

## CHAPTER 1

### INTRODUCTION

Food is one of the basic human needs, but with an emphasis modern society puts on eating healthy, sophisticated and yet quickly (we have so much to do besides eating!) restaurants are often the place to go when we need a bite.

Dining out became a new kind of a pastime and an opportunity to diversify our life or just to have a good time. However, with the growing number of different eateries it's not always easy to make a right choice. Or, on the other hand, we may have a great restaurant that we will definitely like just two blocks from us, but we may also don't know that it even exists.

So as to solve such industry's issues, we can build a restaurant finder app. Moreover, we will not only make lives of many people much easier but also can take a great advantage of it!

If the city is big enough, choosing from a list of places nearby can be hard, without the restaurant ratings and reviews being available in our food finder app. Also we need to provide some suggestions (top places for lunch, if it's a lunchtime or hottest nightclub if it's Saturday night), as well as categories available, such as:

- Breakfast
- Lunch
- Drinks and nightlife
- Cafe

Restaurant Finder is a project which aims in developing a computerized system to maintain all the daily work of different Restaurants. It has a facility of admin login, through which the admin can monitor the whole system. It also has a facility where the user after logging in their accounts can see list of restaurants present in and around the city. Overall, this project of is being developed to help the admin and the user as well as the restaurant owners to maintain the details of the hotels in the best way possible and reduce the human efforts.

## 1.1 OBJECTIVES

The basic objectives of this project are:

- To understand and realize the real world project, understand the problems associated in developing it and explore the appropriate solution to it.
- To understand the application development in desktop application based system.
- To identify Java language components and how they work together in applications.
- To understand how to develop and implement swing GUI.
- To analyse and apply the JDBC concepts and create applications based on the database.
- To learn how to extend Java classes with inheritance and dynamic binding.
- To learn how to use exception handling in Java applications.
- To learn Java generics and how to use the Java Collections API.
- To understand how to design applications with threads in Java.
- To think in a way so as to make the life simpler and easier with the available technology and resources.

The specific objectives of this project are:

- To search easily for restaurants, cafes by location, cuisine (e.g. Chinese, Italian, Indian) and name (e.g. Starbucks, Dominos, McDonalds)
- To explore every restaurant in the city and use search filters to find the one that suits the user best.
- To rate and review the restaurants that the customers have been to.
- To get detailed information of the restaurant that the user is searching for.
- To get list of restaurant types in a city.
- To send a confirmation mail and message to the user whenever the user registers.

## **1.2 PROBLEM STATEMENT**

- To develop a computerized restaurant search application in Java using the OOPs concepts that meets the rising user interest in searching restaurants based on their convenience.
- The system should be able to locate a list of restaurants based on the location and type of the cuisine entered by the user.
- The user should, not only find all the restaurants in the city, but also should be able to make a choice of the best restaurant based on the rating.
- The user should also be given an option where they can give their opinion about the restaurants by giving a rating and writing a review.
- The system should be able to use a GUI for the front end and JDBC concepts for the database in the backend.
- The system should be able to check the validity of input data and give a feedback to the user in case of errors or inconsistency using exception handling.
- The system should be user friendly and convenient to use.

## **1.3 OUTCOMES OF THE PROJECT WORK CARRIED OUT**

All the mentioned objectives of this project are carried out successfully.

- Understood the application development in desktop application based system.
- Analysed and applied the JDBC concepts and linked the database and Java UI.
- Understood how to implement and develop swing GUI.
- The system is made user friendly and convenient to use.
- The system is able to check the validity of input data and give a feedback to the user in case of errors or inconsistency using exception handling.
- Restaurants, cafes can be searched easily based on the region and cuisines.
- Detailed information of the restaurant will be displayed to the users searching for it.
- The user can rate and review the restaurants that they have been to.
- The rating provided by the users who previously visited the restaurants will be displayed to the other users helping them choose the restaurants.

## CHAPTER 2

# JAVA FEATURES AND OOPS CONCEPT

### 2.1 FEATURES

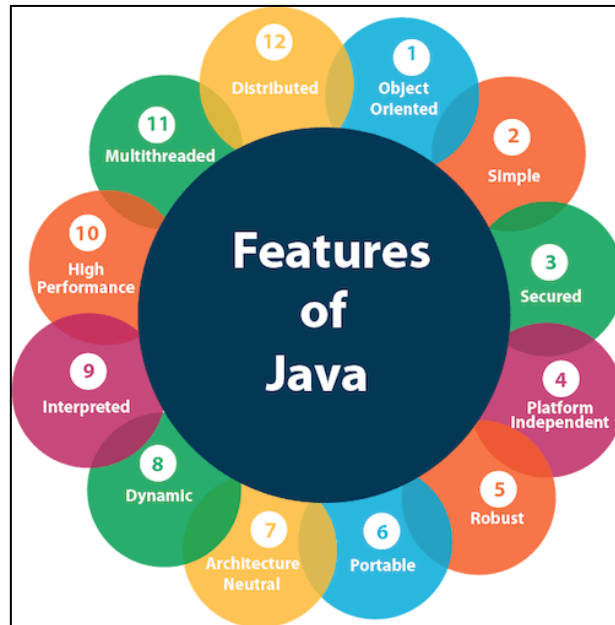


Figure 2.1.1: Features of Java

- **Simple-** Java is easy to learn and its syntax is quite simple, clean and easy to understand.
- **Object Oriented-** In Java everything is Object which has some data and behaviour. Java can be easily extended as it is based on Object Model.
- **Robust-** Java makes an effort to eliminate error prone codes by emphasizing mainly on compile time error checking and runtime checking.
- **Platform Independent-** Java is guaranteed to be write-once, run-anywhere language.
- **Secure-** Java program always runs in Java runtime environment with almost null interaction with system OS, hence it is more secure.
- **Multi threading-** Java multithreading feature makes it possible to write program that can do many tasks simultaneously.
- **Architectural Neutral-** Compiler generates byte codes, which have nothing to do with particular computer architecture; hence a Java program is easy to interpret on any machine.

## 2.2 OOPs CONCEPTS

The core OOPs concepts are:

- **Abstraction-** Abstraction is the concept of hiding the internal details and describing things in simple terms. The internal processing of the method is hidden from the outer world. There are many ways to achieve abstraction in object-oriented programming, such as encapsulation and inheritance.

```
public class UserLogin extends JFrame {  
    static UserLogin frame;  
    private JPanel contentPane;           //Hiding of data  
    private JTextField textField;  
    private JPasswordField passwordField;
```

Figure 2.2.1: Abstraction

- **Encapsulation-** Encapsulation is the technique used to implement abstraction in object-oriented programming. Encapsulation is used for access restriction to class members and methods. It is the way of wrapping data and methods inside in a class.

```
public class AdminSuccess extends JFrame {  
    static AdminSuccess frame;           //Data Members  
    private JPanel contentPane;  
  
    public static void main(String[] args) {  
        EventQueue.invokeLater(new Runnable() {  
            public void run() {           //Member Methods  
                try {  
                    frame = new AdminSuccess();  
                    frame.setVisible(true);  
                } catch (Exception e) {  
                    e.printStackTrace();  
                }  
            }  
        });  
    }  
}
```

