Mini Project Mid Term Report

on

e-Learning Platform

Department of Computer Engineering & Applications

Institute of Engineering & Technology

Submitted By:

Atul Singh

Ayush Singh

Adarsh Chauhan

Rishabh Gupta

Abhishek Jaiswal

Submitted To:

Dr. Manoj Varshnay

Abstract

E-learning is another form of distance learning where education and training courses are delivered using computer technology. Typically, this means that courses are delivered either via the Internet, or on computer networks (linked computers). With the increased availability of smartphones and PCs with Internet access, e-learning is becoming more and more popular.This online application aims to provide a platform to any small to midsize educational organization who can’t afford their own online education software. In addition to providing a simple yet powerful platform to educational organizations, this application is also beneficial to small to midsize coaching institutes, now they can make their courses available to a bigger audience without having a need to set up physical centers in different locations.

The idea of e-learning is very cost effective and efficient and has great potential, with this project we aim to showcase the same.

Table of Contents

Introduction

Problem Statement :

Develop an e-learning platform.

About the project :

This application is currently web based and can be used using any device that has a browser that supports today's web standards (HTML, CSS, JavaScript).

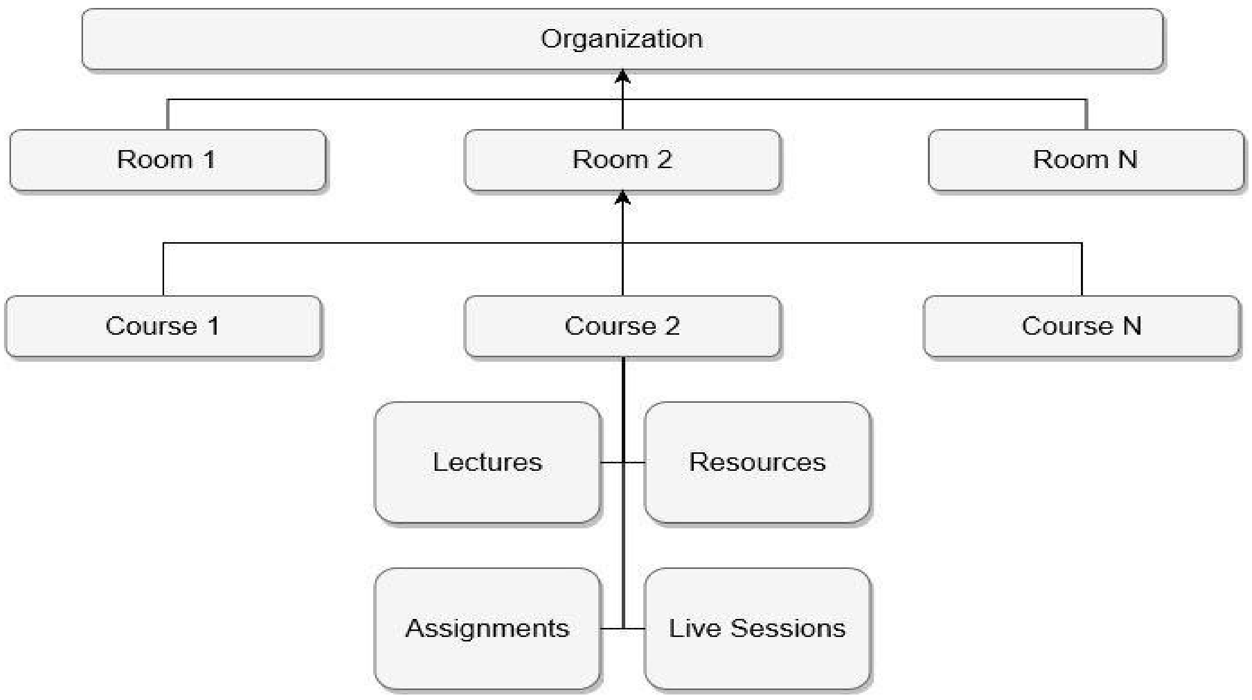
Through this application any organization can sign up using an email address and they will be presented with a dashboard using it they can create Rooms (equivalent to classes) and create courses inside those rooms along with assigning instructors for each course created.

Teachers can then start uploading content in assigned courses, they can upload lectures, course resources and resources specific to each lecture and also give

assignments. There is also a feature for live sessions that can be used for doubt clearing sessions or directly interacting with students.

With creation of every room, a link is generated that is accessible from the dashboard itself and can be used to add students to that room. Once students sign up using the room link provided to them, they will be added to the corresponding room, after which they can login and start viewing courses running in that room.

A general structure of the application is shown in figure below


Organization
Room 1
Room 2
Room N
Course 1
Course 2
Course N
Lectures
Resources
Assignments
Live Sessions

In this project, we have used front-end technologies like – HTML, CSS, Javascript and Django framework is used in the back-end part.

There are mainly three parts in our project –

- Administration section for the admin of the organization,
- Teacher section for the staff of organization
- Student section.

