2CSDE86 Application Development Frameworks

Lab-3 Task

Submitted by: Labdhi Sheth 18BCE101

Aim:

Perform CRUD operation and connect database with the website. Store all the data and retrieve the data accordingly.

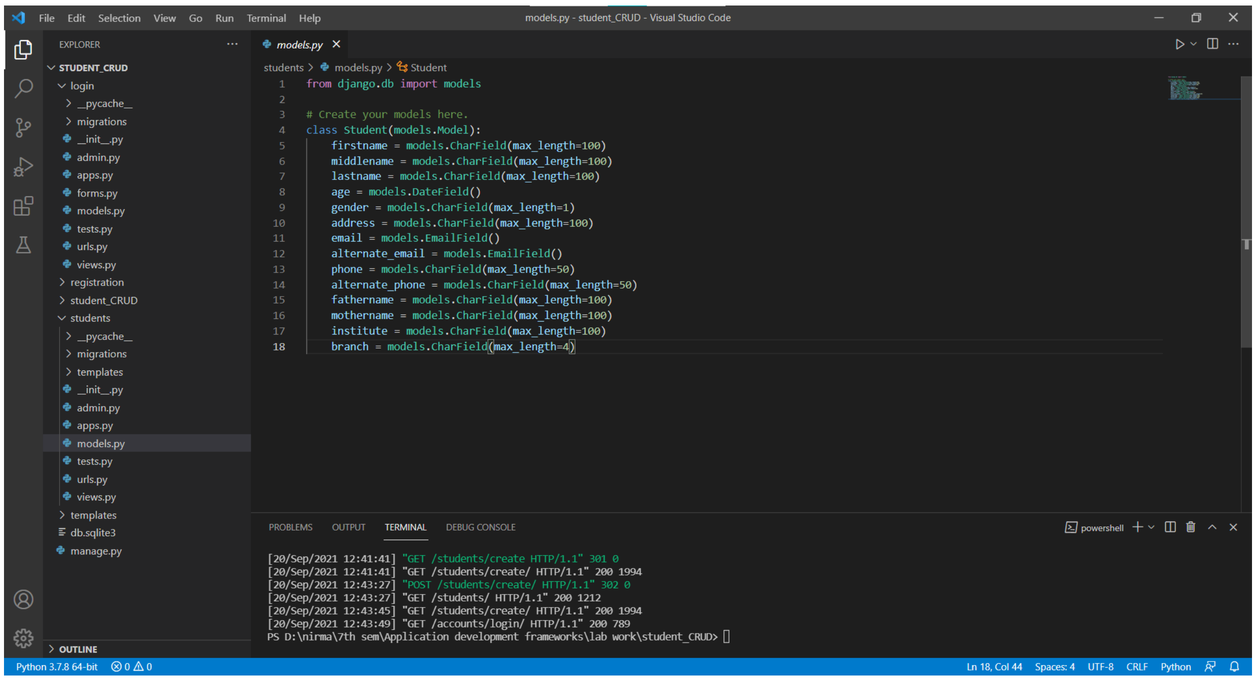
Methodology:

1. Open the anaconda terminal and write the command
   django-admin.py startproject student_CRUD

2. Open the project in vscode and open new terminal.
   In the terminal write the commands to start a new app:
   python manage.py startapp students
   python manage.py startapp login
   python manage.py startapp registration

   After making the apps make the templates folder as:
   cd students
   mkdir templates

3. students app
   a. models.py
      Under class Student the model is made with reference to the different fields in CRUD operation and the type of variables. This model will be used further to perform the CRUD operations.

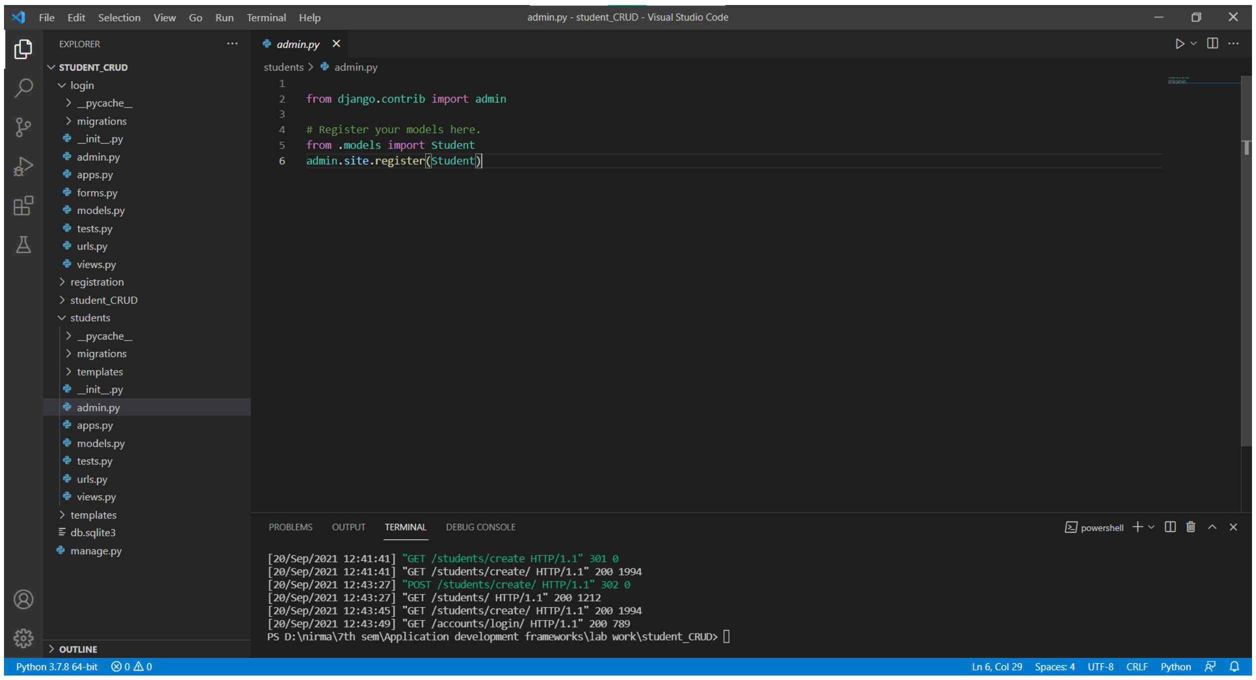


```python
from django.db import models

# Create your models here.
class Student(models.Model):
    firstname = models.CharField(max_length=100)
    middlename = models.CharField(max_length=100)
    lastname = models.CharField(max_length=100)
    age = models.DateField()
    gender = models.CharField(max_length=1)
    address = models.CharField(max_length=100)
    email = models.EmailField()
    alternate_email = models.EmailField()
    phone = models.CharField(max_length=50)
    alternate_phone = models.CharField(max_length=50)
    fathername = models.CharField(max_length=100)
    mothername = models.CharField(max_length=100)
    institute = models.CharField(max_length=100)
    branch = models.CharField(max_length=4)
```

```
[20/Sep/2021 12:41:41] "GET /students/create HTTP/1.1" 301 0
[20/Sep/2021 12:41:41] "GET /students/create/ HTTP/1.1" 200 1994
[20/Sep/2021 12:43:27] "POST /students/create/ HTTP/1.1" 302 0
[20/Sep/2021 12:43:27] "GET /students/ HTTP/1.1" 200 1212
[20/Sep/2021 12:43:45] "GET /students/create/ HTTP/1.1" 200 1994
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

b. admin.py

Next we are registering our model in django administration using admin.py.

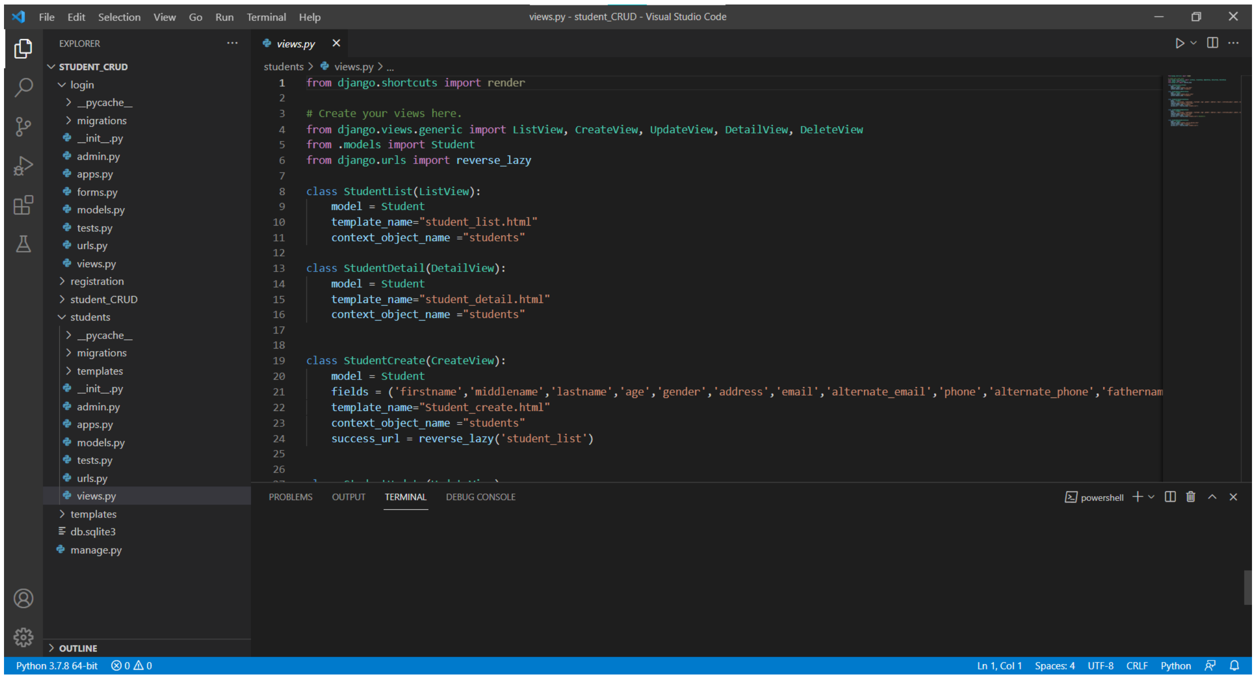


