PROJECT REVIEW AND EVALUATION SYSTEM

A

Mini Project Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

By

<NARSINI SHRAVANI><1602-20-737-039>

<THODETI SHIVACHARAN><1602-20-737-038>

<ZOHA TABASSUM><1602-20-737-053>

Department of Information Technology

Vasavi College of Engineering (Autonomous)

VASAVI COLLEGE OF ENGINEERING (AUTONOMOUS)

(AFFILIATED TO OSMANIA UNIVERSITY)

HYDERABAD - 500 030

Department of Information Technology

DECLARATION BY CANDIDATE

We, **< NARSINI SHRAVANI >, <THODETI SHIVACHARAN >, <ZOHA TABASSUM>,** bearing hallticket number, **<1602-20-737-039>,<1602-20-737-038>,<1602-20-737-053>**
hereby declare that the project report entitled **<"PROJECT REVIEW AND EVALUATION SYSTEM">** Department of Information Technology, Vasavi College of Engineering, Hyderabad, is submitted in partial fulfillment of the requirement for the award of the degree of **Bachelor of Engineering** in **Information Technology**

This is a record of bonafide work carried out by me and the results embodied in this project report has not been submitted to any other university or institute for the award of any other degree or diploma.

<NARSINI SHRAVANI><1602-20-737-039>
<THODETI SHIVACHARAN><1602-20-737-038>
<ZOHA TABASSUM><1602-20-737-053>

VASAVI COLLEGE OF ENGINEERING (AUTONOMOUS)

(AFFILIATED TO OSMANIA UNIVERSITY)

HYDERABAD - 500 030

Department of Information Technology

BONAFIDE CERTIFICATE

This is to certify that the project entitled "**PROJECT REVIEW AND EVALUATION SYSTEM**" being submitted by SHRAVANI,SHIVACHARAN,ZOHA bearing 1602-20-737-039,1602-20-737-038,1602-20-737-053 in partial fulfillment of the requirements for the completionof MINI PROJECT of Bachelor of Engineering in Information Technology is a record of bonafide work carried out by them under my guidance.

Internal Guide	External Examiner	Dr. K Ram Mohan Rao
Sireesha		HOD, IT

ACKNOWLEDGEMENT

We thank the department of INFORMATION TECHNOLOGY, for introducing the subject “Mini Project-2” in BE fifth semester.

We would also like to show our appreciation to our Honorable principal, Dr S V Ramana sir, our HOD K. Ram Mohan Rao for supporting us and our mini project lecturer, Mrs Sireesha mam, for letting us properly understand the process of doing a project and for providing valuable insight and expertise that has greatly assisted us in the making of the project.

TABLE OF CONTENTS:

ABSTRACT

Academic Project management whether Mini project or Major project is a major issue which is faced by many educational institutes, the main reason is, there is no automated system followed in any institute. College management/staff gather all the project reports and project sources from students and store them physically in some locations probably libraries. Since future is unpredictable and we cannot expect things to happen smoothly offline so in such cases we need everything to be done online without any problem.

To overcome this practical problem we have come up with a Project approval system. In this system we have separate login and registration for students, internal guides and HOD. Students can enter their batch details, send their abstract to internal guides and HOD. Internal Guides and HOD can either approve or reject a project and can provide suggestions. One of the features is that, if the abstracts or project ideas of one or more groups are matching, then the batches are notified. Similarity of the project titles are checked using NLP and status of each and every batch is also displayed. Feedback can also be given to the students

CHAPTER 1

INTRODUCTION

What is project review and evaluation system?

Project Review and evaluation system is a system which helps to review and evaluate the projects done by the students by their respective guides and hods in every academic year in every colleges.

1.1 PURPOSE

Every undergraduate student will have to deal with submission of projects either in final or in penultimate year in college. Projects play an essential role in enhancing the learning experience for the students, as they involve in real life application of classroom knowledge. So, such an important academic quantity should be dealt with fairly and should be made sure that no mistakes happen when the idea of project is conveyed to the respective faculty. Offline submission surely has its merits of getting direct insights from the teachers, but having an online platform for submission is something that stays at a higher place.

Some students might want to submit their abstract to a faculty at a specific time, but to their bad luck, the faculty may not be available. And other students might choose the same topic as their peers did, and end up colliding with them. Based on such real-life experiences, the creation of Project Evaluation and Review System took place

1.2PRODUCT SCOPE

Projects plays an important role for every students. So inorder to maintain them and evaluate them correctly this project review and evaluation system plays a major role

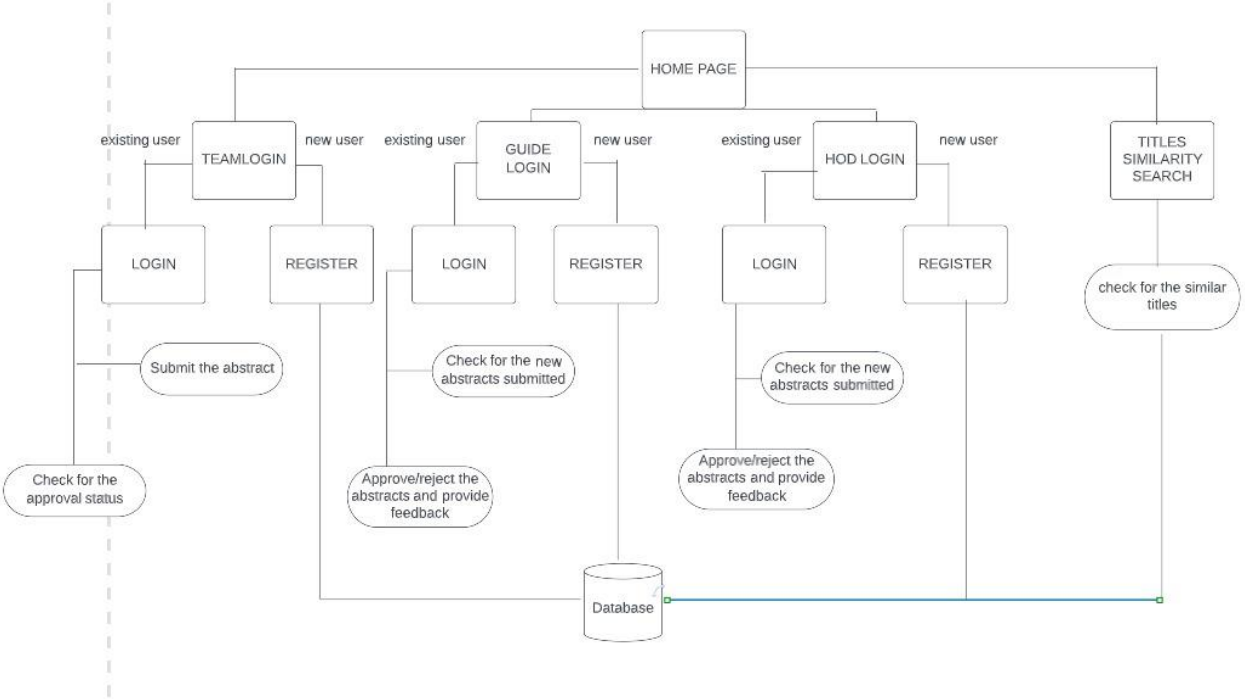
1.3PROBLEM DEFINATION

Project Review and Evaluation Project is a novel idea that aims at solving the real-life problems faced by students and faculty when dealing with the submission of projects at graduation level. The main reason behind the existing inconvenience is there is no automated system followed in any educational institution . To ensure the smooth functioning of project assessment, we need to have an online interface, which consists of all the necessary features.

CHAPTER 3

PROPOSED WORK –

3.1 Architecture



Project Evaluation and Review System has several modules designed for a very different purpose. The platform is divided into four main parts, three out of which, namely Student Login, Guide Login, HOD Login, provides a varied space for students, faculty and HOD respectively. The other part, Title Similarity Search is a feature that is utilized to search for the project abstract.

3.2 Architecture and Technology used –

Technology used –

The tool using which the 'Project Review and evaluation system' was made is VSCODE.

3.4 Module Design Student Registration and Login:

Student Registration helps the students to register their team on the platform by providing appropriate team name and password accordingly. Once the registration is successful, they can login to their account to submit their project abstracts, to view the replies from faculty and HOD and to check the status of their project. HTML, CSS and Flask are used to implement this module[5]. The user details are stored in databases which are created and managed by using MySQL Alchemy.

