Web Programming Lab – 10

220905390 CSE-D2 46 RISHIT MANDAL

Additional Question – Lab 9

Assume a table “Institutes” with institute_id, name, and no_of_courses are the fields.

Create a web page that retrieves all the data from “Institutes” table displays only

Institute names in the list box

views.py

```
from django.shortcuts import render
from .models import Institute

def institute_list(request):
institutes = Institute.objects.all()
return render(request, 'institute_list.html', {'institutes': institutes})
```

urls.py

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

urlpatterns = [
path('', institute_list, name='institute_list'),
]
```

html

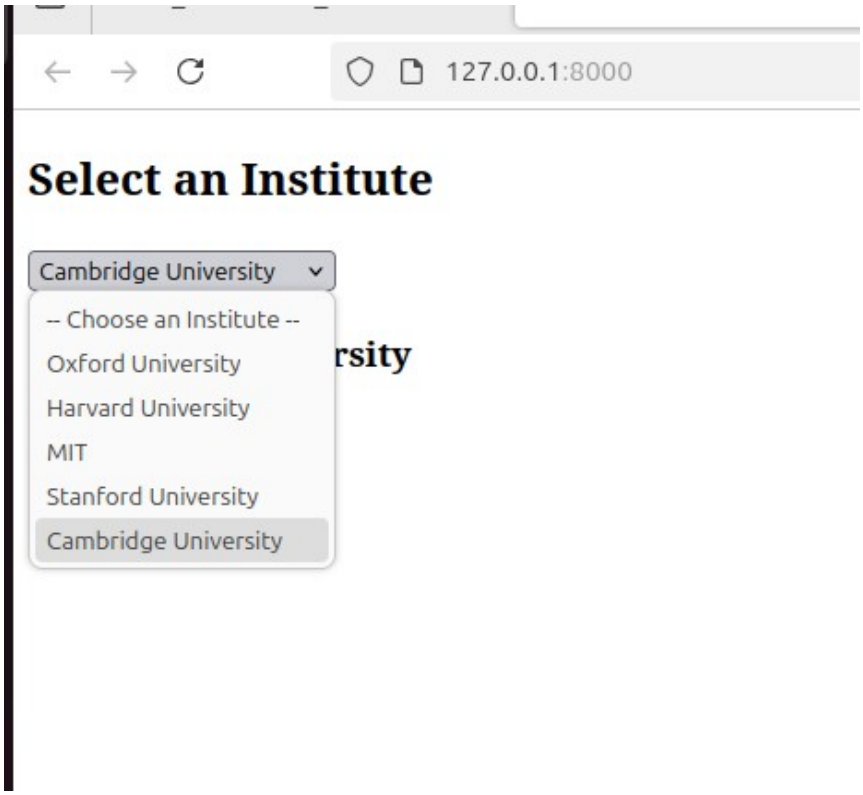
```
<!DOCTYPE html>
<html lang="en">

