MVJ COLLEGE OF ENGINEERING
Since 1982
An Autonomous Institute
(Affiliated to Visvesvaraya Technological University, Belagavi
Approved By AICTE, New Delhi,
Recognized by UGC under 2(f) & 12(B)
Accredited by NBA and NAAC)

A MINI PROJECT REPORT

ON

"STUDENT TEACHER NOTES"

Submitted in partial fulfillment of requirements for the award of 5th semester,

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted By:

Tanush R	**1MJ20CS218**
Vishal Gali	**1MJ20CS240**
Tellakulla Sathvik	**1MJ20CS220**
Varun M	**1MJ20CS233**

Under the Guidance of
Mrs. Navya
Assistant Professor, Department of computer science.

Department of Computer Science & Engineering

MVJ COLLEGE OF ENGINEERING
BENGALURU-67
ACADEMIC YEAR 2022-23

ABSTRACT

STUDENT TEACHER NOTES, An online platform for teachers to share notes, events, and photos of past events with their students is a useful tool for improving communication and collaboration between educators and learners. This type of website allows teachers to post important information, such as class notes and homework assignments, in a centralized location where students can access them at any time. In addition, the platform can be used to announce upcoming events, such as tests and field trips, and to share photos of past events, such as class projects and school outings. This helps to keep students informed and engaged in their studies, and allows teachers to more effectively communicate with their students outside of the classroom.

One potential benefit of this type of website is that it can make it easier for teachers to share resources with their students. Rather than having to distribute physical copies of notes or handouts in class, teachers can simply post them on the website for students to access at their convenience. This can save time and resources, and it can also make it easier for students to review material on their own or collaborate with their classmates.

Another benefit of this platform is that it can provide a more interactive and engaging learning experience for students. By allowing teachers to post photos and videos, as well as text-based content, the website can give students a more immersive and dynamic view of their coursework. It can also encourage greater participation and engagement from students, as they can ask questions, share their own work, and interact with their teachers and peers through the platform. Overall, this type of website has the potential to enhance communication, collaboration, and learning for both teachers and students.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success.

We express our sincere gratitude to our Principal **Dr.P. Mahabaaleshwarappa**, and Vice Principal **Dr.M. Brindha**, MVJ College of Engineering for providing the facilities.

We wish to place on record my grateful thanks to **Dr Sudhan MB**, Head of the Department, Computer Science and Engineering MVJ College of Engineering, Bangalore for providing encouragement and guidance.

We consider it a privilege and honor to express my sincere gratitude to my guide **Mrs. Navya,** CSE Department of Computer Science & Engineering for their valuable guidance throughout the tenure of this mini project work and whose support and encouragement made this work possible.

We wish to thank the faculty of Computer Science and Engineering department whose suggestions have enabled me to surpass many of the seemingly impossible hurdles.

Thank you.

TABLE OF CONTENT

INTRODUCTION

Database is a collection of data and Management System is a set of programs to store and retrieve those data. Based on this one can define DBMS as a collection of inter-related data and set of programs to store & access those data in an easy and effective manner.

1.1 What is the need of DBMS?

Database systems are basically developed for large amount of data. When dealing with huge amount of data, there are two things that require optimization: Storage of data and retrieval of data.
Storage: According to the principles of database systems, the data is stored in such a way that it acquires lot less space as the redundant data (duplicate data) has been removed before storage. Let's take a layman example to understand this. In a banking system, suppose a customer is having two accounts, one is saving account and another is salary account. Let's say bank stores saving account data at one place (these places are called tables we will learn them later) and salary account data at another place, in that case if the customer information such as customer name, address etc. are stored at both places then this is just a wastage of storage (redundancy/ duplication of data), to organize the data in a better way the information should be stored at one place and both the accounts should be linked to that information somehow. The same thing we achieve in DBMS.
Fast Retrieval of data: Along with storing the data in an optimized and systematic manner, it is also important that we retrieve the data quickly when needed. Database systems ensure that the data is retrieved as quickly as possible.
The choice of a database product is often influenced by factors such as:

- the computing platform (i.e., hardware, operating system)
- the volume of data to be managed
- the number of transactions required per second
- existing applications or interfaces that an organization may have
- support for heterogeneous and/or distributed computing
- cost
- vendor support

1.2 Design and Modeling:

The first task of a database designer is to produce a conceptual datamodel that reflects the structure of the information to be held in the database. A common approach to this is to develop an entity-relationship model, often with the aid of drawing tools. Another popular approach is the Unified Modeling Language. A successful data model will accurately reflect the possible state of the external world being modeled: for example, if people can have more than one phone number, it will allow this information to be captured.