Guide and HOD Registration and Login:

Guide Registration will follow the similar steps at the student module does. Once the faculty successfully creates an account, they can check for the submitted abstracts. They can download the abstracts by simply clicking on it, and hence the project abstract can be will be available offline as well. After going through the abstract, they will have to update the status, that will be shown on the student team's account. There are two options for the status- Accept, when the project abstract meets the appropriate standards and reject, otherwise. The faculty will be provided space for remarks if they want to give any suggestion to the team. HTML, CSS and Flask are used to implement this module. The user details are stored in databases which are created and managed by using MySQL Alchemy.

Text Similarity Search:

In order to avoid submitting project abstracts with the same name, the students can avail this module to check for similar project abstracts. This module only shows the accepted abstracts. The rejected abstracts will have to be submitted again for approval, in order to be displayed in the results of the search. NLP is used to implement this module. The search results are the part of records from the stored database of accepted abstracts. The databases are created and are managed by MySQL Alchemy.

CODE:

```
import imp

from flask import Flask,render_template,url_for,flash,redirect,request,session,send_file

from students.forms import
UploadFileForm,RegistrationForm,LoginForm,ReplyFileForm,guideloginform,guideRegistrationForm,h
odloginform,hodRegistrationForm,UploadFileFormhod

from students import app,db,bcrypt

import os

from students.models import
User,Abstracts,Upload,Reply,Userguide,Userhod,Abstractshod,Replyhod,Final

from werkzeug.utils import secure_filename

from wtforms import FileField, SubmitField

from flask_login import login_user,current_user,logout_user,login_required

from flask_sqlalchemy import SQLAlchemy

from io import BytesIO

import csv

from students.dataset import Dataset

from students.utils import timer

from students.sentence_similarity import SentenceSimilarity

@app.route("/")

@app.route("/home")

def home():

    return render_template('home.html')

@app.route("/search")

def search():

    return render_template('search.html')

@app.route('/search', methods=["GET", "POST"])

def search_request():


    with open('data.csv','w',newline='') as csvfile:

        csvw=csv.writer(csvfile,delimiter=',')

        r=Final.query.all()

        for x in r:
```

```
            csvw.writerow([x.title,x.teamname])
    dataset = timer(Dataset, 'data.csv')
    sentence_sim = timer(SentenceSimilarity, dataset=dataset)
    query = request.form["input"]
    most_sim_docs = sentence_sim.get_most_similar(query)
    hits = [{"body": doc} for doc in most_sim_docs]
    d = []
    for title in hits:
        dt = {}
        qs = Replyhod.query.filter_by(title=title['body']).first()
        # tag = title['body']
        # search = "%{}%".format(tag)
        # qs = Replyhod.query.filter(Replyhod.title.like(search)).all()
        # print(f"Team {qs} and Title {title['body']}")
        # dt.update({'title':title})
        if qs:
            print("Team is:",qs.teamname)
            title.update({'teamname':qs.teamname})
        d.append(title)
        res = {}
    res['total'] = len(most_sim_docs)
    # print("AM from Db and Dataset",d)
    res['hits'] = d # hits
    return render_template('results.html', res=res)
@app.route("/home/guideregister",methods=['GET','POST'])
@app.route("/results")
def results():
    return render_template('results.html')
@app.route("/about")
def about():
    return render_template('about.html')
```

```
@app.route("/teamregister",methods=['GET','POST'])
def teamregister():
    if current_user.is_authenticated:
        return redirect(url_for('student'))
    form=RegistrationForm()
    if form.validate_on_submit():
        user = User(teamname=form.teamname.data, guidename=form.guidename.data,
password=form.password.data)
        db.session.add(user)
        db.session.commit()
        flash(f'Account created for {form.teamname.data}!','success')
        return redirect(url_for('student'))
    return render_template('teamregister.html',form=form)
@app.route("/home/studentlogin", methods=['GET', 'POST'])
def studentlogin():
    if current_user.is_authenticated:
        return redirect(url_for('student'))
    form=LoginForm()
    if form.validate_on_submit():
        user=User.query.filter_by(teamname=form.teamname.data).first()
        if user and user.password==form.password.data:
            print("Team",user.teamname)
            session['tmname']=user.teamname
            session['st_guidename'] = user.guidename
            login_user(user,remember=form.remember.data)
            return redirect(url_for('student'))
        else:
            flash('Login Unsuccessful. Please check teamname and password', 'danger')
            return redirect(url_for('teamregister'))
    return render_template('studentlogin.html',form=form)
 @app.route("/home/studentlogin/student", methods=['GET', 'POST'])
def student():
```

```
    if request.method == 'GET':
        return render_template('student.html')
    elif request.method == 'POST':
        if request.form['submit'] == 'Upload Project details to Guide':
            redirect(url_for('upg.html'))
        elif  request.form['submit'] == 'Upload Project details to HOD':
            print("hello2")
            return render_template('uph.html')
        elif  request.form['submit'] == 'View reply from Guide':
           return render_template('vrg.html')
        elif  request.form['submit'] == 'View reply from HOD':
           return render_template('vrh.html')
        return render_template('student.html')
@app.route("/home/studentlogin/student/upg",methods=['GET','POST'])
def upg():
    form=UploadFileForm()
    if request.method == 'POST':
        file = request.files['file']
        abs =
Abstracts(teamname=form.teamname.data,guidename=form.guidename.data,title=form.project.dat
a,technology=form.domain.data,filename=file.filename,data=file.read())
        db.session.add(abs)
        db.session.commit()
        return f'Uploaded: {file.filename}'
    return render_template('upg.html',form=form)
 ("/home/studentlogin/student/uph",methods=['GET','POST'])
def uph():
    form=UploadFileFormhod()
    if request.method == 'POST':
        file = request.files['file']
        absh = Abstractshod(
           teamname=form.teamname.data,
```

```
            hodname=form.hodname.data,
            guidename=form.guidename.data,
            title=form.project.data,
            technology=form.domain.data,
            filename=file.filename,
            data=file.read())
        db.session.add(absh)
        db.session.commit()
        return f'Uploaded: {file.filename}'
    # form.update({'guidename':session['st_guidename'] })
    return render_template('uph.html',form=form)
@app.route("/home/studentlogin/student/vrg")
def  vrg():
    #data=Abstracts.query.filter_by(guidename='g1').all()
    #current_user="d1"
    current_user=session['tmname']
data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname==cu
rrent_user) & (Reply.title==Abstracts.title) &
(Reply.filename==Abstracts.filename)).filter_by(teamname=current_user).all()
    return render_template('vrg.html',data=data)
@app.route("/home/studentlogin/student/download_files/<id>",methods=['GET','POST'])
def download_files(id):
    abs_qs=Abstracts.query.filter_by(id=id).first()
    return send_file(BytesIO(abs_qs.data), attachment_filename=abs_qs.filename,
as_attachment=True)
@app.route("/home/studentlogin/student/delete_replay_guide/<id>")
def delete_replay_guide(id):
    # Reply.query.filter_by(id=id).delete()
    qs = Reply.query.filter_by(id=id).first()
    qs_hod =
Abstracts.query.filter_by(teamname=qs.teamname,title=qs.title,filename=qs.filename).first()
    Reply.query.filter(Reply.id==id).delete()
    Abstracts.query.filter(Abstracts.id==qs_hod.id).delete()
```

```
    db.session.commit()
    current_user=session['tmname']
data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname==current_user) & (Reply.title==Abstracts.title) &
(Reply.filename==Abstracts.filename)).filter_by(teamname=current_user).all()
    return render_template('vrg.html',data=data)
@app.route("/home/studentlogin/student/upload_hod_simplyfy/<id>")
def upload_hod_simplyfy(id):
    qs = Abstracts.query.filter_by(id=id).first()
    qs_hod =
Abstractshod.query.filter_by(teamname=qs.teamname,guidename=qs.guidename,title=qs.title,technology=qs.technology,filename=qs.filename).first()
    print(qs_hod)
    if qs_hod:
        flash('Already Uploaded')
    else:
        absh = Abstractshod(
            teamname=qs.teamname,
            hodname='My HOD',
            guidename=qs.guidename,
            title=qs.title,
            technology=qs.technology,
            filename=qs.filename,
            data=qs.data)
        db.session.add(absh)
        db.session.commit()
    current_user=session['tmname']
data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname==current_user) & (Reply.title==Abstracts.title) &
(Reply.filename==Abstracts.filename)).filter_by(teamname=current_user).all()
    return render_template('vrg.html',data=data)
@app.route("/home/studentlogin/student/upload_hod_final/<id>")
def upload_hod_final(id):
    qs = Abstractshod.query.filter_by(id=id).first()
```

```
    qs_hod=Final.query.filter_by(teamname=qs.teamname).first()
    print(qs_hod)
    if  qs_hod:
        flash("Already Uploaded")
    else:
        absh = Final(
            teamname=qs.teamname,
            guidename=qs.guidename,
            title=qs.title,
            technology=qs.technology,
            filename=qs.filename,
            data=qs.data)
        db.session.add(absh)
        db.session.commit()

    current_user=session['tmname']

data=db.session.query(Abstractshod,Replyhod).join(Abstractshod,(Replyhod.teamname==Abstractsh
od.teamname==current_user) & (Replyhod.title==Abstractshod.title) &
(Replyhod.filename==Abstractshod.filename)).filter_by(teamname=current_user).all()
    return render_template('vrh.html',data=data)

@app.route("/home/studentlogin/student/vrh")
def  vrh():
    current_user=session['tmname']

data=db.session.query(Abstractshod,Replyhod).join(Abstractshod,(Replyhod.teamname==Abstractsh
od.teamname==current_user) & (Replyhod.title==Abstractshod.title) &
(Replyhod.filename==Abstractshod.filename)).filter_by(teamname=current_user).all()
    return render_template('vrh.html',data=data)

@app.route("/home/studentlogin/student/download_filevrh/<id>",methods=['GET','POST'])
def download_filevrh(id):
    abs_qs=Abstractshod.query.filter_by(id=id).first()
```

```
    return send_file(BytesIO(abs_qs.data), attachment_filename=abs_qs.filename,
as_attachment=True)


@app.route("/guideregister",methods=['GET','POST'])
def guideregister():
    if current_user.is_authenticated:
        return redirect(url_for('guidepage'))
    form=guideRegistrationForm()
    if form.validate_on_submit():
        userguide = Userguide(username=form.username.data, email=form.email.data,
password=form.password.data)
        db.session.add(userguide)
        db.session.commit()
        flash(f'Account created for {form.username.data}!','success')
        return redirect(url_for('guidepage'))


    return render_template('guideregister.html',form=form)
@app.route("/home/guidelogin", methods=['GET', 'POST'])
def guidelogin():
