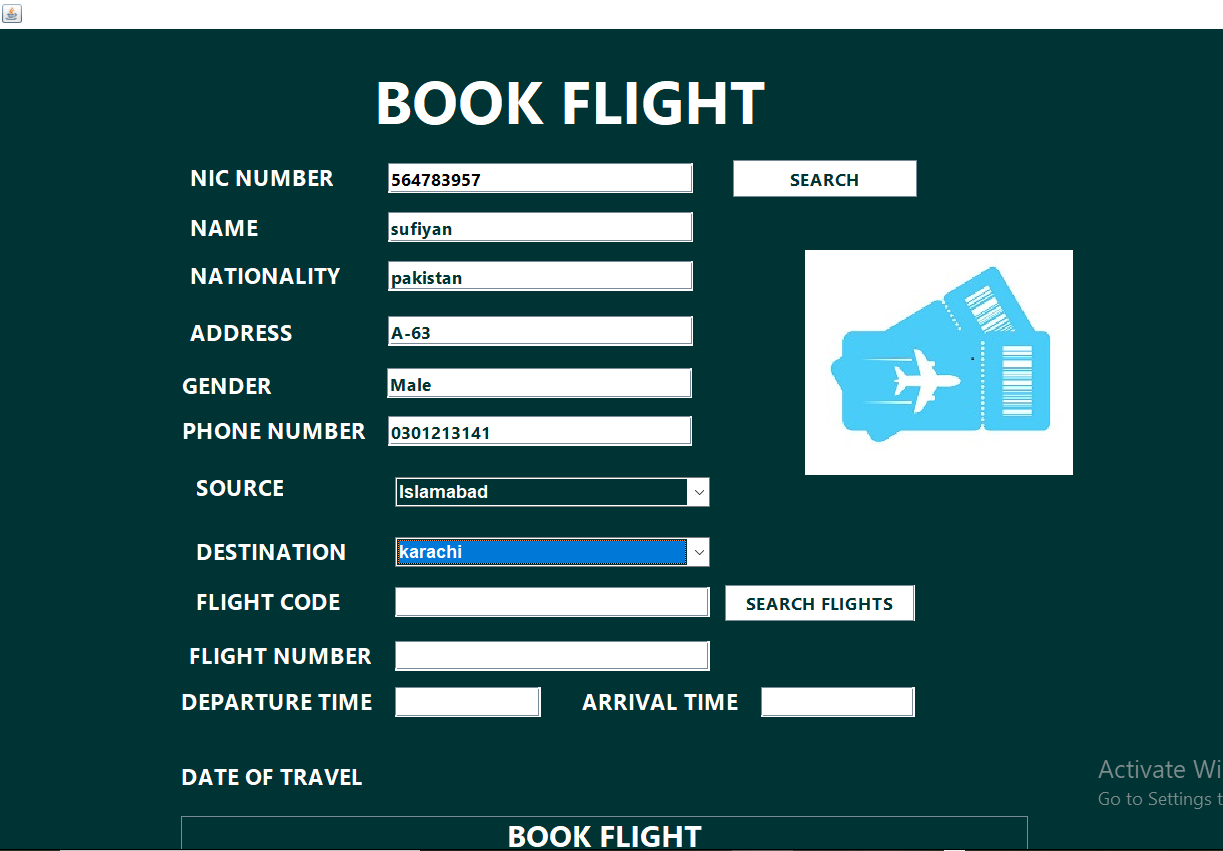
NOW CLICK SAVE TO ADD CUSTOMER:

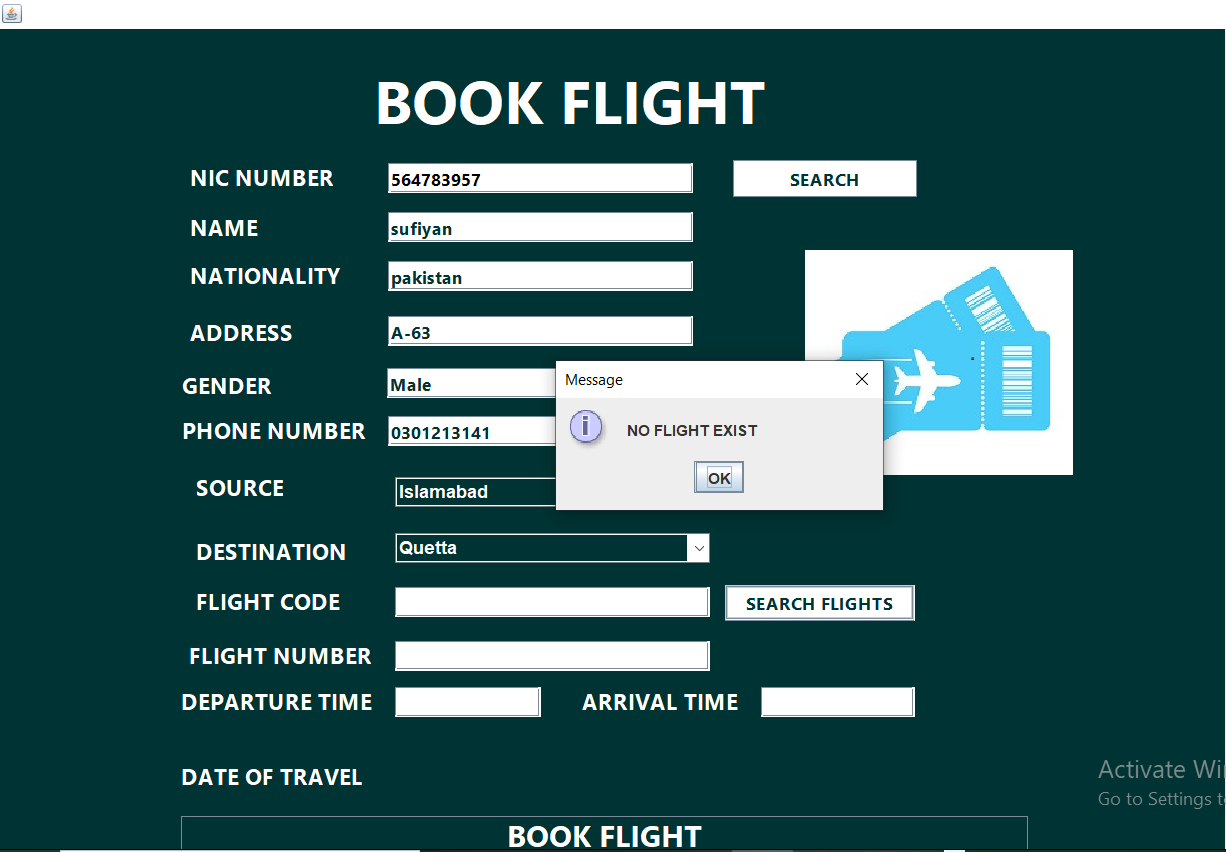
AFTER CLICK OK WE GO TO CUSTOMER INFO TABLE WHERE CUSTOMER IS ADDED:

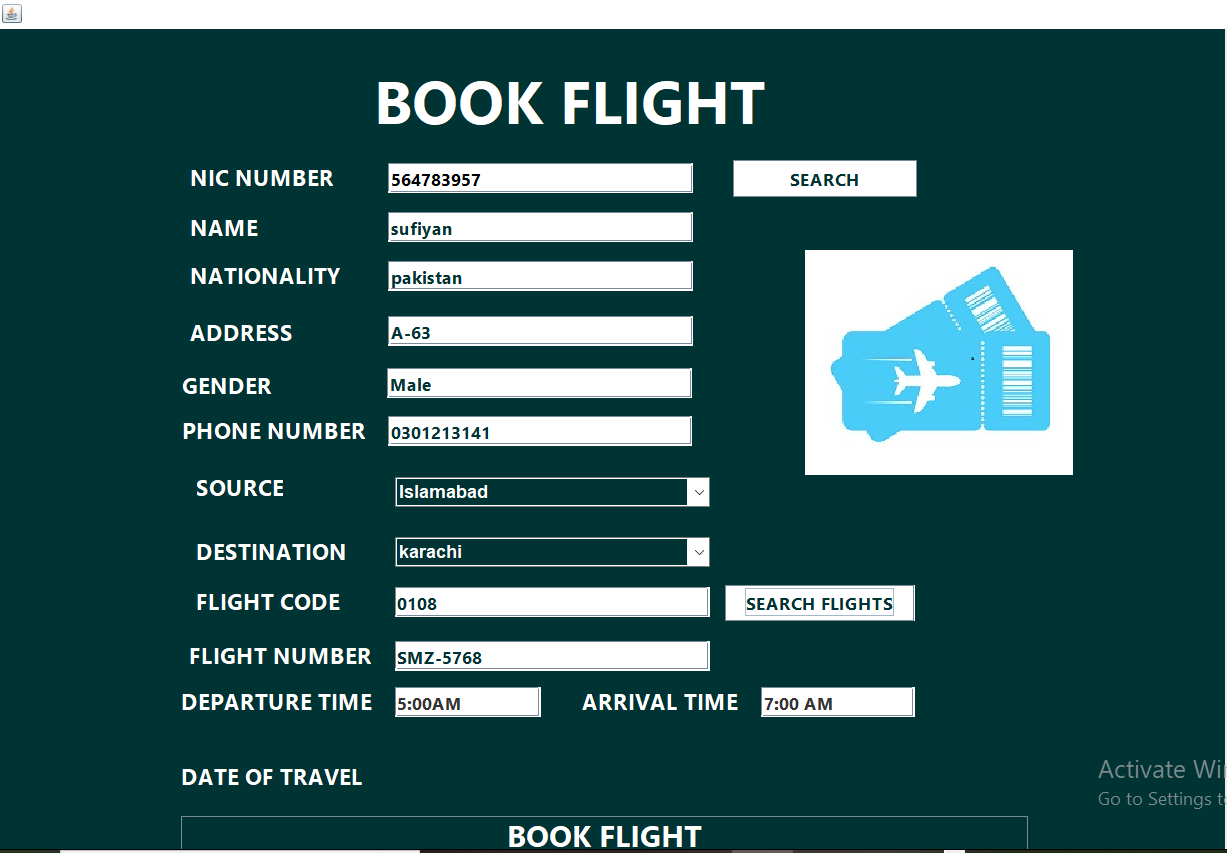
NOW COPY NIC NUMBER FROM THE TABLE AND CLICK OK BUTTON:

AFTER CLICK OK BUTTON YOU GO ON BOOK GIVE NIC NUMBER IF INCORRECT (GIVE MESSAGE NO USER)

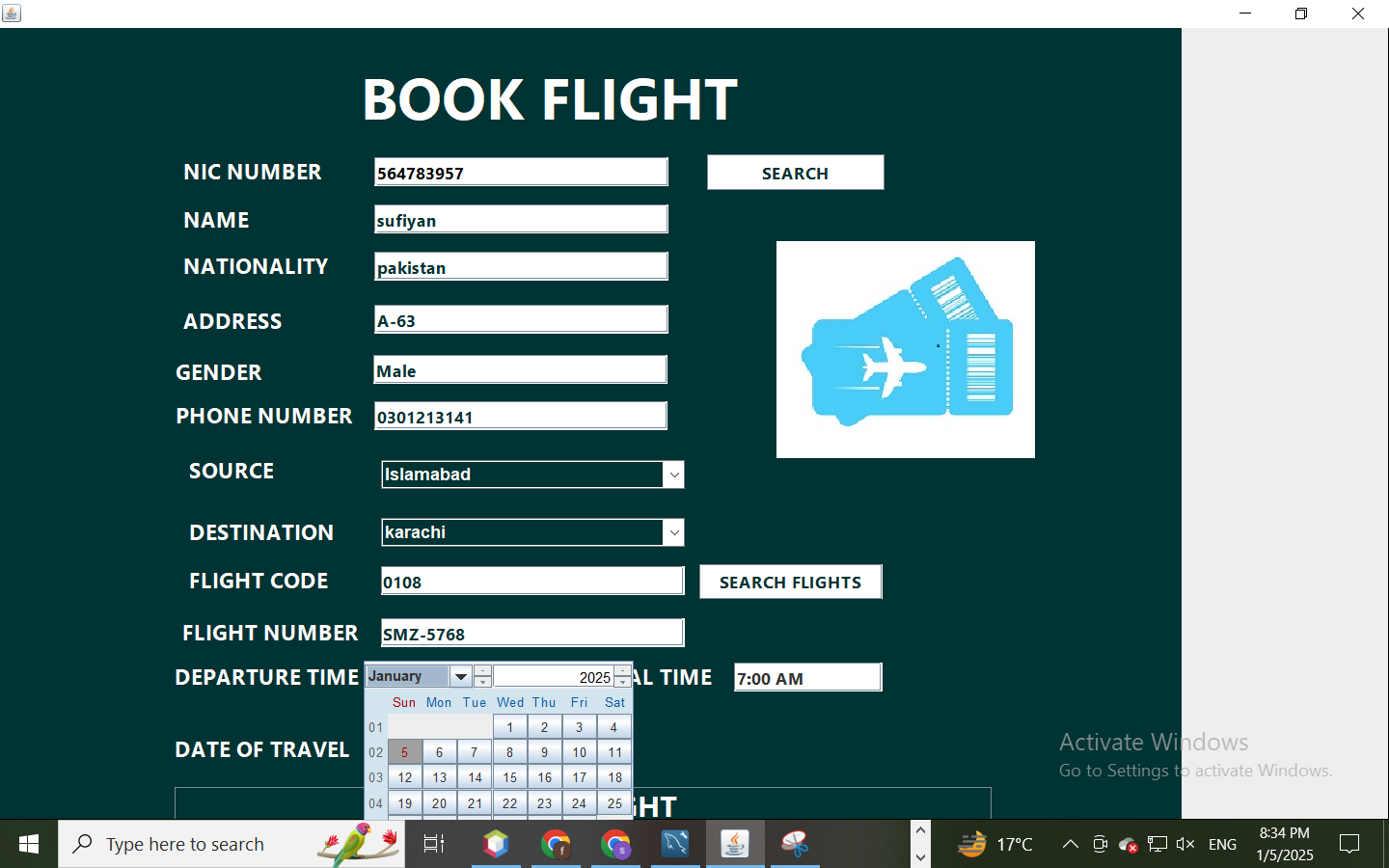


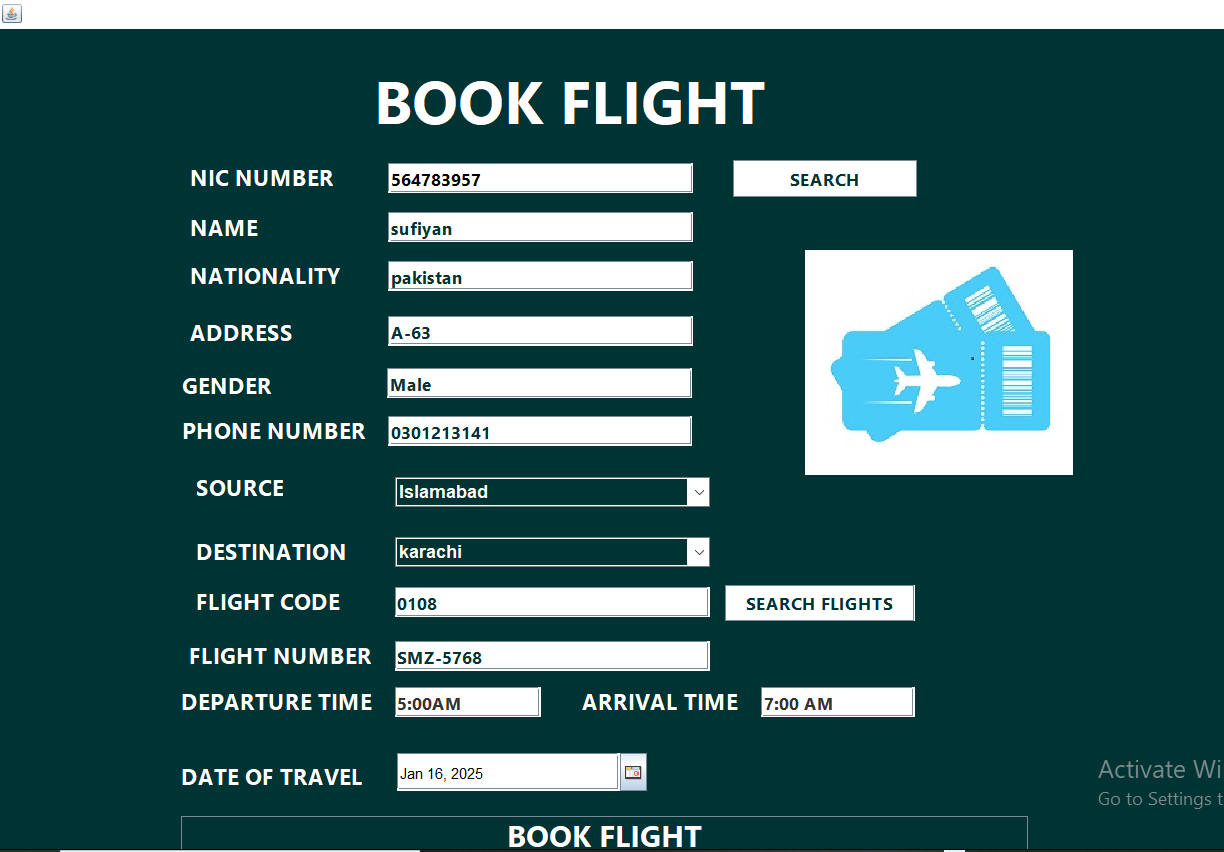
AFTER PASTE CORRECT NIC AND CLICK SEARCH BUTTON (ALL INFO OF CUSTOMER VISIBLE)

