OBJECT ORIENTED PROGRAMMING LAB

(3+1 Credit Hours)

CSL-210

COLLEGE ADMISSION SYSTEM
Project Report

Couse Instructor: Sameena Javed
Lab Instructor: Hafsa Munawar

Group Members:

Enrollment	Name
02-134222-003	SUHA SHAHID
02-134222-010	MOHAMMAD SOBAN
02-134222-048	SAKEENA YAMIN
02-134222-086	MOHAMMAD SAAD KHAN

DEPARTMENT OF COMPUTER SCIENCE

BAHRIA UNIVERSITY, KARACHI, PAKISTAN

Table of Content:

Abstract:

The college admission process is a critical stage that shapes the future of student. However, it often becomes a complex and time-consuming task for both applicants and institutions. This report presents a detailed overview of a Java-based college admission system designed to streamline and automate the admission process, ultimately leading to improved efficiency and transparency.

Key Points:

1. Dealing with Complex Admission Process:

The system involves easy to use GUIs which will help the students and the admins to easily enter their details and view the details with just few clicks.

2. Requirement Analysis:

The required data will be analysed and the results plus the colleges will be shown accordingly making the work easier for both the admin and the students.

3. System Design:

The system design phase involves creating a conceptual model of the system's architecture, defining data structures, and designing the user interface. Emphasis is placed on scalability, modularity, and security to ensure a robust and reliable system.

4. Database Design:

An essential aspect of the system is the design of the database, which accurately represents the data entities involved in the admission process. Proper normalization principles and data integrity constraints are applied to maintain data consistency and reliability.

5. User Interface Development:

The user interface plays a crucial role in providing a user-friendly experience. The system employs Java's Swing and JavaFX libraries to develop intuitive forms for application submission, displaying application status, and providing necessary instructions to applicants.

6. Selection Criteria:

The system incorporates predefined selection criteria to evaluate and shortlist candidates. These criteria may include academic performance.

7. Implementation:

The system is implemented using Java as the primary programming language due to its versatility, platform independence, and extensive libraries. The development environment is set up, the database is implemented, and the user interface is developed using suitable Java libraries.

8. Testing:

Various testing methods are conducted to ensure the reliability and functionality of the system. Errors and bugs are identified to make the system as efficient and reliable as possible.

9. Benefits:

The Java-based college admission system offers several benefits, such as automation of the admission process, improved efficiency, reduced errors, and enhanced transparency. It simplifies the application submission process by providing a user-friendly experience to applicants and admission authorities.

Introduction:

Background information:

This system aims to streamline and automate the admission process, thereby enhancing efficiency and transparency.

The traditional approach to college admissions involves handling a large volume of applications, evaluating candidate profiles, and making admission decisions based on predefined criteria. However, this manual process is prone to delays, errors, and inconsistencies. Additionally, it places a significant burden on administrative staff, leading to inefficiencies and a lack of transparency in the process.

Problem Statement:

The proposed Java-based college admission system offers a solution to these problems. By leveraging the power of technology, the system automates various aspects of the admission process, making it more efficient and reliable. This automation allows for the seamless flow of information, reduces manual effort, and enables faster decision-making.

Objectives and Scope:

The key objective of the system is to simplify the application and selection process for prospective students. It achieves this by providing an intuitive and user-friendly interface for applicants to submit their applications online.

Furthermore, the introduction of the system ensures transparency and fairness in the admission process. The predefined selection criteria are consistently applied to evaluate candidates, eliminating biases and providing a level playing field. By automating the process, the system reduces human errors and ensures the accuracy and integrity of data.

The implementation of the system is based on a comprehensive methodology that includes requirement analysis, system design, database design, user interface development, and the establishment of an efficient application workflow. Java is chosen as the primary programming language for its versatility and extensive libraries, enabling robust and scalable system development.

Methodology:

1. **NetBeans 8.2**: NetBeans is an open-source IDE that provides a comprehensive development environment for Java applications. It offers features like code editing, debugging, and project management, making it suitable for Java-based projects.

