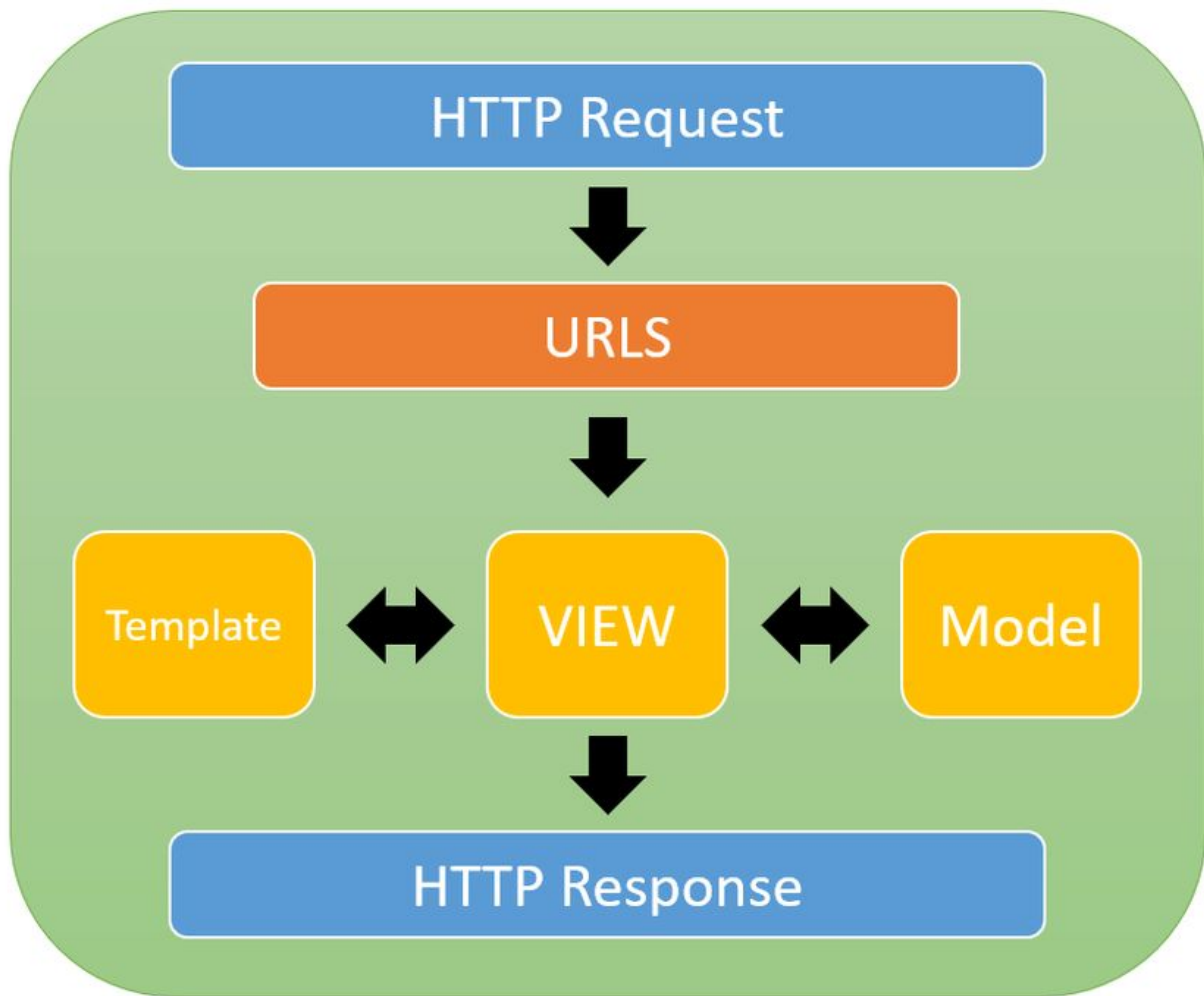
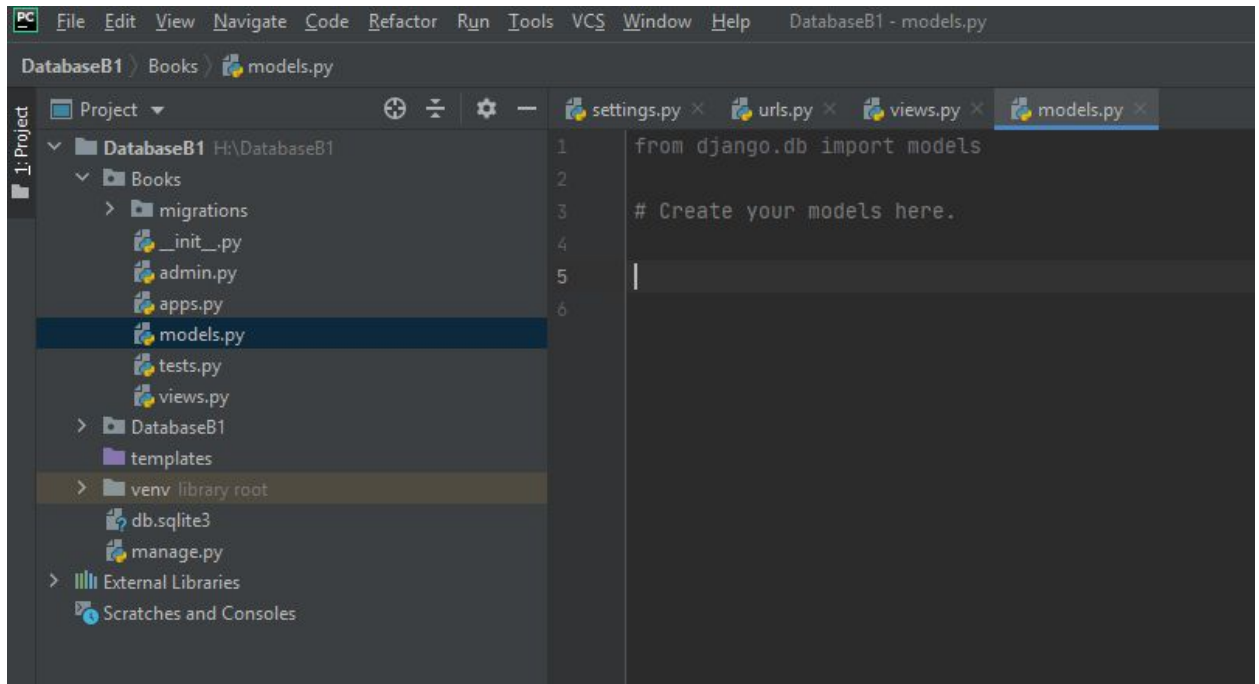


Django Structure



Database set up

1. Create a new Django Project
2. Create a new application
3. Go to models.py to create database tables



4.