    if current_user.is_authenticated:
        return redirect(url_for('guidepage'))
    form=guideloginform()
    if form.validate_on_submit():
        userguide=Userguide.query.filter_by(username=form.username.data).first()
            if userguide and userguide.password==form.password.data:
          session['tmname']=userguide.username
          login_user(userguide,remember=form.remember.data)
          return redirect(url_for('guidepage'))
        else:
          flash('Login Unsuccessful. Please check teamname and password', 'danger')
          return redirect(url_for('guideregister'))
    return render_template('guidelogin.html',form=form)
```

```
@app.route("/home/guidelogin/guidepage", methods=['GET', 'POST'])
def guidepage():
   if request.method == 'GET':
      return render_template('guidepage.html')
   elif request.method == 'POST':
      if request.form['submit'] == 'View Project Details':
         redirect(url_for('vpd.html'))
      elif  request.form['submit'] == 'View project Status':
        return render_template('vpds.html')
      return render_template('guidepage.html')
@app.route("/home/guidelogin/guidepage/vpd",methods=['GET','POST'])
def vpd():
    #data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname)
& (Reply.title==Abstracts.title) & (Reply.filename==Abstracts.filename)).all()
    current_user=session['tmname']
    data=Abstracts.query.filter_by(guidename=current_user).all()

#data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname==c
urrent_user) & (Reply.title==Abstracts.title) &
(Reply.filename==Abstracts.filename)).filter_by(guidename=current_user).all()
    return render_template('vpd.html',data=data)
@app.route("/home/guidelogin/guidepage/vpd/download_file/<id>",methods=['GET','POST'])
def download_file(id):
   abs_qs=Abstracts.query.filter_by(id=id).first()
   return send_file(BytesIO(abs_qs.data), attachment_filename=abs_qs.filename,
as_attachment=True)
@app.route("/home/guidelogin/guidepage/vps",methods=['GET','POST'])
def vps():
    current_user=session['tmname']
    data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname)
& (Reply.title==Abstracts.title) &
(Reply.filename==Abstracts.filename)).filter_by(guidename=current_user).all()
        #data=Abstracts.query.filter_by(guidename=current_user).all()
    #data=Abstracts.query.filter_by(guidename='g1').all()
```

```
    return render_template('vps.html',data=data)
@app.route("/home/guidelogin/guidepage/vpd/name/<id>",methods=['GET','POST'])
def name(id):
   return render_template('name.html',abstracts=Abstracts.query.filter_by(id=id))
@app.route("/home/guidelogin/guidepage/vpd/name/appr_rej/<id>",methods=['GET','POST'])
def appr_rej(id):
   form=ReplyFileForm()
   if form.validate_on_submit():
         if form.status.data=='A':
            val="Approve"
         if form.status.data=='R':
            val="Reject"
         data=Abstracts.query.filter_by(id=id).all()
         teamname=data[0].teamname
         title=data[0].title
         filename=data[0].filename
         guidereply =
Reply(reply=form.reply.data,status=val,teamname=teamname,title=title,filename=filename)
         db.session.add(guidereply)
         db.session.commit()
         return redirect(url_for('vpd'))
   return render_template('appr_rej.html',abstracts=Abstracts.query.filter_by(id=id),form=form)
@app.route("/table/appr_rej/<id>/appr",methods=['GET','POST'])
def appr(id):
   return render_template('appr.html',abstracts=Abstracts.query.filter_by(id=id))
@app.route("/table")
def table():
   data=db.session.query(User,Abstracts).join(Abstracts,User.teamname==Abstracts.teamname).all()
   data1=User.query.all()
   data2=Abstracts.query.all()
   #data=User.join(Abstracts,User.teamname==Abstracts.teamname)
```

```
    return render_template('table.html',data=data,data1=data1,data2=data2)
@app.route("/hodregister",methods=['GET','POST'])
def hodregister():
    if current_user.is_authenticated:
        return redirect(url_for('hodpage'))
    form=guideRegistrationForm()
    if form.validate_on_submit():
        userhod = Userhod(username=form.username.data, email=form.email.data,
password=form.password.data)
        db.session.add(userhod)
        db.session.commit()
        flash(f'Account created for {form.username.data}!','success')
        return redirect(url_for('hodpage'))
    return render_template('hodregister.html',form=form)
@app.route("/home/hodlogin", methods=['GET', 'POST'])
def hodlogin():
    if current_user.is_authenticated:
        return redirect(url_for('hodpage'))
    form=hodloginform()
    if form.validate_on_submit():
        userhod=Userhod.query.filter_by(username=form.username.data).first()

        if userhod and userhod.password==form.password.data:
            session['temname']=userhod.username
            login_user(userhod,remember=form.remember.data)
            return redirect(url_for('hodpage'))
        else:
            flash('Login Unsuccessful. Please check teamname and password', 'danger')
            return redirect(url_for('hodregister'))
    return render_template('hodlogin.html',form=form)
@app.route("/home/hodlogin/hodpage", methods=['GET', 'POST'])
def hodpage():
```

```
    if request.method == 'GET':
        return render_template('hodpage.html')
    elif request.method == 'POST':
        if request.form['submit'] == 'View Project Details':
            redirect(url_for('vpdh.html'))
        elif  request.form['submit'] == 'View Project Status':
            return render_template('vpsh.html')

        return render_template('hodpage.html')
@app.route("/home/hodlogin/hodpage/vpdh",methods=['GET','POST'])
def vpdh():
    current_user=session['temname']
    & (Reply.title==Abstracts.title) & (Reply.filename==Abstracts.filename)).all()
    #data=Abstractshod.query.filter_by(hodname=current_user).all()
    data=Abstractshod.query.filter_by().all()
    return render_template('vpdh.html',data=data)
@app.route("/home/hodlogin/hodpage/vpdh/download_fileh/<id>",methods=['GET','POST'])
def download_fileh(id):
    abs_qs=Abstractshod.query.filter_by(id=id).first()
    return send_file(BytesIO(abs_qs.data), attachment_filename=abs_qs.filename,
as_attachment=True)
@app.route("/home/hodlogin/hodpage/vpsh",methods=['GET','POST'])
def vpsh():
    #current_user=session['tmname']

data=db.session.query(Abstractshod,Replyhod).join(Abstractshod,(Replyhod.teamname==Abstractsh
od.teamname) & (Replyhod.title==Abstractshod.title) &
(Replyhod.filename==Abstractshod.filename)).all()
    #data=Abstracts.query.filter_by(guidename=current_user).all()
    #data=Abstracts.query.filter_by(guidename='g1').all()
    return render_template('vpsh.html',data=data)
@app.route("/home/hodlogin/hodpage/vpdh/nameh/<id>",methods=['GET','POST'])
def nameh(id):
```

```
    return render_template('nameh.html',abstracts=Abstractshod.query.filter_by(id=id))
@app.route("/home/hodlogin/hodpage/vpdh/nameh/appr_rejh/<id>",methods=['GET','POST'])
def appr_rejh(id):
    form=ReplyFileForm()
    if form.validate_on_submit():
        if form.status.data=='A':
            val="Approve"
        if form.status.data=='R':
            val="Reject"
        data=Abstractshod.query.filter_by(id=id).all()
        teamname=data[0].teamname
        title=data[0].title
        filename=data[0].filename
        hodreply =
Replyhod(reply=form.reply.data,status=val,teamname=teamname,title=title,filename=filename)
        db.session.add(hodreply)
        db.session.commit()
        return redirect(url_for('vpdh'))
    return
render_template('appr_rejh.html',abstracts=Abstractshod.query.filter_by(id=id),form=form)
@app.route("/home/studentlogin/student/delete_replay_hod/<id>")
def delete_replay_hod(id):
    # Reply.query.filter_by(id=id).delete()
    Replyhod.query.filter(Replyhod.id==id).delete()
    db.session.commit()
    current_user=session['tmname']

data=db.session.query(Abstractshod,Replyhod).join(Abstractshod,(Replyhod.teamname==Abstractsh
od.teamname==current_user) & (Replyhod.title==Abstractshod.title) &
(Replyhod.filename==Abstractshod.filename)).filter_by(teamname=current_user).all()
    return render_template('vrh.html',data=data)
@app.route("/home/project_status1/reply_by_hod")
def reply_by_hod():
```

```
    return render_template('reply_by_hod.html')


@app.route("/home/project_status/reply_by_guide")
def reply_by_guide():
    return render_template('reply_by_guide.html')
@app.route("/home/")
def abg():
    return render_template('abg.html')


@app.route('/guide')
def guide():
    if current_user.is_authenticated:
        return redirect(url_for('guidepage'))
    form=guideloginform()
    if form.validate_on_submit():
        userguide=Userguide.query.filter_by(username=form.username.data).first()
        if userguide and userguide.password==form.password.data:
            login_user(userguide,remember=form.remember.data)
            return redirect(url_for('guidepage'))
        else:
            flash('Login Unsuccessful. Please check teamname and password', 'danger')
            return redirect(url_for('guideregister'))
    return render_template('guidelogin.html',form=form)
@app.route('/hod')
def hod():
    if current_user.is_authenticated:
        return redirect(url_for('hodpage'))
    form=hodloginform()
    if form.validate_on_submit():
        userhod=Userhod.query.filter_by(username=form.username.data).first()
        if userhod and userhod.password==form.password.data:
```

```
            login_user(userhod,remember=form.remember.data)
            return redirect(url_for('hodpage'))
        else:
            flash('Login Unsuccessful. Please check teamname and password', 'danger')
            return redirect(url_for('hodregister'))
    return render_template('hodlogin.html',form=form)
@app.route("/home/hodlogin/hodpage/vd",methods=['GET','POST'])
def vd():
    #data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname)
& (Reply.title==Abstracts.title) & (Reply.filename==Abstracts.filename)).all
    data=Final.query.all()
#data=db.session.query(Abstracts,Reply).join(Abstracts,(Reply.teamname==Abstracts.teamname==c
urrent_user) & (Reply.title==Abstracts.title) &
(Reply.filename==Abstracts.filename)).filter_by(guidename=current_user).all()
    return render_template('vd.html',data=data)
@app.route("/logout")
def logout():
    logout_user()
    return redirect(url_for('home'))
@app.route("/edit")
@login_required
def edit():
    return render_template('edit.html')
@app.route("/home/guidepage",methods=['GET','POST'])
def backguidepage1():
    return render_template('guidepage.html')
@app.route("/home/hodpage",methods=['GET','POST'])
def backhodpage2():
    return render_template('hodpage.html')
@app.route("/home",methods=['GET','POST'])
def backstudent():
    return render_template('home.html')
@app.route("/home/student",methods=['GET','POST'])
```

```
def backvrg():
    return render_template('home.html')
@app.route("/home/student",methods=['GET','POST'])
def backvrh():
    return render_template('home.html')
@app.route("/home",methods=['GET','POST'])
def backguidepage():
    return render_template('home.html')
@app.route("/home",methods=['GET','POST'])
def backhodpage():
    return render_template('home.html')
@app.route("/home/guidelogin/guidepage/vpd",methods=['GET','POST'])
def backname():
    return render_template('vpd.html')
@app.route("/home/hodlogin/hodpage/vpdh",methods=['GET','POST'])
def backnameh():
    return render_template('vpdh.html')
@app.before_first_request
def create_tables():
    db.create_all()
if __name__ == '__main__':
    app.run(debug=True)
    db.create_all()
```