1.3 Objective

The main objective of this project is to determine how an interactive Website helps to interact between students and teachers to share resource material and upcoming events in campus. This project is a two-tier architecture application.

1.4 Problem Statement

Existing methods are conventional which can be overcome using digital platform lot paperwork can be avoided, sources are sorted can be easily accessed, lesser time required to access the source.

1.5 Scope of the report

The essential framework of this report would be to elaborate the design of E.R-diagram, Schema Diagram and to display how the modules of the program work in order to achieve the automation

SYSTEM REQUIREMENTS

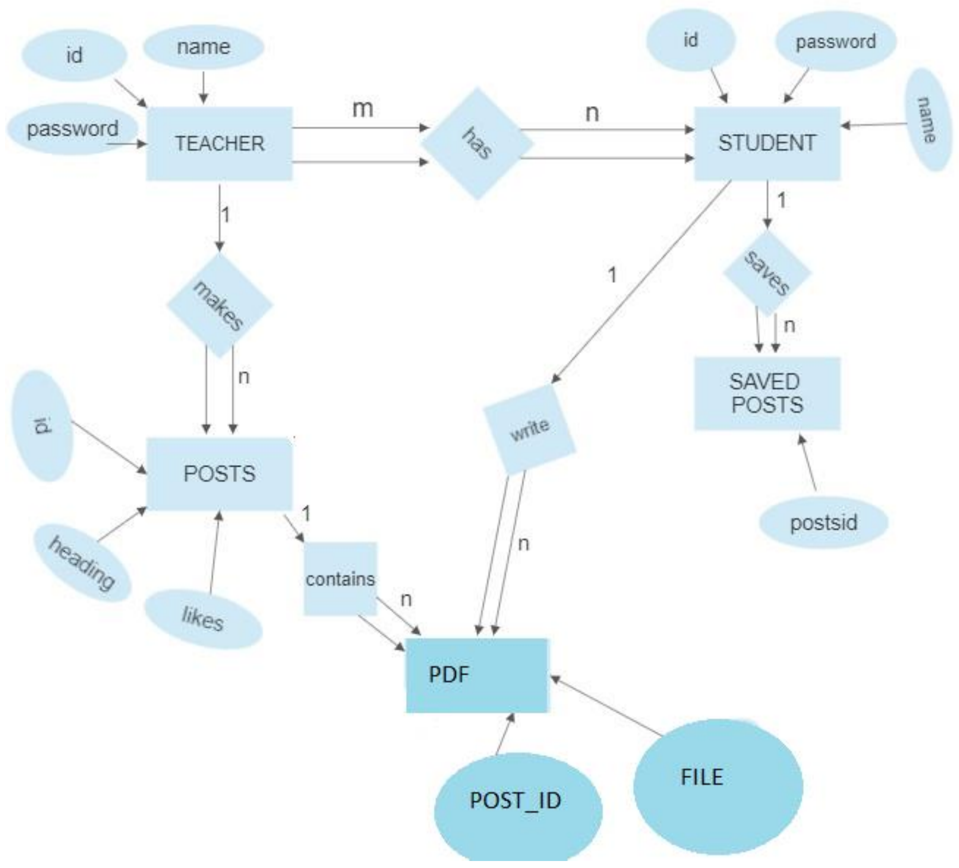
Hardware Requirements

. Windows
. Operating System: Linux ARCH distribution
. Processor: Intel i3 4th Gen
. Memory: 4GB minimum, 8GB recommended
. Screen resolution:1280*1024 or larger
. Application Window Size:1024*680 or larger
. Internet Connection: Required

Software Requirements

. Client: Operating System{any}
. Web Server: Flask (Python), ReactCLI
. Database: MySQL
. Language: HTML, CSS, JS, SQL, Python

