**Web Programming Lab – 10**

220962340 CSE-(AI-ML) 55 MEDHA CHAWLA

**Lab Exercises**

Q1. There are three tables in the database an author table has a first name, a last name and

an email address. A publisher table has a name, a street address, a city, a state/

province, a country, and a Web site. A book table has a title and a publication date. It

also has one or more authors (a many-to-many relationship with authors) and a single

publisher (a one-to-many relationship - aka foreign key - to publishers). Design a

form which populates and retrieves the information from the above database using

Django.

**views.py:**

from django.shortcuts import render

from .forms import BookForm

def book\_view(request):

form = BookForm(request.POST or None)

if form.is\_valid():

form.save()

return render(request, 'books/book\_form.html', {'form': form})

**books/urls.py :**

from django.urls import path

from .views import book\_view

from django.views.generic import RedirectView

from django.urls import reverse\_lazy

urlpatterns = [

path('', RedirectView.as\_view(url=reverse\_lazy('add-book'))),

path('add-book/', book\_view, name='add-book'),

]

**library\_project/urls.py :**

from django.contrib import admin

from django.urls import path

from django.urls import include

urlpatterns = [

path('', include('books.urls')),

path('admin/', admin.site.urls),

]

**forms.py:**

from django import forms

from .models import Book, Author, Publisher

class BookForm(forms.ModelForm):

class Meta:

model = Book

fields = '\_\_all\_\_'

widgets = {

'publication\_date': forms.DateInput(attrs={'type': 'date'}),

'authors': forms.SelectMultiple(attrs={'size': 5}),

'publisher': forms.Select()

}

**models.py:**

from django.db import models

class Author(models.Model):

first\_name = models.CharField(max\_length=100)

last\_name = models.CharField(max\_length=100)

email = models.EmailField()

def \_\_str\_\_(self):

return f"{self.first\_name} {self.last\_name}"

class Publisher(models.Model):

name = models.CharField(max\_length=200)

street\_address = models.CharField(max\_length=200)

city = models.CharField(max\_length=100)

state\_province = models.CharField(max\_length=100)

country = models.CharField(max\_length=100)

website = models.URLField()

def \_\_str\_\_(self):

return self.name

class Book(models.Model):

