

# PYTHON PROJECT

## DJANGO STUDY ROOM

DONE BY:

T. GOKULNATH

# **INDEX**

- **ABOUT PYTHON**
- **ABOUT DJANGO**
- **PROJECT DESCRIPTION**
- **SOFTWARE SPECIFICATION**
- **SOURCE CODE**
- **SAMPLE OUTPUT**
- **CONCLUSION**
- **BIBLIOGRAPHY**

## ABOUT PYTHON: -

Python programming language was developed by Guido Van Rossum in February 1991. Python is based on or influenced with two programming languages: ABC language, a teaching language created as a replacement of BASIC, and Modula-3. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace.

## ABOUT DJANGO: -

Django is a powerful and versatile web framework that enables rapid development of secure and maintainable websites. It includes an ORM (Object-Relational Mapping) system for database interactions, a robust URL routing system, and a templating engine for rendering dynamic content.

### Installation:

To get started with Django, you need to install it. You can use the following pip command:

```
pip install Django
```

## Creating a Django Project:

After installation, you can create a new Django project using the following command:

```
django-admin startproject projectname
```

This command will create a directory with the project structure

## Defining Models:

Django models define the structure of your database tables. You can create models by defining classes in the 'models.py' file of your app. For example:

```
from django.db import models

class MyModel(models.Model):
    field1 = models.CharField(max_length=100)
    field2 = models.IntegerField()
```

## Creating Migrations:

After defining models, you need to create migrations to apply changes to the database. Run the following commands:

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

## Creating Views:

Views in Django handle the logic of your application. They receive requests, process them, and return responses. Views are defined in the 'views.py' file. Example:

```
from django.shortcuts import render
from django.http import HttpResponse

def my_view(request):
    return HttpResponse("Hello, Django!")
```

## URL Routing:

URL patterns are defined in the `urls.py` file. This tells Django which view to call for a given URL. Example:

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

urlpatterns = [
    path('myurl/', my_view, name='my-url'),
]
```

## Templates:

Django templates allow you to generate dynamic HTML. Create a templates directory in your app and use the render function in your views to render templates.

## Static Files:

Store your CSS, JavaScript, and other static files in the static directory. Django automatically collects these files and serves them efficiently.

## Admin Interface:

Django comes with a built-in admin interface. You can customize it to manage your models easily. Register your models in the `admin.py` file.

## Middleware:

Middleware components process requests and responses globally before reaching the view. You can use middleware for tasks like authentication, security, etc.

## Authentication and Authorization:

Django provides a robust authentication system. You can easily integrate user authentication and control access to views using decorators.

# PROJECT DESCRIPTION: -

The Study Room project is a web application developed using Django, HTML, and CSS.

The project includes the following main components:

- User authentication for account creation and login.
- Subjects model to categorize study rooms.
- Books model to represent individual study materials.
- Views to handle user interactions.
- HTML templates for rendering dynamic content.
- CSS styling for a visually appealing and user-friendly interface

## Features:

- User registration and authentication.
- Display of subjects with brief descriptions.
- User-specific reading lists.
- Ability to add books to the reading list.
- Responsive design for various devices.
- Clean and organized layout.

## Project Structure:

- Study room/: Django project folder.
- Base/: Django app folder.
- models.py: Defines the Subject and Book models.
- views.py: Contains views for handling user interactions.
- urls.py: Defines URL patterns for the app.
- templates/: Folder for HTML templates.
- static/: Folder for CSS and other static files.

## **User Authentication:**

The project utilizes Django's built-in authentication system for user registration and login.

- Users can create accounts or log in to personalize their study experience.
- Django's User model is extended to include additional fields.

## **Views:**

Home View:

- Displays a list of available subjects.
- Shows the user's reading list with the option to add books

## **Templates (HTML):**

Base Template:

- Includes the common structure (header, footer, navigation).
- Uses Django template tags for dynamic content.

Home Template:

- Displays the list of subjects.
- Shows the user's reading list with options to add books.

## **CSS Styling:**

- Responsive design for various devices.
- Calm and conducive colour scheme.
- Readable fonts for a comfortable reading experience.
- Clean and organized layout for easy navigation.

# **SOFTWARE SPECIFICATION: -**

Operating system: Windows 11,

Platform: Visual studio code, Python idle 3.12 64 bits

Language: Python Django, HTML, CSS

# SOURCE CODE: -

views.py:

```
1 from django.shortcuts import render, redirect
2 from django.http import HttpResponseRedirect
3 from django.contrib import messages
4 from django.contrib.auth.decorators import login_required
5 from django.db.models import Q
6 from django.contrib.auth import authenticate, login, logout
7 from .models import Room, Topic, Message, User
8 from .forms import RoomForm, UserForm, MyUserCreationForm
9
10
11
12
13 def loginPage(request):
14     page = 'login'
15     if request.user.is_authenticated:
16         return redirect('home')
17
18     if request.method == 'POST':
19         email = request.POST.get('email').lower()
20         password = request.POST.get('password')
21
22         try:
23             user = User.objects.get(email=email)
24         except:
25             messages.error(request, 'User does not exist')
26
27         user = authenticate(request, email=email, password=password)
```



