MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
*(A constituent unit of MAHE, Manipal)*

# Mini Project Report
**of**
# Internet Technologies Lab (CSE 3262)

# Development of a Centralized Information Management System for the Department of Computer Science and Engineering at Manipal Institute of Technology

**SUBMITTED**
**BY**


**Dhruv Bajaj 210905202, 38, A**
**Lakshay Saxena 210905384, 59, A**
**Disha Agarwal 210905412, 62, A**



**Department of Computer Science and Engineering**
**Manipal Institute of Technology, Manipal.**

**April 2024**

MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
*(A constituent unit of MAHE, Manipal)*

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Manipal**
**10/04/2024**

# CERTIFICATE

This is to certify that the project titled **Development of a Centralized Information Management System for the Department of Computer Science and Engineering at Manipal Institute of Technology** is a record of the bonafide work done by **Dhruv Bajaj (Reg. No. 210905202), Lakshay Saxena (Reg. No. 210905384), Disha Agarwal (Reg. No. 210905412)** 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. , Assistant Professor, CSE Dept.**

# TABLE OF CONTENTS

# ABSTRACT

This project aimed to address the outdated online presence of the Department of Computer Science and Engineering (CSE) at Manipal Institute of Technology (MIT). The solution involved developing a comprehensive website built using Django, a Python web framework, alongside Bootstrap and jQuery for design and interactivity.

The website offers a centralized platform for managing and disseminating departmental information. Faculty and authorized personnel can now seamlessly update course offerings and departmental publications, ensuring current and readily accessible information. Prospective and current students benefit from dedicated sections showcasing faculty profiles, student body size, program information, and crucial departmental updates.

To facilitate secure access and information management, the website implements a login system with differentiated access levels for students, faculty, and administrators. Additionally, the website incorporates timetables for all branches within the CSE department, streamlining course planning for students and faculty.

Overall, the project successfully established a user-friendly and feature-rich website that strengthens the online presence of the CSE department. This platform fosters improved communication, information management, and a more robust online representation of the department's academic excellence.

# CHAPTER 1: INTRODUCTION

This project aimed to revitalize the online presence of the Department of Computer Science and Engineering (CSE) at Manipal Institute of Technology (MIT). The result is a comprehensive and user-friendly website built using Django, a high-level Python web framework, along with Bootstrap for responsive design and jQuery for interactivity.

The website offers a centralized platform for managing and disseminating departmental information. Faculty and authorized personnel can now seamlessly display and modify course offerings, ensuring the curriculum remains up to date. Similarly, the system facilitates the publication of departmental research, showcasing the department's intellectual contributions.

The website also serves as a valuable resource for prospective and current students. Features include the ability to view information on faculty members, student body size, and the various branches of computer science offered at MIT. Additionally, a dedicated section allows for displaying crucial updates and announcements directly to students.

To ensure a secure and organized user experience, the website implements a login system with differentiated access levels for students, faculty, and administrators. This tiered approach grants authorized personnel the ability to modify and update departmental information, while students can access relevant resources and updates. Finally, the website incorporates timetables for all branches within the CSE department. This readily available information streamlines course planning and scheduling for students and faculty alike. Overall, the newly developed website serves as a central hub for the CSE department at MIT, fostering improved communication, information dissemination, and a more robust online presence.

# CHAPTER 2: PROBLEM STATEMENT & OBJECTIVES

The Department of Computer Science and Engineering (CSE) at Manipal Institute of Technology (MIT) lacked a centralized and user-friendly platform for managing and disseminating departmental information. The existing website was outdated and did not effectively cater to the needs of students, faculty, and prospective students. This presented several challenges:

1. **Inconsistent and Outdated Information:** The department lacked a central location to maintain and update course offerings, faculty profiles, and departmental news. This resulted in information being scattered across different sources, potentially leading to confusion and frustration for users.
2. **Limited Communication Channels:** The department needed a more effective way to communicate with students and prospective students. The existing website did not provide a dedicated space for displaying important updates, announcements, and program information.
3. **Inefficient Information Management:** Updating and presenting departmental information, such as faculty profiles, publications, and student body size, was a cumbersome process using the existing system. This inefficiency hindered the department's ability to showcase its academic achievements and offerings.
4. **Lack of Access Control:** The existing website did not offer a secure login system, making it difficult to manage access to sensitive information.

To address these challenges, the project aimed to develop a comprehensive website with the following objectives:

1. **Centralized Information Management System:** Create a user-friendly platform for authorized personnel to manage and update course offerings, faculty profiles, departmental publications, and student information.
2. **Improved Communication Channels:** Develop dedicated sections for displaying departmental news, announcements, and program information directly to students and prospective students.
3. **Streamlined Information Access:** Provide a central location for students to access timetables, faculty profiles, and other relevant information about the department's various branches.
4. **Enhanced Security:** Implement a secure login system with differentiated access levels for students, faculty, and administrators to ensure data integrity and user privacy.