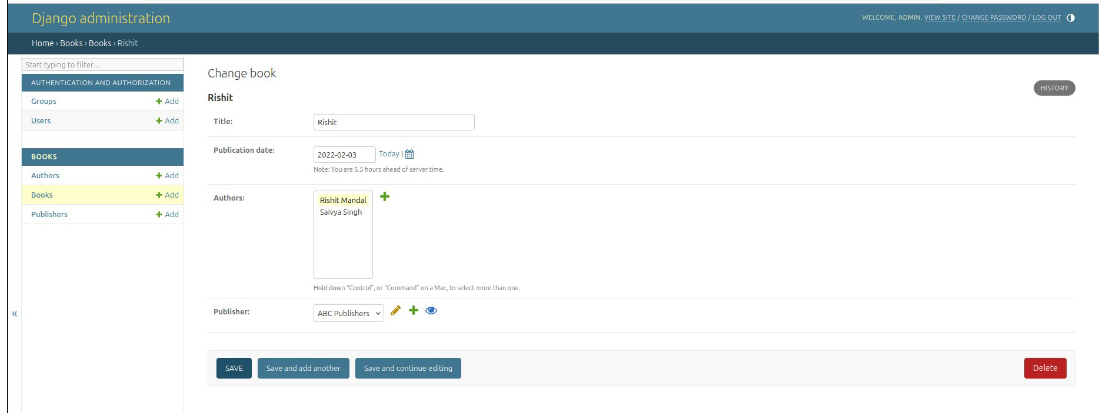
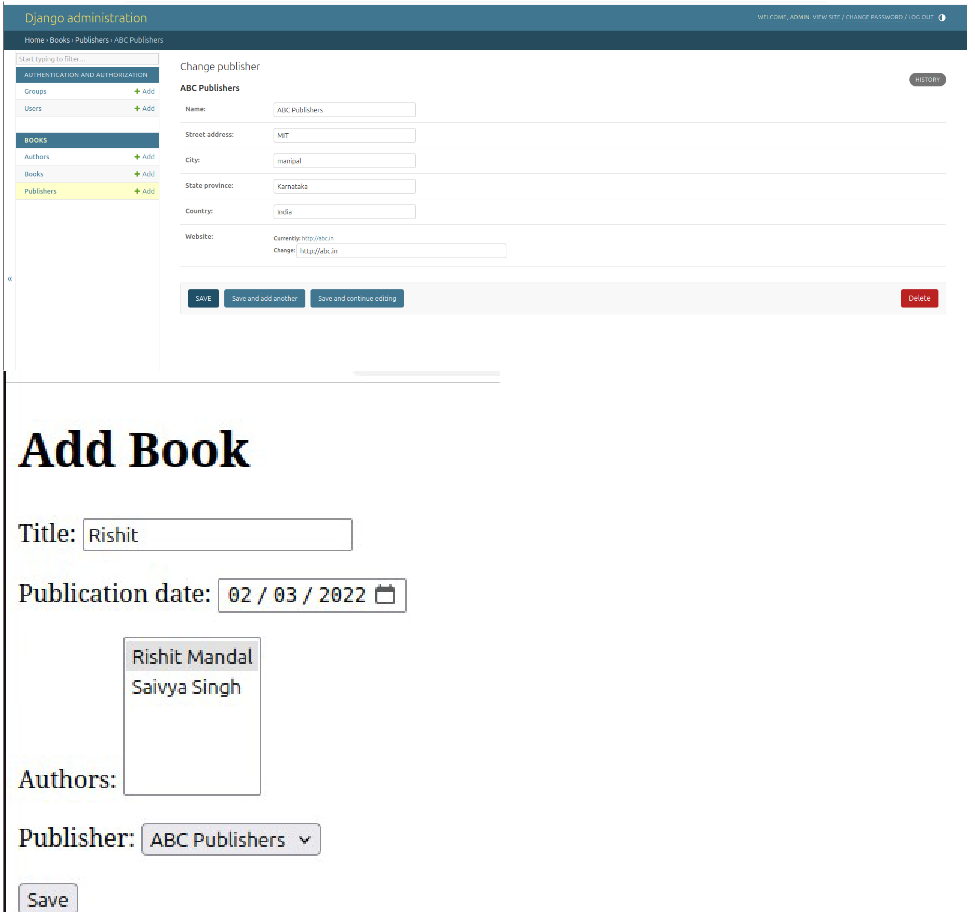
title = models.CharField(max\_length=200)

publication\_date = models.DateField()

authors = models.ManyToManyField(Author)

publisher = models.ForeignKey(Publisher, on\_delete=models.CASCADE)

def \_\_str\_\_(self):

return self.title  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
2. Create a Django Page for entry of a Product information (title, price and description)

and save it into the db. Create the index page where you would view the product

entries in an unordered list

**views.py**

from django.shortcuts import render, redirect

from .models import Product

from .forms import ProductForm

def product\_list(request):

products = Product.objects.all()

return render(request, 'product\_list.html', {'products': products})

def add\_product(request):

if request.method == 'POST':

form = ProductForm(request.POST)

if form.is\_valid():

form.save()

return redirect('product\_list')

else:

form = ProductForm()

return render(request, 'add\_product.html', {'form': form})

**urls.py**

from django.urls import path

from .views import product\_list, add\_product

urlpatterns = [

path('', product\_list, name='product\_list'),

path('add/', add\_product, name='add\_product'),

]

**product\_list.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Product List</title>

<!-- Bootstrap CSS -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"

rel="stylesheet">

</head>

<body class="bg-light">

<div class="container mt-5">

<h1 class="text-center mb-4">Products List</h1>

<ul class="list-group">

{% for product in products %}

<li class="list-group-item">

<h5 class="fw-bold">{{ product.title }}</h5>

<p class="text-muted">Price: Rs. {{ product.price }}</p>

<p>{{ product.description }}</p>

</li>

{% endfor %}

</ul>

<div class="text-center mt-4">

<a href="{% url 'add\_product' %}" class="btn btn-primary">Add New Product</a>

</div>

</div>

<!-- Bootstrap JS -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></

script>

</body>

</html>

**add\_product.html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Add Product</title>

<!-- Bootstrap CSS -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"

