**CHAPTER 1**

**INTRODUCTION**

**1.1 About the project**

The purpose of making this software is to provide a single-window for all the services within an educational institution. The user can access all services at one place. The target users of this software are Head Of the Department, Class advisor and Students. The invention satisfies the foregoing needs and avoids the drawbacks, limitations and frustrations of the prior art, and provides a better, more timely and effective process of managing the students data at one-hand. The strategy behind this software is Bigdata and NoSQL. Formerly, the data is maintained by handwritten documents, files and online storages. This is a very time consuming process. Reliability, integrity and accuracy of the data is not pretty sure. But, in this project all the needed services are digitized in order to handle the data in an easiest way.

This software makes use of the NoSQL open source database MongoDB for storing the data. It can store texts, images and various file types that are used in the institution. It can be easily scaled for larger dataset. In order to analyse the data, the MapReduce framework of Apache Hadoop is utilized. It splits the process across various nodes in the network and produces the result in a rapid manner.

**1.2 Project description**

The project is developed as various modules in order to maintain the integrity of the system. The modules defined in the system are,

1. GUI module
2. Authentication module
3. Result Analysis module
4. Feedback module
5. Syllabus module
6. Result Display module
7. Attendance Module

**GUI Module:**

Graphical User Interface is designed using Java Swing framework. Swing API is a set of extensible GUI Components to ease the developer's life to create JAVA based Front End/GUI Applications. It is built on top of AWT API and acts as a replacement of AWT API, since it has almost every control corresponding to AWT controls. Swing component follows a Model-View-Controller architecture to fulfil the following criteria.

* A single API is to be sufficient to support multiple look and feel.
* API is to be model driven so that the highest level API is not required to have data.
* API is to use the Java Bean model so that Builder Tools and IDE can provide better services to the developers for use.

**Swing Features**

* **Light Weight** − Swing components are independent of native Operating System's API as Swing API controls are rendered mostly using pure JAVA code instead of underlying operating system calls.
* **Rich Controls** − Swing provides a rich set of advanced controls like Tree, TabbedPane, slider, color picker, and table controls.
* **Highly Customizable** − Swing controls can be customized in a very easy way as visual appearance is independent of internal representation.
* **Pluggable look-and-feel** − SWING based GUI Application look and feel can be changed at run-time, based on available values.

**Authentication Module:**

All the users are authenticated to enter into the software. Admin provides username and password for all users. The credentials are stored in the MongoDB database and while login, the username and password is verified using business logic. If they are same as the data stored in database, then they’ll be allowed to use the software. Three different login forms are created for HoD, Class Advisors and Students respectively. All the users can access only their dashboard. They are restricted to access the others dashboard are service. For example, Student can’t enter into staff portal to update the Attendance or Internal Marks.

**Result Analysis Module:**

In this module, the superiors can able to analyse the students result for a particular amount of periods. For example, we can analyse the pass/fail percentage of students department-wise. The toppers can be easily filtered out using their CGPA. It needs a large amount of data to be searched and filtered. It takes much time if we use SQL databases. But, in our case I used a NoSQL database which supports MapReduce, hence it retrieves filtered data very sooner than SQL databases. It supports easy scalability than SQL. We use **Apache Hadoop’s MapReduce framework** for processing of this large data.

**Feedback Module:**

This module allows students and staffs to communicate in an easiest way. Students can feedback about classes and teachings after Authentication process. They can enter their Name and Register number along with a feedback and submit the form. It will stored in the database along with date and time. Whenever the Superiors requested to view the feedbacks, it will be displayed along with Students name, date and time of submission. It will be very useful for further improvements of the teaching and providing a formal-interaction between staffs and students.

**Syllabus Module:**

When the student wants to view the Syllabus of any subject, he can go to the Syllabus window. To display the PDF file within the system, I used an Adobe Acrobat library acrobat.jar. It produces a new Frame and there we can view the Syllabus. It provides an easy way to Zoom in, Zoom out and Go To options.

**Result Display:**

Now a days, the Semester results are provided through web servers and students can access them from the website. But we can’t store previous semester results at that time. Only current semester is stored in the web database. Because the web server can’t hold that much data at same time. It permits only a specified number of requests at a time. On day of publishing result, the website access is very slow due to the high load of requests. Even we can’t scale the server’s processing speed and database at low cost. For that we’ve provided a service through this application to view the results of any semester at any time. This can be scaled easily when data goes big at low cost.

**Attendance Module:**

Currently, the daily Attendance is entered and stored as a paper only. They always used to calculate the Attendance percentage by hand. It is a very time-consuming process. This module provides an Attendance entry frame using that the Staff can enter the daily attendance through an Excel sheet. We used an external library to write CSV file inside the Application. It will be uploaded to the database. It can be retrieved at any time from the database. It automatically maintains the attendance percentage on each entry. It can be viewed at any time. When students request leave, the Class Advisor can easily check their current Attendance percentage and decide whether allowed or not to take leave.

**CHAPTER 2**

**LITERATURE SURVEY**

**Big Data Analysis using Hadoop: A Survey**

|  |  |
| --- | --- |
| **Rotsnarani Sethy**  PhD Scholar, Dept. of Computer Science Utkal University Bhubaneswar, India | **Mrutyunjaya Panda**  Reader, Dept. of Computer Science  Utkal University Bhubaneswar, India |

***Abstract— Big data is a collection of large data sets that include different types such as structured, unstructured and semi-structured data. This data can be generated from different sources like social media, audios, images, log files, sensor data, transactional applications, web etc. To process or analyse this huge amount of data or extracting meaningful information is a challenging task now a days. Big data exceeds the processing capability of traditional database to capture, manage, and process the voluminous amount of data. In this paper we first introduce the general background of big data and then focus on Hadoop platform using map reduce algorithm which provide the environment to implement application in distributed environment and it can capable of handling node failure.***

***Keywords— Big Data, Hadoop, HDFS, Map Reduce, Hadoop Components.***

**I. INTRODUCTION**

In general, big data shall mean the datasets that could not be perceived, acquired, managed, and processed by traditional IT and software/hardware tools within a tolerable time. Big Data describes any massive volume of structured, semi structured and unstructured data that are difficult to process using traditional database system such as RDBMS [1]. An example of big data may be Exabyte‟s (1024 terabytes) of data consisting of trillions of records of millions of people from different sources such as websites, social media, mobile data, web servers, online transactions and so on [2]. In the past, type of information available was limited. There was a well-defined set of technology approaches for managing information. But in today’s world, the amount of data has been exploding. It has grown to terabytes and petabytes. Because in every minute, there are 280,000 tweets, more than 100 million emails are sent. Some of the applications of big data are in areas such as social media, healthcare, traffic management, banking, retail, education and so on.

***A. CHARACTERISTICS OF BIG DATA***

As the data is too big and comes from various sources in different form, it is characterized by the following five components:

* **VARIETY**

Data being produced is not of single category as it not only includes the traditional data but also the semi structured data from various resources like web Pages, Web Log Files, social media sites, e-mail, documents, sensor devices data both from active passive devices. All this data is totally different consisting of raw, structured, semi structured and even unstructured data which is difficult to be handled by the existing traditional analytic systems.

* **VOLUME**

The Big word in Big data itself defines the volume. At present the data existing is in petabytes and is supposed to increase to zettabytes in nearby future. The social networking sites existing are themselves producing data in order of terabytes every day and this amount of data is definitely difficult to be handled using the existing traditional systems.

* **VELOCITY**

Velocity in Big data is a concept which deals with the speed of the data coming from various sources. This characteristic is not being limited to the speed of incoming data but also speed at which the data flows. For example, the data from the sensor devices would be constantly moving to the database store and this amount won’t be small enough. Thus our traditional systems are not capable enough on performing the analytics on the data which is constantly in motion.