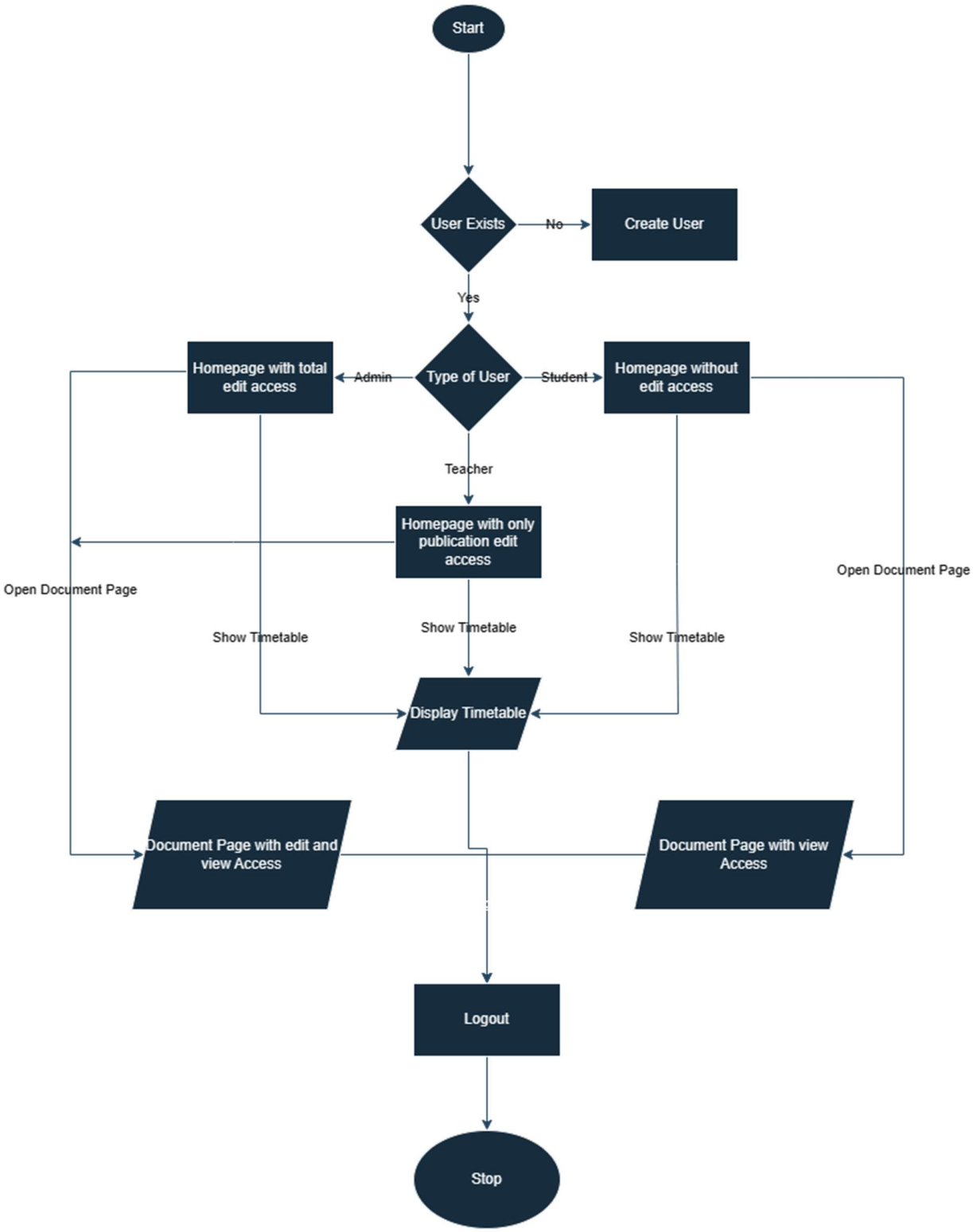
By achieving these objectives, the project aimed to establish a modern and user-friendly website that strengthens the online presence of the CSE department at MIT. This comprehensive platform would facilitate efficient information management, improve communication with students and stakeholders, and better showcase the department's academic achievements.

# CHAPTER 3: METHODOLOGY

The development of the CSE department website followed a structured methodology encompassing four key phases:

1. **Ideation and Requirements Gathering:** The project began with extensive research both online and offline to understand the specific needs and expectations of the faculty, administrators and students for the website. This phase involved brainstorming sessions to identify the functionalities crucial for efficient information management and improved user experience. Following these discussions, a detailed requirements document was created, outlining the website's features, functionalities, and user access levels.
2. **UI Design and Database Modelling:** Building upon the established requirements, the project team focused on designing a user-friendly and visually appealing interface. Mockups and wireframes were created utilizing a design framework like Bootstrap, ensuring responsiveness across various devices. Concurrently, a robust database model was developed using a relational database management system. This model effectively stored and organized departmental data, including course information, faculty profiles, publications, student enrollment numbers, and departmental updates.
3. **Coding and Unit Testing:** The website was developed using Django, a high-level Python web framework. Django's functionalities streamlined development, particularly with regards to user authentication, database interaction, and URL routing. Throughout the coding process, a rigorous unit testing approach was implemented. Individual code modules were tested in isolation to ensure they functioned as intended, mitigating potential errors and bugs.
4. **System Testing and Validation:** Following the completion of individual code modules, comprehensive system testing was conducted. This involved testing all functionalities, user logins, and integration between different components of the website. Additionally, user acceptance testing was performed, involving faculty members and students to obtain feedback on the website's usability and functionality. Based on this feedback, necessary refinements were made to ensure the website met the needs of its target audience.

