



# **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**

**APRIL 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. In order to accelerate and to make it an effortless process for issuing Bonafide\LOR along with checking of background details. 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. 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 valid or not.

## **TABLE OF CONTENTS**

<b>CHAPTERS</b>	<b>TITLE</b>	<b>PAGE NUMBER</b>
	<b>ABSTRACT</b>	<b>vi</b>
	<b>LIST OF TABLES</b>	<b>v</b>
	<b>LIST OF FIGURES</b>	<b>vii</b>
	<b>LIST OF SYMBOLS ABBREVIATION</b>	<b>viii</b>
<b>1.</b>	<b>INTRODUCTION</b>	
	1.1 Overview	2
	1.2 Problem Definition	2
<b>2.</b>	<b>SYSTEM ANALYSIS</b>	
	2.1 Existing System	4
	2.2 Proposed System	4
	2.3 Hardware and Software Requirement	5
<b>3.</b>	<b>SYSTEM DESIGN</b>	
	3.1 ER Diagram	7
	3.2 Data Dictionary	8
	3.3 UML Diagrams	9
<b>4.</b>	<b>SYSTEM ARCHITECTURE</b>	
	4.1Architecture Diagram	15
	4.2 Request for Bonafide	16
	4.3Request for LOR	17
	4.4 Background Verification	19
<b>5.</b>	<b>SYSTEM IMPLEMENTATION</b>	
	5.1 Client-Side Coding	21
	5.2 Server-Side Coding	22

<b>6.</b>	<b>SYSTEM TESTING</b>	
	6.1 Unit Testing	36
	6.2 Integration Testing	39
<b>7.</b>	<b>CONCLUSION</b>	
	7.1 Conclusion and Future Enhancements	44
	<b>APPENDICES</b>	
	A.1 Sample Screens	45
	A.2 Publications	49
	<b>REFERENCES</b>	51

## **LIST OF TABLES**

<b>TABLE NO</b>	<b>TABLE NAME</b>	<b>PAGE NO</b>
3.2	Data Dictionary	8
6.1	Unit Testing Table of LOR	36
6.2	Unit Testing Table of Bonafide	37
6.3	Unit Testing of Background Verification	38
6.4	Integration Testing of Bonafide\LOR	41
6.5	Integration Testing of Background Verification	42

## LIST OF FIGURES

<b>FIG NO.</b>	<b>FIGURE NAME</b>	<b>PAGE NO</b>
3.1	ER Diagram	7
3.3	Use Case Diagram	9
3.4	Activity Diagram	10
3.5	Sequence Diagram	11
3.6	Collaboration Diagram	12
3.7	Class Diagram	13
4.1	Architecture Diagram	15
A.1.1	Front Page of Automated System And Services	45
A.1.2	Registration Page Of Students/Staff	45
A.1.3	Admin Page for Authorizing Access	46
A.1.4	Login Page	46
A.1.5	Student Home Page	47
A.1.6	Bonafide Page	47
A.1.7	Lor Page	48
A.1.8	Background Verification Software	48
A.1.9	Checking Status in Console	49
A.1.10	The Verification Status	49



## **LIST OF ABBREVIATIONS**

<b>S. No</b>	<b>ABBREAVATIONS</b>	<b>EXPANSION</b>
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. The Background Verification Software is completely automated. When the software is instantiated, it automatically logs in into the gmail account to which the emails are sent for verification. The contents of the letter are parsed and a comparison is performed with the database maintained by the college. 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.