3.5.3 – GitHub Links –

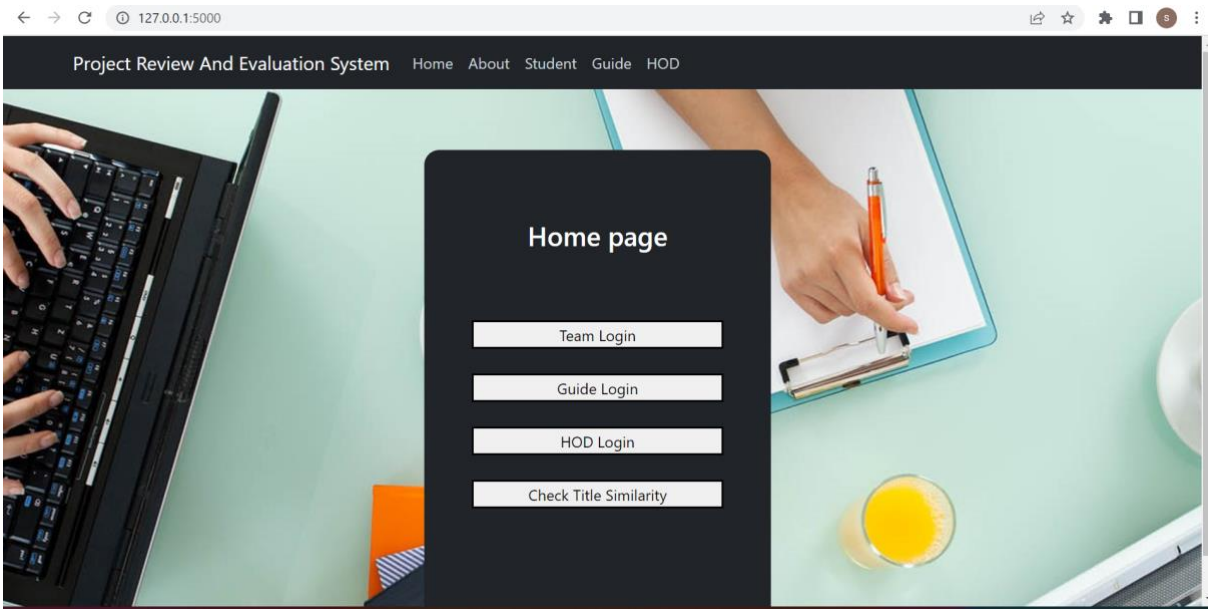
https://github.com/ShravaniNarsini/Project-Review-and-evaluation