Simple Book table

ID (PK)	Book Title (String)	Author Name (String)	Price (Integer)
1			

2			
---	--	--	--

```

1 from django.db import models
2
3 # Create your models here.
4
5
6 class Book(models.Model):
7     title = models.CharField(max_length=200, default="Book Title", #string
8     author = models.CharField(max_length=200, default="Author")
9     price = models.IntegerField(blank=True)]
10
11

```

5. Install the new app

```

27
28
29 # Application definition
30
31
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'Books.apps.BooksConfig',
41 ]
42

```

6. python manage.py makemigrations

```

(venv) H:\DatabaseB1>python manage.py makemigrations
Migrations for 'Books':
  Books\migrations\0001_initial.py
    - Create model Book

```

7. python manage.py migrate

```

(venv) H:\DatabaseB1>python manage.py migrate
Operations to perform:
  Apply all migrations: Books, admin, auth, contenttypes, sessions
Running migrations:
  Applying Books.0001_initial... OK
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying auth.0010_alter_group_name_max_length... OK
  Applying auth.0011_update_proxy_permissions... OK
  Applying auth.0012_alter_user_first_name_max_length... OK
  Applying sessions.0001_initial... OK

```

8. Python manage.py createsuperuser

```

(venv) H:\DatabaseB1>python manage.py createsuperuser
Username (leave blank to use 'asd'): tsr
Email address:
Password:
Password (again):
The password is too similar to the username.
This password is too short. It must contain at least 8 characters.
Bypass password validation and create user anyway? [y/N]: y
Superuser created successfully.

```

Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add

Change

Users

+ Add

Change

Recent actions

My actions

9. Register database table (model) in admin.py

```
DatabaseB1 > Books > admin.py
Project
  DatabaseB1
    Books
      migrations
      __init__.py
      admin.py
      apps.py
      models.py
      tests.py
      views.py
    DatabaseB1

1 from django.contrib import admin
2 from .models import Book
3 # Register your models here.
4
5 admin.site.register(Book)
6
7
```

10. admin:

Django administration

Site administration

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add

Change

Users

+ Add

Change

BOOKS

Books

+ Add

Change

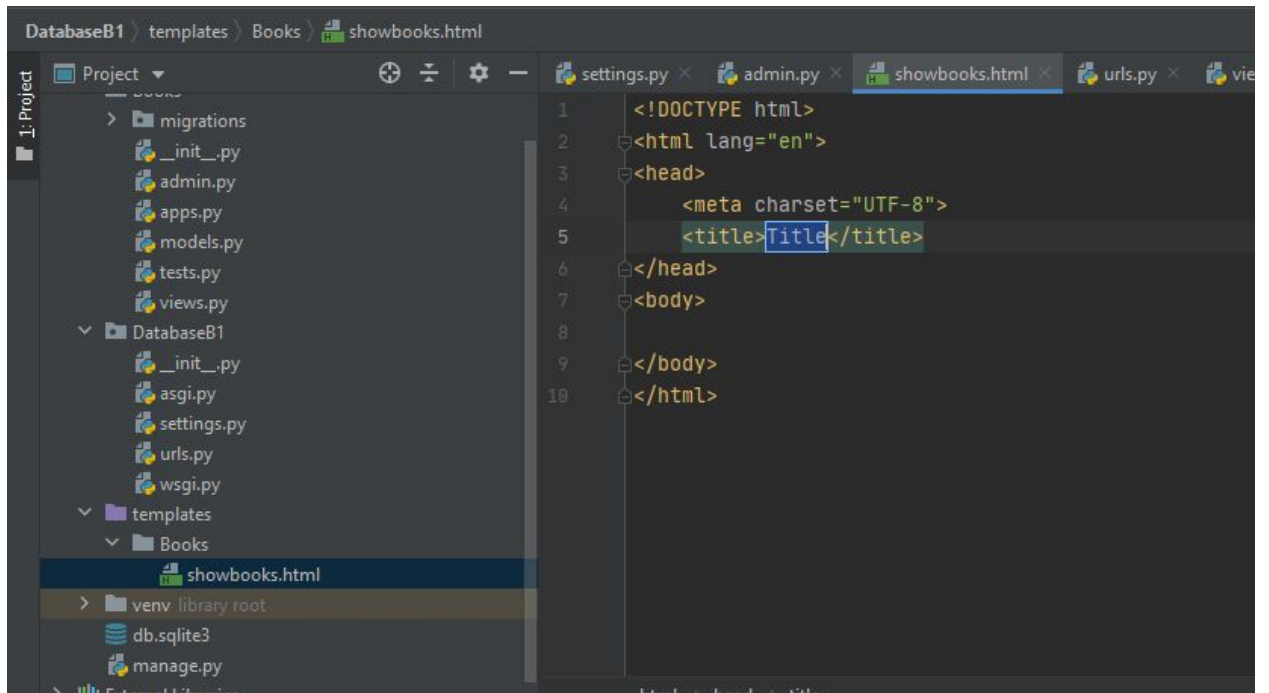
Recent actions

My actions

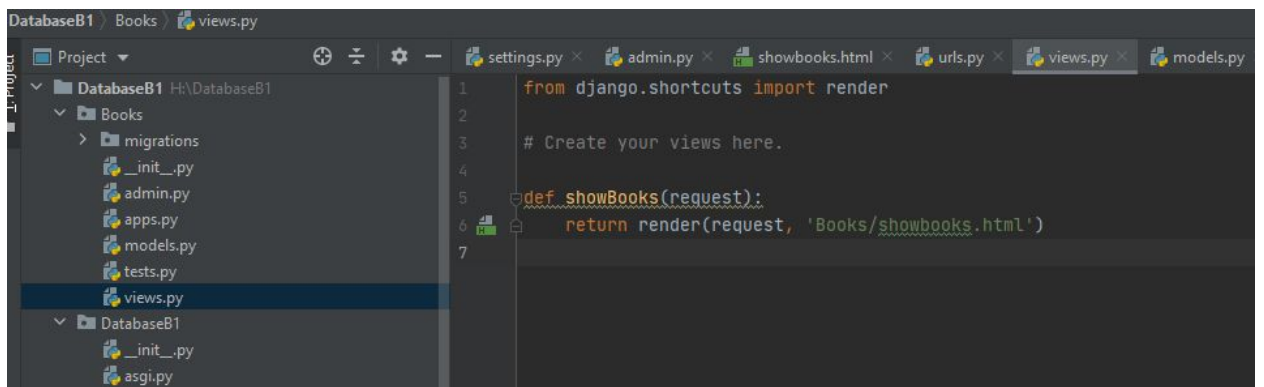
+ Book object (2)
Book

+ Book object (1)
Book

11. Create an empty HTML to show books from database



12. Connect this html page with views



13. Connect Books->views with main->urls


```
DatabaseB1 > DatabaseB1 > urls.py

Project
├── DatabaseB1
│   ├── Books
│   │   ├── migrations
│   │   ├── _init_.py
│   │   ├── admin.py
│   │   ├── apps.py
│   │   ├── models.py
│   │   ├── tests.py
│   │   └── views.py
│   ├── DatabaseB1
│   │   ├── _init_.py
│   │   ├── asgi.py
│   │   ├── settings.py
│   │   ├── urls.py
│   │   └── wsgi.py
│   └── templates
│       └── Books
│           └── showbooks.html

9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

Class-based views
1. Add an import: from other_app.views import Home
2. Add a URL to urlpatterns: path('', Home.as_view(), na
Including another URLconf
1. Import the include() function: from django.urls import
2. Add a URL to urlpatterns: path('blog/', include('blog

from django.contrib import admin
from django.urls import path
from Books import views as bookviews

urlpatterns = [
    path('admin/', admin.site.urls),
    path('books/', bookviews.showBooks)]
```

14. Views.py (controller). Get information from database

```
DatabaseB1 > Books > views.py

Project
├── DatabaseB1
│   ├── Books
│   │   ├── migrations
│   │   ├── _init_.py
│   │   ├── admin.py
│   │   ├── apps.py
│   │   ├── models.py
│   │   ├── tests.py
│   │   └── views.py
│   ├── DatabaseB1
│   │   ├── _init_.py
│   │   ├── asgi.py
│   │   ├── settings.py
│   │   ├── urls.py
│   │   └── wsgi.py
│   └── templates
│       └── Books
│           └── showbooks.html

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

from django.shortcuts import render
from .models import Book

# Create your views here.

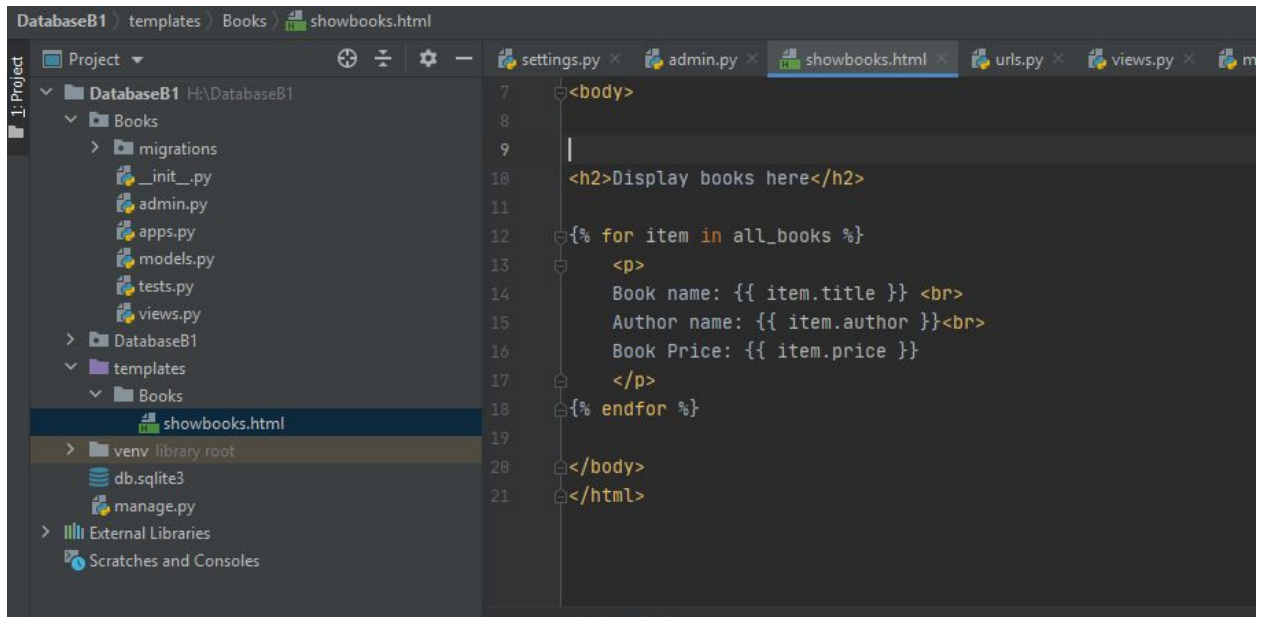
def showBooks(request):
    # models - views - html

    # models - views
    books = Book.objects.all() # THIS IS A LIST OF OBJECTS

    # views - html
    context = {
        'all_books': books
    }

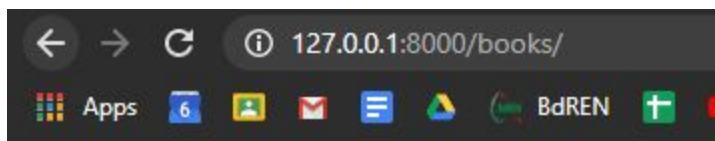
    return render(request, 'Books/showbooks.html', context)
```