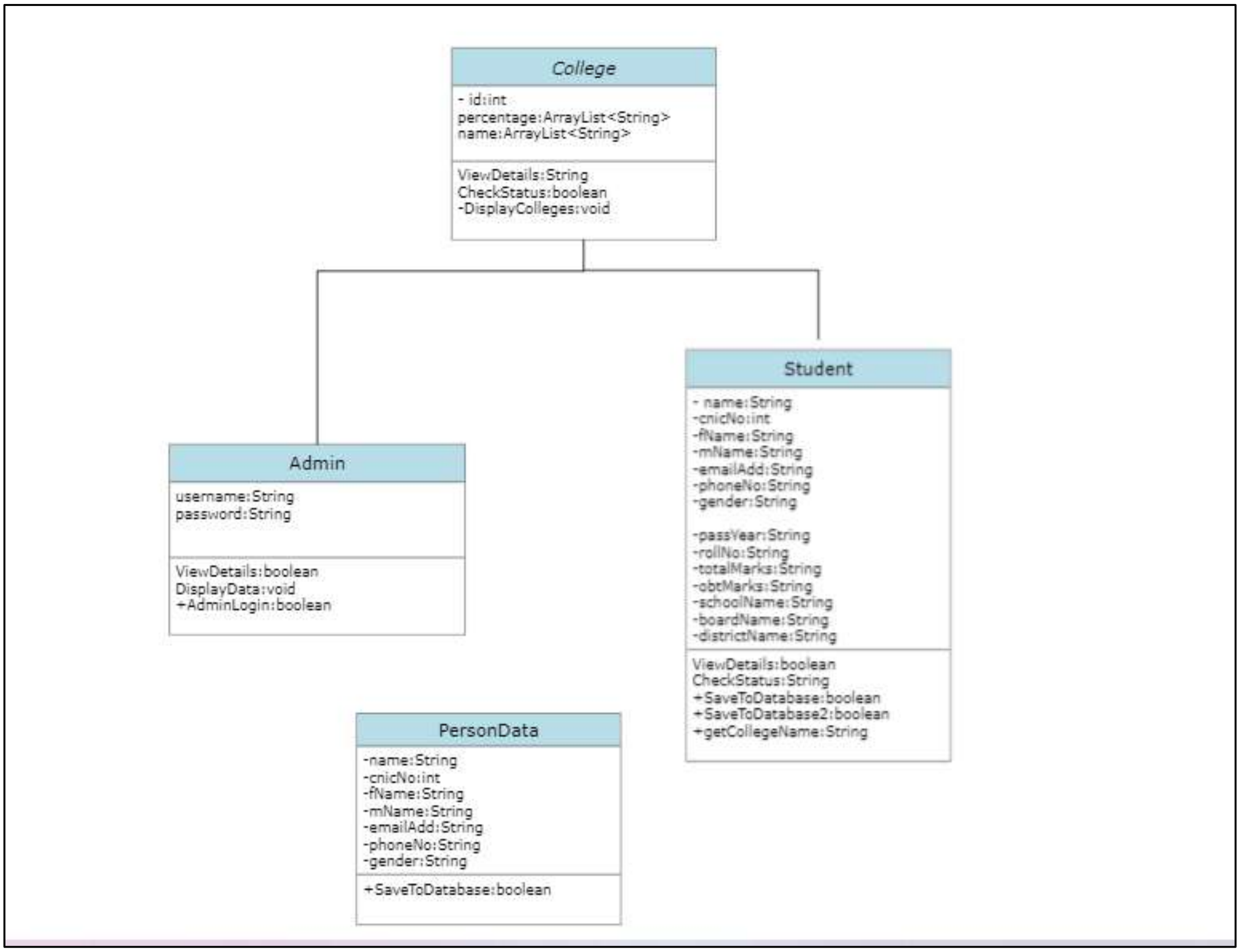
2. **Java**: The project was developed using the Java programming language, which offers strong object-oriented capabilities and a robust ecosystem for building enterprise-level applications.

