DJANGO AUTHENTICATION & AUTHORIZATION

CONTENTS:

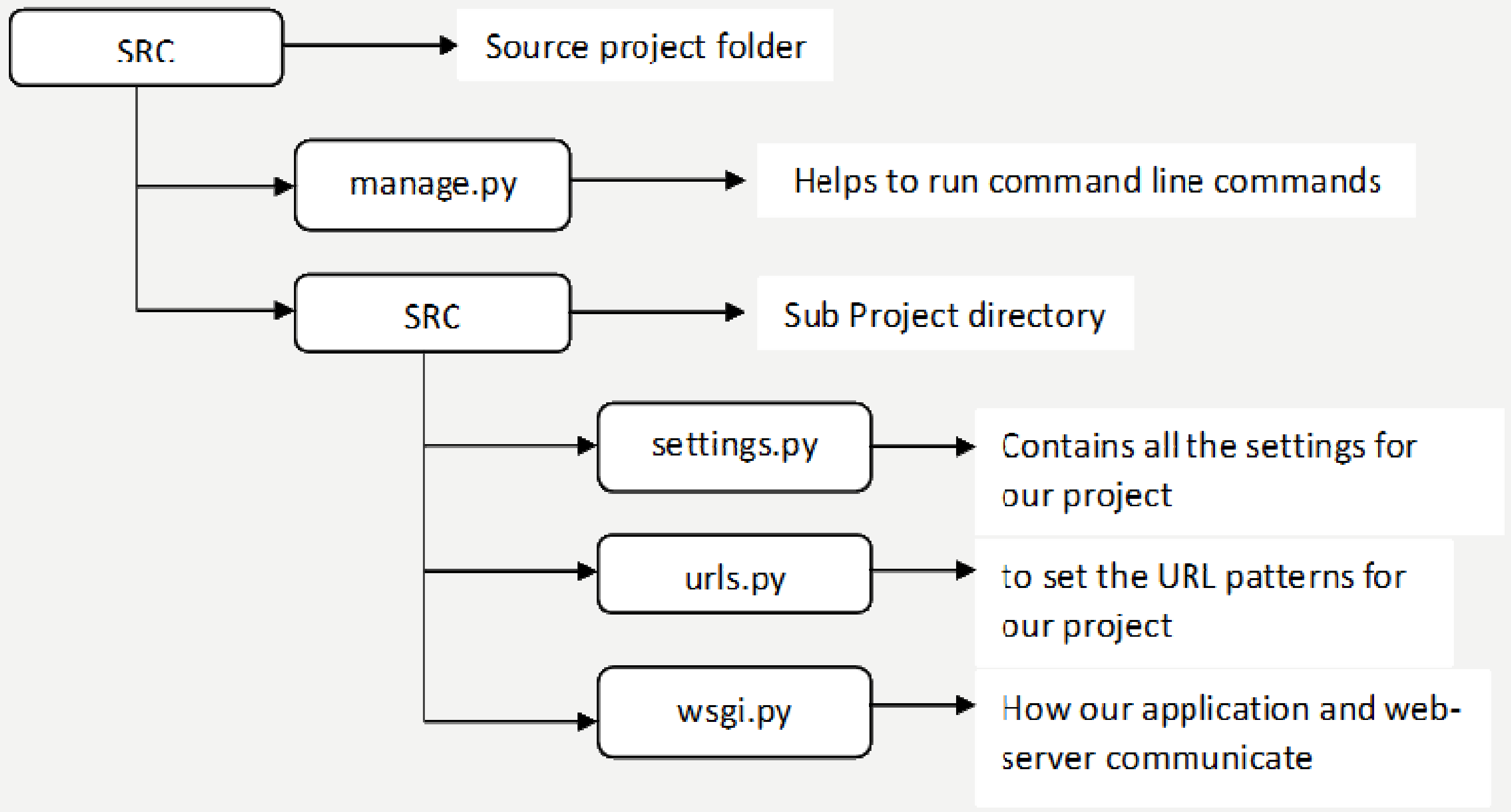
- In this presentation we have a brief introduction about how we can create a basic Django project.
- So, we are creating a Learning Management System in which our first Django app will be about how a Learner can register and login on portal and can able to access all functionality.
- Then, we will do Trainer Registration in which a user can register himself/herself as a trainer and a learner can also register himself/herself as a trainer.
- After that an Admin will check trainer details through admin dashboard and give authorization to user to login as a trainer.
- In last step we will see after authorization how a trainer can create a course.

Step 1: Create a project

```
django-admin startproject SRC
```

This will create an SRC folder in your current directory.

Project Structure


SRC
Source project folder
manage.py
Helps to run command line commands
SRC
Sub Project directory
settings.py
Contains all the settings for our project
urls.py
to set the URL patterns for our project
wsgi.py
How our application and web-server communicate

Step 2: Create a library app

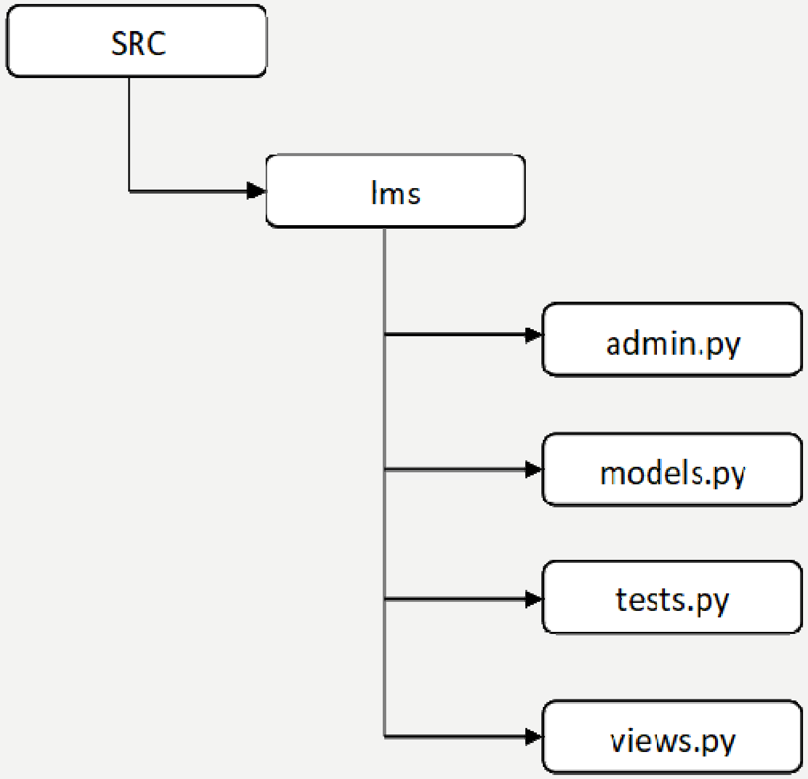
Now change a directory to SRC by following command:

```
cd SRC
```

Now we will create an app by using following command:

```
python manage.py startapp lms
```

App Structure



Step 3:

Now we will create a home page where we can display login, Sign Up and other buttons.

views.py:

```python
from django.shortcuts import render, redirect
from django.contrib import messages
from django.contrib.auth.models import User, auth

def home(request):
    return render(request, 'lms/home.html')
```

urls.py:

```python
from django.contrib import admin
from django.urls import path
from .views import *

app_name = 'lms'

urlpatterns = [
    path('', home, name="home"),
]
```

What is render():

Combines a given template with a given context dictionary and returns an HttpResponse object wit
that rendered text.

Django does not provide a shortcut function which returns a TemplateResponse because the constructor of TemplateResponse offers the same level of convenience as render().

Required arguments:

request:

The request object used to generate this response.

template_name:

The full name of a template to use or sequence of template names. If a sequence is given, the first template that exists will be used. See the template loading documentation for more information on how templates are found.

What is redirect():

Returns an HttpResponseRedirect to the appropriate URL for the arguments passed.

The arguments could be:

- A model: the model’s get_absolute_url() function will be called.
- A view name, possibly with arguments: reverse() will be used to reverse-resolve the name.
- An absolute or relative URL, which will be used as-is for the redirect location.

By default issues a temporary redirect; pass permanent=True to issue a permanent redirect.

What are messages in Django?

The Django web frameworks comes with a messaging system that allows us to store messages that we can check for on each page load. If there are some messages, we can display them to the user. For these messages, we could show them however we see fit. With materialize.

User authentication in Django:

Django comes with a user authentication system. It handles user accounts, groups, permissions and cookie-based user sessions.

Authentication support is bundled as a Django contrib module in django.contrib.auth By default, the required configuration is already included in the settings.py generatec by django-admin startproject, these consist of two items listed in your INSTALLED_APPS setting:

1. 'django.contrib.auth' contains the core of the authentication framework, and its default models.
2. 'django.contrib.contenttypes' is the Django content type system, which allows permissions to be associated with models you create.

and these items in your MIDDLEWARE setting:

1. SessionMiddleware manages sessions across requests.
2. AuthenticationMiddleware associates users with requests using sessions.

The Django admin site:

One of the most powerful parts of Django is the automatic admin interface. It reads metadata from your models to provide a quick, model-centric interface where trusted users can manage content on your site. The admin's recommended use is limited to an organization's internal management tool. It's not intended for building your entire front end around.

The admin has many hooks for customization, but beware of trying to use those hooks exclusively. If you need to provide a more process-centric interface that abstracts away the implementation details of database tables and fields, then it's probably time to write your own views.

What is path in Django?

path is a new function defined in django 2.0 .It returns an element for inclusion in urlpatterns in urls.py

Request and response objects:

Django uses request and response objects to pass state through the system.

When a page is requested, Django creates an HttpRequest object that contains metadata about the request. Then Djang loads the appropriate view, passing the HttpRequest as the first argument to the view function. Each view is responsible for returning an HttpResponse object.

Now we will create a template folder in our app then we will create one more folder name as lms inside template folder. Now we will create ‘home.html’ and ‘home_base.html’ file inside lms folder:

We have added bootstrap files in home_base.html.

Now we will learn how to integrate bootstrap in our template.

CSS

Copy-paste the stylesheet <link> into your <head> before all other stylesheets to load CSS.

```
<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstra
p.min.css">
```

JS

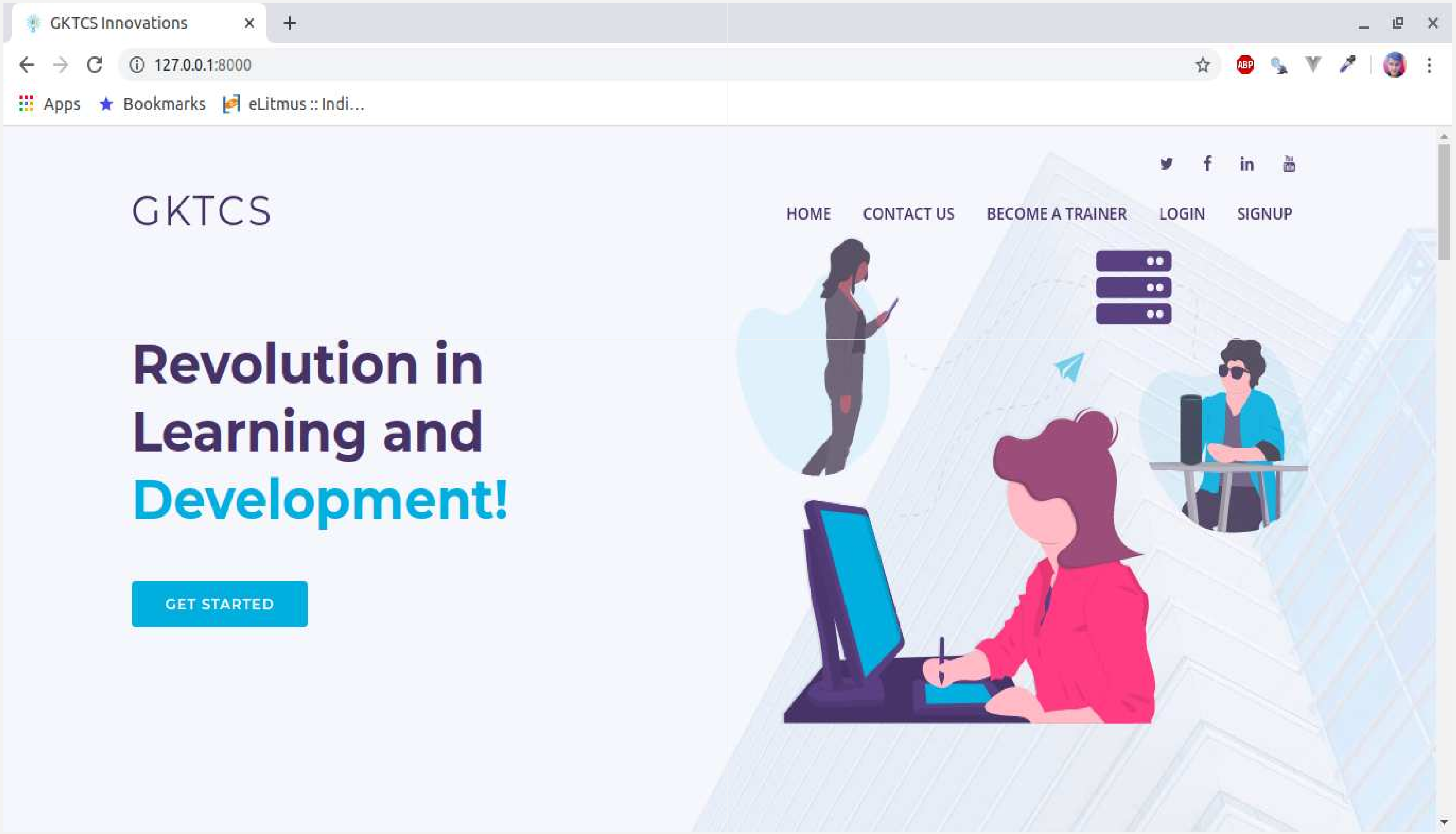
Place the following <script>s near the end of your pages, right before the closing </body> tag, to enable them. jQuery must come first, then Popper.js, and then our JavaScript plugins.

```
<script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js">
</script>
<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></sc
ript>
```