* **VARIABILITY**

Variability considers the inconsistencies of the data flow. Data loads become challenging to be maintained especially with the increase in usage of the social media which generally causes peak in data loads with certain events occurring.

* **COMPLEXITY**

It is quite an undertaking to link, match, cleanse and transform data across systems coming from various sources. It is also necessary to connect and correlate relationships, hierarchies and multiple data linkages or data can quickly spiral out of control.

**II. TOOLS AND TECHNOLOGIES**

For the purpose of processing the large amount of data, the big data requires exceptional technologies. The various techniques and technologies have been introduced for manipulating, analysing and visualizing the big data. There are many solutions to handle the Big Data, but the Hadoop is one of the most widely used technologies [3].

***A. HADOOP***

Using the solution provided by Google, Doug Cutting and his team developed an Open Source Project called HADOOP. Hadoop runs applications using the Map Reduce algorithm, where the data is processed in parallel with others. Hadoop is used to develop applications that could perform complete statistical analysis on huge amounts of data. Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage. It is quite expensive to build bigger servers with heavy configurations that handle large scale processing, but as an alternative, you can tie together many commodity computers with single-CPU, as a single functional distributed system and practically, the clustered machines can read the dataset in parallel and provide a much higher throughput. Moreover, it is cheaper than one high-end server. So this is the first motivational factor behind using Hadoop that it runs across clustered and low-cost machines. Hadoop runs code across a cluster of computers. This process includes the following core tasks that Hadoop perform. Data is initially divided into directories and files. Files are divided into uniform sized blocks of 128M and 64M (preferably 128M).These files are then distributed across various cluster nodes for further processing. HDFS, being on top of the local file system, supervises the processing [8]. Blocks are replicated for handling hardware failure. Checking that the code was executed successfully, performing the sort that takes place between the map and reduce stages, sending the sorted data to a certain computer, writing the debugging logs for each job. Hadoop has two major layers namely:

* Processing/Computation layer (Map Reduce), and
* Storage layer (Hadoop Distributed File System).

***B. MAP REDUCE***

Map Reduce is a parallel programming model for writing distributed applications devised at Google for efficient processing of large amounts of data (multiterabyte data-sets), on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. The Map Reduce program runs on Hadoop which is an Apache open-source framework [4]. It is a processing technique and a program model for distributed computing based on java. The Map Reduce algorithm contains two important tasks, namely Map and Reduce. Map takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). Secondly, reduce task, which takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name Map Reduce implies, the reduce task is always performed after the map job. The major advantage of Map Reduce is that it is easy to scale data processing over multiple computing nodes. Under the Map Reduce model, the data processing primitives are called mappers and reducers. Decomposing a data processing application into mappers and reducers is sometimes nontrivial. But, once we write an application in the Map Reduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a configuration change. This simple scalability is what has attracted many programmers to use the Map Reduce model.

**The stages of Map Reduce Program**

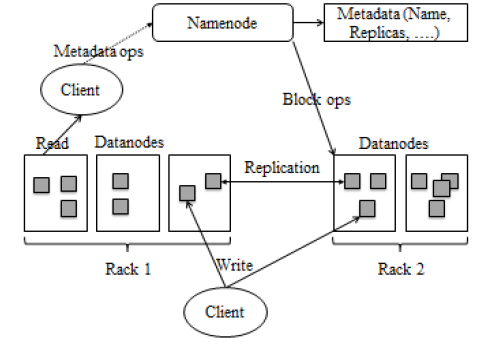
Generally Map Reduce paradigm is based on sending the computer to where the data resides! Map Reduce program executes in two stages, namely map stage and reduce stage.

* Map stage: The map or mapper’s job is to process the input data. Generally the input data is in the form of file or directory and is stored in the Hadoop file system (HDFS). The input file is passed to the mapper function line by line. The mapper processes the data and creates several small chunks of data.
* Reduce stage: This stage is the combination of the Shuffle stage and the Reduce stage. The Reducer’s job is to process the data that comes from the mapper. After processing, it produces a new set of output, which will be stored in the HDFS

During a Map Reduce job, Hadoop sends the Map and Reduce tasks [5] to the appropriate servers in the cluster. The framework manages all the details of data-passing such as issuing tasks, verifying task completion, and copying data around the cluster between the nodes. Most of the computing takes place on nodes with data on local disks that reduces the network traffic. After completion of the given tasks, the cluster collects and reduces the data to form an appropriate result, and sends it back to the Hadoop server.

***C. HADOOP DISTRIBUTED FILE SYSTEM (HDFS)***

The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS) and provides a distributed file system that is designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant.

******

**Fig 2.1: HDFS Architecture**

It is highly fault-tolerant and is designed to be deployed on low-cost hardware. It provides high throughput access to application data and is suitable for applications having large datasets. HDFS holds very large amount of data and provides easier access. To store such huge data, the files are stored across multiple machines. These files are stored in redundant fashion to rescue the system from possible data losses in case of failure. It is suitable for the distributed storage and processing. Hadoop provides a command interface to interact with HDFS. The built-in servers of name node and data node help users to easily check the status of cluster. HDFS provides file permissions and authentication.Fig.1 briefly describe the HDFS architecture.

HDFS follows the master-slave architecture and it has the following elements.

* **NAME NODE**

The name node is the commodity hardware that contains the GNU/Linux operating system and the name node software. It is software that can be run on commodity hardware. The system having the name node acts as the master server and it does the following tasks: Manages the file system namespace. Regulates client’s access to files and It also executes file system operations such as renaming, closing, and opening files and directories.

* **DATA NODE**

The data node is a commodity hardware having the GNU/Linux operating system and data node software. For every node (Commodity hardware/System) in a cluster, there will be a data node. These nodes manage the data storage of their system. Data nodes perform read-write operations on the file systems, as per client request. They also perform operations such as block creation, deletion, and replication according to the instructions of the name node.

* **BLOCK**

Generally the user data is stored in the files of HDFS. The file in a file system will be divided into one or more segments and/or stored in individual data nodes. These file segments are called as blocks. In other words, the minimum amount of data that HDFS can read or write is called a Block. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

**OTHER DIFFERENT COMPONENTS [7] OF HADOOP ARE:**

**Apache Pig**: software for analysing large data sets that consists of a high-level language similar to SQL for expressing data analysis programs, coupled with infrastructure for evaluating these programs. It contains a compiler that produces sequences of Map- Reduce programs.

**HBase:** non-relational columnar distributed database designed to run on top of Hadoop Distributed File system (HDFS). It is written in Java and modelled after Google‟s Big Table. HBase is an example of a NoSQL data store.

**Hive**: it is Data warehousing application that provides the SQL interface and relational model. Hive infrastructure is built on the top of Hadoop that help in providing summarization, query and analysis.

**Cascading**: software abstraction layer for Hadoop, intended to hide the underlying complexity of Map Reduce jobs. Cascading allows users to create and execute data processing workflows on Hadoop clusters using any JVM-based language.

**Avro**: it is a data serialization system and data exchange service. It is basically used in Apache Hadoop. These services can be used together as well as independently.

**Big Top**: It is used for packaging and testing the Hadoop ecosystem.

**Oozie**: Oozie is a java based web-application that runs in a java servlet. Oozie uses the database to store definition of Workflow that is a collection of actions. It manages the Hadoop jobs.

So there are many advantages of hadoop that are: Hadoop framework allows the user to quickly write and test distributed systems. It is efficient, and it automatic distributes the data and work across the machines and in turn, utilizes the underlying parallelism of the CPU cores. Hadoop does not rely on hardware to provide fault-tolerance and high availability (FTHA), rather Hadoop library itself has been designed to detect and handle failures at the application layer. Servers can be added or removed from the cluster dynamically and Hadoop continues to operate without interruption [9]. Another big advantage of Hadoop is that apart from being open source, it is compatible on all the platforms since it is Java based.

