CHAPTER-1

INTRODUCTION

1.1 Introduction

In every field of daily life to live , management is the process of execution and saving of time. Management of things makes us more easier to reach the solution or to abstract the information. University Management System (UMS) is a comprehensive software solution designed to streamline and automate various administrative and academic processes within a university. These systems are crucial for managing the vast amount of data and tasks associated with students, faculty, courses, examinations, admissions, and other administrative functions.

A fully functional University Management System Project in Java that being developed in Java Programming using Net Beans and MySQL Database as the System's Back-End. It is created for the University and its affiliated institutions to conduct, monitor, and analyze complicated activities such as Centralized Admission, Centralized Examination, and much more.

1.2 Objective

The objectives of a University Management System is to enhance the efficiency, transparency, and effectiveness of various administrative and academic processes within a university. Here are the key objectives of implementing a UMS.

1. To automate administrative tasks such as admissions, enrollment, fee collection, and document management to reduce paperwork and manual efforts.

2. To facilitate effective management of academic activities including curriculum planning, class scheduling, attendance tracking, and examination management to ensure smooth conduct of classes and exams.

3. To improve communication and collaboration among students, faculty, and staff through internal messaging systems, announcements, and online discussion platforms.

1.3 Methodology

The methodology for developing University Management System involves a structured approach to designing, developing, testing, and deploying the system.

1.3.1 Requirements Gathering:

- Define the project scope and objectives.
- Conduct interviews and discussions with stakeholders (administrators, teachers, students, etc.) to understand their needs and expectations.

1.3.2 System Design:

- Architecture Design: Plan the system architecture, including databases, servers, and software components.
- User Interface Design: Design intuitive and user-friendly interfaces for students, faculty, and administrators.
- Database Design: Create a robust database schema to store student information, academic data, and administrative records securely.

1.3.3 Development:

- Select Technology Stack: Choose appropriate programming languages, frameworks, and databases based on the project requirements.
- Iterative Development: Develop the system iteratively, focusing on core modules like student management, academic management, and communication features.

1.3.4 Testing :

- Quality Assurance: Implement rigorous testing, including unit testing, integration testing, and user acceptance testing, to identify and fix issues promptly.

1.3.5 Implementation and Deployment:

- Data Migration: Migrate existing data from legacy systems to the new UMS without data loss or corruption.
- Pilot Testing: Deploy the system in a controlled environment for pilot testing. Gather feedback and make necessary adjustments.
- Full Deployment: Deploy the system institution-wide, ensuring necessary training and support for users.

1.3.6 Training and Support:

- User Training: Conduct training sessions for students, faculty, and staff to familiarize them with the new system's features and functionalities.
- Provide Support: Establish a support system to address user queries, issues, and provide ongoing assistance.

1.3.7. Security and Compliance:

- Data Security: Implement robust security measures to protect sensitive data, including encryption, secure login mechanisms, and regular security audits.
- Compliance: Ensure the system complies with data protection laws, educational regulations, and other legal requirements.

1.3.8 Monitoring and Maintenance:

- Monitoring: Implement monitoring tools to track system performance, user activities, and security threats.
- Regular Updates: Provide regular updates and patches to fix bugs, introduce new features, and enhance system security.
- Performance Optimization: Continuously optimize the system for better performance and scalability based on usage patterns.

CHAPTER-2
LITERATURE SURVEY

The literature survey for University Management System project would cover several areas, including:

1. Student Information Management: Store and manage detailed information about students, including personal details, academic history, attendance records, etc.

In [1] , Analytical results indicate that project network embedded significantly influences the development of creativity. To stimulate the creativity of student teams, teachers can encourage team members to interact with each other and to view and emulate ideas and suggestions related to other projects.

2. Academic Management: Define and manage course curricula, including subjects, modules, and manage timetables for classes, exams, and other academic activities.

According to [2], Teachers can use the Academic Management System Mobile Application to take attendance, grade assignments and submit progress reports.

3. Faculty Management: Store and manage information about faculty members, including qualifications, research, and teaching experience.

From[3]Faculty management can be used for management and arrangement of faculty members .

4. Administrative Functions: Manage student fees, scholarships, and financial transactions.Manage student enrollment in various courses and programs. Integrate with Learning Management Systems (LMS) for online courses and resources.

According to [3], we can manage administration stuffs in centralized way where data cannot be lost and is secured

CHAPTER-3

HARADWARE AND SOFTAWARE REQUIREMENTS

A System Requirements Specification (SRS) (also known as a Software Requirements Specification) is a document or set of documentation that describes the features and behaviour of a system or software application. It includes a variety of elements that attempts to define the intended functionality required.

3.1 Hardware Requirements

Hardware requirement analysis is to define and analyse a complete set of functional,

operational, performance, interface, quality factors, design, criticality, and test requirements.

For execution and developing this project, system must contain following details:

Items	Type/Version
Processor Brand	Intel
Processor Type	Core i3
Processor Speed	2 GHz
Processor Count	1
RAM Size	2 GB
Memory Technology	DDR3
Hard Drive Size	160 GB

3.2 SOFTWARE REQUIREMENTS

For coding and logic, we need following software requirements to complete the project :-

Operating system	Windows 10
Application server	JAVA (Net Beans)
Front end	JAVA
Connectivity	JDBC Driver
Database connectivity	XAMPP (MySQL Database)

JDK: The Java Development Kit (JDK) is a software development kit used for developing applications in Java. It includes a set of libraries, tools, and a Java Virtual Machine (JVM) to run Java code on various platforms. The JDK provides everything required to compile, debug, and run Java applications, including the Java runtime environment and the Java class libraries.