3. **Object-Oriented Programming (OOP)**: The project follows the principles of object-oriented programming, which focuses on encapsulating data and behaviour within objects. This approach allows for modular and reusable code, promoting code organization and maintainability.

4. **User Interface and GUIs**: The user interface plays a crucial role in providing a user-friendly experience. The system should have intuitive forms for application submission, displaying application status, and providing necessary instructions to the applicants. Java's Swing or JavaFX libraries can be used for developing the graphical user interface.

5. **Database**: The project utilizes a database for storing and retrieving data related to garments, users, and orders. The specific database management system used may vary based on the project requirements and preferences, such as MySQL, PostgreSQL, or SQLite.

UML:


College
- id:int
percentage:ArrayList<String>
name:ArrayList<String>
ViewDetails:String
CheckStatus:boolean
-DisplayColleges:void
Admin
username:String
password:String
ViewDetails:boolean
DisplayData:void
+AdminLogin:boolean
Student
- name:String
-cnicNo:int
-fName:String
-mName:String
-emailAdd:String
-phoneNo:String
-gender:String
-passYear:String
-rollNo:String
-totalMarks:String
-obtMarks:String
-schoolName:String
-boardName:String
-districtName:String
ViewDetails:boolean
CheckStatus:String
+SaveToDatabase:boolean
+SaveToDatabase2:boolean
+getCollegeName:String
PersonData
-name:String
-cnicNo:int
-fName:String
-mName:String
-emailAdd:String
-phoneNo:String
-gender:String
+SaveToDatabase:boolean

OOP Principles and Concepts Utilized:

• **Inheritance**: Inheritance was utilized to establish an "is-a" relationship between classes, such as the inheritance relationship between the College, Admin, and Student classes. It allowed for code reuse and the specialization of behaviour.

• **Abstraction**: Abstraction was applied through the use of abstract classes and interfaces to define common behaviours and properties for related classes. For example, the abstract College class defined common attributes and methods shared by Admin and Student classes.

• **Polymorphism**: Polymorphism was achieved by treating objects of different classes as instances of a common parent class, allowing for flexibility and extensibility. For instance, a list of users could contain both Admin and Student objects, and common operations could be performed on all user objects.

• **Encapsulation**: Encapsulation was employed to encapsulate data and methods within classes, controlling access to class members and providing data protection. Getters and setters were used to access and modify the private attributes of classes.

Implementation:

1. **Admin Login**: The admins have their own set of login details which can lead to better security and protection of data of any student. The admin can log in anytime an perform various actions.

2. **Admin Assigning Colleges**: Admin can view the student details and assign the college to the student accordingly. The colleges criteria are set which can be viewed by the admin and this will help the admin to easily assign the most suitable college the students.

3. **Student Forms**: The students can enter their required details in 2 forms. They will enter personal details and the educational details. The details will then be saved and later can be viewed by both the admins and the students.

4. **Viewing Details**: This feature will help the students to later on check the details they have entered. The same can be done by the admin directly through their admin panel which is only accessible by the admins.

5. **Checking Status**: This is a unique feature which will help the student to check which college is assigned to them after the admin has assigned them a college. This feature will act as the final step in the admission process of the student.

6. **Database**: This whole procedure is handled through the MS access database. The admin data as well as the student data is directly saved in the database file. The details saved in the database is then accessible in the system or we can also save the data by making changes.