```python

from django.contrib import admin

# Register your models here.
from .models import Student
admin.site.register(Student)
```

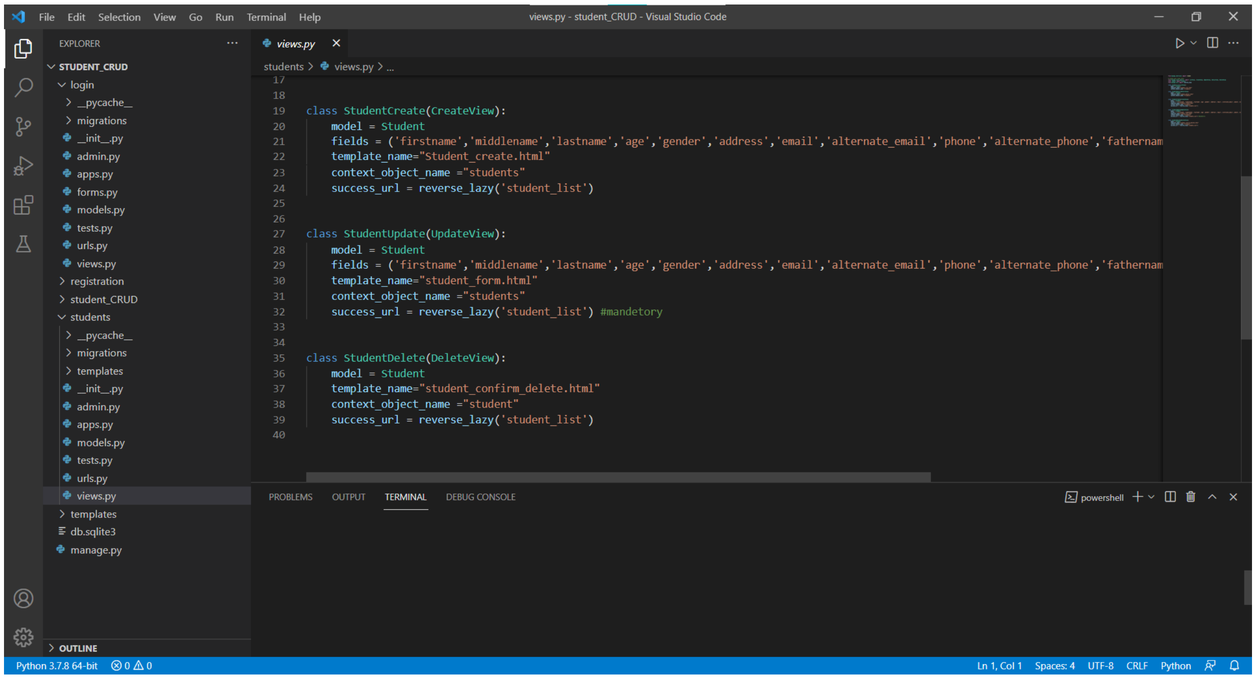
```
[20/Sep/2021 12:41:41] "GET /students/create HTTP/1.1" 301 0
[20/Sep/2021 12:41:41] "GET /students/create/ HTTP/1.1" 200 1994
[20/Sep/2021 12:43:27] "POST /students/create/ HTTP/1.1" 302 0
[20/Sep/2021 12:43:27] "GET /students/ HTTP/1.1" 200 1212
[20/Sep/2021 12:43:45] "GET /students/create/ HTTP/1.1" 200 1994
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

c. views.py

We have created class based views here for displaying the lists, creating, updating, and deleting the entries. Under model we are calling Students which is our registered model. Under templates we are calling the html file which will send the response and then we have context name for reference and success url which says that if this view has been successfully done then show this as the response.



```python
from django.shortcuts import render

# Create your views here.
from django.views.generic import ListView, CreateView, UpdateView, DetailView, DeleteView
from .models import Student
from django.urls import reverse_lazy

class StudentList(ListView):
    model = Student
    template_name="student_list.html"
    context_object_name ="students"

class StudentDetail(DetailView):
    model = Student
    template_name="student_detail.html"
    context_object_name ="students"


class StudentCreate(CreateView):
    model = Student
    fields = ('firstname','middlename','lastname','age','gender','address','email','alternate_email','phone','alternate_phone','fathernam
    template_name="Student_create.html"
    context_object_name ="students"
    success_url = reverse_lazy('student_list')

```



```python


class StudentCreate(CreateView):
    model = Student
    fields = ('firstname','middlename','lastname','age','gender','address','email','alternate_email','phone','alternate_phone','fathernam
    template_name="Student_create.html"
    context_object_name ="students"
    success_url = reverse_lazy('student_list')


class StudentUpdate(UpdateView):
    model = Student
    fields = ('firstname','middlename','lastname','age','gender','address','email','alternate_email','phone','alternate_phone','fathernam
    template_name="student_form.html"
    context_object_name ="students"
    success_url = reverse_lazy('student_list') #mandetory


class StudentDelete(DeleteView):
    model = Student
    template_name="student_confirm_delete.html"
    context_object_name ="student"
    success_url = reverse_lazy('student_list')

```

d. Html files under the templates:

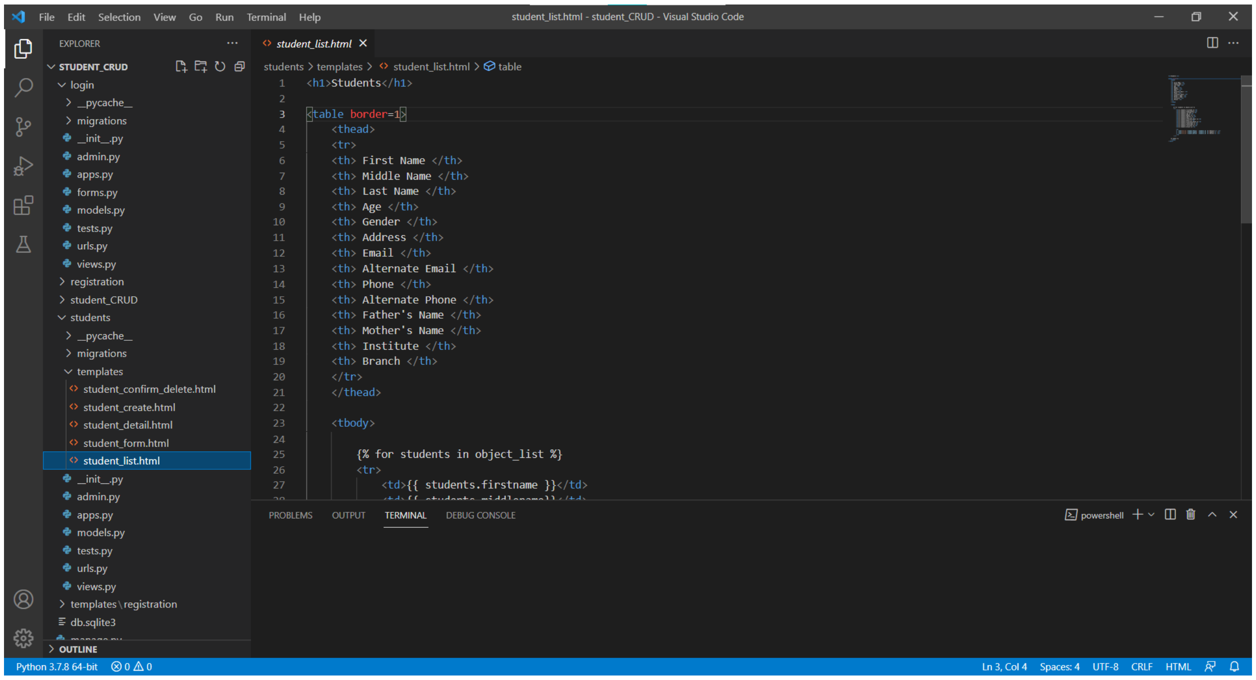
- student_list.html

  Create it using

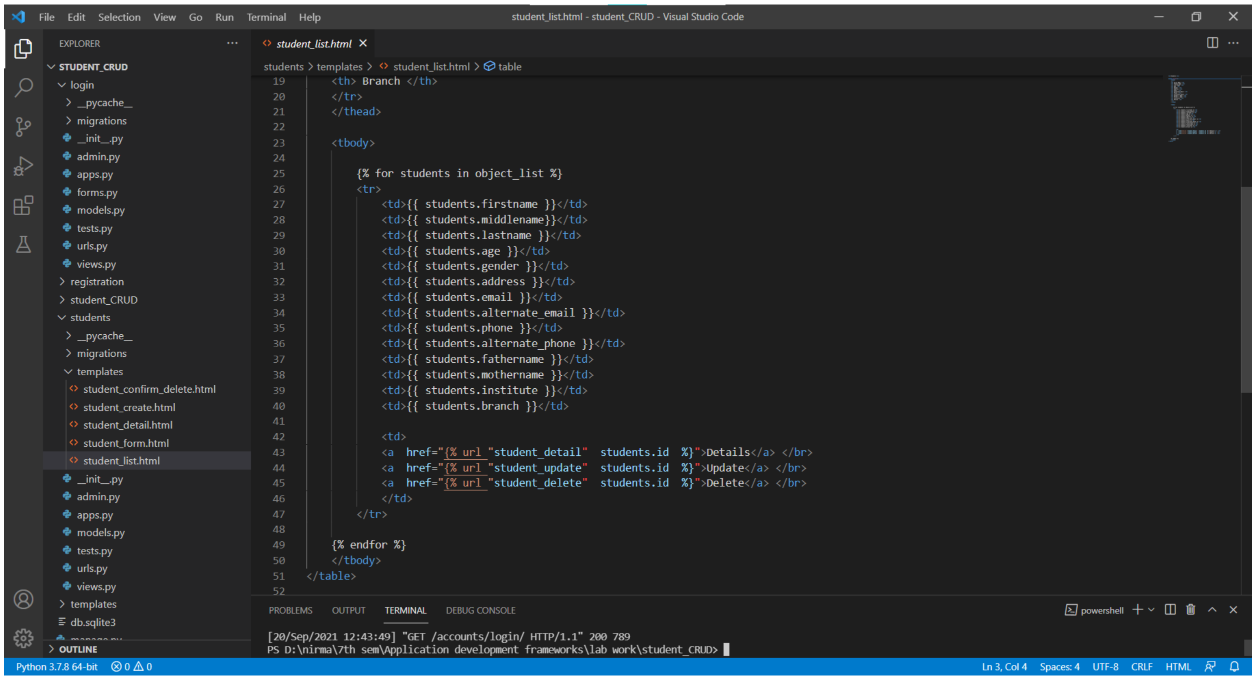
  `cd students`

  `cd templates`

  `code -r student_list.html`



```
<h1>Students</h1>

<table border=1>
    <thead>
    <tr>
    <th> First Name </th>
    <th> Middle Name </th>
    <th> Last Name </th>
    <th> Age </th>
    <th> Gender </th>
    <th> Address </th>
    <th> Email </th>
    <th> Alternate Email </th>
    <th> Phone </th>
    <th> Alternate Phone </th>
    <th> Father's Name </th>
    <th> Mother's Name </th>
    <th> Institute </th>
    <th> Branch </th>
    </tr>
    </thead>

    <tbody>

        {% for students in object_list %}
        <tr>
            <td>{{ students.firstname }}</td>
```