## **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.

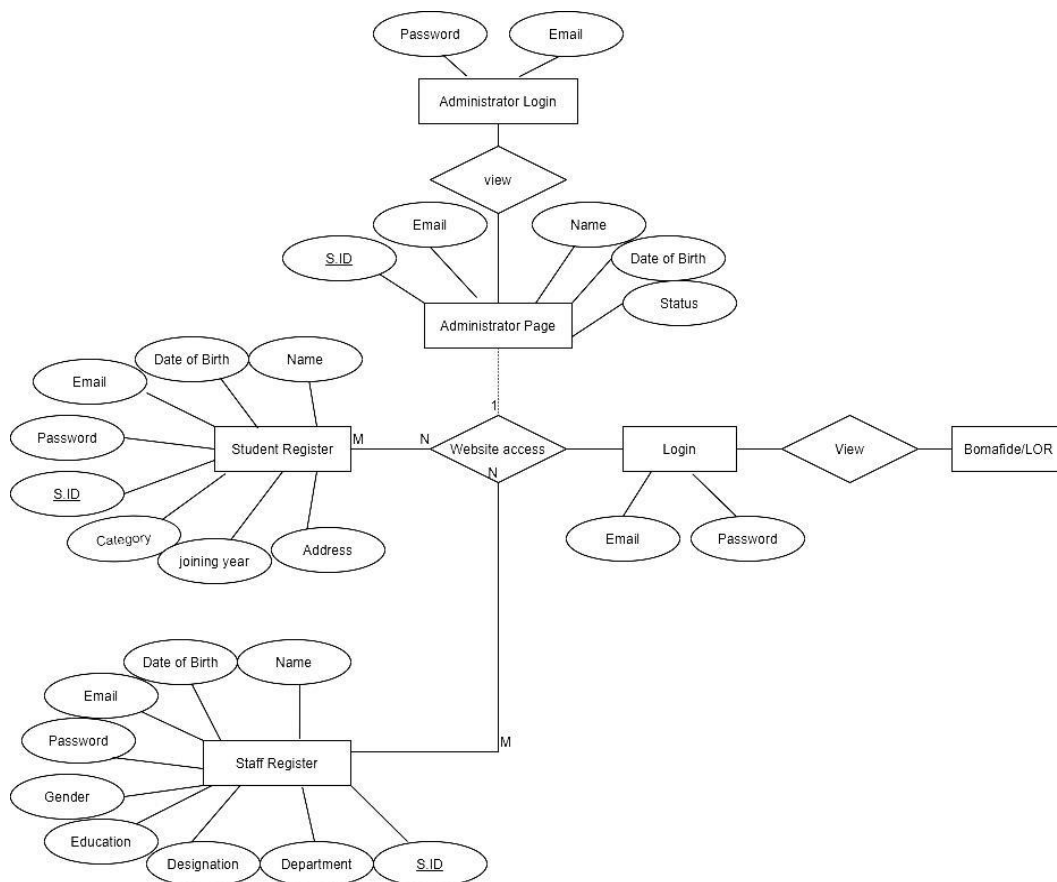


Fig:3.1 ER diagram for bonafide/lor



### 3.2 DATA DICTIONARY

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
Roll Number	Integer	10	2018PECCS100
Candidate's Name	Text	20	Lakshmi Priya
College	Text	20	Panimalar engineering college
Registration Number	Integer	10	2114568
CGPA	Numeric(p,s)	(38,0)	9.0
Year of Passing	Year	10	2020
Qualifications	Text	20	Bachelor of Engineering
Mode of Qualifications	Text	20	Full time