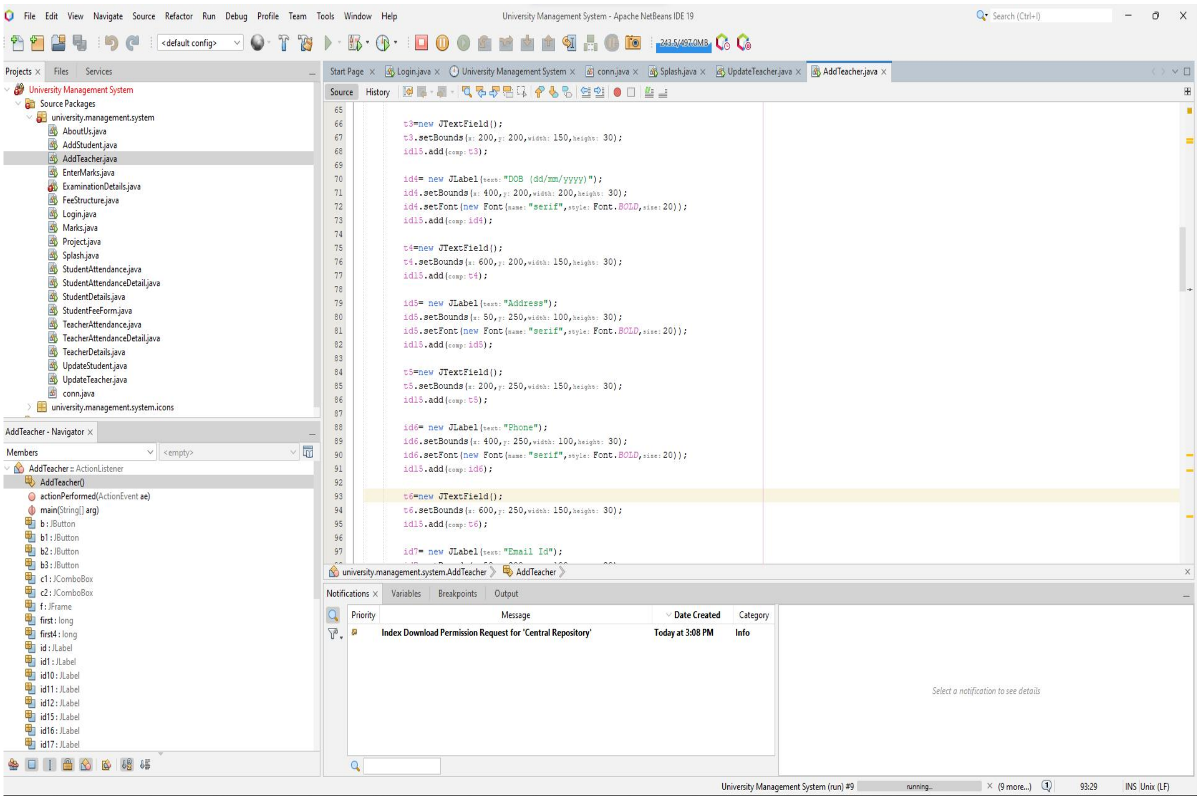
JRE: JRE stands for Java Runtime Environment. It is a software package that provides the environment necessary to run Java applications on a computer. The JRE includes the Java Virtual Machine (JVM), necessary libraries and files, and a class loader to execute Java code. It is part of the Java Development Kit (JDK), which also includes tools for Java developers to create and compile Java applications.

JVM: JVM stands for Java Virtual Machine, it is an abstract computing machine that enables a computer to run Java programs. It is a software that acts as an intermediary between the end user's software application and the computer hardware, converting Java code into machine language and executing it on the computer. JVM also manages memory, provides security, and runs multiple applications at the same time.

AWT: AWT (Abstract Window Toolkit) is a graphical user interface (GUI) toolkit in Java that provides a set of classes and methods for creating and managing graphical user interfaces for Java applications. It is part of the Java SE (Standard Edition) platform, and it provides a way to create windows, dialog, buttons, labels, text fields, and other GUI components for building desktop applications in Java.

SWING

Swing is a Java graphical user interface (GUI) toolkit that is part of the Java Foundation Classes (JFC). It provides a set of components for creating desktop applications with rich and interactive graphical interfaces. Unlike AWT, Java Swing provides platform-independent and lightweight components.The javax.swing package provides classes for java swing API such as J Button, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.


