1. INTRODUCTION

This section gives a scope description and overview of everything included this Project Report. Also, the purpose for this document is described and system overview along with goal and vision are listed.

1.1 Purpose

The purpose of this document is to give a detailed description of Fee Management Project. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with Window based application. Which is made java net beans. This document is primarily intended to anyone who wants to get an overview of how Fee Management works, its outcomes and possible usages in future.

1.2 System Overview

In the existing system, most of the records are maintained on paper. It becomes very inconvenient to modify the data. In the existing system, here is a possibility that the same data in different registers may have different values which means the entries of the same data do not match. This inconsistent state does not supply the concrete information which poses a problem in the case information related to particular search record.

Our project is very useful. User is no longer required to check his register in search of records, as now it can be searched over the software by choosing some options. The user need not to type in most of the information. He/she is just required to enter the desired options. On the whole it liberates the user from keeping lengthy manual records. In a nutshell, it abates the work load of an organization.

In today’s world, no one likes to perform calculations on calculator or manually when computer is there. Every one wants his/her work to be done by computer automatically and displaying the result for further manipulations. So this project is about providing convenience regarding fee management system.

1.3 Problem Statement

- Which is user friendly.
- Which will restrict the user from accessing other user’s data.
- Which will help user in viewing his data and privileges.
- Which will help the administrator and accountant to handle all the changes.
- Accountant will change all the specific details of student.
- All the details will be taken by the student than after fee receipt will be generated by the accountant.

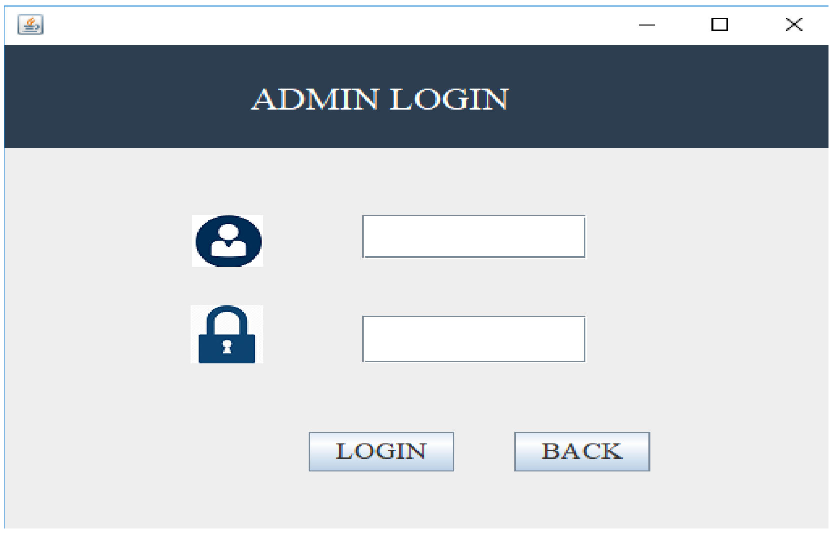
2.REQUIREMENTS SPECIFICATIONS

2.1 User characteristics

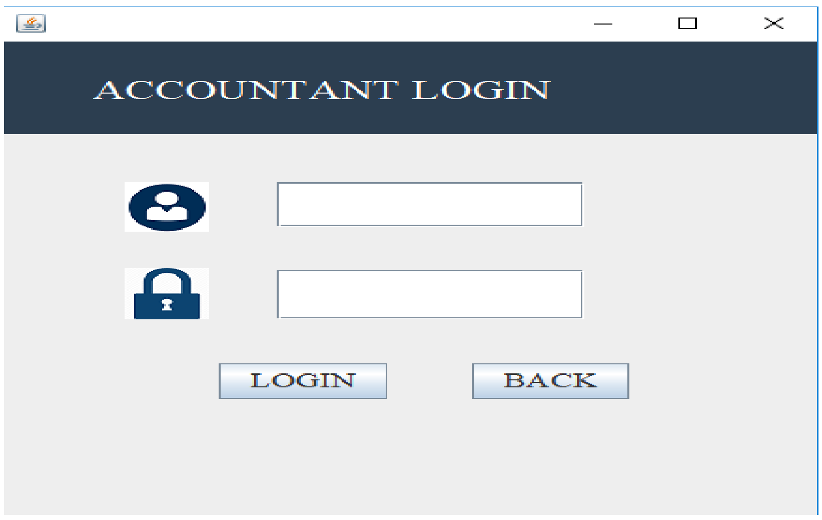
There are two types of users that interact with the system:-

- Administrator.
- Accountant.

Snapshot of Administrator interface.



Snapshot of accountant interface.



2.2 Functional Requirement

User 1:- Administrator.

Actor: Administrator.

Input:- Feed Text as Input.

Description: Admin has access to the system through GUI based form which is design in java swing using netbeans IDE.

User 2:- Accountant.

Actor: Accountant**.**

Input:- Feed Text as Input**.**

Description: Accountant has access to the system through GUI based form which is design in java swing using net-beans IDE. Which is adding student information, Update students information, delete students information. One of the main module of accountant is to generate a fee receipt of students.

2.3 Module Description

◆ Administrator Login.

- Administrator Section.
- Add Accountant.
- View Accountant.
- Update Accountant.
- Delete Accountant.
- Back.
- Logout.

- ◆ Accountant Login.
  - View Students.
  - Update Students.
  - Delete Student.
  - Back.
  - Logout.
- ◆ Fee Receipt.
  - Search Students.
  - Generate Receipt.
  - Reset Receipt.
  - Print Receipt.
  - Back.

2.4 Technology used.

Front-end

- ■ Net-beans IDE 8.0.2
- ■ Java

Back-end

- ■ MYSQL

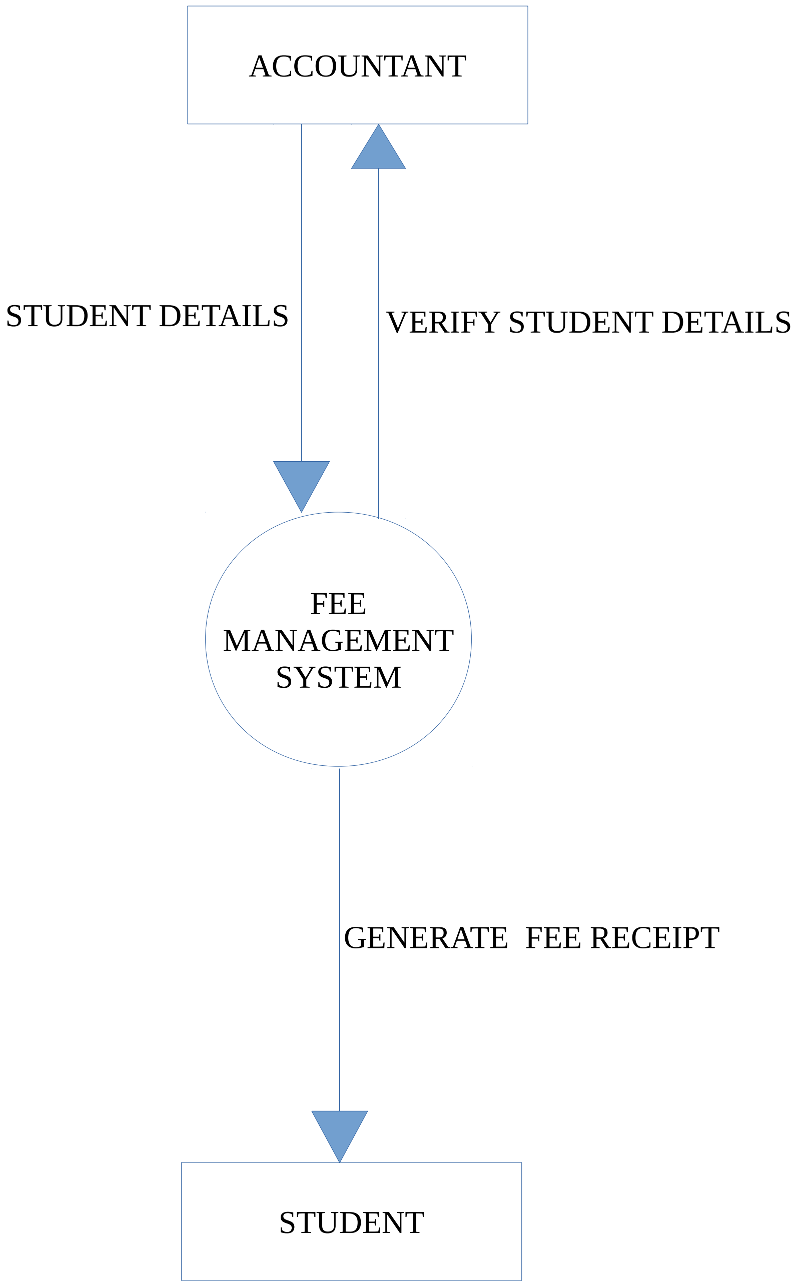
2.4 Hardware Requirement.

To access a Fee Management System, its only need a PC/Laptop and printer for generating the existing fee receipt of students.

