# RV COLLEGE OF ENGINEERING

# BENGALURU – 560059

(Autonomous Institution Affiliated to VTU, Belagavi)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**



**HUMAN RESOURCE ANALYSIS**

**AND**

**STUDENT RECORD MANAGEMENT**

**MINI-PROJECT REPORT**

**OBJECT ORIENTED PROGRAMMING USING JAVA (16CS44)**

**IV SEMESTER**

**2018-2019**

**Submitted by**

**PRAJWAL.V A - 1RV17CS107**

**PRATHEEKSHA .P - 1RV17CS112**

**S.MEGHNA - 1RV17CS129**

**Under the Guidance of**

**Dr. SOUMYA. M**

**Department of CSE, RVCE,**

**Bengaluru - 560059**

**RV College of Engineering, BENGALURU - 560059**

**(Autonomous Institution Affiliated to VTU, Belagavi)**

# Department of Computer Science and Engineering



**CERTIFICATE**

Certified that the **Mini-**project work titled “HUMAN RESOURCE ANALYSIS AND STUDENT RECORD MANAGEMENT” has been carried out by PRAJWAL.V. A (1RV17CS107), PRATHEEKSHA. P (1RV17CS112) AND S. MEGHNA (1RV17CS129) bonafide students ofR.V. College of Engineering, Bengaluru, have submitted in partial fulfillment for the **Assessment of** Course**: OBJECT ORIENTED PROGRAMMING USING JAVA (16CS44) – Open-Ended Experiments** during the year 2018-2019. It is certified that all corrections/suggestions indicated for the internal assessment have been incorporated in the report.

**Faculty Incharge:**

**Dr. Sowmya. M**

Department of CSE,

RVCE., Bengaluru –59

**Head of Department**

Department of CSE,

RVCE, Bengaluru–59

**RV College of Engineering, bENGALURU - 560059**

**(Autonomous Institution Affiliated to VTU)**

# Department of Computer Science and Engineering

**DECLARATION**

We, PRAJWAL.V. A(1RV17CS107), PRATHEEKSHA. P(1RV17CS112) and S. MEGHNA(1RV17CS129) the students of 4th Semester B.E., Department of Computer Science and Engineering, R.V. College of Engineering, Bengaluru hereby declare that the mini-project titled **“**HUMAN RESOURCE ANALYSIS**”** has been carried out by us and submitted in partial fulfillment for the **Assessment of** Course**: OBJECT ORIENTED PROGRAMMING USING JAVA (16CS44) - Open-Ended Experiment** during the year 2018-2019.

***Student’s Name/Signature***

**Place: Bengaluru PRAJWAL.V.A**

**Date: PRATHEEKSHA P**

**S.MEGHNA**

**ABSTRACT**

**HUMAN RESOURCE ANALYSIS**

To develop a front end for an ANN based prediction, whether an employee will continue to work for the company or quit their job, based on their experience in the company.

It allows the HR of the organization to predict the same by providing details about the employee such as his/her tenure in the company, working experience, his/her contribution to the company’s development, number of projects completed, etc.

The prediction is done by a neural network which is trained to predict the same.

**STUDENT RECORD MANAGEMENT**

The student information management system is required to provide a useable and managed interface for Student, Parent and Administrator users to view and manipulate the data for which it is responsible.

The student record system maintains the records of the students, the courses and the results of their studies. It also maintains the Personal Information for each student registered at the university.

**CONTENTS**

1. Introduction
2. Basic OOP concepts
3. Design
4. Java FX
5. Deep learning
6. Technology stack and References
7. Conclusion
8. Future Scope
9. Requirement Specifications
10. Snapshots of result
11. Bibliography

APPENDIX A - SOURCE CODE

1. **INTRODUCTION**

**Traditional vs Object Oriented Approach**

All computer programs consist of two elements: code and data. Furthermore, a program can be conceptually organised around its code or around its data. That is, some programs are written around “what is happening” and others are written around who is being affected. These are two paradigms that govern how a program is constructed.

The first way is the traditional approach wherein this approach sees a program as a series of linear steps (that is, code). The traditional approach can be thought of as code acting on data. Some of the languages which makes use of traditional approach are C, Fortran etc.

The second way which is the object oriented approach organises a program around its data (that is, objects) and a set of well defined interfaces to that data. An object oriented program can be characterized as data controlling access to code.

**OOA, OOD & OOP and their relationship**

Object-oriented analysis is a method of analysis that examines requirements from the perspective of the classes and objects found in the vocabulary of the problem domain. Object-oriented design is a method of design encompassing the process of object oriented decomposition and a notation for depicting both logical and physical as well as static and dynamic models of the system under design.

Object-oriented programming is a method of implementation in which programs are organized as cooperative collections of objects, each of which represents an instance of some class, and whose classes are all members of a hierarchy of classes united via inheritance relationships.

Relationship between them-

The products or results of object oriented analysis serve as models from which we start an object oriented design. Later on, the products or results obtained from object oriented design can then be used used as blueprints for completely implementing a system using object oriented programming methods.

**Features of Java**

Some of the features of Java include-

1. Java is simple- Java was designed to be easy for the professional programer to learn and use effectively.
2. Java is object-oriented- Java makes use of object-oriented approach wherein data is treated as objects and the code revolves around the data.
3. Java is robust- The multi-platformed environment of the Web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. Thus, Java must be robust in nature.
4. Java is multithreaded- Java was designed to meet the real-world requirement of creating interactive, networked programs. To accomplish this, Java supports multithreaded programming.
5. Java is architecture-neutral- Java is independent of which operating system is used to execute it. It works on all operating system. Hence making it portable.
6. Java is interpreted and has high performance- Java enables the creation of cross platform by compiling into an intermediate representation called Java bytecode. This code can be executed on any system that implements the Java Virtual Machine (JVM). It uses a just-in-time compiler to increase its performance.
7. Java is distributed- Java is designed for the distributed environment of the Internet because it handles TCP/IP protocols.
8. Java is dynamic- Java allows to dynamically link code in a safe and expedient manner. This is crucial to the robustness of the Java environment.
9. **BASIC OOP CONCEPTS**

* Implementation of classes and objects

Objects are the basic unit of object oriented programming. Both data and function that operate on a data are bundled together and are called objects.

Class is a user defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. JavaFX has Main class, which contains the objects of Stage and Scene classes. All JavaFX applications have at least one stage and one scene.

* Data abstraction

Data abstraction refers to, providing only essential information to the outside world and hiding their background details, i.e., to represent the needed information in program without presenting the details.

For example, a database system hides certain details of how data is stored and created and maintained

The front end as seen by the user hides all the backend complex ANN implementation.

* Inheritance

Inheritance is an OOPS concept in which one object acquires the properties and behaviors of the parent object. It’s creating a parent-child relationship between two classes. The property of inheritance wasn’t fully exploited in JSP project, as instead of creating classes which inherit from super classes, we relied upon linking different programs together.

In JavaFX, Inheritance property was used. The Main class is inherited by an Application class.

**public class** Main **extends** Application {---}

* Polymorphism

It is a concept by which we can perform a single action in different ways. Polymorphism is derived from 2 Greek words: poly and morphs. The word “poly” means many and “morphs” means forms. So, polymorphism means many forms. There are two types of polymorphism in Java: compile-time polymorphism and runtime polymorphism. We can perform polymorphism in Java by method overloading and method overriding. If we overload a static method in Java, it is the example of compile-time polymorphism. Otherwise, it is called runtime polymorphism. Every class of all the question types makes use of “queans” function and “input” function. However these are used in different places for different purposes hence exhibiting polymorphism.

* Multithreading

Multithreaded programming is a concept wherein a program wherein a program two or more parts that can run concurrently. Each part of such a program is called a thread, and each thread defines a separate path of execution. Thus, multithreading is a specialised form of multitasking.