ER DIAGRAM



SCHEMA DIAGRAM

TEACHER

<u>id</u>	**name**	**password**

STUDENT

<u>id</u>	**name**	**password**

POSTS

<u>id</u>	**heading**	**likes**

PDF

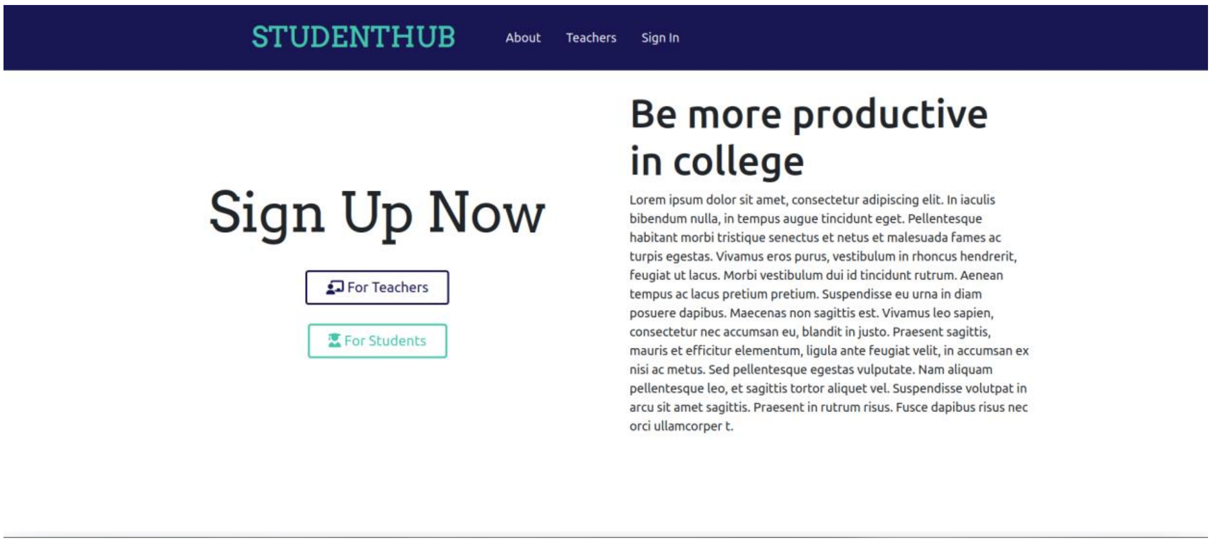
<u>post_id</u>	**file**

SAVED_POSTS

<u>post_id</u>	**<u>student_email</u>**

QUICK PEEK

Login Page :

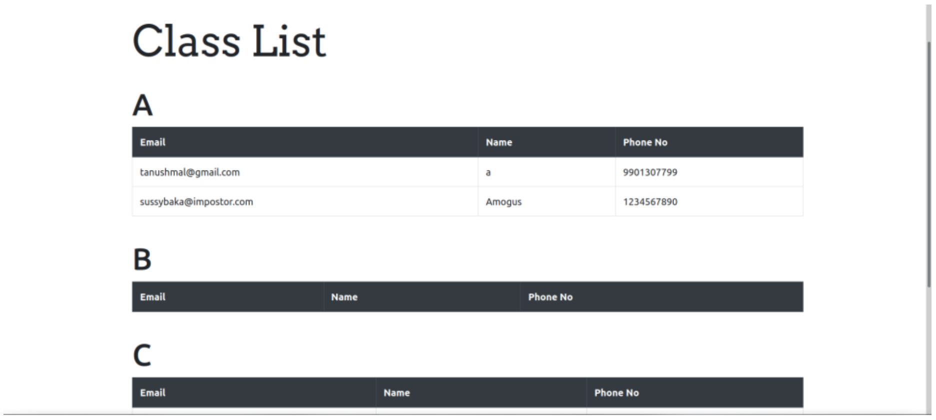


Sign Up Now

Be more productive in college

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In iaculis bibendum nulla, in tempus augue tincidunt eget. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Vivamus eros purus, vestibulum in rhoncus hendrerit, feugiat ut lacus. Morbi vestibulum dui id tincidunt rutrum. Aenean tempus ac lacus pretium pretium. Suspendisse eu urna in diam posuere dapibus. Maecenas non sagittis est. Vivamus leo sapien, consectetur nec accumsan eu, blandit in justo. Praesent sagittis, mauris et efficitur elementum, ligula ante feugiat velit, in accumsan ex nisi ac metus. Sed pellentesque egestas vulputate. Nam aliquam pellentesque leo, et sagittis tortor aliquet vel. Suspendisse volutpat in arcu sit amet sagittis. Praesent in rutrum risus. Fusce dapibus risus nec orci ullamcorper t.