This iterative development process, encompassing ideation, design, coding, testing, and validation, culminated in the creation of a user-friendly and feature-rich website that effectively serves the needs of the CSE department at MIT.


Start
User Exists
No
Create User
Yes
Homepage with total edit access
Admin
Type of User
Student
Homepage without edit access
Teacher
Homepage with only publication edit access
Open Document Page
Open Document Page
Show Timetable
Show Timetable
Show Timetable
Display Timetable
Document Page with edit and view Access
Document Page with view Access
Logout
Stop

# CHAPTER 4: RESULTS & SNAPSHOTS

urls.py

```
from django.urls import path, include
from . import views
from django.contrib.auth import views as auth_views

urlpatterns = [
    # path('', views.index, name='index'),
    # path('mail/', views.simple, name='simple'),
    # path('password_reset/', auth_views.PasswordResetView.as_view(), name='password_reset'),
    # path('password_reset_done/', auth_views.PasswordResetDoneView.as_view(),
name='password_reset_done'),
    # path('password_reset_confirm/<uidb64>/<token>/',
auth_views.PasswordResetConfirmView.as_view(), name='password_reset_confirm'),
    # path('password_reset_complete/', auth_views.PasswordResetCompleteView.as_view(),
name='password_reset_complete'),

    path('', views.login_view, name='login_view'),
    path('register/', views.register, name='register'),
    # path('logout/', auth_views.LogoutView.as_view(next_page='login_view'), name='logout'),
    path('logout/', views.custom_logout, name='logout'),


    path('homepage/', views.homepage, name='homepage'),
    path('delete_course/<int:course_id>/', views.delete_course, name='delete_course'),
    path('add_course/', views.add_course, name='add_course'),
    path('add_pub/', views.add_pub, name='add_pub'),
    path('delete_pub/<int:pub_id>/', views.delete_pub, name='delete_pub'),

    path('timetable/', views.timetable, name='timetable'),

    path('important_documents/', views.important_documents, name='important_documents'),
    path('view_document/<int:document_id>/', views.view_document, name='view_document'),
    path('delete_document/<int:document_id>/', views.delete_document, name='delete_document'),

]
```