**III. DATA MINING FOR BIG DATA**

Data mining is the process of extracting information from a large data sets and transform it into an understandable form for further use. Data mining can be used in such a case where database is large and the classification of such a data is difficult. There are many techniques used in data mining to process and mine the uncertain data [10]. Clustering is the important technique in data analysis and data mining applications. It is the task of grouping a set of objects so that objects in the same group are more similar to each other than to those in other groups called clusters. The research in Big Data analysis using data mining especially with clustering technique still considered to be young, which attracts many researchers to conduct further research in this potential area.

**IV. LITERATURE REVIEW**

The authors [1] pointed out that handling of huge data using earlier RDBMS tools is little bit complex, hence feels the necessity of alternate tools that can handle such a huge data which is usually referred to as „big data***”.*** In this, the authors argued that big data differs from other data in 5 dimensions such as volume, velocity, variety, value and complexity. They illustrated the Hadoop architecture consisting of name node, data node, edge node, HDFS to handle big data systems. The term “Big data” is used for large data sets whose size is beyond the ability of commonly used software tools to capture, manage, and process the data within a tolerable elapsed time. Big data sizes are a constantly moving target currently ranging from a few dozen terabytes to many petabytes of data in a single data set. Difficulties include capture, storage, search, sharing, analytics and visualizing. The Authors [3] **have** done a lot of experiment on the big data problem. At last he found that the Hadoop cluster, Hadoop Distributed File System (HDFS) for storage and map reduce method for parallel processing on a large volume of data. The Authors [4] emphasizes on a prominent data processing tool Map Reduce survey which will help in understanding various technical aspects of the Map Reduce framework. In this survey, the author expresses different views on Map Reduce framework and introduces its optimization strategies. Author also hands a challenge on parallel data analysis with Map Reduce framework. The Authors [5] defines big data Problem using Hadoop and Map Reduce” reports the experimental research on the big data problems in various domains. It describe the optimal and efficient solutions using Hadoop cluster, Hadoop Distributed File System (HDFS) for storage data and Map Reduce framework for parallel processing to process massive data sets and records. The Authors [6] discussed an overview of big data's concept, tools, techniques, applications, advantages and challenges. They used Hadoop technology for the implementation purpose. The authors have briefly discussed about HDFS and Map Reduce technology to process massive data sets and records. The Authors [7] Stated that streaming data analysis in real time is becoming the fastest and most efficient way to obtain useful knowledge, allowing organizations to react quickly when problem appear or detect to improve performance. Huge amount of data is created everyday termed as “big data”. The tools used for mining big data are apache Hadoop, apache pig, cascading, scribe, storm, apache HBase, apache mahout, MOA, etc. Thus, he instructed that our ability to handle many Exabytes of data mainly dependent on existence of rich variety dataset, technique, software framework. The Authors [8] discussed that the big data refers to a collection of vast amount of structured, unstructured and semi structured data that are very difficult to manage, process and to store using common database management system. To store and manage this huge amount of different data the data storage technique are used such as clustered network attached storage (NAS) and object based storage. The Hadoop architecture is best in this case to manage and different data structure using map reduce method. The Authors [9] talks about the theoretical assumptions, that improves the performance of Hadoop/map reduce and purposed the optimal reduce task assignment schemes that minimize the fetching cost per job and performs the both simulation and real system deployment with experimental evolution. The advantage of this paper is improves the performance of large scale Hadoop clusters. The disadvantage of this paper is environmental factors such as network topologies effect on a reduce task in map reduce clusters. The Authors [10] Discussed about the Data Mining and some Clustering Techniques for the criteria’s of big data. They also stated that with the beginning in the era of big data, the data is increasing at rapid speed not only in size but also in variety. There come challenges and difficulties to handle such large amount of data with the growing data. Big data exhibits different characteristics like volume, variety, variability, value, velocity and complexity due to which it is very difficult to analyse data and obtain information with traditional data mining techniques.

**CHAPTER 3**

**SYSTEM REQUIREMENTS**

**Hardware and Software Requirements**

**Hardware Requirements**

The hardware requirements may serve as the basic for a contract for the implementation of the system and should therefore be a complete and consistent specification of the whole design.

Processor : Intel Core I5

Target Device : Windows and Linux Systems

Primary Memory : 8GB

Secondary Memory : 100GB

Input Device : Mouse & Keyboard

Output Device : VGA Monitor

**Software Requirements**

The software requirements document is the specification of the system. It should include both definition and specification requirements. It is a set of what the system should do rather than how it should do it.

The software requirements provide a basis for creating the software requirements specification. It is useful in estimation cost, planning, team activities, performance tasks and tracking the teams and tracking the team’s progress.

Operating System : Windows 10

Front End : Java Swing, AWT

Back End : Java, MongoDB, Hadoop MapReduce

Development Tool : Eclipse IDE

Emulator : Java Virtual Machine

**CHAPTER 4**

**SYSTEM DESIGN**

**4.1 Overview:**

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modelling its process aspects. A DFD is often used as a preliminary step to create an overview of the system without going into great detail, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

A DFD shows what kind of information will be input to system and output from the system, how the data will advance through the system, and where the data will be stored . It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel unlike a flowchart which also shows this information.

 DFD is a designing tool used in the top-down approach to system design. The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole.

**4.2 Representation of Components:**

DFDs only involve four symbols. They are:

* Process
* Data Object
* Data Store
* External entity

|  |  |
| --- | --- |
| http://members.tripod.com/~myyee/cs457/process.gif | **Process** Transform of incoming data flow(s) to outgoing flow(s). |
| http://members.tripod.com/~myyee/cs457/data.gif | **Data Flow** Movement of data in the system. |  |
| http://members.tripod.com/~myyee/cs457/datastore.gif | **Data Store** Data repositories for data that are not moving. It may be as simple as a buffer or a queue or a s sophisticated as a relational database. |  |
| http://members.tripod.com/~myyee/cs457/external.gif | **External Entity** Sources of destinations outside the specified system boundary. |  |

**CHAPTER 5**

**PROJECT DESCRIPTION**

**5.1 Scope of the project:**

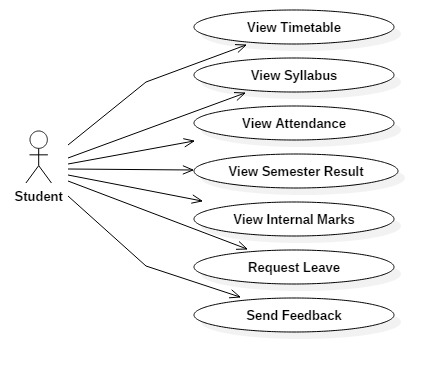
The scope of this project is to create a Desktop Application that can be run on any operating system platform. This Application is targeted for Staffs, HoDs, and Students of the institution.

It enables the user to do and get their work done in a single-window and gives an elegant experience. It is written on Java. Java is an open-source programming language and a platform that can run on any operating system instance.

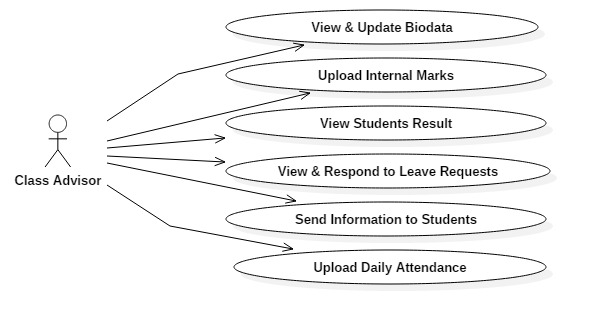
Students can view their class timetable, internal marks and semester result. They can send feedback about the class through this application. They can view and print the syllabus of the subjects via this application. They can also send Leave Requests to their class advisors and get permission from them via a formal manner.