Posts:



Class List

A

Email	Name	Phone No
tanushmal@gmail.com	a	9901307799
sussybaka@impostor.com	Amogus	1234567890

B

Email	Name	Phone No

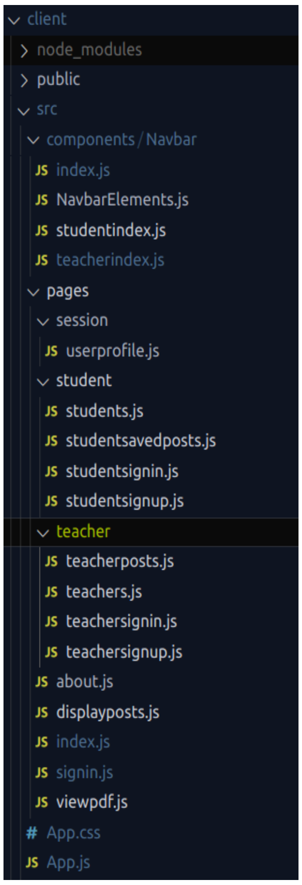
C

Email	Name	Phone No

IMPLEMENTATION

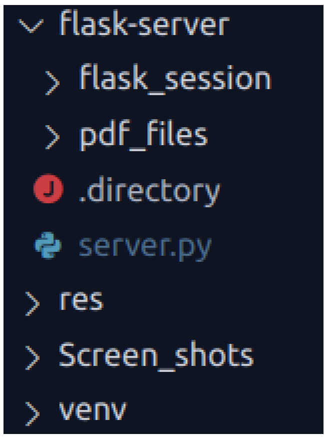
FILE STRUCTURE

React Client



```
client
  node_modules
  public
  src
    components / Navbar
      index.js
      NavbarElements.js
      studentindex.js
      teacherindex.js
    pages
      session
        userprofile.js
      student
        students.js
        studentsavedposts.js
        studentsignin.js
        studentsignup.js
      teacher
        teacherposts.js
        teachers.js
        teachersignin.js
        teachersignup.js
      about.js
      displayposts.js
      index.js
      signin.js
      viewpdf.js
    App.css
    App.js
```

Flask Server



```
flask-server
  flask_session
  pdf_files
  .directory
  server.py
res
Screen_shots
venv
```