* Exception handling

An exception is an abnormal condition that arises in a code sequence at run-time. In other words, an exception is a run-time error. These exceptions are handled manually - by making use of some standard keywords like-”try”, “catch”, “finally”, “throw” and “throws”. All the exceptions are derived from a standard inbuilt class called the Exception class. This class can be extended to create custom exceptions or user-defined exceptions.

* Lambda expressions

Lambda expressions is essentially an anonymous (that is, unnamed) method . However, this method is not executed on its own. Instead, it is used to implement a method defined by a functional interface. Lambda expression is often referred to as closures.

Lambda expression always needs a functional interface for it to be executed. A functional interface is an interface that contains one and only one abstract method. Thus, a functional interface represents a single action. Functional interface is sometimes referred to as Single Abstract Method (SAM).

* Interfaces and Packages

Interfaces are used to use the same set of functions for many classes. It is an alternative to multiple inheritance as multiple inheritance is not supported in Java. By providing the interface keyword, Java allows you to fully utilize the “one interface, multiple methods” aspect of polymorphism.

Packages are used to avoid any name collisions between the file names. Java provides this facility in order partition the class name space into more manageable chunks. The package is both a naming and a visibility control mechanism

* JAVAFX Framework

Java FX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using Java FX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc..

In this project, we create the front end using java FX which takes the necessary inputs and feeds it to the ANN.

* JDBC

JDBC refers to Java Database Connectivity. It provides the utilities to create a connection between the SQL server and the program. Every database language has a connector .jar file which is used to establish a connection between the program and its server. There exists a class called a DriverManager class which is used to establish a connection between the program and the database. There are many interfaces like ResultSet, Statement, PreparedStatement, CallableStatement etc.We have JDBC in order to retrieve information from the database

* JSP

JSP may be viewed as a high-level [abstraction](https://en.wikipedia.org/wiki/Abstraction_(computer_science)) of [Java servlets](https://en.wikipedia.org/wiki/Java_servlet). JSPs are translated into [servlets](https://en.wikipedia.org/wiki/Java_Servlet) at runtime, therefore JSP is a Servlet; each JSP servlet is cached and re-used until the original JSP is modified.

JSP allows Java code and certain predefined actions to be interleaved with static web markup content, such as HTML, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, contain Java bytecode rather than [machine code](https://en.wikipedia.org/wiki/Machine_code).

1. **DESIGN**

Design is the first step in the development phase of any technique and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization.

The human resource analysis project is developed using **ARTIFICIAL NEURAL NETWORKS.** The ANN is trained initially in python.

The student record management project is developed using SQL and JDBC. It is HTML form based Access portal to the admin the parent of the ward and the student. Each has a different table that they can access. Privileged access only to the admin is enabled.

1. **JAVA FX**

The modules that have been incorporated in this project are the login details of the HR and the second module consists of the information that has to be provided by the HR to predict.

* HR login

This form takes in three inputs, that is the name of the HR accessing the module, his mail id and password to login.

* Full name
* E-Mail id
* Password
* Prediction form

This form consists of fields that take in information required for prediction.

These will be provided as inputs to the ANN model.

* Satisfaction level
* Average monthly hours
* Number of projects
* Number of years in the company
* Work accident
* Promotion in last 5 years
* Last evaluation
* Department
* Salary

Satisfaction level: This sub module holds the rate of satisfaction on a scale of 10.

Average monthly hours: This sub module contains the average number of hours the employee has worked.

Number of projects: This sub module contains the number of projects completed by the employee.

Number of years in the company: This sub module holds the number of years the employee has worked in the company.

Work accident: This sub module takes in yes /no. If the employee has made any mistake in the past which landed a cost on the company or if the employee was blacklisted for the same it takes a yes.

Promotion in last 5 years: This sub module takes a yes if the employee has had a promotion.

Last evaluation: This sub module takes good if the previous evaluation was good.

Department: This sub module holds the information of which department the employee works in. (accounting, HR, IT, management, marketing, project management, RndD, sales, support, technical)

Salary: This sub module contains the range of the salary.

1. **DEEP LEARNING**

The dataset was taken from Kaggle that consists data of appx 15000 employees.

The data is preprocessed, trained/fit and finally verified on a smaller dataset. All of these procedures are performed in python as it supports a lot of deep learning based libraries such as Keras, Tensoflow.

Two files namely JSON and hdf5.

JSON holds the weights of the trained ANN.

Hdf5 file holds the ANN model configuration.

These two files are put into a MultiLayerNetwork model generated in java using the library DeepLearning4J.

The input taken from the java FX form is fed into the neural network to predict if the employee will continue in the organization or not.

Using getter – setter methods we pass these into the ANN.

The values are stored in a Special Interface called INDarray that utilises collections library.

The data that is sent to the neural network is analyzed by the model to predict the same.

If the predicted value is below 0.5 it means that the employee might quit, if above 0.5 the employee will not quit.

1. **TECHNOLOGY STACK AND REFERENCES**

* DL4J Library and Documentation.
* Python – data analysis
* Keras and sklearn libraries – ANN development and data preprocessing
* JSON and Hdf5 file format – transferring information between java and python
* JavaFX library – front end development.
* Kaggle – attaining the dataset

1. **REQUIREMENT SPECIFICATIONS**

**Hardware Requirements:**

* Processor (CPU) with 2 gigahertz (GHz) frequency or above
* A minimum of 4 GB of RAM
* Monitor Resolution 1024 X 768 or higher
* A minimum of 20 GB of available space on the hard disk
* Internet Connection Broadband (high-speed) Internet connection with a speed of 4 Mbps or higher

**Software Requirements:**

**Operating Systems**

* Microsoft Windows 8 and above
* Apple macOS 10 and above
* Modern GNU/Linux OS distributions

**Appropriate IDE**

* Eclipse IDE
* Netbeans 8 and above
* IntelliJ IDEA

**SQL Database Software**

* Oracle MySQL
* PostgreSQL
* Apache Derby
* Microsoft SQL Server
* ADABAS

**Browsers**

* Chrome\* 36+
* Edge\* 20+
* Mozilla Firefox 31+
* Internet Explorer 11+ (Windows only)
* Safari 6+ (MacOS only)

**VIII.CONCLUSION**

**HUMAN RESOURCE ANALYSIS**

This application reduces any manual error that might occur. The prediction is more accurate since we are using a well-trained artificial neural network which reduces the relative error that can be caused.

This application can be used to let the employer know if an important employee will quit or not and based on that he can assess the situation and address the issue accordingly.

**STUDENT RECORD MANAGEMENT**

It is helpful is managing new admissions and enrolment

Easier to remove any student data

Parents can view their wards results and other details

**IX. FUTURE SCOPE**

**HUMAN RESOURCE ANALYSIS**

Companies with huge workforce might find it difficult to manage and analyse each employee individually. The HR team can utilize our application to get a bird’s view over the employee’s status. This will greatly reduce the manual burden of going through record of each and every employee. Instead they can quickly analyse and plan their reports accordingly.