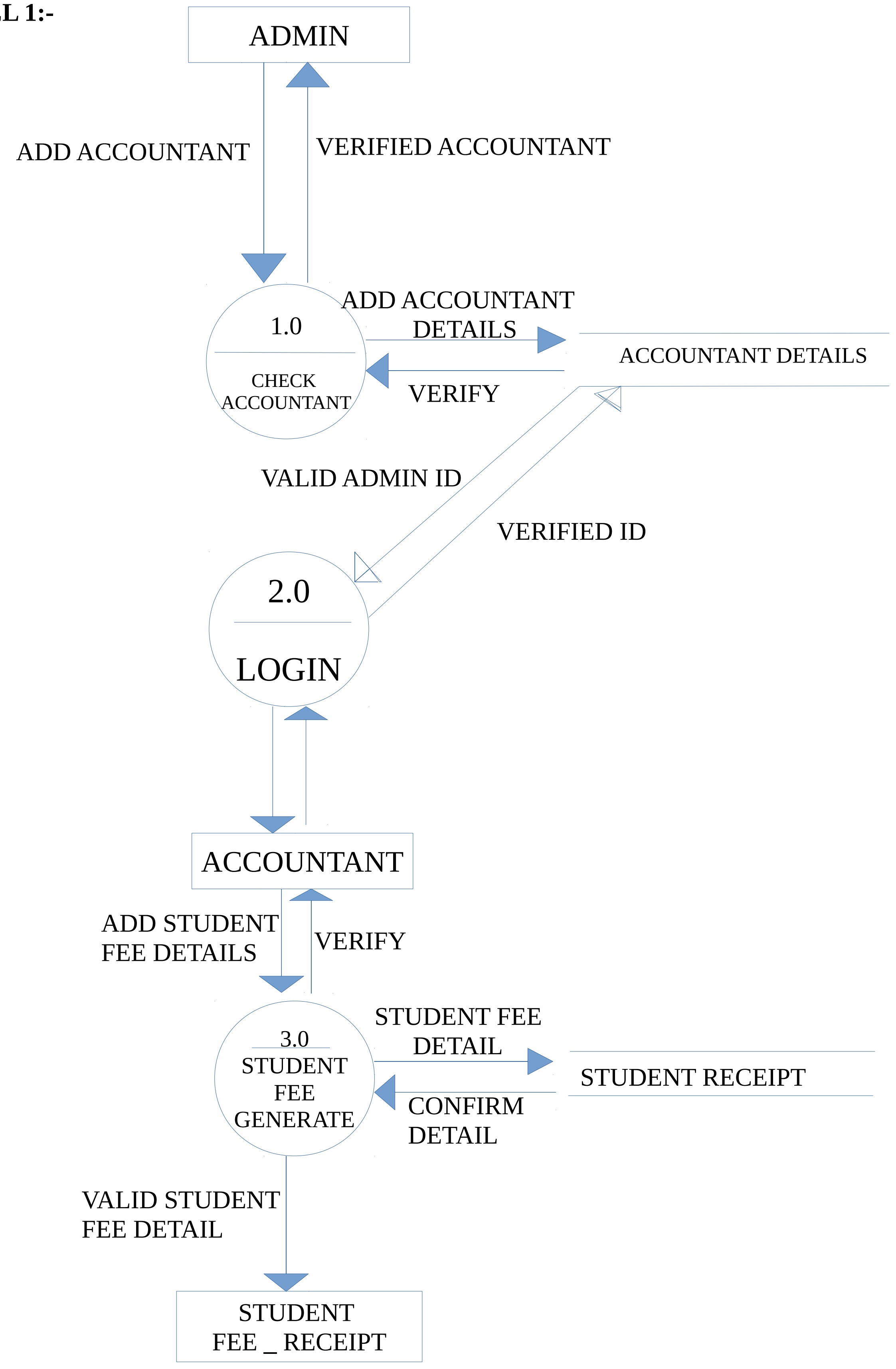
3. DESIGN

3.1 DFD:- FEE MANAGEMENT SYSTEM

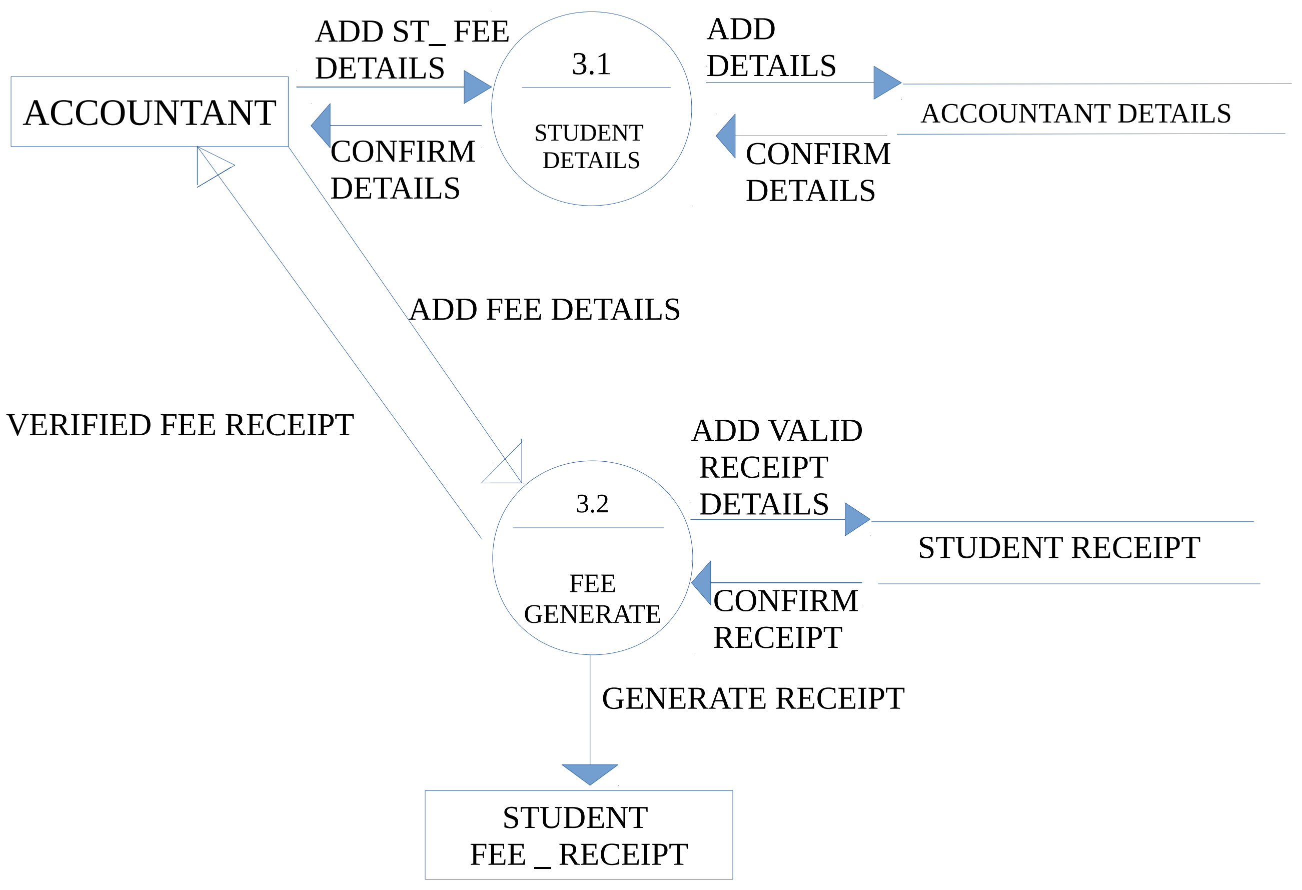
LEVEL 0:-



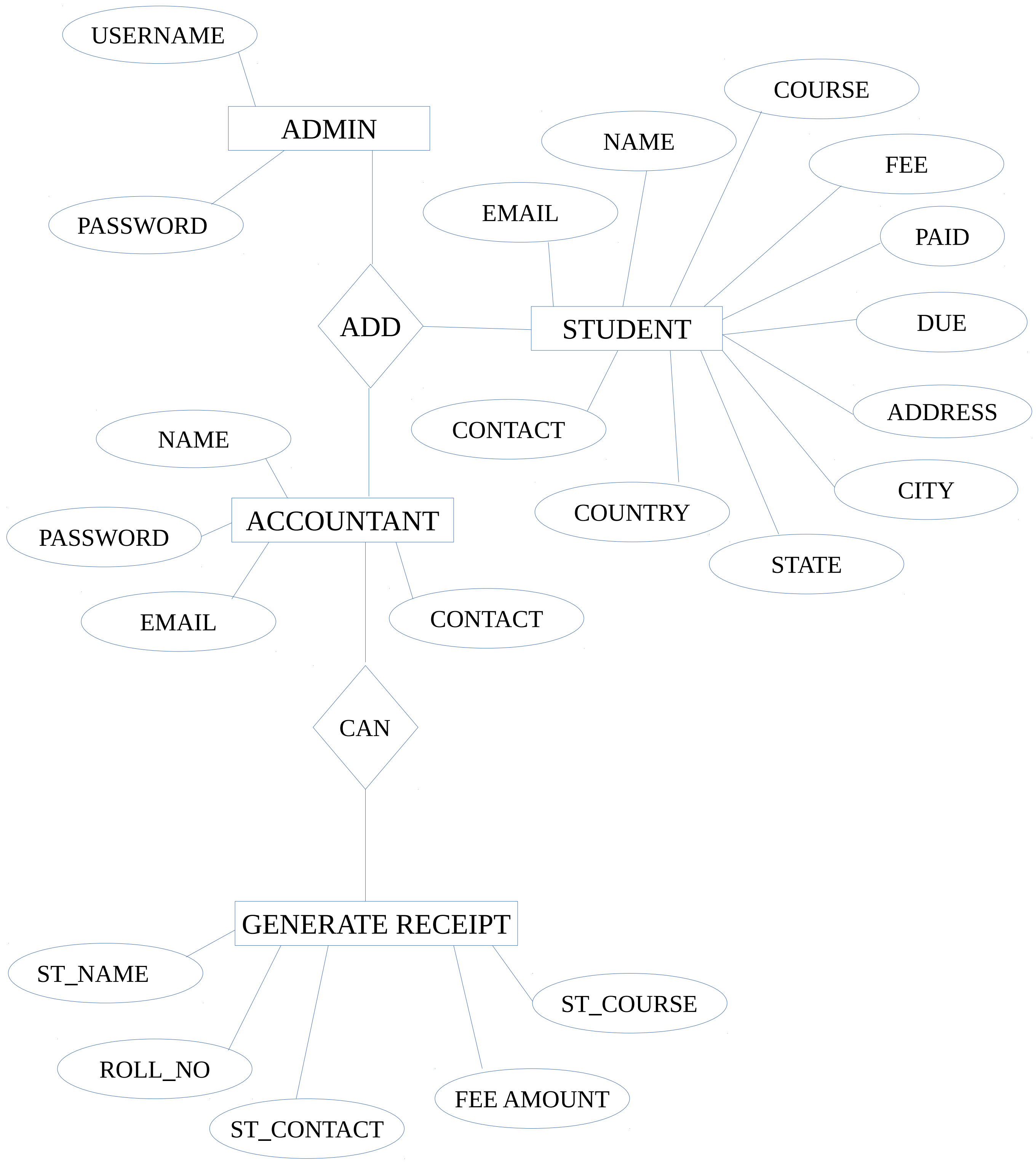
LEVEL 1:-


ADMIN
ADD ACCOUNTANT
VERIFIED ACCOUNTANT
1.0
CHECK ACCOUNTANT
ADD ACCOUNTANT DETAILS
VERIFY
ACCOUNTANT DETAILS
VALID ADMIN ID
VERIFIED ID
2.0
LOGIN
ACCOUNTANT
ADD STUDENT FEE DETAILS
VERIFY
3.0
STUDENT FEE GENERATE
STUDENT FEE DETAIL
CONFIRM DETAIL
STUDENT RECEIPT
VALID STUDENT FEE DETAIL
STUDENT FEE _ RECEIPT

LEVEL 2:-


ACCOUNTANT
ADD ST_ FEE DETAILS
CONFIRM DETAILS
3.1
STUDENT DETAILS
ADD DETAILS
CONFIRM DETAILS
ACCOUNTANT DETAILS
ADD FEE DETAILS
VERIFIED FEE RECEIPT
3.2
FEE GENERATE
ADD VALID RECEIPT DETAILS
CONFIRM RECEIPT
STUDENT RECEIPT
GENERATE RECEIPT
STUDENT FEE _ RECEIPT

3.2 ERD:- FEE MANAGEMENT SYSTEM



3.3 DATABASE DESIGN

Database used for this Window based application is Mysql. In this database there are four table used in fee management system that store all the details of the students which is stored by the accountant of this project.

