

AN INTERNSHIP PROJECT REPORT

on

ONLINE EXAMINATION PLATFORM

Submitted in partial fulfillment of the award of the degree

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by

**MANYAM SRIDEVI
(21025A0570)**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY COLLEGE OF ENGINEERING KAKINADA
Jawaharlal Nehru Technological University Kakinada
Kakinada-533003, Andhra Pradesh, INDIA**

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
UNIVERSITY COLLEGE OF ENGINEERING KAKINADA
Jawaharlal Nehru Technological University Kakinada
Kakinada-533003, Andhra Pradesh, INDIA



CERTIFICATE

This is to certify that this project report entitled “**ONLINE EXAMINATION PLATFORM**” is a bonafide record of the work being submitted by **MANYAM SRIDEVI** bearing the roll number **21025A0570**, in the partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in **COMPUTER SCIENCE AND ENGINEERING** to the **UCEK(A), JNTUK**, Kakinada, Andhra Pradesh, India. It has been found satisfactory and hereby found satisfactory and hereby approved for submission.

Signature of Head of the Department

Dr. O. Srinivasa Rao


Professor & HOD

Department of CSE


UCEK (A)

JNTU KAKINADA

INTERNSHIP CERTIFICATE



Employment Express
Infinite Possibilities




APSCHE
ANDHRA PRADESH STATE COUNCIL OF HIGHER EDUCATION
ఆంధ్ర ప్రదేశ్ హై విద్యా కౌన్సిల్

VIRTUAL INTERNSHIP COMPLETION CERTIFICATE


This is to certify that

Sridevi Manyam


has successfully completed 6 weeks
Full Stack Python Virtual Internship
During **Oct-Nov 2022**




Dr. Ankit Sharma
Director, E2V Academy




Certificate Id: **EMPIFS003274**



Prof. Y. Nazeer Ahammed
Secretary, AP State Council of
Higher Education



Knowledge Partner



Microsoft
Technology
Associate

ABSTRACT

"Study Sphere" is an advanced online examination platform designed to streamline the assessment process for educational institutions. This innovative system features two distinct logins catering to administrators and students, each offering specialized functionalities.

For administrators, the platform presents a user-friendly interface with three key tabs. The "Student Details" tab empowers administrators to effortlessly create new student logins and access existing student information. Meanwhile, the "Results" tab provides a comprehensive overview of pass or fail details for every student. The system also includes a dedicated "Logout" tab, ensuring secure session management for administrators.

Students, on the other hand, experience a seamless assessment journey. After a thorough review of rules and regulations, students engage in a photo capture process using their device's camera. The platform dynamically checks for the availability of essential peripherals like the microphone and camera. Once the prerequisites are confirmed, students navigate through a series of multiple-choice questions. Upon completion, a straightforward submission process triggers the display of results. A convenient logout button concludes the student's session.

Security and privacy are paramount considerations, with stringent measures implemented to safeguard user data. The documentation provides an in-depth exploration of the system's technical requirements, troubleshooting guidelines, and outlines potential future enhancements.

"Study Sphere" stands as a testament to the evolution of online examination platforms, offering an intuitive and secure environment for administrators and students alike. This documentation serves as a comprehensive guide, empowering users to harness the full potential of this cutting-edge educational tool.

CONTENTS

Page No.

Chapter 1 INTRODUCTION

- 1.1** Problem Statement 1-4
- 1.2** Why to use?
- 1.3** Objective

Chapter 2 SYSTEM DESIGN

- 2.1** Software Requirements
- 2.2** Hardware Requirements
- 2.3** System Requirements

5-7

Chapter 3 IMPLEMENTATION

- 3.1** Modules
- 3.2** Module description
- 3.3** Introduction of technologies used
- 3.4** Coding

8-23

Chapter 4 OUTPUT SCREENSHOTS

26-21

Chapter 5 CONCLUSION

32-33

NOMENCLATURE

- **UI** – User Interface
- **HTML** – Hypertext Markup Language
- **CSS** – Cascading Style Sheets
- **JS** – Java Script
- **FLASK** – Python web framework that provides useful tools and features.
- **HTTP** – Hypertext Transfer Protocol
- **Bootstrap** – Popular CSS Framework for developing responsive and mobile-first websites
- **OS** – Operating Systems

CHAPTER I

INTRODUCTION

INTRODUCTION

In the dynamic landscape of education, the demand for efficient and secure assessment platforms has spurred the development of "Study Sphere" – an innovative online examination system designed to cater to the evolving needs of educational institutions. Utilizing Flask, Python, and a MySQL database within the XAMPP server environment, this platform stands at the forefront of technological advancements in online education.

"Study Sphere" is characterized by a user-centric approach, providing an intuitive interface for both administrators and students. Administrators, equipped with a dedicated dashboard, can seamlessly manage student details, create new logins, and gain insights into student performance through the "Results" tab. This empowerment streamlines administrative processes, facilitating informed decision-making.

For students, the assessment process is carefully orchestrated, beginning with a thorough review of rules and regulations. The platform ensures a secure environment for photo capture, validates the availability of essential peripherals, and guides students through a series of multiple-choice questions. The result is a streamlined and user-friendly experience that redefines online assessments.

This comprehensive guide aims to provide insights into the technical implementation, user workflows, and the thoughtful design of "Study Sphere." Whether you are an administrator seeking efficient assessment tools or a student navigating the examination process, "Study Sphere" is poised to redefine your online education experience.

1.1 PROBLEM STATEMENT

- Traditional examination methods, primarily paper-based, often result in time-consuming processes for creating, distributing, and managing exams. There is a pressing need for a more streamlined and automated approach to enhance overall efficiency.
- Existing online examination platforms often suffer from a lack of user-friendly interfaces. Both administrators and students encounter difficulties navigating through the system. Improving the accessibility of features such as result viewing, student management, and assessment procedures is crucial to cater to diverse user needs.
- The growing trend towards online assessments brings forth concerns related to the security and integrity of examination data. Challenges such as unauthorized access, data breaches, and insufficient privacy measures highlight the necessity for a robust and secure online examination platform.