### 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 Bonafie 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.

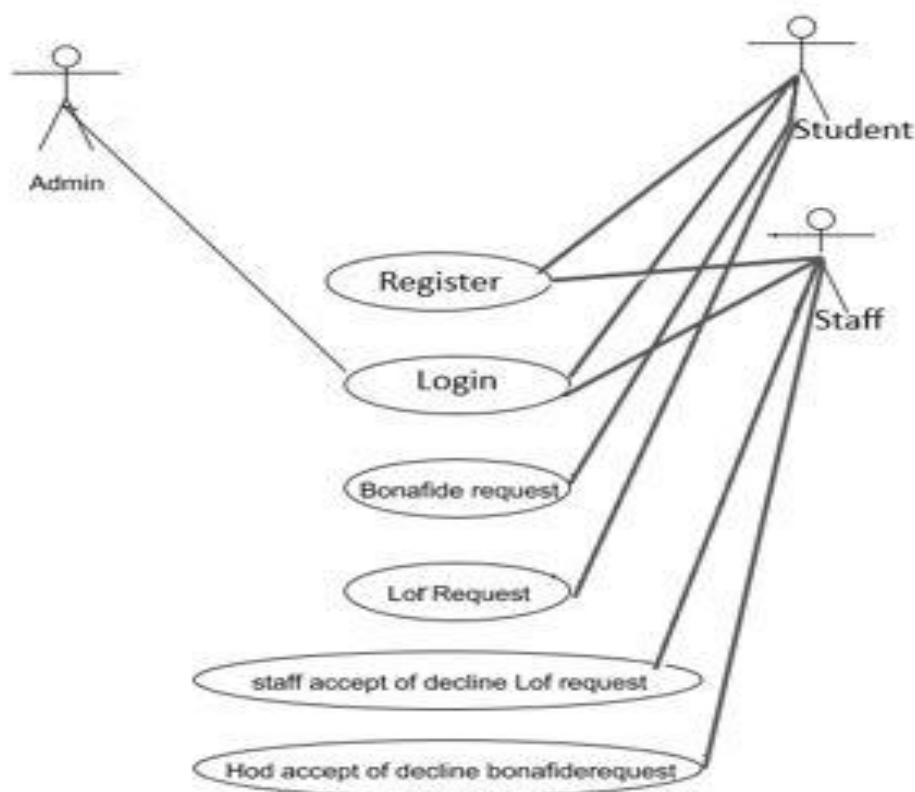


Fig:3.3 Use case diagram for bonafide/lor

### 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.

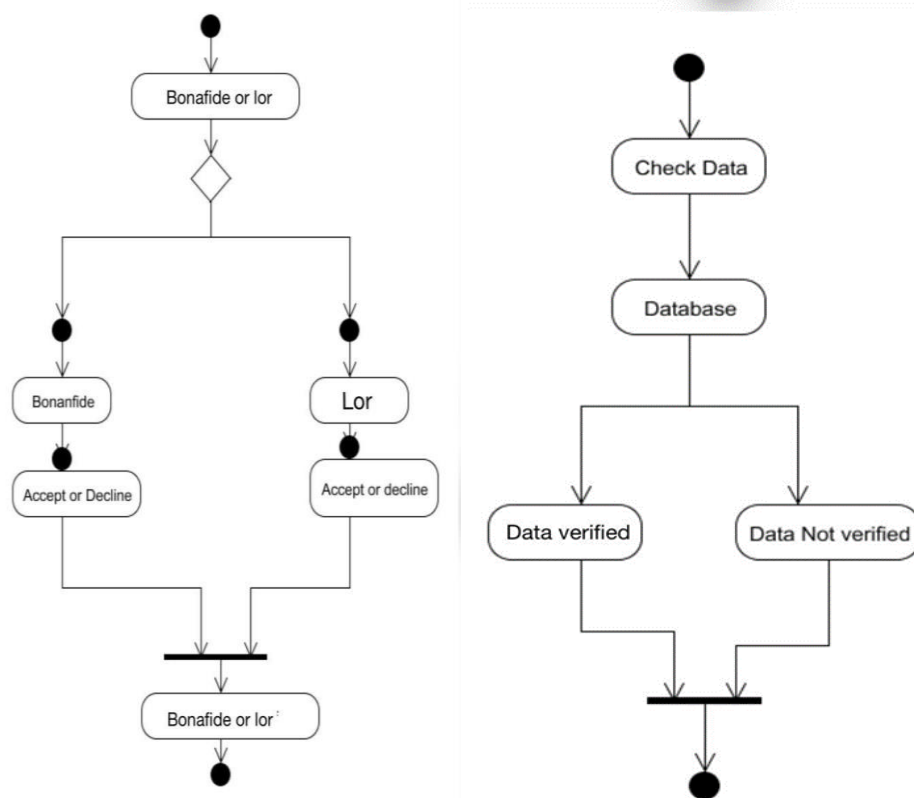


Fig:3.4 Activity diagram for bonafide/lor and student verification system

### 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.

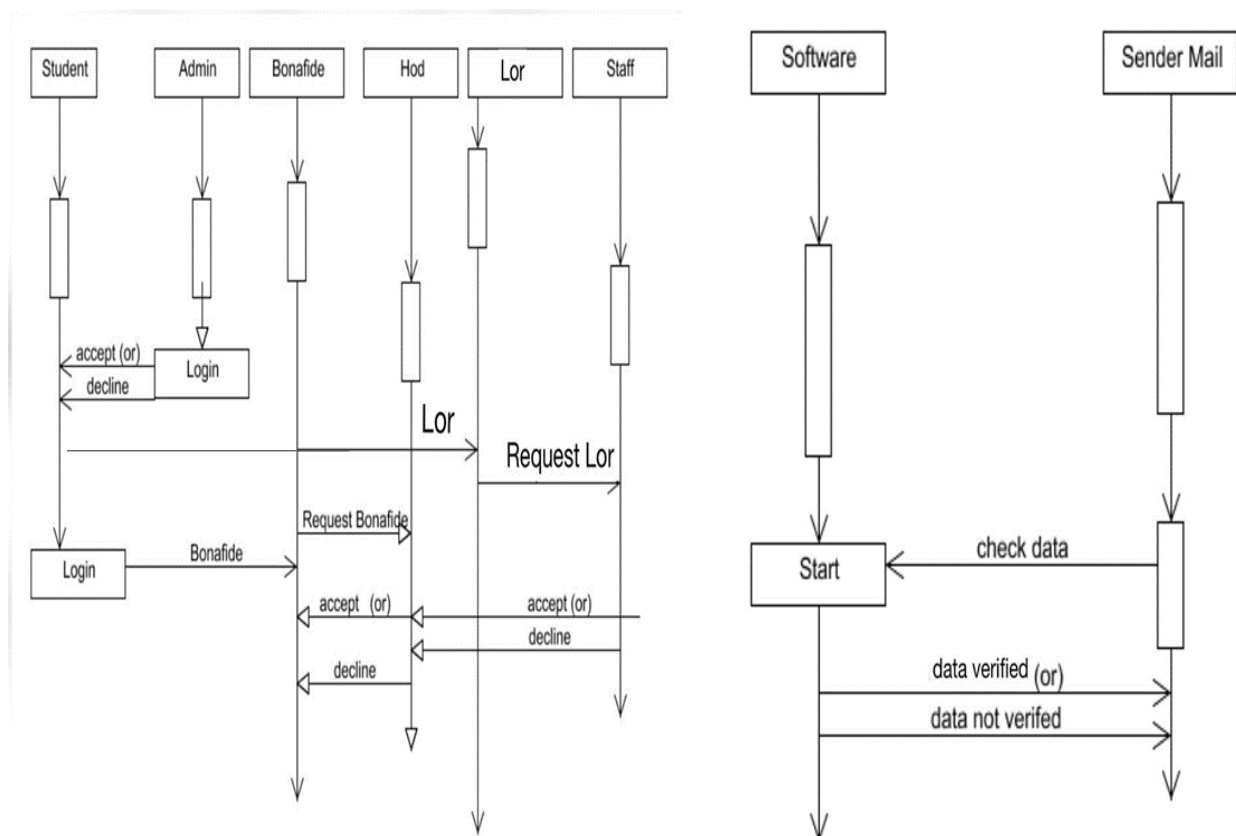


Fig:3.5 Sequence diagram for bonafide/lor and student verification system

### 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.

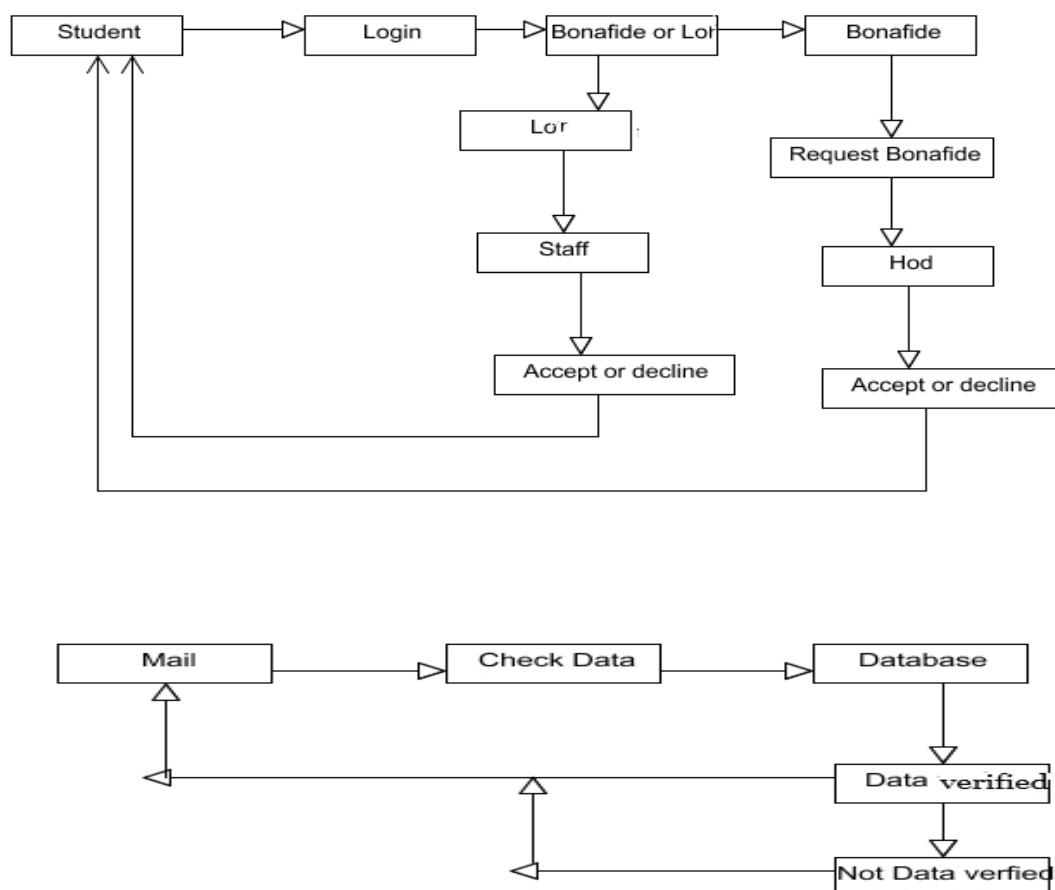


Fig:3.6 Collaboration diagram for bonafide/lor and student verification system

### 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 BonCertificate 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 doPost () and void doGet () similar attributes and methods are been performed under the Classes of Bon Request, BonStore, LORStore, LOr Request and under Checking LOR.

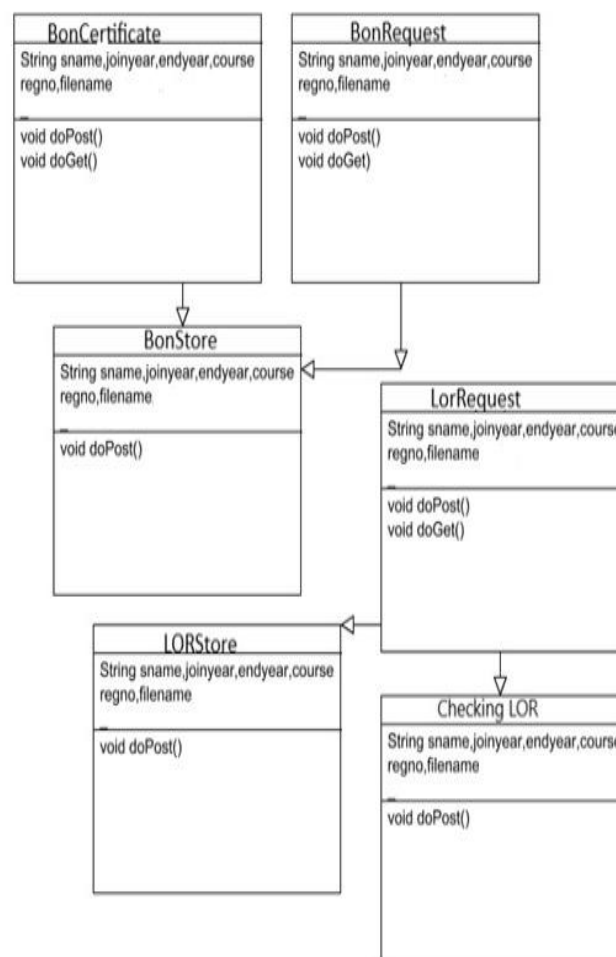