```
29         if user is not None:
30             login(request, user)
31             return redirect('home')
32         else:
33             messages.error(request, 'Username OR password does not exit')
34
35     context = {'page': page}
36     return render(request, 'base/login_register.html', context)
37
38
39 def logoutUser(request):
40     logout(request)
41     return redirect('home')
42
43
44 def registerPage(request):
45     form = MyUserCreationForm()
46
47     if request.method == 'POST':
48         form = MyUserCreationForm(request.POST)
49         if form.is_valid():
50             user = form.save(commit=False)
51             user.username = user.username.lower()
52             user.save()
53             login(request, user)
54             return redirect('home')
```

```

55         else:
56             messages.error(request, 'An error occurred during registration')
57
58     return render(request, 'base/login_register.html', {'form': form})
59
60
61 def home(request):
62     q = request.GET.get('q') if request.GET.get('q') != None else ''
63
64     rooms = Room.objects.filter(
65         Q(topic__name__icontains=q) |
66         Q(name__icontains=q) |
67         Q(description__icontains=q)
68     )
69
70     topics = Topic.objects.all()[0:5]
71     room_count = rooms.count()
72     room_messages = Message.objects.filter(
73         Q(room__topic__name__icontains=q))[0:3]
74
75     context = {'rooms': rooms, 'topics': topics,
76               'room_count': room_count, 'room_messages': room_messages}
77     return render(request, 'base/home.html', context)
78
79
80 def room(request, pk):
81     room = Room.objects.get(id=pk)
82     room_messages = room.message_set.all()
83     participants = room.participants.all()
84
85     if request.method == 'POST':
86         message = Message.objects.create(
87             user=request.user,
88             room=room,
89             body=request.POST.get('body')
90         )
91         room.participants.add(request.user)
92         return redirect('room', pk=room.id)
93
94     context = {'room': room, 'room_messages': room_messages,
95               'participants': participants}
96     return render(request, 'base/room.html', context)
97
98
99 def userProfile(request, pk):
100     user = User.objects.get(id=pk)
101     rooms = user.room_set.all()
102     room_messages = user.message_set.all()
103     topics = Topic.objects.all()
104     context = {'user': user, 'rooms': rooms,
105               'room_messages': room_messages, 'topics': topics}

```

```

106     return render(request, 'base/profile.html', context)
107
108
109 @login_required(login_url='login')
110 def createRoom(request):
111     form = RoomForm()
112     topics = Topic.objects.all()
113     if request.method == 'POST':
114         topic_name = request.POST.get('topic')
115         topic, created = Topic.objects.get_or_create(name=topic_name)
116
117         Room.objects.create(
118             host=request.user,
119             topic=topic,
120             name=request.POST.get('name'),
121             description=request.POST.get('description'),
122         )
123         return redirect('home')
124
125     context = {'form': form, 'topics': topics}
126     return render(request, 'base/room_form.html', context)
127
128
129 @login_required(login_url='login')
130 def updateRoom(request, pk):
131     room = Room.objects.get(id=pk)
132     form = RoomForm(instance=room)
133     topics = Topic.objects.all()
134     if request.user != room.host:
135         return HttpResponse('Your are not allowed here!!')
136
137     if request.method == 'POST':
138         topic_name = request.POST.get('topic')
139         topic, created = Topic.objects.get_or_create(name=topic_name)
140         room.name = request.POST.get('name')
141         room.topic = topic
142         room.description = request.POST.get('description')
143         room.save()
144         return redirect('home')
145
146     context = {'form': form, 'topics': topics, 'room': room}
147     return render(request, 'base/room_form.html', context)
148
149
150 @login_required(login_url='login')
151 def deleteRoom(request, pk):
152     room = Room.objects.get(id=pk)
153
154     if request.user != room.host:
155         return HttpResponse('Your are not allowed here!!')

```

```

157     if request.method == 'POST':
158         room.delete()
159         return redirect('home')
160     return render(request, 'base/delete.html', {'obj': room})
161
162
163 @login_required(login_url='login')
164 def deleteMessage(request, pk):
165     message = Message.objects.get(id=pk)
166
167     if request.user != message.user:
168         return HttpResponse('Your are not allowed here!!')
169
170     if request.method == 'POST':
171         message.delete()
172         return redirect('home')
173     return render(request, 'base/delete.html', {'obj': message})
174
175
176 @login_required(login_url='login')
177 def updateUser(request):
178     user = request.user
179     form = UserForm(instance=user)
180
181     if request.method == 'POST':
182         form = UserForm(request.POST, request.FILES, instance=user)
183         if form.is_valid():
184             form.save()
185             return redirect('user-profile', pk=user.id)
186
187     return render(request, 'base/update-user.html', {'form': form})
188
189
190 def topicsPage(request):
191     q = request.GET.get('q') if request.GET.get('q') != None else ''
192     topics = Topic.objects.filter(name__icontains=q)
193     return render(request, 'base/topics.html', {'topics': topics})

```



## Form.py:

```
1 from django.forms import ModelForm
2 from django.contrib.auth.forms import UserCreationForm
3 from .models import Room, User
4
5
6 class MyUserCreationForm(UserCreationForm):
7     class Meta:
8         model = User
9         fields = ['name', 'username', 'email', 'password1', 'password2']
10
11
12 class RoomForm(ModelForm):
13     class Meta:
14         model = Room
15         fields = '__all__'
16         exclude = ['host', 'participants']
17
18
19 class UserForm(ModelForm):
20     class Meta:
21         model = User
22         fields = ['avatar', 'name', 'username', 'email', 'bio']
```

## Models.py:

```
1 from django.db import models
2 from django.contrib.auth.models import AbstractUser
3
4
5 class User(AbstractUser):
6     name = models.CharField(max_length=200, null=True)
7     email = models.EmailField(unique=True, null=True)
8     bio = models.TextField(null=True)
9
10     avatar = models.ImageField(null=True, default="avatar.svg")
11
12     USERNAME_FIELD = 'email'
13     REQUIRED_FIELDS = []
14
15
16 class Topic(models.Model):
17     name = models.CharField(max_length=200)
18
19     def __str__(self):
20         return self.name
```

```

23 class Room(models.Model):
24     host = models.ForeignKey(User, on_delete=models.SET_NULL, null=True)
25     topic = models.ForeignKey(Topic, on_delete=models.SET_NULL, null=True)
26     name = models.CharField(max_length=200)
27     description = models.TextField(null=True, blank=True)
28     participants = models.ManyToManyField(
29         User, related_name='participants', blank=True)
30     updated = models.DateTimeField(auto_now=True)
31     created = models.DateTimeField(auto_now_add=True)
32
33     class Meta:
34         ordering = ['-updated', '-created']
35
36     def __str__(self):
37         return self.name
38
39
40 class Message(models.Model):
41     user = models.ForeignKey(User, on_delete=models.CASCADE)
42     room = models.ForeignKey(Room, on_delete=models.CASCADE)
43     body = models.TextField()
44     updated = models.DateTimeField(auto_now=True)
45     created = models.DateTimeField(auto_now_add=True)
46
47     class Meta:
48         ordering = ['-updated', '-created']
49
50     def __str__(self):
51         return self.body[0:50]

```

## Urls.py:

```

1  from django.urls import path
2  from . import views
3
4  urlpatterns = [
5      path('login/', views.loginPage, name="login"),
6      path('logout/', views.logoutUser, name="logout"),
7      path('register/', views.registerPage, name="register"),
8
9      path('', views.home, name="home"),
10     path('room/<str:pk>', views.room, name="room"),
11     path('profile/<str:pk>', views.userProfile, name="user-profile"),
12
13     path('create-room/', views.createRoom, name="create-room"),
14     path('update-room/<str:pk>', views.updateRoom, name="update-room"),
15     path('delete-room/<str:pk>', views.deleteRoom, name="delete-room"),
16     path('delete-message/<str:pk>', views.deleteMessage, name="delete-message"),
17
18     path('update-user/', views.updateUser, name="update-user"),
19
20     path('topics/', views.topicsPage, name="topics"),
21 ]

```

## Admin.py:

```
1  from django.contrib import admin
2
3  # Register your models here.
4
5  from .models import Room, Topic, Message, User
6
7  admin.site.register(User)
8  admin.site.register(Room)
9  admin.site.register(Topic)
10 admin.site.register(Message)
```

## URL to connect the app:

```
17 from django.contrib import admin
18 from django.urls import path, include
19 from django.conf import settings
20 from django.conf.urls.static import static
21
22
23 urlpatterns = [
24     path('admin/', admin.site.urls),
25     path('', include('base.urls')),
26 ]
27
28 urlpatterns += static(settings.MEDIA_URL, document_root=settings.MEDIA_ROOT)
29
```

## Html source code:

## Login.html:

```

1  {% extends 'main.html' %}
2
3  {% block content %}
4  <main class="auth layout">
5
6      {% if page == 'login' %}
7      <div class="container">
8          <div class="layout_box">
9              <div class="layout_boxHeader">
10                 <div class="layout_boxTitle">
11                     <h3>Login</h3>
12                 </div>
13             </div>
14             <div class="layout_body">
15                 <h2 class="auth_tagline">Find your study partner</h2>
16
17                 <form class="form" action="" method="POST">
18                     {% csrf_token %}
19                     <div class="form_group form_group">
20                         <label for="room_name">Email</label>
21                         <input id="username" name="email" type="email" placeholder="e.g. dennis_ivy@email.com" />
22                     </div>
23                     <div class="form_group">
24                         <label for="password">Password</label>
25                         <input id="password" name="password" type="password"
26                             placeholder="&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;&bull;" />
27                     </div>
28
29                     <button class="btn btn--main" type="submit">
30                         <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
31                             <title>lock</title>
32                             <path
33                                 d="M27 12h-1v-2c0-5.514-4.486-10-10s-10 4.486-10 10v2h-1c-0.553 0-1 0.447-1 1v18c0 0.553 0.447 1 1 1h22c0.553 0 1 0.447
34                                 1-1v-18c0-0.553-0.447-1-1-1zM8 10c0-4.411 3.589-8 8-8s8 3.589 8 8v2h-16v-2zM26 30h-20v-16h20v16z">
35                             </path>
36                             <path
37                                 d="M15 21.694v4.306h2v-4.306c0.587-0.348 1-0.961 1-1.694 0-1.105-0.895-2-2-2s-2 0.895-2 2c0 0.732 0.413 1.345 1 1.694z">
38                             </path>
39                         </svg>
40
41                     Login
42                 </button>
43             </form>
44
45             <div class="auth_action">
46                 <p>Haven't signed up yet?</p>
47                 <a href="{% url 'register' %}" class="btn btn--link">Sign Up</a>
48             </div>
49         </div>
50     </div>
51 {% else %}
52 <div class="container">
53     <div class="layout_box">
54         <div class="layout_boxHeader">
55             <div class="layout_boxTitle">
56                 <h3>Register</h3>

```

05:3



```

57     </div>
58 </div>
59 <div class="layout_body">
60     <h2 class="auth_tagline">Find your study partner</h2>
61
62     <form class="form" action="" method="POST">
63         {% csrf_token %}
64
65         {% for field in form %}
66         <div class="form_group form_group">
67             <label for="room_name">{{field.label}}</label>
68             {{field}}
69         </div>
70         {% endfor %}
71
72
73         <button class="btn btn--main" type="submit">
74             <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
75                 <title>lock</title>
76                 <path
77                     d="M27 12h-1v-2c0-5.514-4.486-10-10-10s-10 4.486-10 10v2h-1c-0.553 0-1 0.447-1 1v18c0 0.553 0.447 1 1h22c0.553 0 1 0.447
78                     1-1v-18c0-0.553-0.447-1-1z" />
79                 </path>
80                 <path
81                     d="M15 21.694v4.306h2v-4.306c0.587-0.348 1-0.961 1-1.694 0-1.105-0.895-2-2-2s-2 0.895-2 2c0 0.732 0.413 1.345 1 1.694z" />
82                 </path>
83             </svg>
84             Register
85         </button>
86     </form>
87
88     <div class="auth_action">
89         <p>Already signed up yet?</p>
90         <a href="{% url 'login' %}" class="btn btn--link">Sign Up</a>
91     </div>
92 </div>
93 </div>
94 </div>
95 {% endif %}
96 </main>
97 {% endblock content %}

```

05:3

## Home.html:

```

1  {% extends 'main.html' %}
2
3  {% block content %}
4
5  <main class="layout layout--3">
6      <div class="container">
7          <!-- Topics Start -->
8          {% include 'base/topics_component.html' %}
9
10         <!-- Topics End -->
11
12         <!-- Room List Start -->
13         <div class="roomList">
14             <div class="mobile-menu">
15                 <form action="{% url 'home' %}" method="GET" class="header_search">
16                     <label>
17                         <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
18                             <title>search</title>
19                             <path
20                                 d="M32 30.586l-10.845-10.845c1.771-2.092 2.845-4.791 2.845-7.741 0-6.617-5.383-12-12-12s-12 5.383-12 12c0 6.617 5.383 12 12
21                                 12 2.949 0 5.649-1.074 7.741-2.845" />
22                             </path>
23                         </svg>
24                         <input placeholder="Search for posts" />
25                     </label>
26                 </form>

```

```

26         <div class="mobile-menuItems">
27             <a class="btn btn--main btn--pill" href="{% url 'topics' %}">Browse Topics</a>
28         </div>
29     </div>
30     <div class="roomList_header">
31         <div>
32             <h2>Study Room</h2>
33             <p>{{room_count}} Rooms available</p>
34         </div>
35         <a class="btn btn--main" href="{% url 'create-room' %}">
36             <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
37                 <title>add</title>
38                 <path
39                     d="M16.943 0.943h-1.885v14.115h-14.115v1.885h14.115v14.115h1.885v-14.115h14.115v1.885h-14.115v-14.115z"
40                 ></path>
41             </svg>
42             Create Room
43         </a>
44     </div>
45
46     {% include 'base/feed_component.html' %}
47
48     <!-- Room List End -->
49 </div>
50 </main>
51 {% endblock %}
52

```

## Room.html:

```

1  {% extends 'main.html' %}
2
3  {% block content %}
4  <main class="profile-page layout layout--2">
5    <div class="container">
6      <!-- Room Start -->
7      <div class="room">
8        <div class="room_top">
9          <div class="room_topLeft">
10            <a href="{% url 'home' %}">
11              <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
12                <title>arrow-left</title>
13                <path
14                  d="M13.723 2.2861-13.723 13.714 13.719 13.714 1.616-1.611-10.96-10.96h27.625v-2.286h-27.625l10.965-10.965-1.616-1.607z">
15                </path>
16              </svg>
17            </a>
18            <h3>Study Room</h3>
19          </div>
20          {% if room.host == request.user %}
21          <div class="room_topRight">
22            <a href="{% url 'update-room' room.id %}">
23              <svg enable-background="new 0 0 24 24" height="32" viewBox="0 0 24 24" width="32"
24                xmlns="http://www.w3.org/2000/svg">
25                <title>edit</title>
26                <g>
27                  <path d="m23.5 22h-15c-.276 0-.5-.224-.5-.5s.224-.5-.5-.5h15c.276 0 .5.224.5.5s-.224.5-.5.5z" />
28                </g>
29                <g>
30                  <g>
31                    <path
32                      d="m2.5 22c-.131 0-.259-.052-.354-.146-.123-.123-.173-.3-.133-.468l1.09-4.625c.021-.09.067-.173.133-.239l14.143-14.143c.
33                      565-.566 1.554-.566 2.121 0l2.121 2.121c.283.283.439.66.439 1.061s-.156.778-.439 1.061l-14.142 14.141c-.065.066-.148.
34                      112-.239.133l-4.625 1.09c-.038.01-.077.014-.115.014zm1.544-4.873-.872 3.7 3.7-.872 14.042-14.041c.095-.095.146-.22.146-.
35                      354 0-.133-.052-.259-.146-.354l-2.121-2.121c-.19-.189-.518-.189-.707 0zm3.081 3.283h.01z" />
36                    </g>
37                  <g>
38                    <path
39                      d="m17.889 10.146c-.128 0-.256-.049-.354-.146l-1.461-3.535c-3.536c-.195-.195-.195-.512 0-.707s.512-.195.707 0l3.536c.195.
40                      195.195.512 0 .707-.098.098-.226.146-.354.146z" />
41                    </g>
42                  </g>
43                </g>
44              </svg>
45            </a>
46            <a href="{% url 'delete-room' room.id %}">
47              <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
48                <title>remove</title>
49                <path
50                  d="M27.314 6.019l-1.333-1.333-9.98 9.981-9.981-1.333 1.333 9.981 9.981-9.981 9.98 1.333 1.333 9.981-9.98 9.98 9.98 1.
51                  333-1.333-9.98-9.98 9.98-9.981z">
52                </path>
53              </svg>
54            </a>
55          </div>
56          {% endif %}
57        </div>
58      </div>
59    </div>
60  </main>
61  {% endblock %}

```

```

52     </div>
53     <div class="room_box scroll">
54         <div class="room_header scroll">
55             <div class="room_info">
56                 <h3>{{room.name}}</h3>
57                 <span>{{room.created|timesince}} ago</span>
58             </div>
59             <div class="room_hosted">
60                 <p>Hosted By</p>
61                 <a href="{% url 'user-profile' room.host.id %}" class="room_author">
62                     <div class="avatar avatar--small">
63                         
64                     </div>
65                     <span>@{{room.host.username}}</span>
66                 </a>
67             </div>
68
69             <span class="room_topics">{{room.topic}}</span>
70         </div>
71
72         <div class="room_conversation">
73             <div class="threads scroll">
74
75                 {% for message in room_messages %}
76                 <div class="thread">
77                     <div class="thread_top">
78                         <div class="thread_author">
79                             <a href="{% url 'user-profile' message.user.id %}" class="thread_authorInfo">
80                                 <div class="avatar avatar--small">
81                                     
82                                 </div>
83                                 <span>@{{message.user.username}}</span>
84                             </a>
85                             <span class="thread_date">{{message.created|timesince}} ago</span>
86                         </div>
87
88                         {% if request.user == message.user %}
89                         <a href="{% url 'delete-message' message.id %}">
90                             <div class="thread_delete">
91                                 <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
92                                     <title>remove</title>
93                                     <path
94                                         d="M27.314 6.0191-1.333-1.333-9.98 9.981-9.981-9.981-1.333 1.333 9.981-9.981 9.98 1.333 1.333 9.981-9.98 9.98
95                                         9.98 1.333-1.333-9.98-9.98 9.98-9.981z">
96                                     </path>
97                                 </svg>
98                             </div>
99                         </a>
100                     {% endif %}
101                 </div>
102                 <div class="thread_details">
103                     {{message.body}}
104                 </div>

```