<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
```

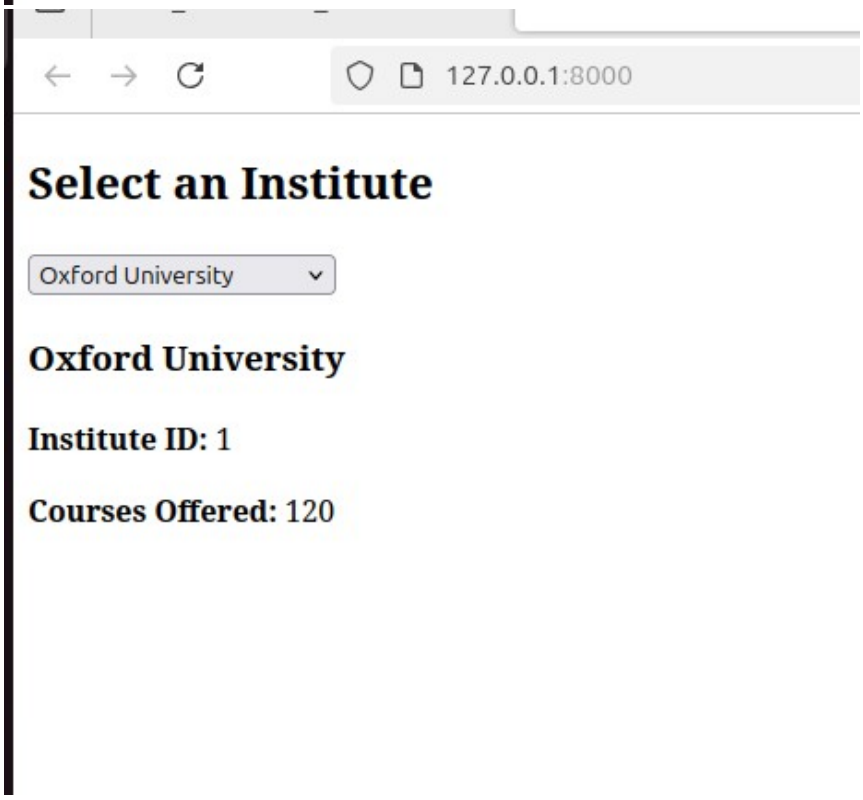
```
<title>Institutes</title>
<script>
function showDetails() {
let dropdown = document.getElementById("instituteDropdown");
let selectedIndex = dropdown.selectedIndex;
let detailsDiv = document.getElementById("details");

if (selectedIndex > 0) {
let selectedOption = dropdown.options[selectedIndex];
let id = selectedOption.getAttribute("data-id");
let name = selectedOption.text;
let courses = selectedOption.getAttribute("data-courses");

detailsDiv.innerHTML = `
<h3>${name}</h3>
<p><strong>Institute ID:</strong> ${id}</p>
<p><strong>Courses Offered:</strong> ${courses}</p>
`;
} else {
detailsDiv.innerHTML = "";
}
}
</script>
</head>

<body>
<h2>Select an Institute</h2>
<select id="instituteDropdown" onchange="showDetails()">
<option value="">-- Choose an Institute --</option>
{% for institute in institutes %}
<option data-id="{{ institute.institute_id }}" data-courses="{{ institute.no_of_courses }}">
{{ institute.name }}
</option>
{% endfor %}
</select>

<div id="details" style="margin-top: 20px;"></div>
</body>

</html>
```

models.py

```
from django.db import models
```

```
class Institute(models.Model):
institute_id = models.AutoField(primary_key=True)
name = models.CharField(max_length=255)
no_of_courses = models.IntegerField()

def __str__(self):
return self.name
```





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
```

Django administration

WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Books › Publishers › ABC Publishers

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

BOOKS

Authors + Add

Books + Add

Publishers + Add

Change publisher

HISTORY

ABC Publishers

Name: ABC Publishers

Street address: MIT

City: manipal

State province: Karnataka

Country: India

Website: Currently: http://abc.in
Change: http://abc.in

SAVE | Save and add another | Save and continue editing | Delete

Add Book

Title: Rishit

Publication date: 02 / 03 / 2022

Authors: Rishit Mandal, Saivya Singh

Publisher: ABC Publishers

Save

Django administration

WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT

Home › Books › Books › Rishit

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

BOOKS

Authors + Add

Books + Add

Publishers + Add

Change book

HISTORY

Rishit

Title: Rishit

Publication date: 2022-02-03 Today |

Note: You are 5.5 hours ahead of server time.

Authors: Rishit Mandal, Saivya Singh

Hold down "Control", or "Command" on a Mac, to select more than one.

