A Project Report

on

PROFICIENT COMMUNICATOR

Submitted for partial fulfillment of the requirements for the award of the degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

BY

MOHD ABDUL RAHEEM 160316737056

Under the guidance of

Mr. FEROZ AMER

Assistant Professor

Dept. of Information Technology

DCET, Hyderabad.

Department of Information Technology

Deccan College of Engineering and Technology

(Affiliated to Osmania University) Hyderabad

C E R T I F I C A T E

This is to certify that the project work entitled **“PROFICIENT COMMUNICATOR**” is a bonafide work carried out by **MOHD ABDUL RAHEEM (160316737056)** in partial fulfillment of the requirement for the award of Degree of **BACHELOR OF ENGINEERING IN INFORMATION TECHNOLOGY** by the **OSMANIA UNIVERSITY**, Hyderabad, under our guidance and supervision.

The results embodied in this report have not being submitted to any other university or institute for the award of any degree or diploma.

Internal Guide	Project Coordinator	Head of the Department
Mr. FEROZ AMER	**Mr. FEROZ AMER**	**DR. AYESHA AMEEN**
Assistant Professor	Assistant Professor	Professor & HOD
Dept. of I. T	Dept. of I. T	Dept. of I. T
DCET, Hyderabad	DCET, Hyderabad	DCET, Hyderabad

DECLARATION

This is to certify that the work reported in the present project entitled **“PROFICIENT COMMUNICATOR” is** a record of work done by me in the Department of Information Technology, Deccan College of Engineering and Technology, Osmania University. The reports are based on the project work done entirely by me and not copied from any other source.

Mohd Abdul Raheem - 160316737056

ACKNOWLEDGEMENT

I would like to express my sincere gratitude and indebtedness to my project supervisor **Mr. FEROZ AMER for** his valuable suggestions and interest throughout the course of this project.

I am also thankful to **DR. AYESHA AMEEN**, H.O.D, IT Department for providing excellent infrastructure and a nice atmosphere for completing this project successfully.
I convey my heartfelt thanks to the lab staff for allowing me to use the required equipment whenever needed.

Finally, I would like to take this opportunity to thank my family for their support through the work. I sincerely acknowledge and thank all those who gave directly or indirectly their support in completion of this work

Mohd Abdul Raheem - 160316737056

TABLE OF CONTENTS

ABSTRACT

INTRODUCTION

Communication is an important aspect of our day-to-day life; we communicate using different techniques. These techniques involve verbal communication, nonverbal communication, body language and many more. The basic communication that we use is verbal communication i.e., communicating using different kinds of languages respective to different regions of the globe. The other is nonverbal communication. Nonverbal communication includes Eye contact, Facial expressions, gestures, posture and body orientation, body language, space, and distance proximities. Now one might think how two individuals can communicate non-verbally if they are not visible to each other? This is a valid question, it was impossible to communicate without being visible to one other in the late 1970's, but now with the evolved technologies like internet, networking, smartphones and many other devices revolutionized the field of communication. These new technologies made possible of making connection between two or more individuals. That's where Proficient Communicator comes in picture, Proficient Communicator allows a user to communicate with each other in large groups.

EXISTING SYSTEM

There are many applications that allow users to send text messages to each other and as well as groups Basically existing applications like WhatsApp, Facebook, Instagram, hike, etc. allows users to create groups among themselves and communicate in groups as well as communicate in a separate personal chat with one another.

DRAWBACKS OF EXISTING SYSTEM

The existing system is preferable in a personal consent but not in the professional organizations Such as Offices, Colleges. etc Here the communication is necessary, but the existing system provides the communication at the cost of privacy breaching. Example WhatsApp users are required to lose their most sensitive information i.e., their mobile number This may result in unwanted messages beyond the organization timings and spamming as well.

PROPOSED SYSTEM

The interface of the application is user-friendly and easy to understand. Apart from providing the basic features of existing applications there are also some key boundaries that allow only admins to operate each interaction under the platform.

SOFTWARE REQUIREMENTS

- ✓ Operating system: -Windows XP or Above.
- ✓ Any Browser: - Google Chrome, UC Browser…etc.
- ✓ Any device with access to internet.

HARDWARE REQUIREMENTS

1. Computer 386 and upward for better Efficient performance
2. Power Required 63. watt
3. CPU 80386 and upward
4. CPU clock speed processor Compatible (up to 66 MHZ)
5. Memory size: 1gb on system board
6. Mass storage Device: 50gb
7. Keyboard Up to 105keys

TECHNOLOGIES USED

- Hypertext Markup Language (HTML)
- Cascading Style Sheets (CSS)
- Materialize UI
- Cloud Storage
- Cloud Server
- Java Script
- Node.js
- Socket.io
- Bootstrap

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1 INTRODUCTION

Communication is an important aspect of our day-to-day life; we communicate using different techniques. These techniques involve verbal communication, non verbal communication, body language and many more. The basic communication that we use is verbal communication i.e. communicating using different kinds of languages respective to different regions of the globe. The other is nonverbal communication. Non verbal communication includes Eye contact, Facial expression, gestures, posture and body orientation, body language, space and distance proximity.etc. Now one might think how two individuals can communicate non-verbally if they are not visible to each other? This is a valid question, it was impossible to communicate without being visible to one other in the late 1970's, but now with the evolved technologies like internet, networking, smartphones and many other devices revolutionarized the field of communication. These new technologies made possible of making connection between two or more individuals. That's where Proficient Communicator comes in picture; Proficient Communicator allows a user to communicate with each other in large groups.

1.2 PROBLEM DEFINITION

There are many application that allow users to send text messages to each other and as well as in groups .Basically existing applications like WhatsApp, Telegram, Signal etc allows users to create groups among themselves and communicate in groups as well as communicate in separate personal chat with one another, but the problem of these current chatting applications start with the breach in privacy and is not suitable for the professional organization. For example if we take the application "WhatsApp " into consideration which is being used in many professional organizations .It exposes one individual's most private information i.e. their phone number. Apart from that there is much more privacy breach from the application itself which collects user's personal data and target's the users with personalized advertisements. This breach of privacy accusation holds much more validation due to the recent privacy policy change of WhatsApp that lead to a huge digital migration. Moreover due to the personal chat feature it enables the possibilities of spam messages and calls on the personal number of the user which defeats the privacy of an individual. As all the current chatting applications are designed to facilitate one’s personal life but aren’t suitable for professional life. Due to this breach of personal life we often have observed employees and even students maintain two phone numbers. This merely solves the problem by creating a much more hassle. One individual have to maintain two different accounts on the application each for personal and professional respectively. If we consider this as an alternative then this raises another issue i.e. one application like “WhatsApp” doesn’t allow users to have two accounts at once. This situation can be defined as solving a problem with more problems.

1.3 OBJECTIVES

- The main objective of the project is to provide a safe and controlled environment for professional organisations.
- This project fulfills the need of an intelligent system that can ensure both efficiency and effectiveness.
- It can easily differentiate between personal and professional lives of one individual.
- It secures the privacy of its users.
- The whole environment is under control of an Administrator with respect to the organization.
- As the information of one user isn’t exposed to the other thus this would prevent spamming.

1.4 MOTIVATION

The main motive of the project is to secure the privacy of the users on the platform. Hence there is need of an intelligent system that can ensure efficiency, effectiveness and security. There is privacy breach in the current system this project is intended to produce a system that not only provides a safe and controlled environment but also ensures the need of the users. As the proposed system is developed specifically for professional organizations this would serve much more ease to its users rather than those present system which are designed for personal use. In this the users are enrolled by the administrator which would prevent the intruders on the system. Proficient communicator using cloud computing facilitates an individual needs of its users who wants a differentiation between their personal and professional lives. This system would eliminate the possibility of harassments which is becoming an issue as we move forward to the future.

1.5 PROPOSED SYSTEM

The Proficient Communicator is web-application that is designed to fulfill the needs of its user's privacy. However, there is no existing system that deals with issues of privacy. It provides a safe and controlled environment for all its users. In the proposed system the administrator has the full authorization of maintaining the whole platform. The users are allowed to contact each other only if the administrator permits. This would avoid unnecessary involvements of the users with the other users. The proposed system has a feature called "Notice Board" which allows the administrator and other permitted users to tag important information which would be available to all the users. As the users aren't enrolled through sign up option this eliminates the possibility of intruders on the platform.

1.6 TECHNICAL APPROACH

REST API, also known as Representational state transfer application programming interface, is one of the most powerful web application modules that provides all the necessary resources from one place to all its users. REST API provides security to the web application by client-server architecture. In this architecture the client’s requests are handled at the server side rather than processing the information on the client’s side. On the client’s side the requests are taken and are then processed on the server side this ensures the integrity of the platform. REST’s client-server separation of concerns simplifies the component implementation, reduces the complexity of connector semantics, improve the effectiveness of performance tuning, and increases the scalability of pure server components. A web browser, for example, may be the client and an application running on a computer hosting a website may be the server. The client submits an HTTP request message to the server. The server, which provides resources such as HTML files and other content, or performs other functions on behalf of the client, returns a response message to the client. The response contains completion status information about the request and may also contain requested content in its message body.