```

106         {% endfor %}
107     </div>
108 </div>
109
110 </div>
111 <div class="room_message">
112     <form action="" method="POST">
113         {% csrf_token %}
114         <input name="body" placeholder="Write your message here..." />
115     </form>
116 </div>
117 </div>
118 <!-- Room End -->
119
120 <!-- Start -->
121 <div class="participants">
122     <h3 class="participants_top">Participants <span>({{participants.count}} Joined)</span></h3>
123     <div class="participants_list scroll">
124         {% for user in participants %}
125         <a href="{% url 'user-profile' user.id %}" class="participant">
126             <div class="avatar avatar--medium">
127                 
128             </div>
129             <p>
130                 {{user.name}}
131                 <span>@{{user.username}}</span>
129             <p>
130                 {{user.name}}
131                 <span>@{{user.username}}</span>
132             </p>
133         </a>
134         {% endfor %}
135     </div>
136 </div>
137 <!-- End -->
138 </div>
139 </main>
140 <script src="script.js"></script>
141 {% endblock content %}

```

## Room\_form.html:

```

1  {% extends 'main.html' %}
2
3  {% block content %}
4  <main class="create-room layout">
5      <div class="container">
6          <div class="layout_box">
7              <div class="layout_boxHeader">
8                  <div class="layout_boxTitle">
9                      <a href="{{request.META.HTTP_REFERER}}">
10                         <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
11                             <title>arrow-left</title>
12                             <path
13                                 d="M13.723 2.2861-13.723 13.714 13.719 13.714 1.616-1.611-10.96-10.96h27.625v-2.286h-27.625l10.965-10.965-1.616-1.607z">
14                             </path>
15                         </svg>
16                     </a>
17                     <h3>Create/Update Study Room</h3>
18                 </div>
19             </div>
20             <div class="layout_body">
21                 <form class="form" action="" method="POST">
22                     {% csrf_token %}
23
24                     <div class="form_group">
25                         <label for="room_topic">Enter a Topic</label>
26                         <input required type="text" value="{{room.topic.name}}" name="topic" list="topic-list" />
27                         <datalist id="topic-list">
28                             <select id="room_topic">

```



```

29         {% for topic in topics %}
30         <option value="{{topic.name}}">{{topic.name}}</option>
31         {% endfor %}
32     </select>
33 </datalist>
34 </div>
35
36
37 <div class="form_group">
38     <label for="room_name">Room Name</label>
39     {{form.name}}
40 </div>
41
42 <div class="form_group">
43     <label for="room_description">Room Description</label>
44     {{form.description}}
45 </div>
46
47
48
49
50 <div class="form_action">
51     <a class="btn btn--dark" href="{{request.META.HTTP_REFERER}}">Cancel</a>
52     <button class="btn btn--main" type="submit">Submit</button>
53 </div>
54 </form>
55 </div>
56 </div>
57 </div>

```

## Topics.html:

```

1  {% extends 'main.html' %}
2
3  {% block content %}
4  <main class="create-room layout">
5    <div class="container">
6      <div class="layout_box">
7        <div class="layout_boxHeader">
8          <div class="layout_boxTitle">
9            <a href="{% url 'home' %}">
10              <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
11                <title>arrow-left</title>
12                <path
13                  d="M13.723 2.2861-13.723 13.714 13.719 13.714 1.616-1.611-10.96-10.96h27.625v-2.286h-27.625l10.965-10.965-1.616-1.607z">
14                </path>
15              </svg>
16            </a>
17            <h3>Browse Topics</h3>
18          </div>
19        </div>
20
21      <div class="topics-page layout_body">
22        <form action="" method="GET" class="header_search">
23          <label>
24            <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
25              <title>search</title>

```