University Management System - Apache NetBeans IDE 19
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Start Page
Login.java
University Management System
conn.java
Splash.java
UpdateTeacher.java
AddTeacher.java
Source
History
university.management.system.AddTeacher
AddTeacher
Notifications
Variables
Breakpoints
Output

```
t3=new JTextField();
t3.setBounds(x: 200,y: 200,width: 150,height: 30);
id15.add(comp: t3);

id4= new JLabel(text: "DOB (dd/mm/yyyy)");
id4.setBounds(x: 400,y: 200,width: 200,height: 30);
id4.setFont(new Font(name: "serif",style: Font.BOLD,size: 20));
id15.add(comp: id4);

t4=new JTextField();
t4.setBounds(x: 600,y: 200,width: 150,height: 30);
id15.add(comp: t4);

id5= new JLabel(text: "Address");
id5.setBounds(x: 50,y: 250,width: 100,height: 30);
id5.setFont(new Font(name: "serif",style: Font.BOLD,size: 20));
id15.add(comp: id5);

t5=new JTextField();
t5.setBounds(x: 200,y: 250,width: 150,height: 30);
id15.add(comp: t5);

id6= new JLabel(text: "Phone");
id6.setBounds(x: 400,y: 250,width: 100,height: 30);
id6.setFont(new Font(name: "serif",style: Font.BOLD,size: 20));
id15.add(comp: id6);

t6=new JTextField();
t6.setBounds(x: 600,y: 250,width: 150,height: 30);
id15.add(comp: t6);

id7= new JLabel(text: "Email Id");
```

Priority	Message	Date Created	Category
	Index Download Permission Request for 'Central Repository'	Today at 3:08 PM	Info

Fig:Apache Net Beans Software

CHAPTER-4

SYSTEM DESIGN

4.1 E-R Diagram

An Entity-Relationship Diagram (ERD) is a visual representation of the data model that represents entities within a system and the relationships between those entities. An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research

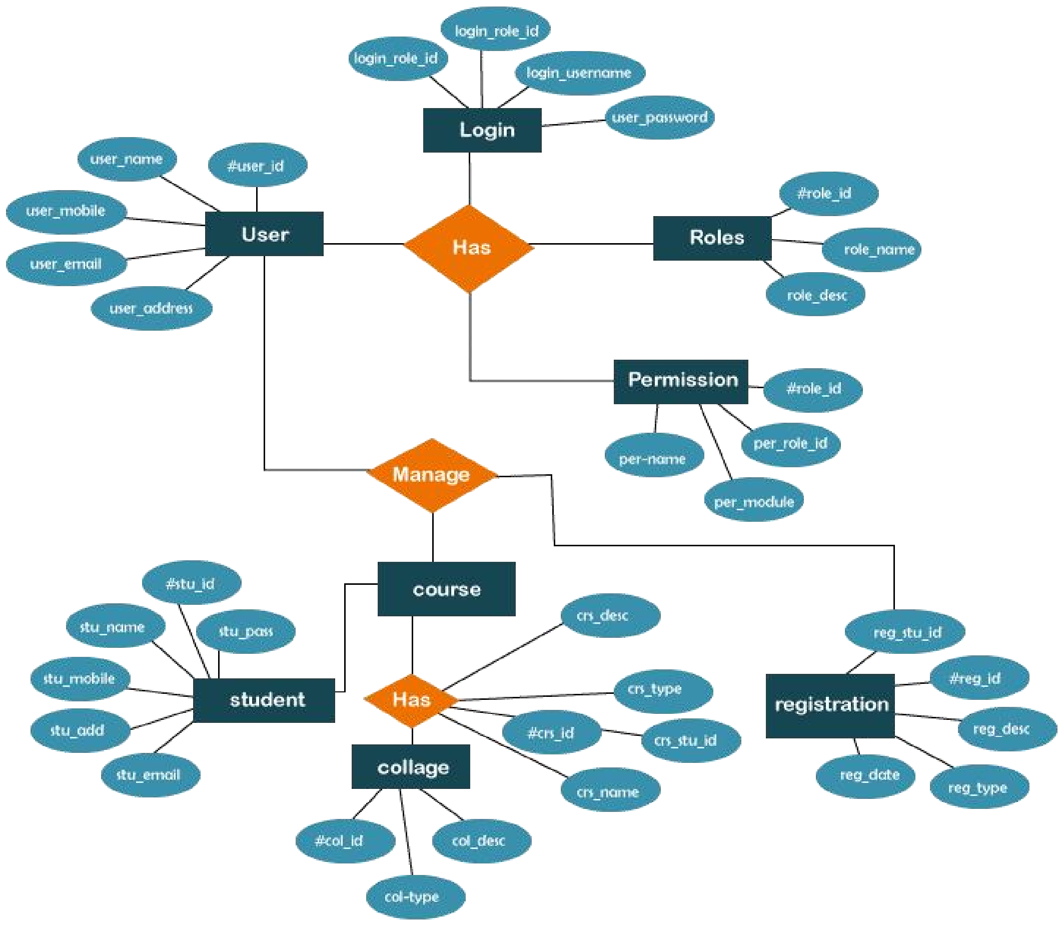


Fig:E-R Diagram

4.2 Flowchart

A flowchart is a diagrammatic representation of a process. It illustrates the steps or activities involved in a workflow, showing the direction of the process flow. Flowcharts are used in various fields, such as programming, business process analysis, and system design, to visually represent the steps and decision points in a process.