views.py

```
from django.shortcuts import render, redirect, get_object_or_404
from django.template import loader

from .forms import SignUpForm, LoginForm, DocumentForm
from django.contrib.auth import authenticate, login, logout

from django.http import (
    JsonResponse,
    HttpResponseBadRequest,
    HttpResponse,
    HttpResponseServerError,
)
from django.conf import settings
import os

from .models import Document, User, Courses, Pubs
from .forms import PubsForm
```

```python
from django.http import HttpResponse
from django.core.mail import send_mail

from django.contrib.auth.decorators import login_required
from django.http import HttpResponseForbidden


def simple(request):
    send_mail(
        subject="this is a test",
        message="message body",
        from_email="django@demomailtrap.com",
        recipient_list=["cse.department.webpage.2024@gmail.com"],
    )
    return HttpResponse("Mail sent!")


def register(request):
    msg = None
    if request.method == "POST":
        form = SignUpForm(request.POST)
        if form.is_valid():
            username = form.cleaned_data["username"]
            password = form.cleaned_data["password1"]
            email = form.cleaned_data["email"]
            code = form.cleaned_data["code"]
            is_admin = form.cleaned_data["is_admin"]
            is_student = form.cleaned_data["is_student"]
            is_teacher = form.cleaned_data["is_teacher"]

            if is_admin and not code == "123456":
                msg = "Wrong Code"
            elif is_teacher and not code == "654321":
                msg = "Wrong Code"
            else:

                user = User.objects.create_user(username=username, password=password)
                if not (is_admin or is_teacher or is_student):
                    is_student = True
                user.is_admin = is_admin
                user.is_student = is_student
                user.is_teacher = is_teacher
                user.save()
                msg = "User created successfully"
                return redirect("login_view")
        else:
            msg = "form is not valid"
    else:
        form = SignUpForm()
    return render(request, "register.html", {"form": form, "msg": msg})


def login_view(request):
    form = LoginForm(request.POST or None)
    msg = None
    if request.method == "POST":
        if form.is_valid():
            username = form.cleaned_data.get("username")
            password = form.cleaned_data.get("password")
            remember_me = form.cleaned_data.get("remember_me")

            if remember_me:

                request.session.set_expiry(604800)
            else:
```

```
                request.session.set_expiry(0)

            user = authenticate(username=username, password=password)
            if user is not None:
                login(request, user)

                return redirect("homepage")
            else:
                msg = "invalid credentials"
        else:
            msg = "error validating the form"
    elif request.user.is_authenticated:
        return redirect("homepage")
    return render(request, "login.html", {"form": form, "msg": msg})


def custom_logout(request):
    logout(request)
    return redirect("login_view")


def delete_course(request, course_id):
    course = get_object_or_404(Courses, pk=course_id)
    course.delete()
    return redirect("homepage")


def delete_pub(request, pub_id):
    pub = get_object_or_404(Pubs, pk=pub_id)
    pub.delete()
    return redirect("homepage")


def add_course(request):
    if request.method == "POST":
        cname = request.POST.get("cname")
        c_code = request.POST.get("c_code")
        cred = request.POST.get("cred")

        course = Courses(cname=cname, c_code=c_code, cred=cred)
        course.save()
        return redirect("homepage")
    return HttpResponse("Method not allowed", status=405)


def add_pub(request):
    if request.method == "POST":
        auth = request.POST.get("auth")
        pub_title = request.POST.get("pub_title")
        topic = request.POST.get("topic")
        pub_date = request.POST.get("pub_date")
        print(auth, pub_title, topic, pub_date)

        pub = Pubs(auth=auth, pub_title=pub_title, topic=topic, pub_date=pub_date)
        pub.save()
        return redirect("homepage")
    return HttpResponse("Method not allowed", status=405)


@login_required
def homepage(request):
    num_students = User.objects.filter(is_student=True).count()
    num_teachers = User.objects.filter(is_teacher=True).count()
    data = Courses.objects.all().order_by("-id").values()
```

```
    data2 = Pubs.objects.all().order_by("-id").values()
    num_courses = Courses.objects.count()
    num_pubs = Pubs.objects.count()

    is_teacher_perm, is_admin_perm = False, False
    if request.user.is_teacher:
        is_teacher_perm = True
    if request.user.is_admin:
        is_admin_perm = True

    context = {
        "Courses": data,
        "Pubs": data2,
        "Branches": 3,
        "num_students": num_students,
        "num_teachers": num_teachers,
        "num_pubs": num_pubs,
        "num_courses": num_courses,
        "is_teacher_perm": is_teacher_perm,
        "is_admin_perm": is_admin_perm,
    }
    template = loader.get_template("homepage.html")
    return HttpResponse(template.render(context=context, request=request))


@login_required
def timetable(request):
    return render(request, "timetable.html")


@login_required
def important_documents(request):

    if request.user.is_authenticated:

        documents = Document.objects.all().order_by("-id")

        if request.user.is_teacher or request.user.is_admin:
            uploadform = DocumentForm(request.POST, request.FILES)

            if request.method == "POST" and request.FILES.getlist("pdf"):
                pdf_files = request.FILES.getlist("pdf")
                for pdf_file in pdf_files:
                    Document.objects.create(file=pdf_file, name=pdf_file.name)
                return redirect("important_documents")

            return render(
                request,
                "important_documents.html",
                {
                    "documents": documents,
                    "allow_upload_delete": True,
                    "uploadform": uploadform,
                },
            )
        else:

            return render(
                request,
                "important_documents.html",
                {"documents": documents, "allow_upload_delete": False},
            )
    else:

        return redirect("login_view")
```

```
@login_required
def list_documents(request):
    documents = Document.objects.all()
    return render(request, "important_documents.html", {"documents": documents})


@login_required
def view_document(request, document_id):
    document = Document.objects.get(id=document_id)

    document_path = document.file.path
    with open(document_path, "rb") as f:
        response = HttpResponse(f.read(), content_type="application/pdf")
        response["Content-Disposition"] = "inline; filename=" + document.name
    return response


@login_required
def delete_document(request, document_id):
    document = get_object_or_404(Document, id=document_id)
    document_path = document.file.path
    document.delete()
    try:
        os.remove(document_path)
    except OSError as e:

        return HttpResponseServerError("Error deleting file: {}".format(e))
    return redirect("important_documents")
```