Relevant Code Snippets:

Getting data from GUI:

```
String n = name.getText();
    String f = fName.getText();
    String m = mName.getText();
    String e = emailAdd.getText();
    String p = phoneNo.getText();
    String c = cnicNo.getText();
    String d = gender.getSelectedItem().toString();
    int cnicNumber = Integer.parseInt(c);
        System.out.println(cnicNumber);
    try {
        if (n.isEmpty() || c.isEmpty() || f.isEmpty() || m.isEmpty() || e.isEmpty() || p.isEmpty() || d.isEmpty()) {
            jFormattedTextField1.setText("*Please enter all information.");
            return;
        }
        try {
         } catch (NumberFormatException ex) {
            jFormattedTextField1.setText("*Please enter valid information for phone number and CNIC number.");
            return;
        }
    } catch (NumberFormatException nfe) {
        jFormattedTextField1.setText("*Please enter valid information.");
        return;
    }
    obj = new Students(n,cnicNumber , f , m , e , p , d);
    boolean saved = obj.saveDataToDatabase2();

    if (saved) {
        System.out.println("Data inserted successfully!");
    } else {
        System.out.println("Failed to insert data!");
    }
    edu.setVisible(true);
    this.setVisible(false);
    }
```

Assigning Colleges:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String name = userName.getText();
        int id = Integer.valueOf(Id.getText());
        String clg = assClg.getText();
        Admin assign = new Admin(name,clg,id);
        assign.CheckStatus();

        JOptionPane.showMessageDialog(null, "COLLEGE ASSIGNED SUCCESFULLY!");
    }
```

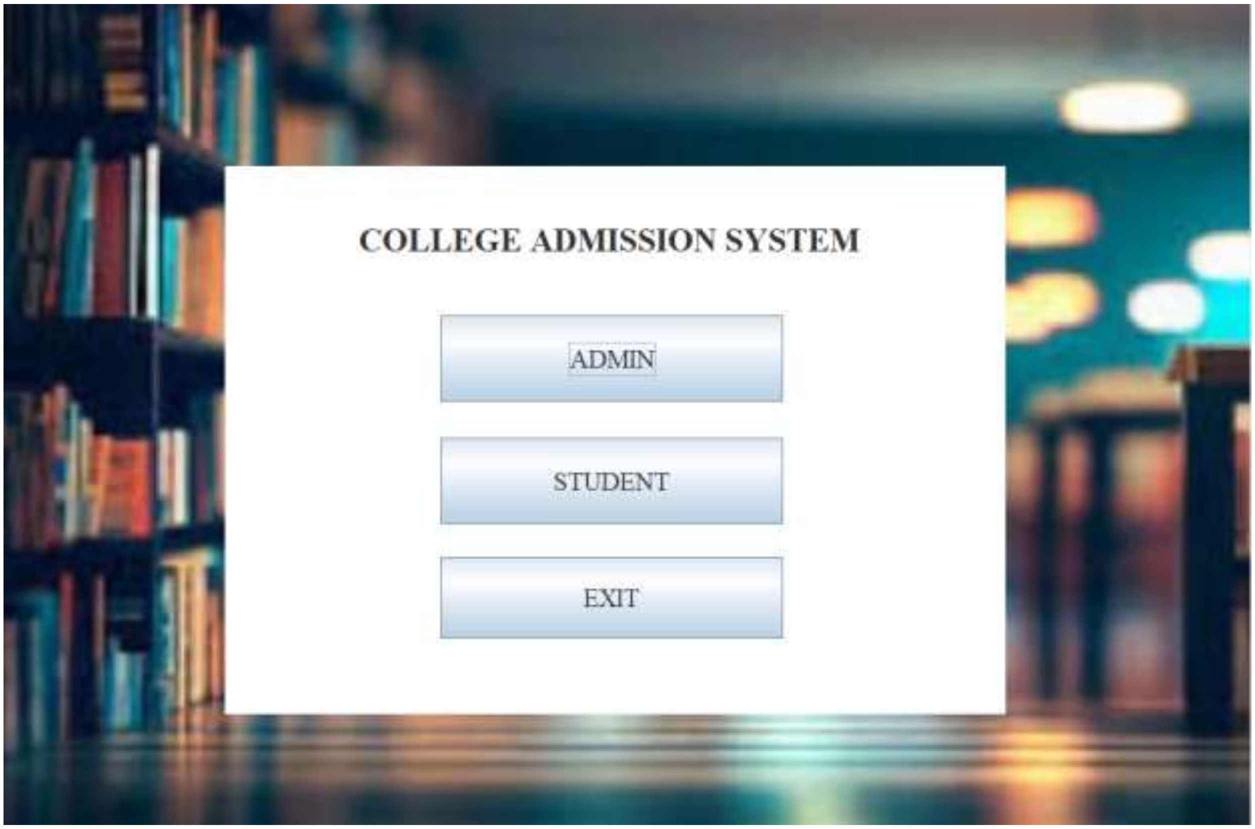
Checking Status:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String name = stdName.getText();
        int id =  Integer.valueOf(stdId.getText());
        Students obj = new Students(name,id);
        String AssignedCollege =  obj.CheckStatus();
         formatted.setText(AssignedCollege);
    }
```