We have taken reference from following website for our template:

https://bootstrapmade.com/demo/Rapid/

Output: -


GKTCS Innovations
127.0.0.1:8000
Apps
Bookmarks
eLitmus :: Indi...
GKTCS
HOME
CONTACT US
BECOME A TRAINER
LOGIN
SIGNUP
Revolution in Learning and Development!
GET STARTED

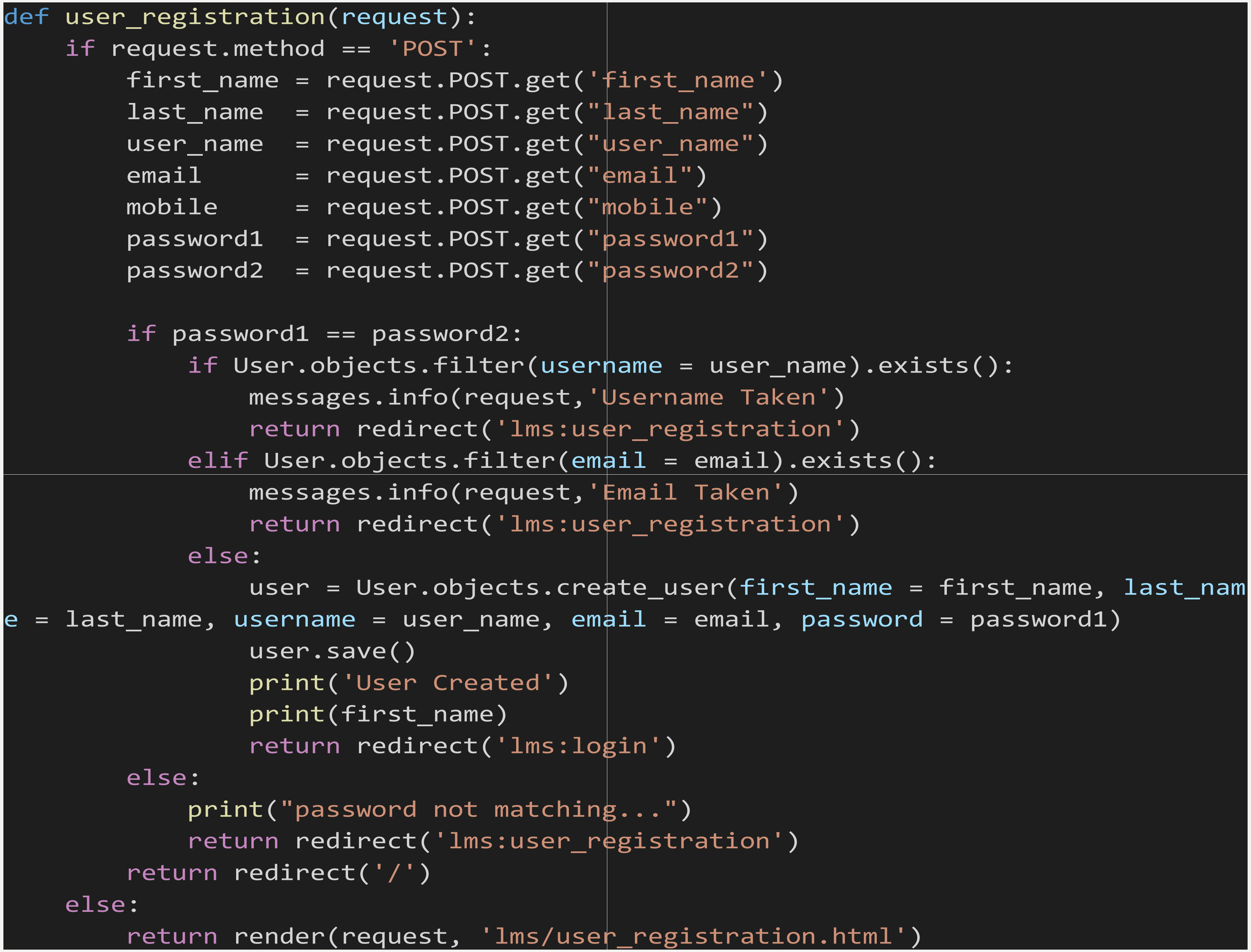
Step 4: -

Our next step will be user registration, user login and logout.

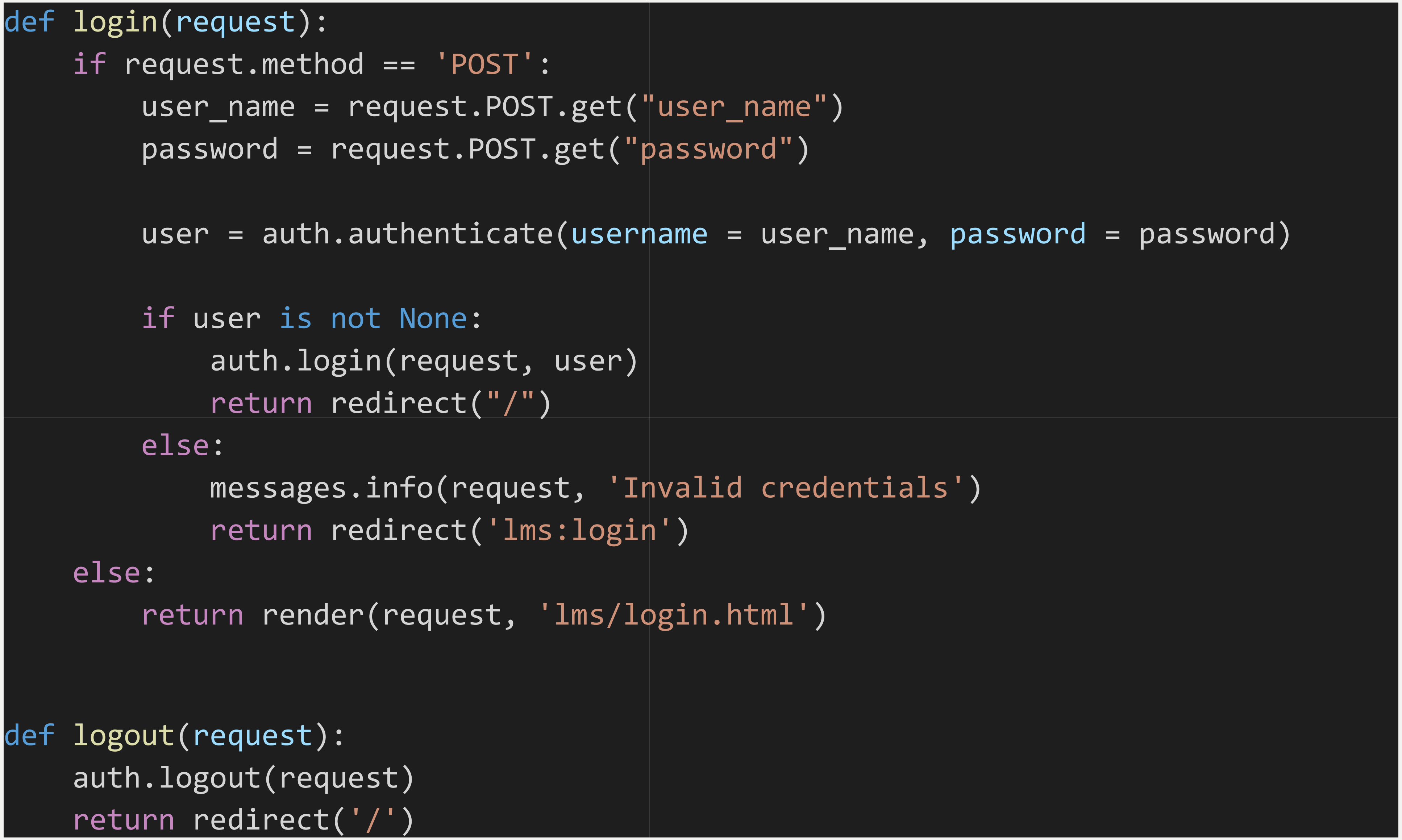
Django comes with a lot of built-in resources for the most common use cases of a Web application. The registration app i a very good example and a good thing about it is that the features can be used out-of-the-box.

Before we start, make sure you have django.contrib.auth in your INSTALLED_APPS and the authentication middleware properly configured in the MIDDLEWARE_CLASSES settings.

Both come already configured when you start a new Django project using the command startproject. So if you did not remove the initial configurations you should be all set up.



```python
def user_registration(request):
    if request.method == 'POST':
        first_name = request.POST.get('first_name')
        last_name  = request.POST.get("last_name")
        user_name  = request.POST.get("user_name")
        email      = request.POST.get("email")
        mobile     = request.POST.get("mobile")
        password1  = request.POST.get("password1")
        password2  = request.POST.get("password2")

        if password1 == password2:
            if User.objects.filter(username = user_name).exists():
                messages.info(request,'Username Taken')
                return redirect('lms:user_registration')
            elif User.objects.filter(email = email).exists():
                messages.info(request,'Email Taken')
                return redirect('lms:user_registration')
            else:
                user = User.objects.create_user(first_name = first_name, last_nam
e = last_name, username = user_name, email = email, password = password1)
                user.save()
                print('User Created')
                print(first_name)
                return redirect('lms:login')
        else:
            print("password not matching...")
            return redirect('lms:user_registration')
        return redirect('/')
    else:
        return render(request, 'lms/user_registration.html')
```



```python
def login(request):
    if request.method == 'POST':
        user_name = request.POST.get("user_name")
        password = request.POST.get("password")

        user = auth.authenticate(username = user_name, password = password)

        if user is not None:
            auth.login(request, user)
            return redirect("/")
        else:
            messages.info(request, 'Invalid credentials')
            return redirect('lms:login')
    else:
        return render(request, 'lms/login.html')


def logout(request):
    auth.logout(request)
    return redirect('/')
```

urls.py:

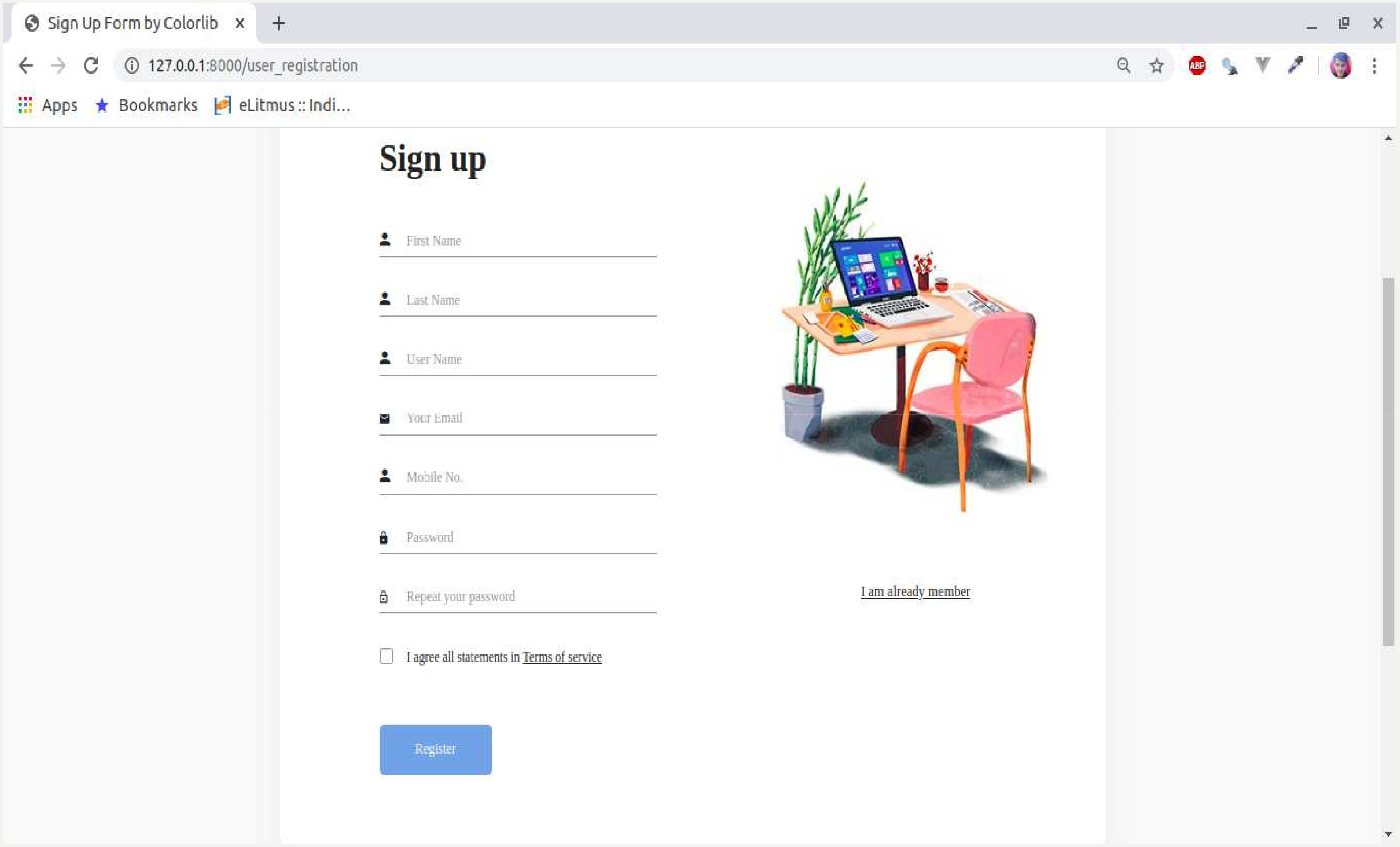
```python
from django.contrib import admin
from django.urls import path
from .views import *

app_name = 'lms'

urlpatterns = [
    path('', home, name="home"),
    path('user_registration', user_registration, name="user_registration"),
    path('login', login, name="login"),
    path('logout', logout, name="logout"),
]
```