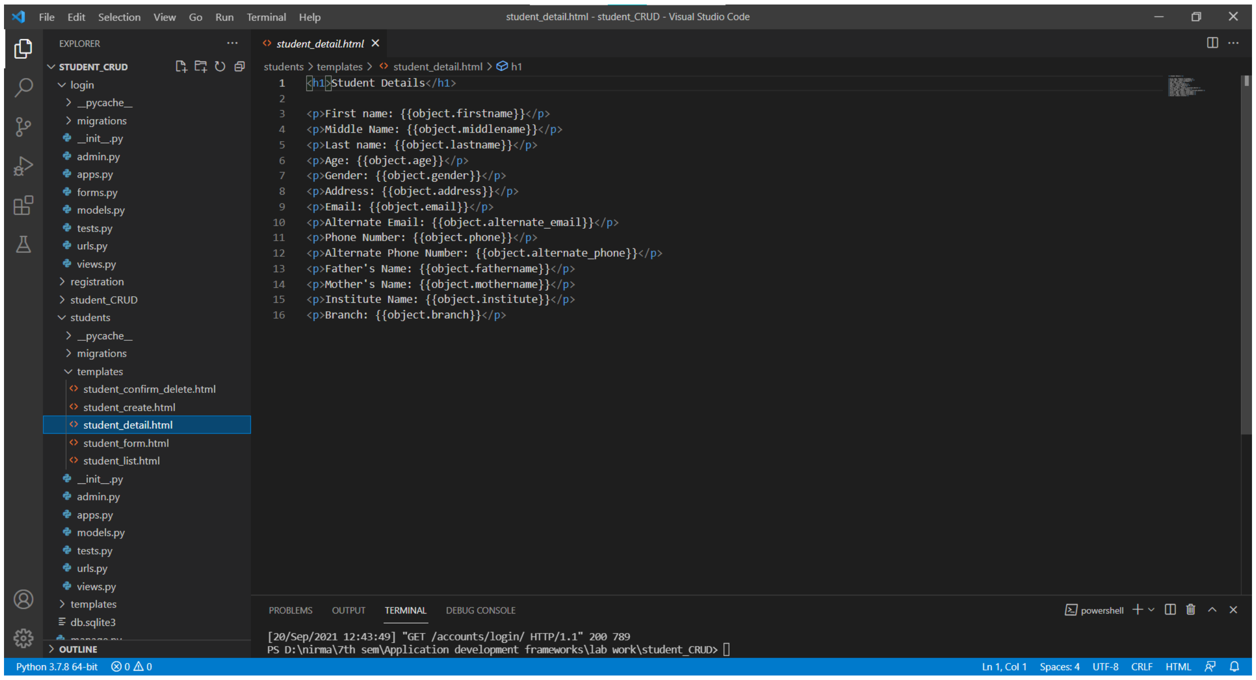
```
    <th> Branch </th>
    </tr>
    </thead>

    <tbody>

        {% for students in object_list %}
        <tr>
            <td>{{ students.firstname }}</td>
            <td>{{ students.middlename}}</td>
            <td>{{ students.lastname }}</td>
            <td>{{ students.age }}</td>
            <td>{{ students.gender }}</td>
            <td>{{ students.address }}</td>
            <td>{{ students.email }}</td>
            <td>{{ students.alternate_email }}</td>
            <td>{{ students.phone }}</td>
            <td>{{ students.alternate_phone }}</td>
            <td>{{ students.fathername }}</td>
            <td>{{ students.mothername }}</td>
            <td>{{ students.institute }}</td>
            <td>{{ students.branch }}</td>

            <td>
            <a  href="{% url "student_detail"  students.id  %}">Details</a> </br>
            <a  href="{% url "student_update"  students.id  %}">Update</a> </br>
            <a  href="{% url "student_delete"  students.id  %}">Delete</a> </br>
            </td>
        </tr>

    {% endfor %}
    </tbody>
</table>
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

- student_detail.html

  `code -r student_detail.html`

  This file creates a object so as to fetch the student details



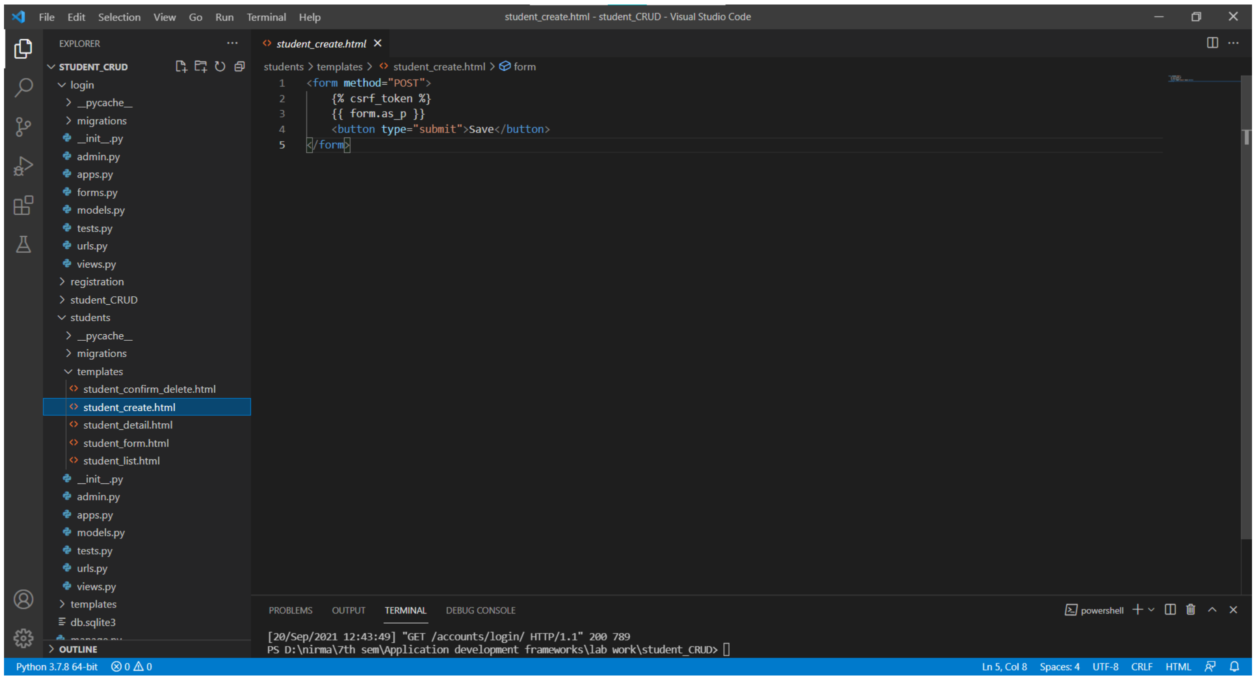
```
<h1>Student Details</h1>

<p>First name: {{object.firstname}}</p>
<p>Middle Name: {{object.middlename}}</p>
<p>Last name: {{object.lastname}}</p>
<p>Age: {{object.age}}</p>
<p>Gender: {{object.gender}}</p>
<p>Address: {{object.address}}</p>
<p>Email: {{object.email}}</p>
<p>Alternate Email: {{object.alternate_email}}</p>
<p>Phone Number: {{object.phone}}</p>
<p>Alternate Phone Number: {{object.alternate_phone}}</p>
<p>Father's Name: {{object.fathername}}</p>
<p>Mother's Name: {{object.mothername}}</p>
<p>Institute Name: {{object.institute}}</p>
<p>Branch: {{object.branch}}</p>
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

- student_create.html

`code -r student_create.html`

This file calls a form to fill in the details.



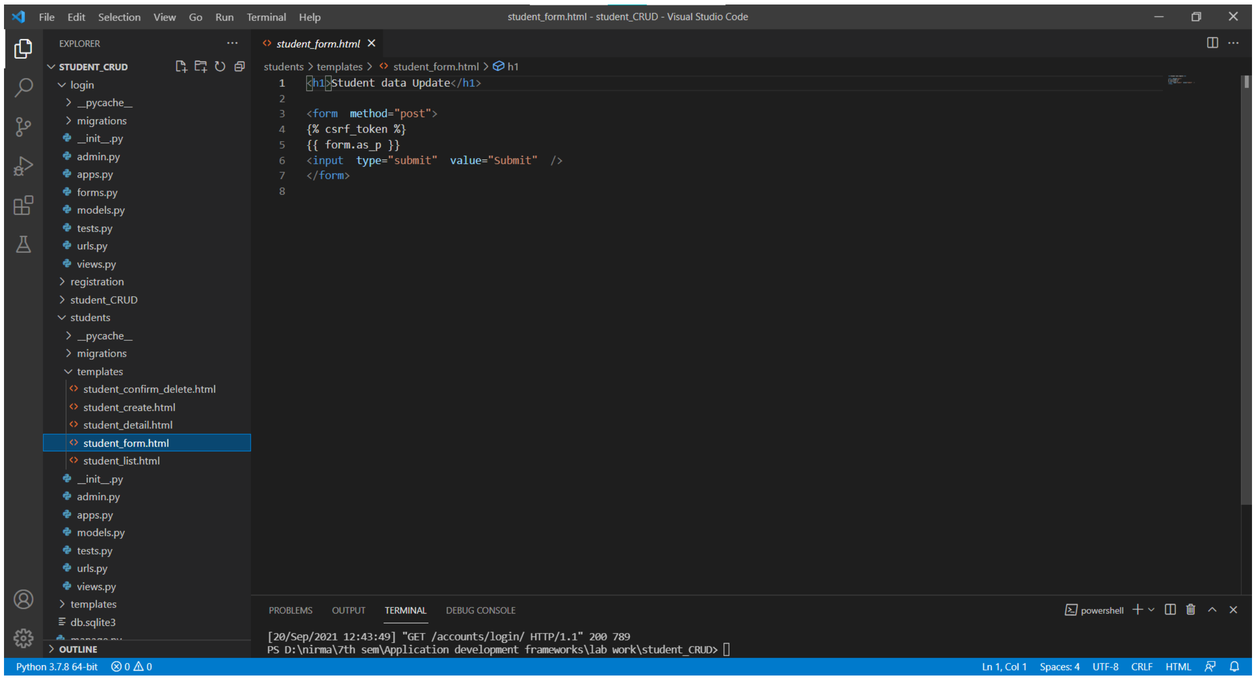
```
<form method="POST">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Save</button>
</form>
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

- student_form.html

