AUTOMATION OF STUDENT EDUCATION VERIFICATION AND SERVICES

A PROJECT REPORT

Submitted by

H. POOJA [REGISTER NO: 211417104184]

PRIYADARSHINI VENKATESAN [REGISTERNO:211417104201]

S.SHERLINE CALISTA [REGISTERNO:211417104256]

in the partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE, CHENNAI-600123.

ANNA UNIVERSITY: CHENNAI 600 025

AUGUST 2021

BONAFIDE CERTIFICATE

Certified that this project report **"AUTOMATION OF STUDENT EDUCATION VERIFICATION AND SERVICE"** is the Bonafide work of "**H.POOJA** (211417104184), **PRIYADARSHINI VENKATESAN (**211417104201**), S.SHERLINE CALISTA (**211417104256)**"** who carried out the project work under my supervision.

SIGNATURE
Dr.S.MURUGAVALLI,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
NASARATHPETTAI
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE
DR.L.JABA SHEELA
SUPERVISOR
PROFESSOR

DEPARTMENT OF CSE,
PANIMALARENGINEERING COLLEGE,
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidates were examined in the Anna University Project Viva-Voce Examination held on.....

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We would like to express our sincere thanks to our Directors **Tmt. C. VIJAYARAJESWARI** , **Dr.C.SAKTHIKUMAR , M.E. , Ph.D** and **Tmt. SARANYASREE SAKTHIKUMAR B.E.,M.B.A.,** for providing us with the necessary facilities for completion of this project.

We also express our gratitude to our Principal **Dr.K.Mani, M.E., Ph.D.** for his timely concern and encouragement provided to us throughout the course.

We thank the HOD of Computer Science and Engineering Department, **Dr. S. MURUGAVALLI, M.E. ,Ph.D.,** for the support extended throughout the project.

We would like to thank our Project Guide **Dr.L. JABA SHEELA** and all the faculty members of the Department of CSE for their advice and suggestions for the successful completion of the project.

We would like to thank our **Parents** and all the family members for their advice and suggestions for the successful completion of the project.

H. POOJA

PRIYADARSHI VENKATESAN

S.SHERLINE CALISTA

ABSTRACT

A Bonafide is a kind of proof that one is studying or working in a particular institution or firm for a specific duration. The certificate is demanded while applying for a passport, educational loans, universities, or when seeking a new job. In the same manner, a Letter of Recommendation is given to the admissions boards of the colleges a person is applying to. Rendering a Bonafide certificate, Letter of Recommendation and verifying a student's background is an obligatory activity of every college. However, well established institutions at times may find it tedious as well as assiduous work to issue them when the procedure is extensive or when there are profuse number of applicant to be verified within a shorter span of time. The proposed system is completely automated which decreases the time consumption and manual tasks performed during background verification and while issuing a Bonafide or an LOR. Using this web application, the student authentication process, request of Bonafide from the HOD, request of letter from their respected staffs and approval of the letter or the Bonafide with proper authorization of the Management can be done on a single platform from any place making it quickly accessible. Along with that we have a developed software for Background Verification that parses the mail contents on being sent in a table format by a background verifier and matches it with the student database to find if the information is valid or not. If the information matches accurately with the data, an email is of the data verified mail is sent back to the sender and not being verified a data not verified mail is been sent .

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVATIONS

S. No	ABBREAVATIONS	EXPANSION
1	JDK	Java Development Kit
2.	DEX	Dalvik Executables
3.	TCP	Transmission Control Protocol
4.	IP	Internet Protocol
5.	HTTP	Hyper Text Transfer Protocol
6.	ADT	Android Development Tool

CHAPTER 1

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The main objective of Automated Student Verification Software and Services is to ease the process of requesting for a Bonafide certificate and Letter of Recommendation (LOR) from their faculties. As an added advantage, a software is also proposed dedicatedly for automating the process of verifying the student's data directly with the database maintained by the college and simultaneously sending automated reply emails to sender stating if the data is verified or not. If an exact match is found, then the reply mail states that data is verified. Else, the reply mail states that data is not verified.

1.2 PROBLEM DEFINITION

In order to get the Bonafide certificate, one has to appeal to the head of the institute/organization in writing. The entire procedure follows an extensive and a very complicated process. The student has to manually draft a letter, get it signed from the dignitaries and submit it to the college's admin office to receive a Bonafide. The procedure for requesting for an LOR follows pretty much a similar process to that of the Bonafide where as there is drafting of a letter by user requesting the particular staff for approval, getting an authorized sign of the staff and seal of officials and that could be a long walk. The traditional ways of verifying the basic details of a student is through the academic records which they have obtained throughout the course of study. Every detail are verified by the Admissions and Records office of the school or institution manually that includes cross verifications of the students with their Roll no, Name and joining year of the college with their earlier registers, marksheets etc. These conventional methods are time consuming and require more of man-power and effort to complete the work.

CHAPTER 2

CHAPTER 2
SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The current scenario involves the students to manually draft a letter while requesting for a Bonafide by stating the valid reason. Once the HOD approves the letter it is further forwarded to the admin office from where the Bonafide can be collected. The LOR also follows a similar process.

Typically, background check will verify the basic details of a student and the academic records obtained throughout the course of study. Every detail are verified by the Admissions and Records office of the school or institution manually.

2.2 PROPOSED SYSTEM

The proposed system contains three modules with the first one being a website for the purpose of issuing Bonafide and LOR and the other aspect is a software dedicatedly meant for Background Verification. Website is developed for requesting and issuing a BONAFIDE and LOR where students can easily login/sign up giving in the right credentials and request for Bonafide or an LOR by selecting the specific reason for applying. Furthermore, the request is made available in the HOD's login. The HOD has the privilege for approving/declining the student's request for Bonafide. The request for LOR is made available in the respected staff's login to which the request is raised to. Upon the approval of the faculty, the respected document is made available in the student's login. software is implemented in such a way that whenever background verification companies send email to institutions requesting for background verification of passed out students, the software reads the unread mails, parse out the data from them with the help of keywords obtained and cross-checks them with the existing student database to find if the information provided is genuine or not. The reply email follows a tabular format .

Advantages

- This system will make the process of requesting and retrieving certificates automated.
- This process will reduce the effort, time and man power.
- By making use of this portal, one can collect their certificates on or before the time they need it right from their place of stay.

2.3 HARDWARE AND SOFTWARE REQUIREMENTS

HARDWARE REQUIREMENTS

- Hard Disk : 80GB and Above
- RAM : 4GB and Above
- Processor : P IV and Above

SOFTWARE REQUIREMENTS

- Windows 7 and above(64 bit)
- JDK 1.8
- Tomcat 8.5
- MySQL 5.0
- Java
- Apache Tomcat Server
- Web designing: HTML 5,CSS3
- Animation Effect : jQuery
- Front-end Framework: Bootstrap 3
- J2EE:JSP,Servlet
- Core Framework: Collections
- Script: Java Script

CHAPTER 3

CHAPTER 3

SYSTEM DESIGN

3.1 ER DIAGRAM

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). In the Fig 3.1 shown below the attributes of Student Register, Staff Register are similar where S.ID to be the primary key,whereas Name, Date of Birth, Email, Password, Catgeory, Address to be some of the common attributes to both the entities, the relationship set defined here is website access which has one to one Relation with the administrator page who can view the registered students details and authenticate their access to the website. Further, we have the Login entity with email and password as attributes for viewing Bonafide/Lor for the students.

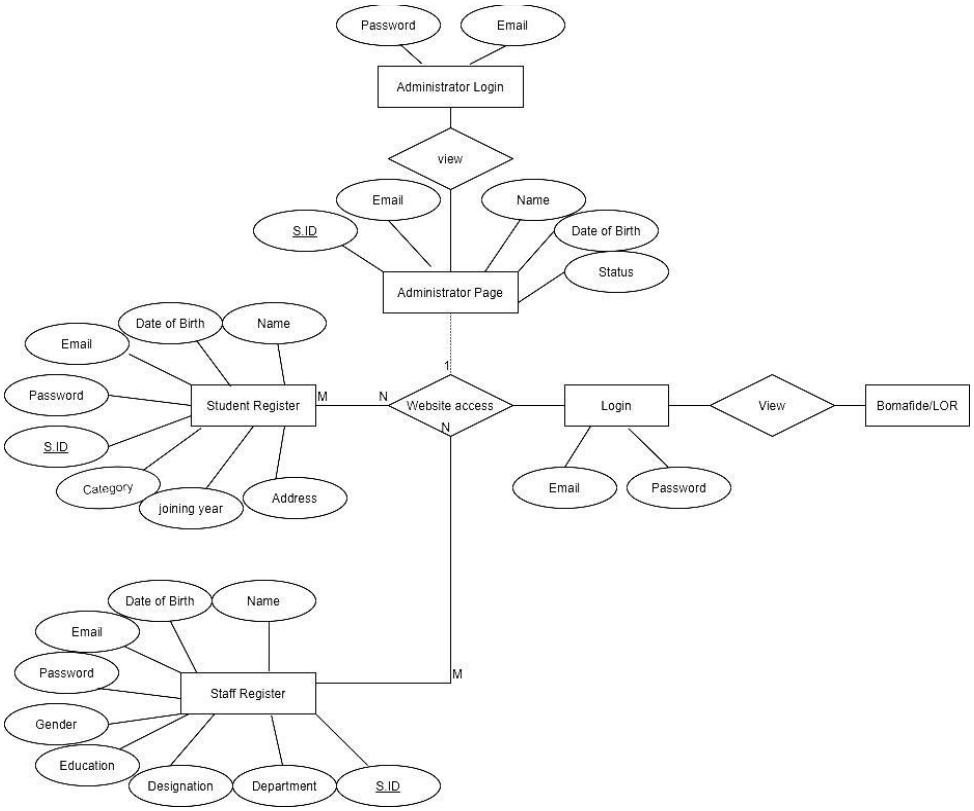


Fig:3.1 ER diagram for Bonafide/lor

3.2 DATA DICTONARY

Data dictionary is a centralized repository of metadata. Metadata is data about data Such as What table(s) each field exists in, What database(s) each field exists in and the data types, e.g., integer, real, character, and image of all fields in the organization's databases. The data dictionary described below is of a Student Background verification table which consists of field name like SNo, Candidate's Name, College, Registration Number, CGPA, Year of Passing, Qualifications and Mode of Qualifications each of these having a specific DataType and allocating distinct Field size of Displays for the Field Name. Whereas, for the candidate's name, College, Qualifications and Mode of Qualifications we have allocated DataType as Text with Field size of 20. In case of CGPA the data type allocated is Numeric (p,s) with field size of (38,0) where the precision is 38 and scale is 0 and Roll Number and Registration Number the data type of Integer of field size 10 is been allocated.

Table:3.2 Data Dictionary

Field Name	Data Type	Field Size for Display	Example
Student ID	Integer	10	2017pecce456
Registration Number	Integer	30	kavya
Email	String	-	ramya@gmail.com
Password	String	15	Kavya6
Course	String	25	Mech
Category	Character	2	
Gender	string	10	female
Address	String	40	No: 6 north street, Appts laurel
Pin code	Integer	6	600090

3.3 UML DAIGRAMS

3.3.1 USECASE DIAGRAM

Unified Modeling Language (UML) is a standardized general-purpose modeling language in the field of software engineering. The UML is managed and was created by the Object Management Group. UML use case diagram consists of two parts a Use case to describes a sequence of actions and an actor is a person, organization that plays a role in interaction with the system. In the diagram represented below we have 3 actors Admin, Student and Staff there sequence of actions is performed for the Bonafide and LOR. A student can perform the action such as Registering, Login, Request for Bonafide and LOR. Whereas, a staff performs explicit actions of Login and accepting of Bonafide/LOR. The basis for the Admin actor is to allow access of authorized students into the website.

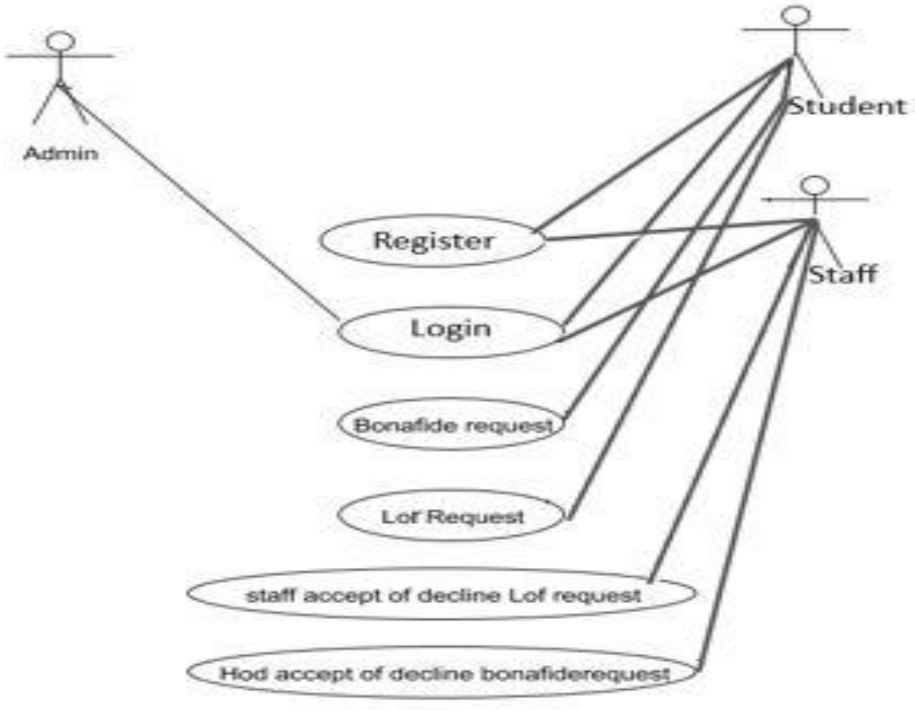


Fig:3.3 Use case diagram for Bonafide/lor

3.3.2 ACTIVITY DIAGRAM

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control. Below is the representation of activity diagram of the Automation of student Education Verification and Services. Overhead of the first activity represented is Bonafide/LOR. Further on making a decision of either Bonafide or LOR activity they have the accept or decline action to perform. Once the activities are performed the workflow comes to an end. Whereas, in the student verification system opening activity is checking of the data with the database, Later, if the data is found activity of data verified message is passed to user and if the data is not found a data not verified message is sent to the user. This is the end of workflow in a student verification system.