Now we have to create two html files ‘user_registration.html’, ‘login.html’ and we need to modify ‘home.html’

user_registration.html:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>Registration</title>
</head>
<body>
<form action="user_registration" method="POST">
        {% csrf_token %}
        <input type="text" name="first_name" placeholder="First Name" required><br>
        <input type="text" name="last_name" placeholder="Last Name" required><br>
        <input type="text" name="user_name" placeholder="User Name" required><br>
        <input type="email" name="email" placeholder="email" required><br>
        <input type="text" name="mobile" placeholder="Mobile" required><br>
        <input type="password" name="password1" placeholder="Password"        required>
        <input type="password" name="password2" placeholder="Confirm Password"
        required><br>
        <input type="submit">
</form>
<div>
        {% for message in messages %}
                <h3>{{ message }}</h3>
        {% endfor %}
</div>
</body>
```

After applying CSS & Bootstrap our output will look like this:


Sign Up Form by Colorlib

127.0.0.1:8000/user_registration

Apps Bookmarks eLitmus :: Indi...

Sign up

First Name

Last Name

User Name

Your Email

Mobile No.

Password

Repeat your password

I agree all statements in Terms of service

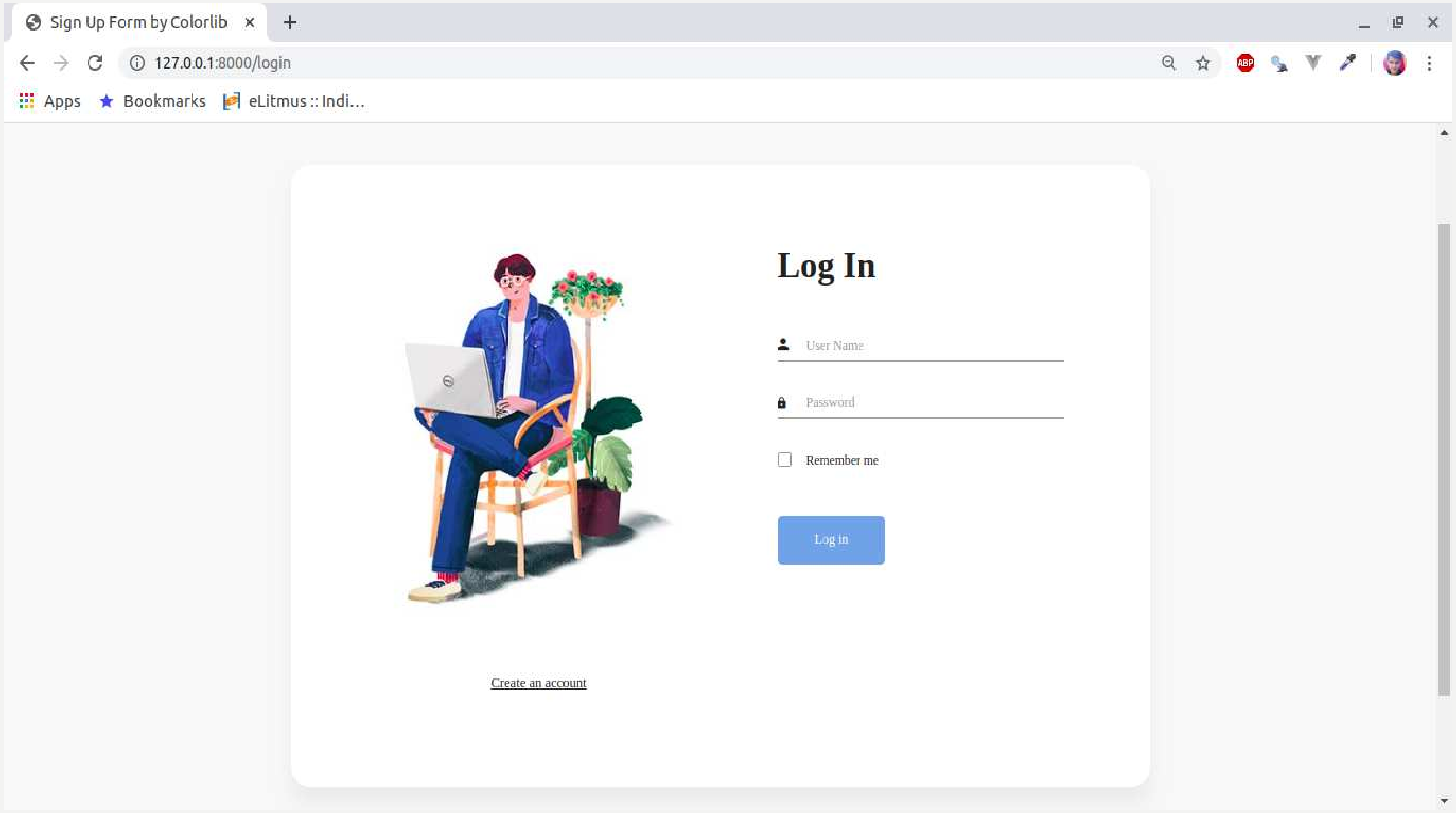
Register

I am already member

login.html: -

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>User login</title>
</head>
<body>
        <form action="login" method="POST">
                {% csrf_token %}
                <input type="text" name="user_name" placeholder="User Name"
        required><br>
                <input type="password" name="password" placeholder="Password"
        required><br>
                <input type="submit">
        </form>
        <div>
                {% for message in messages %}
                        <h3>{{ message }}</h3>
                {% endfor %}
        </div>
</body>
```

Output: -


Sign Up Form by Colorlib

127.0.0.1:8000/login

Apps Bookmarks eLitmus :: Indi...

Log In

User Name

Password

Remember me

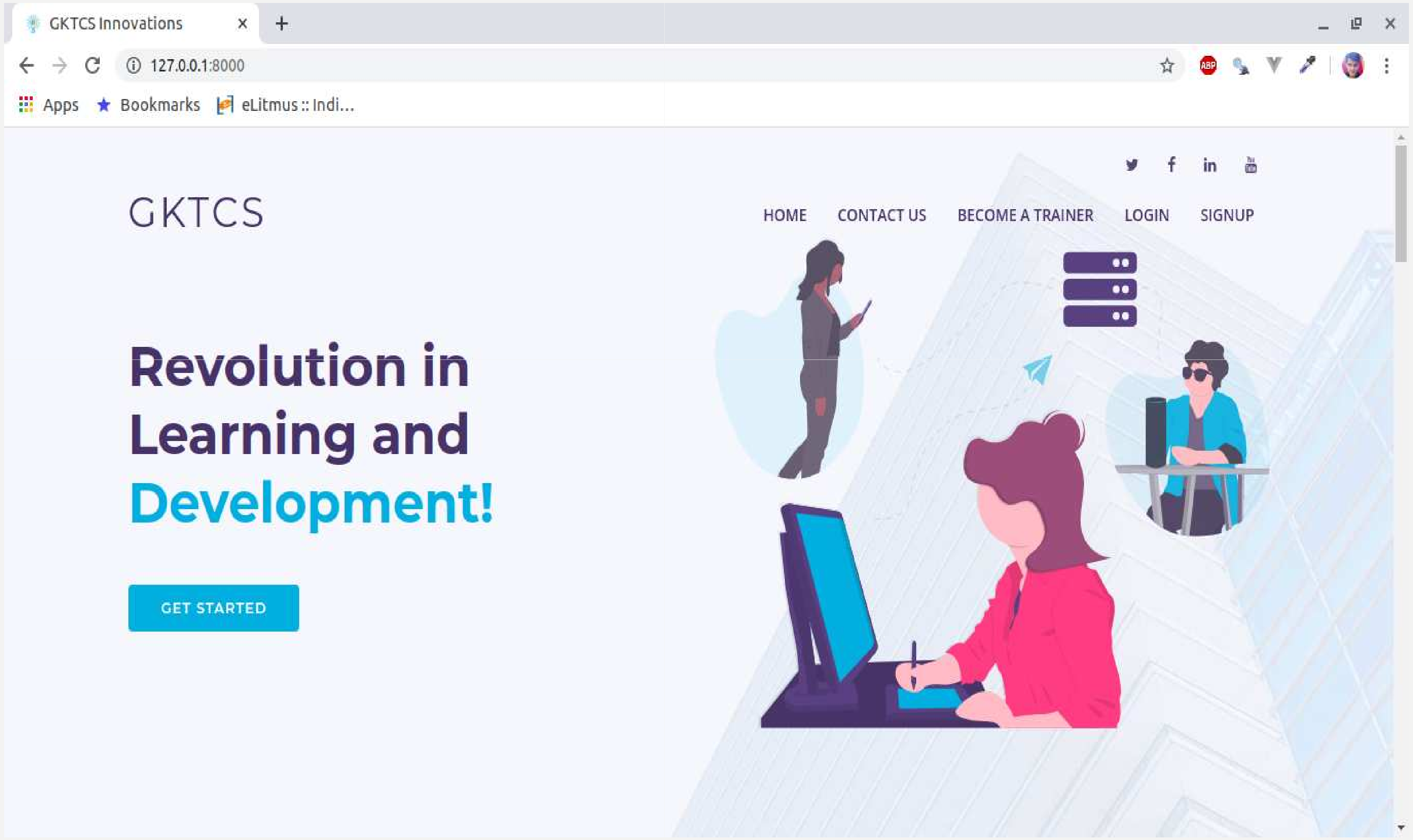
Log in

Create an account

home.html: -

```html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>GKTCS Innovations</title>
</head>
<body>
        {% if user.is_authenticated %}
        Hello, {{ user.first_name }}
        <a href="{% url 'lms:logout' %}">Logout</a> &nbsp;
        {% else %}
        <a href="{% url 'lms:login' %}">Login</a> &nbsp;
        <a href="{% url 'lms:user_registration' %}">SignUp</a> &nbsp;
        {% endif %}
</body>
</html>
```

Output:-



Revolution in Learning and Development!