1.7 APPLICATIONS

- It can be used in any professional organizations.
- It provides the security and privacy to an individual user’s information.
- It provides a safe and controlled environment for all the users.
- An individual can differentiate its personal and professional lives.
- It maintains key boundaries that allow only admin to operate each and every interaction under the platform.
- The user’s personal information isn’t exposed to other users.
- The users can’t interact with each other and are anonymous to each other which provides spamming
- Notice board feature eliminates the use of mail which is used in the existing system for distribution of important files.
- As the user is enrolled manually by the administrator the platform is free from the intruders.

1.8 ORGANISATION OF THESIS

The study shows how realistic the system can be and how it could help the system. The thesis has been divided into five chapters which are included as follows

Chapter 1

This chapter describes the general background, statement of the problem, objective of the study, the motivation that led to the study, significance of the system, technology utilized along with its applications. It introduces and motivates the need for the system.

Chapter 2

This chapter reviews what already has been written in the field on the topic of the research. It serves as the foundation on which topic is built. It explains about the previous studies of the topic and specifies what has been known and identifies the unknown about the topics. There are several studies that has been previously done on the topic which are briefly explained in the chapter.

Chapter 3

In this chapter, the software requirements for the system have been specified. It includes hardware requirements, and the system is analyzed and discussed thoroughly.

Chapter 4

This chapter provides the design and the architecture of the system which includes several useful diagrams such as Use case, architecture which helps to understand how the system works. It helps us in clearly understanding the proposed system.

Chapter 5

Here in this chapter, we discuss the efficient algorithm that has been used in this system and it also consists of the implementation techniques along with the sample code segments and the collection of datasets.

Chapter 6

This chapter provides us with the results of the system which consists of the system testing, test cases with vivid screenshots and finally the snapshots of the results. Here, we present the success of the system that we have proposed.

Chapter 7

This chapter gives us the summary of the system as well as discusses its potential in the future. It suggests the ideas for future related work and how it can be enhanced as per the user’s needs, and here we have drawn conclusions from the proposed system.

CHAPTER II

LITERATURE SURVEY

Case 1: Viber

Viber is an instant messaging and Voice over IP (VoIP) application for smartphones developed by Viber Media. In addition to instant messaging, users can exchange images, video and audio media messages. Viber recently supported the end-to-end encryption to their service, but only for one-to-one and group conversations in which all participants are using the latest Viber version 6.0 for Android, iOS or Windows 10. At this time, in the Viber iOS application for iPhone and iPad, attachments such as images and videos which are sent via the iOS Share Extension does not support end-to-end encryption. Viber has privacy issues such as adding a friend without his knowledge or adding him to a group without his permission. Plus that, local storage is not secured. It is not open source making it difficult to evaluation.

Case 2: Telegram

Telegram is an open source instant messaging service enables users to send messages, photos, videos, stickers and files. Telegram provides two modes of messaging are regular chat and secret chat. Regular chat is client- server based on cloud-based messaging; it does not provide end-to-end encryption, stores all messages on its servers and synchronizes with all user devices. More, local storage is not encrypted by default. Secret chat is client-client provides end-to-end encryption. Contrary to regular chat messages, messages that are sent in a secret chat can only be accessed on the device that has been initiated a secret chat and the device that has been accepted a secret chat they cannot be accessed on other devices. Messages sent within secret chats can be deleted at any time and can optionally self-destruct.

Telegram uses its own cryptographic protocol MTProto, and has been criticized by a significant part of the cryptographic community about its security.

The registration process of Telegram, Viber and WhatsApp depend on SMS. SMS is transported via Signaling System 7 (SS7) protocol. The vulnerability lies in SS7. Attackers exploited SS7 protocol to login into victim's account by intercepting SMS messages. Because of Telegram cloud-based, the attacker exploits it and makes full control of the victim account and can prevent him to enter into his account. To make the account more secure should activate two-factor authentication.

Case 3: WhatsApp

WhatsApp is one of the most popular messaging applications, recently enabled end-to-end encryption for its 1 billion users across all platforms. WhatsApp uses part of a security protocol developed by Open Whisper System, so provides a security-verification code that can share with a contact to ensure that the conversation is encrypted. It is difficult to trust in WhatsApp application completely because the application is not open source, making it difficult to verify the functioning process and match them with the work of the encryption protocol which was announced.

CHAPTER III

PROBLEM SPECIFICATION

3.1 SYSTEM REQUIREMENT SPECIFICATION

HARDWARE REQUIREMENTS

- Computer 386 and upward for better Efficient performance
- Power Required 63. watt
- CPU 80386 and upward
- CPU clock speed processor Compatible(up to 66 MHZ)
- Memory size: 1GB on system board
- Mass storage Device: 50gb
- Keyboard Up to 105keys
- Any device with access to internet

SOFTWARE REQUIREMENTS

- Operating system:-Windows XP or Above.
- Any Browser: - Google Chrome, UC Browser…etc.

3.2 SYSTEM ANALYSIS AND DISCUSSION

A. **Cloud Server**. Cloud Server is basically a massive powerful computer that is working 24/7 and is responsible to all the requests that are sent to our server. Here our website server is hosted and made sure that the server is up and running 24/7. There are different companies that provide the server facilities like Amazon's AWS, Digital Ocean, Microsoft's Azure and many more.

B. **Client Machine**. Client machine could be any personal device like mobile phone, laptop, desktop or smart TV. As this is a web application not an application designed for a certain operating system like android or iOS. Any device could act like a Client Machine as long as it has access to a web browser and Internet.

C. **Cost Analysis**, the cost of maintaining the proficient communicator would be the Cloud Server cost that a company charges for making the server active on the internet. They have a reasonable pricing for different services they provide for around 10$ per month, but even that depends on the usage of the whole web application.

D. **Efficiency.** Talking about the efficiency of the Proficient communicator as it uses the latest technologies in the market, it's highly efficient as well as highly secure.

E. **Reliability Analysis** of Proficient Communicator based on different issues such as cost, speed, flexibility and many other stuff, it is important to determine whether a web application is reliable or not. Proficient communicator is reliable because its highly scalable. The response of the application is dynamic. As the number of users grows the system is adaptable to handle such changes.

CHAPTER IV

SYSTEM DESIGN

4.1 USE CASE DIAGRAM

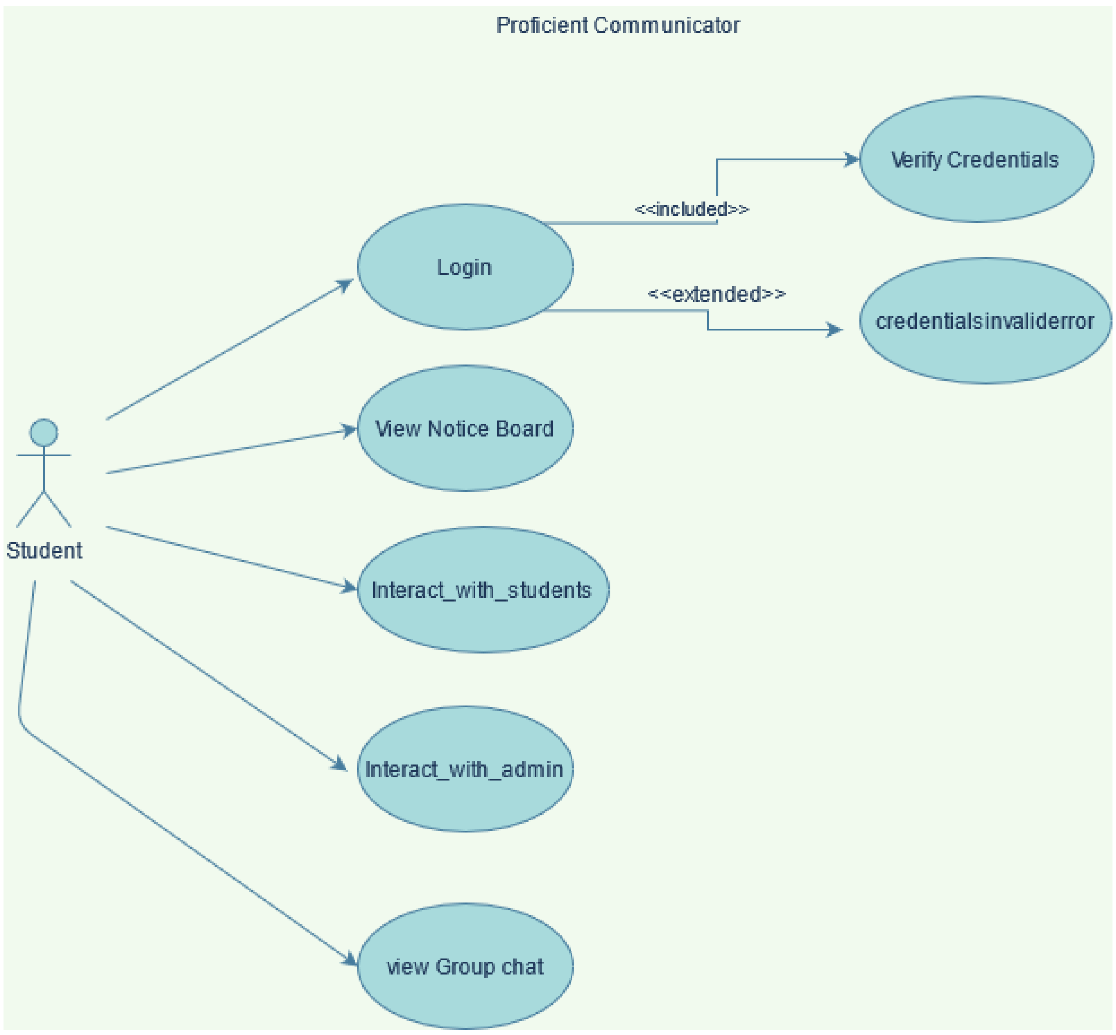


Figure 4.1: Use case diagram of student.