Publisher: ABC Publishers

SAVE | Save and add another | Save and continue editing | Delete

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

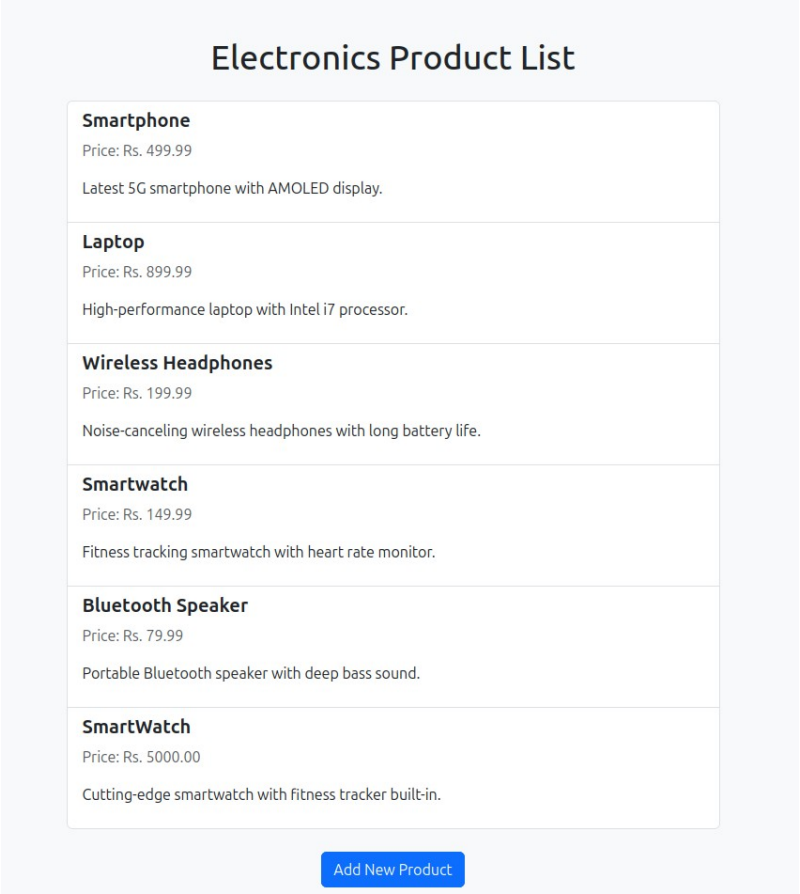
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Add Product</title>
<!-- Bootstrap CSS -->
```

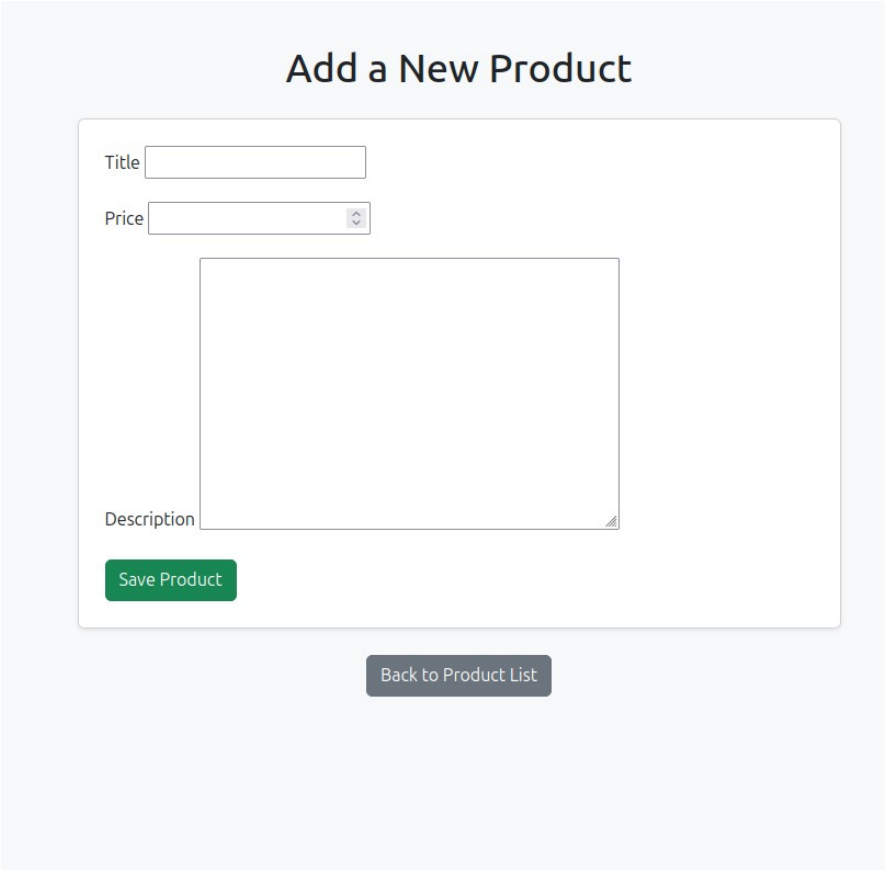
```
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
</head>
<body class="bg-light">
<div class="container mt-5">
<h1 class="text-center mb-4">Add a New Product</h1>