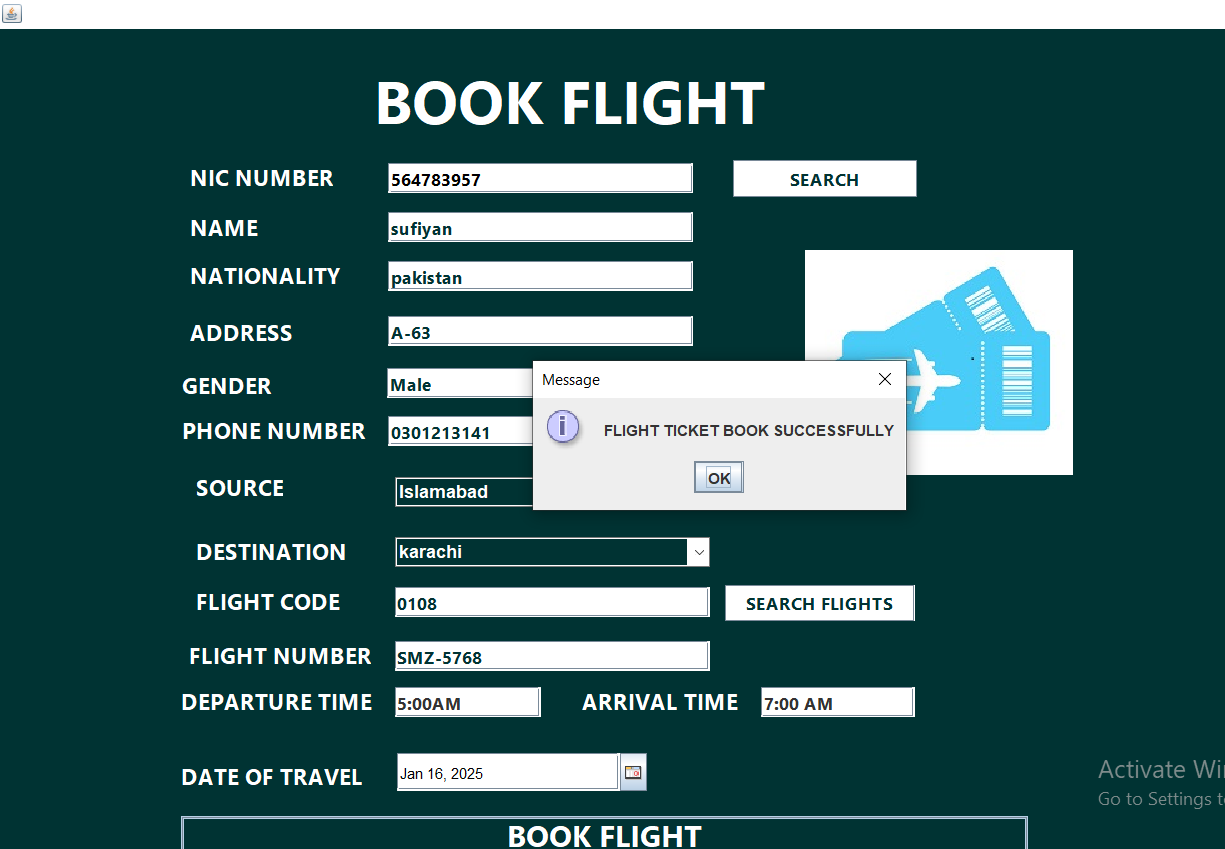
NOW SELECT FLIGHT (IF FIGHT NOT EXIST IT GIVE MESSAGE)

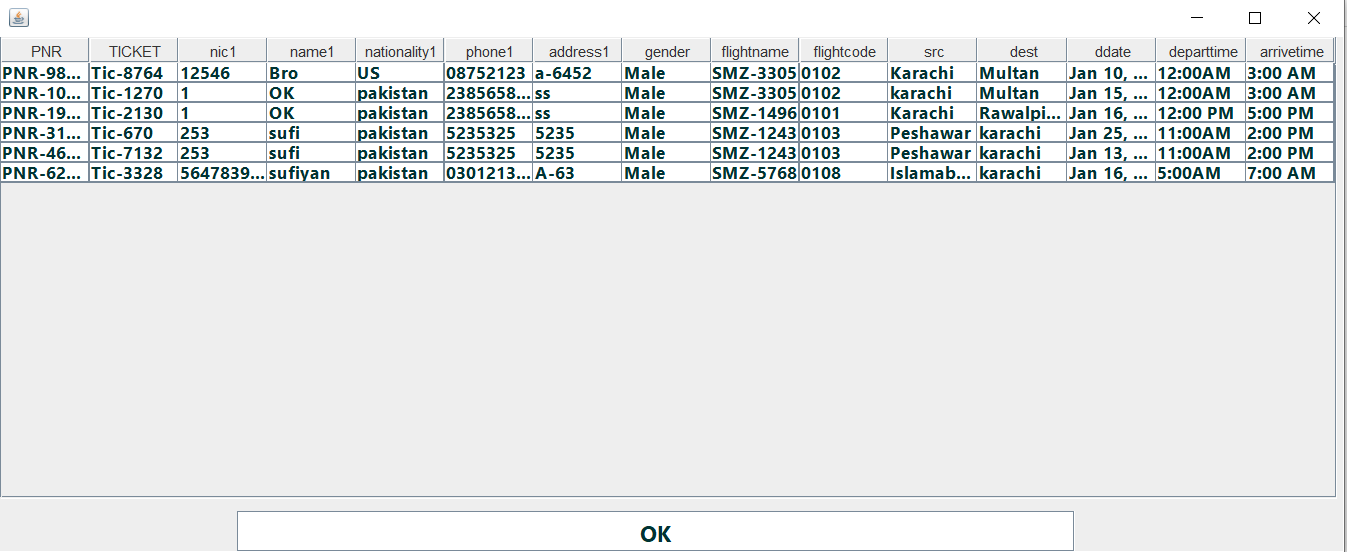
AFTER PICK CORRECT FLIGHT AND CLICK SEARCH (ALL FLIGHT INFO VISIBLE):