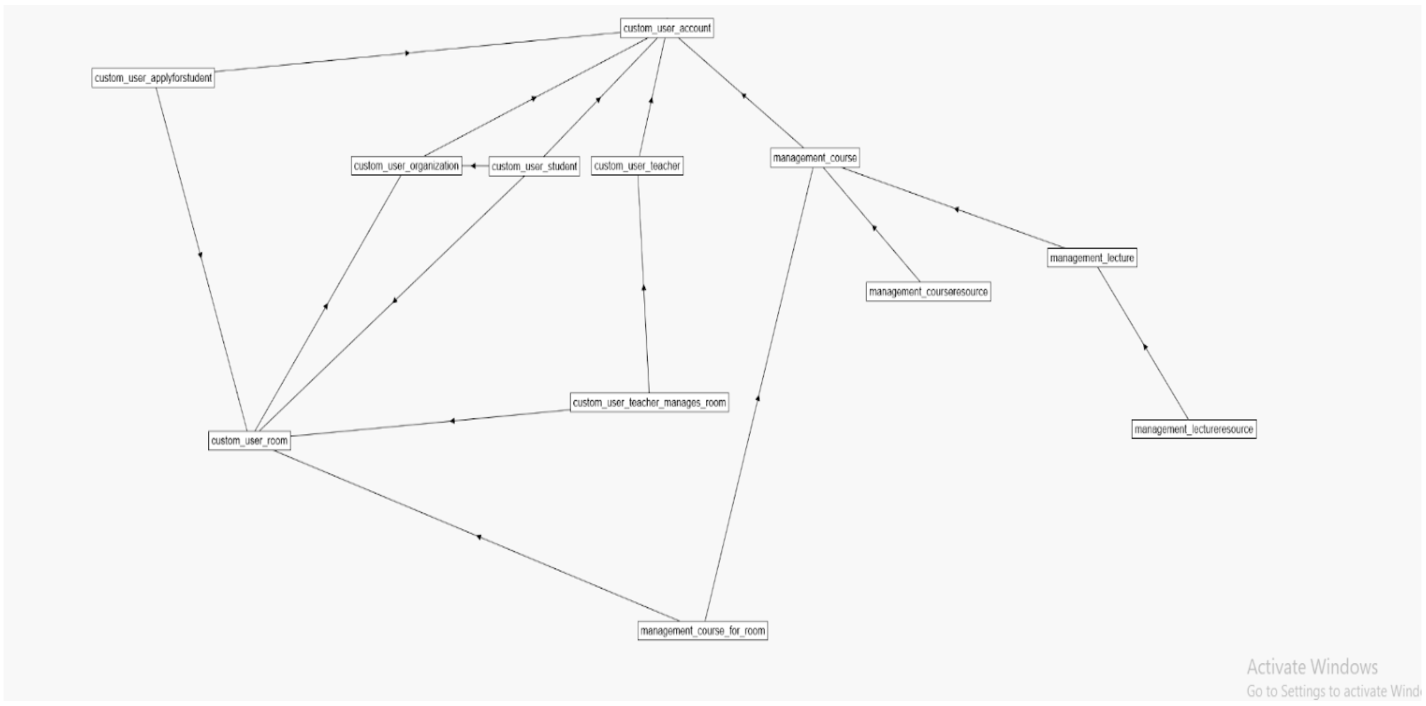
In administration part, admin will be able to add the rooms according to his requirements and add courses in it for the organization. He also add teachers and students of the organization.

In teacher section, teacher is able to add the materials, resources, related to the courses and also add the links of the video lectures as well.

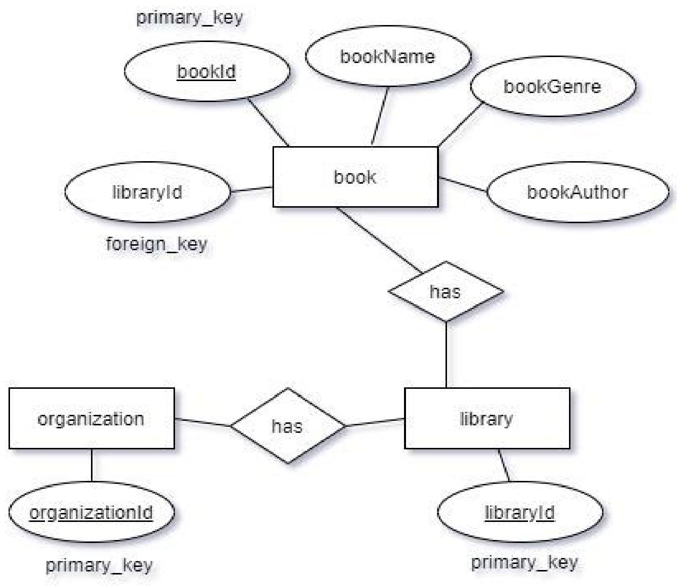
In the student section, student will login into his account provided by the organization and attend the lectures of the course which is present in his respective curriculum and he is also able to download all the corresponding materials and video lectures related to his courses.

The work that we have completed in the project is shown below with the help of the screenshots:-

It shows the structure of database used in our project:


custom_user_account
custom_user_applyforstudent
custom_user_organization
custom_user_student
custom_user_teacher
management_course
management_lecture
management_courseresource
custom_user_teacher_manages_room
management_lectureresource
custom_user_room
management_course_for_room

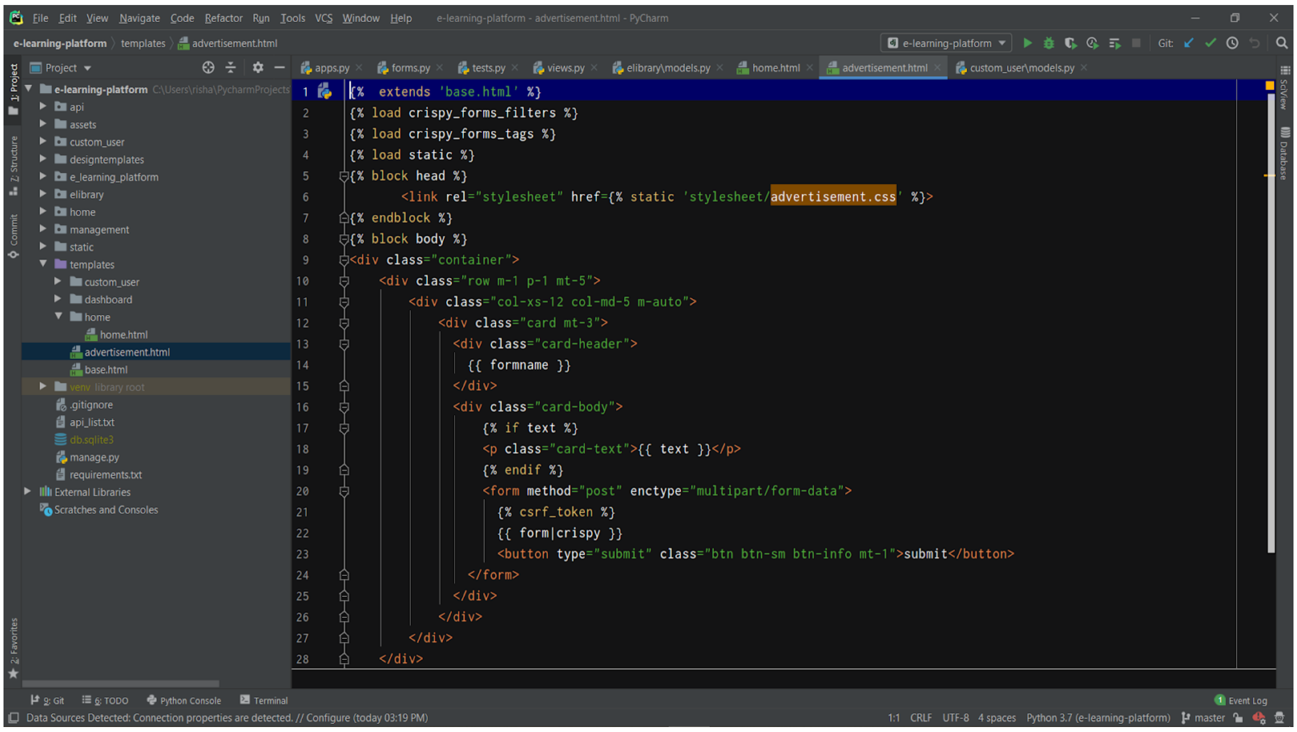
It shows the ER model for e-Library Management System:



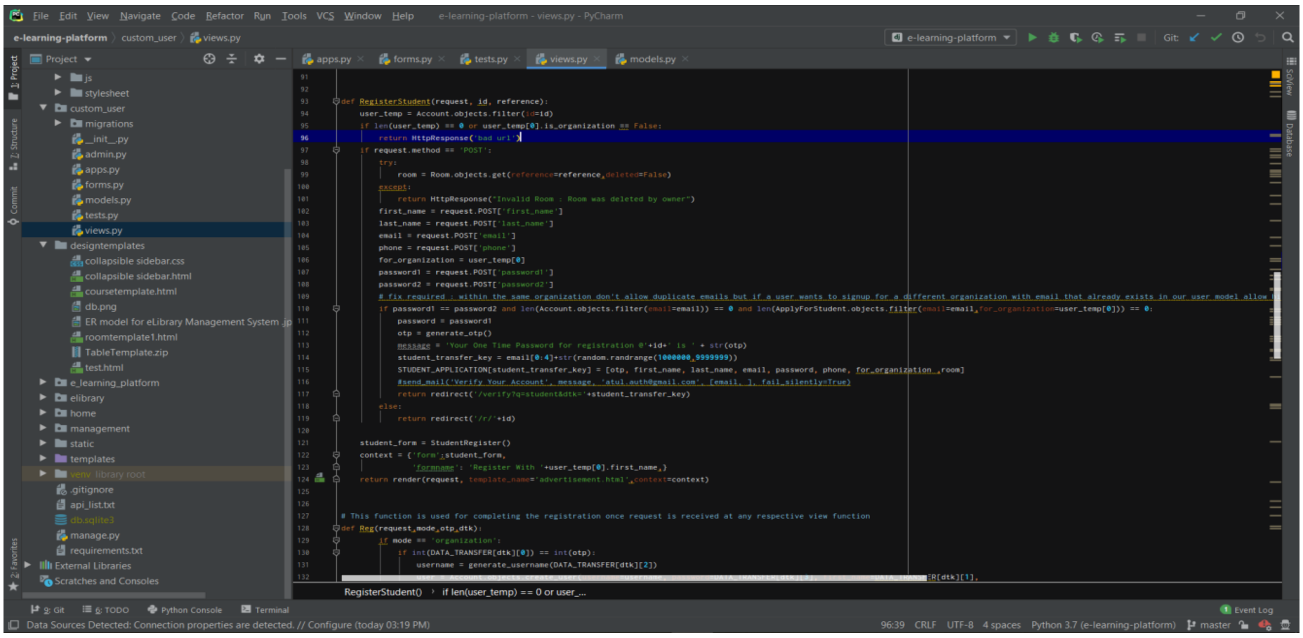
Functions :-

- `addBook()`
- `deleteBook()`
- `searchBook()`
- `listAllBooks()`
- `editBook()`