15. Showbooks.html (view)



```
7 <body>
8
9 |
10 <h2>Display books here</h2>
11
12 {% for item in all_books %}
13 <p>
14     Book name: {{ item.title }} <br>
15     Author name: {{ item.author }}<br>
16     Book Price: {{ item.price }}
17 </p>
18 {% endfor %}
19
20 </body>
21 </html>
```

16. Display books



Display books here

Book name: The Alchemist
Author name: Paulo Coelho
Book Price: 500

Book name: The Library of Babel
Author name: Jorge Luis Borges
Book Price: 2000

Book name: Short Stories
Author name: kafka franz
Book Price: 1000

Add information from HTML to DB

I want to add information like admin

Add student

Name:

Email:

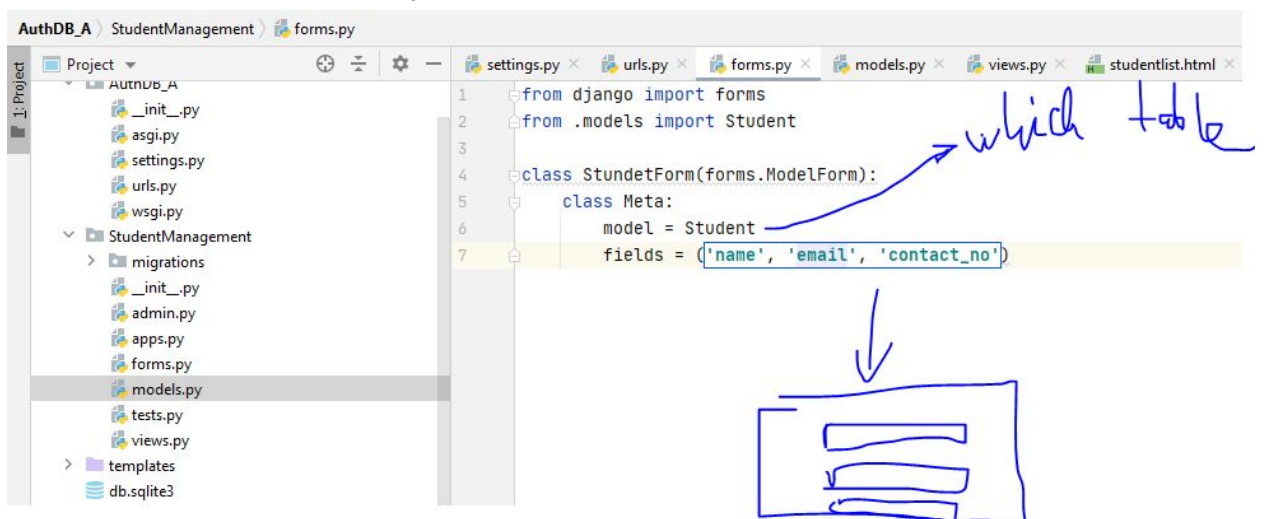
Contact no:

Save and add another

Save and continue editing

SAVE

1. Create a new file name forms.py and add the codes like this

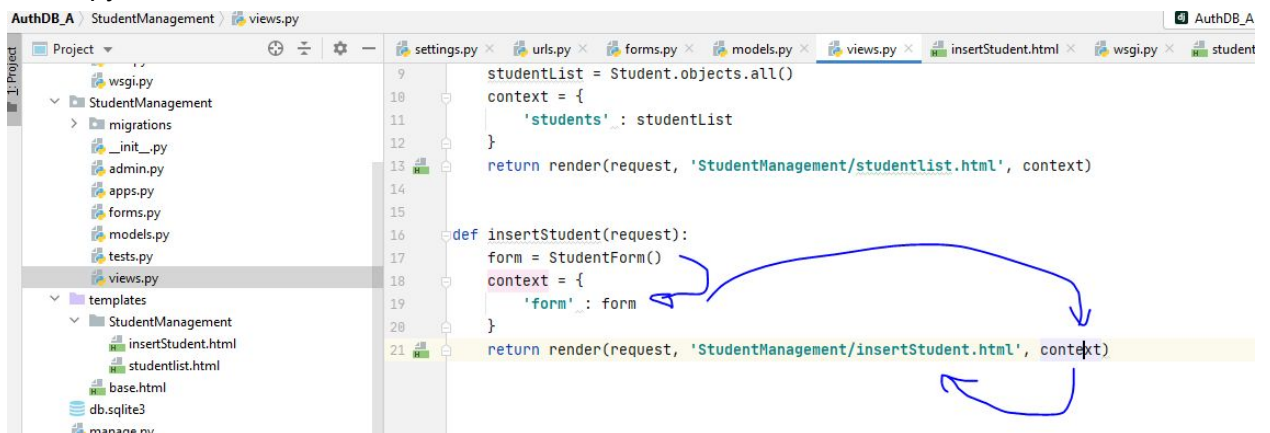


```
AuthDB_A StudentManagement forms.py
Project
  AuthDB_A
    __init__.py
    asgi.py
    settings.py
    urls.py
    wsgi.py
  StudentManagement
    migrations
    __init__.py
    admin.py
    apps.py
    forms.py
    models.py
    tests.py
    views.py
  templates
  db.sqlite3

1 from django import forms
2 from .models import Student
3
4 class StudentForm(forms.ModelForm):
5     class Meta:
6         model = Student
7         fields = ('name', 'email', 'contact_no')
```

Handwritten notes: "which table" with an arrow pointing to the `model = Student` line, and a diagram of a form with three input fields.