**STUDENT RECORD MANAGEMENT**

All admission related work for the student can be done using this system

It provides Electronic workspace

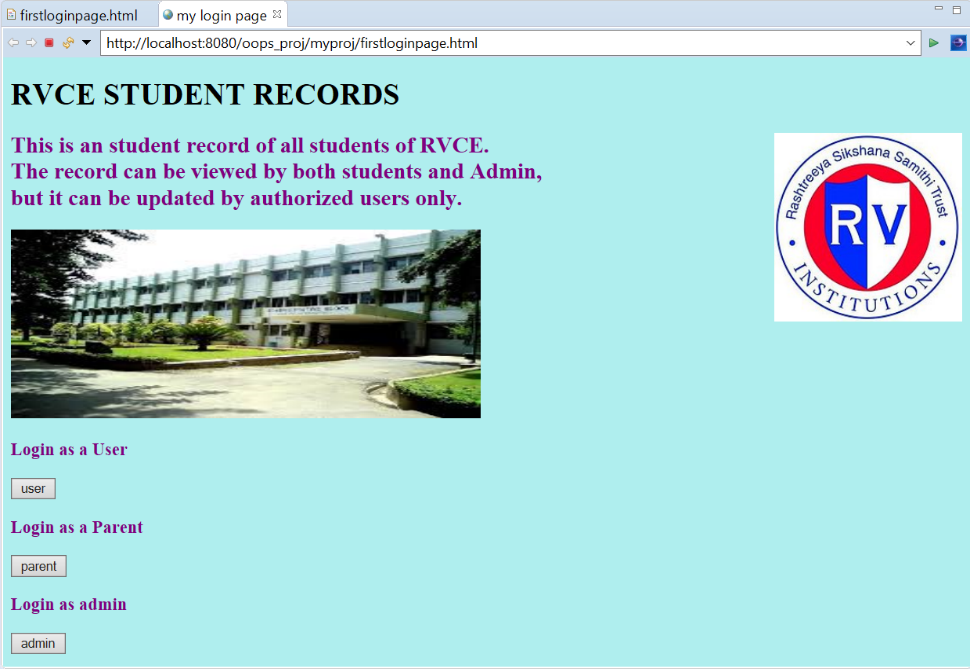
Reduces the burden/workload on the Staff members.

Parents can view their ward details anytime, therefore its convenient.

