A PROJECT REPORT ON

EMPLOYEE MANAGEMENT SYSTEM

Submitted in partial fulfillment of the Requirements
for the Degree of

MASTER OF COMPUTER APPLICATION

by

VISHAL GUPTA
(Univ. Roll No.: 1900290149111)

Under the Supervision of
Mr. Ankit Verma
Assistant Professor

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206
DR. A. P. J. ABDUL KALAM TECHNICAL UNIVERSITY
LUCKNOW (AUGUST ,2021)

DECLARATION

I hereby declare that the work presented in this report entitled “EMPLOYEE MANAGEMENT SYSTEM ", was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute.

I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources.

I affirm that no portion of my work is plagiarized, and the experiments and results reportedin the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name: Vishal Gupta

Roll.No.:1900290149111

Branch: MCA

Gupta

(Candidate Signature)

CERTIFICATE

Certified that **Vishal Gupta (**enrollment no 190029014005296) has carried out the project work presented in this thesis entitled **“EMPLOYEE MANAGEMENT SYSTEM”** for the award of **Master of Computer Application** from Dr. APJ Abdul Kalam Technical University, Lucknow under my supervision. The thesis embodies results of original work, and studies are carried out by the student herself and the contents of the thesis do not formthe basis for the award of any other degree to the candidate or to anybody else from this or any other University.

Mr. Ankit Verma **External Examiner**

(Internal Examiner)

Assistant Professor

Dept. of Computer Applications

KIET Group of Institutions, Ghaziabad

Date:

ACKNOWLEDGEMENT

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Mr. Ankit Verma** for his guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enoughto express my gratitude to **Dr. Ajay Kumar Shrivastava**, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot in many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kinds of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Vishal Gupta

Enrollment no-190029014005296

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION TO PROJECT

Employee Management system is an application that enables users to create and store Employee Records. The application also provides facilities of a payroll system which enables user to generate Pay slips too. This application is helpful to department ofthe organization which maintains data of employees related to an organization .

Java is a platform independent language. Its created applications can be used on a standalone machine as well as on distributed network. More over applications developed in java can be extended to Internet based applications.

Thus java was chosen as background to design this application.

Employee Management System is a distributed application, developed to maintain the details of employees working in any organization. It maintains the information about the personal details of their employees, also the details about the payroll system which enable to generate the payslip. The application is actually a suite of applications developed using Java.

It is simple to understand and can be used by anyone who is not even familiar with simple employees system. It is user friendly and just asks the user to follow step by step operations by giving him few options. It is fast and can perform many operations of a company.

earphones. For object recognition, deep learning is used and implemented using Raspberry pi. It recognizes the obstacles which are present in the dataset of the system and informs about it through earplugs. It detects all kinds of obstacles as it controls the peripheral components that alert the direction to the user by ticking on the forehead. The implemented system is fast, and easy to use and an innovative affordable solution to blind and visually impaired people.

1.1.1 OBJECTIVE OF PROJECT

In this world of growing technologies everything has been computerized. With large number of work opportunities the Human workforce has increased. Thus there is a need of a system which can handle the data of such a large number of Employees in an organization. This project simplifies the task of maintain records because of its user friendly nature Computer vision technologies, especially the deep convolutional neural network, have been rapidly developed in recent years. It is promising to use the state-of-art computer vision techniques to help people with vision loss. In this project, we want to explore the possibility of using the hearing sense to understand visual objects. The sense of sight and hearing sense share a striking similarity: both visual object and audio sound can be spatially localized. It is not often realized by many people that we can identify the spatial location of a sound source just by hearing it with two ears. In our project, we build a real-time object detection model.

1.2 PROJECT DESCRIPTION

Software Engineers have been trying various tools, methods and procedures to control the process of software development in order to build high quality software with high productivity. This method provides "how it is" for building the software while the tools provide automated or semi automated support for the methods. They are used in all stages of software development process, namely, planning, analysis, design, development and maintenance. The software development procedure integrates the methods and tools together and enables rational and timely development of the software system.

Visually impaired people find difficulties detecting and recognizing obstacles in front of them, during walking in the street, which makes it dangerous. The smart eye comes as a proposed solution to enable them to get notified when obstacles are around them. In this project we propose a solution, represented in a cap with ultrasonic sensor to detect obstacles and a servo motor attached with a needle through which direction of any obstacle can be notified to the user by ticking of needle on forehead, within a range of fourmeters. In this, a pi camera module is also mounted on the cap for the recognition of obstacles which later on inform the person through earplugs by converting text to sound. It will recognize the objects presented in its dataset and inform the user through earplugs.

1.2.1 Advantages of Project

a. Satisfy the user requirements

b. Be easy to understand by user and operator

c. Be easy to operate

d. Have a good user interface

e. Be easy to modify

f. Be expandable

g. Have adequate security control against the misuse of data

h. Handle the errors and exceptions satisfactorily

i. Be delivered on schedule within the budget

1.2.2 Disadvantages of Project

- Recognition of employee slow.
- It does not detect pit holes.
- It only recognizes objects which are present in the dataset.

1.3 PROJECT SCOPE

Nowadays, technology and human life cannot be separated as it has become the phenomenon of the world. But, how can technology help people that are visually impaired? [1] Blind people usually can estimate the obstacle in front of them without knowing the actual distance of the obstacle from them. Mobility for the blind people can be defined as mobility to move with safety and ease through the environment without relying on others. Most commonly mobility aid used by the blind are cane andguide dogs to facilitate their movement. But there are problems with this navigation support. The cane provides limited preview for the user and as a result, the user has to be more careful to walk and mobile very slowly. As for the guide dogs, the training and coordinating the dogs with blind people is a difficult task and the results are minimal. In order to overcome this problem, research on the assistant devices for theblind has been done by many people to help reduce the limited ability of the blind

people. The assistive headgear for the blind, isa device that can help visually impaired

1.4 HARDWARE / SOFTWARE

1.4.1 HARDWARE

Hardwarre	Configuration
Processorr	Intel Pentium G2030 clocked at 3.0 GHz
RAM	4GB DDR3
Monitor	Dell Backlit21” LED
Modem	Internet Connectivity
Keyboardd	Dell Standard 102 Keys & Optical Mouse

1.4.2 SOFTWARE

Software	Configuration
peratingSystem	s XP /7/8/10 &mac os
Software	rome, Microsoft Edge,Visual Studio Code

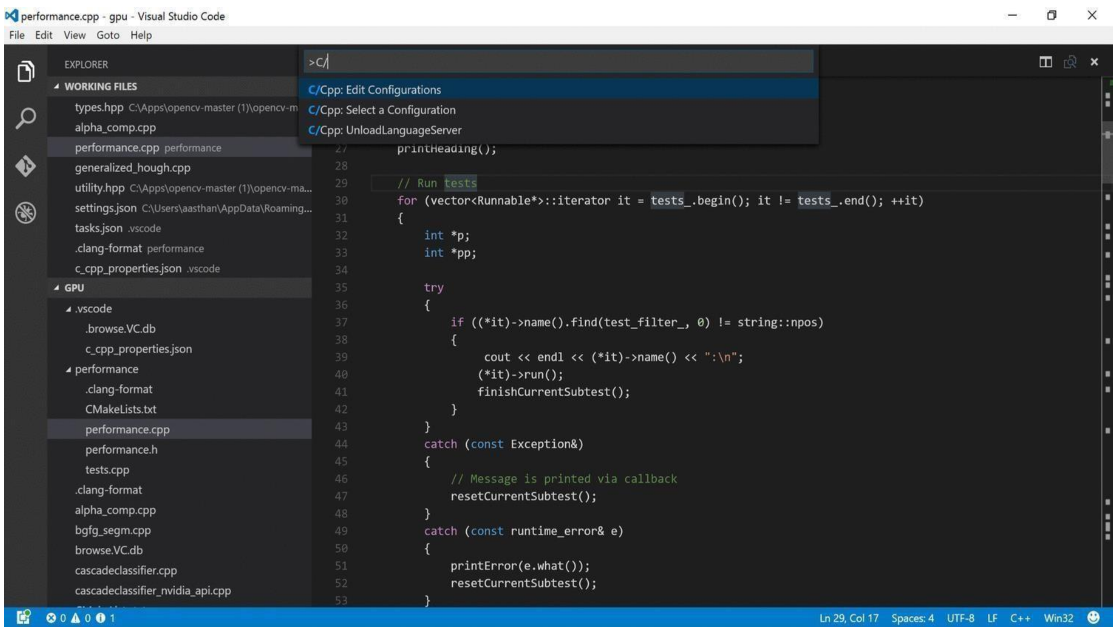
1.4.2.1 Visual Studio Code

Visual Studio Code is a source-code editor developed by Microsoft for Linux and macOS. It includes embedded Git and support for debugging, syntax highlighting, intelligent code completion, snippets, and code refactoring. It is highly customizable, allowing users to change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality. The source code is free and open- source, released under the permissive MIT License. The compiled binaries are freeware for any use.

Visual Studio Code is a source code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js and C++. It is based ont he Electron framework, which is used to develop Node.js web apps that run on the Blink Layout engine.Although it uses the Electron framework, the software does not use

Atom and instead employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

Instead of a project system, it allows users to open one or more directories, which can then be saved in workspaces for future reuse.


performance.cpp - gpu - Visual Studio Code
File
Edit
View
Goto
Help
EXPLORER
WORKING FILES
types.hpp C:\Apps\opencv-master (1)\opencv-m
alpha_comp.cpp
performance.cpp performance
generalized_hough.cpp
utility.hpp C:\Apps\opencv-master (1)\opencv-ma...
settings.json C:\Users\aasthan\AppData\Roaming...
tasks.json .vscode
.clang-format performance
c_cpp_properties.json .vscode
GPU
.vscode
.browse.VC.db
c_cpp_properties.json
performance
.clang-format
CMakeLists.txt
performance.cpp
performance.h
tests.cpp
.clang-format
alpha_comp.cpp
bgfg_segm.cpp
browse.VC.db
cascadeclassifier.cpp
cascadeclassifier_nvidia_api.cpp
>C/
C/Cpp: Edit Configurations
C/Cpp: Select a Configuration
C/Cpp: UnloadLanguageServer
⊗ 0 ⚠ 0 ⓘ 1
Ln 29, Col 17
Spaces: 4
UTF-8
LF
C++
Win32

```
    printHeading();

    // Run tests
    for (vector<Runnable*>::iterator it = tests_.begin(); it != tests_.end(); ++it)
    {
        int *p;
        int *pp;

        try
        {
            if ((*it)->name().find(test_filter_, 0) != string::npos)
            {
                 cout << endl << (*it)->name() << ":\n";
                (*it)->run();
                finishCurrentSubtest();
            }
        }
        catch (const Exception&)
        {
            // Message is printed via callback
            resetCurrentSubtest();
        }
        catch (const runtime_error& e)
        {
            printError(e.what());
            resetCurrentSubtest();
        }
```

Fig-1.9 Visual Studio Code

CHAPTER 2

LITERATURE REVIEW

FUNDAMENTAL

The Java programming created in 1995 by Sun Microsystems headed by James Gosling, It is a language is a machine-independent language. It means that a Java program written on a computer will run on any other computer even if the hardware configuration or the operating system is different.

Java is an Object Oriented language Most of the applications that we download on mobile phones for Internet games, are developed in Java.

It's a general purpose, class based and one of the most used programing .

FEATURES

*It's one of the easiest language to learn and use.

*Java enables high performance, with the use of Just-in-time compliers.

*It's designed to adapt to an evolving environment

*It's a portable language
.
*It has a security feature

*It's based on model object.