2. Views.py

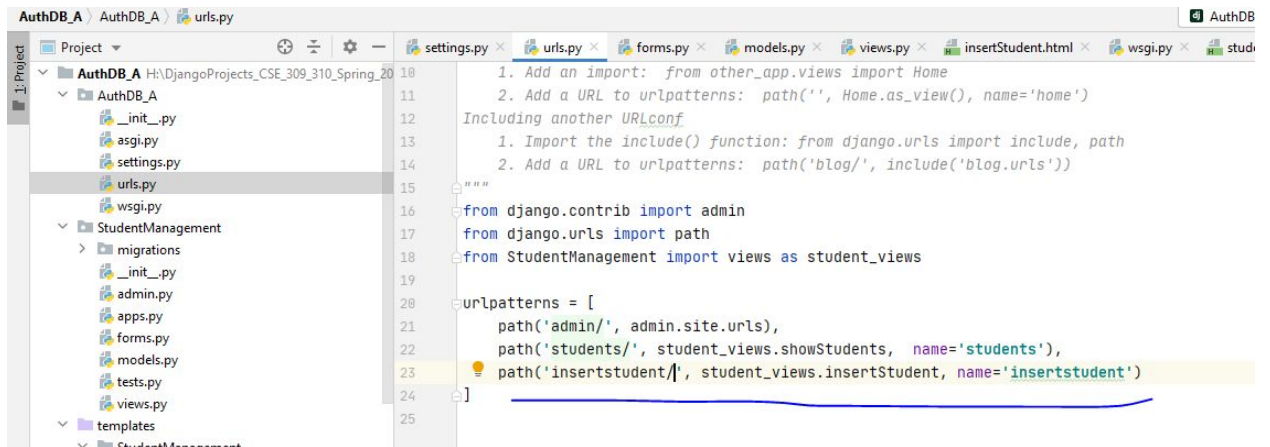


```
AuthDB_A StudentManagement views.py
Project
  AuthDB_A
    wsgi.py
  StudentManagement
    migrations
    __init__.py
    admin.py
    apps.py
    forms.py
    models.py
    tests.py
    views.py
  templates
    StudentManagement
      insertStudent.html
      studentlist.html
    base.html
    db.sqlite3
    manage.py

9 studentList = Student.objects.all()
10 context = {
11     'students': studentList
12 }
13 return render(request, 'StudentManagement/studentlist.html', context)
14
15
16 def insertStudent(request):
17     form = StudentForm()
18     context = {
19         'form': form
20     }
21     return render(request, 'StudentManagement/insertStudent.html', context)
```

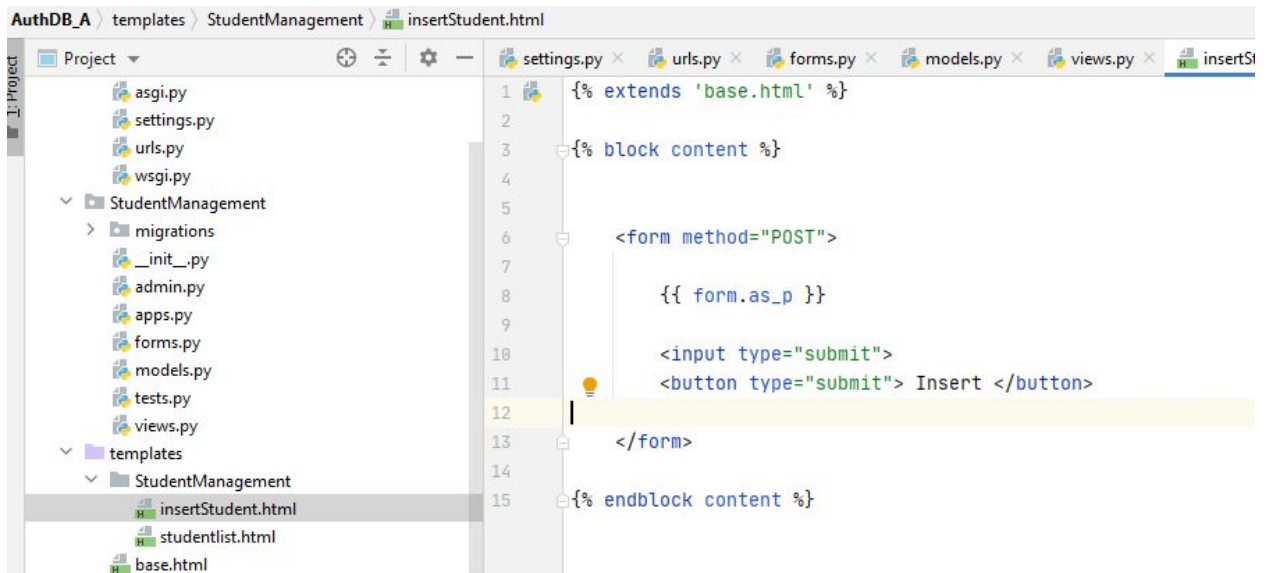
Handwritten notes: Arrows pointing from the `form = StudentForm()` line to the `'form': form` line in the context dictionary, and from the `context` argument in the `render` function call.

3. urls

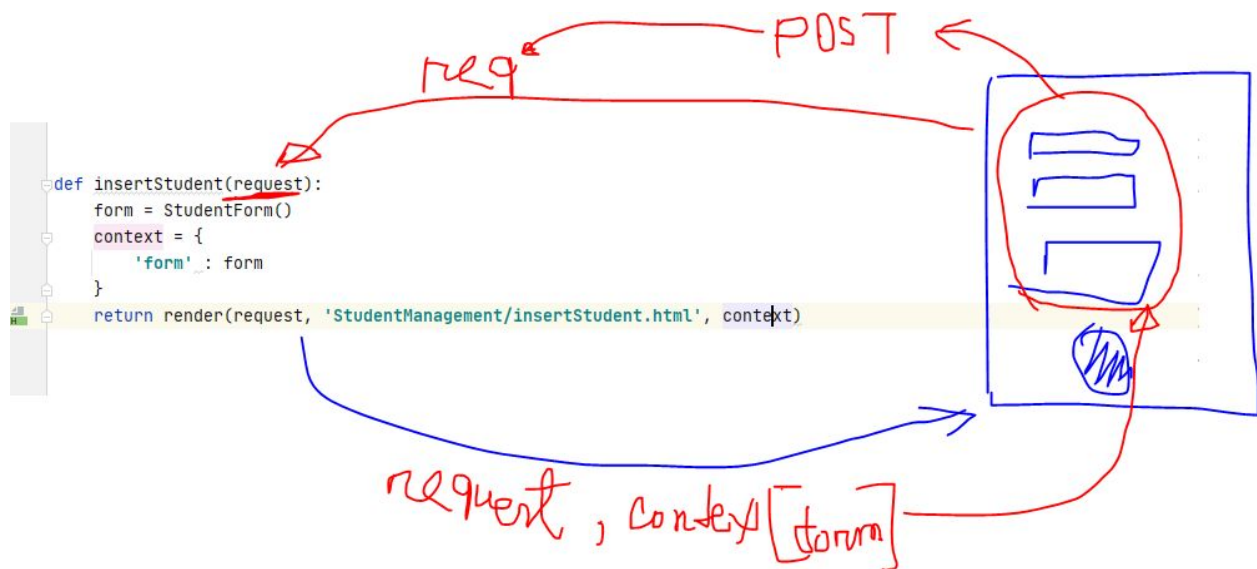


```
18 1. Add an import: from other_app.views import Home
11 2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13 1. Import the include() function: from django.urls import include, path
14 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15 """
16 from django.contrib import admin
17 from django.urls import path
18 from StudentManagement import views as student_views
19
20 urlpatterns = [
21     path('admin/', admin.site.urls),
22     path('students/', student_views.showStudents, name='students'),
23     path('insertstudent/', student_views.insertStudent, name='insertstudent')
24 ]
25
```