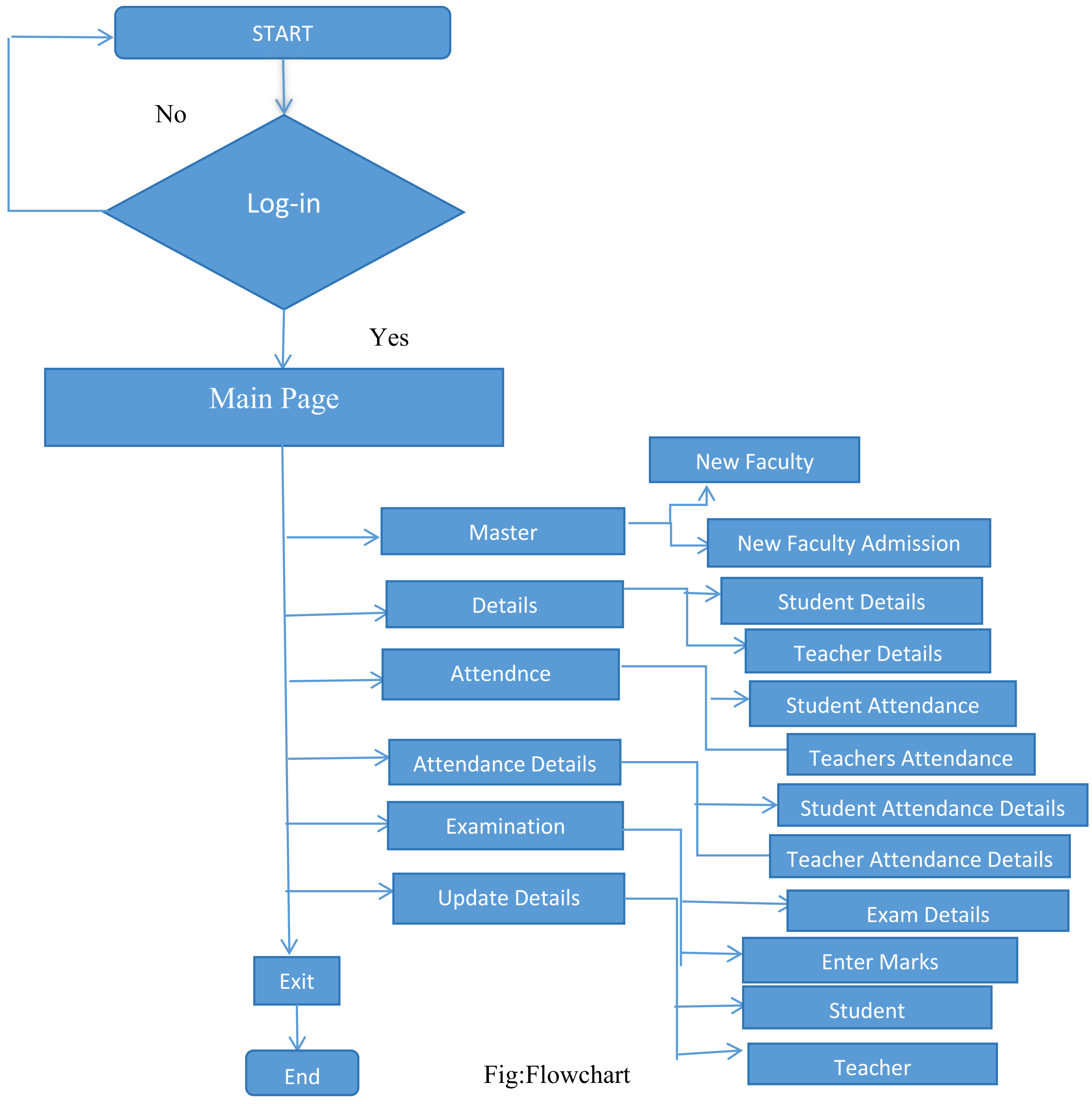


Fig:Flowchart

CHAPTER-5

DEVELOPMENT

5.1 Source Code :

```
package university.management.system;


//import java.awt.event.ActionEvent;

import java.sql.ResultSet;

import javax.swing.JOptionPane;



public class newlogin extends javax.swing.JFrame {


    public newlogin() {

        initComponents();

    }



    @SuppressWarnings("unchecked")

    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

    private void initComponents() {


        jProgressBar1 = new javax.swing.JProgressBar();
```

```
jPanel1 = new javax.swing.JPanel();

left = new javax.swing.JPanel();

jLabel4 = new javax.swing.JLabel();

jLabel5 = new javax.swing.JLabel();

jPanel2 = new javax.swing.JPanel();

jLabel1 = new javax.swing.JLabel();

jLabel2 = new javax.swing.JLabel();

jTextField1 = new javax.swing.JTextField();

jLabel3 = new javax.swing.JLabel();

jPasswordField1 = new javax.swing.JPasswordField();

jButton1 = new javax.swing.JButton();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);


jPanel1.setPreferredSize(new java.awt.Dimension(800, 500));

jPanel1.setLayout(null);


left.setBackground(new java.awt.Color(204, 204, 204));

left.setPreferredSize(new java.awt.Dimension(400, 500));


jLabel4.setFont(new java.awt.Font("Times New Roman", 1, 48)); // NOI18N

jLabel4.setForeground(new java.awt.Color(0, 102, 102));

jLabel4.setText("COLLEGE");
```

```
jLabel5.setFont(new java.awt.Font("Times New Roman", 1, 18)); // NOI18N
jLabel5.setForeground(new java.awt.Color(0, 102, 102));
jLabel5.setText("Management System");


javax.swing.GroupLayout leftLayout = new javax.swing.GroupLayout(left);
left.setLayout(leftLayout);
leftLayout.setHorizontalGroup(
   leftLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
   .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
leftLayout.createSequentialGroup()
      .addContainerGap(120, Short.MAX_VALUE)
      .addGroup(leftLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.
TRAILING)
         .addComponent(jLabel5)
         .addComponent(jLabel4))
      .addGap(43, 43, 43))
);
leftLayout.setVerticalGroup(
   leftLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
   .addGroup(leftLayout.createSequentialGroup()
      .addGap(184, 184, 184)
      .addComponent(jLabel4,    javax.swing.GroupLayout.PREFERRED_SIZE,    47,
javax.swing.GroupLayout.PREFERRED_SIZE)
      .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
      .addComponent(jLabel5)
```

```
        .addContainerGap(242, Short.MAX_VALUE))
);

jPanel1.add(left);
left.setBounds(0, 0, 400, 500);

jPanel2.setBackground(new java.awt.Color(0, 102, 102));
jPanel2.setPreferredSize(new java.awt.Dimension(400, 500));

jLabel1.setFont(new java.awt.Font("Times New Roman", 1, 28)); // NOI18N
jLabel1.setForeground(new java.awt.Color(204, 204, 204));
jLabel1.setText("Login");

jLabel2.setFont(new java.awt.Font("Times New Roman", 0, 16)); // NOI18N
jLabel2.setForeground(new java.awt.Color(204, 204, 204));
jLabel2.setText("Email");

jTextField1.setPreferredSize(new java.awt.Dimension(65, 25));
jTextField1.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jTextField1ActionPerformed(evt);
    }
});
```

```
        jLabel3.setFont(new java.awt.Font("Times New Roman", 0, 16)); // NOI18N
        jLabel3.setForeground(new java.awt.Color(204, 204, 204));
        jLabel3.setText("Password");

        jPasswordField1.setPreferredSize(new java.awt.Dimension(65, 25));

        jButton1.setBackground(new java.awt.Color(0, 102, 102));
        jButton1.setFont(new java.awt.Font("Times New Roman", 0, 18)); // NOI18N
        jButton1.setForeground(new java.awt.Color(204, 204, 204));
        jButton1.setText("Login");
        jButton1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                actionPerformed(evt);
            }
        });

        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignm
ent.LEADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGap(159, 159, 159)
                        .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 83,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGap(46, 46, 46)