CHAPTER 4 –

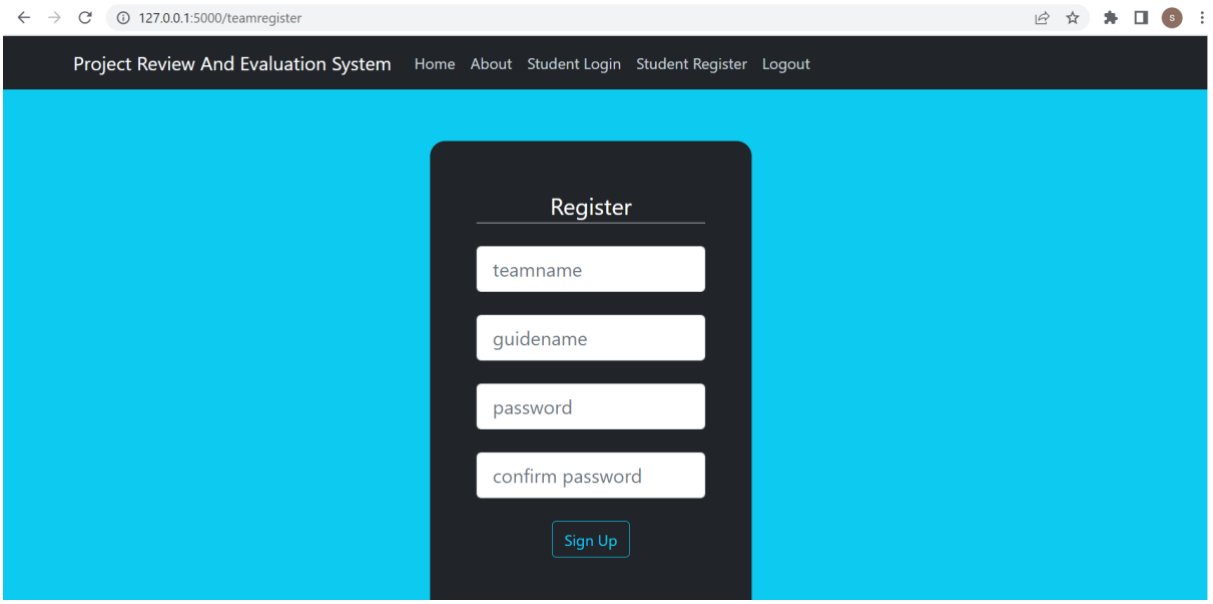
RESULTS

The following are the results obtained after implementation –

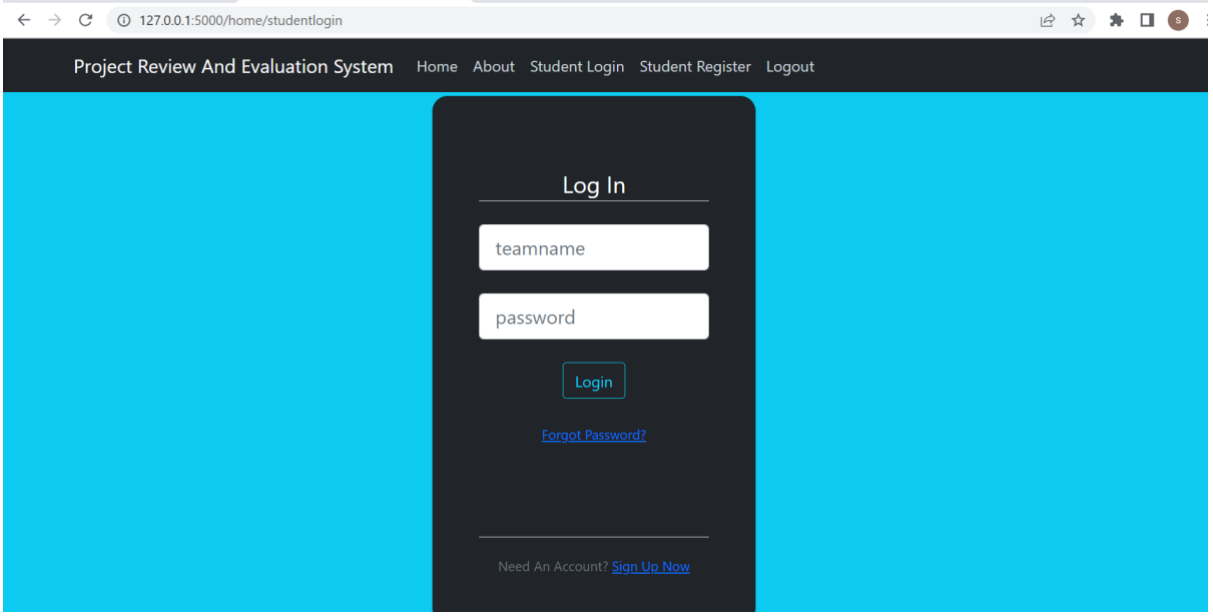
UI –

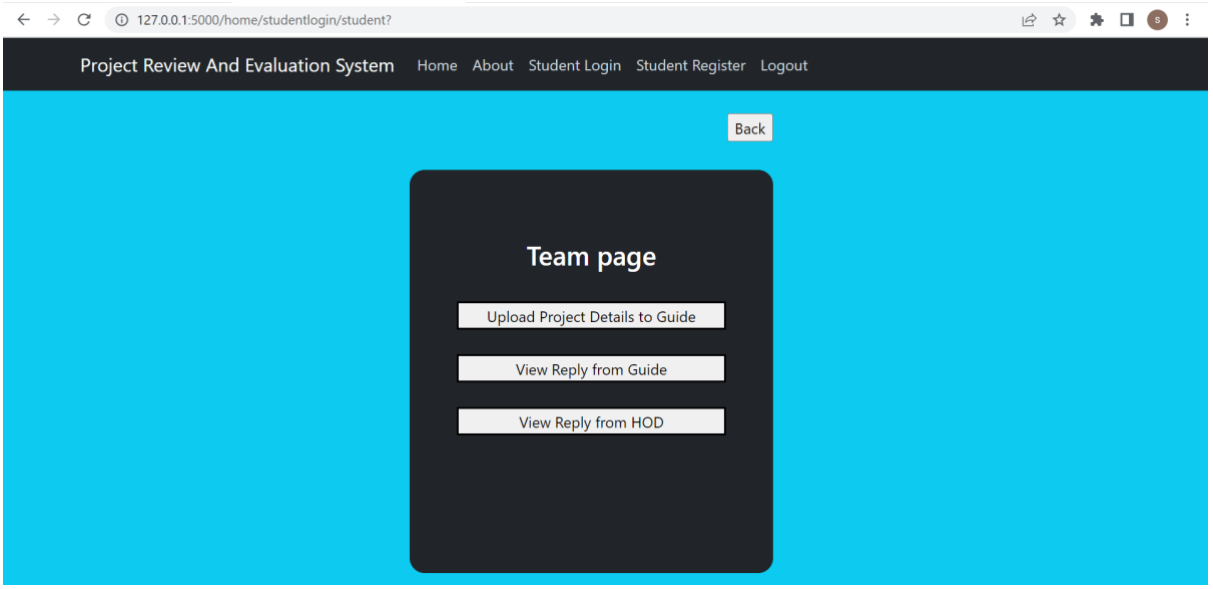


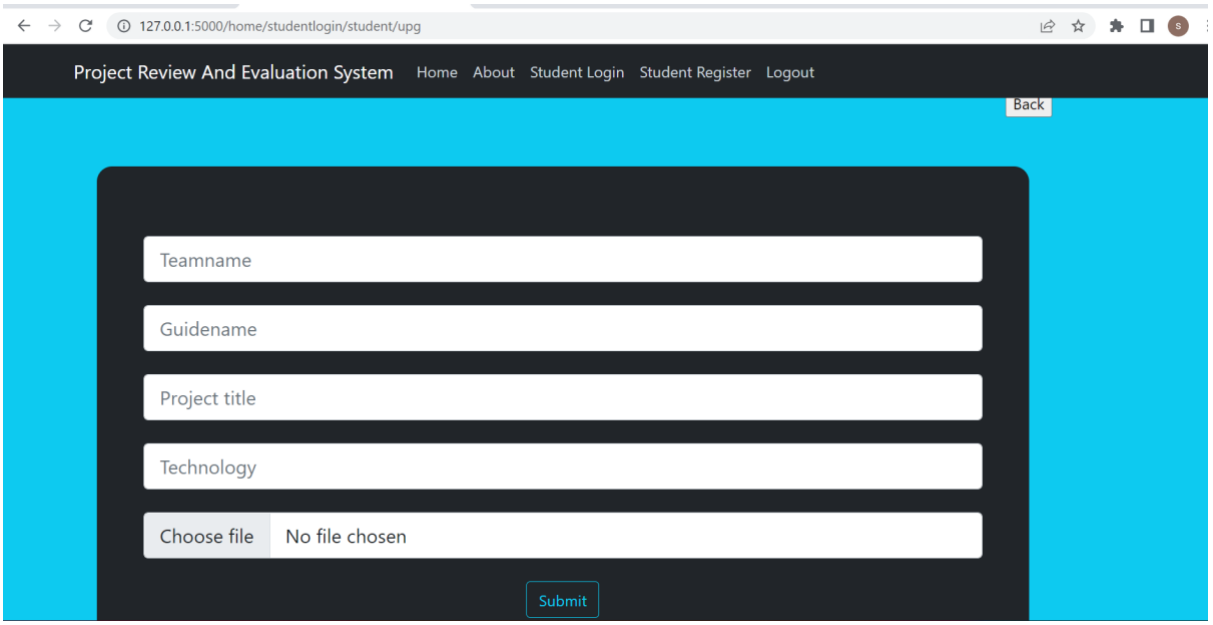
TEAM REGISTRATION PAGE:

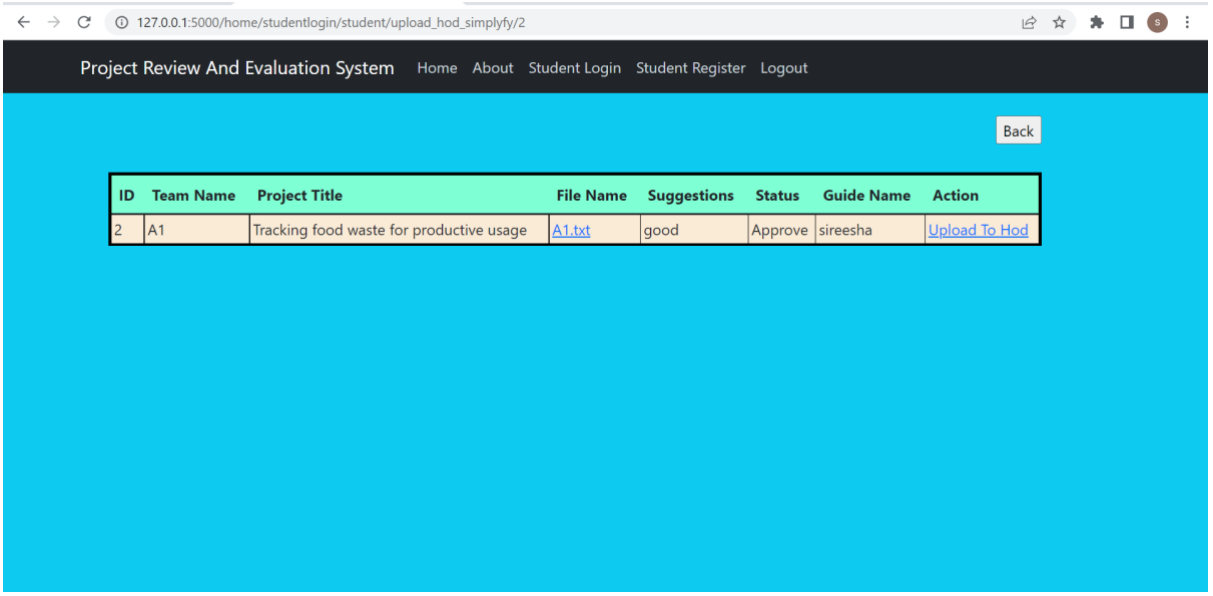


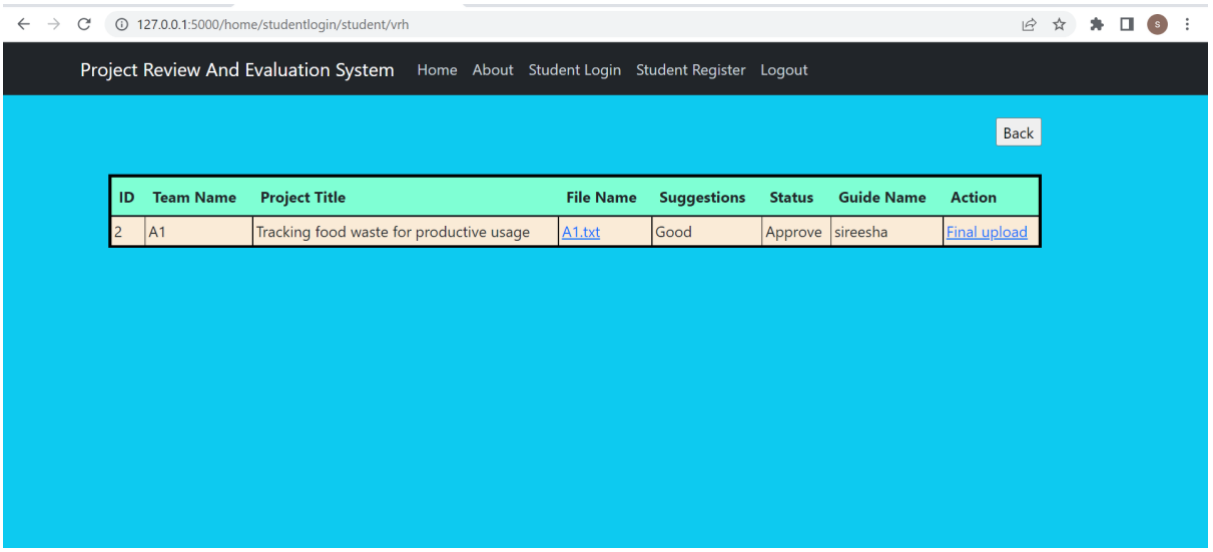
Team Login:

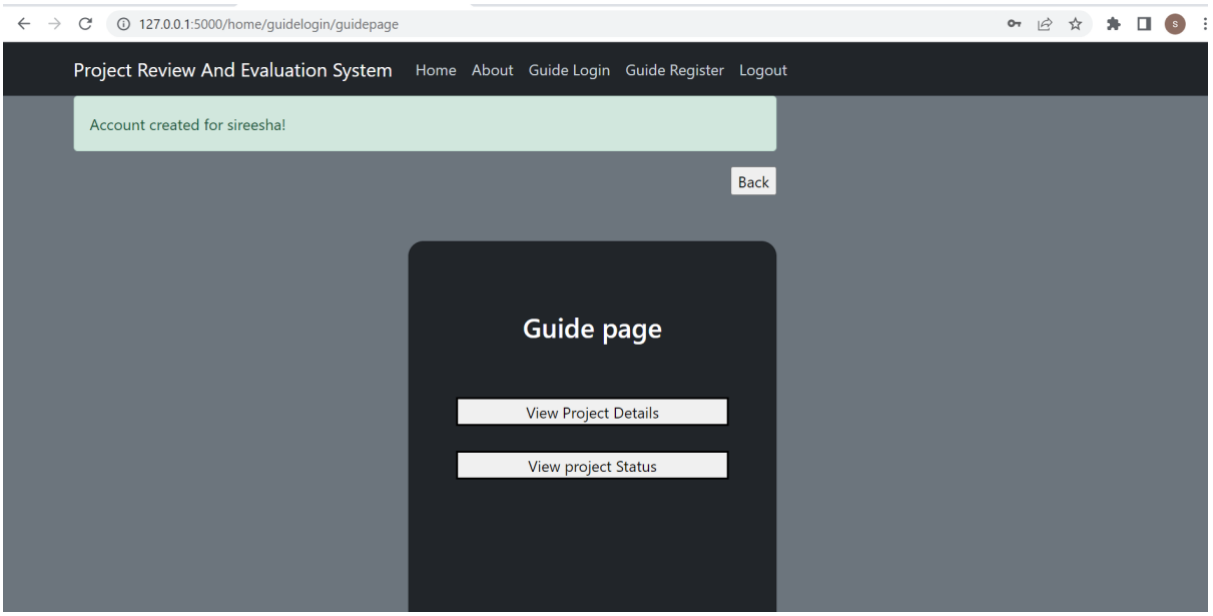


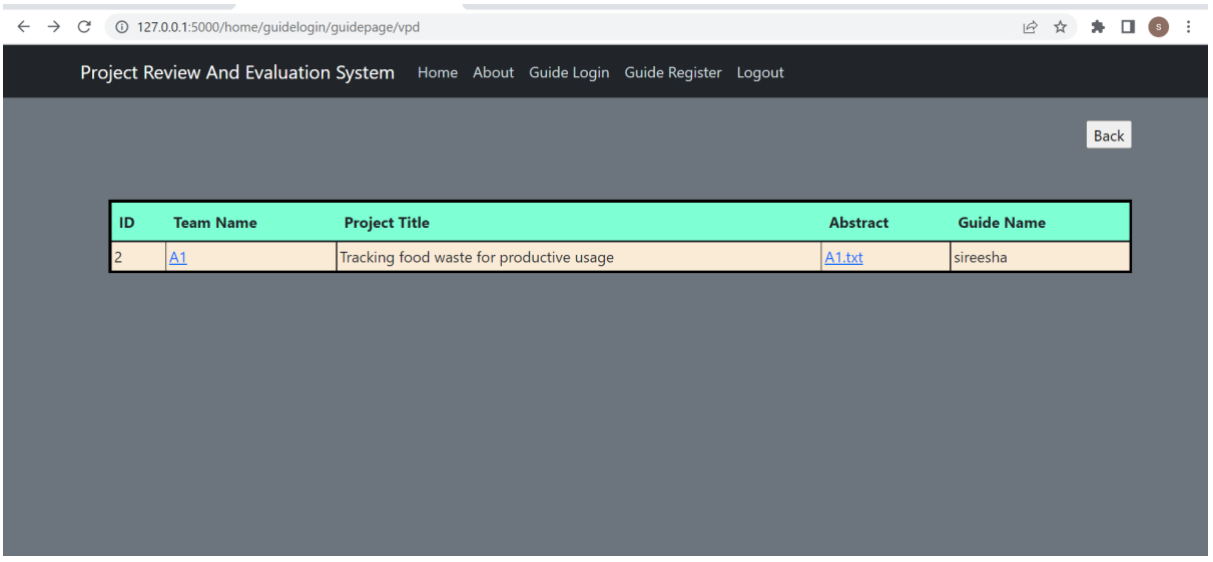


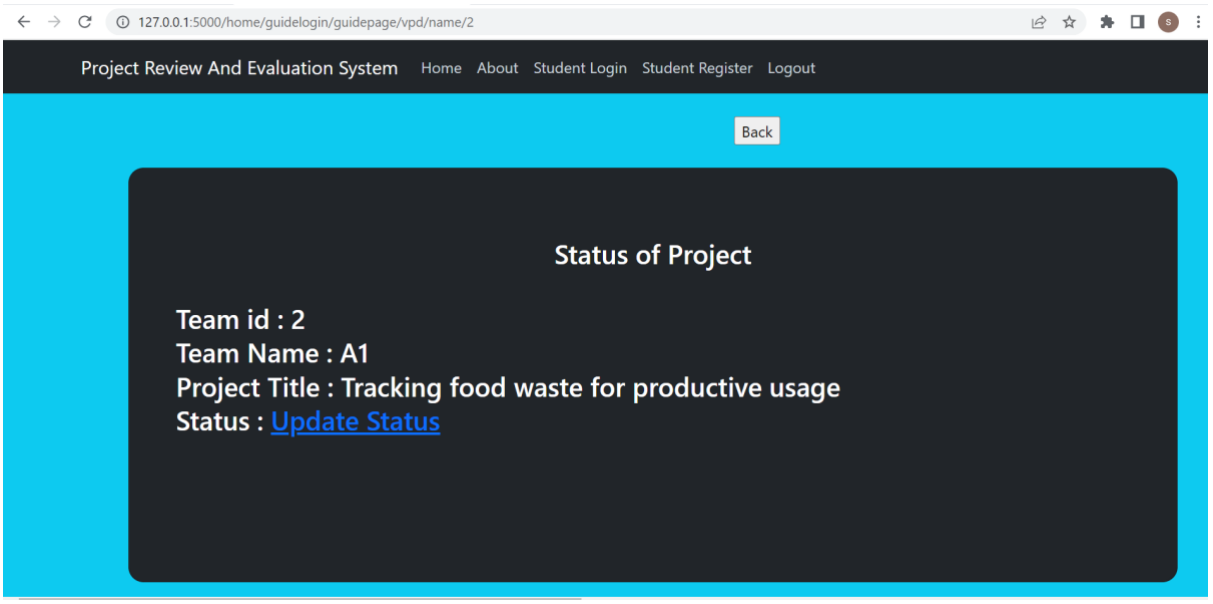

127.0.0.1:5000/home/studentlogin/student/upg
Project Review And Evaluation System
Home
About
Student Login
Student Register
Logout
Back
Teamname
Guidename
Project title
Technology
Choose file
No file chosen
Submit

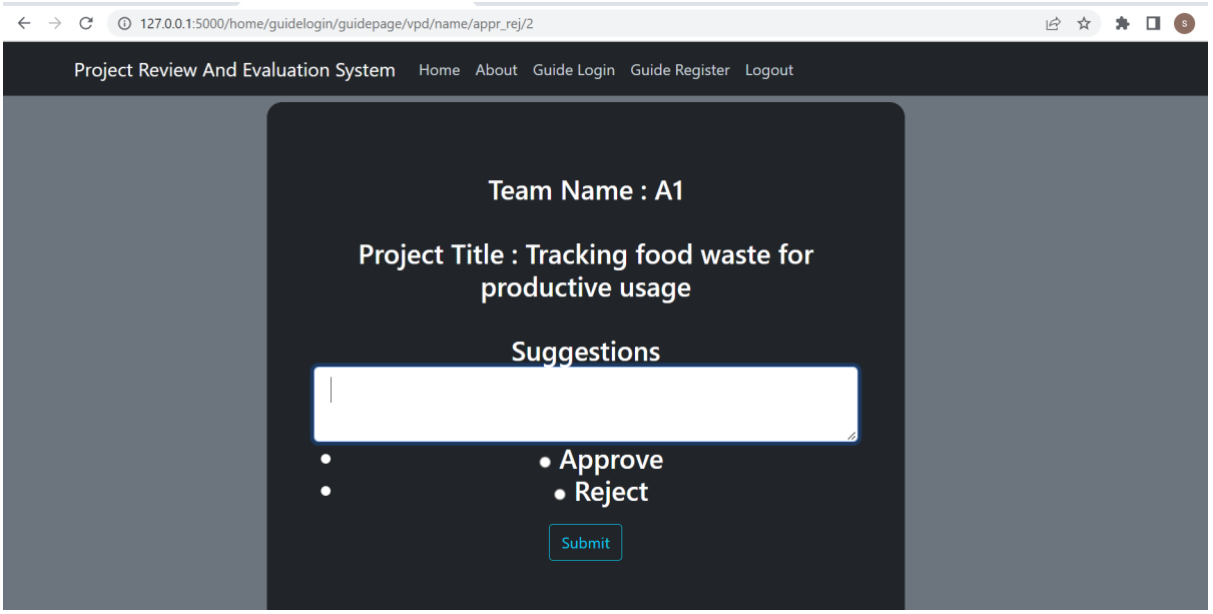

127.0.0.1:5000/home/studentlogin/student/upload_hod_simplyfy/2
Project Review And Evaluation System
Home
About
Student Login
Student Register
Logout
Back
ID
Team Name
Project Title
File Name
Suggestions
Status
Guide Name
Action
2
A1
Tracking food waste for productive usage
A1.txt
good
Approve
sireesha
Upload To Hod

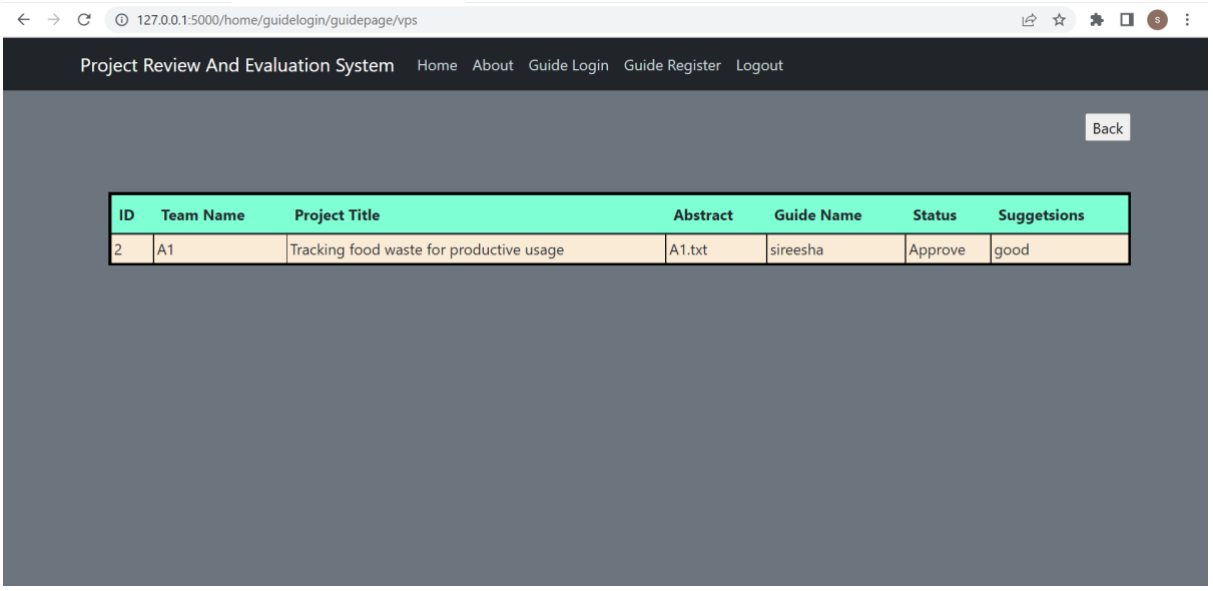

127.0.0.1:5000/home/studentlogin/student/vrh
Project Review And Evaluation System
Home
About
Student Login
Student Register
Logout
Back
ID
Team Name
Project Title
File Name
Suggestions
Status
Guide Name
Action
2
A1
Tracking food waste for productive usage
A1.txt
Good
Approve
sireesha
Final upload


127.0.0.1:5000/home/guidelogin/guidepage
Project Review And Evaluation System
Home
About
Guide Login
Guide Register
Logout
Account created for sireesha!
Back
Guide page
View Project Details
View project Status