Fig:3.7 Class diagram for bonafide/lor and student verification system

## **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 updatment 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 automatically logs in into the gmail account, parses the email contents and checks with the database maintained by the college. 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.

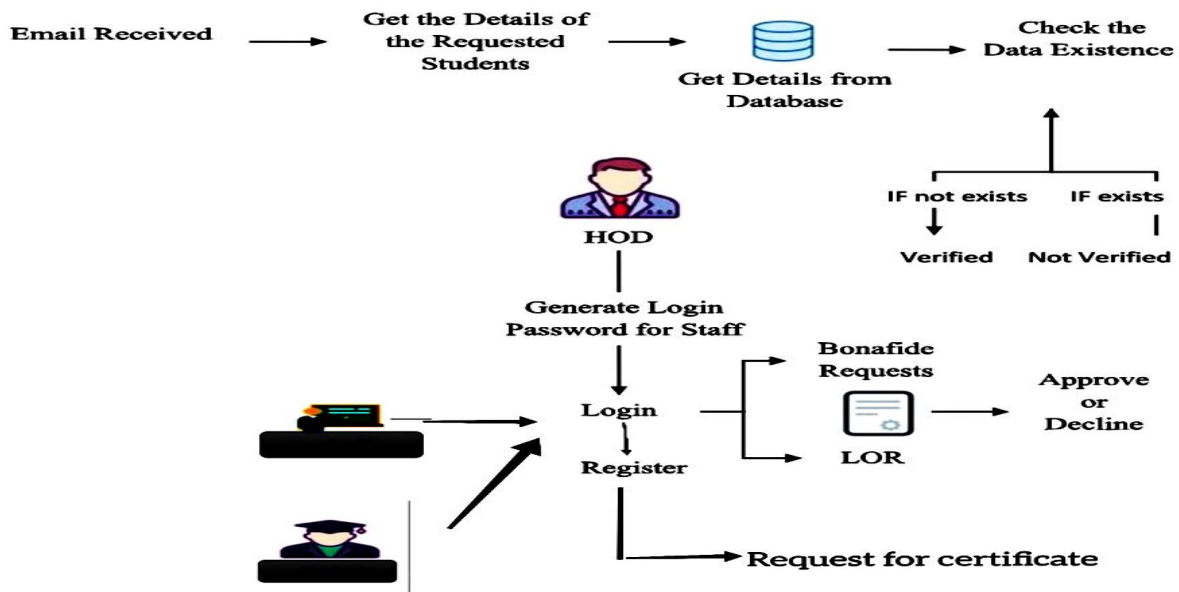


Fig:4.1 Architecture Overview of automated student verification system and services



## **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 REQUEST (LOR)**

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 .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 there details will be present (students can select there Staff proceed by clicking on request button)
- This leads us to another page Letter Of Request (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 for passed out students, the software automatically open the mail account and parse/scrape out the data such as (name, register number, marks passed out year) 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.