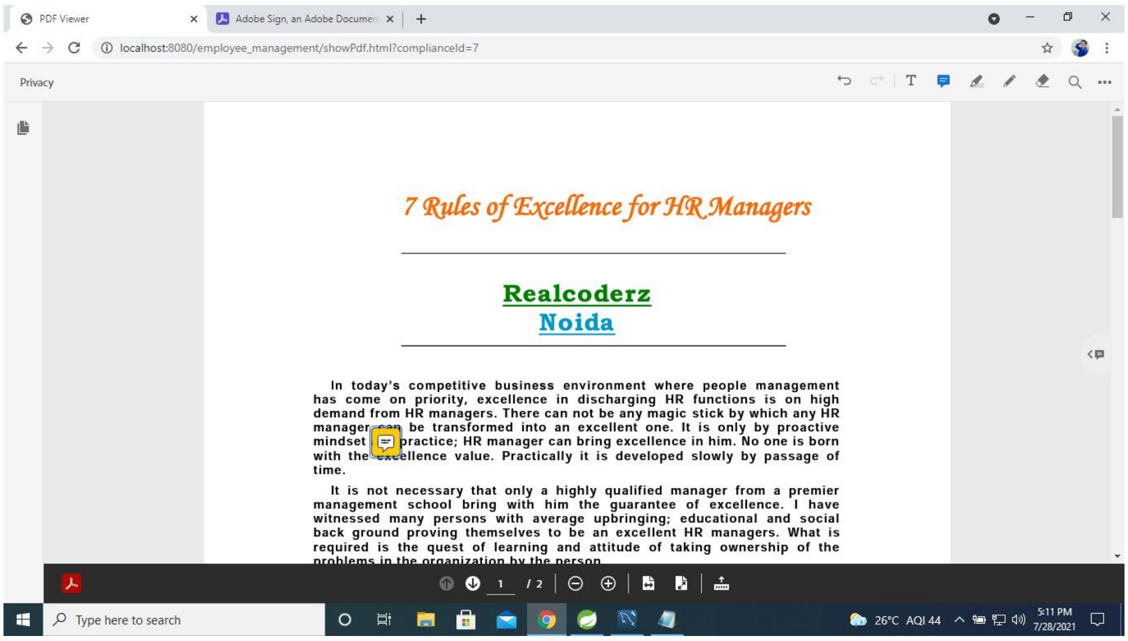
* It's a collection of programs .

*That helps programmers to run and develop Java.

* It is a set of specifications and software

COMPONENTS

A. JDK(Java development kit):This kit helps programmers to run Java programs.We can install more than one JDK in computer.
B. JVM (Java Virtual Machine); It converts Java bytecode into Machine language. It's an engine to drive a Java code.
C. RE(Java Runtime Environment)Programmers need JRE to run Java.It is designed to run other software.
D. Lists are similar to arrays in C/C++.
E. 1.Java Fx;it's used to develop rich internet applications.
F. 2.Java MicroEdition:it runs on small devices like phones.
G. 3.Java Enterprise Edition;It is use to develop large scale applications.
H. 4.Java Standard Edition;It is used for networking, and GUI
I. It's one of the easiest language to learn and use.
J. Java enables high performance,with the use of Just-in-time compliers.
K. It's designed to adapt to an evolving environment
L. It's a portable language.
M. It has a security feature
N. It's based on model object.


PDF Viewer
Adobe Sign, an Adobe Documen
localhost:8080/employee_management/showPdf.html?complianceId=7
Privacy
1 / 2
Type here to search
26°C AQI 44
5:11 PM
7/28/2021

7 Rules of Excellence for HR Managers

Realcoderz
Noida

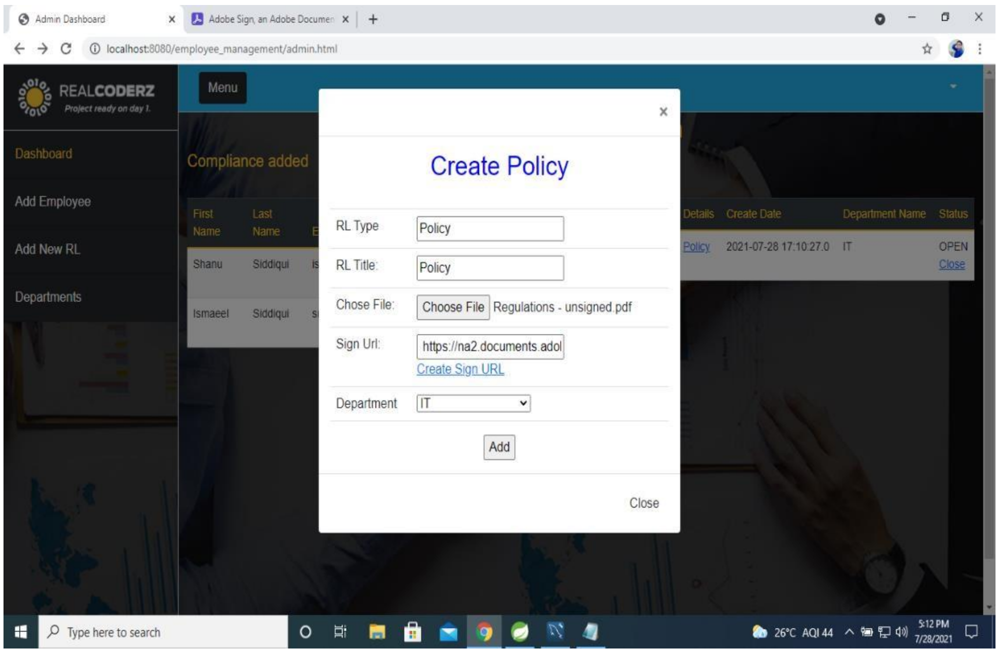
In today's competitive business environment where people management has come on priority, excellence in discharging HR functions is on high demand from HR managers. There can not be any magic stick by which any HR manager can be transformed into an excellent one. It is only by proactive mindset practice; HR manager can bring excellence in him. No one is born with the excellence value. Practically it is developed slowly by passage of time.

It is not necessary that only a highly qualified manager from a premier management school bring with him the guarantee of excellence. I have witnessed many persons with average upbringing; educational and social back ground proving themselves to be an excellent HR managers. What is required is the quest of learning and attitude of taking ownership of the problems in the organization by the person

Figure 1Showing Employee Pdf

JAVASCRIPT ADVANTAGE

- **It is a special flexible and powerful language.**
- **It is a high level programming.**
- **It is used for several websites for scripting the web pages.**



First Name	Last Name	…	Details	Create Date	Department Name	Status
Shanu	Siddiqui	is	Policy	2021-07-28 17:10:27.0	IT	OPEN Close
Ismaeel	Siddiqui	s				

Create Policy

RL Type	Policy
RL Title:	Policy
Chose File:	Choose File Regulations - unsigned.pdf
Sign Url:	https://na2.documents.adol Create Sign URL
Department	IT

Figure 2Employee Sign Up

The real power of Programming lies in the liberal usage of its data structure. The common criticism of Arduino is that it is slow compared to C/C++. This is especially true if multiple for-loops are used in programming.. Hence for-loops are not the most efficient way for programming. The alternative is to use data structures such as lists, tuples, dictionary and sets. We describe each of these data structures in this section

Data Structure: In data structures, we organize data stored in a computer in such a way so that we can use it effectively. Use of data structures in programming gives added advantage to it. Most of the programmers would like to use C++ in comparison to Arduino because adeno is slow. In spite of using multiple for loops, we should use data structures such as tuples, sets, lists and dictionary. Following is the description of each of these data structures:

Lists: A list is far more better than array which can store only the same data type. We use lists to store multiple items in a single variable. List items may be ordered or changeable. They may have duplicate items also. Lists are created using square bracket [ ] containing list items separated by comma. We can easily add or remove any item in a list. A list may contain another list also. All this makes lists very powerful and flexible. We can understand this as described below

List functions: following list functions are there:

a) **Index ( )** : Elements are indexed from 0. We can access an element by mentioning it's index enclosed in [ ].

b) **Append( )** : Used to add an element at the end of a list.

c) **List Copy ( )** : We use it to get a shallow copy of a list.

d) **List Clear ( )** : We can this function to remove all list items.

e) **List Count ( )** : It tells us number of elements in a list.

f) **List Extend ( )** : We use it to add iterable elements to the end of a list.

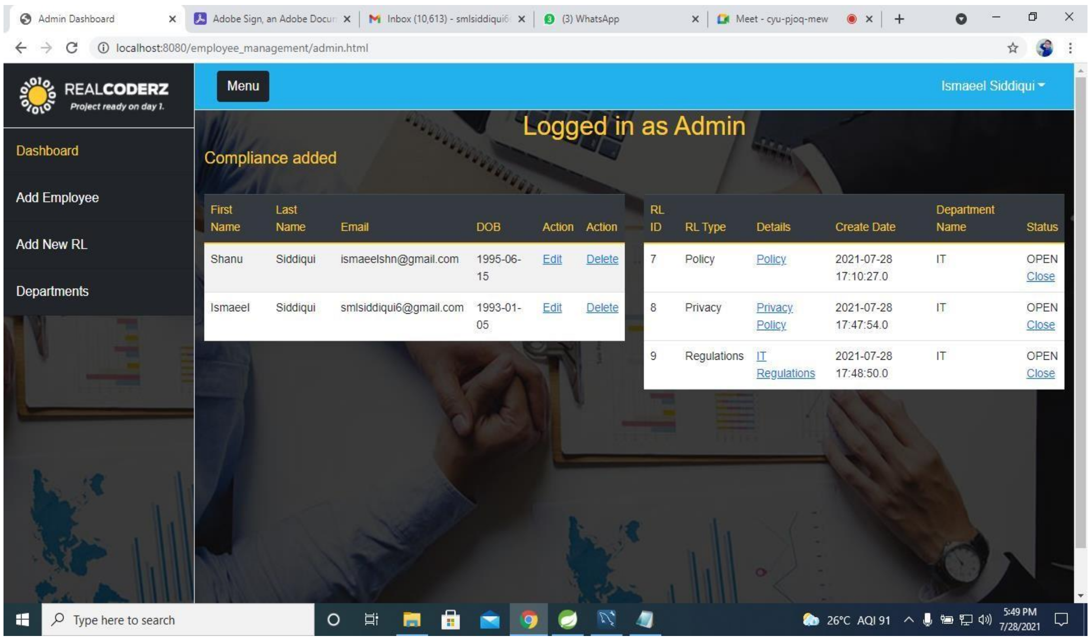
g) **List Insert ( )** : To add an element in a list at a particular position.

h) **List Remove ( )** : To remove elements from a list.

i) **List Pop ( )** : It is used to remove element at a particular index.

j) **List Sort ( )** : To arrange elements in a particular order.

k) **List Reverse ( )** : It will reverse all list elements.



Logged in as Admin

Compliance added

First Name	Last Name	Email	DOB	Action	Action
Shanu	Siddiqui	ismaeelshn@gmail.com	1995-06-15	Edit	Delete
Ismaeel	Siddiqui	smlsiddiqui6@gmail.com	1993-01-05	Edit	Delete

RL ID	RL Type	Details	Create Date	Department Name	Status
7	Policy	Policy	2021-07-28 17:10:27.0	IT	OPEN Close
8	Privacy	Privacy Policy	2021-07-28 17:47:54.0	IT	OPEN Close
9	Regulations	IT Regulations	2021-07-28 17:48:50.0	IT	OPEN Close

Figure 3Admin Dashboard

A set is an unordered collection of objects. A set can contain objects of any data type. To create a set, we need to use the function set. We create a set from a list containing four values. Then we create a set containing a tuple. The elements of a set need to be unique. Hence when the content of s2 is printed, we notice that the duplicates have been eliminated. Sets can be operated using many of the common mathematical operations on sets such as union, intersection, set difference, symmetric difference etc.