```


22     <form action="" method="GET" class="header__search">
23         <label>
24             <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
25                 <title>search</title>
26                 <path
27                     d="M32 30.5861-10.845-10.845c1.771-2.092 2.845-4.791 2.845-7.741 0-6.617-5.383-12-12-12 5.383-12 12c0 6.617 5.383 12 12
12 2.949 0 5.649-1.074 7.741-2.845l10.845 10.845 1.414-1.414zM12 22c-5.514 0-10-4.486-10-10s4.486-10 10-10c5.514 0 10 4.486
10 10s-4.486 10-10 10z">
28                 </path>
29             </svg>
30             <input name="q" placeholder="Search for topics" />
31         </label>
32     </form>
33
34     <ul class="topics__list">
35         <li>
36             <a href="{% url 'topics' %}" class="active">All <span>{{topics.count}}</span></a>
37         </li>
38         {% for topic in topics %}
39         <li>
40             <a href="{% url 'home' %}?q={{topic.name}}">{{ topic.name }} <span>{{topic.room_set.all.count}}</span></a>
41         </li>
42         {% endfor %}
43     </ul>
44 </div>
45 </div>
46 </div>
47 </div>
48 </main>
1 <div class="topics">
2     <div class="topics_header">
3         <h2>Browse Topics</h2>
4     </div>
5     <ul class="topics__list">
6         <li>
7             <a href="{% url 'home' %}" class="active">All <span>{{topics.count}}</span></a>
8         </li>
9         {% for topic in topics %}
10        <li>
11            <a href="{% url 'home' %}?q={{topic.name}}">{{topic.name}} <span>{{topic.room_set.all.count}}</span></a>
12        </li>
13        {% endfor %}