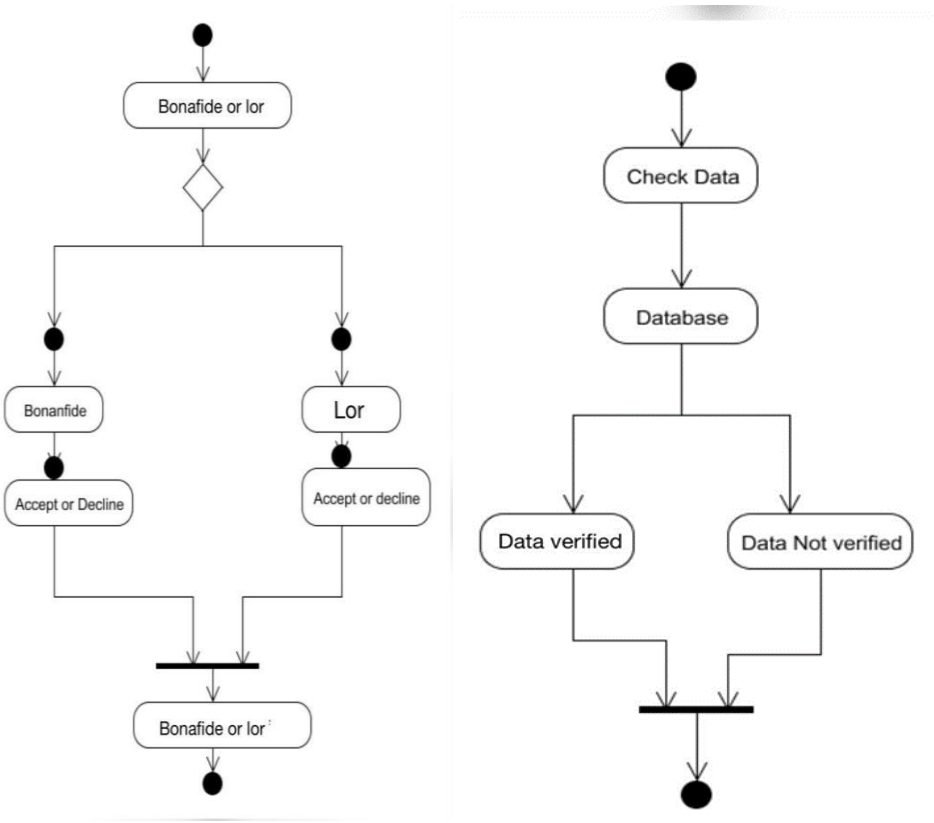


Fig:3.4 Activity diagram for Bonafide/lor and student verification system

3.3.3 SEQUENCE DIAGRAM

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams and are sometimes called event diagrams, event sceneries and timing diagrams. The Sequence Diagram represented below represents the sequence of an Bonafide/LOR as follows. Initially on Admin accepting approval or decline the student is further allowed to Login and raise a request for Bonafide/LOR this message is processed forward to the other two objects i.e, Hod or the staff. Their message of accept or decline is reverted back to the students. Whereas for Background Verification software there are two objects presented Software and sender mail, the sender mail sends the message of check data, the software on data being verified or not sends back an response.

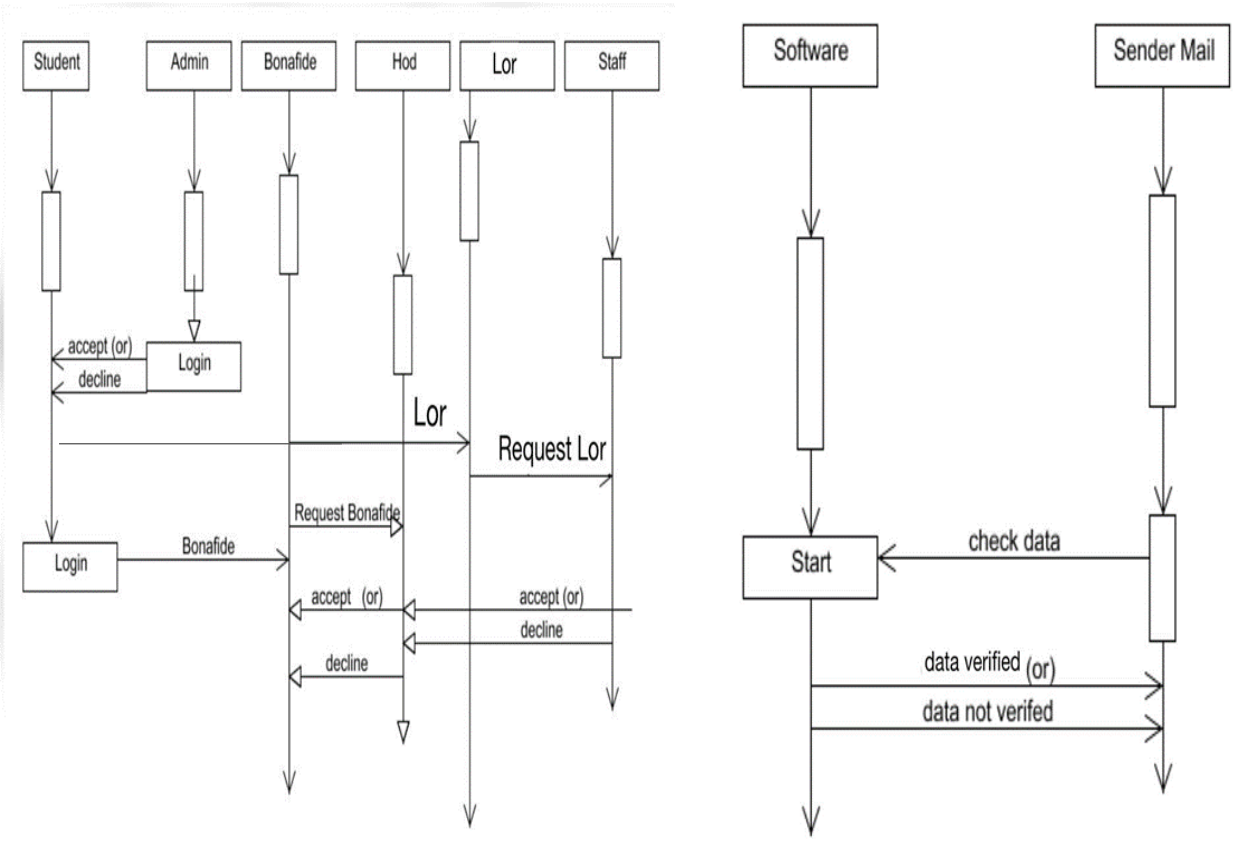


Fig: 3.5 Sequence diagram for Bonafide/lor and student verification system

3.3.4 COLLABORATION DIAGRAM

UML Collaboration Diagrams illustrate the relationship and interaction between software objects. They require use cases, system operation contracts and domain model to already exist. The collaboration diagram illustrates messages being sent between classes and objects. Student Logins into account. On performing the required action of Bonafide or an LOR. For the Request of Bonafide action message is notified to HOD for the approval or decline. Further, this message is notified to the student. Similarly for an LOR where the staff approves the invitation or decline it. In the Collaboration diagram of Student Verification system there is an mail object which parses the data verifies with the database. Message is sent to the next object of Data verification or Data not verification. Ultimately the message reaches the mail object.

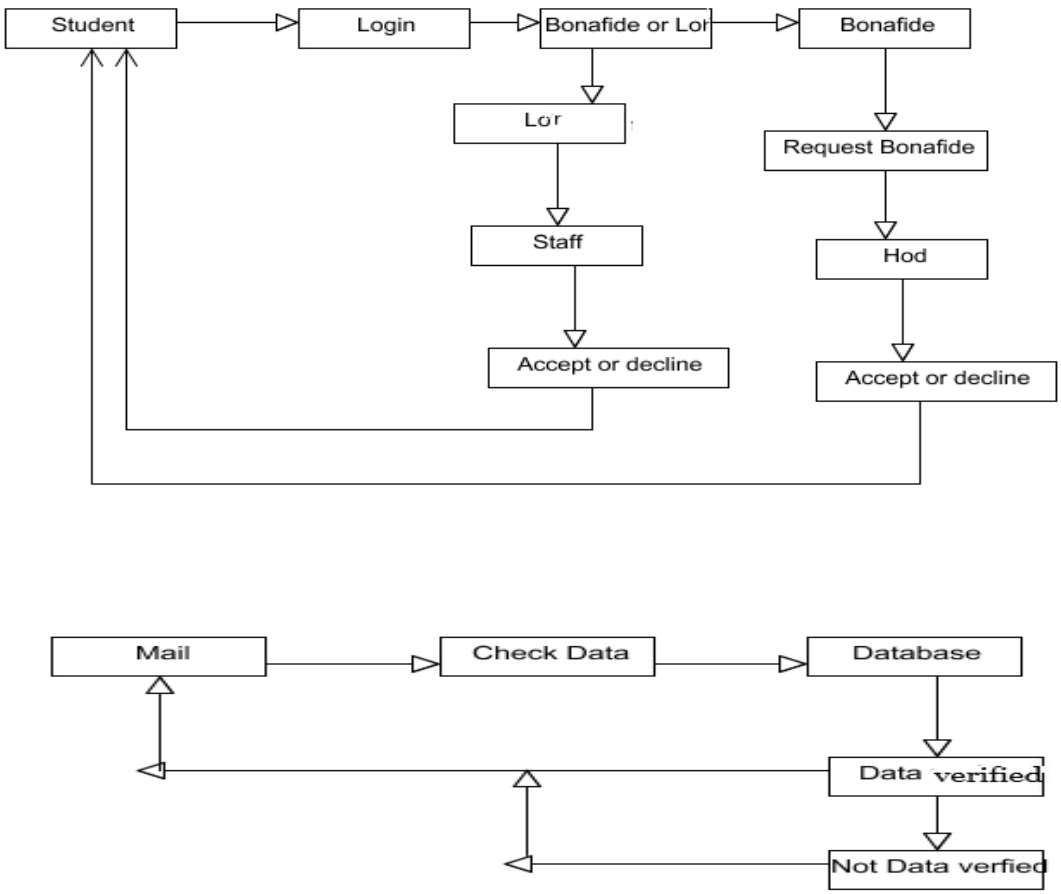


Fig:3.6 Collaboration diagram for Bonafide/lor and student verification system

3.3.5 CLASS DIAGRAM

A Class diagram in the Unified Modeling Language is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. In the class diagram there is the Bon Certificate class while the attributes under it are name, register number, file name, course, joining year, end year and the methods performed under this class are void do Post () and void do Get () similar attributes and methods are been performed under the Classes of Bon Request, Bon Store, LOR Store, LOR Request and under Checking LOR.

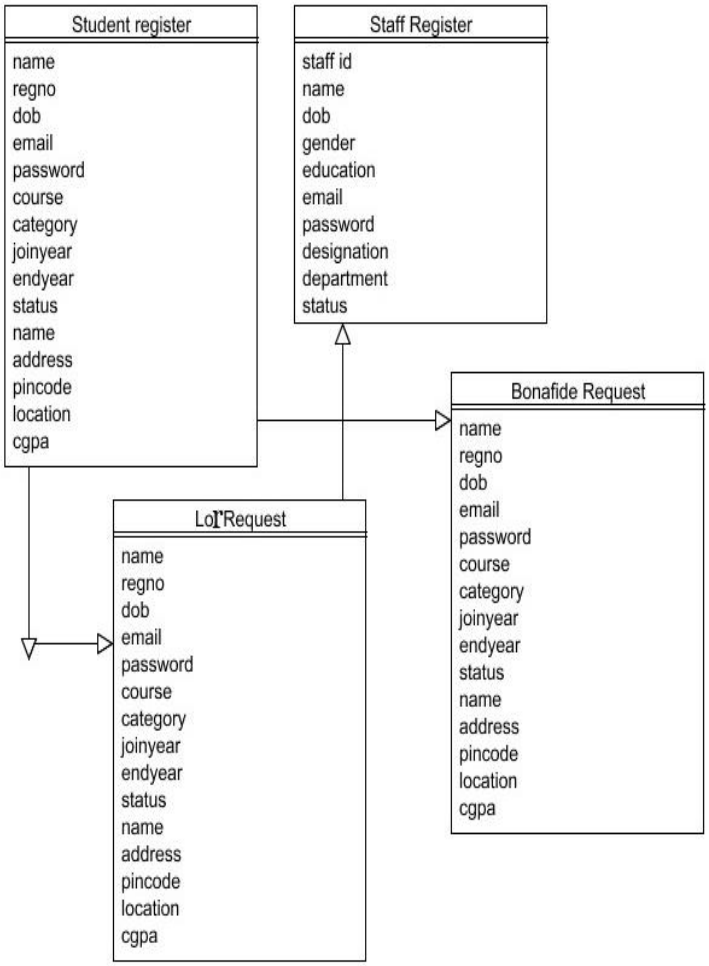


Fig:3.7 Class diagram for Bonafide/lor and student verification system

CHAPTER 4

CHAPTER 4

SYSTEM ARCHITECTURE

4.1 ARCHITECTURE OVERVIEW

The overall system architecture for the Automation of Student Education and Verification Services is present in Fig:4.1.Student can register themselves in the website for applying for a Bonafide/LOR from the college faculty. After successful registration, the students can Login and raise a request for obtaining a Bonafide/LOR from the user's account.The request on being accepted by the Hod/Staffs, an immediate status updatement is shown under the user account. The documents are then made available in the student's login. They can either view/ download or print it as per the user's needs.Additionally, the Background verification system logs into the Gmail account, parses the email contents with the keywords and cross-checks with the database maintained by the college. If a student data matches with the data from the database, then the adjacent row next to the keyword states that the data is "Verified". Else, the field states that that the data is "Not Verified".

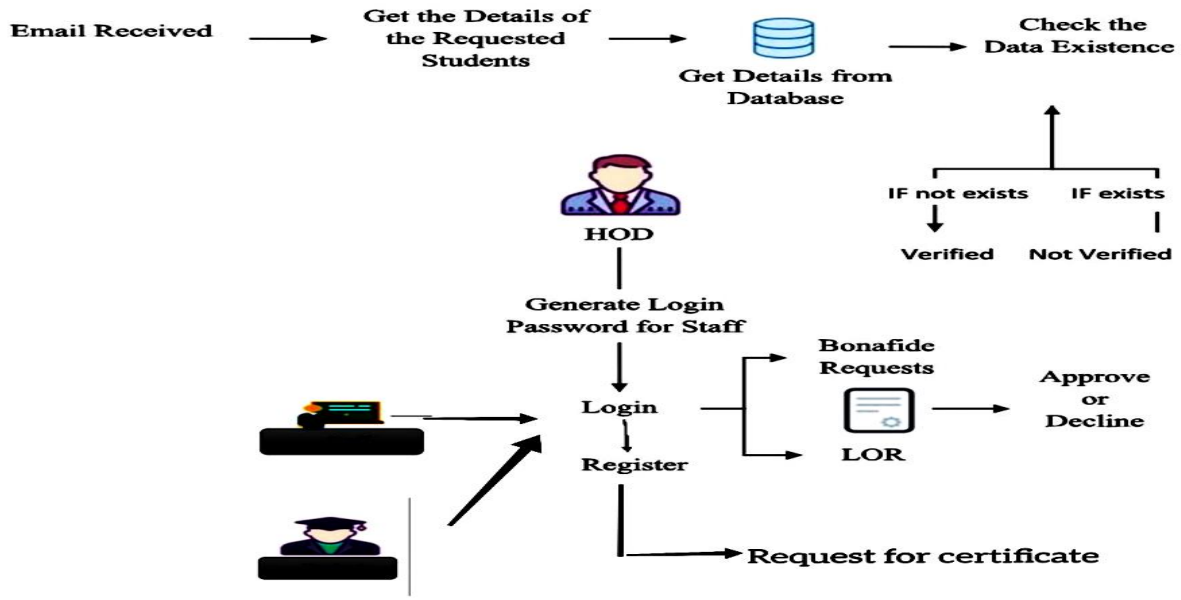


Fig:4.1 Architecture Overview of automated student verification system and services

4.2 REQUEST FOR BONAFIDE

In this module, student will register and the login with their login credentials. The students can request for the Bonafide certificate by selecting a reason from the pre-defined list which is present in the website. The HOD can approve or decline the certificate which are requested by the students. If the HOD approves the request, the request has been approved. Following which, the student's name will be appended in the Bonafide template that is maintained already along with the seal and signature of the authority. The student can check the status of their Bonafide by logging into their account with the correct credentials. If the HOD approves the request for Bonafide, a download button will be provided in the student login page from which they can download their Bonafide.