CHAPTER 3

FEASIBILITY STUDY

3.1 TECHNICAL FEASIBILITY

A large part of determining resources has to do with assessing technical feasibility.The technical requirements considered in the proposed project are as this project is basically built for blind people so the basic requirement is to generate an environment forthe person through the help of the project so that he can detect and recognize the obstacleson his way. The technical work which is done in project Object Recognition forVisually Impaired People is considered technically feasible because its internal technical capability is sufficient to support the project requirements.

Here we are using an ultrasonic sensor with an Arduino microcontroller and pi camera withRaspberry pi with some set of instructions. These instructions are all about obstacle detection as well as recognition and providing info to the user through earplugs of the user and direction by ticking on the forehead. This project will use 4 major hardware components i.e. Arduino UNO, Pi camera, Raspberry pi and ultrasonic sonar.

3.2 OPERATIONAL FEASIBILITY

Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented.

Operational feasibility is a measure of how well a proposed system solves the problems and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

Operational feasibility reviews the willingness of the organization to support the proposed system. This is probably the most difficult of the feasibility to gauge. In order to determine this feasibility, it is important to understand the management commitment the proposed project.

When we enable this device on the head, it will detect any hurdle or obstacle which comes in its path, and informs the direction to the user by ticking on the forehead and informs what type of obstacle is present through earplugs, so that they can take further action to avoid them.

3.3 BEHAVIORAL FEASIBILITY

Behavioral Feasibility includes how strong the reaction of users will be towards the development of a new system that involves an embedded system to be used in their dailylife. So resistant to change is identified. It is a device which is fixed on the head, Visual challenge is a global problem among people and a large population is suffering due to thisproblem. The sense of dependency on someone else hurts a lot. Here we are building technical solution to ease this pain.

3.4 ECONOMICAL FEASIBILITY

Smart headgear is economically very feasible, and accurate results because we are trying to make it at minimal cost so that the needy can afford it. It will use a minimal amount ofpower for its working which will be provided by a 9v battery or users can use other sourcesof power conveniently.