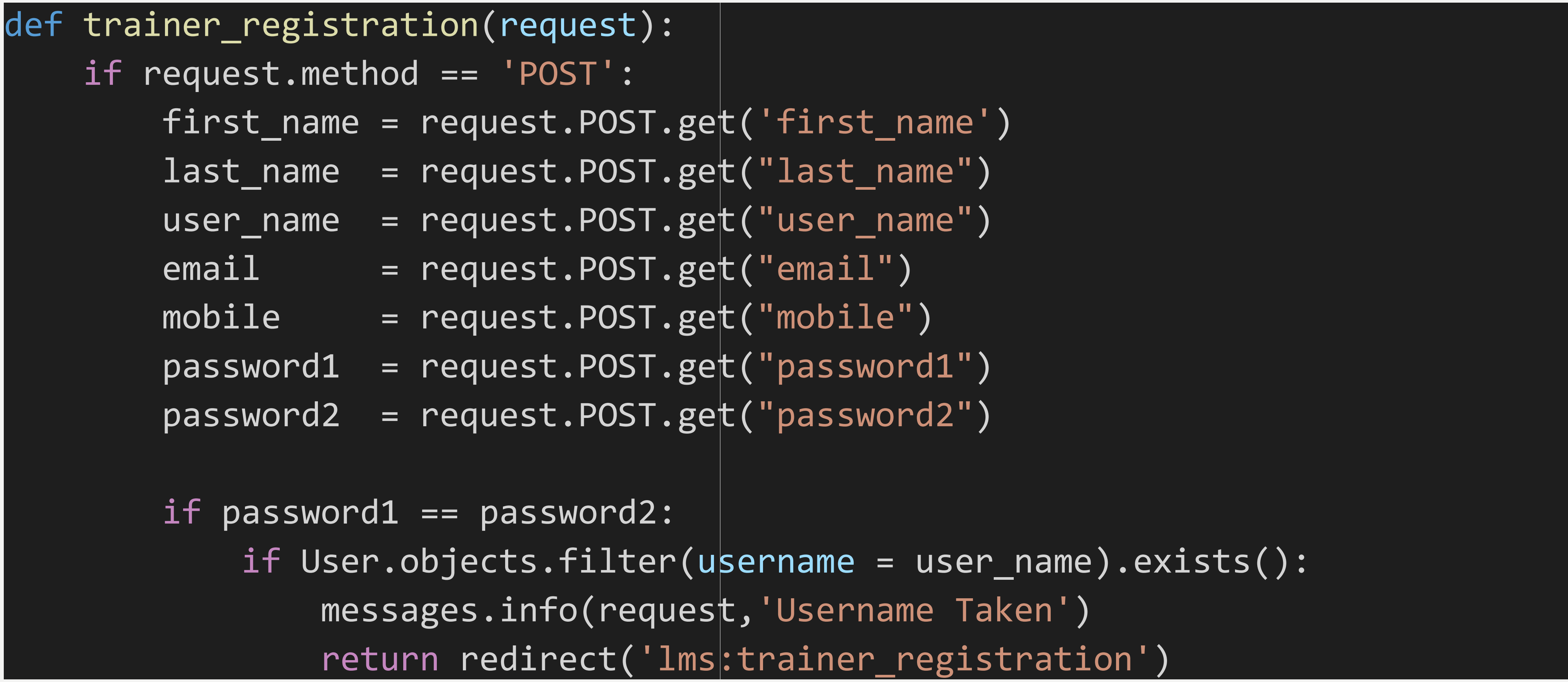
step 5:

In this step we will cover trainer registration and a learner who also wants to register as a trainer as well.

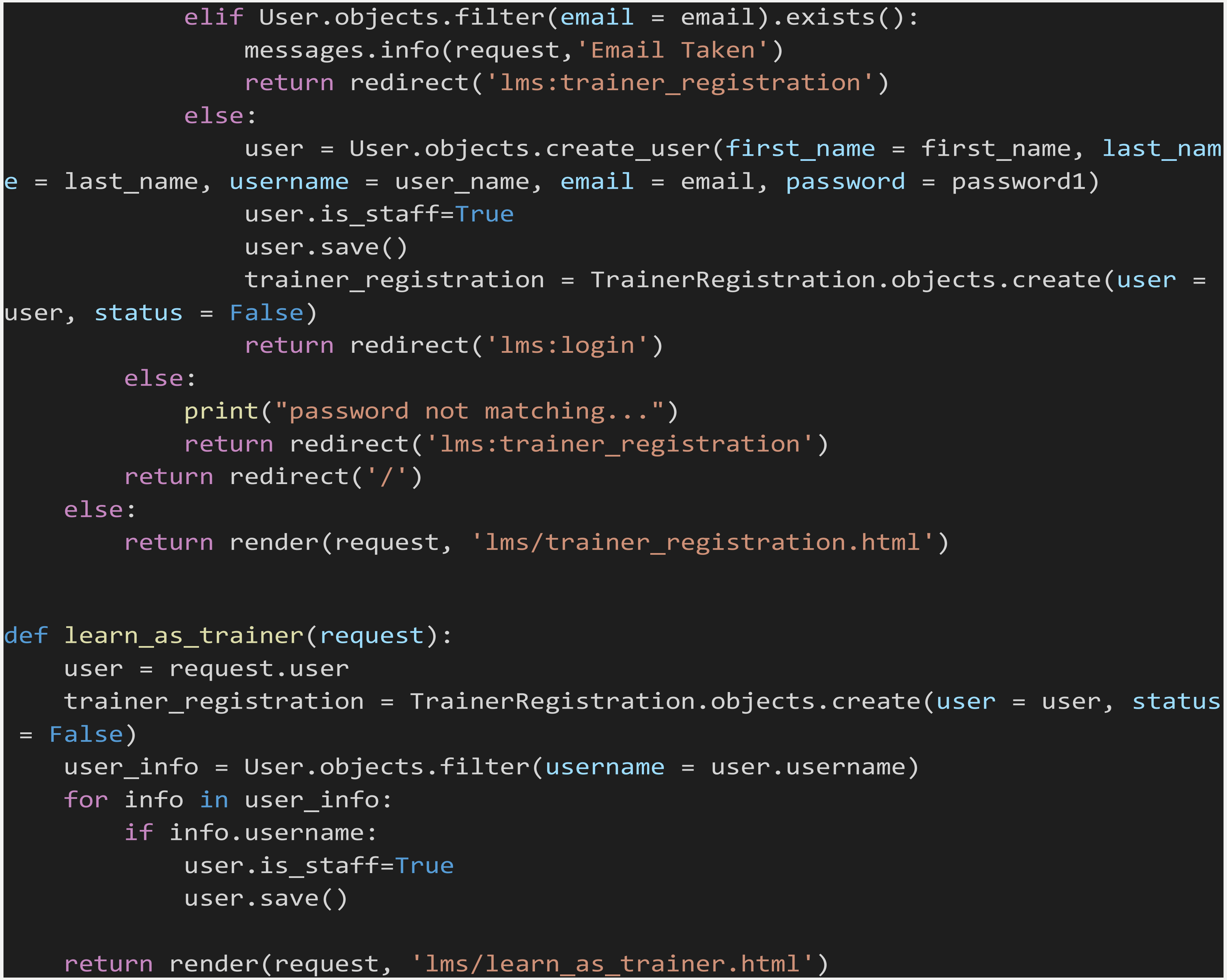
models.py: -

```python
from django.db import models
from django.db.models.signals import pre_save
from SRC.utils import *
from django.contrib.auth.models import User

class TrainerRegistration(models.Model):
    user          = models.ForeignKey(User, on_delete=models.CASCADE)
    status        = models.BooleanField()

    def __str__(self):
        return self.user.first_name
```

views.py: -



```python
def trainer_registration(request):
    if request.method == 'POST':
        first_name = request.POST.get('first_name')
        last_name  = request.POST.get("last_name")
        user_name  = request.POST.get("user_name")
        email      = request.POST.get("email")
        mobile     = request.POST.get("mobile")
        password1  = request.POST.get("password1")
        password2  = request.POST.get("password2")

        if password1 == password2:
            if User.objects.filter(username = user_name).exists():
                messages.info(request,'Username Taken')
                return redirect('lms:trainer_registration')
```



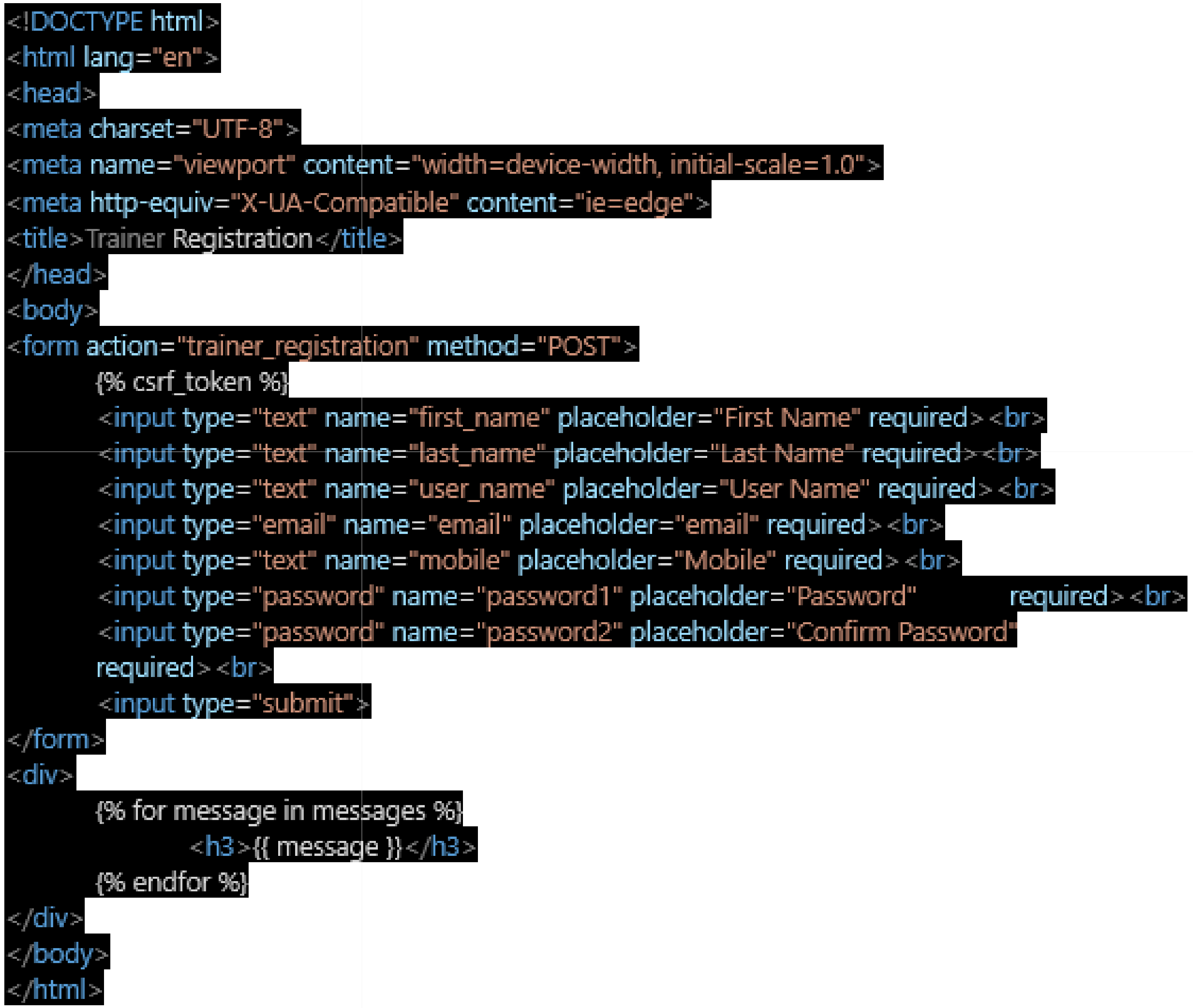
```
            elif User.objects.filter(email = email).exists():
                messages.info(request,'Email Taken')
                return redirect('lms:trainer_registration')
            else:
                user = User.objects.create_user(first_name = first_name, last_nam
e = last_name, username = user_name, email = email, password = password1)
                user.is_staff=True
                user.save()
                trainer_registration = TrainerRegistration.objects.create(user =
user, status = False)
                return redirect('lms:login')
        else:
            print("password not matching...")
            return redirect('lms:trainer_registration')
        return redirect('/')
    else:
        return render(request, 'lms/trainer_registration.html')


def learn_as_trainer(request):
    user = request.user
    trainer_registration = TrainerRegistration.objects.create(user = user, status
 = False)
    user_info = User.objects.filter(username = user.username)
    for info in user_info:
        if info.username:
            user.is_staff=True
            user.save()

    return render(request, 'lms/learn_as_trainer.html')
```

urls.py:

```
urlpatterns = [
    path('', home, name="home"),
    path('user_registration', user_registration, name="user_registration"),
    path('login', login, name="login"),
    path('logout', logout, name="logout"),
    path('trainer_registration', trainer_registration, name="trainer_registrati
"),
    path('learn_as_trainer', learn_as_trainer, name="learn_as_trainer"),
]
```

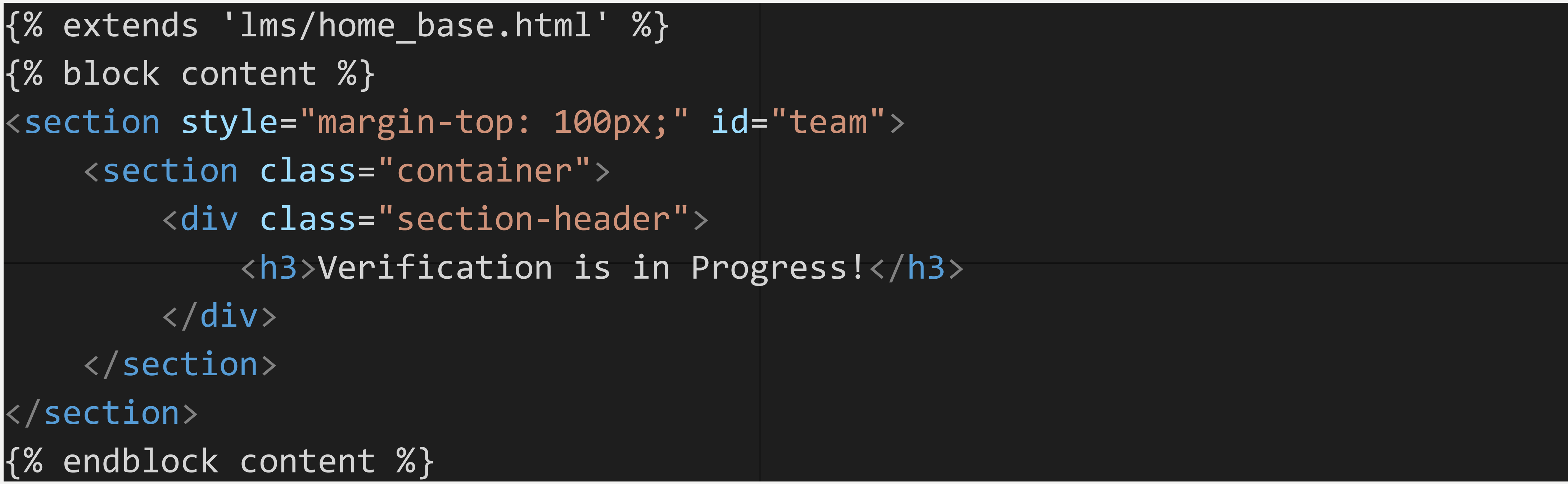
Now, we need to create two templates ‘trainer_registration.html’ and ‘learner_as_trainer.html’.