Connectivity of database:- A **Database connection** is a facility in computer science that allows client software to talk to **database** server software, whether on the same machine or not. A **connection** is required to send commands and receive answers, usually in the form of a result set. Connections are a key concept in data-centric programming

Steps:-

1. Import JDBC Packages.

This is for making the JDBC API classes immediately available to the application program. The following import statement should be included in the program irrespective of the JDBC driver being used:

import java.sql.*;

Additionally, depending on the features being used, Oracle-supplied JDBC packages might need to be imported. For example, the following packages might need to be imported while using the Oracle extensions to JDBC such as using advanced data types such as BLOB, and so on.

import oracle.jdbc.driver.*;

import oracle.sql.*;

2. Load and Register the JDBC Driver.

This is for establishing a communication between the JDBC program and the Oracle database. This is done by using the static registerDriver() method of the DriverManager class of the JDBC API. The following line of code does this job:

DriverManager.registerDriver(new oracle.jdbc.driver.OracleDriver());

3. JDBC Driver Registration.

For the entire Java application, the JDBC driver is registered only once per each database that needs to be accessed. This is true even when there are multiple database connections to the same data server.

Alternatively, the forName() method of the java.lang.Class class can be used to load and register the JDBC driver:

Class.forName("oracle.jdbc.driver.OracleDriver");

4.Connecting to a Database

Once the required packages have been imported and the Oracle JDBC driver has been loaded and registered, a database connection must be established. This is done by using the getConnection() method of the DriverManager class. A call to this method creates an object instance of the java.sql.Connection class. The getConnection() requires three input parameters, namely, a connect string, a username, and a password. The connect string should specify the JDBC driver to be yes and the database instance to connect to

The getConnection() method is an overloaded method that takes

- Three parameters, one each for the URL, username, and password.
- Only one parameter for the database URL. In this case, the URL contains the username and password.

The following lines of code illustrate using the getConnection() method:

Connection conn = DriverManager.getConnection(URL, username, passwd);

Connection conn = DriverManager.getConnection(URL);

5.Querying the Database

Querying the database involves two steps: first, creating a statement object to perform a query, and second, executing the query and returning a resultset.

6.Creating a Statement Object

This is to instantiate objects that run the query against the database connected to. This is done by the createStatement() method of the conn Connection object created above. A call to this method creates an object instance of the Statement class. The following line of code illustrates this:

Statement sql_stmt = conn.createStatement();

7.Closing the ResultSet and Statement

Once the ResultSet and Statement objects have been used, they must be closed explicitly. This is done by calls to the close() method of the ResultSet and Statement classes. The following code illustrates this:

rset.close();

sql_stmt.close();

If not closed explicitly, there are two disadvantages:

- Memory leaks can occur
- Maximum Open cursors can be exceeded
- Closing the ResultSet and Statement objects frees the corresponding cursor in the database.

8.Closing the Connection

The last step is to close the database connection opened in the beginning after importing the packages and loading the JDBC drivers. This is done by a call to the close() method of the Connection class.

The following line of code does this:

conn.close();

3.4 CONNECTIVITY OF FEE MANAGEMENT DATABASE:-

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import javax.swing.JOptionPane;

 try {
         Class.forName("com.mysql.jdbc.Driver");
                                                          try (Connection con =
DriverManager.getConnection("jdbc:MySQL://localhost:3306/fee","root","")) {
                 String sql="select * from accountant_detail where Accountant_name=?
and Accountant_password=?";
             PreparedStatement pst=con.prepareStatement(sql);
             pst.setString(1,tf_Accountant_username.getText());
```

```
            pst.setString(2,tf_Accountant_password.getText());
            ResultSet rs=pst.executeQuery();
            if (rs.next()) {
               JOptionPane.showMessageDialog(null, "!!!!!Welcome Accountant!!!!");
               Accountant_section s1;
               s1 = new Accountant_section();
               s1.setVisible(true);
               setVisible(false);
            } else {
                 JOptionPane.showMessageDialog(null, "username and password Dont
Matched");
               tf_Accountant_username.setText("");
               tf_Accountant_password.setText("");
            }
            // TODO add your handling code here:
         }
      }
      catch(Exception e){
         JOptionPane.showMessageDialog(null,e);
      }
```

3.5 TABLE STRUCTURE:-

Features of admin module:-

- Admin itself have the admin detailed in admin table.
- Admin can add new accountant.

Features of accountant module:-

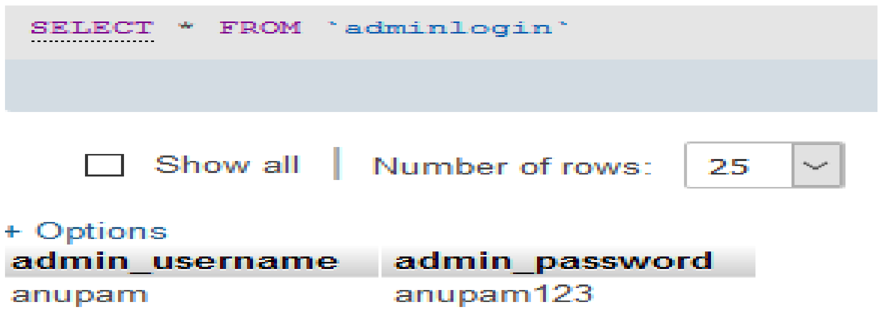
- Adding new student details.
- View students details.
- Update all the students details.
- Delete the students.
- Accountant can generate fee receipt that can automatically store in receipt table.

1. Table:- Admin

Fields:-

- Admin user-name.
- Admin Password.

Snapshot of Table:-



```
SELECT * FROM `adminlogin`
```

Show all | Number of rows: 25

+ Options

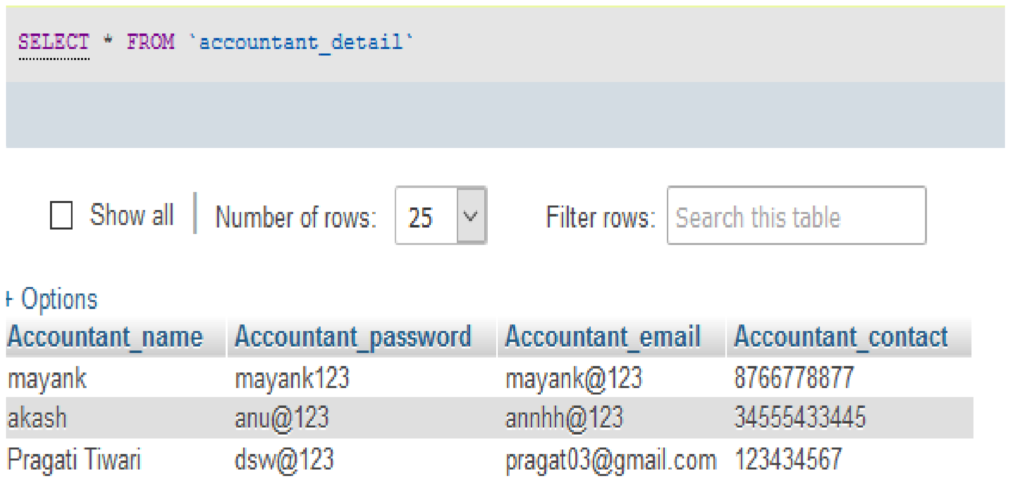
admin_username	admin_password
anupam	anupam123

2. Table:- Accountant.

Fields:-

- Accountant User-name.
- Accountant Password.
- Accountant E-mail.
- Accountant Contact.
-

Snapshot of Table:-



```
SELECT * FROM `accountant_detail`
```

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

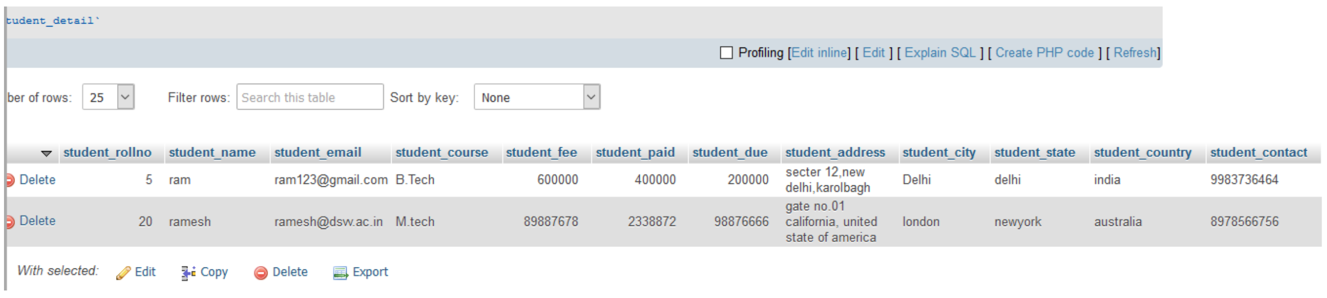
Accountant_name	Accountant_password	Accountant_email	Accountant_contact
mayank	mayank123	mayank@123	8766778877
akash	anu@123	annhh@123	34555433445
Pragati Tiwari	dsw@123	pragat03@gmail.com	123434567

3. Table:- Student Details.

Fields:-

- Student name.
- Student roll_no.
- Student Course.
- Student email.
- Student fee.
- Student paid.
- Student due.
- Student address.
- Student city.
- Student state.
- Student Country.
- Student contact.

Snapshot of Table:-



tudent_detail`

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

ber of rows: 25 Filter rows: Search this table Sort by key: None

	student_rollno	student_name	student_email	student_course	student_fee	student_paid	student_due	student_address	student_city	student_state	student_country	student_contact
Delete	5	ram	ram123@gmail.com	B.Tech	600000	400000	200000	secter 12,new delhi,karolbagh	Delhi	delhi	india	9983736464
Delete	20	ramesh	ramesh@dsw.ac.in	M.tech	89887678	2338872	98876666	gate no.01 california, united state of america	london	newyork	australia	8978566756

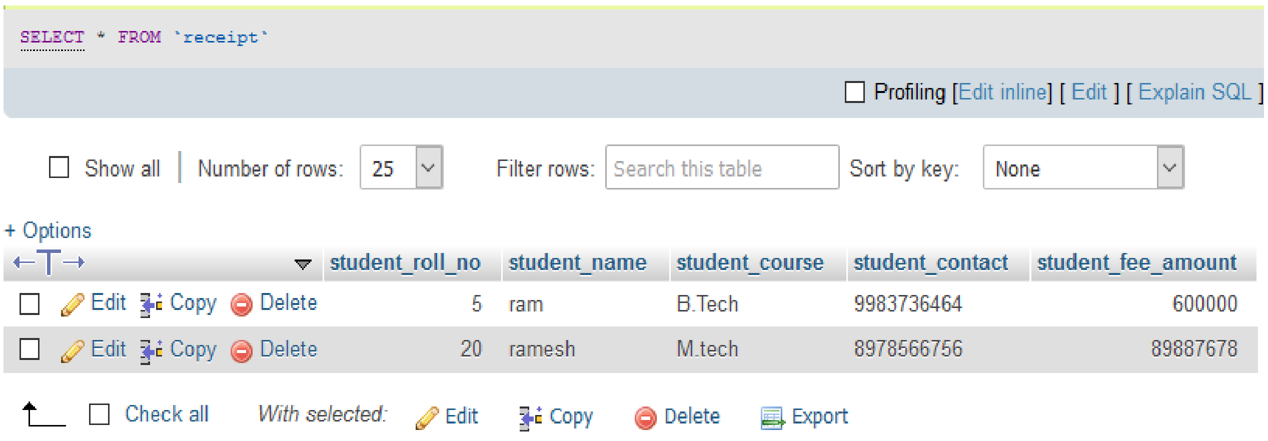
With selected: Edit Copy Delete Export

4. Table:- Receipt.

Fields:-

- Student name.
- Student roll_no.
- Student Course.
- Student Contact.
- Student fee amount.

Snapshot of Table:-