Class Advisor can view and update the biodata of all their students in a easy way. He/ She can upload the internal marks of students as a CSV file format to this application. He/ She can view the semester result of their students as soon as the result has been published. He/ She can enter daily attendance and can view the attendance percentage of students at any time. He/ She can send information to students through mail and mobile number instantly through this handy application. He/ She can also view the Leave Requests and respond to them.

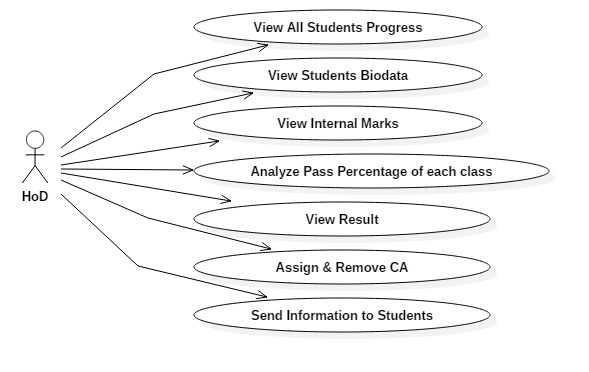
HoD can view the progress of his department students at a single hand. He/ She can view the biodata, internal marks and semester result of all students of the department. He/ She can analyse the pass percentage of each class. He/ She can assign and remove CAs of a class. He/ She can also send information to students instantly through mail and mobile number.



**Fig 5.1.1: Use case diagram - Students**



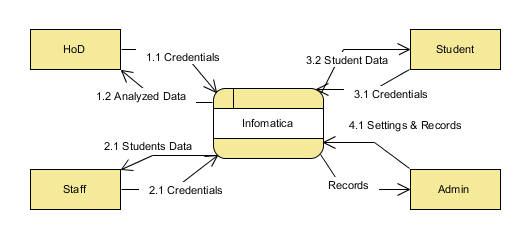
**Fig 5.1.2: Use case diagram – Class Advisor**



