Name:-

PRN: -

Class:- Ty (btech)

Batch:

PRACTICAL: 4

AIM: Design a django project for "Notes Application"

THEORY

INTRODUCTION:

In this practical we are going to build the notes application using django that will allow user to add, edit and remove the notes.

STARTING THE PROJECT

To initiate a project of Django on Your PC, open Terminal and Enter the following command

```
django-admin mysite
```

A New Folder with the name projectName will be created. To enter in the project using the terminal enter command

```
cd mysite
```

Now let's run the server and see everything is working fine or not. To run the server, type the below command in the terminal.

```
python manage.py runserver
```

Creating an app named quiz

```
python manage.py startapp blog
```

- **Settings.py:**

  Add your app name in settings.py

```
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'api.apps.ApiConfig',
    'rest_framework',
    'corsheaders',
]
```

- **Urls.py:**

  Now add your apps urls in the project

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

urlpatterns = [
    path('', getRoutes, name="routes"),
    path('notes/', getNotes, name="notes"),
    path('notes/<str:pk>/update/', updateNote, name="update-note"),
    path('notes/<str:pk>/delete/', deleteNote, name="delete-note"),
    path('notes/create/', createNote, name="create-note"),
    path('notes/<str:pk>/', getNote, name="note"),
]
```

- **Models.py:**

```
 from django.db import models

# Create your models here.

class Note(models.Model):
    body = models.TextField(null=True, blank=True)
    updated = models.DateTimeField(auto_now=True)
    created = models.DateTimeField(auto_now_add=True)
    def _str_(self):
        return self.body[0:69]
```

- **Admin.py:**

  Here, we are registering our model to the admin site, so that we can update or access the database from the admin panel also. But we need a superuser to access the admin site.

```
from django.contrib import admin
from .models import Note

# Register your models here.
```

```
admin.site.register(Note)
```

Migrate the database and create the super user

```
Python manage.py makemigrations
Python manage.py migrate
Python manage.py createsuperuser
```

- **Apps.py:**

```
from django.apps import AppConfig

class ApiConfig(AppConfig):
    default_auto_field = 'django.db.models.BigAutoField'
    name = 'api'
```

- **Views.py:**

```
from django.shortcuts import render
from rest_framework.decorators import api_view
from rest_framework.response import Response
from .serializers import NoteSerializer
from .models import Note

# Create your views here.

@api_view(['GET'])
def getRoutes(request):
    routes = [
        {
            'Endpoint': '/notes/',
            'method': 'GET',
            'body': None,
            'description': 'Returns an array of notes'
        },
        {
            'Endpoint': '/notes/id',
            'method': 'GET',
            'body': None,
            'description': 'Returns a single note object'
        },
        {
            'Endpoint': '/notes/create/',
            'method': 'POST',
            'body': {'body': ""},
            'description': 'Creates new note with data sent in post request'
        },
        {
            'Endpoint': '/notes/id/update/',
            'method': 'PUT',
            'body': {'body': ""},
            'description': 'Creates an existing note with data sent in post request'
        },
        {
            'Endpoint': '/notes/id/delete/',
            'method': 'DELETE',
            'body': None,
            'description': 'Deletes and exiting note'
        },
    ]
    return Response(routes)

@api_view(['GET'])
def getNotes(request):
    notes  =  Note.objects.all().order_by('-created')
    serializer = NoteSerializer(notes, many=True)
    return Response(serializer.data)

@api_view(['GET'])
def getNote(request, pk):
    note = Note.objects.get(id=pk)
    serializer = NoteSerializer(note, many=False)
    return Response(serializer.data)

@api_view(['PUT'])
def updateNote(request, pk):
    note = Note.objects.get(id=pk)
    serializer = NoteSerializer(instance=note, data=request.data)
    if serializer.is_valid():
        serializer.save()
    return Response(serializer.data)

@api_view(['DELETE'])
def deleteNote(request, pk):
    note = Note.objects.get(id=pk)
    note.delete()
    return Response('Note was deleted!')