```
code -r student_form.html
```

This file is for updating the records.



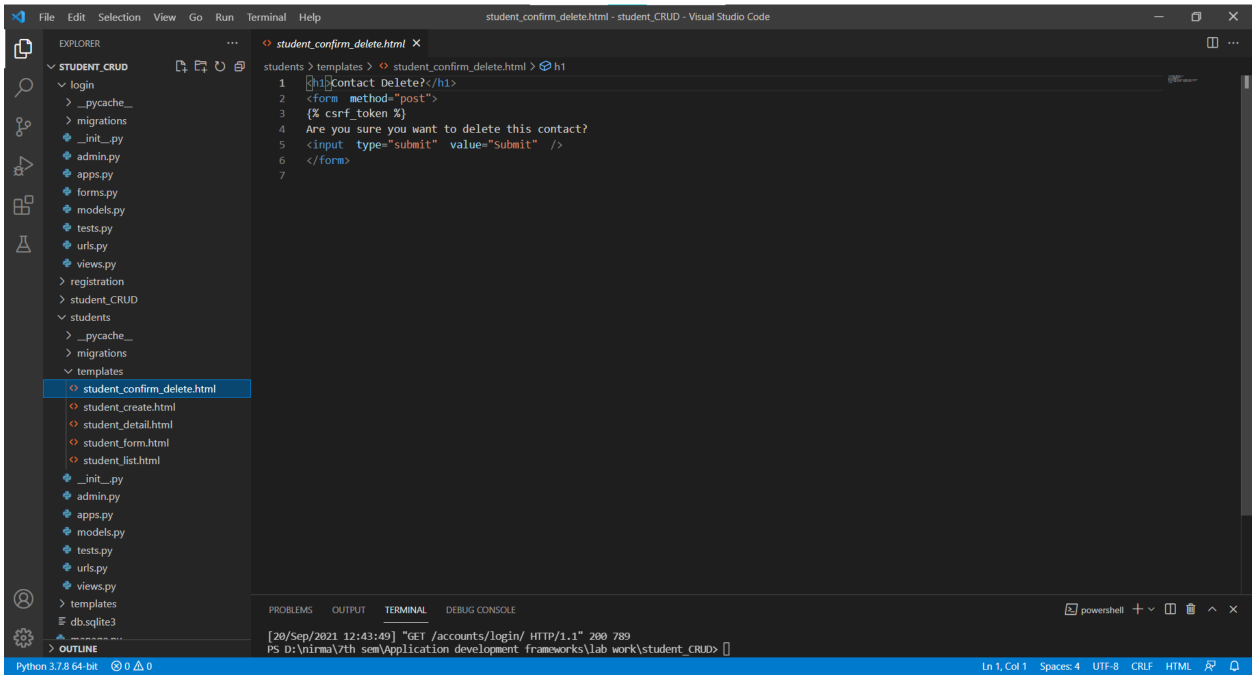
```
<h1>Student data Update</h1>

<form  method="post">
{% csrf_token %}
{{ form.as_p }}
<input  type="submit"  value="Submit"  />
</form>
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

- student_confirm_delete.html

```
code -r student_confirm_delete.html
```

This file is for deleing the records.

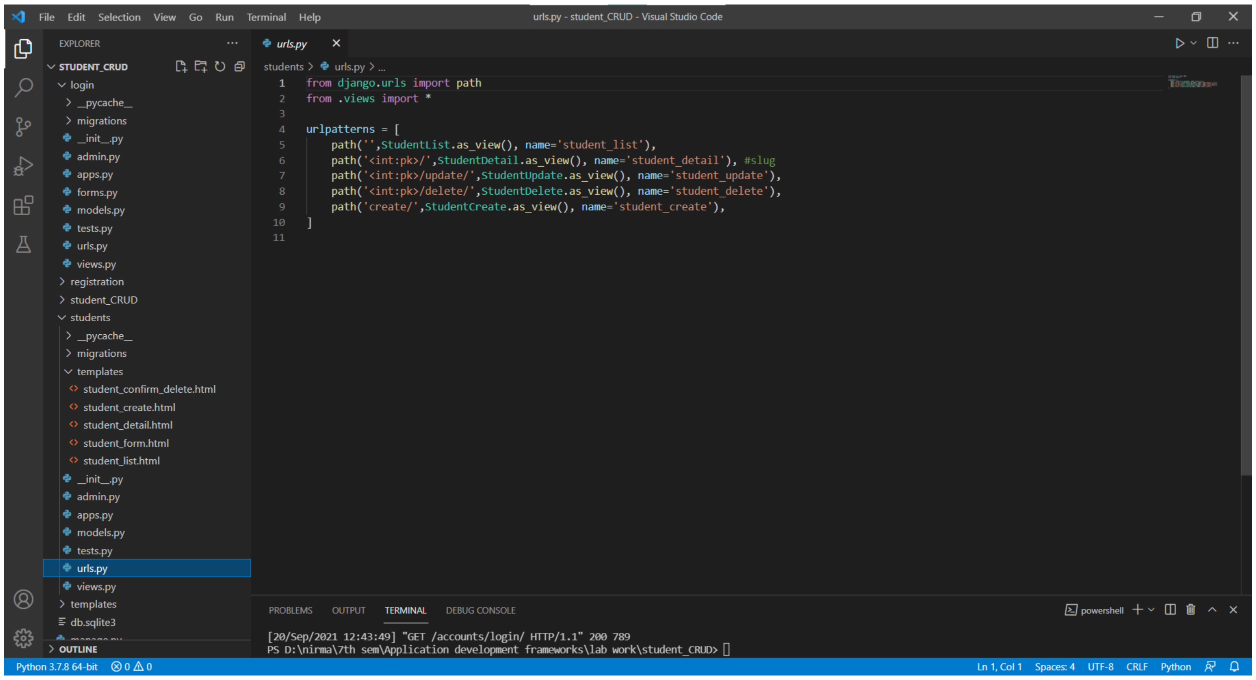


```
<h1>Contact Delete?</h1>
<form  method="post">
{% csrf_token %}
Are you sure you want to delete this contact?
<input  type="submit"  value="Submit"  />
</form>
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

e. urls.py

To call all these files create a specific pattern in urls.py. It calls by id.

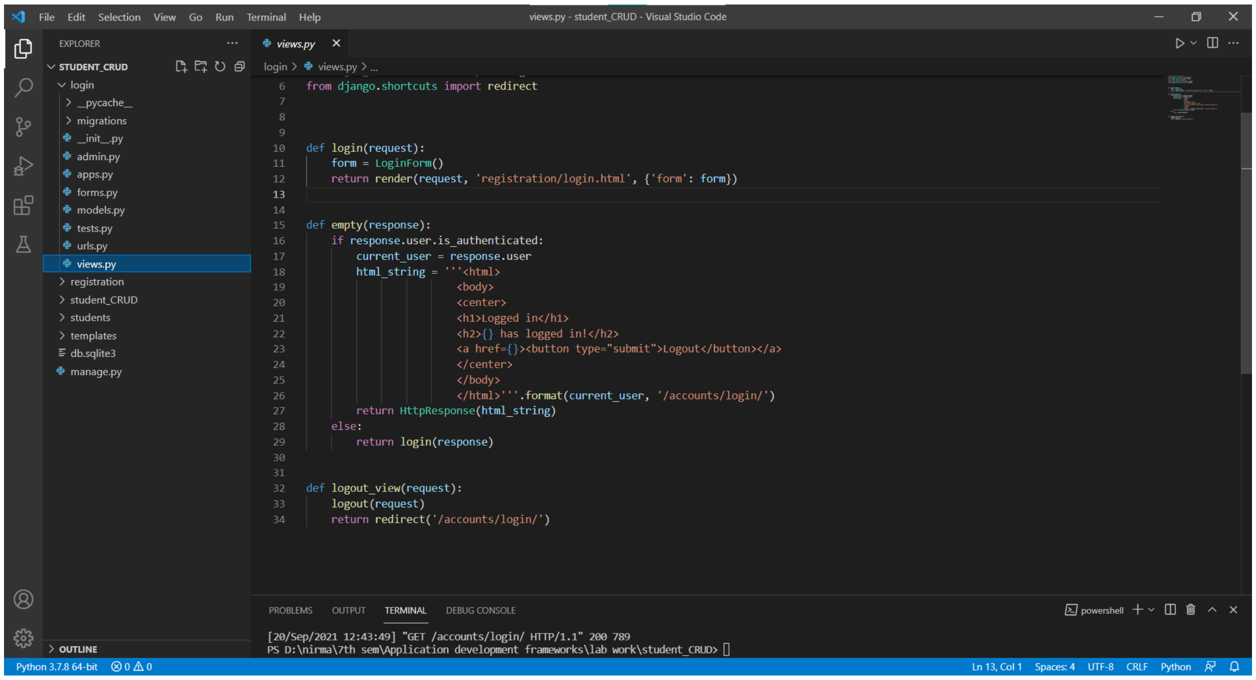


```python
from django.urls import path
from .views import *

urlpatterns = [
    path('',StudentList.as_view(), name='student_list'),
    path('<int:pk>/',StudentDetail.as_view(), name='student_detail'), #slug
    path('<int:pk>/update/',StudentUpdate.as_view(), name='student_update'),
    path('<int:pk>/delete/',StudentDelete.as_view(), name='student_delete'),
    path('create/',StudentCreate.as_view(), name='student_create'),
]
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