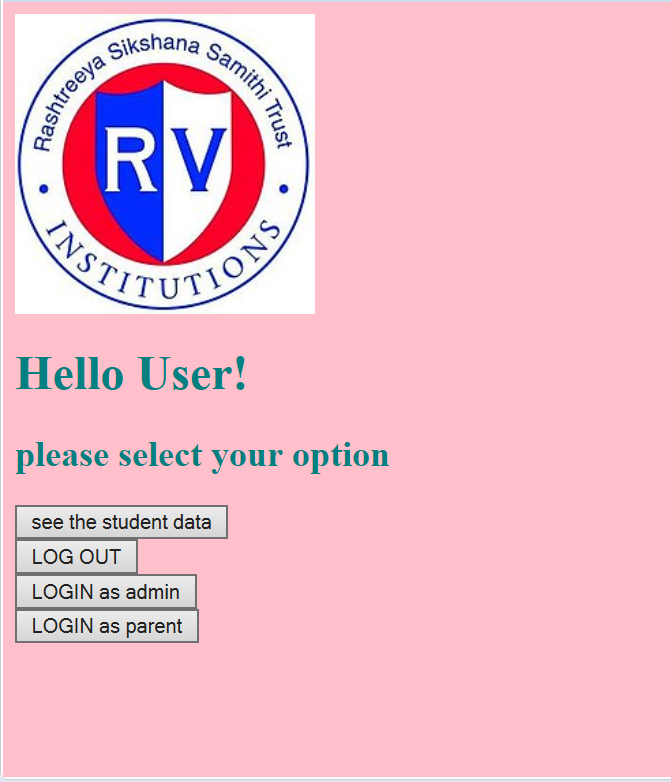
**X.SNAPSHOTS OF OUTPUT**

STUDENT RECORD

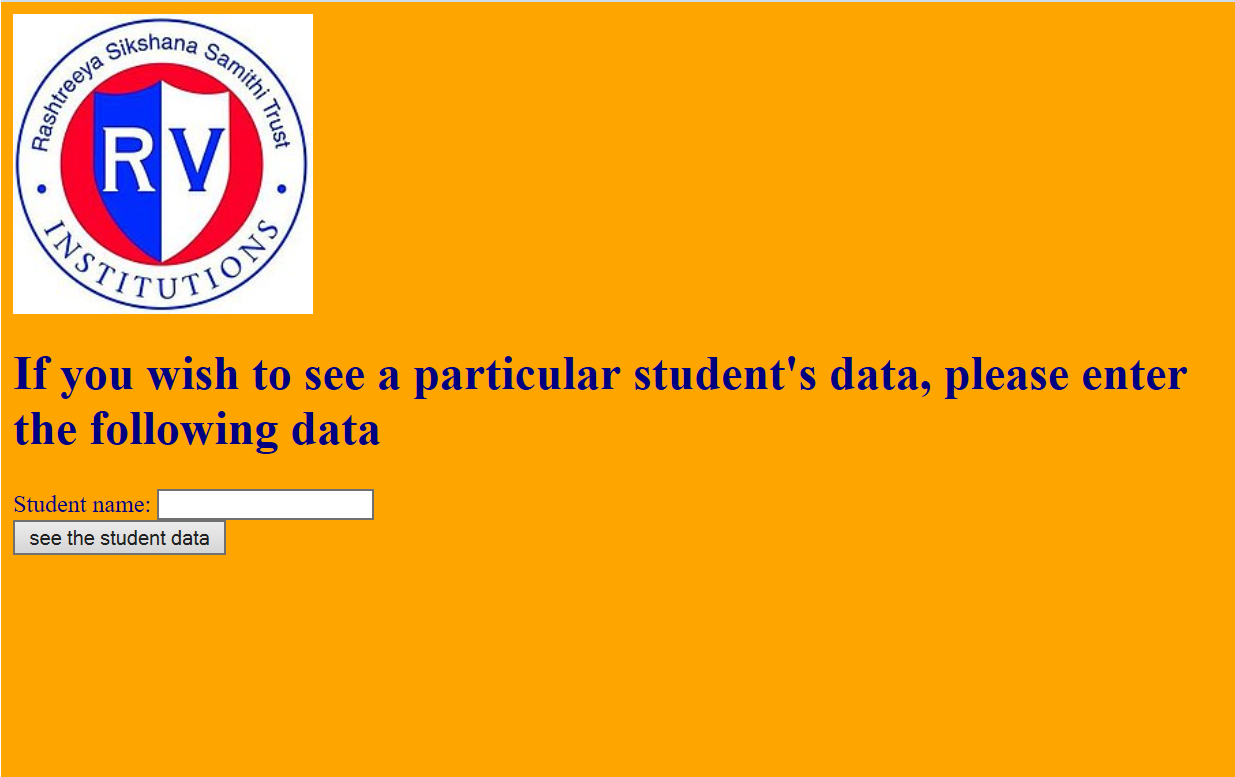
* Login page



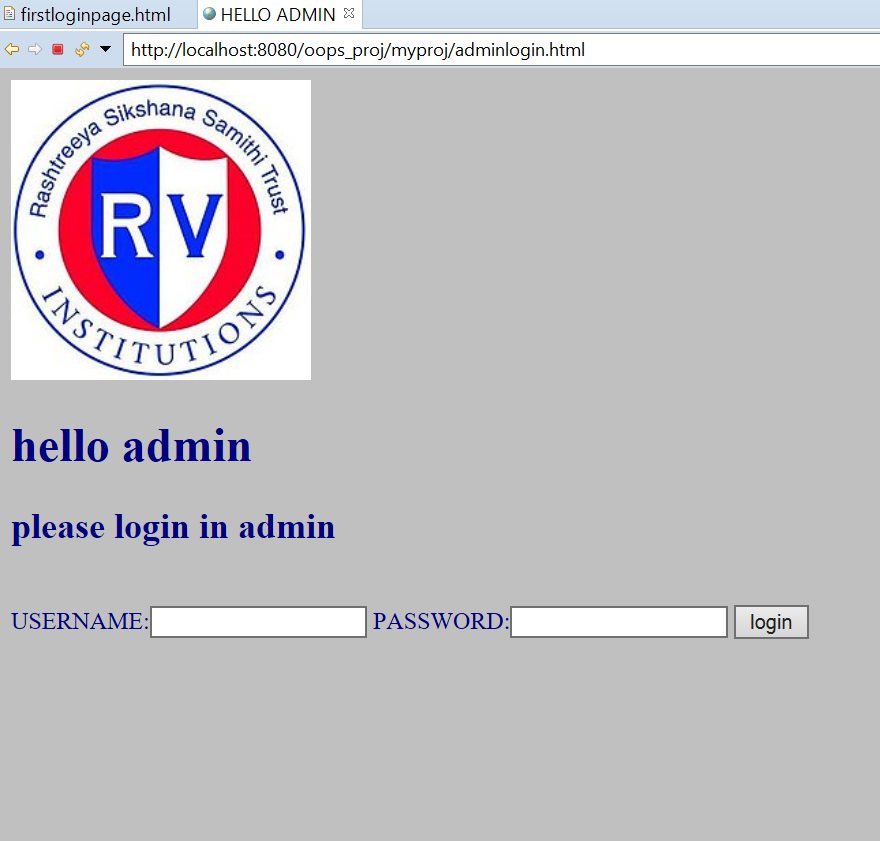
* User login



* Inside user login



* Admin login



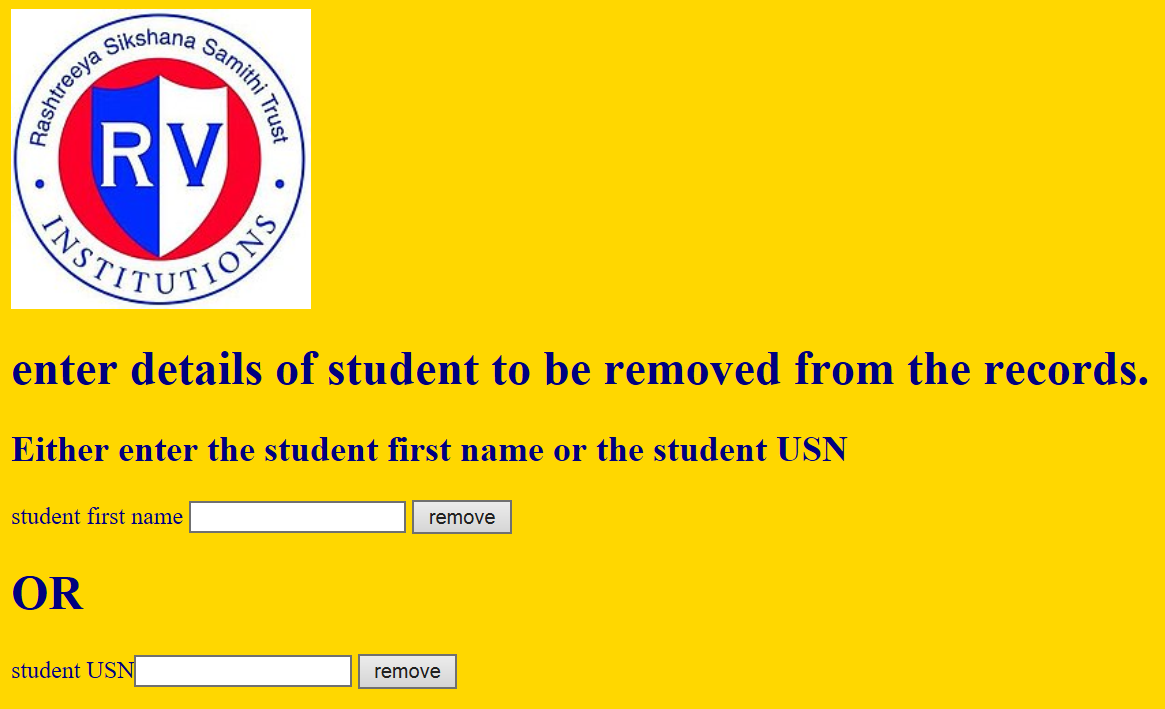
* Inside admin login