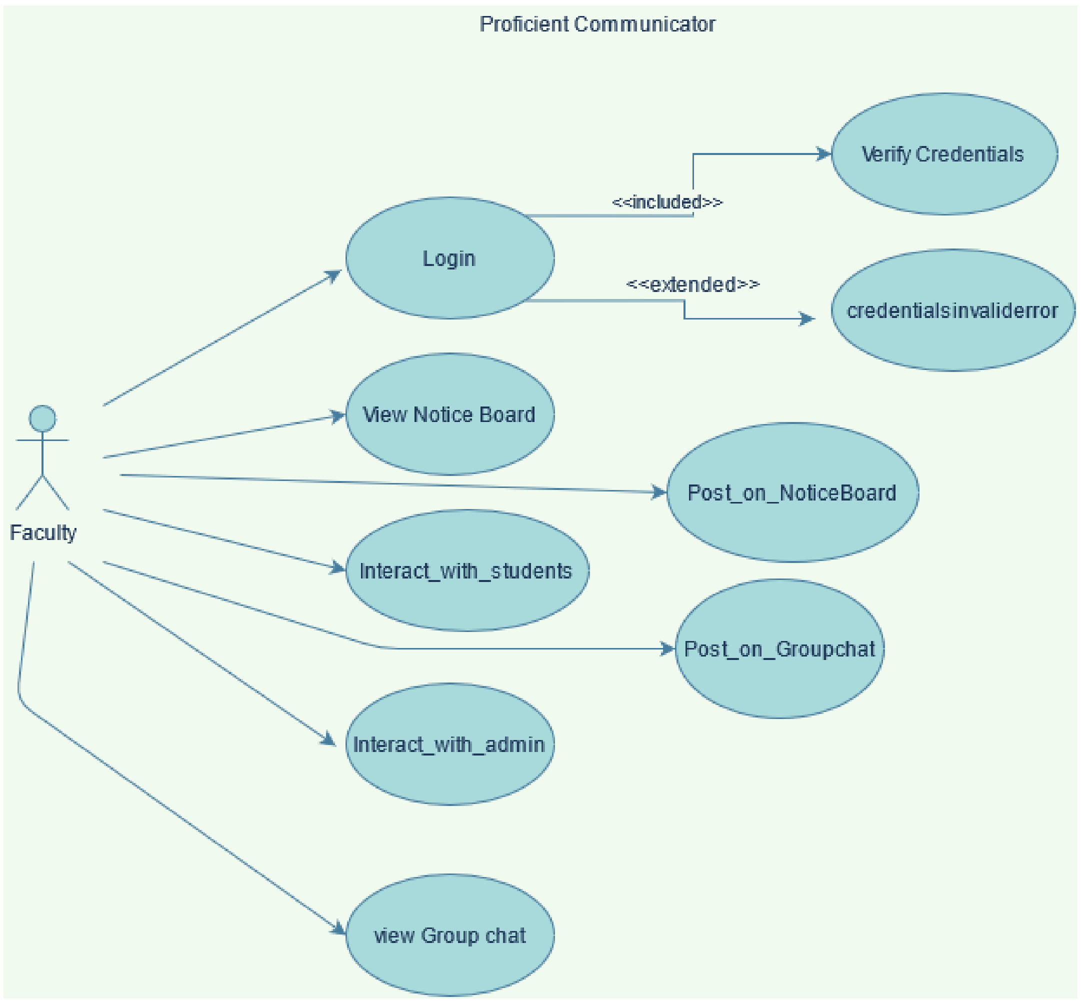


Figure 4.2: Use case diagram of Teacher.

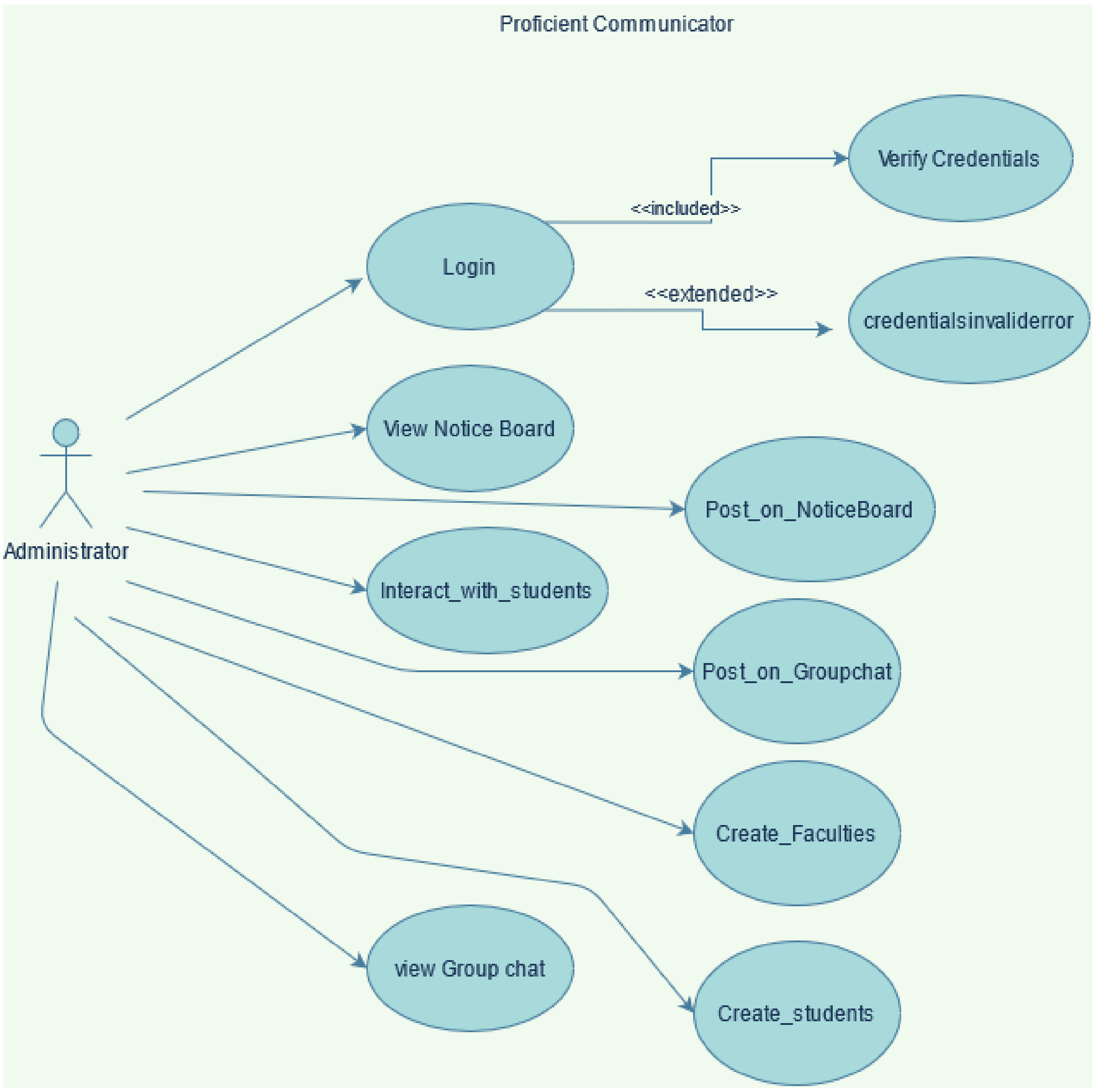


Figure 4.3: Use case diagram of Administrator.