```
SELECT * FROM `receipt`
```

☐ Profiling [Edit inline] [Edit] [Explain SQL]

☐ Show all | Number of rows: 25 Filter rows: Search this table Sort by key: None

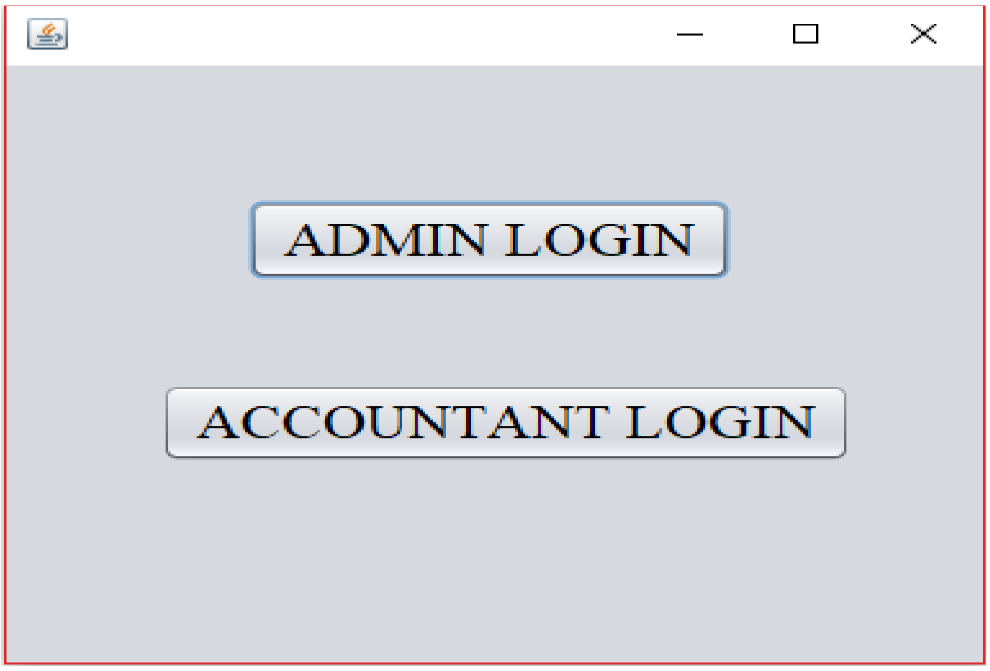
+ Options

	student_roll_no	student_name	student_course	student_contact	student_fee_amount
☐ Edit Copy Delete	5	ram	B.Tech	9983736464	600000
☐ Edit Copy Delete	20	ramesh	M.tech	8978566756	89887678

☐ Check all *With selected:* Edit Copy Delete Export

3.6 GUI DESIGN FOR FRONT-END

1. Administrator Login and Accountant Login Interface.

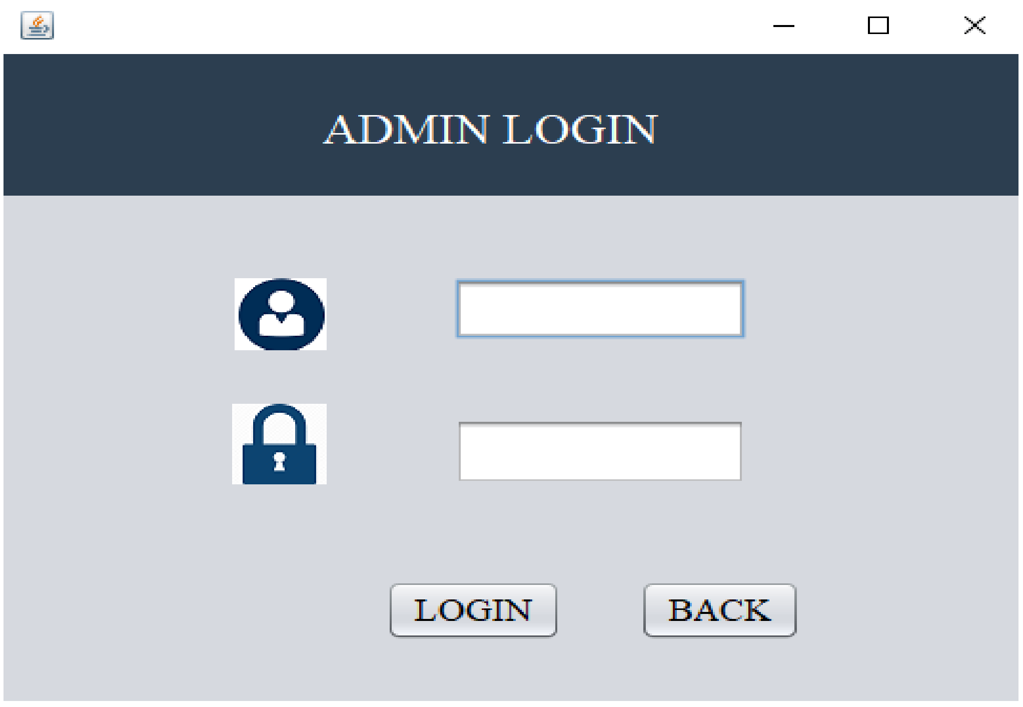


Snapshot detail:-

Admin Login:- Administrator can login into the system through admin button.

Accountant Login:- Accountant can login into the system through accountant button.

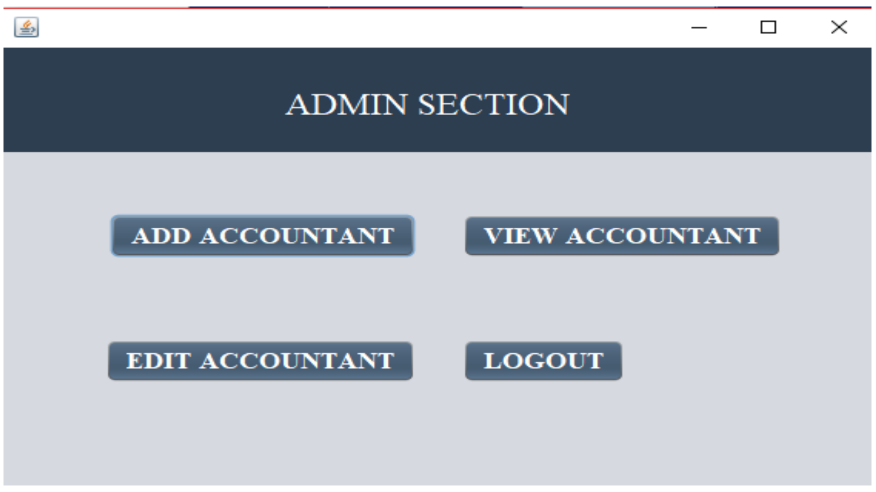
2. Administrator Login Interface.



Snapshot detail:-

Administrator Login:- Admin can enter the correct user-name and password to enter into the admin section. The admin can need only user-name and password detail.

3. Administrator Section Interface.



Snapshot detail:-

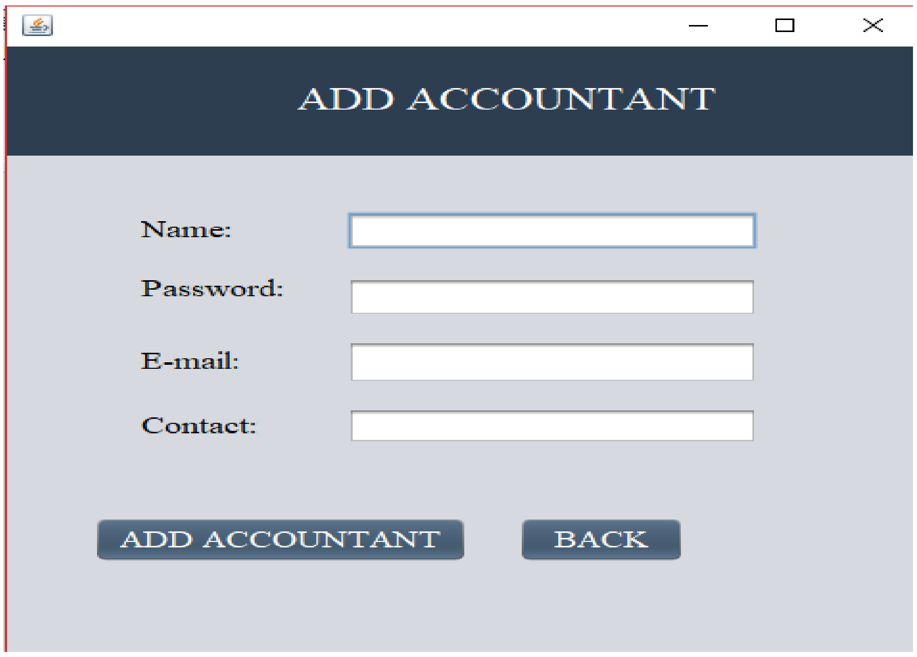
Add Accountant Button:- Administrator can add accountant for the system then accountant can manage all the details of the student.

View Accountant Button:- Administrator can view accountant details after adding accountant detail by administrator.

Edit Accountant Button:- After clicking the edit Accountant Button administrator can update the details of accountant that is already stored in the database. Also admin can delete the accountant permanently.

Administrator Logout Button:- Admin can logout by pressing the logout button.

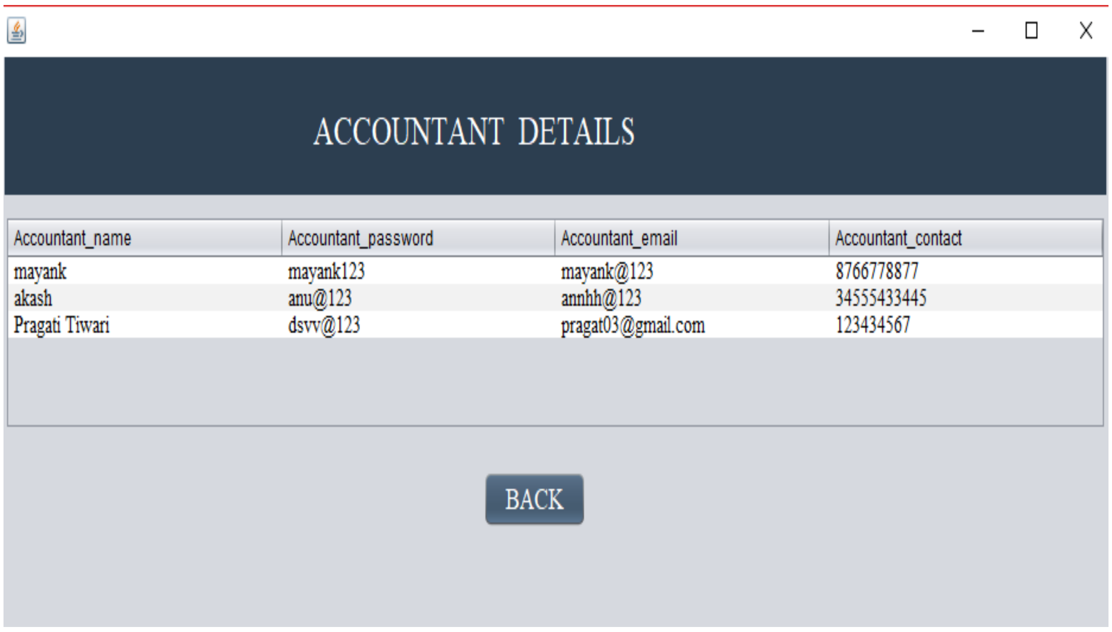
4. Add Accountant Interface.



Snapshot detail:-