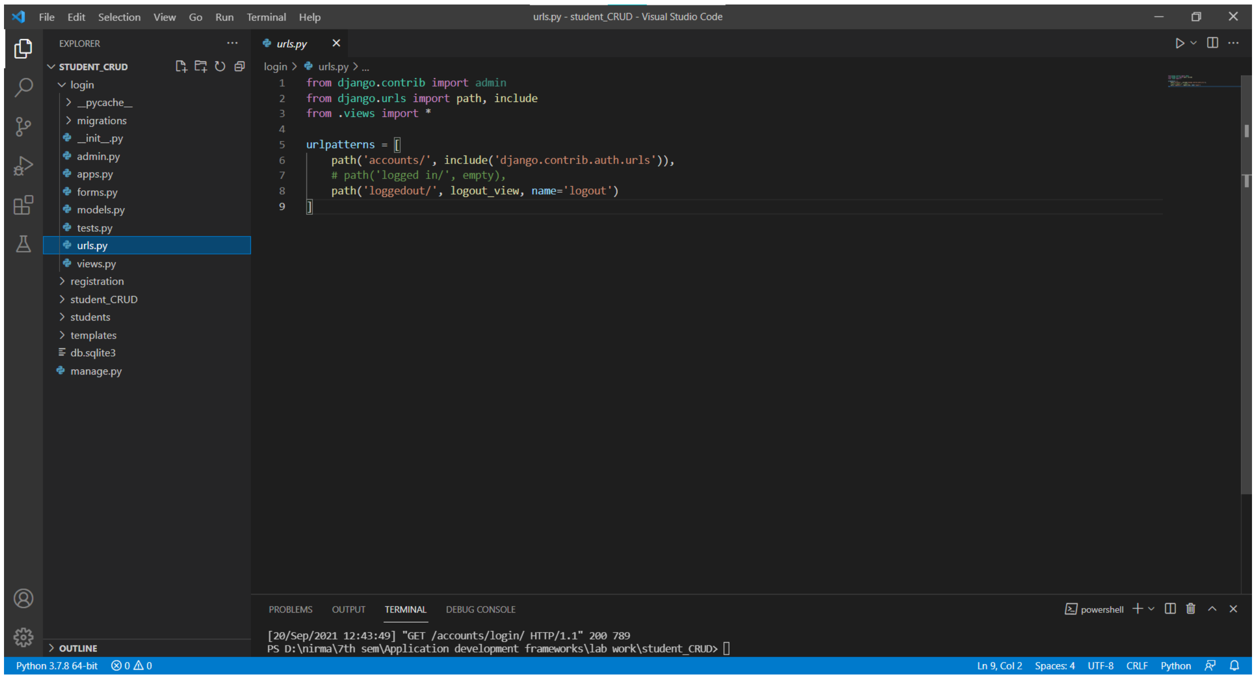
4. Login app

It is same as done in practical 2 the only differences is in the views.py and urls.py which is as follows:



```python
from django.shortcuts import redirect



def login(request):
    form = LoginForm()
    return render(request, 'registration/login.html', {'form': form})


def empty(response):
    if response.user.is_authenticated:
        current_user = response.user
        html_string = '''<html>
                    <body>
                    <center>
                    <h1>Logged in</h1>
                    <h2>{} has logged in!</h2>
                    <a href={}><button type="submit">Logout</button></a>
                    </center>
                    </body>
                    </html>'''.format(current_user, '/accounts/login/')
        return HttpResponse(html_string)
    else:
        return login(response)


def logout_view(request):
    logout(request)
    return redirect('/accounts/login/')
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```



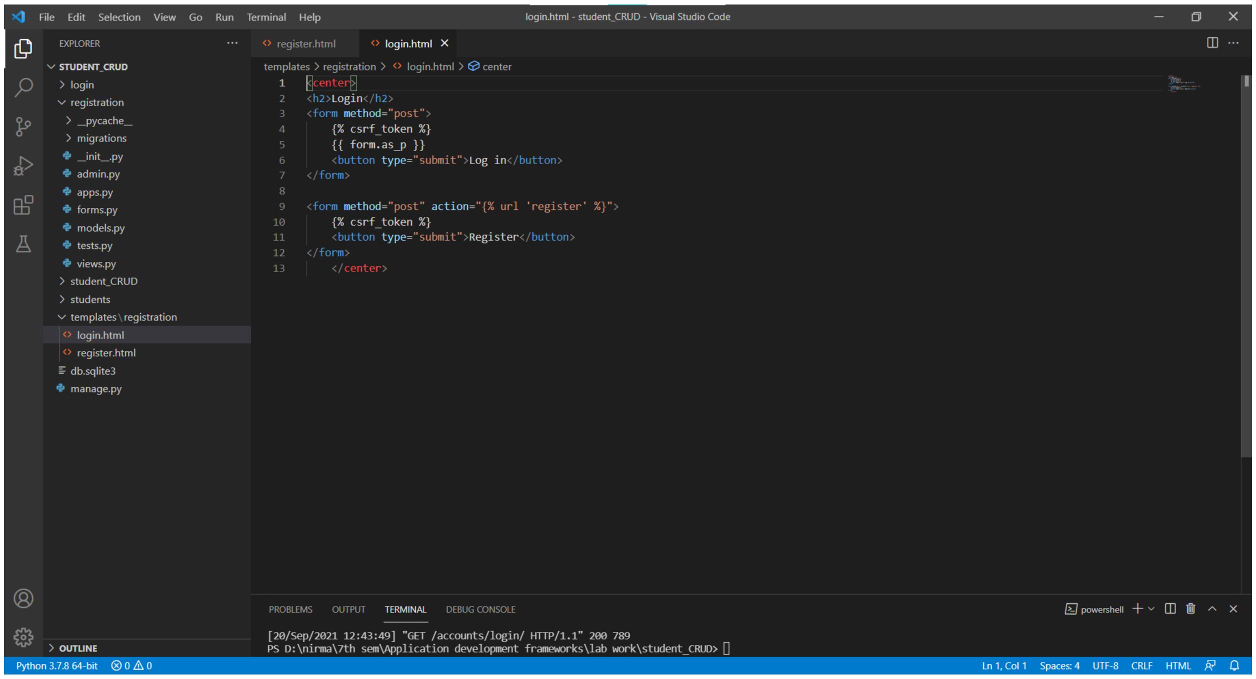
```python
from django.contrib import admin
from django.urls import path, include
from .views import *

urlpatterns = [
    path('accounts/', include('django.contrib.auth.urls')),
    # path('logged in/', empty),
    path('loggedout/', logout_view, name='logout')
]
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

5. Register app
Everything is same as practical 2.

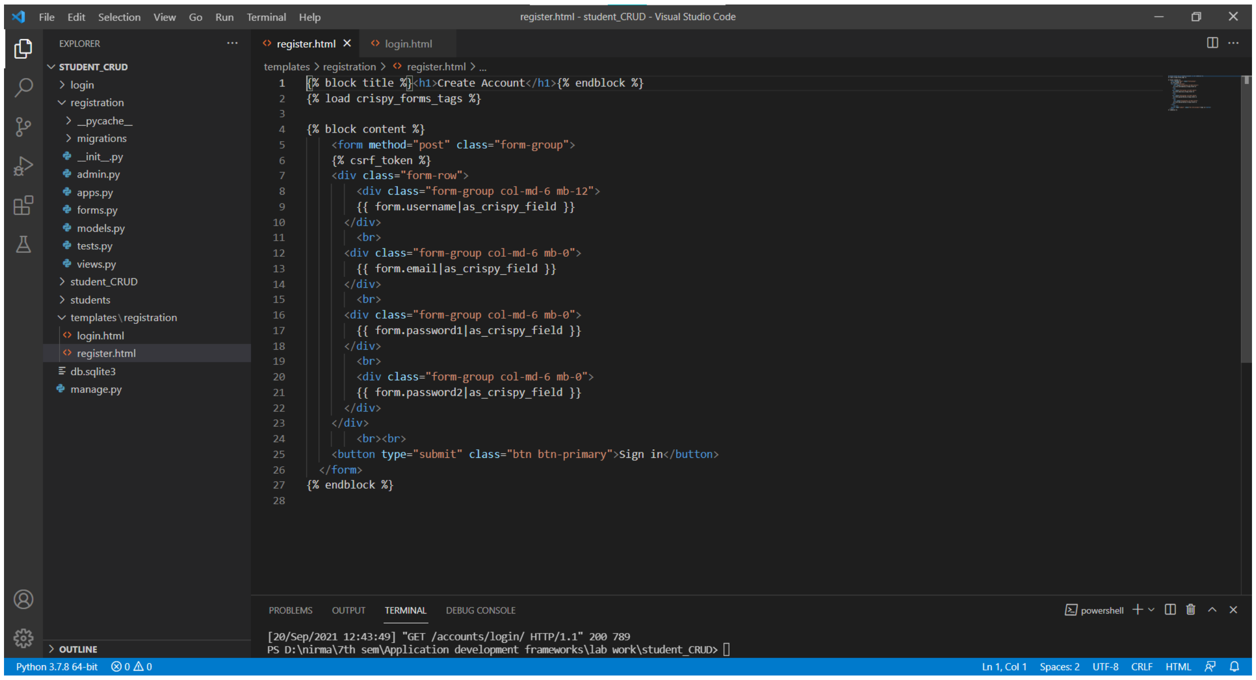
6. For calling the login.html and register.html go to new terminal /student_CRUD, and write

```
mkdir templates
cd templates
mkdir registration
cd registration
code -r login.html
code -r register.html
```



```
<center>
<h2>Login</h2>
<form method="post">
    {% csrf_token %}
    {{ form.as_p }}
    <button type="submit">Log in</button>
</form>

<form method="post" action="{% url 'register' %}">
    {% csrf_token %}
    <button type="submit">Register</button>
</form>
    </center>
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```



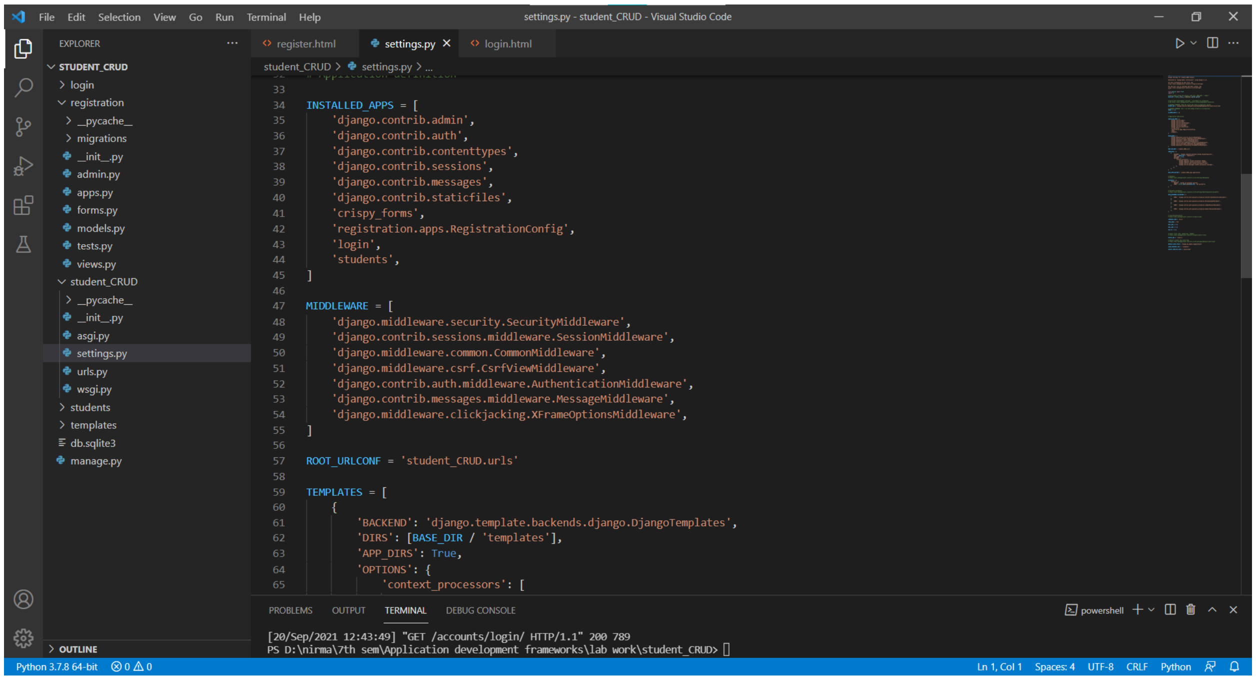
```
{% block title %}<h1>Create Account</h1>{% endblock %}
{% load crispy_forms_tags %}