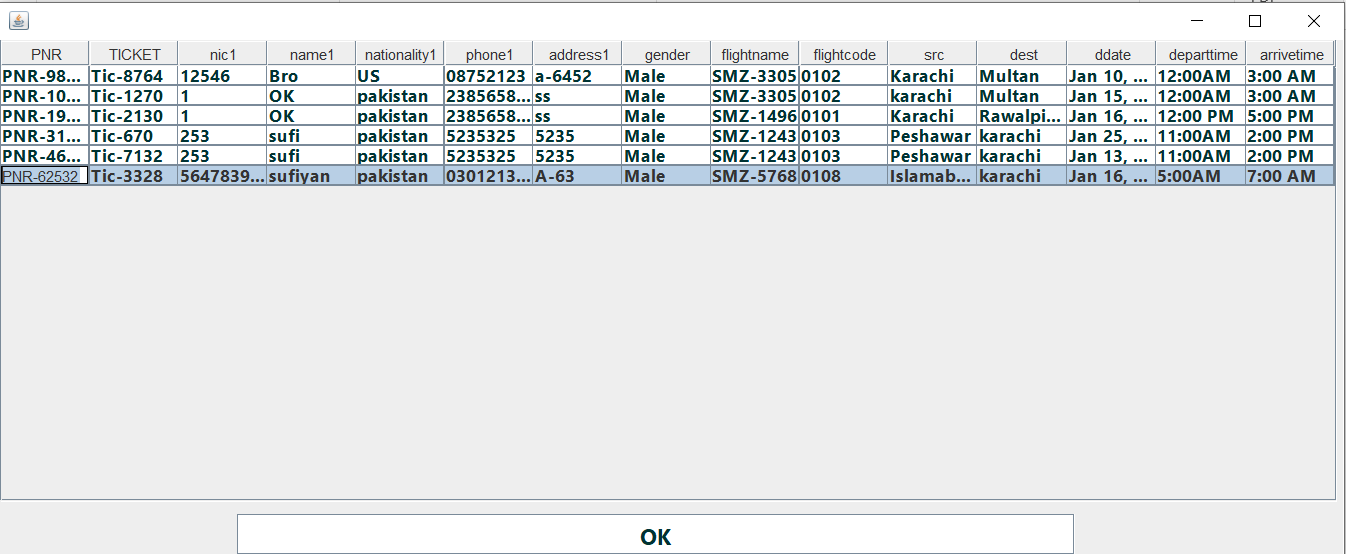
NOW CHOSSE DATE OF TRAVEL (FOR JCALENDAR):

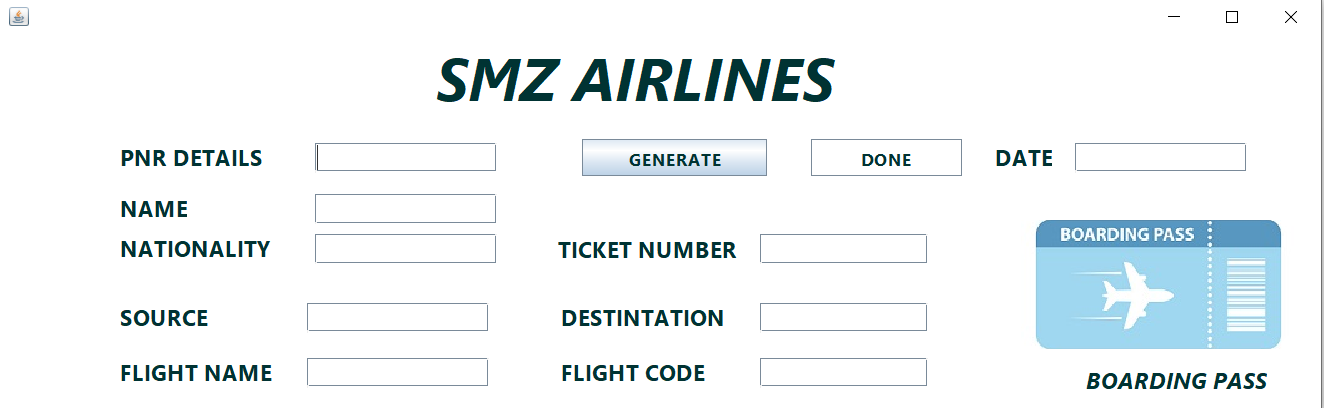


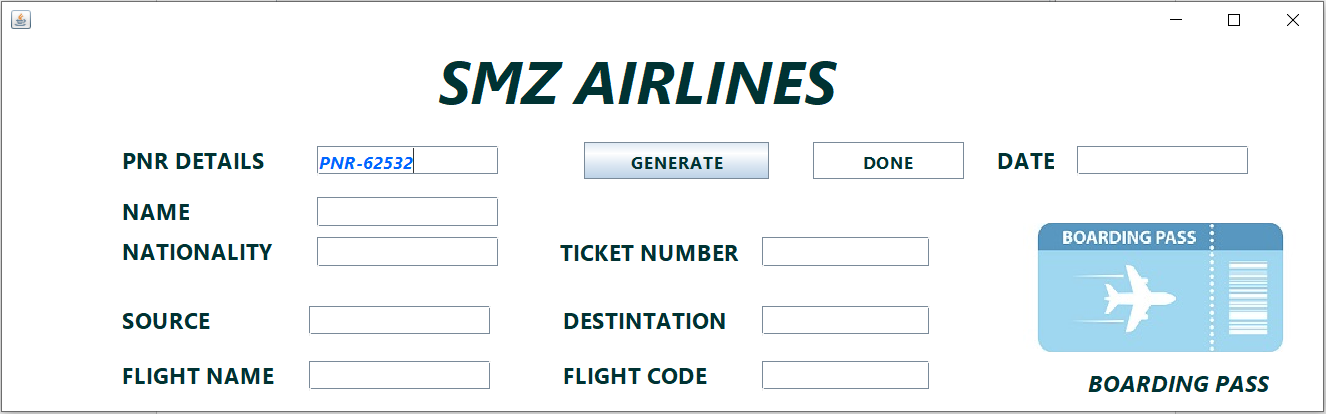
DATE IS SELECTED:

CLICK BOOK FLIGHT AND THE FLIGHT IS BOOKED:

AFTER CLICK OK YOU GO NO ALL INFO PAGE WHERE RECENTLY ADDED FLIGHT IS ALSO VISIBLE:

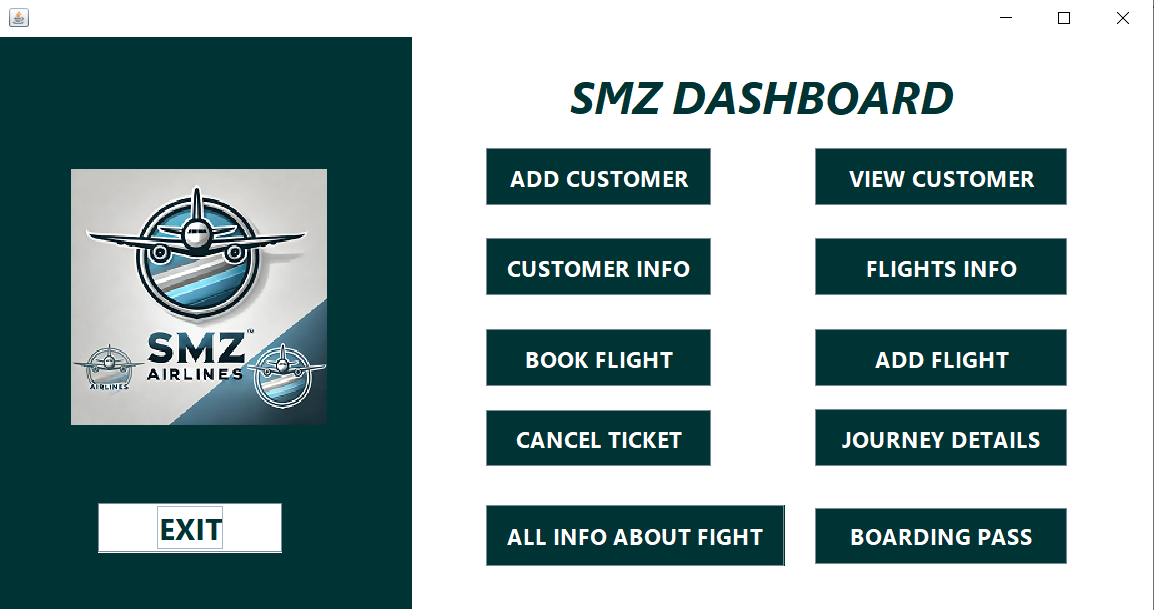
NOW FROM HERE COPY PNR NUMBER:

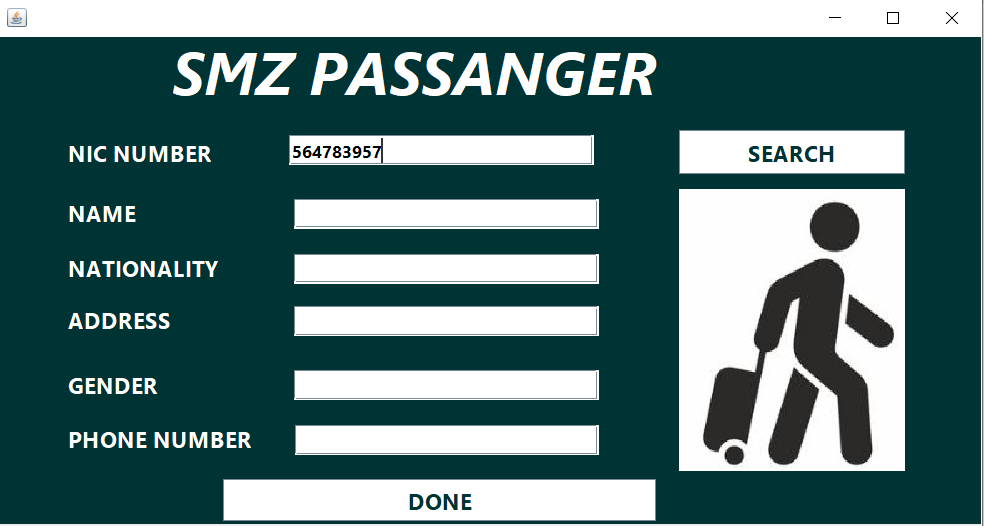
AFTER CLICK OK YOU GO ON BOARDING PASS PAGE:

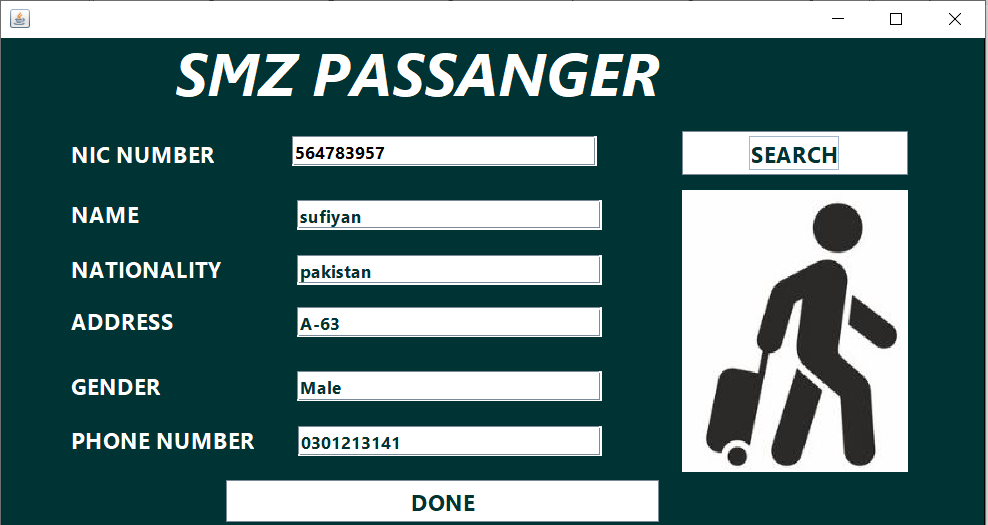
PASTE PNR NUMBER HERE AND CLICK GENERATE:

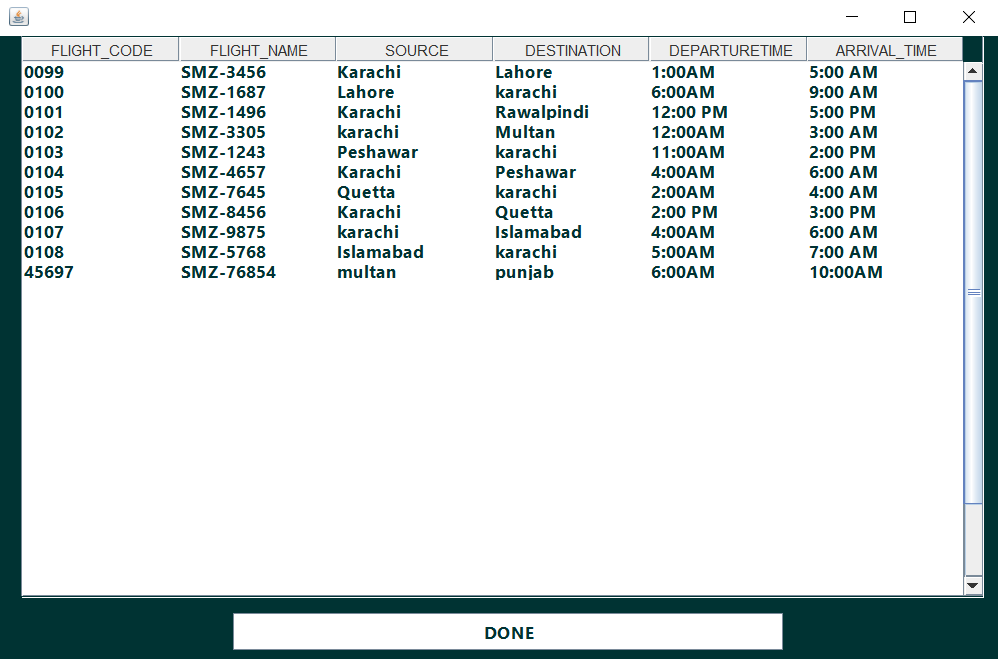
ALL INFO OF CUSTOMER VISIBLE NOW CLICK DONE:

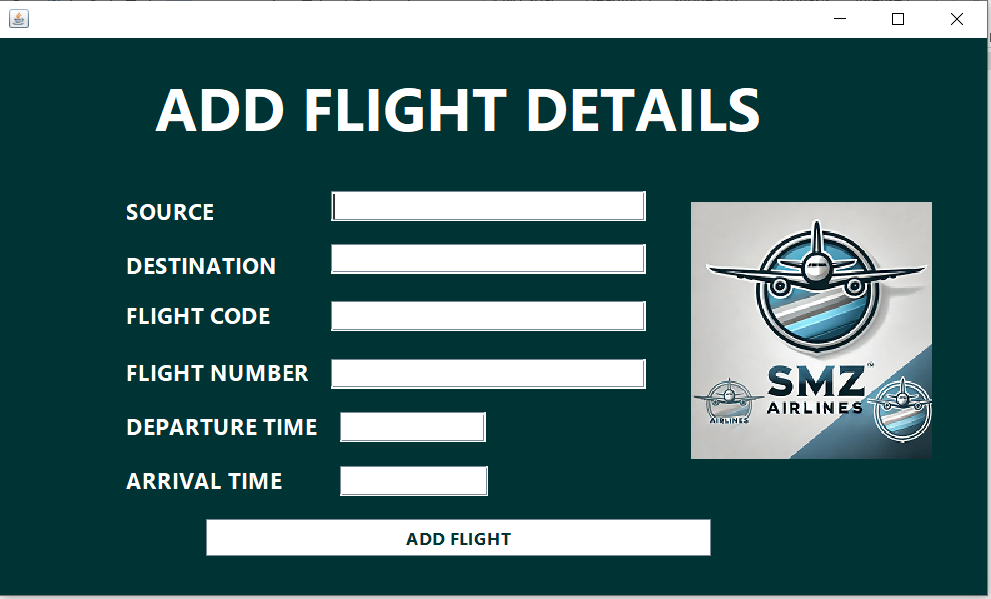
AFTER CLICK DONE BUTTON THANKYOU PAGE IS OPEN:

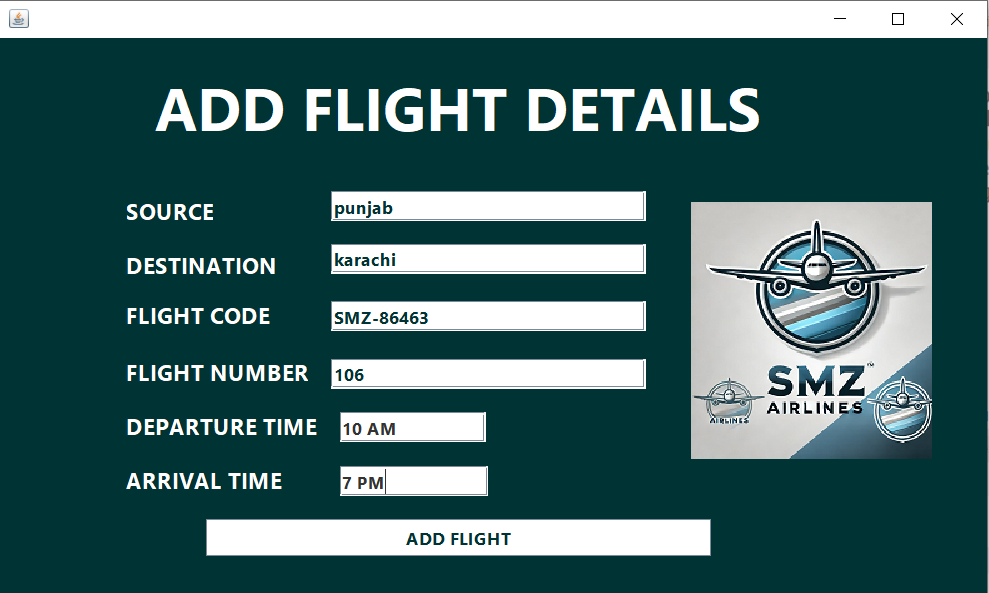
NOW DONE AND YOU ARE ON DASHBOARD PAGE:

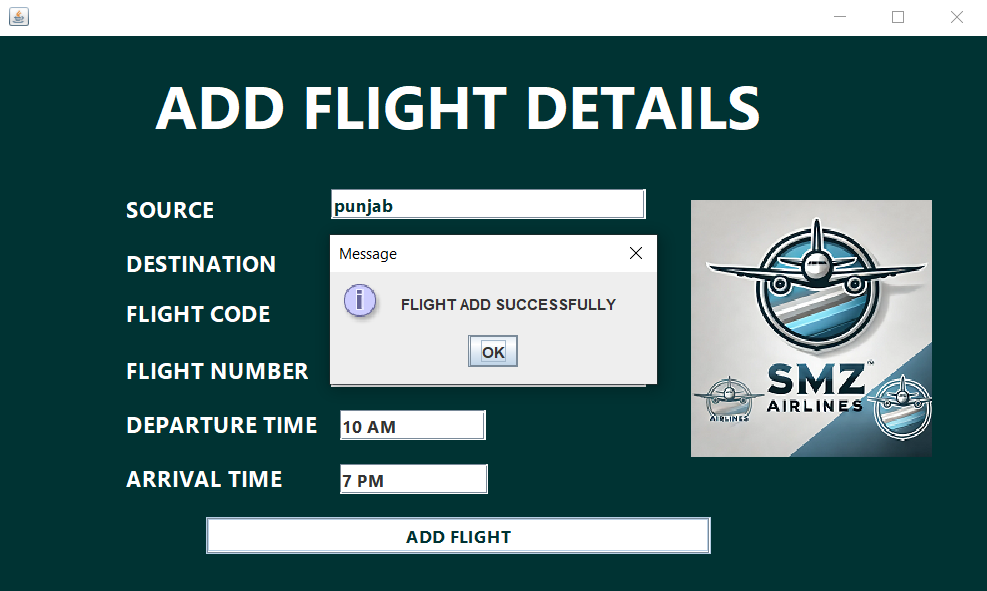
FROM DASHBOARD PAGE CLICK VIEW CUSTOMER AND ADD NIC NUMBER THEN SEARCH:

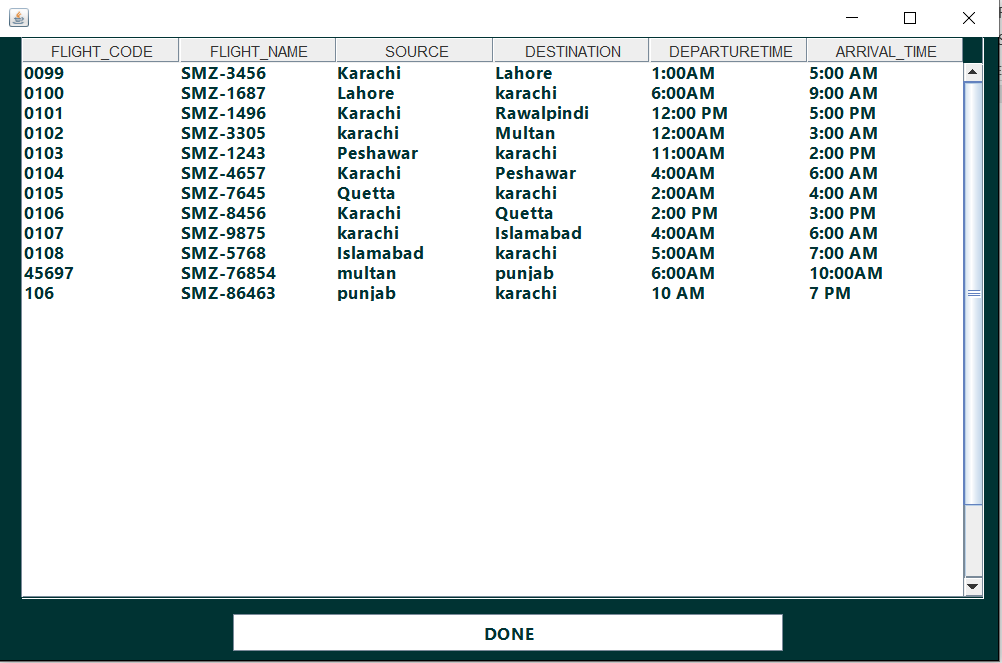
AFTER CLICK SEARCH BUTTON THE ALL DETAILS WERE VISIBEL:

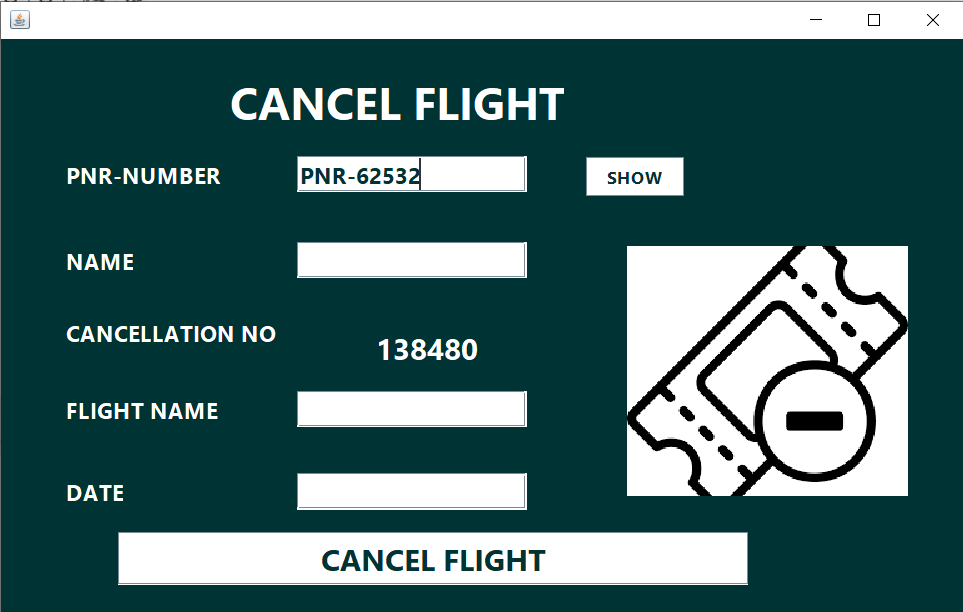
NOW FROM DASHBOARD WE CLICK ON FLIGHT INFO BUTTON (A BUTTON VISIBEL ON DASHBOARD):

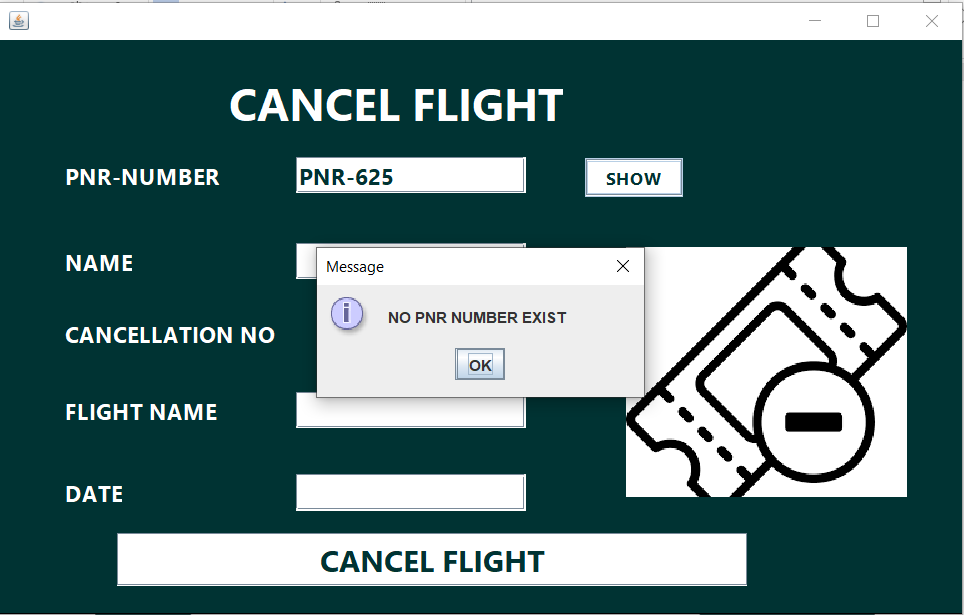
 NOW FROM DASHBOARD WE CLICK ON ADD FLIGHT BUTTON :