forms.py

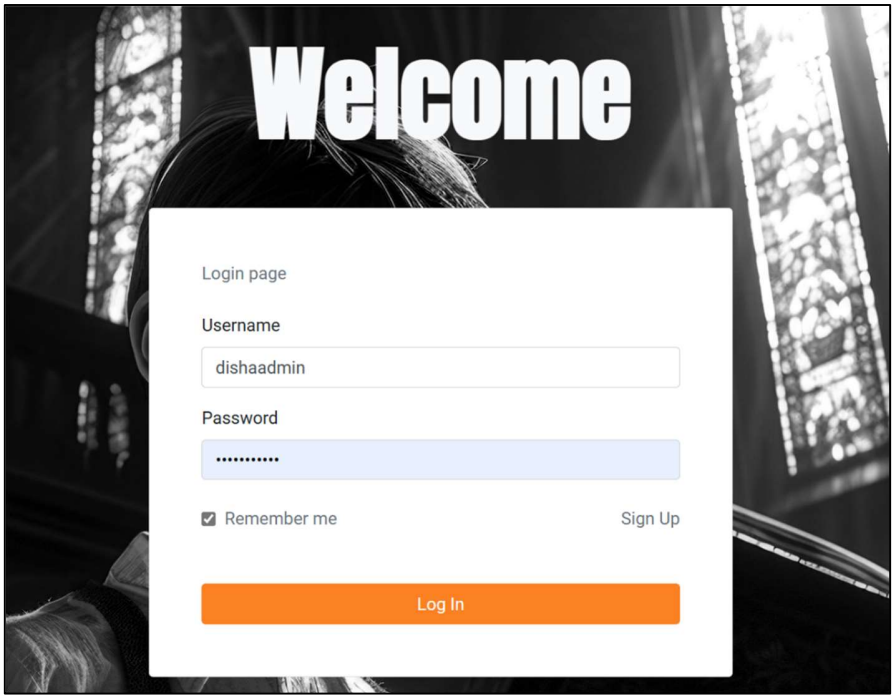
```
from django import forms
from django.contrib.auth.forms import UserCreationForm
from .models import User, Document, Pubs


class LoginForm(forms.Form):
    username = forms.CharField(widget=forms.TextInput(attrs={"class": "form-control"}))
    password = forms.CharField(
        widget=forms.PasswordInput(attrs={"class": "form-control"})
    )
    remember_me = forms.BooleanField(required=False)


class SignUpForm(UserCreationForm):
    username = forms.CharField(widget=forms.TextInput(attrs={"class": "form-control"}))
    password1 = forms.CharField(
        widget=forms.PasswordInput(attrs={"class": "form-control"})
    )
    password2 = forms.CharField(
        widget=forms.PasswordInput(attrs={"class": "form-control"})
    )
    email = forms.CharField(widget=forms.TextInput(attrs={"class": "form-control"}))

    code = forms.CharField(
        required=False, widget=forms.TextInput(attrs={"class": "form-control"})
    )

    class Meta:
        model = User
```

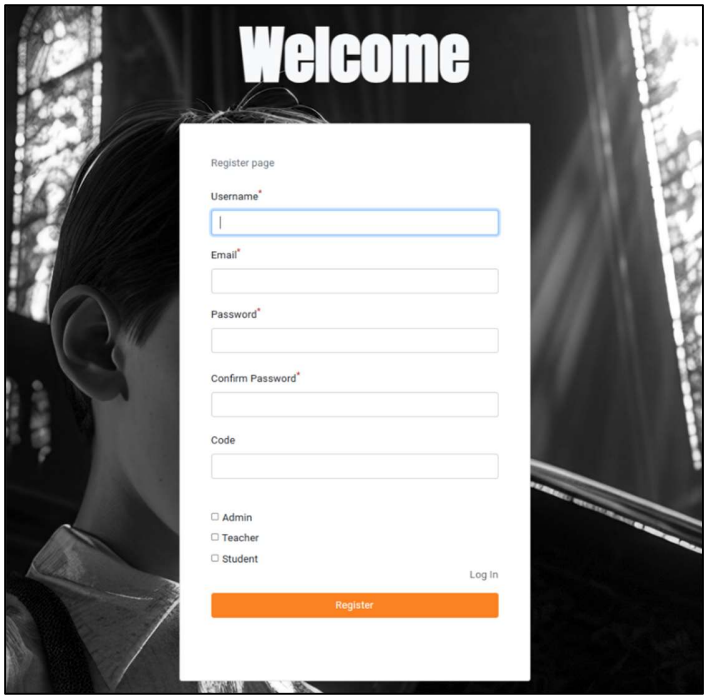
```
        fields = (
            "username",
            "email",
            "password1",
            "password2",
            "code",
            "is_admin",
            "is_teacher",
            "is_student",
        )


class DocumentForm(forms.ModelForm):
    pdf = forms.FileField(
        widget=forms.TextInput(
            attrs={
                "name": "pdf_files",  # Name attribute for the input field
                "type": "file",  # Input type
                "class": "form-control",  # Bootstrap class for styling
                "multiple": True,  # Allowing multiple files to be selected
                "accept": ".pdf",  # Limiting to PDF files
            }
        ),
        label="",
    )  # Label for the field (optional)

    class Meta:
        model = Document  # Assuming PdfModel is the model associated with PDF files
        fields = ["pdf"]  # Field(s) to include in the form


class PubsForm(forms.ModelForm):
    class Meta:
        model = Pubs
        fields = ["auth", "pub_title", "topic", "pub_date"]
```