## **CHAPTER 5**

## CHAPTER 5

### SYSTEM IMPLEMENTATION

#### 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>
        <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">
            <div class="col-md-6 col-sm-6 col-xs-12 btm-footer-links">
                <a href="#"></a>
                <a href="#"></a>
            </div>

```

## **Background verification Client-side**

```
package com.model;

import java.io.*;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.*;
import javax.mail.*;
import javax.mail.Flags.Flag;
import javax.mail.search.FlagTerm;
import com.dao.ReadProper;

public class MailReader {
    HashMap<String, String> hm = new HashMap<String, String>();
    String email;
    String name = null;
    String yop = null;
    String cgpa = null;
    String gpa = null;
    Folder inbox;
    int i;
    boolean bIsUid = false;
    String frp1 = "";
    String frp2 = "";
    // Constructor of the class.
    public MailReader() {
        /* Set the mail properties */
        Properties props = System.getProperties();
        props.setProperty("mail.store.protocol", "imaps");
    }
}
```



```

try {
    ReadProper rp = new ReadProper();
    String[] words=s1.split("\\s");//splitting using whitespace
    for(String w:words){
        frp.add(w);
    }
    inbox = store.getFolder("Inbox");
    System.out.println("No of Unread Messages : " +
        inbox.getUnreadMessageCount());
    inbox.open(Folder.READ_WRITE);
    Message[] messages = inbox.search(new FlagTerm(new
Flags(Flag.SEEN), false));
    FetchProfile fp = new FetchProfile();
    fp.add(FetchProfile.Item.ENVELOPE);
    fp.add(FetchProfile.Item.CONTENT_INFO);
    inbox.fetch(messages, fp);
    try {
        printAllMessages(messages);
        inbox = props.SetFlag(i,bIsUid,"SEEN",1);
        inbox.close(true);
        store.close();
    } catch (Exception ex) {
        System.out.println("Exception arise at the time of read mail");
        ex.printStackTrace();
    }
} catch (NoSuchProviderException e) {
    e.printStackTrace();
    System.exit(1);
} catch (MessagingException e) {

```

```

        e.printStackTrace();
        System.exit(2);
    }
}

public void printEnvelope(Message message) throws Exception {
    Address[] a;
    String from = null, to = null;
    // FROM
    if ((a = message.getFrom()) != null)
    if ((a = message.getRecipients(Message.RecipientType.TO)) != null) {
        for (int j = 0; j < a.length; j++) {
            to = a[j].toString();
            System.out.println("TO: " + a[j].toString());
        }
    }
    String subject = message.getSubject();
    Date receivedDate = message.getReceivedDate();
    String content = message.getContent().toString();
    public void getContent(Message msg, String from, String to, String
subject) {
        int c;
        System.out.println("Message : ");
        while ((c = is.read()) != -1) {
            System.out.write
InputStreamReader isReader = new InputStreamReader(is);
            BufferedReader reader = new BufferedReader(isReader);
            StringBuffer sb = new StringBuffer();
            String str;
            while ((str = reader.readLine()) != null) {
                sb.append(str);
            }
        }
    }
}

```

```

        System.out.println("-----"+str);
        str = reader.readLine();
    }
    System.out.println(sb.toString());

    ArrayList<String> reg = new ArrayList<String>();
    String message = sb.toString();
    System.out.println("=====From=====>>>" + from);
    System.out.println("=====To=====>>>" + to);
    System.out.println("=====Subject=====>>>" + subject);
    System.out.println("=====Message=====>>>" + message)

    String[] e1 = from.split("<");
    email = e1[1].replace(">", "");
    System.out.println("=====email=====sender ===>>>" + email)
    String findex = "";
    String sindex = "";
    String tindex = "";

    ArrayList<String> b1 = new ArrayList<String>();
    try{
        if(msg !=null)
        {
            String [] c = msg.split("\\s");
            for(String aa1 : c)
            {
                b1.add(aa1);
            }
        }
        findex = b1.get(0);
        sindex = b1.get(1);
        tindex = b1.get(2);
        System.out.println("findex-----"+findex);
    }

```

```

if(findex.contains("name")||findex.contains("Name"))
{
    try{
        Class.forName("com.mysql.jdbc.Driver");
        Connection con=DriverManager.getConnection(
            "jdbc:mysql://localhost:3306/autosystem","root","root");

        Statement stmt=con.createStatement();
        ResultSet rs=stmt.executeQuery("select * from student   where
            name='"+sindex+"' and regno='"+fourindex+"'");
        if(rs.next())
        {
            System.out.println("Data Found");
            System.out.println("Message Sent to Mail");
            String sub = "Verification Status";
            String text = "Data Verified";
            Mail sme = new Mail(empuserid, email , sub, text, empuserid,
                                empmailpassword);

            else
            {
                System.out.println("Data Found Not");
                System.out.println("Message Sent to Mail-----");
                String sub = "Verification Status";
                String text = "Data Not Verified";
                Mail sme = new Mail(empuserid, email , sub, text, empuserid,
                                    empmailpassword);
            }
        }
    }catch(Exception e){ System.out.println(e);}
}
catch(Exception e)

```