Some Screenshots from the code of the project



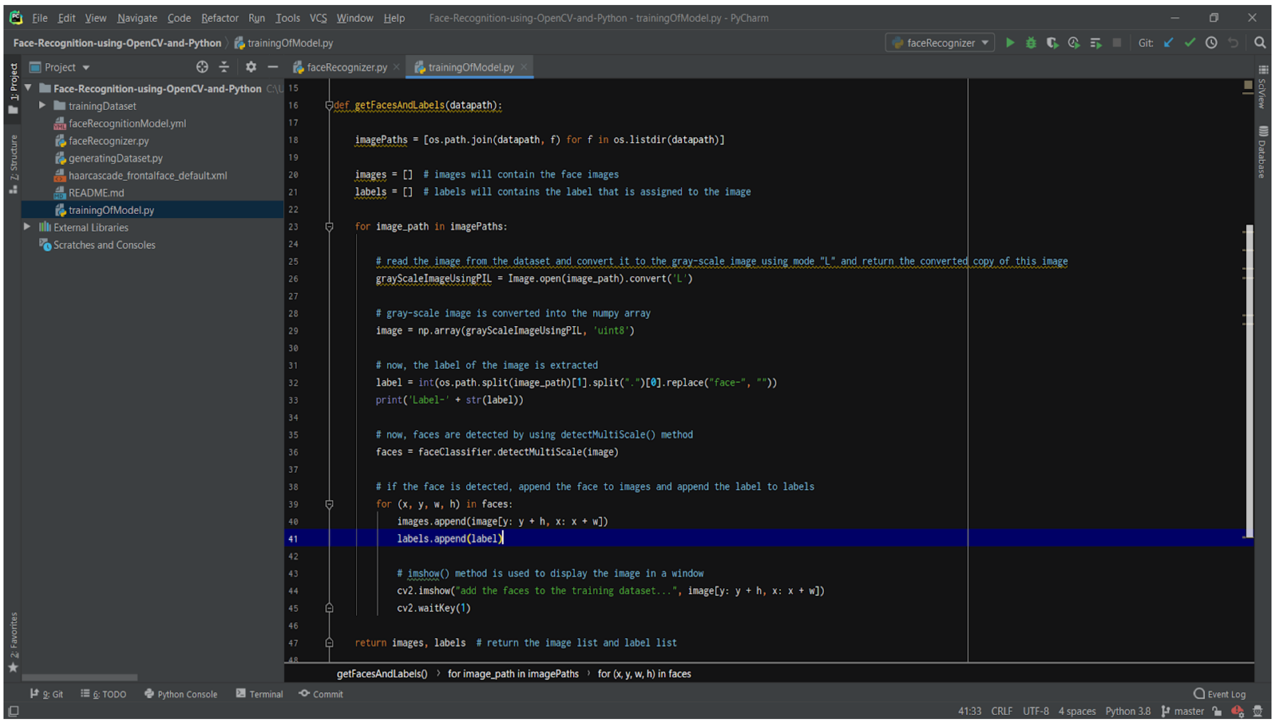
```
{%  extends 'base.html' %}
{% load crispy_forms_filters %}
{% load crispy_forms_tags %}
{% load static %}
{% block head %}
       <link rel="stylesheet" href="{% static 'stylesheet/advertisement.css' %}">
{% endblock %}
{% block body %}
<div class="container">
    <div class="row m-1 p-1 mt-5">
        <div class="col-xs-12 col-md-5 m-auto">
            <div class="card mt-3">
              <div class="card-header">
                {{ formname }}
              </div>
              <div class="card-body">
                  {% if text %}
                  <p class="card-text">{{ text }}</p>
                  {% endif %}
                  <form method="post" enctype="multipart/form-data">
                    {% csrf_token %}
                    {{ form|crispy }}
                    <button type="submit" class="btn btn-sm btn-info mt-1">submit</button>
                </form>
              </div>
            </div>
        </div>
    </div>
```



```
def RegisterStudent(request, id, reference):
    user_temp = Account.objects.filter(id=id)
    if len(user_temp) == 0 or user_temp[0].is_organization == False:
        return HttpResponse('bad url')
    if request.method == 'POST':
        try:
            room = Room.objects.get(reference=reference,deleted=False)
        except:
            return HttpResponse("Invalid Room : Room was deleted by owner")
        first_name = request.POST['first_name']
        last_name = request.POST['last_name']
        email = request.POST['email']
        phone = request.POST['phone']
        for_organization = user_temp[0]
        password1 = request.POST['password1']
        password2 = request.POST['password2']
        # fix required : within the same organization don't allow duplicate emails but if a user wants to signup for a different organization with email that already exists in our user model allow hi
        if password1 == password2 and len(Account.objects.filter(email=email)) == 0 and len(ApplyForStudent.objects.filter(email=email,for_organization=user_temp[0])) == 0:
            password = password1
            otp = generate_otp()
            message = 'Your One Time Password for registration @'+id+' is ' + str(otp)
            student_transfer_key = email[0:4]+str(random.randrange(1000000,9999999))
            STUDENT_APPLICATION[student_transfer_key] = [otp, first_name, last_name, email, password, phone, for_organization ,room]
            #send_mail('Verify Your Account', message, 'atul.auth@gmail.com', [email, ], fail_silently=True)
            return redirect('/verify?q=student&dtk='+student_transfer_key)
        else:
            return redirect('/r/'+id)

    student_form = StudentRegister()
    context = {'form':student_form,
               'formname': 'Register With '+user_temp[0].first_name,}
    return render(request, template_name='advertisement.html',context=context)


# This function is used for completing the registration once request is received at any respective view function
def Reg(request,mode,otp,dtk):
    if mode == 'organization':
        if int(DATA_TRANSFER[dtk][0]) == int(otp):
            username = generate_username(DATA_TRANSFER[dtk][2])
            user = Account.objects.create_user(username=username, password=DATA_TRANSFER[dtk][3], [illegible]ER[dtk][1],
```



```python
def getFacesAndLabels(datapath):

    imagePaths = [os.path.join(datapath, f) for f in os.listdir(datapath)]

    images = []  # images will contain the face images
    labels = []  # labels will contains the label that is assigned to the image

    for image_path in imagePaths:

        # read the image from the dataset and convert it to the gray-scale image using mode "L" and return the converted copy of this image
        grayScaleImageUsingPIL = Image.open(image_path).convert('L')

        # gray-scale image is converted into the numpy array
        image = np.array(grayScaleImageUsingPIL, 'uint8')

        # now, the label of the image is extracted
        label = int(os.path.split(image_path)[1].split(".")[0].replace("face-", ""))
        print('Label-' + str(label))

        # now, faces are detected by using detectMultiScale() method
        faces = faceClassifier.detectMultiScale(image)

        # if the face is detected, append the face to images and append the label to labels
        for (x, y, w, h) in faces:
            images.append(image[y: y + h, x: x + w])
            labels.append(label)

            # imshow() method is used to display the image in a window
            cv2.imshow("add the faces to the training dataset...", image[y: y + h, x: x + w])
            cv2.waitKey(1)

    return images, labels  # return the image list and label list
```

Requirement Analysis

Technologies Used :

● Operating System	:	Cross Platform
● Front End Technologies	:	Html, JavaScript, CSS
● Back End Technologies	:	Python, Django, Machine Learning
● IDE	:	Pycharm
● Database	:	SQLite
● Python Version	:	3.8.2
● Supporting Technologies	:	Git, GitHub

Software Requirement* :

● Operating System	:	Cross Platform
● Browser	:	Any browser supporting Html, CSS & JavaScript

Hardware Requirement *:

● Hardware	:	Any Standard System That Can Run a Web Browser Equipped With HTML, CSS & JavaScript.
● Key Board	:	Standard Windows Keyboard
● Mouse	:	Two or Three Button Mouse
● Monitor	:	SVGA

Progress

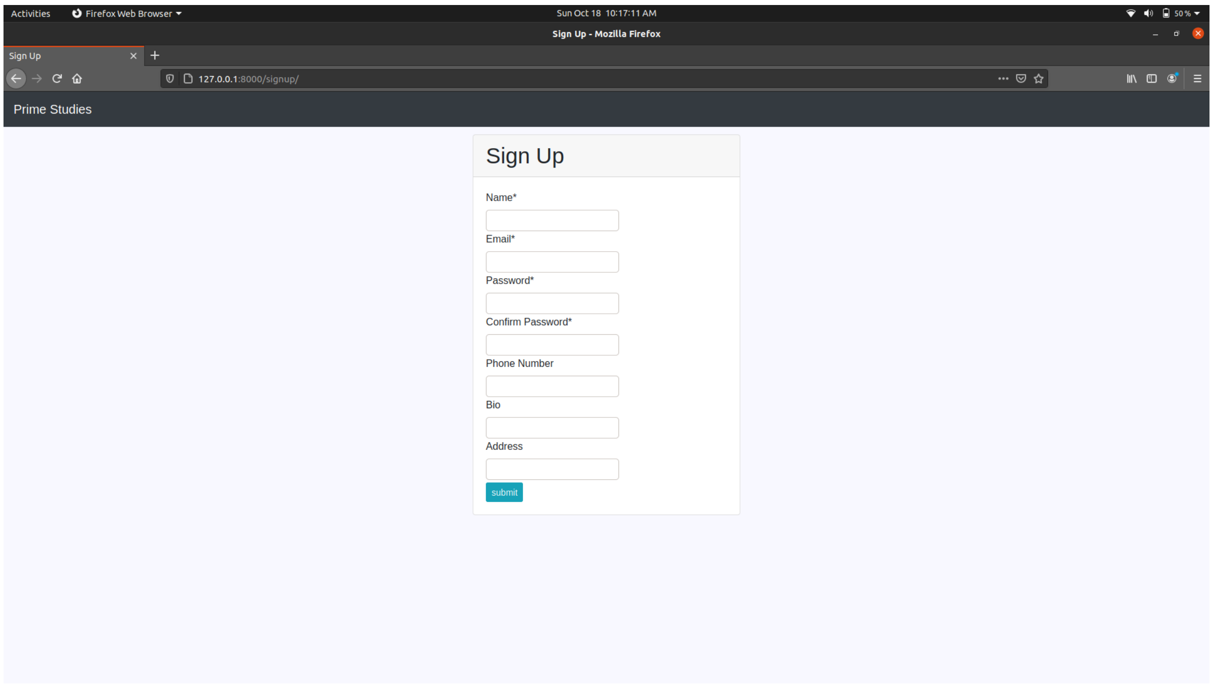
Database Is ready and is tested using dummy data added from the django administration panel.