server.py

```
from flask import Flask,jsonify,request, session, make_response, send_file
from flask_session import Session
import mysql.connector as mysql
import time
import os

app = Flask(__name__)


# app.config["SESSION_COOKIE_SAMESITE"] = "None"
# app.config["SESSION_COOKIE_SECURE"] = True
# app.secret_key = 'your-secret-key'

mydb = mysql.connect(
  host="localhost",
  user="client",
  password="client",
  database = "studenthub"
)


@app.route("/teachers_list_api")
def teachers_list_api():
    classList = ["5A","5B","5C","5D"]
    ret_dict = {}
    for classSec in classList:
        mycursor = mydb.cursor()
        mycursor.execute("SELECT * FROM teachers WHERE CLASS = 
%s;",(classSec,))
        classSecTeachers = mycursor.fetchall()
        ret_dict[classSec[1]] = classSecTeachers
    return ret_dict

@app.route("/students_list_api")
def students_list_api():
    classList = ["5A","5B","5C","5D"]
    ret_dict = {}
    for classSec in classList:
        mycursor = mydb.cursor()
        mycursor.execute("SELECT * FROM students WHERE CLASS =
%s;",(classSec,))
        classSecTeachers = mycursor.fetchall()
        ret_dict[classSec[1]] = classSecTeachers
    return ret_dict

@app.route("/sign_up_teacher", methods =['POST'])
def sign_up_teacher():
    content = request.json
    email = content['email']
    name = content['name']
    phone = str(content['phone'])
    password = content['password']
    classVal = content['class']
    mycursor = mydb.cursor()
    mycursor.execute("INSERT INTO
teachers(EMAIL,NAME,PHONE,PASSWORD,CLASS) VALUES
(%s,%s,%s,%s,%s);",(email,name,phone,password,classVal))
    myresult = mycursor.fetchall()
    mydb.commit()
    return jsonify(myresult)

@app.route("/sign_in_teacher", methods=['POST'])
def sign_in_teacher():
    content = request.json
    email = content['email']
    password = content['password']
    mycursor = mydb.cursor()
    mycursor.execute("SELECT * FROM teachers WHERE EMAIL = %s
AND PASSWORD = %s;",(email,password))
    myresult = mycursor.fetchone()
    print(myresult)
    if myresult == None:
        return jsonify({"exists":"no"})
    session['email'] = email
    session['name'] = myresult[2]
    session['sem'] = myresult[5]
    session['logged'] = True
    session['type'] = "teacher"
    print(session['email'])
    return jsonify({"exists":"yes"})

@app.route("/sign_up_student", methods=["POST"])
def sign_up_student_api():
    content = request.json
    email = content['email']
    name = content['name']
    phone = str(content['phone'])
    password = content['password']
    classVal = content['class']
    mycursor = mydb.cursor()
    mycursor.execute("INSERT INTO
students(EMAIL,NAME,PHONE,PASSWORD,CLASS) VALUES
(%s,%s,%s,%s,%s);",(email,name,phone,password,classVal))
    myresult = mycursor.fetchall()
    mydb.commit()
    return jsonify(myresult)

@app.route("/sign_in_student", methods=['POST'])
def sign_in_student():
    content = request.json
    email = content['email']
    password = content['password']
    mycursor = mydb.cursor()
    mycursor.execute("SELECT * FROM students WHERE EMAIL = %s
AND PASSWORD = %s;",(email,password))
    myresult = mycursor.fetchone()
    print(myresult)
    if myresult == None:
        return jsonify({"exists":"no"})
    session['email'] = email
    session['name'] = myresult[2]
    session['sem'] = myresult[5]
    session['logged'] = True
    session['type'] = "student"
    print(session['email'])
    return jsonify({"exists":"yes"})
```

```
@app.route("/session_details_api", methods=['GET'])
def session_check_api():
    try:
        return jsonify({
            "logged": session["logged"],
            "type": session["type"],
            "details": {
                "email":  session["email"],
                "name": session["name"],
                "sem": session["sem"]
            }
    })
    except:
        return jsonify({
            "logged": False })

@app.route("/session_logout_api", methods=['GET'])
def session_logout_api():
    session["logged"] = False,
    session["email"] = None,
    session["type"] = None,
    session["name"] = None,
    session["sem"] = None
    return jsonify({"logged":session["logged"]})


@app.route("/create_post_api",methods=["POST"])
def create_post_api():
    content = request.json
    title = content['title']
    content_api = content['content']
    temail = session["email"]
    created_date = time.strftime('%Y-%m-%d %H:%M:%S')
    notes_pdf = "No"
    mycursor = mydb.cursor()
    mycursor.execute("INSERT INTO posts(title, content, temail,
created_date, notes_pdf) VALUES (%s,%s,%s,%s,%s);",(title,
content_api, temail, created_date, notes_pdf))
    myresult = mycursor.fetchall()
    mydb.commit()
    mycursor.execute("SELECT pid from posts where title = %s and temail
= %s",(title,temail))
    pid = mycursor.fetchone()[0]
    session["pid"] = pid
    print(pid)
    return jsonify({"success":True})

@app.route("/upload_pdf_api",methods=["POST","GET"])
def upload_pdf_api():
    file = request.files['file']
    if file.filename[-4:] != ".pdf":
        return(jsonify({"success":False}))
    pid = session["pid"]
    mycursor = mydb.cursor()
    mycursor.execute("UPDATE posts set notes_pdf='Yes' where pid =
%s",(pid,))
    myresult = mycursor.fetchall()
    file.save(os.path.join(app.config['UPLOAD_FOLDER'],"Notes" +
str(pid) + ".pdf"))
    session.pop("pid")
    return(jsonify({"success":True}))

@app.route("/show_posts_api",methods=["GET"])
def show_posts_api():
    mycursor = mydb.cursor()
    mycursor.execute("SELECT
title,content,name,created_date,pid,notes_pdf FROM posts,teachers where
posts.temail = teachers.email ORDER BY created_date DESC")
    posts = mycursor.fetchall()
    mydb.commit()
    if session["type"] != "student":
        return jsonify({
            "result":posts,
            "student":False
            })
    email = session["email"]
    mycursor = mydb.cursor()
    mycursor.execute("SELECT pid FROM saved_posts WHERE email =
%s",(email,))
    myresult = [x[0] for x in mycursor.fetchall()]
    print(myresult)
    return jsonify({
            "result":posts,
            "student":True,
            "saved": myresult
            })

@app.route("/save_post_api",methods=["POST"])
def save_post_api():
    content = request.json
    print(content)
    pid = content["pid"]
    email = session["email"]
    mycursor = mydb.cursor()
    mycursor.execute("INSERT INTO saved_posts VALUES
(%s,%s)",(email,pid))
    myresult = mycursor.fetchall()
    mydb.commit()
    return myresult

@app.route("/show_saved_posts_api",methods=["GET"])
def show_saved_posts_api():
    email = session["email"]
    mycursor = mydb.cursor()
    mycursor.execute("SELECT title,content,name,created_date,posts.pid
FROM posts,teachers,saved_posts where posts.temail = teachers.email and
posts.pid = saved_posts.pid and saved_posts.email  = %s ORDER BY
created_date DESC",(email,))
    myresult = mycursor.fetchall()
    print(myresult)
    return jsonify({"result":myresult})

@app.route("/get_pdf_notes/<key>",methods=["GET"])
def get_pdf_notes(key):
    # with open(os.path.join(app.config['UPLOAD_FOLDER'],"Notes" +
str(32) + ".pdf"), "r") as file :
    #     response = make_response(file)
    #     cd = f"attachment; filename=Notes"+str(32)+".pdf"
    #     response.headers['Content-Disposition'] = cd
    #     response.mimetype='application/pdf'
    #     return response
    return send_file('./pdf_files/Notes'+str(key)+'.pdf')

if __name__ == "__main__":
    SECRET_KEY = "changeme"
    SESSION_TYPE = "filesystem"
    UPLOAD_FOLDER = "./pdf_files/"
    app.config.from_object(__name__)
    app.run(debug = True)
```