rel="stylesheet">

</head>

<body class="bg-light">

<div class="container mt-5">

<h1 class="text-center mb-4">Add a New Product</h1>

<div class="card shadow-sm p-4">

<form method="post">

{% csrf\_token %}

<div class="mb-3">

<label class="form-label">Title</label>

{{ form.title }}

</div>

<div class="mb-3">

<label class="form-label">Price</label>

{{ form.price }}

</div>

<div class="mb-3">

<label class="form-label">Description</label>

{{ form.description }}

</div>

<button type="submit" class="btn btn-success">Save Product</button>

</form>

</div>

<div class="text-center mt-4">

<a href="{% url 'product\_list' %}" class="btn btn-secondary">Back to Product List</a>

</div>

</div>

<!-- Bootstrap JS -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></

script>

</body>

</html>

**models.py**

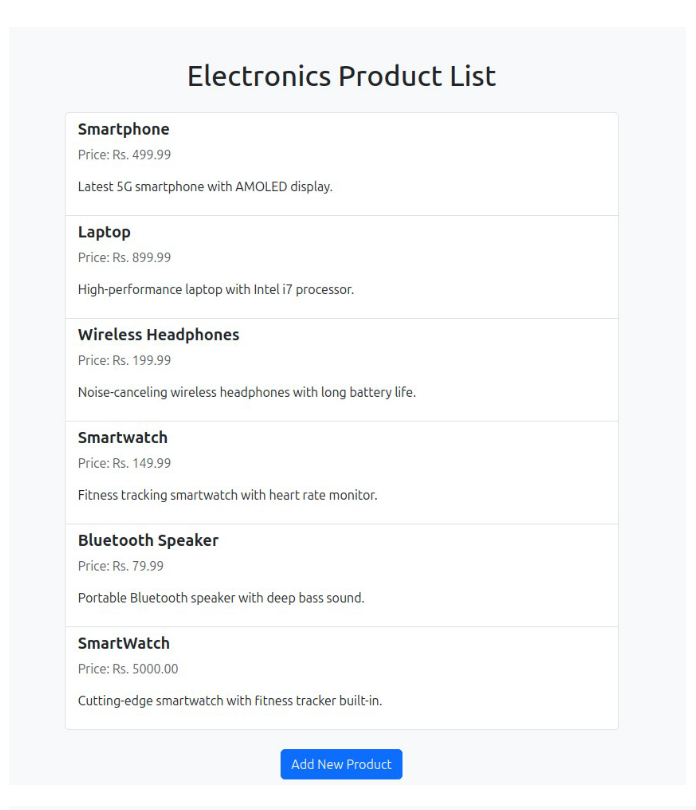
from django.db import models

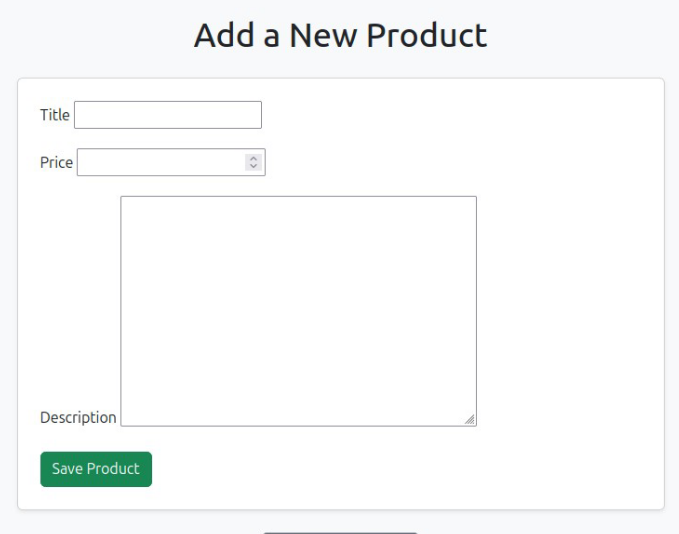
class Product(models.Model):

title = models.CharField(max\_length=200)

price = models.DecimalField(max\_digits=10, decimal\_places=2)

description = models.TextField()

def \_\_str\_\_(self):

return self.title  
  
  
  


Q3. Create a web page with DropDownList, Textboxes and Buttons. Assume the table

‘Human’ with First name, Last name, Phone, Address and City as fields.