Figure 2.2.2: Encapsulation

- **Polymorphism-** Polymorphism is the concept where an object behaves differently in different situations. There are two types of polymorphism- compile time and runtime.

```
public void actionPerformed(ActionEvent e) {
    String name = textField.getText();
    String password = String.valueOf(passwordField.getPassword());
    //System.out.println(name+" "+password);
    if (UserDao.validate(name, password)) {
        UserSuccess.main(new String[]{});
        frame.dispose();
    } else {
        JOptionPane.showMessageDialog( parentComponent: UserLogin.this,
            message: "Sorry, Username or Password Error",
            title: "Login Error!", JOptionPane.ERROR_MESSAGE);
        textField.setText("");
        passwordField.setText("");
    }
}

});

JButton btnBack = new JButton( text: "Back");
btnBack.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        Restaurant.main(new String[]{});
        frame.dispose();
    }
});
```

Figure 2.2.3: Polymorphism

- **Inheritance-** Inheritance is the mechanism of code reuse. The object that is getting inherited is called super class and the object that inherits the super class is called subclass.

```
/**
 * Extends Swing JFrame class
 */
public class UserForm extends JFrame {
    static UserForm frame;
    private JPanel contentPane;
    private JTextField textField;
    private JTextField textField_1;
    private JTextField textField_2;
    private JTextField textField_3;
    private JPasswordField passwordField;
```

Figure 2.2.4: Inheritance

## CHAPTER 3

# REQUIREMENTS AND DESIGN

The project involved analysing the design of few applications so as to make the application more users friendly. To do so, it was really important to keep the navigation from one screen to the other well ordered and at the same time reducing the amount of typing the user needs to do. In order to make the application more accessible, the JDK version had to be chosen. As Java is a platform independent language, it can run on any system in which Java JDK is installed. It can be Windows, MAC or any UNIX distribution.

### 3.1 HARDWARE SPECIFICATIONS

- Processor - Intel Core i5
- Speed - 1.8 GHz
- RAM - 256 MB
- Hard Disk - 10 GB

### 3.2 SOFTWARE SPECIFICATIONS

- Operating System - Any JDK installed OS
- Database - MySQL
- GUI used – Java Swing

```
public class DB {  
    public static Connection getConnection() {  
        Connection con=null;  
        try{  
            Class.forName("com.mysql.jdbc.Driver");  
            con=DriverManager.getConnection( url: "jdbc:mysql://localhost:3306", user: "root", password: "");  
            con.prepareStatement( sql: "CREATE DATABASE IF NOT EXISTS `test`").execute();  
            con.prepareStatement( sql: "USE `restaurant`").execute();  
        }catch(Exception e){System.out.println(e);}  
        return con;  
    }  
}
```

Figure 3.2.1: JDBC Connection

### 3.3 USE CASE DIAGRAM

The purpose of a use case diagram is to demonstrate the different ways that a user might interact with a system. It captures the actors and the role they perform in a system.

Here, there are two actors, represented by stick figures:

- Admin
- User

The horizontal shaped ovals represent the different use cases.

The association is represented by a line between actors and use cases.

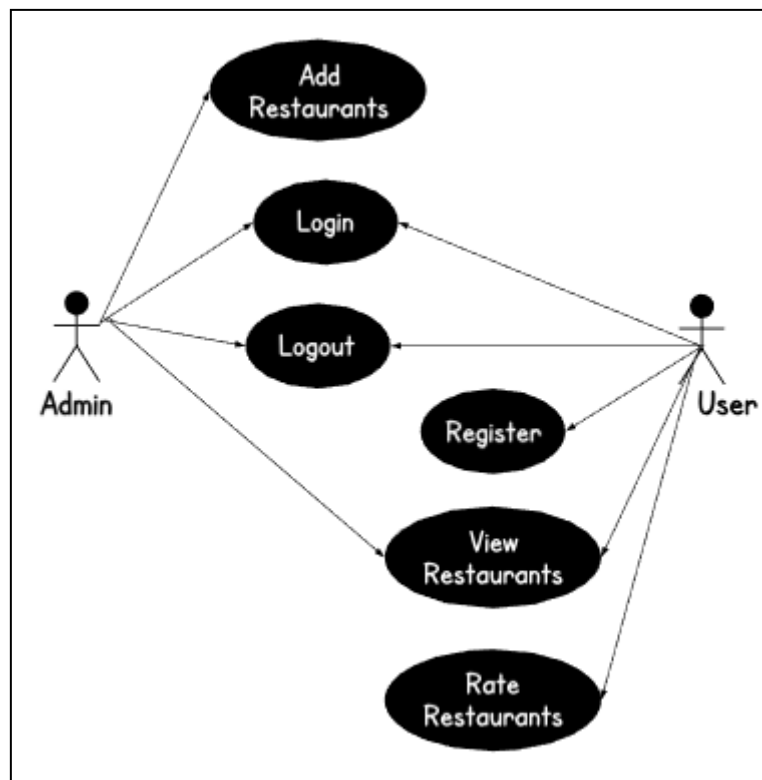


Figure 3.3.1: Use Case Diagram



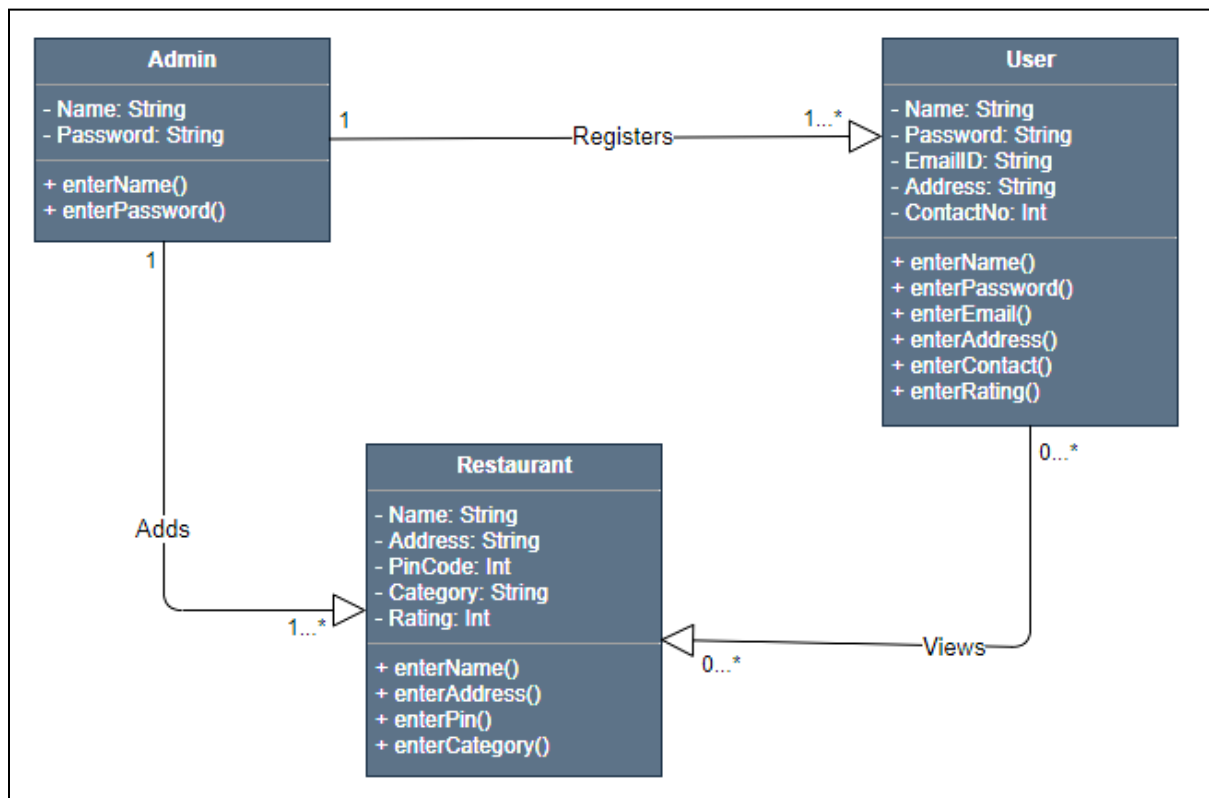
### 3.4 CLASS DIAGRAM

Class diagrams offer a number of benefits. We use UML class diagrams to:

- Illustrate data models for information systems, no matter how simple or complex.
- Better understand the general overview of the schematics of an application.
- Visually express any specific needs of a system and disseminate that information throughout the business.