EXPLANATION:

All these Figures 4.1,4.2,4.3 elaborate the privileges of each individual respective to the system. There are three types of users students, faculty and administrator with their own privileges. As we can see a student can login and view the group chat and notice board but the faculty can view and also post on those two sections. Similarly, a student can initiate chat with the faculty, but the faculty can't initiate the chat a faculty can only chat with those students who initiated the chat not the other way around

4.2 SYSTEM ARCHITECTURE

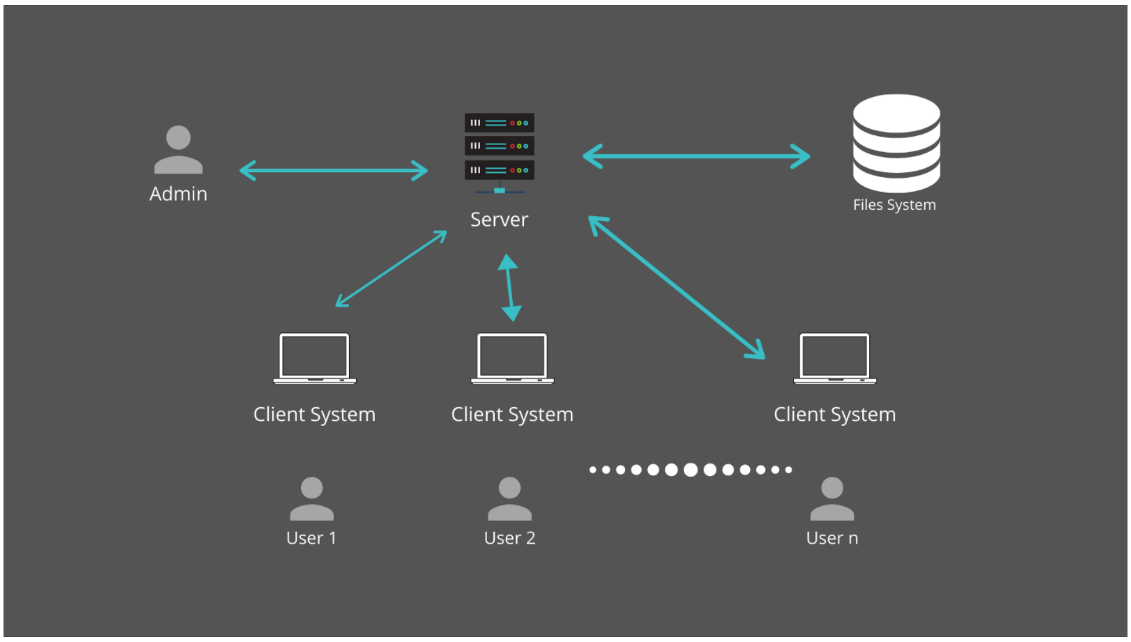


Figure 4.4: System Architecture

EXPLANATION:

This diagram elaborates the privileges of each individual respective to the system. There are three types of users students, faculty and administrator with their own privileges. As we can see a student can login and view the group chat and notice board but the faculty can view and also post on those two sections. Similarly a student can initiate chat with the faculty but the faculty can't initiate the chat a faculty can only chat with those students who initiated the chat not the other way around.

4.2.1 Message Exchange Architecture

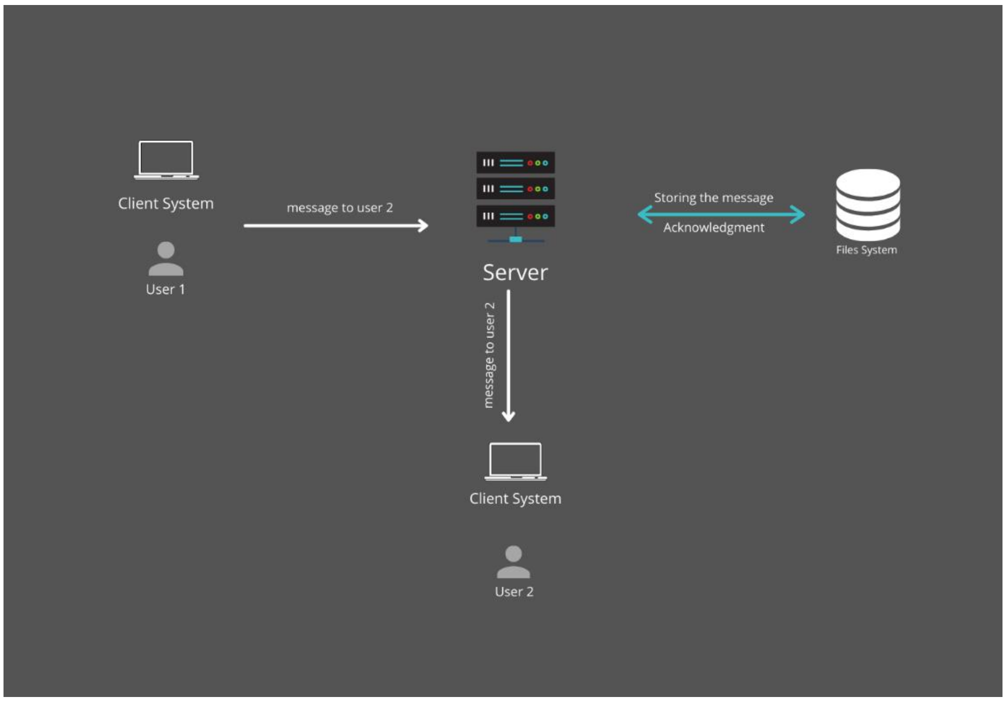


Figure 4.5: Message Exchange Architecture

EXPLANATION:

This diagram elaborates how the message request is handled by the server and then the request is taken to the other user. In the above figure we can see that the user 1 wants to send message to user 2 , so this request is sent to the server and the server first passes the data request to the file system to get stored then the server sends that data to the user 2 . This happens every time in the entire conversation from either user.

CHAPTER V

IMPLEMENTATION TECHNIQUES & ALGORITHMS

5.1 ALGORITHM

```
Step 1: Start
Step 2: Initialize the server
Step 3: Wait for the requests from the clients
Step 4: If the request is for message request then go to step 5, else go to step 7
Step 5: Store the message into the file system
Step 6: Take the message data to the appropriate user.
Step 7: Check whether the request is for notice board
Step 8:  If the requesting user is a student then grant view access else if the user
is a faculty grants view access and post access.
Step 9: Stop.
```