```
{  
    System.out.println("Exeeption e ---"+e);  
}  
  
public static void main(String args[]) {  
    new MailReader();  
}  
}
```

## 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 eyyear = request.getParameter("eyyear");
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+"--"+eyyear+"--"+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 rand = new Random();

//Generate random integers in range 0 to 999
int r = rand.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, eyyear);
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, eyyear);
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

```
package com.controller;

import java.io.IOException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.model.Scheduler;

public class StartServ extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse response)

        throws ServletException, IOException {

        System.out.println("START SER");

        boolean a = true;

        Scheduler sch = new Scheduler();

        sch.waitMethod(a);

        public void doPost

            throws ServletException, IOException {

            response.setContentType("text/html");

            PrintWriter out = response.getWriter();

            out.println("<HTML>");

            out.println(" <HEAD><TITLE>A Servlet</TITLE></HEAD>");

            out.println(" <BODY>");

            out.print("  This is ");

            out.print(this.getClass());

            out.println(", using the POST method");

            out.println(" </BODY>");

            out.println("</HTML>");

            out.flush();

            out.close();}}}
```

## **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

S No:	Test case Objective	Test Case Description	Expected Result
1	Check the interface link between Register and Login Module.	Enter the Register details and able to Login via details for next usage.	To be directed to Login Page of Student / staff.
2	Check the interface Link between Admin Login and View List .	Enter admin Login credentials and view pending List.	The Admin will be able to approve/decline the students and staffs.
3	Check the interface link between Student Login and student home page	In the student home page two choice seen Bonafide and LOR.	The Students directed to specific page of Bonafide or LOR.
4	Check the interface link between student home page and student LOR	Staff list of the department is and click on Register button.	Request for LOR is processed to that staff.
5	Check the interface link between Staff Login and staff home.	Staff Home page carries the Reason of students and click on.	Letter will be displayed.

Firstly, is the unit testing of LOR represented in tabular format with test case objectives and test case description with the expected results followed

Table 6.2 Unit Testing For Bonafide

S.No:	Test case Objective	Test case Description	Expected Result
1	Check the interface link between Register and Login Module.	Enter the Register details and able to Login via details for next usage.	To be directed to Login Page of Student/staff.
2	Check the interface link between Admin Login and View List	Enter admin Login credentials and view pending List.	The Administrator approve/decline the students and staffs of the college.
3	Check the interface link between Student Login and student home page	In the student home page two choices displayed Bonafide LOR.	The Student directed to specific page of Bonafide or LOR.
4	Check the interface link between student home page and Bonafide	Student home page consist of reasons to select from pre-defined list.	The request for Bonafide is submitted successfully is displayed.
5	Check the interface link between HOD login and student Request List page	Student Request List is displayed and click under status of approval or decline.	On approval “ Request for Bonafide is Successfully processed” message is displayed on HOD page.

