**PART A**

**Experiment No. 08**

**A.1 Aim:** Create Employee Management System using Django to perform (CRUD operations

**Objective:** To learn and understand how use CRUD operations in Django

**A.2 Prerequisite:** HTML, CSS, Javascript

**A.3 Outcome:**

After successful completion of these applications students will be able to understand Django Framework, Creating Views and Templates:, Dynamic Content Rendering and using css and js in Django

**A.4 Theory**

**CRUD Operations in Django**

CRUD stands for **Create**, **Read**, **Update**, and **Delete**, which are the four basic operations used to manage data in a database cane be handled by the function in views.py.

**Queries for Retrieve and Filter Employees by Salary**

**1. Retrieve Employees Ordered by Salary**

To retrieve employees ordered by their salary, you can modify the employee\_list view to sort the employees based on their salary.

**Updated employee\_list View:**

def employee\_list(request):

employees = Employee.objects.all().order\_by('salary') # Retrieves employees ordered by salary (ascending)

return render(request, 'employee\_list.html', {'employees': employees})

**SQL Equivalent Query:** SELECT \* FROM Employee ORDER BY salary ASC;

**2. Filter Employees by Salary Range**

**Updated employee\_list View with Salary Range Filter:**

def employee\_list(request):

min\_salary = request.GET.get('min\_salary', 0)

max\_salary = request.GET.get('max\_salary', 999999)

employees = Employee.objects.filter(salary\_\_gte=min\_salary, salary\_\_lte=max\_salary).order\_by('salary') # Filter by salary range

return render(request, 'employee\_list.html', {'employees': employees})

**SQL Equivalent Query:** SELECT \* FROM Employee WHERE salary >= min\_salary AND salary <= max\_salary ORDER BY salary ASC;

**Scenario Problem statement 1: Employee Management System Using Django**

You are tasked with building a basic Employee Management System using Django, where users can **Create, Read, Update, and Delete (CRUD) employee records**.

**As a given above try to include the queries in views.py along with crud operations** Your system should allow for the following operations:

1. Retrieve Employees Ordered by Salary: Implement functionality to list employees ordered by their salary.
2. Filter Employees by Salary Range: Allow users to filter employees based on a specific salary range**.**

Hints:

myproject/ # Root directory of your Django project

│

├── myproject/ # Django settings and configuration files

│ ├── \_\_init\_\_.py

│ ├── settings.py # Project settings

│ ├── urls.py # Root URL configuration

│ ├── asgi.py

│ └── wsgi.py

│

├── myapp3/ # Application directory (the app for employee management)

│ ├── \_\_init\_\_.py

│ ├── admin.py

│ ├── apps.py # Application configuration

│ ├── forms.py # Django forms for Employee model

│ ├── migrations/ # Database migrations for the app

│ │ └── \_\_init\_\_.py

│ ├── models.py # Employee model definition

│ ├── views.py # Application views (CRUD and employee-related operations)

│ ├── urls.py # URL routing for the app (employee CRUD operations)

│ └── templates/

│ ├── employee\_form.html # Template for employee creation and update

│ ├── employee\_list.html # Template to display employee list

│ └── employee\_confirm\_delete.html # Template for employee deletion confirmation

│

├── manage.py

└── db.sqlite3

**Sampe 2. URL Configuration (myapp3/urls.py)**

from django.urls import path

from . import views

urlpatterns = [

path('', views.employee\_list, name='employee\_list'), # List all employees

path('create/', views.employee\_create, name='employee\_create'), # Create a new employee

path('update/<int:id>/', views.employee\_update, name='employee\_update'), # Update an employee

path('delete/<int:id>/', views.employee\_delete, name='employee\_delete'), # Delete an employee

]

**Problem statement 2: using Django shell run the following quires**

1. How to create employees (Insert Operation)?
2. How to retrieve all employees (Select Operation)?
3. How to retrieve employees ordered by salary?
4. How to filter employees by salary range?
5. How to retrieve the first and last employee?
6. How to count the number of employees?
7. How to perform aggregation on salaries?
8. How to retrieve employees with selected fields?
9. How to search employees by name (startswith, endswith, and contains)?
10. How to check if an employee exists?
11. How to update employee salaries?
12. How to delete employee records?

**ART B**

(PART B: TO BE COMPLETED BY STUDENTS)

(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)

|  |  |
| --- | --- |
| Roll No. : | Name: |
| Class : | Batch : |
| Date of Experiment : | Date/Time of Submission : |
| Grade : |  |

**B.1 Code:**

*(Paste your Code here)*

**B.2 Output**

*(Take screen shots of the output at run time and paste it here)*

**B.3 Conclusion:**

*(Students must write the conclusion as per the attainment of individual outcome listed above)*

**B.3 Observations and Learning:**

*(Students must write their observations and learnings as per the attainment of individual outcome listed above)*

**B.4 Question of Curiosity**

*(To be answered by student based on the practical performed and learning/observations)*

1. How does Django's ORM facilitate CRUD operations, and what are some key methods used in the ORM for database interaction?
2. How do you handle form submissions in Django to create new records in a database?
3. What strategies can be used to handle errors in update and delete operations in Django CRUD?
4. How do you retrieve a specific record from the database to update it in Django?
5. How do you pass context data from a Django view to a template when rendering a list of records?
6. What steps are involved in deleting a record in Django, and how do you confirm the deletion?