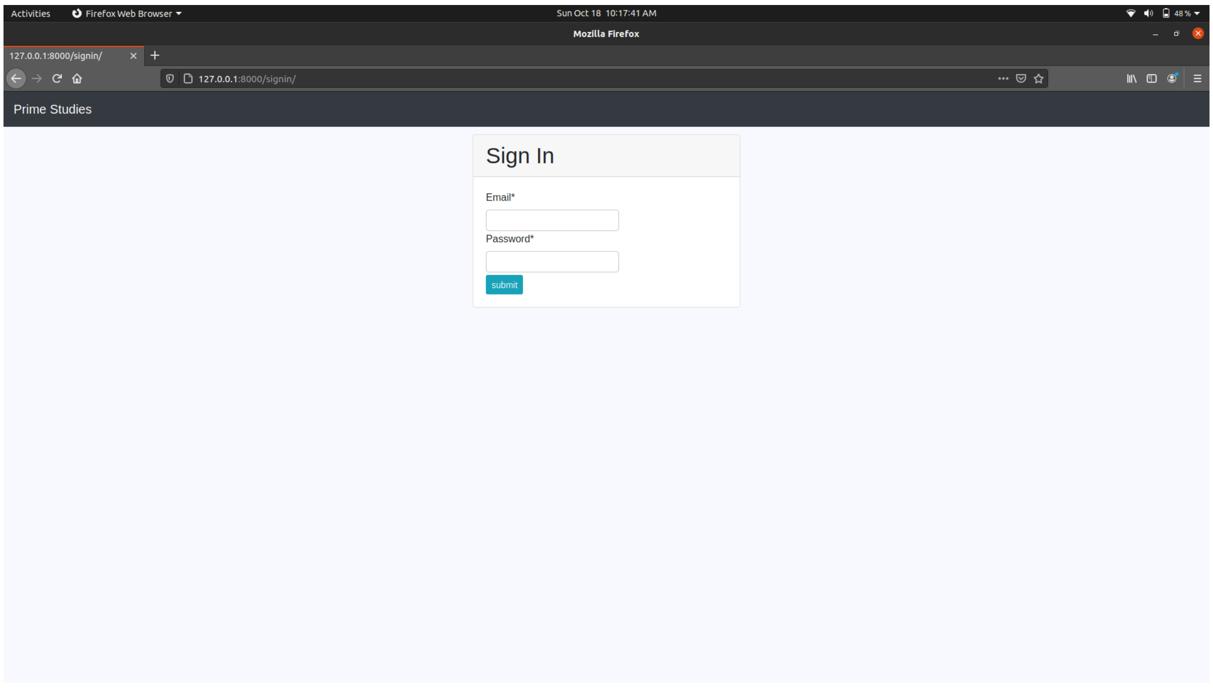
All basic functionalities like adding room, course, student, teacher, assigning the right room to the right teacher and student is done. Students can be directly admitted to a particular room by allowing them to sign up using that particular room's admission link which can then be accepted by the admin.

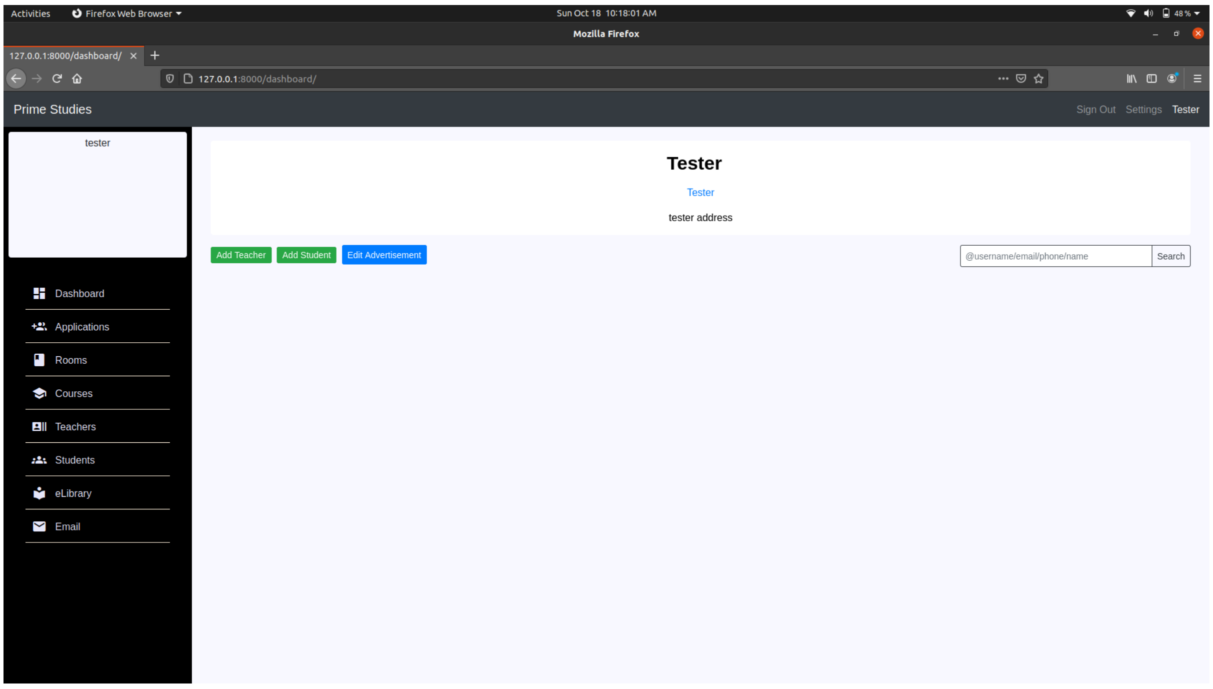
Backend is almost ready (functionalities like an authentication system with only one active session per student, CRUD operations on Rooms, Course, Resources , etc.) . All the rest-api interface is written with more than 50% of them is implemented and tested.