5.2 SAMPLE CODE SEGMENTS

```
//jshint esversion:6


var express = require("express");

const app = express();

var bodyParser = require("body-parser");


app.use(express.static("public"));

app.set("view engine", "ejs");

app.use(bodyParser.urlencoded({ extended: true }));

const server = require("http").createServer(app);

const io = require("socket.io")(server, { cors: { origin: "*" } });

const multer = require("multer");

const path = require("path");

const fs = require("fs");

const { createBrotliCompress } = require("zlib");

const { pathToFileURL } = require("url");

const { Console } = require("console");


////////////////////////////Initializing Variables//////////////////

var user_name_pass = 4;

var groupchat = 4;
var i = 0;

var n = 0;

var p = false;

var g = 0;

var MessageTemplate = 4;

var messageCount = 4;

var chats = 4;

var chats_length = 0;

var chat_length2 = 0;

var j = true;

var active_chat_template = 0;

var download_template = 0;


//////////////////Multer Set Storage Engine/////////////////////


const storage = multer.diskStorage({

  destination: "./public/uploads/",

  filename: function (req, file, cb) {

    cb(

      null,

      file.originalname + "-" + Date.now() + path.extname(file.originalname)

    );

  },
});

////////////Init upload////////////////////////////////////////

const upload = multer({
  storage: storage,
}).single("myfile");

/////////////////Loading User name and Passwords//////////////////

// fs.readFile(`${__dirname}/user_password.json`, 'utf-8', (err, data) => {
//   const productData = JSON.parse(data);
//   user_name_pass = productData;
//   n = productData.length;

// });

///////////////////loading Group chat//////////////////

fs.readFile(`${__dirname}/groupchat.json`, "utf-8", (err, data) => {
  const productData = JSON.parse(data);
  groupchat = productData;
  g = productData.length;
```

```
});

//////////////////loading all    chats//////////////////

fs.readFile(`${__dirname}/chats.json`, "utf-8", (err, data) => {
  const productData = JSON.parse(data);
  chats = productData;
  chats_length = productData.length;
});

/////////////////////loading message template///////////////////////

MessageTemplate = fs.readFileSync(`${__dirname}/tempMesg.html`, "utf-8");
active_chat_template = fs.readFileSync(
  `${__dirname}/tempActiveChat.html`,
  "utf-8"
);
download_template = fs.readFileSync(
  `${__dirname}/DownloadTemplate.html`,
  "utf-8"
);
```

```
///////////////////data filling function/////////////////////////


const replaceTemplate = (temp, el) => {
  let output = temp.replace(/{%%username%%}/g, el.username);
  output = output.replace(/{%%message%%}/g, el.message);
  return output;
};


//////////////Starting server///////////////////////////////


let port = process.env.PORT;
if (port == null || port == "") {
  port = 3000;
}


server.listen(port, function () {
  console.log(`Server Has started on port ${port}  Successfully`);
});


/////////////////////////Getting Requests from the Server/////////////////
app.get("/", function (req, res) {
  res.render("index", { message: "" });
});
```

```
app.get("/test", function (req, res) {
  fs.readFile(`${__dirname}/files.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);
    user_name_pass = productData;
    n = productData.length;
    var y = 0;
    var s = "";

    for (i = n - 1; i >= 0; i--) {
      var y = download_template.replace(
        /{%%description%%}/g,
        productData[i].Description
      );
      y = y.replace(/{%%username%%}/g, productData[i].username);
      y = y.replace(/{%%number%%}/g, i);

      s = s + y;
    }

    res.render("Studentupload", { data: s, username: "req.body.username1" });
  });
});
```

```
app.get("/testui", function (req, res) {
  res.render("whitechat");
});

app.get("/@@!Admin", function (req, res) {
  res.render("Admin", { message: "" });
});

////////////////////Upload Post Method For accepting Files////////////////

app.post("/upload", (req, res) => {
  upload(req, res, (err) => {
    fs.readFile(`${__dirname}/files.json`, "utf-8", (err, data) => {
      const productData = JSON.parse(data);

      n = productData.length;

      productData.push({
        filename: `${req.file.filename}`,
        Description: `${req.body.Description}`,
        username: `${req.body.username1}`,
      });
      fs.writeFile(
`${__dirname}/files.json`,
JSON.stringify(productData),
"utf-8",
function (err) {
  if (err) throw err;

  fs.readFile(`${__dirname}/files.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);
    user_name_pass = productData;
    n = productData.length;
    var y = 0;
    var s = "";

    for (i = n - 1; i >= 0; i--) {
      var y = download_template.replace(
        /{%%description%%}/g,
        productData[i].Description
      );
      y = y.replace(/{%%username%%}/g, productData[i].username);
      y = y.replace(/{%%number%%}/g, i);

      s = s + y;
    }
```

```
            res.render("upload", { data: s, username: req.body.username1 });
          });
        }
      );
    });
  });
});

////////////////User Validation post Method//////////////

app.post("/Login", function (req, res) {
  var p = false;

  fs.readFile(
    `${__dirname}/Teacher_user_password.json`,
    "utf-8",
    (err, data) => {
      const productData = JSON.parse(data);
      user_name_pass = productData;
      n = productData.length;

      for (i = 0; i < n; i++) {
```

```
  if (
    user_name_pass[i].username == req.body.urId &&
    user_name_pass[i].password == req.body.urPass
  ) {
    res.render("home", { message: req.body.urId });

    p = true;
  }
}

fs.readFile(
  `${__dirname}/Student_user_password.json`,
  "utf-8",
  (err, data) => {
    const productData = JSON.parse(data);
    user_name_pass = productData;
    n = productData.length;

    for (i = 0; i < n; i++) {
      if (
        user_name_pass[i].username == req.body.urId &&
        user_name_pass[i].password == req.body.urPass
      ) {
            res.render("Studenthome", { message: req.body.urId });

            p = true;
          }
        }
        if (p) {
        } else {
          res.render("index", { message: "Id or Password is incorrect" });
        }
      }
    );
  }
 );
});

app.get("/download/:id", (req, res) => {
  fs.readFile(`${__dirname}/files.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);

    var x =
      __dirname + "/public/uploads/" + productData[req.params.id].filename;
    res.download(x);
  });
```

```
});

app.post("/NoticeBoard", (req, res) => {
  fs.readFile(`${__dirname}/files.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);
    user_name_pass = productData;
    n = productData.length;
    var y = 0;
    var s = "";

    for (i = n - 1; i >= 0; i--) {
      var y = download_template.replace(
        /{%%description%%}/g,
        productData[i].Description
      );
      y = y.replace(/{%%username%%}/g, productData[i].username);
      y = y.replace(/{%%number%%}/g, i);

      s = s + y;
    }

    fs.readFile(
      `${__dirname}/Teacher_user_password.json`,
      "utf-8",
      (err, data) => {
        const productData = JSON.parse(data);
        user_name_pass = productData;
        n = productData.length;

        for (i = 0; i < n; i++) {
          if (user_name_pass[i].username == req.body.username1) {
            res.render("upload", { data: s, username: req.body.username1 });
          }
        }
      }
    );
    fs.readFile(
      `${__dirname}/Student_user_password.json`,
      "utf-8",
      (err, data) => {
        const productData = JSON.parse(data);
        user_name_pass = productData;
        n = productData.length;

        for (i = 0; i < n; i++) {
          if (user_name_pass[i].username == req.body.username1) {
            res.render("Studentupload", { data: s });
          }
        }
      }
    );

    //
  });
});

///////////////////Socket io////////////////////////

io.on("connection", (socket) => {
  fs.readFile(`${__dirname}/groupchat.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);
    groupchat = productData;
    g = productData.length;

    const y = groupchat
      .map((el) => replaceTemplate(MessageTemplate, el))
      .join("");

    socket.emit("chats", y);
```

```
});


socket.on("active_chats", (username) => {
  var h = "hello";
  fs.readFile(`${__dirname}/chats.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);
    groupchat = productData;
    g = productData.length;
    var c = [];

    for (var i = 0; i < productData.length; i++) {
      if (productData[i].username == username) {
        c.push(productData[i].with_user);
      } else if (productData[i].with_user == username) {
        c.push(productData[i].username);
      }
    }
    const y = c
      .map((el) => replaceTemplate2(active_chat_template, el))
      .join("");

    socket.emit("active_chats", y, username);
  });
```

```
});


const replaceTemplate2 = (temp, el) => {
  let output = temp.replace(/{%%username%%}/g, el);


  return output;
};


///////////////////chat clicked //////////////////


socket.on("chat_clicked", (with_user, username) => {
  var i,
    j = false;


  if (with_user == "groupchat") {
    fs.readFile(`${__dirname}/groupchat.json`, "utf-8", (err, data) => {
      const productData = JSON.parse(data);
      groupchat = productData;
      g = productData.length;


      const y = groupchat
        .map((el) => replaceTemplate(MessageTemplate, el))
        .join("");
      socket.emit("chats", y);
    });
  } else {
    fs.readFile(`${__dirname}/chats.json`, "utf-8", (err, data) => {
      const productData = JSON.parse(data);
      chats = productData;
      chats_length = productData.length;

      for (i = 0; i < chats_length; i++) {
        if (
          (chats[i].username == username &&
            chats[i].with_user == with_user) ||
          (chats[i].username == with_user && chats[i].with_user == username)
        ) {
          const y = chats[i].chat
            .map((el) => replaceTemplate(MessageTemplate, el))
            .join("");

          socket.emit("chat_received", username, with_user, y);
          socket.broadcast.emit("chat_received", username, with_user, y);

          j = true;
        }
      }

      if (j) {
      } else {
        fs.readFile(`${__dirname}/chats.json`, "utf-8", (err, data) => {
          const productData = JSON.parse(data);
          productData.push({
            username: `${username}`,
            with_user: `${with_user}`,
            chat: [],
          });

          fs.writeFile(
            `${__dirname}/chats.json`,
            JSON.stringify(productData),
            "utf-8",
            function (err) {
              if (err) throw err;
            }
          );

          socket.broadcast.emit("reload");
```

```
          socket.emit("reload");
        });
      }
    });
  }
});

//////////////On recieving the message from user///////////////
socket.on("message", (data, id, with_user) => {
  const mesg = data;
  if (data != null && data != "") {
    /////////////////////////////Storing the chat////////////////
    if (with_user == "groupchat") {
      fs.readFile(`${__dirname}/groupchat.json`, "utf-8", (err, data) => {
        const productData = JSON.parse(data);
        productData.push({ username: `${id}`, message: `${mesg}` });

        fs.writeFile(
          `${__dirname}/groupchat.json`,
          JSON.stringify(productData),
          "utf-8",
          function (err) {
            if (err) throw err;
      }
    );
  });
} else {
  fs.readFile(`${__dirname}/chats.json`, "utf-8", (err, data) => {
    const productData = JSON.parse(data);
    chats = productData;

    chat_length2 = productData.length;
    setTimeout(function () {
      for (i = 0; i < chat_length2; i++) {
        if (
          (chats[i].username == id && chats[i].with_user == with_user) ||
          (chats[i].username == with_user && chats[i].with_user == id)
        ) {
          chats[i].chat.push({ username: `${id}`, message: `${mesg}` });

          fs.writeFile(
            `${__dirname}/chats.json`,
            JSON.stringify(chats),
            "utf-8",
            function (err) {
              if (err) throw err;
```

```
                }
              );
            }
          }
        }, 1);
      });

      var i = 0;
    }

    ////////////////////////////////

    socket.emit("message", data, id, with_user);
    socket.broadcast.emit("message", data, id, with_user);
  }
 });
});

/////////////////////////////////////////////////////
```

```
//////////////// Adding users    /////////////////////

app.post("/add", function (req, res) {

  const c = req.body.urId;

  const d = req.body.urPass;

  const u = "Student_user_password";

  const p = "Teacher_user_password";

  var t = false;

  const k = req.body.filename1;


  if (c != null && c != "") {

    if (d != null && d != "") {

      t = true;

    }

  }


  if (t) {

    if (k == "Student") {

      fs.readFile(`${__dirname}/${u}.json`, "utf-8", (err, data) => {

        const productData = JSON.parse(data);

        productData.push({ username: `${c}`, password: `${d}` });


        fs.writeFile

`${__dirname}/${u}.json`,
          JSON.stringify(productData),
          "utf-8",
          function (err) {
            if (err) throw err;
          }
        );
      });
    }

    if (k == "Teacher") {
      fs.readFile(`${__dirname}/${p}.json`, "utf-8", (err, data) => {
   const productData = JSON.parse(data);
 productData.push({ username: `${c}`, password: `${d}` });
        fs.writeFile(
          `${__dirname}/${p}.json`,
          JSON.stringify(productData),  "utf-8",
          function (err) {
            if (err) throw err;
          }
        );
      }); }res.render("Admin", { message: "Added" });
  } else res.render("Admin", { message: "Please enter the values" });});
//////////////////////////////////////////////////////
```

CHAPTER VI

RESULTS

6.1 TESTING STRATEGY

IMPLEMENTATION AND TESTING

Implementation is one of the most important tasks in project phases in which one must be cautious because all the efforts undertaken during the project will be very interactive.