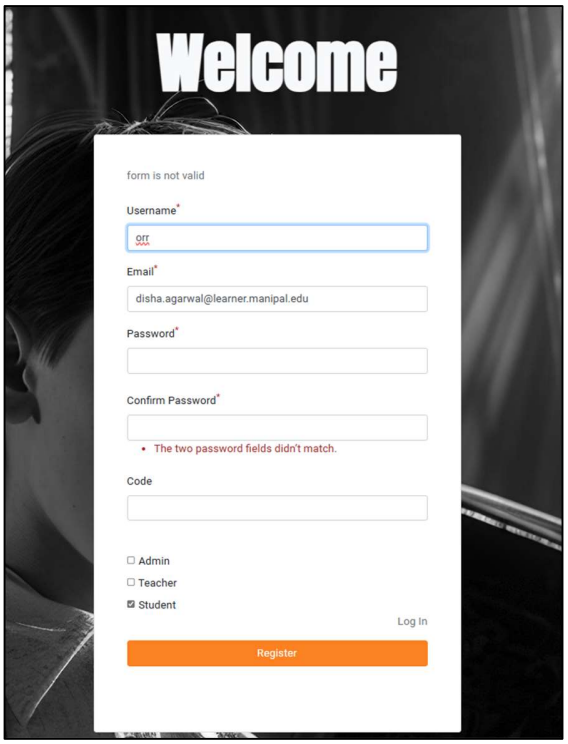
## models.py

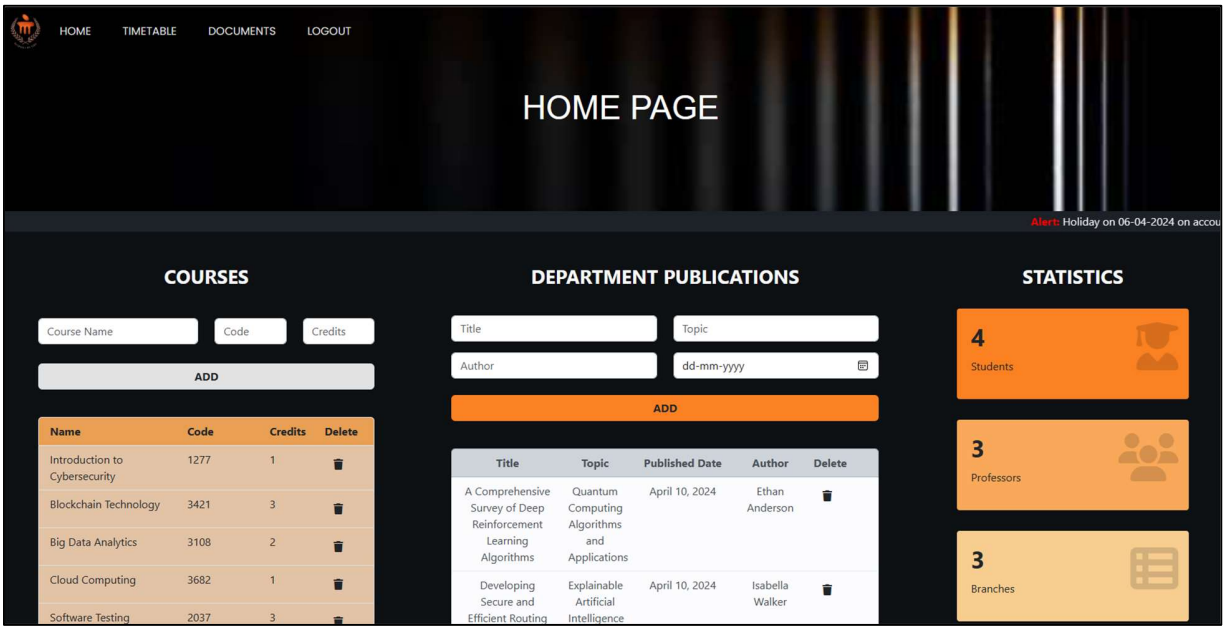
```
from django.db import models
from django.contrib.auth.models import AbstractUser
# Create your models here.

class User(AbstractUser):
    is_admin = models.BooleanField('Is admin', default=False)
    is_student = models.BooleanField('Is student', default=False)
    is_teacher = models.BooleanField('Is teacher', default=False)

class Document(models.Model):
    file = models.FileField(upload_to='documents/')
    name = models.CharField(max_length=255)
    uploaded_at = models.DateTimeField(auto_now_add=True)

class Courses(models.Model):
    class Meta:
        ordering = ['cname']
    cname = models.CharField(max_length=255)
    c_code = models.IntegerField()
    cred = models.IntegerField()

class Pubs(models.Model):
    class Meta:
        ordering = ['auth']
    auth = models.CharField(max_length=1024)
```