<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

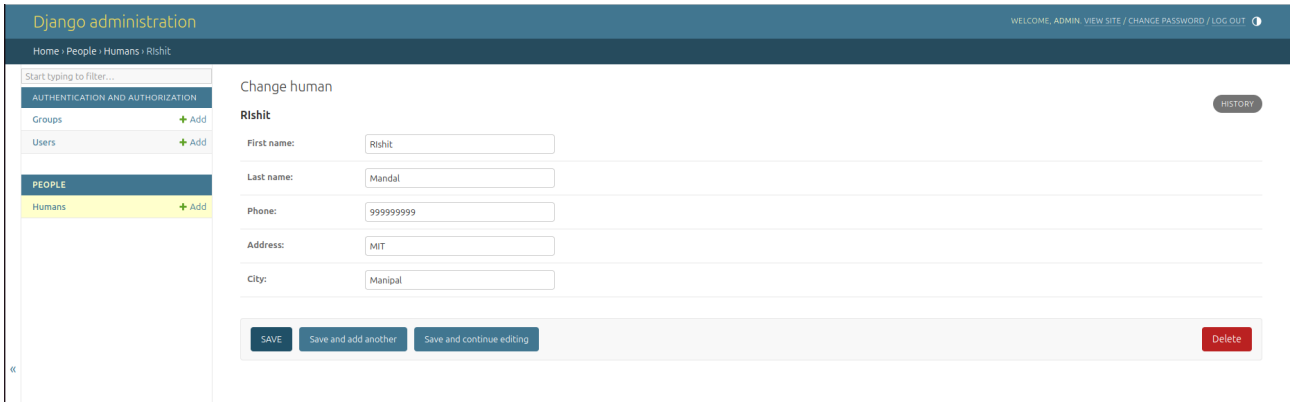
```
<!DOCTYPE html>
<html>
<head>
    <title>Human CRUD</title>
    <script>
        function loadData() {
            const id = document.getElementById("human-select").value;
            fetch("/get-human/?id=" + id)
            .then(res => res.json())
            .then(data => {
```

```
            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__'
```