#### Database Assumptions:

- One admin can add one or more restaurants. But one restaurant can be added by one and only one admin.
- One user can view zero or many restaurants. Also one restaurant can be viewed by zero or many users.
- One admin can register one or more user. But one user can be registered by one and only one admin.



3.4.1: Class Diagram

### 3.5 SCHEMA

The term "database schema" can refer to a visual representation of a database, a set of rules that govern a database, or to the entire set of objects belonging to a particular user.

There are 2 tables in the database:

- Users- To store the details of various users that use the application
- Restaurants- To store the details of the restaurants added by the administrator.

```
mysql> desc users;
```

Field	Type	Null	Key	Default	Extra
u_id	int(5)	NO	PRI	NULL	auto_increment
name	varchar(100)	NO		NULL	
password	varchar(100)	NO		NULL	
email	varchar(100)	NO		NULL	
address	varchar(200)	NO		NULL	
contact	varchar(20)	NO		NULL	

6 rows in set (0.00 sec)

Figure 3.5.1: Users table description

```
mysql> desc restaurants;
```

Field	Type	Null	Key	Default	Extra
r_id	int(10)	NO	PRI	NULL	auto_increment
name	varchar(100)	NO		NULL	
address	varchar(100)	NO		NULL	
pincode	int(6)	NO		NULL	
category	varchar(50)	NO		NULL	
rating	float	NO		0	
no_of_ratings	int(10)	NO		0	

7 rows in set (0.00 sec)

Figure 3.5.2: Restaurants table description

### 3.6 SEQUENCE DIAGRAM

Sequence Diagrams are interaction diagrams that detail how operations are carried out. They capture the interaction between objects in the context of collaboration. Sequence Diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

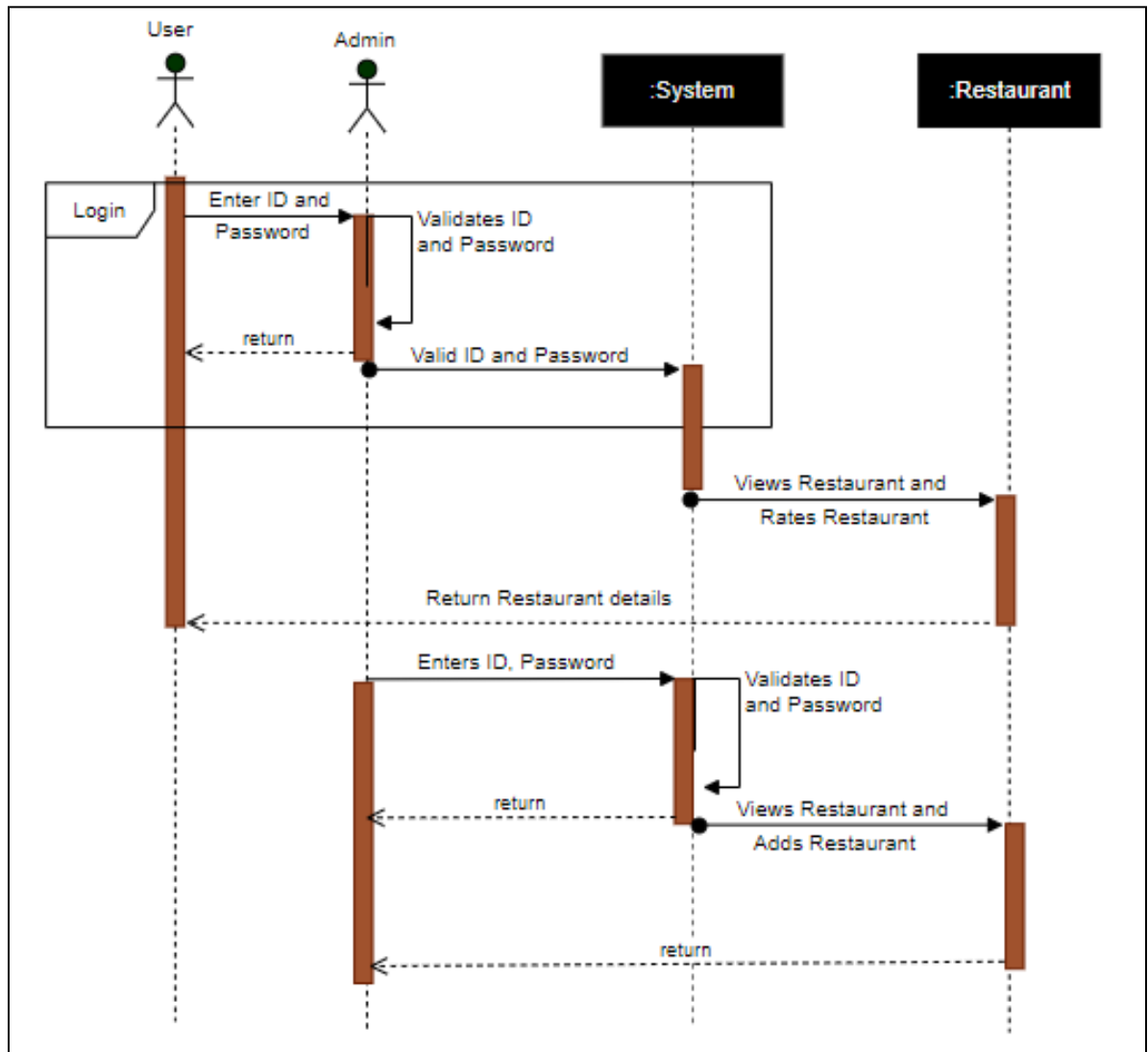


Figure 3.6.1: Sequence Diagram

## CHAPTER 4

### IMPLEMENTATION

JAVA Swing has been used for the User Interface part which is used to make the Content Box, Buttons, Dialog Box and other styling part. MySQL is used for storing of the data. The data includes the detail of the restaurants and the users. The details of the restaurant include ID, name, address, rating and number of ratings, email ID and phone number of the restaurant. The details of the user includes the name, phone number, email ID of the user.

- The first screen which opens after running the project is where the viewer decides to login as an Admin or a User. If the viewer is Admin, he/she can login using the admin name and the password. The Admin has the authority to add a new restaurant. He/she can also delete the restaurant if required.
- On the other hand, if the viewer is a User, he/she has to login with the username and password. The user can view the restaurants present. He/she can search restaurants based on different categories, like locality or ratings. The user can also rate the restaurant which he/she has visited.

#### a) User Login

##### ➤ Description of feature:

This feature is used by the user to login to the system. They are required to enter their user id and password before they can enter the system. The user id and password will be verified and if invalid id is encountered, user cannot enter the system.

##### ➤ Functional requirements:

- User id is provided when they register.
- The system must only allow user with valid id and password to enter the system.
- The system performs authorization process which decides what user level can access to.
- The user must be able to logout after they finish using the system.

**b) New User Registration**

➤ Description of feature:

This feature can be performed by all users to register new user to create account.