                        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Ali
gnment.LEADING, false)
                            .addComponent(jLabel2)
                            .addComponent(jLabel3)
                            .addComponent(jTextField1, javax.swing.GroupLayout.DEFAULT_SIZE,
328, Short.MAX_VALUE)
                            .addComponent(jButton1)
                            .addComponent(jPasswordField1,
javax.swing.GroupLayout.DEFAULT_SIZE,    javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))))
                .addContainerGap(26, Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(

jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGap(73, 73, 73)
                .addComponent(jLabel1,    javax.swing.GroupLayout.PREFERRED_SIZE,    51,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(18, 18, 18)
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
    javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(35, 35, 35)
                .addComponent(jLabel3)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jPasswordField1, javax.swing.GroupLayout.PREFERRED_SIZE,
    39, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(31, 31, 31)
                .addComponent(jButton1, javax.swing.GroupLayout.PREFERRED_SIZE, 43,
    javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(118, Short.MAX_VALUE))
        );


        jPanel1.add(jPanel2);
        jPanel2.setBounds(400, 0, 400, 500);


        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
                layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel1,                    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            );

            pack();
        }// </editor-fold>//GEN-END:initComponents

        private    void    jTextField1ActionPerformed(java.awt.event.ActionEvent    evt)    {//GEN-
    FIRST:event_jTextField1ActionPerformed
            // TODO add your handling code here:
        }//GEN-LAST:event_jTextField1ActionPerformed

        private        void        actionPerformed(java.awt.event.MouseEvent        evt)        {//GEN-
    FIRST:event_actionPerformed
            try{
                conn c1 = new conn();
                String u =  jTextField1.getText();
                String v = jPasswordField1.getText();

                String q = "select * from login where username='"+u+"' and password='"+v+"'";

                ResultSet rs = c1.s.executeQuery(q);
                if(rs.next()){
                    new Project().setVisible(true);
                    setVisible(false);
                }else{
                    JOptionPane.showMessageDialog(null, "Invalid login");
                    setVisible(false);
                }
            }catch(Exception e){
                e.printStackTrace();
            }
        }//GEN-LAST:event_actionPerformed

        public static void main(String args[]) {

            java.awt.EventQueue.invokeLater(new Runnable() {
                public void run() {
                    new newlogin().setVisible(true);
                }
            });
        }

        // Variables declaration - do not modify//GEN-BEGIN:variables
        private javax.swing.JButton jButton1;
        private javax.swing.JLabel jLabel1;
        private javax.swing.JLabel jLabel2;
        private javax.swing.JLabel jLabel3;
```

```
    private javax.swing.JLabel jLabel4;

    private javax.swing.JLabel jLabel5;

    private javax.swing.JPanel jPanel1;

    private javax.swing.JPanel jPanel2;

    private javax.swing.JPasswordField jPasswordField1;

    private javax.swing.JProgressBar jProgressBar1;

    private javax.swing.JTextField jTextField1;

    private javax.swing.JPanel left;

    // End of variables declaration//GEN-END:variables

}
```

CHAPTER-6

IMPLEMENTATION AND TESTING

6.1 Implementation

6.1.1 Database Implementation:

Create the necessary database tables and relationships based on the system's requirements.Implement database operations for data manipulation, retrieval, and storage.