**Fig 5.1.3: Use case diagram - HoD**

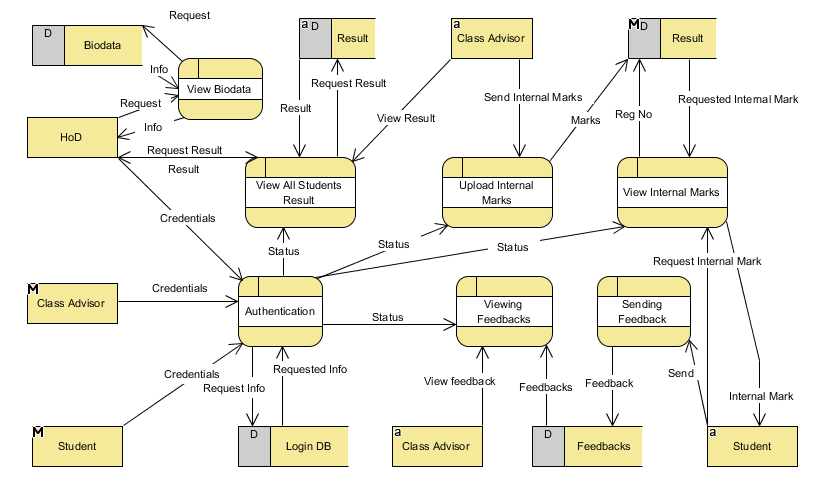
**5.2 System Architecture and Design:**

**Context Diagram:**

****

**Fig 5.2.1: Context Diagram**

**Level 1 DFD:**

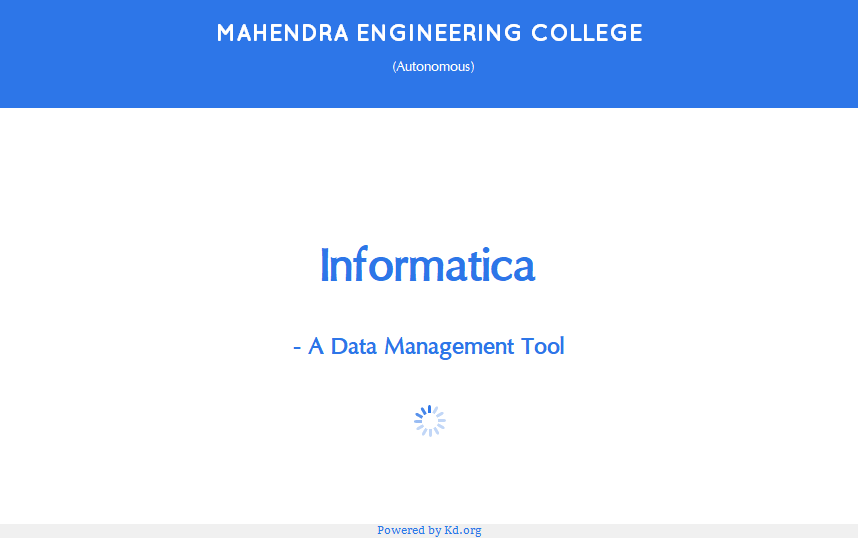
****

**Fig 5.2.2: Level 1 DFD**

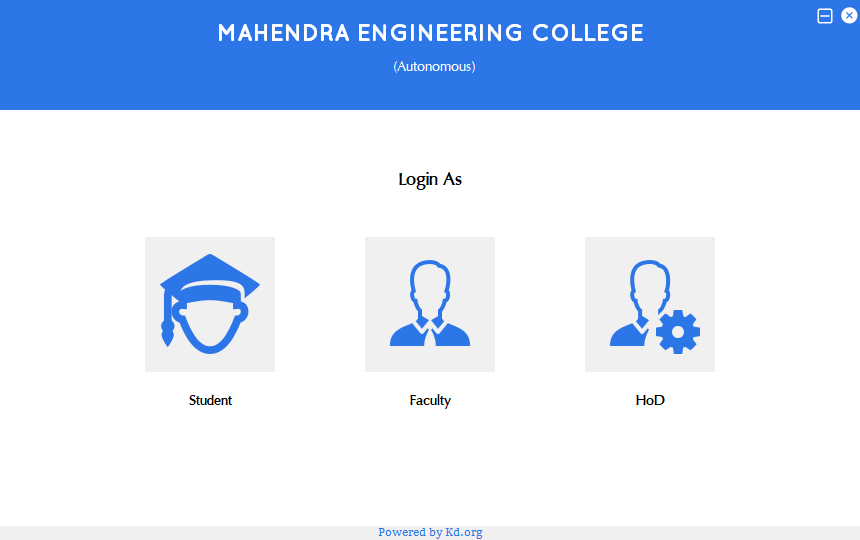
**CHAPTER 6**

**APPENDIX**

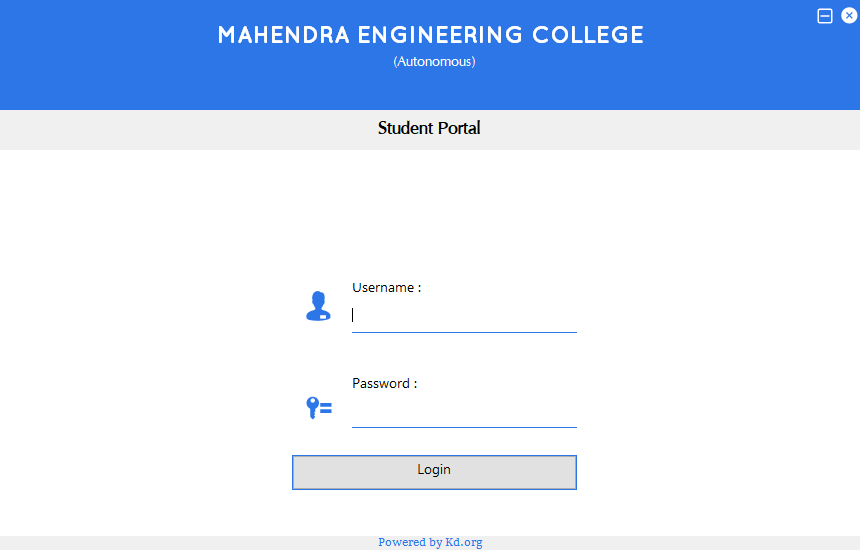
**6.1 Screenshots:**

****

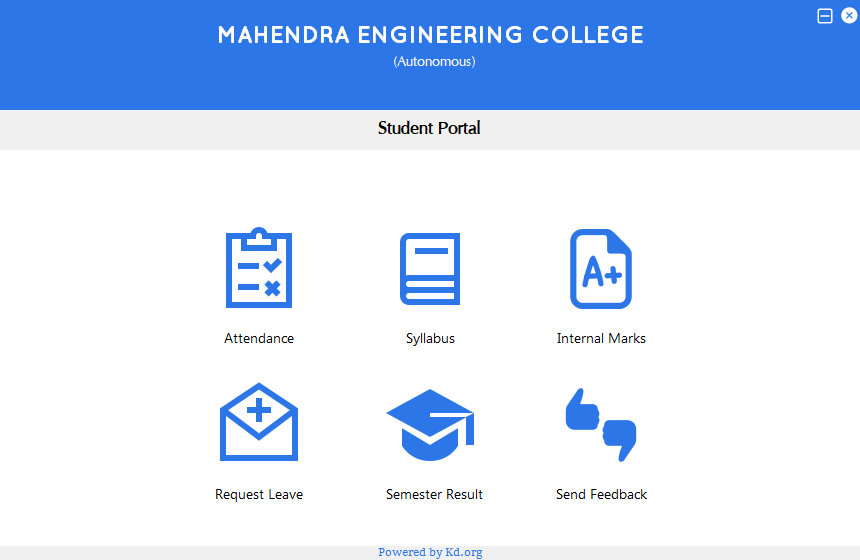
**Fig 6.1.1: Title**

****

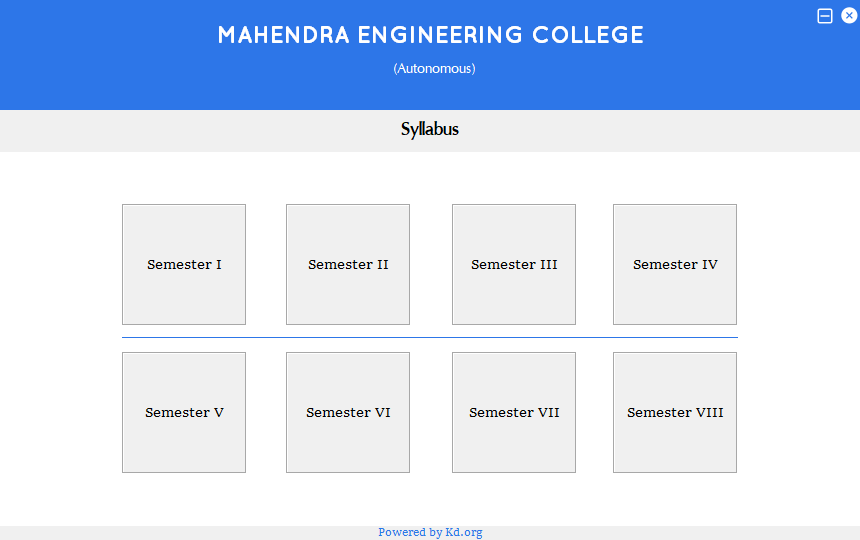
**Fig 6.1.2: Home**

****

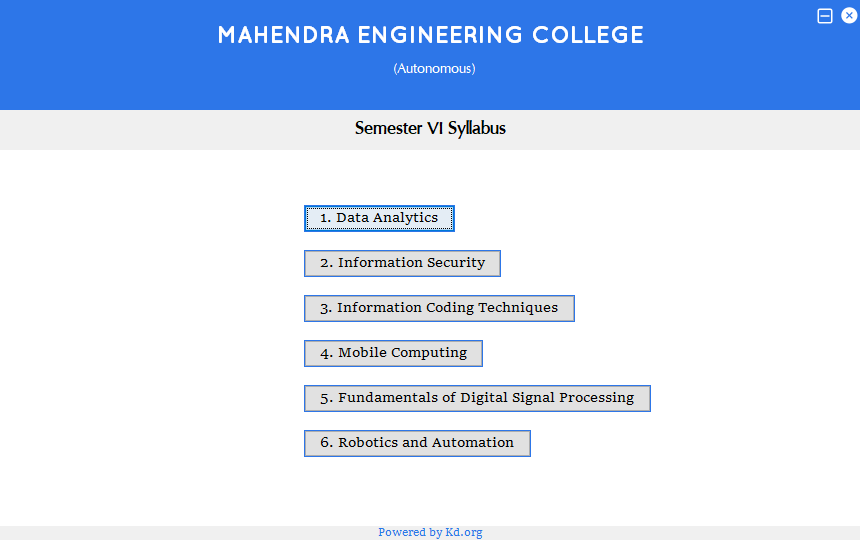
**Fig 6.1.3: Student Login**

****

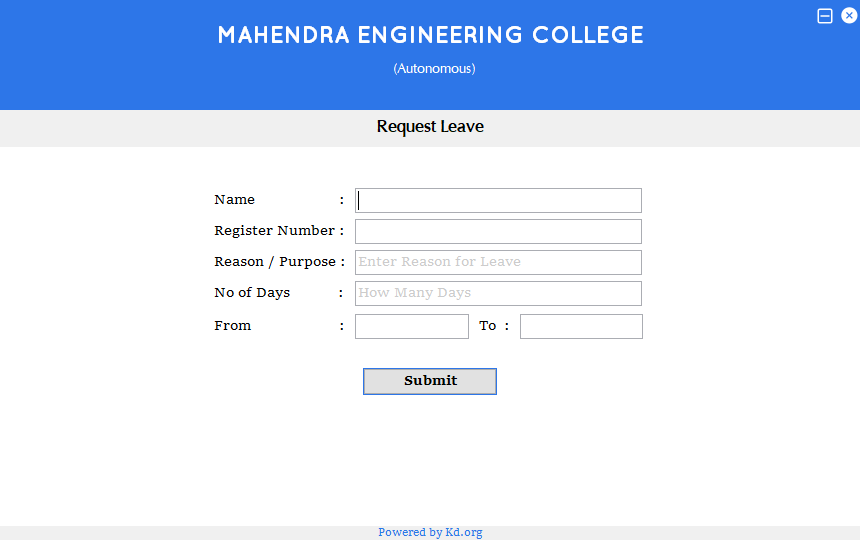
**Fig 6.1.4: Student Portal**

****

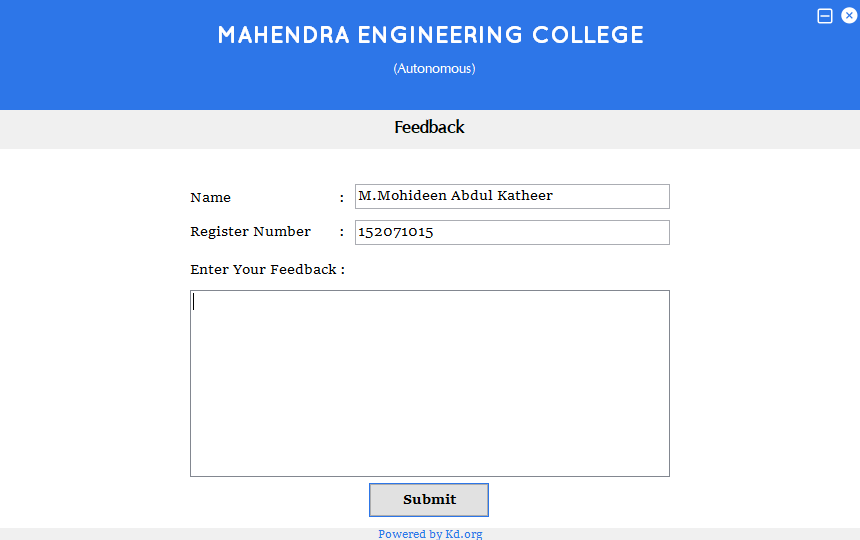
**Fig 6.1.5: Syllabus**

****

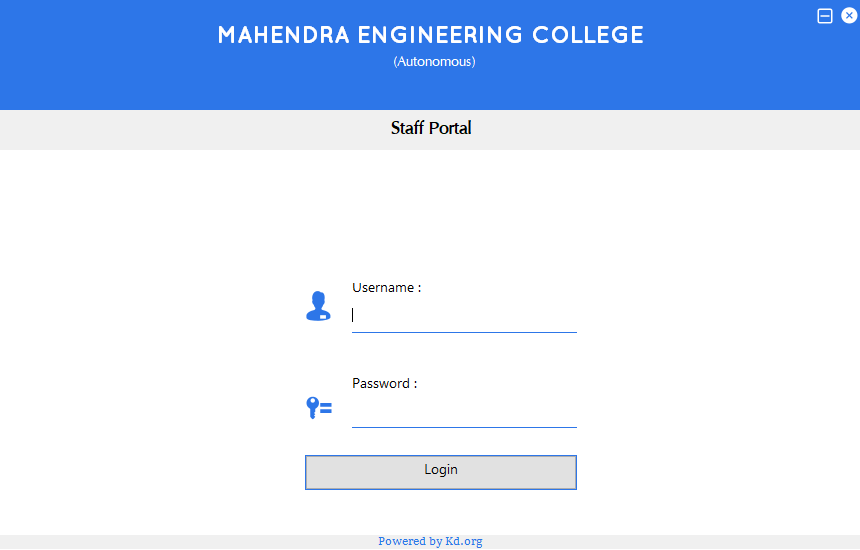
**Fig 6.1.6: Sem VI Syllabus**

****

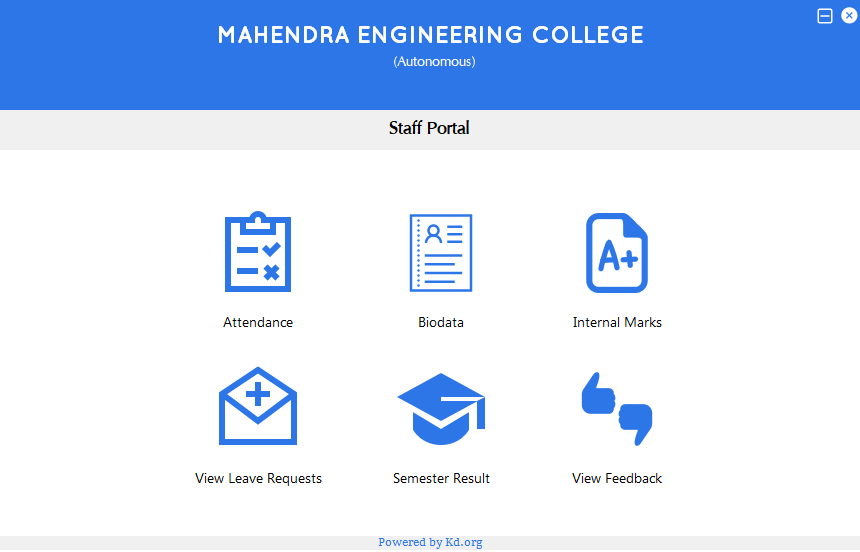
**Fig 6.1.7: Request Leave**

****

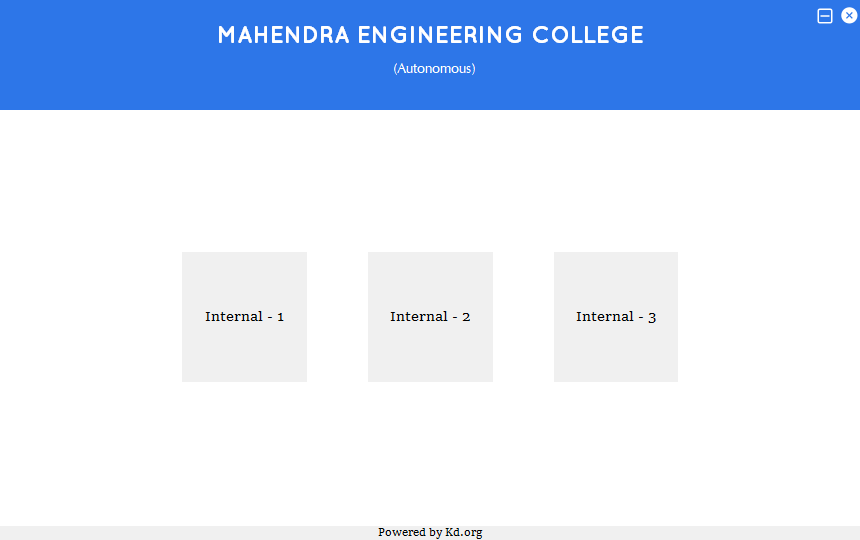
**Fig 6.1.8: Feedback**

****

**Fig 6.1.9: Staff Login**

****

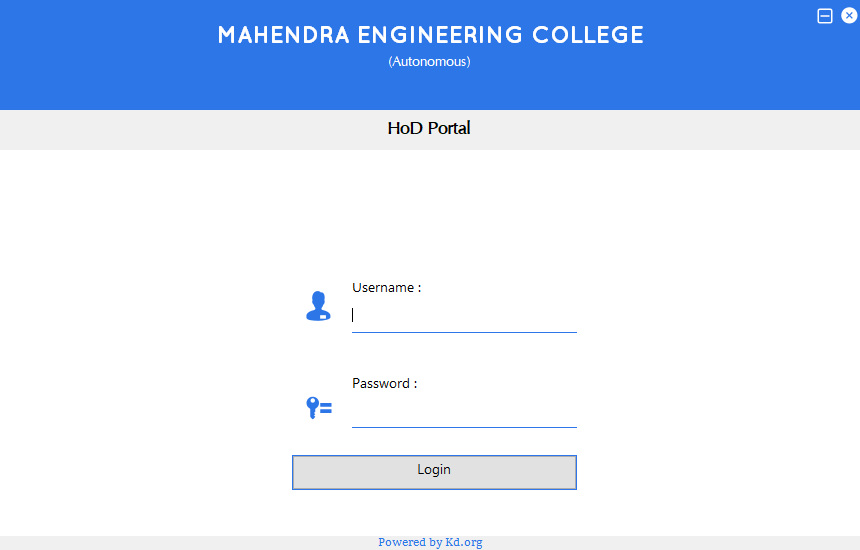
**Fig 6.1.10: Staff Portal**

****

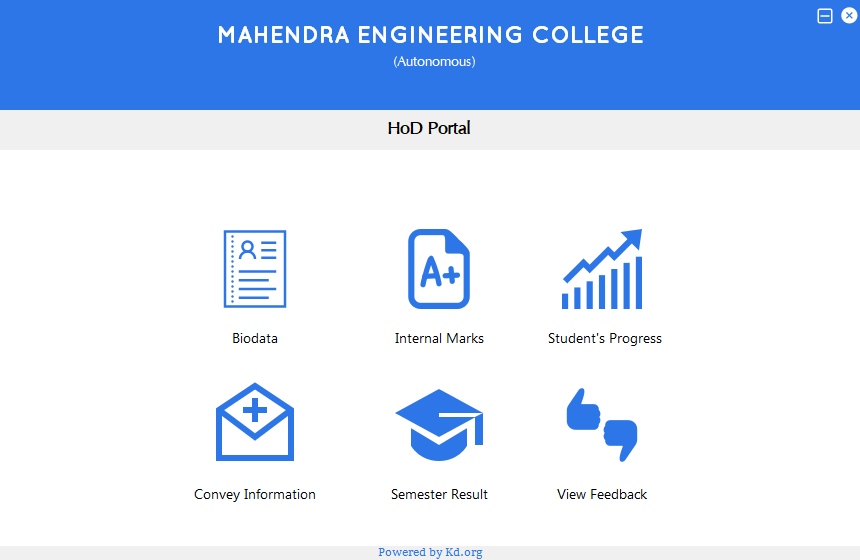
**Fig 6.1.11: Internal Marks**

****

**Fig 6.1.12: Semester Result**

****

**Fig 6.1.13: HoD Login**

****

**Fig 6.1.14: HoD Portal**

**6.2 Sample Code:**

**Title.java:**

package gui;

import com.adobe.acrobat.Viewer;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Toolkit;

import java.awt.event.WindowEvent;

import java.io.File;

import java.io.FileInputStream;

import java.util.logging.Level;

import java.util.logging.Logger;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

public class Title extends javax.swing.JFrame {

public Title() throws InterruptedException {

initComponents();

}

//1. Code for Panel Movement Starts

int g\_xx, g\_yy;

private void MainPanelMousePressed(java.awt.event.MouseEvent evt) {

//Lightens the frame on dragging

setOpacity((float)0.8);

g\_xx = evt.getX();

g\_yy = evt.getY();

}

private void MainPanelMouseDragged(java.awt.event.MouseEvent evt) {

int m\_x = evt.getXOnScreen();

int m\_y = evt.getYOnScreen();

this.setLocation(m\_x-g\_xx, m\_y-g\_yy);

}

//1. Code for Panel Movement Ends

//2. Changing color on mouse enter

void setColor(JLabel p){

p.setOpaque(true);

p.setBackground(new Color(45,118,232));

}

void resetColor(JLabel p){

p.setBackground(new Color(240,240,240));

p.setOpaque(false);

}

private void MainPanelMouseReleased(java.awt.event.MouseEvent evt) {

setOpacity((float)1.0);

}

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