4. Html file



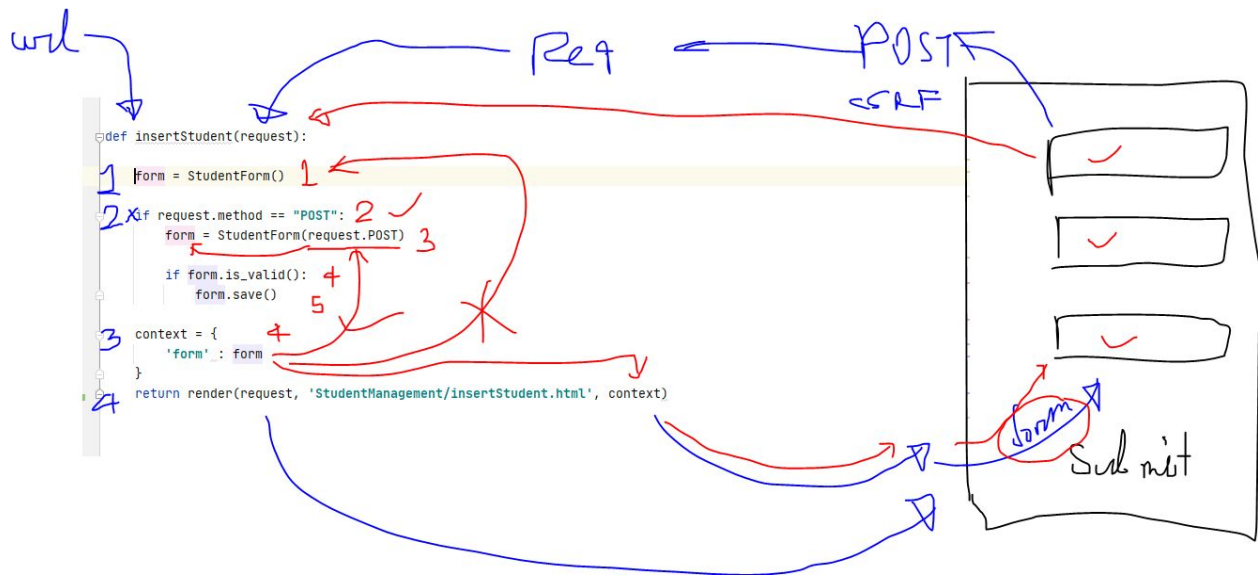
```
1 {% extends 'base.html' %}
2
3 {% block content %}
4
5
6     <form method="POST">
7
8         {{ form.as_p }}
9
10        <input type="submit">
11        <button type="submit"> Insert </button>
12
13    </form>
14
15 {% endblock content %}
```



5. Save into database

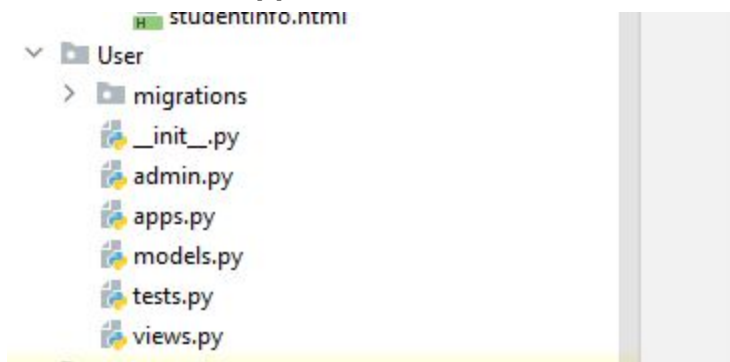
The screenshot shows a Django project in an IDE. The left sidebar displays the project structure, including the `StudentManagement` app. The main editor shows the `views.py` file with the `insertStudent` function. The function is updated to handle POST requests and save the form to the database.

```
def insertStudent(request):  
    message = ""  
    form = StudentForm()  
  
    if request.method == "POST":  
        form = StudentForm(request.POST)  
        message = "Invalid input. Please try again!"  
        if form.is_valid():  
            form.save()  
            message = "Student is inserted to DB. You can insert a new student now"  
            form = StudentForm()  
  
    context = {  
        'form': form,  
        'message': message  
    }  
    return render(request, 'StudentManagement/insertStudent.html', context)
```



Django Authentication

1. Create an User application



2. Create a registration html

DatabaseB2 > templates > User > register.html

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <title>Title</title>
6 </head>
7 <body>
8 <h2> you will the registration form here</h2>
9
10 </body>
11 </html>

```

3. Create views for registration html page

DatabaseB2 > User > views.py