When the page is loaded, only first names will be displayed in the drop-down list. On

selecting the name, other details will be displayed in the respective TextBoxes. On

clicking the update button, the table will be updated with new entries made in the text

box. On clicking the delete button, the selected record will be deleted from the table,

and the DropDownList is refreshed.

**views.py**

from django.shortcuts import render, get\_object\_or\_404

from django.http import JsonResponse

from .models import Human

from .forms import HumanForm

def index(request):

humans = Human.objects.all()

form = HumanForm()

return render(request, 'people/index.html', {'humans': humans, 'form': form})

def get\_human(request):

if request.method == 'GET':

id = request.GET.get('id')

human = get\_object\_or\_404(Human, id=id)

return JsonResponse({

'last\_name': human.last\_name,

'phone': human.phone,

'address': human.address,

'city': human.city,

})

def update\_human(request):

if request.method == 'POST':

id = request.POST.get('id')

human = get\_object\_or\_404(Human, id=id)

form = HumanForm(request.POST, instance=human)

if form.is\_valid():

form.save()

return JsonResponse({'status': 'updated'})

def delete\_human(request):

if request.method == 'POST':

id = request.POST.get('id')

human = get\_object\_or\_404(Human, id=id)

human.delete()

return JsonResponse({'status': 'deleted'})

**model.py**

from django.db import models

class Human(models.Model):

first\_name = models.CharField(max\_length=100)

last\_name = models.CharField(max\_length=100)

phone = models.CharField(max\_length=20)

address = models.CharField(max\_length=200)

city = models.CharField(max\_length=100)

def \_\_str\_\_(self):

return self.first\_name

**urls.py**

from django.urls import path

from . import views

urlpatterns = [

path('', views.index, name='index'),

path('get-human/', views.get\_human, name='get\_human'),

path('update-human/', views.update\_human, name='update\_human'),

path('delete-human/', views.delete\_human, name='delete\_human'),

]

**index.html**

<!DOCTYPE html>

<html>

<head>

<title>Human CRUD</title>

<script>

function loadData() {

const id = document.getElementById("human-select").value;

fetch("/get-human/?id=" + id)

.then(res => res.json())

.then(data => {

document.getElementById("id").value = id;

document.getElementById("id\_last\_name").value = data.last\_name;

document.getElementById("id\_phone").value = data.phone;

document.getElementById("id\_address").value = data.address;

document.getElementById("id\_city").value = data.city;

});

}

function updateData() {

const form = document.getElementById("human-form");

const formData = new FormData(form);

fetch("/update-human/", {

method: "POST",

headers: {'X-CSRFToken': getCookie("csrftoken")},

body: formData

})

.then(res => res.json())

.then(data => location.reload());

}

function deleteData() {

const formData = new FormData();

formData.append("id", document.getElementById("id").value);

fetch("/delete-human/", {

method: "POST",

headers: {'X-CSRFToken': getCookie("csrftoken")},

body: formData

})

.then(res => res.json())

.then(data => location.reload());

}

function getCookie(name) {

let cookieValue = null;

if (document.cookie && document.cookie !== '') {

const cookies = document.cookie.split(';');

for (let cookie of cookies) {

cookie = cookie.trim();

if (cookie.startsWith(name + '=')) {

cookieValue = decodeURIComponent(cookie.slice(name.length + 1));

break;

}

}

}

return cookieValue;

}

</script>

</head>

<body>

<h1>Human Info</h1>

<label>First Name:</label>

<select id="human-select" onchange="loadData()">

<option disabled selected>Select a person</option>

{% for h in humans %}

<option value="{{ h.id }}">{{ h.first\_name }}</option>

{% endfor %}

</select>

<form id="human-form">

<input type="hidden" name="id" id="id">

{{ form.as\_p }}

</form>

<button onclick="updateData()">Update</button>

<button onclick="deleteData()">Delete</button>

</body>

</html>

**forms.py**

from django import forms

from .models import Human

class HumanForm(forms.ModelForm):

class Meta:

model = Human

fields = '\_\_all\_\_'