➤ Functional requirements:

- System must be able to verify information.
- System must be able to delete information if the entered details are wrong.

**c) Add New Restaurants- By the Admin**

➤ Description of feature:

This feature allows the admin to add new restaurants to the Restaurant Functional requirements.

➤ Functional requirements:

- System must be able to verify information.
- System must be able to enter the number of copies into the database table.
- System must be able to not allow 2 restaurants with the same restaurant id.

## CHAPTER 5

### OUTPUT SNAPSHOTS

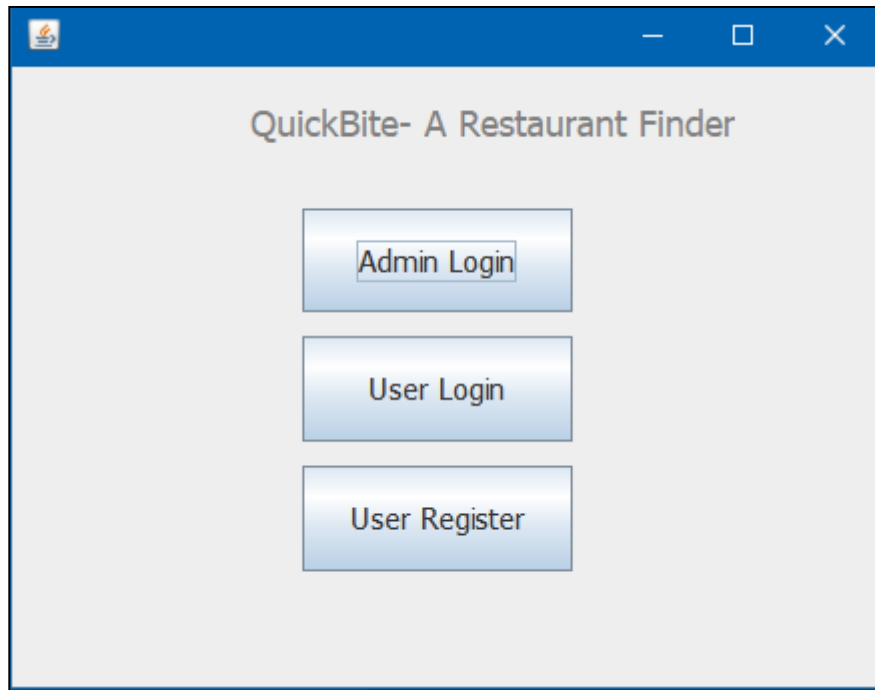


Figure 5.1: Home Screen

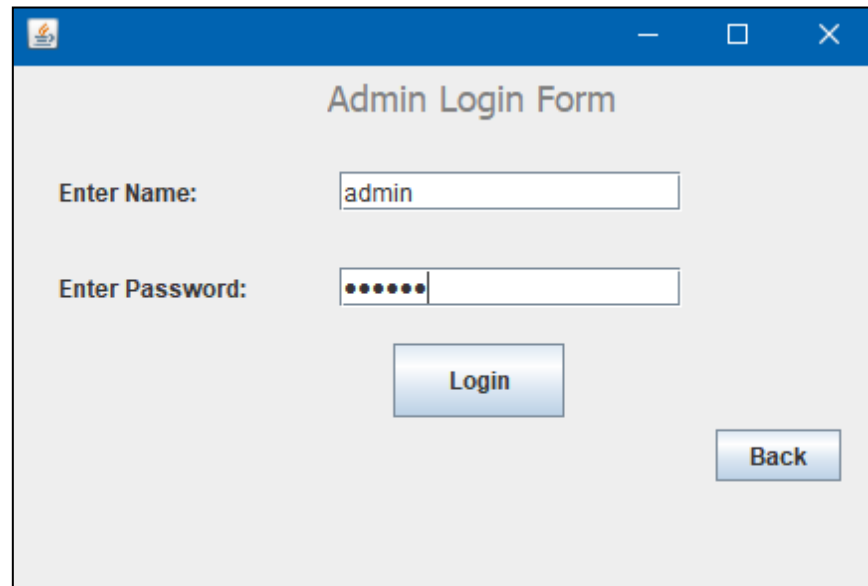


Figure 5.2: Admin Login

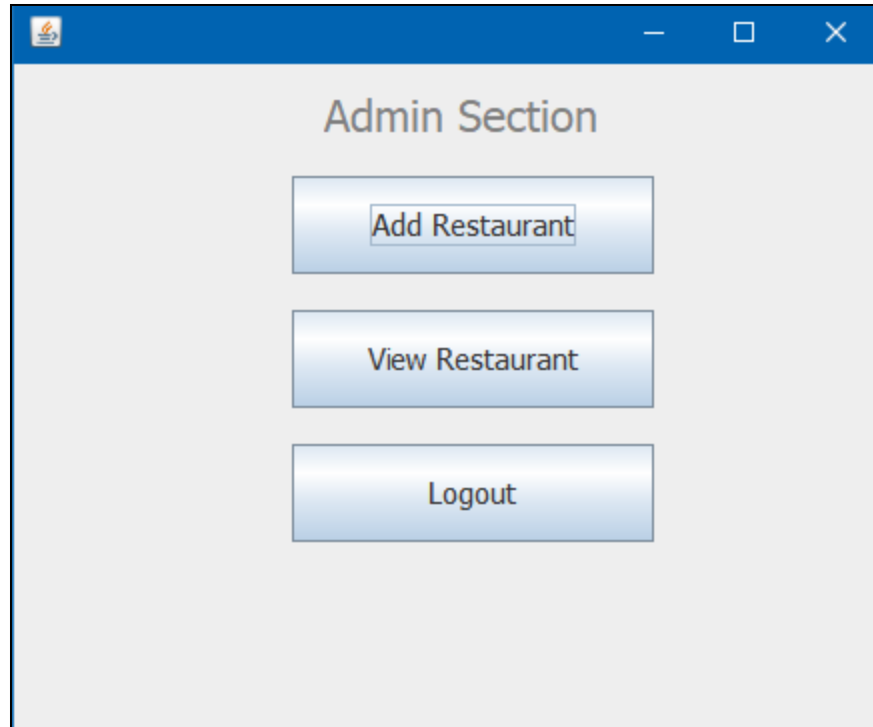


Figure 5.3: Admin Section

A screenshot of a web application window titled "Add Restaurant". The window has a blue header bar with a small logo on the left and standard window controls (minimize, maximize, close) on the right. The main content area is light gray and contains a form with four labeled text input fields: "Name:" with the value "Meghana's Biryani", "Address:" with the value "Marathalli, Bangalore", "PIN Code:" with the value "560037", and "Category:" with the value "Biryani". Below the form are two blue buttons with white text: "Add" and "Back".