1.2 OBJECTIVE OF THE PROJECT

The objective of the "Study Sphere" project is to enhance the efficiency of traditional examination processes by developing an advanced online platform. This includes creating a user-friendly experience for administrators and students, ensuring robust security measures, leveraging modern technologies such as Flask, Python, and MySQL, and designing a scalable solution adaptable to the evolving needs of educational institutions. The project aims to streamline assessments, foster technological advancement, and establish a secure, user-centric, and sustainable online examination environment.

ADVANTAGES

1. **Efficiency Gains:** Streamlined and automated examination processes reduce administrative burdens, allowing educational institutions to conduct assessments more efficiently.
2. **User-Centric Design:** An intuitive and user-friendly interface benefits both administrators and students, enhancing the overall experience and ease of navigation within the platform.
3. **Enhanced Security Measures:** Robust security protocols ensure the confidentiality and integrity of examination data, addressing concerns related to unauthorized access and data breaches.
4. **Technological Advancement:** Leveraging modern technologies such as Flask, Python, and MySQL within the XAMPP server environment ensures a technologically advanced and responsive online examination solution.
5. **Scalability and Adaptability:** The platform is designed to scale seamlessly, accommodating the evolving needs of educational institutions and ensuring long-term relevance in diverse educational environments.

CHAPTER II

SYSTEM DESIGN

DESIGN OF THE SYSTEM

2.1 SOFTWARE REQUIREMENTS

Platform	Platform Independent
The Operating System	Windows 7 & above
Framework	Flask
Front-End Tool	Google Chrome
Database	MySQL

2.2 HARDWARE REQUIREMENTS

Processor	Intel Pentium IV 2.9 GHz Other
RAM	Minimum 4 GB
Graphics	Integrated graphics card
Hard Disk	Minimum 300 GB

2.3 SYSTEM REQUIREMENTS

"Study Sphere" demands a robust technological foundation to deliver an efficient and secure online examination platform. The web server component, anchored by Apache (minimum version 2.4), is pivotal for hosting the Flask application, facilitating smooth communication between the backend and the user interface. Simultaneously, the MySQL database server (minimum version 5.7) forms the backbone for managing student information, assessment results, and other critical data. Python 3.6 or higher, along with the Flask web framework (minimum version 2.0), propels the backend development, ensuring the core functionalities operate seamlessly.

On the frontend, the platform relies on standard web technologies—HTML, CSS, and JavaScript—ensuring a user-friendly and responsive interface across modern browsers. The integration of the MySQL Connector library for Python (minimum version 8.0) ensures efficient and secure communication with the MySQL database, enhancing the reliability of data retrieval and storage. The image processing tasks, fundamental for secure user authentication through photo capture, are facilitated by the OpenCV library (minimum version 4.0), adding a layer of sophistication to identity verification.

Functional Requirements:

The functional requirements of "Study Sphere" include secure user authentication for administrators and students, streamlined student registration and profile management, and a cohesive assessment workflow. This encompasses rules review, secure photo capture, multiple-choice questions, and result submission, ensuring a compact yet comprehensive online examination system prioritizing security and user efficiency.

CHAPTER III

IMPLEMENTATION

IMPLEMENTATION

During the implementation phase, the development team will translate the design specifications into actual code. They will follow software development best practices, such as writing clean and modular code, conducting unit tests, and ensuring proper documentation. The modules and implementation process play a crucial role in bringing the system design to life and creating a functional and reliable scheduling platform for doctors and patients.

3.1 MODULES

Creating a comprehensive set of modules in code requires defining classes or functions for each module. The few main modules are as listed below and the usage of the modules is as given below

- 1.flask
- 2.Flask_mysqlldb
- 3.Cv2



```
app.py > ...
1  import math
2  from flask import Flask, render_template, request, jsonify, redirect, url_for, Response, flash
3  import os
4  from flask_mysqlldb import MySQL
5  import warnings
6  import utils
7  import time
8  import cv2
```

Fig - 3.1 Modules

3.2 MODULES DESCRIPTION

1. FLASK :

Flask is a lightweight and extensible web framework for Python. It is designed to be easy to use and flexible, making it a popular choice for building web applications and APIs. Flask follows the WSGI (Web Server Gateway Interface) standard, allowing it to work with various web servers.

Here are some key features and components of the Flask framework:

Routing: Flask allows you to define routes, which map URLs to Python functions. This makes it easy to create different endpoints for handling various HTTP methods (GET, POST, etc.).

Templates: Flask supports template engines, such as Jinja2, which enable you to separate your Python code from HTML. This helps maintain clean and readable code for both server-side and client-side components.

Extensions: Flask has a modular architecture that supports extensions for adding additional functionality. There are numerous extensions available for tasks such as authentication, database integration, and more.

Integrated Development Server: Flask comes with a built-in development server that simplifies the process of testing and debugging your application during development.

2. MySQL:

The MySQL module in "Study Sphere" facilitates seamless interaction with the MySQL database. It manages database connections, executes queries, and handles data retrieval. This module is crucial for storing and retrieving student information, assessment results, and other relevant data. It ensures the integrity and security of the database, contributing to the robustness of the overall system.

3.3 INTRODUCTION OF TECHNOLOGIES USED

- **HTML**

The Hyper Text Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

- **CSS**

CSS stands for Cascading Style Sheets. It is a style sheet language which is used to describe the look and formatting of a document written in markup language. It provides an additional feature to HTML. It is generally used with HTML to change the style of web pages and user interfaces. It can also be used with any kind of XML documents including plain XML, SVG and XUL. CSS is used along with HTML and JavaScript in most websites to create user interfaces for web applications and user interfaces for many mobile applications.

- **JavaScript**

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as LiveScript, but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name LiveScript. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers. JavaScript has a vast

ecosystem of third-party libraries and frameworks that simplify API integration. Libraries like Axios, jQuery, and Fetch API provide convenient methods and utilities for making API requests, handling responses, and managing API authentication.

Overall, JavaScript's versatility and capabilities make it indispensable for linking an API to a web application. It enables data retrieval, manipulation, user interaction, and seamless integration of dynamic content, empowering developers to create robust and interactive web applications that leverage the power of external APIs.