6.1.2 Back end Development:

Develop server-side components in Java, handling business logic, data processing, and interactions with the database.

6.1.3 Front end Development:

Design and implement user interfaces using JavaFX, Swing, Java GUI framework.Ensured a user-friendly and intuitive interface for end-users.

6.1.4 Security Implementation:

Implement authentication and authorization mechanisms to ensure secure access control. Encrypt sensitive data and implement secure communication protocols.

6.1.5 Deployment:

Deploy the University Management System (UMS) on a server or cloud platform.Configure server settings, databases, and other required components.

6.2 Testing of UMS

6.2.1 Unit Testing:

Individual components and modules of the system is tested in isolation to ensure they function as intended. Use JUnit or similar testing frameworks to automate unit tests for classes and methods.

6.2.2 Integration Testing:

Test how different modules interact with each other.Ensure data flow and integration points work seamlessly.

6.2.3 System Testing:

Test the entire system as a whole.Validate all system requirements against the implemented system.Test various use cases and scenarios to verify the system’s behavior in different situations.

6.2.4 User Acceptance Testing (UAT):

Allowed end-users (faculty, staff, and administrators) to use the system in a controlled environment. Gather feedback and ensure the system meets the users' needs and expectations.

CHAPTER-7

SNAPSHOTS

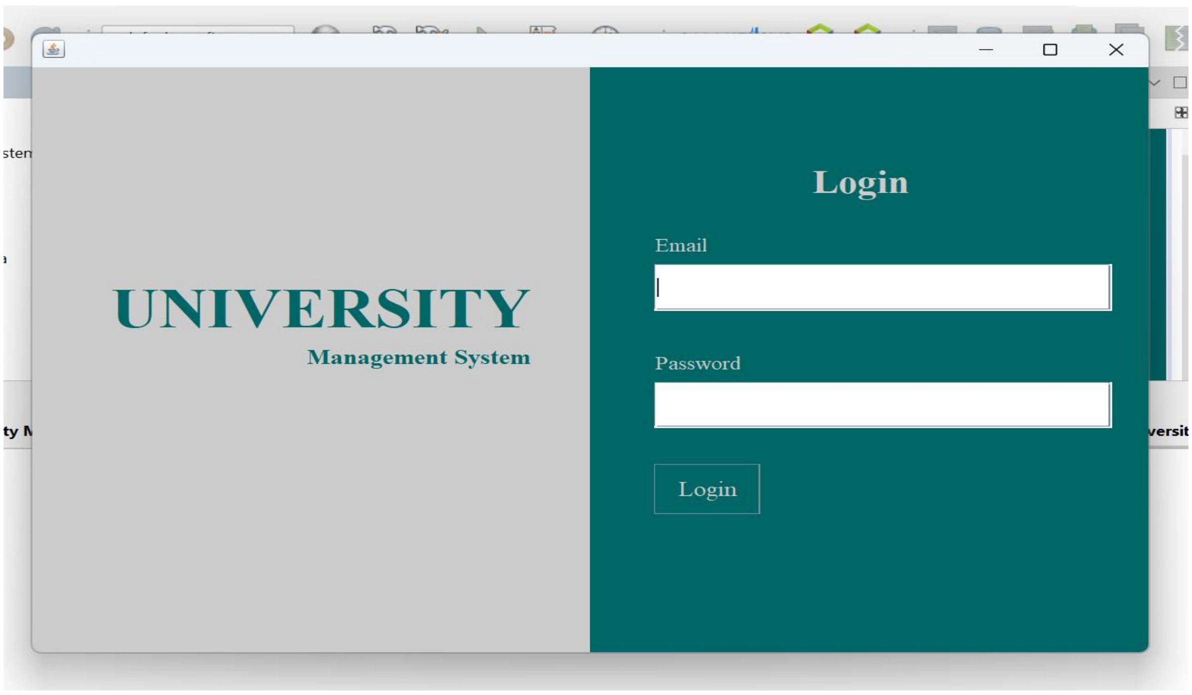
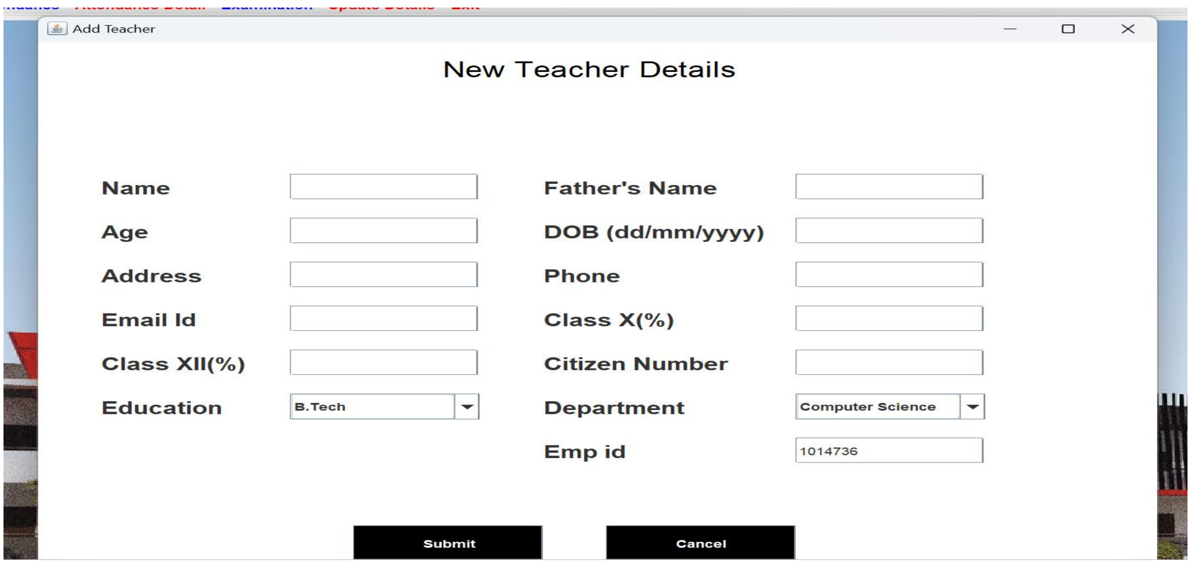
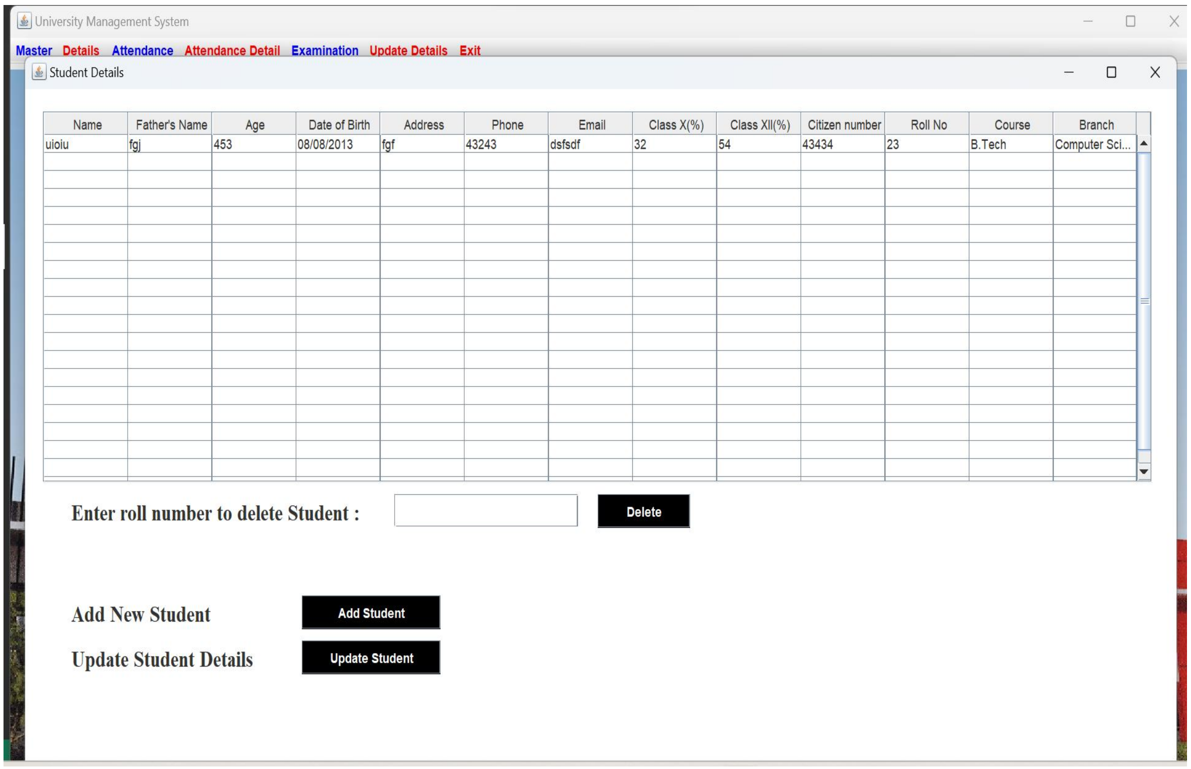


Fig:Login Panel



Name		Father's Name	
Age		DOB (dd/mm/yyyy)	
Address		Phone	
Email Id		Class X(%)	
Class XII(%)		Citizen Number	
Education	B.Tech	Department	Computer Science
		Emp id	1014736

Fig:Teacher Details Panel


University Management System
Master Details Attendance Attendance Detail Examination Update Details Exit
Student Details