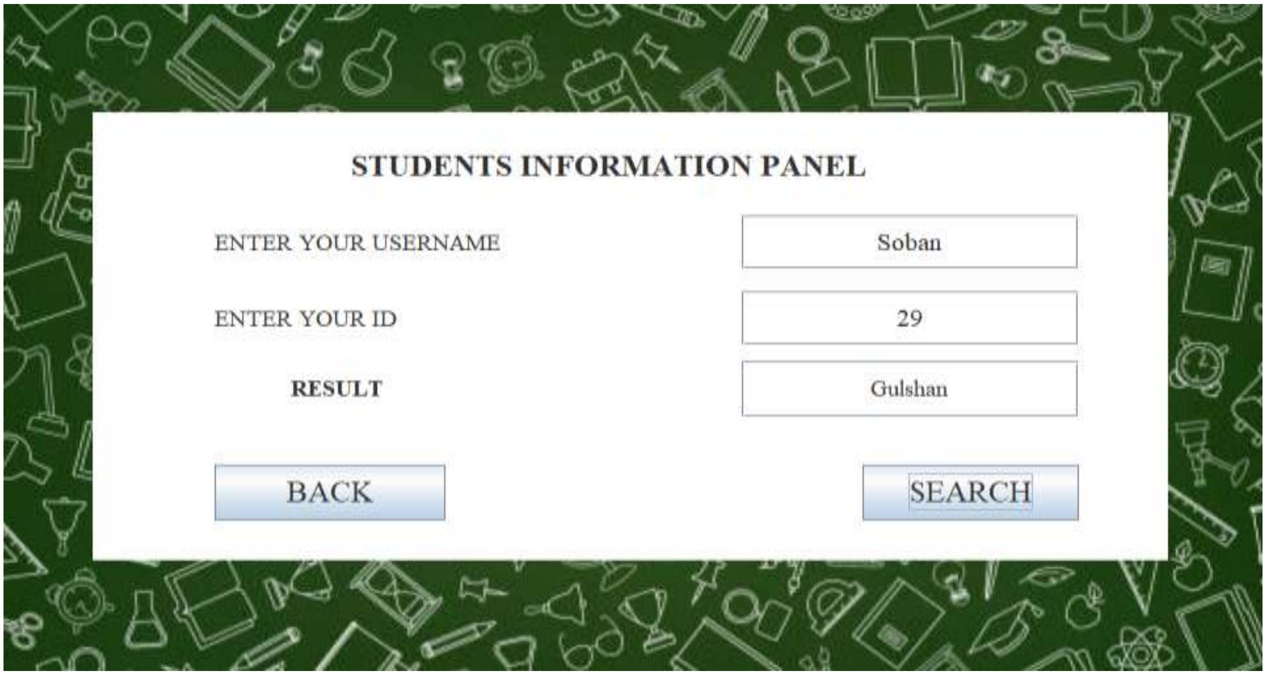
Abstraction:

```
public abstract class  College {
    private int id;
//    ArrayList<String> seats;
    ArrayList<String>percentage;
    ArrayList<String> name;
 public College() {
//        seats = new ArrayList<>();
        percentage = new ArrayList<>();
        name = new ArrayList<>();
    }
   abstract boolean ViewDetails ();
   abstract String CheckStatus();
   private void Displaycolleges() {
    DBConnector dbConnector = new DBConnector();
    String query = "SELECT CollegeName,ReqPercentage FROM College";
    ResultSet resultSet = dbConnector.RunSelect(query);
    try {
        while (resultSet.next()) {
            String collegeName = resultSet.getString("CollegeName");
            String collegePercentage = resultSet.getString("ReqPercentage");
            percentage.add(collegePercentage);
            name.add(collegeName);
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
}
    public ArrayList<String> getCollegeNames() {
    Displaycolleges();
        return name;
}
    public ArrayList<String> getCollegePercentage() {
    Displaycolleges();
        return percentage;
}
}
```