5. Add Accountant Button:- Administrator can add accountant with the attribute of accountant name, password, email, contact.

6. View Accountant Interface.



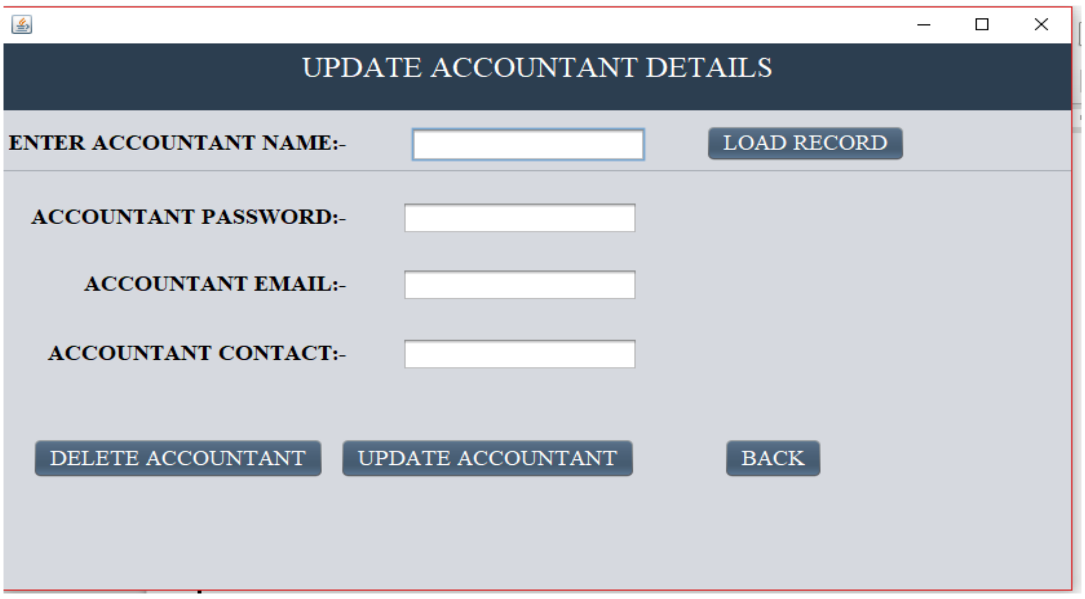
ACCOUNTANT DETAILS

Accountant_name	Accountant_password	Accountant_email	Accountant_contact
mayank	mayank123	mayank@123	8766778877
akash	anu@123	annhh@123	34555433445
Pragati Tiwari	dsvv@123	pragat03@gmail.com	123434567

Snapshot detail:-

View Accountant Button:- Administrator can view accountant details after adding accountant detail by administrator.

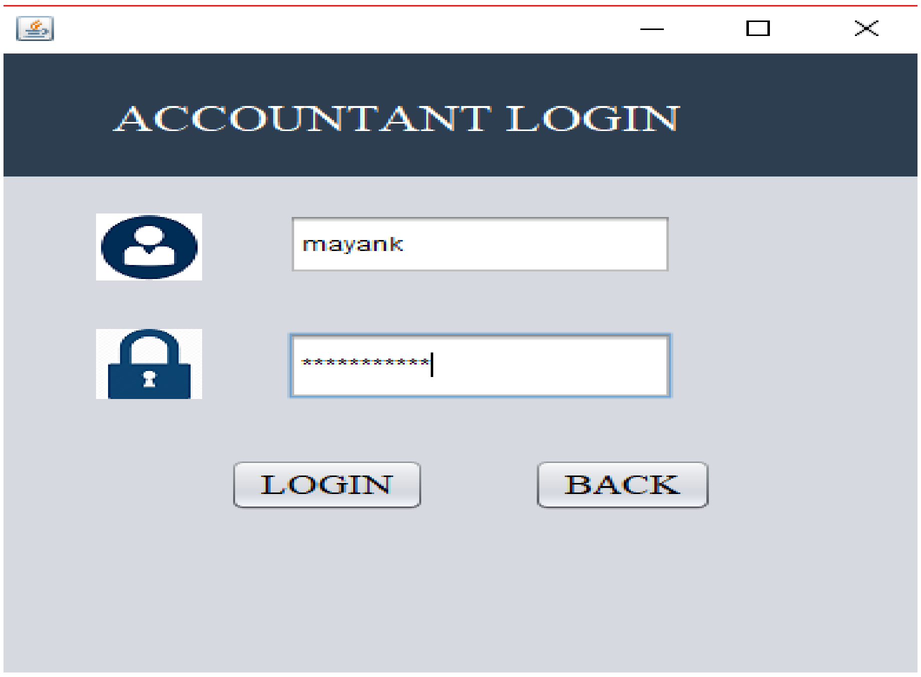
7. Edit Accountant Interface.



Snapshot detail:-

8. Edit Accountant Button:- After clicking the load Button administrator can fetch the detail by accountant name of the accountant. After fetching the accountant name detail will be updated or deleted by the admin. the details of accountant that are already stored in the database.

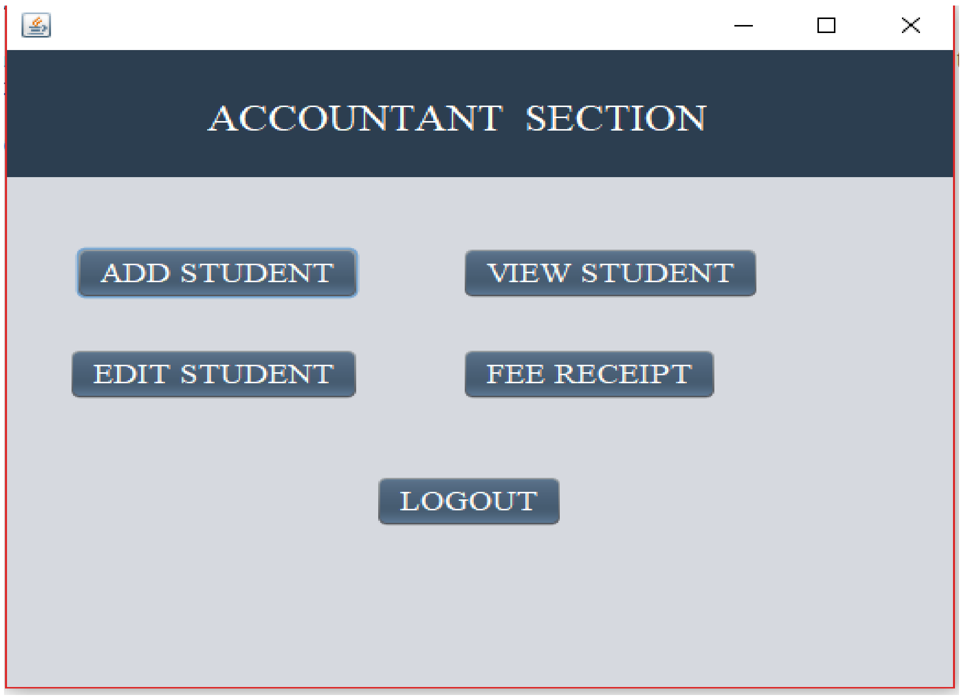
9.. Accountant Login Interface.



Snapshot detail:-

10. Accountant Login:- Accountant can enter the correct user-name and password to enter into the accountant section. The accountant can need only user-name and password detail.

11. Accountant Section Interface.



Snapshot detail:-

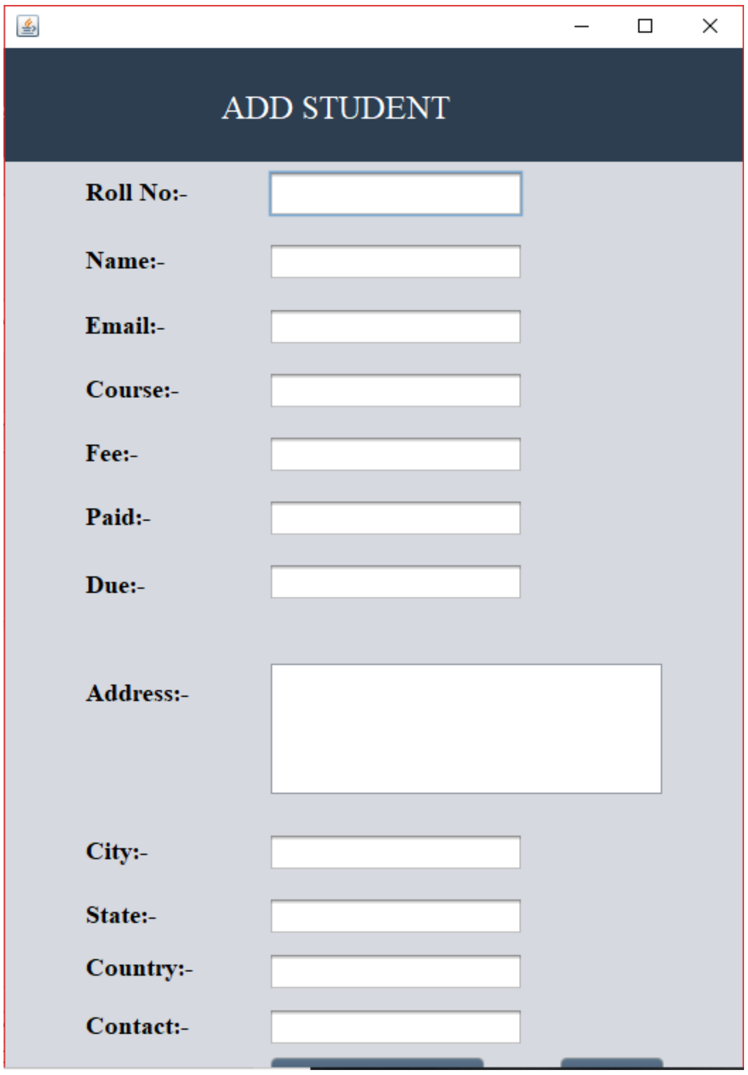
Add Student Button:- Accountant can add Student record.

View student Button:- Accountant can view student details after adding student detail by accountant.

Edit student Button:- After clicking the edit student Button accountant can update the details of student that is already stored in the database. Also accountant can delete the student record permanently.

Accountant Logout Button:- Accountant can logout by pressing the logout button.