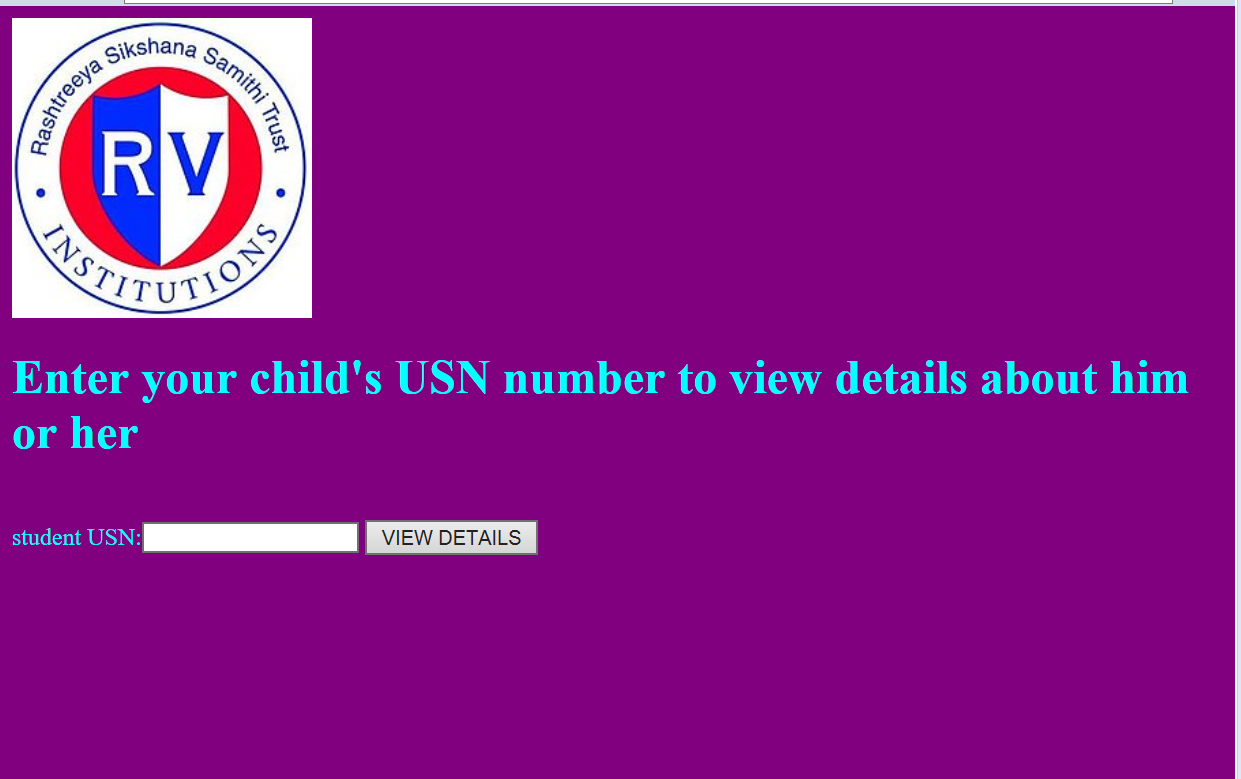
* Add a student record



* Remove a student record

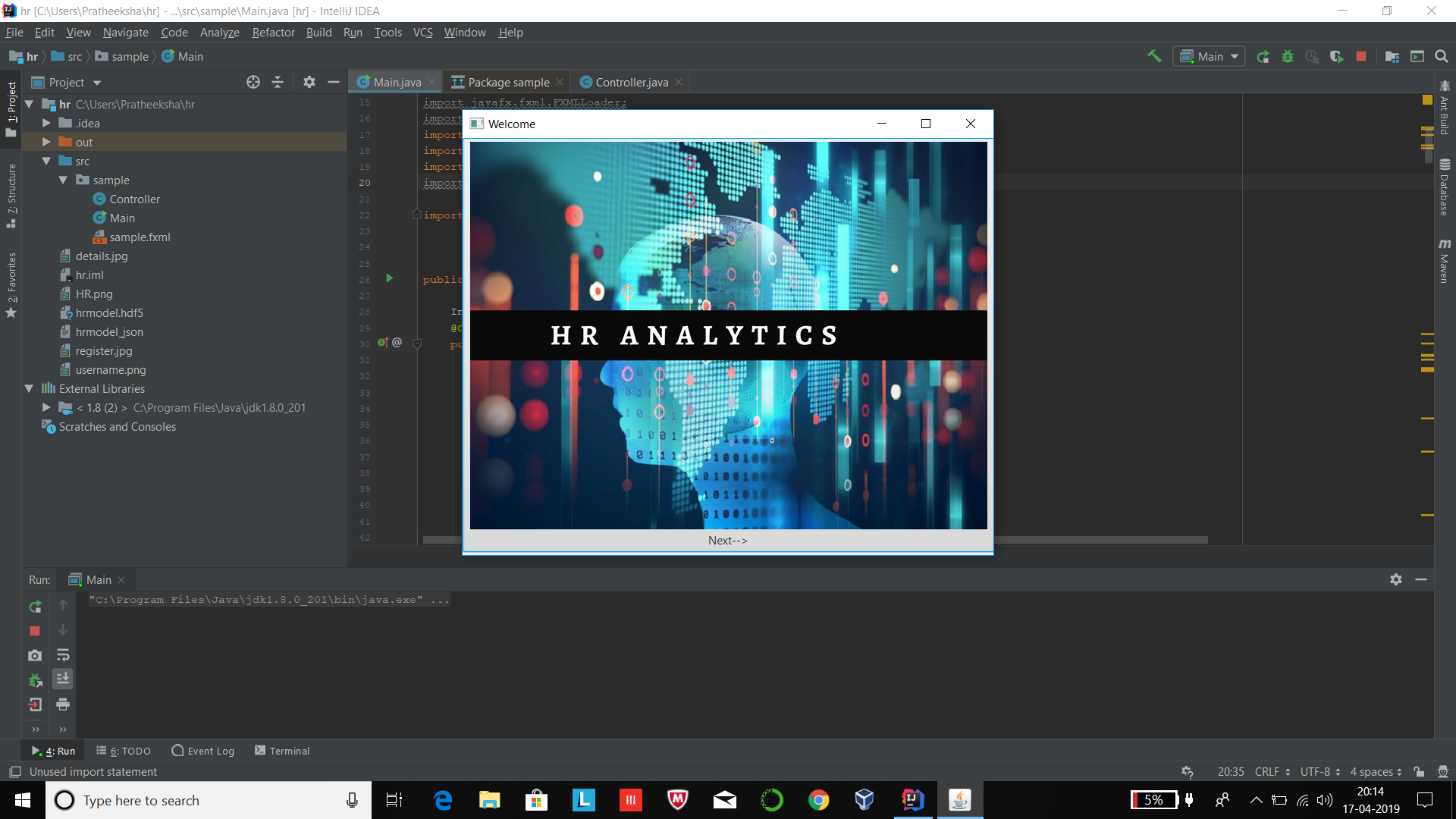


* Parent login

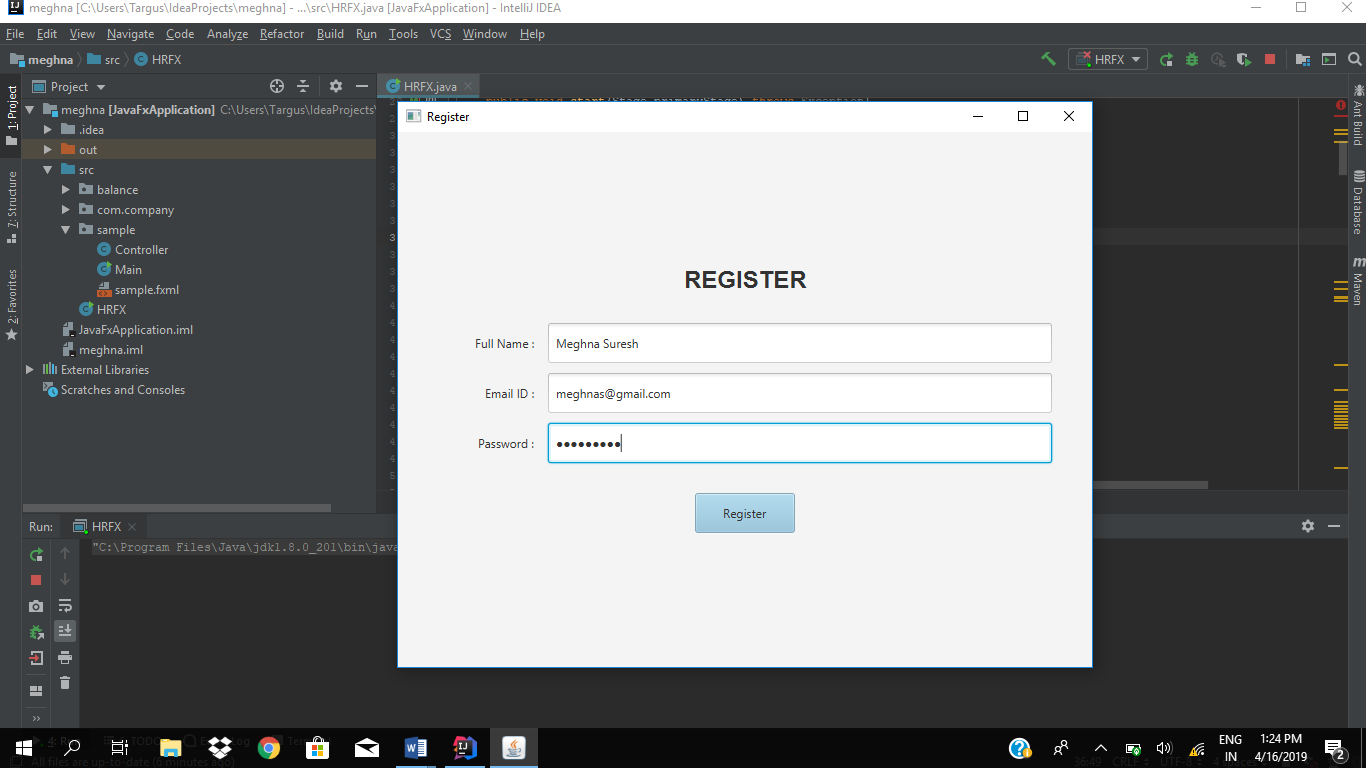


**HR ANALYSIS**

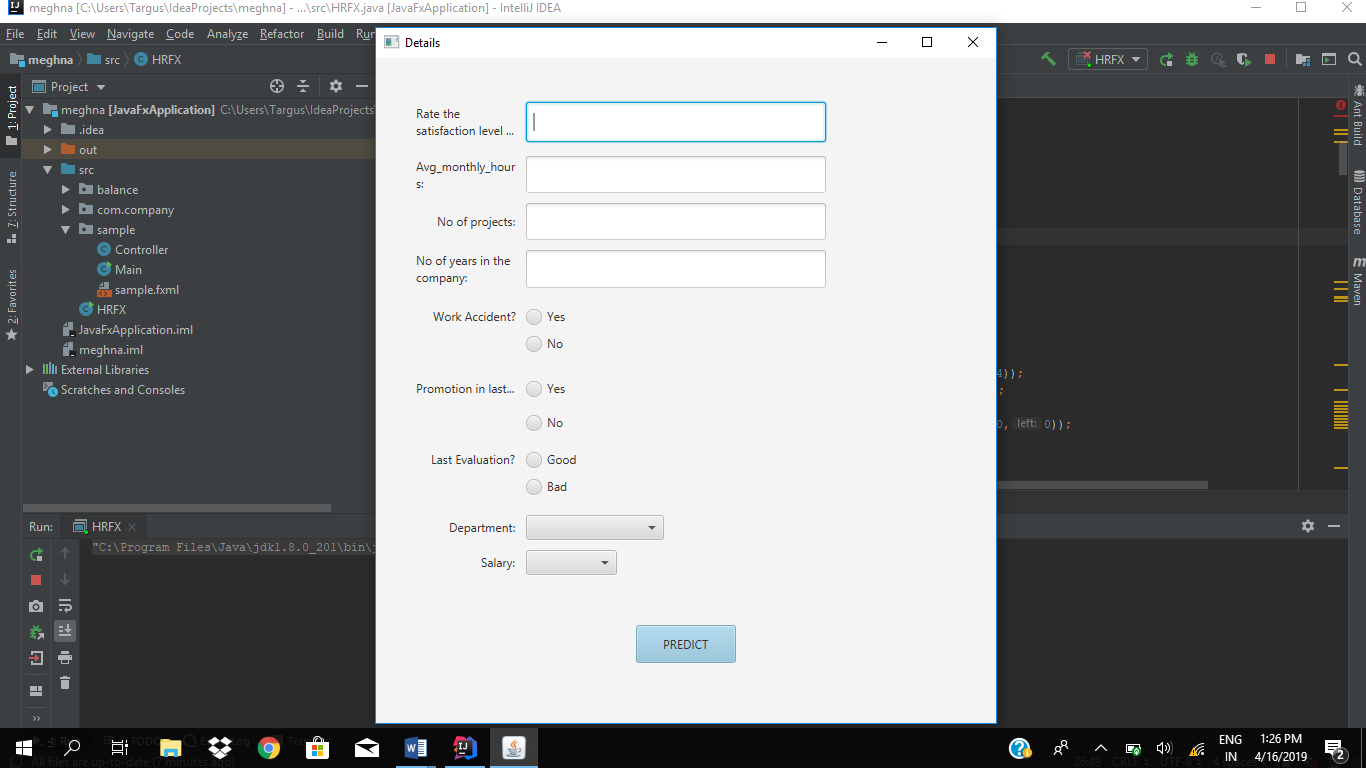
* Welcome page



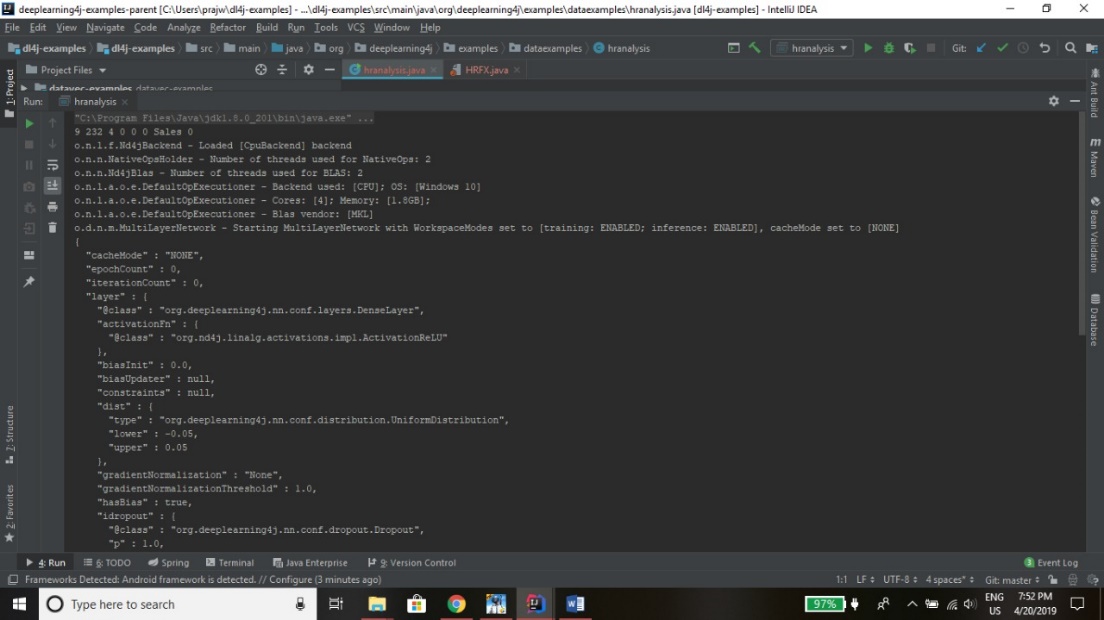
* HR login page