Relevant Output Snippets:



(MAIN PAGE)

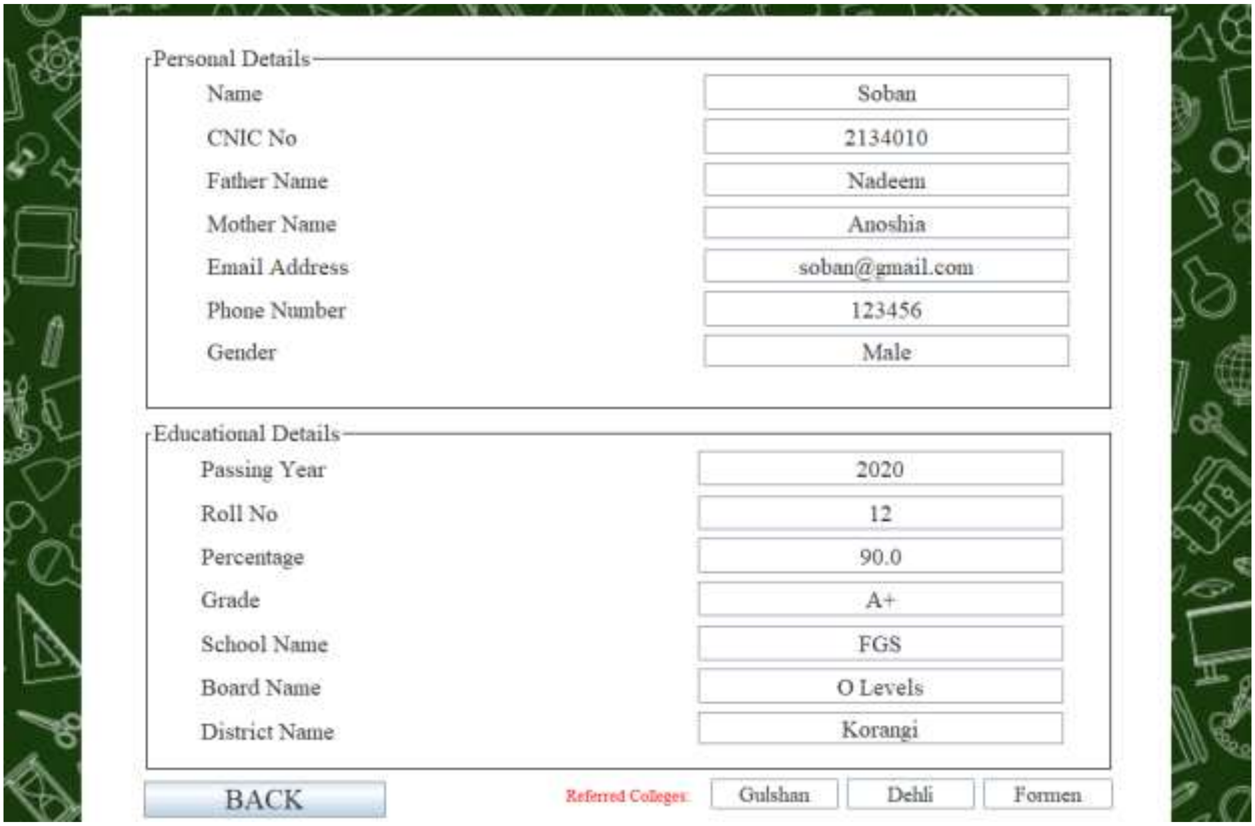

STUDENTS INFORMATION PANEL

ENTER YOUR USERNAME: Soban

ENTER YOUR ID: 29

RESULT: Gulshan

(CHECKING STATUS)