Figure 5.4: Add Restaurant

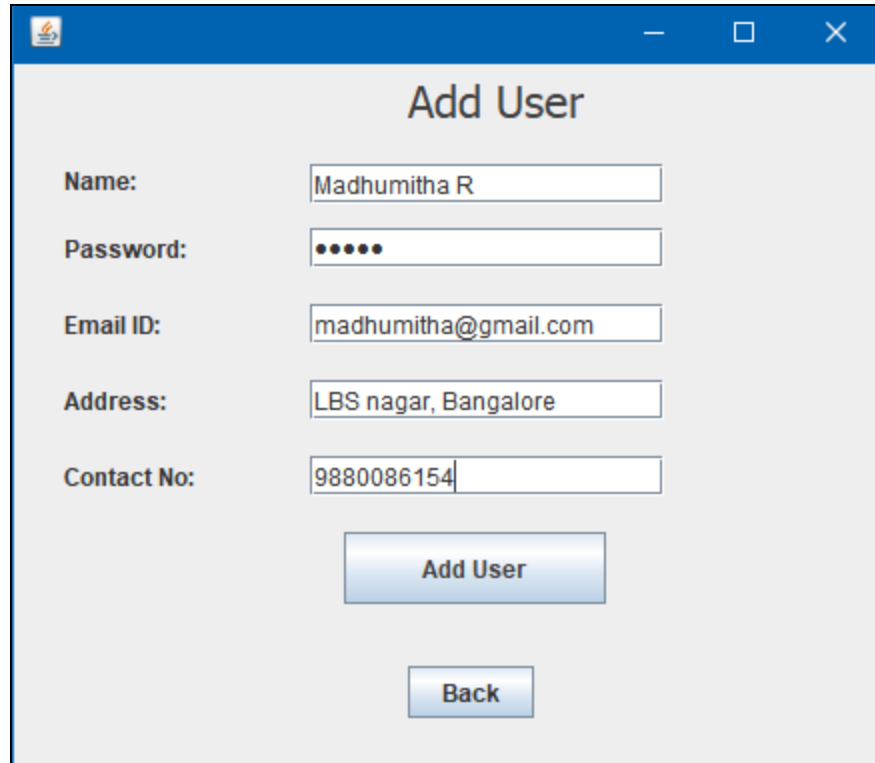
The screenshot shows a window titled "Add Restaurant". It contains four input fields: "Name:" with the text "Meghana's Biryani", "Address:", "PIN Code:", and "Category:". Below these fields are two buttons: "Add" and "Back". A modal message box is displayed in the center, titled "Message", with an information icon and the text "Restaurant added successfully!". The message box has an "OK" button.

Figure 5.5: Restaurant added successfully prompt

r_id	name	address	pincode	category	rating	no_of_ratings
4	Meghana's Biryani	Marathalli, Bangalore	560037	Biryani	4.5	2
5	Gooley & Fluffy	LBS nagar, Bangalore	560023	Bakery	3	1
6	Warm Oven	Hal, Bangalore	560017	Bakery	4	1
7	Behrouz Biryani	Wadala, Mumbai	730021	Biryani	3.8	5
8	Charcoal Eats	Parel, Mumbai	730045	Fast food	4	1
9	The Good Bowl	LP nagar, Delhi	450067	North Indian	2.5	2
10	Smoor	Indiranagar, Bangalore	560032	Cafe	4	1
11	Dum Biryani Hub	Indiranagar, Bangalore	560032	Biryani	5	1
12	Chai Point	Wadala, Mumbai	730021	Cafe	3	1
13	Lazeez Xpress	LBS nagar, Bangalore	560023	North Indian	3.75	4
14	Delhi Highway	LP nagar, Delhi	450067	Fast food	3.5	2

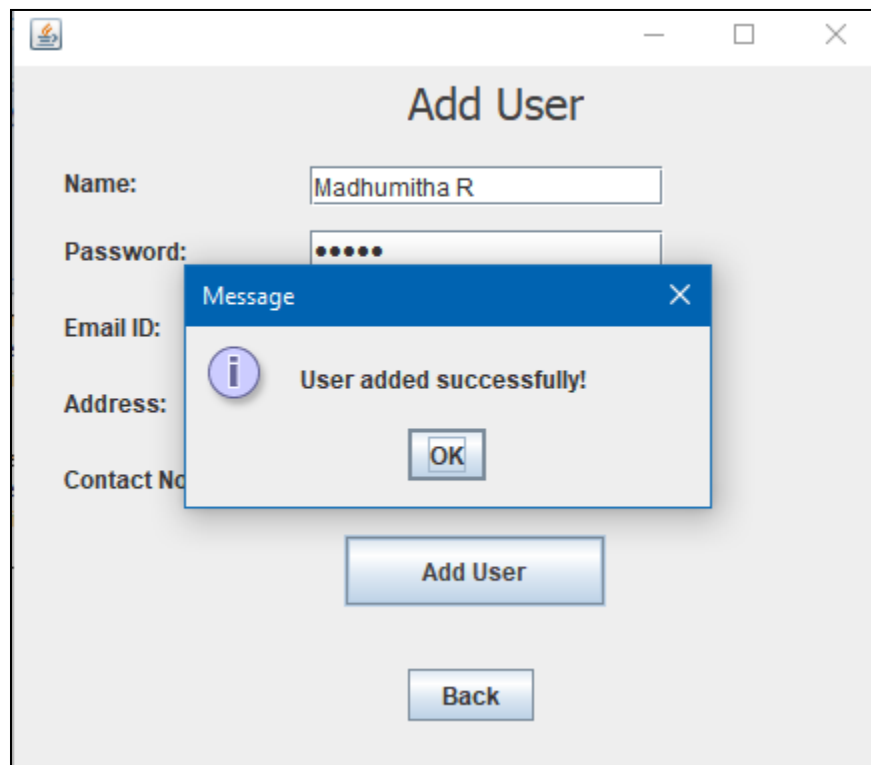
Figure 5.6: View Restaurant





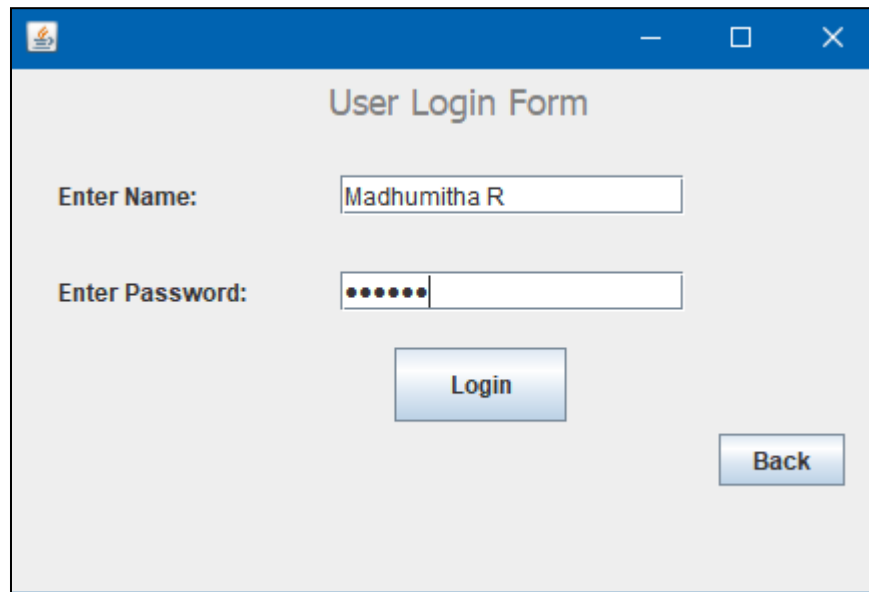
The screenshot shows a window titled "Add User" with a blue header bar. Inside the window, there are five input fields with labels to their left: "Name:" with the text "Madhumitha R", "Password:" with five dots, "Email ID:" with the text "madhumitha@gmail.com", "Address:" with the text "LBS nagar, Bangalore", and "Contact No:" with the text "9880086154". Below these fields are two buttons: "Add User" and "Back".