The functionalities that are present under the website are:

STUDENT REGISTRATION

- Initially the student has to register his/her details into the website.
- The Information gathered would include (Student Name, DOB ,Email , Password, Course, Category, Joining Year, Gender, Address, Location and pin code)
- Finally, in order to save up the details click on "Register" button.

STAFF REGISTRATION

- At first the staff has to register their details into the site.
- Information gathered would include (Staff Name, DOB , Gender, Education, Email, Password, Designation, Department)
- Within the Staff Registration includes the Hod registration into the website .
- Finally, In order to save up the details click on "Register" button.
- The details are saved securely in the Backend of the website.

ADMIN LOGIN

- They are the ones who verify the students and staff who has registered on the website and consent their access to it.
- Once the verification process is being approved by the Administrator students can login into website by providing the registered mail Id and password and then apply for a Bonafide or an LOR.

STUDENT LOGIN

On clicking BONAFIDE

- A Student is provided to click there reasons which are provided from the drop down list.
- Pop up message of Immediate request is sent will be displayed .

HOD LOGIN

- Logs into the website by providing email Id and password .Further, she can check on pending status of approval and decline.
- The Built in Bonafide appears required changes can be done and can be provided to the student.

4.3 REQUEST FOR LETTER OF RECOMMENDATION (LOR)

In this module, the student will login with their login credentials. The students can request for the Letter of Recommendation (LOR) certificate to any selected staff .Upload a letter of your LOR and choose the faculty's name from the pre-defined list. Once the “Request for LOR” Button is clicked, the corresponding faculty is notified in their login. On approving there request, the student will be notified / a pop up message will appear in the login stating that the LOR request has been accepted. Following which, the contents of the letter will be appended in the LOR template maintained in the website along with the seal and signature of the authority. The student can check the status of their LOR by logging into their account with the correct credentials. A download button will be provided in the student login page by which they can download their LOR.

The functionalities that are present under the website are:

STUDENT REGISTRATION

- Initially the student has to register his/her details into the website.
- The Information gathered would include (Student Name, DOB ,Email , Password, Course, Category, Joining Year, Gender, Address, Location and pin code)
- Finally, in order to save up the details click on “Register” button.

STAFF REGISTRATION

- At first the staff has to register their details into the site
- Information gathered would include(Staff Name, DOB ,Gender, Education, Email, Password, Designation, Department)

ADMIN LOGIN

- They are the ones who verify the students and staff who has registered on the website and consent their access to it.
- Once the verification process is being approved by the Administrator students can login into website by providing the registered mail Id and password and then apply for a Bonafide or an LOR.

STUDENT LOGIN

On Clicking LOR

- A list of staff’s name along with their details will be present (students can select there Staff proceed by clicking on request button)
- This leads us to another page Letter Of Recommendation (provided space you can type in your letter) and provide your signature as a evidence.
- Pop up message of Immediate Request message is sent will be displayed.

STAFF LOGIN

- Logs into the website by providing email Id and password. Further, she can check on pending status of approval and decline.
- On approval can look into the LOR of the student .Later they can decline or approve it, on approving.
- Staff need's to upload their signature and student can print it from there login page.

4.4 BACKGROUND VERIFICATION

Initially, background verification companies/third-party background verifiers send an email to college requesting for background verification for passed out students. The proposed software is implemented in such a way that whenever background verification companies send email to institutions requesting for background verification of passed out students, the software reads the unread mails, parse out the data from them with the help of keywords such as Candidate Name, Register number, College Name, Qualifications, CGPA obtained and cross-checks them with the existing student database to find if the information provided is genuine or not. The reply email follows a tabular format. It consists of all the fields that are checked with the database with a corresponding field for each row stating if the data is verified or not. If a student data matches with the data from the database, then the adjacent row next to the keyword states that the data is "Verified". Else, the field states that that the data is "Not Verified". The tables for reply email are created using HTML and CSS. If the user of the software wants to change the email from which the data is being parsed, they can login as an admin and input the credentials of the new gmail account.

CHAPTER 5

CHAPTER 5

SYSTEM IMPLMENTATION

5.1 Client-side coding

Bonafide/LOR Client-side coding

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Home</title>
    <meta content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=no"  name="viewport">
   <link rel="stylesheet" href="scss/main.css">
   <link rel="stylesheet" href="scss/skin.css">
  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
   <script src="./script/index.js"></script>
</head>
<body id="wrapper"
<header>
        <nav class="navbar navbar-inverse">
          <div class="container">
           <div class="row">
            <div class="navbar-header">
                                    <button type="button" class="navbar-toggle collapsed" data-
                toggle="collapse"          data-target="#navbar" aria-expanded="false" aria-
                                                                                controls="navbar">
                    <span class="sr-only">Toggle navigation</span>
                      </button>
                     <a class="navbar-brand" href="#">
```

```
                <h1></h1><span></span></a>
 <div id="navbar" class="collapse navbar-collapse navbar-right">
                <ul class="nav navbar-nav">
                <li class="active"><a href="#">Home</a></li>
                        <li><a href="register1.jsp">Register</a></li>
                         <!-- <li><a href="addstaff.jsp">Staff Register</a></li>
                         <li><a href="admin.jsp">Admin</a></li>
                    <li><a href="hod.jsp">HOD</a></li>
                    <!-- <li><a href="student.jsp">Student Login</a></li>
<div id="myCarousel" class="carousel slide">
    <!-- Indicators -->
    <ol class="carousel-indicators">
      <li data-target="#myCarousel" data-slide-to="0" class="active"></li>
      <li data-target="#myCarousel" data-slide-to="1"></li>
      <li data-target="#myCarousel" data-slide-to="2"></li>
    </ol>
     <!-- Wrapper for slides -->
    <div class="carousel-inner">
      <div class="item active">
        <div class="fill" style="background-image:url('img/banner-slide-
1.jpg');"></div>
        <div class="carousel-caption slide-up">
        ` <h1 class="banner_heading">Automated System For Bonafide Certificate and
                                                    LOR Request<span> </span>
          <!-- <div class="slider_btn">
          <button type="button" class="btn btn-default slide">Learn More <i class="fa
                                                    fa-caret-right"></i></button>
                    <button type="button" class="btn btn-primary slide">Learn More <i
                                                    class="fa fa-caret-right"></i></button>
 <section id="bottom-footer">
    <div class="container">
      <div class="row">
```

Background verification Client-side

```
<%@page contentType="text/html" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html lang="en">
<head>
<title></title>
 <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=Edge">
    <meta name="description" content="">
    <meta name="keywords" content="">
    <meta name="team" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1">
    <link rel="stylesheet" href="css/bootstrap.min.css">
    <link rel="stylesheet" href="css/owl.carousel.css">
    <link rel="stylesheet" href="css/owl.theme.default.min.css">
    <link rel="stylesheet" href="css/font-awesome.min.css">
    <!-- MAIN CSS -->
    <link rel="stylesheet" href="css/tooplate-style.css">
</head>
<body>
    <!-- PRE LOADER -->
    <section class="preloader">
        <div class="spinner">
            <span class="spinner-rotate"></span>
          </div>
    </section>
```

```
    <!-- MENU -->
    <section class="navbar custom-navbar navbar-fixed-top" role="navigation">
        <div class="container">
            <div class="navbar-header">
             <button class="navbar-toggle" data-toggle="collapse" data-
target=".navbar-collapse">
                <span class="icon icon-bar"></span>
                <span class="icon icon-bar"></span>
                <span class="icon icon-bar"></span>
            </button>
 <!-- lOGO TEXT HERE -->
            <a href="" class="navbar-brand">Background Verification
Software</a>
        </div>
<!-- MENU LINKS -->
        <div class="collapse navbar-collapse">
            <ul class="nav navbar-nav">
              <li><a href="index2.jsp" class="smoothScroll">Start</a></li>
                <li><a href="index.jsp" class="smoothScroll">Stop</a></li>
            </ul>
            <ul class="nav navbar-nav navbar-right">
           </ul>
        </div>
    </section>
    <!-- FEATURE -->
    <section id="home" data-stellar-background-ratio="0.5">
        <div class="overlay"></div>
        <div class="container">
            <div class="row">
```

```
                <div class="col-md-offset-3 col-md-6 col-sm-12">
                    <div class="home-info">
                        <h3>Verification</h3>
                        <h1>Background Verification </h1>
                        <form action="" method="get" class="online-form">
                       </form>
                     </div>
                     </div>
                    </div>
                    </div>
                    </section>
<div align="center"><iframe src="testpage.jsp" width="800" height="600" frameborder="0"></iframe></div>
    <!-- FOOTER -->
    <footer id="footer" data-stellar-background-ratio="0.5">
        <div class="container">
            <div class="row">
    <div class="copyright-text col-md-12 col-sm-12">
                    <div class="col-md-6 col-sm-6">
                        <p>Copyright &copy; 2021 </p>
                    </div>
    <div class="col-md-6 col-sm-6">
                        <ul class="social-icon">
                     </ul>
                    </div>
                </div>
            </footer>
   <!-- SCRIPTS -->
    <script src="js/jquery.js"></script>
```

```
    <script src="js/bootstrap.min.js"></script>
    <script src="js/jquery.stellar.min.js"></script>
    <script src="js/owl.carousel.min.js"></script>
    <script src="js/smoothscroll.js"></script>
    <script src="js/custom.js"></script>

</body>
</html>
```

5.2 Server-side coding

Bonafide/LOR Server -side

```
package com.servlet;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.SQLIntegrityConstraintViolationException;
import java.sql.Statement;
import java.util.Random;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.dao.ConnectionProvider;

public class BorRequest extends HttpServlet {

	public BorRequest() {
		super();
	}
```

```
	public void destroy() {
		super.destroy(); // Just puts "destroy" string in log
		// Put your code here
	}

	public void doGet(HttpServletRequest request, HttpServletResponse
response)
				throws ServletException, IOException {
System.out.println("Bor request coming");

String stuid = request.getParameter("stuid");
System.out.println("=="+stuid);
String regno = request.getParameter("regno");
String dob = request.getParameter("dob");
String email = request.getParameter("emails");
String password = request.getParameter("passwords");
String course = request.getParameter("course");
String category = request.getParameter("category");
String jyear = request.getParameter("jyear");
String eyear = request.getParameter("eyear");
String gender = request.getParameter("gender");
String reasonbof = request.getParameter("reasonbof");
String name = request.getParameter("name");
String stat = "waiting";


System.out.println(stuid+"--"+regno+"--"+dob+"--"+email+"--"+password+"--
"+course+"--"+category+"--"+jyear+"-"+eyear+"--"+gender+"--"+reasonbof);
```

```
//duplicate entry check

try{
		Connection conssss= ConnectionProvider.getCon();
		Statement statement=conssss.createStatement();
		String sql ="select * from Bonafide_request_copy where regno
="'"+regno+"'";
		ResultSet resultSet = statement.executeQuery(sql);
		if(resultSet.next()){

System.out.println("duplicate entry-------");
request.setAttribute("dup", "Previous Boncertificate Request is Pending");
RequestDispatcher rd = request
.getRequestDispatcher("studenthome.jsp");
rd.forward(request, response);;
		}

}
catch (SQLIntegrityConstraintViolationException e) {
  System.out.println("duplicate entry----"+e);


} catch (SQLException e) {
		// TODO Auto-generated catch block
		request.setAttribute("dup", "No entry");
		 RequestDispatcher rd = request
				.getRequestDispatcher("studenthome.jsp");
		rd.forward(request, response);;
		e.printStackTrace();
```

```
}


try{
        Connection conssss= ConnectionProvider.getCon();
        Statement statement=conssss.createStatement();
        String sql ="select * from Bonafide_request where regno ="'+regno+"'";
        ResultSet resultSet = statement.executeQuery(sql);
        if(resultSet.next()){

                request.setAttribute("dup", "Previous Boncertificate Request is
Pending");
                RequestDispatcher rd = request
                .getRequestDispatcher("studenthome.jsp");
                rd.forward(request, response);;

        }

}
catch (SQLIntegrityConstraintViolationException e) {
   // Duplicate entry
         System.out.println("duplicate entry----"+e);

} catch (SQLException e) {
        // TODO Auto-generated catch block
        request.setAttribute("dup", "No entry");
         RequestDispatcher rd = request
                .getRequestDispatcher("studenthome.jsp");
        rd.forward(request, response);;
```

```
            e.printStackTrace();
}
Random rands = new Random();

//Generate random integers in range 0 to 999
int r = rands.nextInt(1000);
System.out.println("unique id ----"+r);

try{
            int status = 0;
            Connection con= ConnectionProvider.getCon();
            String tb1 = "insert into Bonafide_request values
(?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
              PreparedStatement ps=con.prepareStatement(tb1);
    ps.setString(1,stuid );
    ps.setString(2,regno );
    ps.setString(3, dob);
    ps.setString(4, email);
    ps.setString(5, password);
    ps.setString(6, course);
    ps.setString(7, category);
    ps.setString(8, jyear);
    ps.setString(9, eyear);
    ps.setString(10, gender);
    ps.setString(11, reasonbof);
    ps.setString(12, name);
    ps.setString(13, stat);
    ps.setInt(14, r);
```

```
  status=ps.executeUpdate();
  System.out.println(" Bor 1  request succes");
  String tb2 = "insert into Bonafide_request_copy values
(?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
            PreparedStatement ps1=con.prepareStatement(tb2);
    ps1.setString(1,stuid );
    ps1.setString(2,regno );
    ps1.setString(3, dob);
    ps1.setString(4, email);
    ps1.setString(5, password);
    ps1.setString(6, course);
    ps1.setString(7, category);
    ps1.setString(8, jyear);
    ps1.setString(9, eyear);
    ps1.setString(10, gender);
    ps1.setString(11, reasonbof);
    ps1.setString(12, name);
    ps1.setString(13, stat);
    ps1.setInt(14, r);
    status=ps1.executeUpdate();
    System.out.println("zz");
     System.out.println(" Bor 2  request succes");
     System.out.println("bor 3 request");
    //third table
    String tb22 = "insert into Bonafide_request_copy1 values
(?,?,?,?,?,?,?,?,?,?,?,?,?,?)";
             PreparedStatement ps12=con.prepareStatement(tb22);
  ps12.setString(1,stuid );
  ps12.setString(2,regno );
```