Implementation is the most crucial stage in achieving successful system and giving the user confidence that the new system is workable and effective. Each program is tested individually at the time if development using the sample data and has verified that these programs link together in the way specified in the program specification.

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

The test process is initiated by developing a comprehensive plan to test the general functionality and special features on a variety of platform combinations. Strict quality control procedures are used. The process verifies that the application meets the requirements specified in the system requirements document and is bug free. The following are the considerations used to develop the framework from developing the testing methodologies.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test.

System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

Performance Test

The Performance test ensures that the output is produced within the time limits, and the time taken by the system for compiling, giving response to the users and request being send to the system for to retrieve the results.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Acceptance testing for Data Synchronization:

- The Acknowledgements will be received by the Sender Node after the Packets are received by the Destination Node.
- The Route add operation is done only when there is a Route request in need.
- The Status of Nodes information is done automatically in the Cache Updating process

Build the test plan.

Any project can be divided into units that can be further performed for detailed processing. Then a testing strategy for each of this unit is carried out. Unit testing helps to identity the possible bugs in the individual component, so the component that has bugs can be identified and can be rectified from errors.

Implementation

The implementation phase is less creative than system design. It is primarily concern with user training, and file conversion. The system may be requiring extensive user training.

6.2 RESULTS SNAPSHOTS

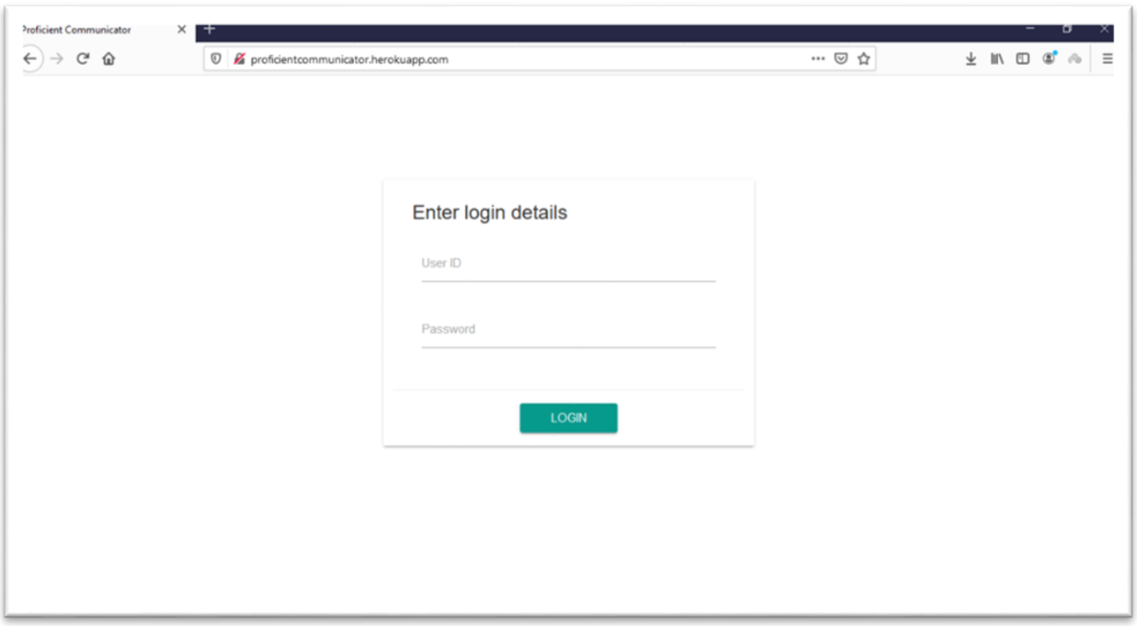
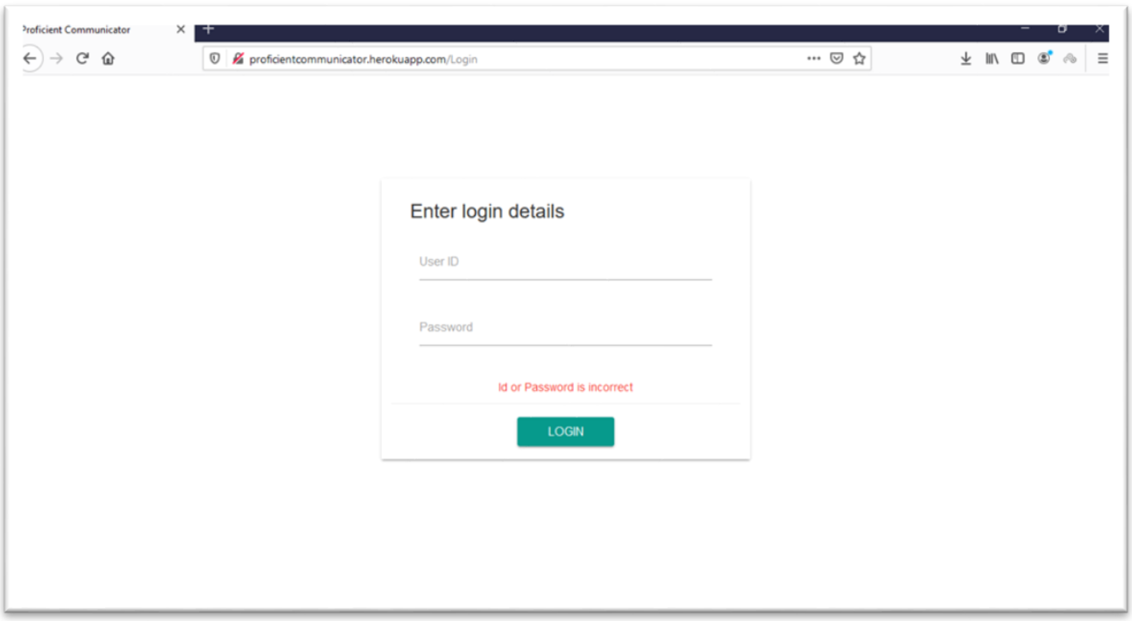


Fig 6.1: User is served the login page.



Id or Password is incorrect

Fig 6.2: When the user provides invalid credentials then the message is prompted

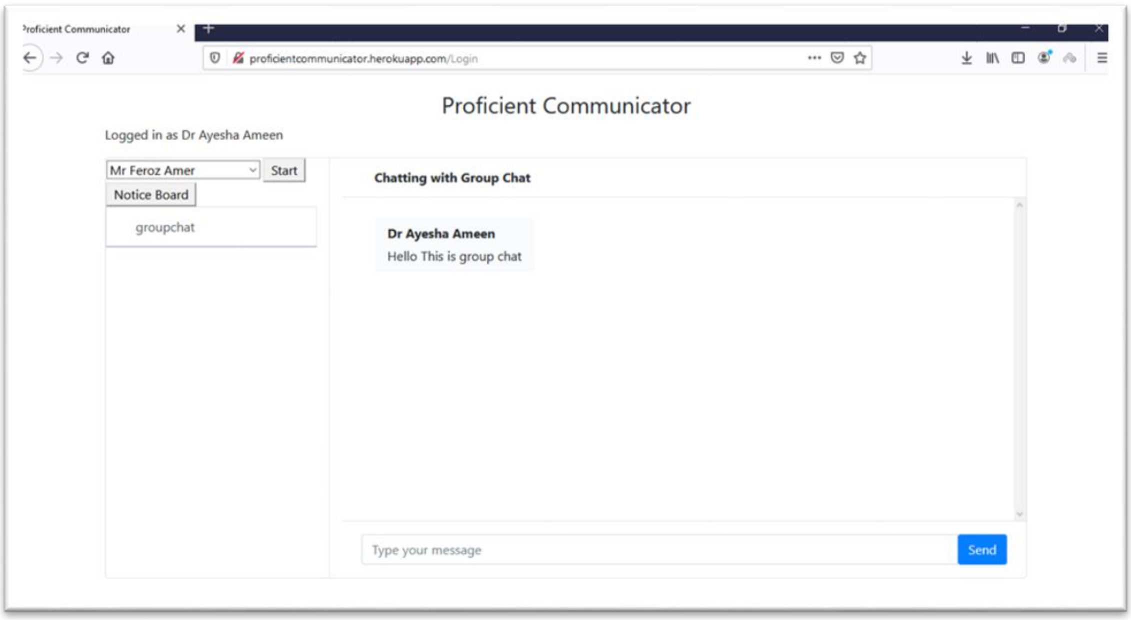


Fig 6.3 When the user is authorized as faculty then he/she is allowed to post on the group chat

