Information Technology Lab

Student – Teacher Dashboard (Django)

**SUBMITTED BY** 

**SUJEET SANJAY AMBERKAR 200905092**

**Under the Guidance of:**

**Dr. Roopalkshmi R and Ms Priya Kamath**

**Department of Computer Science and Engineering**    
**Manipal Institute of Technology, Manipal Karnataka – 576104**

**April 2024**



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Manipal

05/03/2024

CERTIFICATE

This is to certify that the project titled Student Teacher Dashboard is a record of the bonafide work done by Student Sujeet Sanjay Amberkar   
(Reg. No. 200905092 ) submitted in partial fulfilment of the requirements for the award of the Degree of Bachelor of Technology (B.Tech.) in COMPUTER SCIENCE & ENGINEERING of Manipal Institute of Technology, Manipal, Karnataka, (A Constituent Institute of Manipal Academy of Higher Education), during the academic year 2022-2023.

Name and Signature of Examiners:

1. Dr. Roopalakshmi R, Associate Professor, CSE Dept.
2. Prof. Priya Kamath, Assistant Professor, CSE Dept.

Index

|  |  |  |
| --- | --- | --- |
| **Sr NO** | **Heading** | **Page Number** |
| **1** | Abstract | 4 |
| **2** | Introduction | 4 |
| **3** | File Structure | 5 |
| **4** | Methodology | 6- 7 |
| **5** | Data Flow Diagram | 8 - 9 |
| **6** | Code | 10-11 |
| **5** | Results And Snapshots | 12-15 |
| **6** | Conclusion | 16 |
| **7** | Limitation and Future Work | 16 |
| **8** | Refrences | 16 |
| **9** | GitHub Link | 16 |

# Abstract

The aim of the project “Student-Teacher Dashboard” is to create a structured communication channel between teachers and students. It facilitates various academic and administrative functions, including notice dissemination, access to course materials, and display of marks. This report documents the project’s functionalities, steps of installation and limitations.

# Introduction

The student–teacher dashboard is the one-stop solution for all the communication between teacher and student. Using this software, teachers will be able to share study material, manage marks and also publish notices for students.

# File Structure

A screen shot of a computer

Description automatically generated

# Methodology

## Project Structure

The project is structured into a main Django project named "Student-Teacher dashboard," encompassing two key applications: student and teacher.’’

* Student Application: Manages student interactions with the platform, including login, notice viewing, access to course materials, and marks review.
* Teacher Application: Handles teacher-related functionalities such as login, posting notices, uploading course materials, and entering student marks.

## Implementation

### Groups and Permissions

For Students:

1. **Login**: Secure authentication to access personal and course-related information.
2. **Master**: This views the entire dashboard for students. It contains all the notices, lecture notes, video lectures and marks obtained by that particular student.

For Teachers:

1. **Login**: Secure authentication for accessing administrative interfaces.
2. **Post Notices**: Upload notices relevant to students.
3. **Upload Course Materials**: Share educational resources with students.
4. **Enter Marks**: Record and publish student performance metrics.

### User Management

1. The "Dashboard" project employs a centralized user management system to maintain control ensure secure access prevent any unauthorized access and maintain the integrity of the records.
2. There is no signup page, The superuser will create the username and password and give it to students and teachers

## Database Design

### User Management

1. We are using the SQLite database to manage users and groups, which is facilitated through the auth\_user and auth\_group tables.

Teachers have an option to edit data but students can only view the data

### Notices Table:

1. id: Primary key.
2. title: The title of the notice.
3. content: The detailed content of the notice.
4. posted\_by: Foreign key linking to the auth\_user table to identify the teacher.
5. created\_at: Timestamp indicating when the notice was posted.

### Course Material

This includes materials uploaded by teachers, such as chapter names, YouTube URLs, and notes in PDF format.

1. CourseMaterial Table:
2. id: Primary key.
3. chapter\_name: The name of the chapter.
4. video\_url: URL of the corresponding YouTube video.
5. pdf\_file: FileField to store the uploaded PDF file.
6. created\_at: Timestamp for when the material was uploaded.