3.4 CODING

The following shows the code which is used to display output and to design a pleasant looking website.

HTML CODE :

Index.html

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login</title>
  <link rel="stylesheet" type="text/css" href="{{url_for('static',
filename='css/login.css')}}">
  <link rel="stylesheet" type="text/css" href="{{url_for('static',
filename='css/all.min.css')}}">
  <link href="https://fonts.googleapis.com/css?family=Poppins:600&display=swap"
rel="stylesheet">
```

```

<script src="https://kit.fontawesome.com/a81368914c.js"></script>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel='Shortcut Icon' type='image/x-icon' href="{{url_for('static',
filename='img/title.png')}}">
</head>

<body>
  
  <div class="container">
    <div class="img">
      
    </div>
    <div class="login-content">
      <form action="{{url_for('login')}}" method="post">
        
        <h2 class="title">Welcome</h2>
        <div class="input-div one">
          <div class="i">
            <i class="fas fa-user"></i>
          </div>
          <div class="div">
            <h5>Username</h5>
            <input type="text" class="input" name="username" id="username" required>
          </div>
        </div>
        <div class="input-div pass">
          <div class="i">
            <i class="fas fa-lock"></i>
          </div>
          <div class="div">
            <h5>Password</h5>
            <input type="password" class="input" name="password" id="password"
required>
          </div>
        </div>
        <input type="submit" class="btn" value="login">
      </form>
    </div>
  </div>
  <script type="text/javascript" src="{{url_for('static', filename='js/login.js')}}"></script>
</body>

</html>

```

CSS CODE:

Style.css

```
@import url(https://fonts.googleapis.com/css?family=Droid+Sans);
@import url(http://weloveiconfonts.com/api/?family=fontawesome);
/* fontawesome */
[class*="fontawesome-"]:before {
    font-family: 'FontAwesome', sans-serif;
}
* {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}
/*----- utility classes -----*/
.professional{
    align-items: center;
}
/*its also known as clearfix*/
.group:before,
.group:after {
    content: "";
    display: table;
}
.group:after {
    clear: both;
}
.group {
    zoom: 1; /*For IE 6/7 (trigger hasLayout) */
}

body {
    background: #e2e2e2;
    font-family: 'Droid Sans', sans-serif;
    line-height: 1;
    font-size: 16px;
}
nav{
    background: #5f9ea0;
    line-height: 2;
```

```

}

nav span{
    font-size: 30px;
    color: #f0ecec;
    margin-left: 30px;
}

.wrapper {

}

.pricing-table {
    width: 40%;
    margin: 50px auto;
    text-align: center;
    padding: 10px;
    padding-right: 0;
}

.pricing-table .heading{
    color: #9C9E9F;
    text-transform: uppercase;
    font-size: 1.3rem;
    margin-bottom: 4rem;
}

.block{
    width: 30%;
    margin: 0 15px;
    overflow: hidden;
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    border-radius: 5px;
    /*    border: 1px solid red;*/
}

/*Shared properties*/
.pt-footer{
    color: #e2e2e2;
    text-transform: capitalize;
    line-height: 2.5;
    position: relative;
}

.title{
    font-size: 30px;
    color: #e2e2e2;
    text-transform: capitalize;
    line-height: 2;
    position: relative;
}

```

```

.pt-footer a{
  text-decoration: none;
  color: #e2e2e2;
  font-size: 20px;
}
.content{
  position: relative;
  color: #e2e2e2;
  padding: 20px 0 10px 0;
}
/*arrow creation*/
.content:after, .content:before, .pt-footer:before, .pt-footer:after {
  top: 100%;
  left: 50%;
  border: solid transparent;
  content: " ";
  height: 0;
  width: 0;
  position: absolute;
  pointer-events: none;
}
.pt-footer:after, .pt-footer:before{
  top:0;
}
.content:after, .pt-footer:after {
  border-color: rgba(136, 183, 213, 0);
  border-width: 5px;
  margin-left: -5px;
}
/*arrow creation*/

```

```

.features{
  list-style-type: none;
  background: #FFFFFF;
  text-align: left;
  color: #9C9C9C;
  padding: 30px 22%;
  font-size: 20px;
}
.features li{
  padding: 15px 0;
  width: 100%;
}
.features li span{
  padding-right: 0.4rem;
}

```

```

.pt-footer{
    font-size: 0.95rem;
    text-transform: capitalize;
}
/*PROFESSIONAL*/
.professional .title{
    background: #5f9ea0;
}
.professional .content,.professional .pt-footer{
    background: #5f9ea0;
}
.professional .content:after{
    border-top-color: #5f9ea0;
}
.professional .pt-footer:after{
    border-top-color: #FFFFFF;
}
/*BUSINESS*/

```

FLASK CODE:

app.py

```

import math
from flask import Flask, render_template, request, jsonify, redirect, url_for, Response, flash
import os
from flask_mysql import MySQL
import warnings
import utils
import time
import cv2

#variables
studentInfo=None
camera=None
profileName=None

#Flak's Application Configuration
warnings.filterwarnings("ignore")
app = Flask(__name__, template_folder='templates', static_folder='static')
app.secret_key = 'xyz'
os.path.dirname("../templates")

#Flak's Database Configuration
app.config['MYSQL_HOST'] = 'localhost'
app.config['MYSQL_USER'] = 'root'

```

```
app.config['MYSQL_PASSWORD'] = ''
app.config['MYSQL_DB'] = 'studysphere'
mysql = MySQL(app)
```

#Function to show face detection's Rectangle in Face Input Page

```
def capture_by_frames():
    global camera
    utils.cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
    while True:
        success, frame = utils.cap.read() # read the camera frame
        detector=cv2.CascadeClassifier('Haarcascades/haarcascade_frontalface_default.xml')
        faces=detector.detectMultiScale(frame,1.2,6)
        #Draw the rectangle around each face
        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 3)
        ret, buffer = cv2.imencode('.jpg', frame)
        frame = buffer.tobytes()
        yield (b'--frame\r\n'
               b'Content-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
```

#Function to run Cheat Detection when we start run the Application