Figure 5.7: Add User



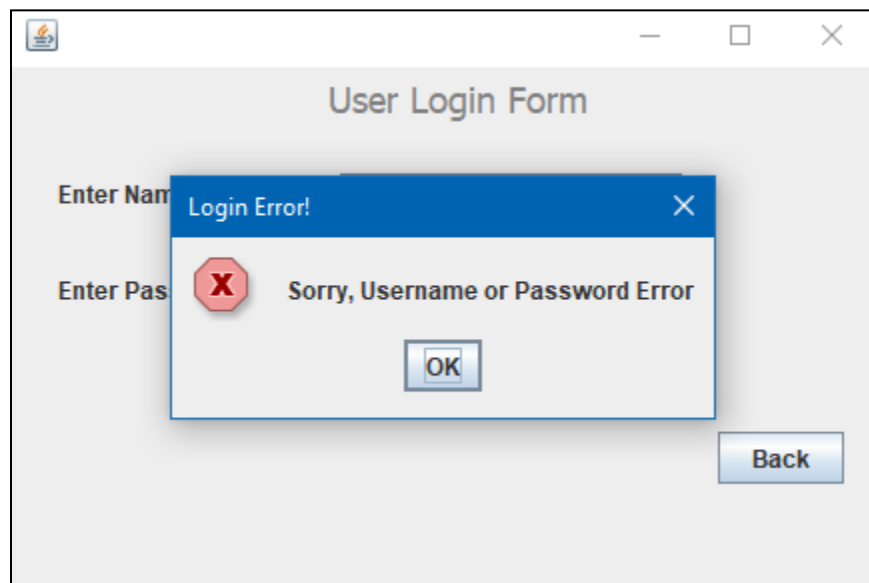
This screenshot shows the same "Add User" form as Figure 5.7, but with a modal dialog box overlaid in the center. The dialog box has a blue header bar with the title "Message" and a close button (X). The main content of the dialog box features an information icon (i) on the left and the text "User added successfully!" on the right. Below this text is an "OK" button. The background form is partially visible behind the dialog box.

Figure 5.8: User added successfully prompt



A screenshot of a Windows application window titled "User Login Form". The window has a blue title bar with standard minimize, maximize, and close buttons. The main area is light gray. It contains two labels: "Enter Name:" and "Enter Password:". The "Enter Name:" label is followed by a text box containing the text "Madhumitha R". The "Enter Password:" label is followed by a password box with seven dots and an empty space. Below the password box is a "Login" button. To the right of the "Login" button is a "Back" button.

Figure 5.9: User login



A screenshot of the same "User Login Form" window, but with an error dialog box overlaid. The dialog box has a blue title bar with the text "Login Error!" and a close button. The main area of the dialog box is white and contains a red octagonal icon with a white "X" on the left. To the right of the icon is the text "Sorry, Username or Password Error". Below this text is an "OK" button. The "User Login Form" window is partially visible behind the dialog box, showing the "Enter Name:" and "Enter Password:" labels and the "Back" button.