CHAPTER 4

CODING

```
package
employeemanagement; import
java.sql.Connection;     import
java.sql.DriverManager;
import
java.sql.PreparedStatement;
import java.sql.ResultSet;
import
java.sql.SQLException;
import        java.sql.Statement;
import java.util.logging.Level;
import
java.util.logging.Logger;
public class AddEmp extends javax.swing.JFrame
{Connection con;
Statement              stmt;
PreparedStatement   stat;
ResultSet rs;
static int eid=0;

/** Creates new form AddEmp */

public AddEmp() throws ClassNotFoundException, SQLException {
initComponents();
this.setVisible(true);     Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con=DriverManager.getConnection("jdbc:odbc:Employee");
stmt=con.createStatement();


stmt=con.createStatement(rs.TYPE_SCROLL_SENSITIVE,rs.CONCUR_READ_ON
L Y);
rs=stmt.executeQuery("select *  from  emp");
rs.last();
eid=rs.getInt(10);
```

```
 }
@SuppressWarnings("unchecked")
 // <editor-fold defaultstate="collapsed" desc="Generated Code">
 private void initComponents() {


    label1  =  new  java.awt.Label();
    label2  =  new  java.awt.Label();
    label3  =  new  java.awt.Label();
    label4  =  new  java.awt.Label();
    label5  =  new  java.awt.Label();
    label6  =  new  java.awt.Label();
    label7  =  new  java.awt.Label();
    label8  =  new  java.awt.Label();
    label9  =  new  java.awt.Label();
    label10 = new java.awt.Label();
    label11 = new java.awt.Label();
    textField1  =  new  java.awt.TextField();
    textField2  =  new  java.awt.TextField();
    textField8  =  new  java.awt.TextField();
    textField9  =  new  java.awt.TextField();
    textField10 = new java.awt.TextField();
    textField11 = new java.awt.TextField();
    textField12 = new java.awt.TextField();
    textField13 = new java.awt.TextField();
    textField14 = new java.awt.TextField();
    button1    =    new    java.awt.Button();
    button2    =    new    java.awt.Button();
    label12     =     new     java.awt.Label();
    button3 = new java.awt.Button();


    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    label1.setText("NAME");
    label2.setText("ADDRESS");
    label3.setText("SALARY");
```

```
         label4.setText("POST");
        label5.setText("EXPERIANCE");
        label6.setText("Allowences");
        label7.setText("HOUSE");
        label8.setText("TRAVELLING");
        label9.setText("Deductions");
        label10.setText("PROVIDENT          FUND");
        label11.setText("INCOME                    TAX");
        button1.setLabel("SUBMIT");
         button1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button1ActionPerformed(evt);

}

});
         button2.setLabel("CLEAR");
         button3.setLabel("BACK");
         button3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button3ActionPerformed(evt);

}

});


         javax.swing.GroupLayout                    layout                    =                    new
  javax.swing.GroupLayout(getContentPane());
         getContentPane().setLayout(layout);
         layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()


  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
  )
                  .addGroup(layout.createSequentialGroup()
                      .addGap(46, 46, 46)

  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
  )
                        .addComponent(label1,
  javax.swing.GroupLayout.PREFERRED_SIZE,                                          116,
  javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label5,
  javax.swing.GroupLayout.PREFERRED_SIZE,
  javax.swing.GroupLayout.DEFAULT_SIZE,
  javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label2,
  javax.swing.GroupLayout.PREFERRED_SIZE,
  javax.swing.GroupLayout.DEFAULT_SIZE,
  javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label4,
  javax.swing.GroupLayout.PREFERRED_SIZE,
  javax.swing.GroupLayout.DEFAULT_SIZE,
  javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label3,
  javax.swing.GroupLayout.PREFERRED_SIZE,
  javax.swing.GroupLayout.DEFAULT_SIZE,
  javax.swing.GroupLayout.PREFERRED_SIZE))

  .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

  .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
  )
                        .addComponent(textField1,
  javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField2,
  javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField13,
  javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
```

```
                        .addComponent(textField8,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField14,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField9,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField10,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField11,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addComponent(textField12,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                        .addGroup(layout.createSequentialGroup()

                            .addComponent(button1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)


.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,          24,
Short.MAX_VALUE)
                            .addComponent(button3,
javax.swing.GroupLayout.PREFERRED_SIZE,                                       77,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(23, 23, 23)

                            .addComponent(button2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(18, 18, 18))))

                    .addGroup(layout.createSequentialGroup()

                        .addContainerGap()

                        .addComponent(label6,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                    .addGroup(layout.createSequentialGroup()
                        .addGap(43, 43, 43)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                            .addComponent(label7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(label8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(label9,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(45, 45, 45)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                            .addComponent(label11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(label10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(label12,
javax.swing.GroupLayout.PREFERRED_SIZE,                                   169,
javax.swing.GroupLayout.PREFERRED_SIZE))))
```

```
            .addContainerGap(71, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
          layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(layout.createSequentialGroup()
            .addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
, false)
              .addGroup(layout.createSequentialGroup()
                .addComponent(label1,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(label2,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(label4,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
              .addGroup(layout.createSequentialGroup()
                .addComponent(textField1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(textField2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(textField13,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label5,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label3,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(textField8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(textField14,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILIN
G)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label6,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(2, 2, 2)
                    .addComponent(label7,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addComponent(textField9,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
, false)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label8,    javax.swing.GroupLayout.PREFERRED_SIZE,
20, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label9,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label10,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(textField10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addComponent(textField11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addComponent(label11,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(textField12,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(34, 34, 34)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(button1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addGap(19, 19, 19)
                    .addComponent(label12,   javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                    .addComponent(button2,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(button3,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(21, Short.MAX_VALUE))
        );


        pack();
    }// </editor-fold>


private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
            try {
// TODO add your handling code here:
                stat = con.prepareStatement("Insert into Emp values(?,?,?,?,?,?,?,?,?)");
            }         catch         (SQLException         ex)         {
                Logger.getLogger(AddEmp.class.getName()).log(Level.SEVERE, null, ex);
            }
            String   name   =   textField1.getText();
            String   address   =   textField2.getText();
            String   post   =   textField13.getText();
            String experiance = textField8.getText();
            String   salary   =   textField14.getText();
            String   house   =   textField9.getText();
            String travelling = textField10.getText();
            String pf = textField11.getText();
            String it = textField12.getText();
            stat.setString(1,        name);
            stat.setString(2,      address);
            stat.setString(3, post);
            stat.setInt(4,Integer.parseInt(experiance));
            stat.setInt(5,Integer.parseInt(salary));
            stat.setInt(6,Integer.parseInt(house));
            stat.setInt(7,Integer.parseInt(travelling));
            stat.setInt(8,Integer.parseInt(pf));
            stat.setInt(9,Integer.parseInt(it));
            stat.setInt(10,++eid);
            label12.setText("Success");
            stat.executeUpdate();
            con.close();
            this.setVisible(false);
            Menuscrn menu = new Menuscrn();
            menu.setVisible(true);
        }       catch       (SQLException       ex)       {
            Logger.getLogger(AddEmp.class.getName()).log(Level.SEVERE, null, ex);
        }



}


private void button3ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
    this.setVisible(false);
    Menuscrn menu = new Menuscrn();
    menu.setVisible(true);
}
    public    static    void    main(String    args[])    {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    new AddEmp().setVisible(true);
                }     catch     (ClassNotFoundException     ex)     {
                    Logger.getLogger(AddEmp.class.getName()).log(Level.SEVERE, null, ex);
                }     catch     (SQLException     ex)     {
                    Logger.getLogger(AddEmp.class.getName()).log(Level.SEVERE, null, ex);
                }
            }
        });
    }

    // Variables declaration - do not modify
    private java.awt.Button button1;
    private java.awt.Button button2;
    private java.awt.Button button3;
    private java.awt.Label label1;
    private java.awt.Label label10;
    private java.awt.Label label11;
    private java.awt.Label label12;
    private java.awt.Label label2;
    private java.awt.Label label3;
    private java.awt.Label label4;
    private java.awt.Label label5;
    private java.awt.Label label6;
    private java.awt.Label label7;
    private java.awt.Label label8;
    private java.awt.Label label9;
    private java.awt.TextField textField1;
    private java.awt.TextField textField10;
    private java.awt.TextField textField11;
    private java.awt.TextField textField12;
    private java.awt.TextField textField13;
    private java.awt.TextField textField14;
    private java.awt.TextField textField2;
    private java.awt.TextField textField8;
    private java.awt.TextField textField9;
    // End of variables declaration

}
```

```
/*
 * Intro.java
 */



package employeemanagement;


public class Intro extends javax.swing.JFrame {



    /** Creates new form Intro */
    public Intro() {
        initComponents();
        textField2.setEchoChar('*');
        this.setVisible(true);
    }


    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {


        label1 = new java.awt.Label();
        label2 = new java.awt.Label();
        label3 = new java.awt.Label();
        textField1 = new java.awt.TextField();
        textField2 = new java.awt.TextField();
        button1    =    new    java.awt.Button();
        button2    =    new    java.awt.Button();
        label4 = new java.awt.Label();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setBackground(new java.awt.Color(51, 0, 51));

        label1.setBackground(new java.awt.Color(0, 153, 0));

        label1.setFont(new java.awt.Font("Arial", 3, 18)); // NOI18N
        label1.setText("WELCOME TO EMPLOYEE MANAGEMENT SYSTEM");

        label2.setText("User Name");
        label3.setText("Password");
        button1.setLabel("Submit");

        button1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button1ActionPerformed(evt);
}
});

        button2.setLabel("Reset");

        javax.swing.GroupLayout layout = new
    javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
    layout.createSequentialGroup()
                .addGap(96, 96, 96)
                .addComponent(label1, javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addGap(67, 67, 67))
            .addGroup(layout.createSequentialGroup()
                .addGap(161, 161, 161)
    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
    )
                        .addComponent(label2,        javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label3,        javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(38, 38, 38)


    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
    )
                        .addComponent(textField2,      javax.swing.GroupLayout.DEFAULT_SIZE,
    166, Short.MAX_VALUE)
                        .addComponent(textField1,      javax.swing.GroupLayout.DEFAULT_SIZE,
    166, Short.MAX_VALUE))
                .addGap(178, 178, 178))
            .addGroup(layout.createSequentialGroup()
                .addGap(206, 206, 206)
                .addComponent(button1,         javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(66, 66, 66)
                .addComponent(button2,         javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(235, Short.MAX_VALUE))
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
    layout.createSequentialGroup()
                .addContainerGap(62, Short.MAX_VALUE)
                .addComponent(label4,    javax.swing.GroupLayout.PREFERRED_SIZE,    511,
    javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(38, 38, 38))
```

```
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(label1,   javax.swing.GroupLayout.PREFERRED_SIZE, 86,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(41, 41, 41)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(textField1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(textField2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label2,   javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label3,   javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addGap(39, 39, 39)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addComponent(button1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(button2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
            .addGap(22, 22, 22)
            .addComponent(label4, javax.swing.GroupLayout.PREFERRED_SIZE, 39,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addContainerGap())
    );


    pack();
  }// </editor-fold>


private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:


if((textField1.getText().equals("Admin"))&&(textField2.getText().equals("adminadmin
")))
  {
    label4.setText("");
    this.setVisible(false);
    Menuscrn menu = new Menuscrn();
    menu.setVisible(true);
  }
  else
  label4.setText("Wrong Password/Username");
}


  /**
```

```
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new RunnableImpl());
    }


    // Variables declaration - do not modify
    private java.awt.Button button1;
    private java.awt.Button button2;
    private java.awt.Label label1;
    private java.awt.Label label2;
    private java.awt.Label label3;
    private java.awt.Label label4;
    private java.awt.TextField textField1;
    private java.awt.TextField textField2;
    // End of variables declaration
    private static class RunnableImpl implements Runnable {
        public RunnableImpl() {
        }


        public void run() {
            new Intro().setVisible(true);




/*
 * Menuscrn.java
 */
 package employeemanagement;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import   java.sql.SQLException;
import         java.sql.Statement;
import    java.util.logging.Level;
import java.util.logging.Logger;


public class Menuscrn extends javax.swing.JFrame {
        Connection con;
        Statement            stmt;
        PreparedStatement stat;
        ResultSet rs;


    /** Creates new form Menuscrn */
    public Menuscrn() {
        try {
            initComponents();
            this.setDefaultCloseOperation(DO_NOTHING_ON_CLOSE);
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:Employee");
            stmt=con.createStatement();
            rs=stmt.executeQuery("select * from Emp" );
        }      catch      (SQLException      ex)      {
            if(ex.getErrorCode()==208)
            {
                try {
                    rs = stmt.executeQuery("CREATE TABLE [dbo].[Emp]([Name] [char](20)
NOT    NULL,[Address]    [char](30)    NOT    NULL,[Post]    [char](10)    NOT
NULL,[Experiance] [int] NULL,[Salary] [int] NOT NULL,[HRA] [int] NULL,[TA] [int]
NULL,[PF] [int] NULL,[ITax] [int] NULL,[Eid] [int] NOT NULL) ON [PRIMARY]");
                }           catch           (SQLException           ex1)           {
                    Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE,      null,
ex1);
                }
            }
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }             catch             (ClassNotFoundException             ex)             {
          Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }
    }


    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        button5 = new java.awt.Button();
        button6 = new java.awt.Button();
        button3 = new java.awt.Button();
        button1 = new java.awt.Button();
        button2 = new java.awt.Button();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosed(java.awt.event.WindowEvent evt) {
                formWindowClosed(evt);
            }
            public void windowClosing(java.awt.event.WindowEvent evt) {
                formWindowClosing(evt);
            }
        });

        button5.setLabel("GENERATE                              SLIP");
        button5.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button5ActionPerformed(evt);
}

});


        button6.setLabel("EXIT");

        button6.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button6ActionPerformed(evt);

}

});


        button3.setLabel("VIEW                              DETAILS");
        button3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button3ActionPerformed(evt);
            }

        });


        button1.setLabel("EMPLOYEE                          LIST");
        button1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button1ActionPerformed(evt);
}

});


        button2.setLabel("ADD                               EMPLOYEE");
        button2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button2ActionPerformed(evt);
}

});


        javax.swing.GroupLayout            layout            =            new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(button2,     javax.swing.GroupLayout.DEFAULT_SIZE,     224,
Short.MAX_VALUE)
            .addComponent(button1,     javax.swing.GroupLayout.DEFAULT_SIZE,     224,
Short.MAX_VALUE)
            .addComponent(button3,        javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 224, Short.MAX_VALUE)
            .addComponent(button5,     javax.swing.GroupLayout.DEFAULT_SIZE,     224,
Short.MAX_VALUE)
            .addComponent(button6,     javax.swing.GroupLayout.DEFAULT_SIZE,     224,
Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addComponent(button2, javax.swing.GroupLayout.PREFERRED_SIZE,  46,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(1, 1, 1)
                .addComponent(button1, javax.swing.GroupLayout.PREFERRED_SIZE, 45,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(1, 1, 1)
                .addComponent(button3, javax.swing.GroupLayout.PREFERRED_SIZE,  51,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(2, 2, 2)
                .addComponent(button5, javax.swing.GroupLayout.PREFERRED_SIZE, 49,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(2, 2, 2)
                .addComponent(button6, javax.swing.GroupLayout.PREFERRED_SIZE,  46,
javax.swing.GroupLayout.PREFERRED_SIZE))
        );
        pack();
    }// </editor-fold>

private void button5ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
         EmpSlip emp = new EmpSlip();
}

private void button2ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
// TODO add your handling code here:
            this.setVisible(false);
            AddEmp emp = new AddEmp();
        }           catch           (ClassNotFoundException           ex)           {
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }           catch           (SQLException           ex)           {
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }
}

private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
// TODO add your handling code here:
            this.setVisible(false);
            EmpList emp = new EmpList();
        }           catch           (ClassNotFoundException           ex)           {
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }           catch           (SQLException           ex)           {
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }
}

private void button3ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
```

```
// TODO add your handling code here:
            this.setVisible(false);
            EmpDetail emp = new EmpDetail();
        }               catch               (ClassNotFoundException               ex)               {
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }               catch               (SQLException               ex)               {
            Logger.getLogger(Menuscrn.class.getName()).log(Level.SEVERE, null, ex);
        }

}


private void button6ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
 System.exit(0);
}


private void formWindowClosed(java.awt.event.WindowEvent evt) {
// TODO add your handling code here:


}


private void formWindowClosing(java.awt.event.WindowEvent evt) {
// TODO add your handling code here:


}


    /**
    * @param args the command line arguments
    */
    public     static     void     main(String     args[])     {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new Menuscrn().setVisible(true);
}
});
```

```
/*
 * EmpList.java
 */
 package employeemanagement;


import          java.sql.Statement;
import       java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import   java.sql.SQLException;
import    java.util.logging.Level;
import java.util.logging.Logger;
public class EmpList extends javax.swing.JFrame {
    Connection con;
    Statement             stmt;
    PreparedStatement  stat;
    ResultSet rs;
    /** Creates new form EmpList */
    public EmpList() throws ClassNotFoundException, SQLException {
        initComponents();
        this.setVisible(true);
        textArea1.setEditable(false);
        Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
        con=DriverManager.getConnection("jdbc:odbc:Employee");
        stmt=con.createStatement();
        rs=stmt.executeQuery("Select              *             from             emp");
        textArea1.append("Name\t\tAddress\t\t\tPost\t\tEMPID\n");
        while(rs.next())
        {
            textArea1.append(rs.getString(1)+"\t");
            textArea1.append(rs.getString(2)+"\t");
            textArea1.append(rs.getString(3)+"\t");
            textArea1.append(Integer.toString(rs.getInt(10))+"\t");
            textArea1.append("\n");
          }
        }


        /** This method is called from within the constructor to
         * initialize the form.
         * WARNING: Do NOT modify this code. The content of this method is
         * always regenerated by the Form Editor.
         */
        @SuppressWarnings("unchecked")
        // <editor-fold defaultstate="collapsed" desc="Generated Code">
        private void initComponents() {

          menuBar1 = new java.awt.MenuBar();
          menu1    =    new    java.awt.Menu();
          textArea1 = new java.awt.TextArea();
          button1 = new java.awt.Button();

          menu1.setLabel("Menu");
          menuBar1.add(menu1);

          setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);


          button1.setLabel("Back");
          button1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
              button1ActionPerformed(evt);
    }
    });


          javax.swing.GroupLayout          layout          =          new
      javax.swing.GroupLayout(getContentPane());
          getContentPane().setLayout(layout);
          layout.setHorizontalGroup(
          layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addComponent(textArea1,    javax.swing.GroupLayout.DEFAULT_SIZE,    723,
Short.MAX_VALUE)
          .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
             .addContainerGap(328, Short.MAX_VALUE)
             .addComponent(button1, javax.swing.GroupLayout.PREFERRED_SIZE,  95,
javax.swing.GroupLayout.PREFERRED_SIZE)
             .addGap(300, 300, 300))
       );
       layout.setVerticalGroup(
          layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
          .addGroup(layout.createSequentialGroup()
             .addComponent(textArea1,      javax.swing.GroupLayout.PREFERRED_SIZE,
439, javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,           14,
Short.MAX_VALUE)
             .addComponent(button1,        javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
             .addContainerGap())
       );


       pack();
   }// </editor-fold>


private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   this.setVisible(false);
   Menuscrn menu = new Menuscrn();
   menu.setVisible(true);
}
```

```
    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    new EmpList().setVisible(true);
                } catch (ClassNotFoundException ex) {
                    Logger.getLogger(EmpList.class.getName()).log(Level.SEVERE, null, ex);
                } catch (SQLException ex) {
                    Logger.getLogger(EmpList.class.getName()).log(Level.SEVERE, null, ex);
                }
}
});
    }


    // Variables declaration - do not modify
    private java.awt.Button button1;
    private java.awt.Menu menu1;
    private java.awt.MenuBar menuBar1;
    private java.awt.TextArea textArea1;
    // End of variables declaration


  }
```

```
/*
 * EmpDetail.java
 */
package employeemanagement;
import      java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import  java.sql.SQLException;
import       java.sql.Statement;
import   java.util.logging.Level;
import java.util.logging.Logger;


public class EmpDetail extends javax.swing.JFrame {
    Connection con;
    Statement       stmt;
    PreparedStatement stat;
    ResultSet rs;
    int i=0;
    //int selected_id;
  /** Creates new form EmpDetail */
  public    EmpDetail()    throws    ClassNotFoundException,    SQLException    {
    initComponents();
    this.setVisible(true);
    textField1.setEditable(false);
    textField2.setEditable(false);
    textField13.setEditable(false);
    textField8.setEditable(false);
    textField14.setEditable(false);
    textField9.setEditable(false);
    textField10.setEditable(false);
    textField11.setEditable(false);
    textField12.setEditable(false);
    button1.setEnabled(false);
    button2.setEnabled(false);
    Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
    con=DriverManager.getConnection("jdbc:odbc:Employee");
    stmt=con.createStatement();
    rs=stmt.executeQuery("Select * from emp");
    while(rs.next())
    {
        choice1.insert(Integer.toString(rs.getInt(10)), i);
    }


}


/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

    label1 = new java.awt.Label();
    label5 = new java.awt.Label();
    label2 = new java.awt.Label();
    label4 = new java.awt.Label();
    label3 = new java.awt.Label();
    textField1  =  new  java.awt.TextField();
    textField2  =  new  java.awt.TextField();
    textField13 = new java.awt.TextField();
    textField8  =  new  java.awt.TextField();
    textField14 = new java.awt.TextField();
    textField9  =  new  java.awt.TextField();
    textField10 = new java.awt.TextField();
    textField11 = new java.awt.TextField();
```

```
            textField12 = new java.awt.TextField();
            button1 = new java.awt.Button();
            label6  =  new  java.awt.Label();
            label7  =  new  java.awt.Label();
            label8  =  new  java.awt.Label();
            label9  =  new  java.awt.Label();
            label11  =  new  java.awt.Label();
            label10  =  new  java.awt.Label();
            label12  =  new  java.awt.Label();
            button2 = new java.awt.Button();
            button3 = new java.awt.Button();
            choice1 = new java.awt.Choice();
            label13  =  new  java.awt.Label();
            button4 = new java.awt.Button();


            setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
            label1.setText("NAME");
            label5.setText("EXPERIANCE");
            label2.setText("ADDRESS");
            label4.setText("POST");
            label3.setText("SALARY");
            button1.setLabel("EDIT");
            button1.addActionListener(new java.awt.event.ActionListener() {
               public void actionPerformed(java.awt.event.ActionEvent evt) {
                  button1ActionPerformed(evt);
   }
   });
```

```
        label6.setText("Allowences");


        label7.setText("HOUSE");
        label8.setText("TRAVELLING");
        label9.setText("Deductions");
        label11.setText("INCOME          TAX");
        label10.setText("PROVIDENT FUND");
        button2.setLabel("DELETE");
        button2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button2ActionPerformed(evt);
}
});


        button3.setLabel("BACK");
        button3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button3ActionPerformed(evt);
}
});
        label13.setText("EMPID");
        button4.setLabel("OK");
        button4.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button4ActionPerformed(evt);
}
});
```

```
        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                    .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                            .addGroup(layout.createSequentialGroup()
                                .addGap(46, 46, 46)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                                    .addComponent(label1,
javax.swing.GroupLayout.PREFERRED_SIZE, 116,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(label5,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(label2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                    .addComponent(label4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                              .addComponent(label3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                        .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
layout.createSequentialGroup()
                           .addGap(83, 83, 83)
                           .addComponent(button1,
javax.swing.GroupLayout.DEFAULT_SIZE, 83, Short.MAX_VALUE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
, false)
.addGroup(layout.createSequentialGroup()

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
, false)
                              .addComponent(textField1,
javax.swing.GroupLayout.DEFAULT_SIZE, 258, Short.MAX_VALUE)
                              .addComponent(textField2,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                              .addComponent(textField13,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                              .addComponent(textField8,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                              .addComponent(textField14,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                              .addComponent(textField9,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                  .addComponent(textField10,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                  .addComponent(textField11,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                  .addComponent(textField12,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                          .addGroup(layout.createSequentialGroup()
                              .addGap(45, 45, 45)
                              .addComponent(button2,
javax.swing.GroupLayout.PREFERRED_SIZE,                                78,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                              .addComponent(button4,
javax.swing.GroupLayout.PREFERRED_SIZE,                                88,
javax.swing.GroupLayout.PREFERRED_SIZE))))
                  .addGroup(layout.createSequentialGroup()
                      .addContainerGap()
                      .addComponent(label6,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                  .addGroup(layout.createSequentialGroup()
                      .addGap(43, 43, 43)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                          .addComponent(label7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                 .addGroup(layout.createSequentialGroup()
                    .addContainerGap()
                    .addComponent(label9,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                 .addGroup(layout.createSequentialGroup()
                    .addGap(45, 45, 45)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                        .addComponent(label11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(label10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,         279,
Short.MAX_VALUE)))

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                 .addGroup(layout.createSequentialGroup()
                    .addGap(18, 18, 18)
                    .addComponent(label13,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                    .addGap(22, 22, 22)
                    .addComponent(choice1, javax.swing.GroupLayout.PREFERRED_SIZE,
70, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addGap(36, 36, 36)
                    .addComponent(button3, javax.swing.GroupLayout.PREFERRED_SIZE,
82, javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addContainerGap())
        .addGroup(layout.createSequentialGroup()
            .addGap(69, 69, 69)
            .addComponent(label12, javax.swing.GroupLayout.DEFAULT_SIZE, 244,
Short.MAX_VALUE)
            .addGap(282, 282, 282))
    );
    layout.setVerticalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
, false)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(label1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(label2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                              .addComponent(label4,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                          .addGroup(layout.createSequentialGroup()
                              .addComponent(textField1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                              .addComponent(textField2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                              .addComponent(textField13,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                          .addGroup(layout.createSequentialGroup()
                              .addComponent(label5,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(label3,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(textField8,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(textField14,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILIN
G)
                            .addGroup(layout.createSequentialGroup()
                                .addComponent(label6,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addGap(2, 2, 2)
                                .addComponent(label7,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                        .addComponent(textField9,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
, false)
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(label8,
javax.swing.GroupLayout.PREFERRED_SIZE,                                    20,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(label9,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(label10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(layout.createSequentialGroup()
                            .addComponent(textField10,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(textField11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                        .addComponent(label11,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addComponent(textField12,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                  .addComponent(label13,      javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                  .addComponent(choice1,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
               .addGap(83, 83, 83)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                  .addGroup(layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                        .addComponent(button1,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
                        .addComponent(button2,
javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                     .addComponent(label12,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                 .addComponent(button3,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                 .addComponent(button4,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
              .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
       );


       pack();
    }// </editor-fold>


private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
       textField1.setEditable(true);
       textField2.setEditable(true);
       textField13.setEditable(true);
       textField8.setEditable(true);
       textField14.setEditable(true);
       textField9.setEditable(true);
       textField10.setEditable(true);
       textField11.setEditable(true);
       textField12.setEditable(true);
       button1.setLabel("SAVE");
       //evt.setSource("NULL");
```

```
        int selected_id =Integer.parseInt(label12.getText());
        if(evt.getActionCommand().equals("SAVE"))
        {
            try {
                stat=con.prepareStatement("UPDATE Emp SET Name = ?,Address = ?,Post =
?,Experiance = ?,Salary = ?,HRA = ?,TA = ?,PF = ?,ITax = ? WHERE Eid="+
selected_id);
                stat.setString(1,textField1.getText());
                stat.setString(2,                    textField2.getText());
                stat.setString(3,                  textField13.getText());
                stat.setInt(4,Integer.parseInt(textField8.getText()));
                stat.setInt(5,Integer.parseInt(textField14.getText()));
                stat.setInt(6,Integer.parseInt(textField9.getText()));
                stat.setInt(7,Integer.parseInt(textField10.getText()));
                stat.setInt(8,Integer.parseInt(textField11.getText()));
                stat.setInt(9,Integer.parseInt(textField12.getText()));
                stat.executeUpdate();
                label12.setText("SAVED");
            }              catch              (SQLException              ex)              {
                Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
            }
        }


}


private void button4ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
//      TODO      add      your      handling      code      here:
            label12.setText(choice1.getSelectedItem().toString());
            int selected_id =Integer.parseInt(label12.getText());
            try {
                stmt = con.createStatement();
            }              catch              (SQLException              ex)              {
                Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
            }
            try {
                rs = stmt.executeQuery("Select * from emp WHERE Eid="+ selected_id);
            }           catch            (SQLException            ex)          {
                Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
            }
            while(rs.next())
            {
            textField1.setText(rs.getString(1));
            textField2.setText(rs.getString(2));
            textField13.setText(rs.getString(3));
            textField8.setText(Integer.toString(rs.getInt(4)));
            textField14.setText(Integer.toString(rs.getInt(5)));
            textField9.setText(Integer.toString(rs.getInt(6)));
            textField10.setText(Integer.toString(rs.getInt(7)));
            textField11.setText(Integer.toString(rs.getInt(8)));
            textField12.setText(Integer.toString(rs.getInt(9)));
            }
            button1.setEnabled(true);
            button2.setEnabled(true);
        }catch              (SQLException              ex)             {
            Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
        }
}

private void button3ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
   this.setVisible(false);
        try {
            con.close();
        }          catch            (SQLException            ex)          {
            Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
        }
   Menuscrn menu = new Menuscrn();
```

```
    menu.setVisible(true);
}

private void button2ActionPerformed(java.awt.event.ActionEvent evt) {

        try {
// TODO add your handling code here:
            button2.setLabel("SAVE");
            int selected_id = Integer.parseInt(label12.getText());
            stat = con.prepareStatement("DELETE from Emp WHERE Eid = ?");
            stat.setInt(1, selected_id);
            if (evt.getActionCommand().equals("SAVE")) {
                stat.executeUpdate();
                label12.setText("CHANGES SAVED");
            }
        }        catch        (SQLException        ex)        {
            Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
        }
}

    /**
    * @param args the command line arguments
    */
    public    static    void    main(String    args[])    {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                try {
                    new EmpDetail().setVisible(true);
                }        catch        (ClassNotFoundException        ex)        {
                    Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE,    null,
ex);
                }        catch        (SQLException        ex)        {
                    Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE,    null,

ex);
```

```
                    }
}
});
        }


        // Variables declaration - do not modify
        private      java.awt.Button       button1;
        private      java.awt.Button       button2;
        private      java.awt.Button       button3;
        private      java.awt.Button       button4;
        private      java.awt.Choice      choice1;
        private java.awt.Label label1;
        private java.awt.Label label10;
        private java.awt.Label label11;
        private java.awt.Label label12;
        private java.awt.Label label13;
        private  java.awt.Label  label2;
        private  java.awt.Label  label3;
        private  java.awt.Label  label4;
        private  java.awt.Label  label5;
        private  java.awt.Label  label6;
        private  java.awt.Label  label7;
        private  java.awt.Label  label8;
        private java.awt.Label label9;
        private  java.awt.TextField  textField1;
        private java.awt.TextField textField10;
        private java.awt.TextField textField11;
        private java.awt.TextField textField12;
        private java.awt.TextField textField13;
        private java.awt.TextField textField14;
        private  java.awt.TextField  textField2;
        private  java.awt.TextField  textField8;
        private java.awt.TextField textField9;
        // End of variables declaration
```

```
/*
 * EmpSlip.java
 */
 package employeemanagement;
import java.awt.HeadlessException;
import java.awt.print.PageFormat;
import java.awt.print.Pageable;
import java.awt.print.Printable;
import java.awt.print.PrinterException;
import java.awt.print.PrinterJob;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.Calendar;
import java.util.Date;
import java.util.logging.Level;
import java.util.logging.Logger;

public class EmpSlip extends javax.swing.JFrame {

    Connection con;
    Statement stmt;
    PreparedStatement stat;
    ResultSet rs;
    int i=0;
  /** Creates new form EmpSlip */
  public EmpSlip() {
    try {
      initComponents();
      this.setVisible(true);
```

```
            button2.setEnabled(false);

            Calendar cal=Calendar.getInstance();
            Date            now=new            Date();
            now=cal.getTime();
            SimpleDateFormat formatter=new SimpleDateFormat();

           //                                              System.out.println(now);
            label18.setText(formatter.format(now));
            Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
            con = DriverManager.getConnection("jdbc:odbc:Employee");

            stmt                                                                         =
con.createStatement(rs.TYPE_SCROLL_SENSITIVE,rs.CONCUR_READ_ONLY);
            rs = stmt.executeQuery("Select * from emp");
            while                              (rs.next())                              {
               choice1.insert(Integer.toString(rs.getInt(10)),++i);
            }

         }                   catch                   (SQLException                   ex)                   {
            Logger.getLogger(EmpSlip.class.getName()).log(Level.SEVERE, null, ex);
         }                catch                (ClassNotFoundException                ex)                {
            Logger.getLogger(EmpSlip.class.getName()).log(Level.SEVERE, null, ex);
         }

    }


    /** This method is called from within the constructor to

     * initialize the form.

     * WARNING: Do NOT modify this code. The content of this method is

     * always regenerated by the Form Editor.

     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {


        label1  =  new  java.awt.Label();
        label2  =  new  java.awt.Label();
        button1 = new java.awt.Button();
```

```
button2 = new java.awt.Button();
choice1 = new java.awt.Choice();
label3  =  new  java.awt.Label();
label4  =  new  java.awt.Label();
label5  =  new  java.awt.Label();
label7  =  new  java.awt.Label();
label8  =  new  java.awt.Label();
label9  =  new  java.awt.Label();
label10  =  new  java.awt.Label();
label11  =  new  java.awt.Label();
label12  =  new  java.awt.Label();
label13  =  new  java.awt.Label();
label14  =  new  java.awt.Label();
label15  =  new  java.awt.Label();
label16  =  new  java.awt.Label();
label17  =  new  java.awt.Label();
label6  =  new  java.awt.Label();
button3 = new java.awt.Button();
label18 = new java.awt.Label();


setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setBackground(new java.awt.Color(255, 255, 255));


label1.setBackground(new java.awt.Color(0, 0, 0));

label1.setForeground(new java.awt.Color(255, 255, 255));
label1.setText("COMPUTER GENRATED PAYSLIP");


label2.setBackground(new java.awt.Color(255, 255, 255));

label2.setForeground(new java.awt.Color(0, 0, 0));


button1.setLabel("OK");

button1.addActionListener(new java.awt.event.ActionListener() {
   public void actionPerformed(java.awt.event.ActionEvent evt) {
      button1ActionPerformed(evt);
```

```
}
});

        button2.setLabel("PRINT");
        button2.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button2ActionPerformed(evt);
}
});

        label3.setText("EMP                              ID");
        label4.setText("BASIC                       PAY");
        label5.setText("+ HOUSE ALLOWANCE");
        label7.setText("+ TRAVELLING ALLOWANCE");

        label8.setBackground(new java.awt.Color(255, 255, 255));
        label8.setForeground(new java.awt.Color(0, 0, 0));
        label9.setText("-        PROVIDENT        FUND");
        label10.setText("-           INCOME           TAX");
        label16.setText("NET PAY");
        label17.setText("-----------------------------------------------------------------------------------------------------------------------
  ____");

        button3.setLabel("BACK");
        button3.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                button3ActionPerformed(evt);
```

```
}
});

        javax.swing.GroupLayout layout = new
    javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()

    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
    )
                    .addGroup(layout.createSequentialGroup()

    .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
    )
                            .addGroup(layout.createSequentialGroup()
                                .addContainerGap()
                                .addComponent(label7,
    javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addGap(19, 19, 19)
                                .addComponent(label4,
    javax.swing.GroupLayout.PREFERRED_SIZE,                                  107,
    javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addContainerGap()
                                .addComponent(label5,
    javax.swing.GroupLayout.PREFERRED_SIZE,
    javax.swing.GroupLayout.DEFAULT_SIZE,
    javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(layout.createSequentialGroup()
                                .addContainerGap()


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILIN
G, false)
                                    .addComponent(label9,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                    .addComponent(label10,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                            .addComponent(label2,
javax.swing.GroupLayout.PREFERRED_SIZE,                              556,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILIN
G, false)
                                .addComponent(label15,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(label14,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                                .addComponent(label13,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(label12,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(label11,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 99, Short.MAX_VALUE)
                            .addComponent(label8,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.PREFERRED_SIZE,                                        157,
javax.swing.GroupLayout.PREFERRED_SIZE))))
.addGroup(layout.createSequentialGroup()
                     .addContainerGap()


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                         .addGroup(layout.createSequentialGroup()
                            .addComponent(label17,
javax.swing.GroupLayout.PREFERRED_SIZE,                                        227,
javax.swing.GroupLayout.PREFERRED_SIZE)


.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                            .addComponent(button1,
javax.swing.GroupLayout.PREFERRED_SIZE,                                         98,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(53, 53, 53)
                            .addComponent(button2,
javax.swing.GroupLayout.PREFERRED_SIZE,                                         84,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addGap(56, 56, 56)
                            .addComponent(button3,
javax.swing.GroupLayout.PREFERRED_SIZE,                                         84,
javax.swing.GroupLayout.PREFERRED_SIZE))
```

```
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(label16,
javax.swing.GroupLayout.PREFERRED_SIZE,                         144,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(27, 27, 27)
                        .addComponent(label6,
javax.swing.GroupLayout.PREFERRED_SIZE,                         137,
javax.swing.GroupLayout.PREFERRED_SIZE)))))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(label3,   javax.swing.GroupLayout.PREFERRED_SIZE, 54,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(choice1, javax.swing.GroupLayout.PREFERRED_SIZE, 64,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(109, 109, 109)
                .addComponent(label1,        javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,      162,
Short.MAX_VALUE)
                .addComponent(label18, javax.swing.GroupLayout.PREFERRED_SIZE, 126,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addGap(23, 23, 23))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addComponent(label3,           javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(label1,       javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addComponent(choice1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(label18,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
             .addComponent(label2,   javax.swing.GroupLayout.PREFERRED_SIZE, 30,
javax.swing.GroupLayout.PREFERRED_SIZE)
             .addGap(1, 1, 1)
             .addComponent(label8,          javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addGroup(layout.createSequentialGroup()
                   .addComponent(label4,   javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                   .addComponent(label5,   javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label11,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label12,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)


.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label13,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label14,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                    .addComponent(label15,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGroup(layout.createSequentialGroup()
                    .addComponent(label7,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                  .addComponent(label9,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)

.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                  .addComponent(label10,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
            .addGap(1, 1, 1)
            .addComponent(label17,        javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
            .addGap(10, 10, 10)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING
)
               .addGroup(layout.createSequentialGroup()
                  .addComponent(label6,    javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                  .addGap(67, 67, 67)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILIN
G)
                     .addComponent(button1,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 38, Short.MAX_VALUE)
                     .addComponent(button3, javax.swing.GroupLayout.DEFAULT_SIZE,
38, Short.MAX_VALUE)
                     .addComponent(button2,
javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 38, Short.MAX_VALUE)))
               .addComponent(label16,     javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
        );


        pack();
    }// </editor-fold>


private void button1ActionPerformed(java.awt.event.ActionEvent evt) {
        try {
// TODO add your handling code here:
            int selected_id = Integer.parseInt(choice1.getSelectedItem().toString());
            stmt                                                                    =
con.createStatement(rs.TYPE_SCROLL_SENSITIVE,rs.CONCUR_READ_ONLY);
            rs = stmt.executeQuery("Select * from emp WHERE Eid=" + selected_id);
            while (rs.next()) {
                label2.setText(rs.getString(1).trim()+","+rs.getString(2).trim());
                label8.setText("POSITION:                              "+rs.getString(3));
                label11.setText(Integer.toString(rs.getInt(5)));
                label12.setText(Integer.toString(rs.getInt(6)));
                label13.setText(Integer.toString(rs.getInt(7)));
                label14.setText(Integer.toString(rs.getInt(8)));
                label15.setText(Integer.toString(rs.getInt(9)));
                rs.first();
                double npay=rs.getInt(5)+rs.getInt(6)+rs.getInt(7)-rs.getInt(8)-rs.getInt(9);
                label6.setText(Double.toString(npay));
            }
            button2.setEnabled(true);
        }          catch          (SQLException          ex)          {
            Logger.getLogger(EmpSlip.class.getName()).log(Level.SEVERE, null, ex);
        }
}


private void button2ActionPerformed(java.awt.event.ActionEvent evt) {
// TODO add your handling code here:
```

```
final PrinterJob prn=new PrinterJob() {

        @Override
        public void setPrintable(Printable painter) {
            throw new UnsupportedOperationException("Not supported yet.");
        }

        @Override
        public void setPrintable(Printable painter, PageFormat format) {
            throw new UnsupportedOperationException("Not supported yet.");
        }

        @Override
        public void setPageable(Pageable document) throws NullPointerException {
            throw new UnsupportedOperationException("Not supported yet.");
        }

        @Override
        public boolean printDialog() throws HeadlessException {
            throw new UnsupportedOperationException("Not supported yet.");
        }

        @Override
        public PageFormat pageDialog(PageFormat page) throws HeadlessException {
            throw new UnsupportedOperationException("Not supported yet.");
        }

        @Override
        public PageFormat defaultPage(PageFormat page) {
            throw new UnsupportedOperationException("Not supported yet.");
        }

        @Override
```

```
public PageFormat validatePage(PageFormat page) {
```

```
throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public void print() throws PrinterException {
              throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public void setCopies(int copies) {
throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public int getCopies() {
                throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public String getUserName() {
                throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public void setJobName(String jobName) {
                throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public String getJobName() {
throw new UnsupportedOperationException("Not supported yet.");
```

```
}

@Override
```

```
            public void cancel() {
    throw new UnsupportedOperationException("Not supported yet.");
            }

            @Override
            public boolean isCancelled() {
    throw new UnsupportedOperationException("Not supported yet.");
            }
        };
    }

    private void button3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
       this.setVisible(false);
          try {
            con.close();
          }          catch          (SQLException          ex)          {
            Logger.getLogger(EmpDetail.class.getName()).log(Level.SEVERE, null, ex);
          }
       Menuscrn menu = new Menuscrn();
       menu.setVisible(true);
    }

       /**
       * @param args the command line arguments
       */
       public    static    void    main(String    args[])    {
          java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
               new EmpSlip().setVisible(true);
}
});
       }
    // Variables declaration - do not modify
    private     java.awt.Button      button1;
    private     java.awt.Button      button2;
    private     java.awt.Button      button3;
    private     java.awt.Choice      choice1;
    private java.awt.Label label1;
    private java.awt.Label label10;
    private java.awt.Label label11;
    private java.awt.Label label12;
    private java.awt.Label label13;
    private java.awt.Label label14;
    private java.awt.Label label15;
    private java.awt.Label label16;
    private java.awt.Label label17;
    private java.awt.Label label18;
    private  java.awt.Label  label2;
    private  java.awt.Label  label3;
    private  java.awt.Label  label4;
    private  java.awt.Label  label5;
    private  java.awt.Label  label6;
    private  java.awt.Label  label7;
    private  java.awt.Label  label8;
    private java.awt.Label label9;
    // End of variables declaration


}
```

```
/*
*Main.java
 * To change this template, choose Tools | Templates
 * and open the template in the editor.
 */


package employeemanagement;
public class Main {
    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
            // TODO code application logic here
            Intro intr = new Intro();
    }


}
```