dispose();

Home h = new Home();

h.setVisible(true);

}

public static void main(String args[]) throws InterruptedException {

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Windows".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Title.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Title.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Title.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Title.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

try {

new Title().setVisible(true);

} catch (InterruptedException ex) {

Logger.getLogger(Title.class.getName()).log(Level.SEVERE, null, ex);

}

}

});

}

// Variables declaration - do not modify

private javax.swing.JPanel Content;

private javax.swing.JPanel Footer\_panel;

private javax.swing.JLabel Footer\_txt;

private javax.swing.JPanel Head;

private javax.swing.JLabel Loader;

private javax.swing.JPanel LoaderPanel;

private javax.swing.JPanel MainPanel;

private javax.swing.JLabel SubTitle;

private javax.swing.JLabel Title;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JPanel jPanel1;

private javax.swing.JButton submitButton;

// End of variables declaration

}

**Home.java:**

package gui;

import java.awt.Color;

import javax.swing.JLabel;

import javax.swing.JPanel;

public class Home extends javax.swing.JFrame {

public Home() {

initComponents();

}

//1. Code for Panel Movement Starts

int g\_xx, g\_yy;

private void MainPanelMousePressed(java.awt.event.MouseEvent evt) {

//Lightens the frame on dragging

setOpacity((float)0.8);

g\_xx = evt.getX();

g\_yy = evt.getY();

}

private void MainPanelMouseDragged(java.awt.event.MouseEvent evt) {

int m\_x = evt.getXOnScreen();

int m\_y = evt.getYOnScreen();

this.setLocation(m\_x-g\_xx, m\_y-g\_yy);

}

//1. Code for Panel Movement Ends

//Icon Functions to Create new Frame

private void StudentIconMouseClicked(java.awt.event.MouseEvent evt) {

Student\_Login s = new Student\_Login();

s.setVisible(true);

}

private void StaffIconMouseClicked(java.awt.event.MouseEvent evt) {

Faculty\_Login f = new Faculty\_Login();

f.setVisible(true);

}

private void HoDIconMouseClicked(java.awt.event.MouseEvent evt) {

Hod\_Login h = new Hod\_Login();

h.setVisible(true);

}

//2. Changing color on mouse enter

void setColor(JLabel p){

p.setOpaque(true);

p.setBackground(new Color(45,118,232));

}

void resetColor(JLabel p){

p.setBackground(new Color(240,240,240));

p.setOpaque(false);

}

private void Student\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Student\_txt);

}