14    </ul>
15    <a class="btn btn--link" href="{% url 'topics' %}">
16        More
17        <svg version="1.1" xmlns="http://www.w3.org/2000/svg" width="32" height="32" viewBox="0 0 32 32">
18            <title>chevron-down</title>
19            <path d="M16 21l-13-13h-3l16 16-16-31z"></path>
20        </svg>
21    </a>
22 </div>
23 </div>

```

05:45

## SAMPLE OUTPUT:


**Freedom Concept**


**gokulnath**  
 @gokulnath

BROWSE TOPICS

All 1

PYTHON 0

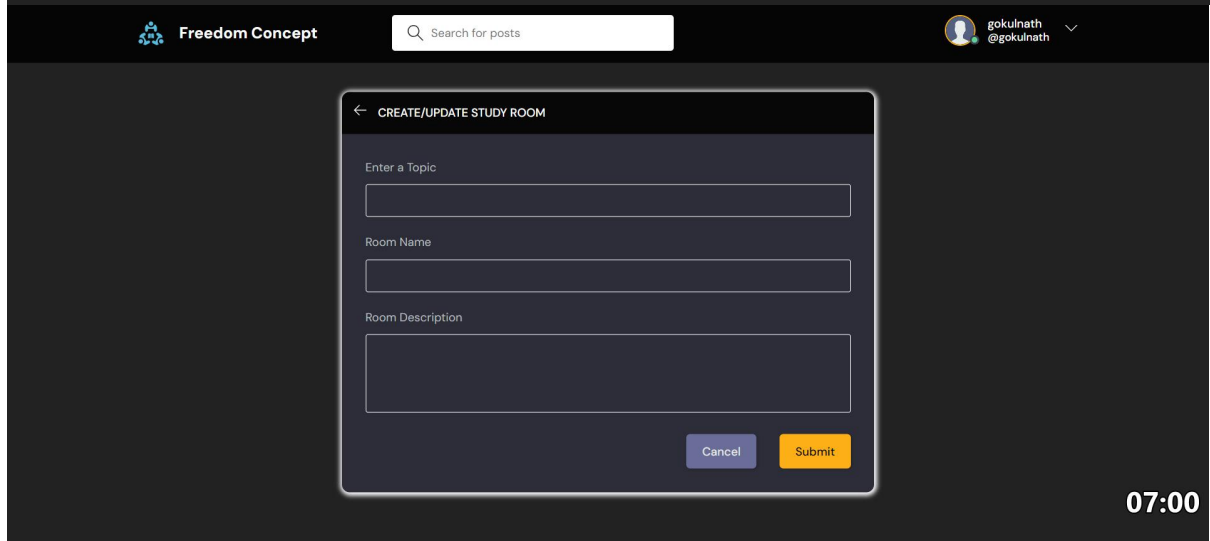
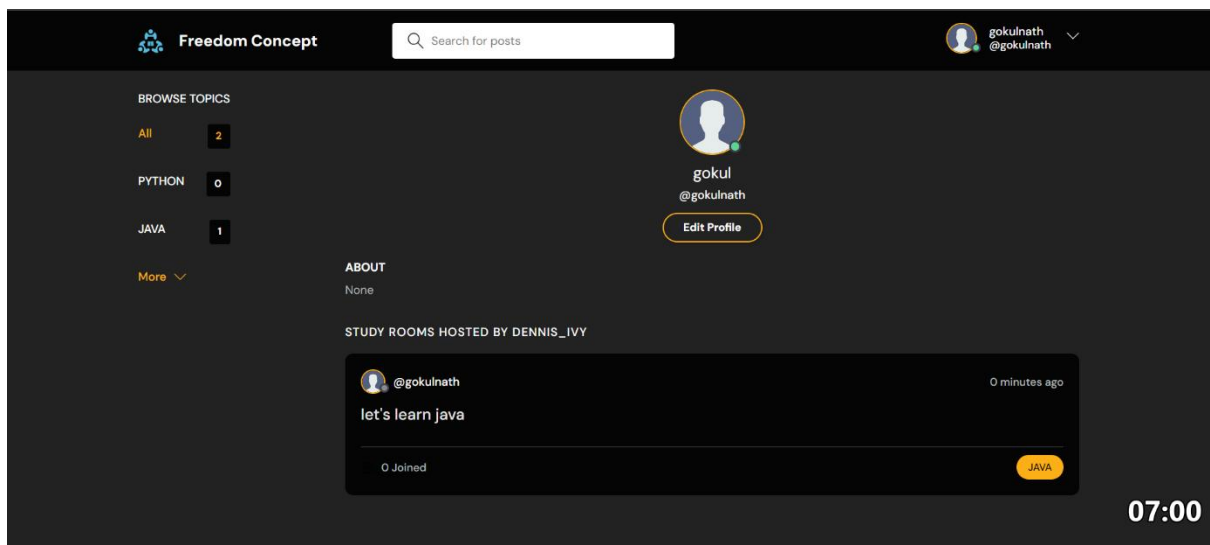
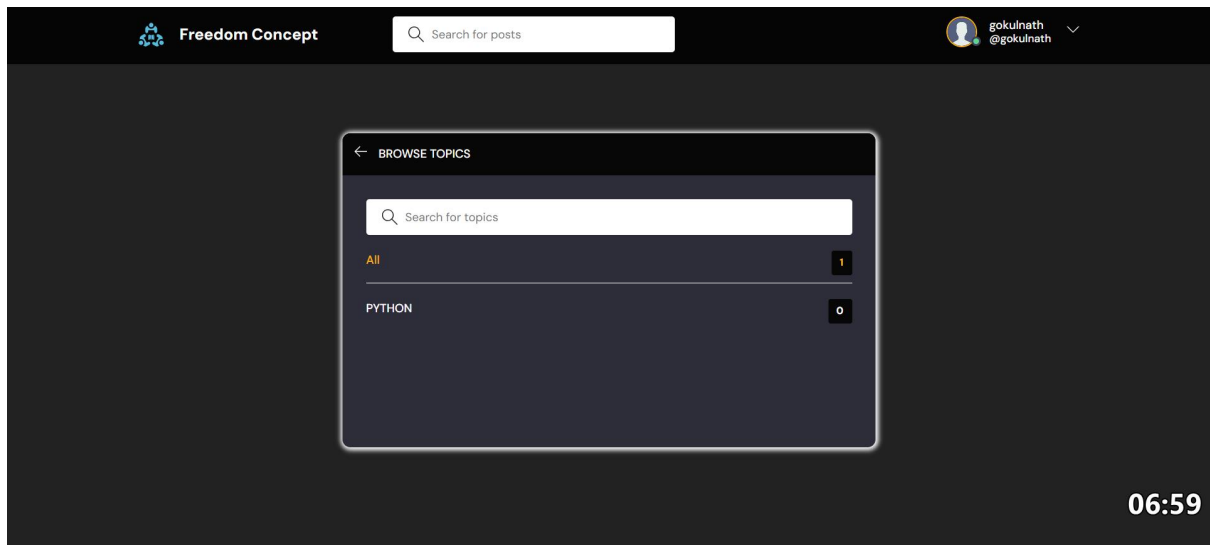
More ▾

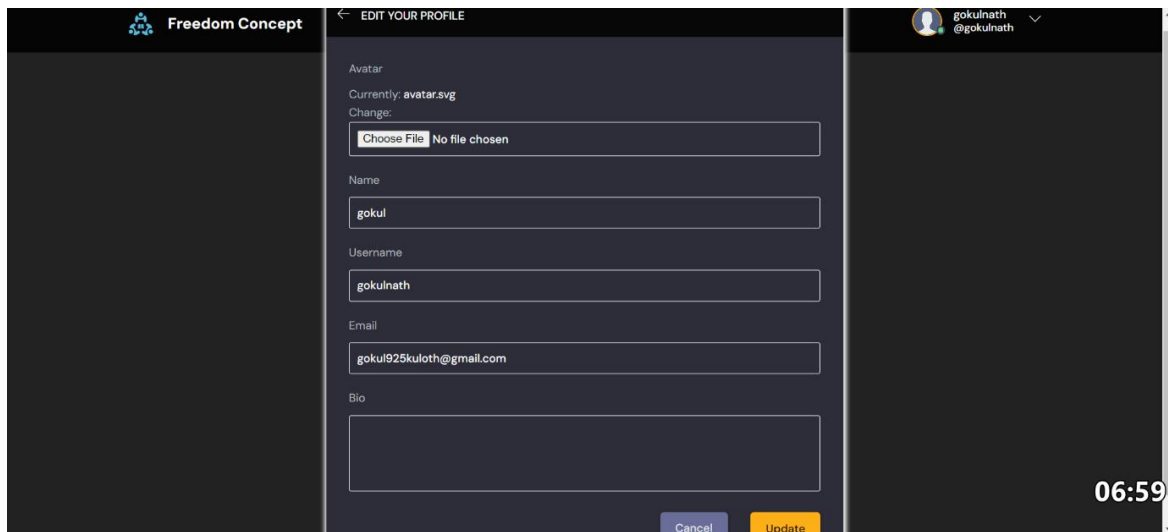
STUDY ROOM

0 Rooms available

+ Create Room

06:59





## CONCLUSION:

The Study Room project demonstrates the effectiveness of Django in creating a versatile and feature-rich web application. Through thoughtful design and careful consideration of user needs, the project provides a valuable tool for individuals seeking an organized and efficient study environment. The development process adhered to best practices, and the documentation serves as a valuable resource for both users and future developers.

As the project continues to evolve, incorporating user feedback and introducing new features will contribute to its ongoing success. The Study Room project stands as a testament to the capabilities of web development with Django, offering a scalable foundation for educational platforms and study management applications.

## BIBLIOGRAPHY:

- Traversy media django-YouTube



- <https://www.djangoproject.com>
- Bro code-YouTube