```

1 from django.shortcuts import render
2
3 # Create your views here.
4 def user_registration(request):
5     return render(request, 'User/register.html')

```

4. Connect views with main urls

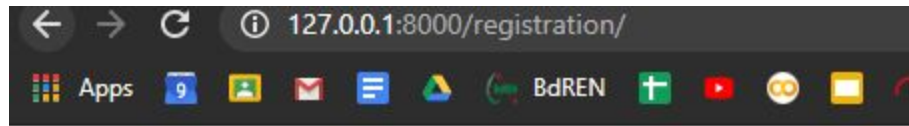
DatabaseB2 > DatabaseB2 > urls.py

```

9 Class-based views
10 1. Add an import: from other_app.views import Home
11 2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
12 Including another URLconf
13 1. Import the include() function: from django.urls import include, path
14 2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
15
16 from django.contrib import admin
17 from django.urls import path
18 from Student import views as student_views
19 from User import views as user_views
20
21 urlpatterns = [
22     path('admin/', admin.site.urls),
23     path('students/', student_views.studentInfo),
24     path('registration/', user_views.user_registration)
25 ]

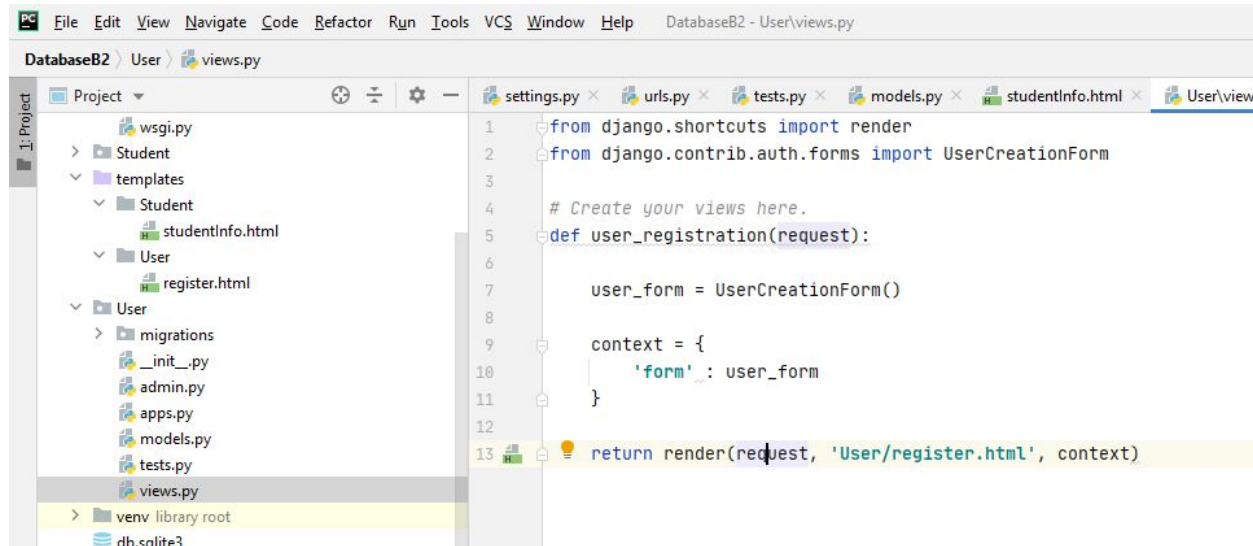
```

5. Runserver to check the page

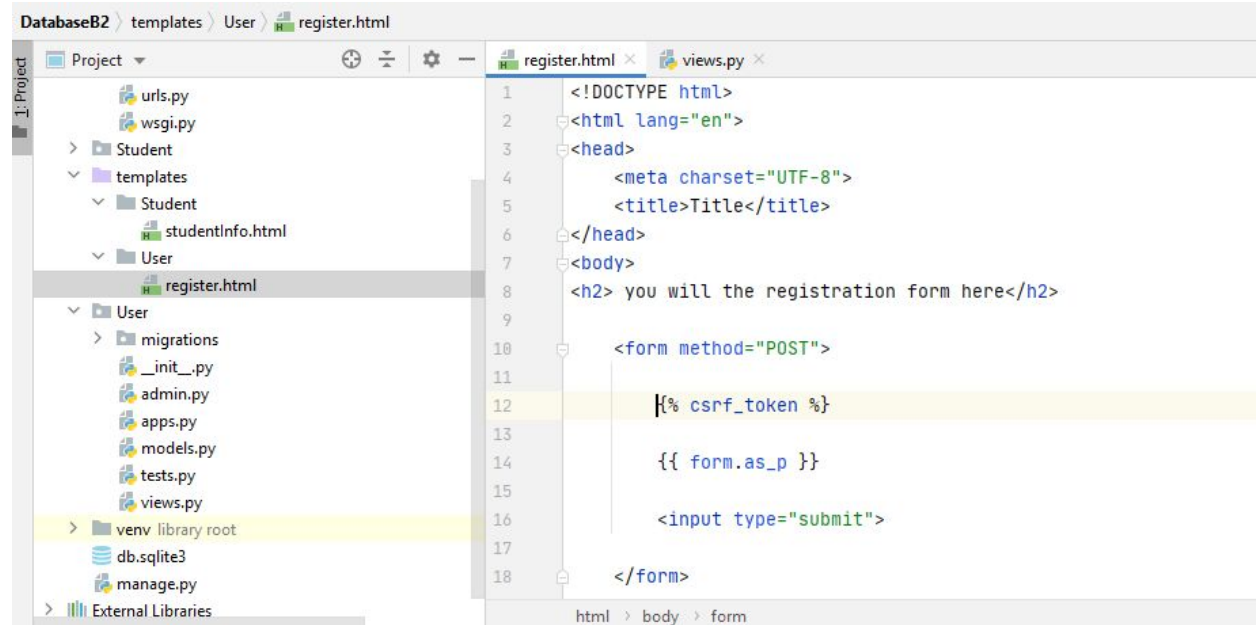


you will the registration form here

6. Import UserCreationForm in views.py



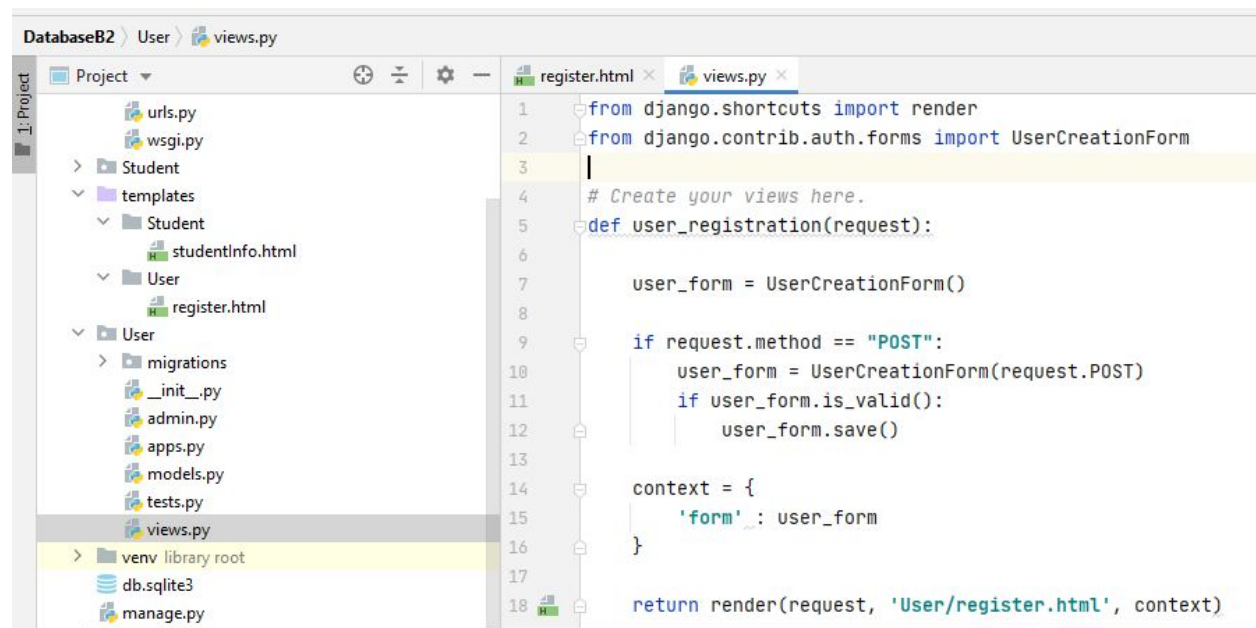
7. Add form in html page