```
@app.before_request
def start_loop():
    pass
```

#Login Related

```
@app.route('/')
def main():
    return render_template('login.html')
```

```
@app.route('/login', methods=['POST'])
def login():
    global studentInfo
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        cur = mysql.connection.cursor()
        cur.execute("SELECT * FROM students where Email='" + username + "' and Password='" +
password + "'")
        data = cur.fetchone()
        if data is None:
            flash('Your Email or Password is incorrect, try again.', category='error')
            return redirect(url_for('main'))
        else:
            id, name, email,password, role = data
            studentInfo={ "Id": id, "Name": name, "Email": email, "Password": password}
            if role == 'STUDENT':
```



```

        utils.Student_Name = name
        return redirect(url_for('rules'))
    else:
        return redirect(url_for('adminStudents'))

```

```

@app.route('/logout')
def logout():
    return render_template('login.html')

```

```

#Student Related
@app.route('/rules')
def rules():
    return render_template('ExamRules.html')

```

```

@app.route('/faceInput')
def faceInput():
    return render_template('ExamFaceInput.html')

```

```

@app.route('/video_capture')
def video_capture():
    return Response(capture_by_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')

```

```

@app.route('/saveFaceInput')
def saveFaceInput():
    global profileName
    if utils.cap.isOpened():
        utils.cap.release()
    cam = cv2.VideoCapture(0)
    success, frame = cam.read() # read the camera frame
    profileName=f"{studentInfo['Name']}_{utils.get_resultId():03}" + "Profile.jpg"
    cv2.imwrite(profileName,frame)
    utils.move_file_to_output_folder(profileName,'Profiles')
    cam.release()
    return redirect(url_for('confirmFaceInput'))

```

```

@app.route('/confirmFaceInput')
def confirmFaceInput():
    profile = profileName
    # utils.fr.encode_faces()
    return render_template('ExamConfirmFaceInput.html', profile = profile)

```

```

@app.route('/systemCheck')
def systemCheck():
    return render_template('ExamSystemCheck.html')

```

```

@app.route('/systemCheck', methods=["POST"])
def systemCheckRoute():

```

```

    if request.method == 'POST':
        examData = request.json
        output = 'exam'
        if 'Not available' in examData['input'].split(';'): output = 'systemCheckError'
    return jsonify({"output": output})

```

```

@app.route('/systemCheckError')
def systemCheckError():
    return render_template('ExamSystemCheckError.html')

```

```

@app.route('/exam')
def exam():
    return render_template('Exam.html')

```

```

@app.route('/exam', methods=["POST"])
def examAction():
    link = ''
    if request.method == 'POST':
        examData = request.json
        if(examData['input']!=''):
            totalMark= math.floor(float(examData['input'])* 6.6667)
            if totalMark < 50:
                status="Fail"
                link = 'showResultFail'
            else:
                status="Pass"
                link = 'showResultPass'
            utils.write_json({
                "Id": utils.get_resultId(),
                "Name": studentInfo['Name'],
                "TotalMark": totalMark,
                "Status": status,
                "Date": time.strftime("%Y-%m-%d", time.localtime(time.time())),
                "StId": studentInfo['Id'],
                "Link" : profileName
            }, "result.json")
            resultStatus=
studentInfo['Name']+';'+str(totalMark)+';'+status+';'+time.strftime("%Y-%m-%d",
time.localtime(time.time()))
            else:
                resultStatus=''
    return jsonify({"output": resultStatus, "link": link})

```

```

@app.route('/showResultPass/<result_status>')
def showResultPass(result_status):
    return render_template('ExamResultPass.html', result_status=result_status)

```

```

@app.route('/showResultFail/<result_status>')
def showResultFail(result_status):
    return render_template('ExamResultFail.html', result_status=result_status)

#Admin Related
@app.route('/adminResults')
def adminResults():
    results = utils.getResults()
    return render_template('Results.html', results=results)

@app.route('/adminStudents')
def adminStudents():
    cur = mysql.connection.cursor()
    cur.execute("SELECT * FROM students where Role='STUDENT'")
    data = cur.fetchall()
    cur.close()
    return render_template('Students.html', students=data)

@app.route('/insertStudent', methods=['POST'])
def insertStudent():
    if request.method == "POST":
        name = request.form['username']
        email = request.form['email']
        password = request.form['password']
        cur = mysql.connection.cursor()
        cur.execute("INSERT INTO students (Name, Email, Password, Role) VALUES (%s, %s, %s, %s)",
(name, email, password, 'STUDENT'))
        mysql.connection.commit()
        return redirect(url_for('adminStudents'))

@app.route('/deleteStudent/<string:stdId>', methods=['GET'])
def deleteStudent(stdId):
    flash("Record Has Been Deleted Successfully")
    cur = mysql.connection.cursor()
    cur.execute("DELETE FROM students WHERE ID=%s", (stdId,))
    mysql.connection.commit()
    return redirect(url_for('adminStudents'))

@app.route('/updateStudent', methods=['POST', 'GET'])
def updateStudent():
    if request.method == 'POST':
        id_data = request.form['id']
        name = request.form['name']
        email = request.form['email']
        password = request.form['password']
        cur = mysql.connection.cursor()
        cur.execute("""

```

```
        UPDATE students
        SET Name=%s, Email=%s, Password=%s
        WHERE ID=%s
        """, (name, email, password, id_data))
    mysql.connection.commit()
    return redirect(url_for('adminStudents'))

if __name__ == '__main__':
    app.run(debug=True)
```

Source code reference: <https://github.com/SrideviManyam/Study-Sphere>

RUNNING THE ABOVE APPLICATION USING TERMINAL:

Running a Flask application in the VS Code terminal involves a series of steps. Here's a step-by-step procedure:

1. Open vs code

Open Visual Studio Code and ensure that you have your Flask project opened in the workspace.

2. Set Up a Virtual Environment

While not strictly necessary, it's often a good practice to create a virtual environment for your Flask project. This helps manage dependencies and keeps your project isolated.

In the terminal, navigate to your project directory and create a virtual environment:

```
cd path/to/your/flask/app
python -m venv venv
```

Activate the virtual environment:

- On Windows:

```
bash
.\venv\Scripts\activate
```

- On macOS/Linux:

```
bash
source venv/bin/activate
```

3. Install Flask

With the virtual environment activated, install Flask using pip:

```
bash
pip install Flask
```

4. Set Flask App and Environment Variables

In the VS Code terminal, set the FLASK_APP variable to the name of your Flask application file (assuming it's app.py for this example):

```
bash
```

```
export FLASK_APP=app.py
```

On Windows:

```
bash
```

```
set FLASK_APP=app.py
```

Optionally, set the environment to development for automatic reloading:

```
bash
```

```
export FLASK_ENV=development
```

On Windows:

```
bash
```

```
set FLASK_ENV=development
```

5. Run the Flask Application

Now, you can run your Flask application using the flask run command:

```
bash
```

```
flask run
```

This command starts the development server. You should see output in the terminal indicating that the server is running.

6. Access the Application

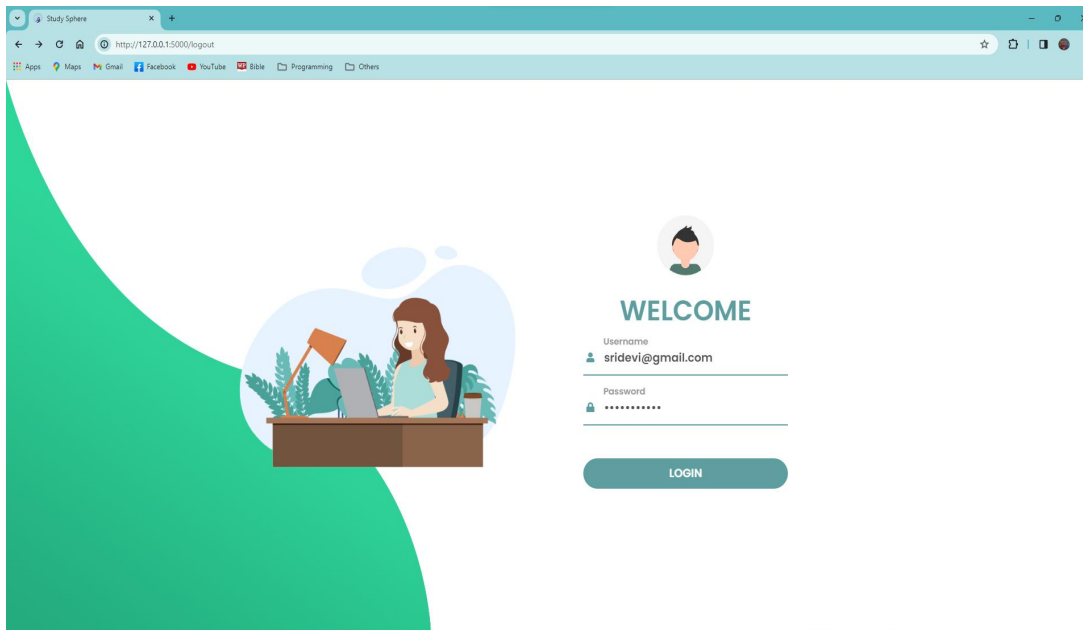
Open your web browser and navigate to `http://127.0.0.1:5000/` (or the address provided in the terminal). You should see your Flask application running.

CHAPTER IV

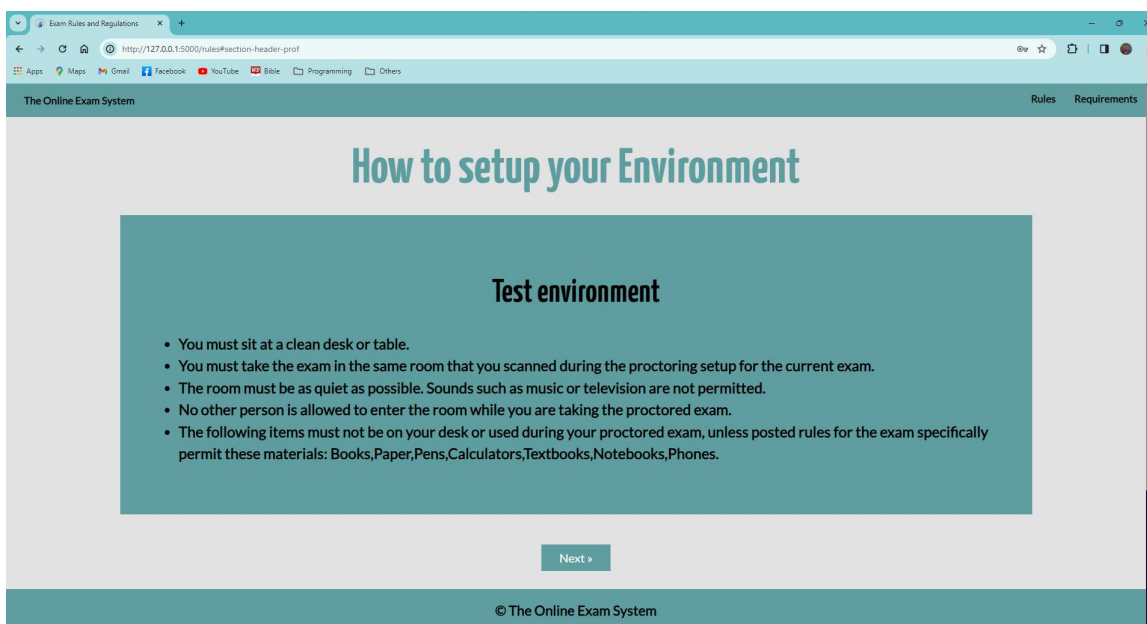
OUTPUT SCREENSHOTS

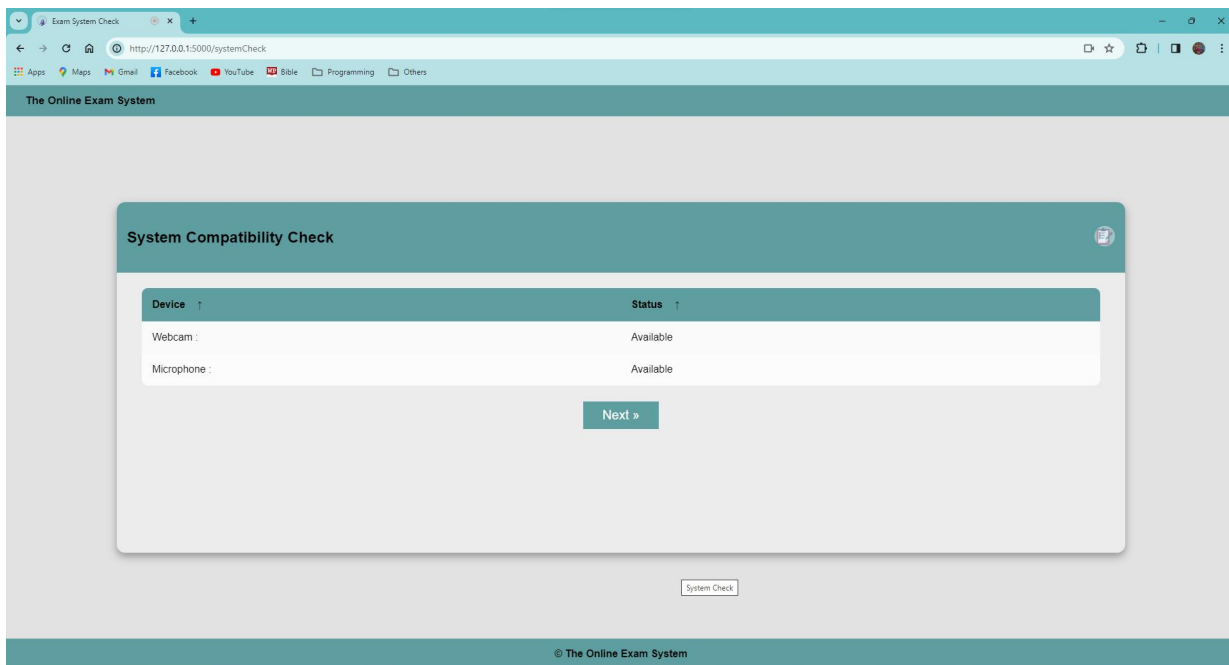
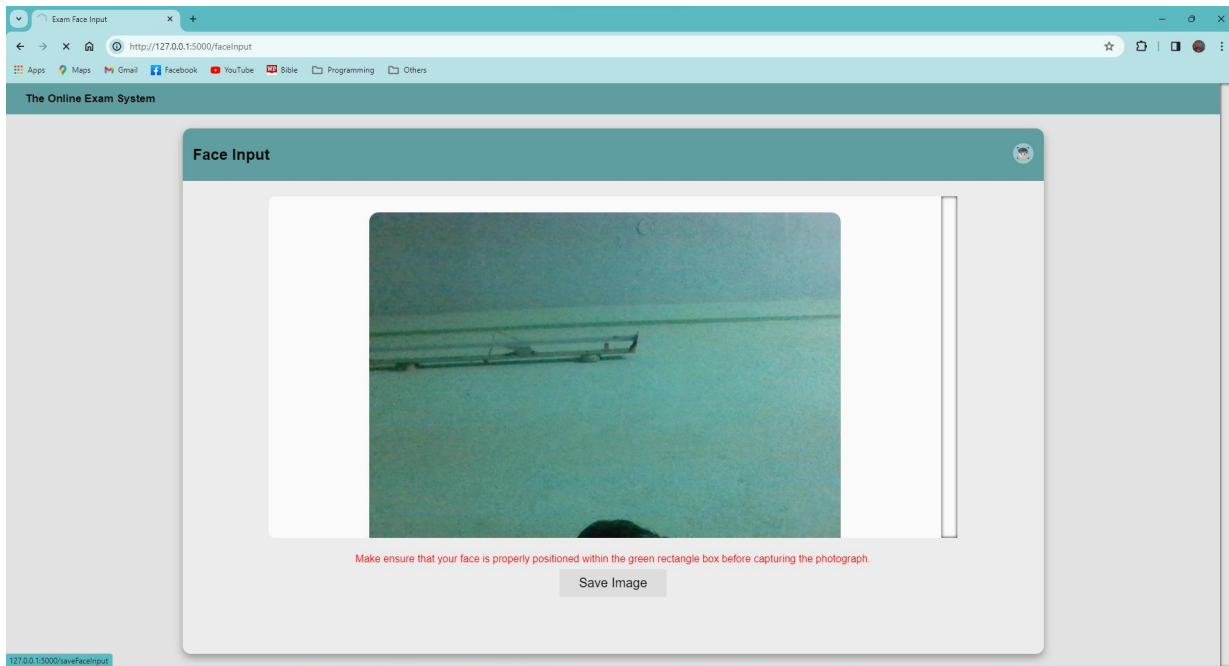
SCREENSHOTS OF OUTPUT

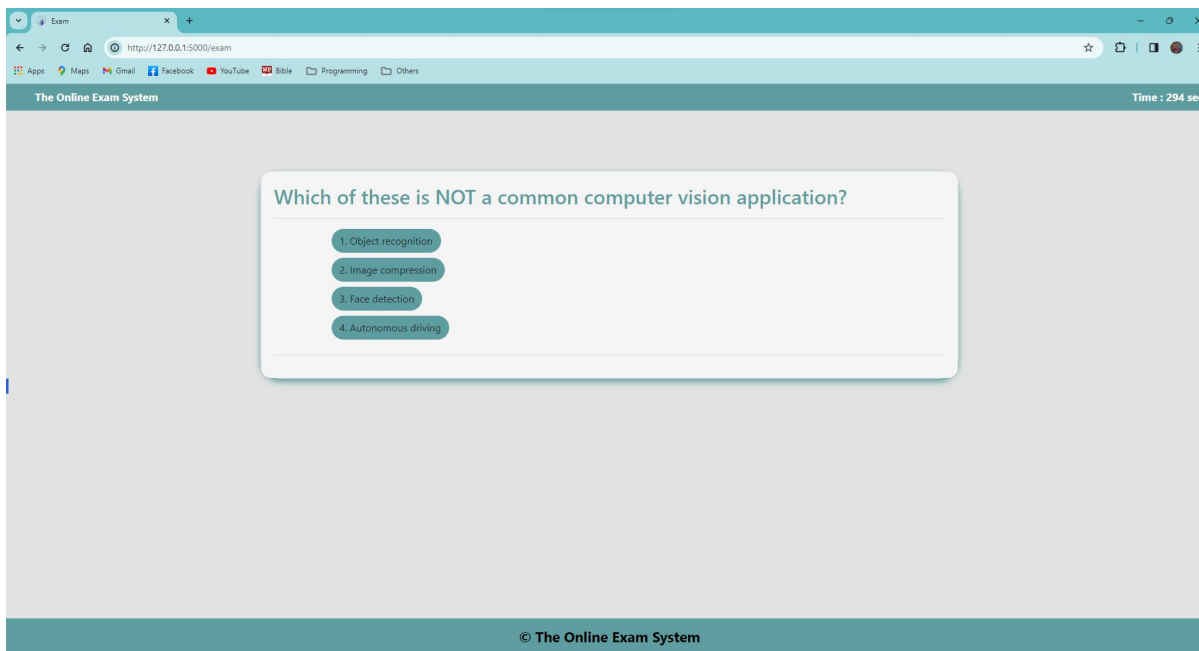
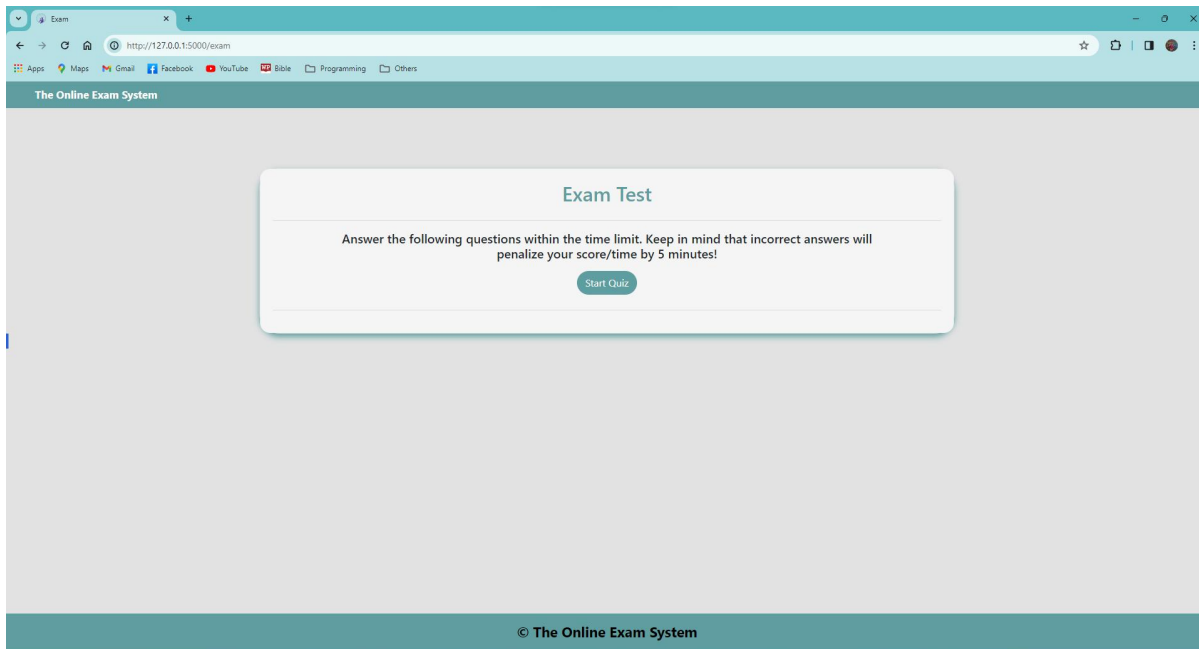
Home Screen :

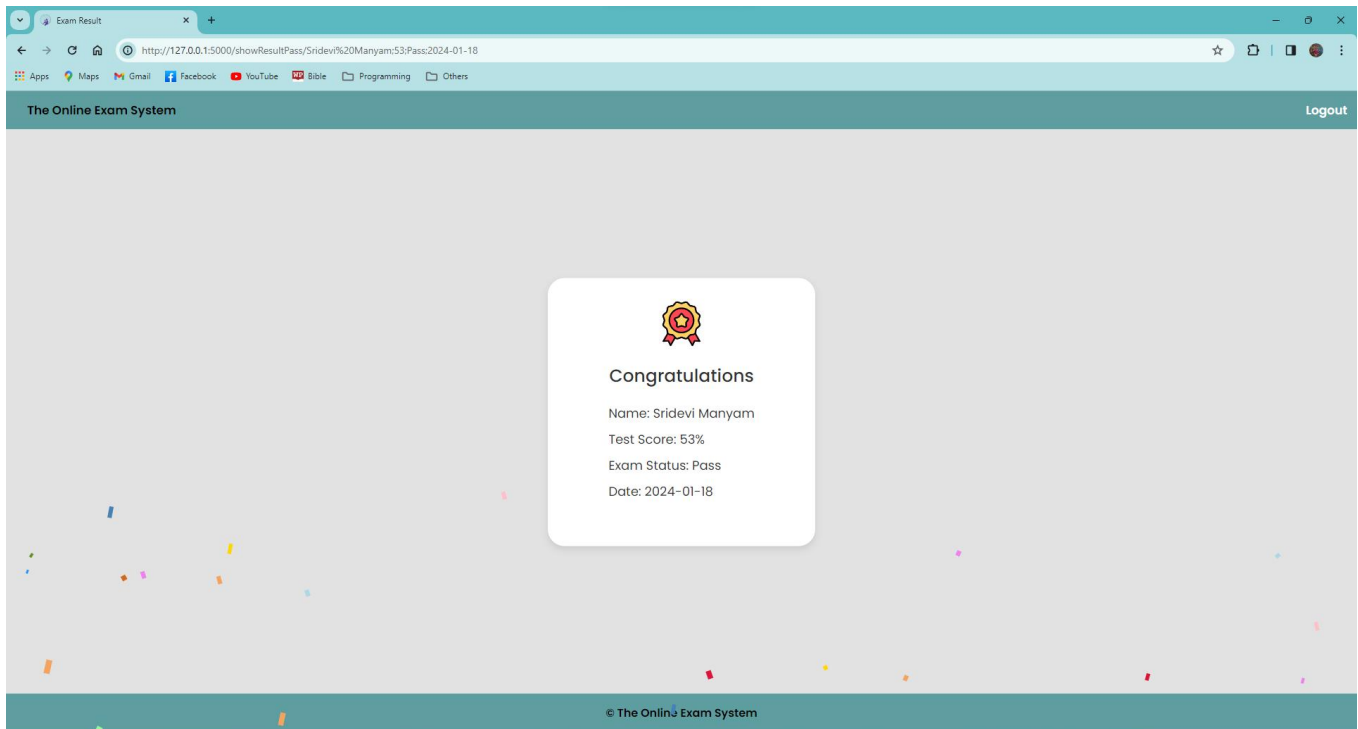


Student Login Pages :

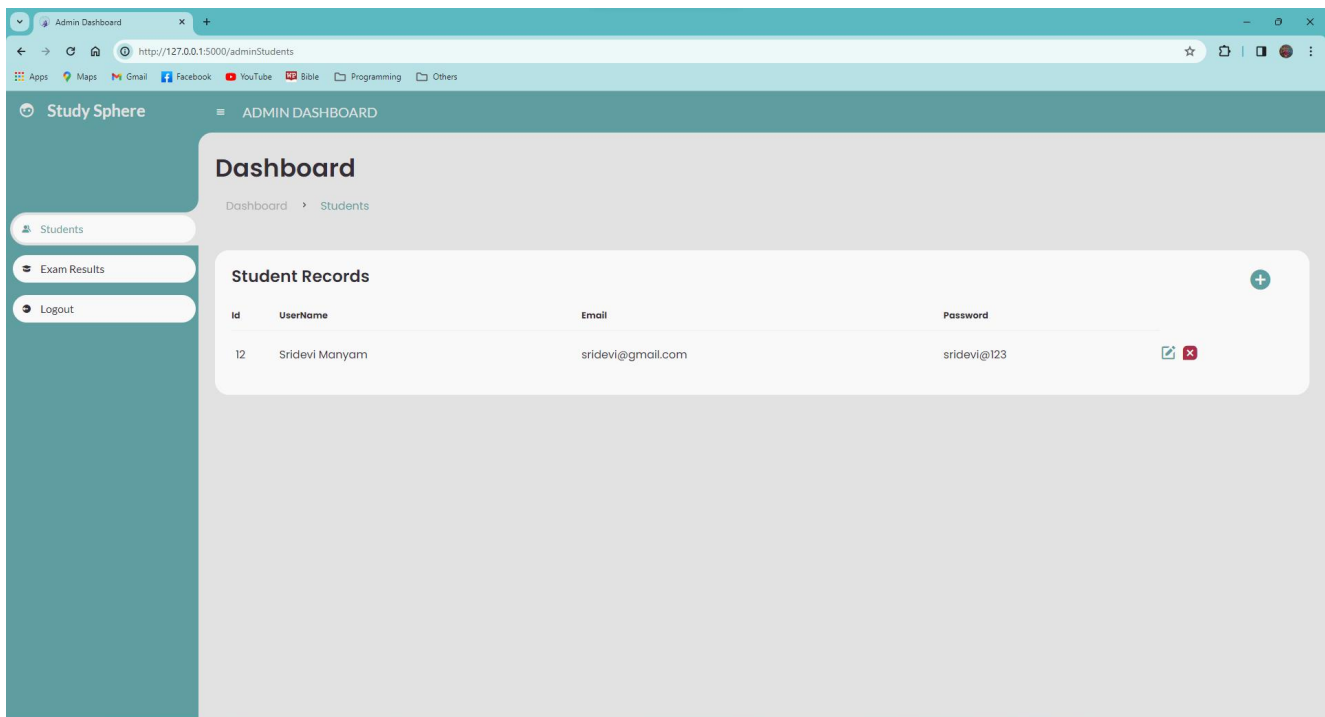








Administrator Login Pages :



Admin Dashboard

http://127.0.0.1:5000/adminResults

AppsMapsGmailFacebookYouTubeBibleProgrammingOthers

Study Sphere

ADMIN DASHBOARD