Personal Details

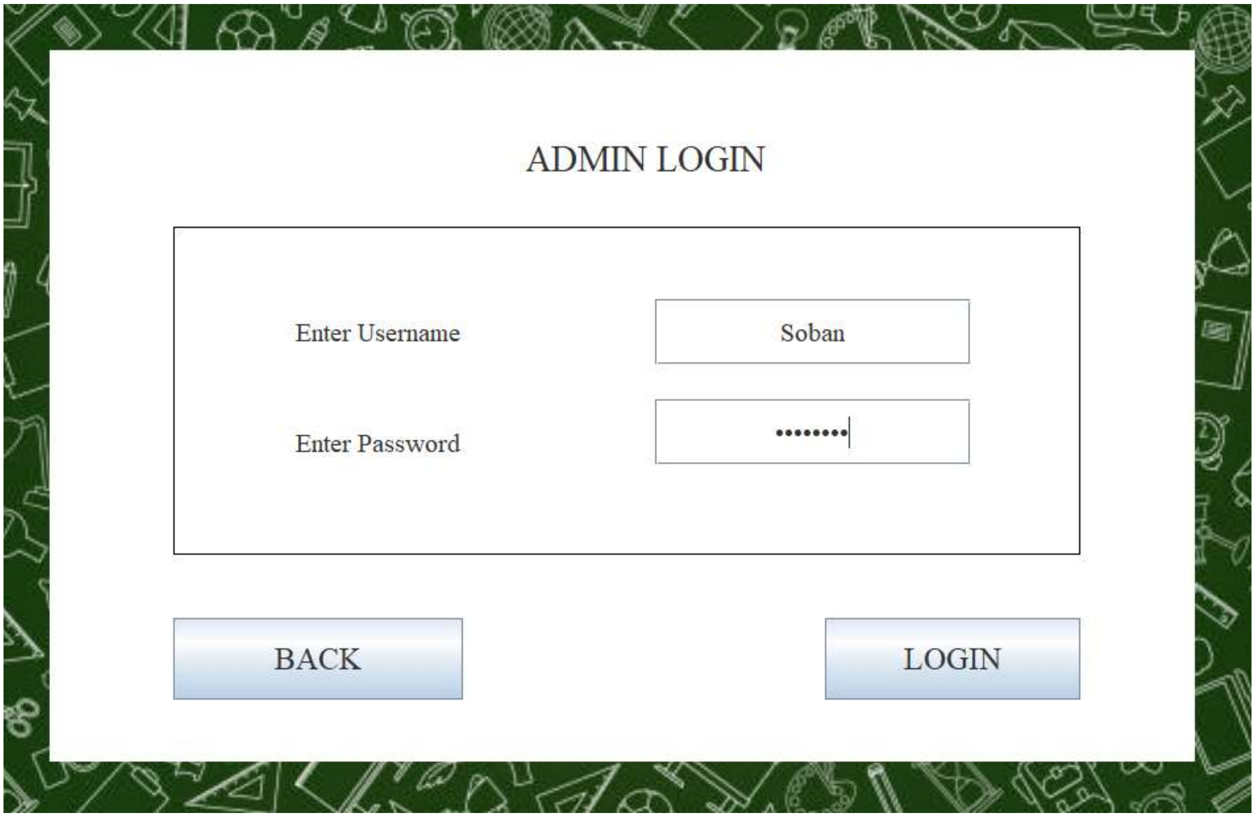
Name	Soban
CNIC No	2134010
Father Name	Nadeem
Mother Name	Anoshia
Email Address	soban@gmail.com
Phone Number	123456
Gender	Male