```
 ps12.setString(3, dob);
 ps12.setString(4, email);
 ps12.setString(5, password);
 ps12.setString(6, course);
 ps12.setString(7, category);
 ps12.setString(8, jyear);
 ps12.setString(9, eyear);
 ps12.setString(10, gender);
 ps12.setString(11, reasonbof);
 ps12.setString(12, name);
 ps12.setString(13, stat);
 ps12.setInt(14, r);
 status=ps12.executeUpdate();
 request.setAttribute("success", "Bonafide Certificate Request Submitted !!");
 RequestDispatcher rd = request .getRequestDispatcher("studenthome.jsp");
rd.forward(request, response);;
}
catch(Exception e)
{
        e.printStackTrace();
}
        /**
         * Initialization of the servlet. <br>
         *
         * @throws ServletException if an error occurs
         */
        public void init() throws ServletException {
                // Put your code here
        }
```

Background verification Server-side

```
 <%@page import="javax.mail.internet.InternetAddress"%>
<%@page import="javax.mail.Address"%>
<%@page import="java.net.URLDecoder"%>
<%@page import="javax.mail.MessagingException"%>
<%@page import="javax.mail.Message"%>
<%@page import="java.io.BufferedReader"%>
<%@page import="java.io.InputStreamReader"%>
<%@page import="java.io.InputStream"%>
<%@page import="javax.mail.search.FlagTerm"%>
<%@page import="javax.mail.Flags"%>
<%@page import="javax.mail.Folder"%>
<%@page import="javax.mail.Store"%>
<%@page import="javax.mail.Session"%>
<%@page import="java.util.Properties"%>
<%@ page contentType="text/html; charset=iso-8859-1" language="java" import="java.sql.*" errorPage="" %>
<%@ include file="dbconnect.jsp" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"
<title>Untitled Document</title>
 </head>
<body bgcolor="#FFFFFF">
   <%
   String em="",pw="";
```

```
 try
{
  Statement stmt4=Con.createStatement();
    ResultSet rs4=stmt4.executeQuery("select * from admin");
   rs4.next();
     em=rs4.getString("email");
     pw=rs4.getString("pass");
catch(Exception e)
{
   //out.print(e.getMessage());
}
Folder inbox;
     final String SSL_FACTORY = "javax.net.ssl.SSLSocketFactory";

   //Set email properties
   Properties props = System.getProperties();
  props.setProperty("mail.imap.socketFactory.class", SSL_FACTORY);
   props.setProperty("mail.imap.socketFactory.fallback", "false");
   props.setProperty("mail.imap.port", "993");
   props.setProperty("mail.imap.socketFactory.port", "993");
   props.put("mail.imap.host", "imap.gmail.com");
   try
   {
      //Create the session and get the store to read email
      Session session1 = Session.getDefaultInstance(props, null);
      Store store = session1.getStore("imap");
      store.connect("imap.gmail.com", 993, em, pw);
      inbox = store.getFolder("INBOX");
      inbox.open(Folder.READ_ONLY)
```

```
            Flags seenEmail = new Flags(Flags.Flag.SEEN);
            FlagTerm unseenFlagTerm = new FlagTerm(seenEmail, false);
            Message messages[] = inbox.search(unseenFlagTerm);
            ////out.println("Nº of unread messages: " + messages.length);
            //Use a FetchProfile
            ////FetchProfile fp = new FetchProfile()
            ////fp.add(FetchProfile.Item.ENVELOPE);
            ////fp.add(FetchProfile.Item.CONTENT_INFO);
            ////inbox.fetch(messages, fp);
             String sender="";
             String subj="";
             String mess="";
             String dtt="";
             String sss="";
             Address[] a;
             String from = null, to = null;
             Statement stmt2=Con.createStatement();
             Statement stmt3=Con.createStatement();
             Statement stmt4=Con.createStatement();
            try
            {
                ////System.out.println("Count="+messages.length);
               ////System.out.println(messages[0]);
                ////printAllMessages(messages);
              if (messages.length == 0) System.out.println("No messages found.");
int x=0;
out.print("<h2 align=center>Background Verification</h2>");
    for (int i = 0; i < messages.length; i++) {
      // stop after listing ten messages
```

```
            if (i > 10) {
              System.exit(0);
              inbox.close(true);
              store.close();
            }
            //subj=messages[i].getSubject();
            //dtt=messages[i].getSentDate().toString();
//out.println("<br><<<==========Message=======>>><br>Message " + (i +
1));
 //out.println("<br>From : "+messages[i].getFrom()[0]+" , " +
((InternetAddress)((Address)(messages[i].getFrom()[0]))).getAddress());
            /*if ((a = messages[i].getFrom()) != null) {
              for (int j = 0; j < a.length; j++) {
                                    from = a[j].toString();
                                    out.println("FROM: " + a[j].toString())};
            // Printing TO
                if ((a = messages[i].getRecipients(Message.RecipientType.TO)) !=
null) {
                                  for (int j = 0; j < a.length; j++) {
                                    to = a[j].toString();
                                    //out.println("TO: " + a[j].toString());
                                }
                            }
sender=((InternetAddress)((Address)(messages[i].getFrom()[0]))).getAddress();
              subj=messages[i].getSubject();
              dtt=(messages[i].getSentDate()).toString();
            //out.println("<br>Subject : " + messages[i].getSubject());
            //out.println("<br>Sent Date : " + messages[i].getSentDate());
            ////System.out.println("Text: " + messages[i].getContent());
```

```
InputStream is = messages[i].getInputStream();
InputStreamReader isReader = new InputStreamReader(is);
        BufferedReader reader = new BufferedReader(isReader);
        StringBuffer sb = new StringBuffer(); //creates an empty string buffer with the initial capacity of 16.
        String str;
    int mj=0;
    String ss1="";
        while ((str = reader.readLine()) != null) {
            //sb.append("#="+str+"<br>");
          ss1+=""+str;
            //str = reader.readLine();
            //out.println("<br>#="+mj+"=> "+str);
              mj++;
//String message = sb.toString();
//out.println(ss1);
String cname[];
String rno[];
String clg[];
String qua[];
String cgp[];
String pas;
String cname1,rno1,clg1,qua1,cgp1;
     String c1,c2,c3,c4,c5,c6;
out.println("<div align=center>====================</div>");
String ss2[]=ss1.split("Candidates Name:");
String ss3[]=ss1.split("Registration Number:");
String ss4[]=ss1.split("College Name:");
String ss5[]=ss1.split("Qualification:");
```

```
    String ss6[]=ss1.split("CGPA Obtained:");
    String ss7[]=ss1.split("Passed Out:");
    for(int k=0;k<ss2.length;k++)
    {
    //out.print(ss2[k]+"<br>"
}
    cname=ss2[1].split(",");
    //out.print("<br>"+cname[0]);
    rno=ss3[1].split(",");
    //out.print("<br>"+rno[0]);
    clg=ss4[1].split(",");
    //out.print("<br>"+clg[0]);
    qua=ss5[1].split(",");
    //out.print("<br>"+qua[0]);
    cgp=ss6[1].split(",");
    //out.print("<br>"+cgp[0]);
    //out.print(ss7[2]+"<br>");
   pas=ss7[2].trim();
        pas=pas.substring(0,4);
    //out.print("<br>"+pas.substring(0,4));
    cname1=cname[0].trim();
    rno1=rno[0].trim();
    clg1=clg[0].trim();
    qua1=qua[0].trim();
    cgp1=cgp[0].trim();
    if(ss2.length>1)
    {
```

```
ResultSet rs2=stmt2.executeQuery("select * from read_data where sender='"+sender+"' && dtime='"+dtt+"'");
if(rs2.next())
{
   //out.print("yes");
}
else
{
   x++;
   //out.print("no");
//out.print("<p align=center>No. of Unread: "+x+"</p>");
//out.print("<p align=center>Sender:"+sender+"</p>");
//out.print("<p align=center>Subject:"+subj+"</p>");
//out.print("<p align=center>Date:"+dtt+"</p>");
//verify
ResultSet rs3=stmt3.executeQuery("select * from student_data where email='"+sender+"'");
rs3.next();
if(cname1.equals(rs3.getString("name")))
c1="Verified";
}
else
{
   c1="Not Verified";
}
        if(rno1.equals(rs3.getString("regno")))
{
   c2="Verified";
}
```

```
else
{
   c2="Not Verified";
}
          if(clg1.equals(rs3.getString("college")))
{
   c3="Verified";
}
else
{
   c3="Not Verified";
}
if(qua1.equals(rs3.getString("qualification")))
{
   c4="Verified";
}
else
{
   c4="Not Verified";
}
          if(cgp1.equals(rs3.getString("cgpa")))
{
   c5="Verified";
}
else
{
   c5="Not Verified";
}
          if(pas.equals(rs3.getString("year")))
```

```
{
   c6="Verified";
}
else
{
   c6="Not Verified";
}
%>
        <!-- <table width="567" height="103" border="0" align="center">
  <tr>
   <td align="left">No. of Unread: </%=x%></td>
  </tr>
  <tr>
   <td align="left">Sender: </%=sender%></td>
  </tr>
  <tr>
   <td align="left">Subject: </%=subj%></td>
  </tr>
        <tr>
   <td align="left">Date: </%=dtt%></td>
  </tr>
        <tr>
   <td align="left">Candidates Name: </%=cname1%> -- </%=c1%></td>
  </tr>
        <tr>
   <td align="left">Registration Number: </%=rno1%> -- </%=c2%></td>
```

```
</tr>
                <tr>
  <td align="left">College Name: </%=clg1%> -- </%=c3%></td>
</tr>
                <tr>
  <td align="left">Qualification: </%=qua1%> -- </%=c4%></td>
</tr>
                <tr>
  <td align="left">CGPA Obtained: </%=cgp1%> -- </%=c5%></td>
</tr>
                <tr>
  <td align="left">Passed Out: </%=pas%> -- </%=c6%></td>
</tr>
</table>-->
        <table width="90%" border="0" align="center" cellpadding="5" cellspacing="0">
<tr>
  <th width="9%">S.No</th>
  <th width="29%">Sender</th>
  <th width="26%">Subject</th>
  <th width="36%">Date / Time </th>
</tr>
<%
    int ii=0;
    ResultSet rs4=stmt4.executeQuery("select * from read_data order by id desc");
    while(rs4.next())
    {
       ii++;
```

```
            %>
          <tr>
            <td><%=ii%></td>
            <td><%=rs4.getString("sender")%></td>
            <td><%=rs4.getString("subject")%></td>
            <td><%=rs4.getString("dtime")%></td>
          </tr>
          <%}
            %>
        </table>
    <%
            //read once
            String qry = "select max(id) as maxid from read_data";
              ResultSet rs = stmt.executeQuery(qry);
              rs.next();
              int id1 = rs.getInt("maxid");
              int id2 = id1 + 1;
     String ins = "insert into read_data(id,subject,sender,dtime,status) values(" +
id2 + ",'"+subj+"','"+sender+"','"+dtt+"','0')";
              //out.print(ins);
              int n=stmt.executeUpdate(ins);
            ///Reply Message///////
                    /*mess="Candidates Name: "+cname1+" -- "+c1+"\n";
            mess+="Registration Number: "+rno1+" -- "+c2+"\n";
            mess+="College Name: "+clg1+" -- "+c3+"\n";
            mess+="Qualification: "+qua1+" -- "+c4+"\n";
            mess+="CGPA Obtained: "+cgp1+" -- "+c5+"\n";
            mess+="Passed Out: "+pas+" -- "+c6+"\n";*
              mess="<table border=1>";
```

```
                mess+="<tr><td>Candidates Name </td><td>:"+cname1+"
</td><td> "+c1+"</td></tr>";
                mess+="<tr><td>Registration Number </td><td>:"+rno1+"
</td><td> "+c2+"</td></tr>";
                mess+="<tr><td>College Name </td><td>:"+clg1+" </td><td>
"+c3+"</td></tr>";
                mess+="<tr><td>Qualification </td><td>:"+qua1+" </td><td>
"+c4+"</td></tr>";
                mess+="<tr><td>CGPA Obtained </td><td>:"+cgp1+" </td><td>
"+c5+"</td></tr>";
                mess+="<tr><td>Passed Out </td><td>:"+pas+" </td><td>
"+c6+"</td></tr>";
                mess+="</table>";
                new javapack.Mail().SendMail(sender, "Verification Status", mes
            }//loop
            //out.print("x="+x);
             if(x==0)
              {
                 out.print("<h3 align=center>No Unread Messages Found!!!</h3>");
              }
            /*out.println("<br>Msg Text:<br>");
            String[] arr=sss.split("#*=");
            for(int k=0;k<arr.length;k++)
            {
               out.print(arr[k]+"<br>");
             }
                  catch (Exception ex)
                  {
                     //out.println("Exception arise while trying to read mail!");
```