trainer_registraion.html:



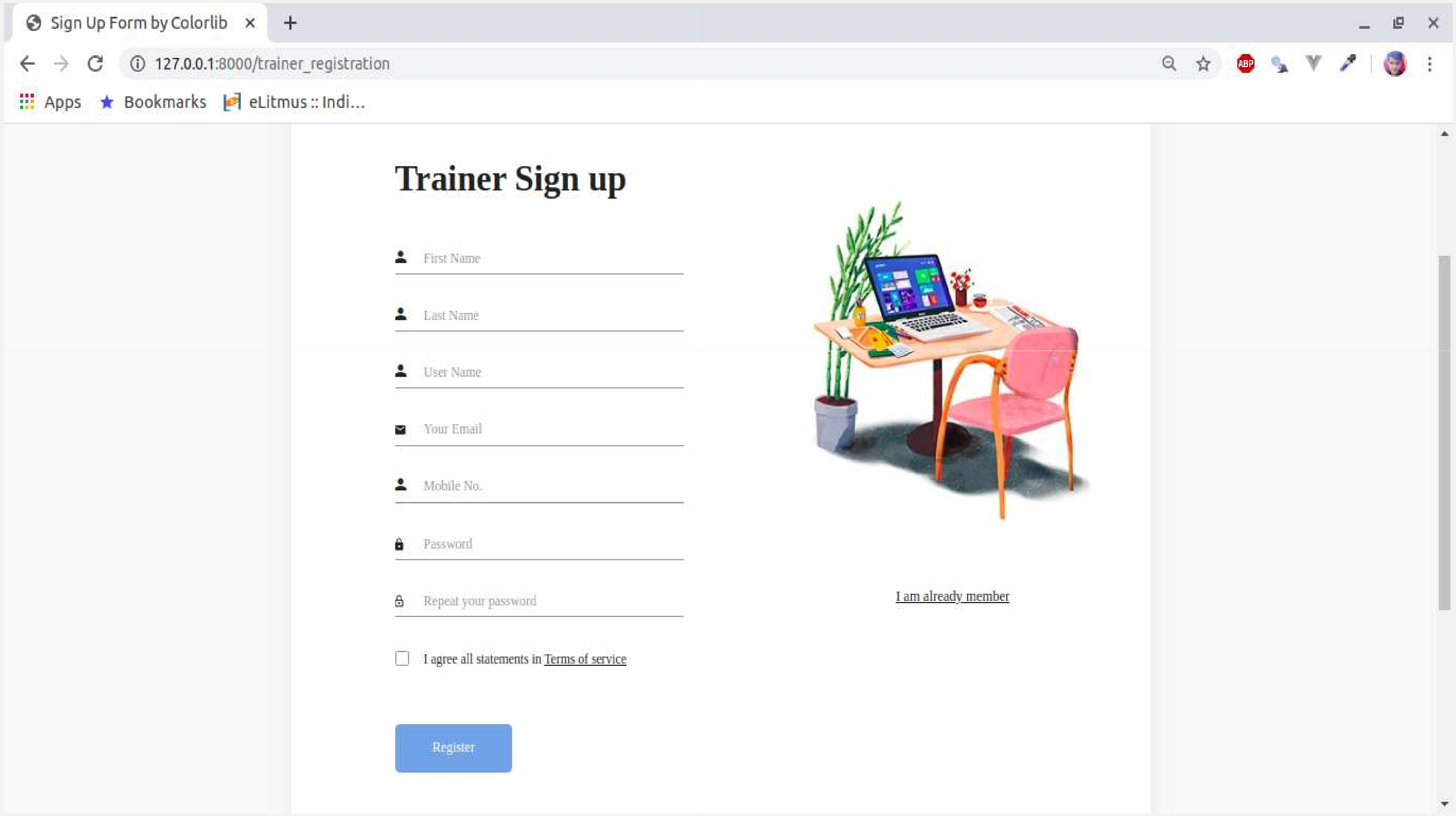
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>Trainer Registration</title>
</head>
<body>
<form action="trainer_registration" method="POST">
        {% csrf_token %}
        <input type="text" name="first_name" placeholder="First Name" required> <br>
        <input type="text" name="last_name" placeholder="Last Name" required> <br>
        <input type="text" name="user_name" placeholder="User Name" required> <br>
        <input type="email" name="email" placeholder="email" required> <br>
        <input type="text" name="mobile" placeholder="Mobile" required> <br>
        <input type="password" name="password1" placeholder="Password"         required> <br>
        <input type="password" name="password2" placeholder="Confirm Password"
        required> <br>
        <input type="submit">
</form>
<div>
        {% for message in messages %}
                <h3>{{ message }}</h3>
        {% endfor %}
</div>
</body>
</html>
```

learn_as_trainer.html:



```
{% extends 'lms/home_base.html' %}
{% block content %}
<section style="margin-top: 100px;" id="team">
    <section class="container">
        <div class="section-header">
            <h3>Verification is in Progress!</h3>
        </div>
    </section>
</section>
{% endblock content %}
```

Output:



Sign Up Form by Colorlib

127.0.0.1:8000/trainer_registration

Apps Bookmarks eLitmus :: Indi...

Trainer Sign up

First Name

Last Name

User Name

Your Email

Mobile No.

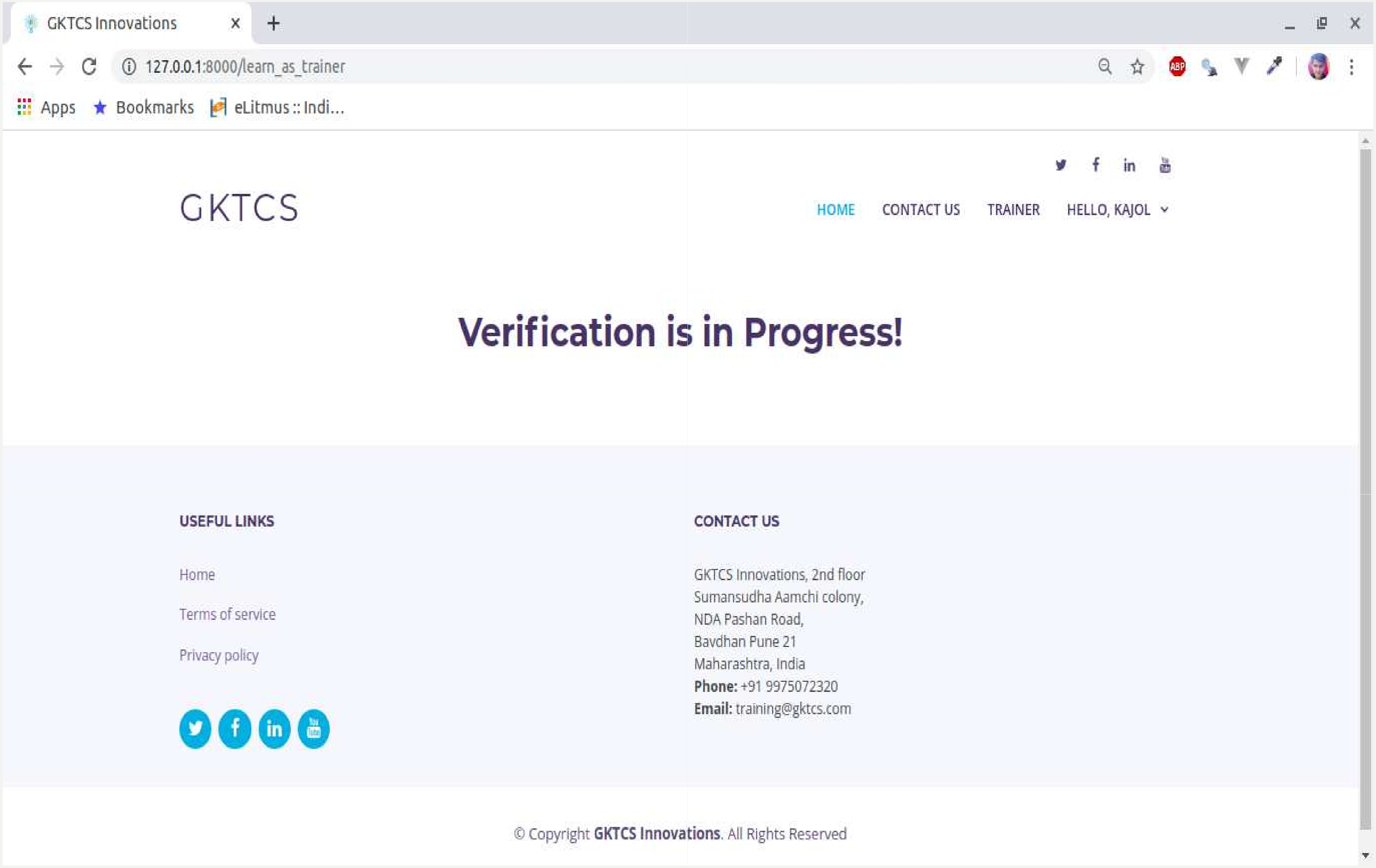
Password

Repeat your password

☐ I agree all statements in Terms of service

Register

I am already member



Verification is in Progress!

USEFUL LINKS

Home

Terms of service

Privacy policy

CONTACT US

GKTCS Innovations, 2nd floor
Sumansudha Aamchi colony,
NDA Pashan Road,
Bavdhan Pune 21
Maharashtra, India
Phone: +91 9975072320
Email: training@gktcs.com

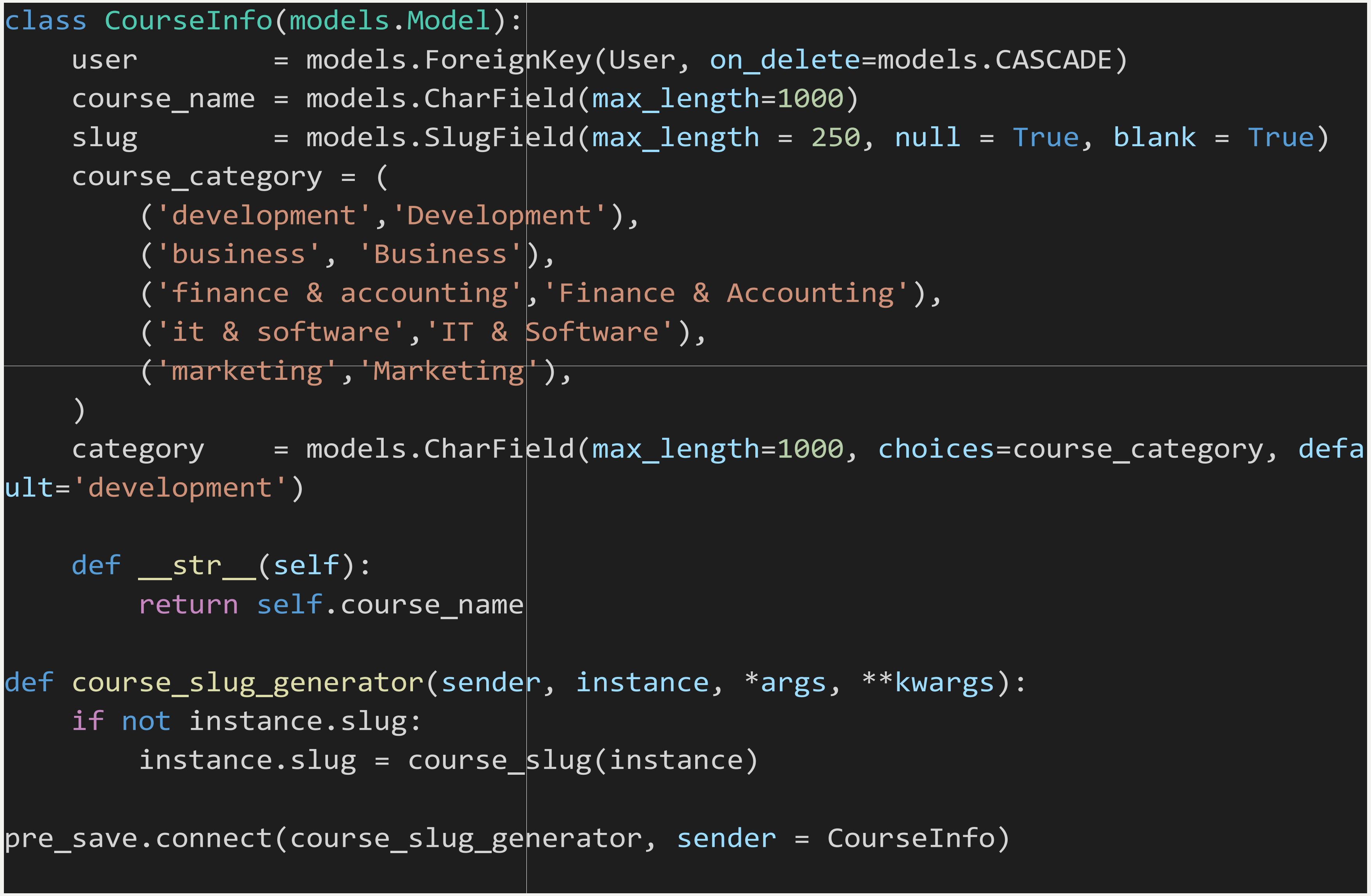
© Copyright **GKTCS Innovations**. All Rights Reserved

Now, Admin will check trainer details through admin dashboard and verify it and give authorization to trainer.

Step 6:

Course creation for authorized trainer.

models.py:



```
class CourseInfo(models.Model):
    user          = models.ForeignKey(User, on_delete=models.CASCADE)
    course_name = models.CharField(max_length=1000)
    slug          = models.SlugField(max_length = 250, null = True, blank = True)
    course_category = (
        ('development','Development'),
        ('business', 'Business'),
        ('finance & accounting','Finance & Accounting'),
        ('it & software','IT & Software'),
        ('marketing','Marketing'),
    )
    category     = models.CharField(max_length=1000, choices=course_category, defa
ult='development')

    def __str__(self):
        return self.course_name

def course_slug_generator(sender, instance, *args, **kwargs):
    if not instance.slug:
        instance.slug = course_slug(instance)

pre_save.connect(course_slug_generator, sender = CourseInfo)
```

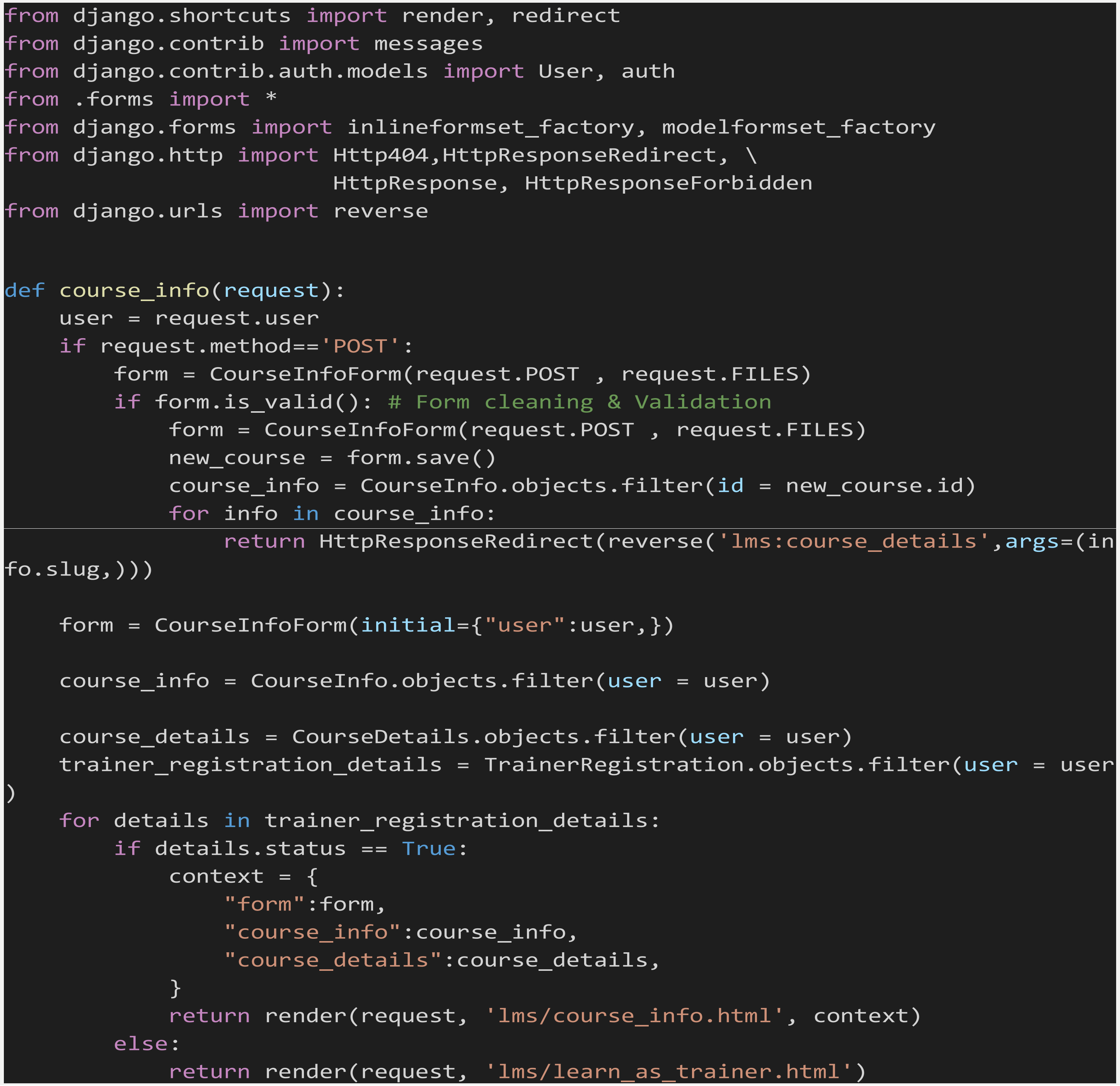
models.py:

```python
class CourseDetails(models.Model):
    user         = models.ForeignKey(User, on_delete=models.CASCADE)
    course_info  = models.ForeignKey(CourseInfo, on_delete = models.CASCADE)
    course_image = models.ImageField(blank=True, null=True)
    course_desc  = models.TextField()
```

forms.py:

```python
from lms.models import *
from django import forms

class CourseInfoForm(forms.ModelForm):
    class Meta:
        model  = CourseInfo
        fields = "__all__"
        # widgets = {'user': forms.HiddenInput(), 'slug': forms.HiddenInput()}


class CourseDetailsForm(forms.ModelForm):
    class Meta:
        model = CourseDetails
        fields = "__all__"
```

views.py:

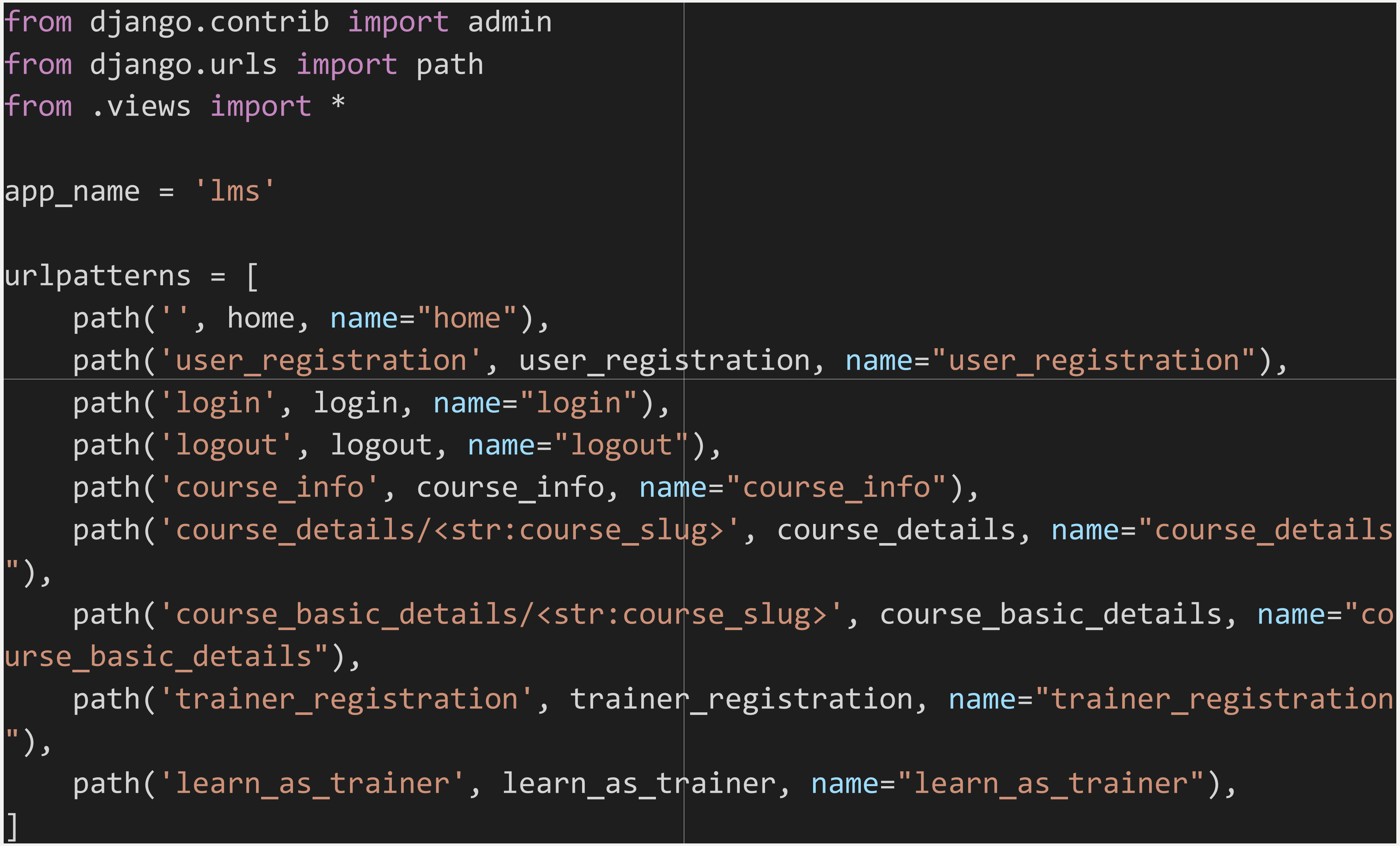


```python
from django.shortcuts import render, redirect
from django.contrib import messages
from django.contrib.auth.models import User, auth
from .forms import *
from django.forms import inlineformset_factory, modelformset_factory
from django.http import Http404,HttpResponseRedirect, \
                        HttpResponse, HttpResponseForbidden
from django.urls import reverse


def course_info(request):
    user = request.user
    if request.method=='POST':
        form = CourseInfoForm(request.POST , request.FILES)
        if form.is_valid(): # Form cleaning & Validation
            form = CourseInfoForm(request.POST , request.FILES)
            new_course = form.save()
            course_info = CourseInfo.objects.filter(id = new_course.id)
            for info in course_info:
                return HttpResponseRedirect(reverse('lms:course_details',args=(in
fo.slug,)))

    form = CourseInfoForm(initial={"user":user,})

    course_info = CourseInfo.objects.filter(user = user)

    course_details = CourseDetails.objects.filter(user = user)
    trainer_registration_details = TrainerRegistration.objects.filter(user = user
)
    for details in trainer_registration_details:
        if details.status == True:
            context = {
                "form":form,
                "course_info":course_info,
                "course_details":course_details,
            }
            return render(request, 'lms/course_info.html', context)
        else:
            return render(request, 'lms/learn_as_trainer.html')
```

views.py:

```python
def course_details(request, course_slug):
    course_info = CourseInfo.objects.get(slug = course_slug)

    context = {
        "course_slug":course_slug,
        "course_info":course_info,
    }
    return render(request, 'lms/course_details.html', context)


def course_basic_details(request, course_slug):
    user = request.user
    course_info = CourseInfo.objects.get(slug = course_slug)

    if request.method=='POST':
        form = CourseDetailsForm(request.POST , request.FILES)
        if form.is_valid(): # Form cleaning & Validation
            form = CourseDetailsForm(request.POST , request.FILES)
            form.save()