Educational Details

Passing Year	2020
Roll No	12
Percentage	90.0
Grade	A+
School Name	FGS
Board Name	O Levels
District Name	Korangi

Referred Colleges: Gulshan Dehli Formen

(Student Details)

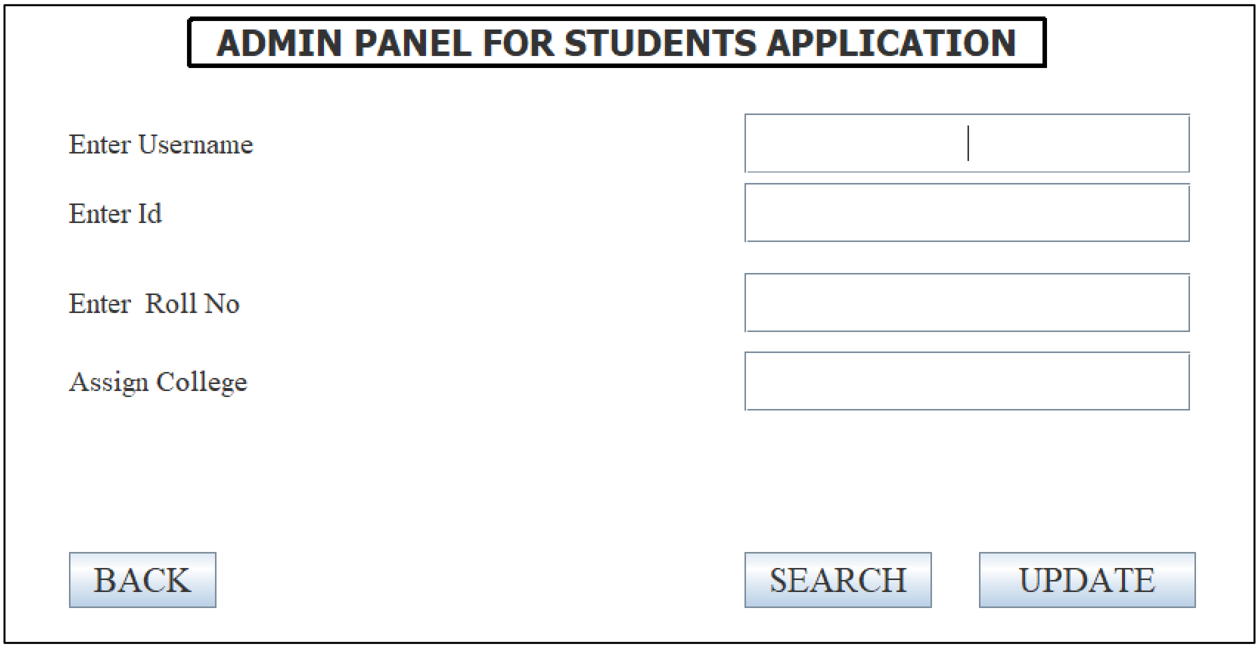


ADMIN LOGIN

Enter Username: Soban

Enter Password: ••••••••

(Admin Login)



ADMIN PANEL FOR STUDENTS APPLICATION

Enter Username

Enter Id

Enter Roll No

Assign College

(Admin Panel)

Conclusion:

The college admission system offers numerous benefits, including automation of the admission process, improved efficiency, reduced errors, and enhanced transparency. By leveraging the power of Java and database technologies, the system simplifies application submission and merit-based selection. It streamlines the workflow and provides a user-friendly experience to both applicants and admission authorities. The system's successful implementation can significantly enhance the overall admission process, making it smoother and more effective for all stakeholders involved.

Overall, this system has successfully achieved its objectives of providing a comprehensive and user-friendly platform for the process of admissions of students in different colleges. It has streamlined processes, enhanced user experience, facilitated accurate management, and effectively utilized OOP principles and database integration.