```
DatabaseB2 > templates > User > register.html
Project
  urls.py
  wsgi.py
  Student
  templates
    Student
      studentInfo.html
    User
      register.html
  migrations
  __init__.py
  admin.py
  apps.py
  models.py
  tests.py
  views.py
  venv library root
  db.sqlite3
  manage.py
  External Libraries

register.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Title</title>
6 </head>
7 <body>
8   <h2> you will the registration form here</h2>
9
10  <form method="POST">
11
12    {% csrf_token %}
13
14    {{ form.as_p }}
15
16    <input type="submit">
17
18  </form>
```

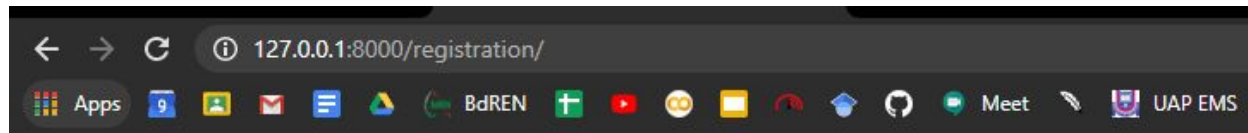
8. Handle POST request in the views.py



```
DatabaseB2 > User > views.py
Project
  urls.py
  wsgi.py
  Student
  templates
    Student
      studentInfo.html
    User
      register.html
  migrations
  __init__.py
  admin.py
  apps.py
  models.py
  tests.py
  views.py
  venv library root
  db.sqlite3
  manage.py

views.py
1 from django.shortcuts import render
2 from django.contrib.auth.forms import UserCreationForm
3
4 # Create your views here.
5 def user_registration(request):
6
7     user_form = UserCreationForm()
8
9     if request.method == "POST":
10         user_form = UserCreationForm(request.POST)
11         if user_form.is_valid():
12             user_form.save()
13
14     context = {
15         'form': user_form
16     }
17
18     return render(request, 'User/register.html', context)
```

9. Go to registration page from server



you will the registration form here

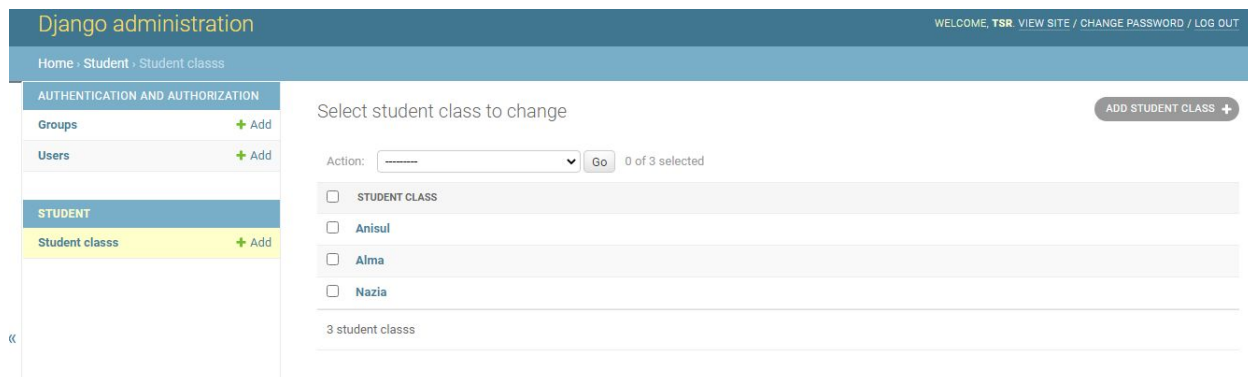
Username: Required. 150 characters or fewer. Letters, digits and @/./+/_ only.

Password:

- Your password can't be too similar to your other personal information.
- Your password must contain at least 8 characters.
- Your password can't be a commonly used password.
- Your password can't be entirely numeric.

Password confirmation: Enter the same password as before, for verification.

10. Submit information correctly



Static