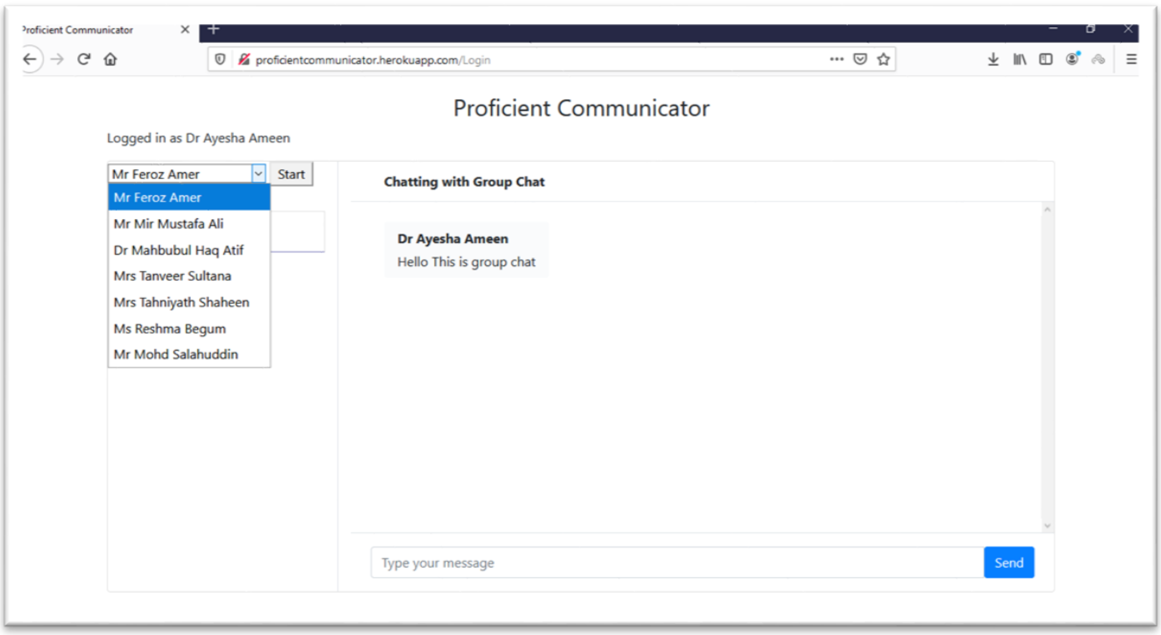
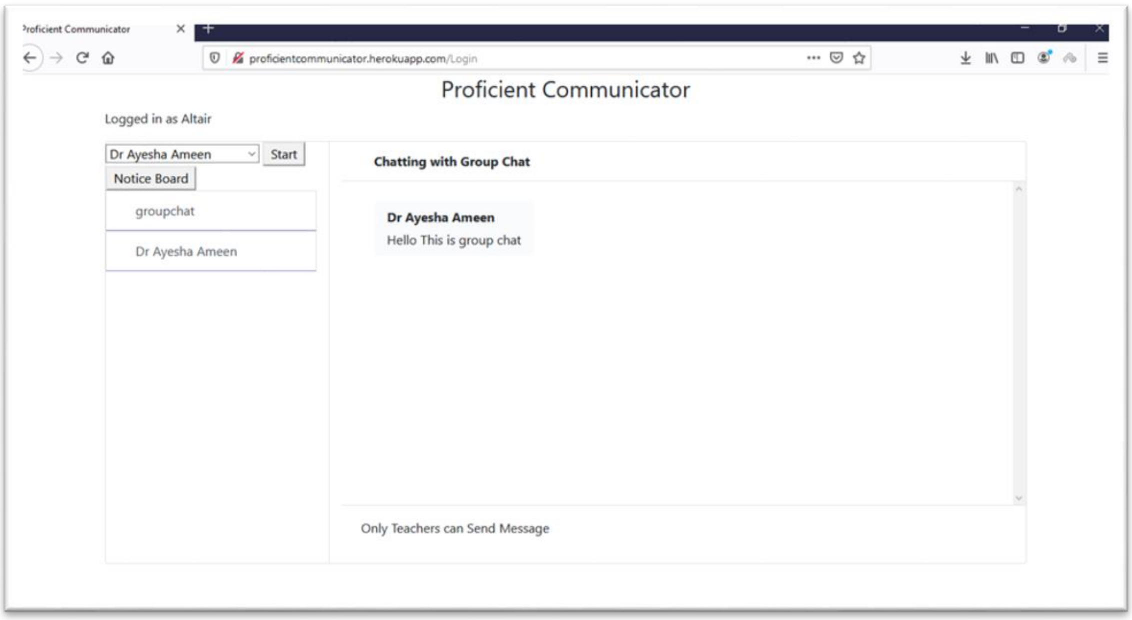


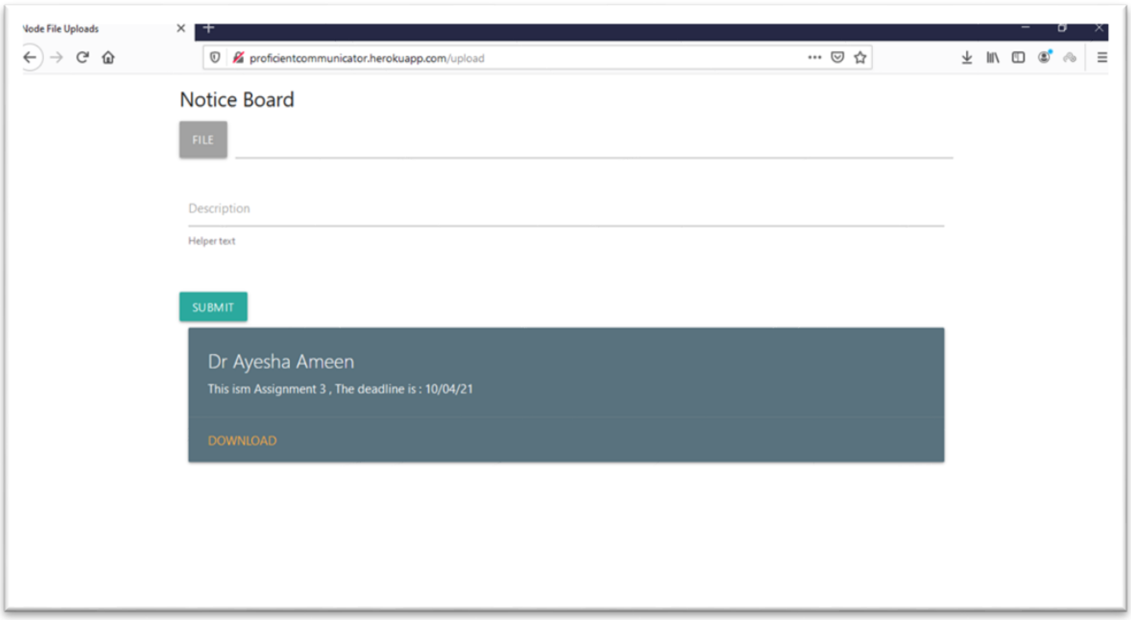
Fig 6.4 The user is allowed to initiate chat with faculties



Dr Ayesha Ameen
Hello This is group chat

Only Teachers can Send Message

Fig 6.5: When the user is authorized as student then he/she is not allowed to post on the group chat



Dr Ayesha Ameen
This ism Assignment 3 , The deadline is : 10/04/21

Fig 6.6: When the user is authorized as faculty then he/she is allowed to post on the notice board

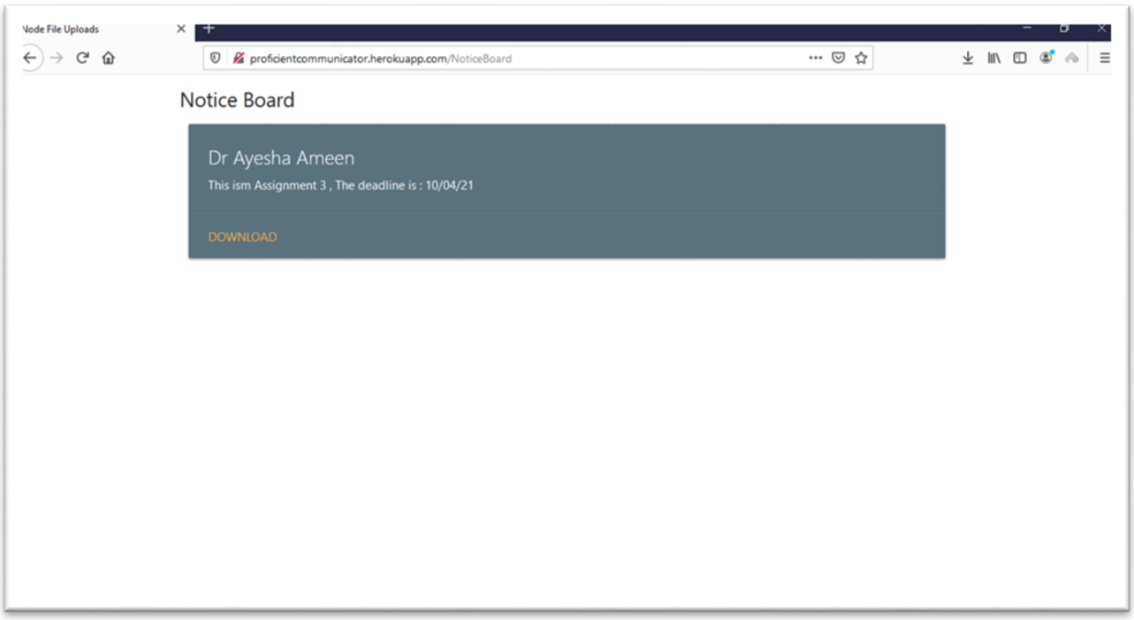

Node File Uploads
proficientcommunicator.herokuapp.com/NoticeBoard
Notice Board
Dr Ayesha Ameen
This ism Assignment 3 , The deadline is : 10/04/21
DOWNLOAD