Figure 5.10: Incorrect Password Prompt

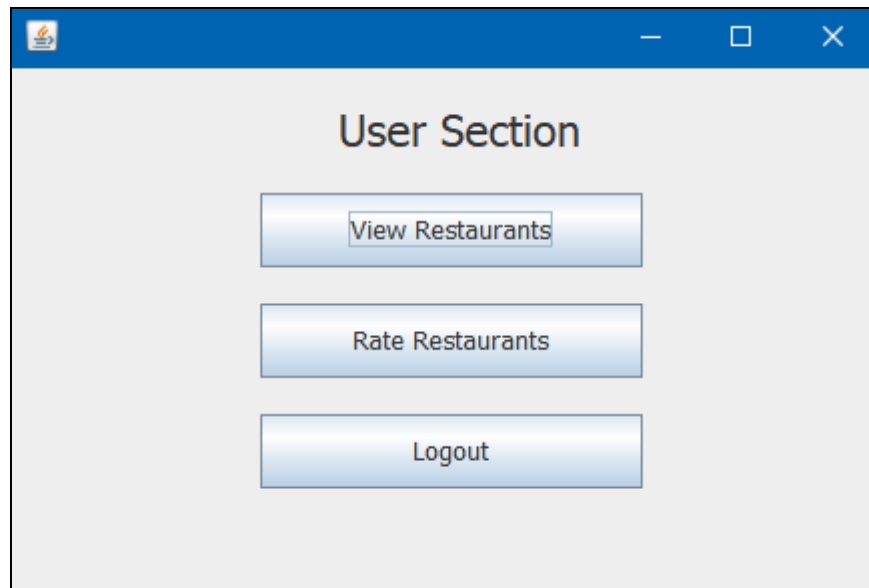


Figure 5.11: User Section

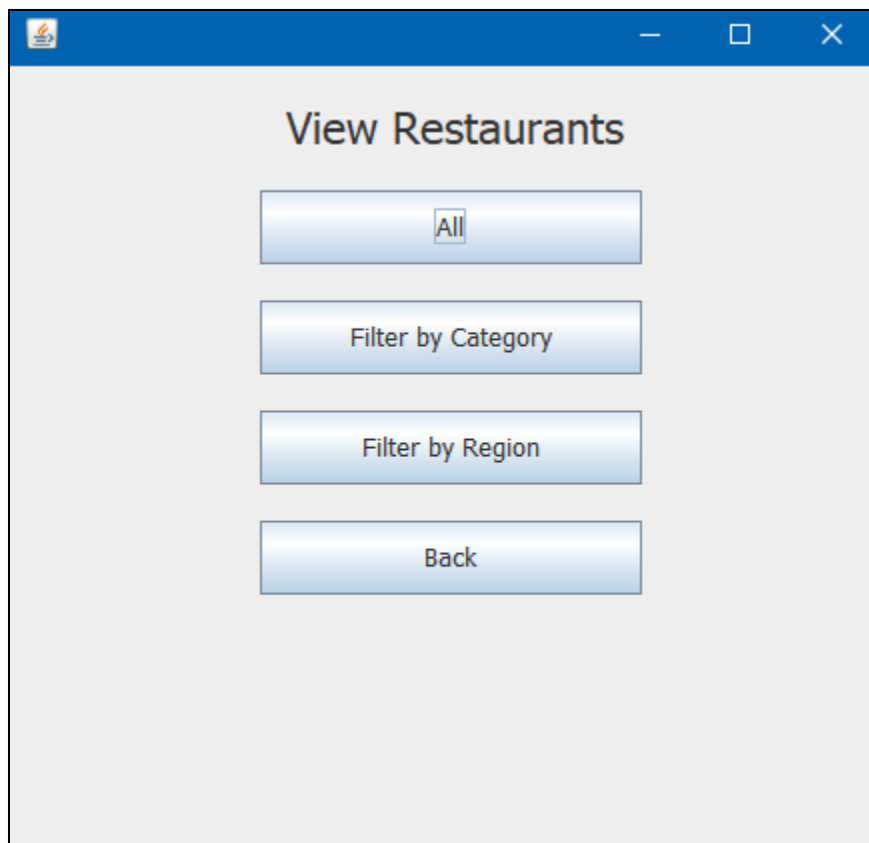
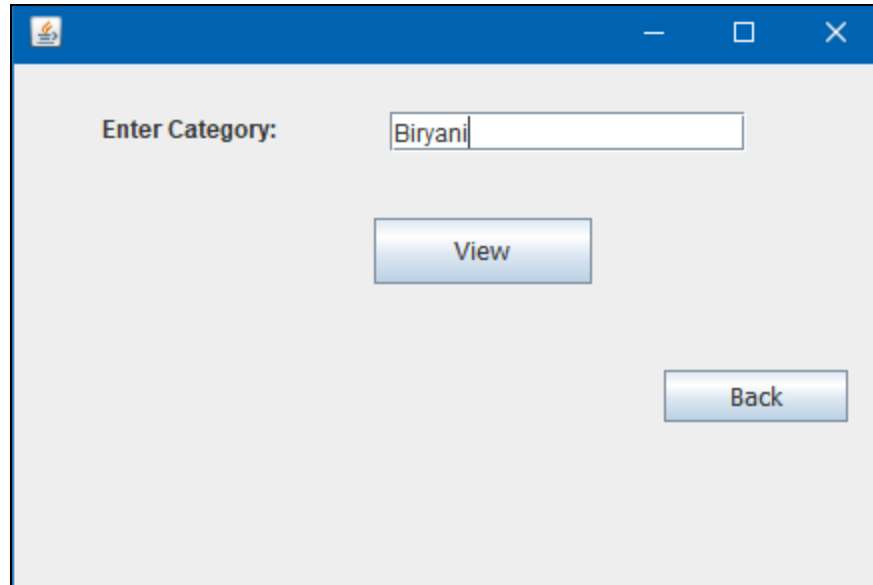
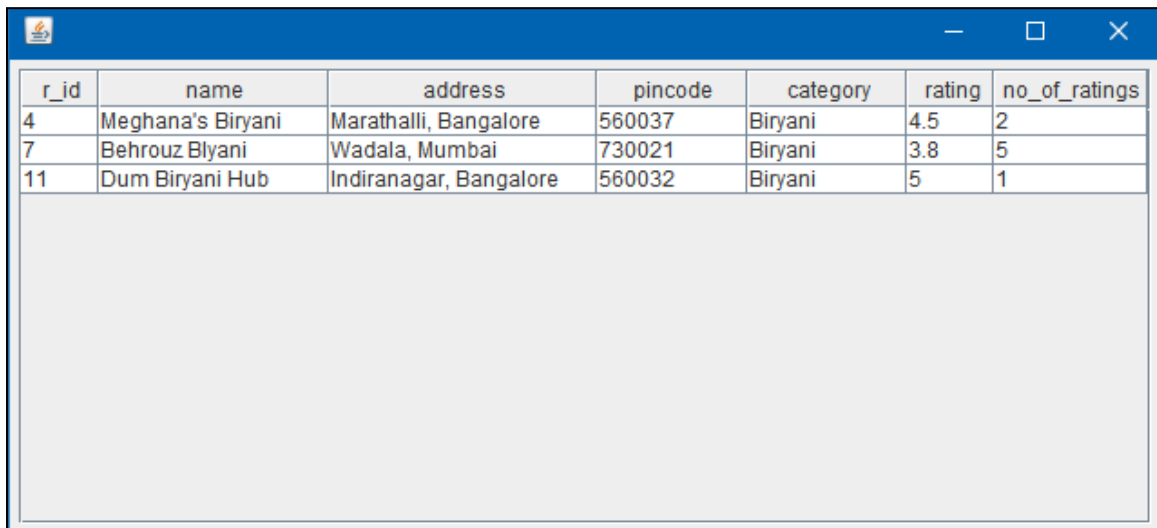


Figure 5.12: Restaurant View options



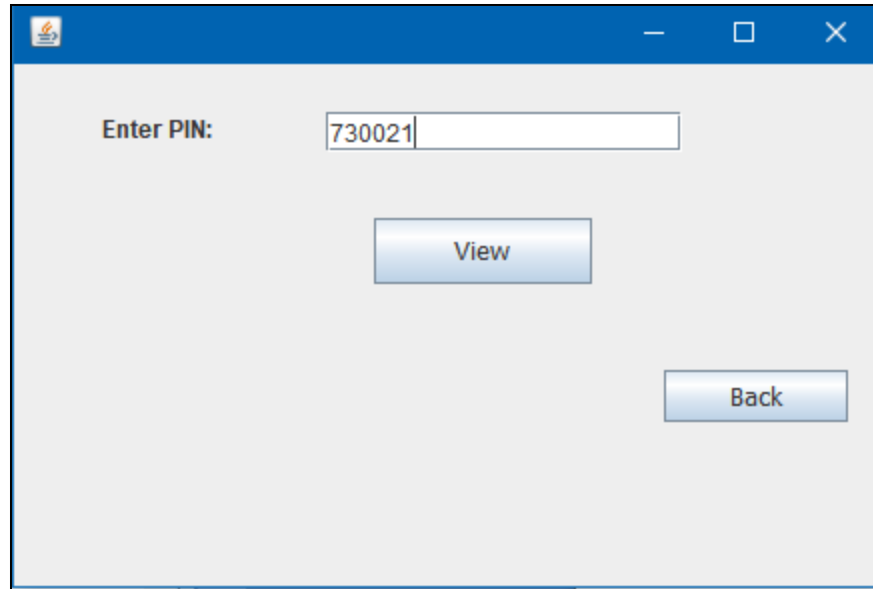
Enter Category:

Figure 5.13: Filter by Category



r_id	name	address	pincode	category	rating	no_of_ratings
4	Meghana's Biryani	Marathalli, Bangalore	560037	Biryani	4.5	2
7	Behrouz Blyani	Wadala, Mumbai	730021	Biryani	3.8	5
11	Dum Biryani Hub	Indiranagar, Bangalore	560032	Biryani	5	1