CHAPTER 5

TESTING

6.1 Software Testing

Software testing During systems testing, the system is used experimentally to ensure that the software does not fail. In other words, we can say that it will run according to its specifications and in the way users expect. Special test data are input for processing, and the results examined. A limited number of users may be allowed to use the system so that analyst can see whether they try to use it in unforeseen ways. It is desirable to discover any surprises before the organization implements the system and depends on it.

Software modules are tested for their functionality as per the requirements identified during the requirements analysis phase. To test the functionality of the file transfer and chat application, a Quality Assurance (QA) team is formed. The requirements identified during the requirements analysis phase are submitted to the QA team. In this phase the QA team tests the application for these requirements. The development team submits a test case report to the QA team so that the application can be tested in the various possible scenarios.

The software development project, errors can be injected at any stage during development i.e. if there is an error injected in the design phase then it can be detectedin the coding phase because there is the product to be executed ultimately on the machine, so we employ a testing process**.**

During the testing the program to be tested is executed with certain test cases and output of these test cases is evaluated to check the correctness of the program. It is thetesting that performs first step in determining the errors in the program .

6.2 Verification

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

6.3 Validation

Validation is the process to make sure the product satisfies the specified requirementsat the end of the development phase. In other words, to make sure the product is builtas per customer requirements.

6.4 Types of testing

There are many types of testing like-

6.4.1 Functional Testing

Functional Testing verifies that each function of the software application operatesin conformance with the requirement specification. Functional testing is a quality assurance (QA) process and mainly involves Black Box Testing. It is concerned abouttheresults of processing and not about the source code of the application.

Definition: Functional Testing is a type of Software Testing whereby the system is tested against the functional requirements/specifications.

Functions (or features) are tested by providing appropriate input and examining the output. The actual results are then compared with expected results. Functional testing ensures that the requirements are properly satisfied by the application. The testing canbedone either manually or using automation.

Functional Testing is a type of software testing whereby the system is tested against the functional requirements/specifications.

Functions (or features) are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application.This type of testing is not concerned with how processing occurs, but rather, with the results of processing. It simulates actual system usage but does not make any system structure assumptions.

During functional testing, Black Box Testing technique is used in which the internal logic of the system being tested is not known to the tester. Functional testing is normally performed during the levels of System Testing and Acceptance Testing. Typically, functional testing involves the following steps:

- Identify functions that the software is expected to perform.

- Create input data based on the function’s specifications.

- Determine the output based on the function’s specifications.

- Execute the test case.

Functional testing is more effective when the test conditions are created directly from user/business requirements. When test conditions are created from the system documentation (system requirements/ design documents), the defects in that documentation will not be detected through testing and this may be the cause of end-users' wrath when they finallyuse the software.

6.4.2 Unit Testing

In computer programming, unit testing is a software testing method by which individualunits of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures, are tested to determine whether they are fit for use.

Intuitively, one can view a unit as the smallest testable part of an application. In procedural programming, a unit could be an entire module, but it is more commonly an individual functionor procedure. In object-oriented programming, a unit is often an entireinterface, such as a class, but could be an individual method Unit tests are short code fragments created by programmers or occasionally by white box testers during the development process. It forms the basis for component testing.

Ideally, each test case is independent from the others. Substitutes such as methodstubs, mock objects, fakes, and test harnesses can be used to assist testing a module in isolation. Unit tests are typically written and run by software developers to ensure that code meetsits design and behaves as intended.

Because some units may have references to other units, testing a single unit can frequentlyspill over into testing another unit. A common example of this is units that depend on a database: in order to test the unit, the tester often writes code that interacts with the database. This is a mistake, because a unit test should usually not go outside of its own unit boundary, and especially should not cross such process/network boundaries becausethis can introduce unacceptable performance problems to the unit test-suite. Crossing such unit boundaries turns unit tests into integration tests, and when such test cases fail, it may be unclear which component is causing the failure. Instead, the software developer should create an abstract interface around the database queries, andthen implement that interface with their own mock object. By abstracting this necessary attachment from the code (temporarily reducing the net effective coupling),an independent unit can be more thoroughly tested than may have been previously achieved. This results in a higher-quality unit that is also more maintainable.

Unit testing is commonly automated, but may still be performed manually. The IEEE does not favor one over the other. The objective in unit testing is to isolate a unit and validate its correctness. A manual approach to unit testing may employ a step-by-step instructional document. However, automation is efficient for achieving this, and enables the many benefits listed in this article. Conversely, if not planned carefully, a careless manual unit test case may execute as an integration test case that involves many softwarecomponents, and thus preclude the achievement of most if not all of the goals established for unit testing.

To fully realize the effect of isolation while using an automated approach, the unit or code body under test is executed within a framework outside of its natural environment. In other words, it is executed outside of the product or calling context for which it is intended. Testing in such an isolated manner reveals unnecessary dependencies between the code being tested and other units or data spaces in the product. These dependencies can then be eliminated.

Using an automation framework, the developer codes criteria, or a test oracle or result that is known to be good, into the test to verify the unit’s correctness. During test case execution, the framework logs tests that fail any criterion. Many frameworks will also automatically flag these failed test cases and report them in a summary. Depending upon the severity of a failure, the framework may halt subsequent testing.

As a consequence, unit testing is traditionally a motivator for programmers to create decoupled and cohesive code bodies. This practice promotes healthy habits in software development. Software design patterns, unit testing, and code refactoring often work together so that the best solution may emerge.

Writing and maintaining unit tests can be made faster by using Parameterized Tests (PUTs). These allow the execution of one test multiple times with different input sets, thus reducing test code duplication. Unlike traditional unit tests, which are usually closed methods and test invariant conditions, PUTs take any set of parameters. PUTs havebeen supported by TestNG, JUnit and its .Net counterpart, XUnit. Suitable parameters forthe unit tests may be supplied manually or in some cases are automatically generated by the test framework. In recent years support was added for writing more powerful (unit) tests,leveraging the concept of Theory. A parameterized test uses the same execution steps with multiple input sets that are pre-defined. A theoryis a test case that executes thesame steps still, but inputs can be provided by a data generating method at run time.

The goal of unit testing is to isolate each part of the program and show that the individualparts are correct. A unit test provides a strict, written contract that the piece of
code must satisfy. As a result, it affords several benefits.

Unit testing finds problems early in the development cycle. This includes both bugs in the programmer's implementation and flaws or missing parts of the specificationfor the unit. The process of writing a thorough set of tests forces the author to think through inputs, outputs, and error conditions, and thus more crisply define the unit's desired behavior. The cost of finding a bug before coding begins or when the code is first writtenis considerably lower than the cost of detecting, identifying, and correcting the bug later. Bugs in released code may also cause costly problems for the end-users of the software Code can be impossible or difficult to unit test if poorly written, thus unit testing can forcedevelopers to structure functions and objects in better ways.

In test-driven development (TDD), which is frequently used in both extreme programming and scrum, unit tests are created before the code itself is written. When the tests pass, that code is considered complete. The same unit tests are run against that function frequently as the larger code base is developed either as the code is changed or via an automated process with the build. If the unit tests fail, it is considered to be a bug either in the changed code or the tests themselves. The unit tests then allow the location of the fault or failure to be easily traced. Since the unit tests alert the development team of the problem before handing the code off to testers or clients, potential problems are caught early in the development process.

Unit testing allows the programmer to refactor code or upgrade system libraries ata laterdate, and make sure the module still works correctly (e.g., in regression testing). The procedure is to write test cases for all functions and methods so that whenever a change causes a fault, it can be quickly identified. Unit tests detect changes which may break a design contract.

Unit testing may reduce uncertainty in the units themselves and can be used in a bottom- up testing style approach. By testing the parts of a program first and then testing the sumof its parts, integration testing becomes much easier.

Unit testing provides a sort of living documentation of the system. Developers looking tolearn what functionality is provided by a unit, and how to use it, can look at the unit teststo gain a basic understanding of the unit's interface (API).

Unit test cases embody characteristics that are critical to the success of the unit. These characteristics can indicate appropriate/inappropriate use of a unit as well as negative behaviors that are to be trapped by the unit. A unit test case, in and of itself, documents these critical characteristics, although many software development environments do not rely solely upon code to document the product in development.

When software is developed using a test-driven approach, the combination of writing the unit test to specify the interface plus the refactoring activities performed after the test has passed, may take the place of formal design. Each unit test can be seen as a design elementspecifying classes, methods, and observable behavior.

Testing will not catch every error in the program, because it cannot evaluate every execution path in any but the most trivial programs. This problem is a superset of the halting problem, which is undecidable. The same is true for unit testing. Additionally,unit testing by definition only tests the functionality of the units themselves. Therefore, itwillnot catch integration errors or broader system-level errors (such as functions performed across multiple units, or non-functional test areas such as performance). Unit testing should be done in conjunction with other software testing activities, as they can only show the presence or absence of particular errors; they cannot prove a complete absence of errors. To guarantee correct behavior for every execution path and every possible input, and ensure the absence of errors, other techniques are required, namelythe application offormal methods to proving that a software component has no unexpected behavior.

An elaborate hierarchy of unit tests does not equal integration testing. Integrationwith peripheral units should be included in integration tests, but not in unit tests. Integration testing typically still relies heavily on humans testing manually; high-level or global-scope testingcan be difficult to automate, such that manual testing often appearsfaster andcheaper.

Software testing is a combinatorial problem. For example, every Boolean decision statement requires at least two tests: one with an outcome of "true" and one with an outcome of "false". As a result, for every line of code written, programmers often need 3to 5 lines of test code. This obviously takes time and its investment may not be worth the effort. There are problems that cannot easily be tested at all – for example those that are non-deterministic or involve multiple threads. In addition, code for a unit test is likelyto be at least as buggy as the code it is testing. Fred Brooks in The Mythical Man-Month. Never go to sea with two chronometers; take one or three. Meaning, if two chronometerscontradict, how do you know which one is correct?

Another challenge related to writing the unit tests is the difficulty of setting up realistic and useful tests. It is necessary to create relevant initial conditions so the part ofthe application being tested behaves like part of the complete system. If these initial conditions are not set correctly, the test will not be exercising the code in a realistic context, which diminishes the value and accuracy of unit test results.

To obtain the intended benefits from unit testing, rigorous discipline is needed throughout the software development process. It is essential to keep careful records not only of the tests that have been performed, but also of all changes that have been madeto the source code of this or any other unit in the software. Use of a version controlsystem is essential. If a later version of the unit fails a particular test that it had previously passed, the version-control software can provide a list of thesource codechanges (if any) that have been applied to the unit since that time.

It is also essential to implement a sustainable process for ensuring that test case failuresare reviewed regularly and addressed immediately. If such a process is not implemented and ingrained into the team's workflow, the application will evolve out of sync with the unit test suite, increasing false positives and reducing the effectivenessof the test suite.

Unit testing embedded system software presents a unique challenge: Because the software is being developed on a different platform than the one it will eventually runon,you cannot readily run a test program in the actual deployment environment, as is possiblewith desktop programs.

Unit tests tend to be easiest when a method has input parameters and some output.It is not as easy to create unit tests when a major function of the method is to interact with something external to the application. For example, a method that willwork with a database might require a mock up of database interactions to be created, which probably won't be as comprehensive as the real database interactions.

6.4.3 Smoke Testing

Smoke Testing is performed after software build to ascertain that the critical functionalities of the program are working fine. It is executed "before" any detailed functional or regression tests are executed on the software build. The purpose is to rejecta badly broken application, so that the QA team does not waste time installing and testingthe software application. It is a part of Functional Testing. It can be performed by the developers or testers. Sometimes developers also test the application after the deployment, before handing it over to QA to test. Or sometimes QA team Smoke Test the application before accepting any new build from the developersto avoid any time wastage.