private void Student\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Student\_txt);

}

private void Faculty\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Faculty\_txt);

}

private void Faculty\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Faculty\_txt);

}

private void HoD\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(HoD\_txt);

}

private void HoD\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(HoD\_txt);

}

private void StudentIconMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Student\_txt);

}

private void StudentIconMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Student\_txt);

}

private void StaffIconMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Faculty\_txt);

}

private void StaffIconMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Faculty\_txt);

}

private void HoDIconMouseEntered(java.awt.event.MouseEvent evt) {

setColor(HoD\_txt);

}

private void HoDIconMouseExited(java.awt.event.MouseEvent evt) {

resetColor(HoD\_txt);

}

private void Student\_txtMouseClicked(java.awt.event.MouseEvent evt) {

new Student\_Login().show();

}

//2. Changing color ends

private void Faculty\_txtMouseClicked(java.awt.event.MouseEvent evt) {

new Faculty\_Login().show();

}

private void HoD\_txtMouseClicked(java.awt.event.MouseEvent evt) {

new Hod\_Login().show();

}

private void MainPanelMouseReleased(java.awt.event.MouseEvent evt) {

setOpacity((float)1.0);

}

private void exitMousePressed(java.awt.event.MouseEvent evt) {

dispose();

}

private void minimizeMousePressed(java.awt.event.MouseEvent evt) {

this.setState(Home.ICONIFIED);

}

public static void main(String args[]) {

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Windows".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Home.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Home().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JPanel Content;

private javax.swing.JPanel Faculty\_panel;

private javax.swing.JLabel Faculty\_txt;

private javax.swing.JPanel Footer\_panel;

private javax.swing.JLabel Footer\_txt;

private javax.swing.JPanel Head;

private javax.swing.JLabel HoDIcon;

private javax.swing.JLabel HoD\_txt;

private javax.swing.JPanel Hod\_panel;

private javax.swing.JLabel LoginAs\_txt;

private javax.swing.JPanel MainPanel;

private javax.swing.JLabel StaffIcon;

private javax.swing.JLabel StudentIcon;

private javax.swing.JPanel Student\_panel;

private javax.swing.JLabel Student\_txt;

private javax.swing.JLabel exit;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel jLabel2;

private javax.swing.JLabel minimize;