            # return HttpResponseRedirect('/')

    form = CourseDetailsForm(initial={'course_info':course_info,'user':user,})

    context = {
        "course_slug":course_slug,
        "course_info":course_info,
        "form":form,
    }
    return render(request, 'lms/course_basic_details.html', context)
```

urls.py:

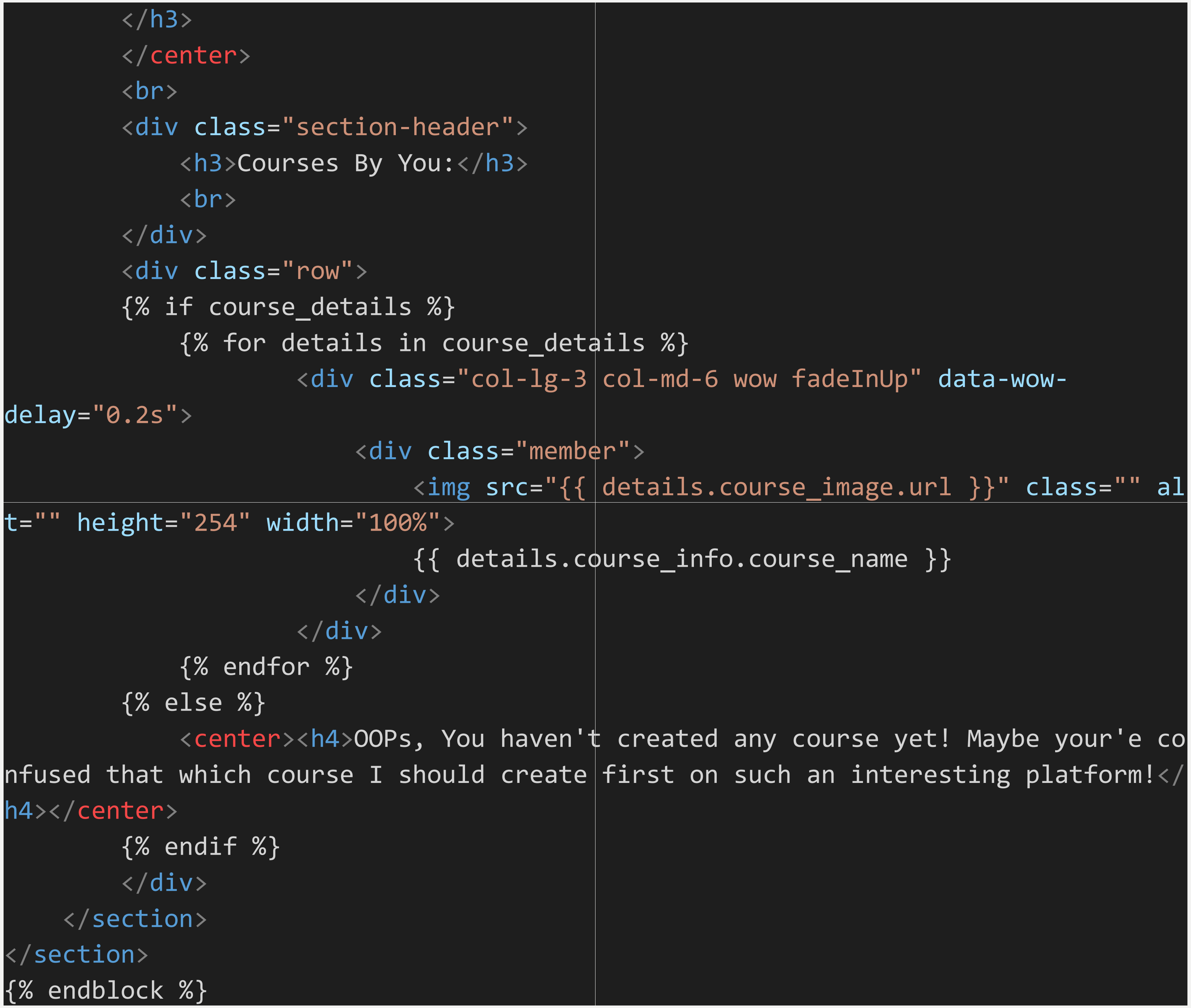


```
from django.contrib import admin
from django.urls import path
from .views import *

app_name = 'lms'

urlpatterns = [
    path('', home, name="home"),
    path('user_registration', user_registration, name="user_registration"),
    path('login', login, name="login"),
    path('logout', logout, name="logout"),
    path('course_info', course_info, name="course_info"),
    path('course_details/<str:course_slug>', course_details, name="course_details
"),
    path('course_basic_details/<str:course_slug>', course_basic_details, name="co
urse_basic_details"),
    path('trainer_registration', trainer_registration, name="trainer_registration
"),
    path('learn_as_trainer', learn_as_trainer, name="learn_as_trainer"),
]
```

Now, we need to create three html files ‘course_info.html’, ‘course_details.html’, ‘course_basic_details.html’.

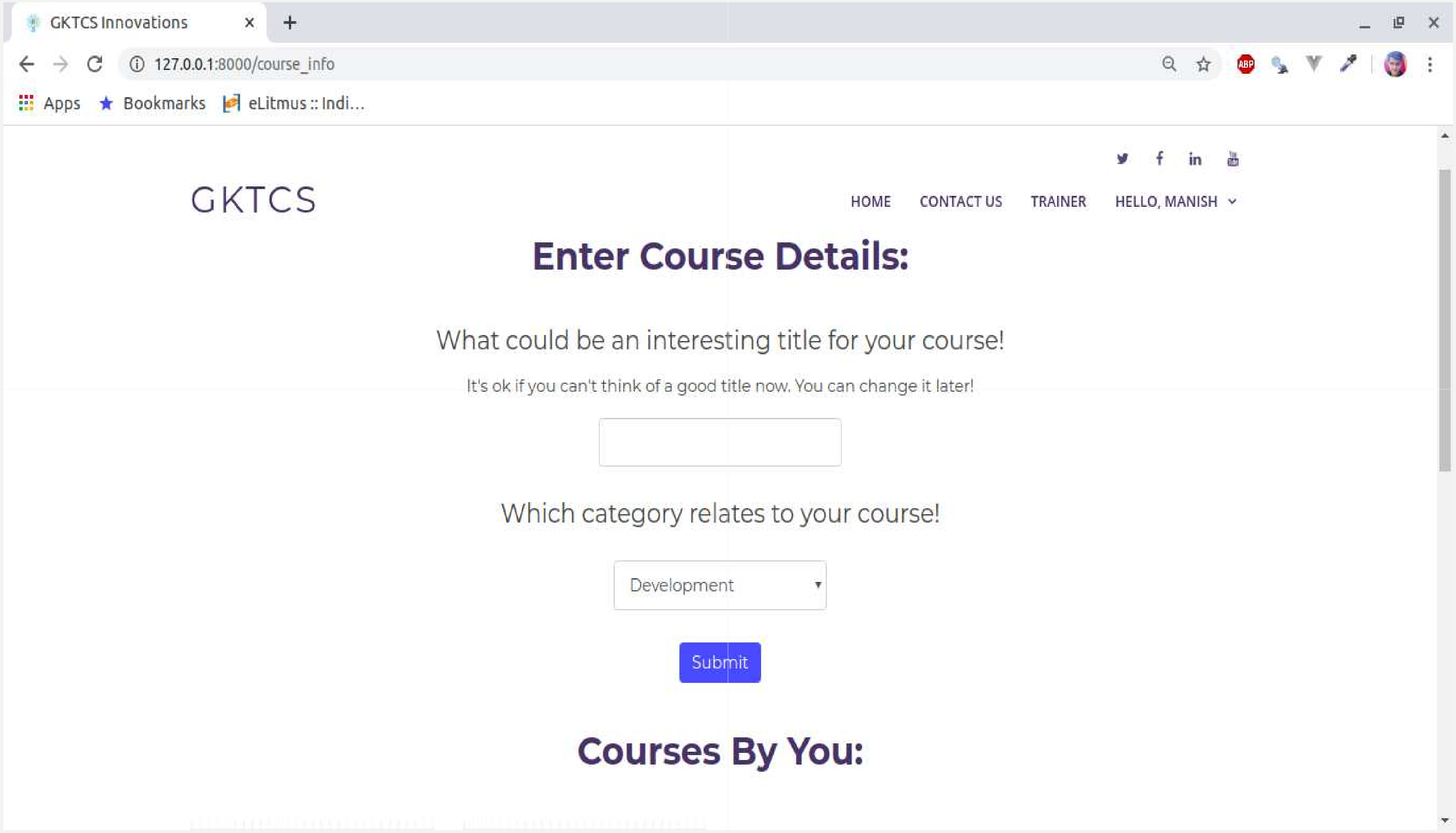
course_info.html:

```
{% extends 'lms/home_base.html' %}
{% block content %}
<section style="margin-top: 100px;" id="team">
    <section class="container">
        <div class="section-header">
            <h3>Enter Course Details:</h3>
            <br>
        </div>
        <center>
            <h3>
        <form method="POST" enctype="multipart/form-data">
            {% csrf_token %}
            <h4>What could be an interesting title for your course!</h4>
            <h6>It's ok if you can't think of a good title now. You can change it
 later!</h6>
            <p>{{ form.course_name }}</p>
            <h4>Which category relates to your course!</h4>
            <p><h5>{{ form.category }}</h5></p>
            {{ form.user }}
            <input type="submit" class="btn btn-primary">
        </form>
```



```
        </h3>
        </center>
        <br>
        <div class="section-header">
            <h3>Courses By You:</h3>
            <br>
        </div>
        <div class="row">
        {% if course_details %}
            {% for details in course_details %}
                    <div class="col-lg-3 col-md-6 wow fadeInUp" data-wow-
delay="0.2s">
                        <div class="member">
                            <img src="{{ details.course_image.url }}" class="" al
t="" height="254" width="100%">
                            {{ details.course_info.course_name }}
                        </div>
                    </div>
            {% endfor %}
        {% else %}
            <center><h4>OOPs, You haven't created any course yet! Maybe your'e co
nfused that which course I should create first on such an interesting platform!</
h4></center>
        {% endif %}
        </div>
    </section>
</section>
{% endblock %}
```

Output:



GKTCS Innovations

127.0.0.1:8000/course_info

Apps Bookmarks eLitmus :: Indi...

GKTCS

HOME CONTACT US TRAINER HELLO, MANISH

Enter Course Details:

What could be an interesting title for your course!

It's ok if you can't think of a good title now. You can change it later!

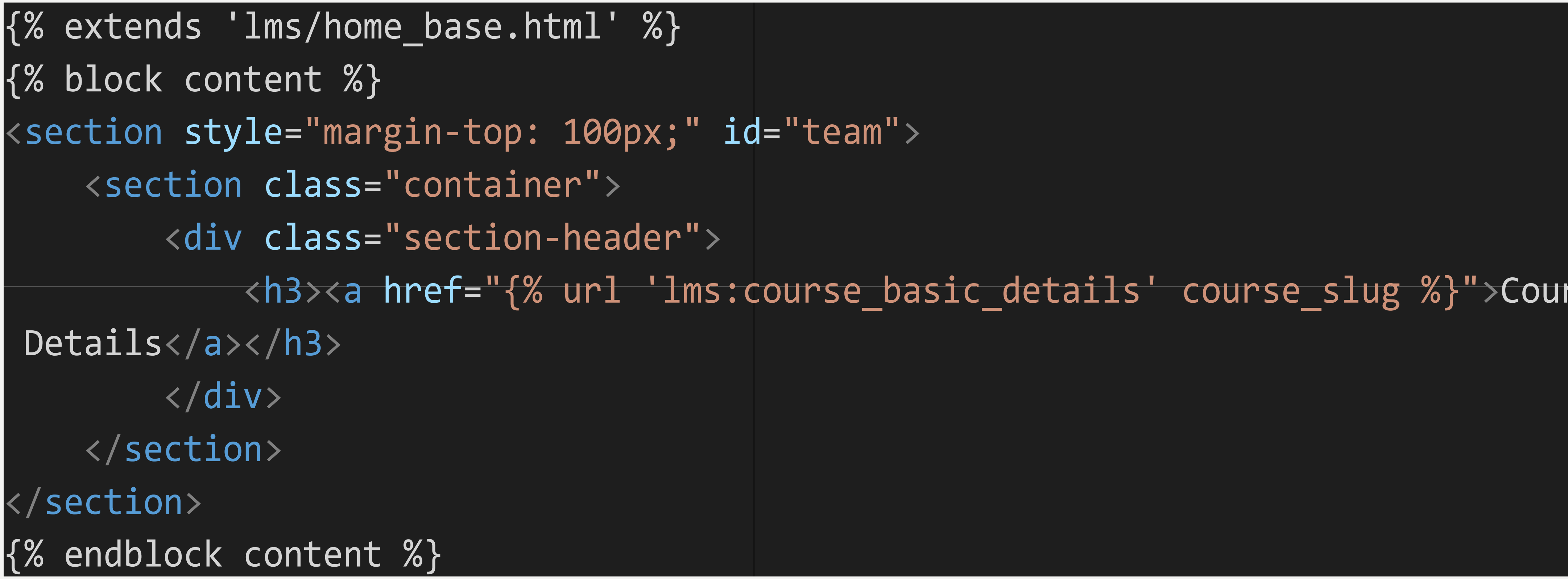
Which category relates to your course!

Development

Submit

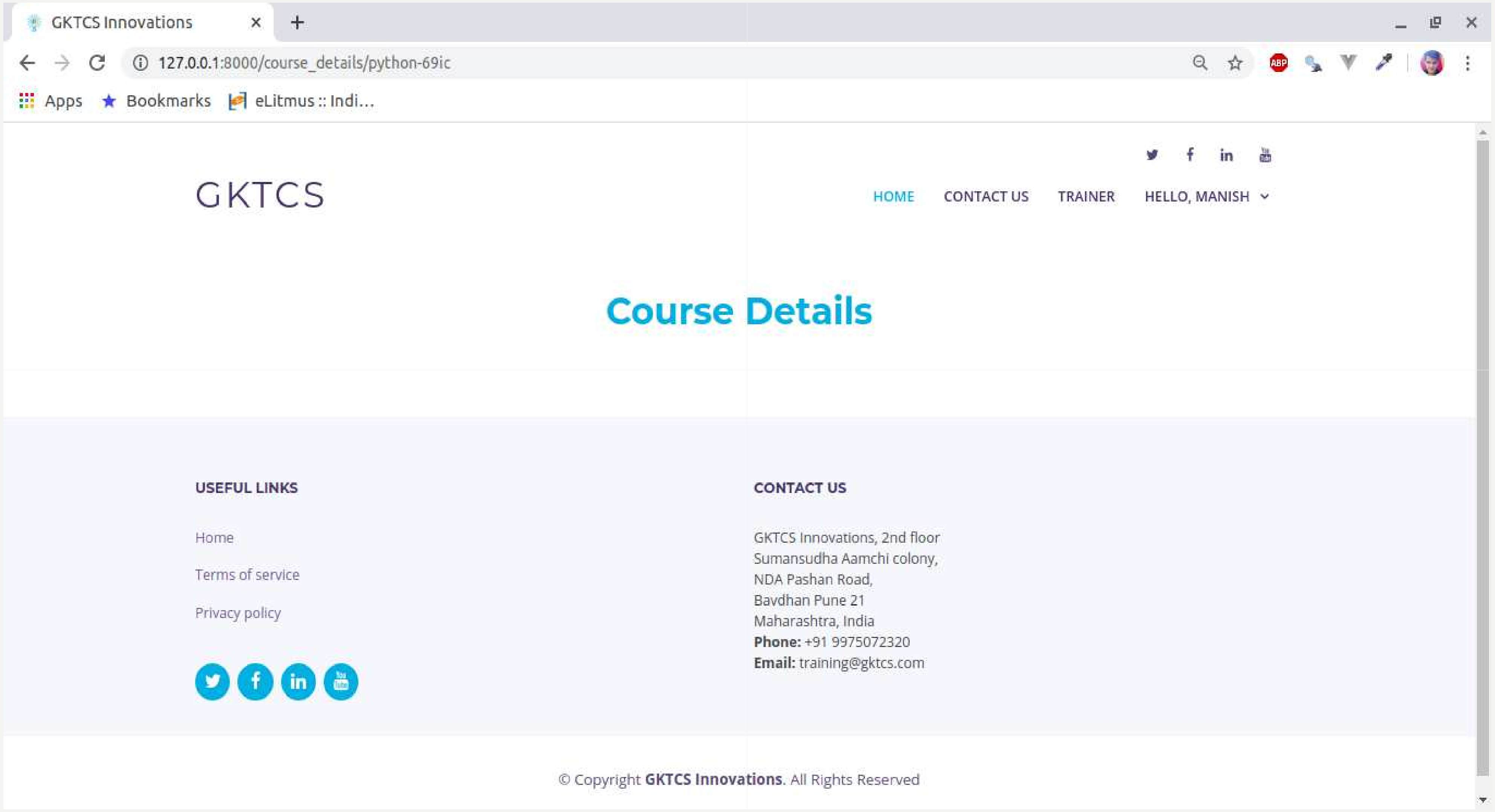
Courses By You:

course_details.html:



```
{% extends 'lms/home_base.html' %}
{% block content %}
<section style="margin-top: 100px;" id="team">
    <section class="container">
        <div class="section-header">
            <h3><a href="{% url 'lms:course_basic_details' course_slug %}">Cour
 Details</a></h3>
        </div>
    </section>
</section>
{% endblock content %}
```

Output:


GKTCS Innovations

127.0.0.1:8000/course_details/python-69ic

Apps Bookmarks eLitmus :: Indi...

GKTCS

HOME CONTACT US TRAINER HELLO, MANISH

Course Details

USEFUL LINKS

Home

Terms of service

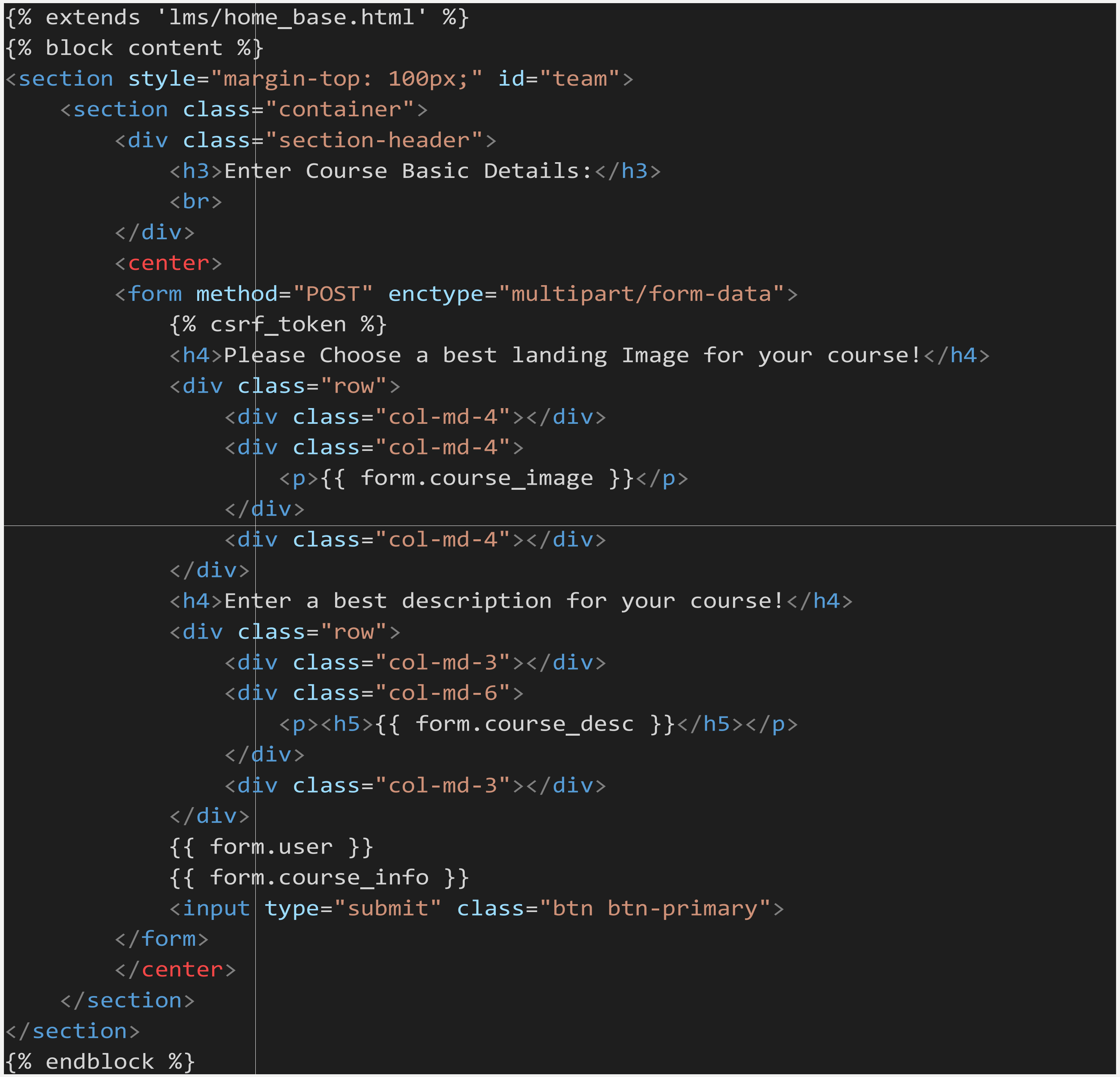
Privacy policy

CONTACT US

GKTCS Innovations, 2nd floor
Sumansudha Aamchi colony,
NDA Pashan Road,
Bavdhan Pune 21
Maharashtra, India
Phone: +91 9975072320
Email: training@gktcs.com

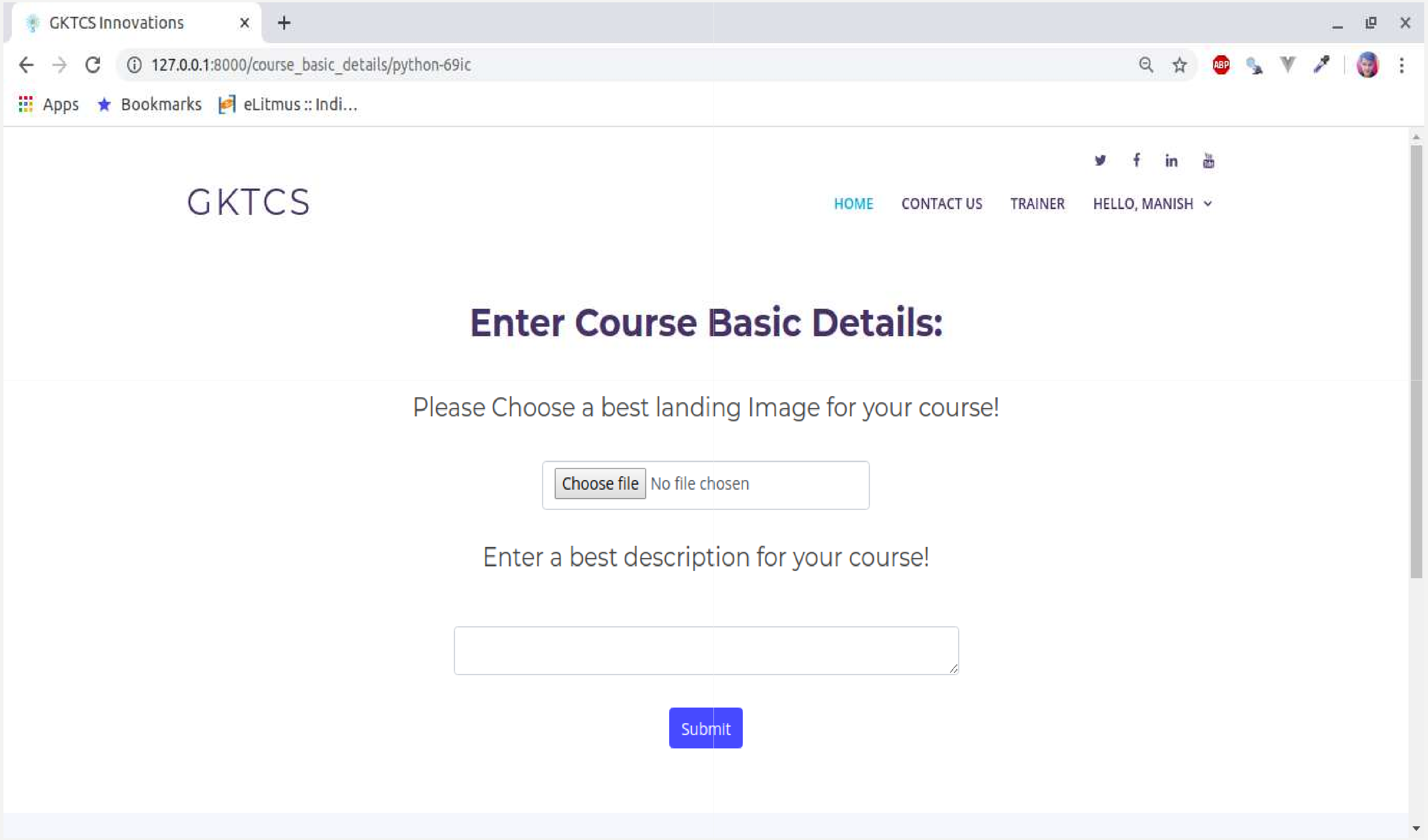
© Copyright **GKTCS Innovations**. All Rights Reserved

course_basic_details.html:



```
{% extends 'lms/home_base.html' %}
{% block content %}
<section style="margin-top: 100px;" id="team">
    <section class="container">
        <div class="section-header">
            <h3>Enter Course Basic Details:</h3>
            <br>
        </div>
        <center>
        <form method="POST" enctype="multipart/form-data">
            {% csrf_token %}
            <h4>Please Choose a best landing Image for your course!</h4>
            <div class="row">
                <div class="col-md-4"></div>
                <div class="col-md-4">
                    <p>{{ form.course_image }}</p>
                </div>
                <div class="col-md-4"></div>
            </div>
            <h4>Enter a best description for your course!</h4>
            <div class="row">
                <div class="col-md-3"></div>
                <div class="col-md-6">
                    <p><h5>{{ form.course_desc }}</h5></p>
                </div>
                <div class="col-md-3"></div>
            </div>
            {{ form.user }}
            {{ form.course_info }}
            <input type="submit" class="btn btn-primary">
        </form>
        </center>
    </section>
</section>
{% endblock %}
```

Output:



GKTCS Innovations

127.0.0.1:8000/course_basic_details/python-69ic

Apps · Bookmarks · eLitmus :: Indi...

GKTCS

HOME · CONTACT US · TRAINER · HELLO, MANISH

Enter Course Basic Details:

Please Choose a best landing Image for your course!

Choose file · No file chosen

Enter a best description for your course!

Submit

THANK YOU