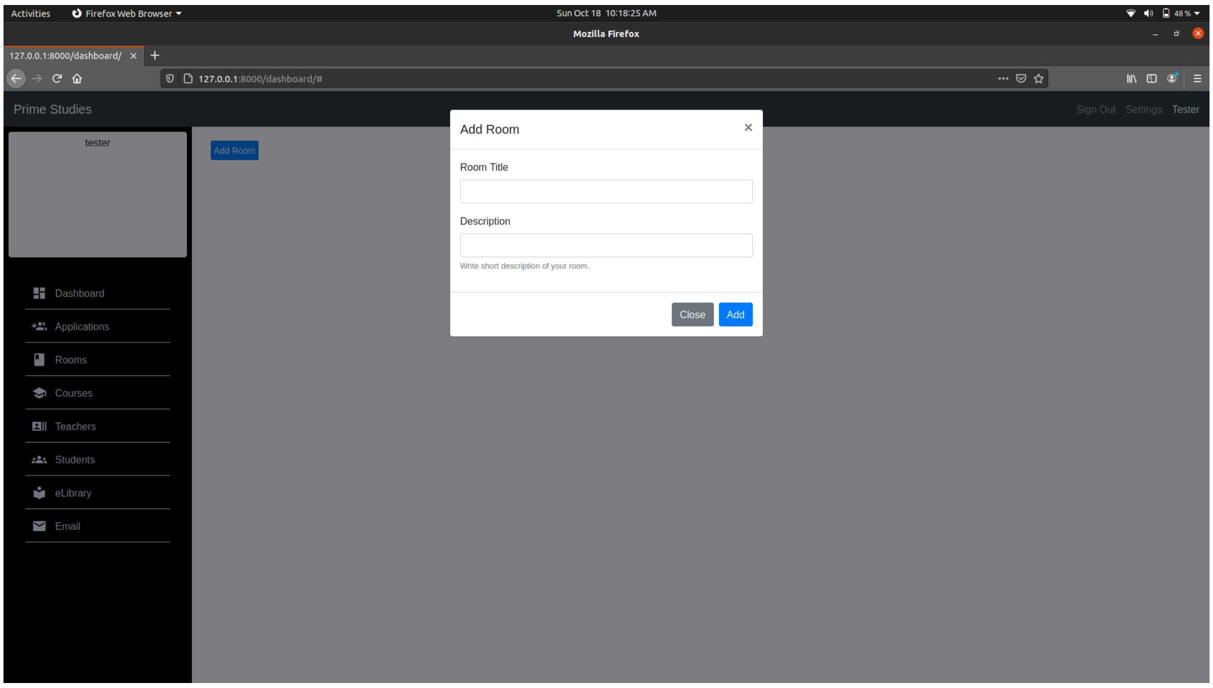
In the front end of the project basic functionalities like adding room, course, resource are implemented for the admin account of the school and some basic front end functionalities are also implemented for the student accounts as well.

Some screenshots are added below from the working part of the project and some demo videos are also added in the github repository of the project.




Activities
Firefox Web Browser
Sun Oct 18 10:17:41 AM
Mozilla Firefox
127.0.0.1:8000/signin/
Prime Studies
Sign In
Email*
Password*
submit


Activities
Firefox Web Browser
Sun Oct 18 10:18:01 AM
Mozilla Firefox
127.0.0.1:8000/dashboard/
Prime Studies
Sign Out
Settings
Tester
tester
Tester
Tester
tester address
Add Teacher
Add Student
Edit Advertisement
@username/email/phone/name
Search
Dashboard
Applications
Rooms
Courses
Teachers
Students
eLibrary
Email

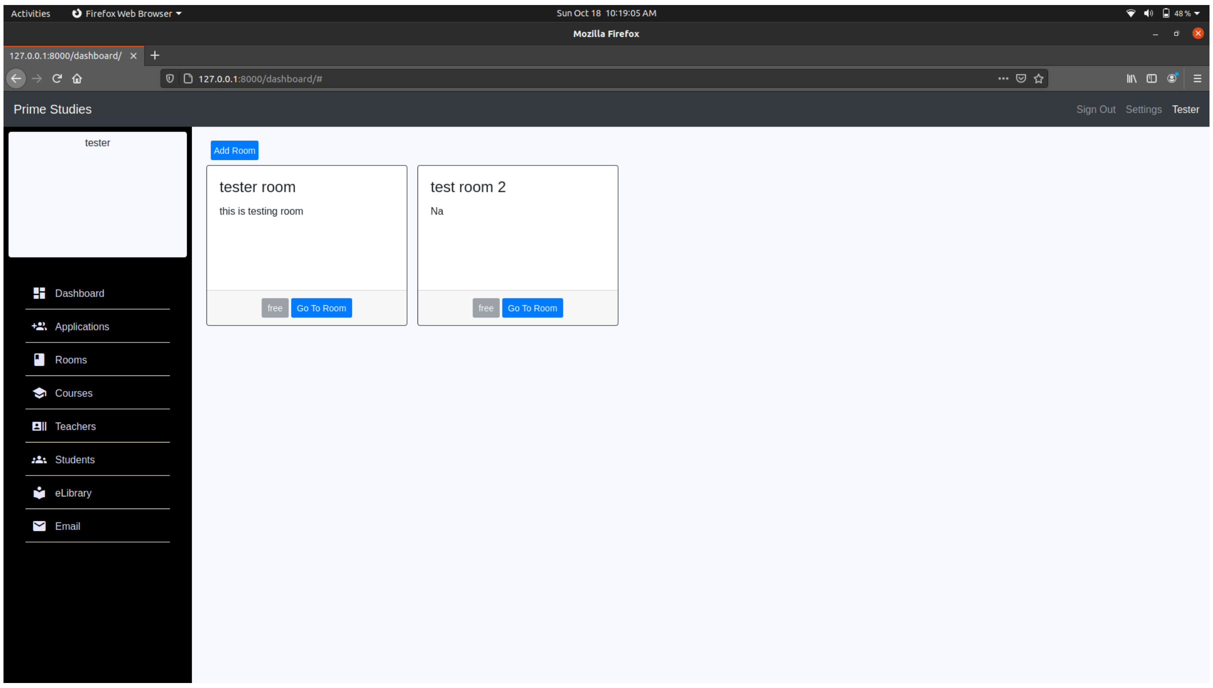


Add Room

Room Title

Description

Write short description of your room.