12. Add Student Interface:-

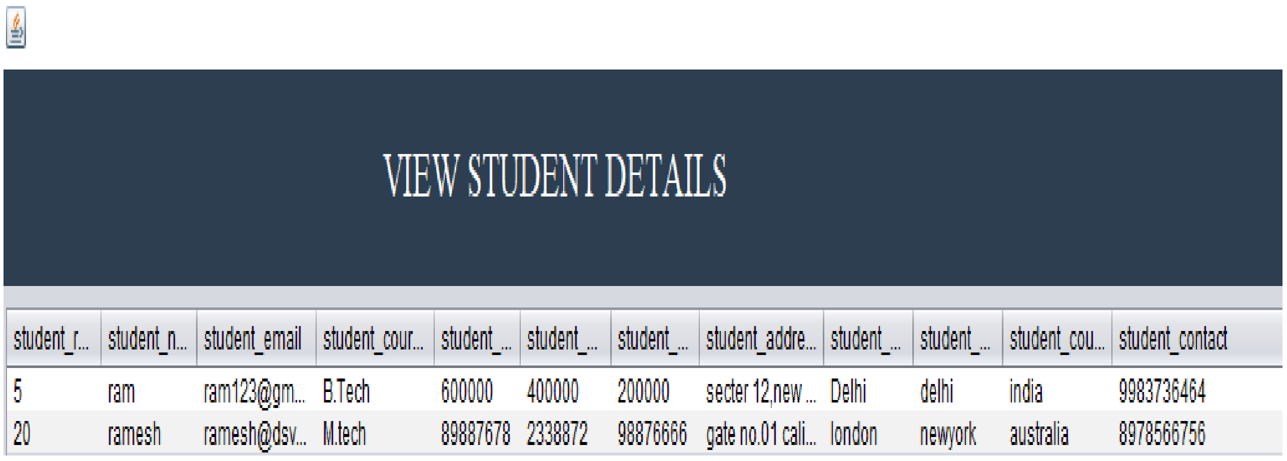

ADD STUDENT

Roll No:-	
Name:-	
Email:-	
Course:-	
Fee:-	
Paid:-	
Due:-	
Address:-	
City:-	
State:-	
Country:-	
Contact:-	

Snapshot detail:-

Add Student Button:- Accountant can add student with the attribute of student name, roll_no, email, contact, fee, paid, due, city, state, country.

13. View Student Interface:-



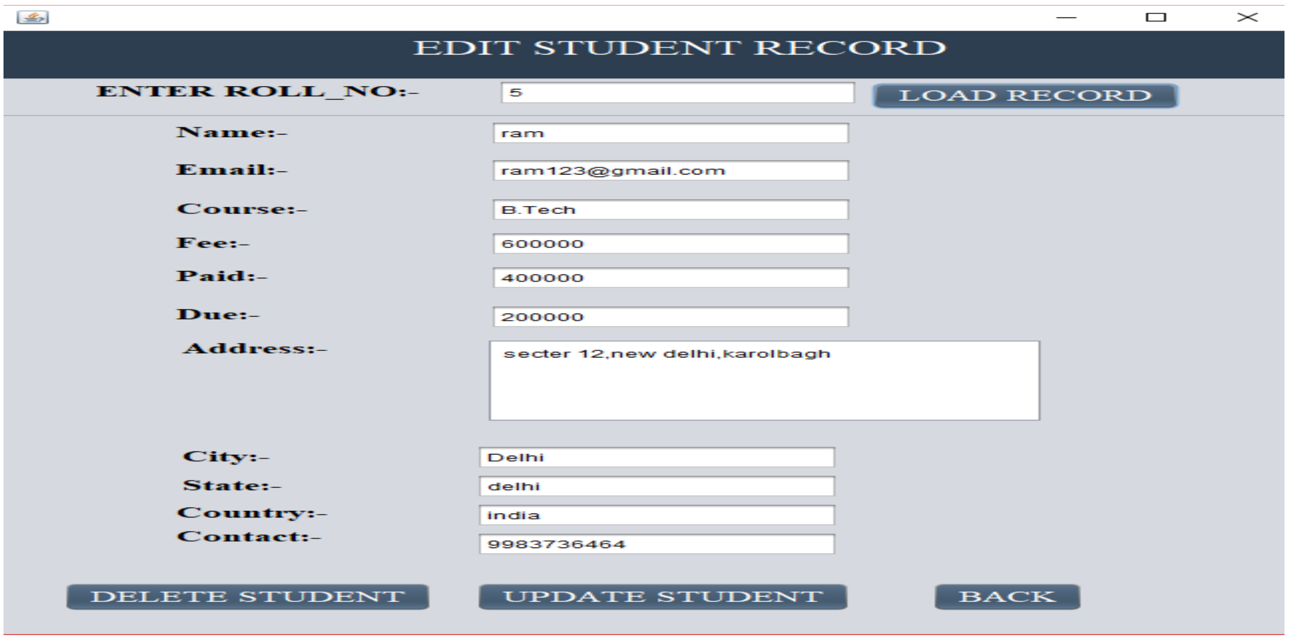
VIEW STUDENT DETAILS

student_r...	student_n...	student_email	student_cour...	student_...	student_...	student_...	student_addre...	student_...	student_...	student_cou...	student_contact
5	ram	ram123@gm...	B.Tech	600000	400000	200000	secter 12,new ...	Delhi	delhi	india	9983736464
20	ramesh	ramesh@dsv...	M.tech	89887678	2338872	98876666	gate no.01 cali...	london	newyork	australia	8978566756

Snapshot detail:-

View student Button:- Accountant can view students details after adding student detail by accountant.

14. Edit Student Interface:-



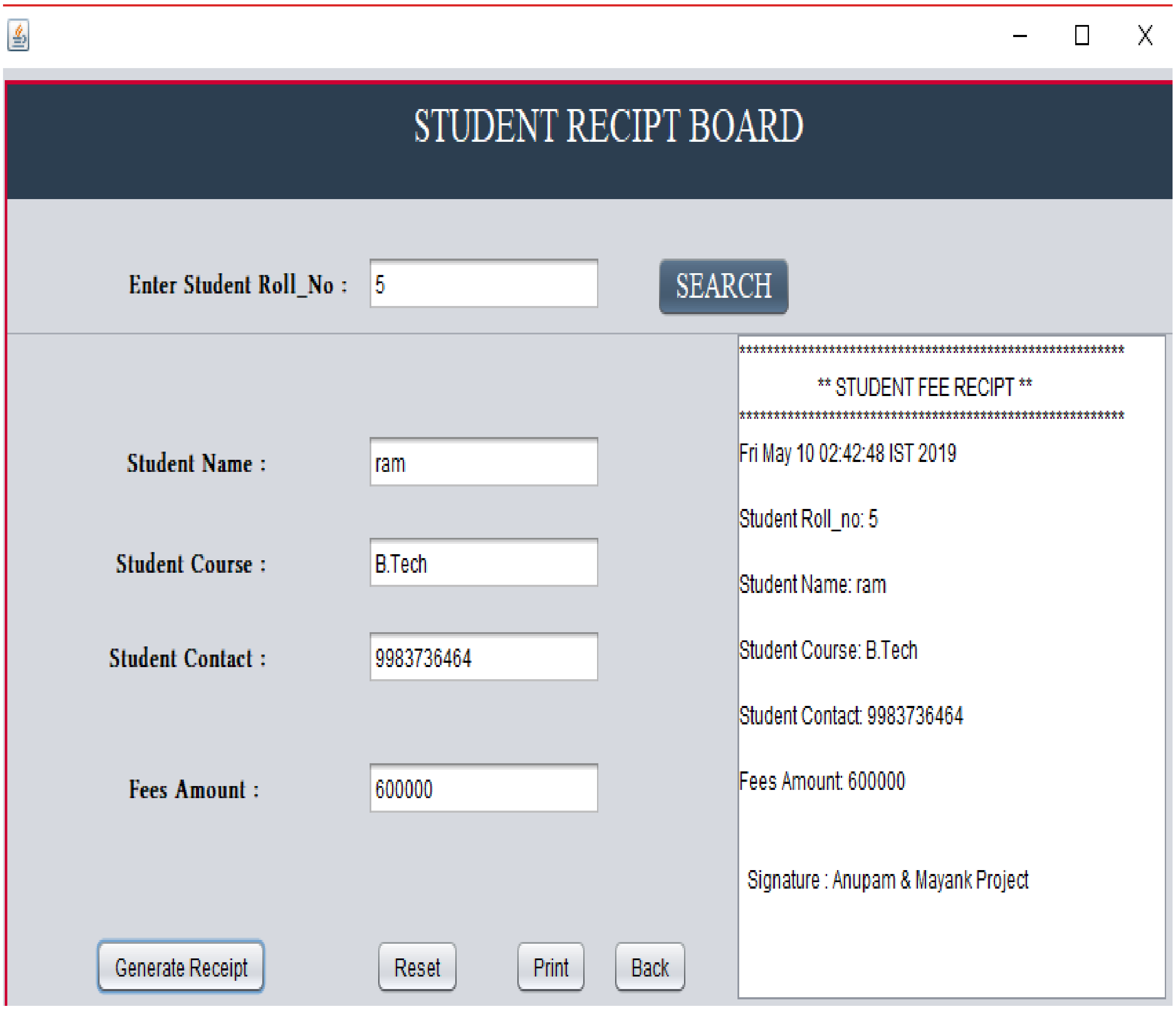
EDIT STUDENT RECORD

ENTER ROLL_NO:-	5
Name:-	ram
Email:-	ram123@gmail.com
Course:-	B.Tech
Fee:-	600000
Paid:-	400000
Due:-	200000
Address:-	secter 12,new delhi,karolbagh
City:-	Delhi
State:-	delhi
Country:-	india
Contact:-	9983736464

Snapshot detail:-

Edit student Button:- After clicking the load Button accountant can fetch the detail of student by roll_no of the student. After fetching the student roll_no detail will be updated or deleted by the accountant. The details of student that are already stored in the database.

15. Fee Receipt Interface:-



Enter Student Roll_No : 5

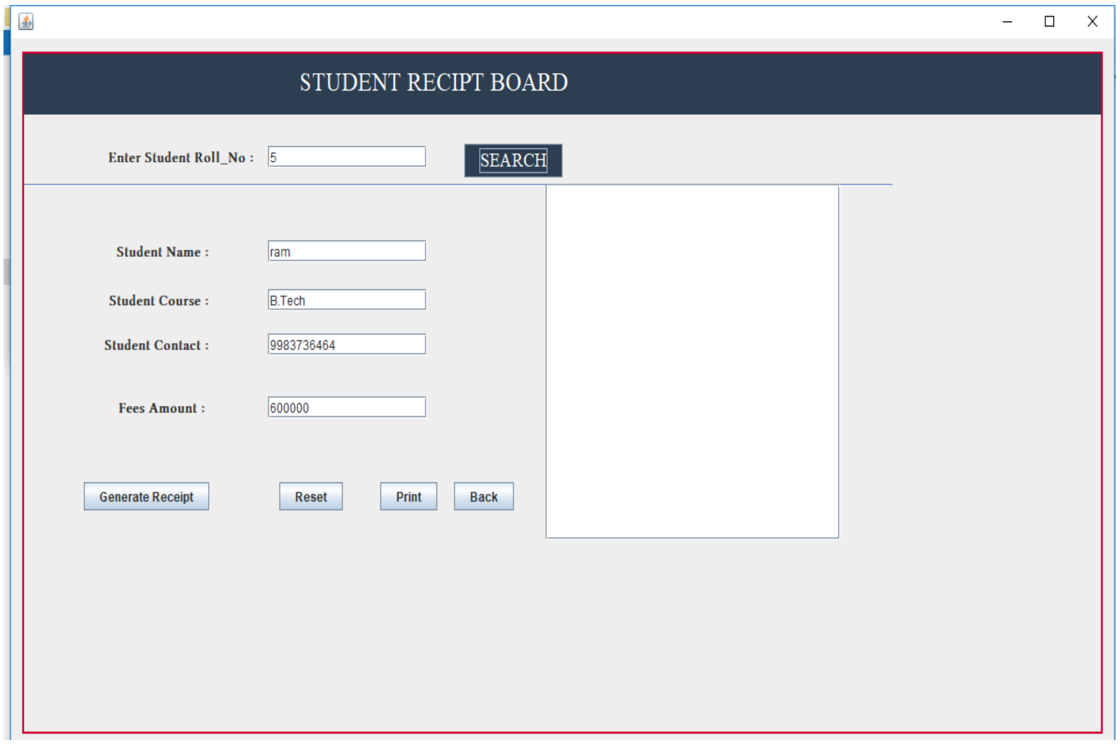
Student Name : ram

Student Course : B.Tech

Student Contact : 9983736464

Fees Amount : 600000