App.js

```
import React from 'react';
import './App.css';
import Navbar from './components/Navbar';
import { BrowserRouter as Router, Routes, Route, useLocation} from
'react-router-dom';
import { useEffect, useState } from 'react';
import Home from './pages';
import About from './pages/about';
import SignIn from './pages/signin';
import TeacherSignUp from './pages/teacher/teachersignup';
import Teachers from './pages/teacher/teachers'
import TeacherSignIn from './pages/teacher/teachersignin'
import TeacherPosts from './pages/teacher/teacherposts';
import TeacherNavbar from './components/Navbar/teacherindex';
import DisplayPosts from './pages/displayposts'
import StudentSignUp from './pages/student/studentsignup';
import StudentSignIn from './pages/student/studentsignin';
import axios from 'axios';
import LogOut from './logout';
import Students from './pages/student/students';
import StudentNavbar from './components/Navbar/studentindex';
import DisplaySavedPosts from './pages/student/studentsavedposts';
import ViewPdf from './pages/viewpdf';

function App() {
        const [profile, setProfile] = useState("")
        const [location, setLocation] = useState("");
        // const [loading,setLoading] = useState(false)
        useEffect(
                () => {axios.get("/session_details_api").then(
                        response => {
                                setProfile(response.data.type)
                        })
                }, [location] )
        const SettingLocation = () => {
                setLocation(useLocation())
        }
        return (
                <Router>
                <SettingLocation />
                {profile === 'teacher' ?
                <TeacherNavbar /> :
                profile === 'student' ?
                <StudentNavbar /> : <Navbar />
                }
                <div class="container pt-4">
                        <Routes>
                                <Route path='/' element={<Home />} />
                                <Route path='/about' element={<About/>} />
                                <Route path='/sign-in' element={<SignIn/>} />
                                <Route path='/teacher-sign-in'
element={<TeacherSignIn/>} />
                                <Route path='/teachers' element={<Teachers/>} />
                                <Route path='/teacher-sign-up'
element={<TeacherSignUp/>} />
                                <Route path='/teacher-posts'
element={<TeacherPosts/>} />
                                <Route path='/display-posts' element={<
DisplayPosts />} />
                                <Route path='/students' element={<Students/>} />
                                <Route path='/student-sign-in'
element={<StudentSignIn/>} />
                                <Route path='/student-sign-up'
element={<StudentSignUp/>} />
                                <Route path='/log-out' element={<LogOut/>} />
```

```
                                        <Route path='/show-saved-posts'
element={<DisplaySavedPosts />} />
                                        <Route path='/view-pdf/:id' element={<ViewPdf
/>} />
                              </Routes>
                    </div>
                    </Router>
);
}

export default App;
```

App.css

```
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}

@keyframes App-logo-spin {
  from {
    transform: rotate(0deg);
  }
  to {
    transform: rotate(360deg);
  }
}

body {
  font-family: Ubuntu;
}

.vertical-center {
  min-height: 50vh; /* These two lines are counted as one :-)       */

  display: flex;
  align-items: center;
}

h1 {
  font-family: 'Arvo';
  font-size: 70px;
}
h2 {
  font-size: 50px;
}

.btn-outline-primary:hover, .btn-outline-primary:active, .btn-outline-
primary:visited {
  background-color:#191654;
}

.btn-outline-primary {
  border: 3px solid #191654;
  color:#191654;
  font-size: large;
}

.btn-outline-success:hover, .btn-outline-success:active, .btn-outline-
success:visited {
  background-color:#43C6AC;
}

.btn-outline-success {
  border: 3px solid #43C6AC;
  color:#43C6AC;
  font-size: large;
}

img {
  width: 100px;
}

iframe {
  position:absolute;
}
/* .btn-danger {
  color:#ff3759;
} */
```