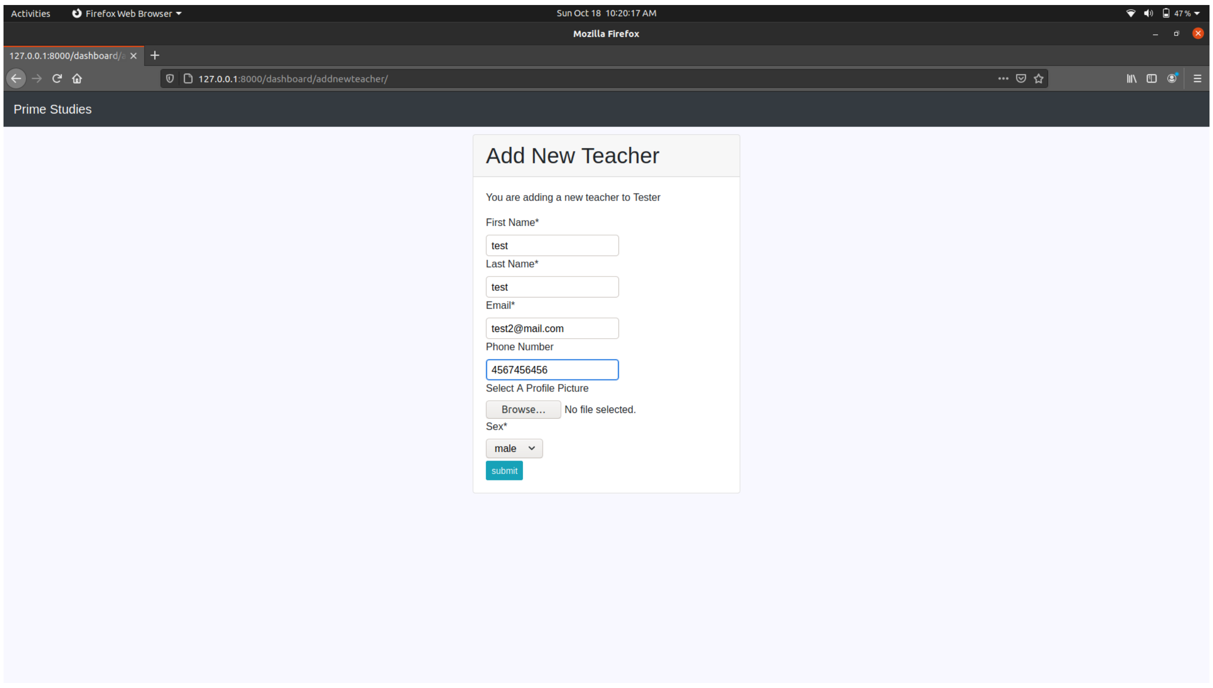


tester room

this is testing room

test room 2

Na



Add New Teacher

You are adding a new teacher to Tester

First Name*: test

Last Name*: test

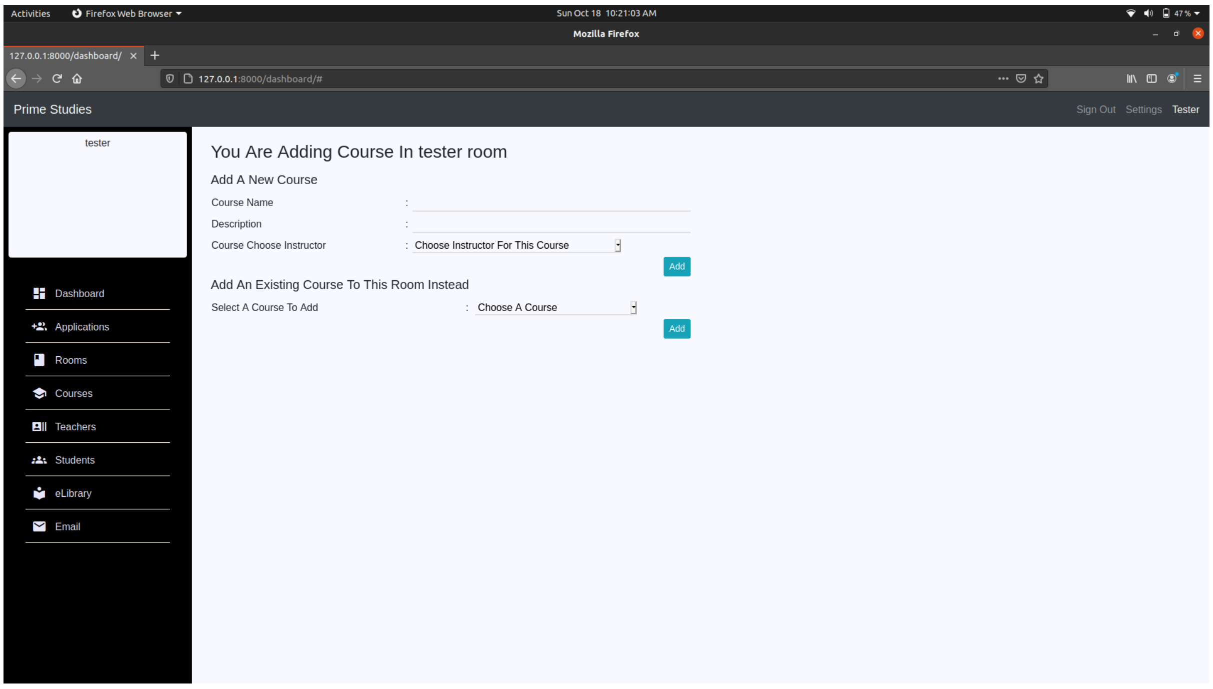
Email*: test2@mail.com

Phone Number: 4567456456

Select A Profile Picture: Browse... No file selected.