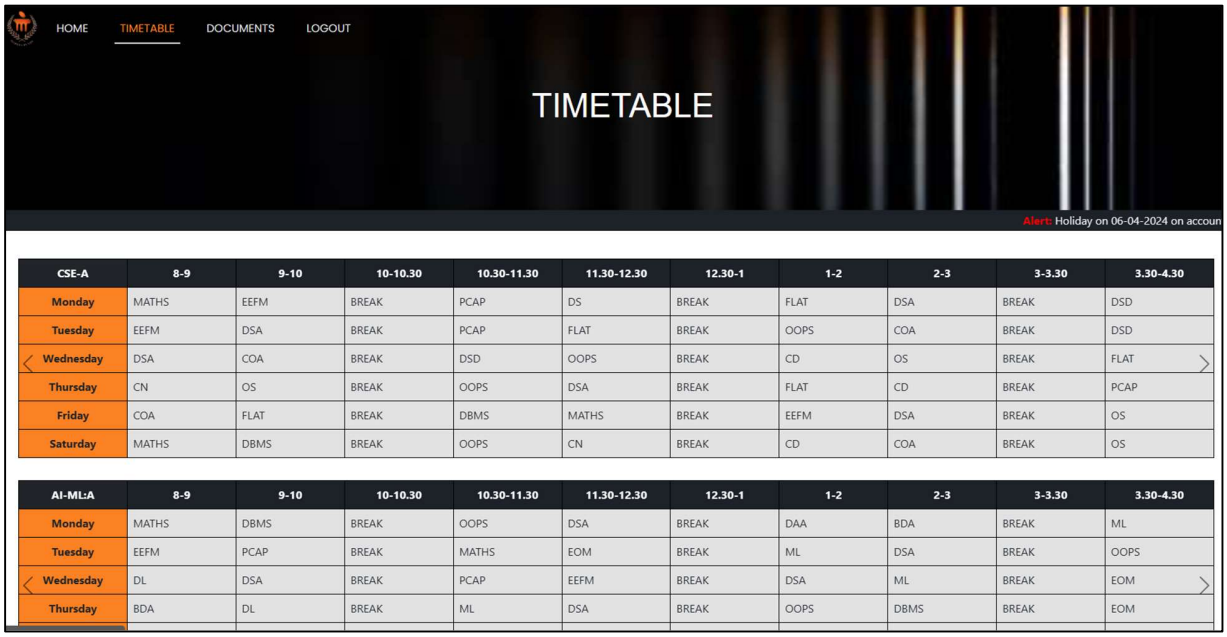
```
    pub_title = models.CharField(max_length=255)
    topic = models.CharField(max_length=255)
    pub_date = models.DateField()
```



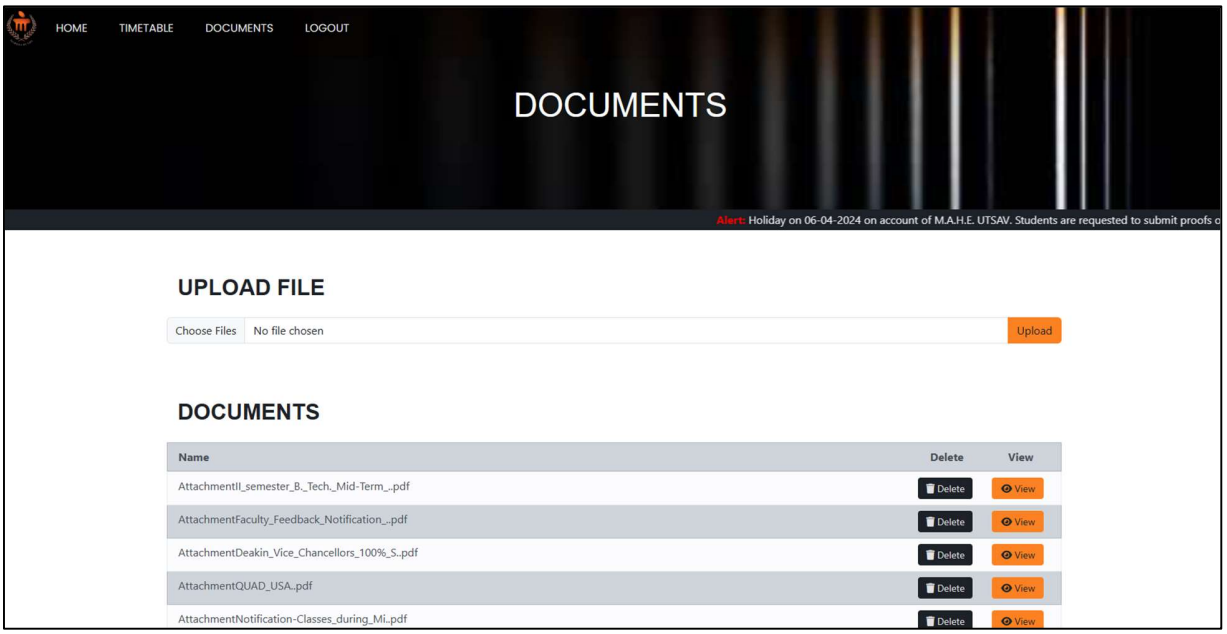






| CSE-A | 8-9 | 9-10 | 10-10.30 | 10.30-11.30 | 11.30-12.30 | 12.30-1 | 1-2 | 2-3 | 3-3.30 | 3.30-4.30 |
|---|---|---|---|---|---|---|---|---|---|---|
| Monday | MATHS | EEFM | BREAK | PCAP | DS | BREAK | FLAT | DSA | BREAK | DSD |
| Tuesday | EEFM | DSA | BREAK | PCAP | FLAT | BREAK | OOPS | COA | BREAK | DSD |
| Wednesday | DSA | COA | BREAK | DSD | OOPS | BREAK | CD | OS | BREAK | FLAT |
| Thursday | CN | OS | BREAK | OOPS | DSA | BREAK | FLAT | CD | BREAK | PCAP |
| Friday | COA | FLAT | BREAK | DBMS | MATHS | BREAK | EEFM | DSA | BREAK | OS |
| Saturday | MATHS | DBMS | BREAK | OOPS | CN | BREAK | CD | COA | BREAK | OS |