Fig 6.7: When the user is authorized as student then he/she is not allowed to post on the notice

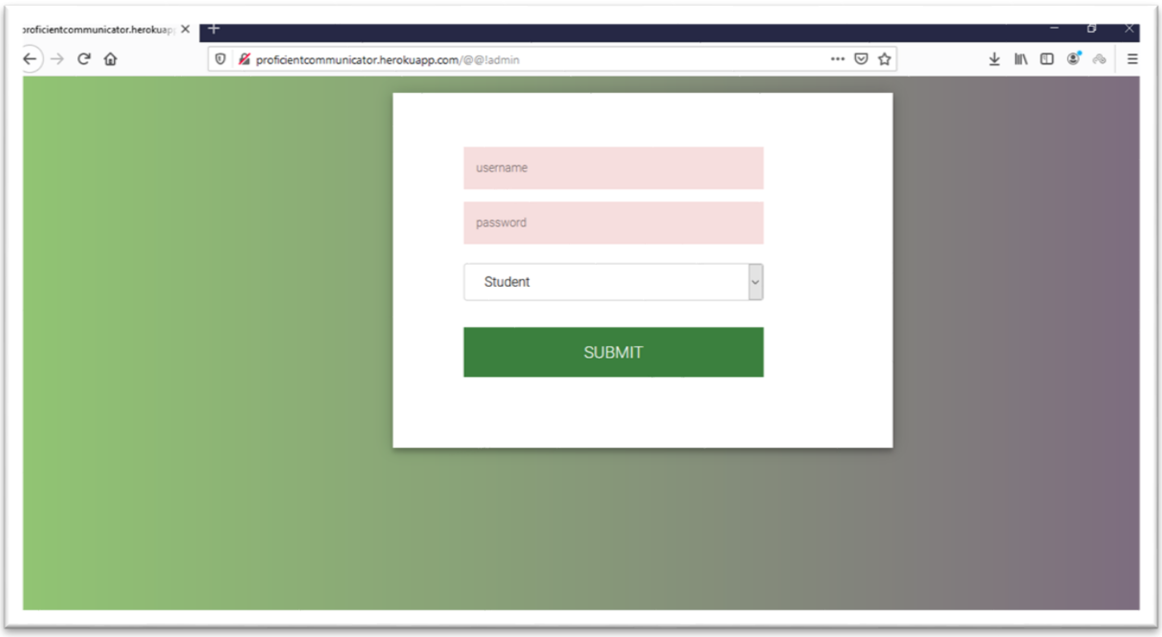


Fig 6.8: Admin panel is only authorized to the administrator.

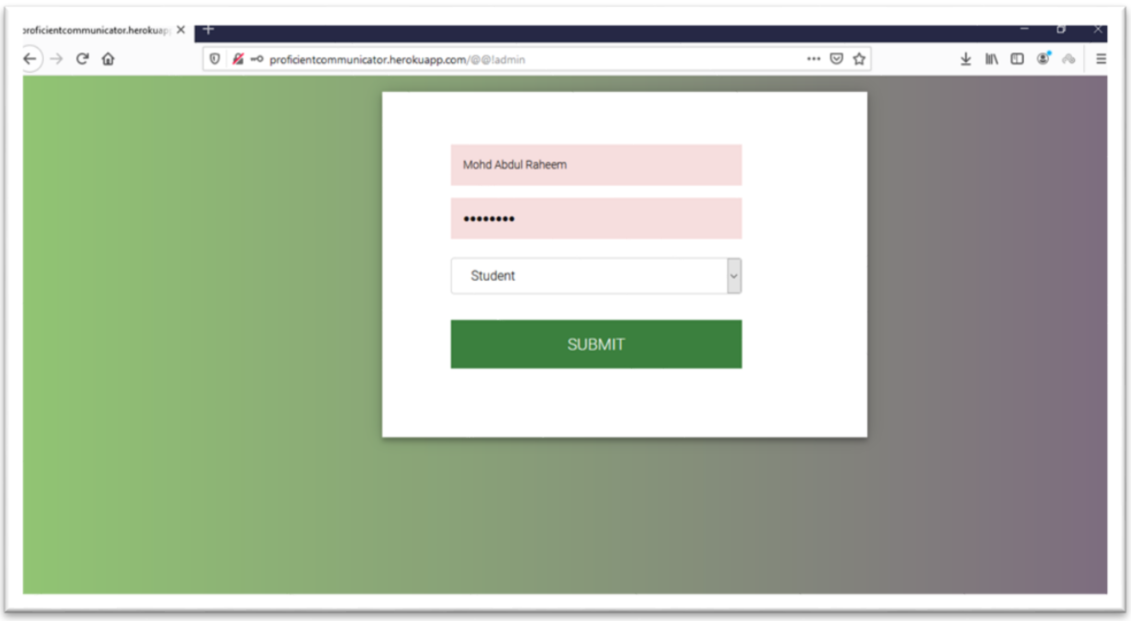


Fig 6.9: Administrator adds the users and authorizes the privileges of individual user simultaneously.

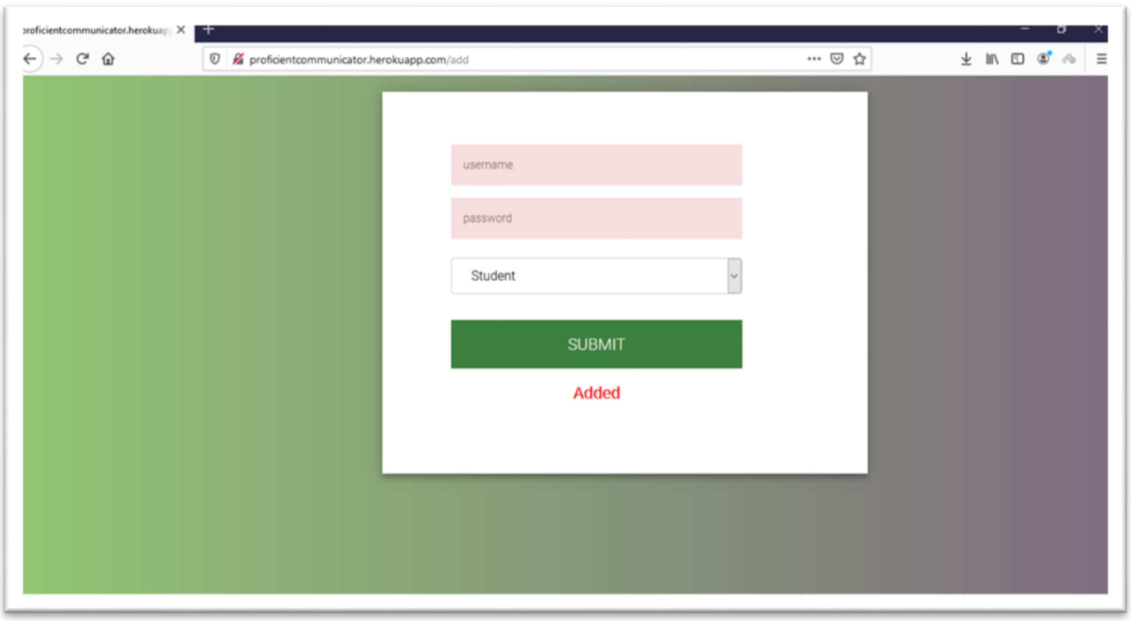


Fig 6.10: When the user is registered successfully then the Added alert is prompted

CHAPTER VII

CONCLUSION & FUTURE ENHANCEMENTS

7.1 FUTURE WORK

This project introduced as Dynamic Web application that fulfills all the professional needs of its user. It maintains privacy and security of every user. Using cloud server it is accessible to all the users throughout the globe, as the project is scalable it can handle the growth of users and accordingly provide features to the administrator of every organization respectively.

The project can be accessed through the below link.

proficientcommunicator.herokuapp.com/

7.2 CONCLUSION

The paper discusses the flaws of the existing system used in the organization and comes up with a solution that not only provides all the features of existing system but also additional features to the administrator which ensures a safe and controlled platform. The proposed system is easily acquirable due to its friendly user inter

REFERENCES

[1] “The complete 2021 Web Development Bootcamp”, a course by Angela Yu.

[2]”Modern JavaScript From the Beginning”, a course by traversy media.

[3]”React Front to Back”, a course by traversy media.

[4]”HTML and CSS : Design and Build Web Sites” , a book by Jon Duckett.

[5]”Java script and Jquery interactive Front-End Web Development”, a book by David McFarland.