Sex*: male

submit



You Are Adding Course In tester room

Add A New Course

Course Name :

Description :

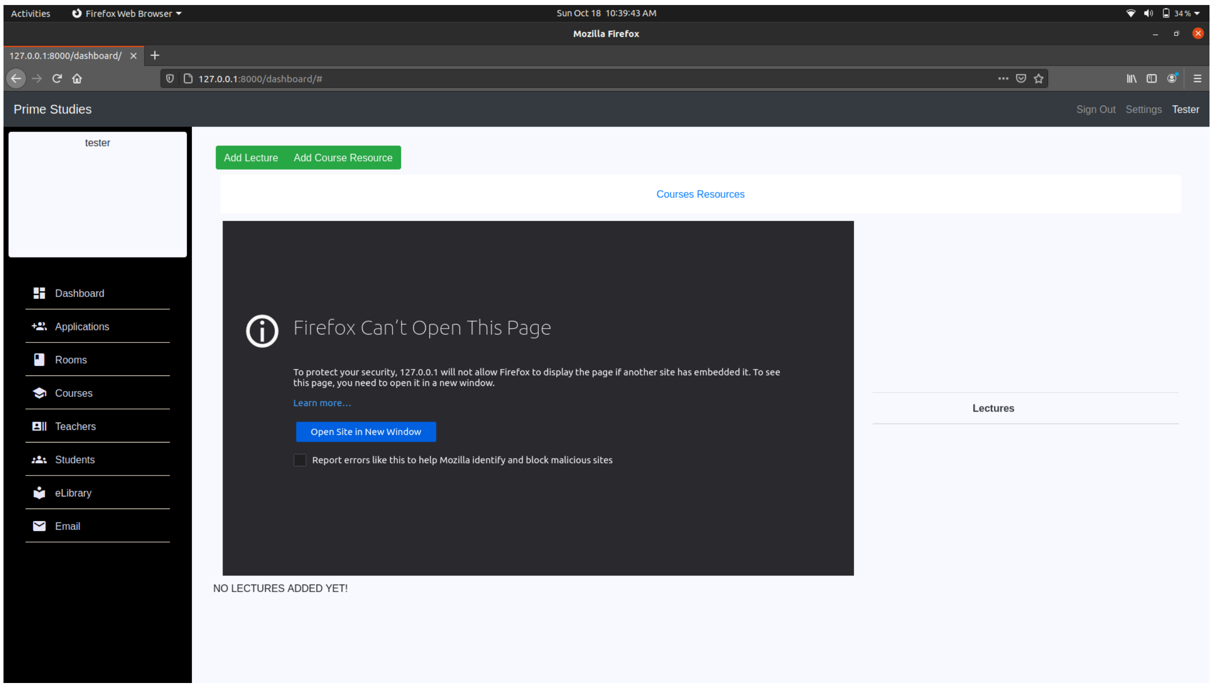
Course Choose Instructor : Choose Instructor For This Course

Add

Add An Existing Course To This Room Instead

Select A Course To Add : Choose A Course

Add


Activities   Firefox Web Browser   Sun Oct 18 10:39:43 AM

Mozilla Firefox

127.0.0.1:8000/dashboard/#

Prime Studies   Sign Out   Settings   Tester

tester

Dashboard
Applications
Rooms
Courses
Teachers
Students
eLibrary
Email

Add Lecture   Add Course Resource

Courses Resources

Firefox Can't Open This Page

To protect your security, 127.0.0.1 will not allow Firefox to display the page if another site has embedded it. To see this page, you need to open it in a new window.

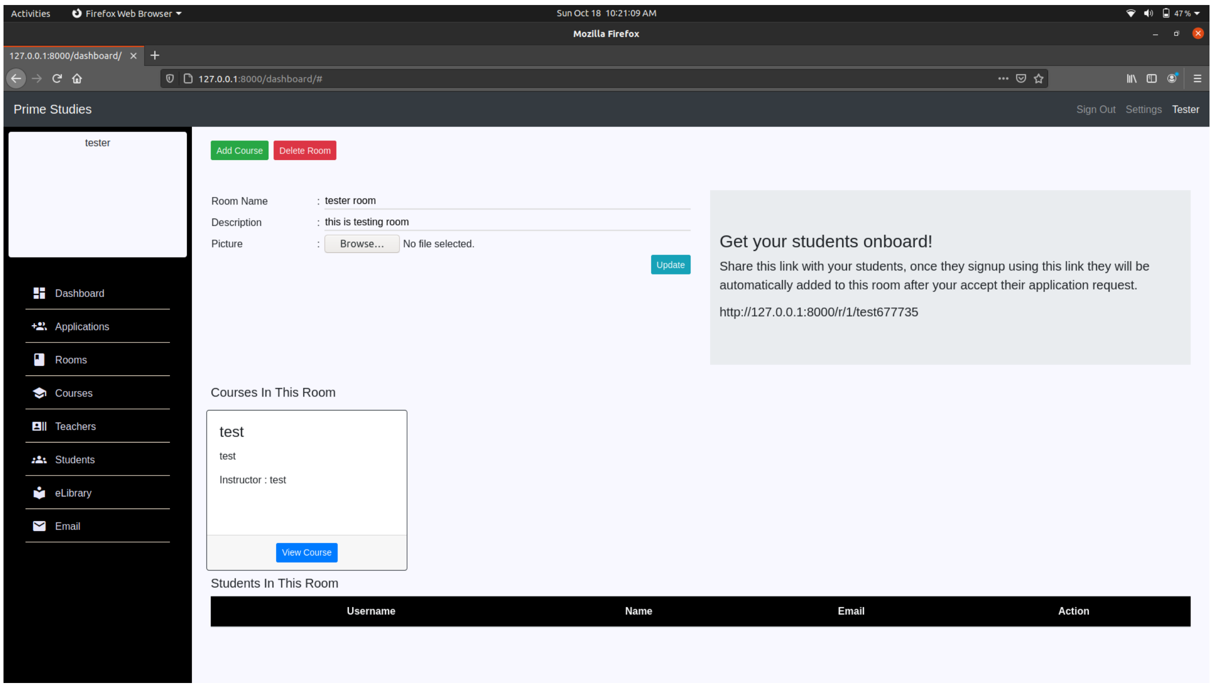
Learn more…

Open Site in New Window

Report errors like this to help Mozilla identify and block malicious sites

Lectures

NO LECTURES ADDED YET!


Activities   Firefox Web Browser   Sun Oct 18 10:21:09 AM

Mozilla Firefox

127.0.0.1:8000/dashboard/#

Prime Studies   Sign Out   Settings   Tester

tester

Dashboard
Applications
Rooms
Courses
Teachers
Students
eLibrary
Email

Add Course   Delete Room

Room Name	:	tester room
Description	:	this is testing room
Picture	:	Browse… No file selected.

Update

Get your students onboard!

Share this link with your students, once they signup using this link they will be automatically added to this room after your accept their application request.

http://127.0.0.1:8000/r/1/test677735

Courses In This Room

test

test

Instructor : test

View Course

Students In This Room

Username	Name	Email	Action

<u>Pending Work :</u>

In the backend time-table ms and live class scheduler, mail and library management is left.

In the front end ui for the student and teacher is left and some features are yet to be implemented in the admin account as well.

<u>References</u>:

- https://docs.djangoproject.com/en/3.1/
- https://www.youtube.com/
- https://www.w3schools.com/html/
- https://www.w3schools.com/js/js_htmldom_document.asp