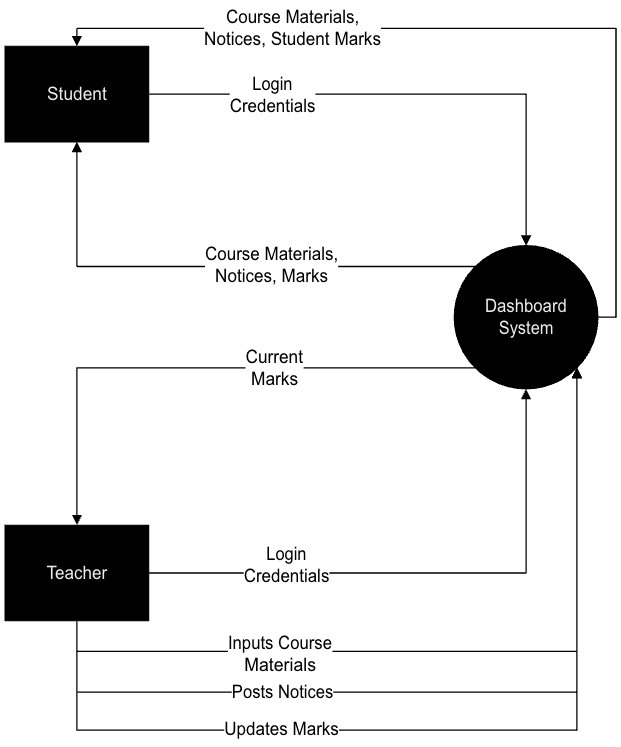
### Marks

1. Student (Foreign Key)
2. Marks Obtained
3. Total Marks

To manage migrations and interact with the database, Django provides the following commands:

* python manage.py makemigrations: Prepares migrations based on model changes.
* python manage.py migrate: Applies migrations to the database, structuring it according to defined models.

# Data Flow Diagram



Level 0 : DFD Diagram

A screenshot of a computer

Description automatically generated   
Level 1 DFD Diagram

# Code:

## teacher/views.py

from django.shortcuts import render,redirect

from django.http import HttpResponse

from django.shortcuts import redirect

from django.contrib.auth import authenticate, login

from django.contrib.auth.forms import AuthenticationForm

from .models import Notice , StudentMarks

from .forms import CourseMaterialForm, MarksForm

from django.contrib.auth.models import User

from django.forms import modelformset\_factory

from django.contrib.auth.decorators import login\_required

from django.contrib.auth.models import User, Group

def home(request):

return render(request, 'teacher/home.html')

def teacher\_login(request):

if request.method == 'POST':

form = AuthenticationForm(request, request.POST)

if form.is\_valid():

username = form.cleaned\_data.get('username')

password = form.cleaned\_data.get('password')

user = authenticate(username=username, password=password)

if user is not None and user.groups.filter(name='teachers').exists():

login(request, user)

return redirect('teacher\_home')

else:

return render(request, 'teacher/login.html', {'form': form, 'error': 'Invalid credentials or you do not have permission to access this page.'})

else:

form = AuthenticationForm()

return render(request, 'teacher/login.html', {'form': form})

@login\_required

def teacher\_notice(request):

if request.method == 'POST':

notice\_content = request.POST.get('notice')

if notice\_content:

Notice.objects.create(content=notice\_content)

return redirect('teacher\_home')

else:

return render(request, 'teacher/teachernotice.html', {

'error': 'Please enter some text for the notice.'