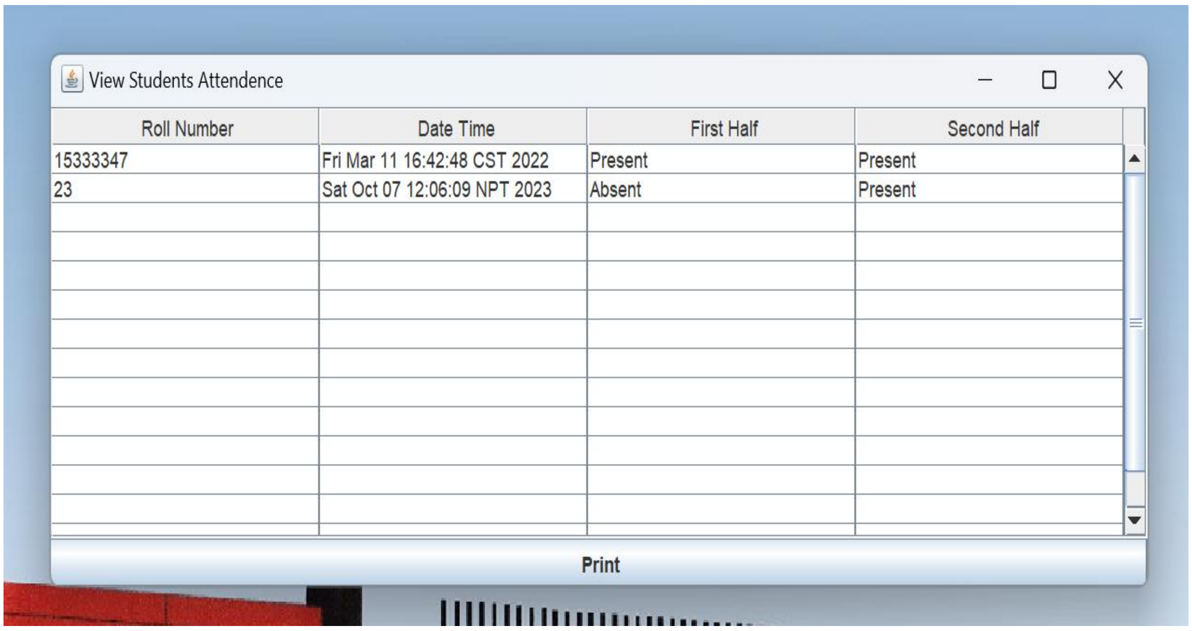
Name	Father's Name	Age	Date of Birth	Address	Phone	Email	Class X(%)	Class XII(%)	Citizen number	Roll No	Course	Branch
uioiu	fgj	453	08/08/2013	fgf	43243	dsfsdf	32	54	43434	23	B.Tech	Computer Sci...

Enter roll number to delete Student :

Add New Student

Update Student Details

Fig:Student information table



Roll Number	Date Time	First Half	Second Half
15333347	Fri Mar 11 16:42:48 CST 2022	Present	Present
23	Sat Oct 07 12:06:09 NPT 2023	Absent	Present

Fig:Student Attendance Table

CHAPTER-8

RESULT AND IMPORTANT OF UMS

7.1 Result

If University Management System is designed and executed perfectly, system can handle the vast and bulk of data of students, faculty, faculty members, academic session, admission session and staff details and information in arranged way. In other hand, data are secured and listed in database, so we can access the data by using only own private key(i.e, Passwords) and we can find the unique data at a specific time .

Overall, the project is secured and useful for university environment and can arrange data of students and staffs in centralized way, which saves our time for finding the specific data of specific students or staffs or about college fees structure or may be about college admission process and fees structure .

7.2 Important of UMS

Importance of UMS is given below :

1. It is very easy for the student to access information and educational service.
2. It provides transparency in all the processes.
3. It provides better communication between all the stakeholders of the university.
4. All the tasks are executed in a very simplifying matter.
5. The academic and admission process works have the management of the time-critical activity.

CHAPTER-9

CONCLUSION

We can manage the bulk numbers of data in specific and centralized ways. Due to management system, our daily life is being more easier for living . University Management System is project which keeps data and information of total university environment related cases and students as well as staffs in secured form. From where data can not be decrypt . For visualizing of data, we have own private key to access the information secured in database of management system.

We can conclude that, by use of management system, our data is safe and it is user friendly those who have access to database and keeps data separately and secured.

CHAPTER-10

FUTURE ENHACEMENTS

In future, management system is very important for every organizations, schools, colleges, universities. To keep data in secured and organized form, data must be kept in database which stores data in centralized form .

Management system may be for our daily life routines, may be for arrangement of data , may be used for journals, system may also be used for ticketing and billing system.

University Management System is future decoration of educational sector. University contains many more colleges which are affiliated with university, we can also keep details and information of that colleges. We can manage students attendance in managed and secured way .

We can manage our daily life in specific way, if we manage our stuffs and organization in centralized way .

REFERENCES

1. Yang, Heng-Li, and Hsiu-Hua Cheng. "Creativity of student information system projects: From the perspective of network embeddedness." *Computers & Education* 54.1 (2010): 209-221.

2. https://www.geniusedusoft.com/school-management-system/academics-management.html

3. https://itsourcecode.com/free-projects/java-projects/university-management-system-project-in-java-with-source-code/?fbclid=IwAR0YXlsB8P44H-0coU42Z7939KHEKU8BvfSZAtohHK7bvAHlAwSEbFN-6b4

4. Chat Gpt

5. https://www.freeprojectz.com/entity-relationship/university-management-system-er-diagram