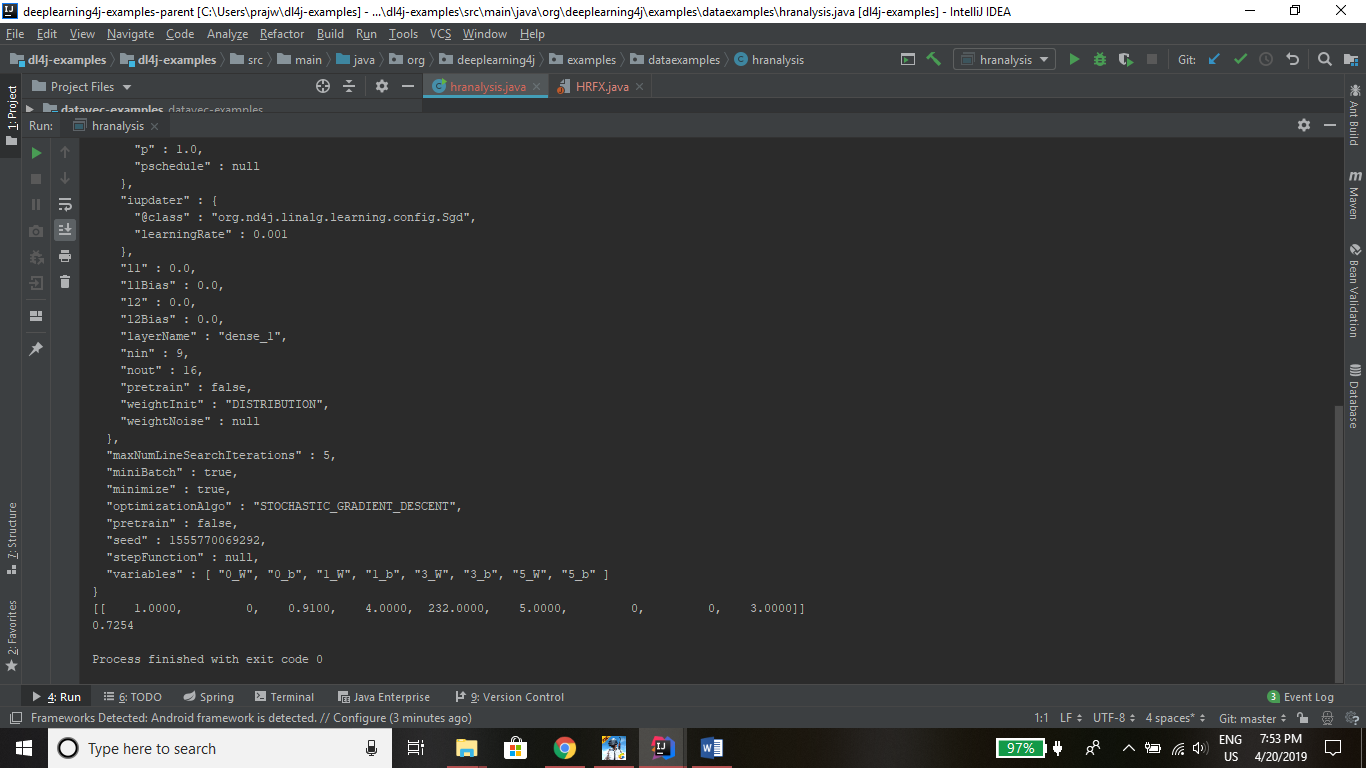


* Employee details



* Output





The final value for the set of inputs given is 0.7354 >0.5. This means the employee is less likely to leave the organization. We predict 1 for values above 0.5 and 0 for values below 0.5 to signify whether or not the employee will leave.