```
*****************************************************
        ** STUDENT FEE RECIPT **
*****************************************************
Fri May 10 02:42:48 IST 2019

Student Roll_no: 5

Student Name: ram

Student Course: B.Tech

Student Contact: 9983736464

Fees Amount: 600000


 Signature : Anupam & Mayank Project
```


STUDENT RECIPT BOARD

Enter Student Roll_No : 5

Student Name : ram

Student Course : B.Tech

Student Contact : 9983736464

Fees Amount : 600000

Snapshot detail:-

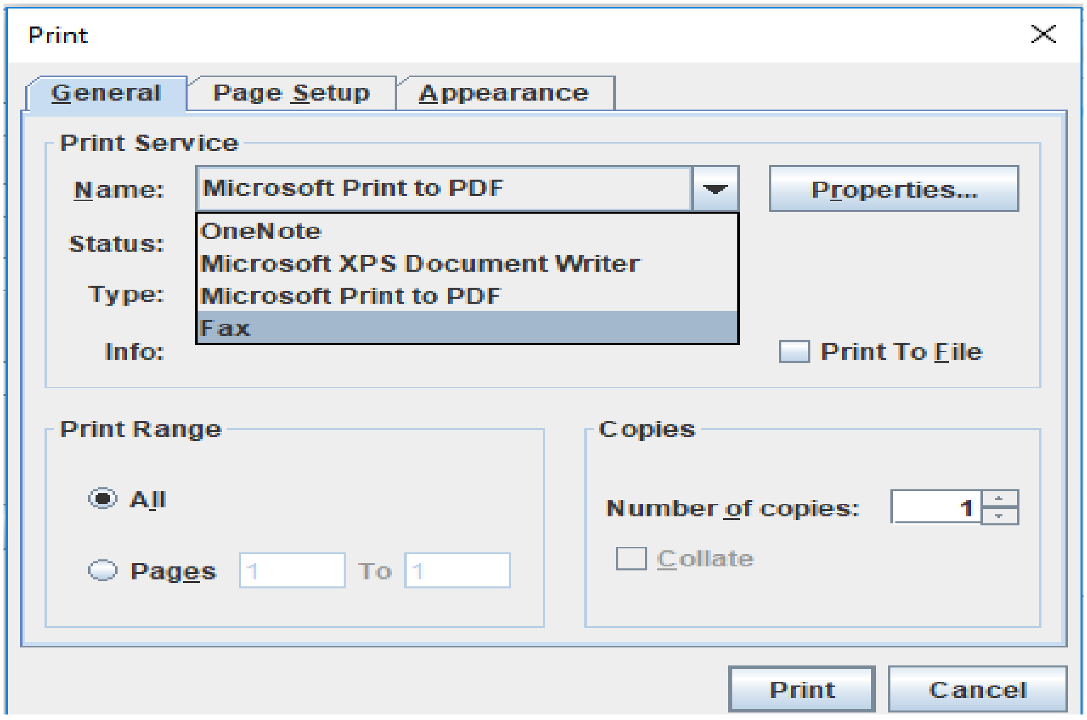
Fee Receipt Button:- Student receipt board open after clicking fee receipt button then accountant can able to generate student fee receipt before accountant can search the students with roll_no. Than print the particular student fee receipt as per confirmation of due amount.

Search Button:- In student fee receipt board through the search button accountant can able to find the student record by student roll_no.

Generate receipt Button:- Generate receipt button can generate the receipt detail in text area after searching the students record.

Reset Button:- Reset button can reset all the details of student that is appear on the text fields.

16. Print Interface:-



Snapshot detail:-

Print Button:- Print Button can print all the details of student that is appear on the text area. After pressing the print button can open the printer interface than it will be printed directly by the printer and also it can be saved in PDF or DOC formate further user can print.

4. SYSTEM ANALYSIS

WHAT IS JAVA LANGUAGE:-

Java is one of the most popular and widely used programming language and platform. A platform is an environment that helps to develop and run programs written in any programming language.

Java is fast, reliable and secure. From desktop to web applications, scientific supercomputers to gaming consoles, cell phones to the Internet, Java is used in every nook and corner.

Java is easy to learn and its syntax is simple and easy to understand. It is based on C++ (so easier for programmers who know C++). Java has removed many confusing and rarely-used features e.g. explicit pointers, operator overloading etc. Java also takes care of memory management and for that, it provides an automatic garbage collector. This collects the unused objects automatically.

Why Java was Created? A Brief History

In 1991, the team "Green Team" led by James Gosling at Sun Microsystems created a programming language for digital consumer devices. The language was called Oak then. Why Oak? Because there was an oak tree outside Gosling's office.

The "Green Team" demonstrated the use of the language with an interactive television. However, it was too advanced for the digital cable television at the time, and more suitable a technology that was starting to take off, the Internet.

Later, the language was renamed **Green** and finally renamed **Java** from Java coffee; hence the coffee-cup logo.

Since C/C++ was popular back then, James Gosling designed the language with

C/C++ style syntax, and philosophy "write once, run anywhere". After years, Sun Microsystems released the first public implementation of Java in 1995. It was announced that the Netscape Navigator Internet browser would incorporate Java technology.

In 2010, Sun Microsystems was completely acquired by Oracle Corporation along with Java.

Java Version History

1. June 1991 - Java language project was initiated
2. JDK 1.0 - January, 1996
3. JDK 1.1 - February, 1997
4. J2SE 1.2 - December, 1998
5. J2SE 1.3 - May, 2000
6. J2SE 1.4 - February, 2002
7. J2SE 5.0 - September, 2004
8. Java SE 6 - December, 2006
9. Java SE 7 - July, 2011
10. Java SE 8 (latest version) - March 18, 2014
11. Java SE 9 - July, 2017 (announced release date)

Feature of java programming

1. Java is platform independent

Java was built with the philosophy of "write once, run anywhere" (WORA). The Java code (pure Java code and libraries) you write on one platform (operating system) will run on other platforms with no modification.

To run Java, an abstract machine called Java Virtual Machine (JVM) is used. The JVM executes the Java bytecode. Then, the CPU executes the JVM. Since all JVMs works exactly the same, the same code works on other operating systems as well, making Java platform-independent.

2. An object-oriented Language

There are different styles of programming. Object-oriented approach is one of the popular programming styles. In object-oriented programming, a complex problem is divided into smaller sets by creating objects. This makes your code reusable, has design benefits, and makes code easier to maintain.

Many programming languages including Java, Python, and C++ has object-oriented features. If you are serious about programming, you should definitely learn object-oriented style of programming.

3. Java is fast

The earlier versions of Java were criticized for being slow. However, things are completely different now. The new JVMs are significantly faster. And, the CPU that executes JVM are also getting more and more powerful.

Now, Java is one of the fastest programming languages. Well optimized Java code is nearly as fast as lower level languages like C/C++, and much faster than Python, PHP etc.

4. Java is secure

The Java platform provides various features for security of Java applications. Some of the high-level features that Java handles are:

- provides secure platform for developing and running applications
- automatic memory management, reduces memory corruption and vulnerabilities
- provides secure communication by protecting the integrity and privacy of data transmitted

5. Large Standard Library

One of the reasons why Java is widely used is because of the availability of huge standard library. The Java environment has hundreds of classes and methods under different packages to help software developers like us. For example,

java.lang- for advanced features of strings, arrays etc.
java.util- for data structures, regular expressions, date and time functions etc.
java.io- for file i/o, exception handling etc.

4.2 NET-BEANS IDE

NetBeans IDE 8.2 provides out-of-the-box code analyzers and editors for working with the latest Java 8 technologies--Java SE 8, Java SE Embedded 8, and Java ME Embedded 8. The IDE also has a range of new tools for HTML5/JavaScript, in particular for Node.js, KnockoutJS, and AngularJS; enhancements that further improve its support for Maven and Java EE with PrimeFaces; and improvements to PHP and C/C++ support.

NetBeans IDE 8.2 is available in English, Brazilian Portuguese, Japanese, Russian, and Simplified Chinese.

Features:-

Best Support for Latest Java Technologies

NetBeans IDE is the official IDE for Java 8. With its editors, code analyzers, 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.
Batch analyzers and converters are provided to search through multiple applications at the same time, matching patterns for conversion to new Java 8 language constructs.
With its constantly improving Java Editor, many rich features and an extensive range of tools, templates and samples, NetBeans IDE sets the standard for developing with cutting edge technologies out of the box.

Fast & Smart Code Editing

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.

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.

Easy & Efficient Project Management

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.

When new developers join your project, they can understand the structure of your application because your code is well-organized.

Rapid User Interface Development

Design GUIs for Java SE, HTML5, Java EE, PHP, C/C++, and Java ME applications quickly and smoothly by using editors and drag-and-drop tools in the IDE.

For Java SE applications, the NetBeans GUI Builder automatically takes care of correct spacing and alignment, while supporting in-place editing, as well. The GUI builder is so easy to use and intuitive that it has been used to prototype GUIs live at customer presentations.

Write Bug Free Code

The cost of buggy code increases the longer it remains unfixed. NetBeans provides static analysis tools, especially integration with the widely used FindBugs tool, for identifying and fixing common problems in Java code. In addition, the NetBeans Debugger lets you place breakpoints in your source code, add field watches, step through your code, run into methods, take snapshots and monitor execution as it occurs.

The NetBeans Profiler provides expert assistance for optimizing your application's speed and memory usage, and makes it easier to build reliable and scalable Java SE, JavaFX and Java EE applications. NetBeans IDE includes a visual debugger for Java SE applications, letting you debug user interfaces without looking into source code. Take GUI snapshots of your applications and click on user interface elements to jump back into the related source code.

4.3 MYSQL DATABASE

What is MySQL?

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

- **MySQL is a database management system.**

  A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

- **MySQL databases are relational.**

  A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

  The SQL part of“MySQL”stands for“Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

  SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual,“SQL-92”refers to the standard released in 1992,“SQL:1999”refers to the standard released in 1999, and“SQL:2003”refers to the current version of the standard. We use the phrase“the SQL standard”to mean the current version of the SQL Standard at any time.

- **MySQL software is Open Source.**

  Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License),to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information.

- **The MySQL Database Server is very fast, reliable, scalable, and easy to use.**

  If that is what you are looking for, you should give it a try. MySQL Server can run comfortably on a desktop or laptop, alongside your other applications, web servers, and so on, requiring little or no attention. If you dedicate an entire machine to MySQL, you can adjust the settings to take advantage of all the memory, CPU power, and I/O capacity available. MySQL can also scale up to clusters of machines, networked together.

  MySQL Server was originally developed to handle large databases much faster than existing solutions and has been successfully used in highly demanding production environments for several years. Although under constant development, MySQL Server today offers a rich and useful set of functions. Its connectivity, speed, and security make MySQL Server highly suited for accessing databases on the Internet.

- **MySQL Server works in client/server or embedded systems.**

  The MySQL Database Software is a client/server system that consists of a multithreaded SQL server that supports different back ends, several different client programs and libraries, administrative tools, and a wide range of application programming interfaces (APIs).

  We also provide MySQL Server as an embedded multithreaded library that you can link into your application to get a smaller, faster, easier-to-manage standalone product.

5. CODING OF PROJECT

5.1 INSERTION.

CODE:-