{% block content %}
    <form method="post" class="form-group">
    {% csrf_token %}
    <div class="form-row">
        <div class="form-group col-md-6 mb-12">
        {{ form.username|as_crispy_field }}
      </div>
        <br>
      <div class="form-group col-md-6 mb-0">
        {{ form.email|as_crispy_field }}
      </div>
        <br>
      <div class="form-group col-md-6 mb-0">
        {{ form.password1|as_crispy_field }}
      </div>
        <br>
        <div class="form-group col-md-6 mb-0">
        {{ form.password2|as_crispy_field }}
      </div>
    </div>
        <br><br>
    <button type="submit" class="btn btn-primary">Sign in</button>
  </form>
{% endblock %}

```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

7. All the changes in the app and templates has been done, now we'll be adding to the main project files.

   a. settings.py
      Crispy forms, login, students and registration.apps has been added in IINSTALLED_APPS. Root url is of student.urls as the manin operation is

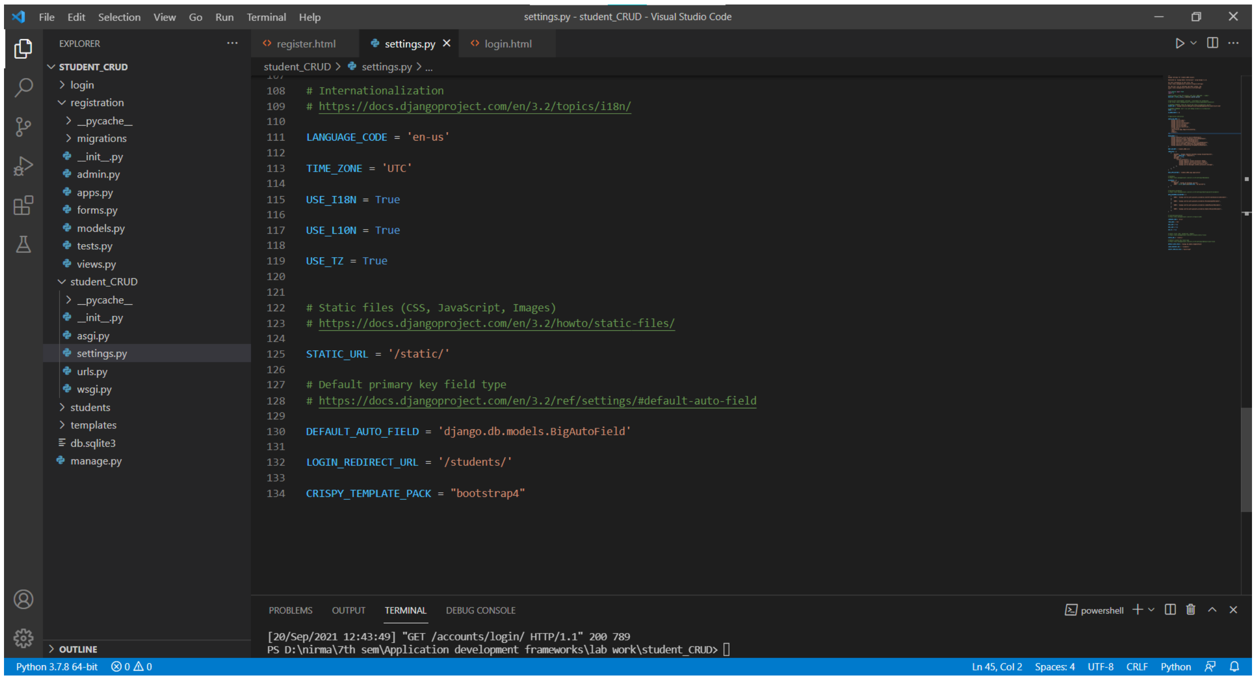
performed there and in templates list, base url of templates has been added so as to use login.html and register.html.



```
    # Application definition

INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'crispy_forms',
    'registration.apps.RegistrationConfig',
    'login',
    'students',
]

MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]

ROOT_URLCONF = 'student_CRUD.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [BASE_DIR / 'templates'],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

Add these in the end for calling bootstrap and after login call the students urls.

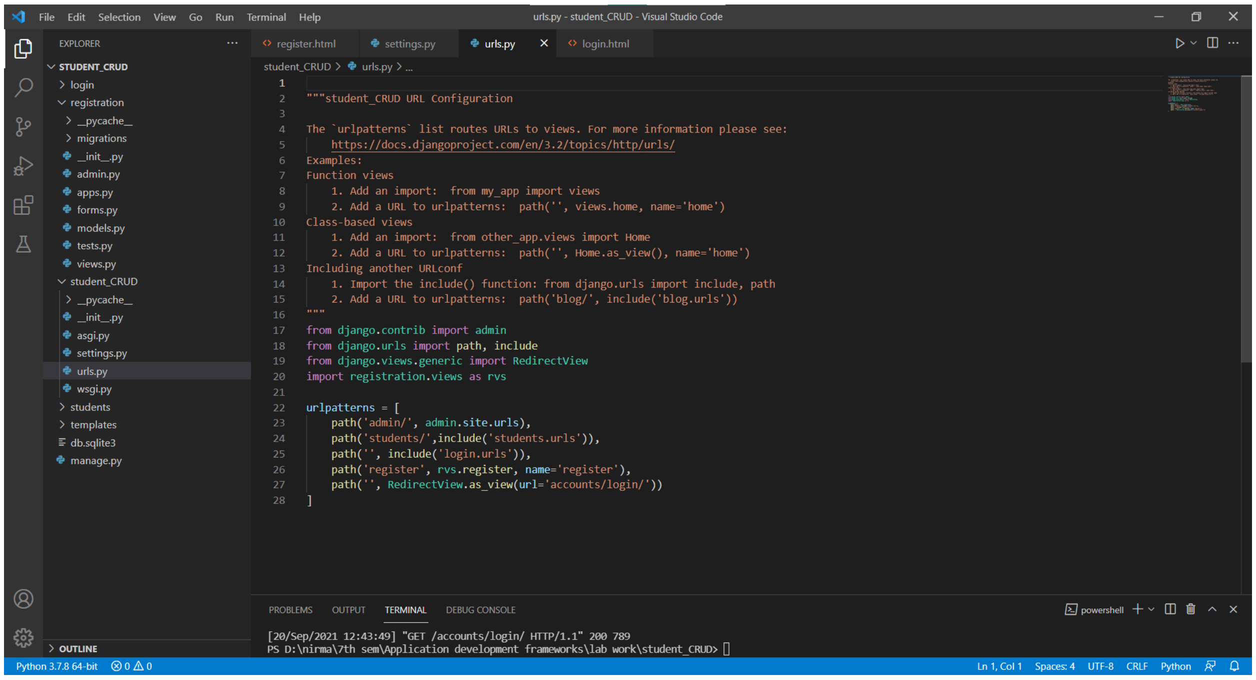


```
# Internationalization
# https://docs.djangoproject.com/en/3.2/topics/i18n/

LANGUAGE_CODE = 'en-us'

TIME_ZONE = 'UTC'

USE_I18N = True

USE_L10N = True

USE_TZ = True


# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC_URL = '/static/'

# Default primary key field type
# https://docs.djangoproject.com/en/3.2/ref/settings/#default-auto-field

DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'

LOGIN_REDIRECT_URL = '/students/'

CRISPY_TEMPLATE_PACK = "bootstrap4"
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

b. urls.py

Configure all the urls in this file. First it opens login page wherein the user either logs in or registers and thereafter it redirects to the students page so as to perform CRUD operation.



```python

"""student_CRUD URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import:  from my_app import views
    2. Add a URL to urlpatterns:  path('', views.home, name='home')
Class-based views
    1. Add an import:  from other_app.views import Home
    2. Add a URL to urlpatterns:  path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns:  path('blog/', include('blog.urls'))
"""
from django.contrib import admin
from django.urls import path, include
from django.views.generic import RedirectView
import registration.views as rvs

urlpatterns = [
    path('admin/', admin.site.urls),
    path('students/',include('students.urls')),
    path('', include('login.urls')),
    path('register', rvs.register, name='register'),
    path('', RedirectView.as_view(url='accounts/login/'))
]
```

```
[20/Sep/2021 12:43:49] "GET /accounts/login/ HTTP/1.1" 200 789
PS D:\nirma\7th sem\Application development frameworks\lab work\student_CRUD>
```

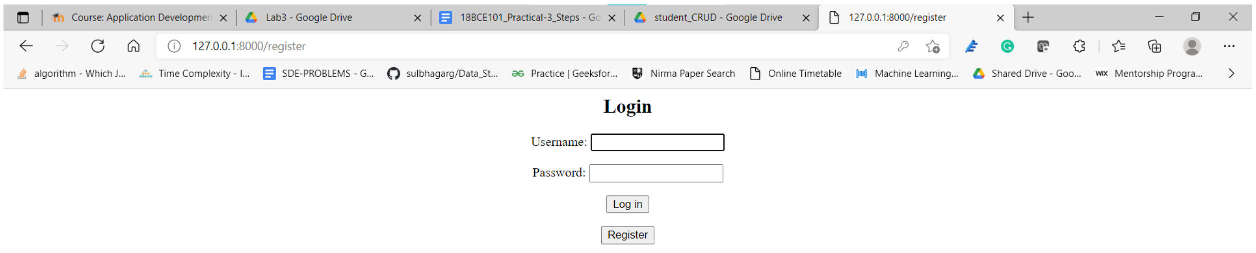
Output:

In the terminal run the following commands:

```
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
```

Click on 127.0.0.1:8000

1. Login page opens, click on register and register as follows.



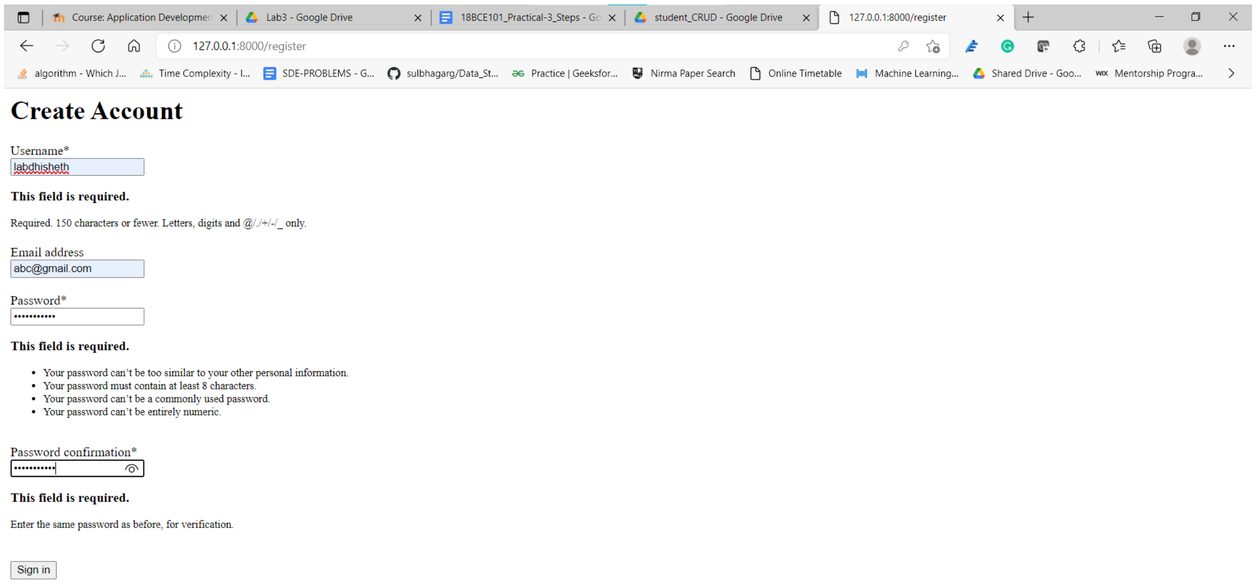
Login

Username:

Password:

Log in

Register



Create Account

Username*
labdhisheth

This field is required.

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

Email address
abc@gmail.com

Password*
•••••••••••

This field is required.

- 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.

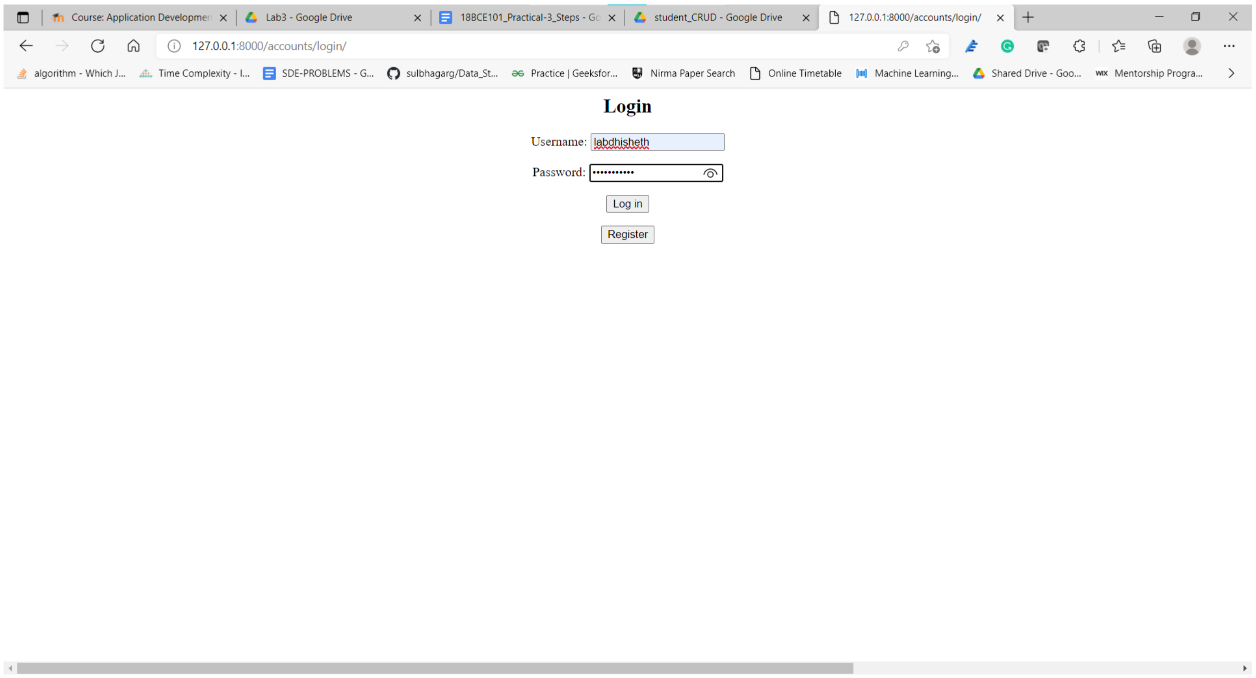
Password confirmation*
•••••••••••

This field is required.

Enter the same password as before, for verification.

Sign in

In the login page enter the details and press Log in button


Login

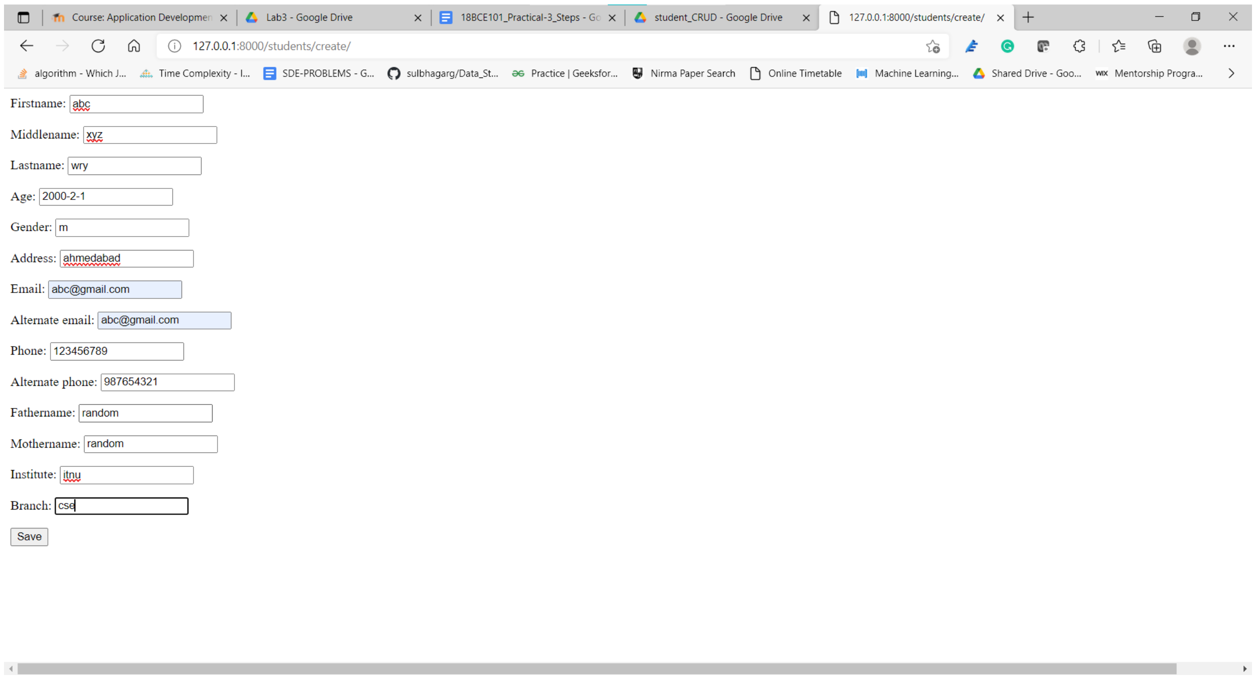
Username: labdhisheth

Password: ••••••••••••

Log in

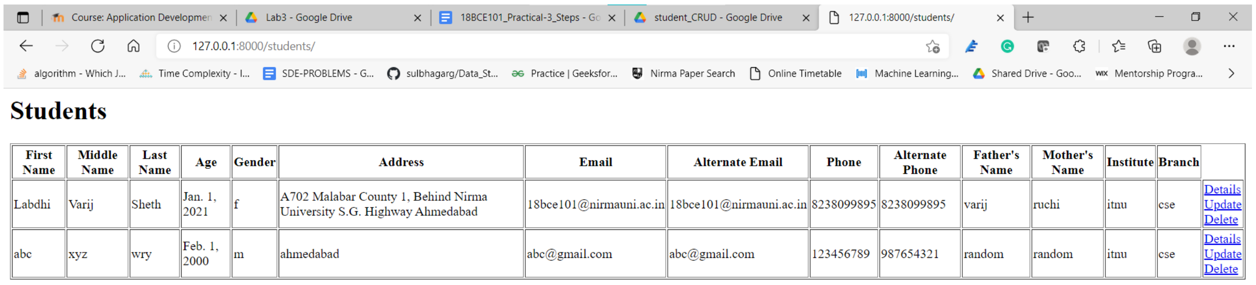
Register

2. Create operation:
   Add the details in the form and click on “Save” button.


Firstname:	abc
Middlename:	xyz
Lastname:	wry
Age:	2000-2-1
Gender:	m
Address:	ahmedabad
Email:	abc@gmail.com
Alternate email:	abc@gmail.com
Phone:	123456789
Alternate phone:	987654321
Fathername:	random
Mothername:	random
Institute:	itnu
Branch:	cse

Save

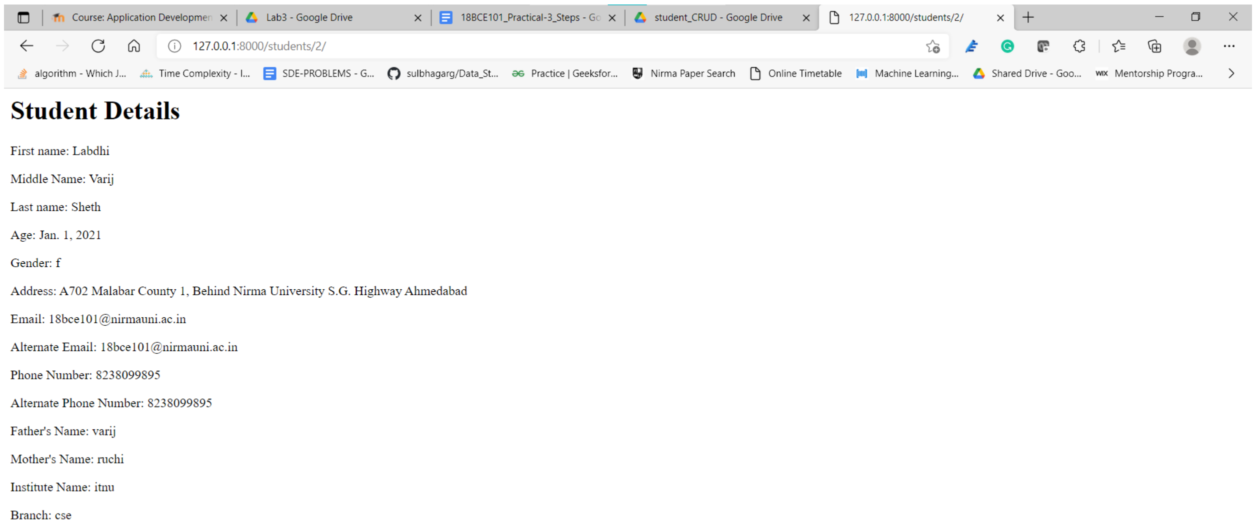
3. Performing this redirects to the students page where list of students is mentioned for example



Students

First Name	Middle Name	Last Name	Age	Gender	Address	Email	Alternate Email	Phone	Alternate Phone	Father's Name	Mother's Name	Institute	Branch	
Labdhi	Varij	Sheth	Jan. 1, 2021	f	A702 Malabar County 1, Behind Nirma University S.G. Highway Ahmedabad	18bce101@nirmauni.ac.in	18bce101@nirmauni.ac.in	8238099895	8238099895	varij	ruchi	itnu	cse	Details Update Delete
abc	xyz	wry	Feb. 1, 2000	m	ahmedabad	abc@gmail.com	abc@gmail.com	123456789	987654321	random	random	itnu	cse	Details Update Delete

4. To see the details, click on details in the right most side and it redirects to:



Student Details

First name: Labdhi

Middle Name: Varij

Last name: Sheth

Age: Jan. 1, 2021

Gender: f

Address: A702 Malabar County 1, Behind Nirma University S.G. Highway Ahmedabad