**XI.BIBLIOGRAPHY**

The following documents were used for the implementation of the HR analysis. We would like to thank all the people who were behind this documentation

* Object-Oriented Analysis and Design with Applications by Grady Booch, Robert. A. Maksimchuk, Micheal. W. Engle, Bobbi.J.Young, Jim Conallen, Kelli.A.Houston.
* The Complete Reference 9th Edition Herbert Schildt.
* Documentation of DL4j library.
* Keras Documentation
* Dataset from Kaggle.
* FreeTTS Programmer’s guide which is in the following link:

<https://freetts.sourceforge.io/docs/ProgrammerGuide.html>

* Java Standard Edition Documentation by Oracle which is in the link:

<https://docs.oracle.com/en/java/javase/12/>

**APENDIX A – SOURCE CODE**

**ANN in Python:**

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

# Importing the dataset

dataset = pd.read\_csv('C:\\Users\\prajw\\OneDrive\\Desktop\\HRdata.csv')

X = dataset.iloc[:, :8].values

y = dataset.iloc[:, 9].values

from sklearn.utils import shuffle

X, y = shuffle(X, y)

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

labelencoder\_X\_1 = LabelEncoder()

X[:, 6] = labelencoder\_X\_1.fit\_transform(X[:, 6])

labelencoder\_X\_2 = LabelEncoder()

X[:, 7] = labelencoder\_X\_2.fit\_transform(X[:, 7])

onehotencoder = OneHotEncoder(categorical\_features = [1])

X = onehotencoder.fit\_transform(X).toarray()

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.2, random\_state = 0)

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train\_sc = sc.fit\_transform(X\_train)

X\_test\_sc = sc.transform(X\_test)

from sklearn.model\_selection import KFold

kf = KFold(n\_splits=5, random\_state=None)

for train\_index, test\_index in kf.split(X):

print("Train:", train\_index, "Validation:",test\_index)

X\_train, X\_test = X[train\_index], X[test\_index]

y\_train, y\_test = y[train\_index], y[test\_index]

from keras.models import Sequential

from keras.layers import Dense, Dropout

# Initialising the ANN

classifier = Sequential()

# Adding the input layer and the first hidden layer

classifier.add(Dense(units = 16, kernel\_initializer = 'uniform', activation = 'relu', input\_dim = 9))

# Adding the second hidden layer

classifier.add(Dense(units = 8, kernel\_initializer = 'uniform', activation = 'relu'))

classifier.add(Dropout(0.2))

classifier.add(Dense(units = 6, kernel\_initializer = 'uniform', activation = 'relu'))

classifier.add(Dropout(0.2))

# Adding the output layer

classifier.add(Dense(units = 1, kernel\_initializer = 'uniform', activation = 'sigmoid'))

# Compiling the ANN

classifier.compile(optimizer = 'adam', loss = 'binary\_crossentropy', metrics = ['accuracy'])

# Fitting the ANN to the Training set

classifier.fit(X\_train, y\_train, batch\_size = 10, epochs = 80)

testset = pd.read\_csv('C:\\Users\\prajw\\OneDrive\\Desktop\\test.csv')

A = testset.iloc[:, :8].values

from sklearn.preprocessing import LabelEncoder, OneHotEncoder

labelencoder\_A\_1 = LabelEncoder()

A[:, 6] = labelencoder\_A\_1.fit\_transform(A[:, 6])

labelencoder\_A\_2 = LabelEncoder()

A[:, 7] = labelencoder\_A\_2.fit\_transform(A[:, 7])

onehotencoder = OneHotEncoder(categorical\_features = [1])

A = onehotencoder.fit\_transform(A).toarray()

b = classifier.predict\_classes(A)

b

classifier.save\_weights("hrmodel.hdf5")

model\_json = classifier.to\_json()

tfile =open("hrmodel\_json","w")

tfile.write(model\_json)

tfile.close()

**DL4J Java Code**

package org.deeplearning4j.examples.dataexamples;

import sample.\*;

import org.deeplearning4j.nn.modelimport.keras.exceptions.InvalidKerasConfigurationException;

import org.deeplearning4j.nn.modelimport.keras.exceptions.UnsupportedKerasConfigurationException;

import org.deeplearning4j.nn.multilayer.MultiLayerNetwork;

import org.nd4j.linalg.api.ndarray.INDArray;

import org.nd4j.linalg.factory.Nd4j;

import org.nd4j.linalg.indexing.conditions.Condition;

import org.nd4j.linalg.io.ClassPathResource;

import org.slf4j.Logger;

import org.slf4j.LoggerFactory;

import org.deeplearning4j.nn.modelimport.keras.\*;

import java.io.IOException;

public class hranalysis {

private static Logger log = LoggerFactory.getLogger(hranalysis.class);

public static void main(String [] args) throws IOException, UnsupportedKerasConfigurationException, InvalidKerasConfigurationException {

sample.HRFX ob= new HRFX();

HRFX.main(new String[]{});

String mymodel = new ClassPathResource("hrmodel.hdf5").getFile().getPath();

MultiLayerNetwork model = KerasModelImport.importKerasSequentialModelAndWeights("C:\\Users\\prajw\\hrmodel\_json",mymodel);

System.out.println(model.conf().toJson());

int sala;

int prev\_evaluation=ob.get\_prev\_eval();

if(prev\_evaluation==0)

sala =1;

else

sala=0;

Integer satisfy\_level=ob.get\_satisfaction\_level();

Integer proj\_num =ob.get\_num\_proj();

Integer avg\_mon\_hrs=ob.get\_avg\_monthly\_hours();

Integer years =ob.get\_num\_years();

int workaccident=ob.get\_workaccident();

int prev\_prom=ob.get\_prev\_promotion();

int dep =ob.get\_dept();

INDArray myarray = Nd4j.zeros(1,9);

myarray.putScalar(0,0,sala);

myarray.putScalar(0,1,prev\_evaluation);

myarray.putScalar(0,2,satisfy\_level);

myarray.putScalar(0,3,proj\_num);

myarray.putScalar(0,4,avg\_mon\_hrs);

myarray.putScalar(0,5,years);

myarray.putScalar(0,6,workaccident);

myarray.putScalar(0,7,prev\_prom);

myarray.putScalar(0,8,dep);

INDArray output = model.output(myarray);

System.out.println(myarray);

System.out.println(output);

}

}

**JavaFX:**

package sample;

import javafx.application.\*;

import javafx.collections.FXCollections;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.event.EventHandler;

import javafx.scene.control.\*;

import javafx.geometry.HPos;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.text.Font;

import javafx.scene.text.FontWeight;

import javafx.stage.Window;

import javafx.scene.Parent;

import javafx.scene.\*;

import javafx.scene.layout.\*;

import javafx.stage.\*;

import javafx.animation.Animation;

import javafx.scene.image.\*;

public class HRFX extends Application {

Integer satisfy,sala, avg\_monthly\_hours, num\_proj, num\_years ;

int workacc,prev\_promot,eval, dom;

@Override

public void start(Stage primaryStage) throws Exception{

FlowPane root1=new FlowPane();

primaryStage.setTitle("Welcome");

Scene scene1=new Scene(root1, 700, 550);

primaryStage.setScene(scene1);

Image hr=new Image("file:HR.png");

ImageView hr1=new ImageView(hr);

Button b=new Button("Next-->",hr1);

b.setContentDisplay(ContentDisplay.TOP);

root1.getChildren().addAll(b);

primaryStage.show();

GridPane root2=createRegistrationFormPane();

//addUIControls(root2);

Label headerLabel = new Label("REGISTER");

headerLabel.setFont(Font.font("Arial", FontWeight.BOLD, 24));

root2.add(headerLabel, 0,0,2,1);

GridPane.setHalignment(headerLabel, HPos.CENTER);

GridPane.setMargin(headerLabel, new Insets(20, 0,20,0));

Label nameLabel = new Label("Full Name : ");

root2.add(nameLabel, 0,1);

TextField nameField = new TextField();

nameField.setPrefHeight(40);

root2.add(nameField, 1,1);

Label emailLabel = new Label("Email ID : ");

root2.add(emailLabel, 0, 2);

TextField emailField = new TextField();

emailField.setPrefHeight(40);

root2.add(emailField, 1, 2);

Label passwordLabel = new Label("Password : ");

root2.add(passwordLabel, 0, 3);

PasswordField passwordField = new PasswordField();

passwordField.setPrefHeight(40);

root2.add(passwordField, 1, 3);