about.js

```
import React from "react";

const About = () => {
return (
        <div>
        <h1>
                About StudentHub
        </h1>
        <p>StudentHub is a project created using React +
Flask(Python)</p>
        </div>
);
};

export default About;
```

displayposts.js

```
import React, {useState, useEffect} from 'react'
import axios from 'axios';
import { Link } from 'react-router-dom';

export default function DisplayPosts() {

   const [data, setData] = useState({})
   const [saved, setSaved] = useState({})
   const [student, setStudent] = useState(false)
   const [load,setLoad] = useState(false)

   useEffect(() => {
      axios.get("/show_posts_api").then((response) => {
         setData(response.data.result)
         setStudent(response.data.student)
         setSaved(response.data.saved)
         setLoad(true)
         console.log(data)
      })
   }, [] );

   const savePost = (pid) => {
      const returnJSON = {"pid":pid}
      axios.post("/save_post_api",returnJSON).then((response) => {
         console.log(response)
      } )
      document.getElementById(pid).className = "btn btn-danger
disabled"
      document.getElementById(pid).innerHTML = "Post Saved"
```

```
    }

      return !load ? <p>Loading...</p> : (
       <>
              { Array.from(data).map(
                 bruh => <div>
                    <div className="card border-dark mb-3 mt-5">
              <div className="card-header">{bruh[2]} <div
className="small">{bruh[3].slice(4,17)}</div></div>
              <div className="card-body text-dark">
                 <h3 className="card-title">{bruh[0]}</h3>
                 <p className="card-text pt-4">{bruh[1]}</p>
              </div>
              <div className="card-footer">
              {student ?
              <div>

                 {saved.includes(bruh[4]) ?
                 <button id={bruh[4]} className='btn btn-danger disabled'>
                    Post Saved
                 </button>
                 :
                 <button id={bruh[4]} className='btn btn-danger'
onClick={() => savePost(bruh[4])}>
                    Save Post
                 </button>
                 }

                </div>
              : <></>}
                 <div className='mt-2'>
                 {bruh[5] === "Yes" ?
                 <Link to={"/view-pdf/"+bruh[4]} className='btn btn-
danger'>
                    View PDF
                 </Link> : <></>}
                 </div>
              </div>
              </div>
                 </div>
```

```
            )}
        </>
    )
}
```

index.js

```
import React from 'react';
import { Link } from "react-router-dom";
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
import { faChalkboardTeacher, faUserGraduate } from
'@fortawesome/free-solid-svg-icons'

const Home = () => {
return (
         <>
        <div className="row d-flex align-items-center">
            <div className="col vertical-center">
                <div className="text-center ml-5">
                    <h1>Sign Up Now</h1>
                        <Link to="/teacher-sign-up" className="btn btn-outline-
primary mt-4 pl-4 pr-4"><FontAwesomeIcon
icon={faChalkboardTeacher} />                    For Teachers</Link> <br/>
                        <Link to="/student-sign-up" className="btn btn-outline-
success mt-4 pl-4 pr-4"><FontAwesomeIcon icon={faUserGraduate} />
For Students</Link>
                    </div>
                </div>

                <div className="col">
                    <div className="container">
                        <h2>Create Posts, Manage Students</h2>
                        <p>
                            Studenthub is the place for all your needs, teacher or
student. Support for four sections in the class. Your post can contain
10000 characters. Now that is a lot of
```

```
                        characters. You can save as many posts you want for
your future reference. This amazing website is powered by both Python
and
                        JavaScript. Create your account now!
                    </p>
                 </div>
          </div>
                </div>
        </>
);
};

export default Home;
```