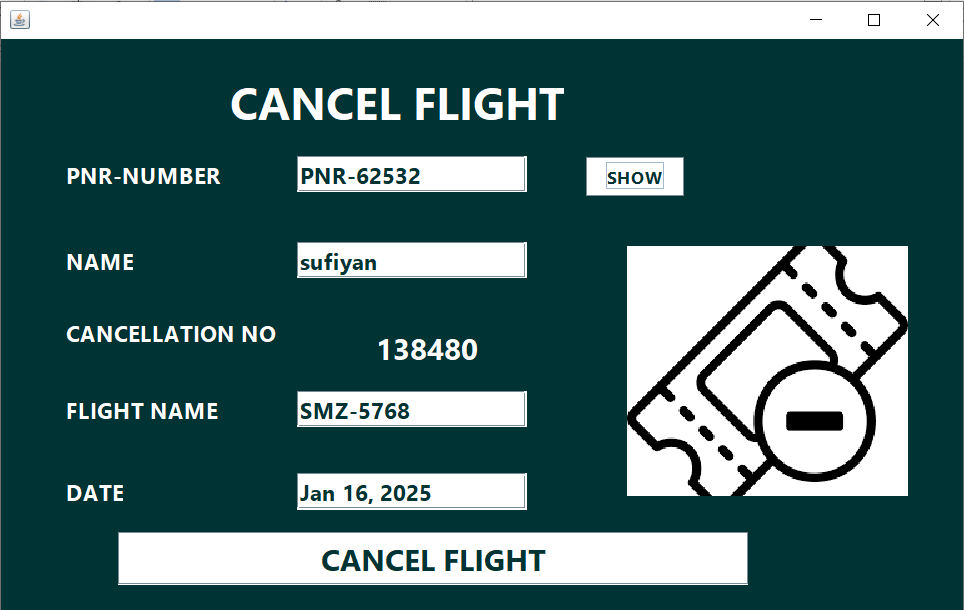
ADD FLIGHT DETAILS:

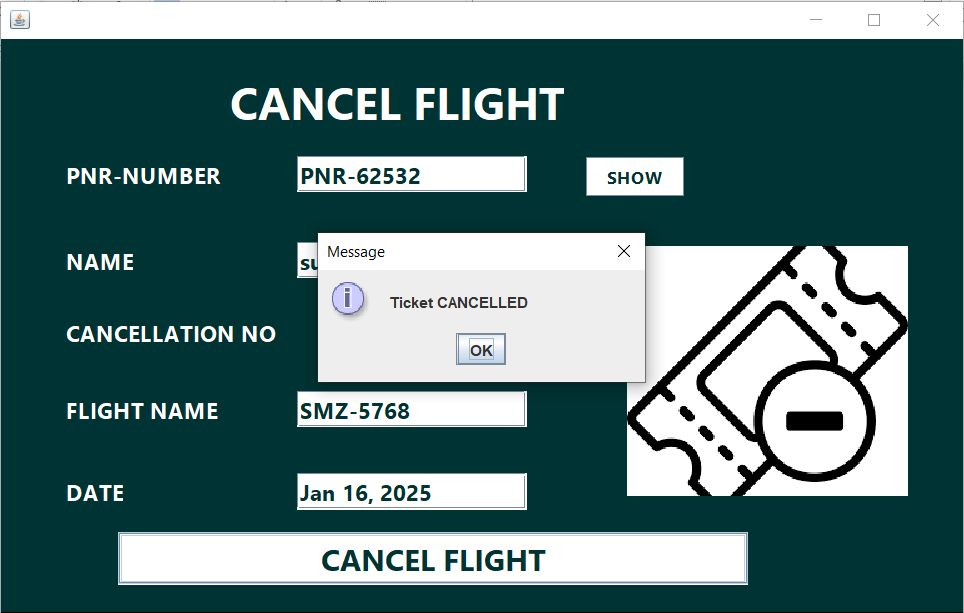
CLICK ADD FLIGHT AND MESSAGE WILL SHOW FLIGHT ADDED:

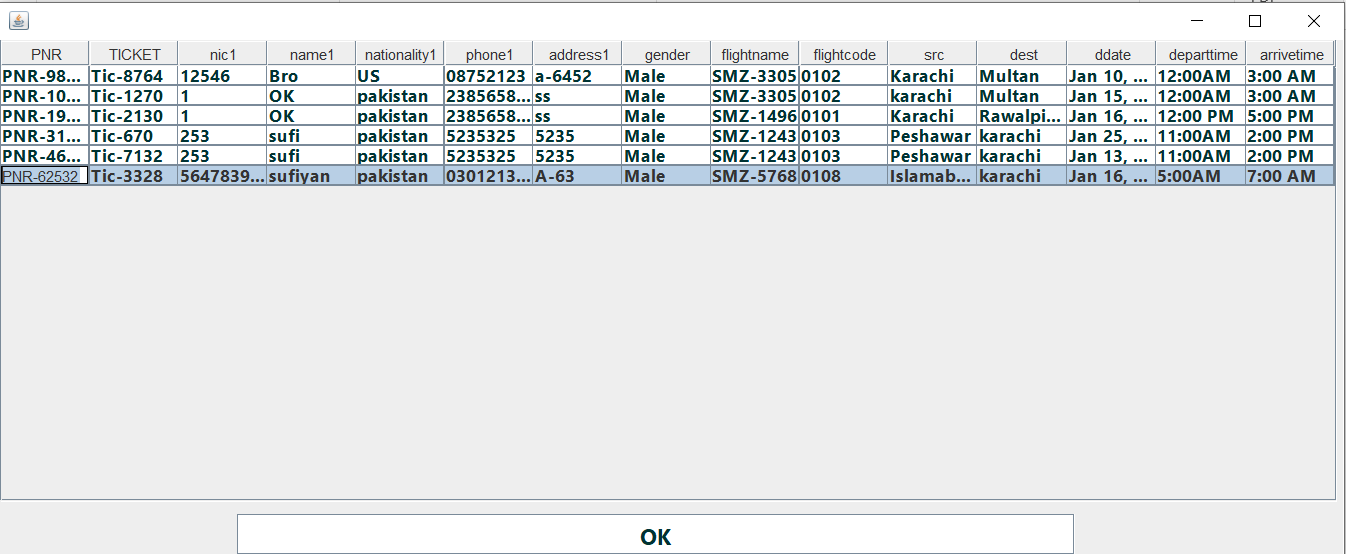
NOW WE GO ON FLIGHT INFO PAGE (WHERE A RECENTLY ADDED FLIGHT IS ALSO VISIBLE):