```
.
.
.
try
        {
            Class.forName("com.mysql.jdbc.Driver");
                try (Connection con =
DriverManager.getConnection("jdbc:MySQL://localhost:3306/fee","root","")) {
                    String sql="insert into accountant_detail values (?,?,?,?)";
                    PreparedStatement pst=con.prepareStatement(sql);
                    pst.setString(1,tf_Accountant_name.getText());
                    pst.setString(2,tf_Accountant_password.getText());
                    pst.setString(3,tf_Accountant_email.getText());
                    pst.setString(4,tf_Accountant_contact.getText());
                    pst.execute();
                    JOptionPane.showMessageDialog(null, "Added Succesfull");
                }
        }
            catch (HeadlessException | ClassNotFoundException | SQLException e)
                    {
                 JOptionPane.showMessageDialog(null, e);
                }


.
.
```

5.2 UPDATION

CODE:-

.

.

.

```
try
        {
            Class.forName("com.mysql.jdbc.Driver");
            try (Connection con =
DriverManager.getConnection("jdbc:MySQL://localhost:3306/fee","root","")) {
                String sql="UPDATE add_student_detail SET student_name='"+
tf_st_update_name.getText()+"',"
                        + " student_email='"+tf_st_update_email.getText()+"',"
                        + "student_course='"+tf_st_update_course.getText()+"',"
                         + "student_fee='"+tf_st_update_fee.getText()+"',"
                         + "student_paid='"+tf_st_update_paid.getText()+"',"
                         + "student_due='"+tf_st_update_due.getText()+"',"
                         + "student_address='"+tf_st_update_address.getText()+"',"
                         + "student_city='"+tf_st_update_city.getText()+"',"
                         + "student_state='"+tf_st_update_state.getText()+"',"
                         + "student_country='"+tf_st_update_country.getText()+"',"
                         + "student_contact='"+tf_st_update_contact.getText()+"'"
                         + " where student_rollno='"+tf_enter_st_roll.getText()+"'";


                PreparedStatement pst=con.prepareStatement(sql);
                //pst.setString(1,String.valueOf(tf_enter_Accountant_name.getText()));
            pst.execute();
            JOptionPane.showMessageDialog(null, " Student Updated Succesfull");
         }
      }
      catch (Exception e)
      {
         JOptionPane.showMessageDialog(null, e);
      }
```

.

.

.

.

5.3 DELETION

CODE:-

.

.

.

```
 try
      {
         Class.forName("com.mysql.jdbc.Driver");
         Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/fee","root","");
            String sql="select * from add_student_detail where student_rollno=?";
            PreparedStatement pst=con.prepareStatement(sql);
            pst.setString(1,tf_enter_st_roll.getText());
            ResultSet rs=pst.executeQuery();
            if(rs.next()){
            String setname=rs.getString("student_name");
            tf_st_update_name.setText(setname);
            String setemail=rs.getString("student_email");
            tf_st_update_email.setText(setemail);
            String setcourse=rs.getString("student_course");
            tf_st_update_course.setText(setcourse);
            String setfee=rs.getString("student_fee");
            tf_st_update_fee.setText(setfee);
            String setpaid=rs.getString("student_paid");
            tf_st_update_paid.setText(setpaid);
            String setdue=rs.getString("student_due");
            tf_st_update_due.setText(setdue);
            String setaddress=rs.getString("student_address");
            tf_st_update_address.setText(setaddress);
            String setcity=rs.getString("student_city");
            tf_st_update_city.setText(setcity);
            String setstate=rs.getString("student_state");
            tf_st_update_state.setText(setstate);
            String setcountry=rs.getString("student_country");
            tf_st_update_country.setText(setcountry);
            String setcontact=rs.getString("student_contact");
            tf_st_update_contact.setText(setcontact);
            pst.executeQuery();
          JOptionPane.showMessageDialog(null, " STUDENT RECORD LOADED
SUCCESSFULLY!!");
          }
      }
      catch (Exception e)
      {
          JOptionPane.showMessageDialog(null, e);
      }
.
```

.

.

.

5.4 SEARCHING

CODE:-

.

.

.

```
  try
        {
             Class.forName("com.mysql.jdbc.Driver");
             Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/fee","root","");
             String sql="select * from add_student_detail where student_rollno=?";
             PreparedStatement pst=con.prepareStatement(sql);
             pst.setString(1,st_roll_no.getText());
             ResultSet rs=pst.executeQuery();
             if(rs.next()){
                 String setname=rs.getString("student_name");
                 st_name.setText(setname);
                 String setcourse=rs.getString("student_course");
                 st_course.setText(setcourse);
                 String setcontact=rs.getString("student_contact");
                 st_contact.setText(setcontact);
                 String setfee=rs.getString("student_fee");
                 st_fee.setText(setfee);
                 pst.executeQuery();
                 JOptionPane.showMessageDialog(null, " STUDENT RECORD LOADED
SUCCESSFULLY!!");
```

```
        }
    }
    catch (Exception e)
    {
        JOptionPane.showMessageDialog(null, e);
    }
```

.

.

.

.

.

.

.

6. FUTURE SCOPE OF THE PROJECT

My project “FEE MANAGEMENT SYSTEM” will be able to implement in future after making some changes and modifications as we make our project at a very low level. So the modifications that can be done in our project is to add one major change which can be done in this project is that to add the snaps of the student of which the record is entered. This will result in total identification of the given student. Similarly various modifications can be done to enhance the usability of the given project as suitable for user’s requirement.

7. CONCLUSION

From this project we can conclude that if this program is very useful in fee management as it provides more convenience than the manual work. It provides easy methods to manage the load of work easily for the users. It is much fast and more efficient as the data once entered can be modified and accessed easily. The program can be used per the requirement of the user as it is very easy to understand.

8. REFERENCES

1. https://www.slideshare.net/Divya_Gupta19/synopsis-of-fee-management-syst
2. https://www.javatpoint.com/fee-report-project-in-java
3. https://www.google.com/search?client=firefox-b-d&q=summary+of+fee+management+system
4. https://1000projects.org/fee-management-system-project-documentation.html
5. https://www.freeprojectz.com/project-report/6766
6. https://dev.mysql.com/doc/refman/5.7/en/what-is-mysql.html
7. https://netbeans.org/features/
8. https://www.programiz.com/java-programming#what-is-java
9. https://www.geeksforgeeks.org/java/
10. https://netbeans.org/community/releases/82/
11. https://searchoracle.techtarget.com/definition/MySQL
12. https://www.tutorialspoint.com/mysql/mysql-introduction.htm