signin.js

```
import React from 'react';
import { Link } from "react-router-dom";
import { FontAwesomeIcon } from '@fortawesome/react-fontawesome'
import { faChalkboardTeacher, faUserGraduate } from
'@fortawesome/free-solid-svg-icons'
import { FaUserGraduate } from 'react-icons/fa';

const SignIn = () => {
return (
        <>
       <div className="row d-flex align-items-center">
          <div className="col vertical-center">
             <div className="text-center ml-5">
                <h1>Login Now</h1>
                   <Link to="/teacher-sign-in" className="btn btn-outline-
primary mt-4 pl-4 pr-4"><FontAwesomeIcon
icon={faChalkboardTeacher} />                For Teachers</Link> <br/>
                   <Link to="/student-sign-in" className="btn btn-outline-
success mt-4 pl-4 pr-4"><FontAwesomeIcon icon={faUserGraduate} />
For Students</Link>
                   <p className='mt-5'>Haven't created a profile yet? <Link
to="/sign-up" className=">Sign up now</Link></p>
                </div>
                </div>

                <div className="col">
                    <div className="container">
                        <h2>Unlimited posts!</h2>
                        <p>
                            By using MySQL DBMS powered systems we can store
as many posts as we like.
                        </p>
                        <h2 className='mt-2'>Powered By</h2>
                        <div className='row mt-4'>
                            <div className='col'>
                                <img
src="https://assets.stickpng.com/images/5848152fcef1014c0b5e4967.png"
className="img-rounded" />
                            </div>
                            <div className='col'>
                                <img
src="https://upload.wikimedia.org/wikipedia/commons/thumb/a/a7/React-
icon.svg/2300px-React-icon.svg.png" className="img-rounded" />
                            </div>
                            <div className='col'>
                                <img
src="https://www.freepnglogos.com/uploads/logo-mysql-png/logo-mysql-
mysql-logo-png-images-are-download-crazypng-21.png"
className="img-rounded" />
                            </div>
                        </div>
                    </div>
              </div>
                  </div>

          </>
);
};

export default SignIn;
```

viewpdf.js

```
import React from 'react'
import { useParams } from 'react-router-dom';

export default function ViewPdf() {
        let { id } = useParams();
        return (
                <div className=">
                        <iframe src={"http://localhost:5000/get_pdf_notes/"+id}
width="75%" height="75%">
                        <p>This browser does not support PDFs. Please download
the PDF to view it: <a href="/get_pdf_notes">Download PDF</a></p>
                        </iframe>
                </div>
        )
}
```

index.js(NavBar)

```
import React from "react";
import UserProfile from "../../pages/session/userprofile";
import { Nav, NavLink, NavMenu, NavHead }
         from "./NavbarElements";
import { useNavigate } from "react-router-dom";

export default function Navbar() {
    const navigate = useNavigate();


    return (
        <>
        <Nav>

            <NavMenu>
                <NavHead to="/">
                    STUDENTHUB
```

```
          </NavHead>
          <NavLink to="/about" activeStyle>
            About
          </NavLink>
          <NavLink to="/teachers" activeStyle>
            Teachers
          </NavLink>
          <NavLink to="/students" activeStyle>
            Students
          </NavLink>
            <NavLink to="/sign-in" activeStyle>
              Sign In
            </NavLink>



        </NavMenu>
      </Nav>
      </>
);
};
```

NavbarElements.js

```
import { FaBars } from "react-icons/fa";
import { NavLink as Link } from "react-router-dom";
import styled from "styled-components";

export const Nav = styled.nav`
background: #191654;
color: #43C6AC;
height: 85px;
display: flex;
justify-content: space-between;

padding: 0.2rem calc((100vw - 1000px) / 2);
z-index: 12;
`;
```

```
export const NavHead = styled(Link)`
color: #43C6AC;
display: flex;
align-items: center;
text-decoration: none;
font-family: Arvo;
padding: 0 3rem;
height: 100%;
font-weight: 1000;
font-size: 35px;
&:focus, &:hover, &:visited, &:link, &:active {
        text-decoration: none;
    }
`
export const NavLink = styled(Link)`
color: #f5f5f5;
display: flex;
align-items: center;
text-decoration: none;
padding: 0 1rem;
height: 100%;
font-family: Ubuntu;
cursor: pointer;
&:focus, &:hover, &:visited, &:link, &:active {
        text-decoration: none;
    }
`;

export const Bars = styled(FaBars)`
display: none;
color: #808080;
@media screen and (max-width: 768px) {
          display: block;
          position: absolute;
          top: 0;
          right: 0;
          transform: translate(-100%, 75%);
          font-size: 1.8rem;
          cursor: pointer;
}
```

```
`;

export const NavMenu = styled.div`
display: flex;
align-items: center;
margin-right: -24px;
/* Second Nav */
/* margin-right: 24px; */
/* Third Nav */
/* width: 100vw;
white-space: nowrap; */
@media screen and (max-width: 768px) {
        display: none;
}
`;
```

REFERENCES

During the course of this project the following materials were referenced:

1. https://reactjs.org/docs/getting-started.html
2. https://flask.palletsprojects.com/en/2.2.x/