- 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

S No:	Test case Objective	Test Case Description	Output
1	Check the interface link between background verification software and mail.	Is the background verification software parsing the data.	Reply mail is sent from received mail id.

- 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 parses the data accurately from the gmail contents, if the data is been accurate in table format, then reply mail is sent based on verifying with the database.
- If the data is not found after checking in the database, then a reply mail is sent back to user regarding the student is not authenticated and verified.
- If the data is not present in Tabular format then data is not been parsed and respond message is sent back to the user asking to reply in tabular format .

## 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

S No:	Test Scenario	Test Steps	Test Data	Expected 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.	Name: Sam DOB :2/14/1998 Gender: Male Education: B. tech Email:sam@gmail.com Password: Designation:M.tech Department:CSE.	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.	<u>Userid=</u> <u>sam@gmail.com</u> Password: sam123	User should Login into the application	Pass
3	Check User providing Bonafide with signature	1.Enter user ID. 2.Enter password. 3.Click approve.	Choosing File: Choose user signature	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 format to see if the software parses the data exactly and check with the correct fields in the database and returns a positive result to the sender's email or not.
- In the same way, several other emails are sent with invalid data to check if the software is able to parse out the contents properly and find if a match is found or not.

Table 6.5 Integration Testing for Background Verification System

S No.	Test Scenario	Test Steps	Test Data	Expected Results	Actual Results	Pass\Fail
1	Check User provides with valid Data	1.The mail is sent with the content.	Candidate Name: Ram.V Roll no:211417104001 Passing Year:Apr,2007 Course: B.E Department:CSE Backlog status:0	Data is been parsed with the Background verification software and mail and reply mail is sent	explicitly	Pass
2	Check User provides with invalid Data	1.The mail is sent without any content.	Null	As No data is present in the mail ,no process is been performed.	explicitly	Fail

## **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. With regards to background verification, the proposed software can parse the contents which are only in table format. 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

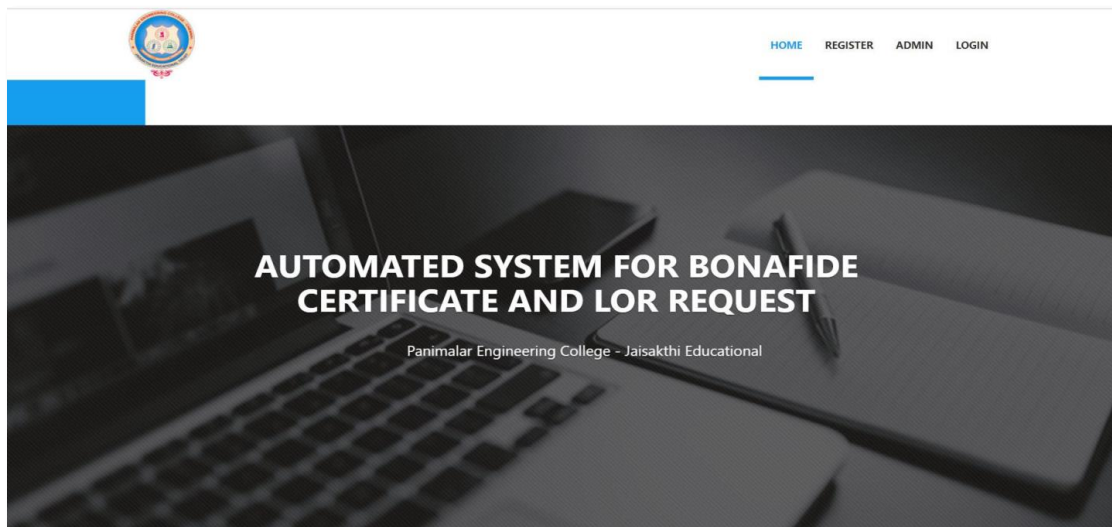


Fig: A.1.1-Front Page of Automated System For Bonafide Certificate and LOR

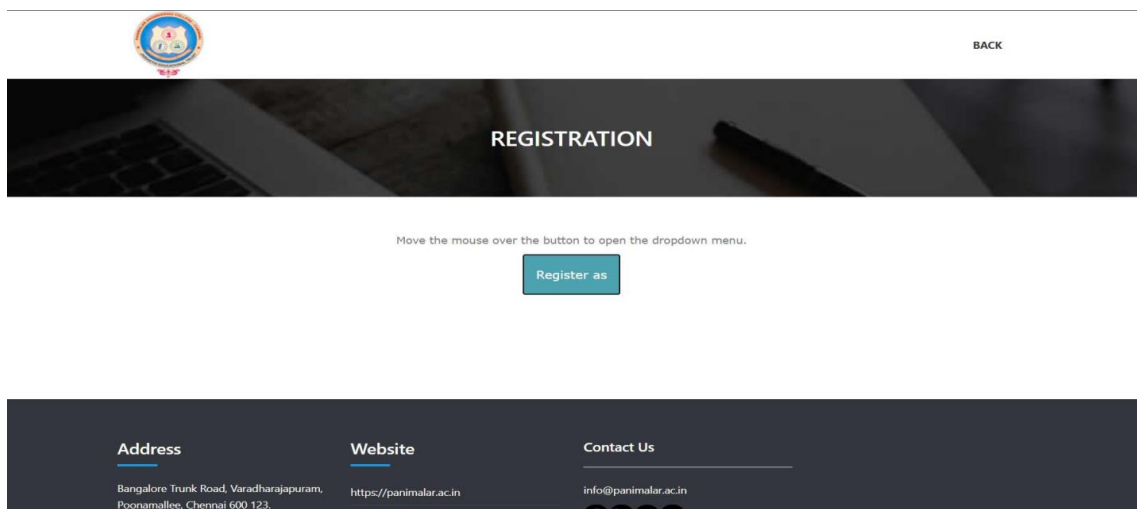



Fig:A.1.2 -Registration Page for the Students and Staffs




[HOME](#)   [ADMIN HOME](#)   [BACK](#)   [LOG OUT](#)

## STUDENT LIST

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



[BACK](#)

## LOGIN

Move the mouse over the button to open the dropdown menu.

Login as

HOD  
Staff  
Student

**Address**

Bangalore Trunk Road, Varadharajapuram,  
Poonamallee, Chennai 600 123.

**Website**

<https://panimalar.ac.in>

**Contact Us**

[info@panimalar.ac.in](mailto:info@panimalar.ac.in)

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

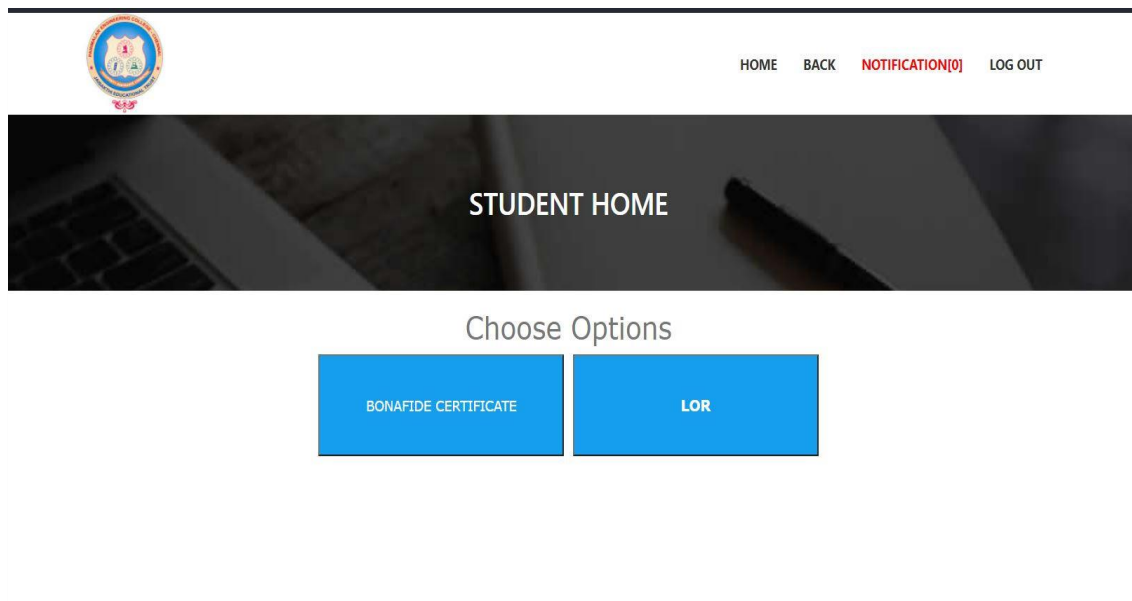


Fig: A.1.5 -Student Home Page After Logging in

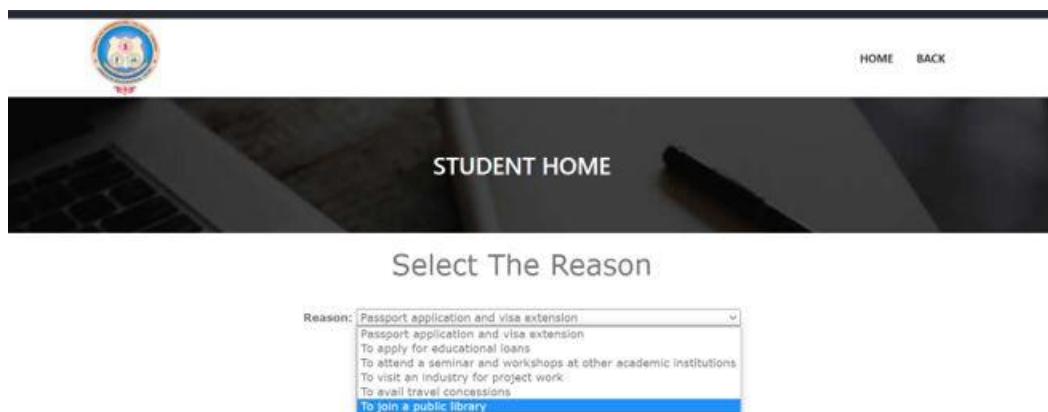


Fig: A.1.6 -Bonafide page for selecting reasons





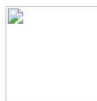
[HOME](#) [BACK](#)

## STUDENT HOME

### Letter of Request

WRITE YOUR MESSAGE

Your Signature



No file chosen

Fig: A.1.7 -LOR Page for drafting letter

Background Verification Software

[Start](#)

[Stop](#)

VERIFICATION

# Background Verification

Fig: A.1.8 -Front page of Background Verification Software