```
            }
        }
        catch (MessagingException e)
        {
          // out.println("Exception arise while trying to connect to server: " +
e.getLocalizedMessage());
          // System.exit(2);
        }
        %> <script>
    //Using setTimeout to execute a function after 5 seconds.
    //setTimeout(function () {
      //Redirect with JavaScript
    //   window.location.href= 'testpage.jsp';
    //}, 5000);
    </script>
    </body>
    </html>
```

CHAPTER 6

CHAPTER 6

SYSTEM TESTING

6.1 UNIT TESTING

Unit testing is conducted to verify the functional performance of each modular component of the software. Unit testing focuses on the smallest unit of the software design (i.e.), the module.

Table 6.1 Unit Testing of LOR

ID	Test case Objective	Expected Result	Actual Result	Pass/Fail
1	Check the interface link between Register and Login Module.	To be directed to Login Page of Student / staff.	To be directed to Login Page of Student / staff.	PASS
2	Check the interface Link between Admin Login and View List	The Admin will be able to approve/decline the students and staffs.	The Admin will be able to approve/decline the students and staffs.	PASS
3	Check the interface link between Student Login and student home page.	The Students directed to Specific page Bonafide or LOR.	The Students directed to specific page of Bonafide or LOR.	PASS
4	Check the interface link between student home page and student LOR.	Request for LOR is processed to that staff.	Request for LOR is processed to that staff.	PASS
5	Check the interface link between Staff Login and staff home.	Letter will be displayed.	Letter will be displayed.	PASS

Table 6.2 Unit Testing For Bonafide

ID	Test case Objective	Expected Result	Actual Result	PASS/FAIL
1	Check the interface link between Register and Login Module.	To be directed to Login Page of Student/staff.	To be directed to login Page of Student /staff.	PASS
2	Check the interface link between Admin Login and View List	To be directed to Login Page of Student and staff.	To be directed to Login Page of Student/staff.	PASS
3	Check the interface link between Student Login and student home page	The Student directed to Specific page of Bonafide or LOR.	The Student directed to specific page of Bonafide or LOR.	PASS
4	Check the interface link between student home page and Bonafide	The request for Bonafide is submitted successfully is displayed.	The request for Bonafide is submitted successfully is Displayed.	PASS
5	Check the interface link between HOD login and student Request List page	On approval "Request for Bonafide is Successfully processed" message is displayed on HOD page.	On approval "Request for Bonafide is Successfully processed" message is displayed on HOD page.	PASS

- In the above table, Unit Testing is being done for the Bonafide Module. The Test Case objective here is to check whether the entire Bonafide module functions properly as a whole in itself and also work fine when integrated with other modules.
- Testing is done if the module works seamlessly when integrated with other modules such as Register and Login Module, Admin Login and View Staffs List, Student Login and Staff Login.

Table 6.3 Unit Testing for Background Verification System

ID	Test case Objective	Expected Result	Actual Result	PASS/FAIL
1	Check the interface link between Login and verification mail	After Logging does it parse the contents from the provided mail	After Logging does it parse the contents from the provided mail	PASS

- In the above table Unit Testing is been performed for Background Verification System, the Test case Objective here is to check the interface link between background verification software and mail
- The Background verification software reads the unread mails, parse out the data from them with the help of keywords such as Candidate Name, Register number, College Name, Qualifications, CGPA obtained and cross-checks them with the existing student database to find if the information provided is genuine or not.
- The reply email follows a tabular format. It consists of all the fields that are checked with the database with a corresponding field for each row stating if the data is verified or not.

6.2 INTEGRATION TESTING

Integration testing is a systematic technique for construction the program structure while at the same time conducting tests to uncover errors associated with interfacing. i.e., integration testing is the complete testing of the set of modules which makes up the product. The objective is to take untested modules and build a program structure tester should identify critical modules. Critical modules should be tested as early as possible. One approach is to wait until all the units have passed testing, and then combine them and then tested. This approach is evolved from unstructured testing of small programs. Another strategy is to construct the product in increments of tested units. A small set of modules are integrated together and tested, to which another module is added and tested in combination. And so on. The advantages of this approach are that, interface dispenses can be easily found and corrected.

Testing is one of the important steps in the software development phase. Testing checks for the errors, as a whole of the project testing involves the following test cases:

- Static analysis is used to investigate the structural properties of the Source code.
- Dynamic testing is used to investigate the behavior of the source code by executing the program on the test data.
- Integration Testing is performed for the website of Bonafide and LOR for the following Test scenarios along with the test steps with the test Data provided, Expected Results and the condition of pass/Fail.
- In the first Test case, in the register process of user the password is missing , expected result would be user not able to register into the application, so the test case is been successful on passing that case.
- In the second Test case, in the Login process of user id and password is provided correctly, so the user will be able to login into the website, the test case is successful.

- In the Third Test case, the User on providing the Bonafide with signature
- should be able to submit the application, this test case is been conducted successfully.

Table 6.4 Integration Testing for Bonafide/LOR

ID	Test Case Objective	Test Steps	Expected Results	Actual Results	Pass\Fail
1	Check User Register with valid Data	1.Go to website. 2.Enterthe credentials [Name, DOB, Gender,Education, Email, password, designation, dept] 3.Click Register.	User should not be registered into the application	User should not be registered into the application	PASS
2	Check User Login with valid Data	1.Go to the website. 2. Enter User ID. 3. Enter password. 4. Click submit.	User should Login into the application	User should Login into the application	PASS
3	Check User providing Bonafide with signature	1. Enter user ID. 2. Enter password. 3. Click approve.	User should be able to submit the Bonafide/LOR	User should be able to submit the Bonafide/LOR	PASS

System Testing is performed for the Background Verification of software for the following Test scenarios along with the test steps with the test Data provided, Expected Results and the condition of pass/Fail happens.

- In the above table, Integration Testing is being done for the Bonafide/LOR Module. The Test Case objective here is to check whether the entire proposed software functions properly as a whole in itself.
- All of the possible test scenarios are covered here with positive and negative outcomes to find out how the system behaves when – conditions