Students

Exam Results

Logout

Dashboard

Exam Results

Exam Result Records

Name	Date	Status	Total Mark
Sridevi Manyam	2024-01-18	Pass	53%

CHAPTER V

CONCLUSION

CONCLUSION

In a rapidly evolving educational landscape, "Study Sphere" not only addresses the inefficiencies in traditional examination processes but also introduces a paradigm shift in how educational assessments are conducted. By incorporating a sophisticated authentication module, the platform ensures a secure and reliable identification process, adding an extra layer of integrity to the examination system.

The user-centric design, characterized by an intuitive interface and streamlined workflows, not only simplifies administrative tasks but also enhances the overall experience for students. The assessment module, with its careful consideration of rules, secure photo capture, and submission processes, reflects a commitment to creating a user-friendly and transparent examination environment.

Furthermore, the scalability and adaptability of "Study Sphere" make it well-equipped to meet the evolving needs of educational institutions. The platform's ability to seamlessly integrate modern technologies not only positions it as a technologically advanced solution but also future-proofs it against emerging trends and requirements.

In the grander scope of online education, "Study Sphere" is more than just a platform; it represents a forward-looking approach to assessments. Its success lies not only in its technical capabilities but in its potential to transform the educational experience, making examinations more accessible, secure, and aligned with the demands of the digital age. As we move forward, "Study Sphere" serves as a beacon for innovation in educational technology, influencing how assessments are conceived and conducted.