})

return render(request, 'teacher/teachernotice.html')

def add\_course\_material(request):

if request.method == 'POST':

form = CourseMaterialForm(request.POST, request.FILES)

if form.is\_valid():

form.save()

return redirect('teacher\_home') # Adjust as necessary

else:

form = CourseMaterialForm()

return render(request, 'teacher/teachercoursematerial.html', {'form': form})

@login\_required

def enter\_update\_marks(request):

student\_group = Group.objects.get(name='students')

students = student\_group.user\_set.all()

if request.method == 'POST':

for student in students:

marks\_obtained = request.POST.get(f'marks\_obtained\_{student.id}')

total\_marks = request.POST.get(f'total\_marks\_{student.id}')

StudentMarks.objects.update\_or\_create(

student=student,

defaults={

'marks\_obtained': marks\_obtained,

'total\_marks': total\_marks

}

)

return redirect('enter\_update\_marks')

marks = {mark.student\_id: mark for mark in StudentMarks.objects.filter(student\_\_in=students)}

return render(request, 'teacher/teachermarks.html', {'students': students, 'marks': marks})

## teacher/ forms.py

from django import forms

from .models import CourseMaterial

from .models import StudentMarks

class CourseMaterialForm(forms.ModelForm):

class Meta:

model = CourseMaterial

fields = ['chapter\_name', 'pdf\_file', 'video\_url']

class MarksForm(forms.ModelForm):

class Meta:

model = StudentMarks

fields = ['student', 'marks\_obtained', 'total\_marks']

teacher/ models.py

from django.db import models

from django.contrib.auth.models import User

from django.conf import settings

class Notice(models.Model):

content = models.TextField()

created\_at = models.DateTimeField(auto\_now\_add=True)

class CourseMaterial(models.Model):

chapter\_name = models.CharField(max\_length=255)

pdf\_file = models.FileField(upload\_to='course\_materials/pdfs/')

video\_url = models.URLField(max\_length=1024)

def \_\_str\_\_(self):

return self.chapter\_name

class StudentMarks(models.Model):

student = models.ForeignKey(settings.AUTH\_USER\_MODEL, on\_delete=models.CASCADE, related\_name='marks')

marks\_obtained = models.IntegerField(default=0)

total\_marks = models.IntegerField(default=100)

def \_\_str\_\_(self):

return f"{self.student.username} - {self.marks\_obtained}/{self.total\_marks}"

student/views.py

from django.shortcuts import render

from teacher.models import Notice

from django.shortcuts import render, redirect

from django.contrib.auth import authenticate, login, login\_required

from django.contrib.auth.forms import AuthenticationForm

from django.http import HttpResponseRedirect

from teacher.models import CourseMaterial, StudentMarks

def student\_login(request):

if request.method == 'POST':

form = AuthenticationForm(request, request.POST)

if form.is\_valid():

username = form.cleaned\_data.get('username')

password = form.cleaned\_data.get('password')

user = authenticate(username=username, password=password)

if user is not None and user.groups.filter(name='students').exists():

login(request, user)

return redirect('master') # Redirect to the student's home page

else:

return render(request, 'student/login.html', {'form': form, 'error': 'Invalid credentials or you do not have permission to access this page.'})

else:

form = AuthenticationForm()

return render(request, 'student/login.html', {'form': form})

@login\_required

def master\_view(request):

notices = Notice.objects.all().order\_by('-created\_at')

materials = CourseMaterial.objects.all()

try:

marks\_instance = StudentMarks.objects.get(student=request.user)

percentage = (marks\_instance.marks\_obtained / marks\_instance.total\_marks) \* 100

except StudentMarks.DoesNotExist:

marks\_instance = None

percentage = None

context = {

'user': request.user,

'notices': notices,

'materials': materials,

'marks\_instance': marks\_instance,

'percentage': percentage

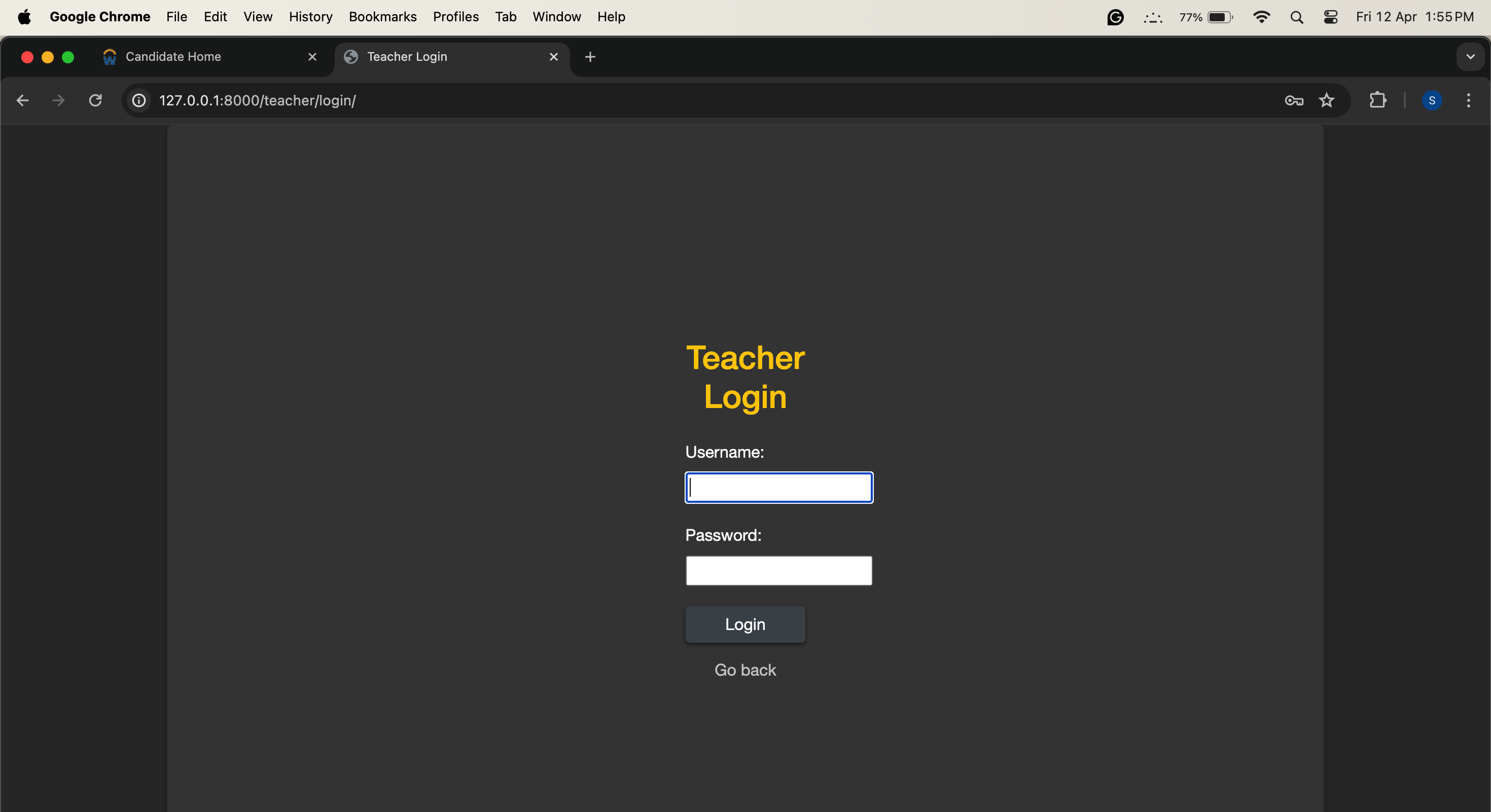
}

return render(request, 'student/master.html', context)