| AI-ML:A | 8-9 | 9-10 | 10-10.30 | 10.30-11.30 | 11.30-12.30 | 12.30-1 | 1-2 | 2-3 | 3-3.30 | 3.30-4.30 |
|---|---|---|---|---|---|---|---|---|---|---|
| Monday | MATHS | DBMS | BREAK | OOPS | DSA | BREAK | DAA | BDA | BREAK | ML |
| Tuesday | EEFM | PCAP | BREAK | MATHS | EOM | BREAK | ML | DSA | BREAK | OOPS |
| Wednesday | DL | DSA | BREAK | PCAP | EEFM | BREAK | DSA | ML | BREAK | EOM |
| Thursday | BDA | DL | BREAK | ML | DSA | BREAK | OOPS | DBMS | BREAK | EOM |


HOME
TIMETABLE
DOCUMENTS
LOGOUT
DOCUMENTS
Alert: Holiday on 06-04-2024 on account of M.A.H.E. UTSAV. Students are requested to submit proofs o
UPLOAD FILE
Choose Files
No file chosen
Upload
DOCUMENTS
Name
Delete
View
AttachmentII_semester_B._Tech._Mid-Term_..pdf
AttachmentFaculty_Feedback_Notification_..pdf
AttachmentDeakin_Vice_Chancellors_100%_S..pdf
AttachmentQUAD_USA..pdf
AttachmentNotification-Classes_during_Mi..pdf

# CHAPTER 5: CONCLUSIONS

This project successfully addressed the need for a modernized and centralized online platform for the Department of Computer Science and Engineering (CSE) at Manipal Institute of Technology (MIT). The resulting website, built using Django, Bootstrap, and jQuery, offers a comprehensive suite of features designed to enhance information management, communication, and user experience.

Faculty and authorized personnel can now seamlessly manage and update course offerings, ensuring the curriculum remains current and readily accessible to students. Similarly, the system facilitates the publication of departmental research, showcasing the department's intellectual contributions and fostering a sense of academic achievement.

For prospective and current students, the website serves as a valuable resource center. It provides readily available information on faculty members, student body size, and the various branches of computer science offered at MIT. Additionally, a dedicated section allows for displaying crucial updates and announcements directly to students, streamlining communication and ensuring they are kept informed of important departmental news.

To ensure a secure and organized user experience, the website implements a login system with differentiated access levels for students, faculty, and administrators. This tiered approach grants authorized personnel the ability to modify and update departmental information, while students can access relevant resources and updates pertaining to their academic journey.

Finally, the website incorporates timetables for all branches within the CSE department. This readily available information streamlines course planning and scheduling for students and faculty alike. Overall, the newly developed website serves as a central hub for the CSE department at MIT. It fosters improved communication, information dissemination, and a more robust online presence that reflects the department's academic excellence. This user-friendly platform empowers faculty, students, and prospective students to effectively engage with the department and its offerings.

# CHAPTER 6: LIMITATIONS & FUTURE WORK

Though immense effort has been put in to make this as holistic as possible, there still remains some scope for development and improvement. These limitations are:

1. **Initial Data Population:** Populating the website with existing departmental data was a manual process. Future iterations could explore functionalities for importing data from external sources to streamline this process.
2. **Content Management System Integration:** The current version does not integrate with a Content Management System (CMS). This limits the ease with which non-technical users can update certain content sections. Future development could explore integrating a user-friendly CMS for content management.
3. **Advanced Search Functionality:** The current search functionality is basic. Future iterations could implement more advanced search capabilities to allow users to find specific information more efficiently.

Beyond the current features, the project holds exciting possibilities for future development. One area of focus could be the integration of interactive elements, such as online course registration or the ability for students to submit feedback directly through the website. Additionally, the website's functionality could be expanded to include features like online forums or alumni networking capabilities. Furthermore, exploring the use of real-time data could provide students with dynamic information, such as classroom availability or faculty office hours. By continually evaluating user needs and technological advancements, the CSE department website can evolve into a truly comprehensive and interactive platform that fosters a vibrant online community.

# CHAPTER 7: REFERENCES


[1] Django Project Documentation https://docs.djangoproject.com/en/5.0/
[2] Bootstrap 5 Documentation https://getbootstrap.com/
[3] jQuery API Documentation https://api.jquery.com/
[4] W3Schools - Python Tutorial https://www.w3schools.com/python/
[5] W3Schools - Django Tutorial https://www.w3schools.com/django/
[6] W3Schools - JQuery Tutorial https://www.w3schools.com/jquery/default.asp
[7] W3Schools - Bootstrap Tutorial https://www.w3schools.com/bootstrap/
[8] Django Girls Tutorial https://tutorial.djangogirls.org/en/
[9] Real Python - Django Tutorial https://realpython.com/tutorials/django/
[10] Mozilla Developer Network - Web Development Guide https://developer.mozilla.org/
[11] The Official jQuery Website https://jquery.com/
[12] The Django Packages Project https://pypi.org/project/Django/