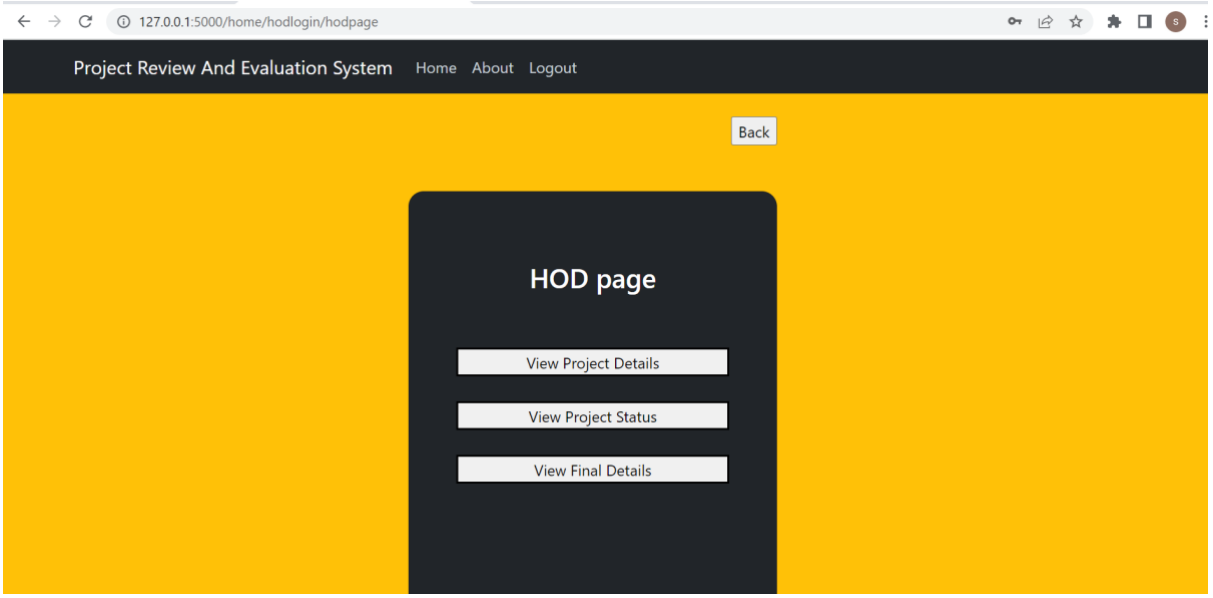

127.0.0.1:5000/home/guidelogin/guidepage/vpd
Project Review And Evaluation System
Home
About
Guide Login
Guide Register
Logout
Back
ID
Team Name
Project Title
Abstract
Guide Name
2
A1
Tracking food waste for productive usage
A1.txt
sireesha

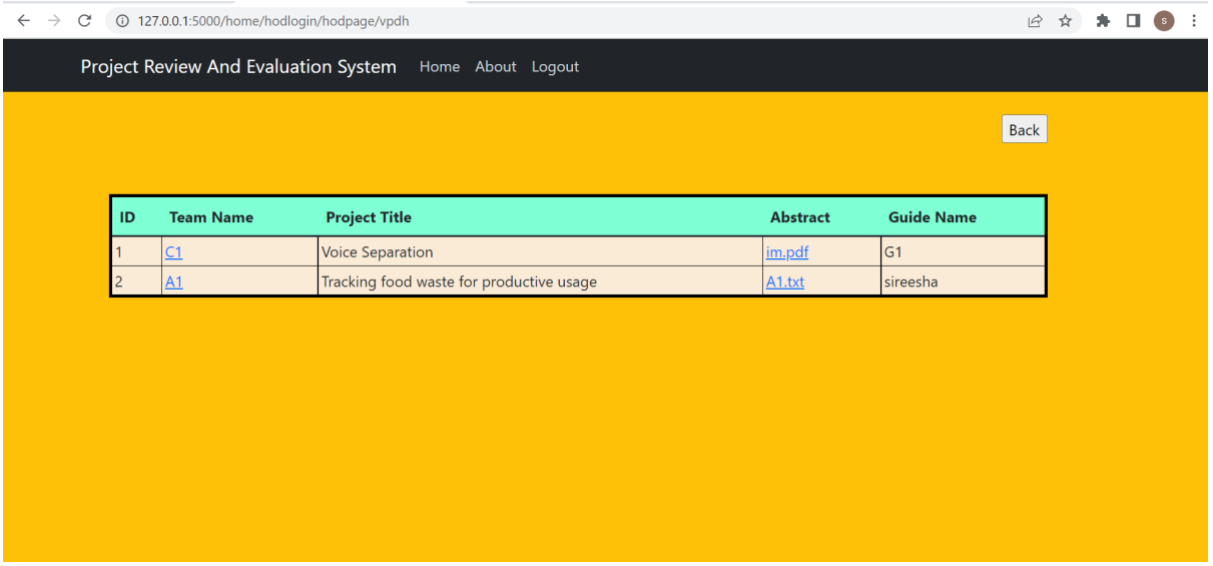




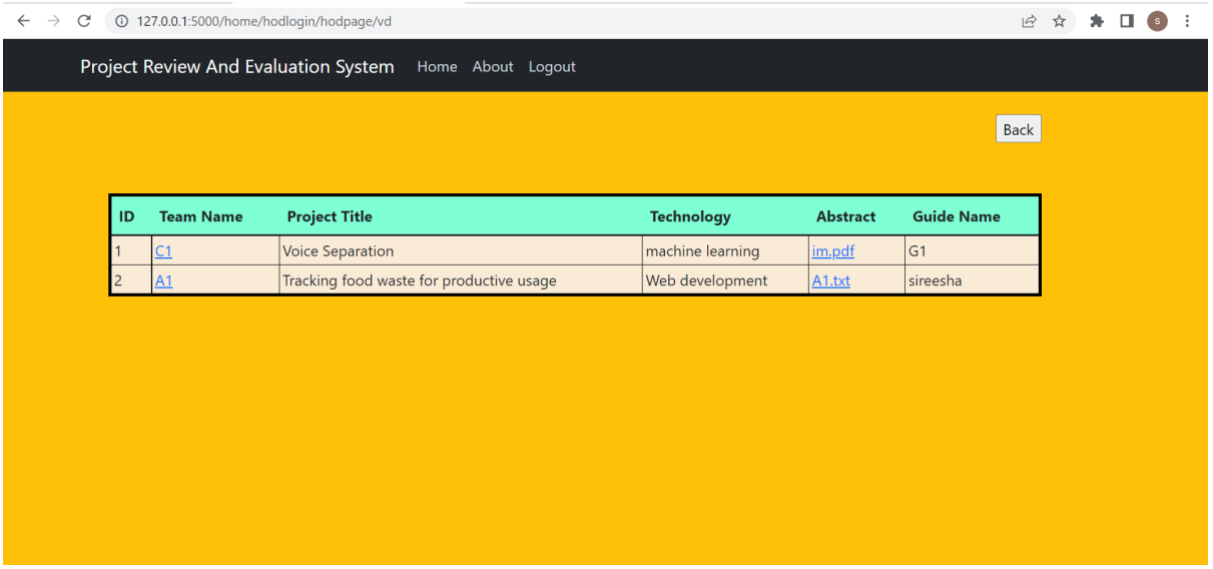


ID	Team Name	Project Title	Abstract	Guide Name	Status	Suggetsions
2	A1	Tracking food waste for productive usage	A1.txt	sireesha	Approve	good





ID	Team Name	Project Title	Abstract	Guide Name
1	C1	Voice Separation	im.pdf	G1
2	A1	Tracking food waste for productive usage	A1.txt	sireesha



ID	Team Name	Project Title	Technology	Abstract	Guide Name
1	C1	Voice Separation	machine learning	im.pdf	G1
2	A1	Tracking food waste for productive usage	Web development	A1.txt	sireesha

CHAPTER 5

DISCUSSION AND FUTURE WORK –

The project review and evaluation system is aimed at automating the existing manual system for review and evaluation of project and process the approval request through faculty.[9] It helps in maintaining the records of the students which will help the faculty team to manage project evaluation and documentation. It is useful in organizations with
a large number of students with various departments[10]. It's a quick process as it takes less time when compared to a manual process. It is very reliable and it leads to efficient data management.
For future work mailing module can be planned which will allow students to send email to HOD and guides with attachments. View/send/reply email options can be provided. Alerts or notification module can be added separately which will show notification icon and which when clicked will open the related information for which notification was received.

CHAPTER 6

REFERENCES –

https://www.academia.edu/28590094/An_Integrated_Project_Evaluation_System