

## Python + MongoDB Assessment (Django/FastAPI)

*Note:- Candidate should be able to explain the code provided and run locally*

### Section 1: Setup

Use either:

FastAPI

Django REST Framework (DRF)

**Database: MongoDB (Local)**

**Database name: assessment\_db Collection name: employees**

### Sample Document Structure

```
{  
  
  "employee_id": "E123",  
  
  "name": "John Doe",  
  
  "department": "Engineering",  
  
  "salary": 75000,  
  
  "joining_date": "2023-01-15",  
  
  "skills": ["Python", "MongoDB", "APIs"]  
}
```

### Section 2: Core CRUD APIs

#### 1. Create Employee

Endpoint: POST /employees Task:

Insert a new employee record.

Validation: Ensure employee\_id is unique.

#### 2. Get Employee by ID

Endpoint: GET /employees/{employee\_id}

Task: Fetch employee details by employee\_id.

Check: Return 404 if not found.

### **3. Update Employee**

Endpoint: PUT /employees/{employee\_id}

Task: Update employee details.

Requirement: Allow partial updates (update only provided fields).

### **4. Delete Employee**

Endpoint: DELETE /employees/{employee\_id}

Task: Delete employee record.

Response: Success/failure message.

## **Section 3: Querying & Aggregation**

### **5. List Employees by Department**

Endpoint: GET /employees?department=Engineering

Task: Return employees in a department, sorted by joining\_date (newest first).

### **6. Average Salary by Department**

Endpoint: GET /employees/avg-salary

**Task: Use MongoDB aggregation to compute average salary grouped by department.**

Expected Output:

```
[  
  {"department": "Engineering", "avg_salary": 80000},  
  {"department": "HR", "avg_salary": 60000}
```

]

## 7. Search Employees by Skill

Endpoint: GET /employees/search?skill=Python

**Task: Return employees who have the given skill in their skills array.**

## Section 4: Bonus Challenges (Optional)

- Implement pagination for employee listing.
- Add MongoDB index on employee\_id.
- Implement schema validation (e.g., using MongoDB JSON Schema).
- Add JWT authentication for protected routes.