Email: 18bce101@nirmauni.ac.in

Alternate Email: 18bce101@nirmauni.ac.in

Phone Number: 8238099895

Alternate Phone Number: 8238099895

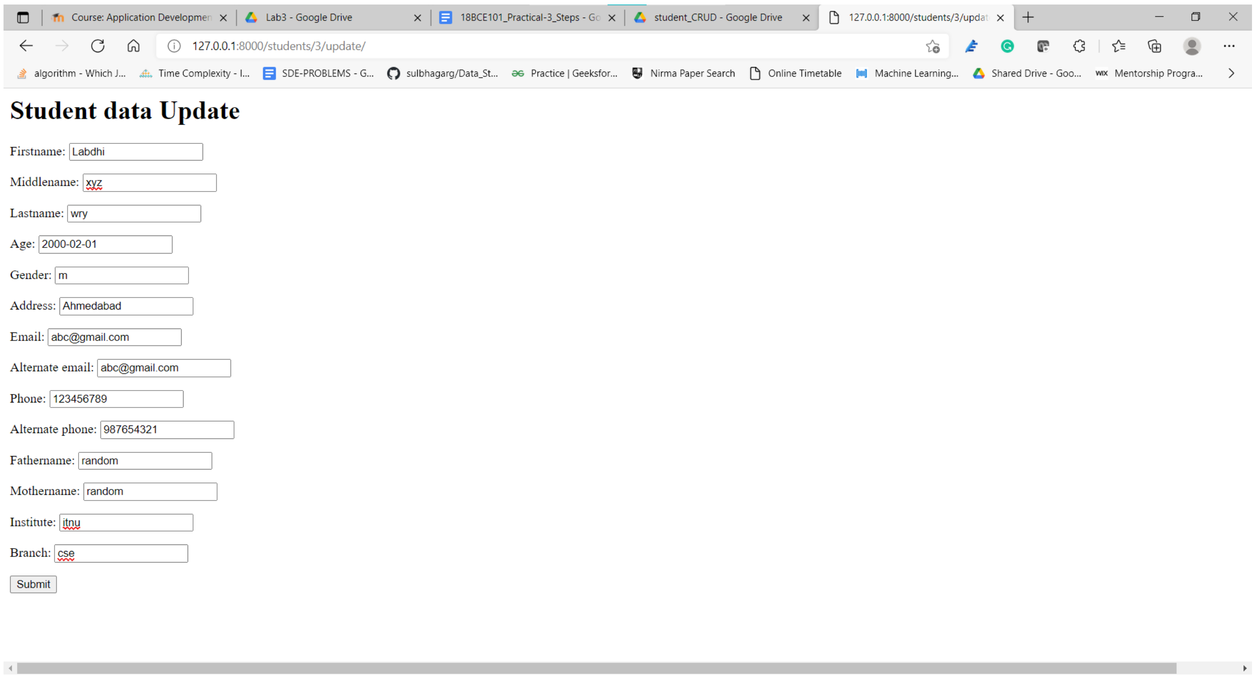
Father's Name: varij

Mother's Name: ruchi

Institute Name: itnu

Branch: cse

5. To update the records, click on update in right hand side which redirects to this page, click on “Submit”



Student data Update

Firstname: Labdhi

Middlename: xyz

Lastname: wry

Age: 2000-02-01

Gender: m

Address: Ahmedabad

Email: abc@gmail.com

Alternate email: abc@gmail.com

Phone: 123456789

Alternate phone: 987654321

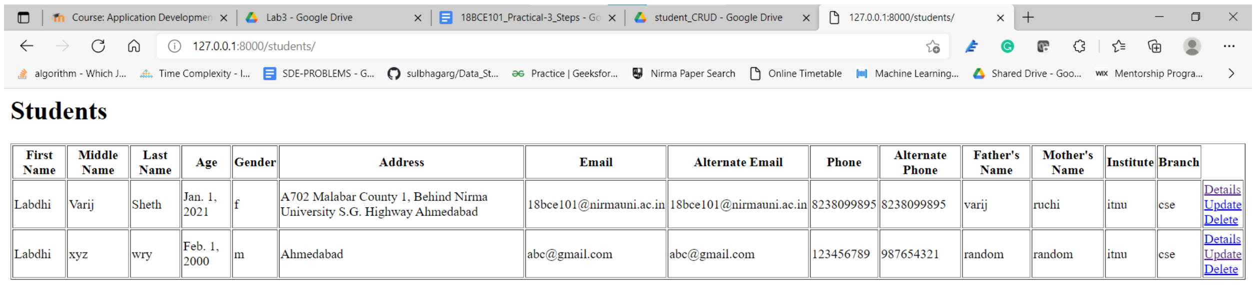
Fathername: random

Mothername: random

Institute: itnu

Branch: cse

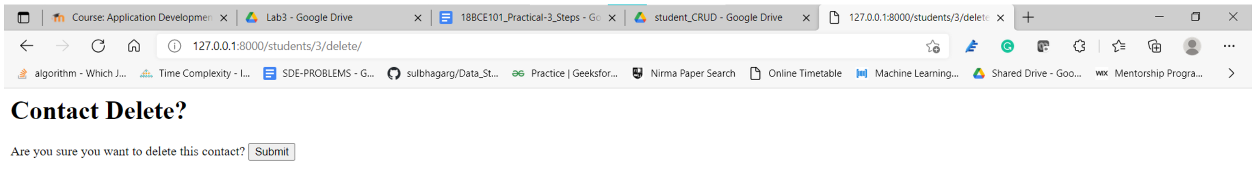
Submit



Students

First Name	Middle Name	Last Name	Age	Gender	Address	Email	Alternate Email	Phone	Alternate Phone	Father's Name	Mother's Name	Institute	Branch	
Labdhi	Varij	Sheth	Jan. 1, 2021	f	A702 Malabar County 1, Behind Nirma University S.G. Highway Ahmedabad	18bce101@nirmauni.ac.in	18bce101@nirmauni.ac.in	8238099895	8238099895	varij	ruchi	itnu	cse	Details Update Delete
Labdhi	xyz	wry	Feb. 1, 2000	m	Ahmedabad	abc@gmail.com	abc@gmail.com	123456789	987654321	random	random	itnu	cse	Details Update Delete

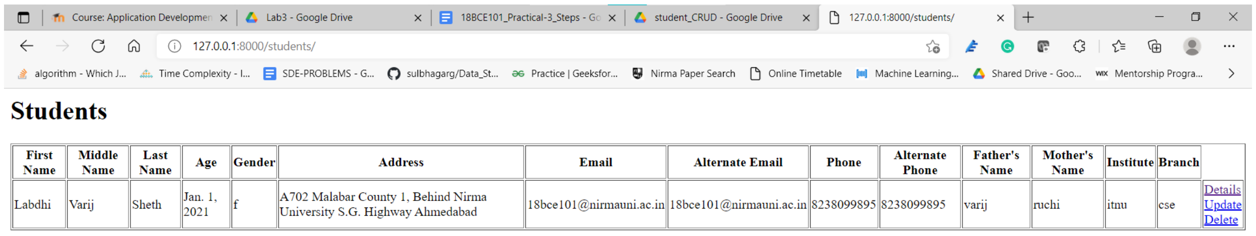
6. To delete, click on delete and the it will ask if you are sure to delete.



Contact Delete?

Are you sure you want to delete this contact? Submit

If you are sure to delete press on “Submit”



Students

First Name	Middle Name	Last Name	Age	Gender	Address	Email	Alternate Email	Phone	Alternate Phone	Father's Name	Mother's Name	Institute	Branch	
Labdhi	Varij	Sheth	Jan. 1, 2021	f	A702 Malabar County 1, Behind Nirma University S.G. Highway Ahmedabad	18bce101@nirmauni.ac.in	18bce101@nirmauni.ac.in	8238099895	8238099895	varij	ruchi	itnu	cse	Details Update Delete

Conclusion:

This practical helped us in understanding that how Djano makes it simple by writing short codes to perform CRUD operations. All the details stay in the database. We even learned the concept of redireceting the flow of the webpage after a certain operation has been completed.