Button submitButton = new Button("Register");

submitButton.setPrefHeight(40);

submitButton.setDefaultButton(true);

submitButton.setPrefWidth(100);

root2.add(submitButton, 0, 4, 2, 1);

GridPane.setHalignment(submitButton, HPos.CENTER);

GridPane.setMargin(submitButton, new Insets(20, 0,20,0));

//scene3

GridPane root3=createRegistrationFormPane();

Scene scene3=new Scene(root3,800,800);

Label l1 = new Label();

l1.setText("Rate the satisfaction level on a scale of 10:");

l1.setWrapText(true);

root3.add(l1, 0,0);

TextField t1 = new TextField();

t1.setPrefHeight(40);

t1.setMaxWidth(300);

root3.add(t1, 1,0);

Label l4= new Label("Last Evaluation?");

root3.add(l4, 0,11);

l4.setWrapText(true);

RadioButton good=new RadioButton("Good");

RadioButton bad=new RadioButton("Bad");

ToggleGroup tg2=new ToggleGroup();

good.setToggleGroup(tg2);

bad.setToggleGroup(tg2);

root3.add(good,1,11);

root3.add(bad,1,12);

Label l2= new Label("Avg\_monthly\_hours:");

l2.setWrapText(true);

root3.add(l2, 0,1);

TextField t2 = new TextField();

t2.setPrefHeight(40);

t2.setMaxWidth(300);

root3.add(t2, 1,1);

Label l3= new Label("No of projects:");

root3.add(l3, 0,2);

l3.setWrapText(true);

TextField t3 = new TextField();

t3.setPrefHeight(40);

t3.setMaxWidth(300);

root3.add(t3, 1,2);

Label l5= new Label("No of years in the company:");

root3.add(l5, 0,3);

l5.setWrapText(true);

TextField t5 = new TextField();

t5.setPrefHeight(40);

t5.setMaxWidth(300);

root3.add(t5, 1,3);

Label l6= new Label("Work Accident?");

root3.add(l6, 0,5);

l6.setWrapText(true);

RadioButton yes=new RadioButton("Yes");

RadioButton no=new RadioButton("No");

ToggleGroup tg=new ToggleGroup();

yes.setToggleGroup(tg);

no.setToggleGroup(tg);

root3.add(yes,1,5);

root3.add(no,1,6);

Label l7= new Label("Promotion in last 5 years?");

root3.add(l7, 0,8);

l7.setWrapText(true);

RadioButton yes1=new RadioButton("Yes");

RadioButton no1=new RadioButton("No");

ToggleGroup tg1=new ToggleGroup();

yes1.setToggleGroup(tg1);

no1.setToggleGroup(tg1);

root3.add(yes1,1,8);

root3.add(no1,1,9);

Label l8=new Label("Department:");

root3.add(l8, 0,14);

ObservableList<String> dept= FXCollections.observableArrayList("Accounting","HR","IT","Management",

"Marketing","Product Manager","RndD","Sales","Support","Technical");

ComboBox<String> dep=new ComboBox<String>(dept);

root3.add(dep,1,14);

Label l9=new Label("Salary:");

root3.add(l9, 0,15);

ObservableList<String> salary= FXCollections.observableArrayList("Low","Medium","High");

ComboBox<String> sal=new ComboBox<String>(salary);

root3.add(sal,1,15);

Button predict = new Button("PREDICT");

predict.setPrefHeight(40);

predict.setDefaultButton(true);

predict.setPrefWidth(100);

root3.add(predict, 0, 18, 2, 1);

root3.setHalignment(predict, HPos.CENTER);

root3.setMargin(predict, new Insets(20, 0,20,0));

predict.setOnAction(new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

satisfy = Integer.parseInt(t1.getText());

num\_years = Integer.parseInt(t5.getText());

avg\_monthly\_hours = Integer.parseInt(t2.getText());

num\_proj = Integer.parseInt(t3.getText());

RadioButton rb = (RadioButton) tg.getSelectedToggle();

if (rb.getText().matches("Yes"))

workacc = 1;

else

workacc = 0;

RadioButton rb1 = (RadioButton) tg1.getSelectedToggle();

if (rb1.getText().matches("Yes"))

prev\_promot = 1;

else

prev\_promot = 0;

RadioButton rb2 = (RadioButton) tg2.getSelectedToggle();

if (rb2.getText().matches("Good"))

eval = 1;

else

eval = 0;

String domain = dep.getValue();

switch (domain)

{

case "Accounting":

dom = 2;break;

case "Sales":

dom = 7;break;

case "IT":

dom = 0;break;

case "HR":

dom = 3;break;

case "Technical" :

dom = 9;break;

case "Support":

dom = 8;break;

case "Management":

dom =4;break;

case "Product Manager":

dom = 6;break;

case "Marketing":

dom = 5;break;

case "RndD":

dom = 1;break;

}

String salary=sal.getValue();

if(salary.equals("High"))

sala=1;

else

sala=0

System.out.println(satisfy+" "+avg\_monthly\_hours+" "+num\_proj+" "+workacc+" "+prev\_promot+" "+eval+" "+domain+" "+sala);

}

});

submitButton.setOnAction(new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

if(nameField.getText().isEmpty()) {

showAlert(Alert.AlertType.ERROR, root2.getScene().getWindow(), "Form Error!", "Please enter your name");

return;

}

if(emailField.getText().isEmpty()) {

showAlert(Alert.AlertType.ERROR, root2.getScene().getWindow(), "Form Error!", "Please enter your email id");

return;

}

if(passwordField.getText().isEmpty()) {

showAlert(Alert.AlertType.ERROR, root2.getScene().getWindow(), "Form Error!", "Please enter a password");

return;

}

primaryStage.setScene(scene3);

primaryStage.setTitle("Details");

primaryStage.show();

// showAlert(Alert.AlertType.CONFIRMATION, gridPane.getScene().getWindow(), "", "Welcome " + nameField.getText());

}

});

Scene scene2=new Scene(root2, 700, 550);

b.setOnAction(new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

primaryStage.setScene(scene2);

primaryStage.setTitle("Register");

}

});

}

public int get\_dept()

{

return dom;

}

public Integer get\_salary(){

return sala;

}

public int get\_prev\_eval(){

return eval;

}

public Integer get\_avg\_monthly\_hours()

{

return avg\_monthly\_hours;

}

public Integer get\_satisfaction\_level(){

return satisfy;

}

public Integer get\_num\_proj(){

return num\_proj;

}

public int get\_workaccident(){

return workacc;

}

public int get\_prev\_promotion(){

return prev\_promot;

}

public Integer get\_num\_years(){

return num\_years;

}

private GridPane createRegistrationFormPane() {

// Instantiate a new Grid Pane

GridPane gridPane = new GridPane();

// Position the pane at the center of the screen, both vertically and horizontally

gridPane.setAlignment(Pos.CENTER);

// Set a padding of 20px on each side

gridPane.setPadding(new Insets(40, 40, 40, 40));

// Set the horizontal gap between columns

gridPane.setHgap(10);

// Set the vertical gap between rows

gridPane.setVgap(10);

// Add Column Constraints

// columnOneConstraints will be applied to all the nodes placed in column one.

ColumnConstraints columnOneConstraints = new ColumnConstraints(100, 100, Double.MAX\_VALUE);

columnOneConstraints.setHalignment(HPos.RIGHT);

// columnTwoConstraints will be applied to all the nodes placed in column two.

ColumnConstraints columnTwoConstrains = new ColumnConstraints(200,200, Double.MAX\_VALUE);

columnTwoConstrains.setHgrow(Priority.ALWAYS);

gridPane.getColumnConstraints().addAll(columnOneConstraints, columnTwoConstrains);

return gridPane;

}

private void showAlert(Alert.AlertType alertType, Window owner, String title, String message) {

Alert alert = new Alert(alertType);

alert.setTitle(title);

alert.setHeaderText(null);

alert.setContentText(message);

alert.initOwner(owner);

alert.show();

}

public static void main(String[] args) {

launch(args);

}

}