@api_view(['POST'])
def createNote(request):
    data = request.data
    note = Note.objects.create(
        body=data['body']
    )
    serializer = NoteSerializer(note, many=False)
    return Response(serializer.data)
```

- **Urls.py:**
  Add the urls.py file in your app folder

```
from django.contrib import admin
from django.urls import path, include
from django.views.generic import TemplateView

urlpatterns = [
    path('admin/', admin.site.urls),
    path('api/', include('api.urls')),
    path('', TemplateView.as_view(template_name='index.html')),
]
```

- **index.html:**
  Create a index.html in templates folder.

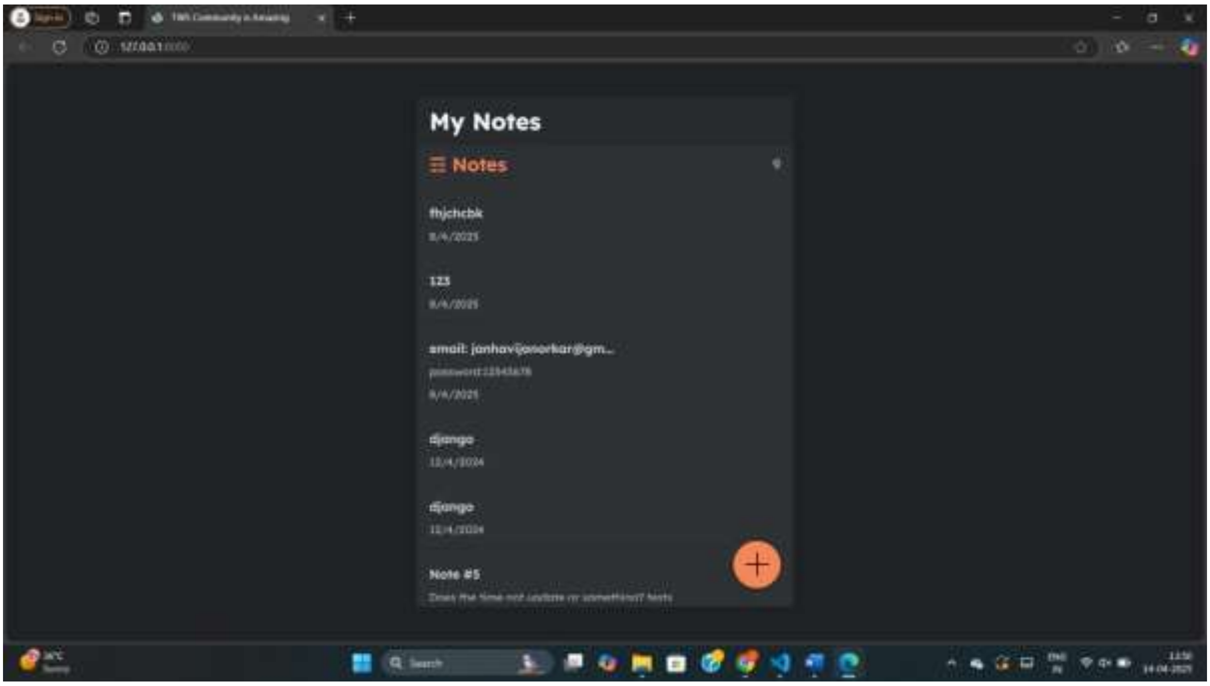
```
<!doctype html>
<html lang="en">
    <head>
        <meta charset="utf-8"/>
        <link rel="icon" href="/favicon.ico"/>
        <meta name="viewport" content="width=device-width,initial-scale=1"/>
        <meta name="theme-color" content="#000000"/>
        <meta name="description" content="Web site created using create-react-app"/>
        <link rel="apple-touch-icon" href="/logo192.png"/>
        <link rel="manifest" href="/manifest.json"/>
        <title>TWS Community is Amazing</title>
        <script defer="defer" src="/static/js/main.08442c14.js">

        </script>
        <link href="/static/css/main.e7772a38.css" rel="stylesheet">
    </head>
    <body>
        <noscript>You need to enable JavaScript to run this app.</noscript>
        <div id="root">

         </div>
     </body>
    </html>
```

OUTPUT:

Home Page



127.0.0.1:8000

My Notes

☰ Notes

fhjchcbk
8/4/2025

123
8/4/2025

email: janhavijanorkar@gm...
password:12345678
8/4/2025

django
12/4/2024

django
12/4/2024

Note #5
Does the time not update or something? tests

CONCLUSION:

Hence, in this practical we learnt to make notes using django

Sign of Teacher