Figure 5.14: Result of Filter by Category

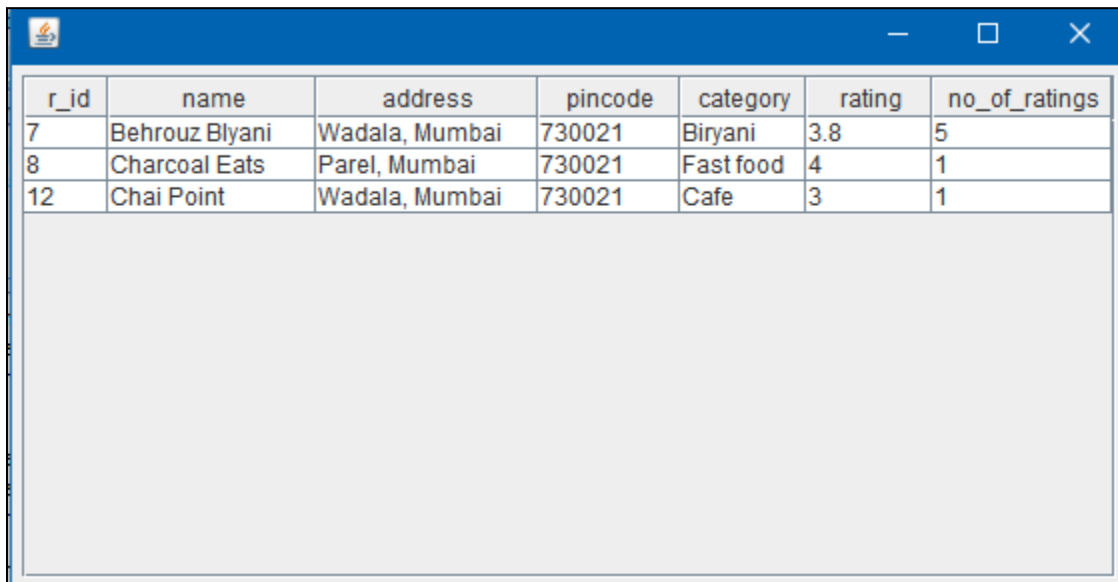


Enter PIN: 730021

View

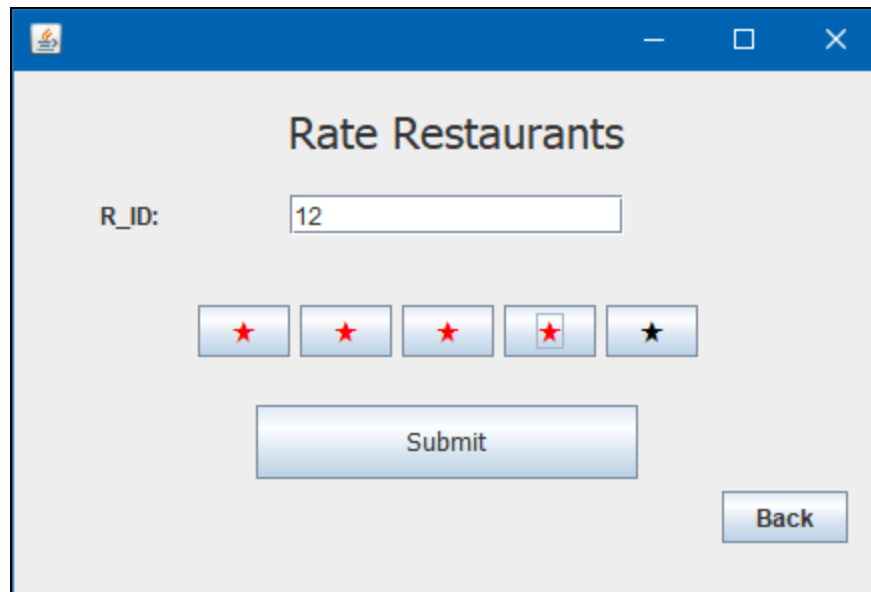
Back

Figure 5.15: Filter by Region



r_id	name	address	pincode	category	rating	no_of_ratings
7	Behrouz Blyani	Wadala, Mumbai	730021	Biryani	3.8	5
8	Charcoal Eats	Parel, Mumbai	730021	Fast food	4	1
12	Chai Point	Wadala, Mumbai	730021	Cafe	3	1

Figure 5.16: Result of Filter by Region



The screenshot shows a window titled "Rate Restaurants" with a blue header bar. Inside, there is a label "R\_ID:" followed by a text input field containing the number "12". Below the input field, there are five star buttons in a row. The first four stars are red, and the fifth is black. Below the stars is a large "Submit" button and a smaller "Back" button in the bottom right corner.

Figure 5.17: Rate Restaurants



This screenshot shows the same "Rate Restaurants" window as Figure 5.17, but with a modal message box overlaid in the center. The message box has a blue header "Message" and a close button (X). It contains an information icon (i) and the text "Thank you for rating!". Below the text is an "OK" button. The "Back" button from the original window is still visible in the bottom right corner.

Figure 5.18: Rated Restaurants successfully prompt

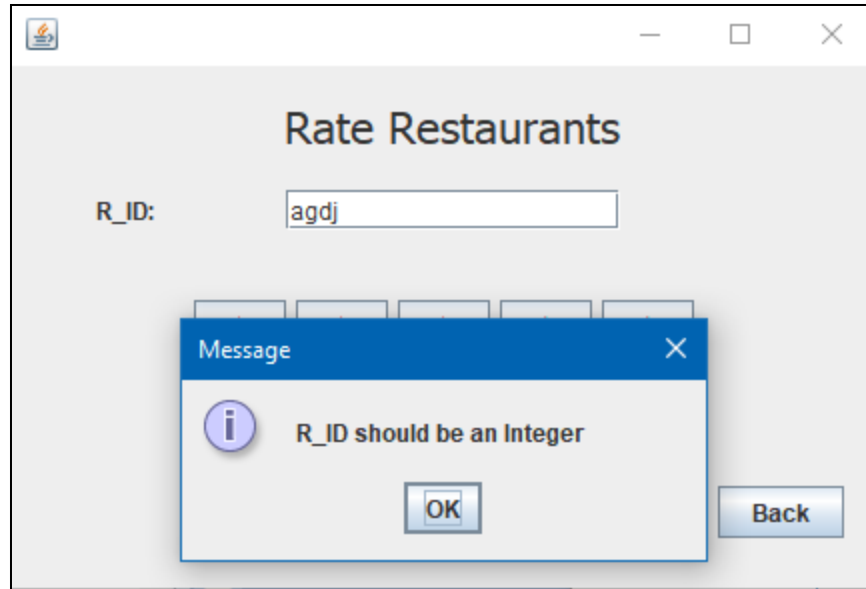


Figure 5.19: Invalid characters prompt

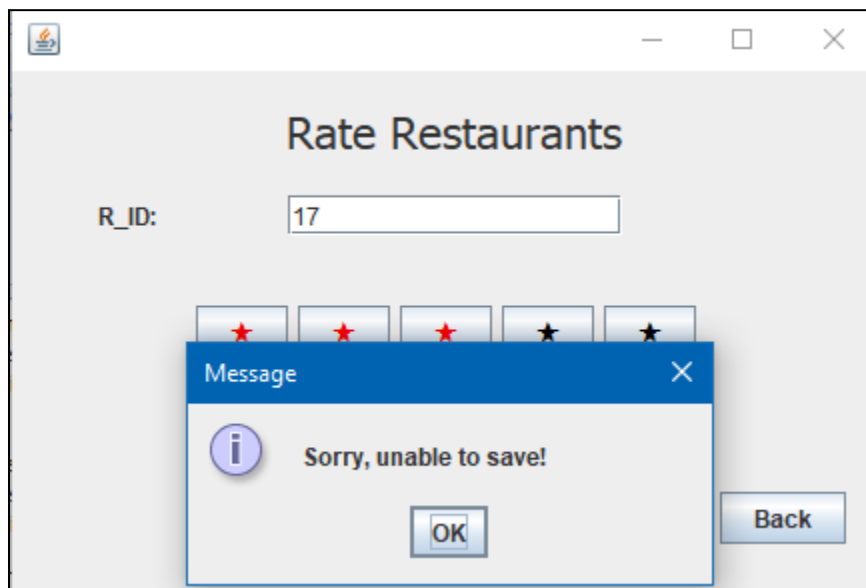


Figure 5.20: Invalid Restaurant ID prompt

## CHAPTER 6

### CONCLUSION AND FUTURE SCOPE

- This project has more scope in the future and can be integrated further.
- This project is successfully implemented with all the features mentioned earlier.
- Deployment of this application will help the user to reduce the unnecessary wastage of time in going and searching the restaurant manually.
- Therefore, we are successfully able to reach the goals and target of the project.

#### 6.1 FUTURE SCOPE

In a nutshell it can be summarized that the future scope of the project circles around maintaining information regarding:

- We can create a mobile application.
- We can give for advance front end facilities.
- We can host the platform on online servers to make it accessible worldwide.
- Create a master and a slave database structure to reduce the overload of the database queries.
- Implement the backup mechanism for taking backup of codebase and database on regular basis on different servers.



## CHAPTER 7

### REFERENCES

- [1] Java™: The Complete Reference, Seventh Edition. Herbert Schildt.
- [2] <https://www.javatpoint.com/>
- [3] <https://www.tutorialspoint.com/java/>
- [4] <https://www.w3schools.in>
- [5] <https://www.guru99.com/java-tutorial.html>
- [6] <https://www.guru99.com/java-tutorial.html>
- [7] <https://docs.oracle.com/javase/tutorial/>
- [8] <https://www.udemy.com/java-tutorial/>
- [9] <https://www.journaldev.com>