// End of variables declaration

}

**Syllabus.java:**

package gui;

import java.awt.Color;

import java.awt.GraphicsEnvironment;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

public class Syllabus extends javax.swing.JFrame {

public Syllabus() {

initComponents();

}

//1. Code for Panel Movement Starts

int g\_xx, g\_yy;

private void MainPanelMousePressed(java.awt.event.MouseEvent evt) {

//Lightens the frame on dragging

setOpacity((float)0.8);

g\_xx = evt.getX();

g\_yy = evt.getY();

}

private void MainPanelMouseDragged(java.awt.event.MouseEvent evt) {

int m\_x = evt.getXOnScreen();

int m\_y = evt.getYOnScreen();

this.setLocation(m\_x-g\_xx, m\_y-g\_yy);

}

//1. Code for Panel Movement Ends

//2. Changing color on mouse enter

void setColor(JLabel p){

p.setOpaque(true);

p.setBackground(new Color(45,118,232));

}

void resetColor(JLabel p){

p.setBackground(new Color(240,240,240));

p.setOpaque(false);

}

private void MainPanelMouseReleased(java.awt.event.MouseEvent evt) {

setOpacity((float)1.0);

}

private void Sem1\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem1\_txt);

}

private void Sem1\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem1\_txt);

}

private void Sem2\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem2\_txt);

}

private void Sem2\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem2\_txt);

}

private void Sem3\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem3\_txt);

}

private void Sem3\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem3\_txt);

}

private void Sem4\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem4\_txt);

}

private void Sem4\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem4\_txt);

}

private void Sem5\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem5\_txt);

}

private void Sem5\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem5\_txt);

}

private void Sem6\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem6\_txt);

}

private void Sem6\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem6\_txt);

}

private void Sem7\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem7\_txt);

}

private void Sem7\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem7\_txt);

}

private void Sem8\_txtMouseEntered(java.awt.event.MouseEvent evt) {

setColor(Sem8\_txt);

}

private void Sem8\_txtMouseExited(java.awt.event.MouseEvent evt) {

resetColor(Sem8\_txt);

}

private void Sem6\_txtMousePressed(java.awt.event.MouseEvent evt) {

new Sem\_6\_Syllabus().show();

}

private void exitMousePressed(java.awt.event.MouseEvent evt) {

dispose();

}

private void minimizeMousePressed(java.awt.event.MouseEvent evt) {

this.setState(Syllabus.ICONIFIED);

}

//2. Change color ends

//3. Maximize button

static boolean g\_maximized = true;

public static void main(String args[]) {

try {

for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Windows".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break;

}

}

} catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(Syllabus.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(Syllabus.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(Syllabus.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(Syllabus.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

}

//</editor-fold>

/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() {

new Syllabus().setVisible(true);

}

});

}

// Variables declaration - do not modify

private javax.swing.JLabel Autonomous;

private javax.swing.JPanel Content;

private javax.swing.JPanel Footer\_panel;

private javax.swing.JLabel Footer\_txt;

private javax.swing.JPanel Head;

private javax.swing.JLabel MEC;

private javax.swing.JPanel MainPanel;

private javax.swing.JPanel Sem1\_panel;

private javax.swing.JLabel Sem1\_txt;

private javax.swing.JPanel Sem2\_panel;

private javax.swing.JLabel Sem2\_txt;

private javax.swing.JPanel Sem3\_panel;

private javax.swing.JLabel Sem3\_txt;

private javax.swing.JPanel Sem4\_panel;

private javax.swing.JLabel Sem4\_txt;

private javax.swing.JPanel Sem5\_panel;

private javax.swing.JLabel Sem5\_txt;

private javax.swing.JPanel Sem6\_panel;

private javax.swing.JLabel Sem6\_txt;

private javax.swing.JPanel Sem7\_panel;

private javax.swing.JLabel Sem7\_txt;

private javax.swing.JPanel Sem8\_panel;

private javax.swing.JLabel Sem8\_txt;

private javax.swing.JSeparator Seperator;

private javax.swing.JPanel SubHeading;

private javax.swing.JLabel exit;

private javax.swing.JLabel jLabel1;

private javax.swing.JLabel minimize;

// End of variables declaration

}

**CHAPTER 7**

**CONCLUSION**

At the end of the day, we’ve created a Desktop Application using Java and MapReduce. This application will make much of the documentation works very easier for staffs and Head Of the Departments. It can be deployed on any kind of operating systems such as MacOS, Windows OS and Linux OS. As this application has a lighter and elegant UI, users can have a better working experience. It is developed such that it can be scaled horizontally by adding Commodity hardware to the system when the institution goes big and data becomes large. Since it is made completely of free and open source software tools, the cost of development and maintenance becomes considerably cheaper than other technologies. The services can also be added to this application at any time. It’ll be act like a workmate along with you to make your works simpler and faster.

**CHAPTER 8**

**REFERENCES**

|  |  |
| --- | --- |
| [1] | Aditya B. Patel, Manashvi Birla, Ushma Nair, ‘Addressing Big Data Problem Using Hadoop and Map Reduce’, 6-8 Dec. 2012. |
| [2] | Albert Bifet, **‘**Mining Big Data In Real Time**’,** Informatica 37 (2013) 15–20 DEC 2012. |
| [3] | Bernice Purcell, **‘**The emergence of “big data” technology and analytics**’,** Journal of Technology Research 2013. 1994 2/13/04 |
| [4] | Dong, X.L. Srivastava, D. Data Engineering (ICDE), ‘ Big data integration’, IEEE International Conference on 29(2013) 1245–1248 |
| [5] | D. Rajasekar, C. Dhanamani, S.K. Sandhya, ‘A Survey on Big Data Concepts and Tools’, Volume 5 Issue 2 (February.2015) IJETAE-2250-2459. |
| [6] | Francesco Marchioni (2017), ‘MongoDB for Java Developers’, Packt Publications. |
| [7] | Jeanne Boyarsky and Scott Selikoff (2015), ‘Oracle Certified Associate Java SE 8 Programmer I Study Guide’, Wiley Publications. |
| [8] | Kyong-Ha Lee Hyunsik Choi, ‘Parallel Data Processing with Map Reduce: A Survey’,SIGMOD Record December 2011 (Vol. 40, No. 4) |
| [9] | Mala Gupta (2017), ‘OCA Java SE 8 Programmer I Certification Guide’, Manning Shelter Island Publications. |
| [10] | Mukherjee, A. Datta, J. Jorapur, R. Singhvi, R. Haloi, S. Akram, ‘Shared disk big data analytics with Apache Hadoop’, 2012, 18-22 |
| [11] | Mrigank Mridul, Akashdeep Khajuria, Snehasish Dutta, Kumar N, ‘ Analysis of Bidgata using Apache Hadoop and Map Reduce’,Volume 4, Issue 5, May 2014 |
| [12] | S. Vikram Phaneendra, E. Madhusudhan Reddy, ‘Big Data- solutions for RDBMS problems-A Survey’,In 12th IEEE/IFIP Network Operations & Management Symposium (NOMS 2010) (Osaka, Japan, Apr 19{23 2013). |
| [13] | <http://hadoop.apache.org/> |
| [14] | <https://docs.mongodb.com/> |
| [15] | <https://www.javatpoint.com/java-tutorial> |
| [16] | <https://www.smartdraw.com/data-flow-diagram/> |
| [17] | <https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html> |
| [18] | <https://hadoop.apache.org/docs/current/hadoop-mapreduce-client/hadoop-mapreduce-client-core/MapReduceTutorial.html> |
| [19] | <https://docs.oracle.com/en/bigdata/> |