Smoke testing covers most of the major functions of the software but none of themin depth.The result of this test is used to decide whether to proceed with further testing. If the smoke test passes, go ahead with further testing. If it fails, halt further tests and askfor a new build with the required fixes. If an application is badly broken, detailed testing mightbe a waste of time and effort.

Smoke test helps in exposing integration and major problems early in the cycle. It can be conducted on both newly created software and enhanced software. Smoke test isperformedmanually or with the help of automation tools/scripts. If builds are prepared frequently, itis best to automate smoke testing.

As and when an application becomes mature, with addition of more functionalitiesetc, thesmoke test needs to be made more expansive. Sometimes, it takesjust one incorrect character in the code to render an entire application useless.

6.4.4 Sanity Testing

The terminologies such as Smoke Test or Build Verification Test or Basic Acceptance Test or Sanity Test are interchangeably used, however, each one of them is used under aslightly different scenario. It is a part of Functional Testing.

Sanity Testing is a software testing technique performed by the test team for somebasic tests. Whenever a new build is received, after minor changes in code or functionality, Sanity testing is performed to ascertain that the bugs have been fixed.

Sanity Testing is done after thorough regression testing is over. It is done to make sure that any defect fixes or changes after regression testing does not break the core functionality of the product. The goal is to determine that the proposed functionalityworks roughly as expected. It follows a narrow and deep approach and concentrates on detailed testing of some limited and main features.

Sanity Test determines whether it is reasonable to proceed with further testing.Sanity tests are mostly non scripted.

6.4.5 Regression Testing

Regression testing is the retesting of a software system to confirm that changes made to few parts of the codes has not any side effects on existing system functionalities.It is to ensure that old codes are still working as they were before introduction of the newchange. The ideal process would be to create an extensive test suite and run it after each and everychange.

ISTQB Definition: Regression testing is a type of software testing that intends toensure that changes (enhancements or defect fixes) to the software have not adversely affected it.

As software is updated or changed, or reused on a modified target, emergence of new faults and/or re-emergence of old faults is quite common. Sometimes re-emergence occurs because a fix gets lost through poor revision control practices (or simple human error in revision control). Often, a fix for a problem will be "fragile" in that it fixes the problem in the narrow case where it was first observed but not in more general cases which may arise over the lifetime of the software. Frequently, a fix for a problem in one area inadvertently causes a software bug in another area. Finally, it may happen that, when some feature is redesigned, some of the same mistakes that were made in theoriginalimplementation of the feature are made in the redesign.

Therefore, in most software development situations, it is considered good codingpractice, when a bug is located and fixed, to record a test that exposes the bug andre-run that test regularly after subsequent changes to the program. Although this may be done

through manual testing procedures using programming techniques, it is often done usingautomated testing tools. Such a test suite contains software tools that allow the testing environment to execute all the regression test cases automatically; some projects even set up automated systems to re-run all regression tests at specified intervals and report anyfailures (which could imply a regression or an out-of-date test). Common strategies are torun such a system after every successful compile (for small projects), every night, or oncea week. Those strategies can be automated by an external tool.

Regression testing is an integral part of the extreme programming software development method. In this method, design documents are replaced by extensive, repeatable, and automated testing of the entire software package throughout each stage ofthe software development process. Regression testing is done after functional testing hasconcluded, toverify that the other functionalities are working.

In the corporate world, regression testing has traditionally been performed by a software quality assurance team after the development team has completed work. However, defectsfound at this stage are the most costly to fix. This problem is being addressed by the rise of unit testing. Although developers have always written test cases as part of the development cycle, these test cases have generally been either functional tests or unit tests that verify only intended outcomes. Developer testing compels a developer to focus on unit testing and to include both positive and negative test cases.

6.4.5.1 When to perform Regression Testing

We do software regression testing whenever the production code is modified. Usually, we do execute regression tests in the following cases: Some regression testing examplesare here.

- When new functionalities are added to the application.

Example: A website has a login functionality which allows users to do login only with Email. Now the new features look like "providing a new feature to do login using Facebook".

- When there is a Change Requirement (In organizations, we call it as CR).

Example: Remember password should be removed from the login page which isavailableearlier.

- When there is a Defect Fix.

Example: Imagine Login button is not working in a login page and a tester reports a bug stating that the login button is broken. Once the bug is fixed by the developers, testers test it to make sure whether the Login button is working as per the expected result. Simultaneously testers test other functionalities which are related to login button

- When there is a Performance Issue Fix.

Example: Loading the home page takes 5 seconds. Reducing the load time to 2seconds.

- When there is an Environment change.

Example: Updating the Database from MySQL to Oracle)

So far we have seen what regression is and when do we do regression Now let'ssee howwe do regression testing.

6.4.5.2 How to select Test for Regression Test Suite

- Include the test cases which have frequent or recent defects.

- Include the test cases which verify core features of the product.

- Include the test cases for Functionalities which have undergone more andrecentchanges.

- Include set Integration test cases which touch the entire major component.

- Include all Complex Test Cases.

- Include P1 & P2 test cases for regression testing.

- made to code withina project, regression testing should be implemented. Modified pieces of code canbe checked using existing tests to determine if the changes broke anything that was previously functioning properly, and by writing new tests when needed. It's important to test fixed bugs and to watch for side effects of such fixes. Be aware,while a bug is being fixed, it's possible for another one to be introduced.

- Set requirements for testing Collaborations- between business stakeholders, developers, and software testing engineers should identify the appropriate test/usecases for the project or organization at hand. Agreeing upon and creating this list of cases provides a framework for future cost (and time) savings for all parties involved. By doing so, the test cases put in place can be leveraged down the roadand used many times.

- Develop entry criteria for regression testing-To effectively determine when tostarttesting, entry criteria need to be put into place relating to the project. Entry criteriaare the minimum eligibility or minimum set of conditions that should be met before starting testing work. Before regression testing, the engineer should: Confirm that the defect is repeatable and has been properly documented. Open a defect tracking record to identify and track the regression testing efforts. Createa regression test that's specific to the defect,and get the test reviewed and acceptedby project managers

- Develop exit criteria for regression testing-Similar to entry criteria, exit criteria should be developed to set the minimum eligibility or minimum conditions that need to be met before the testing phase is closed. These elements are agreed uponduring the test-planning phase and signed off prior to product release. During this process, software engineers should: Make sure all tests have been run. Ensure code coverage requirement levels have been met. Leave no severebugs untested or unfixed. Ensure all "high-risk" areas have been fully tested Follow a schedule-When the above steps have been completed, and the components that need regression testing have been identified, it's time to create a testing schedule. Be sure to identify when initial unit tests for the targeted components should becompleted (within one hour, one day, two days, 5 days, etc.). Regression tests can also bescheduled to automatically run at a specific time for more controlled, thorough testing ofthe applications. While this is just a short list of steps that should be followed when implementing a regression testing strategy, we feel they are critical in maintaining maximum code coverage for frequently changed components. VectorCAST/Manage is a valuable tool for obtaining the regression testing results of previously developed unit andintegration test environments, including code coverage data aggregation, helping organizations achieve a valuable continuous integration process. Failing to test and analyze software after changes have been made can result in defects and put lives at risk,especially in safety-critical applications.

6.5 TEST CASESTest Case

During Test Cases that are good at revealing the presence of faults is central to successful testing. The reason for this is that if there is a fault in the program, the program can still provide the expected behavior on the certain inputs. Only for the set ofinputs the faults that exercise the fault in the program will the output of the program devise from the expected behavior. Hence, it is fair to say that testing is as good as its test case.

The number of test cases used to determine errors in the program should be minimum. There are two fundamental goals of a practical testing activity:-
maximize the number of errors detected
and.minimize the number of test cases.

As these two goals are contradictory so the problem of selecting test cases is a complex one. While selecting the test cases the primary objective is to ensure that if there is an error or fault in the program, it is exercised by one of its test cases. An ideal test case is one which succeeds (meaning that there are no errors, revealed in its execution) only it there are no errors in the program one possible set of ideal test cases is one which includes all the possible inputs to the program. This is often called "exhaustive testing", however it is impractical and infeasible as even a small program can have an infinite input domain.

REFERENCES

1. Rodriguez, S., E. N. Baydak, F. F. Schoeggl, S. D. Taylor, G. Hay, and H. W. Yarranton. "Regular solution based approach to modeling asphaltene precipitation from native and reacted oils: Part 3, visbroken oils." *Fuel* 257 (2019): 116079.

2. Tabarés, Raúl. "HTML5 and the evolution of HTML; tracing the origins of digital platforms." *Technology in Society* 65 (2021): 101529.

3. Chen, Peihong, Jinling Liu, Kaijun Zhang, Dongzhen Huang, Siyu Huang, Qingchun Xie, Fan Yang et al. "Preparation of clarithromycin floating core- shell systems (CSS) using multi-nozzle semi-solid extrusion-based 3D printing." *International Journal of Pharmaceutics* (2021): 120837.

4. Mojirsheibani, Majid. "A note on the performance of bootstrap kernel density estimation with small re-sample sizes." *Statistics & Probability Letters* (2021):109189.

5. Alex, Scaria, and T. Dhiliphan Rajkumar. "2 Spider Bird Swarm Algorithm with Deep Belief Network for Malicious JavaScript Detection." *Computers &Security* (2021): 102301.

6. Hallensleben, Cynthia, Eline Meijer, Jaco Biewenga, Regien MM Kievits-Smeets, Marjan Veltman, Xiaoyue Song, Job FM van Boven, and Niels H. Chavannes. "REducing Delay through edUcation on eXacerbations (REDUX) in patients with COPD: a pilot study." *Clinical eHealth* 3 (2020): 63-68.

7. Hlina, Benjamin L., Daniel M. Glassman, Auston D. Chhor, Brooke S. Etherington, Chris K. Elvidge, Benjamin K. Diggles, and Steven J. Cooke. "Hook retention but not hooking injury is associated with behavioral differences in Bluegill." *Fisheries Research* 242 (2021): 106034.

8. Khanna, Reena, Brian Bressler, Barrett G. Levesque, Guangyong Zou, Larry

W. Stitt, Gordon R. Greenberg, Remo Panaccione et al. "Early combined immunosuppression for the management of Crohn's disease (REACT): a cluster randomised controlled trial." *The Lancet* 386, no. 10006 (2015): 1825-1834

9. David, Matthew. *Developing websites with jQuery mobile*. Focal Press, 2011..

10. Hayashi, Eriko. "JS6-3 Cancer treatment support for patients who havedifficulties in daily life." *Annals of Oncology* 32 (2021): S238.
11. Imamura, Yoshinori. "JS5-3 Current status and core measures of cancer-associated thromboembolism." *Annals of Oncology* 32 (2021): S237.
12. Gao, Ya, Ziyu Zhu, Xiaoxue Xi, Tingwei Cao, Wei Wen, Xiuhua Zhang, and Shengfu Wang. "An aptamer-based hook-effect-recognizable three-line lateral flow biosensor for rapid detection of thrombin." *Biosensors and*

Bioelectronics 133 (2019): 177-182.

13. Merz, Justin, Varaprasad Bandaru, Quinn Hart, Nathan Parker, and Bryan M.Jenkins. "Hybrid Poplar based Biorefinery Siting Web Application (HP- BiSWA): An online decision support application for siting hybrid poplar based biorefineries." *Computers and Electronics in Agriculture* 155 (2018): 76-83.

14. Olejnik, Richard, Teodor-Florin Fortiş, and Bernard Toursel. "Webservices oriented data mining in knowledge architecture." *Future Generation Computer Systems* 25, no. 4(2009): 436-443

15. Douglas. *JavaScript: : The Good Parts*. " O'Reilly Media, Inc.", 2008.