Human Info

First Name: RIshit

First name: RIshit

Last name: Mandal

Phone: 9999999999

Address: MIT

City: Manipal

Update Delete

Human Info

First Name: RIshit

First name: RISHIT

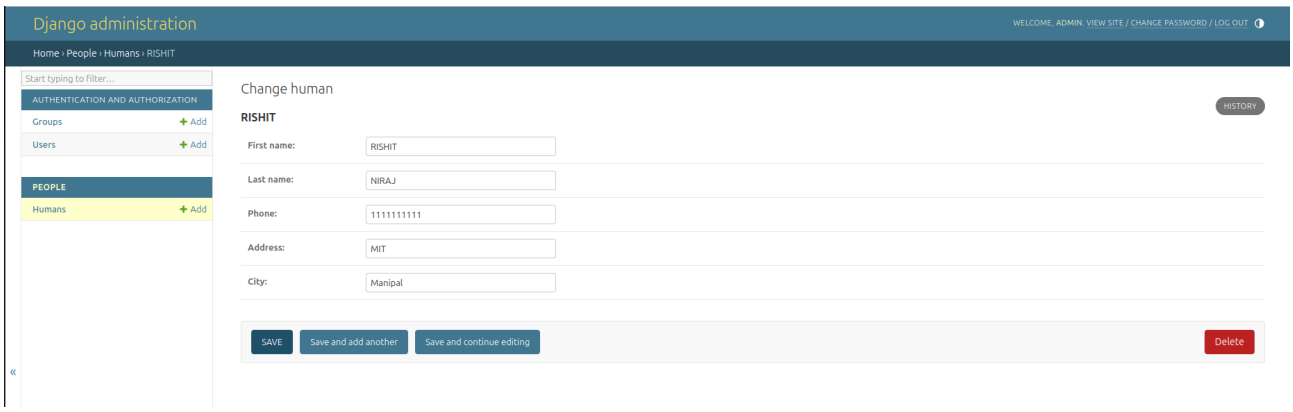
Last name: NIRAJ

Phone: 1111111111

Address: MIT

City: Manipal

Update Delete



LAB 10 – Additional Question

Create a web page that receives the following information from a set of students: Student Id, Student Name, Course Name and Date of Birth. The application should also display the information of all the students once the data is Entered.

models.py:

```
from django.db import models

class Student(models.Model):
    student_id = models.CharField(max_length=20, primary_key=True)
    name = models.CharField(max_length=100)
    course = models.CharField(max_length=100)
    dob = models.DateField()

    def __str__(self):
        return self.name
```

views.py:

```
from django.shortcuts import render, redirect
from .forms import StudentForm
from .models import Student

def student_view(request):
    if request.method == 'POST':
        form = StudentForm(request.POST)
        if form.is_valid():
            form.save()
```

```
            return redirect('student')  # Prevent form resubmission
    else:
        form = StudentForm()

    students = Student.objects.all()
    return render(request, 'student_form.html', {'form': form, 'students': students})
```

studentproject/urls.py:

```
from django.contrib import admin
from django.urls import path, include

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('studentapp.urls')),
]
```

studentapp/urls.py:

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

urlpatterns = [
    path('', views.student_view, name='student'),
]
```

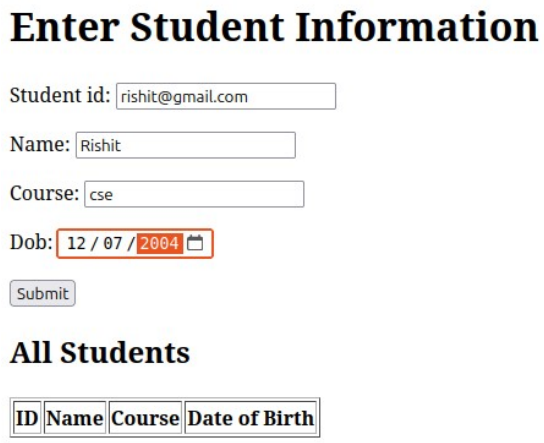
forms.py:

```
from django import forms
from .models import Student

class StudentForm(forms.ModelForm):
    class Meta:
        model = Student
        fields = ['student_id', 'name', 'course', 'dob']
        widgets = {
            'dob': forms.DateInput(attrs={'type': 'date'}),
        }
```

student_form.html:

```
<!DOCTYPE html>
<html>
<head>
    <title>Student Registration</title>
</head>
<body>
```

```
    <h1>Enter Student Information</h1>
    <form method="post">
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit">Submit</button>
    </form>

    <h2>All Students</h2>
    <table border="1">
        <tr>
            <th>ID</th><th>Name</th><th>Course</th><th>Date of Birth</th>
        </tr>
        {% for student in students %}
        <tr>
            <td>{{ student.student_id }}</td>
            <td>{{ student.name }}</td>
            <td>{{ student.course }}</td>
            <td>{{ student.dob }}</td>
        </tr>
        {% endfor %}
    </table>
</body>
</html>
```



Enter Student Information

Student id:

Name:

Course:

Dob: mm / dd / yyyy

Submit

All Students

ID	Name	Course	Date of Birth
rishit@gmail.com	Rishit	cse	Dec. 7, 2004
avadhgandhi@gmail.co	avadh	ece	Jan. 12, 2003