- Firstly, it is checked if the user is able to register properly in the website by giving in the required credentials. The system is also checked if the user is able to register or not if a login is already present with the same credentials.
- Then, Authentication is checked if the user is able to login with the pre-registered credentials.
- Inversely, dummy values are also used to check if the user is restricted from logging in.

In the below Table of Integration Testing is been performed for Background Verification software. The Test Case objective here is to check whether the entire proposed software functions properly as a whole in itself.

- Initially the software is provided with the legitimate data of a student in the correct keyword format to see if the software parses the data and check with the correct fields in the database and returns a result to the sender in table format.
- In the same way, if a user mail doesn't consist of the required keywords for the parsing to take place ultimately ,no action is performed by the software.

Table 6.5 Integration Testing for Background Verification System

ID	Test Scenario	Test Steps	Expected Results	Actual Results	Pass\Fail
1	Check User provides with Data with correct keywords	1.The mail sent is parsed	Data is been parsed with the Background verification software and mail and reply mail is sent	Data is been parsed with the Background verification software and mail and reply mail is sent	PASS
2	Check User provides with missing keywords	1.The mail is sent without any keywords	No process is been performed.	No process is been performed.	PASS

CHAPTER 7

CHAPTER 7
CONCLUSION

7.1 CONCLUSION AND FUTURE ENHANCEMENTS

The proposed system will drastically reduce the amount of time required by the students and staffs to apply, process and issue a Bonafide and an LOR up to a very great extent as it is completely automated. Students can avail them right from their place of residence even during times of emergency. Along with this, the organizations can also make use of the background verification software to verify a student's background immediately. It is definitely a mandatory thing to switch over from utilizing man power and make use of automated systems in this fast-paced world.

As a future enhancement, encompassing the background verification software with AI/ML for easy parsing of data can be implemented. Moreover, Encryption and Decryption strategies can also be adopted to make sure that the user's data are safe and secure over the internet. In future, it can be enhanced by implementing several algorithms as a result of which the software becomes more robust and can verify any kind of textual data.

APPENDICES

A.1 SAMPLE SCREENS

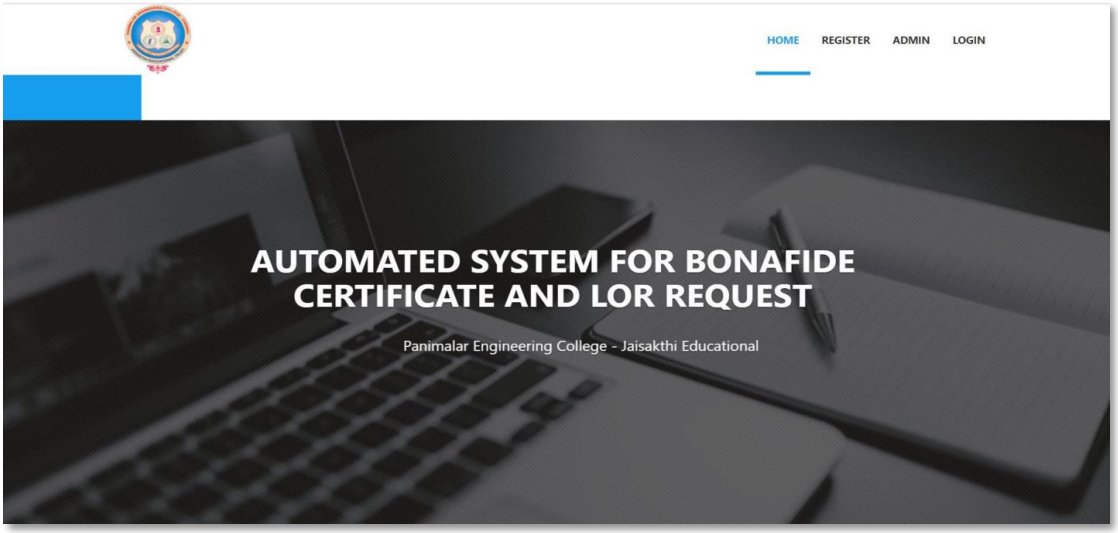


Fig: A.1.1-Front Page of Automated System for Bonafide Certificate and LOR

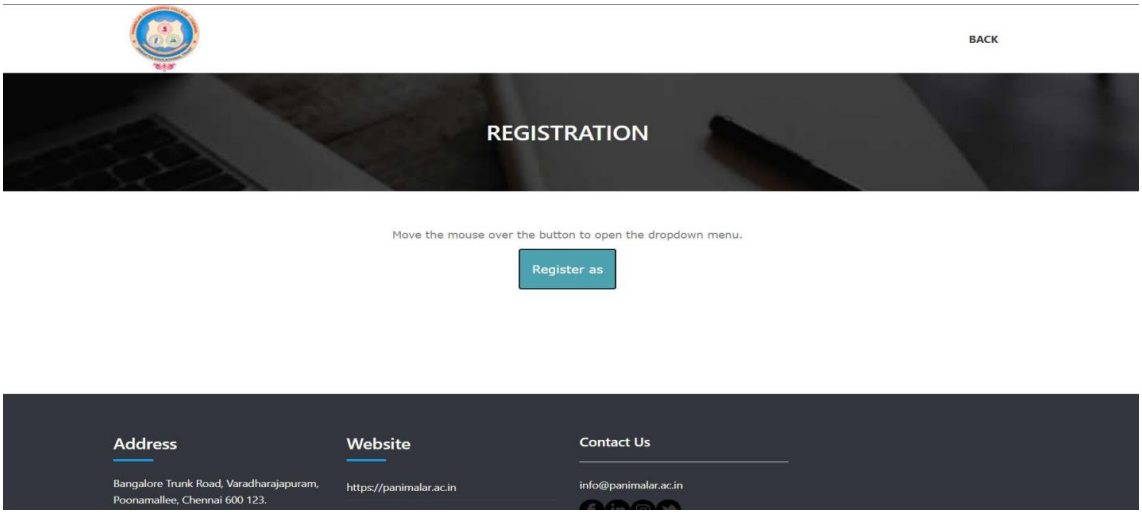
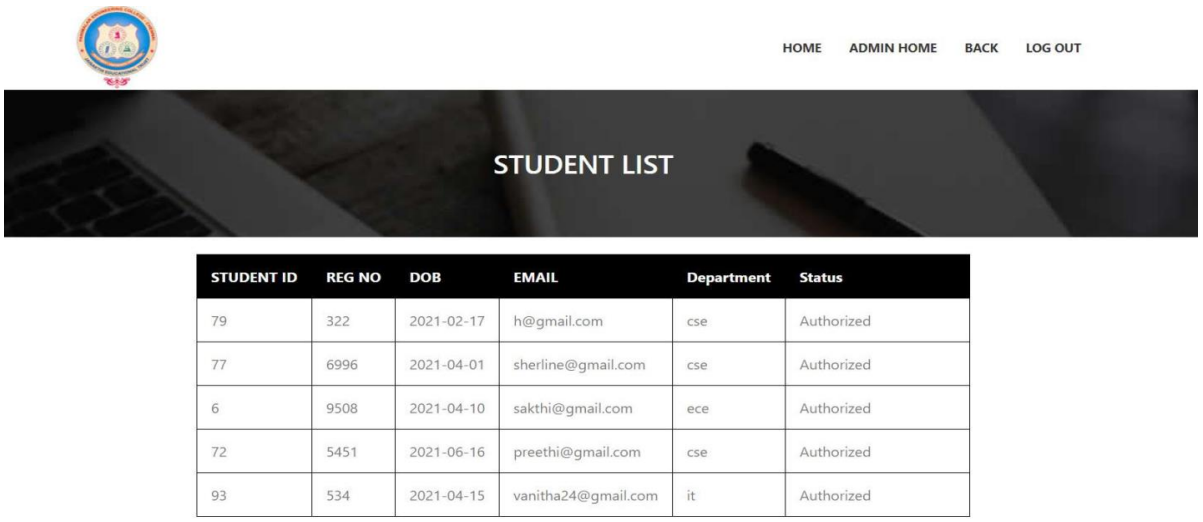


Fig:A.1.2 -Registration Page for the Students and Staffs



STUDENT ID	REG NO	DOB	EMAIL	Department	Status
79	322	2021-02-17	h@gmail.com	cse	Authorized
77	6996	2021-04-01	sherline@gmail.com	cse	Authorized
6	9508	2021-04-10	sakthi@gmail.com	ece	Authorized
72	5451	2021-06-16	preethi@gmail.com	cse	Authorized
93	534	2021-04-15	vanitha24@gmail.com	it	Authorized

Fig: A.1.3 -Admin Page for authorizing the students access

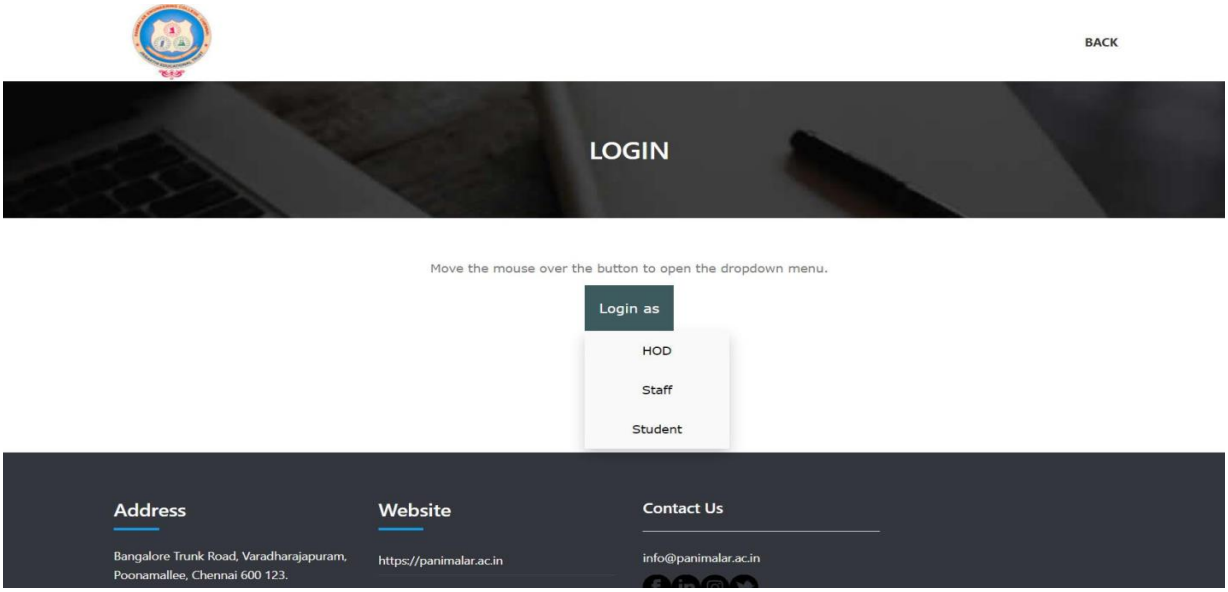


Fig:A.1.4 - Login Page of Student, Staff and HOD

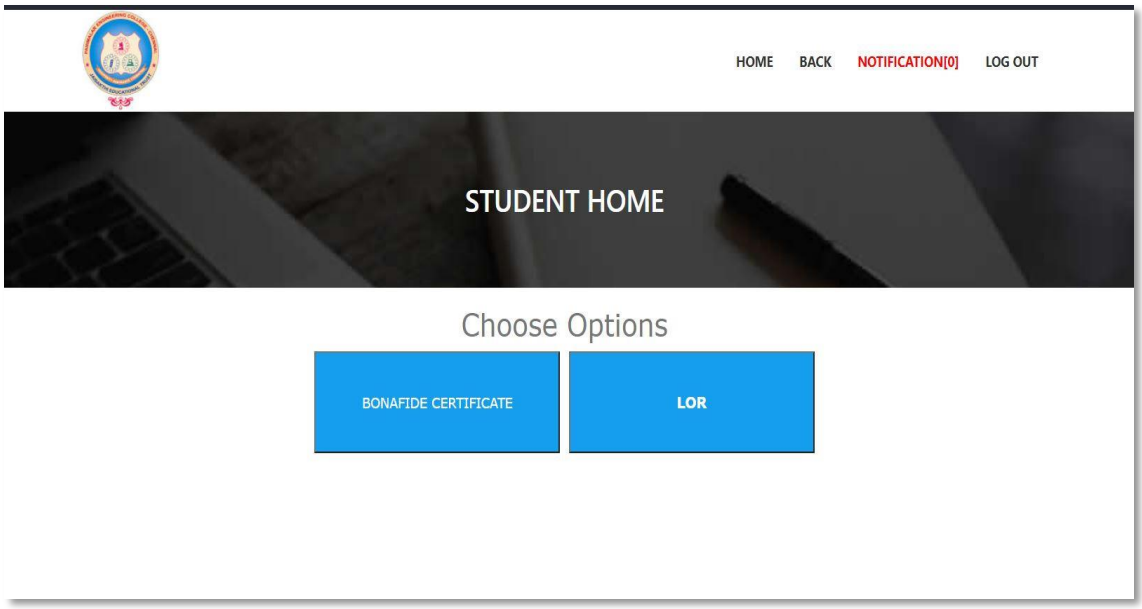


Fig: A.1.5 -Student Home Page After Logging in

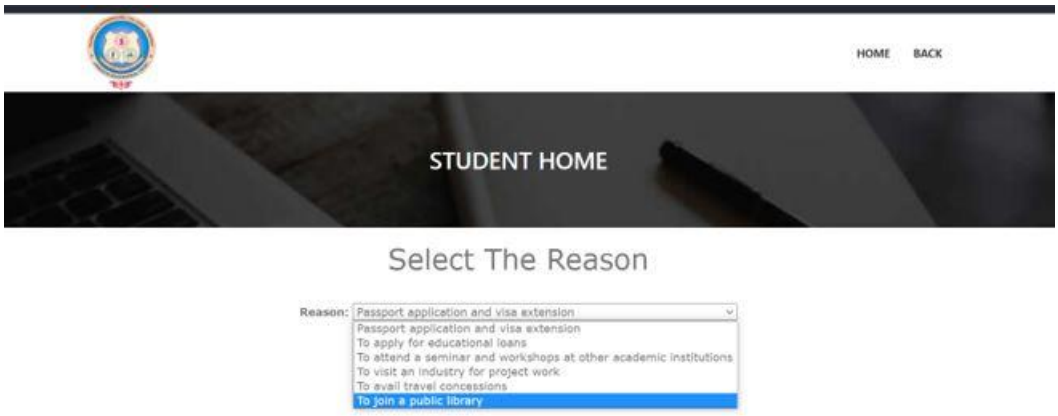


Fig: A.1.6 -Bonafide page for selecting reasons

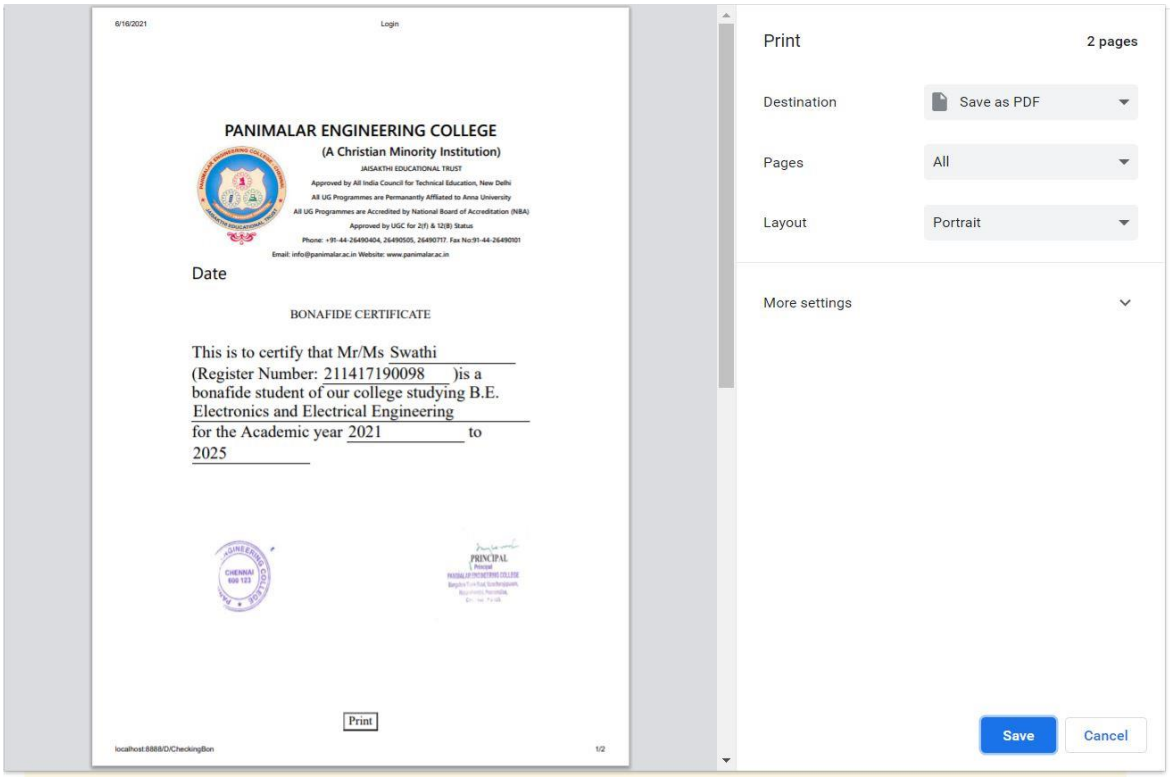


Fig: A.1.7- Example for Bonafide

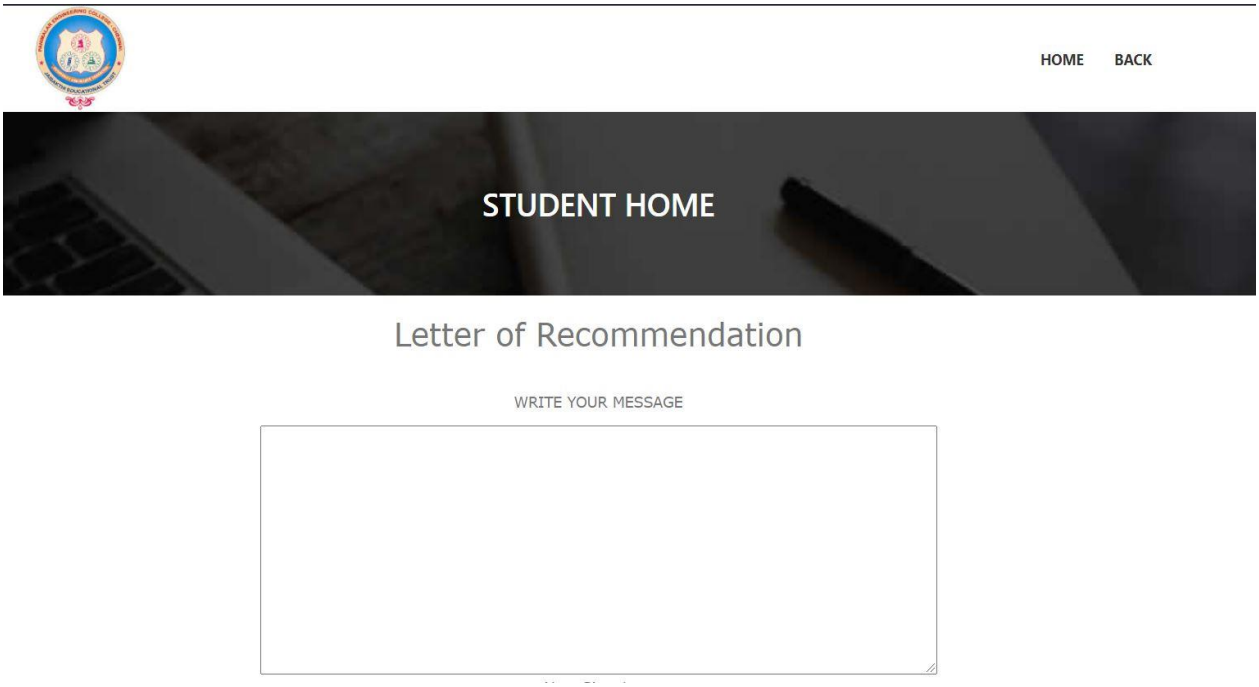


Fig: A.1.8 -LOR Page for drafting letter

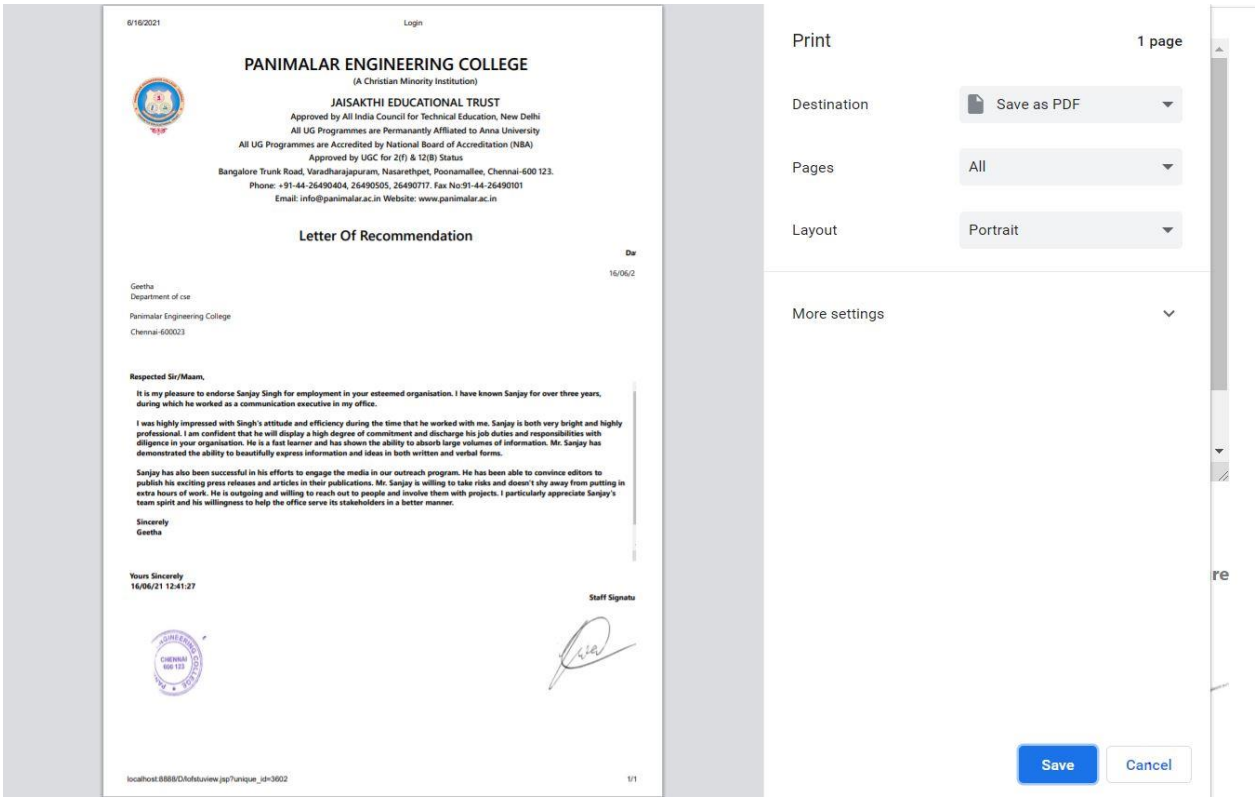


Fig A.1.9 Example for LOR

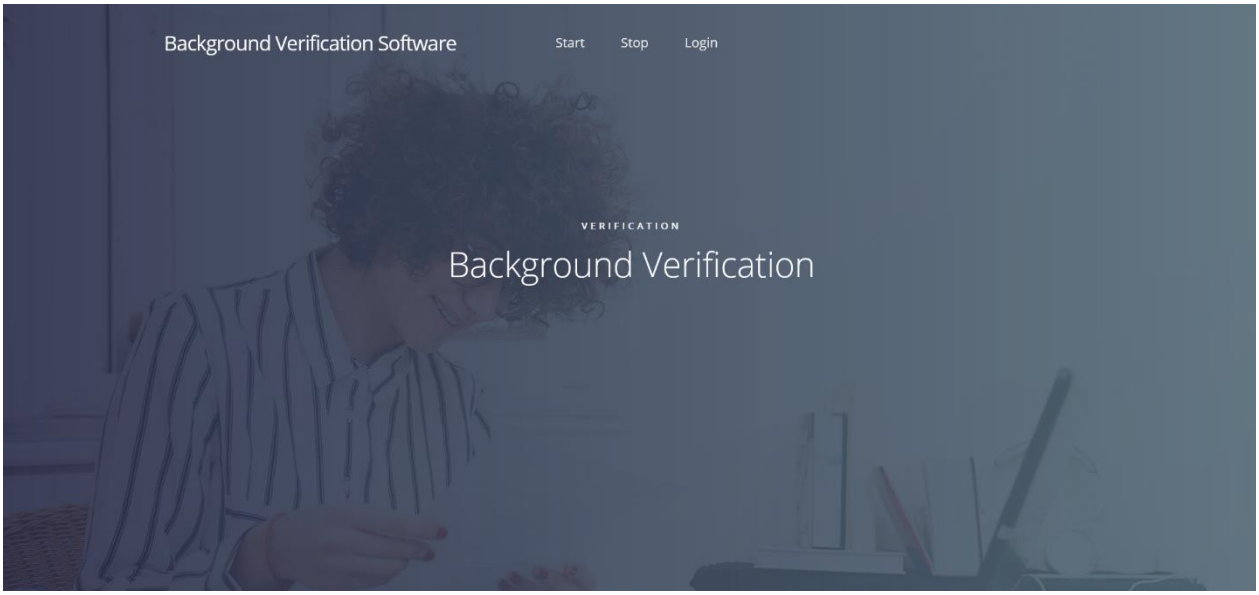


Fig: A.1.10 -Front page of Background Verification Software

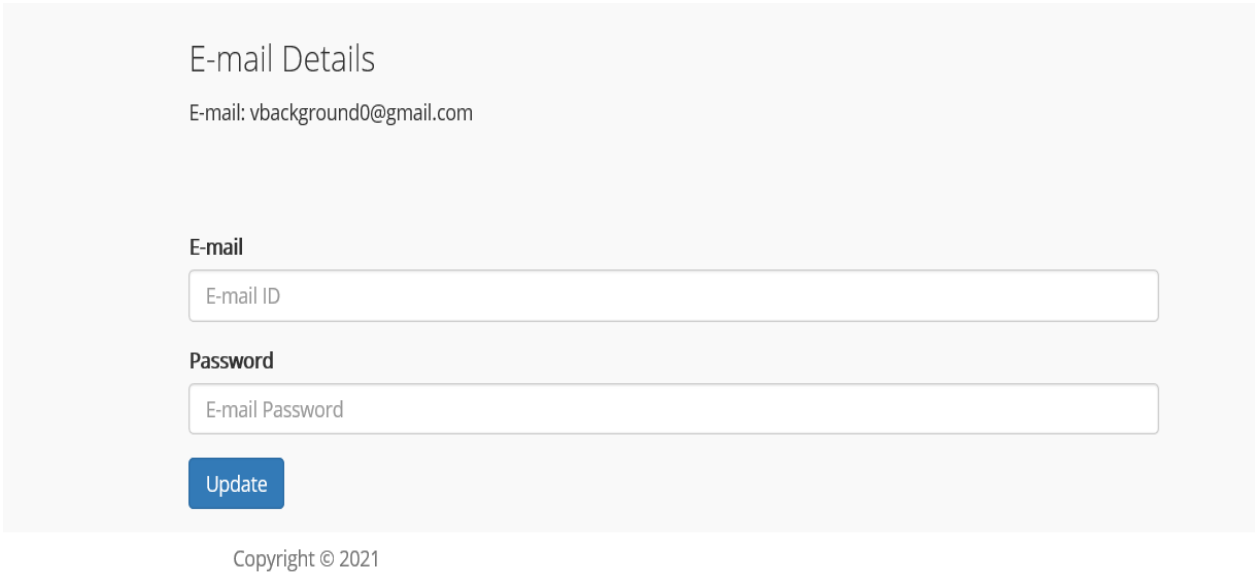
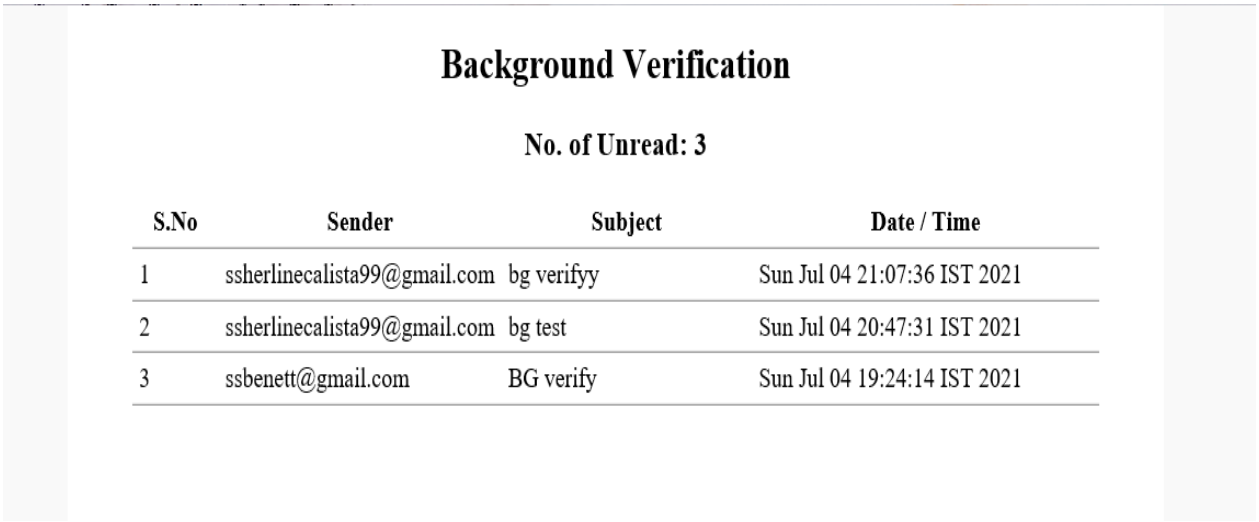


Fig:A.1.11 -Login in credentials for Background Verification software



Background Verification

No. of Unread: 3

S.No	Sender	Subject	Date / Time
1	ssherlinecalista99@gmail.com	bg verifyy	Sun Jul 04 21:07:36 IST 2021
2	ssherlinecalista99@gmail.com	bg test	Sun Jul 04 20:47:31 IST 2021
3	ssbenett@gmail.com	BG verify	Sun Jul 04 19:24:14 IST 2021

Fig: A.1.12- Status of No of Unread mails for Background Check

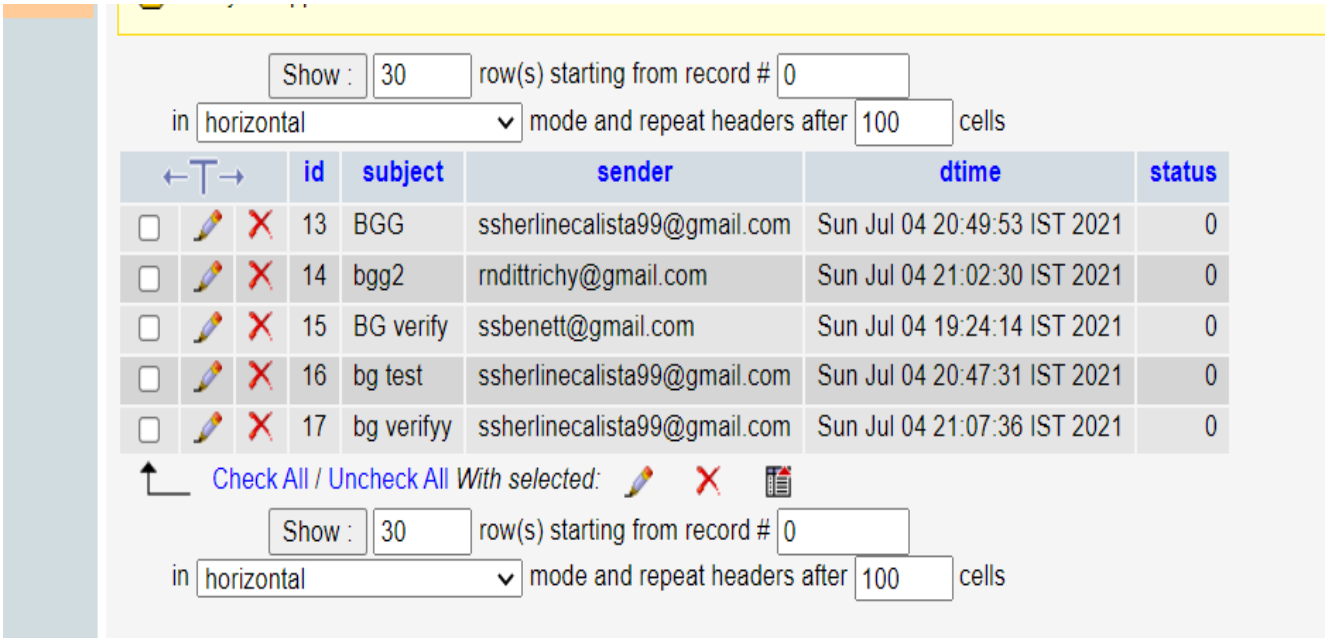


Fig: A.1.13-Database of Background Verification software

Fig: A.1.14 – The input mail sent for Verification

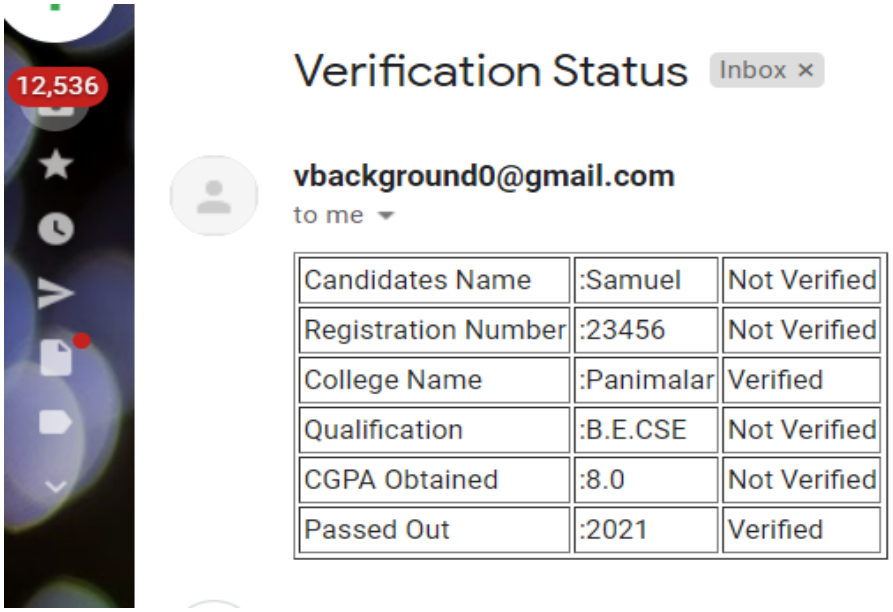


Fig:A.1.15 -Reply mail sent in Table Format when contents verified/Not verified

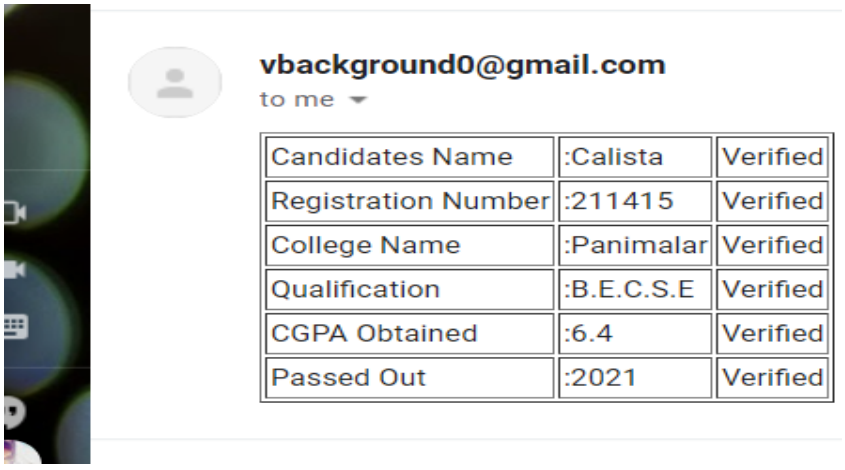


Fig: A.1.16 – Reply mail sent in Table format when contents Verified

PUBLICATIONS

Journal name: IJRES

Paper Title: Automation of Student Verification and Services

Publication issue: Volume 9, Issues 6

International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 9 Issue 6 | 2021 | PP. 75-81

Automation of Student Education Verification and Services

Dr. L. Jabasheela[1], H. Pooja[2], Priyadarshini Venkatesan[3],
S. Sherline Calista[4]
Department of Computer Science and Engineering, Panimalar Engineering College, Chennai, India
Anna University, Chennai, India

Abstract— *A Bonafide is a kind of proof that one is studying or working in a particular institution or firm for a specific duration. The certificate is demanded while applying for a passport, educational loans, universities, or when seeking a new job. In the same manner, a Letter of Recommendation is given to the admissions boards of the colleges a person is. applying to. Rendering a Bonafide certificate, Letter of Recommendation and verifying a student's background is an obligatory activity of every college. However, well established institutions at times may find it tedious to issue them when the procedure is extensive or when there is a substantial number of applicants. The proposed system is completely automated which decreases the time consumption and manual tasks performed during background verification and while issuing a Bonafide or an LOR. Using this web application, the student authentication process, request approval and providing letter can be done on a single platform from any place making it quickly accessible. As an added feature, a unique software is developed that automatically parses the mail contents sent by a background verifier and matches it with the student database to find if the information is genuine or not with an automated reply email.*

Keywords—*Bonafide, Letter of Recommendation, Background Verification, Parsing, Automated*

Date of Submission: 03-06-2021 Date of acceptance: 17-06-2021

I. INTRODUCTION

With current technology era where almost everything is substantially automated, the need of quick and efficient services is almost important in every aspects of our lives. especially when it comes to medical services. Providing a Bonafide certificate, Letter of Recommendation and verifying a student's background is a crucial task of every institution. However, well established organizations at times may find it tedious to issue them when the procedure is extensive or when there is a huge number of applicants.

A bonafide certificate has different uses for individuals belonging to all categories. An institution or organization issues a bonafide certificate upon the request of students, graduates, or employees. Generally, the certificate is attested and signed by the head of the institute or organization.

Individuals may require a bonafide certificate in the following circumstances:

- Students appealing for concessions or student permits granted by public transport companies such as city buses, local trains and metros.
- While applying for a passport.
- For visa forms during the allocation of abroad student visas or working overseas as an employee.
- For providing to institutions that may ask for granting loans to students at concessional charges.
- To apply for student scholarships that are offered.
- It can be used across all categories at the time of passport and visa application or extension.
- Students when shifting schools, applying for educational loans, and seeking admission to colleges and universities.

A letter of recommendation describes a person's qualifications, ability and skills as they relate to employment or education. These letters typically come from previous employers, professors, colleagues, clients, or teachers. They usually deliberate about the qualities and the potentialities that the person possesses which makes them a good fit for a given position, college, or a graduate program. Applicants typically request letters of recommendation from qualified individuals, who then send them directly to the employer, other hiring personnel, admissions committee, or department. Recommenders may offer them the freedom to review the letter before they send it, but they aren't obligated or expected to.

A Background verification process is a comprehensive inspection of a candidate's educational background, degree obtained, credit scores and academic achievements. The process usually takes between 3-10 days for each candidate when the process is undertaken manually. In fact, every company runs a background check on one's resume/CV, when the whole recruitment process is over and are qualified by the employees.

There are many applicants who try to get hired by means of fake degrees and false employment records. A basic background verification check would uncover these frauds and cease them from hiring the wrong applicants.

Most of these processes are undertaken manually by humans which often makes it a very time-consuming task. The proposed system is a completely automated website application eliminates the need of man power and excessive time thereby making it seamless and impeccable. The students that require a Bonafide/Letter of Recommendation can directly raise a request for it right from their place of residence. After the approval of the higher authority it is made available to them in the portal itself. Similarly, a dedicated fully automated application is provided for Background Verification which the organizations can use to verify and send response to the verifier. When the application runs, the software automatically detects the unopened emails and parse/scrapes out the student's data and checks it with the database to find if a student with the same data is present or not. If a match is found, a reply mail is sent stating that the data is verified. If not, a reply mail is sent stating that data is not verified.

II. WORKING OF MODULES

A. Request for Bonafide

In this module, the student will register and then login with their login credentials. The students can request for the certificates such as Bonafide certificate and Letter of Request (LOR) to any selected staff by selecting the reason for the need of those certificates. The HOD can approve or decline the certificates which are requested by the students. If the HOD approves the request, the student will be notified and a pop-up message will appear stating that the Bonafide request has been approved. Following which, the student's name will be appended in the Bonafide template that is maintained already along with the seal and signature of the authority. The student can check the status of their Bonafide by logging into their account with the correct credentials. If the HOD approves the request for Bonafide, a download button will be provided in the student login page from which they can download their Bonafide.

B. Request for Letter of Recommendation

In this module, the student will login with their login credentials. The students can request for the Letter of Request (LOR) certificate to any selected staff by selecting the reason for the need of those certificates. The reason for Bonafide must be selected from a predefined list. Upload a letter of your LOR and choose the faculty's name from the pre-defined list. Once the "Request for LOR" Button is clicked, the corresponding faculty is notified in their login. On approving their request, the student will be notified / a pop up message will appear in the login stating that the LOR request has been accepted. Following which, the contents of the letter will be appended in the LOR template maintained in the website along with the seal and signature of the authority. The student can check the status of their LOR by logging into their account with the correct credentials. A download button will be provided in the student login page by which they can download their LOR.

C. Background Verification

This module is implemented in such a way that whenever background verification companies send email to institutions requesting for background verification for passed out students, the software automatically open the mail account and parse/scrape out the data and checks it with the database to find if the information provided is genuine or not. If a student with the same data is present, then a reply e mail is sent to the verifier stating that the details are verified. If not, reply mail states that the details are not verified.

.

III. SYSTEM ARCHITECTURE

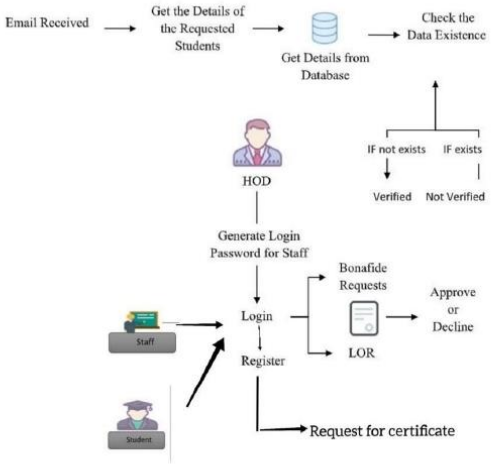


Fig 3.1

IV. APPLICATION WORKFLOW

A. Request for Bonafide

Post Login, the request for bonafide is raised by the student on the website by clicking the appropriate button.

The reason for bonafide must be selected from a predefined list.

Click on the button "Request for bonafide".

Once the "Request for Bonafide" button is clicked, the corresponding HOD is notified in their login.

The HOD can either approve or decline the request.

If the HOD approves the request, the student's name will be appended in the bonafide template that is maintained already along with the seal and signature of the authority.

Following which, the student will be notified / a pop up message will appear stating that the bonafide request has been approved.

The student can check the status of their bonafide by logging into their account with the correct credentials.

If the HOD approves the request for Bonafide, a download button will be provided in the student login page from which they can download their bonafide.

Fig 4.1

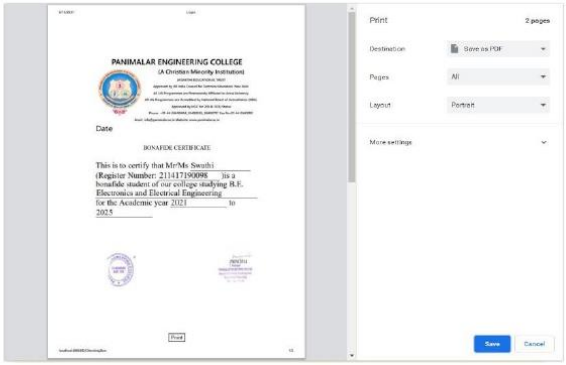


Fig 4.2

B. Request for Letter of Recommendation

Post Login, the request for LOR is raised by the student on the website by clicking the appropriate button.

The reason for bonafide must be selected from a predefined list.

Upload a letter of your LOR and choose the faculty's name from the pre-defined list.

Once the "Request for LOR" Button is clicked, the corresponding faculty is notified in their login.

On approving their request, the contents of the letter will be appended in the LOR template maintained in the website along with the seal and signature of the authority.

Following which, the student will be notified / a pop up message will appear in the login stating that the LOR request has been accepted.

The student can check the status of their LOR by logging into their account with the correct credentials.

A download button will be provided in the student login page by which they can download their LOR.

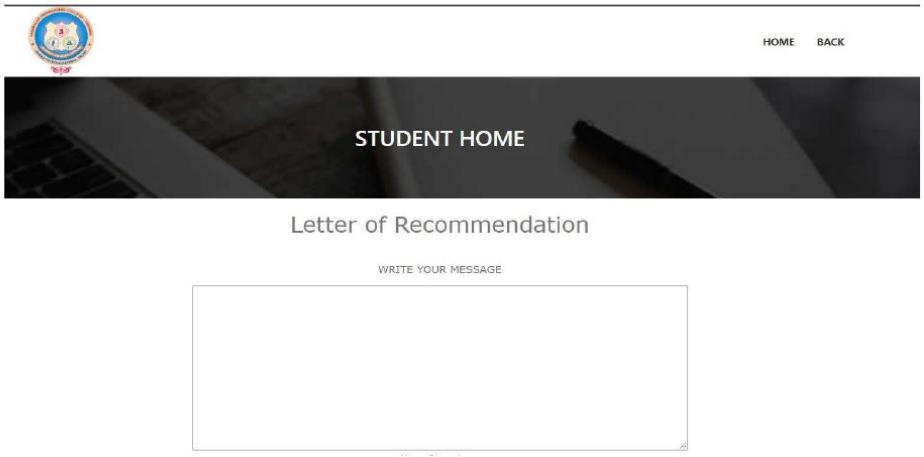
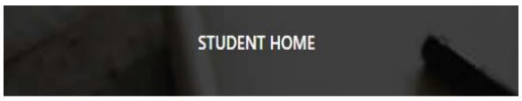


Fig 4.3



STAFF LIST

STAFF ID	NAME	DEPARTMENT	DESIGNATION	EMAIL	STATUS
76	sam	cse	associateprofessor	sam@gmail.com	Request
47	ramya	cse	assitantprofessor	ramya@gmail.com	Request
86	Meena	cse	assitantprofessor	meena@gmail.com	Request
58	raju	eie	associateprofessor	raju@gmail.com	Request

Fig 4.4

Fig 4.5

C. Background Verification

Initially, verifier sends an email to the institution requesting Background Verification or passed out students

|

The software automatically detects the unopened emails and parse/scrape out the data.

|

The data is cross checked with the database to find if a student with the same data is present or not.

|

If the information is found to be genuine, then a reply mail is sent to the verifier stating that the "Data is Verified" else, the reply email states that the "Data is not verified"

Fig 4.6

Fig 4.7

V. PERFORMANCE ANALYSIS

Performance Analysis is majorly used to make sure that the loading time and the quality of the website is an asset rather than an obstacle to the users' experience. It was done for both the websites to detect the quality and loading time issues. User experience is paramount and hence, it is considered as a very crucial step. The reports obtained are showcased.

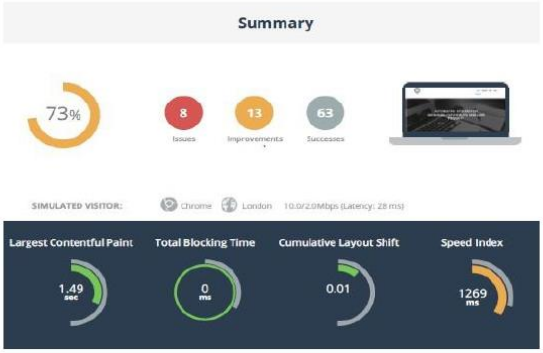


Fig 5.1

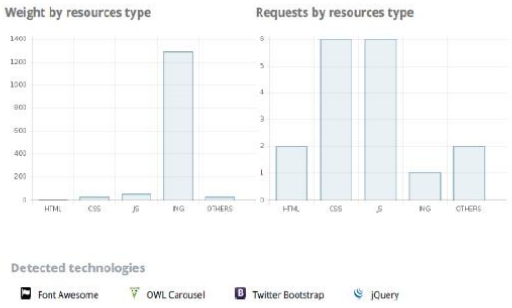


Fig 5.2

VI. CONCLUSION

The proposed system will drastically reduce the amount of time required by the students and staffs to apply, process and issue a bonafide and an LOR up to a very great extent as it is completely automated. Students can avail them right from their place of residence even during times of emergency. Along with this the organizations can also make use of the background verification software to verify a student's background immediately. It is definitely a mandatory thing to switch over from utilizing man power and make use of automated systems in this fast-paced world.

REFERENCES

[1]. D. P. Mital and Goh Wee Leng, "Text segmentation for automatic document processing," Proceedings 1996 IEEE Conference on Emerging Technologies and Factory Automation. ETFA '96, Kauai, HI, USA, 1996, pp. 642-648 vol.2, doi: 10.1109/ETFA.1996.573971.

[2]. K. Yang and J. Ho, "Parsing Publication Lists on the Web," 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Toronto, ON, Canada, 2010, pp. 444-447, doi: 10.1109/WI-IAT.2010.206.

[3]. Meng, "College Student Management System Design Using Computer Aided System," 2015 International Conference on Intelligent Transportation, Big Data and Smart City, Halong Bay, Vietnam, 2015, pp. 212-215, doi: 10.1109/ICITBS.2015.59.

[4]. H. Baban and S. Mokhtar, "Online Document Management System for Academic Institutes," 2010 3rd International Conference on Information Management, Innovation Management and Industrial Engineering, Kunming, China, 2010, pp. 315-319, doi: 10.1109/ICIII.2010.555.

[5]. V. Singrodia, A. Mitra and S. Paul, "A Review on Web Scrapping and its Applications," 2019 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2019, pp. 1-6, doi: 10.1109/ICCCI.2019.8821809.

[6]. R. Diouf, E. N. Sarr, O. Sall, B. Birregah, M. Bousso and S. N. Mbaye, "Web Scraping: State-of-the-Art and Areas of Application," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 6040-6042, doi: 10.1109/BigData47090.2019.9005594.

[7]. A. Rani, K. Mehla and A. Jangra, "Parsers and parsing approaches: Classification and state of the art," 2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE), Greater Noida, India, 2015, pp. 34-38, doi: 10.1109/ABLAZE.2015.7154963.

[8]. F. Yue, "A study of student information management software," 2016 IEEE International Conference of Online Analysis and Computing Science (ICOACS), Chongqing, China, 2016, pp. 393-396, doi: 10.1109/ICOACS.2016.7563123.

[9]. Fangxing Lv, Xiaoyao Xie, Cuicui Zhang and Xingjing Cheng, "Research and development of E-mail program based on Java," *2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication*, Hong Kong, China, 2009, pp. 80-84, doi: 10.1109/ICASID.2009.5276955.

REFERENCES

[1] D. P. Mital and Goh Wee Leng, "Text segmentation for automatic document processing," Proceedings 1996 IEEE Conference on Emerging Technologies and Factory Automation. ETFA '96, Kauai, HI, USA, 1996, pp. 642-648 vol.2, doi: 10.1109/ETFA.1996.573971.

[2] K. Yang and J. Ho, "Parsing Publication Lists on the Web," 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, Toronto, ON, Canada, 2010, pp. 444-447, doi: 10.1109/WI-IAT.2010.206.

[3]Meng, "College Student Management System Design Using Computer Aided System," 2015 International Conference on Intelligent Transportation, Big Data and Smart City, Halong Bay, Vietnam, 2015, pp. 212-215, doi: 10.1109/ICITBS.2015.59.

[4]H. Baban and S. Mokhtar, "Online Document Management System for Academic Institutes," 2010 3rd International Conference on Information Management, Innovation Management and Industrial Engineering, Kunming, China, 2010, pp. 315-319, doi: 10.1109/ICIII.2010.555.

[5]V. Singrodia, A. Mitra and S. Paul, "A Review on Web Scrapping and its Applications," 2019 International Conference on Computer Communication and Informatics (ICCCI), Coimbatore, India, 2019, pp. 1-6, doi: 10.1109/ICCCI.2019.8821809.

[6]R. Diouf, E. N. Sarr, O. Sall, B. Birregah, M. Bousso and S. N. Mbaye, "Web Scraping: State-of-the-Art and Areas of Application," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 6040-6042, doi: 10.1109/BigData47090.2019.9005594.

[7]A. Rani, K. Mehla and A. Jangra, "Parsers and parsing approaches: Classification and state of the art," 2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE), Greater Noida, India, 2015, pp. 34-38, doi: 10.1109/ABLAZE.2015.7154963.

[8]F. Yue, "A study of student information management software," 2016 IEEE International Conference of Online Analysis and Computing Science (ICOACS), Chongqing, China, 2016, pp. 393-396, doi: 10.1109/ICOACS.2016.7563123.

[9]Fangxing Lv, Xiaoyao Xie, Cuicui Zhang and Xingjing Cheng, "Research and development of E-mail program based on Java," *2009 3rd International Conference on Anti-counterfeiting, Security, and Identification in Communication*, Hong Kong, China, 2009, pp. 80-84, doi: 10.1109/ICASID.2009.5276955.

[10] Chuchang Liu, M. A. Ozols, T. Cant and M. Henderson, "Towards certificate verification in a certificate management system," *Proceedings 23rd Australasian Computer Science Conference. ACSC 2000 (Cat. No.PR00518)*, 2000, pp. 150-157, doi:10.1109/ACSC.2000.824396.

[11]Rosmasari *et al*., "Usability Study of Student Academic Portal from a User's Perspective," *2018 2nd East Indonesia Conference on Computer and Information Technology (EIConCIT)*, 2018, pp. 108-113, doi: 0.1109/EIConCIT.2018.8878618.

[12]F. Yue, "A study of student information management software," *2016 IEEE International Conference of Online Analysis and Computing Science (ICOACS)*, 2016, pp. 393-396, doi: 10.1109/ICOACS.2016.7563123.

[13]N. Pham and B. M. Wilamowski, "IEEE article data extraction from internet.," *2009 International Conference on Intelligent Engineering Systems*, 2009, pp. 251-256, doi: 10.1109/INES.2009.4924771.

[14]P. T. Kannan and S. K. Bansal, "Unimate: A student information system," *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 2013, pp. 1251-1256, doi: 10.1109/ICACCI.2013.6637357.

[15]D. K. Farkas, "A university Website design project: the design process, the prototype and some design issues," *Proceedings of IPCC 97. Communication*, 1997, pp. 311-319, doi: 10.1109/IPCC.1997.637059.