# Results And Snapshots

A screenshot of a computer

Description automatically generated

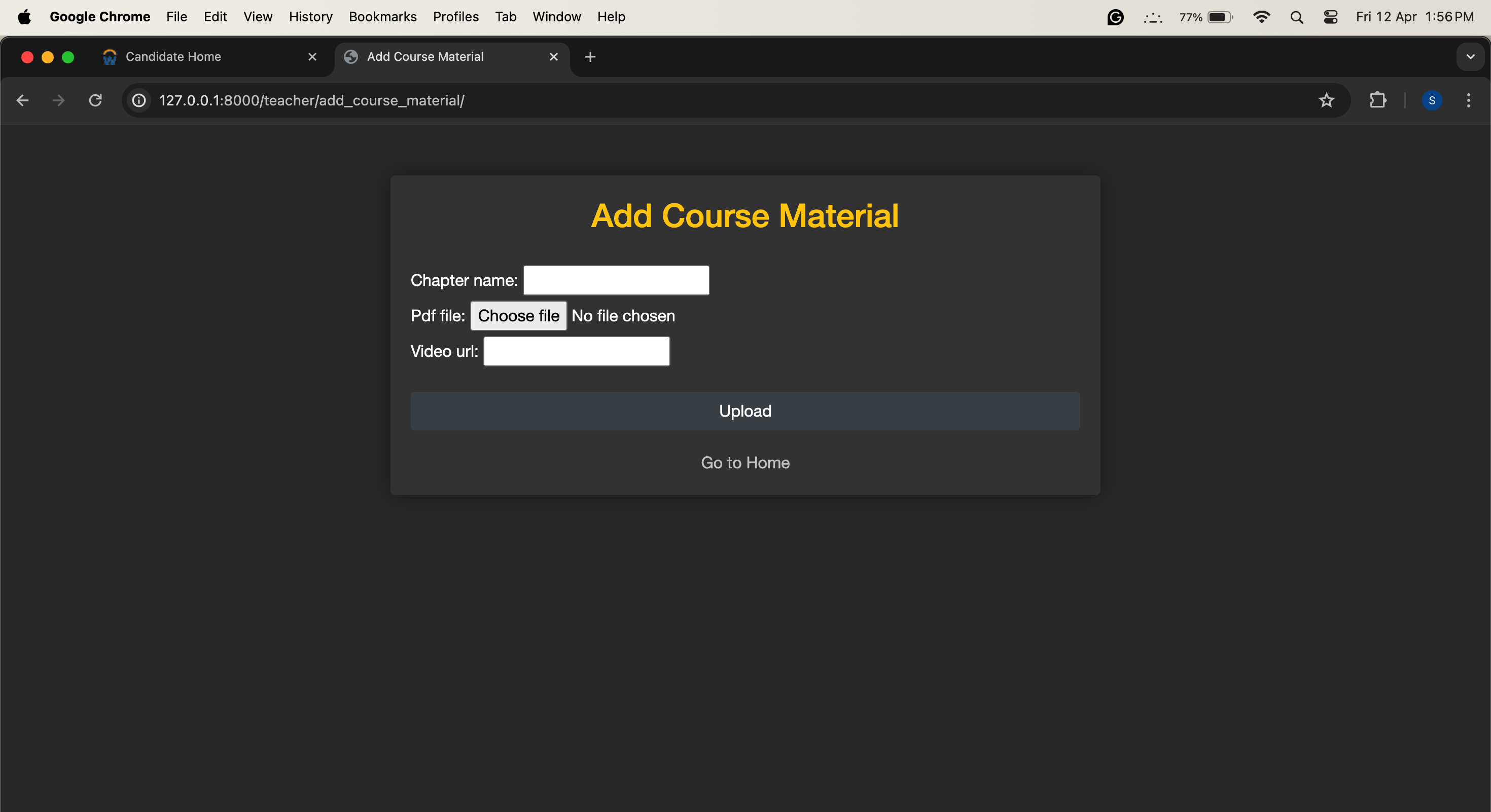


A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated



A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

# Conclusion

The "Dashboard" project represents a significant step forward in simplifying the formal communication between teachers and students.

By leveraging the power of Django we are able to create a user friendly platform to share study material, notices and displaying maths

# Future Directions

1. **Feature Expansion**: More features can be added to this project like daily attendance and a way for students to submit their assignments
2. **Use Media**- Query to make give better user experience of mobiles

# References

1. Django Documentation:

<https://docs.djangoproject.com/en/5.0/>

1. Bootstrap Documentation:  
   <https://getbootstrap.com/docs/4.1/getting-started/introduction/>

# Github :

<https://github.com/sujeetamberkar/it_Project_Dashboard>