```
Commons Daemon Service Runner
Mar 07, 2021 2:31:24 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 2066 ms
START SER
a-----true
always running program ==> Sun Mar 07 14:31:44 IST 2021
vbackground0@gmail.com
Backgroundverification123
frp1-----vbackground0@gmail.com-----Backgroundverification123
No of Unread Messages : 1
FROM: Sherline Calista <:ssherlinecalista99@gmail.com>
TO: vbackground0@gmail.com
Subject : Education Verification_Sam
-----name sam reg 8490 dob 2021-03-24 join 2021-03-09 end
=====From=====>>>Sherline Calista <:ssherlinecalista99@gmail.com>
=====To=====>>>vbackground0@gmail.com
=====Subject=====>>>Education Verification_Sam
=====Message=====>>>name sam reg 8490 dob 2021-03-24 join 2021-03-09 end 2025-03-08 c 6.5
msg-----name sam reg 8490 dob 2021-03-24 join 2021-03-09 end 2025-03-08 c 6.5
=====email=====sender ==>>>ssherlinecalista99@gmail.com
findex-----name
Data Found
Message Sent to Mail-----
22 8490 2021-03-24
Prop frp1 -----vbackground0@gmail.com-----frp2----Backgroundverification123
```

Fig:A.1.9 -Checking Status in Console

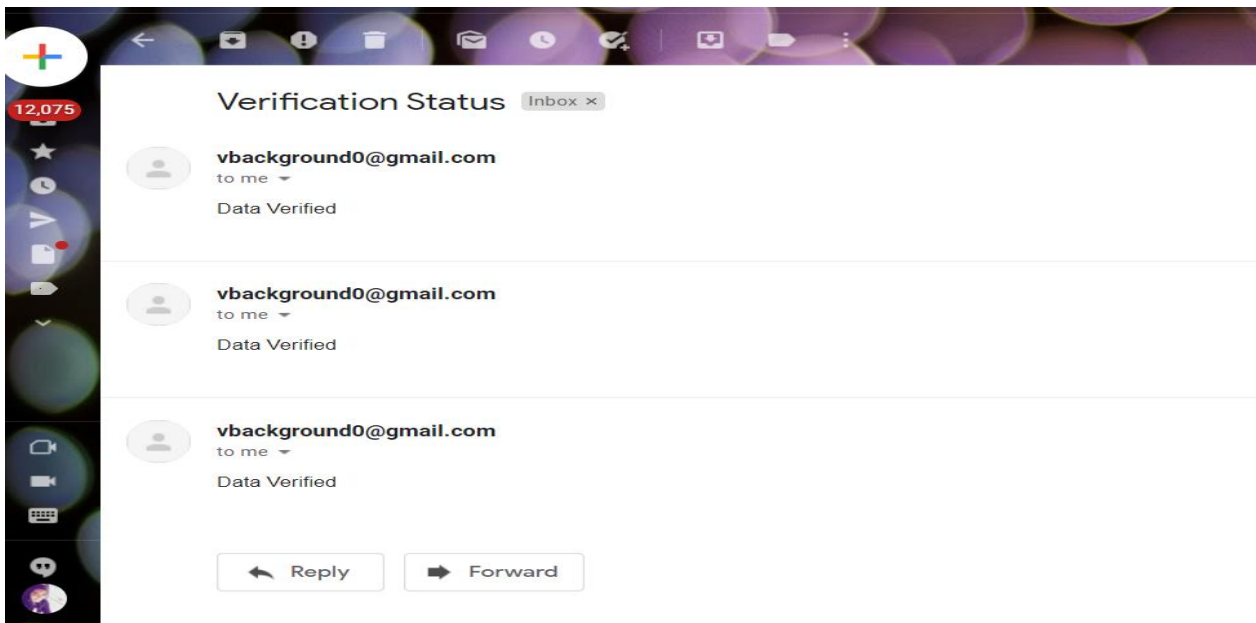


Fig: A.1.10 -The Verification Status

## **PUBLICATIONS**

## 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/ICIIM.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: 10.1109/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.