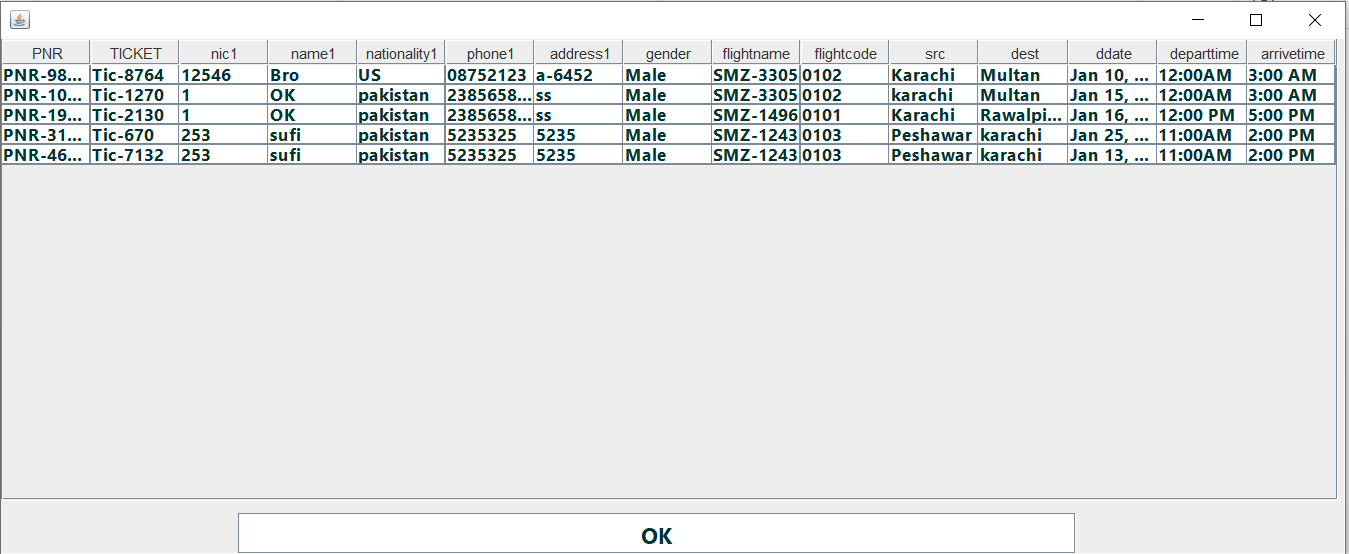
NOW WE CLICK ON CANCLE FLIGHT BUTTON AND ADD PNR NUMBER:

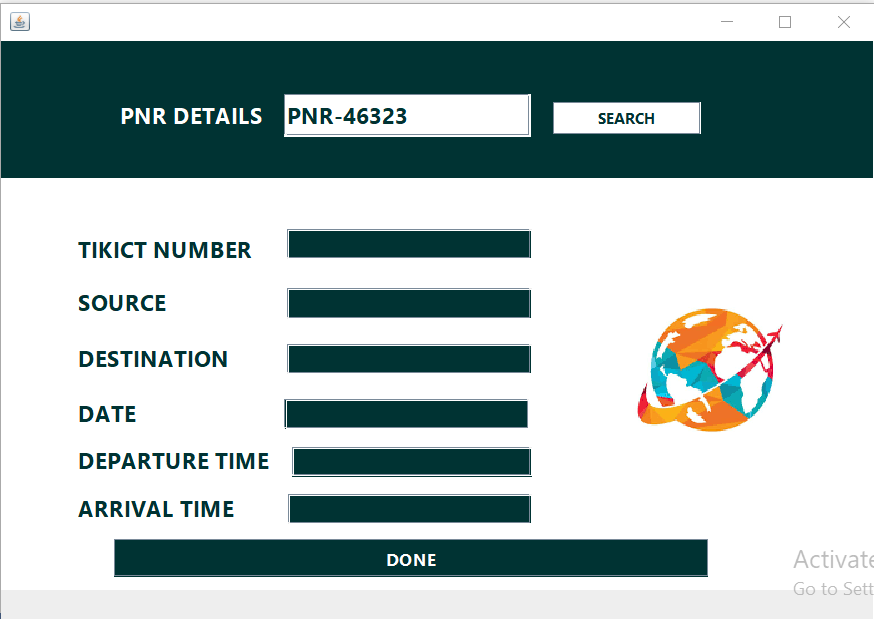
IF PNR NUMBER IS WRONG THE MESSAGE SHOW(NO PNR NO EXIST):

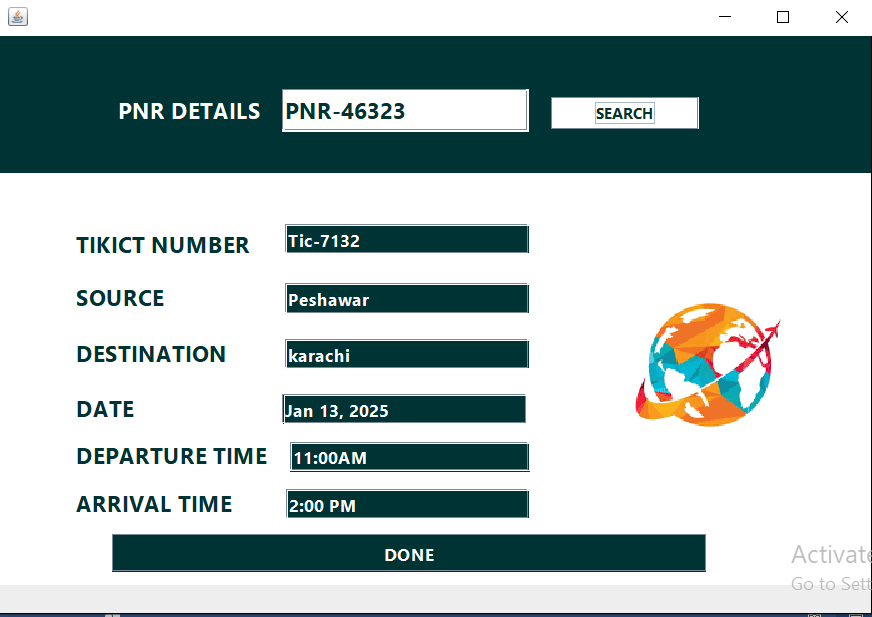
NOW I CLICKED SHOW WITH EXIST PNR NUMBER:

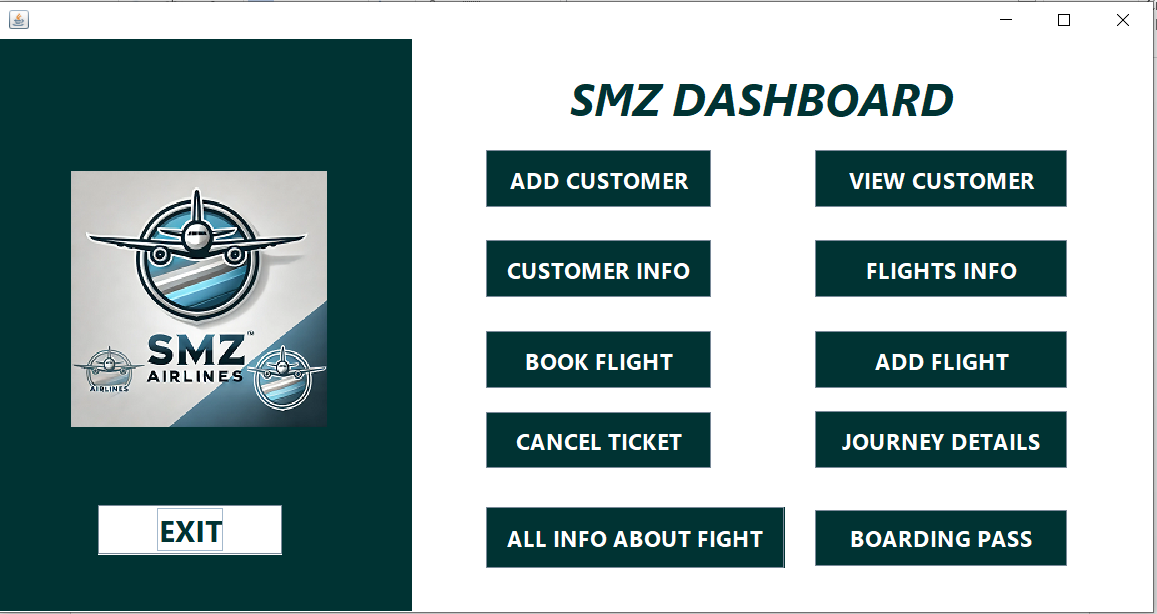
CLICK CANCEL FLIGHT BUTTON THE BOOKED FLIGHT IS DELETED:

BEFORE DELETION OF BOOKED FILGHT:

AFTER DELETE THE BOOKED FLIGHT:

NOW WE CLICK ON JOURNEY DETALS BUTTON:

ADD PNR NUMBER AND SEARCH THE INFO IS VISIBLE:

CLICK DONE BUTTON AND WE ARE AGIN ON THE DASHBOARD: